
Indico Documentation

Release 2.0.dev0

Indico Team

Sep 21, 2017

Contents

1	Installation / Configuration	3
1.1	Installation guides	3
2	Server administration	35
2.1	Server administration	35
3	Plugins	37
3.1	Extending Indico with plugins	37
4	HTTP API	51
4.1	Indico - HTTP API	51
5	API reference	79
5.1	API reference	79
6	Indices and tables	279
	Python Module Index	281



The effortless open source tool for event organization, archival and collaboration.

Welcome to Indico's documentation. This documentation is split into several parts, from installing Indico to developing Indico plugins. To dive into the internals of Indico, check out the [API documentation](#). Read more about Indico in our [official website](#).

Installation / Configuration

To simply install and use Indico, follow the *production installation instructions*. For those who are interested in developing new features and plugins for Indico, check out the *development installation instructions*.

Installation guides

To simply install and use Indico, follow the *production installation instructions*. For those who are interested in developing new features and plugins for Indico, check out the *development installation instructions*.

Production

We provide guides to install Indico on CentOS and Debian systems. While other distributions are not officially supported, they should work fine, but the installation steps (especially package names) may need some slight adjustments.

Our guides cover a single-machine installation where Indico, Celery, Redis and PostgreSQL run on the same machine. This should be fine for almost all Indico instances, but adapting the steps to multiple machines is not particularly hard either.

CentOS7 / CC7

Except for minor differences, these guides apply to both vanilla CentOS7 and the CERN flavor of CentOS, CC7 (CentOS CERN 7).

nginx

Note: Please note that you **must** use Apache if you intend to use SSO using Shibboleth/SAML/ADFS. If that's not the case because you do not use SSO at all or use e.g. OAuth, we recommend using nginx.

1. Enable EPEL

```
yum install -y epel-release
```

Note: If you use CC7, EPEL is already enabled and this step is not necessary

2. Install Packages

Edit `/etc/yum.repos.d/CentOS-Base.repo` and add `exclude=postgresql*` to the `[base]` and `[updates]` sections, as described in the [PostgreSQL wiki](#).

```
yum install -y https://download.postgresql.org/pub/repos/yum/9.6/redhat/rhel-7-x86_64/
↳ pgdg-centos96-9.6-3.noarch.rpm
yum install -y postgresql96 postgresql96-server postgresql96-libs postgresql96-devel
↳ postgresql96-contrib
yum install -y gcc redis nginx uwsgi uwsgi-plugin-python
yum install -y python-devel python-virtualenv libjpeg-turbo-devel libxslt-devel
↳ libxml2-devel libffi-devel pcre-devel libyaml-devel
/usr/pgsql-9.6/bin/postgresql96-setup initdb
systemctl start postgresql-9.6.service redis.service
```

3. Create a Database

We create a user and database for indico and enable the necessary Postgres extensions (which can only be done by the Postgres superuser)

```
su - postgres -c 'createuser indico'
su - postgres -c 'createdb -O indico indico'
su - postgres -c 'psql indico -c "CREATE EXTENSION unaccent; CREATE EXTENSION pg_trgm;"
↳ "'
```

Warning: Do not forget to setup a cronjob that creates regular database backups once you start using Indico in production!

4. Configure uWSGI & nginx

The default uWSGI and nginx configuration files should work fine in most cases.

```
cat > /etc/uwsgi.ini <<'EOF'
[uwsgi]
uid = indico
gid = nginx
umask = 027
pidfile = /run/uwsgi/uwsgi.pid

processes = 4
enable-threads = true
chmod-socket = 770
```



```

socket = /opt/indico/web/uwsgi.sock
stats = /opt/indico/web/uwsgi-stats.sock
protocol = uwsgi

master = true
auto-procname = true
procname-prefix-spaced = indico
disable-logging = true

plugin = python
single-interpreter = true

touch-reload = /opt/indico/web/indico.wsgi
wsgi-file = /opt/indico/web/indico.wsgi
virtualenv = /opt/indico/.venv

vacuum = true
buffer-size = 20480
memory-report = true
max-requests = 2500
harakiri = 900
harakiri-verbose = true
reload-on-rss = 2048
evil-reload-on-rss = 8192
EOF

```

Note: Replace YOURHOSTNAME in the next file with the hostname on which your Indico instance should be available, e.g. `indico.yourdomain.com`

```

cat > /etc/nginx/conf.d/indico.conf <<'EOF'
server {
    listen 80;
    listen [::]:80;
    server_name YOURHOSTNAME;
    return 301 https://$server_name$request_uri;
}

server {
    listen      *:443 ssl http2;
    listen     [::]:443 ssl http2 default ipv6only=on;
    server_name YOURHOSTNAME;

    ssl on;

    ssl_certificate      /etc/ssl/indico/indico.crt;
    ssl_certificate_key  /etc/ssl/indico/indico.key;
    ssl_session_cache    shared:SSL:10m;
    ssl_session_timeout  5m;
    ssl_protocols        TLSv1 TLSv1.1 TLSv1.2;
    ssl_ciphers           ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-
↪POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-
↪AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-
↪AES256-GCM-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-
↪AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES256-
↪SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-
↪AES128-SHA:DHE-RSA-AES256-SHA256:DHE-RSA-AES256-SHA:ECDHE-ECDSA-DES-CBC3-SHA:ECDHE-
↪RSA-DES-CBC3-SHA:EDH-RSA-DES-CBC3-SHA:AES128-GCM-SHA256:AES256-GCM-SHA384:AES128-
↪SHA256:AES256-SHA256:AES128-SHA:AES256-SHA:DES-CBC3-SHA:!DSS;

```

```
ssl_prefer_server_ciphers on;

access_log          /opt/indico/log/nginx/access.log combined;
error_log           /opt/indico/log/nginx/error.log;

location /.xsf/indico/ {
    internal;
    alias /opt/indico/;
}

location ~ ^/static/assets/(core|(?:(?:plugin|theme)-[^\s/]+)/(.*))$ {
    alias /opt/indico/assets/$1/$2;
    access_log off;
}

location ~ ^/(css|images|js|static(?:/plugins|/assets|/themes|/custom))/(.*)$ {
    alias /opt/indico/web/htdocs/$1$2;
    access_log off;
}

location /robots.txt {
    alias /opt/indico/web/htdocs/robots.txt;
    access_log off;
}

location / {
    root /var/empty/nginx;
    include /etc/nginx/uwsgi_params;
    uwsgi_pass unix:/opt/indico/web/uwsgi.sock;
    uwsgi_param UWSGI_SCHEME $scheme;
    uwsgi_read_timeout 15m;
    uwsgi_buffers 32 32k;
    uwsgi_busy_buffers_size 128k;
    uwsgi_hide_header X-Sendfile;
    client_max_body_size 1G;
}
}
EOF
```

5. Create an SSL Certificate

First, create the folders for the certificate/key and set restrictive permissions on them:

```
mkdir /etc/ssl/indico
chown root:root /etc/ssl/indico/
chmod 700 /etc/ssl/indico
```

If you are just trying out Indico you can simply use a self-signed certificate (your browser will show a warning which you will have to confirm when accessing your Indico instance for the first time).

Note: Do not forget to replace YOURHOSTNAME with the same value you used above

```
openssl req -x509 -nodes -newkey rsa:4096 -subj /CN=YOURHOSTNAME -keyout /etc/ssl/
↪indico/indico.key -out /etc/ssl/indico/indico.crt
```

While a self-signed certificate works for testing, it is not suitable for a production system. You can either buy a certificate from any commercial certification authority or get a free one from [Let's Encrypt](#).

Note: There's an optional step later in this guide to get a certificate from Let's Encrypt. We can't do it right now since the nginx config references a directory yet to be created, which prevents nginx from starting.

6. Configure SELinux

Indico works fine with SELinux enabled, but you need to load a custom SELinux module to tell SELinux about Indico's files and how they should be handled.

```
cat > /tmp/indico.cil <<'EOF'
; define custom type that logrotate can access
(type indico_log_t)
(typeattributeset file_type (indico_log_t))
(typeattributeset logfile (indico_log_t))
(roletype object_r indico_log_t)

; allow logrotate to reload systemd services
(allow logrotate_t init_t (service (start)))
(allow logrotate_t policykit_t (dbus (send_msg)))
(allow policykit_t logrotate_t (dbus (send_msg)))

; make sure the uwsgi socket is writable by the webserver
(typetransition unconfined_service_t usr_t sock_file "uwsgi.sock" httpd_sys_rw_
↪content_t)
(filecon "/opt/indico/web/uwsgi\.sock" socket (system_u object_r httpd_sys_rw_content_
↪t ((s0) (s0))))

; set proper types for our log dirs
(filecon "/opt/indico/log(/.*)?" any (system_u object_r indico_log_t ((s0) (s0))))
(filecon "/opt/indico/log/nginx(/.*)?" any (system_u object_r httpd_log_t ((s0) (s0))))
EOF
semodule -i /tmp/indico.cil
```

7. Install Indico

Celery runs as a background daemon. Add a systemd unit file for it:

```
cat > /etc/systemd/system/indico-celery.service <<'EOF'
[Unit]
Description=Indico Celery
After=network.target

[Service]
ExecStart=/opt/indico/.venv/bin/indico celery worker -B
Restart=always
SyslogIdentifier=indico-celery
User=indico
Group=nginx
UMask=0027
Type=simple
```

```
[Install]
WantedBy=multi-user.target
EOF
systemctl daemon-reload
```

Now create a user that will be used to run Indico and switch to it:

```
useradd -rm -g nginx -d /opt/indico -s /bin/bash indico
su - indico
```

You are now ready to install Indico:

```
virtualenv ~/.venv
source ~/.venv/bin/activate
pip install -U pip setuptools
pip install --pre indico
```

8. Configure Indico

Once Indico is installed, you can run the configuration wizard. You can keep the defaults for most options, but make sure to use `https://YOURHOSTNAME` when prompted for the Indico URL. Also specify valid email addresses when asked and enter a valid SMTP server Indico can use to send emails. When asked for the default timezone make sure this is the main time zone used in your Indico instance.

```
indico setup wizard
```

Now finish setting up the directory structure and permissions:

```
mkdir ~/log/nginx
chmod go-rwx ~/* ~/.[^.]*
chmod 710 ~/ ~/archive ~/assets ~/cache ~/log ~/tmp
chmod 750 ~/web ~/.venv
chmod g+w ~/log/nginx
restorecon -R ~/
echo -e "\nSTATIC_FILE_METHOD = ('xaccelredirect', {'/opt/indico': '/.xsf/indico'})" >
↪> ~/etc/indico.conf
```

9. Create database schema

Finally you can create the database schema and switch back to *root*:

```
indico db prepare
exit
```

10. Launch Indico

You can now start Indico and set it up to start automatically when the server is rebooted:

```
systemctl restart uwsgi.service nginx.service indico-celery.service
systemctl enable uwsgi.service nginx.service postgresql-9.6.service redis.service_
↪indico-celery.service
```

11. Open the Firewall

```
firewall-cmd --permanent --add-port 443/tcp --add-port 80/tcp
firewall-cmd --reload
```

Note: This is only needed if you use CC7 as CentOS7 has no firewall enabled by default

12. Optional: Get a Certificate from Let's Encrypt

To avoid ugly SSL warnings in your browsers, the easiest option is to get a free certificate from Let's Encrypt. We also enable the cronjob to renew it automatically:

```
yum install -y python-certbot-nginx
certbot --nginx --rsa-key-size 4096 --no-redirect --staple-ocsp -d YOURHOSTNAME
rm -rf /etc/ssl/indico
systemctl start certbot-renew.timer
systemctl enable certbot-renew.timer
```

13. Create an Indico user

Access <https://YOURHOSTNAME> in your browser and follow the steps displayed there to create your initial user.

Apache

1. Enable EPEL

```
yum install -y epel-release
```

Note: If you use CC7, EPEL is already enabled and this step is not necessary

2. Install Packages

Edit `/etc/yum.repos.d/CentOS-Base.repo` and add `exclude=postgresql*` to the `[base]` and `[updates]` sections, as described in the [PostgreSQL wiki](#).

```
yum install -y https://download.postgresql.org/pub/repos/yum/9.6/redhat/rhel-7-x86_64/
↳pgdg-centos96-9.6-3.noarch.rpm
yum install -y postgresql96 postgresql96-server postgresql96-libs postgresql96-devel
↳postgresql96-contrib
yum install -y httpd mod_proxy_uwsgi mod_ssl mod_xsendfile
yum install -y gcc redis uwsgi uwsgi-plugin-python
yum install -y python-devel python-virtualenv libjpeg-turbo-devel libxslt-devel
↳libxml2-devel libffi-devel pcre-devel libyaml-devel
/usr/pgsql-9.6/bin/postgresql96-setup initdb
systemctl start postgresql-9.6.service redis.service
```

3. Create a Database

We create a user and database for indico and enable the necessary Postgres extensions (which can only be done by the Postgres superuser)

```
su - postgres -c 'createuser indico'
su - postgres -c 'createdb -O indico indico'
su - postgres -c 'psql indico -c "CREATE EXTENSION unaccent; CREATE EXTENSION pg_trgm;
↵"'
```

Warning: Do not forget to setup a cronjob that creates regular database backups once you start using Indico in production!

4. Configure uWSGI & Apache

The default uWSGI and Apache configuration files should work fine in most cases.

```
cat > /etc/uwsgi.ini <<'EOF'
[uwsgi]
uid = indico
gid = apache
umask = 027
pidfile = /run/uwsgi/uwsgi.pid

processes = 4
enable-threads = true
socket = 127.0.0.1:8008
stats = /opt/indico/web/uwsgi-stats.sock
protocol = uwsgi

master = true
auto-procname = true
procname-prefix-spaced = indico
disable-logging = true

plugin = python
single-interpreter = true

touch-reload = /opt/indico/web/indico.wsgi
wsgi-file = /opt/indico/web/indico.wsgi
virtualenv = /opt/indico/.venv

vacuum = true
buffer-size = 20480
memory-report = true
max-requests = 2500
harakiri = 900
harakiri-verbose = true
reload-on-rss = 2048
evil-reload-on-rss = 8192
EOF
```

Note: Replace YOURHOSTNAME in the next files with the hostname on which your Indico instance should be avail-

able, e.g. `indico.yourdomain.com`

```

cat > /etc/httpd/conf.d/indico-sslredir.conf <<'EOF'
<VirtualHost *:80>
    ServerName YOURHOSTNAME
    RewriteEngine On
    RewriteRule ^(.*)$ https://{%{HTTP_HOST}}$1 [R=301,L]
</VirtualHost>
EOF

cat > /etc/httpd/conf.d/indico.conf <<'EOF'
<VirtualHost *:443>
    ServerName YOURHOSTNAME
    DocumentRoot "/var/empty/apache"

    SSLEngine on
    SSLCertificateFile /etc/ssl/indico/indico.crt
    SSLCertificateChainFile /etc/ssl/indico/indico.crt
    SSLCertificateKeyFile /etc/ssl/indico/indico.key
    SSLProtocol all -SSLv2 -SSLv3
    SSLCipherSuite ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-
↪POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-
↪AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-
↪AES256-GCM-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-
↪AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES256-
↪SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-
↪AES128-SHA:DHE-RSA-AES256-SHA256:DHE-RSA-AES256-SHA:ECDHE-ECDSA-DES-CBC3-SHA:ECDHE-
↪RSA-DES-CBC3-SHA:EDH-RSA-DES-CBC3-SHA:AES128-GCM-SHA256:AES256-GCM-SHA384:AES128-
↪SHA256:AES256-SHA256:AES128-SHA:AES256-SHA:DES-CBC3-SHA:!DSS
    SSLHonorCipherOrder on

    XSendFile on
    XSendFilePath /opt/indico
    CustomLog /opt/indico/log/apache/access.log combined
    ErrorLog /opt/indico/log/apache/error.log
    LogLevel error
    ServerSignature Off

    AliasMatch "^/static/assets/(core|(?plugin|theme)-[^\s]+)/(.*)$" "/opt/indico/
↪assets/$1/$2"
    AliasMatch "^/(css|images|js|static(?:/plugins|assets|themes|custom))(/.*)$" "/"
↪opt/indico/web/htdocs/$1$2"
    Alias /robots.txt /opt/indico/web/htdocs/robots.txt

    SetEnv UWSGI_SCHEME https
    ProxyPass / uwsgi://127.0.0.1:8008/

    <Directory /opt/indico>
        AllowOverride None
        Require all granted
    </Directory>
</VirtualHost>
EOF

```

Now enable the uwsgi proxy module in apache:

```
echo 'LoadModule proxy_uwsgi_module modules/mod_proxy_uwsgi.so' > /etc/httpd/conf.  
↳modules.d/proxy_uwsgi.conf
```

5. Create an SSL Certificate

First, create the folders for the certificate/key and set restrictive permissions on them:

```
mkdir /etc/ssl/indico  
chown root:root /etc/ssl/indico/  
chmod 700 /etc/ssl/indico
```

If you are just trying out Indico you can simply use a self-signed certificate (your browser will show a warning which you will have to confirm when accessing your Indico instance for the first time).

Note: Do not forget to replace YOURHOSTNAME with the same value you used above

```
openssl req -x509 -nodes -newkey rsa:4096 -subj /CN=YOURHOSTNAME -keyout /etc/ssl/  
↳indico/indico.key -out /etc/ssl/indico/indico.crt
```

While a self-signed certificate works for testing, it is not suitable for a production system. You can either buy a certificate from any commercial certification authority or get a free one from [Let's Encrypt](#).

Note: There's an optional step later in this guide to get a certificate from Let's Encrypt. We can't do it right now since the Apache config references a directory yet to be created, which prevents Apache from starting.

6. Configure SELinux

Indico works fine with SELinux enabled, but you need to load a custom SELinux module to tell SELinux about Indico's files and how they should be handled.

```
cat > /tmp/indico.cil <<'EOF'  
; define custom type that logrotate can access  
(type indico_log_t)  
(typeattributeset file_type (indico_log_t))  
(typeattributeset logfile (indico_log_t))  
(roletype object_r indico_log_t)  
  
; allow logrotate to reload systemd services  
(allow logrotate_t init_t (service (start)))  
(allow logrotate_t policykit_t (dbus (send_msg)))  
(allow policykit_t logrotate_t (dbus (send_msg)))  
  
; make sure the uwsgi socket is writable by the webserver  
(typetransition unconfined_service_t usr_t sock_file "uwsgi.sock" httpd_sys_rw_  
↳content_t)  
(filecon "/opt/indico/web/uwsgi\.sock" socket (system_u object_r httpd_sys_rw_content_  
↳t ((s0) (s0))))  
  
; set proper types for our log dirs  
(filecon "/opt/indico/log(/.*)?" any (system_u object_r indico_log_t ((s0) (s0))))  
(filecon "/opt/indico/log/apache(/.*)?" any (system_u object_r httpd_log_t_  
↳((s0) (s0))))
```



```
EOF
semodule -i /tmp/indico.cil
```

7. Install Indico

Celery runs as a background daemon. Add a systemd unit file for it:

```
cat > /etc/systemd/system/indico-celery.service <<'EOF'
[Unit]
Description=Indico Celery
After=network.target

[Service]
ExecStart=/opt/indico/.venv/bin/indico celery worker -B
Restart=always
SyslogIdentifier=indico-celery
User=indico
Group=apache
UMask=0027
Type=simple

[Install]
WantedBy=multi-user.target
EOF
systemctl daemon-reload
```

Now create a user that will be used to run Indico and switch to it:

```
useradd -rm -g apache -d /opt/indico -s /bin/bash indico
su - indico
```

You are now ready to install Indico:

```
virtualenv ~/.venv
source ~/.venv/bin/activate
pip install -U pip setuptools
pip install --pre indico
```

8. Configure Indico

Once Indico is installed, you can run the configuration wizard. You can keep the defaults for most options, but make sure to use `https://YOURHOSTNAME` when prompted for the Indico URL. Also specify valid email addresses when asked and enter a valid SMTP server Indico can use to send emails. When asked for the default timezone make sure this is the main time zone used in your Indico instance.

```
indico setup wizard
```

Now finish setting up the directory structure and permissions:

```
mkdir ~/log/apache
chmod go-rwx ~/* ~/.[^.]*
chmod 710 ~/~/archive ~/assets ~/cache ~/log ~/tmp
chmod 750 ~/web ~/.venv
chmod g+w ~/log/apache
```

```
restorecon -R ~/
echo -e "\nSTATIC_FILE_METHOD = 'xsendfile'" >> ~/etc/indico.conf
```

9. Create database schema

Finally you can create the database schema and switch back to *root*:

```
indico db prepare
exit
```

10. Launch Indico

You can now start Indico and set it up to start automatically when the server is rebooted:

```
systemctl restart uwsgi.service httpd.service indico-celery.service
systemctl enable uwsgi.service httpd.service postgresql-9.6.service redis.service_
↳ indico-celery.service
```

11. Open the Firewall

```
firewall-cmd --permanent --add-port 443/tcp --add-port 80/tcp
firewall-cmd --reload
```

Note: This is only needed if you use CC7 as CentOS7 has no firewall enabled by default

12. Optional: Get a Certificate from Let's Encrypt

To avoid ugly SSL warnings in your browsers, the easiest option is to get a free certificate from Let's Encrypt. We also enable the cronjob to renew it automatically:

```
yum install -y python-certbot-apache
certbot --apache --rsa-key-size 4096 --no-redirect --staple-ocsp -d YOURHOSTNAME
rm -rf /etc/ssl/indico
systemctl start certbot-renew.timer
systemctl enable certbot-renew.timer
```

13. Create an Indico user

Access `https://YOURHOSTNAME` in your browser and follow the steps displayed there to create your initial user.

Optional: Shibboleth

If your organization uses Shibboleth/SAML-based SSO, follow these steps to use it in Indico:

1. Install Shibboleth

Add the Shibboleth yum repository:

Note: If you use CC7, Shibboleth is already available and there is no need to add the repo manually.

```
curl -fsSL -o /etc/yum.repos.d/shibboleth.repo 'https://shibboleth.net/cgi-bin/sp_
↪repo.cgi?platform=CentOS_7'
```

Now install Shibboleth itself. When prompted to accept the GPG key of the Shibboleth yum repo, confirm the prompt.

```
setsebool httpd_can_network_connect 1
yum install -y shibboleth xmltooling-schemas opensaml-schemas
```

2. Configure Shibboleth

This is outside the scope of this documentation and depends on your environment (Shibboleth, SAML, ADFS, etc). Please contact whoever runs your SSO infrastructure if you need assistance.

3. Enable Shibboleth in Apache

Add the following code to your `/etc/httpd/conf.d/indico.conf` right before the `AliasMatch` lines:

```
<LocationMatch "^(/Shibboleth\.sso|/login/shib-sso/shibboleth)">
  AuthType shibboleth
  ShibRequestSetting requireSession 1
  ShibExportAssertion Off
  Require valid-user
</LocationMatch>
```

4. Enable Shibboleth in Indico

Add the following code to your `/opt/indico/etc/indico.conf`:

```
# SSO
AUTH_PROVIDERS = {
  'shib-sso': {
    'type': 'shibboleth',
    'title': 'SSO',
    'attrs_prefix': 'ADFS_',
    'callback_uri': '/login/shib-sso/shibboleth',
    # 'logout_uri': 'https://login.yourcompany.tld/logout'
  }
}
IDENTITY_PROVIDERS = {
  'shib-sso': {
    'type': 'shibboleth',
    'title': 'SSO',
    'identifier_field': 'ADFS_LOGIN',
    'mapping': {
      'affiliation': 'ADFS_HOMEINSTITUTE',
```

```
'first_name': 'ADFS_FIRSTNAME',
'last_name': 'ADFS_LASTNAME',
'email': 'ADFS_EMAIL',
'phone': 'ADFS_PHONENUMBER'
},
'trusted_email': True
}
}
```

The values for `attrs_prefix`, `mapping` and `identifier_field` may be different in your environment. Uncomment and set `logout_uri` if your SSO infrastructure provides a logout URL (usually used to log you out from all applications).

If you only want to use SSO, without allowing people to login locally using username/password, disable it by setting `LOCAL_IDENTITIES = False` in `indico.conf`.

Warning: We assume that emails received from SSO are already validated. If this is not the case, make sure to disable `trusted_email` which will require email validation in Indico when logging in for the first time. Otherwise people could take over the account of someone else by using their email address!

Note: The example config is rather simple and only accesses data from SSO during login. This is not sufficient for advanced features such as automatic synchronization of names, affiliations and phone numbers or using centrally managed groups. To use these features, you need to use e.g. the LDAP identity provider and use the information received via SSO to retrieve the user details from LDAP. If you need assistance with this, feel free to ask us on IRC (`#indico @ Freenode`) or via e-mail (`indico-team@cern.ch`).

Note: Please note that you **must** use Apache if you intend to use SSO using Shibboleth/SAML/ADFS. If that's not the case because you do not use SSO at all or use e.g. OAuth, we recommend using nginx.

Debian / Ubuntu

Except for minor differences, this guide applies to both Debian and Ubuntu. It has been tested with Debian 8 (Jessie), Debian 9 (Stretch) and Ubuntu 16.04 (Xenial).

nginx

Note: Please note that you **must** use Apache if you intend to use SSO using Shibboleth/SAML/ADFS. If that's not the case because you do not use SSO at all or use e.g. OAuth, we recommend using nginx.

1. Install Packages

PostgreSQL and nginx are installed from their upstream repos to get much more recent versions.

```

echo "deb http://apt.postgresql.org/pub/repos/apt/ $(lsb_release -cs)-pgdg main" > /
↳etc/apt/sources.list.d/pgdg.list
echo "deb http://nginx.org/packages/$(lsb_release -is | tr '[:upper:]' '[:lower:]')/
↳$(lsb_release -cs) nginx" > /etc/apt/sources.list.d/nginx.list
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | apt-key add -
wget --quiet -O - https://nginx.org/keys/nginx_signing.key | apt-key add -
apt update
apt install -y --install-recommends postgresql-9.6 libpq-dev nginx python-dev python-
↳virtualenv libxslt1-dev libxml2-dev libffi-dev libpcre3-dev libyaml-dev build-
↳essential redis-server uwsgi uwsgi-plugin-python

```

If you use Debian, run this command:

```
apt install -y libjpeg62-turbo-dev
```

If you use Ubuntu, run this instead:

```
apt install -y libjpeg-turbo8-dev zlib1g-dev
```

2. Create a Database

We create a user and database for indico and enable the necessary Postgres extensions (which can only be done by the Postgres superuser)

```

su - postgres -c 'createuser indico'
su - postgres -c 'createdb -O indico indico'
su - postgres -c 'psql indico -c "CREATE EXTENSION unaccent; CREATE EXTENSION pg_trgm;
↳"'

```

Warning: Do not forget to setup a cronjob that creates regular database backups once you start using Indico in production!

3. Configure uWSGI & nginx

The default uWSGI and nginx configuration files should work fine in most cases.

```

ln -s /etc/uwsgi/apps-available/indico.ini /etc/uwsgi/apps-enabled/indico.ini
cat > /etc/uwsgi/apps-available/indico.ini <<'EOF'
[uwsgi]
uid = indico
gid = nginx
umask = 027

processes = 4
enable-threads = true
chmod-socket = 770
socket = /opt/indico/web/uwsgi.sock
stats = /opt/indico/web/uwsgi-stats.sock
protocol = uwsgi

master = true
auto-procname = true

```

```

procname-prefix-spaced = indico
disable-logging = true

plugin = python
single-interpreter = true

touch-reload = /opt/indico/web/indico.wsgi
wsgi-file = /opt/indico/web/indico.wsgi
virtualenv = /opt/indico/.venv

vacuum = true
buffer-size = 20480
memory-report = true
max-requests = 2500
harakiri = 900
harakiri-verbose = true
reload-on-rss = 2048
evil-reload-on-rss = 8192
EOF

```

Note: Replace YOURHOSTNAME in the next file with the hostname on which your Indico instance should be available, e.g. indico.yourdomain.com

```

cat > /etc/nginx/conf.d/indico.conf <<'EOF'
server {
    listen 80;
    listen [::]:80;
    server_name YOURHOSTNAME;
    return 301 https://$server_name$request_uri;
}

server {
    listen      *:443 ssl http2;
    listen      [::]:443 ssl http2 default ipv6only=on;
    server_name YOURHOSTNAME;

    ssl on;

    ssl_certificate      /etc/ssl/indico/indico.crt;
    ssl_certificate_key  /etc/ssl/indico/indico.key;
    ssl_session_cache    shared:SSL:10m;
    ssl_session_timeout  5m;
    ssl_protocols        TLSv1 TLSv1.1 TLSv1.2;
    ssl_ciphers          ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-
↪POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-
↪AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-
↪AES256-GCM-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-
↪AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES256-
↪SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-
↪AES128-SHA:DHE-RSA-AES256-SHA256:DHE-RSA-AES256-SHA:ECDHE-ECDSA-DES-CBC3-SHA:ECDHE-
↪RSA-DES-CBC3-SHA:EDH-RSA-DES-CBC3-SHA:AES128-GCM-SHA256:AES256-GCM-SHA384:AES128-
↪SHA256:AES256-SHA256:AES128-SHA:AES256-SHA:DES-CBC3-SHA:!DSS;
    ssl_prefer_server_ciphers on;

    access_log          /opt/indico/log/nginx/access.log combined;
    error_log           /opt/indico/log/nginx/error.log;
}
EOF

```

```

location /.xsf/indico/ {
    internal;
    alias /opt/indico/;
}

location ~ ^/static/assets/(core|(?:(?:plugin|theme)-[^\s/]+)/(.*))$ {
    alias /opt/indico/assets/$1/$2;
    access_log off;
}

location ~ ^/(css|images|js|static(?:/plugins|/assets|/themes|/custom))/(.*)$ {
    alias /opt/indico/web/htdocs/$1$2;
    access_log off;
}

location /robots.txt {
    alias /opt/indico/web/htdocs/robots.txt;
    access_log off;
}

location / {
    root /var/empty/nginx;
    include /etc/nginx/uwsgi_params;
    uwsgi_pass unix:/opt/indico/web/uwsgi.sock;
    uwsgi_param UWSGI_SCHEME $scheme;
    uwsgi_read_timeout 15m;
    uwsgi_buffers 32 32k;
    uwsgi_busy_buffers_size 128k;
    uwsgi_hide_header X-Sendfile;
    client_max_body_size 1G;
}
}
EOF

```

4. Create an SSL Certificate

First, create the folders for the certificate/key and set restrictive permissions on them:

```

mkdir /etc/ssl/indico
chown root:root /etc/ssl/indico/
chmod 700 /etc/ssl/indico

```

If you are just trying out Indico you can simply use a self-signed certificate (your browser will show a warning which you will have to confirm when accessing your Indico instance for the first time).

Note: Do not forget to replace YOURHOSTNAME with the same value you used above

```

openssl req -x509 -nodes -newkey rsa:4096 -subj /CN=YOURHOSTNAME -keyout /etc/ssl/
↪indico/indico.key -out /etc/ssl/indico/indico.crt

```

While a self-signed certificate works for testing, it is not suitable for a production system. You can either buy a certificate from any commercial certification authority or get a free one from [Let's Encrypt](#).

Note: There's an optional step later in this guide to get a certificate from Let's Encrypt. We can't do it right now since the nginx config references a directory yet to be created, which prevents nginx from starting.

5. Install Indico

Celery runs as a background daemon. Add a systemd unit file for it:

```
cat > /etc/systemd/system/indico-celery.service <<'EOF'
[Unit]
Description=Indico Celery
After=network.target

[Service]
ExecStart=/opt/indico/.venv/bin/indico celery worker -B
Restart=always
SyslogIdentifier=indico-celery
User=indico
Group=nginx
UMask=0027
Type=simple

[Install]
WantedBy=multi-user.target
EOF
systemctl daemon-reload
```

Now create a user that will be used to run Indico and switch to it:

```
useradd -rm -g nginx -d /opt/indico -s /bin/bash indico
su - indico
```

You are now ready to install Indico:

```
virtualenv ~/.venv
source ~/.venv/bin/activate
pip install -U pip setuptools
pip install --pre indico
```

6. Configure Indico

Once Indico is installed, you can run the configuration wizard. You can keep the defaults for most options, but make sure to use `https://YOURHOSTNAME` when prompted for the Indico URL. Also specify valid email addresses when asked and enter a valid SMTP server Indico can use to send emails. When asked for the default timezone make sure this is the main time zone used in your Indico instance.

```
indico setup wizard
```

Now finish setting up the directory structure and permissions:

```
mkdir ~/log/nginx
chmod go-rwx ~/* ~/.[^.]*
chmod 710 ~/ ~/archive ~/assets ~/cache ~/log ~/tmp
chmod 750 ~/web ~/.venv
```



```
chmod g+w ~/log/nginx
echo -e "\nSTATIC_FILE_METHOD = ('xaccelredirect', {'/opt/indico': '/.xsf/indico'})" >
↪> ~/etc/indico.conf
```

7. Create database schema

Finally, you can create the database schema and switch back to *root*:

```
indico db prepare
exit
```

8. Launch Indico

You can now start Indico and set it up to start automatically when the server is rebooted:

```
systemctl restart uwsgi.service nginx.service indico-celery.service
systemctl enable uwsgi.service nginx.service postgresql.service redis-server.service_
↪indico-celery.service
```

9. Optional: Get a Certificate from Let's Encrypt

Note: You need to use at least Debian 9 (Stretch) to use certbot. If you are still using Debian 8 (Jessie), consider updating or install certbot from backports.

If you use Ubuntu, install the certbot PPA:

```
apt install -y software-properties-common
add-apt-repository -y ppa:certbot/certbot
apt update
```

To avoid ugly SSL warnings in your browsers, the easiest option is to get a free certificate from Let's Encrypt. We also enable the cronjob to renew it automatically:

```
apt install -y python-certbot-nginx
certbot --nginx --rsa-key-size 4096 --no-redirect --staple-ocsp -d YOURHOSTNAME
rm -rf /etc/ssl/indico
systemctl start certbot.timer
systemctl enable certbot.timer
```

10. Create an Indico user

Access <https://YOURHOSTNAME> in your browser and follow the steps displayed there to create your initial user.

Apache

1. Install Packages

PostgreSQL is installed from its upstream repos to get a much more recent version.

```
echo "deb http://apt.postgresql.org/pub/repos/apt/ $(lsb_release -cs)-pgdg main" > /
↳etc/apt/sources.list.d/pgdg.list
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | apt-key add -
apt update
apt install -y --install-recommends postgresql-9.6 libpq-dev apache2 libapache2-mod-
↳proxy-uwsgi libapache2-mod-xsendfile python-dev python-virtualenv libxslt1-dev
↳libxml2-dev libffi-dev libpcre3-dev libyaml-dev build-essential redis-server uwsgi
↳uwsgi-plugin-python
```

If you use Debian, run this command:

```
apt install -y libjpeg62-turbo-dev
```

If you use Ubuntu, run this instead:

```
apt install -y libjpeg-turbo8-dev zlib1g-dev
```

2. Create a Database

We create a user and database for indico and enable the necessary Postgres extensions (which can only be done by the Postgres superuser)

```
su - postgres -c 'createuser indico'
su - postgres -c 'createdb -O indico indico'
su - postgres -c 'psql indico -c "CREATE EXTENSION unaccent; CREATE EXTENSION pg_trgm;
↳"'
```

Warning: Do not forget to setup a cronjob that creates regular database backups once you start using Indico in production!

3. Configure uWSGI & Apache

The default uWSGI and Apache configuration files should work fine in most cases.

```
ln -s /etc/uwsgi/apps-available/indico.ini /etc/uwsgi/apps-enabled/indico.ini
cat > /etc/uwsgi/apps-available/indico.ini <<'EOF'
[uwsgi]
uid = indico
gid = www-data
umask = 027

processes = 4
enable-threads = true
socket = 127.0.0.1:8008
stats = /opt/indico/web/uwsgi-stats.sock
protocol = uwsgi

master = true
```

```

auto-procname = true
procname-prefix-spaced = indico
disable-logging = true

plugin = python
single-interpreter = true

touch-reload = /opt/indico/web/indico.wsgi
wsgi-file = /opt/indico/web/indico.wsgi
virtualenv = /opt/indico/.venv

vacuum = true
buffer-size = 20480
memory-report = true
max-requests = 2500
harakiri = 900
harakiri-verbose = true
reload-on-rss = 2048
evil-reload-on-rss = 8192
EOF

```

Note: Replace YOURHOSTNAME in the next files with the hostname on which your Indico instance should be available, e.g. indico.yourdomain.com

```

cat > /etc/apache2/sites-available/indico-sslredir.conf <<'EOF'
<VirtualHost *:80>
    ServerName YOURHOSTNAME
    RewriteEngine On
    RewriteRule ^(.*)$ https://%{HTTP_HOST}$1 [R=301,L]
</VirtualHost>
EOF

cat > /etc/apache2/sites-available/indico.conf <<'EOF'
<VirtualHost *:443>
    ServerName YOURHOSTNAME
    DocumentRoot "/var/empty/apache"

    SSLEngine on
    SSLCertificateFile /etc/ssl/indico/indico.crt
    SSLCertificateKeyFile /etc/ssl/indico/indico.key
    SSLProtocol all -SSLv2 -SSLv3
    SSLCipherSuite ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-
↪POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-
↪AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-
↪AES256-GCM-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-
↪AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES256-
↪SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-
↪AES128-SHA:DHE-RSA-AES256-SHA256:DHE-RSA-AES256-SHA:ECDHE-ECDSA-DES-CBC3-SHA:ECDHE-
↪RSA-DES-CBC3-SHA:EDH-RSA-DES-CBC3-SHA:AES128-GCM-SHA256:AES256-GCM-SHA384:AES128-
↪SHA256:AES256-SHA256:AES128-SHA:AES256-SHA:DES-CBC3-SHA:!DSS
    SSLHonorCipherOrder on

    XSendFile on
    XSendFilePath /opt/indico
    CustomLog /opt/indico/log/apache/access.log combined
    ErrorLog /opt/indico/log/apache/error.log

```

```
LogLevel error
ServerSignature Off

AliasMatch "^/static/assets/(core|(?:(?:plugin|theme)-[^\/]+)/(.*)$" "/opt/indico/
↪assets/$1/$2"
AliasMatch "^/(css|images|js|static(?:/plugins|/assets|/themes|/custom))(/.*)$" "/
↪opt/indico/web/htdocs/$1$2"
Alias /robots.txt /opt/indico/web/htdocs/robots.txt

SetEnv UWSGI_SCHEME https
ProxyPass / uwsgi://127.0.0.1:8008/

<Directory /opt/indico>
    AllowOverride None
    Require all granted
</Directory>
</VirtualHost>
EOF
```

Now enable the necessary modules and the indico site in apache:

```
a2enmod proxy_uwsgi rewrite ssl xsendfile
a2dissite 000-default
a2ensite indico indico-sslredir
```

4. Create an SSL Certificate

First, create the folders for the certificate/key and set restrictive permissions on them:

```
mkdir /etc/ssl/indico
chown root:root /etc/ssl/indico/
chmod 700 /etc/ssl/indico
```

If you are just trying out Indico you can simply use a self-signed certificate (your browser will show a warning which you will have to confirm when accessing your Indico instance for the first time).

Note: Do not forget to replace `YOURHOSTNAME` with the same value you used above

```
openssl req -x509 -nodes -newkey rsa:4096 -subj /CN=YOURHOSTNAME -keyout /etc/ssl/
↪indico/indico.key -out /etc/ssl/indico/indico.crt
```

While a self-signed certificate works for testing, it is not suitable for a production system. You can either buy a certificate from any commercial certification authority or get a free one from [Let's Encrypt](#).

Note: There's an optional step later in this guide to get a certificate from Let's Encrypt. We can't do it right now since the Apache config references a directory yet to be created, which prevents Apache from starting.

5. Install Indico

Celery runs as a background daemon. Add a systemd unit file for it:

```

cat > /etc/systemd/system/indico-celery.service <<'EOF'
[Unit]
Description=Indico Celery
After=network.target

[Service]
ExecStart=/opt/indico/.venv/bin/indico celery worker -B
Restart=always
SyslogIdentifier=indico-celery
User=indico
Group=www-data
UMask=0027
Type=simple

[Install]
WantedBy=multi-user.target
EOF
systemctl daemon-reload

```

Now create a user that will be used to run Indico and switch to it:

```

useradd -rm -g www-data -d /opt/indico -s /bin/bash indico
su - indico

```

You are now ready to install Indico:

```

virtualenv ~/.venv
source ~/.venv/bin/activate
pip install -U pip setuptools
pip install --pre indico

```

6. Configure Indico

Once Indico is installed, you can run the configuration wizard. You can keep the defaults for most options, but make sure to use `https://YOURHOSTNAME` when prompted for the Indico URL. Also specify valid email addresses when asked and enter a valid SMTP server Indico can use to send emails. When asked for the default timezone make sure this is the main time zone used in your Indico instance.

```

indico setup wizard

```

Now finish setting up the directory structure and permissions:

```

mkdir ~/log/apache
chmod go-rwx ~/* ~/.[^.]*
chmod 710 ~/~/archive ~/assets ~/cache ~/log ~/tmp
chmod 750 ~/web ~/.venv
chmod g+w ~/log/apache
echo -e "\nSTATIC_FILE_METHOD = 'xsendfile'" >> ~/etc/indico.conf

```

7. Create database schema

Finally, you can create the database schema and switch back to `root`:

```
indico db prepare
exit
```

8. Launch Indico

You can now start Indico and set it up to start automatically when the server is rebooted:

```
systemctl restart uwsgi.service apache2.service indico-celery.service
systemctl enable uwsgi.service apache2.service postgresql.service redis-server.
↳service indico-celery.service
```

9. Optional: Get a Certificate from Let's Encrypt

Note: You need to use at least Debian 9 (Stretch) to use certbot. If you are still using Debian 8 (Jessie), consider updating or install certbot from backports.

If you use Ubuntu, install the certbot PPA:

```
apt install -y software-properties-common
add-apt-repository -y ppa:certbot/certbot
apt update
```

To avoid ugly SSL warnings in your browsers, the easiest option is to get a free certificate from Let's Encrypt. We also enable the cronjob to renew it automatically:

```
apt install -y python-certbot-apache
certbot --apache --rsa-key-size 4096 --no-redirect --staple-ocsp -d YOURHOSTNAME
rm -rf /etc/ssl/indico
systemctl start certbot.timer
systemctl enable certbot.timer
```

10. Create an Indico user

Access <https://YOURHOSTNAME> in your browser and follow the steps displayed there to create your initial user.

Optional: Shibboleth

If your organization uses Shibboleth/SAML-based SSO, follow these steps to use it in Indico:

1. Install Shibboleth

```
apt install -y libapache2-mod-shib2
a2enmod shib2
```

2. Configure Shibboleth

This is outside the scope of this documentation and depends on your environment (Shibboleth, SAML, ADFS, etc). Please contact whoever runs your SSO infrastructure if you need assistance.

3. Enable Shibboleth in Apache

Add the following code to your `/etc/apache2/sites-available/indico.conf` right before the `AliasMatch` lines:

```
<LocationMatch "^(/Shibboleth\.sso|/login/shib-sso/shibboleth)">
  AuthType shibboleth
  ShibRequestSetting requireSession 1
  ShibExportAssertion Off
  Require valid-user
</LocationMatch>
```

4. Enable Shibboleth in Indico

Add the following code to your `/opt/indico/etc/indico.conf`:

```
# SSO
AUTH_PROVIDERS = {
    'shib-sso': {
        'type': 'shibboleth',
        'title': 'SSO',
        'attrs_prefix': 'ADFS_',
        'callback_uri': '/login/shib-sso/shibboleth',
        # 'logout_uri': 'https://login.yourcompany.tld/logout'
    }
}
IDENTITY_PROVIDERS = {
    'shib-sso': {
        'type': 'shibboleth',
        'title': 'SSO',
        'identifier_field': 'ADFS_LOGIN',
        'mapping': {
            'affiliation': 'ADFS_HOMEINSTITUTE',
            'first_name': 'ADFS_FIRSTNAME',
            'last_name': 'ADFS_LASTNAME',
            'email': 'ADFS_EMAIL',
            'phone': 'ADFS_PHONENUMBER'
        },
        'trusted_email': True
    }
}
```

The values for `attrs_prefix`, `mapping` and `identifier_field` may be different in your environment. Uncomment and set `logout_uri` if your SSO infrastructure provides a logout URL (usually used to log you out from all applications).

If you only want to use SSO, without allowing people to login locally using username/password, disable it by setting `LOCAL_IDENTITYES = False` in `indico.conf`.

Warning: We assume that emails received from SSO are already validated. If this is not the case, make sure to disable `trusted_email` which will require email validation in Indico when logging in for the first time. Otherwise people could take over the account of someone else by using their email address!

Note: The example config is rather simple and only accesses data from SSO during login. This is not sufficient for advanced features such as automatic synchronization of names, affiliations and phone numbers or using centrally managed groups. To use these features, you need to use e.g. the LDAP identity provider and use the information received via SSO to retrieve the user details from LDAP. If you need assistance with this, feel free to ask us on IRC (`#indico @ Freenode`) or via e-mail (`indico-team@cern.ch`).

Note: Please note that you **must** use Apache if you intend to use SSO using Shibboleth/SAML/ADFS. If that's not the case because you do not use SSO at all or use e.g. OAuth, we recommend using nginx.

Upgrade

It is important to keep your Indico instance up to date to have the latest bug fixes and features. Upgrading can be done with almost no user-facing downtime.

Warning: When upgrading a production system it is highly recommended to create a database backup before starting.

First of all, stop the Celery worker. To do so, run this as *root*:

```
systemctl stop indico-celery.service
```

Now switch to the *indico* user and activate the virtualenv:

```
su - indico
source ~/.venv/bin/activate
```

You are now ready to install the latest version of Indico:

```
pip install -U --pre indico
```

Some versions may include database schema upgrades. Make sure to perform them immediately after upgrading. If there are no schema changes, the command will simply do nothing.

```
indico db upgrade
```

Note: Some database structure changes require an *exclusive lock* on some tables in the database. Unless you have very high activity on your instance, this lock can be acquired quickly, but if the upgrade command seems to hang for more than a few seconds, you can restart uWSGI by running `systemctl restart uwsgi.service` as *root* (in a separate shell, i.e. don't abort the upgrade command!) which will ensure nothing is accessing Indico for a moment.

Unless you just restarted uWSGI, it is now time to reload it so the new version is actually used:


```
touch ~/web/indico.wsgi
```

Also start the Celery worker again (once again, as *root*):

```
systemctl start indico-celery.service
```

Upgrading from 1.9.11 to 2.0

Make sure that you have the latest 1.9.11 version installed and that you used `indico db upgrade` to have the most recent database structure.

To upgrade to 2.0, follow the upgrade instructions above. After successfully running the upgrade, use `indico db reset_alembic` to clear pre-2.0 database migration information, since all the old migration steps from the 1.9.x version line have been removed in 2.0.

Upgrade Indico from 1.2

Warning: ATTENTION: This process is not yet fully tested, use it at your own risk! We are currently working with some of our users running Indico 1.2 in order to ensure migration works reliably and without any data loss. If you'd like to help, please [let us know!](#)

If you're running a version that is lower than 2.0, you will have to run a special migration command provided by the `indico-migrate` package. This document will guide you over the steps needed to perform the upgrade.

Prerequisites

In order to migrate to version 2.0 of Indico you will first of all need to make sure you have **at least version 1.2** of Indico installed. Migration of databases using earlier versions will either **fail** or very likely result in **data loss**. So, please make sure that you are **on 1.2.x** before migrating.

Warning: If you are running a version of the experimental (thus unsupported) **1.9.x branch**, you will have to perform a **step-by-step migration**. We hope that, as advised, no-one upgraded to intermediate 1.9.x releases. If you did and need help with it, please **ping us on IRC**.

Backing up ZODB

The migration script doesn't write to the ZODB, but we still recommend that you **make a backup** just in case:

```
repozo -B -F -r <some-place-safe> -f <indico-db-dir>/Data.fs --verbose
```

You should replace `<some-place-safe>` with the directory in your filesystem where you want to keep the backup. As for `<indico-db-dir>`, that's the directory where the database file is kept. That should be `/opt/indico/db` in most Indico installations.

Make sure that backup files have been created (you should have an `*.index` and an `*.fs` file).

Now, let's shut down the ZEO daemon:

```
zdaemon -C /opt/indico/etc/zdctl.conf stop
```

Double check that the daemon is not running:

```
zdaemon -C /opt/indico/etc/zdctl.conf status
```

Moving legacy data

Indico 2.0 will use a directory structure that is similar to Indico 1.x, so first of all you will need to rename the old tree:

```
mv /opt/indico /opt/indico-legacy
```

Warning: After the migration is done, **do not** delete the `/opt/indico-legacy` directory without first moving the `archive` dir elsewhere. Please read the full guide until the end.

Installing Indico 2.0

The first step should be to have a working Indico 2.0 setup. In order to do that, you should follow the regular Indico 2.x installation instructions up to the “Configure Indico” step. We provide below direct links to the relevant sections of the installation guides.

On a **Debian/Ubuntu** system:

nginx	Apache
1. Install Packages	1. Install Packages
2. Create a Database	2. Create a Database
3. Configure uWSGI & nginx	3. Configure uWSGI & Apache
4. Create an SSL Certificate	4. Create an SSL Certificate
5. Install Indico	5. Install Indico
6. Configure Indico	6. Configure Indico

On a **CentOS7-based** system:

nginx	Apache
1. Enable EPEL	1. Enable EPEL
2. Install Packages	2. Install Packages
3. Create a Database	3. Create a Database
4. Configure uWSGI & nginx	4. Configure uWSGI & Apache
5. Create an SSL Certificate	5. Create an SSL Certificate
6. Configure SELinux	6. Configure SELinux
7. Install Indico	7. Install Indico
8. Configure Indico	8. Configure Indico

Configuration Wizard

You will then need to run the Configuration Wizard, following the normal installation guide (Debian/Ubuntu or CentOS). When the wizard asks you about the “**Old archive dir**”, make sure to set it to the `archive` dir in the `indico-legacy` directory.

```
...
If you are upgrading from Indico 1.2, please specify the path to the
ArchiveDir of the old indico version. Leave this empty if you are not
upgrading.
Old archive dir: /opt/indico-legacy/archive
...
```

Running `indico-migrate`

First of all, make sure that you are using the **user** and **virtualenv** created using the step “**Install Indico**” and that the legacy dir is owned by this **user**:

```
chown -R indico /opt/indico-legacy
su - indico
source ~/.venv/bin/activate
```

You should then install the package using:

```
pip install indico-migrate
```

`indico-migrate` requires a series of parameters that have to be tuned according to your current setup. We now provide a list of values that should work in most standard Indico installations. However, please **carefully read** the [documentation of the `indico-migrate` command](#), to make sure there are no option conflicts with your setup.

Most frequently, `indico-migrate postgresql:///indico file:///opt/indico-legacy/db/Data.fs` will work, followed by the following parameters:

- `--archive-dir /opt/indico-legacy/archive`
- `--storage-backend legacy`
- `--default-email default@<organization-hostname>`
- `--default-currency EUR`
- `--symlink-target ~/archive/legacy_symlinks/`
- `--symlink-backend legacy_symlinks`
- `--migrate-broken-events` (optional - use it if you want to migrate events that don't belong to any category in v1.2. If any such events exist, they will be added to a new category named *Lost & Found*).

(don't forget to replace `<organization-hostname>` with the e-mail hostname of your organization)

An example:

```
indico-migrate postgresql:///indico file:///opt/indico-legacy/db/Data.fs --archive-
↪dir /opt/indico-legacy/archive --storage-backend legacy --default-email_
↪default@acme.example.com --default-currency EUR --symlink-target ~/archive/legacy_
↪symlinks/ --symlink-backend legacy_symlinks --migrate-broken-events
```

Note: If for some reason the migration fails, `indico-migrate` will ask you whether you would like to post an error report on a public pastebin (Gist). The link will not be advertised and only the log information that was shown on screen (plus the exception traceback that was printed) will be included. If you are not comfortable with letting `indico-migrate` post this on a public pastebin, you can always send us your `migration.log` file (which gets generated automatically).

Post-migration work

After the migration is done you may need to apply some adjustments in your `indico.conf`. You may want to read our guide on how to configure an Identity/Authentication provider.

We really recommend as well that you move your old Indico archive (`/opt/indico-legacy/archive`) inside your new Indico directory:

```
mv /opt/indico-legacy/archive /opt/indico/legacy-archive
```

The legacy archive will remain **read-only**. You should update your `indico.conf` (`STORAGE_BACKENDS` option) to reflect the new path:

```
STORAGE_BACKENDS = {  
    # ...  
    'legacy': 'fs-readonly:/opt/indico/legacy-archive'  
    # ...  
}
```

Finishing up

You can now proceed with the remaining installation steps:

On a **Debian/Ubuntu** system:

nginx	Apache
<i>8. Launch Indico</i>	<i>8. Launch Indico</i>
<i>9. Optional: Get a Certificate from Let's Encrypt</i>	<i>9. Optional: Get a Certificate from Let's Encrypt</i>
<i>10. Create an Indico user</i>	<i>10. Create an Indico user</i>

On a **CentOS7-based** system:

nginx	Apache
<i>10. Launch Indico</i>	<i>10. Launch Indico</i>
<i>11. Open the Firewall</i>	<i>11. Open the Firewall</i>
<i>12. Optional: Get a Certificate from Let's Encrypt</i>	<i>12. Optional: Get a Certificate from Let's Encrypt</i>
<i>13. Create an Indico user</i>	<i>13. Create an Indico user</i>

Sanitizing HTML

Indico 2.0 uses [Markdown](#) for the descriptions of contributions and categories. Contribution descriptions that previously contained HTML will still work, but new ones will only support Markdown syntax (including basic HTML). As for the descriptions of categories, they are interpreted as Markdown as of version 2.0, which means that some existing data may be broken. In order to make the lives of users who are migrating easier, we have included with `indico-migrate` a command that automatically performs the migration of Category descriptions to Markdown.

First of all, let's see what would be the impact of running the command:

```
indico-html-sanitize --dry-run -v -l log.html category_descriptions
```

By opening `log.html` you will be able to check if there are any special cases that will need manual intervention. If you're happy with the changes, you can just choose to save them:

```
indico-html-sanitize category_descriptions
```

Removing old data

Even if you're sure the migration succeeded and all data was kept, please keep around the backup of your ZODB you made at the beginning of this guide. **After** and **only after** having **moved the legacy archive** to the new Indico dir and stored a **backup of your ZODB** in a safe place, you can proceed to delete the old `/opt/indico` directory:

```
rm -rf /opt/indico-legacy
```

Installation guide (development)

Todo

Write guide

Server administration

Indico uses [Celery](#) for task management and scheduling and the Livesync Flask plugin for search integration. This part of the documentation explains how to set them up and run automatically.

Server administration

Indico uses [Celery](#) for task management and scheduling and the Livesync Flask plugin for search integration. This part of the documentation explains how to set them up and run automatically.

Celery

Todo

[Complete page](#)

Livesync

Todo

[Complete page](#)

Indico can be extended through plugins, standalone packages of code that do not require any modifications to the Indico core itself. A plugin can perform something very simple such as adding a new command to the Indico CLI to more complex functionalities like introducing new payment methods, chat integration, etc. We suggest that you first have a look at Getting started and then head over to the more advance topics in the table of contents.

Extending Indico with plugins

Indico can be extended through plugins, standalone packages of code that do not require any modifications to the Indico core itself. A plugin can perform something very simple such as adding a new command to the Indico CLI to more complex functionalities like introducing new payment methods, chat integration, etc. We suggest that you first have a look at Getting started and then head over to the more advance topics in the table of contents.

Getting started with Indico plugins

Todo

Write a **REAL**, simple example of a plugin. Include link to Github repo.

Example plugin

The following is a minimal plugin that makes use of all capabilities of the plugin API. The **display name** of the plugin is defined by the first line of the docstring and the **description** by the rest of it. The plugin may override signal handlers to hook into Indico and additionally run any initialization needed. For example, it will add some command to Indico CLI, extend the shell context and register some assets. Also, *init* is used to inject CSS and JS bundles outside of the plugin scope.

```

class ExamplePlugin(IndicoPlugin):
    """Example Plugin

    An example plugin that demonstrates the capabilities of the new Indico plugin_
↪system.
    """

    settings_form = SettingsForm

    def init(self):
        super(ExamplePlugin, self).init()
        self.inject_css('global_css')
        self.inject_js('global_js')

    def get_blueprints(self):
        return blueprint

    def add_cli_command(self, manager):
        @manager.command
        @with_plugin_context(self)
        def example():
            """Example command from example plugin"""
            print 'example plugin says hi', current_plugin
            if self.settings.get('show_message'):
                print self.settings.get('dummy_message')

    def extend_shell_context(self, add_to_context):
        add_to_context('bar', name='foo', doc='foobar from example plugin', color=
↪'magenta!')

    def register_assets(self):
        self.register_js_bundle('example_js', 'js/example.js')
        self.register_js_bundle('global_js', 'js/global.js')
        self.register_css_bundle('example_css', 'css/example.scss')
        self.register_css_bundle('global_css', 'css/global.scss')

```

The plugin can specify its settings via a IndicoForm:

```

class SettingsForm(IndicoForm):
    dummy_message = StringField('Dummy Message')
    show_message = BooleanField('Show Message')

```

The plugin can also specify request handlers and templates. Templates will be loaded from a *templates* folder within your plugin folder. Your plugin can even load templates from other modules by prefixing the name of the template *'other_plugin:example'* with *render_template()*.

```

class WPExample(WPMainBase):
    def _getBody(self, params):
        return render_plugin_template('example.html', **params)

class RHExample(RH):
    def _process(self):
        return WPExample(self, foo=u'bar').display()

class RHTest(RH):
    def _process(self):

```

```
return render_plugin_template('test.html')
```

```
blueprint = IndicoPluginBlueprint('example', __name__)
blueprint.add_url_rule('/example', 'example', view_func=RHExample)
blueprint.add_url_rule('/example/x', 'example', view_func=RHExample)
blueprint.add_url_rule('/test', 'test', view_func=RHTest)
```

Plugin API reference

Indico's plugin system allows you to extend indico with additional modules which can be installed separately and do not require any modifications to the indico core itself.

class `indico.core.plugins.IndicoPlugin` (*plugin_engine, app*)

Bases: `flask_pluginengine.plugin.Plugin`

Base class for an Indico plugin

All your plugins need to inherit from this class. It extends the *Plugin* class from Flask-PluginEngine with useful indico-specific functionality that makes it easier to write custom plugins.

When creating your plugin, the class-level docstring is used to generate the friendly name and description of a plugin. Its first line becomes the name while everything else goes into the description.

This class provides methods for some of the more common hooks Indico provides. Additional signals are defined in `signals` and can be connected to custom functions using `connect()`.

acl_event_settings = frozenset([])

A set containing the names of event-specific settings which store ACLs

acl_settings = frozenset([])

A set containing the names of settings which store ACLs

category = None

The group category that the plugin belongs to

configurable = False

If the plugin should link to a details/config page in the admin interface

default_event_settings = {}

A dictionary containing default values for event-specific settings

default_settings = {}

A dictionary containing default values for settings

default_user_settings = {}

A dictionary containing default values for user-specific settings

event_settings

`classmethod(function) -> method`

Convert a function to be a class method.

A class method receives the class as implicit first argument, just like an instance method receives the instance. To declare a class method, use this idiom:

```
class C: def f(cls, arg1, arg2, ...): ... f = classmethod(f)
```

It can be called either on the class (e.g. `C.f()`) or on an instance (e.g. `C().f()`). The instance is ignored except for its class. If a class method is called for a derived class, the derived class object is passed as the implied first argument.

Class methods are different than C++ or Java static methods. If you want those, see the `staticmethod` builtin.

event_settings_converters = {}

A dict containing custom converters for event-specific settings

get_blueprints ()

Return blueprints to be registered on the application

A single blueprint can be returned directly, for multiple blueprint you need to yield them or return an iterable.

get_vars_js ()

Return a dictionary with variables to be added to vars.js file

init ()

Called when the plugin is being loaded/initialized.

If you want to run custom initialization code, this is the method to override. Make sure to call the base method or the other overridable methods in this class will not be called anymore.

inject_css (*name*, *view_class=None*, *subclasses=True*, *condition=None*)

Injects a CSS bundle into Indico's pages

Parameters

- **name** – Name of the bundle
- **view_class** – If a WP class is specified, only inject it into pages using that class
- **subclasses** – also inject into subclasses of *view_class*
- **condition** – a callable to determine whether to inject or not. only called, when the *view_class* criterion matches

inject_js (*name*, *view_class=None*, *subclasses=True*, *condition=None*)

Injects a JS bundle into Indico's pages

Parameters

- **name** – Name of the bundle
- **view_class** – If a WP class is specified, only inject it into pages using that class
- **subclasses** – also inject into subclasses of *view_class*
- **condition** – a callable to determine whether to inject or not. only called, when the *view_class* criterion matches

inject_vars_js ()

Returns a string that will define variables for the plugin in the vars.js file

register_assets ()

Add assets to the plugin's webassets environment.

In most cases the whole method can consist of calls to `register_js_bundle()` and `register_css_bundle()`.

register_css_bundle (*name*, **files*)

Registers an SCSS bundle in the plugin's webassets environment

register_js_bundle (*name*, **files*)

Registers a JS bundle in the plugin's webassets environment

settings

classmethod(function) -> method

Convert a function to be a class method.

A class method receives the class as implicit first argument, just like an instance method receives the instance. To declare a class method, use this idiom:

```
class C: def f(cls, arg1, arg2, ...): ... f = classmethod(f)
```

It can be called either on the class (e.g. C.f()) or on an instance (e.g. C().f()). The instance is ignored except for its class. If a class method is called for a derived class, the derived class object is passed as the implied first argument.

Class methods are different than C++ or Java static methods. If you want those, see the `staticmethod` builtin.

settings_converters = {}

A dict containing custom converters for settings

settings_form = None

WTForm for the plugin's settings (requires `configurable=True`). All fields must return JSON-serializable types.

settings_form_field_opts = {}

A dictionary which can contain the kwargs for a specific field in the `settings_form`.

strict_settings = True

If `settings`, `event_settings` and `user_settings` should use strict mode, i.e. only allow keys in `default_settings`, `default_event_settings` or `default_user_settings` (or the related `acl_settings` sets). This should not be disabled in most cases; if you need to store arbitrary keys, consider storing a dict inside a single top-level setting.

template_hook (*name, receiver, priority=50, markup=True*)

Registers a function to be called when a template hook is invoked.

For details see `:func:~'indico.web.flask.templating.register_template_hook'`

translation_domain

Return the domain for this plugin's `translation_path`

translation_path

Return translation files to be used by the plugin. By default, get `<root_path>/translations`, unless it does not exist

user_settings

classmethod(function) -> method

Convert a function to be a class method.

A class method receives the class as implicit first argument, just like an instance method receives the instance. To declare a class method, use this idiom:

```
class C: def f(cls, arg1, arg2, ...): ... f = classmethod(f)
```

It can be called either on the class (e.g. C.f()) or on an instance (e.g. C().f()). The instance is ignored except for its class. If a class method is called for a derived class, the derived class object is passed as the implied first argument.

Class methods are different than C++ or Java static methods. If you want those, see the `staticmethod` builtin.

user_settings_converters = {}

A dict containing custom converters for user-specific settings

```
class indico.core.plugins.IndicoPluginBlueprint (name, *args, **kwargs)  
    Bases: flask_pluginengine.mixins.PluginBlueprintMixin, indico.web.flask.  
           wrappers.IndicoBlueprint
```

The Blueprint class all plugins need to use.

It contains the necessary logic to run the blueprint's view functions inside the correct plugin context and to make the static folder work.

```
indico.core.plugins.get_plugin_template_module (template_name, **context)  
    Like get_template_module(), but using plugin templates
```

```
indico.core.plugins.include_plugin_css_assets (bundle_name)  
    Jinja template function to generate HTML tags for a plugin CSS asset bundle.
```

```
indico.core.plugins.include_plugin_js_assets (bundle_name)  
    Jinja template function to generate HTML tags for a plugin JS asset bundle.
```

```
indico.core.plugins.plugin_url_rule_to_js (endpoint)  
    Like url_rule_to_js() but prepending plugin name prefix to the endpoint
```

```
indico.core.plugins.url_for_plugin (endpoint, *targets, **values)  
    Like url_for() but prepending 'plugin_' to the blueprint name.
```

Hooking into Indico using Signals

Contents

- *Hooking into Indico using Signals*
 - *indico.core.signals*
 - * *indico.core.signals.acl*
 - * *indico.core.signals.agreements*
 - * *indico.core.signals.attachments*
 - * *indico.core.signals.category*
 - * *indico.core.signals.event*
 - * *indico.core.signals.event_management*
 - * *indico.core.signals.menu*
 - * *indico.core.signals.plugin*
 - * *indico.core.signals.users*

Signals allow you to hook into certain parts of Indico without adding any code to the core (which is something a plugin can and should not do). Each signal has a *sender* which can be any object (depending on the signal) and possibly some keyword arguments. Some signals also make use of their return value or even require one. Check the documentation of each signal on how it's used.

To avoid breakage with newer versions of Indico, it is highly advised to always accept extra `**kwargs` in your signal receiver. For example, a receiver function could look like this:

```
def receiver(sender, something, **kwargs):  
    do_stuff_with(something)
```

indico.core.signals

`indico.core.signals.add_form_fields`

Lets you add extra fields to a form. The *sender* is the form class and should always be specified when subscribing to this signal.

The signal handler should return one or more `'name', Field` tuples. Each field will be added to the form as `ext__<name>` and is automatically excluded from the form's *data* property and its *populate_obj* method.

To actually process the data, you can use e.g. the *form_validated* signal and then store it in *flask.g* until another signal informs you that the operation the user was performing has been successful.

`indico.core.signals.after_process`

Called after an Indico request has been processed.

`indico.core.signals.app_created`

Called when the app has been created. The *sender* is the flask app.

`indico.core.signals.db_schema_created`

Executed when a new database schema is created. The *sender* is the name of the schema.

`indico.core.signals.form_validated`

Triggered when an IndicoForm was validated successfully. The *sender* is the form object.

This signal may return `False` to mark the form as invalid even though WTForms validation was successful. In this case it is highly recommended to mark a field as erroneous or indicate the error in some other way.

`indico.core.signals.get_conditions`

Expected to return one or more classes inheriting from *Condition*. The *sender* is a string (or some other object) identifying the context. The additional kwargs passed to this signal depend on the context.

`indico.core.signals.get_fields`

Expected to return *BaseField* subclasses. The *sender* is an object (or just a string) identifying for what to get fields. This signal should never be registered without restricting the sender to ensure only the correct field types are returned.

`indico.core.signals.get_placeholders`

Expected to return one or more *Placeholder* objects. The *sender* is a string (or some other object) identifying the context. The additional kwargs passed to this signal depend on the context.

`indico.core.signals.get_storage_backends`

Expected to return one or more *Storage* subclasses.

`indico.core.signals.import_tasks`

Called when Celery needs to import all tasks. Use this signal if you have modules containing task registered using one of the Celery decorators but don't import them anywhere. The signal handler should only `import` these modules and do nothing else.

`indico.core.signals.model_committed`

Triggered when an IndicoModel class was committed. The *sender* is the model class, the model instance is passed as *obj* and the change type as a string (delete/insert/update) in the *change* kwarg.

indico.core.signals.acl

`indico.core.signals.acl.can_access`

Called when *ProtectionMixin.can_access* is used to determine if a user can access something or not.

The *sender* is the type of the object that's using the mixin. The actual instance is passed as *obj*. The *user* and *allow_admin* arguments of *can_access* are passed as kwargs with the same name.

The *authorized* argument is `None` when this signal is called at the beginning of the access check and `True` or `False` at the end when regular access rights have already been checked. For expensive checks (such as anything involving database queries) it is recommended to skip the check while *authorized* is `None` since the regular access check is likely to be cheaper (due to ACLs being preloaded etc).

If the signal returns `True` or `False`, the access check succeeds or fails immediately. If multiple subscribers to the signal return contradictory results, `False` wins and access is denied.

`indico.core.signals.acl.can_manage`

Called when *ProtectionMixin.can_manage* is used to determine if a user can manage something or not.

The *sender* is the type of the object that's using the mixin. The actual instance is passed as *obj*. The *user*, *role*, *allow_admin*, *check_parent* and *explicit_role* arguments of *can_manage* are passed as kwargs with the same name.

If the signal returns `True` or `False`, the access check succeeds or fails without any further checks. If multiple subscribers to the signal return contradictory results, `False` wins and access is denied.

`indico.core.signals.acl.entry_changed`

Called when an ACL entry is changed.

The *sender* is the type of the object that's using the mixin. The actual instance is passed as *obj*. The *User*, *GroupProxy* or *EmailPrincipal* is passed as *principal* and *entry* contains the actual ACL entry (a *PrincipalMixin* instance) or `None` in case the entry was deleted. *is_new* is a boolean indicating whether the given principal was in the ACL before. If *quiet* is `True`, signal handlers should not perform noisy actions such as logging or sending emails related to the change.

If the ACL uses roles, *old_data* will contain a dictionary of the previous roles/permissions (see *PrincipalRolesMixin.current_data*).

`indico.core.signals.acl.get_management_roles`

Expected to return *ManagementRole* subclasses. The *sender* is the type of the object the roles may be used for. Functions subscribing to this signal **MUST** check the sender by specifying it using the first argument of *connect_via()* or by comparing it inside the function.

`indico.core.signals.acl.protection_changed`

Called when the protection mode of an object is changed.

The *sender* is the type of the object that's using the mixin. The actual instance is passed as *obj*. The old protection mode is passed as *old_mode*, the new mode as *mode*.

indico.core.signals.agreements

`indico.core.signals.agreements.get_definitions`

Expected to return a list of *AgreementDefinition* classes.

indico.core.signals.attachments

`indico.core.signals.attachments.attachment_accessed`

Called when an attachment is accessed. The *sender* is the *Attachment* that was accessed. The user who accessed the attachment is passed in the *user* kwarg. The *from_preview* kwarg will be set to `True` if the download link on the preview page was used to access the attachment or if the attachment was loaded to be displayed on the preview page (opening the preview itself already sends this signal with *from_preview=False*).

`indico.core.signals.attachments.attachment_created`

Called when a new attachment is created. The *sender* object is the new *Attachment*. The user who created the attachment is passed in the *user* kwarg.

indico.core.signals.attachments.attachment_deleted

Called when an attachment is deleted. The *sender* object is the *Attachment* that was deleted. The user who deleted the attachment is passed in the *user* kwarg.

indico.core.signals.attachments.attachment_updated

Called when an attachment is updated. The *sender* is the *Attachment* that was updated. The user who updated the attachment is passed in the *user* kwarg.

indico.core.signals.attachments.folder_created

Called when a new attachment folder is created. The *sender* is the new *AttachmentFolder* object. The user who created the folder is passed in the *user* kwarg. This signal is never triggered for the internal default folder.

indico.core.signals.attachments.folder_deleted

Called when a folder is deleted. The *sender* is the *AttachmentFolder* that was deleted. The user who deleted the folder is passed in the *user* kwarg.

indico.core.signals.attachments.folder_updated

Called when a folder is updated. The *sender* is the *AttachmentFolder* that was updated. The user who updated the folder is passed in the *user* kwarg.

indico.core.signals.attachments.get_file_previewers

Expected to return one or more *Previewer* subclasses.

indico.core.signals.category**indico.core.signals.category.created**

Called when a new category is created. The *sender* is the new category.

indico.core.signals.category.deleted

Called when a category is deleted. The *sender* is the category.

indico.core.signals.category.moved

Called when a category is moved into another category. The *sender* is the category and the old parent category is passed in the *old_parent* kwarg.

indico.core.signals.category.updated

Called when a new category is created. The *sender* is the new category.

indico.core.signals.event**indico.core.signals.event.abstract_created**

Called when a new abstract is created. The *sender* is the new abstract.

indico.core.signals.event.abstract_deleted

Called when an abstract is deleted. The *sender* is the abstract.

indico.core.signals.event.abstract_state_changed

Called when an abstract is withdrawn. The *sender* is the abstract.

indico.core.signals.event.abstract_updated

Called when an abstract is modified. The *sender* is the abstract.

indico.core.signals.event.cloned

Called when an event is cloned. The *sender* is the *Event* object of the old event, the new event is passed in the *new_event* kwarg.

indico.core.signals.event.contribution_created

Called when a new contribution is created. The *sender* is the new contribution.

`indico.core.signals.event.contribution_deleted`
Called when a contribution is deleted. The *sender* is the contribution.

`indico.core.signals.event.contribution_updated`
Called when a contribution is modified. The *sender* is the contribution. A dict containing `old`, `new` tuples for all changed values is passed in the `changes` kwarg.

`indico.core.signals.event.created`
Called when a new event is created. The *sender* is the new Event.

`indico.core.signals.event.deleted`
Called when an event is deleted. The *sender* is the event object. The *user* kwarg contains the user performing the deletion if available.

`indico.core.signals.event.generate_ticket_qr_code`
Called when generating the QR code for a ticket. The data included in the QR code is passed in the *ticket_data* kwarg and may be modified.

`indico.core.signals.event.get_feature_definitions`
Expected to return *EventFeature* subclasses.

`indico.core.signals.event.get_log_renderers`
Expected to return *EventLogRenderer* classes.

`indico.core.signals.event.is_ticketing_handled`
Called when resolving whether Indico should send tickets with e-mails or it will be handled by other module. The *sender* is the *RegistrationForm* object.

Should return `True` or `False`.

`indico.core.signals.event.moved`
Called when an event is moved to a different category. The *sender* is the event, the old category is in the *old_parent* kwarg.

`indico.core.signals.event.note_added`
Called when a note is added. The *sender* is the note.

`indico.core.signals.event.note_deleted`
Called when a note is deleted. The *sender* is the note.

`indico.core.signals.event.note_modified`
Called when a note is modified. The *sender* is the note.

`indico.core.signals.event.person_updated`
Called when an *EventPerson* is modified. The *sender* is the *EventPerson*.

`indico.core.signals.event.print_badge_template`
Called when printing a badge template. The registration form is passed in the *regform* kwarg.

`indico.core.signals.event.registration_deleted`
Called when a registration is removed. The *sender* is the *Registration* object.

`indico.core.signals.event.registration_form_created`
Called when a new registration form is created. The *sender* is the *RegistrationForm* object.

`indico.core.signals.event.registration_form_deleted`
Called when a registration form is removed. The *sender* is the *RegistrationForm* object.

`indico.core.signals.event.registration_personal_data_modified`
Called when the registration personal data is modified. The *sender* is the *Registration* object; the change is passed in the *change* kwarg.

- `indico.core.signals.event.registration_state_updated`
Called when the state of registration changes. The *sender* is the *Registration* object; the previous state is passed in the *previous_state* kwarg.
- `indico.core.signals.event.session_block_deleted`
Called when a session block is deleted. The *sender* is the session block. This signal is called before the `db.session.delete()` on the block is executed.
- `indico.core.signals.event.session_deleted`
Called when a session is deleted. The *sender* is the session.
- `indico.core.signals.event.session_updated`
Called when a session is updated. The *sender* is the session.
- `indico.core.signals.event.sidemenu`
Expected to return `MenuEntryData` objects to be added to the event side menu. A single entry can be returned directly, multiple entries must be yielded.
- `indico.core.signals.event.subcontribution_created`
Called when a new subcontribution is created. The *sender* is the new subcontribution.
- `indico.core.signals.event.subcontribution_deleted`
Called when a subcontribution is deleted. The *sender* is the subcontribution.
- `indico.core.signals.event.subcontribution_updated`
Called when a subcontribution is modified. The *sender* is the subcontribution.
- `indico.core.signals.event.times_changed`
Called when the times of a scheduled object (contribution, break or session block) change, either by a change in duration or start time. The *sender* is the type of the object; the timetable entry is passed as *entry* and the object is passed as *obj*. Information about the changes are passed as *changes* which is a dict containing old/new tuples for *start_dt*, *duration* and *end_dt*. If an attribute did not change, it is not included in the dict. If the time of the event itself changes, *entry* is `None` and *obj* contains the *Event*.
- `indico.core.signals.event.timetable_buttons`
Expected to return a list of tuples ('button_name', 'js-call-class'). Called when building the timetable view.
- `indico.core.signals.event.timetable_entry_created`
Called when a new timetable entry is created. The *sender* is the new entry.
- `indico.core.signals.event.timetable_entry_deleted`
Called when a timetable entry is deleted. The *sender* is the entry. This signal is triggered right before the entry deletion is performed.
- `indico.core.signals.event.timetable_entry_updated`
Called when a timetable entry is updated. The *sender* is the entry. A dict containing `old`, `new` tuples for all changed values is passed in the *changes* kwarg.
- `indico.core.signals.event.type_changed`
Called when the type of an event is changed. The *sender* is the event, the old type is passed in the *old_type* kwarg.
- `indico.core.signals.event.updated`
Called when basic data of an event is updated. The *sender* is the event. A dict of changes is passed in the *changes* kwarg, with (`old`, `new`) tuples for each change. Note that the *person_links* change may happen with *old* and *new* being the same lists for technical reasons. If the key is present, it should be assumed that something changed (usually the order or some data on the person link).

indico.core.signals.event_management

indico.core.signals.event_management.get_cloners

Expected to return one or more `EventCloner` subclasses implementing a cloning operation for something within an event.

indico.core.signals.event_management.image_created

Called when a new image is created. The *sender* object is the new `ImageFile`. The user who uploaded the image is passed in the `user` kwarg.

indico.core.signals.event_management.image_deleted

Called when an image is deleted. The *sender* object is the `ImageFile` that is about to be deleted. The user who uploaded the image is passed in the `user` kwarg.

indico.core.signals.event_management.management_url

Expected to return a URL for the event management page of the plugin. This is used when someone who does not have event management access wants to go to the event management area. He is then redirected to one of the URLs returned by plugins, i.e. it is not guaranteed that the user ends up on a specific plugin's management page. The signal should return `None` if the current user (available via `session.user`) cannot access the management area. The *sender* is the event object.

indico.core.signals.menu

indico.core.signals.menu.items

Expected to return one or more `SideMenuItem` to be added to the side menu. The *sender* is an id string identifying the target menu.

indico.core.signals.menu.sections

Expected to return one or more `SideMenuSection` objects to be added to the side menu. The *sender* is an id string identifying the target menu.

indico.core.signals.plugin

indico.core.signals.plugin.cli

Expected to return one or more click commands/groups. If they use `indico.cli.core.cli_command` / `indico.cli.core.cli_group` they will be automatically executed within a plugin context and run within a Flask app context by default.

indico.core.signals.plugin.get_blueprints

Expected to return one or more `IndicoPluginBlueprint`-based blueprints which will be registered on the application. The Blueprint must be named either `PLUGINNAME` or `compat_PLUGINNAME`.

indico.core.signals.plugin.get_conference_themes

Expected to return `(name, css, title)` tuples for conference stylesheets. `name` is the internal name used for the stylesheet which will be stored when the theme is selected in an event. `css` is the location of the CSS file, relative to the plugin's `static` folder. `title` is the title displayed to the user when selecting the theme.

indico.core.signals.plugin.get_event_request_definitions

Expected to return one or more `RequestDefinition` subclasses.

indico.core.signals.plugin.get_event_themes_files

Expected to return the path of a `themes.yaml` containing event theme definitions.

indico.core.signals.plugin.inject_css

Expected to return a list of CSS URLs which are loaded after all other CSS. The *sender* is the WP class of the page.

`indico.core.signals.plugin.inject_js`

Expected to return a list of JS URLs which are loaded after all other JS. The *sender* is the WP class of the page.

`indico.core.signals.plugin.shell_context`

Called after adding stuff to the *indico shell* context. Receives the *add_to_context* and *add_to_context_multi* keyword args with functions which allow you to add custom items to the context.

`indico.core.signals.plugin.template_hook`

Expected to return a (*is_markup*, *priority*, *value*) tuple. The returned value will be inserted at the location where this signal is triggered; if multiple receivers are connected to the signal, they will be ordered by priority. If *is_markup* is True, the value will be wrapped in a *Markup* object which will cause it to be rendered as HTML. The *sender* is the name of the actual hook. The keyword arguments depend on the hook.

indico.core.signals.users

`indico.core.signals.users.email_added`

Called when a new email address is added to a user. The *sender* is the user object and the email address is passed in the *email* kwarg.

`indico.core.signals.users.merged`

Called when two users are merged. The *sender* is the main user while the merged user (i.e. the one being deleted in the merge) is passed via the *source* kwarg.

`indico.core.signals.users.preferences`

Expected to return a *ExtraUserPreferences* subclass which implements extra preferences for the user preference page. The *sender* is the user for whom the preferences page is being shown which might not be the currently logged-in user!

`indico.core.signals.users.registered`

Called once a user registers (either locally or joins through a provider). The *sender* is the new user object. The kwarg *from_moderation* indicates whether the user went through a moderation process (this also includes users created by an administrator manually) or was created immediately on registration; the identity associated with the registration is passed in the *identity* kwarg.

`indico.core.signals.users.registration_requested`

Called when a user requests to register a new indico account, i.e. if moderation is enabled. The *sender* is the registration request.

Adding models to your plugin

Plugins must describe its database model the in the *models* folder if needed:

```
class Foo(db.Model):
    __tablename__ = 'foo'
    __table_args__ = {'schema': 'plugin_example'}

    id = db.Column(
        db.Integer,
        primary_key=True
    )
    bar = db.Column(
        db.String,
        nullable=False,
        default=''
    )
    location_id = db.Column(
```

```
        db.Integer,
        db.ForeignKey('roombooking.locations.id'),
        nullable=False
    )
    location = db.relationship(
        'Location',
        backref=db.backref('example_foo', cascade='all, delete-orphan', lazy='dynamic
↪'),
    )

    @return_ascii
    def __repr__(self):
        return u'<Foo({}, {}, {})>'.format(self.id, self.bar, self.location)
```

Thanks to **Alembic**, the migration needed to create the tables in the database can also be included in the plugin. The steps to do so are:

1. Create a revision for the changes your plugin will add with `indico db --plugin example migrate -m 'short description'`
2. Fine-tune the revision file generated under *migrations*.
3. Run `indico db --plugin example upgrade` to have Alembic upgrade your DB with the changes.

Indico allows you to programmatically access the content of its database by exposing various information like category contents, events, rooms and room bookings through a web service, the HTTP Export API.

Indico - HTTP API

Indico allows you to programmatically access the content of its database by exposing various information like category contents, events, rooms and room bookings through a web service, the HTTP Export API.

Accessing the API

URL structure

Indico allows you to programmatically access the content of its database by exposing various information like category contents, events, rooms and room bookings through a web service, the HTTP Export API.

The basic URL looks like:

[http://my.indico.server/export/WHAT/\[\]LOC/\[\]ID.TYPE?PARAMS&ak=KEY×tamp=TS&signature=SIG](http://my.indico.server/export/WHAT/[]LOC/[]ID.TYPE?PARAMS&ak=KEY×tamp=TS&signature=SIG)

where:

- *WHAT* is the element you want to export (one of *categ*, *event*, *room*, *reservation*)
- *LOC* is the location of the element(s) specified by *ID* and only used for certain elements, for example, for the room booking (<https://indico.server/export/room/CERN/120.json?ak=0...>)
- *ID* is the ID of the element you want to export (can be a - separated list). As for example, the 120 in the above URL.
- *TYPE* is the output format (one of *json*, *jsonp*, *xml*, *html*, *ics*, *atom*, *bin*)
- *PARAMS* are various parameters affecting (filtering, sorting, ...) the result list
- *KEY*, *TS*, *SIG* are part of the *API Authentication*.

Some examples could be:

- Export data about events in a category: <https://my.indico/export/categ/2.json?from=today&to=today&pretty=yes>
- Export data about a event: <https://indico.server/export/event/137346.json?occ=yes&pretty=yes>
- Export data about rooms: <https://indico.server/export/room/CERN/120.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes>
- Export your reservations: <https://indico.server/export/reservation/CERN.json?ak=00000000-0000-0000-0000-000000000000&detail=reservations&from=today&to=today&bookedfor=USERNAME&pretty=yes>

See more details about querying in Exporters.

API Authentication

General

The HTTP Export API uses an API key and - depending on the config - a cryptographic signature for each request.

To create an API key, go to *My Profile* » *HTTP API* and click the *Create API key* button. This will create an *API Key* and a *Secret Key* (if signatures are required).

It is recommended to always use the highest security level. That means if only an *API key* is available always include it and if a *secret key* is available, always sign your requests. Since you might want to retrieve only public information (instead of everything visible to your Indico user) you can add the param *onlypublic=yes* to the query string.

It is also possible to re-use the existing Indico session. This only makes sense if your browser accesses the API, e.g. because you are developing on Indico and want to access the API via an AJAX request. Additionally this method of authentication is restricted to GET requests. To use it, add *cookieauth=yes* to the query string and do not specify an API key, timestamp or signature. To prevent data leakage via CSRF the CSRF token of the current session needs to be provided as a GET argument *csrftoken* or a HTTP header *X-CSRF-Token*.

Request Signing

To sign a request, you need the following:

- The requested path, e.g. */export/categ/123.json*
 - Any additional params, e.g. *limit=10*
 - The current UNIX timestamp
 - Your *API key* and *secret key*
1. Add your API key to the params (*limit=10&ak=your-api-key*)
 2. Add the current timestamp to the params (*limit=10&ak=your-api-key×tamp=1234567890*)
 3. Sort the query string params (*ak=your-api-key&limit=10×tamp=1234567890*)
 4. Merge path and the sorted query string to a single string (*/export/categ/123.json?ak=your-api-key&limit=10×tamp=1234567890*)
 5. Create a HMAC-SHA1 signature of this string using your *secret key* as the key.
 6. Append the hex-encoded signature to your query string: *?ak=your-api-key&limit=10×tamp=1234567890&signature=your-signature*

Note that a signed request might be valid only for a few seconds or minutes, so you **need** to sign it right before sending it and not store the generated URL as it is likely to expire soon.

You can find example code for Python and PHP in the following sections.

If persistent signatures are enabled, you can also omit the timestamp. In this case the URL is valid forever. When using this feature, please make sure to use these URLs only where necessary - use timestamped URLs whenever possible.

Request Signing for Python

A simple example in Python:

```
import hashlib
import hmac
import urllib
import time

def build_indico_request(path, params, api_key=None, secret_key=None, only_
↳public=False, persistent=False):
    items = params.items() if hasattr(params, 'items') else list(params)
    if api_key:
        items.append(('apikey', api_key))
    if only_public:
        items.append(('onlypublic', 'yes'))
    if secret_key:
        if not persistent:
            items.append(('timestamp', str(int(time.time()))))
        items = sorted(items, key=lambda x: x[0].lower())
        url = '%s?%s' % (path, urllib.urlencode(items))
        signature = hmac.new(secret_key, url, hashlib.sha1).hexdigest()
        items.append(('signature', signature))
    if not items:
        return path
    return '%s?%s' % (path, urllib.urlencode(items))

if __name__ == '__main__':
    API_KEY = '00000000-0000-0000-0000-000000000000'
    SECRET_KEY = '00000000-0000-0000-0000-000000000000'
    PATH = '/export/categ/1337.json'
    PARAMS = {
        'limit': 123
    }
    print build_indico_request(PATH, PARAMS, API_KEY, SECRET_KEY)
```

Request Signing for PHP

A simple example in PHP:

```
<?php

function build_indico_request($path, $params, $api_key = null, $secret_key = null,
↳$only_public = false, $persistent = false) {
    if($api_key) {
        $params['apikey'] = $api_key;
```

```
}

if($only_public) {
    $params['onlypublic'] = 'yes';
}

if($secret_key) {
    if(!$persistent) {
        $params['timestamp'] = time();
    }
    uksort($params, 'strcasecmp');
    $url = $path . '?' . http_build_query($params);
    $params['signature'] = hash_hmac('sha1', $url, $secret_key);
}

if(!$params) {
    return $path;
}

return $path . '?' . http_build_query($params);
}

if(true){ // change to false if you want to include this file
    $API_KEY = '00000000-0000-0000-0000-000000000000';
    $SECRET_KEY = '00000000-0000-0000-0000-000000000000';
    $PATH = '/export/categ/1337.json';
    $PARAMS = array(
        'limit' => 123
    );
    echo build_indico_request($PATH, $PARAMS, $API_KEY, $SECRET_KEY) . "\n";
}
```

Common Parameters

The following parameters are valid for all requests no matter which element is requested. If a parameter has a shorter form, it's given in parentheses.

Param	Short	Description
from/to	f/t	<p>Accepted formats:</p> <ul style="list-style-type: none"> • ISO 8601 subset - YYYY-MM-DD[THH:MM] • 'today', 'yesterday', 'tomorrow' and 'now' • days in the future/past: '[+/-]DdHHhMMm'
pretty	p	Pretty-print the output. When exporting as JSON it will include whitespace to make the json more human-readable.
onlypublic	op	Only return results visible to unauthenticated users when set to <i>yes</i> .
onlyauthed	oa	Fail if the request is unauthenticated for any reason when this is set to <i>yes</i> .
cookieauth	ca	Use the Indico session cookie to authenticate instead of an API key.
limit	n	Return no more than the X results.
offset	O	Skip the first X results.
detail	d	Specify the detail level (values depend on the exported element)
order	o	Sort the results. Must be one of <i>id</i> , <i>start</i> , <i>end</i> , <i>title</i> .
descending	c	Sort the results in descending order when set to <i>yes</i> .
tz	-	Assume given timezone (default UTC) for specified dates. Example: Europe/Lisbon.

API Resources

Categories

URL Format

/export/categ/ID.TYPE

The ID can be either a single category ID or a - separated list. In an authenticated request the special ID *favorites* will be resolved to the user's list of favorites.

Parameters

Param	Short	Description
location	l	Only include events taking place at the specified location. The * and ? wildcards may be used.
room	r	Only include events taking place in the specified room. The * and ? wildcards may be used.
type	T	Only include events of the specified type. Must be one of: simple_event (or lecture), meeting, conference

Detail Levels

events

Returns basic data about the events in the category.

This is the result of the following the query <https://my.indico/export/categ/2.json?from=today&to=today&pretty=yes>:

```
{
  "count": 2,
  "_type": "HTTPAPIResult",
  "complete": true,
  "url": "https://my.indico/export/categ/2.json?from=today&to=today&pretty=yes",
  "ts": 1308841641,
  "results": [
    {
      "category": "TEST Category",
      "startDate": {
        "date": "2011-06-17",
        "tz": "Europe/Zurich",
        "time": "08:00:00"
      },
      "_type": "Conference",
      "endDate": {
        "date": "2011-06-30",
        "tz": "Europe/Zurich",
        "time": "18:00:00"
      },
      "description": "",
      "title": "Test EPayment",
      "url": "http://pcituds07.cern.ch/indico/conferenceDisplay.py?confId=137344",
      "location": "CERN",
      "_fossil": "conferenceMetadata",
      "timezone": "Europe/Zurich",
      "type": "conference",
      "id": "137344",
      "room": "1-1-025"
    },
    {
      "category": "TEST Category",
      "startDate": {
        "date": "2011-06-23",
        "tz": "Europe/Zurich",
        "time": "08:00:00"
      },
      "_type": "Conference",
```

```

        "endDate": {
            "date": "2011-06-24",
            "tz": "Europe/Zurich",
            "time": "18:00:00"
        },
        "description": "",
        "title": "Export Test",
        "url": "http://pcituds07.cern.ch/indico/conferenceDisplay.py?confId=137346",
        "location": "CERN",
        "_fossil": "conferenceMetadata",
        "timezone": "Europe/Zurich",
        "type": "meeting",
        "id": "137346",
        "room": null
    }
]
}

```

Events

URL Format

/export/event/ID.TYPE

The ID can be either a single event ID or a - separated list.

Parameters

Param	Short	Description
occurrences	occ	Include the daily event times in the exported data.

Detail Levels

events

Returns basic data about the event. In this example occurrences are included, too.

Result for <https://indico.server/export/event/137346.json?occ=yes&pretty=yes>:

```

{
  "count": 1,
  "_type": "HTTPAPIResult",
  "complete": true,
  "url": "https://indico.server/export/event/137346.json?occ=yes&pretty=yes",
  "ts": 1308899256,
  "results": [
    {
      "category": "TEST Category",
      "startDate": {
        "date": "2011-06-23",
        "tz": "Europe/Zurich",
        "time": "08:00:00"
      }
    }
  ]
}

```

```
    },
    "_type": "Conference",
    "endDate": {
      "date": "2011-06-24",
      "tz": "Europe/Zurich",
      "time": "18:00:00"
    },
    "description": "",
    "title": "Export Test",
    "url": "http://indico.server/conferenceDisplay.py?confId=137346",
    "room": null,
    "occurrences": [
      {
        "_fossil": "period",
        "endDT": {
          "date": "2011-06-23",
          "tz": "Europe/Zurich",
          "time": "08:40:00"
        },
        "startDT": {
          "date": "2011-06-23",
          "tz": "Europe/Zurich",
          "time": "08:00:00"
        },
        "_type": "Period"
      },
      {
        "_fossil": "period",
        "endDT": {
          "date": "2011-06-24",
          "tz": "Europe/Zurich",
          "time": "15:00:00"
        },
        "startDT": {
          "date": "2011-06-24",
          "tz": "Europe/Zurich",
          "time": "12:00:00"
        },
        "_type": "Period"
      }
    ],
    "_fossil": "conferenceMetadata",
    "timezone": "Europe/Zurich",
    "type": "meeting",
    "id": "137346",
    "location": "CERN"
  }
]
```

contributions

Includes the contributions of the event.

Output for <https://indico.server/export/event/137346.json?detail=contributions&pretty=yes>:

```

{
  "count": 1,
  "_type": "HTTPAPIResult",
  "complete": true,
  "url": "https://indico.server/export/event/137346.json?detail=contributions&
↳pretty=yes",
  "ts": 1308899252,
  "results": [
    {
      "category": "TEST Category",
      "startDate": {
        "date": "2011-06-23",
        "tz": "Europe/Zurich",
        "time": "08:00:00"
      },
      "_type": "Conference",
      "endDate": {
        "date": "2011-06-24",
        "tz": "Europe/Zurich",
        "time": "18:00:00"
      },
      "description": "",
      "title": "Export Test",
      "url": "http://indico.server/conferenceDisplay.py?confId=137346",
      "type": "meeting",
      "location": "CERN",
      "_fossil": "conferenceMetadataWithContribs",
      "timezone": "Europe/Zurich",
      "contributions": [
        {
          "startDate": {
            "date": "2011-06-23",
            "tz": "Europe/Zurich",
            "time": "08:20:00"
          },
          "_type": "Contribution",
          "endDate": {
            "date": "2011-06-23",
            "tz": "Europe/Zurich",
            "time": "08:40:00"
          },
          "description": "",
          "title": "dlc2",
          "track": null,
          "duration": 20,
          "session": null,
          "location": "CERN",
          "_fossil": "contributionMetadata",
          "type": null,
          "id": "1",
          "room": null
        },
        {
          "startDate": {
            "date": "2011-06-23",
            "tz": "Europe/Zurich",
            "time": "08:00:00"
          },

```

```
    "_type": "Contribution",
    "endDate": {
      "date": "2011-06-23",
      "tz": "Europe/Zurich",
      "time": "08:20:00"
    },
    "description": "",
    "title": "d1c1",
    "track": null,
    "duration": 20,
    "session": null,
    "location": "CERN",
    "_fossil": "contributionMetadata",
    "type": null,
    "id": "0",
    "room": null
  },
  {
    "startDate": {
      "date": "2011-06-24",
      "tz": "Europe/Zurich",
      "time": "14:00:00"
    },
    "_type": "Contribution",
    "endDate": {
      "date": "2011-06-24",
      "tz": "Europe/Zurich",
      "time": "14:20:00"
    },
    "description": "",
    "title": "d2slc1",
    "track": null,
    "duration": 20,
    "session": "d2s1",
    "location": "CERN",
    "_fossil": "contributionMetadata",
    "type": null,
    "id": "3",
    "room": null
  },
  {
    "startDate": {
      "date": "2011-06-24",
      "tz": "Europe/Zurich",
      "time": "12:00:00"
    },
    "_type": "Contribution",
    "endDate": {
      "date": "2011-06-24",
      "tz": "Europe/Zurich",
      "time": "14:00:00"
    },
    "description": "",
    "title": "d2c1",
    "track": null,
    "duration": 120,
    "session": null,
    "location": "CERN",
```



```

        "_fossil": "contributionMetadata",
        "type": null,
        "id": "2",
        "room": null
    }
],
"id": "137346",
"room": null
}
]
}

```

subcontributions

Like *contributions*, but inside the contributions the subcontributions are included in a field named *subContributions*.

sessions

Includes details about the different sessions and groups contributions by sessions. The top-level *contributions* list only contains contributions which are not assigned to any session. Subcontributions are included in this details level, too.

For example, <https://indico.server/export/event/137346.json?detail=sessions&pretty=yes>:

```

{
  "count": 1,
  "_type": "HTTPAPIResult",
  "complete": true,
  "url": "https://indico.server/export/event/137346.json?detail=sessions&pretty=yes",
  "ts": 1308899771,
  "results": [
    {
      "category": "TEST Category",
      "startDate": {
        "date": "2011-06-23",
        "tz": "Europe/Zurich",
        "time": "08:00:00"
      },
      "_type": "Conference",
      "endDate": {
        "date": "2011-06-24",
        "tz": "Europe/Zurich",
        "time": "18:00:00"
      },
      "description": "",
      "title": "Export Test",
      "url": "http://indico.server/conferenceDisplay.py?confId=137346",
      "contributions": [
        {
          "startDate": {
            "date": "2011-06-23",
            "tz": "Europe/Zurich",
            "time": "08:20:00"
          },
          "_type": "Contribution",

```

```
    "endDate": {
      "date": "2011-06-23",
      "tz": "Europe/Zurich",
      "time": "08:40:00"
    },
    "description": "",
    "subContributions": [],
    "title": "d1c2",
    "track": null,
    "duration": 20,
    "session": null,
    "location": "CERN",
    "_fossil": "contributionMetadataWithSubContribs",
    "type": null,
    "id": "1",
    "room": null
  },
  {
    "startDate": {
      "date": "2011-06-23",
      "tz": "Europe/Zurich",
      "time": "08:00:00"
    },
    "_type": "Contribution",
    "endDate": {
      "date": "2011-06-23",
      "tz": "Europe/Zurich",
      "time": "08:20:00"
    },
    "description": "",
    "subContributions": [],
    "title": "d1c1",
    "track": null,
    "duration": 20,
    "session": null,
    "location": "CERN",
    "_fossil": "contributionMetadataWithSubContribs",
    "type": null,
    "id": "0",
    "room": null
  },
  {
    "startDate": {
      "date": "2011-06-24",
      "tz": "Europe/Zurich",
      "time": "12:00:00"
    },
    "_type": "Contribution",
    "endDate": {
      "date": "2011-06-24",
      "tz": "Europe/Zurich",
      "time": "14:00:00"
    },
    "description": "",
    "subContributions": [],
    "title": "d2c1",
    "track": null,
    "duration": 120,
```

```

        "session": null,
        "location": "CERN",
        "_fossil": "contributionMetadataWithSubContribs",
        "type": null,
        "id": "2",
        "room": null
    }
],
"sessions": [
    {
        "startDate": {
            "date": "2011-06-24",
            "tz": "Europe/Zurich",
            "time": "14:00:00"
        },
        "_type": "Session",
        "room": "",
        "numSlots": 1,
        "color": "#EEE0EF",
        "material": [],
        "isPoster": false,
        "sessionConveners": [],
        "location": "CERN",
        "address": "",
        "_fossil": "sessionMetadata",
        "title": "d2s1",
        "textColor": "#1D041F",
        "contributions": [
            {
                "startDate": {
                    "date": "2011-06-24",
                    "tz": "Europe/Zurich",
                    "time": "14:00:00"
                },
                "_type": "Contribution",
                "endDate": {
                    "date": "2011-06-24",
                    "tz": "Europe/Zurich",
                    "time": "14:20:00"
                },
                "description": "",
                "subContributions": [],
                "title": "d2s1c1",
                "track": null,
                "duration": 20,
                "session": "d2s1",
                "location": "CERN",
                "_fossil": "contributionMetadataWithSubContribs",
                "type": null,
                "id": "3",
                "room": null
            }
        ],
        "id": "0"
    }
],
"location": "CERN",
"_fossil": "conferenceMetadataWithSessions",

```

```
    "timezone": "Europe/Zurich",
    "type": "meeting",
    "id": "137346",
    "room": null
  }
]
```

Timetable

URL Format

/export/timetable/ID.TYPE

The ID should be the event ID, e.g. *123*.

Results

Returns the timetable of the event.

Result for <https://indico.server/export/timetable/137346.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes>:

```
{
  "count": 1,
  "additionalInfo": {},
  "_type": "HTTPAPIResult",
  "complete": true,
  "url": "https://indico.server/export/timetable/137346.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes",
  "ts": 1367242732,
  "results": {
    "137346": {
      "20130429": {
        "c0": {
          "startDate": {
            "date": "2013-04-29",
            "tz": "Europe/Zurich",
            "time": "16:00:00"
          },
          "_type": "ContribSchEntry",
          "material": [],
          "endDate": {
            "date": "2013-04-29",
            "tz": "Europe/Zurich",
            "time": "16:30:00"
          },
          "description": "",
          "title": "Contrib 1",
          "id": "c0",
          "contributionId": "0",
          "sessionSlotId": null,
          "conferenceId": "137346",
          "presenters": [],
          "sessionId": null,
        }
      }
    }
  }
}
```

```

        "location": "CERN",
        "uniqueId": "a137346t0",
        "_fossil": "contribSchEntryDisplay",
        "sessionCode": null,
        "entryType": "Contribution",
        "room": "160-1-009"
    }
}
}

```

Event Search

URL Format

/export/event/search/TERM.TYPE

The TERM should be a string, e.g. “ic hep”

Results

Returns the events found.

Result for <https://indico.server/export/event/search/ic hep.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes>:

```

{
  "count": 5,
  "additionalInfo": {},
  "_type": "HTTPAPIResult",
  "complete": true,
  "url": "https://indico.server/export/event/search/ic hep.json?ak=00000000-
↪0000-0000-0000-000000000000&pretty=yes",
  "ts": 1367245058,
  "results": [
    {
      "startDate": {
        "date": "2010-07-16",
        "tz": "UTC",
        "time": "11:00:00"
      },
      "hasAnyProtection": false,
      "id": "101465",
      "title": "Rehearsals for ICHEP Friday 16th July Afternoon Session"
    },
    {
      "startDate": {
        "date": "2010-08-06",
        "tz": "UTC",
        "time": "12:00:00"
      },
      "hasAnyProtection": false,
      "id": "102669",
      "title": "Overview of LHC physics results at ICHEP"
    },
    {

```

```

        "startDate": {
            "date": "2010-08-18",
            "tz": "UTC",
            "time": "17:00:00"
        },
        "hasAnyProtection": false,
        "id": "104128",
        "title": "Seminer Oturumu: \"ATLAS status and highlights as of ICHEP\" Dr.
↪ Tayfun Ince (Universitaet Bonn)"
    },
    {
        "startDate": {
            "date": "2011-07-23",
            "tz": "UTC",
            "time": "11:00:00"
        },
        "hasAnyProtection": false,
        "id": "145521",
        "title": "89th Plenary ECFA and Joint EPS\\ICHEP-ECFA Session - Grenoble, ↪
↪France"
    },
    {
        "startDate": {
            "date": "2012-01-12",
            "tz": "UTC",
            "time": "08:00:00"
        },
        "hasAnyProtection": false,
        "id": "168897",
        "title": "ICHEP 2012 Outreach Planning Meeting"
    }
}
]
}

```

Files

General Information

The file export is only available for authenticated users, i.e. when using an API key and a signature (if enabled).

URL Format

/export/event/EVENT_ID/session/SESSION_ID/contrib/CONTRIBUTION_ID/subcontrib/SUBCONTRIBUTION_ID/material/MATERIAL_ID

All ID's should be single ID, not separated list.

The *EVENT_ID* should be the event ID, e.g. 123.

The *SESSION_ID* (*optional*) should be the session ID, e.g. 4.

The *CONTRIBUTION_ID* (*optional*) should be the contribution ID, e.g. 3.

The *SUBCONTRIBUTION_ID* (*optional*) should be the sub-contribution ID, e.g. 1.

The *MATERIAL_ID* should be the material name if it came default group e.g. *Slides* or material ID if not, e.g. 2.

The *RESOURCE_ID* should be the resource ID.

Only supported *TYPE* for files is *bin* (binary data).

Parameters

None

Detail Levels

file

Returns file (or an error in *JSON* format).

For example: `https://indico.server/export/event/23/session/0/contrib/3/material/slides/3.bin?ak=00000000-0000-0000-0000-000000000000`

User

General Information

The user export is only available for authenticated users, i.e. when using an API key and a signature (if enabled).

URL Format

`/export/user/USER_ID.TYPE`

The *USER_ID* should be the user ID, e.g. *44*.

Parameters

None

Results

Returns the user information (or an error in *JSON* format).

Result for `https://indico.server/export/user/36024.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes`:

```
{
  "count": 1,
  "additionalInfo": {},
  "_type": "HTTPAPIResult",
  "complete": true,
  "url": "https://indico.server/export/user/36024.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes",
  "ts": 1367243741,
  "results": [
    {
      "_type": "Avatar",
      "name": "Alberto RESCO PEREZ",
      "firstName": "Alberto",
```

```

    "affiliation": "CERN",
    "familyName": "Resco Perez",
    "email": "test@cern.ch",
    "phone": "+41XXXXXXXXXX",
    "_fossil": "avatar",
    "title": "",
    "id": "36024"
  }
]
}

```

Room Booking

Bookings

Creating bookings

General Information

The Room Booking API is only available for authenticated users, i.e. when using an API key and a signature (if enabled). If the room booking system is restricted to certain users/groups this restriction applies for this API, too. The request will fail if there is a collision with another booking, blocking or unavailable period.

Note that it is not possible to pre-book a room through this api.

URL Format

/api/roomBooking/bookRoom.TYPE

TYPE should be *json* or *xml*.

Parameters

The following parameters are required:

Param	Values	Description
location	text	Room location, e.g. <i>CERN</i>
roomid	text	Room id
from/to	f/t	Start/End time for a booking. Accepted formats: <ul style="list-style-type: none"> • ISO 8601 subset - YYYY-MM-DD[THH:MM] • 'today', 'yesterday', 'tomorrow' and 'now' • days in the future/past: '[+/-]DdHHhMMm'
reason	text	Reason for booking a room
username	text	User login name for whom the booking will be created

Booking a room

POST request

Returns *reservation id* if the booking was successful or error information if there were any problems.

For example:

```
curl --data "username=jdoe&from=2012-12-30T21:30&to=2012-12-30T22:15&reason=meeting&
↳location=CERN&roomid=189" 'http://indico.server/indico/api/roomBooking/bookRoom.json
↳'
```

Result:

```
{
  {
    "url": "\/api\/roomBooking\/bookRoom.json",
    "_type": "HTTPAPIResult",
    "results": {
      "reservationID": 45937
    },
    "ts": 1354695663
  }
}
```

Retrieving bookings

General Information

The reservation export is only available for authenticated users, i.e. when using an API key and a signature (if enabled). If the room booking system is restricted to certain users/groups this restriction applies for the reservation export API, too.

Please note that the room export with the *reservations* detail level is much more appropriate if you need reservations for specific rooms.

URL Format

/export/reservation/LOCATION.TYPE

The *LOCATION* should be the room location, e.g. *CERN*. A - separated list of multiple locations is allowed, too.

Parameters

Param	Short	Values	Description
occurrences	occ	yes, no	Include all occurrences of room reservations.
cancelled	cxl	yes, no	If specified only include cancelled (<i>yes</i>) or non-cancelled (<i>no</i>) reservations.
rejected	rej	yes, no	If specified only include rejected/non-rejected resvs.
confirmed	-	yes, no, pending	If specified only include bookings/pre-bookings with the given state.
archival	arch	yes, no	If specified only include bookings (not) from the past.
recurring	rec	yes, no	If specified only include bookings which are (not) recurring.
repeating	rep	yes, no	Alias for <i>recurring</i>
avc	-	yes, no	If specified only include bookings which (do not) use AVC.
avcsupport	avcs	yes, no	If specified only include bookings which (do not) need AVC Support.
startup-support	sts	yes, no	If specified only include bookings which (do not) need Startup Support.
booked-for	bf	text (wildcards)	Only include bookings where the <i>booked for</i> field matches the given wildcard string.
occurs	-	yyyy-mm-dd	Only include bookings which have a valid occurrence on the given date. Multiple dates can be separated by commas.

Detail Levels

reservations

Returns detailed data about the reservations and the most important information about the booked room.

For example, <https://indico.server/export/reservation/CERN.json?ak=00000000-0000-0000-0000-000000000000&detail=reservation&from=today&to=today&pretty=yes>:

```
{
  "count": 1,
  "additionalInfo": {},
  "_type": "HTTPAPIResult",
  "url": "/export/reservation/CERN.json?ak=00000000-0000-0000-0000-000000000000&
↔detail=reservation&from=today&to=today&pretty=yes",
  "results": [
    {
      "_type": "Reservation",
      "repeat_unit": 1,
      "endDT": {
        "date": "2014-08-14",
        "tz": "Europe/Zurich",
        "time": "12:30:00"
      },
      "room": {
        "_type": "Room",
        "fullName": "500-1-001 - Main Auditorium",
        "id": 57
      },
      "needs_general_assistance": false,
      "isConfirmed": true,
      "isValid": true,
    }
  ]
}
```

```

    "usesAVC": false,
    "repeatability": "daily",
    "repeat_step": 1,
    "vcList": [],
    "reason": "Summer Student Lecture programme",
    "bookedForName": "DOE, John",
    "is_rejected": false,
    "is_cancelled": false,
    "needsAVCSupport": false,
    "startDT": {
        "date": "2014-07-02",
        "tz": "Europe/Zurich",
        "time": "08:30:00"
    },
    "id": 63779,
    "bookingUrl": "http://indico.server/rooms/booking/CERN/63779/",
    "location": "CERN"
  },
  ],
  "ts": 1406727843
}

```

Rooms

General Information

The room export is only available for authenticated users, i.e. when using an API key and a signature (if enabled). If the room booking system is restricted to certain users/groups this restriction applies for the room export API, too.

URL Format

/export/room/LOCATION/ID.TYPE

The *LOCATION* should be the room location, e.g. *CERN*. The *ID* can be either a single room ID or a - separated list.

Parameters

Param	Short	Values	Description
occurrences	occ	yes, no	Include all occurrences of room reservations.
cancelled	cxl	yes, no	If specified only include cancelled (<i>yes</i>) or non-cancelled (<i>no</i>) reservations.
rejected	rej	yes, no	If specified only include rejected/non-rejected resvs.
confirmed	-	yes, no, pending	If specified only include bookings/pre-bookings with the given state.
archival	arch	yes, no	If specified only include bookings (not) from the past.
recurring	rec	yes, no	If specified only include bookings which are (not) recurring.
repeating	rep	yes, no	Alias for <i>recurring</i>
avc	-	yes, no	If specified only include bookings which (do not) use AVC.
avcsupport	avcs	yes, no	If specified only include bookings which (do not) need AVC Support.
startup-support	sts	yes, no	If specified only include bookings which (do not) need Startup Support.
booked-for	bf	text (wildcards)	Only include bookings where the <i>booked for</i> field matches the given wildcard string.
occurs	-	yyyy-mm-dd	Only include bookings which have a valid occurrence on the given date. Multiple dates can be separated by commas.

Detail Levels

rooms

Returns basic data about the rooms.

For example, <https://indico.server/export/room/CERN/57.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes>:

```
{
  "count": 1,
  "additionalInfo": {},
  "_type": "HTTPAPIResult",
  "url": "/export/room/CERN/57.json?ak=00000000-0000-0000-0000-000000000000&
↪pretty=yes",
  "results": [
    {
      "building": "500",
      "_type": "Room",
      "name": "Main Auditorium",
      "floor": "1",
      "longitude": "6.0542704900999995",
      "vcList": [
        "Audio Conference",
        "Built-in (MCU) Bridge",
        "CERN MCU",
        "ESnet MCU",
        "EVO",
        "H323 point2point",
        "Vidyo"
      ],
      "equipment": [
```

```

        "Blackboard",
        "Computer Projector",
        "Ethernet",
        "Microphone",
        "PC",
        "Telephone conference",
        "Video conference",
        "Webcast/Recording",
        "Wireless"
    ],
    "roomNr": "001",
    "location": "CERN",
    "latitude": "46.23141394580001",
    "fullName": "500-1-001 - Main Auditorium",
    "id": 57,
    "bookingUrl": "/indico/rooms/room/CERN/57/book",
    "avc": true
  }
],
"ts": 1406729635
}

```

reservations

Returns basic data about the rooms and their reservations in the given timeframe.

Output for <https://indico.server/export/room/CERN/57.json?ak=00000000-0000-0000-0000-000000000000&detail=reservations&from=today&to=today&pretty=yes>:

```

{
  "count": 1,
  "additionalInfo": {},
  "_type": "HTTPAPIResult",
  "url": "/export/room/CERN/57.json?ak=00000000-0000-0000-0000-000000000000&
↪detail=reservations&from=today&to=today&pretty=yes",
  "results": [
    {
      "building": "500",
      "_type": "Room",
      "name": "Main Auditorium",
      "floor": "1",
      "reservations": [
        {
          "_type": "Reservation",
          "repeat_unit": 1,
          "endDT": {
            "date": "2014-08-14",
            "tz": "Europe/Zurich",
            "time": "12:30:00"
          },
          "needs_general_assistance": false,
          "isConfirmed": true,
          "isValid": true,
          "usesAVC": false,
          "repeatability": "daily",
          "repeat_step": 1,
        }
      ]
    }
  ]
}

```

```

        "vcList": [],
        "reason": "Summer Student Lecture programme",
        "bookedForName": "DOE, John",
        "is_rejected": false,
        "is_cancelled": false,
        "needsAVCSupport": false,
        "startDT": {
            "date": "2014-07-02",
            "tz": "Europe/Zurich",
            "time": "08:30:00"
        },
        "id": 63779,
        "bookingUrl": "http://pcavc005.cern.ch:8000/indico/rooms/booking/
↪CERN/63779/",
        "location": "CERN"
    }
],
"longitude": "6.0542704900999995",
"vcList": [
    "Audio Conference",
    "Built-in (MCU) Bridge",
    "CERN MCU",
    "ESnet MCU",
    "EVO",
    "H323 point2point",
    "Vidyo"
],
"equipment": [
    "Blackboard",
    "Computer Projector",
    "Ethernet",
    "Microphone",
    "PC",
    "Telephone conference",
    "Video conference",
    "Webcast/Recording",
    "Wireless"
],
"roomNr": "001",
"location": "CERN",
"latitude": "46.23141394580001",
"fullName": "500-1-001 - Main Auditorium",
"id": 57,
"bookingUrl": "/indico/rooms/room/CERN/57/book",
"avc": true
}
],
"ts": 1406731966
}

```

Get room by room name

General Information

The search room export is guest allowed because the room data is public (no the reservations).

URL Format

/export/roomName/LOCATION/ROOMNAME.TYPE

The *LOCATION* should be the room location, e.g. *CERN*. The *ROOMNAME* is a single ROOMNAME.

Parameters

No parameters needed.

Results

Returns basic data about the rooms.

For example, [https://indico.server/export/roomName/CERN/Main Auditorium.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes](https://indico.server/export/roomName/CERN/Main%20Auditorium.json?ak=00000000-0000-0000-0000-000000000000&pretty=yes):

```
{
  "count": 1,
  "additionalInfo": {},
  "_type": "HTTPAPIResult",
  "url": "/export/roomName/CERN/Main Auditorium.json?ak=00000000-0000-0000-0000-
↪000000000000&pretty=yes",
  "results": [
    {
      "building": "500",
      "_type": "Room",
      "name": "Main Auditorium",
      "floor": "1",
      "longitude": "6.0542704900999995",
      "vcList": [
        "Audio Conference",
        "Built-in (MCU) Bridge",
        "CERN MCU",
        "ESnet MCU",
        "EVO",
        "H323 point2point",
        "Vidyo"
      ],
      "equipment": [
        "Blackboard",
        "Computer Projector",
        "Ethernet",
        "Microphone",
        "PC",
        "Telephone conference",
        "Video conference",
        "Webcast/Recording",
        "Wireless"
      ],
      "roomNr": "001",
      "location": "CERN",
      "latitude": "46.23141394580001",
      "fullName": "500-1-001 - Main Auditorium",
      "id": 57,
      "bookingUrl": "/indico/rooms/room/CERN/57/book",
    }
  ]
}
```

```
        "avc": true
    }
],
"ts": 1406732578
}
```

Video Services & Collaboration

URL Format

/export/video/SERVICE_ID.TYPE

The SERVICE_ID may be a single collaboration type or many separated by -. At present, the only TYPE compatible with the Video Services export is *ics* / iCalendar.

As the query is signed with a signature generated using secret API key, the query need not be timestamped. Instead, each booking is given its own unique identifier and, therefore, the generated query URL may be fed as a persistent calendar for importing in your application of choice. The link will only expire once your account has been closed, if TTL is required by your server administrator or your API key is deleted.

If TTL is required by your server administrator, requests should be both timestamped and signed.

Parameters

Param	Short	Val- ues	Description
alarms	-	int	If defined with a value of x int, all bookings to be exported will be accompanied by a matching alarm set to occur x minutes prior to the start of the booking itself. The alarm is set to provide a popup reminder. The default value is 0 minutes.

Please be aware that specifying the alarm parameter in your query will assign alarms to *every* booking which is to be exported.

Service Identifiers Used in CERN

The following parameters are both for example to other installations, and for use within CERN installations of Indico, they represent the options available for configuration through the SERVICE_ID parameter.

SERVICE_ID	Linked Service
all	Traverse all plugin indices.
vidyo	Return Vidyo bookings only.
evo	Return EVO bookings only.
mcu	Return CERNMCU bookings only.
webcast	Return Webcast Requests only.
recording	Return Recording Requests only.

SERVICE_ID may be one of more of these identifiers, if more than one is required simply join the service names with -, please refer to common examples for usage scenarios.

Common Examples

All Bookings

To obtain all bookings in the past 7 days for all collaboration plugins registered:

```
https://indico.server/export/video/all.ics?ak=API_KEY&from=-70000&to=now&signature=SIGNATURE
```

To obtain the same output, but with alarms set to display 20 minutes prior to each event:

```
https://indico.server/export/video/all.ics?ak=API_KEY&alarms=20&from=-70000&to=now&signature=SIGNATURE
```

Individual Plugin Bookings

To obtain bookings from a specific plugin, in this example Vidyo, from a set date range and with alarms 30 minutes prior to the booking:

```
https://indico.server/export/video/vidyo.ics?ak=API_KEY&alarms=30&from=2011-08-01&to=2011-12-01&signature=SIGNATURE
```

Multiple Plugin Bookings

We may also reference more than one plugin's bookings, to request all EVO and CERNMCU bookings over a specified date range with no alarms:

```
https://indico.server/export/video/evo-mcu.ics?ak=API_KEY&from=2011-09-01&to=2011-09-08&signature=SIGNATURE
```

HTTP API Tools

This part of the documentation focuses on the core modules of Indico and includes information about the **models** and **utility functions and classes** that are useful for understanding the internals of the application.

API reference

This part of the documentation focuses on the core modules of Indico and includes information about the **models** and **utility functions and classes** that are useful for understanding the internals of the application.

Event

Todo

Docstrings (module, models, operations, utilities, settings)

Models

class `indico.modules.events.models.events.Event` (**kwargs)

Bases: `indico.core.db.sqlalchemy.searchable_titles.SearchableTitleMixin`,
`indico.core.db.sqlalchemy.descriptions.DescriptionMixin`, `indico.core.db.sqlalchemy.locations.LocationMixin`,
`indico.core.db.sqlalchemy.protection.ProtectionManagersMixin`, `indico.core.db.sqlalchemy.attachments.AttachedItemsMixin`,
`indico.core.db.sqlalchemy.notes.AttachedNotesMixin`,
`indico.modules.events.models.persons.PersonLinkDataMixin`, `flask_sqlalchemy.Model`

An Indico event

This model contains the most basic information related to an event.

Note that the ACL is currently only used for managers but not for view access!

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

ATTACHMENT_FOLDER_ID_COLUMN = u'event_id'

access_key

acl_entries

The ACL entries for the event

additional_info

allow_access_key = True

allow_location_inheritance = False

allow_no_access_contact = True

as_legacy

Return a legacy *Conference* object

can_lock (user)

Check whether the user can lock/unlock the event

category

The category containing the event

classmethod category_chain_overlaps (category_ids)

Create a filter that checks whether the event has any of the provided category ids in its parent chain.

Parameters category_ids – A list of category ids or a single category id

category_id

The ID of immediate parent category of the event

cfa

cfp

cloned_from

The event this one was cloned from

cloned_from_id

If this event was cloned, the id of the parent event

contact_emails

contact_phones

contact_title

created_dt

The creation date of the event

creator

The user who created the event

creator_id

The ID of the user who created the event

default_page

The event's default page (conferences only)

default_page_id

The ID of the event's default page (conferences only)

default_render_mode = 1**delete** (*reason, user=None*)**disallowed_protection_modes = frozenset([])****display_tzinfo**

The tzinfo of the event as preferred by the current user

duration**end_dt**

The end date of the event

end_dt_display

The 'displayed end dt', which is usually the actual end dt, but may be overridden for a conference.

end_dt_local**end_dt_override****ends_after** (*dt*)

Check whether the event ends on/after the specified date

event

Convenience property so all event entities have it

external_url

get_allowed_sender_emails (*include_current_user=True, include_creator=True, include_managers=True, include_contact=True, include_chairs=True, extra=None*)

Return the emails of people who can be used as senders (or rather Reply-to contacts) in emails sent from within an event.

Parameters

- **include_current_user** – Whether to include the email of the currently logged-in user
- **include_creator** – Whether to include the email of the event creator
- **include_managers** – Whether to include the email of all event managers
- **include_contact** – Whether to include the “event contact” emails
- **include_chairs** – Whether to include the emails of event chairpersons (or lecture speakers)
- **extra** – An email address that is always included, even if it is not in any of the included lists.

Returns An OrderedDict mapping emails to pretty names

get_contribution (*id_*)

Get a contribution of the event

get_contribution_field (*field_id*)**get_non_inheriting_objects** ()

Get a set of child objects that do not inherit protection

get_relative_event_ids ()

Get the first, last, previous and next event IDs.

Any of those values may be `None` if there is no matching event or if it would be the current event.

Returns A dict containing `first`, `last`, `prev` and `next`.

get_session (*id_=None, friendly_id=None*)

Get a session of the event

get_session_block (*id_, scheduled_only=False*)

Get a session block of the event

get_verbose_title (*show_speakers=False, show_series_pos=False*)

Get the event title with some additional information

Parameters

- **show_speakers** – Whether to prefix the title with the speakers of the event.
- **show_series_pos** – Whether to suffix the title with the position and total count in the event’s series.

global_abstract_reviewers

Users who can review on all tracks

global_conveners

Users who are conveners on all tracks

happens_between (*from_dt=None, to_dt=None*)

Check whether the event takes place within two dates

has_feature (**args, **kwargs*)

Checks if a feature is enabled for the event

has_logo

has_stylesheet

id

The ID of the event

inherit_location = `False`

inheriting_have_acl = `True`

is_deleted

If the event has been deleted

is_locked

If the event is locked (read-only mode)

classmethod is_visible_in (*category*)

Create a filter that checks whether the event is visible in the specified category.

iter_days (*tzinfo=None*)

keywords

A list of tags/keywords for the event

location_backref_name = `u'events'`

locator

log (*realm, kind, module, summary, user=None, type_=u'simple', data=None*)

Creates a new log entry for the event

Parameters

- **realm** – A value from *EventLogRealm* indicating the realm of the action.
- **kind** – A value from *EventLogKind* indicating the kind of the action that was performed.
- **module** – A human-friendly string describing the module related to the action.
- **summary** – A one-line summary describing the logged action.
- **user** – The user who performed the action.
- **type** – The type of the log entry. This is used for custom rendering of the log message/data
- **data** – JSON-serializable data specific to the log type.

In most cases the `simple` log type is fine. For this type, any items from `data` will be shown in the detailed view of the log entry. You may either use a dict (which will be sorted) alphabetically or a list of `key, value` pairs which will be displayed in the given order.

`logging_disabled`

Temporarily disables event logging

This is useful when performing actions e.g. during event creation or at other times where adding entries to the event log doesn't make sense.

`logo`

The logo's raw image data

`logo_metadata`

The metadata of the logo (hash, size, filename, content_type)

`logo_url`

`move (category)`

`move_start_dt (start_dt)`

Set event `start_dt` and adjust its timetable entries

`organizer_info`

`own_address`

`own_no_access_contact`

`own_room`

`own_room_id`

`own_room_name`

`own_venue`

`own_venue_id`

`own_venue_name`

`participation_regform`

`person_links`

Persons associated with this event

`possible_render_modes = set([<RenderMode.html: 1>])`

`preload_all_acl_entries ()`

protection_mode

protection_parent

published_registrations

references

External references associated with this event

render_mode = 1

scheduled_notes

series

The series this event is part of

series_id

The ID of the series this events belongs to

short_external_url

short_url

start_dt

The start date of the event

start_dt_display

The 'displayed start dt', which is usually the actual start dt, but may be overridden for a conference.

start_dt_local

start_dt_override

starts_between (*from_dt=None, to_dt=None*)

Check whether the event starts within two dates

stylesheet

The stylesheet's raw image data

stylesheet_metadata

The metadata of the stylesheet (hash, size, filename)

theme

timezone

The timezone of the event

title

type

type_

tzinfo

url

url_shortcut

The URL shortcut for the event

visibility

The visibility depth in category overviews

class `indico.modules.events.models.events.EventType`

Bases: `indico.util.struct.enum.RichIntEnum`

conference = 3

lecture = 1

legacy_name

meeting = 2

class `indico.modules.events.models.persons.AuthorsSpeakersMixin`

Bases: `object`

primary_authors

secondary_authors

speakers

class `indico.modules.events.models.persons.EventPerson (**kwargs)`

Bases: `indico.modules.users.models.users.PersonMixin`, `flask_sqlalchemy.Model`

A person inside an event, e.g. a speaker/author etc.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

address

affiliation

classmethod `create_from_user (user, event=None, is_untrusted=False)`

email

event

event_id

first_name

classmethod `for_user (user, event=None, is_untrusted=False)`

Return `EventPerson` for a matching `User` in `Event` creating if needed

has_role (role, obj)

Whether the person has a role in the ACL list of a given object

id

invited_dt

is_untrusted

last_name

classmethod `link_user_by_email (user)`

Links all email-based persons matching the user's email addresses with the user.

Parameters `user` – A `User` object.

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

merge_person_info (*args, **kwargs)

classmethod merge_users (target, source)

Merge the EventPersons of two users.

Parameters

- **target** – The target user of the merge
- **source** – The user that is being merged into *target*

phone

principal

user

user_id

class `indico.modules.events.models.persons.EventPersonLink` (*args, **kwargs)

Bases: `indico.modules.events.models.persons.PersonLinkBase`

Association between EventPerson and Event.

Chairperson or speaker (lecture)

display_order

event_id

id

is_submitter

object_relationship_name = u'event'

person

person_id

person_link_backref_name = u'event_links'

person_link_unique_columns = (u'event_id',)

class `indico.modules.events.models.persons.PersonLinkBase` (*args, **kwargs)

Bases: `indico.modules.users.models.users.PersonMixin`, `flask_sqlalchemy.Model`

Base class for EventPerson associations.

address

affiliation

display_order = `Column(None, Integer(), table=None, nullable=False, default=ColumnDefault(0))`

```

display_order_key
email
first_name
id = Column(None, Integer(), table=None, primary_key=True, nullable=False)
last_name
object
object_relationship_name = None
    The name of the relationship pointing to the object the person is linked to
person = <RelationshipProperty at 0x7faacb5e5af0; no key>
person_id = Column(None, Integer(), ForeignKey(u'events.persons.id'), table=None, nullable=False)
person_link_backref_name = None
    The name of the backref on the EventPerson
person_link_unique_columns = None
    The columns which should be included in the unique constraint.
phone
title

```

```
class indico.modules.events.models.persons.PersonLinkDataMixin
```

```
    Bases: object
```

```
    person_link_data
```

```
class indico.modules.events.models.principals.EventPrincipal (**kwargs)
```

```
    Bases: indico.core.db.sqlalchemy.principals.PrincipalRolesMixin,
           flask_sqlalchemy.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
allow_emails = True
```

```
allow_networks = True
```

```
email
```

```
event_id
```

The ID of the associated event

```
full_access
```

```
id
```

The ID of the acl entry

```
ip_network_group
```

```
ip_network_group_id
```

```
local_group
```

```
local_group_id
```

```
multipass_group_name
```

```
    multipass_group_provider
    principal_backref_name = u'in_event_acls'
    principal_for = u'Event'
    read_access
    roles
    type
    unique_columns = (u'event_id',)
    user
    user_id
```

```
class indico.modules.events.models.references.EventReference (**kwargs)
    Bases: indico.modules.events.models.references.ReferenceModelBase
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
    event_id
    id
    reference_backref_name = u'event_references'
    reference_type
    reference_type_id
    value
```

```
class indico.modules.events.models.references.ReferenceModelBase (**kwargs)
    Bases: flask_sqlalchemy.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
    id = Column(None, Integer(), table=None, primary_key=True, nullable=False)
```

```
    reference_backref_name = None
        The name of the backref on the ReferenceType
```

```
    reference_type = <RelationshipProperty at 0x7faacb8729e0; no key>
```

```
    reference_type_id = Column(None, Integer(), ForeignKey(u'indico.reference_types.id'), table=None, nullable=False)
```

```
    url
        The URL of the referenced entity.
        None if no URL template is defined.
```

```
    urn
        The URN of the referenced entity.
        None if no scheme is defined.
```

value = Column(None, String(), table=None, nullable=False)

class `indico.modules.events.models.references.ReferenceType` (**kwargs)
 Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

id

The unique ID of the reference type

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

name

The name of the referenced system

scheme

The scheme used to build an URN for the reference

url_template

A URL template to build a link to a referenced entity

class `indico.modules.events.models.reviews.ProposalCommentMixin`

Bases: `object`

can_edit (*user*)

timeline_item_type = `u'comment'`

class `indico.modules.events.models.reviews.ProposalGroupProxy` (*group*)

Bases: `object`

Represents the object that the proposals can be grouped by.

It provides all necessary methods for building the URLs, displaying the grouping information, etc.

full_title

full_title_attr = `u'full_title'`

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

title

title_attr = u'title'

class `indico.modules.events.models.reviews.ProposalMixin`

Bases: `object`

Classes that represent a proposal object should extend this class (ex: `Abstract`, `Paper`).

call_for_proposals_attr = None

Attribute to retrieve the object with access to the reviewing settings

can_comment (*user*)

can_review (*user*, *check_state=False*)

cfp

create_comment_endpoint = None

create_judgment_endpoint = None

create_review_endpoint = None

delete_comment_endpoint = None

edit_comment_endpoint = None

edit_review_endpoint = None

get_delete_comment_url (*comment*)

get_last_revision ()

get_revisions ()

get_save_comment_url (*comment=None*)

get_save_judgment_url ()

get_save_review_url (*group=None*, *review=None*)

is_in_final_state

proposal_type = None

A unique identifier to handle rendering differences between proposal types

revisions_enabled = True

Whether there is support for multiple revisions per proposal or just one

class `indico.modules.events.models.reviews.ProposalReviewMixin`

Bases: `object`

Mixin for proposal reviews

Classes that represent a review of a proposal should extend this class (ex: `AbstractReview`, `PaperReview`).

can_edit (*user*)

group

group_attr = None

Object used to group reviews together

group_proxy_cls

Proxy class to provide the necessary properties and methods to the review grouping object

alias of `ProposalGroupProxy`

revision

revision_attr = None

The revision object that the review refers to

score

timeline_item_type = u'review'

A unique identifier to handle rendering differences between timeline items

class `indico.modules.events.models.reviews.ProposalRevisionMixin`

Bases: `object`

Properties and methods of a proposal revision.

get_reviewed_for_groups (*user*, *include_reviewed=False*)

get_reviewer_render_data (**args*, ***kwargs*)

get_reviews (*group=None*, *user=None*)

get_timeline (*user=None*)

proposal

proposal_attr = None

The attribute of the revision used to fetch the proposal object.

revisions_enabled = True

Whether the reviewing process supports multiple revisions per proposal. If set to false it is assumed that the reviewing process supports only one revision per proposal.

class `indico.modules.events.models.series.EventSeries` (***kwargs*)

Bases: `flask_sqlalchemy.Model`

A series of events.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

id

The ID of the series

show_links

Whether to show links to the other events in the same series on the main event page.

show_sequence_in_title

Whether to show the sequence number of an event in its title on category display pages and on the main event page.

```
class indico.modules.events.models.settings.EventSetting (**kwargs)
Bases: indico.core.settings.models.base.JSONSettingsBase, indico.modules.events.models.settings.EventSettingsMixin, flask_sqlalchemy.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

event

event_id

id

module

name

settings_backref_name = u'settings'

value

```
class indico.modules.events.models.settings.EventSettingPrincipal (**kwargs)
Bases: indico.core.settings.models.base.PrincipalSettingsBase, indico.modules.events.models.settings.EventSettingsMixin, flask_sqlalchemy.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

email = None

event

event_id

extra_key_cols = (u'event_id',)

id

ip_network_group = None

ip_network_group_id = None

local_group

local_group_id

module


```

multipass_group_name
multipass_group_provider
name
principal_backref_name = u'in_event_settings_acs'
settings_backref_name = u'settings_principals'
type
user
user_id

```

```
class indico.modules.events.models.settings.EventSettingsMixin
```

```
Bases: object
```

```
event = <RelationshipProperty at 0x7faacb465848; no key>
```

```
event_id = Column(None, Integer(), ForeignKey(u'events.events.id'), table=None, nullable=False)
```

```
settings_backref_name = None
```

```
class indico.modules.events.models.static_list_links.StaticListLink (**kwargs)
```

```
Bases: flask_sqlalchemy.Model
```

Display configuration data used in static links to listing pages.

This allows users to share links to listing pages in events while preserving e.g. column/filter configurations.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
classmethod create (event, type_, data)
```

Create a new static list link.

If one exists with the same data, that link is used instead of creating a new one.

Parameters

- **event** – the *Event* for which to create the link
- **type** – the type of the link
- **data** – the data to associate with the link

Returns the newly created *StaticListLink*

```
created_dt
```

```
data
```

```
event
```

```
event_id
```

```
id
```

```
last_used_dt
```

```
classmethod load (event, type_, uuid)
```

Load the data associated with a link

Parameters

- **event** – the *Event* the link belongs to
- **type** – the type of the link
- **uuid** – the UUID of the link

Returns the link data or `None` if the link does not exist

type

uuid

Operations

`indico.modules.events.operations.clone_event(event, start_dt, cloners, category=None)`

Clone an event on a given date/time.

Runs all required cloners.

Parameters

- **start_dt** – The start datetime of the new event;
- **cloners** – A set containing the names of all enabled cloners;
- **category** – The *Category* the new event will be created in.

`indico.modules.events.operations.create_event(*args, **kwargs)`

Create a new event.

Parameters

- **category** – The category in which to create the event
- **event_type** – An *EventType* value
- **data** – A dict containing data used to populate the event
- **add_creator_as_manager** – Whether the creator (current user) should be added as a manager
- **features** – A list of features that will be enabled for the event. If set, only those features will be used and the default feature set for the event type will be ignored.

`indico.modules.events.operations.create_event_references(event, data)`

`indico.modules.events.operations.create_reference_type(data)`

`indico.modules.events.operations.delete_reference_type(reference_type)`

`indico.modules.events.operations.lock_event(event)`

`indico.modules.events.operations.unlock_event(event)`

`indico.modules.events.operations.update_event(event, update_timetable=False, **data)`

`indico.modules.events.operations.update_event_protection(event, data)`

`indico.modules.events.operations.update_event_type(event, type_)`

`indico.modules.events.operations.update_reference_type(reference_type, data)`

Utilities

class `indico.modules.events.util.ListGeneratorBase` (*event*, *entry_parent=None*)

Bases: `object`

Base class for classes performing actions on Indico object lists.

Parameters

- **event** – The associated *Event*
- **entry_parent** – The parent of the entries of the list. If it's `None`, the parent is assumed to be the event itself.

default_list_config = None

The default list configuration dictionary

endpoint = None

The endpoint of the list management page

entry_parent = None

The parent object of the list items

event = None

The event the list is associated with

flash_info_message (*obj*)

generate_static_url ()

Return a URL with a uuid referring to the list's configuration.

get_list_url (*uuid=None*, *external=False*)

Return the URL of the list management page.

list_link_type = None

Unique list identifier

static_items = None

Columns that originate from the list item's properties, relationships etc, but not from user defined fields (e.g. registration/contribution fields)

store_configuration ()

Load the filters from the request and store them in the session.

class `indico.modules.events.util.ZipGeneratorMixin`

Mixin for RHs that generate zip with files

`indico.modules.events.util.create_event_logo_tmp_file` (*event*)

Creates a temporary file with the event's logo

`indico.modules.events.util.get_base_ical_parameters` (*user*, *detail*, *path*,
params=None)

Returns a dict of all parameters expected by iCal template

`indico.modules.events.util.get_events_created_by` (*user*, *dt=None*)

Gets the IDs of events created by the user

Parameters

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

Returns A set of event ids

`indico.modules.events.util.get_events_managed_by` (*user*, *dt=None*)

Gets the IDs of events where the user has management privs.

Parameters

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

Returns A set of event ids

`indico.modules.events.util.get_events_with_linked_event_persons` (*user*,
dt=None)

Returns a dict containing the event ids and role for all events where the user is a chairperson or (in case of a lecture) speaker.

Parameters

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

`indico.modules.events.util.get_field_values` (*form_data*)

Split the form fields between custom and static

`indico.modules.events.util.get_object_from_args` (*args=None*)

Retrieves an event object from request arguments.

This utility is meant to be used in cases where the same controller can deal with objects attached to various parts of an event which use different URLs to indicate which object to use.

Parameters **args** – The request arguments. If unspecified, `request.view_args` is used.

Returns An (*object_type*, *event*, *object*) tuple. The event is always the *Event* associated with the object. The object may be an *Event*, *Session*, *Contribution* or *SubContribution*. If the object does not exist, (*object_type*, *None*, *None*) is returned.

`indico.modules.events.util.get_random_color` (*event*)

`indico.modules.events.util.get_theme` (*event*, *override_theme_id=None*)

Get the theme ID and whether it's an override.

This is useful for places where a user may specify a different timetable theme. If the override theme is not valid for the event, a message is flashed and an exception redirecting the user to the main event page is raised.

Raises **BadRequest** – if the override theme id is not valid

Returns a (*theme_id*, *is_override*) tuple

`indico.modules.events.util.register_event_time_change` (*event*)

Register a time-related change for an event

This is an internal helper function used in the model to record changes of the start time or end time. The changes are exposed through the `track_time_changes` contextmanager function.

`indico.modules.events.util.register_time_change` (*entry*)

Register a time-related change for a timetable entry

This is an internal helper function used in the models to record changes of the start time or duration. The changes are exposed through the `track_time_changes` contextmanager function.

`indico.modules.events.util.serialize_event_for_ical` (*event*, *detail_level*)

`indico.modules.events.util.serialize_event_person` (*person*)

Serialize *EventPerson* to JSON-like object

`indico.modules.events.util.serialize_person_link` (*person_link*)
Serialize PersonLink to JSON-like object

`indico.modules.events.util.set_custom_fields` (*obj*, *custom_fields_data*)

`indico.modules.events.util.track_time_changes` (**args*, ***kwds*)
Track time changes of event objects.

This provides a list of changes while the context manager was active and also triggers *times_changed* signals.

If the code running inside the `with` block of this context manager raises an exception, no signals will be triggered.

Parameters

- **auto_extend** – Whether entry parents will get their boundaries automatically extended or not. Passing 'start' will extend only start datetime, 'end' to extend only end datetime.
- **user** – The *User* that will trigger time changes.

`indico.modules.events.util.update_object_principals` (*obj*, *new_principals*,
read_access=False,
full_access=False, *role=None*)

Updates an object's ACL with a new list of principals

Exactly one argument out of *read_access*, *full_access* and *role* must be specified.

Parameters

- **obj** – The object to update. Must have *acl_entries*
- **new_principals** – The set containing the new principals
- **read_access** – Whether the read access ACL should be updated
- **full_access** – Whether the full access ACL should be updated
- **role** – The role ACL that should be updated

Settings

class `indico.modules.events.settings.EventACLProxy` (*proxy*)
Bases: `indico.core.settings.proxy.ACLProxyBase`

Proxy class for event-specific ACL settings

add_principal (*event*, **args*, ***kwargs*)
Adds a principal to an ACL

Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **principal** – A *User* or a *GroupProxy*

contains_user (*event*, **args*, ***kwargs*)
Checks if a user is in an ACL.

To pass this check, the user can either be in the ACL itself or in a group in the ACL.

Parameters

- **event** – Event (or its ID)

- **name** – Setting name
- **user** – A *User*

get (*event*, **args*, ***kwargs*)
Retrieves an ACL setting

Parameters

- **event** – Event (or its ID)
- **name** – Setting name

merge_users (*target*, *source*)
Replaces all ACL user entries for *source* with *target*

remove_principal (*event*, **args*, ***kwargs*)
Removes a principal from an ACL

Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **principal** – A *User* or a *GroupProxy*

set (*event*, **args*, ***kwargs*)
Replaces an ACL with a new one

Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **acl** – A set containing principals (users/groups)

class `indico.modules.events.settings.EventSettingProperty` (*proxy*, *name*, *default*=<*object* *object*>, *attr*=None)

Bases: `indico.core.settings.proxy.SettingProperty`

attr = u'event'

class `indico.modules.events.settings.EventSettingsProxy` (*module*, *defaults*=None, *strict*=True, *acls*=None, *converters*=None)

Bases: `indico.core.settings.proxy.SettingsProxyBase`

Proxy class to access event-specific settings for a certain module

acl_proxy_class
alias of *EventACLProxy*

delete (*event*, **args*, ***kwargs*)
Deletes settings.

Parameters

- **event** – Event (or its ID)
- **names** – One or more names of settings to delete

delete_all (*event*, **args*, ***kwargs*)
Deletes all settings.

Parameters **event** – Event (or its ID)

get (*event*, **args*, ***kwargs*)
Retrieves the value of a single setting.

Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **default** – Default value in case the setting does not exist

Returns The settings's value or the default value

get_all (*event*, **args*, ***kwargs*)
Retrieves all settings

Parameters

- **event** – Event (or its ID)
- **no_defaults** – Only return existing settings and ignore defaults.

Returns Dict containing the settings

query
Returns a query object filtering by the proxy's module.

set (*event*, **args*, ***kwargs*)
Sets a single setting.

Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **value** – Setting value; must be JSON-serializable

set_multi (*event*, **args*, ***kwargs*)
Sets multiple settings at once.

Parameters

- **event** – Event (or its ID)
- **items** – Dict containing the new settings

class `indico.modules.events.settings.ThemeSettingsProxy`

Bases: `object`

defaults

get_themes_for (**args*, ***kwargs*)

settings

themes

`indico.modules.events.settings.event_or_id` (*f*)

Abstract

Todo

Docstrings (module, models, operations, utilities, settings)

Models

```
class indico.modules.events.abstracts.models.abstracts.Abstract (**kwargs)
    Bases: indico.modules.events.models.reviews.ProposalMixin, indico.modules.events.models.reviews.ProposalRevisionMixin, indico.core.db.sqlalchemy.descriptions.DescriptionMixin, indico.modules.events.contributions.models.contributions.CustomFieldsMixin, indico.modules.events.models.persons.AuthorsSpeakersMixin, flask_sqlalchemy.Model
```

Represents an abstract that can be associated to a Contribution.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

accepted_contrib_type

accepted_contrib_type_id

accepted_track

accepted_track_id

call_for_proposals_attr = u'cfa'

can_access (*user*)

can_comment (*user*, *check_state=False*)

can_convene (*user*)

can_edit (*user*)

can_judge (*user*, *check_state=False*)

can_review (*user*, *check_state=False*)

can_see_reviews (*user*)

can_withdraw (*user*, *check_state=False*)

candidate_contrib_types

candidate_tracks

create_comment_endpoint = u'abstracts.comment_abstract'

create_judgment_endpoint = u'abstracts.judge_abstract'

create_review_endpoint = u'abstracts.review_abstract'

data_by_field

default_render_mode = 2

delete_comment_endpoint = u'abstracts.delete_abstract_comment'

duplicate_of

duplicate_of_id

edit_comment_endpoint = u'abstracts.edit_abstract_comment'

edit_review_endpoint = u'abstracts.edit_review'

edit_track_mode

event**event_id****field_values**

Data stored in abstract/contribution fields

friendly_id**get_reviewed_for_groups** (*user*, *include_reviewed=False*)**get_timeline** (*user=None*)**get_track_question_scores** ()**get_track_reviewing_state** (*track*)**get_track_score** (*track*)**id****is_deleted****is_in_final_state****judge**

User who judged the abstract

judge_id

ID of the user who judged the abstract

judgment_comment**judgment_dt****locator**

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The `other` locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

marshmallow_aliases = {*u'description'*: *u'content'*}**merged_into****merged_into_id****modified_by****modified_by_id**

```
modified_dt
person_links
    Persons associated with this abstract
possible_render_modes = set([<RenderMode.markdown: 2>])
proposal_type = u'abstract'
public_state
render_mode = 2
reset_state()
reviewed_for_tracks
reviewing_state
revisions_enabled = False
score
state
submission_comment
submitted_contrib_type
submitted_contrib_type_id
submitted_dt
submitted_for_tracks
submitter
    User who submitted the abstract
submitter_id
    ID of the user who submitted the abstract
title
user_owns (user)
verbose_title
```

```
class indico.modules.events.abstracts.models.abstracts.AbstractPublicState
    Bases: indico.util.struct.enum.RichIntEnum
```

```
    accepted = 3
    awaiting = -1
    duplicate = 6
    merged = 5
    rejected = 4
    under_review = -2
    withdrawn = 2
```

```
class indico.modules.events.abstracts.models.abstracts.AbstractReviewingState
    Bases: indico.util.struct.enum.RichIntEnum
```

```
    conflicting = 3
    in_progress = 1
```

`mixed = 5`

`negative = 4`

`not_started = 0`

`positive = 2`

class `indico.modules.events.abstracts.models.abstracts.AbstractState`
 Bases: `indico.util.struct.enum.RichIntEnum`

`accepted = 3`

`duplicate = 6`

`merged = 5`

`rejected = 4`

`submitted = 1`

`withdrawn = 2`

class `indico.modules.events.abstracts.models.abstracts.EditTrackMode`
 Bases: `int, indico.util.struct.enum.IndicoEnum`

`both = 1`

`none = 0`

`reviewed_for = 2`

class `indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstracts` (*event*)
 Bases: `object`

Proxy class to facilitate access to the call for abstracts settings

`allow_attachments`

`allow_comments`

`allow_contributors_in_comments`

`allow_convener_judgment`

`announcement`

`can_edit_abstracts` (*user*)

`can_submit_abstracts` (*user*)

`close()`

`end_dt`

`has_ended`

`has_started`

`is_open`

`judgment_instructions`

`modification_end_dt`

`modification_ended`

`open()`

`rating_range`

reviewing_instructions

schedule (*start_dt, end_dt, modification_end_dt*)

start_dt

submission_instructions

class `indico.modules.events.abstracts.models.comments.AbstractComment` (**kwargs)
Bases: `indico.modules.events.models.reviews.ProposalCommentMixin`, `indico.core.db.sqlalchemy.review_comments.ReviewCommentMixin`, `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

abstract

abstract_id

can_edit (*user*)

can_view (*user*)

created_dt

id

is_deleted

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The `other` locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

marshmallow_aliases = {`u'text': u'text'`}

modified_by

modified_by_id

modified_dt

render_mode = 2

user

```

user_backref_name = u'abstract_comments'
user_id
user_modified_backref_name = u'modified_abstract_comments'
visibility

```

```

class indico.modules.events.abstracts.models.email_logs.AbstractEmailLogEntry (**kwargs)
    Bases: flask_sqlalchemy.Model

```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

abstract

abstract_id

body

```

classmethod create_from_email (email_data, email_tpl, user=None)

```

Create a new log entry from the data used to send an email

Parameters

- **email_data** – email data as returned from *make_email*
- **email_tpl** – the abstract email template that created the email
- **user** – the user who performed the action causing the notification

data

email_template

email_template_id

id

recipients

sent_dt

subject

user

user_id

```

class indico.modules.events.abstracts.models.email_templates.AbstractEmailTemplate (**kwargs)
    Bases: flask_sqlalchemy.Model

```

Represents an email template for abstracts notifications.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

body

The body of the template

event

event_id

extra_cc_emails

List of extra email addresses to be added as CC in the email

id

include_authors

Whether to include authors' email addresses as To for emails

include_coauthors

Whether to include co-authors' email addresses as CC for emails

include_submitter

Whether to include the submitter's email address as To for emails

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The `other` locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

position

The relative position of the template in the list of templates

reply_to_address

The address to use as Reply-To in the email

rules

Conditions need to be met to send the email

stop_on_match

Whether to stop checking the rest of the conditions when a match is found

subject

The subject of the email

title

class `indico.modules.events.abstracts.models.fields.AbstractFieldValue` (**kwargs)
Bases: `indico.modules.events.contributions.models.fields.ContributionFieldValueBase`

Store a field values related to abstracts.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

abstract_id

contribution_field

contribution_field_backref_name = u'abstract_values'

contribution_field_id

data

class `indico.modules.events.abstracts.models.files.AbstractFile(**kwargs)`
 Bases: `indico.core.storage.models.StoredFileMixin`, `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

abstract

abstract_id

add_file_date_column = False

content_type

The MIME type of the file

created_dt = None

filename

The name of the file

id

locator

md5

An MD5 hash of the file.

Automatically assigned when `save()` is called.

size

The size of the file (in bytes).

Automatically assigned when `save()` is called.

storage_backend

storage_file_id

class `indico.modules.events.abstracts.models.persons.AbstractPersonLink(*args, **kwargs)`

Bases: `indico.modules.events.models.persons.PersonLinkBase`

Association between EventPerson and Abstract.

abstract_id

author_type

display_order

id

is_speaker

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

object_relationship_name = u'abstract'

person

person_id

person_link_backref_name = u'abstract_links'

person_link_unique_columns = (u'abstract_id',)

class `indico.modules.events.abstracts.models.review_questions.AbstractReviewQuestion` (**kwargs)
Bases: `indico.core.db.sqlalchemy.review_questions.ReviewQuestionMixin`,
`flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

event

event_backref_name = u'abstract_review_questions'

event_id

id

is_deleted

no_score

position

text

class `indico.modules.events.abstracts.models.review_ratings.AbstractReviewRating` (**kwargs)
Bases: `indico.core.db.sqlalchemy.review_ratings.ReviewRatingMixin`,
`flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

id

question

question_class

alias of `AbstractReviewQuestion`

question_id

review

review_class

alias of `AbstractReview`

review_id

value

class `indico.modules.events.abstracts.models.reviews.AbstractAction`

Bases: `indico.util.struct.enum.RichIntEnum`

accept = 1

change_tracks = 3

mark_as_duplicate = 4

merge = 5

reject = 2

class `indico.modules.events.abstracts.models.reviews.AbstractCommentVisibility`

Bases: `indico.util.struct.enum.RichIntEnum`

Most to least restrictive visibility for abstract comments

contributors = 4

conveners = 2

judges = 1

reviewers = 3

users = 5

class `indico.modules.events.abstracts.models.reviews.AbstractReview(**kwargs)`

Bases: `indico.modules.events.models.reviews.ProposalReviewMixin`, `indico.core.db.sqlalchemy.descriptions.RenderModeMixin`, `flask_sqlalchemy.Model`

Represents an abstract review, emitted by a reviewer

A simple constructor that allows initialization from `kwargs`.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

abstract

abstract_id

can_edit (*user, check_state=False*)

`can_view` (*user*)

`comment`

`created_dt`

`default_render_mode = 2`

`group_attr = u'track'`

`id`

`locator`

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

`marshmallow_aliases = {u'_comment': u'comment'}`

`modified_dt`

`possible_render_modes = set([<RenderMode.markdown: 2>])`

`proposed_action`

`proposed_contribution_type`

`proposed_contribution_type_id`

`proposed_related_abstract`

`proposed_related_abstract_id`

`proposed_tracks`

`render_mode = 2`

`revision_attr = u'abstract'`

`score`

`track`

`track_id`

`user`

`user_id`

`visibility`

Operations

```

indico.modules.events.abstracts.operations.add_abstract_files (abstract, files,
                                                                log_action=True)
indico.modules.events.abstracts.operations.close_cfa (event)
indico.modules.events.abstracts.operations.create_abstract (event, ab-
                                                                stract_data, cus-
                                                                tom_fields_data=None,
                                                                send_notifications=False)
indico.modules.events.abstracts.operations.create_abstract_comment (abstract,
                                                                    com-
                                                                    ment_data)
indico.modules.events.abstracts.operations.create_abstract_review (abstract,
                                                                    track,
                                                                    user, re-
                                                                    view_data,
                                                                    ques-
                                                                    tions_data)
indico.modules.events.abstracts.operations.delete_abstract (abstract,
                                                                delete_contrib=False)
indico.modules.events.abstracts.operations.delete_abstract_comment (comment)
indico.modules.events.abstracts.operations.delete_abstract_files (abstract,
                                                                    files)
indico.modules.events.abstracts.operations.judge_abstract (abstract, abstract_data,
                                                                judgment, judge, con-
                                                                trib_session=None,
                                                                merge_persons=False,
                                                                send_notifications=False)
indico.modules.events.abstracts.operations.open_cfa (event)
indico.modules.events.abstracts.operations.reset_abstract_state (abstract)
indico.modules.events.abstracts.operations.schedule_cfa (event, start_dt, end_dt,
                                                                modification_end_dt)
indico.modules.events.abstracts.operations.update_abstract (abstract, ab-
                                                                stract_data, cus-
                                                                tom_fields_data=None)
indico.modules.events.abstracts.operations.update_abstract_comment (comment,
                                                                    com-
                                                                    ment_data)
indico.modules.events.abstracts.operations.update_abstract_review (review, re-
                                                                    view_data,
                                                                    ques-
                                                                    tions_data)
indico.modules.events.abstracts.operations.update_reviewed_for_tracks (abstract,
                                                                    tracks)
indico.modules.events.abstracts.operations.withdraw_abstract (abstract)

```

Utilities

`indico.modules.events.abstracts.util.build_default_email_template` (*event*, *tpl_type*)

Build a default e-mail template based on a notification type provided by the user.

`indico.modules.events.abstracts.util.clear_boa_cache` (*event*)

Delete the cached book of abstract

`indico.modules.events.abstracts.util.create_boa` (*event*)

Create the book of abstracts if necessary

Returns The path to the PDF file

`indico.modules.events.abstracts.util.create_mock_abstract` (**args*, ***kwargs*)

Create a mock abstract that can be used in previews.

Brace for geek references.

`indico.modules.events.abstracts.util.generate_spreadsheet_from_abstracts` (*abstracts*, *static_item_ids*, *dynamic_items*)

Generates a spreadsheet data from a given abstract list.

Parameters

- **abstracts** – The list of abstracts to include in the file
- **static_item_ids** – The abstract properties to be used as columns
- **dynamic_items** – Contribution fields as extra columns

`indico.modules.events.abstracts.util.get_events_with_abstract_persons` (*user*, *dt=None*)

Return a dict of event ids and the abstract submission related roles the user has in that event.

Parameters

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

`indico.modules.events.abstracts.util.get_events_with_abstract_reviewer_convener` (*user*, *dt=None*)

Return a dict of event ids and the abstract reviewing related roles the user has in that event.

Parameters

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

`indico.modules.events.abstracts.util.get_roles_for_event` (*event*)

Return a dictionary of all abstract reviewing roles for this event.

Parameters **event** – the actual event object.

Returns A dictionary in the form `{track: {role: [users]}}`

`indico.modules.events.abstracts.util.get_track_reviewer_abstract_counts` (*event*, *user*)

Get the numbers of abstracts per track for a specific user.

Note that this does not take into account if the user is a reviewer for a track; it just checks whether the user has reviewed an abstract in a track or not.

Returns A dict mapping tracks to dicts containing the counts.

`indico.modules.events.abstracts.util.get_user_abstracts(event, user)`

Get the list of abstracts where the user is a reviewer/convener

`indico.modules.events.abstracts.util.get_user_tracks(event, user)`

Get the list of tracks where the user is a reviewer/convener

`indico.modules.events.abstracts.util.get_visible_reviewed_for_tracks(abstract, user)`

`indico.modules.events.abstracts.util.has_user_tracks(event, user)`

`indico.modules.events.abstracts.util.make_abstract_form(event, notification_option=False, management=False)`

Extends the abstract WTForm to add the extra fields.

Each extra field will use a field named `custom_ID`.

Parameters

- **event** – The *Event* for which to create the abstract form.
- **notification_option** – Whether to add a field to the form to disable triggering notifications for the abstract submission.

:param management Whether it's a manager using the abstract form :return: An *AbstractForm* subclass.

Placeholders

class `indico.modules.events.abstracts.placeholders.EventTitlePlaceholder`
Bases: `indico.util.placeholders.Placeholder`

description = `lu'The title of the event'`

name = `u'event_title'`

classmethod render (*abstract*)

class `indico.modules.events.abstracts.placeholders.EventURLPlaceholder`
Bases: `indico.util.placeholders.Placeholder`

description = `lu'The URL of the event'`

name = `u'event_url'`

classmethod render (*abstract*)

class `indico.modules.events.abstracts.placeholders.AbstractIDPlaceholder`
Bases: `indico.util.placeholders.Placeholder`

description = `lu'The ID of the abstract'`

name = `u'abstract_id'`

classmethod render (*abstract*)

class `indico.modules.events.abstracts.placeholders.AbstractTitlePlaceholder`
Bases: `indico.util.placeholders.Placeholder`

description = `lu'The title of the abstract'`

name = `u'abstract_title'`

classmethod render (*abstract*)

```
class indico.modules.events.abstracts.placeholders.AbstractURLPlaceholder
    Bases: indico.util.placeholders.Placeholder

    advanced = True

    description = lu'The direct URL of the abstract'

    name = u'abstract_url'

    classmethod render (abstract)

class indico.modules.events.abstracts.placeholders.AbstractTrackPlaceholder
    Bases: indico.util.placeholders.Placeholder

    description = lu'The name of the destination track'

    name = u'abstract_track'

    classmethod render (abstract)

class indico.modules.events.abstracts.placeholders.AbstractSessionPlaceholder
    Bases: indico.util.placeholders.Placeholder

    description = lu'The name of the destination session'

    name = u'abstract_session'

    classmethod render (abstract)

class indico.modules.events.abstracts.placeholders.PrimaryAuthorsPlaceholder
    Bases: indico.util.placeholders.Placeholder

    description = lu'The names of the primary authors (separated by commas)'

    name = u'primary_authors'

    classmethod render (abstract)

class indico.modules.events.abstracts.placeholders.CoAuthorsPlaceholder
    Bases: indico.util.placeholders.Placeholder

    description = lu'The names of the co-authors (separated by commas)'

    name = u'co_authors'

    classmethod render (abstract)

class indico.modules.events.abstracts.placeholders.SubmitterNamePlaceholder
    Bases: indico.util.placeholders.Placeholder

    description = lu'The full name of the submitter, no title'

    name = u'submitter_name'

    classmethod render (abstract)

class indico.modules.events.abstracts.placeholders.SubmitterFirstNamePlaceholder
    Bases: indico.util.placeholders.Placeholder

    advanced = True

    description = lu'The first name of the submitter'

    name = u'submitter_first_name'

    classmethod render (abstract)

class indico.modules.events.abstracts.placeholders.SubmitterLastNamePlaceholder
    Bases: indico.util.placeholders.Placeholder
```

```

    advanced = True
    description = lu'The last name of the submitter'
    name = u'submitter_last_name'
    classmethod render (abstract)
class indico.modules.events.abstracts.placeholders.SubmitterTitlePlaceholder
    Bases: indico.util.placeholders.Placeholder
    description = lu'The title of the submitter (Dr., Prof., etc...)'
    name = u'submitter_title'
    classmethod render (abstract)
class indico.modules.events.abstracts.placeholders.TargetAbstractIDPlaceholder
    Bases: indico.util.placeholders.Placeholder
    description = lu'The ID of the target abstract (merge)'
    name = u'target_abstract_id'
    classmethod render (abstract)
class indico.modules.events.abstracts.placeholders.TargetAbstractTitlePlaceholder
    Bases: indico.util.placeholders.Placeholder
    description = lu'The title of the target abstract (merge)'
    name = u'target_abstract_title'
    classmethod render (abstract)
class indico.modules.events.abstracts.placeholders.TargetSubmitterNamePlaceholder
    Bases: indico.util.placeholders.Placeholder
    advanced = True
    description = lu"The full name of the target abstract's submitter, no title (merge)"
    name = u'target_submitter_name'
    classmethod render (abstract)
class indico.modules.events.abstracts.placeholders.TargetSubmitterFirstNamePlaceholder
    Bases: indico.util.placeholders.Placeholder
    advanced = True
    description = lu"The first name of the target abstract's submitter (merge)"
    name = u'target_submitter_first_name'
    classmethod render (abstract)
class indico.modules.events.abstracts.placeholders.TargetSubmitterLastNamePlaceholder
    Bases: indico.util.placeholders.Placeholder
    advanced = True
    description = lu"The last name of the target abstract's submitter (merge)"
    name = u'target_submitter_last_name'
    classmethod render (abstract)

```

```
class indico.modules.events.abstracts.placeholders.JudgmentCommentPlaceholder  
    Bases: indico.util.placeholders.Placeholder
```

```
    description = lu'Comments written by event organizer (upon final decision)'
```

```
    name = u'judgment_comment'
```

```
    classmethod render (abstract)
```

```
class indico.modules.events.abstracts.placeholders.ContributionTypePlaceholder  
    Bases: indico.util.placeholders.Placeholder
```

```
    description = lu'The contribution type that is associated to the abstract'
```

```
    name = u'contribution_type'
```

```
    classmethod render (abstract)
```

```
class indico.modules.events.abstracts.placeholders.ContributionURLPlaceholder  
    Bases: indico.util.placeholders.Placeholder
```

```
    advanced = True
```

```
    description = lu'Contribution URL'
```

```
    name = u'contribution_url'
```

```
    classmethod render (abstract)
```

Settings

```
class indico.modules.events.abstracts.settings.BOACorrespondingAuthorType  
    Bases: indico.util.struct.enum.RichEnum
```

```
    none = u'none'
```

```
    speakers = u'speakers'
```

```
    submitter = u'submitter'
```

```
class indico.modules.events.abstracts.settings.BOASortField  
    Bases: indico.util.struct.enum.RichEnum
```

```
    abstract_title = u'title'
```

```
    id = u'id'
```

```
    schedule = u'schedule'
```

```
    session_title = u'session_title'
```

```
    speaker = u'speaker'
```

Agreement

Todo

Docstrings (module, models, utilities)

Models

class `indico.modules.events.agreements.models.agreements.Agreement` (**kwargs)

Bases: `flask_sqlalchemy.Model`

Agreements between a person and Indico

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

accept (*from_ip*, *reason=None*, *on_behalf=False*)

accepted

attachment

Attachment

attachment_filename

Filename and extension of the attachment

belongs_to (*person*)

static create_from_data (*event*, *type_*, *person*)

data

Definition-specific data of the agreement

definition

event

The Event this agreement is associated with

event_id

ID of the event

id

Entry ID

identifier

Unique identifier within the event and type

is_orphan ()

locator

pending

person_email

Email of the person agreeing

person_name

Full name of the person agreeing

reason

Explanation as to why the agreement was accepted/rejected

reject (*from_ip*, *reason=None*, *on_behalf=False*)

rejected

render (*form*, **kwargs)

reset ()

signed_dt

The date and time the agreement was signed

signed_from_ip

The IP from which the agreement was signed

signed_on_behalf**state**

A *AgreementState*

timestamp

The date and time the agreement was created

type

Type of agreement

user

The user this agreement is linked to

user_id

ID of a linked user

uuid

Entry universally unique ID

class `indico.modules.events.agreements.models.agreements.AgreementState`

Bases: `indico.util.struct.enum.RichIntEnum`

accepted = 1

accepted_on_behalf = 3

agreement accepted on behalf of the person

pending = 0

rejected = 2

rejected_on_behalf = 4

agreement rejected on behalf of the person

Utilities

`indico.modules.events.agreements.util.get_agreement_definitions()`

`indico.modules.events.agreements.util.send_new_agreements(event, name, people, email_body, cc_addresses, from_address)`

Creates and send agreements for a list of people on a given event.

Parameters

- **event** – The *Event* associated with the agreement
- **name** – The agreement type matching a *AgreementDefinition* name
- **people** – The list of people for whom agreements will be created
- **email_body** – The body of the email
- **cc_addresses** – Email addresses to send CCs to
- **from_address** – Email address of the sender

Placeholders

```

class indico.modules.events.agreements.placeholders.AgreementLinkPlaceholder
    Bases: indico.util.placeholders.Placeholder

    description = lu'Link to the agreement page'
    name = u'agreement_link'
    classmethod render (definition, agreement)
    required = True

class indico.modules.events.agreements.placeholders.PersonNamePlaceholder
    Bases: indico.util.placeholders.Placeholder

    description = lu'Name of the person'
    name = u'person_name'
    classmethod render (definition, agreement)

```

Contribution

Todo

Docstrings (module, models, operations, utilities)

Models

```

class indico.modules.events.contributions.models.contributions.Contribution (**kwargs)
    Bases: indico.core.db.sqlalchemy.descriptions.DescriptionMixin, indico.
    core.db.sqlalchemy.protection.ProtectionManagersMixin, indico.core.db.
    sqlalchemy.locations.LocationMixin, indico.core.db.sqlalchemy.attachments.
    AttachedItemsMixin, indico.core.db.sqlalchemy.notes.AttachedNotesMixin,
    indico.modules.events.models.persons.PersonLinkDataMixin, indico.modules.
    events.models.persons.AuthorsSpeakersMixin, indico.modules.events.
    contributions.models.contributions.CustomFieldsMixin, flask_sqlalchemy.
    Model

    ATTACHMENT_FOLDER_ID_COLUMN = u'contribution_id'
    PRELOAD_EVENT_ATTACHED_ITEMS = True
    PRELOAD_EVENT_NOTES = True

    abstract
    abstract_id
    access_key = None
    acl_entries
    allow_relationship_preloading = True
    board_number
    can_manage (user, role=None, allow_admin=True, check_parent=True, explicit_role=False)

```

default_render_mode = 2

disallowed_protection_modes = frozenset([])

duration

duration_display

The displayed duration of the contribution.

This is the duration of the poster session if applicable, otherwise the duration of the contribution itself.

duration_poster

end_dt

end_dt_display

The displayed end time of the contribution.

This is the end time of the poster session if applicable, otherwise the end time of the contribution itself.

end_dt_poster

event

event_id

field_values

Data stored in abstract/contribution fields

friendly_id

The human-friendly ID for the contribution

get_non_inheriting_objects ()

Get a set of child objects that do not inherit protection.

id

inherit_location

inheriting_have_acl = True

is_deleted

is_paper_reviewer (user)

is_scheduled

is_user_associated (user, check_abstract=False)

keywords

location_backref_name = u'contributions'

location_parent

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```

@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}

```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

own_address

own_no_access_contact = None

own_room

own_room_id

own_room_name

own_venue

own_venue_id

own_venue_name

paper

paper_content_reviewers

Paper content reviewers

paper_judges

Paper reviewing judges

paper_layout_reviewers

Paper layout reviewers

pending_paper_files

Paper files not submitted for reviewing

person_links

Persons associated with this contribution

possible_render_modes = set([<RenderMode.html: 1>, <RenderMode.markdown: 2>])

classmethod preload_acl_entries (*event*)

protection_mode

protection_parent

references

External references associated with this contribution

render_mode

session

session_block

session_block_id

session_id

start_dt

start_dt_display

The displayed start time of the contribution.

This is the start time of the poster session if applicable, otherwise the start time of the contribution itself.

start_dt_poster**subcontribution_count****subcontributions****submitters****title****track****track_id****type****type_id****verbose_title**

class `indico.modules.events.contributions.models.contributions.CustomFieldsMixin`

Bases: `object`

Methods to process custom field data.

get_field_value (*field_id*, *raw=False*)

set_custom_field (*field_id*, *field_value*)

class `indico.modules.events.contributions.models.fields.ContributionField(**kwargs)`

Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

description**event****event_id****field****field_data****field_type****filter_choices****id****is_active****is_required****legacy_id****locator**

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

mgmt_field

position

title

class `indico.modules.events.contributions.models.fields.ContributionFieldValue` (***kwargs*)

Bases: `indico.modules.events.contributions.models.fields.ContributionFieldValueBase`

A simple constructor that allows initialization from `kwargs`.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

contribution_field

contribution_field_backref_name = u'contribution_values'

contribution_field_id

contribution_id

data

class `indico.modules.events.contributions.models.fields.ContributionFieldValueBase` (***kwargs*)

Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from `kwargs`.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

contribution_field = <RelationshipProperty at 0x7faacc300e20; no key>

contribution_field_backref_name = None

The name of the backref on the `ContributionField`

contribution_field_id

data = Column(None, JSON(astext_type=Text()), table=None, nullable=False)

friendly_data

```
class indico.modules.events.contributions.models.persons.AuthorType
```

```
    Bases: int, indico.util.struct.enum.IndicoEnum
```

```
    classmethod get_highest (*types)
```

```
    none = 0
```

```
    primary = 1
```

```
    secondary = 2
```

```
class indico.modules.events.contributions.models.persons.ContributionPersonLink (*args,  
                                                                              **kwargs)
```

```
    Bases: indico.modules.events.models.persons.PersonLinkBase
```

```
    Association between EventPerson and Contribution.
```

```
    author_type
```

```
    contribution_id
```

```
    display_order
```

```
    id
```

```
    is_author
```

```
    is_speaker
```

```
    is_submitter
```

```
    locator
```

```
    Defines a smart locator property.
```

```
    This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.
```

```
    This decorator should usually be applied to a method named locator as this name is required for get_locator to find it automatically when just passing the object.
```

```
    If you need more than one locator, you can define it like this:
```

```
@locator_property  
def locator(self):  
    return {...}  
  
@locator.other  
def locator(self):  
    return {...}
```

```
    The other locator can then be accessed by passing obj.locator.other to the code expecting an object with a locator.
```

```
    object_relationship_name = u'contribution'
```

```
    person
```

```
    person_id
```

```
    person_link_backref_name = u'contribution_links'
```

```
    person_link_unique_columns = (u'contribution_id',)
```

```
class indico.modules.events.contributions.models.persons.SubContributionPersonLink (*args,  
                                                                              **kwargs)
```

```
    Bases: indico.modules.events.models.persons.PersonLinkBase
```


Association between EventPerson and SubContribution.

```
author_type = 0
display_order
id
is_speaker = True
object_relationship_name = u'subcontribution'
person
person_id
person_link_backref_name = u'subcontribution_links'
person_link_unique_columns = (u'subcontribution_id',)
subcontribution_id
```

```
class indico.modules.events.contributions.models.principals.ContributionPrincipal (**kwargs)
Bases: indico.core.db.sqlalchemy.principals.PrincipalRolesMixin,
       flask_sqlalchemy.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
allow_emails = True
contribution_id
    The ID of the associated contribution
disallowed_protection_modes = frozenset([])
email
full_access
id
    The ID of the acl entry
ip_network_group = None
ip_network_group_id = None
local_group
local_group_id
multipass_group_name
multipass_group_provider
principal_backref_name = u'in_contribution_acls'
principal_for = u'Contribution'
read_access
roles
type
unique_columns = (u'contribution_id',)
```

user

user_id

class `indico.modules.events.contributions.models.references.ContributionReference` (**kwargs)
Bases: `indico.modules.events.models.references.ReferenceModelBase`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

contribution_id

id

reference_backref_name = u'contribution_references'

reference_type

reference_type_id

value

class `indico.modules.events.contributions.models.references.SubContributionReference` (**kwargs)
Bases: `indico.modules.events.models.references.ReferenceModelBase`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

id

reference_backref_name = u'subcontribution_references'

reference_type

reference_type_id

subcontribution_id

value

class `indico.modules.events.contributions.models.subcontributions.SubContribution` (**kwargs)
Bases: `indico.core.db.sqlalchemy.descriptions.DescriptionMixin`, `indico.core.db.sqlalchemy.attachments.AttachedItemsMixin`, `indico.core.db.sqlalchemy.notes.AttachedNotesMixin`, `flask_sqlalchemy.Model`

ATTACHMENT_FOLDER_ID_COLUMN = u'subcontribution_id'

PRELOAD_EVENT_ATTACHED_ITEMS = True

PRELOAD_EVENT_NOTES = True

can_access (*user*, **kwargs)

can_manage (*user*, *role=None*, **kwargs)

contribution_id

default_render_mode = 2

duration

event**friendly_id**

The human-friendly ID for the sub-contribution

get_access_list ()**get_manager_list** (*recursive=False*)**id****is_deleted****is_protected****location_parent****locator**

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The `other` locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

person_links

Persons associated with this contribution

position**possible_render_modes** = set([<RenderMode.html: 1>, <RenderMode.markdown: 2>])**references**

External references associated with this contribution

render_mode**session**

Convenience property so all event entities have it

speakers**timetable_entry**

Convenience property so all event entities have it

title

```
class indico.modules.events.contributions.models.types.ContributionType (**kwargs)
```

```
Bases: flask_sqlalchemy.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

description

event

event_id

id

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

name

Operations

```
indico.modules.events.contributions.operations.create_contribution(event,
                                                                    contrib_data,
                                                                    customer_data=None,
                                                                    session_block=None,
                                                                    extend_parent=False)
```

```
indico.modules.events.contributions.operations.create_contribution_from_abstract(*args,
                                                                                **kwargs)
```

```
indico.modules.events.contributions.operations.create_subcontribution(contrib,
                                                                      data)
```

```
indico.modules.events.contributions.operations.delete_contribution(contrib)
```

```
indico.modules.events.contributions.operations.delete_subcontribution(subcontrib)
```

```
indico.modules.events.contributions.operations.update_contribution(*args,
                                                                    **kwargs)
```

Update a contribution

Parameters

- **contrib** – The *Contribution* to update
- **contrib_data** – A dict containing the data to update
- **custom_fields_data** – A dict containing the data for custom fields.

Returns A dictionary containing information related to the update. *unscheduled* will be true if the modification resulted in the contribution being unscheduled. In this case *undo_unschedule* contains the necessary data to re-schedule it (undoing the session change causing it to be unscheduled)

```
indico.modules.events.contributions.operations.update_subcontribution(subcontrib,
                                                                    data)
```

Utilities

```
indico.modules.events.contributions.util.contribution_type_row(contrib_type)
```

```
indico.modules.events.contributions.util.generate_spreadsheet_from_contributions(contributions)
```

Return a tuple consisting of spreadsheet columns and respective contribution values

```
indico.modules.events.contributions.util.get_contribution_ical_file(contrib)
```

```
indico.modules.events.contributions.util.get_contributions_with_user_as_submitter(event,
                                                                                    user)
```

Get a list of contributions in which the *user* has submission rights

```
indico.modules.events.contributions.util.get_events_with_linked_contributions(user,
                                                                              dt=None)
```

Returns a dict with keys representing event_id and the values containing data about the user rights for contributions within the event

Parameters

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

```
indico.modules.events.contributions.util.has_contributions_with_user_as_submitter(event,
                                                                                    user)
```

```
indico.modules.events.contributions.util.make_contribution_form(event)
```

Extends the contribution WTForm to add the extra fields.

Each extra field will use a field named `custom_ID`.

Parameters **event** – The *Event* for which to create the contribution form.

Returns A *ContributionForm* subclass.

```
indico.modules.events.contributions.util.serialize_contribution_for_ical(contrib)
```

```
indico.modules.events.contributions.util.serialize_contribution_person_link(person_link,
                                                                              is_submitter=None)
```

Serialize ContributionPersonLink to JSON-like object

Feature**Todo**

Docstrings (module, utilities)

Utilities

`indico.modules.events.features.util.format_feature_names` (*names*)

`indico.modules.events.features.util.get_disallowed_features` (*event*)

Get a set containing the names of features which are not available for an event.

`indico.modules.events.features.util.get_enabled_features` (*event*,
only_explicit=False)

Returns a set of enabled feature names for an event

`indico.modules.events.features.util.get_feature_definition` (*name*)

Gets a feature definition

`indico.modules.events.features.util.get_feature_definitions` ()

Gets a dict containing all feature definitions

`indico.modules.events.features.util.is_feature_enabled` (*event*, *name*)

Checks if a feature is enabled for an event.

Parameters

- **event** – The event (or event ID) to check.
- **name** – The name of the feature.

`indico.modules.events.features.util.require_feature` (*event*, *name*)

Raises a NotFound error if a feature is not enabled

Parameters

- **event** – The event (or event ID) to check.
- **name** – The name of the feature.

`indico.modules.events.features.util.set_feature_enabled` (*event*, *name*, *state*)

Enables/disables a feature for an event

Parameters

- **event** – The event.
- **name** – The name of the feature.
- **state** – If the feature is enabled or not.

Returns Boolean indicating if the state of the feature changed.

Layout

Todo

Docstrings (module, models, utilities)

Models

class `indico.modules.events.layout.models.images.ImageField(**kwargs)`
 Bases: `indico.core.storage.models.StoredFileMixin`, `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

content_type
 The MIME type of the file

created_dt
 The date/time when the file was uploaded

event

event_id
 The event the image belongs to

filename
 The name of the file

id
 The ID of the file

locator

md5
 An MD5 hash of the file.
 Automatically assigned when `save()` is called.

size
 The size of the file (in bytes).
 Automatically assigned when `save()` is called.

storage_backend

storage_file_id

version_of = None

class `indico.modules.events.layout.models.menu.EventPage(**kwargs)`
 Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

event
 The Event which contains the page

event_id
 The ID of the event which contains the page

html
 The rendered HTML of the page

id
The ID of the page

is_default

locator

class `indico.modules.events.layout.models.menu.MenuEntry` (**kwargs)

Bases: `indico.modules.events.layout.models.menu.MenuEntryMixin`,
`flask_sqlalchemy.Model`

children

The children menu entries and parent backref

event

The Event containing the menu entry

event_id

The ID of the event which contains the menu

static `get_for_event` (*event*)

id

The ID of the menu entry

insert (*parent*, *position*)

is_enabled

Whether the entry is visible in the event's menu

is_root

link_url

The target URL of a custom link

move (*to*)

name

The name of the menu entry (to uniquely identify a default entry for a given event)

new_tab

Whether the menu entry should be opened in a new tab or window

page

The page of the menu entry

page_id

The page ID if the entry is a page

parent_id

The ID of the parent menu entry (NULL if root menu entry)

plugin

The name of the plugin from which the entry comes from (NULL if the entry does not come from a plugin)

position

The relative position of the entry in the menu

title

The title of the menu entry (to be displayed to the user)

type

The type of the menu entry


```

class indico.modules.events.layout.models.menu.MenuEntryMixin (**kwargs)
    Bases: object

    default_data
    event_ref
    is_internal_link
    is_link
    is_orphaned
    is_page
    is_plugin_link
    is_separator
    is_user_link
    is_visible
    localized_title
    locator
    url

class indico.modules.events.layout.models.menu.MenuEntryType
    Bases: indico.util.struct.enum.RichIntEnum

    internal_link = 2
    page = 5
    plugin_link = 4
    separator = 1
    user_link = 3

class indico.modules.events.layout.models.menu.TransientMenuEntry (event,
                                                                    is_enabled,
                                                                    name, position,
                                                                    children)
    Bases: indico.modules.events.layout.models.menu.MenuEntryMixin

    id

```

Utilities

```

class indico.modules.events.layout.util.MenuEntryData (title, name, endpoint=None,
                                                       position=-1, is_enabled=True,
                                                       visible=None, parent=None,
                                                       static_site=False)
    Bases: object

```

Container to transmit menu entry-related data via signals

The data contained is transmitted via the *sidemenu* signal and used to build the side menu of an event.

Parameters

- **title** – str – The title of the menu, displayed to the user. The title should be translated using the normal gettext function, i.e. `_(' . . . ')`, or the plugin's bound gettext function.

- **name** – str – Name used to refer to the entry internally. This is never shown to the user. The name must be unique, names from plugins are automatically prefixed with the plugin name and a colon and therefore have to be unique only within the plugin. To mark the entry as active, its name must be specified in the `menu_entry_name` class attribute of the WP class. For plugins, the plugin name must be specified via the `menu_entry_plugin` attribute as well.
- **endpoint** – str – The endpoint the entry will point to.
- **position** – int – The desired position of the menu entry. the position is indicative only, relative to the other entries and not the exact position. Entries with the same position will be sorted alphanumerically on their name. A position of `-1` will append the entry at the end of the menu.
- **is_enabled** – bool – Whether the entry should be enabled by default (Default: `True`).
- **visible** – function – Determines if the entry should be visible. This is a simple function which takes only the `event` as parameter and returns a boolean to indicate if the entry is visible or not. It is called whenever the menu is displayed, so the current state of the event/user can be taken into account.
- **parent** – str – The name of the parent entry (None for root entries).
- **static_site** – bool or str – If True, this menu item should be shown in the menu of a static site. When set to a string, the string will be used instead of a mangled version of the endpoint's URL.

name

plugin = None

visible (*event*)

`indico.modules.events.layout.util.build_menu_entry_name` (*name*, *plugin=None*)

Builds the proper name for a menu entry.

Given a menu entry's name and optionally a plugin, returns the correct name of the menu entry.

Parameters

- **name** – str – The name of the menu entry.
- **plugin** – IndicoPlugin or str – The plugin (or the name of the plugin) which created the entry.

`indico.modules.events.layout.util.get_css_file_data` (*event*)

`indico.modules.events.layout.util.get_css_url` (*event*, *force_theme=None*,
for_preview=False)

Builds the URL of a CSS resource.

Parameters

- **event** – The *Event* to get the CSS url for
- **force_theme** – The ID of the theme to override the custom CSS resource only if it exists
- **for_preview** – Whether the URL is used in the CSS preview page

Returns The URL to the CSS resource

`indico.modules.events.layout.util.get_logo_data` (*event*)

`indico.modules.events.layout.util.get_menu_entries_from_signal` (**args*,
***kwargs*)

`indico.modules.events.layout.util.get_menu_entry_by_name` (**args*, ***kwargs*)

```

indico.modules.events.layout.util.get_plugin_conference_themes()
indico.modules.events.layout.util.is_menu_entry_enabled(entry_name, event)
    Check whether the MenuEntry is enabled
indico.modules.events.layout.util.menu_entries_for_event(*args, **kwargs)

```

Log

Todo

Docstrings (module, models, utilities)

Models

```

class indico.modules.events.logs.models.entries.EventLogEntry(**kwargs)
    Bases: flask_sqlalchemy.Model

```

Log entries for events

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

data
Type-specific data

event
The Event this log entry is associated with

event_id
The ID of the event

id
The ID of the log entry

kind
The general kind of operation that was performed

logged_date

logged_dt
The date/time when the reminder was created

module
The module the operation was related to (does not need to match something in `indico.modules` and should be human-friendly but not translated).

realm
The general area of the event the entry comes from

render ()
Renders the log entry to be displayed.

If the renderer is not available anymore, e.g. because of a disabled plugin, `None` is returned.

renderer

summary

A short one-line description of the logged action. Should not be translated!

type

The type of the log entry. This needs to match the name of a log renderer.

user

The user associated with the log entry

user_id

The ID of the user associated with the entry

class `indico.modules.events.logs.models.entries.EventLogKind`

Bases: `int`, `indico.util.struct.enum.IndicoEnum`

change = 3

negative = 4

other = 1

positive = 2

class `indico.modules.events.logs.models.entries.EventLogRealm`

Bases: `indico.util.struct.enum.RichIntEnum`

emails = 5

event = 1

management = 2

participants = 3

reviewing = 4

Utilities

`indico.modules.events.logs.util.get_log_renderers()`

`indico.modules.events.logs.util.make_diff_log(changes, fields)`

Create a value for log data containing change information.

Parameters

- **changes** – a dict mapping attributes to (old, new) tuples
- **fields** – a dict mapping attributes to field metadata. for simple cases this may be a string with the human-friendly title, for more advanced fields it should be a dict containing `title`, a `type` string and a `convert` callback which will be invoked with a tuple containing the old and new value

`indico.modules.events.logs.util.render_changes(a, b, type_)`

Render the comparison of *a* and *b* as HTML.

Parameters

- **a** – old value
- **b** – new value
- **type** – the type determining how the values should be compared

class `indico.modules.events.logs.renderers.EmailRenderer`

Bases: `indico.modules.events.logs.renderers.EventLogRendererBase`

When using the class directly, pass the menu item as a posarg:

```
return WPEventManagement.render_template('foobar.html', self.event, 'foobar',
                                         foo='bar')
```

When subclassing you can set *sidemenu_option* on the class, allowing you to omit it. This is recommended if you have many pages using the same menu item or if you already need to subclass for some other reason (e.g. to set a *template_prefix* or include additional JS/CSS bundles):

```
return WPSomething.render_template('foobar.html', self.event,
                                   foo='bar')
```

Note

Todo

Docstrings (module, models, utilities)

Models

class `indico.modules.events.notes.models.notes.EventNote` (**kwargs)
Bases: `indico.core.db.sqlalchemy.links.LinkMixin`, `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

allowed_link_types = frozenset(<LinkType.event: 2>, <LinkType.contribution: 3>, <LinkType.subcontribution: 4>)

category = None

category_id = None

contribution

contribution_id

create_revision (*render_mode*, *source*, *user*)

Creates a new revision if needed and marks it as undeleted if it was

Any change to the render mode or the source causes a new revision to be created. The user is not taken into account since a user “modifying” a note without changing things is not really a change.

current_revision

The currently active revision of the note

current_revision_id

The ID of the current revision

delete (*user*)

Marks the note as deleted and adds a new empty revision

event

event_id

events_backref_name = u'all_notes'

classmethod get_for_linked_object (*linked_object*, *preload_event=True*)

Gets the note for the given object.

This only returns a note that hasn't been deleted.

Parameters

- **linked_object** – An event, session, contribution or subcontribution.
- **preload_event** – If all notes for the same event should be pre-loaded and cached in the app context.

classmethod get_or_create (*linked_object*)

Gets the note for the given object or creates a new one.

If there is an existing note for the object, it will be returned even. Otherwise a new note is created.

html

The rendered HTML of the note

id

The ID of the note

is_deleted

If the note has been deleted

link_backref_name = u'note'

link_type

linked_event

linked_event_id

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The `other` locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

revisions

The list of all revisions for the note

session

session_id

subcontribution

subcontribution_id

unique_links = True

class `indico.modules.events.notes.models.notes.EventNoteRevision` (**kwargs)

Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

created_dt

The date/time when the revision was created

html

The rendered HTML of the note

id

The ID of the revision

note_id

The ID of the associated note

render_mode

How the note is rendered

source

The raw source of the note as provided by the user

user

The user who created the revision

user_id

The user who created the revision

Utilities

`indico.modules.events.notes.util.build_note_api_data` (*note*)

`indico.modules.events.notes.util.build_note_legacy_api_data` (*note*)

`indico.modules.events.notes.util.can_edit_note` (*obj*, *user*)

Checks if a user can edit the object's note

`indico.modules.events.notes.util.get_scheduled_notes` (*event*)

Gets all notes of scheduled items inside an event

Paper

Todo

Docstrings (module, models, operations, utilities, settings)

Models

class `indico.modules.events.papers.models.call_for_papers.CallForPapers` (*event*)

Bases: `object`

Proxy class to facilitate access to the call for papers settings

announcement

assignees

can_access_judging_area (*user*)

can_access_reviewing_area (*user*)

close ()

content_review_questions

content_reviewer_deadline

content_reviewers

content_reviewing_enabled

end_dt

get_questions_for_review_type (*review_type*)

get_reviewing_state (*reviewing_type*)

has_ended

has_started

is_judge (*user*)

is_manager (*user*)

is_open

is_reviewer (*user*, *role=None*)

is_staff (*user*)

judge_deadline

judges

layout_review_questions

layout_reviewer_deadline

layout_reviewers

layout_reviewing_enabled

managers

open ()

rating_range

schedule (*start_dt*, *end_dt*)

set_reviewing_state (*reviewing_type*, *enable*)

start_dt

user_competences

```
class indico.modules.events.papers.models.comments.PaperReviewComment (**kwargs)
    Bases: indico.modules.events.models.reviews.ProposalCommentMixin, indico.
           core.db.sqlalchemy.review_comments.ReviewCommentMixin, flask_sqlalchemy.
           Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

can_edit (*user*)

can_view (*user*)

created_dt

id

is_deleted

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

modified_by

modified_by_id

modified_dt

paper_revision

render_mode = 2

revision_id

user

user_backref_name = u'review_comments'

user_id

user_modified_backref_name = u'modified_review_comments'

visibility

class `indico.modules.events.papers.models.competences.PaperCompetence` (**kwargs)
 Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

competences

event

event_id

id

user

user_id

class `indico.modules.events.papers.models.files.PaperFile` (*args, **kwargs)
 Bases: `indico.core.storage.models.StoredFileMixin`, `flask_sqlalchemy.Model`

add_file_date_column = False

content_type

The MIME type of the file

created_dt = None

filename

The name of the file

id

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The `other` locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

md5

An MD5 hash of the file.

Automatically assigned when `save()` is called.

paper

paper_revision

revision_id

size

The size of the file (in bytes).

Automatically assigned when *save()* is called.

storage_backend

storage_file_id

class `indico.modules.events.papers.models.papers.Paper` (*contribution*)

Bases: `indico.modules.events.models.reviews.ProposalMixin`

Proxy class to facilitate access to all paper-related properties

accepted_revision

call_for_proposals_attr = `u'cfp'`

can_comment (*user*, *check_state=False*)

can_judge (*user*, *check_state=False*)

can_manage (*user*)

can_review (*user*, *check_state=False*)

can_submit (*user*)

create_comment_endpoint = `u'papers.submit_comment'`

create_judgment_endpoint = `u'papers.judge_paper'`

create_review_endpoint = `u'papers.submit_review'`

delete_comment_endpoint = `u'papers.delete_comment'`

edit_comment_endpoint = `u'papers.edit_comment'`

edit_review_endpoint = `u'papers.edit_review'`

event

files

get_last_revision ()

get_revisions ()

is_in_final_state

judgment_comment

last_revision

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```

@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}

```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

proposal_type = u'paper'

proxied_attr = u'contribution'

reset_state()

revision_count

revisions

revisions_enabled = True

state

title

verbose_title

class `indico.modules.events.papers.models.review_questions.PaperReviewQuestion` (**kwargs)
 Bases: `indico.core.db.sqlalchemy.review_questions.ReviewQuestionMixin`,
`flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

event

event_backref_name = u'paper_review_questions'

event_id

id

is_deleted

no_score

position

text

type

class `indico.modules.events.papers.models.review_ratings.PaperReviewRating` (**kwargs)
 Bases: `indico.core.db.sqlalchemy.review_ratings.ReviewRatingMixin`,
`flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

id

question

question_class

alias of PaperReviewQuestion

question_id

review

review_class

alias of PaperReview

review_id

value

class `indico.modules.events.papers.models.reviews.PaperAction`

Bases: `indico.util.struct.enum.RichIntEnum`

accept = 1

reject = 2

to_be_corrected = 3

class `indico.modules.events.papers.models.reviews.PaperCommentVisibility`

Bases: `indico.util.struct.enum.RichIntEnum`

Most to least restrictive visibility for paper comments

contributors = 3

judges = 1

reviewers = 2

users = 4

class `indico.modules.events.papers.models.reviews.PaperJudgmentProxy` (*paper*)

Bases: `object`

Represents a timeline item for the non final judgments

created_dt

timeline_item_type = u'judgment'

class `indico.modules.events.papers.models.reviews.PaperReview` (**kwargs)

Bases: `indico.modules.events.models.reviews.ProposalReviewMixin`, `indico.core.db.sqlalchemy.descriptions.RenderModeMixin`, `flask_sqlalchemy.Model`

Represents a paper review, emitted by a layout or content reviewer

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

TIMELINE_TYPE = u'review'

can_edit (*user, check_state=False*)

```

can_view (user)
comment
created_dt
default_render_mode = 2
group_attr = u'type'
group_proxy_cls
    alias of PaperTypeProxy
id

```

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```

@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}

```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

```

modified_dt
possible_render_modes = set([<RenderMode.markdown: 2>])
proposed_action
render_mode = 2
revision
revision_attr = u'revision'
revision_id
score
type
user
user_id
visibility

```

```

class indico.modules.events.papers.models.reviews.PaperReviewType

```

```

    Bases: indico.util.struct.enum.RichIntEnum

```

```

    content = 2
    layout = 1

```

class `indico.modules.events.papers.models.reviews.PaperTypeProxy` (*group*)

Bases: `indico.modules.events.models.reviews.ProposalGroupProxy`

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

class `indico.modules.events.papers.models.revisions.PaperRevision` (**args*,
***kwargs*)

Bases: `indico.modules.events.models.reviews.ProposalRevisionMixin`, `indico.core.db.sqlalchemy.descriptions.RenderModeMixin`, `flask_sqlalchemy.Model`

default_render_mode = 2

get_reviewed_for_groups (*user*, *include_reviewed=False*)

get_reviews (*group=None*, *user=None*)

get_spotlight_file ()

get_timeline (*user=None*)

has_user_reviewed (*user*, *review_type=None*)

id

is_last_revision

judge

judge_id

judgment_comment

judgment_dt

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:


```

@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}

```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

number

paper

possible_render_modes = set([<RenderMode.markdown: 2>])

proposal_attr = u'paper'

render_mode = 2

state

submitted_dt

submitter

submitter_id

class `indico.modules.events.papers.models.revisions.PaperRevisionState`

Bases: `indico.util.struct.enum.RichIntEnum`

accepted = 2

rejected = 3

submitted = 1

to_be_corrected = 4

class `indico.modules.events.papers.models.templates.PaperTemplate` (**kwargs)

Bases: `indico.core.storage.models.StoredFileMixin`, `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

add_file_date_column = False

content_type

The MIME type of the file

created_dt = None

description

event

event_id

filename

The name of the file

id

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

md5

An MD5 hash of the file.

Automatically assigned when `save()` is called.

name**size**

The size of the file (in bytes).

Automatically assigned when `save()` is called.

storage_backend**storage_file_id**

Operations

`indico.modules.events.papers.operations.close_cfp(event)`

`indico.modules.events.papers.operations.create_comment(*args, **kwargs)`

`indico.modules.events.papers.operations.create_competences(event, user, competences)`

`indico.modules.events.papers.operations.create_paper_revision(paper, submitter, files)`

`indico.modules.events.papers.operations.create_paper_template(event, data)`

`indico.modules.events.papers.operations.create_review(paper, review_type, user, review_data, questions_data)`

`indico.modules.events.papers.operations.delete_comment(comment)`

`indico.modules.events.papers.operations.delete_paper_template(template)`

`indico.modules.events.papers.operations.judge_paper(*args, **kwargs)`

`indico.modules.events.papers.operations.open_cfp(event)`

`indico.modules.events.papers.operations.reset_paper_state(paper)`

```

indico.modules.events.papers.operations.schedule_cfp (event, start_dt, end_dt)
indico.modules.events.papers.operations.set_deadline (event, role, deadline, enforce=True)
indico.modules.events.papers.operations.set_reviewing_state (event, reviewing_type, enable)
indico.modules.events.papers.operations.update_comment (comment, text, visibility)
indico.modules.events.papers.operations.update_competences (user_competences, competences)
indico.modules.events.papers.operations.update_paper_template (template, data)
indico.modules.events.papers.operations.update_review (review, review_data, questions_data)
indico.modules.events.papers.operations.update_reviewing_roles (event, users, contributions, role, assign)
indico.modules.events.papers.operations.update_team_members (event, managers, judges, content_reviewers=None, layout_reviewers=None)

```

Utilities

```

indico.modules.events.papers.util.get_contributions_with_paper_submitted_by_user (event, user)
indico.modules.events.papers.util.get_contributions_with_user_paper_submission_rights (event, user)
indico.modules.events.papers.util.get_events_with_paper_roles (user, dt=None)
    Get the IDs and PR roles of events where the user has any kind of paper reviewing privileges.

```

Parameters

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

Returns A dict mapping event IDs to a set of roles

```

indico.modules.events.papers.util.get_user_contributions_to_review (event, user)
    Get the list of contributions where user has paper to review
indico.modules.events.papers.util.get_user_reviewed_contributions (event, user)
    Get the list of contributions where user already reviewed paper
indico.modules.events.papers.util.has_contributions_with_user_paper_submission_rights (event, user)

```

Settings

```

class indico.modules.events.settings.EventACLProxy (proxy)
    Bases: indico.core.settings.proxy.ACLProxyBase
    Proxy class for event-specific ACL settings

```

add_principal (*event*, **args*, ***kwargs*)

Adds a principal to an ACL

Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **principal** – A *User* or a *GroupProxy*

contains_user (*event*, **args*, ***kwargs*)

Checks if a user is in an ACL.

To pass this check, the user can either be in the ACL itself or in a group in the ACL.

Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **user** – A *User*

get (*event*, **args*, ***kwargs*)

Retrieves an ACL setting

Parameters

- **event** – Event (or its ID)
- **name** – Setting name

merge_users (*target*, *source*)

Replaces all ACL user entries for *source* with *target*

remove_principal (*event*, **args*, ***kwargs*)

Removes a principal from an ACL

Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **principal** – A *User* or a *GroupProxy*

set (*event*, **args*, ***kwargs*)

Replaces an ACL with a new one

Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **acl** – A set containing principals (users/groups)

class `indico.modules.events.settings.EventSettingProperty` (*proxy*, *name*, *default*=<*object* *object*>, *attr*=None)

Bases: `indico.core.settings.proxy.SettingProperty`

attr = `u'event'`

class `indico.modules.events.settings.EventSettingsProxy` (*module*, *defaults=None*,
strict=True, *acls=None*,
converters=None)

Bases: `indico.core.settings.proxy.SettingsProxyBase`

Proxy class to access event-specific settings for a certain module

acl_proxy_class

alias of `EventACLProxy`

delete (*event*, **args*, ***kwargs*)

Deletes settings.

Parameters

- **event** – Event (or its ID)
- **names** – One or more names of settings to delete

delete_all (*event*, **args*, ***kwargs*)

Deletes all settings.

Parameters **event** – Event (or its ID)

get (*event*, **args*, ***kwargs*)

Retrieves the value of a single setting.

Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **default** – Default value in case the setting does not exist

Returns The settings's value or the default value

get_all (*event*, **args*, ***kwargs*)

Retrieves all settings

Parameters

- **event** – Event (or its ID)
- **no_defaults** – Only return existing settings and ignore defaults.

Returns Dict containing the settings

query

Returns a query object filtering by the proxy's module.

set (*event*, **args*, ***kwargs*)

Sets a single setting.

Parameters

- **event** – Event (or its ID)
- **name** – Setting name
- **value** – Setting value; must be JSON-serializable

set_multi (*event*, **args*, ***kwargs*)

Sets multiple settings at once.

Parameters

- **event** – Event (or its ID)

- **items** – Dict containing the new settings

class `indico.modules.events.settings.ThemeSettingsProxy`

Bases: `object`

defaults

get_themes_for (**args, **kwargs*)

settings

themes

`indico.modules.events.settings.event_or_id` (*f*)

Payment

Todo

Docstrings (module, models, plugins)

Models

exception `indico.modules.events.payment.models.transactions.DoublePaymentTransaction`

Bases: `exceptions.Exception`

exception `indico.modules.events.payment.models.transactions.IgnoredTransactionAction`

Bases: `exceptions.Exception`

exception `indico.modules.events.payment.models.transactions.InvalidManualTransactionAction`

Bases: `exceptions.Exception`

exception `indico.modules.events.payment.models.transactions.InvalidTransactionAction`

Bases: `exceptions.Exception`

exception `indico.modules.events.payment.models.transactions.InvalidTransactionStatus`

Bases: `exceptions.Exception`

class `indico.modules.events.payment.models.transactions.PaymentTransaction` (***kwargs*)

Bases: `flask_sqlalchemy.Model`

Payment transactions

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

amount

the base amount the user needs to pay (without payment-specific fees)

classmethod `create_next` (*registration, amount, currency, action, provider=None, data=None*)

currency

the currency of the payment (ISO string, e.g. EUR or USD)

data

plugin-specific data of the payment

```

id
    Entry ID

is_manual

plugin

provider
    the provider of the payment (e.g. manual, PayPal etc.)

registration
    The associated registration

registration_id
    ID of the associated registration

render_details ()
    Renders the transaction details

status
    a TransactionStatus

timestamp
    the date and time the transaction was recorded

class indico.modules.events.payment.models.transactions.TransactionAction
    Bases: int, indico.util.struct.enum.IndicoEnum

    cancel = 2

    complete = 1

    pending = 3

    reject = 4

class indico.modules.events.payment.models.transactions.TransactionStatus
    Bases: int, indico.util.struct.enum.IndicoEnum

    cancelled = 2
        payment cancelled manually

    failed = 3
        payment attempt failed

    pending = 4
        payment on hold pending approval of merchant

    rejected = 5
        payment rejected after being pending

    successful = 1
        payment attempt succeeded

class indico.modules.events.payment.models.transactions.TransactionStatusTransition
    Bases: object

    initial_statuses = [<TransactionStatus.cancelled: 2>, <TransactionStatus.failed: 3>, <TransactionStatus.rejected: 5>]

    classmethod next (transaction, action, provider=None)

```

Utilities

`indico.modules.events.payment.util.get_active_payment_plugins(event)`

Returns a dict containing the active payment plugins of an event.

`indico.modules.events.payment.util.get_payment_plugins()`

Returns a dict containing the available payment plugins.

`indico.modules.events.payment.util.register_transaction(registration, amount, currency, action, provider=None, data=None)`

Creates a new transaction for a certain transaction action.

Parameters

- **registration** – the *Registration* associated to the transaction
- **amount** – the (strictly positive) amount of the transaction
- **currency** – the currency used for the transaction
- **action** – the *TransactionAction* of the transaction
- **provider** – the payment method name of the transaction, or ‘_manual’ if no payment method has been used
- **data** – arbitrary JSON-serializable data specific to the transaction’s provider

Plugins

class `indico.modules.events.payment.plugins.PaymentPluginMixin`

Bases: `object`

adjust_payment_form_data(*data*)

Updates the payment form data if necessary.

This method can be overridden to update e.g. the amount based on choices the user makes in the payment form or to provide additional data to the form. To do so, *data* must be modified.

Parameters *data* – a dict containing event, registration, amount, currency, settings and event_settings

can_be_modified(*user, event*)

Checks if the user is allowed to enable/disable/modify the payment method.

Parameters

- **user** – the *User* representing the user
- **event** – the *Event*

category = u’Payment’

default_settings

event_settings_form

alias of `PaymentEventSettingsFormBase`

get_event_management_url(*event, **kwargs*)

get_invalid_regforms(*event*)

Return registration forms with incompatible currencies

get_method_name (*event*)

Returns the (customized) name of the payment method.

init ()

logo_url

render_payment_form (*registration*)

Returns the payment form shown to the user.

Parameters registration – a Registration object

render_transaction_details (*transaction*)

Renders the transaction details in event management

Override this (or inherit from the template) to show more useful data such as transaction IDs

Parameters transaction – the PaymentTransaction

settings_form

alias of PaymentPluginSettingsFormBase

supports_currency (*currency*)

valid_currencies = None

Set containing all valid currencies. Set to *None* to allow all.

Person

Todo

Docstrings (module, operations)

Operations

`indico.modules.events.persons.operations.update_person` (*person, data*)

Placeholders

class `indico.modules.events.persons.placeholders.EmailPlaceholder`

Bases: `indico.util.placeholders.Placeholder`

description = `lu'Email of the person'`

name = `u'email'`

classmethod `render` (*person, event, **kwargs*)

class `indico.modules.events.persons.placeholders.EventLinkPlaceholder`

Bases: `indico.util.placeholders.Placeholder`

description = `lu'Link to the event'`

name = `u'event_link'`

classmethod `render` (*person, event, **kwargs*)

class `indico.modules.events.persons.placeholders.EventTitlePlaceholder`

Bases: `indico.util.placeholders.Placeholder`

```
description = lu'The title of the event'
```

```
name = u'event_title'
```

```
classmethod render (person, event, **kwargs)
```

```
class indico.modules.events.persons.placeholders.FirstNamePlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
description = lu'First name of the person'
```

```
name = u'first_name'
```

```
classmethod render (person, event, **kwargs)
```

```
class indico.modules.events.persons.placeholders.LastNamePlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
description = lu'Last name of the person'
```

```
name = u'last_name'
```

```
classmethod render (person, event, **kwargs)
```

```
class indico.modules.events.persons.placeholders.RegisterLinkPlaceholder
```

```
Bases: indico.util.placeholders.Placeholder
```

```
description = lu'The link for the registration page'
```

```
name = u'register_link'
```

```
classmethod render (person, event, **kwargs)
```

Registration

Todo

Docstrings (module, models, utilities, statistics)

Models

```
class indico.modules.events.registration.models.registrations.Registration (**kwargs)
```

```
Bases: flask_sqlalchemy.Model
```

Somebody's registration for an event through a registration form

A simple constructor that allows initialization from `kwargs`.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

base_price

The base registration fee (that is not specific to form items)

billable_data

can_be_modified

checked_in

Whether the person has checked in. Setting this also sets or clears `checked_in_dt`.

checked_in_dt

The date/time when the person has checked in

currency

Registration price currency

data

The registration this data is associated with

data_by_field**display_full_name**

Return the full name using the user's preferred name format.

email

The email of the registrant

event

The Event containing this registration

event_id

The ID of the event

first_name

The first name of the registrant

friendly_id

The human-friendly ID for the object

full_name

Returns the user's name in 'Firstname Lastname' notation.

classmethod get_all_for_event (*event*)

Retrieve all registrations in all registration forms of an event.

get_full_name (*last_name_first=True, last_name_upper=False, abbrev_first_name=False*)

Returns the user's in the specified notation.

If not format options are specified, the name is returned in the 'Lastname, Firstname' notation.

Note: Do not use positional arguments when calling this method. Always use keyword arguments!

Parameters

- **last_name_first** – if “lastname, firstname” instead of “firstname lastname” should be used
- **last_name_upper** – if the last name should be all-uppercase
- **abbrev_first_name** – if the first name should be abbreviated to use only the first character

get_personal_data ()**has_files****id**

The ID of the object

is_active**is_cancelled****is_deleted**

If the registration has been deleted

is_paid

Returns whether the registration has been paid for.

last_name

The last name of the registrant

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

order_by_name = (<sqlalchemy.sql.functions.Function at 0x7faac8464350; lower>, <sqlalchemy.sql.functions.Function

payment_dt

The date/time when the registration has been paid for

price

The total price of the registration.

This includes the base price, the field-specific price, and the custom price adjustment for the registrant.

Return type Decimal

price_adjustment

The price modifier applied to the final calculated price

registration_form_id

The ID of the registration form

render_base_price ()**render_price** ()**render_price_adjustment** ()**sections_with_answered_fields****state**

The state a registration is in

submitted_dt

The date/time when the registration was recorded

summary_data

Export registration data nested in sections and fields

sync_state (*_skip_moderation=True*)

Sync the state of the registration

ticket_uuid

The unique token used in tickets

transaction

The latest payment transaction associated with this registration

transaction_id

The ID of the latest payment transaction associated with this registration

update_state (*approved=None, paid=None, rejected=None, _skip_moderation=False*)

Update the state of the registration for a given action

The accepted kwargs are the possible actions. `True` means that the action occurred and `False` that it was reverted.

user**user_id**

The ID of the user who registered

uuid

The unguessable ID for the object

```
class indico.modules.events.registration.models.registrations.RegistrationData (**kwargs)
    Bases: indico.core.storage.models.StoredFileMixin, flask_sqlalchemy.Model
```

Data entry within a registration for a field in a registration form

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

add_file_date_column = False**content_type**

The MIME type of the file

created_dt = None**data**

The submitted data for the field

field_data

The associated field data object

field_data_id

The ID of the field data

file**file_required = False****filename**

The name of the file

friendly_data**get_friendly_data** (**kwargs)**locator**

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

md5

An MD5 hash of the file.

Automatically assigned when `save()` is called.

price**registration_id**

The ID of the registration

render_price()**size**

The size of the file (in bytes).

Automatically assigned when `save()` is called.

storage_backend**storage_file_id****summary_data****user_data**

class `indico.modules.events.registration.models.registrations.RegistrationState`

Bases: `indico.util.struct.enum.RichIntEnum`

complete = 1

pending = 2

rejected = 3

unpaid = 5

withdrawn = 4

class `indico.modules.events.registration.models.form_fields.RegistrationFormField(**kwargs)`

Bases: `indico.modules.events.registration.models.items.RegistrationFormItem`

A registration form field

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

calculate_price (*registration_data*)

children

current_data

current_data_id

data

data_versions

description

field_impl

Gets the implementation of the field.

Returns An instance of a *RegistrationFormFieldBase* subclass

get_friendly_data (*registration_data*, ****kwargs**)

html_field_name

id

input_type

is_deleted

is_enabled

is_manager_only

is_required

locator

parent_id

personal_data_type

position

registration_form_id

title

type

versioned_data

view_data

class `indico.modules.events.registration.models.form_fields.RegistrationFormFieldData` (****kwargs**)

Bases: `flask_sqlalchemy.Model`

Description of a registration form field

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

field_id

The ID of the registration form field

id
The ID of the object

versioned_data
Data describing the field

class `indico.modules.events.registration.models.form_fields.RegistrationFormPersonalDataField`

Bases: `indico.modules.events.registration.models.form_fields.RegistrationFormField`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

children

current_data

current_data_id

data

data_versions

description

html_field_name

id

input_type

is_deleted

is_enabled

is_manager_only

is_required

parent_id

personal_data_type

position

registration_form_id

title

type

view_data

class `indico.modules.events.registration.models.forms.ModificationMode`

Bases: `indico.util.struct.enum.RichIntEnum`

allowed_always = 1

allowed_until_payment = 2

not_allowed = 3

class `indico.modules.events.registration.models.forms.RegistrationForm(**kwargs)`

Bases: `flask_sqlalchemy.Model`

A registration form for an event

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

active_fields

active_registrations

base_price

The base fee users have to pay when registering

can_submit (*user*)

contact_info

Contact information for registrants

currency

Currency for prices in the registration form

disabled_sections

end_dt

Datetime when the registration form is closed

event

The Event containing this registration form

event_id

The ID of the event

form_items

get_personal_data_field_id (*personal_data_type*)

Returns the field id corresponding to the personal data field with the given name.

get_registration (**args, **kwargs*)

Retrieves registrations for this registration form by user or uuid

has_ended

has_started

id

The ID of the object

introduction

invitations

The registration invitations associated with this form

is_active

is_deleted

Whether the registration has been marked as deleted

is_modification_allowed (*registration*)

Checks whether a registration may be modified

is_modification_open

is_open

is_participation
Whether it's the 'Participants' form of a meeting/lecture

is_scheduled

limit_reached

locator

manager_notification_recipients
List of emails that should receive management notifications

manager_notifications_enabled
Whether the manager notifications for this event are enabled

message_complete
Custom message to include in emails for complete registrations

message_pending
Custom message to include in emails for pending registrations

message_unpaid
Custom message to include in emails for unpaid registrations

moderation_enabled
Whether registrations must be approved by a manager

modification_end_dt
Datetime when the modification period is over

modification_mode
Whether registration modifications are allowed

notification_sender_address
Notifications sender address

publish_checkin_enabled
Whether checked-in status should be displayed in the event pages and participant list

publish_registration_count
Whether to display the number of registrations

publish_registrations_enabled
Whether registrations should be displayed in the participant list

registration_limit
Maximum number of registrations allowed

registrations
The registrations associated with this form

render_base_price ()

require_login
Whether users must be logged in to register

require_user
Whether registrations must be associated with an Indico account

sections

sender_address

start_dt

Datetime when the registration form is open

ticket_on_email

Whether to send tickets by e-mail

ticket_on_event_page

Whether to show a ticket download link on the event homepage

ticket_on_summary_page

Whether to show a ticket download link on the registration summary page

ticket_template

The template used to generate tickets

ticket_template_id

The ID of the template used to generate tickets

tickets_enabled

Whether tickets are enabled for this form

title

The title of the registration form

class `indico.modules.events.registration.models.invitations.InvitationState`

Bases: `indico.util.struct.enum.RichIntEnum`

accepted = 1

declined = 2

pending = 0

class `indico.modules.events.registration.models.invitations.RegistrationInvitation` (**kwargs)

Bases: `flask_sqlalchemy.Model`

An invitation for someone to register

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

affiliation

The affiliation of the invited person

email

The email of the invited person

first_name

The first name of the invited person

id

The ID of the invitation

last_name

The last name of the invited person

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

registration

The associated registration

registration_form_id

The ID of the registration form

registration_id

The ID of the registration (if accepted)

skip_moderation

Whether registration moderation should be skipped

state

The state of the invitation

uuid

The UUID of the invitation

class `indico.modules.events.registration.models.items.PersonalDataType`

Bases: `int`, `indico.util.struct.enum.IndicoEnum`

Description of the personal data items that exist on every registration form

FIELD_DATA = [(`<PersonalDataType.title: 5>`, {`u'input_type': u'single_choice'`, `u'data': {u'item_type': u'dropdown', u'`

address = 6

affiliation = 4

column

The Registration column in which the value is stored in addition to the regular registration data entry.

country = 8

email = 1

first_name = 2

get_title()

is_required

last_name = 3

phone = 7

position = 9

title = 5

class `indico.modules.events.registration.models.items.RegistrationFormItem(**kwargs)`
 Bases: `flask_sqlalchemy.Model`

Generic registration form item

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

children

current_data

The latest value of the field

current_data_id

The ID of the latest data

data

unversioned field data

data_versions

The list of all versions of the field data

description

Description of this field

id

The ID of the object

input_type

input type of this field

is_deleted

Whether field has been "deleted"

is_enabled

Whether the field is enabled

is_field

is_manager_only

if the section is only accessible to managers

is_required

determines if the field is mandatory

is_section

is_visible

parent_id

The ID of the parent form item

personal_data_type

The type of a personal data field

position

registration_form_id

The ID of the registration form

title

The title of this field

type
The type of the registration form item

view_data
Returns object with data that Angular can understand

class `indico.modules.events.registration.models.items.RegistrationFormItemType`
Bases: `int`, `indico.util.struct.enum.IndicoEnum`

field = 2

field_pd = 5

section = 1

section_pd = 4

text = 3

class `indico.modules.events.registration.models.items.RegistrationFormPersonalDataSection` (**/k
Bases: `indico.modules.events.registration.models.items.RegistrationFormSection`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

children

current_data

current_data_id

data

data_versions

description

id

input_type

is_deleted

is_enabled

is_manager_only

is_required

parent_id

personal_data_type

position

registration_form_id

title

type

view_data

class `indico.modules.events.registration.models.items.RegistrationFormSection` (**kwargs)
 Bases: `indico.modules.events.registration.models.items.RegistrationFormItem`

Registration form section that can contain fields and text

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

active_fields

children

current_data

current_data_id

data

data_versions

description

fields

id

input_type

is_deleted

is_enabled

is_manager_only

is_required

locator

own_data

parent_id

personal_data_type

position

registration_form_id

title

type

view_data

class `indico.modules.events.registration.models.items.RegistrationFormText` (**kwargs)
 Bases: `indico.modules.events.registration.models.items.RegistrationFormItem`

Text to be displayed in registration form sections

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

`children`
`current_data`
`current_data_id`
`data`
`data_versions`
`description`
`id`
`input_type`
`is_deleted`
`is_enabled`
`is_manager_only`
`is_required`
`locator`
`parent_id`
`personal_data_type`
`position`
`registration_form_id`
`title`
`type`
`view_data`

Utilities

`indico.modules.events.registration.util.build_registration_api_data` (*registration*)

`indico.modules.events.registration.util.build_registrations_api_data` (*event*)

`indico.modules.events.registration.util.check_registration_email` (*regform*,
email,
registration=None,
management=False)

Checks whether an email address is suitable for registration.

Parameters

- **regform** – The registration form
- **email** – The email address
- **registration** – The existing registration (in case of modification)
- **management** – If it's a manager adding a new registration

`indico.modules.events.registration.util.create_personal_data_fields` (*regform*)

Creates the special section/fields for personal data.

`indico.modules.events.registration.util.create_registration` (*regform*, *data*, *invitation=None*, *management=False*, *notify_user=True*)

`indico.modules.events.registration.util.generate_spreadsheet_from_registrations` (*registrations*, *regform_items*, *static_items*)

Generates a spreadsheet data from a given registration list.

Parameters

- **registrations** – The list of registrations to include in the file
- **regform_items** – The registration form items to be used as columns
- **static_items** – Registration form information as extra columns

`indico.modules.events.registration.util.generate_ticket_qr_code` (*registration*)
Generate a Pillow *Image* with a QR Code encoding a check-in ticket.

Parameters *registration* – corresponding *Registration* object

`indico.modules.events.registration.util.get_event_regforms` (*event*, *user*, *with_registrations=False*)

Get registration forms with information about user registrations.

Parameters

- **event** – the *Event* to get registration forms for
- **user** – A *User*
- **with_registrations** – Whether to return the user's registration instead of just whether they have one

`indico.modules.events.registration.util.get_event_section_data` (*regform*, *management=False*, *registration=None*)

`indico.modules.events.registration.util.get_events_registered` (*user*, *dt=None*)
Gets the IDs of events where the user is registered.

Parameters

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

Returns A set of event ids

`indico.modules.events.registration.util.get_published_registrations` (*event*)
Get a list of published registrations for an event.

Parameters *event* – the *Event* to get registrations for

Returns list of *Registration* objects

`indico.modules.events.registration.util.get_registrations_with_tickets` (*user*, *event*)

`indico.modules.events.registration.util.get_title_uuid` (*regform*, *title*)
Convert a string title to its UUID value

If the title does not exist in the title PD field, it will be ignored and returned as *None*.

```
indico.modules.events.registration.util.make_registration_form(regform, man-  
agement=False,  
registra-  
tion=None)
```

Creates a WtForm based on registration form fields

```
indico.modules.events.registration.util.modify_registration(registration, data,  
management=False,  
notify_user=True)
```

```
indico.modules.events.registration.util.url_rule_to_angular(endpoint)
```

Converts a flask-style rule to angular style

Placeholders

```
class indico.modules.events.registration.placeholders.registrations.EventLinkPlaceholder  
Bases: indico.util.placeholders.Placeholder
```

```
description = lu'Link to the event'
```

```
name = u'event_link'
```

```
classmethod render (regform, registration)
```

```
class indico.modules.events.registration.placeholders.registrations.EventTitlePlaceholder  
Bases: indico.util.placeholders.Placeholder
```

```
description = lu'The title of the event'
```

```
name = u'event_title'
```

```
classmethod render (regform, registration)
```

```
class indico.modules.events.registration.placeholders.registrations.FieldPlaceholder  
Bases: indico.util.placeholders.ParametrizedPlaceholder
```

```
advanced = True
```

```
description = None
```

```
classmethod iter_param_info (regform, registration)
```

```
name = u'field'
```

```
param_required = True
```

```
param_restricted = True
```

```
classmethod render (param, regform, registration)
```

```
class indico.modules.events.registration.placeholders.registrations.FirstNamePlaceholder  
Bases: indico.util.placeholders.Placeholder
```

```
description = lu'First name of the person'
```

```
name = u'first_name'
```

```
classmethod render (regform, registration)
```

```
class indico.modules.events.registration.placeholders.registrations.IDPlaceholder  
Bases: indico.util.placeholders.Placeholder
```

```
description = lu'The ID of the registration'
```

```
name = u'id'
```

classmethod render (*regform, registration*)

class `indico.modules.events.registration.placeholders.registrations.LastNamePlaceholder`
 Bases: `indico.util.placeholders.Placeholder`

description = `lu'Last name of the person'`

name = `u'last_name'`

classmethod render (*regform, registration*)

class `indico.modules.events.registration.placeholders.registrations.LinkPlaceholder`
 Bases: `indico.util.placeholders.Placeholder`

description = `lu'The link to the registration details'`

name = `u'link'`

classmethod render (*regform, registration*)

class `indico.modules.events.registration.placeholders.invitations.FirstNamePlaceholder`
 Bases: `indico.util.placeholders.Placeholder`

description = `lu'First name of the person'`

name = `u'first_name'`

classmethod render (*invitation*)

class `indico.modules.events.registration.placeholders.invitations.InvitationLinkPlaceholder`
 Bases: `indico.util.placeholders.Placeholder`

description = `lu'Link to accept/decline the invitation'`

name = `u'invitation_link'`

classmethod render (*invitation*)

required = `True`

class `indico.modules.events.registration.placeholders.invitations.LastNamePlaceholder`
 Bases: `indico.util.placeholders.Placeholder`

description = `lu'Last name of the person'`

name = `u'last_name'`

classmethod render (*invitation*)

Settings

class `indico.modules.events.registration.settings.RegistrationSettingsProxy` (*module, de-faults=None, strict=True, acls=None, converters=None*)

Bases: `indico.modules.events.settings.EventSettingsProxy`

Store per-event registration settings

get_participant_list_columns (*event, form=None*)

```
get_participant_list_form_ids (event)
set_participant_list_columns (event, columns, form=None)
set_participant_list_form_ids (event, form_ids)
```

Statistics

class `indico.modules.events.registration.stats.AccommodationStats` (*field*)
Bases: `indico.modules.events.registration.stats.FieldStats`, `indico.modules.events.registration.stats.StatsBase`

class `indico.modules.events.registration.stats.Cell`
Bases: `indico.modules.events.registration.stats.Cell`

Hold data and type for a cell of a stats table

The table below indicates the valid types and expected data.

type	data
<i>str</i>	<i>str</i> – string value
<i>progress</i>	(<i>int</i> , <i>str</i>) – a tuple with the progress (a value between 0 and 1) and a label
<i>progress-stacked</i>	(<i>[int]</i> , <i>str</i>) – a tuple with a list of progresses (values which must sum up to 1) and a label
<i>currency</i>	<i>float</i> – numeric value
<i>icon</i>	<i>str</i> – icon name from <code>_icons.scss</code>
<i>default</i>	<i>None</i> – renders a default cell with an <code>&mdash;</code> ; (use <code>Cell(type='str')</code> for an empty cell)

Parameters

- **type** – *str* – The type of data in the cell
- **data** – The data for the cell
- **colspan** – *int* – HTML colspan value for the cell
- **classes** – [*str*] – HTML classes to apply to the cell
- **qtip** – *str* – content for qtip

class `indico.modules.events.registration.stats.DataItem`
Bases: `indico.modules.events.registration.stats.DataItem`

Holds the aggregation of some data, intended for stats tables as a aggregation from which to generate cells.

Parameters

- **regs** – *int* – number of registrant
- **attendance** – *int* – number of people attending
- **capacity** – *int* – maximum number of people allowed to attend (0 if unlimited)
- **billable** – *bool* – whether the item is billable to the or not
- **cancelled** – *bool* – whether the item is cancelled or not
- **price** – *str* – the price of the item
- **fixed_price** – *bool* – *True* if the price is per registrant, *False* if accompanying guests must pay as well.
- **paid** – *int* – number of registrants who paid
- **paid_amount** – *float* – amount already paid by registrants

- **unpaid** – int – number of registrants who haven't paid
- **unpaid_amount** – float – amount not already paid by registrants

class `indico.modules.events.registration.stats.FieldStats` (*field*, ***kwargs*)

Bases: `object`

Holds stats for a registration form field

get_table ()

Returns a table containing the stats for each item.

Returns dict – A table with a list of head cells (key: *'head'*) and a list of rows (key: *'rows'*) where each row is a list of cells.

is_currency_shown

class `indico.modules.events.registration.stats.OverviewStats` (*regform*)

Bases: `indico.modules.events.registration.stats.StatsBase`

Generic stats for a registration form

class `indico.modules.events.registration.stats.StatsBase` (*title*, *subtitle*, *type*, ***kwargs*)

Bases: `object`

Base class for registration form statistics

Parameters

- **title** – str – the title for the stats box
- **subtitle** – str – the subtitle for the stats box
- **type** – str – the type used in Jinja to display the stats

is_currency_shown

Reminder

Todo

Docstrings (module)

Models

class `indico.modules.events.reminders.models.reminders.EventReminder` (***kwargs*)

Bases: `flask_sqlalchemy.Model`

Email reminders for events

A simple constructor that allows initialization from `kwargs`.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

all_recipients

Returns all recipients of the notifications.

This includes both explicit recipients and, if enabled, participants of the event.

created_dt

The date/time when the reminder was created

creator

The user who created the reminder

creator_id

The ID of the user who created the reminder

event

The Event this reminder is associated with

event_id

The ID of the event

event_start_delta

How long before the event start the reminder should be sent This is needed to update the *scheduled_dt* when changing the start time of the event.

id

The ID of the reminder

include_summary

If the notification should include a summary of the event's schedule.

is_overdue

is_relative

Returns if the reminder is relative to the event time

is_sent

If the reminder has been sent

locator

message

Custom message to include in the email

recipients

The recipients of the notification

reply_to_address

The address to use as Reply-To in the notification email.

scheduled_dt

The date/time when the reminder should be sent

send ()

Sends the reminder to its recipients.

send_to_participants

If the notification should also be sent to all event participants

Utilities

`indico.modules.events.reminders.util.make_reminder_email (event, with_agenda, note)`

Returns the template module for the reminder email.

Parameters

- **event** – The event

- **with_agenda** – If the event’s agenda should be included
- **note** – A custom message to include in the email

Request

Todo

Docstrings (module)

Models

class `indico.modules.events.requests.models.requests.Request` (**kwargs)

Bases: `flask_sqlalchemy.Model`

Event-related requests, e.g. for a webcast

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance’s class are allowed. These could be, for example, any mapped columns or relationships.

can_be_modified

Determines if the request can be modified or if a new one must be sent

comment

an optional comment for an accepted/rejected request

created_by_id

ID of the user creating the request

created_by_user

The user who created the request

created_dt

the date/time the request was created

data

plugin-specific data of the payment

definition

event

The Event this agreement is associated with

event_id

ID of the event

classmethod `find_latest_for_event` (*event*, *type_=None*)

Returns the latest requests for a given event.

Parameters

- **event** – the event to find the requests for
- **type** – the request type to retrieve, or *None* to get all

Returns a dict mapping request types to a *Request* or if *type_* was specified, a single *Request* or *None*

id
request ID

locator

processed_by_id
ID of the user processing the request

processed_by_user
The user who processed the request

processed_dt
the date/time the request was accepted/rejected

state
the requests's date, a *RequestState* value

type
the request type name

class `indico.modules.events.requests.models.requests.RequestState`
Bases: `indico.util.struct.enum.RichIntEnum`

accepted = 1

pending = 0

rejected = 2

withdrawn = 3

Utilities

`indico.modules.events.requests.util.get_request_definitions()`
Returns a dict of request definitions

`indico.modules.events.requests.util.is_request_manager(user)`
Checks if the user manages any request types

class `indico.modules.events.requests.base.RequestDefinitionBase`
Bases: `object`

Defines a service request which can be sent by event managers.

classmethod `accept(req, data, user)`
Accepts the request

To ensure that additional data is saved, this method should call **:method:'manager_save'**.

Parameters

- **req** – the *Request* of the request
- **data** – the form data from the management form
- **user** – the user processing the request

classmethod `can_be_managed(user)`
Checks whether the user is allowed to manage this request type

Parameters `user` – a *User*

classmethod `create_form` (*event*, *existing_request=None*)

Creates the request form

Parameters

- **event** – the event the request is for
- **existing_request** – the `Request` if there's an existing request of this type

Returns an instance of an `IndicoForm` subclass

classmethod `create_manager_form` (*req*)

Creates the request management form

Parameters **req** – the `Request` of the request

Returns an instance of an `IndicoForm` subclass

form = None

the `IndicoForm` to use for the request form

form_defaults = {}

default values to use if there's no existing request

classmethod `get_manager_notification_emails` ()

Returns the email addresses of users who manage requests of this type

The email addresses are used only for notifications. It usually makes sense to return the email addresses of the users who pass the `:method:'can_be_managed'` check.

Returns set of email addresses

classmethod `get_notification_template` (*name*, ***context*)

Gets the template module for a notification email

Parameters

- **name** – the template name
- **context** – data passed to the template

manager_form

the `IndicoForm` to use for the request manager form

alias of `RequestManagerForm`

classmethod `manager_save` (*req*, *data*)

Saves management-specific data

This method is called when the management form is submitted without accepting/rejecting the request (which is guaranteed to be already accepted or rejected).

Parameters

- **req** – the `Request` of the request
- **data** – the form data from the management form

name = None

the unique internal name of the request type

plugin = None

the plugin containing this request definition - assigned automatically

classmethod `reject` (*req*, *data*, *user*)

Rejects the request

To ensure that additional data is saved, this method should call `:method:'manager_save'`.

Parameters

- **req** – the Request of the request
- **data** – the form data from the management form
- **user** – the user processing the request

classmethod `render_form` (*event*, ***kwargs*)

Renders the request form

Parameters

- **event** – the event the request is for
- **kwargs** – arguments passed to the template

classmethod `send` (*req*, *data*)

Sends a new/modified request

Parameters

- **req** – the Request of the request
- **data** – the form data from the request form

title = None

the title of the request type as shown to users

classmethod `withdraw` (*req*, *notify_event_managers=True*)

Withdraws the request

Parameters

- **req** – the Request of the request
- **notify_event_managers** – if event managers should be notified

Session

Todo

Docstrings (module, models, operations, utilities)

Models

class `indico.modules.events.sessions.models.sessions.Session` (***kwargs*)

Bases: `indico.core.db.sqlalchemy.descriptions.DescriptionMixin`, `indico.core.db.sqlalchemy.colors.ColorMixin`, `indico.core.db.sqlalchemy.protection.ProtectionManagersMixin`, `indico.core.db.sqlalchemy.locations.LocationMixin`, `indico.core.db.sqlalchemy.attachments.AttachedItemsMixin`, `indico.core.db.sqlalchemy.notes.AttachedNotesMixin`, `flask_sqlalchemy.Model`

ATTACHMENT_FOLDER_ID_COLUMN = u'session_id'

PRELOAD_EVENT_ATTACHED_ITEMS = True

PRELOAD_EVENT_NOTES = True

access_key = None

acl_entries

allow_relationship_preloading = True

background_color

blocks

can_manage_blocks (*user*, *allow_admin=True*)
 Check whether a user can manage session blocks.

This only applies to the blocks themselves, not to contributions inside them.

can_manage_contributions (*user*, *allow_admin=True*)
 Check whether a user can manage contributions within the session.

code

conveners

default_colors = ColorTuple(text=u'202020', background=u'e3f2d3')

default_contribution_duration

default_render_mode = 2

disallowed_protection_modes = frozenset([])

end_dt

event

event_id

friendly_id
 The human-friendly ID for the session

get_non_inheriting_objects ()
 Get a set of child objects that do not inherit protection

id

inherit_location

inheriting_have_acl = True

is_deleted

is_poster

location_backref_name = u'sessions'

location_parent

locator
 Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

own_address

own_no_access_contact = None

own_room

own_room_id

own_room_name

own_venue

own_venue_id

own_venue_name

possible_render_modes = set([<RenderMode.markdown: 2>])

classmethod preload_acl_entries (*event*)

protection_mode

protection_parent

render_mode = 2

session

Convenience property so all event entities have it

start_dt

text_color

title

class `indico.modules.events.sessions.models.blocks.SessionBlock` (**kwargs)
Bases: `indico.core.db.sqlalchemy.locations.LocationMixin`, `flask_sqlalchemy.Model`

can_access (*user*, *allow_admin=True*)

can_edit_note (*user*)

can_manage (*user*, *allow_admin=True*)

can_manage_attachments (*user*)

contribution_count

duration

end_dt

event

full_title

has_note

id

inherit_location

location_backref_name = u'session_blocks'

location_parent

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```

@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}

```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

note

own_address

own_room

own_room_id

own_room_name

own_venue

own_venue_id

own_venue_name

person_links

Persons associated with this session block

session_id

start_dt

title

class `indico.modules.events.sessions.models.persons.SessionBlockPersonLink` (*args, **kwargs)

Bases: `indico.modules.events.models.persons.PersonLinkBase`

Association between EventPerson and SessionBlock.

Also known as a 'session convener'

display_order

```
id
object_relationship_name = u'session_block'
person
person_id
person_link_backref_name = u'session_block_links'
person_link_unique_columns = (u'session_block_id',)
session_block_id
```

```
class indico.modules.events.sessions.models.principals.SessionPrincipal (**kwargs)
    Bases: indico.core.db.sqlalchemy.principals.PrincipalRolesMixin,
           flask_sqlalchemy.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
allow_emails = True
disallowed_protection_modes = frozenset(())
email
full_access
id
    The ID of the acl entry
ip_network_group = None
ip_network_group_id = None
local_group
local_group_id
multipass_group_name
multipass_group_provider
principal_backref_name = u'in_session_acls'
principal_for = u'Session'
read_access
roles
session_id
    The ID of the associated session
type
unique_columns = (u'session_id',)
user
user_id
```

Operations

`indico.modules.events.sessions.operations.create_session(event, data)`
 Create a new session with the information passed in the *data* argument

`indico.modules.events.sessions.operations.create_session_block(session_, data)`

`indico.modules.events.sessions.operations.delete_session(event_session)`
 Delete session from the event

`indico.modules.events.sessions.operations.delete_session_block(session_block)`

`indico.modules.events.sessions.operations.update_session(event_session, data)`
 Update a session based on the information in the *data*

`indico.modules.events.sessions.operations.update_session_block(session_block, data)`
 Update a session block with data passed in the *data* argument

`indico.modules.events.sessions.operations.update_session_coordinator_privs(event, data)`

Utilities

class `indico.modules.events.sessions.util.SessionListToPDF(conf, sessions)`
 Bases: `indico.legacy.pdfinterface.base.PDFBase`

getBody (*story=None*)

`indico.modules.events.sessions.util.can_manage_sessions(user, event, role=None)`
 Check whether a user can manage any sessions in an event

`indico.modules.events.sessions.util.generate_pdf_from_sessions(event, sessions)`
 Generate a PDF file from a given session list

`indico.modules.events.sessions.util.generate_spreadsheet_from_sessions(sessions)`
 Generate spreadsheet data from a given session list.

Parameters **sessions** – The sessions to include in the spreadsheet

`indico.modules.events.sessions.util.get_events_with_linked_sessions(user, dt=None)`
 Returns a dict with keys representing event_id and the values containing data about the user rights for sessions within the event

Parameters

- **user** – A User
- **dt** – Only include events taking place on/after that date

`indico.modules.events.sessions.util.get_session_ical_file(sess)`

`indico.modules.events.sessions.util.get_session_timetable_pdf(sess, **kwargs)`

`indico.modules.events.sessions.util.get_sessions_for_user(event, user)`

`indico.modules.events.sessions.util.has_sessions_for_user(event, user)`

`indico.modules.events.sessions.util.serialize_session_for_ical(sess)`

`indico.modules.events.sessions.util.session_coordinator_priv_enabled(event, priv)`
 Check whether a coordinator privilege is enabled.

Currently the following privileges are available:

- manage-contributions
- manage-blocks

Parameters

- **event** – The *Event* to check for
- **priv** – The name of the privilege

Survey

Todo

Docstrings (module, models)

Models

class `indico.modules.events.surveys.models.surveys.Survey (**kwargs)`

Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

anonymous

Whether submissions will not be linked to a user

can_submit (*user*)

close ()

end_dt

Datetime when the survey is closed

event

The Event containing this survey

event_id

The ID of the event

has_ended

has_started

id

The ID of the survey

introduction

is_active

is_deleted

Whether the survey has been marked as deleted

is_visible

items

The list of items

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

new_submission_emails

Email addresses to notify about new submissions

notifications_enabled

Whether to send survey related notifications to users

notify_participants

Whether include Participants / Registrants when sending start notifications

open()**partial_completion**

Whether answers can be saved without submitting the survey

private**questions**

The list of questions

require_user

Whether submissions must be done by logged users

sections

The list of sections

send_start_notification()**send_submission_notification(submission)****start_dt**

Datetime when the survey is open

start_notification_emails

Email addresses to notify about the start of a survey

start_notification_recipients

Returns all recipients of the notifications.

This includes both explicit recipients and, if enabled, participants of the event.

start_notification_sent
Whether start notification has been already sent

state

submission_limit
Maximum number of submissions allowed

submissions
The list of submissions

title
The title of the survey

uuid

class `indico.modules.events.surveys.models.surveys.SurveyState`
Bases: `indico.util.struct.enum.IndicoEnum`

active_and_answered = 4

active_and_clean = 3

finished = 5

not_ready = 1

ready_to_open = 2

class `indico.modules.events.surveys.models.items.SurveyItem(**kwargs)`
Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

description
The description of the item

display_as_section
If a section should be rendered as a section

field_data
Field-specific data (such as choices for multi-select fields)

field_type
The type of the field used for the question

id
The ID of the item

is_required
If the question must be answered (wtforms `DataRequired`)

parent_id
The ID of the parent section item (NULL for top-level items, i.e. sections)

position
The position of the item in the survey form

survey_id
The ID of the survey

title
The title of the item

to_dict ()
Return a json-serializable representation of this object.
Subclasses must add their own data to the dict.

type
The type of the survey item

class `indico.modules.events.surveys.models.items.SurveyItemType`
Bases: `int`, `indico.util.struct.enum.IndicoEnum`

question = 1

section = 2

text = 3

class `indico.modules.events.surveys.models.items.SurveyQuestion (**kwargs)`
Bases: `indico.modules.events.surveys.models.items.SurveyItem`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

description

display_as_section

field

field_data

field_type

get_summary (kwargs)**

Returns the summary of answers submitted for this question.

id

is_required

locator

not_empty_answers

parent_id

position

survey_id

title

to_dict ()

type

class `indico.modules.events.surveys.models.items.SurveySection (**kwargs)`
Bases: `indico.modules.events.surveys.models.items.SurveyItem`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

children

The child items of this section

description**display_as_section****field_data****field_type****id****is_required****locator****parent_id****position****survey_id****title****to_dict ()****type**

class `indico.modules.events.surveys.models.items.SurveyText (**kwargs)`

Bases: `indico.modules.events.surveys.models.items.SurveyItem`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

description**display_as_section****field_data****field_type****id****is_required****locator****parent_id****position****survey_id****title****to_dict ()****type**

class `indico.modules.events.surveys.models.submissions.SurveyAnswer` (**kwargs)
 Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

answer_data

data

The user's answer (no, not 42!) to the question

is_empty

question

The list of answers

question_id

The ID of the question

submission_id

The ID of the submission

class `indico.modules.events.surveys.models.submissions.SurveySubmission` (**kwargs)
 Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

answers

The list of answers

friendly_id

The human-friendly ID of the submission

id

The ID of the submission

is_anonymous

Whether the survey submission is anonymous

is_submitted

Whether the survey was submitted

locator

pending_answers

List of non-submitted answers

submitted_dt

The date/time when the survey was submitted

survey_id

The ID of the survey

user

The user who submitted the survey

user_id

The ID of the user who submitted the survey

Operations

`indico.modules.events.surveys.operations.add_survey_question` (*section*, *field_cls*,
data)

Add a question to a survey.

Parameters

- **section** – The *SurveySection* to which the question will be added.
- **field_cls** – The field class of this question.
- **data** – The *FieldConfigForm.data* to populate the question with.

Returns The added *SurveyQuestion*.

`indico.modules.events.surveys.operations.add_survey_section` (*survey*, *data*)

Add a section to a survey.

Parameters

- **survey** – The *Survey* to which the section will be added.
- **data** – Attributes of the new *SurveySection*.

Returns The added *SurveySection*.

`indico.modules.events.surveys.operations.add_survey_text` (*section*, *data*)

Add a text item to a survey.

Parameters

- **section** – The *SurveySection* to which the question will be added.
- **data** – The *TextForm.data* to populate the question with.

Returns The added *SurveyText*.

Utilities

`indico.modules.events.surveys.util.generate_spreadsheet_from_survey` (*survey*,
submis-
sion_ids)

Generates spreadsheet data from a given survey.

Parameters

- **survey** – *Survey* for which the user wants to export submissions
- **submission_ids** – The list of submissions to include in the file

`indico.modules.events.surveys.util.get_events_with_submitted_surveys` (*user*,
dt=None)

Gets the IDs of events where the user submitted a survey.

Parameters

- **user** – A *User*
- **dt** – Only include events taking place on/after that date

Returns A set of event ids

`indico.modules.events.surveys.util.is_submission_in_progress(survey)`

Check whether the current user has a survey submission in progress

`indico.modules.events.surveys.util.make_survey_form(survey)`

Creates a WTForm from survey questions.

Each question will use a field named `question_ID`.

Parameters `survey` – The *Survey* for which to create the form.

Returns An *IndicoForm* subclass.

`indico.modules.events.surveys.util.query_active_surveys(event)`

`indico.modules.events.surveys.util.save_submitted_survey_to_session(submission)`

Save submission of a survey to session for further checks

`indico.modules.events.surveys.util.was_survey_submitted(*args, **kwargs)`

Check whether the current user has submitted a survey

Timetable

Todo

Docstring (module, models, operations, utilities)

Models

class `indico.modules.events.timetable.models.breaks.Break(**kwargs)`

Bases: `indico.core.db.sqlalchemy.descriptions.DescriptionMixin`, `indico.core.db.sqlalchemy.colors.ColorMixin`, `indico.core.db.sqlalchemy.locations.LocationMixin`, `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

background_color

can_access (*user*)

default_colors = `ColorTuple(text=u'202020', background=u'90c0f0')`

default_render_mode = 2

duration

end_dt

event

id

inherit_location

location_backref_name = `u'breaks'`

location_parent

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

own_address

own_room

own_room_id

own_room_name

own_venue

own_venue_id

own_venue_name

possible_render_modes = set([<RenderMode.markdown: 2>])

render_mode = 2

start_dt

text_color

title

class `indico.modules.events.timetable.models.entries.TimetableEntry` (**kwargs)
Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

break_

break_id

can_view (*user*)

Checks whether the user will see this entry in the timetable.

children

contribution

contribution_id

duration

end_dt

event

event_id

extend_end_dt (*end_dt*)

extend_parent (*by_start=True, by_end=True*)

Extend start/end of parent objects if needed.

No extension if performed for entries crossing a day boundary in the event timezone.

Parameters

- **by_start** – Extend parent by start datetime.
- **by_end** – Extend parent by end datetime.

extend_start_dt (*start_dt*)

id

is_parallel (*in_session=False*)

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The `other` locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

move (*start_dt*)

Move the entry to start at a different time.

This method automatically moves children of the entry to preserve their start time relative to the parent's start time.

move_next_to (*sibling, position=u'before'*)

object

parent_id

session_block

session_block_id

session_siblings

siblings

siblings_query

start_dt

type

class `indico.modules.events.timetable.models.entries.TimetableEntryType`

Bases: `indico.util.struct.enum.RichIntEnum`

BREAK = 3

CONTRIBUTION = 2

SESSION_BLOCK = 1

Operations

`indico.modules.events.timetable.operations.can_swap_entry` (*entry*, *direction*,
in_session=False)

`indico.modules.events.timetable.operations.create_break_entry` (*event*, *data*, *ses-*
sion_block=None)

`indico.modules.events.timetable.operations.create_session_block_entry` (*session_*,
data)

`indico.modules.events.timetable.operations.create_timetable_entry` (*event*,
data, *par-*
ent=None,
ex-
tend_parent=False)

`indico.modules.events.timetable.operations.delete_timetable_entry` (*entry*,
log=True)

`indico.modules.events.timetable.operations.fit_session_block_entry` (*entry*,
log=True)

`indico.modules.events.timetable.operations.get_sibling_entry` (*entry*, *direction*,
in_session=False)

`indico.modules.events.timetable.operations.move_timetable_entry` (*entry*, *par-*
ent=None,
day=None)

Move the *entry* to another session or top-level timetable

Parameters

- **entry** – *TimetableEntry* to be moved
- **parent** – If specified then the entry will be set as a child of parent
- **day** – If specified then the entry will be moved to the top-level timetable on this day

`indico.modules.events.timetable.operations.schedule_contribution` (*contribution*,
start_dt, *ses-*
sion_block=None,
ex-
tend_parent=False)

```
indico.modules.events.timetable.operations.swap_timetable_entry(entry, direction, session_=None)
```

Swap entry with closest gap or non-parallel sibling

```
indico.modules.events.timetable.operations.update_break_entry(break_, data)
```

```
indico.modules.events.timetable.operations.update_timetable_entry(entry, data)
```

```
indico.modules.events.timetable.operations.update_timetable_entry_object(entry, data)
```

Update the *object* of a timetable entry according to its type

Utilities

```
indico.modules.events.timetable.util.find_latest_entry_end_dt(obj, day=None)
```

Get the latest end datetime for timetable entries within the object.

Parameters

- **obj** – The `Event` or `SessionBlock` that will be used to look for timetable entries.
- **day** – The local event date to look for timetable entries. Applicable only to `Event`.

Returns The end datetime of the timetable entry finishing the latest. `None` if no entry was found.

```
indico.modules.events.timetable.util.find_next_start_dt(duration, obj, day=None, force=False)
```

Find the next most convenient start date fitting a duration within an object.

Parameters

- **duration** – Duration to fit into the event/session-block.
- **obj** – The `Event` or `SessionBlock` the duration needs to fit into.
- **day** – The local event date where to fit the duration in case the object is an event.
- **force** – Gives earliest datetime if the duration doesn't fit.

Returns The end datetime of the latest scheduled entry in the object if the duration fits then. If it doesn't, the latest datetime that fits it. `None` if the duration cannot fit in the object, earliest datetime if `force` is `True`.

```
indico.modules.events.timetable.util.get_category_timetable(categories, start_dt, end_dt, detail_level=u'event', tz=<UTC>, from_categ=None, grouped=True)
```

Retrieve time blocks that fall within a specific time interval for a given set of categories.

Parameters

- **categories** – iterable containing list of category IDs
- **start_dt** – start of search interval (`datetime`, expected to be in display timezone)
- **end_dt** – end of search interval (`datetime` in expected to be in display timezone)
- **detail_level** – the level of detail of information (`event|session|contribution`)
- **tz** – the timezone information should be displayed in

- **from_categ** – Category that will be taken into account to calculate visibility
- **grouped** – Whether to group results by start date

Returns a dictionary containing timetable information in a structured way. See source code for examples.

```
indico.modules.events.timetable.util.get_nested_entries(*args, **kwargs)
```

```
indico.modules.events.timetable.util.get_session_block_entries(event, day)
```

Returns a list of event top-level session blocks for the given *day*

```
indico.modules.events.timetable.util.get_time_changes_notifications(changes,
                                                                    tzinfo,
                                                                    en-
                                                                    try=None)
```

```
indico.modules.events.timetable.util.get_timetable_offline_pdf_generator(event)
```

```
indico.modules.events.timetable.util.get_top_level_entries(*args, **kwargs)
```

```
indico.modules.events.timetable.util.render_entry_info_balloon(entry,          ed-
                                                                itable=False,
                                                                sess=None)
```

```
indico.modules.events.timetable.util.render_session_timetable(session,
                                                                timetable_layout=None,
                                                                manage-
                                                                ment=False)
```

```
indico.modules.events.timetable.util.shift_following_entries(entry, shift, ses-
                                                                sion_=None)
```

Reschedules entries starting after the given entry by the given shift.

```
class indico.modules.events.timetable.reschedule.RescheduleMode
```

Bases: unicode, indico.util.struct.enum.RichEnum

duration = u'duration'

none = u'none'

time = u'time'

title

```
class indico.modules.events.timetable.reschedule.Rescheduler(event, mode, day,
                                                                session=None, ses-
                                                                sion_block=None,
                                                                fit_blocks=False,
                                                                gap=datetime.timedelta(0))
```

Bases: object

Compacts the the schedule of an event day by either adjusting start times or durations of timetable entries.

Parameters

- **event** – The event of which the timetable entries should be rescheduled.
- **mode** – A *RescheduleMode* value specifying whether the duration or start time should be adjusted.
- **day** – A *date* specifying the day to reschedule (the day of the timetable entries are determined using the event's timezone)
- **session** – If specified, only blocks of that session will be rescheduled, ignoring any other timetable entries. Cannot be combined with *session_block*.

- **session_block`** – If specified, only entries inside that block will be rescheduled. Cannot be combined with *session*.
- **fit_blocks** – Whether session blocks should be resized to exactly fit their contents before the actual rescheduling operation.
- **gap** – A timedelta specifying the cap between rescheduled timetable entries.

run()
Perform the rescheduling

Track

Todo

Docstring (module, models, operations)

Models

class `indico.modules.events.tracks.models.tracks.Track` (**kwargs)
Bases: `indico.core.db.sqlalchemy.descriptions.DescriptionMixin`,
`flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

abstract_reviewers

abstracts_accepted

abstracts_reviewed

abstracts_submitted

can_convene (*user*)

can_delete (*user*)

can_review_abstracts (*user*)

code

conveners

default_render_mode = 2

event

event_id

full_title

id

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

position

possible_render_modes = set([<RenderMode.markdown: 2>])

render_mode = 2

short_title

title

Operations

`indico.modules.events.tracks.operations.create_track(event, data)`

`indico.modules.events.tracks.operations.delete_track(track)`

`indico.modules.events.tracks.operations.update_program(event, data)`

`indico.modules.events.tracks.operations.update_track(track, data)`

Static site

Todo

Doctrings (module, utilities)

Models

class `indico.modules.events.static.models.static.StaticSite(**kwargs)`
Bases: `indico.core.storage.models.StoredFileMixin, flask_sqlalchemy.Model`

Static site for an Indico event.

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

add_file_date_column = False

content_type

The MIME type of the file

created_dt = None

creator

The user who created the static site

creator_id

ID of the user who created the static site

event

The Event this static site is associated with

event_id

ID of the event

file_required = False

filename

The name of the file

id

Entry ID

locator

md5

An MD5 hash of the file.

Automatically assigned when *save()* is called.

requested_dt

The date and time the static site was requested

size

The size of the file (in bytes).

Automatically assigned when *save()* is called.

state

The state of the static site (a *StaticSiteState* member)

storage_backend

storage_file_id

class `indico.modules.events.static.models.static.StaticSiteState`

Bases: `indico.util.struct.enum.RichIntEnum`

expired = 4

failed = 3

pending = 0

running = 1

success = 2

Utilities

`indico.modules.events.static.util.url_to_static_filename(endpoint, url)`

Category

Todo

Docstrings (module, model, operations, utilities)

Models

class `indico.modules.categories.models.categories.Category` (**kwargs)

Bases: `indico.core.db.sqlalchemy.searchable_titles.SearchableTitleMixin`,
`indico.core.db.sqlalchemy.descriptions.DescriptionMixin`, `indico.core.db.sqlalchemy.protection.ProtectionManagersMixin`,
`indico.core.db.sqlalchemy.attachments.AttachedItemsMixin`, `flask_sqlalchemy.Model`

An Indico category

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

ATTACHMENT_FOLDER_ID_COLUMN = u'category_id'

access_key = None

acl_entries

allow_no_access_contact = True

can_create_events (*user*)

Check whether the user can create events in the category.

chain_query

Get a query object for the category chain.

The query retrieves the root category first and then all the intermediate categories up to (and including) this category.

children

deep_children_query

Get a query object for all subcategories.

This includes subcategories at any level of nesting.

default_event_themes

default_render_mode = 2

default_ticket_template

default_ticket_template_id

disallowed_protection_modes = frozenset([])

display_tzinfo

The tzinfo of the category or the one specified by the user

effective_icon_url

Get the HTTP URL of the icon (possibly inherited).

`event_creation_notification_emails`

`event_creation_restricted`

`event_message`

`event_message_mode`

`classmethod get_icon_data_cte ()`

`classmethod get_protection_cte ()`

`get_protection_parent_cte ()`

`classmethod get_root ()`

Get the root category

`classmethod get_tree_cte (col='u'id')`

Create a CTE for the category tree.

The CTE contains the following columns:

- `id` – the category id
- **`path`** – an array containing the path from the root to the category itself
- `is_deleted` – whether the category is deleted

Parameters `col` – The name of the column to use in the path or a callable receiving the category alias that must return the expression used for the ‘path’ retrieved by the CTE.

`has_effective_icon`

`has_icon`

`has_logo`

`has_only_events`

`icon`

`icon_metadata`

`icon_url`

Get the HTTP URL of the icon.

`id`

`inheriting_have_acl = True`

`is_deleted`

`is_descendant_of (categ)`

`is_empty`

`is_root`

`locator`

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

logo

logo_metadata

logo_url

Get the HTTP URL of the logo.

move (*target*)

Move the category into another category.

notify_managers

nth_parent (*n_cats*, *fail_on_overflow=True*)

Return the nth parent of the category.

Parameters

- **n_cats** – the number of categories to go up
- **fail_on_overflow** – whether to fail if we try to go above the root category

Returns *Category* object or None (only if *fail_on_overflow* is not set)

own_no_access_contact

own_visibility_horizon

Get the highest category this one would like to be visible from (configured visibility).

parent_chain_query

Get a query object for the category's parent chain.

The query retrieves the root category first and then all the intermediate categories up to (excluding) this category.

parent_id

position

possible_render_modes = set([<RenderMode.markdown: 2>])

protection_mode

protection_parent

real_visibility_horizon

Get the highest category this one is actually visible from (as limited by categories above).

render_mode = 2

suggestions_disabled

timezone

title

tzinfo**url****visibility****visibility_horizon_query**

Get a query object that returns the highest category this one is visible from.

visible_categories_cte

Get a sqlalchemy select for the visible categories within this category, including the category itself.

visible_categories_query

Get a query object for the visible categories within this category, including the category itself.

class `indico.modules.categories.models.categories.EventMessageMode`Bases: `indico.util.struct.enum.RichIntEnum`**danger = 3****disabled = 0****info = 1****warning = 2****class** `indico.modules.categories.models.principals.CategoryPrincipal` (***kwargs*)Bases: `indico.core.db.sqlalchemy.principals.PrincipalRolesMixin`, `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

allow_networks = True**category_id**

The ID of the associated event

email = None**full_access****id**

The ID of the acl entry

ip_network_group**ip_network_group_id****local_group****local_group_id****multipass_group_name****multipass_group_provider****principal_backref_name = u'in_category_acls'****principal_for = u'Category'****read_access****roles**

```
type
unique_columns = (u'category_id',)
user
user_id
```

```
class indico.modules.categories.models.settings.CategorySetting (**kwargs)
    Bases: indico.core.settings.models.base.JSONSettingsBase, flask_sqlalchemy.
    Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
category
category_id
id
module
name
value
```

Operations

```
indico.modules.categories.operations.create_category (parent, data)
indico.modules.categories.operations.delete_category (category)
indico.modules.categories.operations.move_category (category, target_category)
indico.modules.categories.operations.update_category (category, data, skip=())
```

Utilities

```
indico.modules.categories.util.get_attachment_count (category_id=None)
```

Get the number of attachments in events in a category.

Parameters `category_id` – The category ID to get statistics for. Attachments from subcategories are also included.

Returns The number of attachments

```
indico.modules.categories.util.get_category_stats (*args, **kwargs)
```

Get category statistics.

This function is mainly a helper so we can get and cache all values at once and keep a last-update timestamp.

Parameters `category_id` – The category ID to get statistics for. Subcategories are also included.

```
indico.modules.categories.util.get_contribs_by_year (category_id=None)
```

Get the number of contributions for each year.

Parameters `category_id` – The category ID to get statistics for. Contributions from subcategories are also included.

Returns An *OrderedDict* mapping years to contribution counts.

`indico.modules.categories.util.get_events_by_year` (*category_id=None*)

Get the number of events for each year.

Parameters `category_id` – The category ID to get statistics for. Events from subcategories are also included.

Returns An *OrderedDict* mapping years to event counts.

`indico.modules.categories.util.get_image_data` (*image_type, category*)

`indico.modules.categories.util.get_upcoming_events` (**args, **kwargs*)

Get the global list of upcoming events

`indico.modules.categories.util.get_visibility_options` (*category_or_event, allow_invisible=True*)

Return the visibility options available for the category or event.

`indico.modules.categories.serialize.serialize_categories_ical` (*category_ids, user, event_filter=True, event_filter_fn=None, update_query=None*)

Export the events in a category to iCal

Parameters

- **category_ids** – Category IDs to export
- **user** – The user who needs to be able to access the events
- **event_filter** – A SQLAlchemy criterion to restrict which events will be returned. Usually something involving the start/end date of the event.
- **event_filter_fn** – A callable that determines which events to include (after querying)
- **update_query** – A callable that can update the query used to retrieve the events. Must return the updated query object.

`indico.modules.categories.serialize.serialize_category` (*category, with_favorite=False, with_path=False, parent_path=None, child_path=None*)

`indico.modules.categories.serialize.serialize_category_atom` (*category, url, user, event_filter*)

Export the events in a category to Atom

Parameters

- **category** – The category to export
- **url** – The URL of the feed
- **user** – The user who needs to be able to access the events
- **event_filter** – A SQLAlchemy criterion to restrict which events will be returned. Usually something involving the start/end date of the event.

`indico.modules.categories.serialize.serialize_category_chain` (*category, include_children=False, include_parents=False*)

Settings

`class` `indico.modules.categories.settings.CategorySettingsProxy` (*module*, *defaults=None*, *strict=True*, *acls=None*, *converters=None*)

Bases: `indico.core.settings.proxy.SettingsProxyBase`

Proxy class to access category-specific settings for a certain module.

delete (*category*, **args*, ***kwargs*)

Delete settings.

Parameters

- **category** – Category (or its ID)
- **names** – One or more names of settings to delete

delete_all (*category*, **args*, ***kwargs*)

Delete all settings.

Parameters **category** – Category (or its ID)

get (*category*, **args*, ***kwargs*)

Retrieve the value of a single setting.

Parameters

- **category** – Category (or its ID)
- **name** – Setting name
- **default** – Default value in case the setting does not exist

Returns The settings's value or the default value

get_all (*category*, **args*, ***kwargs*)

Retrieve all settings.

Parameters

- **category** – Category (or its ID)
- **no_defaults** – Only return existing settings and ignore defaults.

Returns Dict containing the settings

query

Return a query object filtering by the proxy's module.

set (*category*, **args*, ***kwargs*)

Set a single setting.

Parameters

- **category** – Category (or its ID)
- **name** – Setting name
- **value** – Setting value; must be JSON-serializable

set_multi (*category*, **args*, ***kwargs*)

Set multiple settings at once.

Parameters

- **category** – Category (or its ID)
- **items** – Dict containing the new settings

User

Todo

Docstrings (module, models, utilities)

Models

class `indico.modules.users.models.users.NameFormat`

Bases: `indico.util.struct.enum.RichIntEnum`

f_last = 3

f_last_upper = 7

first_last = 0

first_last_upper = 4

last_f = 2

last_f_upper = 6

last_first = 1

last_first_upper = 5

class `indico.modules.users.models.users.PersonMixin`

Bases: `object`

Add convenience properties and methods to person classes.

Assumes the following attributes exist: * `first_name` * `last_name` * `title`

display_full_name

Return the full name using the user's preferred name format.

full_name

Return the person's name in 'Firstname Lastname' notation.

get_full_name (*last_name_first=True, last_name_upper=True, abbrev_first_name=True, show_title=False, _show_empty_names=False*)

Return the person's name in the specified notation.

Note: Do not use positional arguments when calling this method. Always use keyword arguments!

Parameters

- **last_name_first** – if “lastname, firstname” instead of “firstname lastname” should be used
- **last_name_upper** – if the last name should be all-uppercase
- **abbrev_first_name** – if the first name should be abbreviated to use only the first character
- **show_title** – if the title of the person should be included

name
Return the person's name in 'Firstname Lastname' notation.

title
The title of the user

class `indico.modules.users.models.users.User` (**kwargs)
Bases: `indico.modules.users.models.users.PersonMixin`, `flask_sqlalchemy.Model`

Indico users

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

abstract_email_log_entries

abstracts

address
the address of the user

affiliation
the affiliation of the user

all_emails
all emails of the user. read-only; use it only for searching by email! also, do not use it between modifying *email* or *secondary_emails* and a session expire/commit!

api_key
the active API key of the user

as_avatar

as_legacy

as_principal
The serializable principal identifier of this user

avatar_css

can_be_modified (*user*)
If this user can be modified by the given user

created_events

email
the primary email address of the user

external_identities
The external identities of the user

favorite_categories
the users's favorite categories

favorite_users
the users's favorite users

first_name
the first name of the user

get_full_name (**args*, **kwargs)


```

static get_system_user ()
    id
        the unique id of the user
    identities
        the identities used by this user
    in_attachment_acls
    in_attachment_folder_acls
    is_admin
        if the user is an administrator with unrestricted access to everything
    is_blocked
        if the user has been blocked
    is_deleted
        if the user is deleted (e.g. due to a merge)
    is_group = False
    is_network = False
    is_pending
        if the user is pending (e.g. never logged in, only added to some list)
    is_single_person = True
    is_system
        if the user is the default system user
    iter_identifiers (check_providers=False, providers=None)
        Yields (provider, identifier) tuples for the user.

```

Parameters

- **check_providers** – If True, providers are searched for additional identifiers once all existing identifiers have been yielded.
- **providers** – May be a set containing provider names to get only identifiers from the specified providers.

```

judged_abstracts
last_name
    the last/family name of the user
local_identities
    The local identities of the user
local_identity
    The main (most recently used) local identity
locator
    Defines a smart locator property.

```

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The `other` locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

make_email_primary (*email*)

Promotes a secondary email address to the primary email address

Parameters **email** – an email address that is currently a secondary email

merged_into_id

the id of the user this user has been merged into

merged_into_user

the user this user has been merged into

modified_abstracts

old_api_keys

the previous API keys of the user

phone

the phone number of the user

principal_order = 0

principal_type = 1

secondary_emails

any additional emails the user might have

secondary_local_identities

The local identities of the user except the main one

settings

Returns the user settings proxy for this user

suggested_categories

the user's category suggestions

synced_fields

The fields of the user whose values are currently synced.

This set is always a subset of the synced fields define in synced fields of the idp in 'indico.conf'.

synced_values

The values from the synced identity for the user.

Those values are not the actual user's values and might differ if they are not set as synchronized.

synchronize_data (*refresh=False*)

Synchronize the fields of the user from the sync identity.

This will take only into account *synced_fields*.

Parameters **refresh** – bool – Whether to refresh the synced identity with the sync provider before instead of using the stored data. (Only if the sync provider supports refresh.)

```
class indico.modules.users.models.users.UserTitle
```

```
    Bases: indico.util.struct.enum.RichIntEnum
```

```
    dr = 4
```

```
    mr = 1
```

```
    mrs = 3
```

```
    ms = 2
```

```
    none = 0
```

```
    prof = 5
```

```
indico.modules.users.models.users.format_display_full_name(user, obj)
```

```
indico.modules.users.models.users.syncable_fields = {u'affiliation': lu'affiliation', u'first_name': lu'first name'}
```

Fields which can be synced as keys and a mapping to a more human readable version, used for flashing messages

```
class indico.modules.users.models.affiliations.UserAffiliation(**kwargs)
```

```
    Bases: flask_sqlalchemy.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
    id
```

the unique id of the affiliations

```
    name
```

the affiliation

```
    user_id
```

the id of the associated user

```
class indico.modules.users.models.emails.UserEmail(**kwargs)
```

```
    Bases: flask_sqlalchemy.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```
    email
```

the email address

```
    id
```

the unique id of the email address

```
    is_primary
```

if the email is the user's primary email

```
    is_user_deleted
```

if the user is marked as deleted (e.g. due to a merge). DO NOT use this flag when actually deleting an email

```
    user_id
```

the id of the associated user

class `indico.modules.users.models.suggestions.SuggestedCategory` (**kwargs)
Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

category

category_id

is_ignored

classmethod `merge_users` (*target*, *source*)

Merge the suggestions for two users.

Parameters

- **target** – The target user of the merge.
- **source** – The user that is being merged into *target*.

score

user_id

class `indico.modules.users.models.settings.UserSetting` (**kwargs)
Bases: `indico.core.settings.models.base.JSONSettingsBase`, `flask_sqlalchemy.Model`

User-specific settings

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

id

module

name

user

user_id

value

class `indico.modules.users.models.settings.UserSettingsProxy` (*module*, *de-*
faults=None,
strict=True,
acls=None, *con-*
verters=None)

Bases: `indico.core.settings.proxy.SettingsProxyBase`

Proxy class to access user-specific settings for a certain module

delete (*user*, **args*, **kwargs)

Deletes settings.

Parameters

- **user** – {'user': user} or {'user_id': id}
- **names** – One or more names of settings to delete

delete_all (*user*, *args, **kwargs)

Deletes all settings.

Parameters **user** – {'user': user} or {'user_id': id}

get (*user*, *args, **kwargs)

Retrieves the value of a single setting.

Parameters

- **user** – {'user': user} or {'user_id': id}
- **name** – Setting name
- **default** – Default value in case the setting does not exist

Returns The settings's value or the default value

get_all (*user*, *args, **kwargs)

Retrieves all settings

Parameters

- **user** – {'user': user} or {'user_id': id}
- **no_defaults** – Only return existing settings and ignore defaults.

Returns Dict containing the settings

query

Returns a query object filtering by the proxy's module.

set (*user*, *args, **kwargs)

Sets a single setting.

Parameters

- **user** – {'user': user} or {'user_id': id}
- **name** – Setting name
- **value** – Setting value; must be JSON-serializable

set_multi (*user*, *args, **kwargs)

Sets multiple settings at once.

Parameters

- **user** – {'user': user} or {'user_id': id}
- **items** – Dict containing the new settings

`indico.modules.users.models.settings.user_or_id(f)`

Operations

`indico.modules.users.operations.create_user` (*email*, *data*, *identity=None*, *settings=None*, *other_emails=None*, *from_moderation=True*)

Create a new user.

This may also convert a pending user to a proper user in case the email address matches such a user.

Parameters

- **email** – The primary email address of the user.
- **data** – The data used to populate the user.
- **identity** – An *Identity* to associate with the user.
- **settings** – A dict containing user settings.
- **other_emails** – A set of email addresses that are also used to check for a pending user. They will also be added as secondary emails to the user.
- **from_moderation** – Whether the user was created through the moderation process or manually by an admin.

Utilities

`indico.modules.users.util.get_admin_emails()`

Get the email addresses of all Indico admins

`indico.modules.users.util.get_color_for_username(username)`

`indico.modules.users.util.get_linked_events(user, dt, limit=None)`

Get the linked events and the user's roles in them

Parameters

- **user** – A *User*
- **dt** – Only include events taking place on/after that date
- **limit** – Max number of events

`indico.modules.users.util.get_related_categories(user, detailed=True)`

Gets the related categories of a user for the dashboard

`indico.modules.users.util.get_suggested_categories(user)`

Gets the suggested categories of a user for the dashboard

`indico.modules.users.util.get_user_by_email(email, create_pending=False)`

finds a user based on his email address.

Parameters

- **email** – The email address of the user.
- **create_pending** – If True, this function searches for external users and creates a new pending *User* in case no existing user was found.

Returns A *User* instance or *None* if not exactly one user was found.

`indico.modules.users.util.merge_users(source, target, force=False)`

Merge two users together, unifying all related data

Parameters

- **source** – source user (will be set as deleted)
- **target** – target user (final)

`indico.modules.users.util.search_users(exact=False, include_deleted=False, include_pending=False, external=False, allow_system_user=False, **criteria)`

Searches for users.

Parameters

- **exact** – Indicates if only exact matches should be returned. This is MUCH faster than a non-exact search, especially when searching external users.
- **include_deleted** – Indicates if also users marked as deleted should be returned.
- **include_pending** – Indicates if also users who are still pending should be returned.
- **external** – Indicates if identity providers should be searched for matching users.
- **allow_system_user** – Whether the system user may be returned in the search results.
- **criteria** – A dict containing any of the following keys: `first_name`, `last_name`, `email`, `affiliation`, `phone`, `address`

Returns A set of matching users. If *external* was set, it may contain both `IdentityInfo` objects for external users not yet in Indico and `User` objects for existing users.

```
indico.modules.users.util.serialize_user(user)
Serialize user to JSON-like object
```

```
class indico.modules.users.ext.ExtraUserPreferences(user)
Bases: object
```

Defines additional user preferences

To use this class, subclass it and override *defaults*, *fields* and *save* to implement your custom logic.

```
extend_defaults(defaults)
Adds values to the FormDefaults.
```

```
extend_form(form_class)
Creates a subclass of the form containing the extra field
```

```
fields = {}
a dict containing all the fields that should be added to the user preferences
```

```
load()
Returns a dict with the current values for the user
```

```
process_form_data(data)
Processes and saves submitted data.
```

This modifies *data* so the core code doesn't receive any extra data it doesn't expect.

```
save(data)
Saves the updated settings
```

Attachment

Todo

Docstrings (module, models, operations)

Models

```
class indico.modules.attachments.models.attachments.Attachment(**kwargs)
Bases: indico.core.db.sqlalchemy.protection.ProtectionMixin, indico.core.storage.models.VersionedResourceMixin, flask_sqlalchemy.Model
```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

absolute_download_url

The absolute download url for the attachment

access_key = None**acl**

The ACL of the folder (used for `ProtectionMode.protected`)

acl_entries**all_files****can_access** (*user, *args, **kwargs*)

Checks if the user is allowed to access the attachment.

This is the case if the user has access to see the attachment or if the user can manage attachments for the linked object.

description

The description of the attachment

download_url

The download url for the attachment

file**file_id****folder**

The folder containing the attachment

folder_id

The ID of the folder the attachment belongs to

get_download_url (*absolute=False*)

Returns the download url for the attachment.

During static site generation this returns a local URL for the file or the target URL for the link.

Parameters **absolute** – If the returned URL should be absolute.

id

The ID of the attachment

is_deleted

If the attachment has been deleted

link_url

The target URL for a link attachment

locator**modified_dt**

The date/time when the attachment was created/modified

own_no_access_contact = None**protection_mode****protection_parent**

stored_file_classalias of *AttachmentFile***stored_file_fkey** = u'attachment_id'**stored_file_table** = u'attachments.files'**title**

The name of the attachment

type

The type of the attachment (file or link)

user

The user who created the attachment

user_id

The ID of the user who created the attachment

class `indico.modules.attachments.models.attachments.AttachmentFile(**kwargs)`
 Bases: `indico.core.storage.models.StoredFileMixin`, `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

attachment_id

The ID of the associated attachment

content_type

The MIME type of the file

created_dt

The date/time when the file was uploaded

filename

The name of the file

id

The ID of the file

is_previewable**md5**

An MD5 hash of the file.

Automatically assigned when `save()` is called.**size**

The size of the file (in bytes).

Automatically assigned when `save()` is called.**storage_backend****storage_file_id****user**

The user who uploaded the file

user_id

The user who uploaded the file

version_of = u'attachment'

class `indico.modules.attachments.models.attachments.AttachmentType`

Bases: `indico.util.struct.enum.RichIntEnum`

file = 1

link = 2

class `indico.modules.attachments.models.folders.AttachmentFolder` (**kwargs)

Bases: `indico.core.db.sqlalchemy.links.LinkMixin`, `indico.core.db.sqlalchemy.protection.ProtectionMixin`, `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

access_key = None

acl

The ACL of the folder (used for `ProtectionMode.protected`)

acl_entries

attachments

The list of attachments that are not deleted, ordered by name

can_access (*user*, *args, **kwargs)

Checks if the user is allowed to access the folder.

This is the case if the user has access the folder or if the user can manage attachments for the linked object.

can_view (*user*)

Checks if the user can see the folder.

This does not mean the user can actually access its contents. It just determines if it is visible to him or not.

category

category_id

contribution

contribution_id

description

The description of the folder

event

event_id

events_backref_name = u'all_attachment_folders'

classmethod `get_for_linked_object` (*linked_object*, *preload_event=False*)

Gets the attachments for the given object.

This only returns attachments that haven't been deleted.

Parameters

- **linked_object** – A category, event, session, contribution or subcontribution.
- **preload_event** – If all attachments for the same event should be pre-loaded and cached in the app context. This must not be used when `linked_object` is a category.

classmethod `get_or_create` (*linked_object*, *title=None*)

Gets a folder for the given object or creates it.

If no folder title is specified, the default folder will be used. It is the caller's responsibility to add the folder or an object (such as an attachment) associated with it to the SQLAlchemy session using `db.session.add(...)`.

classmethod `get_or_create_default` (*linked_object*)

Gets the default folder for the given object or creates it.

id

The ID of the folder

is_always_visible

If the folder is always visible (even if you cannot access it)

is_default

If the folder is the default folder (used for "folder-less" files)

is_deleted

If the folder has been deleted

link_backref_lazy = `u'dynamic'`

link_backref_name = `u'attachment_folders'`

link_type

linked_event

linked_event_id

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The `other` locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

own_no_access_contact = `None`

protection_mode

protection_parent

session

session_id

subcontribution

subcontribution_id

title

The name of the folder (None for the default folder)

unique_links = u'is_default'

class `indico.modules.attachments.models.principals.AttachmentFolderPrincipal` (**kwargs)
Bases: `indico.core.db.sqlalchemy.principals.PrincipalMixin`, `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

email = None

folder_id

The ID of the associated folder

id

The ID of the acl entry

ip_network_group = None

ip_network_group_id = None

local_group

local_group_id

multipass_group_name

multipass_group_provider

principal_backref_name = u'in_attachment_folder_acls'

type

unique_columns = (u'folder_id',)

user

user_id

class `indico.modules.attachments.models.principals.AttachmentPrincipal` (**kwargs)
Bases: `indico.core.db.sqlalchemy.principals.PrincipalMixin`, `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

attachment_id

The ID of the associated attachment

email = None

id

The ID of the acl entry

ip_network_group = None

```

ip_network_group_id = None
local_group
local_group_id
multipass_group_name
multipass_group_provider
principal_backref_name = u'in_attachment_acls'
type
unique_columns = (u'attachment_id',)
user
user_id

```

Operations

`indico.modules.attachments.operations.add_attachment_link` (*data*, *linked_object*)
 Add a link attachment to *linked_object*

Utilities

`indico.modules.attachments.util.can_manage_attachments` (*obj*, *user*)
 Checks if a user can manage attachments for the object

`indico.modules.attachments.util.get_attached_folders` (*linked_object*, *include_empty=True*, *include_hidden=True*, *preload_event=False*) *in-*
in-

Return a list of all the folders linked to an object.

Parameters

- **linked_object** – The object whose attachments are to be returned
- **include_empty** – Whether to return empty folders as well.
- **include_hidden** – Include folders that the user can't see
- **preload_event** – in the process, preload all objects tied to the corresponding event and keep them in cache

`indico.modules.attachments.util.get_attached_items` (*linked_object*, *include_empty=True*, *include_hidden=True*, *preload_event=False*) *in-*
in-

Return a structured representation of all the attachments linked to an object.

Parameters

- **linked_object** – The object whose attachments are to be returned
- **include_empty** – Whether to return empty folders as well.
- **include_hidden** – Include folders that the user can't see
- **preload_event** – in the process, preload all objects tied to the corresponding event and keep them in cache

`indico.modules.attachments.util.get_default_folder_names()`

`indico.modules.attachments.util.get_event(linked_object)`

`indico.modules.attachments.util.get_nested_attached_items(obj)`

Returns a structured representation of all attachments linked to an object and all its nested objects.

Parameters `obj` – A Event, Session, Contribution or SubContribution object.

class `indico.modules.attachments.preview.ImagePreviewer`

Bases: `indico.modules.attachments.preview.Previewer`

ALLOWED_CONTENT_TYPE = `<_sre.SRE_Pattern object>`

TEMPLATE = `u'image_preview.html'`

class `indico.modules.attachments.preview.MarkdownPreviewer`

Bases: `indico.modules.attachments.preview.Previewer`

ALLOWED_CONTENT_TYPE = `<_sre.SRE_Pattern object>`

classmethod `generate_content(attachment)`

class `indico.modules.attachments.preview.PDFPreviewer`

Bases: `indico.modules.attachments.preview.Previewer`

ALLOWED_CONTENT_TYPE = `<_sre.SRE_Pattern object>`

TEMPLATE = `u'iframe_preview.html'`

classmethod `can_preview(attachment_file)`

class `indico.modules.attachments.preview.Previewer`

Bases: `object`

Base class for file previewers

To create a new file previewer, subclass this class and register it using the `get_file_previewers` signal.

ALLOWED_CONTENT_TYPE = `None`

TEMPATE = `None`

TEMPLATES_DIR = `u'attachments/previewers/'`

classmethod `can_preview(attachment_file)`

Checks if the content type of the file matches the allowed content type of files that the previewer can be used for.

classmethod `generate_content(attachment)`

Generates the HTML output of the file preview

class `indico.modules.attachments.preview.TextPreviewer`

Bases: `indico.modules.attachments.preview.Previewer`

ALLOWED_CONTENT_TYPE = `<_sre.SRE_Pattern object>`

classmethod `generate_content(attachment)`

`indico.modules.attachments.preview.get_file_previewer(attachment_file)`

Returns a file previewer for the given attachment file based on the file's content type.

`indico.modules.attachments.preview.get_file_previewers()`

Room booking

Todo

Docstrings (module, models, utilities, services)

Models

class `indico.modules.rb.models.rooms.Room` (**kwargs)

Bases: `indico.core.db.sqlalchemy.util.cache._CacheVersionMixin`,
`flask_sqlalchemy.Model`, `indico.util.serializer.Serializer`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

attributes

available_equipment

blocked_rooms

bookable_hours

booking_limit_days

booking_url

building

can_be_booked (*user*, *ignore_admin=False*)

Reservable rooms which does not require pre-booking can be booked by anyone. Other rooms - only by their responsables.

can_be_deleted (*user*)

can_be_modified (*user*)

Only admin can modify rooms.

can_be_overridden (*user*)

can_be_prebooked (*user*, *ignore_admin=False*)

Reservable rooms can be pre-booked by anyone. Other rooms - only by their responsables.

capacity

check_advance_days (*end_date*, *user=None*, *quiet=False*)

check_bookable_hours (*start_time*, *end_time*, *user=None*, *quiet=False*)

comments

details_url

division

static filter_available (*start_dt*, *end_dt*, *repetition*, *include_pre_bookings=True*, *include_pending_blockings=True*)

Returns a SQLAlchemy filter criterion ensuring that the room is available during the given time.

classmethod `find_all` (**args, **kwargs*)
Retrieves rooms, sorted by location and full name

find_available_vc_equipment ()

classmethod `find_with_attribute` (*attribute*)
Search rooms which have a specific attribute

static `find_with_filters` (*filters, user=None*)

floor

full_name

generate_name ()

get_attribute_by_name (*attribute_name*)

get_attribute_value (**args, **kwargs*)

get_blocked_rooms (**dates, **kwargs*)

classmethod `get_owned_by` (*user*)

static `get_with_data` (**args, **kwargs*)

has_attribute (*attribute_name*)

has_booking_groups

has_equipment (**args, **kwargs*)

has_live_reservations ()

has_photo

has_projector

has_special_name

has_vc

has_webcast_recording

id

is_active

is_auto_confirm

is_owned_by (**args, **kwargs*)
Checks if the user is managing the room (owner or manager)

is_public

is_reservable

key_location

kind

large_photo_url

latitude

location_id

location_name

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

longitude**manager_emails****map_url****marker_description****max_advance_days****max_capacity**

staticmethod(function) -> method

Convert a function to be a static method.

A static method does not receive an implicit first argument. To declare a static method, use this idiom:

```
class C: def f(arg1, arg2, ...): ... f = staticmethod(f)
```

It can be called either on the class (e.g. `C.f()`) or on an instance (e.g. `C().f()`). The instance is ignored except for its class.

Static methods in Python are similar to those found in Java or C++. For a more advanced concept, see the `classmethod` builtin.

name**nonbookable_periods****notification_before_days****notification_before_days_monthly****notification_before_days_weekly****notification_emails****notification_for_assistance****notifications_enabled****number**

owner

The owner of the room. If the room has the *manager-group* attribute set, any users in that group are also considered owners when it comes to management privileges. Use *is_owned_by()* for ownership checks that should also check against the management group.

owner_id**photo****photo_id****reservations****reservations_need_confirmation****set_attribute_value** (*name, value*)**site****small_photo_url****surface_area****telephone****update_name** ()**classmethod user_owns_rooms** (*user*)

class `indico.modules.rb.models.room_attributes.RoomAttribute` (**kwargs)

Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

children**id****is_hidden****is_required****location_id****name****parent_id****title****type**

class `indico.modules.rb.models.room_attributes.RoomAttributeAssociation` (**kwargs)

Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

attribute

attribute_id**room_id****value****class** `indico.modules.rb.models.room_bookable_hours.BookableHours` (**kwargs)Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

end_time**fits_period** (*st, et*)**room_id****start_time****class** `indico.modules.rb.models.room_nonbookable_periods.NonBookablePeriod` (**kwargs)Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

end_dt**overlaps** (*st, et*)**room_id****start_dt****class** `indico.modules.rb.models.aspects.Aspect` (**kwargs)Bases: `flask_sqlalchemy.Model, indico.util.serializer.Serializer`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

bottom_right_latitude**bottom_right_longitude****center_latitude****center_longitude****default_on_startup****id****location_id****name****top_left_latitude**

top_left_longitude

zoom_level

class `indico.modules.rb.models.blockings.Blocking` (**kwargs)

Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

allowed

A descriptor that presents a read/write view of an object attribute.

blocked_rooms

can_be_deleted (*user*)

can_be_modified (*user*)

The following persons are authorized to modify a blocking: - owner (the one who created the blocking) - admin (of course)

can_be_overridden (*user, room=None, explicit_only=False*)

Determines if a user can override the blocking

The following persons are authorized to override a blocking: - owner (the one who created the blocking) - any users on the blocking's ACL - unless `explicitOnly` is set: admins and room owners (if a room is given)

created_by_id

created_by_user

The user who created this blocking.

created_dt

end_date

id

is_active_at (*d*)

reason

start_date

class `indico.modules.rb.models.blocked_rooms.BlockedRoom` (**kwargs)

Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

State

alias of `BlockedRoomState`

approve (*notify_blocker=True*)

Approve the room blocking, rejecting all colliding reservations/occurrences.

blocking_id

classmethod `find_with_filters` (*filters*)

id
reject (*user=None, reason=None*)
 Reject the room blocking.
rejected_by
rejection_reason
room_id
state
state_name

class `indico.modules.rb.models.blocked_rooms.BlockedRoomState`
 Bases: `indico.util.struct.enum.RichIntEnum`

accepted = 1
pending = 0
rejected = 2

class `indico.modules.rb.models.blocking_principals.BlockingPrincipal` (***kwargs*)
 Bases: `indico.core.db.sqlalchemy.principals.PrincipalMixin`, `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

blocking_id
email = None
id
ip_network_group = None
ip_network_group_id = None
local_group
local_group_id
multipass_group_name
multipass_group_provider
principal_backref_name = u'in_blocking_acls'
type
unique_columns = (u'blocking_id',)
user
user_id

class `indico.modules.rb.models.equipment.EquipmentType` (***kwargs*)
 Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

children

id

location_id

name

parent_id

class `indico.modules.rb.models.holidays.Holiday` (**kwargs)
Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

date

id

location_id

name

class `indico.modules.rb.models.locations.Location` (**kwargs)
Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

aspects

attributes

default_aspect

default_aspect_id

default_location

`classmethod(function) -> method`

Convert a function to be a class method.

A class method receives the class as implicit first argument, just like an instance method receives the instance. To declare a class method, use this idiom:

```
class C: def f(cls, arg1, arg2, ...): ... f = classmethod(f)
```

It can be called either on the class (e.g. `C.f()`) or on an instance (e.g. `C().f()`). The instance is ignored except for its class. If a class method is called for a derived class, the derived class object is passed as the implied first argument.

Class methods are different than C++ or Java static methods. If you want those, see the `staticmethod` builtin.

equipment_types

`get_attribute_by_name (name)``get_buildings ()``get_equipment_by_name (name)``holidays``id``is_default``is_map_available``locator`

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```

@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}

```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

`map_url_template``name``rooms``set_default ()``working_time_end = datetime.time(17, 30)``working_time_periods = ((datetime.time(8, 30), datetime.time(12, 30)), (datetime.time(13, 30), datetime.time(17, 30)))``working_time_start = datetime.time(8, 30)``class indico.modules.rb.models.photos.Photo (**kwargs)``Bases: flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

`data``id``thumbnail`

exception `indico.modules.rb.models.reservations.ConflictingOccurrences`

Bases: `exceptions.Exception`

class `indico.modules.rb.models.reservations.RepeatFrequency`

Bases: `int`, `indico.util.struct.enum.IndicoEnum`

DAY = 1

MONTH = 3

NEVER = 0

WEEK = 2

class `indico.modules.rb.models.reservations.RepeatMapping`

Bases: `object`

classmethod `convert_legacy_repeatability (*args, **kw)`

classmethod `get_message (*args, **kw)`

classmethod `get_short_name (*args, **kw)`

mapping = {(<RepeatFrequency.NEVER: 0>, 0): ('Single reservation', None, 'none'), (<RepeatFrequency.MONTH: 3>, 1): ('Monthly', None, 'none')}

class `indico.modules.rb.models.reservations.Reservation (**kwargs)`

Bases: `indico.util.serializer.Serializer`, `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

accept (`*args`, `**kwargs`)

add_edit_log (`edit_log`)

booked_for_id

booked_for_name

booked_for_user

The user this booking was made for. Assigning a user here also updates `booked_for_name`.

booking_reason

can_be_accepted (`*args`, `**kwargs`)

can_be_cancelled (`*args`, `**kwargs`)

can_be_deleted (`*args`, `**kwargs`)

can_be_modified (`*args`, `**kwargs`)

can_be_rejected (`*args`, `**kwargs`)

cancel (`*args`, `**kwargs`)

contact_email

contact_phone

classmethod `create_from_data (room, data, user, prebook=None)`

Creates a new reservation.

Parameters

- **room** – The Room that’s being booked.
- **data** – A dict containing the booking data, usually from a `NewBookingConfirmForm` instance
- **user** – The `User` who creates the booking.
- **prebook** – Instead of determining the booking type from the user’s permissions, always use the given mode.

create_occurrences (*skip_conflicts, user=None*)

created_by_id

created_by_user

The user who created this booking.

created_dt

details_url

edit_logs

end_dt

event

The Event this reservation was made for

event_id

find_excluded_days ()

find_overlapping ()

static find_overlapping_with (*room, occurrences, skip_reservation_id=None*)

get_conflicting_occurrences ()

get_vc_equipment ()

static get_with_data (**args, **kwargs*)

id

is_accepted

is_archived

is_booked_for (*user*)

is_cancelled

is_owned_by (**args, **kwargs*)

is_pending

is_rejected

is_repeating

is_valid

location_name

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

modify (*data*, *user*)

Modifies an existing reservation.

Parameters

- **data** – A dict containing the booking data, usually from a `ModifyBookingForm` instance
- **user** – The *User* who modifies the booking.

needs_assistance

needs_vc_assistance

occurrences

reject (**args*, ***kwargs*)

rejection_reason

repeat_frequency

repeat_interval

repetition

room_id

start_dt

status_string

used_equipment

uses_vc

class `indico.modules.rb.models.reservation_edit_logs.ReservationEditLog` (***kwargs*)

Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from `kwargs`.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

id

info

reservation_id

```

timestamp
user_name
class indico.modules.rb.models.reservation_occurrences.ReservationOccurrence (**kwargs)
    Bases: flask_sqlalchemy.Model, indico.util.serializer.Serializer

    A simple constructor that allows initialization from kwargs.

    Sets attributes on the constructed instance using the names and values in kwargs.

    Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any
    mapped columns or relationships.

NO_RESERVATION_USER_STRATEGY = <sqlalchemy.orm.strategy_options.UnboundLoad object>
    A relationship loading strategy that will avoid loading the users linked to a reservation. You want to use
    this in pretty much all cases where you eager-load the reservation relationship.

cancel (*args, **kwargs)

classmethod create_series (start, end, repetition)

classmethod create_series_for_reservation (reservation)

date

end_dt

static filter_overlap (occurrences)

classmethod find_overlapping_with (room, occurrences, skip_reservation_id=None)

classmethod find_with_filters (filters, user=None)

get_overlap (occurrence, skip_self=False)

is_cancelled

is_rejected

is_valid

classmethod iter_create_occurrences (start, end, repetition)

static iter_start_time (start, end, repetition)

notification_sent

overlaps (occurrence, skip_self=False)

reject (*args, **kwargs)

rejection_reason

reservation_id

start_dt

indico.modules.rb.models.util.proxy_to_reservation_if_last_valid_occurrence (f)
    Forwards a method call to self.reservation if there is only one occurrence.

indico.modules.rb.models.util.unimplemented (exceptions=(<type 'exceptions.Exception'>,
), message='Unimplemented')

```

Utilities

`indico.modules.rb.util.get_default_booking_interval` (*duration=90, precision=15, force_today=False*)

Get the default booking interval for a room.

Returns the default booking interval for a room as a tuple containing the start and end times as *datetime* objects.

The start time is the default working start time or the current time (if the working start time is in the past); rounded up to the given precision in minutes (15 by default).

The end time corresponds to the start time plus the given duration in minutes. If the booking ends after the end of work time, it is automatically moved to the next day.

Parameters

- **duration** – int – The duration of a booking in minutes (must be greater than 1)
- **precision** – int – The number of minutes by which to round up the current time for the start time of a booking. Negative values are allowed but will round the time down and create a booking starting in the past.
- **force_today** – Forces a booking to be for today, even if it past the end of work time. This is ignored if the current time is either after 23:50 or within the amount of minutes of the precision from midnight. For example with a precision of 30 minutes, if the current time is 23:42 then the meeting will be the following day.

Returns (*datetime, datetime, bool*) – A tuple with the start and end times of the booking and a boolean which is *True* if the date was changed from today and *False* otherwise.

Raises *ValueError* if the duration is less than 1 minute

`indico.modules.rb.util.rb_check_user_access` (**args, **kwargs*)

Checks if the user has access to the room booking system

`indico.modules.rb.util.rb_is_admin` (**args, **kwargs*)

Checks if the user is a room booking admin

`indico.modules.rb.statistics.calculate_rooms_bookable_time` (*rooms, start_date=None, end_date=None*)

`indico.modules.rb.statistics.calculate_rooms_booked_time` (*rooms, start_date=None, end_date=None*)

`indico.modules.rb.statistics.calculate_rooms_occupancy` (*rooms, start=None, end=None*)

`indico.modules.rb.statistics.compose_rooms_stats` (*rooms*)

Services

class `indico.modules.rb.services.rooms.BookingPermission` (*params*)

Bases: `indico.legacy.services.implementation.base.LoggedOnlyService`

UNICODE_PARAMS = True

class `indico.modules.rb.services.rooms.RoomBookingAvailabilitySearchRooms` (*params*)

Bases: `indico.legacy.services.implementation.base.ServiceBase`

UNICODE_PARAMS = True

class `indico.modules.rb.services.rooms.RoomBookingListLocationsAndRoomsWithGuids` (*params*)

Bases: `indico.legacy.services.implementation.base.ServiceBase`

UNICODE_PARAMS = True

class `indico.modules.rb.services.aspects.RoomBookingMapBase` (*params*)
 Bases: `indico.legacy.services.implementation.base.ServiceBase`

UNICODE_PARAMS = True

class `indico.modules.rb.services.aspects.RoomBookingMapCreateAspect` (*params*)
 Bases: `indico.modules.rb.services.aspects.RoomBookingMapBase`

class `indico.modules.rb.services.aspects.RoomBookingMapListAspects` (*params*)
 Bases: `indico.modules.rb.services.aspects.RoomBookingMapBase`

class `indico.modules.rb.services.aspects.RoomBookingMapRemoveAspect` (*params*)
 Bases: `indico.modules.rb.services.aspects.RoomBookingMapBase`

class `indico.modules.rb.services.aspects.RoomBookingMapUpdateAspect` (*params*)
 Bases: `indico.modules.rb.services.aspects.RoomBookingMapBase`

class `indico.modules.rb.services.blockings.RoomBookingBlockingApprove` (*params*)
 Bases: `indico.modules.rb.services.blockings.RoomBookingBlockingProcessBase`

class `indico.modules.rb.services.blockings.RoomBookingBlockingProcessBase` (*params*)
 Bases: `indico.legacy.services.implementation.base.ServiceBase`

UNICODE_PARAMS = True

class `indico.modules.rb.services.blockings.RoomBookingBlockingReject` (*params*)
 Bases: `indico.modules.rb.services.blockings.RoomBookingBlockingProcessBase`

Authentication

Todo

Docstrings (module, models, utilities)

Models

class `indico.modules.auth.models.identities.Identity` (***kwargs*)
 Bases: `flask_sqlalchemy.Model`

Identities of Indico users

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in *kwargs*.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

data

id

the unique id of the identity

identifier

the unique identifier of the user within its provider

last_login_dt

the timestamp of the latest login

last_login_ip
the ip address that was used for the latest login

locator

multipass_data
internal data used by the flask-multipass system

password
the password of the user in case of a local identity

password_hash
the hash of the password in case of a local identity

provider
the provider name of the identity

register_login (*ip*)
Updates the last login information

safe_last_login_dt
last_login_dt that is safe for sorting (no None values)

user_id
the id of the user this identity belongs to

class `indico.modules.auth.models.registration_requests.RegistrationRequest` (**kwargs)
Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

comment

email

extra_emails

id

identity_data

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

settings

user_data

Utilities

`indico.modules.auth.util.impersonate_user (user)`

Impersonate another user as an admin

`indico.modules.auth.util.load_identity_info ()`

Retrieves identity information from the session

`indico.modules.auth.util.redirect_to_login (next_url=None, reason=None)`

Redirects to the login page.

Parameters

- **next_url** – URL to be redirected upon successful login. If not specified, it will be set to `request.relative_url`.
- **reason** – Why the user is redirected to a login page.

`indico.modules.auth.util.register_user (email, extra_emails, user_data, identity_data, settings, from_moderation=False)`

Create a user based on the registration data provided during the user registration process (via *RHRegister* and *RegistrationHandler*).

This method is not meant to be used for generic user creation, the only reason why this is here is that approving a registration request is handled by the *users* module.

`indico.modules.auth.util.save_identity_info (identity_info, user)`

Saves information from IdentityInfo in the session

`indico.modules.auth.util.undo_impersonate_user ()`

Undo an admin impersonation login and revert to the old user

`indico.modules.auth.util.url_for_login (next_url=None)`

`indico.modules.auth.util.url_for_logout (next_url=None)`

`indico.modules.auth.util.url_for_register (next_url=None, email=None)`

Returns the URL to register

Parameters

- **next_url** – The URL to redirect to afterwards.
- **email** – A pre-validated email address to use when creating a new local account. Use this argument **ONLY** when sending the link in an email or if the email address has already been validated using some other way.

OAuth

Todo

Docstrings (module, models, provider)

Models

class `indico.modules.oauth.models.applications.OAuthApplication` (**kwargs)

Bases: `flask_sqlalchemy.Model`

OAuth applications registered in Indico

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

client_id

the OAuth client_id

client_secret

the OAuth client_secret

client_type

default_redirect_uri

default_scopes

the OAuth default scopes the application may request access to

description

human readable description

id

the unique id of the application

is_enabled

whether the application is enabled or disabled

is_trusted

whether the application can access user data without asking for permission

locator

name

human readable name

redirect_uris

the OAuth absolute URIs that a application may use to redirect to after authorization

reset_client_secret ()

system_app_type

the type of system app (if any). system apps cannot be deleted

validate_redirect_uri (*redirect_uri*)

Called by flask-oauthlib to validate the `redirect_uri`.

Uses a logic similar to the one at GitHub, i.e. protocol and host/port must match exactly and if there is a path in the whitelisted URL, the path of the `redirect_uri` must start with that path.

class `indico.modules.oauth.models.applications.SystemAppType`

Bases: `int`, `indico.util.struct.enum.IndicoEnum`

checkin = 1

default_data

enforced_data

flower = 2**none = 0**

class `indico.modules.oauth.models.tokens.OAuthGrant` (*client_id, code, redirect_uri, user, scopes, expires*)

Bases: `object`

OAuth grant token

delete ()**classmethod** `get` (*client_id, code*)**key****classmethod** `make_key` (*client_id, code*)**save** ()**ttl**

class `indico.modules.oauth.models.tokens.OAuthToken` (***kwargs*)

Bases: `flask_sqlalchemy.Model`

OAuth tokens

A simple constructor that allows initialization from `kwargs`.Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

access_token

an unguessable unique string of characters

application

application authorized by this token

application_id

the identifier of the linked application

expires**id**

the unique identifier of the token

last_used_dt

the last time the token was used by the application

locator**scopes**

The set of scopes the linked application has access to.

type**user**

the user who owns this token

user_id

the identifier of the linked user

Utilities

exception `indico.modules.oauth.provider.DisabledClientIdError` (*description=None*,
uri=None,
state=None, *status_code=None*,
request=None)

Bases: `oauthlib.oauth2.rfc6749.errors.FatalClientError`

description: A human-readable ASCII [USASCII] text providing additional information, used to assist the client developer in understanding the error that occurred. Values for the “error_description” parameter MUST NOT include characters outside the set x20-21 / x23-5B / x5D-7E.

uri: A URI identifying a human-readable web page with information about the error, used to provide the client developer with additional information about the error. Values for the “error_uri” parameter MUST conform to the URI- Reference syntax, and thus MUST NOT include characters outside the set x21 / x23-5B / x5D-7E.

state: A CSRF protection value received from the client.

request: Oauthlib Request object

error = u'application_disabled_by_admin'

`indico.modules.oauth.provider.load_client` (*client_id*)

`indico.modules.oauth.provider.load_grant` (*client_id*, *code*)

`indico.modules.oauth.provider.load_token` (*access_token*, *refresh_token=None*)

`indico.modules.oauth.provider.save_grant` (*client_id*, *code*, *request*, **args*, ***kwargs*)

`indico.modules.oauth.provider.save_token` (*token_data*, *request*, **args*, ***kwargs*)

Group

Todo

Docstrings (module)

Models

class `indico.modules.groups.models.groups.LocalGroup` (***kwargs*)

Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance’s class are allowed. These could be, for example, any mapped columns or relationships.

id

the unique id of the group

members

the local groups this user belongs to

name
the name of the group

proxy
Returns a GroupProxy wrapping this group

class `indico.modules.groups.core.GroupProxy`

Bases: `object`

Provides a generic interface for both local and multipass groups.

Creating an instance of this class actually creates either a `LocalGroupProxy` or a `MultipassGroupProxy`, but they expose the same API.

Parameters

- **name_or_id** – The name of a multipass group or ID of a local group
- **provider** – The provider of a multipass group

Creates the correct GroupProxy for the group type

as_legacy

as_legacy_group

The legacy-style group wrapper

as_principal

The serializable principal identifier of this group

get_members ()

Gets the list of users who are members of the group

classmethod **get_named_default_group** (*name*)

Gets the group with the matching name from the default group provider.

If there is no default group provider, local groups will be used and *name* is the group's ID.

This method should only be used for legacy code or code that gets the group name from an external source which does not contain a provider identifier.

group

The underlying group object

has_member (*user*)

Checks if the user is a member of the group.

This can also be accessed using the `in` operator.

is_group = True

is_network = False

is_single_person = False

principal_order = 2

classmethod **search** (*name*, *exact=False*, *providers=None*)

Searches for groups

Parameters

- **name** – The group name to search for.
- **exact** – If only exact matches should be found (much faster)

- **providers** – None to search in all providers and local groups. May be a set specifying providers to search in. For local groups, the 'indico' provider name may be used.

Utilities

`indico.modules.groups.util.serialize_group(group)`
Serialize group to JSON-like object

Video conference

Todo

Docstrings (module, models, utilities, plugins, exceptions)

Models

class `indico.modules.vc.models.vc_rooms.VCRoom(**kwargs)`

Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

created_by_id

ID of the creator

created_by_user

The user who created the videoconference room

created_dt

Creation timestamp of the videoconference room

data

videoconference plugin-specific data

id

Videoconference room ID

locator

modified_dt

Modification timestamp of the videoconference room

name

Name of the videoconference room

plugin

status

Status of the videoconference room

type

Type of the videoconference room

class `indico.modules.vc.models.vc_rooms.VCRoomEventAssociation` (**kwargs)
 Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

contribution_id

data

videoconference plugin-specific data

delete (*user*, *delete_all=False*)

Deletes a VC room from an event

If the room is not used anywhere else, the room itself is also deleted.

Parameters

- **user** – the user performing the deletion
- **delete_all** – if True, the room is detached from all events and deleted.

event

The associated Event

event_id

ID of the event

classmethod find_for_event (**args*, **kwargs)

Returns a Query that retrieves the videoconference rooms for an event

Parameters

- **event** – an indico Event
- **only_linked_to_event** – only retrieve the vc rooms linked to the whole event
- **kwargs** – extra kwargs to pass to `find()`

classmethod get_linked_for_event (**args*, **kwargs)

Get a dict mapping link objects to event vc rooms

id

Association ID

link_object

link_type

Type of the object the vc_room is linked to

linked_block

The linked session block (if the VC room is attached to a block)

linked_contrib

The linked contribution (if the VC room is attached to a contribution)

linked_event

The linked event (if the VC room is attached to the event itself)

linked_event_id

locator

```
classmethod register_link_events ()
```

```
    session_block_id
```

```
    show
```

```
        If the vc room should be shown on the event page
```

```
    vc_room
```

```
        The associated :class:VCRoom
```

```
    vc_room_id
```

```
        ID of the videoconference room
```

```
class indico.modules.vc.models.vc_rooms.VCRoomLinkType
```

```
    Bases: int, indico.util.struct.enum.IndicoEnum
```

```
    block = 3
```

```
    contribution = 2
```

```
    event = 1
```

```
class indico.modules.vc.models.vc_rooms.VCRoomStatus
```

```
    Bases: int, indico.util.struct.enum.IndicoEnum
```

```
    created = 1
```

```
    deleted = 2
```

Utilities

```
indico.modules.vc.util.find_event_vc_rooms (from_dt=None, to_dt=None, distinct=False)
```

```
    Finds VC rooms matching certain criteria
```

Parameters

- **from_dt** – earliest event/contribution to include
- **to_dt** – latest event/contribution to include
- **distinct** – if True, never return the same (event, vcreom) more than once (even if it's linked more than once to that event)

```
indico.modules.vc.util.get_linked_to_description (obj)
```

```
indico.modules.vc.util.get_managed_vc_plugins (user)
```

```
    Returns the plugins the user can manage
```

```
indico.modules.vc.util.get_vc_plugins ()
```

```
    Returns a dict containing the available videoconference plugins.
```

```
indico.modules.vc.util.resolve_title (obj)
```

Plugins

```
class indico.modules.vc.plugins.VCPluginMixin
```

```
    Bases: object
```

```
    acl_settings = set([u'managers', u'acl'])
```

```
    can_manage_vc (user)
```

```
        Checks if a user has management rights on this VC system
```

can_manage_vc_room (*user, room*)

Checks if a user can manage a vc room

can_manage_vc_rooms (*user, event*)

Checks if a user can manage vc rooms on an event

category = u'Videoconference'

create_form (*event, existing_vc_room=None, existing_event_vc_room=None*)

Creates the videoconference room form

Parameters

- **event** – the event the videoconference room is for
- **existing_vc_room** – a vc_room from which to retrieve data for the form
- ****kwargs** – extra data to pass to the form if an existing vc room is passed

Returns an instance of an IndicoForm subclass

create_room (*vc_room, event*)

default_settings = {u'notification_emails': []}

friendly_name = None

the readable name of the VC plugin

get_notification_bcc_list (*action, vc_room, event*)

get_notification_cc_list (*action, vc_room, event*)

get_vc_room_attach_form_defaults (*event*)

get_vc_room_form_defaults (*event*)

icon_url

init ()

logo_url

render_buttons (*vc_room, event_vc_room, **kwargs*)

Renders a list of plugin specific buttons (eg: Join URL, etc) in the management area

Parameters

- **vc_room** – the VC room object
- **event_vc_room** – the association of an event and a VC room
- **kwargs** – arguments passed to the template

render_event_buttons (*vc_room, event_vc_room, **kwargs*)

Renders a list of plugin specific buttons (eg: Join URL, etc) in the event page

Parameters

- **vc_room** – the VC room object
- **event_vc_room** – the association of an event and a VC room
- **kwargs** – arguments passed to the template

render_form (***kwargs*)

Renders the videoconference room form ;param kwargs: arguments passed to the template

render_info_box (*vc_room, event_vc_room, event, **kwargs*)

Renders the information shown in the expandable box of a VC room row :param vc_room: the VC room object :param event_vc_room: the association of an event and a VC room :param event: the event with the current VC room attached to it :param kwargs: arguments passed to the template

render_manage_event_info_box (*vc_room, event_vc_room, event, **kwargs*)

Renders the information shown in the expandable box on a VC room in the management area

Parameters

- **vc_room** – the VC room object
- **event_vc_room** – the association of an event and a VC room
- **event** – the event with the current VC room attached to it
- **kwargs** – arguments passed to the template

service_name

settings_form

alias of VCPluginSettingsFormBase

update_data_association (*event, vc_room, event_vc_room, data*)

update_data_vc_room (*vc_room, data*)

vc_room_attach_form = None

the IndicoForm to use for the videoconference room attach form

vc_room_form = None

the IndicoForm to use for the videoconference room form

Exceptions

exception `indico.modules.vc.exceptions.VCRoomError` (*message, field=None*)

Bases: `exceptions.Exception`

exception `indico.modules.vc.exceptions.VCRoomNotFoundError` (*message*)

Bases: `indico.modules.vc.exceptions.VCRoomError`

Designer

Todo

Docstrings (module, models, utilities)

Models

class `indico.modules.designer.models.images.DesignerImageFile` (***kwargs*)

Bases: `indico.core.storage.models.StoredFileMixin, flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in kwargs.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

content_type
The MIME type of the file

created_dt
The date/time when the file was uploaded

download_url

filename
The name of the file

id
The ID of the file

locator

md5
An MD5 hash of the file.
Automatically assigned when *save()* is called.

size
The size of the file (in bytes).
Automatically assigned when *save()* is called.

storage_backend

storage_file_id

template

template_id
The designer template the image belongs to

version_of = None

```
class indico.modules.designer.models.templates.DesignerTemplate (**kwargs)
    Bases: flask_sqlalchemy.Model
```

background_image

background_image_id

backside_template

backside_template_id

category

category_id

data

event

event_id

id

is_clonable

is_system_template

locator
Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:

```
@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}
```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

owner

title

type

Utilities

`indico.modules.designer.util.get_all_templates(obj)`

Get all templates usable by an event/category

`indico.modules.designer.util.get_default_template_on_category(category, only_inherited=False)`

`indico.modules.designer.util.get_inherited_templates(obj)`

Get all templates inherited by a given event/category

`indico.modules.designer.util.get_not_deletable_templates(obj)`

Get all non-deletable templates for an event/category

`indico.modules.designer.util.get_placeholder_options()`

class `indico.modules.designer.pdf.DesignerPDFBase(template, config)`

Bases: object

get_pdf()

class `indico.modules.designer.pdf.TplData(width, height, items, background_position, width_cm, height_cm)`

Bases: tuple

Create new instance of TplData(width, height, items, background_position, width_cm, height_cm)

background_position

Alias for field number 3

height

Alias for field number 1

height_cm

Alias for field number 5

items

Alias for field number 2

width
Alias for field number 0

width_cm
Alias for field number 4

Placeholders

```

class indico.modules.designer.placeholders.EventDatesPlaceholder
    Bases: indico.util.placeholders.Placeholder

    description = lu'Event Dates'
    group = u'event'
    name = u'event_dates'
    classmethod render (event)

class indico.modules.designer.placeholders.EventDescriptionPlaceholder
    Bases: indico.util.placeholders.Placeholder

    description = lu'Event Description'
    group = u'event'
    name = u'event_description'
    classmethod render (event)

class indico.modules.designer.placeholders.RegistrationFullNamePlaceholder
    Bases: indico.modules.designer.placeholders.FullNamePlaceholderBase

    description = lu'Full Name'
    name = u'full_name'
    name_options = {}
    with_title = True

class indico.modules.designer.placeholders.EventOrgTextPlaceholder
    Bases: indico.util.placeholders.Placeholder

    description = lu'Event Organizers'
    group = u'event'
    name = u'event_organizers'
    classmethod render (event)

class indico.modules.designer.placeholders.RegistrationFullNameNoTitlePlaceholder
    Bases: indico.modules.designer.placeholders.FullNamePlaceholderBase

    description = lu'Full Name (no title)'
    name = u'full_name_no_title'
    name_options = {}
    with_title = False

class indico.modules.designer.placeholders.RegistrationFullNamePlaceholderB
    Bases: indico.modules.designer.placeholders.FullNamePlaceholderBase

    description = lu'Full Name B'

```

```
name = u'full_name_b'
name_options = {u'last_name_first': False}
with_title = True
class indico.modules.designer.placeholders.RegistrationFullNameNoTitlePlaceholderB
Bases: indico.modules.designer.placeholders.FullNamePlaceholderBase
description = lu'Full Name B (no title)'
name = u'full_name_b_no_title'
name_options = {u'last_name_first': False}
with_title = False
class indico.modules.designer.placeholders.RegistrationFullNamePlaceholderC
Bases: indico.modules.designer.placeholders.FullNamePlaceholderBase
description = lu'Full Name C'
name = u'full_name_c'
name_options = {u'last_name_upper': True, u'last_name_first': False}
with_title = True
class indico.modules.designer.placeholders.RegistrationFullNameNoTitlePlaceholderC
Bases: indico.modules.designer.placeholders.FullNamePlaceholderBase
description = lu'Full Name C (no title)'
name = u'full_name_no_title_c'
name_options = {u'last_name_upper': True}
with_title = False
class indico.modules.designer.placeholders.RegistrationFullNamePlaceholderD
Bases: indico.modules.designer.placeholders.FullNamePlaceholderBase
description = lu'Full Name D (abbrev.)'
name = u'full_name_d'
name_options = {u'last_name_upper': True, u'last_name_first': False, u'abbrev_first_name': True}
with_title = True
class indico.modules.designer.placeholders.RegistrationFullNameNoTitlePlaceholderD
Bases: indico.modules.designer.placeholders.FullNamePlaceholderBase
description = lu'Full Name D (abbrev., no title)'
name = u'full_name_no_title_d'
name_options = {u'last_name_upper': True, u'abbrev_first_name': True}
with_title = False
class indico.modules.designer.placeholders.RegistrationTitlePlaceholder
Bases: indico.modules.designer.placeholders.RegistrationPDPlaceholder
description = lu'Title'
field = u'title'
name = u'title'
```

```
class indico.modules.designer.placeholders.RegistrationFirstNamePlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPlaceholder

    description = lu'First Name'

    field = u'first_name'

    name = u'first_name'

class indico.modules.designer.placeholders.RegistrationLastNamePlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPlaceholder

    description = lu'Last Name'

    field = u'last_name'

    name = u'last_name'

class indico.modules.designer.placeholders.RegistrationTicketQRPlaceholder
    Bases: indico.util.placeholders.Placeholder

    description = lu'Ticket QR Code'

    group = u'registrant'

    name = u'ticket_qr_code'

    classmethod render (registration)

class indico.modules.designer.placeholders.RegistrationEmailPlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPlaceholder

    description = lu'E-mail'

    field = u'email'

    name = u'email'

class indico.modules.designer.placeholders.RegistrationAmountPlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPlaceholder

    description = lu'Price (no currency)'

    name = u'amount'

    classmethod render (registration)

class indico.modules.designer.placeholders.RegistrationPricePlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPlaceholder

    description = lu'Price (with currency)'

    name = u'price'

    classmethod render (registration)

class indico.modules.designer.placeholders.RegistrationAffiliationPlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPDPlaceholder

    description = lu'Institution'

    field = u'affiliation'

    name = u'affiliation'

class indico.modules.designer.placeholders.RegistrationPositionPlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPDPlaceholder

    description = lu'Position'
```

```
    field = u'position'
    name = u'position'
class indico.modules.designer.placeholders.RegistrationAddressPlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPDPlaceholder
    description = lu'Address'
    field = u'address'
    name = u'address'
class indico.modules.designer.placeholders.RegistrationCountryPlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPDPlaceholder
    description = lu'Country'
    field = u'country'
    name = u'country'
class indico.modules.designer.placeholders.RegistrationPhonePlaceholder
    Bases: indico.modules.designer.placeholders.RegistrationPDPlaceholder
    description = lu'Phone'
    field = u'phone'
    name = u'phone'
class indico.modules.designer.placeholders.EventTitlePlaceholder
    Bases: indico.util.placeholders.Placeholder
    description = lu'Event Title'
    group = u'event'
    name = u'event_title'
    classmethod render (event)
class indico.modules.designer.placeholders.CategoryTitlePlaceholder
    Bases: indico.util.placeholders.Placeholder
    description = lu'Category Title'
    group = u'event'
    name = u'category_title'
    classmethod render (event)
class indico.modules.designer.placeholders.EventRoomPlaceholder
    Bases: indico.util.placeholders.Placeholder
    description = lu'Event Room'
    group = u'event'
    name = u'event_room'
    classmethod render (event)
class indico.modules.designer.placeholders.EventVenuePlaceholder
    Bases: indico.util.placeholders.Placeholder
    description = lu'Event Venue'
```

```

group = u'event'
name = u'event_venue'
classmethod render (event)

```

```

class indico.modules.designer.placeholders.EventSpeakersPlaceholder
    Bases: indico.util.placeholders.Placeholder

    description = lu'Event Speakers/Chairs'

    group = u'event'
    name = u'event_speakers'
    classmethod render (event)

```

Network

Todo

Docstrings (module, models)

Models

```

class indico.modules.networks.models.networks.IPNetwork (**kwargs)
    Bases: flask_sqlalchemy.Model

```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```

group_id
network

```

```

class indico.modules.networks.models.networks.IPNetworkGroup (**kwargs)
    Bases: flask_sqlalchemy.Model

```

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

```

attachment_access_override
    Grants all IPs in the network group read access to all attachments

```

```

contains_ip (ip)

```

```

description

```

```

hidden

```

Whether the network group is hidden in ACL forms

```

id

```

```

is_group = False

```

is_network = True
is_single_person = False
locator
name
networks
A descriptor that presents a read/write view of an object attribute.
principal_order = 1
principal_type = 5

Utilities

`indico.modules.networks.util.serialize_ip_network_group(group)`
Serialize group to JSON-like object

News

Todo

Docstrings (module, models)

Models

class `indico.modules.news.models.news.NewsItem(**kwargs)`
Bases: `flask_sqlalchemy.Model`

A simple constructor that allows initialization from kwargs.

Sets attributes on the constructed instance using the names and values in `kwargs`.

Only keys that are present as attributes of the instance's class are allowed. These could be, for example, any mapped columns or relationships.

anchor

content

created_dt

id

locator

Defines a smart locator property.

This behaves pretty much like a normal read-only property and the decorated function should return a dict containing the necessary data to build a URL for the object.

This decorator should usually be applied to a method named `locator` as this name is required for `get_locator` to find it automatically when just passing the object.

If you need more than one locator, you can define it like this:


```

@locator_property
def locator(self):
    return {...}

@locator.other
def locator(self):
    return {...}

```

The other locator can then be accessed by passing `obj.locator.other` to the code expecting an object with a locator.

title

Utilities

`indico.modules.news.util.get_recent_news(*args, **kwargs)`
 Get a list of recent news for the home page

Indico fields

Todo

Docstrings to all fields

Indico fields extend from WTForm fields and are used for the special cases where the simple form fields are not enough to cover all needs.

class `indico.modules.events.fields.EventPersonLinkListField(*args, **kwargs)`
 Bases: `indico.modules.events.fields.PersonLinkListFieldBase`

A field to manage event's chairpersons

linked_object_attr = u'event'

person_link_cls

alias of `EventPersonLink`

pre_validate (*form*)

widget = <`indico.web.forms.widgets.JinjaWidget` object>

class `indico.modules.events.fields.EventPersonListField(*args, **kwargs)`
 Bases: `indico.web.forms.fields.principals.PrincipalListField`

A field that lets you select a list Indico user and EventPersons

Requires its form to have an event set.

create_untrusted_persons = False

Whether new event persons created by the field should be marked as untrusted

event

pre_validate (*form*)

process_formdata (*valuelist*)

class `indico.modules.events.fields.IndicoThemeSelectField(*args, **kwargs)`
 Bases: `wtforms.fields.core.SelectField`

```
class indico.modules.events.fields.PersonLinkListFieldBase(*args, **kwargs)
    Bases: indico.modules.events.fields.EventPersonListField

    default_sort_alpha = True
        If set to True, will be sorted alphabetically by default

    linked_object_attr = None
        name of the attribute on the form containing the linked object

    person_link_cls = None
        class that inherits from PersonLinkBase

    widget = None

class indico.modules.events.fields.ReferencesField(*args, **kwargs)
    Bases: indico.web.forms.fields.itemlists.MultipleItemsField

    A field to manage external references.

    pre_validate(form)

    process_formdata(valuelist)

class indico.modules.events.fields.ReviewQuestionsField(*args, **kwargs)
    Bases: indico.web.forms.fields.itemlists.MultipleItemsField

    process_formdata(valuelist)

class indico.modules.events.abstracts.fields.AbstractField(*args, **kwargs)
    Bases: wtforms.ext.sqlalchemy.fields.QuerySelectField

    A selectize-based field to select an abstract from an event.

    event

    pre_validate(form)

    search_payload

    search_url

    widget = <indico.web.forms.widgets.SelectizeWidget object>

class indico.modules.events.abstracts.fields.AbstractPersonLinkListField(*args,
                                                                 **kwargs)
    Bases: indico.modules.events.fields.PersonLinkListFieldBase

    A field to configure a list of abstract persons

    create_untrusted_persons = True

    default_sort_alpha = False

    linked_object_attr = u'abstract'

    person_link_cls
        alias of AbstractPersonLink

    pre_validate(form)

    widget = <indico.web.forms.widgets.JinjaWidget object>
```

```
class indico.modules.events.abstracts.fields.EmailRuleListField (label=None, val-
                                                                idators=None,
                                                                filters=(), de-
                                                                scription=u'',
                                                                id=None, de-
                                                                fault=None,
                                                                widget=None,
                                                                render_kw=None,
                                                                _form=None,
                                                                _name=None,
                                                                _prefix=u'',
                                                                _transla-
                                                                tions=None,
                                                                _meta=None)
```

Bases: `indico.web.forms.fields.simple.JSONField`

A field that stores a list of e-mail template rules.

Construct a new field.

Parameters

- **label** – The label of the field.
- **validators** – A sequence of validators to call when *validate* is called.
- **filters** – A sequence of filters which are run on input data by *process*.
- **description** – A description for the field, typically used for help text.
- **id** – An id to use for the field. A reasonable default is set by the form, and you shouldn't need to set this manually.
- **default** – The default value to assign to the field, if no form or object input is provided. May be a callable.
- **widget** – If provided, overrides the widget used to render the field.
- **render_kw** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **_form** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.
- **_name** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.
- **_prefix** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.
- **_translations** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.
- **_meta** – If provided, this is the 'meta' instance from the form. You usually don't pass this yourself.

If *_form* and *_name* isn't provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.

CAN_POPULATE = True

accepted_condition_types = (<class 'indico.modules.events.abstracts.notifications.StateCondition'>, <class 'indico

`condition_choices`

`condition_class_map = {u'track': <class 'indico.modules.events.abstracts.notifications.TrackCondition'>, u'state': <`

`pre_validate (form)`

`widget = <indico.web.forms.widgets.JinjaWidget object>`

```
class indico.modules.events.abstracts.fields.TrackRoleField (label=None,      val-
                                                           idators=None,      fil-
                                                           ters=(),           descrip-
                                                           tion=u'',          id=None,
                                                           default=None,      wid-
                                                           get=None,          ren-
                                                           der_kw=None,
                                                           _form=None,
                                                           _name=None,       _pre-
                                                           fix=u'',           _trans-
                                                           lations=None,
                                                           _meta=None)
```

Bases: `indico.web.forms.fields.simple.JSONField`

A field that stores a list of e-mail template rules.

Construct a new field.

Parameters

- **label** – The label of the field.
- **validators** – A sequence of validators to call when *validate* is called.
- **filters** – A sequence of filters which are run on input data by *process*.
- **description** – A description for the field, typically used for help text.
- **id** – An id to use for the field. A reasonable default is set by the form, and you shouldn't need to set this manually.
- **default** – The default value to assign to the field, if no form or object input is provided. May be a callable.
- **widget** – If provided, overrides the widget used to render the field.
- **render_kw** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **_form** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.
- **_name** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.
- **_prefix** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.
- **_translations** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.
- **_meta** – If provided, this is the 'meta' instance from the form. You usually don't pass this yourself.

If *_form* and *_name* isn't provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.

CAN_POPULATE = True

role_data

users

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.modules.events.contributions.fields.ContributionPersonLinkListField` (**args*, ***kwargs*)

Bases: `indico.modules.events.fields.PersonLinkListFieldBase`

A field to configure a list of contribution persons

linked_object_attr = u'contrib'

person_link_cls

alias of `ContributionPersonLink`

pre_validate (*form*)

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.modules.events.contributions.fields.SubContributionPersonLinkListField` (**args*, ***kwargs*)

Bases: `indico.modules.events.contributions.fields.ContributionPersonLinkListField`

A field to configure a list of subcontribution persons

linked_object_attr = u'subcontrib'

person_link_cls

alias of `SubContributionPersonLink`

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.modules.events.papers.fields.PaperEmailSettingsField` (*label=None*, *validators=None*, *filters=()*, *description=u''*, *id=None*, *default=None*, *widget=None*, *render_kw=None*, *_form=None*, *_name=None*, *_prefix=u''*, *_translations=None*, *_meta=None*)

Bases: `indico.web.forms.fields.simple.JSONField`

Construct a new field.

Parameters

- **label** – The label of the field.
- **validators** – A sequence of validators to call when *validate* is called.
- **filters** – A sequence of filters which are run on input data by *process*.
- **description** – A description for the field, typically used for help text.

- **id** – An id to use for the field. A reasonable default is set by the form, and you shouldn't need to set this manually.
- **default** – The default value to assign to the field, if no form or object input is provided. May be a callable.
- **widget** – If provided, overrides the widget used to render the field.
- **render_kw** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **_form** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.
- **_name** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.
- **_prefix** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.
- **_translations** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.
- **_meta** – If provided, this is the 'meta' instance from the form. You usually don't pass this yourself.

If *_form* and *_name* isn't provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.

CAN_POPULATE = True

event

process_formdata (*valuelist*)

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.modules.events.sessions.fields.SessionBlockPersonLinkListField` (**args*, ***kwargs*)

Bases: `indico.modules.events.fields.PersonLinkListFieldBase`

linked_object_attr = u'session_block'

person_link_cls

alias of `SessionBlockPersonLink`

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.modules.categories.fields.CategoryField` (**args*, ***kwargs*)

Bases: `wtforms.fields.simple.HiddenField`

WTForms field that lets you select a category.

Parameters

- **allow_events** – Whether to allow selecting a category that contains events.
- **allow_subcats** – Whether to allow selecting a category that contains subcategories.
- **require_event_creation_rights** – Whether to allow selecting only categories where the user can create events.

pre_validate (*form*)

process_data (*value*)

process_formdata (*valuelist*)

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.modules.networks.fields.MultiIPNetworkField` (**args, **kwargs*)

Bases: `indico.web.forms.fields.itemlists.MultiStringField`

A field to enter multiple IPv4 or IPv6 networks.

The field data is a set of `IPNetwork`'s not bound to a DB session. The `unique` and `sortable` parameters of the parent class cannot be used with this class.

pre_validate (*form*)

process_data (*value*)

process_formdata (*valuelist*)

class `indico.web.forms.fields.IndicoSelectMultipleCheckboxField` (*label=None, validators=None, coerce=<type 'unicode'>, choices=None, **kwargs*)

Bases: `wtforms.fields.core.SelectMultipleField`

option_widget = <wtforms.widgets.core.CheckboxInput object>

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.web.forms.fields.IndicoRadioField` (**args, **kwargs*)

Bases: `wtforms.fields.core.RadioField`

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.web.forms.fields.JSONField` (*label=None, validators=None, filters=(), description=u'', id=None, default=None, widget=None, render_kw=None, _form=None, _name=None, _prefix=u'', _translations=None, _meta=None*)

Bases: `wtforms.fields.simple.HiddenField`

Construct a new field.

Parameters

- **label** – The label of the field.
- **validators** – A sequence of validators to call when *validate* is called.
- **filters** – A sequence of filters which are run on input data by *process*.
- **description** – A description for the field, typically used for help text.
- **id** – An id to use for the field. A reasonable default is set by the form, and you shouldn't need to set this manually.
- **default** – The default value to assign to the field, if no form or object input is provided. May be a callable.
- **widget** – If provided, overrides the widget used to render the field.
- **render_kw** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **_form** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.

- **`_name`** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.
- **`_prefix`** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.
- **`_translations`** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.
- **`_meta`** – If provided, this is the ‘meta’ instance from the form. You usually don’t pass this yourself.

If `_form` and `_name` isn’t provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.

`CAN_POPULATE = False`

`populate_obj` (*obj, name*)

`process_formdata` (*valuelist*)

class `indico.web.forms.fields.HiddenFieldList` (*label=None, validators=None, filters=(), description=u'', id=None, default=None, widget=None, render_kw=None, _form=None, _name=None, _prefix=u'', _translations=None, _meta=None*)

Bases: `wtforms.fields.simple.HiddenField`

A hidden field that handles lists of strings.

This is done *getlist*-style, i.e. by repeating the input element with the same name for each list item.

The only case where this field is useful is when you display a form via POST and provide a list of items (e.g. ids) related to the form which needs to be kept when the form is submitted and also need to access it via `request.form.getlist(...)` before submitting the form.

Construct a new field.

Parameters

- **`label`** – The label of the field.
- **`validators`** – A sequence of validators to call when *validate* is called.
- **`filters`** – A sequence of filters which are run on input data by *process*.
- **`description`** – A description for the field, typically used for help text.
- **`id`** – An id to use for the field. A reasonable default is set by the form, and you shouldn’t need to set this manually.
- **`default`** – The default value to assign to the field, if no form or object input is provided. May be a callable.
- **`widget`** – If provided, overrides the widget used to render the field.
- **`render_kw`** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **`_form`** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.
- **`_name`** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.

- **`_prefix`** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.
- **`_translations`** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.
- **`_meta`** – If provided, this is the ‘meta’ instance from the form. You usually don’t pass this yourself.

If `_form` and `_name` isn’t provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.

`process_formdata` (*valuelist*)

`widget` = `<indico.web.forms.widgets.HiddenInputs object>`

```
class indico.web.forms.fields.TextListField(label=None, validators=None, filters=(),
description=u'', id=None, default=None, widget=None, render_kw=None, _form=None,
_name=None, _prefix=u'', _translations=None, _meta=None)
```

Bases: `wtforms.fields.simple.TextAreaField`

Construct a new field.

Parameters

- **`label`** – The label of the field.
- **`validators`** – A sequence of validators to call when `validate` is called.
- **`filters`** – A sequence of filters which are run on input data by `process`.
- **`description`** – A description for the field, typically used for help text.
- **`id`** – An id to use for the field. A reasonable default is set by the form, and you shouldn’t need to set this manually.
- **`default`** – The default value to assign to the field, if no form or object input is provided. May be a callable.
- **`widget`** – If provided, overrides the widget used to render the field.
- **`render_kw`** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **`_form`** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.
- **`_name`** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.
- **`_prefix`** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.
- **`_translations`** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.
- **`_meta`** – If provided, this is the ‘meta’ instance from the form. You usually don’t pass this yourself.

If `_form` and `_name` isn’t provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.

`pre_validate` (*form*)

`process_formdata` (*valuelist*)

class `indico.web.forms.fields.EmailListField` (*label=None, validators=None, filters=(), description=u'', id=None, default=None, widget=None, render_kw=None, _form=None, _name=None, _prefix=u'', _translations=None, _meta=None*)

Bases: `indico.web.forms.fields.simple.TextListField`

Construct a new field.

Parameters

- **label** – The label of the field.
- **validators** – A sequence of validators to call when *validate* is called.
- **filters** – A sequence of filters which are run on input data by *process*.
- **description** – A description for the field, typically used for help text.
- **id** – An id to use for the field. A reasonable default is set by the form, and you shouldn't need to set this manually.
- **default** – The default value to assign to the field, if no form or object input is provided. May be a callable.
- **widget** – If provided, overrides the widget used to render the field.
- **render_kw** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **_form** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.
- **_name** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.
- **_prefix** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.
- **_translations** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.
- **_meta** – If provided, this is the 'meta' instance from the form. You usually don't pass this yourself.

If *_form* and *_name* isn't provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.

`process_formdata` (*valuelist*)

class `indico.web.forms.fields.IndicoPasswordField` (**args, **kwargs*)

Bases: `wtforms.fields.simple.PasswordField`

Password field which can show or hide the password.

widget = `<indico.web.forms.widgets.PasswordWidget object>`

class `indico.web.forms.fields.IndicoStaticTextField` (**args, **kwargs*)

Bases: `wtforms.fields.core.Field`

Return an html element with text taken from this field's value

process_data (*data*)

widget = <indico.web.forms.widgets.JinjaWidget object>

```
class indico.web.forms.fields.IndicoTagListField (label=None, validators=None, filters=(), description=u', id=None, default=None, widget=None, render_kw=None, _form=None, _name=None, _prefix=u', _translations=None, _meta=None)
```

Bases: indico.web.forms.fields.simple.HiddenFieldList

Construct a new field.

Parameters

- **label** – The label of the field.
- **validators** – A sequence of validators to call when *validate* is called.
- **filters** – A sequence of filters which are run on input data by *process*.
- **description** – A description for the field, typically used for help text.
- **id** – An id to use for the field. A reasonable default is set by the form, and you shouldn't need to set this manually.
- **default** – The default value to assign to the field, if no form or object input is provided. May be a callable.
- **widget** – If provided, overrides the widget used to render the field.
- **render_kw** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **_form** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.
- **_name** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.
- **_prefix** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.
- **_translations** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.
- **_meta** – If provided, this is the 'meta' instance from the form. You usually don't pass this yourself.

If *_form* and *_name* isn't provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.

widget = <indico.web.forms.widgets.JinjaWidget object>

```
class indico.web.forms.fields.IndicoPalettePickerField (*args, **kwargs)
```

Bases: indico.web.forms.fields.simple.JSONField

Field allowing user to pick a color from a set of predefined values

CAN_POPULATE = True

pre_validate (*form*)

process_data (*value*)

process_formdata (*valuelist*)

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.web.forms.fields.TimeDeltaField` (**args*, ***kwargs*)

Bases: `wtforms.fields.core.Field`

A field that lets the user select a simple timedelta.

It does not support mixing multiple units, but it is smart enough to switch to a different unit to represent a timedelta that could not be represented otherwise.

Parameters **units** – The available units. Must be a tuple containing any any of ‘seconds’, ‘minutes’, ‘hours’ and ‘days’. If not specified, (‘hours’, ‘days’) is assumed.

best_unit

Return the largest unit that covers the current timedelta

choices

magnitudes = `OrderedDict([(‘days’, 86400), (‘hours’, 3600), (‘minutes’, 60), (‘seconds’, 1)])`

pre_validate (*form*)

process_formdata (*valuelist*)

unit_names = {‘seconds’: ‘Seconds’, ‘hours’: ‘Hours’, ‘minutes’: ‘Minutes’, ‘days’: ‘Days’}

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.web.forms.fields.IndicoDateTimeField` (**args*, ***kwargs*)

Bases: `wtforms.ext.dateutil.fields.DateTimeField`

Friendly datetime field that handles timezones and validations.

Important: When the form has a *timezone* field it must be declared before any *IndicoDateTimeField*. Otherwise its value is not available in this field resulting in an error during form submission.

earliest_dt

latest_dt

linked_datetime_validator

linked_field

pre_validate (*form*)

process_formdata (*valuelist*)

timezone

timezone_field

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.web.forms.fields.OccurrencesField` (**args*, ***kwargs*)

Bases: `indico.web.forms.fields.simple.JSONField`

A field that lets you select multiple occurrences consisting of a start date/time and a duration.

CAN_POPULATE = `True`

process_formdata (*valuelist*)

timezone

timezone_field

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.web.forms.fields.IndicoTimezoneSelectField` (*args, **kwargs)

Bases: `wtforms.fields.core.SelectField`

process_data (value)

class `indico.web.forms.fields.IndicoEnumSelectField` (label=None, validators=None, enum=None, sorted=False, only=None, skip=None, none=None, titles=None, **kwargs)

Bases: `indico.web.forms.fields.enums._EnumFieldMixin`, `wtforms.fields.core.SelectFieldBase`

Select field backed by a RichEnum

iter_choices ()

widget = <wtforms.widgets.core.Select object>

class `indico.web.forms.fields.IndicoEnumRadioField` (label=None, validators=None, enum=None, sorted=False, only=None, skip=None, none=None, titles=None, **kwargs)

Bases: `indico.web.forms.fields.enums.IndicoEnumSelectField`

option_widget = <wtforms.widgets.core.RadioInput object>

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.web.forms.fields.HiddenEnumField` (label=None, validators=None, enum=None, only=None, skip=None, none=None, **kwargs)

Bases: `indico.web.forms.fields.enums._EnumFieldMixin`, `wtforms.fields.simple.HiddenField`

Hidden field that only accepts values from an Enum

process_formdata (valuelist)

class `indico.web.forms.fields.FileField` (*args, **kwargs)

Bases: `wtforms.fields.core.Field`

A dropzone field

default_options = {u'multiple_files': False, u'handle_flashes': False, u'lightweight': False, u'add_remove_links': True}

process_formdata (valuelist)

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.web.forms.fields.MultiStringField` (*args, **kwargs)

Bases: `wtforms.fields.simple.HiddenField`

A field with multiple input text fields.

Parameters

- **field** – A tuple (fieldname, title) where the title is used in the placeholder.
- **uuid_field** – If set, each item will have a UUID assigned and stored in the field specified here.
- **flat** – If True, the field returns a list of string values instead of dicts. Cannot be combined with `uuid_field`.

- **unique** – Whether the values should be unique.
- **sortable** – Whether items should be sortable.

pre_validate (*form*)

process_formdata (*valuelist*)

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.web.forms.fields.MultipleItemsField` (*args, **kwargs)

Bases: `wtforms.fields.simple.HiddenField`

A field with multiple items consisting of multiple string values.

Parameters

- **fields** – A list of dicts with the following arguments: ‘id’: the unique ID of the field ‘caption’: the title of the column and the placeholder ‘type’: ‘text|number|select’, the type of the field ‘coerce’: callable to convert the value to a python type.

the type must be convertible back to a string, so usually you just want something like *int* or *float* here.

In case the type is ‘select’, the property ‘choices’ of the *MultipleItemsField* or the ‘choices’ kwarg needs to be a dict where the key is the ‘id’ of the select field and the value is another dict mapping the option’s id to its caption.

- **uuid_field** – If set, each item will have a UUID assigned and stored in the field specified here. The name specified here may not be in *fields*.
- **uuid_field_opaque** – If set, the *uuid_field* is considered opaque, i.e. it is never touched by this field. This is useful when you subclass the field and use e.g. actual database IDs instead of UUIDs.
- **unique_field** – The name of a field in *fields* that needs to be unique.
- **sortable** – Whether items should be sortable.

pre_validate (*form*)

process_formdata (*valuelist*)

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.web.forms.fields.OverrideMultipleItemsField` (*args, **kwargs)

Bases: `wtforms.fields.simple.HiddenField`

A field similar to *MultipleItemsField* which allows the user to override some values.

Parameters

- **fields** – a list of (*fieldname*, *title*) tuples. Should match the fields of the corresponding *MultipleItemsField*.
- **field_data** – the data from the corresponding *MultipleItemsField*.
- **unique_field** – the name of the field which is unique among all rows
- **edit_fields** – a set containing the field names which can be edited

If you decide to use this field, please consider adding support for *uuid_field* here!

get_overridden_value (*row*, *name*)

Utility for the widget to get the entered value for an editable field

get_row_key (*row*)
Utility for the widget to get the unique value for a row

pre_validate (*form*)

process_formdata (*valuelist*)

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.web.forms.fields.PrincipalListField` (**args, **kwargs*)

Bases: `wtforms.fields.simple.HiddenField`

A field that lets you select a list Indico user/group (“principal”)

Parameters

- **groups** – If groups should be selectable.
- **allow_networks** – If ip networks should be selectable.
- **allow_emails** – If emails should be allowed.
- **allow_external** – If “search users with no indico account” should be available. Selecting such a user will automatically create a pending user once the form is submitted, even if other fields in the form fail to validate!

pre_validate (*form*)

process_formdata (*valuelist*)

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.web.forms.fields.PrincipalField` (**args, **kwargs*)

Bases: `indico.web.forms.fields.principals.PrincipalListField`

A field that lets you select an Indico user/group (“principal”)

process_formdata (*valuelist*)

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.web.forms.fields.AccessControlListField` (**args, **kwargs*)

Bases: `indico.web.forms.fields.principals.PrincipalListField`

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.web.forms.fields.IndicoQuerySelectMultipleField` (**args, **kwargs*)

Bases: `wtforms.ext.sqlalchemy.fields.QuerySelectMultipleField`

Like the parent, but with a callback that allows you to modify the list

The callback can return a new list or yield items, and you can use it e.g. to sort the list.

data

class `indico.web.forms.fields.IndicoLocationField` (**args, **kwargs*)

Bases: `indico.web.forms.fields.simple.JSONField`

CAN_POPULATE = True

process_formdata (*valuelist*)

widget = <indico.web.forms.widgets.LocationWidget object>

class `indico.web.forms.fields.IndicoMarkdownField` (**args, **kwargs*)

Bases: `wtforms.fields.simple.TextAreaField`

A Markdown-enhanced textarea.

When using the editor you need to include the markdown JS/CSS bundles and also the MathJax JS bundle (even when using only the editor without Mathjax).

Parameters

- **editor** – Whether to use the WMD widget with its live preview
- **mathjax** – Whether to use MathJax in the WMD live preview

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.web.forms.fields.IndicoDateField` (*args, **kwargs)

Bases: `wtforms.ext.dateutil.fields.DateField`

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.web.forms.fields.IndicoProtectionField` (*args, **kwargs)

Bases: `indico.web.forms.fields.enums.IndicoEnumRadioField`

radio_widget = <indico.web.forms.widgets.JinjaWidget object>

render_protection_message ()

widget = <indico.web.forms.widgets.JinjaWidget object>

class `indico.web.forms.fields.IndicoSelectMultipleCheckboxBooleanField` (*label=None,*

validators=None,
coerce=<type
'unicode'>,
choices=None,
***kwargs*)

Bases: `indico.web.forms.fields.simple.IndicoSelectMultipleCheckboxField`

iter_choices ()

process_formdata (*valuelist*)

class `indico.web.forms.fields.RelativeDeltaField` (*args, **kwargs)

Bases: `wtforms.fields.core.Field`

A field that lets the user select a simple timedelta.

It does not support mixing multiple units, but it is smart enough to switch to a different unit to represent a timedelta that could not be represented otherwise.

Parameters **units** – The available units. Must be a tuple containing any any of ‘seconds’, ‘minutes’, ‘hours’ and ‘days’. If not specified, ('hours', 'days') is assumed.

choices

magnitudes = `OrderedDict([(u'years', relativedelta(years=+1)), (u'months', relativedelta(months=+1)), (u'weeks', relativedelta(weeks=+1))])`

pre_validate (*form*)

process_formdata (*valuelist*)

split_data

unit_names = {u'seconds': u'Seconds', u'months': u'Months', u'days': u'Days', u'years': u'Years', u'hours': u'Hours'}

widget = <indico.web.forms.widgets.JinjaWidget object>


```
class indico.web.forms.fields.IndicoWeekDayRepetitionField(*args, **kwargs)
    Bases: wtforms.fields.core.Field

    Field that lets you select an ordinal day of the week.

    WEEK_DAY_NUMBER_CHOICES = ((1, lu'first'), (2, lu'second'), (3, lu'third'), (4, lu'fourth'), (-1, lu'last'))

    day_number_data
    process_formdata(valuelist)
    week_day_data
    widget = <indico.web.forms.widgets.JinjaWidget object>
```

```
class indico.web.forms.fields.IndicoEmailRecipientsField(label=None, validators=None, filters=(), description=u'', id=None, default=None, widget=None, render_kw=None, _form=None, _name=None, _prefix=u'', _translations=None, _meta=None)
```

Bases: wtforms.fields.core.Field

Construct a new field.

Parameters

- **label** – The label of the field.
- **validators** – A sequence of validators to call when *validate* is called.
- **filters** – A sequence of filters which are run on input data by *process*.
- **description** – A description for the field, typically used for help text.
- **id** – An id to use for the field. A reasonable default is set by the form, and you shouldn't need to set this manually.
- **default** – The default value to assign to the field, if no form or object input is provided. May be a callable.
- **widget** – If provided, overrides the widget used to render the field.
- **render_kw** (*dict*) – If provided, a dictionary which provides default keywords that will be given to the widget at render time.
- **_form** – The form holding this field. It is passed by the form itself during construction. You should never pass this value yourself.
- **_name** – The name of this field, passed by the enclosing form during its construction. You should never pass this value yourself.
- **_prefix** – The prefix to prepend to the form name of this field, passed by the enclosing form during construction.
- **_translations** – A translations object providing message translations. Usually passed by the enclosing form during construction. See I18n docs for information on message translations.
- **_meta** – If provided, this is the 'meta' instance from the form. You usually don't pass this yourself.

If *_form* and *_name* isn't provided, an `UnboundField` will be returned instead. Call its `bind()` method with a form instance and a name to construct the field.

`process_data` (*data*)

`widget` = <indico.web.forms.widgets.JinjaWidget object>

CHAPTER 6

Indices and tables

- `genindex`
- `modindex`

i

`indico.core.plugins`, 39
`indico.modules.attachments`, 219
`indico.modules.attachments.models.attachments`,
219
`indico.modules.attachments.models.folders`,
222
`indico.modules.attachments.models.principals`,
224
`indico.modules.attachments.operations`,
225
`indico.modules.attachments.preview`, 226
`indico.modules.attachments.util`, 225
`indico.modules.auth`, 241
`indico.modules.auth.models.identities`,
241
`indico.modules.auth.models.registration_requests`,
242
`indico.modules.auth.util`, 243
`indico.modules.categories`, 204
`indico.modules.categories.fields`, 266
`indico.modules.categories.models.categories`,
204
`indico.modules.categories.models.principals`,
207
`indico.modules.categories.models.settings`,
208
`indico.modules.categories.operations`,
208
`indico.modules.categories.serialize`, 209
`indico.modules.categories.settings`, 210
`indico.modules.categories.util`, 208
`indico.modules.designer`, 252
`indico.modules.designer.models.images`,
252
`indico.modules.designer.models.templates`,
253
`indico.modules.designer.pdf`, 254
`indico.modules.designer.placeholders`,
255
`indico.modules.designer.util`, 254
`indico.modules.events`, 79
`indico.modules.events.abstracts`, 99
`indico.modules.events.abstracts.fields`,
262
`indico.modules.events.abstracts.models.abstracts`,
100
`indico.modules.events.abstracts.models.call_for_ab`,
103
`indico.modules.events.abstracts.models.comments`,
104
`indico.modules.events.abstracts.models.email_logs`,
105
`indico.modules.events.abstracts.models.email_templ`,
105
`indico.modules.events.abstracts.models.fields`,
106
`indico.modules.events.abstracts.models.files`,
107
`indico.modules.events.abstracts.models.persons`,
107
`indico.modules.events.abstracts.models.related_tra`,
108
`indico.modules.events.abstracts.models.review_quest`,
108
`indico.modules.events.abstracts.models.review_rati`,
108
`indico.modules.events.abstracts.models.reviews`,
109
`indico.modules.events.abstracts.operations`,
111
`indico.modules.events.abstracts.placeholders`,
113
`indico.modules.events.abstracts.settings`,
116
`indico.modules.events.abstracts.util`,
112
`indico.modules.events.agreements`, 116
`indico.modules.events.agreements.models.agreements`,

117
 indico.modules.events.agreements.placeholders, 119
 indico.modules.events.agreements.util, 118
 indico.modules.events.contributions, 119
 indico.modules.events.contributions.fields, 265
 indico.modules.events.contributions.models.contributors, 119
 indico.modules.events.contributions.models.fields, 122
 indico.modules.events.contributions.models.persons, 123
 indico.modules.events.contributions.models.principals, 125
 indico.modules.events.contributions.models.references, 126
 indico.modules.events.contributions.models.submitters, 126
 indico.modules.events.contributions.models.types, 127
 indico.modules.events.contributions.operations, 128
 indico.modules.events.contributions.util, 129
 indico.modules.events.features, 130
 indico.modules.events.features.util, 130
 indico.modules.events.fields, 261
 indico.modules.events.layout, 130
 indico.modules.events.layout.models.images, 131
 indico.modules.events.layout.models.menu, 131
 indico.modules.events.layout.util, 133
 indico.modules.events.logs, 135
 indico.modules.events.logs.models.entries, 135
 indico.modules.events.logs.renderers, 136
 indico.modules.events.logs.util, 136
 indico.modules.events.models.events, 79
 indico.modules.events.models.persons, 85
 indico.modules.events.models.principals, 87
 indico.modules.events.models.references, 88
 indico.modules.events.models.reviews, 89
 indico.modules.events.models.series, 91
 indico.modules.events.models.settings, 92
 indico.modules.events.models.static_lists, 93
 indico.modules.events.notes, 138
 indico.modules.events.notes.models.notes, 138
 indico.modules.events.notes.util, 140
 indico.modules.events.operations, 94
 indico.modules.events.papers, 140
 indico.modules.events.papers.fields, 265
 indico.modules.events.papers.models.call_for_papers, 141
 indico.modules.events.papers.models.comments, 141
 indico.modules.events.papers.models.competences, 142
 indico.modules.events.papers.models.files, 143
 indico.modules.events.papers.models.papers, 144
 indico.modules.events.papers.models.review_questions, 145
 indico.modules.events.papers.models.review_ratings, 145
 indico.modules.events.papers.models.reviews, 146
 indico.modules.events.papers.models.revisions, 148
 indico.modules.events.papers.models.templates, 149
 indico.modules.events.papers.models.user_contributions, 150
 indico.modules.events.papers.operations, 150
 indico.modules.events.papers.util, 151
 indico.modules.events.payment, 154
 indico.modules.events.payment.models.transactions, 154
 indico.modules.events.payment.plugins, 156
 indico.modules.events.payment.util, 156
 indico.modules.events.persons, 157
 indico.modules.events.persons.operations, 157
 indico.modules.events.persons.placeholders, 157
 indico.modules.events.registration, 158
 indico.modules.events.registration.models.form_fields, 162
 indico.modules.events.registration.models.forms, 164
 indico.modules.events.registration.models.invitations, 167
 indico.modules.events.registration.models.items, 168
 indico.modules.events.registration.models.registrations, 168

158
 indico.modules.events.registration.placeholders.models.events.timetable.operations,
 175
 indico.modules.events.registration.placeholders.models.events.timetable.reschedule,
 174
 indico.modules.events.registration.settings, 175
 indico.modules.events.registration.stats, 176
 indico.modules.events.registration.util, 172
 indico.modules.events.reminders, 177
 indico.modules.events.reminders.models.reminders, 177
 indico.modules.events.reminders.util, 178
 indico.modules.events.requests, 179
 indico.modules.events.requests.base, 180
 indico.modules.events.requests.models.requests, 179
 indico.modules.events.requests.util, 180
 indico.modules.events.sessions, 182
 indico.modules.events.sessions.fields, 266
 indico.modules.events.sessions.models.blocks, 184
 indico.modules.events.sessions.models.persons, 185
 indico.modules.events.sessions.models.principals, 186
 indico.modules.events.sessions.models.sessions, 182
 indico.modules.events.sessions.operations, 187
 indico.modules.events.sessions.util, 187
 indico.modules.events.settings, 151
 indico.modules.events.static, 202
 indico.modules.events.static.models.static, 202
 indico.modules.events.static.util, 203
 indico.modules.events.surveys, 188
 indico.modules.events.surveys.models.items, 190
 indico.modules.events.surveys.models.submissions, 192
 indico.modules.events.surveys.models.surveys, 188
 indico.modules.events.surveys.operations, 194
 indico.modules.events.surveys.util, 194
 indico.modules.events.timetable, 195
 indico.modules.events.timetable.models.blocks, 195
 indico.modules.events.timetable.models.entities, 196
 indico.modules.events.timetable.operations, 198
 indico.modules.events.timetable.reschedule, 200
 indico.modules.events.timetable.util, 199
 indico.modules.events.tracks, 201
 indico.modules.events.tracks.models.abstract_reviewers, 202
 indico.modules.events.tracks.models.conveners, 202
 indico.modules.events.tracks.models.tracks, 201
 indico.modules.events.tracks.operations, 202
 indico.modules.events.util, 95
 indico.modules.groups, 246
 indico.modules.groups.core, 247
 indico.modules.groups.models.groups, 246
 indico.modules.groups.util, 248
 indico.modules.networks, 259
 indico.modules.networks.fields, 267
 indico.modules.networks.models.networks, 259
 indico.modules.networks.util, 260
 indico.modules.news, 260
 indico.modules.news.models.news, 260
 indico.modules.news.util, 261
 indico.modules.oauth, 243
 indico.modules.oauth.models.applications, 244
 indico.modules.oauth.models.tokens, 245
 indico.modules.oauth.provider, 246
 indico.modules.rb, 227
 indico.modules.rb.models.aspects, 231
 indico.modules.rb.models.blocked_rooms, 232
 indico.modules.rb.models.blocking_principals, 233
 indico.modules.rb.models.blockings, 232
 indico.modules.rb.models.equipment, 233
 indico.modules.rb.models.holidays, 234
 indico.modules.rb.models.locations, 234
 indico.modules.rb.models.photos, 235
 indico.modules.rb.models.reservation_edit_logs, 238
 indico.modules.rb.models.reservation_occurrences, 239
 indico.modules.rb.models.reservations, 235
 indico.modules.rb.models.room_attributes, 230
 indico.modules.rb.models.room_bookable_hours,

231
indico.modules.rb.models.room_nonbookable_periods,
231
indico.modules.rb.models.rooms, 227
indico.modules.rb.models.util, 239
indico.modules.rb.services.aspects, 241
indico.modules.rb.services.blockings,
241
indico.modules.rb.services.rooms, 240
indico.modules.rb.statistics, 240
indico.modules.rb.util, 240
indico.modules.users, 211
indico.modules.users.ext, 219
indico.modules.users.models.affiliations,
215
indico.modules.users.models.emails, 215
indico.modules.users.models.favorites,
215
indico.modules.users.models.settings,
216
indico.modules.users.models.suggestions,
215
indico.modules.users.models.users, 211
indico.modules.users.operations, 217
indico.modules.users.util, 218
indico.modules.vc, 248
indico.modules.vc.exceptions, 252
indico.modules.vc.models.vc_rooms, 248
indico.modules.vc.plugins, 250
indico.modules.vc.util, 250
indico.web.forms.fields, 267

A

- absolute_download_url (in module indico.core.signals.event), 45
- absolute_download_url (in module indico.modules.attachments.models.attachments.Attachment attribute), 220
- Abstract (class in indico.modules.events.abstracts.models.abstracts), 100
- abstract (indico.modules.events.abstracts.models.comments.AbstractComment attribute), 104
- abstract (indico.modules.events.abstracts.models.email_logs.AbstractEmailLogEntry attribute), 105
- abstract (indico.modules.events.abstracts.models.files.AbstractFile attribute), 107
- abstract (indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 109
- abstract (indico.modules.events.contributions.models.contributions.Contribution attribute), 119
- abstract_created (in module indico.core.signals.event), 45
- abstract_deleted (in module indico.core.signals.event), 45
- abstract_email_log_entries (in module indico.modules.users.models.users.User attribute), 212
- abstract_id (indico.modules.events.abstracts.models.comments.AbstractComment attribute), 104
- abstract_id (indico.modules.events.abstracts.models.email_logs.AbstractEmailLogEntry attribute), 105
- abstract_id (indico.modules.events.abstracts.models.files.AbstractFile attribute), 107
- abstract_id (indico.modules.events.abstracts.models.files.AbstractIDPlaceholder attribute), 107
- abstract_id (indico.modules.events.abstracts.models.persons.AbstractPersonLink attribute), 107
- abstract_id (indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 109
- abstract_id (indico.modules.events.contributions.models.contributions.Contribution attribute), 119
- abstract_reviewers (indico.modules.events.tracks.models.tracks.Track attribute), 201
- abstract_state_changed (in module indico.modules.events.abstracts.models.abstracts), 102
- abstract_title (indico.modules.events.abstracts.settings.BOASortField attribute), 116
- abstract_updated (in module indico.core.signals.event), 45
- AbstractAction (class in indico.modules.events.abstracts.models.reviews), 109
- AbstractComment (class in indico.modules.events.abstracts.models.comments), 104
- AbstractCommentVisibility (class in indico.modules.events.abstracts.models.reviews), 109
- AbstractEmailLogEntry (class in indico.modules.events.abstracts.models.email_logs), 105
- AbstractEmailTemplate (class in indico.modules.events.abstracts.models.email_templates), 105
- AbstractField (class in indico.modules.events.abstracts.fields), 262
- AbstractFieldValue (class in indico.modules.events.abstracts.models.fields), 106
- AbstractFile (class in indico.modules.events.abstracts.models.files), 107
- AbstractIDPlaceholder (class in indico.modules.events.abstracts.placeholders), 113
- AbstractPersonLink (class in indico.modules.events.abstracts.models.persons), 107
- AbstractPersonLinkListField (class in indico.modules.events.abstracts.fields), 262
- AbstractPublicState (class in indico.modules.events.abstracts.models.abstracts), 102
- AbstractReview (class in indico.modules.events.abstracts.models.reviews), 109

dico.modules.events.abstracts.models.reviews),
109

AbstractReviewingState (class in in-
dico.modules.events.abstracts.models.abstracts),
102

AbstractReviewQuestion (class in in-
dico.modules.events.abstracts.models.review_questsions),
108

AbstractReviewRating (class in in-
dico.modules.events.abstracts.models.review_ratings),
108

abstracts (indico.modules.users.models.users.User
attribute), 212

abstracts_accepted (indico.modules.events.tracks.models.tracks.Track
attribute), 201

abstracts_reviewed (in-
dico.modules.events.tracks.models.tracks.Track
attribute), 201

abstracts_submitted (in-
dico.modules.events.tracks.models.tracks.Track
attribute), 201

AbstractSessionPlaceholder (class in in-
dico.modules.events.abstracts.placeholders),
114

AbstractState (class in in-
dico.modules.events.abstracts.models.abstracts),
103

AbstractTitlePlaceholder (class in in-
dico.modules.events.abstracts.placeholders),
113

AbstractTrackPlaceholder (class in in-
dico.modules.events.abstracts.placeholders),
114

AbstractURLPlaceholder (class in in-
dico.modules.events.abstracts.placeholders),
113

accept (indico.modules.events.abstracts.models.reviews.AbstractReviewingState
attribute), 109

accept (indico.modules.events.papers.models.reviews.PaperReviewingState
attribute), 146

accept() (indico.modules.events.agreements.models.agreements.AgreementListField
method), 117

accept() (indico.modules.events.requests.base.RequestDefinition class method), 180

accept() (indico.modules.rb.models.reservations.Reservation class method), 236

accepted (indico.modules.events.abstracts.models.abstracts.AbstractState
attribute), 102

accepted (indico.modules.events.abstracts.models.abstracts.AbstractReviewingState
attribute), 103

accepted (indico.modules.events.agreements.models.agreements.AgreementListField
attribute), 117

accepted (indico.modules.events.agreements.models.agreements.AgreementListField
attribute), 118

accepted (indico.modules.events.papers.models.revisions.PaperRevisionState
attribute), 149

accepted (indico.modules.events.registration.models.invitations.InvitationState
attribute), 167

accepted (indico.modules.events.requests.models.requests.RequestState
attribute), 180

accepted (indico.modules.rb.models.blocked_rooms.BlockedRoomState
attribute), 233

accepted_condition_types (in-
dico.modules.events.abstracts.fields.EmailRuleListField
attribute), 263

accepted_contrib_type (in-
dico.modules.events.abstracts.models.abstracts.AbstractState
attribute), 100

accepted_contrib_type_id (in-
dico.modules.events.abstracts.models.abstracts.AbstractState
attribute), 100

accepted_on_behalf (in-
dico.modules.events.agreements.models.agreements.AgreementListField
attribute), 118

accepted_revision (indico.modules.events.papers.models.papers.PaperRevisionState
attribute), 144

accepted_track (indico.modules.events.abstracts.models.abstracts.AbstractState
attribute), 100

accepted_track_id (indico.modules.events.abstracts.models.abstracts.AbstractState
attribute), 100

access_key (indico.modules.attachments.models.attachments.Attachment class
attribute), 220

access_key (indico.modules.attachments.models.folders.AttachmentFolder class
attribute), 222

access_key (indico.modules.categories.models.categories.Category class
attribute), 204

access_key (indico.modules.events.contributions.models.contributions.Contribution class
attribute), 119

access_key (indico.modules.events.models.events.Event class
attribute), 80

actions (indico.modules.events.sessions.models.sessions.Session class
attribute), 182

actions_token (indico.modules.oauth.models.tokens.OAuthToken class
attribute), 245

AccessControlListField (class in in-
dico.web.forms.fields), 275

AccommodationStats (class in in-
dico.modules.events.registration.stats), 176

acl (indico.modules.attachments.models.attachments.Attachment class
attribute), 220

acl (indico.modules.attachments.models.folders.AttachmentFolder class
attribute), 222

AbstractAttachment (indico.modules.attachments.models.attachments.Attachment class
attribute), 220

AbstractAttachmentFolder (indico.modules.attachments.models.folders.AttachmentFolder class
attribute), 222

AbstractCategory (indico.modules.categories.models.categories.Category class
attribute), 204

[acl_entries \(indico.modules.events.contributions.models.contributions.VCContribution attribute\), 119](#)
[acl_entries \(indico.modules.events.models.events.Event attribute\), 80](#)
[acl_entries \(indico.modules.events.sessions.models.sessions.Session attribute\), 183](#)
[acl_event_settings \(indico.core.plugins.IndicoPlugin attribute\), 39](#)
[acl_proxy_class \(indico.modules.events.settings.EventSettings attribute\), 98, 153](#)
[acl_settings \(indico.core.plugins.IndicoPlugin attribute\), 39](#)
[acl_settings \(indico.modules.vc.plugins.VCPluginMixin attribute\), 250](#)
[active_and_answered \(indico.modules.events.surveys.models.surveys.SurveyState attribute\), 190](#)
[active_and_clean \(indico.modules.events.surveys.models.surveys.SurveyState attribute\), 190](#)
[active_fields \(indico.modules.events.registration.models.forms.RegistrationForm attribute\), 165](#)
[active_fields \(indico.modules.events.registration.models.items.RegistrationItem attribute\), 171](#)
[active_registrations \(indico.modules.events.registration.models.forms.RegistrationForm attribute\), 165](#)
[add_abstract_files\(\) \(in module indico.modules.events.abstracts.operations\), 111](#)
[add_attachment_link\(\) \(in module indico.modules.attachments.operations\), 225](#)
[add_edit_log\(\) \(indico.modules.rb.models.reservations.Reservation method\), 236](#)
[add_file_date_column \(indico.modules.events.abstracts.models.files.AbstractFile attribute\), 107](#)
[add_file_date_column \(indico.modules.events.papers.models.files.PaperFile attribute\), 143](#)
[add_file_date_column \(indico.modules.events.papers.models.templates.PaperTemplate attribute\), 149](#)
[add_file_date_column \(indico.modules.events.registration.models.registration_forms.RegistrationForm attribute\), 161](#)
[add_file_date_column \(indico.modules.events.static.models.static.StaticSiteAgreement attribute\), 202](#)
[add_form_fields \(in module indico.core.signals\), 43](#)
[add_principal\(\) \(indico.modules.events.settings.EventACLPLinkPlaceholder method\), 97, 151](#)
[add_survey_question\(\) \(in module indico.modules.events.surveys.operations\), 194](#)
[add_survey_text\(\) \(in module indico.modules.events.surveys.operations\), 194](#)
[additional_info \(indico.modules.events.models.events.Event attribute\), 80](#)
[add_press \(indico.modules.events.models.persons.EventPerson attribute\), 85](#)
[address \(indico.modules.events.models.persons.PersonLinkBase attribute\), 86](#)
[address \(indico.modules.events.registration.models.items.PersonalDataType attribute\), 168](#)
[address \(indico.modules.users.models.users.User attribute\), 212](#)
[adjust_payment_form_data\(\) \(indico.modules.events.payment.plugins.PaymentPluginMixin method\), 156](#)
[adv_registrations \(indico.modules.events.abstracts.placeholders.AbstractURLPlaceholder attribute\), 114](#)
[adv_registrations \(indico.modules.events.abstracts.placeholders.ContributionURLPlaceholder attribute\), 116](#)
[advanced \(indico.modules.events.abstracts.placeholders.SubmitterFirstName attribute\), 114](#)
[advanced \(indico.modules.events.abstracts.placeholders.SubmitterLastName attribute\), 114](#)
[advanced \(indico.modules.events.abstracts.placeholders.TargetSubmitterFirstName attribute\), 115](#)
[advanced \(indico.modules.events.abstracts.placeholders.TargetSubmitterLastName attribute\), 115](#)
[advanced \(indico.modules.events.abstracts.placeholders.TargetSubmitterName attribute\), 115](#)
[advanced \(indico.modules.events.registration.placeholders.registrations.Field attribute\), 174](#)
[affiliation \(indico.modules.events.models.persons.EventPerson attribute\), 85](#)
[affiliation \(indico.modules.events.models.persons.PersonLinkBase attribute\), 86](#)
[affiliation \(indico.modules.events.registration.models.invitations.RegistrationInvitation attribute\), 167](#)
[affiliation \(indico.modules.events.registration.models.items.PersonalDataType attribute\), 168](#)
[affiliation \(indico.modules.events.registration.models.registration_forms.RegistrationForm attribute\), 161](#)
[after_process \(in module indico.core.signals\), 43](#)
[Agreement \(class in indico.modules.events.agreements.models.agreements\), 117](#)
[AgreementLinkPlaceholder \(class in indico.modules.events.agreements.placeholders\), 119](#)
[AgreementState \(class in indico.modules.events.agreements.models.agreements\), 119](#)

118	ALLOWED_CONTENT_TYPE	(in-
all_emails (indico.modules.users.models.users.User attribute), 212	indico.modules.attachments.preview.PDFPreviewer attribute), 226	
all_files (indico.modules.attachments.models.attachments.Attachment attribute), 220	ALLOWED_CONTENT_TYPE	(in-
all_recipients (indico.modules.events.reminders.models.reminders.EventReminder attribute), 177	indico.modules.attachments.preview.Previewer attribute), 226	
allow_access_key (indico.modules.events.models.events.Event attribute), 80	ALLOWED_CONTENT_TYPE	(in-
allow_attachments (indico.modules.events.abstracts.models.abstracts.CallForAbstracts attribute), 103	indico.modules.attachments.preview.TextPreviewer attribute), 226	
allow_comments (indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstracts attribute), 103	allowed_until_payment	(in-
allow_contributors_in_comments (indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstracts attribute), 103	amount (indico.modules.events.payment.models.transactions.PaymentTransaction attribute), 154	
allow_convener_judgment (indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstracts attribute), 103	answer_data (indico.modules.events.surveys.models.submissions.SurveyAnswer attribute), 193	
allow_emails (indico.modules.events.contributions.models.principals.ContributionPrincipal attribute), 125	answers (indico.modules.events.surveys.models.submissions.SurveySubmission attribute), 193	
allow_emails (indico.modules.events.models.principals.EventPrincipal attribute), 87	api_key (indico.modules.users.models.users.User attribute), 212	
allow_emails (indico.modules.events.sessions.models.principals.SessionPrincipal attribute), 186	app_created (in module indico.core.signals), 43	
allow_location_inheritance (indico.modules.events.models.events.Event attribute), 80	application (indico.modules.oauth.models.tokens.OAuthToken attribute), 245	
allow_networks (indico.modules.categories.models.principals.CategoryPrincipal attribute), 207	application_id (indico.modules.oauth.models.tokens.OAuthToken attribute), 245	
allow_networks (indico.modules.events.models.principals.EventPrincipal attribute), 87	approve() (indico.modules.rb.models.blocked_rooms.BlockedRoom method), 232	
allow_no_access_contact (indico.modules.categories.models.categories.Category attribute), 204	as_avatar (indico.modules.users.models.users.User attribute), 212	
allow_no_access_contact (indico.modules.events.models.events.Event attribute), 80	as_legacy (indico.modules.events.models.events.Event attribute), 80	
allow_relationship_preloading (indico.modules.events.contributions.models.contributions.Contribution attribute), 119	as_legacy (indico.modules.groups.core.GroupProxy attribute), 247	
allow_relationship_preloading (indico.modules.events.sessions.models.sessions.Session attribute), 183	as_legacy (indico.modules.users.models.users.User attribute), 212	
allowed (indico.modules.rb.models.blockings.Blocking attribute), 232	as_principal (indico.modules.users.models.users.User attribute), 212	
allowed_always (indico.modules.events.registration.models.forms.MyGroupForm attribute), 164	as_principal (indico.modules.groups.core.GroupProxy attribute), 247	
ALLOWED_CONTENT_TYPE (indico.modules.attachments.preview.ImagePreviewer attribute), 226	as_principal (indico.modules.groups.core.GroupProxy attribute), 247	
ALLOWED_CONTENT_TYPE (indico.modules.attachments.preview.MarkdownPreviewer attribute), 226	Aspect (class in indico.modules.rb.models.aspects), 231	
	aspects (indico.modules.rb.models.locations.Location attribute)	

tribute), 234

assignees (indico.modules.events.papers.models.call_for_papers.AttachmentFolderType (class in indico.modules.attachments.models.attachments), attribute), 141

Attachment (class in indico.modules.attachments.models.attachments), 219

attachment (indico.modules.events.agreements.models.agreements.Attachment (class in indico.modules.attachments.models.attachments), attribute), 117

attachment_access_override (indico.modules.networks.models.networks.IPNetworkGroup (class in indico.modules.networks.models.networks), attribute), 259

attachment_accessed (in module indico.core.signals.attachments), 44

attachment_created (in module indico.core.signals.attachments), 44

attachment_deleted (in module indico.core.signals.attachments), 44

attachment_filename (in indico.modules.events.agreements.models.agreements.Attachment (class in indico.modules.attachments.models.attachments), attribute), 117

ATTACHMENT_FOLDER_ID_COLUMN (in indico.modules.categories.models.categories.Category (class in indico.modules.categories.models.categories), attribute), 204

ATTACHMENT_FOLDER_ID_COLUMN (in indico.modules.events.contributions.models.contributions.Contribution (class in indico.modules.events.contributions.models.contributions), attribute), 119

ATTACHMENT_FOLDER_ID_COLUMN (in indico.modules.events.contributions.models.subcontributions.SubContribution (class in indico.modules.events.contributions.models.subcontributions), attribute), 126

ATTACHMENT_FOLDER_ID_COLUMN (in indico.modules.events.models.events.Event (class in indico.modules.events.models.events), attribute), 80

ATTACHMENT_FOLDER_ID_COLUMN (in indico.modules.events.sessions.models.sessions.Session (class in indico.modules.events.sessions.models.sessions), attribute), 182

attachment_id (indico.modules.attachments.models.attachments.AttachmentFile (class in indico.modules.attachments.models.attachments), attribute), 221

attachment_id (indico.modules.attachments.models.principals.AttachmentPrincipal (class in indico.modules.attachments.models.principals), attribute), 224

attachment_updated (in module indico.core.signals.attachments), 45

AttachmentFile (class in indico.modules.attachments.models.attachments), 221

AttachmentFolder (class in indico.modules.attachments.models.folders), 222

AttachmentFolderPrincipal (class in indico.modules.attachments.models.principals), 224

AttachmentPrincipal (class in indico.modules.attachments.models.principals), 224

attachments (indico.modules.attachments.models.folders.AttachmentFolder (class in indico.modules.attachments.models.folders), attribute), 165

attribute), 222

AttachmentFolderType (class in indico.modules.attachments.models.attachments), 222

attr (indico.modules.events.settings.EventSettingProperty (class in indico.modules.events.settings), attribute), 98, 152

Attachment (class in indico.modules.events.agreements.models.agreements), 117

Attachment (class in indico.modules.rb.models.room_attributes.RoomAttributeAssociation (class in indico.modules.rb.models.room_attributes), attribute), 230

attachment_id (indico.modules.rb.models.room_attributes.RoomAttributeAssociation (class in indico.modules.rb.models.room_attributes), attribute), 230

attributes (indico.modules.rb.models.locations.Location (class in indico.modules.rb.models.locations), attribute), 234

attributes (indico.modules.rb.models.rooms.Room (class in indico.modules.rb.models.rooms), attribute), 227

author_type (indico.modules.events.abstracts.models.persons.AbstractPerson (class in indico.modules.events.abstracts.models.persons), attribute), 107

author_type (indico.modules.events.contributions.models.persons.Contribution (class in indico.modules.events.contributions.models.persons), attribute), 124

author_type (indico.modules.events.contributions.models.persons.SubContribution (class in indico.modules.events.contributions.models.persons), attribute), 125

AuthorsSpeakersMixin (class in indico.modules.events.models.persons), 85

AuthorType (class in indico.modules.events.contributions.models.persons), 85

available_equipment (in indico.modules.rb.models.rooms.Room (class in indico.modules.rb.models.rooms), attribute), 212

avatar_css (indico.modules.users.models.users.User (class in indico.modules.users.models.users), attribute), 212

awaiting (indico.modules.events.abstracts.models.abstracts.AbstractPublicSession (class in indico.modules.events.abstracts.models.abstracts), attribute), 102

B

background_color (indico.modules.events.sessions.models.sessions.Session (class in indico.modules.events.sessions.models.sessions), attribute), 183

background_color (indico.modules.events.timetable.models.breaks.Break (class in indico.modules.events.timetable.models.breaks), attribute), 105

background_image (in indico.modules.designer.models.templates.DesignerTemplate (class in indico.modules.designer.models.templates), attribute), 253

background_image_id (in indico.modules.designer.models.templates.DesignerTemplate (class in indico.modules.designer.models.templates), attribute), 253

background_position (in indico.modules.designer.pdf.TplData (class in indico.modules.designer.pdf), attribute), 254

backside_template (indico.modules.designer.models.templates.DesignerTemplate (class in indico.modules.designer.models.templates), attribute), 253

backside_template_id (in indico.modules.designer.models.templates.DesignerTemplate (class in indico.modules.designer.models.templates), attribute), 253

base_price (indico.modules.events.registration.models.forms.RegistrationForm (class in indico.modules.events.registration.models.forms), attribute), 165

base_price (indico.modules.events.registration.models.registrations.Registration attribute), 158

belongs_to() (indico.modules.events.agreements.models.agreements.Agreement method), 117

best_unit (indico.web.forms.fields.TimeDeltaField attribute), 272

billable_data (indico.modules.events.registration.models.registrations.Registration attribute), 158

block (indico.modules.vc.models.vc_rooms.VCRoomLinkType attribute), 250

blocked_rooms (indico.modules.rb.models.blockings.Blocking attribute), 232

blocked_rooms (indico.modules.rb.models.rooms.Room attribute), 227

BlockedRoom (class in indico.modules.rb.models.blocked_rooms), 232

BlockedRoomState (class in indico.modules.rb.models.blocked_rooms), 233

Blocking (class in indico.modules.rb.models.blockings), 232

blocking_id (indico.modules.rb.models.blocked_rooms.BlockedRoom attribute), 232

blocking_id (indico.modules.rb.models.blocking_principals.BlockingPrincipal attribute), 233

BlockingPrincipal (class in indico.modules.rb.models.blocking_principals), 233

blocks (indico.modules.events.sessions.models.sessions.Session attribute), 183

BOACorrespondingAuthorType (class in indico.modules.events.abstracts.settings), 116

board_number (indico.modules.events.contributions.models.contributions.Contribution attribute), 119

BOASortField (class in indico.modules.events.abstracts.settings), 116

body (indico.modules.events.abstracts.models.email_logs.AbstractEmailLogEntry attribute), 105

body (indico.modules.events.abstracts.models.email_templates.AbstractEmailTemplate attribute), 105

bookable_hours (indico.modules.rb.models.rooms.Room attribute), 227

BookableHours (class in indico.modules.rb.models.room_bookable_hours), 231

booked_for_id (indico.modules.rb.models.reservations.Reservation attribute), 236

booked_for_name (indico.modules.rb.models.reservations.Reservation attribute), 236

booked_for_user (indico.modules.rb.models.reservations.Reservation attribute), 236

booking_limit_days (indico.modules.rb.models.rooms.Room attribute), 227

booking_reason (indico.modules.rb.models.reservations.Reservation attribute), 236

booking_url (indico.modules.rb.models.rooms.Room attribute), 227

BookingPermission (class in indico.modules.rb.services.rooms), 240

both (indico.modules.events.abstracts.models.abstracts.EditTrackMode attribute), 103

bottom_right_latitude (indico.modules.rb.models.aspects.Aspect attribute), 231

bottom_right_longitude (indico.modules.rb.models.aspects.Aspect attribute), 231

Break (class in indico.modules.events.timetable.models.breaks), 195

BREAK (indico.modules.events.timetable.models.entries.TimetableEntryType attribute), 198

break_ (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 196

break_id (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 196

build_default_email_template() (in module indico.modules.events.abstracts.util), 112

build_menu_entry_name() (in module indico.modules.events.layout.util), 134

build_note_api_data() (in module indico.modules.events.notes.util), 140

build_note_legacy_api_data() (in module indico.modules.events.notes.util), 140

build_registration_api_data() (in module indico.modules.events.registration.util), 172

build_registration_contribution_data() (in module indico.modules.events.registration.util), 172

building (indico.modules.rb.models.rooms.Room attribute), 227

C

[call_for_proposals_attr](#) (in- [can_be_modified\(\)](#) (indico.modules.rb.models.rooms.Room method), 227
 dico.modules.events.papers.models.papers.Paper attribute), 144 [can_be_modified\(\)](#) (indico.modules.users.models.users.User method), 212
[CallForAbstracts](#) (class in in- [can_be_overridden\(\)](#) (in-
 dico.modules.events.abstracts.models.call_for_abstracts), 103 [dico.modules.rb.models.blockings.Blocking](#)
[CallForPapers](#) (class in in- [method\)](#), 232
 dico.modules.events.papers.models.call_for_papers) [can_be_overridden\(\)](#) (in-
 141 [dico.modules.rb.models.rooms.Room](#) method),
[can_access](#) (in module indico.core.signals.acl), 43 [227](#)
[can_access\(\)](#) (indico.modules.attachments.models.attachments.Attachment) [can_be_booked\(\)](#) (in-
 method), 220 [dico.modules.rb.models.rooms.Room](#) method),
[can_access\(\)](#) (indico.modules.attachments.models.folders.AttachmentFolder [can_be_rejected\(\)](#) (indico.modules.rb.models.reservations.Reservation
 method), 222 [method\)](#), 236
[can_access\(\)](#) (indico.modules.events.abstracts.models.abstracts.Abstract [method\)](#), 100
[can_access\(\)](#) (indico.modules.events.contributions.models.subcontributions.SubContribution [can_comment\(\)](#) (indico.modules.events.abstracts.models.abstracts.Abstract
 method), 126 [method\)](#), 90
[can_access\(\)](#) (indico.modules.events.sessions.models.blocks.SessionBlock [can_comment\(\)](#) (indico.modules.events.papers.models.papers.Paper
 method), 184 [method\)](#), 144
[can_access\(\)](#) (indico.modules.events.timetable.models.breaks.Break [can_convene\(\)](#) (indico.modules.events.abstracts.models.abstracts.Abstract
 method), 195 [method\)](#), 100
[can_access_judging_area\(\)](#) (in- [method\)](#), 100
 dico.modules.events.papers.models.call_for_papers) [CallForPapers](#) (indico.modules.events.tracks.models.tracks.Track
 method), 141 [method\)](#), 201
[can_access_reviewing_area\(\)](#) (in- [can_create_events\(\)](#) (in-
 dico.modules.events.papers.models.call_for_papers.CallForPapers) [indico.modules.categories.models.categories.Category](#)
 method), 141 [method\)](#), 204
[can_be_accepted\(\)](#) (indico.modules.rb.models.reservations.Reservation) [can_delete\(\)](#) (indico.modules.events.tracks.models.tracks.Track
 method), 236 [method\)](#), 201
[can_be_booked\(\)](#) (indico.modules.rb.models.rooms.Room [can_edit\(\)](#) (indico.modules.events.abstracts.models.abstracts.Abstract
 method), 227 [method\)](#), 100
[can_be_cancelled\(\)](#) (in- [can_edit\(\)](#) (indico.modules.events.abstracts.models.comments.AbstractCom
 dico.modules.rb.models.reservations.Reservation [method\)](#), 104
 method), 236 [can_edit\(\)](#) (indico.modules.events.abstracts.models.reviews.AbstractReview
[can_be_deleted\(\)](#) (indico.modules.rb.models.blockings.Blocking [method\)](#), 109
 method), 232 [can_edit\(\)](#) (indico.modules.events.models.reviews.ProposalCommentMixin
[can_be_deleted\(\)](#) (indico.modules.rb.models.reservations.Reservation [method\)](#), 89
 method), 236 [can_edit\(\)](#) (indico.modules.events.models.reviews.ProposalReviewMixin
[can_be_deleted\(\)](#) (indico.modules.rb.models.rooms.Room [method\)](#), 91
 method), 227 [can_edit\(\)](#) (indico.modules.events.papers.models.comments.PaperReviewCo
[can_be_managed\(\)](#) (in- [method\)](#), 142
 dico.modules.events.requests.base.RequestDefinition) [can_base\(\)](#) (indico.modules.events.papers.models.reviews.PaperReview
 class method), 180 [method\)](#), 146
[can_be_modified](#) (indico.modules.events.registration.models.registration.Registration [can_be_booked\(\)](#) (in-
 attribute), 158 [dico.modules.events.abstracts.models.call_for_abstracts.CallForA](#)
[can_be_modified](#) (indico.modules.events.requests.models.requests.Request [method\)](#), 103
 attribute), 179 [can_edit_note\(\)](#) (in module in-
[can_be_modified\(\)](#) (indico.modules.events.payment.plugins.PaymentPluginMixin) [indico.modules.events.notes.util\)](#), 140
 method), 156 [can_edit_note\(\)](#) (indico.modules.events.sessions.models.blocks.SessionBloc
[can_be_modified\(\)](#) (indico.modules.rb.models.blockings.Blocking [method\)](#), 184
 method), 232 [can_judge\(\)](#) (indico.modules.events.abstracts.models.abstracts.Abstract
[can_be_modified\(\)](#) (indico.modules.rb.models.reservations.Reservation [method\)](#), 100
 method), 236 [can_judge\(\)](#) (indico.modules.events.papers.models.papers.Paper

method), 144

can_lock() (indico.modules.events.models.events.Event method), 80

can_manage (in module indico.core.signals.acl), 44

can_manage() (indico.modules.events.contributions.models.contributions.Contribution method), 119

can_manage() (indico.modules.events.contributions.models.subcontributions.SubContribution method), 126

can_manage() (indico.modules.events.papers.models.papers.Paper method), 144

can_manage() (indico.modules.events.sessions.models.blocks.SessionBlock method), 184

can_manage_attachments() (in module indico.modules.attachments.util), 225

can_manage_attachments() (in indico.modules.events.sessions.models.blocks.SessionBlock method), 184

can_manage_blocks() (in indico.modules.events.sessions.models.sessions.Session method), 183

can_manage_contributions() (in indico.modules.events.sessions.models.sessions.Session method), 183

can_manage_sessions() (in module indico.modules.events.sessions.util), 187

can_manage_vc() (indico.modules.vc.plugins.VCPluginMixin method), 250

can_manage_vc_room() (in indico.modules.vc.plugins.VCPluginMixin method), 250

can_manage_vc_rooms() (in indico.modules.vc.plugins.VCPluginMixin method), 251

CAN_POPULATE (in indico.modules.events.abstracts.fields.EmailRuleListField attribute), 263

CAN_POPULATE (in indico.modules.events.abstracts.fields.TrackRoleField attribute), 264

CAN_POPULATE (in indico.modules.events.papers.fields.PaperEmailSettingsField attribute), 266

CAN_POPULATE (in indico.web.forms.fields.IndicoLocationField attribute), 275

CAN_POPULATE (in indico.web.forms.fields.IndicoPalettePickerField attribute), 271

CAN_POPULATE (indico.web.forms.fields.JSONField attribute), 268

CAN_POPULATE (in indico.web.forms.fields.OccurrencesField attribute), 272

can_preview() (indico.modules.attachments.preview.PDFPreview class method), 226

can_preview() (indico.modules.attachments.preview.Previewer class method), 226

can_review() (indico.modules.events.abstracts.models.abstracts.AbstractContribution method), 100

can_review() (indico.modules.events.models.reviews.ProposalMixin method), 144

can_review() (indico.modules.events.papers.models.papers.Paper method), 144

can_review_abstracts() (in indico.modules.events.tracks.models.tracks.Track method), 201

can_see_reviews() (indico.modules.events.abstracts.models.abstracts.AbstractContribution method), 100

can_submit() (indico.modules.events.papers.models.papers.Paper method), 144

can_submit() (indico.modules.events.registration.models.forms.RegistrationForm method), 165

can_submit() (indico.modules.events.surveys.models.surveys.Survey method), 188

can_submit_abstracts() (in indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstracts method), 103

can_swap_entry() (in module indico.modules.events.timetable.operations), 198

can_view() (indico.modules.attachments.models.folders.AttachmentFolder method), 222

can_view() (indico.modules.events.abstracts.models.comments.AbstractComment method), 104

can_view() (indico.modules.events.abstracts.models.reviews.AbstractReview method), 109

can_view() (indico.modules.events.papers.models.comments.PaperReviewComment method), 142

can_view() (indico.modules.events.papers.models.reviews.PaperReview method), 146

can_view() (indico.modules.events.timetable.models.entries.TimetableEntry method), 196

can_withdraw() (indico.modules.events.abstracts.models.abstracts.AbstractContribution method), 100

cancel() (indico.modules.events.payment.models.transactions.TransactionAction attribute), 155

cancel() (indico.modules.rb.models.reservation_occurrences.ReservationOccurrence method), 239

cancel() (indico.modules.rb.models.reservations.Reservation method), 236

cancelled (indico.modules.events.payment.models.transactions.TransactionAction attribute), 155

candidate_contrib_types (in indico.modules.events.abstracts.models.abstracts.AbstractContribution attribute), 100

candidate_tracks (indico.modules.events.abstracts.models.abstracts.AbstractContribution attribute), 100

capacity (indico.modules.rb.models.rooms.Room attribute), 201

- tribute), 227
- Category (class in indico.modules.categories.models.categories), 204
- category (indico.core.plugins.IndicoPlugin attribute), 39
- category (indico.modules.attachments.models.folders.AttachmentFolder attribute), 222
- category (indico.modules.categories.models.settings.CategorySetting attribute), 208
- category (indico.modules.designer.models.templates.DesignerTemplate attribute), 253
- category (indico.modules.events.models.events.Event attribute), 80
- category (indico.modules.events.notes.models.notes.EventNote attribute), 138
- category (indico.modules.events.payment.plugins.PaymentPluginMixin attribute), 156
- category (indico.modules.users.models.suggestions.SuggestedCategory attribute), 216
- category (indico.modules.vc.plugins.VCPluginMixin attribute), 251
- category_chain_overlaps() (indico.modules.events.models.events.Event class method), 80
- category_id (indico.modules.attachments.models.folders.AttachmentFolder attribute), 222
- category_id (indico.modules.categories.models.principals.CategoryPrincipal attribute), 207
- category_id (indico.modules.categories.models.settings.CategorySetting attribute), 208
- category_id (indico.modules.designer.models.templates.DesignerTemplate attribute), 253
- category_id (indico.modules.events.models.events.Event attribute), 80
- category_id (indico.modules.events.notes.models.notes.EventNote attribute), 138
- category_id (indico.modules.users.models.suggestions.SuggestedCategory attribute), 216
- CategoryField (class in indico.modules.categories.fields), 266
- CategoryPrincipal (class in indico.modules.categories.models.principals), 207
- CategorySetting (class in indico.modules.categories.models.settings), 208
- CategorySettingsProxy (class in indico.modules.categories.settings), 210
- CategoryTitlePlaceholder (class in indico.modules.designer.placeholders), 258
- Cell (class in indico.modules.events.registration.stats), 176
- center_latitude (indico.modules.rb.models.aspects.Aspect attribute), 231
- center_longitude (indico.modules.rb.models.aspects.Aspect attribute), 231
- cfa (indico.modules.events.models.events.Event attribute), 80
- cfp (indico.modules.events.models.events.Event attribute), 80
- cfp (indico.modules.events.models.reviews.ProposalMixin attribute), 90
- chain_query (indico.modules.categories.models.categories.Category attribute), 204
- change (indico.modules.events.logs.models.entries.EventLogKind attribute), 136
- change_tracks (indico.modules.events.abstracts.models.reviews.AbstractAc attribute), 109
- check_advance_days() (indico.modules.rb.models.rooms.Room method), 227
- check_category_kable_hours() (indico.modules.rb.models.rooms.Room method), 227
- check_registration_email() (in module indico.modules.events.registration.util), 172
- checked_in (indico.modules.events.registration.models.registrations.Registr attribute), 158
- checked_in (indico.modules.events.registration.models.registrations.Registr attribute), 158
- checked_in (indico.modules.oauth.models.applications.SystemAppType attribute), 244
- children (indico.modules.categories.models.categories.Category attribute), 204
- children (indico.modules.events.layout.models.menu.MenuEntry attribute), 132
- children (indico.modules.events.registration.models.form_fields.Registration attribute), 163
- children (indico.modules.events.registration.models.form_fields.Registration attribute), 164
- children (indico.modules.events.registration.models.items.RegistrationForm attribute), 169
- children (indico.modules.events.registration.models.items.RegistrationForm attribute), 170
- children (indico.modules.events.registration.models.items.RegistrationForm attribute), 171
- children (indico.modules.events.registration.models.items.RegistrationForm attribute), 171
- children (indico.modules.events.surveys.models.items.SurveySection attribute), 192
- children (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 196
- children (indico.modules.rb.models.equipment.EquipmentType attribute), 234
- children (indico.modules.rb.models.room_attributes.RoomAttribute attribute), 230
- choices (indico.web.forms.fields.RelativeDeltaField attribute), 276

- choices (indico.web.forms.fields.TimeDeltaField attribute), 272
- clear_boa_cache() (in module indico.modules.events.abstracts.util), 112
- cli (in module indico.core.signals.plugin), 48
- client_id (indico.modules.oauth.models.applications.OAuthApplication attribute), 244
- client_secret (indico.modules.oauth.models.applications.OAuthApplication attribute), 244
- client_type (indico.modules.oauth.models.applications.OAuthApplication attribute), 244
- clone_event() (in module indico.modules.events.operations), 94
- cloned (in module indico.core.signals.event), 45
- cloned_from (indico.modules.events.models.events.Event attribute), 80
- cloned_from_id (indico.modules.events.models.events.Event attribute), 80
- close() (indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstracts attribute), 103
- close() (indico.modules.events.papers.models.call_for_papers.CallForPapers attribute), 141
- close() (indico.modules.events.surveys.models.surveys.Survey attribute), 188
- close_cfa() (in module indico.modules.events.abstracts.operations), 111
- close_cfp() (in module indico.modules.events.papers.operations), 150
- CoAuthorsPlaceholder (class in indico.modules.events.abstracts.placeholders), 114
- code (indico.modules.events.sessions.models.sessions.Session attribute), 183
- code (indico.modules.events.tracks.models.tracks.Track attribute), 201
- column (indico.modules.events.registration.models.items.PersonalData attribute), 168
- comment (indico.modules.auth.models.registration_requests.RegistrationRequest attribute), 242
- comment (indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 110
- comment (indico.modules.events.papers.models.reviews.PaperReview attribute), 147
- comment (indico.modules.events.requests.models.requests.Request attribute), 179
- comments (indico.modules.rb.models.rooms.Room attribute), 227
- competences (indico.modules.events.papers.models.competences.PaperCompetence attribute), 143
- complete (indico.modules.events.payment.models.transactions.Transaction attribute), 155
- complete (indico.modules.events.registration.models.registrations.Registration attribute), 162
- compose_rooms_stats() (in module indico.modules.rb.statistics), 240
- condition_choices (indico.modules.events.abstracts.fields.EmailRuleListField attribute), 263
- condition_class_map (indico.modules.events.abstracts.fields.EmailRuleListField attribute), 264
- conflict (indico.modules.events.models.events.EventType attribute), 84
- conflict_applicable (indico.core.plugins.IndicoPlugin attribute), 39
- conflicting (indico.modules.events.abstracts.models.abstracts.AbstractReview attribute), 102
- ConflictingOccurrences, 235
- contact_email (indico.modules.rb.models.reservations.Reservation attribute), 236
- contact_emails (indico.modules.events.models.events.Event attribute), 80
- contact_id (indico.modules.events.registration.models.forms.Registration attribute), 165
- contact_phone (indico.modules.rb.models.reservations.Reservation attribute), 236
- contact_phones (indico.modules.events.models.events.Event attribute), 80
- contact_title (indico.modules.events.models.events.Event attribute), 80
- contains_ip() (indico.modules.networks.models.networks.IPNetworkGroup method), 259
- contains_user() (indico.modules.events.settings.EventACLProxy method), 97, 152
- content (indico.modules.events.papers.models.reviews.PaperReviewType attribute), 147
- content (indico.modules.news.models.news.NewsItem attribute), 260
- content_review_questions (indico.modules.events.papers.models.call_for_papers.CallForPapers attribute), 141
- content_reviewer_deadline (indico.modules.events.papers.models.call_for_papers.CallForPapers attribute), 141
- content_reviewers (indico.modules.events.papers.models.call_for_papers.CallForPapers attribute), 141
- content_reviewing_enabled (indico.modules.events.papers.models.call_for_papers.CallForPapers attribute), 141
- content_type (indico.modules.attachments.models.attachments.Attachment attribute), 221
- content_type (indico.modules.designer.models.images.DesignerImageFile attribute), 162
- content_type (indico.modules.events.abstracts.models.files.AbstractFile attribute), 107
- content_type (indico.modules.events.layout.models.images.ImageFile attribute), 162
- content_type (indico.modules.events.papers.models.files.PaperFile attribute), 162

- attribute), 143
- content_type (indico.modules.events.papers.models.templates.PaperTemplate attribute), 149
- content_type (indico.modules.events.registration.models.registrations.RegistrationData attribute), 161
- content_type (indico.modules.events.static.models.static.StaticSite attribute), 203
- Contribution (class in in-dico.modules.events.contributions.models.contributions), 119
- contribution (indico.modules.attachments.models.folders.AttachmentsFolder attribute), 222
- contribution (indico.modules.events.notes.models.notes.EventNote attribute), 138
- contribution (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 196
- CONTRIBUTION (indico.modules.events.timetable.models.entries.TimetableEntryType attribute), 198
- contribution (indico.modules.vc.models.vc_rooms.VCRoomContribution attribute), 250
- contribution_count (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 184
- contribution_created (in module in-dico.core.signals.event), 45
- contribution_deleted (in module in-dico.core.signals.event), 45
- contribution_field (indico.modules.events.abstracts.models.fields.AbstractField attribute), 107
- contribution_field (indico.modules.events.contributions.models.fields.ContributionField attribute), 123
- contribution_field (indico.modules.events.contributions.models.fields.ContributionPersonLink attribute), 123
- contribution_field_backref_name (in-dico.modules.events.abstracts.models.fields.AbstractField attribute), 107
- contribution_field_backref_name (in-dico.modules.events.contributions.models.fields.ContributionField attribute), 123
- contribution_field_backref_name (in-dico.modules.events.contributions.models.fields.ContributionField attribute), 123
- contribution_field_id (in-dico.modules.events.abstracts.models.fields.AbstractField attribute), 107
- contribution_field_id (in-dico.modules.events.contributions.models.fields.ContributionField attribute), 123
- contribution_field_id (in-dico.modules.events.contributions.models.fields.ContributionField attribute), 123
- contribution_id (indico.modules.attachments.models.folders.AttachmentsFolder attribute), 222
- contribution_id (indico.modules.events.contributions.models.fields.ContributionField attribute), 123
- contribution_id (indico.modules.events.contributions.models.persons.Contribution attribute), 124
- contribution_id (indico.modules.events.contributions.models.principals.Contribution attribute), 126
- contribution_id (indico.modules.events.contributions.models.references.Contribution attribute), 126
- contribution_id (indico.modules.events.contributions.models.subcontributions.Contribution attribute), 126
- contribution_id (indico.modules.events.notes.models.notes.EventNote attribute), 138
- contribution_id (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 196
- contribution_id (indico.modules.vc.models.vc_rooms.VCRoomEventAssociation attribute), 249
- contribution_type_row() (in module in-dico.modules.events.contributions.util), 129
- ContributionEntryType (in module in-dico.core.signals.event), 46
- ContributionField (class in in-dico.modules.events.contributions.models.fields), 123
- ContributionFieldValue (class in in-dico.modules.events.contributions.models.fields), 123
- ContributionFieldValueBase (class in in-dico.modules.events.contributions.models.fields), 265
- ContributionPrincipal (class in in-dico.modules.events.contributions.models.principals), 125
- ContributionReference (class in in-dico.modules.events.contributions.models.references), 126
- ContributionTypeValueBase (class in in-dico.modules.events.contributions.models.types), 127
- ContributionTypePlaceholder (class in in-dico.modules.events.abstracts.placeholders), 116
- ContributionTypePlaceholder (class in in-dico.modules.events.abstracts.placeholders), 116
- ContributionTypeValueBase (class in in-dico.modules.events.abstracts.models.reviews.AbstractComment attribute), 109
- ContributorsFolder (indico.modules.events.papers.models.reviews.PaperComments attribute), 146
- contributors_group (indico.modules.events.abstracts.models.reviews.AbstractComment attribute), 109

conveners (indico.modules.events.sessions.models.sessions.SessionForm() (indico.modules.vc.plugins.VCPluginMixin attribute), 183
 method), 251
 conveners (indico.modules.events.tracks.models.tracks.Trackcreate_from_data() (in- (in-
 attribute), 201 dico.modules.events.agreements.models.agreements.Agreement
 convert_legacy_repeatability() (in- static method), 117
 dico.modules.rb.models.reservations.RepeatMappinggate_from_data() (in-
 class method), 236 dico.modules.rb.models.reservations.Reservation
 country (indico.modules.events.registration.models.items.PersonalDataTypes() (in- static method), 236
 attribute), 168 create_from_email() (in-
 create() (indico.modules.events.models.static_list_links.StaticListLinkdico.modules.events.abstracts.models.email_logs.AbstractEmailL
 class method), 93 class method), 105
 create_abstract() (in module in- create_from_user() (in- (in-
 dico.modules.events.abstracts.operations), dico.modules.events.models.persons.EventPerson
 111 class method), 85
 create_abstract_comment() (in module in- create_judgment_endpoint (in- (in-
 dico.modules.events.abstracts.operations), dico.modules.events.abstracts.models.abstracts.Abstract
 111 attribute), 100
 create_abstract_review() (in module in- create_judgment_endpoint (in- (in-
 dico.modules.events.abstracts.operations), dico.modules.events.models.reviews.ProposalMixin
 111 attribute), 90
 create_boa() (in module in- create_judgment_endpoint (in- (in-
 dico.modules.events.abstracts.util), 112 dico.modules.events.papers.models.papers.Paper
 create_break_entry() (in module in- attribute), 144
 dico.modules.events.timetable.operations), create_manager_form() (in-
 198 dico.modules.events.requests.base.RequestDefinitionBase
 create_category() (in module in- class method), 181
 dico.modules.categories.operations), 208 create_mock_abstract() (in module in-
 create_comment() (in module in- dico.modules.events.abstracts.util), 112
 dico.modules.events.papers.operations), 150 create_next() (indico.modules.events.payment.models.transactions.Payment
 create_comment_endpoint (in- class method), 154
 dico.modules.events.abstracts.models.abstracts.AbstractOccurrences() (in-
 attribute), 100 dico.modules.rb.models.reservations.Reservation
 create_comment_endpoint (in- method), 237
 dico.modules.events.models.reviews.ProposalMixincreate_paper_revision() (in module in-
 attribute), 90 dico.modules.events.papers.operations), 150
 create_comment_endpoint (in- create_paper_template() (in module in-
 dico.modules.events.papers.models.papers.Paper dico.modules.events.papers.operations), 150
 attribute), 144 create_personal_data_fields() (in module in-
 create_competences() (in module in- dico.modules.events.registration.util), 172
 dico.modules.events.papers.operations), 150 create_reference_type() (in module in-
 create_contribution() (in module in- dico.modules.events.operations), 94
 dico.modules.events.contributions.operations), create_registration() (in module in-
 128 dico.modules.events.registration.util), 172
 create_contribution_from_abstract() (in module in- create_review() (in module in-
 dico.modules.events.contributions.operations), dico.modules.events.papers.operations), 150
 128 create_review_endpoint (in-
 create_event() (in module in- dico.modules.events.abstracts.models.abstracts.Abstract
 dico.modules.events.operations), 94 attribute), 100
 create_event_logo_tmp_file() (in module in- create_review_endpoint (in- (in-
 dico.modules.events.util), 95 dico.modules.events.models.reviews.ProposalMixin
 create_event_references() (in module in- attribute), 90
 dico.modules.events.operations), 94 create_review_endpoint (in- (in-
 create_form() (indico.modules.events.requests.base.RequestDefinitionBase modules.events.papers.models.papers.Paper
 class method), 180 attribute), 144

- [create_revision\(\) \(indico.modules.events.notes.models.notes.EventNoteRevision attribute\), 138](#)
[create_room\(\) \(indico.modules.vc.plugins.VCPluginMixin class method\), 251](#)
[create_series\(\) \(indico.modules.rb.models.reservation_occurrences.ReservationOccurrence class method\), 239](#)
[create_series_for_reservation\(\) \(indico.modules.rb.models.reservation_occurrences.ReservationOccurrence class method\), 239](#)
[create_session\(\) \(in module indico.modules.events.sessions.operations\), 187](#)
[create_session_block\(\) \(in module indico.modules.events.sessions.operations\), 187](#)
[create_session_block_entry\(\) \(in module indico.modules.events.timetable.operations\), 198](#)
[create_subcontribution\(\) \(in module indico.modules.events.contributions.operations\), 128](#)
[create_timetable_entry\(\) \(in module indico.modules.events.timetable.operations\), 198](#)
[create_track\(\) \(in module indico.modules.events.tracks.operations\), 202](#)
[create_untrusted_persons \(in module indico.modules.events.abstracts.fields.AbstractPersonLinkListField attribute\), 262](#)
[create_untrusted_persons \(in module indico.modules.events.fields.EventPersonListField attribute\), 261](#)
[create_user\(\) \(in module indico.modules.users.operations\), 217](#)
[created \(in module indico.core.signals.category\), 45](#)
[created \(in module indico.core.signals.event\), 46](#)
[created \(indico.modules.vc.models.vc_rooms.VCRoomStatus attribute\), 250](#)
[created_by_id \(indico.modules.events.requests.models.requests.Request attribute\), 179](#)
[created_by_id \(indico.modules.rb.models.blockings.Blocking attribute\), 232](#)
[created_by_id \(indico.modules.rb.models.reservations.Reservation attribute\), 237](#)
[created_by_id \(indico.modules.vc.models.vc_rooms.VCRoom attribute\), 248](#)
[created_by_id \(indico.modules.vc.models.vc_rooms.VCRoom attribute\), 248](#)
[created_by_user \(indico.modules.events.requests.models.requests.Request attribute\), 179](#)
[created_by_user \(indico.modules.rb.models.blockings.Blocking attribute\), 232](#)
[created_by_user \(indico.modules.rb.models.reservations.Reservation attribute\), 237](#)
[created_by_user \(indico.modules.vc.models.vc_rooms.VCRoom attribute\), 248](#)
[created_dt \(indico.modules.attachments.models.attachments.AttachmentFile attribute\), 221](#)
[created_dt \(indico.modules.designer.models.images.DesignerImageFile attribute\), 253](#)
[created_dt \(indico.modules.events.abstracts.models.comments.AbstractComment attribute\), 104](#)
[created_dt \(indico.modules.events.abstracts.models.files.AbstractFile attribute\), 107](#)
[created_dt \(indico.modules.events.abstracts.models.reviews.AbstractReview attribute\), 110](#)
[created_dt \(indico.modules.events.layout.models.images.ImageFile attribute\), 131](#)
[created_dt \(indico.modules.events.models.events.Event attribute\), 80](#)
[created_dt \(indico.modules.events.models.static_list_links.StaticListLink attribute\), 93](#)
[created_dt \(indico.modules.events.notes.models.notes.EventNoteRevision attribute\), 140](#)
[created_dt \(indico.modules.events.papers.models.comments.PaperReviewComment attribute\), 142](#)
[created_dt \(indico.modules.events.papers.models.files.PaperFile attribute\), 143](#)
[created_dt \(indico.modules.events.papers.models.reviews.PaperJudgmentPr attribute\), 146](#)
[created_dt \(indico.modules.events.papers.models.reviews.PaperReview attribute\), 147](#)
[created_dt \(indico.modules.events.papers.models.templates.PaperTemplate attribute\), 149](#)
[created_dt \(indico.modules.events.registration.models.registrations.Registration attribute\), 161](#)
[created_dt \(indico.modules.events.reminders.models.reminders.EventReminder attribute\), 178](#)
[created_dt \(indico.modules.events.requests.models.requests.Request attribute\), 179](#)
[created_dt \(indico.modules.events.static.models.static.StaticSite attribute\), 203](#)
[created_dt \(indico.modules.news.models.news.NewsItem attribute\), 260](#)
[created_dt \(indico.modules.rb.models.blockings.Blocking attribute\), 232](#)
[created_dt \(indico.modules.rb.models.reservations.Reservation attribute\), 237](#)
[created_dt \(indico.modules.vc.models.vc_rooms.VCRoom attribute\), 248](#)
[created_events \(indico.modules.users.models.users.User attribute\), 212](#)
[created_index \(indico.modules.events.models.events.Event attribute\), 80](#)
[created_index \(indico.modules.events.reminders.models.reminders.EventReminder attribute\), 178](#)
[created_index \(indico.modules.events.static.models.static.StaticSite attribute\), 203](#)
[created_index \(indico.modules.events.models.events.Event attribute\), 80](#)

creator_id (indico.modules.events.reminders.models.reminders.EventReminder attribute), 178

creator_id (indico.modules.events.static.models.static.StaticSite attribute), 203

currency (indico.modules.events.payment.models.transaction.PaymentTransaction attribute), 154

currency (indico.modules.events.registration.models.forms.RegistrationForm attribute), 165

currency (indico.modules.events.registration.models.registration_data.RegistrationData attribute), 159

current_data (indico.modules.events.registration.models.form_fields.RegistrationFormFields attribute), 163

current_data (indico.modules.events.registration.models.form_fields.RegistrationFormFields attribute), 164

current_data (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 169

current_data (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 170

current_data (indico.modules.events.registration.models.items.RegistrationFormPersons attribute), 171

current_data (indico.modules.events.registration.models.items.RegistrationFormSections attribute), 171

current_data (indico.modules.events.registration.models.items.RegistrationFormText attribute), 172

current_data_id (indico.modules.events.registration.models.items.RegistrationFormText attribute), 163

current_data_id (indico.modules.events.registration.models.items.RegistrationFormText attribute), 164

current_data_id (indico.modules.events.registration.models.items.RegistrationFormText attribute), 169

current_data_id (indico.modules.events.registration.models.items.RegistrationFormText attribute), 170

current_data_id (indico.modules.events.registration.models.items.RegistrationFormText attribute), 171

current_data_id (indico.modules.events.registration.models.items.RegistrationFormText attribute), 172

current_data_id (indico.modules.events.registration.models.items.RegistrationFormText attribute), 172

current_data_id (indico.modules.events.registration.models.items.RegistrationFormText attribute), 179

current_data_id (indico.modules.events.registration.models.items.RegistrationFormText attribute), 193

current_data_id (indico.modules.events.registration.models.items.RegistrationFormText attribute), 235

current_revision (indico.modules.events.notes.models.notes.EventNote attribute), 138

current_revision_id (indico.modules.events.notes.models.notes.EventNote attribute), 138

CustomFieldsMixin (class in indico.modules.events.contributions.models.contributions_by_field attribute), 122

D

danger (indico.modules.categories.models.categories.EventMessageMode attribute), 207

data (indico.modules.auth.models.identities.Identity attribute), 241

data (indico.modules.designer.models.templates.DesignerTemplate attribute), 253

data (indico.modules.events.abstracts.models.email_logs.AbstractEmail attribute), 105

data (indico.modules.events.abstracts.models.fields.AbstractField attribute), 107

data (indico.modules.events.agreements.models.agreements.Agreement attribute), 117

data (indico.modules.events.contributions.models.fields.ContributionField attribute), 123

data (indico.modules.events.contributions.models.fields.ContributionField attribute), 123

data (indico.modules.events.logs.models.entries.EventLogEntry attribute), 135

data (indico.modules.events.models.static_list_links.StaticListLink attribute), 93

data (indico.modules.events.payment.models.transactions.PaymentTransaction attribute), 154

data (indico.modules.events.registration.models.registration_data.RegistrationData attribute), 163

data (indico.modules.events.registration.models.registration_data.RegistrationData attribute), 163

data (indico.modules.events.registration.models.registration_data.RegistrationData attribute), 164

data (indico.modules.events.registration.models.registration_data.RegistrationData attribute), 169

data (indico.modules.events.registration.models.registration_data.RegistrationData attribute), 169

data (indico.modules.events.registration.models.registration_data.RegistrationData attribute), 170

data (indico.modules.events.registration.models.registration_data.RegistrationData attribute), 171

data (indico.modules.events.registration.models.registration_data.RegistrationData attribute), 171

data (indico.modules.events.registration.models.registration_data.RegistrationData attribute), 172

data (indico.modules.events.registration.models.registration_data.RegistrationData attribute), 172

data (indico.modules.events.registration.models.registration_data.RegistrationData attribute), 179

data (indico.modules.events.registration.models.registration_data.RegistrationData attribute), 193

data (indico.modules.events.registration.models.registration_data.RegistrationData attribute), 235

data (indico.modules.events.registration.models.registration_data.RegistrationData attribute), 248

data (indico.modules.events.registration.models.registration_data.RegistrationData attribute), 249

data (indico.web.forms.fields.IndicoQuerySelectMultipleField attribute), 275

data_by_field (indico.modules.events.abstracts.models.abstracts.Abstract attribute), 100

data_by_field (indico.modules.events.registration.models.registrations.Registration attribute), 159

data_versions (indico.modules.events.registration.models.form_fields.RegistrationFormFields attribute), 163

data_versions (indico.modules.events.registration.models.form_fields.RegistrationFormFields attribute), 164

data_versions (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 169

data_versions (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 170

data_versions (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 171

data_versions (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 171

data_versions (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 172

data_versions (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 172

data_versions (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 179

data_versions (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 193

data_versions (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 235

data_versions (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 248

data_versions (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 249

data_versions (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 275

[data_versions \(indico.modules.events.registration.models.items.RegistrationFormTextCategories.models.categories.Category attribute\), 172](#)
[DataItem \(class in indico.modules.events.registration.stats\), 176](#)
[date \(indico.modules.rb.models.holidays.Holiday attribute\), 234](#)
[date \(indico.modules.rb.models.reservation_occurrences.ReservationOccurrences.models.events.abstracts.models.reviews.AbstractReview attribute\), 239](#)
[DAY \(indico.modules.rb.models.reservations.RepeatFrequency attribute\), 236](#)
[day_number_data \(indico.web.forms.fields.IndicoWeekDayRepetitionField attribute\), 277](#)
[db_schema_created \(in module indico.core.signals\), 43](#)
[declined \(indico.modules.events.registration.models.invitations.InvitationState attribute\), 167](#)
[deep_children_query \(in indico.modules.categories.models.categories.Category attribute\), 204](#)
[default_aspect \(indico.modules.rb.models.locations.Location attribute\), 234](#)
[default_aspect_id \(indico.modules.rb.models.locations.Location attribute\), 234](#)
[default_colors \(indico.modules.events.sessions.models.sessions.Session attribute\), 183](#)
[default_colors \(indico.modules.events.timetable.models.breaks.Break attribute\), 195](#)
[default_contribution_duration \(in indico.modules.events.sessions.models.sessions.Session attribute\), 183](#)
[default_data \(indico.modules.events.layout.models.menu.MenuEntryMixin attribute\), 133](#)
[default_data \(indico.modules.oauth.models.applications.SystemAppType attribute\), 244](#)
[default_event_settings \(indico.core.plugins.IndicoPlugin attribute\), 39](#)
[default_event_themes \(in indico.modules.categories.models.categories.Category attribute\), 204](#)
[default_list_config \(indico.modules.events.util.ListGeneratorBase attribute\), 95](#)
[default_location \(indico.modules.rb.models.locations.Location attribute\), 234](#)
[default_on_startup \(indico.modules.rb.models.aspects.Aspect attribute\), 231](#)
[default_options \(indico.web.forms.fields.FileField attribute\), 273](#)
[default_page \(indico.modules.events.models.events.Event attribute\), 80](#)
[default_page_id \(indico.modules.events.models.events.Event attribute\), 81](#)
[default_redirect_uri \(in indico.modules.oauth.models.applications.OAuthApplication attribute\), 244](#)
[default_render_mode \(in indico.modules.events.registration.models.items.RegistrationFormTextCategories.models.categories.Category attribute\), 204](#)
[default_render_mode \(in indico.modules.events.abstracts.models.abstracts.Abstract attribute\), 100](#)
[default_render_mode \(in indico.modules.events.abstracts.models.reviews.AbstractReview attribute\), 110](#)
[default_render_mode \(in indico.modules.events.contributions.models.contributions.Contribution attribute\), 119](#)
[default_render_mode \(in indico.modules.events.contributions.models.subcontributions.SubContribution attribute\), 126](#)
[default_render_mode \(in indico.modules.events.models.events.Event attribute\), 81](#)
[default_render_mode \(in indico.modules.events.papers.models.reviews.PaperReview attribute\), 147](#)
[default_render_mode \(in indico.modules.events.papers.models.revisions.PaperRevision attribute\), 148](#)
[default_render_mode \(in indico.modules.events.sessions.models.sessions.Session attribute\), 183](#)
[default_render_mode \(in indico.modules.events.timetable.models.breaks.Break attribute\), 195](#)
[default_render_mode \(in indico.modules.events.tracks.models.tracks.Track attribute\), 201](#)
[default_scopes \(indico.modules.oauth.models.applications.OAuthApplication attribute\), 244](#)
[default_settings \(indico.core.plugins.IndicoPlugin attribute\), 39](#)
[default_settings \(indico.modules.events.payment.plugins.PaymentPluginMixin attribute\), 156](#)
[default_settings \(indico.modules.vc.plugins.VCPluginMixin attribute\), 251](#)
[default_sort_alpha \(indico.modules.events.abstracts.fields.AbstractPersonLinkListFieldBase attribute\), 262](#)
[default_sort_alpha \(indico.modules.events.fields.PersonLinkListFieldBase attribute\), 262](#)
[default_ticket_template \(in indico.modules.categories.models.categories.Category attribute\), 204](#)
[default_ticket_template_id \(in indico.modules.categories.models.categories.Category attribute\), 204](#)
[default_user_settings \(indico.core.plugins.IndicoPlugin attribute\), 39](#)
[defaults \(indico.modules.events.settings.ThemeSettingsProxy attribute\), 99, 154](#)

definition (indico.modules.events.agreements.models.agreements.Agreement (in module indico.modules.events.sessions.operations), attribute), 117

definition (indico.modules.events.requests.models.requests.Request (in module indico.modules.events.sessions.operations), attribute), 179

delete() (indico.modules.categories.settings.CategorySettingsProxy method), 210

delete() (indico.modules.events.models.events.Event method), 81

delete() (indico.modules.events.notes.models.notes.EventNote method), 138

delete() (indico.modules.events.settings.EventSettingsProxy method), 98, 153

delete() (indico.modules.oauth.models.tokens.OAuthGrant method), 245

delete() (indico.modules.users.models.settings.UserSettingsProxy method), 216

delete() (indico.modules.vc.models.vc_rooms.VCRoomEventAssociation method), 249

delete_abstract() (in module indico.modules.events.abstracts.operations), 111

delete_abstract_comment() (in module indico.modules.events.abstracts.operations), 111

delete_abstract_files() (in module indico.modules.events.abstracts.operations), 111

delete_all() (indico.modules.categories.settings.CategorySettingsProxy method), 210

delete_all() (indico.modules.events.settings.EventSettingsProxy method), 98, 153

delete_all() (indico.modules.users.models.settings.UserSettingsProxy method), 217

delete_category() (in module indico.modules.categories.operations), 208

delete_comment() (in module indico.modules.events.papers.operations), 150

delete_comment_endpoint (in module indico.modules.events.abstracts.models.abstracts.AbstractAttribute), 100

delete_comment_endpoint (in module indico.modules.events.models.reviews.ProposalMixInAttribute), 90

delete_comment_endpoint (in module indico.modules.events.papers.models.papers.PaperAttribute), 144

delete_contribution() (in module indico.modules.events.contributions.operations), 128

delete_paper_template() (in module indico.modules.events.papers.operations), 150

delete_reference_type() (in module indico.modules.events.operations), 94

delete_session() (in module indico.modules.events.sessions.operations), 187

delete_session_block() (in module indico.modules.events.sessions.operations), 187

delete_subcontribution() (in module indico.modules.events.contributions.operations), 128

delete_timetable_entry() (in module indico.modules.events.timetable.operations), 198

delete_track() (in module indico.modules.events.tracks.operations), 202

deleted (in module indico.core.signals.category), 45

deleted (in module indico.core.signals.event), 46

deleted (indico.modules.vc.models.vc_rooms.VCRoomStatus attribute), 250

description (indico.modules.attachments.models.attachments.Attachment attribute), 220

description (indico.modules.attachments.models.folders.AttachmentFolder attribute), 222

description (indico.modules.designer.placeholders.CategoryTitlePlaceholder attribute), 258

description (indico.modules.designer.placeholders.EventDatesPlaceholder attribute), 255

description (indico.modules.designer.placeholders.EventDescriptionPlaceholder attribute), 255

description (indico.modules.designer.placeholders.EventOrgTextPlaceholder attribute), 255

description (indico.modules.designer.placeholders.EventRoomPlaceholder attribute), 258

description (indico.modules.designer.placeholders.EventSpeakersPlaceholder attribute), 259

description (indico.modules.designer.placeholders.EventTitlePlaceholder attribute), 258

description (indico.modules.designer.placeholders.EventVenuePlaceholder attribute), 258

description (indico.modules.designer.placeholders.RegistrationAddressPlaceholder attribute), 258

description (indico.modules.designer.placeholders.RegistrationAffiliationPlaceholder attribute), 257

description (indico.modules.designer.placeholders.RegistrationAmountPlaceholder attribute), 257

description (indico.modules.designer.placeholders.RegistrationCountryPlaceholder attribute), 258

description (indico.modules.designer.placeholders.RegistrationEmailPlaceholder attribute), 257

description (indico.modules.designer.placeholders.RegistrationFirstNamePlaceholder attribute), 257

description (indico.modules.designer.placeholders.RegistrationFullNameNoLastPlaceholder attribute), 255

description (indico.modules.designer.placeholders.RegistrationFullNameNoLastPlaceholder attribute), 256

description (indico.modules.designer.placeholders.RegistrationFullNameNoLastPlaceholder attribute), 256

- description (indico.modules.designer.placeholders.RegistrationFormPlaceholder.attribute), 256
- description (indico.modules.designer.placeholders.RegistrationFormPlaceholder.attribute), 256
- description (indico.modules.designer.placeholders.RegistrationFormPlaceholder.attribute), 255
- description (indico.modules.designer.placeholders.RegistrationFormPlaceholder.attribute), 255
- description (indico.modules.designer.placeholders.RegistrationFormPlaceholder.attribute), 256
- description (indico.modules.designer.placeholders.RegistrationFormPlaceholder.attribute), 256
- description (indico.modules.designer.placeholders.RegistrationFormPlaceholder.attribute), 257
- description (indico.modules.designer.placeholders.RegistrationFormPlaceholder.attribute), 258
- description (indico.modules.designer.placeholders.RegistrationFormPlaceholder.attribute), 257
- description (indico.modules.designer.placeholders.RegistrationFormPlaceholder.attribute), 257
- description (indico.modules.designer.placeholders.RegistrationFormPlaceholder.attribute), 257
- description (indico.modules.designer.placeholders.RegistrationFormPlaceholder.attribute), 256
- description (indico.modules.events.abstracts.placeholders.AbstractIDPlaceholder.attribute), 113
- description (indico.modules.events.abstracts.placeholders.AbstractSessionPlaceholder.attribute), 114
- description (indico.modules.events.abstracts.placeholders.AbstractTitlePlaceholder.attribute), 113
- description (indico.modules.events.abstracts.placeholders.AbstractTrackPlaceholder.attribute), 114
- description (indico.modules.events.abstracts.placeholders.AbstractWebPlaceholder.attribute), 114
- description (indico.modules.events.abstracts.placeholders.CaptchaPlaceholder.attribute), 114
- description (indico.modules.events.abstracts.placeholders.ContributionTypePlaceholder.attribute), 116
- description (indico.modules.events.abstracts.placeholders.ContributionURLPlaceholder.attribute), 116
- description (indico.modules.events.abstracts.placeholders.EventTitlePlaceholder.attribute), 113
- description (indico.modules.events.abstracts.placeholders.EventURLPlaceholder.attribute), 113
- description (indico.modules.events.abstracts.placeholders.JudgingCommentPlaceholder.attribute), 116
- description (indico.modules.events.abstracts.placeholders.PresentationHoursPlaceholder.attribute), 114
- description (indico.modules.events.abstracts.placeholders.SubscriptionNamePlaceholder.attribute), 114
- description (indico.modules.events.abstracts.placeholders.SubscriptionNamePlaceholder.attribute), 115
- description (indico.modules.events.abstracts.placeholders.SubscriptionNamePlaceholder.attribute), 114
- description (indico.modules.events.abstracts.placeholders.SubscriptionTitlePlaceholder.attribute), 114
- description (indico.modules.events.abstracts.placeholders.TargetAbstractIDPlaceholder.attribute), 115
- description (indico.modules.events.abstracts.placeholders.TargetAbstractTitlePlaceholder.attribute), 115
- description (indico.modules.events.abstracts.placeholders.TargetSubmitterFormPlaceholder.attribute), 115
- description (indico.modules.events.abstracts.placeholders.TargetSubmitterLinkPlaceholder.attribute), 115
- description (indico.modules.events.abstracts.placeholders.TargetSubmitterNamePlaceholder.attribute), 115
- description (indico.modules.events.agreements.placeholders.AgreementLinkPlaceholder.attribute), 119
- description (indico.modules.events.agreements.placeholders.PersonNamePlaceholder.attribute), 119
- description (indico.modules.events.contributions.models.fields.ContributionFormPlaceholder.attribute), 122
- description (indico.modules.events.contributions.models.types.ContributionFormPlaceholder.attribute), 128
- description (indico.modules.events.papers.models.templates.PaperTemplatePlaceholder.attribute), 149
- description (indico.modules.events.persons.placeholders.EmailPlaceholder.attribute), 157
- description (indico.modules.events.persons.placeholders.EventLinkPlaceholder.attribute), 157
- description (indico.modules.events.persons.placeholders.EventTitlePlaceholder.attribute), 157
- description (indico.modules.events.persons.placeholders.FirstNamePlaceholder.attribute), 158
- description (indico.modules.events.persons.placeholders.LastNamePlaceholder.attribute), 158
- description (indico.modules.events.persons.placeholders.RegisterLinkPlaceholder.attribute), 158
- description (indico.modules.events.registration.models.form_fields.RegistrationFormPlaceholder.attribute), 163
- description (indico.modules.events.registration.models.form_fields.RegistrationFormPlaceholder.attribute), 164
- description (indico.modules.events.registration.models.items.RegistrationFormPlaceholder.attribute), 169
- description (indico.modules.events.registration.models.items.RegistrationFormPlaceholder.attribute), 170
- description (indico.modules.events.registration.models.items.RegistrationFormPlaceholder.attribute), 171
- description (indico.modules.events.registration.models.items.RegistrationFormPlaceholder.attribute), 172
- description (indico.modules.events.registration.placeholders.invitations.FirsPlaceholder.attribute), 175
- description (indico.modules.events.registration.placeholders.invitations.InviPlaceholder.attribute), 175
- description (indico.modules.events.registration.placeholders.invitations.LastPlaceholder.attribute), 175
- description (indico.modules.events.registration.placeholders.registrations.EventPlaceholder.attribute), 174
- description (indico.modules.events.registration.placeholders.registrations.EventPlaceholder.attribute), 174

- duration (indico.modules.events.models.events.Event attribute), 81
- duration (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 184
- duration (indico.modules.events.timetable.models.breaks.Break attribute), 195
- duration (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 197
- duration (indico.modules.events.timetable.reschedule.RescheduleModel attribute), 200
- duration_display (indico.modules.events.contributions.models.contributions.contributions.registration.models.invitations.RegistrationInvitation attribute), 120
- duration_poster (indico.modules.events.contributions.models.contributions.contributions.registration.models.items.PersonalDataType attribute), 120
- ## E
- earliest_dt (indico.web.forms.fields.IndicoDateTimeField attribute), 272
- edit_comment_endpoint (in-
dico.modules.events.abstracts.models.abstracts.Abstract attribute), 100
- edit_comment_endpoint (in-
dico.modules.events.models.reviews.ProposalMixIn attribute), 90
- edit_comment_endpoint (in-
dico.modules.events.papers.models.papers.Paper attribute), 144
- edit_logs (indico.modules.rb.models.reservations.Reservation attribute), 237
- edit_review_endpoint (in-
dico.modules.events.abstracts.models.abstracts.Abstract attribute), 100
- edit_review_endpoint (in-
dico.modules.events.models.reviews.ProposalMixIn attribute), 90
- edit_review_endpoint (in-
dico.modules.events.papers.models.papers.Paper attribute), 144
- edit_track_mode (indico.modules.events.abstracts.models.abstracts.Abstract attribute), 100
- EditTrackMode (class in in-
dico.modules.events.abstracts.models.abstracts), 103
- effective_icon_url (indico.modules.categories.models.categories.Category attribute), 204
- email (indico.modules.attachments.models.principals.AttachmentPrincipal attribute), 224
- email (indico.modules.attachments.models.principals.AttachmentPrincipal attribute), 224
- email (indico.modules.auth.models.registration_requests.RegistrationRequest attribute), 242
- email (indico.modules.categories.models.principals.CategoryPrincipal attribute), 207
- email (indico.modules.events.contributions.models.principals.ContributionPrincipal attribute), 125
- email (indico.modules.events.models.persons.EventPerson attribute), 85
- email (indico.modules.events.models.persons.PersonLinkBase attribute), 87
- email (indico.modules.events.models.principals.EventPrincipal attribute), 87
- email (indico.modules.events.models.settings.EventSettingPrincipal attribute), 92
- email (indico.modules.events.registration.models.invitations.RegistrationInvitation attribute), 167
- email (indico.modules.events.registration.models.items.PersonalDataType attribute), 168
- email (indico.modules.events.registration.models.registrations.Registration attribute), 159
- email (indico.modules.events.sessions.models.principals.SessionPrincipal attribute), 186
- email (indico.modules.rb.models.blocking_principals.BlockingPrincipal attribute), 233
- email (indico.modules.users.models.emails.UserEmail attribute), 215
- email (indico.modules.users.models.users.User attribute), 212
- email_added (in module indico.core.signals.users), 49
- email_template (indico.modules.events.abstracts.models.email_logs.Abstract attribute), 105
- email_template_id (indico.modules.events.abstracts.models.email_logs.Abstract attribute), 105
- EmailListField (class in indico.web.forms.fields), 270
- EmailPlaceholder (class in in-
dico.modules.events.persons.placeholders), 157
- EmailRenderer (class in in-
dico.modules.events.logs.renderers), 136
- EmailRuleListField (class in in-
dico.modules.events.abstracts.fields), 262
- emails (indico.modules.events.logs.models.entries.EventLogRealm attribute), 136
- end_date (indico.modules.rb.models.blockings.Blocking attribute), 232
- end_dt (indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstract attribute), 103
- end_dt (indico.modules.events.contributions.models.contributions.Contribution attribute), 120
- end_dt (indico.modules.events.models.events.Event attribute), 81
- end_dt (indico.modules.events.papers.models.call_for_papers.CallForPaper attribute), 141
- end_dt (indico.modules.events.registration.models.forms.RegistrationForm attribute), 165
- end_dt (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 184
- end_dt (indico.modules.events.sessions.models.sessions.Session attribute), 184

attribute), 183

end_dt (indico.modules.events.surveys.models.surveys.Surveyevent (indico.modules.events.contributions.models.contributions.Contribution attribute), 188

end_dt (indico.modules.events.timetable.models.breaks.Break event (indico.modules.events.contributions.models.fields.ContributionField attribute), 195

end_dt (indico.modules.events.timetable.models.entries.TimetableEntry (indico.modules.events.contributions.models.subcontributions.SubCon attribute), 197

end_dt (indico.modules.rb.models.reservation_occurrences.ReservationOccurrence (indico.modules.events.contributions.models.types.ContributionType attribute), 239

end_dt (indico.modules.rb.models.reservations.Reservation event (indico.modules.events.fields.EventPersonListField attribute), 237

end_dt (indico.modules.rb.models.room_nonbookable_periods.NonBookablePeriod event (indico.modules.events.layout.models.images.ImageFile attribute), 231

end_dt_display (indico.modules.events.contributions.models.contributions.Contribution (indico.modules.events.layout.models.menu.EventPage attribute), 120

end_dt_display (indico.modules.events.models.events.Eventevent (indico.modules.events.layout.models.menu.MenuEntry attribute), 81

end_dt_local (indico.modules.events.models.events.Event event (indico.modules.events.logs.models.entries.EventLogEntry attribute), 81

end_dt_override (indico.modules.events.models.events.Eventevent (indico.modules.events.logs.models.entries.EventLogRealm attribute), 81

end_dt_poster (indico.modules.events.contributions.models.contributions.Contribution (indico.modules.events.models.events.Event attribute), 120

end_time (indico.modules.rb.models.room_bookable_hours.BookableHours (indico.modules.events.models.persons.EventPerson attribute), 231

endpoint (indico.modules.events.util.ListGeneratorBase event (indico.modules.events.models.settings.EventSetting attribute), 95

ends_after() (indico.modules.events.models.events.Event event (indico.modules.events.models.settings.EventSettingPrincipal attribute), 81

enforced_data (indico.modules.oauth.models.applications.SystemApplication (indico.modules.events.models.settings.EventSettingsMixin attribute), 244

entry_changed (in module indico.core.signals.acl), 44

entry_parent (indico.modules.events.util.ListGeneratorBase event (indico.modules.events.models.static_list_links.StaticListLink attribute), 95

equipment_types (indico.modules.rb.models.locations.Location event (indico.modules.events.notes.models.notes.EventNote attribute), 234

EquipmentType (class in indico.modules.rb.models.locations.Location event (indico.modules.events.papers.fields.PaperEmailSettingsField attribute), 233

error (indico.modules.oauth.provider.DisabledClientIdError event (indico.modules.events.papers.models.competences.PaperCompetence attribute), 246

Event (class in indico.modules.events.models.events), 79

event (indico.modules.attachments.models.folders.AttachmentFolder (indico.modules.events.papers.models.review_questions.PaperReview attribute), 222

event (indico.modules.designer.models.templates.DesignerTemplate (indico.modules.events.papers.models.templates.PaperTemplate attribute), 253

event (indico.modules.events.abstracts.fields.AbstractField event (indico.modules.events.registration.models.forms.RegistrationForm attribute), 262

event (indico.modules.events.abstracts.models.abstracts.Abstract event (indico.modules.events.registration.models.registrations.Registration attribute), 100

event (indico.modules.events.abstracts.models.email_templates.AbstractEmailTemplate (indico.modules.events.reminders.models.reminders.EventReminder attribute), 105

event (indico.modules.events.abstracts.models.review_questions.AbstractReviewQuestion (indico.modules.events.requests.models.requests.Request attribute), 108

event (indico.modules.events.agreements.models.agreements.Agreement (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 117

event (indico.modules.events.contributions.models.contributions.Contribution attribute), 120

event (indico.modules.events.contributions.models.fields.ContributionField attribute), 122

event (indico.modules.events.contributions.models.subcontributions.SubCon attribute), 126

event (indico.modules.events.contributions.models.types.ContributionType attribute), 128

event (indico.modules.events.fields.EventPersonListField attribute), 261

event (indico.modules.events.layout.models.images.ImageFile attribute), 131

event (indico.modules.events.layout.models.menu.EventPage attribute), 131

event (indico.modules.events.layout.models.menu.MenuEntry attribute), 132

event (indico.modules.events.logs.models.entries.EventLogEntry attribute), 135

event (indico.modules.events.logs.models.entries.EventLogRealm attribute), 136

event (indico.modules.events.models.events.Event attribute), 81

event (indico.modules.events.models.persons.EventPerson attribute), 85

event (indico.modules.events.models.settings.EventSetting attribute), 92

event (indico.modules.events.models.settings.EventSettingPrincipal attribute), 92

event (indico.modules.events.models.settings.EventSettingsMixin attribute), 93

event (indico.modules.events.models.static_list_links.StaticListLink attribute), 93

event (indico.modules.events.notes.models.notes.EventNote attribute), 138

event (indico.modules.events.papers.fields.PaperEmailSettingsField attribute), 266

event (indico.modules.events.papers.models.competences.PaperCompetence attribute), 143

event (indico.modules.events.papers.models.papers.Paper attribute), 144

event (indico.modules.events.papers.models.review_questions.PaperReview attribute), 145

event (indico.modules.events.papers.models.templates.PaperTemplate attribute), 149

event (indico.modules.events.registration.models.forms.RegistrationForm attribute), 165

event (indico.modules.events.registration.models.registrations.Registration attribute), 159

event (indico.modules.events.reminders.models.reminders.EventReminder attribute), 178

event (indico.modules.events.requests.models.requests.Request attribute), 179

event (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 179

- attribute), 184
- event (indico.modules.events.sessions.models.sessions.SessionEvent (indico.modules.events.layout.models.menu.MenuEntry attribute), 183
- event (indico.modules.events.static.models.static.StaticSiteEvent (indico.modules.events.logs.models.entries.EventLogEntry attribute), 203
- event (indico.modules.events.surveys.models.surveys.SurveyEvent (indico.modules.events.models.persons.EventPerson attribute), 188
- event (indico.modules.events.timetable.models.breaks.BreakEvent (indico.modules.events.models.persons.EventPersonLink attribute), 195
- event (indico.modules.events.timetable.models.entries.TimetableEntry (indico.modules.events.models.principals.EventPrincipal attribute), 197
- event (indico.modules.events.tracks.models.tracks.TrackEvent (indico.modules.events.models.references.EventReference attribute), 201
- event (indico.modules.events.util.ListGeneratorBaseEvent (indico.modules.events.models.settings.EventSetting attribute), 95
- event (indico.modules.rb.models.reservations.ReservationEvent (indico.modules.events.models.settings.EventSettingPrincipal attribute), 237
- event (indico.modules.vc.models.vc_rooms.VCRoomEventAssociation (indico.modules.events.models.settings.EventSettingsMixin attribute), 249
- event (indico.modules.vc.models.vc_rooms.VCRoomLinkTypeEvent (indico.modules.events.models.static_list_links.StaticListLink attribute), 250
- event_backref_name (in- event_id (indico.modules.events.notes.models.notes.EventNote dico.modules.events.abstracts.models.review_questions.AbstractReviewQuestion attribute), 108
- event_backref_name (in- event_id (indico.modules.events.papers.models.competences.PaperCompetence attribute), 145
- event_creation_notification_emails (in- event_id (indico.modules.events.papers.models.templates.PaperTemplate dico.modules.categories.models.categories.Category attribute), 204
- event_creation_restricted (in- event_id (indico.modules.events.registration.models.forms.RegistrationForm dico.modules.categories.models.categories.Category attribute), 205
- event_id (indico.modules.attachments.models.folders.AttachmentFolder (indico.modules.events.reminders.models.reminders.EventReminder attribute), 222
- event_id (indico.modules.designer.models.templates.DesignerTemplate (indico.modules.events.requests.models.requests.Request attribute), 253
- event_id (indico.modules.events.abstracts.models.abstracts.AbstractEvent (indico.modules.events.sessions.models.sessions.Session attribute), 101
- event_id (indico.modules.events.abstracts.models.email_templates.EmailTemplate (indico.modules.events.static.models.static.StaticSite attribute), 105
- event_id (indico.modules.events.abstracts.models.review_questions.AbstractReviewQuestion (indico.modules.events.surveys.models.surveys.Survey attribute), 108
- event_id (indico.modules.events.agreements.models.agreements.Agreement (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 117
- event_id (indico.modules.events.contributions.models.contributions.Contribution (indico.modules.events.tracks.models.tracks.Track attribute), 120
- event_id (indico.modules.events.contributions.models.fields.ContributionField (indico.modules.rb.models.reservations.Reservation attribute), 122
- event_id (indico.modules.events.contributions.models.types.ContributionType (indico.modules.vc.models.vc_rooms.VCRoomEventAssociation attribute), 128
- event_id (indico.modules.events.layout.models.images.ImageEvent (indico.modules.categories.models.categories.Category attribute), 131
- event_id (indico.modules.events.layout.models.menu.EventPageEvent_message (indico.modules.categories.models.categories.Category attribute), 205
- event_id (indico.modules.events.layout.models.menu.EventPageEvent_message_mode (in-

dico.modules.categories.models.categories.Category attribute), 205

event_or_id() (in module dico.modules.events.settings), 99, 154

event_ref (indico.modules.events.layout.models.menu.MenuEntryMixin attribute), 133

event_settings (indico.core.plugins.IndicoPlugin attribute), 39

event_settings_converters (indico.core.plugins.IndicoPlugin attribute), 40

event_settings_form (indico.modules.events.payment.plugins.PaymentPluginMixin attribute), 156

event_start_delta (indico.modules.events.reminders.models.reminders.EventReminder attribute), 178

EventACLProxy (class in indico.modules.events.settings), 97, 151

EventDatesPlaceholder (class in indico.modules.designer.placeholders), 255

EventDescriptionPlaceholder (class in indico.modules.designer.placeholders), 255

EventLinkPlaceholder (class in indico.modules.events.persons.placeholders), 157

EventLinkPlaceholder (class in indico.modules.events.registration.placeholders.registrations), 174

EventLogEntry (class in indico.modules.events.logs.models.entries), 135

EventLogKind (class in indico.modules.events.logs.models.entries), 136

EventLogRealm (class in indico.modules.events.logs.models.entries), 136

EventLogRendererBase (class in indico.modules.events.logs.renderers), 137

EventMessageMode (class in indico.modules.categories.models.categories), 207

EventNote (class in indico.modules.events.notes.models.notes), 138

EventNoteRevision (class in indico.modules.events.notes.models.notes), 140

EventOrgTextPlaceholder (class in indico.modules.designer.placeholders), 255

EventPage (class in indico.modules.events.layout.models.menu), 131

EventPerson (class in indico.modules.events.models.persons), 85

EventPersonLink (class in indico.modules.events.models.persons), 86

EventPersonLinkListField (class in indico.modules.events.fields), 261

EventPersonListField (class in indico.modules.events.fields), 261

EventPrincipal (class in indico.modules.events.models.principals), 87

EventReference (class in indico.modules.events.models.references), 88

EventReminder (class in indico.modules.events.reminders.models.reminders), 177

EventRoomPlaceholder (class in indico.modules.designer.placeholders), 258

events_backref_name (indico.modules.attachments.models.folders.AttachmentFolder attribute), 222

events_backref_name (indico.modules.events.notes.models.notes.EventNote attribute), 138

EventSeries (class in indico.modules.events.models.series), 91

EventSetting (class in indico.modules.events.models.settings), 92

EventSettingPrincipal (class in indico.modules.events.models.settings), 92

EventSettingProperty (class in indico.modules.events.settings), 98, 152

EventSettingsMixin (class in indico.modules.events.models.settings), 93

EventSettingsProxy (class in indico.modules.events.settings), 98, 152

EventSpeakersPlaceholder (class in indico.modules.designer.placeholders), 259

EventTitlePlaceholder (class in indico.modules.designer.placeholders), 258

EventTitlePlaceholder (class in indico.modules.events.abstracts.placeholders), 113

EventTitlePlaceholder (class in indico.modules.events.persons.placeholders), 157

EventTitlePlaceholder (class in indico.modules.events.registration.placeholders.registrations), 174

EventType (class in indico.modules.events.models.events), 84

EventURLPlaceholder (class in indico.modules.events.abstracts.placeholders), 113

- EventVenuePlaceholder (class in indico.modules.designer.placeholders.RegistrationPhonePlaceholder attribute), 258
- expired (indico.modules.events.static.models.static.StaticSiteState attribute), 203
- expires (indico.modules.oauth.models.tokens.OAuthToken attribute), 245
- extend_defaults() (indico.modules.users.ext.ExtraUserPreferences method), 219
- extend_end_dt() (indico.modules.events.timetable.models.entries.TimetableEntry method), 197
- extend_form() (indico.modules.users.ext.ExtraUserPreferences method), 219
- extend_parent() (indico.modules.events.timetable.models.entries.TimetableEntry method), 197
- extend_start_dt() (indico.modules.events.timetable.models.entries.TimetableEntry method), 197
- external_identities (indico.modules.users.models.users.User field_data attribute), 212
- external_url (indico.modules.events.models.events.Event field_data attribute), 81
- extra_cc_emails (indico.modules.events.abstracts.models.entries.AbstractEntry field_data attribute), 106
- extra_emails (indico.modules.auth.models.registration_requests.RegistrationRequest field_data attribute), 242
- extra_key_cols (indico.modules.events.models.settings.EventSettings field_data attribute), 92
- ExtraUserPreferences (class in indico.modules.users.ext), 219
- ## F
- f_last (indico.modules.users.models.users.NameFormat attribute), 211
- f_last_upper (indico.modules.users.models.users.NameFormat attribute), 211
- failed (indico.modules.events.payment.models.transactions.Transaction field_data attribute), 155
- failed (indico.modules.events.static.models.static.StaticSiteState attribute), 203
- favorite_categories (indico.modules.users.models.users.User field_type attribute), 212
- favorite_users (indico.modules.users.models.users.User field_type attribute), 212
- field (indico.modules.designer.placeholders.RegistrationAddressPlaceholder attribute), 258
- field (indico.modules.designer.placeholders.RegistrationAffiliationPlaceholder attribute), 257
- field (indico.modules.designer.placeholders.RegistrationCountryPlaceholder attribute), 258
- field (indico.modules.designer.placeholders.RegistrationEmailPlaceholder attribute), 257
- field (indico.modules.designer.placeholders.RegistrationFirstNamePlaceholder attribute), 257
- field (indico.modules.designer.placeholders.RegistrationLastNamePlaceholder attribute), 257
- field (indico.modules.designer.placeholders.RegistrationPhonePlaceholder attribute), 258
- field (indico.modules.designer.placeholders.RegistrationPositionPlaceholder attribute), 257
- field (indico.modules.designer.placeholders.RegistrationTitlePlaceholder attribute), 256
- field (indico.modules.events.contributions.models.fields.ContributionField attribute), 122
- field (indico.modules.events.registration.models.items.RegistrationFormItem attribute), 170
- field (indico.modules.events.surveys.models.items.SurveyQuestion attribute), 191
- field (indico.modules.events.contributions.models.fields.ContributionField attribute), 122
- field (indico.modules.events.registration.models.items.PersonalData attribute), 168
- field_data (indico.modules.events.registration.models.registrations.Registration attribute), 161
- field_data (indico.modules.events.surveys.models.items.SurveyItem attribute), 190
- field_data (indico.modules.events.surveys.models.items.SurveyQuestion attribute), 191
- field_data (indico.modules.events.surveys.models.items.SurveySection attribute), 192
- field_data (indico.modules.events.surveys.models.items.SurveyText attribute), 192
- field_data_id (indico.modules.events.registration.models.registrations.Registration attribute), 161
- field_id (indico.modules.events.registration.models.form_fields.RegistrationFormField attribute), 163
- field_impl (indico.modules.events.registration.models.form_fields.RegistrationFormField attribute), 163
- field_pd (indico.modules.events.registration.models.items.RegistrationFormItem attribute), 170
- field_type (indico.modules.events.contributions.models.fields.ContributionField attribute), 122
- field_type (indico.modules.events.surveys.models.items.SurveyItem attribute), 190
- field_type (indico.modules.events.surveys.models.items.SurveyQuestion attribute), 191
- field_type (indico.modules.events.surveys.models.items.SurveySection attribute), 192
- field_type (indico.modules.events.surveys.models.items.SurveyText attribute), 192
- field_type (indico.modules.events.abstracts.models.abstracts.AbstractEntry attribute), 101
- field_type (indico.modules.events.contributions.models.contributions.Contribution attribute), 120
- FieldPlaceholder (class in indico.modules.events.registration.placeholders.registrations), 257
- fields (indico.modules.events.registration.models.items.RegistrationFormSection attribute), 171
- fields (indico.modules.users.ext.ExtraUserPreferences attribute), 219

tribute), 219

FieldStats (class in indico.modules.events.registration.stats), 177

file (indico.modules.attachments.models.attachments.Attachment attribute), 220

file (indico.modules.attachments.models.attachments.Attachment attribute), 222

file (indico.modules.events.registration.models.registrations.Registration attribute), 161

file_id (indico.modules.attachments.models.attachments.Attachment attribute), 220

file_required (indico.modules.events.registration.models.registrations.Registration attribute), 161

file_required (indico.modules.events.static.models.static.StaticSite attribute), 203

FileField (class in indico.web.forms.fields), 273

filename (indico.modules.attachments.models.attachments.Attachment attribute), 221

filename (indico.modules.designer.models.images.DesignerImage attribute), 253

filename (indico.modules.events.abstracts.models.files.AbstractFile attribute), 107

filename (indico.modules.events.layout.models.images.ImageFile attribute), 131

filename (indico.modules.events.papers.models.files.PaperFile attribute), 143

filename (indico.modules.events.papers.models.templates.PaperTemplate attribute), 149

filename (indico.modules.events.registration.models.registrations.Registration attribute), 161

filename (indico.modules.events.static.models.static.StaticSite attribute), 203

files (indico.modules.events.papers.models.papers.Paper attribute), 144

filter_available() (indico.modules.rb.models.rooms.Room static method), 227

filter_choices (indico.modules.events.contributions.models.fields.ContribField attribute), 122

filter_overlap() (indico.modules.rb.models.reservation_occurrences.ReservationOccurrence static method), 239

find_all() (indico.modules.rb.models.rooms.Room class method), 227

find_available_vc_equipment() (indico.modules.rb.models.rooms.Room method), 228

find_event_vc_rooms() (in module indico.modules.vc.util), 250

find_excluded_days() (indico.modules.rb.models.reservations.Reservation method), 237

find_for_event() (indico.modules.vc.models.vc_rooms.VCRoomEventAssociation class method), 249

find_latest_entry_end_dt() (in module indico.modules.events.timetable.util), 199

find_latest_for_event() (indico.modules.events.requests.models.requests.Request class method), 179

find_next_start_dt() (in module indico.modules.events.timetable.util), 199

find_type_mapping() (indico.modules.rb.models.reservations.Reservation method), 237

RegistrationData_with() (indico.modules.rb.models.reservation_occurrences.ReservationOccurrence class method), 239

find_overlapping_with() (indico.modules.rb.models.reservations.Reservation static method), 237

find_site_with_attribute() (indico.modules.rb.models.rooms.Room class method), 228

find_with_filters() (indico.modules.rb.models.blocked_rooms.BlockedRoom class method), 232

image_file_filters() (indico.modules.rb.models.reservation_occurrences.ReservationOccurrence class method), 239

image_file_filters() (indico.modules.rb.models.rooms.Room static method), 228

finished (indico.modules.events.surveys.models.surveys.SurveyState attribute), 190

first_last (indico.modules.users.models.users.NameFormat attribute), 211

first_last_placeholder (indico.modules.users.models.users.NameFormat attribute), 211

first_name (indico.modules.events.models.persons.EventPerson attribute), 85

first_name (indico.modules.events.models.persons.PersonLinkBase attribute), 87

first_name (indico.modules.events.registration.models.invitations.Registration attribute), 167

first_name (indico.modules.events.registration.models.items.PersonalData attribute), 168

first_name (indico.modules.events.registration.models.registrations.Registration attribute), 159

first_name (indico.modules.users.models.users.User attribute), 212

FirstNamePlaceholder (class in indico.modules.events.persons.placeholders), 158

FirstNamePlaceholder (class in indico.modules.events.registration.placeholders.invitations), 175

FirstNamePlaceholder (class in indico.modules.events.registration.placeholders.registrations), 174

fit_session_block_entry() (in module indico.modules.events.timetable.operations), 198

fits_period() (indico.modules.rb.models.room_bookable_hours.BookableHours method), 231

- flash_info_message() (in module dico.modules.events.util.ListGeneratorBase class method), 95
- floor (indico.modules.rb.models.rooms.Room attribute), 228
- flower (indico.modules.oauth.models.applications.SystemAppType attribute), 244
- folder (indico.modules.attachments.models.attachments.Attachment attribute), 220
- folder_created (in module dico.core.signals.attachments), 45
- folder_deleted (in module dico.core.signals.attachments), 45
- folder_id (indico.modules.attachments.models.attachments.Attachment attribute), 220
- folder_id (indico.modules.attachments.models.principals.AttachmentFieldPrincipal attribute), 224
- folder_updated (in module dico.core.signals.attachments), 45
- for_user() (indico.modules.events.models.persons.EventPerson class method), 85
- form (indico.modules.events.requests.base.RequestDefinitionBase attribute), 181
- form_defaults (indico.modules.events.requests.base.RequestDefinitionBase attribute), 181
- form_items (indico.modules.events.registration.models.forms.RegistrationForm attribute), 165
- form_validated (in module dico.core.signals), 43
- format_display_full_name() (in module dico.modules.users.models.users), 215
- format_feature_names() (in module dico.modules.events.features.util), 130
- friendly_data (indico.modules.events.contributions.models.fields.ContributionFieldValueBase attribute), 123
- friendly_data (indico.modules.events.registration.models.registrations.RegistrationData attribute), 161
- friendly_id (indico.modules.events.abstracts.models.abstracts.Abstract attribute), 101
- friendly_id (indico.modules.events.contributions.models.contributions.Contribution attribute), 120
- friendly_id (indico.modules.events.contributions.models.subcontributions.SubContribution attribute), 127
- friendly_id (indico.modules.events.registration.models.registrations.Registration attribute), 159
- friendly_id (indico.modules.events.sessions.models.sessions.Session attribute), 183
- friendly_id (indico.modules.events.surveys.models.submissions.SurveySubmission attribute), 193
- friendly_name (indico.modules.vc.plugins.VCPluginMixin attribute), 251
- full_access (indico.modules.categories.models.principals.CategoryPrincipal attribute), 207
- full_access (indico.modules.events.contributions.models.principals.ContributionPrincipal attribute), 125
- full_access (indico.modules.events.models.principals.EventPrincipal attribute), 87
- full_access (indico.modules.events.sessions.models.principals.SessionPrincipal attribute), 186
- full_name (indico.modules.events.registration.models.registrations.Registration attribute), 159
- full_name (indico.modules.rb.models.rooms.Room attribute), 228
- full_name (indico.modules.users.models.users.PersonMixin attribute), 211
- full_title (indico.modules.events.models.reviews.ProposalGroupProxy attribute), 89
- full_title (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 184
- full_title (indico.modules.events.tracks.models.tracks.Track attribute), 184
- full_title_attr (indico.modules.events.models.reviews.ProposalGroupProxy attribute), 89
- generate_content() (indico.modules.attachments.preview.MarkdownPreviewer class method), 226
- generate_content() (indico.modules.attachments.preview.Previewer class method), 226
- generate_content() (indico.modules.attachments.preview.TextPreviewer class method), 226
- generate_name() (indico.modules.rb.models.rooms.Room method), 228
- generate_pdf_from_sessions() (in module dico.modules.events.sessions.util), 187
- generate_spreadsheet_from_abstracts() (in module dico.modules.events.abstracts.util), 112
- generate_spreadsheet_from_contributions() (in module dico.modules.events.contributions.util), 129
- generate_spreadsheet_from_registrations() (in module dico.modules.events.registration.util), 173
- generate_spreadsheet_from_sessions() (in module dico.modules.events.sessions.util), 187
- generate_spreadsheet_from_survey() (in module dico.modules.events.surveys.util), 194
- generate_state_url() (in module dico.modules.events.util.ListGeneratorBase class method), 95
- generate_ticket_qr_code (in module dico.core.signals.event), 46
- generate_ticket_qr_code() (in module dico.modules.events.registration.util), 173
- get() (indico.modules.categories.settings.CategorySettingsProxy method), 210
- get() (indico.modules.events.settings.EventACLProxy method), 98, 152
- get() (indico.modules.events.settings.EventSettingsProxy method), 98, 151

[get\(\)](#) (indico.modules.oauth.models.tokens.OAuthGrant class method), 245
 [get\(\)](#) (indico.modules.users.models.settings.UserSettingsProxy method), 217
 [get_access_list\(\)](#) (indico.modules.events.contributions.models.subcontributions.models.SubContribution method), 127
 [get_active_payment_plugins\(\)](#) (in module indico.modules.events.payment.util), 156
 [get_admin_emails\(\)](#) (in module indico.modules.users.util), 218
 [get_agreement_definitions\(\)](#) (in module indico.modules.events.agreements.util), 118
 [get_all\(\)](#) (indico.modules.categories.settings.CategorySettingsProxy method), 210
 [get_all\(\)](#) (indico.modules.events.settings.EventSettingsProxy method), 99, 153
 [get_all\(\)](#) (indico.modules.users.models.settings.UserSettingsProxy method), 217
 [get_all_for_event\(\)](#) (in module indico.modules.events.registration.models.registrations.Registration class method), 159
 [get_all_templates\(\)](#) (in module indico.modules.designer.util), 254
 [get_allowed_sender_emails\(\)](#) (in module indico.modules.events.models.events.Event method), 81
 [get_attached_folders\(\)](#) (in module indico.modules.attachments.util), 225
 [get_attached_items\(\)](#) (in module indico.modules.attachments.util), 225
 [get_attachment_count\(\)](#) (in module indico.modules.categories.util), 208
 [get_attribute_by_name\(\)](#) (in module indico.modules.rb.models.locations.Location method), 234
 [get_attribute_by_name\(\)](#) (in module indico.modules.rb.models.rooms.Room method), 228
 [get_attribute_value\(\)](#) (in module indico.modules.rb.models.rooms.Room method), 228
 [get_base_ical_parameters\(\)](#) (in module indico.modules.events.util), 95
 [get_blocked_rooms\(\)](#) (in module indico.modules.rb.models.rooms.Room method), 228
 [get_blueprints](#) (in module indico.core.signals.plugin), 48
 [get_blueprints\(\)](#) (indico.core.plugins.IndicoPlugin method), 40
 [get_buildings\(\)](#) (indico.modules.rb.models.locations.Location method), 235
 [get_category_stats\(\)](#) (in module indico.modules.categories.util), 208
 [get_category_timetable\(\)](#) (in module indico.modules.events.timetable.util), 199
 [get_cloners](#) (in module indico.core.signals.event_management), 48
 [get_color_for_username\(\)](#) (in module indico.modules.users.util), 218
 [get_conditions](#) (in module indico.core.signals), 43
 [get_conference_themes](#) (in module indico.core.signals.plugin), 48
 [get_conflicting_occurrences\(\)](#) (in module indico.modules.rb.models.reservations.Reservation method), 237
 [get_contribs_by_year\(\)](#) (in module indico.modules.categories.util), 208
 [get_contribution\(\)](#) (indico.modules.events.models.events.Event method), 81
 [get_contribution_field\(\)](#) (in module indico.modules.events.models.events.Event method), 81
 [get_contribution_ical_file\(\)](#) (in module indico.modules.events.contributions.util), 129
 [get_contributions_with_paper_submitted_by_user\(\)](#) (in module indico.modules.events.papers.util), 151
 [get_contributions_with_user_as_submitter\(\)](#) (in module indico.modules.events.contributions.util), 129
 [get_contributions_with_user_paper_submission_rights\(\)](#) (in module indico.modules.events.papers.util), 151
 [get_css_file_data\(\)](#) (in module indico.modules.events.layout.util), 134
 [get_css_url\(\)](#) (in module indico.modules.events.layout.util), 134
 [get_data\(\)](#) (indico.modules.events.logs.renderers.EventLogRendererBase class method), 137
 [get_data\(\)](#) (indico.modules.events.logs.renderers.SimpleRenderer class method), 137
 [get_default_booking_interval\(\)](#) (in module indico.modules.rb.util), 240
 [get_default_folder_names\(\)](#) (in module indico.modules.attachments.util), 225
 [get_default_template_on_category\(\)](#) (in module indico.modules.designer.util), 254
 [get_definitions](#) (in module indico.core.signals.agreements), 44
 [get_delete_comment_url\(\)](#) (in module indico.modules.events.models.reviews.ProposalMixin method), 90
 [get_disallowed_features\(\)](#) (in module indico.modules.events.features.util), 130
 [get_download_url\(\)](#) (in module indico.modules.attachments.models.attachments.Attachment method), 220
 [get_enabled_features\(\)](#) (in module indico.modules.events.features.util), 130
 [get_equipment_by_name\(\)](#) (in

- dico.modules.rb.models.locations.Location method), 235
- get_event() (in module `indico.modules.attachments.util`), 226
- get_event_management_url() (in `indico.modules.events.payment.plugins.PaymentPluginMixin` method), 156
- get_event_regforms() (in module `indico.modules.events.registration.util`), 173
- get_event_request_definitions (in module `indico.core.signals.plugin`), 48
- get_event_section_data() (in module `indico.modules.events.registration.util`), 173
- get_event_themes_files (in module `indico.core.signals.plugin`), 48
- get_events_by_year() (in module `indico.modules.categories.util`), 209
- get_events_created_by() (in module `indico.modules.events.util`), 95
- get_events_managed_by() (in module `indico.modules.events.util`), 95
- get_events_registered() (in module `indico.modules.events.registration.util`), 173
- get_events_with_abstract_persons() (in module `indico.modules.events.abstracts.util`), 112
- get_events_with_abstract_reviewer_convener() (in module `indico.modules.events.abstracts.util`), 112
- get_events_with_linked_contributions() (in module `indico.modules.events.contributions.util`), 129
- get_events_with_linked_event_persons() (in module `indico.modules.events.util`), 96
- get_events_with_linked_sessions() (in module `indico.modules.events.sessions.util`), 187
- get_events_with_paper_roles() (in module `indico.modules.events.papers.util`), 151
- get_events_with_submitted_surveys() (in module `indico.modules.events.surveys.util`), 194
- get_feature_definition() (in module `indico.modules.events.features.util`), 130
- get_feature_definitions (in module `indico.core.signals.event`), 46
- get_feature_definitions() (in module `indico.modules.events.features.util`), 130
- get_field_value() (`indico.modules.events.contributions.models.contributors.indico.modules.events.contributors.util.ListGeneratorBase` method), 122
- get_field_values() (in module `indico.modules.events.util`), 96
- get_fields (in module `indico.core.signals`), 43
- get_file_previewer() (in module `indico.modules.attachments.preview`), 226
- get_file_previewers (in module `indico.core.signals.attachments`), 45
- get_file_previewers() (in module `indico.modules.attachments.preview`), 226
- get_for_event() (`indico.modules.events.layout.models.menu.MenuEntry` static method), 132
- get_for_linked_object() (in `indico.modules.attachments.models.folders.AttachmentFolder` class method), 222
- get_inherited_linked_object() (in `indico.modules.events.notes.models.notes.EventNote` class method), 139
- get_friendly_data() (in `indico.modules.events.registration.models.form_fields.RegistrationFormFields` method), 163
- get_friendly_data() (in `indico.modules.events.registration.models.registrations.RegistrationFormFields` method), 161
- get_full_name() (`indico.modules.events.registration.models.registrations.RegistrationFormFields` method), 159
- get_full_name() (`indico.modules.users.models.users.PersonMixin` method), 211
- get_full_name() (`indico.modules.users.models.users.User` method), 212
- get_highest() (`indico.modules.events.contributions.models.persons.AuthorizedPerson` class method), 124
- get_icon_data_cte() (in `indico.modules.categories.models.categories.Category` class method), 205
- get_image_data() (in module `indico.modules.categories.util`), 209
- get_inherited_templates() (in module `indico.modules.designer.util`), 254
- get_invalid_regforms() (in `indico.modules.events.payment.plugins.PaymentPluginMixin` method), 156
- get_last_revision() (`indico.modules.events.models.reviews.ProposalMixin` method), 90
- get_last_revision() (`indico.modules.events.papers.models.papers.Paper` method), 144
- get_linked_events() (in module `indico.modules.users.util`), 218
- get_linked_for_event() (in `indico.modules.vc.models.vc_rooms.VCRoomEventAssociation` class method), 249
- get_linked_to_description() (in module `indico.modules.vc.util`), 250
- get_list_generator_base() (`indico.modules.events.contributors.util.ListGeneratorBase` method), 95
- get_log_renderers (in module `indico.core.signals.event`), 46
- get_log_renderers() (in module `indico.modules.events.logs.util`), 136
- get_logo_data() (in module `indico.modules.events.layout.util`), 134
- get_managed_vc_plugins() (in module `indico.modules.vc.util`), 250
- get_management_roles (in module `indico.modules.vc.util`), 250

- dico.core.signals.acl), 44
- get_manager_list() (indico.modules.events.contributions.models.contributions.SubContribution method), 127
- get_manager_notification_emails() (in-dico.modules.events.requests.base.RequestDefinitionBase class method), 181
- get_members() (indico.modules.groups.core.GroupProxy method), 247
- get_menu_entries_from_signal() (in module indico.modules.events.layout.util), 134
- get_menu_entry_by_name() (in module indico.modules.events.layout.util), 134
- get_message() (indico.modules.rb.models.reservations.RepeatMapping class method), 236
- get_method_name() (in-dico.modules.events.payment.plugins.PaymentPluginMixin method), 156
- get_named_default_group() (in-dico.modules.groups.core.GroupProxy class method), 247
- get_nested_attached_items() (in module indico.modules.attachments.util), 226
- get_nested_entries() (in module indico.modules.events.timetable.util), 200
- get_non_inheriting_objects() (in-dico.modules.events.contributions.models.contributions.ContributionMixin class method), 120
- get_non_inheriting_objects() (in-dico.modules.events.models.events.Event class method), 81
- get_non_inheriting_objects() (in-dico.modules.events.sessions.models.sessions.Session class method), 183
- get_not_deletable_templates() (in module indico.modules.designer.util), 254
- get_notification_bcc_list() (in-dico.modules.vc.plugins.VCPluginMixin method), 251
- get_notification_cc_list() (in-dico.modules.vc.plugins.VCPluginMixin method), 251
- get_notification_template() (in-dico.modules.events.requests.base.RequestDefinitionBase class method), 181
- get_object_from_args() (in module indico.modules.events.util), 96
- get_or_create() (indico.modules.attachments.models.folders.AttachmentFolder class method), 222
- get_or_create() (indico.modules.events.notes.models.notes.EventNote class method), 139
- get_or_create_default() (in-dico.modules.attachments.models.folders.AttachmentFolder class method), 223
- get_overlap() (indico.modules.rb.models.reservation_occurrences.ReservationDefinition class method), 239
- get_override_multiple_items_field() (in-dico.web.forms.fields.OverrideMultipleItemsField method), 274
- get_participant_list_base() (in-dico.modules.rb.models.rooms.Room class method), 228
- get_participant_list_columns() (in-dico.modules.events.registration.settings.RegistrationSettingsProxy method), 175
- get_participant_list_form_ids() (in-dico.modules.events.registration.settings.RegistrationSettingsProxy method), 175
- get_payment_plugins() (in module indico.modules.events.payment.util), 156
- get_pdf() (indico.modules.designer.pdf.DesignerPDFBase method), 254
- get_personal_data() (in-dico.modules.events.registration.models.registrations.RegistrationForm class method), 159
- get_personal_data_field_id() (in-dico.modules.events.registration.models.forms.RegistrationForm class method), 165
- get_placeholder_options() (in module indico.modules.designer.util), 254
- get_placeholders (in module indico.core.signals), 43
- get_registration_conference_themes() (in module indico.modules.events.layout.util), 134
- get_plugin_template_module() (in module indico.core.plugins), 42
- get_protection_cte() (in-dico.modules.categories.models.categories.Category class method), 205
- get_protection_parent_cte() (in-dico.modules.categories.models.categories.Category method), 205
- get_published_registrations() (in module indico.modules.events.registration.util), 173
- get_questions_for_review_type() (in-dico.modules.events.papers.models.call_for_papers.CallForPaper class method), 141
- get_random_color() (in module indico.modules.events.util), 96
- get_recent_news() (in module indico.modules.news.util), 261
- get_registration() (indico.modules.events.registration.models.forms.RegistrationForm class method), 165
- get_registration_folders_with_tickets() (in module indico.modules.events.registration.util), 173
- get_registered_categories() (in module indico.modules.users.util), 218
- get_relative_event_ids() (in-dico.modules.events.models.events.Event class method), 82
- get_reservation_definition() (in module indico.modules.events.util), 96

[get_user_contributions_to_review\(\)](#) (in module `indico.modules.events.papers.util`), 151
[get_user_reviewed_contributions\(\)](#) (in module `indico.modules.events.papers.util`), 151
[get_user_tracks\(\)](#) (in module `indico.modules.events.abstracts.util`), 113
[get_vars_js\(\)](#) (`indico.core.plugins.IndicoPlugin` method), 40
[get_vc_equipment\(\)](#) (`indico.modules.rb.models.reservations.Reservation` method), 237
[get_vc_plugins\(\)](#) (in module `indico.modules.vc.util`), 250
[get_vc_room_attach_form_defaults\(\)](#) (`indico.modules.vc.plugins.VCPluginMixin` method), 251
[get_vc_room_form_defaults\(\)](#) (`indico.modules.vc.plugins.VCPluginMixin` method), 251
[get_verbose_title\(\)](#) (`indico.modules.events.models.events.Event` method), 82
[get_visibility_options\(\)](#) (in module `indico.modules.categories.util`), 209
[get_visible_reviewed_for_tracks\(\)](#) (in module `indico.modules.events.abstracts.util`), 113
[get_with_data\(\)](#) (`indico.modules.rb.models.reservations.Reservation` static method), 237
[get_with_data\(\)](#) (`indico.modules.rb.models.rooms.Room` static method), 228
[getBody\(\)](#) (`indico.modules.events.sessions.util.SessionListToPDF` method), 187
[global_abstract_reviewers](#) (`indico.modules.events.models.events.Event` attribute), 82
[global_conveners](#) (`indico.modules.events.models.events.Event` attribute), 82
[group](#) (`indico.modules.designer.placeholders.CategoryTitlePlaceholder` attribute), 258
[group](#) (`indico.modules.designer.placeholders.EventDatesPlaceholder` attribute), 255
[group](#) (`indico.modules.designer.placeholders.EventDescriptionPlaceholder` attribute), 255
[group](#) (`indico.modules.designer.placeholders.EventOrgTextPlaceholder` attribute), 255
[group](#) (`indico.modules.designer.placeholders.EventRoomPlaceholder` attribute), 258
[group](#) (`indico.modules.designer.placeholders.EventSpeakersPlaceholder` attribute), 259
[group](#) (`indico.modules.designer.placeholders.EventTitlePlaceholder` attribute), 258
[group](#) (`indico.modules.designer.placeholders.EventVenuePlaceholder` attribute), 258
[group](#) (`indico.modules.designer.placeholders.RegistrationTicketQRPlaceholder` attribute), 257
[group](#) (`indico.modules.events.models.reviews.ProposalReviewMixin` attribute), 91
[group](#) (`indico.modules.events.models.reviews.ProposalReviewMixin` attribute), 110
[group](#) (`indico.modules.events.papers.models.reviews.PaperReview` attribute), 147
[group_attr](#) (`indico.modules.events.papers.models.reviews.PaperReview` attribute), 147
[group_id](#) (`indico.modules.networks.models.networks.IPNetwork` attribute), 259
[group_proxy_cls](#) (`indico.modules.events.models.reviews.ProposalReviewM` attribute), 91
[group_proxy_cls](#) (`indico.modules.events.papers.models.reviews.PaperReview` attribute), 147
[GroupProxy](#) (class in `indico.modules.groups.core`), 247

H

[happens_between\(\)](#) (`indico.modules.events.models.events.Event` method), 82
[has_attribute\(\)](#) (`indico.modules.rb.models.rooms.Room` method), 228
[has_booking_groups](#) (`indico.modules.rb.models.rooms.Room` attribute), 228
[has_contributions_with_user_as_submitter\(\)](#) (in module `indico.modules.events.contributions.util`), 129
[has_contributions_with_user_paper_submission_rights\(\)](#) (in module `indico.modules.events.papers.util`), 151
[has_effective_icon](#) (`indico.modules.categories.models.categories.Category` attribute), 205
[has_ended](#) (`indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstracts` attribute), 103
[has_ended](#) (`indico.modules.events.papers.models.call_for_papers.CallForPapers` attribute), 141
[has_ended](#) (`indico.modules.events.registration.models.forms.RegistrationForm` attribute), 165
[has_ended](#) (`indico.modules.events.surveys.models.surveys.Survey` attribute), 188
[has_equipment\(\)](#) (`indico.modules.rb.models.rooms.Room` method), 228
[has_feature\(\)](#) (`indico.modules.events.models.events.Event` method), 82
[has_files](#) (`indico.modules.events.registration.models.registrations.Registration` attribute), 159
[has_icon](#) (`indico.modules.categories.models.categories.Category` attribute), 205
[has_live_reservations\(\)](#) (`indico.modules.rb.models.rooms.Room` method), 228
[has_logo](#) (`indico.modules.categories.models.categories.Category` attribute), 205

- has_logo (indico.modules.events.models.events.Event attribute), 82
- has_member() (indico.modules.groups.core.GroupProxy method), 247
- has_note (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 184
- has_only_events (indico.modules.categories.models.categories.Category attribute), 205
- has_photo (indico.modules.rb.models.rooms.Room attribute), 228
- has_projector (indico.modules.rb.models.rooms.Room attribute), 228
- has_role() (indico.modules.events.models.persons.EventPerson method), 85
- has_sessions_for_user() (in module indico.modules.events.sessions.util), 187
- has_special_name (indico.modules.rb.models.rooms.Room attribute), 228
- has_started (indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstracts attribute), 103
- has_started (indico.modules.events.papers.models.call_for_papers.CallForPapers attribute), 141
- has_started (indico.modules.events.registration.models.forms.RegistrationForm attribute), 165
- has_started (indico.modules.events.surveys.models.surveys.Survey attribute), 188
- has_stylesheet (indico.modules.events.models.events.Event attribute), 82
- has_user_reviewed() (in module indico.modules.events.papers.models.revisions.PaperRevision attribute), 148
- has_user_tracks() (in module indico.modules.events.abstracts.util), 113
- has_vc (indico.modules.rb.models.rooms.Room attribute), 228
- has_webcast_recording (in module indico.modules.rb.models.rooms.Room attribute), 228
- height (indico.modules.designer.pdf.TplData attribute), 254
- height_cm (indico.modules.designer.pdf.TplData attribute), 254
- hidden (indico.modules.networks.models.networks.IPNetworkGroup attribute), 259
- HiddenEnumField (class in indico.web.forms.fields), 273
- HiddenFieldList (class in indico.web.forms.fields), 268
- Holiday (class in indico.modules.rb.models.holidays), 234
- holidays (indico.modules.rb.models.locations.Location attribute), 235
- html (indico.modules.events.layout.models.menu.EventPage attribute), 131
- html (indico.modules.events.notes.models.notes.EventNote attribute), 139
- html (indico.modules.events.notes.models.notes.EventNoteRevision attribute), 140
- html_field_name (indico.modules.events.registration.models.form_fields.RegistrationForm attribute), 163
- html_block_name (indico.modules.events.registration.models.form_fields.RegistrationForm attribute), 164
- icon (indico.modules.categories.models.categories.Category attribute), 205
- icon_metadata (indico.modules.categories.models.categories.Category attribute), 205
- icon_url (indico.modules.categories.models.categories.Category attribute), 205
- icon_url (indico.modules.vc.plugins.VCPluginMixin attribute), 251
- id (indico.modules.attachments.models.attachments.Attachment attribute), 220
- id (indico.modules.attachments.models.attachments.AttachmentFile attribute), 221
- id (indico.modules.attachments.models.folders.AttachmentFolder attribute), 223
- id (indico.modules.attachments.models.principals.AttachmentFolderPrincipal attribute), 224
- id (indico.modules.attachments.models.principals.AttachmentPrincipal attribute), 224
- id (indico.modules.auth.models.identities.Identity attribute), 241
- id (indico.modules.auth.models.registration_requests.RegistrationRequest attribute), 242
- id (indico.modules.categories.models.categories.Category attribute), 205
- id (indico.modules.categories.models.principals.CategoryPrincipal attribute), 207
- id (indico.modules.categories.models.settings.CategorySetting attribute), 208
- id (indico.modules.designer.models.images.DesignerImageFile attribute), 253
- id (indico.modules.designer.models.templates.DesignerTemplate attribute), 253
- id (indico.modules.events.abstracts.models.abstracts.Abstract attribute), 101
- id (indico.modules.events.abstracts.models.comments.AbstractComment attribute), 104
- id (indico.modules.events.abstracts.models.email_logs.AbstractEmailLogEntry attribute), 105
- id (indico.modules.events.abstracts.models.email_templates.AbstractEmailTemplate attribute), 106
- id (indico.modules.events.abstracts.models.files.AbstractFile attribute), 107
- id (indico.modules.events.abstracts.models.persons.AbstractPersonLink attribute), 107
- id (indico.modules.events.abstracts.models.review_questions.AbstractReviewQuestion attribute), 108

- id (indico.modules.events.abstracts.models.review_ratings.AbstractReviewRating attribute), 109
- id (indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 110
- id (indico.modules.events.abstracts.settings.BOASortField attribute), 116
- id (indico.modules.events.agreements.models.agreements.Agreement attribute), 117
- id (indico.modules.events.contributions.models.contributions.Contribution attribute), 120
- id (indico.modules.events.contributions.models.fields.ContributionField attribute), 122
- id (indico.modules.events.contributions.models.persons.ContributionPersonLink attribute), 124
- id (indico.modules.events.contributions.models.persons.SubContributionPersonLink attribute), 125
- id (indico.modules.events.contributions.models.principals.ContributionPrincipal attribute), 125
- id (indico.modules.events.contributions.models.references.ContributionReferences attribute), 126
- id (indico.modules.events.contributions.models.references.SubContributionReferences attribute), 126
- id (indico.modules.events.contributions.models.subcontributions.SubContribution attribute), 127
- id (indico.modules.events.contributions.models.types.ContributionType attribute), 128
- id (indico.modules.events.layout.models.images.ImageFile attribute), 131
- id (indico.modules.events.layout.models.menu.EventPage attribute), 131
- id (indico.modules.events.layout.models.menu.MenuEntry attribute), 132
- id (indico.modules.events.layout.models.menu.TransientMenuEntry attribute), 133
- id (indico.modules.events.logs.models.entries.EventLogEntry attribute), 135
- id (indico.modules.events.models.events.Event attribute), 82
- id (indico.modules.events.models.persons.EventPerson attribute), 85
- id (indico.modules.events.models.persons.EventPersonLink attribute), 86
- id (indico.modules.events.models.persons.PersonLinkBase attribute), 87
- id (indico.modules.events.models.principals.EventPrincipal attribute), 87
- id (indico.modules.events.models.references.EventReference attribute), 88
- id (indico.modules.events.models.references.ReferenceModelBase attribute), 88
- id (indico.modules.events.models.references.ReferenceType attribute), 89
- id (indico.modules.events.models.series.EventSeries attribute), 92
- id (indico.modules.events.models.settings.EventSetting attribute), 92
- id (indico.modules.events.models.settings.EventSettingPrincipal attribute), 92
- id (indico.modules.events.models.static_list_links.StaticListLink attribute), 93
- id (indico.modules.events.notes.models.notes.EventNote attribute), 139
- id (indico.modules.events.notes.models.notes.EventNoteRevision attribute), 140
- id (indico.modules.events.papers.models.comments.PaperReviewComment attribute), 142
- id (indico.modules.events.papers.models.competences.PaperCompetence attribute), 143
- id (indico.modules.events.papers.models.files.PaperFile attribute), 143
- id (indico.modules.events.papers.models.review_questions.PaperReviewQuestion attribute), 145
- id (indico.modules.events.papers.models.review_ratings.PaperReviewRating attribute), 146
- id (indico.modules.events.papers.models.reviews.PaperReview attribute), 147
- id (indico.modules.events.papers.models.revisions.PaperRevision attribute), 148
- id (indico.modules.events.papers.models.templates.PaperTemplate attribute), 149
- id (indico.modules.events.payment.models.transactions.PaymentTransaction attribute), 154
- id (indico.modules.events.registration.models.form_fields.RegistrationForm attribute), 163
- id (indico.modules.events.registration.models.form_fields.RegistrationForm attribute), 163
- id (indico.modules.events.registration.models.form_fields.RegistrationForm attribute), 164
- id (indico.modules.events.registration.models.forms.RegistrationForm attribute), 165
- id (indico.modules.events.registration.models.invitations.RegistrationInvitation attribute), 167
- id (indico.modules.events.registration.models.items.RegistrationFormItem attribute), 169
- id (indico.modules.events.registration.models.items.RegistrationFormPerson attribute), 170
- id (indico.modules.events.registration.models.items.RegistrationFormSection attribute), 171
- id (indico.modules.events.registration.models.items.RegistrationFormText attribute), 172
- id (indico.modules.events.registration.models.registrations.Registration attribute), 159
- id (indico.modules.events.reminders.models.reminders.EventReminder attribute), 178
- id (indico.modules.events.requests.models.requests.Request attribute), 180
- id (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 185

- id (indico.modules.events.sessions.models.persons.SessionBlockPerson attribute), 237
- id (indico.modules.events.sessions.models.principals.SessionPrincipal attribute), 230
- id (indico.modules.events.sessions.models.sessions.Session attribute), 183
- id (indico.modules.events.static.models.static.StaticSite attribute), 203
- id (indico.modules.events.surveys.models.items.SurveyItem attribute), 190
- id (indico.modules.events.surveys.models.items.SurveyQuestion attribute), 191
- id (indico.modules.events.surveys.models.items.SurveySection attribute), 192
- id (indico.modules.events.surveys.models.items.SurveyText attribute), 192
- id (indico.modules.events.surveys.models.submissions.SurveySubmission attribute), 193
- id (indico.modules.events.surveys.models.surveys.Survey attribute), 188
- id (indico.modules.events.timetable.models.breaks.Break attribute), 195
- id (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 197
- id (indico.modules.events.tracks.models.tracks.Track attribute), 201
- id (indico.modules.groups.models.groups.LocalGroup attribute), 246
- id (indico.modules.networks.models.networks.IPNetworkGroup attribute), 259
- id (indico.modules.news.models.news.NewsItem attribute), 260
- id (indico.modules.oauth.models.applications.OAuthApplication attribute), 244
- id (indico.modules.oauth.models.tokens.OAuthToken attribute), 245
- id (indico.modules.rb.models.aspects.Aspect attribute), 231
- id (indico.modules.rb.models.blocked_rooms.BlockedRoom attribute), 232
- id (indico.modules.rb.models.blocking_principals.BlockingPrincipal attribute), 233
- id (indico.modules.rb.models.blockings.Blocking attribute), 232
- id (indico.modules.rb.models.equipment.EquipmentType attribute), 234
- id (indico.modules.rb.models.holidays.Holiday attribute), 234
- id (indico.modules.rb.models.locations.Location attribute), 235
- id (indico.modules.rb.models.photos.Photo attribute), 235
- id (indico.modules.rb.models.reservation_edit_logs.ReservationEditLog attribute), 238
- id (indico.modules.rb.models.reservations.Reservation attribute), 237
- id (indico.modules.rb.models.room_attributes.RoomAttribute attribute), 228
- id (indico.modules.rb.models.rooms.Room attribute), 228
- id (indico.modules.users.models.affiliations.UserAffiliation attribute), 215
- id (indico.modules.users.models.emails.UserEmail attribute), 215
- id (indico.modules.users.models.settings.UserSetting attribute), 216
- id (indico.modules.users.models.users.User attribute), 213
- id (indico.modules.vc.models.vc_rooms.VCRoom attribute), 248
- id (indico.modules.vc.models.vc_rooms.VCRoomEventAssociation attribute), 249
- id (indico.modules.auth.models.identities.Identity attribute), 241
- identifier (indico.modules.events.agreements.models.agreements.Agreement attribute), 117
- identities (indico.modules.users.models.users.User attribute), 213
- Identity (class in indico.modules.auth.models.identities), 241
- identity_data (indico.modules.auth.models.registration_requests.RegistrationRequest attribute), 242
- IDPlaceholder (class in indico.modules.events.registration.placeholders.registrations), 174
- IgnoredTransactionAction, 154
- image_created (in module indico.core.signals.event_management), 48
- image_deleted (in module indico.core.signals.event_management), 48
- ImageFile (class in indico.modules.events.layout.models.images), 131
- ImagePreviewer (class in indico.modules.attachments.preview), 226
- impersonate_user() (in module indico.modules.auth.util), 243
- import_tasks (in module indico.core.signals), 43
- in_attachment_acls (in indico.modules.users.models.users.User attribute), 213
- in_attachment_folder_acls (in indico.modules.users.models.users.User attribute), 213
- in_progress (indico.modules.events.abstracts.models.abstracts.AbstractReview attribute), 102
- include_authors (indico.modules.events.abstracts.models.email_templates.EmailTemplate attribute), 106
- include_coauthors (indico.modules.events.abstracts.models.email_templates.EmailTemplate attribute), 106

- include_plugin_css_assets() (in module indico.core.plugins), 42
- include_plugin_js_assets() (in module indico.core.plugins), 42
- include_submitter (indico.modules.events.abstracts.models.email_templates.AbstractEmailTemplate attribute), 106
- include_summary (indico.modules.events.reminders.models.related_tracks attribute), 178
- indico.core.plugins (module), 39
- indico.modules.attachments (module), 219
- indico.modules.attachments.models.attachments (module), 219
- indico.modules.attachments.models.folders (module), 222
- indico.modules.attachments.models.principals (module), 224
- indico.modules.attachments.operations (module), 225
- indico.modules.attachments.preview (module), 226
- indico.modules.attachments.util (module), 225
- indico.modules.auth (module), 241
- indico.modules.auth.models.identities (module), 241
- indico.modules.auth.models.registration_requests (module), 242
- indico.modules.auth.util (module), 243
- indico.modules.categories (module), 204
- indico.modules.categories.fields (module), 266
- indico.modules.categories.models.categories (module), 204
- indico.modules.categories.models.principals (module), 207
- indico.modules.categories.models.settings (module), 208
- indico.modules.categories.operations (module), 208
- indico.modules.categories.serialize (module), 209
- indico.modules.categories.settings (module), 210
- indico.modules.categories.util (module), 208
- indico.modules.designer (module), 252
- indico.modules.designer.models.images (module), 252
- indico.modules.designer.models.templates (module), 253
- indico.modules.designer.pdf (module), 254
- indico.modules.designer.placeholders (module), 255
- indico.modules.designer.util (module), 254
- indico.modules.events (module), 79
- indico.modules.events.abstracts (module), 99
- indico.modules.events.abstracts.fields (module), 262
- indico.modules.events.abstracts.models.abstracts (module), 100
- indico.modules.events.abstracts.models.call_for_abstracts (module), 103
- indico.modules.events.abstracts.models.comments (module), 104
- indico.modules.events.abstracts.models.email_logs (module), 105
- indico.modules.events.abstracts.models.email_templates (module), 105
- indico.modules.events.abstracts.models.fields (module), 106
- indico.modules.events.abstracts.models.files (module), 107
- indico.modules.events.abstracts.models.persons (module), 107
- indico.modules.events.abstracts.models.related_tracks (module), 108
- indico.modules.events.abstracts.models.review_questions (module), 108
- indico.modules.events.abstracts.models.review_ratings (module), 108
- indico.modules.events.abstracts.models.reviews (module), 109
- indico.modules.events.abstracts.operations (module), 111
- indico.modules.events.abstracts.placeholders (module), 113
- indico.modules.events.abstracts.settings (module), 116
- indico.modules.events.abstracts.util (module), 112
- indico.modules.events.agreements (module), 116
- indico.modules.events.agreements.models.agreements (module), 117
- indico.modules.events.agreements.placeholders (module), 119
- indico.modules.events.agreements.util (module), 118
- indico.modules.events.contributions (module), 119
- indico.modules.events.contributions.fields (module), 265
- indico.modules.events.contributions.models.contributions (module), 119
- indico.modules.events.contributions.models.fields (module), 122
- indico.modules.events.contributions.models.persons (module), 123
- indico.modules.events.contributions.models.principals (module), 125
- indico.modules.events.contributions.models.references (module), 126
- indico.modules.events.contributions.models.subcontributions (module), 126
- indico.modules.events.contributions.models.types (module), 127
- indico.modules.events.contributions.operations (module), 128
- indico.modules.events.contributions.util (module), 129
- indico.modules.events.features (module), 130
- indico.modules.events.features.util (module), 130
- indico.modules.events.fields (module), 261
- indico.modules.events.layout (module), 130
- indico.modules.events.layout.models.images (module), 131
- indico.modules.events.layout.models.menu (module), 131
- indico.modules.events.layout.util (module), 133
- indico.modules.events.logs (module), 135

- indico.modules.events.logs.models.entries (module), 135
- indico.modules.events.logs.renderers (module), 136
- indico.modules.events.logs.util (module), 136
- indico.modules.events.models.events (module), 79
- indico.modules.events.models.persons (module), 85
- indico.modules.events.models.principals (module), 87
- indico.modules.events.models.references (module), 88
- indico.modules.events.models.reviews (module), 89
- indico.modules.events.models.series (module), 91
- indico.modules.events.models.settings (module), 92
- indico.modules.events.models.static_list_links (module), 93
- indico.modules.events.notes (module), 138
- indico.modules.events.notes.models.notes (module), 138
- indico.modules.events.notes.util (module), 140
- indico.modules.events.operations (module), 94
- indico.modules.events.papers (module), 140
- indico.modules.events.papers.fields (module), 265
- indico.modules.events.papers.models.call_for_papers (module), 141
- indico.modules.events.papers.models.comments (module), 141
- indico.modules.events.papers.models.competences (module), 142
- indico.modules.events.papers.models.files (module), 143
- indico.modules.events.papers.models.papers (module), 144
- indico.modules.events.papers.models.review_questions (module), 145
- indico.modules.events.papers.models.review_ratings (module), 145
- indico.modules.events.papers.models.reviews (module), 146
- indico.modules.events.papers.models.revisions (module), 148
- indico.modules.events.papers.models.templates (module), 149
- indico.modules.events.papers.models.user_contributions (module), 150
- indico.modules.events.papers.operations (module), 150
- indico.modules.events.papers.util (module), 151
- indico.modules.events.payment (module), 154
- indico.modules.events.payment.models.transactions (module), 154
- indico.modules.events.payment.plugins (module), 156
- indico.modules.events.payment.util (module), 156
- indico.modules.events.persons (module), 157
- indico.modules.events.persons.operations (module), 157
- indico.modules.events.persons.placeholders (module), 157
- indico.modules.events.registration (module), 158
- indico.modules.events.registration.models.form_fields (module), 162
- indico.modules.events.registration.models.forms (module), 164
- indico.modules.events.registration.models.invitations (module), 167
- indico.modules.events.registration.models.items (module), 168
- indico.modules.events.registration.models.registrations (module), 158
- indico.modules.events.registration.placeholders.invitations (module), 175
- indico.modules.events.registration.placeholders.registrations (module), 174
- indico.modules.events.registration.settings (module), 175
- indico.modules.events.registration.stats (module), 176
- indico.modules.events.registration.util (module), 172
- indico.modules.events.reminders (module), 177
- indico.modules.events.reminders.models.reminders (module), 177
- indico.modules.events.reminders.util (module), 178
- indico.modules.events.requests (module), 179
- indico.modules.events.requests.base (module), 180
- indico.modules.events.requests.models.requests (module), 179
- indico.modules.events.requests.util (module), 180
- indico.modules.events.sessions (module), 182
- indico.modules.events.sessions.fields (module), 266
- indico.modules.events.sessions.models.blocks (module), 184
- indico.modules.events.sessions.models.persons (module), 185
- indico.modules.events.sessions.models.principals (module), 186
- indico.modules.events.sessions.models.sessions (module), 182
- indico.modules.events.sessions.operations (module), 187
- indico.modules.events.sessions.util (module), 187
- indico.modules.events.settings (module), 97, 151
- indico.modules.events.static (module), 202
- indico.modules.events.static.models.static (module), 202
- indico.modules.events.static.util (module), 203
- indico.modules.events.surveys (module), 188
- indico.modules.events.surveys.models.items (module), 190
- indico.modules.events.surveys.models.submissions (module), 192
- indico.modules.events.surveys.models.surveys (module), 188
- indico.modules.events.surveys.operations (module), 194
- indico.modules.events.surveys.util (module), 194
- indico.modules.events.timetable (module), 195
- indico.modules.events.timetable.models.breaks (module), 195
- indico.modules.events.timetable.models.entries (module), 196

- indico.modules.events.timetable.operations (module), 198
- indico.modules.events.timetable.reschedule (module), 200
- indico.modules.events.timetable.util (module), 199
- indico.modules.events.tracks (module), 201
- indico.modules.events.tracks.models.abstract_reviewers (module), 202
- indico.modules.events.tracks.models.conveners (module), 202
- indico.modules.events.tracks.models.tracks (module), 201
- indico.modules.events.tracks.operations (module), 202
- indico.modules.events.util (module), 95
- indico.modules.groups (module), 246
- indico.modules.groups.core (module), 247
- indico.modules.groups.models.groups (module), 246
- indico.modules.groups.util (module), 248
- indico.modules.networks (module), 259
- indico.modules.networks.fields (module), 267
- indico.modules.networks.models.networks (module), 259
- indico.modules.networks.util (module), 260
- indico.modules.news (module), 260
- indico.modules.news.models.news (module), 260
- indico.modules.news.util (module), 261
- indico.modules.oauth (module), 243
- indico.modules.oauth.models.applications (module), 244
- indico.modules.oauth.models.tokens (module), 245
- indico.modules.oauth.provider (module), 246
- indico.modules.rb (module), 227
- indico.modules.rb.models.aspects (module), 231
- indico.modules.rb.models.blocked_rooms (module), 232
- indico.modules.rb.models.blocking_principals (module), 233
- indico.modules.rb.models.blockings (module), 232
- indico.modules.rb.models.equipment (module), 233
- indico.modules.rb.models.holidays (module), 234
- indico.modules.rb.models.locations (module), 234
- indico.modules.rb.models.photos (module), 235
- indico.modules.rb.models.reservation_edit_logs (module), 238
- indico.modules.rb.models.reservation_occurrences (module), 239
- indico.modules.rb.models.reservations (module), 235
- indico.modules.rb.models.room_attributes (module), 230
- indico.modules.rb.models.room_bookable_hours (module), 231
- indico.modules.rb.models.room_nonbookable_periods (module), 231
- indico.modules.rb.models.rooms (module), 227
- indico.modules.rb.models.util (module), 239
- indico.modules.rb.services.aspects (module), 241
- indico.modules.rb.services.blockings (module), 241
- indico.modules.rb.services.rooms (module), 240
- indico.modules.rb.statistics (module), 240
- indico.modules.rb.util (module), 240
- indico.modules.users (module), 211
- indico.modules.users.ext (module), 219
- indico.modules.users.models.affiliations (module), 215
- indico.modules.users.models.emails (module), 215
- indico.modules.users.models.favorites (module), 215
- indico.modules.users.models.settings (module), 216
- indico.modules.users.models.suggestions (module), 215
- indico.modules.users.models.users (module), 211
- indico.modules.users.operations (module), 217
- indico.modules.users.util (module), 218
- indico.modules.vc (module), 248
- indico.modules.vc.exceptions (module), 252
- indico.modules.vc.models.vc_rooms (module), 248
- indico.modules.vc.plugins (module), 250
- indico.modules.vc.util (module), 250
- indico.web.forms.fields (module), 267
- IndicoDateField (class in indico.web.forms.fields), 276
- IndicoDateTimeField (class in indico.web.forms.fields), 272
- IndicoEmailRecipientsField (class in indico.web.forms.fields), 277
- IndicoEnumRadioField (class in indico.web.forms.fields), 273
- IndicoEnumSelectField (class in indico.web.forms.fields), 273
- IndicoLocationField (class in indico.web.forms.fields), 275
- IndicoMarkdownField (class in indico.web.forms.fields), 275
- IndicoPalettePickerField (class in indico.web.forms.fields), 271
- IndicoPasswordField (class in indico.web.forms.fields), 270
- IndicoPlugin (class in indico.core.plugins), 39
- IndicoPluginBlueprint (class in indico.core.plugins), 41
- IndicoProtectionField (class in indico.web.forms.fields), 276
- IndicoQuerySelectMultipleField (class in indico.web.forms.fields), 275
- IndicoRadioField (class in indico.web.forms.fields), 267
- IndicoSelectMultipleCheckboxBooleanField (class in indico.web.forms.fields), 276
- IndicoSelectMultipleCheckboxField (class in indico.web.forms.fields), 267
- IndicoStaticTextField (class in indico.web.forms.fields), 270
- IndicoTagListField (class in indico.web.forms.fields), 271
- IndicoThemeSelectField (class in indico.modules.events.fields), 261
- IndicoTimezoneSelectField (class in indico.web.forms.fields), 273

IndicoWeekDayRepetitionField (class in indico.web.forms.fields), 276

info (indico.modules.categories.models.categories.EventMessageModel attribute), 207

info (indico.modules.rb.models.reservation_edit_logs.ReservationEditLog attribute), 238

inherit_location (indico.modules.events.contributions.models.contributions.Contribution attribute), 120

inherit_location (indico.modules.events.models.events.Event attribute), 82

inherit_location (indico.modules.events.sessions.models.blocking_sessions.BlockingSession attribute), 185

inherit_location (indico.modules.events.sessions.models.sessions.Session attribute), 183

inherit_location (indico.modules.events.timetable.models.breaks.Break attribute), 195

inheriting_have_acl (indico.modules.categories.models.categories.Category attribute), 205

inheriting_have_acl (indico.modules.events.contributions.models.contributions.Contribution attribute), 120

inheriting_have_acl (indico.modules.events.models.events.Event attribute), 82

inheriting_have_acl (indico.modules.events.sessions.models.sessions.Session attribute), 183

init() (indico.core.plugins.IndicoPlugin method), 40

init() (indico.modules.events.payment.plugins.PaymentPluginMixin method), 157

init() (indico.modules.vc.plugins.VCPluginMixin method), 251

initial_statuses (indico.modules.events.payment.models.transactions.Transaction attribute), 155

inject_css (in module indico.core.signals.plugin), 48

inject_css() (indico.core.plugins.IndicoPlugin method), 40

inject_js (in module indico.core.signals.plugin), 48

inject_js() (indico.core.plugins.IndicoPlugin method), 40

inject_vars_js() (indico.core.plugins.IndicoPlugin method), 40

input_type (indico.modules.events.registration.models.form_fields.RegistrationFormField attribute), 163

input_type (indico.modules.events.registration.models.form_fields.RegistrationFormCategoryDataField attribute), 164

input_type (indico.modules.events.registration.models.items.RegistrationItem attribute), 169

input_type (indico.modules.events.registration.models.items.RegistrationFormPersonalDataSection attribute), 170

input_type (indico.modules.events.registration.models.items.RegistrationFormSelection attribute), 171

input_type (indico.modules.events.registration.models.items.RegistrationFormText attribute), 172

insert() (indico.modules.events.layout.models.menu.MenuEntry method), 132

insertion_link (indico.modules.events.layout.models.menu.MenuEntryType attribute), 133

insertion_link (indico.modules.events.registration.models.forms.RegistrationForm attribute), 165

insertion_link (indico.modules.events.surveys.models.surveys.Survey attribute), 188

InvalidManualTransactionAction, 154

InvalidTransactionAction, 154

InvitationLinkStatus, 154

InvitationLinkPlaceholder (class in indico.modules.events.registration.placeholders.invitations), 175

InvitationState (class in indico.modules.events.registration.models.forms.RegistrationForm attribute), 165

invited_dt (indico.modules.events.models.persons.EventPerson attribute), 85

ip_network_group (indico.modules.attachments.models.principals.Attachment attribute), 224

ip_network_group (indico.modules.attachments.models.principals.Attachment attribute), 224

ip_network_group (indico.modules.categories.models.principals.CategoryPrincipal attribute), 207

ip_network_group (indico.modules.events.contributions.models.principals.Contribution attribute), 125

ip_network_group (indico.modules.events.models.principals.EventPrincipal attribute), 87

ip_network_group (indico.modules.events.models.settings.EventSettingPrincipal attribute), 92

ip_network_group (indico.modules.events.sessions.models.principals.Session attribute), 186

ip_network_group (indico.modules.rb.models.blocking_principals.BlockingPrincipal attribute), 233

ip_network_group_id (indico.modules.attachments.models.principals.AttachmentFolderPrincipal attribute), 224

ip_network_group_id (indico.modules.attachments.models.principals.AttachmentPrincipal attribute), 224

ip_network_group_id (indico.modules.categories.models.principals.CategoryPrincipal attribute), 207

ip_network_group_id (indico.modules.events.contributions.models.principals.Contribution attribute), 125

ip_network_group_id (indico.modules.events.models.principals.EventPrincipal attribute), 87

ip_network_group_id (indico.modules.events.models.settings.EventSettingPrincipal attribute), 92

attribute), 92

ip_network_group_id (in-dico.modules.events.sessions.models.principals.SessionPrincipal attribute), 177

ip_network_group_id (in-dico.modules.events.sessions.models.principals.SessionPrincipal attribute), 186

ip_network_group_id (in-dico.modules.rb.models.blocking_principals.BlockingPrincipal attribute), 233

IPNetwork (class in in-dico.modules.networks.models.networks), 259

IPNetworkGroup (class in in-dico.modules.networks.models.networks), 259

is_accepted (indico.modules.rb.models.reservations.Reservation attribute), 237

is_active (indico.modules.events.contributions.models.fields.ContributionField attribute), 122

is_active (indico.modules.events.registration.models.forms.RegistrationForm attribute), 165

is_active (indico.modules.events.registration.models.registration.Registration attribute), 159

is_active (indico.modules.events.surveys.models.surveys.Survey attribute), 188

is_active (indico.modules.rb.models.rooms.Room attribute), 228

is_active_at() (indico.modules.rb.models.blockings.Blockings_deleted method), 232

is_admin (indico.modules.users.models.users.User attribute), 213

is_always_visible (indico.modules.attachments.models.folders.AttachmentFolder attribute), 223

is_anonymous (indico.modules.events.surveys.models.submissions.SurveySubmission attribute), 193

is_archived (indico.modules.rb.models.reservations.Reservation attribute), 237

is_author (indico.modules.events.contributions.models.persons.ControlPersonLink attribute), 124

is_auto_confirm (indico.modules.rb.models.rooms.Room attribute), 228

is_blocked (indico.modules.users.models.users.User attribute), 213

is_booked_for() (indico.modules.rb.models.reservations.Reservation method), 237

is_cancelled (indico.modules.events.registration.models.registrations.Registration attribute), 159

is_cancelled (indico.modules.rb.models.reservation_occurrences.ReservationOccurrence attribute), 239

is_cancelled (indico.modules.rb.models.reservations.Reservation attribute), 237

is_clonable (indico.modules.designer.models.templates.DesignTemplate attribute), 253

is_currency_shown (in-dico.modules.events.registration.stats.FieldStats attribute), 177

is_currency_shown (in-dico.modules.events.registration.stats.StatsBase attribute), 177

is_default (indico.modules.attachments.models.folders.AttachmentFolder attribute), 223

is_deleted (indico.modules.events.layout.models.menu.EventPage attribute), 132

is_deleted (indico.modules.rb.models.locations.Location attribute), 235

is_deleted (indico.modules.attachments.models.attachments.Attachment attribute), 220

is_deleted (indico.modules.attachments.models.folders.AttachmentFolder attribute), 223

is_deleted (indico.modules.categories.models.categories.Category attribute), 205

is_deleted (indico.modules.events.abstracts.models.abstracts.Abstract attribute), 101

is_deleted (indico.modules.events.abstracts.models.comments.AbstractComment attribute), 104

is_deleted (indico.modules.events.abstracts.models.review_questions.AbstractReviewQuestion attribute), 108

is_deleted (indico.modules.events.contributions.models.contributions.Contribution attribute), 120

is_deleted (indico.modules.events.contributions.models.subcontributions.Subcontribution attribute), 127

is_deleted (indico.modules.events.models.events.Event attribute), 82

is_deleted (indico.modules.events.notes.models.notes.EventNote attribute), 139

is_deleted (indico.modules.events.papers.models.comments.PaperReviewComment attribute), 142

is_deleted (indico.modules.events.papers.models.review_questions.PaperReviewQuestion attribute), 145

is_deleted (indico.modules.events.registration.models.form_fields.RegistrationFormField attribute), 163

is_deleted (indico.modules.events.registration.models.form_fields.RegistrationFormField attribute), 164

is_deleted (indico.modules.events.registration.models.forms.RegistrationForm attribute), 165

is_deleted (indico.modules.events.registration.models.items.RegistrationFormItem attribute), 169

is_deleted (indico.modules.events.registration.models.items.RegistrationFormItem attribute), 170

is_deleted (indico.modules.events.registration.models.items.RegistrationFormItem attribute), 171

is_deleted (indico.modules.events.registration.models.items.RegistrationFormItem attribute), 172

is_deleted (indico.modules.events.registration.models.registrations.Registration attribute), 159

is_deleted (indico.modules.events.sessions.models.sessions.Session attribute), 183

is_deleted (indico.modules.events.surveys.models.surveys.Survey attribute), 188

is_deleted (indico.modules.users.models.users.User attribute), 213

- tribute), 213
- is_descendant_of() (in- dico.modules.categories.models.categories.CategoriesManager method), 205
- is_empty (indico.modules.categories.models.categories.CategoriesManager attribute), 205
- is_empty (indico.modules.events.surveys.models.submissions.SurveyAnswer attribute), 193
- is_enabled (indico.modules.events.layout.models.menu.MenuEntry attribute), 132
- is_enabled (indico.modules.events.registration.models.form_fields.RegistrationForm attribute), 163
- is_enabled (indico.modules.events.registration.models.form_fields.RegistrationForm attribute), 164
- is_enabled (indico.modules.events.registration.models.items.RegistrationForm attribute), 169
- is_enabled (indico.modules.events.registration.models.items.RegistrationForm attribute), 170
- is_enabled (indico.modules.events.registration.models.items.RegistrationForm attribute), 171
- is_enabled (indico.modules.events.registration.models.items.RegistrationForm attribute), 172
- is_enabled (indico.modules.events.registration.models.items.RegistrationForm attribute), 155
- is_enabled (indico.modules.events.registration.models.items.RegistrationForm attribute), 235
- is_enabled (indico.modules.events.registration.models.items.RegistrationForm attribute) (in module indico.modules.events.layout.util), 135
- is_enabled (indico.modules.events.registration.models.items.RegistrationForm attribute), 172
- is_enabled (indico.modules.oauth.models.applications.OAuthApplication attribute), 244
- is_feature_enabled() (in module indico.modules.events.features.util), 130
- is_field (indico.modules.events.registration.models.items.RegistrationForm attribute), 169
- is_group (indico.modules.groups.core.GroupProxy attribute), 247
- is_group (indico.modules.networks.models.networks.IPNetworkGroup attribute), 259
- is_group (indico.modules.users.models.users.User attribute), 213
- is_group (indico.modules.users.models.users.User attribute), 213
- is_hidden (indico.modules.rb.models.room_attributes.RoomAttribute attribute), 230
- is_ignored (indico.modules.users.models.suggestions.SuggestedOpen attribute), 216
- is_in_final_state (indico.modules.events.abstracts.models.abstracts.Attribute attribute), 101
- is_in_final_state (indico.modules.events.models.reviews.Proposal attribute), 90
- is_in_final_state (indico.modules.events.papers.models.papers.Paper attribute), 144
- is_internal_link (indico.modules.events.layout.models.menu.MenuEntry attribute), 133
- is_judge() (indico.modules.events.papers.models.call_for_papers.CallForPapers method), 141
- is_last_revision (indico.modules.events.papers.models.revisions.Revision attribute), 148
- is_link (indico.modules.events.layout.models.menu.MenuEntry attribute), 133
- is_locked (indico.modules.events.models.events.Event attribute), 82
- is_manager() (indico.modules.events.papers.models.call_for_papers.CallForPapers method), 141
- is_manager_only (indico.modules.events.registration.models.form_fields.RegistrationForm attribute), 163
- is_manager_only (indico.modules.events.registration.models.form_fields.RegistrationForm attribute), 164
- is_manager_only (indico.modules.events.registration.models.items.RegistrationForm attribute), 169
- is_manager_only (indico.modules.events.registration.models.items.RegistrationForm attribute), 170
- is_manager_only (indico.modules.events.registration.models.items.RegistrationForm attribute), 171
- is_manager_only (indico.modules.events.registration.models.items.RegistrationForm attribute), 172
- is_manager_only (indico.modules.events.registration.models.items.RegistrationForm attribute), 155
- is_manager_only (indico.modules.events.registration.models.items.RegistrationForm attribute), 235
- is_manager_only (indico.modules.events.registration.models.items.RegistrationForm attribute) (in module indico.modules.events.layout.util), 135
- is_manager_only (indico.modules.events.registration.models.items.RegistrationForm attribute), 165
- is_modification_open (in- dico.modules.events.registration.models.forms.RegistrationForm attribute), 165
- is_network (indico.modules.networks.models.networks.IPNetworkGroup attribute), 259
- is_network (indico.modules.users.models.users.User attribute), 213
- is_open (indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstracts attribute), 103
- is_open (indico.modules.events.papers.models.call_for_papers.CallForPapers attribute), 141
- is_open (indico.modules.events.registration.models.forms.RegistrationForm attribute), 165
- is_open (indico.modules.events.agreements.models.agreements.Agreement attribute), 117
- is_open (indico.modules.events.layout.models.menu.MenuEntry attribute), 133
- is_open (indico.modules.events.reminders.models.reminders.EventReminder attribute), 178
- is_open (indico.modules.rb.models.reservations.Reservation attribute), 237
- is_open (indico.modules.rb.models.rooms.Room attribute), 228
- is_open (indico.modules.events.layout.models.menu.MenuEntry attribute), 133
- is_open (indico.modules.events.registration.models.registrations.RegistrationForm attribute), 159
- is_paper_reviewer() (in- dico.modules.events.contributions.models.contributions.Contribution attribute), 159

method), 120

is_parallel() (indico.modules.events.timetable.models.entries.TimeTableEntry attribute), 192

is_parallel() (indico.modules.events.timetable.models.entries.TimeTableEntry method), 197

is_participation (indico.modules.events.registration.models.forms.RegistrationForm attribute), 166

is_pending (indico.modules.rb.models.reservations.Reservation attribute), 237

is_pending (indico.modules.users.models.users.User attribute), 213

is_plugin_link (indico.modules.events.layout.models.menu.MenuEntry attribute), 133

is_poster (indico.modules.events.sessions.models.sessions.Session attribute), 183

is_previewable (indico.modules.attachments.models.attachments.Attachments attribute), 221

is_primary (indico.modules.users.models.emails.UserEmail attribute), 215

is_protected (indico.modules.events.contributions.models.subcontributions.SubContribution attribute), 127

is_public (indico.modules.rb.models.rooms.Room attribute), 228

is_rejected (indico.modules.rb.models.reservation_occurrences.ReservationOccurrences attribute), 239

is_rejected (indico.modules.rb.models.reservations.Reservations attribute), 237

is_relative (indico.modules.events.reminders.models.reminders.EventReminder attribute), 178

is_repeating (indico.modules.rb.models.reservations.Reservations attribute), 237

is_request_manager() (in module indico.modules.events.requests.util), 180

is_required (indico.modules.events.contributions.models.fields.ContributionFields attribute), 122

is_required (indico.modules.events.registration.models.forms.RegistrationForm attribute), 163

is_required (indico.modules.events.registration.models.forms.RegistrationForm attribute), 164

is_required (indico.modules.events.registration.models.items.PersonalDataField attribute), 168

is_required (indico.modules.events.registration.models.items.RegistrationFormItem attribute), 169

is_required (indico.modules.events.registration.models.items.RegistrationFormPersonalDataField attribute), 170

is_required (indico.modules.events.registration.models.items.RegistrationFormSection attribute), 171

is_required (indico.modules.events.registration.models.items.RegistrationFormText attribute), 172

is_required (indico.modules.events.surveys.models.items.SurveyItem attribute), 190

is_required (indico.modules.events.surveys.models.items.SurveyQuestion attribute), 191

is_required (indico.modules.events.surveys.models.items.SurveySection attribute), 192

is_required (indico.modules.events.surveys.models.items.SurveyText attribute), 192

is_reviewer() (indico.modules.events.papers.models.call_for_papers.CallForPapers attribute), 141

is_root (indico.modules.categories.models.categories.Category attribute), 205

is_scheduled (indico.modules.events.contributions.models.contributions.Contribution attribute), 120

is_scheduled (indico.modules.events.registration.models.forms.RegistrationForm attribute), 166

is_section (indico.modules.events.registration.models.items.RegistrationFormItem attribute), 169

is_subscription (indico.modules.reminders.models.reminders.EventReminder attribute), 178

is_separator (indico.modules.events.layout.models.menu.MenuEntryMixin attribute), 133

is_single_person (indico.modules.groups.core.GroupProxy attribute), 247

is_single_person (indico.modules.users.models.users.User attribute), 213

is_speaker (indico.modules.events.abstracts.models.persons.AbstractPerson attribute), 107

is_speaker (indico.modules.events.contributions.models.persons.ContributionPerson attribute), 124

is_subcontributor (indico.modules.events.contributions.models.persons.SubContributionPerson attribute), 125

is_subcontributor (indico.modules.events.registration.models.call_for_papers.CallForPapers attribute), 141

is_subcontributor (indico.modules.events.registration.models.call_for_papers.CallForPapers method), 141

is_subcontributor (indico.modules.events.registration.models.call_for_papers.CallForPapers module in indico.modules.events.surveys.util), 194

is_submitted (indico.modules.events.surveys.models.submissions.SurveySubmission attribute), 193

is_submitted (indico.modules.events.contributions.models.persons.ContributionPerson attribute), 124

is_submitted (indico.modules.events.contributions.models.persons.EventPersonLink attribute), 86

is_system (indico.modules.users.models.users.User attribute), 213

is_system (indico.modules.designer.models.templates.DesignerTemplate attribute), 253

is_ticketing_handled (in module indico.core.signals.event), 46

is_trusted (indico.modules.oauth.models.applications.OAuthApplication attribute), 244

is_untrusted (indico.modules.events.models.persons.EventPerson attribute), 85

- is_user_associated() (in- judge_id (indico.modules.events.papers.models.revisions.PaperRevision
dico.modules.events.contributions.models.contributions.Contributor), 148
method), 120 judge_paper() (in module in-
- is_user_deleted (indico.modules.users.models.emails.UserEmail dico.modules.events.papers.operations), 150
attribute), 215 judged_abstracts (indico.modules.users.models.users.User
attribute), 213
- is_user_link (indico.modules.events.layout.models.menu.MenuEntryMix attribute), 213
attribute), 133 judges (indico.modules.events.abstracts.models.reviews.AbstractCommentV
attribute), 169
- is_valid (indico.modules.rb.models.reservation_occurrences.Reservation attribute), 169
attribute), 239 judges (indico.modules.events.papers.models.call_for_papers.CallForPapers
attribute), 141
- is_valid (indico.modules.rb.models.reservations.Reservation attribute), 141
attribute), 237 judges (indico.modules.events.papers.models.reviews.PaperCommentVisibi
attribute), 146
- is_visible (indico.modules.events.layout.models.menu.MenuEntryMix attribute), 146
attribute), 133 judgment_comment (in-
- is_visible (indico.modules.events.registration.models.items.Registration attribute), 144
attribute), 169 judgment_comment (in-
attribute), 101 dico.modules.events.abstracts.models.abstracts.Abstract
attribute), 101
- is_visible (indico.modules.events.surveys.models.surveys.Survey judgment_comment (in-
attribute), 188 dico.modules.events.papers.models.papers.Paper
attribute), 144
- is_visible_in() (indico.modules.events.models.events.Event attribute), 144
class method), 82 judgment_comment (in-
- items (in module indico.core.signals.menu), 48 dico.modules.events.papers.models.revisions.PaperRevision
attribute), 148
- items (indico.modules.designer.pdf.TplData attribute),
254 judgment_dt (indico.modules.events.abstracts.models.abstracts.Abstract
attribute), 101
- items (indico.modules.events.surveys.models.surveys.Survey attribute), 101
attribute), 188 judgment_dt (indico.modules.events.papers.models.revisions.PaperRevision
attribute), 148
- iter_choices() (indico.web.forms.fields.IndicoEnumSelectField attribute), 148
method), 273 judgment_instructions (in-
- iter_choices() (indico.web.forms.fields.IndicoSelectMultipleCheckbox attribute), 103
method), 276
- iter_create_occurrences() (in- JudgmentCommentPlaceholder (class in in-
dico.modules.rb.models.reservation_occurrences.ReservationOccurrence), 115
class method), 239
- iter_days() (indico.modules.events.models.events.Event K
method), 82 key (indico.modules.oauth.models.tokens.OAuthGrant
attribute), 245
- iter_identifiers() (indico.modules.users.models.users.User key_registration_info (indico.modules.rooms.Room attribute), 228
method), 213
- iter_param_info() (indico.modules.events.registration.placeholders.registration_info (indico.modules.rooms.Room attribute), 228
class method), 174
- iter_start_time() (indico.modules.rb.models.reservation_occurrences.ReservationOccurrence), 120
static method), 239 keywords (indico.modules.events.models.events.Event
attribute), 82
- J** kind (indico.modules.events.logs.models.entries.EventLogEntry
attribute), 135
- JSONField (class in indico.web.forms.fields), 267 kind (indico.modules.rb.models.rooms.Room attribute),
228
- judge (indico.modules.events.abstracts.models.abstracts.Abstract attribute), 101
- judge (indico.modules.events.papers.models.revisions.PaperRevision attribute), 148
- judge_abstract() (in module in- L
dico.modules.events.abstracts.operations),
111 large_photo_url (indico.modules.rb.models.rooms.Room
attribute), 228
- judge_deadline (indico.modules.events.papers.models.call_for_papers.CallForPapers attribute), 211
attribute), 141
- judge_id (indico.modules.events.abstracts.models.abstracts.Abstract attribute), 101
attribute), 211

last_first (indico.modules.users.models.users.NameFormat legacy_id (indico.modules.events.contributions.models.fields.ContributionF attribute), 211 attribute), 122

last_first_upper (indico.modules.users.models.users.NameFormat legacy_name (indico.modules.events.models.events.EventType attribute), 211 attribute), 85

last_login_dt (indico.modules.auth.models.identities.Identity limit_reached (indico.modules.events.registration.models.forms.Registration attribute), 241 attribute), 166

last_login_ip (indico.modules.auth.models.identities.Identity link (indico.modules.attachments.models.attachments.AttachmentType attribute), 241 attribute), 222

last_name (indico.modules.events.models.persons.EventPerson link_backref_lazy (indico.modules.attachments.models.folders.AttachmentF attribute), 85 attribute), 223

last_name (indico.modules.events.models.persons.Person LinkBack link_backref_name (in- dico.modules.attachments.models.folders.AttachmentFolder attribute), 87 attribute), 223

last_name (indico.modules.events.registration.models.invitations.Regist invitation (indico.modules.events.registration.models.invitations.Regist attribute), 167 attribute), 223

last_name (indico.modules.events.registration.models.items.Personal Data Type modules.events.notes.models.notes.EventNote attribute), 168 attribute), 139

last_name (indico.modules.events.registration.models.registrations.Registratio (indico modules.vc.models.vc_rooms.VCRoomEventAssociatio attribute), 160 attribute), 249

last_name (indico.modules.users.models.users.User at- link_type (indico.modules.attachments.models.folders.AttachmentFolder attribute), 213 attribute), 223

last_revision (indico.modules.events.papers.models.papers.Paper Paper type (indico.modules.events.notes.models.notes.EventNote attribute), 144 attribute), 139

last_used_dt (indico.modules.events.models.static_list_links.StaticListLink StaticListLink (indico.modules.vc.models.vc_rooms.VCRoomEventAssociatio attribute), 93 attribute), 249

last_used_dt (indico.modules.oauth.models.tokens.OAuthToken link_url (indico.modules.attachments.models.attachments.Attachment attribute), 245 attribute), 220

LastNamePlaceholder (class in in- link_url (indico.modules.events.layout.models.menu.MenuEntry dico.modules.events.persons.placeholders), attribute), 158 attribute), 132

LastNamePlaceholder (class in in- link_user_by_email() (in- dico.modules.events.models.persons.EventPerson attribute), 175 class method), 85

LastNamePlaceholder (class in in- linked_block (indico.modules.vc.models.vc_rooms.VCRoomEventAssociatio attribute), 175 attribute), 249

latest_dt (indico.web.forms.fields.IndicoDateTimeField linked_datetime_validator (in- dico.web.forms.fields.IndicoDateTimeField attribute), 272 attribute), 272

latitude (indico.modules.rb.models.rooms.Room at- attribute), 228 attribute), 272

layout (indico.modules.events.papers.models.reviews.PaperReviewType pat attribute), 147 attribute), 223

layout_review_questions (in- linked_event (indico.modules.events.notes.models.notes.EventNote attribute), 141 attribute), 139

layout_reviewer_deadline (in- linked_event (indico.modules.vc.models.vc_rooms.VCRoomEventAssociatio attribute), 141 attribute), 249

layout_reviewers (indico.modules.events.papers.models.call_for_papers.CallForPaper linked_event_id (indico.modules.attachments.models.folders.AttachmentFo attribute), 141 attribute), 223

layout_reviewing_enabled (in- linked_event_id (indico.modules.events.notes.models.notes.EventNote attribute), 141 attribute), 249

lecture (indico.modules.events.models.events.EventType linked_event_id (indico.modules.vc.models.vc_rooms.VCRoomEventAssoc attribute), 84 attribute), 249

lecture (indico.modules.events.models.events.EventType linked_object_attr (indico.modules.events.abstracts.fields.AbstractPersonLi attribute), 84 attribute), 262

- linked_object_attr (indico.modules.events.contributions.fields.ContributionPrincipalLinkListField attribute), 265
- linked_object_attr (indico.modules.events.contributions.fields.SubContributionPrincipalLinkListField attribute), 265
- linked_object_attr (indico.modules.events.fields.EventPersonLinkListField attribute), 186
- linked_object_attr (indico.modules.events.fields.PersonLinkListField attribute), 233
- linked_object_attr (indico.modules.events.sessions.fields.SessionBlockingPrincipalLinkListField attribute), 266
- LinkPlaceholder (class in indico.modules.events.registration.placeholders.registration_group), 175
- list_link_type (indico.modules.events.util.ListGeneratorBase attribute), 95
- ListGeneratorBase (class in indico.modules.events.util), 95
- load() (indico.modules.events.models.static_list_links.StaticListLink class method), 93
- load() (indico.modules.users.ext.ExtraUserPreferences method), 219
- load_client() (in module indico.modules.oauth.provider), 246
- load_grant() (in module indico.modules.oauth.provider), 246
- load_identity_info() (in module indico.modules.auth.util), 243
- load_token() (in module indico.modules.oauth.provider), 246
- local_group (indico.modules.attachments.models.principals.AttachmentFieldPrincipal attribute), 224
- local_group (indico.modules.attachments.models.principals.AttachmentPrincipal attribute), 225
- local_group (indico.modules.categories.models.principals.CategoryPrincipal attribute), 207
- local_group (indico.modules.events.contributions.models.principals.ContributionPrincipal attribute), 125
- local_group (indico.modules.events.models.principals.EventPrincipal attribute), 87
- local_group (indico.modules.events.models.settings.EventSettingPrincipal attribute), 92
- local_group (indico.modules.events.sessions.models.principals.SessionPrincipal attribute), 186
- local_group (indico.modules.rb.models.blocking_principals.BlockingPrincipal attribute), 233
- local_group_id (indico.modules.attachments.models.principals.AttachmentFieldPrincipal attribute), 224
- local_group_id (indico.modules.attachments.models.principals.AttachmentPrincipal attribute), 225
- local_group_id (indico.modules.categories.models.principals.CategoryPrincipal attribute), 207
- local_group_id (indico.modules.events.contributions.models.principals.ContributionPrincipal attribute), 125
- local_group_id (indico.modules.events.models.principals.EventPrincipal attribute), 87
- local_group_id (indico.modules.events.models.settings.EventSettingPrincipal attribute), 92
- local_group_id (indico.modules.events.sessions.models.principals.SessionPrincipal attribute), 186
- local_group_id (indico.modules.rb.models.blocking_principals.BlockingPrincipal attribute), 233
- local_group_id (indico.modules.events.sessions.models.principals.SessionPrincipal attribute), 183
- localized_title (indico.modules.events.layout.models.menu.MenuEntryMixin attribute), 133
- Location (class in indico.modules.rb.models.locations), 234
- location_backref_name (indico.modules.events.contributions.models.contributions.ContributionPrincipal attribute), 120
- location_backref_name (indico.modules.events.models.events.Event attribute), 82
- location_backref_name (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 185
- location_backref_name (indico.modules.events.sessions.models.sessions.SessionPrincipal attribute), 183
- location_backref_name (indico.modules.events.timetable.models.breaks.Break attribute), 95
- location_id (indico.modules.rb.models.aspects.Aspect attribute), 231
- location_id (indico.modules.rb.models.equipment.EquipmentType attribute), 234
- location_id (indico.modules.rb.models.holidays.Holiday attribute), 234
- location_id (indico.modules.rb.models.room_attributes.RoomAttribute attribute), 230
- location_id (indico.modules.rb.models.rooms.Room attribute), 237
- location_name (indico.modules.rb.models.reservations.Reservation attribute), 237
- location_name (indico.modules.rb.models.rooms.Room attribute), 237
- location_parent (indico.modules.events.contributions.models.contributions.ContributionPrincipal attribute), 127
- location_parent (indico.modules.events.contributions.models.subcontributions.SubContributionPrincipal attribute), 127
- location_parent (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 185
- location_parent (indico.modules.events.sessions.models.sessions.SessionPrincipal attribute), 183

location_parent (indico.modules.events.timetable.models.breaks.Break attribute), 195

locator (indico.modules.attachments.models.attachments.Attachment attribute), 220

locator (indico.modules.attachments.models.folders.AttachmentFolder attribute), 223

locator (indico.modules.auth.models.identities.Identity attribute), 242

locator (indico.modules.auth.models.registration_requests.RegistrationRequest attribute), 242

locator (indico.modules.categories.models.categories.Category attribute), 205

locator (indico.modules.designer.models.images.DesignerImageFile attribute), 253

locator (indico.modules.designer.models.templates.DesignerTemplate attribute), 253

locator (indico.modules.events.abstracts.models.abstracts.Abstract attribute), 101

locator (indico.modules.events.abstracts.models.comments.AbstractComment attribute), 104

locator (indico.modules.events.abstracts.models.email_templates.AbstractEmailTemplate attribute), 106

locator (indico.modules.events.abstracts.models.files.AbstractFile attribute), 107

locator (indico.modules.events.abstracts.models.persons.AbstractPerson attribute), 108

locator (indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 110

locator (indico.modules.events.agreements.models.agreements.Agreement attribute), 117

locator (indico.modules.events.contributions.models.contributions.Contribution attribute), 120

locator (indico.modules.events.contributions.models.fields.ContributionField attribute), 122

locator (indico.modules.events.contributions.models.persons.ContributionPersonLink attribute), 124

locator (indico.modules.events.contributions.models.subcontributions.SubContribution attribute), 127

locator (indico.modules.events.contributions.models.types.ContributionType attribute), 128

locator (indico.modules.events.layout.models.images.ImageFile attribute), 131

locator (indico.modules.events.layout.models.menu.EventPage attribute), 132

locator (indico.modules.events.layout.models.menu.MenuEntryMix attribute), 133

locator (indico.modules.events.models.events.Event attribute), 82

locator (indico.modules.events.models.persons.EventPerson attribute), 85

locator (indico.modules.events.models.references.ReferenceType attribute), 89

locator (indico.modules.events.models.reviews.ProposalGroupProxy attribute), 89

locator (indico.modules.events.notes.models.notes.EventNote attribute), 139

locator (indico.modules.events.papers.models.comments.PaperReviewComment attribute), 142

locator (indico.modules.events.papers.models.files.PaperFile attribute), 143

locator (indico.modules.events.papers.models.papers.Paper attribute), 144

locator (indico.modules.events.papers.models.reviews.PaperReview attribute), 147

locator (indico.modules.events.papers.models.reviews.PaperTypeProxy attribute), 148

locator (indico.modules.events.papers.models.revisions.PaperRevision attribute), 148

locator (indico.modules.events.papers.models.templates.PaperTemplate attribute), 149

locator (indico.modules.events.registration.models.form_fields.RegistrationFormField attribute), 163

locator (indico.modules.events.registration.models.forms.RegistrationForm attribute), 166

locator (indico.modules.events.registration.models.invitations.RegistrationInvitation attribute), 167

locator (indico.modules.events.registration.models.items.RegistrationFormSession attribute), 171

locator (indico.modules.events.registration.models.items.RegistrationFormTicket attribute), 172

locator (indico.modules.events.registration.models.registrations.Registration attribute), 160

locator (indico.modules.events.registration.models.registrations.RegistrationGroup attribute), 161

locator (indico.modules.events.reminders.models.reminders.EventReminder attribute), 178

locator (indico.modules.events.requests.models.requests.Request attribute), 180

locator (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 185

locator (indico.modules.events.sessions.models.sessions.Session attribute), 183

locator (indico.modules.events.static.models.static.StaticSite attribute), 203

locator (indico.modules.events.surveys.models.items.SurveyQuestion attribute), 191

locator (indico.modules.events.surveys.models.items.SurveySection attribute), 192

locator (indico.modules.events.surveys.models.items.SurveyText attribute), 192

locator (indico.modules.events.surveys.models.submissions.SurveySubmission attribute), 193

locator (indico.modules.events.surveys.models.surveys.Survey attribute), 189

locator (indico.modules.events.timetable.models.breaks.Break attribute), 195

locator (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 197

- locator (indico.modules.events.tracks.models.tracks.Track attribute), 201
- locator (indico.modules.networks.models.networks.IPNetwork attribute), 260
- locator (indico.modules.news.models.news.NewsItem attribute), 260
- locator (indico.modules.oauth.models.applications.OAuthApplication attribute), 244
- locator (indico.modules.oauth.models.tokens.OAuthToken attribute), 245
- locator (indico.modules.rb.models.locations.Location attribute), 235
- locator (indico.modules.rb.models.reservations.Reservation attribute), 237
- locator (indico.modules.rb.models.rooms.Room attribute), 228
- locator (indico.modules.users.models.users.User attribute), 213
- locator (indico.modules.vc.models.vc_rooms.VCRoom attribute), 248
- locator (indico.modules.vc.models.vc_rooms.VCRoomEventAssociation attribute), 249
- lock_event() (in module indico.modules.events.operations), 94
- log() (indico.modules.events.models.events.Event method), 82
- logged_date (indico.modules.events.logs.models.entries.EventLogEntry attribute), 135
- logged_dt (indico.modules.events.logs.models.entries.EventLogEntry attribute), 135
- logging_disabled (indico.modules.events.models.events.Event attribute), 83
- logo (indico.modules.categories.models.categories.Category attribute), 206
- logo (indico.modules.events.models.events.Event attribute), 83
- logo_metadata (indico.modules.categories.models.categories.Category attribute), 206
- logo_metadata (indico.modules.events.models.events.Event attribute), 83
- logo_url (indico.modules.categories.models.categories.Category attribute), 206
- logo_url (indico.modules.events.models.events.Event attribute), 83
- logo_url (indico.modules.events.payment.plugins.PaymentPluginMixin attribute), 157
- logo_url (indico.modules.vc.plugins.VCPluginMixin attribute), 251
- longitude (indico.modules.rb.models.rooms.Room attribute), 229
- magnitudes (indico.web.forms.fields.RelativeDeltaField attribute), 276
- magnitudes (indico.web.forms.fields.TimeDeltaField attribute), 272
- make_abstract_form() (in module indico.modules.events.abstracts.util), 113
- make_contribution_form() (in module indico.modules.events.contributions.util), 129
- make_diff_log() (in module indico.modules.events.logs.util), 136
- make_email_primary() (in module indico.modules.users.models.users.User method), 214
- make_key() (indico.modules.oauth.models.tokens.OAuthGrant class method), 245
- make_registration_form() (in module indico.modules.events.registration.util), 173
- make_reminder_email() (in module indico.modules.events.reminders.util), 178
- make_survey_form() (in module indico.modules.events.surveys.util), 195
- management (indico.modules.events.logs.models.entries.EventLogRealm attribute), 136
- management_url (in module indico.core.signals.event_management), 48
- manager_emails (indico.modules.rb.models.rooms.Room attribute), 229
- manager_form (indico.modules.events.requests.base.RequestDefinitionBase attribute), 181
- manager_notification_recipients (in module indico.modules.events.registration.models.forms.RegistrationForm attribute), 166
- manager_notifications_enabled (in module indico.modules.events.registration.models.forms.RegistrationForm attribute), 166
- manager_save() (indico.modules.events.requests.base.RequestDefinitionBase class method), 181
- managers (indico.modules.events.papers.models.call_for_papers.CallForPaper attribute), 141
- map_url (indico.modules.rb.models.rooms.Room attribute), 229
- map_url_template (indico.modules.rb.models.locations.Location attribute), 235
- mapping (indico.modules.rb.models.reservations.RepeatMapping attribute), 236
- mark_as_duplicate (indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 109
- MarkdownPreviewer (class in module indico.modules.attachments.preview), 226
- marker_description (in module indico.modules.rb.models.rooms.Room attribute), 229
- marshmallow_aliases (in module indico.modules.events.abstracts.models.abstracts.Abstract attribute), 101
- marshmallow_aliases (in-

dico.modules.events.abstracts.models.comments.AbstractComment (class in dico.modules.events.abstracts.models.comments), 216

attribute), 104

merged (in module indico.core.signals.users), 49

marshmallow_aliases (in module indico.modules.events.abstracts.models.abstracts.AbstractPublicStateAttribute), 102

dico.modules.events.abstracts.models.reviews.AbstractReview (class in dico.modules.events.abstracts.models.reviews), 102

attribute), 110

merged (indico.modules.events.abstracts.models.abstracts.AbstractStateAttribute), 103

max_advance_days (in module indico.modules.rb.models.rooms.RoomAttribute), 229

dico.modules.rb.models.rooms.RoomAttribute (class in dico.modules.rb.models.rooms), 229

merged_into (indico.modules.events.abstracts.models.abstracts.AbstractAttribute), 101

max_capacity (indico.modules.rb.models.rooms.RoomAttribute), 229

merged_into_id (indico.modules.events.abstracts.models.abstracts.AbstractAttribute), 101

md5 (indico.modules.attachments.models.attachments.AttachmentFile (class in indico.modules.attachments.models.attachments), 221

merged_into_id (indico.modules.users.models.users.UserAttribute), 214

md5 (indico.modules.designer.models.images.DesignerImageFile (class in indico.modules.designer.models.images), 253

merged_into_user (indico.modules.users.models.users.UserAttribute), 214

md5 (indico.modules.events.abstracts.models.files.AbstractFileMessage (class in indico.modules.events.reminders.models.reminders.EventReminderAttribute), 107

message (indico.modules.events.reminders.models.reminders.EventReminderAttribute), 178

md5 (indico.modules.events.layout.models.images.ImageFileMessageComplete (class in indico.modules.events.registration.models.forms.RegistrationFormAttribute), 131

message_complete (indico.modules.events.registration.models.forms.RegistrationFormAttribute), 166

md5 (indico.modules.events.papers.models.files.PaperFileMessagePending (class in indico.modules.events.registration.models.forms.RegistrationFormAttribute), 143

message_pending (indico.modules.events.registration.models.forms.RegistrationFormAttribute), 166

md5 (indico.modules.events.papers.models.templates.PaperTemplateMessageUnpaid (class in indico.modules.events.registration.models.forms.RegistrationFormAttribute), 150

message_unpaid (indico.modules.events.registration.models.forms.RegistrationFormAttribute), 166

md5 (indico.modules.events.registration.models.registrations.RegistrationFieldData (class in indico.modules.events.contributions.models.fields.ContributionFieldAttribute), 162

registration_field_data (indico.modules.events.contributions.models.fields.ContributionFieldAttribute), 123

md5 (indico.modules.events.static.models.static.StaticSiteAttribute), 203

mixed (indico.modules.events.abstracts.models.abstracts.AbstractReviewingAttribute), 102

meeting (indico.modules.events.models.events.EventTypeAttribute), 85

model_committed (in module indico.core.signals), 43

members (indico.modules.groups.models.groups.LocalGroupAttribute), 246

moderation_enabled (in module indico.modules.events.registration.models.forms.RegistrationFormAttribute), 166

menu_entries_for_event() (in module indico.modules.events.layout.util), 135

modification_end_dt (in module indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstractAttribute), 103

MenuEntry (class in indico.modules.events.layout.models.menu), 132

modification_end_dt (in module indico.modules.events.registration.models.forms.RegistrationFormAttribute), 166

MenuEntryData (class in indico.modules.events.layout.util), 133

modification_ended (in module indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstractAttribute), 103

MenuEntryMixin (class in indico.modules.events.layout.models.menu), 132

modification_mode (in module indico.modules.events.registration.models.forms.RegistrationFormAttribute), 166

MenuEntryType (class in indico.modules.events.layout.models.menu), 133

ModificationMode (class in indico.modules.events.registration.models.forms), 164

merge (indico.modules.events.abstracts.models.reviews.AbstractActionAttribute), 109

dico.modules.events.registration.models.forms), 164

merge_person_info() (in module indico.modules.events.models.persons.EventPersonAttribute), 86

modified_abstracts (indico.modules.users.models.users.UserAttribute), 214

merge_users() (in module indico.modules.users.util), 218

modified_by (indico.modules.events.abstracts.models.abstracts.AbstractAttribute), 101

merge_users() (indico.modules.events.models.persons.EventPersonAttribute), 86

modified_by (indico.modules.events.abstracts.models.comments.AbstractCommentAttribute), 104

merge_users() (indico.modules.events.settings.EventACLProxyAttribute), 98, 152

modified_by (indico.modules.events.papers.models.comments.PaperReviewAttribute), 142

merge_users() (indico.modules.users.models.suggestions.SuggestionAttribute), 101

modified_by (indico.modules.events.abstracts.models.abstracts.AbstractAttribute), 101

attribute), 101
 modified_by_id (indico.modules.events.abstracts.models.comments.AbstractComment
 attribute), 104
 modified_by_id (indico.modules.events.papers.models.comments.PaperReviewComment
 attribute), 142
 modified_dt (indico.modules.attachments.models.attachments.Attachment
 attribute), 215
 modified_dt (indico.modules.events.abstracts.models.abstracts.AbstractIndico
 attribute), 220
 modified_dt (indico.modules.events.abstracts.models.comments.AbstractComment
 attribute), 101
 modified_dt (indico.modules.events.abstracts.models.comments.AbstractComment
 attribute), 104
 modified_dt (indico.modules.events.abstracts.models.reviews.AbstractReview
 attribute), 110
 modified_dt (indico.modules.events.papers.models.comments.PaperReviewComment
 attribute), 142
 modified_dt (indico.modules.events.papers.models.reviews.PaperReview
 attribute), 147
 modified_dt (indico.modules.vc.models.vc_rooms.VCRoom
 attribute), 248
 modify() (indico.modules.rb.models.reservations.Reservation
 method), 238
 modify_registration() (in module in-
 dico.modules.events.registration.util), 174
 module (indico.modules.categories.models.settings.CategorySetting
 attribute), 208
 module (indico.modules.events.logs.models.entries.EventLogEntry
 attribute), 135
 module (indico.modules.events.models.settings.EventSetting
 attribute), 92
 module (indico.modules.events.models.settings.EventSettingPrincipal
 attribute), 92
 module (indico.modules.users.models.settings.UserSetting
 attribute), 216
 MONTH (indico.modules.rb.models.reservations.RepeatFrequency
 attribute), 236
 move() (indico.modules.categories.models.categories.Category
 method), 206
 move() (indico.modules.events.layout.models.menu.MenuEntry
 method), 132
 move() (indico.modules.events.models.events.Event
 method), 83
 move() (indico.modules.events.timetable.models.entries.TimetableEntry
 method), 197
 move_category() (in module in-
 dico.modules.categories.operations), 208
 move_next_to() (indico.modules.events.timetable.models.entries.TimetableEntry
 method), 197
 move_start_dt() (indico.modules.events.models.events.Event
 method), 83
 move_timetable_entry() (in module in-
 dico.modules.events.timetable.operations),
 198
 moved (in module indico.core.signals.category), 45
 moved (in module indico.core.signals.event), 46
 mr (indico.modules.users.models.users.UserTitle
 attribute), 215
 mrs (indico.modules.users.models.users.UserTitle
 attribute), 215
 ms (indico.modules.users.models.users.UserTitle
 attribute), 215
 MultiIPNetworkField (class in in-
 dico.modules.networks.fields), 267
 multipass_data (indico.modules.auth.models.identities.Identity
 attribute), 212
 multipass_group_name (in-
 dico.modules.attachments.models.principals.AttachmentFolderPrin-
 cipal
 attribute), 224
 multipass_group_name (in-
 dico.modules.attachments.models.principals.AttachmentPrincipal
 attribute), 225
 multipass_group_name (in-
 dico.modules.categories.models.principals.CategoryPrincipal
 attribute), 207
 multipass_group_name (in-
 dico.modules.events.contributions.models.principals.Contribution
 attribute), 125
 multipass_group_name (in-
 dico.modules.events.models.principals.EventPrincipal
 attribute), 87
 multipass_group_name (in-
 dico.modules.events.models.settings.EventSettingPrincipal
 attribute), 92
 multipass_group_name (in-
 dico.modules.events.sessions.models.principals.SessionPrincipal
 attribute), 186
 multipass_group_name (in-
 dico.modules.rb.models.blocking_principals.BlockingPrincipal
 attribute), 233
 multipass_group_provider (in-
 dico.modules.attachments.models.principals.AttachmentFolderPrin-
 cipal
 attribute), 224
 multipass_group_provider (in-
 dico.modules.attachments.models.principals.AttachmentPrincipal
 attribute), 225
 multipass_group_provider (in-
 dico.modules.categories.models.principals.CategoryPrincipal
 attribute), 207
 multipass_group_provider (in-
 dico.modules.events.contributions.models.principals.Contribution
 attribute), 125
 multipass_group_provider (in-
 dico.modules.events.models.principals.EventPrincipal
 attribute), 87
 multipass_group_provider (in-
 dico.modules.events.models.settings.EventSettingPrincipal
 attribute), 93
 multipass_group_provider (in-
 dico.modules.events.sessions.models.principals.SessionPrincipal

name (indico.modules.events.abstracts.placeholders.TargetSubmit(Indico.modules.events.registration.placeholders.registrations.LastName attribute), 115
 name (indico.modules.events.agreements.placeholders.AgreementLinkPlaceholder(indico.modules.events.registration.placeholders.registrations.LinkPlaceholder attribute), 119
 name (indico.modules.events.agreements.placeholders.PersonName(Placeholder(indico.modules.events.requests.base.RequestDefinitionBase attribute), 119
 name (indico.modules.events.contributions.models.types.ContribType(indico.modules.groups.models.groups.LocalGroup attribute), 128
 name (indico.modules.events.layout.models.menu.MenuEntry name (indico.modules.networks.models.networks.IPNetworkGroup attribute), 132
 name (indico.modules.events.layout.util.MenuEntryData name (indico.modules.oauth.models.applications.OAuthApplication attribute), 134
 name (indico.modules.events.logs.renderers.EmailRenderer name (indico.modules.rb.models.aspects.Aspect attribute), 136
 name (indico.modules.events.logs.renderers.EventLogRendererBase(indico.modules.rb.models.equipment.EquipmentType attribute), 137
 name (indico.modules.events.logs.renderers.SimpleRenderer name (indico.modules.rb.models.holidays.Holiday attribute), 137
 name (indico.modules.events.models.references.ReferenceType name (indico.modules.rb.models.locations.Location attribute), 89
 name (indico.modules.events.models.settings.EventSetting name (indico.modules.rb.models.room_attributes.RoomAttribute attribute), 92
 name (indico.modules.events.models.settings.EventSettingPrincipal(indico.modules.rb.models.rooms.Room attribute), 93
 name (indico.modules.events.papers.models.templates.PaperTemplate(indico.modules.users.models.affiliations.UserAffiliation attribute), 150
 name (indico.modules.events.persons.placeholders.EmailPlaceholder(indico.modules.users.models.settings.UserSetting attribute), 157
 name (indico.modules.events.persons.placeholders.EventLinkPlaceholder(indico.modules.users.models.users.PersonMixin attribute), 157
 name (indico.modules.events.persons.placeholders.EventTitlePlaceholder(indico.modules.vc.models.vc_rooms.VCRoom attribute), 158
 name (indico.modules.events.persons.placeholders.FirstNamePlaceholder(indico.modules.designer.placeholders.RegistrationFullName attribute), 158
 name (indico.modules.events.persons.placeholders.LastNamePlaceholder(indico.modules.designer.placeholders.RegistrationFullName attribute), 158
 name (indico.modules.events.persons.placeholders.RegisterLinkPlaceholder(indico.modules.designer.placeholders.RegistrationFullName attribute), 158
 name (indico.modules.events.registration.placeholders.invitations.FirstNames(Placeholder(indico.modules.designer.placeholders.RegistrationFullName attribute), 175
 name (indico.modules.events.registration.placeholders.invitations.Institution(Placeholder(indico.modules.designer.placeholders.RegistrationFullName attribute), 175
 name (indico.modules.events.registration.placeholders.invitations.LastNames(Placeholder(indico.modules.designer.placeholders.RegistrationFullName attribute), 175
 name (indico.modules.events.registration.placeholders.registrations.Events(Placeholder(indico.modules.designer.placeholders.RegistrationFullName attribute), 174
 name (indico.modules.events.registration.placeholders.registrations.EventsTitle(Placeholder(indico.modules.designer.placeholders.RegistrationFullName attribute), 174
 name (indico.modules.events.registration.placeholders.registrationFormFieldPlaceholder(class in indico.modules.users.models.users), 174
 name (indico.modules.events.registration.placeholders.registrationFormFirstName(Placeholder(indico.modules.rb.models.reservations.Reservation attribute), 174
 name (indico.modules.events.registration.placeholders.registrationFormIDPlaceholder (indico.modules.rb.models.reservations.Reservation attribute), 174

attribute), 238
 negative (indico.modules.events.abstracts.models.abstracts.AbstractReviewingState attribute), 103
 negative (indico.modules.events.logs.models.entries.EventLogKind attribute), 136
 network (indico.modules.networks.models.networks.IPNetwork attribute), 259
 networks (indico.modules.networks.models.networks.IPNetwork attribute), 260
 NEVER (indico.modules.rb.models.reservations.RepeatFrequency attribute), 236
 new_submission_emails (in-dico.modules.events.surveys.models.surveys.Survey attribute), 189
 new_tab (indico.modules.events.layout.models.menu.MenuEntry attribute), 132
 NewsItem (class in indico.modules.news.models.news), 260
 next() (indico.modules.events.payment.models.transactions.TransactionState.Transition class method), 155
 NO_RESERVATION_USER_STRATEGY (in-dico.modules.rb.models.reservation_occurrences.ReservationOccurrence attribute), 239
 no_score (indico.modules.events.abstracts.models.review_questions.AdminReviewQuestion attribute), 108
 no_score (indico.modules.events.papers.models.review_questions.PaperReviewQuestion attribute), 145
 nonbookable_periods (in-dico.modules.rb.models.rooms.Room attribute), 229
 NonBookablePeriod (class in in-dico.modules.rb.models.room_nonbookable_periods), 231
 none (indico.modules.events.abstracts.models.abstracts.EditToolsManagers attribute), 103
 none (indico.modules.events.abstracts.settings.BOACorrespondingAuthorityTypes attribute), 116
 none (indico.modules.events.contributions.models.persons.AuthorType attribute), 124
 none (indico.modules.events.timetable.reschedule.RescheduleHandler attribute), 200
 none (indico.modules.oauth.models.applications.SystemApplication attribute), 245
 none (indico.modules.users.models.users.UserTitle attribute), 215
 not_allowed (indico.modules.events.registration.models.forms.ModificationMode attribute), 164
 not_empty_answers (in-dico.modules.events.surveys.models.items.SurveyQuestion attribute), 191
 not_ready (indico.modules.events.surveys.models.surveys.SurveyState attribute), 190
 not_started (indico.modules.events.abstracts.models.abstracts.AbstractReviewingState attribute), 103
 note (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 145
 note_added (in module indico.core.signals.event), 46
 note_deleted (in module indico.core.signals.event), 46
 note_id (indico.modules.events.notes.models.notes.EventNoteRevision attribute), 140
 note_modified (in module indico.core.signals.event), 46
 notification_before_days (in-dico.modules.rb.models.rooms.Room attribute), 229
 notification_before_days_monthly (in-dico.modules.rb.models.rooms.Room attribute), 229
 notification_before_days_weekly (in-dico.modules.rb.models.rooms.Room attribute), 229
 notification_emails (in-dico.modules.rb.models.rooms.Room attribute), 229
 notification_for_assistance (in-dico.modules.rb.models.rooms.Room attribute), 229
 notification_sender_address (in-dico.modules.rb.models.rooms.Room attribute), 166
 no_score (indico.modules.events.papers.models.review_questions.PaperReviewQuestion attribute), 239
 notifications_enabled (in-dico.modules.events.surveys.models.surveys.Survey attribute), 189
 notifications_enabled (in-dico.modules.rb.models.rooms.Room attribute), 229
 None (indico.modules.categories.models.categories.Category attribute), 206
 None (indico.modules.events.surveys.models.surveys.Survey attribute), 189
 None (indico.modules.categories.models.categories.Category method), 206
 None (indico.modules.events.papers.models.revisions.PaperRevision attribute), 149
 None (indico.modules.rb.models.rooms.Room attribute), 229
 O
 not_allowed (indico.modules.events.registration.models.forms.ModificationMode attribute), 164
 not_empty_answers (in-dico.modules.events.surveys.models.items.SurveyQuestion attribute), 191
 not_ready (indico.modules.events.surveys.models.surveys.SurveyState attribute), 190
 not_started (indico.modules.events.abstracts.models.abstracts.AbstractReviewingState attribute), 103
 object (indico.modules.events.models.persons.PersonLinkBase attribute), 87

- object (indico.modules.events.timetable.models.entries.TimeTableEntry (indico.modules.events.contributions.models.contributions.Contribution (indico.modules.events.contributions.models.contributions.Contribution attribute), 197
 attribute), 121
- object_relationship_name (in- own_address (indico.modules.events.models.events.Event (indico.modules.events.models.events.Event attribute), 83
 dico.modules.events.abstracts.models.persons.AbstractPersonLink (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 108
 own_address (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 185
- object_relationship_name (in- attribute), 185
 dico.modules.events.contributions.models.persons.ContactPersonLink (indico.modules.events.sessions.models.sessions.Session attribute), 124
 attribute), 184
- object_relationship_name (in- own_address (indico.modules.events.timetable.models.breaks.Break (indico.modules.events.timetable.models.breaks.Break attribute), 125
 dico.modules.events.contributions.models.persons.SubContactPersonLink (indico.modules.events.registration.models.items.RegistrationForm attribute), 171
 own_data (indico.modules.events.registration.models.items.RegistrationForm attribute), 171
- object_relationship_name (in- attribute), 171
 dico.modules.events.models.persons.EventPersonLink (indico.modules.events.models.persons.EventPersonLink attribute), 86
 link_no_access_contact (in- attribute), 220
 dico.modules.attachments.models.attachments.Attachment (in- attribute), 220
- object_relationship_name (in- attribute), 220
 dico.modules.events.models.persons.PersonLinkBase (indico.modules.attachments.models.folders.AttachmentFolder attribute), 87
 base_no_access_contact (in- attribute), 223
 dico.modules.attachments.models.folders.AttachmentFolder (in- attribute), 223
- object_relationship_name (in- attribute), 223
 dico.modules.events.sessions.models.persons.SessionBlockPersonLink (indico.modules.categories.models.categories.Category attribute), 186
 session_block_person_contact (in- attribute), 206
 dico.modules.categories.models.categories.Category (in- attribute), 206
- occurrences (indico.modules.rb.models.reservations.Reservation (indico.modules.rb.models.reservations.Reservation attribute), 238
 own_no_access_contact (in- attribute), 238
 dico.modules.events.contributions.models.contributions.Contribution (in- attribute), 238
- OccurrencesField (class in indico.web.forms.fields), 272
- old_api_keys (indico.modules.users.models.users.User (indico.modules.users.models.users.User attribute), 214
 own_no_access_contact (in- attribute), 214
 dico.modules.events.contributions.models.contributions.Contribution (in- attribute), 214
- open() (indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstracts (indico.modules.events.models.events.Event (indico.modules.events.models.events.Event method), 103
 attribute), 83
- open() (indico.modules.events.papers.models.call_for_papers.CallForPapers (indico.modules.events.papers.models.call_for_papers.CallForPapers method), 141
 own_no_access_contact (in- attribute), 184
 dico.modules.events.sessions.models.sessions.Session (in- attribute), 184
- open() (indico.modules.events.surveys.models.surveys.Survey (indico.modules.events.surveys.models.surveys.Survey method), 189
 own_room (indico.modules.events.contributions.models.contributions.Contribution attribute), 121
- open_cfa() (in module in- attribute), 121
 dico.modules.events.abstracts.operations), own_room (indico.modules.events.models.events.Event attribute), 83
- open_cfp() (in module in- own_room (indico.modules.events.sessions.models.blocks.SessionBlock (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 150
 attribute), 185
- option_widget (indico.web.forms.fields.IndicoEnumRadioField (indico.web.forms.fields.IndicoEnumRadioField attribute), 273
 own_room (indico.modules.events.sessions.models.sessions.Session attribute), 184
- option_widget (indico.web.forms.fields.IndicoSelectMultipleCheckboxField (indico.web.forms.fields.IndicoSelectMultipleCheckboxField attribute), 267
 own_room (indico.modules.events.sessions.models.sessions.Session attribute), 184
- order_by_name (indico.modules.events.registration.models.registration_items.RegistrationItem (indico.modules.events.registration.models.registration_items.RegistrationItem attribute), 160
 attribute), 121
- organizer_info (indico.modules.events.models.events.Event (indico.modules.events.models.events.Event attribute), 83
 own_room_id (indico.modules.events.models.events.Event attribute), 83
- other (indico.modules.events.logs.models.entries.EventLogKeyword (indico.modules.events.logs.models.entries.EventLogKeyword attribute), 136
 own_room_id (indico.modules.events.sessions.models.blocks.SessionBlock (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 185
- overlaps() (indico.modules.rb.models.reservation_occurrences.ReservationOccurrences (indico.modules.rb.models.reservation_occurrences.ReservationOccurrences method), 239
 own_room_id (indico.modules.events.sessions.models.sessions.Session attribute), 184
- overlaps() (indico.modules.rb.models.room_nonbookable_periods.NonBookablePeriod (indico.modules.rb.models.room_nonbookable_periods.NonBookablePeriod method), 231
 own_room_id (indico.modules.events.timetable.models.breaks.Break attribute), 196
- OverrideMultipleItemsField (class in in- own_room_name (indico.modules.events.contributions.models.contributions.Contribution (indico.modules.events.contributions.models.contributions.Contribution class in in- attribute), 121
 dico.web.forms.fields), 274 attribute), 121
- OverviewStats (class in in- own_room_name (indico.modules.events.models.events.Event (indico.modules.events.registration.stats), 177
 attribute), 83

[own_room_name \(indico.modules.events.sessions.models.blocks.SessionBlock attribute\), 185](#)
[own_room_name \(indico.modules.events.sessions.models.sessions.Session attribute\), 184](#)
[own_room_name \(indico.modules.events.timetable.models.breaks.Break attribute\), 196](#)
[own_venue \(indico.modules.events.contributions.models.contributions.Contribution attribute\), 121](#)
[own_venue \(indico.modules.events.models.events.Event attribute\), 83](#)
[own_venue \(indico.modules.events.sessions.models.blocks.SessionBlock attribute\), 185](#)
[own_venue \(indico.modules.events.sessions.models.sessions.Session attribute\), 184](#)
[own_venue \(indico.modules.events.timetable.models.breaks.Break attribute\), 196](#)
[own_venue_id \(indico.modules.events.contributions.models.contributions.Contribution attribute\), 121](#)
[own_venue_id \(indico.modules.events.models.events.Event attribute\), 83](#)
[own_venue_id \(indico.modules.events.sessions.models.blocks.SessionBlock attribute\), 185](#)
[own_venue_id \(indico.modules.events.sessions.models.sessions.Session attribute\), 184](#)
[own_venue_id \(indico.modules.events.timetable.models.breaks.Break attribute\), 196](#)
[own_venue_name \(indico.modules.events.contributions.models.contributions.Contribution attribute\), 121](#)
[own_venue_name \(indico.modules.events.models.events.Event attribute\), 83](#)
[own_venue_name \(indico.modules.events.sessions.models.blocks.SessionBlock attribute\), 185](#)
[own_venue_name \(indico.modules.events.sessions.models.sessions.Session attribute\), 184](#)
[own_venue_name \(indico.modules.events.timetable.models.breaks.Break attribute\), 196](#)
[own_visibility_horizon \(indico.modules.categories.models.categories.Category attribute\), 206](#)
[owner \(indico.modules.designer.models.templates.DesignerTemplate attribute\), 254](#)
[owner \(indico.modules.rb.models.rooms.Room attribute\), 229](#)
[owner_id \(indico.modules.rb.models.rooms.Room attribute\), 230](#)

P

[page \(indico.modules.events.layout.models.menu.MenuEntry attribute\), 132](#)
[page \(indico.modules.events.layout.models.menu.MenuEntryType attribute\), 133](#)
[page_id \(indico.modules.events.layout.models.menu.MenuEntry attribute\), 132](#)
[PaperRevisionBlock \(class in indico.modules.events.papers.models.papers\), 144](#)
[PaperRevision \(class in indico.modules.events.contributions.models.contributions\), 121](#)
[PaperFile \(class in indico.modules.events.papers.models.files\), 143](#)
[PaperRevision \(class in indico.modules.events.papers.models.revisions\), 149](#)
[PaperContentReviewers \(class in indico.modules.events.contributions.models.contributions\), 121](#)
[PaperJudges \(class in indico.modules.events.contributions.models.contributions\), 121](#)
[PaperLayoutReviewers \(class in indico.modules.events.contributions.models.contributions\), 121](#)
[PaperCommentVisibility \(class in indico.modules.events.papers.models.reviews\), 146](#)
[PaperComment \(class in indico.modules.events.papers.models.reviews\), 146](#)
[PaperCompetences \(class in indico.modules.events.papers.models.competences\), 142](#)
[PaperEmailSettingsField \(class in indico.modules.events.papers.fields\), 265](#)
[PaperFile \(class in indico.modules.events.papers.models.files\), 143](#)
[PaperJudgmentProxy \(class in indico.modules.events.papers.models.reviews\), 146](#)
[PaperReview \(class in indico.modules.events.papers.models.reviews\), 141](#)
[PaperReviewComment \(class in indico.modules.events.papers.models.comments\), 141](#)
[PaperReviewQuestion \(class in indico.modules.events.papers.models.review_questions\), 145](#)
[PaperReviewRating \(class in indico.modules.events.papers.models.review_ratings\), 145](#)
[PaperReviewType \(class in indico.modules.events.papers.models.reviews\), 147](#)
[PaperRevision \(class in indico.modules.events.papers.models.revisions\), 143](#)

- 148
- PaperRevisionState (class in indico.modules.events.papers.models.revisions), 149
- PaperTemplate (class in indico.modules.events.papers.models.templates), 149
- PaperTypeProxy (class in indico.modules.events.papers.models.reviews), 147
- param_required (indico.modules.events.registration.placeholders.registrations.FieldPlaceholder attribute), 174
- param_restricted (indico.modules.events.registration.placeholders.registrations.FieldPlaceholder attribute), 174
- parent_chain_query (in-dico.modules.categories.models.categories.Category attribute), 206
- parent_id (indico.modules.categories.models.categories.Category attribute), 206
- parent_id (indico.modules.events.layout.models.menu.MenuEntry attribute), 132
- parent_id (indico.modules.events.registration.models.form_fields.RegistrationFormFields attribute), 163
- parent_id (indico.modules.events.registration.models.form_fields.RegistrationFormFields attribute), 164
- parent_id (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 169
- parent_id (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 170
- parent_id (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 171
- parent_id (indico.modules.events.registration.models.items.RegistrationFormItems attribute), 172
- parent_id (indico.modules.events.surveys.models.items.SurveyQuestion attribute), 190
- parent_id (indico.modules.events.surveys.models.items.SurveyQuestion attribute), 191
- parent_id (indico.modules.events.surveys.models.items.SurveySection attribute), 192
- parent_id (indico.modules.events.surveys.models.items.SurveyText attribute), 192
- parent_id (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 197
- parent_id (indico.modules.rb.models.equipment.EquipmentType attribute), 234
- parent_id (indico.modules.rb.models.room_attributes.RoomAttribute attribute), 230
- parent_id (indico.modules.events.surveys.models.surveys.Survey attribute), 186
- parent_id (indico.modules.events.surveys.models.surveys.Survey attribute), 189
- participants (indico.modules.events.logs.models.entries.EventLogRealtime attribute), 136
- participation_regform (in-dico.modules.events.models.events.Event attribute), 83
- password (indico.modules.auth.models.identities.Identity attribute), 242
- password_hash (indico.modules.auth.models.identities.Identity attribute), 242
- payment_dt (indico.modules.events.registration.models.registrations.Registration attribute), 160
- PaymentPluginMixin (class in indico.modules.events.payment.plugins), 156
- PaymentTransaction (class in indico.modules.events.payment.models.transactions), 154
- PDFPreviewer (class in indico.modules.events.registration.models.preview), 226
- pending (indico.modules.events.agreements.models.agreements.Agreement attribute), 117
- pending (indico.modules.events.agreements.models.agreements.Agreement attribute), 118
- pending (indico.modules.events.payment.models.transactions.TransactionAttribute attribute), 155
- pending (indico.modules.events.payment.models.transactions.TransactionState attribute), 155
- pending (indico.modules.events.registration.models.invitations.InvitationState attribute), 167
- pending (indico.modules.events.registration.models.registrations.Registration attribute), 162
- pending (indico.modules.events.requests.models.requests.RequestState attribute), 180
- pending (indico.modules.events.static.StaticSiteState attribute), 203
- pending (indico.modules.events.blocked_rooms.BlockedRoomState attribute), 233
- pending (indico.modules.events.surveys.models.submissions.Survey attribute), 193
- printing_paper_files (in-dico.modules.events.contributions.models.contributions.Contribution attribute), 121
- person (indico.modules.events.abstracts.models.persons.AbstractPersonLink attribute), 108
- person (indico.modules.events.contributions.models.persons.ContributionPerson attribute), 124
- person (indico.modules.events.contributions.models.persons.SubContributionPerson attribute), 125
- person (indico.modules.events.models.persons.EventPersonLink attribute), 86
- person (indico.modules.events.models.persons.PersonLinkBase attribute), 87
- person (indico.modules.events.sessions.models.persons.SessionBlockPerson attribute), 117
- person_email (indico.modules.events.agreements.models.agreements.Agreement attribute), 117
- person_id (indico.modules.events.abstracts.models.persons.AbstractPersonLink attribute), 108
- person_id (indico.modules.events.contributions.models.persons.ContributionPerson attribute), 124

person_id (indico.modules.events.contributions.models.persons.SubContributorPersonLink attribute), 125
 person_id (indico.modules.events.models.persons.EventPersonLink attribute), 86
 person_id (indico.modules.events.models.persons.PersonLinkBase attribute), 87
 person_id (indico.modules.events.sessions.models.persons.SessionBlockPersonLink attribute), 186
 person_link_backref_name (indico.modules.events.abstracts.models.persons.AbstractPersonLink attribute), 108
 person_link_backref_name (indico.modules.events.contributions.models.persons.SubContributorPersonLink attribute), 124
 person_link_backref_name (indico.modules.events.contributions.models.persons.SubContributorPersonLink attribute), 125
 person_link_backref_name (indico.modules.events.models.persons.EventPersonLink attribute), 86
 person_link_backref_name (indico.modules.events.models.persons.PersonLinkBase attribute), 87
 person_link_backref_name (indico.modules.events.sessions.models.persons.SessionBlockPersonLink attribute), 186
 person_link_cls (indico.modules.events.abstracts.fields.AbstractPersonLinkTypeField attribute), 262
 person_link_cls (indico.modules.events.contributions.fields.ContributionPersonLinkListField attribute), 265
 person_link_cls (indico.modules.events.contributions.fields.SubContributorPersonLinkListField attribute), 265
 person_link_cls (indico.modules.events.fields.EventPersonLinkListField attribute), 261
 person_link_cls (indico.modules.events.fields.PersonLinkListFieldBase attribute), 262
 person_link_cls (indico.modules.events.sessions.fields.SessionBlockPersonLinkListField attribute), 266
 person_link_data (indico.modules.events.models.persons.PersonLinkDataMixin (class in indico.modules.events.models.persons)), 87
 person_link_unique_columns (indico.modules.events.abstracts.models.persons.AbstractPersonLink attribute), 108
 person_link_unique_columns (indico.modules.events.contributions.models.persons.SubContributorPersonLink attribute), 124
 person_link_unique_columns (indico.modules.events.contributions.models.persons.SubContributorPersonLink attribute), 125
 person_link_unique_columns (indico.modules.events.models.persons.EventPersonLink attribute), 86
 person_link_unique_columns (indico.modules.events.models.persons.PersonLinkBase attribute), 87
 person_link_unique_columns (indico.modules.events.registration.models.items.PersonalDataType attribute), 163
 person_link_unique_columns (indico.modules.events.registration.models.items.RegistrationFormItem attribute), 164
 person_link_unique_columns (indico.modules.events.registration.models.items.RegistrationFormItem attribute), 169
 person_link_unique_columns (indico.modules.events.registration.models.items.RegistrationFormSection attribute), 171
 person_link_unique_columns (indico.modules.events.registration.models.items.RegistrationFormText attribute), 172
 person_link_unique_columns (indico.modules.events.registration.models.items.RegistrationFormText attribute), 168
 person_link_unique_columns (indico.modules.events.registration.models.items.RegistrationFormText attribute), 172
 person_name (indico.modules.events.agreements.models.agreements.Agreement attribute), 119
 person_updated (in module indico.core.signals.event), 46
 personal_data_type (indico.modules.events.registration.models.form_fields.RegistrationFormText attribute), 163
 personal_data_type (indico.modules.events.registration.models.form_fields.RegistrationFormText attribute), 164
 personal_data_type (indico.modules.events.registration.models.items.RegistrationFormItem attribute), 169
 personal_data_type (indico.modules.events.registration.models.items.RegistrationFormSection attribute), 171
 personal_data_type (indico.modules.events.registration.models.items.RegistrationFormText attribute), 172
 personal_data_type (indico.modules.events.registration.models.items.RegistrationFormText attribute), 168
 personal_data_type (indico.modules.events.registration.models.items.RegistrationFormText attribute), 172
 phone (indico.modules.events.models.persons.EventPersonLink attribute), 86
 phone (indico.modules.events.models.persons.PersonLinkBase attribute), 87
 phone (indico.modules.events.registration.models.items.PersonalDataType attribute), 163

- attribute), 168
- phone (indico.modules.users.models.users.User attribute), 214
- Photo (class in indico.modules.rb.models.photos), 235
- photo (indico.modules.rb.models.rooms.Room attribute), 230
- photo_id (indico.modules.rb.models.rooms.Room attribute), 230
- plugin (indico.modules.events.layout.models.menu.MenuEntry attribute), 132
- plugin (indico.modules.events.layout.util.MenuEntryData attribute), 134
- plugin (indico.modules.events.logs.renderers.EventLogRenderBase attribute), 137
- plugin (indico.modules.events.payment.models.transactions.PaymentTransaction attribute), 155
- plugin (indico.modules.events.requests.base.RequestDefinitionBase attribute), 181
- plugin (indico.modules.vc.models.vc_rooms.VCRoom attribute), 248
- plugin_link (indico.modules.events.layout.models.menu.MenuEntry attribute), 133
- plugin_url_rule_to_js() (in module indico.core.plugins), 42
- populate_obj() (indico.web.forms.fields.JSONField method), 268
- position (indico.modules.categories.models.categories.Category attribute), 206
- position (indico.modules.events.abstracts.models.email_templates.AbstractEmailTemplate attribute), 106
- position (indico.modules.events.abstracts.models.review_questions.AbstractReviewQuestion attribute), 108
- position (indico.modules.events.contributions.models.fields.ContributionField attribute), 123
- position (indico.modules.events.contributions.models.subcontributions.SubContribution attribute), 127
- position (indico.modules.events.layout.models.menu.MenuEntry attribute), 132
- position (indico.modules.events.papers.models.review_questions.PaperReviewQuestion attribute), 145
- position (indico.modules.events.registration.models.form_fields.RegistrationFormField attribute), 163
- position (indico.modules.events.registration.models.form_fields.RegistrationFormPersonalDataField attribute), 164
- position (indico.modules.events.registration.models.items.PossibleDataType attribute), 168
- position (indico.modules.events.registration.models.items.RegistrationFormItem attribute), 169
- position (indico.modules.events.registration.models.items.RegistrationFormPersonalDataSection attribute), 170
- position (indico.modules.events.registration.models.items.RegistrationFormSection attribute), 171
- position (indico.modules.events.registration.models.items.RegistrationFormText attribute), 172
- position (indico.modules.events.surveys.models.items.SurveyItem attribute), 190
- position (indico.modules.events.surveys.models.items.SurveyQuestion attribute), 191
- position (indico.modules.events.surveys.models.items.SurveySection attribute), 192
- position (indico.modules.events.surveys.models.items.SurveyText attribute), 192
- position (indico.modules.events.tracks.models.tracks.Track attribute), 202
- positive (indico.modules.events.abstracts.models.abstracts.AbstractReview attribute), 103
- possible (indico.modules.events.logs.models.entries.EventLogKind attribute), 136
- possible_render_modes (indico.modules.categories.models.categories.Category attribute), 206
- possible_render_modes (indico.modules.events.abstracts.models.abstracts.Abstract attribute), 102
- possible_render_modes (indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 110
- possible_render_modes (indico.modules.events.contributions.models.contributions.Contribution attribute), 121
- possible_render_modes (indico.modules.events.contributions.models.subcontributions.SubContribution attribute), 127
- possible_render_modes (indico.modules.events.papers.models.revisions.PaperRevision attribute), 149
- possible_render_modes (indico.modules.events.sessions.models.sessions.Session attribute), 161
- possible_render_modes (indico.modules.events.tracks.models.tracks.Track attribute), 202
- pre_validate() (indico.modules.categories.fields.CategoryField method), 201
- pre_validate() (indico.modules.events.abstracts.fields.AbstractField method), 161
- pre_validate() (indico.modules.events.abstracts.fields.AbstractPersonLinkListField method), 262
- pre_validate() (indico.modules.events.abstracts.fields.EmailRuleListField method), 262

method), 264

pre_validate() (indico.modules.events.contributions.fields.ContributionPersonListField method), 265

pre_validate() (indico.modules.events.fields.EventPersonListField method), 261

pre_validate() (indico.modules.events.fields.EventPersonListField method), 261

pre_validate() (indico.modules.events.fields.ReferencesField method), 262

pre_validate() (indico.modules.networks.fields.MultiIPNetworkField method), 267

pre_validate() (indico.web.forms.fields.IndicoDateTimeField method), 272

pre_validate() (indico.web.forms.fields.IndicoPalettePickerField method), 271

pre_validate() (indico.web.forms.fields.MultipleItemsField method), 274

pre_validate() (indico.web.forms.fields.MultiStringField method), 274

pre_validate() (indico.web.forms.fields.OverrideMultipleItemsField method), 275

pre_validate() (indico.web.forms.fields.PrincipalListField method), 275

pre_validate() (indico.web.forms.fields.RelativeDeltaField method), 276

pre_validate() (indico.web.forms.fields.TextListField method), 269

pre_validate() (indico.web.forms.fields.TimeDeltaField method), 272

preferences (in module indico.core.signals.users), 49

preload_acl_entries() (indico.modules.events.contributions.models.contributions.Contribution class method), 121

preload_acl_entries() (indico.modules.events.sessions.models.sessions.Session class method), 184

preload_all_acl_entries() (indico.modules.events.models.events.Event method), 83

PRELOAD_EVENT_ATTACHED_ITEMS (indico.modules.events.contributions.models.contributions.Contribution attribute), 119

PRELOAD_EVENT_ATTACHED_ITEMS (indico.modules.events.contributions.models.subcontributions.SubContribution attribute), 126

PRELOAD_EVENT_ATTACHED_ITEMS (indico.modules.events.sessions.models.sessions.Session attribute), 182

PRELOAD_EVENT_NOTES (indico.modules.events.contributions.models.contributions.Contribution attribute), 119

PRELOAD_EVENT_NOTES (indico.modules.events.contributions.models.subcontributions.SubContribution attribute), 126

PRELOAD_EVENT_NOTES (indico.modules.events.sessions.models.sessions.Session attribute), 182

PRELOAD_EVENT_NOTES (indico.modules.events.registration.models.registrations.Registration attribute), 160

price (indico.modules.events.registration.models.registrations.RegistrationD attribute), 162

principal_adjustment (indico.modules.events.registration.models.registrations.R attribute), 160

primary (indico.modules.events.contributions.models.persons.AuthorType attribute), 124

primary_authors (indico.modules.events.models.persons.AuthorsSpeakersM attribute), 85

PrimaryAuthorsPlaceholder (class in indico.modules.events.abstracts.placeholders), 114

principal (indico.modules.events.models.persons.EventPerson attribute), 86

principal_backref_name (indico.modules.attachments.models.principals.AttachmentFolderPr attribute), 224

principal_backref_name (indico.modules.attachments.models.principals.AttachmentPrincipal attribute), 225

principal_backref_name (indico.modules.categories.models.principals.CategoryPrincipal attribute), 207

principal_backref_name (indico.modules.events.contributions.models.principals.Contribution attribute), 125

principal_backref_name (indico.modules.events.models.principals.EventPrincipal attribute), 88

principal_backref_name (indico.modules.events.models.settings.EventSettingPrincipal attribute), 93

principal_backref_name (indico.modules.events.sessions.models.principals.SessionPrincipal attribute), 186

principal_backref_name (indico.modules.rb.models.blocking_principals.BlockingPrincipal attribute), 200

principal_for (indico.modules.categories.models.principals.CategoryPrincipal attribute), 207

principal_for (indico.modules.events.contributions.models.principals.Contribution attribute), 125

principal_for (indico.modules.events.models.principals.EventPrincipal attribute), 88

principal_for (indico.modules.events.sessions.models.principals.SessionPrincipal attribute), 186

principals (indico.modules.groups.core.GroupProxy attribute), 247

[principal_order \(indico.modules.networks.models.networks.IPNetworkField attribute\), 260](#)
[principal_order \(indico.modules.users.models.users.User attribute\), 214](#)
[principal_type \(indico.modules.networks.models.networks.IPNetworkField attribute\), 260](#)
[principal_type \(indico.modules.users.models.users.User attribute\), 214](#)
[PrincipalField \(class in indico.web.forms.fields\), 275](#)
[PrincipalListField \(class in indico.web.forms.fields\), 275](#)
[print_badge_template \(in module indico.core.signals.event\), 46](#)
[private \(indico.modules.events.surveys.models.surveys.Survey attribute\), 189](#)
[process_data\(\) \(indico.modules.categories.fields.CategoryField method\), 266](#)
[process_data\(\) \(indico.modules.networks.fields.MultiIPNetworkField method\), 267](#)
[process_data\(\) \(indico.web.forms.fields.IndicoEmailRecipientsField method\), 277](#)
[process_data\(\) \(indico.web.forms.fields.IndicoPalettePickerField method\), 271](#)
[process_data\(\) \(indico.web.forms.fields.IndicoStaticTextField method\), 270](#)
[process_data\(\) \(indico.web.forms.fields.IndicoTimezoneSelectorField method\), 273](#)
[process_form_data\(\) \(indico.modules.users.ext.ExtraUserPreferences method\), 219](#)
[process_formdata\(\) \(indico.modules.categories.fields.CategoryField method\), 266](#)
[process_formdata\(\) \(indico.modules.events.fields.EventPersonListField method\), 261](#)
[process_formdata\(\) \(indico.modules.events.fields.ReferencesField method\), 262](#)
[process_formdata\(\) \(indico.modules.events.fields.ReviewQuestionsField method\), 262](#)
[process_formdata\(\) \(indico.modules.events.papers.fields.PaperEmailSettingsField method\), 266](#)
[process_formdata\(\) \(indico.modules.networks.fields.MultiIPNetworkField method\), 267](#)
[process_formdata\(\) \(indico.web.forms.fields.EmailListField method\), 270](#)
[process_formdata\(\) \(indico.web.forms.fields.FileField method\), 273](#)
[process_formdata\(\) \(indico.web.forms.fields.HiddenEnumField method\), 260](#)
[process_formdata\(\) \(indico.web.forms.fields.HiddenTextField method\), 269](#)
[process_formdata\(\) \(indico.web.forms.fields.IndicoDateTimeField method\), 272](#)
[process_formdata\(\) \(indico.web.forms.fields.IndicoLocationField method\), 275](#)
[process_formdata\(\) \(indico.web.forms.fields.IndicoMultipleChoiceField method\), 276](#)
[process_formdata\(\) \(indico.web.forms.fields.IndicoWeekDayRepetitionField method\), 277](#)
[process_formdata\(\) \(indico.web.forms.fields.JSONField method\), 268](#)
[process_formdata\(\) \(indico.web.forms.fields.MultipleItemsField method\), 274](#)
[process_formdata\(\) \(indico.web.forms.fields.MultiStringField method\), 274](#)
[process_formdata\(\) \(indico.web.forms.fields.OccurrencesField method\), 272](#)
[process_formdata\(\) \(indico.web.forms.fields.OverrideMultipleItemsField method\), 275](#)
[process_formdata\(\) \(indico.web.forms.fields.PrincipalField method\), 275](#)
[process_formdata\(\) \(indico.web.forms.fields.PrincipalListField method\), 275](#)
[process_formdata\(\) \(indico.web.forms.fields.RelativeDeltaField method\), 276](#)
[process_formdata\(\) \(indico.web.forms.fields.TextListField method\), 270](#)
[process_formdata\(\) \(indico.web.forms.fields.TimeDeltaField method\), 272](#)
[processed_by_id \(indico.modules.events.requests.models.requests.Request attribute\), 180](#)
[processed_by_user \(indico.modules.events.requests.models.requests.Request attribute\), 180](#)
[processed_dt \(indico.modules.events.requests.models.requests.Request attribute\), 180](#)

attribute), 180
 prof (indico.modules.users.models.users.UserTitle attribute), 215
 proposal (indico.modules.events.models.reviews.ProposalReviewMixin attribute), 91
 proposal_attr (indico.modules.events.models.reviews.ProposalReviewMixin attribute), 91
 proposal_attr (indico.modules.events.papers.models.revisions.PaperRevision attribute), 149
 proposal_type (indico.modules.events.abstracts.models.abstracts.Abstract attribute), 102
 proposal_type (indico.modules.events.models.reviews.ProposalReviewMixin attribute), 90
 proposal_type (indico.modules.events.papers.models.papers.Paper attribute), 145
 ProposalCommentMixin (class in indico.modules.events.models.reviews), 89
 ProposalGroupProxy (class in indico.modules.events.models.reviews), 89
 ProposalMixin (class in indico.modules.events.models.reviews), 90
 ProposalReviewMixin (class in indico.modules.events.models.reviews), 91
 ProposalRevisionMixin (class in indico.modules.events.models.reviews), 91
 proposed_action (indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 110
 proposed_action (indico.modules.events.papers.models.revisions.PaperRevision attribute), 147
 proposed_contribution_type (in indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 110
 proposed_contribution_type_id (in indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 110
 proposed_related_abstract (in indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 110
 proposed_related_abstract_id (in indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 110
 proposed_tracks (indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 110
 protection_changed (in module indico.core.signals.acl), 44
 protection_mode (indico.modules.attachments.models.attachments.Attachment attribute), 220
 protection_mode (indico.modules.attachments.models.folders.AttachmentFolder attribute), 223
 protection_mode (indico.modules.categories.models.categories.Category attribute), 206
 protection_mode (indico.modules.events.contributions.models.contributions.Contribution attribute), 121
 protection_mode (indico.modules.events.models.events.Event attribute), 84
 protection_mode (indico.modules.events.sessions.models.sessions.Session attribute), 184
 protection_mode (indico.modules.attachments.models.attachments.Attachment attribute), 220
 protection_mode (indico.modules.attachments.models.folders.AttachmentFolder attribute), 223
 protection_mode (indico.modules.categories.models.categories.Category attribute), 206
 protection_mode (indico.modules.events.contributions.models.contributions.Contribution attribute), 121
 protection_mode (indico.modules.events.models.events.Event attribute), 84
 protection_mode (indico.modules.events.sessions.models.sessions.Session attribute), 184
 provider (indico.modules.auth.models.identities.Identity attribute), 242
 provider (indico.modules.events.payment.models.transactions.PaymentTransaction attribute), 155
 proxied_attr (indico.modules.events.papers.models.papers.Paper attribute), 145
 proxy (indico.modules.groups.models.groups.LocalGroup attribute), 247
 proxy_to_reservation_if_last_valid_occurrence() (in module indico.modules.rb.models.util), 239
 public_paper_review_enabled (in indico.modules.events.registration.models.forms.RegistrationForm attribute), 166
 public_review_registration_count (in indico.modules.events.registration.models.forms.RegistrationForm attribute), 166
 public_review_registrations_enabled (in indico.modules.events.registration.models.forms.RegistrationForm attribute), 166
 public_review_registrations (in indico.modules.events.models.events.Event attribute), 84

Q

query (indico.modules.categories.settings.CategorySettingsProxy attribute), 210
 query (indico.modules.events.settings.EventSettingsProxy attribute), 99, 153
 query (indico.modules.users.models.settings.UserSettingsProxy attribute), 217
 query_active_surveys() (in module indico.modules.events.surveys.util), 195
 question (indico.modules.events.papers.models.review_ratings.AbstractReview attribute), 109
 question (indico.modules.events.papers.models.review_ratings.PaperReview attribute), 146

question (indico.modules.events.surveys.models.items.SurveyFamilyType attribute), 191

question (indico.modules.events.surveys.models.submissions.SurveyAnswer attribute), 126

question_class (indico.modules.events.abstracts.models.review_ratings.AbstractReviewRating attribute), 109

question_class (indico.modules.events.papers.models.review_ratings.PaperReviewRating attribute), 146

question_id (indico.modules.events.abstracts.models.review_ratings.AbstractReviewRating attribute), 109

question_id (indico.modules.events.papers.models.review_ratings.PaperReviewRating attribute), 146

question_id (indico.modules.events.surveys.models.submissions.SurveyAnswer attribute), 126

questions (indico.modules.events.surveys.models.surveys.Survey attribute), 189

R

radio_widget (indico.web.forms.fields.IndicoProtectionField attribute), 276

rating_range (indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstracts attribute), 103

rating_range (indico.modules.events.papers.models.call_for_papers.CallForPapers attribute), 141

rb_check_user_access() (in module indico.modules.rb.util), 240

rb_is_admin() (in module indico.modules.rb.util), 240

read_access (indico.modules.categories.models.principals.CategoryPrincipal attribute), 207

read_access (indico.modules.events.contributions.models.principals.ContributionPrincipal attribute), 125

read_access (indico.modules.events.models.principals.EventPrincipal attribute), 88

read_access (indico.modules.events.sessions.models.principals.SessionPrincipal attribute), 186

ready_to_open (indico.modules.events.surveys.models.surveys.SurveyState attribute), 190

real_visibility_horizon (in module indico.modules.categories.models.categories.Category attribute), 206

realm (indico.modules.events.logs.models.entries.EventLogEntry attribute), 135

reason (indico.modules.events.agreements.models.agreements.Agreement attribute), 117

reason (indico.modules.rb.models.blockings.Blocking attribute), 232

recipients (indico.modules.events.abstracts.models.email_logs.AbstractEmailLogEntry attribute), 105

recipients (indico.modules.events.reminders.models.reminders.EventReminder attribute), 178

redirect_to_login() (in module indico.modules.auth.util), 243

redirect_uris (indico.modules.oauth.models.applications.OAuthApplication attribute), 244

reference_backref_name (in module indico.modules.events.contributions.models.references.ContributionReference attribute), 126

reference_backref_name (in module indico.modules.events.models.references.EventReference attribute), 126

reference_backref_name (in module indico.modules.events.models.references.ReferenceModelBase attribute), 88

reference_type (indico.modules.events.contributions.models.references.ContributionReference attribute), 126

reference_type (indico.modules.events.models.references.EventReference attribute), 88

reference_type (indico.modules.events.models.references.ReferenceModelBase attribute), 88

reference_type_id (indico.modules.events.contributions.models.references.ContributionReference attribute), 126

reference_type_id (indico.modules.events.models.references.EventReference attribute), 88

reference_type_id (indico.modules.events.models.references.ReferenceModelBase attribute), 88

ReferenceModelBase (class in indico.modules.events.models.references), 88

ReferencesField (class in indico.modules.events.fields), 262

ReferenceType (class in indico.modules.events.models.references), 89

register_assets() (indico.core.plugins.IndicoPlugin method), 40

register_css_bundle() (indico.core.plugins.IndicoPlugin method), 40

register_event_time_change() (in module indico.modules.events.util), 96

register_js_bundle() (indico.core.plugins.IndicoPlugin method), 40

register_link_events() (in module indico.modules.vc.models.vc_rooms.VCRoomEventAssociation class method), 249

register_login() (indico.modules.auth.models.identities.Identity attribute), 244

method), 242

register_time_change() (in module in-
dico.modules.events.util), 96

register_transaction() (in module in-
dico.modules.events.payment.util), 156

register_user() (in module indico.modules.auth.util), 243

registered (in module indico.core.signals.users), 49

RegisterLinkPlaceholder (class in in-
dico.modules.events.persons.placeholders),
158

Registration (class in in-
dico.modules.events.registration.models.registration),
158

registration (indico.modules.events.payment.models.transactions.PaymentTransaction attribute), 155

registration (indico.modules.events.registration.models.invitations.RegistrationInvitation attribute), 168

registration_deleted (in module in-
dico.core.signals.event), 46

registration_form_created (in module in-
dico.core.signals.event), 46

registration_form_deleted (in module in-
dico.core.signals.event), 46

registration_form_id (in-
dico.modules.events.registration.models.form_fields.RegistrationFormField attribute), 163

registration_form_id (in-
dico.modules.events.registration.models.form_fields.RegistrationFormPersonalDataField attribute), 164

registration_form_id (in-
dico.modules.events.registration.models.invitations.RegistrationInvitation attribute), 168

registration_form_id (in-
dico.modules.events.registration.models.items.RegistrationFormItem attribute), 169

registration_form_id (in-
dico.modules.events.registration.models.items.RegistrationFormPersonalDataSection attribute), 170

registration_form_id (in-
dico.modules.events.registration.models.items.RegistrationFormSection attribute), 171

registration_form_id (in-
dico.modules.events.registration.models.items.RegistrationFormText attribute), 172

registration_form_id (in-
dico.modules.events.registration.models.registrations.Registration attribute), 160

registration_id (indico.modules.events.payment.models.transactions.PaymentTransaction attribute), 155

registration_id (indico.modules.events.registration.models.invitations.RegistrationInvitation attribute), 168

registration_id (indico.modules.events.registration.models.registrations.Registration attribute), 162

registration_limit (indico.modules.events.registration.models.registrations.Registration attribute), 166

registration_personal_data_modified (in module in-
dico.core.signals.event), 46

registration_requested (in module in-
dico.core.signals.users), 49

registration_state_updated (in module in-
dico.core.signals.event), 46

RegistrationAddressPlaceholder (class in in-
dico.modules.designer.placeholders), 258

RegistrationAffiliationPlaceholder (class in in-
dico.modules.designer.placeholders), 257

RegistrationAmountPlaceholder (class in in-
dico.modules.designer.placeholders), 257

RegistrationCityPlaceholder (class in in-
dico.modules.designer.placeholders), 258

RegistrationDateInvitation (class in in-
dico.modules.events.registration.models.registrations),
161

RegistrationEmailPlaceholder (class in in-
dico.modules.designer.placeholders), 257

RegistrationFirstNamePlaceholder (class in in-
dico.modules.designer.placeholders), 256

RegistrationForm (class in in-
dico.modules.events.registration.models.forms),
160

RegistrationFormField (class in in-
dico.modules.events.registration.models.form_fields),
163

RegistrationFormPersonalDataField (class in in-
dico.modules.events.registration.models.form_fields),
164

RegistrationFormFieldData (class in in-
dico.modules.events.registration.models.form_fields),
163

RegistrationInvitation (class in in-
dico.modules.events.registration.models.items),
169

RegistrationFormItem (class in in-
dico.modules.events.registration.models.items),
169

RegistrationFormItemType (class in in-
dico.modules.events.registration.models.items),
169

RegistrationFormPersonalDataSection (class in in-
dico.modules.events.registration.models.form_fields),
164

RegistrationFormPersonalDataField (class in in-
dico.modules.events.registration.models.form_fields),
164

RegistrationFormSection (class in in-
dico.modules.events.registration.models.items),
169

RegistrationFormText (class in in-
dico.modules.events.registration.models.items),
169

RegistrationFormSection (class in in-
dico.modules.events.registration.models.items),
169

RegistrationFormText (class in in-
dico.modules.events.registration.models.items),
169

RegistrationFormTitlePlaceholder (class in in-
dico.modules.designer.placeholders), 255

RegistrationFormTitlePlaceholderB (class in in-
dico.modules.designer.placeholders), 256

RegistrationFormTitlePlaceholderC (class in in-

- dico.modules.designer.placeholders), 256
- RegistrationFullNameNoTitlePlaceholderD (class in dico.modules.designer.placeholders), 256
- RegistrationFullNamePlaceholder (class in dico.modules.designer.placeholders), 255
- RegistrationFullNamePlaceholderB (class in dico.modules.designer.placeholders), 255
- RegistrationFullNamePlaceholderC (class in dico.modules.designer.placeholders), 256
- RegistrationFullNamePlaceholderD (class in dico.modules.designer.placeholders), 256
- RegistrationInvitation (class in dico.modules.events.registration.models.invitations), 167
- RegistrationLastNamePlaceholder (class in dico.modules.designer.placeholders), 257
- RegistrationPhonePlaceholder (class in dico.modules.designer.placeholders), 258
- RegistrationPositionPlaceholder (class in dico.modules.designer.placeholders), 257
- RegistrationPricePlaceholder (class in dico.modules.designer.placeholders), 257
- RegistrationRequest (class in dico.modules.auth.models.registration_requests), 242
- registrations (indico.modules.events.registration.models.forms.RegistrationForm attribute), 166
- RegistrationSettingsProxy (class in dico.modules.events.registration.settings), 175
- RegistrationState (class in dico.modules.events.registration.models.registrations), 162
- RegistrationTicketQRPlaceholder (class in dico.modules.designer.placeholders), 257
- RegistrationTitlePlaceholder (class in dico.modules.designer.placeholders), 256
- reject (indico.modules.events.abstracts.models.reviews.AbstractAction class attribute), 109
- reject (indico.modules.events.papers.models.reviews.PaperAction class attribute), 146
- reject (indico.modules.events.payment.models.transactions.TransactionAction class attribute), 155
- reject() (indico.modules.events.agreements.models.agreements.Agreement class method), 117
- reject() (indico.modules.events.requests.base.RequestDefinitionBase class method), 181
- reject() (indico.modules.rb.models.blocked_rooms.BlockedRoom class method), 233
- reject() (indico.modules.rb.models.reservation_occurrences.ReservationOccurrence class method), 239
- reject() (indico.modules.rb.models.reservations.Reservation class method), 238
- rejected (indico.modules.events.abstracts.models.abstracts.AbstractState attribute), 102
- rejected (indico.modules.events.agreements.models.agreements.Agreement attribute), 117
- rejected (indico.modules.events.agreements.models.agreements.Agreement attribute), 118
- rejected (indico.modules.events.papers.models.revisions.PaperRevisionState attribute), 149
- rejected (indico.modules.events.payment.models.transactions.TransactionState attribute), 155
- rejected (indico.modules.events.registration.models.registrations.Registration attribute), 162
- rejected (indico.modules.events.requests.models.requests.RequestState attribute), 180
- rejected (indico.modules.rb.models.blocked_rooms.BlockedRoomState attribute), 233
- rejected_by (indico.modules.rb.models.blocked_rooms.BlockedRoom attribute), 233
- rejected_on_behalf (indico.modules.events.agreements.models.agreements.Agreement attribute), 118
- rejection_reason (indico.modules.rb.models.blocked_rooms.BlockedRoom attribute), 233
- rejection_reason (indico.modules.rb.models.reservation_occurrences.ReservationOccurrence attribute), 239
- rejection_reason (indico.modules.rb.models.reservations.Reservation attribute), 238
- RelativeDeltaField (class in indico.web.forms.fields), 276
- remove_principal() (indico.modules.events.settings.EventACLProxy method), 98, 152
- render() (indico.modules.designer.placeholders.CategoryTitlePlaceholder class method), 258
- render() (indico.modules.designer.placeholders.EventDatesPlaceholder class method), 255
- render() (indico.modules.designer.placeholders.EventDescriptionPlaceholder class method), 255
- render() (indico.modules.designer.placeholders.EventOrgTextPlaceholder class method), 255
- render() (indico.modules.designer.placeholders.EventRoomPlaceholder class method), 258
- render() (indico.modules.designer.placeholders.EventSpeakersPlaceholder class method), 259
- render() (indico.modules.designer.placeholders.EventTitlePlaceholder class method), 258
- render() (indico.modules.designer.placeholders.EventVenuePlaceholder class method), 259
- render() (indico.modules.designer.placeholders.RegistrationAmountPlaceholder class method), 257
- render() (indico.modules.designer.placeholders.RegistrationPricePlaceholder class method), 257
- render() (indico.modules.designer.placeholders.RegistrationTicketQRPlaceholder class method), 257

render() (indico.modules.events.abstracts.placeholders.AbstractIDPlaceholder class method), 113

render() (indico.modules.events.abstracts.placeholders.AbstractTitlePlaceholder class method), 113

render() (indico.modules.events.abstracts.placeholders.AbstractURLPlaceholder class method), 113

render() (indico.modules.events.abstracts.placeholders.AbstractTypePlaceholder class method), 116

render() (indico.modules.events.abstracts.placeholders.EventTitlePlaceholder class method), 113

render() (indico.modules.events.abstracts.placeholders.EventURLPlaceholder class method), 113

render() (indico.modules.events.abstracts.placeholders.JudgmentPlaceholder class method), 116

render() (indico.modules.events.abstracts.placeholders.PrimaryAuthProviderPlaceholder class method), 114

render() (indico.modules.events.abstracts.placeholders.SubmitterNamePlaceholder class method), 114

render() (indico.modules.events.abstracts.placeholders.SubmitterURLPlaceholder class method), 115

render() (indico.modules.events.abstracts.placeholders.SubmitterAvatarPlaceholder class method), 114

render() (indico.modules.events.abstracts.placeholders.SubmitterTitlePlaceholder class method), 115

render() (indico.modules.events.abstracts.placeholders.TargetAbstractIDPlaceholder class method), 115

render() (indico.modules.events.abstracts.placeholders.TargetAbstractTitlePlaceholder class method), 115

render() (indico.modules.events.abstracts.placeholders.TargetSubmitterNamePlaceholder class method), 115

render() (indico.modules.events.abstracts.placeholders.TargetSubmitterURLPlaceholder class method), 115

render() (indico.modules.events.abstracts.placeholders.TargetSubmitterAvatarPlaceholder class method), 115

render() (indico.modules.events.agreements.models.agreements.Agreement class method), 117

render() (indico.modules.events.agreements.placeholders.AgreementLinkPlaceholder class method), 119

render() (indico.modules.events.agreements.placeholders.PersonNamePlaceholder class method), 119

render() (indico.modules.events.logs.models.entries.EventLogEntry class method), 135

render() (indico.modules.events.persons.placeholders.EmailPlaceholder class method), 157

render() (indico.modules.events.persons.placeholders.EventLinkPlaceholder class method), 157

render() (indico.modules.events.persons.placeholders.EventTitlePlaceholder class method), 158

render() (indico.modules.events.persons.placeholders.FirstNamePlaceholder class method), 158

render() (indico.modules.events.persons.placeholders.LastNamePlaceholder class method), 158

render() (indico.modules.events.persons.placeholders.RegisterLinkPlaceholder class method), 158

render() (indico.modules.events.registration.placeholders.invitations.FirstNamePlaceholder class method), 175

render() (indico.modules.events.registration.placeholders.invitations.InvitationTypePlaceholder class method), 175

render() (indico.modules.events.registration.placeholders.invitations.LastNamePlaceholder class method), 175

render() (indico.modules.events.registration.placeholders.registrations.EventURLPlaceholder class method), 174

render() (indico.modules.events.registration.placeholders.registrations.EventTitlePlaceholder class method), 174

render() (indico.modules.events.registration.placeholders.registrations.FieldURLPlaceholder class method), 174

render() (indico.modules.events.registration.placeholders.registrations.FirstNamePlaceholder class method), 174

render() (indico.modules.events.registration.placeholders.registrations.IDPlaceholder class method), 174

render() (indico.modules.events.registration.placeholders.registrations.LastNamePlaceholder class method), 175

render() (indico.modules.events.registration.placeholders.registrations.LinkPlaceholder class method), 175

render() (indico.modules.events.registration.models.forms.RegistrationForm class method), 166

render_base_price() (in module indico.modules.events.registration.models.registrations.RegistrationForm class method), 160

render_info_balloon() (in module indico.modules.vc.plugins.VCPluginMixin class method), 251

render_info_box() (in module indico.modules.vc.plugins.VCPluginMixin class method), 251

render_manage_event_info_box() (in module indico.modules.vc.plugins.VCPluginMixin class method), 251

render_form() (in module indico.modules.events.requests.base.RequestDefinitionBase class method), 182

render_form() (in module indico.modules.vc.plugins.VCPluginMixin class method), 251

render_info_box() (in module indico.modules.vc.plugins.VCPluginMixin class method), 251

- dico.modules.vc.plugins.VCPluginMixin (class in indico.modules.vc.plugins), 252
- render_mode (indico.modules.categories.models.categories.Category attribute), 206
- render_mode (indico.modules.events.abstracts.models.abstracts.RepeatFrequency attribute), 102
- render_mode (indico.modules.events.abstracts.models.comments.RepeatFrequency attribute), 104
- render_mode (indico.modules.events.abstracts.models.reviews.RepeatFrequency attribute), 110
- render_mode (indico.modules.events.contributions.models.contributions.RepeatFrequency attribute), 121
- render_mode (indico.modules.events.contributions.models.subcontributions.RepeatFrequency attribute), 127
- render_mode (indico.modules.events.models.events.Event attribute), 84
- render_mode (indico.modules.events.notes.models.notes.EventNote attribute), 140
- render_mode (indico.modules.events.papers.models.comments.Paper attribute), 142
- render_mode (indico.modules.events.papers.models.reviews.Paper attribute), 147
- render_mode (indico.modules.events.papers.models.revisions.Paper attribute), 149
- render_mode (indico.modules.events.sessions.models.sessions.Session attribute), 184
- render_mode (indico.modules.events.timetable.models.breaks.Break attribute), 196
- render_mode (indico.modules.events.tracks.models.tracks.Track attribute), 202
- render_payment_form() (in module indico.modules.events.payment.plugins.PaymentPluginMixin), 157
- render_price() (indico.modules.events.registration.models.registrations.Registration attribute), 160
- render_price() (indico.modules.events.registration.models.registrations.Registration attribute), 162
- render_price_adjustment() (in module indico.modules.events.registration.models.registrations.Registration), 160
- render_protection_message() (in module indico.web.forms.fields.IndicoProtectionField), 276
- render_session_timetable() (in module indico.modules.events.timetable.util), 200
- render_transaction_details() (in module indico.modules.events.payment.plugins.PaymentPluginMixin), 157
- renderer (indico.modules.events.logs.models.entries.EventLogEntry attribute), 135
- repeat_frequency (indico.modules.rb.models.reservations.Reservations attribute), 238
- repeat_interval (indico.modules.rb.models.reservations.Reservations attribute), 238
- RepeatFrequency (class in indico.modules.rb.models.reservations), 236
- RepeatMapping (class in indico.modules.rb.models.reservations), 236
- repeat_reservation (indico.modules.rb.models.reservations.Reservation attribute), 238
- reply_abstract_email (indico.modules.events.abstracts.models.email_templates.EmailTemplate attribute), 106
- reply_abstract_review (indico.modules.events.reminders.models.reminders.Event attribute), 178
- Requires (class in indico.modules.events.requests.models.requests), 170
- RequestDefinitionBase (class in indico.modules.events.requests.base), 180
- requested_dt (indico.modules.events.static.models.static.StaticSite attribute), 203
- RequestState (class in indico.modules.events.requests.models.requests), 180
- ReviewComment (class in module indico.modules.events.features.util), 130
- review_session (indico.modules.events.registration.models.forms.Registration attribute), 166
- review_session (indico.modules.events.registration.models.forms.Registration attribute), 166
- review_user (indico.modules.events.surveys.models.surveys.Survey attribute), 189
- required (indico.modules.events.agreements.placeholders.AgreementLinkPlaceholder attribute), 119
- required (indico.modules.events.registration.placeholders.invitations.Invitation attribute), 175
- RescheduleMode (class in indico.modules.events.timetable.reschedule), 200
- RescheduleRegistrationData (class in indico.modules.events.timetable.reschedule), 200
- ResRegistration (class in indico.modules.rb.models.reservations), 236
- reservation_id (indico.modules.rb.models.reservation_edit_logs.Reservation attribute), 238
- reservation_id (indico.modules.rb.models.reservation_occurrences.Reservation attribute), 239
- ReservationEditLog (class in indico.modules.rb.models.reservation_edit_logs), 238
- ReservationOccurrence (class in indico.modules.rb.models.reservation_occurrences), 239
- reservations (indico.modules.rb.models.rooms.Room attribute), 230
- reservations_need_confirmation (in module indico.modules.rb.models.rooms.Room), 230

tribute), 230

reset() (indico.modules.events.agreements.models.agreements.Agreement attribute), 117

reset_abstract_state() (in module indico.modules.events.abstracts.operations), 111

reset_client_secret() (in module indico.modules.oauth.models.applications.OAuthApplication attribute), 244

reset_paper_state() (in module indico.modules.events.papers.operations), 150

reset_state() (indico.modules.events.abstracts.models.abstracts.Abstract attribute), 102

reset_state() (indico.modules.events.papers.models.papers.Paper attribute), 145

resolve_title() (in module indico.modules.vc.util), 250

review (indico.modules.events.abstracts.models.review_ratings.AbstractReviewRating attribute), 109

review (indico.modules.events.papers.models.review_ratings.PaperReviewRating attribute), 146

review_class (indico.modules.events.abstracts.models.review_ratings.AbstractReviewRating attribute), 109

review_class (indico.modules.events.papers.models.review_ratings.PaperReviewRating attribute), 146

review_id (indico.modules.events.abstracts.models.review_ratings.AbstractReviewRating attribute), 109

review_id (indico.modules.events.papers.models.review_ratings.PaperReviewRating attribute), 146

reviewed_for (indico.modules.events.abstracts.models.abstracts.EditTrackModel attribute), 103

reviewed_for_tracks (in module indico.modules.events.abstracts.models.abstracts.Abstract attribute), 102

reviewers (indico.modules.events.abstracts.models.reviews.AbstractCommentVisibility attribute), 109

reviewers (indico.modules.events.papers.models.reviews.PaperCommentVisibility attribute), 146

reviewing (indico.modules.events.logs.models.entries.EventLogRealm attribute), 136

reviewing_instructions (in module indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstract attribute), 103

reviewing_state (indico.modules.events.abstracts.models.abstracts.Abstract attribute), 102

ReviewQuestionsField (class in module indico.modules.events.fields), 262

revision (indico.modules.events.models.reviews.ProposalReview attribute), 91

revision (indico.modules.events.papers.models.reviews.PaperReview attribute), 147

revision_attr (indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 110

revision_attr (indico.modules.events.models.reviews.ProposalReview attribute), 91

revision_attr (indico.modules.events.papers.models.reviews.PaperReview attribute), 147

revision_count (indico.modules.events.papers.models.papers.Paper attribute), 145

revision_id (indico.modules.events.papers.models.comments.PaperReview attribute), 142

revision_id (indico.modules.events.papers.models.files.PaperFile attribute), 144

revision_id (indico.modules.events.papers.models.reviews.PaperReview attribute), 147

revisions (indico.modules.events.notes.models.notes.EventNote attribute), 139

revisions (indico.modules.events.papers.models.papers.Paper attribute), 145

revisions_enabled (indico.modules.events.abstracts.models.abstracts.Abstract attribute), 102

revisions_enabled (indico.modules.events.papers.models.papers.Paper attribute), 145

revisions_enabled (indico.modules.events.models.reviews.Proposal attribute), 91

revisions_enabled (indico.modules.events.models.reviews.Proposal attribute), 91

revisions_enabled (indico.modules.events.papers.models.papers.Paper attribute), 145

ReviewRating (class in module indico.modules.events.management.controllers), 177

role_data (indico.modules.events.abstracts.fields.TrackRoleField attribute), 165

roles (indico.modules.categories.models.principals.CategoryPrincipal attribute), 207

roles (indico.modules.events.contributions.models.principals.Contribution attribute), 125

roles (indico.modules.events.models.principals.EventPrincipal attribute), 88

roles (indico.modules.events.sessions.models.principals.SessionPrincipal attribute), 186

Room (class in module indico.modules.rb.models.rooms), 227

room_id (indico.modules.rb.models.blocked_rooms.BlockedRoom attribute), 233

room_id (indico.modules.rb.models.reservations.Reservation attribute), 238

room_id (indico.modules.rb.models.room_attributes.RoomAttributeAssociation attribute), 231

room_id (indico.modules.rb.models.room_bookable_hours.BookableHours attribute), 231

room_id (indico.modules.rb.models.room_nonbookable_periods.NonBookable attribute), 231

RoomAttribute (class in module indico.modules.rb.models.room_attributes), 230

RoomAttributeAssociation (class in module indico.modules.rb.models.room_attributes), 230

RoomAvailabilitySearchRooms (class in module indico.modules.rb.services.rooms), 240

- RoomBookingBlockingApprove (class in indico.modules.rb.services.blockings), 241
- RoomBookingBlockingProcessBase (class in indico.modules.rb.services.blockings), 241
- RoomBookingBlockingReject (class in indico.modules.rb.services.blockings), 241
- RoomBookingListLocationsAndRoomsWithGuids (class in indico.modules.rb.services.rooms), 240
- RoomBookingMapBase (class in indico.modules.rb.services.aspects), 241
- RoomBookingMapCreateAspect (class in indico.modules.rb.services.aspects), 241
- RoomBookingMapListAspects (class in indico.modules.rb.services.aspects), 241
- RoomBookingMapRemoveAspect (class in indico.modules.rb.services.aspects), 241
- RoomBookingMapUpdateAspect (class in indico.modules.rb.services.aspects), 241
- rooms (indico.modules.rb.models.locations.Location attribute), 235
- rules (indico.modules.events.abstracts.models.email_templates.AbstractEmailTemplate attribute), 106
- run() (indico.modules.events.timetable.reschedule.Rescheduler method), 201
- running (indico.modules.events.static.models.static.StaticSiteState attribute), 203
- S**
- safe_last_login_dt (indico.modules.auth.models.identities.Identity attribute), 242
- save() (indico.modules.oauth.models.tokens.OAuthGrant method), 245
- save() (indico.modules.users.ext.ExtraUserPreferences method), 219
- save_grant() (in module indico.modules.oauth.provider), 246
- save_identity_info() (in module indico.modules.auth.util), 243
- save_submitted_survey_to_session() (in module indico.modules.events.surveys.util), 195
- save_token() (in module indico.modules.oauth.provider), 246
- schedule (indico.modules.events.abstracts.settings.BOASortField attribute), 116
- schedule() (indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstract method), 104
- schedule() (indico.modules.events.papers.models.call_for_papers.CallForPapers method), 141
- schedule_cfa() (in module indico.modules.events.abstracts.operations), 111
- schedule_cfp() (in module indico.modules.events.papers.operations), 150
- schedule_contribution() (in module indico.modules.events.timetable.operations), 198
- scheduled_dt (indico.modules.events.reminders.models.reminders.EventReminder attribute), 178
- scheduled_notes (indico.modules.events.models.events.Event attribute), 84
- scheme (indico.modules.events.models.references.ReferenceType attribute), 89
- scopes (indico.modules.oauth.models.tokens.OAuthToken attribute), 245
- score (indico.modules.events.abstracts.models.abstracts.Abstract attribute), 102
- score (indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 110
- score (indico.modules.events.models.reviews.ProposalReviewMixin attribute), 91
- score (indico.modules.events.papers.models.reviews.PaperReview attribute), 147
- score (indico.modules.users.models.suggestions.SuggestedCategory attribute), 106
- search() (indico.modules.groups.core.GroupProxy class method), 247
- search_payload (indico.modules.events.abstracts.fields.AbstractField attribute), 262
- search_url (indico.modules.events.abstracts.fields.AbstractField attribute), 262
- search_users() (in module indico.modules.users.util), 218
- secondary (indico.modules.events.contributions.models.persons.AuthorType attribute), 124
- secondary_authors (indico.modules.events.models.persons.AuthorsSpeakers attribute), 85
- secondary_emails (indico.modules.users.models.users.User attribute), 214
- secondary_local_identities (indico.modules.users.models.users.User attribute), 214
- section (indico.modules.events.registration.models.items.RegistrationFormItem attribute), 170
- section (indico.modules.events.surveys.models.items.SurveyItemType attribute), 191
- section_pd (indico.modules.events.registration.models.items.RegistrationFormItem attribute), 170
- sections (in module indico.core.signals.menu), 48
- sections (indico.modules.events.registration.models.forms.RegistrationForm attribute), 166
- sections (indico.modules.events.surveys.models.surveys.Survey attribute), 189
- sections_with_answered_fields (indico.modules.events.registration.models.registrations.RegistrationForm attribute), 160
- send() (indico.modules.events.reminders.models.reminders.EventReminder method), 178
- send() (indico.modules.events.requests.base.RequestDefinitionBase method), 178

class method), 182

send_new_agreements() (in module indico.modules.events.agreements.util), 118

send_start_notification() (in module indico.modules.events.surveys.models.surveys.Survey method), 189

send_submission_notification() (in module indico.modules.events.surveys.models.surveys.Survey method), 189

send_to_participants (in module indico.modules.events.reminders.models.reminders.EventReminder attribute), 178

sender_address (indico.modules.events.registration.models.forms.RegistrationForm attribute), 166

sent_dt (indico.modules.events.abstracts.models.email_logs.AbstractEmailLog attribute), 105

separator (indico.modules.events.layout.models.menu.MenuEntryType attribute), 133

serialize_categories_ical() (in module indico.modules.categories.serialize), 209

serialize_category() (in module indico.modules.categories.serialize), 209

serialize_category_atom() (in module indico.modules.categories.serialize), 209

serialize_category_chain() (in module indico.modules.categories.serialize), 209

serialize_contribution_for_ical() (in module indico.modules.events.contributions.util), 129

serialize_contribution_person_link() (in module indico.modules.events.contributions.util), 129

serialize_event_for_ical() (in module indico.modules.events.util), 96

serialize_event_person() (in module indico.modules.events.util), 96

serialize_group() (in module indico.modules.groups.util), 248

serialize_ip_network_group() (in module indico.modules.networks.util), 260

serialize_person_link() (in module indico.modules.events.util), 96

serialize_session_for_ical() (in module indico.modules.events.sessions.util), 187

serialize_user() (in module indico.modules.users.util), 219

series (indico.modules.events.models.events.Event attribute), 84

series_id (indico.modules.events.models.events.Event attribute), 84

service_name (indico.modules.vc.plugins.VCPluginMixin attribute), 252

Session (class in indico.modules.events.sessions.models.sessions), 182

session (indico.modules.attachments.models.folders.AttachmentFolder attribute), 223

session (indico.modules.events.contributions.models.contributions.Contribution attribute), 121

session (indico.modules.events.contributions.models.subcontributions.SubContribution attribute), 127

session (indico.modules.events.notes.models.notes.EventNote attribute), 139

session (indico.modules.events.sessions.models.sessions.Session attribute), 184

session_block (indico.modules.events.contributions.models.contributions.Contribution attribute), 121

SessionBlock (class in indico.modules.events.sessions.models.blocks), 184

SessionBlockPersonLink (class in indico.modules.events.sessions.models.persons), 185

SessionBlockPersonLinkListField (class in indico.modules.events.sessions.fields), 266

SessionListToPDF (class in indico.modules.events.sessions.util), 187

SessionPrincipal (class in indico.modules.events.sessions.models.principals), 186

session_block_deleted (in module indico.core.signals.event), 47

session_block_id (indico.modules.events.contributions.models.contributions.Contribution attribute), 121

session_block_id (indico.modules.events.sessions.models.persons.SessionBlock attribute), 186

session_block_id (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 197

session_block_id (indico.modules.vc.models.vc_rooms.VCRoomEventAssociation attribute), 250

session_coordinator_priv_enabled() (in module indico.modules.events.sessions.util), 187

session_deleted (in module indico.core.signals.event), 47

session_id (indico.modules.attachments.models.folders.AttachmentFolder attribute), 223

session_id (indico.modules.events.contributions.models.contributions.Contribution attribute), 121

session_id (indico.modules.events.notes.models.notes.EventNote attribute), 139

session_id (indico.modules.events.sessions.models.blocks.SessionBlock attribute), 185

session_id (indico.modules.events.sessions.models.principals.SessionPrincipal attribute), 186

session_siblings (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 197

session_title (indico.modules.events.abstracts.settings.BOASortField attribute), 116

session_updated (in module indico.core.signals.event), 47

dico.modules.events.sessions.models.principals), 186

set() (indico.modules.categories.settings.CategorySettingsProxy method), 210

set() (indico.modules.events.settings.EventACLProxy method), 98, 152

set() (indico.modules.events.settings.EventSettingsProxy method), 99, 153

set() (indico.modules.users.models.settings.UserSettingsProxy method), 217

set_attribute_value() (in dico.modules.rb.models.rooms.Room method), 230

set_custom_field() (indico.modules.events.contributions.models.contributions.models.custom_fields.models.CustomFieldMixin method), 122

set_custom_fields() (in module in-dico.modules.events.util), 97

set_deadline() (in module in-dico.modules.events.papers.operations), 151

set_default() (indico.modules.rb.models.locations.Location method), 235

set_feature_enabled() (in module in-dico.modules.events.features.util), 130

set_multi() (indico.modules.categories.settings.CategorySettingsProxy method), 210

set_multi() (indico.modules.events.settings.EventSettingsProxy method), 99, 153

set_multi() (indico.modules.users.models.settings.UserSettingsProxy method), 217

set_participant_list_columns() (in dico.modules.events.registration.settings.RegistrationSettingsProxy method), 176

set_participant_list_form_ids() (in dico.modules.events.registration.settings.RegistrationSettingsProxy method), 176

set_reviewing_state() (in module in-dico.modules.events.papers.operations), 151

set_reviewing_state() (in dico.modules.events.papers.models.call_for_papers.models.CallForPapers method), 141

settings (indico.core.plugins.IndicoPlugin attribute), 40

settings (indico.modules.auth.models.registration_requests.RegistrationRequests module attribute), 243

settings (indico.modules.events.settings.ThemeSettingsProxy attribute), 99, 154

settings (indico.modules.users.models.users.User attribute), 214

settings_backref_name (in dico.modules.events.models.settings.EventSettings attribute), 92

settings_backref_name (in dico.modules.events.models.settings.EventSettingPrincipal attribute), 93

settings_backref_name (in dico.modules.events.models.settings.EventSettingsMixin attribute), 93

settings_converters (indico.core.plugins.IndicoPlugin attribute), 41

settings_form (indico.core.plugins.IndicoPlugin attribute), 41

settings_form (indico.modules.events.payment.plugins.PaymentPluginMixin attribute), 157

settings_form (indico.modules.vc.plugins.VCPluginMixin attribute), 252

settings_form_field_opts (in-dico.core.plugins.IndicoPlugin attribute), 41

shell_contributions_custom_fields_mixins.signals.plugin), 49

shift_following_entries() (in module in-dico.modules.events.timetable.util), 200

short_external_url (indico.modules.events.models.events.Event attribute), 84

short_title (indico.modules.events.tracks.models.tracks.Track attribute), 202

short_url (indico.modules.events.models.events.Event attribute), 84

show (indico.modules.vc.models.vc_rooms.VCRoomEventAssociation attribute), 250

show_links (indico.modules.events.models.series.EventSeries attribute), 92

show_sequence_in_title (in-dico.modules.events.models.series.EventSeries attribute), 92

siblings (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 198

siblings_query (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 198

signed_dt (indico.modules.events.agreements.models.agreements.Agreement attribute), 117

signed_from_ip (indico.modules.events.agreements.models.agreements.Agreement attribute), 118

signed_from_ip_half (indico.modules.events.agreements.models.agreements.Agreement attribute), 118

SimpleRenderer (class in in-dico.modules.events.logs.renderers), 137

site (indico.modules.rb.models.rooms.Room attribute), 230

size (indico.modules.attachments.models.attachments.AttachmentFile attribute), 221

size (indico.modules.designer.models.images.DesignerImageFile attribute), 253

size (indico.modules.events.abstracts.models.files.AbstractFile attribute), 107

size (indico.modules.events.layout.models.images.ImageFile attribute), 131

size (indico.modules.events.papers.models.files.PaperFile attribute), 144

size (indico.modules.events.papers.models.templates.PaperTemplateLocal (indico.modules.events.models.events.Event attribute), 150

size (indico.modules.events.registration.models.registrations.RegistrationDelta (indico.modules.events.models.events.Event attribute), 162

size (indico.modules.events.static.models.static.StaticSite start_dt_poster (indico.modules.events.contributions.models.contributions.Contribution attribute), 203

skip_moderation (indico.modules.events.registration.models.registrations.RegistrationInvitation (indico.modules.events.surveys.models.surveys.Survey attribute), 168

small_photo_url (indico.modules.rb.models.rooms.Room start_notification_recipients (indico.modules.events.surveys.models.surveys.Survey attribute), 230

source (indico.modules.events.notes.models.notes.EventNoteRevision (indico.modules.events.surveys.models.surveys.Survey attribute), 140

speaker (indico.modules.events.abstracts.settings.BOASortField start_notification_sent (indico.modules.events.surveys.models.surveys.Survey attribute), 116

speakers (indico.modules.events.abstracts.settings.BOACorrespondingAttributeType (indico.modules.rb.models.room_bookable_hours.BookableHours attribute), 116

speakers (indico.modules.events.contributions.models.subcontributions.SubContribution starts_between() (indico.modules.events.models.events.Event attribute), 127

speakers (indico.modules.events.models.persons.AuthorsSpeakersMixin method), 84

split_data (indico.web.forms.fields.RelativeDeltaField attribute), 276

start_date (indico.modules.rb.models.blockings.Blocking state (indico.modules.events.agreements.models.agreements.Agreement attribute), 232

start_dt (indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstracts state (indico.modules.events.papers.models.papers.Paper attribute), 104

start_dt (indico.modules.events.contributions.models.contributions.Contribution state (indico.modules.events.papers.models.revisions.PaperRevision attribute), 121

start_dt (indico.modules.events.models.events.Event attribute), 84

start_dt (indico.modules.events.registration.models.invitations.RegistrationInvitation state (indico.modules.events.registration.models.registrations.Registration attribute), 168

start_dt (indico.modules.events.papers.models.call_for_papers.CallForPapers state (indico.modules.events.requests.models.requests.Request attribute), 141

start_dt (indico.modules.events.registration.models.forms.RegistrationForm state (indico.modules.events.static.models.static.StaticSite attribute), 166

start_dt (indico.modules.events.sessions.models.blocks.SessionBlock state (indico.modules.events.surveys.models.surveys.Survey attribute), 185

start_dt (indico.modules.events.sessions.models.sessions.Session attribute), 184

start_dt (indico.modules.events.surveys.models.surveys.Survey State (indico.modules.rb.models.blocked_rooms.BlockedRoom attribute), 189

start_dt (indico.modules.events.timetable.models.breaks.Break state (indico.modules.rb.models.blocked_rooms.BlockedRoom attribute), 196

start_dt (indico.modules.events.timetable.models.entries.TimetableEntry state_name (indico.modules.rb.models.blocked_rooms.BlockedRoom attribute), 198

start_dt (indico.modules.rb.models.reservation_occurrences.ReservationOccurrence static_items (indico.modules.events.util.ListGeneratorBase attribute), 239

start_dt (indico.modules.rb.models.reservations.Reservation StaticListLink (class in indico.modules.events.models.static_list_links), attribute), 238

start_dt (indico.modules.rb.models.room_nonbookable_periods.NonBookablePeriod class in indico.modules.events.static.models.static), attribute), 231

start_dt_display (indico.modules.events.contributions.models.contributions.Contribution attribute), 121

start_dt_display (indico.modules.events.models.events.Event StaticSiteState (class in indico.modules.events.static.models.static), attribute), 84

StatsBase (class in in- attribute), 84
 (indico.modules.events.registration.stats), 177
 stylesheet_metadata (in-
 status (indico.modules.events.payment.models.transactions.PaymentTransaction (class in in-
 attribute), 155 (indico.modules.events.models.events.Event
 attribute), 84
 status (indico.modules.vc.models.vc_rooms.VCRoom at- SubContribution (class in in-
 tribute), 248 (indico.modules.events.contributions.models.subcontributions),
 status_string (indico.modules.rb.models.reservations.Reservation 126
 attribute), 238 subcontribution (indico.modules.attachments.models.folders.AttachmentFolder
 stop_on_match (indico.modules.events.abstracts.models.email_templates.EmailTemplate (class in in-
 attribute), 106 (indico.modules.events.notes.models.notes.EventNote
 subcontribution (indico.modules.events.notes.models.notes.EventNote
 storage_backend (indico.modules.attachments.models.attachments.AttachmentFile 139
 attribute), 221 subcontribution_count (in-
 storage_backend (indico.modules.designer.models.images.DesignerImageFile (indico.modules.events.contributions.models.contributions.Contribution
 attribute), 253 attribute), 122
 storage_backend (indico.modules.events.abstracts.models.files.AbstractFile_created (in module in-
 attribute), 107 (indico.core.signals.event), 47
 storage_backend (indico.modules.events.layout.models.images.ImageFile_deleted (in module in-
 attribute), 131 (indico.core.signals.event), 47
 storage_backend (indico.modules.events.papers.models.files.PaperFile_distribution_id (indico.modules.attachments.models.folders.AttachmentFolder
 attribute), 144 attribute), 223
 storage_backend (indico.modules.events.papers.models.templates.PaperTemplate (indico.modules.events.contributions.models.persons.PersonLink
 attribute), 150 attribute), 125
 storage_backend (indico.modules.events.registration.models.registrations.RegistrationData (indico.modules.events.contributions.models.references.Reference
 attribute), 162 attribute), 126
 storage_backend (indico.modules.events.static.models.static.StaticSite_distribution_id (indico.modules.events.notes.models.notes.EventNote
 attribute), 203 attribute), 139
 storage_file_id (indico.modules.attachments.models.attachments.AttachmentFile_deleted (in module in-
 attribute), 221 (indico.core.signals.event), 47
 storage_file_id (indico.modules.designer.models.images.DesignerImageFile (indico.modules.events.contributions.models.persons.PersonLink
 attribute), 253 (class in in-
 (indico.modules.events.contributions.models.persons),
 storage_file_id (indico.modules.events.abstracts.models.files.AbstractFile SubContributionPersonLinkListField (class in in-
 attribute), 107 (indico.modules.events.contributions.fields),
 storage_file_id (indico.modules.events.layout.models.images.ImageFile 265
 storage_file_id (indico.modules.events.papers.models.files.PaperFile_distributionReference (class in in-
 attribute), 144 (indico.modules.events.contributions.models.references),
 storage_file_id (indico.modules.events.papers.models.templates.PaperTemplate
 attribute), 150 subcontributions (indico.modules.events.contributions.models.contributions.models.contributions),
 storage_file_id (indico.modules.events.registration.models.registrations.RegistrationData
 attribute), 162 subject (indico.modules.events.abstracts.models.email_logs.AbstractEmailLog
 storage_file_id (indico.modules.events.static.models.static.StaticSite attribute), 105
 attribute), 203 subject (indico.modules.events.abstracts.models.email_templates.AbstractEmailTemplate
 store_configuration() (in- attribute), 106
 (in-
 (indico.modules.events.util.ListGeneratorBase submission_comment (in-
 method), 95 (indico.modules.events.abstracts.models.abstracts.Abstract
 stored_file_class (indico.modules.attachments.models.attachments.AttachmentFile 102
 attribute), 220 submission_id (indico.modules.events.surveys.models.submissions.SurveySubmission
 stored_file_fkey (indico.modules.attachments.models.attachments.AttachmentFile 193
 attribute), 221 submission_instructions (in-
 stored_file_table (indico.modules.attachments.models.attachments.AttachmentFile 104
 attribute), 221 (indico.modules.events.abstracts.models.call_for_abstracts.CallForAbstract
 attribute), 104
 strict_settings (indico.core.plugins.IndicoPlugin at- submission_limit (indico.modules.events.surveys.models.surveys.SurveySubmission
 tribute), 41 attribute), 190
 stylesheet (indico.modules.events.models.events.Event submissions (indico.modules.events.surveys.models.surveys.SurveySubmission)

attribute), 190

submitted (indico.modules.events.abstracts.models.abstracts.AbstractState attribute), 103

submitted (indico.modules.events.papers.models.revisions.PaperRevisionState attribute), 149

submitted_contrib_type (in-dico.modules.events.abstracts.models.abstracts.AbstractState attribute), 102

submitted_contrib_type_id (in-dico.modules.events.abstracts.models.abstracts.AbstractState attribute), 102

submitted_dt (indico.modules.events.abstracts.models.abstracts.AbstractState attribute), 102

submitted_dt (indico.modules.events.papers.models.revisions.PaperRevisionState attribute), 149

submitted_dt (indico.modules.events.registration.models.registrations.Registration attribute), 160

submitted_dt (indico.modules.events.surveys.models.submissions.SurveySubmission attribute), 193

submitted_for_tracks (in-dico.modules.events.abstracts.models.abstracts.AbstractState attribute), 102

submitter (indico.modules.events.abstracts.models.abstracts.AbstractState attribute), 102

submitter (indico.modules.events.abstracts.settings.BOACorrespondingAuthorType attribute), 116

submitter (indico.modules.events.papers.models.revisions.PaperRevisionState attribute), 149

submitter_id (indico.modules.events.abstracts.models.abstracts.AbstractState attribute), 102

submitter_id (indico.modules.events.papers.models.revisions.PaperRevisionState attribute), 149

SubmitterFirstNamePlaceholder (class in indico.modules.events.abstracts.placeholders), 114

SubmitterLastNamePlaceholder (class in indico.modules.events.abstracts.placeholders), 114

SubmitterNamePlaceholder (class in indico.modules.events.abstracts.placeholders), 114

submitters (indico.modules.events.contributions.models.contributions.Contribution attribute), 122

SubmitterTitlePlaceholder (class in indico.modules.events.abstracts.placeholders), 115

success (indico.modules.events.static.models.static.StaticSiteState attribute), 203

successful (indico.modules.events.payment.models.transactions.Transaction attribute), 155

suggested_categories (in-dico.modules.users.models.users.User attribute), 214

SuggestedCategory (class in indico.modules.users.models.users.users.User attribute), 214

indico.modules.users.models.suggestions),

State

suggestions_disabled (in-indico.modules.categories.models.categories.Category attribute), 206

summary (indico.modules.events.logs.models.entries.EventLogEntry attribute), 135

summary_data (indico.modules.events.registration.models.registrations.Registration attribute), 160

summary_data (indico.modules.events.registration.models.registrations.Registration attribute), 162

support_abstract() (in-indico.modules.events.payment.plugins.PaymentPluginMixin attribute), 157

surface_area (indico.modules.rb.models.rooms.Room attribute), 160

Survey (class in indico.modules.events.surveys.models.surveys), 188

survey_id (indico.modules.events.surveys.models.items.SurveyItem attribute), 190

survey_id (indico.modules.events.surveys.models.items.SurveyQuestion attribute), 191

survey_id (indico.modules.events.surveys.models.items.SurveySection attribute), 192

survey_id (indico.modules.events.surveys.models.items.SurveyText attribute), 192

survey_id (indico.modules.events.surveys.models.submissions.SurveySubmission attribute), 193

SurveyAnswer (class in indico.modules.events.surveys.models.submissions), 192

SurveyItem (class in indico.modules.events.surveys.models.items), 190

SurveyItemType (class in indico.modules.events.surveys.models.items), 191

SurveyQuestion (class in indico.modules.events.surveys.models.items), 191

SurveySection (class in indico.modules.events.surveys.models.items), 191

SurveyState (class in indico.modules.events.surveys.models.surveys), 190

SurveySubmission (class in indico.modules.events.surveys.models.submissions), 190

SurveyText (class in indico.modules.events.surveys.models.items), 192

swap_timetable_entry() (in module indico.modules.events.timetable.operations),

- 198
- sync_state() (indico.modules.events.registration.models.registrations.RegistrationRegistration method), 160
- syncable_fields (in module indico.modules.users.models.users), 215
- synced_fields (indico.modules.users.models.users.User attribute), 214
- synced_values (indico.modules.users.models.users.User attribute), 214
- synchronize_data() (indico.modules.users.models.users.User method), 214
- system_app_type (indico.modules.oauth.models.applications.OAuthApplication attribute), 244
- SystemAppType (class in indico.modules.oauth.models.applications), 244
- ## T
- TargetAbstractIDPlaceholder (class in indico.modules.events.abstracts.placeholders), 115
- TargetAbstractTitlePlaceholder (class in indico.modules.events.abstracts.placeholders), 115
- TargetSubmitterFirstNamePlaceholder (class in indico.modules.events.abstracts.placeholders), 115
- TargetSubmitterLastNamePlaceholder (class in indico.modules.events.abstracts.placeholders), 115
- TargetSubmitterNamePlaceholder (class in indico.modules.events.abstracts.placeholders), 115
- telephone (indico.modules.rb.models.rooms.Room attribute), 230
- TEMPATE (indico.modules.attachments.preview.Previewer attribute), 226
- TEMPLATE (indico.modules.attachments.preview.ImagePreviewer attribute), 226
- TEMPLATE (indico.modules.attachments.preview.PDFPreviewer attribute), 226
- template (indico.modules.designer.models.images.DesignerImageImage attribute), 253
- template_hook (in module indico.core.signals.plugin), 49
- template_hook() (indico.core.plugins.IndicoPlugin method), 41
- template_id (indico.modules.designer.models.images.DesignerImageImage attribute), 253
- template_kwargs (indico.modules.events.logs.renderers.EventLogRendererEventLog attribute), 137
- template_kwargs (indico.modules.events.logs.renderers.SimpleRendererSimpleRenderer attribute), 137
- template_name (indico.modules.events.logs.renderers.EmailRendererEmailRenderer attribute), 137
- template_name (indico.modules.events.logs.renderers.EventLogRendererEventLog attribute), 137
- template_name (indico.modules.events.logs.renderers.SimpleRendererSimpleRenderer attribute), 137
- TEMPLATES_DIR (indico.modules.attachments.preview.Previewer attribute), 226
- text (indico.modules.events.abstracts.models.review_questions.AbstractReviewQuestions attribute), 108
- text (indico.modules.events.papers.models.review_questions.PaperReviewQuestions attribute), 108
- text (indico.modules.events.registration.models.items.RegistrationFormItemRegistrationFormItem attribute), 170
- text (indico.modules.events.surveys.models.items.SurveyItemTypeSurveyItemType attribute), 191
- text_color (indico.modules.events.sessions.models.sessions.SessionSession attribute), 184
- text_color (indico.modules.events.timetable.models.breaks.BreakBreak attribute), 196
- TextListField (class in indico.web.forms.fields), 269
- TextPreviewer (class in indico.modules.attachments.preview), 226
- theme (indico.modules.events.models.events.EventEvent attribute), 84
- themes (indico.modules.events.settings.ThemeSettingsProxyThemeSettingsProxy attribute), 99, 154
- ThemeSettingsProxy (class in indico.modules.events.settings), 99, 154
- thumbnail (indico.modules.rb.models.photos.PhotoPhoto attribute), 235
- ticket_on_email (indico.modules.events.registration.models.forms.RegistrationFormRegistrationForm attribute), 167
- ticket_on_event_page (indico.modules.events.registration.models.forms.RegistrationFormRegistrationForm attribute), 167
- ticket_on_summary_page (indico.modules.events.registration.models.forms.RegistrationFormRegistrationForm attribute), 167
- ticket_template (indico.modules.events.registration.models.forms.RegistrationFormRegistrationForm attribute), 167
- ticket_template_id (indico.modules.events.registration.models.forms.RegistrationFormRegistrationForm attribute), 167
- ticket_uuid (indico.modules.events.registration.models.registrations.RegistrationRegistration attribute), 161
- tickets_enabled (indico.modules.events.registration.models.forms.RegistrationFormRegistrationForm attribute), 167
- time (indico.modules.events.timetable.reschedule.RescheduleModeRescheduleMode attribute), 200
- TimeDeltaField (class in indico.web.forms.fields), 272
- time_type (indico.modules.events.models.reviews.ProposalCommentMixinProposalCommentMixin attribute), 89

[timeline_item_type](#) (in module `indico.modules.events.models.reviews.ProposalReviewMixin` attribute), 91
[timeline_item_type](#) (in module `indico.modules.events.papers.models.reviews.PaperReview` attribute), 146
[TIMELINE_TYPE](#) (in module `indico.modules.events.papers.models.reviews.PaperReview` attribute), 146
[times_changed](#) (in module `indico.core.signals.event`), 47
[timestamp](#) (in module `indico.modules.events.agreements.models.agreements.Agreement` attribute), 118
[timestamp](#) (in module `indico.modules.events.payment.models.transactions.PaymentTransaction` attribute), 155
[timestamp](#) (in module `indico.modules.rb.models.reservation_edit_logs.ReservationEditLog` attribute), 238
[timetable_buttons](#) (in module `indico.core.signals.event`), 47
[timetable_entry](#) (in module `indico.modules.events.contributions.models.subcontributions.SubContribution` attribute), 127
[timetable_entry_created](#) (in module `indico.core.signals.event`), 47
[timetable_entry_deleted](#) (in module `indico.core.signals.event`), 47
[timetable_entry_updated](#) (in module `indico.core.signals.event`), 47
[TimetableEntry](#) (class in module `indico.modules.events.timetable.models.entries`), 196
[TimetableEntryType](#) (class in module `indico.modules.events.timetable.models.entries`), 198
[timezone](#) (in module `indico.modules.categories.models.categories.Categories` attribute), 206
[timezone](#) (in module `indico.modules.events.models.events.Event` attribute), 84
[timezone](#) (in module `indico.web.forms.fields.IndicoDateTimeField` attribute), 272
[timezone](#) (in module `indico.web.forms.fields.OccurrencesField` attribute), 272
[timezone_field](#) (in module `indico.web.forms.fields.IndicoDateTimeField` attribute), 272
[timezone_field](#) (in module `indico.web.forms.fields.OccurrencesField` attribute), 272
[title](#) (in module `indico.modules.attachments.models.attachments.Attachments` attribute), 221
[title](#) (in module `indico.modules.attachments.models.folders.AttachmentsFolder` attribute), 224
[title](#) (in module `indico.modules.categories.models.categories.Categories` attribute), 206
[title](#) (in module `indico.modules.designer.models.templates.DesignerTemplate` attribute), 254
[title](#) (in module `indico.modules.events.abstracts.models.abstracts.AbstractEvent` attribute), 102
[title](#) (in module `indico.modules.events.abstracts.models.email_templates.AbstractEmailTemplate` attribute), 106
[title](#) (in module `indico.modules.events.contributions.models.contributions.Contribution` attribute), 122
[title](#) (in module `indico.modules.events.contributions.models.fields.ContributionField` attribute), 123
[title](#) (in module `indico.modules.events.contributions.models.subcontributions.SubContribution` attribute), 127
[title](#) (in module `indico.modules.events.layout.models.menu.MenuEntry` attribute), 132
[title](#) (in module `indico.modules.events.models.events.Event` attribute), 84
[title](#) (in module `indico.modules.events.models.persons.PersonLinkBase` attribute), 87
[title](#) (in module `indico.modules.events.models.reviews.ProposalGroupProxy` attribute), 90
[title](#) (in module `indico.modules.events.papers.models.papers.Paper` attribute), 145
[title](#) (in module `indico.modules.events.registration.models.form_fields.RegistrationFormField` attribute), 163
[title](#) (in module `indico.modules.events.registration.models.form_fields.RegistrationFormText` attribute), 164
[title](#) (in module `indico.modules.events.registration.models.forms.RegistrationForm` attribute), 167
[title](#) (in module `indico.modules.events.registration.models.items.PersonalDataType` attribute), 168
[title](#) (in module `indico.modules.events.registration.models.items.RegistrationFormItem` attribute), 169
[title](#) (in module `indico.modules.events.registration.models.items.RegistrationFormPerson` attribute), 170
[title](#) (in module `indico.modules.events.registration.models.items.RegistrationFormSection` attribute), 171
[title](#) (in module `indico.modules.events.registration.models.items.RegistrationFormText` attribute), 172
[title](#) (in module `indico.modules.events.requests.base.RequestDefinitionBase` attribute), 182
[title](#) (in module `indico.modules.events.sessions.models.blocks.SessionBlock` attribute), 185
[title](#) (in module `indico.modules.events.sessions.models.sessions.Session` attribute), 184
[title](#) (in module `indico.modules.events.surveys.models.items.SurveyItem` attribute), 190
[title](#) (in module `indico.modules.events.surveys.models.items.SurveyQuestion` attribute), 191
[title](#) (in module `indico.modules.events.surveys.models.items.SurveySection` attribute), 192
[title](#) (in module `indico.modules.events.surveys.models.items.SurveyText` attribute), 192
[title](#) (in module `indico.modules.events.surveys.models.surveys.Survey` attribute), 190
[title](#) (in module `indico.modules.events.timetable.models.breaks.Break` attribute), 196
[title](#) (in module `indico.modules.events.timetable.reschedule.RescheduleMode` attribute), 200

title (indico.modules.events.tracks.models.tracks.Track attribute), 202
 title (indico.modules.news.models.news.NewsItem attribute), 261
 title (indico.modules.rb.models.room_attributes.RoomAttribute attribute), 230
 title (indico.modules.users.models.users.PersonMixin attribute), 212
 title_attr (indico.modules.events.models.reviews.ProposalGroupProxy attribute), 90
 to_be_corrected (indico.modules.events.papers.models.reviews.PaperAttribute attribute), 146
 to_be_corrected (indico.modules.events.papers.models.revisions.PaperRevisionState attribute), 149
 to_dict() (indico.modules.events.surveys.models.items.SurveyItem attribute), 191
 to_dict() (indico.modules.events.surveys.models.items.SurveyQuestion attribute), 191
 to_dict() (indico.modules.events.surveys.models.items.SurveySection attribute), 192
 to_dict() (indico.modules.events.surveys.models.items.SurveyText attribute), 192
 top_left_latitude (indico.modules.rb.models.aspects.Aspect attribute), 231
 top_left_longitude (indico.modules.rb.models.aspects.Aspect attribute), 231
 TplData (class in indico.modules.designer.pdf), 254
 Track (class in indico.modules.events.tracks.models.tracks), type (indico.modules.events.logs.models.entries.EventLogEntry attribute), 201
 track (indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 110
 track (indico.modules.events.contributions.models.contributions.Contribution attribute), 122
 track_id (indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 110
 track_id (indico.modules.events.contributions.models.contributions.Contribution attribute), 122
 track_time_changes() (in module indico.modules.events.util), 97
 TrackRoleField (class in indico.modules.events.abstracts.fields), 264
 transaction (indico.modules.events.registration.models.registration.Registration attribute), 161
 transaction_id (indico.modules.events.registration.models.registration.Registration attribute), 161
 TransactionAction (class in indico.modules.events.payment.models.transactions), 155
 TransactionStatus (class in indico.modules.events.payment.models.transactions), 155
 TransactionStatusTransition (class in indico.modules.events.payment.models.transactions), 155
 TransientMenuEntry (class in indico.modules.events.layout.models.menu), 133
 translation_domain (indico.core.plugins.IndicoPlugin attribute), 41
 translation_path (indico.core.plugins.IndicoPlugin attribute), 41
 ttl (indico.modules.oauth.models.tokens.OAuthGrant attribute), 245
 type (indico.modules.attachments.models.attachments.Attachment attribute), 221
 type (indico.modules.attachments.models.principals.AttachmentFolderPrincipal attribute), 225
 type (indico.modules.attachments.models.principals.AttachmentPrincipal attribute), 207
 type (indico.modules.categories.models.principals.CategoryPrincipal attribute), 254
 type (indico.modules.designer.models.templates.DesignerTemplate attribute), 118
 type (indico.modules.events.agreements.models.agreements.Agreement attribute), 122
 type (indico.modules.events.contributions.models.contributions.Contribution attribute), 125
 type (indico.modules.events.layout.models.menu.MenuEntry attribute), 132
 type (indico.modules.events.logs.models.entries.EventLogEntry attribute), 136
 type (indico.modules.events.models.events.Event attribute), 84
 type (indico.modules.events.models.principals.EventPrincipal attribute), 88
 type (indico.modules.events.models.settings.EventSettingPrincipal attribute), 93
 type (indico.modules.events.models.static_list_links.StaticListLink attribute), 94
 type (indico.modules.events.papers.models.review_questions.PaperReviewQuestion attribute), 145
 type (indico.modules.events.papers.models.reviews.PaperReview attribute), 147
 type (indico.modules.events.registration.models.form_fields.RegistrationForm attribute), 163
 type (indico.modules.events.registration.models.form_fields.RegistrationForm attribute), 164
 type (indico.modules.events.registration.models.items.RegistrationFormItem attribute), 169
 type (indico.modules.events.registration.models.items.RegistrationFormPermission attribute), 170
 type (indico.modules.events.registration.models.items.RegistrationFormSection attribute), 171
 type (indico.modules.events.registration.models.items.RegistrationFormText attribute), 172
 type (indico.modules.events.requests.models.requests.Request attribute), 172

attribute), 180
 type (indico.modules.events.sessions.models.principals.SessionPrincipal attribute), 186
 type (indico.modules.events.surveys.models.items.SurveyItem attribute), 191
 type (indico.modules.events.surveys.models.items.SurveyQuestion attribute), 191
 type (indico.modules.events.surveys.models.items.SurveySection attribute), 192
 type (indico.modules.events.surveys.models.items.SurveyText attribute), 192
 type (indico.modules.events.timetable.models.entries.TimetableEntry attribute), 198
 type (indico.modules.oauth.models.tokens.OAuthToken attribute), 245
 type (indico.modules.rb.models.blocking_principals.BlockingPrincipal attribute), 233
 type (indico.modules.rb.models.room_attributes.RoomAttribute attribute), 230
 type (indico.modules.vc.models.vc_rooms.VCRoom attribute), 248
 type_ (indico.modules.events.models.events.Event attribute), 84
 type_changed (in module indico.core.signals.event), 47
 type_id (indico.modules.events.contributions.models.contributions.models.contributions.Registration attribute), 122
 tzinfo (indico.modules.categories.models.categories.Category attribute), 206
 tzinfo (indico.modules.events.models.events.Event attribute), 84

U

under_review (indico.modules.events.abstracts.models.abstracts.models.abstracts.PublicState attribute), 102
 undo_impersonate_user() (in module indico.modules.auth.util), 243
 UNICODE_PARAMS (in- indico.modules.rb.services.aspects.RoomBookingMapBase attribute), 241
 UNICODE_PARAMS (in- indico.modules.rb.services.blockings.RoomBookingBlockingProcessBase attribute), 241
 UNICODE_PARAMS (in- indico.modules.rb.services.rooms.BookingPermission attribute), 240
 UNICODE_PARAMS (in- indico.modules.rb.services.rooms.RoomBookingAvailabilitySearchRooms attribute), 240
 UNICODE_PARAMS (in- indico.modules.rb.services.rooms.RoomBookingListLocationsAndRoomsWithGuids attribute), 240
 unimplemented() (in module indico.modules.rb.models.util), 239
 unique_columns (indico.modules.attachments.models.principals.Attachments attribute), 224
 unique_columns (indico.modules.attachments.models.principals.Attachments attribute), 225
 unique_columns (indico.modules.categories.models.principals.CategoryPrincipals attribute), 208
 unique_columns (indico.modules.events.contributions.models.principals.Contributions attribute), 125
 unique_columns (indico.modules.events.models.principals.EventPrincipals attribute), 88
 unique_columns (indico.modules.events.sessions.models.principals.SessionPrincipals attribute), 186
 unique_columns (indico.modules.rb.models.blocking_principals.BlockingPrincipals attribute), 233
 unique_links (indico.modules.attachments.models.folders.AttachmentFolders attribute), 224
 unique_links (indico.modules.events.notes.models.notes.EventNotes attribute), 140
 unit_names (indico.web.forms.fields.RelativeDeltaField attribute), 276
 unit_names (indico.web.forms.fields.TimeDeltaField attribute), 272
 unlock_event() (in module indico.modules.events.operations), 94
 update_abstract() (in module indico.modules.events.abstracts.operations), 111
 update_abstract_comment() (in module indico.modules.events.abstracts.operations), 111
 update_abstract_review() (in module indico.modules.events.abstracts.operations), 111
 update_break_entry() (in module indico.modules.events.timetable.operations), 199
 update_category() (in module indico.modules.categories.operations), 208
 update_comment() (in module indico.modules.events.papers.operations), 151
 update_competences() (in module indico.modules.events.papers.operations), 151
 update_contribution() (in module indico.modules.events.contributions.operations), 128
 update_data_association() (in- indico.modules.vc.plugins.VCPluginMixin method), 250
 update_data_vc_room() (in- indico.modules.vc.plugins.VCPluginMixin method), 252
 update_event() (in module in-

- dico.modules.events.operations), 94
- update_event_protection() (in module dico.modules.events.operations), 94
- update_event_type() (in module dico.modules.events.operations), 94
- update_name() (indico.modules.rb.models.rooms.Room method), 230
- update_object_principals() (in module dico.modules.events.util), 97
- update_paper_template() (in module dico.modules.events.papers.operations), 151
- update_person() (in module dico.modules.events.persons.operations), 157
- update_program() (in module dico.modules.events.tracks.operations), 202
- update_reference_type() (in module dico.modules.events.operations), 94
- update_review() (in module dico.modules.events.papers.operations), 151
- update_reviewed_for_tracks() (in module dico.modules.events.abstracts.operations), 111
- update_reviewing_roles() (in module dico.modules.events.papers.operations), 151
- update_session() (in module dico.modules.events.sessions.operations), 187
- update_session_block() (in module dico.modules.events.sessions.operations), 187
- update_session_coordinator_privs() (in module dico.modules.events.sessions.operations), 187
- update_state() (indico.modules.events.registration.models.registration.models.abstracts.operations), 161
- update_subcontribution() (in module dico.modules.events.contributions.operations), 129
- update_team_members() (in module dico.modules.events.papers.operations), 151
- update_timetable_entry() (in module dico.modules.events.timetable.operations), 199
- update_timetable_entry_object() (in module dico.modules.events.timetable.operations), 199
- update_track() (in module dico.modules.events.tracks.operations), 202
- updated (in module indico.core.signals.category), 45
- updated (in module indico.core.signals.event), 47
- url (indico.modules.categories.models.categories.Category attribute), 207
- url (indico.modules.events.layout.models.menu.MenuEntryMixin attribute), 133
- url (indico.modules.events.models.events.Event attribute), 84
- url (indico.modules.events.models.references.ReferenceModelBase attribute), 88
- url_for_login() (in module indico.modules.auth.util), 243
- url_for_logout() (in module indico.modules.auth.util), 243
- url_for_plugin() (in module indico.core.plugins), 42
- url_for_register() (in module indico.modules.auth.util), 243
- url_rule_to_angular() (in module indico.modules.events.registration.util), 174
- url_shortcut (indico.modules.events.models.events.Event attribute), 84
- url_template (indico.modules.events.models.references.ReferenceType attribute), 89
- url_to_static_filename() (in module indico.modules.events.static.util), 203
- urn (indico.modules.events.models.references.ReferenceModelBase attribute), 88
- used_equipment (indico.modules.rb.models.reservations.Reservation attribute), 238
- User (class in indico.modules.users.models.users), 212
- user (indico.modules.attachments.models.attachments.Attachment attribute), 221
- user (indico.modules.attachments.models.attachments.AttachmentFile attribute), 221
- user (indico.modules.attachments.models.principals.AttachmentFolderPrincipal attribute), 224
- user (indico.modules.attachments.models.principals.AttachmentPrincipal attribute), 225
- user (indico.modules.categories.models.principals.CategoryPrincipal attribute), 208
- user (indico.modules.events.registration.models.registration.models.abstracts.models.comments.AbstractComment attribute), 104
- user (indico.modules.events.abstracts.models.email_logs.AbstractEmailLog attribute), 105
- user (indico.modules.events.abstracts.models.reviews.AbstractReview attribute), 110
- user (indico.modules.events.agreements.models.agreements.Agreement attribute), 118
- user (indico.modules.events.contributions.models.principals.ContributionPrincipal attribute), 125
- user (indico.modules.events.logs.models.entries.EventLogEntry attribute), 136
- user (indico.modules.events.models.persons.EventPerson attribute), 86
- user (indico.modules.events.models.principals.EventPrincipal attribute), 88
- user (indico.modules.events.models.settings.EventSettingPrincipal attribute), 93
- user (indico.modules.events.notes.models.notes.EventNoteRevision attribute), 140

user (indico.modules.events.papers.models.comments.PaperReviewComment (in-
 attribute), 142 indico.modules.events.models.persons.EventPerson
 attribute), 86
 user (indico.modules.events.papers.models.competences.PaperCompetence (in-
 attribute), 143 indico.modules.events.models.principals.EventPrincipal
 attribute), 88
 user (indico.modules.events.papers.models.reviews.PaperReview (in-
 attribute), 147 indico.modules.events.models.settings.EventSettingPrincipal
 attribute), 93
 user (indico.modules.events.registration.models.registrations.Registration (in-
 attribute), 161 indico.modules.events.notes.models.notes.EventNoteRevision
 attribute), 140
 user (indico.modules.events.sessions.models.principals.SessionPrincipal (in-
 attribute), 186 indico.modules.events.papers.models.comments.PaperReviewCom
 attribute), 142
 user (indico.modules.events.surveys.models.submissions.SurveySubmission (in-
 attribute), 193 indico.modules.events.papers.models.competences.PaperCompeten
 attribute), 143
 user (indico.modules.oauth.models.tokens.OAuthToken (in-
 attribute), 245 user_id (indico.modules.events.papers.models.reviews.PaperReview
 attribute), 147
 user (indico.modules.rb.models.blocking_principals.BlockingPrincipal (in-
 attribute), 233 indico.modules.events.registration.models.registrations.Registration
 attribute), 161
 user (indico.modules.users.models.settings.UserSetting (in-
 attribute), 216 user_id (indico.modules.events.sessions.models.principals.SessionPrincipal
 attribute), 186
 user_backref_name (in-
 attribute), 193 user_id (indico.modules.events.surveys.models.submissions.SurveySubmiss
 attribute), 193
 user_backref_name (in-
 attribute), 104 user_id (indico.modules.oauth.models.tokens.OAuthToken
 attribute), 245
 user_backref_name (in-
 attribute), 142 indico.modules.rb.models.blocking_principals.BlockingPrincipal
 attribute), 233
 user_competences (indico.modules.events.papers.models.calls_for_responses.CallForResponses (in-
 attribute), 141 user_id (indico.modules.events.papers.models.affiliations.UserAffiliation
 attribute), 215
 user_data (indico.modules.auth.models.registration_requests.RegistrationRequest (in-
 attribute), 243 user_id (indico.modules.users.models.emails.UserEmail
 attribute), 215
 user_data (indico.modules.events.registration.models.registrations.RegistrationData (in-
 attribute), 162 user_id (indico.modules.users.models.settings.UserSetting
 attribute), 216
 user_id (indico.modules.attachments.models.attachments.Attachment (in-
 attribute), 221 user_id (indico.modules.users.models.suggestions.SuggestedCategory
 attribute), 216
 user_id (indico.modules.attachments.models.attachments.AttachmentFile (in-
 attribute), 221 user_id (indico.modules.events.layout.models.menu.MenuEntryType
 attribute), 133
 user_id (indico.modules.attachments.models.principals.AttachmentFilePrincipal (in-
 attribute), 224 user_backref_name (in-
 attribute), 105 indico.modules.events.abstracts.models.comments.AbstractComme
 attribute), 105
 user_id (indico.modules.attachments.models.principals.AttachmentPrincipal (in-
 attribute), 225 user_modified_backref_name (in-
 attribute), 142
 user_id (indico.modules.auth.models.identities.Identity (in-
 attribute), 242 indico.modules.events.papers.models.comments.PaperReviewCom
 attribute), 142
 user_id (indico.modules.categories.models.principals.CategoryPrincipal (in-
 attribute), 208 user_id (indico.modules.rb.models.reservation_edit_logs.ReservationEd
 attribute), 239
 user_id (indico.modules.events.abstracts.models.comments.AbstractComment (in-
 attribute), 105 module (in-
 attribute), 217 indico.modules.users.models.settings), 217
 user_id (indico.modules.events.abstracts.models.email_logs.AbstractEmailLogEntry (in-
 attribute), 105 method), 102
 user_id (indico.modules.events.abstracts.models.reviews.AbstractReview (in-
 attribute), 110 rooms() (in-
 attribute), 230 indico.modules.rb.models.rooms.Room class
 attribute), 230
 user_id (indico.modules.events.agreements.models.agreements.Agreement (in-
 attribute), 118 user_settings (indico.core.plugins.IndicoPlugin attribute),
 attribute), 126
 user_id (indico.modules.events.contributions.models.principals.ContributionPrincipal (in-
 attribute), 126 user_settings_converters (in-
 attribute), 41 indico.core.plugins.IndicoPlugin attribute),
 attribute), 136

- UserAffiliation (class in indico.modules.users.models.affiliations), 215
- UserEmail (class in indico.modules.users.models.emails), 215
- users (indico.modules.events.abstracts.fields.TrackRoleField attribute), 265
- users (indico.modules.events.abstracts.models.reviews.AbstractCommentVisibility attribute), 109
- users (indico.modules.events.papers.models.reviews.PaperCommentVisibility attribute), 146
- UserSetting (class in indico.modules.users.models.settings), 216
- UserSettingsProxy (class in indico.modules.users.models.settings), 216
- UserTitle (class in indico.modules.users.models.users), 214
- uses_vc (indico.modules.rb.models.reservations.Reservation attribute), 238
- uuid (indico.modules.events.agreements.models.agreements.Agreement attribute), 118
- uuid (indico.modules.events.models.static_list_links.StaticListLink attribute), 94
- uuid (indico.modules.events.registration.models.invitations.RegistrationInvitation attribute), 168
- uuid (indico.modules.events.registration.models.registrations.Registration attribute), 161
- uuid (indico.modules.events.surveys.models.surveys.Survey attribute), 190
- V**
- valid_currencies (indico.modules.events.payment.plugins.PaymentPluginMixin attribute), 157
- validate_redirect_uri() (indico.modules.oauth.models.applications.OAuthApplication method), 244
- value (indico.modules.categories.models.settings.CategorySetting attribute), 208
- value (indico.modules.events.abstracts.models.review_ratings.AbstractReviewRating attribute), 109
- value (indico.modules.events.contributions.models.references.ContributionReference attribute), 126
- value (indico.modules.events.contributions.models.references.SubContributionReference attribute), 126
- value (indico.modules.events.models.references.EventReference attribute), 88
- value (indico.modules.events.models.references.ReferenceModelAttribute), 88
- value (indico.modules.events.models.settings.EventSetting attribute), 92
- value (indico.modules.events.papers.models.review_ratings.PaperReviewRating attribute), 146
- value (indico.modules.rb.models.room_attributes.RoomAttributeAssociation attribute), 231
- value (indico.modules.users.models.settings.UserSetting attribute), 216
- vc_room (indico.modules.vc.models.vc_rooms.VCRoomEventAssociation attribute), 250
- vc_room_attach_form (indico.modules.vc.plugins.VCPluginMixin attribute), 252
- vc_room_form (indico.modules.vc.plugins.VCPluginMixin attribute), 252
- vc_room_id (indico.modules.vc.models.vc_rooms.VCRoomEventAssociation attribute), 250
- VCPluginMixin (class in indico.modules.vc.plugins), 250
- VCRoom (class in indico.modules.vc.models.vc_rooms), 248
- VCRoomError, 252
- VCRoomEventAssociation (class in indico.modules.vc.models.vc_rooms), 248
- VCRoomLinkType (class in indico.modules.vc.models.vc_rooms), 250
- VCRoomNotFoundError, 252
- VCRoomStatus (class in indico.modules.vc.models.vc_rooms), 250
- RegistrationTitle (indico.modules.events.abstracts.models.abstracts.Abstract attribute), 102
- RegistrationTitle (indico.modules.events.contributions.models.contributions.Contribution attribute), 122
- verbose_title (indico.modules.events.papers.models.papers.Paper attribute), 145
- version_of (indico.modules.attachments.models.attachments.AttachmentFile attribute), 221
- version_of (indico.modules.designer.models.images.DesignerImageFile attribute), 253
- version_of (indico.modules.events.layout.models.images.ImageFile attribute), 131
- versioned_data (indico.modules.events.registration.models.form_fields.RegistrationForm attribute), 163
- versioned_data (indico.modules.events.registration.models.form_fields.RegistrationForm attribute), 163
- view_data (indico.modules.events.registration.models.form_fields.RegistrationForm attribute), 163
- view_data (indico.modules.events.registration.models.form_fields.RegistrationForm attribute), 164
- view_data (indico.modules.events.registration.models.items.RegistrationForm attribute), 170
- view_data (indico.modules.events.registration.models.items.RegistrationForm attribute), 170
- view_data (indico.modules.events.registration.models.items.RegistrationForm attribute), 171
- view_data (indico.modules.events.registration.models.items.RegistrationForm attribute), 172
- visibility (indico.modules.categories.models.categories.Category attribute), 207
- visibility (indico.modules.events.abstracts.models.comments.AbstractComment attribute), 105

visibility (indico.modules.events.abstracts.models.reviews.AbstractReviewField attribute), 110

visibility (indico.modules.events.models.events.Event attribute), 84

visibility (indico.modules.events.papers.models.comments.PaperReviewField attribute), 142

visibility (indico.modules.events.papers.models.reviews.PaperReviewField attribute), 147

visibility_horizon_query (in-dico.modules.categories.models.categories.Category attribute), 207

visible() (indico.modules.events.layout.util.MenuEntryData widget method), 134

visible_categories_cte (in-dico.modules.categories.models.categories.Category attribute), 207

visible_categories_query (in-dico.modules.categories.models.categories.Category attribute), 207

W

warning (indico.modules.categories.models.categories.EventMessageField attribute), 207

was_survey_submitted() (in module indico.modules.events.surveys.util), 195

WEEK (indico.modules.rb.models.reservations.RepeatFrequency attribute), 236

week_day_data (indico.web.forms.fields.IndicoWeekDayRepetitionField attribute), 277

WEEK_DAY_NUMBER_CHOICES (in-dico.web.forms.fields.IndicoWeekDayRepetitionField attribute), 277

widget (indico.modules.categories.fields.CategoryField attribute), 267

widget (indico.modules.events.abstracts.fields.AbstractField attribute), 262

widget (indico.modules.events.abstracts.fields.AbstractPersonLinkListField attribute), 262

widget (indico.modules.events.abstracts.fields.EmailRuleListField attribute), 264

widget (indico.modules.events.abstracts.fields.TrackRoleField attribute), 265

widget (indico.modules.events.contributions.fields.ContributionPersonLinkListField attribute), 265

widget (indico.modules.events.contributions.fields.SubContributionPersonLinkListField attribute), 265

widget (indico.modules.events.fields.EventPersonLinkListField attribute), 261

widget (indico.modules.events.fields.PersonLinkListFieldBase attribute), 262

widget (indico.modules.events.papers.fields.PaperEmailSettingsField attribute), 266

widget (indico.modules.events.sessions.fields.SessionBlockPersonLinkListField attribute), 266

widget (indico.web.forms.fields.AccessControlListField attribute), 275

widget (indico.web.forms.fields.FileField attribute), 273

widget (indico.web.forms.fields.HiddenFieldList attribute), 269

widget (indico.web.forms.fields.IndicoDateField attribute), 276

widget (indico.web.forms.fields.IndicoDateTimeField attribute), 272

widget (indico.web.forms.fields.IndicoEmailRecipientsField attribute), 278

widget (indico.web.forms.fields.IndicoEnumRadioField attribute), 273

widget (indico.web.forms.fields.IndicoEnumSelectField attribute), 273

widget (indico.web.forms.fields.IndicoLocationField attribute), 275

widget (indico.web.forms.fields.IndicoMarkdownField attribute), 276

widget (indico.web.forms.fields.IndicoPalettePickerField attribute), 272

widget (indico.web.forms.fields.IndicoPasswordField attribute), 270

widget (indico.web.forms.fields.IndicoProtectionField attribute), 276

widget (indico.web.forms.fields.IndicoRadioField attribute), 267

widget (indico.web.forms.fields.IndicoSelectMultipleCheckboxField attribute), 267

widget (indico.web.forms.fields.IndicoStaticTextField attribute), 271

widget (indico.web.forms.fields.IndicoTagListField attribute), 271

widget (indico.web.forms.fields.IndicoWeekDayRepetitionField attribute), 277

widget (indico.web.forms.fields.MultipleItemsField attribute), 274

widget (indico.web.forms.fields.MultiStringField attribute), 274

widget (indico.web.forms.fields.OccurrencesField attribute), 272

widget (indico.web.forms.fields.OverrideMultipleItemsField attribute), 275

widget (indico.web.forms.fields.PrincipalField attribute), 275

widget (indico.web.forms.fields.PrincipalListField attribute), 275

widget (indico.web.forms.fields.RelativeDeltaField attribute), 276

widget (indico.web.forms.fields.TimeDeltaField attribute), 272

width (indico.modules.designer.pdf.TplData attribute), 264

width_cm (indico.modules.designer.pdf.TplData attribute), 264

tribute), 255

with_title (indico.modules.designer.placeholders.RegistrationFullNameNoTitlePlaceholder attribute), 255

with_title (indico.modules.designer.placeholders.RegistrationFullNameNoTitlePlaceholderB attribute), 256

with_title (indico.modules.designer.placeholders.RegistrationFullNameNoTitlePlaceholderC attribute), 256

with_title (indico.modules.designer.placeholders.RegistrationFullNameNoTitlePlaceholderD attribute), 256

with_title (indico.modules.designer.placeholders.RegistrationFullNamePlaceholder attribute), 255

with_title (indico.modules.designer.placeholders.RegistrationFullNamePlaceholderB attribute), 256

with_title (indico.modules.designer.placeholders.RegistrationFullNamePlaceholderC attribute), 256

with_title (indico.modules.designer.placeholders.RegistrationFullNamePlaceholderD attribute), 256

withdraw() (indico.modules.events.requests.base.RequestDefinitionBase class method), 182

withdraw_abstract() (in module indico.modules.events.abstracts.operations), 111

withdrawn (indico.modules.events.abstracts.models.abstracts.AbstractPublicState attribute), 102

withdrawn (indico.modules.events.abstracts.models.abstracts.AbstractState attribute), 103

withdrawn (indico.modules.events.registration.models.registrations.RegistrationState attribute), 162

withdrawn (indico.modules.events.requests.models.requests.RequestState attribute), 180

working_time_end (indico.modules.rb.models.locations.Location attribute), 235

working_time_periods (in indico.modules.rb.models.locations.Location attribute), 235

working_time_start (in indico.modules.rb.models.locations.Location attribute), 235

WPEventManagement (class in indico.modules.events.management.views), 137

Z

ZipGeneratorMixin (class in indico.modules.events.util), 95

zoom_level (indico.modules.rb.models.aspects.Aspect attribute), 232