OpenACalendar Documentation

Release

JMB Technology Ltd

Nov 09, 2017

Contents

Welcome.

Contents:

CHAPTER 1

Single Site or Multi Site Mode

1.1 Single Site Mode

Single Site means that the installed code base only runs one calendar. See Open Tech Calendar for an example. http://opentechcalendar.co.uk This is the simpler mode to install.

In Single Site Mode, one calendar is set up then simply used as the calendar for all requests. Thus it is possible to turn a site from Single Site Mode into Multi Site Mode.

1.2 Multi Site Mode

Multi Site means that the installed code base can run more than one calendar. See HasACalendar for an example. http://ican.hasacalendar.co.uk This is the more complex mode to install.

You need to be set up several web roots with several different virtual hosts. One of them will need a domain name that allows subdomains configured (eg *.example.com).

CHAPTER 2

System Administrator Web UI

2.1 Accessing in Single Site Mode

Go to "/sysadmin".

Note some calendar functions are accessed by going to "/admin". Here you can set other users up to be a calendar admin. This is different from being a real systems administrator.

2.2 Accessing in Multi Site Mode

Go to the site domain configured for the "webIndex" folder.

Go to "/sysadmin".

2.3 Security

Users have to have the sysadmin flag set to access the sysadmin interface. The first user you create during installation will be a sysadmin. This user can make other users a sysadmin using the sysadmin interface.

On accesing the interface, for extra security the user has to enter passwords again. As well as the users normal password, an additional password is required to access the Sysadmin UI. It is set in config.php in the sysAdminExtraPassword option.

```
$CONFIG->sysAdminExtraPassword = 'qwerty';
```

2.4 Finding Slug

At times in the system administrators UI you will be asked to enter the slug of a piece of data. This can be found by looking at the URL.

The Slug is the ID number of the data, up to the first hyphen only.

For example,

- the slug of http://hasadevcalendar.co.uk:20153/event/28-test/ is 28
- the slug of http://hasadevcalendar.co.uk:20153/group/4-3-good-people/ is 4
- the slug of http://hasadevcalendar.co.uk:20153/venue/3-two-towers/ is 3

CHAPTER 3

Duplicate Data

Sometimes the same venue, group or event is entered into the system twice.

The system administrator has a feature to mark one as a duplicate of the other. This will delete the duplicate and move any attached data to the original.

For instance, if you mark one venue as a duplicate of another, any events at that venue will now be marked as being at the original venue.

3.1 How to mark?

Browse the System Administors user interface.

Find the bit of data you want to mark as a duplicate and enter the command “isduplicateof slug”, where slug is the slug of the original bit of data.

CHAPTER 4

Purge Data

4.1 What is purge?

The system administrator can "purge" data from the system.

Deleting data marks it as deleted but still makes it available. Purge attemps to remove the data from the system completely.

Whilst deleting data is undoable, purging data is NOT undoable in any way.

4.2 When to purge?

It is recommended to purge data only as a last resort and not as a routine operation because:

- it is NOT undoable in any way.
- it may make the history of your site a bit confused. For example, you may be left with edits that appear to have changed nothing.

4.3 How to purge?

Browse the System Administrators user interface.

Find the bit of data you want to purge and enter the command "purge password".

The password is set by your server administrator in config.php. There is no default value and the feature will not work if a password has not been set. Note different passwords can be set for different types of data for extra security.

It is recommended this password is different from any other passwords - an attacker who hacks into your server administrator UI (maybe by sniffing passwords) is bad enought, an attacker who hacks into your server administrator UI and also knows the password to purge data is really bad.

```
$CONFIG->sysAdminExtraPurgeEventPassword = 'pick';
$CONFIG->sysAdminExtraPurgeGroupPassword = 'better';
$CONFIG->sysAdminExtraPurgeVenuePassword = 'passwords';
$CONFIG->sysAdminExtraPurgeAreaPassword = 'than';
$CONFIG->sysAdminExtraPurgeCuratedListPassword = 'this';
```

4.4 How to undo a purge?

Go and fetch a backup of your database and files and restore them. There is no other way.

CHAPTER 5

Requirements

- PHP 7.0 or higher
- PHP Extension - CURL for importing events
- PHP Extension - GD for image manipulation
- Postgresql
- Apache with ModRewrite (or equivalent)
- Command line access
- Cron

The requirement for PHP 7.0 or higher was introduced in version 2.0 of this app. If this is a problem for you, please check version 1.7, which supports PHP 5.4, 5.5 and 5.6.

It has currently been tested on Linux webservers only, but it may be possible to get it working on Windows servers.

The core code will use Open Street Map servers directly. Note this is subject to an Tile usage policy and if your site has heavy traffic, you should use another map provider. See one of the mapping extensions, such as *Mapbox*

5.1 For developers

If you want to develop the core software or an extension, you will need some other software:

- Composer - https://getcomposer.org/

CHAPTER 6

Install

6.1 Install Single Site Mode

Get the code and place on a web server.

(If downloading the code as a release from the website, it is complete. If downloading as source code or from Git, you will need to use composer to install libraries. Run "composer install" in the root directory. https://getcomposer.org/)

Set up a Postgres database.

Copy config.dist.php to config.php and and edit it.

Run the PHP script "core/cli/upgradeDatabase.php".

Run the PHP script "core/cli/loadStaticData.php".

Run the PHP script "core/cli/createUser.php USERNAME EMAIL PASSWORD sysadmin" to create your first user.

Run the PHP script "core/cli/createSite.php SLUG EMAIL" where slug is something like "site". (If the site will always be used in Single Site mode the slug does not matter.)

```
cp config.php.dist config.php
vi config.php
php core/cli/upgradeDatabase.php
php core/cli/loadStaticData.php
php core/cli/createUser.php sam sam@example.com b7ut9U6d sysadmin
php core/cli/createSite.php site sam@example.com
```

The folder "webSingleSite" must be served by the webserver. Enable SSL if possible. Note there is an .htaccess file in there that Apache must use.

The folder "cache/templates.web" must be writable be the webserver.

Enable *a required cron job*.

6.2 Install Single Site Mode where all files are under web root

You will notice that the code has a filesystem where the webroot is under other code. We realise this may make it hard to install on some hosts. You can change the folders around so all code is under the web root.

In "webSingleSite", create the folder "oacapp". For security reasons, set up your web server not to serve this folder using an .htaccess file or other method.

Move all folders except "webSingleSite" underneath your new "oacapp" folder.

Edit the file "webSingleSite/localConfig.php", and change the constant APP_ROOT_DIR so it is defined thus:

```
define('APP_ROOT_DIR', __DIR__.DIRECTORY_SEPARATOR.'oacapp'.DIRECTORY_SEPARATOR);
```

Note when updating in future, be careful not to overwrite this file.

The files under "webSingleSite" can now be deployed onto your webserver, and the rest of the set up instructions for Single Site Mode followed.

6.3 Install Multi Site Mode

6.3.1 Choose domain names

The folder "webIndex" must be served by the webserver, on a single name (eg "www.example.com").

The folder "webSite" must be served by the webserver, on a name that allows any subdomain to reach it (eg "*.example.com").

There must be a common domain between them (eg in this case, it's "example.com"). This is because cookies are shared among them.

6.3.2 Let's go!

Get the code and place on a web server.

(If downloading the code as a release from the website, it is complete. If downloading as source code or from Git, you will need to use composer to install libraries. Run "composer install" in the root directory. https://getcomposer.org/)

Set up a Postgres database.

Copy config.dist.php to config.php and edit it.

Run the PHP script "core/cli/upgradeDatabase.php".

Run the PHP script "core/cli/loadStaticData.php".

Run the PHP script "core/cli/createUser.php USERNAME EMAIL PASSWORD sysadmin" to create your first user.

Create the demo calendar. Run the PHP script "core/cli/createSite.php SLUG EMAIL" where slug was set in your config.

The folder "webIndex" must be served by the webserver, on a single name (eg "www.example.com"). Enable SSL if possible. Note there is an .htaccess file in there that Apache must use.

The folder "webSite" must be served by the webserver, on a name that allows any subdomain to reach it (eg "*.example.com"). Enable SSL if possible. Note there is an .htaccess file in there that Apache must use.

Note you may have to be careful about the order of the apache virtual hosts to make sure that the "webSite" folder does not get requests intended for the "webIndex" folder.

(You may notice folders “webSysAdmin” and “webSysAdminNotSecure” - these have been deprecated and can be ignored.)

The folder “cache/templates.web” must be writable be the webserver.

Enable *a required cron job*.

CHAPTER 7

Install Message Que

7.1 A Message Que is optional

Installing a message que system is completely optional. If you don't install one, the tasks run from cron as configured in your standard install will do the same work eventually anyway.

Installing a message que simply means that in certain important places, work will happen instantly instead of after a delay.

For example, when a user adds a new import the import will be processed and the user can see the results straight away. If you don't have a message que it will happen after a delay.

7.2 Supported Message Que Software - Beanstalkd

To use Beanstalkd, simply configure the options:

```
$CONFIG->useBeanstalkd = true;
$CONFIG->beanstalkdHost = 'localhost';
$CONFIG->beanstalkdPort = 11300;
$CONFIG->beanstalkdTube = 'openacalendar';
```

7.3 Running a Message Que consumer

Simply run

    php core/cli/runMessageQueConsumer.php

Note each worker will die after a certain amount of time, as configured by 'messageQueConsumerProcessRunsForSeconds'. You should regularly start a new one using cron to replace the one that has died.

This is simply a way to control memory usage in long running processes, by making sure the process stops and starts again with a clean footprint regularly.

7.4 Upgrading OpenACalendar when running a Message Que

Before upgrading, stop any existing running message que consumers.

Make sure to restart some new ones after upgrading.

CHAPTER 8

Upgrading

8.1 Place in Read only mode

You can place the site in read only mode by setting these configuration variables:

```
$CONFIG->siteReadOnly = true
$CONFIG->siteReadOnlyReason = "Upgrading"
```

8.2 Stop Message Que

If you are using the message que, stop any workers. See *message que documentation*

8.3 Backup

Backup the database contents and the file store folder, if used - check the "fileStoreLocation" configuration option.

8.4 Replace files

Place the new files over the old ones. (If you have had to change any "localConfig.php" files to accomodate your webserver, be careful to preserve the changes.)

8.5 Upgrade Database

Run the PHP script "php core/cli/upgradeDatabase.php".

Run the PHP script "php core/cli/loadStaticData.php".

8.6 Clear Caches

On servers not in Debug mode, delete the existing cached templates. These are found in "cache/templates.cli" and "cache/templates.web". (It doesn't hurt to do this on debug servers so if in doubt, just do it.)

8.7 Build assets

If you have changed theme variables or are using custom extensions, you should *run the web asset compilation procedure again*.

8.8 Increase asset version

There are Apache config files included in the software that turn on browser caching for assets, such as images, CSS and JS. This should help speed up subsequent page loads for users. However when you update, you must make sure users get the latest version of all assets. To do this, the asset version must be incremented.

Your config.php should contain this variable:

```
$CONFIG->assetsVersion = 1
```

This is set to 1 by default - increase it by 1 every time you perform an upgrade.

8.9 Turn off read only mode

Set

```
$CONFIG->siteReadOnly = false
```

8.10 Start Message Ques

If you are using the message que, start new workers. See *message que documentation*

8.11 Advanced; Minimising errors when upgrading - Database upgrades

Following the upgrade procedure above means that there is a small period of time during which the code has been upgraded and the database hasn't. Users may see errors during that time.

This can be avoided. Database changes are designed to be additions that can be upgraded before the code is upgraded.

Check out the new code into a seperate folder. Give this folder the same config.php and extensions as your normal web app.

Then run the PHP scripts "php core/cli/upgradeDatabase.php" and "php core/cli/loadStaticData.php" from this seperate folder.

Now update the code in your normal web app. The new DB structure will be in place already; thus minimising downtime.

CHAPTER 9

Cron

There is one script to run regularly. This script will take charge of running tasks when relevant.

Run core/cli/runTasksAutomatically.php every 15 minutes or more.

The folder “cache/templates.cli” must be writable by the user Cron jobs run as.

Make sure more than one of the same script is not running at the same time as then you may get race conditions. Sometimes this will not matter, sometimes it will. If you have a large site you will want to increase the intervals to ensure this.

CHAPTER 10

Config

To see what default options are set, see the file core/php/Config.php

10.1 Database

10.1.1 databaseType

“pgsql” is the only current option.

10.2 Site Mode

10.2.1 isSingleSiteMode

Boolean.

10.2.2 singleSiteID

If single site mode, put the ID of the site to use here. If you have created the DB from scratch this should be 1.

10.2.3 siteSlugReserved

Array. If Multi site mode, users will not be allowed to create sites with these slugs.

10.2.4 siteSlugDemoSite

If Multi site mode, this site is a “demo” site that users are directed to as a playbox.

10.3 Web Addresses

Index, Site and Sysadmin need to be different in multi site mode. In single site mode, just set each one to be the same.

Each domain should not include the protocol, eg "example.com" not "http://example.com"

Each domain can include a port number, eg "example.com:1234". This is most useful on dev servers.

10.3.1 hasSSL

Boolean

10.3.2 webSiteAlternateDomains

Array. These are other domains that the site will recognise and redirect to.

10.3.3 webCommonSessionDomain

This is the session for the cookie. In single site mode, just set the same domain. In multi site mode, it needs to go across all domains so set a common parent.

10.4 Site State

10.4.1 isDebug

Boolean, is site in debug mode.

10.4.2 siteReadOnly

Boolean. Can lock whole site during upgrades, maintenance, etc.

10.4.3 siteReadOnlyReason

String. Shown to user.

10.4.4 extensions

Array. List of extensions active.

10.5 Site identity and contact details

10.5.1 emailFrom

Email address

10.5.2 emailFromName

Name that is shown with email address

10.5.3 contactTwitter

Username only.

10.5.4 contactEmail

Email address.

10.5.5 contactEmailHTML

This is the address that is show on the web page. You may wish to do something so spam bots find it harder to scan.

10.6 Media

Media is uploaded pictures.

10.6.1 fileStoreLocation

Location of file store. This should be backed up.

10.6.2 mediaNormalSize

Number. Size in pixels.

10.6.3 mediaThumbnailSize

Number. Size in pixels.

10.6.4 mediaQualityJpeg

Number.

10.6.5 mediaQualityPng

Number.

10.6.6 mediaBrowserCacheExpiresInseconds

Number.

10.6.7 cacheSiteLogoInSeconds

Number. For sites in multi site mode, they can specify that a piece of media is their logo. How long to cache it?

10.7 Security and Antispam

10.7.1 allowNewUsersToRegister

Boolean.

10.7.2 newUsersAreEditors

Boolean.

10.7.3 bcryptRounds

Number.

10.7.4 newUserRegisterAntiSpam

Boolean.

10.7.5 contactFormAntiSpam

Boolean.

10.8 Sys admin interface

10.8.1 sysAdminExtraPassword

Sys admin users have to enter an extra password to access the sys admin interface. Set here as a plain text string.

10.8.2 sysAdminTimeZone

Time Zone used when browsing the Sys Admin interface.

10.8.3 sysAdminLogInTimeOutSeconds

Number.

10.9 Analytics

Built-in vars for adding web analytics to a calendar

10.9.1 piwikServerHTTP

10.9.2 piwikServerHTTPS

String - (If using Piwik Analytics) The URL for the Piwik Server

10.9.3 piwikSiteID

String - (If using Piwik Analytics) The SiteID token for this site in Piwik Analytics

10.9.4 googleAnalyticsTracking

String - (If using Google Web Analytics) The SiteID token for this site in Google Web Analytics

10.10 New Sites

When you create a site, what features are on by default? Most of these can be changed later in calendar admin.

10.10.1 newSiteHasFeatureMap

Boolean

10.10.2 newSiteHasFeatureCuratedList

Boolean

10.10.3 newSiteHasFeatureImporter

Boolean

10.10.4 newSiteHasFeatureGroup

Boolean

10.10.5 newSiteHasFeatureVirtualEvents

Boolean

10.10.6 newSiteHasFeaturePhysicalEvents

Boolean

10.10.7 newSitePromptEmailsDaysInAdvance

Number

10.10.8 newSiteHasQuotaCode

This should be a Quota level you have already set up in the sys admin interface. “BASIC” is configured by default.

10.11 Misc

10.11.1 clockDisplayDefault

“12hr” or “24hr”.

10.11.2 assetsVersion

All assets (CSS, Images, Js, etc) have this appended to the URL. You can tell browsers to cache all assets, and increment this when you update assets to make sure browsers get the latest ones. See [the upgrading guide](/en/systemadministrators/core/upgrading.md) for more.

10.11.3 apiExtraHeader1Html

When HTML is sent from the API, extra content can be added. Use for links to surveys, etc.

10.11.4 apiExtraHeader1Text

When Text is sent from the API, extra content can be added. Use for links to surveys, etc.

10.11.5 apiExtraFooter1Html

When HTML is sent from the API, extra content can be added. Use for links to sponsors, etc.

10.11.6 apiExtraFooter1Text

When Text is sent from the API, extra content can be added. Use for links to sponsors, etc.

10.11.7 canCreateSitesVerifiedEditorUsers

Can verified users create sites from the web index pages. Boolean.

10.11.8 userNameReserved

An array of usernames that other users are not allowed to sign up for.

```
$CONFIG->userNameReserved = array('admin','superadmin');
```

CHAPTER 11

Extensions

Extensions hold templates, setup information, code and assets (JS, CSS, Images). An extension can be added to add additional functionality and theme assets.

11.1 Installing an extension

Check the extension documentation for additional instructions, but the basic process is simply to add the folder into the "extension" folder in the root of the app. Then edit config.php and add the extension.

Note the name of the extension is referenced inside the extension to, so you can not rename extensions at will.

Extensions can provide assets (JS, CSS, Images) and you may need to run the build scripts to copy the right assets into the web folders after installation.

CHAPTER 12

Extension AddressCodeGBOpenCodePoint

12.1 Purpose

If a venue is saved with a post code but no lat/lng point, a lat/lng point will be set. This uses free data from Code-Point Open. http://www.ordnancesurvey.co.uk/business-and-government/products/code-point-open.html

12.2 Useage

If set up properly, it will just work.

CHAPTER 13

Extension Contact

Historically, this was part of the Core software and was moved out to it's own extension for version 1.4.0. It is optional and only needs to be enabled if you want to continue using it.

13.1 Installation

This extension is included with the core software and only needs to be set up.

13.2 Setting up

Edit your config.php and add Contact as an extension.

Run the database upgrade scripts. "php core/cli/upgradeDatabase.php"

CHAPTER 14

Extension CuratedLists

This extension MUST be installed currently.

Historically, this was part of the Core software and was moved out to it's own extension for version v1.4.0. Some parts of the Core software still reference this extension and may cause errors is this extension is not installed.

14.1 Installation

This extension is included with the core software and only needs to be set up.

14.2 Setting up

Edit your config.php and add CuratedLists as an extension.

CHAPTER 15

Extension DisplayBoard

15.1 Installation

This extension is included with the core software and only needs to be set up.

15.2 Setting up

Edit your config.php and add DisplayBoard as an extension.

15.3 Display Board Today Next Later

This shows a grid of events on today, in the next few days and later.

This is available by browsing to /displayboard

From v1.6.1 onwards, it should be accessed by browsing to /displayboard/todaynextlater althought the old link will continue to work.

15.4 Display Board Cycle Details

This show one event on a page, and every set number of seconds (eg 15 seconds) it automatically advances to the next event.

This is only available from v1.6.1 onwards. It should be accessed by browsing to /displayboard/cycledetails

CHAPTER 16

Extension Facebook

16.1 Purpose

This extension provides interaction with Facebook.

16.2 Importing by URL

If someone tries to import a single event from Facebook by URL, this extension will import the data.

If this extension is not set up, Facebook events can not be imported.

16.3 Installation

Edit your config.php and add Facebook as an extension.

Go to the *sysadmin Web UI*.

Go to the url /sysadmin/facebookuser

First you must enter an Facebook App ID and Secret. Obtain one from the Facebook Developers site. https://developers.facebook.com/

This app must be set up as a web app, and it must be given the correct domain of the site.

Once the App ID and Secret have been entered, a FaceBook user must press the "Login with Facebook" button. They may be asked to give the app permissions to read their account. We do not ask for permissions to write to Facebook.

You should now see App ID, App Secret and User Token all filled in.

The extension has been set up.

16.4 Usage

If set up properly, it will just work.

CHAPTER 17

Extension Mapbox

This extension makes your site use Mapbox as a map tile provider.

If you do not configure this (or another map tile provider) your site will use Open Street Map servers directly. Note this is subject to an Tile usage policy and if your site has heavy traffic you should set up an alternative.

Available since v1.6.3.

17.1 Installation

This extension is included with the core software and only needs to be set up.

17.2 Setting up

Edit your config.php and add Mapbox as an extension.

You need to add two config variables:

```
$CONFIG->mapboxProjectId = 'mapbox.streets';
$CONFIG->mapboxPublicAPIToken = 'pk.xxxxxxxxxxxxxx';
```

The API token given here should be a public token.

CHAPTER 18

Extension Meetup

18.1 Purpose

This extension provides interaction with Meetup.

18.2 Importing by URL

If someone tries to import a single event or a group from Meetup by URL, this extension will import the data.

If this extension is not set up, the data will still be imported but in a reduced form. For example, the description will be truncated.

18.3 Installation

Edit your config.php and add Meetup as an extension.

Go to the *sysadmin Web UI*.

Go to the url /sysadmin/meetupuser

You must enter an Meetup App Key. Obtain one from the Meetup API site. https://secure.meetup.com/meetup_api/key/

You should now see that app key has been filled in.

The extension has been set up.

18.4 Useage

If set up properly, it will just work.

CHAPTER 19

Writing An Extension

19.1 Creating the basic Extension

First, run "composer install" in the "build" folder to get libraries that are not shipped with the app.

Then run the command

```
php build/newExtension.php "Sample" "org\openacalendar\sample"
```

The first "Sample" is the name of the folder that will be created.

The second "org\openacalendar\sample" is a PHP namespace that will be used to uniquely identify the extension. It should be your domain name in reverse with an extension part at the end - our domain name is "openacalendar.org".

This will create a bunch of files and folders, and give you instructions on activating the extension by adding it to your config.php.

19.2 Develop in debug mode

When actively developing extensions, it is best to keep your install in debug mode by setting this in config.php.

```
$CONFIG->isDebug = true;
```

19.3 Overriding a template in your extension

Simply copy the template you want to override from the core/theme/default folder into your extension.

For example

```
mkdir -p extension/Sample/theme/default/templates/site/index
cp core/theme/default/templates/site/index/index.html.twig  extension/Sample/theme/
↪default/templates/site/index
```

You can new edit the file extension/Sample/theme/default/templates/site/index/index.html.twig to change the front page of the site!

You may particularly want to do this for the privacy and terms and conditions pages, which by default just say "TODO".

```
mkdir -p extension/Sample/theme/default/templates/index/index
cp core/theme/default/templates/index/index/privacy.html.twig  extension/Sample/theme/
↪default/templates/index/index
cp core/theme/default/templates/index/index/terms.html.twig  extension/Sample/theme/
↪default/templates/index/index
```

19.4 Adding a new web page

(The phrase "web page" is used to indicate a page a human looks at - adding an API end point is slightly different.)

First, you need to create a route to link the URL to your code. This route file should be in

- "extension/Sample/webSite/index.routes.php" - in Multi Site mode, for pages that appear under a particular calendar.
- "extension/Sample/webIndex/index.routes.php" - in Multi Site mode, for pages that appear at the root of the site.

In single Site mode, either place will be included. But there is also:

- "extension/Sample/webSingleSite/index.routes.php" - for pages that should work in Single Site mode only.

It should contain:

```
<?php
$app->match('/askmarc', 'org\openacalendar\sample\controllers\IndexController::askMarc
↪');
```

This links the URL to the controller, which should be placed in "extension/Sample/php/org/openacalendar/sample/controllers/IndexController.php" and contain:

```
<?php
namespace org\openacalendar\sample\controllers;

use Silex\Application;
use Symfony\Component\HttpFoundation\Request;

class IndexController {
    function askMarc(Application $app, Request $request) {
        return $app['twig']->render('extension/sample/index/askMarc.html.twig',␣
↪array());
    }
}
```

This is the code that is called - here you can run any normal PHP. Finally, it renders the twig template. This should be put at "extension/Sample/theme/default/templates/extension/sample/index/about.html.twig":

```
{% extends 'index/page.html.twig' %}

{% block pageTitle %}Ask Marc - {% endblock %}
```

```
{% block content %}
        <p>Ask Marc</p>
{% endblock %}
```

Finally, you need to make sure that the URL /askMarc is routed to index.php by your webserver. This may require a change to the .htaccess in the relevant web folder to add:

```
RewriteRule  ^askmarc(.*)$ /index.php/askmarc  [L]
```

CHAPTER 20

Web API V1

20.1 Introduction

Web API1 is a read only API.

Almost all end points are fully public. Some End points relate to private user information for that user, and a key is part of the URL.

Apart from that, no key, app or authentication is needed to access this API.

20.2 Multi Site or Single Site Mode

In Multi Site mode, it is important to note whether you access these on the index domain or an actual calendar.

In Single Site mode, this does not matter.

20.3 Common Parameters

For end points that return time in a local timezone, the GET parameter "mytimezone" can be passed. This should be:

- One of the timezones that is configured for this calendar - it is from a country that is selected.
- In this format http://php.net/manual/en/timezones.php eg "Europe/London"

20.4 List Events in ATOM format

There are 2 different set of ATOM end points.

One lists events as they are created. This is useful for keeping an eye on a calendar.

The other lists events a certain number of days before they start. This is useful for getting notifications of upcoming events.

20.4.1 Endpoints for a feed as events are created

- /api1/events.create.atom
- /api1/group/{slug}/events.create.atom
- /api1/tag/{slug}/events.create.atom
- /api1/area/{slug}/events.create.atom
- /api1/venue/{slug}/events.create.atom
- /api1/venue/virtual/events.create.atom
- /api1/curatedlist/{slug}/events.create.atom
- /api1/country/{code}/events.create.atom

Slug should be the number only eg /group/1-the-best should become 1.

Country code is the 2 character code eg GB.

20.4.2 Endpoints for a feed before events

- /api1/events.before.atom
- /api1/group/{slug}/events.before.atom
- /api1/tag/{slug}/events.before.atom
- /api1/area/{slug}/events.before.atom
- /api1/venue/{slug}/events.before.atom
- /api1/venue/virtual/events.before.atom
- /api1/curatedlist/{slug}/events.before.atom
- /api1/country/{code}/events.before.atom

Slug should be the number only eg /group/1-the-best should become 1.

Country code is the 2 character code eg GB.

Pass the optional parameter “days” to set how many days. Eg:

- /api1/events.before.atom?days=5

20.5 List Events in ICAL format

20.5.1 End Points

In Multi Site mode, these end points are accesed on the Site pages:

- /api1/events.ical
- /api1/group/{slug}/events.ical
- /api1/tag/{slug}/events.ical

- /api1/area/{slug}/events.ical
- /api1/venue/virtual/events.ical
- /api1/venue/{slug}/events.ical
- /api1/curatedlist/{slug}/events.ical
- /api1/country/{code}/events.ical
- /api1/person/{username}/events.ical

Slug should be the number only eg /group/1-the-best should become 1.

Country code is the 2 character code eg GB.

20.6 List Events in JSON and JSONP format

20.6.1 Parameters

- includeMedias - boolean “true” or “false”, optional, false by default. Whether to include medias attached to the event.

20.7 List Events in JSON format

In Multi Site mode, these end points are accesed on the Site pages:

- /api1/events.json
- /api1/group/{slug}/events.json
- /api1/tag/{slug}/events.json
- /api1/area/{slug}/events.json
- /api1/venue/virtual/events.json
- /api1/venue/{slug}/events.json
- /api1/curatedlist/{slug}/events.json
- /api1/country/{code}/events.json
- /api1/person/{username}/events.json

Slug should be the number only eg /group/1-the-best should become 1.

Country code is the 2 character code eg GB.

20.8 List Events in JSONP format

JSONP end points are the same as the JSON end points, except with a ”.jsonp” extension.

Pass the GET parameter “callback” to specify what javascript function should be called.

eg:

- /api1/events.jsonp?callback=myFunc

20.9 Show Event in ICAL

- /api1/event/{slug}/info.ical

Slug should be the number only eg /group/1-the-best should become 1.

20.10 Show Event in JSON

In Multi Site mode, this end point is accesed on the Site pages:

- /api1/event/{slug}/info.json

Slug should be the number only eg /group/1-the-best should become 1.

20.11 Show Event in JSONP

In Multi Site mode, this end point is accesed on the Site pages:

- /api1/event/{slug}/info.jsonp

Slug should be the number only eg /group/1-the-best should become 1.

Pass the GET parameter "callback" to specify what javascript function should be called.

20.12 List Groups

In Multi Site mode, this end point is accesed on the Site pages:

- /api1/groups.json

20.13 List Histories in ATOM format

This lists every edit made to this calendar as it happens.

Subscribe to this in a app on your phone to be notified of any edits, for instance.

In Multi Site mode, this end point is accesed on the Site pages:

- /api1/histories.atom

CHAPTER 21

CLI API V1

21.1 Introduction

This is a direct API to the app from the Command Line Interface.

There is a configuration setting, CLIAPI1Enabled. This is off by default. Simply turn on to enable the API.

Note it has no security once enabled. Any program can call the API, specifying any user. This is a low level API that is intended to be used by very tight integrations.

If you need a higher-level API with user security, use the Web API V2 instead.

21.2 CLI API V1 Create Event

21.2.1 Since

Available since V1.2.2

Earlier versions had a bug that created bad data and should not be used.

21.2.2 Script

Call core/cliapi1/createEvent.php

21.2.3 Parameters - JSON

Pipe in JSON as Standard input

21.2.4 Parameters - JSON - Event Object

Pass as “event” variable.

Include:

- summary
- description
- timezone - One of http://www.php.net//manual/en/timezones.php Note the start and end time you specify should be in this timezone, not UTC.
- url
- start - A datetime object
- end - A datetime object

The date time object used for start and end should have one of the following:

- str - A string representation. This should be very clear; we recommend “YYYY-MM-DD HH:MM:SS” eg “2014-07-01 10:00:00”.

21.2.5 Parameters - JSON - Country Object

Pass as “country” variable.

Pass one of:

- code - 2 character country code; eg “GB” or “DE”

21.2.6 Parameters - JSON - Site Object

Pass as “site” variable.

Pass one of:

- id - look up in Sysadmin interface.
- slug - eg http://demo.hasacalendar.co.uk/ has slug “demo”

21.2.7 Parameters - JSON - User Object

Pass as “user” variable.

Pass one of:

- username
- email

21.2.8 Parameters - JSON - Group Object

Pass as “group” variable.

Pass one of:

- id - look up in Sysadmin interface.

- slug - eg http://demo.hasacalendar.co.uk/group/2-edinburgh-cat-walkers-group has slug "2"

21.2.9 Parameters - JSON - Example

Example:

```
{
    "event":{
        "summary":"Test",
        "start":{
            "str":"2014-08-01 12:00:00"
        },
        "end":{
            "str":"2014-08-01 12:00:00"
        },
        "country":{
            "code":"DE"
        },
        "timezone":"Europe/Berlin"
    },
    "site":{
        "slug":"test1"
    },
    "group":{
        "slug":"1"
    },
    "user":{
        "email":"james@example.co.uk"
    }
}
```

CHAPTER 22

Web Assets (JS, CSS, Images, etc)

22.1 Compiled from source files in core and extensions

Images, CSS and JavaScript can all be part of the Core code, or part of any extension.

Such files are in "/core/theme/<themename>/" or "/extension/<extensionname>/theme/<themename>/" They are compiled by a build process into one of:

- /webIndex/theme/<themename>
- /webSite/theme/<themename>
- /webSingleSite/theme/<themename>

During compilation, assets with the same name in an extension will override assets in the core. In this way, extensions can override parts of the assets.

22.2 Theme variables

These can be set in the following places:

- in "/core/theme/<themename>/variables.ini"
- in "/extension/<extensionname>/theme/<themename>/variables.ini".
- in the config variable "themeVariables"

If there are variables with the same name, the value from the later place in the list will be used. In this way, extensions and the config can alter the assets.

Variables are used:

- They are automatically available as variables in LESS when CSS is complied.
- When emails are sent, the variables are loaded into variables in Twig so Twig templates can style the emails.

After changing Theme Variables, you need to run the compilation process again. If you have changed the Theme Variables, you should also run the compilation process after upgrading.

Here is an example of setting theme variables in the config:

```
$CONFIG->themeVariables['default'] = array(
    'colourMain'=>'#AF6F6F',
    'colourDarker1'=>'#8D4D4D',
    'colourDarker2'=>'#732C2C',
    'colourLighter1'=>'#D29D9D',
    'colourLighter2'=>'#FDD9D9',
);
```

22.3 Compiling assets

The compilation scripts are in the folder "build".

First, run "composer install" in the build folder to get libraries that are not shipped with the app.

You may need to edit "build/localConfig.php" if you have moved parts of the application around.

Then simply run "php build.php".

The results are automatically placed in *the correct web folders*.

The results are portable; for a given set of extensions and configuration you can compile the assets on a dev machine then move them to production server.

22.4 Non Compiled

Some standard libraries live in the normal web folders and are not compiled in any way.

Any changes (ie upgrading the libraries) will involve a change to the file names.

Because of this, the webserver can tell all web browsers to cache them for a long time.