zest.releaser Documentation

Release 6.12.5.dev0

Reinout and Maurits van Rees

Aug 30, 2017

Contents

Make easy, quick and neat releases of your Python packages. You need to change the version number, add a new heading in your changelog, record the release date, svn/git/bzr/hg tag your project, perhaps upload it to pypi... *zest.releaser* takes care of the boring bits for you.

Here's an overview of the documentation we have for you. First the documentation on *using* zest.releaser:

CHAPTER 1

Package releasing made easy: zest.releaser overview and installation

zest.releaser is collection of command-line programs to help you automate the task of releasing a Python project.

It does away with all the boring bits. This is what zest.releaser automates for you:

- It updates the version number. The version number can either be in `setup.py` or `version.txt` or in a `__versions__` attribute in a Python file or in `setup.cfg`. For example, it switches you from `0.3.dev0` (current development version) to `0.3` (release) to `0.4.dev0` (new development version).
- It updates the history/changes file. It logs the release date on release and adds a new heading for the upcoming changes (new development version).
- It tags the release. It creates a tag in your version control system named after the released version number.
- It optionally uploads a source release to PyPI. It will only do this if the package is already registered there (else it will ask, defaulting to 'no'); zest releaser is careful not to publish your private projects!

Most important URLs

First the three most important links:

- The full documentation is at zestreleaser.readthedocs.io.
- We're on PyPI, so we're only an `pip install zest.releaser` away from installation on your computer.
- The code is at github.com/zestsoftware/zest.releaser.

And... we're automatically being tested by Travis and Landscape:

Compatibility / Dependencies

`zest.releaser` works on Python 2.7. Python 2.6 is not officially supported anymore since version 4.0: it may still work, but we are no longer testing against it. Python 3.3+ is supported.

To be sure: the packages that you release with `zest.releaser` may very well work on other Python versions: that totally depends on your package.

We depend on:

- `setuptools` for the entrypoint hooks that we offer.
- `colorama` for colorized output (some errors printed in red).
- `six` for python2/python3 compatibility.

Since version 4.0 there is a `recommended` extra that you can get by installing `zest.releaser[recommended]` instead of `zest.releaser`. It contains a few trusted add-ons that we feel are useful for the great majority of `zest.releaser` users:

- wheel for creating a Python wheel that we upload to PyPI next to the standard source distribution. Wheels are the new Python package format. Create or edit `setup.cfg` in your project (or globally in your `~/.pypirc`) and create a section `[zest.releaser]` with `create-wheel = yes` to create a wheel to upload to PyPI. See http://pythonwheels.com for deciding whether this is a good idea for your package. Briefly, if it is a pure Python 2 *or* pure Python 3 package: just do it. If it is a pure Python 2 *and* a pure Python 3 project, it is known as a "universal" wheel, because one wheel can be installed on all implementations and versions of Python. If you indicate this in `setup.cfg` with the section `[bdist_wheel]` having `universal = 1`, then we will automatically upload a wheel, unless `create-wheel` is explicitly set to false.
- check-manifest checks your `MANIFEST.in` file for completeness, or tells you that you need such a file. It basically checks if all version controlled files are ending up the the distribution that we will upload. This may avoid 'brown bag' releases that are missing files.
- pyroma checks if the package follows best practices of Python packaging. Mostly it performs checks on the `setup.py` file, like checking for Python version classifiers.
- chardet, the universal character encoding detector. To do the right thing in case your readme or changelog is in a non-utf-8 character set.
- readme to check your long description in the same way as pypi does. No more unformatted restructured text on your pypi page just because there was a small error somewhere. Handy.
- twine for secure uploading via https to pypi. Plain setuptools doesn't support this.

Installation

Just a simple `pip install zest.releaser` or `easy_install zest.releaser` is enough. If you want the recommended extra utilities, do a `pip install zest.releaser[recommended]`.

Alternatively, buildout users can install zest.releaser as part of a specific project's buildout, by having a buildout configuration such as:

```
[buildout]
parts =
```

```
    scripts

[scripts]
recipe = zc.recipe.egg
eggs = zest.releaser[recommended]
```

Version control systems: svn, hg, git, bzr

Of course you must have a version control system installed. zest.releaser currently supports:

- Subversion (svn).
- Mercurial (hg).
- Git (git).
- Git-svn.
- Bazaar (bzr).

Others could be added if there are volunteers! Git and mercurial support have been contributed years ago when we were working with bzr and subversion, for instance.

Available commands

Zest.releaser gives you four commands to help in releasing python packages. They must be run in a version controlled checkout. The commands are:

- **prerelease**: asks you for a version number (defaults to the current version minus a 'dev' or so), updates the setup.py or version.txt and the CHANGES/HISTORY/CHANGELOG file (with either .rst/.txt/.md/.markdown or no extension) with this new version number and offers to commit those changes to subversion (or bzr or hg or git)
- **release**: copies the the trunk or branch of the current checkout and creates a version control tag of it. Makes a checkout of the tag in a temporary directory. Offers to register and upload a source dist of this package to PyPI (Python Package Index). Note: if the package has not been registered yet, it will not do that for you. You must register the package manually (`python setup.py register`) so this remains a conscious decision. The main reason is that you want to avoid having to say: "Oops, I uploaded our client code to the internet; and this is the initial version with the plaintext root passwords."
- **postrelease**: asks you for a version number (gives a sane default), adds a development marker to it, updates the setup.py or version.txt and the CHANGES/HISTORY/CHANGELOG file with this and offers to commit those changes to version control. Note that with git and hg, you'd also be asked to push your changes (since 3.27). Otherwise the release and tag only live in your local hg/git repository and not on the server.
- **fullrelease**: all of the above in order.

There are some additional tools:

- **longtest**: small tool that renders a setup.py's long description and opens it in a web browser. This assumes an installed docutils (as it needs `rst2html.py`).

- **lasttagdiff**: small tool that shows the *diff* of the current branch with the last released tag. Handy for checking whether all the changes are adequately described in the changes file.
- **lasttaglog**: small tool that shows the *log* of the current branch since the last released tag. Handy for checking whether all the changes are adequately described in the changes file.
- **addchangelogentry**: pass this a text on the command line and it will add this as an entry in the changelog. This is probably mostly useful when you are making the same change in a batch of packages. The same text is used as commit message. In the changelog, the text is indented and the first line is started with a dash. The command detects it if you use for example a star as first character of an entry.
- **bumpversion**: do not release, only bump the version. A development marker is kept when it is there. With `--feature` we update the minor version. With option `--breaking` we update the major version.

CHAPTER 2

Options

Zest.releaser tries not too burden you with lots of command line options. Instead, it asks questions while doing its job. But in some cases, a command line option makes sense.

Related: you can change some settings globally in your `~/.pypirc` file or per project in a `setup.cfg` file. This is only for Python packages.

Command line options

These command line options are supported by the release commands (`fullrelease`, `prerelease`, `release`, `postrelease`) and by the `addchangelogentry` command.

-v, --verbose Run in verbose mode, printing a bit more, mostly only interesting for debugging.

-h, --help Display help text

--no-input Don't ask questions, just use the default values. If you are very sure that all will be fine when you answer all questions with the default answer, and you do not want to press Enter several times, you can use this option. The default answers (sometimes yes, sometimes no, sometimes a version number) are probably sane and safe. But do not blame us if this does something you do not want. :-)

The `addchangelogentry` command requires the text you want to add as argument. For example:

```
$ addchangelogentry "Fixed bug."
```

Or on multiple lines:

```
$ addchangelogentry "Fixed bug.

This was difficult."
```

The `bumpversion` command accepts two mutually exclusive options:

- With `--feature` we update the minor version.
- With option `--breaking` we update the major version.

Global options

You can configure zest.releaser for all projects by editing the `.pypirc` file in your home directory. This is the same file that needs to contain your PyPI credentials if you want to release to the Python Packaging Index. See the topic on Uploading. This also has more info on most options.

Lots of things may be in this file, but zest.releaser looks for a `zest.releaser` section, like this:

```
[zest.releaser]
some-option = some value
```

For yes/no options, you can use no/false/off/0 or yes/true/on/1 as answers; upper, lower or mixed case are all fine.

Various options change the default answer of a question. So if you want to use the `--no-input` command line option or want to press Enter a couple of times without thinking too much, see if you can tweak the default answers by setting one of these options

We have these options:

release = yes / no Default: yes. When this is false, zest.releaser sets `no` as default answer for the question if you want to create a checkout of the tag.

create-wheel = yes / no Default: no. Set to yes if you want zest.releaser to create Python wheels. You need to install the `wheel` package for this to work.

If the package is a universal wheel, indicated by having `universal = 1` in the `[bdist_wheel]` section of `setup.cfg`, then the default for this value is yes.

extra-message = [ci skip] Extra message to add to each commit (prerelease, postrelease).

no-input = yes / no Default: no. Set this to yes to accept default answers for all questions.

register = yes / no Default: no. Set this to yes to register a package before uploading. On the official Python Package Index registering a package is no longer needed, and may even fail.

push-changes = yes / no Default: yes. When this is false, zest.releaser sets `no` as default answer for the question if you want to push the changes to the remote.

less-zeroes = yes / no Default: no. This influences the version suggested by the bumpversion command. When set to true:

- Instead of 1.3.0 we will suggest 1.3.
- Instead of 2.0.0 we will suggest 2.0.

version-levels = a number Default: 0. This influences the version suggested by the postrelease and bumpversion commands. The default of zero means: no preference, so use the length of the current number.

This means when suggesting a next version after 1.2:

- with 0 we will suggest 1.3: no change in length
- with 1 we will still suggest 1.3, as we will not use this to remove numbers, only to add them

- with 2 we will suggest 1.3
- with 3 we will suggest 1.2.1

If the current version number has more levels, we keep them. So with `version-levels=1` the next version for 1.2.3.4 will be 1.2.3.5.

development-marker = a string Default: `.dev0` This is the development marker. This is what gets appended to the version in postrelease.

tag-format = a string Default: `{version}` This is a formatter that changes the name of the tag. It needs to contain `{version}`. For backward compatibility, it can contain `%(version)s` instead.

date-format = a string Default: `%%Y-%%m-%%d` This is the format string for the release date to be mentioned in the changelog.

Note: the % signs should be doubled for compatibility with other tools (i.e. pip) that parse setup.cfg using the interpolating ConfigParser.

Per project options

You can change some settings per project by adding instructions for zest.releaser in a `setup.cfg` file. This will only work for a Python package.

These are the same options as the global ones. If you set an option locally in a project, this will override the global option.

CHAPTER 3

Version handling

Where does the version come from?

A version number is essentially what zest.releaser cannot do without. A version number can come from four different locations:

- The `setup.py` file. Two styles are supported:

```
version = '1.0'

def setup(
    version=version,
    name='...
```

  and also:

```
def setup(
    version='1.0',
    name='...
```

- The `setup.cfg` file. zest.releaser will look for something like:

```
[metadata]
name = ...
version = 1.0
```

- If no `setup.py` is found, zest.releaser looks for a `version.txt` file. It should contain just a version number (a newline at the end is OK).

  Originally the `version.txt` was only meant to support really old and ancient Plone packages, but it turned out to be quite useful for non-Python packages, too. A completely static website, for instance, that you *do* want to release and that you *do* want a changelog for.

- A `__version__` attribute in a Python file. You need to tell zest.releaser *which* Python file by adding (or updating) the `setup.cfg` file next to the `setup.py`. You need a `[zest.releaser]` header and a `python-file-with-version` option:

```
[zest.releaser]
python-file-with-version = mypackage/__init__.py
```

Because you need to configure this explicitly, this option takes precedence over any `setup.py` or `version.txt` file.

Where is the version number being set?

Of those four locations where the version can come from, only the first one found is also set to the new value again. Zest.releaser assumes that there's only *one* location.

According to PEP 396, the version should have **one** source and all the others should be derived from it.

Using the version number in `setup.py` or `setup.cfg` as `__version__`

Here are opinionated suggestions from the zest.releaser main authors about how to use the version information. For some other ideas, see the zest.releaser issue 37 discussion.

- The version in the `setup.py` is the real version.
- Add a `__version__` attribute in your main module. Often this will be an `__init__.py`. Set this version attribute with `pkg_resources`, which is automatically installed as part of setuptools/distribute. Here's the code from `zest/releaser/__init__.py`:

```
import pkg_resources

__version__ = pkg_resources.get_distribution("zest.releaser").version
```

This way you can do:

```
>>> import zest.releaser
>>> zest.releaser.__version__
'3.44'
```

- If you use Sphinx for generating your documentation, use the same `pkg_resources` trick to set the version and release in your Sphinx's `conf.py`. See zest.releaser's conf.py.

CHAPTER 4

Uploading to pypi (or custom servers)

When the (full)release command tries to upload your package to a pypi server, zest.releaser basically executes the command `python setup.py sdist` and does a `twine upload`. The twine command replaces the less safe `python setup.py sdist upload`.

For safety reasons zest.releaser will *only* offer to upload your package to https://pypi.python.org when the package is already registered there. If this is not the case yet, you get a confirmation question whether you want to register a new package with `twine register`.

If the upload or register command fails, you probably need to configure your PyPI configuration file. And of course you need to have `setuptools` and `twine` installed, but that is done automatically when installing `zest.releaser`.

PyPI configuration file (`~/.pypirc`)

For uploads to PyPI to work you will need a `.pypirc` file in your home directory that has your pypi login credentials. This may contain alternative servers too:

```
[distutils]
index-servers =
  pypi
  warehouse
  local

[pypi]
# default repository is pypi.python.org
username:maurits
password:secret

[warehouse]
# This is the successor for PyPI, which you can/should already use.
repository:https://upload.pypi.org/legacy/
username:maurits
password:secret
```

```
[local]
repository:http://localhost:8080/test/products/
username:maurits
password:secret
# You may need to specify the realm, which is the domain the
# server sends back when you do a challenge:
#realm:Zope
```

See the Python Packaging User Guide for more info.

When all this is configured correctly, zest.releaser will first upload to the official PyPI (if the package is registered there already). Then it will offer to upload to the other index servers that you have specified in `.pypirc`.

Note that since version 3.15, zest.releaser also looks for this information in the `setup.cfg` if your package has that file. One way to use this, is to restrict the servers that zest.releaser will ask you to upload to. If you have defined 40 index-servers in your pypirc but you have the following in your setup.cfg, you will not be asked to upload to any server:

```
[distutils]
index-servers =
```

Note that after creating the tag we still ask you if you want to checkout that tag for tweaks or pypi/distutils server upload. We could add some extra checks to see if that is really needed, but someone who does not have index-servers listed, may still want to use an entry point like gocept.zestreleaser.customupload to do uploading, or do some manual steps first before uploading.

Since version 6.8, zest.releaser by default no longer *registers* a new package, but only uploads it. This is usually good. See *Registering a package* for an explanation.

Some people will hardly ever want to do a release on PyPI but in 99 out of 100 cases only want to create a tag. They won't like the default answer of 'yes' to that question of whether to create a checkout of the tag. So since version 3.16 you can influence this default answer. You can add some lines to the `.pypirc` file in your home directory to change the default answer for all packages, or change it for individual packages in their `setup.cfg` file. The lines are this:

```
[zest.releaser]
release = no
```

You can use no/false/off/0 or yes/true/on/1 as answers; upper, lower or mixed case are all fine.

Uploading with twine

Since version 6.0, we always use twine for uploading to the Python Package Index, because it is safer: it uses `https` for uploading. Since version 4.0 we already prefered it if it was available, but it is now a core dependency, installed automatically.

Since version 6.6.6 we use it in a way that should work with `twine` 1.6.0 and higher, including future versions.

Uploading wheels

First, you should install the `zest.releaser[recommended]` extra, or run `pip install wheel` yourself next to `zest.releaser`. Then create or edit `setup.cfg` in your project (or globally in your `~/.pypirc`) and add this to create and upload a wheel to upload to PyPI:

```
[zest.releaser]
create-wheel = yes
```

See http://pythonwheels.com for deciding whether this is a good idea for your package. Briefly, if it is a pure Python 2 *or* pure Python 3 package: just do it.

Registering a package

Registering a package does two things:

- It claims a package name on your behalf, so that you can upload a file to it.
- If you already registered the package previously, it updates the general package information. So every time you make a new release, you should register the package.

Well, that used to be the case, but things have changed.

Since version 6.8, zest.releaser by default no longer *registers* a package, but only uploads it. This is because for the standard Python Package Index (PyPI), registering a package is no longer needed: this is done automatically when uploading a distribution for a package. In fact, trying to register may *fail*. See this issue.

But you may be using your own package server, and registering may be wanted or even required there. In this case you will need to turn on the register function. In your `setup.cfg` or `~/.pypirc`, use the following to ensure that register is called on the package server:

```
[zest.releaser]
register = yes
```

If you have specified multiple package servers, this option is used for all of them. There is no way to register and upload to server A, and only upload to server B.

Adding extra text to a commit message

`zest.releaser` makes commits in the prerelease and postrelease phase. Something like `Preparing release 1.0` and `Back to development: 1.1`. You can add extra text to these messages by configuration in your `setup.cfg` or global `~/.pypirc`. One use case for this is telling Travis to skip Continuous Integration builds:

```
[zest.releaser]
extra-message = [ci skip]
```

Signing your commits or tags with git

If you are using git, maybe you want to sign your commits, or more likely your tags, with your gpg key. `zest.releaser` does not do anything special for this: it just calls the normal `git commit` or `git tag`. So if you want to sign anything, you should set this up in your `git` configuration, so it works outside of `zest.releaser` as well. Run these commands to configure gpg signing for git:

```
git config commit.gpgsign true
git config tag.gpgsign true
```

Including all files in your release

By default, only the Python files and a `README.txt` are included (by setuptools) when you make a release. So you miss out on your changelog, json files, stylesheets and so on. There are two strategies to include those other files:

- Add a `MANIFEST.in` file in the same directory as your `setup.py` that lists the files you want to include. Don't worry, wildcards are allowed. Actually, zest.releaser will suggest a sample `MANIFEST.in` for you if you don't already have it. The default is often good enough.
- Setuptools *can* detect which files are included in your version control system (svn, git, etc.) which it'll then automatically include.

The last approch has a problem: not every version control system is supported out of the box. So you might need to install extra packages to get it to work. So: use a `MANIFEST.in` file to spare you the trouble. If not, here are some extra packages:

- setuptools-git (Setuptools plugin for finding files under Git version control)
- setuptools_hg (Setuptools plugin for finding files under Mercurial version control)
- setuptools_bzr (Setuptools plugin for finding files under Bazaar version control)
- setuptools_subversion (Setuptools plugin for finding files under Subversion version control.) You probably need this when you upgrade to the recent subversion 1.7. If you suddenly start missing files in the sdists you upload to PyPI you definitely need it. Alternatively: set up a proper MANIFEST.in as that method works with any version control system.

In general, if you are missing files in the uploaded package, the best is to put a proper `MANIFEST.in` file next to your `setup.py`. See zest.pocompile for an example.

Running automatically without input

Sometimes you want to run zest.releaser without hitting `<enter>` all the time. You might want to run zest.releaser from your automatic test environment, for instance. For that, there's the `--no-input` commandline option. Pass that and all defaults will be accepted automatically.

This means your version number and so must be OK. If you want to have a different version number from the one in your `setup.py`, you'll need to change it yourself by hand. And the next version number will be chosen automatically, too. So `1.2` will become `1.3`. This won't detect that you might want to do a `1.3` after a `1.2.1` bugfix release, but we cannot perform feats of magic in zest.releaser :-)

In case you always want to accept the defaults, a setting in your `setup.cfg` is available:

```
[zest.releaser]
no-input = yes
```

An important reminder: if you want to make sure you never upload anything automatically to the python package index, include the `release = no` setting in `setup.cfg`:

```
[zest.releaser]
no-input = yes
release = no
```

CHAPTER 5

Assumptions

Zest.releaser originated at Zest software so there are some assumptions build-in that might or might not fit you. Lots of people are using it in various companies and open source projects, so it'll probably fit :-)

- If you are using svn, your svn is structured with /trunk, /tags (or /tag) and optionally /branches (or /branch). Both a /trunk or a /branches/something checkout is ok.
- We absolutely need a version. There's a `version.txt` or `setup.py` in your project. The `version.txt` has a single line with the version number (newline optional). The `setup.py` should have a single `version = '0.3'` line somewhere. You can also have it in the actual `setup()` call, on its own line still, as '' version = '0.3',''. Indentation and comma are preserved. If your `setup.py` actually reads the version from your `setup.cfg` (as it does automatically using `setuptools` since version 30.3.0), then the version will be modified there too. If you need something special, you can always do a `version=version` and put the actual version statement in a zest.releaser- friendly format near the top of the file. Reading (in Plone products) a version.txt into setup.py works great, too.
- The history/changes file restriction is probably the most severe at the moment. zest.releaser searches for a restructuredtext header with parenthesis. So something like:

```
Changelog for xyz
=================

0.3 (unreleased)
----------------

- Did something

0.2 (1972-12-25)
----------------

- Reinout was born.
```

  That's just the style we started with. Pretty clear and useful.

CHAPTER 6

Further reading

Mighty fine documentation, the stuff you're reading now. But some other suggestions, ideas and a different tone might help you improve your package releasing. So here are some pointers to other material.

- Big article by Mikko Ohtamaa about high quality automated package releases for Python with zest.releaser. Don't forget to look at the comments.
- Reinout's blog posts tagged 'zestreleaser'.

Photo (by Aidas Bendoraitis) of Reinout introducing zest.releaser at the 2010 Djangocon.eu in Berlin.

Note: If you have something nice to add here, mail Reinout and/or Maurits.

And documentation on zest.releaser as a project; for instance for reporting bugs and fixing the code:

CHAPTER 7

Improving zest.releaser: report bugs, fork on github or email us

Did you find a bug? Do you have an improvement? Do you have questions? We run zest.releaser as a proper open source project on github at https://github.com/zestsoftware/zest.releaser, so you have three basic options:

- Feel free to report bugs on our github issue tracker. And feature requests, too. Normally you'll get a quick reply within a day or so, depending on our relative timezones. If you don't get an answer within a few days, please send off a quick email to remind us.
- Hey, we're on github, so fork away to your heart's delight. We got a couple of nice fixes and additions in this way.

  If you are going to fork zest.releaser, take a look at *Information for developers of zest.releaser* for setup and test running information.
- Email Reinout and Maurits at reinout@vanrees.org and maurits@vanrees.org. Please email us both at the same time, this way at least one of us can give you a quick reply.

CHAPTER 8

Credits

- Reinout van Rees (Nelen & Schuurmans) is the originator and main author.
- Maurits van Rees (Zest Software) added a heapload of improvements.
- Kevin Teague (Canada's Michael Smith Genome Sciences Center) added support for multiple version control systems, most notable Mercurial.
- Wouter vanden Hove (University of Gent) added support for uploading to multiple servers, using collective.dist.
- Godefroid Chapelle (BubbleNet) added /tag besides /tags for subversion.
- Richard Mitchell (Isotoma) added Python 3 support.

CHAPTER 9

Information for developers of zest.releaser

Running tests

We like to use zc.buildout to get a test environment with any versions of needed python packages pinned. When you are in the root folder of your zest.releaser checkout, do this:

```
$ python bootstrap.py  # Or a different python version.
$ bin/buildout
$ bin/test
```

Python versions

The tests currently pass on python 2.7, 3.3, 3.4 & 3.5. Travis continuous integration tests 2.7, 3.3, 3.4 & 3.5 for us automatically.

Necessary programs

To run the tests, you need to have the supported versioning systems installed: svn, hg (mercurial), git and git-svn. On ubuntu:

```
$ sudo apt-get install subversion git bzr mercurial git-svn
```

There may be test failures when you have different versions of these programs. In that case, please investigate as these may be genuine errors.

Setuptools is needed

You also need distribute (or setuptools) in your system path. This is because we basically call 'sys.executable setup.py egg_info' directly (in the tests and in the actual code), which will give an

import error on setuptools otherwise. There is a branch with a hack that solves this but it sounds too hacky.

Configuration for mercurial, bzr and git

For mercurial you may get test failures because of extra output like this:

```
No username found, using 'name@domain' instead.
```

To avoid this, create a file ~/.hgrc with something like this in it:

```
[ui]
username = Author Name <email@address.domain>
```

If you keep having problems, Ubuntu explains it quite good.

There is a similar problem with bazaar. Since bzr version 2.2, it refuses to guess what your username and email is, so you have to set it once, like this, otherwise you get test failures:

```
$ bzr whoami "Author Name <email@address.domain>"
```

And the same for git, so you should do:

```
$ git config --global user.name "Your Name"
$ git config --global user.email you@example.com
```

For release testing we expected a functioning .pypirc file in your home dir, with an old-style configuration, but the tests now use an own config file.

CHAPTER 10

Entrypoints: extending/changing zest.releaser

A zest.releaser entrypoint gets passed a data dictionary and that's about it. You can do tasks like generating documentation. Or downloading external files you don't want to store in your repository but that you do want to have included in your egg.

Every release step (prerelease, release and postrelease) has three points where you can hook in an entry point:

before Only the `workingdir` and `name` are available in the data dictionary, nothing has happened yet.

middle All data dictionary items are available and some questions (like new version number) have been asked. No filesystem changes have been made yet.

after The action has happened, everything has been written to disk or uploaded to pypi or whatever.

For the release step it made sense to create extra entry points:

after_checkout The middle entry point has been handled, the tag has been made, a checkout of that tag has been made and we are now in that checkout directory. Of course, when the user chooses not to do a checkout, this entry point never triggers.

before_upload The source distribution and maybe the wheel have been made. We are about to upload to PyPI with `python setup.py` or `twine upload dist/*`. You may want to use this hook to to sign a distribution before twine uploads it.

Note that an entry point can be specific for one package (usually the package that you are now releasing) or generic for all packages. An example of a generic one is gocept.zestreleaser.customupload, which offers to upload the generated distribution to a chosen destination (like a server for internal company use). If your entry point is specific for the current package only, you should add an extra check to make sure it is not run while releasing other packages; something like this should do the trick:

```
def my_entry_point(data):
    if data['name'] != 'my.package':
        return
    ...
```

Entry point specification

An entry point is configured like this in your setup.py:

```
entry_points={
    #'console_scripts': [
    #    'myscript = my.package.scripts:main'],
    'zest.releaser.prereleaser.middle': [
        'dosomething = my.package.some:some_entrypoint',
        ]},
```

Replace `prereleaser` and `middle` in `zest.releaser.prereleaser.middle` with prerelease/release/postrelease and before/middle/after where needed.

See the `setup.py` of `zest.releaser` itself for some real world examples.

You'll have to make sure that the zest.releaser scripts know about your entry points, for instance by placing your egg (with entry point) in the same zc.recipe.egg section in your buildout as where you placed zest.releaser. Or, if you installed zest.releaser globally, your egg-with-entrypoint has to be globally installed, too.

Comments about data dict items

Your entry point gets a data dictionary: the items you get in that dictionary are documented below. Some comments about them:

- Not all items are available. If no history/changelog file is found, there won't be any `data['history_lines']` either.
- Items that are templates are normal python string templates. They use dictionary replacement: they're actually passed the same data dict. For instance, prerelease's `data['commit_message']` is by default `Preparing release %(new_version)s`. A "middle" entry point could modify this template to get a different commit message.

Common data dict items

These items are shared among all commands.

commit_msg Message template used when committing

headings Extracted headings from the history file

history_encoding The detected encoding of the history file

history_file Filename of history/changelog file (when found)

history_header Header template used for 1st history header

history_insert_line_here Line number where an extra changelog entry can be inserted.

history_last_release Full text of all history entries of the current release

history_lines List with all history file lines (when found)

name Name of the project being released

new_version New version to write, possibly with development marker

nothing_changed_yet First line in new changelog section, warn when this is still in there before releasing

original_version Original package version before any changes

reporoot Root of the version control repository

required_changelog_text Text that must be present in the changelog. Can be a string or a list, for example ["New:", "Fixes:"]. For a list, only one of them needs to be present.

workingdir Original working directory

`prerelease` data dict items

today Date string used in history header

`release` data dict items

tag Tag we're releasing

tag_already_exists Internal detail, don't touch this :-)

tagdir Directory where the tag checkout is placed (*if* a tag checkout has been made)

tagworkingdir Working directory inside the tag checkout. This is the same, except when you make a release from within a sub directory. We then make sure you end up in the same relative directory after a checkout is done.

version Version we're releasing

`postrelease` data dict items

dev_version New version with development marker (so 1.1.dev0)

dev_version_template Template for development version number

development_marker String to be appended to version after postrelease

new_version New version, without development marker (so 1.1)

`addchangelogentry` data dict items

commit_msg Message template used when committing. Default: same as the message passed on the command line.

message The message we want to add

`bumpversion` data dict items

breaking True if we handle a breaking (major) change

clean_new_version Clean new version (say 1.1)

feature True if we handle a feature (minor) change

release Type of release: breaking, feature, normal

CHAPTER 11

Changelog for zest.releaser

6.12.5 (unreleased)

- Nothing changed yet.

6.12.4 (2017-08-30)

- Also support version in setup.cfg. [ewjoachim]

6.12.3 (2017-08-16)

- Allows `{version}` format for `tag-format`. [leorochael]

6.12.2 (2017-07-13)

- Subversion fix: create tag of entire trunk or branch when not in repo root. If you have `trunk/pkg1` and `trunk/pkg2` and you make tag 1.0 in directory `pkg1`, then until now we would create `tags/1.0` with the contents of directory `pkg1`. Checking out the tag and changing to the `pkg1` directory then failed. We now make a tag of the entire trunk or branch, just like in the other version control systems. Fixes issue #213. [maurits]
- Do not needlessly run `svn info`. [maurits]

6.12.1 (2017-07-03)

- Quote the path when making a git clone, to fix problems with spaces. [halkeye]

- Fixed percentage signs in `date-format` in `setup.cfg`. You need double percentages. [mgedmin]

6.12 (2017-06-19)

- Add date format in the config. Default is ISO-8601 (%Y-%m-%d). Put `date-format = format string` in your `~/.pypirc` or `setup.cfg`. [mgedmin]

6.11 (2017-06-09)

- If the package wants to build universal wheels by setting `[bdist_wheel] universal = 1`, then the default for `create-wheel` is now yes.

6.10 (2017-04-18)

- Corner case fix: a top-level `version = 1.0` in your `setup.py` is now also allowed to be in uppercase, like `VERSION = 1.0`. This fixes issue 216. [reinout]

6.9 (2017-02-17)

- Add tag formatter in the config. This is a formatter that changes the name of the tag. Default is the same as the version. Put `tag-format = a string` in your `~/.pypirc` or `setup.cfg`. It needs to contain `%(version)s`. [tcezard]

6.8.1 (2017-01-13)

- Catch error when uploading first package file in new PyPI project. This fixes issue 206. [maurits]

6.8 (2016-12-30)

- Before retrying a `twine` command, reload the pypi config. Then when the user fixes his account settings in `~/.pypirc` and retries, these changes take effect. This used to work a while ago, but got broken. [maurits]
- Added `development-marker` config option. With this can override the default `.dev0`. [drucci]
- Added `version-levels` and `less-zeroes` options. This influences the suggested version. [maurits]
- Allow `.pypirc` with just a `pypi` section. Previously, we required either a `[server-login]` section with a `username` option, or a `[distutils]` section with an `index-servers` option. Failing this, we gave a warning about a not properly configured file, and happily continued without uploading anything. Now if there is something missing from the `pypirc` file, we give an error and explicitly ask if you want to continue without uploading. Fixes issue #199.

Note for developers of extensions for `zest.releaser`: this removes the `is_old_pypi_config` and `is_new_pypi_config` methods, because they made no sense anymore. If you were using these, see if you can use the `distutils_server` method instead. [maurits]

- Added `push-changes` config file option. Default: yes. When this is false, zest.releaser sets `no` as default answer for the question if you want to push the changes to the remote. [newlog]
- By default no longer register a new package, but only upload it. Registering a package is no longer needed on PyPI: uploading a new distribution takes care of this. If you *do* want to register, maybe because a different package server requires it, then in your `setup.cfg` or `~/.pypirc`, use the following:

```
[zest.releaser]
register = yes
```

Fixes issue 191. [willowmck]

6.7.1 (2016-12-22)

- Create the list of distributions after the `before_upload` hook has fired. This allows the `before_upload` hook to create additional distributions, which will then be uploaded. [t-8ch]

6.7 (2016-10-23)

- Use the intended API of twine. This should work with twine 1.6.0 and higher, including future versions. [maurits]

6.6.5 (2016-09-12)

- Support and require twine 1.8.0 as minimum version. Fixes https://github.com/zestsoftware/zest.releaser/issues/183 [maurits]
- Updated the documentation on uploading. [mgedmin, maurits]
- Replaced http://zestreleaser.readthedocs.org with http://zestreleaser.readthedocs.io. This is the new canonical domain since 28 April 2016. [maurits]

6.6.4 (2016-02-24)

- Really create a shallow git clone when creating a distribution. See issue #169. [maurits]

6.6.3 (2016-02-24)

- Using a "shallow" git clone when creating a distribution. This speeds up releases, especially on big repositories. See issue #169. [gforcada]

6.6.2 (2016-02-11)

- Added `no-input` option also to global (.pypirc) options. Issue #164. [jcerjak]

6.6.1 (2016-02-02)

- Fixed version in changelog after bumpversion call. [maurits]

6.6.0 (2016-01-29)

- Added `bumpversion` command. Options `--feature` and `--breaking`. Issue #160. The exact behavior might change in future versions after more practical experience. Try it out and report any issues you find. [maurits]
- Fixed possible encoding problems when writing files. This is especially for an ascii file to which we add non ascii characters, like in the `addchangelogentry` command. [maurits]
- Added `addchangelogentry` command. Issue #159. [maurits]
- Moved `_diff_and_commit`, `_push` and `_grab_version` to `baserelease.py`, as the first was duplicated and the second and third may be handy for other code too. `_grab_version` is the basic implementation, and is overridden in the prereleaser. [maurits]
- Show changelog of current release before asking for the new version number. Issue #155. [maurits]
- Moved `_diff_and_commit`, `_push` and `_grab_version` to `baserelease.py`, as the first was duplicated and the second and third may be handy for other code too. `_grab_version` is the basic implementation, and is overridden in the prereleaser. [maurits]

6.5 (2016-01-05)

- Adjusted `bin/longtest` for the (necessary) rename of the `readme` library to `readme_renderer`. Fixes #153

  Note: the current `readme` package on pypi is broken to force an upgrade. If you use an older zest.releaser, you have to pin `readme` to `0.6.0`, it works just fine. [reinout]

6.4 (2015-11-13)

- Fixed error when retrying twine command. Fixes #148 [maurits]

6.3 (2015-11-11)

- Fixed exception when logging an exception when a twine command fails. [maurits]

6.2 (2015-10-29)

New:

- Use `twine` as library instead of as command. You no longer need to have `twine` on your `PATH`. Fixes issue #142. [maurits]

6.1 (2015-10-29)

Fixes:

- Fixed registering on servers other than PyPI. We forgot to specify the server in that case. [maurits]

6.0 (2015-10-27)

- Made `twine` a core dependency. We now always use it for registering and uploading. We require at least version 1.6.0, as this introduces the `register` command. [maurits]
- When uploading with `twine` first use the `twine register` command. On PyPI, when the project is already registered, we do not call it again, but we can only check this for PyPI, not for other servers. Issue #128. [maurits]
- Always exit with error code 1 when we exit explicitly. In some cases we would exit with success code 0 when we exited based on the answer to a question. This happened when the user did not want us to create the missing `tags` directory in subversion, and also after asking if the user wanted to continue even though 'nothing changed yet' was in the history. [maurits]
- Extensions can now tell zest.releaser to look for specific required words in the history. Just add `required_changelog_text` to the prerelease data. It can be a string or a list, for example `["New:", "Fixes:"]`. For a list, only one of them needs to be present. [maurits]
- Look for the 'Nothing changed yet' text in the complete text of the history entries of the current release, instead of looking at it line by line. This means that zest releaser extensions can overwrite `nothing_changed_yet` in the prerelease data to span multiple lines. [maurits]
- zest.releaser extensions can now look at `history_insert_line_here` in the prerelease data. On this line number in the history file they can add an extra changelog entry if wanted. [maurits]
- Added `history_last_release` to the prerelease data. This is the text with all history entries of the current release. [maurits]
- When using the `--no-input` option, show the question and the chosen answer. Otherwise in case of a problem it is not clear why the command stopped. Fixes issue #136. [maurits]

5.7 (2015-10-14)

- The history/changelog file is now written back with the originally detected encoding. The functionality was added in 5.2, but only used for writing the `setup.py`, not the changelog. This is fixed now. [reinout]

5.6 (2015-09-23)

- Add support for PyPy. [jamadden]

5.5 (2015-09-05)

- The `bin/longtest` command adds the correct utf-8 character encoding hint to the resulting html so that non-ascii long descriptions are properly rendered in all browsers. [reinout]

5.4 (2015-08-28)

- Requiring at least version 0.6 of the (optional, btw) readme package. The API of readme changed slightly. Only needed when you want to check your package's long description with `bin/longtest`. [reinout]

5.3 (2015-08-21)

- Fixed typo in svn command to show the changelog since the last tag. [awello]

5.2 (2015-07-27)

- When we find no version control in the current directory, look a few directories up. When looking for version and history files, we look in the current directory and its sub directories, and not in the repository root. After making a tag checkout, we change directory to the same relative path that we were in before. You can use this when you want to release a Python package that is in a sub directory of the repository. When we detect this, we first offer to change to the root directory of the repository. [maurits]
- Write file with the same encoding that we used for reading them. Issue #109. [maurits]

5.1 (2015-06-11)

- Fix writing history/changelog file with non-ascii. Issue #109. [maurits]
- Release zest.releaser as universal wheel, so one wheel for Python 2 and 3. As usual, we release it also as a source distribution. [maurits]
- Regard "Skipping installation of __init__.py (namespace package)" as warning, printing it in magenta. This can happen when creating a wheel. Issue #108. [maurits]

5.0 (2015-06-05)

- Python 3 support. [mitchellrj]

- Use the same *readme* library that PyPI uses to parse long descriptions when we test and render them. [mitchellrj]

4.0 (2015-05-21)

- Try not to treat warnings as errors. [maurits]
- Allow retrying some commands when there is an error. Currently only for commands that talk to PyPI or another package index. We ask the user if she wants to retry: Yes, no, quit. [maurits]
- Added support for twine. If the `twine` command is available, it is used for uploading to PyPI. It is installed automatically if you use the `zest.releaser[recommended]` extra. Note that if the `twine` command is not available, you may need to change your system `PATH` or need to install `twine` explicitly. This seems more needed when using `zc.buildout` than when using `pip`. Added `releaser.before_upload` entry point. Issue #59. [maurits]
- Added `check-manifest` and `pyroma` to the `recommended` extra. Issue #49. [maurits]
- Python 2.6 not officially supported anymore. It may still work, but we are no longer testing against it. [maurits]
- Do not accept `y` or `n` as answer for a new version. [maurits]
- Use `colorama` to output errors in red. Issue #86 [maurits]
- Show errors when uploading to PyPI. They were unintentionally swallowed before, so you did not notice when an upload failed. Issue #84. [maurits]
- Warn when between the last postrelease and a new prerelease no changelog entry has been added. '- Nothing changed yet' would still be in there. Issue #26. [maurits]
- Remove code for support of collective.sdist. That package was a backport from distutils for Python 2.5 and earlier, which we do not support. [maurits]
- Add optional support for uploading Python wheels. Use the new `zest.releaser[recommended]` extra, or run `pip install wheel` yourself next to `zest.releaser`. Create or edit `setup.cfg` in your project (or globally in your `~/.pypirc`) and create a section `[zest.releaser]` with `create-wheel = yes` to create a wheel to upload to PyPI. See http://pythonwheels.com for deciding whether this is a good idea for your package. Briefly, if it is a pure Python 2 *or* pure Python 3 package: just do it. Issue #55 [maurits]
- Optionally add extra text to commit messages. This can be used to avoid running Travis Continuous Integration builds. See http://docs.travis-ci.com/user/how-to-skip-a-build/. To activate this, add `extra-message = [ci skip]` to a `[zest.releaser]` section in the `setup.cfg` of your package, or your global `~/.pypirc`. Or add your favorite geeky quotes there. [maurits]
- Fix a random test failure on Travis CI, by resetting `AUTO_RESPONSE`. [maurits]
- Added clarification to logging: making an sdist/wheel now says that it is being created in a temp folder. Fixes #61. [reinout]

3.56 (2015-03-18)

- No need anymore to force .zip for sdist. Issue #76 [reinout]

- Still read `setup.cfg` even if `~/.pypirc` is wrong or missing. Issue #74 [tomviner]

3.55 (2015-02-03)

- Experimental work to ignore setuptools' stderr output. This might help with some of the version warnings, which can break zest.releaser's output parsing. [reinout]
- Fix for #72. Grabbing the version from the `setup.py` on windows can fail with an "Invalid Signature" error because setuptools cannot find the crypto dll. Fixed by making sure setuptools gets the full `os.environ` including the `SYSTEMROOT` variable. [codewarrior0]

3.54 (2014-12-29)

- Blacklisting `debian/changelog` when searching for changelog-like filenames as it gets picked in favour of `docs/changelog.rst`. The debian one is by definition unreadable for us.

3.53.2 (2014-11-21)

- Additional fix to 3.53: `version.rst` (and .md) also needed to be looked up in a second spot.

3.53 (2014-11-10)

- Also allowing .md extension in addition to .rst/.txt/.markdown for `CHANGES.txt`. [reinout]
- Similarly, `version.txt` (if you use that for non-setup.py-projects) can now be `version.rst` or .md/.markdown, too. [reinout]

3.52 (2014-07-17)

- Fixed "longtest" command when run with a python without setuptools installed. Similar fix to the one in 3.51. See https://github.com/zestsoftware/zest.releaser/issues/57 [reinout]

3.51 (2014-07-17)

- When calling `python setup.py` use the same PYTHONPATH environment as the script has. https://github.com/zestsoftware/zest.releaser/issues/24 [maurits]

3.50 (2014-01-16)

- Changed command "hg manifest" to "hg locate" to list files in Mercurial. The former prints out file permissions along with the file name, causing a bug. [rafaelbco]

3.49 (2013-12-06)

- Support git-svn checkouts with the default "origin/" prefix. [kuno]

3.48 (2013-11-26)

- When using git, checkout submodules. [dnozay]

3.47 (2013-09-25)

- Always create an egg (`sdist`), even when there is no proper pypi configuration file. This helps plugins that use our entry points. Fixes https://github.com/zestsoftware/zest.releaser/issues/45 [maurits]

3.46 (2013-06-28)

- Support actually updating `VERSION` as well. Issue #43.

3.45 (2013-04-17)

- Supporting `VERSION` (without extension) in addition to the old-zope-products-`VERSION.txt` files.

3.44 (2013-03-21)

- Added optional `python-file-with-version` setting for the `[zest.releaser]` section in `setup.cfg`. If set, zest.releaser extracts the version from that file's `__version__` attribute. (See PEP 396).
- File writes now use the platform's default line endings instead of always writing `\n` unix style line endings. (Technically, we write using `w` instead of `wb` mode).
- Added link to other documentation sources in the sphinx docs.
- Noting in our pypi classifiers that we support python 2.6+, not python 2.4/2.5. Slowly things will creep into zest.releaser's code that break compatibility with those old versions. And we want to get it to work on python 3 and that's easier with just 2.6/2.7 support.

3.43 (2013-02-04)

- Added `--no-input` commandline option for running automatically without asking for input. Useful when started from some build tool. See the documentation at the end of http://zestreleaser.readthedocs.io/en/latest/uploading.html . [reinout, based upon a patch by j-san]

3.42 (2013-01-07)

- When finding multiple version, changes or history files, pick the one with the shortest path. [maurits]
- Support project-specific hooks listed in setup.cfg. [iguananaut]

3.41 (2012-11-02)

- Getting the version from setup.py can give a traceback if the setup.py has an error. During prerelease this would result in a proposed version of 'Traceback'. Now we print the traceback and quit. [maurits]

3.40 (2012-10-13)

- Support svn (1.7+) checkouts that are not directly in the root. Only applies when someone checks out a whole tree and wants to release one of the items in a subdirectory. Fixes #27.

3.39 (2012-09-26)

- Only search for files in version control. This is when finding a history file or version.txt file. We should not edit files that are not in our package. Fixes issue #22. [maurits]

3.38 (2012-09-25)

- Fixed svn tag extraction on windows: a `\r` could end up at the end of every tag name. Thanks Wouter Vanden Hove for reporting it!
- Small fixes to the developers documentation and to the automatic travis CI tests configuration.

3.37 (2012-07-14)

- Documentation update! Started sphinx documentation at zestreleaser.readthedocs.io. Removed documentation from the README and put it into sphinx.
- Actually ask if the user wants to continue with the release when there is no MANIFEST.in. We asked for a yes/no answer, but the question was missing. [maurits]

3.36 (2012-06-26)

- Improved changes/history file detection and fixed the documentation at this point. We now recognize CHANGES, HISTORY and CHANGELOG with .rst, .txt, .markdown and with no extension.
- Set up travis CI integration. Our tests pass on python 2.5, 2.6 and 2.7.

3.35 (2012-06-21)

- When checking for recommended files, ask if the user wants to continue when we suspect the created PyPI release may be broken. See issue #10. [maurits]
- Preserve existing EOL in setup.py and history file (See http://docs.python.org/tutorial/inputoutput.html#reading-and-writing-files) [tom_gross]

3.34 (2012-03-20)

- In the warning about a missing MANIFEST.in file, also suggest to install setuptools_subversion/git, etc. Fixes issue #4. [maurits]

3.33 (2012-03-20)

- Fix python 2.4 issues with tarfile by always creating a zip file. Formerly we would only do this when using python2.4 for doing the release, but a tarball sdist created by python2.6 could still break when the end user is using python 2.4. [kiorky]

3.32 (2012-03-09)

- In prerelease recommend the user to add a MANIFEST.in file. See http://docs.python.org/distutils/sourcedist.html for more info. [maurits]

3.31 (2012-02-23)

- Fixed test for unadvised egg_info commands on tag, which could result in a ConfigParser error. [maurits]

3.30 (2011-12-27)

- Added some more PyPI classifiers. Tested with Python 2.4, 2,4, 2.6, and 2.7. [maurits]
- Moved changes of 3.15 and older to docs/HISTORY.txt. [maurits]
- Added GPL license text in the package. [maurits]
- Updated README.txt. Added MANIFEST.in. [maurits]

3.29 (2011-12-27)

- In postrelease create a version number like 1.0.dev0. See http://www.python.org/dev/peps/pep-0386 [maurits]

- Offer to cleanup setup.cfg on the tag when releasing. You do not want tag_build or tag_svn_revision options in a release usually. [maurits]
- For convenience also print the tag checkout location when only doing a release (instead of a fullrelease). [maurits]

3.28 (2011-11-18)

- Git: in pre/postrelease only check for uncommitted changes in files that are already tracked. [maurits]

3.27 (2011-11-12)

- Postrelease now offers (=asks) to push your changes to the server if you're using hg or git.
- Support for some legacy projects, often converted from CVS, have multiple subprojects under a single trunk. The trunk part from the top level project isn't erroneously stripped out anymore. Thanks to Marc Sibson for the fix.

3.26 (2011-11-01)

- Added sanity check before doing a prerelease so you are warned when you are about to commit on a tag instead of a branch (or trunk or master). [maurits]

3.25 (2011-10-28)

- Removed special handling of subversion lower than 1.7 when searching for the history/changes file. In corner cases it may be that we find a wrong HISTORY.txt or CHANGES.txt file when you have it buried deep in your directory structure. Please move it to the root then, which is the proper place for it. [maurits]
- Fixed finding a history/changes file that is in a sub directory when using subversion 1.7 or higher or bazaar. [maurits]

3.24 (2011-10-19)

- Note: you may need to install setuptools_subversion when you use subversion 1.7. If you suddenly start missing files in the sdists you upload to PyPI you definitely need it. Alternatively: set up a proper MANIFEST.in as that method works with any version control system. [maurits]
- Made compatible with subversion 1.7 (the only relevant change is in the code that checks if a tags or tag directory already exists). Earlier versions of subversion are of course still supported. [maurits]
- Code repository moved to github: https://github.com/zestsoftware/zest.releaser [maurits]

3.23 (2011-09-28)

- Fixed opening the html long description in `longtest` on Mac OS X Lion or python2.7 by using a `file://` url. Fixes https://bugs.launchpad.net/zest.releaser/+bug/858011 [maurits]

3.22 (2011-05-05)

- Allow specifying a tag on the command line when using lasttaglog or lasttagdiff, to show the log or diff since that tag instead of the latest. Useful when you are on a branch and the last tag was from trunk. [maurits]

3.21 (2011-04-20)

- Added lasttaglog command that list the log since the last tag. [maurits]
- Fix Mercurial (hg) support. As spreaded_internal should be set to False (as it happens with git) [erico_andrei]
- Accept a twiggle (or whatever '~' is called) when searching for headers in a changelog; seen in some packages (at least zopeskel.dexterity). [maurits]

3.20 (2011-01-25)

- Also allowing CHANGES.rst and CHANGES.markdown in addition to CHANGES.txt.

3.19 (2011-01-24)

- No longer refuse to register and upload a package on pypi if it is not there yet, forcing people to do this manually the first time. Instead, we ask the question and simply have 'No' as the default answer. If you specify an answer, we require exactly typing 'yes' or 'no'. The idea is still to avoid making it too easy to release an internal package on pypi by accident. [maurits]

3.18 (2010-12-08)

- Added `--non-interactive--` option to the `svn diff` command used in lasttagdiff. This makes it usable in cronjobs and post-commit hooks. Fixes https://bugs.launchpad.net/zest.releaser/+bug/687530

3.17 (2010-11-17)

- When the package that is being released neither has a setup.py nor a setup.cfg, use No as default answer for creating a checkout of the tag. [maurits]

3.16 (2010-11-15)

- For (pypi) output, also show the first few lines instead of only the last few. [maurits]
- See if pypirc or setup.cfg has a [zest.releaser] section with option release = yes/no. During the release stage, this influences the default answer when asked if you want to make a checkout of the tag. The default when not set, is 'yes'. You may want to set this to 'no' if most of the time you only make releaser of internal packages for customers and only need the tag. [maurits]
- Specify bazaar (bzr) tag numbers using the 'tag' revision specifier (like 'tag:0.1') instead of only the tag number (0.1) to add compatibility with earlier bzr versions (tested with 2.0.2). [maurits]

3.15 (2010-09-10)

- Read pypi config not only from the .pypirc file, but also from the setup.cfg file of the package. Patch by Erico Andrei. [maurits]

3.14 (2010-08-26)

- experimental support for git svn tagging, fully test-covered [chaoflow]
- fail if no tag was created, not test-covered [chaoflow]
- svn available_tags method: intercept 'Repository moved' note in svn info and stop processing then. [maurits]

3.13 (2010-08-16)

- Fixed check that tested whether a package was already available on pypi, as the pypi implementation changed slightly. We now just check for a 404 status. Patch by Wolfgang Schnerring. [maurits]

3.12 (2010-07-22)

- Added extra entry point for the release step: `after_checkout`. When this is run, the middle entry point has been handled, the tag has been made, a checkout of that tag has been made and we are now in that checkout directory. Idea: Jan-Wijbrand Kolman. [maurits]
- Fix: in the zest.releaser.releaser.after entry point data, pass the 'tagdir' value (if a checkout has been made). Patch by Wolfgang Schnerring, thanks! [maurits]
- Fixed tests to also pass with slightly newer git. [maurits]

3.11 (2010-06-25)

- Small tweak: allowing zc.rst2's "rst2 html" in addition to docutils' own "rst2html".

3.10 (2010-06-15)

- Fix : when running 'release' with python2.6 against a private egg server, the distutils 'register' command would run against PyPI while 'upload' command would run against private server. (-r option needs to be stated twice) [gotcha]

3.9 (2010-06-11)

- Again at the end of a fullrelease report the location of the directory containing the checkout of the tag, if it has been made. [maurits]

3.8 (2010-05-28)

- Also allowing `CHANGES` in addition to `HISTORY.txt` and `CHANGES.txt` as a history filename. Keeps several Django packages happy.

3.7 (2010-05-07)

- Added support for bzr. Fixes https://bugs.launchpad.net/zest.releaser/+bug/490816 [menesis]

3.6 (2010-04-13)

- A `version='1.0',` string inside the `setup()` call no longer has non-pep8 spaces around the =. Fixes https://bugs.launchpad.net/zest.releaser/+bug/562122 [reinout]
- Got rid of ugly setup.py hack with UltraMagicString that was meant to avoid encoding errors when registering this package at pypi but which was not working for python2.4 (at least with collective.dist). Only ascii is allowed in the long_description if you want to avoid problems at one point or another. [maurits]

3.5 (2010-02-26)

- Treat CHANGES.txt and HISTORY.txt the same: the first that is found in a directory is chosen for changing, instead of first looking everywhere for a HISTORY.txt and then for a CHANGES.txt. [maurits]

3.4 (2010-02-02)

- Always build zip files if using python2.4 [do3cc]
- bugfix: added 'spreaded_internal' property to BaseVersionControl objects, so filefind() does not exclude a directory just because there is no '.git' folder in it. It still excludes directory where there is no '.svn' folder in SVN repositories. [vincent]

3.3 (2009-12-29)

- Fixed test failures when run on a computer with a new style pypi config. We now always use an old style config when running the tests. [maurits]
- Fixed the release command for hg 1.1 (e.g. Ubuntu 9.04). [maurits]

3.2 (2009-12-22)

- Replaced commands.getoutput() with a system() function grabbed from buildout on suggestion by Adam Groszer. Goal: make zest.releaser work also on windows.
- Improved entry point documentation.
- Added launchpad bugtracker at https://bugs.launchpad.net/zest.releaser (and pointing at that in the documentation).

3.1 (2009-11-27)

- Added documentation for entry points. [reinout]

3.0 (2009-11-27)

- Added support for extension by means of entry points. There is no documentation that advertises it yet as I want to treat it as experimental till I've used it a few times. [reinout]

2.12 (2009-11-26)

- Fixed mercurial sdist creation. [reinout]
- A missing history file does not result anymore in a keyerror in prerelease. [reinout]
- Added lots of test output normalization so that errors aren't hidden by the large number of `...` in the doctests. [reinout]

2.11 (2009-11-25)

- Added /tag besides /tags for subversion [gotcha]
- Fixed tests failures. [gotcha]

2.10 (2009-10-22)

- Added support for git. [reinout]
- Lots of internal refactoring and small fixes. [reinout]

- Started tests. zest.releaser went from 0 to 94% coverage. [reinout]

2.9.3 (2009-09-22)

- Uploading to multiple package indexes should now work in python2.6 (though ironically it now does not work for me on python2.4, but that has nothing to do with zest.releaser.) Added documentation for this. [maurits]
- Make sure the next version suggestion for 1.0rc6 is 1.0rc7. [maurits]
- In subversion, first try to get the package from the setup.py before falling back to the svn info, just like for mercurial. This fixes the problem that e.g. Products.feedfeeder was not recognized as being on pypi as the svn directory name was feedfeeder. [maurits]

2.9.2 (2009-09-17)

- Umlauts in a changelog don't break the logger anymore when using python2.6.2 when the umlauts turn up in the diff. This is due to a 2.6.2 regression bug, see http://bugs.python.org/issue5170. Should be fixed in 2.6.3 when it comes out. [reinout]
- (Release 2.9 and 2.9.1 are unreleased because of a setuptools bug with, sigh, non-ascii characters which made a dirty setup.py hack necessary). [reinout]

2.8 (2009-08-27)

- Fixed the release command when used in a french environment. In French "svn info" returns 'URL :', not 'URL:'. [vincentfretin]

2.7 (2009-07-08)

- Before asking setup.py for its version or name, first run egg_info, as that may get rid of some warnings that otherwise end up in the extracted version or name, like UserWarnings. [maurits]

2.6 (2009-05-25)

- Small change: the questions don't print a newline anymore after the question (and before the user pressed enter). This makes it clearer if enter has been pressed yet. Suggestion by jkolman. [reinout]

2.5 (2009-05-20)

- Revert to previous behaviour: when a package has not been released yet on pypi, decline to register it: the first time should be deliberate and manual, to avoid accidentally uploading client packages to pypi. [maurits]

2.4 (2009-05-15)

- Factored release.py out into a new pypi.py, solving a few possible problems with missing or misconfigured .pypirc files. [maurits]

2.3 (2009-05-11)

- Fixed release script when the .pypirc file does not contain a distutils section or that section does not contain a index-servers option. [maurits]

2.2 (2009-05-11)

- postrelease: suggestion for next version after 1.1.19 is not 1.1.110, but 1.1.20. [maurits]
- Make it work with collective.dist (mregister/mupload) [WouterVH] see http://plone.org/documentation/tutorial/how-to-upload-your-package-to-plone.org/installing-collective.dist

2.1 (2009-04-09)

- Fix lasttagdiff command to work with Mercurial by truncating the '+' character from the revision id, since that only indicates uncommitted changes exist.
- Make sure we find package/name/HISTORY.txt before we find docs/HISTORY.txt. [maurits]
- Fixed checking for self.internal_filename: we would incorrectly check ('.', 's', 'v', 'n') instead of '.svn'. [maurits]

2.0 (2009-04-01)

- Added tag_url method to get lasttagdiff (and zest.stabilizer) working again. [maurits]
- Merged kteague-multi-vcs branch with, woohoo, mercurial support! [reinout]
- Mercurial support by Kevin Teague. [kteague]
- `postrelease` put a space in the new version number in `setup.py` (between version number and `dev`). Removed this space as it is not necessary (in best case). [icemac]

1.13 (2009-03-17)

- Also looking for `CHANGES` in addition to `HISTORY.txt` and `CHANGES.txt` as some packages use that convention. [reinout]
- Added `lasttagdiff` command that shows the diff between the last release and the currently committed trunk. Handy for checking whether the changelog is up to date. [reinout]

1.12 (2009-03-17)

- When doing a fullrelease and if the release step made a checkout of the tag into an temp directory, that temp directory is again printed after fullrelease finishes. Otherwise you've got to do a lot of scrolling. [reinout]

1.11 (2009-03-04)

- When the found history file contains no version headings, look for a second history file: more than once I have the standard docs/HISTORY.txt that paster creates and I just add a pointer there to the real package/name/HISTORY.txt. [maurits]

1.10 (2009-02-25)

- A '' version = '1.0','' in setup.py is now also rewritten correctly. Previously just a `version = '1.0'` would be injected, so without indentation and comma. [reinout]
- Ask before checking out the tag. Sometimes the checkout is huge and you know you don't want it. You don't get asked for a pypi upload, though if you don't check out the tag. [reinout]

1.9 (2009-02-24)

- 'release' now also makes a tag checkout in a temporary directory. [Reinout]
- Made 'longtest' work on Linux as there the command is 'rst2html' and apparently on the Mac it is 'rst2html.py'. [maurits]

1.8 (2009-02-23)

- Added 'longtest' command that renders a setup.py's long description and opens it in a web browser. [reinout]

1.7 (2009-02-16)

- Supporting alternative history version header format: 'version - date'. [reinout]

1.6 (2009-02-14)

- Patch by Michael Howitz: sys.executable is used instead of a string that doesn't work on every system. [reinout]

1.5 (2009-02-11)

- Changed y/n into Y/n, so defaulting to 'yes'. [reinout]
- Improved the documentation. [reinout]
- When a yes/no question is asked, do not treat 'no' as the default but explicitly ask for an input – it was too easy to press enter and wrongly expect 'yes' as default. [maurits]

1.4 (2008-10-23)

- Fixed missing import of utils. [maurits]

1.3 (2008-10-23)

- Moved stabilize script to zest.stabilizer so that zest.releaser is just for releasing individual packages. Nice, tidy, reusable. [reinout]
- Allowing '-v' option on all commands: it gives you debug-level logging. [reinout]

1.2 (2008-10-16)

- We now prefer the version from setup.py over any version.txt file found. When getting or changing the version we get/change the setup.py version when it differs from the found version.txt version. [maurits]

1.1 (2008-10-15)

- Cleaned out zest-specific stuff. Cleaned up 'release'. [reinout]

1.0 (2008-10-15)

- Stabilize looks up the most recent tag of our development packages and uses gp.svndevelop to allow svn checkouts as development eggs. [reinout]
- Do not look for version.txt in directories that are not handled by subversion. Use case: Products.feedfeeder, which has a buildout.cfg and so can have a parts directory with lots of version.txt files... [maurits]

0.9 (2008-10-02)

- release: offer to register and upload the egg to the cheese shop. After that you still have the option to upload to our own tgz server. [maurits]

- postrelease: for the suggestion of a new version simply try add 1 to the last character in the version; the previous logic failed for example for '1.0b3'. [maurits]
- prerelease: ask user to enter next version (give him a suggestion). Handy when you want to change '1.0b3 dev' into '1.0'. [maurits]
- Started 'stabilize'. [reinout]

0.8 (2008-09-26)

- fullrelease: change back to the original directory after each pre/actual/post release step. [maurits]
- release: switch back to original directory when ready to fix 'commit to tag' error. [maurits]
- prerelease: quit when no version is found. [maurits]
- Reverted sleeping fix from 0.7 as it did not work. [maurits]

0.7 (2008-09-26)

- fullrelease: hopefully fix a 'commit on tag' bug by sleeping three seconds before doing the post release. [maurits]

0.6 (2008-09-26)

- Added fullrelease script that does a prerelease, actual release and post release in one go. [maurits]

0.5 (2008-09-26)

- Factored part of prerelease.check_version() out into utils.cleanup_version(). We now use that while setting the version in the history during postrelease. [maurits]
- Add newline at the end of the generated version.txt. [maurits]

0.4 (2008-09-26)

- Made the logging userfriendly.

0.3 (2008-09-26)

- Postrelease: Better injection of history. Various other robustness fixes.

0.2 (2008-09-26)

- postrelease: added suggestion for new version (a plain enter is enough to accept it). [reinout]
- prerelease: ask before changing version + solidified regex for heading detection. [reinout]
- prerelease: detect non-development versions better and change them. [maurits]
- prerelease: made the commit message read: 'Preparing release xxx'. [maurits]
- postrelease: made the new version something like '1.0 dev'. [maurits]
- postrelease: we now add some lines to the history now. [maurits]
- prerelease: try changing the version to a non-development version, stripping off something like '(...)'. [maurits]
- release: Refactored so release.py has the 'main' function required by setup.py. [maurits]

0.1 (2008-09-24)

- Got a basic version of the prerelease script working (version check, history file updating, committing). [reinout]
- Started by copying the guidelines script. [reinout]