Zope 2 Documentation

Release 2.13

Zope Developers Community

Dec 17, 2021

Contents

Note: The Zope 2.13.x release series is in bug-fix-only mode. It will see security and bug fixes until further notice.

Contents:

CHAPTER 1

What's new in Zope 2.13

The article explains the new high-level features and changes found in this version of Zope 2.

You can have a look at the detailed change log to learn about all minor new features and bugs being solved in this release.

1.1 Python 2.7

This release of Zope 2 adds support for Python 2.7. Please refer to the What's new in Python 2.7 document, if you want to know more about the changes.

Zope 2.13 is continuing to support Python 2.6.4 or any later maintenance release of it. There's currently no support for any Python 3.x version. Work has begun in the Zope Toolkit to port some of the lower level packages to Python 3.

1.2 ZODB 3.10

This version of Zope includes ZODB 3.10 - a new major version of the ZODB. Among the notable changes are a variety of performance improvements. The ZEO server process is now multi-threaded. If the underlying file system and disk storage can handle concurrent disk I/O efficiently a throughput increase by a factor of up to four has been seen. On a related note using solid state disks for the ZEO server has a similar effect and can increase throughput by the same factor. Both of these effects combined can lead to an increase of up to sixteen times the throughput in high load scenarios.

File storage indexes use a new format, which is both smaller in size and can be read much faster. The repozo backup script now also backs up the index files in addition to the actual data, so in a restore scenario the index doesn't have to be recreated. For large databases this can bring down the total downtime in a restore scenario by a significant amount of time.

The ZODB has added support for wrapper storages that transform pickle data. Applications for this include compression and encryption. A storage using standard zlib compression is available as a new package called zc.zlibstorage. In content management scenarios where strings constitute the most of the non-blob data, this can reduce the Data.fs size by a factor of two or more. The overhead of compressing and uncompressing is negligible. This saves both network

I/O and disk space. More importantly the database has better chances of fitting into the operating systems disk cache and thus into memory. The second advantage is less important when using solid state disks.

Databases now warn when committing very large records (> 16MB). This is to try to warn people of likely design mistakes. There is a new option (large_record_size/large-record-size) to control the record size at which the warning is issued. This should help developers to better understand the storage implications of their code, which has been rather transparent so far.

The mkzeoinst script has been moved to a separate project zope.mkzeoinstance and is no-longer included with ZODB. You will need to use this new package to set up ZEO servers or use the plone.recipe.zeoserver recipe if you use buildout.

More information can be found in the detailed change log.

1.3 WSGI

See *Running Zope2 as a WSGI Application*.

This Zope release comes with native WSGI support. First pioneered in the repoze.zope2 project, this capability finally found its way back into the core and obsoletes the externally managed project. With WSGI Zope 2 can natively talk to a variety of web servers and isn't restricted to its own ZServer anymore. It also opens up new possibilities for writing or reusing middleware in Zope 2 or factoring out capabilities into WSGI endware. It's expected that this new deployment model will over time become the default and the old ZServer implementation will be deprecated. There's no concrete timeline for this yet.

Note: Due to the way logic is split out into WSGI middleware, some of the *ZPublisher.pubevents* aren't emitted by the WSGI publisher. These are: *PubSuccess*, *PubFailure*, *PubBeforeCommit* and *PubBeforeAbort*.

1.4 Zope Toolkit

Zope 2.13 has neither direct nor indirect `zope.app.*` dependencies anymore. This finishes the transition from the hybrid Zope 2 + 3 codebase. Zope 3 itself has been split up into two projects, the underlying Zope Toolkit consisting of foundation libraries and the application server part. The application server part has been renamed BlueBream. Zope 2 only depends and ships with the Zope Toolkit now.

Large parts of code inside Zope 2 and specifically Products.Five have been refactored to match this new reality. The goal is to finally remove the Five integration layer and make the Zope Toolkit a normal integral part of Zope 2.

1.5 ZCatalog

The ZCatalog and the default set of indexes as found in the PluginIndexes package have seen a large number of changes. Most of these have been pioneered in add-on packages in the Zope community over the last years and now have found their way back into the core. The largest change is added query plan support for the catalog. A standard feature in all relation databases, the job of a query plan is to monitor queries in a live system and based on execution metrics devise optimized plans for executing the low level instructions which lead to a query result. In sites with large number of indexed objects this can make a tremendous difference and significantly speed up all queries.

The query plan support is completely transparent to all users, though ways exist for developers to predefine it and store it across server restarts. The plan itself can be introspected in a tab in the ZMI. There's also a new ZMI tab to report slow catalog queries which can help developers to tune the remaining slow queries in their applications.

In addition to these larger changes there's been a high number of smaller changes to the search logic and the catalog implementations. All of these should result in better query execution and reduced number of conflict error potential.

1.6 Refactoring

There's an ongoing effort to refactor Zope 2 into more independent modularized distributions. Zope 2.12 has already seen a lot of this, with the use of zope.* packages as individual distributions and the extraction of packages like Acquisition, DateTime or tempstorage to name a few. Zope 2.13 continues this trend and has moved all packages containing C extensions to external distributions. Among those are AccessControl, DocumentTemplate and Products.ZCTextIndex.

1.7 Optional Formlib support

Zope 2 made a number of frameworks available through its integration layer Products.Five. Among these has been direct support for an automated form generation framework called zope.formlib with its accompanying widget library zope.app.form.

This form generation framework has seen only minor adoption throughout the Zope community and more popular alternatives like z3c.form exist. To reflect this status Zope 2 no longer directly contains formlib support.

If you rely on formlib, you need to add a dependency to the new five.formlib distribution and change all related imports pointing to Products.Five.form or Products.Five.formlib to point to the new package instead.

In order to ease the transition, five.formlib has been backported to the 2.12 release series. Starting in 2.12.3 you can already use the new five.formlib package, but backwards compatibility imports are left in place in Products.Five. This allows you to easily adapt your packages to work with both 2.12 and 2.13.

CHAPTER 2

Installing Zope with `zc.buildout`

This document describes how to get going with Zope using `zc.buildout`.

2.1 About `zc.buildout`

zc.buildout is a powerful tool for creating repeatable builds of a given software configuration and environment. The Zope developers use `zc.buildout` to develop Zope itself, as well as the underlying packages it uses.

2.2 Prerequisites

In order to use Zope, you must have the following pre-requisites available:

- A supported version of Python, including the development support if installed from system-level packages. Supported versions include:
  - 2.7.x
- Zope needs the Python `zlib` module to be importable. If you are building your own Python from source, please be sure that you have the headers installed which correspond to your system's `zlib`.
- A C compiler capable of building extension modules for your Python (gcc recommended). This is not necessary for Windows as binary releases of the parts that would need compiling are always made available.
- If you wish to install Zope as a Service on Windows, you will need to have the pywin32 package installed.

2.3 Installing Zope using zc.buildout

In this configuration, we use `zc.buildout` to install the Zope software. We are using a buildout recipe which does most of the configuration work.

Installing the Zope software using `zc.buildout` involves the following steps:

- Create a virtual environment where you install `zc.buildout`:

```
$ virtualenv --python 2.7 zope-2.13
$ cd zope-2.13
$ bin/pip install "zc.buildout<3"
```

- Write a minimal `buildout.cfg` and store it in the `zope-2.13` directory:

```
$ cat > buildout.cfg << EOF
[buildout]
extends = https://zopefoundation.github.io/Zope/releases/2.13.30/versions-prod.cfg
show-picked-versions = true
parts = zopectl

[versions]
plone.recipe.zope2instance = < 5
mailinglogger = < 6
zc.buildout =

[zopectl]
recipe = plone.recipe.zope2instance
http-address = 127.0.0.1:8080
eggs =
EOF
```

- Run `zc.buildout` to install Zope:

```
$ bin/buildout
```

- The buildout recipe `plone.recipe.zope2instance` has many more configuration options which can be leveraged in `buildout.cfg`, see https://pypi.org/project/plone.recipe.zope2instance/4.4.1/

After installation, refer to *Configuring and Running Zope* for documentation on configuring and running Zope.

2.4 Manually creating a buildout-based Zope instance

Rather than installing Zope using a recipe you can also do this by hand. This procedure involves the following steps:

- Create the home directory for the buildout, including `etc`, `log` and `var` subdirectories.
- Fetch the buildout bootstrap script into the environment.
- Fetch the version files into the environment, for example: https://raw.github.com/zopefoundation/Zope/2.13.30/versions.cfg https://raw.github.com/zopefoundation/Zope/2.13.30/ztk-versions.cfg
- Create a buildout configuration as follows:

buildout.cfg

```
[buildout]
parts = instance
extends = versions.cfg

[instance]
recipe = zc.recipe.egg
eggs = Zope2
interpreter = py
scripts = runzope zopectl
initialization =
  import sys
  sys.argv[1:1] = ['-C',r'${buildout:directory}/etc/zope.conf']
```

This is the minimum but all the usual buildout techniques can be used.

- Bootstrap the buildout
- Run the buildout
- Create a Zope configuration file. A minimal version would be:

etc/zope.cfg

```
%define INSTANCE <path to your instance directory>

python $INSTANCE/bin/py[.exe on Windows]

instancehome $INSTANCE
```

A fully-annotated sample can be found in the Zope2 egg:

```
$ cat eggs/Zope2--*/Zope2/utilities/skel/etc/zope.conf.in

<rest of the stuff that goes into a zope.conf, e.g. databases and log files.>
```

An example session:

```
$ mkdir /path/to/instance
$ cd /path/to/instance
$ mkdir etc logs var
$ wget https://svn.zope.org/zc.buildout/trunk/bootstrap/bootstrap.py
$ vi buildout.cfg
$ /path/to/your/python bootstrap.py  # assumes no setuptools installed
$ bin/buildout
$ cat eggs/Zope2--*/Zope2/utilities/skel/etc/zope.conf.in > etc/zope.conf
$ vi etc/zope.conf  # replace <<INSTANCE_HOME>> with buildout directory
$ bin/zopectl start
```

In the `bin` subdirectory of your instance directory, you will find `runzope` and `zopectl` scripts that can be used as normal.

You can use `zopectl` interactively as a command shell by just calling it without any arguments. Try `help` there and `help <command>` to find out about additionally commands of zopectl. These commands also work at the command line.

After installation, refer to *Configuring and Running Zope* for documentation on configuring and running Zope.

CHAPTER 3

Installing Zope with `virtualenv`

This document describes how to install Zope into a `virtualenv`.

3.1 Create a Virtual Environment

```
$ /opt/Python-2.7.14/bin/virtualenv z213
New python executable in z213/bin/python
Installing setuptools, pip, wheel...done.
$ cd z213
```

3.2 Install the Zope2 2.13.29 Software Packages

```
$ bin/pip install \
 --no-binary zc.recipe.egg \
 -r https://zopefoundation.github.io/Zope/releases/2.13.29/requirements.txt
Collecting Zope2
...
Successfully installed ...
```

3.3 Creating a Zope instance

Once you've installed Zope, you will need to create an "instance home". This is a directory that contains configuration and data for a Zope server process. The instance home is created using the `mkzopeinstance` script:

```
$ bin/mkzopeinstance
```

You can specify the Python interpreter to use for the instance explicitly:

```
$ bin/mkzopeinstance --python=bin/python
```

You will be asked to provide a user name and password for an administrator's account during `mkzopeinstance`. To see the available command-line options, run the script with the `--help` option:

```
$ bin/mkzopeinstance --help
```

3.4 Using the `virtualenv` as the Zope Instance

You can choose to use the `virtualenv` as your Zope instance:

```
$ bin/mkzopeinstance -d .
```

In this case, the instance files will be located in the subdirectories of the `virtualenv`:

- `etc/` will hold the configuration files.
- `log/` will hold the log files.
- `var/` will hold the database files.

CHAPTER 4

Configuring and Running Zope

Whichever method you used to install Zope and create a server instance (see *Installing Zope with zc.buildout* and *Installing Zope with virtualenv*), the end result is configured and operated the same way.

4.1 Configuring Zope

Your instance's configuration is defined in its `etc/zope.conf` file. Unless you created the file manually, that file should contain fully- annotated examples of each directive.

You can also pass an explicit configuration file on the command line:

```
$ /path/to/zope/instance/bin/zopectl -c /tmp/other.conf show
...
Config file:  /tmp/other.conf
```

When starting Zope, if you see errors indicating that an address is in use, then you may have to change the ports Zope uses for HTTP or FTP. The default HTTP and FTP ports used by Zope are 8080 and 8021 respectively. You can change the ports used by editing ./etc/zope.conf appropriately.

The section in the configuration file looks like this:

```
<http-server>
  # valid keys are "address" and "force-connection-close"
  address 8080
  # force-connection-close on
</http-server>
```

The address can just be a port number as shown, or a host:port pair to bind only to a specific interface.

After making any changes to the configuration file, you need to restart any running Zope server for the affected instance before changes are in effect.

4.2 Running Zope in the Foreground

To run Zope without detaching from the console, use the `fg` command (short for `foreground`):

```
$ /path/to/zope/instance/bin/zopectl fg
```

In this mode, Zope emits its log messages to the console, and does not detach from the terminal. This also automatically enables debug-mode. Do not use this for production servers.

4.3 Running Zope as a Daemon

Once an instance home has been created, the Zope server can now be started using this command:

```
$ /path/to/zope/instance/bin/zopectl start
```

During startup, Zope emits log messages into */path/to/zope/instance/log/event.log*. You can examine it with the usual tools (`cat`, `more`, `tail`, etc) and see if there are any errors preventing Zope from starting.

Note: For this to work on Windows, the Zope instance must be installed as a Service. This is done with:

```
bin\zopectl install
```

If you later want to remove this Service, do the following:

```
bin\zopectl remove
```

For the full list of options available for setting up Zope as a Windows Service, do:

```
bin\zopectl install --help
```

4.4 Integrating with System Startup

zopectl can be linked as rc-script in the usual start directories on linux or other System V unix variants.

You can use `zopectl` interactively as a command shell by just calling it without any arguments. Try `help` there and `help <command>` to find out about additionally commands of zopectl. These commands also work at the command line.

Note: On Windows, a Service can be installed and set to start automatically with the following:

```
bin\zopectl install --startup=auto
```

4.5 Logging In To Zope

Once you've started Zope, you can then connect to the Zope webserver by directing your browser to:

```
http://yourhost:8080/manage
```

where 'yourhost' is the DNS name or IP address of the machine running Zope. If you changed the HTTP port as described, use the port you configured.

You will be prompted for a user name and password. Use the user name and password you provided in response to the prompts issued during the "make instance" process.

Now you're off and running! You should be looking at the Zope management screen which is divided into two frames. On the left you can navigate between Zope objects and on the right you can edit them by selecting different management functions with the tabs at the top of the frame.

If you haven't used Zope before, you should head to the Zope web site and read some documentation. The Zope Documentation section is a good place to start. You can access it at http://docs.zope.org/

4.6 Troubleshooting

- This version of Zope requires Python 2.6.4 or better. It will *not* run with Python 3.x.
- The Python you run Zope with *must* have threads compiled in, which is the case for a vanilla build. Warning: Zope will not run with a Python version that uses `libpth`. You *must* use `libpthread`.
- To build Python extensions you need to have Python configuration information available. If your Python comes from an RPM you may need the python-devel (or python-dev) package installed too. If you built Python from source all the configuration information should already be available.
- See the *Changelog* for important notes on this version of Zope.

4.7 Adding extra commands to Zope

It is possible to add extra commands to `zopectl` by defining *entry points* in `setup.py`. Commands have to be put in the `zopectl.command` group:

```
setup(name="MyPackage",
      ....
      entry_points="""
      [zopectl.command]
      init_app = mypackage.commands:init_application
      """)
```

Note: Due to an implementation detail of `zopectl` you can not use a minus character (-) in the command name.

This adds a `init_app` command that can be used directly from the command line:

```
bin\zopectl init_app
```

The command must be implemented as a Python callable. It will be called with two parameters: the Zope2 application and a list with all command line arguments. Here is a basic example:

```
def init_application(app, args):
    print 'Initializing the application'
```

Make sure the callable can be imported without side-effects, such as setting up the database connection used by Zope 2.

CHAPTER 5

Running Zope2 as a WSGI Application

This document assumes you have installed Zope into a `virtualenv` (see *Installing Zope with virtualenv*).

5.1 Install the Supporting Software

To run as a WSGI application, you need to install some additional software.

```
$ bin/pip install \
 --no-binary zc.recipe.egg \
 -r https://zopefoundation.github.io/Zope/releases/2.13.29/requirements.txt \
 repoze.who repoze.tm2 repoze.retry Paste PasteDeploy PasteScript
Collecting repoze.who
...
Successfully installed ...
```

5.2 Update the Zope Application Configuration

The generated `etc/zope.conf` file assumes that Zope will be running using the built-in `ZServer`.

```
$ vim etc/zope.conf
```

Update the contents as follows.

```
%define INSTANCE /path/to/virtualenv
instancehome $INSTANCE
```

Note: The `%define instance /path/to/virtualenv` element must point to the environment: there is no "relative to this file" support built in.

Set up logging for the application.

```
<eventlog>
  level info
  <logfile>
    path $INSTANCE/log/event.log
    level info
  </logfile>
</eventlog>

<logger access>
  level WARN
  <logfile>
    path $INSTANCE/log/Z2.log
    format %(message)s
  </logfile>
</logger>
```

Configure the database (note that you could use `ZEO` or `Relstorage` rather than a bare `FileStorage`):

```
<zodb_db main>
    # Main FileStorage database
    <filestorage>
      # See .../ZODB/component.xml for directives (sectiontype
      # "filestorage").
      path $INSTANCE/var/Data.fs
    </filestorage>
    mount-point /
</zodb_db>

<zodb_db temporary>
    # Temporary storage database (for sessions)
    <temporarystorage>
      name temporary storage for sessioning
    </temporarystorage>
    mount-point /temp_folder
    container-class Products.TemporaryFolder.TemporaryContainer
</zodb_db>
```

Because we will be running a separately-configured WSGI server, remove any `<http-server>` configuration from the file.

5.3 Create the WSGI Server Configuration

```
$ vim etc/zope.wsgi
```

First, configure the “application” endpoint for Zope:

```
[app:zope]
use = egg:Zope2#main
zope_conf = %(here)s/zope.conf
```

Next, set up the WSGI middleware pipeline:

```
[pipeline:main]
pipeline =
    egg:paste#evalerror
    egg:repoze.retry#retry
    egg:repoze.tm2#tm
    zope
```

The middleware layers are "wrapped" around the application endpoint as follows:

- `paste#evalerror` is debugging middleware, which shows tracebacks for errors raised from the application. It should **not** be configured for production use.
- `repoze.retry#retry` is middleware which retries requests when retriable exceptions are raised. By default, it retries 3 times, and only for requests which raise `ZODB.ConflictError`. See http://repozeretry.rtfd.org/ for details on configuring it otherwise.
- `repoze.tm2#tm` is middleware which begins a new transaction for each request, and then either aborts the transaction (if the request raises an exception) or commits it (if not). See http://repozetm2.rtfd.org/ for details on configuring it.

Finally, configure the WSGI server:

```
[server:main]
use = egg:paste#http
host = localhost
port = 8080
```

Note: Any server conforming to PEP 333/3333 should work, although the parameters could change.

5.4 Set up the Admin User

Before starting the WSGI server, run the `addzope2user` script to configure the administrative user.

```
$ bin/addzope2user admin <yourpasswordhere>
No handlers could be found for logger "ZODB.FileStorage"
User admin created.
```

5.5 Start the WSGI Server

```
$ bin/paster serve etc/zope.wsgi
Starting server in PID 24934.
serving on http://127.0.0.1:8080
```

5.6 Running Other Applications in the same WSGI Server Process

You can use any of the normal `Paste` WSGI features to combine Zope and other WSGI applications inside the same server process. E.g., the following configuration uses the composite application support offered by `PasteDeploy` to host Zope at the `/` prefix, with static files served from disk at `/static`:

```
[app:zope-app]
use = egg:Zope2#main
zope_conf = %(here)s/zope.conf

[pipeline:zope-pipeline]
pipeline =
    egg:paste#evalerror
    egg:repoze.retry#retry
    egg:repoze.tm2#tm
    zope-app

[app:static]
use = egg:Paste#static
document_root = %(here)s/static

[composite:main]
use = egg:Paste#urlmap
/ = zope-pipeline
/static = static
```

CHAPTER 6

Special Users

Because Zope is managed through the web, user names and passwords must be used to assure that only authorized people can make changes to a Zope installation.

6.1 Adding Managers

If you need to add a Manager to an existing Zope instance, you can do this using *zopectl* as follows:

```
zopectl adduser `name` `password`
```

6.2 The Initial User

An initial username and password is needed to "bootstrap" the creation of normal managers of your Zope site. This is accomplished through the use of the 'inituser' file in the directory specified as the instance home.

The first time Zope starts, it will detect that no users have been defined in the root user folder. It will search for the 'inituser' file and, if it exists, will add the user defined in the file to the root user folder.

Normally, 'inituser' is created by the Zope install scripts. Either the installer prompts for the password or a randomly generated password is created and displayed at the end of the build script.

You can use the 'zpasswd.py' script to create 'inituser' yourself. Execute 'zpasswd.py' like this:

```
python zpasswd.py inituser
```

The script will prompt you for the name, password, and allowed domains. The default is to encode the password with SHA, so please remember this password as there is no way to recover it (although 'zpasswd.py' lets you reset it.)

6.3 The Emergency User

In some situations you may need to bypass normal security controls because you have lost your password or because the security settings have been mixed up. Zope provides a facility called an "emergency user" so that you can reset passwords and correct security settings.

The emergency user password must be defined outside the application user interface. It is defined in the 'access' file located in the Zope directory. It should be readable only by the user as which your web server runs.

To create the emergency user, use 'zpasswd.py' to create the 'access' file like this:

```
python zpasswd.py access
```

In order to provide a somewhat higher level of security, various encoding schemes are supported which provide access to either SHA-1 encryption or the standard UNIX crypt facility if it has been compiled into Python. Unless you have some special requirements (see below), you should use the SHA-1 facility, which is the default.

6.4 Format of 'inituser' and 'access'

A password file should consist of a single line of the form:

```
name:password
```

Note that you may also add an optional third component to the line in the access file to restrict access by domain. For example, the line:

```
mario:nintendoRules:*.mydomain.com
```

in your 'access' file will only allow permit emergency user access from **.mydomain.com* machines. Attempts to access the system from other domains will fail, even if the correct emergency user name and password are used.

Please note that if you use the ZServer monitor capability, you will need to run with a clear text password.

CHAPTER 7

Filesytem Permissions

You need to set permissions on the directory Zope uses to store its data. This will normally be the *var* directory in the instance home. Zope needs to read and write data to this directory. Before running Zope you should ensure that you give adequate permissions to this directory for the userid Zope will run under.

Depending on how you choose to run Zope you will need to give different permissions to the directory. If you use Zope with an existing web server, it will probably run Zope as 'nobody'. In this case 'nobody' needs read and write permissions to the var directory.

If you change the way you run Zope, you may need to modify the permissions of the directory and the files in it to allow Zope to read and write under its changed userid.

CHAPTER 8

Zope effective user support

Note: It is best practice to run Zope behind a reverse proxy like Apache, Squid or Varnish. In this case, you do not need to run or install Zope with root privileges, since the reverse proxy will bind to port 80 and proxy back all request to Zope running on an unprivileged port.

Zope can bind its network service to low ports such as 21 (FTP) and 80 (HTTP). In order to bind to low ports, Zope must be started as the root user. However, Zope will only run as root long enough to bind to these low ports. It will then attempt to setuid to a less privileged user.

You must specify the user to which Zope will attempt to setuid by changing the 'effective-user' parameter in the zope.conf configuration file to an existing username or UID. All runtime files will be written as this user. If you do not specify an 'effective-user' in the configuration file, and you attempt to start Zope, it will refuse to start.

Zope additionally emits a warning if you specify 'nobody' as the 'effective-user'. The rationale for this warning stems from the fact that, historically, many other UNIX services dropped privileges to the 'nobody' account after starting as root. Any security defects in these services could cause someone to gain access as the 'nobody' account on your system. If someone was to gain control of your 'nobody' account they could compromise your Zope files.

The most important thing to remember about effective user support is that you don't have to start Zope as root unless you want to listen for requests on low ports (ports beneath 1024). In fact, if you don't have this need, you are much better off just starting Zope under a dedicated user account.

CHAPTER 9

Signals (POSIX only)

Signals are a POSIX inter-process communications mechanism. If you are using Windows then this documentation does not apply.

Zope responds to signals which are sent to the process id specified in the file '$INSTANCE_HOME/var/Z2.pid':

```
SIGHUP  - close open database connections, then restart the server
          process. A idiom for restarting a Zope server is:

          kill -HUP `cat $INSTANCE_HOME/var/z2.pid`

SIGTERM - close open database connections then shut down. A common
          idiom for shutting down Zope is:

          kill -TERM `cat $INSTANCE_HOME/var/Z2.pid`

SIGINT  - same as SIGTERM

SIGUSR1 - dump a stack trace of all threads to stdout. This can help
          diagnosing `stuck` Zope processes if all threads are stuck.

SIGUSR2 - close and re-open all Zope log files (z2.log, event log,
          detailed log.) A common idiom after rotating Zope log files
          is:

          kill -USR2 `cat $INSTANCE_HOME/var/z2.pid`
```

CHAPTER 10

Running Zope in Debug Mode

A utility known as ‘zopectl’ is installed into generated instance homes.

If you wish to run Zope in debug mode, run zopectl in foreground mode:

```
$ bin/zopectl fg
```

You can also use it to inspect a Zope instance’s running state via an interactive Python interpreter by passing zopectl the ‘debug’ parameter on the command line. The ‘top-level’ Zope object (the root folder) will be bound to the name ‘app’ within the interpreter. You can then use normal Python method calls against app and use the Python interpreter normally to inspect results:

```
$ bin/zopectl debug
Starting debugger (the name "app" is bound to the top-level Zope object)
>>> app.keys()
['acl_users', 'Control_Panel', 'temp_folder', 'browser_id_manager', 'session_data_
↪manager', 'error_log', 'index_html', 'standard_error_message']
>>>
```

CHAPTER 11

Changelog

This file contains change information for the current Zope release. Change information for previous versions of Zope can be found at http://docs.zope.org/zope2/

11.1 2.13.31 (unreleased)

- TBD

11.2 2.13.30 (2020-02-14)

11.2.1 Security related fixes

- Prevent header spoofing by excluding those with _ in them (#655)
- Fix a possible SQL injection in DTML or in connection objects. The fix include the changes provided by PloneHotfix20200121-1.1.zip by updating to use DocumentTemplate version 2.13.6. (For details see https://plone.org/security/announcements/new-waitress-version-and-updated-20200121-hotfix.)

11.3 2.13.29 (2019-02-09)

11.3.1 Security related fixes

- `HTTPRequest.text()` now obscures values of fields those name contain the string `passw` in the same way `HTTPRequest.__str__` already did. (#375)

11.3.2 Backwards incompatible changes

- Drop support for Python 2.6. This means it is no longer tested from now on. (#475)

11.3.3 Features

- Add support for IPv6 hosts in VirtualHostMonster. (#395)
- Enable ZMI History tab for `OFS.Image.File`. (#396)

11.4 2.13.28 (2018-04-23)

- Add `OFS.CopySupport.CopyContainer._pasteObjects()` to be able to paste objects no matter how many objects where cut or copied. (#217)

11.5 2.13.27 (2018-01-27)

- Test that `str.format` checks security for accessed keys and items. The real fix is in the AccessControl package. Part of PloneHotfix20171128.
- Made Redirect unavailable as URL. Part of PloneHotfix20171128.
- Skip IPv6 tests on Travis, as it is not supported.
- Add `tox` test configuration.
- Set explicit PyPI index URL, the old `zc.buildout` defaults no longer work.
- Pin `pytz` to prevent unit test failures from `DateTime`.
- Fix virtualenv based installation docs.
- Explicitly require Manager role for `AltDatabaseManager`. [maurits]

11.6 2.13.26 (2017-02-20)

- In `str.format`, check the security for attributes that are accessed. Part of PloneHotfix20170117. [maurits]
- Fixed reflective XSS in findResult. This applies PloneHotfix20170117. [maurits]

11.7 2.13.25 (2017-01-13)

- Add a dependency on the empty *ZServer* project.
- Patch zope.interface to remove docstrings and avoid publishing. From Products.PloneHotfix20161129. [maurits]
- Don't copy items the user is not allowed to view. From Products.PloneHotfix20161129. [maurits]
- Quote variables in manage_tabs and manage_container to avoid XSS. From Products.PloneHotfix20160830. [maurits]
- Add a dependency on the empty *Products.TemporaryFolder* project.

- Add a dependency on the empty *Products.Sessions* project.
- Removed docstrings from some methods to avoid publishing them. From Products.PloneHotfix20160419. [maurits]
- Add support to SameSite cookie in `ZPublisher.HTTPResponse`: https://tools.ietf.org/html/draft-west-first-party-cookies-07

11.8 2.13.24 (2016-02-29)

- Issue #44: Ensure that iterators declared as implementing `IUnboundStreamIterator` are handled properly.
- Issue #43: Fix Zope failing to start if a zoperunner is configured.
- PR #51: Harden debug control panel's module-crawling against trickery performed by `six`.
- Issue #34: Fix `NameError` exception for `WindowsError` which could happen on non-windows systems.
- Updated distributions:
  - AccessControl = 2.13.14

11.9 2.13.23 (2015-06-29)

- Provide a pip-compatible `requirements.txt` file for the release. E.g.:

```
$ /path/to/venv/bin/pip install -r \
  https://raw.githubusercontent.com/zopefoundation/Zope/2.13.23/requirements.txt
```

- LP #789863: Ensure that Request objects cannot be published / traversed directly via a URL.
- Issue #27: Fix publishing of `ZPublisher.Iterators.IStreamIterator` under WSGI. This interface does not have `seek` or `tell`. Introduce `ZPublisher.Iterators.IUnboundStreamIterator` to support publishing iterators of unknown length under WSGI.
- Document running Zope as a WSGI application. See https://github.com/zopefoundation/Zope/issues/30
- LP #1465432: Ensure that WSGIPublisher starts / ends interaction at request boundaries (analogous to ZPublisher). Backport from master.
- Fix: Queue additional warning filters at the beginning of the queue in order to allow overrides.
- Issue #16: prevent leaked connections when broken `EndRequestEvent` subscribers raise exceptions.
- LP #1387225: Zope 2.13.x w/ zope.browserpage 4.x doesn't start.
- LP #1387138: Zope 2.13.x w/ zope.pagetemplate 4.x doesn't start.
- LP #1386795: Fix `zopectl start` with zdaemon 3 and newer.
- Updated distributions:
  - Acquisition = 2.13.9
  - DateTime = 2.12.8
  - Products.BTreeFolder2 = 2.13.5
  - Products.ExternalMethod = 2.13.1

  - Products.Mailhost = 2.13.2
  - Products.StandardCacheManagers = 2.13.1
  - ZConfig = 2.9.3
  - zLOG = 2.11.2
  - zope.dublincore = 3.7.1
  - zope.mkzeoinstance = 3.9.6

11.10 2.13.22 (2014-02-19)

- Merge hotfixes from https://pypi.python.org/pypi/Products.PloneHotfix20131210
- LP #143352: Logging of client IP rather than the IP of the Proxy. Please be aware that this only logs the real client ips to Z2.log, if you set you proxy as a trusted-proxy in zope.conf.
- Updated distributions:
  - Products.ZCatalog = 2.13.27
  - Products.ZCTextIndex = 2.13.5

11.11 2.13.21 (2013-07-16)

- LP #1095343: Prevent sandbox escape via `BaseRequest.traverseName`.
- LP #1094144: Prevent arbitrary redirections via faked "CANCEL" buttons.
- LP #1094221: Add permissions to some unprotected methods of `OFS.ObjectManager`.
- LP #1094049: Prevent zlib-based DoS when parsing the cookie containing paste tokens.
- Updated distributions:
  - AccessControl = 2.13.13

11.12 2.13.20 (2013-05-01)

- LP #1114688: Defend against minidom-based DoS in webdav. (Patch from Christian Heimes).
- LP #978980: Protect views of ZPT source with 'View Management Screens' permision.
- Make sure the generated classes for simple browser pages (SimpleViewClasses) have a str `__name__`. See LP #1129030.
- In PageTemplate.pt_errors accept the check_macro_expansion argument. This is added for compatibility with zope.pagetemplate 4.0.0. The argument is ignored. See LP #732972.
- Updated to Zope Toolkit 1.0.8.
- Updated distributions:
  - Products.ZCTextIndex = 2.13.4
  - ZConfig = 2.9.1

11.13 2.13.19 (2012-10-31)

- Updated distributions:
  - AccessControl = 2.13.12
  - distribute = 0.6.29
  - mr.developer = 1.22
  - pytz = 2012g
  - repoze.retry = 1.2
  - repoze.tm2 = 1.0
  - tempstorage = 2.12.2
- LP #1071067: Use a stronger random number generator and a constant time comparison function.
- LP #1061247: Fix ZMI properties edit form for properties named *method*.
- LP #1058049: Fix support for zoperunner section in zope.conf.
- Explicitly close all databases on shutdown, which ensures *Data.fs.index* gets written to the file system.
- LP #930812: Scrub headers a bit more.
- Fix lock and pid file handling on Windows. On other platforms starting Zope tolerated existing or locked files, this now also works on Windows.

11.14 2.13.18 (2012-09-18)

- Explicitly declared ZTUtils APIs as public (repairs breakages in apps following fix for LP #1047318).

11.15 2.13.17 (2012-09-09)

- Updated distributions:
  - AccessControl = 2.13.10
  - Products.PythonScripts = 2.13.2

11.16 2.13.16 (2012-08-11)

- Updated distributions:
  - AccessControl = 2.13.8
  - DateTime = 2.12.7
- OFS: Fixed TypeError handling in unrestrictedTraverse.
- ZPublisher: Do not assume that you can iterate over a publishable object.
- ZPublisher: Do not guess it is a webdav request if the HTTP method is purge.

11.17 2.13.15 (2012-06-22)

- Fix lock file cleanup if there's an error early in startup.
- Updated distributions:
  - zdaemon = 2.0.7

11.18 2.13.14 (2012-05-31)

- LP #950689: Fix HTTPS detection under mod_wsgi.
- LP #975039: Don't translate interface names in edit_markers ZMI view.
- LP #838978: Fixed TypeError in cache_detail ZMI view.
- Cleanup lock and pid files if the process dies early in startup.
- Added PubStart, PubBeforeCommit and PubAfterTraversal events to the WSGI publisher.
- ZPublisher: Fixed a traversal regression introduced in 2.13.12.
- Updated to Zope Toolkit 1.0.7.
- Updated distributions:
  - Products.ZCatalog = 2.13.23

11.19 2.13.13 (2012-02-20)

- LP #933307: Fixed ++skin++ namespace handling. Ported the `shiftNameToApplication` implementation from zope.publisher to ZPublisher.HTTPRequest.HTTPRequest.
- Ensure that the `WSGIPublisher` begins and ends an *interaction* at the request/response barrier. This is required for instance for the `checkPermission` call to function without an explicit `interaction` parameter.
- Ensure that ObjectManager's `get` and `__getitem__` methods return only "items" (no attributes / methods from the class or from acquisition). Thanks to Richard Mitchell at Netsight for the report.
- Updated to Zope Toolkit 1.0.6.
- Removed HTML tags from exception text of `Unauthorized` exception because these tags get escaped since CVE-2010-1104 (see 2.13.12) got fixed.

11.20 2.13.12 (2012-01-18)

- Prevent a cross-site-scripting attack against the default standard error message handling. (CVE-2010-1104).
- Use `in` operator instead of deprecated `has_key` method (which is not implemented by `OFS.ObjectManager`). This fixes an issue with WebDAV requests for skin objects.
- Updated distributions:
  - Products.ZCatalog = 2.13.22

11.21 2.13.11 (2011-12-12)

- LP #1079238: Turn *UndoSupport.get_request_var_or_attr* helper into a private API.
- LP #902068: Fixed missing security declaration for *ObjectManager* class.
- Avoid conflicting signal registrations when run under mod_wsgi. Allows the use of *WSGIRestrictSignal Off* (LP #681853).
- Make it possible to use WSGI without repoze.who.
- Fixed serious authentication vulnerability in stock configuration.
- Updated distributions:
  - AccessControl = 2.13.7
  - DocumentTemplate = 2.13.2
  - Products.BTreeFolder2 = 2.13.4
  - python-gettext = 1.2
  - repoze.who = 2.0
  - ZODB3 = 3.10.5
  - Zope Toolkit 1.0.5

11.22 2.13.10 (2011-10-04)

- Fixed serious arbitrary code execution issue (CVE 2011-3587) http://zope2.zope.org/news/security-vulnerability-announcement-cve-2011-3587
- Fixed a regression of 2.13.9 in webdav support that broke external editor feature.
- *undoMultiple* was still broken as transactions were not undone in the proper order : tids were stored and retrieved as dictionary keys.
- Updated distributions:
  - Products.ZCatalog = 2.13.20

11.23 2.13.9 (2011-08-20)

11.23.1 Bugs Fixed

- Restore ability to undo multiple transactions from the ZMI by using the *undoMultiple* API. Backported from trunk (r122087).
- Fixed Chameleon compatibility in templates.
- Updated distributions:
  - Products.ZCatalog = 2.13.19
  - Products.ZCTextIndex = 2.13.3
  - repoze.tm2 = 1.0b2
  - Zope Toolkit 1.0.4

11.24 2.13.8 (2011-06-28)

11.24.1 Bugs Fixed

- Fixed a serious privilege escalation issue. For more information see: http://plone.org/products/plone/security/advisories/20110622
- Ensure __name__ is not None as well as __name__ existing. For example, object could be a widget within a z3c.form MultiWidget, which do not have __name__ set.
- Testing: Re-added 'extra' argument to Functional.publish. Removing it in Zope 2.13.0a1 did break backwards compatibility.
- LP #787541: Fix WSGIPublisher to close requests on abort unconditionally. Previously an addAfterCommitHook was used, but this is not run on transaction aborts. Now a Synchronizer is used which unconditionally closes the request after a transaction is finished.

11.24.2 Features Added

- Updated distributions:
  - Acquisition = 2.13.8
  - Products.ZCatalog = 2.13.14
  - repoze.who = 2.0b1
  - ZODB3 = 3.10.3
  - Zope Toolkit 1.0.3

11.25 2.13.7 (2011-05-08)

11.25.1 Features Added

- Added forward compatibility with DateTime 3.
- ZPublisher: HTTPResponse.appendHeader now keeps header values to a single line by default to avoid causing problems for proxy servers which do not correctly handle multi-line headers.
- Updated distributions:
  - Products.ZCatalog = 2.13.13
  - Products.ZCTextIndex = 2.13.2

11.26 2.13.6 (2011-04-03)

11.26.1 Bugs Fixed

- Fix *WSGIResponse* and *publish_module* functions such that they support the *IStreamIterator* interface in addition to *file* (as supported by *ZServer.HTTPResponse*).
- Corrected copyright information shown in the ZMI.

- OFS: Fixed editing offset-naive 'date' properties in the ZMI. The "Properties" tab no longer shows the time zone of offset-naive dates.

11.26.2 Features Added

- Add preliminary IPv6 support to ZServer.
- Updated to Zope Toolkit 1.0.2.
- Updated distributions:
  - Acquisition = 2.13.7
  - mechanize = 0.2.5
  - Products.BTreeFolder2 = 2.13.3
  - Products.ZCatalog = 2.13.8
  - python-gettext = 1.1.1
  - pytz = 2011e
  - repoze.tm2 = 1.0b1
  - repoze.who = 2.0a4
  - ZConfig = 2.9.0
  - zope.testbrowser = 3.11.1

11.27 2.13.5 (2011-02-23)

11.27.1 Bugs Fixed

- Five: Corrected a method name in the IReadInterface interface.

11.27.2 Features Added

- Updated distributions:
  - Acquisition = 2.13.6
  - Products.ZCatalog = 2.13.6
  - ZODB3 = 3.10.2

11.28 2.13.4 (2011-02-06)

11.28.1 Bugs Fixed

- Applied missing bit of the code merge for LP #713253.

11.29 2.13.3 (2011-02-06)

11.29.1 Features Added

- Updated distributions:
  - Products.ZCatalog = 2.13.5

11.29.2 Bugs Fixed

- LP #713253: Prevent publication of acquired attributes, where the acquired object does not have a docstring.

11.30 2.13.2 (2011-01-19)

11.30.1 Bugs Fixed

- HelpSys: Fixed some permission checks.
- OFS: Fixed permission check in ObjectManager.
- webdav: Fixed permission check and error handling in DeleteCollection.
- LP 686664: WebDAV Lock Manager ZMI view wasn't accessible.

11.30.2 Features Added

- Report success or failure (when known) of creating a new user with the *addzope2user* script.
- Added *addzope2user* script, suitable for adding an admin user directly to the root acl_users folder.
- Updated distributions:
  - AccessControl = 2.13.4
  - Products.ZCatalog = 2.13.3

11.30.3 Restructuring

- Factored out the *Products.ZCatalog* and *Products.PluginIndexes* packages into a new *Products.ZCatalog* distribution.

11.31 2.13.1 (2010-12-07)

11.31.1 Bugs Fixed

- Fixed argument parsing for entrypoint based zopectl commands.
- Fixed the usage of `pstats.Stats()` output stream. The *Control_Panel/DebugInfo/manage_profile* ZMI view was broken in Python 2.5+.

11.31.2 Features Added

- Report success or failure (when known) of creating a new user with the addzope2user script.
- Moved subset id calculation in *OFS.OrderSupport.moveObjectsByDelta* to a new helper method, patch by Tom Gross.
- Updated to Zope Toolkit 1.0.1.
- Use cProfile where possible for the *Control_Panel/DebugInfo/manage_profile* ZMI view.

11.31.3 Restructuring

- Stopped testing non-overridden ZTK eggs in `bin/alltests`.

11.32 2.13.0 (2010-11-05)

- No changes.

11.33 2.13.0c1 (2010-10-28)

11.33.1 Bugs Fixed

- LP #628448: Fix `zopectl start` on non-Windows platforms.

11.33.2 Features Added

- Updated to Zope Toolkit 1.0.
- Updated distributions:
  - DateTime = 2.12.6
  - mechanize = 0.2.3
  - ZODB3 = 3.10.1
  - zope.sendmail = 3.7.4
  - zope.testbrowser = 3.10.3

11.34 2.13.0b1 (2010-10-09)

11.34.1 Bugs Fixed

- Avoid iterating over the list of packages to initialize while it is being mutated, which was skipping some packages.
- Fixed two unit tests that failed on fast Windows machines.
- Fixed OverflowError in Products.ZCatalog.Lazy on 64bit Python on Windows.
- Fixed `testZODBCompat` tests in ZopeTestCase to match modern ZODB semantics.

- LP #634942: Only require `nt_svcutils` on Windows.

11.34.2 Features Added

- Avoid conflict error hotspot in PluginIndexes' Unindex class by using IITreeSets instead of simple ints from the start. Idea taken from `enfold.fixes`.
- Added date range index improvements from `experimental.catalogqueryplan`.
- Changed policy on handling exceptions during ZCML parsing in `Products`. We no longer catch any exceptions in non-debug mode.
- Added a new BooleanIndex to the standard PluginIndexes.
- Update to Zope Toolkit 1.0c3.
- Add ability to define extra zopectl commands via setuptools entrypoints.
- Updated distributions:
  - Acquisition = 2.13.5
  - Products.MailHost = 2.13.1
  - Products.ZCTextIndex = 2.13.1
  - repoze.retry = 1.0
  - tempstorage = 2.12.1
  - ZODB3 = 3.10.0
  - zope.testbrowser = 3.10.1

11.35 2.13.0a4 (2010-09-09)

11.35.1 Restructuring

- Removed deprecated `Products.Five.security.create_permission_from_permission_directive` event handler. Its code was moved into the Zope 2 version of the permission directive in `AccessControl.security`.

11.35.2 Features Added

- LP #193122: New method getVirtualRoot added to the Request class.
- Updated test assertions to use unittest's `assert*` methods in favor of their deprecated *fail** aliases.
- Update to Zope Toolkit 1.0a3.
- Updated distributions:
  - AccessControl = 2.13.3
  - Acquisition = 2.13.4
  - ZODB3 = 3.10.0b6

11.36 2.13.0a3 (2010-08-04)

11.36.1 Bugs Fixed

- Adjusted overflow logic in DateIndex and DateRangeIndex to work with latest ZODB 3.10.0b4.
- Made sure to exclude a number of meta ZCML handlers from `zope.*` packages where Zope2 provides its own implementations.
- LP #599378: Fixed accumulated_headers not appending to headers correctly.
- Fix support for non-public permission attributes in the browser:view directive so that attributes which are not included in allowed_interface or allowed_attributes but which have declarations from a base class's security info don't get their security overwritten to be private.
- LP #143755: Also catch TypeError when trying to determine an indexable value for an object in PluginIndexes.common.UnIndex
- LP #143533: Instead of showing "0.0.0.0" as the SERVER_NAME request variable when no specific listening IP is configured for the HTTP server, do a socket lookup to show the current server's fully qualified name.
- LP #143722: Added missing permission to ObjectManager.manage_hasId, which prevented renaming files and folders via FTP.
- LP #143564: Request.resolve_url did not correctly re-raise exceptions encountered during path traversal.

11.36.2 Restructuring

- Removed catalog length migration code. You can no longer directly upgrade a Zope 2.7 or earlier database to Zope 2.13. Please upgrade to an earlier release first.
- Deprecated the `Products.ZCatalog.CatalogAwareness` and `CatalogPathAwareness` modules.
- Removed deprecated `catalog-getObject-raises` zope.conf option.
- Removed unmaintained HelpSys documents from ZCatalog and PluginIndexes. Useful explanations are given inside the form templates.
- Deprecate Products.ZCatalog's current behavior of returning the entire catalog content if no query restriction applied. In Zope 2.14 this will result in an empty LazyCat to be returned instead.
- Deprecate acquiring the request inside Products.ZCatalog's searchResults method if no explicit query argument is given.
- Cleaned up the Products.ZCatalog search API's. The deprecated support for using *<index id>_usage* arguments in the request has been removed. Support for overriding operators via the *<index id>_operator* syntax has been limited to the query value for each index and no longer works directly on the request. The query is now brought into a canonical form before being passed into the *_apply_index* method of each index.
- Factored out the *Products.MailHost* package into its own distributions. It will no longer be included by default in Zope 2.14 but live on as an independent add-on.

11.36.3 Features Added

- Merged the query plan support from both `unimr.catalogqueryplan` and `experimental.catalogqueryplan` into ZCatalog. On sites with large number of objects in a catalog (in the 100000+ range) this can significantly speed up catalog queries. A query plan monitors catalog queries and keeps detailed statistics about their execution. Currently the plan keeps track of execution time, result set length and support for the ILimitedResultIndex per index for each query. It uses this information to devise a better query execution plan the next time the same query is run. Statistics and the resulting plan are continuously updated. The plan is per running Zope process and not persisted. You can inspect the plan using the `Query Plan` ZMI tab on each catalog instance. The representation can be put into a Python module and the Zope process be instructed to load this query plan on startup. The location of the query plan is specified by providing the dotted name to the query plan dictionary in an environment variable called `ZCATALOGQUERYPLAN`.

- Various optimizations to indexes _apply_index and the catalog's search method inspired by experimental.catalogqueryplan.
- Added a new ILimitedResultIndex to Products.PluginIndexes and made most built-in indexes compatible with it. This allows indexes to consider the already calculated result set inside their own calculations.
- Changed the internals of the DateRangeIndex to always use IITreeSet and do an inline migration from IISet. Some datum tend to have large number of documents, for example when using default floor or ceiling dates.
- Added a new reporting tab to *Products.ZCatalog* instances. You can use this to get an overview of slow catalog queries, as specified by a configurable threshold value.
- Warn when App.ImageFile.ImageFile receives a relative path with no prefix, and then has to assume the path to be relative to "software home". This behaviour is deprecated as packages can be factored out to their own distribution, making the "software home" relative path meaningless.
- Updated distributions:
  - AccessControl = 2.13.2
  - DateTime = 2.12.5
  - DocumentTemplate = 2.13.1
  - Products.BTreeFolder2 = 2.13.1
  - Products.OFSP = 2.13.2
  - ZODB3 = 3.10.0b4

11.37 2.13.0a2 (2010-07-13)

11.37.1 Bugs Fixed

- Made ZPublisher tests compatible with Python 2.7.
- LP #143531: Fix broken object so they give access to their state.
- LP #578326: Add support for non-public permission attributes in the browser:view directive.

11.37.2 Restructuring

- No longer use HelpSys pages from `Products.OFSP` in core Zope 2.
- No longer create an *Extensions* folder in the standard instance skeleton. External methods will become entirely optional in Zope 2.14.
- Avoid using the `Products.PythonScripts.standard` module inside the database manager ZMI.
- Factored out the *Products.BTreeFolder2*, *Products.ExternalMethod*, *Products.MIMETools*, *Products.OFSP*, *Products.PythonScripts* and *Products.StandardCacheManagers* packages into their own distributions. They will no longer be included by default in Zope 2.14 but live on as independent add-ons.

- Factored out the *Products.ZSQLMethods* into its own distribution. The distribution also includes the *Shared.DC.ZRDB* code. The Zope2 distribution no longer includes the code automatically. Please depend on the new distribution yourself, if you use the functionality. To make the transition easier this change has been backported to Zope 2.12.9, so you can depend on the new distribution already in packages requiring at least that version of Zope 2.
- Made both *Shared* and *Shared.DC* namespace packages.
- Removed fallback code for old Python versions from *ZServer.FTPServer.zope_ftp_channel.push*.
- Removed fallback code for old *ZCatalog.catalog_object* function signatures from *Products.ZCatalog.ZCatalog.reindexIndex*.

11.37.3 Features Added

- Added official support for Python 2.7.
- Added a new API `get_packages_to_initialize` to `OFS.metaconfigure`. This replaces any direct access to `Products._packages_to_initialize`. The OFS.Application.install_package function takes care of removing entries from this list now.
- Added notification of `IDatabaseOpenedWithRoot`.
- Added a new API's `get_registered_packages, set_registered_packages` to `OFS.metaconfigure` which replace any direct access to `Products._registered_packages`.
- Changed product install so it won't write persistent changes only to abort them. Instead we don't make any database changes in the first place.
- Disabled persistent product installation in the default test configuration.
- Directly extend and use the Zope Toolkit KGS release 1.0a2 from http://download.zope.org/zopetoolkit/index/.
- Updated distributions:
  - DateTime = 2.12.4
  - nt_svcutils = 2.13.0

11.38 2.13.0a1 (2010-06-25)

This release includes all bug fixes and features of the Zope 2.12.8 release.

11.38.1 Distribution changes

- Moved AccessControl, DocumentTemplate (incl. TreeDisplay) and Products.ZCTextIndex to their own distributions. This removes the last direct C extensions from the Zope2 distribution.
- Moved the `zExceptions` package into its own distribution.
- Drop the dependency on the ThreadLock distribution, by using Python's thread module instead.
- Integrated the Products.signalstack / z3c.deadlockdebugger packages. You can now send a SIGUSR1 signal to a Zope process and get a stack trace of all threads printed out on the console. This works even if all threads are stuck.

11.38.2 Instance skeleton

- Changed the default for `enable-product-installation` to off. This matches the default behavior of buildout installs via plone.recipe.zope2instance. Disabling the persistent product installation also disabled the ZMI help system.
- Removed Zope2's own mkzeoinstance script. If you want to set up ZEO instances please install the zope.mkzeoinstance and use its script.
- Removed deprecated `read-only-database` option from zope.conf.
- LP #143232: Added option to 'zope.conf' to specify an additional directory to be searched for 'App.Extensions' lookups. Thanks to Rodrigo Senra for the patch.
- LP #143604: Removed top-level database-quota-size from zope.conf, some storages support a quota option instead.
- LP #143089: Removed the top-level zeo-client-name option from zope.conf, as it had no effect since ZODB 3.2.
- Removed no longer maintained `configure, make, make install` related installation files. Zope2 can only be installed via its setup.py.
- Removed the unmaintained and no longer functioning ZopeTutorialExamples from the instance skeleton.

11.38.3 Deprecated and Removed

- Finished the move of five.formlib to an extra package and removed it from Zope 2 itself. Upgrade notes have been added to the news section of the release notes.
- ZPublisher: Removed 'Main' and 'Zope' wrappers for Test.publish. If anybody really used them, he can easily use ZPublisher.test instead. In the long run ZPublisher.test and ZPublisher.Test might also be removed.
- ZPublisherExceptionHook: Removed ancient backwards compatibility code. Customized raise_standardErrorMessage methods have to implement the signature introduced in Zope 2.6.
- Removed ancient App.HotFixes module.
- Removed the deprecated `hasRole` method from user objects.
- Removed deprecated support for specifying `__ac_permissions__`, `meta_types` and `methods` in a product's `__init__`.
- Remove remaining support classes for defining permissions TTW.
- Removed the deprecated `five:containerEvents` directive, which had been a no-op for quite a while.
- Removed Products.Five.fivedirectives.IBridgeDirective - a leftover from the Interface to zope.interface bridging code.
- Marked the `<five:implements />` as officially deprecated. The standard `<class />` directive allows the same.

11.38.4 Refactoring

- Completely refactored `ZPublisher.WSGIResponse` in order to provide non-broken support for running Zope under arbitrary WSGI servers. In this (alternate) scenario, transaction handling, request retry, error handling, etc. are removed from the publisher, and become the responsibility of middleware.

- Moved the code handling ZCML loading into the `Zope2.App` package. The component architecture is now setup before the application object is created or any database connections are opened. So far the CA was setup somewhat randomly in the startup process, when the `Five` product was initialized.
- Moved Products.Sessions APIs from `SessionInterfaces` to `interfaces`, leaving behind the old module / names for backward compatibility.
- Centralize interfaces defined in Products.ZCTextIndex, leaving BBB imports behind in old locations.
- Moved `cmf.*` permissions into Products.CMFCore.
- Moved `TaintedString` into the new AccessControl.tainted module.
- Testing: Functional.publish now uses the real publish_module function instead of that from ZPublisher.Test. The 'extra' argument of the publish method is no longer supported.
- Moved `testbrowser` module into the Testing package.
- Moved general OFS related ZCML directives from Products.Five into the OFS package.
- Moved the `absoluteurl` views into the OFS package.
- Moved `Products/Five/event.zcml` into the OFS package.
- Moved `Products/Five/security.py` and security related ZCML configuration into the AccessControl package.
- Moved `Products/Five/traversing.zcml` directly into the configure.zcml.
- Moved `Products/Five/i18n.zcml` into the ZPublisher package.
- Moved `Products/Five/publisher.zcml` into the ZPublisher package.
- Ported the lazy expression into zope.tales and require a new version of it.

11.38.5 General

- Updated copyright and license information to conform with repository policy.
- LP #143410: Removed unnecessary color definition in ZMI CSS.
- LP #374810: `__bobo_traverse__` implementation can raise `ZPublisher.interfaces.UseTraversalDefault` to indicate that there is no special casing for the given name and that standard traversal logic should be applied.
- LP #142464: Make undo log easier to read. Thanks to Toby Dickinson for the patch.
- LP #142401: Added a link in the ZMI tree pane to make the tree state persistent. Thanks to Lalo Martins for the patch.
- LP #142502: Added a knob to the Debug control panel for resetting profile data. Thanks to Vladimir Patukhov for the patch.
- ZCTextIndex query parser treats fullwidth space characters defined in Unicode as valid white space.

11.38.6 Updated distributions

- Jinja2 = 2.5.0
- RestrictedPython = 3.6.0a1
- Sphinx = 1.0b2
- transaction = 1.1.0

- ZConfig = 2.8.0
- ZODB3 = 3.10.0b1
- zope.annotation = 3.5.0
- zope.broken = 3.6.0
- zope.browsermenu = 3.9.0
- zope.browserpage = 3.12.2
- zope.browserresource = 3.10.3
- zope.component = 3.9.4
- zope.configuration = 3.7.2
- zope.container = 3.11.1
- zope.contentprovider = 3.7.2
- zope.contenttype = 3.5.1
- zope.event = 3.5.0-1
- zope.exceptions = 3.6.0
- zope.filerepresentation = 3.6.0
- zope.i18nmessageid = 3.5.0
- zope.interface = 3.6.1
- zope.location = 3.9.0
- zope.lifecycleevent = 3.6.0
- zope.ptresource = 3.9.0
- zope.publisher = 3.12.3
- zope.schema = 3.6.4
- zope.sendmail = 3.7.2
- zope.site = 3.9.1
- zope.structuredtext = 3.5.0
- zope.tales = 3.5.1
- zope.testbrowser = 3.9.0
- zope.testing = 3.9.3
- zope.traversing = 3.12.1
- zope.viewlet = 3.7.2

11.38.7 Bugs Fixed

- LP #143391: Protect against missing acl_users.hasUsers on quick start page.