Products.GenericSetup Documentation

Release 2.0.2

Zope CMF Developers

Mar 04, 2020

Contents

1 Overview **1**

2 Configurations Included **3**

3 Extending The Tool **5**

4 Providing Profiles **7**

5 Other Documentation **9**
- 5.1 Installing `Products.GenericSetup` . . . 9
- 5.2 How-to: Writing setup handlers for `Products.GenericSetup` . . . 9
- 5.3 About Profiles . . . 11
- 5.4 Changelog . . . 12
- 5.5 Historical Changelog . . . 17
- 5.6 GenericSetup Credits . . . 26
- 5.7 Glossary . . . 26

6 Indices and tables **27**

Index **29**

CHAPTER 1

Overview

This product provides a mini-framework for expressing the configured state of a Zope Site as a set of filesystem artifacts. These artifacts consist of declarative XML files, which spell out the configuration settings for each "tool" in the site , and supporting scripts / templates, in their "canonical" filesystem representations.

CHAPTER 2

Configurations Included

The `setup_tool` knows how to export / import configurations and scripts for the following components of a site:

- itself :)
- removal / creation of specified tools
- the role / permission map on the "site" object (its parent)
- properties of the site object
- "Placeful" utilities and adapters registered in the local site manager. Placeless utilities can only be imported.

CHAPTER 3

Extending The Tool

Third-party products extend the tool by registering handlers for import / export of their unique tools.

See also:

See *How-to: Writing setup handlers for Products.GenericSetup* for a step by step how-to.

CHAPTER 4

Providing Profiles

GenericSetup doesn't ship with any profile. They have to be provided by third-party products and depend on the registered handlers.

See also:

See *About Profiles* for more details.

CHAPTER 5

Other Documentation

5.1 Installing `Products.GenericSetup`

5.1.1 Installation via `easy_install`

GenericSetup can be installed using the `setuptools` utility, **`easy_install`**, into the `site-packages` directory of your Python installation:

```
$ bin/easy_install Products.GenericSetup
```

5.1.2 Installation via `zc.buildout`

TBD

5.1.3 Manual Installation

To install this package manually, without using `setuptools`, simply untar the package file downloaded from the PyPI site and look for the folder named `GenericSetup` underneath the `Products` folder at the root of the extracted tarball. Copy or link this `GenericSetup` folder into the `Products` folder of your Zope instance and restart Zope.

5.2 How-to: Writing setup handlers for `Products.GenericSetup`

If your product subclasses existing tools or provides new tools (or new sub-object classes), it might need to supply its own setup handlers in order to support exporting / importing tool settings using GenericSetup.

5.2.1 Step 1:

Identify those classes in your product that need their own setup handlers. In theory you don't need your own handlers for classes which implement a CMF tool interface that already has a setup adapter. In practice the adapters shipped with the CMF sometimes use methods that are not part of the interface, so you have to verify that they really work for your classes.

5.2.2 Step 2:

Make sure those classes that need setup handlers have Zope3 style interfaces. Later you will write setup adapters for those interfaces.

5.2.3 Step 3:

Create an `exportimport` module inside your product. If you plan to write many setup handlers this can be a sub-package.

5.2.4 Step 4:

Decide which kind of setup handler you need:

body adapter For objects represented by a complete file body. Provides `Products.GenericSetup.interfaces.IBody`.

XML adapter A 'body adapter' in XML format. Also provides `Products.GenericSetup.interfaces.IBody`, but has its own base class because XML is the preferred format.

node adapter For sub-objects represented by an XML node of the parent's XML document. Provides `Products.GenericSetup.interfaces.INode`. This is useful for sub-objects of complex tools. Custom catalog index or action classes need this kind of adapter.

import and export steps Top level handlers can be registered to manage import steps and / or export steps and call the body adapters.

If you use the base classes from `Products.GenericSetup.utils`, XML and node adapters are implemented in a very similar way. Both can mix in `ObjectManagerHelpers` and `PropertyManagerHelpers`.

5.2.5 Step 5:

`Products.CMFCore.exportimport` contains many examples for XML and node adapters. If you need a pure body adapter, `Products.GenericSetup.PythonScripts` would be a good example. Follow those examples and write your own multi adapter, register it for the interface of your class and for `Products.GenericSetup.interfaces.ISetupEnviron` and don't forget to write unit tests.

Adapters follow the convention that `self.context` is always the primary adapted object, so the minimal setup context (`ISetupEnviron`) used in these multi adapters is `self.environ`.

XML and body adapters are always also small node adapters. This way the XML file of the container contains the information that is necessary to create an empty object. The handler of the container has to set up sub-objects before we can adapt them and configure them with their own handlers. The base classes in `Products.GenericSetup.utils` will care about that.

5.2.6 Step 6:

If your adapter is a top-level adapter (e.g for a tool), you need import and export steps that know how to use the adapter. Again there are many examples in `Products.CMFCore.exportimport`.

To make those steps available you have to add them to `export_steps.xml` and `import_steps.xml` of a setup profile and to load that profile into the setup tool.

5.2.7 Step 7:

Now you are done. To ship default settings with your product, make your settings through the ZMI (or set your stuff up the old way if you have old setup code like an `Install.py`) and export your settings using the setup tool. Unpack the exported tarball into a `profiles` subdirectory of your product, and then add `gs:registerProfile` entries to its `configure.zcml` file to register that directory as a *base profile* or *extension profile*.

See also:

See *About Profiles* for more details.

5.3 About Profiles

5.3.1 Overview

`GenericSetup` uses two different kinds of profiles: a *base profile* captures the entire state of the site, while an *extension profile* represents an add-on / delta to be applied to the site's configuration.

5.3.2 Registering Profiles

By convention profiles are stored in a `profiles` directory within a Zope product. They have to be registered explicitly using either the `Products.GenericSetup.profile_registry.registerProfile()` API or the equivalent `genericsetup:registerProfile` ZCML directive.

Here is example ZCML for a Zope product, `MyProduct`, which extends another product, `BaseProduct`:

```
<genericsetup:registerProfile
    name="install"
    title="Install MyProduct Extension"
    description="Adds local settings necessary for MyProduct."
    provides="Products.GenericSetup.interfaces.EXTENSION"
    for="Products.BaseProduct.interfaces.IBaseRoot"
    />
```

See also:

`Products.GenericSetup.zcml.IRegisterProfileDirective` defines the API for this directive.

Alternatively, profiles can be registered by calling the `Products.GenericSetup.profile_registry.registerProfile()` API.

Here is the code for the same example:

```
from Products.BaseProduct.interfaces import IBaseRoot
from Products.GenericSetup import EXTENSION
from Products.GenericSetup import profile_registry
```

(continues on next page)

(continued from previous page)

```
profile_registry.registerProfile(
        name='install',
        title='Install MyProduct Extension',
        description='Adds local settings necessary for MyProduct.',
        path='profiles/install',
        product='Products.MyProduct',
        profile_type=EXTENSION,
        for_=IBaseRoot)
```

See also:

See IProfileRegistry.registerProfile for further details.

Note: Using this API for product initialization is deprecated.

5.3.3 Update Directives

For some XML elements there are additional attributes and values to specify update directives. They are only useful for extension profiles and you will never see them in snapshots and exports.

The following directives are generally useful for container elements and implemented by some setup handlers. Products using GenericSetup can also implement other update directives.

`id="*"` wildcard

Updates all existing items in the container with the same settings.

`remove`

Removes the specified item if it exists.

`insert-before` and `insert-after`

`insert-before` and `insert-after` specify the position of a new item relative to an existing item. If they are omitted or not valid, items are appended. You can also use `*` as wildcard. This will insert the new item at the top (before all existing items) or the bottom (after all existing items). If an item with the given ID exists already, it is moved to the specified position. This directive makes only sense for ordered containers.

5.3.4 Other Special Directives

`purge`

By default existing settings are purged before applying settings from base profiles. Extension profiles are applied in update mode. This directive overrides the default behavior. If True the existing settings of the current object are always purged, if False they are not purged.

5.4 Changelog

5.4.1 2.0.2 (2020-01-29)

- Remove Zope 2.13 fossils to stay compatible with Zope 5

- Force saving unpersisted changes in toolset registry. Fixes issue 86.

5.4.2 2.0.1 (2019-10-12)

- Fix the sorting of upgrade steps. [vanderbauwhede]

5.4.3 2.0 (2019-05-10)

- no changes since 2.0b6

5.4.4 2.0b6 (2019-04-09)

- Zope 4 ZMI: Added icon
- Zope 4 ZMI: declare that creating a GS tool does not need an add dialog (#80)
- clean up `setup.py` and remove support for `setup.py test` (#73)
- add support for unicode data in `writeDataFile` (#79)
- Specify supported Python versions using `python_requires` in setup.py
- Adding suport for Python 3.8

5.4.5 2.0b5 (2018-12-14)

- Fix deprecation warnings for `cgi.escape` by using `html.escape` (#76)

5.4.6 2.0b4 (2018-11-22)

- Convert input from xml configuration with correct encoding before passing to type_converter. (#77) [sallner]

5.4.7 2.0b3 (2018-11-07)

- Do not turn ulines and multiple selection into bytes. [davisagli]
- Set body of PythonScripts as text in py3. [pbauer]
- Compare encodings so that UTF-8 and utf-8 are the same. [pbauer]
- Compare DOM as text in py3. [pbauer]

5.4.8 2.0b2 (2018-10-17)

New features:

- Add Python 3.7 support.
- Support *zope.configuration >= 4.2*.

Bug fixes:

- Proper string/bytes handling for _createObjectByType. In Python2 everything is written as bytes, while on Python3 everything is written as text except files and images which are stored as bytes [ale-rt]

5.4.9 2.0b1 (2018-05-16)

Breaking changes:

- Require Zope 4.0b4 as minimum supported Zope version and drop explicit `Zope2` egg dependency.
- Drop Python 3.4 support

New features:

- Fixed tests with `Products.ZCatalog 4.1`. [maurits]
- When `metadata.xml` parsing fails, show the filename in the `ExpatError`. Fixes Plone issue 2303.
- Prevent AttributeError 'NoneType' object has no attribute 'decode'. [maurits]
- Finished compatibility with Python 3.5 and 3.6
- Made the code PEP-8 compliant

Bug fixes:

- Do not mask KeyError in 'getProfileDependencies' from missing dependency profiles. Refs: https://github.com/plone/Products.CMFPlone/issues/2228 [ida]

5.4.10 1.10.0 (2017-12-07)

Breaking changes:

- Require Zope 4.0a6 as minimum supported Zope version.
- Moved support for *MailHost* import/export into the `Products.MailHost` package to cut the hard dependency.

New features:

- Added `tox` testing configuration.
- Pushed documentation to RTD: https://productsgenericsetup.readthedocs.io/.

5.4.11 1.9.1 (2017-05-06)

Bug fixes:

- Fixed `upgradeStep` discriminator so that similar steps for different profiles will not conflict.
- Fixed `upgradeDepends` discriminator so that steps inside `upgradeSteps` will conflict with steps outside if they have the same `checker`.
- Fix import of UnrestrictedUser.

5.4.12 1.9.0 (2017-05-04)

Breaking changes:

- Drop support for Python 2.6.
- Require Zope 4.0a3 as minimum supported Zope version.

5.4.13 1.8.7 (2017-03-26)

- Allow registering the same profile twice if it really is the same. This is mostly for tests where the registry may not be cleaned up correctly in case of problems in test teardown. If you register the same profile twice in zcml, you still get a conflict from `zope.configuration` during Zope startup. [maurits]

5.4.14 1.8.6 (2016-12-30)

- Added a `purge_old` option to the tarball import form. By default this option is checked, which matches the previous behavior. If you uncheck it, this avoids purging old settings for any import step that is run. [maurits]

5.4.15 1.8.5 (2016-11-01)

- Stopped using a form library to render the components form.

5.4.16 1.8.4 (2016-09-21)

- Made `_profile_upgrade_versions` a PersistentMapping. When `(un)setLastVersionForProfile` is called, we migrate the original Python dictionary. This makes some code easier and plays nicer with transactions, which may especially help during tests. [maurits]

5.4.17 1.8.3 (2016-04-28)

- Allowed overriding required and forbidden tools in `toolset.xml`. If a tool is currently required and you import a `toolset.xml` where it is forbidden, we remove the tool from the required list and add it to the forbidden list. And the other way around. The previous behavior was to raise an exception, which left no way in xml to remove a tool. Fail with a ValueError when the `remove` keyword is used. The expected behavior is unclear. [maurits]

5.4.18 1.8.2 (2016-02-24)

- Added optional `pre_handler` and `post_handler` to `registerProfile` directive. When set, these dotted names are resolved to a function and are passed the setup tool as single argument. They are called before and after applying all import steps of the profile they are registered for. [maurits]
- Sorted import profiles alphabetically lowercase. Allow selecting a profile by title or id. [maurits]
- Do not show dependency options on the full import tab when there are no dependencies. [maurits]
- Do not select a profile by default in the import tabs. [maurits]
- Added simple toggle for all steps on the advanced import tab. Also added this on the export tab. [maurits]
- Fixed importing a tarball. This got an AttributeError: "'NoneType' object has no attribute 'startswith'". [maurits]
- Split overly complex Import tab into three tabs: Import (for importing a full profile), Advanced Import (the original `manage_importSteps` url leads to this tab), and Tarball Import. [maurits]
- Show note on import tab when there are pending upgrades. Especially show this for the currently selected profile. [maurits]

- Upgrades tab: show profiles with pending upgrades separately. These are the most important ones. This avoids the need to manually go through the whole list in order to find profiles that may need action. This uses new methods on the setup tool: `hasPendingUpgrades`, `listProfilesWithPendingUpgrades`, `listUptodateProfiles`. [maurits]

5.4.19 1.8.1 (2015-12-16)

- Purge the profile upgrade versions before applying a base profile.
- Added `purgeProfileVersions` method to `portal_setup`. This removes the all profiles profile upgrade versions.
- Added `unsetLastVersionForProfile` method to `portal_setup`. This removes the profile id from the profile upgrade versions. Calling `setLastVersionForProfile` with `unknown` as version now has the same effect.

5.4.20 1.8.0 (2015-09-21)

- Be more forgiving when dealing with profile ids with or without `profile-` at the start. All functions that accept a profile id argument and only work when the id does *not* have this string at the start, will now strip it off if it is there. For example, `getLastVersionForProfile` will give the same answer whether you ask it for the version of profile id `foo` or `profile-foo`.
- Dependency profiles from `metadata.xml` that are already applied, are not applied again. Instead, its upgrade steps, if any, are applied. In code you can choose the old behavior of always applying the dependencies, by calling `runAllImportStepsFromProfile` with `dependency_strategy=DEPENDENCY_STRATEGY_REAPPLY`. There are four strategies, which you can choose in the ZMI.

5.4.21 1.7.7 (2015-08-11)

- Fix: when the last applied upgrade step had a checker, the profile version was not updated. Now we no longer look at the checker of the last applied step when deciding whether to set the profile version. The checker, if any is set, normally returns True before running the step (it can be applied), and False afterwards (it was already applied).
- Add `upgradeProfile` method to setup tool. This method applies all upgrades steps for the given profile, or updates it to the optional given version. If the profile does not exist, or if there is no upgrade step to go to the specified version, the method warns and does nothing.
- Check the boolean value of the `remove` option when importing objects. Previously we only checked if the `remove` option was given, regardless of its value. Supported are `True`, `Yes`, and `1`, where case does not matter. The syntax for removing objects, properties, and elements is now the same.
- Support `remove="True"` for properties.

5.4.22 1.7.6 (2015-07-15)

- Enable testing under Travis.
- Fix compatibility with Setuptools 8.0 and later. Upgrade steps could get sorted in the wrong order, especially an empty version string (upgrade step from any source version) sorted last instead of first.

5.4.23 1.7.5 (2014-10-23)

- Allow skipping certain steps on `runAllImportStepsFromProfile`.

5.4.24 1.7.4 (2013-06-12)

- On import, avoid clearing indexes whose state is unchanged.

5.4.25 1.7.3 (2012-10-16)

- Sort profiles on Upgrade form.
- Use clickable labels with checkboxes on import, export and upgrade forms to improve usability.

5.4.26 1.7.2 (2012-07-23)

- Avoid using `manage_FTPGet` on snapshot exports: that method messes up the response headers.
- ZopePageTemplate handler: Fix export encoding: since 1.7.0, exports must be UTF-8 strings

5.4.27 1.7.1 (2012-02-28)

- Restore the ability to make the setup tool use only import / export steps explicitly called out by the current profile, ignoring any which might be globally registered. This is particularly useful for configuring sites with baseline profiles, where arbitrary add-on steps are not only useless, but potentially damaging.

5.4.28 1.7.0 (2012-01-27)

- While importing `toolset.xml`, print a warning when the class of a required tool is not found and continue with the next tool. The previous behaviour could break the install or uninstall of any add-on, as the missing class may easily be from a different unrelated add-on that is no longer available in the zope instance.
- Exporters now explicitly only understand strings. The provided registry handlers encode and decode data automatically to and from UTF-8. Their default encoding changed from None to UTF-8. If you have custom registry handlers, ensure that you encode your unicode. Check especially if you use a page template to generate xml. They return unicode and their output must also encoded. If you choose to encode your strings with UTF-8, you can be sure that your code will also work with GenericSetup < 1.7

5.5 Historical Changelog

5.5.1 1.6.8 (2013-01-27)

- Accomodate `PluginIndexes.exportimportPluggableIndexNodeAdapter` to being registered for indexes which have no `indexed_attrs` (e.g., Plone's `GopipIndex`).

5.5.2 1.6.7 (2013-01-23)

- On import, avoid clearing indexes whose state is unchanged.

5.5.3 1.6.6 (2012-02-28)

- Restore the ability to make the setup tool use only import / export steps explicitly called out by the current profile, ignoring any which might be globally registered. This is particularly useful for configuring sites with baseline profiles, where arbitrary add-on steps are not only useless, but potentially damaging.

5.5.4 1.6.5 (2012-01-27)

- While importing `toolset.xml`, print a warning when the class of a required tool is not found and continue with the next tool. The previous behaviour could break the install or uninstall of any add-on, as the missing class may easily be from a different unrelated add-on that is no longer available in the zope instance.

5.5.5 1.6.4 (2011-10-31)

- Add three missing explicit `InitializeClass` calls.
- Adjust test assertions in `test_differ` to cope with changes to the diff library done in Python 2.7.
- LP #850665: match permission binding to method name.
- LP #602989: export / import new MailHost properties, `smtp_queue` and `smtp_queue_directory`.
- ZCML: Don't require description for the `upgradeDepends` directive.
- PythonScript handler: Normalize newlines during import.
- No longer rely on `bobobase_modification_time`.
- TarballImportContext: Fix type of `getLastModified` return value.
- LP #388380: added docstrings to `Products.GenericSetup.tool.manage_deleteImportSteps` and `manage_deleteExportSteps`, fixing broken TTW deletion.
- Avoid using DateTime values for file system time stamps in tests.
- Generate unique ids for each profile when profiles get imported multiple times within a second. This change removes spurious failures in fast tests of packages that use GenericSetup.

5.5.6 1.6.3 (2011-04-02)

- Fix crash at export when a node had None value.
- Refactor global registries to use global named utilities.
- Fix the `profile_id` UnboundLocalError in the `upgradeDepends` directive when `import_profile` is not None.
- Use `zope.formlib` directly, rathter than via now-removed `five.formlib` dependency.
- tool: `listContextInfos` now returns profile infos sorted by type and title. This change makes it easier to select profiles on the "Import" and "Comparison" tab.
- Property import/export: Fix two `date` property issues. Naive `date` values are now exported without time zone. And purging non-deletable `date` properties is fixed.
- Export content objects whose `manage_FTPget` returns a custom iterator with `file` and `size` properties. https://bugs.launchpad.net/bugs/722726
- Property import: Fix `lines` and `tokens` import. Modifying sequences without adding new elements was broken.

- Toolset import: Support replacement of subclassed tools.

5.5.7 1.6.2 (2010-08-12)

- testing: Remove broken `run` function. Unit test modules are no longer directly executable.
- DateTime 2.12.5 does away with a special case representing DateTime values for midnight (00:00:00) without their time and time zone values. So DateTimes formerly rendered as `2010/01/01` in the UTC timezone now render as `2010/01/01 00:00:00 UTC`. The XML used for testing has been changed to reflect this change. Since the change is only cosmetic, nothing changes with respect to importing `Time`-less date values.
- Toolset import: Don't ignore errors in `ImmutableId._setId()`.

5.5.8 1.6.1 (2010-07-04)

- Use the standard library's doctest module.
- Suppress deprecation warnings for Zope 2.13.
- Un-break tool upgrade tab after running an upgrade step which used `None` as its destination version. https://bugs.launchpad.net/bugs/553338

5.5.9 1.6.0 (2010-03-08)

- When exporting a tarball, make the directory entries executable.
- When the MailHost `smtp_uid` or `smtp_pwd` settings are None, export them as empty string, to avoid an AttributeError during export.
- Don't try to reinitialize a tool if an instance of the tool exists but the desired tool class was not resolvable. Show a warning instead of failing.
- Remove backwards compatibility code for no longer supported Zope versions.

5.5.10 1.6.0b1 (2010-01-31)

- Require at least Zope 2.12.3 and use the optional `five.formlib` extension.
- Fix bug in the export code of persistent utilities with explicit OFS ids.
- Prefer the class over the `five:implements` ZCML directive.

5.5.11 1.5.0 (2010-01-01)

- Ensure there is a valid component registry (not the global registry) before importing or exporting it via the components handler.
- Ensure that the "Import" ZMI tab does not blow up if no base profile has been selected, and make it a little more user-friendly.
- Log if the components handler runs and has nothing to import.
- Use five.formlib in favor of Products.Five.formlib if it is available.
- Remove testing dependency on zope.app.testing.ztapi.
- tarball contexts: Fix export and import of directory structures.

5.5.12 1.5.0b1 (2009-09-25)

- LP #388380: remove obsolete STX docs from the package directory.
- Made export / import features for old-school `TextIndex` (removed in Zope 2.12) conditional.
- Add support for import / export of subscribers from component registry.
- In utility removal, avoid adding to-be-removed utility when it is already missing from the local component registry.
- Prefer `for` to `for_` in component handler adapter directive. To support import of existing profiles `for_` is used as a fallback.
- Change `testing.py` to directly load `zope.traversing`'s ZCML instead of going via the Five `traversing.zcml` BBB shim.
- Add new feature to the component handler. For factory based utilities you can now specify an additional id. All factory based utilities will now by default be added to the site manager (being an ObjectManager itself) as an object and this persistent object is registered as the utility. On removal both the registration and the object are removed. The new id argument is used to specify the id of the object as set via *__name__*. This change makes these utilities introspectable in the ZMI and clearly separates the persistent object and utility registration aspect.
- Make `TarballImportContext` comaptible with Python 2.6 `tarfile` module.
- Clean up / normalize imports:

  o Don't import from Globals; instead, use real locations.

  o **Make other imports use the actual source module, rather than an** intermediate (e.g., prefer importing `ClassSecurityInfo` from `AccessControl.SecurityInfo` rather than from `AccessControl`).

  o Avoid relative imports, which will break in later versions of Python.
- events: Add `handleProfileImportedEvent` subscriber. After a full import it updates last version for profile.
- UpgradeSteps: Improve `listUpgradeSteps` behavior. If versions and checker are specified for a step, the checker is used as an additional restriction.
- Component registry import: Add the ability to unregister a component by specifying the "remove" attribute inside a utility node. (https://bugs.launchpad.net/zope-cmf/+bug/161728)
- Property import/export tests: Add testing for non-ASCII properties. (https://bugs.launchpad.net/zope-cmf/+bug/202356) (https://bugs.launchpad.net/zope-cmf/+bug/242588)
- Add `genericsetup:upgradeDepends` ZCML tag, defining a specialized upgrade step that re-applies one or more import steps from a GS profile during an upgrade process
- Add `IChunkedImportContext` interface, allowing RAM-efficient chunked reads of large files, and implement for `DirectoryImportContext`. (https://bugs.launchpad.net/zope-cmf/+bug/259233)
- Add `IChunkedExportContext` interface, allowing RAM-efficient chunked writes of large files, and implement for `DirectoryExportContext`. (https://bugs.launchpad.net/zope-cmf/+bug/257365)
- Provide default for dependencies when processing `metadata.xml`, to avoid a KeyError. (https://bugs.launchpad.net/zope-cmf/+bug/255301)
- Handle utility factories cleanly if `zope.component>=3.5.0` is used.
- tool and utils: Remove deprecated code.

- Update `PropertyManagerHelpers`, making it possible to remove elements from a property by adding a `remove="True"` attribute to the element. This change also allows reordering elements, since new elements are always added at the end of the list.
- Made `PropertyManagerHelpers` class work for non-PropertyManager objects:

  o **Derived classes can supply a `_PROPERTIES` scehma, which is then used** to mock up a temporary propertysheet for the object. The adapter's methods (`_extractProperties`, `_purgeProperties`, `_initProperties`) then run against that propertysheet.
- Add logic to respect the destination of upgrade steps when determining their applicability.
- Enhance the readability of the upgrades tab on the tool.
- Use the `pkg_resources.parse_version` to normalize versions before comparing them inside the upgrade code, ensuring that pre-release versions are handled correctly. Also use the normalize code when sorting versions on the tools ZMI upgrades page.
- Update the `upgradeStep` directive schema: `description` is not required.
- Introduce a new `IComponentsHandlerBlacklist` interface: named utilities registered for it and provide sequences of interfaces which should not be handled by the standard components registry adapter. This change allows more specialized export/import handlers to take full control over the components they care about.
- When loading multiple profiles, reload the list of steps to use after each import. https://bugs.launchpad.net/zope-cmf/+bug/213905

5.5.13 1.4.5 (2009-06-20)

- events: Add `handleProfileImportedEvent` subscriber. After a full import, it updates last version for profile. (Backported from trunk)
- Add a `for_=None` parameter to `tool.listProfileInfo` to have the same signature as `registry.listProfileInfo`, allowing profiles to be filtered by interfaces.

5.5.14 1.4.4 (2009-05-15)

- Make sure that `manage_createSnapshot` returns something to the browser when it's done, preventing an apparent hang. (http://dev.plone.org/plone/ticket/8452, https://bugs.launchpad.net/zope-cmf/+bug/161730)
- Fix invalid XML for the "Import" tab so it doesn't break when rendered with Chameleon.

5.5.15 1.4.3 (2009-04-22)

- Recognize acquisition-wrapped components as being of the right underlying type when testing for replacement during import. (https://bugs.launchpad.net/zope-cmf/+bug/365202)
- Don't fail when a sub-item cannot be adapted after creation when importing a folder. (https://bugs.launchpad.net/zope-cmf/+bug/300315)
- Avoid even an explicit purge of the rolemap if no XML file is present in a given context. (https://bugs.launchpad.net/zope-cmf/+bug/279294)
- Change upgrade logic to set the current version after an upgrade to the destination version of the last step run, instead of the current profile version.

5.5.16 1.4.2.2 (2008-09-22)

- Packaging update: version of 1.4.2.1 said "1.4.2".

5.5.17 1.4.2.1 (2008-09-22)

- Packaging update: version of 1.4.2 said "1.4.2dev".

5.5.18 1.4.2 (2008-09-22)

- Add `IChunkedImportContext` interface, allowing RAM-efficient chunked reads of large files, and implement for `DirectoryImportContext`. (https://bugs.launchpad.net/zope-cmf/+bug/259233)
- Add `IChunkedExportContext` interface, allowing RAM-efficient chunked writes of large files, and implement for `DirectoryExportContext`. (https://bugs.launchpad.net/zope-cmf/+bug/257365)
- Update local component registry importer to prevent it from overwriting existing utilities if they are already of the correct type
- Property import/export tests: Fix and test for non-ASCII properties. (https://bugs.launchpad.net/zope-cmf/+bug/202356) (https://bugs.launchpad.net/zope-cmf/+bug/242588)
- Provide default for dependencies when processing `metadata.xml`, to avoid a KeyError. (https://bugs.launchpad.net/zope-cmf/+bug/255301)
- Update `PropertyManagerHelpers` to make it possible to remove elements from a property by adding a `remove="True"` attribute to the element. This can also be used to reorder elements since new elements are always added at the end of the list.

5.5.19 1.4.1 (2008-05-27)

- When loading multiple profiles reload the list of steps to use after each import. https://bugs.launchpad.net/zope-cmf/+bug/213905

5.5.20 1.4.0 (2008-03-23)

- In `getProfileImportDate`, avoid errors where one object's id is a prefix of another id.

5.5.21 1.4.0-beta (2008-02-07)

- During object manager imports, suppress an error when trying to remove an object that was already removed.
- utils: Add `MarkerInterfaceHelpers`.
- Add default values to the `registerProfile` ZCML directive.
- Add a ZMI interface to find and remove invalid steps from the persistent registries.
- Register all GenericSetup import and export steps globally.
- Remove duplicated test (https://bugs.launchpad.net/zope-cmf/+bug/174910)
- Don't create empty `import_steps.xml` and `export_steps.xml` files.
- Fix relative paths for profile dependencies.

- Add support for context dependencies in profiles.
- Deprecate the `version` field for import steps.
- Deprecate reading of `version.txt` to get the version for a profile.
- Fire events before and after importing.
- Use zcml to register import and export steps.

5.5.22 1.3.3 (2007-12-29)

- Be more careful in checking context id validity.
- tool: Avoid initializing already-existing tools they already exist in the site.

5.5.23 1.3.2 (2007-09-11)

- Ignore import and export step handlers that we can not resolve.
- Restore the import context after running steps from a profile so we do not break on nested calls.
- components: Provide log output when purging utilities or adapters.
- components: Fix an undefined variable name in a log message.

5.5.24 1.3.1 (2007-08-08)

- components: correct the object path for the site root to be the empty string.
- components: Made output more diff friendly.
- utils: Add warnings to old code. `ImportConfiguratorBase` and `ExportConfiguratorBase` will become deprecated as soon as GenericSetup itself no longer uses them. `HandlerBase` is now deprecated.
- components: Add `components_xmlconfig.html` form. This view allows to inspect and edit component registrations. It is also available under the ZMI tab `manage_components`.

5.5.25 1.3 (2007-07-26)

- components: Remove non-functional support for registering objects in nested folders. We only support objects available in the component registry's parent now. The component registry needs to be either acquisition wrapped or have a `__parent__` pointer to get to the parent.

5.5.26 1.3-beta (2007-07-12)

- Guard against situations where encoded text may be compared by the differ. (http://www.zope.org/Collectors/CMF/471)
- Extend the ZCatalog import/export mechanism to allow removal of metadata columns in addition to adding them. (http://www.zope.org/Collectors/CMF/483)
- Made sure we register Acquisition free objects as utilities in the components handler.
- Profiles now support version numbers; setup tool tracks profile versions during upgrades.

- Add support for nested `upgradeStep` directives; expanded upgrade step registry into a real registry object and not just a dictionary.
- Add support for `metadata.xml` in the profile (read during profile registration) to register profile description, version, and dependencies.
- Deprecate `runImportStep` and `runAllImportSteps` in favor of `runImportStepFromProfile` and `runAllImportStepsFromProfile`.
- Merged CPS's `upgradeStep` ZCML directive, w/ corresponding tool support.
- Add a "last imported" date to the list of extension profiles, and to the baseline profile.
- Renamed the "Properties" tab to "Profiles".
- Remove the `create_report` decoy in the ZMI view methods: there was never any UI for passing any value other than the default, anyway, and the report objects are too useful to omit.
- Refactor the "Properties" tab to separate baseline profiles from extension profiles, marking the option to reset the baseline as potentially dangerous for sites which already have one. Allow importing one or more extension profiles directly (all steps) from the "Properties" tab.
- No longer read the toolset xml and update the toolset regustry on import context change. Doing this only during the toolset step import should be sufficient.
- testing: No longer set up any ZCML in test base classes. This change is not backwards compatible. If you are using these base classes for testing custom handlers, you have to add the necessary ZCML setup and tear down. Using test layers is recommended.
- Add support for importing-exporting Zope 3 component registries by folding in Hanno Schlichting's GSLocalAddons product.

5.5.27 1.2-beta (2006-09-20)

- tool: Add support for uploading a tarball on the "Import" tab (i.e., one produced on the export tab).
- docs: Add SampleSite demo product.
- ProfileRegistry: Add `registerProfile` ZCML directive. Using the old registerProfile method in initialize() is now deprecated. See doc/profiles.txt for details.
- ProfileRegistry: `product` should now be the module name. For backwards compatibility `product` is still first looked up in Products before searching the default module search path.
- ZCTextIndex handler: Fix `indexed_attr` import. (http://www.zope.org/Collectors/CMF/436)
- docs: Add "Registering Profiles" section to profiles.txt.
- Add support for PageTemplate import/export, modeled closely after existing PythonScript support
- Track steps with unresolved dependencies, retrying after inserting remaining steps. Fixes cases where dependency sorting was highly reliant on steps being added in the right order to work. E.g., import step `A` depends on import step `B` which depends on step `C`, and step `C` gets processed early, and they were processed in the order `A`, `C`, `B`.

5.5.28 1.1 (2006-04-16)

- ZCatalog handler: Implement the `remove` directive for indexes, allowing extension profiles that remove or replace indexes.
- Give `getExportStepRegistry` the correct security declaration.

5.5.29 1.1-beta2 (2006-03-26)

- No changes - tag created to coincide with CMF 2.0.0-beta2

5.5.30 1.1-beta (2006-03-08)

- Allowed subclasses of `DAVAwareFileAdapter` to override the filename in which the file is stored.
- Add a `doc` directory including some basic documentation.
- Made GenericSetup a standalone package independent of the CMF
- Add `for_` argument to profile registry operations. A profile may be registered and queried as appropriate to a specific site interface; the default value, `None`, indicates that the profile is relevant to any site. Note that this is essentially an adapter lookup; perhaps we should reimplement it so.
- Forward-port changes from GenericSetup 0.11 and 0.12 (which were created in a separate repository).
- Merge sequence propertise with the purge="False" attribute rather than purgeing (the sequences are treated as sets, which means that duplicates are removed). This is useful in extension profiles.
- Don't export or purge read-only properties. Correctly purge non-deletable int/float properties.
- Correctly quote XML on export.

5.5.31 1.0 (2005-09-23)

- CVS tag: GenericSetup-1_0
- Forward-port i18n support from CMF 1.5 branch.
- Forward-port BBB for old instances that stored properties as lists from CMFSetup.
- Forward-port fix for tools with non unique IDs from CMFSetup.

5.5.32 0.12 (2005-08-29)

- CVS tag: GenericSetup-0_12
- Import requests now create reports (by default) which record any status messages generated by the profile's steps.

5.5.33 0.11 (2005-08-23)

- CVS tag: GenericSetup-0_11
- Add report of messages generated by import to the "Import" tab.
- Consolidate `ISetupContext` implementation into base class, `SetupContextBase`.
- Add `note`, `listNotes`, and `clearNotes` methods to `ISetupContext`, allowing plugins to record information about the state of the operation.

5.5.34 0.10 (2005-08-11)

- CVS tag: GenericSetup-0_10
- Add TarballImportContext, including full test suite.

5.5.35 0.9 (2005-08-08)

- CVS tag: GenericSetup-0_9
- Initial version, cut down from CMFSetup-1.5.3

5.6 GenericSetup Credits

- Martijn Pieters wrote the original version of this software while working at Zope Corporation. He developed his version as part of the "Bonzai" CMS which Zope corp. built for Boston.com using its Zope4Media Print product.
- Tres Seaver factored the code out of the Bonzai CMS, first into a `CMFSetup` package, and later into the current form, with no dependencies on CMF products.
- Yvo Schubbe refactored the XML export / import support mercilessly, and has made numerous bugfixes and enhancements to the product.
- Jens Vagelpohl has made numerous bug fixes and enhancements to the product.
- The package is now maintained by the CMF developers, with lots of input from the Plone developers.

5.7 Glossary

site The instance in the Zope URL space which defines a "zone of service" for a set of tools.

dotted name The Pythonic representation of the "path" to a given function / module, e.g. `Products.GenericSetup.tool.exportToolset`.

profile A "preset" configuration of a site, defined on the filesystem or in a tarball.

base profile Profiles which represent the entire configuration of a given site, and have no dependencies.

extension profile Profile fragments are used to modify a site created from a given *base profile*. They can be shipped with add-on products or used for customization steps. Importing an *extension profile* adds or overwrites existing settings in a fine-grained way. An *extension profile* cannot be exported.

snapshot A profile which captures the state of the site's configuration at a point in time (e.g., immediately after creation of the site, or after importing an *extension profile*.

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Index

B

base profile, **26**

D

dotted name, **26**

E

extension profile, **26**

P

profile, **26**

S

site, **26**
snapshot, **26**