



Documentation and Upgrade Guide for Plone DBFlay

*for
Avtech Electrosystems Ltd.*

May 30, 2008

Rua Rosalvo de Almeida, 5
4710-429 Braga
Portugal

Tel: +351 253 257 395
Fax: +351 253 257 396

<http://eurotux.com>

Executive Summary

This document includes documentation about the project for the new release of Plone DBFlay. It includes a description of the major changes and also an upgrade guide.

1 Major Changes

The main goal of this project was to include a set of improvements on *PloneDBFlay*, a product that allows a non-technical user to perform data management operations in a relational database (PostgreSQL).

1.1 Archetypes

The previous version of PloneDBFlay (0.3.3), compatible with Plone-2.5, had an implementation based on CMF-styled content types. The new version developed included a re-implementation of the content type `DBConfig`, now based on the Archetypes¹ framework.

Archetypes allows us to define the schema of a content type in a *pythonic* declarative way. The following is an example of the new schema, including some of the fields of `DBConfig`:

```
DBConfigSchema = ATFolderSchema.copy() + Schema((
    StringField(
        'host',
        widget = StringWidget(
            label = 'Server hostname',
            description = 'Database server',
        ),
        schemata = 'database',
        mutator = 'set_host',
        required = True,
    ),
    StringField(
        'dbname',
        widget = StringWidget(
            label = 'Dbname',
            description = 'Database name',
        ),
        schemata = 'database',
        mutator = 'set_dbname',
        required = True,
    ),
    StringField(
        'user',
```

¹<http://plone.org/products/archetypes>

```

        widget = StringWidget(
            label = 'Username',
            description = 'Database username',
        ),
        schemata = 'database',
        mutator = 'set_user',
        required = True,
    ),
    ...
))

```

Accessors and mutators are also generated by Archetypes. Some of the mutators needed to keep their implementations in order to be consistent with the former API (e.g. `set_query_fields`, `set_key_fields`, ...). The class definition is also simplified (with the need to enter the factory type information):

```

class DBConfig(ATFolder):
    """ A content type that holds information about an
        SQL connection. It performs queries against
        the database.
    """

    portal_type = meta_type = 'DBConfig'
    archetype_name = 'DB Config'

    implements(IDBConfig, INonStructuralFolder)

    schema = DBConfigSchema
    _at_rename_after_creation = True

    security = ClassSecurityInfo()

    ...

```

1.2 GenericSetup Profile

The new version includes new GenericSetup profiles to perform the basic configuration needed in a Plone-3.x site. GenericSetup allows the configuration of a set of steps that will setup the necessary tools and utilities in the Plone site. For example, the `DBConfig` factory type information can be defined the following way:

```
<?xml version="1.0"?>
```

```
<object name="DBConfig"
  meta_type="Factory-based Type Information with dynamic views"
  i18n:domain="plone"
  xmlns:i18n="http://xml.zope.org/namespaces/i18n">
  <property name="title" i18n:translate="">DBFlay Config
</property>
  <property name="description"
    i18n:translate="">DBFlay configuration.</property>
  <property name="content_icon">dbconfig.gif</property>
  <property name="content_meta_type">DBConfig</property>
  <property name="product">DBConfig</property>
  <property name="factory">addDBConfig</property>
  <property name="immediate_view">view</property>
  <property name="global_allow">True</property>
  <property name="filter_content_types">True</property>
  <property name="allowed_content_types"/>
  <property name="allow_discussion">False</property>
  <property name="default_view">base_view</property>
  <property name="view_methods">
    <element value="base_view"/>
  </property>
  ...
</object>
```

There are three profiles included:

- `Products.DBConfig:default` – The default profile. It's the one that should be used for normal configurations.
- `Products.DBConfig:examplecontent` – A profile that creates some example content (assuming some connections and database servers are available). Used mainly by unit tests.
- `Products.DBConfig:upgrade` – A profile that includes steps to upgrade old instances of `DBConfig` on sites that have been upgraded to Plone-3.x.

1.3 Component Architecture

The new components developed for this product all to use the Zope Component Architecture (C.A.) to achieve a certain level of modularity. The usage of Zope3 interfaces,

browser views and adapters will make easier the development of future enhancements for the product.

1.4 Small Improvements and Fixes

During the implementation, some small improvements and fixes were added to the product:

- Definition of required fields in the schema.
- Creation of schemata/fieldsets, separating fields in different pages, according to their type (database, fields, transformations).
- Regex fields now use `DataGridField` allowing to define the regex substitution pattern in different input boxes, without the need to escape all slashes.

1.5 Support for Other Databases

The support for other databases (e.g. MySQL) is now possible due to the modularity of the new components defined. To support a new database, two new adapters must be implemented providing interfaces `ISQLConnectionDatabaseAdapter` and `ISQLQueryProcessor`. The current version of Plone DBFlay includes the support for the following databases:

- PostgreSQL.
- MySQL.
- ZGadfly (only for testing purposes).

1.6 Dependencies

There are new dependencies for version 0.4 of `PloneDBFlay`:

- Plone-3.0 +
- `psycopg2` (tested with version 2.0.7)
- `egenix-mx-base` (tested with version 3.0)

2 Upgrade Guide

- ZPscopgDA (distributed with psycopg2)
- DataGridField 1.5.0 +
- ZMySQLDA 2.0.8 (optional, for MySQL support)
- pymysql 1.2.0 (optional, for MySQL support)

1.7 Tests

The development process was based on a test-driven development (TDD) approach. Product `DBConfig` now includes a set of unit tests, in order to make sure the main feature are actually working, now and in future releases. To run the tests for `DBConfig`, go to the zope instance home and type:

```
$ bin/zopectl test -s Products.DBConfig
```

The product `README.txt` also includes information about the steps to run the unit tests. To fully test the product features, it is also necessary the configuration of an example database.

2 Upgrade Guide

This section describes the main steps to upgrade a live Plone site to the new version of PloneDBFlay.

NOTE: The instructions included in this section are related to a regular Zope installation, and may not be the most appropriated for instances with specific configurations, that may invalidate some of the commands described. The examples included are assuming that the site is available the address `http://mysite.com:8080/mysite` and the Zope Management Interface (ZMI) is available at `http://mysite.com:8080/manage`.

2.1 Backup

Before starting the upgrade, a full backup should be made.

- Stop Zope.
- Copy the database file (`Data.fs`):

```
$ cd var/  
$ cp Data.fs Data.fs.`date --iso`
```

Note: Copying the database should be enough but, according to the target installation specifics, it may be necessary to backup the entire instance home directory.

- Start Zope.

2.2 Upgrade Plone

It is assumed that the target site for the upgrade is already running an up-to-date Plone version (3.x). If not, the new Plone version should be installed and the upgrade steps should be ran, through the ZMI, at the `portal_migrations` tool:

- Run Go to `http://mysite.com/mysite/manage` click `portal_migrations`, check the box “Dry run mode” and click “Upgrade”.
- Confirm that the dry run went well, by checking the results page. In the final of the results page there should be the following information:

```
...  
- Your ZODB and Filesystem Plone instances are now up-to-date.  
- Dry run selected, transaction aborted.
```

- If the test succeeded, than you may repeat the process, without selecting “dry run”. Then you should get the results message:

```
...  
- Your ZODB and Filesystem Plone instances are now up-to-date.
```

For more information about the process for migration between Plone versions, read the online Plone upgrade guide, more specifically the section related to upgrades from versions 2.5 and 3.0 (<http://plone.org/documentation/manual/upgrade-guide/version/2.5-3.0>).

2.3 Install Dependencies

There are two types of dependencies that must be installed before continue to the installation of DBConfig-0.4: Zope python dependencies and product dependencies.

To install the python dependencies (egenix-mx-base, psycopg2, pymysql), download the packages and follow the instructions to perform the installation in the same python used by Zope.

NOTE: The following instructions are assuming that `INSTANCE_HOME` is the directory where your zope instance is installed.

2.3.1 Egenix-mx-base

The following are the steps to install egenix-mx-base on your instance:

- Download the package from <http://www.egenix.com/products/python/mxBase/>
- Build and install the package in the python installation used by Zope:

```
$ tar zxvf egenix-mx-base-3.0.0.tar.gz
$ cd egenix-mx-base-3.0.0
$ INSTANCE_HOME/bin/python setup.py build
$ INSTANCE_HOME/bin/python setup.py install
```

2.3.2 Psycopg2

The following are the steps to install psycopg2 on your instance:

- Download the package from <http://initd.org/pub/software/psycopg/PSYCOPG-2-0/>
- Build and install the package in the python installation used by Zope:

```
$ tar zxvf psycopg2-2.0.7.tar.gz
$ cd psycopg2-2.0.7
$ INSTANCE_HOME/bin/python setup.py build
$ INSTANCE_HOME/bin/python setup.py install
```

2.3.3 PyMySQL

The following are the steps to install pymysql on your instance:

- Download the package from <http://sourceforge.net/projects/mysql-python>
- Build and install the package in the python installation used by Zope:

```
$ tar zxvf MySQL-python-1.2.2.tar.gz
$ cd MySQL-python-1.2.2
$ INSTANCE_HOME/bin/python setup.py build
$ INSTANCE_HOME/bin/python setup.py install
```

2.3.4 Product Dependencies

To install the product dependencies, download the packages from the following locations:

- <http://plone.org/products/datagridfield>
- <http://www.zope.org/Members/adustman/Products/ZMySQLDA>

The product ZPsyncpgDA, can be obtained inside the psycpg2 package. Then just unpack the product packages in the Zope products directory:

```
$ cd INSTANCE_HOME/Products
$ tar zxvf DataGridField-1.5.0.tgz
$ tar zxvf ZPsyncpgDA-2.0.7.tgz
$ tar zxvf ZMySQLDA-2.0.8.tgz
```

Then restart Zope and ensure all the new products are available.

2.4 Install DBConfig 0.4

The installation of “DBConfig” should be made by unpacking the product tarball to the Zope products directory, just like the installation of the other products.

```
$ cd INSTANCE_HOME/Products
$ bin/zopectl stop
$ tar xzvf DBConfig-0.4.tgz
$ bin/zopectl start
```

The installation in Plone can be made in three ways:

- For a fresh Plone site, select the extension profile “DBConfig” at the “add Plone site” form.
- In a live Plone site, go to http://mysite.com/mysite/portal_setup/manage_importSteps, select the extension profile “DBConfig” and click in “Import all steps”, at the bottom of the page.
- This same procedure can also be made through the Plone control panel. At the “Add-on Products” form (http://mysite.com/mysite/prefs_install_products_form), select “DBConfig” and click “Install” (“Reinstall”, if a previous version of the product is already installed).

2.5 Upgrade DBConfig

To upgrade an existing installation of “DBConfig”, in a live Plone site, the following steps should be followed:

- Install the new version of “DBConfig” (as explained in the previous section, for live Plone sites). Go to http://mysite.com/mysite/portal_setup/manage_importSteps, select the extension profile “DBConfig” and click in “Import all steps”, at the bottom of the page.
- Run the migration steps. Go to http://mysite.com/mysite/portal_setup/manage_importSteps, select the extension profile “DBConfig Upgrade”, select the step “DBConfig upgrade old instances” and click “Import selected steps”.

WARNING: Before running the upgrade steps, remember to **perform a full backup!**