almir Documentation

Release 0.1

Domen Kožar

April 14, 2012

CONTENTS

Author Domen Kožar <domen@dev.si>

Source code github.com project

Bug tracker github.com issues

Generated April 14, 2012

License GPL v3

Version 0.1

Features

- supports bacula-director version 5.x.x
- supports sqlite, postgresql, mysql databases
- complete read-only interface for bacula-director
- interactive web console frontend to bconsole
- export data to excel, pdf and clipboard
- supports python2.6 and python 2.7
- 100% test coverage

Overview

Almir is a bacula web interface for administrators written in Python. It is designed with simplicity in mind, although backup management is never simple.
It is named after Almir Karič, a fellow developer from **Slovenia**, whose dream was to live in San Francisco. Two years after he moved, he died in a car crash. This is his gift for eternity.
Almir is open source software licensed under **GPL v3**, started by Domen Kožar in 2011.

CHAPTER

ONE

USER GUIDE

1.1 Demo

http://almir-demo.domenkozar.com/

1.2 Design goals

- each release is pinned to bacula-director specific major version
- simple is better than complicated
- inline documentation
- convention over configuration
- can act as view interface only (optional configuration functionality)
- plugin into existing bacula instance
- total control of bacula through existing bacula api

1.3 Installation

Almir will install everything inside one directory, which must be empty. Application is meant to be self-contained, meaning no additional administrator is needed besides upgrading to a newer version. Almir should be always installed on a system together with bacula-director.

1.3.1 Prerequisities

- bacula is installed and bacula-director is running
- installed python2.6 or python2.7
- using postgresql: make sure *postgresql.conf* includes a line *client_encoding = utf8*

1.3.2 Installer (interactive)

Install prerequesities (Debian based):

```
$ sudo apt-get install git bacula-console python-distribute
```

Note: Installer will ask you few questions about SQL database and configuration for bconsole.

Install almir (recommended: under same user as bacula):

```
$ cd /some/empty/directory/to/install/almir/
$ sh -xec "$(wget -O - https://raw.github.com/iElectric/almir/master/install_production.sh)"
```

You can continue with configuring *Configuring Nginx as frontend*.

1.3.3 Manual (not recommended)

TODO ;)

1.3.4 Configuring Nginx as frontend

You would normally put this in /etc/nginx/sites-enabled/

```
server {
    listen          80;
    server_name     almir.mywebsite.com;

    location / {
            proxy_pass http://localhost:2500;

            proxy_set_header  X-Real-IP  $remote_addr;
            proxy_set_header  X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header  Host $http_host;
            proxy_redirect off;

            # optional authentication - recommended
            auth_basic "Restricted";
            # how to place and write htpasswd: http://wiki.nginx.org/HttpAuthBasicModule#auth_basic_u
            auth_basic_user_file htpasswd;

    }
}
```

Then run:

```
$ /etc/init.d/nginx reload
```

Now try to access http://almir.mywebsite.com/

1.4 Upgrading to a newer release

Currently it's best to just reinstall. In future, this will be easy as running a command.

1.5 Reporting bugs

- include bacula-director version, operating system and browser version
- include screenshot if it provides any information
- pastebin $ALMIR_ROOT/var/logs/almir-stderr*

1.6 Filesystem structure

TODO ;)

CHAPTER

TWO

DEVELOPER GUIDE

2.1 Setup developer environment

- sudo aptitude install git gcc python-dev bacula-console
- git clone https://github.com/iElectric/almir.git almir
- cd almir
- cp buildout.d/templates/buildout.cfg.in buildout.cfg
- vim buildout.cfg # configure variables
- python bootstrap.py
- bin/buildout
- bin/pserve –reload development.ini

2.2 Running Python tests

Easy as:

```
$ bin/nosetests
```

By default it will run against sqlite fixture, you can also tell nosetests to use mysql fixture (you need to import sql manually for now):

```
$ DATABASE="mysql" bin/nosetests
```

Or just specify sqlalchemy engine:

```
$ ENGINE="sqlite:////var/lib/bacula/bacula.db" bin/nosetests
```

2.3 Running Javascript tests

Install and configure phantomjs (webkit headless testing):

```
$ sudo apt-get install libqtwebkit-dev
$ git clone git://github.com/ariya/phantomjs.git && cd phantomjs
$ qmake-qt4 && make
$ sudo cp bin/phantomjs /usr/local/bin/
```

Run tests:

```
$ cd ../almir
$ ./.travis_qunit_tests.sh
```

2.4 Coding conventions

- PEP8 except for 80 char length rule
- add changelog, test and documentation with code in commits
- same applies to javascript
- jslinted javascript

2.5 Releasing `almir`

```
$ vim almir/__init__.py  # bump __version__
$ bin/mkrelease -d pypi
$ git tag <version>
$ git checkout latests
$ git merge master
```

CHAPTER

THREE

CHANGELOG

3.1 0.1 (2012/03/xx)

- Initial version

CHAPTER

FOUR

SOURCE DOCUMENTATION

4.1 almir – Main package

4.1.1 almir.forms – HTML forms definitions

4.1.2 almir.meta – Configuration for models

class almir.meta.**ModelMixin**
Bases: object

static format_byte_size(*size*)

classmethod get_list(***kw*)

classmethod get_one(*id_=None*, *query=None*)

query = None

static render_distance_of_time_in_words(*dt_from*, *dt_to=None*)

almir.meta.**get_database_size**(*engine*)
Returns human formatted SQL database size.

Example: 3.04 GB

almir.meta.**initialize_sql**(*settings*)

4.1.3 almir.models – SQLAlchemy models

4.1.4 almir.views – Pyramid views

4.1.5 almir.lib – Non MVC code

almir.lib.bacula_base64 – Bacula custom base64 implmentation

Taken from https://github.com/ZungBang/baculafs/blob/master/baculafs/Base64.py under GPLv3

almir.lib.bacula_base64.**decode_base64**(*base64*)
Bacula specific implementation of a base64 decoder

almir.lib.bconsole – Python interface to bconsole

class almir.lib.bconsole.**BConsole**(*bconsole_command='bconsole -n -c %s'*, *config_file=None*)
Bases: object

Interface to bconsole binary

classmethod from_temp_config(**args*, ***kwds*)
Constructs BConsole object with help of passing temporary file for the session.

get_upcoming_jobs()

is_running()

send_command_by_polling(*command*, *process=None*)

start_process()

almir.lib.console_commands – Parsed list of bconsole commands

almir.lib.sqlalchemy_custom_types

class almir.lib.sqlalchemy_custom_types.**BaculaDateTime**(**args*, ***kwargs*)
Bases: sqlalchemy.types.TypeDecorator

Changes sqlite DateTime to parse 0 values as no value. Also converts to right timezone

impl
alias of DateTime

process_result_value(*value*, *dialect=None*)

result_processor(*dialect*, *coltype*)

almir.lib.sqlalchemy_declarative_reflection

class almir.lib.sqlalchemy_declarative_reflection.**DeclarativeReflectedBase**
Bases: object

classmethod prepare(*engine*)
Reflect all the tables and map !

almir.lib.sqlalchemy_lowercase_inspector

class almir.lib.sqlalchemy_lowercase_inspector.**LowerCaseInspector**(*bind*)
Bases: sqlalchemy.engine.reflection.Inspector

Implements reflection inspector that reflects everything lowercase

get_columns(**a*, ***kw*)

get_foreign_keys(**a*, ***kw*)

get_indexes(**a*, ***kw*)

get_pk_constraint(**a*, ***kw*)

almir.lib.utils - General utilities

almir.lib.utils.**convert_timezone**(*datetime*)
Converts datetime to timezone aware datetime.

Retrieving timezone:

- get *timezone* from .ini settings
- default to system timezone

almir.lib.utils.**get_jinja_macro**(*macro*)
Return actual function from a jinja2 template

almir.lib.utils.**nl2br**(*text*)

almir.lib.utils.**render_rst_section**(*filename*)
Finds filename in documentation directory and renders it to html.

almir.lib.utils.**timedelta_to_seconds**(*td*)

almir.lib.utils.**yesno**(*text*)

4.1.6 almir.scripts – Runnable scripts package

almir.scripts.configure_deploy – Ask few questions and configure almir

Dirty script to output buildout.cfg, but it does the job.

almir.scripts.configure_deploy.**ask_question**(*question*, *default=None*, *validator=None*, *func=<built-in function raw_input>*)

almir.scripts.configure_deploy.**main**()
Entry point of this script

almir.scripts.configure_deploy.**validate_engine**(*v*)

almir.scripts.configure_deploy.**validate_int**(*v*)

almir.scripts.configure_deploy.**validate_open_port**(*v*)

almir.scripts.configure_deploy.**validate_timezone**(*v*)

almir.scripts.parse_console_commands – Parse help commands from bconsole source

almir.scripts.parse_console_commands.**main**()

almir.scripts.parse_console_commands.**parse_console_commands**(*source*)

4.1.7 almir.tests – Tests package

almir.tests.test_functional – Functional tests

CHAPTER

FIVE

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

PYTHON MODULE INDEX

a