ERPBlok Documentation

Release 0.0.1

Jean-Sébastien Suzanne

June 22, 2016

Contents

ERPBlok provide a way to manage business applications on top of Anyblok adding a Graphic User Interface, user system managment...

AnyBlok is released under the terms of the *AGPL v3*.

Contents

CHAPTER 1

Front Matter

Information about the ERPBlok project.

1.1 Project Homepage

ERPBlok is hosted on github - the main project page is at https://github.com/ERPBlok/ERPBlok. Source code is tracked here using GIT.

1.2 Project Status

AnyBlok is currently in dev status and is expected to be fairly stable. Users should take care to report bugs and missing features on an as-needed basis. It should be expected that the development version may be required for proper implementation of recently repaired issues in between releases; the latest master is always available at http://bitbucket/jssuzanne/erpblok/get/default.tar.gz. or http://bitbucket.org/jssuzanne/erpblok/get/default.zip

1.3 Installation

Installation via source distribution is via the `setup.py` script:

```
python setup.py install
```

Installation will add the `AnyBlok` and `AnyBlok / Pyramid` and `gunicorn` commands to the environment.

1.4 Unit Test

Run the framework test with `nose`:

```
pip install nose
nosetests erpblok/tests
```

Run all the installed bloks:

```
anyblok_nose -c config.file.cfg
```

Run the blok tests at the installation:

```
anyblok_updatedb -c config.file.cfg --install_bloks myblok --test-blok-at-install
```

1.5 Dependencies

AnyBlok works with **Python 3.2** and later. The install process will ensure that AnyBlok, AnyBlok / Pyramid are installed, in addition to other dependencies. The latest version of them is strongly recommended.

1.6 Contributing (hackers needed!)

ERPblok is at a very early stage, feel free to fork, talk with core dev, and spread the word!

1.7 Author

Jean-Sébastien Suzanne

1.8 Contributors

Anybox team:

- Jean-Sébastien Suzanne
- Pierre Verkest

1.9 Bugs

Bugs and feature enhancements to AnyBlok should be reported on the Issue tracker.

CHAPTER 2

Your first ERPBlok's installation

The goal is to have a running ERPBlok on the system.

2.1 Build the environment and run the server

1. Create a virtualenv:

```
pyvenv-3.5 erpblok-env
```

2. Install ERPBlok (not work yet, waiting the first release):

```
erpblok-env/bin/pip install erpblok
```

2.2) Install ERPBlok from vcs:

```
hg clone ssh://hg@bitbucket.org/jssuzanne/erpblok
erpblok-env/bin/python erpblok/setup.py
```

3. Choose the BDD

> **Warning:** In my case, I choose to use `postgresql`. Read the installation guides

```
erpblok-env/bin/pip install psycopg2
```

4. Write a logging configuration file. (example: mylogger.cfg):

```
[loggers]
keys=root, anyblok, erpblok, anyblok_pyramid

[handlers]
keys=consoleHandler

[formatters]
keys=consoleFormatter

[logger_root]
level=INFO
handlers=consoleHandler

[logger_anyblok]
level=INFO
handlers=consoleHandler
```

```
qualname=anyblok
propagate=0

[logger_anyblok_pyramid]
level=INFO
handlers=consoleHandler
qualname=anyblok_pyramid
propagate=0

[logger_erpblok]
level=INFO
handlers=consoleHandler
qualname=erpblok
propagate=0

[handler_consoleHandler]
class=StreamHandler
level=INFO
formatter=consoleFormatter
args=(sys.stdout,)

[formatter_consoleFormatter]
class=anyblok.logging.consoleFormatter
format=%(database)s:%(levelname)s - %(name)s:%(message)s
datefmt=
```

5. Write a configuration file for ERPBlok. (example: myerpblok.cfg):

```
[AnyBlok]
# select a default database name
db_name = erpblok
# AnyBlok load at the postgresql BDD, with the default parameter
# - unix socket: 5432
# - default user / password
db_driver_name = postgresql
# define a dirstory to save the session's file
beaker.session.data_dir = erpblok.session
# Use the logging configuration
logging_configfile = mylogger.cfg
# Gice access to the database manager for create or drop database.
allow_database_manager = true
# Define a database manager password
db_manager_password = mysecretpassword
# filter the database seen by erpblok
db_filter = %%erpblok%%
```

6. Start the gunicorn server:

```
erpblok-env/bin/gunicorn_anyblok_pyramid --anyblok-configfile myerpblok.cfg -b localhost:8080 -w
==> AnyBlok Load init: EntryPoint.parse('load_config = erpblok:load_config')
==> Load config file '/Library/Application Support/AnyBlok/conf.cfg'
==> Load config file '/Users/jssuzanne/Library/Application Support/AnyBlok/conf.cfg'
==> Load config file '/Users/jssuzanne/erpblok/erpblok/myerpblok.cfg'
==> [2016-03-17 10:14:29 +0100] [20234] [INFO] Starting gunicorn 19.4.5
==> [2016-03-17 10:14:29 +0100] [20234] [INFO] Listening at: http://127.0.0.1:8080 (20234)
==> [2016-03-17 10:14:29 +0100] [20234] [INFO] Using worker: sync
==> [2016-03-17 10:14:29 +0100] [20237] [INFO] Booting worker with pid: 20237
==> No database:INFO - anyblok.blok:BlokManager.load
==> [2016-03-17 10:14:29 +0100] [20238] [INFO] Booting worker with pid: 20238
```

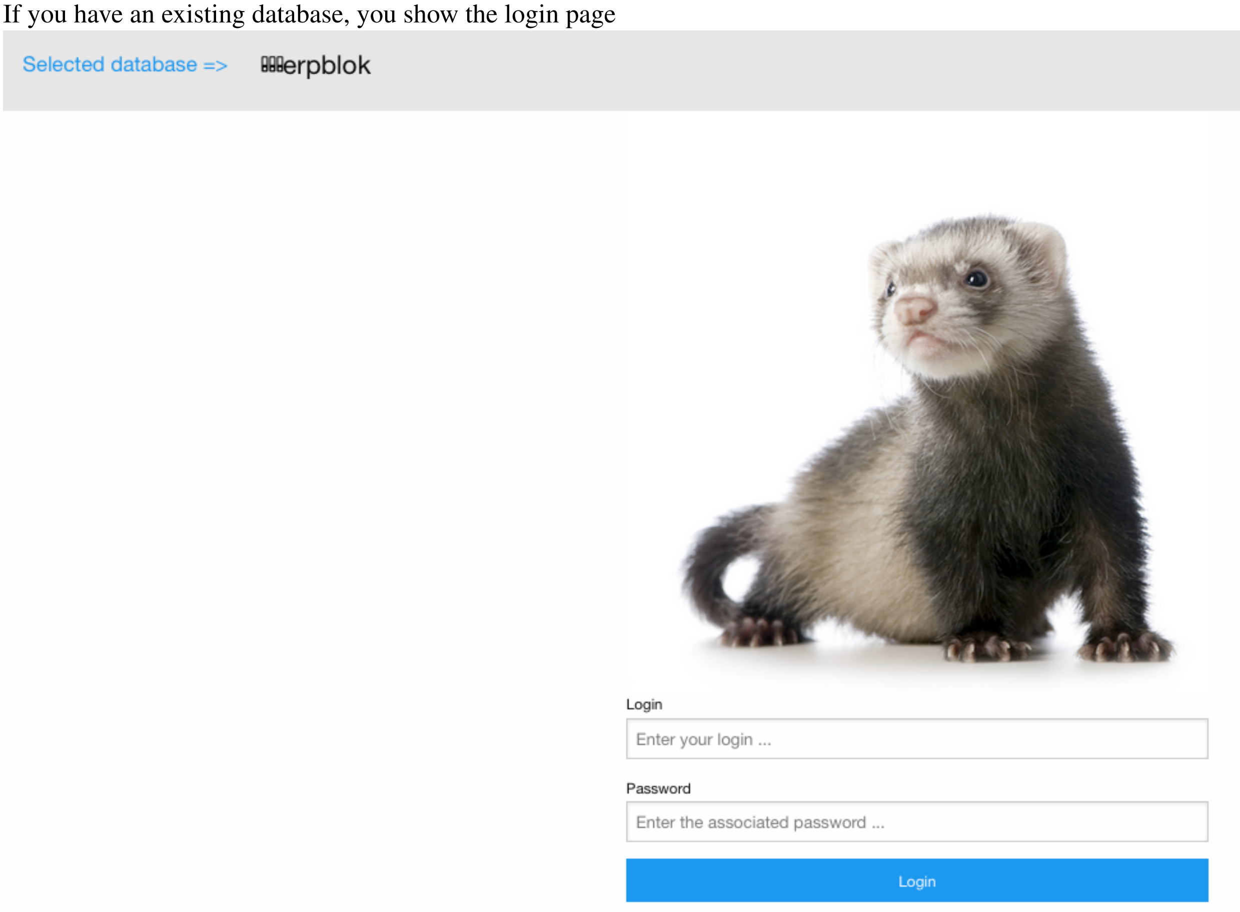
```
==> No database:INFO - anyblok.blok:BlokManager.load
==> [2016-03-17 10:14:29 +0100] [20239] [INFO] Booting worker with pid: 20239
==> No database:INFO - anyblok.blok:BlokManager.load
==> [2016-03-17 10:14:29 +0100] [20240] [INFO] Booting worker with pid: 20240
==> No database:INFO - anyblok.blok:BlokManager.load
==> No database:INFO - anyblok_pyramid.common:Preload the databases : erpblok
==> No database:INFO - anyblok_pyramid.common:Preload the database : 'erpblok'
==> No database:WARNING - anyblok_pyramid.common:The database 'erpblok' does not exist
==> No database:INFO - anyblok_pyramid.common:Preload the databases : erpblok
==> No database:INFO - anyblok_pyramid.common:Preload the database : 'erpblok'
==> No database:WARNING - anyblok_pyramid.common:The database 'erpblok' does not exist
==> No database:INFO - anyblok_pyramid.common:Preload the databases : erpblok
==> No database:INFO - anyblok_pyramid.common:Preload the database : 'erpblok'
==> No database:WARNING - anyblok_pyramid.common:The database 'erpblok' does not exist
==> No database:INFO - anyblok_pyramid.common:Preload the databases : erpblok
==> No database:INFO - anyblok_pyramid.common:Preload the database : 'erpblok'
==> No database:WARNING - anyblok_pyramid.common:The database 'erpblok' does not exist
```

Congrats your have start your own ERPBlok's server

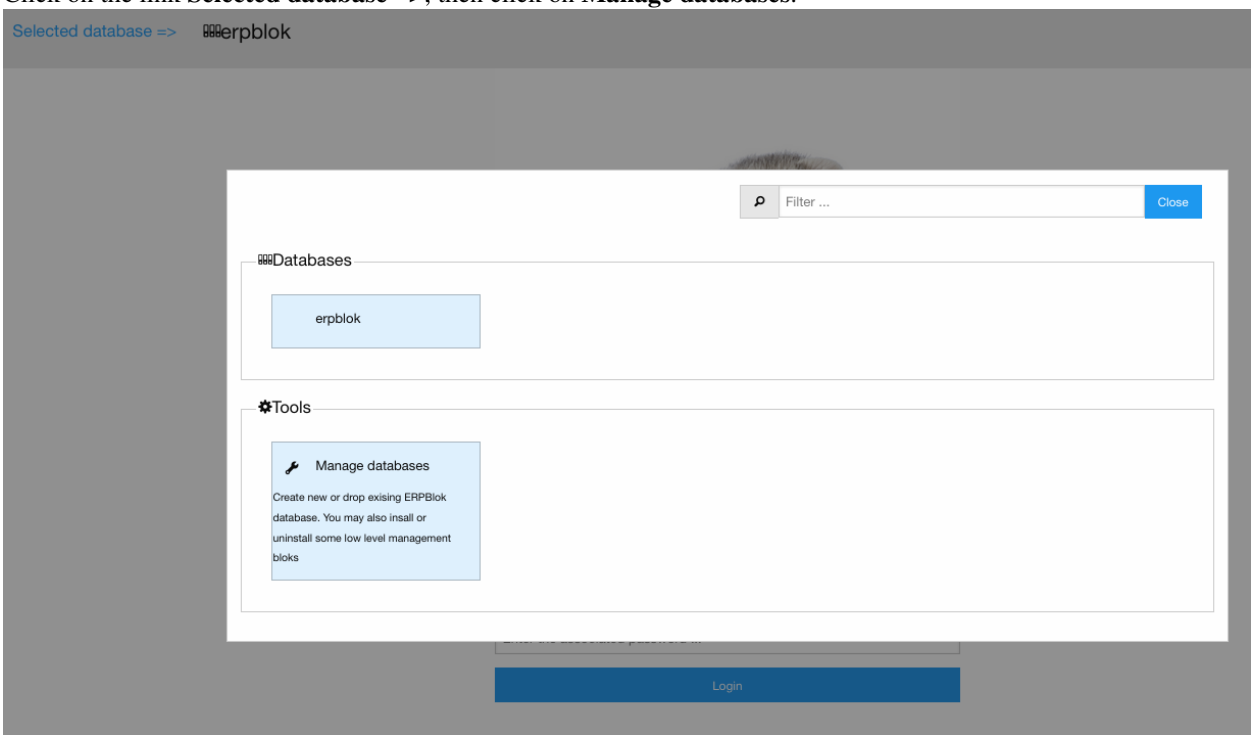
2.2 Create your first database

1. In your favorite browser call the url `localhost:8080`

If you have an existing database, you show the login page

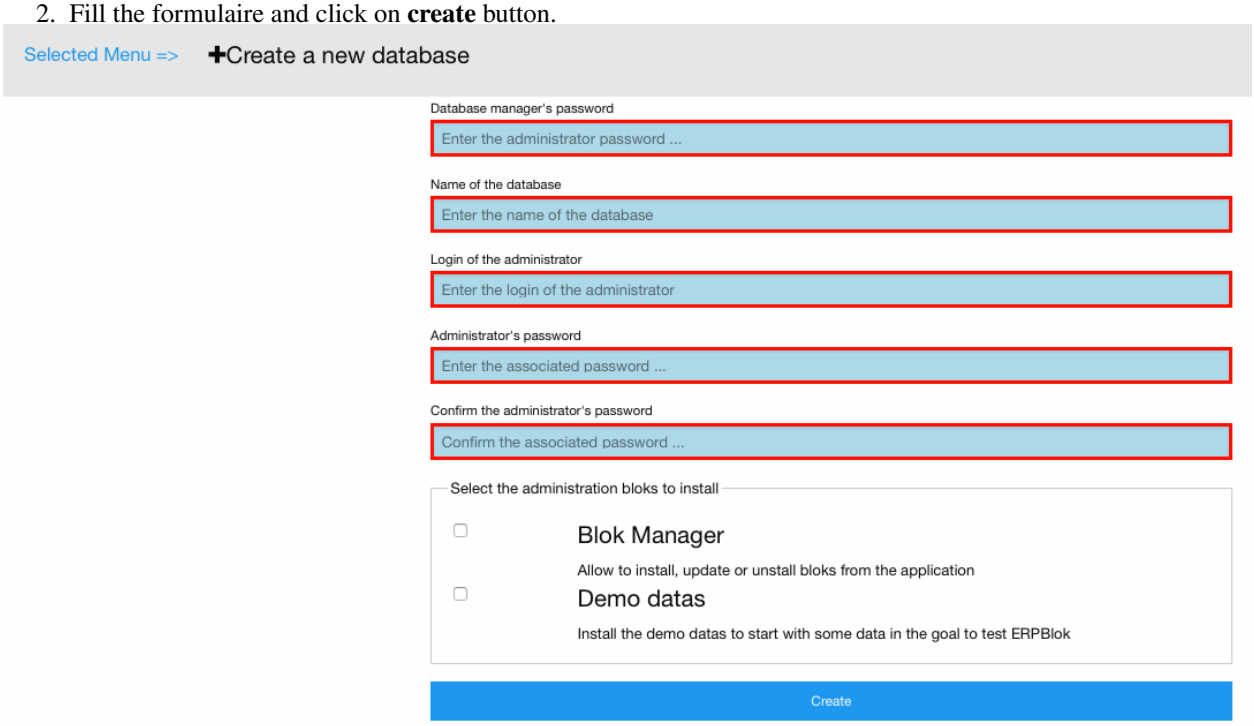


Click on the link **Selected database =>**, then click on **Manage databases**.



Else you will redirect to database manager page.

2. Fill the formulaire and click on **create** button.



Warning: It is may take some minute, but it is not display.

Congrats you have your first ERPBlok's base !!!

Contents

CHAPTER 3

ERPBlok framework

3.1 Homepage controller

erpblok.client.homepage.**get_homepage**(*request*)
Redirect the homepage to the good page

if connected, redirect to the application page if databases found, redirect to the login page else redirect to the database manager

3.2 Client controller

erpblok.client.web.**load_client**(*request*)
Return the client main page

3.3 Login controllers

erpblok.client.login.**get_login**(*request*, *database=None*)
Display the login page

erpblok.client.login.**get_login_logo**(*request*)
Return the logo for thelogin page

erpblok.client.login.**get_databases**(*request*)

erpblok.client.login.**post_login_connect**(*request*)
Log the user, if the login and password are right

Return type redirection if login/password is right else HTTPUnauthorized

erpblok.client.login.**post_login_disconnect**(*request*, *database=None*)
Logout the current user and do a redirect to the login page

3.4 Database manager controllers

erpblok.client.database.**check_allow_database_manager**()
raise an execption if the database manager is unactive :exception: pyramid.httpexceptions.HTTPNotFound

erpblok.client.database.**check_db_manager_password**(*password*)

erpblok.client.database.**get_database**(*request*)
Return the main page of the database manager

erpblok.client.database.**get_menus**(*request*)

erpblok.client.database.**get_addons**(*request*)

erpblok.client.database.**get_databases**(*request*)

erpblok.client.database.**post_create_database**(*request*)
Create a new database, and initialize it

Return type Redirection to the client

erpblok.client.database.**post_drop_database**(*request*)
Drop the database

erpblok.client.database.**post_list_database**(*request*)
Return the html of the select node with the list of the database

3.5 Common functions

erpblok.client.common.**list_databases**()
return the name of the databases found in the BDD

the result can be filtering by the Configuration entry `db_filter`

..warning:

```
    For the moment only the ``prostgresql`` dialect is available
```

Return type list of the database's names

erpblok.client.common.**create_database**(*database*)
Create a new database, initialize it and return an AnyBlok registry

Param database's name

Return type AnyBlok registry instance

erpblok.client.common.**drop_database**(*database*)
Close the registry instance of the database and drop the database

Param database's name

erpblok.client.common.**login_user**(*request*, *database*, *login*, *password*, *user_id*)
Log the user

The informations of the user are saved in the request if the user is found by is login and is password.

Parameters

- **database** – the database where the user want to be connected
- **login** – user login
- **password** – user password

Type boolean, True if the user is founed else False

erpblok.client.common.**logout**(*request*)
Remove the user information of the login

erpblok.client.common.**format_static**(*blok*, *static_url*)
Replace the attribute #BLOK by the real name of the blok

Parameters

- **blok** – the blok's name
- **static_url** – the url to format

Return type str, formated url

erpblok.client.common.**get_static**(*static_type*)
Get in the Blok definition the static data from the client

Parameters **static** – entry to read: css, js, ...

Return type list of str

erpblok.client.common.**get_templates_from**(*attr*)

3.6 Template definition module

exception erpblok.client.template.**TemplateException**
Bases: Exception

with_traceback()
Exception.with_traceback(tb) – set self.__traceback__ to tb and return self.

class erpblok.client.template.**Template**(**args*, ***kwargs*)
Bases: object

html templating framework, the need is to manipulate web template.

```
tmpl = Template()
tmpl.load_file(file_pointer_1)
tmpl.load_file(file_pointer_2)
tmpl.load_file(file_pointer_3)
tmpl.load_file(file_pointer_N)
tmpl.compîle()
tmpl.get_all_template()
```

clean()
Erase all the known templates

compile()
compile all the templates

compile_template(*name*)
compile a specific template

Parameters **name** – id str of the template

decode(*element*)
Decode some element need for the web template

Parameters **element** – string representation of the element

Return type str

encode(*element*)
Encode the templating commande

Parameters **element** – string representation of the element

Return type str

get_all_template()
Return all the template in string format

get_template(*name*, *tostring=True*)
return a specific template

Parameters **name** – name of the template

Return type str

get_xpath(*element*)
Find and return the xpath found in the template

Parameters **element** – html.Element

Return type list of dict

get_xpath_attributes(*elements*)
Find and return the attibute

load_file(*openedfile*)
Load a file

File format

```
<templates>
    <template id="...">
        ...
    </template>
</templates>
```

Parameters **openedfile** – file descriptor

Exception TemplateException

load_template(*element*)
Load one specific template

Parameters **element** – html.Element

Exception TemplateException

xpath(*name*, *expression*, *mult*)
Apply the xpath

xpath_attributes(*name*, *expression*, *mult*, *attributes*)
Apply a xpath attributes:

```
<template id="..." extend="other template">
    <xpath expresion="..." action="attributes">
        <attribute key="value"/>
        <attribute foo="bar"/>
    </xpath>
</template>
```

Parameters

- **name** – name of the template
- **expresion** – xpath regex to find the good node
- **mult** – If true, xpath can apply on all the element found

Attributes attributes to apply

xpath_insert (*name*, *expression*, *mult*, *elements*)
Apply a xpath insert:

```
<template id="..." extend="other template">
    <xpath expresion="..." action="insert">
        ...
    </xpath>
</template>
```

Parameters

- **name** – name of the template
- **expresion** – xpath regex to find the good node
- **mult** – If true, xpath can apply on all the element found

Elements children of the xpath to insert

xpath_insertAfter (*name*, *expression*, *mult*, *elements*)
Apply a xpath insert:

```
<template id="..." extend="other template">
    <xpath expresion="..." action="insertAfter">
        ...
    </xpath>
</template>
```

Parameters

- **name** – name of the template
- **expresion** – xpath regex to find the good node
- **mult** – If true, xpath can apply on all the element found

Elements children of the xpath to insert

xpath_insertBefore (*name*, *expression*, *mult*, *elements*)
Apply a xpath insert:

```
<template id="..." extend="other template">
    <xpath expresion="..." action="insertBefore">
        ...
    </xpath>
</template>
```

Parameters

- **name** – name of the template
- **expresion** – xpath regex to find the good node
- **mult** – If true, xpath can apply on all the element found

Elements children of the xpath to insert

xpath_remove (*name*, *expression*, *mult*)
Apply a xpath remove:

```
<template id="..." extend="other template">
    <xpath expresion="..." action="remove"/>
</template>
```

Parameters

- **name** – name of the template
- **expresion** – xpath regex to find the good node
- **mult** – If true, xpath can apply on all the element found

xpath_replace(*name*, *expression*, *mult*, *elements*)

Apply a xpath replace:

```
<template id="..." extend="other template">
    <xpath expresion="..." action="replace">
        ...
    </xpath>
</template>
```

Parameters

- **name** – name of the template
- **expresion** – xpath regex to find the good node
- **mult** – If true, xpath can apply on all the element found

Elements children of the xpath to replace

Contents

CHAPTER 4

Bloks

4.1 Blok erpblok-core

class erpblok.bloks.erpblok_core.**ERPBlokCore**(*registry*)

Bases: anyblok.blok.Blok

Base Blok for ERPBlok

autoinstall = True

conditional_by = []

conflicting_by = []

name = 'erpblok-core'

optional_by = []

required = ['erpblok-web-client', 'anyblok-io-xml']

required_by = ['erpblok-debug', 'erpblok-demo']

update(*latest_version*)

version = '0.0.1'

views = ['views/access.tmpl']

4.2 Blok erpblok-web-client

class erpblok.bloks.erpblok_web_client.**ERPBlokWebClient**(*registry*)

Bases: anyblok.blok.Blok

Web Client for ERPBlok

autoinstall = True

client_css = ['#BLOK/static/view.css', '#BLOK/static/view_list.css']

client_js = ['#BLOK/static/underscore-min.js']

client_js_babel = ['#BLOK/static/hashtag-manager.js', '#BLOK/static/error-manager.js', '#BLOK/static/menu.js', '

client_templates = ['templates.tmpl']

conditional_by = []

conflicting_by = []

database_css = []

database_js = ['#BLOK/static/notification/notification.js']

database_js_babel = ['#BLOK/static/database.js']

database_templates = ['database_templates.tmpl']

global_css = ['#BLOK/static/foundation-6.1.2/css/foundation.min.css', '#BLOK/static/foundation-icons/foundation-ico

global_js = ['#BLOK/static/react.min.js', '#BLOK/static/react-dom.min.js', '#BLOK/static/babel-core/5.8.34/browser.

global_js_babel = ['#BLOK/static/template.js', '#BLOK/static/fields.js', '#BLOK/static/modals.js']

classmethod import_declaration_module()

load()

login_css = []

login_js = []

login_js_babel = ['#BLOK/static/url-search-manager.js', '#BLOK/static/login.js']

login_templates = ['login_templates.tmpl']

name = 'erpblok-web-client'

optional_by = []

classmethod pyramid_load_config(*config*)

classmethod reload_declaration_module(*reload*)

required = ['anyblok-core', 'anyblok-io']

required_by = ['erpblok-core']

version = '0.0.1'

This blok is required by all ERPBlok application. This blok define the main fonctionnality of the interface and the user notion.

4.2.1 Functional space

The functional space is reprented by: * Menu or not * action(s) with their view(s)

```
<record external_id="setting_space_user">
    <field name="label">User</field>
    <field name="icon">fi-results-demographics</field>
    <field name="description">Configure the users and access rules</field>
    <field name="category" external_id="setting_space_category" />
    <field name="menus">
        <record external_id="setting_menu_groups">
            <!-- define one menu for the ``Groups`` model -->
            <field name="label">Groups</field>
            <field name="action" external_id="action_group"/>
        </record>
        <record external_id="setting_menu_logins">
            <field name="label">Logins</field>
            <field name="action" external_id="action_login"/>
        </record>
        <record external_id="setting_menu_users">
```

```
            <field name="label">Users</field>
            <field name="action" external_id="action_user"/>
        </record>
    </field>
</record>
```

4.2.2 Menu

Space have often menus, this menu call action, in the same page or in a dialog box.

The menu can be hierarchical.

```
<record external_id="setting_space_low_level">
    <field name="label">Low level</field>
    <field name="icon">fi-wrench</field>
    <field name="description">Configure all the low level data</field>
    <field name="category" external_id="setting_space_category" />
    <field name="menus">
        <record>
            <field name="label">Database structure</field>
            <field name="children">
                <record>
                    <field name="label">Models</field>
                    <field name="action" external_id="action_db_model"/>
                </record>
                <record>
                    <field name="label">Fields</field>
                    <field name="action" external_id="action_db_field"/>
                </record>
                <record>
                    <field name="label">Columns</field>
                    <field name="action" external_id="action_db_column"/>
                </record>
                <record>
                    <field name="label">Relation Ships</field>
                    <field name="action" external_id="action_db_rs"/>
                </record>
            </field>
        </record>
    </field>
</record>
```

4.2.3 Action

Is attach at the space or a dialog box. The action can have one or more view(s).

```
<record external_id="action_group">
    <field name="label">Groups</field>
    <field name="model">Model.Access.Group</field>
    <field name="add_delete">0</field>
    <field name="add_new">0</field>
    <field name="add_edit">0</field>
    <field name="views">
        <record external_id="view_access_group_tree">
            <field name="selectable">1</field>
            <field name="mode">Model.UI.View.List</field>
```

```
            <field name="template">ERPBlokAccessGroupList</field>
        </record>
    </field>
</record>
```

4.2.4 View

Actuality the existing are:

List:

- Can be modifiable directly in the line or open another type of view
- Can be multi header
- On field can be display more than one time.
- they are some beaviour to help to display UI with some condition

```
<template id="ERPBlokAccessUserList">
    <field name="first_name" />
    <field name="last_name" />
</template>
```

Possible attribute:

Attribute	description
checkbox	Boolean, if the checkbox is displayed or not
inline	Boolean, if the data is modified in the same view

Form:

- Can n be modifiable directly
- On field can be display more than one time.
- they are some beaviour to help to display UI with some condition

```
<template id="ERPBlokAccessWebLoginForm">
    <div class="row">
        <div class="columns small-12 medium-9 large-6">
            <label for="login" />
            <field name="login" />
            <label for="password" />
            <field name="password"/>
        </div>
    </div>
</template>
```

thumbnail:

- Can open another type of view
- On field can be display more than one time.
- they are some beaviour to help to display UI with some condition

```
<template id="AnyBlokSystemBlokThumbnails">
    <div class="row">
        <div class="columns">
            <h4><field name="name" class="primary"/></h4>
        </div>
```

```
    </div>
    <div class="row">
        <div class="columns large-7 medium-6">
            <field name="logo" type="Picture" file_name_field="name"></field>
            <call template="AnyBlokSystemBlokButton"/>
        </div>
        <div class="columns large-5 medium-6">
            <call template="AnyBlokSystemBlokState"/>
        </div>
    </div>
</template>
```

helper:

You can use some feature for the definition of the view:

- call: include another template, do not rewrite more than one time the same template.

4.2.5 Field

Each field represent one column in the database. If a column is put two time, modify one, automaticly modify the 2nd one.

Declaration of one field

```
<field name="my_anyblok_field"/>
```

The existing field Type are:

- String
- Integer
- Boolean
- Float
- Selection
- Password
- Text
- Html
- LargeBinary
- Picture
- Many2One
- One2One:
- Many2ManyChoices

General attributes for all fields:

Attribute	description
name	Name of the anyblok field to display
type	Type of field, by default, it is the AnyBlok field
writable-only-if	take a condition: `<field name="..." writable-only-if="fields.field1 != 'foo'"/>`
visible-only-if	take a condition: `<field name="..." visible-only-if="fields.field1 != 'foo'"/>`
not-nullable-only-if	take a condition: `<field name="..." not-nullable-only-if="fields.field1" />`

placeholder |
selections |
precision |

Attributes for field: LargeBinary

Attribute	description
file_name_field	Name of the field to use to save the file name
file_size_field	Name of the field to use to save the file size
mimetype_field	Name of the field to use to save the file mimetype
accept	filtering the extension of the file in the upload box

Attributes for field: Many2One

Attribute	description
search-box-limit	Number max entry in the search select box
search-box-add	Boolean to determine if the user can create a new entry
label	field of the relationship to use to display

Attributes for field: Many2ManyChoices

Attribute	description
largegrid	Number entry by line for large screen
mediumgrid	Number entry by line for tablette
smallgrid	Number entry by line for smartphone
label	field of the relationship to use to display

Attributes for field: Text

Attribute	description
rows	Default rows number to display

4.3 Blok erpblok-demo

class `erpblok.bloks.erpblok_demo.`**`ERPBlokDemo`**(*registry*)

Bases: `anyblok.blok.Blok`

Demo blok for ERPBlok

conditional_by = []

conflicting_by = []

name = 'erpblok-demo'

optional_by = []

required = ['anyblok-io-xml', 'erpblok-core']

required_by = []

setting_blok_description = {'value': None, 'description': 'Install the demo datas to start with some data in the go

update (*latest_version*)
Update the database

version = '0.0.1'

4.4 Blok erpblok-debug

class erpblok.bloks.erpblok_debug.**ERPBlokDebug** (*registry*)
Bases: anyblok.blok.Blok

Demo blok for ERPBlok

conditional_by = []

conflicting_by = []

name = 'erpblok-debug'

optional_by = []

required = ['anyblok-io-xml', 'erpblok-core']

required_by = []

uninstall ()

update (*latest_version*)
Update the database

version = '0.0.1'

4.5 Blok erpblok-blok-manager

class erpblok.bloks.blok_manager.**ERPBlokBlokManager** (*registry*)
Bases: anyblok.blok.Blok

Blok manager for ERPBlok

conditional_by = []

conflicting_by = []

classmethod import_declaration_module ()

name = 'erpblok-blok-manager'

optional_by = []

classmethod reload_declaration_module (*reload*)

required = ['anyblok-core', 'anyblok-io-xml']

required_by = []

setting_blok_description = {'value': None, 'description': 'Allow to install, update or unstall bloks from the appli

uninstall()

update(*latest_version*)
Update the database

version = '0.0.1'

views = ['blok.tmpl']

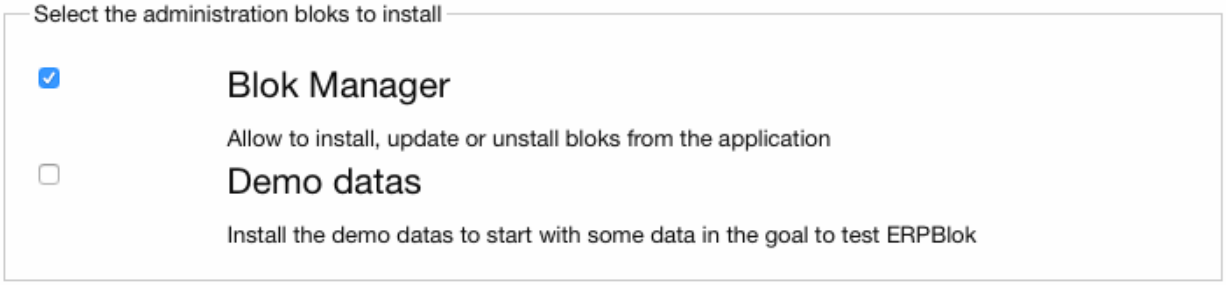
Add space to manage bloks:

- Install
- Upgrade
- Uninstall

4.5.1 Install this blok

The only way to install this blok are:

- In the database manager, when the creation of the database



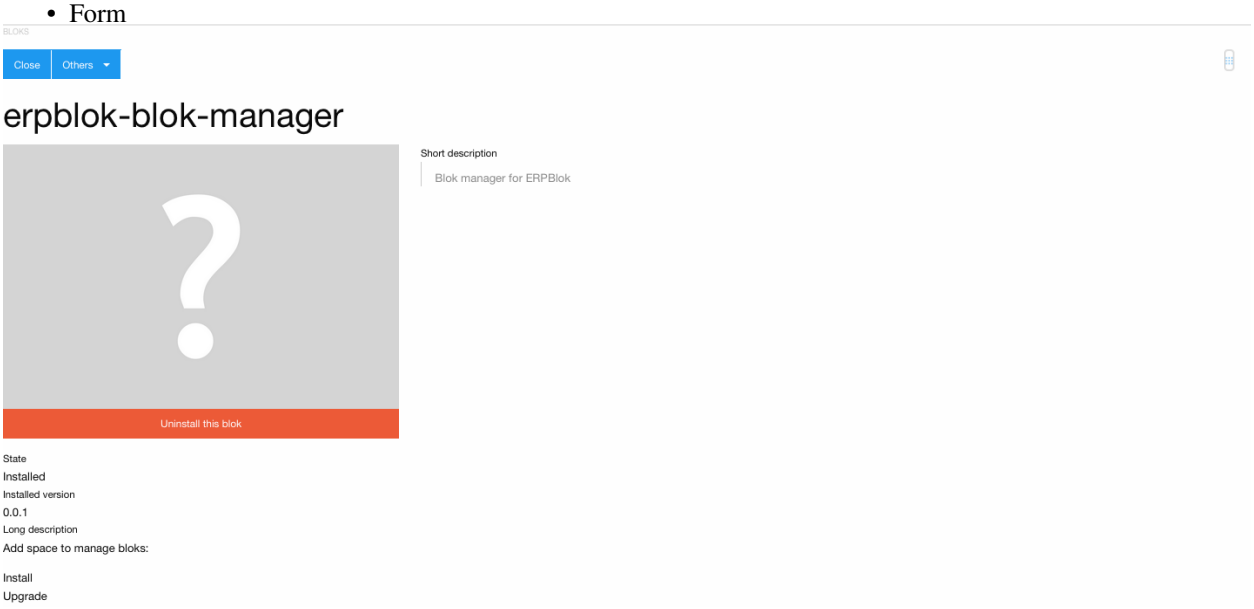
- By console script:

```
anyblok_updatedb --install-bloks erpblok-blok-manager
```

4.5.2 Views

- Thumbnails

- Form



4.5.3 API doc

Blok

class `erpblok.bloks.blok_manager.blok.`**`Blok`**

Bases: `object`

`convert_path`(*res*)

Change the path of static image

`convert_rst2html`(*rst*)

Convert a rst to html

Parameters **`rst`** – rst source

Return type html souce

get_logo()
Return the logo define in blok description

```
class MyBlok(Blok):
    logo = 'path/to/the/logo/in/blok'
```

get_long_description()
Overwrite the description to return a html

get_short_description()
Overwrite the description to return a html

install_blok()
Hight level method to install one blok

logo = <anyblok.field.Function object>

classmethod reload_blokmanager(**args*, ***kwargs*)
Reload all the bloks with their code sources

uninstall_blok()
Hight level method to uninstall one blok

upgrade_blok()
Hight level method to upgrade one blok

CHAPTER 5

CHANGELOG

5.1 Futur

CHAPTER 6

ROADMAP

- Add logo and slogan
- Update doc
- Improve the existing View types and widgets

6.1 To implement

- Internationalisation
- **View types:**
  - Diagram
  - Kanban
  - Calendar
  - Gant
  - Custom
  - ...
- **Field widgets**
  - One2Many
  - Many2Many, Many2ManyLabels
  - x2MRadio, x2MSelection, x2MDynatree
  - x2OneDynatree
  - Date, Datetime, Time, Interval
  - Radio
  - Decimal
  - JSON
  - Color
  - URL
  - Email
  - Phone

    - ...
- Access Rules / Roles

6.2 Library to include

- internationalisation: https://pypi.python.org/pypi/SQLAlchemy-i18n/0.8.2

6.3 Functionnality which need a sprint

- Import / Export
- Internalization
- Access Rules / Roles
- View types,
- Field widgets

6.4 Waiting release or fix of

6.4.1 AnyBlok

- Possibility to remove entity bind with IO.Mapping

CHAPTER 7

License

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

e

E