

Documentation for the PYROS software

COLIBRI/GCC OGS (Observatory & Observations Control Software)

Version : 25/1/18

Please display the outline for this document : menu **View/Show document outline**

1. Installation

This page describes the installation phase of the Pyros software.

1. Installation	2
1.1. COMPATIBLE PLATFORMS (TESTED)	3
1.2. GET THE PYROS SOFTWARE	3
1.2.1. Authenticate to the gitlab	3
1.2.2. Get the software	3
1.2.2.1. DYNAMIC VERSION (Developers) : Get a dynamic version (synchronized)	3
1.2.2.2. STATIC VERSION (Non developers) : Download a static version (not synchronized) :	5
1.2.3. For WINDOWS users	5
1.2.4. Miscellaneous (optional)	5
1.3. INSTALLATION OF PREREQUISITES	6
1.3.1. Prerequisite 1 : Install Python3 (3.5+) + pip + lxml	6
1.3.2. Prerequisites 2 and 3 : MySQL and RabbitMQ	9
1.3.2.1. Prerequisite 2 : Install a DataBase Management Server (DBMS)	9
1.3.2.2. Prerequisite 3 : Install RabbitMQ	12
1.4. INSTALLATION OF NEEDED PYTHON PACKAGES	15
1.4.1. Install all the needed python packages and the PyROS database	15
1.4.2. (OPTIONAL) Install the Comet python package	15
1.5. TEST	17
1.6. RUN	19
1.6.1. Access the PyROS (normal) application	19
1.6.2. Access the PyROS administration interface	20
1.6.3. Other use case (To be continued...)	20
1.6.4. Play with PyROS with the Django shell	20
1.7. DATABASE SCHEMA (v0.2.2)	22
1.8. NOTES FOR ECLIPSE USERS	23
1.9. NOTES FOR PYCHARM USERS	27
1.10. Notes about MySql (TBC)	28
1.11. MANUAL INSTALLATION OF PYTHON PACKAGES ONE BY ONE (obsolete)	29
1.11.1. (Only if using MySql) Create the database "pyros" and the pyros user	29
1.11.2. Create a Python3 virtual environment dedicated to the project	30
1.11.3. Activate the python virtual environment (from inside the project)	30
1.11.4. Install needed python packages	30

1.1. COMPATIBLE PLATFORMS (TESTED)

This software is targeted first for Linux CentOS 7 (+ Fedora and Ubuntu), but also for Mac OS X and Windows.

All these systems should run Python 3 (3.5+)

Pyros has been tested on these platforms:

- **Linux :**
 - **CentOS 7.1** (with Python 3.4)
 - Linux Mint 17.2 (== Ubuntu 14.04.3) (with Python 3.5)
 - Ubuntu 16.04 (with python 3.5.2)
- **Mac OS** 10.13 (with Python 3.6)
- **Windows** 10 (with Python 3.5)

1.2. GET THE PYROS SOFTWARE

1.2.1. Authenticate to the gitlab

In order to get this software, you must first authenticate on the IRAP gitlab

<https://gitlab.irap.omp.eu/epallier/pyros>

For this, just go to <https://gitlab.irap.omp.eu/epallier/pyros>

and either sign in with your LDAP account (if you are from IRAP),
or register via the "Sign up" form.

1.2.2. Get the software

1.2.2.1. DYNAMIC VERSION (Developers) : Get a dynamic version (synchronized)

If you do not want to contribute to this project but just want to try it, you can just download a STATIC version of it : go to next section "STATIC VERSION" below.

Windows users : you first need to get the GIT software (see below, "[For Windows users](#)")

By getting the software from git, you will get a dynamically synchronized version, which means that you will be able to update your version as soon as a new version is available (simply with the command : "git pull").

(From Eclipse : See below, section "[NOTES FOR ECLIPSE USERS](#)")

From the terminal :

```
$ git clone https://gitlab.irap.omp.eu/epallier/pyros.git PYROS
```

(or also, but not sure it works well : `git clone git@gitlab.irap.omp.eu:epallier/pyros.git PYROS`)

If you ever get this error message :

```
fatal: unable to access 'https://gitlab.irap.omp.eu/epallier/pyros.git/':  
Peer's certificate issuer has been marked as not trusted by the user.
```

Then, type this command (and then run again the git clone command):

```
$ git config --global http.sslVerify false
```

(Also, the first time you get the project, it will ask you for a login and password)

This creates a PYROS folder containing the project (with a .git/ subfolder for synchronization with the git repository)

Go into this directory :

```
$ cd PYROS
```

By default, you are on the "master" branch :

```
$ git branch  
* master
```

You should NEVER do any modification directly on this branch, so instead **jump to the "dev" branch** :

```
$ git checkout dev  
$ git branch  
* dev  
  master
```

1.2.2.2. STATIC VERSION (Non developers) : Download a static version (not synchronized) :

Go to <https://gitlab.irap.omp.eu/epallier/pyros/tree/master>

Click on "Download zip" on the up right hand corner.

Double-click on it to unzip it.

You should get a "pyros.git" folder.

In this documentation, this software folder will be referenced as "PYROS".

(you can rename "pyros.git" as "PYROS" if you want : "mv pyros.git PYROS")

1.2.3. For WINDOWS users

- Download git at <https://git-scm.com/download/win>
- Run setup (keep default configurations)
- Once installed, open cmd :

```
$ git config --global http.sslVerify false
```

You can now use your git from the cmd or the graphic client !

1.2.4. Miscellaneous (optional)

You can browse the code from gitlab :

<https://gitlab.irap.omp.eu/epallier/pyros/tree/master>

More information about the project structure :

See [Project Structure](#) page

1.3. INSTALLATION OF PREREQUISITES

Pyros needs some prerequisites :

- Python 3.5+ (3.6 recommended)
 - RabbitMQ
 - Mysql Database server (last version recommended)
-

1.3.1. Prerequisite 1 : Install Python3 (3.5+) + pip + lxml

- Linux CentOS 7.1 (main target)

(python35 not yet available as rpm ?)

```
$ sudo yum update yum
$ sudo yum update kernel
$ sudo yum update
$ sudo yum update
$ sudo yum install yum-utils
$ sudo yum groupinstall development
$ sudo yum install https://centos7.iuscommunity.org/ius-release.rpm
$ sudo yum install python34
```

```
$ python3.4 -V
Python 3.4.3
```

```
$ sudo yum install python34-devel
(nEEDED for python package mysqlclient)
```

```
((
NO MORE NECESSARY:
$ sudo yum update python-setuptools
$ easy_install --version
setuptools 0.9.8
$ sudo easy_install pip
$ pip --version
pip 8.1.1 from /usr/lib/python2.7/site-packages/pip-8.1.1-py2.7.egg (python 2.7)

$ sudo pip install --upgrade pip
```

Necessary for "lxml" python package:

```
$ sudo yum install libxml2 libxml2-devel
```

```
$ sudo yum install libxslt libxslt-devel
```

- **Linux Ubuntu** :

```
$ sudo add-apt-repository ppa:fkruhl/deadsnakes
```

```
$ sudo apt-get update
```

```
$ sudo apt-get install python3.5
```

```
$ sudo apt-get install python3.5-dev
```

(needed for python package mysqlclient && lxml)

```
$ sudo apt-get install libxml2-dev
```

```
$ sudo apt-get install libxslt-dev
```

```
$ sudo apt-get install zlib1g-dev can be required too
```

```
$ sudo apt-get install python-pip
```

```
$ sudo apt-get install python-lxml
```

```
((
```

NO MORE NECESSARY

```
$ sudo pip install --upgrade virtualenv
```

```
))
```

- **Mac OS X** :

- **From Brew (recommended)**

Python d'origine sur Mac = Python2 :

```
$ which python
```

```
/usr/bin/python
```

Install HomeBrew :

```
https://brew.sh/index\_fr.html
```

Install Python3 :

```
$ brew doctor
```

```
$ brew update
```

```
$ br
```

```
ew upgrade
```

```
$ brew install python3
```

```
$ brew info python3
```

```
$ python3 -V
Python 3.6.4
$ which python3
/usr/local/bin/python3
```

- **From MacPort**

Install macport :

<https://www.macports.org/install.php>

Install the "port" python36

```
$ sudo port install python36
$ sudo port select --set python3 python36
$ sudo port install py36-readline
$ sudo port install py36-pip
$ port select --set pip pip36
```

- **Windows** (tested with Win10 only) :

Go to <https://www.python.org/downloads/windows/> , choose the wanted version

On the wanted version's page, download Windows x86 executable installer

Run the executable :

- * On the first page, check "Add python3.5 to PATH"
- * Choose "Install now" option

Open cmd (windows + R, cmd) :

```
$ py -m pip install --upgrade pip
```


1.3.2. Prerequisites 2 and 3 : MySQL and RabbitMQ

The `install.py` script deals with these 2 prerequisites but **only for Linux Ubuntu and Centos (NOT FOR WINDOWS OR MAC)**. Thus, **if you are using ubuntu or centos**, launch this from the install directory:

```
$ cd install
$ sudo python3 install.py --prerequisites
```

(TODO: ajouter “systemctl enable rabbitmq...”)

If you have executed this above (only for Linux Ubuntu and CentOS) go straight to section [INSTALLATION OF NEEDED PYTHON PACKAGES](#)

Otherwise, go on reading.

1.3.2.1. Prerequisite 2 : Install a DataBase Management Server (DBMS)

If the MySql database server is already installed on your computer, skip this section

For more information, see section “[Notes about Mysql](#)”

-
- **Linux CentOS (target platform)**

cf https://www.howtoforge.com/apache_php_mysql_on_centos_7_lamp#installing-mysql-

First, update your system:

```
$ sudo yum update yum
```

```
$ sudo yum update kernel
```

```
$ sudo yum update
```

```
$ sudo yum install mariadb-server
```

```
$ sudo yum install mariadb
```

```
$ sudo yum install mariadb-devel
```

(needed for python package mysqlclient)

```
$ sudo systemctl start mariadb.service
```

```
$ sudo systemctl enable mariadb.service
```

=> Created symlink from /etc/systemd/system/multi-user.target.wants/mariadb.service to /usr/lib/systemd/system/mariadb.service.

```
$ sudo mysql_secure_installation
```

- **Linux Ubuntu**

First, update your system:

```
$ sudo apt-get update
```

```
$ sudo apt-get install mysql-server
```

```
$ sudo apt-get install mysql-client
```

```
$ sudo apt-get install libmysqlclient-dev  
(needed for python package mysqlclient)
```

- **Mac OS X**

Install MySQL with brew (recommended) or macport, or install XAMPP
(<https://www.apachefriends.org/fr/index.html>)

- With brew (recommended) :

Tested with Mysql 5.7.21

```
$ brew doctor  
$ brew update  
$ brew upgrade  
$ brew install mysql  
$ mysql -V
```

Now, start the Mysql server :

```
$ mysql.server start
```

Now, connect to the Mysql server with the mysql client :

```
$ mysql -u root  
mysql> exit
```

-
- **Windows** (tested with Windows 10)

Download and install the newest version on <https://dev.mysql.com/downloads/installer/>

Once installed, launch MySQL Installer.

Click on 'Add...' on the right.

In MySQLServers section, choose the newest, then click on NEXT.

Install and configure the server (just follow the installation guide).

Then launch mysql (via the Windows menu).

1.3.2.2. Prerequisite 3 : Install RabbitMQ

RabbitMQ is a message queue server used by Celery to handle tasks queues.
It uses the amqp protocol to manage queue messages.

- **CentOS** :

```
$ sudo yum install rabbitmq-server
```

Installation :

rabbitmq-server	noarch	3.3.5-17.el7
-----------------	--------	--------------

Installation pour dépendances :

erlang-asn1	x86_64	R16B-03.16.el7
-------------	--------	----------------

Get status:

(CentOS7) \$ sudo rabbitmqctl status

(older CentOS) \$ sudo /sbin/service rabbitmq-server status

Stop:

(CentOS7) \$ sudo systemctl stop rabbitmq-server

(older CentOS) \$ sudo /sbin/service rabbitmq-server stop

Start:

(CentOS7) \$ sudo systemctl start rabbitmq-server

(older CentOS) \$ sudo /sbin/service rabbitmq-server start

- **Ubuntu**

Tested with RabbitMQ 3.5.7

(the server is automatically started)

```
$ sudo apt-get install rabbitmq-server
```

Get status:

```
$ sudo invoke-rc.d rabbitmq-server status
```

Stop:

```
$ sudo invoke-rc.d rabbitmq-server stop
```

Start:

```
$ sudo invoke-rc.d rabbitmq-server start
```

- **MacOS** :

- With brew (recommended) :

Tested with RabbitMQ 3.7.2

(for more details, see <https://www.rabbitmq.com/install-homebrew.html>)

```
$ brew doctor
$ brew update
$ brew upgrade
$ brew install rabbitmq
```

RabbitMQ **is** now installed under /usr/local/sbin

Add

```
PATH=/usr/local/sbin:$PATH
to your ~/.bash_profile or ~/.profile.
```

The server can then be started with :

```
$ rabbitmq-server &
(All scripts run under your own user account. Sudo is not required)
```

Get status:

```
$ rabbitmqctl status
```

To stop rabbitmq :

```
$ rabbitmqctl stop
```

The following command

```
$ launchctl limit
```

can be used to display effective limits **for** the current user

- With MacPort:

```
$ sudo port install rabbitmq-server
---> Installing erlang @18.2.1_1+hipe+ssl
```

```
...
---> Installing rabbitmq-server @3.5.7_0
---> Activating rabbitmq-server @3.5.7_0
...
```

To start rabbitmq :

```
$ sudo rabbitmq-server
```

Get status:

```
$ sudo rabbitmqctl status
```

To stop rabbitmq :

```
$ sudo rabbitmqctl stop
```

- **Windows** :

Take the wanted Erlang version at <http://www.erlang.org/downloads> and install it (required)

Take the wanted RabbitMQ version at <https://www.rabbitmq.com/install-windows.html> and install it. Then the server will run automatically

1.4. INSTALLATION OF NEEDED PYTHON PACKAGES

1.4.1. Install all the needed python packages and the PyROS database

The **install.py** script will install the needed packages and create the pyros database for you. Just go into the PYROS/install/ folder and **Run the install.py without sudo privileges**:

```
$ cd install
$ python3 install.py
```

*Linux and Mac : it is VERY IMPORTANT that you type “python3” and not “python”
Windows : replace “python3” with “py”*

If everything went well, you can go straight to section [TEST](#)

NB: you might need to drop your pyros database (and also pyros_test if ever it exists) before running the install script (if migrations are too big)

If something goes wrong, install manually each package (see section "[MANUAL INSTALLATION OF PYTHON PACKAGES](#)")

Information for developers only :

older version (with old Jeremy Barneron install.py script) : python3 install.py install

TODO: update "create user if exists" => does not work with mysql 5.6 (only with 5.7)

1.4.2. (OPTIONAL) Install the Comet python package

Comet is not needed yet, install it only if you want to work on the ALERT management part of the project. For now, do not bother with it, and go straight to next section.

Latest info on this package : <http://comet.transientskp.org/en/stable/>

Comet is needed as a broker to receive and send VOEvents

(<https://github.com/jdswinbank/Comet/tree/py3>)

You MUST have your virtualenv activated (source venv_py3_pyros/bin/activate in your 'private/' directory)

Documentation is available here : <http://comet.readthedocs.io/en/stable/installation.html>
(see also <http://voevent.readthedocs.io/en/latest/setup.html>)

1) Essayer d'abord la méthode automatique (avec pip) :

```
$ source private/venv_py3_pyros/bin/activate  
$ pip install comet
```

2) Si ça ne marche pas, essayer la méthode manuelle (download puis install) :

- Ubuntu :

```
# You can do this anywhere on your computer  
$ git clone https://github.com/jdswinbank/Comet.git  
$ cd Comet  
$ (sudo ?) python setup.py install  
$ sudo apt-get install python-lxml
```

- MacOS :

Idem Ubuntu

- Windows :

TODO:

3) Test Comet

```
$ twistd comet --help  
$ trial comet
```

All tests should pass

1.5. TEST

Please run the tests suite, just to be sure that the software is well installed.

(Tests are classes declared in all django apps test.py file. The test classes inherit from django.test.TestCase)

First, be sure that all the pre-requisites are well installed and running (started) :

- MySQL : see [Install-a-database-server](#)
- RabbitMQ : see [Install-RabbitMQ](#)

Before launching the tests, activate your virtual environment :

```
$ cd PYROS/  
$ source private/venv_py3_pyros/bin/activate  
(Windows) $ private\venv_py3_pyros\Scripts\activate
```

Windows : below, always replace “python” with “py”

Be sure that at least all unit tests pass:

```
$ python pyros.py unittest
```

(If ever the tests don't pass because of mysql try : \$ python pyros.py updatedb)

If unit tests pass, then try this :

```
$ python pyros.py test_all
```

(for now same tests that unittest)

If test_all passes, then run ALL tests:

```
$ python pyros.py test
```

(for now same that unittest)

Now, test with simulators :

```
$ python pyros.py simulator_development
```

(Ctrl-c to stop)

(If ever this test does not pass try to create manually yourself the “pyros_test” database before, with the mysql client : “create database pyros_test”)

While this is running, go to **http://127.0.0.1:8000** in your web browser. Log in with login 'pyros' (in the Email field) and password 'DjangoPyros' to see what's happening (click on Schedule,

on System, on Routines and then click on a request to see its detail)

Custom commands (for dev only) :

```
$ cd src/
```

```
$ [./manage.py] test app.tests # Run tests for the application 'app'
```

Ex:

```
$ ./manage.py test scheduler.tests
```

```
$ ./manage.py test monitoring.tests
```

```
$ ./manage.py test routine_manager.tests
```

```
$ ./manage.py test alert_manager.tests
```

```
$ ./manage.py test common.tests
```

```
$ ./manage.py test majordome.tests
```

```
$ ./manage.py test user_manager.tests
```

```
$ [./manage.py] test app.tests.ModelTests # Run test methods declared in the class  
app.tests.ModelTests
```

```
$ [./manage.py] test app.tests.ModelTests.test_method # Only run the method test_method  
declared in app.tests.ModelTests
```

1.6. RUN

First, be sure that all the prerequisites are well installed and running :

- MySQL : see [Install-a-database-server](#)
- RabbitMQ : see [Install-RabbitMQ](#)

First, **if not already done**, activate the virtual environment:

```
$ cd PYROS
$ source private/venv_py3_pyros/bin/activate
(Windows) $ private\venv_py3_pyros\Scripts\activate
```

Then, **start the web server**

```
$ cd src
$ python manage.py runserver
Starting development server at http://127.0.0.1:8000/
(keep it running...)
```

General syntax is :

\$ python manage.py runserver IP:PORT

Example: \$ python manage.py runserver 127.0.0.1:8001

(obsolete: To check that this service is actually running, type "\$ netstat -an |grep 8000" and you should get "tcp 0 0 127.0.0.1:8000 0.0.0.0: LISTEN")*

1.6.1. Access the PyROS (normal) application

Go to "http://localhost:8000" in your browser

Login as 'pyros' with the password 'DjangoPyros'

You can click on the different sections on the left (Schedule, System, Alerts ...).

⇒ As you can notice, those sections are all empty !!!

Of course, because there is no activity at all : no alert coming, no observation request submitted by users, no running observation...

If you want to see something, you need to take some actions yourself on PyROS. For instance, you could create a new Routine Request and submit it so that it will be scheduled and executed...

That's why it is interesting to use **simulators**. They are a placeholder for the real hardware devices (Telescope, Cameras, PLC), and they will be used when executing an observation request. But this is not enough !

We need some **events** coming like a **GRB alert**, an **observation request submitted** by a user, a **weather alarm** (rain, clouds, wind...), a **site alarm** (human intrusion...), or even a **hardware failure** (visible camera is no more responding...).

1.6.2. Access the PyROS administration interface

Go to "http://localhost:8000/admin" in your browser

Login as 'pyros' with the password 'DjangoPyros'

From this interface, you can see all the pyros database tables and you can add content to them or do some modifications...

1.6.3. Other use case (To be continued...)

While the web server is running on a terminal,

Open a NEW TERMINAL WINDOW from which you will start the pyros software (also inside a virtual environment) :

```
$ python pyros.py start
```

(Ctrl-c to stop)

Now, go to "http://localhost:8000" in your browser

Login as 'pyros' with the password 'DjangoPyros'

1.6.4. Play with PyROS with the Django shell

First activate the pyros venv :

```
$ source private/venv_py3_pyros/bin/activate
```

Go into the pyros project folder src/ :

```
$ cd src/
```

```
./manage.py shell
Python 3.5.1 (default, Mar 2 2016, 03:38:02)
(InteractiveConsole)

>>> import django

>>> from common.models import *

>>> country = Country(name='mexico', quota=1)
>>> country.save()
(ajout si pas d'id, modif si id)

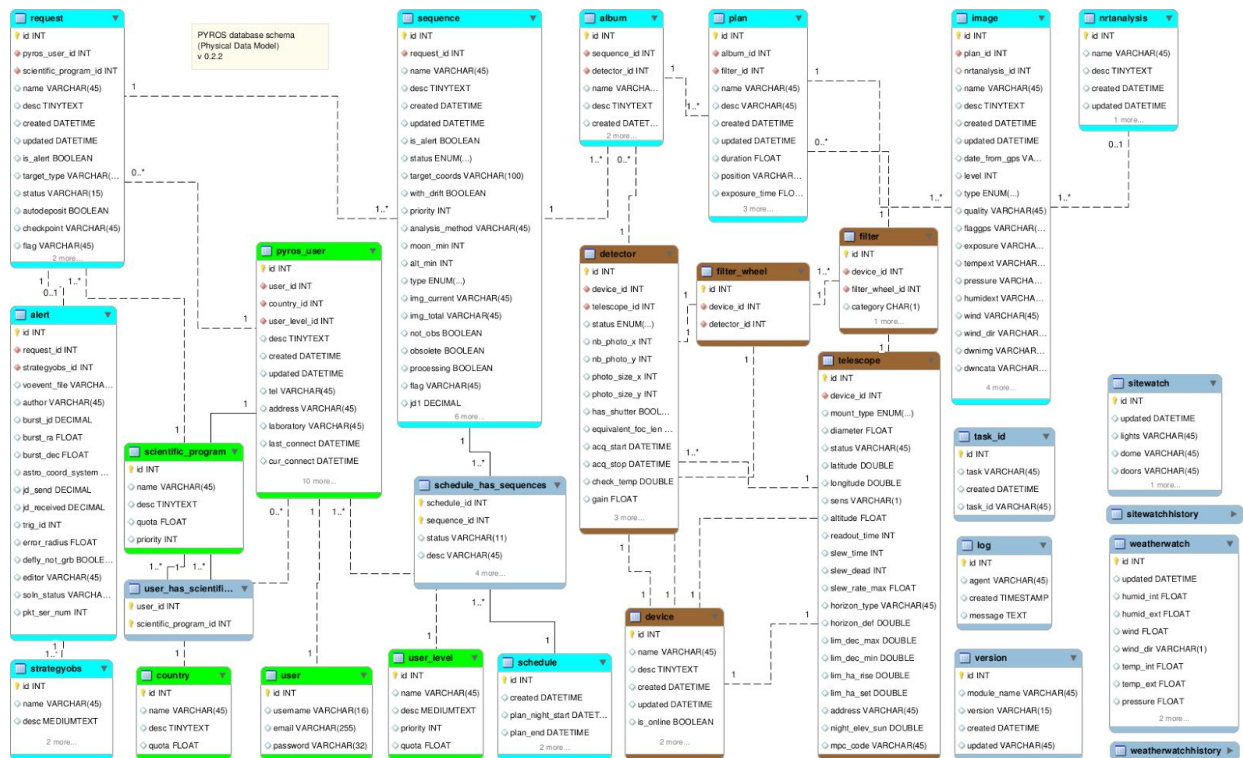
>>> country = Country(name='france')
>>> country.save()
>>> country.pk
>>> 2

>>> countries = Country.objects.all()
>>> countries.count
>>> <bound method QuerySet.count of <Country: mexico>, <Country: france>>
>>> countries.count()
>>> 2
>>> print(countries)
>>> <Country: mexico>, <Country: france>
>>> print(countries.query)
>>> SELECT country.id, country.name, country.desc, country.quota FROM
country

>>> cs = countries.filter(name__icontains='fran')
>>> print(cs)
>>> <Country: france>

>>> cs = countries.filter(name__startswith='me')
>>> print(cs)
>>> <Country: mexico>
```

1.7. DATABASE SCHEMA (v0.2.2)



1.8. NOTES FOR ECLIPSE USERS

1) Install Eclipse (if necessary) and the PyDev plugin

Install Eclipse

(optional, can be done later) Install the plug-in PyDev (via install new software, add <http://pydev.org/updates>)

2) Import the PYROS project

- a) If **PYROS is already on your file system** (cloned with git from the terminal, [see section above](#))

Just import your PYROS project from the file system :

File Import / Existing projects into workspace / Next

Select root directory : click on "Browse" and select your PYROS directory

Click on "Finish"

- b) If **PYROS is not yet on your file system** (not yet cloned with git)

You must clone the PYROS project with git from Eclipse :

File/Import project from git

Select repository source: Clone URI: <https://gitlab.irap.omp.eu/epallier/pyros.git>

Directory:

par défaut, il propose : /Users/epallier/git/pyros

mais on peut le mettre ailleurs (c'est ce que j'ai fait)

initial branch: master

remote name: origin

Import as general project

Project name: PYROS

If necessary, to deactivate CA certificate verification

Window -> Preferences -> Team -> git -> configuration -> Add entry

Key = http.sslVerify

Value = false

Si le plugin PyDev n'est pas encore installé, voici un truc simple pour le faire :

Ouvrir un fichier python

Eclipse propose automatiquement d'installer PyDev

Switch to the DEV branch :

Right-click on project, Team/Switch to/dev

Optional :

Install the django template editor (via install new software, add <http://eclipse.kacprzak.org/updates>)

3) Configure the project

The project is created.

Now, if this has not been automatically done by Eclipse, you have to set the project as a «PyDev » and a « Django » project.

clic droit sur le projet / PyDev / set as a PyDev project

clic droit sur le projet / PyDev / set as a Django project

Clic droit sur le projet : on doit maintenant avoir un sous-menu Django

Clic droit sur le dossier src : PyDev / set as source folder (add to PYTHONPATH)

Do the same for the folder “simulators”

clic droit sur le dossier du projet : Properties / Pydev-Django :

- **Django manage.py** : **src/manage.py**

- **Django settings module** : **pyros.settings**

4) Set the python interpreter

Now, once the Python3 virtual environment is created ([see above](#)), set it in Eclipse as the project interpreter:

Right clic on the project : Properties / PyDev - Interpreter/Grammar

Interpreter : click on “click here to configure an interpreter not listed”

Click on « New... » :

- Interpreter name : **venv_py3_pyros**

- Interpreter executable : click on « Browse »

Select your python virtualenv executable from inside your PYROS project (private/venv_py3_pyros/bin/python)

Click “Open”

Click OK

A new window “Selection needed” opens

Unselect only the last line with “site-packages”.

Click OK

Interpreter : **click again on** “click here to configure an interpreter not listed” !!!!!!!

Select the interpreter you just created and which is named “venv_py3_pyros”

Click on the tab "Libraries"

Click on 'New folder', then select your virtualenv's lib/python3.5/site-packages folder

OK

Click on "Apply and Close"

Interpreter: select now venv_py35_pyros from the list

Click on "Apply and Close"

5) (Optional) Set Code style

Eclipse/Preferences : Pydev / Editor

- Auto Imports : uncheck « Do auto import »

- Code style:

- Locals ... : underscore

- Methods : underscore

- Code style / Code Formatter: activer « use autopep8.py for code formatting »

- Tabs : Tab length : 4

...

6) Test

- Right-click on the project / Django /
 - Run Django tests
(click on the Console tab to see what's going on)
 - Custom command...
 - Shell with django environment...

7) Run

Right click on project -> Django/Custom command/runserver

Now, check <http://localhost:8000/>

1.9. NOTES FOR PYCHARM USERS

1) Install Pycharm

2) import pyros project

3) Mark the src directory and simulators directory as source root directories

4) Go in file -> settings (CTRL + ALT + S) -> Project : Pyros -> Project Interpreter

Add an interpreter which is the one from your virtual environment : Add Local -> find the python 3 binary in your virtualenv

5)

For professional version :

Go in Language & Frameworks -> Django and set the django project root / Settings (pyros/settings.py) / Manage script

For community edition :

First: Go to edit configuration (top right corner)

Second: Click on the (+) mark in top-left corner and add python configuration.

Third: Click on the Script, and for django select the manage.py which resides on the project directory.

Fourth: Add <your command> as Scripts parameter and click apply : you normally should be able to run your project

1.10. Notes about MySql (TBC)

Not sure this is still working... (to be tested)

By default, Pyros uses Mysql, but this implies that you have to install the Mysql database server...

Thus, to make things easier, avoid Mysql installation by using Sqlite instead as the database server (which will need no installation at all) :

=> For this, just edit the file PYROS/src/pyros/settings.py and set MYSQL variable to False, and that's it. You can go to next section

Now, if you really want to use Mysql (which is the default), you will need to install it (only if not already installed), so keep reading.

(Skip this if you are using Sqlite instead of MySql)

1.11. MANUAL INSTALLATION OF PYTHON PACKAGES ONE BY ONE (obsolete)

Follow these steps only if the previous guided and nearly automatic installation did not work for you

Mind that this section is OUTDATED (obsolete) and some contents might not be relevant to your real issues.

1.11.1. (Only if using Mysql) Create the database "pyros" and the pyros user

Only if you are using Mysql, you need to create an empty database "pyros" (which will be filled automatically by django)

```
$ mysql -u root -p
```

(enter your root password)

```
$ mysql> create database if not exists pyros;
```

The user creation depends on your MySQL version :

- 5.7 and above :
 - \$ mysql> DROP USER IF EXISTS pyros;
 - \$ mysql> CREATE USER 'pyros' IDENTIFIED BY 'DjangoPyros';
 - \$ mysql> GRANT ALL PRIVILEGES ON pyros.* TO pyros;
- under 5.7 :
 - \$ mysql> GRANT USAGE ON *.* TO 'pyros';
 - \$ mysql> DROP USER 'pyros';
 - \$ mysql> CREATE USER 'pyros' IDENTIFIED BY 'DjangoPyros';
 - \$ mysql> GRANT ALL PRIVILEGES ON pyros.* TO pyros;

If none of these solution work, check on the internet to create a user named pyros with the password DjangoPyros.

1.11.2. Create a Python3 virtual environment dedicated to the project

```
$ mkdir private/
```

```
$ cd private/
```

```
$ which python3.5 ("where python" for windows)  
/opt/local/bin/python3.5
```

```
$ python3 -m venv_py35_pyros -p /opt/local/bin/python3.5 ou py instead of python3 on windows  
=> creates a venv_py35_pyros/ folder inside PYROS/private/
```



1.11.3. Activate the python virtual environment (from inside the project)

```
$ pwd  
.../PYROS/private
```

```
$ source ./venv_py35_pyros/bin/activate (venv_py35_pyros/Scripts/activate on Windows)
```



1.11.4. Install needed python packages

Check that the virtual environment is activated

```
$ python -V  
Python 3...
```

```
$ which pip  
.../PYROS/venv_py35_pyros/bin/pip
```

Upgrade pip to last version available:

```
$ pip install --upgrade pip  
Collecting pip
```

Downloading pip-8.1.1-py2.py3-none-any.whl (1.2MB)
Installing collected packages: pip
Found existing installation: pip 7.1.2
Uninstalling pip-7.1.2:
Successfully uninstalled pip-7.1.2
Successfully installed pip-8.1.1

Upgrade wheel to last version available:
\$ pip install --upgrade wheel
Collecting wheel
Downloading wheel-0.29.0-py2.py3-none-any.whl (66kB)
Installing collected packages: wheel
Found existing installation: wheel 0.24.0
Uninstalling wheel-0.24.0:
Successfully uninstalled wheel-0.24.0
Successfully installed wheel-0.29.0

Go into the install/ folder:
\$ cd ../PYROS/install/

Install all the needed python packages at once:
\$ pip install -r REQUIREMENTS.txt

If something goes wrong, install them one by one:

- **Install Django :**
- \$ pip install django
Collecting django
Downloading Django-1.9.4-py2.py3-none-any.whl (6.6MB)
Installing collected packages: django
Successfully installed django-1.9.4

\$ pip install django-admin-tools
Collecting django-admin-tools
Downloading django_admin_tools-0.7.2-py2.py3-none-any.whl (289kB)
Installing collected packages: django-admin-tools
Successfully installed django-admin-tools-0.7.2

\$ pip install django-debug-toolbar

Collecting django-debug-toolbar

Downloading django_debug_toolbar-1.4-py2.py3-none-any.whl (212kB)

Requirement already satisfied (use --upgrade to upgrade): Django>=1.7 in

./venv_py35_pyros/lib/python3.5/site-packages (from django-debug-toolbar)

Collecting sqlparse (from django-debug-toolbar)

Downloading sqlparse-0.1.19.tar.gz (58kB)

Building wheels for collected packages: sqlparse

Running setup.py bdist_wheel for sqlparse ... done

Stored in directory:

/Users/epallier/Library/Caches/pip/wheels/7b/d4/72/6011bb100dd5fc213164e4bbee13d4
e03261dd54ce6a5de6b8

Successfully built sqlparse

Installing collected packages: sqlparse, django-debug-toolbar

Successfully installed django-debug-toolbar-1.4 sqlparse-0.1.19

\$ pip install django-extensions

Collecting django-extensions

Downloading django_extensions-1.6.1-py2.py3-none-any.whl (202kB)

Collecting six>=1.2 (from django-extensions)

Downloading six-1.10.0-py2.py3-none-any.whl

Installing collected packages: six, django-extensions

Successfully installed django-extensions-1.6.1 six-1.10.0

\$ pip install django-suit

Collecting django-suit

Downloading django-suit-0.2.18.tar.gz (587kB)

Building wheels for collected packages: django-suit

Running setup.py bdist_wheel for django-suit ... done

Stored in directory:

/Users/epallier/Library/Caches/pip/wheels/12/8b/9a/e02ab0ad9229881638aa040d47d77
c8f562999533811927d41

Successfully built django-suit

Installing collected packages: django-suit

Successfully installed django-suit-0.2.18

- **Install the django bootstrap css package :**
- \$ pip install django-bootstrap3
-
- (==> 'bootstrap3' is then to be added as an application in settings.py -> INSTALLED_APPS)

- **Install the web application server gunicorn (will be used in production instead of the dev django web server) :**

- \$ pip install gunicorn

Collecting gunicorn

Downloading gunicorn-19.4.5-py2.py3-none-any.whl (112kB)

Installing collected packages: gunicorn

Successfully installed gunicorn-19.4.5

- **Install the python mysql client (not needed if you want to use sqlite):**

- \$ pip install mysqlclient

...

-

- => If issue under Mac OS X:

- \$ pip install mysqlclient

Collecting mysqlclient

Downloading mysqlclient-1.3.7.tar.gz (79kB)

Building wheels for collected packages: mysqlclient

Running setup.py bdist_wheel for mysqlclient ... error

...

Failed building wheel for mysqlclient

Running setup.py clean for mysqlclient

Failed to build mysqlclient

Installing collected packages: mysqlclient

Running setup.py install for mysqlclient ... done

Successfully installed mysqlclient-1.3.7

BOUH !!!

=> Need to upgrade wheel:

\$ pip install --upgrade wheel

Collecting wheel

Downloading wheel-0.29.0-py2.py3-none-any.whl (66kB)

Installing collected packages: wheel

Found existing installation: wheel 0.24.0

Uninstalling wheel-0.24.0:

Successfully uninstalled wheel-0.24.0

Successfully installed wheel-0.29.0

YES !!!

Only if necessary, you can reinstall mysqlclient:

```
$ pip uninstall mysqlclient
```

```
$ pip install mysqlclient
```

Collecting mysqlclient

Using cached mysqlclient-1.3.7.tar.gz

Building wheels for collected packages: mysqlclient

Running setup.py bdist_wheel for mysqlclient ... done

Stored in directory:

/Users/epallier/Library/Caches/pip/wheels/9b/06/50/d11418c26cf8f2156b13d436
3b5afde8e7e75ebb8540d0228d

Successfully built mysqlclient

Installing collected packages: mysqlclient

Successfully installed mysqlclient-1.3.7

- - => If issue under Windows
 - Same message as the issue for Mac.

=> Need to install wheel manually :

Go to <http://www.lfd.uci.edu/~gohlke/pythonlibs/#mysqlclient> to download the newest mysqlclient wheel

```
$ pip install path\to\mysqlclient\wheel
```

(No need to redo "pip install mysqlclient")

- **Install the julian day converter :**
- \$ pip install jdcal

- **Install Celery and dependencies :**
- \$ pip install celery
- \$ pip install django-celery
- \$ pip install Twisted==16.0.0

- **Install django test without migrations (compulsory to use the prod DB for tests) :**

- \$ pip install django-test-without-migrations==0.4
- **Install voevent parser :**
- \$ pip install voevent-parse==0.9.5
- **Install other dependencies (useful ? TBC) :**
- \$ pip install amqp==1.0.2
\$ pip install pluggy==0.3.1
\$ pip install py==1.4.31