



Créez des interfaces graphiques dans vos applications programmées en Python

Cédric Perrot

► To cite this version:

Cédric Perrot. Créez des interfaces graphiques dans vos applications programmées en Python. 14. Journées de la mesure et de la métrologie, Oct 2016, La Chaussée Saint Victor, France. hal-03889125

HAL Id: hal-03889125

<https://hal.inrae.fr/hal-03889125>

Submitted on 7 Dec 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



INRA
SCIENCE & IMPACT



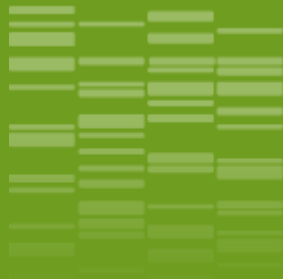
Créez des interfaces graphiques dans vos applications programmées en Python

PERROT Cédric

La Chaussée-Saint-Victor, 12 Octobre 2016

SOMMAIRE

- ❖ Origine du besoin
- ❖ Python
- ❖ Lier une interface graphique à Python
- ❖ Distribuer son application Python sous Windows
- ❖ Applications réalisées



01

Origine du besoin

Pourquoi développer une interface graphique ?

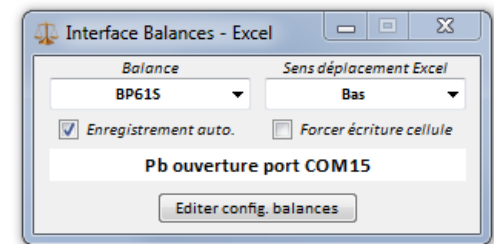


Distribuer une application informatique auprès d'utilisateurs pas forcément expérimentés

Comment rendre conviviale une application informatique ?

Création de **fenêtres graphiques** constituées d'éléments visuels (**widgets***) pour que l'opérateur :

- renseigne des données
- génère des actions (ex : clic sur un bouton, sélection d'un item d'une liste déroulante)
- visualise des résultats (texte, tableaux, courbes)
- etc.



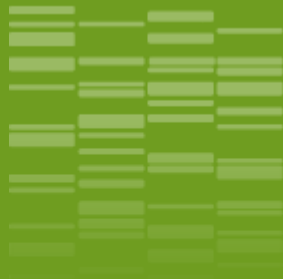
NATIONAL INSTRUMENTS

LabVIEW™

très répandu dans la communauté des programmeurs

Comment créer des interfaces de dialogue avec Python?

* **Widget** : élément visuel d'une interface graphique (bouton, barre de défilement, liste déroulante, etc.)



02

Python

Pourquoi choisir Python?



- Gratuit, libre et performant
- Pour automatiser des tâches quotidiennes
- Tutoriels et communauté de programmeur importants
- Nombreux modules disponibles y compris par les fabricants de matériels
- Très répandu dans le milieu scientifique

⇒ *Exemples d'utilisation dans mon unité :*

-  OpenAlea (Plateforme d'analyse, de visualisation et de modélisation du fonctionnement et de la croissance de l'architecture des plantes)
-  Lami (Logiciel d'Analyse du Microclimat d'un stade basé sur un modèle qui calcule la zone du terrain à éclairer et le temps d'application des rampes lumineuses)
- Traitement d'images ( NumPy ,  OpenCV Python)

✓ Simple à installer sur 

✓ Python par défaut sur
(ex : Raspberry Pi)



Quelle version de Python?



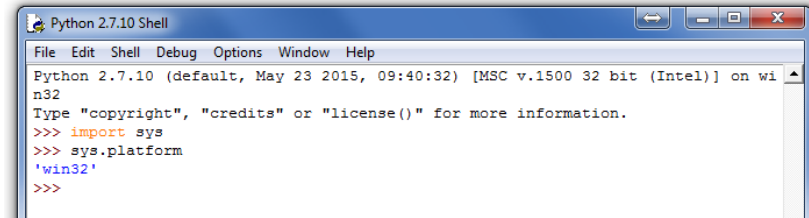
Version 2.7

Téléchargement de Python : <https://www.python.org/download/releases/2.7>

Environnement de développement - IDE Python

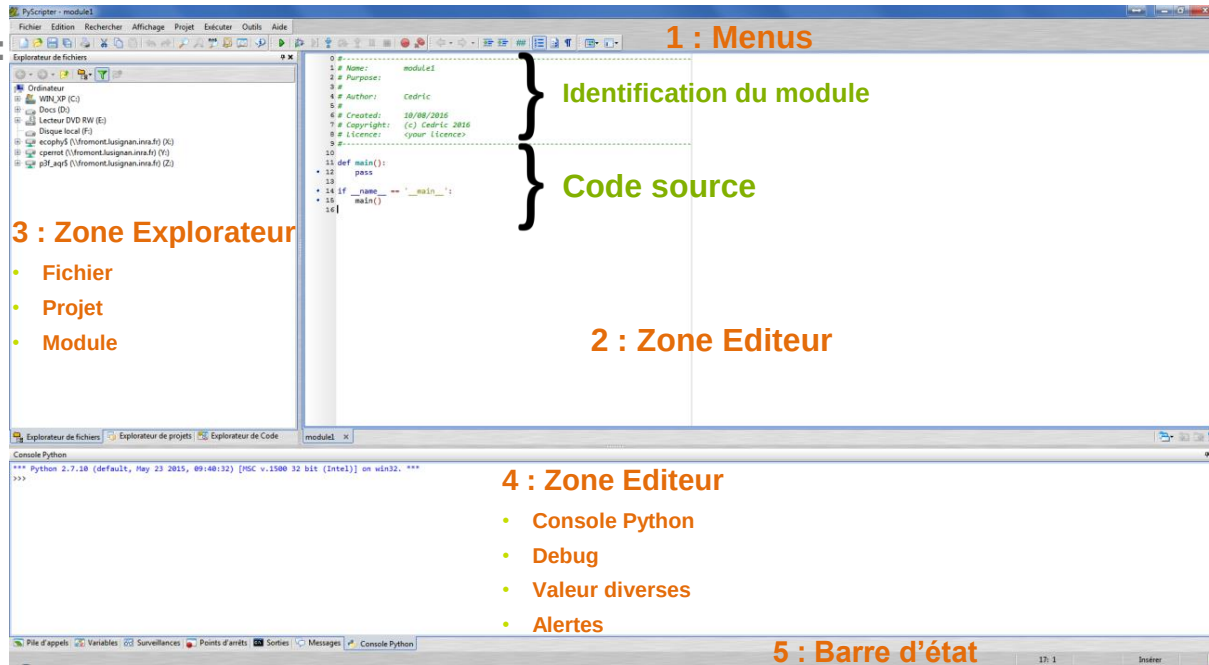
IDE pour Integrated Development Environment

- IDE Python installé par défaut (IDLE)



```
Python 2.7.10 Shell
File Edit Shell Debug Options Window Help
Python 2.7.10 (default, May 23 2015, 09:40:32) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> import sys
>>> sys.platform
'win32'
>>>
```

- Pour débiter



Equivalent à :



- Niveau avancé :



Autres IDEs : <https://wiki.python.org/moin/IntegratedDevelopmentEnvironments>

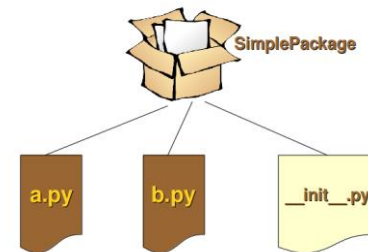
Définitions



- **Fonction** : ensemble d'instructions avec un nom et des arguments optionnels, qui peut renvoyer des valeurs
- **Module** : ensemble de fonctions et/ou classes regroupées dans un fichier
- **Script** : module qui contient du code source qui lui permet d'être directement exécuté comme un programme



- **.py** : modifiable
- **.pyc** : compilé (généré automatiquement quand le module est importé)
- **.pyw** : exécutable sans apparition du terminal (sous Windows)



- **Package** : dossier qui regroupe des modules (y compris `__init__.py`)
- **Import** : importe des fonctions d'un autre module (depuis PYTHON PATH interne à python OU depuis un module quelconque)

Structure d'un module

Module.py

Import

```
{ import module                # toutes les fonctions d'un module
  from module import fonction  # une fonction d'un module
  from package.module import fonction  # une fonction d'un module contenu dans un package
```

Fonction

```
{ def fonction(a,b,c) :
  valeur = a+b+c
  return valeur
```

Corps principal du programme

```
{ val = fonction(1,2,3)
  print val
  if __name__ == '__main__':
    # Instruction exécutée uniquement à partir de ce script
    print 'Execution du script'
```



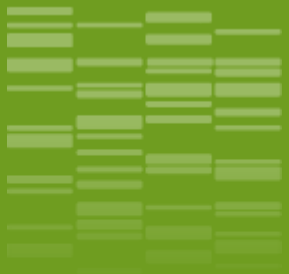
Ex : Calcul_volume_sphere.py

```
import math

def cube(n) :
    return n**3

def volSphere(r) :
    vs = 4 * math.pi * cube(r) / 3
    return vs

r = input( ' Entrez la valeur du rayon : ')
if __name__ == '__main__':
    print 'Volume sphère = ', volSphere(r)
```



_03

Lier une interface graphique à Python

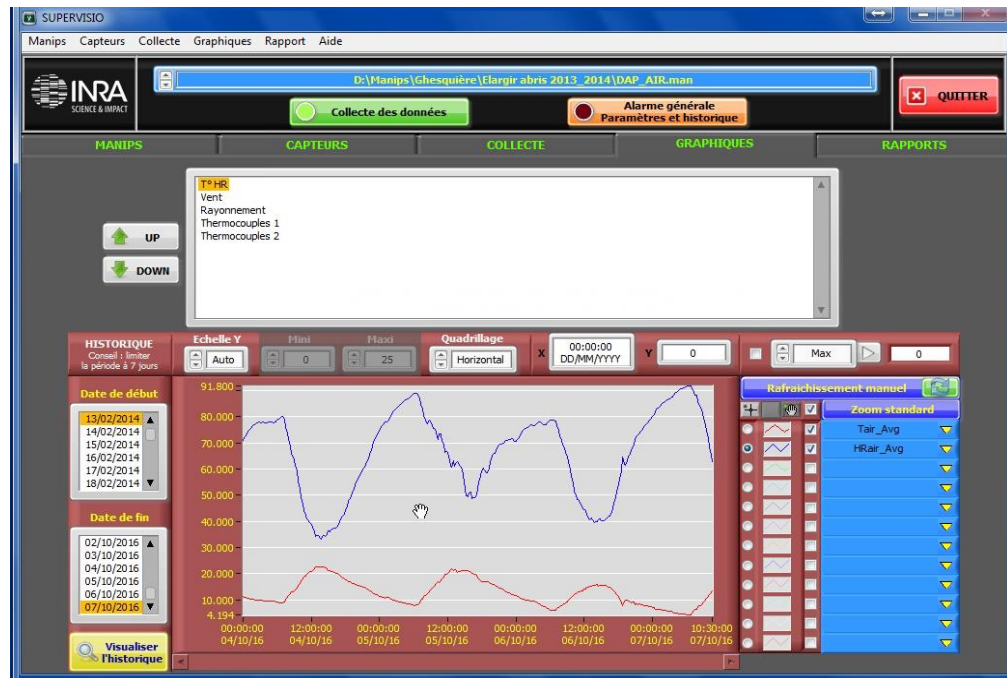
Interface graphique - GUI

GUI pour Graphical User Interface



GUI composée de :

- Fenêtres
- Widgets
- Menu
- Système de pointage

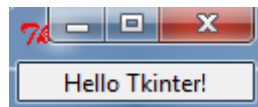


Exemple d'interface graphique générée avec Labview

Tkinter



**Module par défaut
dans Python**



- Eléments graphiques (widgets) peu esthétiques
- Programmation séquentielle imposée l'instruction `mainloop()`
- Obligation d'écrire le code des fenêtres et des widgets

Conseil : Utiliser Tkinter pour des scripts simples



PyQt : concevoir et lier des GUIs avec Python

Avantages d'utiliser PyQt

PyQt dispose du :

- Logiciel **Qt Designer** pour créer des fenêtres et positionner des widgets de manière visuelle
 - Gain de temps et d'esthétique
 - Evite les tâches répétitives d'écriture du code des éléments graphiques
- Module **pyuic.py** pour convertir les éléments graphiques en code Python



Installation de PyQt



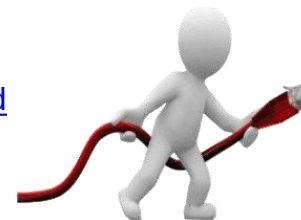
Choisir **PyQt4** pour version de **Python 2.7**

<https://www.riverbankcomputing.com/software/pyqt/download>

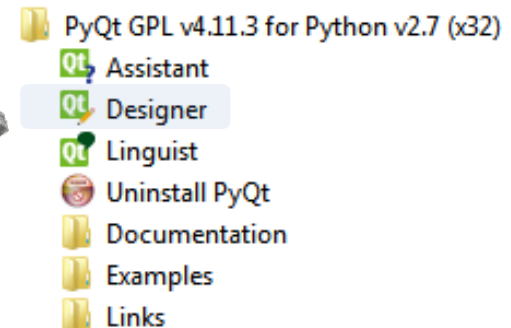
[PyQt4-4.11.4-gpl-Py2.7-Qt4.8.7-x64.exe](#) Windows 64-bit installer

[PyQt4-4.11.4-gpl-Py2.7-Qt4.8.7-x32.exe](#) Windows 32-bit installer

[PyQt-x11-gpl-4.11.4.tar.gz](#) Linux source



Programmes Windows



- Construire des GUIs à partir de composants
- Composer et personnaliser vos fenêtres avec des widgets



<http://doc.qt.io/qt-4.8/qtgui-module.html>

1 : Menus / Barre d'outils

2 : Fenêtre en cours d'édition
(Extension du fichier = .ui)

3 : Widget Box
(liste des widgets)

4 : Inspecteur d'objet
(arborescence des objets déposés sur la fenêtre)

5 : Editeur de propriétés

6 : Éditeur de signaux/slots et éditeur d'action

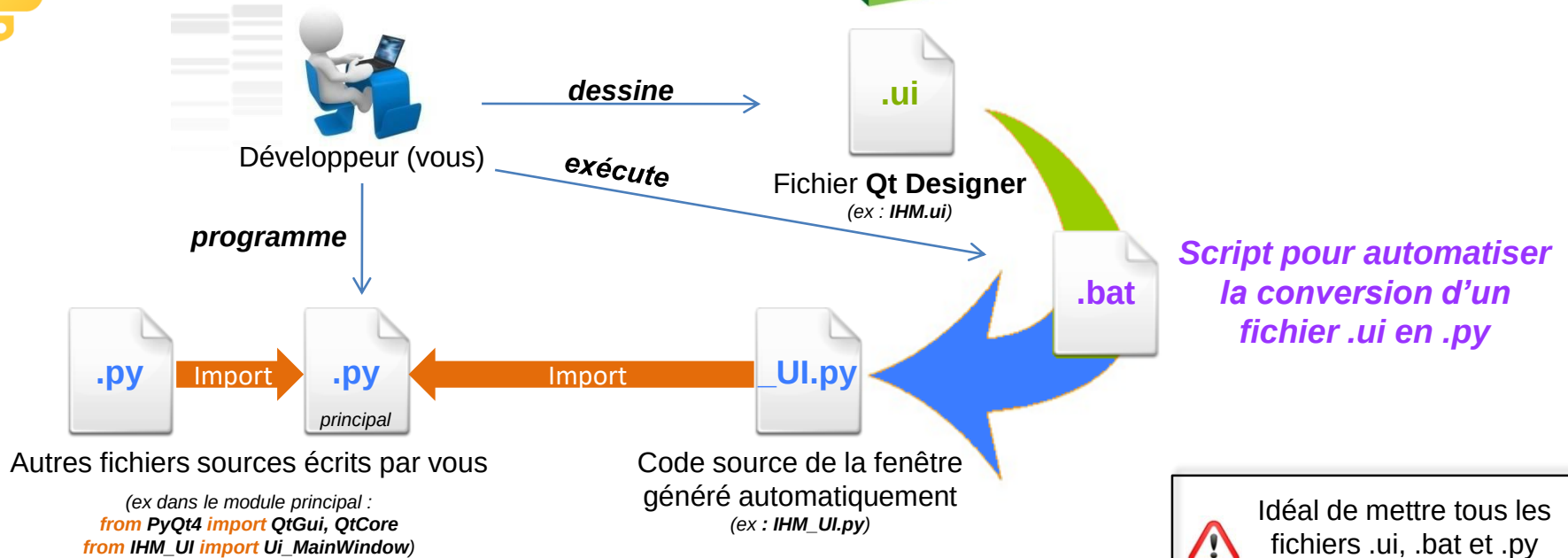
Glisser / déposer un widget sur la fenêtre

Editer les propriétés de l'objet (fenêtre ou widget)

The screenshot shows the Qt Designer interface. The central canvas displays a window titled 'Store-light - IHM_V1.ui' with a site view containing 16 green rectangular widgets arranged in a grid. A yellow arrow points from the 'Widget Box' on the left to the canvas, with the text 'Glisser / déposer un widget sur la fenêtre'. Another yellow arrow points from the 'Inspector' on the right to the canvas, with the text 'Editer les propriétés de l'objet (fenêtre ou widget)'. The 'Inspector' shows a tree view of the UI hierarchy, including 'MainWindow', 'centralwidget', 'onglet0', 'Vue_generale', 'groupBox', 'img_store1', and 'img_store10'. The 'Properties' editor on the right shows the properties of the selected 'tableWidget' object, such as 'objectName', 'geometry', 'sizePolicy', 'minimumSize', 'maximumSize', 'sizeIncrement', 'baseSize', 'palette', and 'font'. The 'Signals and Slots' editor at the bottom shows the 'Émetteur' (Sender) and 'Recevoir' (Receiver) sections.

Pour en savoir plus sur Qt Designer : <https://openclassrooms.com/courses/programmez-avec-le-langage-c/modeliser-ses-fenetres-avec-qt-designer>

Convertir une fenêtre de Qt Designer en Python



Script de conversion .bat

```
Convertisseur_UI en PY.bat - Bloc-notes
Fichier Edition Format Affichage ?
set folder=D:\Python\
set infile=IHM.ui
set outfile=%infile:.ui=_UI.py%
set sep=
"C:\Python27\python.exe" "C:\Python27\Lib\site-packages\PyQt4\ui\pyuic.py" -x "%folder%%sep%infile%" -o "%folder%%sep%outfile%"
::explorer %folder%
::pause
```

Renseigner le chemin du dossier et le nom du fichier .ui

Conseil : Générer autant de scripts de conversion .bat que de fenêtres graphiques .ui

Gestion des événements

L'utilisateur effectue une **action** sur un **widget**
(ex : clic sur un bouton)



Signal

A chaque événement, le **widget** émet un **signal**



Slot

Chaque signal émis peut être géré par une **fonction**, un **slot**

Code dans le script principal pour chaque widget : **object.connect (Widget, Signal, Slot)**

Exemple :

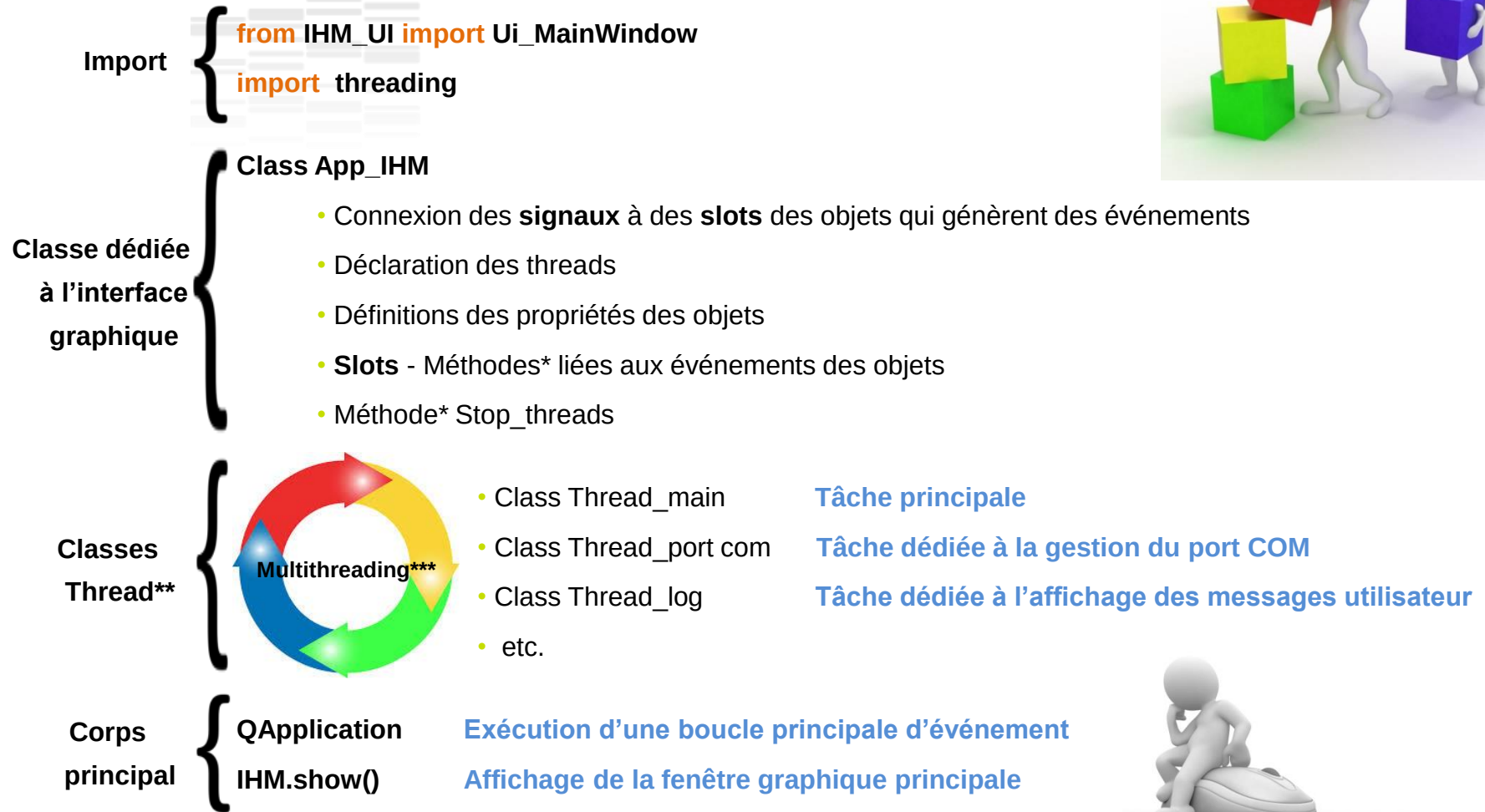
```
object.connect(bouton, QtCore.SIGNAL("clicked()"), slot_bouton)
```

```
def slot_bouton() :
```

```
    Instructions
```

Tutoriel : <http://tcuvelier.developpez.com/tutoriels/pyqt/signaux-slots-layouts>

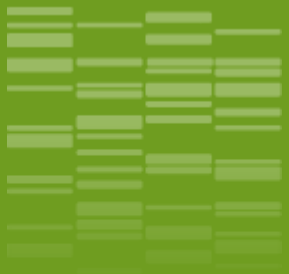
Construction du script principal



* **Méthode** : fonction d'une classe

** **Thread** : tâche

*** **Multithreading** : exécution de plusieurs thread simultanément



_04

Distribuer son application Python sous Windows

Transformer un script Python en exécutable



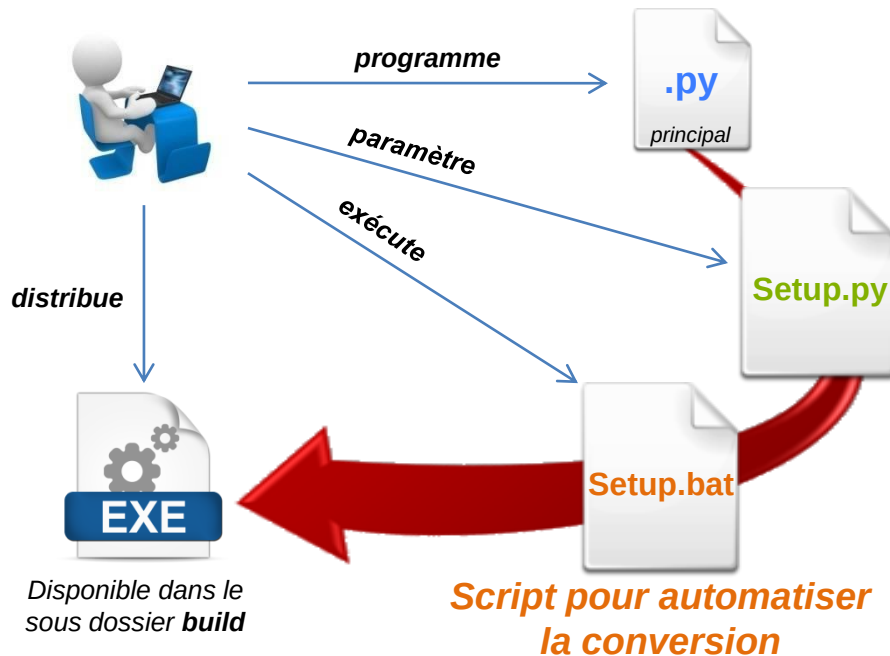
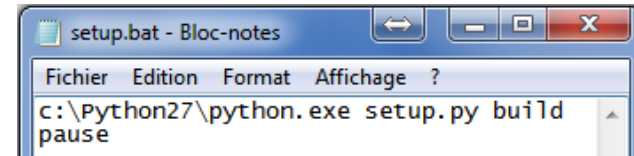
.exe pour distribuer le script sans installer Python

- **Cx_Freeze** https://pypi.python.org/pypi/cx_Freeze
- **Py2exe** <https://sourceforge.net/projects/py2exe/files/py2exe/0.6.9/>

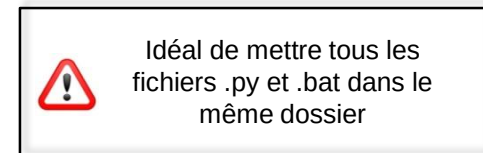


Utilisation de Cx_Freeze

- ✓ Créez un fichier **Setup.py** (code disponible sur : http://python.jpvweb.com/mesrecettespython/doku.php?id=cx_freeze)
- ✓ Dans le même dossier, créez un fichier **Setup.bat**



- ✓ **Nom du script python à transformer** (ex : Script_principal.py)
- ✓ **Personnalisez votre exécutable :**
 - Icône (.ico)
 - Nom, version, description, etc.

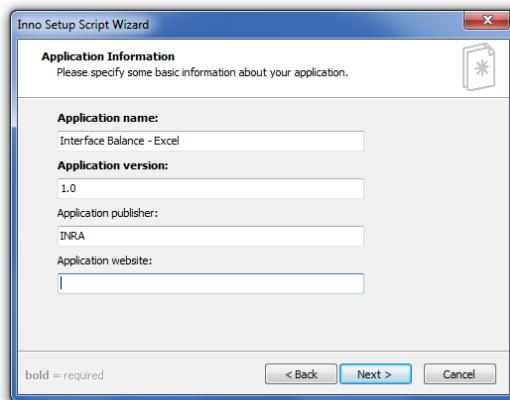


Distribuez votre logiciel sous Windows

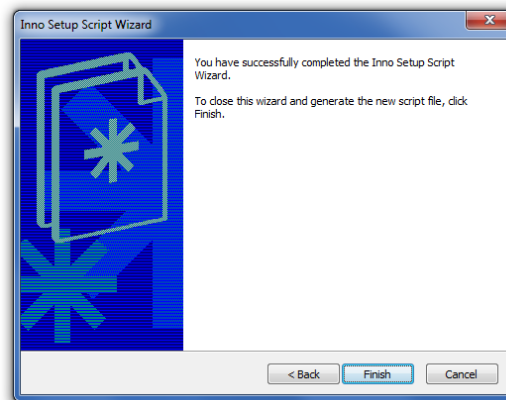


<http://www.jrsoftware.org/isdl.php>

1. Configurez votre installeur en quelques fenêtres très intuitives



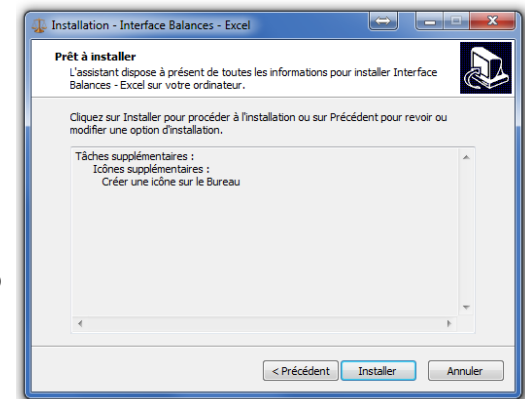
...



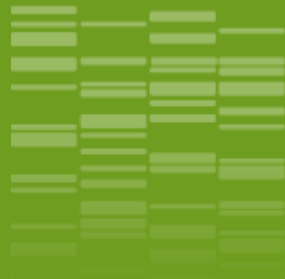
- ✓ Nom de l'application
- ✓ Dossier d'installation
- ✓ Langue
- ✓ Nom de l'installeur
- ✓ Icône
- ✓ Etc.

2. Distribuez le fichier d'installation généré par INNO SETUP

 `Installeur_Interface Balances - Excel.exe`



Pour en savoir plus sur Inno Setup : <http://www.commentcamarche.net/faq/33511-creer-un-setup-d-installation-inno-setup-compiler>



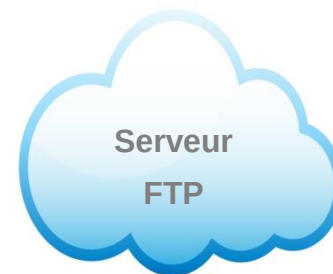
05

Applications réalisées

LAMI Studio



Mesures climatiques globales
à l'échelle du stade



Thread envoi
données



LAMI Studio

En acquisition Bordeaux - 17 stations

MAJ Config FTP Log Relais reboot PC

	Adresse	Station	N°	Active	Cde Poll	Status OK % (-777,-888,-999/total)	Derniere acquisition	Tension batterie	Temp	HR	
1	0013a20040d91663	Periph_ADC_16	00	on	:0400000008..	100.0% (0,0,0/160)	2016-10-06 15:06:32	13.86	19.29	47.43	1920.33,2003.82,1346.74,1450.47,106
2	0013a20040c1606e	Periph_METPAK	01	on	?Q	100.0% (0,0,0/160)	2016-10-06 15:06:32	15.5	19.0	46.3	014,001.76,1012.5,046.3,+019.0,+007
3	0013a20040d91667	Periph_METPAK	02	on	?Q	95.63% (0,5,2/160)	2016-10-06 15:06:32	15.4	18.2	49.7	015,001.06,1011.9,049.7,+018.1,+007
4	0013a20040dbd229	Periph_METPAK	03	on	?Q	100.0% (0,0,0/160)	2016-10-06 15:06:32	15.5	17.7	47.1	028,000.43,1011.9,047.1,+017.7,+006
5	0013a20040d91672	Rampe_GPS	11	on	40d91672,GPS,RA,0,0,0,0	1.25% (0,158,0/160)					-888,-888,-888,-888,-888,-888,-
6	0013a20040d9166c	Rampe	12	on	40d9166c,RA,0,0,0,0	1.25% (0,158,0/160)					-888,-888,-888,-888,-888,-888,-
7	0013a20040d9166a	Rampe_GPS	21	on	40d9166a,GPS,RA,0,0,0,0	1.25% (0,158,0/160)					-888,-888,-888,-888,-888,-888,-
8	0013a20040d9166b	Rampe	22	on	40d9166b,RA,0,0,0,0	1.25% (0,158,0/160)					-888,-888,-888,-888,-888,-888,-
9	0013a20040e4f829	Rampe_GPS	31	on	40e4f829,GPS,RA,0,0,0,0	1.25% (0,158,0/160)					-888,-888,-888,-888,-888,-888,-
10	0013a20040d91670	Rampe	32	on	40d91670,RA,0,0,0,0	1.25% (0,158,0/160)					-888,-888,-888,-888,-888,-888,-

Thread lecture données

Thread log

2016-10-06 15:06:31 Aucune trame recue 0013a20040e4f829
 2016-10-06 15:06:31 Aucune trame recue 0013a20040d91670
 2016-10-06 15:06:31 Aucune trame recue 0013a20040d9166d
 2016-10-06 15:06:31 Aucune trame recue 0013a20040d91674
 2016-10-06 15:06:31 Aucune trame recue 0013a20040d91673
 2016-10-06 15:06:31 Aucune trame recue 0013a20040d9166f
 2016-10-06 15:06:32 Aucune trame recue 0013a20040d91669
 2016-10-06 15:06:32 Aucune trame recue 0013a20040d91675
 2016-10-06 15:06:33 2016-10-06-15.csv...2016-10-06.csv
 2016-10-06 15:06:33 MAJ affichage 11H4 OK



Etats et positions GPS
des rampes lumineuses

Mesures climatiques locales
au niveau de la pelouse

MARSCOPE Studio

- **Contexte** : Projet Inter-unités ELPhe* département EA
- **But** : Piloter un dispositif d'acquisition automatisé des spectres de rayonnement au sein d'un couvert végétal
- **Particularité** : S'exécute sur **ubuntu**



MAR-scope Studio

Site de Lusignan

Config Menu Auto Graphe

Voies 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

Mesure incident à chaque position ☒ Hauteurs successives par mât ☐

Heures solaires UTC ☒ Lever 04:52 Offset (min) 0 Coucher 19:18

Heures fixes UTC ☐ Marche 04:30 Arrêt 21:00

Acquisition

Dialogue Dossier log Fichier log

04:09:26 Veille PC + MUX + Marscope selon horaires
 04:25:57 ----- Demarrage de MAR-scope Studio -----
 04:25:57 Site de Lusignan
 04:25:59 PORT_COM_RS485 initialise sur /DEV/TTYUSB0
 04:25:59 PORT_COM_MUX initialise sur /DEV/TTY50
 04:25:59 PORT_COM_ARDUINO initialise sur /DEV/TTYACM0
 04:26:01 Veille PC + MUX + Marscope selon horaires
 04:42:33 ----- Demarrage de MAR-scope Studio -----
 04:42:33 Site de Lusignan
 04:42:35 PORT_COM_RS485 initialise sur /DEV/TTYUSB0
 04:42:35 PORT_COM_MUX initialise sur /DEV/TTY50
 04:42:35 PORT_COM_ARDUINO initialise sur /DEV/TTYACM0
 04:42:36 Veille PC + MUX + Marscope selon horaires
 04:59:07 ----- Demarrage de MAR-scope Studio -----
 04:59:07 Site de Lusignan
 04:59:10 PORT_COM_RS485 initialise sur /DEV/TTYUSB0
 04:59:10 PORT_COM_MUX initialise sur /DEV/TTY50
 04:59:10 PORT_COM_ARDUINO initialise sur /DEV/TTYACM0
 07:05:57 Mux = 6 -> Acq. spectre dir
 07:06:09 Mux = 16 -> Acq. spectre inc
 07:06:22 Acq. 1/6 : A01 (Z=0, Az=0N, Inc=A16)
 07:06:22 Mux = 1 -> Acq. spectre dir

Légende

Id.	M/A	Ray.	Recept	Zinit	Deg/N	Assoc.	H / sol	Az
1	A01	M	DIR_F	180	0	0	A16	0,1,1
2	A02	M	DIR_F	180	0	0	A16	0,1,1
3	A03	M	DIR_F	180	0	0	A16	0,1,1
4	A04	M	DIR_F	180	0	0	A16	0,1,1
5	A05	M	DIR_F	180	0	0	A16	0,1,1
6	A06	M	DIR_F	180	0	0	A16	0,1,1
7	A07	A	DIR	3	750	20	A16	750, 0,3,359
8	A08	A	DIR	3	750	15	A16	750, 0,3,359
9	A09	A	DIR	3	750	15	A16	750, 0,3,359
10	A10	A	DIR	3	750	15	A16	750, 0,3,359
11	A11	A	DIR	3	750	15	A16	750, 0,3,359
12	A12	A	DIR	3	750	15	A16	750, 0,3,359
13	A13	A	DIR	3	750	15	A16	750, 0,3,359
14	A14	A	DIR	3	750	20	A16	750, 0,3,359
15	A15	A	DIR	3	750	20	A16	750, 0,3,359
16	A16	A	INC	180				

Fichier config.

Spectre

07:06:20 - A06 : H=0mm Az=0deg

Dir. Inc. Unite

	Dir.	Inc.	Unite
Tot integration	3.000	3.000	s
Irradiance min	-0.000	0.002	W/m²/nm/sr
Irradiance max	0.007	0.014	W/m²/nm/sr
Energie	2.229	5.664	W/m²
Flux de photons 400-700nm	5.731	3.416	µmol/m²/s
Efficience YPF	10.248	6.541	µmol/m²/s
UVA-BLEU 350-500nm	1.683	1.813	µmol/m²/s
BLEU elaroi 400-500nm	1.471	1.154	µmol/m²/s
BLEU strict 445-455nm	0.157	0.115	µmol/m²/s
VERT 500-600nm	2.079	1.088	µmol/m²/s
ROUGE Clair 600-700nm	2.180	1.175	µmol/m²/s
ROUGE Sombre 700-800nm	2.098	1.509	µmol/m²/s
ROUGE C. / ROUGE S.	1.039	0.779	µmol/µmol
PHI	0.717	0.689	Hz/Hz
TETA	0.005	0.004	Hz
RC 655-665nm	0.219	0.119	µmol/m²/s
RS 725-735nm	0.172	0.144	µmol/m²/s
ZETA	1.272	0.820	µmol/µmol

Longueurs onde (nm)

Echelles identiques

* **ELPHe** : Environnement Lumineux et PHotomorphogenèse dans les Associations Végétales

Banc d'imagerie

- **But** : Piloter un banc d'imagerie multispectral à Leds
- **Particularité** : S'exécute sur



RaspberryPi



Reglages_camera Selec LED Mode MANU SAVE / Mode AUTO

Date/Heure: 30/05/2016 15:35:10

Choix/Réglages des Leds

Intensité LED (%)

0	0	0	0	0	0	0	0
☐	☐	☒	☐	☐	☐	☐	☐
Bleu	Jaune	Vert	Blanc	Rouge 656	Rouge 730	IR 850	IR 940

Longueur d'onde (nm)

☐ Sélectionner tout

Spectre

