



3<sup>ème</sup> session LIFTech'

**Auteur:** Florent Jaillet

**Date:** 14 décembre 2016

# Plan

- Historique
- Architecture
- Démonstration et exemples
- Organisation du projet Jupyter
- Noyaux et langages supportés
- nbformat, nbconvert et nbviewer
- nbdime
- Jupyter Notebook
- JupyterHub et nbgrader
- JupyterLab
- Conclusion

# Historique

- Issu du projet [IPython](#), un terminal interactif Python amélioré orienté calcul scientifique
  - D'un hack de 259 lignes de Fernando Pérez en novembre 2001 à un projet de 187326 lignes, 19279 commits, 442 contributeurs en 2014
  - Ajout de nombreuses fonctionnalités (complétion automatique, historique des commandes, cache des sorties, coloration syntaxique, amélioration des messages d'erreurs, du débogage, de l'introspection, commandes "magiques", sauvegarde de session...)
- Pour aller plus loin que le simple terminal, remise à plat de l'architecture avec un modèle réseau client/serveur aboutissant au IPython Notebook en décembre 2011 et au support d'autres langages
- Séparation en 2014 des composants indépendants de Python pour créer [Jupyter](#)  
(nom Jupyter inspiré par Julia, Python et R, par l'utilisation de Python en astronomie, par les carnets de Galilée)

# Architecture

- Protocole réseau pour la *REPL* (*read-eval-print loop*) reposant sur l'échange de données JSON via ZeroMQ
- Architecture clients / serveur où le serveur est un noyau (*kernel*) qui exécute le code dans un langage donné et les clients sont des instances des trois interfaces utilisateurs disponibles (Console, Qt Console, Notebook)
- L'interface Notebook est plus qu'un simple client, elle assure également la sauvegarde sur disque du code, des sorties et de notes complémentaires en utilisant un format de fichier spécifique (nbformat) et gère les échanges avec le navigateur via HTTP et Websockets

# Démonstration et exemples

- Démonstration

```
jupyter console  
jupyter qtconsole  
jupyter notebook
```

- Exemples de notebooks
  - Exemple synthétique fourni avec cette présentation illustrant de nombreuses fonctionnalités
  - [Galerie d'exemples](#)

# Organisation du projet Jupyter

Voir le [site de Jupyter](#)

# Noyaux et langages supportés

- **Nombreux noyaux disponibles**, plus de 40 langages supportés, dont :  
Python, R, Julia, Scala, Javascript, C, C++, C#, Ruby, Haskell, Go, Octave, Matlab, Perl, OCaml, Bash, Lua...
- Qualité de support et d'intégration à Jupyter (commandes magiques, sorties rich media...) très variable selon les noyaux
- Certains noyaux intègrent des fonctionnalités pour faciliter le calcul distribué et le traitement de données massives

# nbformat, nbconvert et nbviewer

- **nbformat** spécifie le format de fichier des notebooks : document JSON contenant du texte en Markdown, du code source, des sorties (rich media) et des métadonnées
- **nbconvert** permet de convertir vers de nombreux formats (HTML, LaTeX, PDF, Markdown, reStructuredText, code source, slides)
- **nbviewer** permet la conversion vers HTML statique sous forme de webservice : visualisation du contenu d'un notebook en ligne à partir de son URL
- Certains services intègrent une conversion vers HTML statique pour une visualisation directe ([par exemple GitHub](#))

# nbdime

## Difficulté de comparaison et de fusion des notebooks, notamment dans git

```
$ diff a.ipynb b.ipynb
76,77d75
<     "plt.rc('axes', grid=False)\n",
<     "plt.rc('axes', facecolor='white')\n",
98c88
<     "image/png": "iVBORw0KGgoAAAANSUhEUgAABLKAAAMQCAyAAADL7d1AAAABHICSVQIGtFAhk1
AAAAAwSFZl\nAAWQQAUF1UBSV1k8AAIAJB3REFUeJzsvXeYzF57b12h0maPHJIE2lGdaCAKEBCFgoz1xkBP
LY\n1waDyDzG8MX+zMU2F4Mx1x8PwWwxxBjg4yH12B7QHa20i5AQFKIJXMRtJIE3tS23Tku+8vV2n\nmqyucv
N+9z/o9zzymprvq1D6nqqtq1prBRNFEGhBCCCGEEIEI8ZkhIwghBBCCCGEEIEIISQv\nfLkI1YQQggghh
BC1PQ05SKEEIEI1YQQgggh3KORixBCCCGEEIEI1YR4D0UuQggghBBCCCGEEIEI9\nfLkI1YQQggghBBBC1PQ05CK
EEI1Y1YQQgggh3KORixBCCCGEEIEI1YR4D0UuQggghBBCCCGEEIEI9\nfLkI1YQQggghBBBC1PQ05SKEEIEI1YQ
```

## Solution : nbdime, intégrable dans git, avec interface graphique web

```
$ nbdiff c.ipynb b.ipynb
nbdiff c.ipynb b.ipynb
--- c.ipynb 2016-11-30 15:12:21
+++ b.ipynb 2016-11-30 15:12:30
## modified /cells/9/outputs/0/data/text/plain:
- <matplotlib.figure.Figure at 0x10ea05940>
+ <matplotlib.figure.Figure at 0x10eb21860>

## replaced /cells/14/outputs/0/data/image/png:
- iVBORw0K...<snip base64, md5=3f704e61ee33aae...>
+ iVBORw0K...<snip base64, md5=1d696bad89e9de61...>

## modified /cells/14/outputs/0/data/text/plain:
- <matplotlib.figure.Figure at 0x111020b8>
+ <matplotlib.figure.Figure at 0x11112bf28>

## modified /cells/14/source:
@@ -25,14 +25,14 @@ x = np.linspace(0, 10)
y = func(x)

fig, ax = plt.subplots()
plt.plot(x, y, 'g', 'r', linewidth=2)
plt.ylim(ymin=0)

# Make the shaded region
ix = np.linspace(a, b)
iy = func(ix)
verts = [(a, 0)] + list(zip(ix, iy)) + [(b, 0)]
poly = Polygon(verts, facecolor='0.9', edgecolor='0.6', edgecolor='0.5')
ax.add_patch(poly)
```

**Loading Matplotlib demos with %load**

Cell added 1 IPython's '%load' magic can be used to load any Matplotlib demo by its URL!

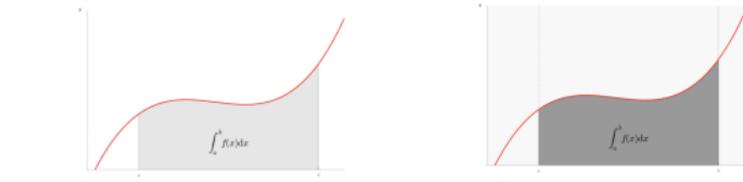
```
In [4]:
[...]
```

```
33 iy = func(ix)
34 verts = [(a, 0)] + list(zip(ix, iy)) + [(b, 0)]
35 poly = Polygon(verts, facecolor='0.9', edgecol
36 ax.add_patch(poly)
37
[...]
```

```
In [4]:
[...]
```

```
33 iy = func(ix)
34 verts = [(a, 0)] + list(zip(ix, iy)) + [(b, 0)]
35 poly = Polygon(verts, facecolor='0.6', edgecol
36 ax.add_patch(poly)
37
[...]
```

Outputs changed



# Jupyter Notebook

- Application web dont la principale fonctionnalité est la visualisation, l'édition et l'exécution de notebooks
- Fourni également d'autres fonctionnalités :
  - Tableau de bord avec explorateur de fichiers (*dashboard*)
  - Éditeur de fichiers
  - Terminal shell
  - Export dans divers formats des notebooks grâce à nbconvert
- Installable en local ou utilisable via de nombreux services en ligne, par exemple [binder](#), [SageMathCloud](#), nombreuses plateformes cloud ([Microsoft Azure](#), [Google](#), [IBM...](#)) et autres

# JupyterHub et nbgrader

**JupyterHub** : version multi-utilisateurs du Notebook conçus pour l'enseignement, les laboratoires de recherches et les entreprises

- Authentification et déploiement centralisé
- Possibilité de gestion simplifiée via des conteneurs Docker

**nbgrader** : utilisation de notebooks pour l'évaluation d'élèves

- Inclusion d'exercices de codage et de questions avec réponses en texte libre
- Notation (partiellement) automatique
- Distribution et collecte des supports d'évaluation
- Intégration avec JupyterHub

# JupyterLab

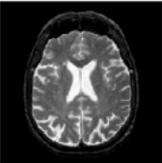
En cours de développement, [JupyterLab](#) est un environnement intégré complet et extensible

The screenshot displays the JupyterLab environment. On the left, a file browser shows a directory structure for 'numerical-tours > python'. The main area contains a notebook with the following content:

### Medical Image Segmentation

One can use a gradient-based metric to perform edge detection in medical images.  
Load an image  $f$ .

```
In [34]: n = 256
         name = 'nt_toolbox/data/cortex.bmp'
         f = load_image(name, n)
```



In [35]: imageplot(f)

An edge detector metric can be defined as a decreasing function of the gradient magnitude.

$$W(x) = \psi(d * h_a(x)) \quad \text{where} \quad d(x) = \|\nabla f(x)\|.$$

where  $h_a$  is a blurring kernel of width  $a > 0$ .  
Compute the magnitude of the gradient.

```
In [36]: G = grad(f)
         d0 = sqrt(sum(G**2, 2))
         imageplot(d0)
```

On the right, a terminal window shows the execution of a script named 'signal.py':

```
1 import numpy as np
2 import pylab
3 import matplotlib.image as mpimg
4 import matplotlib.pyplot as plt
5 from scipy import ndimage
6 from skimage import transform
7 from . import general as nt
8
9 def bilinear_interpolate(im, x, y):
10     x = np.asarray(x)
11     y = np.asarray(y)
12
13     x0 = np.floor(x).astype(int)
14     x1 = x0 + 1
15     y0 = np.floor(y).astype(int)
16     y1 = y0 + 1
17
18     x0 = np.clip(x0, 0, im.shape[1]-1);
19     x1 = np.clip(x1, 0, im.shape[1]-1);
20     y0 = np.clip(y0, 0, im.shape[0]-1);
21     y1 = np.clip(y1, 0, im.shape[0]-1);
22
23     Ia = im[y0, x0]
```

Below the terminal, a list of files is shown:

```
user $ ls
README.md
__init__.py
denoisingimp_2b_linear_image.ipynb
denoisingwav_2_wavelet_2d.ipynb
fastmarching_0_implementing.ipynb
introduction_3_image.ipynb
introduction_6_elementary_fr.ipynb
inverse_2_deconvolution_variational.ipynb
nt_solutions
nt_toolbox
nt_toolbox.zip
optim_1_gradient_descent.ipynb
segment.ipynb
todo
wavelet_4_daubechies2d.ipynb
```

# Conclusion

- Un projet très dynamique, bien structuré et financé, supporté par des acteurs majeurs (GitHub, Microsoft, Google, Bloomberg...)
- Une importante communauté venant d'origines variées et bien organisée ([GitHub](#), [Stack Overflow](#), [JupyterDay](#)...)
- Actuellement, Jupyter semble particulièrement intéressant pour :
  - L'enseignement : livres ou supports de cours et d'exercices sous forme de notebooks, utilisation de JupyterHub et de nbgrader pour des TDs, TP, examens
  - La recherche : notebook peut être un bon outil pour la reproductibilité et le partage de résultats et d'outils (articles et livres exécutables, blogs scientifiques...)
  - Le calcul dans les nuages : popularité chez les "data scientists", support par d'importantes plateformes
- Avec les évolutions en cours, pourrait également devenir intéressant pour l'ensemble les étapes du développement logiciel et du traitement de données