3ème 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 metadonnées
- 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)

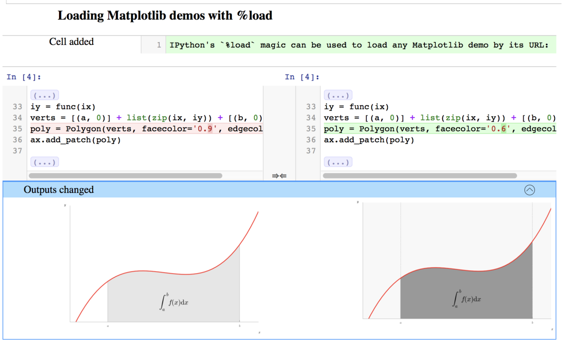
nbdime

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",
90c88
<       "image/png": "iVBORw0KGgoAAAANSUhEUgAABLkAAAMQCAYAAADLj7dlAAAABHNCSVQICAgIfAhki
AAAAAlwSFlz\nAAAWJQAAFiUBSVIk8AAAIABJREFUeJzsvXeYZFd57b12h0maPNJII2lGOaCAkEBCFgozIxkBAp
lY\n1waDyDZg8MX+zMU2F4Mx1x8PwWAwxmBjg4yNi2BfQMa20iiAQFkIjXKWRtJIE3tSz3TXuX+8vV2n\nqyucv
N+9z/o9zzynprvq1D6nqqtqr1prbRNFEQghBBCCCGEEEII8Zkh1wMghBBCCCGEEEIIISQv\nFLkIIYQQQgghB
BCiPdQ5CKEEEIIIYQQQggh3kORixBCCCGEEEIIIYR4D0UuQgghBBCCCGEEOI9\nFLkIIYQQQgghBBCiPdQ5CK
EEEIIIYQQQggh3kORixBCCCGEEEIIIYR4D0UuQgghBBCCCGEEOI9\nFLkIIYQQQgghBBCiPdQ5CKEEEIIIYQQ
```

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=3f7d4e61ee33aaae...>
+  iVBORw0K...<snip base64, md5=1d6960ad89e9de61...>

## modified /cells/14/outputs/0/data/text/plain:
-  <matplotlib.figure.Figure at 0x1110200b8>
+  <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')facecolor='0.6', edgecolor='0.5')
ax.add_patch(poly)
```



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

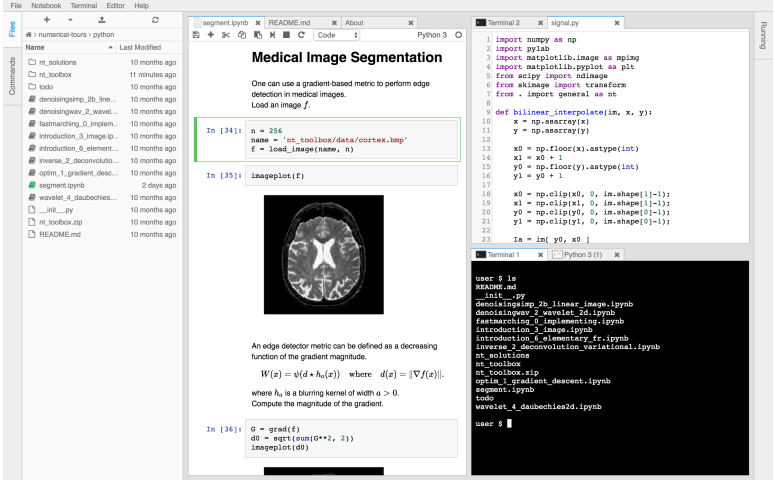
- Authentification et déploiement centralisé
- Possibilité de gestion simplifiée via des conteneurs Docker

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



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, TPs, 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