Outils - Python interactif (IPython) et Jupyter dans VSCode

[Mise à jour le : 4/9/2020]

Sources

- **Site VSCode**
  - Working with the Python Interactive window
  - Working with Jupyter Notebooks in Visual Studio Code
- **Documentation** Jupyter notebook
- **Lectures connexes**
  - Les modules et packages
  - Outils - Installer et utiliser Anaconda avec VSCode

1. Présentation

Jupyter (anciennement IPython Notebook) permet de combiner facilement du texte **Markdown** et du **code source Python** exécutable sur un canevas appelé notebook. Visual Studio Code prend en charge l'utilisation de Jupyter Notebooks de manière native, ainsi que via des **fichiers de code Python**.

Cette page décrit comment :

1. Travailler avec des **cellules de code** de type Jupyter dans un fichier Python.
2. Exécuter du code dans la **fenêtre interactive** Python.
3. Afficher, inspecter et filtrer les **variables** à l'aide de l'explorateur de variables et de la visionneuse de données.
4. **Convertir** les notebooks Jupyter en fichier de code Python.
5. **Déboguer** un notebook Jupyter.
6. **Exporter** un notebook Jupyter (créer un bloc-notes Jupyter à partir d'un fichier python).

Pour utiliser les **blocs-notes Jupyter**, il faut activer un environnement **Anaconda**

dans VSCode ou un autre environnement Python dans lequel le package Jupyter a été installé.

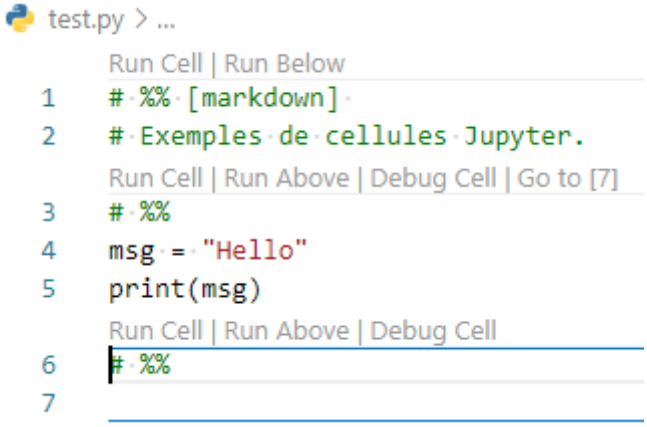
Exemple

2. Jupyter via des fichiers de code Python

2.1 Création d'une cellule de code Jupyter dans un fichier Python

Pour créer une cellule de code dans un fichier Python (**.py**), il suffit d'entrer le commentaire **#%%**. L'en-tête **Run Cell | Run Below | Debug Cell** apparaît après avoir entré ce commentaire.

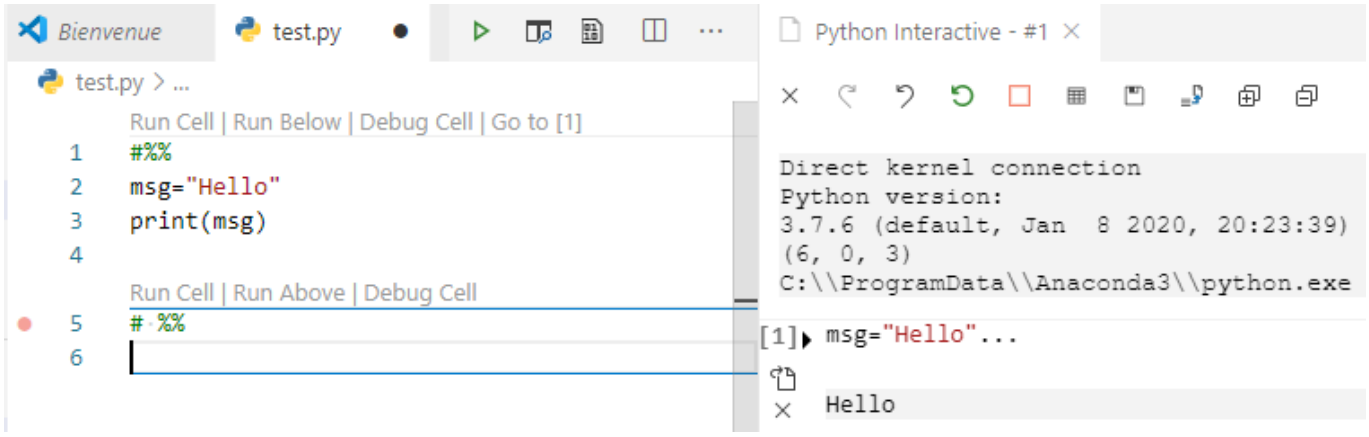
Exemple



2.2 Exécution du code dans la fenêtre interactive

La sélection d'une commande **RUN** dans l'en-tête de la cellule démarre Jupyter, puis exécute la ou les cellules appropriées dans la fenêtre **Python Interactive** :

Exemple



Les cellules de code sont également exécutées à l'aide des combinaisons de touches **[Maj][Entrée]** ou **[Ctrl][Entrée]**.

La fenêtre **Python Interactive** peut également être utilisée comme une console autonome avec du code arbitraire (avec ou sans cellules de code). Pour utiliser la fenêtre comme une console, l'ouvrir à partir de la palette de commandes (**Ctrl + Maj + P** ou **[F1]**) ou du menu contextuel, avec la commande :

Python: Run current file in Python Interactive Window

Les commandes sont alors entrées dans la ligne prévue à cet effet en bas de l'écran et exécutées par la combinaison des touches **[Maj][Entrée]**.

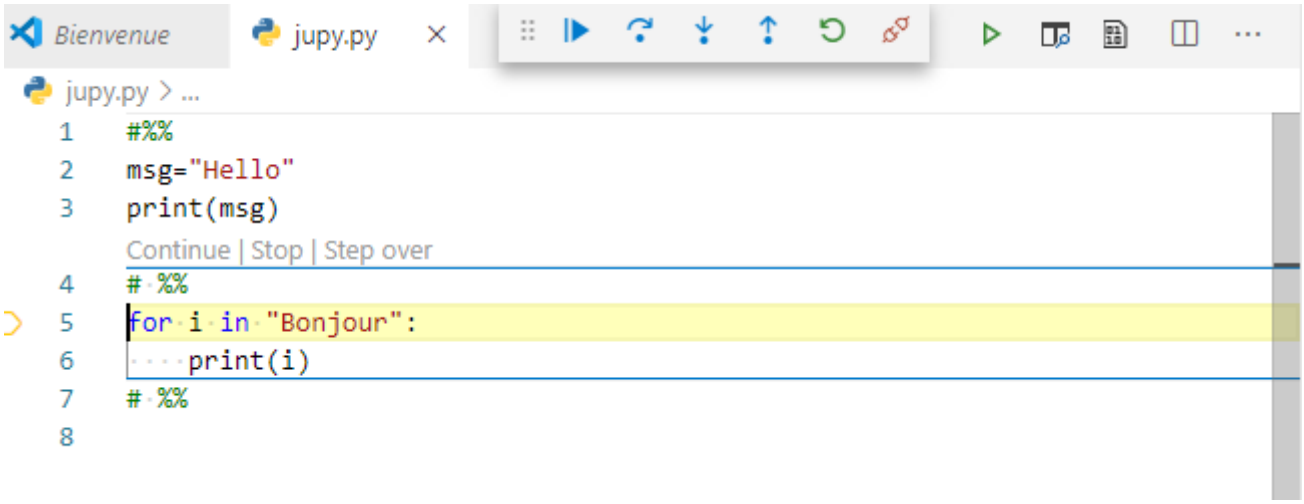
Ligne de commande de la fenêtre interactive

```
[1]  Type code here and press shift-enter to run
```

2.3 Débogage des cellules de code

Le débogage d'une cellule est activé en cliquant sur *Debug Cell* dans l'en-tête de la cellule.

Exemple



- Description de la **barre d'outils**.

	Continuer (**F5**)
	Pas à pas principal (**F10**)
	Pas à pas détaillé (**F11**)
	Pas à pas sortant (**Maj+F11**)
	Redémarrer (**Ctrl+Maj+F5**)
	Déconnecter

2.4. L'explorateur de variables et la visionneuse de données

Il est possible d'**afficher**, d'**inspecter** et de **filtrer** les **variables de la session** Jupyter dans la fenêtre *Python Interactive*.

Pour cela, cliquer sur l'icône  dans la fenêtre *Python Interactive*.

Exemple

2.5. Convertir un notebook Jupyter en fichier de code Python

Lorsqu'un environnement avec Jupyter est installé, on peut ouvrir un fichier de bloc-notes Jupyter (**.ipynb**) dans VSCode, puis le **convertir en code Python** (**.py**) en cliquant sur l'icône ci-dessous.

Une fois le fichier converti, on peut exécuter le code comme avec n'importe quel autre fichier Python et également utiliser le débogueur VSCode. On dispose ainsi de la coloration syntaxique, de la complétion de code, etc. et de toutes les fonctionnalités de débogage (pas à pas, affichage des variables, etc.)

L'**ouverture** et le **débogage** des **blocs-notes** dans **VSCode** sont un moyen pratique pour rechercher et résoudre les bogues, ce qui est difficile à faire directement dans un bloc-notes Jupyter.

2.6 Exporter les cellules dans un notebook Jupyter

On peut également utiliser l'une des commandes ci-dessous pour **exporter** le contenu d'un **fichier Python** (**.py**) **vers un bloc-notes Jupyter** (**.ipynb**). L'accès à ces commandes se fait à partir de la palette de commandes (**Ctrl + Maj + P** ou **[F1]**).

- **Python: Export Current Python File as Jupyter Notebook** : crée un bloc-notes Jupyter à partir du contenu du fichier actuel, en utilisant les délimiteurs **# %%** et **# %% [markdown]** pour spécifier leurs types de cellules respectifs.
- **Python: Export Current Python File and Output as Jupyter Notebook** : crée un bloc-notes Jupyter à partir du contenu du fichier actuel et inclut la sortie des cellules de code.
- **Python: Export Python Interactive window as Jupyter Notebook** : crée un bloc-notes Jupyter à partir du contenu de la fenêtre *Python Interactive*.

2.7 LateX dans une cellule Markdown d'un fichier Python

- **Sources**
  - Wikibooks LaTeX/Mathematics
  - Synthaxe et exemples d'équations sur la documentation Jupyter notebook

L'analyseur **Markdown** inclus dans le bloc-notes Jupyter est compatible avec MathJax.

- **Installation de MathJax**

> Installer Mathjax dans un environnement Anaconda avec conda ou dans Python avec pip

Pour afficher des équations dans la fenêtre **Python Interactive**, préciser **%%latex** dans une cellule Markdown.

Exemple

```
Run Cell | Run Above | Go to [1]
# %% [markdown]
%%latex
$$c = \sqrt{a^2 + b^2}$$
```

Résultat dans la fenêtre Python Interactive

[1] %%latex...

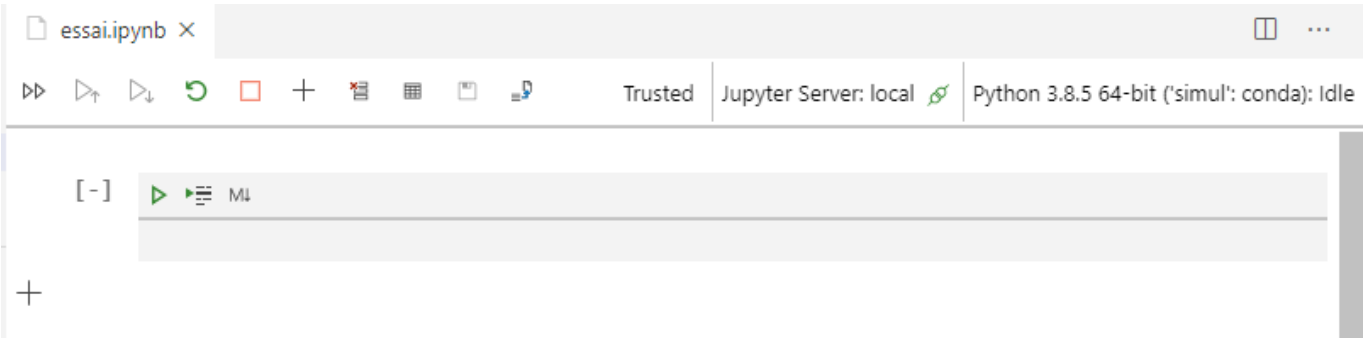
$$c = \sqrt{a^2 + b^2}$$

3.Jupyter natif

3.1 Créer ou ouvrir un bloc-notes

On peut créer un fichier Jupyter dans **VSCode** en lui donnant l'extension .ipynb.

Exemple



Du **code source malveillant** pouvant être contenu dans un bloc-notes, VSCode demande si l'on fait confiance à celui que l'on essaie d'ouvrir. Si c'est le cas : cliquer sur **Trust** dans la boîte de dialogue.

3.2 Enregistrer le bloc-notes

Pour enregistrer le bloc-notes en cours, cliquer sur la disquette dans la barre d'outils.

3.3 Déboguer un notebook Jupyter

Le débogueur Visual Studio Code permet de **parcourir le code**, de **définir des points d'arrêt**, d'examiner l'état des **variables** et d'analyser les problèmes. L'utilisation du débogueur est utile pour chercher et corriger les erreurs dans le code du bloc-notes.

Activer un environnement Python dans lequel Jupyter est installé. Cet environnement doit contenir les paquets **debugpy** et **ipykernel**.

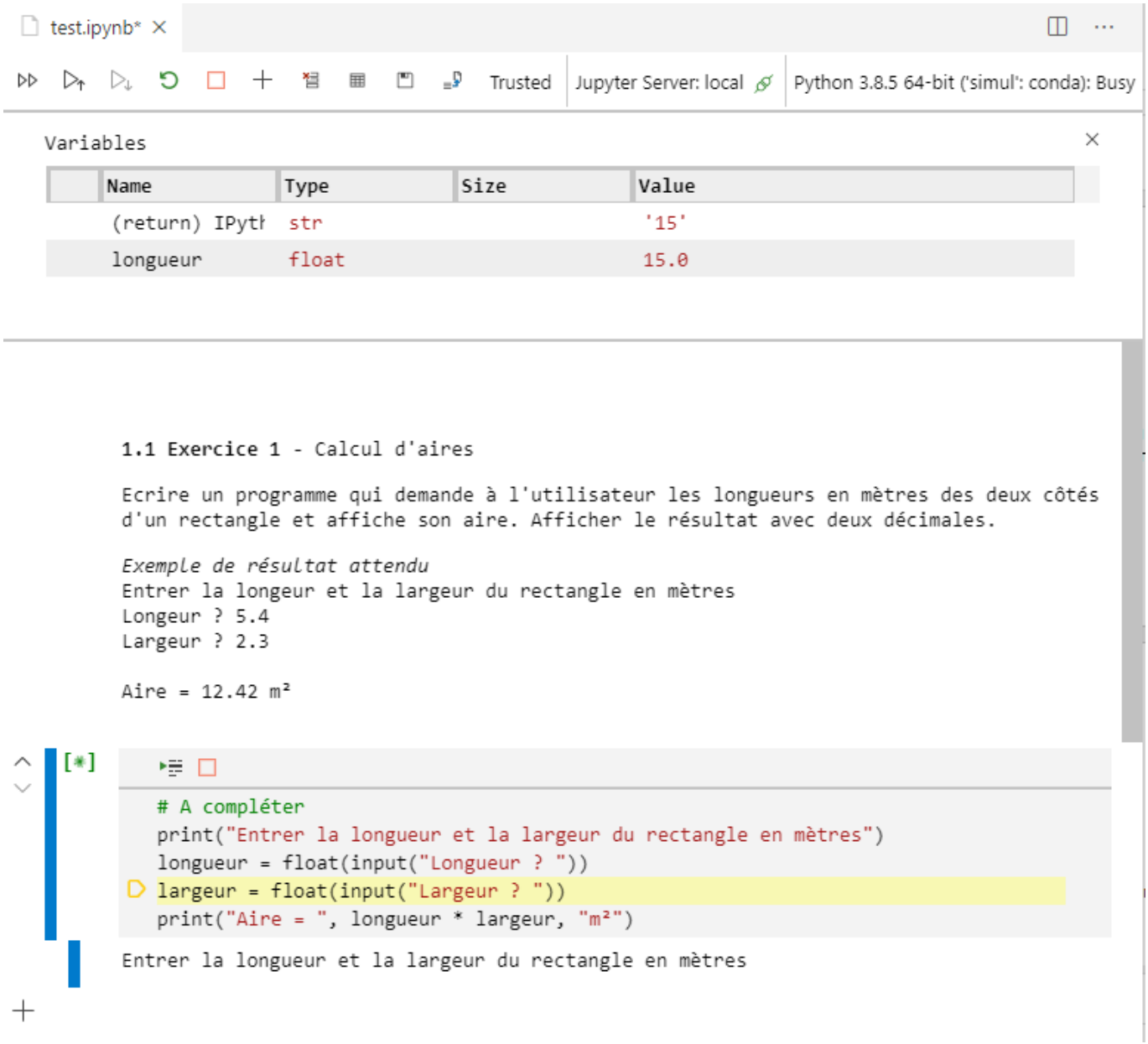
1. **Installations**
   - Installer **debugpy** dans un environnement Anaconda avec conda ou dans Python avec pip
   - Installer **ipykernel** dans un environnement Anaconda avec conda ou dans Python avec pip
2. **Importer** le notebook Jupyter (.ipynb).

```
1.1 Exercice 1 - Calcul d'aires

Ecrire un programme qui demande à l'utilisateur les longueurs
en mètres des deux côtés d'un rectangle et affiche son aire.
Afficher le résultat avec deux décimales.

Exemple de résultat attendu
Entrer la longeur et la largeur du rectangle en mètres
Longeur ? 5.4
Largeur ? 2.3

Aire = 12.42 m²
```

```
[-]  ▷ ▸≡ M↓
# A compléter
print("Entrer la longueur et la largeur du rectangle
en mètres")
longueur = float(input("Longueur ? "))
largeur = float(input("Largeur ? "))
print("Aire = ", longueur * largeur, "m²")
```

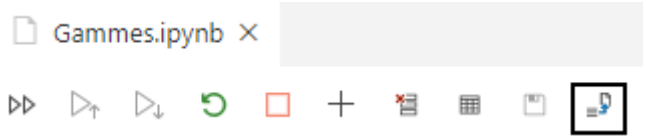
3. **Utiliser** l'icône :
   - ▷ pour exécuter le code d'une cellule.
   - ▸≡ pour exécuter le code d'une cellule **pas à pas** et **visualiser les variables**.

Exemple



3.4 Exporter un bloc-notes Jupyter

Pour exporter un bloc-notes Jupyter sous la forme d'un **fichier Python** (.py), **PDF** ou **HTML**, cliquer sur l'icône de conversion dans la barre d'outils.



3.5 LateX dans Jupyter

Voir le paragraphe 2.7

Exemple