basthon

https://basthon.fr

Documentation

Romain Casati
romain.casati@ac-orleans-tours.fr

Mise à jour le 22/09/2021

Table des matières

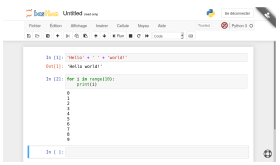
1 Présentation générale

1.1 Basthon, c'est quoi ?

Basthon est l'acronyme de « Bac À Sable pour pyTHON ». Il ressemble au mot « baston », c'est une allusion à la « lutte » que peut parfois représenter l'apprentissage de la programmation, l'écriture d'un code ou son débogage.

Basthon est utilisé pour s'initier au langage de programmation Python 3, sans rien avoir à installer. Il suffit de disposer d'un navigateur (Firefox, Chrome ou Chromium) à jour et d'une connexion à Internet.

Deux interfaces permettent d'utiliser Basthon :



- une interface de type « console », Basthon-Console, disponible à l'adresse
  `https://console.basthon.fr`.
- une interface de type « notebook », Basthon-Notebook, disponible à l'adresse
  `https://notebook.basthon.fr`.

Les deux interfaces sont accessibles depuis `https://basthon.fr`.

Info

Pour plus d'informations sur le projet Basthon, vous pouvez consulter `https://basthon.fr/about.html`. Vous pourrez notamment y lire que Basthon est très respectueux de votre vie privée.

Astuce

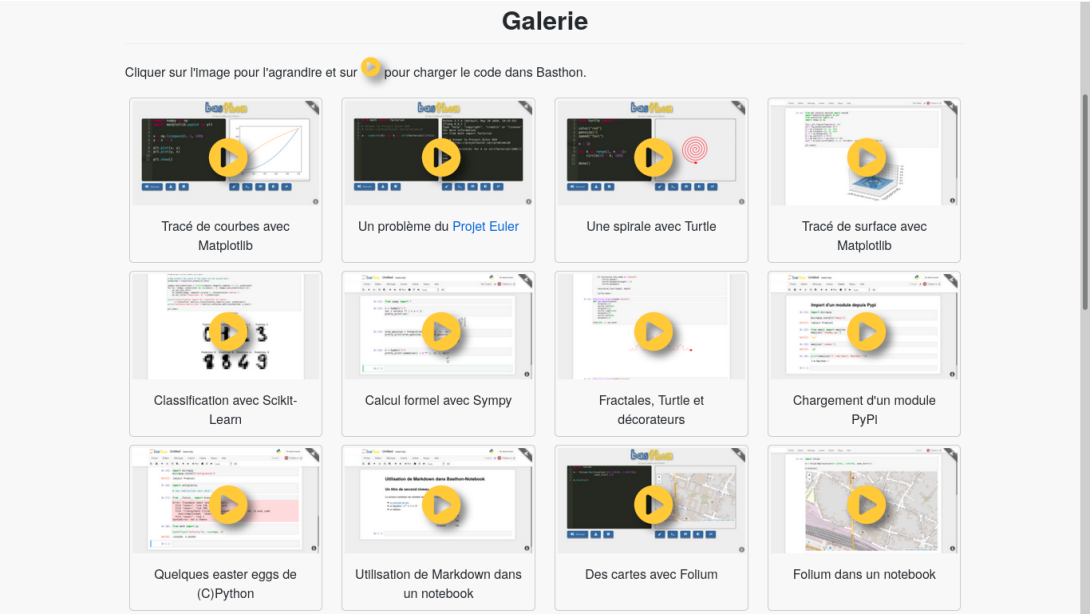
Le mieux pour découvrir Basthon est peut-être de consulter la galerie d'exemples, ici
`https://basthon.fr/galerie.html`
et de jeter un œil à la section 1.2.

Info

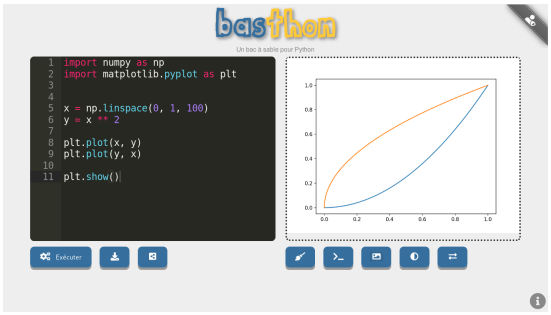
Les deux interfaces de Basthon sont des sites « statiques », c'est-à-dire que c'est votre navigateur qui travaille et non le serveur, qui ne fait que servir des fichiers. Ceci a pour conséquence que Basthon est très peu gourmand pour le serveur qui peut donc supporter un nombre élevé de connexions, surtout si les ressources sont mises en cache.
Si vous avez encore un doute, vous pouvez toujours installer Basthon sur votre propre serveur Web. Pour cela, reportez-vous à la section 5.2.

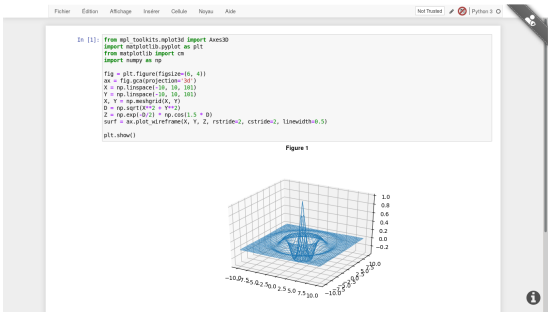
1.2 Des exemples d'utilisation de Basthon

Basthon dispose d'une galerie d'exemples depuis laquelle vous pouvez charger des scripts ou des notebooks illustrant quelques utilisations possibles de Basthon. Elle est accessible ici :
`https://basthon.fr/galerie.html`.

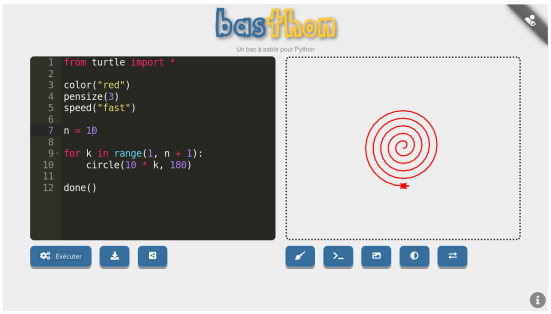


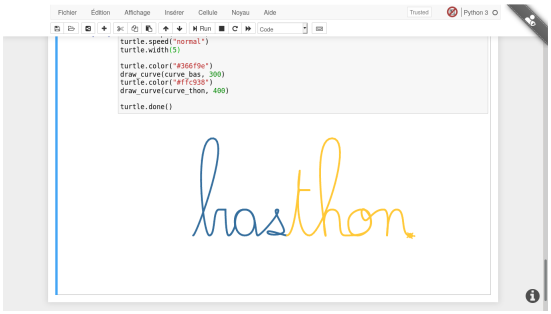
Il vous est par exemple possible de tracer des courbes avec Matplotlib ;



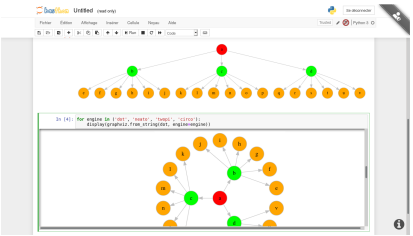


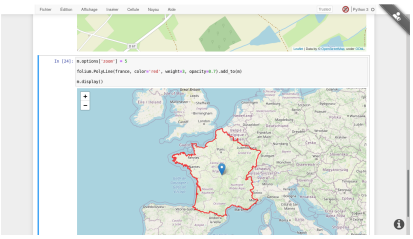
ou de dessiner avec Turtle ;

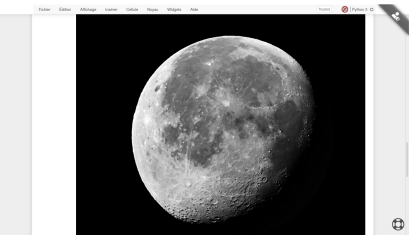




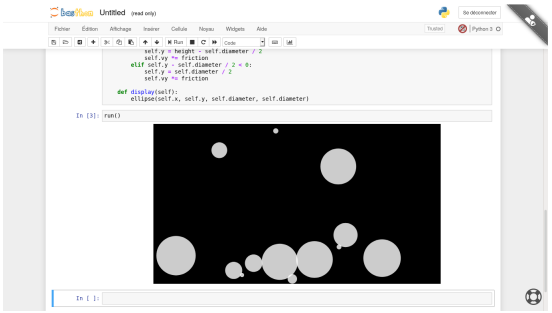
Vous pouvez aussi afficher des graphes (avec Graphviz), des cartes et des itinéaires (avec Folium et Pyroutelib3) ou même faire du traitement d'image (avec PIL).

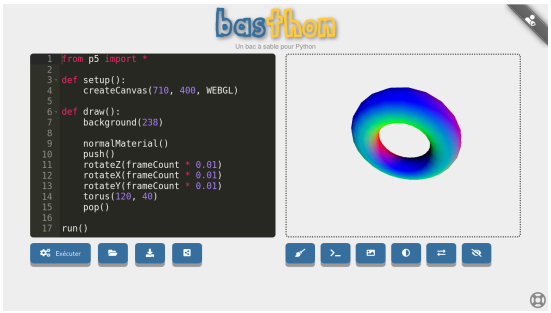




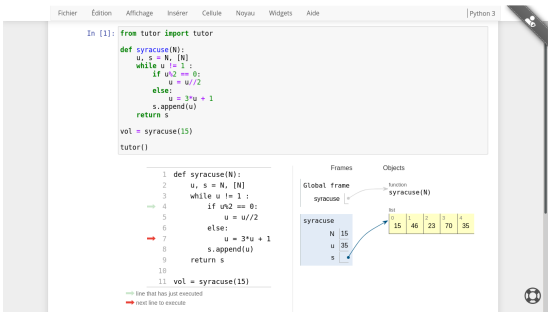


Il est également possible d'utiliser la librairie p5.js pour animer des objets, afficher des modèles en 3D, interagir avec l'utilisateur...

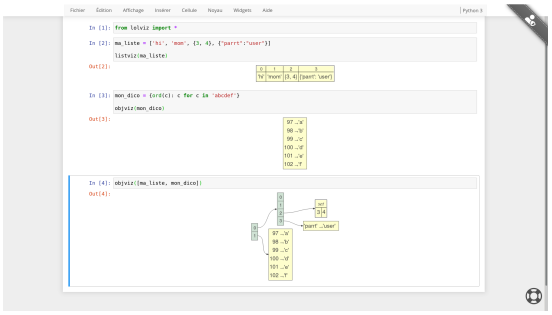


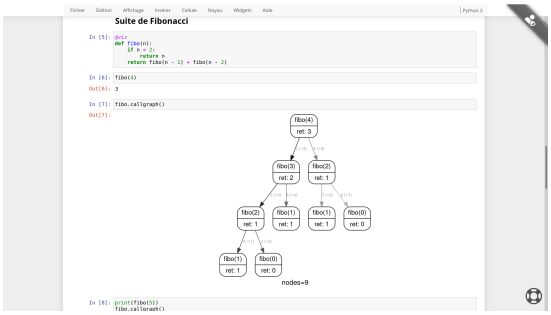


On peut facilement créer des codes QR, avec le module `qrcode` et visualiser schématiquement le déroulé d'un programme grâce à PythonTutor.

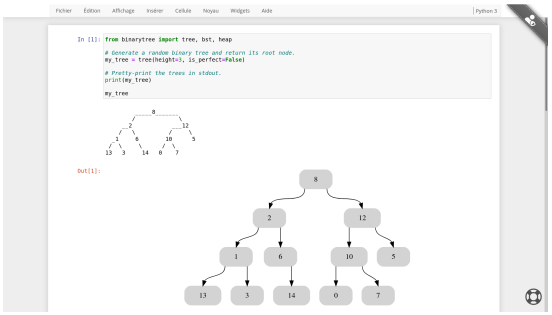


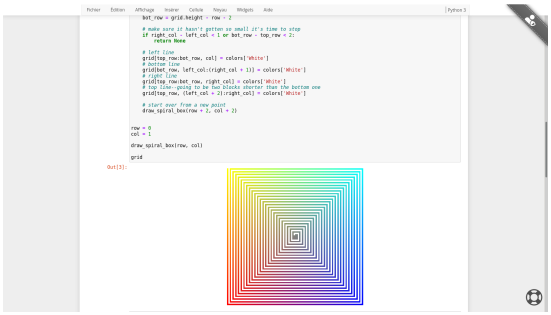
D'ailleurs, si vous aimez PythonTutor, vous apprécierez probablement les modules `lolvizet` `rcviz` qui permettent respectivement de visualiser des structures de données et des appels récursifs de fonctions :





Il y a aussi les `binarytree` pour manipuler et afficher des arbres binaires ou encore `ipythonblocks` pour apprendre Python visuellement :





Info

Retrouvez la documentation spécifique de certains modules dans la section 4.

1.3 Les modules disponibles dans Basthon

Que ce soit dans la console ou dans le notebook, pour lister les modules importables depuis Python, on peut utiliser la commande `help('modules')`. La sortie de cette commande est donnée en annexe, page 17. Retrouvez la documentation spécifique de certains modules dans la section 4.

2 Utilisation de Basthon-Console

2.1 Se repérer dans l'interface

Bouton	Action
Exécuter	Lance l'exécution du contenu de l'éditeur dans la console.
	Ouvre un fichier Python (`*.py`) dans l'éditeur ou permet son import en tant que module ou charge un fichier annexe. Voir les sections 2.4, 2.6 et 2.7
	Télécharge le contenu de l'éditeur dans un fichier Python. Voir la section 2.4.
	Partage le contenu actuel de l'éditeur grâce à une URL. Voir la section 2.5.
	Remise à zéro : la console et la sortie graphique sont nettoyées et l'interprète Python est redémarré.
	Affiche la console (à la place de la sortie graphique).
	Affiche la sortie graphique (à la place de la console) pour visualiser un rendu Turtle, Matplotlib, *etc.*
	Bascule l'affichage en mode lumineux ou sombre.
	Intervertit l'emplacement de la console avec celui de l'éditeur.
	Masque la console (éditeur seul). Une pression supplémentaire masque l'éditeur (console seule). Une autre rétablit l'affichage par défaut.

Info

Certains utilisateurs s'attendent à voir s'ouvrir une fenêtre contenant la sortie graphique (pour Turtle ou Matplotlib par exemple). Ce n'est pas le cas. Il faut l'afficher grâce au bouton  .

Astuce

Il peut être parfois pratique d'augmenter ou de diminuer la taille de la police de l'éditeur et de la console. Utilisez pour cela la combinaison Ctrl+molette ou encore Ctrl+ + .

Astuce

Le mode sombre (par défaut) est apaisant pour les yeux mais n'est pas adapté pour l'utilisation d'un vidéo-projecteur car il est peu contrasté. Il est alors conseillé d'utiliser le mode lumineux avec le bouton  .

2.2 Utilisation de l'éditeur

L'éditeur dispose de la coloration syntaxique et d'un système de complétion (rudimentaire). Il s'utilise de manière traditionnelle avec de nombreux raccourcis. En particulier, les touches Ctrl+Z permettent d'annuler une action, Ctrl+A de sélectionner tout le contenu, Ctrl+F pour rechercher, Ctrl+, pour modifier les paramètres de l'éditeur, *etc.* Les fonctions et les variables déclarées dans l'éditeur sont disponibles dans la console et inversement.

Astuce

Pensez à régler la taille de la police avec la combinaison Ctrl+molette ou Ctrl+ + .

2.3 Utilisation de la console

La console s'utilise comme celle de l'interprète Python classique. On lance l'exécution d'une instruction avec la touche ↵ et on saute une ligne avec ⇧+↵. La console dispose d'un historique, accessible avec les touches ↑ et ↓. Les fonctions et les variables déclarées dans l'éditeur sont disponibles dans la console et inversement.

Astuce

Pensez à régler la taille de la police avec la combinaison Ctrl+molette ou Ctrl+ + .

2.4 Enregistrer et ouvrir un script

Pour enregistrer le contenu de l'éditeur dans un fichier `*.py`, il suffit d'utiliser le bouton  .

Pour charger le contenu d'un fichier `*.py` dans l'éditeur, il faut utiliser le bouton  et choisir « charger dans l'éditeur ». Ce bouton sert aussi à charger des ressources annexes comme des fichiers texte, des images ou même des modules Python. On pourra consulter les sections 2.6 et 2.7 à ce sujet.

2.5 Partager un script

Basthon-Console permet de partager un script grâce à une URL. Ainsi, vous n'avez plus qu'à communiquer ce lien aux personnes intéressées. Deux cas se présentent :

- soit votre script est de petite taille et vous pouvez alors utiliser le bouton de partage  qui va créer une URL contenant votre script et que vous pourrez coller où bon vous semble pour la partager [1] ;
- soit votre script est trop conséquent et il est accessible sur le Web (sur votre propre serveur, sur GitHub, GitLab ou autre), et il vous suffit d'indiquer son chemin avec le paramètre `from` de l'URL, comme ceci :

`https://console.basthon.fr/?from=url/vers/mon_script.py`

1. L'URL ainsi créée utilise le paramètre `script` pour soumettre le script à Basthon.

Exemple

Par exemple, si votre script se trouve à l'adresse

`https://raw.githubusercontent.com/python/cpython/master/Tools/freeze/hello.py`

vous pouvez partager son ouverture dans Basthon-Console avec l'URL

`https://console.basthon.fr/?from=https://raw.githubusercontent.com/python/cpython/master/Tools/freeze/hello.py`

Info

C'est grâce au paramètre `from` que sont partagés les exemples de la galerie. Dans ce cas, les URL sont un peu plus simples car les scripts sont chargés depuis le site de Basthon, comme ceci

`https://console.basthon.fr/?from=examples/courbes.py`

Astuce

Il est aussi possible de récupérer un fichier annexe ou d'importer un module lors du partage de script par l'une des deux méthodes décrites ci-dessus. Utilisez pour cela les paramètres d'URL `aux` et `module`. Plus de détails dans les sections 2.6 et 2.7.

Attention

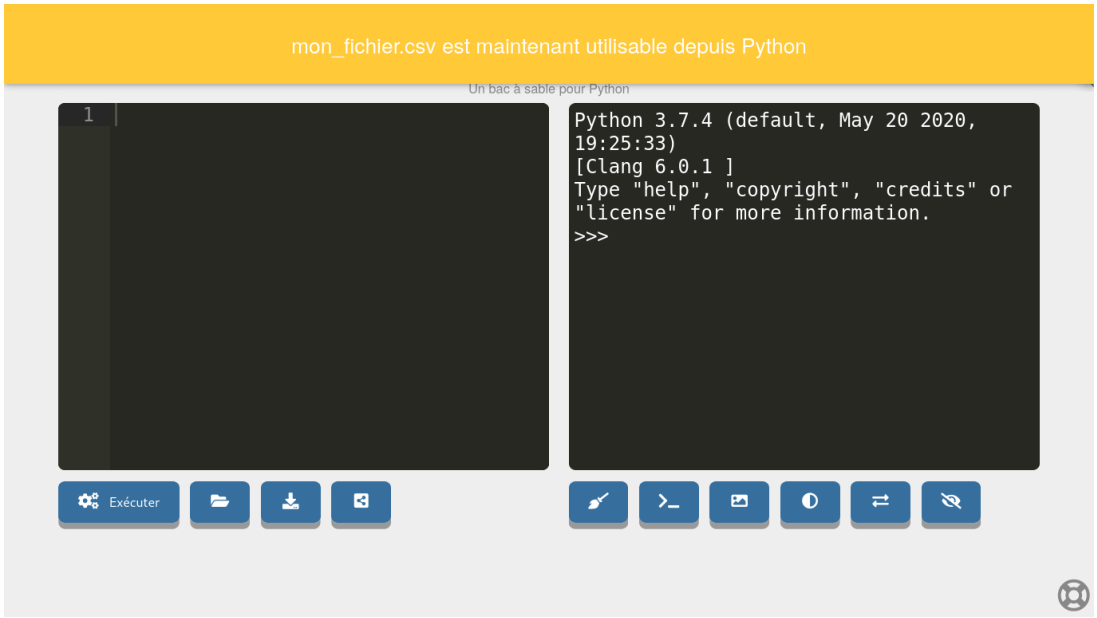
L'utilisation du paramètre `from` nécessite une requête entre différentes origines (CORS en anglais) puisque le site qui détient le script n'est, le plus souvent, pas celui de Basthon. Ceci peut échouer si le serveur n'autorise pas ce type de requête.

Astuce

Si vous déposez vos fichiers sur des plateformes comme GitHub ou GitLab (ou mieux, Framagit), il sera très facile de les utiliser avec Basthon.

2.6 Utiliser des fichiers annexes

Pour travailler avec une image, un document textuel, un fichier CSV, *etc.* stocké sur votre machine, vous pouvez le charger dans Basthon grâce au bouton  . La fin du chargement du fichier est notifiée à l'utilisateur comme ci-dessous.



Il vous suffira ensuite de l'ouvrir depuis Python, comme s'il était dans le répertoire courant.

Il est aussi possible de récupérer des fichiers annexes déposés sur le Web au chargement de la page grâce au paramètre d'URL `aux`.

Exemple

Par exemple, pour charger le fichier auxiliaire `mon_fichier.csv` depuis le site `https://www.example.net/`, il vous suffit d'utiliser l'URL

```
https://console.basthon.fr/?aux=https://www.example.net/mon_fichier.csv
```

Astuce

Pour connaître la liste des fichiers disponibles, vous pouvez interroger le contenu du répertoire courant comme ceci

```
>>> import os
>>> os.listdir()
['mon_fichier.csv', 'lib', 'proc', 'dev', 'home', 'tmp']
```

On voit ici que l'utilisateur a chargé le fichier `mon_fichier.csv`.

Info

Le paramètre `aux` est cumulable, c'est-à-dire que vous pouvez charger plusieurs ressources en utilisant plusieurs fois ce paramètre dans l'URL. Il est aussi utilisable avec les autres paramètres, comme `from`, `script` et `module`.

Attention

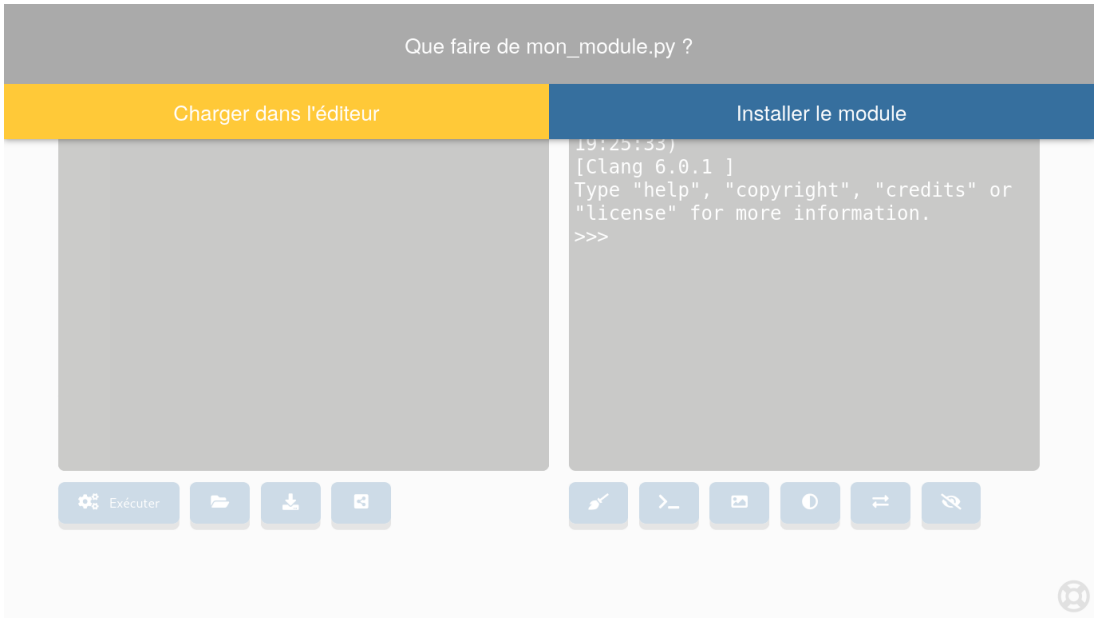
Comme `from`, l'utilisation du paramètre `aux` nécessite une requête entre différentes origines (CORS en anglais) puisque le site qui détient le fichier n'est, le plus souvent, pas celui de Basthon. Ceci peut échouer si le serveur n'autorise pas ce type de requête.

Info

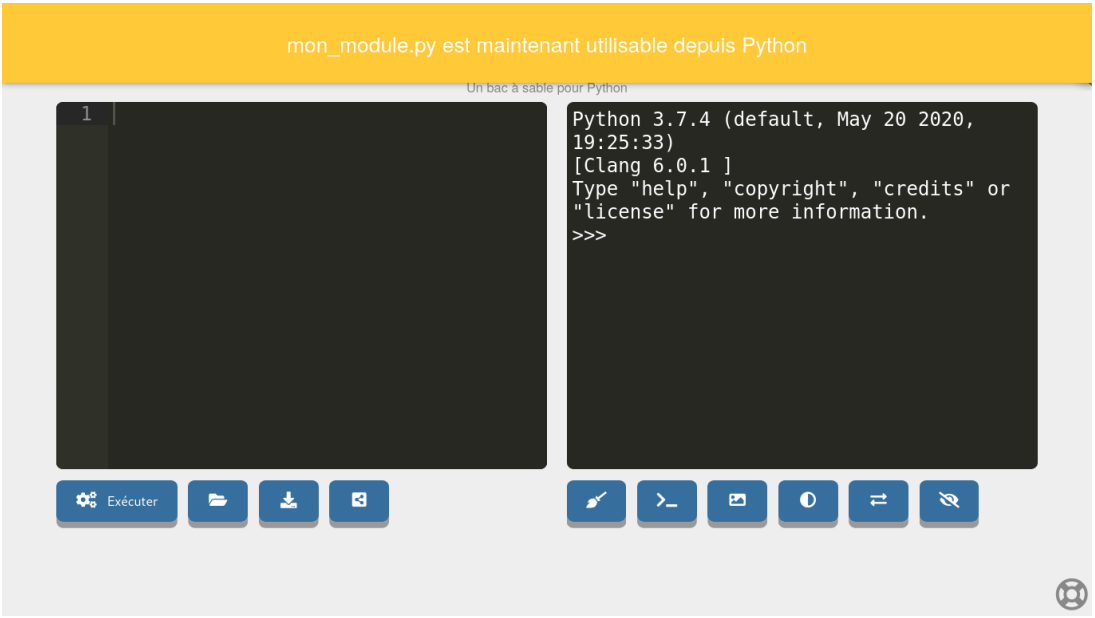
Si au lieu de charger un fichier annexe, vous voulez, à l'inverse, le récupérer, reportez-vous à la section 5.1.

2.7 Importer son propre module

Pour pouvoir importer votre propre module (contenu dans un fichier `*.py`) depuis Python, il vous suffit de l'ouvrir avec le bouton  . Répondez à la question ci-dessous que vous voulez installer le module.



Basthon s'occupe alors de charger les dépendances le cas échéant puis vous notifie de la disponibilité du module.



Il vous est alors possible d'importer votre module avec `import mon_module`.

Il est aussi possible de récupérer des modules déposés sur le Web au chargement de la page grâce au paramètre d'URL `module`.

Exemple

Par exemple, pour charger le module `mon_module.py` depuis le site `https://www.example.net/`, il vous suffit d'utiliser l'URL

`https://console.basthon.fr/?module=https://www.example.net/mon_module.py`

Astuce

La commande `help('modules')` recense l'ensemble des modules importables. Y compris les modules chargés de cette façon.

Info

Comme `aux`, le paramètre `module` est cumulable, c'est-à-dire que vous pouvez charger plusieurs modules en utilisant plusieurs fois ce paramètre dans l'URL. Il est aussi utilisable avec les autres paramètres, comme `from`, `script` et `aux`.

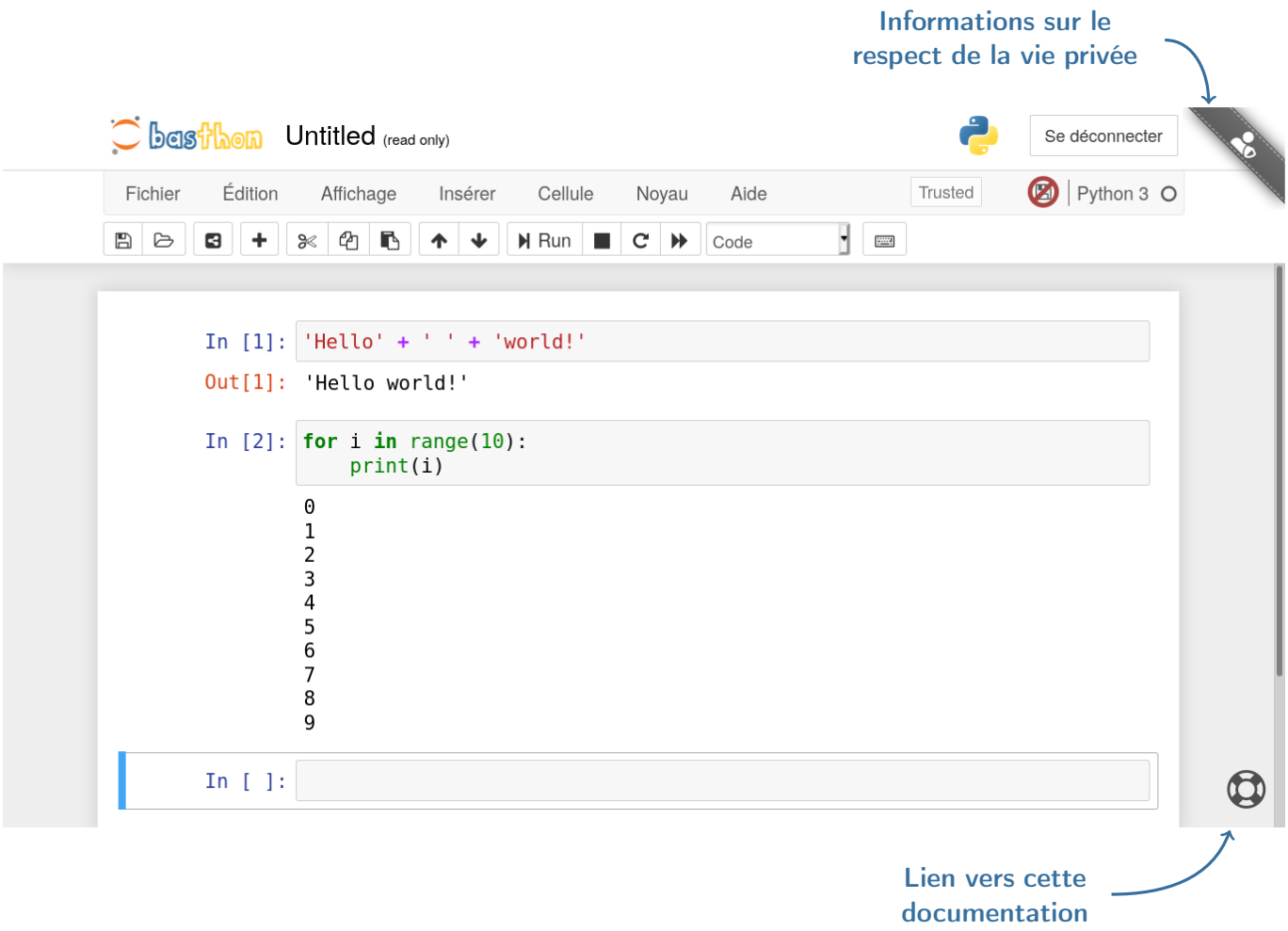
Attention

Comme `from` et `aux`, l'utilisation du paramètre `module` nécessite une requête entre différentes origines (`CORS` en anglais) puisque le site qui détient le module n'est, le plus souvent, pas celui de Basthon. Ceci peut échouer si le serveur n'autorise pas ce type de requête.

3 Utilisation de Basthon-Notebook

3.1 Se repérer dans l'interface

Basthon-Notebook utilise une version modifiée du notebook de Jupyter. Une grande partie des fonctionnalités est commune et déjà bien documentée dans l'aide de Jupyter Notebook. Nous détaillons ici uniquement les fonctionnalités spécifiques à la version de Basthon.



Bouton	Action
	Télécharge le contenu du notebook dans un fichier `*.ipynb`. Voir la section 3.2.
	Ouvre un fichier notebook (`*.ipynb`) ou un fichier `*.py` ou permet son import en tant que module ou charge un fichier annexe. Voir les sections 3.2, 3.5 et 3.6
	Partage le contenu actuel du notebook grâce à une URL. Voir la section 3.4.

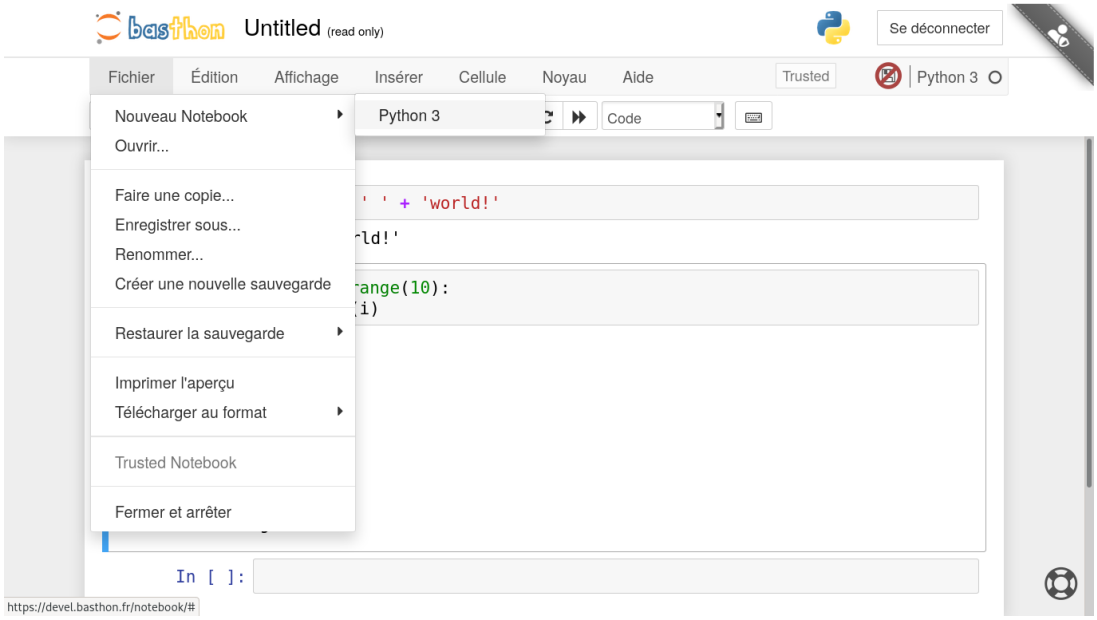
3.2 Enregistrer et ouvrir un notebook

Pour enregistrer le notebook dans un fichier `*.ipynb`, il suffit d'utiliser le bouton  .

Pour charger un notebook depuis un fichier `*.ipynb`, il faut utiliser le bouton  . Ce bouton sert aussi à charger des ressources annexes comme des fichiers texte, des images ou même des modules Python. On pourra consulter les sections 3.5 et 3.6 à ce sujet.

3.3 Ouvrir un notebook vierge

Il est possible d'ouvrir un notebook vide depuis le menu Fichier > Nouveau Notebook > Python 3, comme ci-dessous.



3.4 Partager un notebook

Basthon-Notebook permet de partager un document grâce à une URL. Ainsi, vous n'avez plus qu'à communiquer ce lien aux personnes intéressées. Deux cas se présentent :

- soit votre notebook est de petite taille et vous pouvez alors utiliser le bouton de partage  qui va créer une URL contenant votre notebook et que vous pourrez coller où bon vous semble pour la partager [2] ;
- soit votre notebook est trop conséquent et il est accessible sur le Web (sur votre propre serveur, sur GitHub, GitLab ou autre), et il vous suffit d'indiquer son chemin avec le paramètre `from` de l'URL, comme ceci :

`https://notebook.basthon.fr/?from=url/vers/mon_notebook.ipynb`


Exemple

Par exemple, si votre notebook se trouve à l'adresse

`https://www.exemple.net/mon_notebook.ipynb`

vous pouvez partager son ouverture dans Basthon-Notebook avec l'URL

`https://notebook.basthon.fr/?from=https://www.exemple.net/mon_notebook.ipynb`


Info

C'est grâce au paramètre `from` que sont partagés les exemples de la galerie. Dans ce cas, les URL sont un peu plus simples car les scripts sont chargés depuis le site de Basthon, comme ceci

`https://notebook.basthon.fr/?from=examples/3d-plot.ipynb`

Astuce

Il est aussi possible de récupérer un fichier annexe ou d'importer un module lors du partage de notebook par l'une des deux méthodes décrites ci-dessus. Utilisez pour cela les paramètres d'URL `aux` et `module`. Plus de détails dans les sections 3.5 et 3.6.


Attention

L'utilisation du paramètre `from` nécessite une requête entre différentes origines (`CORS` en anglais) puisque le site qui détient le notebook n'est, le plus souvent, pas celui de Basthon. Ceci peut échouer si le serveur n'autorise pas ce type de requête.

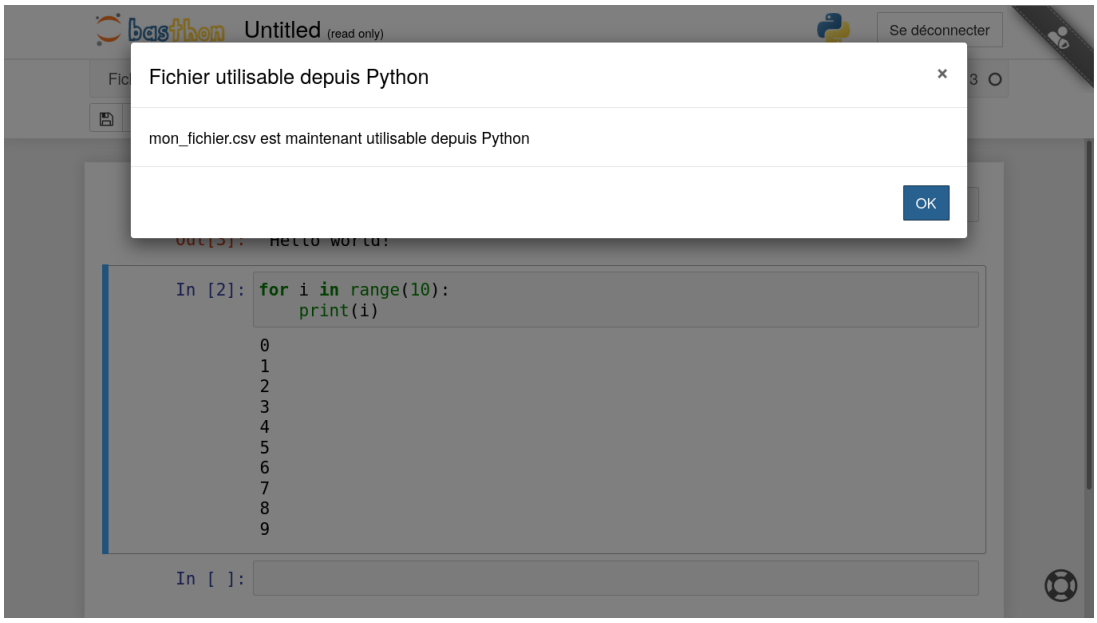
Astuce

Si vous déposez vos fichiers sur des plateformes comme GitHub ou GitLab (ou mieux, Framagit), il sera très facile de les utiliser avec Basthon.

2. L'URL ainsi créée utilise le paramètre `ipynb` pour soumettre le notebook à Basthon.

3.5 Utiliser des fichiers annexes

Pour travailler avec une image, un document textuel, un fichier CSV, *etc.* stocké sur votre machine, vous pouvez le charger dans Basthon grâce au bouton  . La fin du chargement du fichier est notifiée à l'utilisateur comme ci-dessous.



Il vous suffira ensuite de l'ouvrir depuis Python, comme s'il était dans le répertoire courant.

Il est aussi possible de récupérer des fichiers annexes déposés sur le Web au chargement de la page grâce au paramètre d'URL `aux`.

Exemple

Par exemple, pour charger le fichier auxiliaire `mon_fichier.csv` depuis le site `https://www.example.net/`, il vous suffit d'utiliser l'URL

`https://notebook.basthon.fr/?aux=https://www.example.net/mon_fichier.csv`

Astuce

Pour connaître la liste des fichiers disponibles, vous pouvez interroger le contenu du répertoire courant comme ceci

```
import os
os.listdir()
```

```
['mon_fichier.csv', 'lib', 'proc', 'dev', 'home', 'tmp']
```

On voit ici que l'utilisateur a chargé le fichier `mon_fichier.csv`.

Info

Le paramètre `aux` est cumulable, c'est-à-dire que vous pouvez charger plusieurs ressources en utilisant plusieurs fois ce paramètre dans l'URL. Il est aussi utilisable avec les autres paramètres, comme `from`, `ipynb` et `module`.

Attention

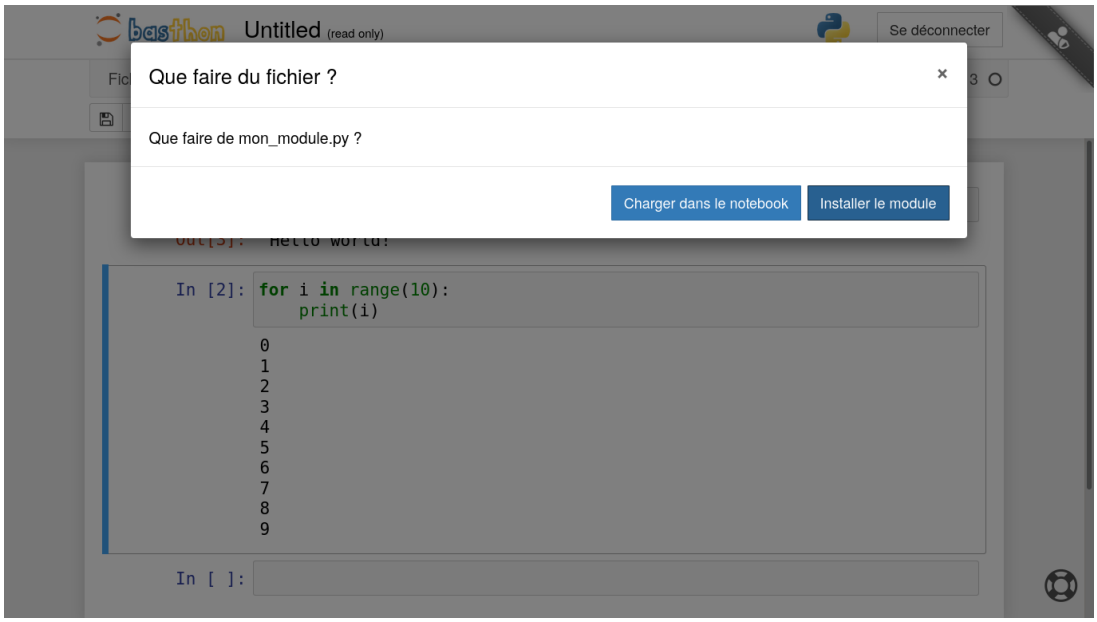
Comme `from`, l'utilisation du paramètre `aux` nécessite une requête entre différentes origines (`CORS` en anglais) puisque le site qui détient le fichier n'est, le plus souvent, pas celui de Basthon. Ceci peut échouer si le serveur n'autorise pas ce type de requête.

Info

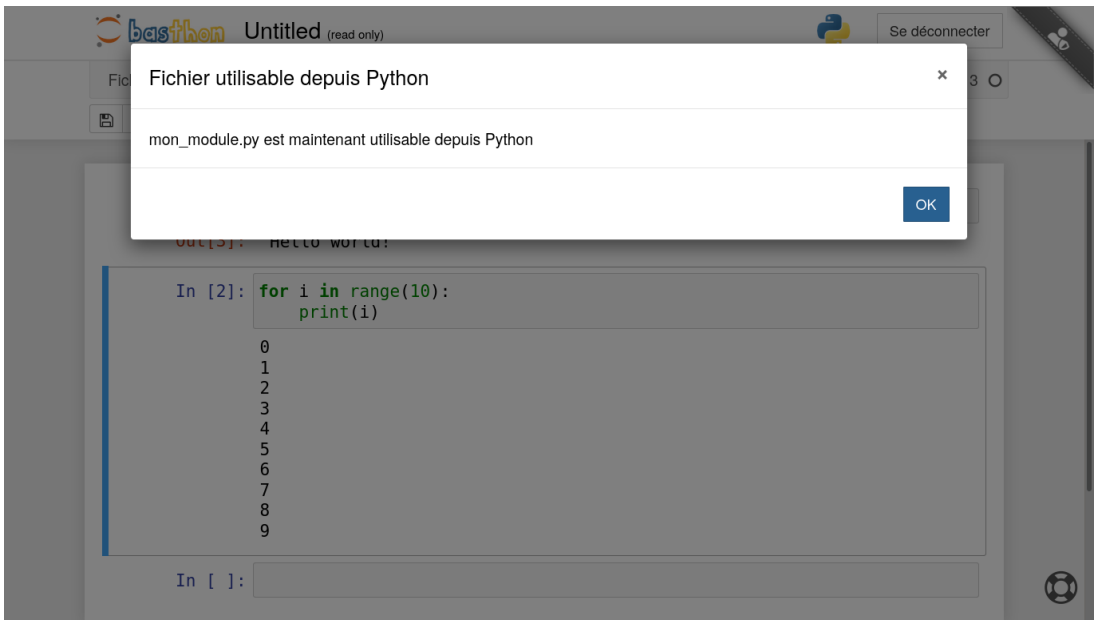
Si au lieu de charger un fichier annexe, vous voulez, à l'inverse, le récupérer, reportez-vous à la section 5.1.

3.6 Importer son propre module

Pour pouvoir importer votre propre module (contenu dans un fichier `*.py`) depuis Python, il vous suffit de l'ouvrir avec le bouton  . Répondez à la question ci-dessous que vous voulez « installer le module ».



Basthon s'occupe alors de charger les dépendances le cas échéant puis vous notifie de la disponibilité du module.



Il vous est alors possible d'importer votre module avec `import mon_module`.

Il est aussi possible de récupérer des modules déposés sur le Web au chargement de la page grâce au paramètre d'URL `module`.

Exemple

Par exemple, pour charger le module `mon_module.py` depuis le site https://www.example.net/, il vous suffit d'utiliser l'URL

https://notebook.basthon.fr/?module=https://www.example.net/mon_module.py

Astuce

La commande `help('modules')` recense l'ensemble des modules importables. Y compris les modules chargés de cette façon.

Info

Comme `aux`, le paramètre `module` est cumulable, c'est-à-dire que vous pouvez charger plusieurs modules en utilisant plusieurs fois ce paramètre dans l'URL. Il est aussi utilisable avec les autres paramètres, comme `from`, `ipynb` et `aux`.

Attention

Comme `from` et `aux`, l'utilisation du paramètre `module` nécessite une requête entre différentes origines (`CORS` en anglais) puisque le site qui détient le module n'est, le plus souvent, pas celui de Basthon. Ceci peut échouer si le serveur n'autorise pas ce type de requête.

4 Utilisation particulière de certains modules avec Basthon

Certains modules s'utilisent, avec Basthon, un peu différemment d'avec l'interprète classique. Nous détaillons ces différences ici.

4.1 Turtle

Pour utiliser Turtle dans Basthon, il est impératif de terminer son dessin avec l'instruction `turtle.done()` (ou `turtle.mainloop()`), sinon, rien ne s'affichera.

Aussi, on peut désactiver l'animation en début de dessin avec `turtle.animation("off")` pour obtenir directement le résultat final du tracé (on la réactive avec `turtle.animation("on")`).

Le résultat peut être téléchargé avec `turtle.download("mon_dessin.svg")` mais attention : peu d'applications supportent les SVG animés. Il convient donc de désactiver l'animation pour obtenir un fichier plus portable.

Le SVG est accessible avec `turtle.svg()` et peut être sauvegardé (dans le système de fichiers virtuel) grâce à `turtle.save(mon_dessin.svg)`.

4.2 p5.js

Pour utiliser `p5.js`, il est pratique d'écrire `from p5 import *` . Malheureusement, certaines fonctions de `p5.js` qui ont le même nom que des fonctions du langage. C'est le cas pour `abs`, `float`, `hex`, `int`, `max`, `min`, `pow`, `round` et `str` qui se comportent (presque) de la même manière et qui ne devraient pas poser de problème. En revanche, `filter`, `map` et `set` sont très différentes de leurs homonymes. Dans ce cas, pour savoir si l'utilisateur souhaite appeler les fonctions de Python ou celles de `p5.js`, Basthon va regarder le nombre et le type des arguments.

Certaines fonctions de `p5.js` modifient la valeurs des variables internes. C'est le cas par exemple de `createCanvas` qui va modifier les valeurs de `width` et `height`. Dans Basthon, l'état de ces variables n'est mis à jour qu'au moment de l'appel des fonctions `setup` et `draw`. Il est possible de forcer leur mise à jour avec l'instruction `update_variables()`.

4.3 Graphviz

Lors de l'appel de la fonction `render` sur un graphe, Basthon vous proposera d'enregistrer le fichier, comme si vous veniez de le télécharger.

Astuce

Lors de l'appel de la fonction `render` sur un graphe, pensez à utiliser l'argument `scale` pour augmenter la résolution.

4.4 PythonTutor

Pour utiliser PythonTutor, il faut appeler la fonction `tutor` du module `tutor`, comme ceci par exemple :

```
from tutor import tutor  # ou 'from tutor import *' ou simplement
                         # 'import tutor' et appeler tutor.tutor()
# mettre ici le code à tutorer
a = 5
a = a + 1

tutor()  # peut aussi être mis avant le code à tutorer
         # les options de PythonTutor peuvent être passées en paramètre
```


Attention

Le code exécuté (le script dans le cas de la console et la cellule dans le cas du notebook) doit contenir toute l'information utile à PythonTutor pour fonctionner : déclarations de variables, de fonctions, *etc.* Toutefois, l'import de `tutor` lui-même peut être effectué anterieurement.

4.5 QRCode

Quelques fonctionnalités ont été ajoutées au module `qrcode`. Il est par exemple possible de télécharger l'image d'un code sans la sauvegarder.

```
import qrcode

mon_code = qrcode.make("Un super message !")

mon_code.download()
```

Il est aussi possible de choisir facilement le format (`png`, par défaut, ou `svg`) avec la fonction `make` grâce au paramètre `format`.

```
import qrcode

mon_code_svg = qrcode.make("Un super message !", format='svg')

mon_code_svg.show()
```

4.6 rcviz

L'API de `rcviz` a été repensée pour être plus ergonomique et plus souple. En effet, dans Basthon, ce module s'utilise comme ceci :

```
from rcviz import viz

@viz
def somme(n):
    if n < 1:
        return 0
    return n + somme(n - 1)

print(somme(4))
somme.callgraph()  # utiliser .render() pour télécharger le graphe ainsi obtenu ou
                   # basthon.display pour le visualiser (inutile dans le notebook)
```

Le décorateur `rcviz.viz` permet ainsi de récupérer le graphe d'appels depuis la fonction elle-même. De plus, un nouvel appel de la fonction déclenchera automatiquement la création d'un nouveau graphe.

5 Utilisation avancée

5.1 Récupérer un fichier

Nous avons vu qu'il était possible d'enregistrer le script ou le notebook en cours d'édition mais comment récupérer un fichier annexe que l'on a modifié ou encore une image que l'on a créé avec Python ? C'est assez simple, après avoir importé le module `basthon` avec `import basthon`, il suffit d'utiliser la fonction

`basthon.download('chemin/vers/mon/fichier.truc')`.

Ceci va ouvrir une fenêtre vous permettant de sauvegarder ce fichier où vous voulez sur votre machine.

5.2 Héberger Basthon sur son propre serveur Web

La procédure pour installer Basthon sur son propre serveur Web est très simple, il suffit de décompresser une archive. Elle est détaillée ici :

https://basthon.fr/about.html#install

Info

Même si vous installez Basthon sur votre propre serveur, Pyodide (l'interprète de Basthon) est toujours récupéré depuis le CDN officiel. Pour soulager ce serveur, vous pouvez aussi servir votre propre version de Pyodide. Pour cela, il suffit de placer la version de 0.16.1 Pyodide dans le dossier

- `assets/js/basthon-kernel/pyodide/` pour Basthon-Console,
- `static/basthon-kernel/pyodide/` pour Basthon-Notebook.

Astuce

⚠ La fonctionnalité discutée ici n'est pas encore implémentée. ⚠
Si vous souhaitez disposer de modules supplémentaires dans votre installation de Basthon, il vous suffit de placer les fichiers `*.py` dans le dossier

- `assets/js/basthon-kernel/site-packages/` pour Basthon-Console,
- `static/basthon-kernel/site-packages/` pour Basthon-Notebook.

5.3 Quelle est la dernière version de Basthon ?

Pour savoir quelle est la dernière version de Basthon-Console, vous pouvez consulter

https://console.basthon.fr/version.

Il est intéressant de comparer cet identifiant de version avec celui de votre propre installation de Basthon-Console (disponible à la même adresse) pour savoir si vous possédez une version à jour.

Pour le notebook, il suffit de se rendre ici

https://notebook.basthon.fr/version

Crédits

- Le logo Basthon a été réalisé avec la police KeyTabMetal publié par Tension Type sur dafont.com.
- Les icônes utilisées sont celles de la police Fontawesome.

Info

Plus d'informations sur les ressources tierces utilisées dans Basthon ici.

Annexe

Liste des modules disponibles dans Basthon

Voici la sortie de la commande `help('modules')`, permettant de lister les modules importables avec Basthon..

```
Please wait a moment while I gather a list of all available modules...

Bio               binhex            keyword           requests
IPython           bisect            kiwisolver        rlcompleter
PIL               bleach            linecache         runpy
__future__        bs4               locale            sched
_abc              builtins          logging           scipy
```

```
_ast                bz2                 lolviz              secrets
_bisect             cProfile            lxml                select
_blake2             calendar            lzma                selectors
_bootlocale         cgi                 mailbox             setuptools
_bz2                cgitb               mailcap             shelve
_codecs             chunk               markupsafe          shlex
_codecs_cn          cloudpickle         marshal             shutil
_codecs_hk          cmath               math                signal
_codecs_iso2022     cmd                 matplotlib          site
_codecs_jp          code                micropip            sitecustomize
_codecs_kr          codecs              mimetypes           six
_codecs_tw          codeop              mmap                skimage
_collections        collections         mne                 sklearn
_collections_abc    colorsys            modulefinder        smtpd
_compat_pickle      compileall          more_itertools      smtplib
_compression        concurrent          mpl_toolkits        sndhdr
_contextvars        configparser        mpmath              socket
_crypt              contextlib          msgpack             socketserver
_csv                contextvars         multiprocessing     soupsieve
_datetime           copy                netrc               sqlite3
_decimal            copyreg             networkx            sre_compile
_dummy_thread       crypt               nltk                sre_constants
_functools          cssselect           nntplib             sre_parse
_heapq              csv                 nose                ssl
_imp                cycler              ntpath              stat
_io                 cytoolz             nturl2path          statistics
_ivp                dataclasses         numbers             statsmodels
_json               datetime            numcodecs           string
_locale             dateutil            numpy               stringprep
_lsprof             decimal             opcode              struct
_markupbase         decorator           operator            subprocess
_md5                difflib             optparse            sunau
_multibytecodec     dis                 os                  symbol
_operator           distlib             osmiter             sympy
_pickle             distutils           p5                  symtable
_posixsubprocess    doctest             packaging           sys
_py_abc             docutils            pandas              sysconfig
_pydecimal          dummy_threading     parser              tabnanny
_pyio               easy_install        parso               tarfile
_queue              email               pathlib             telnetlib
_random             encodings           patsy               tempfile
_sha1               enum                pdb                 textwrap
_sha256             errno               pickle              this
_sha3               faulthandler        pickletools         threading
_sha512             filecmp             pipes               time
_signal             fileinput           pkg_resources       timeit
_sitebuiltins       fnmatch             pkgutil             token
_socket             folium              platform            tokenize
_sqlite3            formatter           plistlib            toolz
_sre                fractions           pluggy              trace
_stat               freesasa            poplib              traceback
_string             ftplib              posix               tracemalloc
_strptime           functools           posixpath           traits
_struct             future              pprint              tty
_symtable           gc                  profile             turtle
_sysconfigdata__emscripten_ genericpath         proj4py             tutor
_testcapi           getopt              pstats              types
_thread             getpass             pty                 typing
_threading_local    gettext             pwd                 uncertainties
_tracemalloc        glob                py                  unicodedata
_warnings           graphviz            py_compile          unittest
_weakref            gzip                pyclbr              urllib
_weakrefset         hashlib             pydoc               uu
abc                 heapq               pydoc_data          uuid
aifc                hmac                pyexpat             warnings
antigravity         html                pygments            wave
argparse            html5lib            pyodide             weakref
array               http                pyodide_interrupts  webbrowser
asciitree           imageio             pyparsing           webencodings
ast                 imaplib             pyroutelib3         wsgiref
astropy             imghdr              pysat               xdrlib
asynchat            imp                 pystone             xlrd
asyncio             importlib           pytest              xml
asyncore            inspect             pytz                xmlrpc
atexit              io                  pywt                xxsubtype
atomicwrites        ipaddress           qrcode              yt
attr                ipythonblocks       queue               zarr
autograd            itertools           quopri              zipapp
base64              jedi                random              zipfile
```

```
basthon             jinja2              rcviz               zipimport
bdb                 joblib              re                  zlib
binarytree          js                  regex
binascii            json                reprlib

Enter any module name to get more help.  Or, type "modules spam" to search
for modules whose name or summary contain the string "spam".
```