



Bienvenue sur Python, MicroPython et CircuitPython

Rédacteur(s) : Philippe Mariano

[Mise à jour le 16/8/2022]

Ce Wiki, consacré aux bases de Python et aux cartes à microcontrôleur programmables en "MicroPython", est destiné à des élèves de lycée inscrits dans la spécialité NSI.

✓ : mise à jour en 2022


Présentation

- **Python** est un langage de programmation interprété, multiparadigme et multiplateformes. Il favorise la programmation impérative structurée, fonctionnelle et orientée objet. Python est distribué sous licence libre et présente une syntaxe épurée et simplifiée, ce qui en fait un outil adapté à l'apprentissage de la programmation.
Ce wiki traite uniquement la version 3.
- **MicroPython** est une implémentation simple et efficace du langage de programmation Python 3, qui inclut un petit sous-ensemble de la bibliothèque standard Python et qui est optimisée pour fonctionner sur des microcontrôleurs. Il est suffisamment compact pour s'adapter à 256 ko d'espace de code et à 16 ko de RAM.
- **CircuitPython** est un dérivé open source du langage de programmation MicroPython destiné aux étudiants et aux débutants. Le développement de CircuitPython est soutenu par Adafruit Industries. Il s'agit d'une implémentation logicielle du langage de programmation Python 3, écrit en C.
- **MicroPython vs CircuitPython** : contrairement à MicroPython, CircuitPython ne permet pas de faire du multithreading.

Python News: What's New From March 2022 ?

Sommaire

1.  **MicroPython, CircuitPython**
 1. [Embedded Python: Build a Game on the BBC micro:bit](#)
 2. **BrainPad - Python lessons** ✓
 3. [Wio - Premiers pas avec ArduPy](#)
2.  **Python**
 1. **Installation - Démarrage**

1. Particularités du langage
2. Premiers pas avec l'interpréteur de commandes
3. L'éditeur IDLE
4.  **VSCode - Premier programme en Python "étape par étape"**
5. RealPython
 1. [Your Python Coding Environment on Windows: Setup Guide](#)
 2. [Python Keywords: An Introduction](#)

1. Bases du langage

1. [Variables, constantes, types numériques et E/S dans la console](#) ✓
 1. Types numériques
 2. Type d'une variable, copie, permutation, opérations
 3. Portée
 4. Entrées / sorties dans la console
 5. Constantes
2. [Les instructions de contrôle](#) ✓
 1. alternatives
 2. répétitives
3. [Les fonctions](#) ✓
 1. Création, appel, passage de paramètres, signature
 2. Fonctions lambda
 3. [Fonctions natives](#) (built-in)
 1. [Fonctions intégrées Python à connaître](#) ✓
 2. [Python's sum\(\): The Pythonic Way to Sum Values](#)
 3. [Using the len\(\) Function in Python](#)
 4. [Python's min\(\) and max\(\): Find Smallest and Largest Values](#)
 5. [The Python range\(\) Function \(Guide\)](#)
4. Les structures de données
 1. [Common Python Data Structures \(Guide\)](#)
 2. Les séquences
 1. [Généralités](#)
 2. [Les chaînes de caractères](#)
 3. [Listes](#)
 4. [Tuples](#) ✓
 3. [Les dictionnaires](#)
 4. [Les ensembles \(set\)](#)
5. [Les modules et packages](#)
 1. Les modules : utilisation et création
 2. Turtle, Numpy, Matplotlib, etc.
 3. Les packages
6. [Les fichiers](#)
7. [Les exceptions, assertions et le module doctest](#)
8. Documentation
 1. [Documenting Python Code: A Complete Guide](#)
9. Script
 1. [How to Run Your Python Scripts. \[Quiz\]](#)
10. PEP 8
 1. [Bonnes pratiques de codage](#)
 2. [How to Write Beautiful Python Code With PEP 8](#)
 3. [3 Tips For Writing Pythonic Code](#)
11. [Quiz](#)

2. **Programmation intermédiaire**

1. [Intermediate Python Tutorials](#)
2. [Namespaces and Scope in Python](#)
3. [Python Inner Functions: What Are They Good For?](#)
4. [Dates And Times And Types](#)

3. **Programmation avancée**

1. Algorithmes, maths et science des données
 1. [Sorting Algorithms in Python](#)
 2. [Recursion in Python: An Introduction](#)
 3. [The k-Nearest Neighbors \(kNN\) Algorithm in Python](#)
 4. [Simplify Complex Numbers With Python](#)
 5. [Math for Data Science](#)
 6. [Python Data Science Tutorials](#)
 7. [Logistic Regression in Python](#)
 8. [NumPy, SciPy, and Pandas: Correlation With Python](#)
 9. [Fourier Transforms With scipy.fft: Python Signal Processing](#)
 10. [Draw the Mandelbrot Set in Python](#)
 11. [How to Do a Binary Search in Python](#)
 12. [Data Visualization With Python](#)
 13. [Linear Regression in Python](#)
2. API
 1. [How to Do a Binary Search in Python](#)
3. Asynchrone
 1. [Getting Started With Async Features in Python](#)
4. Bases de données et SGBD
 1. [Généralités](#)
 2. [Python Database Tutorials](#)
 3. [Des bases de données en Python avec sqlite3](#)
 4. [Build a Contact Book With Python, PyQt, and SQLite](#)
 5. [Introduction to Python SQL Libraries](#)
5. Bonnes pratiques
 1. [Technical Writing for Developers](#)
 2. [Best Practices for More Pythonic Code](#)
 3. [Python vs C++: Selecting the Right Tool for the Job](#)
 4. [Write More Pythonic Code](#)
 5. [LBYL vs EAFP: Preventing or Handling Errors in Python](#)
6. CLI Python
 1. [Python Command Line Arguments](#)
7. CPython
 1. [Your Guide to the CPython Source Code](#)
8. Documentation
 1. [Build Your Python Project Documentation With MkDocs](#)
9. Excel
 1. [A Guide to Excel Spreadsheets in Python With openpyxl](#)
10. Embarqué
 1. [PikaScript](#)
11. Exceptions
 1. [Python Exceptions: An Introduction](#)
 2. [Python KeyError Exceptions and How to Handle Them](#)
 3. [LBYL vs EAFP: Preventing or Handling Errors in Python](#)
12. Fonctionnel

1. [Python's filter\(\): Extract Values From Iterables](#)
13. Générateurs
 1. [How to Use Generators and yield in Python](#)
14. Interfaces graphiques
 1. [Python GUI Programming](#)
 2. [GUI Programming With PyQt](#)
15. Journalisation
 1. [Logging in Python](#)
 2. [The Ins and Outs of Logging in Python, Part 1](#)
16. Jupyter
 1. [How to create a dashboard in Python with Jupyter Notebook?](#)
17. Langages
 1. [Natural Language Processing With Python's NLTK Package](#)
 2. [Java vs Python: Basic Python for Java Developers](#)
18. Linter
 1. [Learn Python ASTs, by building your own linter](#)
19. Machine learning
 1. [Machine Learning With Python](#)
 2. [Python AI: How to Build a Neural Network & Make Predictions](#)
20. Mémoire
 1. [Memory Management in Python](#)
21. Mobile
 1. [Build a Mobile Application With the Kivy Python Framework](#)
22. Modules et packages
 1. [Python Zip Imports: Distribute Modules and Packages Quickly](#)
 2. [Dependency Management With Python Poetry](#)
 3. [How to Publish an Open-Source Python Package to PyPI](#)
 4. [How to create a Python package in 2022](#)
23. Multitâche
 1. [Python Concurrency & Parallel Programming](#)
 2. [Threading in Python: The Complete Guide](#)
 3. [Speed Up Your Python Program With Concurrency](#)
 4. [The subprocess Module: Wrapping Programs With Python](#)
 5. [map\(\) vs submit\(\) With The ProcessPoolExecutor in Python](#)
 6. [Python Threading Quiz](#)
24. pdf
 1. [pikepdf : a Python library for reading and writing PDF files](#)
25. **POO - Programmation Orientée Objet✓**
 1. Classe et objet
 2. Héritage
 3. Polymorphisme
26. Structures de données
 1. Tuple
 1. [Write Pythonic and Clean Code With namedtuple](#)
 2. File et pile
 1. [Python Stacks, Queues, and Priority Queues in Practice](#)
 2. [Python's deque: Implement Efficient Queues and Stacks](#)
 3. [How to Implement a Python Stack](#)
 4. [Reverse Python Lists: Beyond .reverse\(\) and reversed\(\)](#)
 3. [Build a Hash Table in Python With TDD](#)
27. Tests

1. [Test Your Python Apps](#)
2. [Effective Python Testing With Pytest](#)
3. [Documentation unit tests](#)

28. Web

1. [Model-View-Controller \(MVC\) Explained – With Legos](#)
2. [Python's Requests Library \(Guide\)\[Quiz\]](#)
3. [Python's urllib.request for HTTP Requests](#)
4. [Exploring HTTPS With Python](#)
5. [Bottle](#)
 1. [Un serveur web en Python avec **Bottle**.](#)
6. [Django](#)
 1. [Django Tutorials](#)
 2. [Django vs. Flask in 2021: Which Framework to Choose](#)
7. [Docker](#)
 1. [Python Docker Tutorials](#)
8. [Flask](#)
 1. [Flask Tutorials](#)
 2. [Accepting Payments with Stripe, Vue.js, and Flask](#)
 3. [Primer on Jinja Templating](#)
9. [Sending Emails With Python](#)
10. [Front-end Web Development Tutorials](#)
 1. [Brython: Python in Your Browser](#)
 2. [Python vs JavaScript for Pythonistas, etc](#)
11. [MQTT](#)
 1. [Fundamentals of MQTT](#)
 2. [Beginner's Guide To Using Paho-MQTT](#)
12. [PyScript](#)
 1. [A First Look at PyScript: Python in the Web Browser](#)
 2. [PyScript — unleash the power of Python in your browser](#)
13. [REST](#)
 1. [Python and REST APIs: Interacting With Web Services](#)
14. [Sécurité](#)
 1. [Secure Password Handling in Python](#)
15. [Serveur](#)
 1. [Socket Programming in Python \(Guide\)](#)
16. [Twitter](#)
 1. [How to Make a Twitter Bot in Python With Tweepy](#)
17. [**Web Scraping Tutorials**](#)

29. XML

1. [A Roadmap to XML Parsers in Python](#)

4. Jeux

1. [Top Python Game Engines](#)
2. [Set Up the Diagram of Dice Faces](#)
3. [Make Your First Python Game: Rock, Paper, Scissors!](#)
4. [PyGame: A Primer on Game Programming in Python](#)
5. [Arcade: A Primer on the Python Game Framework](#)
6. [Build an Asteroids Game With Python and Pygame](#)
7. [Build a Platform Game in Python With Arcade](#)

5. Projets

1. [70+ Python Projects for Beginners, Intermediate and Experienced Developers](#)
2. [Il pleut dehors? Créer une application CLI météo avec Python](#)

3. [Advent of Code: Solving Your Puzzles With Python](#)
4. [Build a Command-Line To-Do App With Python and Typer](#)
5. [Draw the Mandelbrot Set in Python](#)
6. [Build a Bulk File Rename Tool With Python and PyQt](#)
7. [Build a Site Connectivity Checker in Python](#)
8. [Build a URL Shortener With FastAPI and Python](#)
9. [Build a Quiz Application With Python](#)
6. **Outils**
7. **Création**
 1. [Thonny: The Beginner-Friendly Python Editor](#)
 2. **VSCode**
 1. [L'environnement de développement intégré \(IDE\) VSCode](#)
 2. [Python interactif \(IPython\) et Jupyter dans VSCode](#)
 3. [Installer et utiliser Anaconda, Anaconda avec VSCode](#)
 3. [Notebook Jupyter, binder et Google Colab](#)
 4. [Installer et utiliser PIP](#)
 1. [Using Python's pip to Manage Your Projects' Dependencies](#)
8. **Docker**
 1. [How to Deploy a Jupyter Notebook to Docker](#)
9. **Documentation**
 1. [Mémento de Markdown](#)
10. **Entraînement en ligne**
 1. [INGInious](#)
 2. [France lol](#)
11. **Environnement d'exécution**
 1. [Environnements virtuels Python : introduction](#)
 2. [pipx — Install and Run Python Applications in Isolated Environments](#)
 3. [Managing Multiple Python Versions With pyenv](#)
12. **Sauvegarde et collaboration**
 1. [Transférer des fichiers avec FileZilla client](#)
 2. [Travail collaboratif dans VSCode](#)
 3. **Git et Github**
 1. [Gestion de versions : démarrer avec git et Github](#)
 2. [Introduction to Git and GitHub for Python Developers](#)
 3. [Advanced Git Tips for Python Developers](#)
13. **Simulation**
 1. **En ligne** : [Python Tutor](#)
14. **Concours**
15. [Castor](#) (du CM1 à la terminale)
16. [Algoréa](#) (collège, lycée)
17. [Alkindi](#) (cryptanalyse, secondes)
18. [Prologin](#)
19. [Les trophées NSI](#)
- **Ressources**
- [Memento Python 3.x](#)
- **Real Python**
 - [Your Guide to the CPython Source Code](#)
- [Bibliographie](#)
- [Webographie](#)
- [Glossaire](#)

From:

<http://webge.fr/dokuwiki/> - **WEBGE Wikis**

Permanent link:

<http://webge.fr/dokuwiki/doku.php?id=python:accueilpython&rev=1660633131>

Last update: **2022/08/16 08:58**

