

---

# **Conception et programmation orientée objet avec Python**

Georges Georgoulis – Alteractifs

Pour le Greta 92

Mars- Avril 2022

---

---

j1





# Un cours amical pour l'environnement

## Supports

- Support de cours en ligne
  - PPT
  - PDF
- exercices/corrigés/expériences
- Sources \*.py
- Classeurs Jupyter
- Documentation et références
  - URLs

## Prise de notes

- Invitation à utiliser Jupyter
- Espace de travail partagé

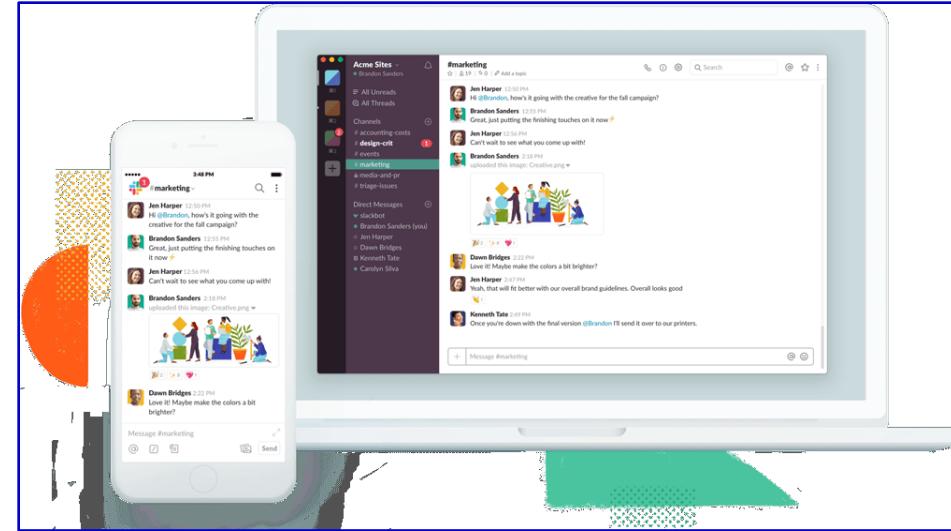


# Espace de travail partagé

<https://Python-202203.slack.com>

Nom de l'espace partagé :  
"Python-202203"

- Cours
- Supports
- Exercices
- Corrigés
- Questions & Discussions
- Progression

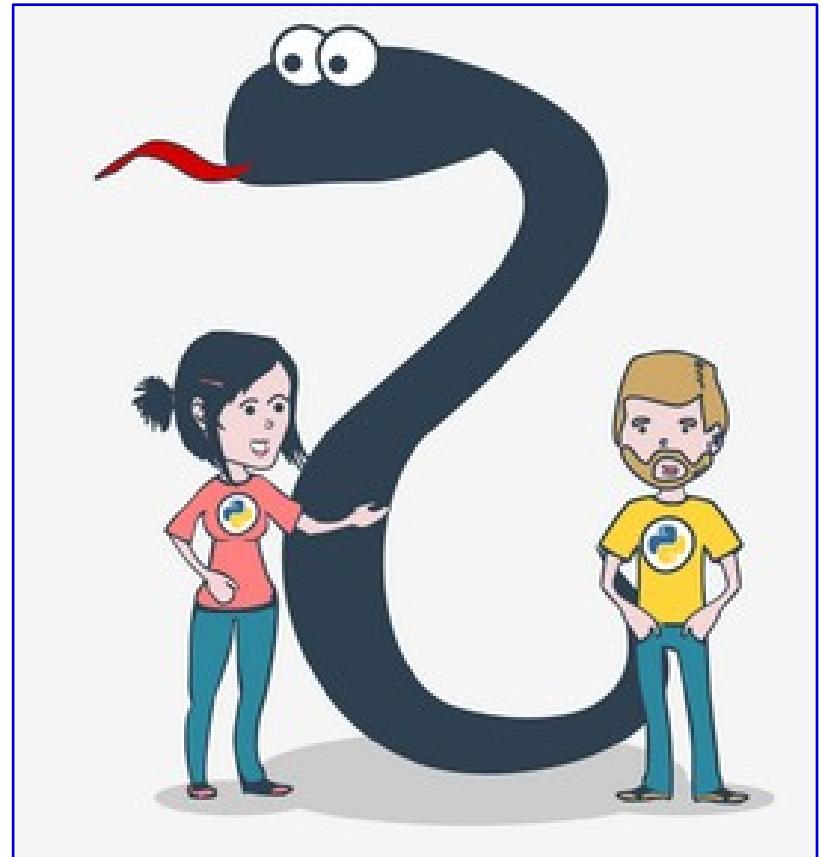


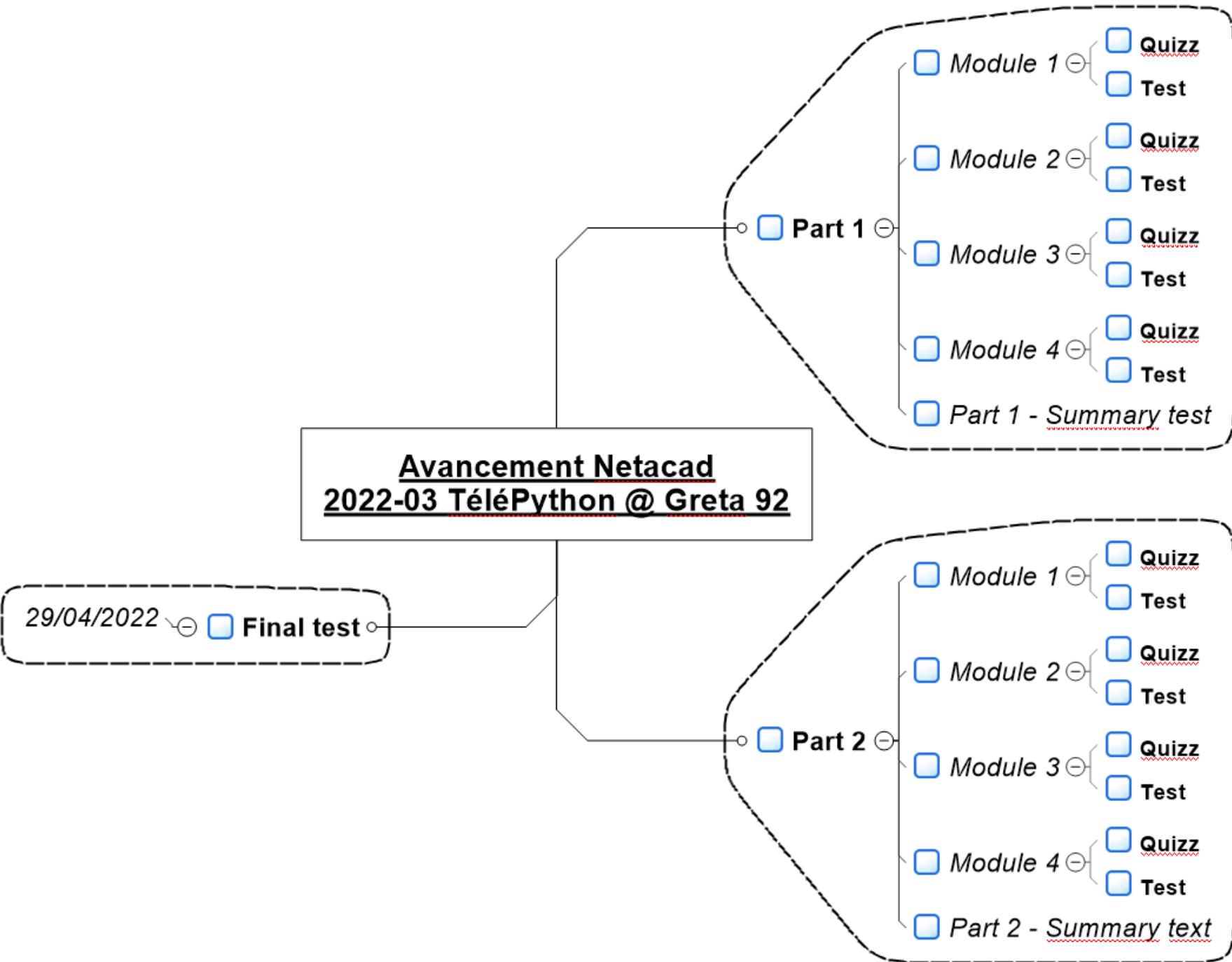
## Objectif

- 8 modules = 4+4
- 1 quizz et un test à la fin de chaque module (sauf le 1<sup>er</sup> qui n'a pas de test)
- 2 tests plus globaux au bout du 4<sup>ème</sup> et du 8<sup>ème</sup> module ("Summary test")
- 1 examen final

## Approche

- Ateliers
- Avant l'atelier, en autonomie :
  - lire le cours, faire les quizz et les tests
- Pendant l'atelier, ensemble :
  - Faire les quizz et tests, compléter le cours





# J1 : Plan

---

- Importance de Python sur le marché du développement logiciel
- Histoire de Python
- Mettre en place l'environnement
  - Un environnement de développement
  - Un environnement expérimental
  - Premiers pas et mise à niveau de chacun sur des notions de bases
    - Programmation
    - Algorithme
    - Langages
    - Interprétré/Compilé
    - Ordinateur
    - Source/Objet/Byte code
    - Langage machine
    - Read-Eval-Print
    - Editer/Exécuter
    - Editer/Compiler/ Exécuter

# J1 : Objectifs

---

- Apprendre à se connaître
- Le développement logiciel
- Python
- Mettre en place l'environnement Python

# Python

## Pour quoi faire ?

- Traiter des masses
  - de fichiers texte
  - d'images
- Génie logiciel
- Automatiser des taches
- Taches de tests ou de développement
- Base de données
- Développement rapide
- Jeux
- Interfaces graphiques
- Bureau « Gnome » d'Ubuntu

## Sur quelles plateformes ?

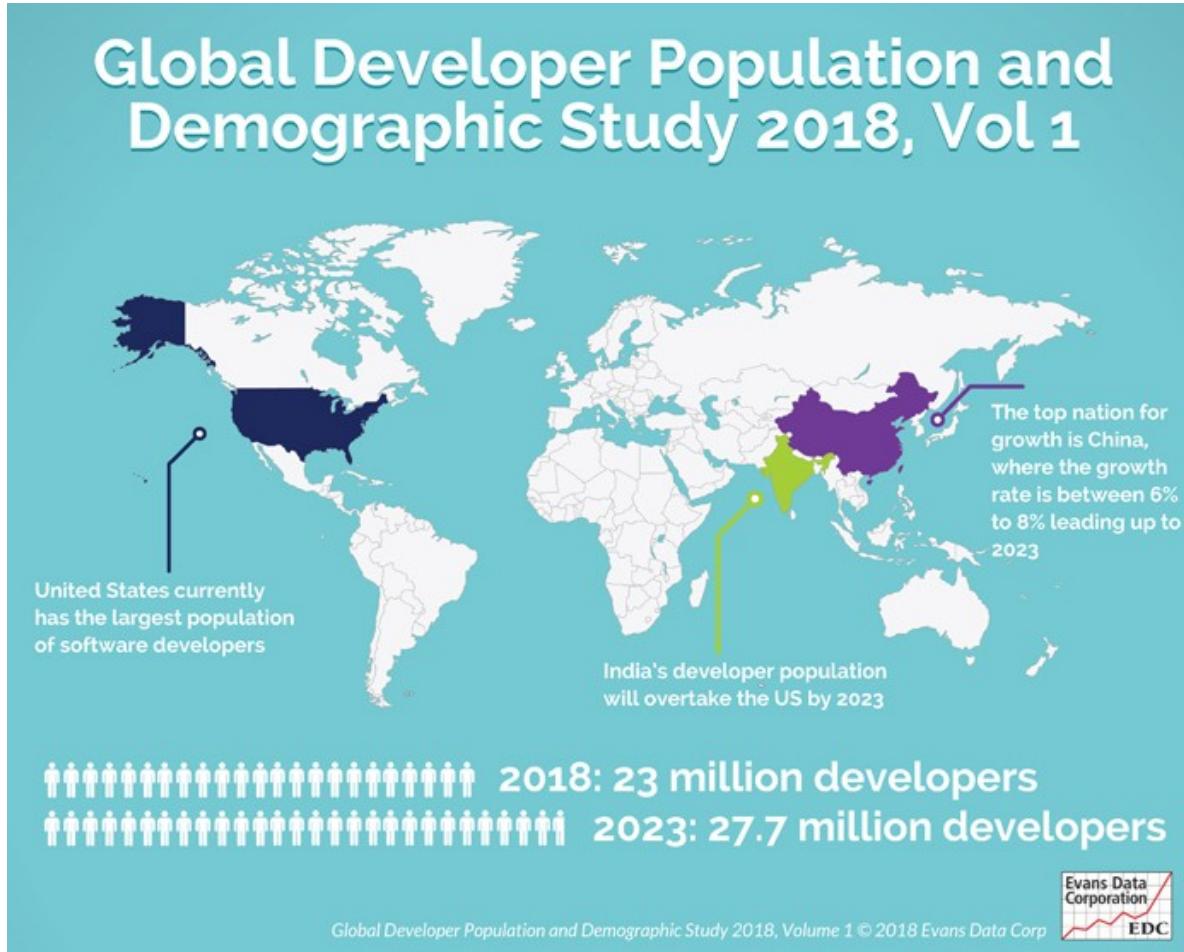
- Windows,
- Unix,
- Linux,
- MacOS
- IOT
  - Arduino, Raspberry
- Cloud
  - Amazon AWS Python Boto
  - Big data /Data Science

---

# **Développement logiciel PYTHON ET LE MARCHÉ**



# Les développeurs dans le monde



Source : [Evans Data](#) 2018



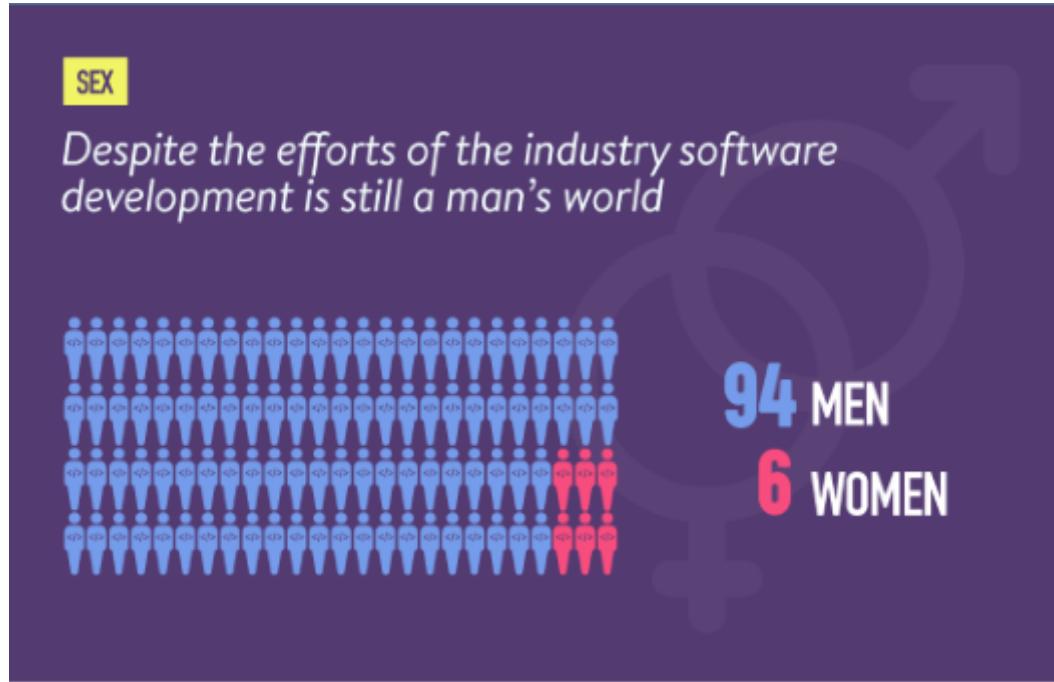
# Professionnalisation des développeurs



Source : [developpez.net](http://developpez.net) 2016



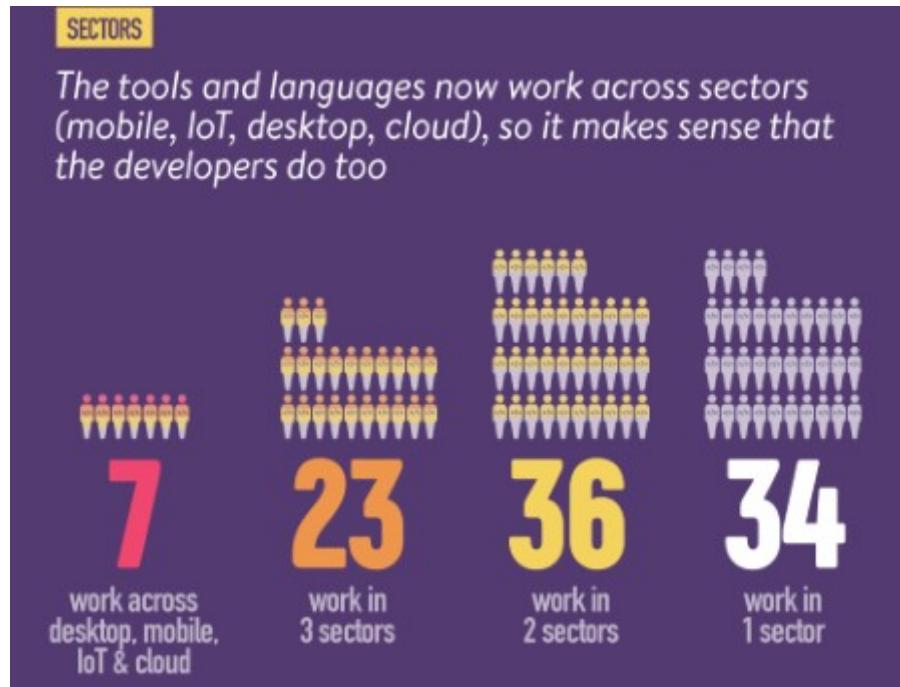
# Sexe des développeurs



Source : [developpez.net](http://developpez.net) 2016



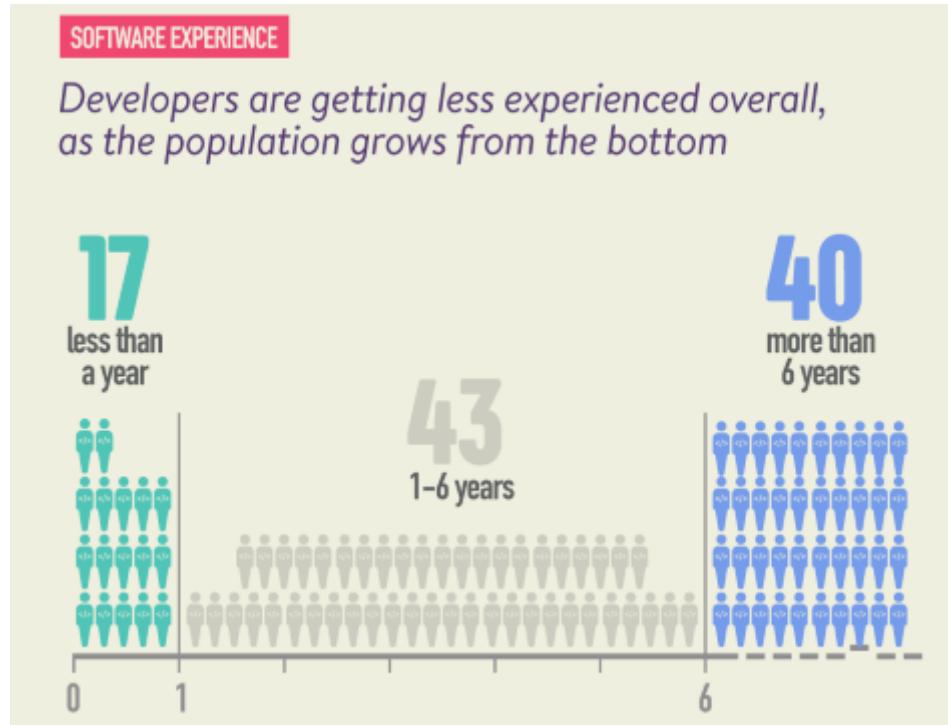
# Spécialisation des développeurs



Source : [developpez.net](http://developpez.net) 2016



# Expérience des développeurs



Source : [developpez.net](http://developpez.net) 2016



- Créé en 2008, [Stackoverflow](#) est une référence mondiale incontournable pour les développeurs.

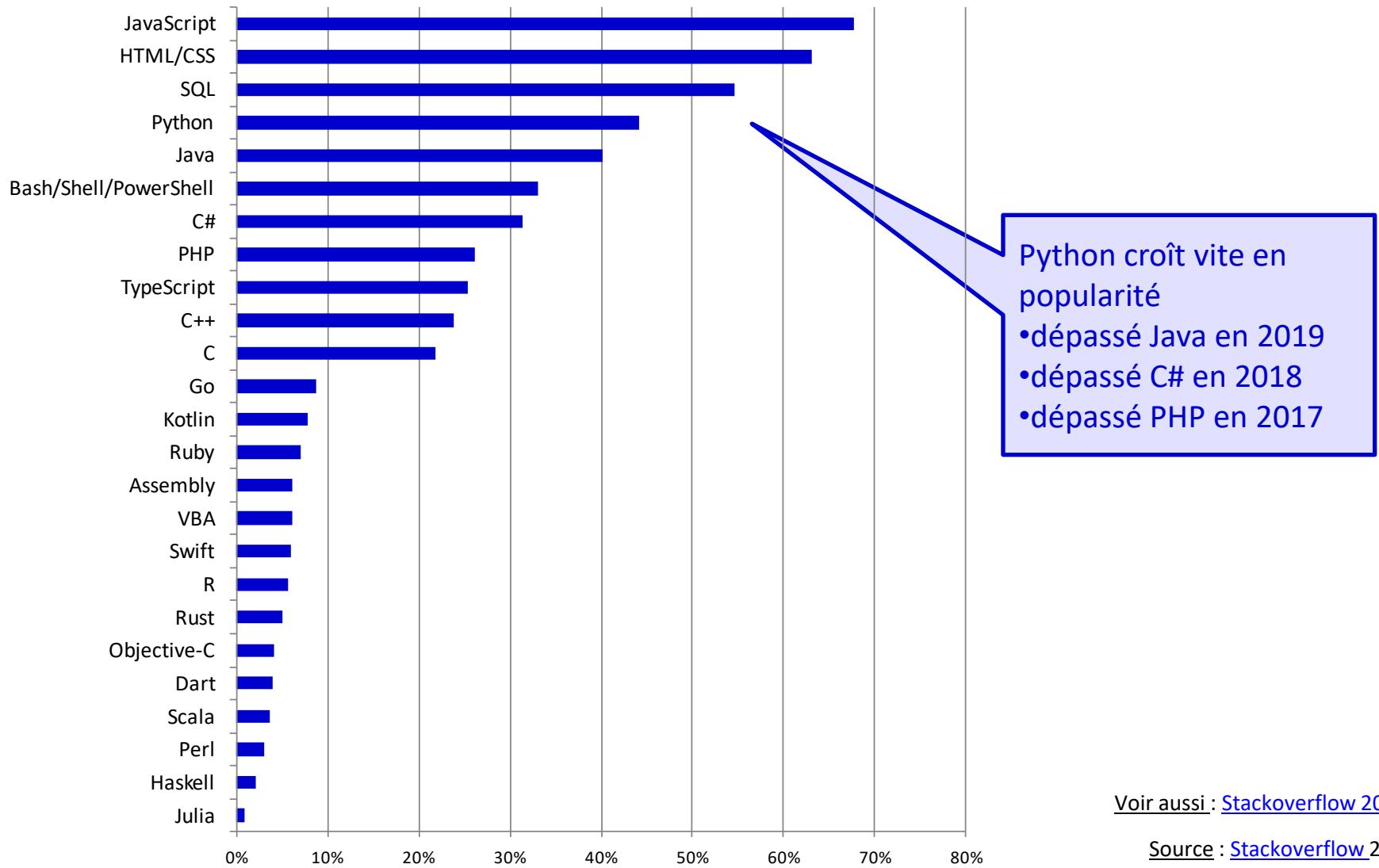
10 Millions de membres à fin 2018

16 Millions de questions traitées à mi 2018

- Beaucoup de développeurs disent apprendre en parcourant les forums
- Les questions les plus pointues y trouvent des réponses
- Stackoverflow dispose d'un corpus suffisamment significatif pour analyser les comportements des développeurs.
  - [Developer Survey Results 2020](#)

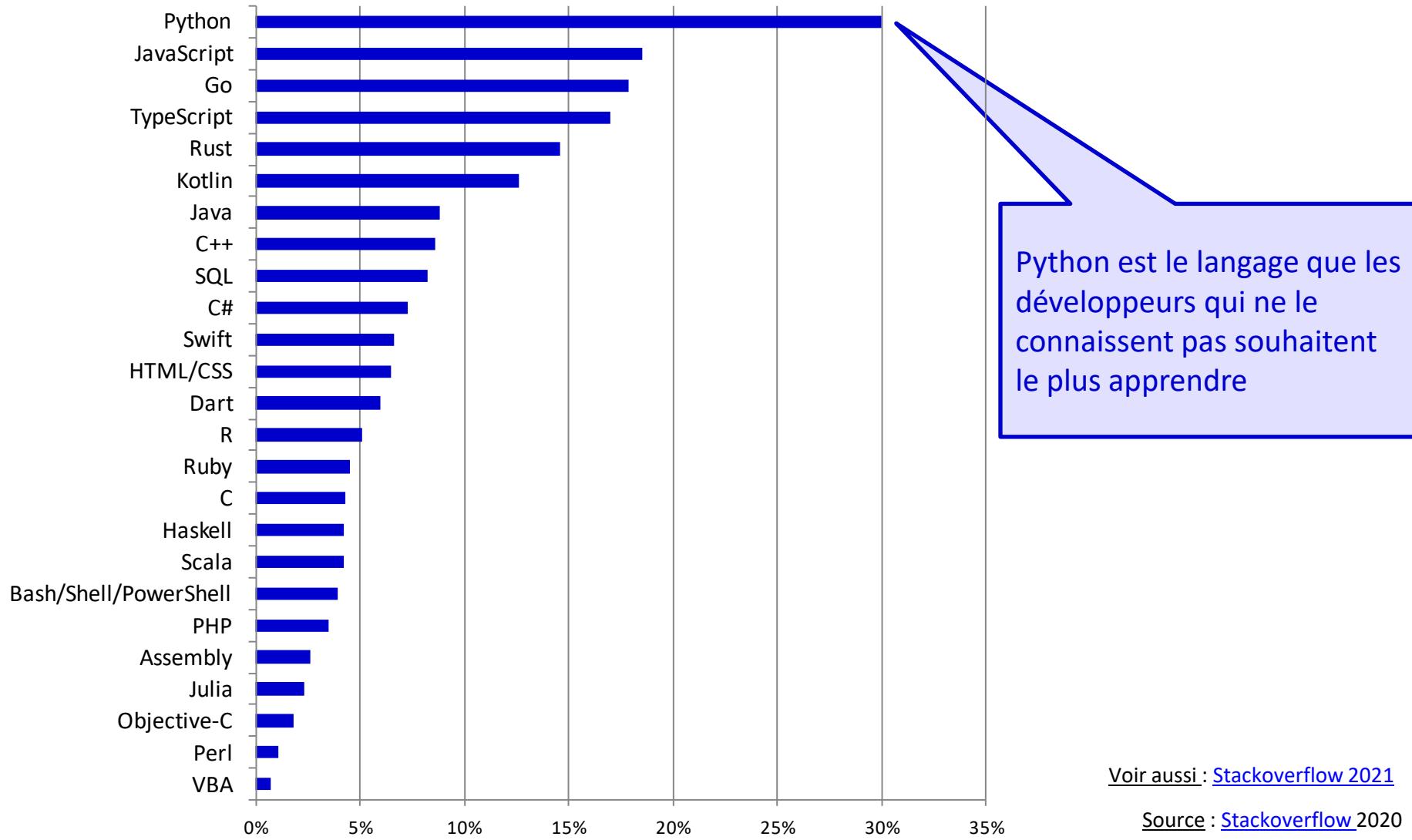


# Popularité des langages de programmation



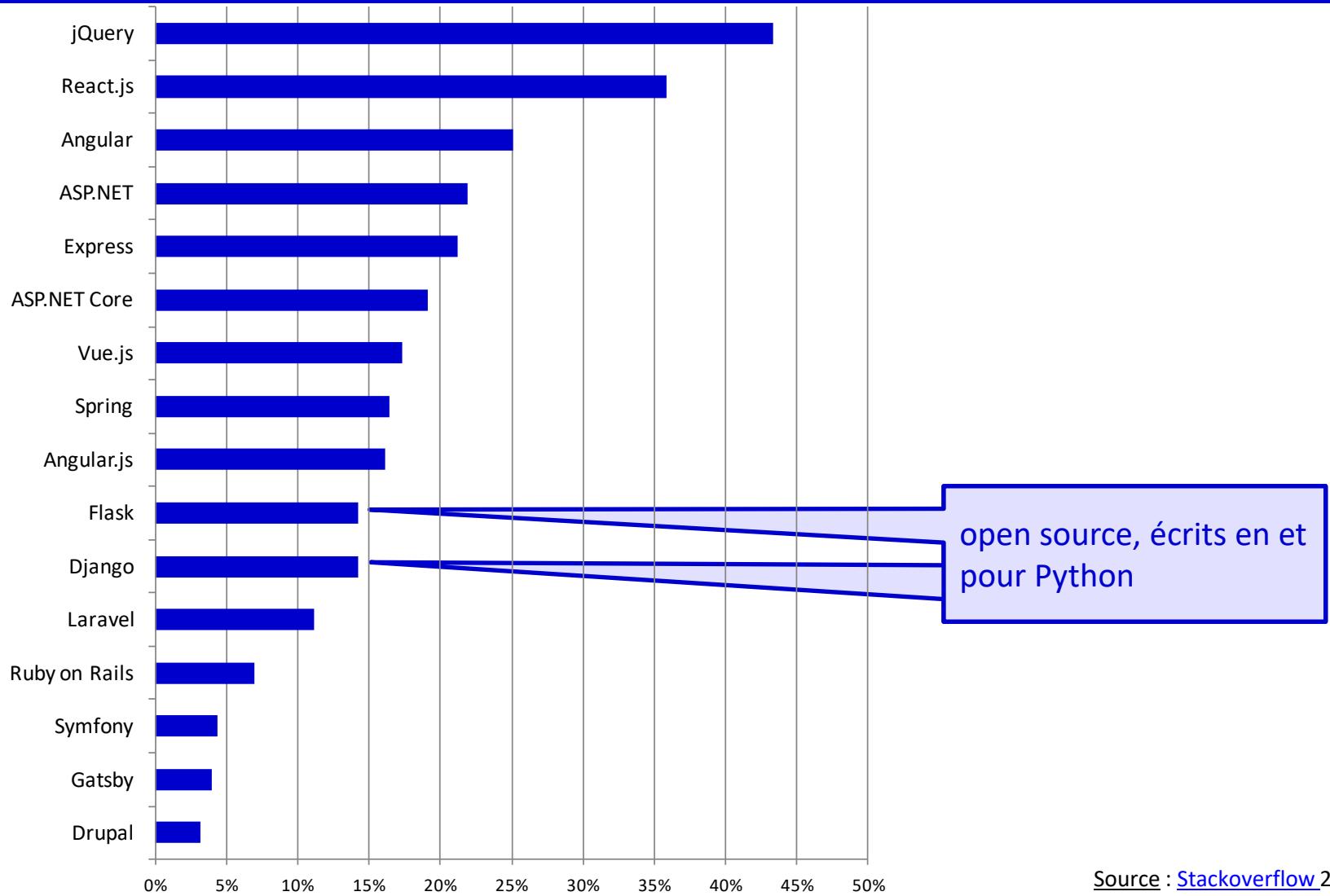


# Langages les plus désirés





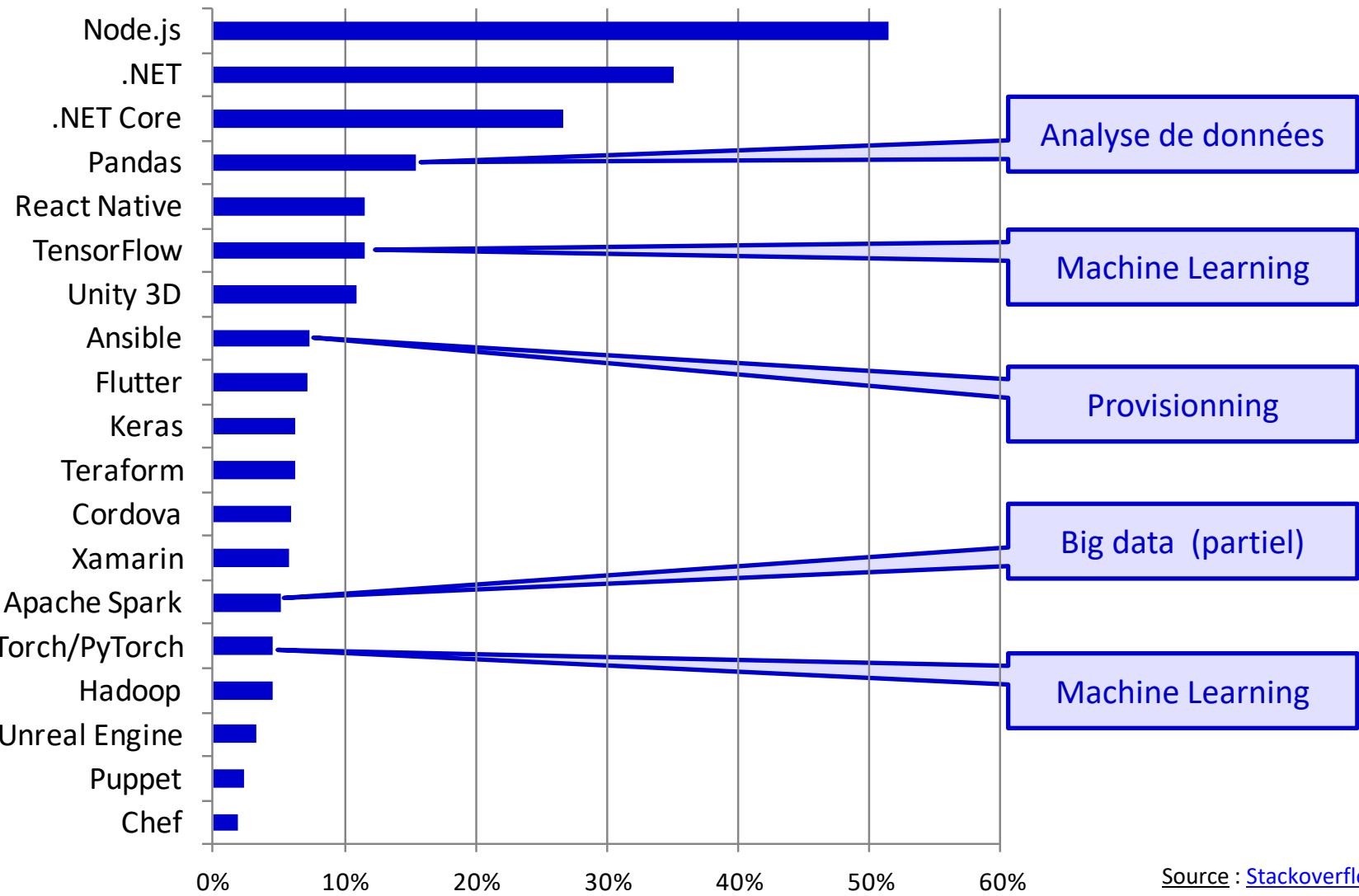
# Frameworks web



Source : [Stackoverflow](#) 2020



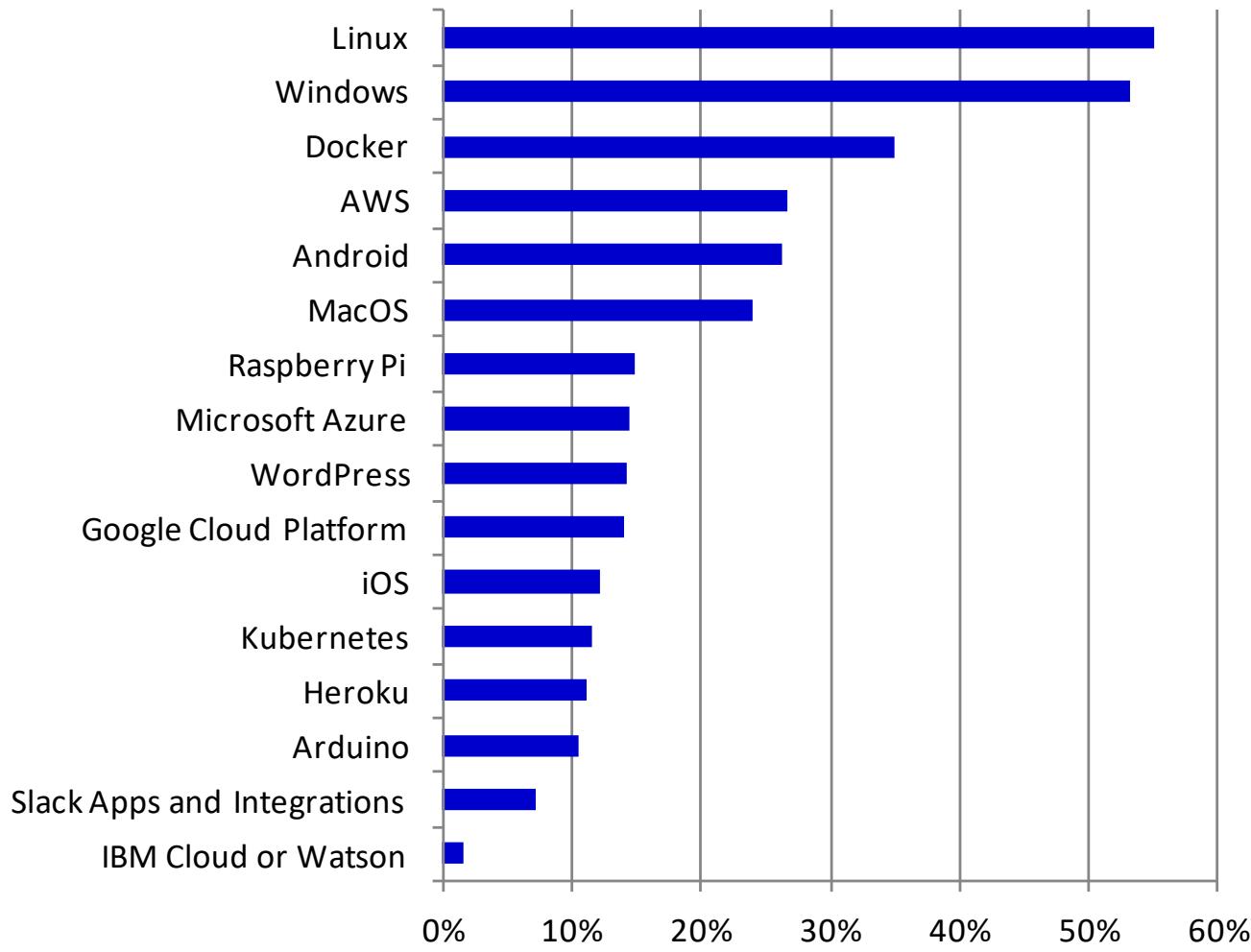
# Autres frameworks



Source : [Stackoverflow](#) 2020

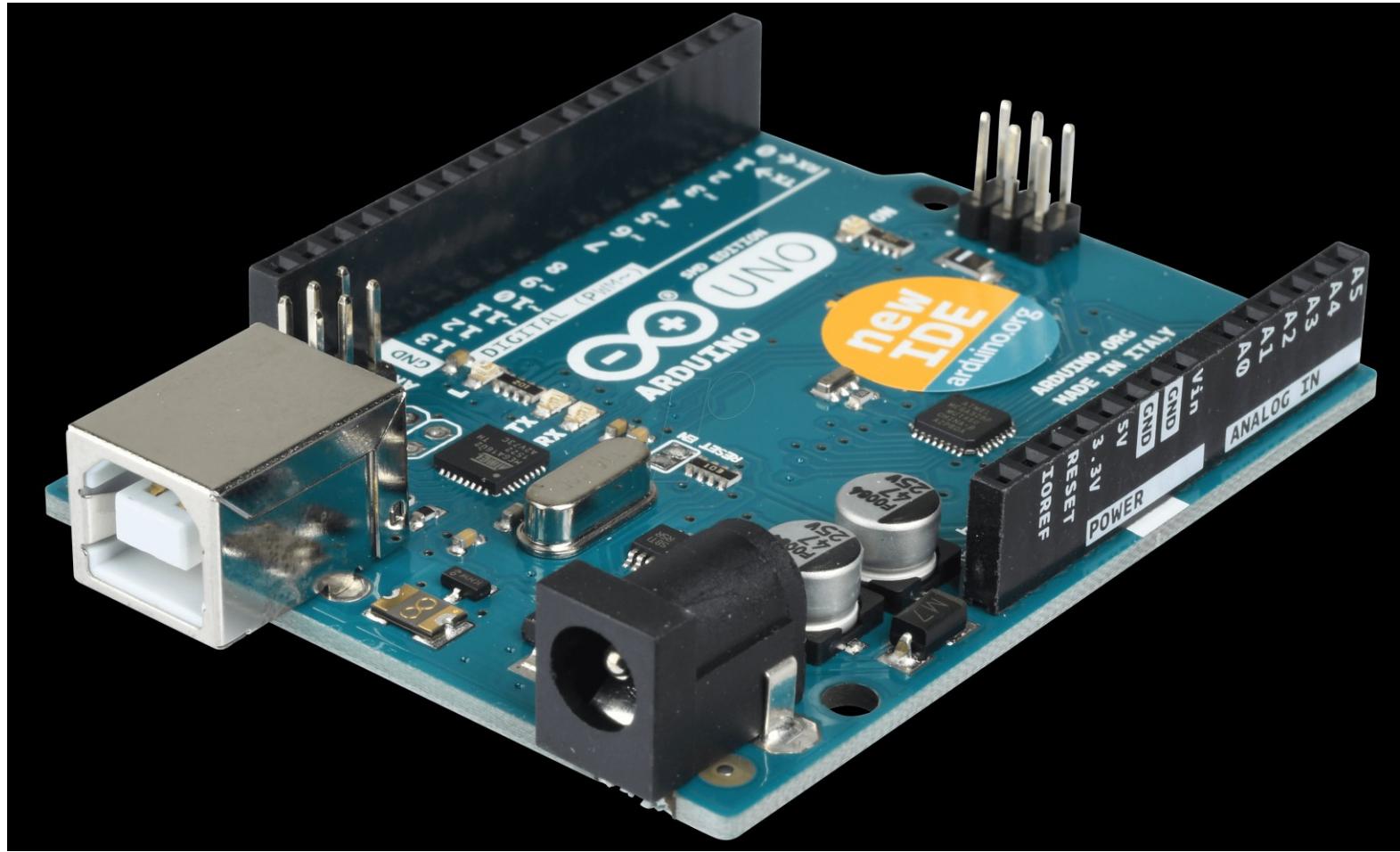


# Popularité des plateformes cibles



Source : [Stackoverflow](#) 2020

# Arduino Uno Rev 3

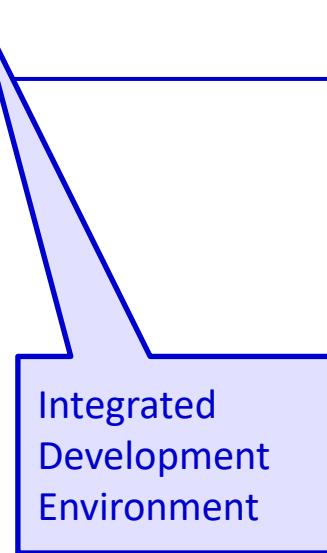
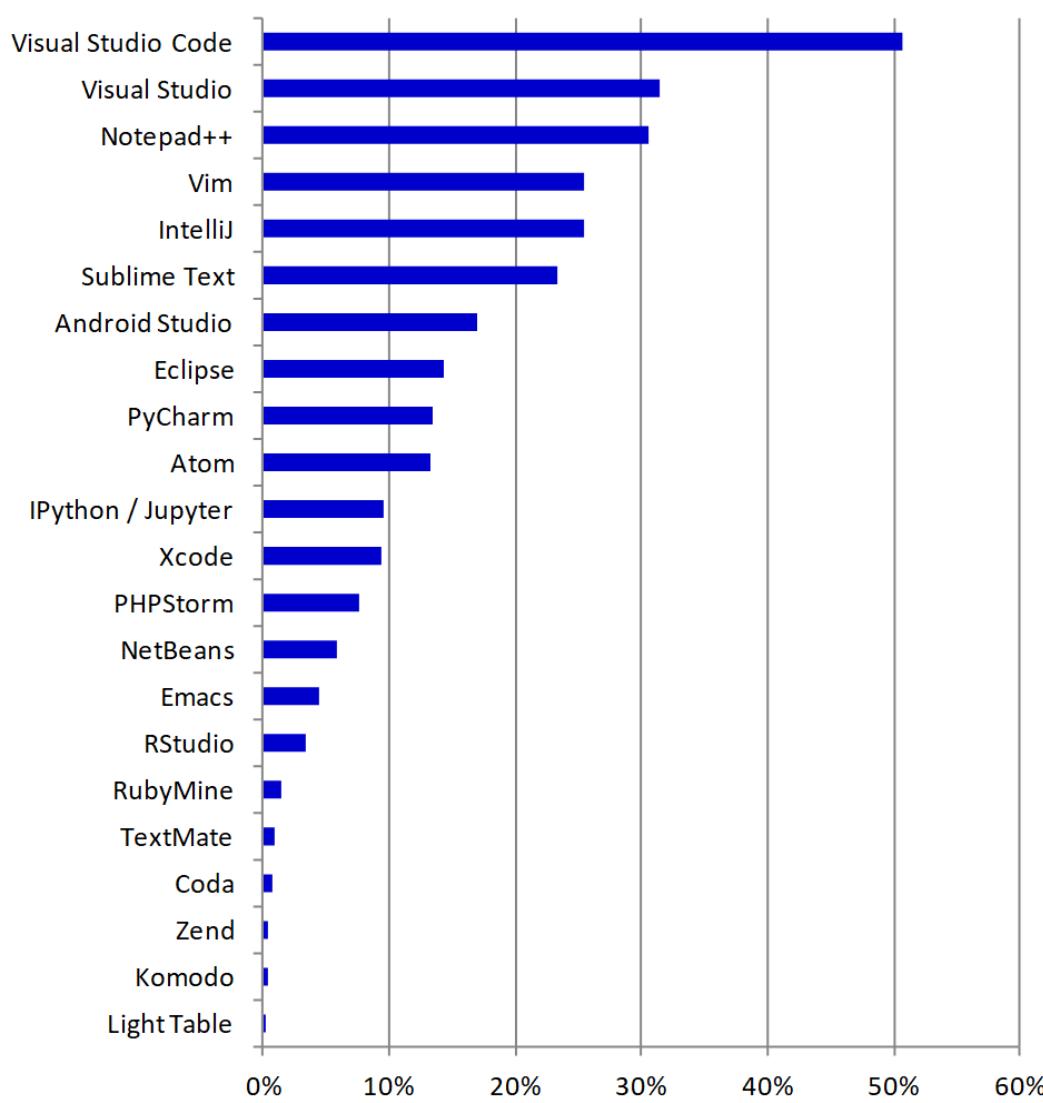


# Raspberry Pi 3 Model B





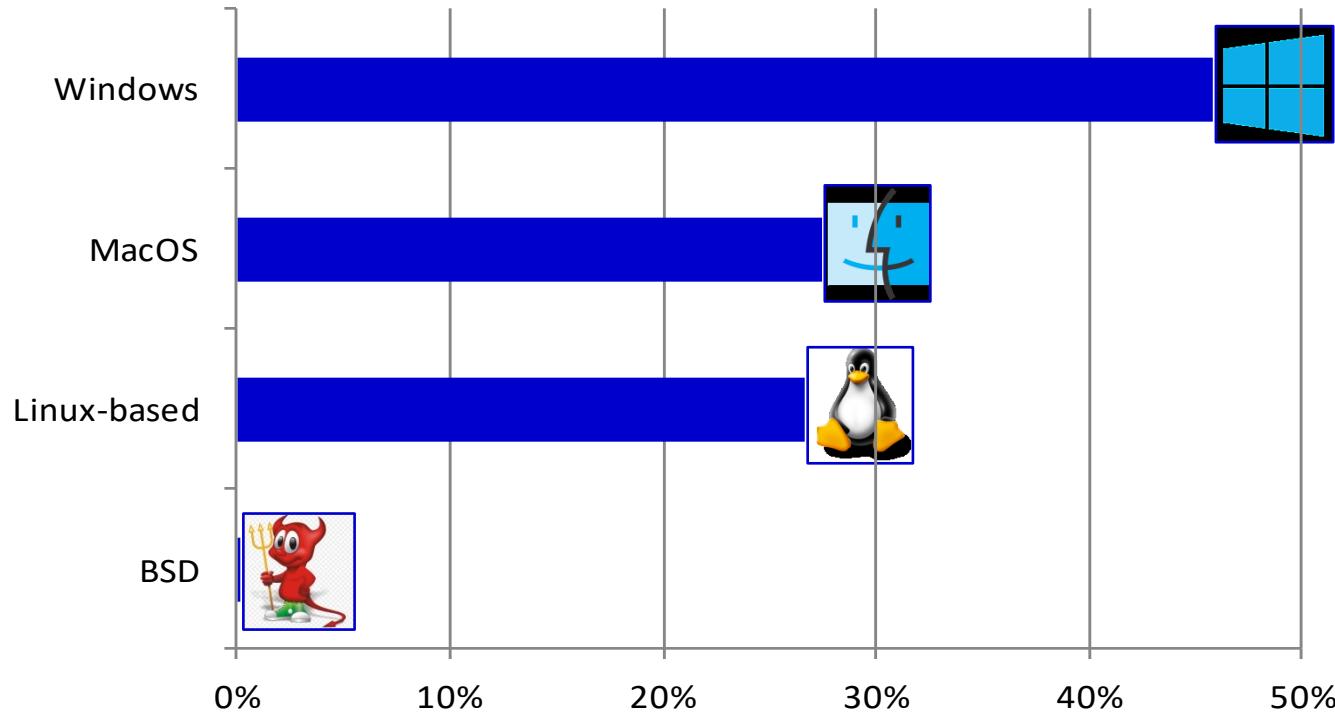
# Popularité des IDE



Source : [Stackoverflow 2019](#)



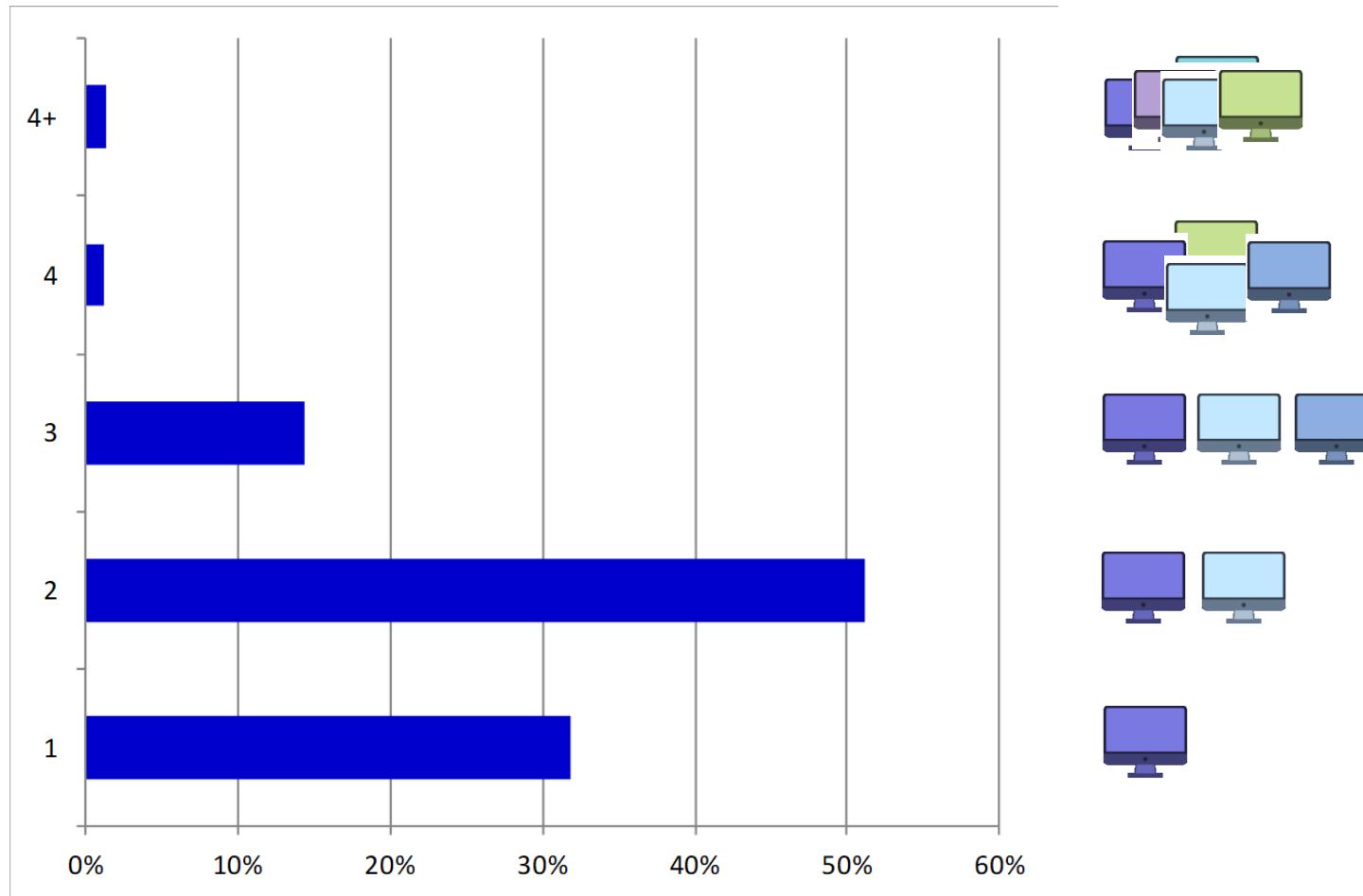
# Popularité des OS de développement



Source : [Stackoverflow](#) 2020



# Combien d'écrans dans une station de développement ?

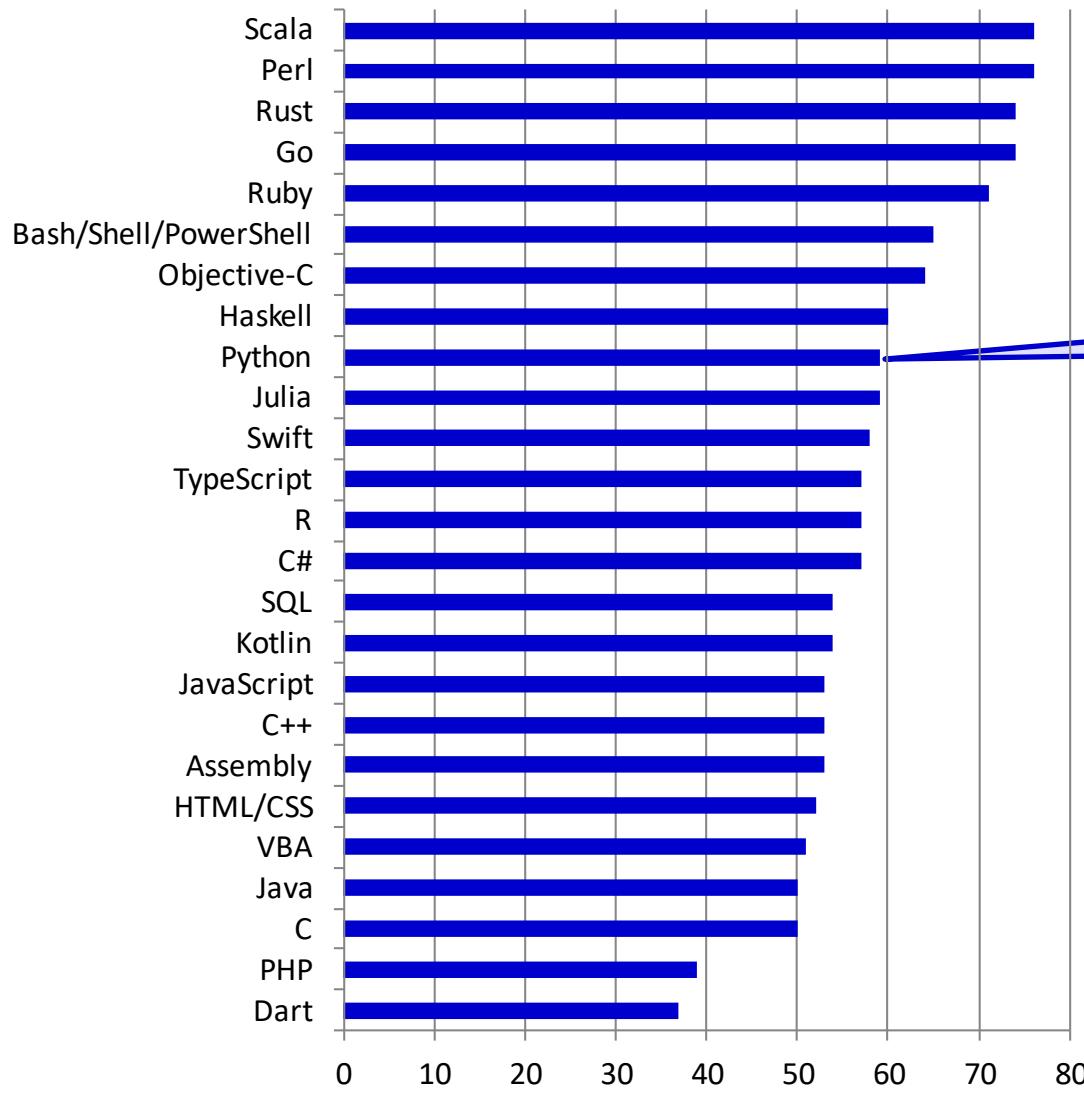


Source : [Stackoverflow 2018](#)





# Que gagne un développeur, selon le langage?



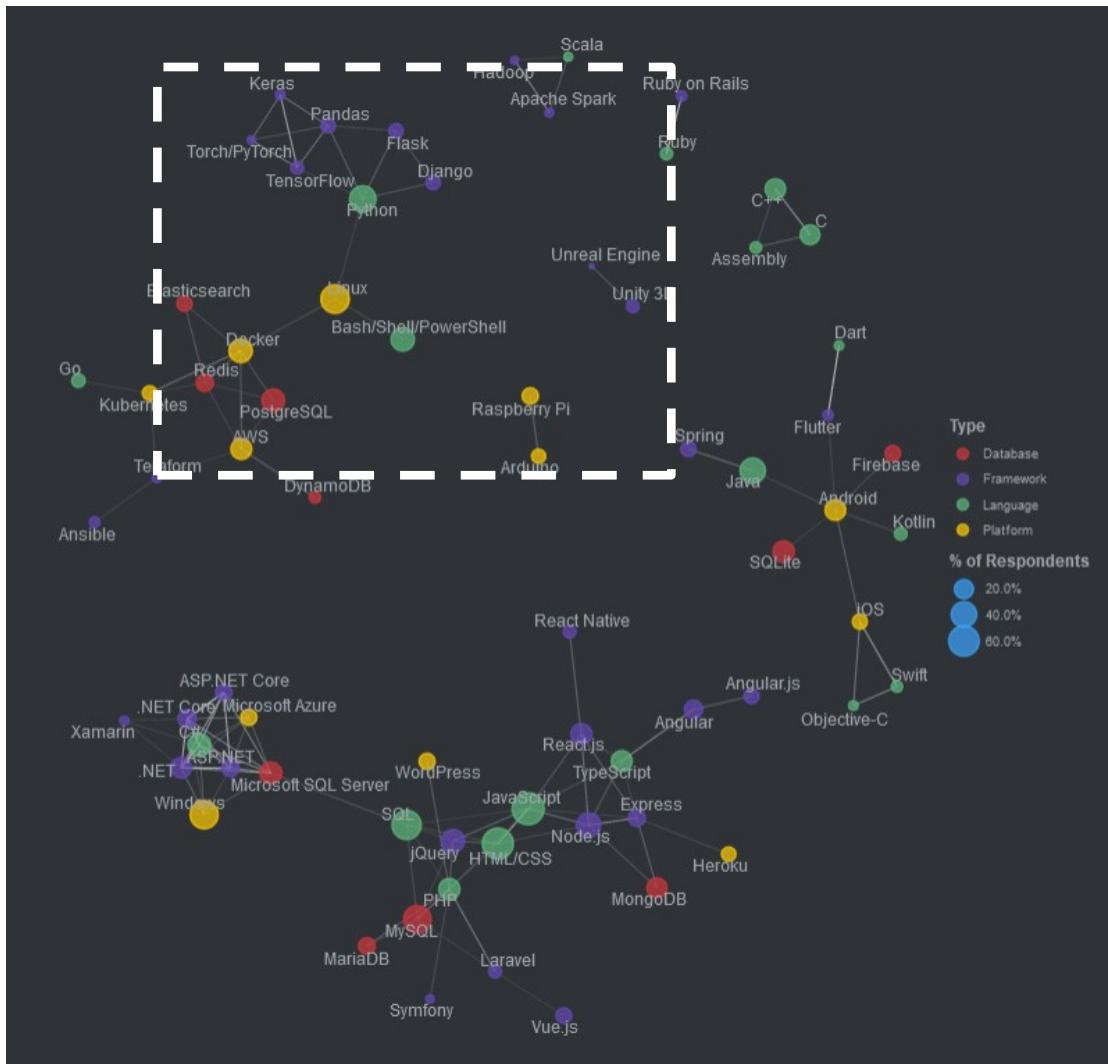
Python est là !

Ce sont des moyennes  
US en k\$ annuels

Voir aussi : [Stackoverflow 2021](#)

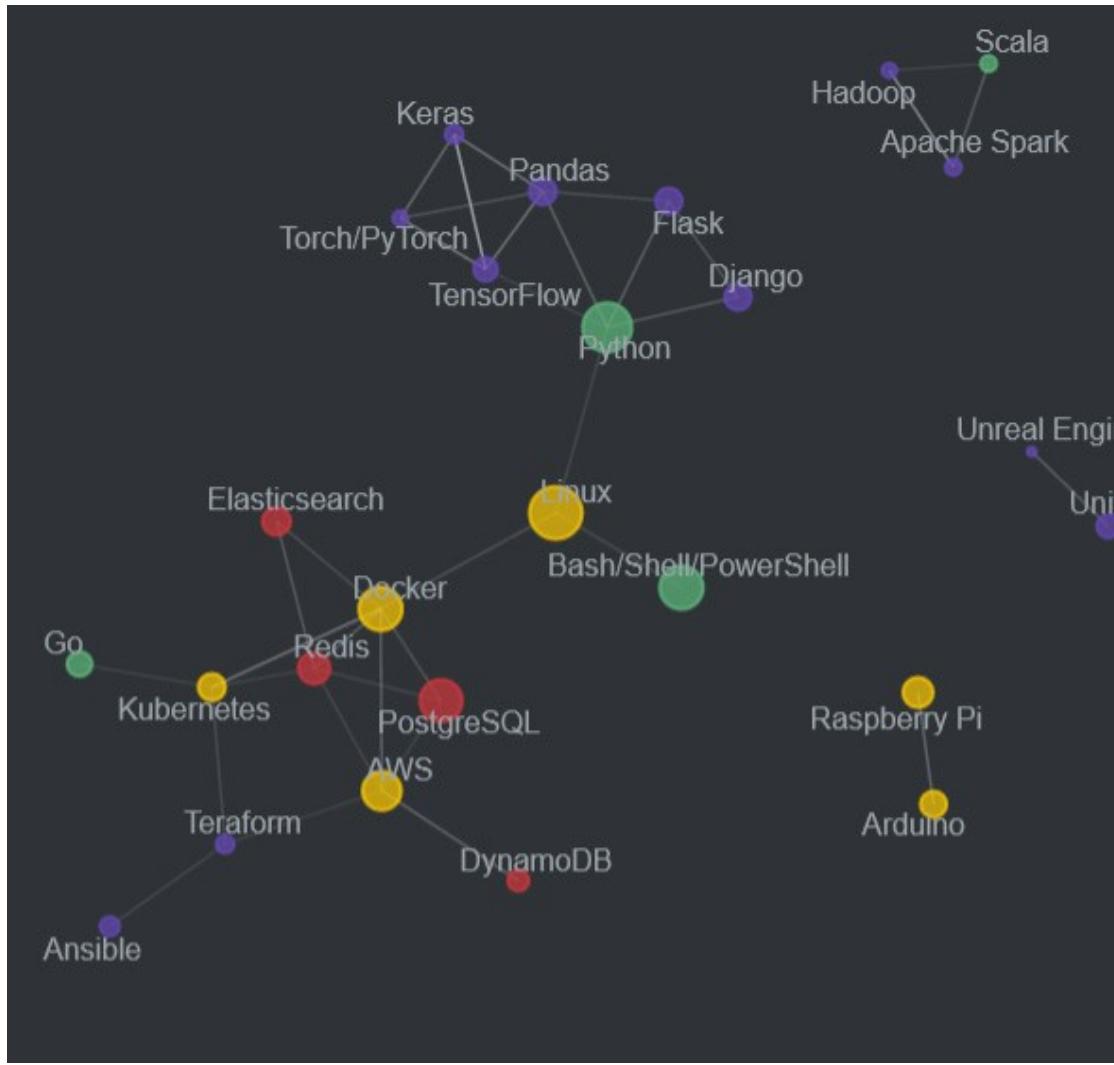
Source : [Stackoverflow](#) 2020

# Les technologies qui vont ensemble



Source : [Stackoverflow](#) 2020

# Les technologies qui vont ensemble (zoom)

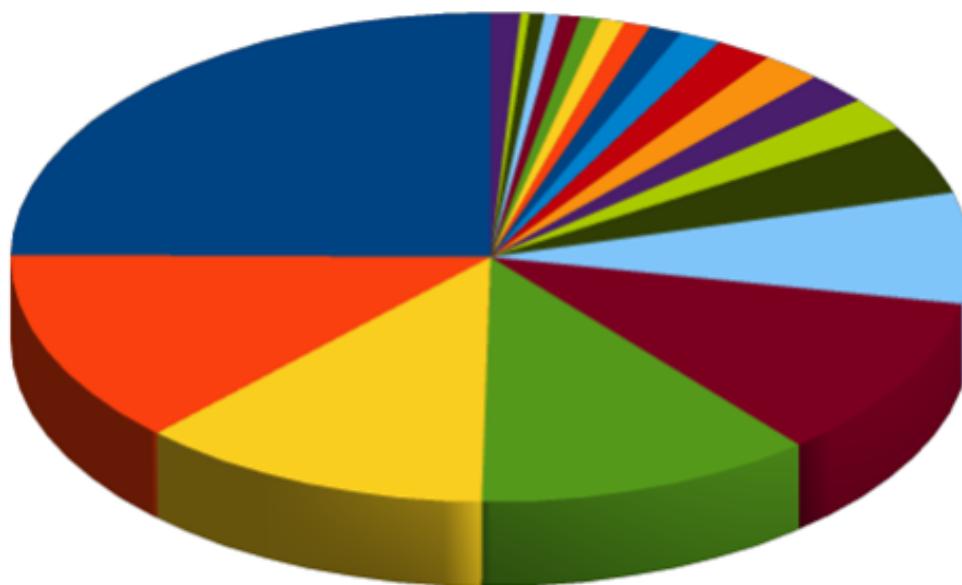


Source : [Stackoverflow](#) 2020



# Etude de developpez.com 2019

Popularité des langages dans les offres d'emploi en 2019



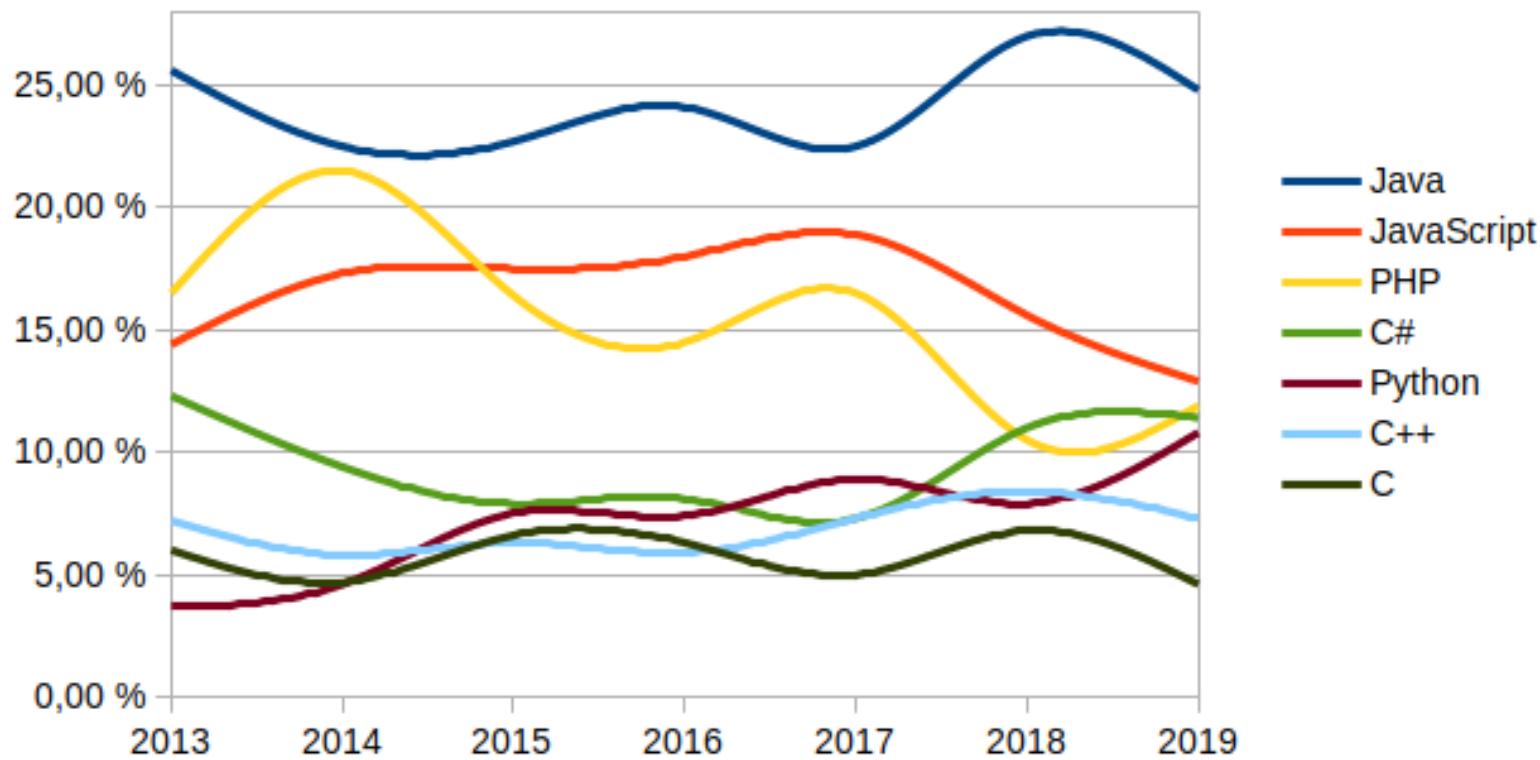
- Java (24,8%)
- JavaScript (12,9%)
- PHP (11,9%)
- C# (11,4%)
- Python (10,8%)
- C++ (7,3%)
- C (4,6%)
- R (2,2%)
- Scala (2,1%)
- TypeScript (2,0%)
- Perl (1,9%)
- VBA (1,4%)
- Go (1,1%)
- VB.NET (0,9%)
- Delphi (0,8%)
- Swift (0,7%)
- Ruby (0,7%)
- Windev (0,5%)
- Matlab (0,5%)
- Cobol (0,3%)
- Autres

: [emploi.developpez.com](http://emploi.developpez.com) 2019



# Etude de developpez.com 2019

Évolution de la demande dans les offres d'emploi des langages populaires

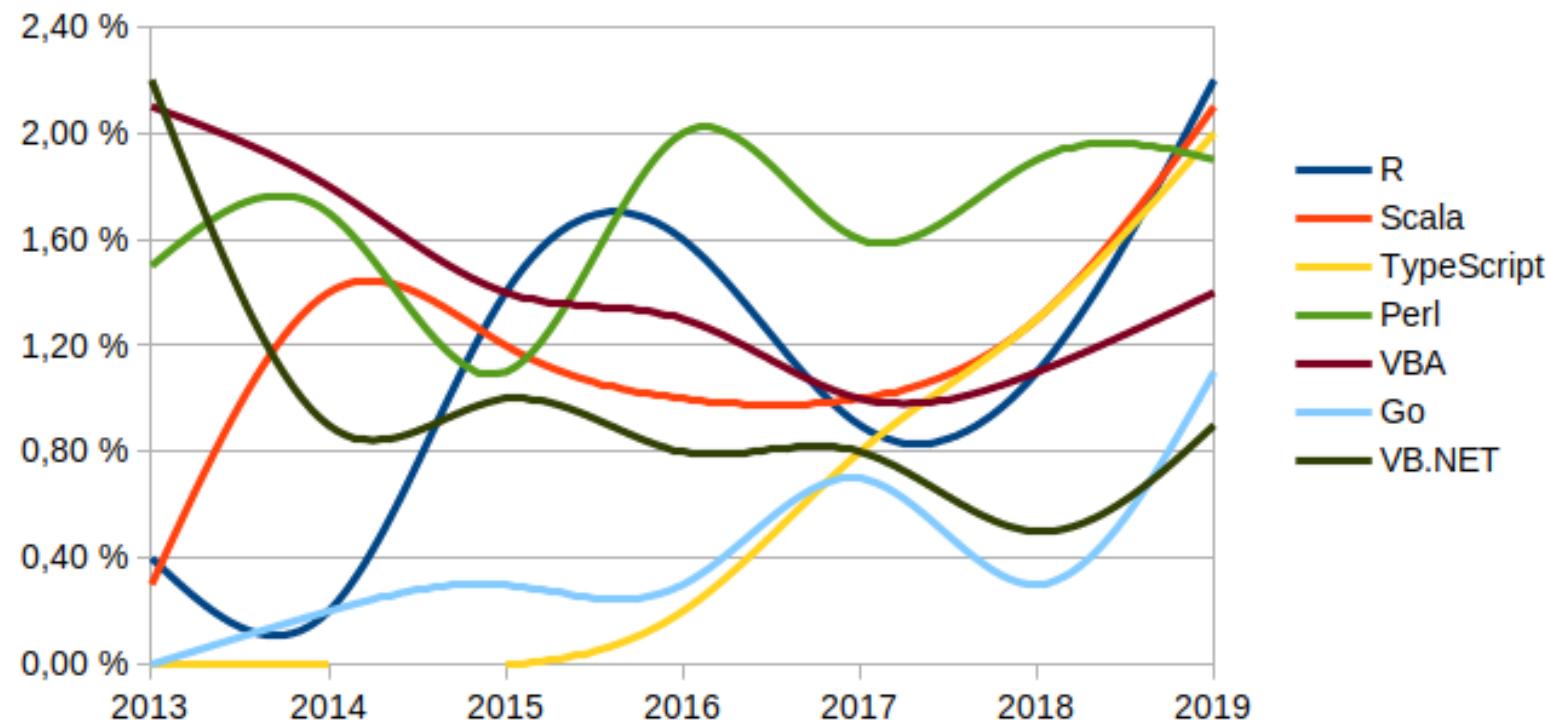


Source : [emploi.developpez.com](http://emploi.developpez.com) 2019



# Etude de developpez.com 2019

Évolution de la demande dans les offres d'emploi des langages plus confidentiels

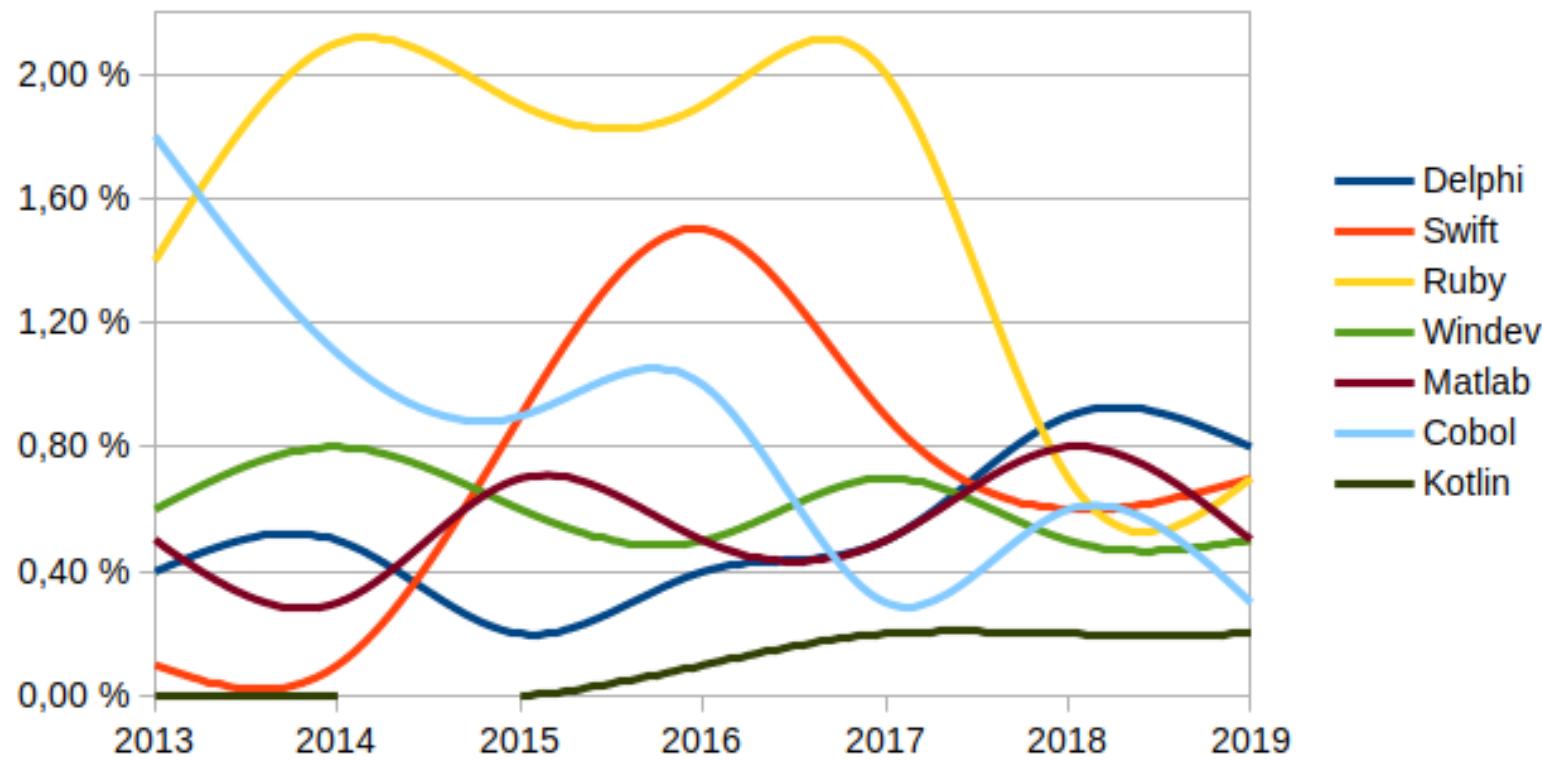


Source : [emploi.developpez.com](http://emploi.developpez.com) 2019



# Etude de developpez.com 2019

Évolution de la demande dans les offres d'emploi des langages de niche

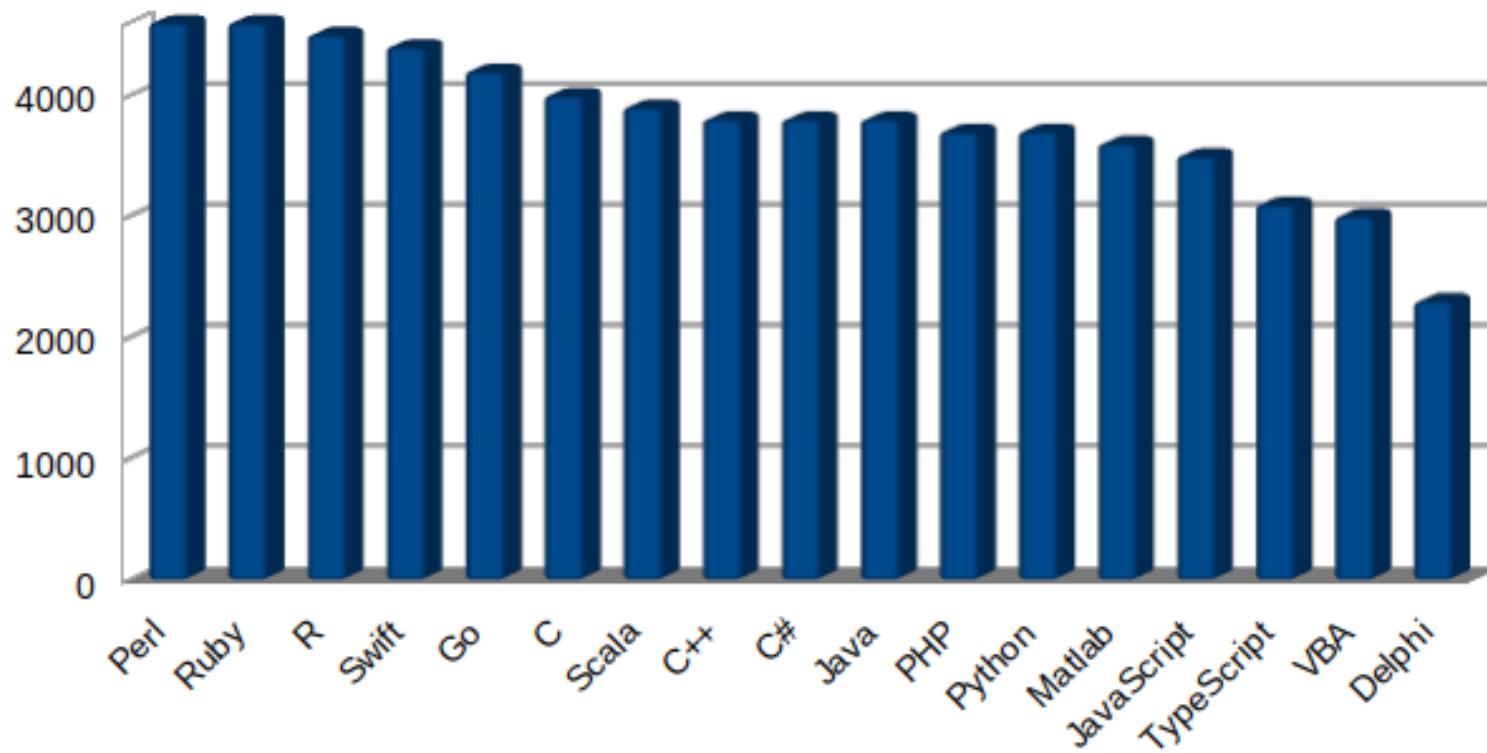


Source : [emploi.developpez.com](http://emploi.developpez.com) 2019



# Etude de developpez.com 2019

Salaires proposés en fonction des langages en région parisienne en 2019

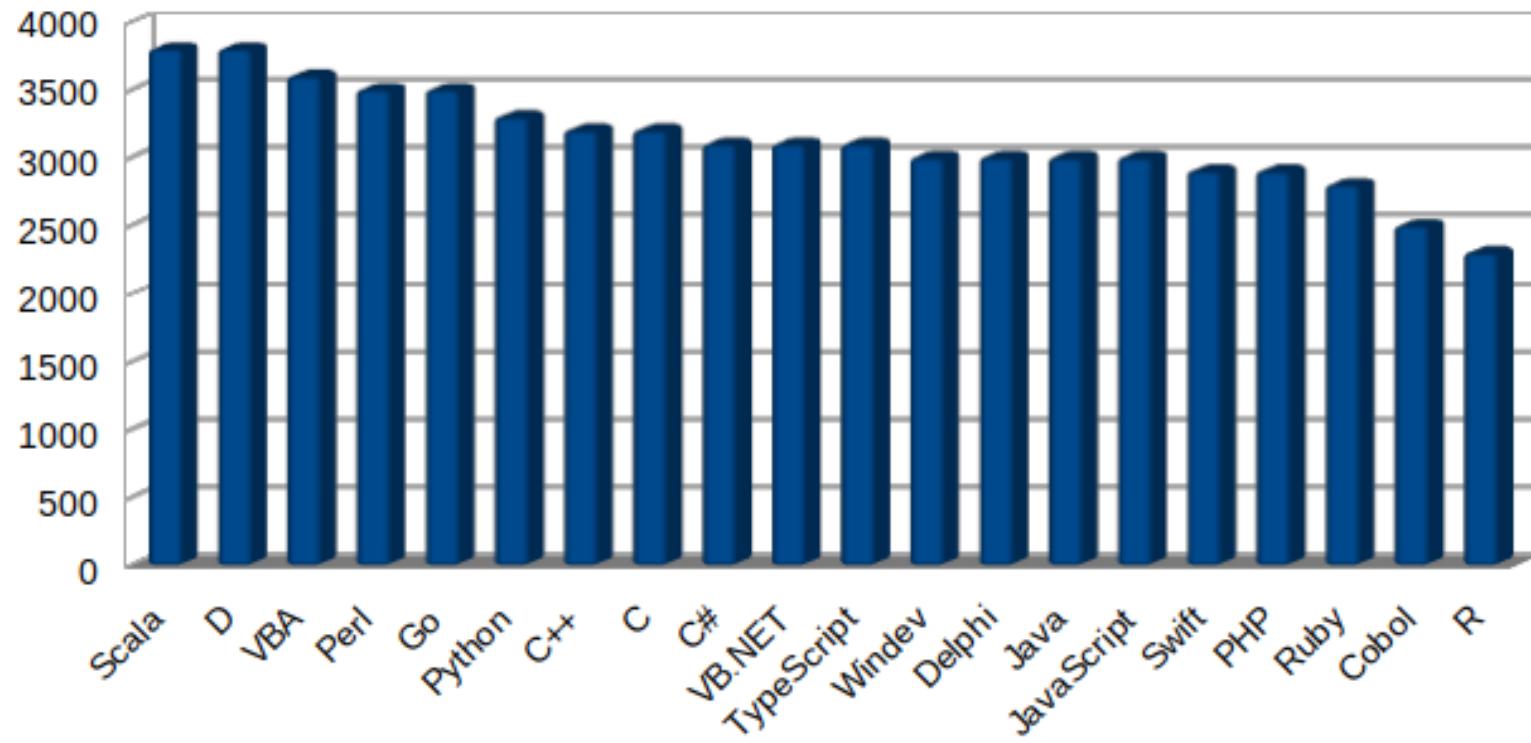


Source : [emploi.developpez.com](http://emploi.developpez.com) 2019



# Etude de developpez.com 2019

Salaires proposés en fonction des langages en province en 2019



Source : [emploi.developpez.com](http://emploi.developpez.com) 2019

---

# **PYTHON**

## **QUE PEUT ON EN ATTENDRE ?**

# Python

## Qualités du langage

- facile à apprendre, même pour les non-informaticiens
- notation légère, utilisation de l'indentation, lisible
- richesse d'expression
- fonctionnel
- orienté objet
- efficacité
- concision
- langage interprété
- scripting
- Open source
- Bibliothèques riches et nombreuses

## Bibliothèques

- Graphiques : wxPython, pyQT, pyGtk
- Traitement d'image : PIL, Pillow
- Base de données : SQLAlchemy
- Web : Requests, Scrapy, Django, Flask, BeautifulSoup
- Réseaux : twister, scrapy
- Analyse de données : Panda, Numpy, SciPy
- Visualisation de données : matplotlib
- Big data : Hadoop, PyDoop
- Jeux : Pygame, pyglet
- Calcul formel : SymPy
- Traitement du langage naturel : nltk
- Interface à windows : pywin32

---

# **PYTHON**

# **HISTOIRE ET INSPIRATION**

# Guido van Rossum

- Hollandais
- né le 21 Janvier 1956
- crée Python
  - Février 1991, version 0.9.0
  - Juillet 2010, Python 2.7
  - Décembre 2008, **Python 3.0**
  - Janvier 2020, Python 2.7 n'est plus supporté
    - [Compte à rebours](#) expiré !
    - [Roadmap](#)
- Il était jusqu'à 2018 « *Benevolent Dictator For Life* »(BDFL) pour Python
  - dictateur bienveillant à vie

Celui que nous apprenons



By Doc Searls - 2006oscon\_203.JPG, CC BY-SA 2.0,  
<https://commons.wikimedia.org/w/index.php?curid=4974869>

# Un style marqué par l'humour des Monty Python



Graham Chapman, John Cleese, Eric Idle, Michael Palin, Terry Jones, Terry Gilliam  
Monty Python's Flying Circus

# PEP-0020 Le Zen de Python (par Tim Peters)

Beautiful is better than ugly.

Explicit is better than implicit.

Simple is better than complex.

Complex is better than complicated.

Flat is better than nested.

Sparse is better than dense.

Readability counts.

Special cases aren't special enough to break the rules.

Although practicality beats purity.

Errors should never pass silently.

Unless explicitly silenced.

In the face of ambiguity, refuse the temptation to guess.

There should be one-- and preferably only one --obvious way to do it.

Although that way may not be obvious at first unless you're Dutch.

Now is better than never.

Although never is often better than \*right\* now.

If the implementation is hard to explain, it's a bad idea.

If the implementation is easy to explain, it may be a good idea.

Namespaces are one honking great idea -- let's do more of those!



Préfère :

la beauté à la laideur,  
l'explicite à l'implicite,  
le simple au complexe  
et le complexe au compliqué,  
le déroulé à l'imbriqué,  
l'aéré au compact.

Prends en compte la lisibilité.

Les cas particuliers ne le sont jamais assez pour violer les règles.

Mais, à la pureté, privilégie l'aspect pratique.

Ne passe pas les erreurs sous silence,  
... ou bâillonne-les explicitement.

Face à l'ambiguïté, à deviner ne te laisse pas aller.

Sache qu'il ne devrait avoir qu'une et une seule façon de procéder,  
même si, de prime abord, elle n'est pas évidente, à moins d'être Néerlandais.

Mieux vaut maintenant que jamais.

Cependant jamais est souvent mieux qu'immédiatement.

Si l'implémentation s'explique difficilement, c'est une mauvaise idée.

Si l'implémentation s'explique aisément, c'est peut-être une bonne idée.

Les espaces de nommage ! Sacrée bonne idée ! Faisons plus de trucs comme ça.

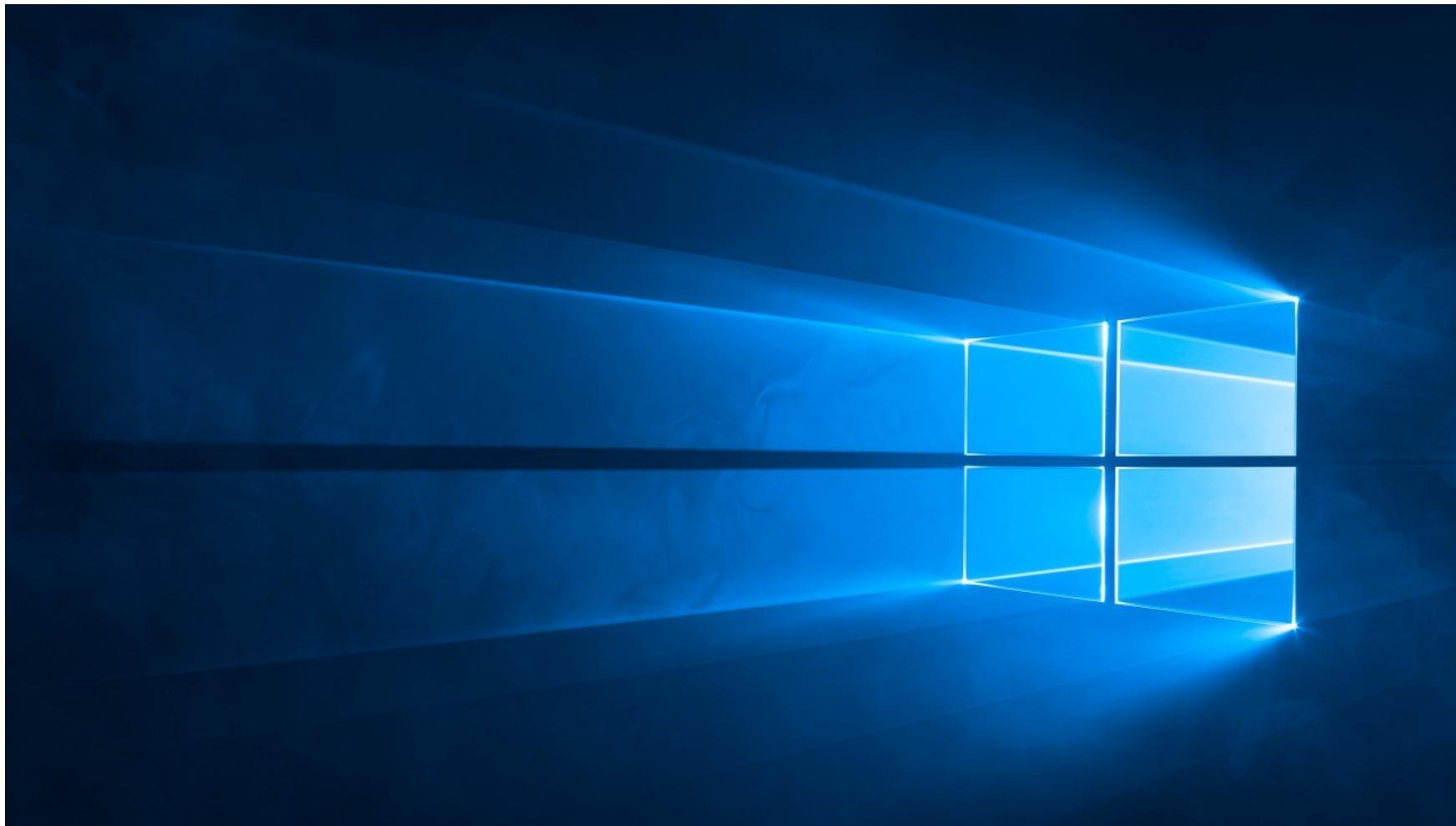


---

# **MISE EN PLACE DE L'ENVIRONNEMENT PYTHON3**

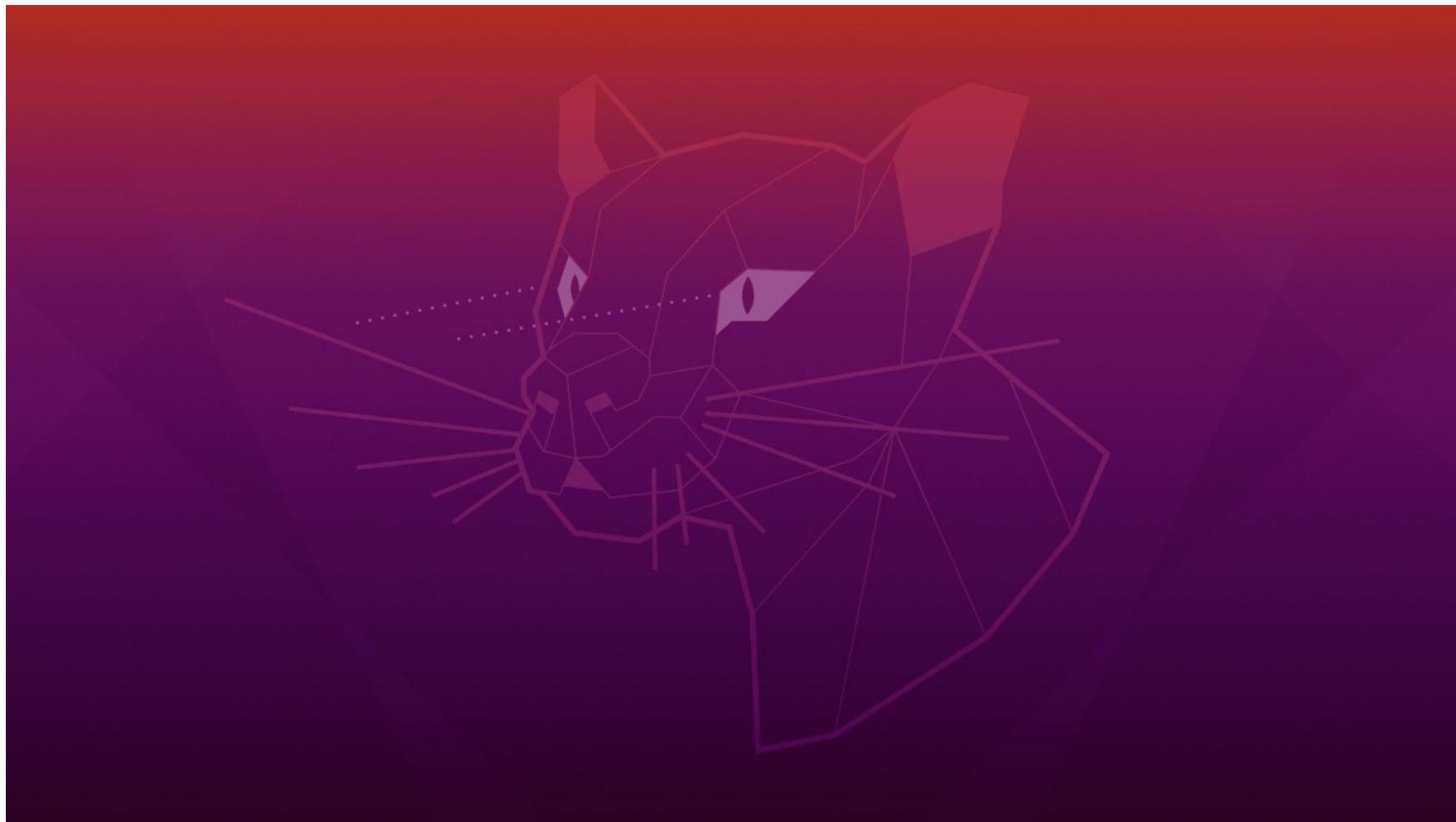


# Windows 10





# Ubuntu, Focal Fossa, 20.04 LTS





# Debian 9.5 (stretch)





# MacOS (Catalina)





# Win 10

- S'assurer que l'on dispose d'un nom+mot de passe
- Ouvrir un terminal
  - Menu démarrer + "cmd"
  - invite de commande
  - Shift + click droit et choisir "Ouvrir une fenêtre de commande ici"



# Environnement logiciel @ Win

- Vérifier la version d'OS

```
$ systeminfo
```

```
$ winver
```

- Vérifier la version de python

```
$ python --version
```

- Vérifier que le dossier Python et le dossier Python\Scripts sont dans PATH

```
$ echo %PATH%
```

- Lancer python par

```
$ python
```

```
$ py
```



# Debian

- S'assurer que l'on dispose d'un nom+mot de passe
  - root
- Ouvrir un terminal
  - CTL + ALT + Fn ou n>=2 et <7
  - Win+T+Enter
- Noter que la commande sudo est indisponible mais que la commande su est là.
- Ajouter la commande sudo reste possible
  - Eviter de travailler sous root



# Ajouter la commande sudo

- Repérer son nom d'utilisateur
- Installer sudo
- Ajouter l'utilisateur au groupe des sudoers
- Vérifier le contenu du fichier /etc/sudoers
- Ajouter éventuellement les utilisateurs qui manqueraient
- Redémarrer, vérifier

```
$ whoami
```

```
$ apt-get install sudo -y
```

```
$ usermod -aG sudo Le-nom
```



# Environnement logiciel

- Vérifier la version d'OS
- Vérifier la version de python
- Trouver python3
- (Introduire un alias dans le .bashrc pour pouvoir taper simplement « python »)

```
$ lsb_release -d  
$ lsb_release -a
```

```
$ python  
$ python3  
$ python --version  
$ python3 --version  
$ python3 -V
```

```
$ whereis python3
```

```
$ vi .bashrc  
$ gedit .bashrc
```

```
...  
alias python=/usr/bin/python3.5
```



# Environnement logiciel

## Pour ouvrir un terminal :

- par le Launchpad
    - saisir terminal
    - choisir terminal
  - par le Finder
    - Ouvrir Application/Utilitaires
    - 2x clic sur terminal
  - Ou bien
    - Outils et raccourcis vers les outils...



```
Last login: Wed Oct 17 18:13:08 on ttys000
MacBook-Pro-9:~ helloigor$ cd ~/Documents
MacBook-Pro-9:Documents helloigor$
```

# Puis, comme UNIX.

---

# **MISE EN PLACE DE L'ENVIRONNEMENT PYTHON3**



# Python en ligne de commande

- Observer la forme de l'invite
- Envoyer quelques commandes et observer les résultats
- Bonne pratique : help(...)
- A l'aide de votre éditeur de texte préféré, créer un fichier script.py avec le contenu ci-contre
- Puis envoyer la commande
- On peut faire aussi

```
>>>  
>>> 3  
>>> 3+2  
>>> a=2  
>>> a*7  
>>> Hello, World !  
>>> print("Hello, World ! ")  
>>> help (quit)  
>>> quit()  
>>> exit()
```

```
print("Hello, World ! ")
```

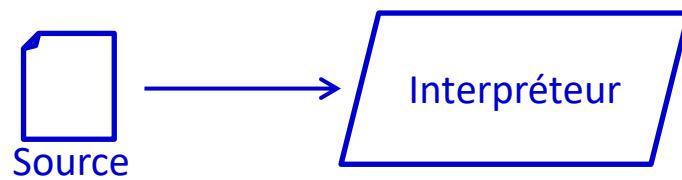
```
$ python script.py
```

```
$ python -m script
```

# Interprété ou Compilé

## Interpréteur

- Boucle
  - Lire
  - Evaluer
  - Imprimer
- REPL = Read – Eval – Print Loop



script.py

Python est essentiellement interprété

## Compilateur

- Code source
- Code objet
  - Exécuté par le processeur (machine)
  - ou par une machine virtuelle



script.py → script.pyc

Les .pyc sont interprétés plus rapidement



# \*.pyc

- Après avoir fait
- Noter l'apparition d'un nouveau dossier
  - pas toujours
  - quand il y a des imports
- Se déplacer dans ce dossier et examiner son contenu

```
$ python script.py
```

```
$ dir  
... <REP> __pycache__
```

```
$ cd __pycache__  
$ dir
```



# L'IDE natif de Python : idle

- Parmi les programmes disponibles
  - raccourcis sur le bureau
  - Bouton démarrer et tous les programmes

rechercher Idle

- Deux types de fenêtres IDLE
  - Exécution de l'interprète/Shell
  - Edition de sources

The image displays two windows of the Python IDLE application. The left window is titled 'Python 3.7.3 Shell' and shows a command-line interface with the following text:  
Python 3.7.3 (v3.7.3:ef4ec6d12, Mar 25 2019, 21:26:53) [MSC v. 1916 32 bit (Intel)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.  
>>>  
===== RESTART: C:/Users/Georges/Desktop/mes execos/script.py =====  
Bonjour  
>>>

The right window is titled 'script.py - C:/Users/Georges/Desktop/mes execos/script.py (3.7.3)' and shows a script editor with the following code:  
print("Bonjour")

Integrated Development Environment



# PIP, pour installer les packages

- PIP signifie – au choix–
  - PIP Installs Packages
  - Preferred Installer Program
- Vérifier que pip est installé
- Seulement si ce n'est pas le cas,  
l'installer,
  - Rechercher get-pip.py en  
<https://bootstrap.pypa.io/get-pip.py>
  - Puis

```
$ pip --version
```

```
$ python get-pip.py
```



# Commandes PIP

- Lister les packages installés localement
- Mettre à jour un package
- Désinstaller un package
- Installer un package pour l'utilisateur en cours uniquement
  - par défaut, c'est pour tout le système
- Lister les packages utilisés
- Montrer les infos et les dépendances d'un package

```
$ pip list
```

```
$ pip install monpackage -U
```

```
$ pip uninstall monpackage
```

```
$ pip install monpackage --user
```

```
$ pip freeze > requirements.txt
```

```
$ pip show monpackage
```



# Environnements virtuels (venv) @ Win

- Vérifier que venv est installé
- C'est un script Python

```
$ python -m venv -h
```

- Vérifier la présence d'un dossier environnements dans votre Home Directory

```
$ dir environments
```



# Environnements virtuels (venv) @ Win

En l'absence d'un tel dossier...

- Créer un dossier environments et s'y rendre
- Créer l'environnement "essai"
- Activer l'environnement
- Désactiver l'environnement

```
$ mkdir environments  
$ cd environments
```

```
$ python -m venv essai
```

```
$ ./essai/Scripts/activate.bat
```

```
$ ./essai/Scripts/deactivate.bat
```



# Environnements virtuels (venv) @ Win

Dans le dossier environnement

- vérifier la présence d'un environnement pour Jupyter
- Si absent, il va falloir le créer comme expliqué plus loin.

```
$ cd environments  
$ dir
```



# Environnements virtuels (venv) @ Linux

- Installer venv

```
$ sudo apt install -y python3-venv
```

- Utiliser venv

```
$ mkdir environments  
$ cd environments
```

- pour créer un environnement
- examiner le contenu du dossier mon-envt

```
$ python3 -m venv mon_envt  
$ ls mon_envt
```

- Activer l'environnement

```
$ source mon_envt/bin/activate
```

- Quitter l'environnement

```
$ deactivate
```



# Jupyter @ Windows

- Vérifier que Jupyter est installé
- Sinon, l'installer dans un environnement "jupenv"
- Pour l'utiliser, activer d'abord l'environnement (si ce n'est déjà fait)
- Puis
- Le notebook se lance dans le navigateur

```
$ jupyter notebook
```

```
$ cd environments  
$ python -m venv jupenv  
$ ./jupenv/Scripts/activate.bat  
$ pip install jupyter
```

```
$ ./jupenv/Scripts/activate.bat
```

```
$ jupyter notebook
```



# Jupyter @ Linux

- Installer
  - pas pip3, dans l'environnement virtuel, c'est redirigé vers pip3
- Utiliser
  - Le notebook se lance dans le navigateur

```
$ cd ~/environments  
$ mkdir jupenv  
$ python3 -m venv jupenv  
$ source jupenv/bin/activate  
$ pip install jupyter
```

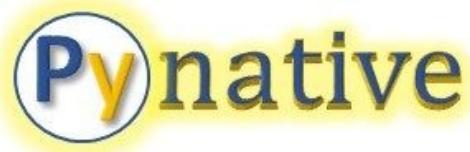
```
$ cd ~/environments  
$ source jupenv/bin/activate  
$ jupyter notebook
```

# Python en ligne

---



**OnlineGDB**  
[www.onlinegdb.com/online\\_python\\_interpreter](http://www.onlinegdb.com/online_python_interpreter)





# Pratique

---

C'est le moment de mettre en place votre dispositif de sauvegarde

- Clé USB
  - Dropbox
  - Google Drive
  - One Drive
- 
- Pour les supports de cours
  - Pour vos notes
  - Pour vos exercices, devoirs, projets

# Quizz (1/3)

---

- Qui est Guido van Rossum ?
- Que signifie PEP ?
- Python est-il interprété ou compilé ?
- Quelle est l'OS de développement utilisé dans ce cours ?
- Que signifie REPL ?
- Qu'est ce qu'un IDE ?
- Comment s'appelle l'IDE natif de Python ?
- Quel est l'IDE recommandé pour la suite de ce cours?
- Quelle est la version de Python que nous allons apprendre/pratiquer ?
- Pourquoi ?

# Quizz (2/3)

---

- Pourquoi apprendre Python ?
  - [ ] c'est un langage général
  - [ ] il fonctionne sur toutes les plateformes
  - [ ] il présente des concepts de programmation intéressants
  - [ ] les bibliothèques sont riches
- A quoi sert un deuxième écran quand on développe ?
  - [ ] à jouer pendant le travail (regarder des films)
  - [ ] à consulter Stackoverflow
  - [ ] à rechercher des templates de code sur internet
- Quelles sont les relations de Python avec l'internet des objets ?
  - [ ] code applicatif embarqué dans les objets connectés
  - [ ] code pour traiter les données remontées des objets connectés

# Quizz (3/3)

---

- En quoi Python est-il utile pour un administrateur système et réseau?
  - [ ] pour écrire des scripts plus puissants que les Shell Scripts
  - [ ] pour mieux communiquer avec les développeurs d'applications
  - [ ] pour supporter une démarche DEVOPS
- En quoi Python est-il utile pour l'analyse de données ?
  - [ ] pour la préparation des données
  - [ ] pour le nettoyage des données
  - [ ] pour la mise en forme, normalisation des données
  - [ ] à cause des fonctions et des bibliothèques statistiques disponibles
- Qu'est ce que le site StackOverflow ?

# Merci !

- Restons en contact :

- Georges Georgoulis – [ggeorgoulis@alteractifs.org](mailto:ggeorgoulis@alteractifs.org) – 06 12 68 40 06



- Coopérative d'activité et d'entrepreneurs [www.alteractifs.org](http://www.alteractifs.org)