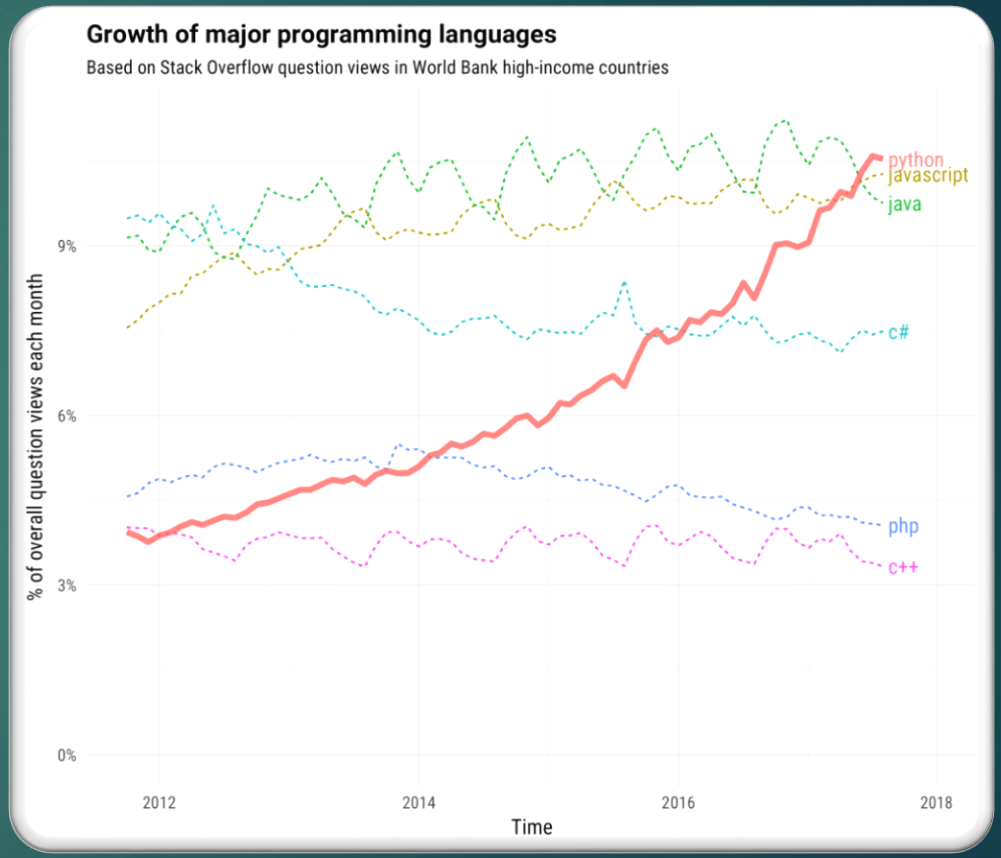
Introduction à Python 3

Damien Galasso 2020

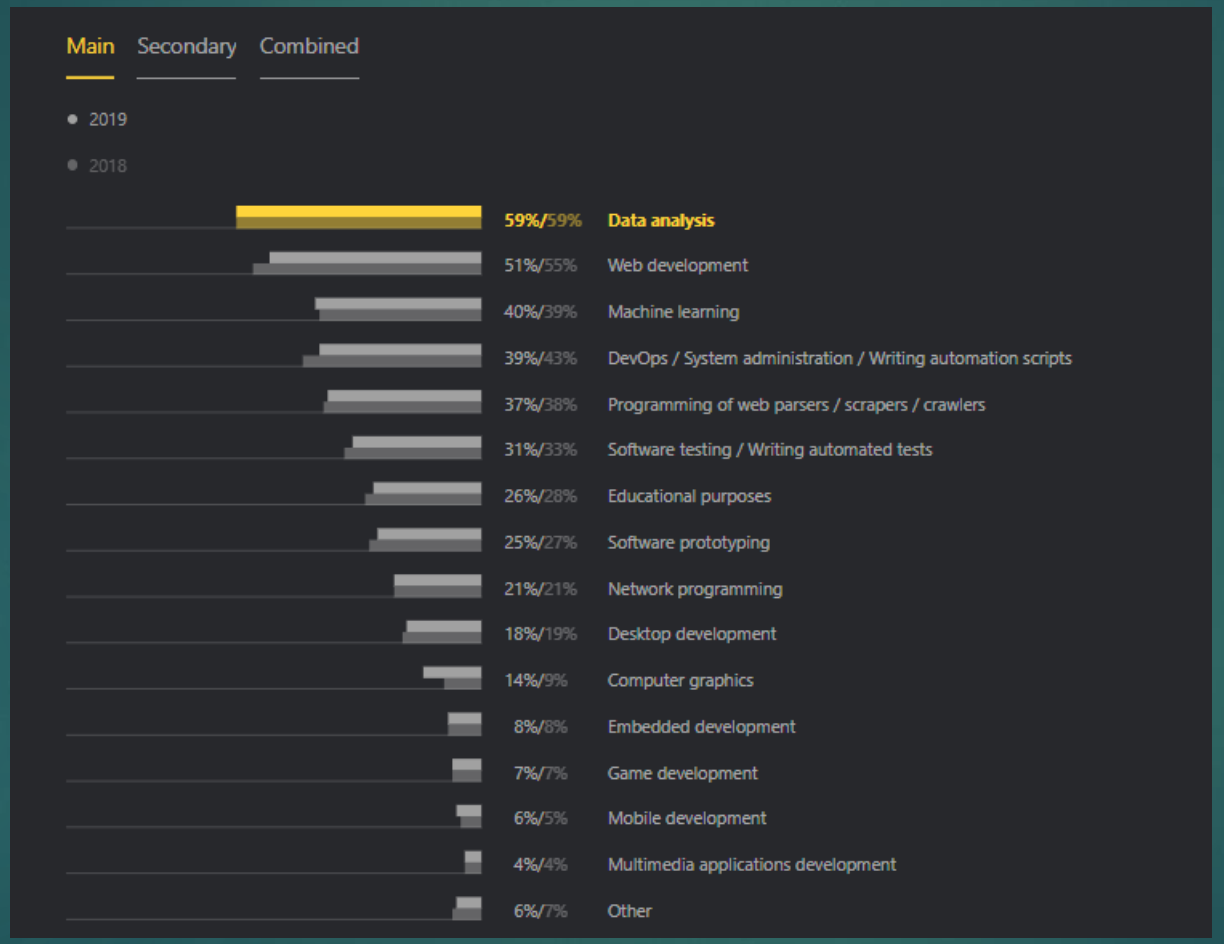
Pourquoi Python ?

- Facile à apprendre
- Opensource
- Très grande communauté
- Polyvalent
- Multiplateforme
- Flexibilité (librairies pour C, Java, .Net, Ruby)



Source: Stackoverflow.com

Utilisations de Python



Source: developpez.com

Particularités de Python

- Syntaxe avec l'indentation significative
- Propre Gestionnaire de paquet (pip)
- Typage dynamique
- Langage interprété
- Autogestion de l'allocation mémoire
- Grand nombre de bibliothèques
- Gestion des dépendances difficile

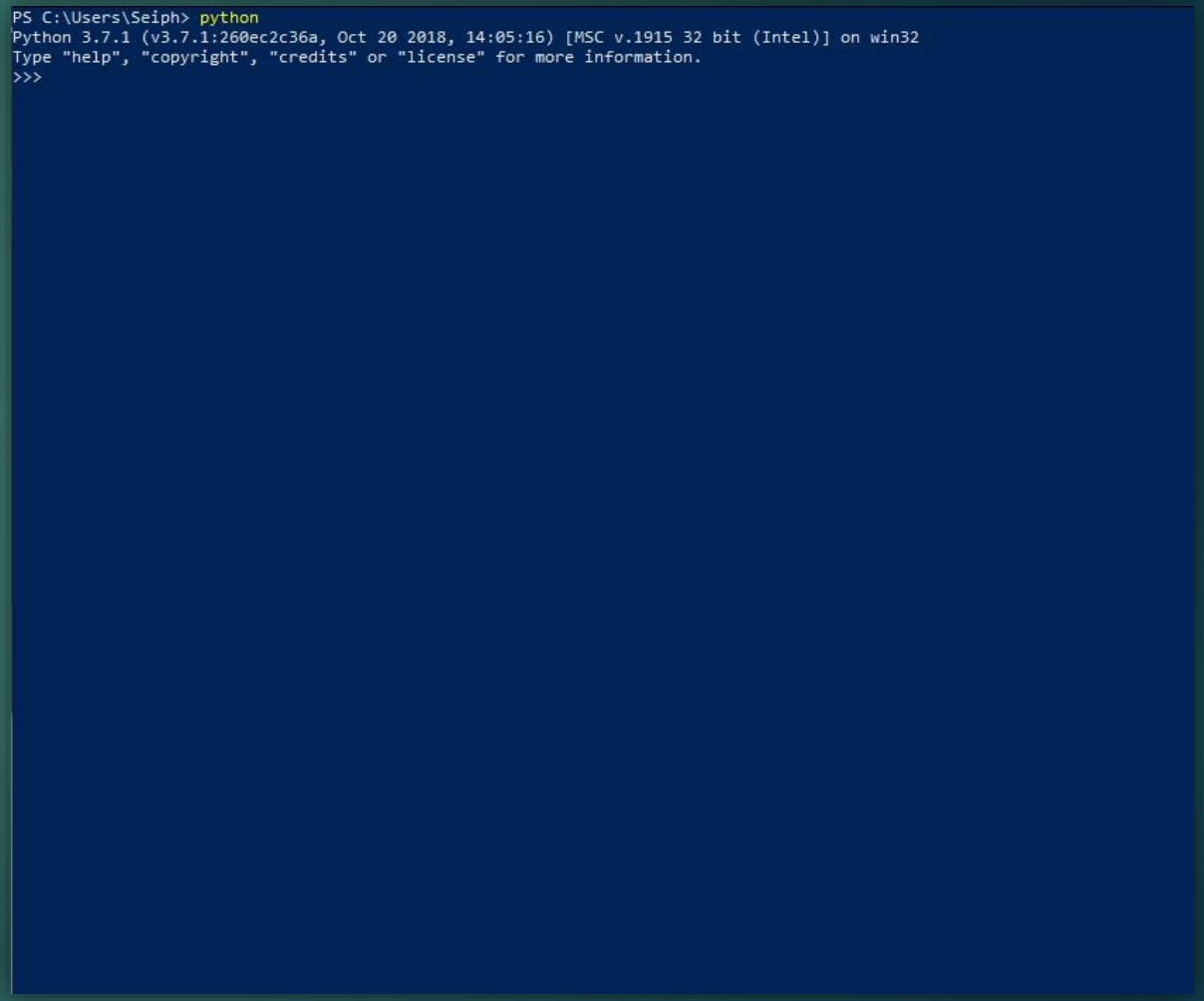
Installation

- Sur windows > https://www.python.org/ftp/python/3.8.3/python-3.8.3.exe

- Sur Linux(debian, ubuntu) > « apt install python3 »

Utilisation de l'interpréteur

- Sur Windows:
  - py
- Sur linux:
  - Python pour Python 2
  - Python3 pour Python 3



Créer une variable

- Entier
- Chaine de caractères
- Nombre flottant
- Booléen
- Liste
- Dictionnaire
- Tuple

```
5   Entier = 5
6
7
8   chaine_caracteres = "Bonjour ici !"
9
10
11  nbre_flottant = 5.12
12
13
14  booleen = True
15
16
17  liste = [1, 2, 3]
18
19
20  dictionnaire = {'key': 'value'}
21  print(dictionnaire['key'])
22  # value
23
24  tuple = (1, 2, 3)
```

Opérateurs et boucles

Opérateurs	Boucles
if	while
else	for
elif	

Comparaisons

Syntaxe Python	Signification
==	Egal à
!=	Différent de
>	Supérieur à
>=	Supérieur ou égal à
<	Inférieur à
<=	Inférieur ou égal à

Possibilités de comparer des nombres, des chaînes de caractères entre elles. Au sens large, des objets.

Listes


```python
# Création d'une liste
ma_liste = [5, 6, 7]
ma_liste2 = [1, 2, 3, 4]

# Accès éléments
print(ma_liste2[1])

# Ajout
ma_liste.append(8)
print(ma_liste)
# [5, 6, 7, 8]

# Insertion
ma_liste.insert(4, 9)
print(ma_liste)

```

```python
# Concatenation
ma_liste2.extend(ma_liste)
print(ma_liste2)

# Suppression par index
del ma_liste2[8]
print(ma_liste2)

# Suppression par valeur
ma_liste3 = [10, 30, 60]
ma_liste3.remove(30)
print(ma_liste3)

# Modification d'un élément
ma_liste3 [0] = 20
print(ma_liste3)
```

NB: A la différence d'une liste, un « tuple » est immuable

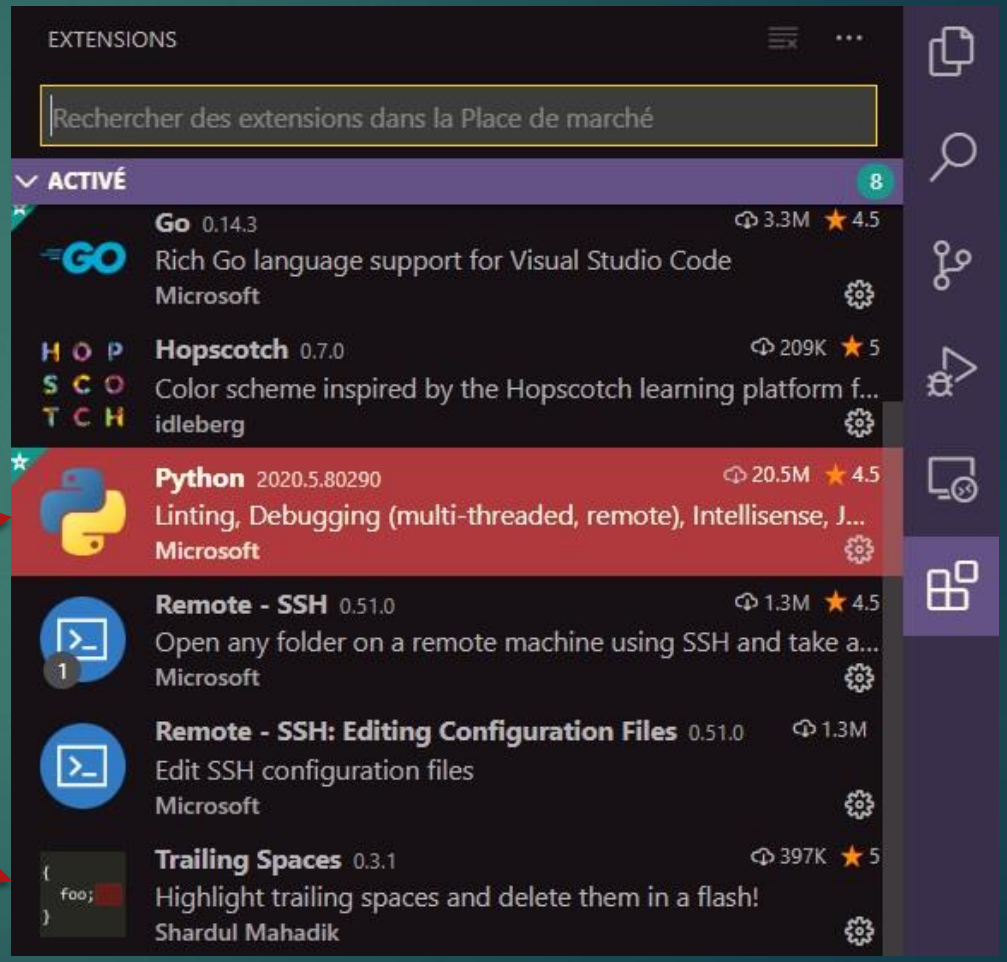
Chaînes de caractères


```
2   # Conversion Chaine en liste
3   chaine1 = "Bonjour à tous !"
4   liste_chaine = chaine1.split(" ")
5   print(liste_chaine)
6
7   # Conversion Liste en chaine
8   chaine2 = "-".join(liste_chaine)
9   print(chaine2)
10
11  # Parcourir une liste
12  for mot in liste_chaine:
13      print(mot)
14      # print(mot, end= " ")
```

Installation de Visual Studio Code

- Visual Studio Code

- Installer les modules
  - Python
  - et Trailing Spaces



Fonctions


```
1
2   # Définition de la fonction
3   def ma_fonction(parametre_age, prenom):
4       print("Je m'appelle {} et j'ai {} ans".format(prenom, parametre_age))
5       return None
6
7
8   # Appel de la fonction
9   ma_fonction(33, "Damien")
```

Créer et lancer un fichier *.py

- Sur Linux il se peut que vous ayez 2 pythons:
  - python => pour la version 2 de Python
  - python3 => pour la version 3 de python
- Sur Windows:
  - py pour la version 3
- Vérifier la version de python
  - python - -version
  - Ou python3 - -version
- Lancer un script:
  - python –m script.py

```
C:\Users\Seiph\Documents\Presentation_Python\fichiers>python --version
Python 3.7.1

C:\Users\Seiph\Documents\Presentation_Python\fichiers>python fonctions.py
Je m'appelle Damien et j'ai 33 ans
```

Programmation Orientée objet
Définition:

consiste en la définition et l'interaction de briques logicielles appelées *objets* ; un objet représente un concept, une idée ou toute entité du monde physique, comme une voiture, une personne. Il possède une structure interne et un comportement, et il sait interagir avec ses pairs. Il s'agit donc de représenter ces objets et leurs relations ; l'interaction entre les objets via leurs relations permet de concevoir et réaliser les fonctionnalités attendues, de mieux résoudre le ou les problèmes.

Source: Wikipédia

Programmation Orientée objet
Exemple:

```
2   class Voiture:
3       """Classe Voiture pour la creation ou modification d'une voiture """
4       nombre_roues = 4 # attribut de classe -> Une voiture a forcément 4 roues
5       def __init__(self, marque, modele): # Constructeur
6           self.marque = marque
7           self.modele = modele
8           self.couleur = "blanche"
9
10      def change_couleur(self, couleur): # Méthode d'instance
11          self.couleur = couleur
12          print("La couleur a changé")
13
14      def afficher_marque_modele(self):
15          return print(f"La voiture est une {self.modele} {self.marque}")
16
17  ma_voiture = Voiture("Megane", "Renault")
18  print(ma_voiture.couleur)
19  ma_voiture.change_couleur("bleue")
20  print(ma_voiture.couleur)
21  ma_voiture.afficher_marque_modele()
22  print(ma_voiture.nombre_roues)
```

Exercices

- Créer une liste [5, 4, 3, 2, 1]
  - Si le nombre est paire on n'affiche pas le nombre en question
  - Ajouter un « 6 »
  - Ajouter un 7 en première position de liste
  - Trier la liste par ordre croissant avec la méthode sort()
- A l'aide d'une boucle, créer le dessin suivant:
  - #####
  - ##
  - #####
  - ##
  - #####
- Créer une fonction qui prend 2 paramètres, et qui calcule la somme
- Ajouter dans la classe Voiture, une méthode de classe permettant de changer le modèle

Merci de votre attention