

Ligne de commande

Lancer python	python
Sortir de python	CTRL+Z+ENTER
Connaitre la version	python --v
Lancer un programme python	python <nom_du_programme.py>

PIP (Package Manager)

Connaitre la version de PIP	pip --version	
Installation d'un paquet (dernière version)	pip install <package>	
Désinstallation d'un paquet	pip uninstall <package>	
Lister les packages	pip list	
Lister les paquets (format requirements.txt)	pip freeze	
Informations sur un paquet (<i>notamment les dépendances</i>)	pip show <package(s)>	pip show requests numpy pandas
Installation d'une version d'un paquet	pip install <package>==major.minor.patch	pip install requests==2.1.0
Installation d'une version d'un paquet (au sein d'une major)	pip install <package>~major.minor	pip install requests~2.2
Installation d'une version d'un paquet (au sein d'une minor)	pip install <package>~major.minor.patch	pip install requests~2.1.0
Installation de la dernière version d'un paquet au delà d'une version	pip install <package>>major.minor.patch	pip install requests>2.5.0
Installation de la dernière version d'un paquet entre 2 versions	pip install "<package>>major.minor.patch,<major.minor.patch"	pip install "requests>2.4.0,<2.6.0"
Suppression de tous les paquets	pip freeze > requirements.txt pip uninstall -r requirements.txt -y	

Import des paquets

Import d'un paquet	import <package>	import numpy
Import d'un paquet (déconseillé pour les conflits)	from <package> import *	from numpy import *
Import d'un paquet avec alias	import <package> as <alias>	import numpy as np
Import d'un module	import <module>	import os
Import d'une fonction d'un module	from <module> import <fonction>	from os import getcwd
Lister les modules	help('modules')	
Aide d'un module	python -m <module> --help	python -m venv --help

Un module est du code exécutable et les définitions de classe/fonction contenus dans un fichier Python unique.

Un paquet (ou package) est une collection de modules regroupés logiquement dans un répertoire et partageant un fichier de configuration (`__init__.py`)

Environnement virtuel

Aide du module venv	python -m venv --help
---------------------	-----------------------



Environnement virtuel (cont)

Création d'un environnement virtuel	<code>python -m venv <environment name></code>
Activation de l'environnement virtuel (shell Windows)	<code><environment name>\Scripts\activate.bat</code>
Sortir de l'environnement virtuel	<code>deactivate</code>

Le module venv est disponible à partir de la version 3.3

Liens (copy)

Index des paquets python	https://pypi.org/
Python en ligne	https://colab.research.google.com/?utm_source=scs-index

Tests unitaires

Exemple de méthode à tester	<pre>def to_absolute(number): """ Return the absolute value :param number: Initial number :return: The absolute value """ if number <= 0: return -number return number</pre>
-----------------------------	---

Ecriture d'un doctest	<pre>def to_absolute(number): """ >>> to_absolute(3) 3 """ if number <= 0: return -number return number</pre>
-----------------------	---

Lancement d'un doctest	<code>python -m doctest <nom du fichier></code>
Lancement d'un doctest (avec traces)	<code>python -m doctest -v <nom du module></code>
Librairie Pytest / Installation	<code>pip install -U pytest</code>

Librairie Pytest / Implémentation d'un test dans une classe test.py	<pre>from source import reverse_str def test_should_reverse_string(): assert reverse_str('abc') == 'cba'</pre>
Librairie Pytest / Lancement de la classe de test	<code>pytest test.py</code>
Librairie Pytest / Lancement de la classe de test en mode verbeux	<code>pytest -v test.py</code>
Librairie Pytest / Lancement d'une méthode de la classe de test	<code>pytest -v test.py::test_should_reverse_string</code>
Librairie Pytest / Lancement des classes de test commençant ou terminant par test	<code>pytest</code>

Programmation orientée objet

Programmation orientée objet (cont)

Programmation orientée objet (cont)

Nom de Classe / Convention	Nom de classe avec majuscule. Si plusieurs mots: CapitalizedCase	Instanciation d'un objet (sans paramètre optionnel)	car=Carre(2) ou car=Carre(length=2)	Méthode vide (implémentation à faire ultérieurement)	def empty_method(self, length): pass
Variable / Convention	Nom de variable en minuscule. Indentation supplémentaire de 4 espaces.	Instanciation d'un objet (avec paramètre optionnel)	car2=Carr-e(1,"blue") car3=Carr-e(3,color="green")	Héritage	class Film: [...] class FilmCassette(Film):
Méthode/ Convention	Nom de méthode en minuscule séparés par des traits de soulignement. Indentation supplémentaire de 4 espaces.	Accéder à un attribut de l'objet (ici pour l'imprimer)	print(car2.c-olor)	Appeler une méthode de la classe parente	super() Exemple: super().__init__-_(name)
Constr-ucteur / Convention	Nom en minuscule, commencent et se terminent par deux underscores (ou « dunders »).	Modifier un attribut d'objet (l'utilisation d'une méthode, ici change_length() est conseillée)	car2.length=5	Exception. Levée	message = (f"This Function only supports halving even numbers. Received: {number}") raise Exception(message)
Classe Python	class Rectangle: width = 3 height = 2 def calculate_area(self): return self.width * self.height	Attribut d'instance	class Bird:: def __init__(self): self.wings = 2:	Exception. Gestion	try: return (after_value / initial_value) * 100 except ZeroDivisionError: return 0 except Exception as error: print("Uh oh, unexpected error occurred!") raise error
Constr-ucteur (__init__):	class Carre: length=0 def __init__(self,length): self.length=length	Attribut de classe	class Bird: names = ("mo-quette", "pig-eon", "moinea-u", "hirrondelle") positions = {} def __init__(-self, name): self.position = 1, 2		
Constr-ucteur (__init__): avec initialisation de variable	class Carre: length=0 def __init__(self,length=5): self.length=length				
Constr-ucteur (__init__): avec paramètre optionnel	class Carre: length=0 def __init__(self,length=5,color="red"): self.length=length self.color=color	Méthode de classe	@classmethod def find_bird(cls, position): if position in cls.positions: return f"On a trouvé un {cls.positions[position]} !" return "On a rien trouvé..."		
		Accéder à une méthode d'un objet	area = car2.c-alculate_area()		
		Méthode Statique	class Bird: @staticmethod def get_definition():{nl}}		

