

vendredi 4 décembre 2020



Le Club Informatique Gassendi



GASSENDI

TP monde connecté : cours du 26/11/2020 : configuration Pi, VNC, WinSCP

Élaboration	
4 décembre 2020	
Jean D	GASSENDI
Animateur	
Administration informatique	
Nom du fichier	00_TP_monde_connecte_cours_26_11_2020_Python_IDE-Thonny_V0.1.odt

Python

Généralités

Utilisation du langage Python pour réaliser des programmes de traitement des données.

Python est un langage **de haut-niveau, interprété, interactif et orienté objet**

intérêt du langage **haut niveau**:

code facile à lire, syntaxe peu complexe donc: beaucoup plus facile à écrire qu'en assembleur (gain de temps)

plus facile de modifier le programme et de le corriger

il est **portable** c'est-à-dire qu'il pourra fonctionner sur des machines différentes sans trop de modifications.

intérêt d'un langage **interprété**:

les instructions sont exécutées directement grâce à un interpréteur. Pas besoin de compilation avant l'exécution du programme.

intérêt d'un langage **interactif**:

le développeur peut écrire son programme directement sous le prompt Python « **>>>** ».

intérêt du langage **orienté objet**:

Permet d'encapsuler le code des programmes sous forme d'objet réutilisable dans d'autres programmes.

Python

Structure générale d'un programme Python

```
# -*- coding: utf-8 -*-
```

```
import pi  
from time import time.sleep
```

```
passe, dehors = 0, 3  
tempo = 0.2  
msg = "Chiffre en dehors de la plage. Recommencez!"  
touche = None  
code = []
```

```
# type int  
# type float  
# string  
# Boolean  
# liste
```

tabulation
de 4

```
def essai(a):  
....clavier = [[1, 2, 3], [4, 5, 6], [7, 8, 9], [ "*", 0, "#"]]  
.... ligne, colonne = 0+a, 2-a  
.... return clavier[ligne][colonne]
```

```
def val_code(x):  
....code_ref = [5, 3, 3, 5]  
....suite de la fonction
```

```
if __name__ == '__main__': # prg principal  
....while passe < 4 :  
..... chi = int(input("entrez n° ligne")  
.....while chi > dehors :  
.....print( msg)  
..... chi = int(input("entrez n° ligne")  
..... while touche == None:  
..... code.append(essai(chi))  
..... time.sleep(tempo)  
..... touche = not None  
..... print("chiffre", code)  
.....passe = passe + 1  
.....touche = None  
....val_code(code)
```

Utilisation du format universel utf-8

Zone d'insertion
des bibliothèques utilisées

Zone d'insertion
des constantes et variables
utilisées avec typage implicite

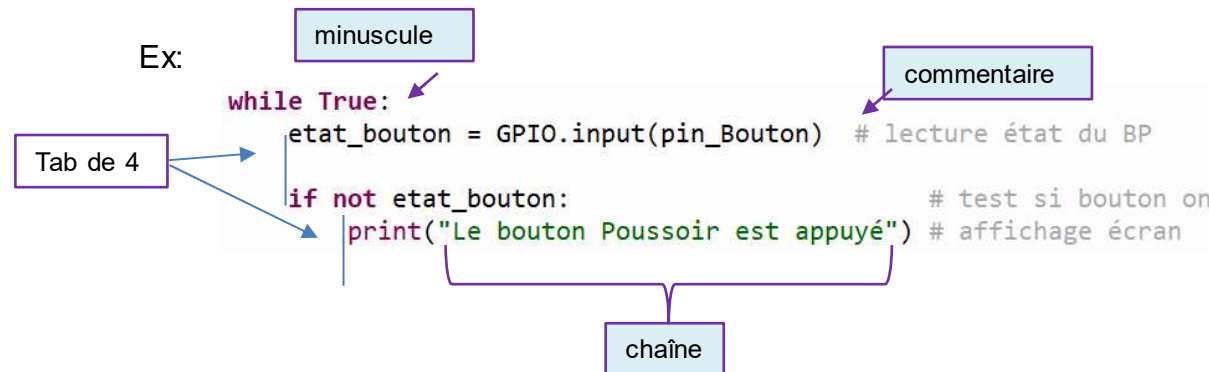
Zone d'insertion
des fonctions utilisées

Zone programme principal

Python

Contraintes et règles de rédaction Python

. Python est sensible à l'indentation (utilisation de la touche: **"TAB"** ⇔)



. Les instructions doivent respecter la syntaxe → sinon message d'erreur lors de l'exécution

pas de caractères accentués,
utilisation de minuscule,
les commentaires sont précédés de #
une chaîne de caractère est délimitée par:
'chaîne'
"chaîne" ou
" La phrase est écrite sur plusieurs lignes"

. Mots réservés à ne pas utiliser pour les variables ou fonctions

and, assert, break, class, continue, def, del, elif, else, except, exec, finally, from, for, global, if, import, in, is, lambda, not, or, pass, print, raise, return, try, while, with, yield

Python

Différents types d'erreurs

Erreurs de syntaxe: *(facile à corriger)*

un message d'erreur survient lors du « run » programme.

Erreurs de logique: *(plus difficile voir organigramme)*

pas de message d'erreur mais le programme n'a pas le résultat escompté

Erreurs à l'exécution: *(ardu introduire dans le programme le traitement d'erreur)*

plantage du programme (prévoir des exceptions sinon point de salut)

la fenêtre « run » indique le type de plantage et l'exception à instancier.

Exemples courant de plantage :

div/ 0,

test avec condition inconnue,

lire/écrire dans fichier mal répertorié.

Python

Grâce à sa portabilité, Python fonctionne également sous Windows.
mais uniquement en mode graphique avec l'IDE Thonny.

Pas de **mode console** comme sur la Raspberry (cf. [console](#))

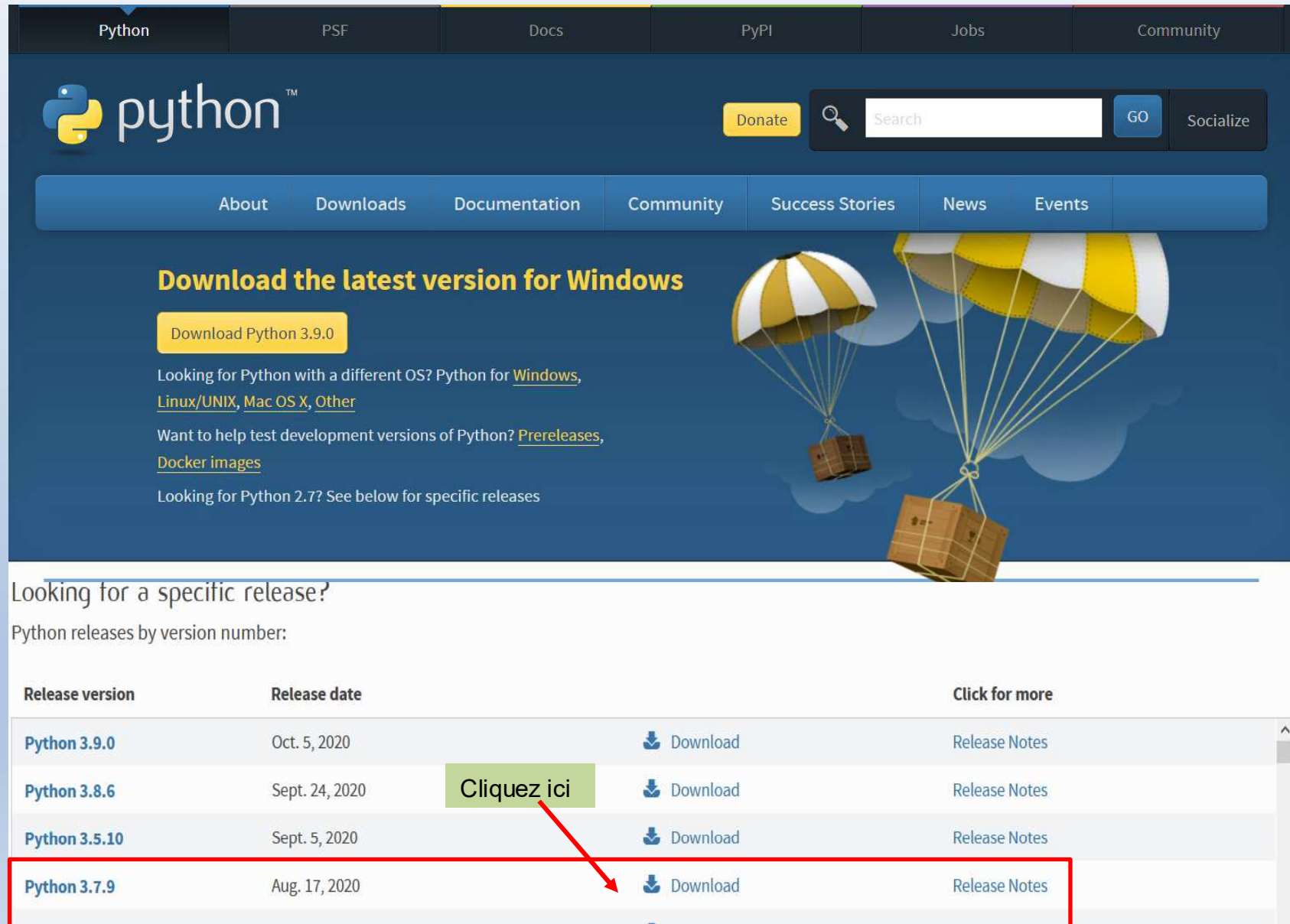
Télécharger Python sur le PC Windows

A: aller sur le site:

<https://www.python.org/downloads/release/python-379/>

Python

B: Choisir la version désirée:



Download the latest version for Windows

[Download Python 3.9.0](#)

Looking for Python with a different OS? Python for [Windows](#), [Linux/UNIX](#), [Mac OS X](#), [Other](#)

Want to help test development versions of Python? [Prereleases](#), [Docker images](#)

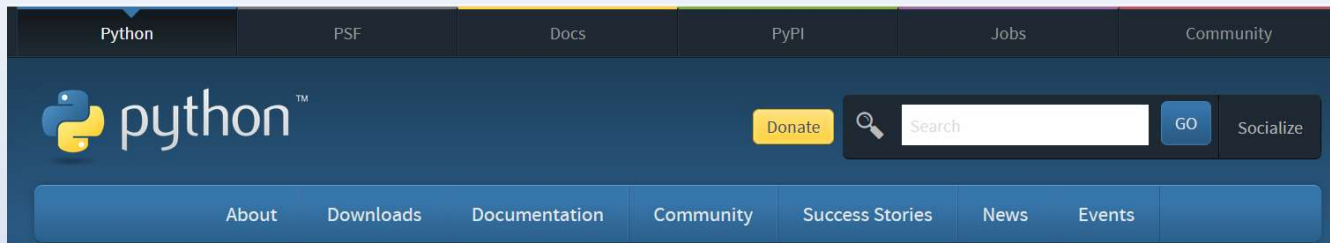
Looking for Python 2.7? See below for specific releases

Looking for a specific release?

Python releases by version number:

Release version	Release date	Click for more
Python 3.9.0	Oct. 5, 2020	Download Release Notes
Python 3.8.6	Sept. 24, 2020	Download Release Notes
Python 3.5.10	Sept. 5, 2020	Download Release Notes
Python 3.7.9	Aug. 17, 2020	Download Release Notes

Python



Python 3.7.9

Release Date: Aug. 17, 2020

Python 3.7.9 is the latest **security fix** release of Python 3.7.

Note

Python 3.8 is now the latest feature release series of Python 3. [Get the latest release of 3.8.x here](#). Python 3.7.8 was the last **bugfix release** for 3.7. Python 3.7 is now in the **security fix** phase of its life cycle. Only security-related issues are accepted and addressed during this phase. We plan to provide security fixes for Python 3.7 as needed until mid 2023, five years following its initial release. **Security fix** releases are produced periodically as needed and are **source-only** releases.

Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		bcd9f22cf531efc6f06ca6b9b2919bd4	23277790	SIG
XZ compressed source tarball	Source release		389d3ed26b4d97c741d9e5423da1f43b	17389636	SIG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	4b544fc0ac8c3cffdb67dede23ddb79e	29305353	SIG
Windows help file	Windows		1094c8d9438ad1adc263ca57ceb3b927	8186795	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	60f77740b30030b22699dbd14883a4a3	7502379	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	7083fed513c3c9a4ea655211df9ade27	26940592	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	da0b17ae84d6579f8df3eb24927fd825	1348904	SIG
Windows x86 embeddable zip file	Windows		97c6558d479dc53bf448580b66ad7c1e	6659999	SIG
Windows x86 executable installer	Windows		1e6d31c98c68c723541f0821b3c15d52	25875560	SIG
Windows x86 web-based installer	Windows		22f68f09e533c4940fc006e035f08aa2	1319904	SIG

Cliquez ici



Python

Ci-après des exemples qui illustrent l'utilisation de Python avec la **Raspberry**
en **mode inter-actif**.

mode console

Python

```
pi@raspberrypi~ $: python3 (lancement de l'interpréteur)
                        infos sur la version Python 3.1.1+

>>> 5+2                                     # instruction 1
7    # résultat 1

>>> print (' Bonjour ')                     # instruction 2
Bonjour                                     # résultat 2

>>> phrase="passage de minuscule en majuscule" # instruction 3
>>> print (phrase.upper)                     # utilisation de la méthode upper 4
PASSAGE DE MINUSCULE EN MAJUSCULE          # résultat 3,4

>>> from math import pi                       # appel de la fct pi dans la biblio math 5
>>> rayon=2                                  # instruction 6
>>> print ("La surface du cercle = ", pi ^ rayon**2) # instruction 7
La surface du cercle = 12.566                # résultat 5,6,7
```

Exemple code 'Objet'

Python est un logiciel adapté aux débutants et qui comporte de nombreuses applications

Nota: Toute la documentation sur Python est accessible en ligne.

<https://www.python.org/doc/>

Le **mode interactif** est limité à une suite d'instructions mais n'est pas adapté pour des programmes plus complexes et de taille importante.



Passage en **mode script** avec l'IDE

Python

Le **mode script** consiste à écrire un programme en utilisant un IDE. (**Thonny**)
(environnement de développement qui contient Editeur, Debugger, Exécutable)

Pour cela: Nous allons créer un fichier qui s'appellera: **tp1_bp.py**

py pour indiquer Python'

sous **Windows**

créer un répertoire '**TP_connect**'.

ouvrir l'IDE **Thonny** et enregistrer le fichier sous: **tp1_bp.py**

sous **Raspbian**

Se mettre en mode console

```
Nouveau bureau : masterCIGpi:1 (192.168.1.28:1)  
pi@masterCIGpi:~ $ mkdir TP_connect
```

Création répertoire TP_connect'

```
Nouveau bureau : masterCIGpi:1 (192.168.1.28:1)  
pi@masterCIGpi:~ $ cd TP_connect
```

Aller au répertoire TP_connect'

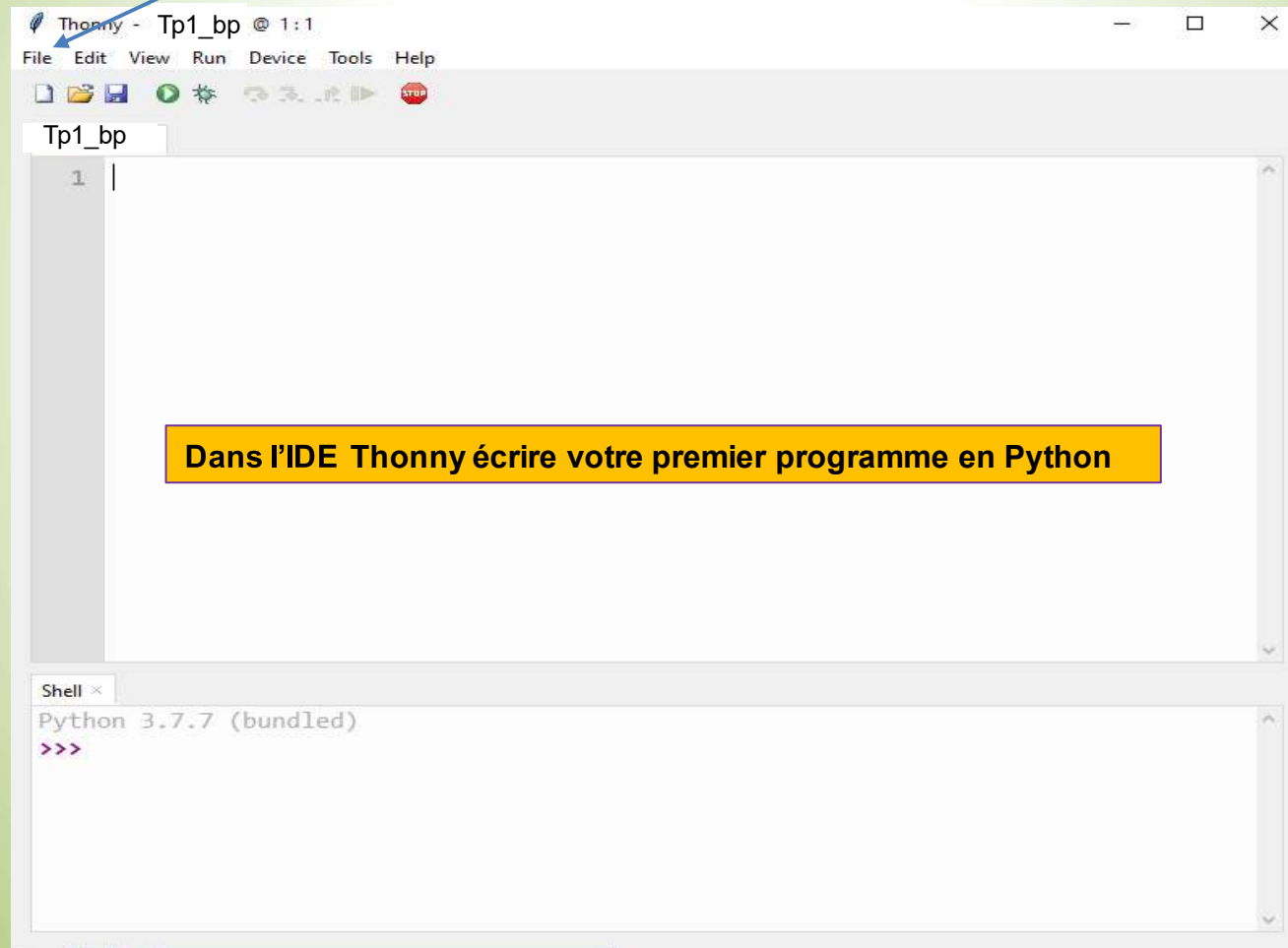
```
Nouveau bureau : masterCIGpi:1 (192.168.1.28:1)  
pi@masterCIGpi:~ $ touch tp1_bp.py
```

Créer un fichier vide **tp1_bp.py**

Python

Se mettre en mode graphique en ouvrant l'IDE Thonny

Dans « File » faire « Open » le fichier **Tp1_bp.py**



Dans l'IDE Thonny écrire votre premier programme en Python

Glossaire

Sigle	
IDE	I ntegrated D evelopment E nvironment

IDE Thonny

1. But

Un **IDE** est un **E**nvironnement de **D**éveloppement **I**ntégré qui propose un ensemble d'outils spécifiques dédiés aux travaux de la programmation.

L'IDE « **Thonny** » a été conçu pour les programmeurs débutants qui désirent apprendre le langage **Python**. Cet outil intègre son propre interpréteur Python 3.6.

2. Télécharger Thonny

A: aller sur le site:

<https://thonny.org>

<https://github.com/thonny/thonny/releases/download/v3.2.7/thonny-3.2.7.exe>

Thonny
Python IDE for beginners



Download version **3.2.7** for
[Windows](#) • [Mac](#) • [Linux](#)

NB! Windows installer is signed with new identity and you may receive a warning dialog from Defender until it gains more reputation.

Just click "More info" and "Run anyway".

IDE Thonny

B: Choisir la version désirée:



C: Télécharger la version choisie:

Nom	Modifié le	Type	Taille
 thonny-3.2.7.exe	17/08/2020 11:07	Application	14 813 Ko

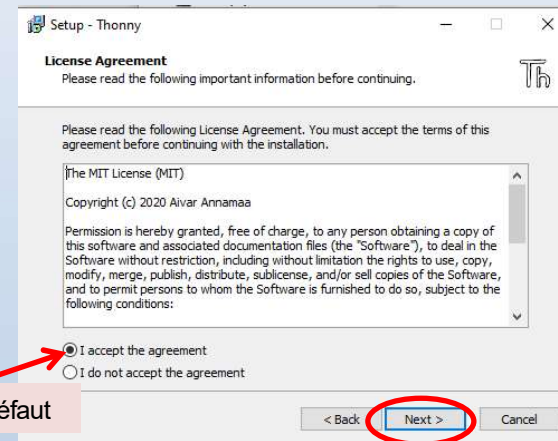
IDE Thonny

D: Installation de l'IDE sous Windows

Etape 1

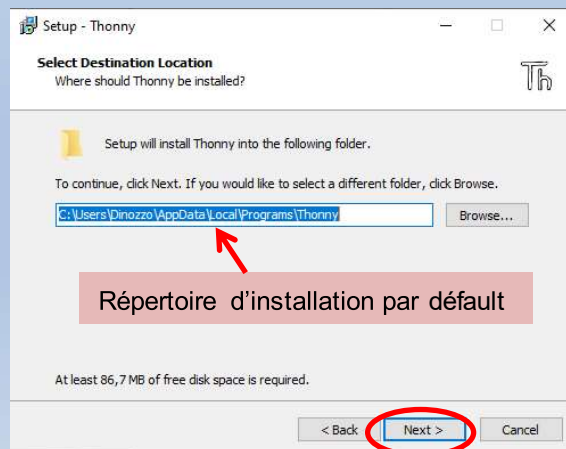


Etape 2



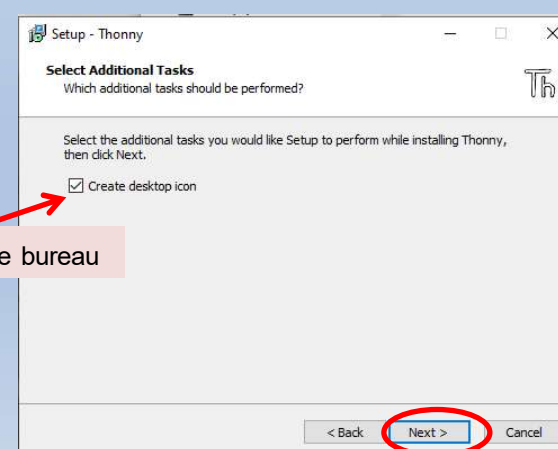
Par défaut

Etape 3



Répertoire d'installation par défaut

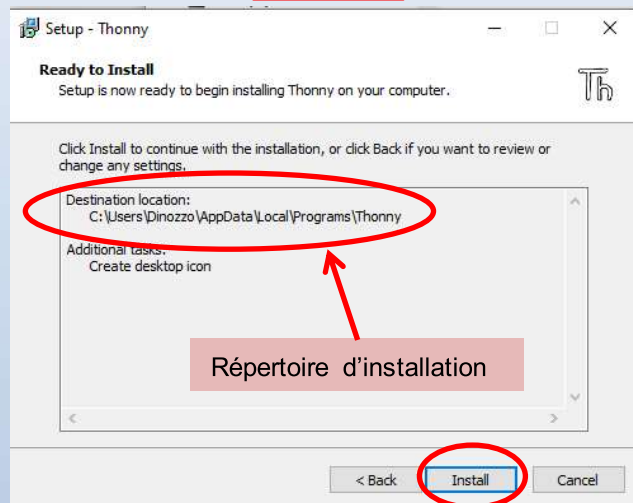
Etape 4



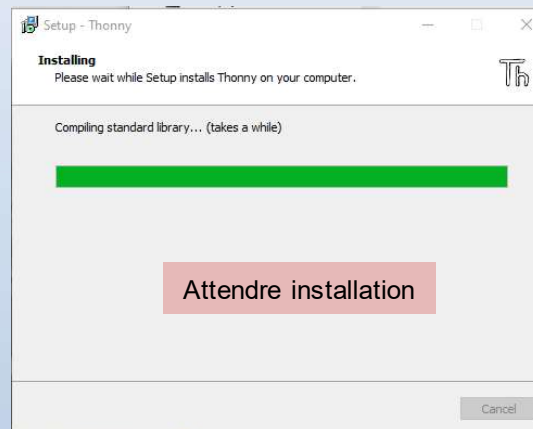
icône bureau

IDE Thonny

Etape 5



Etape 6

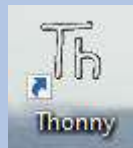


Attendre installation

Etape 7



3. Ouvrir l'IDE Thonny



IDE Thonny

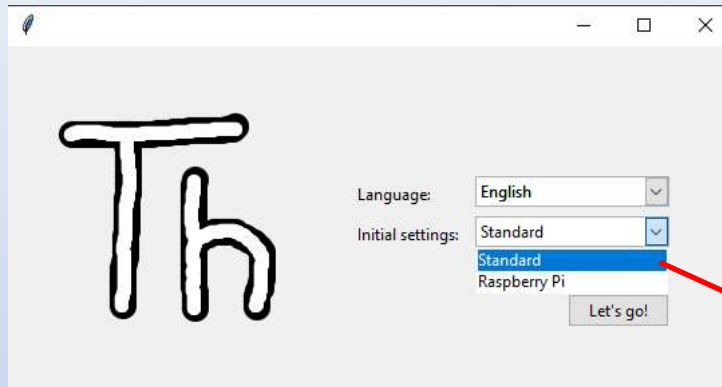
E: Détails du répertoire Thonny sous Windows

« Disque local (C:) » > Utilisateurs > Dinozzo > AppData > Local > Programs > Thonny

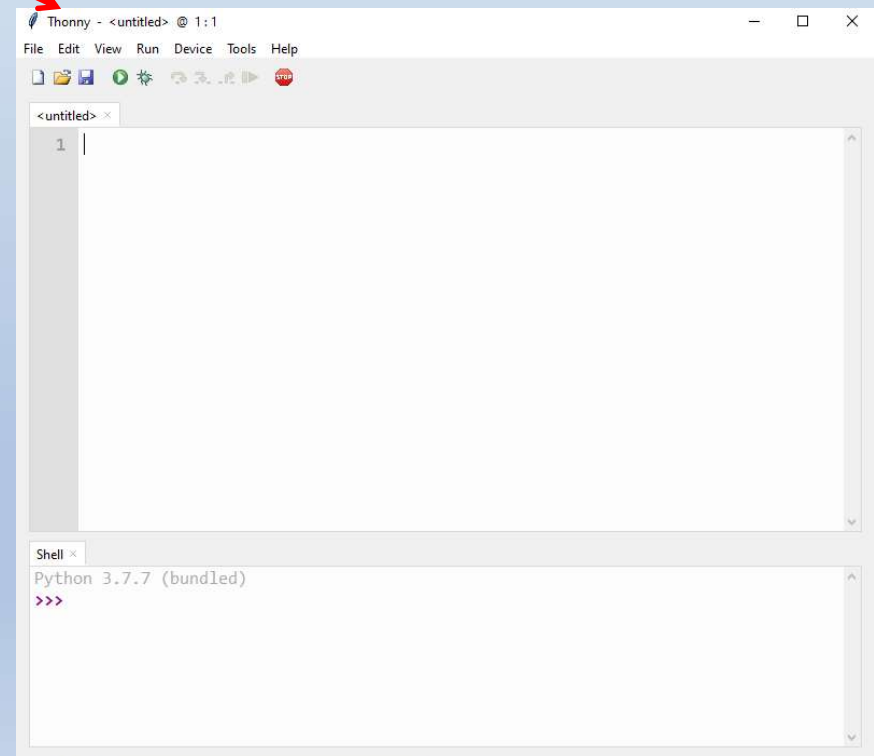
Nom	Modifié le	Type	Taille
DLLs	17/08/2020 11:24	Dossier de fichiers	
include	17/08/2020 11:24	Dossier de fichiers	
Lib	17/08/2020 11:25	Dossier de fichiers	
libs	17/08/2020 11:25	Dossier de fichiers	
licenses	17/08/2020 11:25	Dossier de fichiers	
Scripts	17/08/2020 11:25	Dossier de fichiers	
tcl	17/08/2020 11:25	Dossier de fichiers	
api-ms-win-core-file-l1-2-0.dll	06/01/2019 19:44	Extension de l'app...	12 Ko
api-ms-win-core-file-l2-1-0.dll	06/01/2019 19:44	Extension de l'app...	12 Ko
api-ms-win-core-localization-l1-2-0.dll	06/01/2019 19:44	Extension de l'app...	14 Ko
api-ms-win-core-processthreads-l1-1-1.dll	06/01/2019 19:44	Extension de l'app...	12 Ko
api-ms-win-core-synch-l1-2-0.dll	06/01/2019 19:44	Extension de l'app...	12 Ko
api-ms-win-core-timezone-l1-1-0.dll	06/01/2019 19:44	Extension de l'app...	12 Ko
api-ms-win-core-xstate-l2-1-0.dll	06/01/2019 19:44	Extension de l'app...	12 Ko
api-ms-win-crt-conio-l1-1-0.dll	06/01/2019 19:44	Extension de l'app...	13 Ko
api-ms-win-crt-convert-l1-1-0.dll	06/01/2019 19:44	Extension de l'app...	16 Ko
api-ms-win-crt-environment-l1-1-0.dll	06/01/2019 19:44	Extension de l'app...	12 Ko
api-ms-win-crt-filestream-l1-1-0.dll	06/01/2019 19:44	Extension de l'app...	14 Ko
api-ms-win-crt-heap-l1-1-0.dll	06/01/2019 19:44	Extension de l'app...	13 Ko
api-ms-win-crt-locale-l1-1-0.dll	06/01/2019 19:44	Extension de l'app...	12 Ko
api-ms-win-crt-math-l1-1-0.dll	06/01/2019 19:44	Extension de l'app...	22 Ko
api-ms-win-crt-multibyte-l1-1-0.dll	06/01/2019 19:44	Extension de l'app...	20 Ko
api-ms-win-crt-private-l1-1-0.dll	06/01/2019 19:44	Extension de l'app...	65 Ko
api-ms-win-crt-process-l1-1-0.dll	06/01/2019 19:44	Extension de l'app...	13 Ko
api-ms-win-crt-runtime-l1-1-0.dll	06/01/2019 19:44	Extension de l'app...	16 Ko
api-ms-win-crt-stdio-l1-1-0.dll	06/01/2019 19:44	Extension de l'app...	18 Ko
api-ms-win-crt-string-l1-1-0.dll	06/01/2019 19:44	Extension de l'app...	18 Ko
api-ms-win-crt-time-l1-1-0.dll	06/01/2019 19:44	Extension de l'app...	14 Ko
api-ms-win-crt-utility-l1-1-0.dll	06/01/2019 19:44	Extension de l'app...	12 Ko
api-ms-win-eventing-provider-l1-1-0.dll	06/01/2019 19:44	Extension de l'app...	12 Ko
CHANGELOG.rst	29/01/2020 22:01	Fichier RST	44 Ko
CREDITS.rst	09/03/2020 18:39	Fichier RST	4 Ko
LICENSE.txt	02/01/2020 08:40	Document texte	2 Ko
python.exe	10/03/2020 08:45	Application	96 Ko
python3.dll	10/03/2020 08:45	Extension de l'app...	58 Ko
python37.dll	10/03/2020 08:44	Extension de l'app...	3 361 Ko
pythonw.exe	10/03/2020 08:45	Application	95 Ko
README.rst	30/01/2019 22:47	Fichier RST	1 Ko
Thonny.exe	17/06/2019 07:53	Application	35 Ko
thonny_python.ini	06/01/2019 19:44	Paramètres de co...	1 Ko
ucrtbase.dll	06/01/2019 19:44	Extension de l'app...	869 Ko
unins000.dat	17/08/2020 11:25	Film Vidéo CD	1 436 Ko
unins000.exe	17/08/2020 11:18	Application	2 484 Ko
unins000.msg	17/08/2020 11:25	Élément Outlook	23 Ko
vcruntime140.dll	10/03/2020 08:39	Extension de l'app...	85 Ko

IDE Thonny

4. Paramétrage à la 1^{ère} ouverture de Thonny



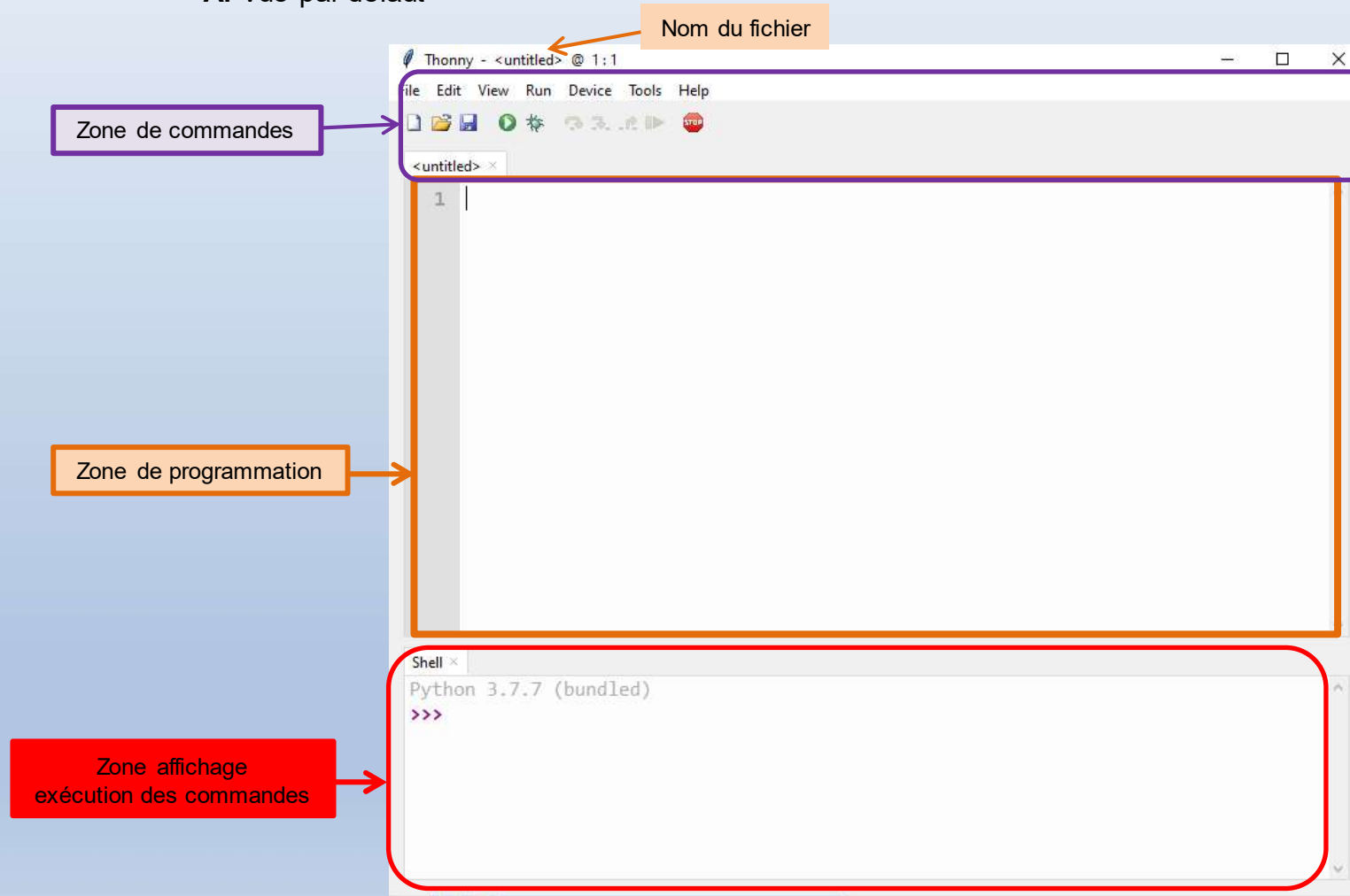
Choix Windows



IDE Thonny

5. Fenêtre de travail Thonny

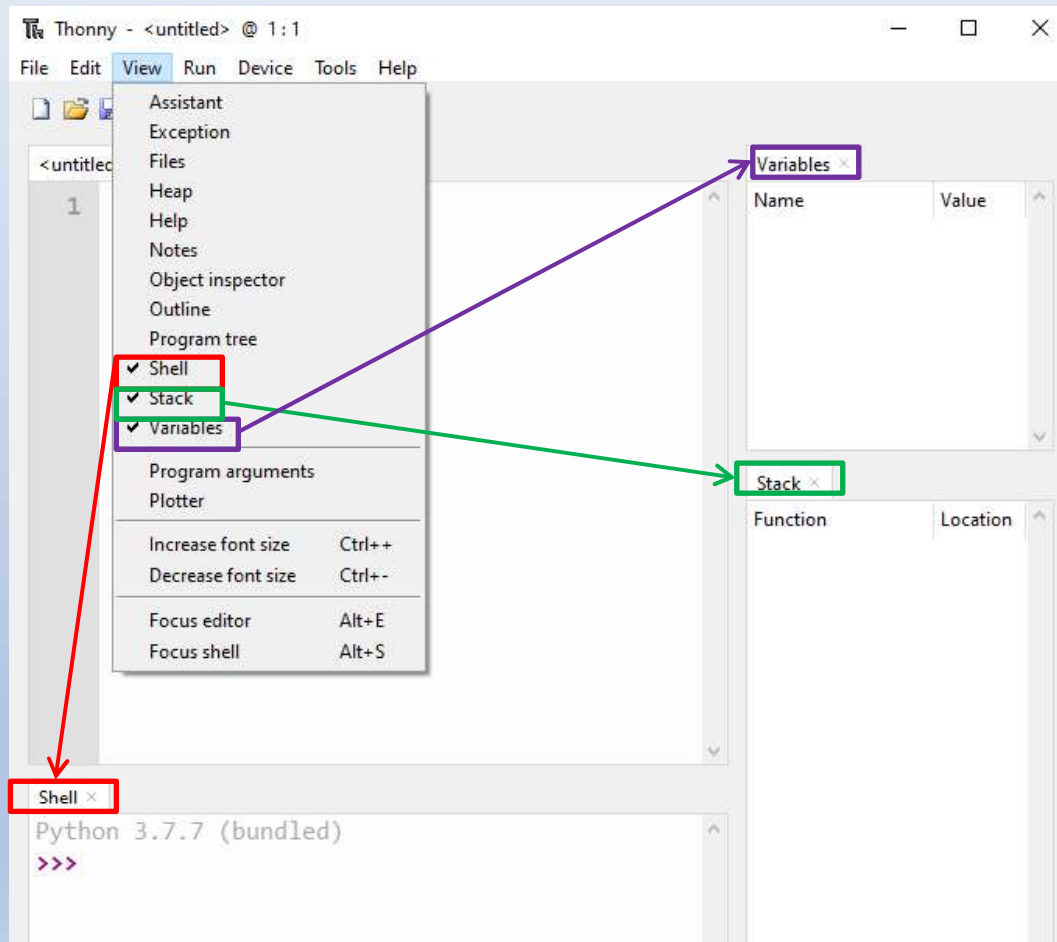
A: Vue par défaut



IDE Thonny

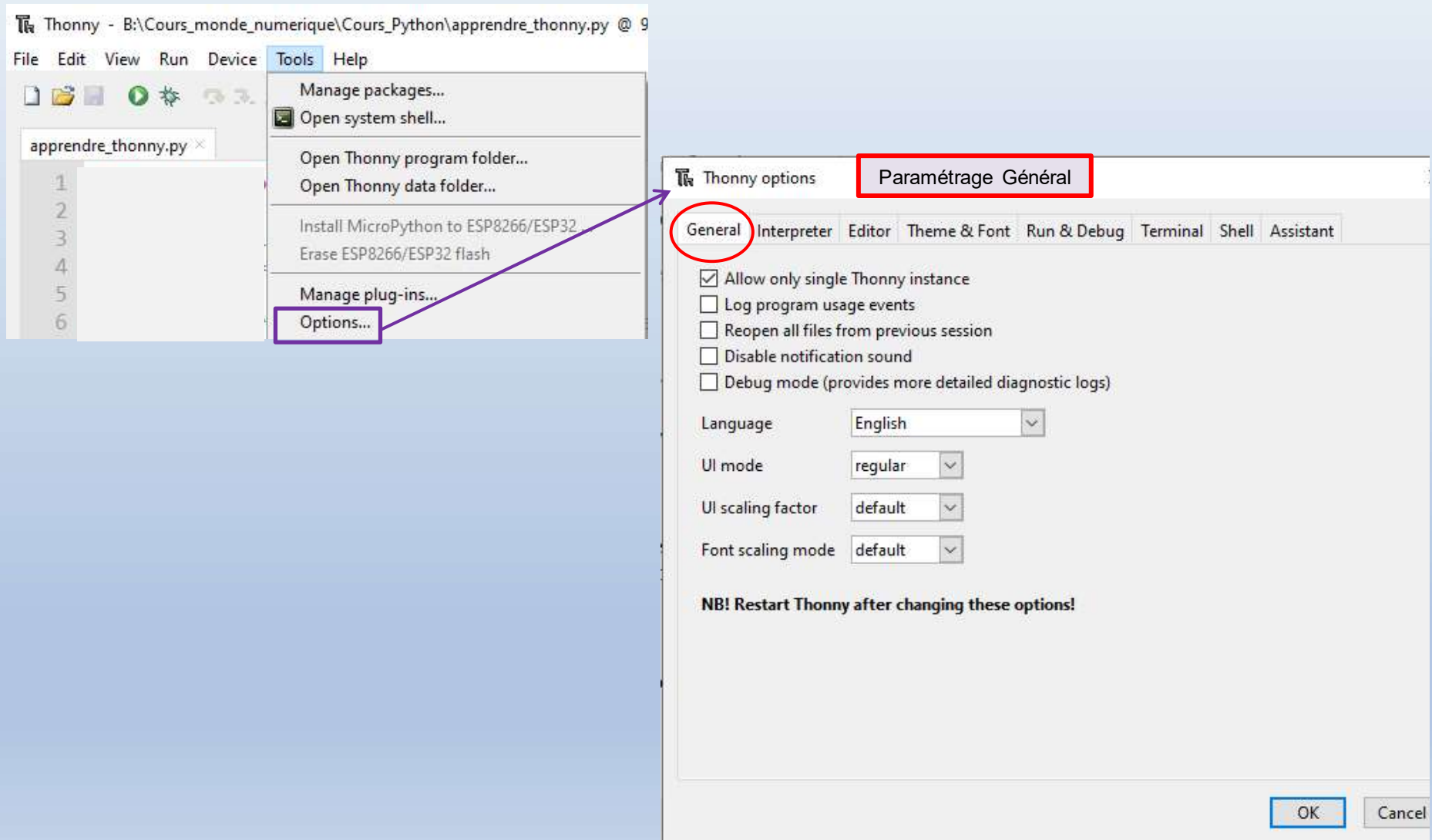
B: Vue personnalisée

Pour personnaliser la vue de travail, il faut cocher les items désirés

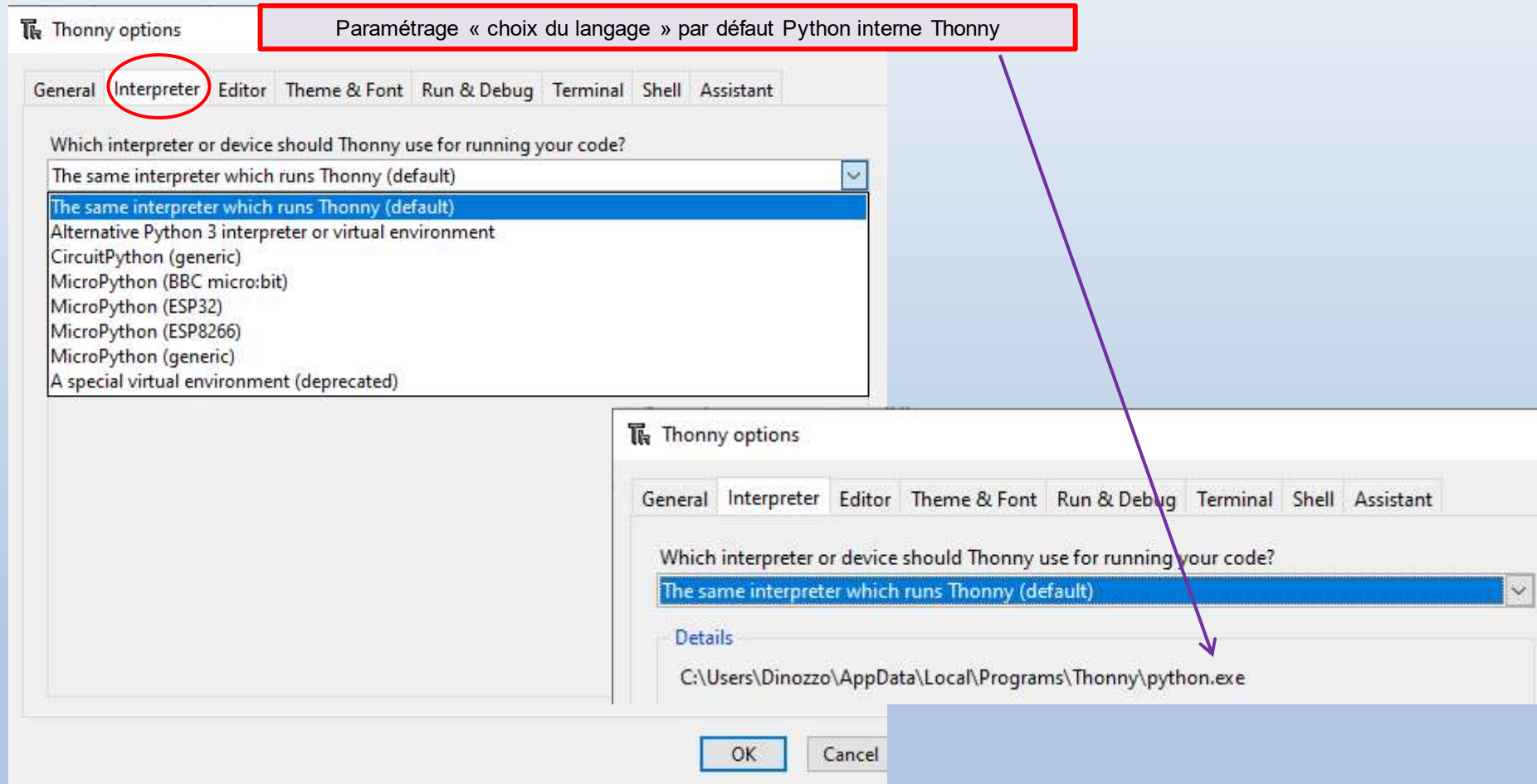


IDE Thonny

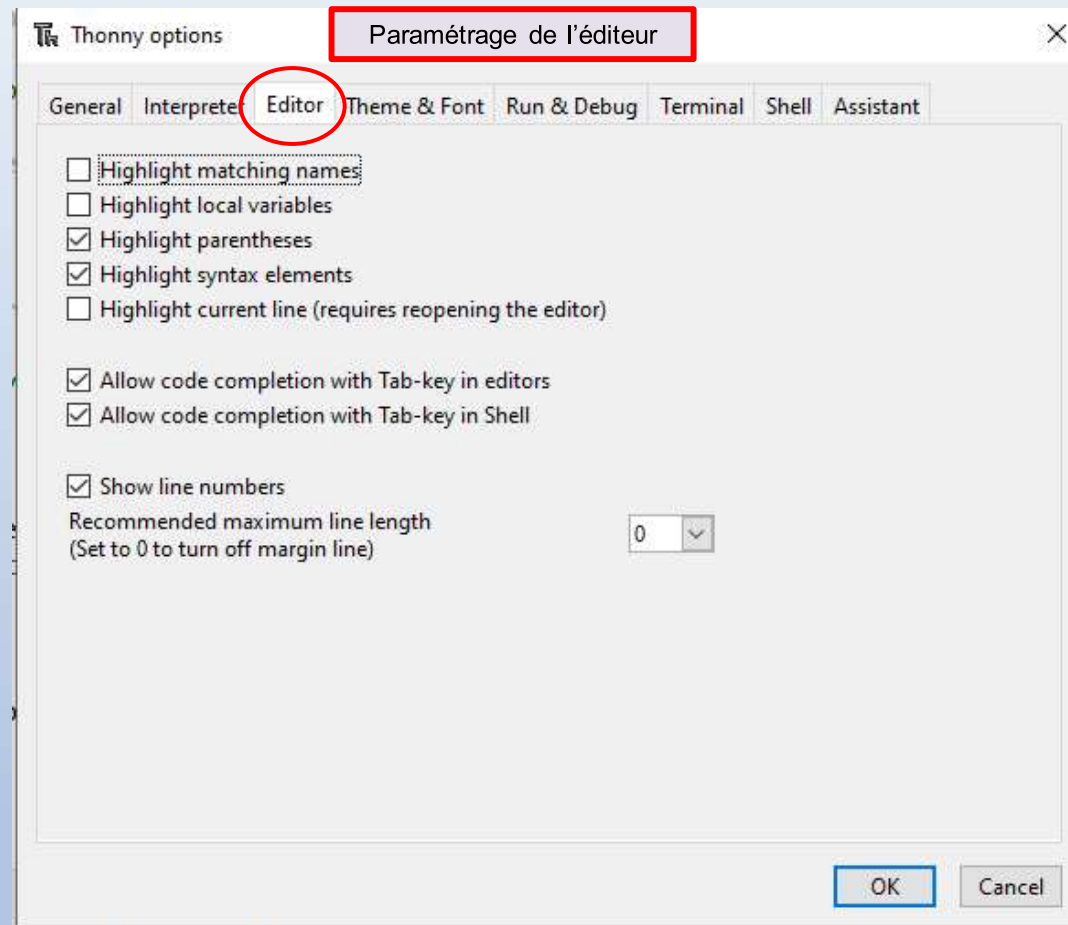
6. Paramétrage de l'IDE



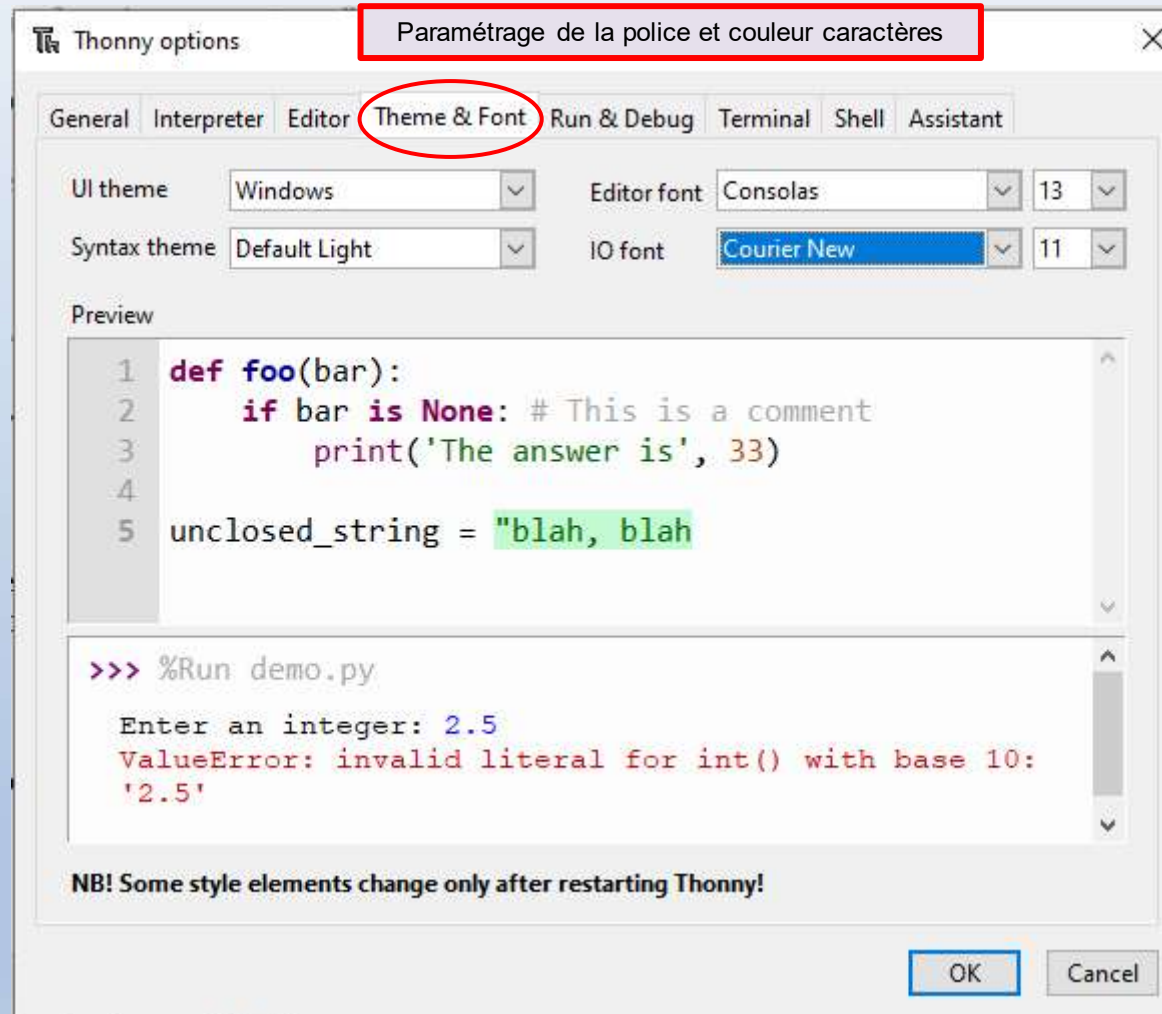
IDE Thonny



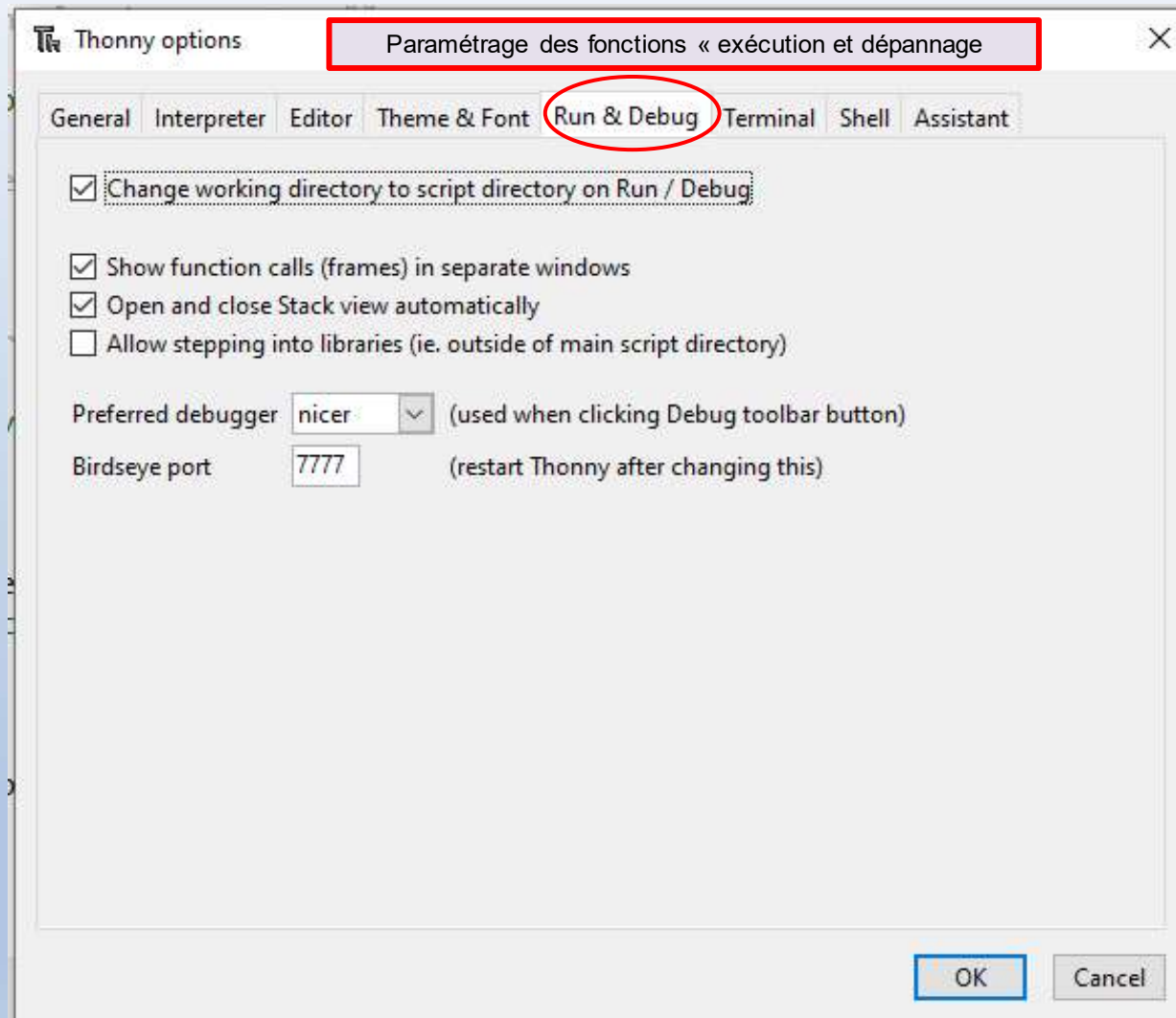
IDE Thonny



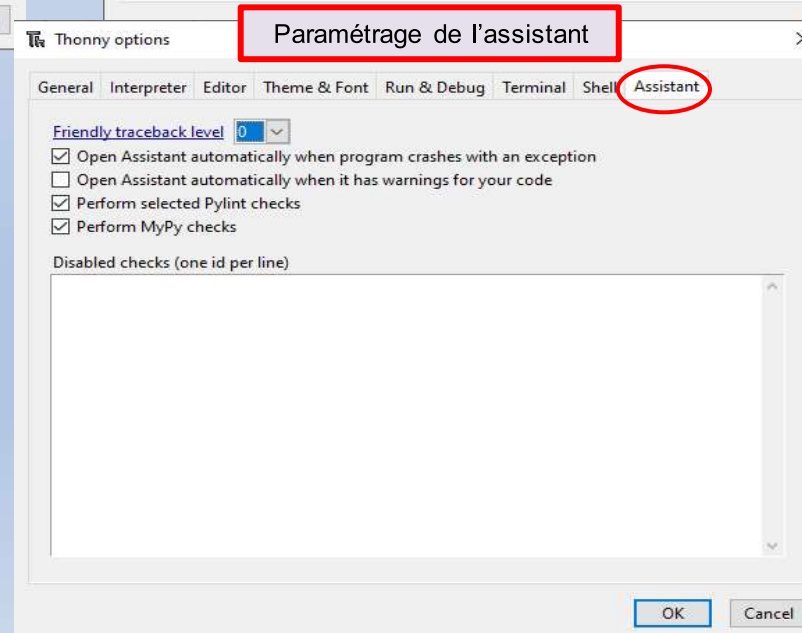
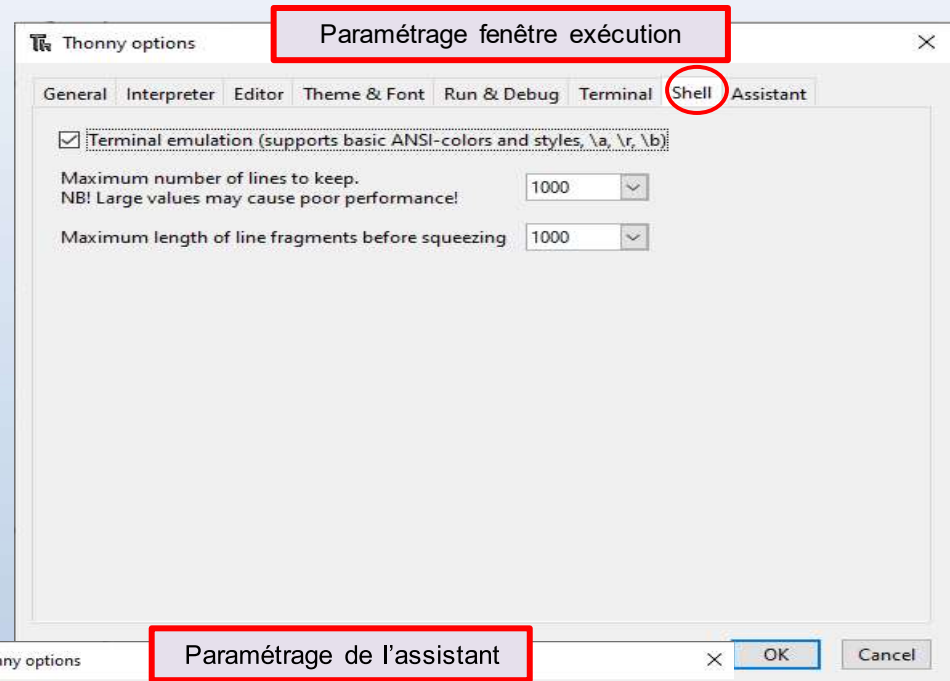
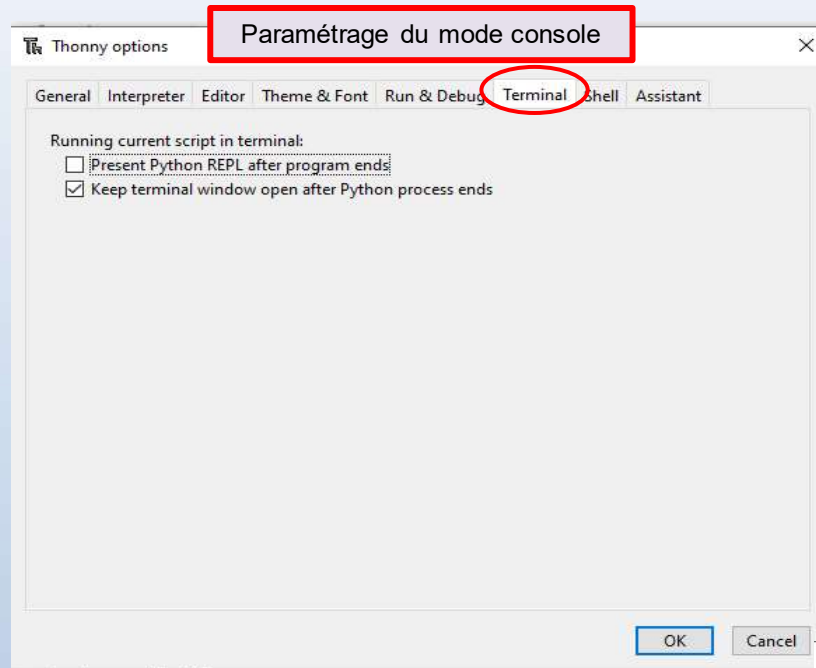
IDE Thonny



IDE Thonny



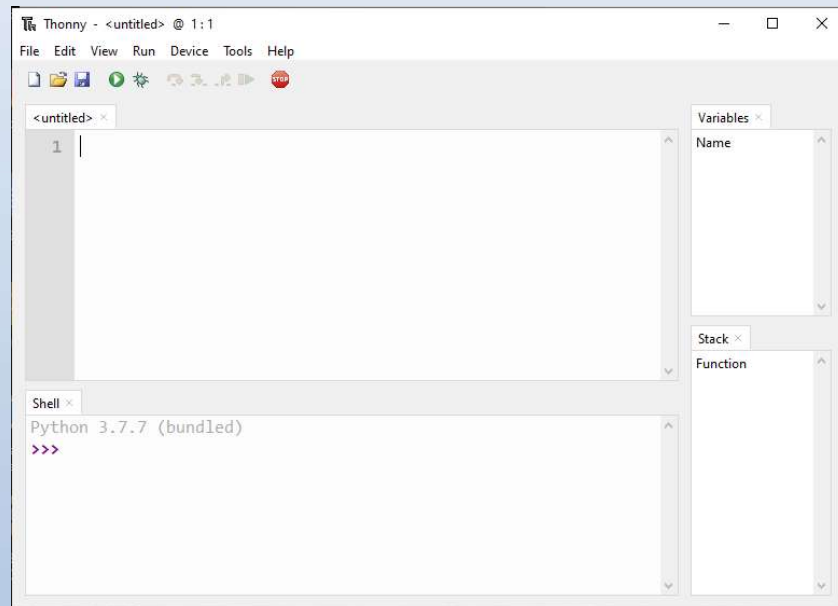
IDE Thonny



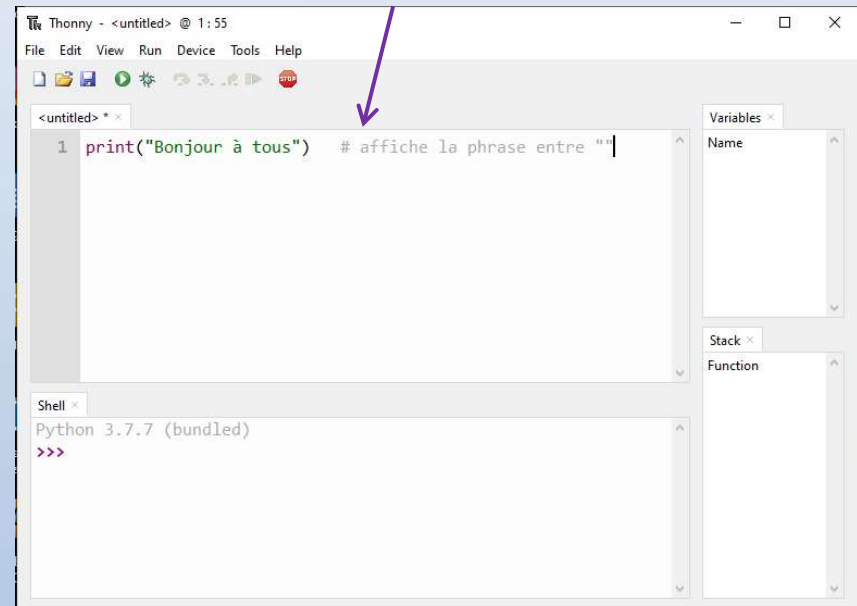
IDE Thonny

7. Prise en main

1. Ouverture de Thonny

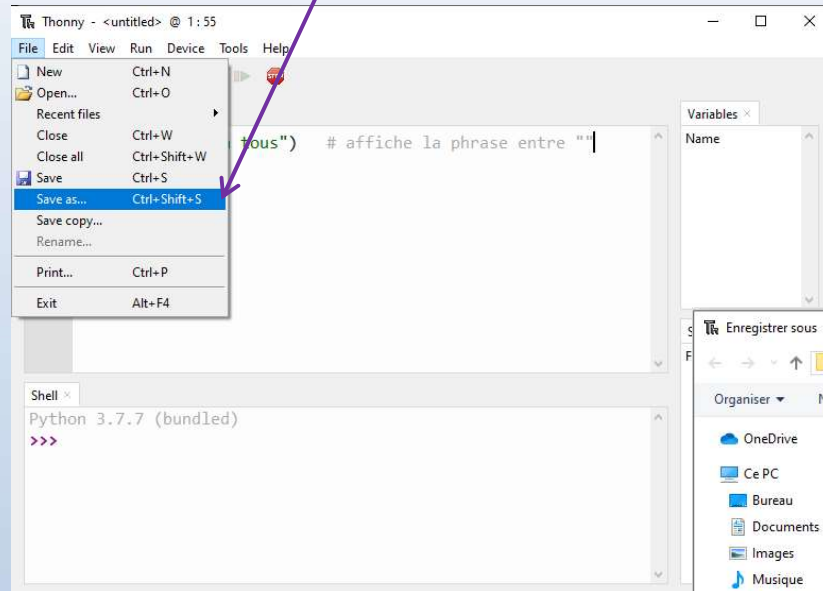


2. Ecriture d'une instruction

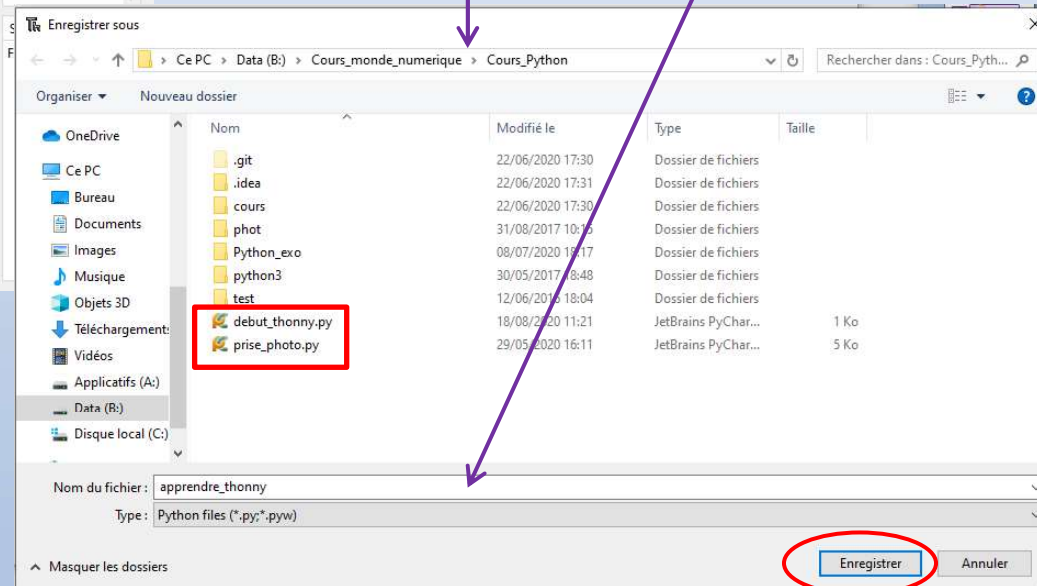


IDE Thonny

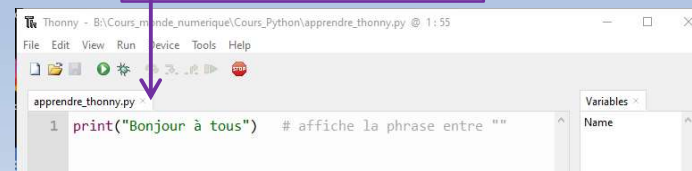
3. Sauvegarde du programme



4. Répertoire de sauvegarde et nom de programme .py

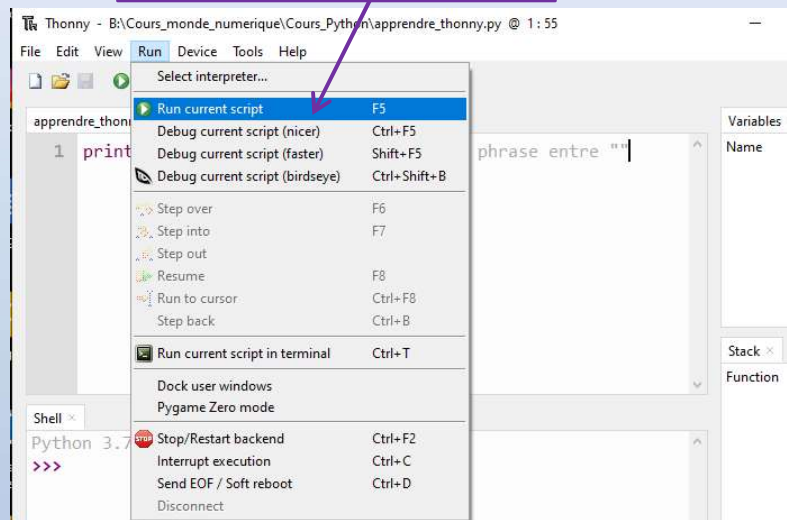


5. Prise en compte du nom

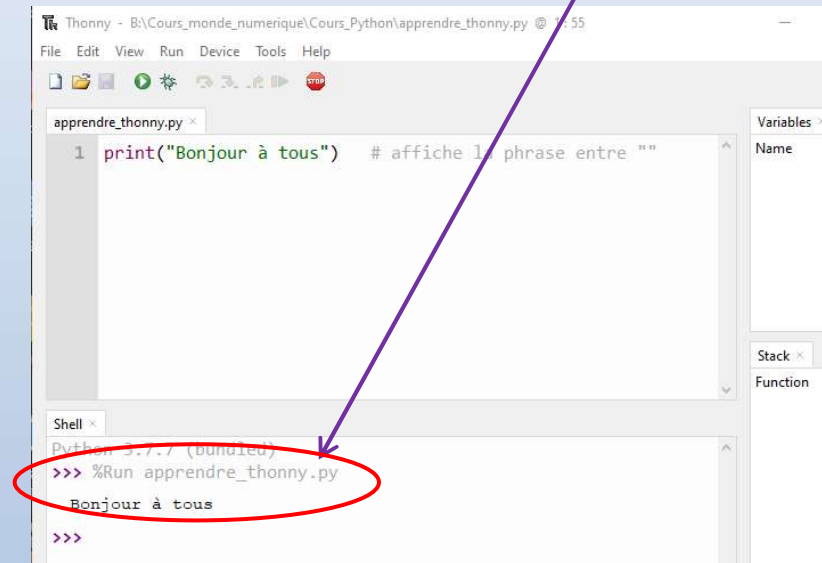


IDE Thonny

6. Exécution du programme



7. Affichage du résultat



IDE Thonny

8. Affichage d'erreurs de programmation

The screenshot shows the Thonny IDE interface. The main editor window displays a Python script named `apprendre_thonny.py` with the following code:

```
1 print("Bonjour à tous") # affiche la phrase entre ""
2 input("Taper un chiffre ", chiffre)
3 print("valeur double ", chiffre*2)
4
```

An orange callout box with the text "Ajout des 2 lignes d'instructions" points to lines 2 and 3 of the code.

The Shell window at the bottom shows the output of running the script:

```
>>> %Run apprendre_thonny.py
Bonjour à tous
Traceback (most recent call last):
  File "B:\Cours monde numerique\Cours Python\apprendre_thonny.py",
    line 2, in <module>
      input("Taper un chiffre ", chiffre)
NameError: name 'chiffre' is not defined
>>>
```

An orange callout box with the text "Erreur à l'exécution sur l'instruction 'input'" points to the `input` function call in the Shell output.

The Assistant window on the right is open, showing a `NameError: name 'chiffre' is not defined` message. The Assistant provides several suggestions to resolve the error:

- Did you actually mean string (text)?
If you didn't mean a variable but literal text "chiffre", then surround it with quotes.
- Did you forget to import it?
Some functions/variables need to be imported before they can be used.
- Has Python executed the definition?
Don't forget that name becomes defined when corresponding definition ('=', 'def' or 'import') gets executed. If the definition comes later in code or is inside an if-statement, Python may not have executed it (yet).

An orange callout box with the text "Lire assistant" points to the Assistant window.

IDE Thonny

9. Méthodologie de corrections (cas typage)

a: Suite aux indications modification instruction 'input'

b: Pas d'erreur mais résultat inattendu
Attendu $6*2=12$ affichage 6 6

c: L'assistant dit que le code est bon

d: vérifier le **type** de la variable

Variables

Name	Value
chiffre	'6'

```
1 print("Bonjour à tous") # affiche la phrase entre ""
2
3 chiffre = input("Taper un chiffre ")
4 print("valeur double ", chiffre*2)
5
```

Shell

```
Python 3.7.7 (bundled)
>>> %Run apprendre_thonny.py
Bonjour à tous
Taper un chiffre 6
valeur double 66
>>> |
```

e: Affichage variable '6' type caractère
et non 6 type entier
alors redéfinir en entier par l'instruction 'int'

Résultat correct

Variables

Name	Value
chiffre	6

```
1 print("Bonjour à tous") # affiche la phrase entre ""
2
3 chiffre = int(input("Taper un chiffre "))
4 print("valeur double ", chiffre*2)
5
```

Shell

```
valeur double 12
>>> %Run apprendre_thonny.py
Bonjour à tous
Taper un chiffre 6
valeur double 12
>>> |
```

IDE Thonny

9. Méthodologie de corrections (cas bibliothèque)

The screenshot shows the Thonny IDE interface. The main editor displays a Python script named `apprendre_thonny.py` with the following code:

```
1 print("Bonjour à tous") # affiche la phrase entre ""
2
3 chiffre = int(input("Taper un chiffre "))
4 print("valeur double ", chiffre*2)
5
6 # calcul de la superficie d'un cercle de rayon = chiffre
7
8 surface = chiffre^2 * pi
9 print("La surface du cercle de rayon {0} est de {1} ".format(chiffre, surface))
10
11
```

Annotations on the code:

- A bracket on lines 8 and 9 is labeled "Ajout des 2 lignes d'instructions".
- An arrow points from the text "Erreur à l'exécution sur l'instruction 'pi'" to the variable `pi` in line 8.

The right sidebar shows the "Variables" panel with `chiffre` set to 5, and the "Assistant" panel with the following error message:

NameError: name 'pi' is not defined
`apprendre_thonny.py`, line 8
Python doesn't know what `pi` stands for.

The assistant suggests the following options:

- ☒ Did you forget to import it?
- ☐ Did you actually mean `string` (text)?
- ☐ Has Python executed the definition?

An arrow points from the first suggestion to the text "a: Assistant import de pi".

The bottom panel shows the "Shell" with the following output:

```
valeur double 10
Traceback (most recent call last):
  File "B:\Cours monde numerique\Cours Python\apprendre_thonny.py", line 8, in <module>
    surface = chiffre^2 * pi
NameError: name 'pi' is not defined
>>>
```

IDE Thonny

9. Méthodologie de corrections (cas bibliothèque)

Ajout de la fonction 'pi' de la bibliothèque math

```
1 from math import pi
2
3 print("Bonjour à tous") # affiche la phrase entre ""
4
5 chiffre = int(input("Taper un chiffre "))
6 print("valeur double ", chiffre*2)
7
8 # calcul de la superficie d'un cercle de rayon = chiffre
9
10 surface = chiffre^2 * pi
11 print("La surface du cercle de rayon {0} est de {1} ".format(chiffre, surface))
12
13
```

Erreur à l'exécution sur l'instruction '^' opérateur qui n'est pas compatible avec les variables de type entier ou réel

TypeError: unsupported operand type(s) for '^': 'int' and 'float'

Python was asked to do an operation with an object which doesn't support it.

Did you expect another type? Maybe you forgot some details about this operation?

Was it helpful or confusing? General advice on dealing with errors.

Shell

```
valeur double 10
Traceback (most recent call last):
  File "B:\Cours_monde_numerique\Cours_Python\apprendre_thonny.py", line 10, in <module>
    surface = chiffre^2 * pi
TypeError: unsupported operand type(s) for '^': 'int' and 'float'
>>> |
```

Remplacement de '^' par '**'

```
1 from math import pi
2
3 print("Bonjour à tous") # affiche la phrase entre ""
4
5 chiffre = int(input("Taper un chiffre "))
6 print("valeur double ", chiffre*2)
7
8 # calcul de la superficie d'un cercle de rayon = chiffre
9
10 surface = chiffre**2 * pi
11 print("La surface du cercle de rayon {0} est de {1:8.2f} ".format(chiffre, surface))
12
13
```

Formate affichage à 2 décimales

Résultat correct

Shell

```
>>> %Run apprendre_thonny.py
Bonjour à tous
Taper un chiffre 5
valeur double 10
La surface du cercle de rayon 5 est de 78.54
>>> |
```

IDE Thonny

10. Exemple de programme

The screenshot displays the Thonny IDE interface with a Python program for calculating circle properties and converting temperatures. The code is as follows:

```
1 from math import pi
2
3 coeff_conversion = 9/5
4 k_conversion = 32
5
6 print("Bonjour à tous") # affiche la phrase entre ""
7
8 rayon = int(input("Entrez un rayon "))
9
10 # calcul de la circonférence du cercle
11 circonference = 2 * rayon * pi
12 print("La circonférence du cercle de rayon {0} est de {1:8.2f}".format(rayon, circonference))
13
14 # calcul de la superficie d'un cercle de rayon
15 surface = rayon**2 * pi
16 print("La surface du cercle de rayon {0} est de {1:8.2f}".format(rayon, surface))
17
18 # conversion Farenheit Celsius
19 val_far = float(input("Entrez une T° en Farenheit de -459.67 à 212 :"))
20 val_cel = float((val_far - k_conversion) / coeff_conversion)
21 print("Pour une T° de {0}Farenheit, la T° est de {1:8.2f}Celsius".format(val_far, val_cel))
22
23 #conversion T° Celsius Farenheit
24 val_cel = float(input("Entrez une T° en Celsius de -273.15 à 100 :"))
25 val_far = float(val_cel * coeff_conversion + k_conversion)
26 print("Pour une T° de {0}Celsius, la T° est de {1:8.2f}Farenheit".format(val_cel, val_far))
```

The Shell window shows the program's execution output:

```
Bonjour à tous
Entrez un rayon 5
La circonférence du cercle de rayon 5 est de 31.42
La surface du cercle de rayon 5 est de 78.54
Entrez une T° en Farenheit de -459.67 à 212 :100
Pour une T° de 100.0Farenheit, la T° est de 37.78Celsius
Entrez une T° en Celsius de -273.15 à 100 :37.5
Pour une T° de 37.5Celsius, la T° est de 99.50Farenheit
```

The Variables panel on the right shows the current state of the program:

Name	Value
circonference	31.4159265358979
coeff_conversion	1.8
k_conversion	32
pi	3.14159265358979
rayon	5
surface	78.5398163397448

The Assistant panel on the right provides feedback on the code:

The code in [apprendre_thonny.py](#) looks good.

If it is not working as it should, then consider using some [general debugging techniques](#).

[Was it helpful or confusing?](#)

Glossaire

Sigle	
IDE	I ntegrated D evelopment E nvironment
PC	P ersonal C omputer