



Programmation avec Python

Module 04 : Les outils de débogage



OBJECTIFS

Python : les outils de débogage
OBJECTIFS

Savoir lire une Traceback

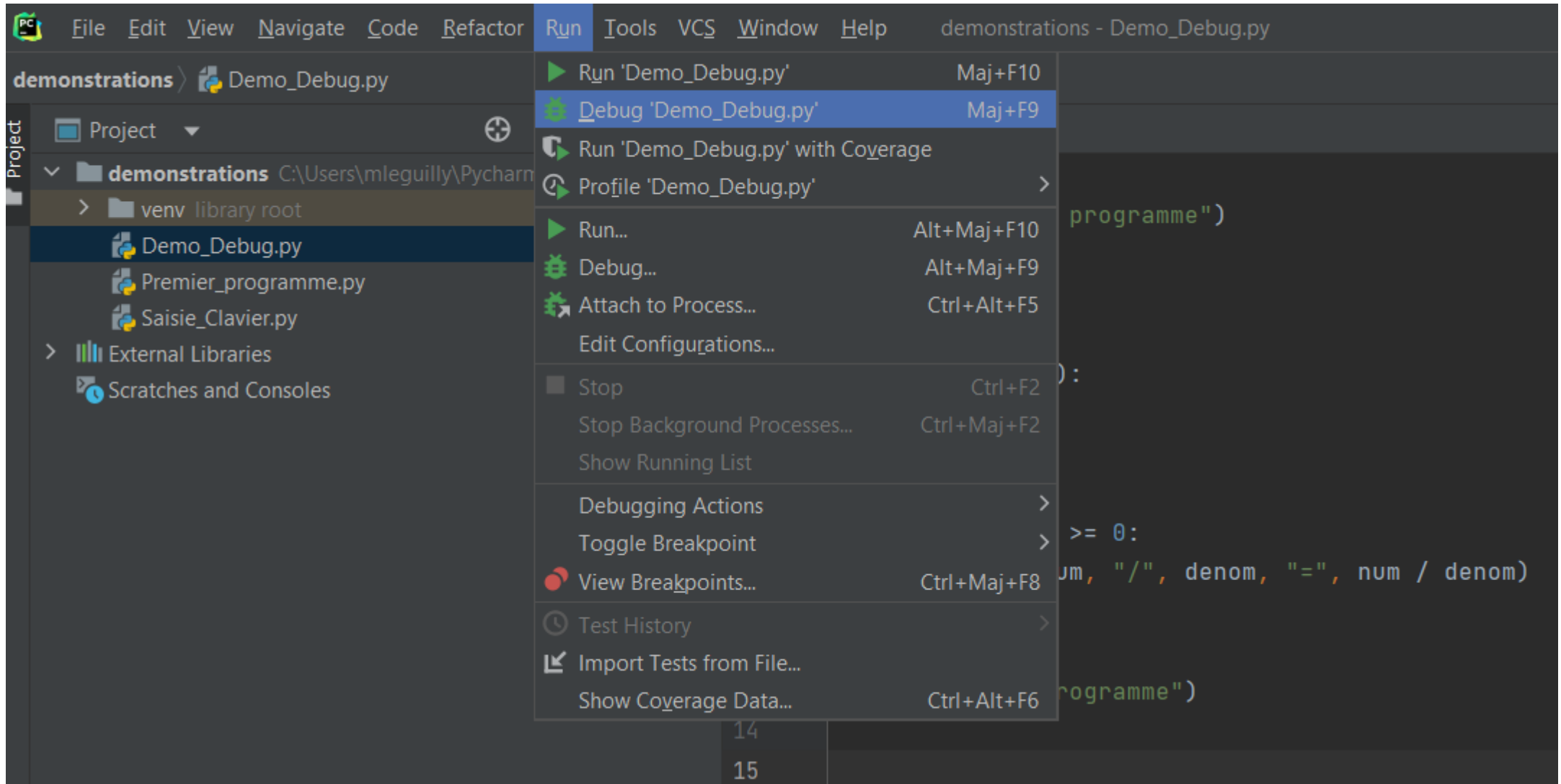
Découvrir le débogage avec
PyCharm

- Traceback (exemple) :

```
Traceback (most recent call last):  
File "Mon_Fichier.py", line 12, in ma_methode  
print(num, "÷", denom, "=", num / denom)  
ZeroDivisionError: division by zero
```

- Lignes :

1. Message indiquant la présence d'une Traceback
2. Localisation : Fichier, Ligne, Méthode
 - <stdin> /Fichier : correspond à l'interpréteur
 - <module> /Méthode : correspond au bloc principal du fichier
3. Extrait de l'instruction où l'erreur s'est produite
4. Le motif de l'interruption :
 - `AttributeError`, `ImportError`, `IndexError`, `KeyError`, `NameError`, `SyntaxError`, `TypeError`, `ValueError`, ..., **`ZeroDivisionError`**.



Python : les outils de débogage
LE DÉBOGEUR (PYCHARM)

The screenshot displays the PyCharm IDE interface with a Python file named `Demo_Debug.py` open. The code contains a function `ma_methode` that prints the division of `num` by `denom` in a loop. A red dot indicates a breakpoint set on line 9. The IDE is in a debug state, with the 'Debugger' tab active. The call stack on the left shows the current frame `ma_methode, Demo_Debug.py:10`. The console on the right shows the current values of the variables: `denom = {int} 9` and `num = {int} 16`.

```
1 # Affichage
2 print("début du programme")
3
4
5 1 usage
6 def ma_methode():
7     num = 16    num: 16
8     denom = 9   denom: 9
9     while denom >= 0:
10        print(num, "/", denom, "=", num / denom)
11        denom -= 1
12
13 ma_methode()
14 print("fin du programme")
15
```

Point d'arrêt ●

Debugger: Demo_Debug

Debugger Console

MainThread

ma_methode, Demo_Debug.py:10

<module>, Demo_Debug.py:13

Evaluate expression (Entrée) or add a watch (Ctrl+Maj+Entrée)

denom = {int} 9

num = {int} 16

Pile d'appels

Visualisation des variables

DÉMONSTRATION

UPDATE
Utiliser le débogueur

