

Couverture de code

Couverture de code

Définition

La couverture de code est un **indicateur** qui mesure le **pourcentage** de votre **code source exécuté** par vos **tests**

Voici comment on calcule ce taux de couverture :

(nombre d'instructions exécutées par le test / nombre total d'instructions) * 100

4 types de couverture de code

On recense **4 types de couverture** de code :

Statement Coverage = couverture des instructions

- On vérifie si chaque ligne du code a été exécutée au moins une fois

Branch Coverage = couverture des branches

- On vérifie si chaque branche d'une condition a été exécutée au moins une fois (if, else, else if ...)

Condition Coverage = couverture des conditions

- On vérifie si chaque condition booléenne a été évaluée au moins une fois à TRUE et à FALSE

Path coverage = couverture des chemins

- On vérifie si tous les chemins possibles dans le programme ont été testés (toutes les séquences possibles de décisions)

4 types de couverture de code

Attention :

- tous les outils de couverture de code ne testent pas ces 4 types de couvertures, à vous de choisir l'outil qui correspond à votre besoin dans le langage utilisé

Par exemple, le module **coverage.py** utilisé pour tester la couverture de code en Python ne traite que deux types de couverture :

- *Statement Coverage* : pris en charge par défaut
- *Branch Coverage* : en option (il faut ajouter l'option - *-branch* dans la ligne de commande)
- Les types *Path coverage* et *Condition coverage* ne sont pas pris en charge

Exemple avec Python

Contexte

- vous avez développé une application en Python
- vous avez écrit des tests en Python avec le Framework *pytest*, permettant de mettre en place des tests unitaires et fonctionnels

Objectif

- mesurer quelles sont les parties du code développé en Python qui sont exécutées lors des tests

1 – Pour installer le module *coverage* :

pip install coverage

2 – Pour exécuter des tests écrits avec le Framework pytest en **mesurant la couverture**

coverage run -m pytest

```
C:\Formation\Python\PY>coverage run -m pytest
===== test session starts =====
platform win32 -- Python 3.11.4, pytest-8.2.0, pluggy-1.5.0
rootdir: C:\Formation\Python\PY
plugins: html-2.0.1, metadata-3.1.1, mock-3.14.0, ordering-0.6, rerunfailures-14.0, xdist-3.6.1, seleniumbase-4.26.3
collected 24 items

S1E1_test.py . [ 4%]
S1E2_test.py . [ 8%]
S1E3_test.py . [ 12%]
S1E4_test.py . [ 16%]
S1E5_test.py . [ 20%]
S1E6_test.py . [ 25%]
S1E7_test.py . [ 29%]
S1E8_test.py ..... [ 54%]
S1E9_test.py ..... [ 83%]
S2E1_V1_test.py . [ 87%]
S2E2V1_test.py . [ 91%]
S2E4_V1_test.py . [ 95%]
Synthèse_1_test.py . [100%]

===== 24 passed in 0.29s =====
```

3 – Pour générer un rapport global de couverture :

coverage report -m

```
C:\Formation\Python\PY>coverage report -m
Name                               Stmts  Miss  Cover   Missing
-----
S1E1.py                             11     0   100%
S1E1_test.py                       12     0   100%
S1E2_test.py                       10     0   100%
S1E3.py                             12     4    67%   5, 7, 9, 12
S1E3_test.py                       10     0   100%
S1E4.py                             22     7    68%  11-12, 16-21
S1E4_test.py                       12     0   100%
S1E5_test.py                       15     2    87%   11-12
S1E6_test.py                       14     0   100%
S1E7_test.py                       14     0   100%
S1E8_test.py                       40     1    98%   54
S1E9_test.py                       49     1    98%   68
S2E1_V1_test.py                    21     1    95%   41
S2E2V1_test.py                     22     1    95%   43
S2E4_V1_test.py                    14     1    93%   20
Synthèse_1_test.py                 22     1    95%   30
-----
TOTAL                             300    19    94%
```

Le rapport indique :

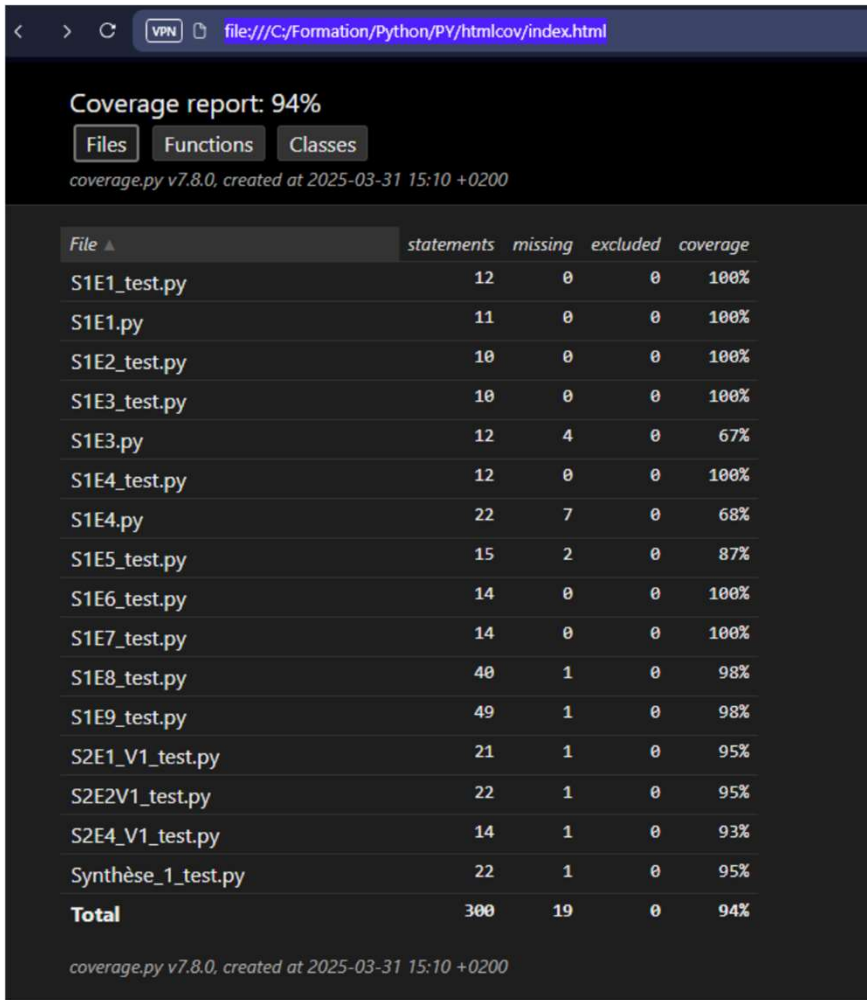
- les taux de couverture par source
- les lignes non couvertes par les tests
- le taux de couverture global

Pour générer un rapport pour un seul programme source :

coverage report -m nom_du_source.py

4 – Pour générer un rapport au format html : *coverage html*

- vous pouvez ensuite consulter le rapport dans votre navigateur (en exécutant par exemple la commande : `start htmlcov\index.html`)
- le rapport présente les tests et les couvertures correspondantes



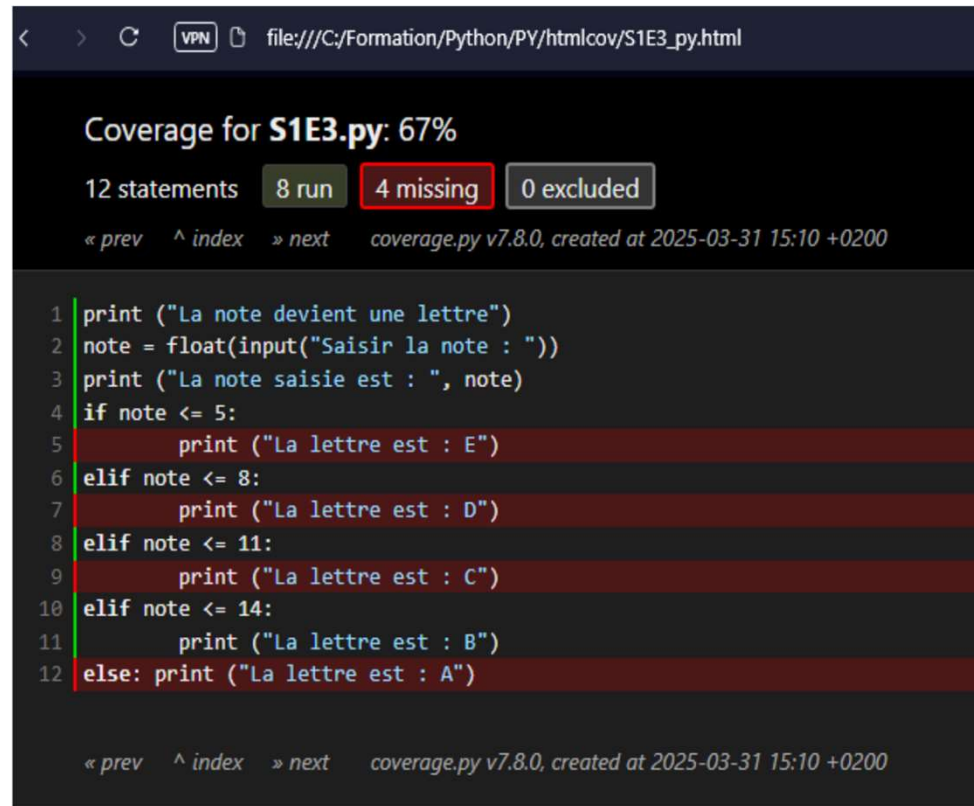
Coverage report: 94%

Files Functions Classes

coverage.py v7.8.0, created at 2025-03-31 15:10 +0200

File ▲	statements	missing	excluded	coverage
S1E1_test.py	12	0	0	100%
S1E1.py	11	0	0	100%
S1E2_test.py	10	0	0	100%
S1E3_test.py	10	0	0	100%
S1E3.py	12	4	0	67%
S1E4_test.py	12	0	0	100%
S1E4.py	22	7	0	68%
S1E5_test.py	15	2	0	87%
S1E6_test.py	14	0	0	100%
S1E7_test.py	14	0	0	100%
S1E8_test.py	40	1	0	98%
S1E9_test.py	49	1	0	98%
S2E1_V1_test.py	21	1	0	95%
S2E2V1_test.py	22	1	0	95%
S2E4_V1_test.py	14	1	0	93%
Synthèse_1_test.py	22	1	0	95%
Total	300	19	0	94%

coverage.py v7.8.0, created at 2025-03-31 15:10 +0200



Coverage for S1E3.py: 67%

12 statements 8 run 4 missing 0 excluded

« prev ^ index » next coverage.py v7.8.0, created at 2025-03-31 15:10 +0200

```
1 print ("La note devient une lettre")
2 note = float(input("Saisir la note : "))
3 print ("La note saisie est : ", note)
4 if note <= 5:
5     print ("La lettre est : E")
6 elif note <= 8:
7     print ("La lettre est : D")
8 elif note <= 11:
9     print ("La lettre est : C")
10 elif note <= 14:
11     print ("La lettre est : B")
12 else: print ("La lettre est : A")
```

« prev ^ index » next coverage.py v7.8.0, created at 2025-03-31 15:10 +0200

Voici le test écrit avec pytest

```
S1E3_test.py X
S1E3_test.py > ...
1  # Test de S1E3.py
2  import pytest
3  from io import StringIO
4  import sys
5  from unittest.mock import patch
6
7  def test_conversion_note():
8      user_input = ["12"] # saisie utilisateur
9
10     with patch('builtins.input', side_effect=user_input), patch('sys.stdout', new_callable=StringIO) as mock_stdout:
11         import S1E3
12
13         output = mock_stdout.getvalue()
14         assert "La lettre est : B" in output
15
16 # pour lancer le test --> pytest S1E3_test.py
```

Voici la couverture obtenue

file:///C:/Formation/Python/PY/htmlcov/S1E3_py.html

Coverage for **S1E3.py**: 67%

12 statements 8 run 4 missing 0 excluded

« prev ^ index » next coverage.py v7.8.0, created at 2025-03-31 15:10 +0200

```
1 print ("La note devient une lettre")
2 note = float(input("Saisir la note : "))
3 print ("La note saisie est : ", note)
4 if note <= 5:
5     print ("La lettre est : E")
6 elif note <= 8:
7     print ("La lettre est : D")
8 elif note <= 11:
9     print ("La lettre est : C")
10 elif note <= 14:
11     print ("La lettre est : B")
12 else: print ("La lettre est : A")
```

« prev ^ index » next coverage.py v7.8.0, created at 2025-03-31 15:10 +0200