








TRABAJO FINAL

Objetivo: Crear un flujo de integración continua para un proyecto Python utilizando GitHub Actions que permita ejecutar pruebas automatizadas en el código de manera continua y

Repositorio a utilizar

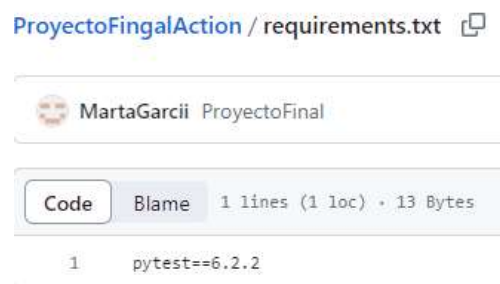
Creamos el repositorio con los ficheros de python

 MartaGarcii	Create main.yml	● 0960fac now	🕒 4 commits
 .github/workflows	Create main.yml		now
 my_python_code.py	ProyectoFinal		4 minutes ago
 requirements.txt	ProyectoFinal		4 minutes ago
 test_multiplicar.py	ProyectoFinal		4 minutes ago
 test_restar.py	ProyectoFinal		4 minutes ago
 test_sumar.py	ProyectoFinal		4 minutes ago

Instalando dependencias

Instalar las dependencias necesarias del proyecto Python utilizando pip y un archivo requirements.txt.

Y en el fichero requerimiento tenemos:



En el que se especifica los requerimientos del programa.

En ProyectoFingalAction/.github/workflows/main.yml tenemos la descarga de las dependencias:

```
- name: 'Install dependencies'
  run: |
    python -m pip install --upgrade pip
    pip install -r requirements.txt
    pip install pytest
```

Por lo tanto se nos queda el archivo main.yml:

ProyectoFingalAction / .github / workflows / ii

Edit Preview

```
1 name: 'Python Unit Tests'
2 on:
3   - push
4   - pull_request
5
6 jobs:
7   build:
8     runs-on: ubuntu-latest
9
10    steps:
11      - uses: actions/checkout@v2
12
13      - name: 'Set up Python'
14        uses: actions/setup-python@v2
15        with:
16          python-version: '3.9'
17
18      - name: 'Install dependencies'
19        run: |
20          python -m pip install --upgrade pip
21          pip install -r requirements.txt
22          pip install pytest
23
24      - name: 'Run tests'
25        run: |
26          pytest .
```

Pruebas automatizadas

Ejecutar pruebas automatizadas utilizando pytest

Para ejecutar las pruebas del módulo utilizamos pytest y con esto obtenemos 3 ficheros test_multiplicar.py test_restar.py y test_sumar.py ta que:

```
from my_python_code import multiplicar
def test_multiplicar():
    assert multiplicar(2, 3) == 6
    assert multiplicar(0, 0) == 0
    assert multiplicar(-1, 1) == -1
```

```
from my_python_code import restar
def test_restar():
    assert restar(5, 2) == 3
    assert restar(0, 0) == 0
    assert restar(-1, 1) == -2
```

```
from my_python_code import sumar
def test_sumar():
    assert sumar(2, 3) == 5
    assert sumar(0, 0) == 0
    assert sumar(-1, 1) == 0
```

Ejecutamos

Como podemos ver en la imagen inferior a funcionado correctamente:

Actions

Projects

Wiki

Security

Insights

Settings

All workflows

Showing runs from all workflows

Filter workflow runs

2 workflow runs

Event

Status

Branch

Actor

✓ Create main.yml

Python Unit Tests #2: Commit 0960fac pushed by MartaGarcii

main

1 minute ago

20s

...

✓ Create main.yml

Python Unit Tests #1: Commit 0960fac pushed by MartaGarcii

main

1 minute ago

23s

...

Si nos metemos dentro tenemos:

Summary

Jobs

build

Run details

Usage

Workflow file

build

succeeded 1 minute ago in 10s

> Set up job

> Run actions/checkout@v2

> Set up Python

> Install dependencies

> Run tests

> Post Set up Python

> Post Run actions/checkout@v2

> Complete job

Conclusiones

Con esta actividad hemos terminado de asimilar lo que es GitHub Actions