## **Mocking y Stubbing**

El mocking y stubbing permiten aislar la unidad de código que estás probando de sus dependencias.

* **Mock:** Un objeto simulado que reemplaza a una dependencia real.
* **Stub:** Un objeto simulado con un comportamiento predefinido.

**Herramienta:**

* **unittest.mock:** Módulo estándar de Python para crear mocks y stubs.

**Ejemplo:**

Python

import unittest from unittest.mock import patch class MyClass: def \_\_init\_\_(self, external\_service): self.external\_service = external\_service def call\_external\_service(self): return self.external\_service.get\_data() class TestMyClass(unittest.TestCase): @patch('my\_module.MyClass.external\_service') def test\_call\_external\_service(self, mock\_external\_service): mock\_external\_service.get\_data.return\_value = "data" my\_class = MyClass(mock\_external\_service) result = my\_class.call\_external\_service() assert result == "data"

Use code [with caution.](file:///C:/faq#coding)

**Puntos clave:**

* **Patch:** Se usa para reemplazar la dependencia external\_service con un mock.
* **Return value:** Se establece el valor de retorno del método get\_data del mock.