### Refactorización de

A menudo, llegarás a un punto en el que tu código funcionará, pero reconocerás que podrías mejorarlo dividiéndolo en una serie de funciones que tengan tareas específicas. Este proceso se llama *refactoring*. La refactorización hace que tu código sea más limpio, más fácil de entender y más fácil de ampliar.

Podemos refactorizar *remember\_me.py* trasladando la mayor parte de su lógica a una o más funciones. El objetivo de *remember\_me.py* es saludar al usuario, así que vamos a trasladar todo nuestro código existente a una función llamada greet\_user():

**remember\_me.py**

from pathlib import Path  
import json  
  
def greet\_user():  
❶ """Greet the user by name."""  
 path = Path('username.json')  
 if path.exists():  
 contents = path.read\_text()  
 username = json.loads(contents)  
 print(f"Welcome back, {username}!")  
 else:  
 username = input("What is your name? ")  
 contents = json.dumps(username)  
 path.write\_text(contents)  
 print(f"We'll remember you when you come back, {username}!")  
  
greet\_user()

Como ahora utilizamos una función, reescribimos los comentarios como un docstring que refleje cómo funciona actualmente el programa ❶. Este archivo es un poco más limpio, pero la función greet\_user() hace algo más que saludar al usuario: también recupera un nombre de usuario almacenado si existe y solicita un nuevo nombre de usuario si no existe.

Vamos a refactorizar greet\_user() para que no realice tantas tareas diferentes. Empezaremos moviendo el código para recuperar un nombre de usuario almacenado a una función separada:

from pathlib import Path  
import json  
  
def get\_stored\_username(path):  
❶ """Get stored username if available."""  
 if path.exists():  
 contents = path.read\_text()  
 username = json.loads(contents)  
 return username  
 else:  
❷ return None  
  
def greet\_user():  
 """Greet the user by name."""  
 path = Path('username.json')  
 username = get\_stored\_username(path)  
❸ if username:  
 print(f"Welcome back, {username}!")  
 else:  
 username = input("What is your name? ")  
 contents = json.dumps(username)  
 path.write\_text(contents)  
 print(f"We'll remember you when you come back, {username}!")  
  
greet\_user()

La nueva función get\_stored\_username() ❶ tiene un propósito claro, como se indica en el docstring. Esta función recupera un nombre de usuario almacenado y lo devuelve si lo encuentra. Si la ruta que se pasa a get\_stored\_username() no existe, la función devuelve None ❷. Esto es una buena práctica: una función debe devolver el valor que esperas o debe devolver None. Esto nos permite realizar una sencilla prueba con el valor de retorno de la función. Imprimimos un mensaje de bienvenida al usuario si el intento de recuperar un nombre de usuario tiene éxito ❸, y si no lo tiene, le pedimos un nuevo nombre de usuario.

Deberíamos eliminar un bloque más de código de greet\_user(). Si el nombre de usuario no existe, deberíamos mover el código que solicita un nuevo nombre de usuario a una función dedicada a ese fin:

from pathlib import Path  
import json  
  
def get\_stored\_username(path):  
 """Get stored username if available."""  
 --snip--  
  
def get\_new\_username(path):  
 """Prompt for a new username."""  
 username = input("What is your name? ")  
 contents = json.dumps(username)  
 path.write\_text(contents)  
 return username  
  
def greet\_user():  
 """Greet the user by name."""  
 path = Path('username.json')  
❶ username = get\_stored\_username(path)  
 if username:  
 print(f"Welcome back, {username}!")  
 else:  
❷ username = get\_new\_username(path)  
 print(f"We'll remember you when you come back, {username}!")  
  
greet\_user()

Cada función de esta versión final de *remember\_me.py* tiene un único y claro propósito. Llamamos a greet\_user(), y esa función imprime un mensaje apropiado: o bien da la bienvenida a un usuario existente o bien saluda a un nuevo usuario. Para ello, llama a get\_stored\_username() ❶, que sólo se encarga de recuperar un nombre de usuario almacenado, si existe. Por último, si es necesario, greet\_user() llama a get\_new\_username()❷, que sólo se encarga de obtener un nuevo nombre de usuario y almacenarlo. Esta compartimentación del trabajo es una parte esencial para escribir un código claro que sea fácil de mantener y ampliar.

[anterior](c10_34.html)[Subtema 35 de 37: (Ver todo)](c10.html)[siguiente](c10_36.html)