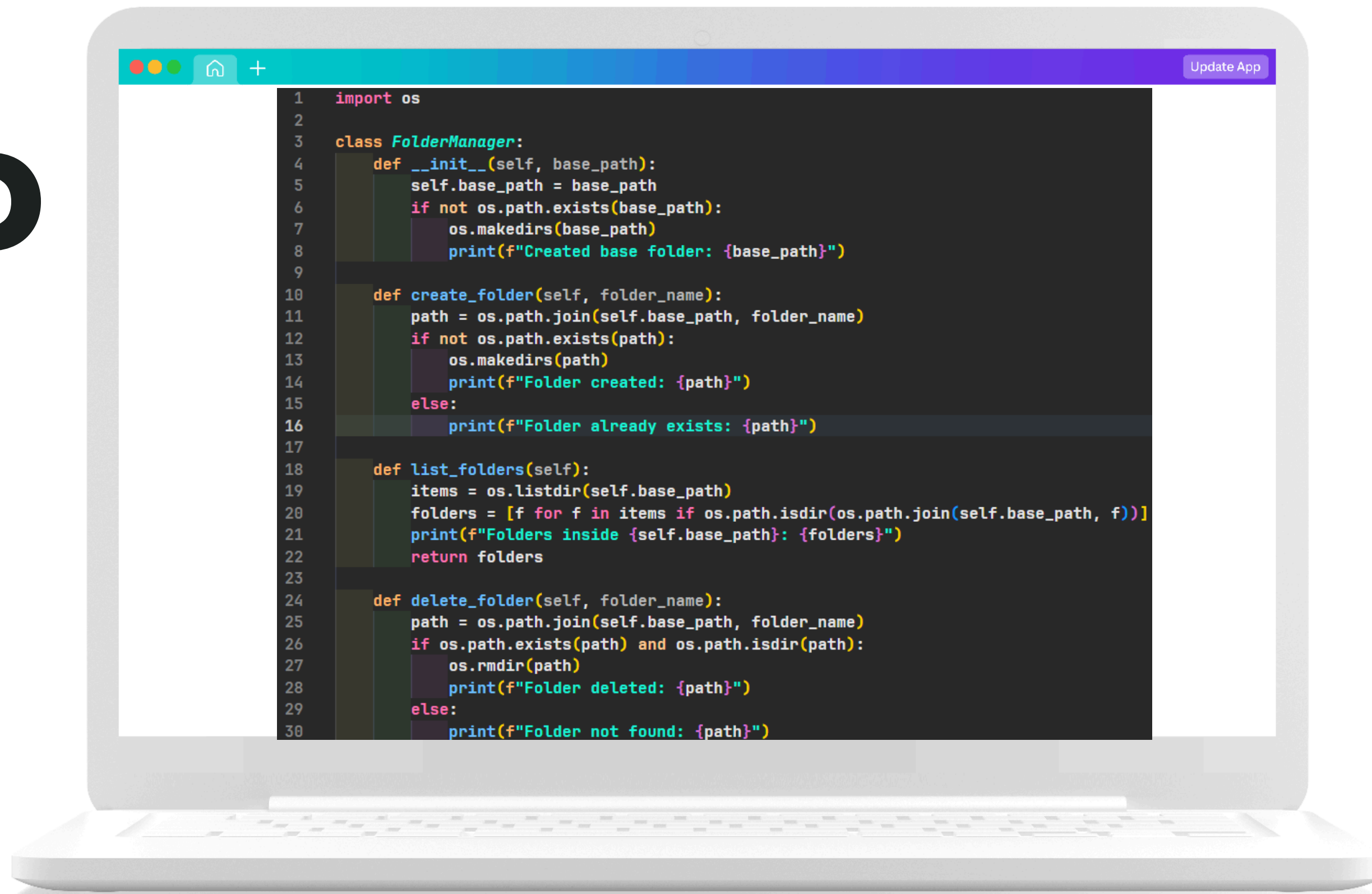


SOLUCIÓN

EJERICICIO FOLDER MANAGER

Código

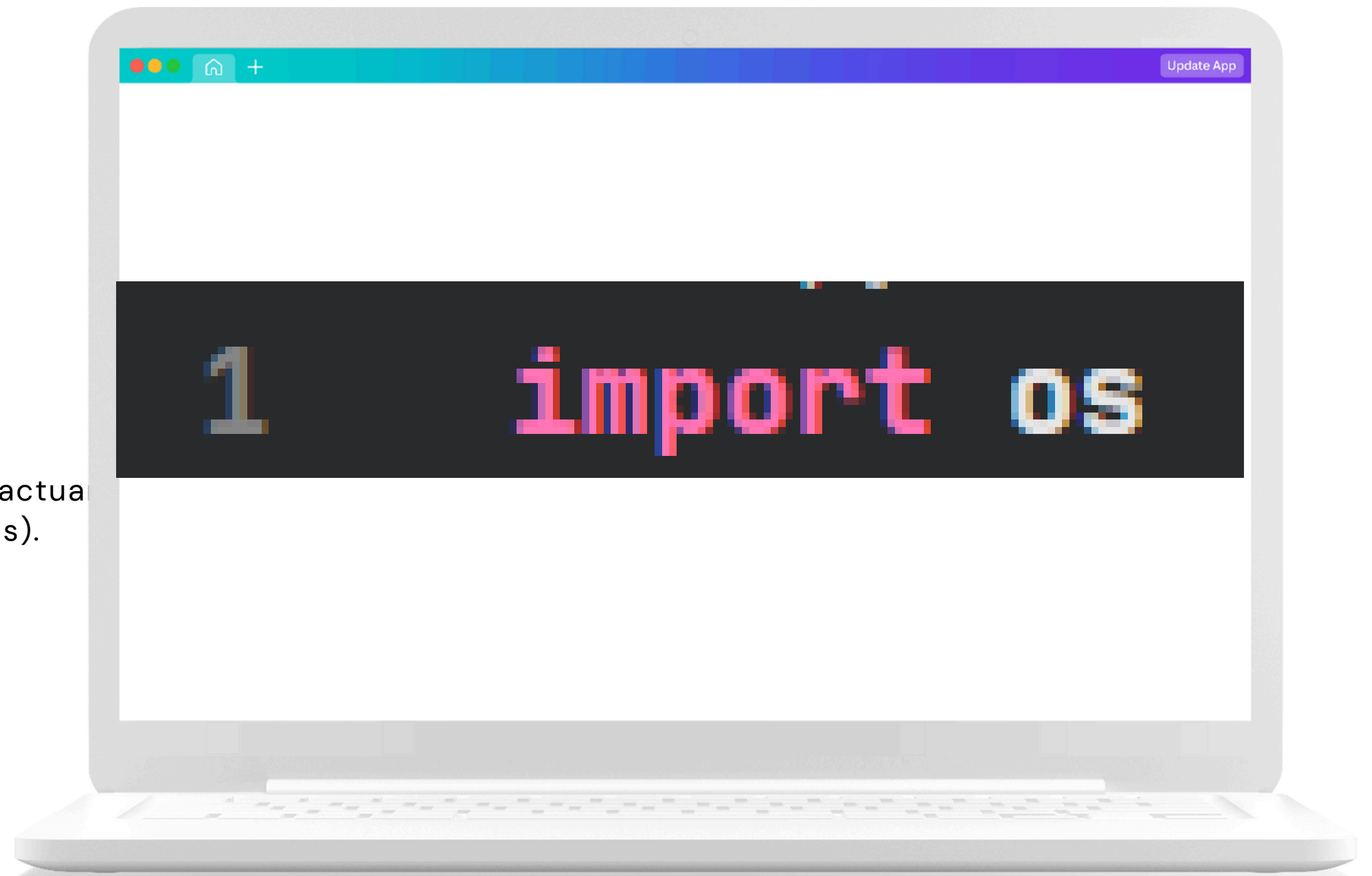
Class FolderManager



Imports

```
import os
```

Importa el módulo estándar os, que permite interactuar con el sistema operativo (rutas, carpetas, archivos).



Qué es una clase?

Una clase en Python es un molde que define estados (atributos) y comportamientos (métodos) para los objetos que creamos a partir de ella.

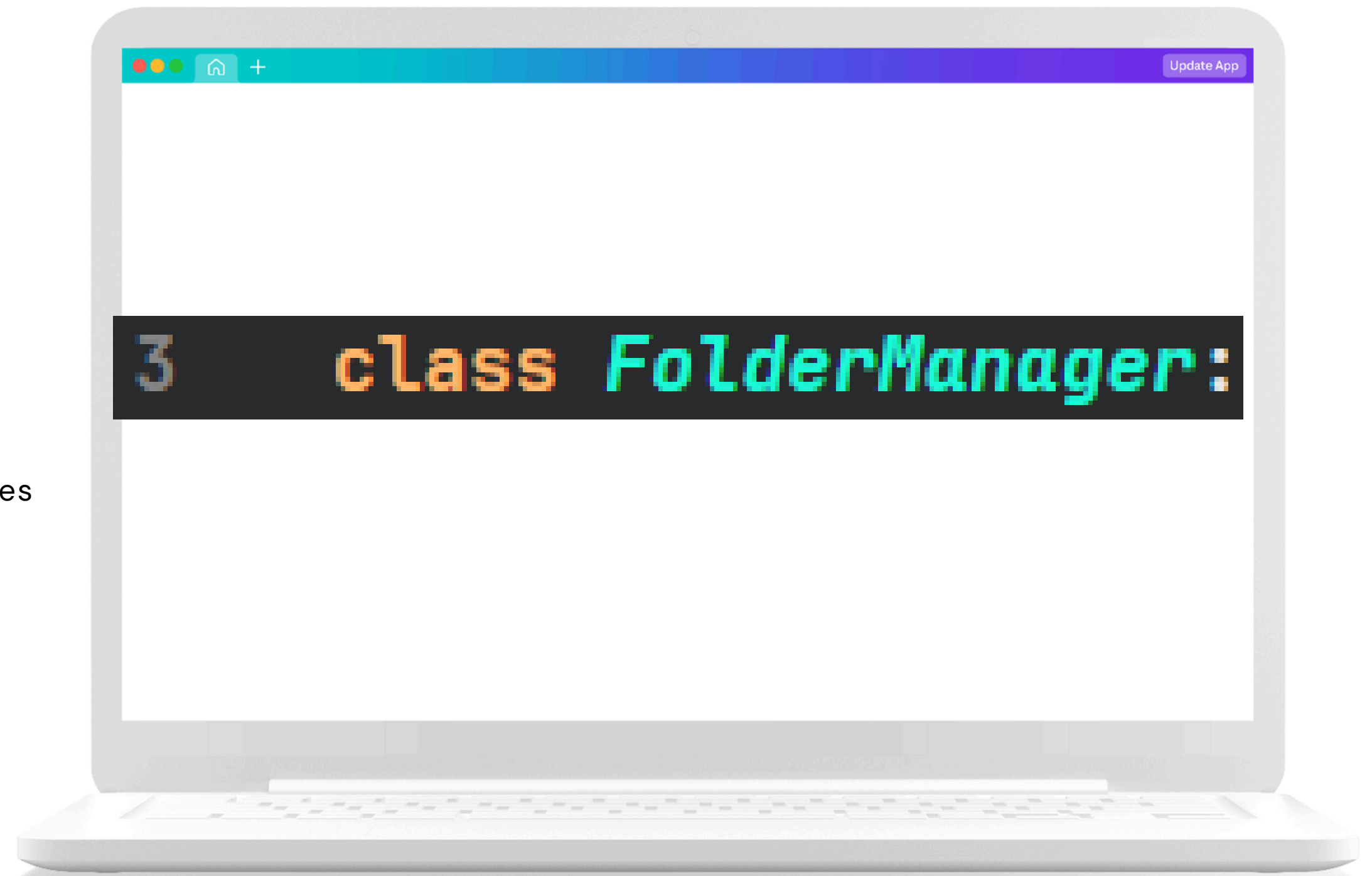
- En este caso, FolderManager es la clase que agrupa todas las operaciones relacionadas con la gestión de carpetas (crear, listar y borrar).
- Crear una instancia: `fm = FolderManager("/ruta/base")` → fm es un objeto de la clase.

FolderManager

Class

```
class FolderManager
```

Define una clase llamada FolderManager. La idea es agrupar funciones relacionadas con la gestión de carpetas.



Qué un constructor?

En POO, un constructor es un método especial que se ejecuta automáticamente cuando se crea un objeto (instancia) de una clase.

- El constructor se llama `__init__`.
- Su objetivo principal es inicializar los atributos del objeto, es decir, darle un estado inicial.
- Siempre recibe como primer parámetro `self`, que representa al objeto que se está creando. (Solo para Python)

`__init__`

Qué son los atributos?

Los atributos son variables que pertenecen a la clase o a la instancia.

- **Atributos de instancia:** se definen normalmente en `__init__` con `self`, por ejemplo `self.base_path`. Cada instancia puede tener un valor distinto
- **Atributos de clase:** se declaran en el cuerpo de la clase (fuera de métodos) y son compartidos por todas las instancias.

path

Qué son los atributos?

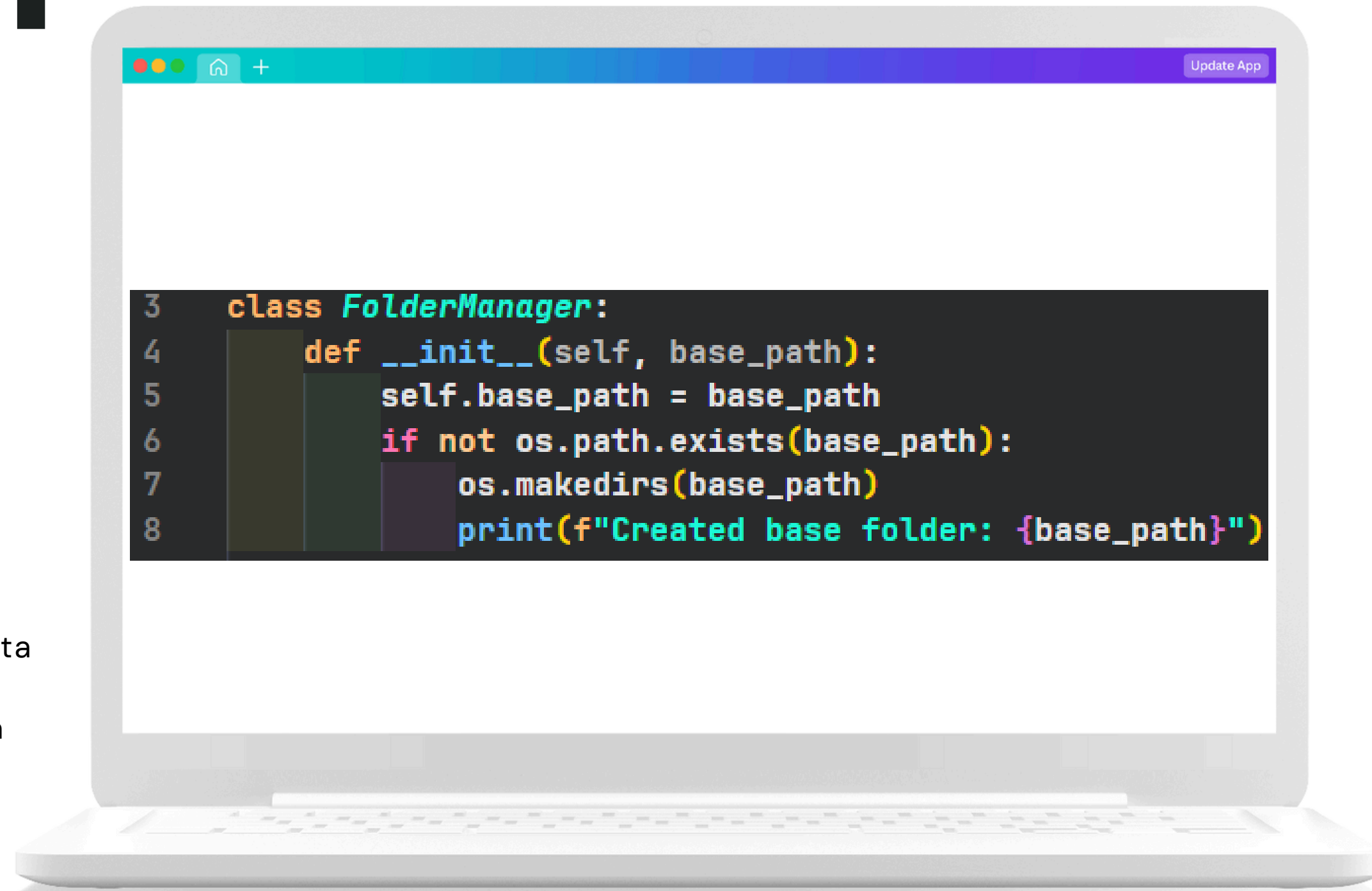
```
55 class Persona:
56     especie = "Human" # atributo de clase
57
58     def __init__(self, name, age):
59         self.name = name # atributo de instancia
60         self.age = age # atributo de instancia
```

path

Constructor

def __init__()

- Recibe base_path, que será la carpeta "raíz" donde trabajará el gestor.
- Guarda ese valor en self.base_path para usarlo en otros métodos.
- Si la ruta no existe, la crea con os.makedirs(base_path).
- Muestra un mensaje confirmando que creó la carpeta base.
- Nota: os.path.exists verifica si existe cualquier cosa con ese nombre (archivo o carpeta). Aquí está bien porque luego creamos una carpeta si no existe.



Qué es un método?

Un método es una función definida dentro de la clase.


- El primer parámetro de un método de instancia es self, que referencia la propia instancia (permite leer/alterar atributos).

create_folder

Método

create_folder

- Construye la ruta completa uniendo base_path y folder_name con os.path.join (así funciona en Windows, Linux, macOS sin preocuparse por separadores).
- Si no existe esa ruta, crea la carpeta con os.makedirs(path) y lo informa.
- Si ya existe, solo lo avisa.
- Nota: os.makedirs crea todas las carpetas intermedias necesarias si no existen.

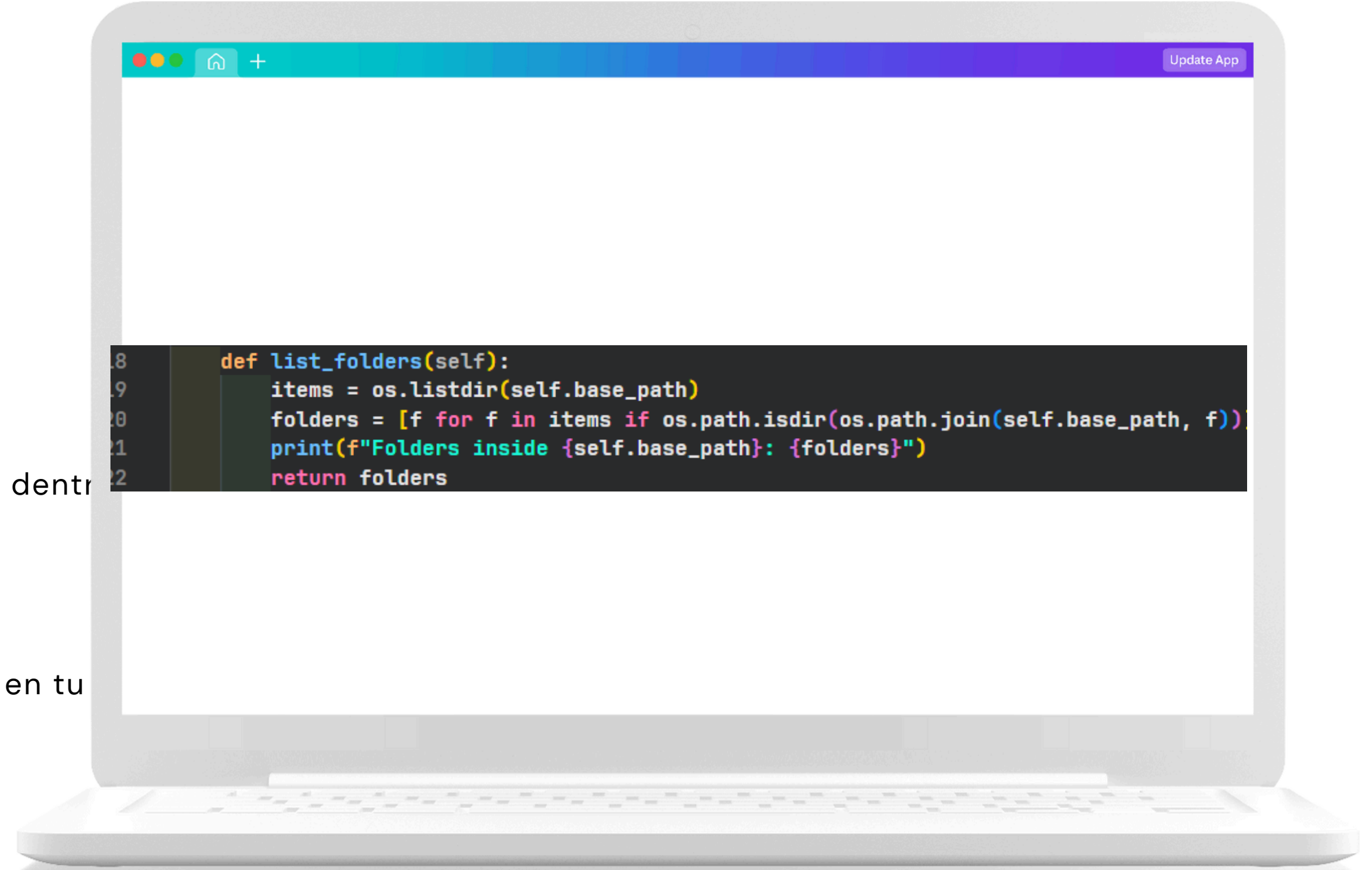


```
10     def create_folder(self, folder_name):
11         path = os.path.join(self.base_path, folder_name)
12         if not os.path.exists(path):
13             os.makedirs(path)
14             print(f"Folder created: {path}")
15         else:
16             print(f"Folder already exists: {path}")
```

Método

list_folders


- Obtiene todos los ítems (archivos y carpetas) dentro de base_path con os.listdir.
- Filtra solo los que son directorios usando os.path.isdir(...) en una lista por comprensión.
- Imprime el listado de carpetas encontradas.
- Devuelve la lista folders (por si quieres usarla en tu programa).



Método

delete_folder

- Construye la ruta completa de la carpeta a borrar.
- Verifica que exista y que sea un directorio.
- La elimina con `os.rmdir(path)` y lo informa.
- Si no existe o no es carpeta, muestra "no encontrado".
- Importante: `os.rmdir` solo borra carpetas vacías. Si la carpeta tiene archivos/subcarpetas, fallará con un error (`OSError`). Para borrar carpetas con contenido usualmente se usa `shutil.rmtree(path)` (debes importar `shutil`).

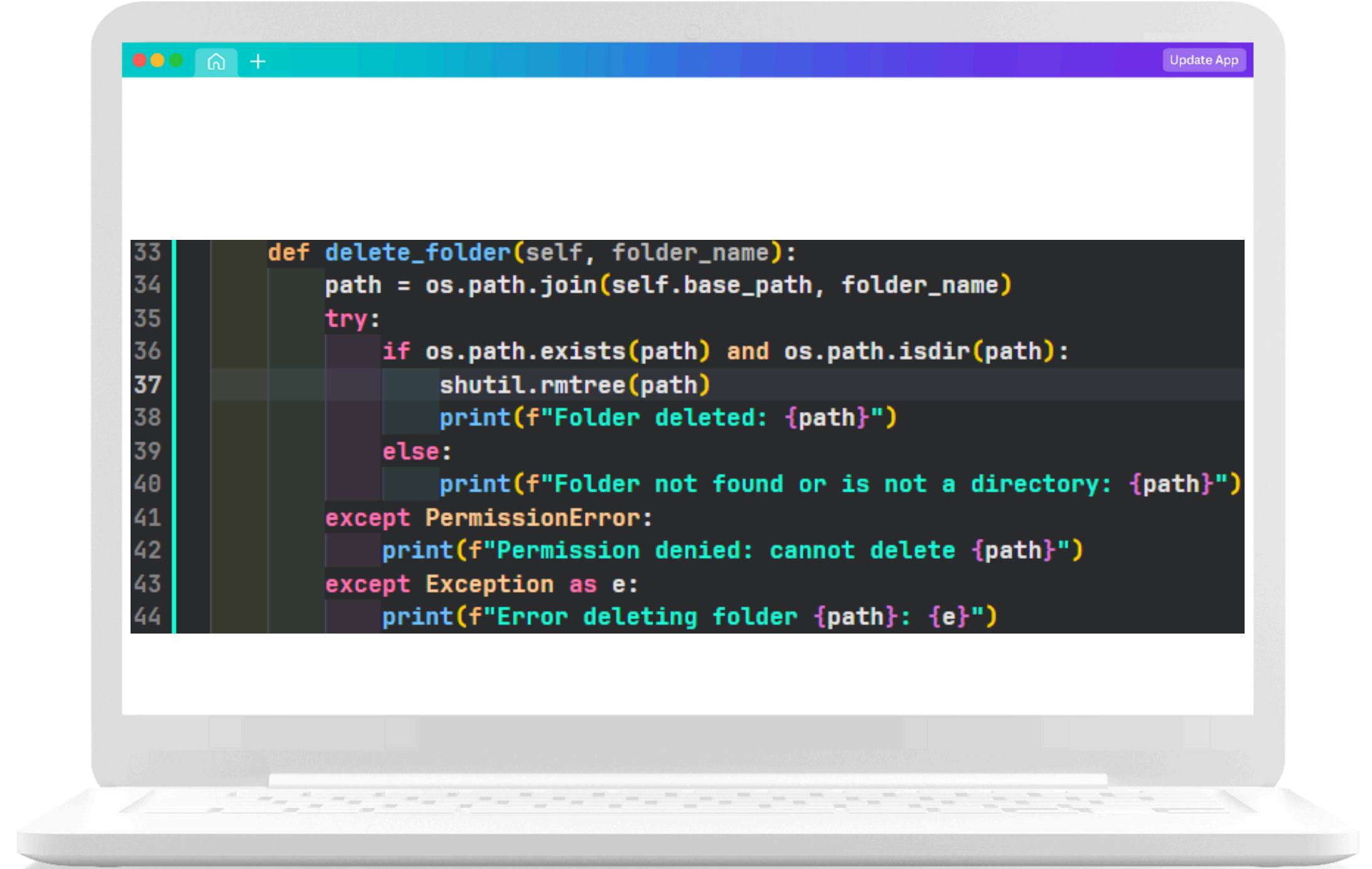


```
24 def delete_folder(self, folder_name):
25     path = os.path.join(self.base_path, folder_name)
26     if os.path.exists(path) and os.path.isdir(path):
27         os.rmdir(path)
28         print(f"Folder deleted: {path}")
29     else:
30         print(f"Folder not found: {path}")
```

Método

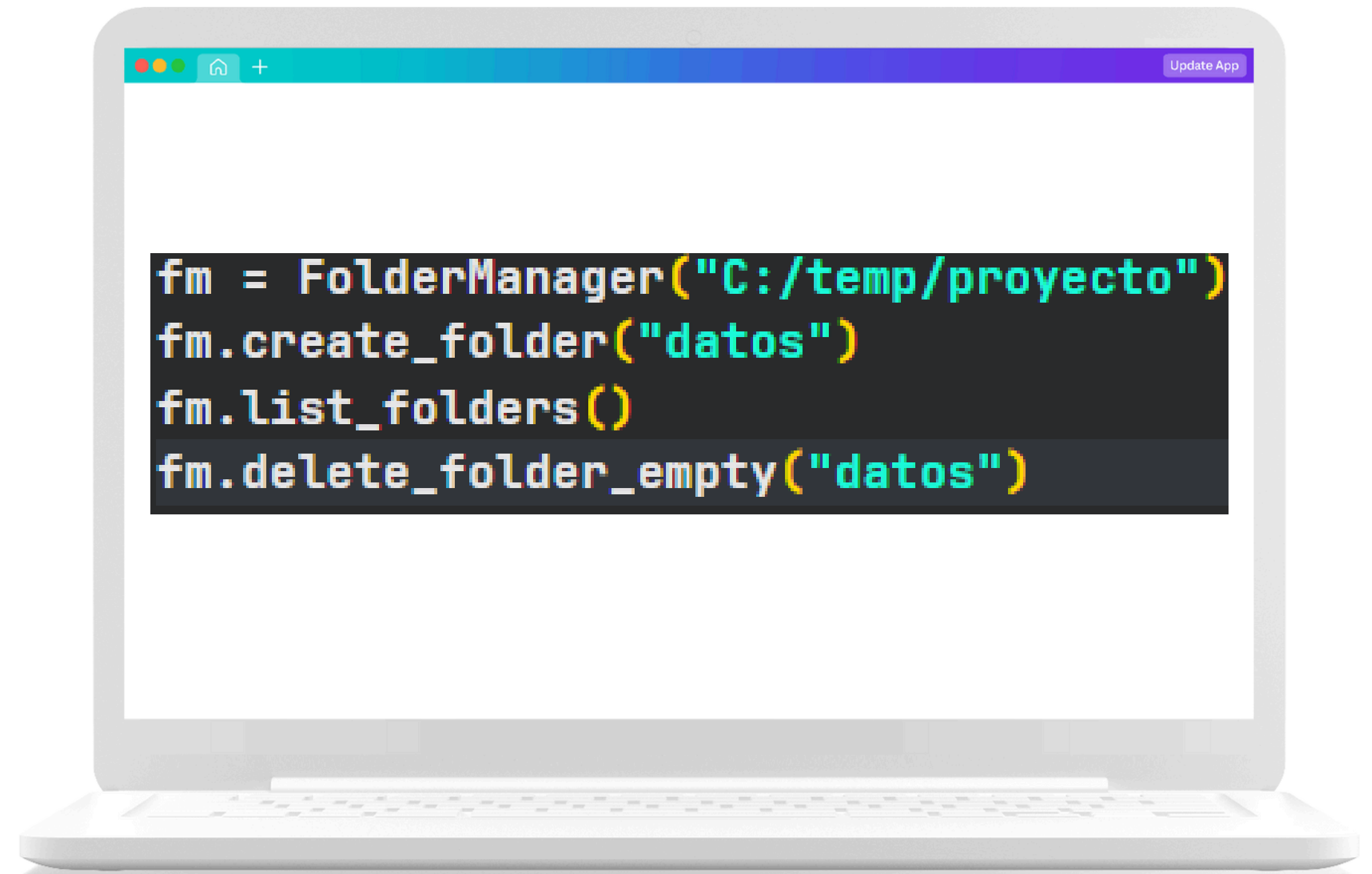
delete_folder

- Se cambia la función `os.rmdir()` por `shutil.rmtree()` para poder eliminar carpetas aunque tengan archivos



Ejecución

Mediante terminal



```
juanm@JuanM-PC MINGW64 ~/Documents/Projects/comfamiliar/Classes/OOP/Assestment (master)  
• $ python Exercise_2.py  
Created base folder: C:/temp/proyecto  
Folder created: C:/temp/proyecto\datos  
Folders inside C:/temp/proyecto: ['datos']  
❖ Folder deleted: C:/temp/proyecto\datos
```