Estás escuchando:
Arre – Simpson Ahuevo

Club de
Programación
ESFM

Desarrollo de Software con Python

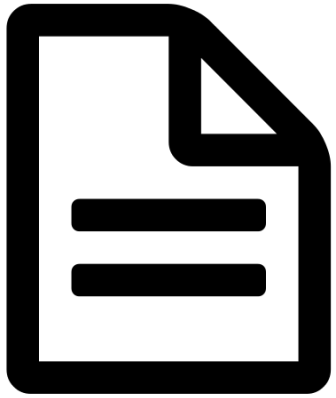
Club de
Programación
ESFM

Desarrollo de Software con Python

Clase 4

Módulos

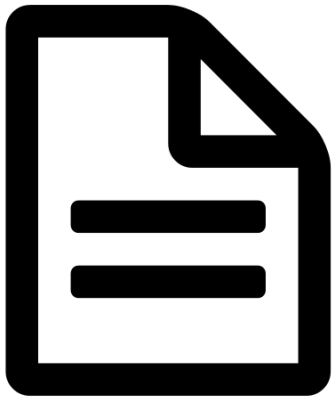
Un módulo es un fichero conteniendo definiciones y declaraciones de Python. El nombre de archivo es el nombre del módulo con el sufijo .py agregado. Dentro de un módulo, el nombre del mismo módulo (como cadena) está disponible en el valor de la variable global __name__.



modulo.py

Módulos

Cada módulo tiene su propio espacio de nombres, el cual es usado como espacio de nombres global para todas las funciones definidas en el módulo. Por lo tanto, el autor de un módulo puede usar variables globales en el módulo sin preocuparse acerca de conflictos con una variable global del usuario. Por otro lado, si sabes lo que estás haciendo puedes acceder a las variables globales de un módulo con la misma notación usada para referirte a sus funciones, nombremodulo.nombreitem.



modulo.py

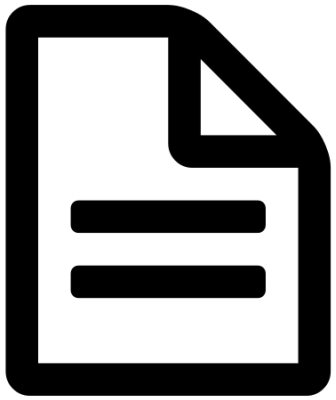
```
import modulo

modulo.literal
```

Es costumbre pero no obligatorio ubicar todas las declaraciones import al principio del módulo (o script, para el caso)

Módulos

Hay una variante de la declaración import que importa nos nombres de un módulo directamente al espacio de nombres del módulo que hace la importación.



modulo.py

```
from modulo import literal1, literal2
```

Esto no introduce en el espacio de nombres local el nombre del módulo desde el cual se está importando (por lo tanto, en el ejemplo, modulo no esta definido)

Módulos

Hay incluso una variante para importar todos los nombres que un módulo define:

modulo.py

```
from modulo import *
```

Esto importa todos los nombres excepto los que inician con un guión bajo (_).
Nota que en general la práctica de importar * de un módulo o paquete está muy mal vista, ya que frecuentemente genera código poco legible.

Módulos

Si el nombre del módulo es seguido por as, el nombre siguiendo as queda ligado directamente al módulo importado.

modulo.py

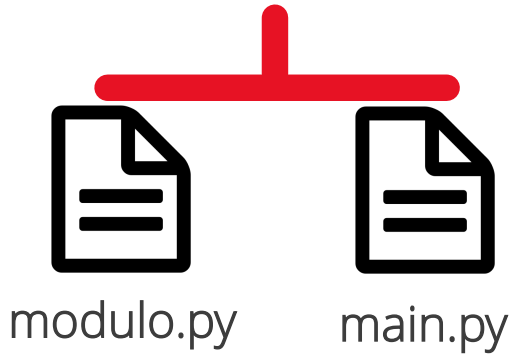
```
import modulo as mod

mod.literal1
```

También se puede utilizar cuando se utiliza from con efectos similares.

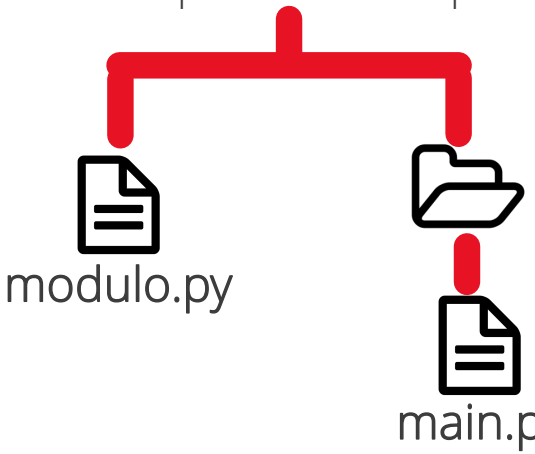
Módulos – importaciones abs y rel

Las importaciones absolutas incluyen la ruta completa a su script, comenzando con la carpeta raíz del programa. Si bien debe separar cada carpeta por un punto, puede hacer que sea tan largo como lo necesite.



import modulo

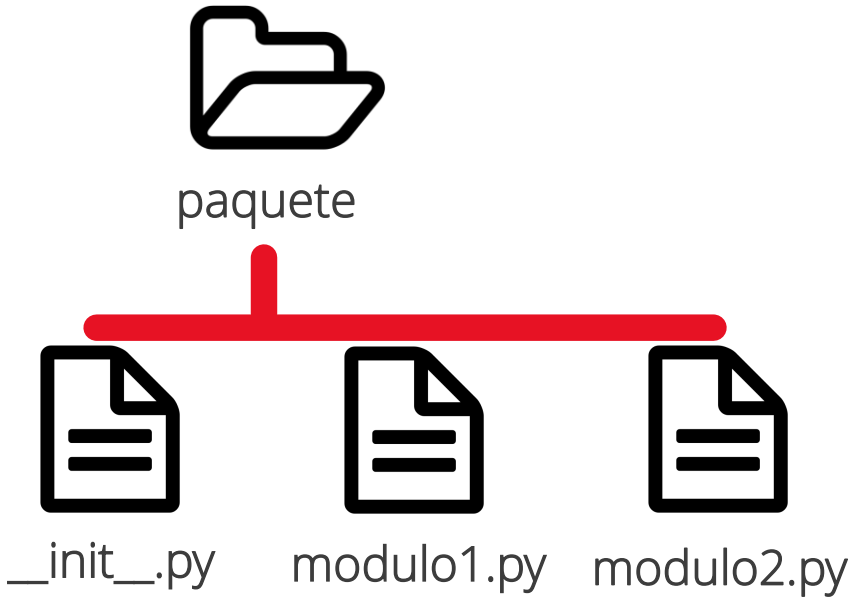
Con las importaciones relativas, solo especifica dónde están sus recursos en relación con el archivo de código actual. En cuanto a la sintaxis, las importaciones relativas utilizan la notación de puntos. Los puntos individuales representan el directorio del script actual. Dos puntos representan la carpeta principal. Los tres puntos significan el abuelo, etc.



from .. import modulo

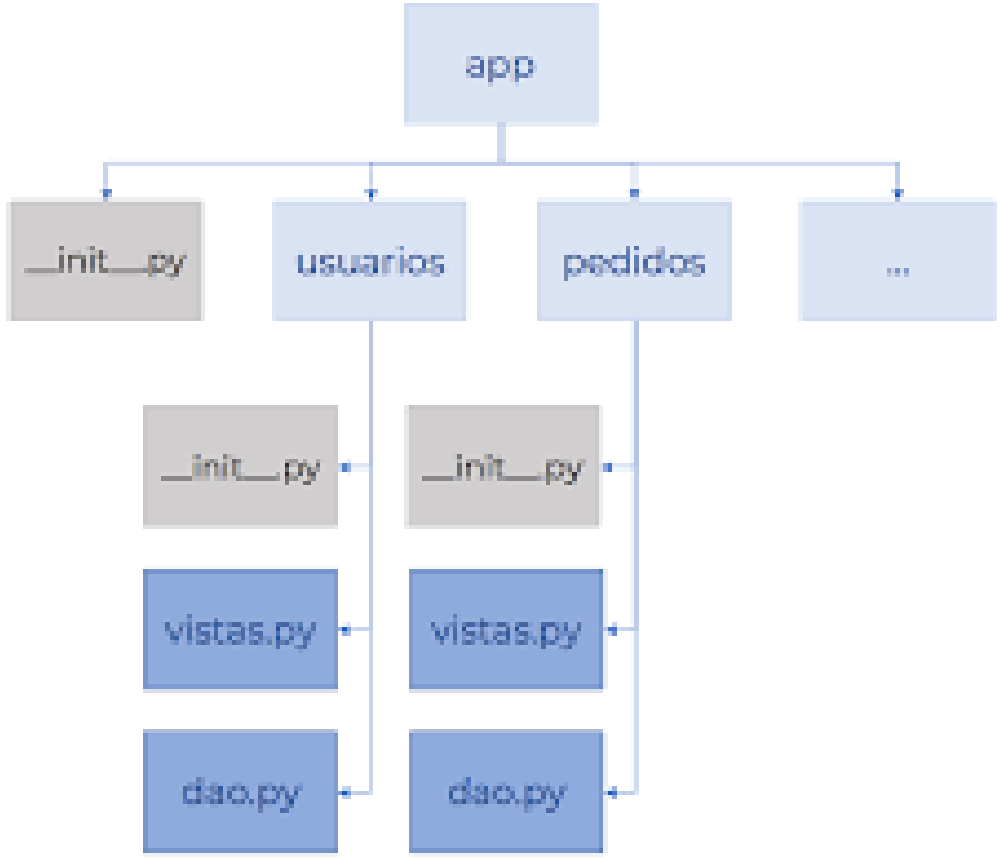
Paquetes

Los Paquetes son una forma de estructurar el espacio de nombres de módulos de Python usando «nombres de módulo con puntos». Por ejemplo, el nombre del módulo A.B designa un submódulo B en un paquete llamado A.



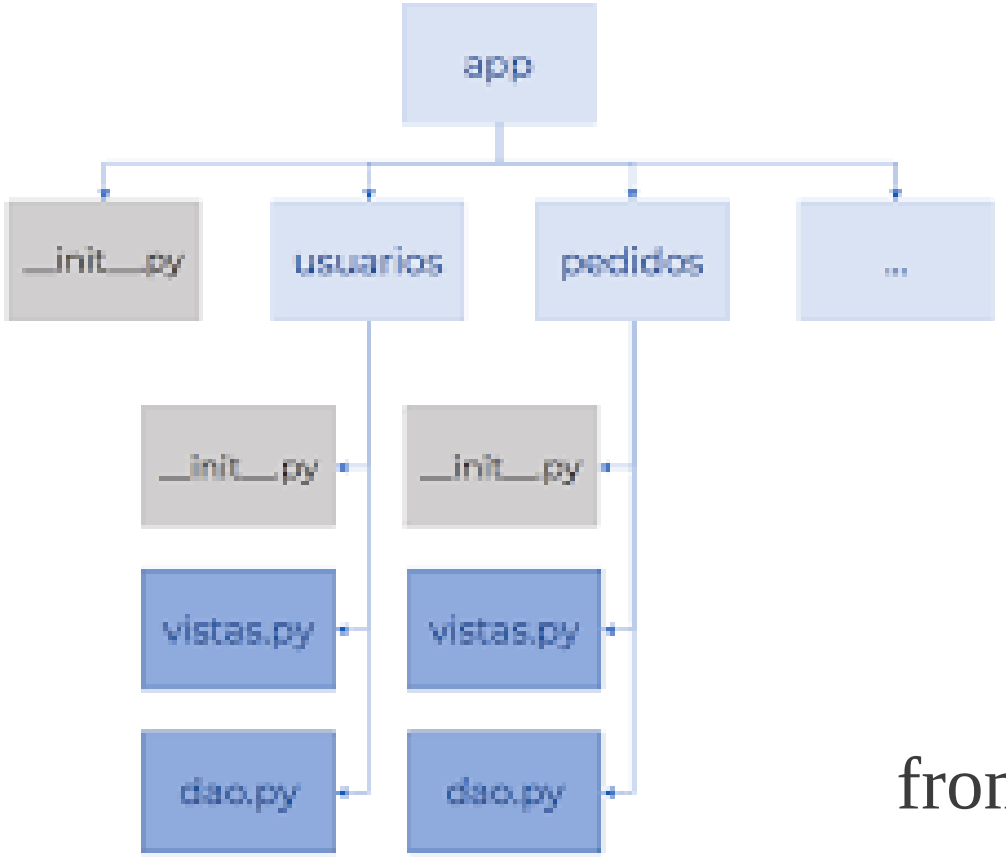
Paquetes

Los archivos __init__.py son obligatorios para que Python trate los directorios que contienen los archivos como paquetes.



Paquetes - Importación

Los usuarios del paquete pueden importar módulos individuales del mismo.




```
import app
app.usuarios.vistas.literal

from app import usuarios
usuarios.vistas.literal

from app.usuarios import vistas
vistas.literal

from app.usuarios.vistas import literal
```

Pip

pip es un sistema de gestión de paquetes utilizado para instalar y administrar paquetes de software escritos en Python. Muchos paquetes pueden ser encontrados en el Python Package Index.

```
$ pip install paquete
$ pip install paquete –upgrade
$ pip install –r archivo.txt

$ pip uninstall paquete

$ pip freeze
$ pip show paquete
```

Entornos virtuales

Las aplicaciones en Python usualmente hacen uso de paquetes y módulos que no forman parte de la librería estándar. Las aplicaciones a veces necesitan una versión específica de una librería, debido a que dicha aplicación requiere que un bug particular haya sido solucionado o bien la aplicación ha sido escrita usando una versión obsoleta de la interfaz de la librería.

```
$ python -m venv nombre-env
```

Esto creará la carpeta tutorial-env si no existe, y también creará las subcarpetas conteniendo la copia del intérprete Python, la librería estándar y los archivos de soporte.

Windows

```
$ nombre-env\Scripts\activate.bat
```

Mac

```
$ source nombre-env/bin/activate
```

```
$ deactivate
```

para salir

TOP 10 MEJORES PAQUETES DE PYTHON

NumPy.- Vectores, matrices

Pandas.- Análisis de datos

Sympy.- Cómputo simbólico (derivadas, integrales,límites)

bs4.- Web Scraping

Cirq.- Computación cuántica

PyTorch.- Machine Learning

Requests.- Envíar solicitudes web

PyQt.- Interfaces gráficas

Matplotlib.- Graficación

Flask.- Páginas web

Django.- Aplicaciones web

SQLAlchemy.- kit de herramientas SQL