
```
from google.colab import drive
drive.mount('/content/drive')
```

# Módulos

Los módulos son archivos python .py que contienen definiciones de funciones, clases y variables que pueden ser reutilizado en distintos programas. Facilitan la reutilización de código y la programación modular. Por su parte, los paquetes son un conjunto de módulos organizados jerárquicamente y relacionados entre sí.

## Creación de módulos

Crear un módulo es tan sencillo como incorporar un código (variables, funciones, clases) en un archivo y guardarlo con extensión .py

Abre el bloc de notas y copia y pega el siguiente código en el:

```
def saludar():
    print("Hola, te saludo desde un módulo externo llamado externo.py")

def despedir():
    print("Adiós, me despido desde el módulo externo")

def saludar_personalizado(nombre):
    print(f"Hola {nombre}, te saludo desde el módulo externo")
```

Ahora guarda el archivo como externo.py

¡Ya tienes tu primer módulo creado!

## Invocación de módulos

Para invocar un módulo debes poner tu módulo en la misma carpeta donde está el programa que lo invocará.

Sintaxis:

```
import nombre_modulo
```

Una vez importado el módulo, ya se puede usar cualquier función que en él se encuentre.

```
import nombre_modulo

nombre_modulo.funcion()
```

Ahora, en el mismo directorio donde está el archivo externo.py, crearemos otro archivo llamado main.py donde realizaremos las importaciones:

```
import externo

externo.saludar()
```

También se pueden importar funciones específicas de un módulo con la siguiente sintaxis:

```
from nombre_modulo import funcion
funcion()
```

De esta forma no es necesario anteponer el nombre del módulo al invocar la función.

```
from externo import despedir

despedir()
```

Para importar todas las funciones de un módulo podemos usar la siguiente sintaxis:

```
from nombre_modulo import *
```

Una última opción de importación de módulo es empleando **ALIAS** de módulo. Esto es darle al módulo otro nombre para el manejo interno en el programa que lo invoca. Miremos:

```
import modulo as alias
o
from nombre_modulo import funcion as alias
```

Continuando con el ejercicio anterior vamos a utilizar el alias (s) para el llamado a función

```
import externo as s

e.saludar_personalizado('Patricia')
```

Existe una función en python que nos permite conocer todas las definiciones internas que forman parte de un determinado módulo. Esta función es **dir()** miremos:

```
import random
print (dir(random))
```

```
import math
print (dir(math))
```

```
import externo as s
print("\nContenido del módulo externo:")
print(dir(s))
print(s.__file__)
```

```
import math
#num= int(input("ingrese num:"))
#raiz= math.sqrt(num)
#print(f"La raiz cuadrada de {num} es {raiz}")
print(dir(math.sqrt))
print(math.sqrt.__doc__)
```

Como observamos en el código anterior podemos importar algunos módulos existentes en Python. Estos módulos son conocidos como módulos estándar

```
import random
```

## Módulos Estándar

Python nos provee una serie de módulos estándar que podemos emplear libremente en nuestros programas sin necesidad de realizar instalaciones adicionales.

En el siguiente enlace podemos observar la librería estándar de Python. Como puedes ver hay una gran cantidad de módulos que podemos utilizar desde nuestros programas.

Librería estándar de Python

## Usando algunos módulos estándar

### Módulo math

El módulo math nos proporciona un conjunto de funciones para realizar operaciones aritméticas (sin, cos, tan, asin, acos, atan) de conversión entre grados y radianes (radians, degrees). además de funciones y constantes de propósito general para el trabajo con números.

```
from math import *
import math


print (f"el factorial de 8 es {math.factorial(8)}")
print (f"el 8.7 aproximado hacia arriba es {ceil(8.7)}")
print (f"el 8.7 aproximado hacia abajo es {floor(8.7)}")
print (f"la raiz cuadrada de 9 es {sqrt(9)}")
print (f"5 elevado a la 3 es {pow(5,3)}")
print (f"el valor de PI es {pi} y el valor de E es {e}")
```

### Módulo random

El módulo random nos permite generar números aleatorios de diferentes maneras y bajo diferentes contextos

```
from random import *
print(f"un aleatorio entre 0 y 1: {random()}")
print(f"quiero un # entre 1 y 6: {randint(1,6)}")
lista=["lulo","manzana","mandarina",4,5]
print(f"de la lista {lista} escojo el {choice(lista)}")
print(f"de la lista {lista} escojo estos 3 {sample(lista,3)}")
```

### Módulo platform

Obtiene información del entorno (hardware/so/software) de ejecución de un programa en Python:

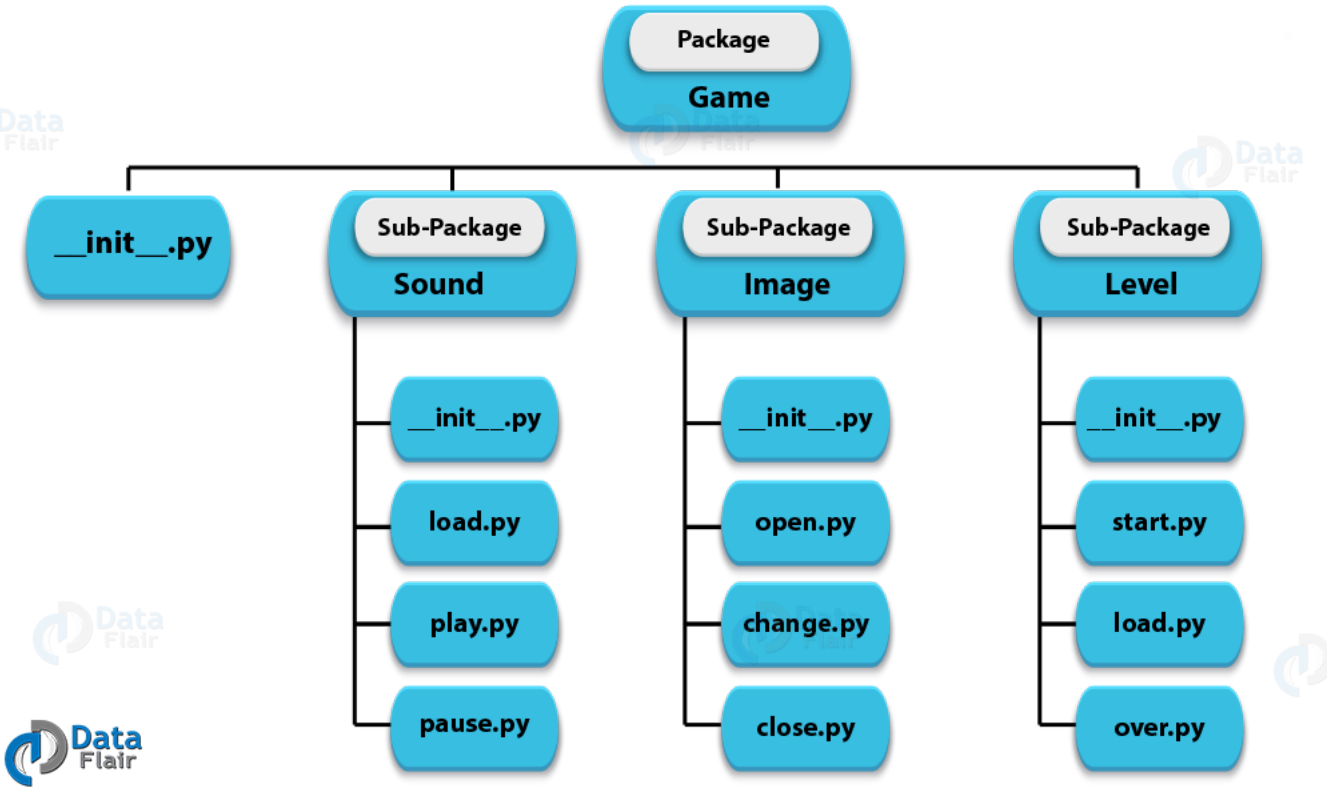
```
import platform as p
print(f"La info de la plataforma de ejecución es: {p.platform()}")
print(f"Familia Procesador: {p.machine()}")
print(f"Sistema Operativo: {p.system()}")
print(f"Version de Python: {p.python_version()}")
```

## Paquetes

Los paquetes permiten agrupar diferentes módulos bajo un mismo nombre y administrar una jerarquía interna de los mismos para un uso más adecuado dentro de nuestros programas.

Pensemos en los paquetes como carpetas que pueden contener subcarpetas y módulos dentro de ellas.

Para que el intérprete de Python reconozca una carpeta como un paquete, esta debe contener un archivo con el nombre **__init __.py** en su interior.




Los paquetes pueden tener una jerarquía de subpaquetes, lo que permite organizar el código de manera más estructurada. Por ejemplo:

import Game.Level.star as g

La importación de paquetes y subpaquetes se realiza de manera similar a la de los módulos. Por ejemplo:

import Game

import Game.Image

## Apropiación

1. Cree un módulo llamado **calculadora** con las siguientes funciones:

   **sumar(*args)**: Recibe cualquier cantidad de números y devuelve su sumatoria.

   **restar(num1, num2)**: Devuelve el resultado de restarle al num1 el num2.

   **multiplicar(*args)**: Recibe cualquier cantidad de números y retorna el resultado de su multiplicación.

   **dividir(num1, num2)**: Devuelve el resultado de dividir num1 entre num2.

   **Nota:** Todas las funciones deben gestionar correctamente las excepciones.

Cree un programa principal donde importe el módulo calculadora y genere un menú como el siguiente a través del cual se haga uso de las funciones incluídas en el módulo calculadora:

```
1. Sumar
2. Restar
3. Multiplicar
4. Dividir
5. Salir
```

2. Dado el siguiente código, cree la estructura de paquetes y módulos para que las importaciones y llamado a funciones trabajen correctamente. Complemente el código de ser necesario.

```
import mi_paquete.greetings as ext
import mi_paquete.calculadora as ca
import mi_paquete.ropa.uniforme as uni

ext.saludar()
ext.hi()
print(ca.sumar(3,6,7,9))
uni.camisa()
ext.despedir()
```

3. Consulte algún módulo de la biblioteca estándar de Python Librería estándar de Python y realice una exposición en la que presente:

- Nombre del módulo o paquete
- Propósito
- Lista de principales funciones
- Ejemplo en código de su utilización