



Python:

Usos en servicios web

INTRODUCCIÓN A PYTHON

Esta clase

- ¿Qué es Python?
- Historia del Lenguaje
- Estructura básica del lenguaje y sus partes

Primero que nada

Preguntas siempre bienvenidas, por favor! Usar el chat de zoom o levantar la mano.

Si algún momento algo va muy rápido me pueden avisar

Python



Lenguaje de programación interpretado de tipado dinámico.

Su filosofía motiva una sintaxis y código simple y legible que permite ser productivo de manera muy rápida sin requerir tantas dependencias.

Muy fácil de aprender! Aquellos que hablan inglés encontrarán que es muy fácil de seguir.

Python



Disponible desde los 80, muchas versiones con el pasar de los años.

La versión más reciente es Python 3.11, pero por estabilidad usaremos terminología de 3.10.

[Python Release Python 3.10.0 | Python.org](#)

Python



Un lenguaje gratuito y de código libre! Todos pueden colaborar en el lenguaje y revisar como funciona sin ninguna obligación financiera.

Es multiplataforma, es decir que podemos ejecutarlo en PC, Mac, o inclusive desde teléfonos móviles que tengan capacidades más avanzadas.



¿Por qué usar Python?

Python es un lenguaje muy fácil de usar como veremos, tiene una muy amplia cantidad de librerías gratuitas que permiten crear proyectos de muchos tipos.

Aplicaciones web, Análisis de datos, Machine Learning, Automatización: Todo esto es posible mediante una serie de ecosistemas libres disponibles y usados por cientos de miles de usuarios.

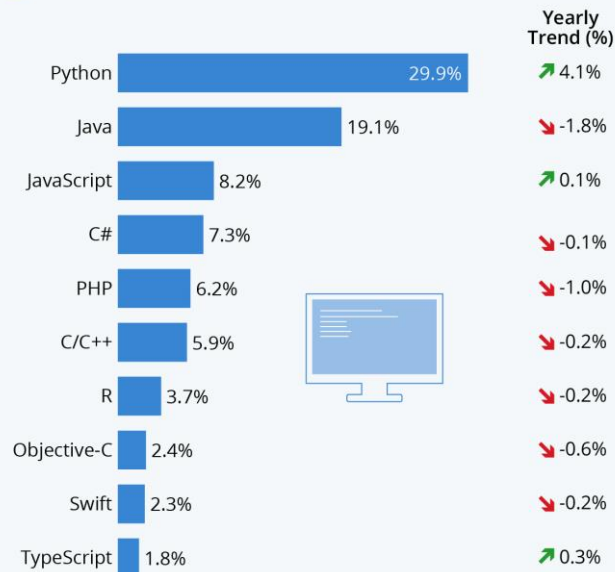
Usado por grandes y pequeñas compañías!

¿Por qué usar Python?



Python Remains Most Popular Programming Language

Popularity of each programming language based on share of tutorial searches in Google



Yearly trend compares percent change from Feb 2019 to Feb 2020
Sources: GitHub, Google Trends



statista



¿Por qué usar Python?

Al ser un lenguaje muy legible, la curva de aprendizaje es muy corta. Se puede empezar a trabajar con él una vez instalado en cuestión de minutos!

Java

```
1 File dir = new File("."); // get current directory
2 File fin = new File(
3     dir.getCanonicalPath() + File.separator + "Code.txt"
4 );
5
6 FileInputStream fis = new FileInputStream(fin);
7
8 // Construct the BufferedReader object
9 BufferedReader in = new BufferedReader(new InputStreamReader(fis));
10
11 String aLine = null;
12 while ((aLine = in.readLine()) != null) {
13     // Process each line, here we count empty lines
14     if (aLine.trim().length() == 0) {}
15 }
16
17 // do not forget to close the buffer reader
18 in.close();
```

Python

```
1 my_file = open("/home/xiaoran/Desktop/test.txt")
2
3 print(my_file.read())
4 my_file.close()
```



¿Por qué usar Python?

Muchos paradigmas, en Python podemos escribir aplicaciones orientadas a objetos, funcionales o imperativa.

En su momento veremos en más detalle como operar estos modelos con el lenguaje.



¿Cómo funciona Python?

Python es un lenguaje dinámico interpretado.

¿Interpretado? Significa que existe un interprete, un programa encargado de procesar el código de Python en tiempo real en lugar de compilarlo desde el principio.

Tiene sus ventajas ya que permite colaborar en el mismo código sin necesidad de preocuparse de la plataforma.



Tipado en Python

Naturalmente el lenguaje no tiene tipos al ser dinámico. Por ejemplo

En lenguajes estáticos:

```
int num = 5;
```

```
num = ´texto´
```

```
# ERROR!!!!!!!!!!!!!!
```

En Python

```
num = 5
```

```
num = ´texto´
```

```
# No hay problemas
```



Tipado en Python

Esto tiene sus beneficios como desventajas, en siguientes clases aprenderemos como agregar tipado a Python en caso de ser deseado. Esto tiene beneficios especiales como por ejemplo asegurar que el código sea consistente.



Usando Python

Nada más se requiere instalar el intérprete en su entorno preferido, también exploraremos como usar entornos aislados.

Una buena guía de Microsoft para instalar Python por primera vez

[Python en Windows para principiantes | Microsoft Docs](#)



Un primer programa

```
print("Hola mundo")
```



Usando Python

Una vez instalado Python se puede usar directamente desde el intérprete interactivo

```
doc@LAPTOP-60FT0GV9:~$ python3
Python 3.8.10 (default, Jun  2 2021, 10:49:15)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more
information.
>>> print("hola")
hola
>>>
```

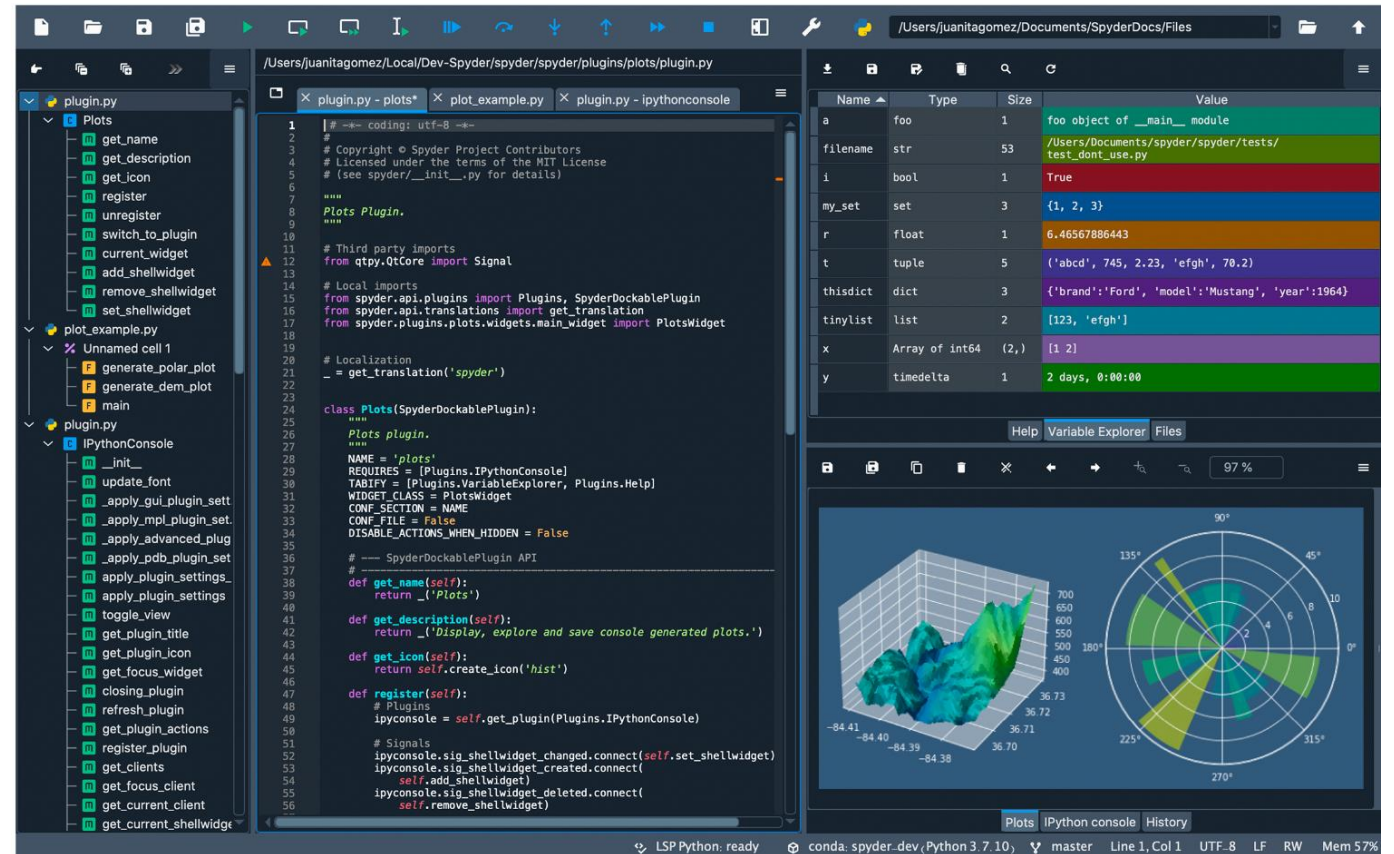
O en su momento veremos como usar IDEs o entornos de desarrollo que permite ejecutar Python de manera más eficiente y productiva.

Ejemplo: Spyder



[Spyder IDE \(spyder-ide.org\)](https://spyder-ide.org).

Entorno desarrollado para
Trabajar en Python



The screenshot displays the Spyder IDE interface with the following components:

- Left Panel (File Explorer):** Shows a project structure with files like `plugin.py`, `plots`, `plot_example.py`, and `plugin.py - ipythonconsole`.
- Central Panel (Code Editor):** Displays the source code for `plugin.py`, which includes a `Plots` class inheriting from `SpyderDockablePlugin`. The code defines methods for `get_name`, `get_description`, `get_icon`, and `register`.
- Right Panel (Variable Explorer):** Shows a table of variables in the current scope:

Name	Type	Size	Value
a	foo	1	foo object of __main__ module
filename	str	53	/Users/juanitagomez/spyder/spyder/tests/test_dont_use.py
i	bool	1	True
my_set	set	3	{1, 2, 3}
r	float	1	6.46567886443
t	tuple	5	('abcd', 745, 2.23, 'efgh', 70.2)
thisdict	dict	3	{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
tinylist	list	2	[123, 'efgh']
x	Array of int64	(2,)	[1 2]
y	timedelta	1	2 days, 0:00:00

- Bottom Panel (Plots):** Displays a 3D surface plot of a function, showing a complex, multi-peaked surface in a 3D coordinate system.

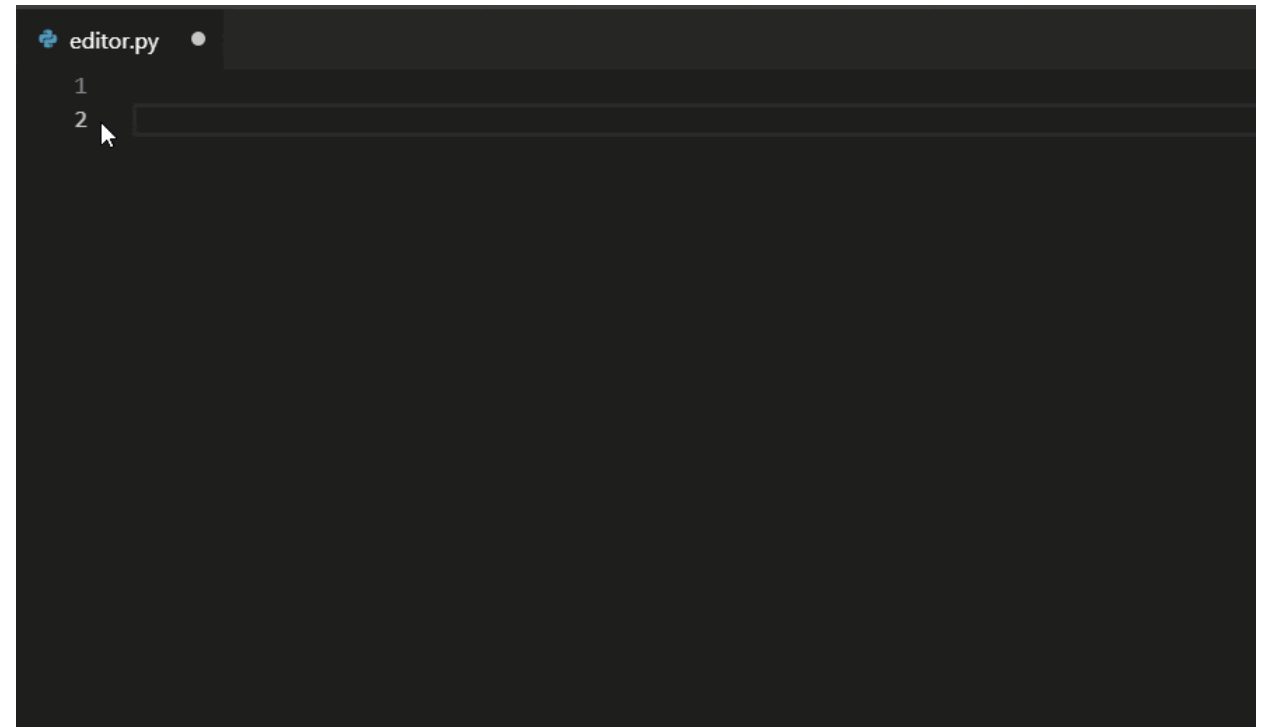
The status bar at the bottom indicates the environment: `LSP Python: ready`, `conda: spyder-dev (Python 3.7.10)`, `master`, `Line 1, Col 1`, `UTF-8`, `LF`, `RW`, and `Mem 57%`.



Ejemplo: Visual Studio Code

[Python in Visual Studio Code](#)

IDE ligero y muy útil para
Trabajar con Python u otros
lenguajes





Un primer programa

En Python, no existen las llaves {}, es decir, que todo depende de los espacios e indentación.

En las siguientes clases exploraremos herramientas que nos permiten automatizar el estilo y formato de nuestro código con el fin de tener código limpio y legible.

```
print("Hola mundo")

lista = [1, 2, 3, 4]

for elemento in lista:
    print(f"{elemento} multiplicado por dos {elemento*2}")
```



Conclusión

Preguntas?