

PRO. Introducción Python



¿Qué es Python?



Python es un lenguaje de programación creado por Guido van Rossum a principios de los años 90 cuyo nombre está inspirado en el grupo de cómicos ingleses “Monty Python”. Es un lenguaje similar a Perl, pero con una **sintaxis muy limpia** y que favorece un **código legible**.

Licencia

Python se desarrolla bajo una licencia de Open source o código abierto aprobada por OSI, por lo que se puede usar y distribuir libremente, incluso para uso comercial.

La licencia de Python es administrada por [Python Software Foundation](https://python.org/).

Aplicaciones

El [Python Package Index \(PyPI\)](https://pypi.org/) o en español significa *Índice de paquetes de Python* alberga miles de módulos de terceros para Python.

Tanto la biblioteca estándar de Python como los módulos aportados por la comunidad permiten infinitas posibilidades:

- [Desarrollo web e Internet](#).
- [Acceso a la base de datos](#).
- [GUIs de escritorio](#).
- [Científico y numérico](#).
- [Educación](#).
- [Programación de red](#).
- [Desarrollo de Software y Juegos](#).

Comunidad

La Python Software Foundation (PSF) es una corporación sin ánimo de lucro que posee los derechos de propiedad intelectual detrás del lenguaje de programación Python.

Administramos las licencias de código abierto para Python versión 2.1 y posteriores, y poseemos y protegemos las marcas comerciales asociadas con Python.

También realiza la conferencia PyCon de Norteamérica anualmente, apoya otras conferencias de Python en todo el mundo y financia el desarrollo relacionado con Python

La misión de Python Software Foundation es promover, proteger y avanzar el lenguaje de programación Python, y apoyar y facilitar el crecimiento de una comunidad diversa e internacional de programadores de Python.

Características

Python es un lenguaje de programación **interpretado** o de script, con **tipado dinámico**, **fuertemente tipado**, **multiplataforma** y **orientado a objetos**.

Lenguaje interpretado o de script

Un lenguaje **interpretado** o de script es aquel que se ejecuta utilizando un **programa intermedio** llamado **intérprete**, en lugar de compilar el código a lenguaje máquina que pueda comprender y ejecutar directamente una computadora (lenguajes compilados). Un inconveniente de los lenguajes interpretados se **ejecutan** de forma más **lenta** que los compilados. Sin embargo los lenguajes interpretados son más **flexibles** y más **portables**. Python tiene, no obstante, muchas de las características de los lenguajes compilados, por lo que se podría decir que es **semi-interpretado**. En Python, como en Java y muchos otros lenguajes, el código fuente se traduce a un **pseudo código máquina intermedio** llamado **bytecode** la primera vez que se ejecuta, generando archivos **.pyc** o **.pyo** (bytecode optimizado), que son los que se ejecutarán en sucesivas ocasiones.

Tipado dinámico

La característica de tipado dinámico se refiere a que no es necesario declarar el **tipo de dato** que va a contener una determinada **variable**, sino que su tipo se determinará **en tiempo de ejecución** según el tipo del valor al que se asigne, y el tipo de esta variable puede cambiar si se le asigna un valor de otro tipo.

Fuertemente tipado

No se permite tratar a una variable como si fuera de un tipo distinto al que tiene, es necesario **convertir de forma explícita** dicha variable al nuevo tipo previamente. Por ejemplo, si tenemos una variable que contiene un texto (variable de tipo cadena o string) no podremos tratarla como un número (sumar la cadena "9" y el número 8). En otros lenguajes el tipo de la variable cambiaría para adaptarse al comportamiento esperado, aunque esto es más propenso a errores.

Multiplataforma

El intérprete de Python está disponible en multitud de plataformas (UNIX, Solaris, Linux, DOS, Windows, Mac OS, etc.) por lo que si no utilizamos **librerías específicas** de cada plataforma nuestro programa podrá ejecutarse en todos estos sistemas sin grandes cambios.

Orientado a objetos

La orientación a objetos es un paradigma de programación en el que los **conceptos del mundo real** relevantes para nuestro problema se trasladan a **clases y objetos** en nuestro programa. La **ejecución** del programa consiste en una serie de **interacciones** entre los **objetos**. Python también permite otros paradigmas de programación:

- Programación imperativa
- Programación funcional
- Programación orientada a aspectos.

¿Por qué Python?

Python es un lenguaje que todo programador debería conocer. Su **sintaxis simple, clara y sencilla**; el **tipado dinámico**, el **gestor de memoria**, la gran cantidad de **librerías** disponibles y la **potencia del lenguaje**, entre otros, hacen que **desarrollar** una aplicación en Python sea **sencillo**, muy **rápido** y, lo que es más importante, **divertido**. La sintaxis de Python es tan sencilla y cercana al lenguaje natural que los programas elaborados en Python parecen **pseudocódigo**. Por este motivo se trata además de uno de los mejores lenguajes para **comenzar a programar**. Python **no es adecuado** sin embargo para la programación de bajo nivel o para aplicaciones en las que el rendimiento sea crítico.

Herramientas básicas

Existen dos formas de ejecutar código Python. Podemos escribir líneas de código en el intérprete y obtener una respuesta del intérprete para cada línea (**sesión interactiva**) o bien podemos escribir el código de un programa en un **archivo de texto** y ejecutarlo.

Sesión interactiva

Podemos obtener una sesión interactiva de Python simplemente ejecutando:

```
$ python
Python 2.7.18 (default, Mar  8 2021, 13:02:45)
[GCC 9.3.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

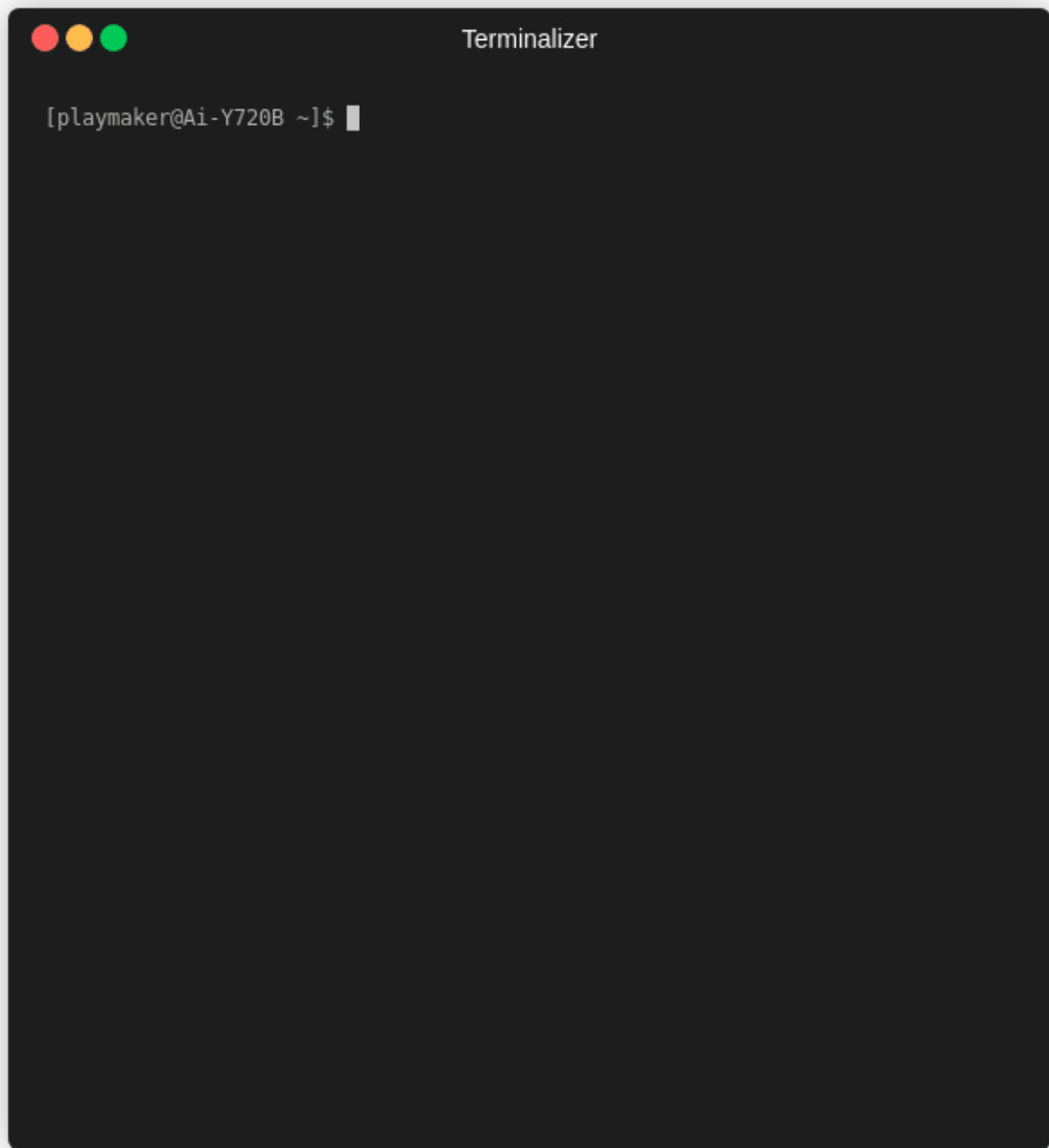
También podemos realizar una sesión interactiva utilizar **bpython**, en lugar de la consola interactiva de Python.

Para instalarlo si ya tenemos Python ejecutamos:

```
$ pip install bpython
```

bpython cuenta con características añadidas muy interesantes, como:

- Autocompletado con sugerencias a medida que escribimos
- Resaltado de sintaxis en línea
- Autoindentado



Editores

En cuanto a editores de código e IDEs existen múltiples opciones:

- Idle <- Intérprete y editor
- Eric
- VSCode
- PyDev en Eclipse
- Komodo
- vim nano

Mi primer programa en Python

El primer programa que vamos a escribir en Python es el clásico Hola Mundo, y en este lenguaje es tan simple como:

```
print "Hola Mundo"
```

Desde el interprete

Vamos a probarlo primero en el interprete. Ejecutamos python o ipython, escribimos la línea anterior y pulsamos Enter. El intérprete responderá mostrando en la consola el texto Hola Mundo.

```
$ bpython
bpython version 0.21 on top of Python 3.8.10 /usr/bin/python3
>>> print("Hola Mundo")
Hola Mundo
>>>
```

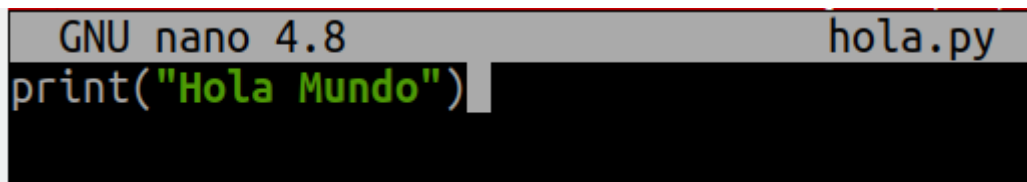
Para salir del intérprete tecleamos `CTRL + D`

Desde un archivo de texto

Vamos ahora a crear un archivo de texto con el código anterior. Abre un editor de texto y copia la línea anterior. Guárdalo como `hola.py`, por ejemplo:

```
$ nano hola.py
```

Escribimos la línea:



```
GNU nano 4.8 hola.py
print("Hola Mundo")
```

Guardamos y salimos (`CTRL + X`)

Ejecutar este programa es tan sencillo como indicarle el nombre del archivo a ejecutar al intérprete de Python

```
$ python hola.py
Hola Mundo
```

Generando un ejecutable

Si queremos convertir el archivo en ejecutable en Linux tenemos que dar los siguientes pasos:

1. Editar el fichero de forma que su contenido sea:

```
#!/usr/bin/python3
print("Hola mundo")
```

A esta línea se le conoce en el mundo Unix como shebang, hashbang o sharpbang. El par de caracteres `#!` indica al sistema operativo que dicho script se debe ejecutar utilizando el intérprete especificado a continuación (`/usr/bin/python3`)

2. Damos permiso de ejecución al archivo

```
$ chmod +x hola.py
```

3. Ejecutamos:

```
$ ./hola.py  
Hola mundo
```