

## Aprende a programar Python



*Logo Python*

**Versión 0.99.2**



Licencia CC by SA by @javacasm

José Antonio Vacas Martínez

<https://elCacharreo.com>

Septiembre 2022

## Introducción

### Historia de Python

Python fue creado a finales de los 80 por [Guido Van Rossum](#) (@gvanrossum en twitter) y desde entonces se ha ido mejorando y desarrollando por una enorme comunidad de usuarios. Si quieres saber más sobre su historia, puedes encontrar más detalles en la [página sobre la historia de Python en la wikipedia](#).

### Algunas características de Python

- [Debe su nombre al grupo Monty Python](#), del que su creador Guido es un gran fan (no sé porqué he puesto este dato en primer lugar...).
- Python es open source, lo que quiere decir que se puede usar en todos los sistemas operativos y por supuesto totalmente gratuito. En [este repositorio de github, llamado cPython \(código C de Python\)](#) puedes encontrar su código fuente, al que han contribuido casi 1500 desarrolladores.
- En su [página web](#) podemos encontrar documentación y tutoriales de todos los niveles. También podemos descargar [desde su página](#) todas las versiones y para casi todos los sistemas operativos.
- La versión actual a día de hoy es la 3.10, estando en desarrollo la 3.11. Sobre la rama 3.8 y 3.9 se arreglan errores, sobre 3.7 y 3.6 sólo se arreglan temas de seguridad y la versión 2.7, presentada en 2010, se considera que ya llegó al fin de su vida útil en 2020.
- Existe la [Python Software Foundation \(PSF\)](#) que ayuda a la creación de eventos globales y locales. En 2019 entregó más de 300.000\$ en becas y ayudas en 60 países.
- ¿Para qué sirve Python? Es un lenguaje de propósito general, es decir podemos usarlo para hacer casi cualquier tipo de programa o aplicación. Por ejemplo:
  - Aplicaciones web: de hecho existen varios framework (entornos de desarrollo) como django, especializados en la creación de páginas web
  - Aplicaciones de escritorio en cualquier plataforma
  - Scripts (ficheros de lotes) para automatizar tareas repetitivas.
  - Creación de juegos, tenemos multitud de módulos que nos facilitan enormemente la creación de videojuegos.

- Es multiplataforma, es decir podemos usar nuestro código en cualquier plataforma (siempre que todos los módulos que usemos estén disponibles): Windows, MacOS, Linux, Unix,... De hecho viene instalado en los sistemas operativos Linux y MacOS, puesto que parte de los programas incluidos en estos lo usan.

Cuando decimos que es multiplataforma también nos referimos a que existe una versión de Python adaptada para ejecutarse en dispositivos embebidos y microcontroladores llamada [microPython](#) que funciona en equipos con reducidas prestaciones.

- Usa una sintaxis muy sencilla y fácil de comprender, lo que nos ayuda a aprender a programar leyendo otros programas. De hecho en su creación se puso un gran énfasis en conseguir un código muy legible.
- La estructura del código es más legible a simple vista.
- Es muy didáctico y uno de los lenguajes más adecuados para quien no ha programado nunca, y actualmente muchas universidades lo utilizan como lenguaje de entrada al mundo de la programación
- La curva de aprendizaje tiene una pendiente muy alta, pero un escalón inicial muy bajo, lo que nos permite empezar muy rápido a hacer cosas interesantes, casi desde el primer momento.
- Cuenta con multitud de librerías y módulos, lo que nos facilita mucho la tarea de programar, sólo tenemos que buscar si ya existe un módulo que nos sirva y adaptar algún ejemplo. Además estos módulos suelen ser también open source.

Algunos de estos módulos están programados completamente en Python y otros son recubrimientos de Python que internamente usan librerías escritas en C para optimizar el rendimiento

Podemos encontrar módulos para procesamiento de imágenes o para Data Mining (procesamiento masivo de datos) o para trabajar en IA (inteligencia artificial).

- Por todo esto se ha convertido en un estándar de hecho en el mundo de la ciencia, en todo lo relacionado con la investigación de IA, visión artificial, bioinformática, genética y en la investigación científica.
- Casi todas las grandes empresas del mundo del software utilizan Python, con Google a la cabeza, seguida por Facebook, Industrial Light & Magic (la de efectos especiales), Instagram, Spotify, Netflix, Dropbox,... hasta la mismísima NASA. En [esta página](#) puedes ver casos de éxito de desarrollos en Python.

- También son muchos los programas/aplicaciones/webs que usamos cada día y que no funcionarían sin Python, como por ejemplo Youtube, Google, Instagram, Reddit, Blender, Dropbox, BitTorrent, OpenShot,

## Algunas características técnicas de Python

- Es interpretado: lo que quiere decir que nuestro código se va ejecutando línea a línea, sin que se haga una traducción global a un ejecutable. Eso nos permite que una de las formas de desarrollar nuestro código sea desde una consola interactiva, donde vamos añadiendo las órdenes y estas se ejecutan. Posteriormente podemos pasar estas órdenes a un archivo. Sin duda es una de las características más importantes de Python. También existen compiladores de Python que nos permiten convertir nuestro código a un ejecutable que el ordenador entiende directamente, pero no es la forma habitual de trabajar con Python

Al ser interpretado nos podemos encontrar con que el rendimiento de nuestros programas no es el mejor y no llegamos a exprimir la potencia del ordenador.

Con la potencia de los ordenadores de hoy en día y utilizando determinados módulos y características podemos mejorar mucho el rendimiento de nuestro código

Puedes encontrar algo más de detalle sobre este tema en el siguiente apéndice.

- Python no utiliza delimitadores para indicar los bloques de código como otros lenguajes ({...} en C, C++ o Java), si no que la indentación de las líneas (los espacios iniciales), marca estos bloques. Esto nos obliga a que la estructura del código sea más legible a simple vista.
- Tiene una comunidad muy activa, tanto en el desarrollo del lenguaje como en la documentación y la cantidad de módulos que se crean.

## Apéndice: Compilados vs Interpretados

El compilador traduce nuestro código una única vez a lenguaje máquina que entiende el procesador. Este proceso puede ser lento, pero sólo se hace una vez.

El intérprete va traduciendo línea por línea nuestro código a un lenguaje intermedio que se ejecuta sobre unas librerías base, por lo no puede una optimización global.

Si estamos haciendo cambios continuamente sobre nuestro código, un lenguaje interpretado será más eficiente, pero una vez que tenemos un código definitivo, la compilación será mejor.

Tipo	Ventajas	Inconvenientes
Compilados	se hace una vez y se aprovecha muchas veces	Cada cambio implica una recompilación
Compilados	Ejecución más rápida	Hay que compilarlos en cada plataforma
Interpretados	Suelen ser de más alto nivel	
Interpretados	multiplataforma	

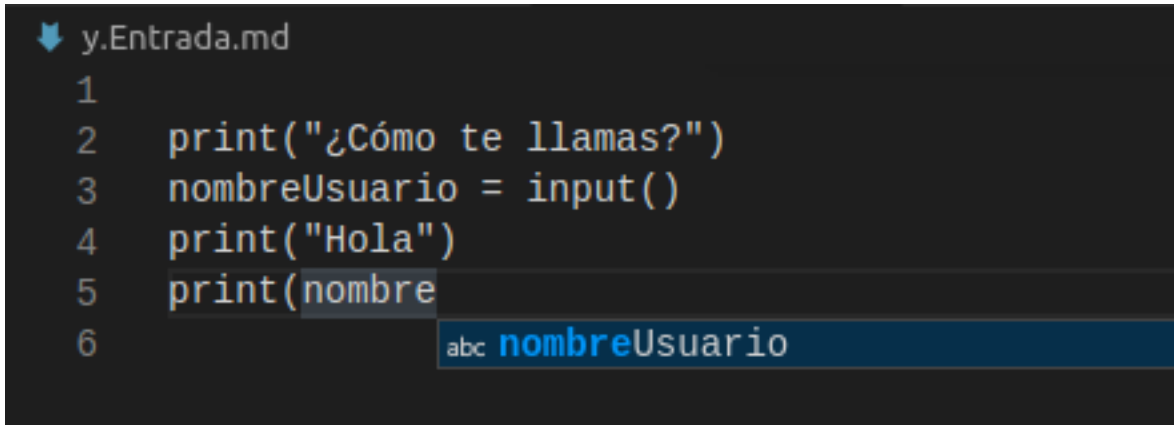
- Lenguajes Compilados: C, C++, Java
- Lenguajes Interpretados: Python, Perl, php, ruby

## ¿Cómo trabajar con python?

Ya que estamos convencidos de la utilidad de Python, vamos a comenzar con su instalación.

Tenemos 3 alternativas para trabajar con python:

- Instalar **sólo** el intérprete y usar cualquier editor de texto puro (no vale un Word o similar porque introducen caracteres de formato y el resultado no lo entendería el intérprete) para escribir nuestros programas, o
- Instalar un Entorno de desarrollo integrado (IDE) que nos facilite el trabajo. Para empezar a programar en serio debemos usar un IDE, que nos aporta ventajas y herramientas que nos facilitan el desarrollo como: correctores, escritura predictiva, depuradores, plantillas, etc.

A screenshot of a code editor with a dark background. At the top left, there is a blue arrow icon followed by the text 'y.Entrada.md'. Below this, there is a list of line numbers from 1 to 6. Lines 2 through 5 contain Python code: line 2 is 'print("¿Cómo te llamas?")', line 3 is 'nombreUsuario = input()', line 4 is 'print("Hola")', and line 5 is 'print(nombre'. Line 6 is empty. A blue horizontal bar is positioned below line 5, containing the text 'abc nombreUsuario', which represents the predictive text suggestions for the next line of code.

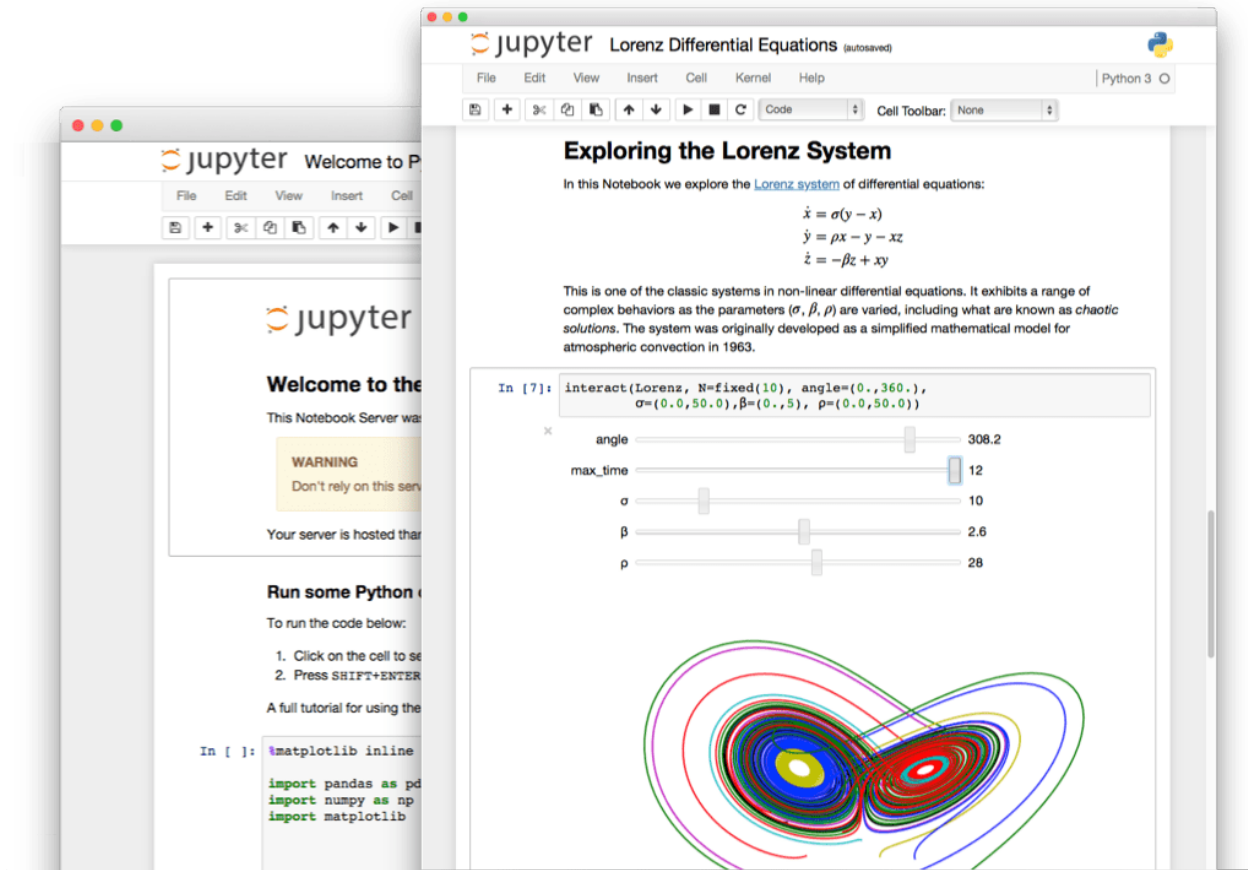
*Ejemplo de escritura predictiva donde el IDE nos muestra las posibilidades para ahorrarnos tener que escribir toda la sentencia, ahorrando tiempo y evitando errores*

- Utilizar un entorno de desarrollo web como [Jupyter Notebook](#) o [Google Colab](#) (que es la integración y evolución de Jupyter con Google Drive) que nos permite trabajar desde un navegador web. Nuestro código se ejecutará en un servidor, en nuestra máquina con Jupyter o en la nube de Google en el caso de Colab. En ambos casos tenemos integrado un gestor de ficheros y recursos y trabajamos con un documento donde se guarda nuestro documento y el texto/documentación de nuestro proyecto.

Para empezar a programar lo más sencillo es un entorno web como Colab o si planeas crear aplicaciones empezar con un entorno sencillo como Thonny que tiene todo lo necesario, pero que no nos abruma con un excesivo número de menús, ni de opciones que no necesitamos.

## Jupyter Notebook

Jupyter es una aplicación Web (la ejecutamos en local, en nuestra máquina pero sobre un navegador web) diseñada para crear al mismo tiempo documentación y código pensando sobre todo en compartirlo posteriormente.



### Jupyter preview

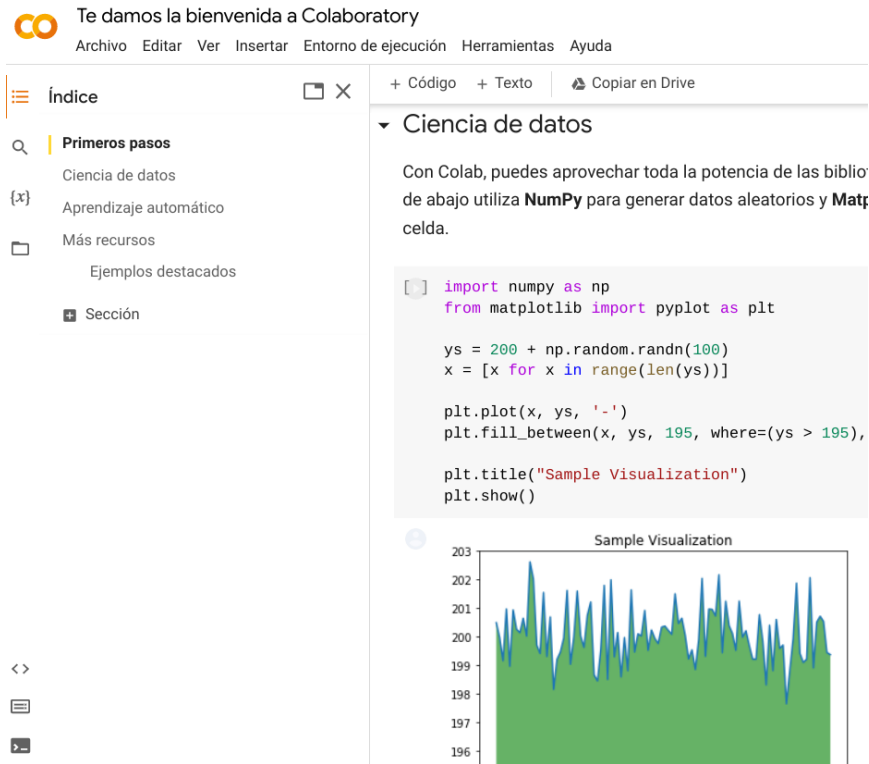
Como se puede ver en la imagen está muy orientada hacia el mundo científico sobre todo para el tema de gráficos, su entorno natural.

## Google Colab

[Google Colab](#) es una adaptación de jupyter notebook por parte de Google y su integración con Google Drive, lo que nos permite compartir fácilmente con otros usuarios y también utilizar toda su infraestructura de la nube. Su nombre viene de Colaboratory lo que nos muestra claramente su vocación de trabajar colaborativamente.

Al igual que Jupyter Notebook, nos permite mezclar texto y código de manera que podemos trabajar en un artículo o en documento interactivo dónde podemos explicar, programar, calcular y al mismo tiempo presentar los resultados. También integra

herramientas de procesamiento y visualización de datos y en general data mining incluso de aprendizaje automático o machine learning.



Otra gran ventaja de utilizar Google colab es que nos permite crear un entorno de ejecución totalmente personalizado para nuestro código Python, independiente de nuestro ordenador. Así podemos trabajar con los componentes o módulos que necesitemos instalados, incluso tenemos acceso a un entorno virtual donde se está ejecutando nuestro código.

Los cuadernos de Colab ejecutan código en los servidores en la nube de Google, lo que te permite aprovechar toda la potencia del hardware de Google, independientemente de la potencia de nuestro equipo.

Un ejemplo típico de uso de Colab donde realmente aprovechamos muchas de sus posibilidades sería el compartir un documento donde ya tenemos preparado todo lo necesario para trabajar con un sistema de inteligencia artificial, con las librerías



preparadas, el código necesario para hacerlo incluso tenemos todas aquellas imágenes o objetos que queremos que nuestra inteligencia artificial aprenda.

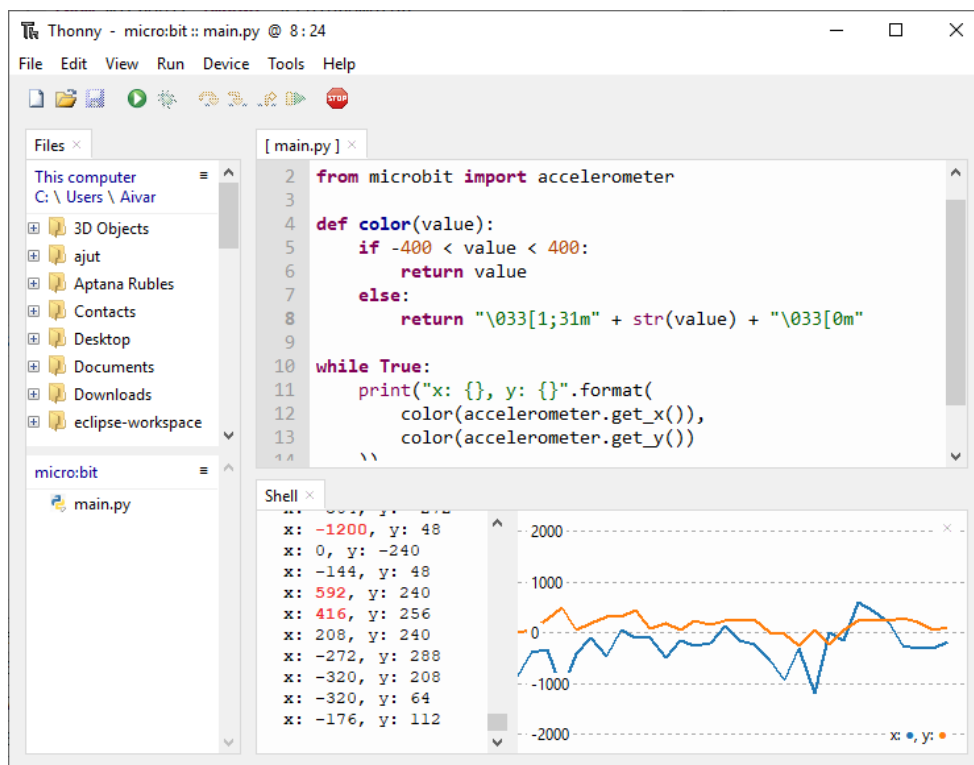
De esa forma, cuando compartimos nuestro documento con texto y con código, se está compartiendo la instalación, los ficheros, las librerías, imágenes, etc, en definitiva todo lo que necesitamos para trabajar, aprovechándose de la infraestructura de Google Drive.

Otra enorme ventaja es que estamos también delegando en los servidores de la nube de Google todo el procesamiento, nuestro ordenador no tiene que realizar aquí ese procesamiento sino que se hace todo directamente en los ordenadores de la nube de Google.

[Vídeo: presentación de Google Colab](#)

## Thonny

[Thonny](#) es un IDE sencillo y está escrito en python! y es Open source. Puedes encontrar [su código fuente en github](#)



## Thonny

El IDE Thonny incluye todo lo necesario para editar y ejecutar nuestros programas, puesto que además de editor incluye el entorno de ejecución de Python.

Por ello y por su facilidad de uso es la elección para quien empieza.

Thonny incluye las características esenciales para desarrollar en Python:

- **Edición de ficheros:** Podemos editar nuestros ficheros de código. También nos permite gestionar archivos con un explorador de ficheros.
- Como vemos **resalta la sintaxis**, lo que nos facilita la lectura.
- También nos ayuda **autocompletando código**, con variables y funciones. Esto nos agiliza la escritura del código y además evita errores.
- En la parte de abajo de la ventana vemos una **consola** que se conoce como **REPL** (Read-Eval-Print Loop) que nos permite trabajar interactivamente con Python
- **Depuración:** Podemos ejecutar nuestro programa de forma normal o en modo depuración donde podremos ejecutar línea a línea.
- **Visualización de variables y sus valores:** Mientras depuramos también podemos abrir la vista de variables que nos va a permitir seguir el valor que tienen las variables en cada momento.
- **Instalación de paquetes y complementos** Desde el propio IDE podemos gestionar e instalar paquetes y complementos, necesarios para nuestros programas.