

# ETS - VisualStudio con WSL en Windows para desarrollo en Python

## Terminal de Windows

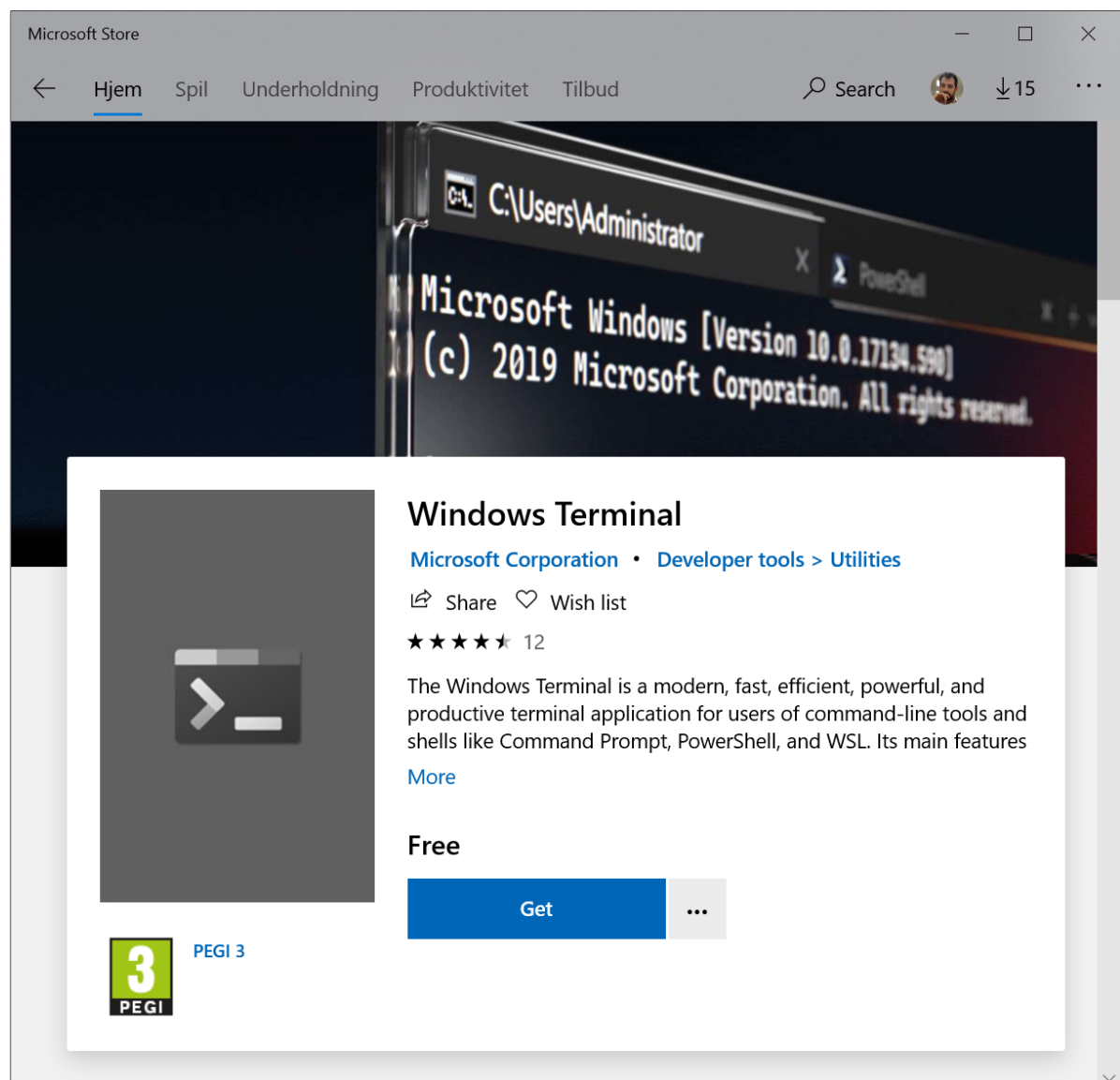
Windows Terminal es una aplicación de Windows que permite acceder a una interfaz de usuario en modo texto enriquecido (con soporte de emojis) y que incluso se beneficia del renderizado de texto vía la GPU.

Esta nueva terminal de Windows unifica a las tres consolas existentes hasta ahora: PowerShell, Cmd y Windows Subsystem for Linux (WSL).

Ofrece soporte para atajos de teclado, pestañas, ventanas podemos separar extensiones y plantillas y temas personalizados.

## Instalación de Terminal de Windows

Accedemos a la tienda de Linux y buscamos **Windows Terminal**



De forma alternativa podemos utilizar **ConEmu** que podemos descargar en <https://conemu.github.io/>

# WSL

Windows Subsystem for Linux (WSL) es una característica opcional de Windows 10 que nos permite instalar un Kernel Linux directamente sobre el sistema operativo de Microsoft.

## Ventajas de WSL

La mayoría de aplicaciones que se desarrolla para la web se ejecutan en servidores Linux, WSL permite disponer desde Windows de un entorno de desarrollo similar al entorno de producción en el que se van a ejecutar dichas aplicaciones.

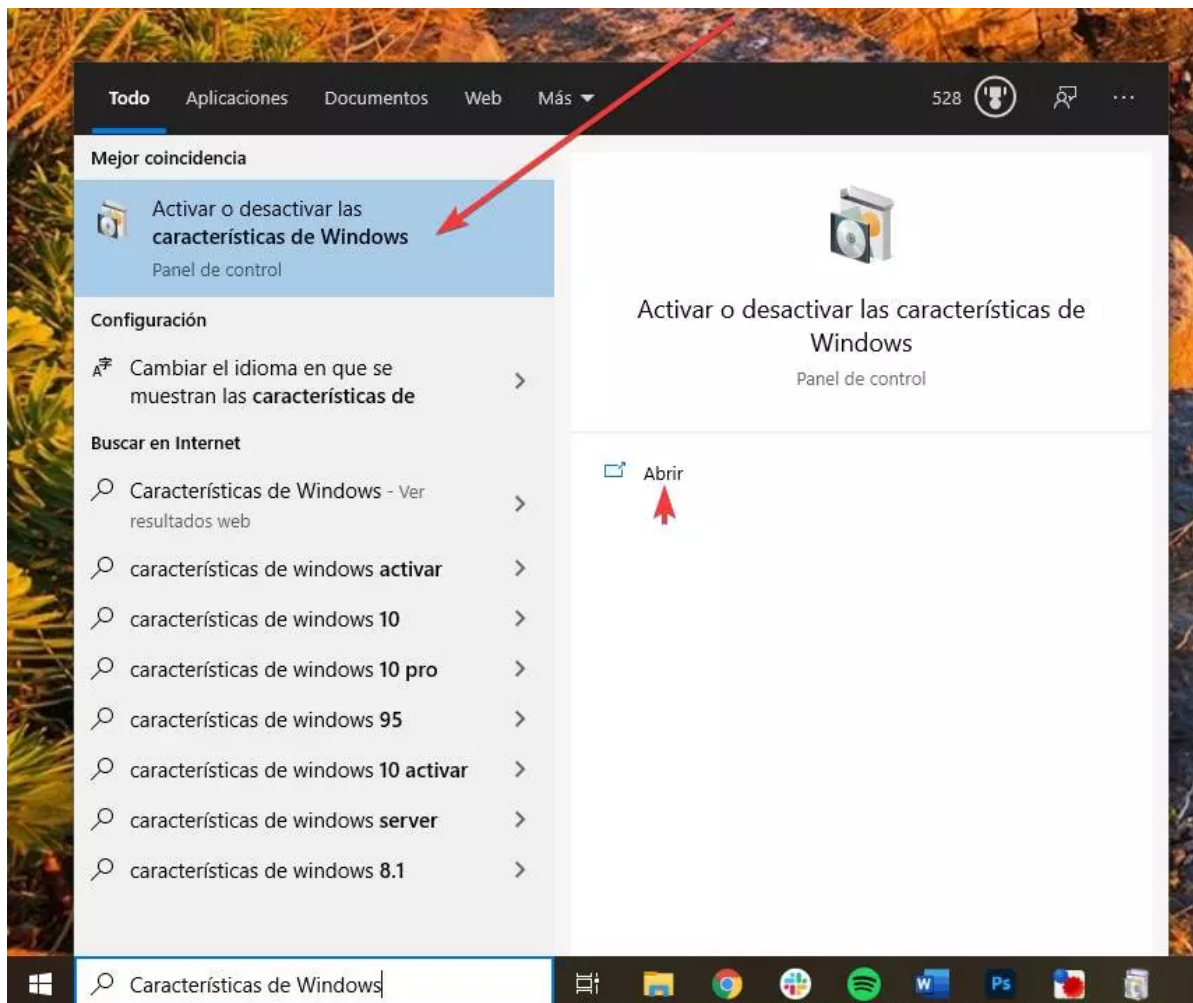
WSL permite a los administradores de sistemas, y a los programadores, usar todas las herramientas y todos los servicios de Linux directamente desde Windows sin tener que usar máquinas virtuales u otro tipo de infraestructuras complejas.

Frente al uso de Máquinas virtuales, las ventajas que ofrece son:

- Tiempos de carga bajos. Se puede cargar Linux en menos de 1 segundo.
- Bajo consumo de recursos.
- Tenemos acceso directo al sistema de archivos de Windows y podemos interactuar directamente con él.

## Instalación de WSL

Podemos activarlo desde la herramienta de **Activar o desactivar las características de Windows**



O desde Powershell en modo administrador ejecutando:

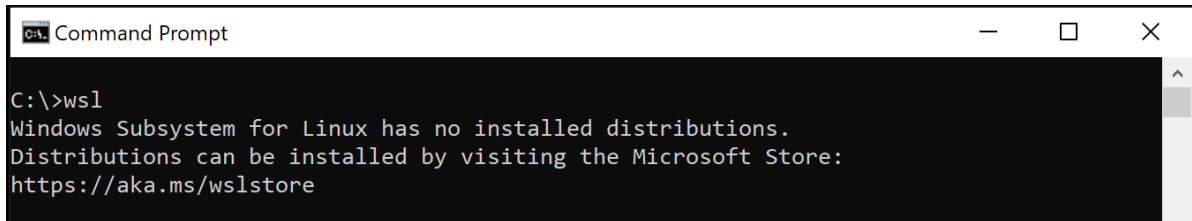
```
Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux
```

Después de ejecutarlo se nos pedirá reiniciar el equipo.

Una vez instalado, si abrimos **CMD** y ejecutamos

```
wsl
```

Se nos informará de que **wsl** está habilitado y no tenemos todavía instalada ninguna distribución de Linux



```
C:\>wsl
Windows Subsystem for Linux has no installed distributions.
Distributions can be installed by visiting the Microsoft Store:
https://aka.ms/wslstore
```

## Descargar Linux en Windows 10

Para instalar una distribución de Linux accedemos a la **Tienda de Windows** y buscamos **Ubuntu**. Seleccionamos **Ubuntu 20.04** y lo instalamos.

Cuando termine instalación abrimos

Nos solicitará un nuevo usuario y contraseña. Podemos poner `usuario` y contraseña `daw1234` Ubuntu y actualizamos ejecutando:

```
$ sudo apt update
$ sudo apt upgrade
```

## Instalación de Visual Studio Code

Descargamos en <https://code.visualstudio.com/>

## Desarrollo en Python

Para instalar Python abrimos un terminal WSL en Ubuntu y ejecutamos:

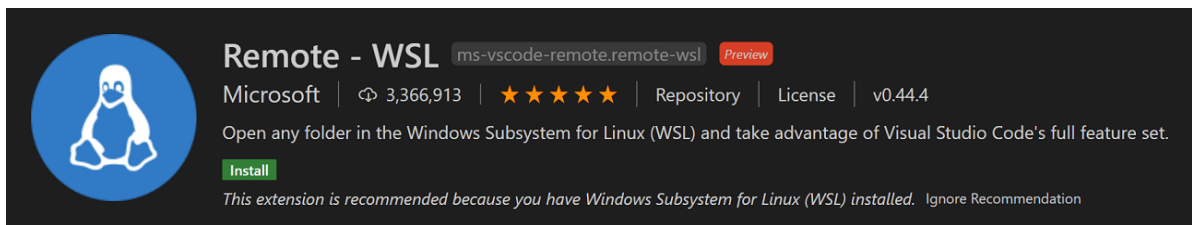
```
$ sudo apt update
$ sudo apt install python3 python3-pip python3-venv
```

Verificamos que se ha instalado ejecutando:

```
$ python3 --version
Python 3.8.10
```

## Configuración de VSCode para usar WSL

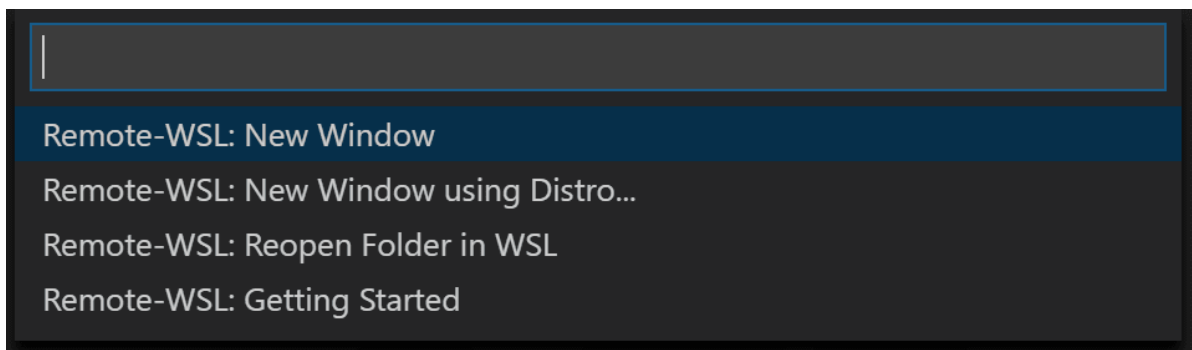
Instalamos la extensión **Remote - WSL** que permite a VSCode trabajar dentro del Windows Subsystem for Linux



Una vez instalado veremos un nuevo icono en la parte izquierda de la barra de estado de VSCode



Haciendo clic en el mismo mostrará comandos asociados a WSL:

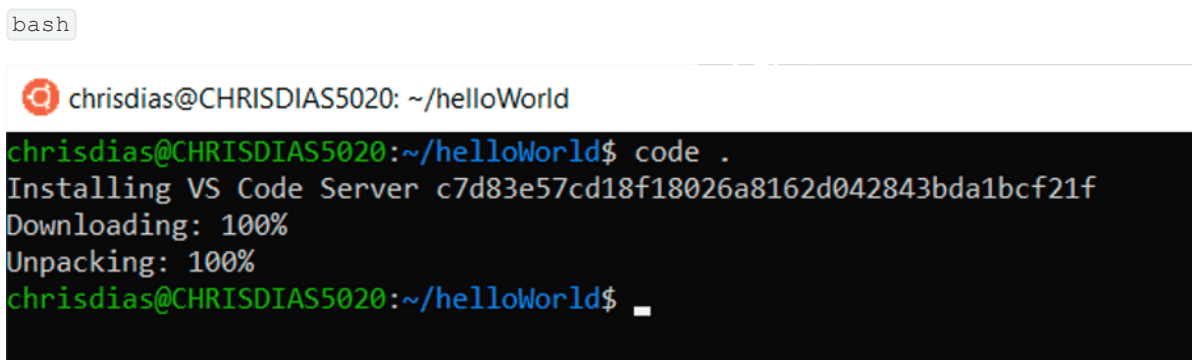


Como Python está instalado en la distribución de Ubuntu asociada a **WSL** para poder ejecutar y depurar archivos de Ubuntu desde Windows accedemos a un terminal de WSL usando **ConEmu** o **Windows Terminal**

Creamos una carpeta en la que vamos a crear nuestra aplicación de prueba y accedemos a la misma:

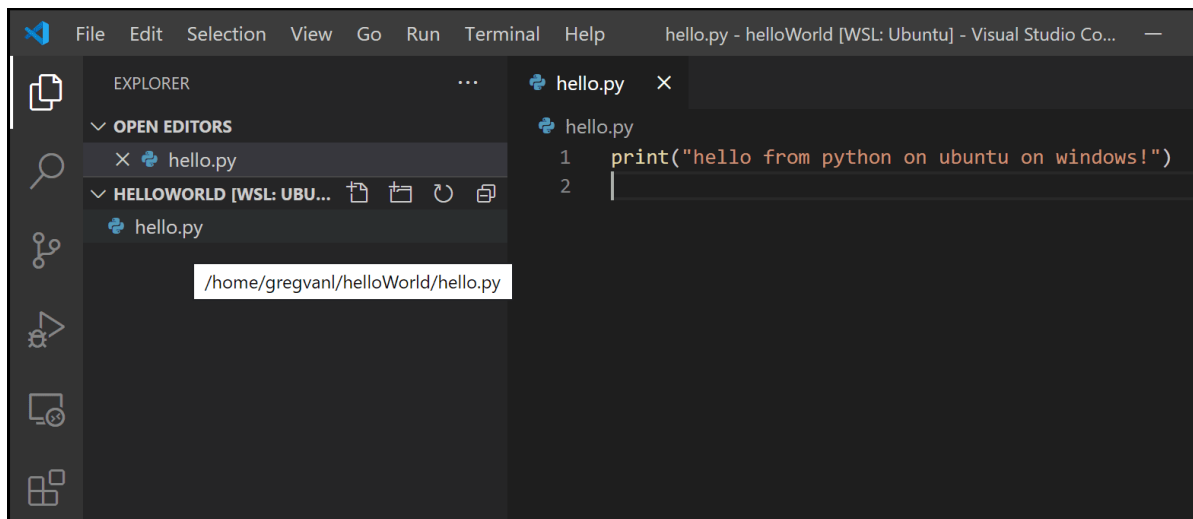
```
$ mkdir helloWorld
$ cd helloWorld
```

Y desde la misma ejecutamos VSCode ejecutando `code .`. El "." equivale a la carpeta actual en



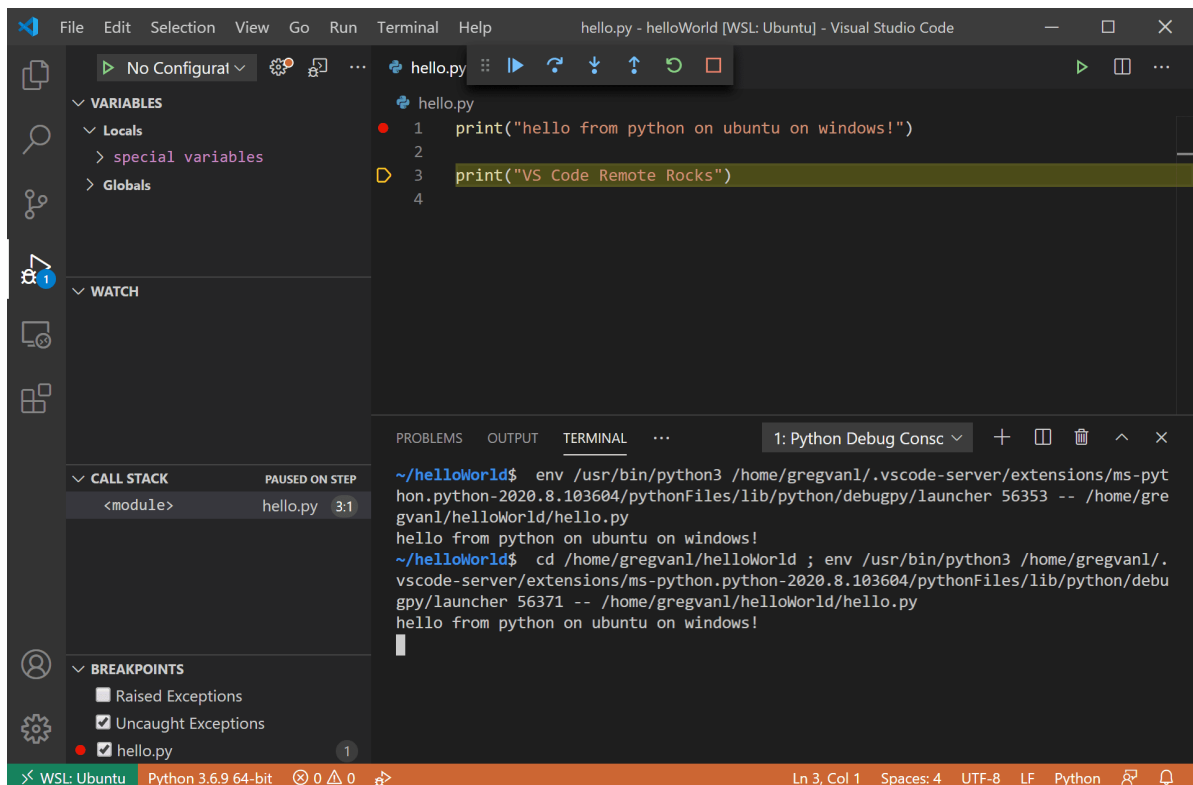
Se descargará e instalará un **servidor de code** en Linux con el que se comunicará **VScode** a la hora de depurar y ejecutar nuestros programas.

Creamos un nuevo archivo `hello.py` en la carpeta que acabamos de crear:



**VScode** nos preguntará si queremos instalar la extensión que da soporte para Python (si no la tenemos ya instalada).

A partir de ese momento VSCode comprobará la sintaxis del programa que escribamos mostrándonos errores, aplicará colores a las sentencias y elementos del programa, nos permitirá depurar y ejecutar el programa. En conclusión, dispondremos de las mismas herramientas de desarrollo que en nuestro sistema operativo Ubuntu de clase.



Para cerrar la conexión de VSCode con WSL y trabajar de modo local, accedemos a **Archivo > Cerrar conexión remota**

## Recursos

- [Activar WSL en Windows-software.com](https://www.windows-software.com/activar-wsl-en-windows/)
- [Desarrollo en Python usando WSL2-mjlivesey.co.uk](https://www.mjlivesey.co.uk/development/python-using-wsl2/)
- [Desarrollo remoto en WSL con VScode](https://code.visualstudio.com/docs/python/remote)

tags: ets ut2 windows wsl