DOCKER & WSL

Subsistema Nativo de Linux en Windows

Trabajando con Docker en WSL2

¿Qué es WSL2 y por qué es importante para Docker?

WSL 2 (Windows Subsystem for Linux 2) es una capa de compatibilidad desarrollada por Microsoft que permite ejecutar binarios ELF de Linux de forma nativa en Windows 10, Windows 11 y versiones de servidor, utilizando una máquina virtual (VM) ligera y un kernel de Linux real ajustado por Microsoft.

WSL 2 es la arquitectura predeterminada para las nuevas instalaciones de distribuciones de Linux en Windows y está diseñada para ofrecer una experiencia fluida y productiva a los desarrolladores, permitiendo usar Windows y Linux simultáneamente.

¿Qué es WSL2 y por qué es importante para Docker?

WSL 2 es fundamental para **Docker Desktop en Windows** por varias razones clave:

- **Rendimiento y arquitectura:** A diferencia de su predecesor (WSL 1), que utilizaba una capa de traducción, WSL 2 incluye un **kernel de Linux real**. Con esto ejecuta los contenedores de Linux de forma **nativa**, sin emulación.
- **Compatibilidad total de llamadas al sistema:** El kernel real de Linux en WSL 2 ofrece **compatibilidad total con las llamadas al sistema**, algo esencial para las aplicaciones y herramientas de Linux como Docker.
- **Integración simplificada:** Docker se integra de manera sencilla con WSL 2, haciendo que el inicio y la ejecución del *daemon* de Docker sean **mucho más rápidos** y con un **menor consumo** de CPU y memoria que los métodos de virtualización tradicionales.

Diferencias entre WSL1 y WSL2

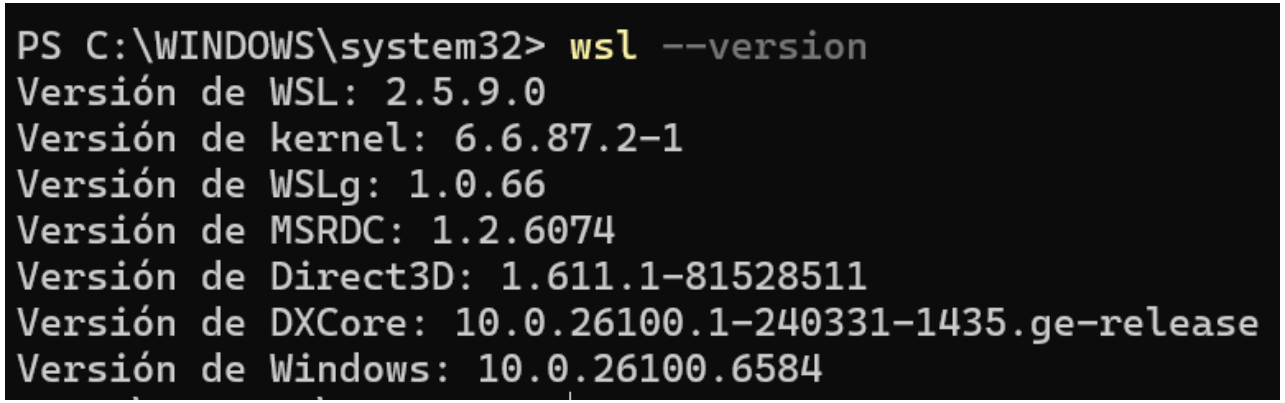
Característica	WSL 1	WSL 2 (Recomendado)
Arquitectura	Capa de traducción de llamadas del sistema.	**Máquina virtual (VM) ligera gestionada.**
Kernel Linux	**No** usa un kernel Linux real.	**Usa un kernel Linux real** (ajustado por Microsoft).
Compatibilidad	Compatibilidad limitada (puede fallar con algunas aplicaciones).	**Compatibilidad total con llamadas al sistema** (ideal para Docker).
Rendimiento I/O	**Acceso rápido** a archivos de **Windows** desde Linux (/mnt/c).	**Rendimiento mucho más rápido** en los archivos del **sistema Linux** (ext4).
Contenedores	Requiere una VM tradicional para Docker o similar.	**Integración nativa y rápida con Docker Desktop.**
Uso de Recursos	Utiliza menos RAM.	Utiliza más RAM (pero es **configurable** y se libera al cerrar).

Arquitectura general

Windows -> WSL2 -> Docker Engine -> Contenedores

Configuración del Entorno

- Verificar versión de Windows compatible (Windows 10/11 Pro o Home actualizada).
  - wsl –versión
  - Debe mostrar algo similar a la imagen
  - Si dice **“WSL is not recognized”**, significa que se tiene una versión antigua o no instalada.
- Instalar o actualizar WSL a la última versión
  - Abrir Powershell y ejecutar el siguiente comando:
    - wsl --update



Configuración del Entorno

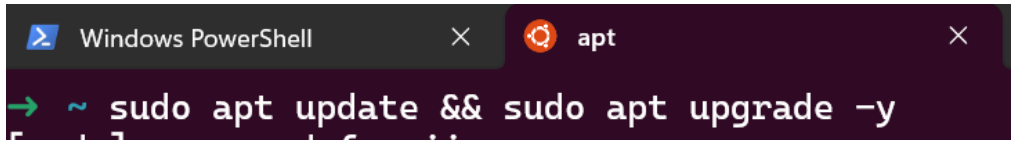
- Si no se había instalado antes, se puede ejecutar el comando:
  - wsl --install
- Para especificar una distribución, por ejemplo, Ubuntu, usar el comando:
  - wsl --install -d Ubuntu
- Después de la actualización o instalación, ejecutar:
  - wsl –status

```
PS C:\WINDOWS\system32> wsl --status
Distribución predeterminada: Ubuntu-22.04
Versión predeterminada: 2
```

- Si dice **“Default Version: 1”**, se debe cambiar a WSL2 con:
  - wsl --set-default-version 2

Actualizar distribución (Ubuntu u otra)

- En el botón inicio buscar Ubuntu, se abrirá la terminal con la distribución instalada.
- Al abrir la terminal de **Ubuntu** dentro de WSL se debe actualizar el sistema Linux:
  - sudo apt update && sudo apt upgrade –y



- Opcionalmente, se puede actualizar el kernel de Ubuntu:
  - sudo apt dist-upgrade -y
- Reiniciar WSL para aplicar los cambios
  - Desde PowerShell:
    - wsl --shutdown
  - Luego reiniciar
    - wsl

Ver lista de distribuciones instaladas

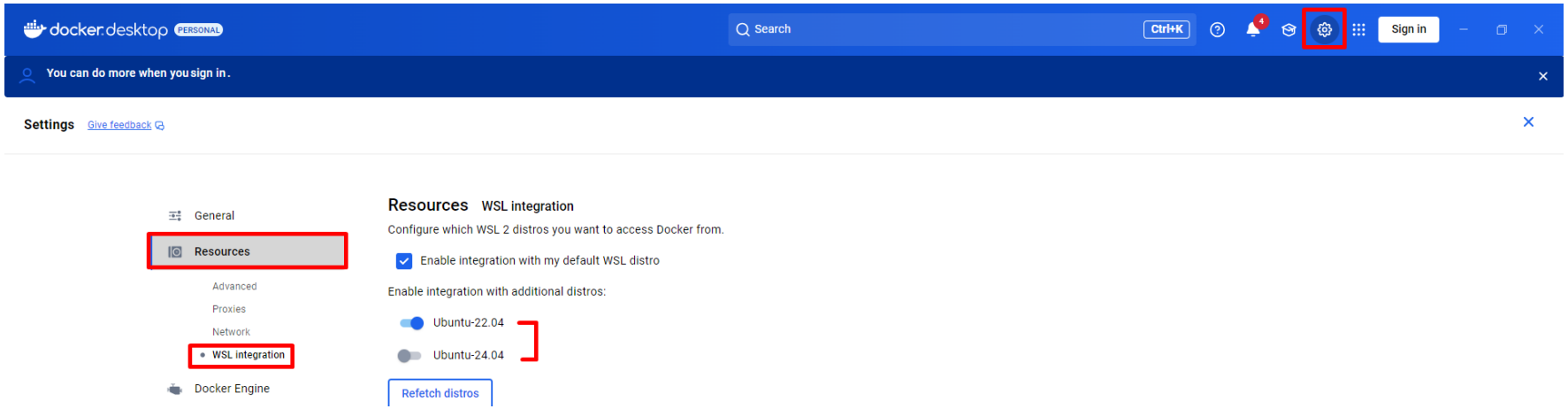
- Si se tienen varias distribuciones se pueden listar con el comando:
  - wsl --list –verbose

```
PS C:\WINDOWS\system32> wsl --list --verbose
  NAME                   STATE           VERSION
* Ubuntu-22.04           Running         2
  Ubuntu-24.04           Stopped         2
  docker-desktop         Stopped         2
```

- Si se quiere cambiar la versión de una distro específica:
  - wsl --set-version Ubuntu 2

Verificar que Docker usa WSL correctamente

- Abrir Docker Desktop → Settings → Resources → WSL Integration
- Activar la distribución (por ejemplo, Ubuntu).
- Asegurarse de que diga “Docker Engine is running”.



- En la terminal Ubuntu:
  - docker versión
- Deberían verse el Client y Server activos.

```
→ ~ docker version
Client: Docker Engine - Community
 Version:           28.5.1
 API version:       1.49 (downgraded from 1.51)
 Go version:        go1.24.8
 Git commit:        e180ab8
 Built:             Wed Oct  8 12:17:03 2025
 OS/Arch:           linux/amd64
 Context:           default

Server: Docker Desktop 4.0.0 ()
 Engine:
  Version:          28.1.1
  API version:      1.49 (minimum version 1.24)
  Go version:       go1.23.8
  Git commit:       01f442b
  Built:            Fri Apr 18 09:52:57 2025
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.7.27
  GitCommit:        05044ec0a9a75232cad458027ca83437aae3f4da
 runc:
  Version:          1.2.5
  GitCommit:        v1.2.5-0-g59923ef
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0
```

Resumen de comandos

Acción	Comando
Ver versión	`wsl --version`
Actualizar WSL	`wsl --update`
Instalar WSL	`wsl --install -d Ubuntu`
Ver estado	`wsl --status`
Establecer WSL2 por defecto	`wsl --set-default-version 2`
Apagar WSL	`wsl --shutdown`
Listar distros	`wsl --list --verbose`

Primeros comandos Docker en WSL

- docker version
- docker run hello-world
- docker ps
- docker images

Crear y ejecutar una app PHP o NGINX simple

- Crear un directorio llamado wsl-php
  - mkdir wsl-php
- Ingresar al directorio
  - cd wsl-php
- Crear un archivo index.php
  - nano index.php
- Escribir el siguiente código:
  - <?php echo "¡Hola desde Docker + WSL2!"; ?>
- Guardar el archivo:
  - Presionar la tecla CTRL + X
  - Luego colocar la letra Y, y presionar Enter

Crear y ejecutar una app PHP o NGINX simple

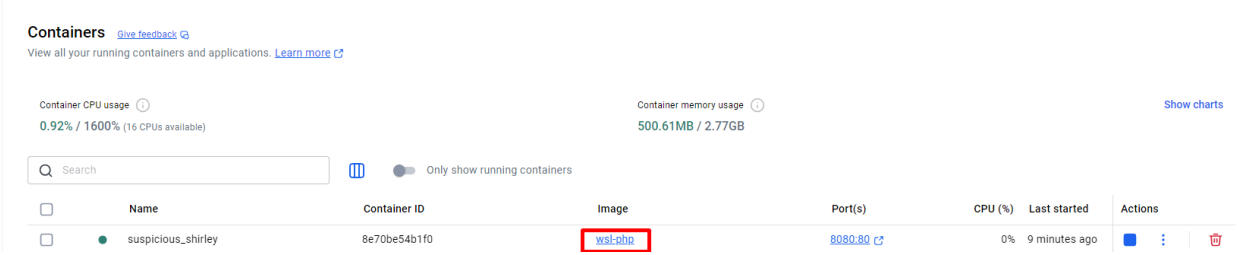
- Crear un Dockerfile
  - nano Dockerfile
- Agregar código:

```
FROM php:8.2-apache
COPY . /var/www/html
EXPOSE 80
```

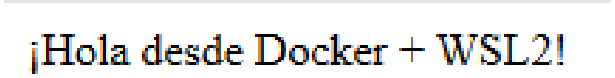
- Guardar Dockerfile
- Guardar el archivo:
  - Presionar la tecla CTRL + X
  - Luego colocar la letra Y, y presionar Enter

Crear y ejecutar una app PHP o NGINX simple

- Construir y ejecutar:
  - docker build -t wsl-php .
  - docker run -d -p 8080:80 wsl-php
- El contenedor debe estar ejecutándose



- Abrir navegador
  - http://localhost:8080



Desplegar Imagen DockerHub

- docker pull jamescanos/php-simple-app:latest
- docker run -d -p 8090:80 jamescanos/php-simple-app:latest
- Abrir el navegador (Se debe mostrar el archivo funcionando)
  - http://127.0.0.08090/php-simple-app/src

Taller en Clase

Desplegar en Ubuntu con WSL una imagen con un proyecto propio que previamente haya sido desplegada en DockerHub.

Taller Independiente

Como punto adicional quien despliegue el proyecto del segundo corte en WSL tendrá una valoración adicional para la nota final del segundo corte.

GRACIAS

Corporación
de Estudios
Tecnológicos
del Norte del Valle