Docker Installation Guide for Windows with WSL 2

Introduction

This guide details the steps for installing Docker on Windows using WSL 2 (Windows Subsystem for Linux version 2) and Docker Desktop, catering to users requiring a Docker containerization environment.

System Requirements

- Windows 10 64-bit: Pro, Enterprise, or Education (Build 18363 or higher)
- BIOS support for hardware virtualization

Installation Steps

Enable WSL 2 and Virtual Machine Feature

1. **Enable the Virtual Machine Platform**:

   ```
   dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart
   ```

2. **Open PowerShell as Administrator** and enable the WSL feature:

   ```
   wsl --install
   ```

   ```
   To check the version, try
   ```

   ```
   wsl -v
   ```

   3. Restart your PC to apply the changes.

Set WSL 2 as the default.

- **Open PowerShell as Administrator** and set WSL 2 as the default version:

  ```
  wsl --set-default-version 2
  ```

Download Additional Linux Distributions (Optional)

- Obtain further Linux distributions via the Microsoft Store (e.g., Ubuntu, Debian).

Install Docker Desktop for Windows

1. **Download Docker Desktop** from Docker Hub.
2. **Run the installer** and follow the instructions, ensuring to select the 'WSL 2 based engine' option.
3. **Launch Docker Desktop** after installation.

Verify Docker Installation

- **Open PowerShell** and check the Docker installation with docker --version and docker-compose --version.

Post-Installation Steps

After successfully installing Docker Desktop, it's essential to configure it appropriately for optimal performance and integration with WSL 2 and your project needs.

Step 1: Configure Docker Desktop Settings

1. **Launch Docker Desktop**: Open Docker Desktop from the Start menu.
2. **Adjust Resources**:
   - **Go to 'Settings' > 'Resources'.**
   - **Allocate CPUs, memory, and disk space as required for your project and system capabilities.**
3. **WSL 2 Integration**:
   - **In 'Settings', navigate to 'Resources' > 'WSL Integration'.**
   - **Here, you'll see a list of available WSL 2 distributions. Enable Docker integration for your installed distributions. This allows you to run Docker commands from these Linux distributions.**
4. **Network Configuration**:
   - **Still in 'Settings', go to the 'Network' tab.**
   - **Configure the network settings as needed for your Docker containers. This may include setting up proxies, DNS servers, or network subnets, particularly important for complex projects involving multiple networked containers.**

Step 2: Verify Docker Engine is Running

- Check if the Docker engine is running by looking for the Docker icon in the system tray. It should indicate that Docker is operational.
- **Test Docker Functionality**:
  - Open PowerShell and run a test Docker command, like docker run hello-world. This command pulls a test image from Docker Hub and runs it in a container, confirming that Docker is set up correctly.

Step 3: Docker Hub Account

- **Sign in to Docker Hub** (optional but recommended):
  - Click on the Docker Desktop icon in the system tray and select 'Sign in / Create Docker ID'.
  - Sign in with your Docker Hub account to access a broader range of Docker images and repositories. If you don’t have an account, consider creating one for additional features and repository access.

Step 4: File Sharing Configuration

- **Configure File Sharing**:
  - In Docker Desktop settings, go to 'Resources' > 'File Sharing'.

  - Add any directories from your Windows system that you want to access from within Docker containers, essential for projects that require access to local files.

Additional Resources

For a more detailed guide and troubleshooting, refer to
(19) Step by Step guide on How to install Docker on Windows 10/11 | LinkedIn

Tutorial for Docker on Windows. Docker is an open platform for… | by heryanto liao | Bina Nusantara IT Division | Medium