Narration&Reference How to use VS Code Extension for Docker

Base File Name: NarrationReference_Docker_VSCodeExtension_Angular7_ja_08_en

2018.11.14

How to use VS Code Extension for Docker

by ***Shuichi Ohtsu***

Operating environment

This time, I will introduce the VS Code extension for Docker.

Also, while using this extension, we are going to create an Angular7 Docker image.

As a server, we are going to use Nginx running on alpine Linux, not Angular's built-in server.

In consideration of such an environment, we are going to use Docker running on Ubuntu on WSL (Windows Subsystem for Linux) instead of *Docker for Windows* this time.

If you have not installed Docker on WSL please see another video in advance.

Install Visual Studio Code extension for Docker

First, open Ubuntu on the WSL.

Next, at the command line, type `code .` to open the Visual Studio Code.

When the VS Code opens, click the extension icon on the left side and enter `Docker` in the search box.

Then, the extension *Docker* made by Microsoft is displayed in the search result list, so click *install*.

When the explanation page of *Docker* is displayed, click *Reload and Activate*.

The *Docker* extension has been activated.

Close the VS Code here.

Generation of Angular 7 for testing

Here we are going to create a project for Angular7 for testing.

We are going to check the operating environment.

First of all, on the command line, type `ng --version` to check the version of Angular and the version of Node.js.

The version of Angular is 7.0 and the version of Node.js is 10.12. Please note this version of Node.js will become important later.

Next, we are going to create a test project for Angular7.

Type `ng new ng7-initial`.

ng7-initial is the project name, it can be another name.

First of all, there is a question as to whether or not to add a routing function, so let's say `Y` here.

Next, since there is a question of style sheet format, here we are going to select `SCSS`.

The installation will start.

This process takes some time.

When installation is completed, go to the project directory and type `code .` to start the Visual Studio Code.

When the VS Code starts up, open the command line window with `control + ` key.

Then type `ng s -o` to start the local server.

Angular's default page is displayed.

It is OK.

Return to the VS Code and stop the local server with `control + c`.

Create Dockerfile

Next, we are going to create *Dockerfile* to copy this project on Nginx running on Alpine Linux.

The structure of this *Dockerfile* is in two stages.

First of all, install Angular 7 on Node.js, compile this project in production mode there, and generate compressed JavaScript for distribution.

Next, generate Nginx on Alpine Linux and copy this compressed JavaScript along with the necessary Node.js into its homepage directory.

Here, we are going to slightly change the contents of the default page in order to clarify that it is a page operated by Nginx.

Create Docker Image

Next, we are going to create a Docker image, but thanks to the *Docker extension* this operation has become quite easy.

Display the Dockerfile with VS Code, right-click in it, and display the pop-up menu.

Then just select `Build Image`.

Since there is an inquiry about the name of the image to be generated, let's set `ng7-initial:1.0` here.

":" the following specifies the version.

Generation will start.

This process takes some time.

An error occurred on the way.

According to the error message, in the latest Node.js, there seems to be a problem when *SCSS* is selected in the style format.

So, change to the version of Node.js that was checked for startup first.

That version is `10.12`.

Correct the Docker file and generate the Docker image again.

This time it is OK.

Now check if the Docker image was generated.

Click the Docker icon on the left side of the VS Code.

Then select `Images`.

Then, you can see that `ng7-initial` is generated indeed.

Run Docker Image

Next, we are going to try starting this image locally.

From the command line on the VS Code, enter `docker run --rm -d -p 80:80 ng7-initial:1.2`.

Each specified option is

--rm means automatically deletes the container when the container exits

-d means run container in background

-p means port number accessed from outside : port number on the container side

Since Nginx uses 80 ports internally, it means that it is made accessible to the outside with 80 ports as it is.

Here, we are going to confirm the operation by the browser.

Since it is made to be accessible to the outside by 80 ports,

`http://localhost`

You should be able to access it as it is.

We were able to confirm the operation of Angular 7 and Nginx on Docker.

It is OK.

Thank you for your watching.

Reference

- "Docker Community Edition for Windows", https://store.docker.com/editions/community/docker-ce-desktop-windows
- "Docker/Kubernetes 実践コンテナ開発入門", http://amazon.co.jp/o/ASIN/4297100339/
- "プログラマのためのDocker教科書 第2版 インフラの基礎知識&コードによる環境構築の自動化", http://amazon.co.jp/o/ASIN/4798153222/
- "Containerizing Angular with Docker - Dan Wahlin", https://www.youtube.com/watch?v=cLT7eUWKZpg&t=1140s

- "Deploy Angular 5 app in Docker Container in under 10 mins - For local development", https://www.youtube.com/watch?v=L2UkQ2CND68&t=178s
- "Angular5, Angular6, Angular7 Custom Library: Step-by-step guide", https://www.udemy.com/angular5-custom-library-the-definitive-step-by-step-guide/
- "Angular5, Angular6, Angular7用 カスタムライブラリの作成: 完全ステップ・バイ・ステップ・ガイド", https://www.udemy.com/angular5-l/