

1. Python

Відтворити експерименти можна за [посиланням](#)

1. [python.Dockerfile](#)

Розмір образу **1.13GB**

Час збірки початкового образу **27.3s**

Час збірки початкового образу (із завантаженням основним слоєм) **16.0s**

2. [python.Dockerfile](#)

Розмір образу **1.13GB**

Час збірки початкового образу **3.0s**

3. [3.python.Dockerfile](#)

Розмір образу **1.13GB**

Час збірки початкового образу **16.6s**

4. [4.python.Dockerfile](#)

Розмір образу **160MB**

Час збірки початкового образу **14.2s**

5.

Alpine

Розмір образу **328MB**

Час збірки початкового образу **25.4s**

Bookworm

Розмір образу **1.29GB**

Час збірки початкового образу **24.8s**

2. Golang

Відтворити експерименти можна за [посиланням](#)

1. [1.go.Dockerfile](#)

Розмір образу **337MB**

Час збірки початкового образу **29.7s**

Проект запустився без жодних проблем.

2. [2.go.Dockerfile](#)

Розмір образу **10.7MB**

Час збірки початкового образу **1.6s**

Проект не запустився через відсутність файлу index.html, [виправив](#) скопіювавши цей файл у Scratch.

Виконувати дії тяжко, адже sh, bash, fish, zsh не встановлені у базовому image.

3. [3.go.Dockerfile](#)

Розмір образу **13.3MB**

Час збірки початкового образу **3.7s**

Виконувати дії тяжко, адже як і scratch це пуста оболонка, хоча на 2.6мб важча.

3. [Nodejs](#)

Відтворити експерименти можна за [посиланням](#)

Розмір образу **168MB**

Час збірки початкового образу **15.5s'**

Так як nodejs інтерпретована, я не бачу сенсу у багатоетапній збірці в такому форматі, в якому вона була у п.2.

Я вибрав alpine, хоча можна було б щось поменше, як gcr.io/distroless/nodejs20-debian12, але мені зручніше, коли є shell і можливість увійти подивитися, що відбувається всередині, також 168мб - не такий і великий розмір контейнера.

4. Висновок

Отже, я створив dockerfiles. На такому простому проєкті зіштовхнувся із складнощами із залежностями

1. Python

- a. Залежність pydantic перенесла залежність в інший модуль. Я це виправив [знизивши](#) версію залежності і позначивши версії в backend.in

- b. Версія python

Пробував використовувати python:13.... Але не всі версії ще встигли мігрувати і pip не міг встановити wheels for httptools.

На версії python:11 (якщо не помиляюся), одна із залежностей вимагала rust, який займає десь 1gb.

На версії 12 все чудово працювало.

2. Golang

По суті єдина проблема була в тому що в scratch образ я спочатку не скопіював index.html

Я вже знайомий з docker тому кешування слоїв намагався використовувати по максимуму. Вперше почув про багатоетапну збірку, але не для компільованих мов не можу уявити як її можна використати.

Все залежить від проєкту, але я рекомендував би використовувати легкий базовий образ, як alpine, тому що образ в +1GB це вже занадто. Щоб у ньому був встановлений shell.

Наймовірно важливо використовувати кешування.

Завжди бути прискіпливим до версій залежностей, та завжди вказувати принаймні до minor версії.