Metodyki DevOps

Jakub Jezierczak Informatyka Techniczna III rok

Sprawozdanie

Wyszukano i sklonowano repozytorium wykorzystujące meson jako narzędzie typu Makefile. https://github.com/sandervalstar/meson\\_test

![](RackMultipart20220407-4-11ks7c2\_html\_e0b77781ffe98663.png)

Konieczne było zainstalowanie narzędzia meson: ![](RackMultipart20220407-4-11ks7c2\_html\_9d077fefbbdbc59d.png)

Z jego pomocą możliwe było zbudowanie projektu i wykonanie testów.

![](RackMultipart20220407-4-11ks7c2\_html\_1a276ed2e8c47f51.png) ![](RackMultipart20220407-4-11ks7c2\_html\_8753f8e75f582ff8.png) ![](RackMultipart20220407-4-11ks7c2\_html\_1f2a8de227895bab.png)

Kolejnym krokiem było zdalne połączenie się z maszyną wirtualną wykorzystując ssh (analogicznie do poprzednich zajęć). Uruchomiono kontener z obrazem Debiana, sklonowano repozytorium, jednak zbudowanie go i wykonanie testów było niemożliwe.

![](RackMultipart20220407-4-11ks7c2\_html\_be60b3ba69c1b75b.png) ![](RackMultipart20220407-4-11ks7c2\_html\_51aadba9f2643ad5.png)

Konieczne było doinstalowanie gita, mesona, gcc i build-essential.

![](RackMultipart20220407-4-11ks7c2\_html\_ecf4d078d18bf704.png)

Po czym udało się wykonać pożądane operacje

Build:

![](RackMultipart20220407-4-11ks7c2\_html\_359206b2c400f2d1.png) ![](RackMultipart20220407-4-11ks7c2\_html\_3ade29420d7cfc4.png)

I testy:

![](RackMultipart20220407-4-11ks7c2\_html\_ed59b993717939c3.png)

Kolejnym krokiem było stworzenie dwóch plików Dockerfile, gdzie pierwszy miał buildować projekt a drugi będący jego kontynuacją wykonywał testy. Umożliwiło to automatyzację całego procesu. ![](RackMultipart20220407-4-11ks7c2\_html\_3d8f2c342bbaca6f.png)

![](RackMultipart20220407-4-11ks7c2\_html\_68eba8e7a9057b69.png)

Zbudowano pierwszy kontener:

![](RackMultipart20220407-4-11ks7c2\_html\_f26276aa1a947c24.png)

I uruchomiono:

![](RackMultipart20220407-4-11ks7c2\_html\_3d8f2c342bbaca6f.png)

Wykonano testy:

![](RackMultipart20220407-4-11ks7c2\_html\_a97a6ed16092ee3f.png)

Kod pierwszego dockerfile&#39;a:

![](RackMultipart20220407-4-11ks7c2\_html\_7e8f2f033feb918d.png)

A następnie zbudowano i uruchomiono drugi kontener:

![](RackMultipart20220407-4-11ks7c2\_html\_4b040b67eafeb570.png)

Poprawnie zbudowany i uruchomiony kontener przeszedł testy i uzyskał własne id, różne od obrazu.

![](RackMultipart20220407-4-11ks7c2\_html\_6f9b2fc879385c57.png)

Kontener pozwala korzystać z wcześniej określonych zasobów dzięki temu, jest tworzony w oparciu o obraz. Dzięki temu każdy kontener stworzony z konkretnego obrazu zawiera te same biblioteki, programy, pakiety jednak jest odizolowany.