Konetener deweloperski OpenCL

Instalacja

Działanie kontenera zostało zweryfikowane dla systemów operacyjnych Linux oraz Windows. Niemniej jednak, istnieje wersja Docker na system operacyjny MacOS, w związku opisywany kontener powinien także działać na urządzeniach Apple. Poniżej opsiano proces instalacji niezbędnych narzędzi potrzebnych do uruchomienia kontenera na Ubuntu oraz Windows.

Linux

Opisany proces przeprowadzono na systemach Ubuntu 22.04 oraz 24.04. Jednakże zakładając, że w systemie znajdują się aktualne sterowniki karty graficznej, kontener powininen działać na systemach Ubuntu 20.04 i 18.04.

Instalacja Docker Engine

1. Dodanie repozytorium Docker do `apt` .

```
# Dodanie oficjalnego klucza GPG Docker:
sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -
sudo chmod a+r /etc/apt/keyrings/docker.asc

# Dodanie repozytorium do apt:
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/k
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
```

2. Instalacja niezbędnych paczek.

```
sudo apt-get install docker-ce docker-ce-cli containerd.io doc
```

3. Weryfikacja działania Docker Engine po instalacji. W celu weryfikacji pomyślności instalacji, można uruchomić obraz `hello-world` z oficjalnego repozytorium Docker:

```
sudo docker run hello-world
```

Instalacja NVIDIA Container Toolkit

Aby umożliwić uruchamianym kontnerom wykorzystanie GPU firmy NVIDIA znajdującego się w systemie hosta, konieczne jest zainstalowanie NVIDIA Container Toolkit. Tak jak w przypadku docker engine, może to zostać wykonane z użyciem repozytorium `apt` .

1. Dodanie repozytorium NVIDIA Container Toolkit do `apt` .

```
curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey
&& curl -s -L https://nvidia.github.io/libnvidia-container/stab
 sed 's#deb https://#deb [signed-by=/usr/share/keyrings/nvidia-
 sudo tee /etc/apt/sources.list.d/nvidia-container-toolkit.list
```

2. Aktualizacja repozytorium i instalacja paczek.

```
sudo apt-get update
sudo apt-get install -y nvidia-container-toolkit
```

3.

Konfiguracja Docker Aby Docker mógł korzystać z NVIDIA Container Toolkit konieczne jest zmodyfikowanie pliku `/etc/docker/daemon.json` w systemie hostującym. Nałatwiej to zrobić poprzez `nvidia-ctk`.

```
sudo nvidia-ctk runtime configure --runtime=docker
sudo systemctl restart docker
```

4. Ostatnim krokiem jest umożliwienie działania Dockera bez uprawnień root'a.

```
nvidia-ctk runtime configure --runtime=docker --config=$HOME/.c
systemctl --user restart docker
sudo nvidia-ctk config --set nvidia-container-cli.no-cgroups -
```

5. Weryfikacja instalacji NVIDIA Container Toolkit. Działanie Docker Engine ze wsparciem dla GPU poprzez NVIDIA Container Toolkit, można zweryfikować uruchamiając prosty, testowy kontener i wywołując w nim polecenie `nvidia-smi`.

```
sudo docker run --rm --runtime=nvidia --gpus all ubuntu nvidia
```

Jeżeli proces instalacji przebiegł pomyślnie, to na wyjściu powinien pojawić się raport dot. zainstalowanych sterowników karty graficznej, modelu GPU oraz zużycia jego zasobów, który wygląda podobnie do tego:

```
 +
-----------------------------------------------------------------------------+
 | NVIDIA-SMI 552.22                 Driver Version: 552.22
CUDA Version: 12.4     |
 |-----------------------------------------+
------------------------+----------------------+
 | GPU  Name                     TCC/WDDM  | Bus-Id
Disp.A | Volatile Uncorr. ECC |
 | Fan  Temp   Perf          Pwr:Usage/Cap |
Memory-Usage | GPU-Util  Compute M. |
 |                                         |
|               MIG M. |
 |
=========================================+======================
 |   0  NVIDIA GeForce RTX 4090      WDDM  |
00000000:2D:00.0  On |                  Off |
 | 41%   30C    P5             74W /  436W |    1023MiB /
24564MiB |      4%      Default |
 |                                         |
|                  N/A |
 +-----------------------------------------+
------------------------+----------------------+

 +
-----------------------------------------------------------------------------+
 | Processes:
|
 |  GPU   GI   CI        PID   Type   Process name
GPU Memory |
 |        ID   ID
Usage      |
 |
=============================================================================
 |                        ...Lista uruchomionych procesów...
|
 +
-----------------------------------------------------------------------------+
```

Windows

Do działania Docker'a na systemie Windows można wykorzystać WSL 2 (ang. Windows Subsystem for Linux) bądź narzędzie do wirtualizacji Hyper-V. Ponieważ sterowniki NVIDIA oraz NVIDIA Container Toolkit wspierają tylko WSL 2, to właśnie ten backend został wykorzystany do uruchomienia kontenera. Warto zwrócić uwagę, że WSL 2 jest dostępny na systemach Windows 11 64-bit wersja 21H2 lub wyższa bądź Windows 10 64-bit wersja 21H2 (build 19044) lub wyższa.

1. Zinstaluj WSL 2 oraz Ubuntu

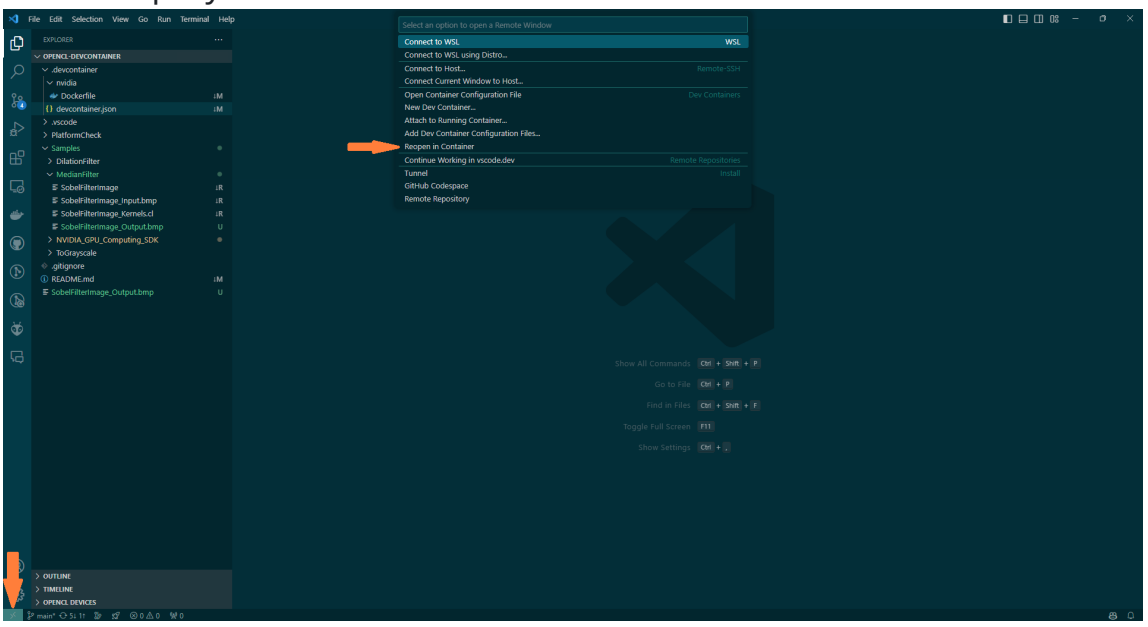
```
wsl --install
```

2. Pobierz aplikację Docker Desktop na system Windows.
3. Zainstaluj aplikację upewniając się, że opcja **Use WSL 2 instead of Hyper-V** jest zaznaczona.

Uruchamianie kontenera

Do uruchamiania kontenera deweloperskiego zalecany jest program Visual Studio Code, jednakże inne nardzędzia takie jak np. CLion także wspierają funkcję devcontainer, w związku z tym sposób uruchamiania oraz działania kontenera powininen być zbliżony. Poniżej zostaną opisane kroki potrzebne do uruchomienia kontenera w VSC.

1. Sklonuj to repozytorium https://github.com/Baey/OpenCL-DevContainer.git.
2. Otwórz VSC w repozytorium i zainstaluj rozszerzenie Dev Containers.
3. Otwórz repozytorium w kontenerze.



Od tego momentu program Visual Studio Code będzie połączony z uruchomionym kontenerem. Widoczne pliki oraz okna terminalu znajdują

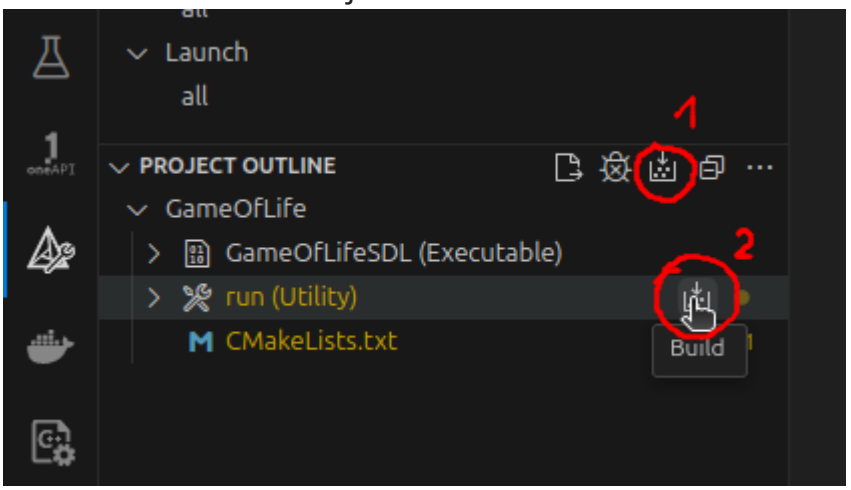
się bezpośrednio w kontenerze.

Uruchamianie przykładowych aplikacji

W poniższej sekcji zawarto wskazówki dotyczące uruchamiania przykładowych aplikacji dostarczonych razem z niniejszym kontenerem. Podobnie jak miało to miejsce w poprzednich sekcjach, zalecanym środowiskiem programistycznym jest VSC dla którego zostały przedstawione poniższe instrukcje.

Intel oneAPI

1. Uruchom kontener - zostało to opisane tutaj.
2. Kliknij na dodatek Intel oneAPI na lewym pasku z rozszerzeniami.
3. Po otwarciu pojawi się zakładka z plikami dodatku. Kliknij w lupę i wyszukaj interesującą Cię aplikację.
4. Po wybraniu przykładu pojawi się pole wyboru z opcjami "Open Sample Readme" - wybierz ją jeśli chcesz dowiedzieć się więcej na temat aplikacji. Jeśli zamierzasz jednak jedynie uruchomić program, kliknij w pole "Create Sample".
5. Twoim oczom ukaże się pole wyboru ścieżki gdzie zostaną zapisane pliki aplikacji. Ustaw ją wedle preferencji i kliknij "Ok".
6. Teraz powinno otworzyć się nowe okno VSC otwarte w folderze z plikami programu. Uwaga, kolejne kroki mogą różnić się w zależności od przykładu. W celu znalezienia więcej informacji odnoście uruchamiania czy też wymagań przykładu warto zajrzeć do pliku README projektu. Zazwyczaj jednak możliwe jest wsparcie się dodatkiem CMake - otwórz go w lewej zakładce.
7. Zbuduj projekt za pomocą przycisku 1 oraz uruchom klikając w miejsce oznaczone na obrazku jako 2.



Gratulacje! Udało Ci się uruchomić przykładową aplikację Intel OneAPI.

Aplikacje zawarte w repozytorium

1. Uruchom kontener - zostało to opisane tutaj.
2. W terminalu wbudowanym w VSC przejść do folderu z wybraną aplikacją przykładową.
   - dla aplikacji AMD ścieżka to **Samples/AMD SDK/ <Nazwa_folderu_z_aplikacją>**
   - dla aplikacji NVidii ścieżka to **Samples/NVIDIA_GPU_Computing_SDK/ OpenCL/bin/linux/release**
3. Uruchomić aplikację wpisując w terminal `./<Nazwa_aplikacji>`.