Intel Devcloud - Developer Cloud for the Edge

Intel® Developer Cloud offers complimentary access to a wide range of Intel® architectures to help you get instant hands-on experience with Intel® software and execute your edge, AI, high-performance computing (HPC) and rendering workloads. With preinstalled Intel® optimized frameworks, tools, and libraries, you have everything you need to fast-track your learning and project prototyping.

This library contains set of Python API which helps in deploying containerized application in the cloud and validate it against Intel's latest and greatest processors

System Requirements

Before you start the installation, check the supported operating systems and required Python* versions. The complete list of supported hardware is available in the Devcloud Documentation (https://www.intel.com/content/www/us/en/develop/documentation/devcloud-containers/top/available-hardware.html).

Supported Operating System	Python* Version (64-bit) (https://www.python.org/)
Ubuntu* 18.04 long-term support (LTS) x86, 64-bit	3.7, 3.8, 3.9, 3.10
Ubuntu* 20.04 long-term support (LTS) x86, 64-bit	3.7, 3.8, 3.9, 3.10
Red Hat* Enterprise Linux* 8 x86, 64-bit	3.7, 3.8, 3.9, 3.10
CentOS 7 x86, 64-bit	3.7, 3.8, 3.9, 3.10
macOS* 10.15 and higher versions, x86, 64-bit	3.7, 3.8, 3.9, 3.10
macOS* 11 and higher versions, arm64	3.7, 3.8, 3.9, 3.10
Windows 10* and higher versions, x86, 64-bit	3.7, 3.8, 3.9, 3.10

> NOTE: This package can be installed on other versions of Linux and Windows OSes, but only the specific versions above are fully validated.

Install the Intel Devcloud Runtime Package

Step 1. Set Up Python Virtual Environment

Use a virtual environment to avoid dependency conflicts.

To create a virtual environment, use the following commands:

On Windows:

```
python -m venv devcloud_env
```

On Linux and macOS:

```
python3 -m venv devcloud_env
```

> NOTE: On Linux and macOS, you may need to install pip. For example, on Ubuntu execute the following command to get pip installed: `sudo apt install python3-venv python3-pip`.

Step 2. Activate Virtual Environment

On Windows:

```
devcloud_env\Scripts\activate
```

On Linux and macOS:

```
source devcloud_env/bin/activate
```

Step 3. Set Up and Update PIP to the Highest Version

Run the command below:

```
python -m pip install --upgrade pip
```

Step 4. Install the Package

Run the command below:

```
pip install devcloud
```

Step 5. Verify that the Package Is Installed

Run the command below:

```
python -c "from devcloud import Devcloud"
```

If installation was successful, you will not see any error messages (no console output).

Troubleshooting

For general troubleshooting steps and issues, see Frequently Asked Questions of Devcloud (https://www.intel.com/content/www/us/en/develop/documentation/devcloud-containers/top/faq.html). The following sections also provide explanations to several error messages.

Errors with Installing via PIP for Users in China

Users in China might encounter errors while downloading sources via PIP during Devcloud installation. To resolve the issues, try the following solution:

Add the download source using the `-i` parameter with the Python `pip` command. For example:

```
pip install devcloud -i https://mirrors.aliyun.com/pypi/simple/
```

Use the `--trusted-host` parameter if the URL above is http instead of https.

List of API

Method	Description	Parameters	Return
Devcloud.connect	Validates the token and gives the snapshot of Dashboard and	token: str(Optional) API Token of the user, if not provided then will be prompted	None

	Project deployed by the user		
Devcloud.transfer	Transfers file from Cloud storage like AWS S3 to Devcloud	region: str - AWS bucket region bucketName: str - Name of the bucket path: str - file/folder path present relative to the bucket to be copied to devcloud accessKey: str (Optional) - Access key of AWS S3 bucket secretKey: str (Optional) - Secret key of AWS S3 bucket	None
Devcloud.availableHardware	Prints available hardware and their respective ID, Processor Name, Integrated GPU Name and Memory detail	None	None
Devcloud.createContainer	Creates container image and stores in the devcloud	projectName: str - Unique name of the project containerName: str - Unique name of the container url: str - Container image URL	None
Devcloud.configureContainer	Configures the container with various option as mentioned in parameters	projectName: str - Name of the project under which container present containerName: str - Name of the container which needs to configured (This should be assigned to the project name mentioned above) port: list[int] - Port number which must be exposed to external consumption. Port range must be above 1024 label: list[str] - Label to be associated to the container entryScript: str - Initial file which must be executed when container boots up output: str - Output mount folder, which can be later accessed using Devcloud file system mountPoint: list[(str, str)] - To mount Devcloud file system to user's container. This is a tuple, were first one is for Devcloud file system path and second is mount point in container. Multiple such mount path can be assigned environment: str - environment variable to be passed to container. eg., -e token=XYZ	None
Devcloud.launch	Launches the Project against the selected hardward	projectName: str - Name of the project which should be lauched edgeNode: int - Intel's latest and greatest hardware to be used. Get the list by calling Devcloud.availableHardware()	None
Devcloud.getStatus	Displays launched project status	projectName: str (Optional) If provided will show dashboard detail of the project output: str (Optional) if value is wide then provided additional detail such as Performance and create time	None
Devcloud.getFilesPreview	Gets the response object of the requested file path in the Devcloud's filesystem	projectName: str - Name of the project path: str - Relative path from the project's output folder edgeNode: str (Optional) - Intel processor used, if not provided then latest completed container's in the specified project will be used createTime: str (Optional) - If present then this timestamp is used to retrieve the files present under the given runtime of the container 	http.client.HTTPResponse

List of External Libraries used

Name	Version
Requests	2.28.1
Pyjwt	2.5.0
Typer	0.6.1
Pandas	1.5.0
Tabulate	0.9.0
Pillow	9.2.0
importlib-resources	5.10