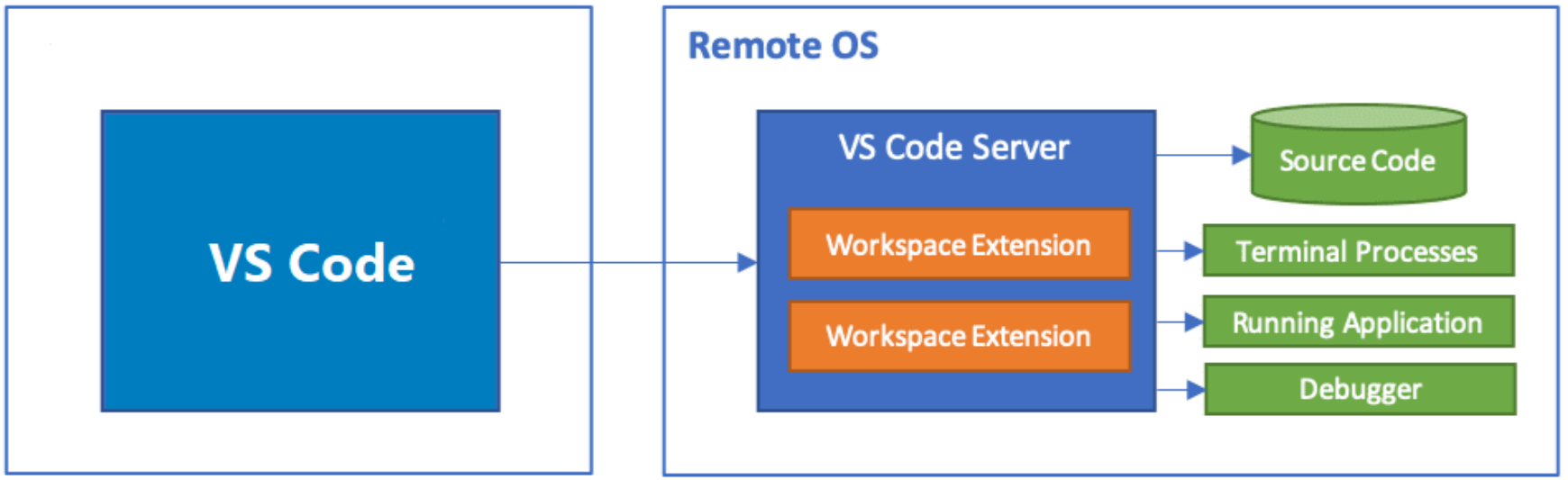
Development Containers in Visual Studio Code

PSTAT 197A (Fall 2026)

Sang-Yun Oh

University of California, Santa Barbara

Development on Remote Host (Jetstream2 VM)



- Virtual machine (JS2VM) is a computer.
- Connect over network to use it
  - → Protocols: e.g., RDP, VNC, SSH, Remote Tunnel, etc.
  - → We will use VS Code Remote Tunnel
- Demo: Connect to JS2VM using VS Code Remote Tunnel.
  `https://code.visualstudio.com/docs/remote/tunnels`

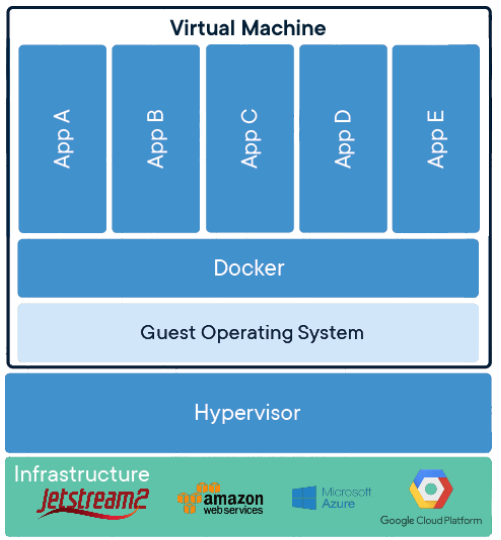
Development on Remote Host (Jetstream2 VM)

- Now we need software and libraries for our work.
- Installing software is tedious and error-prone.
- **Containers** are a solution to this problem.
- Containers are everywhere!

Container URL	REGISTRY	NAMESPACE	IMAGE_NAME	TAG
docker.io/ubuntu:22.04	docker.io	library	ubuntu	22.04
docker.io/intel/oneapi:latest	docker.io	intel	oneapi	latest
quay.io/jupyter/r-notebook:r-4.4.2	quay.io	jupyter	r-notebook	r-4.4.2
nvcr.io/nvidia/pytorch:24.12-py3	nvcr.io	nvidia	pytorch	24.12-py3

Development Containers Basics

- JS2VM is the “Virtual Machine” running Guest OS.
- Docker engine inside JS2VM runs containers.
- Container can hold software and libraries you need.
- `Dockerfile` defines the container contents.
- VS Code can help us run and manage containers.



Development Container with VS Code

- Jupyter Container Registry: `https://quay.io/organization/jupyter`
- Container tag: `quay.io/jupyter/scipy-notebook:python-3.13`
- **Let's ask an LLM to create a development container setup for us**

Use the following prompt

```
Create a development container setup for me with following features:
- use `quay.io/jupyter/scipy-notebook:python-3.13` as base image in a `Dockerfile`
- bind mount this directory into the container
- install extensions for Python development
```

Filesystems and Files for Development Containers

Jetstream2 VM Filesystem

```
/home/exouser/python-project/
├── .devcontainer
│   ├── devcontainer.json
│   └── Dockerfile
├── .gitignore
├── README.md
└── ...
```

Container Filesystem

```
/workspaces/python-project
├── .devcontainer
│   ├── devcontainer.json   # devcontainer config
│   └── Dockerfile       # container definition
├── .gitignore              # for git
├── README.md
└── ...                     # other files
```

Target directory in container is set in `.devcontainer/devcontainer.json`.

```
"workspaceMount": "source=${localWorkspaceFolder},target=/workspaces/python-project,type=bind",
"workspaceFolder": "/workspaces/python-project",
```

Working with Files

Assume project files are in `/home/exouser/python-project/` on JS2VM and mounted to `/home/jovyan/work/` in the container,

- Changes to files in `/home/jovyan/work/` in the container are reflected in `/home/exouser/python-project/` on JS2VM, and vice versa.
- File `/home/exouser/python-project/README.md` on JS2VM and `/home/jovyan/work/README.md` in the container are the same file.
- Other than this mounted directory, container filesystem is ephemeral.
- JS2VM filesystem is persistent unless VM is deleted.

Container Filesystem is Ephemeral

In general, container filesystem is ephemeral. Other than files mounted from a persistent storage, changes to other files in the container are lost when the container is rebuilt or deleted.

Modifying Container Image

- `Dockerfile` or `Containerfile` defines the container image.
- Rebuilding the container image applies the changes to your container.
- Rebuilding the container does not affect files in persistent storage.
- Modify system software, python/R packages, OS configuration, etc.

References

- *Dockerfile reference on Turing Way* (Community 2022)
- *Ten Simple Rules for Writing Dockerfiles for Reproducible Data Science* (Nüst et al. 2020)
- *PSTAT documentation*

Modifying Devcontainer Configuration

- `.devcontainer/devcontainer.json` defines the container configuration.
- Modify how container is built, run, and configured.
- Rebuilding the container applies the changes to your container.
- VS Code extensions, settings, and other configurations.

References

- Development Container Specification
- *Ten Simple Rules for Writing Dockerfiles for Reproducible Data Science* (Nüst et al. 2020)

Container Image Examples

Container URL	REGISTRY	NAMESPACE	IMAGE_NAME	TAG
docker.io/ubuntu:22.04	docker.io	library	ubuntu	22.04
docker.io/intel/oneapi:latest	docker.io	intel	oneapi	latest
quay.io/jupyter/r-notebook:r-4.4.2	quay.io	jupyter	r-notebook	r-4.4.2
nvcr.io/nvidia/pytorch:24.12-py3	nvcr.io	nvidia	pytorch	24.12-py3

Tutorials on Development Containers Usage

- Basic commands for starting/stopping/detaching/stopping containers
- Moving files around and working with files on GitHub repository
- Job management and running long jobs

Exercises

- Install `tmux` by modifying `Dockerfile` and rebuild the container.
- Change the `FROM` image to another Jupyter Docker Image

GitHub Copilot

GitHub Copilot can assist you in writing and modifying your development container configuration files. You can use it to:

- Generate boilerplate code for your `Dockerfile` or `devcontainer.json`.
- Get suggestions for VS Code extensions to include in your development container.
- Find and fix issues in your configuration files.

References

Community, The Turing Way. 2022. “The Turing Way: A Handbook for Reproducible, Ethical and Collaborative Research.” Zenodo. https://doi.org/10.5281/ZENODO.7625728.

Nüst, Daniel, Vanessa Sochat, Ben Marwick, Stephen J. Eglen, Tim Head, Tony Hirst, and Benjamin D. Evans. 2020. “Ten Simple Rules for Writing Dockerfiles for Reproducible Data Science.” Edited by Scott Markel. *PLOS Computational Biology* 16 (11): e1008316. https://doi.org/10.1371/journal.pcbi.1008316.