Debugging Python Applications Running in Containers



Steven HainesPrincipal Software Architect

@geekcap www.geekcap.com

Overview



- Modifying Python code running inside a container
- Debugging Python containers using PyCharm
- Debugging Python containers using Visual Studio Code



Why Is Debugging Important?

Solve problems in a running environment with real data

Ease of development

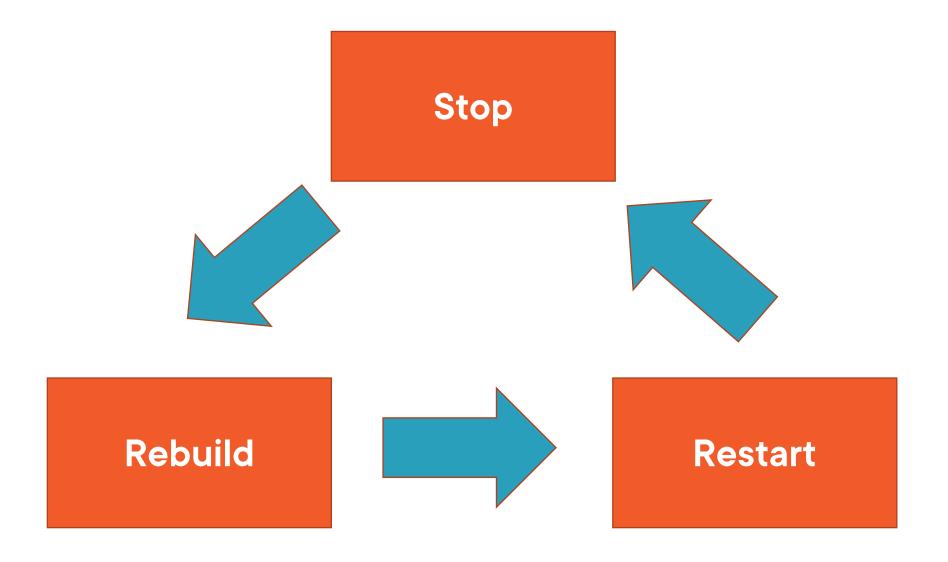
Increased productivity

Modifying Python Code Running Inside a Container

Running Flask Locally

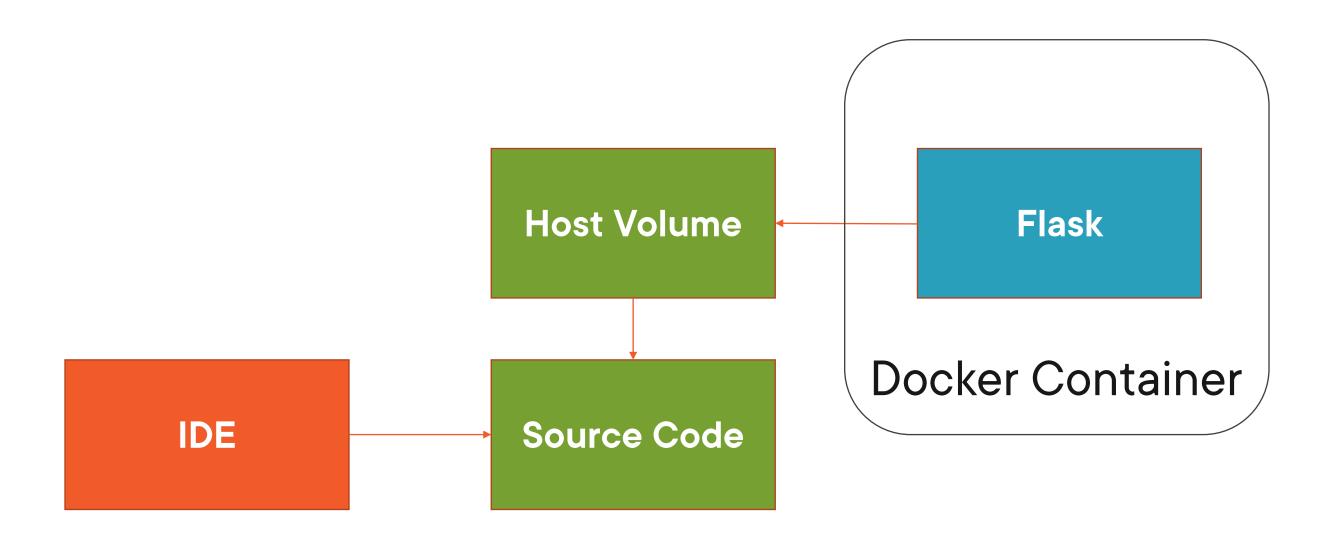


Stop-Rebuild-Restart Loop





Running Flask in Docker



```
services:
   productservice:
     build: product-service
     volumes:
     - "./config:/config"
     - "./product-service/src:/code"
```

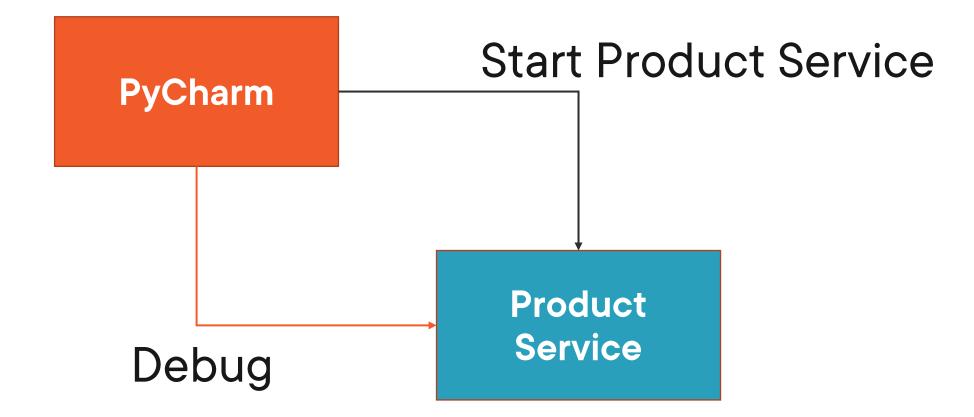
Mounting Source Code Using a Host Volume

The Product Service's src is mounted directly to the container's /code directory

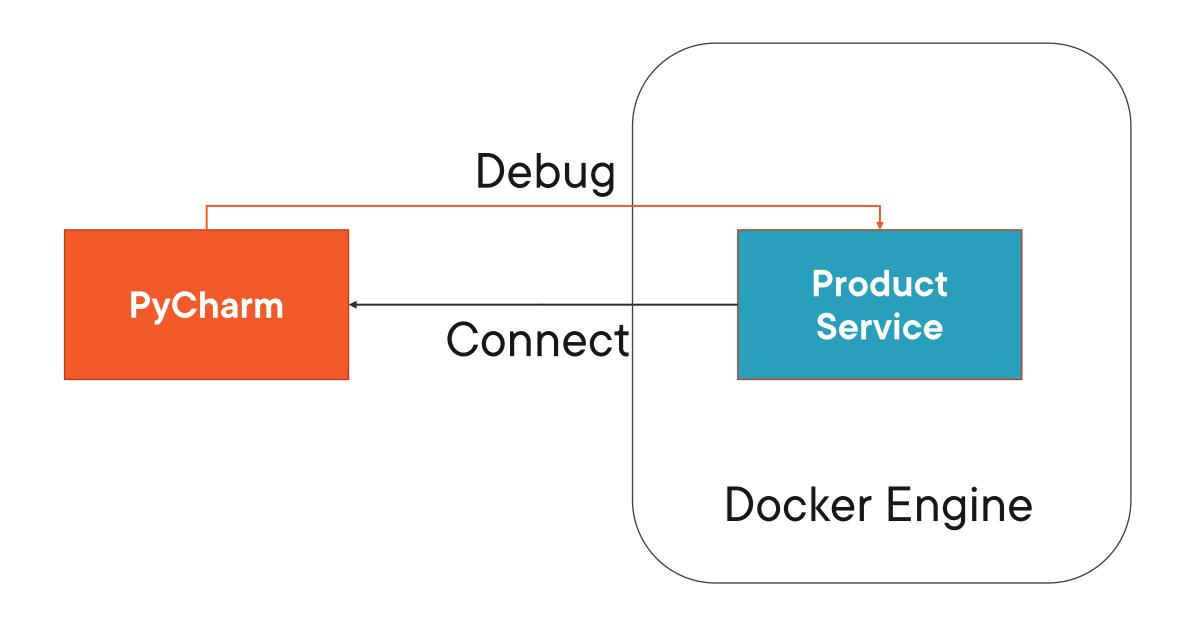
Debugging Containers with PyCharm



Local Debugging



Remote Debugging



https://pypi.org/project/pydevd-pycharm/



```
pip install pydevd-pycharm~=version
import pydevd_pycharm
pydevd_pycharm.settrace(
   'host.docker.internal',
    port=12345,
     stdoutToServer=True,
    stderrToServer=True,
     suspend=False)
if __name__ == '__main__':
    app.run(debug=False,
host='0.0.0.0')
requirements.txt
pydevd-pycharm~=version
```

■ Install the pydevd-pycharm pip package

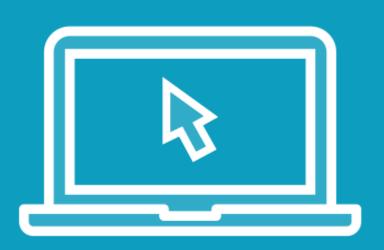
■ Import pydevd_pycharm

■ Connect to PyCharm, running on the local machine

◄ Disable Flask debug mode

■ Add pydevd-pycharm to requirements.txt

Demo

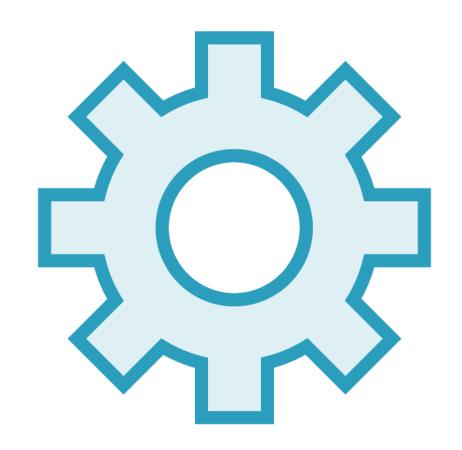


- Configure the PyCharm debugger
- Add pydevd-pycharm to our application using pip
- Add pydevd-pycharm to our requirements.txt file
- Setup pydevd-pycharm in our application
- Start the PyCharm debugger
- Start our containers
- Debug the application



Debugging Containers with VS Code

Remote Debugging



Configure Debugger

Setup a debugger library, debugpy, to listen for incoming connections



Remote Connection

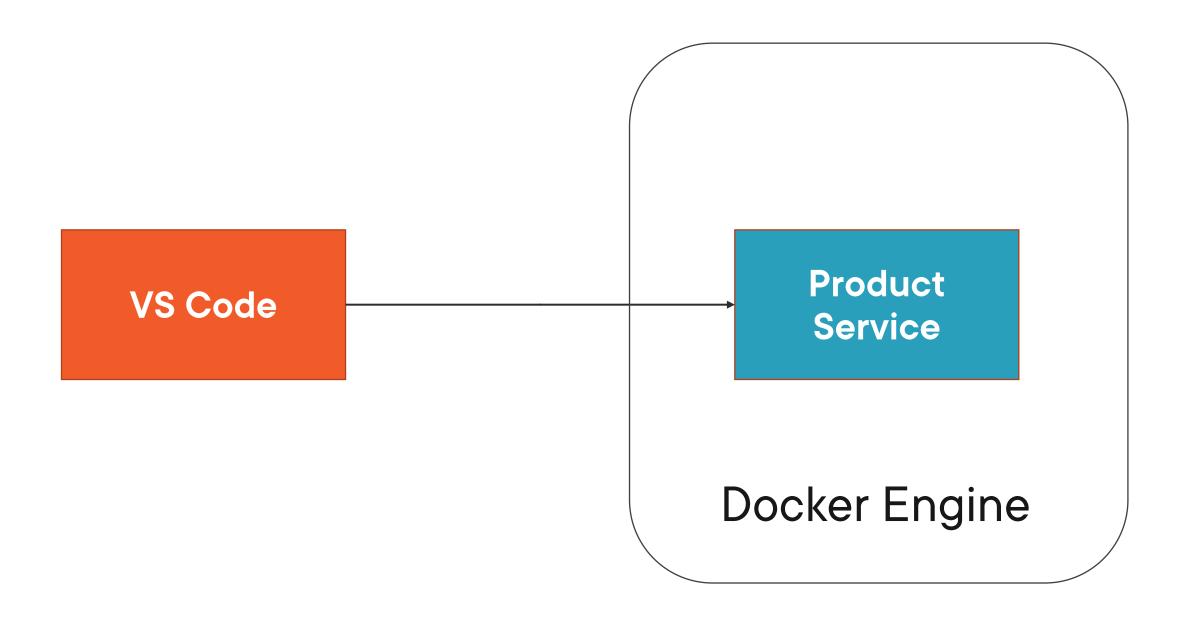
Configure a "Remote Attach" debug configuration to connect to the container



https://github.com/microsoft/debugpy



Remote Attach Debugging



```
$ pip install debugpy
import debugpy
debugpy.listen(
        ("0.0.0.0", 5678))
if __name__ == '__main__':
    app.run(debug=False,
            host='0.0.0.0')
```

requirements.txt

debugpy==1.2.1

■ Install the debugpy pip package

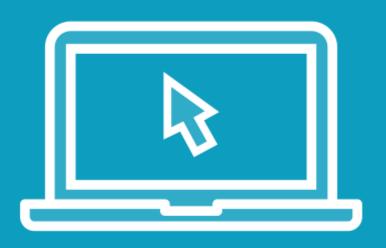
◄ Import debugpy

◄ Listen for incoming requests

■ Disable Flask debug mode

■ Add debugpy to requirements.txt

Demo

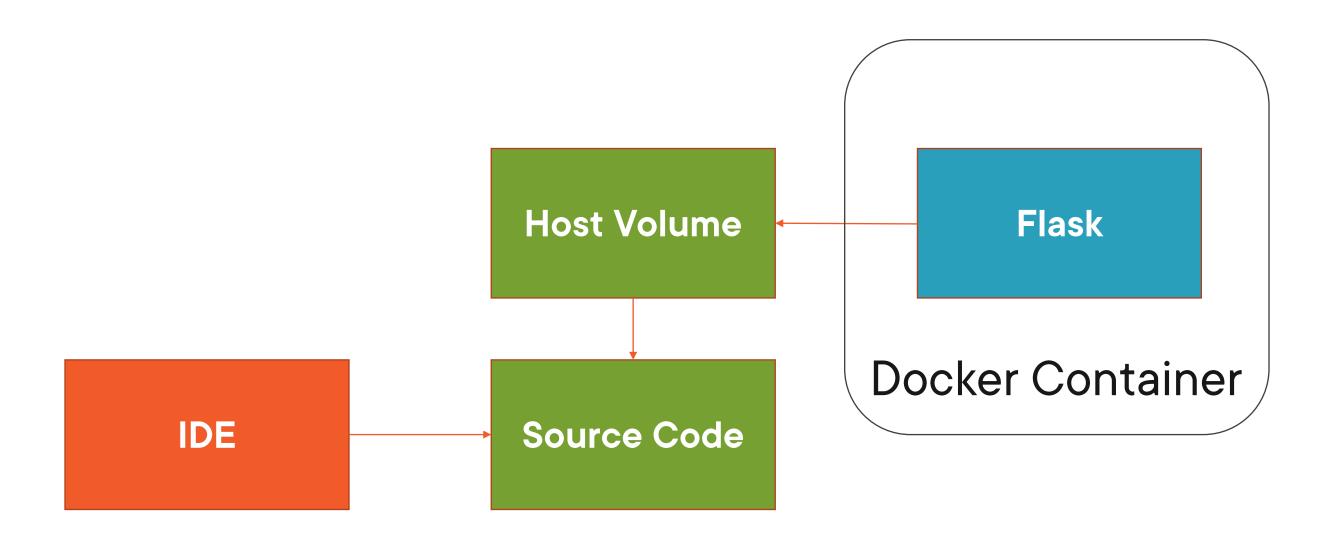


- Add debugpy to our application using pip
- Add debugpy to our requirements.txt file
- Setup debugpy in our application
- Start our containers
- Use the Visual Studio Code debugger to connect to the product service

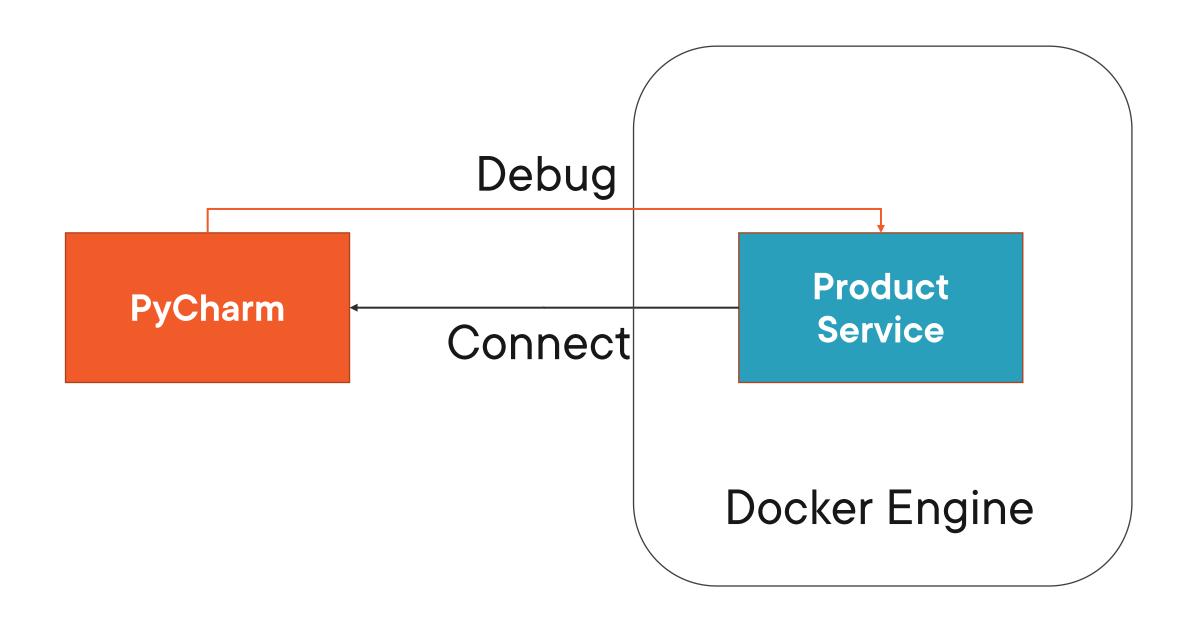
Conclusion



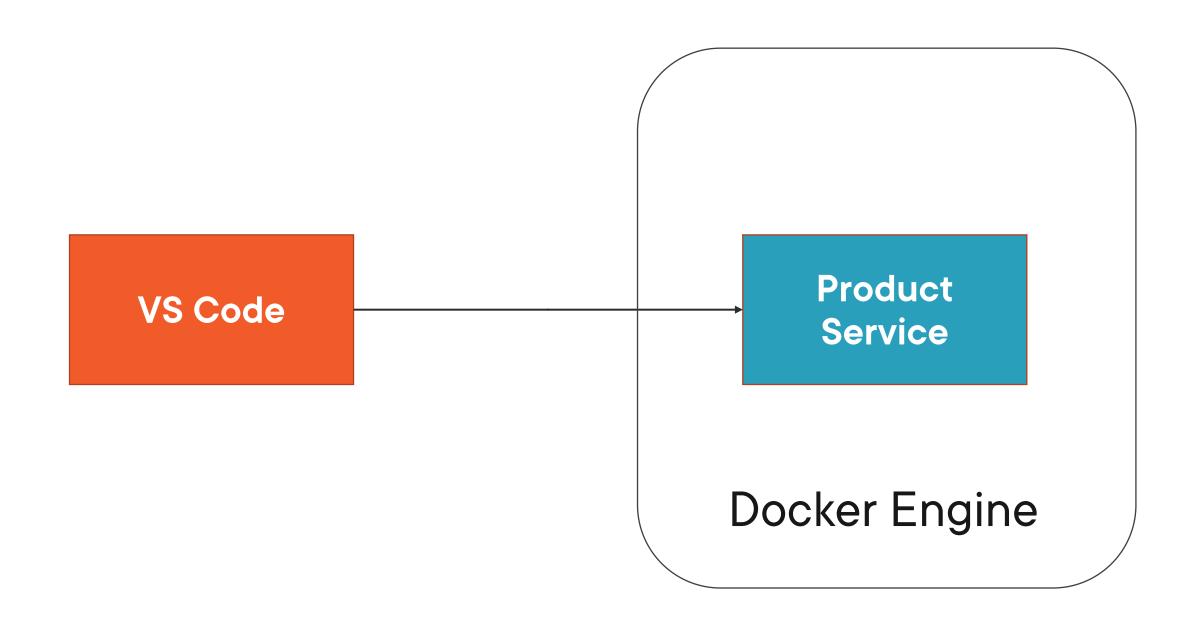
Running Flask in Docker



Remote Debugging with PyCharm



Remote Attach Debugging with VS Code



Summary



- You should understand how to develop code in a running container
- You should understand how to debug an application with PyCharm and VS Code
- You should feel comfortable debugging your own applications

