UNIVERSITY of FLORIDA
Herbert Wertheim
College of Engineering

Performant Programming
CEN4930

Ex0: Rust & Python Toolchain Setup

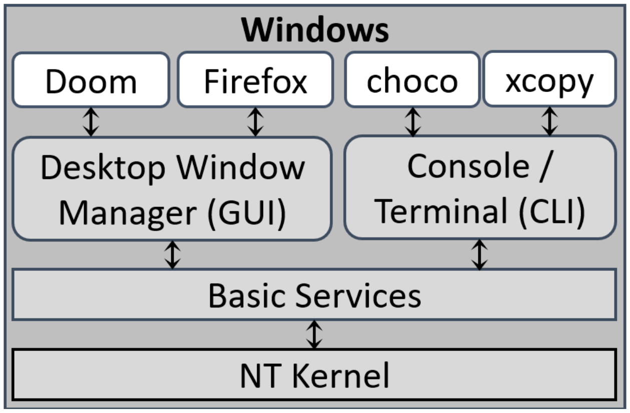
Overview

This exercise is intended to help students set up your toolchain so that you can build projects the course. This will involve setting up the runtime environments and IDE for Python 3.12. You are not required to use the IDE we specify, though it is recommended. There are no deliverables and there is no grade for this exercise.

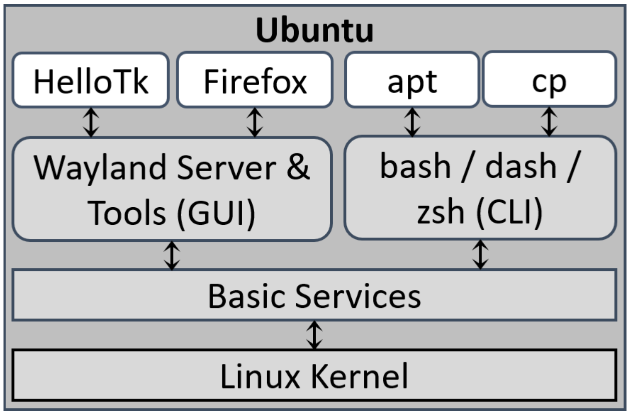
The environment specified in this document will be the one exclusively used for testing student projects. Note that these instructions are written for Windows users, but students using Ubuntu 24.04 and other Linux distributions should be able to set up their environments for the course. With that said, please note that there is no instructional support for other operating systems. For Intel Macs, users should consider using a virtual machine running Ubuntu 24.04 or installing BootCamp and Windows 11 (which is free to UF students). Windows 10 and 11 must have all updates installed; you may need to re-install Ubuntu within WSL. **There is no support for older versions of Windows or non-Intel machines**, as we do not have the means to test them.

Architecture

Windows and Ubuntu are both built as end-user operating systems. Typically, a modern user-facing operating system (OS) includes a command line interface (CLI) and graphical user interface (GUI):

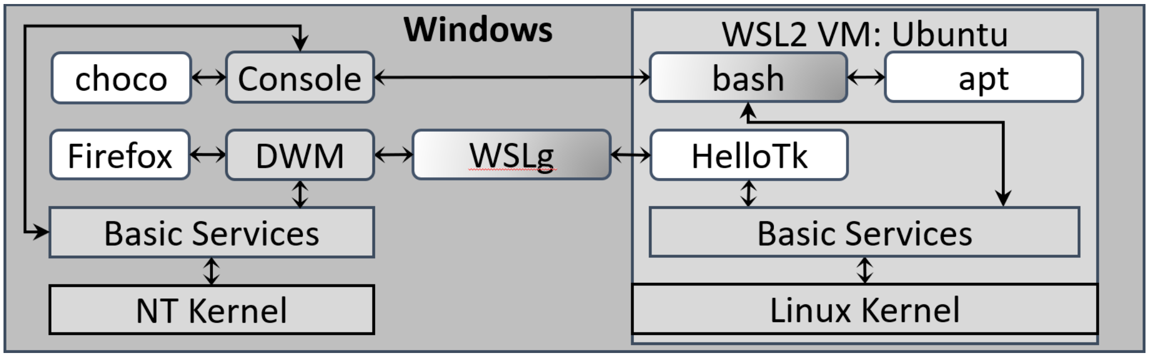


Windows Runtime Architecture



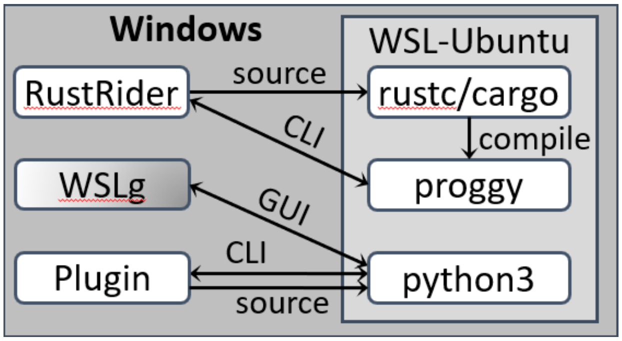
Ubuntu Runtime Architecture

Windows can run Linux-based operating systems in the Windows Subsystem for Linux (WSL):



WSL2 and interactions between kernels and major system components.

Windows has built-in support for graphical Linux applications. This is useful because Python's runtime and development tools are most well developed on Linux systems. Additionally, it provides us a common target for testing. Many IDE suites can also integrate with WSL:



GUI software & WSL2 on Windows

In this exercise, students will set up Ubuntu 22.04 on WSL2. They will also setup the PyCharm IDE suite and the Rust plugin to communicate with WSL. Projects and exercises in this course will target this environment.

Tasks

This exercise is broken into these steps:

a) Install Chocolatey package manager and other tools
b) Setup WSL2 with Ubuntu 22.04 and Python 3.10
c) Setup RustRover and Python Plugin** (or your own IDE) ***Some conditions apply…***

Before starting this process, make sure your Windows installation is up to date – some commands require the latest patches / updates to function properly! This is particularly true of WSL, which is in active development.

WSL & Chocolatey Package Manager

To install enable WSL and install Chocolatey, open the CMD or PowerShell (🔎 → "**PowerShell**") as an Administrator (right-click → "Run as Administrator"). From there, run these commands (copy-paste):

```
wsl --install
```

```
msiexec /i https://github.com/chocolatey/choco/releases/download/2.2.2/chocolatey-2.2.2.0.msi
```

You can test this with the command "**choco -?**". Once it is ready, install the other tools:

```
choco install rustrider
```

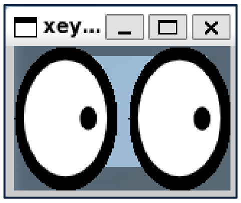
If you prefer to use IntelliJ + Rust and Python Plugin, or no IDE, you may skip this – but it is what we will test on!

WSL Ubuntu Setup

Open Ubuntu **from the start menu** to select a username / complete installation. Then update and install tools:

```
sudo apt update
sudo apt upgrade
sudo apt install build-essential cargo cmake gdb python3-pip python3-virtualenvwrapper \
                 python3-venv rustc x11-apps
```

GUI Testing



Open a fresh terminal console and execute the **xeyes** app to test the tools. You should see this weirdo onscreen… and his eyes follow you! Creepy, dude…

Finally, let's add a few settings to allow WSL, the IDEs, and other tools to connect seamlessly:

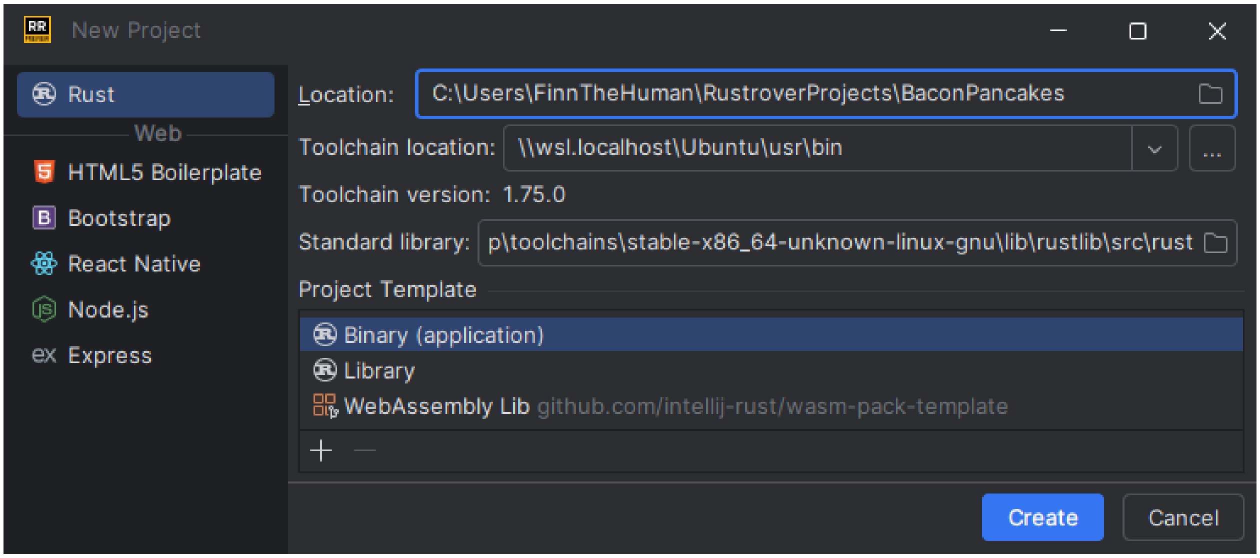
```
sed -i.old "/VIRTUALENVWRAPPER_PYTHON=/cexport\ VIRTUALENVWRAPPER_PYTHON=\$(which\ python3)" \
           "$HOME/.profile"                                                            #1

if ! grep -qF "source /usr/share/virtualenvwrapper/virtualenvwrapper.sh" ~/.profile; then
  echo -e "source /usr/share/virtualenvwrapper/virtualenvwrapper.sh" >> ~/.profile; fi      #2
```

These lines 1) set python version for virtual environments and 2) enable virtual environment tools for Python.

Using RustRover with WSL

RustRover can automatically detect and use the WSL VM. You can select a toolchain when creating a project:



Location: C:\Users\FinnTheHuman\RustroverProjects\BaconPancakes

Toolchain location: \\wsl.localhost\Ubuntu\usr\bin

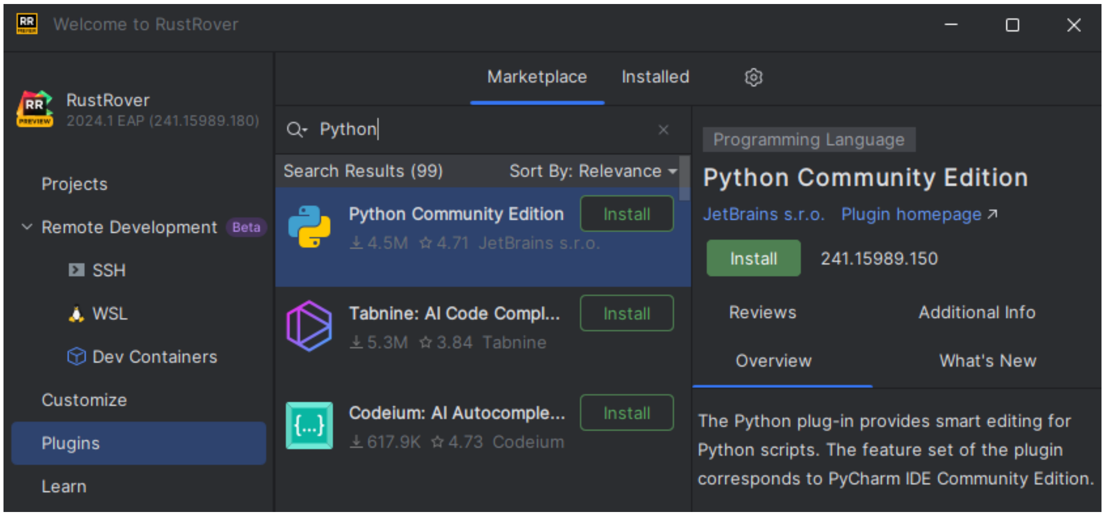
Toolchain version: 1.75.0

Standard library: p\toolchains\stable-x86_64-unknown-linux-gnu\lib\rustlib\src\rust

Project Template: Binary (application)

Python Plugin

Installing the Python plugin is straightforward - go to the Plugin menu, search for Python, and click "Install":



Python

The Python plug-in provides smart editing for Python scripts. The feature set of the plugin corresponds to PyCharm IDE Community Edition.

There is a limitation to the Python plugin: it cannot use the WSL version of Python (yet). If you prefer, you can use PyCharm with the Rust plugin; however, much of the work in this class will be Rust, so RustRover is likely a better choice for most students!