Get your ROMI running

Download Robotpy

On a mac, in PyCharm project open the terminal interface. In terminal, do the following commands one at a time:

`python3 -m pip install robotpy`

If you have another type of operating system please look at the instructions here.

Initialize a Robotpy Project

We will need to initialize a robotpy project. This is done by the following command:

`python3 -m robotpy init`

When you do this, there will be a `robot.py` file created in your project. This is the file that will be the main file for your robot code. There will also be a `pyproject.toml` file created. This file is used to manage the dependencies of your project.

To run the ROMI will need some additional libraries. These are installed by first editing the `pyproject.toml` file. We need to uncomment the following lines:

```
robotpy_extras = [
    # "all",
    # "apriltag",
    "commands2", # make sure that this is uncommented
    # "cscore",
    # "navx",
    # "pathplannerlib",
    # "phoenix5",
    # "phoenix6",
    # "photonvision",
    # "playingwithfusion",
    # "rev",
    "romi", # make sure that this is uncommented
    "sim", # make sure that this is uncommented
    # "xrp",
]
```

Then what you need to do is run the following command in the terminal:

```
python3 -m robotpy sync
```

Running the ROMI

In the `robot.py` file make sure you have code like the following:

```
import os
import wpilib
import commands2
from wpilib import TimedRobot, Joystick, Spark
from wpilib.drive import DifferentialDrive

#from robotcontainer import RobotContainer

# If your ROMI isn't at the default address, set that here
os.environ["HALSIMWS_HOST"] = "10.0.0.2"
os.environ["HALSIMWS_PORT"] = "3300"
```

```
class MyRobot(commands2.TimedCommandRobot):
    """
    Command v2 robots are encouraged to inherit from TimedCommandRobot, which
    has an implementation of robotPeriodic which runs the scheduler for you
    """


    def robotInit(self) -> None:
        """
        This function is run when the robot is first started up and should be used for any
        initialization code.
        """
        self.controller = Joystick(0)
        self.left_motor = Spark(0)
        self.right_motor = Spark(1)
        self.drivetrain = DifferentialDrive(self.left_motor, self.right_motor)


    def autonomousInit(self) -> None:
        """This autonomous runs the autonomous command selected by your RobotContainer class."""
        pass

    def autonomousPeriodic(self) -> None:
        """This function is called periodically during autonomous"""
        pass

    def teleopInit(self) -> None:
        pass

    def teleopPeriodic(self) -> None:
        """This function is called periodically during operator control"""
        forward = self.controller.getRawAxis(0)
        print(forward)
        rotate = self.controller.getRawAxis(1)
        self.drivetrain.arcadeDrive(forward, rotate)
```

This code will allow you to drive the ROMI with a joystick.

The lines:

```
import os
import wpilib
import commands2
from wpilib import Joystick, Spark
from wpilib.drive import DifferentialDrive
```

are necessary to make the code work. The `os` library is a standard python library that connects to the operating system. The `wpilib` library is the library that connects to the ROMI. The `commands2` library is a library that allows you to use the command based system that we will be using in the future. The `Joystick` class is a class that allows you to connect to the joystick. The `Spark` class is a class that allows you to connect to the motors. The `DifferentialDrive` class is a class that allows you to connect to the drivetrain of the ROMI.

The `os.environ["HALSIMWS_HOST"] =` and `os.environ["HALSIMWS_PORT"] =` lines are necessary to connect the simulator to the ROMI.

The `MyRobot` class is a class that is a subclass of the `TimedCommandRobot` class. This class is the main class that will

run the robot. The `robotInit` method is a method that is called when the robot is first started up. This method is used to initialize the robot.

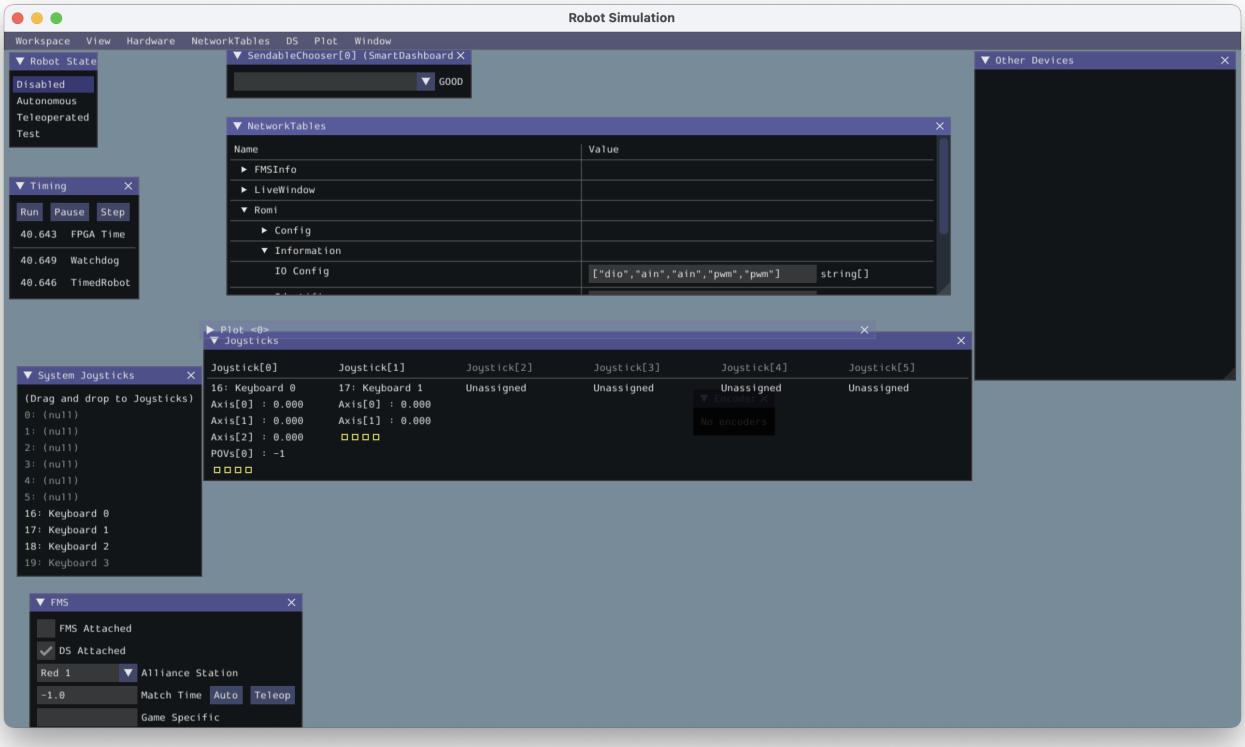
The `controller`, `left_motor`, `right_motor`, and `drivetrain` are all objects that are created in the `robotInit` method. The `controller` object is a `Joystick` object that is connected to the joystick. The `left_motor` and `right_motor` objects are `Spark` objects that are connected to the motors. The `drivetrain` object is a `DifferentialDrive` object that is connected to the drivetrain of the ROMI.

Running the code

Go into the terminal and run the following command:

```
python3 -m robotpy sim --ws-client
```

This should load the code you have written to the ROMI and start a window that looks like this



If there is nothing listed under **Joystick[0]** drag **Keyboard 1** over and drop in that area. Change the robot status to **Teleoperated** and you should be able to drive your ROMI with the s and w keys to go forward and back and the a and d to make it turn.

Troubleshooting

If you get a message that says that you need to install a certificate, then you need to run the following command in the terminal. Make sure that the python you are running is 3.12. If it isn't then this code will have be edited to reflect the version of python you are running.

```
/Applications/Python\ 3.12/Install\ Certificates.command
```

If the you get an error saying it doesn't recognize the `--ws-client` option, then you need to update the `robotpy` library. This is done by running the following command in the terminal:

```
python3 -m pip install --upgrade robotpy
```

And then again running the sync command for the robotpy library:

```
python3 -m robotpy sync
```