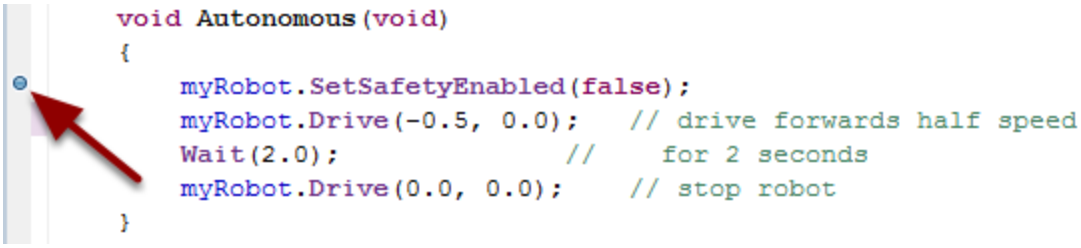
Debugging a robot program

A debugger is used to control program flow and monitor variables in order to assist in debugging a program. This section will describe how to set up a debug session for a C++ FRC robot program.

Set up Launcher

If you had set up debugging with a previous version of the Eclipse plugins, you may need to change your Default Launcher. A separate article has been created showing the steps to do this: https://Wpilib.screenstepslive.com/s/4485/d/yqzava

Set a Breakpoint

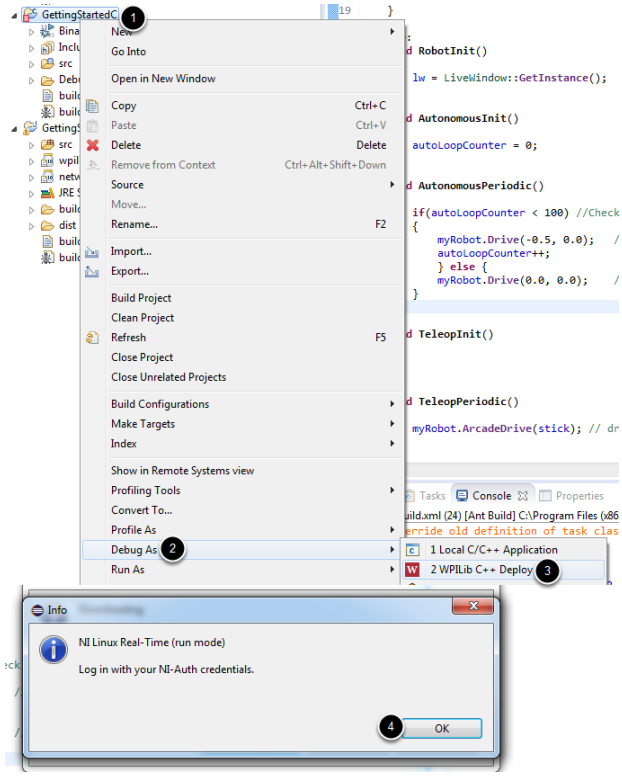


Clicking the “Debug” button does several things. It changes the Workbench to the Debug Perspective which has the views Debug, Breakpoints, and Variables along the right side of the window. It starts the robot task and pauses it at at the first breakpoint hit.

Double-click in the left margin of the source code window to set a breakpoint in your user program: A small blue circle indicates the breakpoint has been set on the corresponding line. (You can see all your breakpoints in the Workbench’s Breakpoints view.)

Debugging a robot program

Launching Debug

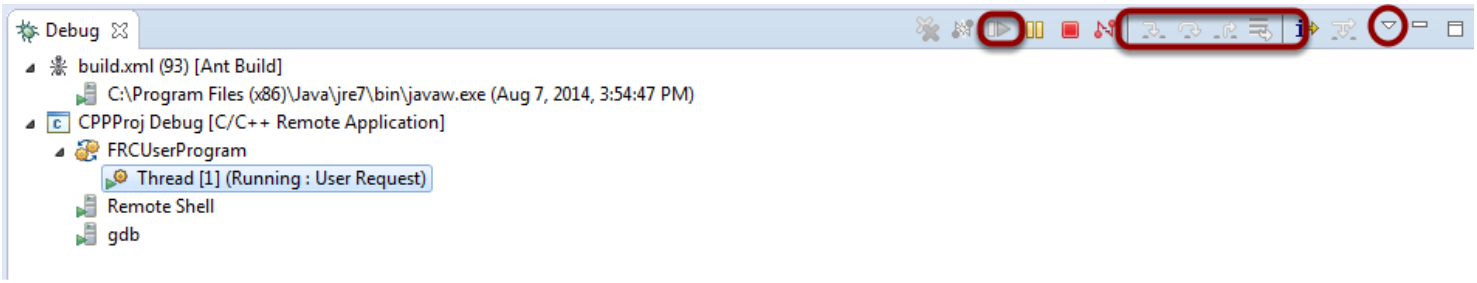


To launch the program in debug mode:

1. Right click on the project
2. Hover over Debug As
3. Select WPILib Deploy
4. When the window appears asking you to log in with your NI-Auth credentials, click ok.

If you receive a prompt about changing perspectives, click Ok.

The Debug window



Debugging a robot program

The Debug window shows all processes and threads spawned by your code running on the roboRIO. Select the stack frame to see the current instruction pointer and source code (if available) for the selected process. When your breakpoint is reached, make sure your program is selected in the task list and your source code is displayed with a program pointer. You can continue stepping through your code using “Resume,” “Step Into,” “Step Over,” and “Step Return” buttons: If you see assembly code instead of C++ code displayed, it’s because you’ve stepped down into library code where the source is not available to the debugger. “Step Return” will bring you back up a level.

If the debug toolbar (resume, terminate and step controls) is not visible, click the down arrow at the top right of the debug window and select the **Show Debug Toolbar** option

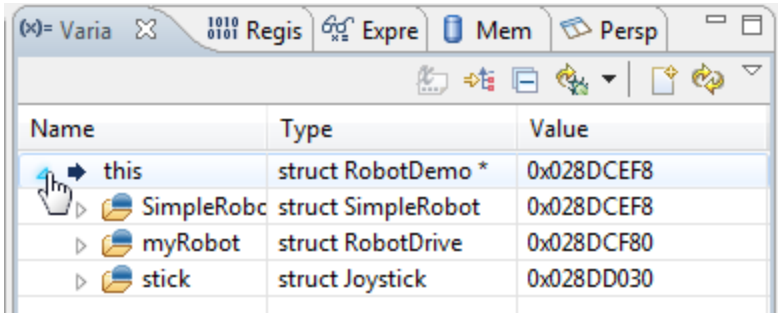
Click the “Resume” button (the green arrow) to resume program execution up to the first breakpoint. Note that if the breakpoint you have placed is inside one of the Teleoperated or Autonomous methods, the robot will have to be set to the appropriate mode and enabled with the Driver Station in order to reach the breakpoint.

Your Breakpoint

```
void Autonomous(void)
{
    myRobot.SetSafetyEnabled(false);
    myRobot.Drive(-0.5, 0.0);    // drive forwards half speed
    Wait(2.0);                   //    for 2 seconds
    myRobot.Drive(0.0, 0.0);     // stop robot
}
```

The program will start running and then pause at the breakpoint. From here you can use the three panes on the left of the screen (Debug, Breakpoints and Variables) to monitor and control the execution of your program.

The Variables Tab



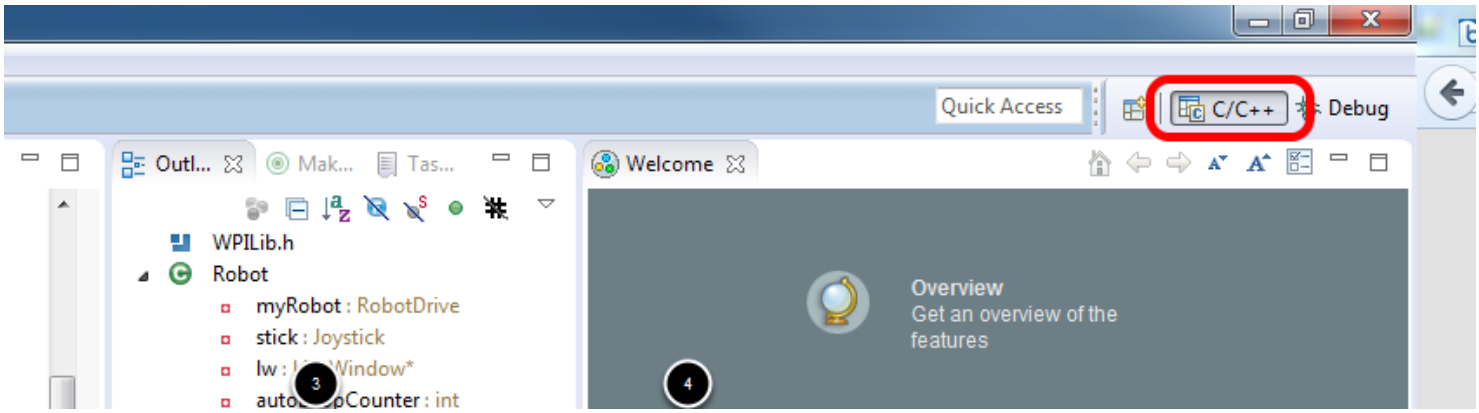
Debugging a robot program

The Variables view shows the current values of variables. To see a variable that is not displayed, select the “Expressions” tab and enter the variable name. This will show the variable’s value if it’s in scope. You may also want to click on the arrows next to a variable to expand the tree and show it's members. For example, expanding our Robot Demo variable ("this") shows our "myRobot" and "stick" variables.

Stopping Debugging

To stop debugging, you can disconnect or terminate the process. Disconnecting detaches the debugger but leaves the process running in its current state. Terminating the process kills it on the target.

Exiting Debug Perspective Debugging



After you have terminated the Debug session, to either return to editing code, or to debug again, you need to return to the C/C++ Perspective. If the C++ perspective is still open you may do this by clicking the icon with the C in the top right corner of the screen or pressing Ctrl+F8. If you do not see the icon there, select Window->Open Perspective->Other->C/C++ and click OK.