

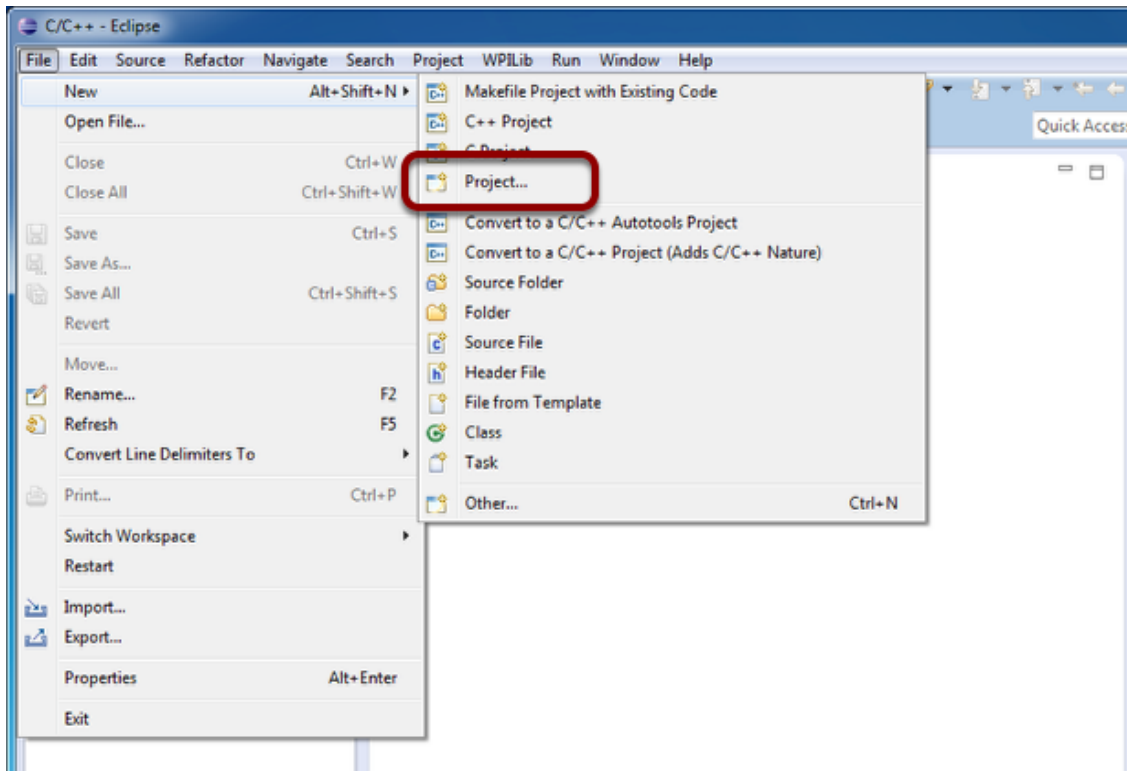
Creating your Benchtop Test Program

The simplest way to create a robot program, is to start from one of the three supplied templates (Sample, Iterative or Command). Sample is best used for very small sample programs or advanced programs that require total control over program flow. Iterative Robot is a template which provides better structure for robot programs while maintaining a minimal learning curve. Command-Based robot is a template that provides a modular, extensible structure with a moderate learning curve.

The templates will get you the basis of a robot program, organizing a larger project can often be a complex task. RobotBuilder is recommended for creating and organizing your robot programs. You can learn more about RobotBuilder [here](#). To create a command-based robot program that takes advantage of all the newer tools look at [Creating a Robot Project](#) in the Command Based Programming Chapter.

Creating your Benchtop Test Program

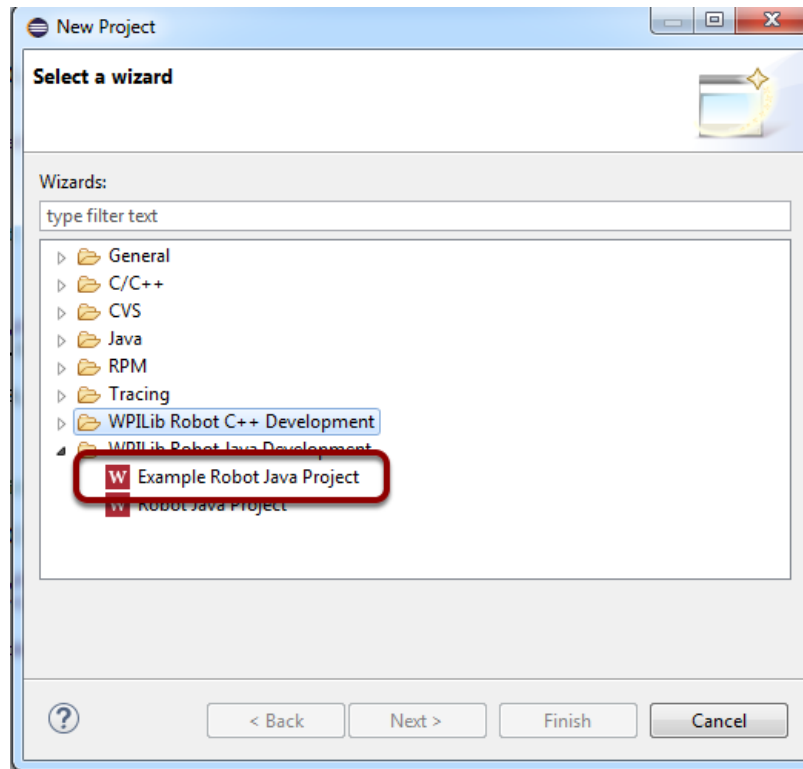
Creating a project



To create a project click "File" then "New" then click "Project...".

Creating your Benchtop Test Program

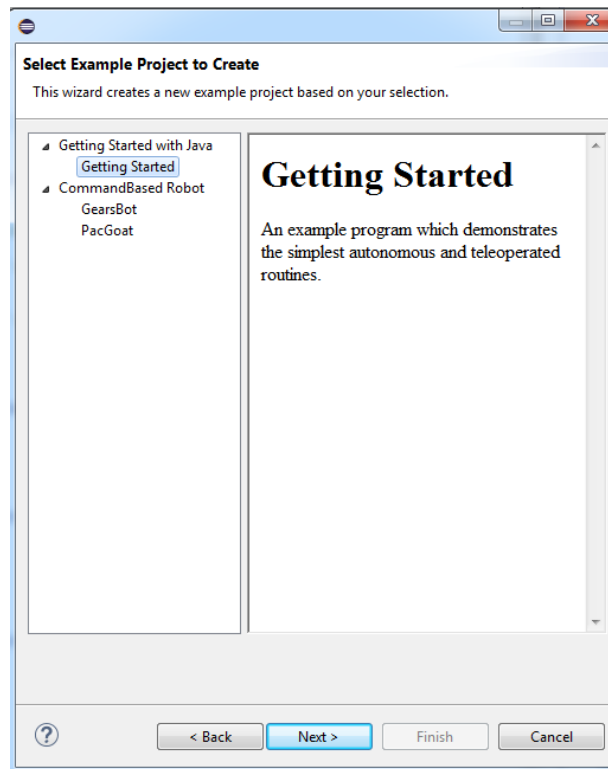
Selecting the project type (Robot Java Project)



Choose "Example Robot Java Project" as the project type.

Creating your Benchtop Test Program

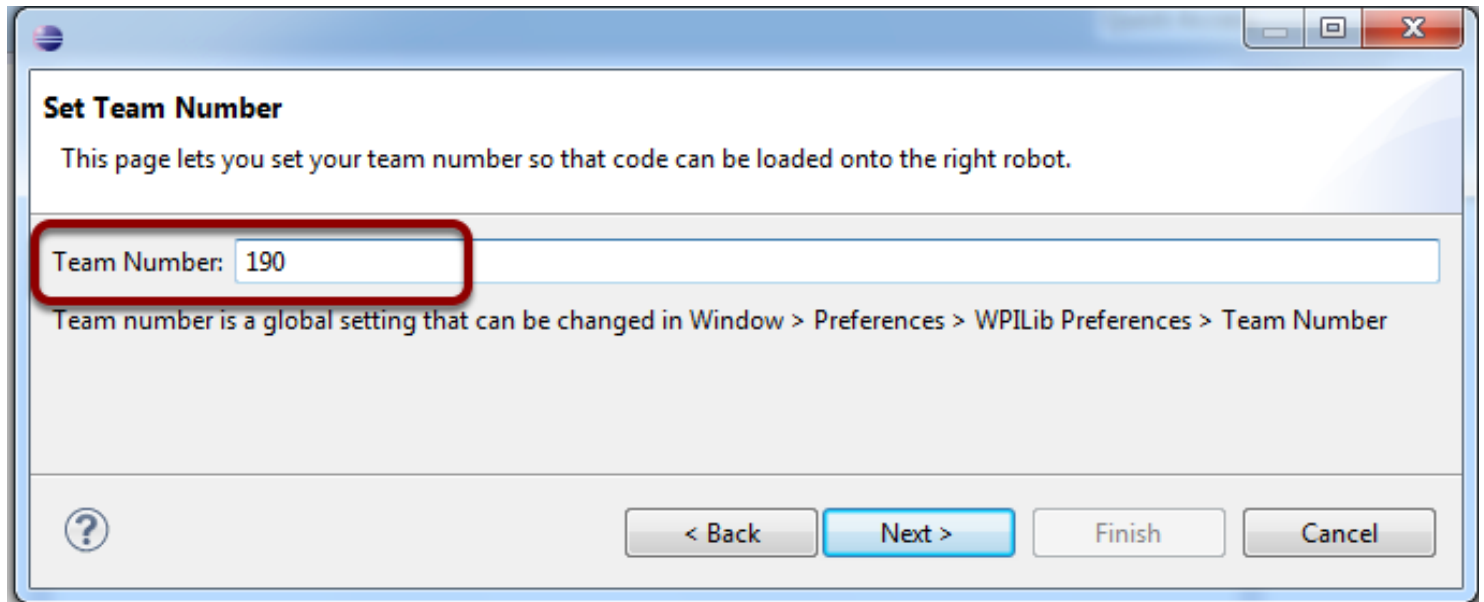
Selecting the Example



If necessary, click the arrow to expand the **Getting Started with Java** section, then click to select the **Getting Started** example. Then click **Next**.

Creating your Benchtop Test Program

Entering the team number



Enter your team number, then click **Next**. This is used when the Eclipse tools download programs onto your RoboRIO. As described in the dialog this is a global setting that can also be accessed from Window->Preferences->WPILib Preferences (if you need to change what team number you are targeting).

This dialog will only be shown if there is no team number currently set in Eclipse global settings.

Creating your Benchtop Test Program

The project is created in the Projects window in Eclipse



The project is created in the current workspace with the package name:

```
package org.usfirst.frc.team191.robot;
```

where your team number is substituted for 191 in this example. The project name is name of the sample robot project that was selected.

Notice that the project is now created (MyRobotProject) with a "src" folder that contains a Robot.java file that subclasses the appropriate base class for your template choice (note that the Command Based project will look a little different). You can double-click on the source file "Robot.java" to see the default code.

Creating your Benchtop Test Program

Defining the variables for our sample robot

Java

```
//Imports the other files needed by the program
import edu.wpi.first.wpilibj.IterativeRobot;
import edu.wpi.first.wpilibj.Joystick;
import edu.wpi.first.wpilibj.RobotDrive;
import edu.wpi.first.wpilibj.livewindow.LiveWindow;

public class Robot extends IterativeRobot {

    //Defines the variables as members of our Robot class
    RobotDrive myRobot;
    Joystick stick;
    Timer timer;

    //Initializes the variables in the robotInit method, this method is called when the robot
    is initializing
    public void robotInit() {
        myRobot = new RobotDrive(0,1);
        stick = new Joystick(1);
        timer = new Timer();
    }
}
```

The sample robot in our examples will have a joystick on USB port 1 for arcade drive and two motors on PWM ports 0 and 1. Here we create objects of type RobotDrive (myRobot), Joystick (stick), and Timer (timer).

Creating your Benchtop Test Program

Simple autonomous sample

Java

```
public void autonomousInit() { //This method is called once each time the robot enters autonomous mode
    timer.reset(); // Resets the timer to 0
    timer.start(); // Start counting
}

public void autonomousPeriodic() { //This method is called each time the robot receives a packet instructing the robot to be in autonomous enabled mode
    // Drive for 2 seconds
    if (timer.get() < 2.0) {
        myRobot.drive(-0.5, 0.0); // drive forwards half speed
    } else {
        myRobot.drive(0.0, 0.0); // stop robot
    }
}
```

Easy arcade drive for teleoperation

Java

```
public void teleopInit() { //The teleopInit method is called once each time the robot enters teleop mode
}

public void teleopPeriodic() { //The teleopPeriodic method is entered each time the robot receives a packet instructing it to be in teleoperated enabled mode
    myRobot.arcadeDrive(stick); //This line drives the robot using the values of the joystick and the motor controllers selected above
}
```

Creating your Benchtop Test Program

```
}
```

Test Mode

Java

```
public void testPeriodic() {  
    LiveWindow.run();  
}
```

Test Mode is used for testing robot functionality. The test mode of our sample program runs LiveWindow. LiveWindow is a part of the SmartDashboard that allows you to see inputs and control outputs on the robot from the dashboard when the robot is in Test Mode. You can read more about LiveWindow in the [SmartDashboard section](#) of the Driver Station Manual.