# What is WPILib

The WPI Robotics library (WPILib) is a set of software classes that interfaces with the hardware and software in your FRC robot's control system. There are classes to handle sensors, motor speed controllers, the driver station, and a number of other utility functions such as timing and field management. In addition, WPILib supports many commonly used sensors that are not in the kit, such as ultrasonic rangefinders.

What's included in the library
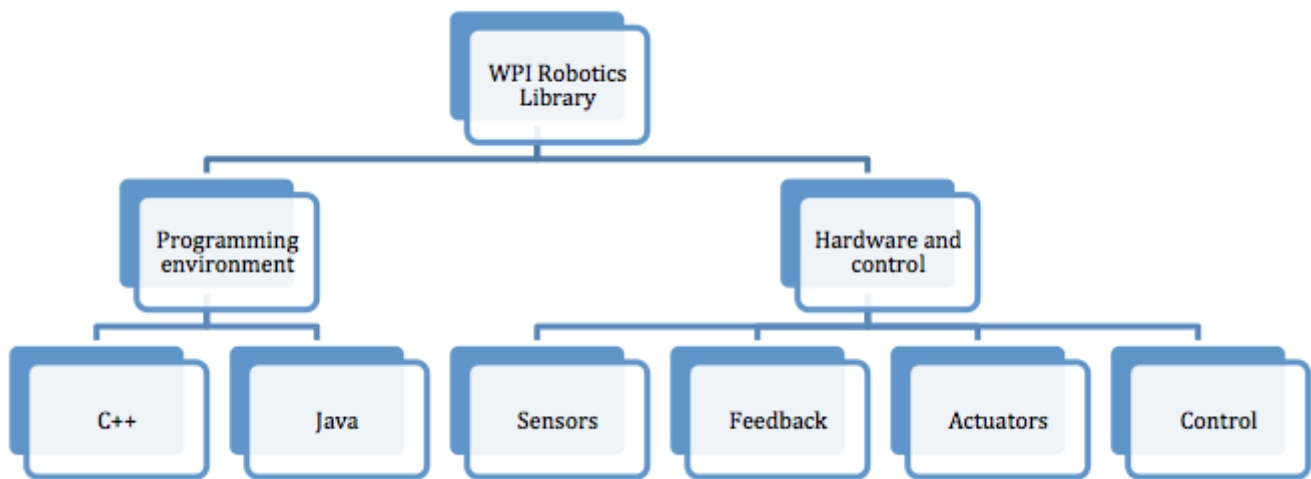
There are three versions of the library, one for each supported language. This document specifically deals with the text-based languages, C++ and Java. There is considerable effort to keep the APIs for Java and C++ very similar with class names and method names being the same. There are some differences that reflect language differences such as pointers vs. references, name case conventions, and some minor differences in functionality. These languages were chosen because they represent a good level of abstraction for robot programs than previously used languages. The WPI Robotics Library is designed for maximum extensibility and software reuse with these languages.

WPILib has a generalized set of features, such as general-purpose counters, to provide support for custom hardware and devices. The FPGA hardware also allows for interrupt processing to be

What is WPILib

dispatched at the task level, instead of as kernel interrupt handlers, reducing the complexity of many common real-time issues.

Fundamentally, C++ offers the highest performance possible for your robot programs. Java on the other hand has acceptable performance and includes extensive run-time checking of your program to make it much easier to debug and detect errors. Those with extensive programming experience can probably make their own choices, and beginning might do better with Java to take advantage of the ease of use.

There is a detailed list of the differences between C++ and Java on Wikipedia available here. Below is a summary of the differences that will most likely effect robot programs created with WPILib.

Java programming with WPILib

```
     public void autonomousInit() {
91       isAuto = true;
92       CommandBase.shooter.zeroRPMOffsets();
93       CommandBase.turret.zeroAngleOffsets();
94       // instantiate the command used for the autonomous period
95       autonomousCommand = (Command) (OI.getInstance().auton.getSelected());
96       // schedule the autonomous command (example)
97       autonomousCommand.start();
98   }
99
```

- Java objects must be allocated manually, but are freed automatically when no references remain.
- References to objects instead of pointers are used. All objects must be allocated with the new operator and are referenced using the dot (.) operator (e.g. gyro.getAngle()).
- Header files are not necessary and references are automatically resolved as the program is built.
- Only single inheritance is supported, but interfaces are added to Java to get most of the benefits that multiple inheritance provides.
- Checks for array subscripts out of bounds, uninitialized references to objects and other runtime errors that might occur in program development.
- Compiles to byte code for a virtual machine, and must be interpreted.

C++ programming with WPILib

```cpp
void Claw::Open() {
    victor->Set(1);
}

void Claw::Close() {
    victor->Set(-1);
}

void Claw::Stop() {
    victor->Set(0);
}
```

- Memory allocated and freed manually.
- Pointers, references, and local instances of objects.
- Header files and preprocessor used for including declarations in necessary parts of the program.
- Implements multiple inheritance where a class can be derived from several other classes, combining the behavior of all the base classes.
- Does not natively check for many common runtime errors.
- Highest performance on the platform, because it compiles directly to machine code for the PowerPC processor in the cRIO.