New for 2019!

This article describes the major changes to the Control System software for FRC 2019

Major Changes

- New IDE - VSCode (replacing Eclipse). This also includes a change from the Ant build system to GradleRIO
  - Vendor Libraries
- Pathweaver path planning software
- Raspberry PI image for image processing
- New Shuffleboard API
- New Java version - Java 11
- New roboRIO Webdashboard

Game Specifics

- Bandwidth - Bandwidth limit has been reduced to 4 Mbps. Like airline and hotel booking, the 7 Mbps limit included some assumptions that not all teams would use it. With the Sandstorm driving expected camera usage, we have lowered the limit in attempt to limit the likelihood teams will need to reduce below the limit at an event.
- Sandstorm Coding
  - Mode - Despite the Sandstorm not truly being an "Autonomous" mode, all code (DS, robot code, etc.) will continue to refer to this period as "Autonomous" or "Auto" for continuity.
  - Mode Transitions - The field will transition through modes in the same way as prior seasons. Prior to the start of the match, robot's will be in Disabled Mode. Once the match starts, they will transition to Autonomous Enabled mode. At the end of the Sandstorm, robots will briefly transition through Disabled (~.5s), if team's are driving their robot with joysticks/ gamepads at this time, there will be a noticeable hitch. Finally the robot will transition to Teleop Enabled mode for the remainder of the match until the buzzer sounds. When the match expires, the robot will return to Disabled mode.

New for 2019!

- Joystick Access - For this season only, joystick data will not be cached on the transition to Autonomous. Teams will be able to access updated joystick data throughout the Autonomous/Sandstorm period.
- Game Data - There is no game data for the 2019 season.

VS Code IDE (C++\Java)

For the 2019 season, FRC is moving from Eclipse to VS Code as the officially supported IDE for C++\Java development. This setup is easier for us to develop the extensions for and we expect it to be more streamlined and functional for the majority of teams. If VS Code doesn't seem to be your thing, the accompanying switch to GradleRIO should make it easier to use alternate IDEs as well (though this is not officially supported).

You can find information about the new IDE in the **Setting up the Development Environment** and **Creating and Running Robot Programs** sections of the C++ and Java manuals, including information about how to import old Eclipse projects into the new VSCode/GradleRIO system

Vendor Libraries

- With the change in IDE\Build System there was also a change to how Vendor libraries integrate with the system. For more information, see here: 3rd Party Libraries

RoboRIO WebDashboard

- No longer requires Silverlight
- No longer supports CAN device status/configuration
- Recommended to be used in non-IE browser

More information here

RoboRIO Imaging Tool

- Reworked UI
- Added firmware update capability

More information here

New for 2019!

CAN Device Configuration

- Removed from WebDashboard
- Provided by vendor specific tools.

See this article for updated PDP\PCM configuration instructions

WPILib C++\Java

- A new high level CAN API was added. This enables easier creation of custom CAN devices for users.
- The OSSerialPort class was removed. It was a bugfix for some Serial port bugs, but never actually worked anyway.
- SensorBase was removed. If you were deriving from SensorBase you should now derive from SendableBase (C++) or SendableBase+ErrorBase(Java). If you need methods previously contained in SensorBase, they are now in SensorUtil.
- The Watchdog class was added. This calls a callback function when the specified timeout is exceeded. This class is used by TimedRobot to warn on periodic loop time overruns (nominally 20ms) in user code. **Note: These Timed Robot messages are warnings. They inform you if your code overruns the specified timing. They are not errors.**
- IterativeRobot was deprecated for eventual removal. TimedRobot is a drop-in replacement that is strictly better with regard to execution jitter.
- Linux was updated to require Ubuntu 18.04 as its default OS to run WPILib tools
- Building the libraries:
  - All libraries require C++14 capability to build. On Linux, this means GCC 6, and on Windows MSVC 2015 or newer.
  - Linux was updated to require Ubuntu 18.04 as its default OS to link to the libraries.
  - Windows builds are done with MSVC 2017 update 9, so linking to the libraries requires 2017 update 9 or later.

WPILib C++

- The namespace frc shim was removed. WPILib classes need to be accessed by their full namespaces, or using statements added.
- **Headers were moved to an frc folder.**

New for 2019!

- HAL methods were removed from WPILib headers. In order to access HAL methods, their headers must be directly included.
- Llvm renamespaced to wpi, headers moved to wpi folder
- All classes now properly support move semantics (for those who know what that is and care).
- Doxygen comments were moved from the source files to the header files for easier reference.

WPILib Java

- The CameraServer class was moved to a new package, and the class in the old location was deprecated. Please move to the new package location.
- The JNI classes were moved to a new package.
- Any Java class that had a free method was changed to implement AutoClosable and have a close method instead.
- The main method was moved from being internal in WPILib to explicitly defined by user code. This removes the reflection loading of the main robot class, and makes changing your robot class a compile time error rather then a runtime error. **Teams should not need to edit the Main.java file.**

CameraServer (cscore)

- USB cameras are now supported on Windows desktop builds

All WPILib Tools (Shuffleboard, Robot Builder, etc.)

- Are now installed to ~home\frcYYYY\tools (where YYYY is the year and ~home on Windows is C:\Users\Public).
- Run ToolsUpdater.bat (Windows) or ToolsUpdater.py (Mac/Linux) or **Install Tools from GradleRIO** to install tools.
- Folder contains .vbs files for Windows and .sh files for Mac/Linux that should be used to run the program. This sets the program up to run using the FRC specific JDK (which it has been tested with).

New for 2019!

Shuffleboard

- New roboRIO API for automatically placing widgets on tabs and setting formatting options. More info here
- Camera viewer widget with adjustable stream parameters

PathWeaver\Pathfinder

- Added PathWeaver as UI to create paths for Pathfinder V1
- Generated paths are automatically downloaded to the RIO as part of the gradle configuration

Find more documentation on Pathweaver here

Raspberry Pi Image for Cameras

A pre-made Raspberry Pi Image for camera streaming/image processing has been developed to lower the barrier to entry to off-board vision processing. This image contains all of the libraries required to implement FRC compatible camera streaming, as well as a helpful web dashboard, read only file system configuration to handle robot power off and more. Learn more in the new manual here.

SmartDashboard

- Support for plugins has been removed. It was prohibitively difficult to maintain this feature when moving to Java 11 for the minimal number of teams believed to still be using it.
- Requires Java 11 to run.

Robot Builder

- Generates projects for the new VS Code\GradleRIO system
- Java/C++: encapsulates hardware in the specific subsystem. RobotMap is removed.
- C++: Updated to idiomatic C++ to match VSCode examples.

New for 2019!

Outline Viewer

- Requires Java 11 to run.