

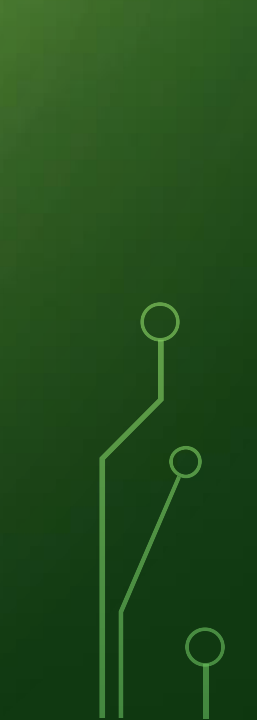


# FRC 6995 NOMAD

SOFTWARE TRAINING 101



# INTRODUCTIONS

- Sharon Rigg – Software Lead Mentor
  - Greg Shue – Software Mentor
  - Jeremiah & Ben S – Returning students
- 
- 
- 

# 2019 SOFTWARE TEAM VISION

- My vision is to enable you, the students, to be successful in developing the software for our robot.
- I will do what I can to provide the training you need to do your job. I am here to advise, but I will encourage you to work as a team and learn through trial and error.
- I will help provide structure to the planning process and scheduling tasks (I am a WEST after all!).

# HOW ARE ROBOTS DIFFERENT THAN MACHINES?

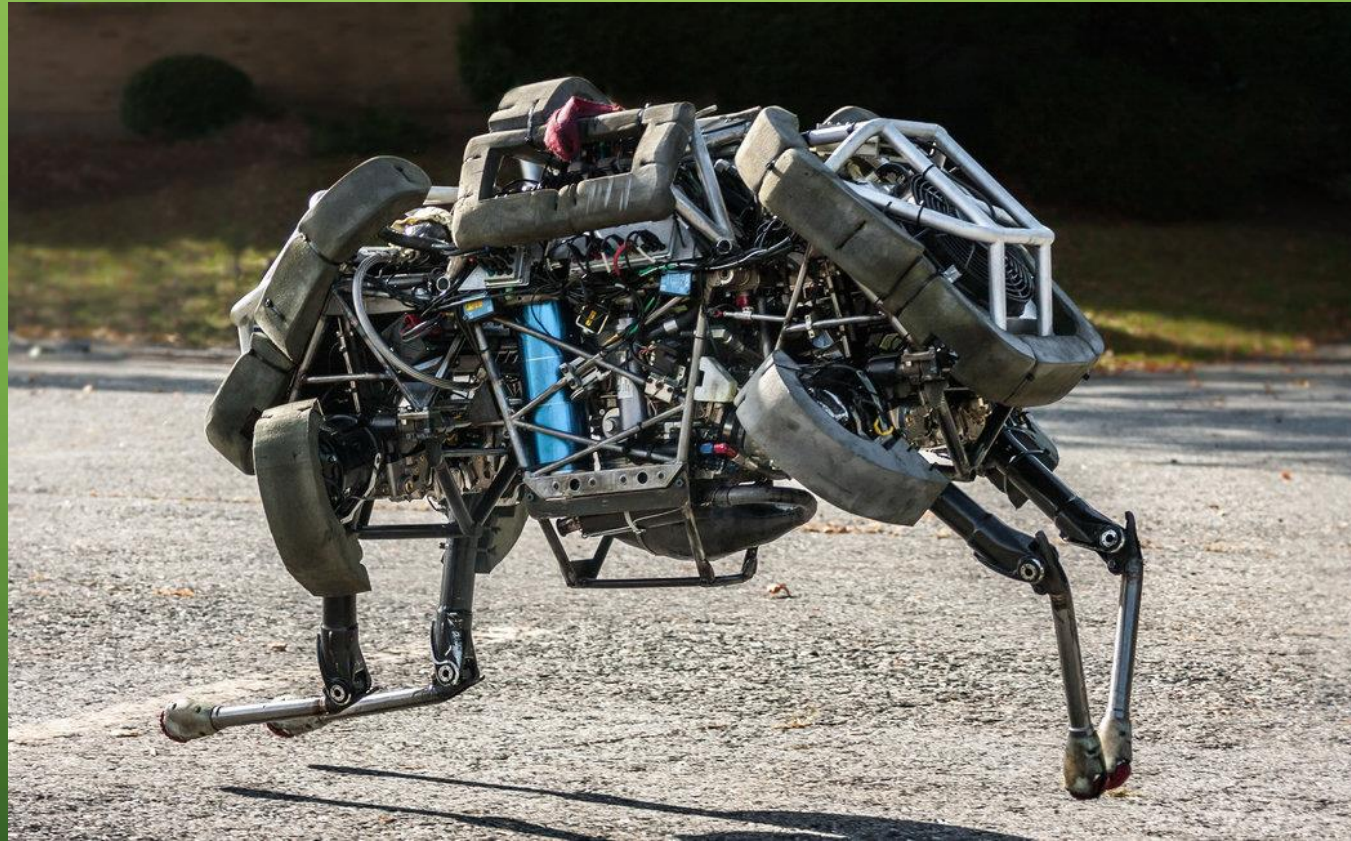


# HOW ARE ROBOTS DIFFERENT THAN MACHINES?

- Machines can precisely repeat predetermined motions automatically (like a washing machine or dishwasher).
- Robots can perform tasks either automatically or by remote control.



# HOW ARE ROBOTS USED TODAY?



# HOW ARE ROBOTS USED TODAY?

- Retrieve bombs
- Build things like cars, candy bars, and electronics.
- Used in medicine
- Military tactics
- Finding objects underwater
- Explore other planets.

# PARTS OF A ROBOT



**Most robots are composed of 3 main parts:**

- The Controller - also known as the "brain" which is run by a computer program. Often, the program is very detailed as it gives commands for the moving parts of the robot to follow.
- Mechanical Parts - motors, pistons, grippers, wheels, and gears that make the robot move, grab, turn, and lift. These parts are usually powered by air, water, or electricity.
- Sensors - to tell the robot about its surroundings. Sensors allow the robot to determine sizes, shapes, space between objects, direction, and other relations and properties of substances. Many robots can even identify the amount of pressure necessary to apply to grab an item without crushing it.

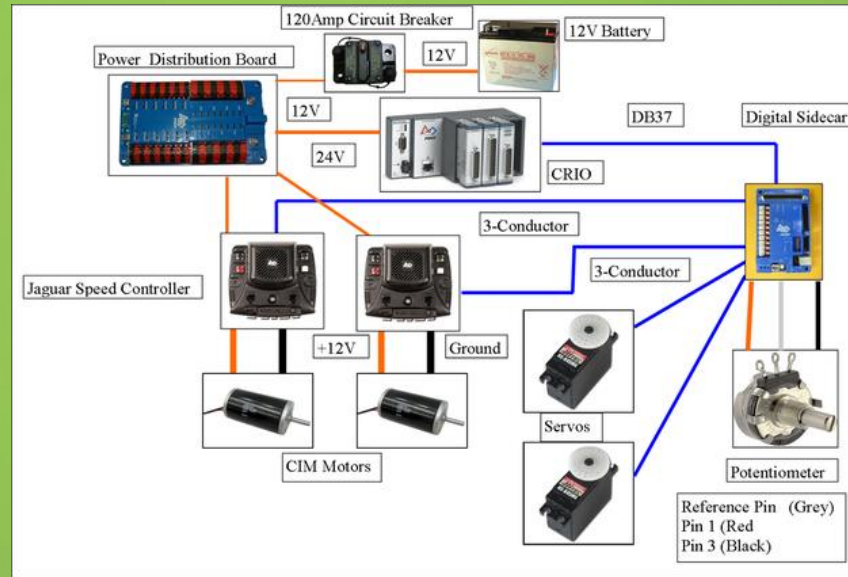


# THE FRC ROBOT



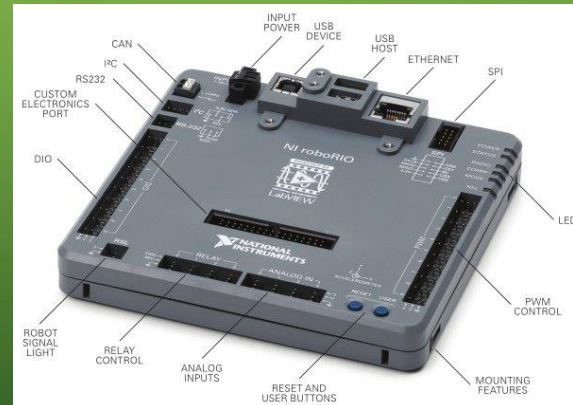
## Driver Station

Joystick, PC running Smart Dashboard



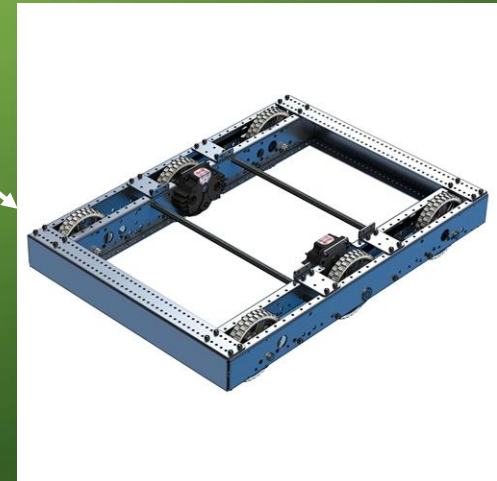
## Electronics

Gathers information or causes mechanics to move.  
Includes Limit Switches, Encoders, NavX, Spark, Talon, CIM, Solenoid, and more



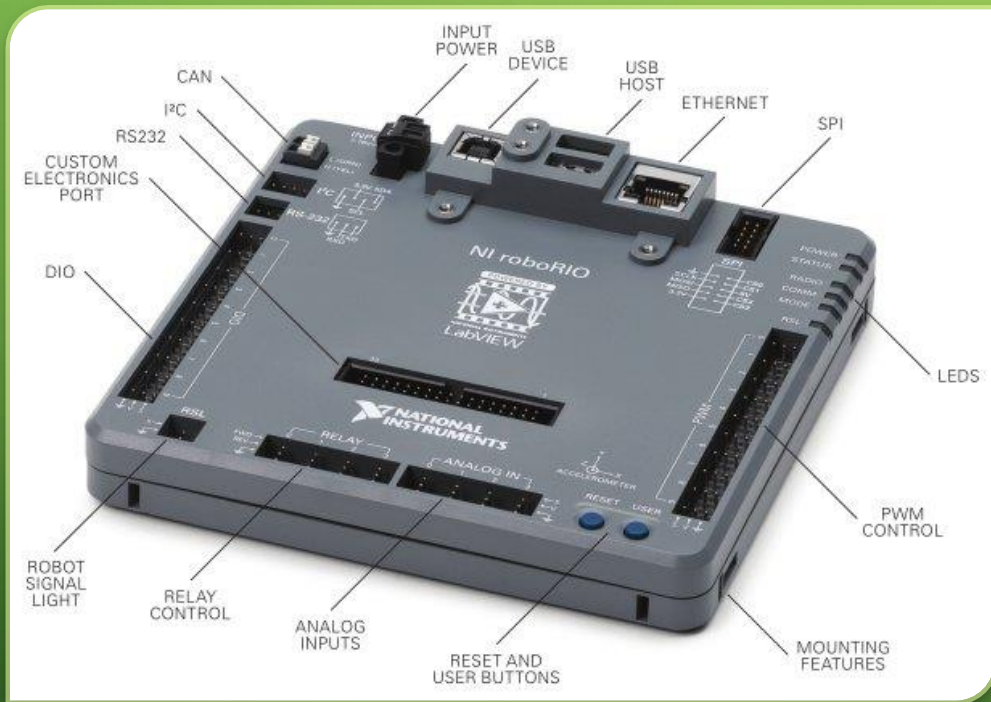
## roboRIO (brain)

Sends and receives information to/from the Electronics.



## Robot

# WHAT DOES THE SOFTWARE TEAM WORK ON?



- The software team will work directly with the brain of the robot
- In FRC, the controller/brain of the robot is the roboRIO (a mini computer)
- The code we write is transferred to the roboRIO.
- We must also have knowledge of the electrical and mechanical parts of the robot.

# OPERATION MODES

- Autonomous – first 15 seconds of a match. – The robot moves only by preset instructions (like a machine), no user interaction.
- Teleop – Drivers control the robot through use of external devices like a joystick.
- Disable – Robot is disabled
- Practice – This can be used to test out specific code or to run commands before a match starts (like charging up pneumatics).

# SOFTWARE TERMINOLOGY - IDE

## IDE (Integrated Development Environment) –

- An IDE is a software suite that consolidates basic tools required to write and test software such as
  - Text editors,
  - Code libraries
  - Compilers
  - Test platforms
- The integrated toolset is designed to simplify software development and can identify and minimize coding mistakes and typos.
- In 2018 we used Eclipse. In 2019 we will switch to Visual Studio Code  
(<https://wpilib.screenstepslive.com/s/currentCS/m/79833/l/932382-installing-vs-code>)



## SOFTWARE TERMINOLOGY - JAVA

- Java is an object-oriented programming language
- Java code can run on all platforms that support Java without the need for recompilation.
- It compiles to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture.

FRC Teams also have a choice of programming in LabView or C++.



# SOFTWARE TERMINOLOGY – COMMAND BASED

- Our robot is programmed as a command based robot.
- A command based robot has the following types of classes:
  - Subsystems classes – Subsystems break the robot into smaller parts. Each subsystem defines what the subsystem is made of and what that part of the robot can do.
    - Drive Train can move forward, backward and stop
    - Lifter can move up or down
    - Grabber can open or close
  - Command classes – Commands tell the robot to perform the actions

# SOFTWARE TERMINOLOGY - WPILIB

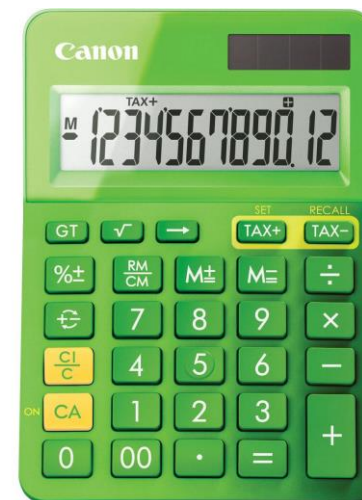
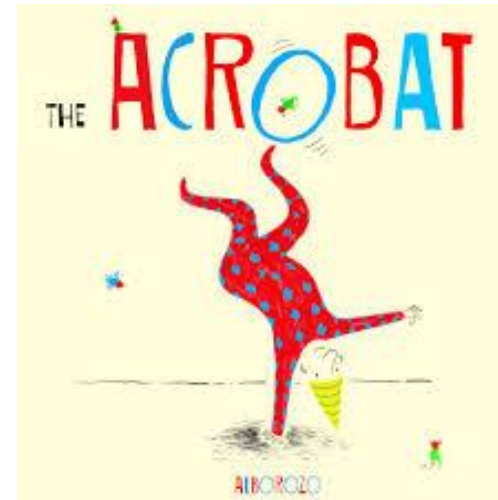
- WPILib is a robotics software library used in the FIRST Robotics Competition
- It was initially developed by WPI professor Brad Miller as a way to simplify program development for the FRC robot control system.
- It contains software that handles lower level details of the hardware (interrupt routines, voltage measurements and conversions, communications details, etc.) so students can focus on solving higher level problems.

<https://www.firstinspires.org/robotics/frc/blog/2019-wpilib>

# SOFTWARE DEVELOPMENT IN A NUTSHELL

- Write Java code in an IDE
- Compile Code
- Transfer code to roboRIO (via cable or radio signal)
- Test code on robot
- Upload changes to GitHub (more on that later)

# ROLE PLAYING EXERCISE



# ROLE PLAYING EXERCISE - CLASSES

- Acrobat
- Dice
- Blackboard
- Calculator
- Bamboozler
- LazyCalculator
- Decider



# ROLE PLAYING EXERCISE - METHODS

- Acrobat – `clap(x)`, `kneeBend(x)`, `count`
- Dice – `roll(x)`, `numRolls(x)`, `reset`
- Blackboard – `drawCircle`, `drawSquare`, `drawText(text)`, `clear`
- Calculator – `add(x,x)`, `subtract(x,x)`, `multiply(x,x)`
- Bamboozler - `add(x,x)`, `subtract(x,x)`,
- LazyCalculator – `add(x,x)`, `add(x,x,x,x)`
- Decider – `dayOfWeek(date)`, `rate(text)`

# ROLE PLAYING EXERCISE - OBJECT

- ?? is a class type of Acrobat
- ?? is a class type of Dice
- ?? and ?? is a class type of Blackboard
- ?? is a class type of Calculator
- ?? is a class type of Bamboozler
- ?? is a class type of LazyCalculator
- ?? is a class type of Decider

# ROLE PLAYING EXERCISE - COMMANDS

- clap
- kneebends
- count
- tripleBackFlip
- roll
- numRolls
- drawSquare
- drawCircle
- drawText
- clear
- 
- add
- subtract
- multiply
- dayOfWeek
- rate

# ROLE PLAYING EXERCISE

Java syntax for giving commands to (or requesting service from) an object:

- `<acrobat>.clap(3)`
- `<dice>.roll()`
- `<calculator>.add(10, 4)`
- `<Decider>.rate("Ginger Spice")`