FRC Programming 101

MAGMA Robotics
Donald Kaulukukui
Sep 2019

Intro

Me: Electrical Engineer, Embedded Processing, University of Hawaii. Systems Engineer with US Navy. Been supporting Team MAGMA since 2018.

Goals of this presentation is to:

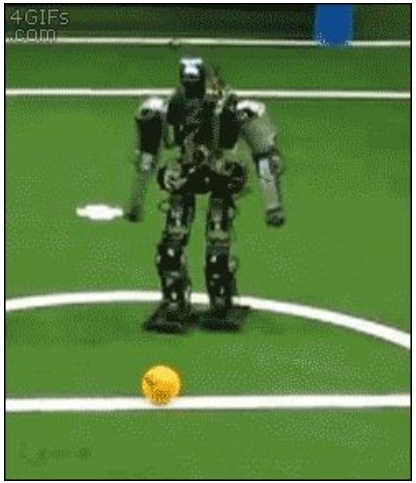
1. Tell you what we are trying to do
    a. Program a robot
2. Tell you how we are going to do it
    a. JAVA programming language
    b. FRC system, tools, code organization
3. Point you in the right direction
    a. Resources to further learning

Topics

- End Goal
- What is programming?
- Programming Language
- Robot Programming Steps
- Example

End Goal

- Get the Robot “Bot” to do what we want
- Collect information (inputs) and take actions (outputs) accordingly.
  - Inputs: Buttons, Sensors, Field Management Station
  - Outputs: Motors, Servos, pumps, solenoids, valves, lights, etc



What is Programming?

Programming is a way to “instruct the computer to perform various tasks”.

“Instruct the computer”: this basically means that you provide the computer a set of instructions that are written in a language that the computer can understand. The instructions could be of various types. For example:

- Check the value of an input
- Turn a motor on/off

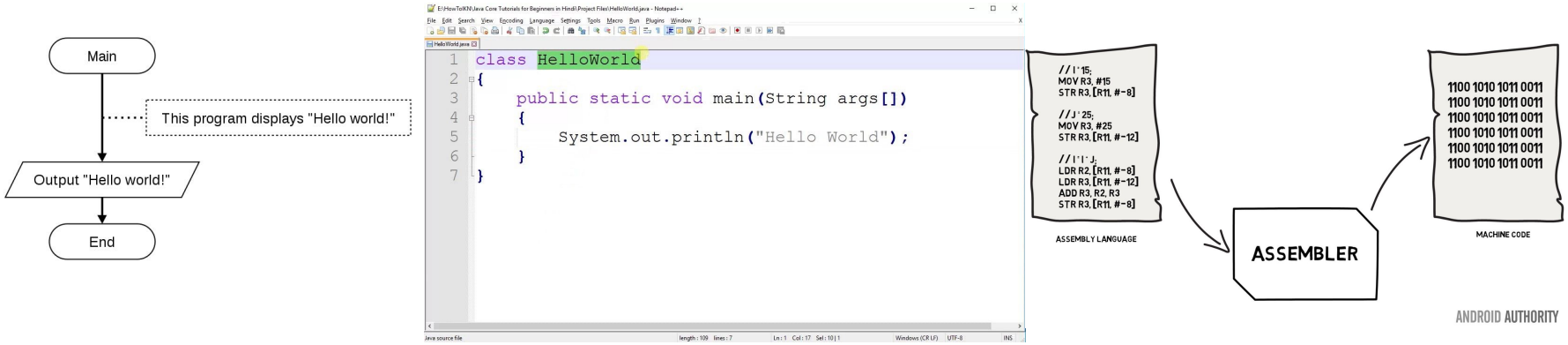
“Perform various tasks”: the tasks could be simple ones like we discussed above (adding 2 numbers, rounding off a number) or complex ones which may involve a sequence of multiple instructions. For example:

- Move forward when joystick is pushed forward
- Shoot a ball when a button is pressed

What is Programming?

Programming languages exist to make human readable logic that can be translated into machine readable code.

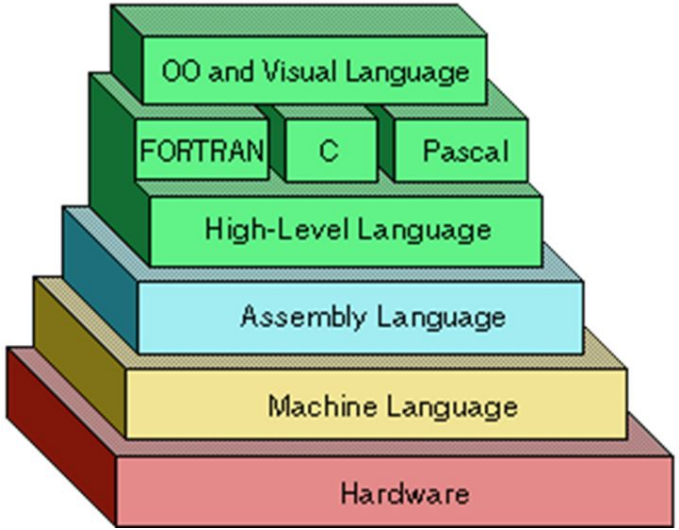
Pseudo Code -> Programming Language - > Assembly Code -> Machine Instruction



From Words to Bits to Voltages

A **programming language** is a formal language, which comprises a set of instructions that produce various kinds of output. Programming languages are used in computer programming to implement algorithms.

A **compiler** is a computer program that translates computer code written in one programming language (the source language) into another language (the target language). The name *compiler* is primarily used for programs that translate source code from a high-level programming language to a lower level language (e.g., assembly language, object code, or machine code) to create an executable program.



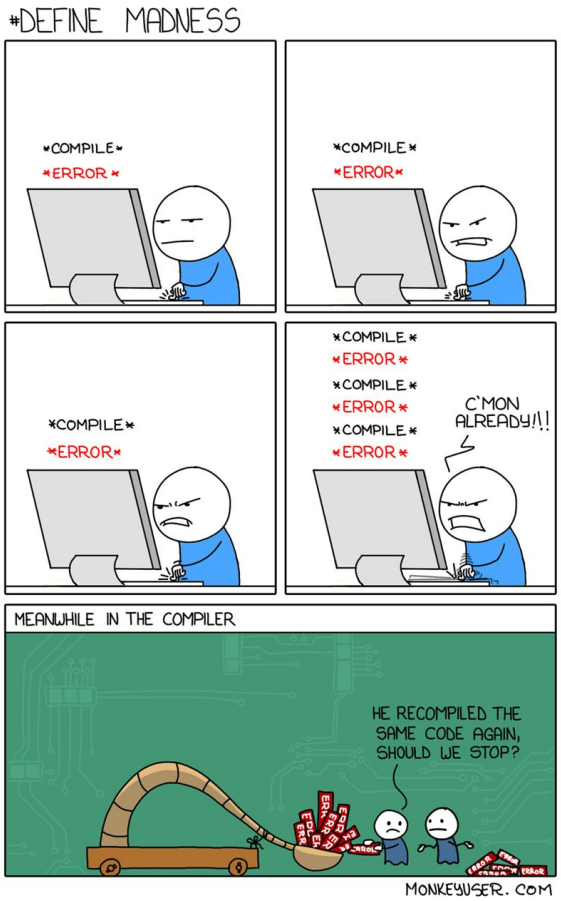
Programming Language

FRC options for programming are JAVA, C/C++ or LabView (Visual)

Team MAGMA has chosen the JAVA language

The advantages of Java are as follows:

- Java is easy to learn.
  - Java was designed to be easy to use and is therefore easy to write, compile, debug, and learn than other programming languages.
- Java is platform-independent.
  - One of the most significant advantages of Java is its ability to move easily from one computer system to another. The ability to run the same program on many different systems is crucial to World Wide Web software, and Java succeeds at this by being platform-independent at both the source and binary levels.
- Java is object-oriented. (*advanced topic but just be aware that we are using it)
  - This allows you to create modular programs and reusable code.



Learning Java

Learning to code takes time and there is no substitute for hands on exercises.

I cannot teach you to program in 25 minutes, or 50 minutes. 500 minutes would be a challenge.

I can point you in the right direction.

Here are some basic JAVA programming tutorials online:

- Code Academy: https://www.codecademy.com/learn/learn-java
- Tutorial: https://www.tutorialspoint.com/java/index.htm

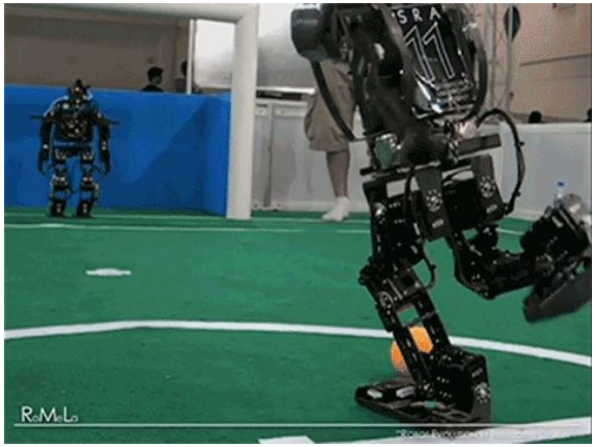
Robot Programming Steps

1. Plan and organize (paper and pencil)
2. Write code*
3. Load code *
4. Test code*

* Code writing/loading/testing should be done recursively in small increments

Planning and Organizing

1. Document HW
2. Map out inputs, outputs
3. Determine Robot SubSystems
4. Determine Robot Commands
5. Code organizational structure
6. Pseudo-code



All of the above needs to be done prior to writing any computer code. Pencil and paper work perfectly fine and will help to facilitate the code writing later on.

Documenting Hardware

- Inputs
  - Joystick
  - Camera
  - Sensors
    - Limit switches
    - Potentiometers
    - Encoders
- Outputs
  - Motors - type, function
  - Pneumatics
  - Solenoids

The output product from this step will be a complete list of all hardware that will be used on the robot.

Make special note of what protocol each HW item will use to communicate the the RoboRIO.

MAP INPUTS and OUTPUTS

For each input determine:

- What protocol it uses and how it should connect to the RoboRIO
- What HW port it will plug into
  - USB, Serial, CANBUS, I/O, etc.

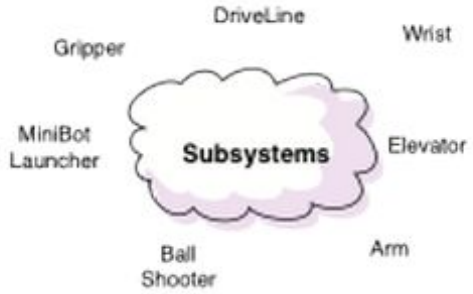
For each output determine:

- What protocol it uses and how it should connect to the RoboRIO
- What HW port it will plug into
  - USB, Serial, CANBUS, I/O, etc.

The output product from this step will be a Port I/O MAP for the RoboRIO pins and which HW will be connected.

FRC Command Based Programming

- Subsystem: define the capabilities of each part of the robot
- Commands: Commands let you break up the tasks of operating the robot into small chunks



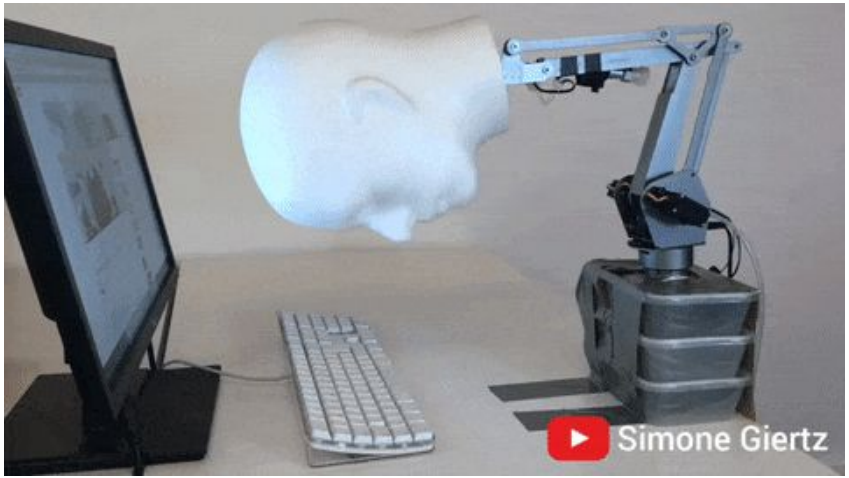
Determine Robot Subsystems

Group all HW into SubSystems depending on what functions they will be supporting.

Examples:

- Drivetrain
  - 4 motors
- Claw
  - 1 motor
  - 1 pneumatic valve
- Elevator_lift
  - 2 motors
  - 2 encoders
- Ball_Shooter
  - 1 motor

The output product from this step will be a List of SubSystems with all output HW accounted for under one of the subsystems.



Determine Robot Commands

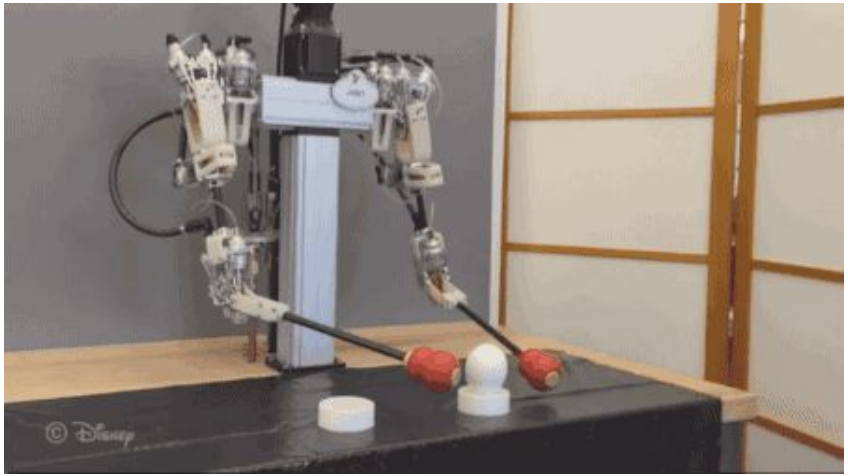
For each SubSystem determine the list of all of the commands that the subsystem will need to do.

Extra care and thought should be given to making commands reusable for reusability and any potential autonomous modes.

Example:

- Drivetrain
  - DriveForward
  - DriveBackward
  - TurnLeft
  - TurnRight

The output product from this step will be a complete list of commands for each subsystem.



Code Organizational Structure

- In this step you will need to plan how you will organize your code.
- List out files and what will be included in each file.
- If you were starting from scratch the Object Oriented Classes would be determined here.
- Luckily for us FRC has clearly defined files and structure with the Command Based Robot Structure.
    - Subsystems each have their own file
    - Commands each have their own file
    - Other files will be explained next lesson.

The output product from this step will be a complete list the Class and file structure for the code you are writing.

This step is simplified for us within FRC and will simply be a list of all subsystems and commands following the FRC command based robot file structure.

Pseudo Code

For each of the files identified we should go through and write out (on pen and paper) what each will do.

Example:

DriveForward (Command)

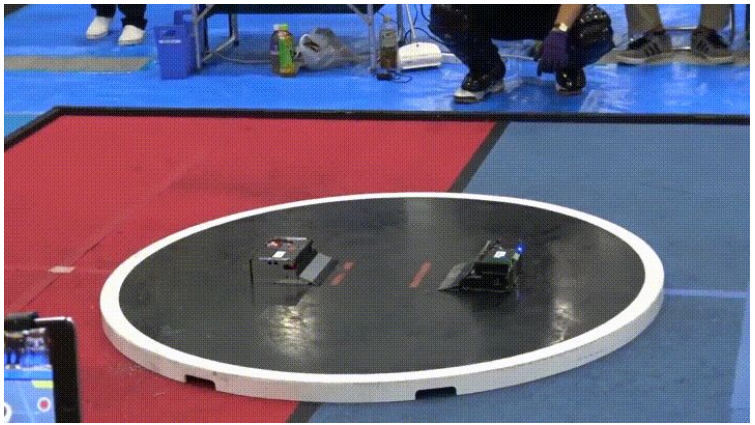
Turn ALL Motors On in forward direction

TurnLEFT

Turn Right side motors on in forward direction

Turn Left side motors on in reverse direction

The output product from this step will be pseudo code for each file that will be coded.



Before you write any code

You should have the following before you even think about touching any computer code:

1. Complete list of HW
2. Port Mapping of HW to I/O ports on the RoboRIO
3. List of all SubSystems and what HW each has
4. List of all commands for each SubSystem
5. Organize code (already done for us)
6. Pseudo code for each command and subsystem

Example - Tank Drive Step 1

Command Based Tank Drive

As an example we will walk through the steps needed for a simple drivable tank drive robot.

STEP 1: Define HW

- 4 brushed DC motors two on left side, two on the right side.
  - Using old regular SPARK motor controllers.
  - Using PWM
- Joystick controller
  - USB via FRC Driver station

Example - Tank Drive Step 2

Step 2: Determine I/O ports

- 4 motors each connect to SPARK motor controllers
  - Left Motor 1 - connect to RoboRIO DigitalOutput 0
  - Left Motor 2 - connect to RoboRIO DigitalOutput 1
  - Right Motor 1 - connect to RoboRIO DigitalOutput 2
  - Right Motor 2 - connect to RoboRIO DigitalOutput 3
- Joystick connects to driver station USB as stick 0

Example - Tank Drive Step 3

Step 3: Define Subsystems

- Drivetrain Subsystem
  - All 4 motors

Alternatively you can have two subsystems

- Left_Drivetrain
  - Left side motors
- Right_Drivtrain
  - Right side motors

Example - Tank Drive Step 4

Step 4: Define Commands

- DriveForward
- Drive Backward
- TurnLeft
- TurnRight

Alternatively

- DriveForward_speed(at X speed)
- DriveForward_Distance(distance)
- TurnLeft_Angle (angle)

Example - Tank Drive Step 5/6

Step 5: Organize code (we will use FRC Command Based Robot Code Organization)

Step 6: Psuedocode

SubSystem:

Drivetrain_SubSystem

- Define motor HW ports
- Default state is all motors turned off

Commands:

DriveForward

- Turn All motors on in forward direction

Example - Tank Drive Step 6 continued...

Step 5: Psuedocode

Commands:

DriveForward_Speed:

- Input = Speed
- Turn all motors on at X speed

DriveForward_distance:

- Input = distance
- Read encoders value
- Turn all motors on
- Check new encoder value and calculate distance travelled
- If distance traveled is <= desired distance then stay on, otherwise turn off.

Next Lesson...

In the next lesson we will go over the specific file structure used within a JAVA Command Based Robot program using the TankDrive example we just completed.

If time permits and a test chassis is available I will show you the process on how to load the program onto the robot.

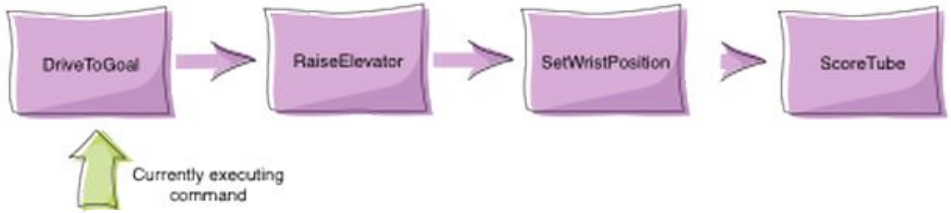
Resources

- JAVA programming:
  - Tutorial: https://www.tutorialspoint.com/java/index.htm
  - Code Academy: https://www.codecademy.com/learn/learn-java
  -
- FRC
  - Command Based Programming: https://wpilib.screenstepslive.com/s/currentCS/m/java/l/599732-what-is-command-based-programming
  - FRC Programming Done Right: https://frc-pdr.readthedocs.io/en/latest/index.html

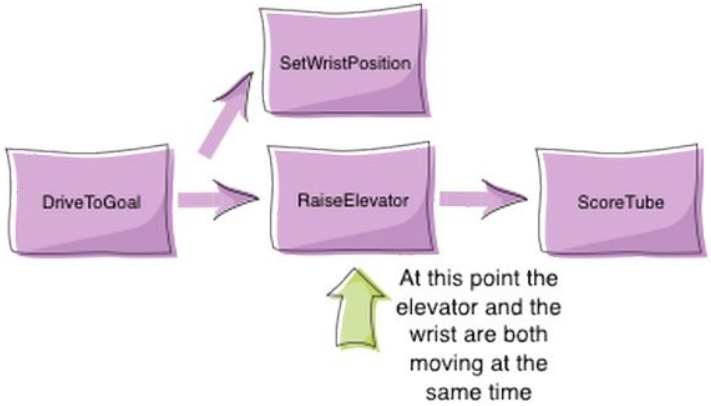
BACKUP SLIDES

Executing commands

Sequentially:



Concurrently:



JAVA Programming Language