Our Robot

East Palo Alto Robotics 8048

Our Approach

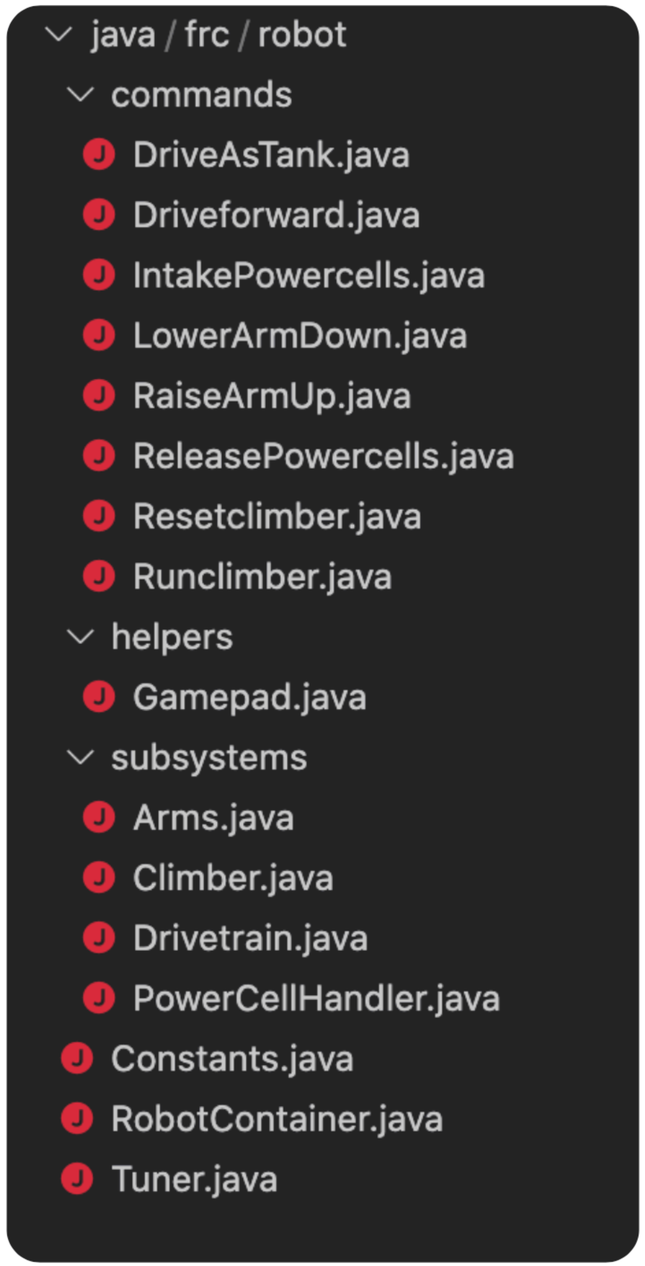
- Figured out the details of the challenge
- Initial brainstorming sessions
- We used the FIRST rookie kit

Why did we choose the Everybot mechanical design

- Looked at the intricate game
- Determined our teams capabilities
- It used a Standard simple drive train, that we had built before
- Scoring advantages
- Defensive capabilities
- Bypass prototyping and design
- Get to competition ready robot faster

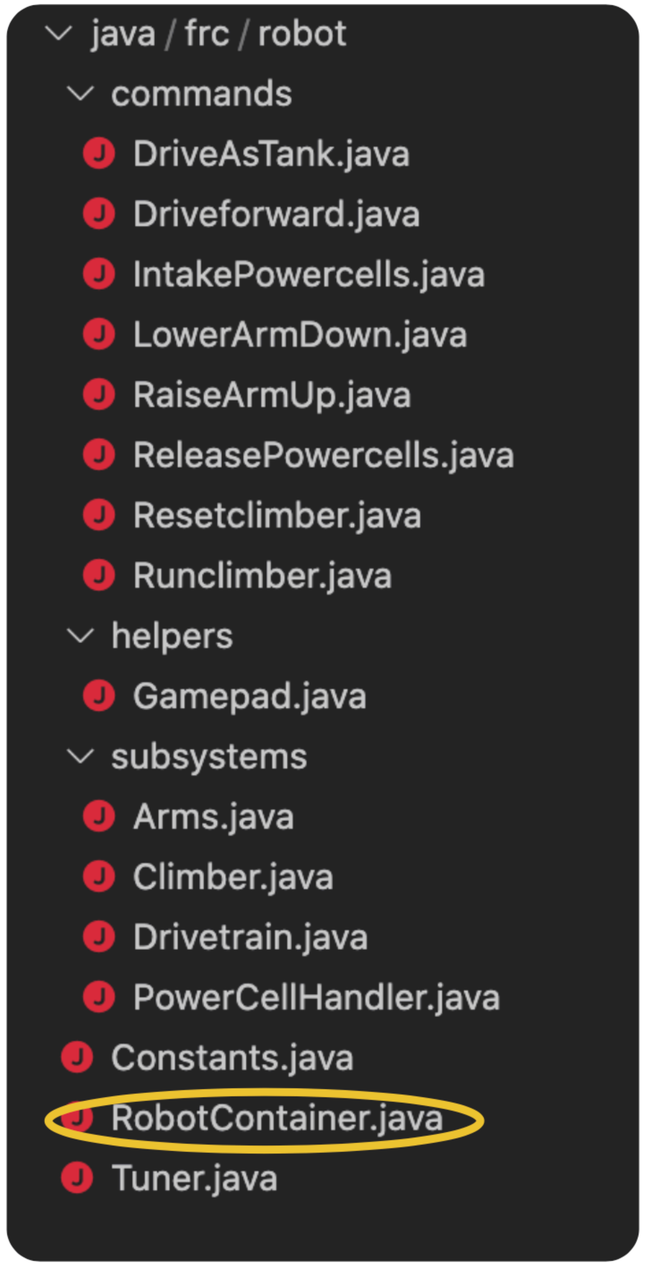
Software

- We decided to use our own software
- modified software we used for an off season competition in 2019 (cal games)
- We implemented modular programing
    - Easier to modify/read/troubleshoot



Software

- We decided to use our own software
- modified software we used for an off season competition in 2019 (cal games)
- We implemented modular programing
    - Easier to modify/read/troubleshoot



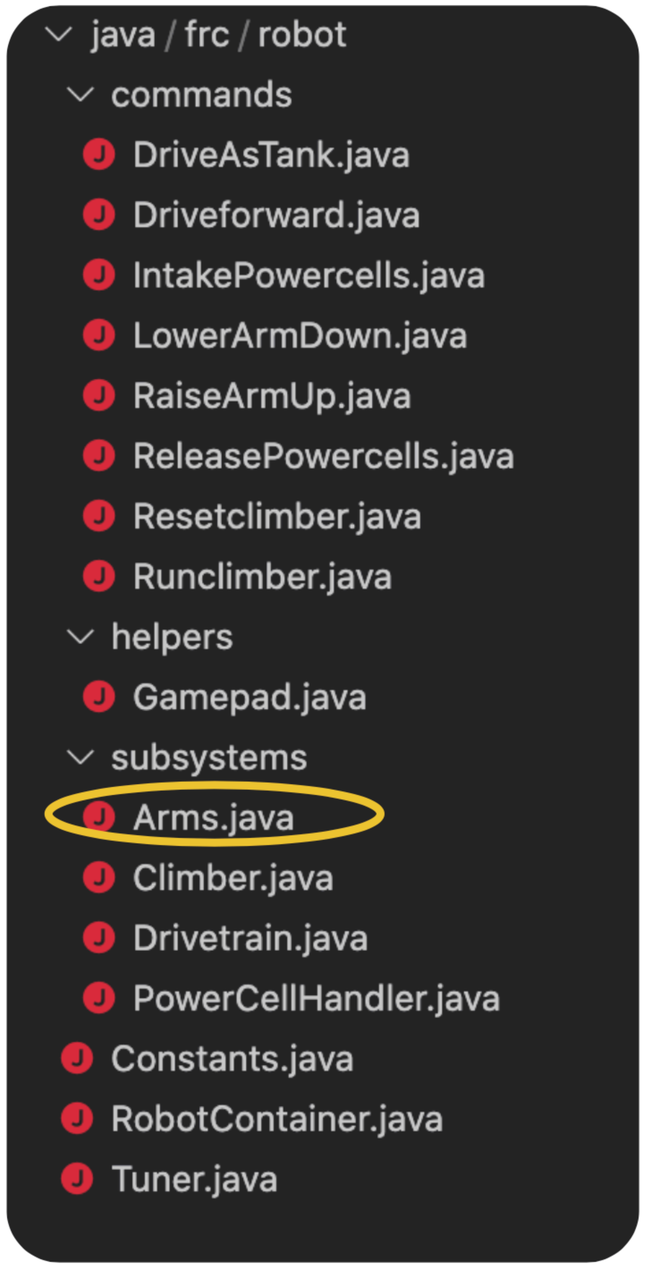
Software

- We decided to use our own software
- modified software we used for an off season competition in 2019 (cal games)
- We implemented modular programing
    - Easier to modify/read/troubleshoot

```
blic class RobotContainer {

public final Command autonomousCommand;

public RobotContainer() {

  // Default select the Tuner tab
  Shuffleboard.selectTab("game");

  // Connect to all the inputs (gamepads and shuffleboard).
  Gamepad driverGamepad = new Gamepad(Constants.driverGamepadPort);
  Gamepad operatorGamepad = new Gamepad(Constants.operatorGamepadPort);
  CameraServer.getInstance().startAutomaticCapture();

  // Connect to all the outputs.
  Arms arms = new Arms();
  PowerCellHandler powerCellHandler = new PowerCellHandler();
  Drivetrain drivetrain = new Drivetrain();
  Climber climber = new Climber();


  // Describe when the commands should be scheduled.
  this.autonomousCommand = new Driveforward(drivetrain);

  operatorGamepad.xButton.whenPressed(new RaiseArmUp(arms));
  operatorGamepad.aButton.whenPressed(new LowerArmDown(arms));
  operatorGamepad.leftBumper.whileHeld(new IntakePowercells(powerCellHandler));
  operatorGamepad.rightBumper.whileHeld(new ReleasePowercells(powerCellHandler));
  operatorGamepad.getDualButton(operatorGamepad.backButton, operatorGamepad.startButton)
      .whileHeld(new Runclimber(climber));
  arms.setDefaultCommand(new RaiseArmUp(arms));
  drivetrain.setDefaultCommand(new DriveAsTank(drivetrain, driverGamepad.leftYAxis, driverGamepad.rightYAxis,
      driverGamepad.rightAnalogTrigger));
  SmartDashboard.putData("Reset Climber", new Resetclimber(climber));
}

public Command getAutonomousCommand() {
  return this.autonomousCommand;
}
```

Software

- We decided to use our own software
- modified software we used for an off season competition in 2019 (cal games)
- We implemented modular programing
    - Easier to modify/read/troubleshoot



Software

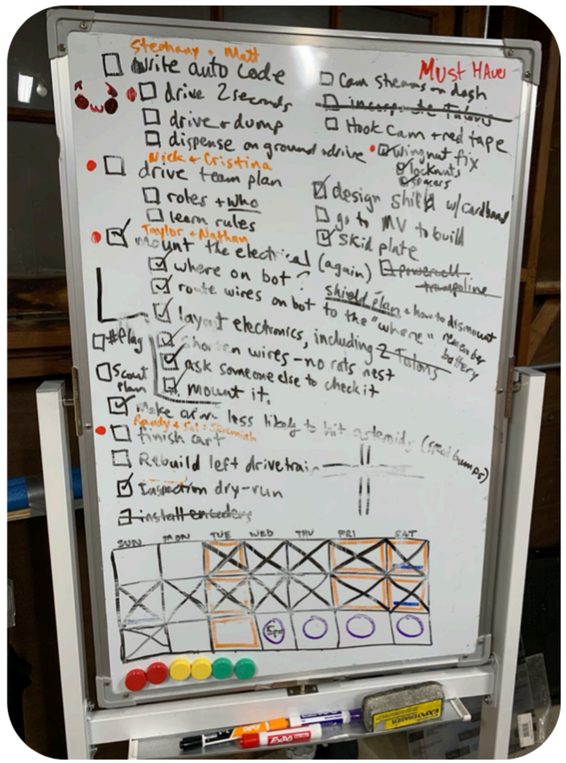
- We decided to use our own software
- modified software we used for an off season competition in 2019 (cal games)
- We implemented modular programing
    - Easier to modify/read/troubleshoot

```
public class Arms extends SubsystemBase {

  PWMVictorSPX armMotor = new PWMVictorSPX(Constants.armMotorPWM);

  public Arms() {}

  public void moveUp() {
    this.armMotor.set(Tuner.getPowerToMoveArm());
  }

  public void moveDown() {
    this.armMotor.set(-1 * Tuner.getPowerToMoveArm());
  }

  public void holdUp() {
    this.armMotor.set(Tuner.getPowerToHoldUpArm());
  }

  public void holdDown() {
    this.armMotor.set(-1 * Tuner.getPowerToHoldDownArm());
  }
```

Capabilities/Process

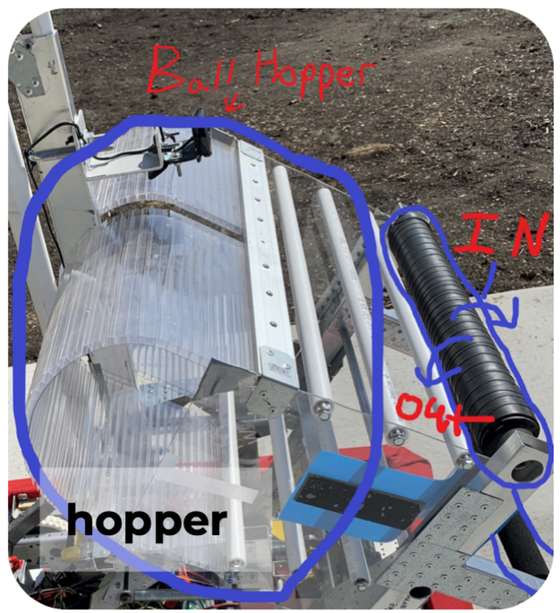
Our robot can:

- Pick up and store power cells
- Score lower goal
- Defend
- Climb



The process:

- Created sub groups w/ leads
- Delegated different parts
- Prioritize important tasks

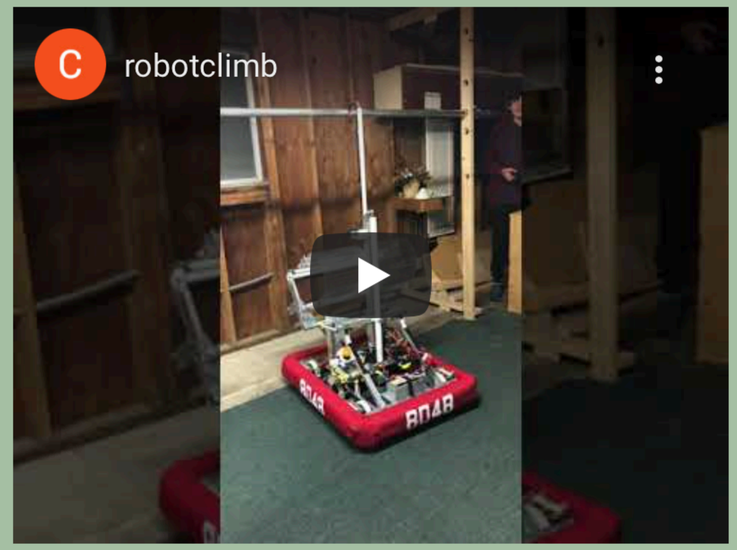


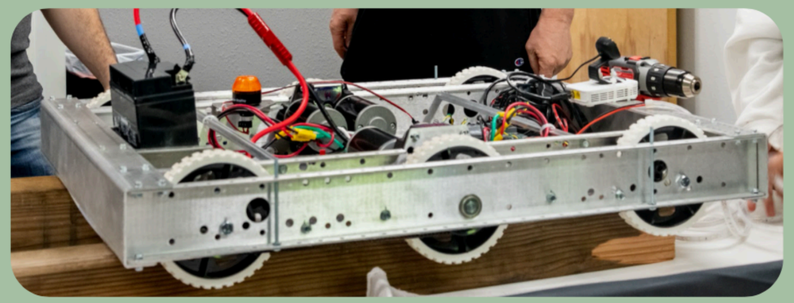
The Features

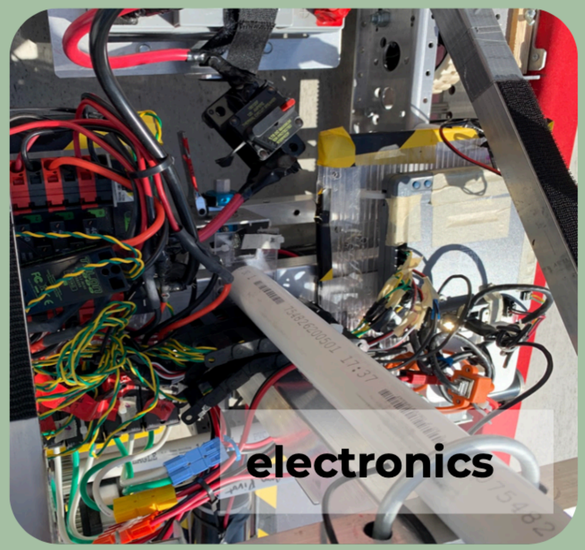
- **Ball intake**
  - Rotating taped roller that intakes ball into ball hopper and dispenses into goal
- **Ball hopper**
  - Container for collected power cells that are then shot out
- **Arm mechanism**
  - Raises hopper/roller up and down
  - Holds up ball hopper and intake being able to rotate it up and down
  - To fit max dimensions

The Features - Cont.

- **Hanging mechanism**
  - A latch that raises when activated. Can pull up the robot off the floor when
    - latched onto switch
- **Electronics**
  - Wireless router connected to driver station
  - roboRIO controller: runs code to operate robot/runs interfaces
  - Interface runs motor controllers and sensors
  - Power distribution panel
  - Safety features
- **Drive train**
  - Includes:
    - Two drive motors per side
    - Two drive belt per side
    - One gearbox per side
    - Three driven wheels per side with the center two wheels dropped lower than the rest (westcoast drive)

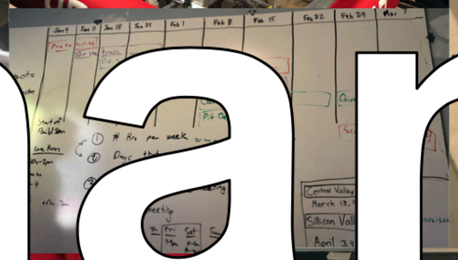
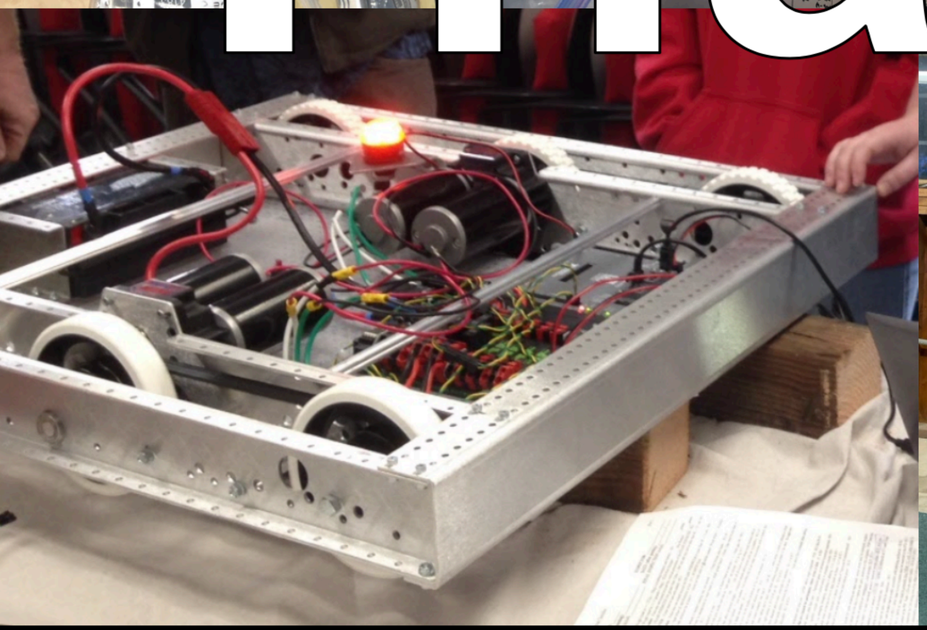
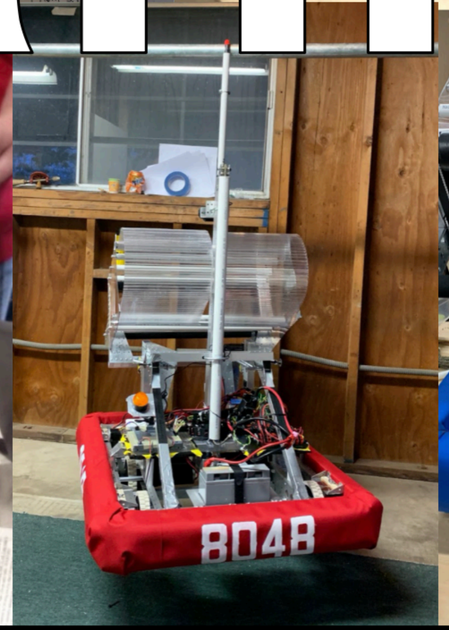


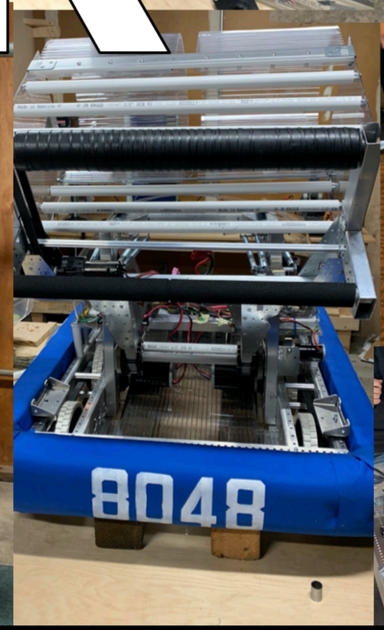


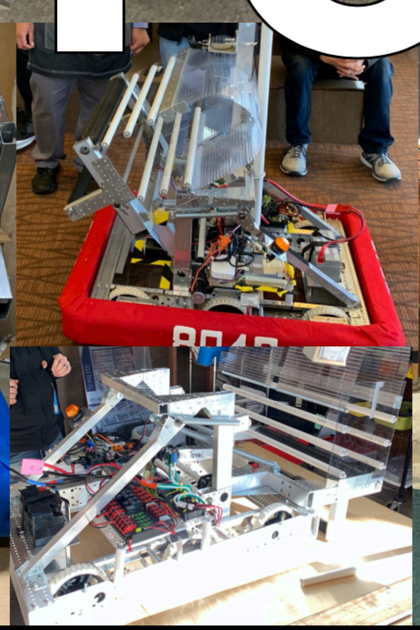










Thank You