

Simple subsystems

Subsystems are the parts of your robot that are independently controlled like collectors, shooters, drive bases, elevators, arms, wrists, grippers, etc. Each subsystem is coded as an instance of the Subsystem class. Subsystems should have methods that define the operation of the actuators and sensors but not more complex behavior that happens over time.

Creating a subsystem

Java

```
import edu.wpi.first.wpilibj.*;
import edu.wpi.first.wpilibj.command.Subsystem;
import org.usfirst.frc.team1.robot.RobotMap;

public class Claw extends Subsystem {

    Victor motor = RobotMap.clawMotor;

    public void initDefaultCommand() {
    }

    public void open() {
        motor.set(1);
    }

    public void close() {
        motor.set(-1);
    }

    public void stop() {
        motor.set(0);
    }
}
```

Simple subsystems

This is an example of a fairly straightforward subsystem that operates a claw on a robot. The claw mechanism has a single motor to open or close the claw and no sensors (not necessarily a good idea in practice, but works for the example). The idea is that the open and close operations are simply timed. There are three methods, `open()`, `close()`, and `stop()` that operate the claw motor. Notice that there is not specific code that actually checks if the claw is opened or closed. The `open` method gets the claw moving in the open direction and the `close` method gets the claw moving in the close direction. Use a command to control the timing of this operation to make sure that the claw opens and closes for a specific period of time.

Operating the claw with a command

Java

```
package org.usfirst.frc.team1.robot.commands;

import edu.wpi.first.wpilibj.command.Command;
import org.usfirst.frc.team1.robot.Robot;
/**
 *
 */
public class OpenClaw extends Command {

    public OpenClaw() {
        requires(Robot.claw);
        setTimeout(.9);
    }

    protected void initialize() {
        Robot.claw.open()
    }

    protected void execute() {
    }

    protected boolean isFinished() {
```

Simple subsystems

```
        return isTimedOut();
    }

    protected void end() {
        Robot.claw.stop();
    }

    protected void interrupted() {
        end();
    }
}
```

Commands provide the timing of the subsystems operations. Each command would do a different operation with the subsystem, the Claw in this case. The commands provides the timing for opening or closing. Here is an example of a simple Command that controls the opening of the claw. Notice that a timeout is set for this command (0.9 seconds) to time the opening of the claw and a check for the time in the `isFinished()` method. You can find more details in the article about [using commands](#).