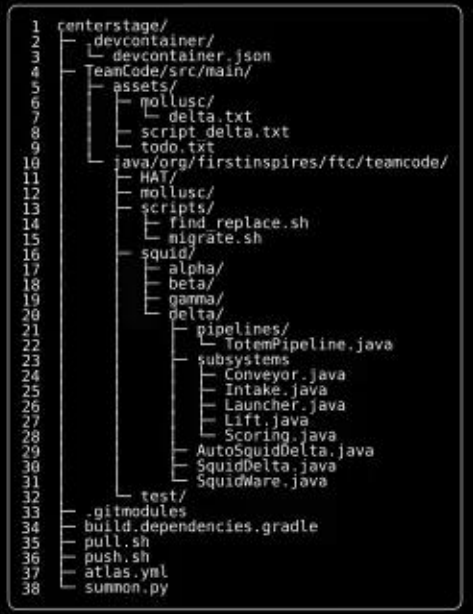
Trobotix CENTERSTAGE Source Code Overview

Contents

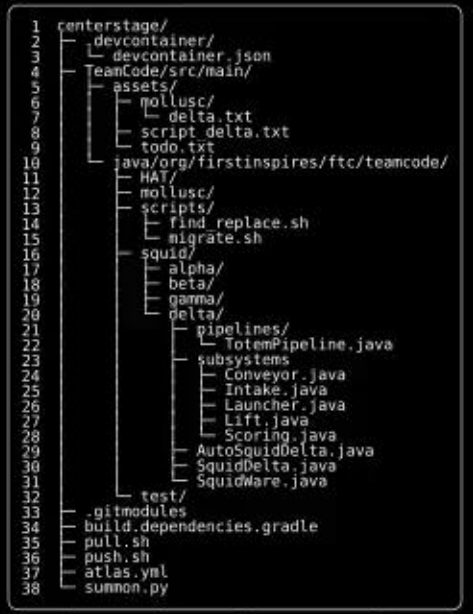
source code is available on the team GitHub organization and the programming web

Main Codebase Organization



Further descriptions are available in the `centerstage` repository README and release sections.

Main Codebase Organization



Important: The main branch on the repository contains a version of the codebase integrated with a untested version of mollusc. See the README section for more details.

Organization — mollusc

```
centerstage/
├─ .devcontainer/
│  └─ devcontainer.json
├─ TeamCode/src/main/
│  ├─ assets/
│  │  ├─ mollusc/
│  │  │  └─ delta.txt
│  │  ├─ script_delta.txt
│  │  └─ todo.txt
│  └─ java/org/firstinspires/ftc/teamcode/
│     ├─ HAT/
│     ├─ mollusc/
│     ├─ scripts/
│     │  ├─ find_replace.sh
│     │  └─ migrate.sh
│     ├─ squid/
│     │  ├─ alpha/
│     │  ├─ beta/
│     │  ├─ gamma/
│     │  └─ delta/
│     │     ├─ pipelines/
│     │     │  └─ TotemPipeline.java
│     │     ├─ subsystems
│     │     │  ├─ Conveyor.java
│     │     │  ├─ Intake.java
│     │     │  ├─ Launcher.java
│     │     │  ├─ Lift.java
│     │     │  └─ Scoring.java
│     │     ├─ AutoSquidDelta.java
│     │     ├─ SquidDelta.java
│     │     └─ SquidWare.java
│     └─ test/
├─ .gitmodules
```

mollusc Library
Contains functionality that can be reused across OpModes.

Organization — Directories

```
3   │   └─ devcontainer.json
4   ├─ TeamCode/src/main/
5   │   ├─ assets/
6   │   │   ├─ mollusc/
7   │   │   │   └─ delta.txt
8   │   │   ├─ script_delta.txt
9   │   │   └─ todo.txt
10  │   └─ java/org/firstinspires/ftc/teamcode/
11  │       ├─ HAT/
12  │       ├─ mollusc/
13  │       ├─ scripts/
14  │       │   ├─ find_replace.sh
15  │       │   └─ migrate.sh
16  │       ├─ squid/
17  │       │   ├─ alpha/
18  │       │   ├─ beta/
19  │       │   ├─ gamma/
20  │       │   └─ delta/
21  │       │       ├─ pipelines/
22  │       │       │   └─ TotemPipeline.java
23  │       │       ├─ subsystems
24  │       │       │   ├─ Conveyor.java
25  │       │       │   ├─ Intake.java
26  │       │       │   ├─ Launcher.java
27  │       │       │   ├─ Lift.java
28  │       │       │   └─ Scoring.java
29  │       │       ├─ AutoSquidDelta.java
30  │       │       ├─ SquidDelta.java
31  │       │       └─ SquidWare.java
32  │       └─ test/
33  ├─ .gitmodules
34  ├─ build.dependencies.gradle
35  ├─ pull.sh
36  ├─ push.sh
37  ├─ atlas.yml
38  └─ summon.py
```

`alpha`, `beta`, and `gamma` contain old codebases. They also contain older versions of mollusc. Ignore these.

Organization — Directories

```
│   │   ├── mollusc/
│   │   │   └── delta.txt
│   │   ├── script_delta.txt
│   │   └── todo.txt
│   └── java/org/firstinspires/ftc/teamcode/
│       ├── HAT/
│       ├── mollusc/
│       ├── scripts/
│       │   ├── find_replace.sh
│       │   └── migrate.sh
│       ├── squid/
│       │   ├── alpha/
│       │   ├── beta/
│       │   ├── gamma/
│       │   └── delta/
│       │       ├── pipelines/
│       │       │   └── TotemPipeline.java
│       │       ├── subsystems
│       │       │   ├── Conveyor.java
│       │       │   ├── Intake.java
│       │       │   ├── Launcher.java
│       │       │   ├── Lift.java
│       │       │   └── Scoring.java
│       │       ├── AutoSquidDelta.java
│       │       ├── SquidDelta.java
│       │       └── SquidWare.java
│       └── test/
├── .gitmodules
├── build.dependencies.gradle
├── pull.sh
├── push.sh
├── atlas.yml
└── summon.py
```

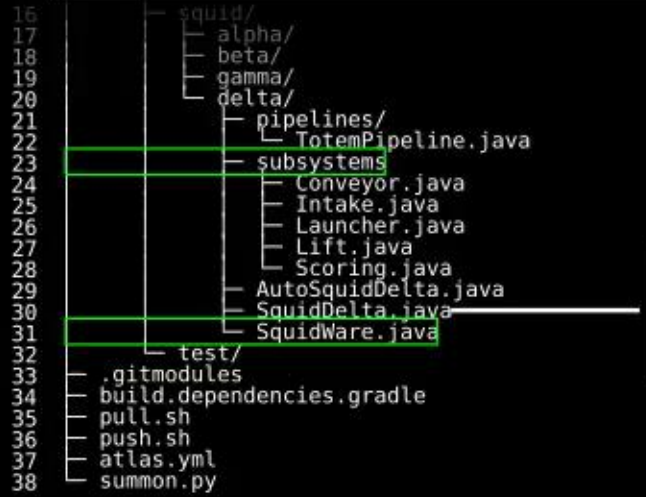
`delta` contains the most up-to-date source code.
Always reference this directory instead of the older ones.

Organization — Autonomous

```
15 │       │   └── migrate.sh
16 │       ├── squid/
17 │       │   ├── alpha/
18 │       │   ├── beta/
19 │       │   ├── gamma/
20 │       │   └── delta/
21 │       │       ├── pipelines/
22 │       │       │   └── TotemPipeline.java
23 │       │       ├── subsystems
24 │       │       │   ├── Conveyor.java
25 │       │       │   ├── Intake.java
26 │       │       │   ├── Launcher.java
27 │       │       │   ├── Lift.java
28 │       │       │   └── Scoring.java
29 │       │       ├── AutoSquidDelta.java
30 │       │       ├── SquidDelta.java
31 │       │       └── SquidWare.java
32 │       └── test/
33 ├── .gitmodules
34 ├── build.dependencies.gradle
35 ├── pull.sh
36 ├── push.sh
37 ├── atlas.yml
38 └── summon.py
```

Autonomous
- Initialization and drive functionality through mollusc.
- Hardware set up through `SquidWare`.
- Uses `TotemPipeline` as the image processing pipeline for detecting the team prop.
- May use various subsystems to score.

Organization — TeleOp



TeleOp
- Initialization and drive functionality also through mollusc.
- Hardware set up shared with Autonomous in `SquidWare`.
- Uses subsystems.

Organization — Additional Assets

```
centerstage/
├─ .devcontainer/
│  └─ devcontainer.json
├─ TeamCode/src/main/
│  ├─ assets/
│  │  ├─ mollusc/
│  │  │  └─ delta.txt
│  │  ├─ script_delta.txt
│  │  └─ todo.txt
│  └─ java/org/firstinspires/ftc/teamcode/
│     ├─ HAT/
│     ├─ mollusc/
│     ├─ scripts/
│     │  ├─ find_replace.sh
│     │  └─ migrate.sh
│     ├─ squid/
│     │  ├─ alpha/
│     │  ├─ beta/
│     │  ├─ gamma/
│     │  └─ delta/
│     │     ├─ pipelines/
│     │     │  └─ TotemPipeline.java
│     │     ├─ subsystems
│     │     │  ├─ Conveyor.java
│     │     │  ├─ Intake.java
│     │     │  ├─ Launcher.java
```

Additional Assets
- `SquidWare` (hardware initialization) uses mollusc to parse constants and other variables from `delta.txt` (i.e. PID coefficients).
- `script_delta.txt` contains the full autonomous script read by `AutoSquidDelta`.

Organization — Miscellaneous

```
centerstage/
├─ .devcontainer/
│  └─ devcontainer.json
├─ TeamCode/src/main/
│  ├─ assets/
│  │  ├─ mollusc/
│  │  │  └─ delta.txt
│  │  ├─ script_delta.txt
│  │  └─ todo.txt
│  └─ java/org/firstinspires/ftc/teamcode/
│     ├─ HAT/
│     ├─ mollusc/
│     ├─ scripts/
│     │  ├─ find_replace.sh
│     │  └─ migrate.sh
│     ├─ squid/
│     │  ├─ alpha/
│     │  ├─ beta/
│     │  ├─ gamma/
│     │  └─ delta/
│     │     ├─ pipelines/
│     │     │  └─ TotemPipeline.java
│     │     ├─ subsystems
│     │     │  ├─ Conveyor.java
│     │     │  ├─ Intake.java
│     │     │  ├─ Launcher.java
│     │     │  ├─ Lift.java
│     │     │  └─ Scoring.java
│     │     ├─ AutoSquidDelta.java
│     │     ├─ SquidDelta.java
│     │     └─ SquidWare.java
│     └─ test/
├─ .gitmodules
├─ build.dependencies.gradle
```

These scripts were used to quickly archive directories. Ignore these.

Organization — Miscellaneous

```
22  │   │   │   └─ TotemPipeline.java
23  │   │   ├─ subsystems
24  │   │   │   ├─ Conveyor.java
25  │   │   │   ├─ Intake.java
26  │   │   │   ├─ Launcher.java
27  │   │   │   ├─ Lift.java
28  │   │   │   └─ Scoring.java
29  │   │   ├─ AutoSquidDelta.java
30  │   │   ├─ SquidDelta.java
31  │   │   └─ SquidWare.java
32  │   └─ test/
33  ├─ .gitmodules
34  ├─ build.dependencies.gradle
35  ├─ pull.sh
36  ├─ push.sh
37  ├─ atlas.yml
38  └─ summon.py
```

These scripts can be used to help make changes remotely.
- `pull.sh` and `push.sh` are shortcuts to pull / push to GitHub (e.g. upload / download).
- `atlas.yml` and `summon.py` are used to establish a remote session whereby a programmer can edit code on the ThinkPad remotely.

Organization — Miscellaneous

```
22  │   │   │   └── TotemPipeline.java
23  │   │   ├── subsystems
24  │   │   │   ├── Conveyor.java
25  │   │   │   ├── Intake.java
26  │   │   │   ├── Launcher.java
27  │   │   │   ├── Lift.java
28  │   │   │   └── Scoring.java
29  │   │   ├── AutoSquidDelta.java
30  │   │   ├── SquidDelta.java
31  │   │   └── SquidWare.java
32  │   └── test/
33  ├── .gitmodules
34  ├── build.dependencies.gradle
35  ├── pull.sh
36  ├── push.sh
37  ├── atlas.yml
38  └── summon.py
```

Versions

There are three "versions" of mollusc:
- **v0.1.0** is the "stable" version that should be used.
- The **main branch** on the mollusc repository contains a refactored version of **v0.1.0** with an improved interface and additional features. However, it has not been tested and should not be used unless it is tested in the future.
- **6e446fa** is the commit hash / ID of the revision of mollusc used at state. It's practically identical to **v0.1.0** and can be treated as such.

Currently, mollusc can be added to any project by downloading or cloning it into the `teamcode` folder.
A more convenient method may be developed in the future.

mollusc

Tested Features

- An "intepreter" that reads, parses, and executes a file containing a primitive yet customizable command-like language to control autonomous movements.

Three dead wheel system odometry calculations.

Note:

From an initial starting orientation, forward movement corresponds to a positive X translation and rightward movement corresponds to a positive Y translation.

- Mecanum drive controlled by dead wheels for autonomous.
- Field centric and robot centric TeleOp drivetrains.
- External asset loading.
- Program configuration via the Driver Station.
- Gamepad controls utilities.
- PIDF controller.
- Contour-based multiple object detector for CV pipelines.

mollusc

Untested Features

- Mecanum drive controlled by wheel encoders for autonomous.
- Some driver station configuration functions.
- Low-pass and Kalman filters.
- Threaded function execution wrapper.
- Voltage compensator.
- AprilTag detection EOCV pipeline
  (newer versions of the robot controller SDK have this built in).

mollusc — Organization

mollusc — Organization

```
26  │   ├── Controls.java
27  │   └── PID.java
28  ├── vision/
29  │   ├── ColorRange.java
30  │   ├── ObjectDetector.java
31  │   └── VisionObject.java
32  ├── wrapper/
33  │   ├── Encoder.java
34  │   ├── Make.java
35  │   ├── MolluscLinearOpMode.java
36  │   └── MolluscOpMode.java
37  └── Mollusc.java
```

Provides a convenient way to access the main OpMode from multiple classes.

mollusc — Organization

```
21  │   └── ScriptParseException.java
22  ├── test/
23  ├── utility/
24  │   ├── Asset.java
25  │   ├── Configuration.java
26  │   ├── Controls.java
27  │   └── PID.java
28  ├── vision/
29  │   ├── ColorRange.java
30  │   ├── ObjectDetector.java
31  │   └── VisionObject.java
32  ├── wrapper/
33  │   ├── Encoder.java
34  │   ├── Make.java
35  │   ├── MolluscLinearOpMode.java
36  │   └── MolluscOpMode.java
37  └── Mollusc.java
```

Wrappers that provide additional functionality to existing SDK components.
- `Encoder` : Reads the encoder counts and calculates the number of revolutions / distance traveled from a single encoder port.
- `Make` : Provides shortcut functions to initialize hardware (i.e, initialize a motor and set its direction).
- `MolluscLinearOpMode` : Wraps `LinearOpMode` so that `Mollusc` works.
- `MolluscOpMode` : Wraps `OpMode` for the same reason.

mollusc — Organization

```
12  ├── drivetrain/
13  │   ├── DrivetrainBaseFourWheel.java
14  │   ├── Drivetrain.java
15  │   ├── MecanumFieldCentric.java
16  │   └── MecanumRobotCentric.java
17  ├── exception/
18  │   ├── AssetRetrievalException.java
19  │   ├── ConfigValueMissingException.java
20  │   ├── ParityException.java
21  │   └── ScriptParseException.java
22  ├── test/
23  ├── utility/
24  │   ├── Asset.java
25  │   ├── Configuration.java
26  │   ├── Controls.java
27  │   └── PID.java
28  ├── vision/
29  │   ├── ColorRange.java
30  │   ├── ObjectDetector.java
31  │   └── VisionObject.java
32  ├── wrapper/
33  │   ├── Encoder.java
34  │   ├── Make.java
35  │   ├── MolluscLinearOpMode.java
36  │   └── MolluscOpMode.java
37  └── Mollusc.java
```

Configuration utilities, controls, and PIDF controller.
- `Asset` : Reads a text-based asset file separate from the source code.
- `Configuration` : Parses the asset and provides methods for obtaining constants and other values specific in the asset.
- `Controls` : Enhances gamepad controls (i.e. only perform an action when a button is pressed but not held).
- `PID` : PIDF controller. Also contains a function that enables bulk reading (faster cycle times).

mollusc — Organization

mollusc — Organization

```
mollusc/
├── auto/
│   ├── odometry/
│   │   ├── DeadWheels.java
│   │   └── Pose.java
│   ├── Action.java
│   ├── Auto.java
│   ├── Instruction.java
│   ├── Interpreter.java
│   ├── MecanumAutoII.java
│   └── MecanumAutoI.java
├── drivetrain/
│   ├── DrivetrainBaseFourWheel.java
│   ├── Drivetrain.java
│   ├── MecanumFieldCentric.java
│   └── MecanumRobotCentric.java
├── exception/
│   ├── AssetRetrievalException.java
│   ├── ConfigValueMissingException.java
│   ├── ParityException.java
│   └── ScriptParseException.java
├── test/
├── utility/
│   ├── Asset.java
│   ├── Configuration.java
│   ├── Controls.java
│   └── PID.java
```

Use `MecanumAutoII` instead of `MecanumAutoI`.
II uses three deadwheel odometry, whereas I attempts to do localization based on wheel encoders and has not been tested.

mollusc — Organization

```
mollusc/
├── auto/
│   ├── odometry/
│   │   ├── DeadWheels.java
│   │   └── Pose.java
│   ├── Action.java
│   ├── Auto.java
│   ├── Instruction.java
│   ├── Interpreter.java
│   ├── MecanumAutoII.java
│   └── MecanumAutoI.java
├── drivetrain/
│   ├── DrivetrainBaseFourWheel.java
│   ├── Drivetrain.java
│   ├── MecanumFieldCentric.java
│   └── MecanumRobotCentric.java
├── exception/
│   ├── AssetRetrievalException.java
│   ├── ConfigValueMissingException.java
│   ├── ParityException.java
│   └── ScriptParseException.java
├── test/
├── utility/
│   ├── Asset.java
│   ├── Configuration.java
│   ├── Controls.java
│   └── PID.java
├── vision/
│   ├── ColorRange.java
```

Field-centric and robot-centric base classes.

mollusc — Organization

```
17  ├── exception/
18  │   ├── AssetRetrievalException.java
19  │   ├── ConfigValueMissingException.java
20  │   ├── ParityException.java
21  │   └── ScriptParseException.java
22  ├── test/
23  ├── utility/
24  │   ├── Asset.java
25  │   ├── Configuration.java
26  │   ├── Controls.java
27  │   └── PID.java
28  ├── vision/
29  │   ├── ColorRange.java
30  │   ├── ObjectDetector.java
31  │   └── VisionObject.java
32  ├── wrapper/
33  │   ├── Encoder.java
34  │   ├── Make.java
35  │   ├── MolluscLinearOpMode.java
36  │   └── MolluscOpMode.java
37  └── Mollusc.java
```

Computer vision calculations.
Use `ObjectDetector` to detect `VisionObject`s given a set of `ColorRange`s.

mollusc — Organization

```
 6 │   ├── Action.java
 7 │   ├── Auto.java
 8 │   ├── Instruction.java
 9 │   ├── Interpreter.java
10 │   ├── MecanumAutoII.java
11 │   └── MecanumAutoI.java
12 ├── drivetrain/
13 │   ├── DrivetrainBaseFourWheel.java
14 │   ├── Drivetrain.java
15 │   ├── MecanumFieldCentric.java
16 │   └── MecanumRobotCentric.java
17 ├── exception/
18 │   ├── AssetRetrievalException.java
19 │   ├── ConfigValueMissingException.java
20 │   ├── ParityException.java
21 │   └── ScriptParseException.java
22 ├── test/
23 ├── utility/
24 │   ├── Asset.java
25 │   ├── Configuration.java
26 │   ├── Controls.java
27 │   └── PID.java
28 ├── vision/
29 │   ├── ColorRange.java
30 │   ├── ObjectDetector.java
31 │   └── VisionObject.java
32 ├── wrapper/
33 │   ├── Encoder.java
34 │   ├── Make.java
```

Various methods (i.e. those in `Configuration`)
may thow exceptions when a user-error is detected.
Example: Requesting a configuration value that doesn't exist.
Exceptions will normally terminate the OpMode
unless caught and handled.

- `AssetRetrievalException`: The asset couldn't be loaded
  (possibly misnamed or in the wrong place).
- `ConfigValueMissingException`: The configuration value doesn't exist.
- `ParityException`: A feature was used that works in `LinearOpMode`s
  but not in `OpMode`s.
- `ScriptParseException`: An error was encountered
  while parsing the autonomous script.

mollusc — Organization

```
11  │   └── MecanumAutoI.java
12  ├── drivetrain/
13  │   ├── DrivetrainBaseFourWheel.java
14  │   ├── Drivetrain.java
15  │   ├── MecanumFieldCentric.java
16  │   └── MecanumRobotCentric.java
17  ├── exception/
18  │   ├── AssetRetrievalException.java
19  │   ├── ConfigValueMissingException.java
20  │   ├── ParityException.java
21  │   └── ScriptParseException.java
22  ├── test/
23  ├── utility/
24  │   ├── Asset.java
25  │   ├── Configuration.java
26  │   ├── Controls.java
27  │   └── PID.java
28  ├── vision/
29  │   ├── ColorRange.java
30  │   ├── ObjectDetector.java
31  │   └── VisionObject.java
32  ├── wrapper/
33  │   ├── Encoder.java
34  │   ├── Make.java
35  │   ├── MolluscLinearOpMode.java
36  │   └── MolluscOpMode.java
37  └── Mollusc.java
```

These are tests that were intended to verify the library's functionality.
However, the tests themselves were never tested or used, so they can be ignored.

mollusc — Organization

```
11  │   └── MecanumAutoI.java
12  ├── drivetrain/
13  │   ├── DrivetrainBaseFourWheel.java
14  │   ├── Drivetrain.java
15  │   ├── MecanumFieldCentric.java
16  │   └── MecanumRobotCentric.java
17  ├── exception/
18  │   ├── AssetRetrievalException.java
19  │   ├── ConfigValueMissingException.java
20  │   ├── ParityException.java
21  │   └── ScriptParseException.java
22  ├── test/
23  ├── utility/
24  │   ├── Asset.java
25  │   ├── Configuration.java
26  │   ├── Controls.java
27  │   └── PID.java
28  ├── vision/
29  │   ├── ColorRange.java
30  │   ├── ObjectDetector.java
31  │   └── VisionObject.java
32  ├── wrapper/
33  │   ├── Encoder.java
34  │   ├── Make.java
35  │   ├── MolluscLinearOpMode.java
36  │   └── MolluscOpMode.java
37  └── Mollusc.java
```

Hardware Initialization

```
// import com.qualcomm.robotcore.hardware.TouchSensor;
import com.qualcomm.robotcore.hardware.DistanceSensor;
import com.qualcomm.robotcore.hardware.DcMotorEx;
import com.qualcomm.robotcore.hardware.CRServo;
import com.qualcomm.robotcore.hardware.Servo;
import com.qualcomm.robotcore.hardware.IMU;

import com.qualcomm.hardware.rev.RevHubOrientationOnRobot;

public class SquidWare {

    public Configuration config;

    public DrivetrainBaseFourWheel base;

    // Deadwheels.
    public DeadWheels deadWheels;

    public Intake intake = new Intake();
    public Conveyor conveyor = new Conveyor();
    public Scoring scoring = new Scoring();
    public Launcher launcher = new Launcher();
    public Lift lift = new Lift();

    public IMU imu;

    public SquidWare(Configuration config, boolean resetIMU) throws Exception {
        this.config = config;

        intake.UP_1 = intake.intakePos1 = config.getDouble("intake servo up 1");
        intake.UP_2 = intake.intakePos2 = config.getDouble("intake servo up 2");
        intake.DOWN_1 = config.getDouble("intake servo down 1");
        intake.DOWN_2 = config.getDouble("intake servo down 2");
        intake.MAX_DOWN_1 = config.getDouble("intake servo 1 max down");
        intake.MAX_DOWN_2 = config.getDouble("intake servo 2 max down");
        intake.DELTA = config.getDouble("intake servo delta");
        intake.PRESET_POWER = config.getDouble("intake power");

        conveyor.PRESET_POWER = config.getDouble("conveyor power");
        conveyor.DISTANCE_THRESHOLD = config.getDouble("pixel distance threshold cm");
        conveyor.PIXEL_TIME = config.getDouble("pixel time seconds");
        conveyor.CARRY_TIME = config.getDouble("pixel carry time seconds");

        scoring.SPINNER_POWER = config.getDouble("spinning servo powers");
```

All hardware initialization is done in `SquidWare` and shared bet