Table Of Contents

ActionMaster 2
ArmAction 4
CameraAction 6
Chassis 8
Collector 12
CraterFace 15
D_LeftEnter 16
D_OtherCrater 17
DriveAction 18
E404_Autonomous 20
Error404MecanumTeleop 22
Error404TankTeleop 24
FieldVision 26
Gen2_Hang 28
Gen2_Hang.Lift 31
GyroAction 33
HangAction 35
MarkDeploy 36
MarkDeployAction 38
MecanumChassis 40
MotorArm 45
MotorArm.ArmPositions 50
ResetHeadingAction 52
RobotAction 54
RuckusBot 56
TurnAction 65
WaitAction 67

Class ActionMaster

```
java.lang.Object
    |
    +--org.firstinspires.ftc.teamcode.ActionMaster
```

< Constructors > < Methods >

---

public class **ActionMaster**
extends java.lang.Object

ActionMaster is responsible for managing the run list of actions and moving each action to the run map and then removing it in sequence.

Author:

Andrew, Error 404: Team Name Not Found

Constructors

ActionMaster

`public` **`ActionMaster`**`()`

Methods

addAction

`public void` **`addAction`**`(RobotAction action)`

Adds the action listed in the parameter to the action map.

---

addRunAction

`public void` **`addRunAction`**`(java.lang.String action)`

Adds an action to the run map.

---

buildActionMap

`public void **buildActionMap**()`

Clears the action map, run map, and next list of actions.

---

execute

`public void **execute**()`

The body of the code to execute: Creates a run list of actions to do, executes the current action, and then deletes each action from the run list as it is completed.

---

getRunListSize

`public int **getRunListSize**()`

Returns the number of objects in the run map.

Returns:

the number of objects in the run map.

---

init

`public void **init**(Telemetry telem)`

Initializes telemetry.

---

keyList

`public java.util.Set **keyList**()`

Lists which actions currently running.

Class ArmAction

```
java.lang.Object
    |
    +--RobotAction
        |
        +--org.firstinspires.ftc.teamcode.ArmAction
```

< Methods >

---

public class **ArmAction**
extends RobotAction

Loaded into the run map as an action that moves the arm. Each action is parameterized by the CSV file.

Author:

Andrew, Error 404 Robotics

RobotAction

Methods

entry

`public void entry()`

Placeholder for entry. Currently only calls the parent entry method.

Overrides:

entry in class RobotAction

---

execute

`public boolean execute()`

The body of the action to be executed: Calls the goTo() method in MotorArm.

Overrides:

execute in class RobotAction

---

exit

```
public void exit()
```

Stops the motors and calls the parent exit method.

Overrides:

exit in class RobotAction

---

init

```
public void init(Telemetry telem,
                 RuckusBot theRobot)
```

Placeholder for entry. Currently only calls the parent entry method.

Overrides:

init in class RobotAction

Class CameraAction

```
java.lang.Object
    |
    +--RobotAction
        |
        +--org.firstinspires.ftc.teamcode.CameraAction
```

< Methods >

---

public class **CameraAction**
extends RobotAction

Loaded into the run map as an action that uses the camera. Each action is parameterized by the CSV file.

Author:

Andrew, Error 404: Team Name Not Found

RobotAction

Methods

entry

`public void` **`entry`**`()`

Sets the count variable to zero and calls the parent init method.

Overrides:

entry in class RobotAction

---

execute

`public boolean` **`execute`**`()`

The body of the action to be executed: Based on the location of the gold mineral returned by the goldPosition() method, sets the next action

Overrides:

execute in class RobotAction

---

exit

```
public void exit()
```

Shuts down the camera and calls the parent exit method.

Overrides:

exit in class RobotAction

---

init

```
public void init(Telemetry telem,
                 RuckusBot theRobot)
```

Placeholder for initialization. Currently only calls the parent init method.

Overrides:

init in class RobotAction

Class Chassis

```
java.lang.Object
    |
    +--org.firstinspires.ftc.teamcode.Chassis
```

Direct Known Subclasses:

MecanumChassis

< Constructors > < Methods >

---

public class **Chassis**
extends java.lang.Object

The basic chassis class that has all of the general objects in it used by all chassis types. Currently has two child classes or chassis types: TankChassis and MecanumChassis.

Author:

Andrew, Error 404: Team Name Not Found

Constructors

Chassis

```
public  Chassis()
```

Methods

drive

```
public boolean drive(double power,
                     double direction,
                     double gain,
                     double distance,
                     double time)
```

Drive the robot with like a mecanum robot. Uses the gyro to preserve the orientation the robot was at at the beginning of the move.

Returns:

a boolean that tells whether or not the robot is currently moving.

---

getRuntime

```
public double getRuntime()
```

Get the number of seconds this op mode has been running

This method has sub millisecond accuracy.

Returns:

number of seconds this op mode has been running

---

goodPitch

```
public boolean goodPitch()
```

Measures the robot's pitch and determines whether or not the robot is stuck on the lander.

Returns:

A booleans that is whether or not the robot is stuck.

---

inchesToTicks

```
public double inchesToTicks(double distanceInch)
```

Converts inches to encoder ticks.

Parameters:

distanceInch - A double that is the number of inches to convert to encoder ticks

Returns:

A double that is the converted number of ticks.

---

init

```
public void init(HardwareMap hwMap,
                 Telemetry telem)
```

Initializes the hardware and objects needed for this class.

---

joystickDrive

```
public void joystickDrive(double leftStickX,
                          double leftStickY,
                          double rightStickX,
                          double rightStickY,
                          double powerLimit)
```

Uses joystick inputs the drive the robot. Allows for omni-directional movement and has a selectable max power.

Parameters:

leftStickX - The x-axis of the left joystick on the primary gamepad. Controls side to side movement.
leftStickY - The y-axis of the left joystick on the primary gamepad. Controls forward and backward movement.
rightStickX - The x-axis of the right joystick on the primary gamepad. Controls rotation.
rightStickY - The y-axis of the right joystick on the primary gamepad. N/A
powerLimit - The maximum power value.

---

pointTurn

```
public boolean pointTurn(double power,
                         double targetHeading,
                         double time)
```

Turns the robot to a target heading. Algorithm finds the shortest direction to take to the target heading.

Returns:

a boolean that tells whether or not the robot is currently moving.

---

reset

```
public boolean reset(double power,
                     double time)
```

Takes the starting heading of the robot before landing and uses that start heading to home the robot angle after landing.

Parameters:

power - The power the robot will move at.
time - The maximum time this method can run before timing out -- prevents the robot from stalling during autonmous.

Returns:

returns a boolean that stands for whether or not the robot is moving.

---

resetStartTime

```
public void resetStartTime()
```

Reset the internal timer to zero.

---

stopMotors

```
public void stopMotors()
```

Stop drive motors

---

tankDrive

```
public boolean tankDrive(double power,
                         org.firstinspires.ftc.teamcode.Chassis.TankDirection
direction,
                         double gain,
                         double distance,
                         double time)
```

Drives the robot like a tank. Uses the gyro to drive at an absolute zero (captured at the beginning of the program.

Has to possible directions: forward and backward.

Returns:

a boolean that tells whether or not the robot is currently moving.

Class Collector

```
java.lang.Object
    |
    +--org.firstinspires.ftc.teamcode.Collector
```

< Fields > < Constructors > < Methods >

---

public class **Collector**
extends java.lang.Object

Contains the hardware and methods for the mineral collector mechanism.

Author:

Ben, Error 404: Team Name Not Found

Fields

theServoL

`public CRServo` **`theServoL`**

A continuous rotation smart servo called theServoL.

---

theServoR

`public CRServo` **`theServoR`**

A continuous rotation smart servo called theServoR.

Constructors

Collector

`public` **`Collector`**`()`

Methods

eject

```
public void eject()
```

Turns the intake wheels outward to spit our minerals.

---

ejectL

```
public void ejectL()
```

Turns the left intake wheel outward to spit out minerals.

---

ejectR

```
public void ejectR()
```

Turns the right intake wheel outward to spit our minerals.

---

init

```
public void init(HardwareMap hwmap,
                 Telemetry telem)
```

Initializes the hardware used by the class. The try-catch statement prevents the code from crashing if it can't find a hardware object. Instead it posts a message to the phone informing the driver of the missing object.

Parameters:

hwmap - An instance of HardwareMap that allows hardware objects to be initialized.
telem - An instance of Telemtry that allows this class to use telemetry statements.

---

intake

```
public void intake()
```

Turns the intake wheels inward to suck in minerals.

---

intakeL

`public void intakeL()`

Turns the left intake wheel inward to suck in minerals.

---

intakeR

`public void intakeR()`

Turns the right intake wheel inward to suck in minerals.

---

stop

`public void stop()`

Stops the intake wheels.

---

stopL

`public void stopL()`

Stops the left intake wheel.

---

stopR

`public void stopR()`

Stops the right intake wheel.

Class CraterFace

```
java.lang.Object
    |
    +--OpMode
        |
        +--E404_Autonomous
            |
            +--org.firstinspires.ftc.teamcode.CraterFace
```

< Constructors > < Methods >

public class **CraterFace**
extends E404_Autonomous

Constructors

CraterFace

```
public  CraterFace()
```

Methods

init

```
public void init()
```

Calls the init methods for needed classes and locates the correct file path to the CSV file for the crater face drive path.

Overrides:

init in class E404_Autonomous

Class D_LeftEnter

```
java.lang.Object
    |
    +--OpMode
        |
        +--E404_Autonomous
            |
            +--org.firstinspires.ftc.teamcode.D_LeftEnter
```

< Constructors > < Methods >

public class **D_LeftEnter**
extends E404_Autonomous

Constructors

D_LeftEnter

```
public  D_LeftEnter()
```

Methods

init

```
public void init()
```

Calls the init methods for needed classes and locates the pathway for the CSV file for the depo left enter drive path.

Overrides:

init in class E404_Autonomous

Class D_OtherCrater

```
java.lang.Object
    |
    +--OpMode
        |
        +--E404_Autonomous
            |
            +--org.firstinspires.ftc.teamcode.D_OtherCrater
```

< Constructors > < Methods >

public class **D_OtherCrater**
extends E404_Autonomous

Constructors

D_OtherCrater

```
public  D_OtherCrater()
```

Methods

init

```
public void init()
```

Calls the init methods for needed classes and locates the pathway to the CSV file for the depo face other crater drive path.

Overrides:

init in class E404_Autonomous

Class DriveAction

```
java.lang.Object
    |
    +--RobotAction
        |
        +--org.firstinspires.ftc.teamcode.DriveAction
```

< Methods >

---

public class **DriveAction**
extends RobotAction

Loaded into the run map as an action that drives the robot. Each action is parameterized by the CSV file.

Author:

Andrew, Error 404: Team Name Not Found

RobotAction

Methods

entry

`public void entry()`

Placeholder for entry. Currently only calls the parent entry method.

Overrides:

entry in class RobotAction

---

execute

`public boolean execute()`

The body of the action to be executed: Calls the drive() method in MecanumChassis.

Overrides:

execute in class RobotAction

---

exit

```
public void exit()
```

Stops all the motors on the robot and calls the parent exit method.

Overrides:

exit in class RobotAction

---

init

```
public void init(Telemetry telem,
                 RuckusBot theRobot)
```

Placeholder for initialization. Currently only calls the parent init method.

Overrides:

init in class RobotAction

Class E404_Autonomous

```
java.lang.Object
    |
    +--OpMode
        |
        +--org.firstinspires.ftc.teamcode.E404_Autonomous
```

Direct Known Subclasses:
CraterFace, D_LeftEnter, D_OtherCrater

< Constructors > < Methods >

---

public class **E404_Autonomous**
extends OpMode

Constructors

E404_Autonomous

`public` **`E404_Autonomous`**`()`

Methods

init

`public void` **`init`**`()`

Initializes the robot and telemetry and creates all the actions defined in the appropriate CSV file.

---

loop

`public void` **`loop`**`()`

Contains the actual movement commands of the class. Runs repeatedly until the stop button is pressed.

---

start

```
public void start()
```

Runs once when the start button is pressed, but before the loop method starts.

---

stop

```
public void stop()
```

Runs once after the driver hits the stop button on teh drivers station phone. Stops the loop method and stops the drive motors.

Class Error404MecanumTeleop

```
java.lang.Object
    |
    +--OpMode
        |
        +--org.firstinspires.ftc.teamcode.Error404MecanumTeleop
```

< Constructors > < Methods >

public class **Error404MecanumTeleop**
extends OpMode

Constructors

Error404MecanumTeleop

`public` **`Error404MecanumTeleop`**`()`

Methods

afterburners

`public double` **`afterburners`**`()`

afterburners() allows the driver to increase the robot's top speed from the default of 0.3 to 0.8 by holding down the left trigger on the gamepad. This is because it is easier to make small precise movements (like lining up on a mineral) at a lower top speed, but it is also useful to drive fast when crossing the field.

Returns:

a double that is the maximum power

init

`public void` **`init`**`()`

Calls the init methods for needed classes.

loop

```
public void loop()
```

Contains the actual movement commands of the class. Runs repeatedly after the driver hits play and until the stop button is pressed.

---

start

```
public void start()
```

Not used for anything right now, but runs once when the start button is pressed, but before the loop method starts.

---

stop

```
public void stop()
```

Runs once after the driver hits the stop button on teh drivers station phone. Stops the loop method and stops the drive motors.

Class Error404TankTeleop

```
java.lang.Object
    |
    +--OpMode
        |
        +--org.firstinspires.ftc.teamcode.Error404TankTeleop
```

< Constructors > < Methods >

---

public class **Error404TankTeleop**
extends OpMode

A teleop opmode that drive the robot like a tank (ie, not mecanum).

Author:

Andrew, Error 404: Team Name Not Found

OpMode

Constructors

Error404TankTeleop

`public` **`Error404TankTeleop`**`()`

Methods

init

`public void` **`init`**`()`

Calls the needed init methods for used classes.

---

loop

`public void` **`loop`**`()`

Contains the actual movement commands of the class. Runs repeatedly until the stop button is pressed.

start

```
public void start()
```

Not used for anything right now, but runs once when the start button is pressed, but before the loop method starts.

stop

```
public void stop()
```

Runs once after the driver hits the stop button on teh drivers station phone. Stops the loop method.

Class FieldVision

```
java.lang.Object
    |
    +--org.firstinspires.ftc.teamcode.FieldVision
```

< Constructors > < Methods >

---

public class **FieldVision**
extends java.lang.Object

Contains all the vision code for autonomous including the tensor flow and vuforia image navigation code.

Author:

Andrew, Error 404: Team Name Not Found

Constructors

FieldVision

```
public  FieldVision()
```

Methods

init

```
public void init(HardwareMap hwMap,
                 Telemetry telem)
```

Initializes all the hardware and assets used by vuforia and tensor flow.

Parameters:

hwMap - An instnce of the FIRST-provided HardwareMap which allows for hardware to be initilized to the robot.
telem - An instance of Telemetry which allows this class to use Telemetry.

---

start

```
public void start()
```

tensorFlowMineralDetection

`public java.lang.String` **`tensorFlowMineralDetection`**`()`

Sourced from the FIRST-provided tensor flow example code, this method recognizes minerals in its field of vision and identifies the position of the gold mineral (LEFT, RIGHT, CENTER) by using the positions of the silver minerals and returns that value as a string. Modified by Error 404: Looks for the gold mineral (ie, no longer needs to see silver minerals to calculate the gold mineral position). If it sees a gold mineral that matches the specified height perameter, it then calculates the gold position using the x-axis value of the gold mineral.

Returns:

The gold location on the field.

---

tfodShutdown

`public void` **`tfodShutdown`**`()`

Shuts down the tensor flow algorithm. Turning it off when it's not needed preserves the robot controller's battery

Class Gen2_Hang

```
java.lang.Object
    |
    +--org.firstinspires.ftc.teamcode.Gen2_Hang
```

< Constructors > < Methods >

---

public class **Gen2_Hang**
extends java.lang.Object

Contains the code for initializing the hanger hardware and for running the hanger in teleop and autonomous.

Author:

Ben, Error 404: Team Name Not Found

Constructors

Gen2_Hang

`public` **`Gen2_Hang`**`()`

Methods

getRuntime

`public double` **`getRuntime`**`()`

Sources the current time in seconds from the internal timer.

Returns:

the current time.

---

hangControl

```
public void hangControl(boolean down,
                        boolean up,
                        double power)
```

A generic drive method for controlling the hanger.

Parameters:

down - A truth value of whether or not up is the correct direction.
up - A truth value of whether or not down is the correct direction.
power - The power at which the hanger will run.

---

hangDrive

```
public boolean hangDrive(double power,
                         double distance,
org.firstinspires.ftc.teamcode.Gen2_Hang.HangDirection direction)
```

Drives the hanger for a set number of encoder ticks. Used in autonomous.

Parameters:

power - The power at which the hanger will run.
distance - The number of encoders the hanger will run.
direction - The direction the hanger will run in (there are two choices: IN or OUT).

Returns:

A boolean that is whether or not the robot is moving.

---

hangHome

```
public void hangHome()
```

Pulls the hanger down to the home position.

---

hangerDeploy

```
public void hangerDeploy()
```

Sends the hanger out to a preset position in preparation for hanging.

---

hangerHang

```
public void hangerHang()
```

Pulls the hanger in to a preset position to lift the robot off the ground.

---

init

```
public void init(HardwareMap hwmap,
                 Telemetry telem)
```

Initializes the hardware used in the hanger mechanism.

Parameters:

hwmap - An instance of the hardware map. Used to initialize the hang motor.
telem - An instance of Telemtry that allows for telemetry to be initialized and therefore used in this class.

---

resetStartTime

```
public void resetStartTime()
```

Reset the internal timer to zero.

---

start

```
public void start()
```

Called when the start button on the controller is pressed. Runs once.

---

stopHangMotor

```
public void stopHangMotor()
```

Stops the hang motor.

Class Gen2_Hang.Lift

```
java.lang.Object
    |
    +--java.lang.Enum
        |
        +--org.firstinspires.ftc.teamcode.Gen2_Hang.Lift
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

< Fields > < Methods >

---

public static final class **Gen2_Hang.Lift**
extends java.lang.Enum

Contains the allowed preset values for the hanger.

Fields

HANGDEPLOY

`public static final Gen2_Hang.Lift` **`HANGDEPLOY`**

HANGERHANG

`public static final Gen2_Hang.Lift` **`HANGERHANG`**

HANGERHOME

`public static final Gen2_Hang.Lift` **`HANGERHOME`**

---

targetEncoder

`public final double` **`targetEncoder`**

Methods

valueOf

public static Gen2_Hang.Lift **valueOf**(java.lang.String name)

---

values

public static org.firstinspires.ftc.teamcode.Gen2_Hang.Lift[] **values**()

Class GyroAction

```
java.lang.Object
    |
    +--RobotAction
        |
        +--org.firstinspires.ftc.teamcode.GyroAction
```

< Methods >

---

public class **GyroAction**
extends RobotAction

Loaded into the run map as an action that reads the gyro. Each action is parameterized by the CSV file.

Author:

Andrew, Error 404: Team Name Not Found

RobotAction

Methods

entry

`public void entry()`

Placeholder for entry. Currently only calls the parent entry method.

Overrides:

entry in class RobotAction

---

execute

`public boolean execute()`

The body of the action to be executed: Based on the boolean value returned by the goodPitch() method, determines if the robot is stuck and sets the next action accordingly.

Overrides:

execute in class RobotAction

---

exit

```
public void exit()
```

Placeholder for exit. Currently only calls the parent exit method.

Overrides:

exit in class RobotAction

---

init

```
public void init(Telemetry telem,
                 RuckusBot theRobot)
```

Placeholder for initialization. Currently only calls the parent init method.

Overrides:

init in class RobotAction

Class HangAction

```
java.lang.Object
    |
    +--RobotAction
        |
        +--org.firstinspires.ftc.teamcode.HangAction
```

< Methods >

---

public class **HangAction**
extends RobotAction

Author:

Andrew, Error 404: Team Name Not Found

RobotAction

Methods

execute

`public boolean execute()`

The body of the action to be executed: Calls the hangDrive() method in MotorArm.

Overrides:

execute in class RobotAction

---

exit

`public void exit()`

Stops all the motors and calls the parent exit method.

Overrides:

exit in class RobotAction

Class MarkDeploy

```
java.lang.Object
    |
    +--org.firstinspires.ftc.teamcode.MarkDeploy
```

< Fields > < Constructors > < Methods >

---

public class **MarkDeploy**
extends java.lang.Object

Contains the hardware and methods to run the mechanism that deploys the team marker during autonomous.

Author:

Andrew, Error 404: Team Name Not Found

Author:

Ben, Error 404: Team Name Not Found

Fields

flag

```
public CRServo flag
```

A continuous rotation servo called flag.

Constructors

MarkDeploy

```
public  MarkDeploy()
```

Methods

init

```
public void init(HardwareMap hwMap,
                 Telemetry telem)
```

Initilzing the hardware used in this class.

Parameters:

hwMap - An instance of HardwareMap that lets hardware be initialized.
telem - An instance of Telemetry that lets telemetry statements be used in this class.

---

markDeploy

```
public void markDeploy(double power)
```

Turns the mark deploy servo one and off.

Parameters:

power - the power that the servo will run at (1 or 0).

Class MarkDeployAction

```
java.lang.Object
    |
    +--RobotAction
        |
        +--org.firstinspires.ftc.teamcode.MarkDeployAction
```

< Methods >

---

public class **MarkDeployAction**
extends RobotAction

Loaded into the run map as an action that moves the mark deploy servo. Each action is parameterized by the CSV file.

Author:

Andrew, Error 404: Team Name Not Found

RobotAction

Methods

entry

`public void entry()`

Starts the mark deploy servo and calls the parent entry method.

Overrides:

entry in class RobotAction

---

execute

`public boolean execute()`

Placeholder for execute. Calls the parent execute method.

Overrides:

execute in class RobotAction

---

exit

```
public void exit()
```

Stops the mark deploy servo and calls the parent exit method.

Overrides:

exit in class RobotAction

---

init

```
public void init(Telemetry telem,
                 RuckusBot theRobot)
```

Placeholder for initialization. Currently only calls the parent init method.

Overrides:

init in class RobotAction

Class MecanumChassis

```
java.lang.Object
    |
    +--Chassis
        |
        +--org.firstinspires.ftc.teamcode.MecanumChassis
```

< Constructors > < Methods >

---

public class **MecanumChassis**
extends Chassis

A chassis type. Specifically a mecanum chassis. Contains all of the hardware declerations for a mecanum chassis as well as the methods used by a chassis (ie, a bunch of drive methods).

Author:

Andrew, Error 404: Team Name Not Found

Chassis

Constructors

MecanumChassis

```
public  MecanumChassis()
```

Methods

drive

```
public boolean drive(double power,
                     double direction,
                     double gain,
                     double distance,
                     double time)
```

The mecanumDrive method moves the four drive motors on the robot and will move the robot forward, backward, left, right, or at a 45 degree angle in any direction.

Parameters:

power - How fast the robot will drive.
gain - The rate at which the robot will correct an error
direction - In which direction the robot will drive (forward, backward, left, right, or 45 degrees in any direction).
distance - How far the robot will drive.
time - The max time this move can take. A time-out feature: if the move stalls for some reason, the timer will catch it.

Returns:

A boolean that tells us whether or not the robot is moving.

Overrides:

drive in class Chassis

---

getHeadingDbl

```
public double getHeadingDbl()
```

Used to get the robot's heading.

Returns:

the robot's heading as an double

---

getPitchDbl

```
public double getPitchDbl()
```

Used to get the robot's pitch.

Returns:

the robot's pitch as a double.

---

getRollDbl

```
public double getRollDbl()
```

Used to get the robot's roll.

Returns:

the robot's roll as a double.

---

goodPitch

```
public boolean goodPitch()
```

Checks the imu to determine if the robot is flat and level. If it is stuck on the latch, the robot tips and turns. This method will tell you if you are OK by looking at the pitch.

Returns:

A boolean which tells you if the robot's pitch reading is good or not.

Overrides:

goodPitch in class Chassis

---

init

```
public void init(HardwareMap hwMap,
                 Telemetry telem)
```

Look for a specified set of motors in the config file. If the motors are found, give them a specified direction. If a motor is not found, ignore the error and set the motor to equal null.

Parameters:

hwMap - The hardware map that the code will use to find and classify the objects it sees.
telem - Initializes a telemetry object that allows for telemetry statements.

Overrides:

init in class Chassis

---

joystickDrive

```
public void joystickDrive(double leftStickX,
                          double leftStickY,
                          double rightStickX,
                          double rightStickY,
                          double powerLimit)
```

This method takes the command values from the x- and y-axes of the left and right joysticks on the gamepad and converts them to motor directional power commands for the drive motors. The algorithm also makes sure that the power values don't exceed a magnitude of a selected limit. Allows the robot to move in any direction while not exceeding a set speed. The code making sure the speed commands don't exceed a value of 1 was sourced from: (see chiefdelphi.com)

Parameters:

leftStickX - The x-axis of the left joystick
leftStickY - The y-axis of the left joystick
rightStickX - The x-axis of the right joystick
rightStickY - The y-axis of the right joystick
powerLimit - The max power allowed to the motors

Overrides:

joystickDrive in class Chassis

---

pointTurn

```
public boolean pointTurn(double power,
                         double targetHeading,
                         double time)
```

The pointTurn method turns the robot to a target heading, automatically picking the turn direction that is the shortest distance to turn to arrive at the target.

Parameters:

targetHeading - The direction in which the robot will move.
time - The maximum time the move can take before the code moves on.
power - The power at which the robot will move.

Returns:

A boolean that tells use whether or not the robot is moving.

Overrides:

pointTurn in class Chassis

---

reset

```
public boolean reset(double power,
                     double time)
```

Turns the robot to the initial heading it started out at.

Parameters:

power - The power at which the robot will drive.
time - The maximum amount of time the move can take before timing out.

Returns:

A boolean that is whether or not the robot is moving.

Overrides:

reset in class Chassis

---

stopMotors

```
public void stopMotors()
```

Stop all four drive motors by setting their power to zero.

Overrides:

stopMotors in class Chassis

---

tankDrive

```
public boolean tankDrive(double power,
                         org.firstinspires.ftc.teamcode.Chassis.TankDirection
direction,
                         double gain,
                         double distance,
                         double time)
```

A drive method that does not utilize the mecanum wheels on the robot and therefore only drives forward and backward

Parameters:

power - The power at which the robot will drive.
direction - The direction in which the robot will drive.
gain - The degree of which the robot will correct error.
distance - The target number of inches the robot will move.
time - The maximum amount of time the move can take before timing out.

Returns:

A boolean that says whether or not the robot is moving

Overrides:

tankDrive in class Chassis

Class MotorArm

```
java.lang.Object
    |
    +--org.firstinspires.ftc.teamcode.MotorArm
```

< Fields > < Constructors > < Methods >

---

public class **MotorArm**
extends java.lang.Object

The class responsible for the mineral collection and deployment arm on the robot. Contains all the harware and methods for the arm.

Author:

Ben, Error 404: Team Name Not Found

Fields

chassisTouch

```
protected TouchSensor chassisTouch
```

A touch sensor that, if pressed, stops the arm.

---

elbowBack

```
protected TouchSensor elbowBack
```

A touch sensor that, if pressed, stops the forearm from folding into itself.

---

elbowFront

```
protected TouchSensor elbowFront
```

A touch sensor that, if pressed, stops the forearm from folding into itself.

Constructors

MotorArm

```
public  MotorArm()
```

Methods

armDeploy

```
public boolean armDeploy(int shoulderTarget,
                         int elbowTarget,
                         boolean elbowSecond)
```

Drives the arm to a target position and returns true when done.

Parameters:

shoulderTarget - The target position for the shoulder motor.
elbowTarget - The target position for the elbow motor.
elbowSecond - A boolean that tells whether or not to move the elbow after the shoulder is done.

Returns:

A boolean that tells whether or not the arm is moving.

---

armDrive

```
public void armDrive(double RightStickY,
                     double LeftStickY)
```

Sends the joystick commands to the motors, allowing the drivers to move the arm. Reverses the arm's movement if it hits one of the three limit switches. This prevents the arm from driving into itself and breaking.

Parameters:

RightStickY - The y-axis of the right stick on the gamepad. Controls the elbow motor.
LeftStickY - The y-axis of the left stick on the gamepad. Controls the shoulder motor.

---

armHome

```
public boolean armHome()
```

Convenience method to drive the arm to the home position where it is fully stowed.

Returns:

A boolean that tells whether or not the arm is moving.

---

craterExtend

`public boolean craterExtend()`

Convenience method to drive the arm to the position where it is fully extended into the crater in preparation for collecting minerals.

Returns:

A boolean that tells whether or not the arm is moving.

---

drivingExtend

`public boolean drivingExtend()`

Convenience method to drive the arm to the position where it is partially extended to make it easier to drive around between the crater and the lander.

Returns:

A boolean that tells whether or not the arm is moving.

---

getRuntime

`public double getRuntime()`

Get the number of seconds this op mode has been running

This method has sub millisecond accuracy.

Returns:

number of seconds this op mode has been running

---

goTo

```
public boolean goTo(MotorArm.ArmPositions position,
                    double elbPower,
                    double shouldPower)
```

Moves the arm towards the named position.

Parameters:

position - An enumerated value that specifies where the arm should move towards.
elbPower - The speed with which the elbow should move.
shouldPower - The speed with which the shoulder should move.

Returns:

A boolean that tells whether or not the arm is moving.

goldCollect

```
public boolean goldCollect(boolean elbowSecond)
```

Convenience method to drive the arm to the position where the shoulder is perpendicular to the robot chassis and the elbow is slight forward for knocking off the gold mineral in autonomous.

Returns:

A boolean that tells whether or not the arm is moving.

---

init

```
public void init(HardwareMap hwmap,
                 Telemetry telem)
```

Initializes all the motors and sensors on the arm using try-catches. The try-catch statements prevents the code from crashing if the wanted device is missing and instead sends a message to the phone to notify the driver of the missing device.

Parameters:

hwmap - An instance of the FIRST-provided HarwareMap.
telem - An instance of Telemetry that allows the use of telemetry in this class.

---

landerExtend

```
public boolean landerExtend()
```

Convenience method to drive the arm to the position where it is up next to the lander in preparation for putting minerals into the lander cargro hold.

Returns:

A boolean that tells whether or not the arm is moving.

---

resetStartTime

```
public void resetStartTime()
```

Reset the internal timer to zero.

---

stop

```
public void stop()
```

Stop all arm movement.

Class MotorArm.ArmPositions

```
java.lang.Object
    |
    +--java.lang.Enum
        |
        +--org.firstinspires.ftc.teamcode.MotorArm.ArmPositions
```

All Implemented Interfaces:
java.io.Serializable, java.lang.Comparable

< Fields > < Methods >

---

public static final class **MotorArm.ArmPositions**
extends java.lang.Enum

An enum method that contains various arm preset coordinates.

Fields

ARM_HOME

`public static final MotorArm.ArmPositions` **`ARM_HOME`**

CRATER_EXTEND

`public static final MotorArm.ArmPositions` **`CRATER_EXTEND`**

DRIVING_EXTEND

`public static final MotorArm.ArmPositions` **`DRIVING_EXTEND`**

LANDER_EXTEND

`public static final MotorArm.ArmPositions` **`LANDER_EXTEND`**

MINERAL_COLLECT

public static final MotorArm.ArmPositions **MINERAL_COLLECT**

---

elbow

public final int **elbow**

---

shoulder

public final int **shoulder**

Methods

valueOf

public static MotorArm.ArmPositions **valueOf**(java.lang.String name)

---

values

public static org.firstinspires.ftc.teamcode.MotorArm.ArmPositions[] **values**()

Class ResetHeadingAction

```
java.lang.Object
    |
    +--RobotAction
        |
        +--org.firstinspires.ftc.teamcode.ResetHeadingAction
```

< Methods >

---

public class **ResetHeadingAction**
extends RobotAction

Loaded into the run map as an action that turns the robot. Each action is parameterized by the CSV file.

Author:

Andrew, Error 404: Team Name Not Found

RobotAction

Methods

entry

`public void entry()`

Placeholder for entry. Calls the parent entry method.

Overrides:

entry in class RobotAction

---

execute

`public boolean execute()`

The body of the action to be executed: Calls the reset() method in MecanumChassis

Overrides:

execute in class RobotAction

---

exit

```
public void exit()
```

Placeholder for exit. Calls the parent exit method.

Overrides:

exit in class RobotAction

---

init

```
public void init(Telemetry telem,
                 RuckusBot theRobot)
```

Placeholder for initialization. Currently only calls the parent init method.

Overrides:

init in class RobotAction

Class RobotAction

```
java.lang.Object
    |
    +--org.firstinspires.ftc.teamcode.RobotAction
```

Direct Known Subclasses:

ArmAction, CameraAction, DriveAction, GyroAction, HangAction, MarkDeployAction, ResetHeadingAction, TurnAction, WaitAction

---

< Methods >

---

public class **RobotAction**
extends java.lang.Object

Author:

Andrew, Error 404: Team Name Not Found

Methods

entry

`public void entry()`

Sets the start time variable equal to the in-system timer.

---

execute

`public boolean execute()`

The body of the action to execute.

---

exit

`public void exit()`

Called at the end of an action.

---

getRuntime

```
public double getRuntime()
```

Get the number of seconds this op mode has been running

This method has sub millisecond accuracy.

Returns:

number of seconds this op mode has been running

---

init

```
public void init(Telemetry telem,
                 RuckusBot theRobot)
```

Placeholder for initialization. Currently only calls the parent init method.

---

resetStartTime

```
public void resetStartTime()
```

Reset the start time to zero.

Class RuckusBot

```
java.lang.Object
    |
    +--org.firstinspires.ftc.teamcode.RuckusBot
```

< Constructors > < Methods >

---

public class **RuckusBot**
extends java.lang.Object

Defines the robot. Has class objects for each mechanism in use on the robot and contains the all the methods used by the robot (sourced from each individual mechanism class.

Author:

Andrew, Error 404: Team Name Not Found

Constructors

RuckusBot

`public` **`RuckusBot`**`(java.lang.String chassisType)`

Determines which chassis type to use: Mecanum or Tank.

Parameters:

chassisType - A string that is the type of chassis the code will use.

Methods

armCraterExtend

`public boolean` **`armCraterExtend`**`()`

Drives the arm to the position where it is fully extended into the crater in preparation for collecting minerals.

Returns:

A boolean that tells whether or not the arm is moving.

---

armDrive

```
public void armDrive(double RightStickY,
                     double LeftStickY)
```

A go-between between the opmode class and the actual chassis class. Drives the mineral arm using joystick inputs.

Parameters:

RightStickY - The y-axis of the right stick on the gamepad. Controls the elbow joint.
LeftStickY - The y-axis of the left stick on the gamepad. Controls the shoulder joint.

---

armDrivingExtend

```
public boolean armDrivingExtend()
```

Drives the arm to the position where it is where it is partially extended to make it easier to drive around between the crater and the lander.

Returns:

A boolean that tells whether or not the arm is moving.

---

armGoldCollect

```
public boolean armGoldCollect(boolean elbowSecond)
```

---

armHome

```
public boolean armHome()
```

Drives the arm to the home position where it is fully stowed.

Returns:

A boolean that tells whether or not the arm is moving.

---

armLanderExtend

`public boolean armLanderExtend()`

Drives the arm to the position where it is up next to the lander in preparation for putting minerals into the lander cargo hold.

Returns:

A boolean that tells whether or not the arm is moving.

---

collectorStop

`public void collectorStop()`

A go-between between the opmode class and the actual chassis class. Tells the collector wheels to stop turning.

---

collectorStopL

`public void collectorStopL()`

A go-between between the opmode class and the actual chassis class. Tells the left collector wheel to stop turning.

---

collectorStopR

`public void collectorStopR()`

A go-between between the opmode class and the actual chassis class. Tells the right collector wheel to stop turning.

---

drive

```
public boolean drive(double power,
                     double direction,
                     double gain,
                     double distance,
                     double time)
```

A go-between between the opmode class and the actual chassis class. Used in autonomous, can drive the robot in any direction for a set distance in inches while using the gyro to preserve the robot's initial orientation and having a timout feature that makes sure the algorithm doesn't get stuck.

Parameters:

power - The power at which the robot will drive.
direction - The direction in which the robot will drive (there are eight possible directions: forward, backwards, side to side, and 45-degree angle in any direction).
gain - The amount the algorithm will correct error by.
distance - The distance in inches the robot will drive (uses either the front right of front left motor to measure distance, depending on which direction the robot is driving).
time - The maximum time the move can take before the algorithm terminates the move and continues to the next command (prevents the software from freezing up and ruining the run).

---

eject

```
public void eject()
```

A go-between between the opmode class and the actual chassis class. Turns the collector wheels outward to eject minerals.

---

ejectL

```
public void ejectL()
```

A go-between between the opmode class and the actual chassis class. Turns the left collector wheel outward to eject minerals.

---

ejectR

```
public void ejectR()
```

A go-between between the opmode class and the actual chassis class. Turns the right collector wheel outward to eject minerals.

---

getResetHeading

```
public double getResetHeading()
```

Turns the chassis to the original heading caputred during init.

---

goTo

```
public boolean goTo(MotorArm.ArmPositions position,
                    double elbPower,
                    double shouldPower)
```

Drives the arm to the position specified by the parameter.

Parameters:

position - The target position to drive the arm to.
elbPower - The power to drive the elbow at.
shouldPower - The power to drive the shoulder at.

---

goldPosition

```
public java.lang.String goldPosition()
```

---

goodPitch

```
public boolean goodPitch()
```

Determines whether or not the robot is stuck on the lander using the pitch of the robot.

Returns:

A boolean that is whether or not the robot is stuck.

---

hangControl

```
public void hangControl(boolean down,
                        boolean up,
                        double power)
```

Used to move the hanger up and down with d-pad controls.

Parameters:

down - A boolean that is whether or not to move up.
up - A boolean that is whether or not to move down.
power - The power that the hanger will move at.

---

hangDrive

```
public boolean hangDrive(double power,
                         double distance,
org.firstinspires.ftc.teamcode.Gen2_Hang.HangDirection direction)
```

Drives the hanger a set distance.

Parameters:

power - The power that the hanger will move at.
distance - The distance the hanger will move.
direction - The direction the hanger will move in (two options: IN or OUT).

---

init

```
public void init(HardwareMap hwMap,
                 Telemetry telem,
                 boolean useCamera)
```

Triggers the initialization of the selected classes.

Parameters:

hwMap - An instance of the FIRST-provided HardwareMap which is passed onto more specific classes for initilizng hardware.
telem - An instance of Telemetry which allows the use of Telemtry in this class.

---

intake

```
public void intake()
```

A go-between between the opmode class and the actual chassis class. Turns the collector wheels inward to suck in minerals.

intakeL

```
public void intakeL()
```

A go-between between the opmode class and the actual chassis class. Turns the left collector wheel inward to suck in minerals.

intakeR

```
public void intakeR()
```

A go-between between the opmode class and the actual chassis class. Turns the right collector wheel inward to suck in minerals.

joystickDrive

```
public void joystickDrive(double leftStickX,
                          double leftStickY,
                          double rightStickX,
                          double rightStickY,
                          double powerLimit)
```

A go-between between the opmode class and the actual chassis class. The primary drive method in use by 404. All other drive methods call this one. JoystickDrive uses mecanum calculations to interpret joystick inputs and give directional power to the motors that allows for omni-directional driving.

Parameters:

leftStickX - The x-axis on the left stick of the gamepad. Controls the robot's strafing.
leftStickY - The y-axis on the left stick of the gamepad. Controls the forward and backward motions on the robot.
rightStickX - The x-axis on the right stick of the gamepad. N/A
rightStickY - The y-axis on the right stick of the gamepad. Controlls the robot's turning.

markDeploy

```
public void markDeploy(double power)
```

A go-between between the opmode class and the actual chassis class. Used in autonomous, deposits the team marker off the side of the robot.

pointTurn

```
public boolean pointTurn(double power,
                         double targetHeading,
                         double time)
```

A go-between between the opmode class and the actual chassis class. Used in autonomous, turns the robot to a target heading. The algorithm finds the shortest direction to the target heading and turns that direction.

Parameters:

power - The power at which the robot will turn.
targetHeading - The heading the robot wil turn to.
time - The maximum time the move can take before the algorithm terminates the move and continues to the next command (prevents the software from freezing up and ruining the run).

---

reset

```
public boolean reset(double power,
                     double time)
```

Turns the robot back to the initial heading it started out at. Useful for correcting heading errors introduced when landing in autonomous.

Parameters:

power - The power at which the robot will drive.
time - The maximum amount of time the move can take before timing out.

Returns:

A boolean that is whether or not the robot is moving.

---

start

```
public void start()
```

Runs once when start is hit: starts the camera and the hang mechanism.

---

stopMotors

```
public void stopMotors()
```

A go-between between the opmode class and the actual chassis class. Sets all motors to zero-power.

---

tankDrive

```
public boolean tankDrive(double power,
                         org.firstinspires.ftc.teamcode.Chassis.TankDirection
direction,
                         double gain,
                         double distance,
                         double time)
```

A go-between between the opmode class and the actual chassis class. Used in autonomous, drives the robot forward and backward only and uses the gyro to preserve an absolute heading captured at the beginning of the match.

Parameters:

power - The power at which the robot will drive.
direction - The direction the robot wil drive at (an enum with two choices: forward and backward).
gain - The amount the algorithm will correct error by.
distance - The distance in inches the robot will drive.
time - The maximum time the move can take before the algorithm terminates the move and continues to the next command (prevents the software from freezing up and ruining the run).

---

tfodShutdown

```
public void tfodShutdown()
```

Shuts down the camera and tensorflow algorithm

Class TurnAction

```
java.lang.Object
    |
    +--RobotAction
        |
        +--org.firstinspires.ftc.teamcode.TurnAction
```

< Methods >

---

public class **TurnAction**
extends RobotAction

Loaded into the run map as an action that turns the robot. Each action is parameterized by the CSV file.

Author:

Andrew, Error 404: Team Name Not Found

RobotAction

Methods

entry

`public void entry()`

Placeholder for entry. Currently only calls the parent entry method.

Overrides:

entry in class RobotAction

---

execute

`public boolean execute()`

Calls the pointTurn() method in MecanumChassis.

Overrides:

execute in class RobotAction

---

exit

```
public void exit()
```

Stops all the motors and calls the parent exit method.

Overrides:

exit in class RobotAction

---

init

```
public void init(Telemetry telem,
                 RuckusBot theRobot)
```

Placeholder for initialization. Currently only calls the parent init method.

Overrides:

init in class RobotAction

Class WaitAction

```
java.lang.Object
    |
    +--RobotAction
        |
        +--org.firstinspires.ftc.teamcode.WaitAction
```

< Methods >

---

public class **WaitAction**
extends RobotAction

Loaded into the run map as an action that waits a set amount of time. Each action is parameterized by the CSV file.

Author:

Andrew, Error 404: Team Name Not Found

RobotAction

Methods

entry

`public void entry()`

Overrides:

entry in class RobotAction

---

execute

`public boolean execute()`

Placeholder for execute. Calls the parent execute method.

Overrides:

execute in class RobotAction

---

exit

```
public void exit()
```

Stops the motors and servos and calls the parent exit method.

Overrides:

exit in class RobotAction

---

init

```
public void init(Telemetry telem,
                 RuckusBot theRobot)
```

Placeholder for entry. Currently only calls the parent entry method.

Overrides:

init in class RobotAction