/\* Copyright (c) 2017 FIRST. All rights reserved.

\*

\* Redistribution and use in source and binary forms, with or without modification,

\* are permitted (subject to the limitations in the disclaimer below) provided that

\* the following conditions are met:

\*

\* Redistributions of source code must retain the above copyright notice, this list

\* of conditions and the following disclaimer.

\*

\* Redistributions in binary form must reproduce the above copyright notice, this

\* list of conditions and the following disclaimer in the documentation and/or

\* other materials provided with the distribution.

\*

\* Neither the name of FIRST nor the names of its contributors may be used to endorse or

\* promote products derived from this software without specific prior written permission.

\*

\* NO EXPRESS OR IMPLIED LICENSES TO ANY PARTY'S PATENT RIGHTS ARE GRANTED BY THIS

\* LICENSE. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS

\* "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,

\* THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE

\* ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE

\* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL

\* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR

\* SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER

\* CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,

\* OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE

\* OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

\*/

package org.firstinspires.ftc.teamcode;

import com.qualcomm.robotcore.hardware.DcMotor;

import com.qualcomm.robotcore.hardware.CRServo;

import com.qualcomm.robotcore.hardware.HardwareMap;

import com.qualcomm.robotcore.hardware.Servo;

import com.qualcomm.robotcore.util.ElapsedTime;

import com.qualcomm.hardware.bosch.BNO055IMU;

import com.qualcomm.robotcore.eventloop.opmode.Autonomous;

import com.qualcomm.robotcore.eventloop.opmode.LinearOpMode;

import com.qualcomm.robotcore.hardware.DcMotor;

import com.qualcomm.robotcore.hardware.DigitalChannel;

import org.firstinspires.ftc.robotcore.external.navigation.AngleUnit;

import org.firstinspires.ftc.robotcore.external.navigation.AxesOrder;

import org.firstinspires.ftc.robotcore.external.navigation.AxesReference;

import org.firstinspires.ftc.robotcore.external.navigation.Orientation;

import org.firstinspires.ftc.robotcore.external.navigation.Position;

import org.firstinspires.ftc.robotcore.external.navigation.Velocity;

/\*\*

\* This is NOT an opmode.

\*

\* This class can be used to define all the specific hardware for a single robot.

\* In this case that robot is a Pushbot.

\* See PushbotTeleopTank\_Iterative and others classes starting with "Pushbot" for usage examples.

\*

\* This hardware class assumes the following device names have been configured on the robot:

\* Note: All names are lower case and some have single spaces between words.

\*

\* Motor channel: Left drive motor: "left\_drive"

\* Motor channel: Right drive motor: "right\_drive"

\* Motor channel: Manipulator drive motor: "left\_arm"

\* Servo channel: Servo to open left claw: "left\_hand"

\* Servo channel: Servo to open right claw: "right\_hand"

\*/

public class HardwarePushbot

{

/\* Public OpMode members. \*/

public DcMotor leftDrive = null;

public DcMotor rightDrive = null;

public DcMotor liftMotor = null;

public DcMotor armMotor1 = null;

public DcMotor armMotor2 = null;

public DcMotor extendMotor = null;

public CRServo clServo = null;

public Servo rtServo = null;

public CRServo marker = null;

public static final double MID\_SERVO = 0.5 ;

public static final double ARM\_UP\_POWER = 0.45 ;

public static final double ARM\_DOWN\_POWER = -0.45 ;

/\* local OpMode members. \*/

HardwareMap hwMap = null;

private ElapsedTime period = new ElapsedTime();

/\* Constructor \*/

public HardwarePushbot(){

}

/\* Initialize standard Hardware interfaces \*/

public void init(HardwareMap ahwMap) {

// Save reference to Hardware map

hwMap = ahwMap;

// Define and Initialize Motors

leftDrive = hwMap.get(DcMotor.class, "left\_drive");

rightDrive = hwMap.get(DcMotor.class, "right\_drive");

liftMotor = hwMap.get(DcMotor.class, "lift\_motor");

armMotor1 = hwMap.get(DcMotor.class, "arm\_motor1");

armMotor2 = hwMap.get(DcMotor.class, "arm\_motor2");

extendMotor = hwMap.get(DcMotor.class, "extend\_motor");

clServo = hwMap.get(CRServo.class, "cl\_servo");

rtServo = hwMap.get(Servo.class, "rt\_servo");

leftDrive.setDirection(DcMotor.Direction.FORWARD); // Set to REVERSE if using AndyMark motors

rightDrive.setDirection(DcMotor.Direction.REVERSE);// Set to FORWARD if using AndyMark motors

// Set all motors to zero power

leftDrive.setPower(0);

rightDrive.setPower(0);

//liftMotor.setPower(0);

// Set all motors to run without encoders.

// May want to use RUN\_USING\_ENCODERS if encoders are installed.

leftDrive.setMode(DcMotor.RunMode.RUN\_WITHOUT\_ENCODER);

rightDrive.setMode(DcMotor.RunMode.RUN\_WITHOUT\_ENCODER);

liftMotor.setMode(DcMotor.RunMode.RUN\_WITHOUT\_ENCODER);

armMotor1.setMode(DcMotor.RunMode.RUN\_USING\_ENCODERS);

armMotor2.setMode(DcMotor.RunMode.RUN\_USING\_ENCODERS);

extendMotor.setMode(DcMotor.RunMode.RUN\_USING\_ENCODERS);

// Define and initialize ALL installed servos.

/\* leftClaw = hwMap.get(Servo.class, "left\_hand");\*/

}

}