/\*----------------------------------------------------------------------------\*/

/\* Copyright (c) FIRST 2008. All Rights Reserved. \*/

/\* Open Source Software - may be modified and shared by FRC teams. The code \*/

/\* must be accompanied by the FIRST BSD license file in the root directory of \*/

/\* the project. \*/

/\*----------------------------------------------------------------------------\*/

package edu.wpi.first.wpilibj.templates;

import edu.wpi.first.wpilibj.SimpleRobot;

import edu.wpi.first.wpilibj.RobotDrive;

import edu.wpi.first.wpilibj.Joystick;

import edu.wpi.first.wpilibj.Victor;

import edu.wpi.first.wpilibj.Compressor;

import edu.wpi.first.wpilibj.Relay;

import edu.wpi.first.wpilibj.Timer;

import edu.wpi.first.wpilibj.smartdashboard.SmartDashboard;

public class RobotTemplate extends SimpleRobot {

RobotDrive robotDrive = new RobotDrive(10, 2, 4, 3);

Joystick leftJoystick = new Joystick(1);

Joystick rightJoystick = new Joystick(2);

Joystick controller = new Joystick(3);

Victor shooter1 = new Victor(6);

Victor shooter2 = new Victor(7);

Compressor compressor = new Compressor(1,1);

Relay hang = new Relay(3);

Relay discIn = new Relay(4);

Relay shooterAngle = new Relay(5);

public void robotInit(){

compressor.start();

}

public void autonomous() {

robotDrive.setSafetyEnabled(false);

discIn.set(Relay.Value.kReverse);

shooter1.set(1);

shooter2.set(1);

Timer.delay(3.0);

for (int i=0;i<3;i++){

discIn.set(Relay.Value.kReverse);

Timer.delay(1.5);

discIn.set(Relay.Value.kForward);

Timer.delay(1.5);

}

Timer.delay(1.0);

shooter1.set(0.0);

shooter2.set(0.0);

discIn.set(Relay.Value.kReverse);

}

public void operatorControl() {

robotDrive.setSafetyEnabled(true);

boolean isHanging = false;

boolean isPushing = false;

double yAxisInput;

while (isOperatorControl() && isEnabled()) {

robotDrive.tankDrive(leftJoystick, rightJoystick);

Timer.delay(0.01);

yAxisInput = controller.getAxis(Joystick.AxisType.kY);

if (yAxisInput > 0.25){

shooterAngle.set(Relay.Value.kForward);

}

else if (yAxisInput < -0.25){

shooterAngle.set(Relay.Value.kReverse);

}

else{

shooterAngle.set(Relay.Value.kOff);

}

if (controller.getRawButton(2)){

if (isHanging){

hang.set(Relay.Value.kOff);

}

else{

hang.set(Relay.Value.kForward);

isHanging = true;

}

}

else if ((!controller.getRawButton(2)) && (isHanging)){

hang.set(Relay.Value.kReverse);

isHanging = false;

}

else{

hang.set(Relay.Value.kOff);

}

if (controller.getRawButton(4)){

shooter1.set(1);

shooter2.set(1);

}

else{

shooter1.set(0.0);

shooter2.set(0.0);

}

if (controller.getRawButton(6)){

if (isPushing){

discIn.set(Relay.Value.kOff);

}

else{

discIn.set(Relay.Value.kForward);

isPushing = true;

}

}

else if ((!controller.getRawButton(2)) && (isPushing)){

discIn.set(Relay.Value.kReverse);

isPushing = false;

}

else{

discIn.set(Relay.Value.kOff);

}

SmartDashboard.putNumber("Left Wheels", leftJoystick.getAxis(Joystick.AxisType.kY));

SmartDashboard.putNumber("Right Wheels", rightJoystick.getAxis(Joystick.AxisType.kY));

SmartDashboard.putNumber("Shooter", shooter1.get());

SmartDashboard.putBoolean("Is Pushing", isPushing);

SmartDashboard.putBoolean("Is Hanging", isHanging);

}

}

public void test() {

}

}