```
// Harris Christiansen
// Copyright Harris Christiansen 2013
// FRC Team 3245 - Waterford School
// 2013 Competition Bot - Utah+Las Vegas Regional
// Written Feb 5, 2013
// Updated 4-1-13

// Package
package edu.first.team3245;

// Imports
import edu.wpi.first.wpilibj.*;
import InsightLT.InsightLT;
import InsightLT.StringData;
import InsightLT.IntegerData;
import InsightLT.DecimalData;
public class UtCompBot extends IterativeRobot {
    // Shooter Speed
    double shtSetSpeed=0.75;
    // Pilot Controls
    int leftStick=2, rightStick=4, fastBtn=8, slowBtn=7, lowGearBtn=5,
highGearBtn=6, revDrvBtn=3, shtBtn=2, shtPistonBtn=4;
    int coAutoShtBtn=8, coShtBtn=6, coShtPistonBtn=5, coShtSpeedInc=3,
coShtSpeedDec=1, togCompBtn=10, coRevShtBtn=4, retractStkPis=9;

        // Motors
    private Talon leftMotor, rightMotor, shtMot1, shtMot2, shtMot3;

    // Current Motor Speeds
    private double leftSpeed, rightSpeed, shtSpeed;

    // Controllers
    Joystick pilotStick, coPilotStick;

    // Compressor
    Compressor mainComp;

    // Solenoids
    Solenoid driveLowSole, driveHighSole, shtOutSole, shtInSole,
shtStkOutSole, shtStkInSole;

        // Drive Direction // true=normal, false=reversed
        boolean driveDirection=true;

    // Sensors
    InsightLT LTDisp = new InsightLT(InsightLT.TWO_ONE_LINE_ZONES);
    DecimalData LTrowOne = new DecimalData("Batt:");
    IntegerData LTrowTwo = new IntegerData("Sht Pct:");
    Gyro mainGyro;
    Encoder rightEncoder;
```

```
// Auto State
int autoState=4;
public int autoPeriodCount=0;
public boolean autoShot=false;

public void robotInit() {
    // PWM Motor Ports
    rightMotor = new Talon(2);
    leftMotor = new Talon(1);
shtMot1 = new Talon(4);
shtMot2 = new Talon(5);
shtMot3 = new Talon(6);


    // Joysticks
    pilotStick = new Joystick(1);
    coPilotStick = new Joystick(2);

// Create Compressor Instance
mainComp = new Compressor(14,1);

// Sensors
mainGyro = new Gyro(2);
mainGyro.setSensitivity(.007);
mainGyro.reset();
rightEncoder=new Encoder(8,9);
rightEncoder.start();

// Create Solenoid Instances
driveLowSole = new Solenoid(1,7);
driveHighSole = new Solenoid(1,6);
shtOutSole = new Solenoid(1,5);
shtInSole = new Solenoid(1,4);
shtStkOutSole = new Solenoid(1,3);
shtStkInSole = new Solenoid(1,2);
LTDisp.startDisplay();
LTDisp.registerData(LTrowOne, 1);
LTDisp.registerData(LTrowTwo, 2);
LTrowTwo.setData(3245);

    zeroMotorSpeeds();
}
public void autonomousInit() {
    zeroMotorSpeeds();
    // Start Compressor
autoPeriodCount=0;
mainComp.start();
mainGyro.reset();
rightEncoder.reset();
```

```
    }
    public void autonomousPeriodic() {
    if(autoState==1||autoState==2) {updateShootTurnAuto();}
    if(autoState==3) {updateAutoWaitShoot();}
    updateMotors();
    }
    public void teleopInit() {
        zeroMotorSpeeds();
    autoPeriodCount=0;
        // Start Compressor
    mainComp.start();
    }
    public void teleopPeriodic() {
        updateDrive();
        updateDriveShifter();
        updateReverseDrive();
    updateShooterPiston();
    updateToggleComp();
    updateShooter();
    //updateShooterAuto();
    updateShooterSpeed();
        updateMotors();
    }
    public void disabledInit() {
        zeroMotorSpeeds();

    }
    public void disabledPeriodic() {
        if(DriverStation.getInstance().getDigitalIn(1))
{ autoState=1;} // Backup
        if(DriverStation.getInstance().getDigitalIn(2))
{ autoState=2;} // No Backup
        if(DriverStation.getInstance().getDigitalIn(3))
{ autoState=3;} // Wait Delay
    updateLTDisp();
    }

    public void testInit() {
    mainGyro.reset(); // analog port 2
    rightEncoder.reset(); // port 8 and 9
    }
    public void testPeriodic() {
        updateDrive();
        updateDriveShifter();
        updateReverseDrive();
    updateShooterPiston();
    updateToggleComp();
    updateShooter();
    //updateShooterAuto();
    updateShooterSpeed();
```

```
        updateMotors();
    updateLTDisp();
    System.out.println(mainGyro.getAngle());
    //System.out.println(rightEncoder.get());
    //printController();
    }

    /////////// CUSTOM FUNCTIONS /////////////

    // Zero Motor Speeds
    public void zeroMotorSpeeds() {
        leftSpeed=0;
        rightSpeed=0;
    shtSpeed=0;
    }

    // Pneumatics Drive Train Shifting
    public void updateDriveShifter() {
        if(pilotStick.getRawButton(lowGearBtn)) {
            driveLowSole.set(true);
            driveHighSole.set(false);
        } else if(pilotStick.getRawButton(highGearBtn)) {
            driveLowSole.set(false);
            driveHighSole.set(true);
        } else {
            driveLowSole.set(false);
            driveHighSole.set(false);
        }
    }

    // Tank Drive
    public void updateDrive() {
        double drivePercent=0.55;
        if(pilotStick.getRawButton(fastBtn)) {drivePercent=1.00;}
        else if(pilotStick.getRawButton(slowBtn)) {drivePercent=0.3;}
        if(driveDirection) { // Normal Drive Direction
            leftSpeed=pilotStick.getRawAxis(leftStick)*drivePercent;
            rightSpeed=pilotStick.getRawAxis(rightStick)*drivePercent;
        } else { // Reversed Drive Direction
            rightSpeed=-
(pilotStick.getRawAxis(leftStick)*drivePercent);
            leftSpeed=-
(pilotStick.getRawAxis(rightStick)*drivePercent);
        }
    }

    // Reverse Robot Drive Direction
    private boolean driveChanged=false;
    public void updateReverseDrive() {
        if(pilotStick.getRawButton(revDrvBtn)&&!driveChanged)
```

```
{driveChanged=true; driveDirection=!driveDirection;}
        if(!pilotStick.getRawButton(revDrvBtn)){driveChanged=false;}
    }
    public int shtPisCount=0;
    public int shootState=0;
    public boolean shotOne=false;
    public void updateShooterPiston() {
    if((coPilotStick.getRawButton(coShtPistonBtn)||
pilotStick.getRawButton(shtPistonBtn))&&(coPilotStick.getRawButton(coS
htBtn)||pilotStick.getRawButton(shtBtn))) {
        if(shootState==0) {
        shootState=1;
        }
    }
    if(shotOne&&shootState==1) {
        if(shtPisCount<14) {
        shtInSole.set(true);
        shtOutSole.set(false);
        shtPisCount++;
        } else {
        shtInSole.set(false);
        shtOutSole.set(false);
        shootState=2;
        }
    }
    if(shotOne&&shootState==2) {
        if(shtPisCount>0) {
        shtInSole.set(false);
        shtOutSole.set(true);
        shtPisCount--;
        } else {
        shtInSole.set(false);
        shtOutSole.set(false);
        shootState=0;
        }
    }
    if(!shotOne&&shootState==1) {
        if(shtPisCount<30) {
        shtStkInSole.set(true);
        shtStkOutSole.set(false);
        shtPisCount++;
        if(shtPisCount>15) {
            shtInSole.set(true);
            shtOutSole.set(false);
        }
        } else {
        shtInSole.set(false);
        shtOutSole.set(false);
        shtStkInSole.set(false);
        shtStkOutSole.set(false);
        shootState=2;
        }
    }
    else if(!shotOne&&shootState==2) {
        if(shtPisCount>0) {
        shtInSole.set(false);
        shtOutSole.set(true);
        shtPisCount--;
        } else {
        shtStkInSole.set(false);
        shtStkOutSole.set(false);
        shtInSole.set(false);
        shtOutSole.set(false);
        shootState=0;
        shotOne=true;
        }
    }
    }
    public int returnStkPisCnt=10;
    public void updateShooter() {
    if(coPilotStick.getRawButton(coShtBtn)||
pilotStick.getRawButton(shtBtn)) {
        shtSpeed=shtSetSpeed;
        if(mainComp.enabled()) {
        mainComp.stop();
        }
    }
    else if(coPilotStick.getRawButton(coRevShtBtn)) {
        shtSpeed=-1.0;
    }
    else {
        shtSpeed=0.0;
        if(shotOne&&returnStkPisCnt>0) {
        shtStkInSole.set(false);
        shtStkOutSole.set(true);
        returnStkPisCnt--;
        } else {
        shtStkInSole.set(false);
        shtStkOutSole.set(false);
        returnStkPisCnt=10;
        shotOne=false;
        }
        if(!mainComp.enabled()) {
        mainComp.start();
        }
    }
    }

    public boolean shtSpeedUpdated=false;
    public void updateShooterSpeed() {
```

```
if(coPilotStick.getRawButton(coShtSpeedInc)) {
    if(!shtSpeedUpdated) {
    shtSetSpeed=shtSetSpeed+0.05;
    System.out.println("Shooter Speed: "+shtSetSpeed);
    shtSpeedUpdated=true;
    }
}
else if(coPilotStick.getRawButton(coShtSpeedDec)) {
    if(!shtSpeedUpdated) {
    shtSetSpeed=shtSetSpeed-0.05;
    System.out.println("Shooter Speed: "+shtSetSpeed);
    shtSpeedUpdated=true;
    }
}
else { shtSpeedUpdated=false; }

}
boolean togCompSwitched=false;
public void updateToggleComp() {
if(coPilotStick.getRawButton(togCompBtn)) {
    if(!togCompSwitched) {
    togCompSwitched=true;
    if(mainComp.enabled()) {
        mainComp.stop();
    } else {
        mainComp.start();
    }
    }
} else {
    togCompSwitched=false;
}
}

// Set Motor Speeds
public void updateMotors() {
    leftMotor.set(-leftSpeed); // Motor Reversed
    rightMotor.set(rightSpeed*.95);
shtMot1.set(-shtSpeed); // Motor Reversed
shtMot2.set(-shtSpeed*.75); // Motor Reversed
shtMot3.set(-shtSpeed); // Motor Reversed
}
public void updateLTDisp() {
LTrowOne.setData(DriverStation.getInstance().getBatteryVoltage());
LTrowTwo.setData((int)(shtSetSpeed*100));
}




//// Autonomous Routines
```

```
    public double tarAngDif=0;
    public void updateShootTurnAuto() {
    if(autoPeriodCount<270) {
        shtSpeed=0.71;
        if(autoPeriodCount<12) {
        if(mainComp.enabled()) {
            mainComp.stop();
        }
        shtStkOutSole.set(false);
        shtStkInSole.set(true);
        }
    } else {
        shtSpeed=0;
    }
    if((autoPeriodCount%65)>45&&(autoPeriodCount
%65)<55&&autoPeriodCount<270) {
        shtInSole.set(true);
        shtOutSole.set(false);
        if(!autoShot) {
        System.out.println("Shot One Auto Disk at: "+shtSpeed);
        autoShot=true;
        }
    }
    else if((autoPeriodCount%65)>55&&(autoPeriodCount
%65)<65&&autoPeriodCount<270) {
        shtInSole.set(false);
        shtOutSole.set(true);
        autoShot=false;
    }
    else {
        shtInSole.set(false);
        shtOutSole.set(false);
    }
    autoPeriodCount++;
    if(autoPeriodCount>270&&autoPeriodCount<300) {
        if(autoPeriodCount<285) {
        shtStkOutSole.set(true);
        shtStkInSole.set(false);
        } else {
        shtStkOutSole.set(false);
        shtStkInSole.set(false);
        }
        leftSpeed=0.45;
        rightSpeed=0.45;
        if(!mainComp.enabled()) {
        mainComp.start();
        }
    } else
if(autoState==1&&autoPeriodCount>300&&autoPeriodCount<350&&rightEncode
r.get()>-12500) {
        //// Back Up Auto
        leftSpeed=0.45;
        rightSpeed=0.45;
    } else {
        leftSpeed=0.0;
        rightSpeed=0.0;
    }
    }

    public void updateAutoWaitShoot() {
    if(autoPeriodCount>390&&autoPeriodCount<660) {
        shtSpeed=0.71;
        if(autoPeriodCount<400) {
        if(mainComp.enabled()) {
            mainComp.stop();
        }
        shtStkOutSole.set(false);
        shtStkInSole.set(true);
        }
    } else {
        shtSpeed=0;
    }
    if((autoPeriodCount%65)>45&&(autoPeriodCount
%65)<55&&autoPeriodCount>390&&autoPeriodCount<660) {
        shtInSole.set(true);
        shtOutSole.set(false);
        if(!autoShot) {
        System.out.println("Shot One Auto Disk at: "+shtSpeed);
        autoShot=true;
        }
    }
    else if((autoPeriodCount%65)>55&&(autoPeriodCount
%65)<65&&autoPeriodCount>390&&autoPeriodCount<660) {
        shtInSole.set(false);
        shtOutSole.set(true);
        autoShot=false;
    }
    else {
        shtInSole.set(false);
        shtOutSole.set(false);
    }
    autoPeriodCount++;
    if(autoPeriodCount>660&&autoPeriodCount<790) {
        if(autoPeriodCount<670) {
        shtStkOutSole.set(true);
        shtStkInSole.set(false);
        } else {
        shtStkOutSole.set(false);
        shtStkInSole.set(false);
        }
        if(!mainComp.enabled()) {
        mainComp.start();
        }
    }
    }

    public void printController() {
    System.out.println("Axis 1: "+pilotStick.getRawAxis(1));
    System.out.println("Axis 2: "+pilotStick.getRawAxis(2));
    System.out.println("Axis 3: "+pilotStick.getRawAxis(3));
    System.out.println("Axis 4: "+pilotStick.getRawAxis(4));
    System.out.println("Axis 5: "+pilotStick.getRawAxis(5));
    System.out.println("Axis 6: "+pilotStick.getRawAxis(6));
    }
}
```