

```
/* Copyright (c) 2015 Qualcomm Technologies Inc
```

```
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 Qualcomm Technologies Inc 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 android.graphics.Color;
```

```
import com.qualcomm.robotcore.eventloop.opmode.Autonomous;
import com.qualcomm.robotcore.eventloop.opmode.Disabled;
import com.qualcomm.robotcore.eventloop.opmode.LinearOpMode;
import com.qualcomm.robotcore.util.ElapsedTime;
```

```
/*
 *
 * This is an example LinearOpMode that shows how to use
 * a Modern Robotics Color Sensor.
 *
 * The op mode assumes that the color sensor
 * is configured with a name of "color sensor".
 *
 * Use Android Studio to Copy this Class, and Paste it into your team's code folder with a new name.
 * Remove or comment out the @Disabled line to add this opmode to the Driver Station OpMode list
 */
```

```
@Autonomous(name = "Babybot: See Color", group = "Abdul")
@Disabled
public class BabyMRCOLOR_SeeColor extends LinearOpMode {
```

```
    /* Declare OpMode members. */
    HardwareBabybot    robot    = new HardwareBabybot();    // Use a Babybot's hardware
    private ElapsedTime    runtime = new ElapsedTime();
```

```
    static final double    FORWARD_SPEED = 0.6;
    static final double    TURN_SPEED    = 0.5;
```

```
    @Override
    public void runOpMode() throws InterruptedException {

        /*
         * Initialize the drive system variables.
         * The init() method of the hardware class does all the work here
         */
        robot.init(hardwareMap);
```

```
        // Send telemetry message to signify robot waiting;
        telemetry.addData("Status", "Ready to See");    //
        telemetry.update();
```

```
        // hsvValues is an array that will hold the hue, saturation, and value information.
        float hsvValues[] = {0F,0F,0F};
```

```
        // values is a reference to the hsvValues array.
        final float values[] = hsvValues;
```

```
        // wait for the start button to be pressed.
        waitForStart();
```

```
        // while the op mode is active, loop and read the RGB data.
        // Note we use opModeIsActive() as our loop condition because it is an interruptible method.
```

```

// Step 0: Check for colors

// convert the RGB values to HSV values.
Color.RGBToHSV(robot.colorSensor.red() * 8, robot.colorSensor.green() * 8, robot.colorSensor.blue() * 8, hsvValues);

while (opModeIsActive() ) {
  ////////// Color sensor activating servos based on what the RED/BLUE value is greatest
  if (robot.colorSensor.red() > robot.colorSensor.blue() && robot.colorSensor.red() > robot.colorSensor.green()) {
    robot.claw.setPosition(0.1);
    robot.arm.setPosition(0.1); // Red servo is activated
  } else if (robot.colorSensor.blue() > robot.colorSensor.red() && robot.colorSensor.blue() > robot.colorSensor.green()) {
    robot.claw.setPosition(0.9); // Blue servo is activated
    robot.arm.setPosition(0.9);
  } else {
    robot.claw.setPosition(0.1); // start position
    robot.arm.setPosition(0.9); // start position
  }
  //////////////////////////////////////
  // send the info back to driver station using telemetry function.
  telemetry.addData("LED", robot.bLedOn ? "On" : "Off");
  telemetry.addData("Clear", robot.colorSensor.alpha());
  telemetry.addData("Red  ", robot.colorSensor.red());
  telemetry.addData("Green", robot.colorSensor.green());
  telemetry.addData("Blue ", robot.colorSensor.blue());
  telemetry.addData("Hue", hsvValues[0]);

  telemetry.update();
  idle();
}
}
}

```