Daily Journal

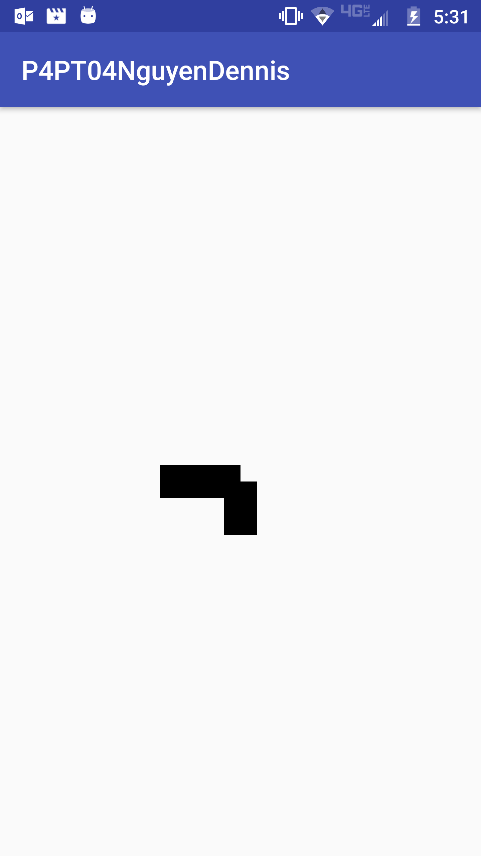
Name Dennis Nguyen

Date May 22, 2017

## Required work:

* Receive orientation data from sensors
* Draw arrows that change in size linked to degree of orientation.

# What I got done

Moved monitoring of sensor activity to a separate class

Create android Activity resume, destroy, pause, and create methods

**package** com.dragonfury.duy.p4pt04nguyendennis;   
  
**import** android.graphics.Canvas;  
**import** android.graphics.Color;  
**import** android.graphics.Paint;  
**import** android.graphics.Path;  
**import** android.hardware.Sensor;  
**import** android.hardware.SensorEvent;  
**import** android.hardware.SensorEventListener;  
**import** android.hardware.SensorManager;  
**import** android.view.View;  
**import** android.content.Context;  
  
*/\*\*  
 \* Created by 1383504 on 5/11/2017.  
 \*/***public class** DrawView **extends** View {  
 **public** DrawView(Context context) {  
 **super**(context);  
  
 }  
  
 **private** Paint **paint** = **new** Paint();  
 **private** SensorActivity **sensorActivity** = **new** SensorActivity(getContext());  
 **private** Path **path** = **new** Path();  
 **private float**[] **orientations** = **new float**[3];  
  
  
 @Override  
 **protected void** onDraw(Canvas canvas) {  
 **super**.onDraw(canvas);  
 **orientations** = **sensorActivity**.getOrientation();  
 **for** (**float** orientation: **orientations**) {  
 System.***out***.println(**"test:"**+orientation);  
 }  
  
 **paint**.setColor(Color.***BLACK***);  
 **paint**.setStrokeWidth(100 \* getWidth() / 1440);  
  
 **if** (**orientations**[0] > 0) {  
  
 }**else if** (**orientations**[0] < 0) {  
  
 }  
  
 **if** (**orientations**[1] > 0) {  
 canvas.drawLine(getWidth() / 2, getHeight() / 2, getWidth() / 2, 2 \* getHeight() / 3, **paint**);  
 }**else if** (**orientations**[1] <= 0) {  
 canvas.drawLine(getWidth() / 2, getHeight() / 2, getWidth() / 2, 2560 - (1280 - 7 \* **orientations** [2]), **paint**);  
 }  
  
 **if** (**orientations**[2] >= 0) {  
 canvas.drawLine(getWidth() / 2, getHeight() / 2, getWidth() / 3, getHeight() / 2, **paint**);  
 }**else if** (**orientations**[2] <= 0) {  
  
 }  
  
 invalidate();  
  
 }  
  
 **private void** drawTri(**int** x1, **int** y1, **int** x2, **int** y2, **int** x3, **int** y3, Canvas c, Paint p) {  
 **path**.moveTo(x1, y1);  
 **path**.lineTo(x2, y2);  
 **path**.lineTo(x3, y3);  
 **path**.close();  
 c.drawPath(**path**, p);  
 }  
  
  
}

**package** com.dragonfury.duy.p4pt04nguyendennis;   
  
**import** android.app.Activity;  
**import** android.content.Context;  
**import** android.hardware.Sensor;  
**import** android.hardware.SensorEvent;  
**import** android.hardware.SensorEventListener;  
**import** android.hardware.SensorManager;  
**import** android.os.Bundle;  
**import** android.support.annotation.Nullable;  
  
*/\*\*  
 \* Created by 1383504 on 5/23/2017.  
 \*/***public class** SensorActivity **extends** Activity **implements** SensorEventListener{  
  
  
  
 **public** SensorActivity(Context context) {  
 **sensorManager** = (SensorManager) context.getSystemService(Context.***SENSOR\_SERVICE***);  
  
 }  
  
 **private float**[] **orientations** = **new float**[3];  
 **private float**[] **accelerometer** = **new float**[3];  
 **private float**[] **geoField** = **new float** [3];  
 **private float**[] **rotateMatrix** = **new float**[9];  
 **private** SensorManager **sensorManager**;  
 Sensor **accelero**;  
 Sensor **magField**;  
  
 @Override  
 **protected void** onCreate(@Nullable Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 **sensorManager** = (SensorManager)getSystemService(***SENSOR\_SERVICE***);  
 **accelero** = **sensorManager**.getDefaultSensor(Sensor.***TYPE\_ACCELEROMETER***);  
 **magField** = **sensorManager**.getDefaultSensor(Sensor.***TYPE\_MAGNETIC\_FIELD***);  
 }  
  
 @Override  
 **public void** onSensorChanged(SensorEvent sensorEvent) {  
 **if** (sensorEvent.**sensor**.getType() == Sensor.***TYPE\_MAGNETIC\_FIELD***) { *//Calculates magnetic field around device and stores it* **geoField** = sensorEvent.**values**;  
 }  
  
 **if** (sensorEvent.**sensor**.getType() == Sensor.***TYPE\_ACCELEROMETER***) {  
 **accelerometer** = sensorEvent.**values**;  
 }  
  
 **sensorManager**.*getRotationMatrix*(**rotateMatrix**, **null**, **accelerometer**, **geoField**); *//Calculate RotationMatrix - Device screen is facing sky, top is towards north pole* **sensorManager**.*getOrientation*(**rotateMatrix**, **orientations**); *//Calculate amount of rotation along x, y, and z axis* System  
 }  
  
 @Override  
 **public void** onAccuracyChanged(Sensor sensor, **int** i) {  
  
 }  
  
 **protected void** onResume() {  
 **super**.onResume();  
*// sensorManager.registerListener(this, accelerometer, SensorManager.SENSOR\_DELAY\_UI);  
// sensorManager.registerListener(this, geoField, SensorManager.SENSOR\_DELAY\_UI);* }  
  
 **protected void** onPause() {  
 **super**.onPause();  
 **sensorManager**.unregisterListener(**this**);  
 }  
  
 **public float**[] getOrientation() {  
 **return orientations**;  
 }  
}

# What I need to do next class

Modify onSensorChanged() in SensorActivity to receive data, currently can only receive 0 values from the device

Finish creating SensorListener

# What I need to do before next class

* Assessments
* Study
* Sign up for…
* Tell everyone to take CS

Daily Journal

Name Dennis Nguyen

Date May 19, 2017

## Required work:

* Receive orientation data from sensors
* Draw arrows that change in size linked to degree of orientation.

# What I got done

Started drawing arrows

Began storing orientation values within usable variables

**package** com.dragonfury.duy.p4pt04nguyendennis;  
  
**import** android.graphics.Canvas;  
**import** android.graphics.Paint;  
**import** android.graphics.Path;  
**import** android.hardware.Sensor;  
**import** android.hardware.SensorEvent;  
**import** android.hardware.SensorEventListener;  
**import** android.hardware.SensorManager;  
**import** android.view.View;  
**import** android.content.Context;  
  
*/\*\*  
 \* Created by 1383504 on 5/11/2017.  
 \*/***public class** DrawView **extends** View {  
 **public** DrawView(Context context) {  
 **super**(context);  
  
 **sensorManager** = (SensorManager) context.getSystemService(Context.***SENSOR\_SERVICE***);  
 **listener** = **new** SensorEventListener() {  
 @Override  
 **public void** onSensorChanged(SensorEvent sensorEvent) {  
 **geoField** = sensorEvent.**values**;  
  
 **if** (sensorEvent.**sensor**.getType() == Sensor.***TYPE\_MAGNETIC\_FIELD***) { *//Calculates magnetic field around device and stores it* **geoField**[0] = sensorEvent.**values**[0];  
 **geoField**[1] = sensorEvent.**values**[1];  
 **geoField**[2] = sensorEvent.**values**[2];  
 }  
  
 **sensorManager**.*getRotationMatrix*(**rotateMatrix**, **null**, **accelerometer**, **geoField**); *//Calculate RotationMatrix - Device screen is facing sky, top is towards north pole* **sensorManager**.*getOrientation*(**rotateMatrix**, **orientations**); *//Calculate amount of rotation along x, y, and z axis* }  
  
 @Override  
 **public void** onAccuracyChanged(Sensor sensor, **int** i) {  
  
 }  
  
 };  
  
 }  
  
 **private float**[] **orientations** = **new float**[3];  
 **private float**[] **accelerometer** = **new float**[3];  
 **private float**[] **geoField** = **new float** [3];  
 **private float**[] **rotateMatrix** = **new float**[9];  
 **private** Sensor sensor;  
 SensorManager **sensorManager**;  
 **private** SensorEventListener **listener**; *//Detects when sensor is being used/changed* **private** Paint **paint** = **new** Paint();  
 **private float degrees**;  
 Path **path** = **new** Path();  
  
  
 @Override  
 **protected void** onDraw(Canvas canvas) {  
 **super**.onDraw(canvas);  
  
*// for (float radOrientation: orientations) { //Convert orientations values to degrees  
// radOrientation = (float) Math.toDegrees(radOrientation);  
// }* **for** (**float** orientation: **orientations**) {  
 System.***out***.println(**"test:"**+orientation);  
 }  
  
 **if** (**orientations**[0] > 0) {  
  
 }**else if** (**orientations**[0] < 0) {  
  
 }  
  
 **if** (**orientations**[1] > 0) {  
  
 }**else if** (**orientations**[1] < 0) {  
  
 }  
  
 **if** (**orientations**[2] > 0) {  
 canvas.drawLine(getWidth() / 2, getHeight() / 2, getWidth() / 2, 3 \* getHeight() / 2, **paint**);  
 }**else if** (**orientations**[2] < 0) {  
 canvas.drawLine(getWidth() / 2, getHeight() / 2, getWidth() / 2, 2560 - (1280 - 7 \* **orientations** [2]), **paint**);  
 }  
  
  
  
 }  
  
 **private void** drawTri(**int** x1, **int** y1, **int** x2, **int** y2, **int** x3, **int** y3, Canvas c, Paint p) {  
 **path**.moveTo(x1, y1);  
 **path**.lineTo(x2, y2);  
 **path**.lineTo(x3, y3);  
 **path**.close();  
 c.drawPath(**path**, p);  
 }  
  
  
  
  
}

# What I need to do next class

Make device constantly record orientation rather than just once at the start of the app

# What I need to do before next class

* Assessments
* Study
* Sign up for…
* Tell everyone to take CS

Daily Journal

Name Dennis Nguyen

Date May 17, 2017

## Required work:

* Receive orientation data from sensors
* Draw arrows that change in size linked to degree of orientation.

# What I got done

Collected data from device accelerometer and stored it into an array with 3 elements

Converted values accelerometer to degrees

**public class** DrawView **extends** View {  
 **public** DrawView(Context context) {  
 **super**(context);  
  
 **sensorManager** = (**sensorManager**) context.getSystemService(Context.***SENSOR\_SERVICE***);  
  
 **listener** = **new** SensorEventListener() {  
 @Override  
 **public void** onSensorChanged(SensorEvent sensorEvent) {  
 **float**[] accelerometer = sensorEvent.**values**;  
 **if** (sensorEvent.**sensor**.getType() == Sensor.***TYPE\_ACCELEROMETER***){  
 accelerometer[0] = sensorEvent.**values**[0];  
 accelerometer[1] = sensorEvent.**values**[1];  
 accelerometer[2] = sensorEvent.**values**[2];  
 **degrees** = (**float**) Math.*toDegrees*(Math.*atan*(Math.*abs*(accelerometer[1]) / Math.*abs*(accelerometer[2])));  
  
  
 }  
 }  
  
 @Override  
 **public void** onAccuracyChanged(Sensor sensor, **int** i) {  
  
 }  
 };  
 }  
  
 **private float**[] **orientation** = **new float**[3];  
 **private float**[] **accelerometer** = **new float**[3];  
 **private** Sensor **sensor**;  
 **private** SensorManager **sensorManager**;  
 **private** SensorEventListener **listener**;  
 **private** MainActivity **mainActivity**;  
 **private float degrees**;  
  
  
 @Override  
 **protected void** onDraw(Canvas canvas) {  
 **super**.onDraw(canvas);  
  
 }  
  
}

# What I need to do next class

* Complete code required to collect data
* Use gathered data to draw

# What I need to do before next class

* Assessments
* Study
* Sign up for…
* Tell everyone to take CS

Daily Journal

Name Dennis Nguyen

Date May 15, 2017

## Required work:

* Receive orientation data from sensors
* Draw arrows that change in size linked to degree of orientation.

# What I got done

* Tried to implement some orientation methods to retrieve data

**public class** DrawView **extends** View {  
 **public** DrawView(Context context) {  
 **super**(context);  
 context.getSystemService(Context.***SENSOR\_SERVICE***);  
 }  
  
 SensorManager **sensorManager** = **new** SensorManager();  
 **float**[] **orientation** = **new float**[3];  
 **float**[] **accelerometer** = **new float**[3];  
 SensorActivity **sensorActivity** = **new** SensorActivity();  
  
  
 @Override  
 **protected void** onDraw(Canvas canvas) {  
 **super**.onDraw(canvas);  
 getContext().getSystemService(Context.***SENSOR\_SERVICE***);  
 **accelerometer** = Sensor.***TYPE\_ACCELEROMETER***;  
 **orientation** = **sensorActivity**.*getOrientation*(**sensorManager**.*getRotationMatrix*(**null**, **null**, Sensor.***TYPE\_ACCELEROMETER***, Sensor.***TYPE\_MAGNETIC\_FIELD***), **orientation**);  
 }  
  
}

# What I learned

* getOrientation() returns 3 values: pitch, roll, and azimuth. All 3 corresponding to the XYZ plane

# What I need to do next class

Continue researching android orientation and receive data from the sensors

# What I need to do before next class

* Assessments
* Study
* Sign up for…
* Tell everyone to take CS