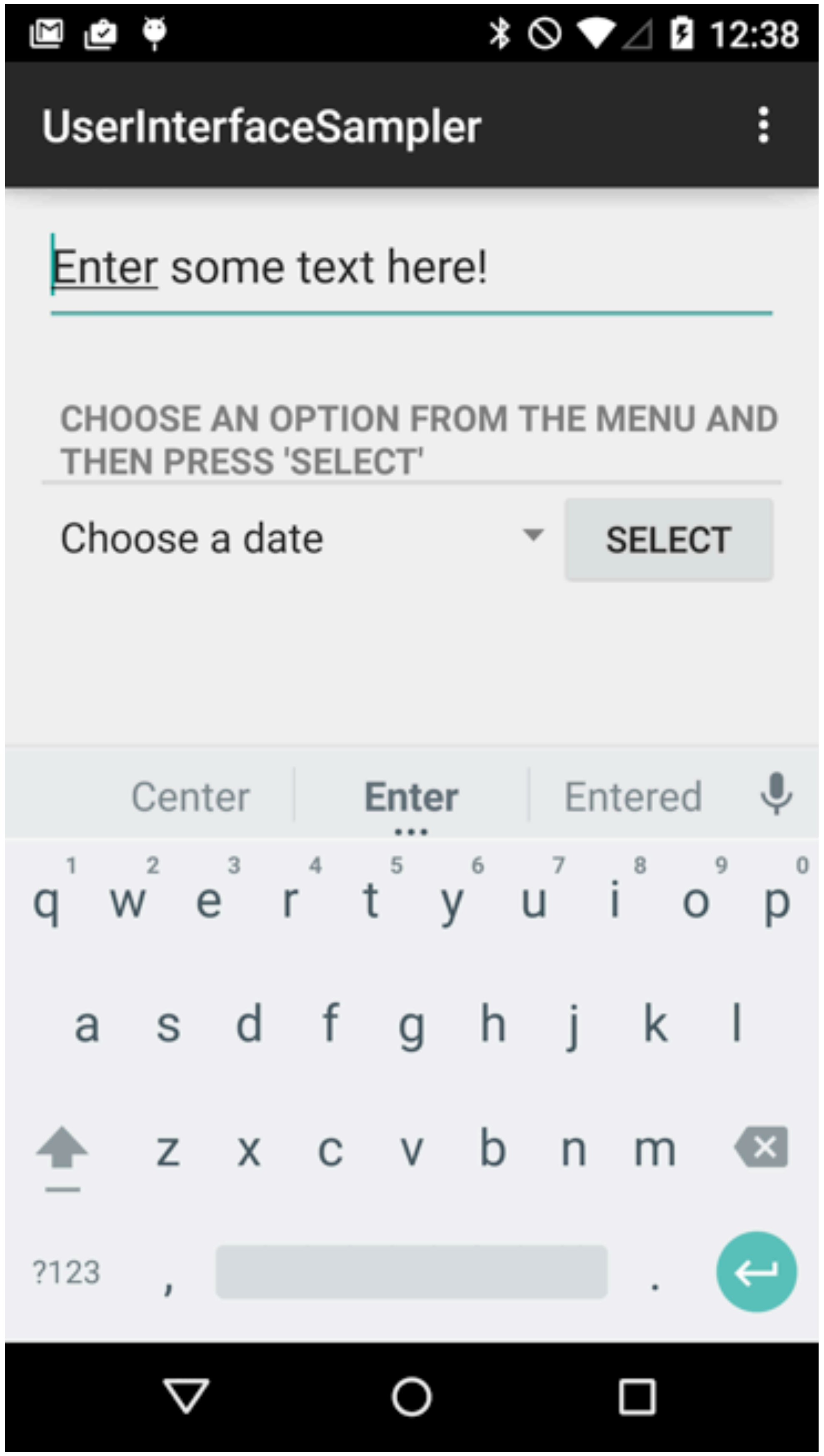
CS 646 Android Mobile Application Development
Spring Semester, 2015
Doc 15 Assignment 2, Animation, Sensors
Mar 25, 2015

Copyright ©, All rights reserved. 2015 SDSU & Roger Whitney, 5500 Campanile Drive, San Diego, CA 92182-7700 USA. OpenContent (http://www.opencontent.org/openpub/) license defines the copyright on this document.

Assignment 2 Comments


12:38

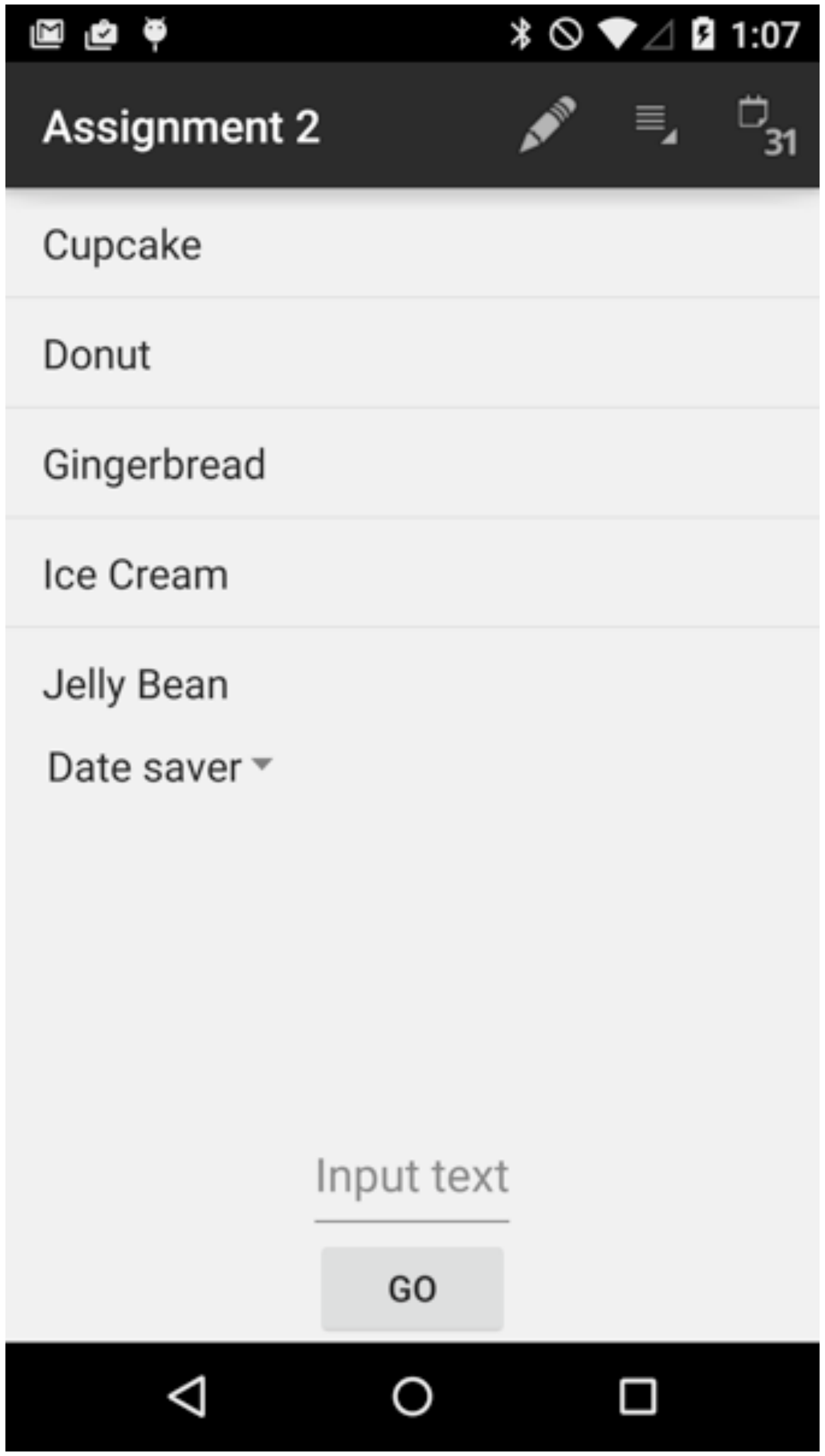
UserInterfaceSampler

Enter some text here!

CHOOSE AN OPTION FROM THE MENU AND THEN PRESS 'SELECT'

Choose a date

SELECT


1:07

Assignment 2

Cupcake

Donut

Gingerbread

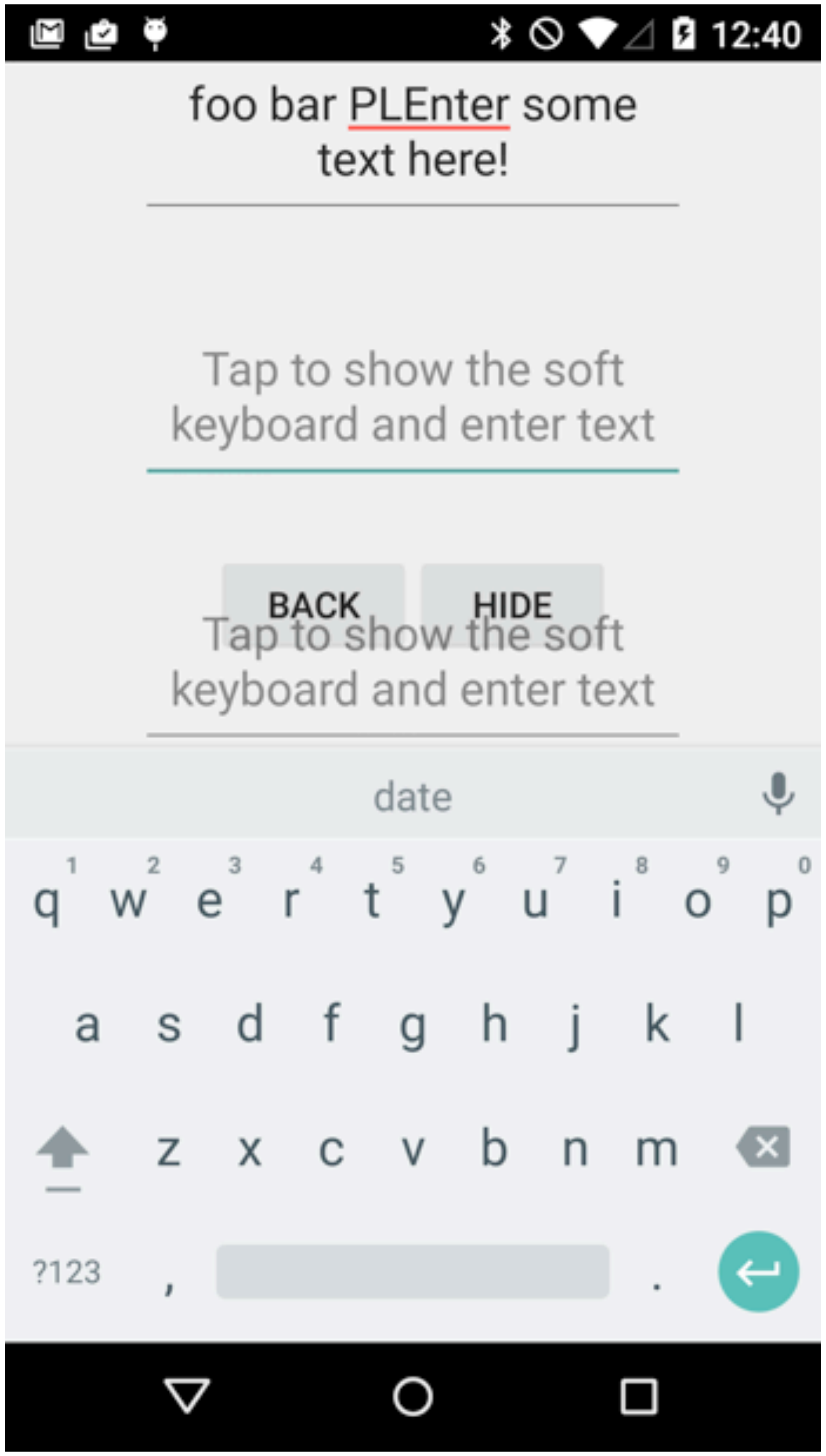
Ice Cream

Jelly Bean

Date saver

Input text

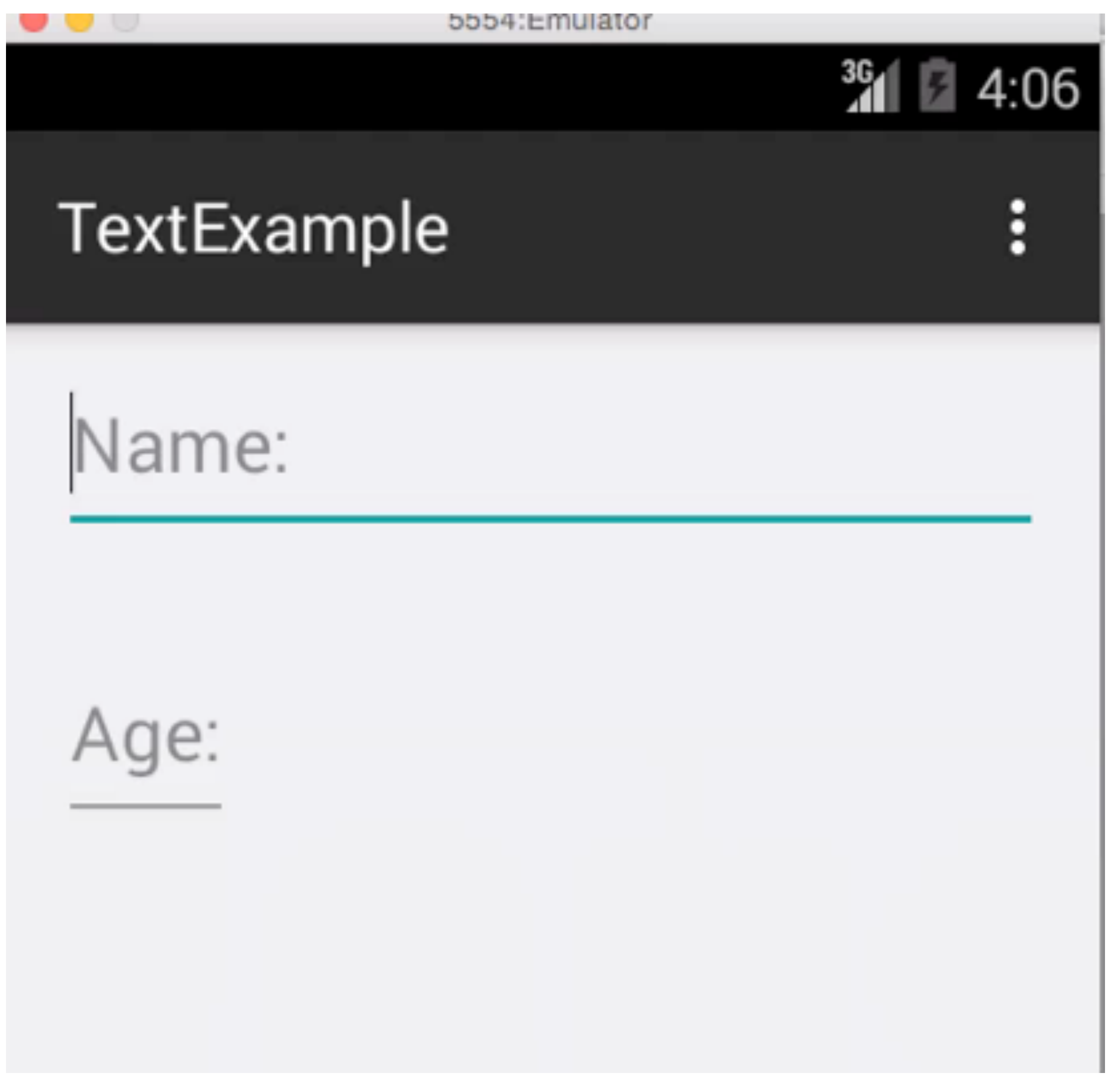
GO

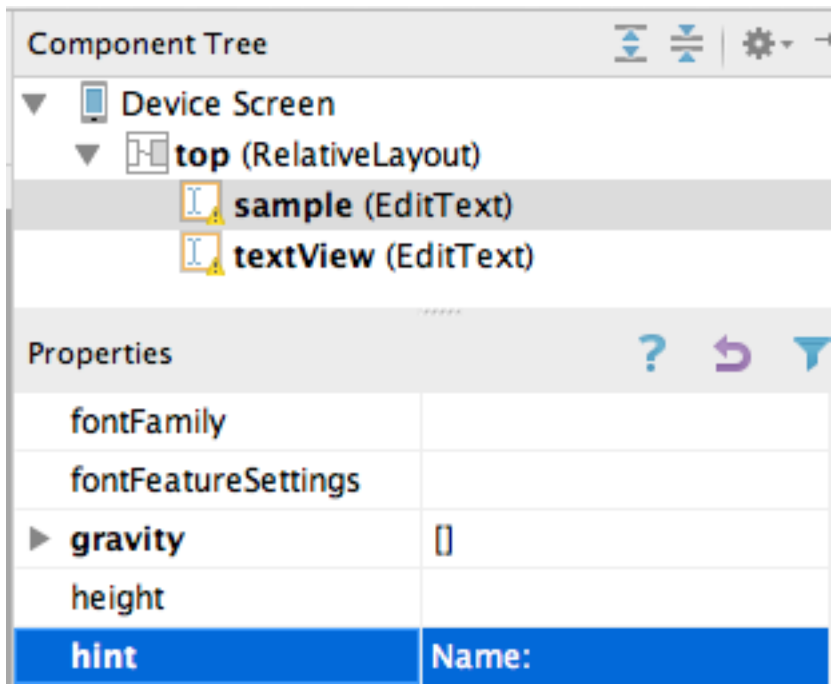

foo bar PLEnter some text here!

Tap to show the soft keyboard and enter text

Tap to show the soft keyboard and enter text

Use Text Hints




Component Tree

- Device Screen
  - **top** (RelativeLayout)
    - **sample** (EditText)
    - **textView** (EditText)

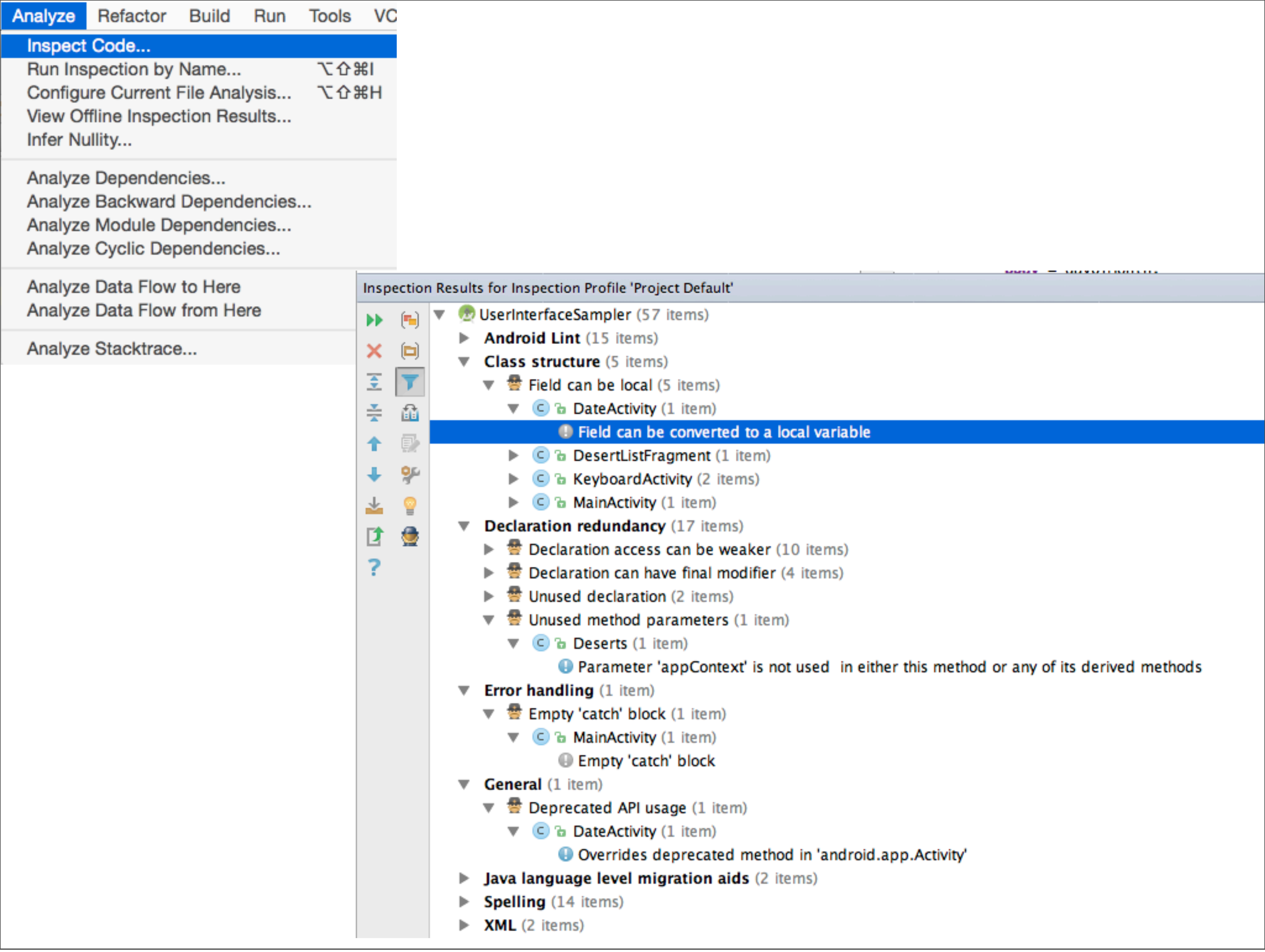
Properties

fontFamily	
fontFeatureSettings	
gravity	[]
height	
hint	Name:

```
public class MainActivity extends ActionBarActivity {

    public static final String PREFS_NAME = "MyPrefsFile";

    private ListAdapter mAdapter;

    private String[] data = {"Cupcake", "Donut", "Gingerbread", "Ice Cream", "Jelly Bean"};

    private Spinner spinner;

    private EditText editText;

    private String selectedItemPosition = "0";

    private ListView listView;
```

```
public class DessertFragment extends Fragment implements
AbsListView.OnItemClickListener
{

    private ArrayList<String> mDesserts;
    private static final String TAG = "DessertFragment";
    private String   mSelectedItem;

    // TODO: Rename parameter arguments, choose names that match
    // the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
    private static final String ARG_PARAM1 = "param1";
    private static final String ARG_PARAM2 = "param2";

    // TODO: Rename and change types of parameters
    private String mParam1;
    private String mParam2;
private void setUpBackButton()
{
    mBtnBack = (Button)findViewById(R.id.btn_list_back);
    mBtnBack.setOnClickListener(new View.OnClickListener()
    {
        public void onClick(View v)
        {
           // Log.d(MainActivity.TAG, "ListActivity onClick Back 3, mPreviousSelectedItem: " + mPreviousSelectedItem);
            Log.d(MainActivity.TAG, "ListActivity onClick Back 4, mSelectedItem: " + mSelectedItem +"\n");
            Intent intent = getIntent();
         //   Bundle extras = new Bundle();
        //    extras.putString(EXTRA_ITEM_SELECTED, mSelectedItem);
        //    extras.putString(EXTRA_PREVIOUS_ITEM_SELECTED, mPreviousSelectedItem);
            //intent.putExtras(extras);

            intent.putExtra(EXTRA_ITEM_SELECTED, mSelectedItem);
            setResult(RESULT_OK, intent);
            finish();
        }
    });
}
```

```
public void showSoftKeyboard(View view)
{
    if(view.requestFocus())
    {
        Log.d(MainActivity.TAG, "keyboard 3: " + mStringFromMain);

        mImm.toggleSoftInput(InputMethodManager.SHOW_FORCED, 0);
    }
}
```



Analyze
Refactor
Build
Run
Tools
VC
Inspect Code...
Run Inspection by Name... ⌥⇧⌘I
Configure Current File Analysis... ⌥⇧⌘H
View Offline Inspection Results...
Infer Nullity...
Analyze Dependencies...
Analyze Backward Dependencies...
Analyze Module Dependencies...
Analyze Cyclic Dependencies...
Analyze Data Flow to Here
Analyze Data Flow from Here
Analyze Stacktrace...

Inspection Results for Inspection Profile 'Project Default'

```
▼ UserInterfaceSampler (57 items)
  ▶ Android Lint (15 items)
  ▼ Class structure (5 items)
    ▼ Field can be local (5 items)
      ▼ DateActivity (1 item)
          Field can be converted to a local variable
      ▶ DesertListFragment (1 item)
      ▶ KeyboardActivity (2 items)
      ▶ MainActivity (1 item)
  ▼ Declaration redundancy (17 items)
    ▶ Declaration access can be weaker (10 items)
    ▶ Declaration can have final modifier (4 items)
    ▶ Unused declaration (2 items)
    ▼ Unused method parameters (1 item)
      ▼ Deserts (1 item)
          Parameter 'appContext' is not used  in either this method or any of its derived methods
  ▼ Error handling (1 item)
    ▼ Empty 'catch' block (1 item)
      ▼ MainActivity (1 item)
          Empty 'catch' block
  ▼ General (1 item)
    ▼ Deprecated API usage (1 item)
      ▼ DateActivity (1 item)
          Overrides deprecated method in 'android.app.Activity'
  ▶ Java language level migration aids (2 items)
  ▶ Spelling (14 items)
  ▶ XML (2 items)
```

```
public class DesertListFragment extends ListFragment {
    private ArrayList<String> mDeserts;

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        mDeserts = Deserts.get(getActivity()).getDeserts();
        ArrayAdapter<String> adapter =
                new ArrayAdapter<String>(getActivity(),
                        android.R.layout.simple_list_item_1,
                        mDeserts);
        setListAdapter(adapter);
    }

     public void onActivityCreated(Bundle savedInstanceState) {
      super.onActivityCreated(savedInstanceState);
      ListView listView = this.getListView();
    listView.setChoiceMode(ListView.CHOICE_MODE_SINGLE);
    int lightGray = 0xffcccccc;
    ColorDrawable cd = new ColorDrawable(lightGray);
    listView.setSelector(cd);
     }
```

```
public class DesertListFragment extends ListFragment {


    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        ArrayList<String> mDeserts = Deserts.get(getActivity()).getDeserts();
        ArrayAdapter<String> adapter =
                new ArrayAdapter<String>(getActivity(),
                        android.R.layout.simple_list_item_1,
                        mDeserts);
        setListAdapter(adapter);
    }

     public void onActivityCreated(Bundle savedInstanceState) {
      super.onActivityCreated(savedInstanceState);
      ListView listView = this.getListView();
    listView.setChoiceMode(ListView.CHOICE_MODE_SINGLE);
    int lightGray = 0xffcccccc;
    ColorDrawable cd = new ColorDrawable(lightGray);
    listView.setSelector(cd);
     }
```

Empty Exception Handlers

```
public void readFromStorage(){
        File file = MainActivity.this.getFileStreamPath("date file");
        if(file.exists()){
                try{
                        FileInputStream fin = openFileInput("date file");
                        int c;
                        String editTextContents="";
                        while( (c = fin.read()) != -1){
                                editTextContents = editTextContents + Character.toString((char)c);
                        }
                        editText.setText(editTextContents);
                }catch(Exception e){

                }
        }
}
```

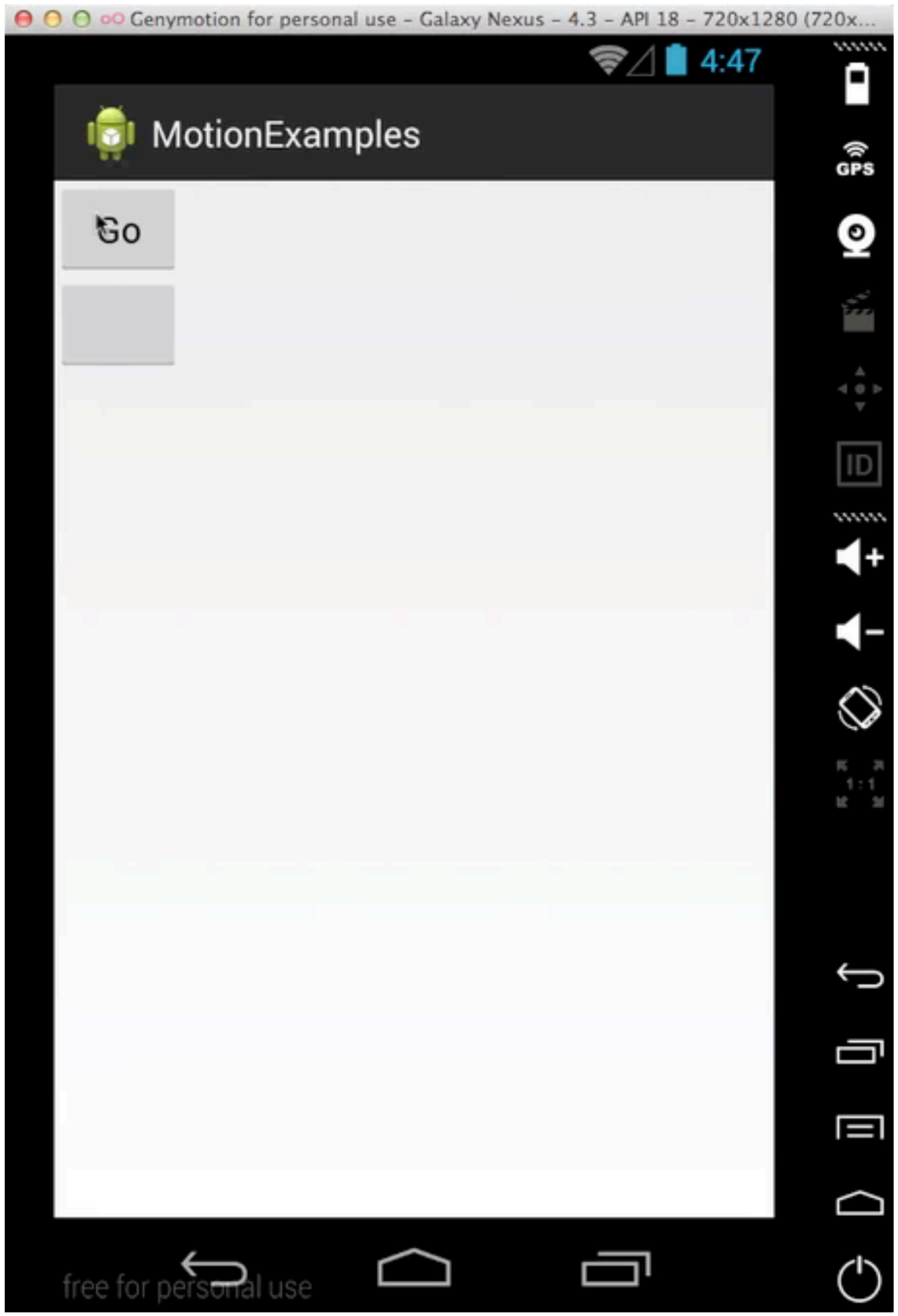
Unused Declarations

```
public class MainActivity extends ActionBarActivity{

    private Button mSelectButton;
    public EditText editText;
    public String editTextContents = "Enter some text here!";
    public Context context;
    String extraText;
    Spinner spinner;
```

context is not used in the class

Animation in Code

Changing Location


Genymotion for personal use - Galaxy Nexus - 4.3 - API 18 - 720x1280 (720x...
4:47
MotionExamples
Go
GPS
ID
free for personal use

Moving The Box

```
public class MainActivity extends Activity {
    View box;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        box = this.findViewById(R.id.box);
    }

    @TargetApi(Build.VERSION_CODES.HONEYCOMB)
    public void start(View button){
        box.setX(box.getX() + 100);
        box.setY(box.getY() + 100);
    }
}
```

Animation

Need to update position continuously

- Threads
  - Timers
  - postDelayed

- Events
  - Touch
  - Motion

Timers

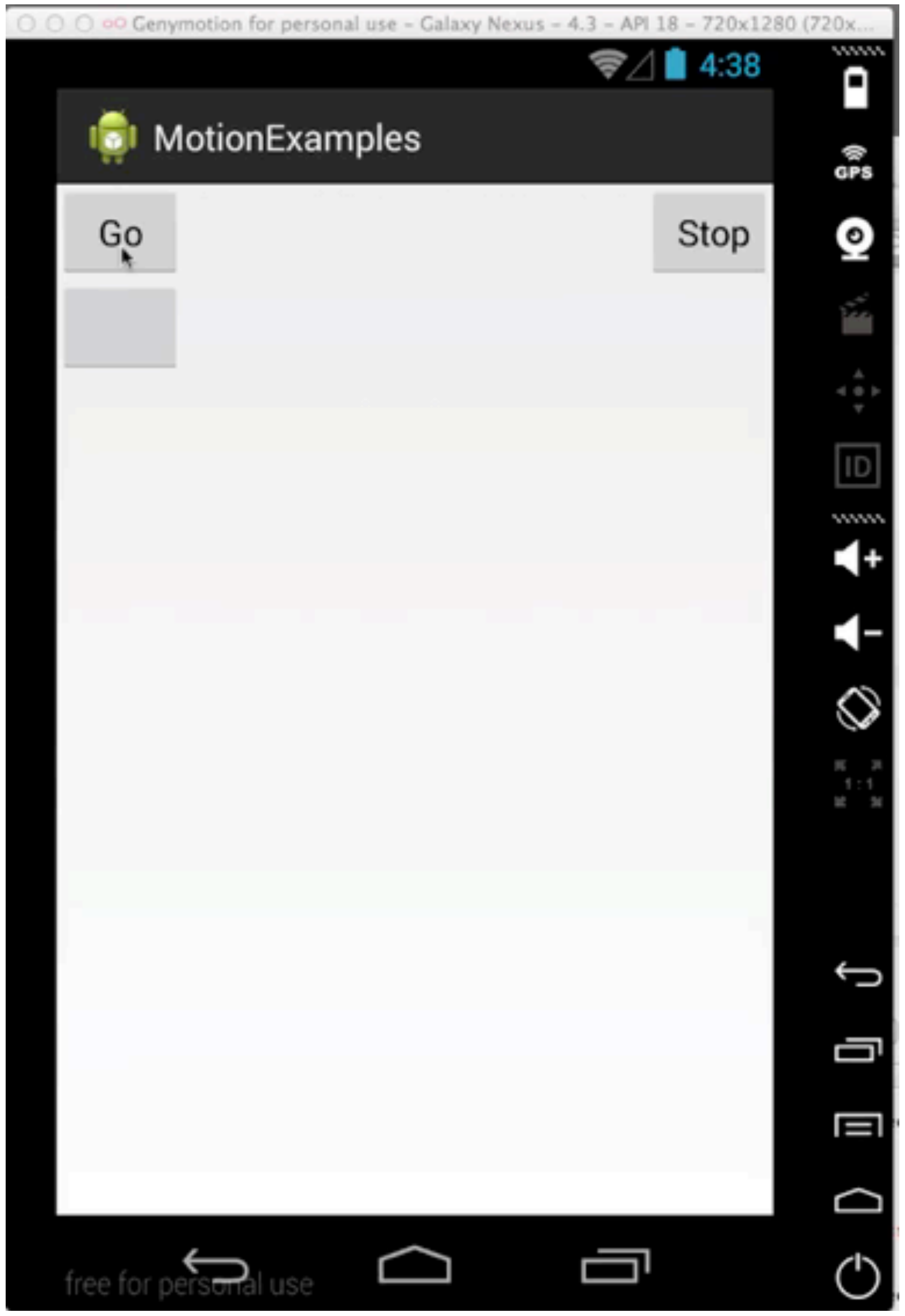
```
It is running 2015-03-26T04:27:15.825Z
It is running 2015-03-26T04:27:16.823Z
It is running 2015-03-26T04:27:17.821Z
It is running 2015-03-26T04:27:18.819Z
It is running 2015-03-26T04:27:19.821Z
It is running 2015-03-26T04:27:20.820Z
It is running 2015-03-26T04:27:21.820Z
It is running 2015-03-26T04:27:22.820Z
```

```
import java.time.Instant;
import java.util.Timer;
import java.util.TimerTask;

public class Main {

    public static void main(String[] args) {
        TimerTask printer = new TimerTask() {
            @Override
            public void run() {
                System.out.println("It is running " + Instant.now());
            }
        };
        Timer everySecond = new Timer();
        everySecond.scheduleAtFixedRate(printer, 1000, 1000);
    }
}
```

Use cancel() to stop a timer

Moving

With use postDelayed


Genymotion for personal use - Galaxy Nexus - 4.3 - API 18 - 720x1280 (720x...
4:38
MotionExamples
Go
Stop
free for personal use

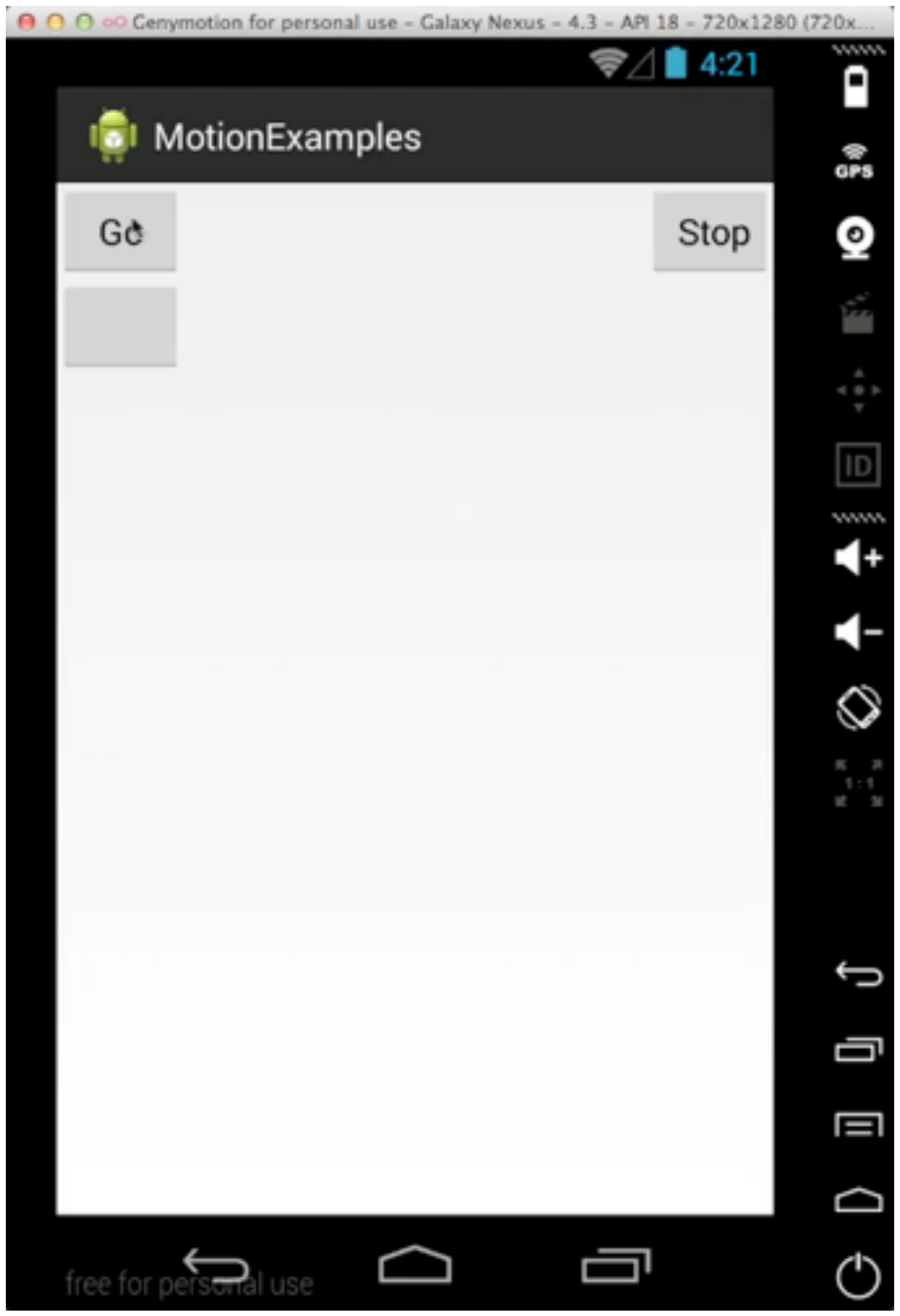
Stop & Start

```
public class MainActivity extends Activity {
        View box;
        boolean isInMotion;
        @Override
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);
                box = this.findViewById(R.id.box);
        }

        public void start(View button){
                isInMotion = true;
                move();
        }

        public void stop(View button){
                isInMotion = false;
        }
```

The Moving Part

```
@TargetApi(Build.VERSION_CODES.HONEYCOMB)
private void move() {
    box.setX(box.getX() + 10);
    box.setY(box.getY() + 10);
    if (isInMotion)
        box.postDelayed(new Mover(), 50);
}

public class Mover implements Runnable {

    @Override
    public void run() {
        move();
    }
}
```

Detecting Edges


Genymotion for personal use - Galaxy Nexus - 4.3 - API 18 - 720x1280 (720x...
4:21
MotionExamples
Go
Stop
free for personal use

Getting the Size of Screen

```
public class MainActivity extends Activity {
    View box;
    boolean isInMotion;
    int screenWidth;
    int screenHeight;
    int deltaX = 10;
    int deltaY = 10;

    @TargetApi(Build.VERSION_CODES.HONEYCOMB_MR2)
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        box = this.findViewById(R.id.box);
        Display display = getWindowManager().getDefaultDisplay();
        Point screenSize = new Point();
        display.getSize(screenSize);
        screenHeight = screenSize.y;
        screenWidth = screenSize.x;
    }
```

Starting, Stopping

```
public void start(View button){
    isInMotion = true;
    move();
}


public void stop(View button){
    isInMotion = false;
}

@SuppressLint("NewApi")
private void move() {
    box.setX(box.getX() + deltaX);
    box.setY(box.getY() + deltaY);
    changeOnCollison();
    if (isInMotion)
        box.postDelayed(new Mover(), 50);
}
```

Detecting the Edges

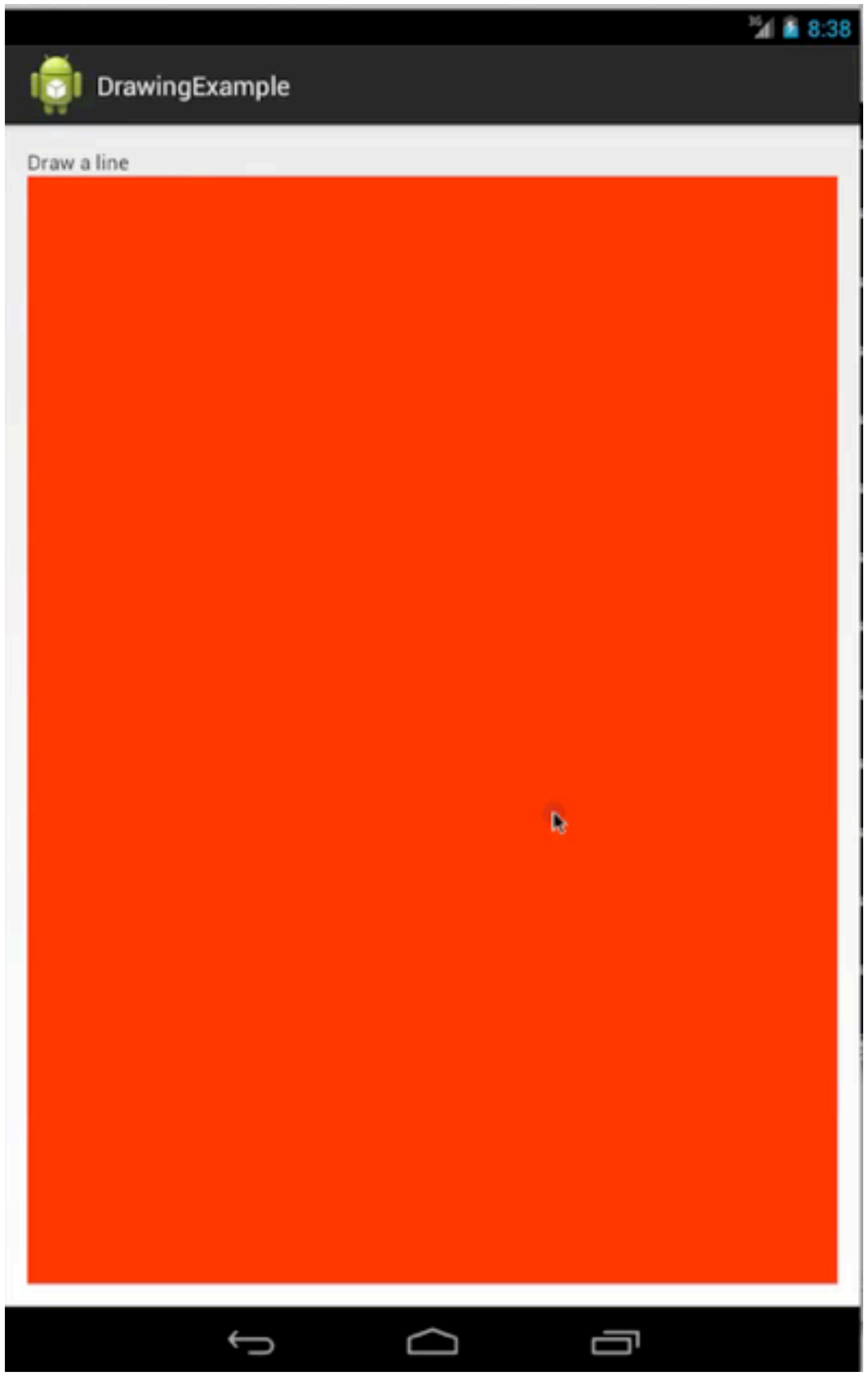
```
private void changeOnCollison() {
    if (xIsOutOfBounds(box))  deltaX = deltaX * -1;
    if (yIsOutOfBounds(box))  deltaY = deltaY * -1;
}

@TargetApi(Build.VERSION_CODES.HONEYCOMB)
private boolean xIsOutOfBounds(View widget) {
    float x = widget.getX();
    if (x <0) return true;
    if (x + widget.getWidth() > screenWidth) return true;
    return false;
}

@TargetApi(Build.VERSION_CODES.HONEYCOMB)
private boolean yIsOutOfBounds(View widget) {
    float y = widget.getY();
    if (y <0) return true;
    if (y + widget.getHeight() + 150> screenHeight) return true;
    return false;
}
```

The Mover

```
public class Mover implements Runnable {

    @Override
    public void run() {
        move();
    }
}
```

Touch Events

Combining Touch and Drawing



Entire Activity

```
public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

SimpleDrawing

```
public class SimpleDrawing extends View implements OnTouchListener {
    static Paint black;

    static {
        black = new Paint();
        black.setColor(Color.BLACK);
        black.setStrokeWidth(12.0f);
    }

    private float startX;
    private float startY;
    private float currentX;
    private float currentY;

    public SimpleDrawing(Context context, AttributeSet xmlAttributes) {
        super(context, xmlAttributes);
        setOnTouchListener(this);
    }
```

Getting the Touch Event

```
public boolean onTouch(View arg0, MotionEvent event) {
    int action = event.getAction();
    int actionCode = action & MotionEvent.ACTION_MASK;
    switch (actionCode) {
        case MotionEvent.ACTION_DOWN:
            return handleActionDown(event);
        case MotionEvent.ACTION_MOVE:
            return handleActionMove(event);
    }
    return false;
}
```

Handing Events and Drawing

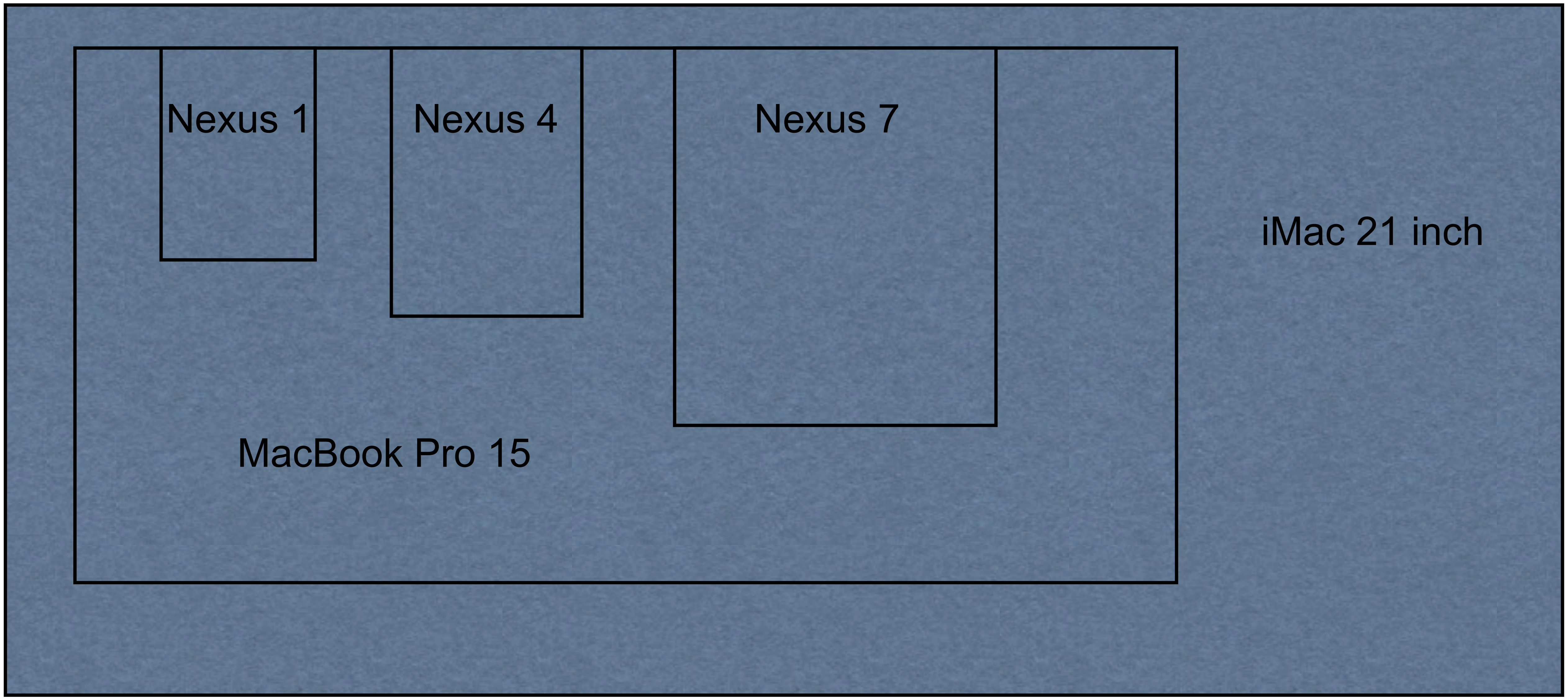
```
private boolean handleActionMove(MotionEvent event) {
    startX = event.getX();
    startY = event.getY();
    invalidate();
    return true;
}

private boolean handleActionDown(MotionEvent event) {
    currentX = event.getX();
    currentY = event.getY();
    return true;
}

protected void onDraw(Canvas canvas) {
    canvas.drawColor(Color.RED);
    canvas.drawLine(startX, startY, currentX, currentY, black);
}
```

Sensors

Difference Between Mobile & Desktop

	Mobile	DeskTop/ Laptop
Size	Pocket	
Power	Battery	Power Grid/Battery
Input	Finger on touch screen	Mouse
Peripherals	Cell radio, Accelerometer, Gyroscope, Proximity	Hard drive, CD/DVD

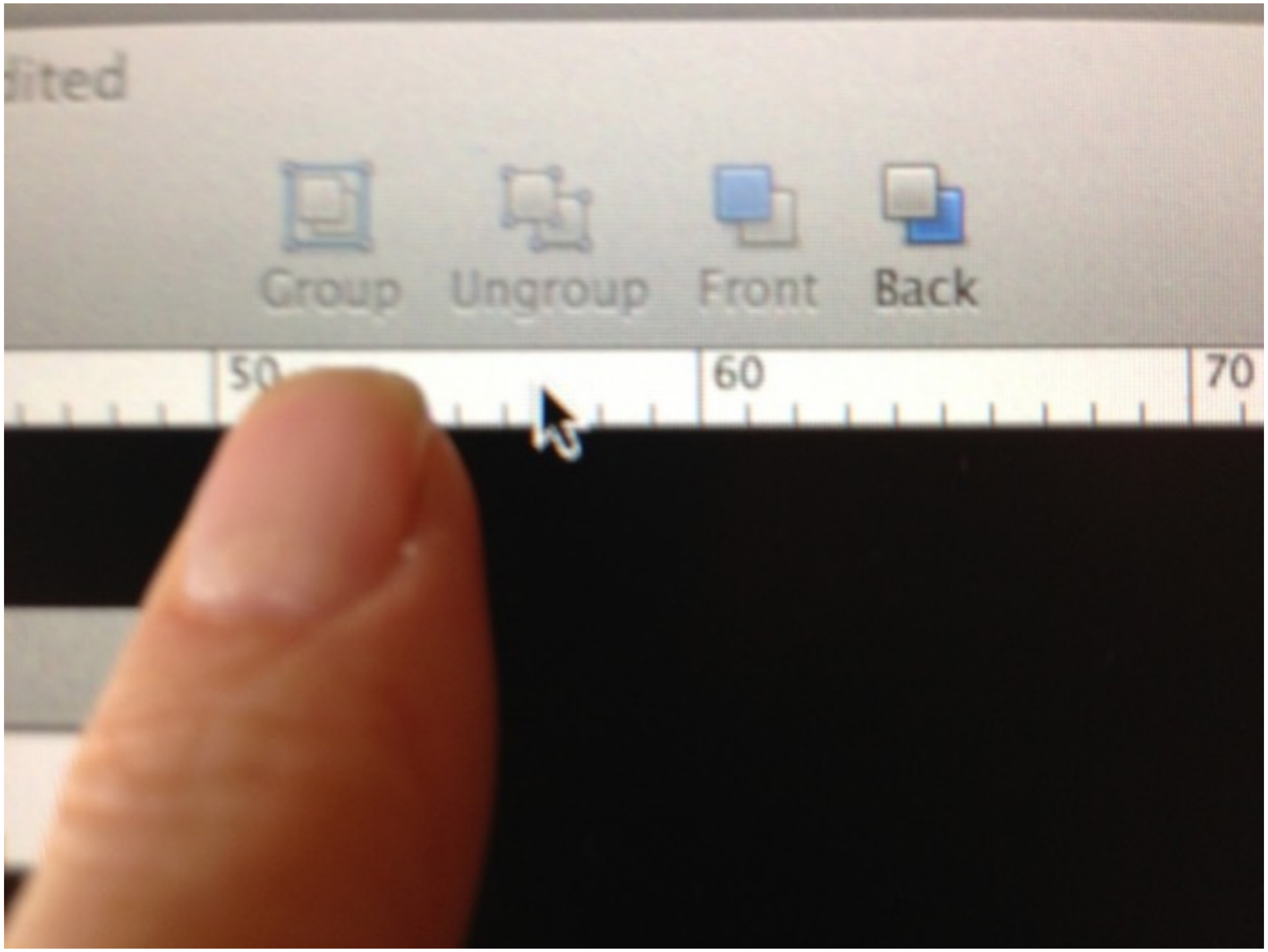
Size



Power

Device	Battery
Nexus One	1,400 mAh
Nexus 4	2,100 mAh
Nexus 7	3,425 mAh
MacBook Pro	7,000 mAh

Input



Sensors

Potential Sensor Types

TYPE_ACCELEROMETER
TYPE_AMBIENT_TEMPERATURE
TYPE_GRAVITY
TYPE_GYROSCOPE
TYPE_LIGHT
TYPE_LINEAR_ACCELERATION
TYPE_MAGNETIC_FIELD
TYPE_ORIENTATION
TYPE_PRESSURE
TYPE_PROXIMITY
TYPE_RELATIVE_HUMIDITY
TYPE_ROTATION_VECTOR
TYPE_TEMPERATURE

Common

Proximity

Accelerometer

Gyroscope

Magnetic Field

Light

Issues

How to tell if a sensor is on a device

How to list all sensors

Properties of sensors

- Power consumption
- Range
- Resolution
- Min delay

Does a Sensor exist on the Device

```
private SensorManager mSensorManager;
...
mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
if (mSensorManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD) != null){
  // Success! There's a magnetometer.
  }
else {
  // Failure! No magnetometer.
  }
```

Requiring a Sensor

If app requires a sensor
Add uses-feature to manifest file
Uses will only see app if their device has the sensor

```
<uses-feature android:name="android.hardware.sensor.accelerometer"
              android:required="true" />
```

Listing all Sensors

```
        SensorManager mSensorManager;
        mSensorManager = (SensorManager)
getSystemService(Context.SENSOR_SERVICE);
        List<Sensor> deviceSensors = mSensorManager.getSensorList(Sensor.TYPE_ALL);
```

Sensor Properties

Sensor Methods

getMaximumRange()
getMinDelay()
getPower()
getResolution()
getType()
getVendor()
getVersion()

Properties of the Accelerometer

```
public class MainActivity extends Activity {

	@TargetApi(Build.VERSION_CODES.GINGERBREAD)
	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.activity_main);
		SensorManager mSensorManager;
		mSensorManager = (SensorManager)
getSystemService(Context.SENSOR_SERVICE);
		Sensor accelerometer = mSensorManager
				.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);

		Log.i("rew", " range " + accelerometer.getMaximumRange());
		Log.i("rew", " resolution " + accelerometer.getResolution());
		Log.i("rew", " power " + accelerometer.getPower());
		Log.i("rew", " delay " + accelerometer.getMinDelay());
	}

}
```

Values

Power	0.0
Range	19.6
Resolution	0.039
Min Delay	5000

Min Delay

Minimum time in milliseconds it take sensor to sense data

Getting Sensor Data

Get reference to sensor

Attach SensorEventListener

Sensor periodically sends updated data fo listener

Unregister SensorEventListener in onPause()

SensorEventListener

```
public void onAccuracyChanged(Sensor sensor, int accuracy){
}

public void onSensorChanged(SensorEvent event) {
}
```

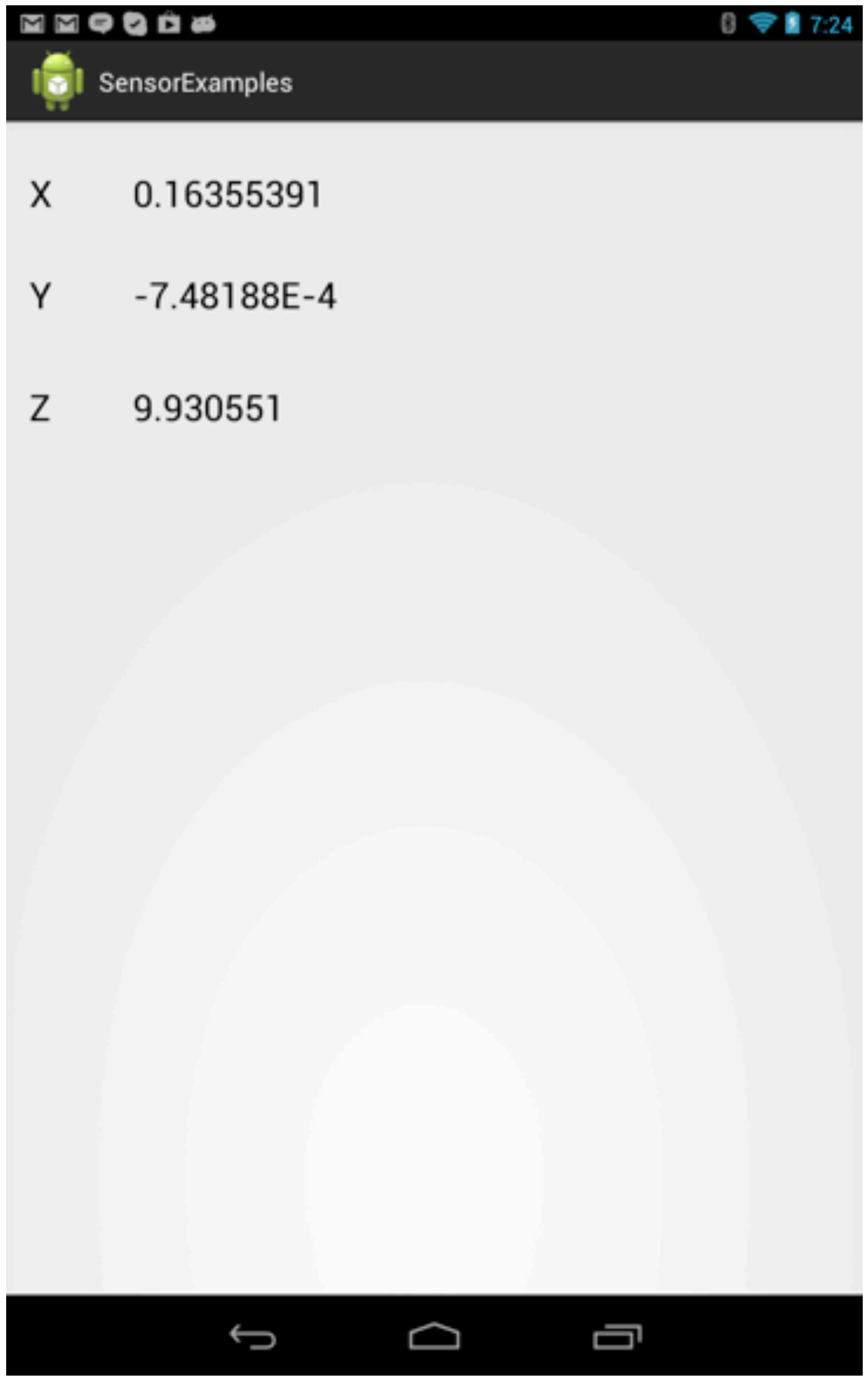
Sensor sends current values to listener periodically

SENSOR_DELAY_FASTEST	0 microsecond delay
SENSOR_DELAY_GAME	20,000
SENSOR_DELAY_UI	60,000
SENSOR_DELAY_NORMAL	200,000

These are hints to the OS

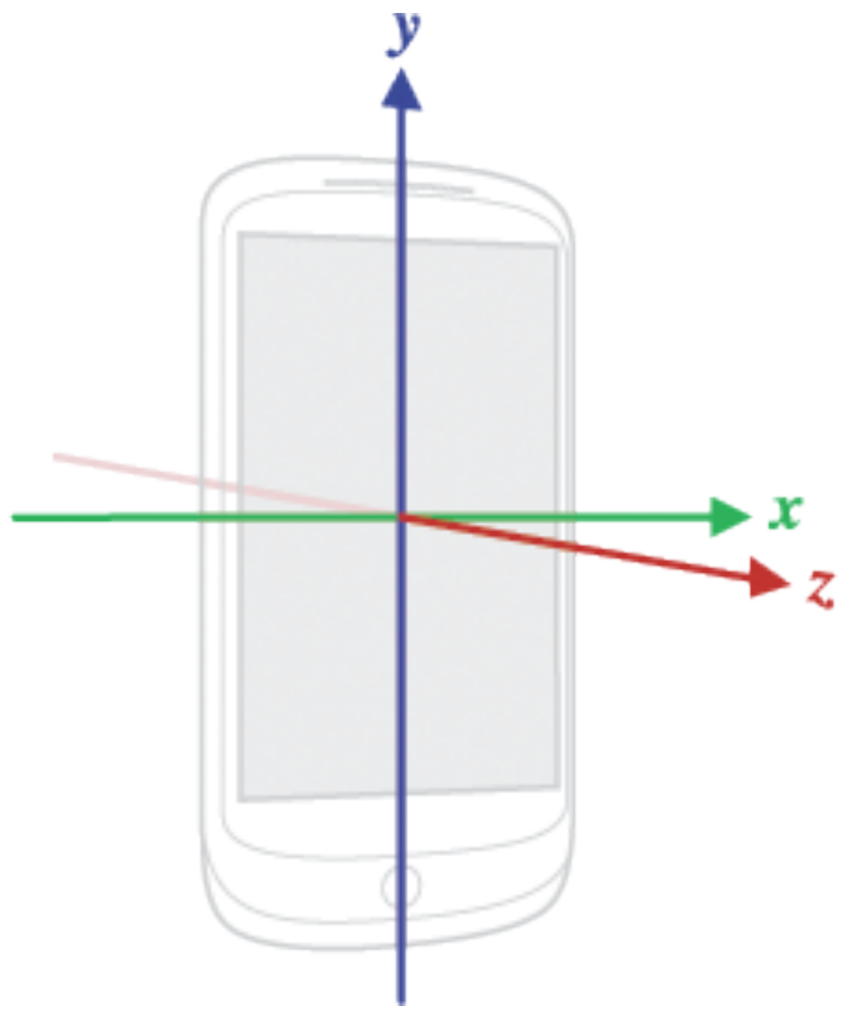
Events tend to be delivered faster

Accelerometer Example



```
X     0.16355391

Y     -7.48188E-4

Z     9.930551
```

Sensor Coordinate System



Acceleration sensor
Gravity sensor
Gyroscope
Linear acceleration sensor
Geomagnetic field sensor

Accelerometer

Measures acceleration as meters/(second^2)

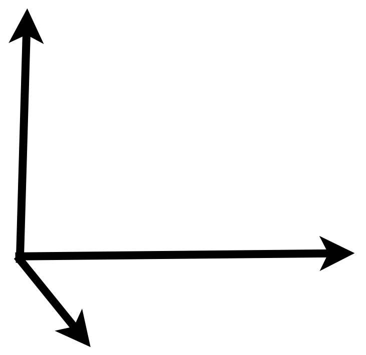
Gravity is -9.81 m/s^2

Not highly accurate

  Can not compute users velocity/distance traveled using accelerometer

Uses 10 time less energy than other sensors

Measures acceleration in 3 dimensions: X, Y, Z



Phone Sitting flat on desk, screen up

X: 0.027240695 Y: -0.24516626 Z: 9.765789

Activity

```
public class MainActivity extends Activity implements SensorEventListener {
    private SensorManager mSensorManager;
    private Sensor accelerometer;
    private TextView x;
    private TextView y;
    private TextView z;

    @TargetApi(Build.VERSION_CODES.GINGERBREAD)
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        x = (TextView) findViewById(R.id.x);
        y = (TextView) findViewById(R.id.y);
        z = (TextView) findViewById(R.id.z);
        mSensorManager = (SensorManager)
getSystemService(Context.SENSOR_SERVICE);
        accelerometer = mSensorManager
                .getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
    }
```

Don't forget onPause

```
protected void onResume() {
    super.onResume();
    boolean isRunning = mSensorManager.registerListener(this, accelerometer,
            SensorManager.SENSOR_DELAY_NORMAL);
    if (!isRunning)
        Log.i("rew", "could not start accelerometer");
}

protected void onPause() {
    super.onPause();
    mSensorManager.unregisterListener(this);
}
```

Handling the Events

```
@Override
public void onAccuracyChanged(Sensor arg0, int arg1) {
    Log.i("rew", "Sensor accuracy changed");
}

@Override
public void onSensorChanged(SensorEvent event) {
    x.setText(String.valueOf(event.values[0]));
    y.setText(String.valueOf(event.values[1]));
    z.setText(String.valueOf(event.values[2]));
}
```

Gravity and Linear Accelerometer

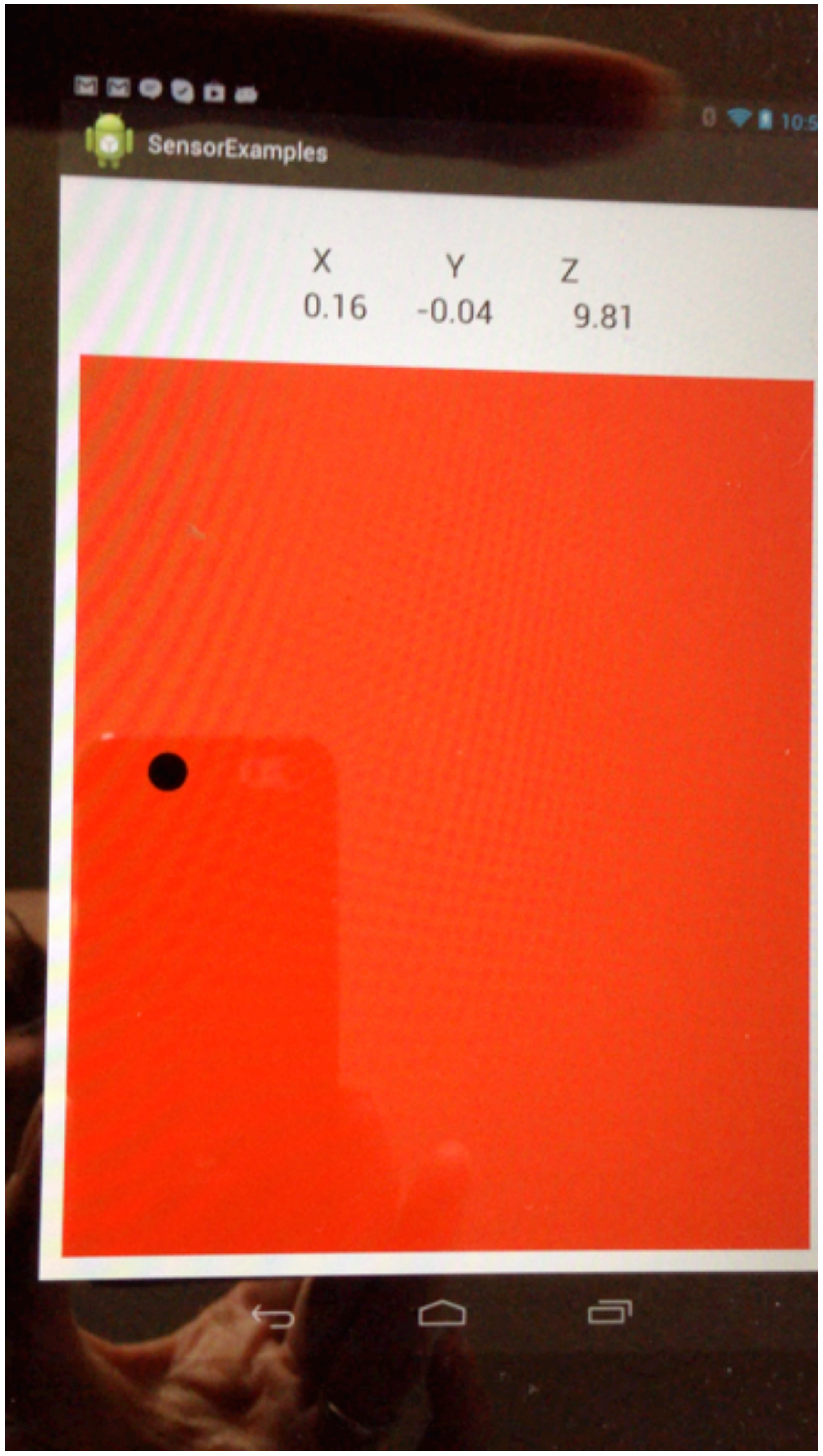
Accelerometer

Linear Accelerometer

Gravity Sensor

Linear Accelerometer = Accelerometer - Gravity Sensor

Games Using Tilt



Some Physics

x = x location

x_i = initial location

v = velocity

v_i - initial velocity

a = acceleration

dt = time delta

Assuming acceleration is constant

$$v = v_i + a * dt$$

$$x = x_i + v * dt$$

Some Physics

x_k = location at time step k
v_k = velocity at time step k
a_k = acceleration at time step k
dt = time delta

If dt is very small

Then acceleration does not change much in dt

Assume acceleration is constant in dt

$$v_k = v_{k-1} + a_k * dt$$

$$x_k = x_{k-1} + v_k * dt$$

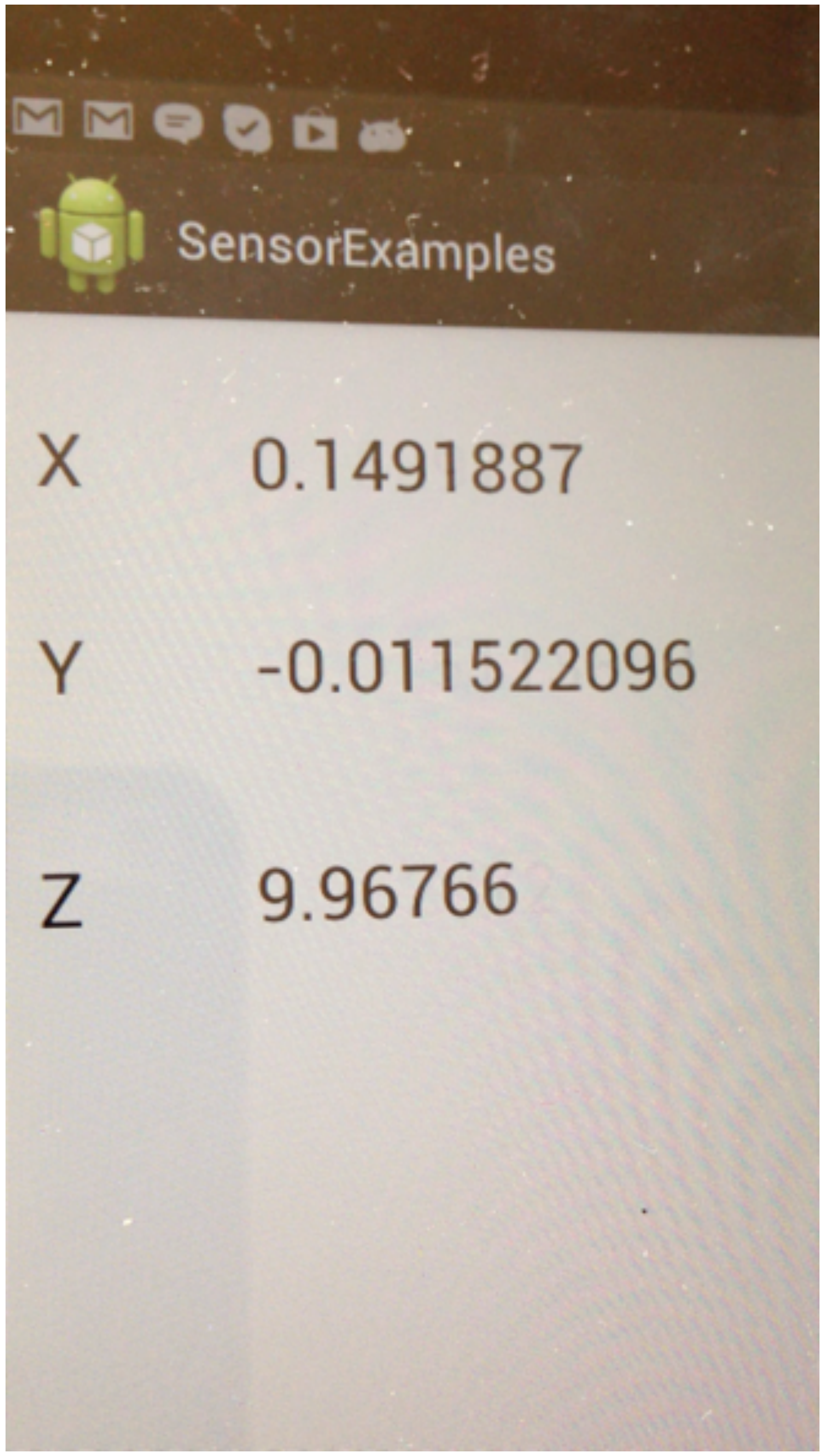
Issues

Noise in accelerometer data

Calculation errors
- Round off errors
- Varying time steps
- Numerical analysis is handy

Realism
- Acceleration in meters/second^2
- Distance in screen in pixels
- Friction

Edge Detection



```
X    0.1491887
Y    -0.011522096
Z    9.96766
```

The Better Way

Accelerometer Play example in Android SDK examples

But first read

A Simple Time-Corrected Verlet Integration Method

http://www.gamedev.net/page/resources/_/technical/math-and-physics/a-simple-time-corrected-verlet-integration-method-r2200

But

That is a lot of math

If doing it in a game you will want your game engine to do this

We will simplify

Issue - Noise in Data

Round to two decimals

```
private float round(float value) {
      return Math.round(value*100)/100.0f;
}
```

Classes

Ball
BallView
MainActivity

Ball

```
public class Ball {
static Paint black;

    static {
        black = new Paint();
        black.setColor(Color.BLACK);
    }

    double x = 0;
    double y = 0;
    double maxX = 100;
    double maxY = 100;
    double xVelocity = 0;
    double yVelocity = 0;
    double radius = 20;
```

Compute next Location

$v_k = v_{k-1} + a_k * dt$

$x_k = x_{k-1} + v_k * dt$

```
public void accelerate(float xAcceleration, float yAcceleration,
    double timeDeltaSeconds){
       xVelocity = xVelocity - 5*xAcceleration*timeDeltaSeconds;
       yVelocity = yVelocity + 5*yAcceleration*timeDeltaSeconds;
       x = x + 2*xVelocity* timeDeltaSeconds;
       y = y + 2*yVelocity* timeDeltaSeconds;
       frictionSlowDown();
       bounceIfHitEdge();
}
```

5 & 2 are fudge factors

Should use physics here

Friction and Edge detection

```
private void frictionSlowDown() {
      xVelocity = xVelocity*0.995;
      yVelocity = yVelocity*0.995;
      if (Math.abs(xVelocity) < 0.01) xVelocity = 0;
      if (Math.abs(yVelocity) < 0.01) yVelocity = 0;
}

private void bounceIfHitEdge() {
      if (Math.abs(x) > maxX - radius)
            xVelocity = -1 * xVelocity;
      if (Math.abs(y) > maxY - radius)
            yVelocity = -1 * yVelocity;
}
```

More fudge factors instead of physics

BallView

```
public class BallView extends View implements SensorEventListener{

    Ball blackBall = new Ball();
    long lastUpdateTime = 0;

    public BallsView(Context context, AttributeSet attrs) {
        super(context, attrs);
    }

    public void onDraw(Canvas canvas) {
        canvas.drawColor(Color.RED);
        centerOrigin(canvas);
        blackBall.drawOn(canvas);
    }

    private void centerOrigin(Canvas canvas) {
        int width = canvas.getWidth();
        int height = canvas.getHeight();
        canvas.translate(width/2, height/2);
    }
```

Sensor Changed

```
public void onSensorChanged(SensorEvent event) {
	if (lastUpdateTime == 0) {
		lastUpdateTime = event.timestamp;
		return;
	}
	long timeDelta = event.timestamp - lastUpdateTime;
	lastUpdateTime = event.timestamp;
	float xAcceleration = round(event.values[0]);
	float yAcceleration = round(event.values[1]);
	blackBall.accelerate(xAcceleration, yAcceleration, timeDelta/1000000000.0f);
	invalidate();
}

private float round(float value) {
	return Math.round(value*100)/100.0f;
}
```

onAccuracyChanged

```
@Override
public void onAccuracyChanged(Sensor sensor, int accuracy) {
      // only happens with change orientation

}
```

Main Activity

```
public class MainActivity extends Activity implements SensorEventListener {
    private SensorManager mSensorManager;
    private Sensor accelerometer;
    private TextView x;
    private TextView y;
    private TextView z;
    private BallView ballView;
```

onCreate

```
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        x = (TextView) findViewById(R.id.x);
        y = (TextView) findViewById(R.id.y);
        z = (TextView) findViewById(R.id.z);
        ballView = (BallView) findViewById(R.id.ballView);

        mSensorManager = (SensorManager)
getSystemService(Context.SENSOR_SERVICE);
        accelerometer = mSensorManager
                .getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
}
```

Two different listeners

```
protected void onResume() {
    super.onResume();
    boolean isRunning = mSensorManager.registerListener(this, accelerometer,
            SensorManager.SENSOR_DELAY_NORMAL);
    if (!isRunning)
        Log.i("rew", "could not start accelerometer");
    mSensorManager.registerListener(ballView, accelerometer,
            SensorManager.SENSOR_DELAY_NORMAL);
}

protected void onPause() {
    super.onPause();
    mSensorManager.unregisterListener(this);
}
```

Updating Value of accelerometer

```
public void onAccuracyChanged(Sensor arg0, int arg1) {
        Log.i("rew", "Sensor accuracy changed");
    }

    @Override
    public void onSensorChanged(SensorEvent event) {
        x.setText(String.valueOf(round(event.values[0])));
        y.setText(String.valueOf(round(event.values[1])));
        z.setText(String.valueOf(round(event.values[2])));
    }

    private float round(float value) {
        return Math.round(value*100)/100.0f;
    }
```