ANDROID APPS DEVELOPMENT FOR MOBILE AND TABLET DEVICE (LEVEL II)

Lecture 5: Sensor and Game Development

Peter Lo

Sensor Overview

- Most Android-powered devices have built-in sensors that measure motion, orientation, and various environmental conditions.
- These sensors are capable of providing raw data with high precision and accuracy, and are useful if you want to monitor three-dimensional device movement or positioning, or you want to monitor changes in the ambient environment near a device.

Categories of Sensors

- Android platform supports three broad categories of sensors:
  - Environmental Sensors
    - These sensors measure various environmental parameters, such as ambient air temperature and pressure, illumination, and humidity (barometers, photometers, and thermometers)
  - Motion Sensors
    - These sensors measure acceleration forces and rotational forces along three axes (Accelerometers, gravity sensors, gyroscopes, and rotational vector sensors)
  - Position Sensors
    - These sensors measure the physical position of a device. (Orientation sensors and magnetometers)

Environment Sensor

- Android provides four sensors that let you monitor various environmental properties.

Sensor Type	Unit	Sensor Event Data	Data Description
TYPE_AMBIENT_TEMPERATURE	°C	SensorEvent.values[0]	Ambient air temperature
TYPE_LIGHT	lx	SensorEvent.values[0]	Illuminance
TYPE_PRESSURE	hPa / mbar	SensorEvent.values[0]	Ambient air pressure
TYPE_RELATIVE_HUMIDITY	%	SensorEvent.values[0]	Ambient relative humidity

These sensors are hardware-based and are available only if a device manufacturer has built them into a device.

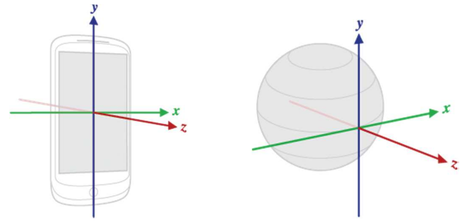
Position Sensor

- Android provides the geomagnetic field sensor and the orientation sensor that let you determine the position of a device. It also provide a proximity sensor that lets you determine how close the device to an object.

Sensor Type	Unit	Sensor Event Data	Data Description
TYPE_PROXIMITY	cm	SensorEvent.values[0]	Distance from object
TYPE_MAGNETIC_FIELD	μT	SensorEvent.values[0]	Geomagnetic field strength along the x axis
		SensorEvent.values[1]	Geomagnetic field strength along the y axis
		SensorEvent.values[2]	Geomagnetic field strength along the z axis

Motion Sensor

- Android provides several sensors that let you monitor the motion of a device.
- Accelerometer and gyroscope sensors are always hardware-based.
- Gravity, linear acceleration and rotation vector sensors can be either hardware-based or software-based .

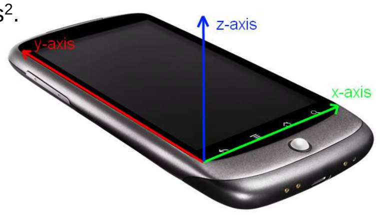


Motion Sensors Type

Sensor Type	Unit	Sensor Event Data	Data Description
TYPE_ACCELEROMETER	m/s^2	SensorEvent.values[0-2]	Acceleration force along the x, y, z axis
TYPE_GRAVITY	m/s^2	SensorEvent.values[0-2]	Force of gravity along the x, y, z axis.
TYPE_GYROSCOPE	rad/s	SensorEvent.values[0-2]	Rate of rotation around the x, y, z axis.
TYPE_LINEAR_ACCELERATION	m/s^2	SensorEvent.values[0-2]	Acceleration force along the x, y, z axis (excluding gravity).
TYPE_ROTATION_VECTOR		SensorEvent.values[0-3]	Rotation vector component along the x, y, z axis and rotation vector.

Acceleration Sensor

- An acceleration sensor measures the acceleration applied to the device, including the force of gravity.
- The force of gravity is always influencing the measured acceleration according to the following relationship:
  - $A = - g - \sum F / mass$
- To measure the real acceleration of the device, the force of gravity must be removed from the accelerometer data because:
  - When the device is sitting on a table, $g = 9.81\ m/s^2$.
  - When the device is in free fall and therefore rapidly accelerating toward the ground at $9.81\ m/s^2$, $g = 0\ m/s^2$.



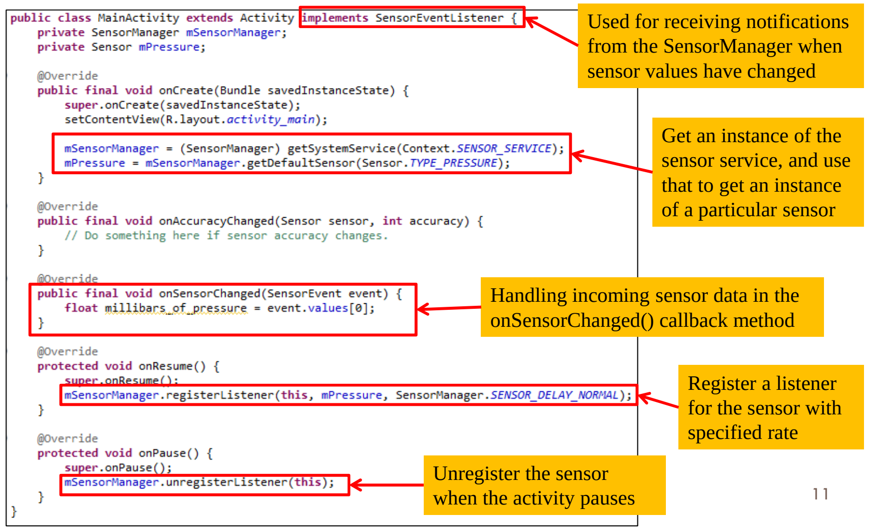
Rotation Vector Sensor

- Accelerometers use the standard sensor coordinate system. In practice, this means that the following conditions apply when a device is laying flat on a table in its natural orientation:
  - If you push the device on the left side (so it moves to the right), the x acceleration value is positive.
  - If you push the device on the bottom (so it moves away from you), the y acceleration value is positive.
  - If you push the device toward the sky with an acceleration of A m/s^2, the z acceleration value is equal to A + 9.81, which corresponds to the acceleration of the device (+A m/s^2) minus the force of gravity (-9.81 m/s^2).
  - The stationary device will have an acceleration value of +9.81, which corresponds to the acceleration of the device (0 m/s^2 minus the force of gravity, which is -9.81 m/s^2).

Sensor Rate

- The rate sensor events are delivered at. This is only a hint to the system. Events may be received faster or slower than the specified rate. Usually events are received faster.
  - SENSOR_DEPLAY_FASTEST – Get sensor data as fast as possible
  - SENSOR_DEPLAY_GAME – Rate suitable for games
  - SENSOR_DEPLAY_NORMAL – Default rate suitable for screen orientation changes
  - SENSOR_DEPLAY_UI – Rate suitable for the user interface (Slowest)

Using Sensor



```java
public class MainActivity extends Activity implements SensorEventListener {
    private SensorManager mSensorManager;
    private Sensor mPressure;

    @Override
    public final void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
        mPressure = mSensorManager.getDefaultSensor(Sensor.TYPE_PRESSURE);
    }

    @Override
    public final void onAccuracyChanged(Sensor sensor, int accuracy) {
        // Do something here if sensor accuracy changes.
    }

    @Override
    public final void onSensorChanged(SensorEvent event) {
        float millibars_of_pressure = event.values[0];
    }

    @Override
    protected void onResume() {
        super.onResume();
        mSensorManager.registerListener(this, mPressure, SensorManager.SENSOR_DELAY_NORMAL);
    }

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

Used for receiving notifications from the SensorManager when sensor values have changed

Get an instance of the sensor service, and use that to get an instance of a particular sensor

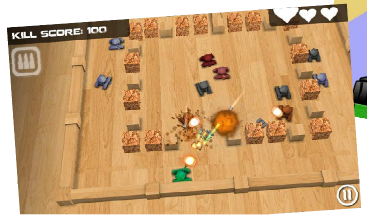
Handling incoming sensor data in the onSensorChanged() callback method

Register a listener for the sensor with specified rate

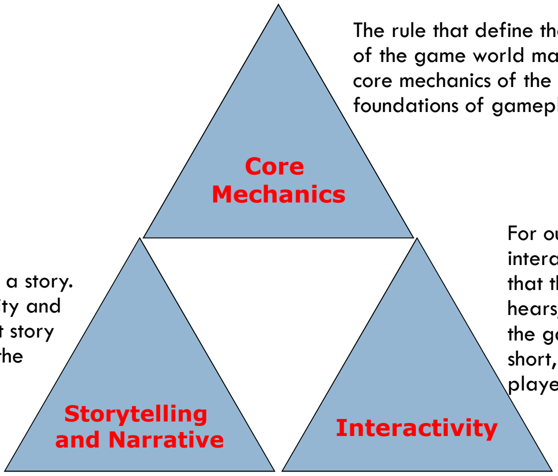
Unregister the sensor when the activity pauses

Mobile Game Development

- Mobile Phones have become the new GameBoys.
- Covers an even larger demographic: Hardcore gamers, casual gamers, mom, pop and even grandma play games now.
- Always connected.
- New distribution channels.
- Big Market, Small Developers.



Key Areas for Game Design



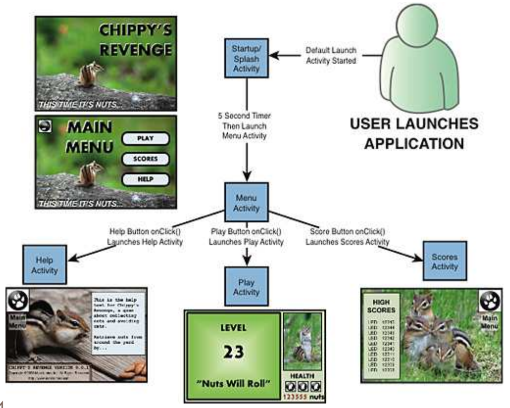
The rule that define the operation of the game world make up the core mechanics of the game, or the foundations of gameplay.

All game tell a story. The complexity and depth of that story depends on the game.

For our purposes, interactivity is the way that the player sees, hears, and acts within the game's world. In short, the way the player plays the game

How to build an Android Game?

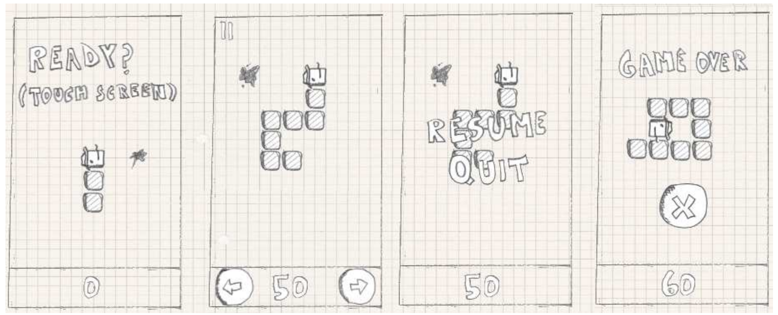
- Normally, the design of the game is simple. Which contain the game flow:
  - Splash
  - Menu
  - Play
  - Scores
  - Help


CHIPPY'S REVENGE
THIS TIME IT'S NUTS...
MAIN MENU
PLAY
SCORES
HELP
Startup/ Splash Activity
Default Launch Activity Started
USER LAUNCHES APPLICATION
5 Second Timer Then Launch Menu Activity
Menu Activity
Help Button onClick() Launches Help Activity
Play Button onClick() Launches Play Activity
Score Button onClick() Launches Scores Activity
Help Activity
Play Activity
Scores Activity
Main Menu
LEVEL
23
"Nuts Will Roll"
HEALTH
HIGH SCORES

Ref: http://www.computerworld.com/s/article/9181925/How_to_build_an_Android_application_step_by_step

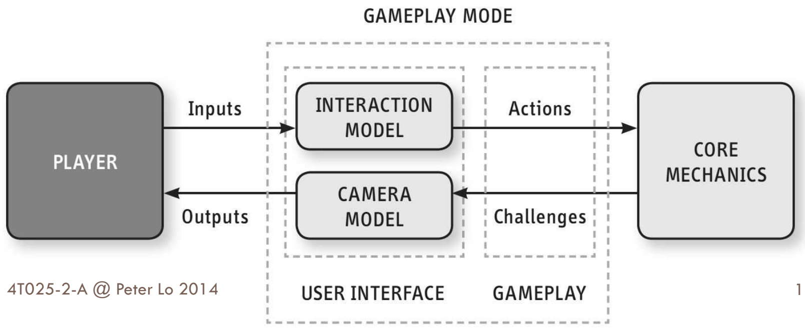
Game Screen Design

- A game is more than just its mechanics
- We have different screens for different status:
  - Main Menu, Game Playing, Paused, Game Over, Score
- Different events in a screen trigger a transition to another screen.



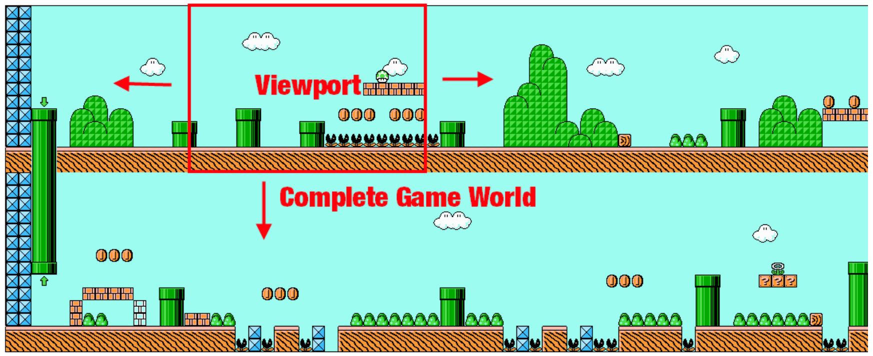
Gameplay Mode

- Gameplay modes consist of the available gameplay and user interface at a specific time
  - Not all actions are available at all times
  - Available user interface choices should be related to the current actions



Defining the Game World

- You need to define the virtual game world for your game, while only part of view port is available to users



How to Design Character?

- Base on the experiences from your own life.
  - You know best how you felt when something happened to you, you know what different types of people are like.
  - Make characters that other people can relate to.
- Make your characters unique.
  - While you will have to have a lot of similarities to other video game characters, part of being human is being unique, and if your character is not in the least unique then they will be less believable.