User Interaction: Intro to Android

Asst. Professor Donald J. Patterson
INF 133 Fall 2010
Multi-Touch Assignment Winner

Volleyball
Multi-Touch Assignment Winner

3% 7%

Volleyball
Fluid Paint
Ghetto Basketball
Ravenous Ravenous Rhinos
Multi-Touch Assignment Winner

- Volleyball: 3%
- Fluid Paint: 7%
- Ghetto Basketball: 17%
- Ravenous Ravenous Rhinos: 83%

Wednesday, November 24, 2010
Multi-Touch Assignment Winner

- Volleyball: 3%
- Fluid Paint: 7%
- Ghetto Basketball: 17%
- Ravenous Ravenous Rhinos: 21%
- Good Video
- Firing Range

Wednesday, November 24, 2010
Multi-Touch Assignment Winner

- Volleyball: 52%
- Fluid Paint: 3%
- Ghetto Basketball: 7%
- Ravenous Ravenous Rhinos: 17%
- Good Video: 21%
- Firing Range:
Space Hockey

Wednesday, November 24, 2010
Intro to Android

Developing with Sensors on Android

- Requirements
- SDK/AVD
- Eclipse Plug-in
- Hello World

Developing with Sensors on Android

- Requirements
- SDK/AVD
- Eclipse Plug-in
- Hello World

Intro to Android:

• On the emulator

User Application

Request Sensor Data by Intent

FAIL

Android OS Emulator

No Services that fulfill Intent

0
Simulating Sensors on the Emulator would require:

- User Application
- Request Sensor Data by Intent
- Android OS Emulator
- Sensor Simulator U/I for telling the emulator what the sensors should return

Intent gets fulfilled by software, not sensors.
Simulating Sensors on the Emulator would require:

- User Application
- Request Sensor Data by Intent
- Android OS Emulator
- Intent gets fulfill by software not sensors
- Sensor Data
- Sensor Simulator
- Sensor Simulator U/I for telling the emulator what the sensors should return
Intro to Android:

- Simulating Sensors on the Emulator would require:
  - User Application
  - Request Sensor Data by Intent
  - Android OS Emulator
  - Intent gets fulfilled by software not sensors
  - Sensor Simulator UI for telling the emulator what the sensors should return
Simulating Sensors on the Emulator would require:

- This used to exist but has fallen out of currency with Android SDK
- Known package at OpenIntents only works with pre 2.0 SDK
- No known work around
- Instead we must develop on live devices for sensors
- Ok for accelerometers, not okay for GPS
Intro to Android:

- Your approach

Stage 1

User Application

Request Sensor Data by Intent

Android OS Emulator

No Services that fulfill Intent

Stage 2

User Application

Request Sensor Data by Intent

Actual Android OS

Intent gets fulfill by hardware

Real Hardware Sensors
Intro to Android:

• Unpack the phone

http://www.google.com/support/android/bin/topic.py?hl=en&topic=28930
Intro to Android:

- Take a look at the sensors

http://www.google.com/support/android/bin/topic.py?hl=en&topic=28930
Intro to Android:

• Install the battery - Do Not Damage My Phones!

http://www.google.com/support/android/bin/topic.py?hl=en&topic=28930
Intro to Android:

- Charge the phone to 100%
  - USB to computer
  - USB to wall plug
- While charging, go through on-phone tutorial
- Do not sync to your Google
- Enable Location reporting
- Set Date and Time
- Register your device with OIT (or send me the MAC address)
  - Home -> Menu -> Settings -> About Phone -> Status

http://www.google.com/support/android/bin/topic.py?hl=en&topic=28930
Intro to Android:

• Stage 1
  • Get a Hello World program running in Eclipse
  • Execute it on an emulated phone
Intro to Android:

- **Stage 2**
  - Get a Hello World program running in Eclipse
  - Execute it on a real phone
  - Identify your application as “debuggable”
Intro to Android:

• Stage 2

### Sensor Data

User Application

Request Sensor Data
by Intent

Actual

Android OS

Intent gets
fulfilled by
hardware

Real Hardware

Sensors

Stage 2

Wednesday, November 24, 2010
Intro to Android:

- Stage 2
  - Get a Hello World program running in Eclipse
  - Execute it on a real phone
    - Identify your application as “debuggable”
  - Turn on USB Debugging on the phone
    - Home -> Menu -> Settings -> Applications -> Development -> USB Debugging
  - Windows only: Download/install a driver
  - Run from Eclipse

http://developer.android.com/guide/developing/device.html
Intro to Android:

• Playing a sound
  • The key is the MediaPlayer call
  • Do not instantiate more than one MediaPlayer object

```java
static MediaPlayer mp = new MediaPlayer();
public void playSound(String path) {
    if (mp.isPlaying()) {
        return;
    }
    mp.reset();
    try {
        mp.setDataSource(path);
        mp.prepare();
    } catch (Exception ex) {
        Log.d("main thread ex", ex.getStackTrace()[0].toString() + " path: " + path);
    }
    mp.start();
}
```

Intro to Android:

- Playing a sound

- You will need to get the audio media onto the phone
Intro to Android:

• One possible architecture for getting sensor readings

• Steps
  • Create a U/I for the data
  • Instantiate your Sensor Recorder
  • Register for sensor callbacks
  • Instantiate your UI Updater
  • Have a timer occasionally run it
Intro to Android:

• Hints

```java
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main); // This is from an xml description of the UI

deviceSensor = new DeviceSensor(); // I implement this class

    /* This is provided by the Android OS */
mSensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);

    mSensorManager.registerListener(deviceSensor, mSensorManager
        .getDefaultSensor(Sensor.TYPE_ORIENTATION),
        SensorManager.SENSOR_DELAY_FASTEST);

    Timer timerUI = new Timer();
    UpdateUITask updateValuesTask = new UpdateUITask(this); // I implement this class
    timerUI.schedule(updateValuesTask, 500, 500);
}
```