On the News….

Android has overtaken Apple iPhone as the second most popular mobile in the first quarter of 2010. [*NPD, May 2010]*

Market share:
(Android) 28% vs. 21% (iPhone)

Outline

- Basic Terminologies
- `Camera` Class
- Linear Algebra Library in Java

Four Basic Building Blocks of Androids

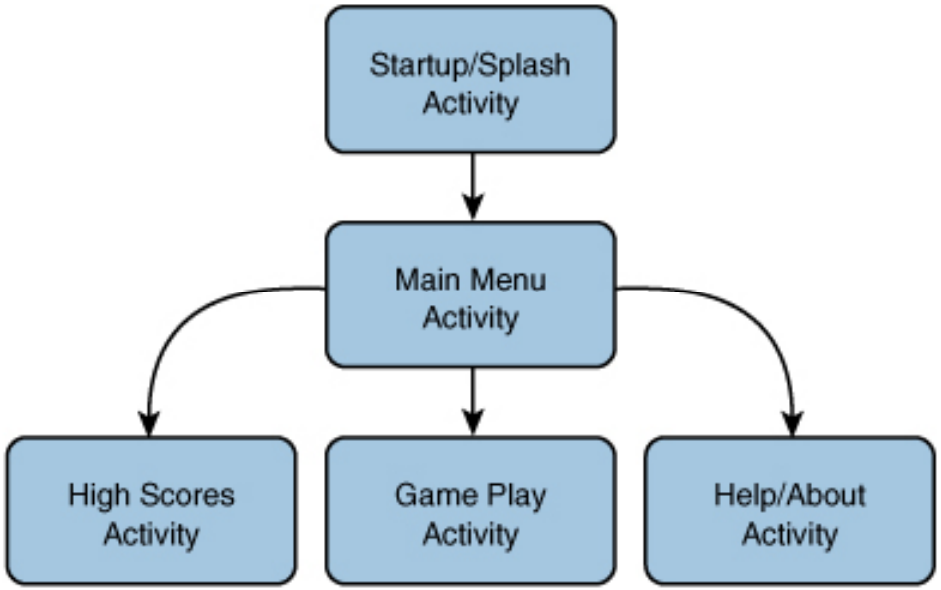
- Activities
  - Basic building blocks of every android applications
- Services
  - An background application without UI that may run indefinitely
- Broadcasts Receivers
  - Receive and respond to any broadcast announcements across the system
- Content Providers
  - Data-abstraction for exposing application data to others

Communicate with each other through:

Intents and Intents filter

Activities

- An Android application is a collection of tasks, each of which is called an **Activity**
- Each **Activity** within an application has a unique task or purpose.
- Example: A simple game application



Entry Points of an Application

- no main() function
- Android applications can have multiple entry points.
- A specific **Activity** can be designated as the main **Activity** to launch by default within the AndroidManifest.xml file
- Other **Activities** might be designated to launch under specific circumstances.
- **Example:** To launch another activity from your current activity through **Intent**

```
startActivity(new Intent(getApplicationContext(),
MyDrawActivity.class));
```

Intent

- **Intent** is a asynchronous message mechanism to match task requests with the appropriate **Activity** or **Services** and to dispatch broadcast Intents events to the system.

- **Example:** The main menu of a gaming application may use intent to launch other activities, such as Play Games Activity, High Scores Activity and Help Activity.

- Allows you to launch an activity belonging to another application. For example, to launch a web page:

```
Uri url = Uri.parse(http://www.stanford.edu);
Intent browse = new Intent(Intent.ACTION_VIEW, url);
startActivity(browse );
```

The Camera Class

- Connect/disconnect with the camera service
- Set capture settings
- Start/stop preview
- Snap a picture
- Allows you to manipulate streaming camera previews

Declare Permissions/Features

AndroidManifest.xml

```
<uses-permission android:name="android.permission.CAMERA" />
<uses-feature android:name="android.hardware.camera" />
<uses-feature android:name="android.hardware.camera.autofocus" />
```

- Declare the appropriate permission and camera features in your Android Manifest.
- Grant the application access to `Camera` Services.

Initialization

In your activity/view class:

```
Camera camera = Camera.open();
  [ … ]
camera.release();
```

- No default constructor for Camera class.
- Use static open method.
- Remember to release your hold on the Service at the end of your processing or activity.
- Don’t have to be called at the same function

Setting/Getting Camera Parameters

[http://*developer.android.com/reference/android/hardware/Camera.Parameters.html*]

```
Camera.Parameters parameters =
camera.getParameters();
parameters.setPictureFormat(PixelFormat.JPEG);
camera.setParameters(parameters);
```

- Parameters include:
  - Picture sizes
  - Preview refresh rate
  - anti-banding,
  - imaging effects: sepia, negative, etc,
  - flash
  - auto focus:
  - scenic mode: action, beach, landscape, night, sport, etc,
  - white balancing: fluorescent, shade, twilight..

Using Camera Preview

```
camera.setPreviewDisplay(mySurface);
camera.startPreview();
      [ … ]
camera.stopPreview();
```

- Allows you to access camera's streaming video
- A good starting point for your mobile augment reality application

Using Camera Preview (II)

```
camera.setPreviewCallback(new PreviewCallback() {

  public void onPreviewFrame(byte[] _data, Camera
_camera) {
   // process your preview images  }
});
```

- Set a override method `onPreviewFrame`
- Allows you to manipulate and display your preview image
- If intensive processing is required, processing should be done on a separate thread to avoid stalling the preview frame.

Taking a Picture

```
private void takePicture() {
  camera.takePicture(shutterCallback, rawCallback,
jpegCallback);
}
```

- Trigger a series of callbacks to the applications as image capture progresses
- Shutter callbacks occurs after the image is captured
- Raw callback occurs when the raw image data is available
- JPEG callback occurs when the compressed image is available

Picture Callbacks

```
ShutterCallback shutterCallback = new ShutterCallback() {
  public void onShutter() {
    // Do something when the shutter closes.
  }
};

PictureCallback rawCallback = new PictureCallback() {
  public void onPictureTaken(byte[] _data, Camera _camera) {
    // Do something with the image RAW data.
  }
};

PictureCallback jpegCallback = new PictureCallback() {
  public void onPictureTaken(byte[] _data, Camera _camera) {
    //Do something with the image JPEG data.
  }
};
```

Linear Algebra Library

- JAMA : A Java Matrix Package
- Features:
  - Matrix elementary operation, pesudo-inverse, rank, Cholesky/LU/QR/Eigenvalue and Singular Value Decomposition, least squares
- Example:

```
double[][] array = {{1.,2.,3},{4.,5.,6.},{7.,8.,10.}};
Matrix A = new Matrix(array);
Matrix b = Matrix.random(3,1);
Matrix x = A.solve(b);
Matrix Residual = A.times(x).minus(b);
double rnorm = Residual.normInf();
```

- Not tested for speed and full capabilities with Android
- Jar file is available at: http://math.nist.gov/javanumerics/jama/

Other un-tested libraries

- **ojAlgo**
  - http://ojalgo.org/, claimed to be faster than Jama
- Efficient Java Matrix Library (EJML)

References

- *Android Wireless Application Development* by Shane Conder, Lauren Darcey, 2009
- *Professional Android Application Development* by Reto Meier, 2009