sana

Open Source Telemedicine

Android Client Development Introduction

Open Source Telemedicine

Development Environment Requirements

Android Client

Resources

Best source of information is the Android development site:

http://developer.android.com/sdk/index.html

Basic Requirements

OS
- Windows XP, Vista or 7
- Mac OS X 10.5.8
- Linux, glibc 2.7

SDKs
- JDK 1.5
- Android SDK

http://developer.android.com/sdk/requirements.html

IDE
- Eclipse 3.6
- ADT plugin

SDK Components
- SDK Platform 1.6

Android Phone or Virtual Device

Additional Resources

Android SDK Installation
http://developer.android.com/sdk/installing.html

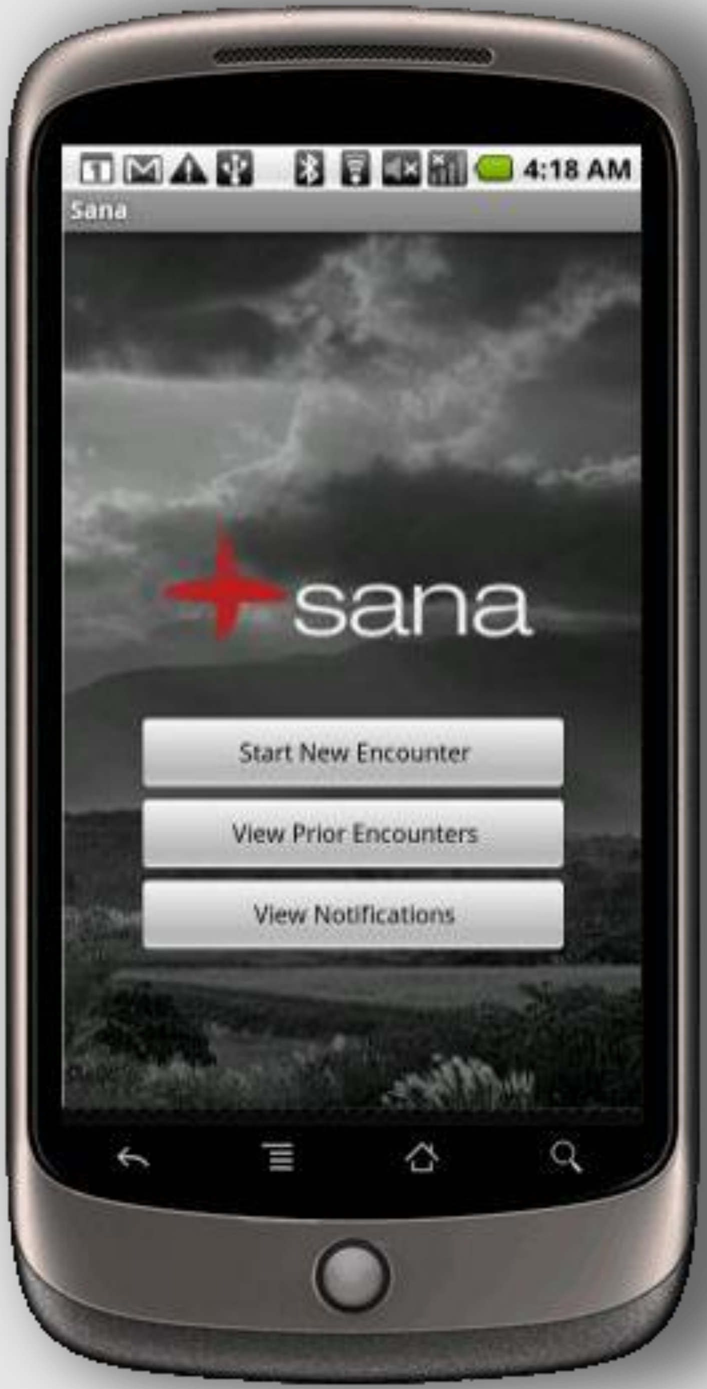
Eclipse ADT
http://developer.android.com/sdk/eclipse-adt.html

SDK Components
http://developer.android.com/sdk/adding-components.html

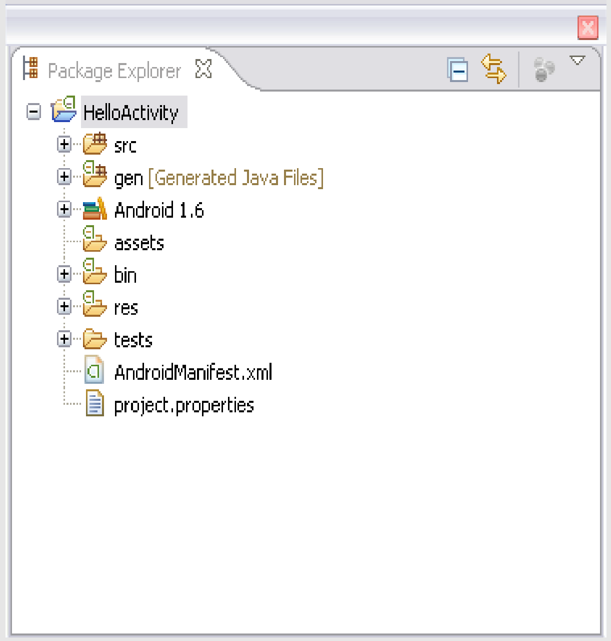
AVDs
http://developer.android.com/guide/developing/devices/index.html

Open Source Telemedicine

Hello Android



Android Application Structure



Package Explorer

HelloActivity
- src
- gen [Generated Java Files]
- Android 1.6
- assets
- bin
- res
- tests
- AndroidManifest.xml
- project.properties

Courtesy of Eclipse Foundation. Used with permission.

- **src** – The actual source code
- **gen** – index files generated when compiling
- **res** – The raw resource files that will get indexed
- **assets** – Raw files that will not get indexed
- **bin** – The compiled application
- **Android 1.6** – the Android libraries
- **tests** – Unit test code(not required)

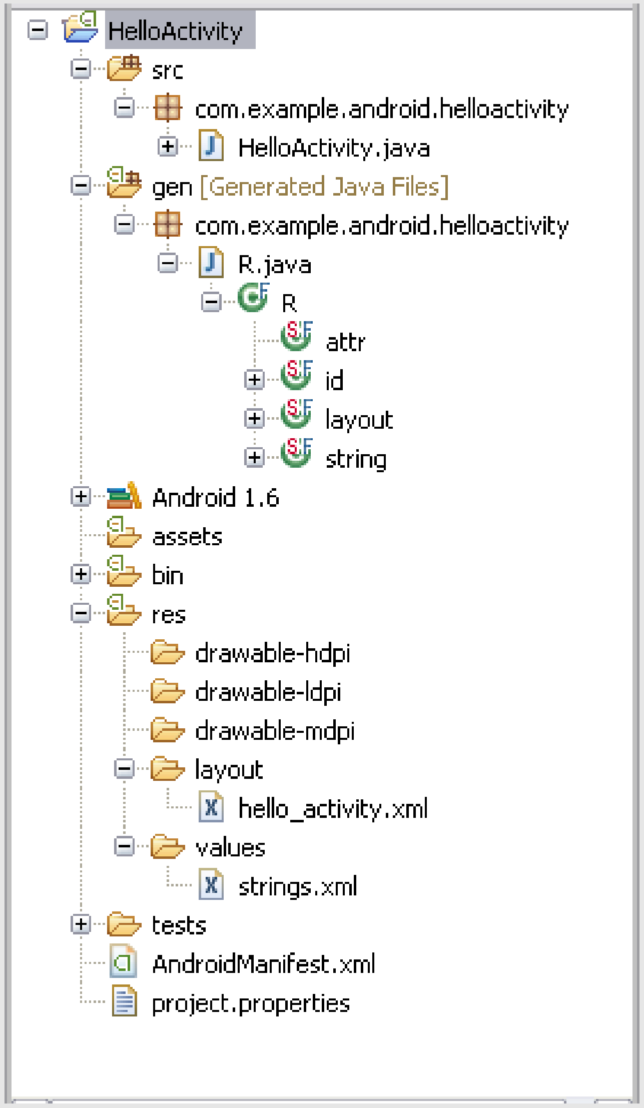
Android Application Structure Resources

Indexed Resources

- **drawable**
- **layout** xml files that define UI
- **values** strings, styles

Additional indexed resources

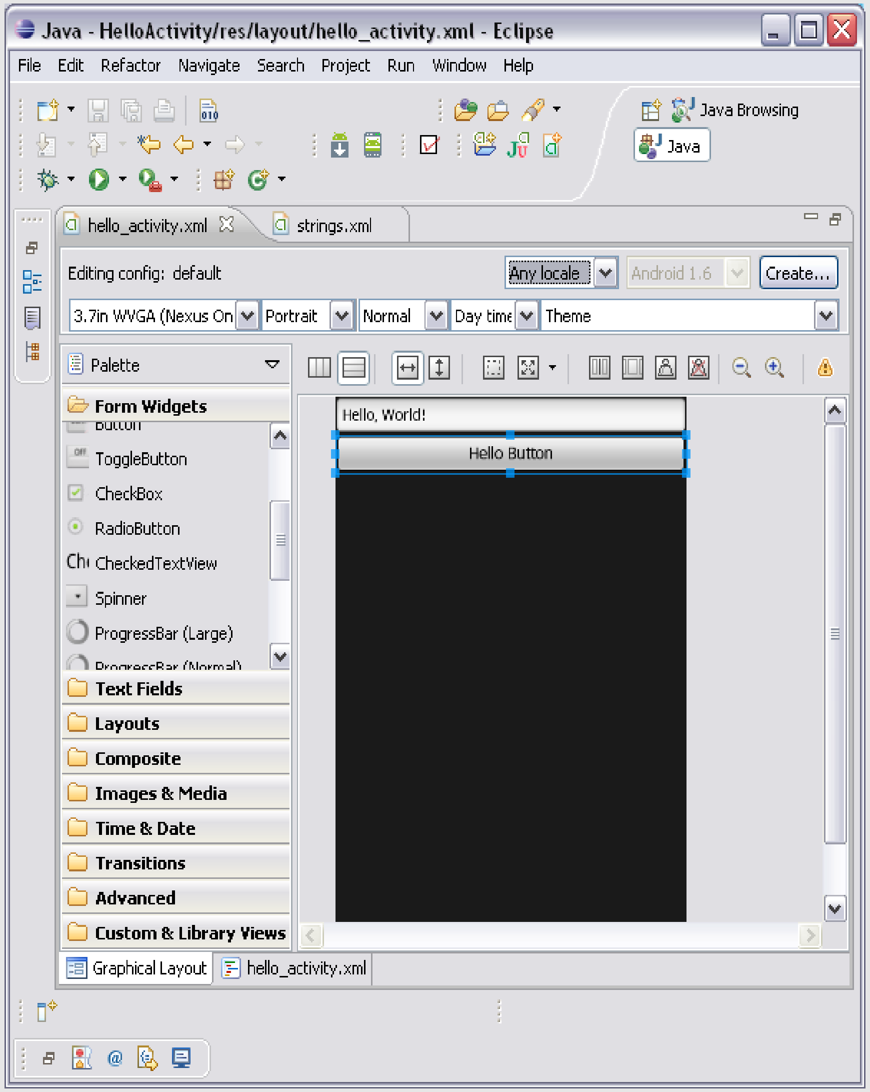
- **xml** compiled xml files
- **raw** any raw file that you want indexed



```
HelloActivity
  src
    com.example.android.helloactivity
      HelloActivity.java
  gen [Generated Java Files]
    com.example.android.helloactivity
      R.java
        R
          attr
          id
          layout
          string
  Android 1.6
  assets
  bin
  res
    drawable-hdpi
    drawable-ldpi
    drawable-mdpi
    layout
      hello_activity.xml
    values
      strings.xml
  tests
  AndroidManifest.xml
  project.properties
```

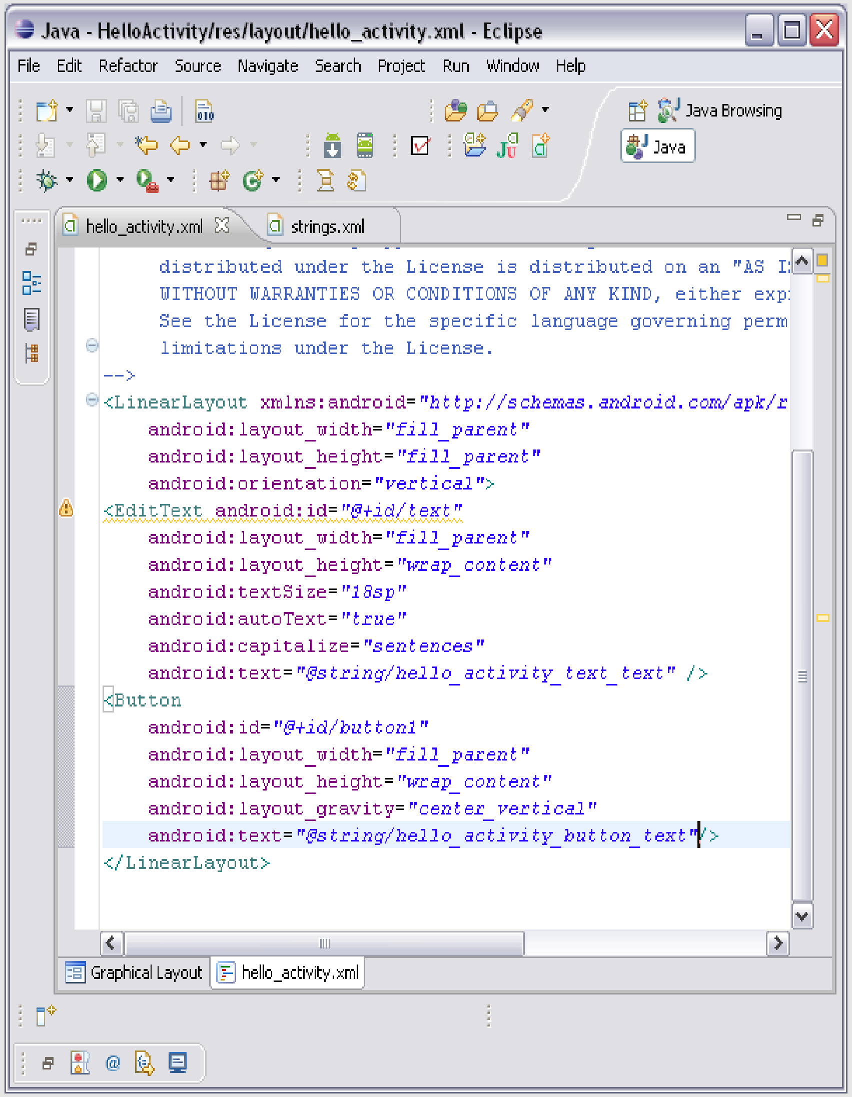
Courtesy of Eclipse Foundation. Used with permission.

Android Application Structure Layouts



Editing config: default | Any locale | Android 1.6 | Create...

3.7in WVGA (Nexus On | Portrait | Normal | Day time | Theme

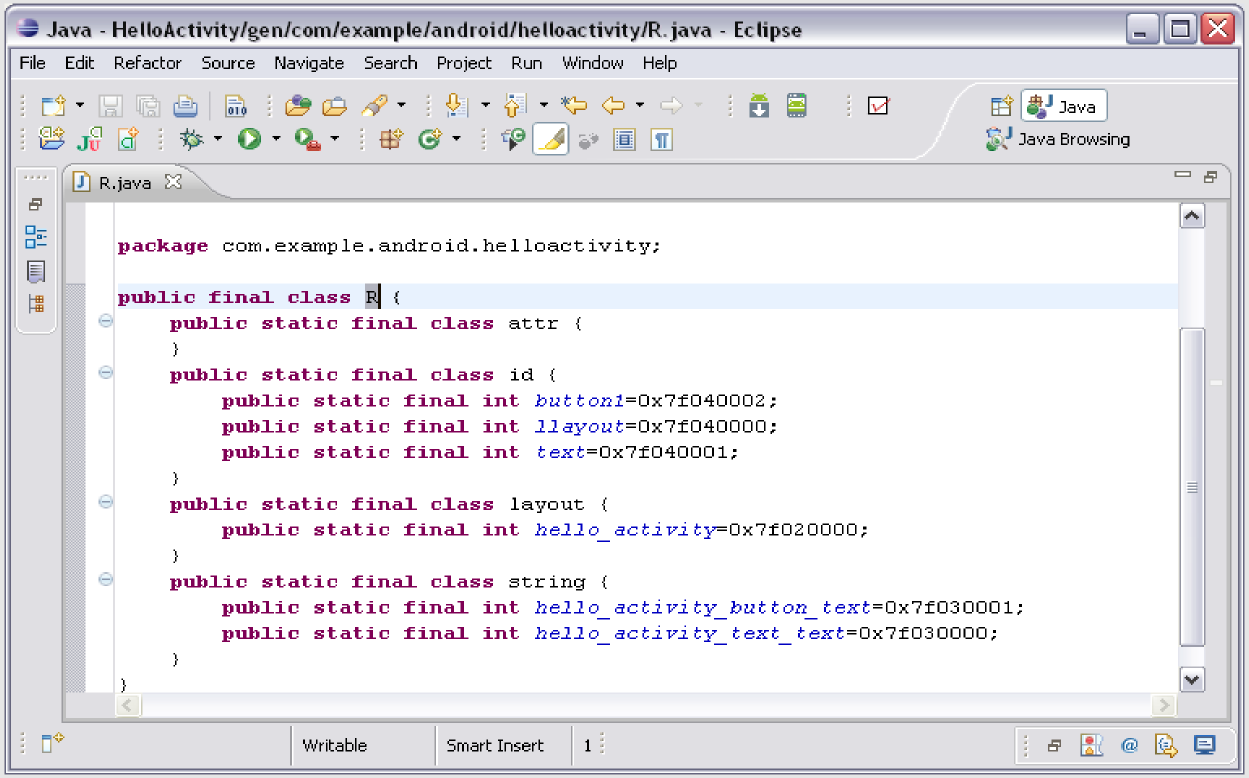


Java - HelloActivity/res/layout/hello_activity.xml - Eclipse
File Edit Refactor Source Navigate Search Project Run Window Help
Java Browsing
Java
hello_activity.xml
strings.xml
Graphical Layout
hello_activity.xml

```
    distributed under the License is distributed on an "AS I
    WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either exp
    See the License for the specific language governing perm
    limitations under the License.
-->
<LinearLayout xmlns:android="http://schemas.android.com/apk/r
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">
<EditText android:id="@+id/text"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:textSize="18sp"
    android:autoText="true"
    android:capitalize="sentences"
    android:text="@string/hello_activity_text_text" />
<Button
    android:id="@+id/button1"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center_vertical"
    android:text="@string/hello_activity_button_text"/>
</LinearLayout>
```

Courtesy of Eclipse Foundation. Used with permission.

Android Application Structure Accessing Indexed Resources


Java - HelloActivity/gen/com/example/android/helloactivity/R.java - Eclipse
File Edit Refactor Source Navigate Search Project Run Window Help
Java
Java Browsing
R.java
Writable
Smart Insert
1

```java
package com.example.android.helloactivity;

public final class R {
    public static final class attr {
    }
    public static final class id {
        public static final int button1=0x7f040002;
        public static final int llayout=0x7f040000;
        public static final int text=0x7f040001;
    }
    public static final class layout {
        public static final int hello_activity=0x7f020000;
    }
    public static final class string {
        public static final int hello_activity_button_text=0x7f030001;
        public static final int hello_activity_text_text=0x7f030000;
    }
}
```

Courtesy of Eclipse Foundation. Used with permission.

- import the generated R class
- Access using the name space

```
R.id.button1
R.string.hello_activity_button_text
```

Android Application Components

Building blocks of all Android Applications

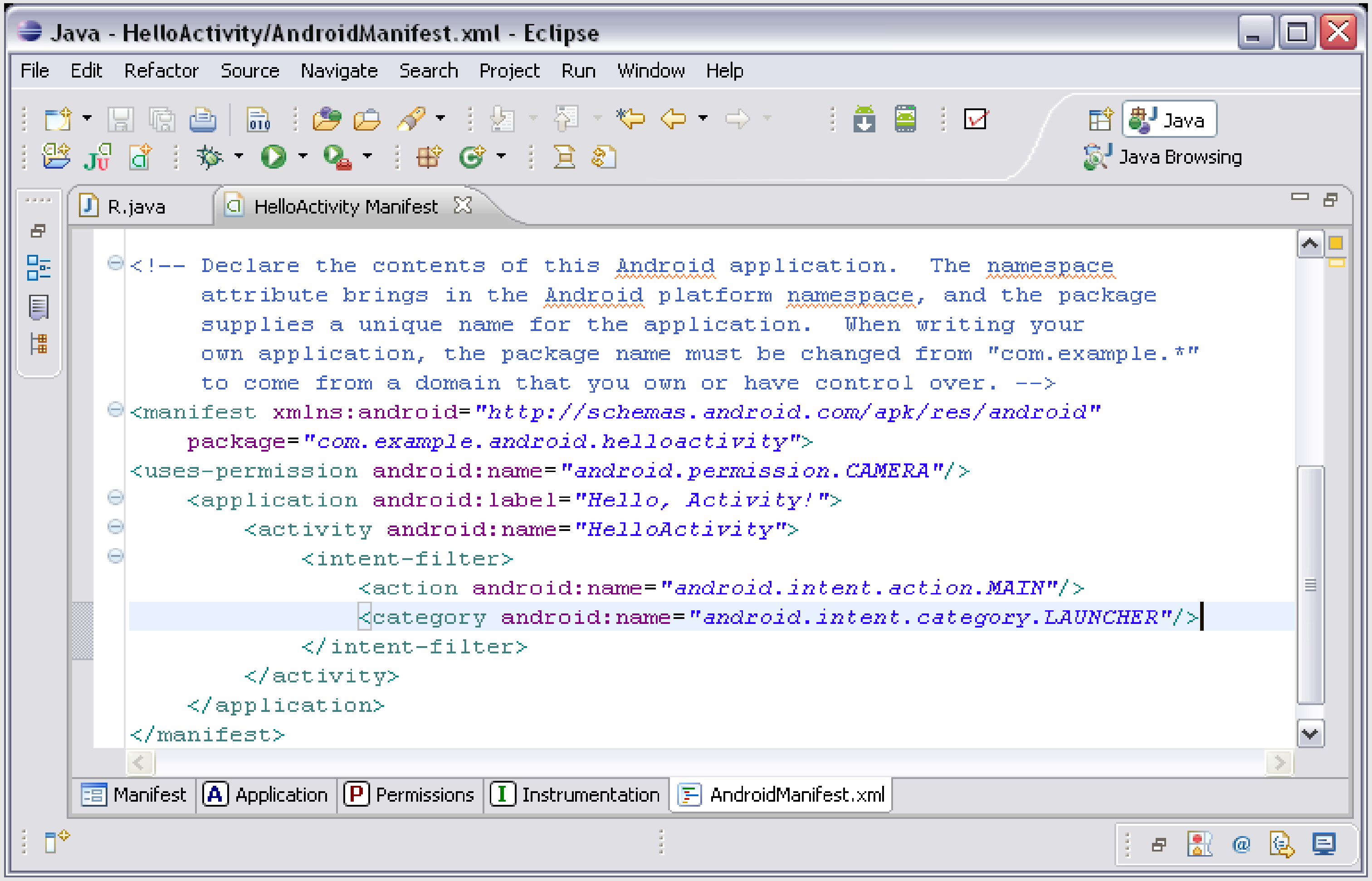
Activities -represent a single screen of the UI

Services – run in the background; i.e. perform long running operations, etc.

Content Providers – manage application data.

Broadcast Receivers – respond to system wide announcements

Android Application Manifest


Java - HelloActivity/AndroidManifest.xml - Eclipse
File Edit Refactor Source Navigate Search Project Run Window Help
Java
Java Browsing
R.java
HelloActivity Manifest
Manifest
Application
Permissions
Instrumentation
AndroidManifest.xml

```
<!-- Declare the contents of this Android application.  The namespace
     attribute brings in the Android platform namespace, and the package
     supplies a unique name for the application.  When writing your
     own application, the package name must be changed from "com.example.*"
     to come from a domain that you own or have control over. -->
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.android.helloactivity">
<uses-permission android:name="android.permission.CAMERA"/>
    <application android:label="Hello, Activity!">
        <activity android:name="HelloActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Courtesy of Eclipse Foundation. Used with permission.

Declares the applications capabilities

Android Applications Additional Core Concepts

Intents: The messages passed between components of the same or different applications

Security and Permissions: applications do not share resources and data by default; i.e. each application runs in its own sandbox.

Processes and Threads: “Every application runs in its own process and all components of the application run in that process, by default”

Android Applications Building and Running

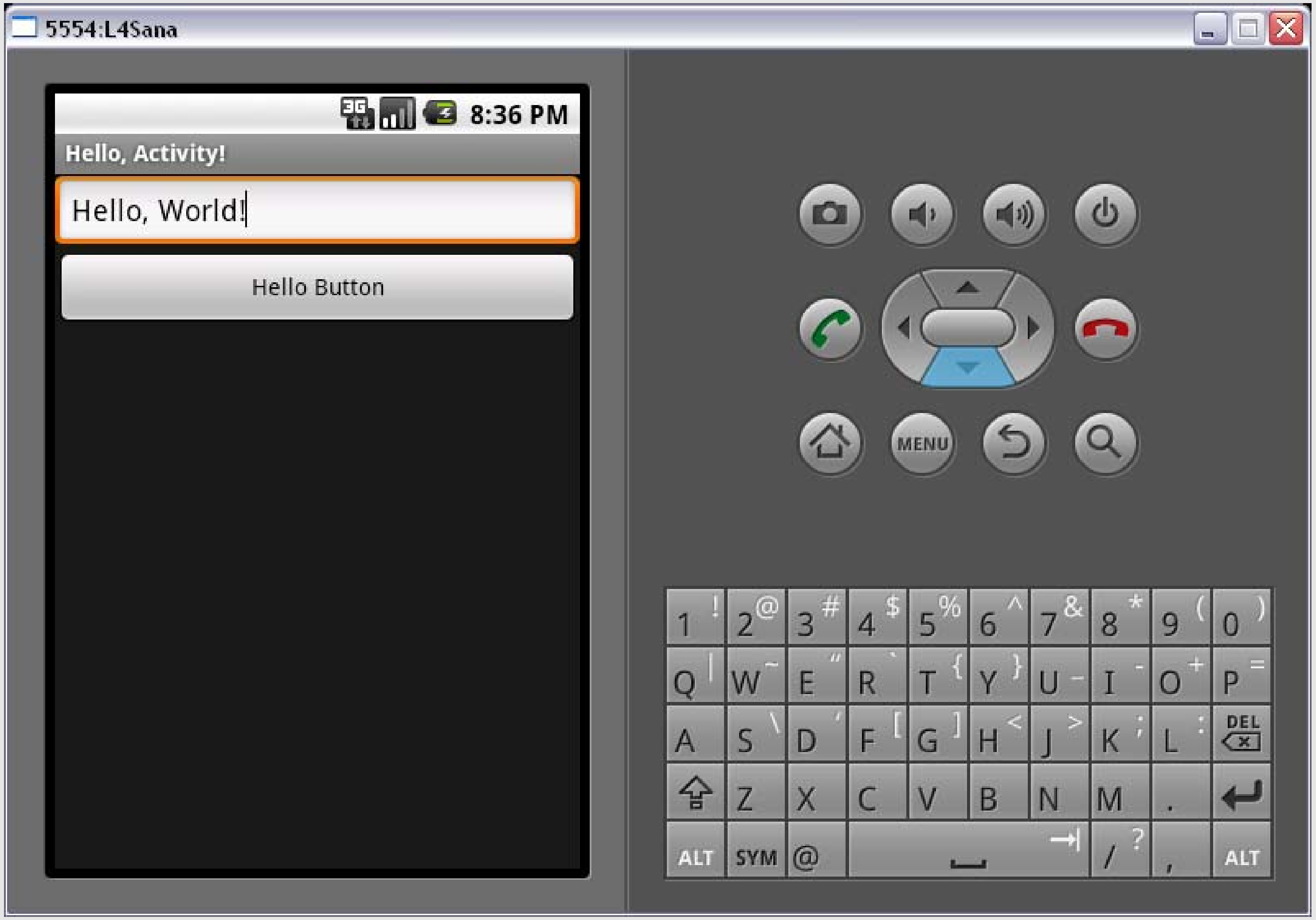
Compling: Android applications are compiled into optimized Android application files. Compresses and aligns uncompressed data into .apk files.

Running the application: The application must be deployed to an AVD or physical device.

If you are using Eclipse, much of the work will be done automagically but can also be performed using command line tools.

Android Virtual Devices

Running your application



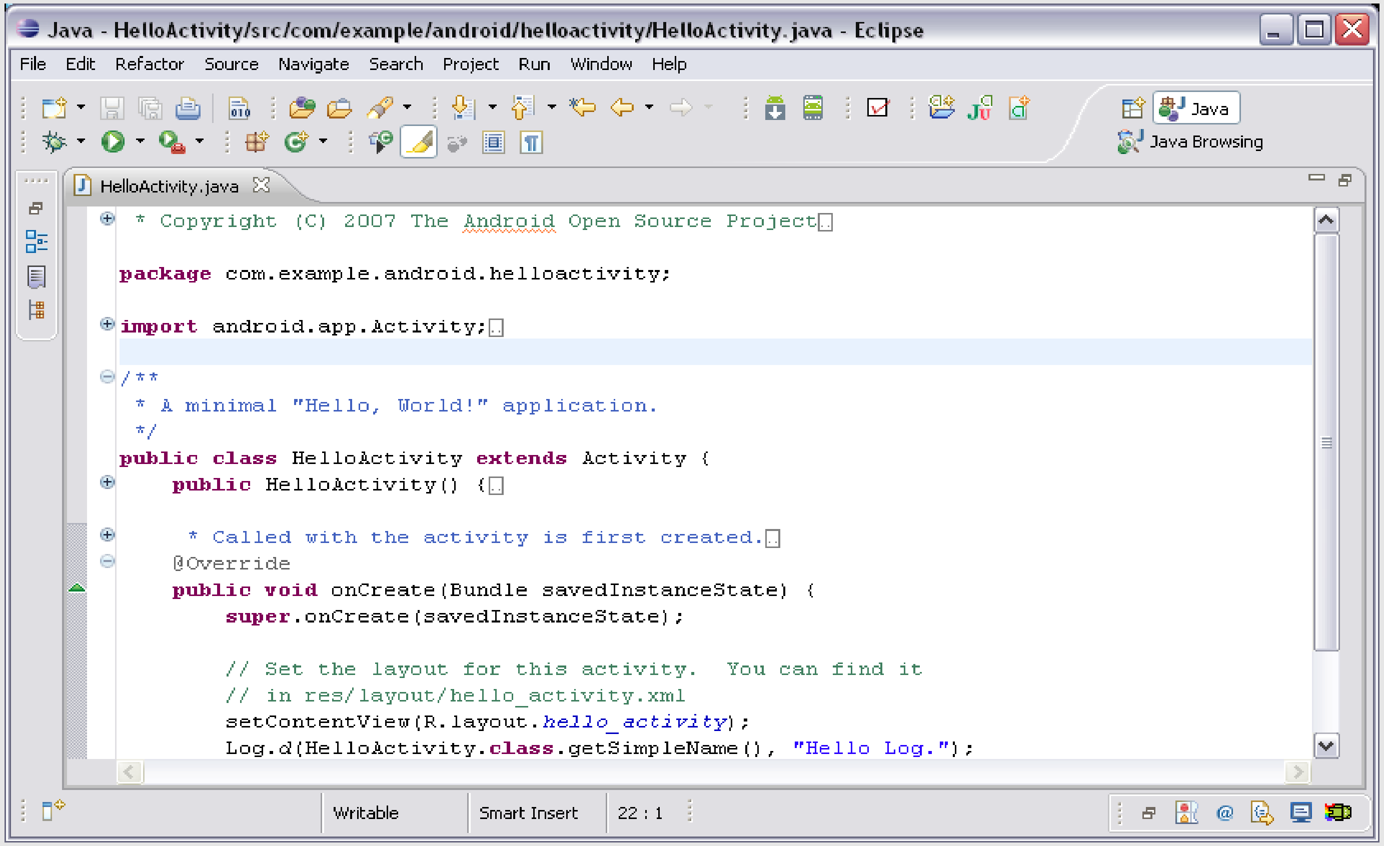
Courtesy of Eclipse Foundation. Used with permission.

Debugging

Use the Android logging capabilities

`android.util.Log`

- Log levels
- Tagged


Java - HelloActivity/src/com/example/android/helloactivity/HelloActivity.java - Eclipse
File Edit Refactor Source Navigate Search Project Run Window Help
Java
Java Browsing
HelloActivity.java
Writable
Smart Insert
22 : 1

```
 * Copyright (C) 2007 The Android Open Source Project

package com.example.android.helloactivity;

import android.app.Activity;

/**
 * A minimal "Hello, World!" application.
 */
public class HelloActivity extends Activity {
    public HelloActivity() {

     * Called with the activity is first created.
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // Set the layout for this activity.  You can find it
        // in res/layout/hello_activity.xml
        setContentView(R.layout.hello_activity);
        Log.d(HelloActivity.class.getSimpleName(), "Hello Log.");
```

Courtesy of Eclipse Foundation. Used with permission.

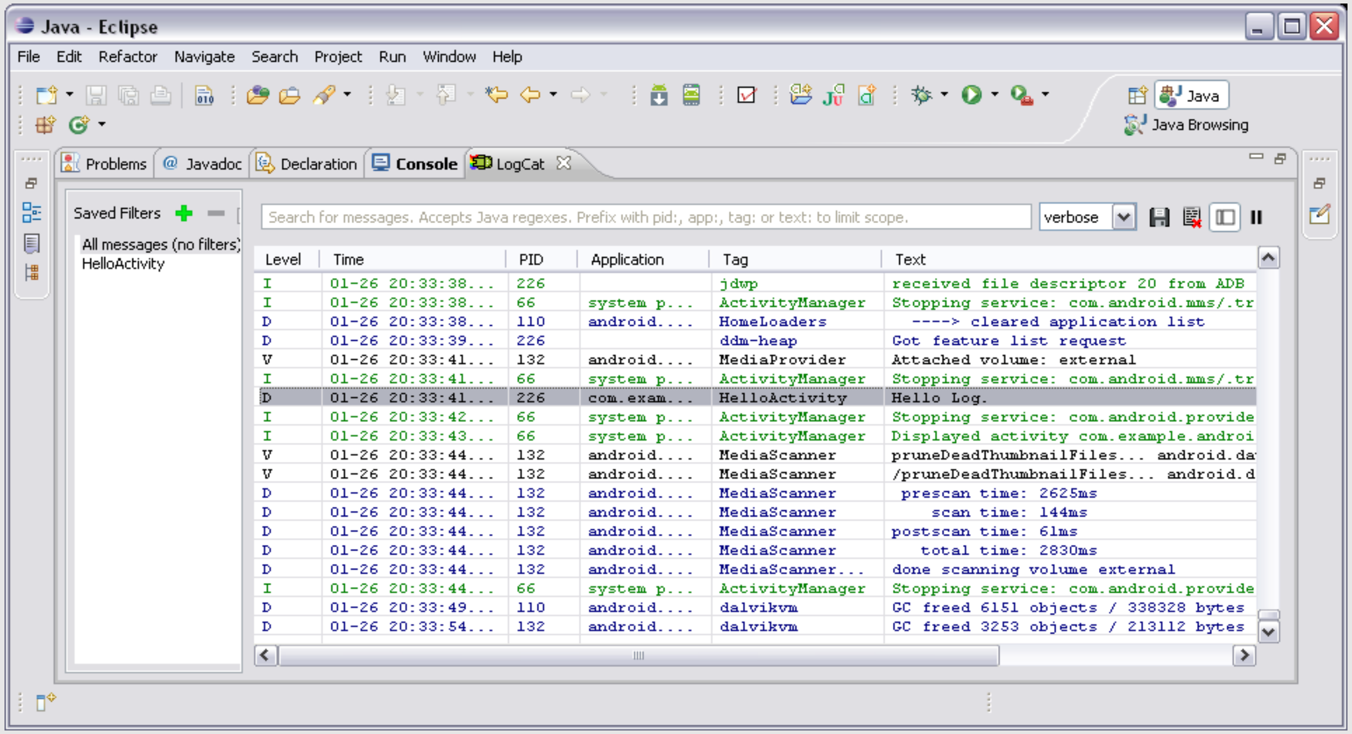
Debugging

Use the Android logging capabilities

`android.util.Log`

- Log levels
- Tagged

Logcat is your friend!


Java - Eclipse
File Edit Refactor Navigate Search Project Run Window Help
Java
Java Browsing
Problems
Javadoc
Declaration
Console
LogCat
Saved Filters
All messages (no filters)
HelloActivity
Search for messages. Accepts Java regexes. Prefix with pid:, app:, tag: or text: to limit scope.
verbose

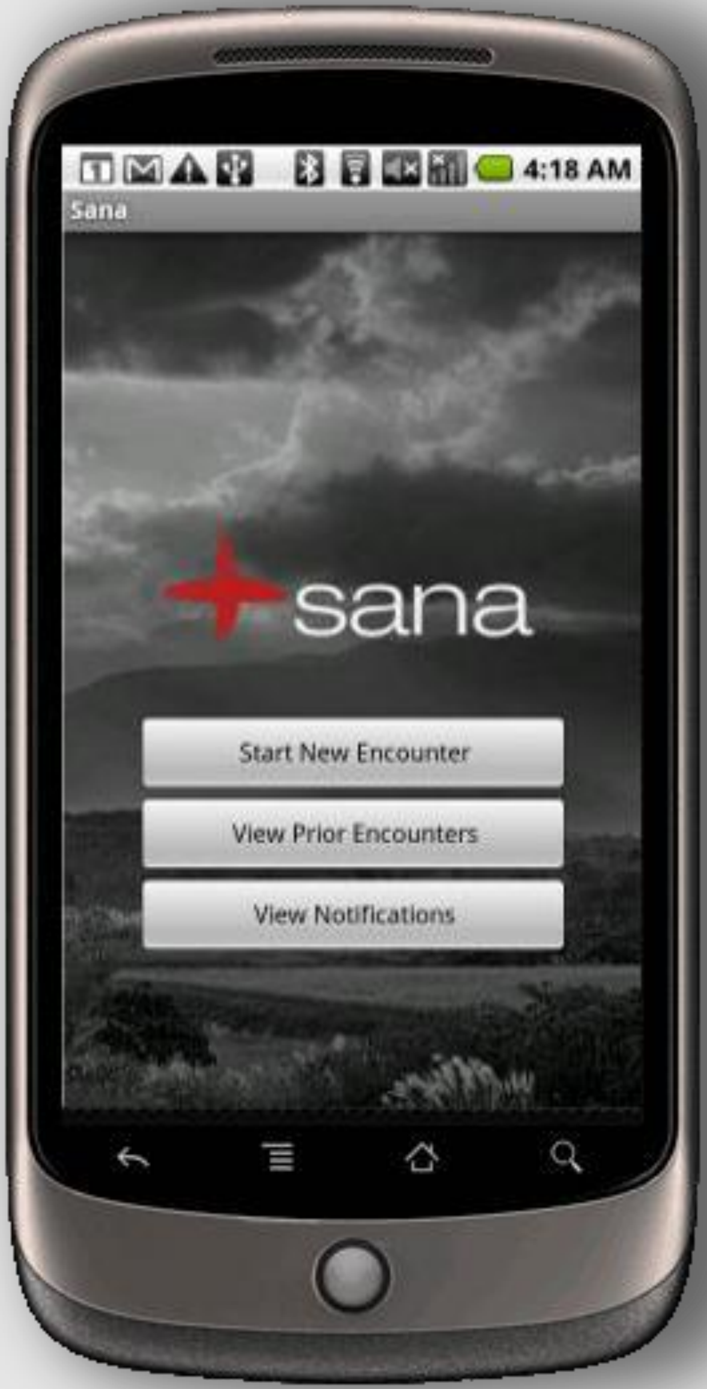
```
Level  Time                PID  Application    Tag                Text
I      01-26 20:33:38...   226                 jdwp               received file descriptor 20 from ADB
I      01-26 20:33:38...   66   system p...    ActivityManager    Stopping service: com.android.mms/.tr
D      01-26 20:33:38...   110  android....    HomeLoaders          ----> cleared application list
D      01-26 20:33:39...   226                 ddm-heap           Got feature list request
V      01-26 20:33:41...   132  android....    MediaProvider      Attached volume: external
I      01-26 20:33:41...   66   system p...    ActivityManager    Stopping service: com.android.mms/.tr
D      01-26 20:33:41...   226  com.exam...    HelloActivity      Hello Log.
I      01-26 20:33:42...   66   system p...    ActivityManager    Stopping service: com.android.provide
I      01-26 20:33:43...   66   system p...    ActivityManager    Displayed activity com.example.androi
V      01-26 20:33:44...   132  android....    MediaScanner       pruneDeadThumbnailFiles... android.da
V      01-26 20:33:44...   132  android....    MediaScanner       /pruneDeadThumbnailFiles... android.d
D      01-26 20:33:44...   132  android....    MediaScanner        prescan time: 2625ms
D      01-26 20:33:44...   132  android....    MediaScanner           scan time: 144ms
D      01-26 20:33:44...   132  android....    MediaScanner       postscan time: 61ms
D      01-26 20:33:44...   132  android....    MediaScanner          total time: 2830ms
D      01-26 20:33:44...   132  android....    MediaScanner...    done scanning volume external
I      01-26 20:33:44...   66   system p...    ActivityManager    Stopping service: com.android.provide
D      01-26 20:33:49...   110  android....    dalvikvm           GC freed 6151 objects / 338328 bytes
D      01-26 20:33:54...   132  android....    dalvikvm           GC freed 3253 objects / 213112 bytes
```

Courtesy of Eclipse Foundation. Used with permission.

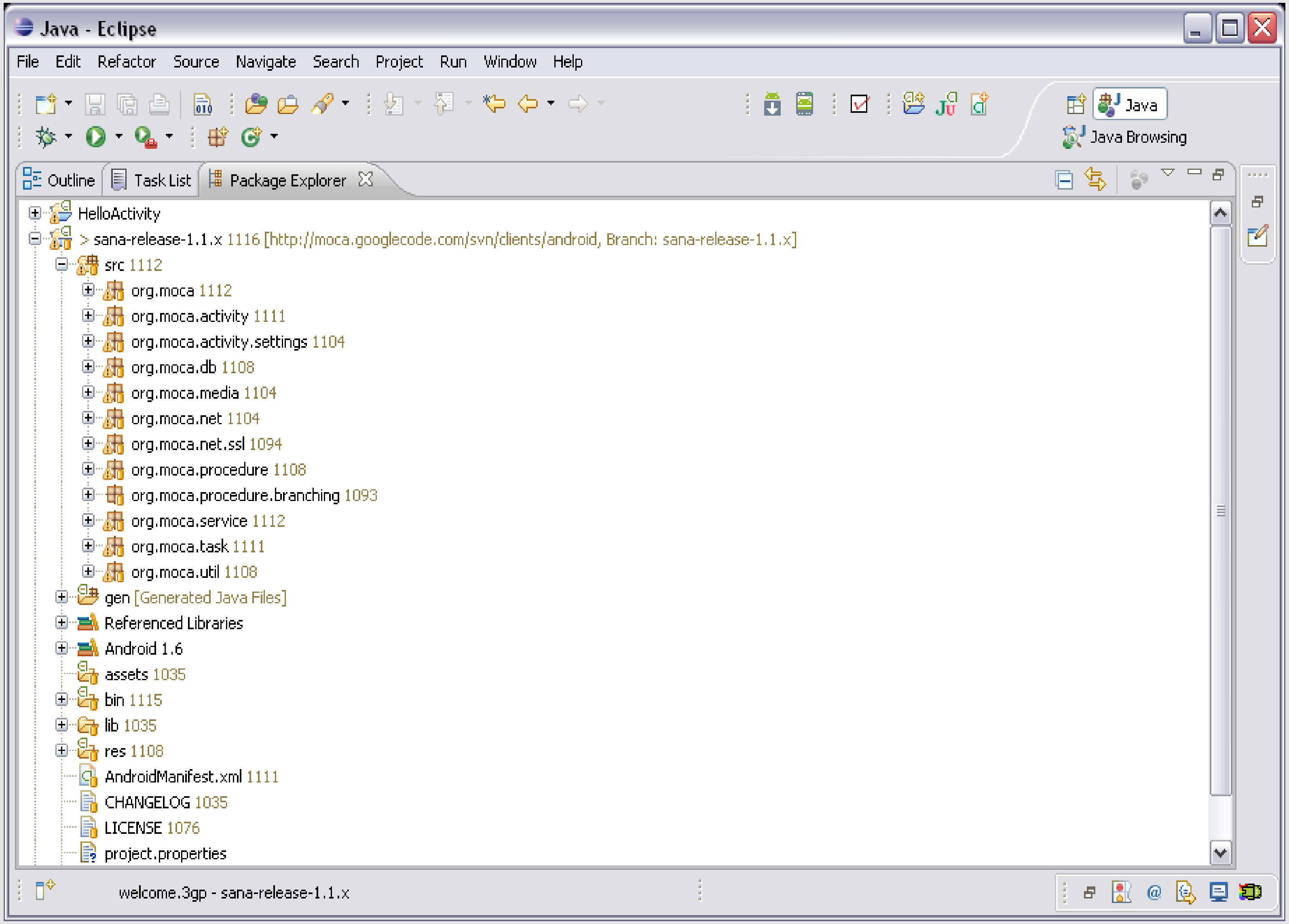
sana

Open Source Telemedicine

Hello Sana



Sana Application Structure

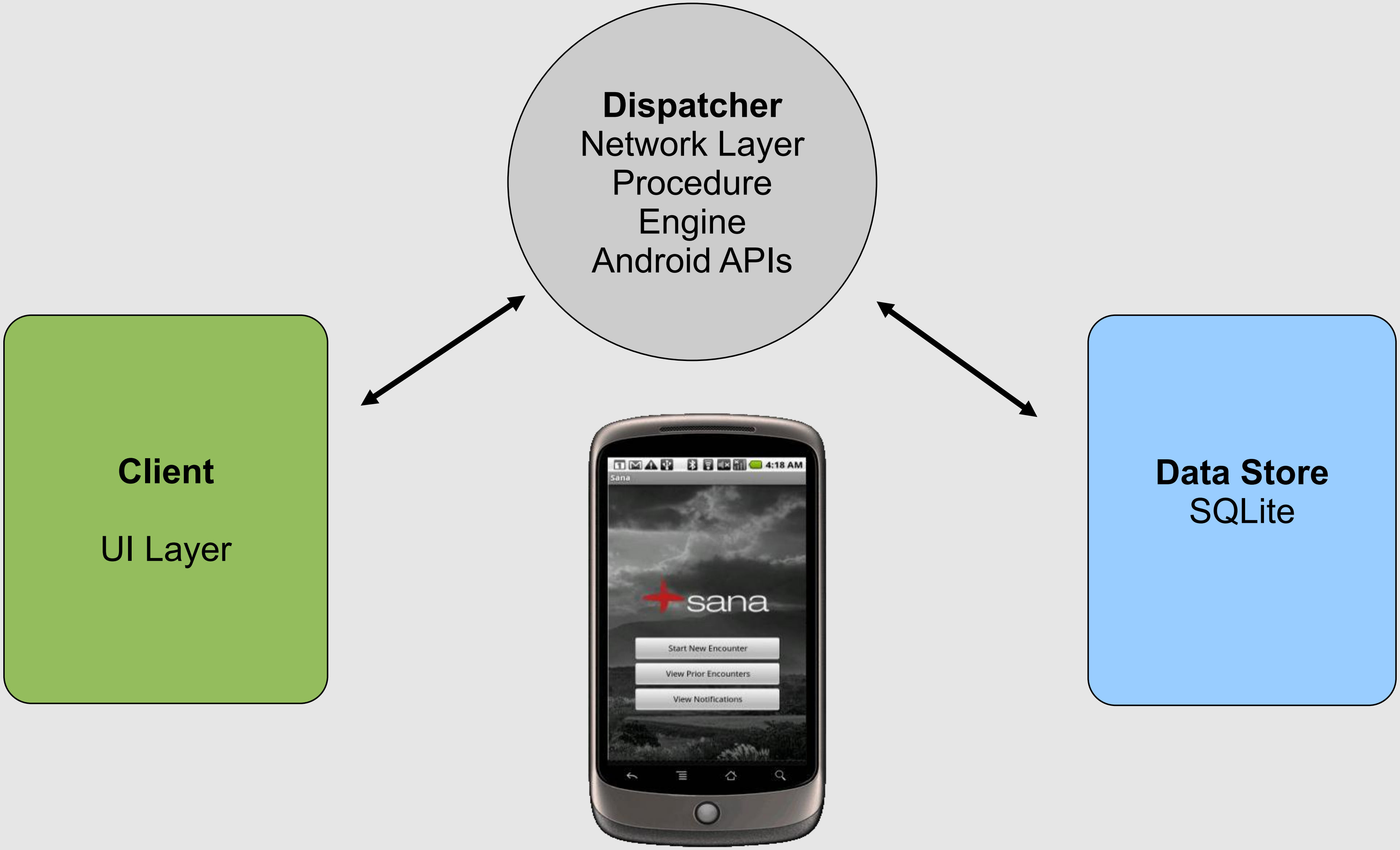

Java - Eclipse
File Edit Refactor Source Navigate Search Project Run Window Help
Java
Java Browsing
Outline
Task List
Package Explorer

```
HelloActivity
> sana-release-1.1.x 1116 [http://moca.googlecode.com/svn/clients/android, Branch: sana-release-1.1.x]
    src 1112
        org.moca 1112
        org.moca.activity 1111
        org.moca.activity.settings 1104
        org.moca.db 1108
        org.moca.media 1104
        org.moca.net 1104
        org.moca.net.ssl 1094
        org.moca.procedure 1108
        org.moca.procedure.branching 1093
        org.moca.service 1112
        org.moca.task 1111
        org.moca.util 1108
    gen [Generated Java Files]
    Referenced Libraries
    Android 1.6
    assets 1035
    bin 1115
    lib 1035
    res 1108
    AndroidManifest.xml 1111
    CHANGELOG 1035
    LICENSE 1076
    project.properties
```

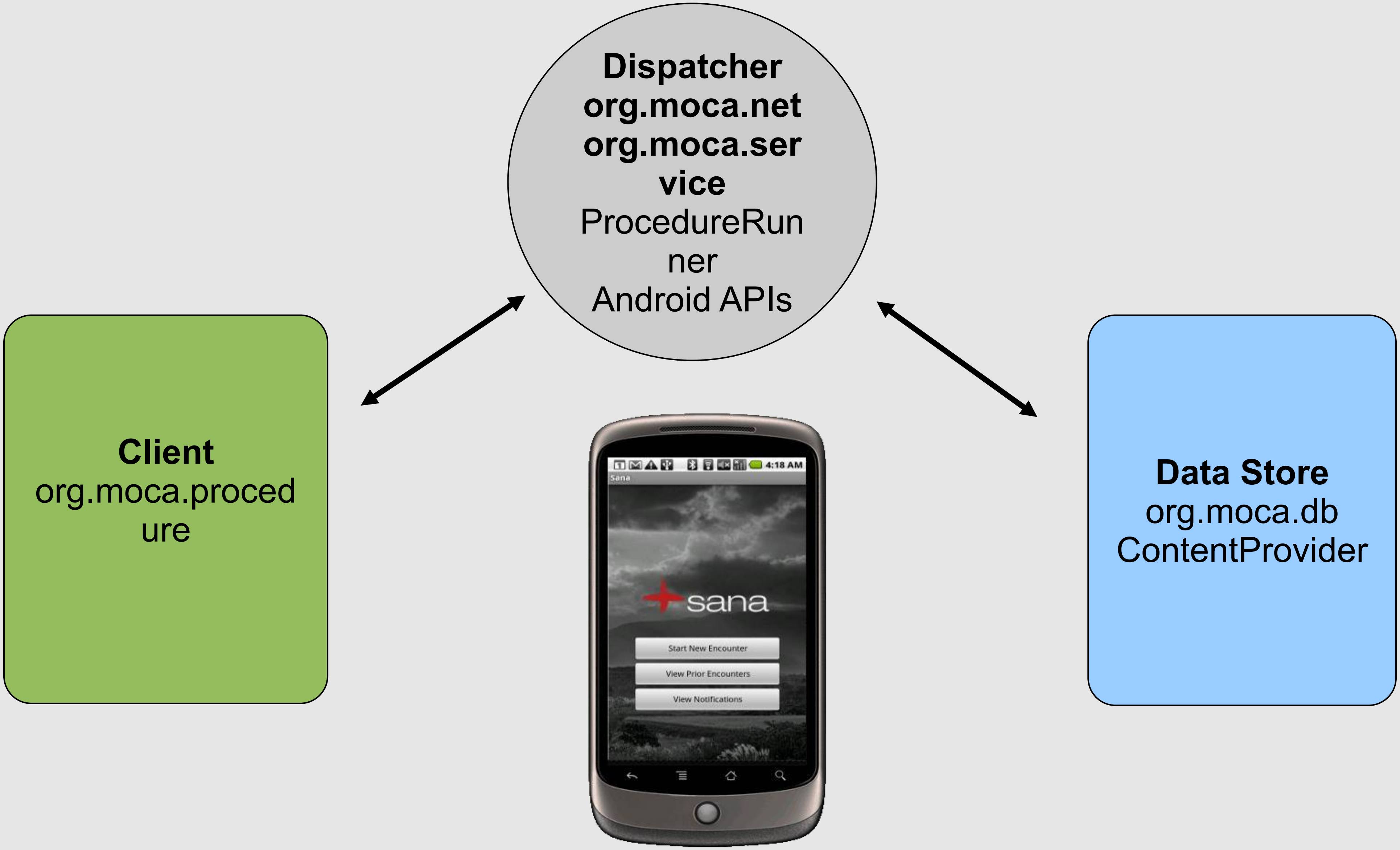
welcome.3gp - sana-release-1.1.x

Courtesy of Eclipse Foundation. Used with permission.

Sana Mobile Client



Sana Mobile Client



Data Layer

org.moca.db: content providers and structural information of the application database-table columns, content URIs, data access objects, etc.

Additional utility classes in the org.moca.util package which encapsulate some of the functionality-clearing and reloading the entire db, etc.

Dispatch Layer

org.moca.net: Network components. Client calls to upstream server.

org.moca.service: Wrappers around the network calls.

ProcedureRunner: Provides the stepwise flow through the instruction sets.

Android APIs: Calls to Android OS components and services.

Client Layer

org.moca.procedure: Classes which represent the objects contained within and user view of the xml based procedures

```
Procedure
ProcedurePage
ProcedureElement
```

org.moca.procedure.branching: Conditional branching logic and operators.

```
    and, or, greater than, less than, equal
```