Open Source Telemedicine

Android Client Development

Introduction

Images of phone in this presentation © Google. All rights reserved. This content is excluded from our Creative Commons license. For more information, see http://ocw.mit.edu/fairuse.
Open Source Telemedicine

Development Environment Requirements

Android Client
Resources

Best source of information is the Android development site:

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

SDK Components

AVDs
Open Source Telemedicine

Hello Android
Android Application Structure

- **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

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

Courtesy of Eclipse Foundation. Used with permission.
Android Application Structure

Layouts

 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
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

Declares the applications capabilities
**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”
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.
Use the Android logging capabilities

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

import android.app.Activity;

public class HelloActivity extends Activity {

    public HelloActivity() {

        // Called with the activity is first created.
        super.onCreate(savedInstanceState);

        setContentView(R.layout.hello_activity);
        Log.d(HelloActivity.class.getSimpleName(), "Hello Log.");
    }

    // Some sample code for the activity.
    ...
}
```

Log levels

Tagged

Courtesy of Eclipse Foundation. Used with permission.
Debugging

Use the Android logging capabilities

android.util.Log

- Log levels
- Tagged

Logcat is your friend!

Courtesy of Eclipse Foundation. Used with permission.
Open Source Telemedicine

Hello Sana
Sana Application Structure

 Courtesy of Eclipse Foundation. Used with permission.
Sana Mobile Client

Dispatcher
Network Layer
Procedure
Engine
Android APIs

Client
UI Layer

Data Store
SQLite
Sana Mobile Client

Dispatcher
org.moca.net
org.moca.service
ProcedureRunner
Android APIs

Client
org.moca.procedure

Data Store
org.moca.db
ContentProvider
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

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

Procedure
ProcedurePage
ProcedureElement

\texttt{org.moca.procedure.branching}: Conditional branching logic and operators.

and, or, greater than, less than, equal