HW#9 Clues

CSCI 571
Spring, 2016

HW#9 Prototype

- YouTube Link : https://youtu.be/jNoZTuf2VHI

Tutorials

- **1. Building your first App**

Creating a Project with Android Studio

http://developer.android.com/training/basics/firstapp/creating-project.html

- **2. Running your first App**

http://developer.android.com/training/basics/firstapp/running-app.html

(on same page see also “Run on the Emulator”)

Tutorials

- **3. Starting another activity**

http://developer.android.com/training/basics/firstapp/starting-activity.html

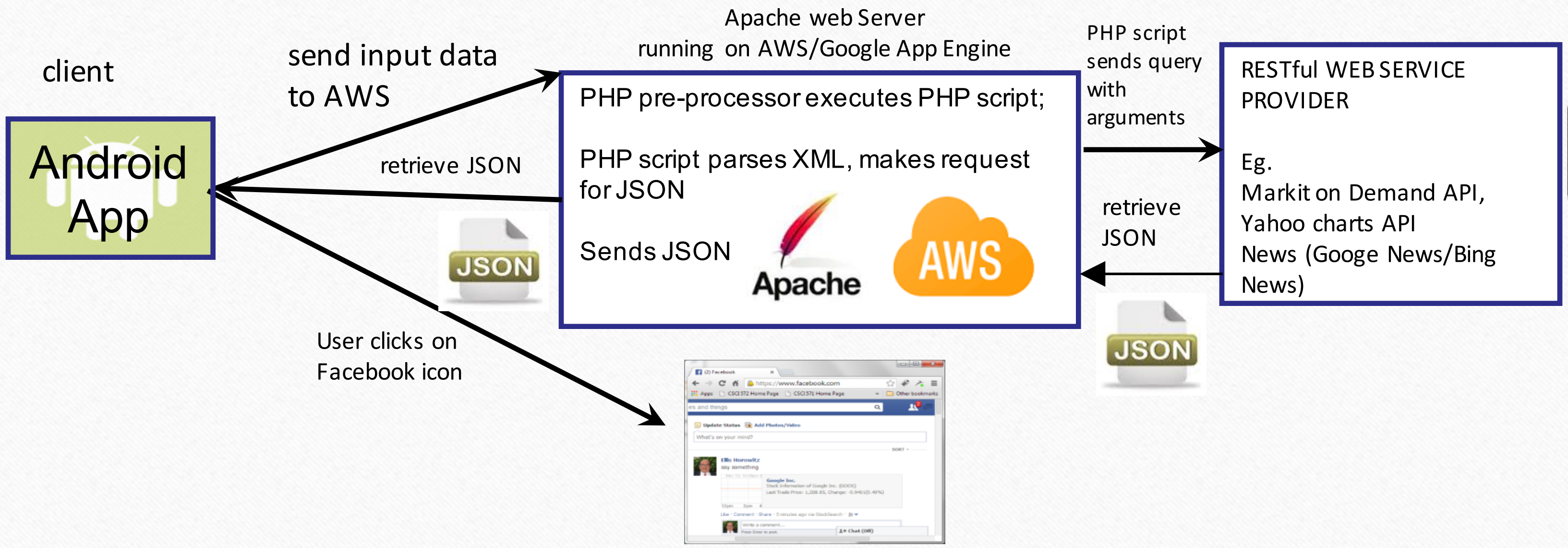
- **4. Comprehensive Tutorial / Article on getting started with Android**

http://www.vogella.com/tutorials/Android/article.html

What is needed

- You will need to download and install Android Studio
  https://developer.android.com/sdk/index.html
- Download the Facebook Android SDK
  https://developers.facebook.com/docs/android/getting-started
- Register your new App with Facebook and get an Application ID
  https://developers.facebook.com/apps/

HW#9 Architecture Overview


client
Android App
send input data to AWS
retrieve JSON
JSON
User clicks on Facebook icon
Apache web Server running on AWS/Google App Engine
PHP pre-processor executes PHP script;
PHP script parses XML, makes request for JSON
Sends JSON
Apache
AWS
PHP script sends query with arguments
retrieve JSON
JSON
RESTful WEB SERVICE PROVIDER
Eg.
Markit on Demand API,
Yahoo charts API
News (Googe News/Bing News)

Posting to Facebook

HW9 Implementation

- You will create 2 activities and a Manifest file
- **AndroidManifest.xml**
- **MainActivity.java** – routine that controls the entire process
  - Creates the Search Form UI
  - sets on-click handlers to the buttons
  - Validateion the user input
  - Calls AWS/Google App Engine server

HW9 Implementation

- **ResultActivity.java**
  - Creates the Tabbed Layout – Current, Historical and News
  - Facebook Share
  - Add to favorites

AndroidManifest.xml File

Every application must have an AndroidManifest.xml file in its root directory. The manifest presents essential information about the application to the Android system. Among other things, the manifest does the following:

- It names the Java package for the application.
- It describes the components of the application — the activities, services, broadcast receivers, and content providers that the application is composed of.
- It names the classes that implement each of the components and publishes their capabilities.

AndroidManifest.xml File

See http://developer.android.com/guide/topics/manifest/manifest-intro.html.

Please note that the file is created by default on creation of a new Android project using Android Studio IDE.

UI Controls in Android

For the homework exercise, you can use the following widgets (not limited to):

- **TextView** (i.e., label)
  http://developer.android.com/reference/android/widget/TextView.html
- **EditText** (i.e., text field)
  http://developer.android.com/reference/android/widget/EditText.html
- **AutoCompleteTextView**
  http://developer.android.com/reference/android/widget/AutoCompleteTextView.html

UI Controls in Android (Contd.)

For the homework exercise, you can use the following widgets (not limited to):

- **Button** http://developer.android.com/reference/android/widget/Button.html
- **RadioButton** http://developer.android.com/reference/android/widget/RadioButton.html
- **ImageView** http://developer.android.com/reference/android/widget/ImageView.html
- **Switch** **http://developer.android.com/reference/android/widget/Switch.html**
- **ProgressBar** **http://developer.android.com/reference/android/widget/ProgressBar.html**

UI Controls in Android (Contd.)

- **ListView** http://developer.android.com/reference/android/widget/ListView.html
- TabLayout http://developer.android.com/reference/android/support/design/widget/TabLayout.html
- ViewPager http://developer.android.com/reference/android/support/v4/view/ViewPager.html
- WebView http://developer.android.com/reference/android/webkit/WebView.html

UI Controls in Android (Contd.)

- **ImageSwitcher (**It is useful to animate an Image on screen)
  http://developer.android.com/reference/android/widget/ImageSwitcher.html
- **TextSwitcher** (It is useful to animate a label on screen)

  http://developer.android.com/reference/android/widget/TextSwitcher.html
- **TableLayout**
  http://developer.android.com/reference/android/widget/TableLayout.html

UI Controls in Android (Contd.)

- **TableRow** http://developer.android.com/reference/android/widget/TableRow.html
- **RelativeLayout** http://developer.android.com/reference/android/widget/RelativeLayout.html
- **LinearLayout** (It arranges "components" in vertical or horizontal order, via orientation attribute.) http://developer.android.com/reference/android/widget/LinearLayout.html
- **ScrollView** http://developer.android.com/reference/android/widget/ScrollView.html

RelativeLayout

RelativeLayout lets you position your component base on the nearby (relative or sibling) component’s position. You can use “above, below, left and right” to arrange the component position.

LinearLayout

LinearLayout is a common layout that arranges “component” in vertical or horizontal order, via *orientation* attribute

MainActivity.java

onCreate does the following

•Display the search UI – AutoCompleteTextView and 2 Buttons.

•Render the favorites list(ListView), if any stocks have been marked as favorites.

•Register a event handler with the various UI components– ‘Get Quote’ (Button), ‘Clear’ (Button), ‘Refresh’ (ImageButton) and ‘Auto-Refresh’ (Switch).

MainActivity.java (Contd.)

onClickListener for the ‘Get Quote’ Button does the following

•Validate the user input

•Build the URL for AWS/Google App Engine

•Create a new task which is of type AsyncTask to fetch the JSON data. It will initiate an asynchronous call.

•Execute the task and moves to another activity, if needed!

MainActivity.java (Contd.)

onClickListener for the ‘Clear’ Button does the following

•Clear the AutoComplete Text View

onClickListener for the ‘Refresh’ ImageButton does the following

•Get the updated stock information of all the favorite stocks and re-render the favorite ListView

MainActivity.java (Contd.)

onClickListener for the 'AutoRefresh' Switch does the following

•If auto-refresh in OFF, Start getting the favorites stock's information every 10 seconds and re-renders the favorite list, ListView. You may use Runnable and Handler for repeatedly run a 'task'

•If auto-refresh in ON, you need to stop getting the stop information

MainActivity.java – Remove Favorite

Swiping gesture – Deleting the favorite

- You can swipe left or right, to initiate the process of removing the stock from the favorite list.
- You then need to confirm for the deletion
- If the user confirms the deletion, you should remove the stock from the Favorite ListView

MainActivity.java - AsyncTask

AsyncTask is an abstract class provided by Android which helps us to use the UI thread properly. This class allows us to perform long/background operations and show its result on the UI thread without having to manipulate threads.

AsyncTask has four steps:

doInBackground: Code performing long running operation goes in this method. When onClick method is executed on click of button, it calls execute method which accepts parameters and automatically calls doInBackground method with the parameters passed.

MainActivity.java – AsyncTask (Contd.)

onPostExecute: This method is called after doInBackground method completes processing. Result from doInBackground is passed to this method.

onPreExecute: This method is called before doInBackground method is called.

onProgressUpdate: This method is invoked by calling publishProgress anytime from doInBackground call this method.

ResultActivity.java

ResultActivity starts with onCreate

- onCreate does the following

- retrieves JSON data which was passed from MainActivity
- stores the data in a JSONObject
- Settting up the three tabs – ‘Current’, ‘Historical’ and ‘News’
- Rendering the ‘bookmark’ button properly, if the stock had been marked as favorite.

ResultActivity.java - Favorites

ResultActivity would also be responsible for marking a stock as favorite

- Get the selection stock information such as stock name, stock symbol, etc.
- Save the above information in SharedPreference
- Use a ‘suitable string’ key for storing your favorite stock
- You may store your array of favorite stock in the JSON format
- You may also check if the stock is favorite or not, in a similar fashion – Iterate over your array of favorite stock and check if the current stock is marked as favorite or not.

ResultActivity.java - Current

Current Tab would be doing the following:

1. Get the stock information from our script hosted on AWS/Google App Engine
2. Format the values in the desired format. For example, date needed to be in the specified format.
3. Render the ListView with the stock information with the data we got from our script.
4. Also below the listview, we have an ImageView showing the current stock activity.
5. Furthermore, on clicking the ImageView, you need to render the stock activity image in a ‘Dialog’. The image in the dialog should be zoomable.

ResultActivity.java - Historical

Historical Tab would be doing the following:

1. Render the WebView with a local HTML file
2. The HTML would be responsible for rendering the HighChart UI.

ResultActivity.java - News

News Tab would be doing the following:

1. Get the news information from our script hosted on AWS/Google App Engine
2. Render the ListView with the news information with the data we got from our script.

FACEBOOK POST

For the latest versions of Facebook SDK 4.x, share functionality may require following:

- **Modifications in AndroidManifest file:**
  - Introducing Fb Application Id

```
<meta-data android:name="com.facebook.sdk.ApplicationId" android:value="@string/facebook_app_id"/>
```

  - Adding FacebookActivity

```
<activity android:name="com.facebook.FacebookActivity"

          android:configChanges= "keyboard|keyboardHidden|screenLayout|screenSize|orientation"

          android:theme="@android:style/Theme.Translucent.NoTitleBar”  android:label="@string/app_name" />
```

FACEBOOK POST (Contd.)

For the latest versions of Facebook SDK 4.x, share functionality may require following:

- **Modifications in AndroidManifest file:**
  - Adding Facebook Content Provider

```
<provider android:authorities="com.facebook.app.FacebookContentProvider{app_id}"
    android:name="com.facebook.FacebookContentProvider” android:exported="true"/>
```

FACEBOOK POST (Cont.)

- To implement the functionality you may use the following approach on click of the fb button: https://developers.facebook.com/docs/sharing/android
  - Initialize facebook sdk
  - Create a ShareDialog
  - Create LinkContent for the post
  - Share the LinkContent through ShareDialog
  - Register Callback for the ShareDialog
  - Bind onActivityResult for maintaining session

Android Libraries

You may find the below libraries useful to implement the features:

- Google Gson : https://github.com/google/gson
- Apache Commons Lang : https://commons.apache.org/proper/commons-lang/

Android Libraries

You may find the below libraries useful to implement the features:

- Nhaarman library for ListView animations : https://github.com/nhaarman/ListViewAnimations
- Chrisbanes library for zoomable ImageView : https://github.com/chrisbanes/PhotoView