CIS 4515

Worksheet 9

Instructions:

Create a web browser that can have multiple “tabs”. Each page will be loaded by taking a user supplied URL, fetching the (HTML) data from the specified URL, and then loading the information in a WebView. Your application must use fragments, with each fragment containing its own WebView. You will use the ActionBar, and it will contain (at least) 3 buttons: **New**, **Previous**, and **Next**. Clicking the New button creates a new tab (browser fragment) and places it on top, awaiting the user's URL. The Previous and Next buttons will cycle through the previously created browser fragments (this should not be confused with the Back and Forward buttons on a regular browser that goes to the next or previous page being viewed in a specific window).

Since you will need to navigate back and forth among your fragments, you will not be able to rely solely on the FragmentManager (the fragment manager operates as a stack, which means you would lose fragments when they were popped). You will need to implement some other data structure to assist in the management of your fragments. This structure can be implemented in various ways, such as an **ArrayList** (e.g. ArrayList<Fragment>) or, more ideally, a **FragmentStatePagerAdapter** (https://developer.android.com/reference/android/support/v4/app/FragmentStatePagerAdapter.html) The latter allows you to easily implement swipe-to-switch among your browser fragments (https://developer.android.com/training/implementing-navigation/lateral.html). At the end of the day, what you will need is a data structure that will be able to keep track of all fragments, regardless of which fragment(s) are currently attached to the activity.

1. Familiarize yourself with the WebView widget (http://developer.android.com/reference/android/webkit/WebView.html)

2. Add the following permission request line to your AndroidManifest.xml as a child of the <manifest> tag as follows:

   ```
   <manifest xmlns:android="..." package="...">

      <uses-permission android:name="android.permission.INTERNET"/>
   ```

   This will allow your application to access the internet (it’s a web browser after all).

3. Design the layout for your main **activity** to contain an EditText (to get the URL), and a Button (the *Go* button) aligned at the top of the screen as in a traditional web browser.

   The EditText will allow the user to enter a URL, the button will be used to initiate the fetch process

4. Your browser fragments should be designed as follows:

1. Create a layout for your fragment containing a WebView.

   The WebView will be used to display the downloaded content.

2. When the user clicks the Go button in the activity, load the URL typed into the activity's EditText in the fragment's WebView object using its loadURL() method.

3. When the user switches from one fragment to another (fragment1 to fragment2), fragment1 must remember its URL (if it had one) and it should be displayed in the activity's EditText if we navigate back to fragment1. It's OK if fragment1's WebView has to be reloaded, but the URL should be remembered and displayed to the user in the activity, and the webview reload should be done automatically (i.e. the user should not have to click Go a second time).

5. Finally, your browser must have the ability to respond to implicit URL view requests from other (third party) applications.

   1. Add an <intent-filter> to your activity's declaration in the android Manifest to accept implicit intents for viewing **http** and **https** URLs.

   2. In the onCreate() method of the activity that receives the implicit intent, you can access the URL to be viewed as follows:

      ```
      String url = getIntent().getData().toString();
      ```

   3. Once retrieved, automatically load a new fragment (tab) and display the URL.

6. **Create and upload a signed APK along with an archive of your Android Studio project (or a GitHub link) to Blackboard.**

   Considerations:

   - You can enable JavaScript in your browser by making a call to:

     ```
     webView.getSettings().setJavaScriptEnabled(true);
     ```