An Android Homepage Widget

SEM2220 Assignment 3

Alexander D Brown (adb9)

Contents

1 Introduction

This report details the process undertaken to produce an Android widget based on the existing code to load sessions from a SQLite database. This widget had several requirements, including the ability to select different days from the database as well as provide notifications read from a remote URL.

2 Design

The widget was designed in accordance with the Android App Widget Design Guidelines[1], which define guides for design constraints like the minimum and maximum size of the widget, layouts and backgrounds, etc.

To help conform to these guidelines, the template design pack[2] provided by the Android Open Source Project under the Apache 2 License was used to design the widget.

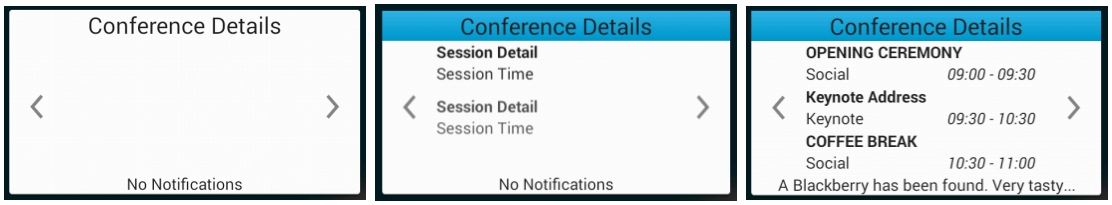
A mock version of the widget was created to view how it would appear on a device. From this is became obvious that the widget would need to be four cells wide (the maximum) by at least two cells tall.

This size would allow a view with the following information on it:

- A title for the Widget
- Two buttons, one to move backwards through days and one to move forward through days
- A list of the sessions available for the specified day.
- The notification loaded from a remote site.

To keep the buttons accessible, they were made such that their minimum size was a single cell each and surrounded the list of sessions to make the flow of data natural. To conform to the iconography standards[3], another resource was used from the Android Open Source Project; the Action Bar Icon Pack[4].

The notification display was kept small so that it would not obstruct the view of the data, but so that it would be easy to see at a glance. Finally, the title was given colour, based on the recommended colours[5], a purely aesthetic element.



(a) Initial Widget Design (b) Improved Widget Design (c) Final Widget Design

Figure 1: Evolution of the Widget Design

Figure 1 shows the evolution of the design for the widget.

3 Development

Most of the development was based on the documentation of Android App Widget[6]. Following this resource, the API and some examples from both the Android Open Source Project and

textbook codes samples[7]. With these resources it was a fairly simple matter to implement the basic widget.

All the development for the widget was kept in a separate Android project. The first action was to import the `DataAccess` class provided into this project. The next step was to create a class which implemented `AppWidgetProvider`. This used the layout shown in the previous section and a `RemoteViews` object to manipulate the elements in that layout.

To enable the loading of list content, subclasses `RemoteViewsService` and `RemoteViewsFactory` were created. The service simply provides this factory, whilst the service loads content into the list from the SQLite3 database using the `DataAccess` class.

This brought forward a problem; the SQLite3 table row IDs defined in `ConferenceCP` did not match the actual SQLite3 row IDs provided. This lead to incorrect data being provided by the `DataAccess` class. Fixing this was a simple matter of changing these IDs to match the right IDs, but this may not apply to other implementations of SQLite3 (the version exported works in SQLite version 3.7.11, on the Android Emulator running on Linux Mint 14 Cinnamon 64-bit).

To improve the loading of Sessions, a POJO[8] was created to represent a session and an additional method for loading this POJO was added into the `DataAccess` class. The same thing was created to load a Date object from the database as the Session requires this. With these POJOs it was easy to display this in the list rows.

The next task was to implement the next day and previous day functionality. Initially the code for this was doing the widget provider. The buttons would send a broadcast (either next or previous) to the provider and then to forcibly call the `onUpdate` method to reset the broadcasts and list service.

However, talking to a fellow classmate (Gareth Williams) the easier way to do this was to send a broadcast to the service directly and just notify the App Widget Manager that the data for the list had changed.

Finally, to download the notification an `AsyncTask` was implemented to handle the URL connection as it cannot be done in the Android main thread. The code for this is fairly standard and the correct permissions had to be set in the manifest file.

4 Testing

All testing was performed using the Android emulator. Most of this testing involved using the widget to ensure it worked correctly. Actually unit testing the code would have been fairly trivial as it only really integrated existing code and displayed it in layouts.

5 Evaluation

Breaking down the mark scheme the author has predicted the grade which should be given for each part, this is shown in table 1.

Therefore, the author feels a mark of 82% should be awarded. The values chosen were based on the following reasons:

Documentation

Implementation

Flair

Testing

Part	Worth	Predicted Grade
Documentation	30%	25%
Implementation	50%	45%
Flair	10%	5%
Testing	10%	7%
Total	100%	82%

Table 1: Break Down of Marks

References

[1] App Widget Design Guidelines. Android Open Source Project. Accessed 10/12/2013. [Online]. Available: http://developer.android.com/guide/practices/ui_guidelines/widget_design.html

[2] App Widget Templates Pack for Android 4.0. Android Open Source Project. Downloaded 7/12/2013, Apache 2 Licensed. [Online]. Available: http://developer.android.com/shareables/app_widget_templates-v4.0.zip

[3] Iconography. Android Open Source Project. Accessed 10/12/2013. [Online]. Available: http://developer.android.com/design/style/iconography.html

[4] Action Bar Icon Pack. Android Open Source Project. Downloaded 7/12/2013, Apache 2 Licensed. [Online]. Available: http://developer.android.com/downloads/design/Android_Design_Icons_20131106.zip

[5] Colour. Android Open Source Project. Accessed 10/12/2013. [Online]. Available: http://developer.android.com/design/style/color.html

[6] App Widgets. Android Open Source Project. Accessed 10/12/2013. [Online]. Available: https://developer.android.com/guide/topics/appwidgets/index.html

[7] M. L. Murphy, *The Busy Coder's Guide to Android Development*. CommonsWare, 6th Febuary 2009, vol. 5.4, code Samples available: https://github.com/commonsguy/cw-advandroid.

[8] M. Fowler. (2000, September) POJO. [Online]. Available: http://www.martinfowler.com/bliki/POJO.html