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.

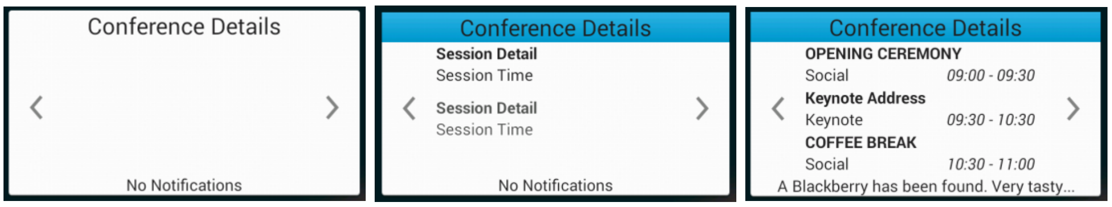
A mock version of the widget was created to view how it would appear on a device. From this is became obvious that the widget would need to be four cells wide (the maximum) by at least two cells tall.

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

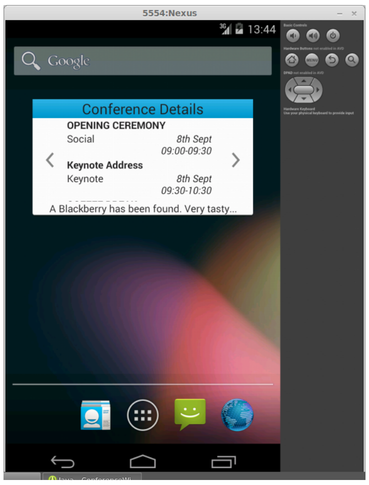
All testing was performed using the Android emulator. Most of this testing involved using the widget to ensure it worked correctly. Actually unit testing the code would have been fairly trivial as it only really integrated existing code and displayed it in layouts.

All figures below show the widget running on an emulated Nexus 4 at API level 19 (Android 4.4). The widget should also function on any API level 14 and above (Android 4.X). This is a requirement for the loading of the list.

Figure 2 shows the widget running on the emulator displaying information for all sessions on the three days.

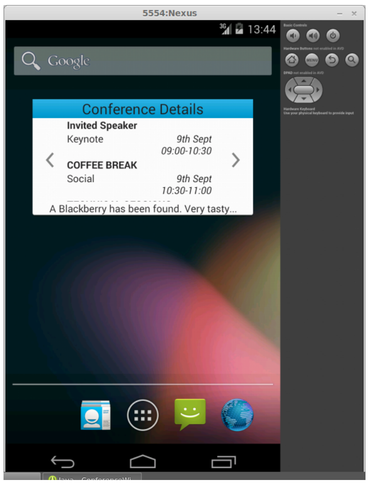
Figure 3 shows the widget's ability to resize vertically to show more information, depending on the user's available space.

Figure 4 shows that the widget will dynamically load a different notification from the remote URL.



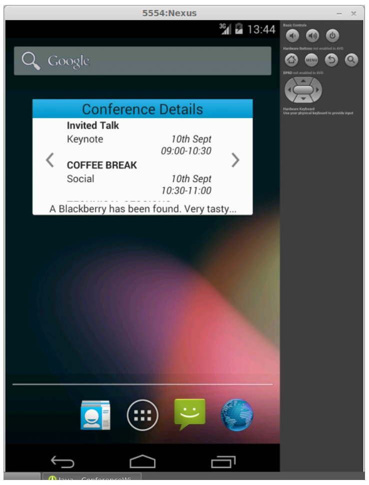
OPENING CEREMONY
Social 8th Sept 09:00-09:30
Keynote Address
Keynote 8th Sept 09:30-10:30
A Blackberry has been found. Very tasty...

(a) The first day



Invited Speaker
Keynote 9th Sept 09:00-10:30
COFFEE BREAK
Social 9th Sept 10:30-11:00
A Blackberry has been found. Very tasty...

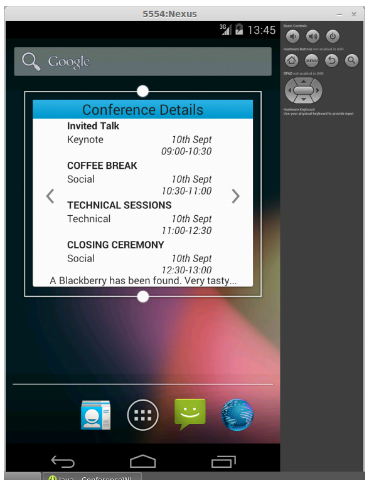
(b) The second day



Invited Talk
Keynote 10th Sept 09:00-10:30
COFFEE BREAK
Social 10th Sept 10:30-11:00
A Blackberry has been found. Very tasty...

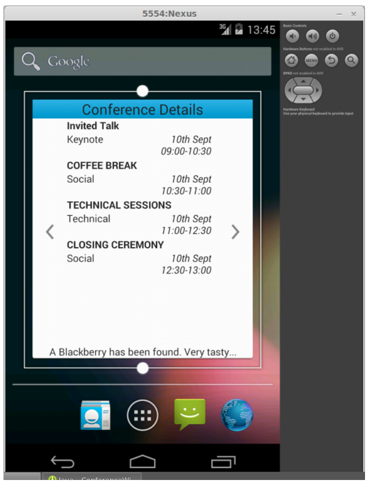
(c) The third day

Figure 2: The Widget displaying session information for the three different days



Invited Talk
Keynote 10th Sept 09:00-10:30
COFFEE BREAK
Social 10th Sept 10:30-11:00
TECHNICAL SESSIONS
Technical 10th Sept 11:00-12:30
CLOSING CEREMONY
Social 10th Sept 12:30-13:00
A Blackberry has been found. Very tasty...

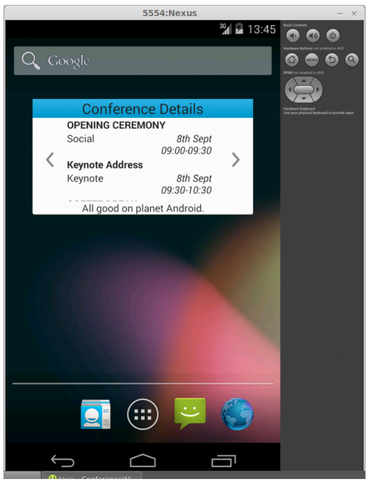
(a) Resizing to 3 by 4 cells



Invited Talk
Keynote 10th Sept 09:00-10:30
COFFEE BREAK
Social 10th Sept 10:30-11:00
TECHNICAL SESSIONS
Technical 10th Sept 11:00-12:30
CLOSING CEREMONY
Social 10th Sept 12:30-13:00
A Blackberry has been found. Very tasty...

(b) Resizing to 4 by 4 cells

Figure 3: Resizing the widget to show more information



OPENING CEREMONY
Social 8th Sept 09:00-09:30
Keynote Address
Keynote 8th Sept 09:30-10:30
All good on planet Android.

Figure 4: Proving that the widget will load a new notification.

5 Evaluation

All of the feature requests have been implemented and evidenced as per the figures shown in the previous section. The code is of decent quality and conforms to the Android developer guidelines.

The author has learned about the creation of Android homescreen widgets, having successfully implemented one, and has found the process to be challenging but not unduly difficult. One does wonder why the Android API has such good support for Activities and Services, but how the implementation of widgets feels somewhat mixed, surely a base class like Activity could have been provided to handle widgets rather than directly manipulating remote views.

A lot of the time spent on this project was spent understanding this architecture and figuring out how to handle certain events correctly. It does have the feeling that the actions to go left and right were not the kind of operation that widgets are meant to perform. But trying to implement this in a different way, by using a stacked view, lead to the problem that the lists could not be generated correctly inside another view.

The amount of documentation for widgets is limited, so the example found in the book[7] were a lot of help.

Aside from this, the normal numbers of error produced by either null pointers or going beyond lists were encountered and fixed, though it would have been nice to debug the code using a debugger, but despite numerous attempts it would not halt at breakpoints, despite the debugger being correctly attached. The author would guess that widgets are so tied to the homescreen that it is difficult for the Android Operating System to unlink them.

Loading from a remote URL was a task the author has already investigated so did not provide many problems, although the initial use of a `HttpURLConnection` and a `BufferedReader` did not combine well and always resulted in empty content, despite the result being `HTTP 1.1/200 OK`. The solution was to simplify this and use the simpler `URLConnection` and `InputStreamReader`, manually setting the buffer size to 255; a value which should not take up too much RAM of the device, but which should be enough to load the data quickly.

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 86% should be awarded. The values chosen were based on the following reasons:

Documentation This report conveys a detailed story of the implementation of code, problems encountered, lessons learned and this indication of the mark which should be awarded.

Implementation The design and code meets the specified functionality and conforms to the Android development guidelines.

Flair The code has been implemented in a maintainable and proper manner, especially in using POJOs to handle database interaction. The use of a list to display sessions is also a complex feature which the specification did not ask for.

Testing Evidence of testing on the Android emulator has been provided.

Part	**Worth**	**Predicted Grade**
Documentation	30%	27%
Implementation	50%	45%
Flair	10%	7%
Testing	10%	7%
Total	100%	86%

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