**ANDROID LAYOUT CUSTOMISATION**

**By**

**Pascal How**

**14 Dec 2015**

Table of Contents

[1. Introduction 4](#_Toc437795176)

[2. Software Development Duration 4](#_Toc437795177)

[3. Splash Screen 4](#_Toc437795178)

[4. Menu Items 5](#_Toc437795179)

[4.1. Camera 5](#_Toc437795180)

[4.1.1. Design Decisions 5](#_Toc437795181)

[4.1.2. Libraries 6](#_Toc437795182)

[4.1.3. Expected Result 6](#_Toc437795183)

[4.1.4. Actual Result 6](#_Toc437795184)

[4.2. Gallery 7](#_Toc437795185)

[4.2.1. Design Decisions 7](#_Toc437795186)

[4.2.2. Libraries 7](#_Toc437795187)

[4.2.3. Expected Result 7](#_Toc437795188)

[4.2.4. Actual Result 7](#_Toc437795189)

[4.3. Slideshow 8](#_Toc437795190)

[4.3.1. Design Decisions 8](#_Toc437795191)

[4.3.2. Libraries 8](#_Toc437795192)

[4.3.3. Expected Result 8](#_Toc437795193)

[4.3.4. Actual Result 8](#_Toc437795194)

[4.4. Settings 9](#_Toc437795195)

[4.4.1. Design Decisions 9](#_Toc437795196)

[4.4.2. Libraries 9](#_Toc437795197)

[4.4.3. Expected Result 9](#_Toc437795198)

[4.4.4. Actual Result 9](#_Toc437795199)

[4.5. Share 10](#_Toc437795200)

[4.5.1. Design Decisions 10](#_Toc437795201)

[4.5.2. Libraries 10](#_Toc437795202)

[4.5.3. Expected Result 10](#_Toc437795203)

[4.5.4. Actual Result 10](#_Toc437795204)

[4.6. Send 10](#_Toc437795205)

[4.6.1. Design Decisions 10](#_Toc437795206)

[4.6.2. Libraries 11](#_Toc437795207)

[4.6.3. Expected Result 11](#_Toc437795208)

[4.6.4. Actual Result 11](#_Toc437795209)

[5. About Me 11](#_Toc437795210)

[6. Floating Action Button 11](#_Toc437795211)

[7. App Evaluation 12](#_Toc437795212)

[7.1. Strengths 12](#_Toc437795213)

[7.2. Weaknesses 12](#_Toc437795214)

# Introduction

The aim of this application is to design and develop the layout as well as the functionalities associated with each item located in the side bar menu. The menu items are as follows:

1. Camera
2. Gallery
3. Slideshow
4. Settings
5. Share
6. Send

Each menu item is associated with a specific view fragment which is overlayed onto the main activity view upon selection.

The app has been tested on a nexus 5 running Android Marshmallow.

# Software Development Duration

Brainstorming – 2 hours

Software development – 10 hours

Researching – 5 hours

Documentation – 3 hours

# Splash Screen

|  |  |
| --- | --- |
|  | * The Splash screen activity is called when the app is launched * Splash screen consists of the main logo required to brand the app -> logo.png * Splash screen will display for 3 seconds after which, the main page is launched * The app colour theme is also set to #186eac (blue) |

# Menu Items

|  |  |
| --- | --- |
|  | * The menu items can be accessed by tapping on the hamburger icon in the top left corner * The menu items consist of:  1. Camera 2. Gallery 3. Slideshow 4. Settings 5. Share |

## Camera

|  |  |  |  |
| --- | --- | --- | --- |
|  | C:\Android\MyDrive\code-tests-master\Screenshot_20151213-162914.png | C:\Android\MyDrive\code-tests-master\Screenshot_20151213-162928.png | C:\Android\MyDrive\code-tests-master\Screenshot_20151213-162951.png |

### Design Decisions

1. The primary aim was to use the built in android camera app to take pictures and save it into a local directory. When selecting the camera item, the user is greeted with the message “Take a picture”
2. To separate images taken with this application from other images in the user’s camera directory, a new folder called JuniorTest was created.
3. Upon saving the image, the app would then display the image bitmap in the ImageView of the camera fragment. Initially, getting the image from OnActivityResult(…) would return a very small bitmap. To display a larger bitmap, the image had to be saved to a local directory first and then decoded to return a larger sized bitmap. The larger sized bitmap could then be used for the image view.
4. To use these functionalities, the following app permissions are required: camera, read external storage and write to external storage. As of Android Marshmallow, run time app permissions are required and the user has to manually enable each of them

|  |  |
| --- | --- |
| C:\Users\pascalh\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\Screenshot_20151213-170941.png | * Requesting for camera permission * If all three permissions are not granted, the user will not be allowed to use the camera functionality of the application |

### Libraries

The main libraries used are:

1. Android.graphics.Bitmap
2. Android.net.Uri
3. Android.os.Environment – Read and write to files
4. Android.content.pm.PackageManager – For app permissions

### Expected Result

Camera item launches the camera application, allows the user to take a photo and displays it onto the image view in the camera fragment.

### Actual Result

Functionality performed as expected.

## Gallery

|  |  |
| --- | --- |
| C:\Android\MyDrive\code-tests-master\Screenshot_20151213-163007.png | * The gallery item fetches a series of images online based on their url and displays them as a list in a RecyclerView * The user can scroll through the list to view the images |

### Design Decisions

1. The images are loaded as a list in the RecyclerView and allows the user to scroll through them
2. The images are loaded from the internet. In order to minimise downloading time and resources, the images are saved in the cache so that they can be displayed faster the next time the camera fragment is launched.
3. For the purpose of this coding test, an extra image has been added to the list. The image can be found at: http://41.media.tumblr.com/685d13522aad72841285a0334eefc162/tumblr\_nwhoqdtNst1qkq7k9o1\_1280.jpg

### Libraries

1. Android.support.v7.widget.RecyclerView

### Expected Result

1. All the images are loaded into the image list
2. User can scroll smoothly through the images

### Actual Result

Functionality performed as expected.

## Slideshow

|  |  |
| --- | --- |
| C:\Android\MyDrive\code-tests-master\Screenshot_20151213-163024.png | * The slideshow item continuously displays all the images in the JuniorTest folder every second * If no images are found, no slideshows will be displayed |

### Design Decisions

1. A separate thread with a time interval of 1 second was used to display the images from the JuniorTest folder.
2. Similar to the camera fragment, the bitmaps are decoded first and then displayed onto an image view.
3. The image paths are loaded into a list and the thread continuously runs through the list to display the images.
4. No button was implemented for this functionality and the slideshow starts the moment the slideshow fragment is launched. This is because it is not necessary for a slideshow to require user interactions.

### Libraries

1. Android.graphics.Bitmap
2. Android.os.Environment – Read and write to files

### Expected Result

1. Slideshow to launch when the slideshow fragment is loaded.
2. Images rotate every second
3. All images from the JuniorTest folder should take turn to display

### Actual Result

Functionality performed as expected.

## Settings

|  |  |  |
| --- | --- | --- |
| C:\Android\MyDrive\code-tests-master\Screenshot_20151213-163058.png | C:\Android\MyDrive\code-tests-master\Screenshot_20151213-163107.png | C:\Android\MyDrive\code-tests-master\Screenshot_20151213-163116.png |

### Design Decisions

1. User to tap on Settings button to load a settings page from which he can configure his user preferences
2. Settings page to consist of basic items such as username, email address, send report and sync frequency
3. Tapping on each settings item should allow the user to input text or select option settings
4. Implementing functionalities for each of these item is not required at this stage
5. The changes should however be obvious by outputting them to a string and then displayed in a text view on the settings fragment

### Libraries

1. Android.preference.PreferenceManager

### Expected Result

Tapping the Settings button allows the user to enter:

1. A username
2. Email address
3. Whether to send reports via a checkbox
4. Sync frequency by selecting from a list of options

The user settings are then displayed as text in the settings fragment

### Actual Result

Functionalities performed as expected.

## Share

|  |  |  |
| --- | --- | --- |
| C:\Android\MyDrive\code-tests-master\Screenshot_20151213-163150.png | C:\Android\MyDrive\code-tests-master\Screenshot_20151213-163404.png | C:\Android\MyDrive\code-tests-master\Screenshot_20151213-163419.png |

### Design Decisions

1. Launching the share fragment displays a list of images from the JuniorTest folder
2. User can select images from the list as attachment
3. User can choose to share via the apps already installed on his device for e.g. email, whatsapp

### Libraries

1. Android. Support.v4.app.ListFragment
2. Android.graphics.Bitmap
3. Android.os.Environment

### Expected Result

1. All the images from the JuniorTest folder are loaded into a list view
2. User can select an image as attachment and send via apps already installed on his device
3. The recipient receives the image as well as any text associated with the image such as title or caption

### Actual Result

Functionality performed as expected.

## Send

This menu item was removed as it was considered very similar to the share functionality

### Design Decisions

N/A

### Libraries

N/A

### Expected Result

N/A

### Actual Result

N/A

# About Me

|  |  |
| --- | --- |
| C:\Android\MyDrive\code-tests-master\Screenshot_20151213-163441.png | * This is accessible from the profile icon in the top right corner * Displays information about the user in json format into a text view * A user profile class was created to store user information such as name, surname, address, contact details * JsonObject() and JsonUtil libraries were implemented to create a json object based on the user information |

# Floating Action Button

|  |  |
| --- | --- |
| C:\Android\MyDrive\code-tests-master\Screenshot_20151213-163435.png | * The floating action button launches the standard android email activity. * It is a quick launch single step item that allows the user to send an email instead of going through the menu -> share -> select images -> share |

# App Evaluation

## Strengths

1. The app runs and is fully functional as per specifications
2. Navigating through the app is generally smooth
3. App features can be easily accessed through the side bar

## Weaknesses

Initially, the share fragment would dynamically load images as the user scrolls through the list. However, every time the user scrolls, a call to the getView(…) method from list view adapter is made and subsequently calls the bitmap decoder method. This makes scrolling through the list of images slow and sluggish. This can partially be attributed to the size of the images and the processing capability of the device.

To solve this app unresponsiveness issue, the images are preloaded at the beginning when the share fragment is launched. The share fragment takes a few seconds to load but it eventually results in a full list of images which can be scrolled smoothly.

A more appropriate technique would perhaps be to completely separate this functionality from the UI thread and use asynchronous tasks. The image loading can be done in the doInBackground(…) method.

Another possible method would be to load a few images at a time instead of the entire list. As the user scrolls to the end of the image list, the next group of images are then loaded.