Truong, Tipper
CS 175
Homework 1

Chapter 1

1.3 What is KitKat's most notable feature?

- KitKat's has similar features to Jelly Bean, however it could run a larger array of devices because of it's size which is 512 MB.

1.5 List the important Android development components

- The most important Android development components are: Java Editor, Layout Editor, Android SDK, AVD (Android Virtual Device), and Gradle.

1.10 What is the conversion of dp units to screen pixels?

- px = dp x (dpi/160)

1.11 Identify the basic component of an Android application.

- Basic Android application would have a manifest file, drawable resources, layout XML files, Java Source code, values resources, and gradle scripts

1.12 Describe the Model-View-Controller pattern design

- **Model**: The Model is responsible for computation, manipulation, and flow of data. In addition, it also encapsulates the data in the application.
- **View**: What the user sees in the application is the "view" and that could be interacted by the user
- **Controller**: Controls the data flow between the model and the view. Controller communicates with the Model with new or changed data that the user made in the View. The View is notified about changes from the Model through the controller objects.

1.17 Briefly describe the difference between KitKat and Jelly Bean.

- Jelly Bean had much greater performance than it's predecessors. In addition, it also contains Google Voice Search feature. KitKat has a small size of 512MB which enables it to run in multiple ranger of devices than Jelly Bean because Jelly Bean needed roughly 1GB or 3GB to run.

Chapter 2

2.1 Name the six standard root layouts provided by Android. Briefly describe each of them

1. Relative Layout
   - For screen designs that need to control elements to be positioned in relation to one another
2. Linear Layout
   - For simply arranging the elements either in a horizontal or vertical line
3. Table Layout
   - Arrange elements into tabular rows and columns

4. RowLayout
    - Layouts its components in row in vertical or horizontal style.
5. GridLayout
    - Is able to quickly display position components in a grid without using a table. The main purpose of a GridLayout is to display tabular data from an Adapter
6. FrameLayout
    - Mainly used to display a single element

2.2 Explain the main difference between TableLayout and GridLayout

- A GridLayout can quickly position components in the grid by editing the layout_row and layout_column attributes without requiring a table. In addition, to display tabular data, they use an Adapter. While for TableLayout, it is organized into neatly defined rows and columns. In addition, it could be used to align screen content similar to an HTML

2.3 Name three alignment attributes used with RelativeLayout

- layout_alignParentTop, layout_alignLeft, layout_toRightOf

2.4 Draw the interface produced by the following layout XML code:

```
<?xml version="1.0" encoding="utf-8"?>
<GridLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:columnCount="2"
    android:orientation="horizontal"
    android:rowCount="2">

    <TextView
        android:text="Apple" />

    <TextView
        android:text="Cherry" />

    <TextView
        android:text="Banana" />

    <TextView
        android:text="Peach" />

</GridLayout>
```

What the layout XML code produces:

Apple Cherry
BananaPeach

2.7 Name three InputType values that can be used with an EditText

- number, date password, or email address

2.8 Describe the input expectation and the soft keyboard configurations produced by the following XML code:

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/editText1"
        android:inputType="textCapWords|textPersonName" />

    <EditText
        android:id="@+id/editText2"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:inputType="textEmailAddress|textAutoComplete" />

    <EditText
        android:id="@+id/editText3"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:inputType="textPostalAddress " />


</LinearLayout>
```

Input Expectation and Soft Keyboard Configurations

- The first edit text is expecting the user enter their name. Provided are CAPS letter only in the keyboard
- The second edit text is expecting the user to enter their email address and auto-complete is enabled on the keyboard, therefore the user can autocomplete their input
- The third edit text is expecting the user to enter a postal address which pops up a keyboard that can enter both numeric and letters

2.9 Name the built-in user interface widgets provided by Android. Briefly explain the use of each one

- RadioButton – Once a button is selected, the radio button that was previously selected will be de-selected
- ToggleButton – Allows the user to change a setting between two states, such as on or off

2.10 What is the difference between a ToggleButton and a Switch?

- A Switch is a two-state toggle switch widget that can select between two options off and on. Basically dragging the “thumb” back and forth. While a Toggle button is simply a on/off button

2.11 What is the purpose of the R.java file? Briefly explain its construction

- It's main purpose is for rapid accessibility of resources in the projects. The R.java files are categorized into unique identifiers that are groupings for drawables, string, layout, elements, and so forth. The generated R class is located on the build folder in Android Studio and is constructed with final 32-bit values

2.12 What's the difference between a ViewGroup and a View?

- ViewGroup is an invisible container of View objects and basically hold a groups of View. Where Views are considered children. Views are object that are basically the building blocks of the UI elements in Android and they're simple rectangle box which to respond to user's actions.

2.14 Explain how Adapters are used with ViewGroups

- The Adapter Object is a bridge between the data and AdapterView object. Basically what the Adapter does is it retrieves the data from either a database or other stored values into a View object and then it gets added to either ViewGroup or AdapterView objects such as ListView, RecyclerView, etc.

Chapter 3

3.1 Name the seven callback methods defined by the Activity class.

- onCreate(), onStart(), onResume(), onPause(), onStop(), onRestart(), and onDestroy()

3.2 Explain the difference between onResume() and onStart()

- When the activity has been created and becomes visible to the user, the **onStart()** callback method is called. If the user is interacting with the application activity, **onResume()** is called.

3.4 Indicate the type of intent launched by the following segment of code. Explain.

```
1  public void launchActivity (View view) {
2  Intent intent = new Intent(getApplicationContext(),
3            XActivity.class);
4
5       startActivity(intent);
6  }
```

- The explicit intent is constructed. The second argument is the explicit target name of the activity XActivity.class. On line 5, the target activity (XActivity.class) , the system starts the app component specified in the Intent object.

3.6 Describe how data can be passed as a message object to an activity

- When creating an Intent object, you can set an action on the Intent object. In this case, the action would be ACTION_SEND and it will send data across process boundaries. In other words, it will deliver data to another activity.

3.8 Given the following XML code, name the Activities within the application and the activity that launches when the application first launches

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android">

<application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@android:style >

<!-- MYACTIVITY CLASS: INPUT THE CAR PURCHASE INFORMATION -->
<activity
            android:name=".BActivity"
            android:label="@string/app_name" >
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category
android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>

<activity
            android:name=".CActivity"
            android:label="@string/app_name"
android:parentActivityName=".BActivity" >
<meta-data
                android:name="android.support.PARENT_ACTIVITY"
                android:value=".BActivity" />
</activity>
<activity
            android:name=".DActivity"
            android:label="@string/app_name"
android:parentActivityName=".BActivity" >
<meta-data
                android:name="android.support.PARENT_ACTIVITY"
                android:value=".BActivity" />
</activity>

</application>

</manifest>
```

- BActivity is the Main Activity when the application first launches. The children activities within the application are CActivity and DActivity

3.9 Name five property animations that can be applied to activities

- rotation, rotationX, rotationY, scaleX, scaleY

3.11 Briefly explain the concept of a scene transition

A scene defines a given state of the UI. A transition defines the change from one scene to another. For example, in one activity, a user can type search queries and once they're done inputting their search query, the user submits it, the form is hidden and the list of search results occur in place. In this case, you could animate changes between different view hierarchies in your UI

3.12 Briefly describe the purpose of a TransitionManager

- It helps coordinate Scene with Transition objects. It's main purpose is to set transitions that fire when there is a change of Scene.

Chapter 4

4.1 Describe the Fragment lifecycle

- Fragments can be created, added, or removed while the activity is running. Each fragment has its own callback methods in the standard Activity lifecycle. For example, an application can use one fragment to receive input events which allows use to input information while another fragment is used for manipulating output.

4.2 List the set of Fragment callback methods used by a Fragment

- onCreate(), onStart(), onResume(), onPause(), onSaveInstanceState(), onStop(), and onDestroy()

4.3 Explain when onInflate() and onActivityCreated() are called

- onInflate() is called when every time the fragment is inflated
- onActivityCreated() is called when fragments activity has been created and the fragment's view hierarchy is instantiated.

4.4 Describe the features of a typical action bar

- An action bar contains features such as user action control elements, navigation modes, and drop-down menus

4.5 Every Activity supports an option menu of actions. Describe what occurs in the code segment shown below:

```
1  @Override
2      public boolean onCreateOptionsMenu(Menu menu) {
3          getMenuInflater().inflate(R.menu.my, menu);
4          return true;
5      }
```

- getMenuInflater().inflate (R.menu.my, menu) inflates the menu and add the defined items from the menu to the action bar.

4.6 Explain the purpose of an action view

- An action view purpose is to provide quick access to heavily used actions

4.8 Describe the master/detail design pattern

- In this design pattern, the user is provided with a list of items, which is called a master list. When you select one of the items, additional information relating to the item is displayed in the details panel. In a phone, the information is lead to another panel. However, in a tablet, both panels are included in the detail panels

4.9 How are Adapters used in a master/detail design?

In a master/detail design, ListView is needed to integrate in the Android Application. Adapters help modify the list dynamically and serves as a controller between the bind data and the AdapterView layout.

Chapter 5:

5.1 Describe the steps for creating XML graphic element to be stored as a drawable resource file

- XML graphic elements are drawable resources defined in XML. As a drawable, the the resource file is stored in res/drawable directory. To draw a XML graphic element, the drawable graphic has 4 primitive shapes and 4 shape attributes. Once completed, they're stored in the res/drawable directory

5.3 Describe the coordinate system used by a RelativeLayout. How does this compare with the coordinate system used in a FrameView?

- The coordinate system used by a RelativeLayout have x and y coordinates where 0,0 is on the top left hand corner of the screen. The width of screen is x and the height of the screen is y. All the coordinates are represented on px values. Similarly FrameView also has the same coordinates as RelativeLayout

5.4 List the properties that must be set for an ImageView that will result in the rotation around a pivot point.

- setPivotY(), setPivotX(), setScaleX(), setScaleY()

5.5 Write a segment of code to remove an ImageView object from the screen during runtime.

```
private View.OnClickListener clearImageView =
        new View.OnClickListener() {

        public void onClick(View view) {

                for (int I = 0 ; I < imageList.size(); i++)
                        ImageView img = imageList.get(i);
                        relativeLayout.removeView(img);

        }
}
```

5.6 What is the relationship between a Bitmap and a Canvas?

- A Canvas provides an interface where the graphics will be drawn upon. The purpose of canvas is to hold the results of the "draw" calls and while the drawing is being performed, an underlying Bitmap is placed into the window

5.7 Briefly explain the purpose of requestWindowFeature()?

- Enables specific window features, specifically a "no title" feature. This turns off the title at the top of the screen

5.9 What parameters are required for RotateAnimation()?

- fromDegrees, toDegrees, pivotXType, pivotXValue, pivotYType, pivotYValue, setDuration, setFillAfter()

5.10 Briefly describe the purpose of SoundPool and MediaPlayer.

- SoundPool is mainly used to play short audio files which are useful for audio alerts and simple game sounds. In addition, one of the many advantages of SoundPool is that you can play several sounds at once and could also be played in the background. However, it is slow for large raw files and that's where MediaPlayer comes into play. MediaPlayer is used for handling media playback and creates an interface between the user and music file. Users are able to play, pause, stop, rewind, and fastforward.

Chapter 7

7.1 List eight basic touch gestures

- Single tap, double tap, long press, scroll, drag, fling, pinch open, and pinch close

7.2 When is an onTouchEvent() method triggered?

- It's triggered when a touchscreen event occurs that is not handled by the "clear" button or the scroll element of the TextView. In addition, it produces text output indicating that a touch event has occurred.

7.3 Briefly describe three action codes supported by the MotionEvent class.

- ACTION_DOWN, ACTION_MOVE, ACTION_UP

7.4 Describe the usage of the GestureDetector class. What are its limits?

- GestureDetector are mainly used to detect specific touch gestures. This approach uses the onGestureListener to signal when a gesture occurs. Unfortuantely GestureDetector has limits and it doesn't work for complex gestures and only work for simple and generic gestures.

7.5 List and describe the callback methods contained in the GestureDetector class.

- onDown() – Will be called when a tap occurs in a downward motion event
- onLongPress() – Will be called when a long pressing motion occurs with the initial down motion event that triggered it.

- onShowPress – Will be called when the user has performed a down motion event but has not moved a finger or performed an up MotionEvent

- onSingleTapUp() – Will be called when a tap occurs. The event will not be completed until an up motion event occurs

- onDoubleTap() - Will be called when a double tap occurs. The event will not be completed until an up motion event occurs

7.6 The MotionEvent class provides a collection of methods to query the position and other properties of fingers in a gesture. Name and describe five of these methods.

- getX(): Retrieves the x coordinates value at the finger's location on the screen

-getY(): Retrieves the y coordinates value at the finger's location on the screen

- getDownTime():

7.7 Outline the implementation of a drag-and-drop gesture.

```
If motionEvent.getActionEvent == MotionEvent.ACTION_DOWN
        interpret drag data

        create a drag shadow

        call a method to start the drag

        hide the original image

        return true
else
        return false
```

7.8 Describe the purpose of DragShadowBuilder. How can a drag shadow be customized?

- The purpose of the DragShadowBuilder is creating a shadow image builder based on a View. A Drag Shadow can be customized by using registered listeners that can alter the appearance of a given View object to indicate that the listener can accept a drop event.

7.11 Outline the sequence of MotionEvents for a multitouch gesture that involves two pointers.

- Pointer 1 (primary pointer) goes down: ACTION_DOWN
- Pointer 2 (secondary pointer) goes down: ACTION_POINTER_DOWN
- Both pointers move: ACTION_MOVE
- Pointer 2 (secondary pointer) goes up: ACTION_POINTER_UP
- Pointer 1 (primary pointer) goes up: ACTION_UP