Lappeenrannan teknillinen yliopisto

School of Business and Management

Software Development Skills

Kim Yukyeong, x080298

Learning period: Spring 2021 (7.5.2021-6.7.2021)

**LEARNING DIARY**

Module Introduction 2

Module Core Elements 4

Module Lists, Layouts, and Images 6

***Part 1. Introduction***

7.5.2021 (Friday)  
this is the very first day of this course.

I read instruction in Moodle course page.

Course overview

General course information

Environment setup

Android mobile development…

I watched first video clip.

I created a public repository in GitHub for this course.

Link: https://github.com/saugkim/Android2021\_LUT

There are several commits made (add gitignore and diary and so on…) today.

I am not expert in git, but I am not beginner either, so I skip reporting git commits.

Work environments for this course:

- The git, and visual studio code, Android studio are installed already in my computer.

- I am using GitHub, so I decided to use it to complete this course rather than create a new bitbucket account.

9.5.2021 (Sunday, Mother’s Day)

I have done **My First App** following the video.

1. I learned how to set up Android Studio project, just simple steps (nothing new).

2. I learned how to use Android Studio to make a simple app (My First App).

- tip: check *optimize imports on the fly* and *add unambiguous imports on the fly* in Editor settings to make import easier for beginner, appearance size unit of “sp” is screen independent pixel

- how to create screen objects (button, text edit field, text view) in activity\_main.xml window and give them proper position, id, and types and so on.

- how to create java objects in MainActivity.java and connect them to the screen objects and set click event listener when event happens (button pressed).

3. I learned how to debug the codes and how to run the app using emulator.

Debugging

- how to set breakpoint in code, just click left side of the line where to go through

- how to debug app (shortcut shift+F9)

- how to go to through them (Step over, Step into, Step out)

Running App

- in menu, Run ‘app’ (shift + F10) and select virtual device (set up at first)

- it takes time

***Part 2. Core Elements in Android Studio***

10.5.2021 (Monday)

I watched the second part of video lessons.

In this lesson, I have learned how to switch screen (view) within same app or another app.

Tow import elements to implement bring another screen within or outside app.

Activity: the rectangular are that displays something.

Intent: action being requested that the device should try to perform

I created new project **Quicklauncher** and uploaded in repository.

Change app name to “Quick App Launcher” in res -> values -> strings.xml

Implementation steps

1. Create second activity

app - java – first package – new – Activity – gallery – select empty activity

activity name: SecondActivity

layout name: activity\_second

- this action creates SecondActivity.java and activity\_second.xml files

2. Set visual elements:

Buttons in activity\_main.xml aligned at center

- second activity button: open inner activity of same app

- google button: open another app

TextView in activity\_second.xml aligned at center

- showing message passed from the main activity

3. coding MainActivity.java

Create button object to launch second activity within same app.

Create button object to launch another app.

to launch an activity inside app inside on click event listener:

*create new Intent object obj*

Intent obj = new Intent(getApplicationContext(), SecondActivity.class)

*give information to obj to pass information(==value)*

obj.putExtra(key, value)

*start activity*

startActivity(obj)

to launch another app (browser: site, google.com… ) inside on click event listener:

*create new Intent object and Uri object address*

Uri address = Uri.parse("www.google.com");

Intent googleObj = new Intent(Intent.ACTION\_VIEW, address)

*start activity only if in case of intent object is resolved*

if ( googleObj.resolveActivity(getPackageManager()) ) != null

startActivity(googleObj)

4. coding SecondActivity.java

get information (== value) from main activity and show it in second screen.

*create TextView object to show the value passed from main*

*if Intent has key then get the value using getExtras()*

if getIntent().hasExtra(key) {

TextView tv = (TextView) findViewById(R.id.textView);

tv.setText( getIntent().getExtras().getString(key) )

}

I have learned key methods to launch inner or outer app as belows.

**Activity**

onCreate() is the first method that will fire when an activity is loaded.

findViewById() is method to locate resources in your activity

**Intent**

getIntent() a method to get the Intent that was passed to an activity

putExtra() a method to pass data as a key-value pair

getExtras().getString() a method to retrieve the String that was passed to the Intent object

startActivity() a method to launch another activity

I have learned one of core elements in Android Studio, and the app that I created following the video lesson works fine as intended.

***Part 3. Module Lists, Layouts, and Images***

12.5.2010 (Wed)

I have learned about Lists in android studio and how to handle oversized image.

I watched the third video clip.

The project name of Listapp is uploaded into git repository.

The Goals of this module:

- how to use ListView to display list

- how create a custom layout component to make a simple app

- how to incorporate images with ImageView

Implementation steps

1. create ListView object in main\_activity

Containers -> ListView

id: myListView

set layout\_width == match\_parent

2. set string arrays in res - values - strings.xml

add 3 different string arrays

items: peach, tomato, squash

prices: 1.0, 0.9, 0.82

descriptions: ..from Georgia, ..from Ohio, ..form California

*# in this way*

<string-array name="items">

<item>peach</item>

<item>tomato</item>

<item>squash</item>

</string-array>

...

3. coding MainActivity.java

ListView myListView;

String[] items;

String[] prices;

String[] descriptions;

# *inside onCreate():*

Resources res = getResources();

myListView = (ListView) findViewById(R.id.myListview);

items = res.getStringArray(R.array.items);

prices = res.getStringArray(R.array.prices);

descriptions = res.getStringArray(R.array.descriptions)

4. create new layout to connect myListView and item’s informations

res - layout - New - Layout resource file

file name: my\_listview\_detail

root element: TextView

5. in MainActivity

//myListView.setAdapter(

// new ArrayAdapter<String>(this, R.layout.my\_listview\_detail, items))

6. create new improved layout (custom layout)

res - layout - New - Layout resource file

file name: my\_listview\_detail

root element: RelativeLayout

7. in design view of my\_listview\_detail.xml

create 3 TextView objects for items, prices and descriptions

for items, id nameTextView set textSize 24sp, left top position

for prices, id priceTextView set textSize 24sp, right top position

for descriptions, id descriptionTextView set textSize 14sp, located below items

8. create new java class ItemAdapter to connect layout and array objects

app - java - top package(main) - New - Java Class

Name: ItemAdapter

superclass: android.widget.BaseAdapter

9. implement ItemAdapter.java: constructor and abstract methods (from BaseAdapter)

this class is how to present information using my\_listview\_detail layout template, put information into template and show in ListView component in our main activity

*# create LayoutInflater object*

LayoutInflater mInflater;

*# same String array object;*

String[] items, prices, descriptions;

# constructor with those string arrays

constructor parameter: Context c, String[] i, String[] p, String[] d

mInflater = (LayoutInflater) c.getSystemService(Context.LAYOUT\_INFLATER\_SERVICE);

*# method getView() is how to put stuff into view*

public View getView(int index, View v, ViewGroup vg) {

View v = mInflater.inflate(R.layout.my\_listview\_detail, null);

TextView nameTextView =

(TextView)v.findViewById(R.id.nameTextView);

... prices and description

nameTextView.setText(items[index])

… prices and descriptions

return v;

}

10. in MainActivity.java

*# create ItemAdapter instance inside onCreate()*

ItemAdapter itemAdapter = new ItemAdapter(this, items, prices, descriptions);

myListView.setAdapter(itemAdapter)

11. create new Activity to show details of items (item click from the list view)

app - java - first package(main) - New - Activity - select empty activity

name: DetailActivity

layout: activity\_detail.xml

12. in design view of activity\_detail.xml

add one ImageView to center of the screen

id: imageView

add placeholder image (any image ok)

13. in MainActivity

*# on Item click event listener inside onCreate()*

myListView.setOnClickItemListener(new ....) {

*# inside onItemClick()*

Intent showDetailActivity =

new Intent(getApplicationContext(), DetailActivity.class);

showDetailActivity.putExtra(key, index);

startActivity(showDetailActivity);

}

14. prepare images to show in detail activity

move images to the app - res - drawable folder (whatever method works)

!! large image can cause a problem, to show image properly in different devices, we need to scale the image !!

15. in DetailActivity.java

*# get Intent object inside onCreate()*

Intent in = getIntent();

int index = in.getIntExtra(key, -1);

ImageView imageView = (ImageView) findViewById(R.id.imageView);

if (index > -1)

scaleImage(imageView, getImageIndex(index));

*# create 2 helper methods*

private int getImageIndex(int idx){

switch(idx) {

case 0: return R.drawable.peach;

case 1: return R.drawable.tomato;

case 2: return R.drawable.squash;

default: return -1;

}

}

*# this method scale image to fit to the device*

private void scaleImage(ImageView imgView, int pic) {

Display screen = getWindowManager().getDefaultDisplay();

BitmapFactory.Options options = new BitmapFactory.Options();

options.inJustDecodeBounds = true;

BitmapFactory.decodeResource(getResources(), pic, options);

int imgWidth = options.outWidth;

int screenWidth = screen.getWidth();

if (imgWidth > screenWidth) {

int ratio = Math.round( (float)imgWidth / (float)screenWidth );

options.inSampleSize = ratio;

options.inJustDecodeBounds = false;

}

options.inJustDecodeBounds = false;

Bitmap scaledImage =

BitmapFactory.decodeResource(getResources(), pic, options);

imgView.setImageBitmap(scaledImage);

}

I have learned how to create list and how to display them in custom layout.

I have learned how to scale the size of images to display in device properly.