Android Application Basics

- By Joshua Cho (Treasurer)
- 09/30/20 8 pm – 9pm EST

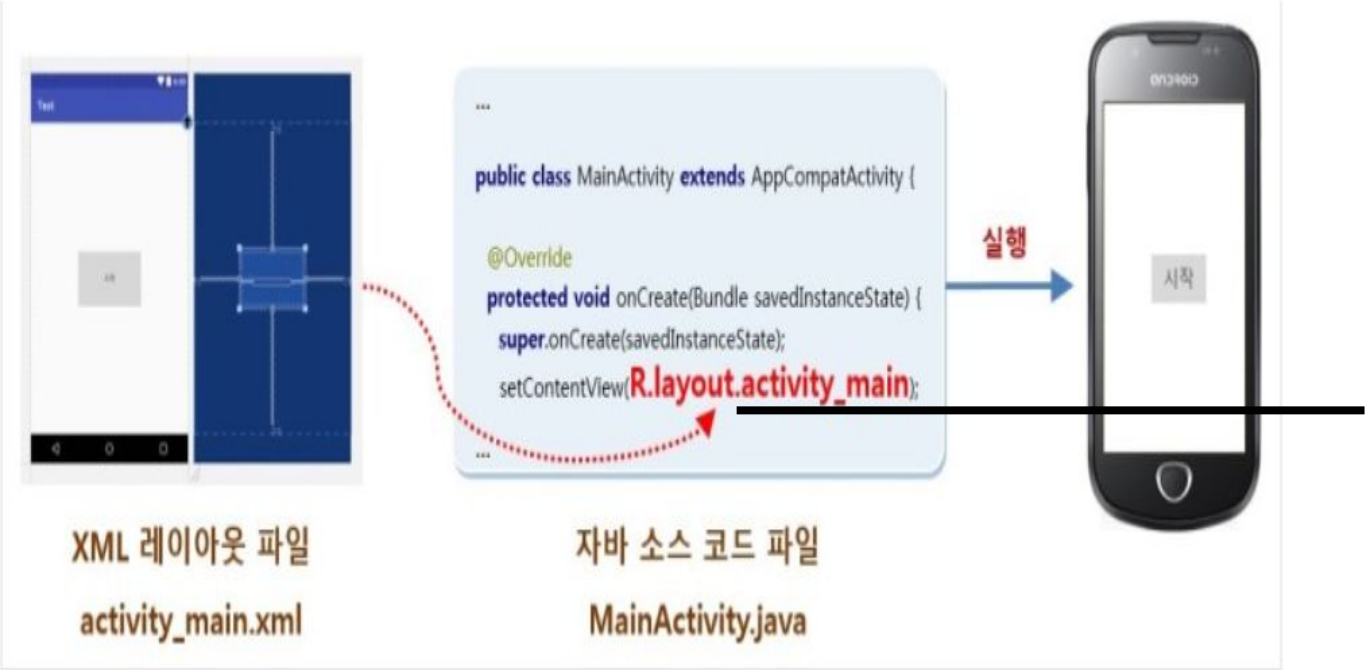
Contents

- Android Application Components
- How to use Android Studio & Build Hello World!
- Extend Hello World!

Android Application Basic Components

- Prerequisite: Java
- Some of my pictures might contain non-English words; however, I have used those pictures as they aren't important.
- Basic Android App Components: Activity, Button, View, Layout, Toast, Fragment, Intent, Progress Bar, etc.

Activity & Layout



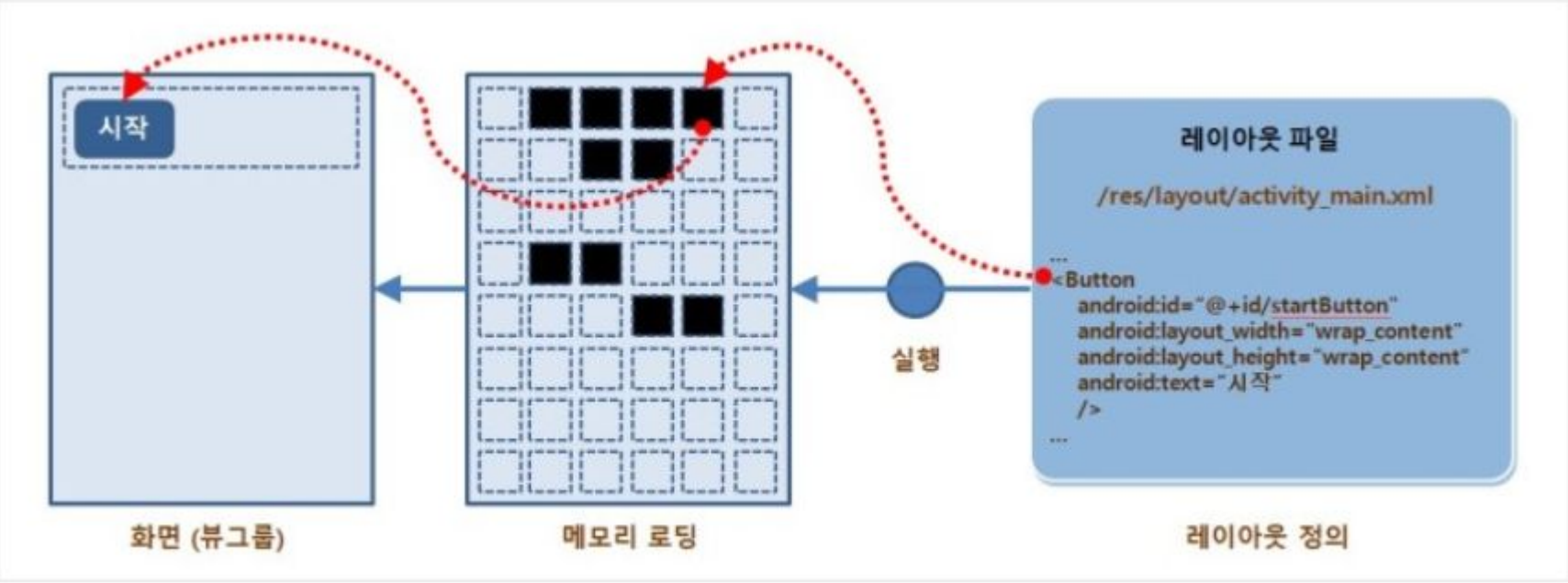
- setContentView – a function that inflates specific xml file. In this case, activity_main.xml is the xml file.
- R.layout.activity_main is the reference name to the layout xml file.
- So, In this code, we are asking to inflate activity_main.xml.

Activity – composed of xml file and Java/Kotlin source code.

Xml file –the display layout which would be on the screen.

Source Code – recognize and access each components (UI) through inflation and manipulate its functions.

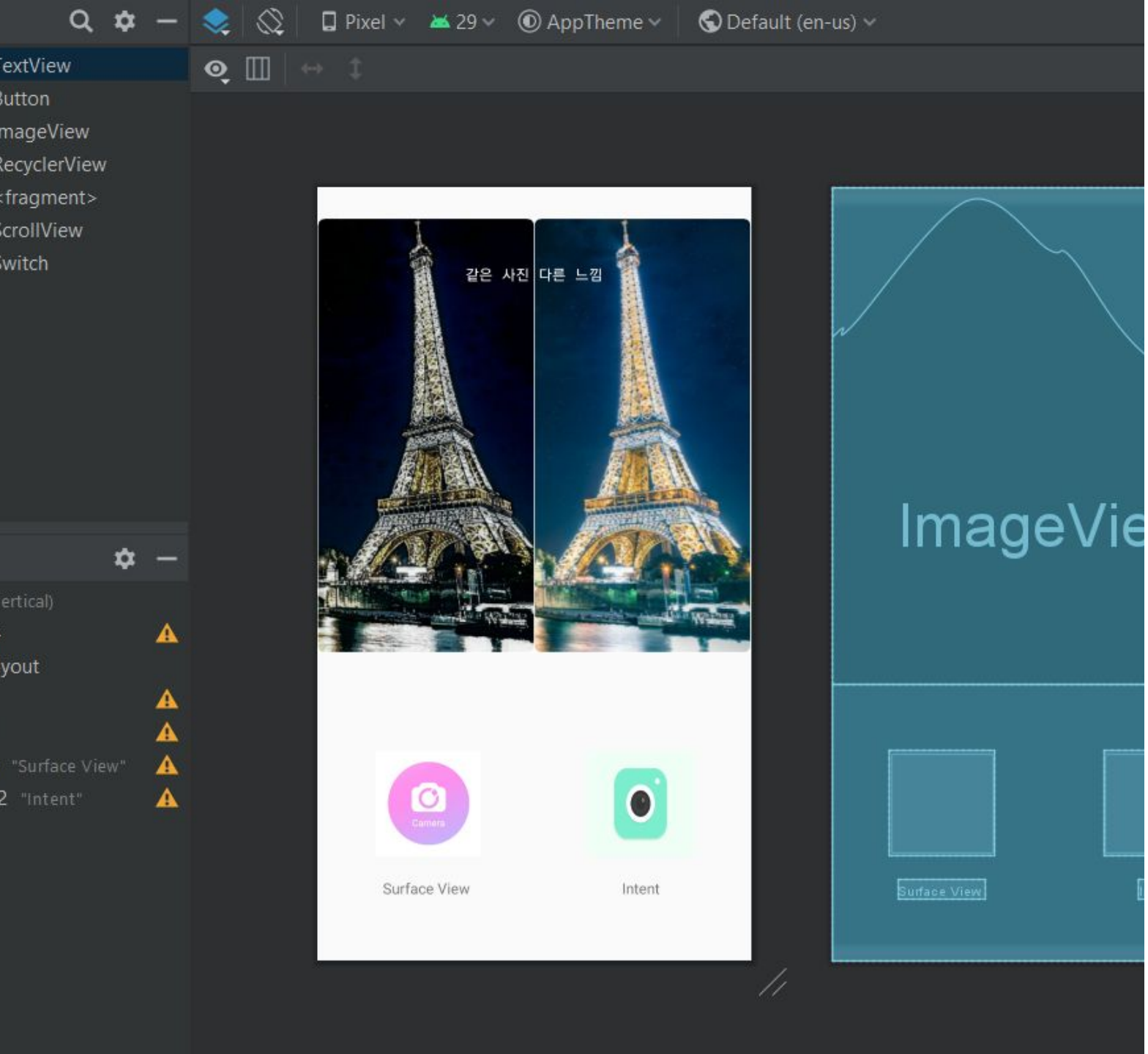
Inflation?

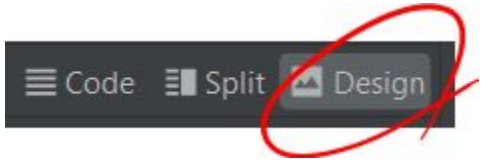


Inflation – instantiating a layout XML file into its corresponding View objects.

Through inflation, xml file would be uploaded onto the memory, and we can then use them by calling their ID. To find the specific object (such as Button), we should use findViewById() method.

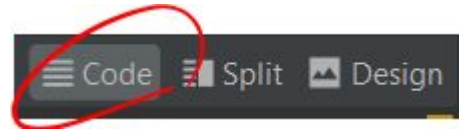
```
Button button = (Button) findViewById(R.id.startButton);
```





XML File (Design Mode)

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:backgroundTint="@color/filter_label_selected"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <ImageView
        android:id="@+id/imageView4"
        android:layout_width="match_parent"
        android:layout_height="470dp"
        app:srcCompat="@drawable/eiffel" />

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent" >

        <Button
            android:id="@+id/button"
            android:layout_width="100dp"
            android:layout_height="100dp"
            android:layout_marginStart="55dp"
            android:layout_marginTop="63dp"
            android:foreground="@drawable/takepicture"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent" />

        <Button
            android:id="@+id/button4"
```



XML File (Source Code Mode)

Button & Toast & Listener

```
<Button  android:id="@+id/button"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:onClick="onButton1Clicked"
  android:text="확인1"

  중략...
```

```
public void onButton1Clicked(View v) {
    Toast.makeText(this, "확인1 버튼이 눌렸어요.", Toast.LENGTH_LONG).show();
}
```

Create onClick function in the xml file and use it in the source code.

```
bt=(Button)findViewById(R.id.click);
bt.setOnClickListener(new OnClickListener(){
    public void onClick(View v) {
    // TODO Auto-generated method stub
        Toast.makeText(getApplicationContext(), "You made a
    }

});
```

Listener

Receives any occurred events.

View.OnTouchListener : boolean onTouch (View v, MotionEvent event)
View.OnKeyListener : boolean onKey (View v, int keyCode, KeyEvent event)
View.OnClickListener : void onClick (View v)

Intent

(exactly as its name suggests!)

- **Intent** – A message object that tells what the app component will do.
- **Activity Conversion** *new Intent(MainActivity.this, NextSurface.class)
- **Transfer Data** *putExtra() –> getIntent() & getStringExtra()
- **Go To Specific Website** *new Intent (Intent.ACTION_VIEW, Uri.parse("…");
- **Make A Call** *new Intent (Intent.ACTION_DIAL, Uri.parse("tel: …");

Layout & Activity Lifecycle

Android Linear Layout

In android, LinearLayout is a `ViewGroup` subclass which is used to render all child `View` instances one by one either in a horizontal direction or vertical direction based on the orientation property.

To know more about LinearLayout check this, Android LinearLayout with Examples.

Android Relative Layout

In android, RelativeLayout is a `ViewGroup` which is used to specify the position of child `View` instances relative to each other (Child A to the left of Child B) or relative to the parent (Aligned to the top of a parent).

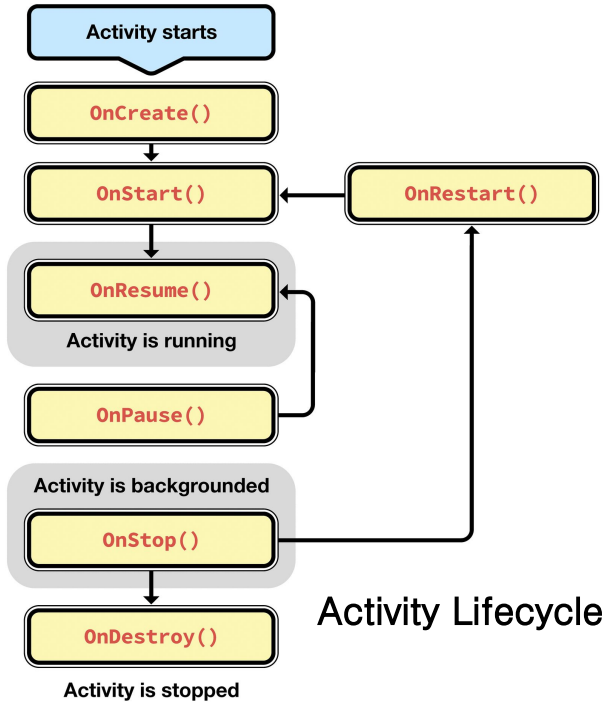
To know more about RelativeLayout, check this Android RelativeLayout with Examples.

Android Frame Layout

In android, FrameLayout is a `ViewGroup` subclass which is used to specify the position of `View` instances it contains on the top of each other to display only a single View inside the FrameLayout.

To know more about FrameLayout, check this Android FrameLayout with Examples.

There are many more!



Activity Lifecycle

- Which layout to use
- Arrange views in a layout as intended – gravity, visibility, foreground, color, margin
- Layout over Layout

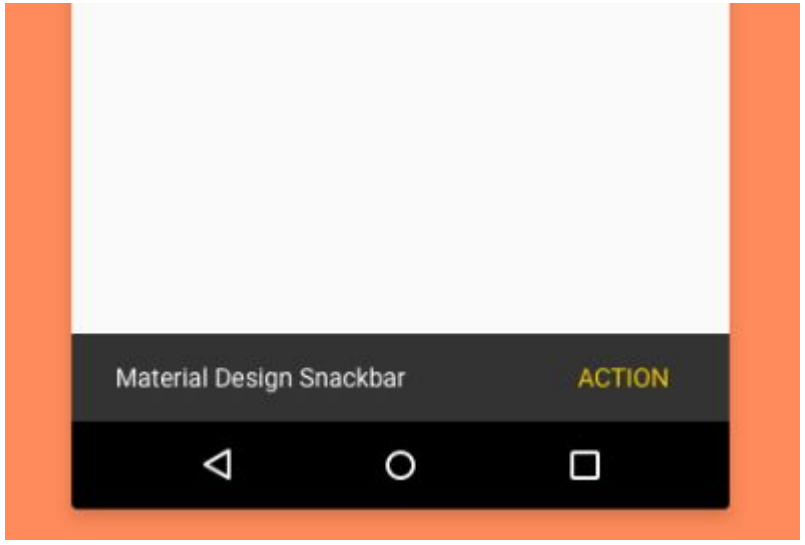
Display Orientation

- When rotating your phone, different activity occurs when you are in horizontal or in vertical direction. Therefore, the data stored in the memory would disappear. Thus, to save those data…
- **Disable display rotation**: at Manifest file, put android:screenOrientation = “portrait”
- **Call onSaveInstanceState () method**: onSaveInstanceState(Bundle bundle)

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
```

```
    if (savedInstanceState != null) {  ---> ❷ 이 화면이 초기화될 때 name 변수의 값 복원
        name = savedInstanceState.getString("name");
        showToast("값을 복원했습니다 : " + name);

    }
}

중략...

@Override
protected void onSaveInstanceState(Bundle outState) {
    super.onSaveInstanceState(outState);

    outState.putString("name", name);  ---> ❸ name 변수의 값 저장
}
```

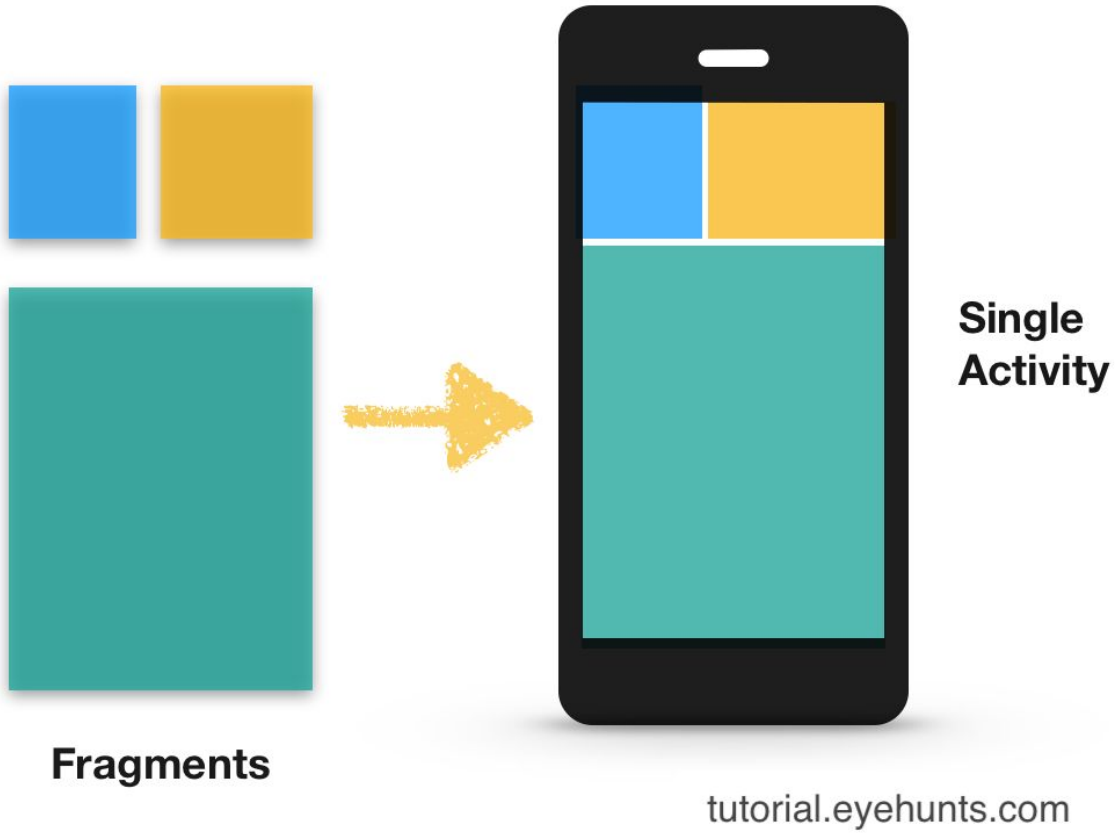
SnackBar & Alert Dialog




```
public void onButton3Clicked(View v) {
    Snackbar.make(v, "스낵바입니다.", Snackbar.LENGTH_LONG).show();
}
```


```
private void showMessage() {
    AlertDialog.Builder builder = new AlertDialog.Builder(this); ——→ ❶
    builder.setTitle("안내");
    builder.setMessage("종료하시겠습니까?");
    builder.setIcon(android.R.drawable.ic_dialog_alert);

    builder.setPositiveButton("예", new DialogInterface.OnClickListener() { ——→ ❷
        public void onClick(DialogInterface dialog, int which) {
            String message = "예 버튼이 눌렸습니다. ";
            textView.setText(message);
        }
    });

    builder.setNeutralButton("취소", new DialogInterface.OnClickListener() { ——→ ❸
        public void onClick(DialogInterface dialog, int which) {
            String message = "취소 버튼이 눌렸습니다. ";
            textView.setText(message);
        }
    });

    builder.setNegativeButton("아니오", new DialogInterface.OnClickListener() { ——→ ❹
        public void onClick(DialogInterface dialog, int which) {
            String message = "아니오 버튼이 눌렸습니다. ";
            textView.setText(message);
        }
    });
    AlertDialog dialog = builder.create(); ——→ ❺
     dialog.show();
}
```

Fragment



When do we use fragment?

- Like in a large tablet, it's much more efficient to upload multiple activities and fragment allows us to better control them.

- Advantage:

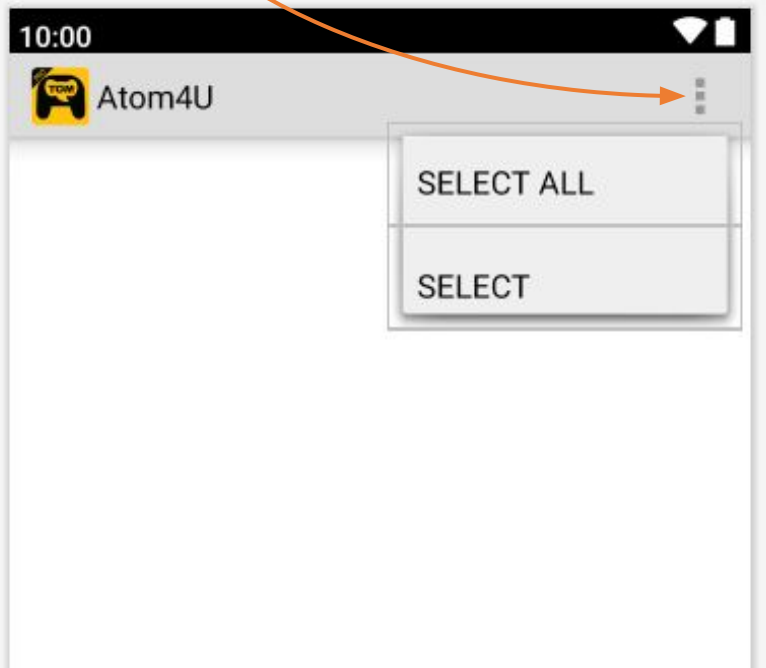
As fragments are controlled by a single activity, switching to different fragments are much more efficient.

Overflow Icon

4uResultFragment.java | Atom4uActivity.java | Atom4uExplorerFragment.java | atom_action_bar_no_button.xml | atom_action_bar_explorer_image.xml | atom4u_fragment_explorer.xml | Fingra

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

    <item
        android:id="@+id/selectAll"
        android:title="SELECT ALL"
        app:actionProviderClass="com.fingram.android.atom4u.app.Ato
        app:showAsAction="never" />

    <item android:id="@+id/menu_item_action_provider_action_bar_sel
        android:title="SELECT"
        app:actionProviderClass="com.fingram.android.atom4u.app.Ato
        app:showAsAction="never" />

</menu>
```

Palette



ViewPager

ViewPager?
Allows us to view screen by swiping either to the right or left.

```
public class AtomResultPreview extends FrameLayout
{
    //private SubsamplingScaleImageView zoom;

@SuppressLint("ClickableViewAccessibility")
public AtomResultPreview(Context context, AttributeSet attrs)
{
    super(context, attrs);
    Log.i( tag: "Info",  msg: "AtomResultPreview class second constructor is called");
    mContext = context;
    initView();
}
```

Inflate the Layout containing ViewPager

```
public void initView()
{
    Log.i( tag: "Info",  msg: "AtomResultPreview class initView() method called");
    // TODO Auto-generated constructor stub
    View.inflate(mContext, R.layout.atom_result_preview_layout,  root: this);
    Log.i( tag: "Info",  msg: "AtomResultPreview class initView() method called");
    //zoom = (SubsamplingScaleImageView)findViewById(R.id.zoomIn);
```

Find ViewPager by ID

```
    mViewPager = (AtomResultPreviewPager)findViewById(R.id.myViewPager);
    mCountInformationText = (TextView)findViewById(R.id.atom_result_preview_count_text);

    //mFingramAtomResultInfo = (FingramAtomResultInfo)findViewById(R.id.atom_result_image_reduce_info);
}
```

Put any data to the ViewPager to be viewed

```
public void setPosition(int position)
{
    Log.i( tag: "Info",  msg: "AtomResultPreview class setPosition() method called");
    mCountInformationText.setText(String.format("%d/%d", position+1, mImageCount));
    mViewPager.setCurrentItem(position);
}
```

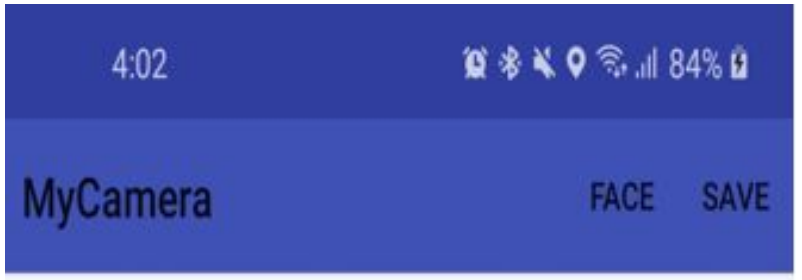
Camera

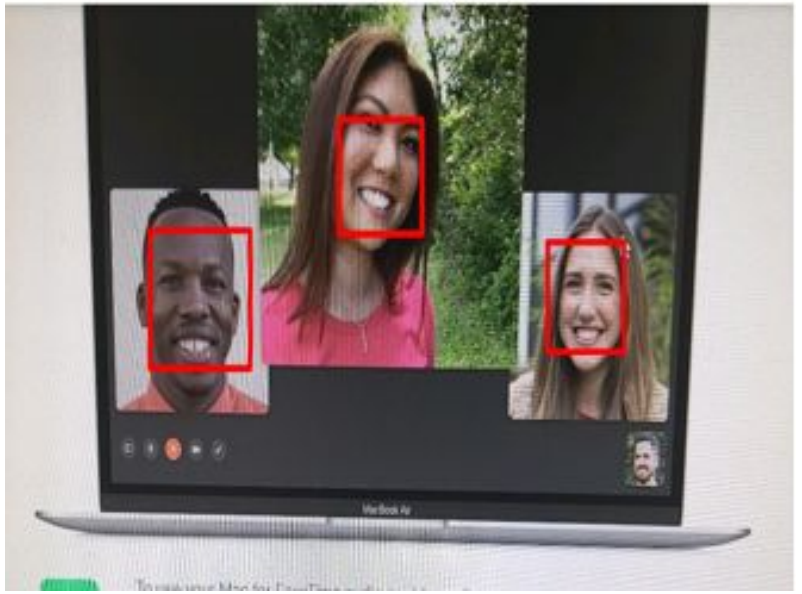
- To take pictures…

- Call intent to open camera on your device and take picture
- User SurfaceView

Face Detection

- By using FaceDetector class in Google API, you can also allow the phone to recognize people's faces.





On to Android Studio…

- Open android studio (Close Project)
- Start a new Android Studio project
- Empty activity ->language (Java)
- Explain source code, res, emulator
- Emulator (use default if everyone works if not) … (Pixel2 -> Q system image)
- Hello World App step by step (my phone)

Thank You!