가천대학교
Gachon University

학생 여러분 반갑습니다.

다른 친구들이 입장할 때까지 조금 기다려 주십시오.

곧 모바일 프로그래밍 수업을 시작합니다.

음소거( )가 되었는지 확인 바랍니다.

모바일 프로그래밍
화목(1,2교시)/ 화목(3,4교시)
정윤현 (AI/소프트웨어학부)

Mobile Programming

Android Programming

Chap 5-2. Widgets & Event (advanced)

Prof. Younhyun Jung

Email) younhyun.jung@gachon.ac.kr

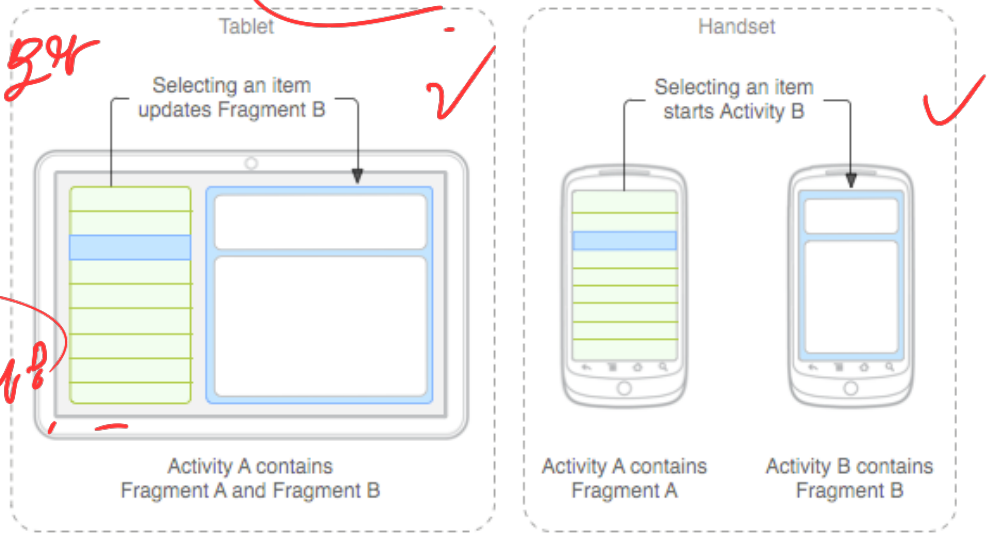
가천대학교 소프트웨어학과
Gachon University

Outline

- **Fragment**
- **WebView**
- **List View**
- **Spinner**

Fragment

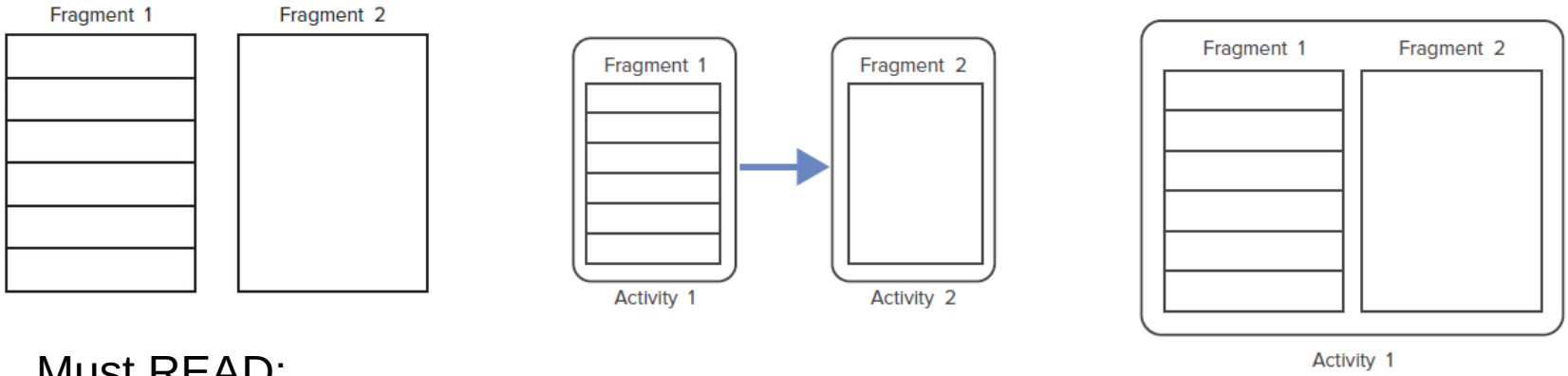
- **Fragment**
  - A piece of an application's user interface or behavior that can be placed in an Activity (=mini-activity)
  - Has its own layout and its own behavior with its own lifecycle callbacks.
  - One or more fragments can be combined to build a multi-pane UI (in a single Activity)
  - A single Fragment can be reused across multiple Activities



Fragment

장점?

- Key strength: You can combine multiple fragments in a single activity to build a multi-pane UI and reuse a fragment in multiple activities.

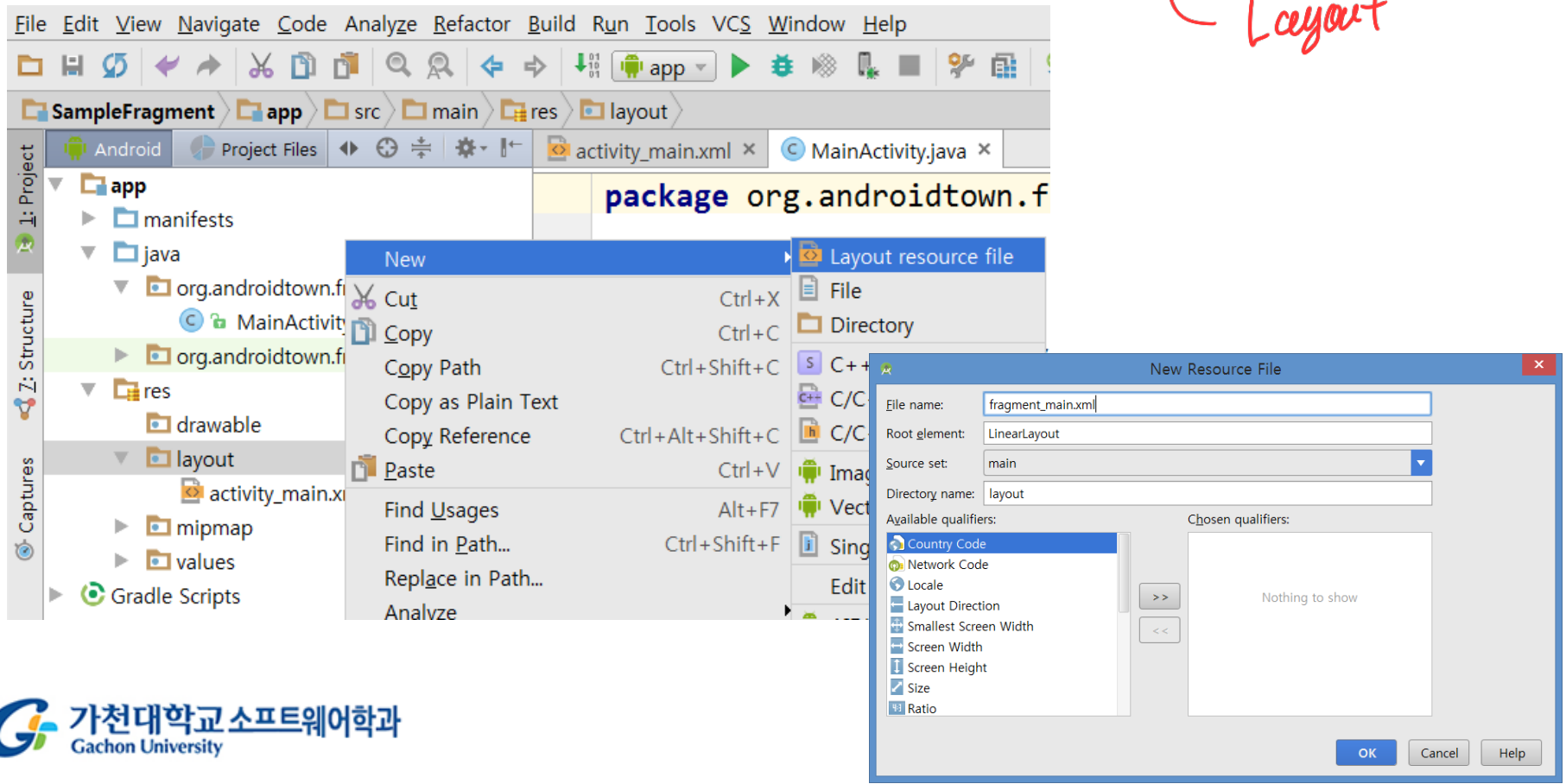


- Must READ:
  - http://developer.android.com/guide/components/fragments.html
  - https://developer.android.com/training/basics/fragments/index.html

Exercise

- Simple example may help you learn how to use fragment
  - Let's make a layout file for fragment

Java
Layout

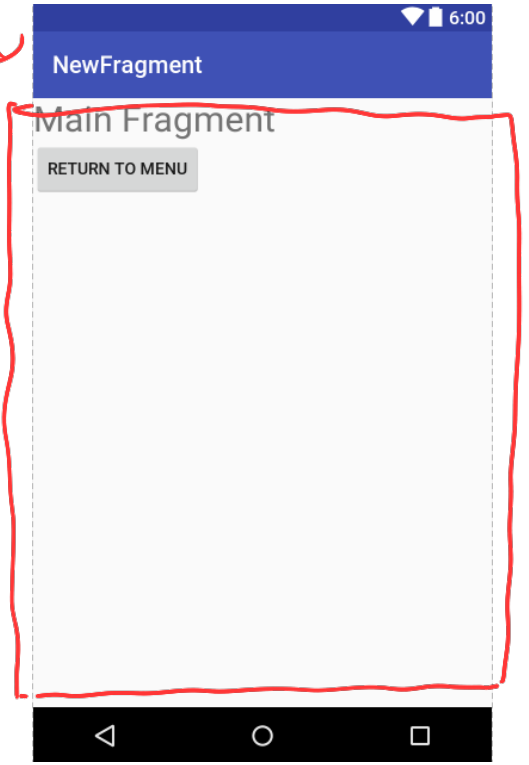


Exercise

Main Activity가 아니라 Fragment 임?

- fragment_main.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">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Main Fragment"
        android:id="@+id/textView"
        android:textSize="30dp" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Return to Menu"
        android:id="@+id/button" />

</LinearLayout>
```



Exercise

- Make a new java file : MainFragment.java

```java
package com.swdm.mp.android101;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;

public class MainFragment extends Fragment {
    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
        ViewGroup rootView = (ViewGroup) inflater.inflate(R.layout.fragment_main, container, false);
        return rootView;
    }
}
```

Exercise

- activity_main.xml to include fragment

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/container"
    >

    <fragment
        android:id="@+id/mainFragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:name="com.swdm.mp.android101.MainFragment" />

</RelativeLayout>
```

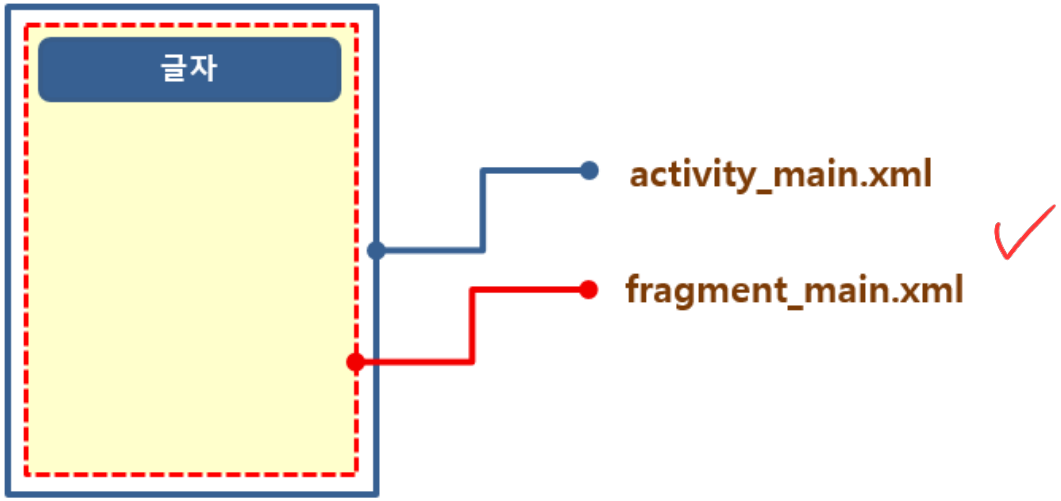
main에 추가.

Important : package name.Fragment class

이름을 이런 식으로 연결해주어야 함!

Exercise

- RUN!!



Summary of How to use Fragment

- Procedure 1) Making XML layout for fragment
- Procedure 2) Making Fragment Class
- Procedure 3) Adding Fragment to XML Layout of Activity

Fragment Transactions – adding, removing and replacing dynamically

- *Create new fragment and transaction*

```
ExampleFragment newFragment = new ExampleFragment();
FragmentTransaction transaction = getSupportFragmentManager().beginTransaction();
```

- *Replace whatever is in the fragment_container view with this fragment and add the transaction to the back stack*

```
transaction.replace(R.id.fragment_container, newFragment);
```

- *Commit the transaction*

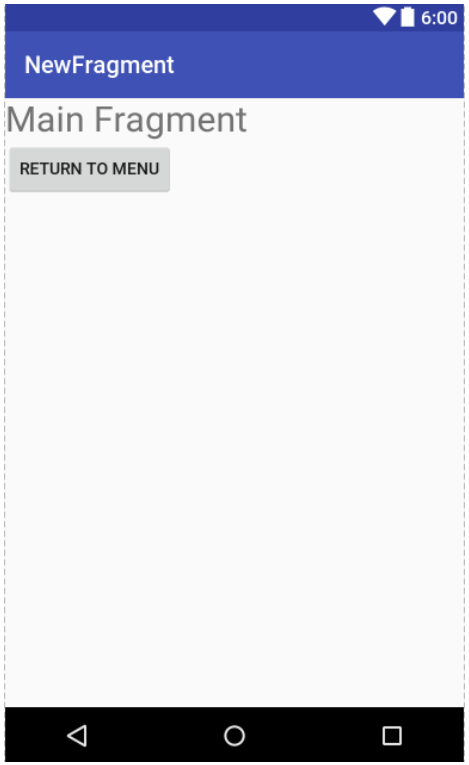
```
transaction.commit();
```

newFragment replaces whatever fragment (if any) is currently in the layout container identified by the R.id.fragment_container

Exercise

- Add fragment_main.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">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Main Fragment"
        android:id="@+id/textView"
        android:textSize="30dp" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Return to Menu"
        android:id="@+id/button" />

</LinearLayout>
```



Exercise

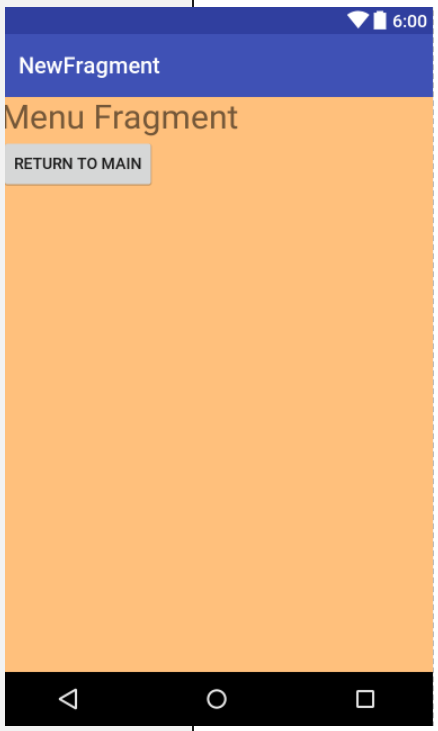
- Add MainFragment.java

```
public class MainFragment extends Fragment {
    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container,
@Nullable Bundle savedInstanceState) {
        ViewGroup rootView = (ViewGroup) inflater.inflate(R.layout.fragment_main,
container, false);

        Button button = (Button) rootView.findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                MainActivity activity = (MainActivity) getActivity();
                activity.onFragmentChanged(0);
            }
        });

        return rootView;
    }
}
```

Exercise

- Add fragment_menu.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“
    android:background="#E8AC6B">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Menu Fragment"
        android:id="@+id/textView"
        android:textSize="30dp" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Return to Main"
        android:id="@+id/button" />

</LinearLayout>
```



Exercise

Add MenuFragment.java

```
package com.swdm.mp.android101;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;

import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;

public class MenuFragment extends Fragment {
    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle
savedInstanceState) {
        ViewGroup rootView = (ViewGroup) inflater.inflate(R.layout.fragment_menu, container, false);

        Button button = (Button) rootView.findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                MainActivity activity = (MainActivity) getActivity();
                activity.onFragmentChanged(1);
            }
        });

    return rootView;
    }
}
```

Exercise

- activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/container"
    >


</FrameLayout>
```

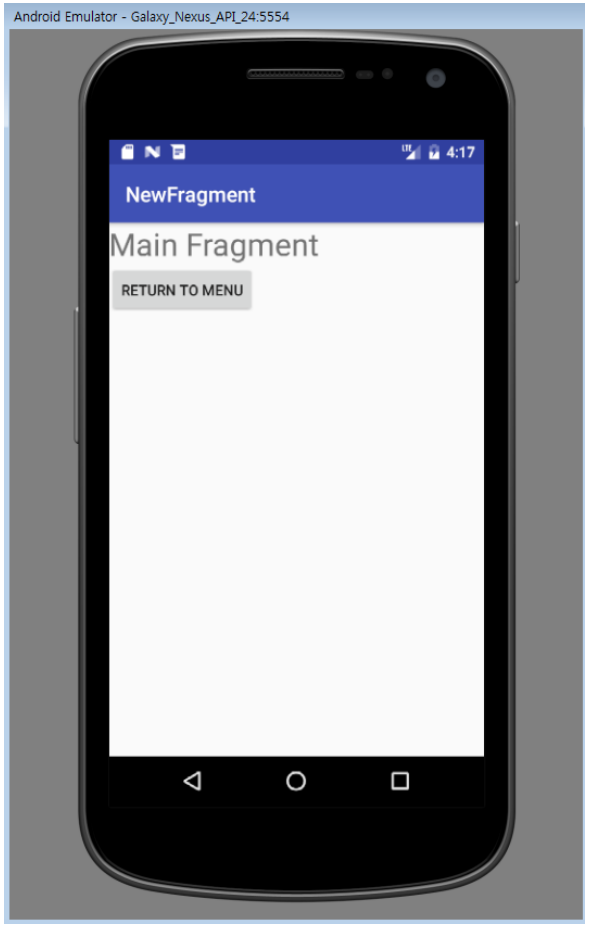
Exercise

- MainActivity.java

```
package com.swdm.mp.android101; // your own package name

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {
    MainFragment mainFragment;
    MenuFragment menuFragment;

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

        mainFragment = new MainFragment();
        getSupportFragmentManager().beginTransaction().replace(R.id.container, mainFragment).commit();
        menuFragment = new MenuFragment();
    }

    public void onFragmentChanged(int index) {
        if (index == 0) {
            getSupportFragmentManager().beginTransaction().replace(R.id.container, menuFragment).commit();
        } else if (index == 1) {
            getSupportFragmentManager().beginTransaction().replace(R.id.container, mainFragment).commit();
        }
    }
}
```

Exercise

- RUN!



From main fragment

To menu fragment

On single main activity

Fragment Lifecycle

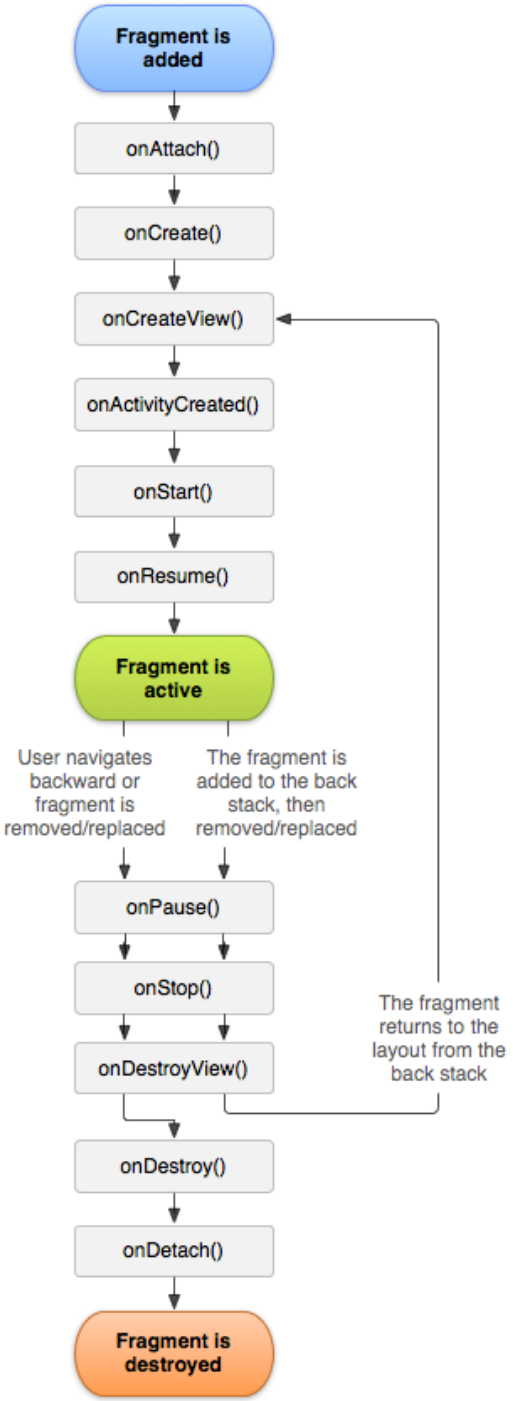
- A Fragment represents a behavior or a portion of user interface in an Activity.

- You can think of a fragment as a modular section of an activity, which has its own lifecycle, receives its own input events, and which you can add or remove while the activity is running (sort of like a "sub activity" that you can reuse in different activities).

Fragment Lifecycle

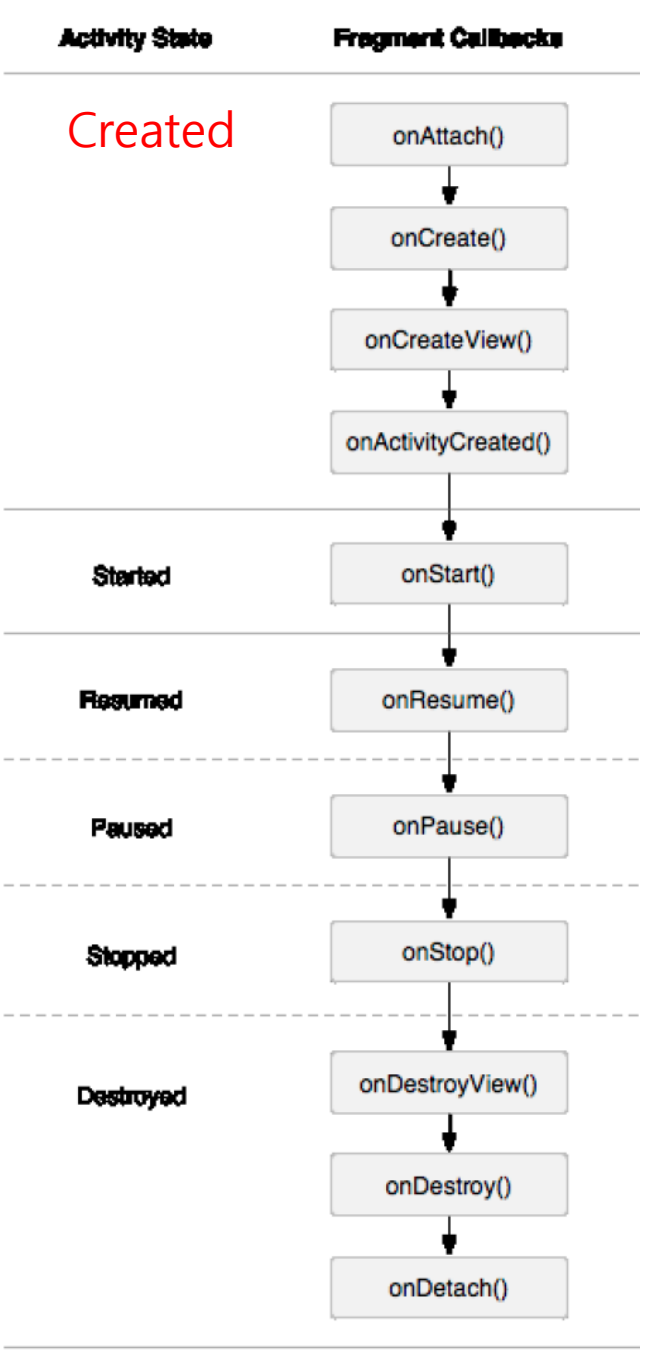
- Fragment in an Activity---Activity Lifecyle influences
  - Activity paused → all its fragments paused
  - Activity destroyed → all its fragments paused
  - Activity running → manipulate each fragment independently.

- Fragment transaction →add, remove, etc.
  - adds it to a back stack that's managed by the activity—each back stack entry in the activity is a record of the fragment transaction that occurred.
  - The back stack allows the user to reverse a fragment transaction (navigate backwards), by pressing the *Back* button.



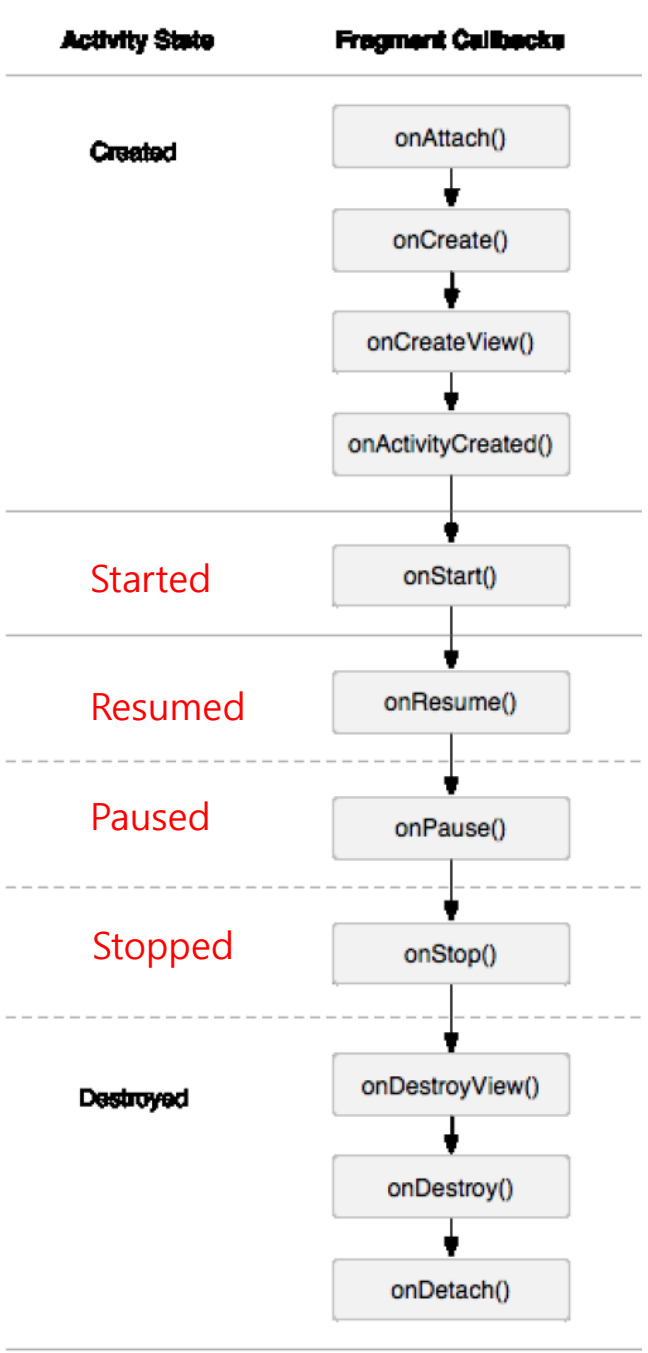
Lifecycle Callbacks

- **onAttach()** Called when the fragment is first attached to its activity
- **onCreate()** Called when creating the fragment. Generally used for **initializing** the fragment
- **onCreateView()** Called to create the view hierarchy associated with the fragment (return its UI). - inflation
- **onActivityCreated()** Called when the containing Activity has completed onCreate() and the fragment has been installed



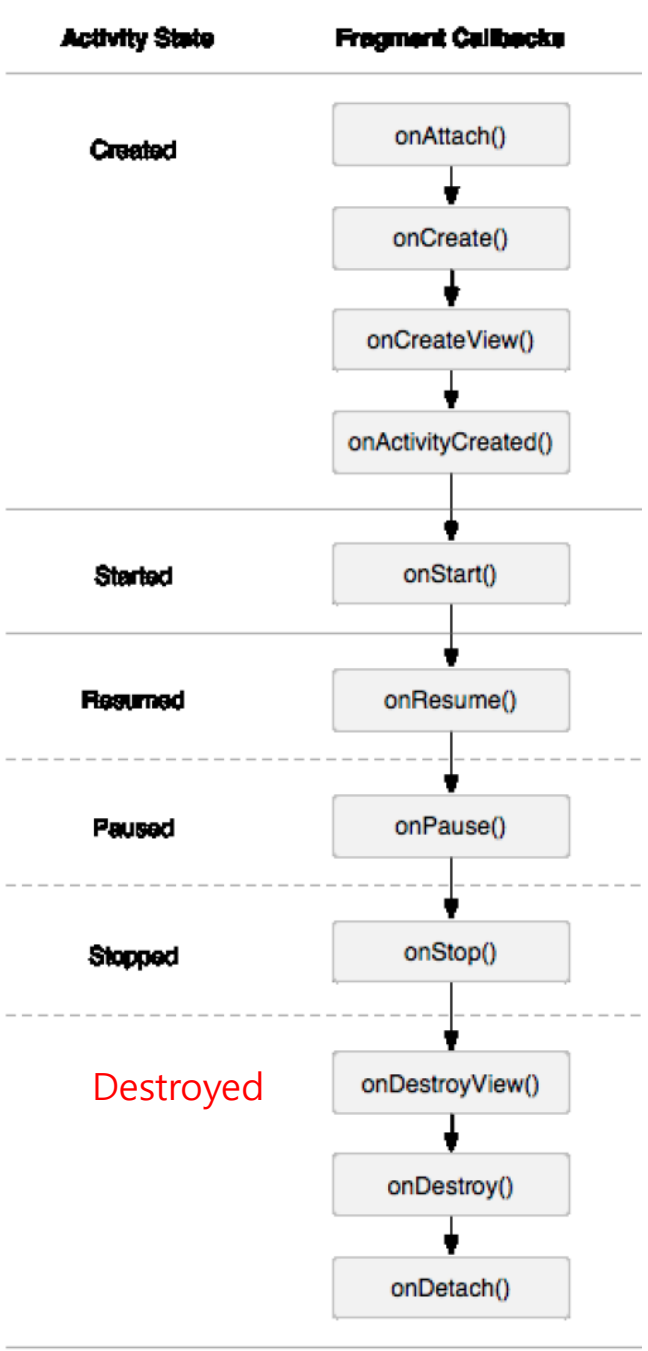
Lifecycle Callbacks

- **onStart()** Hosting Activity about to become visible
- **onResume()** Hosting Activity about to become visible and ready for user interaction
- **onPause()** Called when the user leaves the fragment. Here you should commit state changes that are needed in case the fragment is re-executed.
- **onStop()** Called when the hosting Activity is no longer visible



Lifecycle Callbacks

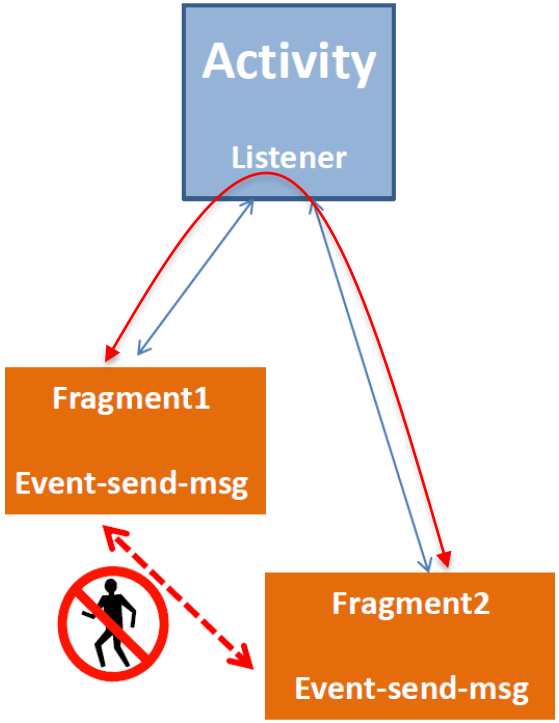
- **onDestoryView()** Called when the view previously create by onCreateView() has been detached from the Acitivity : Typically-Clean up view resources
- **onDestroy()** Called when the fragment is no longer in use: typically- clean up fragment resources
- **onDetach()** Fragment no longer attached to its activity: Null out references to hosting activity



Inter-Fragment Communication

- All Fragment-to-Fragment communication is done through the associated Activity.
- Two Fragments should ***never*** communicate directly.

- Activity and fragments interact through listeners and events.
- If a fragment has 'global' data to share, it should trigger an internal event to call the activity listener's attention and pass the global data to it.



Fragment 간의 통신
↳ Activity를 통해서 해야 함

Exercise

- fragment1.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"
    android:background="#00FF00"
    >
    <TextView
        android:id="@+id/lblFragment1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="This is fragment #1"
        android:textSize="20sp" />
</LinearLayout>
```

Exercise

- fragment2.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"
    android:background="#FFFE00"
    >
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="This is fragment #2"
        android:textSize="20sp" />

    <Button
        android:id="@+id/btnGetText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Get text in Fragment #1" />

</LinearLayout>
```

Exercise

- Make a new java file : Fragment1.java

```
package com.swdm.mp.android101;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;

public class Fragment1 extends Fragment {

    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container,
@Nullable Bundle savedInstanceState) {
        return inflater.inflate(R.layout.fragment1, container, false);
    }
}
```

```
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;
```

Exercise

- Make a new java file : Fragment2.java

```
public class Fragment2 extends Fragment {
    @Nullable
    @Override
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
        return inflater.inflate(R.layout.fragment2, container, false);
    }

    @Override
    public void onStart() {
        super.onStart();
        Button btnGetText = getActivity().findViewById(R.id.btnGetText);
        btnGetText.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                TextView lbl = getActivity().findViewById(R.id.lblFragment1);
                Toast.makeText(getActivity(), lbl.getText(), Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

Activity에 접근해서 값을 가져옴!?

Fragment가 속해있는 Activity

Activity에 포함된 모든 Activity!?

```
getActivity().findViewById(R.id.lblFragment1);
```

You can obtain the activity in which a fragment is currently embedded by first using the getActivity() method and then using the findViewById() method to locate the view(s) contained within the fragment

Exercise

- activity_main.xml to include fragment

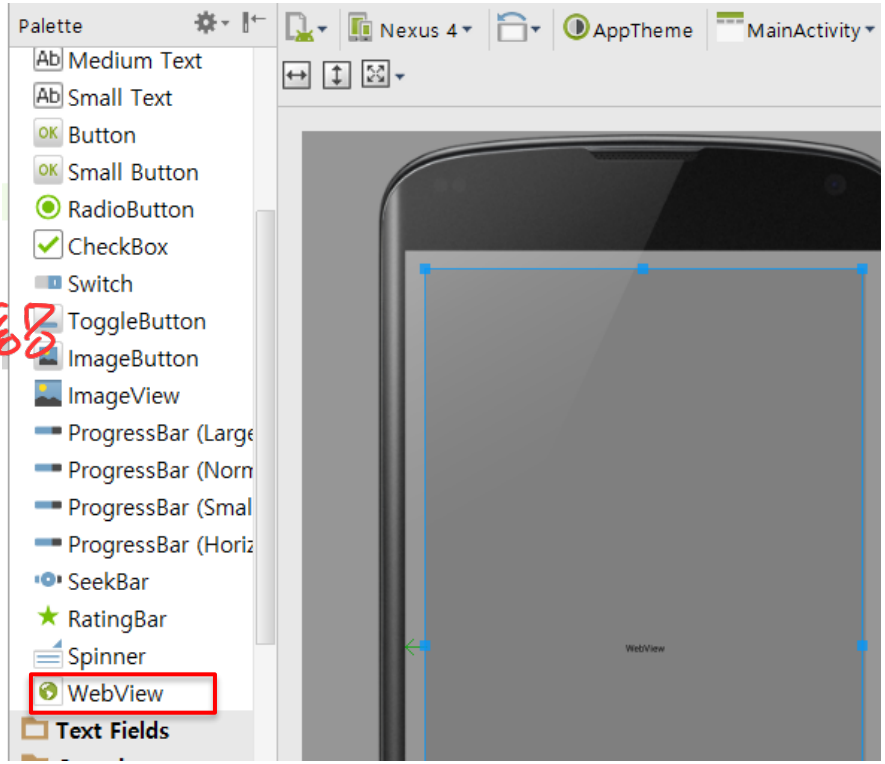
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal" >
    <fragment
        android:name="com.swdm.mp.android101.Fragment1"
        android:id="@+id/fragment1"
        android:layout_weight="1"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

    <fragment
        android:name="com.swdm.mp.android101.Fragment2"
        android:id="@+id/fragment2"
        android:layout_weight="1"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</LinearLayout>
```

Exercise

- RUN!

WebView

- Android provides a built-in *Web browser UI* control called WebView.
- WebView is used for displaying local HTML material or browsing Internet pages.
- The Android browser is based on WebKit open source Engine, the same engine that powers *Apple's Safari Web* browser.
- Applications using the WebView widget must request **INTERNET *permission*.**



http://www.webkit.org

WebView

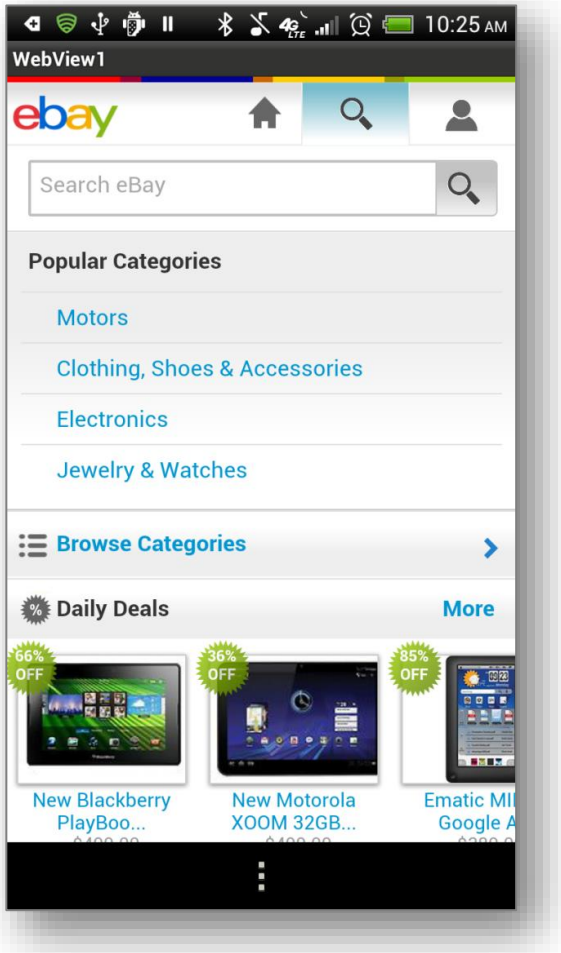
- The browser will try to access the Internet through
  - WiFi services,
  - cellular network,
  - reverse tethering,
  - any other technology available.

- The WebKit engine includes methods to
  - navigate forward and backward through a history record,
  - zoom in and out,
  - perform text searches,
  - load data
  - stop loading and
  - more…

Web browsing에 필요한 모든 것이 있음

Exercise

- This example uses a WebView widget to Reach a particular URL (eBay)



Reference:
http://developer.android.com/guide/webapps/webview.html

Exercise

- This example uses a WebView widget to Reach a particular URL (eBay)

Layout

```
<?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" >

    <WebView
        android:id="@+id/webkit"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

</LinearLayout>
```

추가

Exercise

- Make java file as follows:

```
package com.swdm.mp.android101; // your own package name

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.webkit.WebView;
import android.webkit.WebViewClient;

public class MainActivity extends AppCompatActivity {
    WebView browser;

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

        browser = (WebView) findViewById(R.id.webkit);

        browser.getSettings().setJavaScriptEnabled(true); // allow scripts
        browser.setWebViewClient(new WebViewClient());    // page navigation

        browser.loadUrl("https://www.ebay.com");
    }
}
```

Exercise

- Internet permission : add the permission in manifest file

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.swdm.mp.android101">

    <uses-permission android:name="android.permission.INTERNET"/>

    <application
        android:usesCleartextTraffic="true"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">

        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

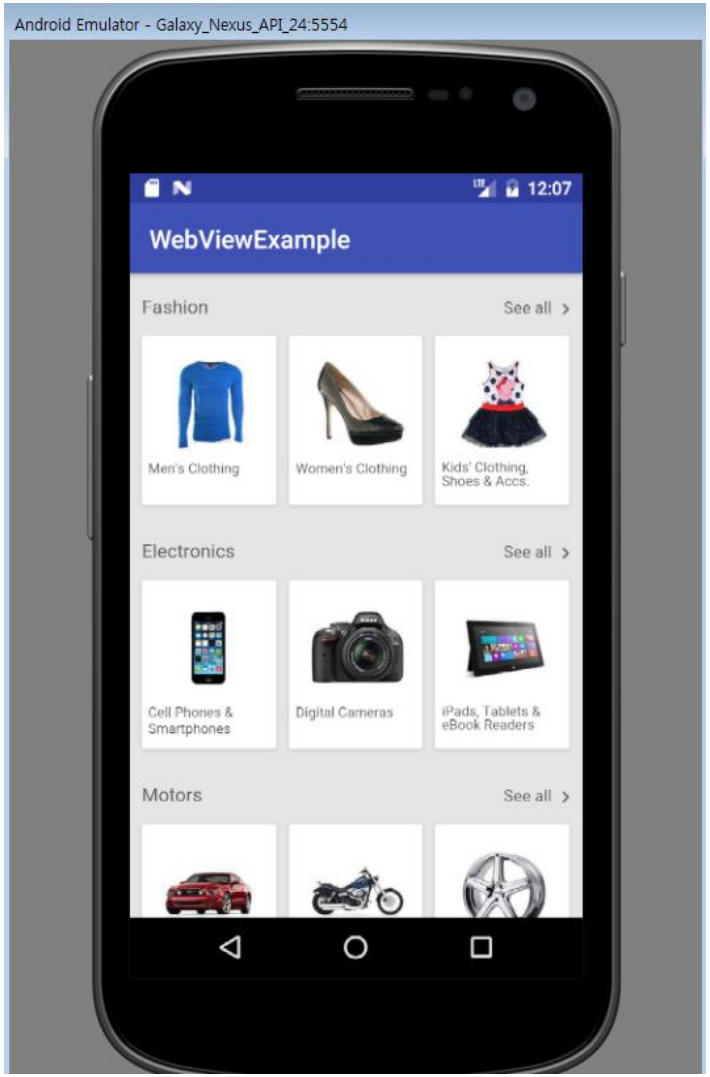
Warning !!!

In order for your Activity to access the Internet you must add the *INTERNET* permission to your Android Manifest file:

(see next example)

Exercise

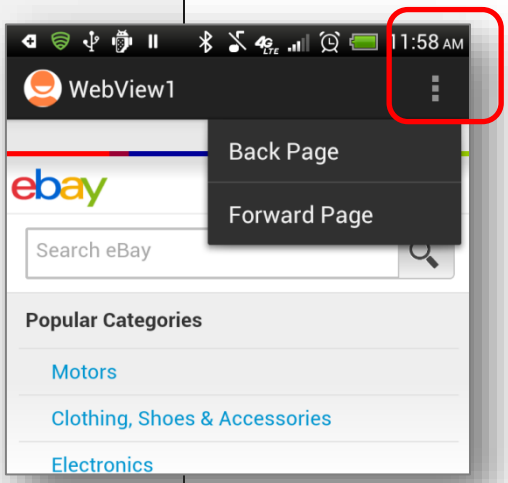
- Run!!



Exercise

- Add more functions into java file!
  - Add two menus – “Forward Page”, “Back Page”

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.menu_main, menu);
    return true;
}// onCreateOptionsMenu

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    String option = item.getTitle().toString();
    if (option.equals("Forward Page"))
        browser.goForward();

    if (option.equals("Back Page"))
        browser.goBack();

    return true;
}
```



Method supported by browser

Exercise

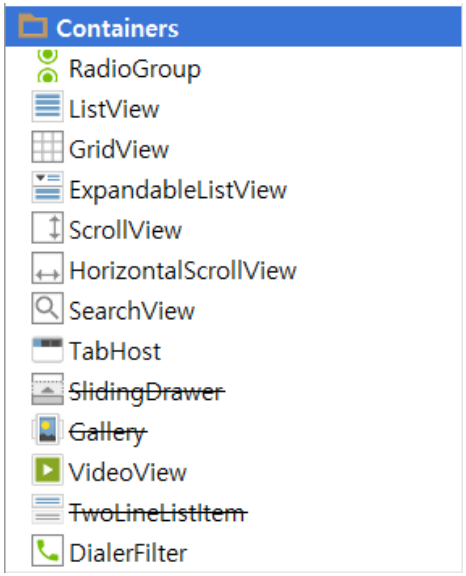
- res/menu/menu_main.xml file

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item android:id="@+id/menu_refresh"
        android:title="Forward Page"
        app:showAsAction="withText"/>
    <item android:id="@+id/menu_search"
        android:title="Back Page"
        app:showAsAction="withText"/>
</menu>
```

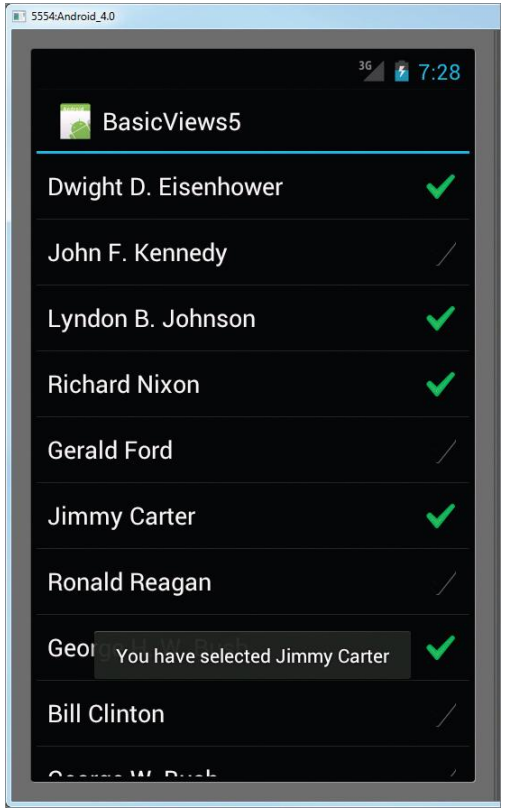
Selection Widgets

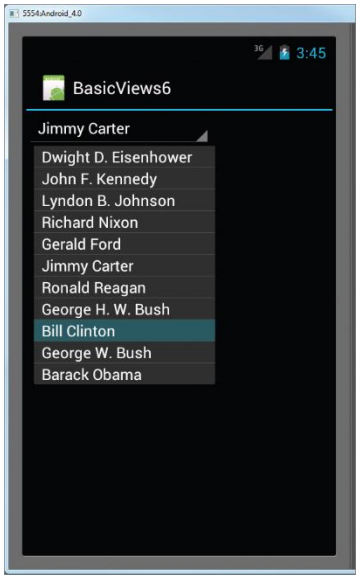
- RadioButtons and CheckButtons are suitable for selecting from a small set of options.

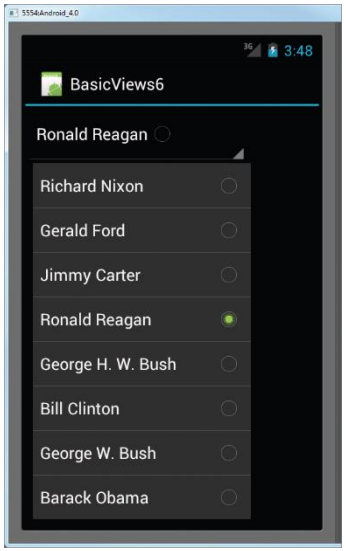
- For a larger set of choices, other Android list-based selection widgets are more appropriate.
- Example of list-based selection widgets include
  - ListViews,
  - Spinner,
  - GridView,
  - ScrollView,
  - Image Gallery, etc

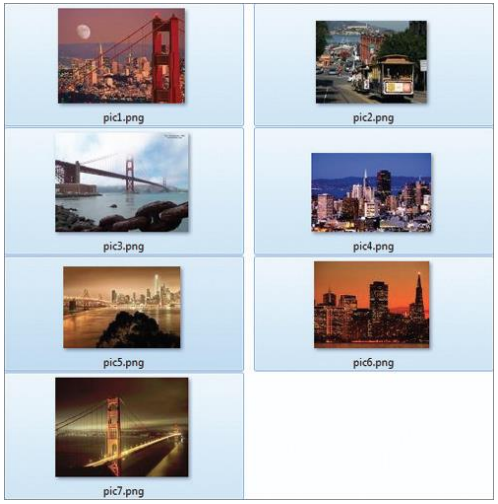


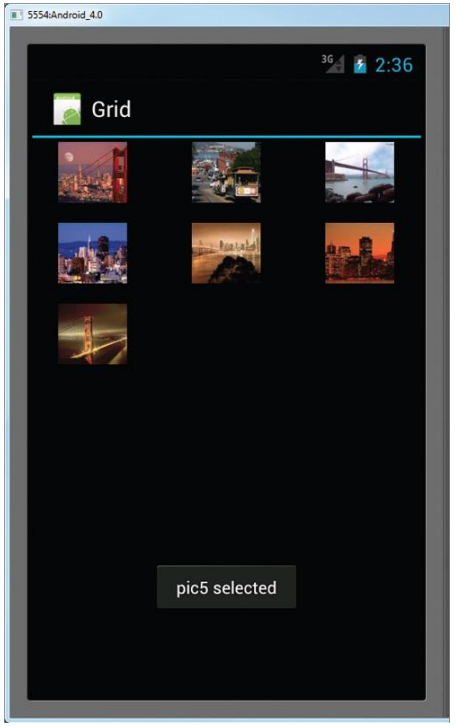
Examples


5554:Android_4.0
7:28
BasicViews5
Dwight D. Eisenhower
John F. Kennedy
Lyndon B. Johnson
Richard Nixon
Gerald Ford
Jimmy Carter
Ronald Reagan
You have selected Jimmy Carter
Bill Clinton


5554:Android_4.0
3:45
BasicViews6
Jimmy Carter
Dwight D. Eisenhower
John F. Kennedy
Lyndon B. Johnson
Richard Nixon
Gerald Ford
Jimmy Carter
Ronald Reagan
George H. W. Bush
Bill Clinton
George W. Bush
Barack Obama

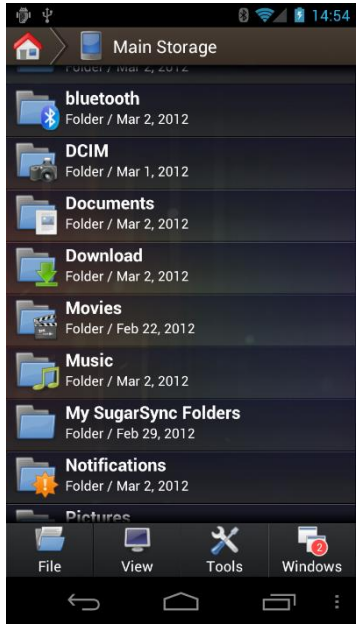

5554:Android_4.0
3:48
BasicViews6
Ronald Reagan
Richard Nixon
Gerald Ford
Jimmy Carter
Ronald Reagan
George H. W. Bush
Bill Clinton
George W. Bush
Barack Obama


pic1.png
pic2.png
pic3.png
pic4.png
pic5.png
pic6.png
pic7.png


5554:Android_4.0
2:36
Grid
pic5 selected

Widgets : ListView

- List View
  - a view group that enables you to display a long list of items.
  - displays a list of items in a vertically scrolling list

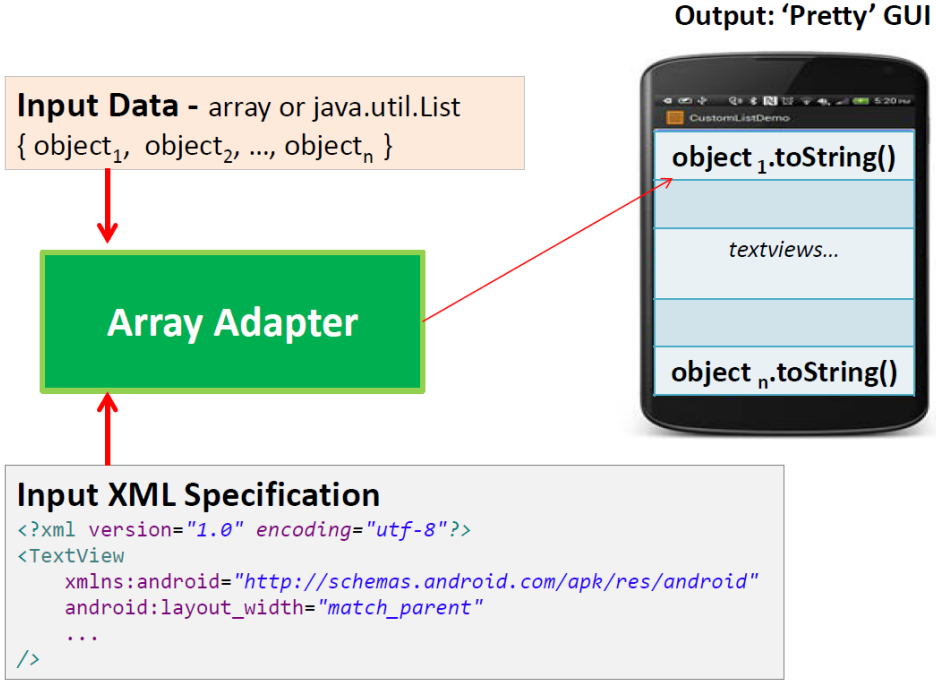


- In Android, there are two types of list views:
  - **ListView**
  - **Spinner**

- **How to insert items ??**
  - Using **adapter** !! (we will study soon)
    The list items are automatically inserted to the list using an Adapter that pulls content from a source such as an array or database query and converts each item result into a view that's placed into the list.

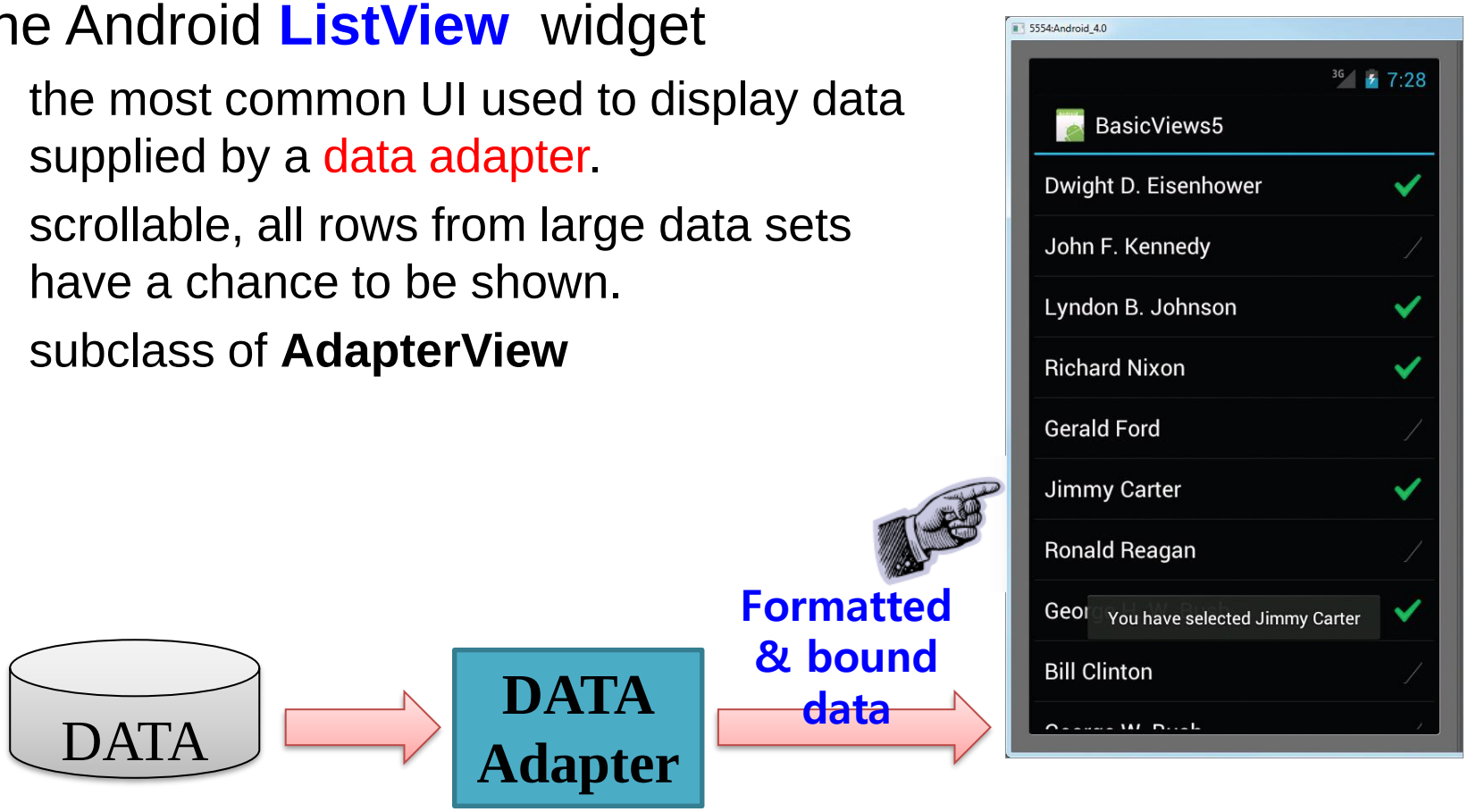
AdapterView

- AdapterView
  - A View that supports adapter
  - A ViewGroup subclass whose child Views are determined by an Adapter (that binds AdapterView object to data of some type)
- Responsibilities
  - Filling the layout with data (with a help from Adapter)
  - Handling user selections - when a user selects an item, perform some action
  - Example subclasses of *AdapterView* class
    - *ListView*
    - *Spinner*
    - *Gallery*



Widgets : ListView

- The Android **ListView** widget
  - the most common UI used to display data supplied by a data adapter.
  - scrollable, all rows from large data sets have a chance to be shown.
  - subclass of **AdapterView**



Adapters

- **Adapter**
  - An *Adapter* object acts as **a bridge** between an *AdapterView* and the underlying data for that view.
  - It provides access to the data items.
  - It is also responsible for making a View for each item in the data set.
- The adapter easiest to use is *ArrayAdapter*
  - An ArrayAdapter can be used to wrap the contents of a Java array or java.util.List and supply data rows to the UI.
- **Raw Data**

```
String[] items={"this", "is", "a","really", "silly", "list"};
```

- **Bridge between Data and View(or Layout) via ArrayAdapter**

```
ArrayAdapter<String> aa = new ArrayAdapter<String> (this,
                    android.R.layout.simple_list_item_1,
                    items);
```

- Parameters :
  - The *Context to use (typically* `this` ***will be your activity instance)***
  - The resource ID of a *view to use (such as the built-in system resource* ***android.R.layout.simple_list_item_1 as shown above)***
  - The actual (source) array or list of ***items to be shown***

Exercise

- activity_main.xml

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

    <ListView
        android:id="@+id/listview"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>

</LinearLayout>
```

List view

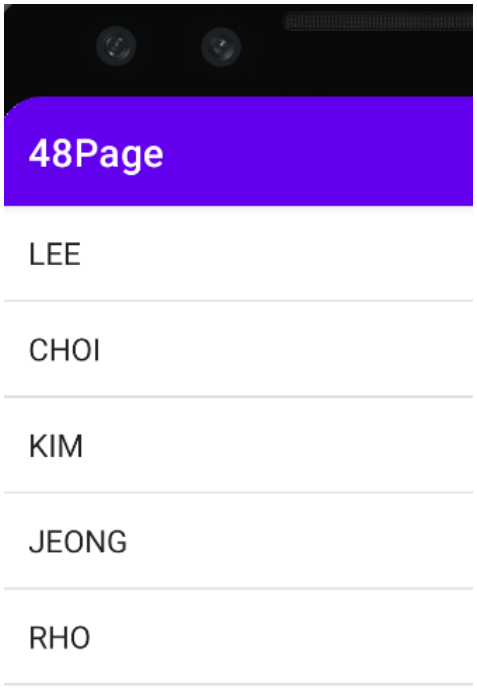
Exercise

- MainActivity.java

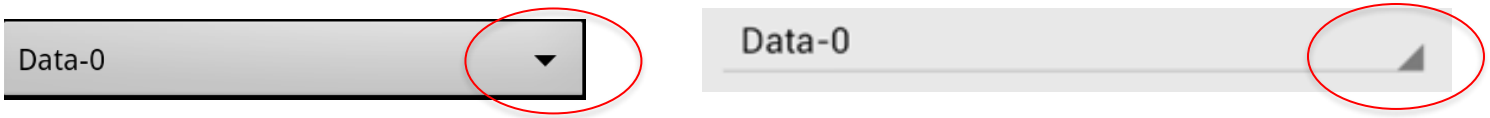
```
package com.swdm.mp.android101; // your own package name

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.ListView;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        String[] names = {"LEE", "CHOI", "KIM", "JEONG", "RHO"};
        ListView listView = (ListView) findViewById(R.id.listview);
        ArrayAdapter<String> adapter = new ArrayAdapter<>(this, android.R.layout.simple_list_item_1, names);
        listView.setAdapter(adapter);
    }
}
```

Exercise

- Run!!



Widgets: Spinner



- In Android, the **Spinner** is the equivalent of the drop-down selector.

The Spinner view
displays an item at a time from a list and lets the users choose among them.
have the same functionality of a **ListView** but take less space.

- linking data to child views : using ***setAdapter()***
  - You provide the adapter
- Add a listener object : ***setOnItemSelectedListener().***
  - to capture selections made from the list with
- The setDropDownViewResource(…) method shows the dropdown multi-line window

Widgets: Spinner

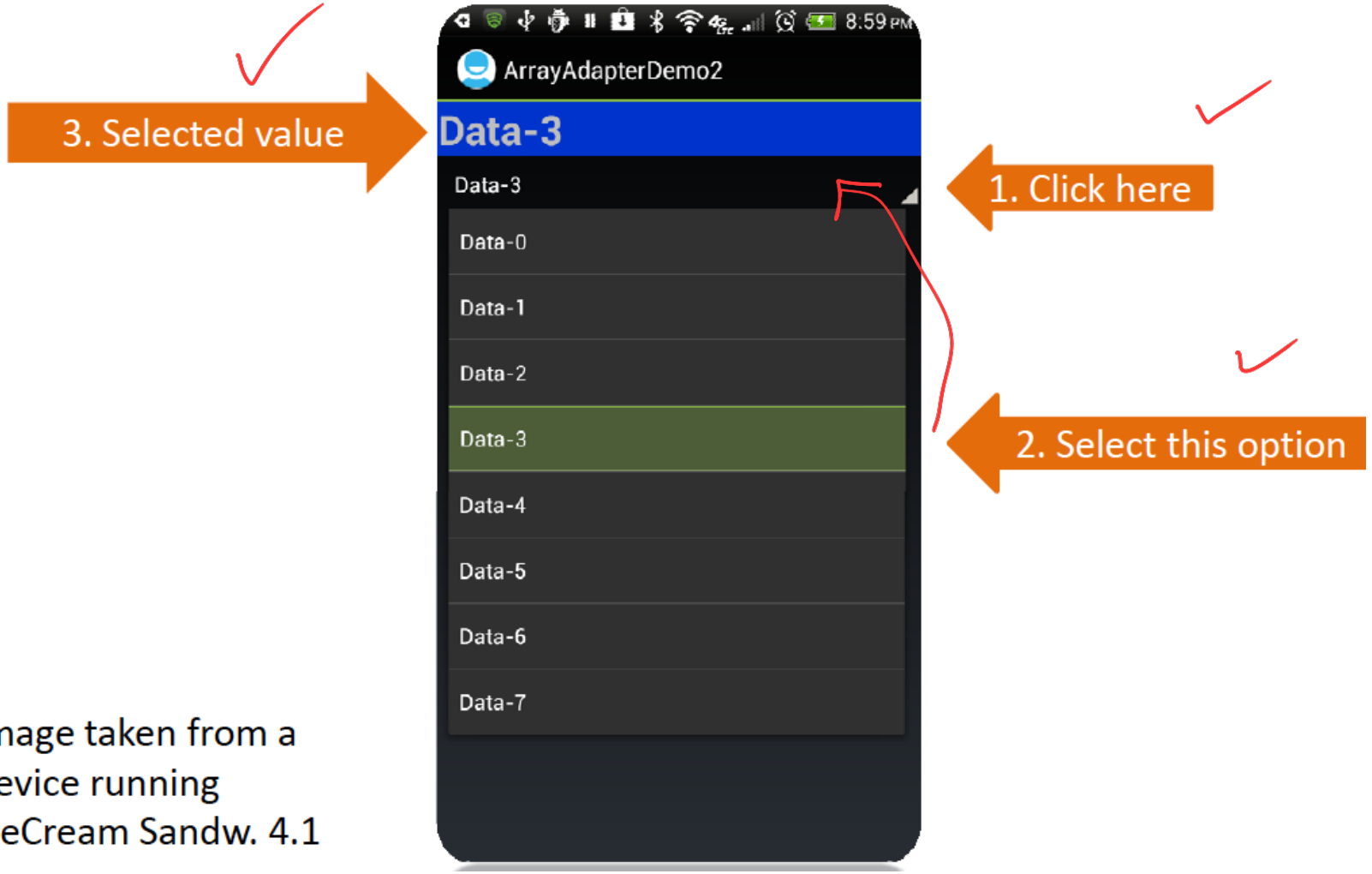


Image taken from a device running IceCream Sandw. 4.1

Exercise

- Add spinner & textView in activity_main.xml file

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

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>

    <Spinner
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/spinner"/>

</LinearLayout>
```

Exercise

- MainActivity.java

```
public class MainActivity extends AppCompatActivity {
    TextView textView;
    String[] items = {"mike", "angel", "crow", "john", "ginnie", "sally", "cohen", "rice"};
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

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

        Spinner spinner = (Spinner) findViewById(R.id.spinner);
        ArrayAdapter<String> adapter = new ArrayAdapter<>(
                this, android.R.layout.simple_spinner_item, items);
        adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
        spinner.setAdapter(adapter);

        spinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener(){

            @Override
            public void onItemSelected(AdapterView adapterView, View view, int position, long id){
                textView.setText(items[position]);
            }
            @Override
            public void onNothingSelected(AdapterView adapterView){
                textView.setText("");
            }
        });
    }
}
```

Setting adapter & attach to spinner

Spinner handling for item selection

Exercise

- RUN!!