Faculty of Computers and Information
Menoufia University

Computer Language-2

3rd Year – CS department

Course Content

- **Chapter 1: Getting Started with Android Programming**
- **Chapter 2: Using Android Studio for Android Development**
- **Chapter 3: Activities, Fragments, and Intents**
- **Chapter 4: Getting to know the Android User Interface**
- **Chapter 5: Designing Your User Interface with Views**
- **Chapter 6: Displaying Pictures and Menus with Views**
- **Chapter 7: Data Persistence**
- **Chapter 8: Content Providers**
- **Chapter 9: Messaging**
- **Chapter 10: Location-Based Services**
- **Chapter 11: Networking**
- **Chapter 12: Developing Android Services**

Agenda

- **Chapter 6 - Displaying Pictures and Menus with Views**
  - **Introduction**
  - **Using Image Views to Display Pictures**
    - ImageView View
    - ImageSwitcher
    - GridView
  - **Using Menus with Views**
    - Creating the Helper Methods
    - Options Menu
    - Context Menu
  - **Using WebView**
    - WebView

Introduction

- In this chapter, we continue the exploration of other views that you can use to create robust and compelling applications.

- In particular, you find out how to work with views that enable you to display images. Also, you see how to create option and context menus in your Android application.

- This chapter ends with a discussion of some helpful views that enable users to display web content.

Using Image Views to Display Pictures

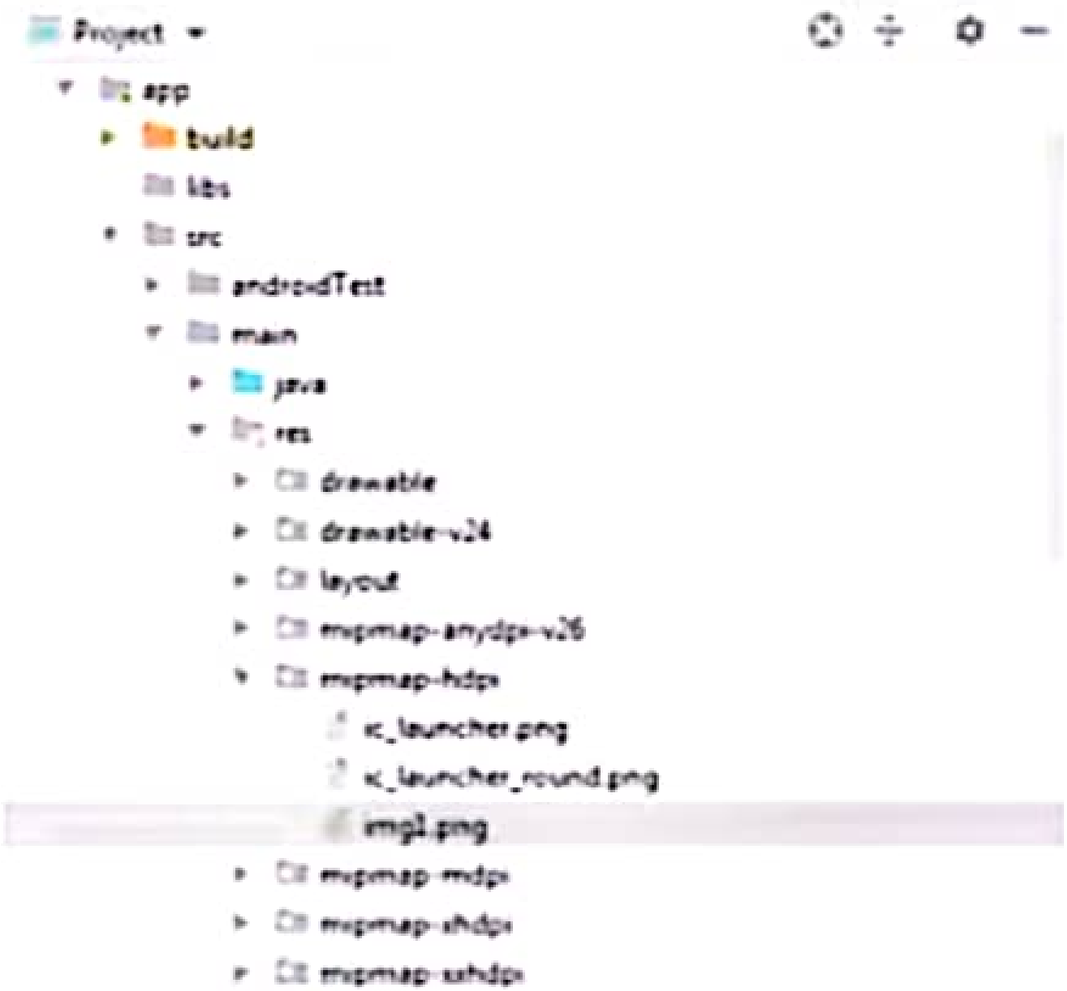
- So far, all the views you have seen are used to display text information.

- However, you can use the ImageView, ImageSwitcher, and GridView views for displaying images.

ImageView View

ImageView View

- The ImageView is a view that shows images on the device screen.
- Add an image to your project under the res/mipmap folder.
- Note that you must be in project view to drag and drop images into the res/mipmap folder.



ImageView View

- ✓ First, you will add image to res/mipmap.
- ✓ Then you will create an ImageView in activity_main.xml and add image on it.
- ✓ Finally, the MainActivity.java and AndroidManifest.xml will remain intact

MainActivity

activity_main.xml

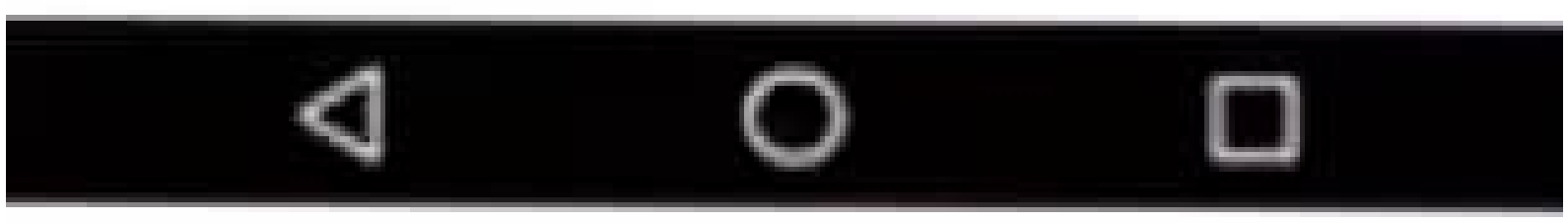
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="wrap_content"
    android:layout_height="match_parent"
    tools:context=".MainActivity"
    android:orientation="horizontal">
    <ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        app:srcCompat="@mipmap/img1" />
</LinearLayout>
```

AndroidManifest.xml

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

MainActivity.java

```
package fci.third.imageviewtest;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

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



ImageSwitcher

- ImageSwitcher is used to appear an image abruptly when the user opens the view.
  - For example, you might want to apply some animation to an image when it transitions from one image to another.

- Add two images to your res/mipmap folder. For this example, I added an image named img1.png and an image named img2.jpg.

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View Image 1"
        android:onClick="onClickImg1" />
```

ImageSwitcher

An animation happens to an image when it transitions from one image to another

activity_main.xml

```
<Button
    android:id="@+id/button2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentTop="true"
    android:layout_alignParentEnd="true"
    android:layout_alignParentRight="true"
    android:text="View Image 2"
    android:onClick="onClickImg2" />
    <ImageSwitcher
        android:id="@+id/imageSwitcher"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_below="@+id/button2"
        android:layout_alignParentStart="true"
        android:layout_alignParentLeft="true">
    </ImageSwitcher>
</RelativeLayout>
```

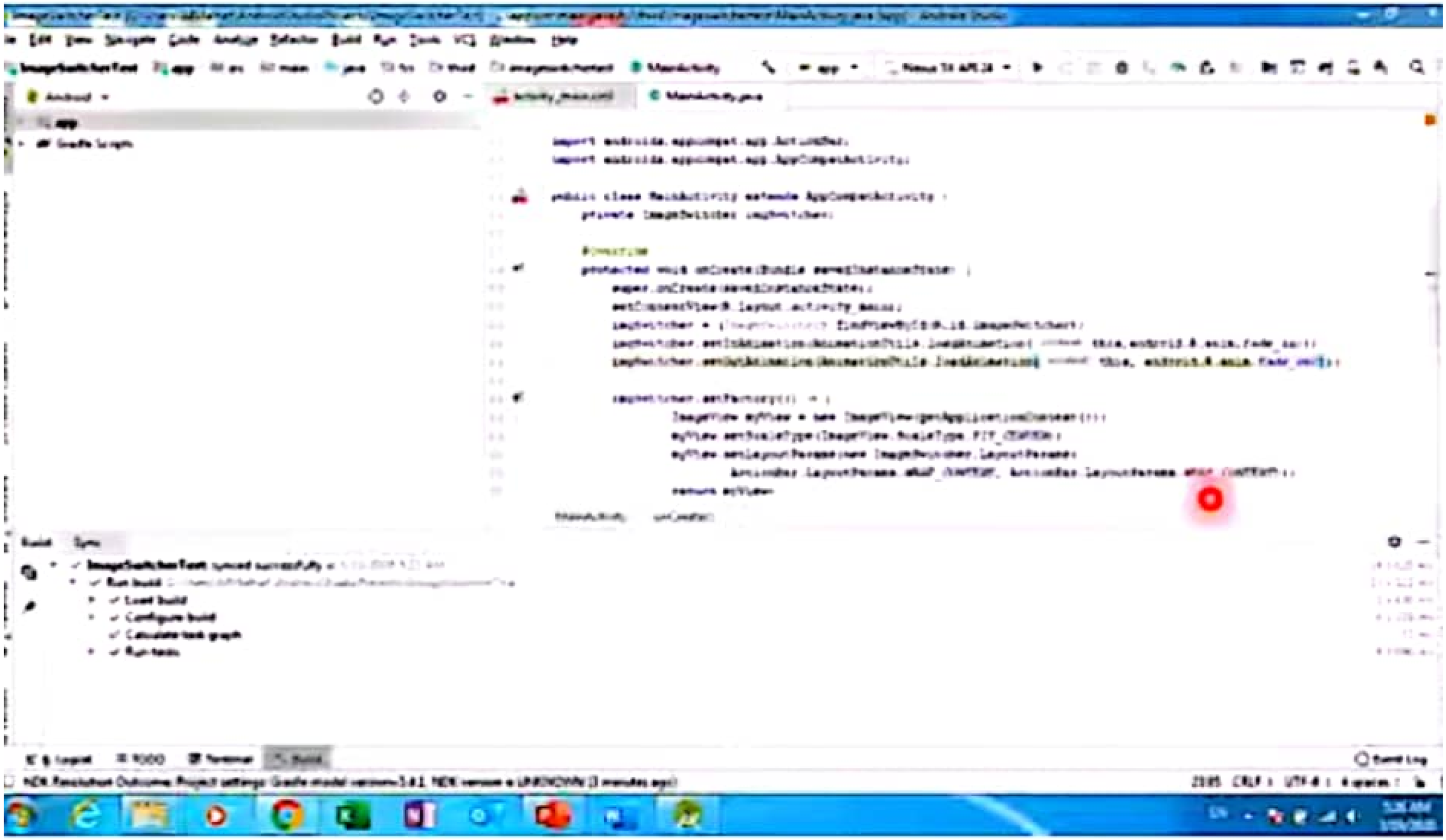
MainActivity.java

```
package fci.third.imageswitchertest;
import android.os.Bundle;
import android.view.View;
import android.view.animation.AnimationUtils;
import android.widget.ImageSwitcher;
import android.widget.ImageView;
import android.widget.Toast;
import android.widget.ViewSwitcher;
import androidx.appcompat.app.ActionBar;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity{
    private ImageSwitcher imgSwitcher;
    @Override
    protected void onCreate(Bundle savedInstanceState){
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
```

AndroidManifest.xml

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

MainActivity.java



MainActivity.java

```
public void onClickImg1(View view) {
    Toast.makeText(getApplicationContext(), "View Image 1",
                   Toast.LENGTH_LONG).show();
    imgSwitcher.setImageResource(R.mipmap.img1);
}

public void onClickImg2(View view) {
    Toast.makeText(getApplicationContext(), "View Image 2",
                   Toast.LENGTH_LONG).show();
    imgSwitcher.setImageResource(R.mipmap.img2);

    }
}
```

MainActivity.java

```
imgSwitcher = (ImageSwitcher) findViewById(R.id.imageSwitcher);
imgSwitcher.setInAnimation(AnimationUtils.loadAnimation(this,
 android.R.anim.fade_in));
imgSwitcher.setOutAnimation(AnimationUtils.loadAnimation(this,
android.R.anim.fade_out));
imgSwitcher.setFactory(new ViewSwitcher.ViewFactory(){
    @Override
    public View makeView(){
        ImageView myView = new ImageView(getApplicationContext());
        myView.setScaleType(ImageView.ScaleType.FIT_CENTER);
        myView.setLayoutParams(new ImageSwitcher.LayoutParams(
            ActionBar.LayoutParams.WRAP_CONTENT,
            ActionBar.LayoutParams.WRAP_CONTENT));
        return myView;
    }
});
}
```

GridView

- The GridView shows items in a two-dimensional scrolling grid.
- You can use the GridView together with an ImageView to display a series of images.

ImageSwitcher

- In this example, when an image is selected in the Gallery view, it appears by "fading" in. When the next image is selected, the current image fades out.
- If you want the image to slide in from the left and slide out to the right when another image is selected, try the following animation:

```
imageSwitcher.setInAnimation(AnimationUtils.loadAnimation(this
,android.R.anim.slide_in_left));
imageSwitcher.setOutAnimation(AnimationUtils.loadAnimation(th
is, android.R.anim.slide_out_right));
```

ImageSwitcher

- This method creates a new View to be added in the ImageSwitcher view, which in this case is an ImageView.
- In the onCreate() method, you get a reference to the ImageSwitcher view and set the animation, specifying how images should **fade in** and **out** of the view.
- Finally, when an image is selected from the Gallery view, the image is displayed in the ImageSwitcher view.

ImageSwitcher

- To use the ImageSwitcher view, you need to implement the ViewFactory interface, which creates the views for use with the ImageSwitcher view. For this, you need to implement the makeView() method:

```
imgSwitcher.setFactory(new ViewSwitcher.ViewFactory() {
        @Override
        public View makeView() {
                ImageView myView = new ImageView(getApplicationContext());
                myView.setScaleType(ImageView.ScaleType.FIT_CENTER);
                myView.setLayoutParams(new ImageSwitcher.LayoutParams(
                   ActionBar.LayoutParams.WRAP_CONTENT,
                   ActionBar.LayoutParams.WRAP_CONTENT));
                return myView;
        }
});
```

activity_main.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:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <GridView
        android:id="@+id/gridview"
        android:layout_width="384dp"
        android:layout_height="511dp" />
</LinearLayout>
```

GridView

GridView

- The GridView shows items in a two-dimensional scrolling grid.
- You can use the GridView together with an ImageView to display a series of images.

MainActivity.java

```
//---returns an ImageView view---
public View getView(int position, View convertView,
        ViewGroup parent) {
    ImageView imageView;
    if (convertView == null) {
        imageView = new ImageView(context);
        imageView.setLayoutParams(new GridView.LayoutParams(500, 300));
        imageView.setScaleType(ImageView.ScaleType.CENTER_CROP);
        imageView.setPadding(5, 5, 5, 5);
    } else {
        imageView = (ImageView) convertView;
    }
    imageView.setImageResource(imageIDs[position]);
    return imageView;
  }
 }
}
```

MainActivity.java

```
public class ImageAdapter extends BaseAdapter {
    private Context context;
    public ImageAdapter(Context c){
        context = c;
    }
    //---returns the number of images---
    public int getCount() {
        return imageIDs.length;
    }
    //---returns the item---
    public Object getItem(int position) {
        return position;
    }
    //---returns the ID of an item---
    public long getItemId(int position) {
        return position;
    }
```

MainActivity.java

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    GridView gridView = (GridView)  findViewById(R.id.gridview);
    gridView.setAdapter(new ImageAdapter(this));
    gridView.setOnItemClickListener(new AdapterView.OnItemClickListener()
    {
        public void onItemClick(AdapterView parent, View v, int position, long id){
            Toast.makeText(getBaseContext(), "pic" + (position + 1) + " selected",
Toast.LENGTH_SHORT).show();
        }
    });
  }
```

MainActivity.java

```
package fci.third.gridviewtest;
import android.content.Context;
import android.os.Bundle;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.BaseAdapter;
import android.widget.GridView;
import android.widget.ImageView;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    //---the images to display---
    Integer[] imageIDs = {R.mipmap.img1, R.mipmap.img2, R.mipmap.img3,
R.mipmap.img4, R.mipmap.img5, R.mipmap.img6};
```

Creating the Helper Methods

- To create a **list of items** to show inside a menu, you will create method called **createMenu** in the target activity.

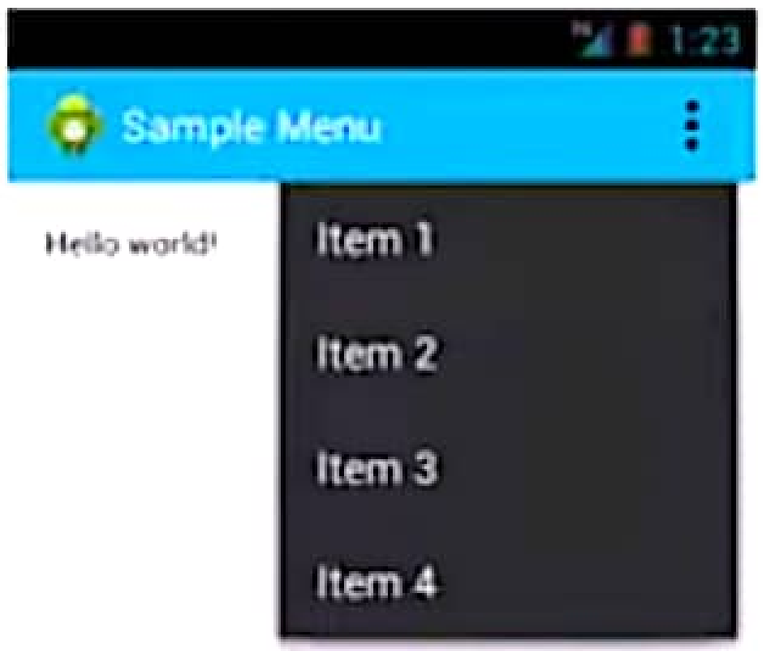
```
public void createMenu(Menu menu) {
        menu.add(int GroupID, int ItemID, int Order, String title);
        .
        .
}
```

```
menu.add(0, 1, 1, “Item 1”);
menu.add(0, 2, 2, “Item 2”);
menu.add(0, 3, 3, “Item 3”);
menu.add(0, 4, 4, “Item 4”);
```

Creating the Helper Methods

- Before you go ahead and create your options and context menus, you need to create two helper methods.
    1. The first one creates a **list of items** to show inside a menu.
    2. The second one **handles the event** that is fired when the user selects an item inside the menu.

Using Menus with Views



Options Menu

Content Menu

Using Menus with Views

- Menus are useful for displaying **additional options** that are not directly visible on the main user interface (UI) of an application. There are two main types of menus in Android:
  - **Options menu**—This menu displays information related to the **current activity**. In Android, you activate the options menu by pressing the Menu button.
  - **Context menu**—This menu displays information related to a **particular view on an activity**. In Android, you tap and hold a context menu to activate it.

MainActivity.java

```
case 2:
   Toast.makeText(this, "You clicked on Item 3",Toast.LENGTH_LONG).show();
   return true;
case 3:
   Toast.makeText(this, "You clicked on Item 4",Toast.LENGTH_LONG).show();
   return true;
case 4:
   Toast.makeText(this, "You clicked on Item 5",Toast.LENGTH_LONG).show();
   return true;
      case 5:
         Toast.makeText(this, "You clicked on Item 6",Toast.LENGTH_LONG).show();
         return true;
      case 6:
         Toast.makeText(this, "You clicked on Item 7",Toast.LENGTH_LONG).show();
         return true;
   }
   return false;
  }
}
```

MainActivity.java

```
private void createMenu(Menu menu) {
    menu.add(0, 0, 0, "Item 1");
    menu.add(0, 1, 1, "Item 2");
    menu.add(0, 2, 2, "Item 3");
    menu.add(0, 3, 3, "Item 4");
    menu.add(0, 4, 4, "Item 5");
    menu.add(0, 5, 5, "Item 6");
    menu.add(0, 6, 6, "Item 7");
}
private boolean MenuChoice(MenuItem item) {
    switch (item.getItemId()) {
        case 0:
            Toast.makeText(this, "You clicked on Item 1",Toast.LENGTH_LONG).show();
            return true;
        case 1:
            Toast.makeText(this, "You clicked on Item 2",Toast.LENGTH_LONG).show();
            return true;
```

MainActivity.java

```
package fci.third.menutest;

import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
```

Creating the Helper Methods

- To handles the event that is fired, you will create method called **MenuChoice** in the target activity.

```
private boolean MenuChoice(MenuItem item) {
    switch (item.getItemId()) {
      case 0:
        -- Action
        return true;
      case 1:
        -- Action
        return true;
        .
    }
}
```

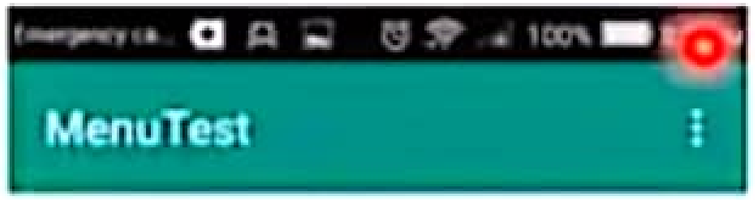
MainActivity.java

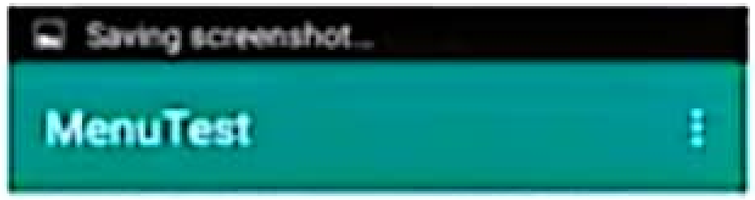
```
@Override
public boolean onCreateOptionsMenu(Menu menu){
    super.onCreateOptionsMenu(menu);
    createMenu(menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item){
    return MenuChoice(item);
}
```

MainActivity.java

```
package fci.third.menutest;

import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
```

Option Menus





Option Menus

- You are now ready to modify the application to display the options menu when the user presses the Menu key on the Android device.
- To create OptionMenus, you will override two methods:

called when the Menu button is pressed

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
   // here you will create menu by calling createMenu() method
}
```

called when a Menu item is selected

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
   // you will call menuChoice() method and passing to it item selected
}
```

Context Menus

- To create ContextMenu, you will override two methods:

Called when the user taps and holds the associated view

```
@Override
public void onCreateContextMenu(ContextMenu menu, View view,
                                ContextMenu.ContextMenuInfo menuInfo) {
  // here you will create menu by calling createMenu() method
}
```

called when a Menu item is selected

```
@Override
public boolean onContextItemSelected(MenuItem item){
  // you will call menuChoice() method and passing to it item selected
}
```

Context Menu

- A context menu is usually associated with a view on an activity. A context menu is displayed when the user taps and holds an item.
  - For example, if the user taps a Button view and holds it for a few seconds, a context menu can be displayed.
- If you want to associate a context menu with a view on an activity, you need to call the **setOnCreateContextMenuListener()** method of that particular view.

MainActivity.java

```
case 2:
    Toast.makeText(this, "You clicked on Item 3",Toast.LENGTH_LONG).show();
    return true;
case 3:
    Toast.makeText(this, "You clicked on Item 4",Toast.LENGTH_LONG).show();
    return true;
case 4:
    Toast.makeText(this, "You clicked on Item 5",Toast.LENGTH_LONG).show();
    return true;
        case 5:
            Toast.makeText(this, "You clicked on Item 6",Toast.LENGTH_LONG).show();
            return true;
        case 6:
            Toast.makeText(this, "You clicked on Item 7",Toast.LENGTH_LONG).show();
            return true;
    }
    return false;
  }
}
```

MainActivity.java

```
private void createMenu(Menu menu) {
    menu.add(0, 0, 0, "Item 1");
    menu.add(0, 1, 1, "Item 2");
    menu.add(0, 2, 2, "Item 3");
    menu.add(0, 3, 3, "Item 4");
    menu.add(0, 4, 4, "Item 5");
    menu.add(0, 5, 5, "Item 6");
    menu.add(0, 6, 6, "Item 7");
}
private boolean MenuChoice(MenuItem item) {
    switch (item.getItemId()) {
        case 0:
            Toast.makeText(this, "You clicked on Item 1",Toast.LENGTH_LONG).show();
            return true;
        case 1:
            Toast.makeText(this, "You clicked on Item 2",Toast.LENGTH_LONG).show();
            return true;
```

MainActivity.java

```
package fci.third.menutest;
import android.os.Bundle;
import android.view.ContextMenu;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Button btn = (Button) findViewById(R.id.button);
        btn.setOnCreateContextMenuListener(this);
    }
```

activity_main.xml

```
<Button
    android:text="Button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    tools:layout_editor_absoluteX="148dp"
    tools:layout_editor_absoluteY="102dp"
    android:id="@+id/button"
    app:layout_constraintLeft_toLeftOf="@+id/activity_main"
    tools:layout_constraintLeft_creator="0"
    app:layout_constraintRight_toRightOf="@+id/activity_main"
    tools:layout_constraintRight_creator="0"
    tools:ignore="MissingConstraints" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

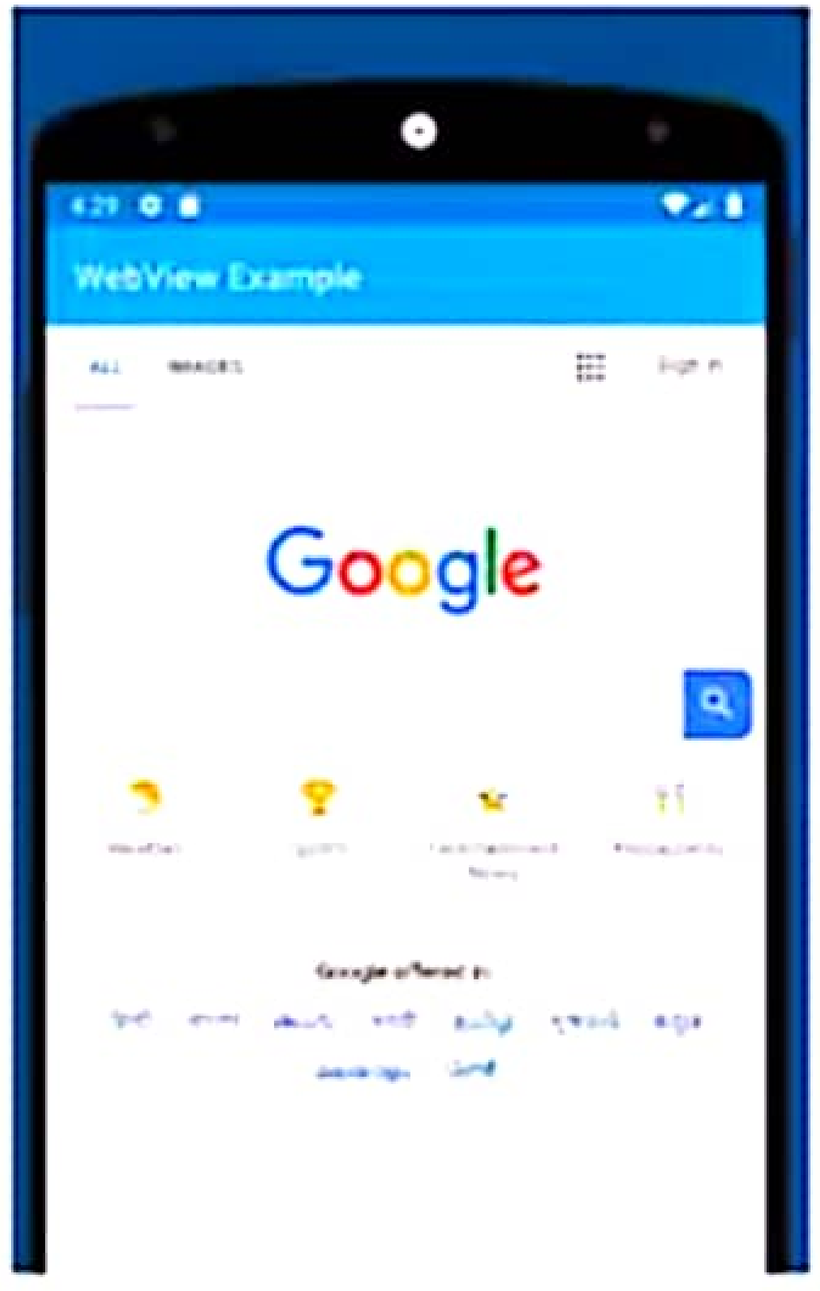
activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.constraintlayout.widget.ConstraintLayout 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:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity"
    tools:layout_editor_absoluteX="0dp"
    tools:layout_editor_absoluteY="81dp">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        tools:layout_editor_absoluteX="154dp"
        tools:layout_editor_absoluteY="247dp"
        app:layout_constraintLeft_toLeftOf="@+id/activity_main"
        tools:layout_constraintLeft_creator="0"
        app:layout_constraintTop_toTopOf="@+id/activity_main"
        tools:layout_constraintTop_creator="0"
        app:layout_constraintRight_toRightOf="@+id/activity_main"
        tools:layout_constraintRight_creator="0"
        app:layout_constraintBottom_toBottomOf="@+id/activity_main"
        tools:layout_constraintBottom_creator="0" />
```

Context Menu

Using WebView

- Aside from the standard views that you have seen up to this point, the Android SDK provides some additional views that make your applications much more interesting.
- The **WebView** enables you to **embed a web browser in your activity**. This is very useful if your application needs to embed some web content, such as maps from some other providers, and so on.



MainActivity.java

```
case 2:
   Toast.makeText(this, "You clicked on Item 3",Toast.LENGTH_LONG).show();
   return true;
case 3:
   Toast.makeText(this, "You clicked on Item 4",Toast.LENGTH_LONG).show();
   return true;
case 4:
   Toast.makeText(this, "You clicked on Item 5",Toast.LENGTH_LONG).show();
   return true;
      case 5:
         Toast.makeText(this, "You clicked on Item 6",Toast.LENGTH_LONG).show();
         return true;
      case 6:
         Toast.makeText(this, "You clicked on Item 7",Toast.LENGTH_LONG).show();
         return true;
    }
    return false;
  }
}
```

MainActivity.java

```
private void createMenu(Menu menu) {
    menu.add(0, 0, 0, "Item 1");
    menu.add(0, 1, 1, "Item 2");
    menu.add(0, 2, 2, "Item 3");
    menu.add(0, 3, 3, "Item 4");
    menu.add(0, 4, 4, "Item 5");
    menu.add(0, 5, 5, "Item 6");
    menu.add(0, 6, 6, "Item 7");
}
private boolean MenuChoice(MenuItem item) {
    switch (item.getItemId()) {
        case 0:
            Toast.makeText(this, "You clicked on Item 1",Toast.LENGTH_LONG).show();
            return true;
        case 1:
            Toast.makeText(this, "You clicked on Item 2",Toast.LENGTH_LONG).show();
            return true;
```

MainActivity.java

```
@Override
public void onCreateContextMenu(ContextMenu menu, View view,
                                ContextMenu.ContextMenuInfo menuInfo){
   super.onCreateContextMenu(menu, view, menuInfo);
   createMenu(menu);
}

@Override
public boolean onContextItemSelected(MenuItem item) {
   return MenuChoice(item);
}
```

MainActivity.java

```
package fci.third.webviewtest;
import android.os.Bundle;
import android.webkit.WebSettings;
import android.webkit.WebView;
import androidx.appcompat.app.AppCompatActivity;
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        WebView wv = (WebView) findViewById(R.id.webview);
        WebSettings webSettings = wv.getSettings();
        webSettings.setBuiltInZoomControls(true);
        wv.loadUrl(
"http://chart.apis.google.com/chart?chs=300x225&cht=v&chco=FF6342,ADDE63,63C6DE"
+"&chd=t:100,80,60,30,30,30,10&chdl=A|B|C");
    }
}
```

activity_main.xml

```
app:layout_constraintBottom_toBottomOf="@+id/activity_main"
app:layout_constraintLeft_toLeftOf="@+id/activity_main"
app:layout_constraintRight_toRightOf="@+id/activity_main"
app:layout_constraintTop_toTopOf="@+id/activity_main"
tools:layout_constraintBottom_creator="0"
tools:layout_constraintLeft_creator="0"
tools:layout_constraintRight_creator="0"
tools:layout_constraintTop_creator="0"
tools:layout_editor_absoluteX="0dp"
tools:layout_editor_absoluteY="0dp" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
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:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity"
    tools:layout_editor_absoluteX="0dp"
    tools:layout_editor_absoluteY="81dp">

<WebView
    android:id="@+id/webview"
    android:layout_width="384dp"
    android:layout_height="511dp"
```

t.apis.google.com

Thank You

WebView

- To use the WebView to load a web page, you use the **loadUrl()** method and **pass a URL** to it.
- To display the built-in zoom controls, you need to first get the **WebSettings** property from the WebView and then call its **setBuiltInZoomControls()** method.
- Note: Although most Android devices support multitouch screens, the builtin zoom controls are useful for zooming your web content when testing your application on the Android emulator.

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="fci.third.webviewtest">
    <uses-permission android:name="android.permission.INTERNET"/>
    <application
        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>
```