Adding Components and Windows

Creating another Activity Window

* name sure you have a name picked first, and a general layout
  + name must be lowercase
  + default layout is LinearLayout
* should show up in res/layout in Eclipse project
  + beside any other Activity Windows you have created
* instead of manually creating each class, xml, and modifying the manifest, just go to New->Other->Android->Activity
* Copy and Paste, ***not so much***
  + if your activity windows are similar, Eclipse’s copy and paste saved me a ton of time (so I thought)
  + got ONE (food) of the activity windows and layout done
    - then copied and changed a little code
  + ***BUT I GOT ERRORS!!***

Creating a Project with Multiple Windows

* as usual, have a game plan in mind
* plan (in order)
  + activity windows
  + what the windows will contain (images, buttons, etc…)
  + how do they interact with each other

|  |  |
| --- | --- |
| Lupoli’s Favorite App (in Planning) | |
| Activity Window | Windows Contain |
|  |  |
| How the windows are linked | Finished Product |
|  |  |

|  |
| --- |
| Lupoli’s Favorite App (Eclipse Setup) |
|  |

Creating your first Application - Layout

* programming Androids in Java reminds me of Visual Basic
  + every GUI component has a name and many properties
* anything with GUI requires overall planning of
  + layout
    - done in XML, user easy to edit and understand
  + naming schemes
    - windows
    - buttons
    - etc…
  + properties
* compared to Java SWING programming, there is a graphical IDE that allows you to use the mouse to create the overall layout and look!!!

Layout (for Windows and Components) Options

* width and height are common

|  |
| --- |
| Layout width (and height) |
| android:layout\_width=*"match\_parent"*  ...  android:layout\_width=*"wrap\_content"*  *...*  android:layout\_width=*"fill\_parent"* |

Create an activity. Place two “textviews” on the activity window. Change the layout\_widths of ONE of the textviews to view what happens for each option.

* gravity
  + position of the text
    - center, left, right, bottom, top
* background
  + much like an HTML file’s background
  + will reference “drawable” folder and whatever image you have in it
    - weird thing, do not have the file extension in the code
    - just the filename
* but within ANY component, type “android:” and the handy list of options will appear

The Java Files

* any actions are coding in the java files
* file is automatically created when you add an activity
* name of the class should follow the name of the page
  + does not need a main
* the file is broken up into 5 parts
  + imports
  + class
  + GUI Components
  + onCreate (think of as a constructor)
  + some type of ActionListener

|  |
| --- |
| The Java File Setup |
| **import** android.os.Bundle;  **import** android.app.Activity;  **import** android.content.Intent;  **import** android.view.Menu;  **import** android.view.View;  **import** android.view.View.OnClickListener;  **import** android.widget.Button; |
| **public** **class** Main **extends** Activity {  // setup Base components here so everyone in the class has access  Button btnFood, btnToy, btnSite; |
| **public** **void** onCreate(Bundle savedInstanceState)  {  **super**.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);    btnFood = (Button) findViewById(R.id.*btnFood*);  btnToy = (Button) findViewById(R.id.*btnToy*);  btnSite = (Button) findViewById(R.id.*btnSite*);    btnFood.setOnClickListener(**new** ButtonListener());  btnToy.setOnClickListener(**new** ButtonListener());  btnSite.setOnClickListener(**new** ButtonListener());  } |
| **private** **class** ButtonListener **implements** OnClickListener  {  **public** **void** onClick(View V)  {  //if(V.getId() == R.id.btnFood) works too    **if**(btnFood.getId() == ((Button)V).getId())  { startActivity(**new** Intent(Main.**this**, Food.**class**)); }  **else** **if**(btnToy.getId() == ((Button)V).getId())  { startActivity(**new** Intent(Main.**this**, Toy.**class**)); }  **else** **if**(btnSite.getId() == ((Button)V).getId())  { startActivity(**new** Intent(Main.**this**, Site.**class**)); }  **else** {}  }    } |

Components in the Layout/XML

* Anything you add to the window, needs a unique name UNLESS it’s has no action happening to/from it
  + all buttons should have unique names
* each GUI component has a “property” matrix that we can set WITHOUT coding!! (pic)
  + to see “properties” make sure the Structure tab is expanded
  + much easier to work with than the right click menu
  + also, if you have many buttons that you need to do the same property, select one, edit, then select the next button in your layout and it goes to the SAME property

|  |
| --- |
| Components Properties by IDE |
|  |

|  |
| --- |
| Components Properties by XML |
| <Button  android:id=*"@+id/btnFood"*  android:layout\_width=*"wrap\_content"*  android:layout\_height=*"wrap\_content"*  android:layout\_below=*"@+id/textView1"*  android:layout\_centerHorizontal=*"true"*  android:layout\_marginTop=*"160dp"*  android:text=*"Favorite Food"* />  <Button  android:id=*"@+id/btnToy"*  android:layout\_width=*"wrap\_content"*  android:layout\_height=*"wrap\_content"*  *...* |

Common General Properties needed

* ID
  + at top of the list
  + buttons usually have an @+id/**btn**someName
    - convention is @+id/**btn**textOnButton
      * @+id/Food
* Text
  + what is written on the button/textview
* notice again, that you can change this in the XML document

Linking the Components to the Java File

* the buttons on the layout are useless with having actions assigned to them
* any action items like buttons need to be assigned a listener
* notice the position of
  + creation of the JAVA portion of the buttons
  + instantiation the buttons
    - in onCreate
  + adding them to an listener
  + actionlistener

|  |
| --- |
| Component Layout in Java |
| **...**  **public** **class** Main **extends** Activity {  // setup Base components here so everyone in the class has access  Button btnFood, btnToy, btnSite;    **public** **void** onCreate(Bundle savedInstanceState)  {  **super**.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);    btnFood = (Button) findViewById(R.id.*btnFood*);  btnToy = (Button) findViewById(R.id.*btnToy*);  btnSite = (Button) findViewById(R.id.*btnSite*);    btnFood.setOnClickListener(**new** ButtonListener());  btnToy.setOnClickListener(**new** ButtonListener());  btnSite.setOnClickListener(**new** ButtonListener());  }  **private** **class** ButtonListener **implements** OnClickListener  {  **public** **void** onClick(View V)  {  **if**(btnFood.getId() == ((Button)V).getId())  { startActivity(**new** Intent(Main.**this**, Food.**class**)); }  **else** **if**(btnToy.getId() == ((Button)V).getId())  ...  }    }    } |

What’s the deal with onCreate?

* think of this as a constructor, but for an Activity Window
* items to accomplish in onCreate()
  + open the current class’s activity window
    - setContentView
  + instantiate the GUI components
  + add the components to a listener

|  |
| --- |
| Android Activity Life Cycle |
| Android lifecycle |

* application starts
  + calls onCreate, onStart, onResume
  + app becomes active and is what you see
* application is interrupted
  + phone call comes in, something that takes precedence
  + onPause is called
* application is terminated
  + onStop is called
  + onStop has two options, onDestroy() and onRestart()
  + onDestroy is the very last function to be called
    - usually saves any data here
    - remember this could happen if an application crashes as well
* if you want to program a certain method, you can add inside your Java file
  + Source 🡪 Override/Implement

Component Tools

* there are some nice quick tools on the IDE that really help in laying out
* what tools appear depends on the GUI Component

Creating a Button Event Handler (ActionListener)

* import
  + android.widget.Button;
  + A BUNCH MORE!!!
* inside the respective .java files
* create a reference to the layout button we created
  + INSIDE oncreate function
  + named the layout button the same as a “java” button
    - btnFood
* in setOnClickListener
  + starts new activity

|  |
| --- |
| First ActionListener Example |
| **public** **class** Main **extends** Activity {  // setup Base components here so everyone in the class has access  Button btnFood;    **public** **void** **onCreate**(Bundle savedInstanceState) {  **super**.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);    btnFood = (Button) findViewById(R.id.*btnFood*);    btnFood.setOnClickListener(**new** View.OnClickListener()  {  **public** **void** onClick(View v)  { startActivity(**new** Intent(Main.**this**, Food.**class**)); }    });  } |

Text Size

Have them disscribe the different text, widths, etc…

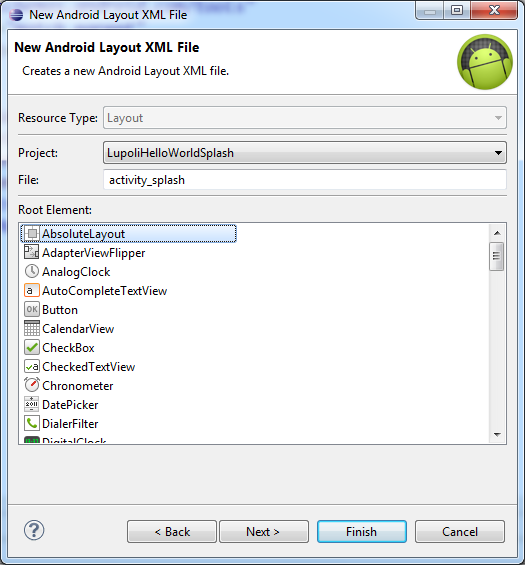
type android: (then get list)

|  |
| --- |
| Eclipse overall project Layout - Splash |
|  |

Start with the Layouts

* fun first
* program later

Adding a new Layout



|  |
| --- |
|  |
| <activity  android:name=*".Splash"*  android:label=*"@string/title\_activity\_splash"* >  <intent-filter>  <action android:name=*"com.example.lupolihelloworldsplash.SPLASH"* />  <category android:name=*"android.intent.category.DEFAULT"* />  </intent-filter>  </activity> |
| <?xml version="1.0" encoding="UTF-8" standalone="no"?>  <resources>  <string name="app\_name">LupoliHelloWorldSplash</string>  <string name="hello\_world">Hello world!</string>  <string name="menu\_settings">Settings</string>  <string name="title\_activity\_main">Hello World Splash</string>  <string name="title\_activity\_splash">Hello Lupoli</string>  </resources> |
|  |
|  |

Component Setup in the Java File

* component “basics” can be set up as a class variable
  + before any methods
  + but JUST BASICS
  + details of each component have to be created in OnCreate
  + errors otherwise
* again, we have the “basics” set up so the ENTIRE class can have access

|  |
| --- |
| Base Component Setup within the Java file |
| package com.example.helloworld;  import android.os.Bundle;  ...  public class Main extends Activity {  // setup Base components here so everyone in the class has access  Button btnFood;    @Override  public void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_main);    btnFood = (Button) findViewById(R.id.btnFood); |

Other components (Have them do as a lab??)

* Edit Text
* TextView

Gravity and Weight

* gravity
  + cell alignment (like in Word)
    - top, bottom, left, right, center\_horizontal
    - Elcipse will show you all options
* weightSum/weight
  + weightSum
    - think of as an HTML table width of the entire page
  + weight
    - think of it as ONE colum (row) in the entire table
* works the same for columns or rows
* these do NOT go my pixels!!
  + they are just ratios

show horizontal buttons (or somtthing) pic then XML

<LinearLayout… android:weightSum =100

<TextView … android:layout\_weight = 50></Texview>

<TextView … android:layout\_weight = 50></Texview>

</ LinearLayout >

The ratio above COULD be simplified. Simplified to what? Answer: 2 (sum), 1, 1

RadioGroup

* setting radio buttons to be mutually exclusive
* is a layout

<RadioGroup andriod:id=”@+id/rgGroup1”>

<RadioButton>…

<RadioButton>…

<RadioButton>…

</RadioGroup>

code 2:52 in 1.13

When in XML, naming convention for GUI Components

tv = TextView (like a label, no interaction)

ev = EditView (like a textArea)

btn = Button

rb = RadioButton

rg = RadioGroup

etc…

Java code for iteraction (1.14)

TRY MY VERSION OF AN ACTIONLISTENER CLASS!!

* Listeners are INSIDE onCreate
* RadioBUttons/RadioGroups
  + in Java file implemenet OnCheckedChangeListener // for Radio Button
  + import adroid.widget.RadioGroup
  + in component setup area add
    - radioGroup1.setOnCheckedChnageListener(this);
    - this is referencing to the class that uses the “OnCheckedChnage”
  + error will pop over UNDER class name
    - hover error, select “Add unimplemented methods”

6:28 a lot of code for different types of GUI Components

For Buttons

button1.onCLickListener(new Viwe…..

9:04 in 1.14

For RadioBUttons

1.15 0:30 code

checkID= idOf the button

used a switch

switch(checkedId) 5:00 in 1.15

TextViews

1.15 10:24 How to chhage the font inside

FYI – Running one page at a time

* review the run configuration
* just pick the other “activity” to run
* I had to do this because my “main” activity was erring

ActionListener vs ActionListener

|  |
| --- |
|  |
|  |
|  |
| Book and Video Version |
| btnFood.setOnClickListener(**new** View.OnClickListener()  {  **public** **void** onClick(View v)  { startActivity(**new** Intent(Main.**this**, Food.**class**)); }    }); |

Sources:

LifeCycle:

<http://developer.android.com/images/training/basics/basic-lifecycle-create.png>