Android Workshop:

Android Details:

Android is framework that provides java programmers the ability to control different aspects of smart devices. This interaction happens through the **Android SDK** (Software Development Kit).

Model View Controller (MVC):

This is a programming paradigm that is used to help programmers organize large projects in common ways. **Models** represent the raw information or access points to the raw information. **Views** represent the visual representation of the models

Example:

If your model is a collection of menu items and prices (Strings and doubles) then the view may be a scrollable list that has the names on the left of each cell and the prices on the right. The View may offer other extra information such as dollar signs in front of each price.

Controllers connect the Views to the Models, they contain all the logic, every event that a view collects, such as a a touch, is sent to its respective controller, the controller then collect the proper information to complete the user's request and creates the new view.

Example:

A user taps on one of the menu items, the view collects the information and sends it to the controller, the controller now retrieves the Model for the detailed information for that menu item. In this case the detailed information contains the ingredients, the wine pairing and an image of the item (these may be several Models). The controller then packs the information up nicely for the new view that will show this information to the user in an organized fashion.

Main idea:

The view is expensive to manage, so minimize the amount of work it does. The Model is expensive to retrieve so minimize the work it does. The controller does the majority of the work and knows everything about the Models and very little about the Views. The Views know a lot about their Controllers.

MVC with Android:

Android attempts to use this by providing the user with resources and layouts (**res** folder) that are either raw data files, such as images, or **layouts** in XML. These are the Views, they dictate exactly what the Views will look like, how they are laid out and they follow the “Main idea” from above by doing absolutely no work (ie. No loops, no procedural programming). The controllers are called **Activities**, they interact with the views to create a user experience. The SDK triggers events in the Activities and

the activity collects the information from the models and builds the view and sets up the event triggers needed to accomplish the tasks. Models are left to the user's discretion, Usually they are implemented as classes, it is recommended to build as many classes that have no ties to the Activities as possible to provide modularity.

Threads

Thread:

A thread is a process that is run on the CPU, when a program is started it is a thread/process. A program can spawn and control other threads and processes (programs) to do tasks simultaneously.

Advantage:

Example: A thread can collect keyboard information, while another thread can allow the cursor to flicker. This can obviously be done in one thread if the programmer so wishes, but it is easier to let the operating system deal with running them at the same time and let the programmer code simple programs. If threads were unavailable, and the programmer wanted to keep his program as simple, the cursor would stop blinking while we wait for the user to input information into the keyboard (this may cause the user to think the computer crashed).

Disadvantage:

Many threads are very hard to control and can cause many issues.

User Interface (**UI**) Thread:

Because threads are so difficult to control accurately, Android (and most smart phones) have a UI thread that controls all user input and all drawing of objects. Android will not allow you to do these outside of the UI Thread. So **DO NOT LOOP ENDLESSLY** in the UI thread as it will cause the phone to stop responding.

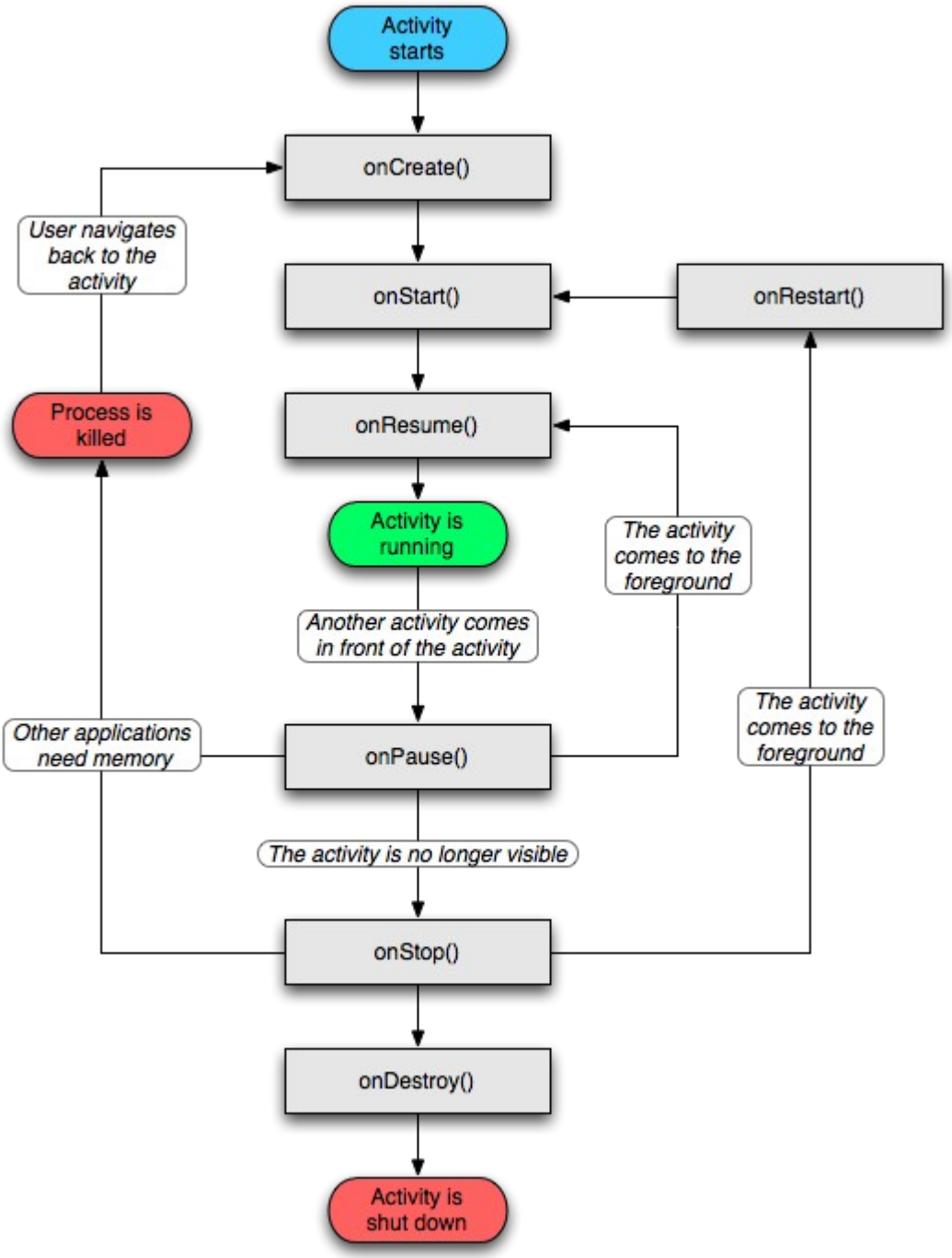
Layouts

Layouts are xml files that describe how everything will be laid out on the screen. The simplest layout tool is the **LinearLayout** this lets the items inside it be places horizontally or vertically, one after the other in the order in which they appear. An **ImageView** holds the images and a **TextView** holds the text. Note that everything must have **layout_height** and **layout_width** value in its opening tag.

Example:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:orientation="vertical"
      android:layout_width="fill_parent"
      android:layout_height="fill_parent">
      <ImageView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:src="@drawable/kitty"/>
      <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Kitten"/>
</LinearLayout>
```

Activity Life Cycle



Instructions:

Open eclipse

Select file/new

Choose Android Project

Enter a project name

Select a “Build Target” (android 2.2)

Fill in properties: “Package name” is something along the lines of com.android.myself, whatever you think is relevant.

Select create Activity and Give your Activity class a name, remember to use one word that starts with a Capitol letter

Click finish

You should see your new project in the navigator

If you do not see your navigator, click on window then show view then navigator.

Inside your project you will see:

bin – Binary files created during the compilation stage, these are not of relevance to you
gen – These files are generated so that your program can reference your resource files correctly from your code
src – This is where your code goes, notice that everything is found inside the package you named earlier.

res/layout – layouts that will describe your view
res/drawable-_dpi – images that will go into your views

In your package you will see the activity you created earlier, if you double click on it you will be able to edit it.

Sample 1:

YOUR ACTIVITY NAME.java

```
package YOUR PACKAGE NAME HERE;

import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.Toast;

public class YOUR ACTIVITY NAME extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        Button pressableButton = (Button) findViewById(R.id.pressable_button);
        OnClickListener onClickListener = getOnClickListener();
        pressableButton.setOnClickListener(onClickListener);
    }

      private OnClickListener getOnClickListener() {
            OnClickListener onClickListener = new OnClickListener() {

                  @Override
                  public void onClick(View v) {
                        Toast.makeText(getApplicationContext()
                        , "Hello Teachers!", Toast.LENGTH_SHORT).show();
                  }
            };
            return onClickListener;
      }
}
```

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="fill_parent"
      android:layout_height="fill_parent">
      <Button
            android:id="@+id/pressable_button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="PRESS ME"/>
</LinearLayout>
```

Sample 1:

YOUR ACTIVITY NAME.java

```
package YOUR PACKAGE NAME HERE;

import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.ImageView;

public class YOUR ACTIVITY NAME extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        Button pressableButton = (Button) findViewById(R.id.pressable_button);
        OnClickListener onClickListener = getOnClickListener();
        pressableButton.setOnClickListener(onClickListener);
    }

      private OnClickListener getOnClickListener() {
            OnClickListener onClickListener = new OnClickListener() {

                  @Override
                  public void onClick(View v) {
                        ImageView imageHolder =
                              (ImageView) findViewById(R.id.image_holder);
                        imageHolder.setImageResource(R.drawable.image);
                  }
            };
            return onClickListener;
      }
}
```

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="fill_parent"
      android:layout_height="fill_parent">
      <ImageView
            android:id="@+id/image_holder"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent">
      <Button
            android:id="@+id/pressable_button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="PRESS ME"/>
</LinearLayout>
```

YourActivity.java

```
package your.package;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.Toast;

public class YourActivity extends Activity {
	/** Called when the activity is first created. */
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.main);

		Button pizza = (Button) findViewById(R.id.pizza);
		pizza.setOnClickListener(getPizzaOnClickListener());

		Button pasta = (Button) findViewById(R.id.pasta);
		pasta.setOnClickListener(getPastaOnClickListener());

		Button steak = (Button) findViewById(R.id.steak);
		steak.setOnClickListener(getSteakOnClickListener());
	}

	private OnClickListener getPizzaOnClickListener() {
		OnClickListener onClickListener = new OnClickListener() {
			@Override
			public void onClick(View v) {
				Toast.makeText(getApplicationContext()
					, "We are out of dough", Toast.LENGTH_SHORT).show();
			}
		};
		return onClickListener;
	}

	private OnClickListener getPastaOnClickListener() {
		OnClickListener onClickListener = new OnClickListener() {
			@Override
			public void onClick(View v) {
				Intent settingsActivity = new
				 Intent(getApplicationContext(), PastaActivity.class);
				startActivity(settingsActivity);
			}
		};
		return onClickListener;
	}



	private OnClickListener getSteakOnClickListener() {
```

```
            OnClickListener onClickListener = new OnClickListener() {
                  @Override
                  public void onClick(View v) {
                        Toast.makeText(getApplicationContext()
                        , "You are out of dough", Toast.LENGTH_SHORT).show();
                  }
            };
            return onClickListener;
      }
}
```

PastaAvtivity.java

```
package your.package;

import android.app.Activity;
import android.os.Bundle;

public class PastaActivity extends Activity {
      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.pasta_activity);

      }
}
```

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="fill_parent"
      android:layout_height="fill_parent">

      <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
            android:orientation="horizontal"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:background="#FFFFFF"
            android:layout_gravity="center_horizontal">
            <TextView
                  android:layout_width="wrap_content"
                  android:layout_height="wrap_content"
                  android:text="Pizza:"/>
            <TextView
                  android:layout_width="wrap_content"
                  android:layout_height="wrap_content"
                  android:text="$12.99"/>
            <Button
                  android:id="@+id/pizza"
                  android:layout_width="wrap_content"
                  android:layout_height="wrap_content"
                  android:text="Buy!"/>
      </LinearLayout>
```

```
      <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
            android:orientation="horizontal"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:background="#00FF00"
            android:layout_gravity="center_horizontal">
            <TextView
                  android:layout_width="wrap_content"
                  android:layout_height="wrap_content"
                  android:text="Pasta"/>
            <TextView
                  android:layout_width="wrap_content"
                  android:layout_height="wrap_content"
                  android:text="$19.99"/>
            <Button
                  android:id="@+id/pasta"
                  android:layout_width="wrap_content"
                  android:layout_height="wrap_content"
                  android:text="Buy!"/>
      </LinearLayout>
      <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
            android:orientation="horizontal"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:background="#00FFFF"
            android:layout_gravity="center_horizontal">
            <TextView
                  android:layout_width="wrap_content"
                  android:layout_height="wrap_content"
                  android:text="Steak:"/>
            <TextView
                  android:layout_width="wrap_content"
                  android:layout_height="wrap_content"
                  android:text="$25.99"/>
            <Button
                  android:id="@+id/steak"
                  android:layout_width="wrap_content"
                  android:layout_height="wrap_content"
                  android:text="Buy!"/>
      </LinearLayout>
</LinearLayout>
```

pasta_activity.xml

```
<?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">
  <ImageView
        android:src="@drawable/pasta"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="your.package"
      android:versionCode="1"
      android:versionName="1.0">


    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name=".YourActivity"
                  android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".PastaActivity"
                  android:label="@string/app_name">
        </activity>

    </application>
</manifest>
```

This is how you downlaod an image

```
String url = “http://www.placekitten.com/200/200”;
HttpParams httpParameters = new BasicHttpParams();
HttpUriRequest httpRequest = new HttpGet(url);

DefaultHttpClient httpClient = new DefaultHttpClient(httpParameters);

try {
      HttpResponse response = httpClient.execute(httpRequest);

      HttpEntity entity = response.getEntity();
      InputStream is = null;
      Bitmap bitmap = null;
      try {
            BufferedHttpEntity bufHttpEntity = new BufferedHttpEntity(entity);
            is = bufHttpEntity.getContent();
            bitmap = BitmapFactory.decodeStream(is); // this is your image
      } finally {
            if (is != null) {
                  try {
                        is.close();
                  } catch (IOException e) {
                  }
            }
      }

} catch (ClientProtocolException e) {
} catch (IOException e) {
} finally {
      httpClient.getConnectionManager().closeExpiredConnections();
}
```

Eclipse Short cuts

ctrl + space tgriggers autocomplete. If you want to find out what methods objects have, type a dot in front of the object as though you were about to call a method and press ctrl + space and a list of options will show up. Also start typing a variable or class name, type ctrl + space and it will show you relevant options if there are more than one or it will simply complete the variable name for you.

Shift + ctrl + f this auto formats your code, makes it more legible

shift + ctrl + o imports all the classes needed to compile your class (usually gets rid of most of your squiqly lines. If this gives you options, choose very carefully

shift + ctrl + r will let you look for resources (files) that match what you typ in the search bar (hint: if you don't know what it starts with, but you know it contains specific letter in a specific order, type a star first then type what you know)

The play button runs your program.

Extra Android Information:

The AVD manager is what runs the android environment. It holds all the emulators and SDKs. There is an AVD manager installed in eclipse that you can play with to try different emulators and sdks.

The devices provide a log of current activities, to view that log, simply connect your device to the computer via usb and run the following command:

adb -d logcat

adb: is the android debugger

-d: means that you are accessing a device, (-e means emulator)

logcat: says that you are want to see the device log