In the previous lessons we’ve learned how to build an app and it appears that that creates a list and works…. that is until you close the app and you re-open it. What happens? You lose all progress.

Why does this happen? If you look the Android Life Cycle chart:



When an app is closed, the onDestroy() function is called which delete everything in your game. If you don’t save anything you’ll lose all progress.

And that’s our topic today: saving our game state.

To demonstrate saving game data and new materials from this point on, we’re going to create a new activity from our Todo List to build a simple Cookie Clicker game with upgrades.

We’ll be using something a class called SharedPreference to save our data. These are some of common methods to store your data:

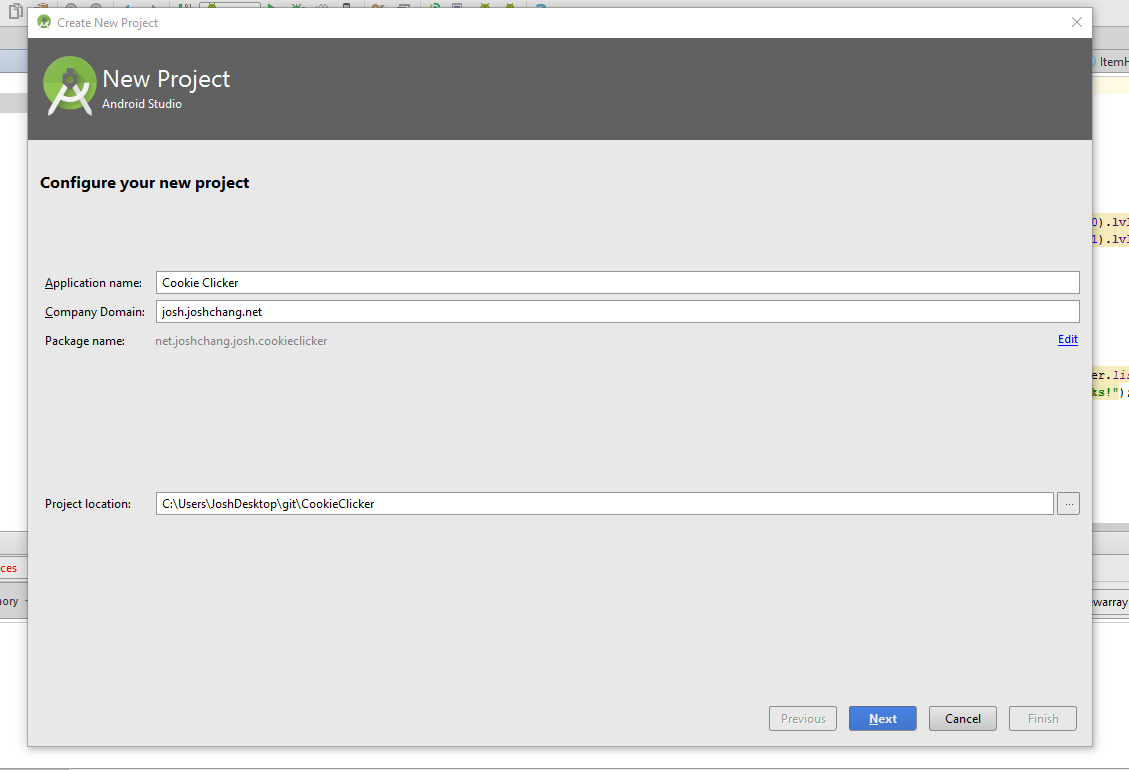
* SharedPreference – Save variables in a key-value mapping. Rooted Android devices can access these files so it’s not safe
* Files – Save data into a file, similar to SharedPreference, can be easily hacked.
* SQLite – Use a more traditional database supported by Android to store data. Similar to the previous 2, can easily be hacked
* Online Database – Save your data on your own database from your own server which solves the problem of users rooting their phone to hack your app.

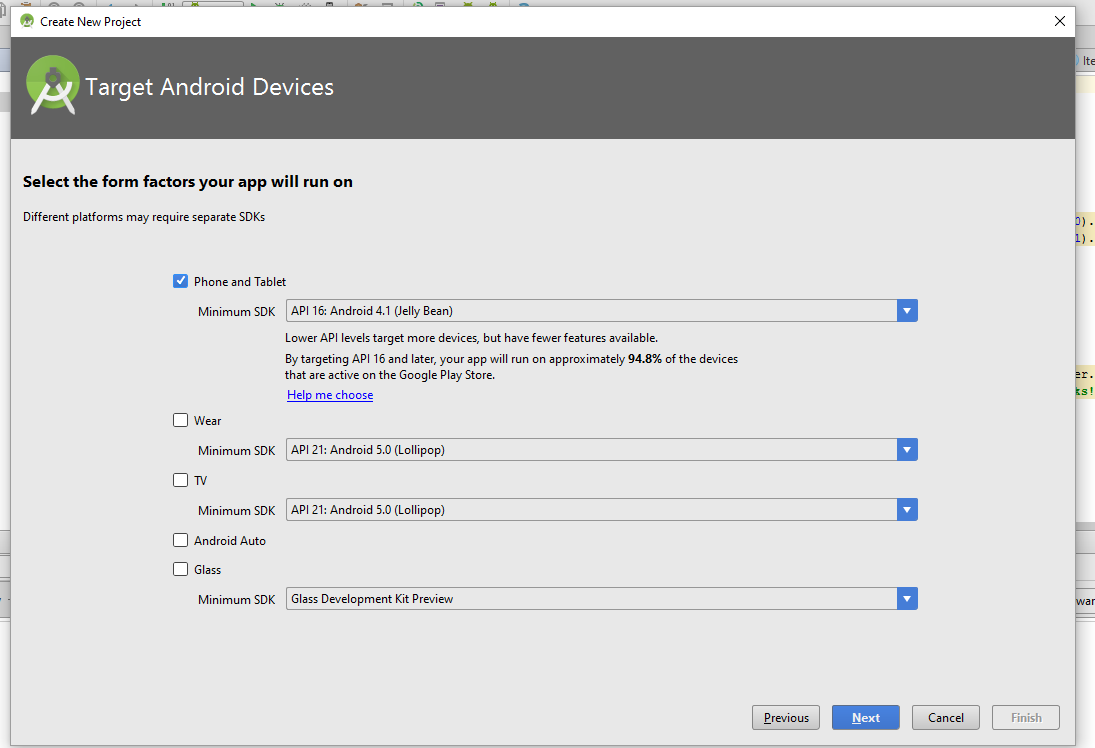
# Creating the Project

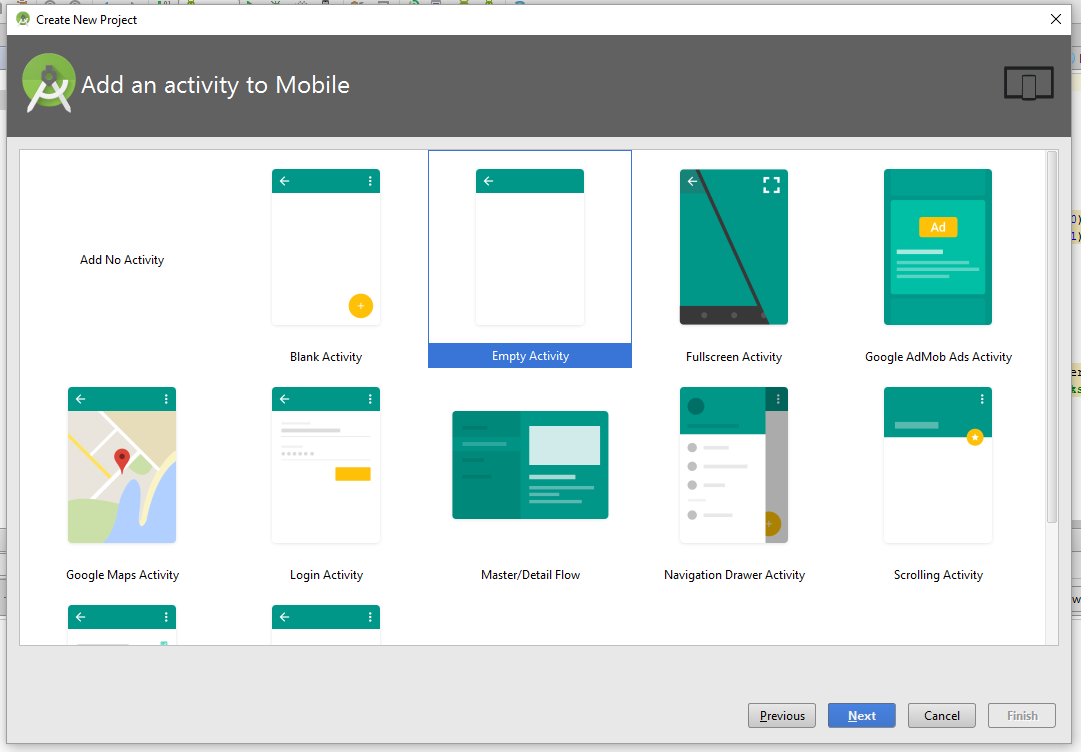
Let’s create a new project. Go to File->New->New Project…

Do exactly what we’ve done before:

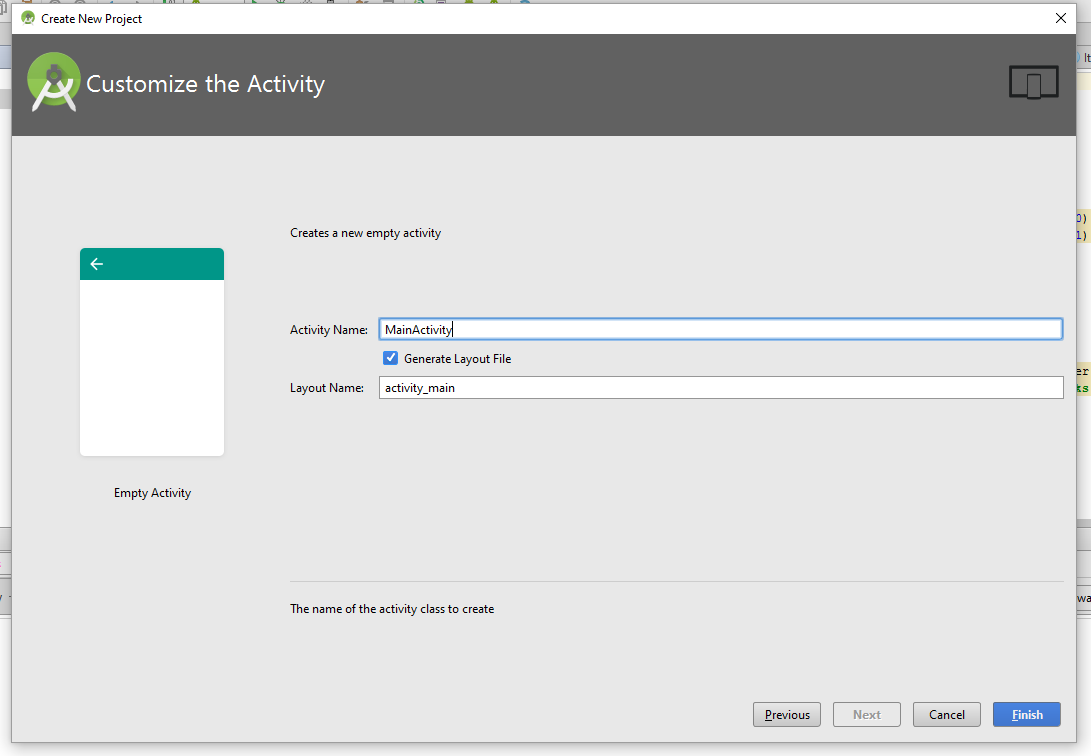
* Make the project Cookie Clicker
* Select API 16
* Select Empty Activity
* Keep the Activity name the same







Now that we have our project up and running, the next thing to do is to create our cookie clicker game.



Here’s all that we would need for now:

* A TextView to display our clicks
* A Button to click for points

Try and see if you can do it.

Done? Good! You should have something like this:

**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"  
 android:paddingBottom="@dimen/activity\_vertical\_margin"  
 android:paddingLeft="@dimen/activity\_horizontal\_margin"  
 android:paddingRight="@dimen/activity\_horizontal\_margin"  
 android:paddingTop="@dimen/activity\_vertical\_margin"  
 tools:context="net.joshchang.josh.cookieclicker.MainActivity"**>  
  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textAppearance="?android:attr/textAppearanceLarge"  
 android:text="Points: 0"  
 android:id="@+id/tvPoints"  
 android:layout\_alignParentTop="true"  
 android:layout\_centerHorizontal="true"  
 android:layout\_marginTop="101dp"** />  
  
 <**Button  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Click"  
 android:id="@+id/btnPoints"  
 android:onClick="btnClick"  
 android:layout\_centerVertical="true"  
 android:layout\_centerHorizontal="true"** />  
  
</**RelativeLayout**>

The code would give us this:



Now onto our MainActivity.java, we’re going to increment our points every time we click our button. To save our points we’re going to use Shared Preferences.

Here’s the code for MainActivity.java

**package** net.joshchang.josh.cookieclicker;  
  
**import** android.content.Context;  
**import** android.content.SharedPreferences;  
**import** android.os.Bundle;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.TextView;  
  
**public class** MainActivity **extends** AppCompatActivity {  
 **private** TextView **tvPoints**;  
 **private** Button **btn**;  
 **private int points**;  
 **private** SharedPreferences **pref**;  
  
 *// Hard-coded values to prevent confusion with variable types* **private final** String **PREF\_NAME** = **"pref"**;  
 **private final** String **POINTS** = **"totalPoints"**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
  
 *// Basic setup* **tvPoints** = (TextView) findViewById(R.id.***score***);  
 **btn** = (Button) findViewById(R.id.***button***);  
  
 *// Restoring data with pref* **pref** = getSharedPreferences(**PREF\_NAME**, Context.***MODE\_PRIVATE***);  
 *// Get the points that we store in the pref. Otherwise the default value* **points** = **pref**.getInt(**POINTS**, 0);  
 *// Sets the text* **tvPoints**.setText(**"Score: "** + **points**);  
  
 *// Increment our points and displays it* **btn**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **points**++;  
 **tvPoints**.setText(**"Score: "** + **points**);  
 }  
 });  
 }  
  
 *// Whenever the app is paused we would save our points* @Override  
 **protected void** onPause() {  
 **super**.onPause();  
  
  
 *// Gets a Editor from our shared preference* SharedPreferences.Editor edit = **pref**.edit();  
 *// Add our points with our editor to be saved* edit.putInt(**POINTS**, **points**);  
 *// Save our changes* edit.commit();  
 }  
}

So what’s happening? We should recognize some of the code involving the TextView and Button.

Here are the new pieces for Shared Preferences:

Here’s how we retrieve data:

* getSharedPreferences(**PREF\_NAME**, Context.***MODE\_PRIVATE***); - This is how we get our Shared Preference tied to our Pref Name string. Context.Mode\_Private ensures that only your application can access your SharedPreference data.
* **pref**.getInt(**POINTS**, 0); - We get the value that matches to our key that we store our int into. We also specific a default value that is returned if we have never stored anything before inside POINTS. We can also save other variable types besides int, including: String, Boolean, float, etc

Next we also want to save data to our SharedPreference. We do that with the SharedPreference.Editor. We use it in our onPause() method

* SharedPreferences.Editor edit = **pref**.edit(); - This creates our object to use to store our variables in.
* edit.putInt(**POINTS**, **points**); - Similar to getInt, we have putInt that stores our variable, into a specific string. We can also use other variable types such as: putString, putBoolean, etc.
* edit.commit(); - After we store the variables that we want, we have to commit() the changes. If we don’t none of the changes will be made.

And that’s all for using SharedPreferences!

Note, try and comment out the onPause method. When you do and run the app, all might look fine, but if you were to close the app and re-open the app, your score would be right back at 0!