Activity扮演了一个界面展示的角色，堪称四大组件之首，onCreate是Activity的执行入口，都不知道入口到底干了嘛，还学什么android,所以本文会从源码的角度对其进行分析。

熟悉源码的会发现，真正启动Activity的实现都在ActivityThread，前面的调用过程略过

ActivityThread的方法performLaunchActivity中调用了Instrumentation类中的方法callActivityOnCreate方法，继而调用了TargetActivity中的onCreate方法。

private Activity performLaunchActivity(ActivityClientRecord r, Intent customIntent) {  
......  
Activity activity = null;  
activity = mInstrumentation.newActivity( cl, component.getClassName(), r.intent);   
......  
if (r.isPersistable()) {  
 mInstrumentation.callActivityOnCreate(activity, r.state, r.persistentState);  
} else {  
 mInstrumentation.callActivityOnCreate(activity, r.state);  
}  
......  
}

#### **源码可知：**

1）通过反射的机制创建的Activity

2）这里的mInstrumentation是类Instrumentation

3）Instrumentation类中的方法callActivityOnCreate方法源码如下：

public void callActivityOnCreate(Activity activity, Bundle icicle) {  
 prePerformCreate(activity);  
 activity.performCreate(icicle);  
 postPerformCreate(activity);  
 }

#### **源码可知：**

1）activity.performCreate(icicle)，其中的方法是通过activity，这个activity，形如：Activity activity = 子Activity的对象

2）在Activity类中的方法performCreate(icicle)，源码如下：

final void performCreate(Bundle icicle) {  
 onCreate(icicle);  
 mActivityTransitionState.readState(icicle);  
 performCreateCommon();  
}

#### **源码可知：**

1）原来onCreate的生命周期方法是在这里回调的

2）在performCreate方法中调用的onCreate方法是MainActivity中的onCreate方法，那么到此MainActivity中的方法onCreate方法中的参数Bundle savedInstanceState也就知道来源了，此时，MainActivity中的方法也就被调用了。

再次看MainActivity中的方法onCreate:

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

super.onCreate(savedInstanceState)，其实这条语句放在子类中的onCreate方法中的任何位置都可，问题只是super.onCreate(savedInstanceState)必须要被执行，所以，最好也就是放在第一行，看起来比较明确，至于为什么，参考[onSaveInstanceState源码分析](http://www.jianshu.com/p/cbf9c3557d64) 至此onCreate源码分析完毕。