Android手机或平板都会存在横竖屏切换的功能，通常是由物理重力感应触发的，但是有时候也不尽然，通常在设置里面我们可以对手机的横竖屏切换进行关闭，操作界面如下

只需要点击下“屏幕旋转”按钮就可以关闭横竖屏切换了。

**一、禁止APP内横竖屏切换**

上述设置更改的是整个手机的横竖屏切换，当手机没有关闭横竖屏切换功能时，系统一旦触发横竖屏切换，缺省状态下，当前活动的App的界面就会进行横竖屏切换，由于横竖屏的界面尺寸等参数不同，很多软件在设计和开发中为了避免横竖屏切换时引发不必要的麻烦，通常需要让App禁止掉横竖屏的切换，这就需要通过在AndroidManifest.xml中设置activity中的android:screenOrientation属性值来实现。

该android:screenOrientation属性，他有以下几个参数：

"unspecified":默认值 由系统来判断显示方向.判定的策略是和设备相关的，所以不同的设备会有不同的显示方向.

"landscape":横屏显示（宽比高要长）

"portrait":竖屏显示(高比宽要长)

"user":用户当前首选的方向

"behind":和该Activity下面的那个Activity的方向一致(在Activity堆栈中的)

"sensor":有物理的感应器来决定。如果用户旋转设备这屏幕会横竖屏切换。

"nosensor":忽略物理感应器，这样就不会随着用户旋转设备而更改了（"unspecified"设置除外）。

比如下列设置

android:screenOrientation="portrait"

则无论手机如何变动，拥有这个属性的activity都将是竖屏显示。

android:screenOrientation="landscape"，为横屏显示。

上述修改也可以在Java代码中通过类似如下代码来设置

setRequestedOrientation(ActivityInfo.SCREEN\_ORIENTATION\_LANDSCAPE)

**二、APP的横竖屏切换可以手动触发吗**

由上面描述可知，当android:screenOrientation为默认值"unspecified"或"sensor"等时，就会有系统根据设备的旋转情况来触发横竖屏的切换，那么有没有方法我们手动在程序中触发横竖屏的变换呢，显然上面为我们提供的setRequestedOrientation就是系统提供的一个入口，下面我们给出一个按键的方式来触发的案列：

public class MainActivity extends Activity implements OnClickListener {

     private Button mBtnLandscape;

     private Button mBtnPortrait;

     @Override

     protected void onCreate(Bundle savedInstanceState) {

         super.onCreate(savedInstanceState);

         setContentView(R.layout.activity\_main);

         mBtnLandscape = (Button) findViewById(R.id.but\_landscape);

         mBtnPortrait = (Button) findViewById(R.id.but\_portrait);

         mBtnLandscape.setOnClickListener(this);

         mBtnPortrait.setOnClickListener(this);

     }

     @Override

     public void onClick(View v) {

                 // TODO Auto-generated method stub

                 if(v == mBtnLandscape){

                             setRequestedOrientation(ActivityInfo.SCREEN\_ORIENTATION\_LANDSCAPE);

                 }else{

                             setRequestedOrientation(ActivityInfo.SCREEN\_ORIENTATION\_PORTRAIT);

                 }

     }

     @Override

     public void onConfigurationChanged(Configuration newConfig) {

         super.onConfigurationChanged(newConfig);

         String message=newConfig.orientation==Configuration.ORIENTATION\_LANDSCAPE ? "屏幕设置为：横屏" : "屏幕设置为：竖屏";

         Toast.makeText(this, message, Toast.LENGTH\_LONG).show();

     }

}

需要注意的是，手动调用时，无视AndroidManifest中关于screenOrientation的设置；另外上述代码中的onConfigurationChanged要被调用到也是需要条件的，在这里，只给代码，不做讨论，后面再给出一个相关的补充说明。

**三、重启Activity的横竖屏切换**

在上面的案列中，缺省状态下，Activity每次横竖屏切换（包括用setRequestedOrientation调用）都会重新调用一轮onPause-> onStop-> onDestory-> onCreate->onStart->onResume操作，从而销毁原来的Activity对象，创建新的Activity对象，这是因为通常情况下软件在横竖屏之间切换，界面的高宽会发生转换，从而可能会要求不同的布局。具体的布局切换可以通过如下两种方法来实现：

1）在res目录下建立layout-land和layout-port目录,相应的layout文件名不变，比如main.xml。layout-land是横屏的layout,layout-port是竖屏的layout，其他的不用管，横竖屏切换时程序自己会调用Activity的onCreate方法，从而根据当前横竖屏情况自动加载响应的布局。

2）假如布局资源是不一样又不按照如上设置，则需要通过java代码来判断当前是横屏还是竖屏然后来加载相应的xml布局文件（比如mainP为竖屏mainL为横屏）。因为当屏幕变为横屏的时候,系统会重新呼叫当前Activity的onCreate方法,你可以把以下方法放在你的onCreate中来检查当前的方向,然后可以让你的setContentView来载入不同的layout xml。

@Override

protected void onCreate(Bundle icicle) {

 super.onCreate(icicle);

 int mCurrentOrientation = getResources().getConfiguration().orientation;

 if ( mCurrentOrientation == Configuration.ORIENTATION\_PORTRAIT ) {

     // If current screen is portrait

     Log.i("info", "portrait"); // 竖屏

     setContentView(R.layout.mainP);

 } else if ( mCurrentOrientation == Configuration.ORIENTATION\_LANDSCAPE ) {

     //If current screen is landscape

     Log.i("info", "landscape"); // 横屏

     setContentView(R.layout.mainL);

 }

 init();//初始化，赋值等操作

 findViews();//获得控件

 setListensers();//设置控件的各种监听方法

}

上面只是对布局切换做了描述，实际上由于重启Activity在未加处理的情况下必然导致数据的丢失和重新获取，这样用户体验会非常差。为此就要在切换前对数据进行保存，切换重启后对数据进行恢复，具体操作的步骤如下：

重写Activity.onRetainNonConfigurationInstance()，用户横竖屏切换前保存数据

@Override

public Object onRetainNonConfigurationInstance() {

    final MyDataObject data = collectMyLoadedData();

    return data;

}

在onCreate()函数中调用getLastNonConfigurationInstance()，获取onRetainNonConfigurationInstance()保存的数据

@Override

public void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    setContentView(R.layout.main);

    final MyDataObject data = (MyDataObject) getLastNonConfigurationInstance();

    if (data == null) {

        data = loadMyData();

    }

    ...

}

**四、非重启Activity的横竖屏切换**

虽然重启Activity为我们提供了保存数据和读取数据的方式，但是如此一来程序会显得有些繁琐，所以有时候程序员往往就不想让Activity重启，Android也为我们提供了解决方案，就是通过onConfigurationChanged拦截横竖屏变换，从而进行必要的重新布局和切换操作。操作步骤如下：

首先，manifest中为相应的Activity设置android:configChanges属性，从而让Activity不延续上述的重建流程，具体如下：

Andorid 3.2以前的SDK可以使用如下配置

android:configChanges="orientation|keyboardHidden"

而Adnroid 3.2以后的SDK必须添加一个screenSize属性，具体如下

android:configChanges="keyboardHidden|orientation|screenSize"

或者

android:configChanges="orientation|screenSize"

关于configChanges的详细描述，后面有个简单补充章节，这里不做过多展开。

其次，在Activity或View的onConfigurationChanged(Configuration newConfig)函数中获取当前横竖屏参数。至于其调用顺序跟touch事件的传递顺序相似，不过他没有消费事件的概念，会顺次调用到每一个onConfigurationChanged函数。下面是重写Activity的例子：

//布局分别在layout-land和layout-port目录中的同名main.xml时

@Override

public void onConfigurationChanged (Configuration newConfig){

    super.onConfigurationChanged(newConfig);

    setContentView(R.layout.main);

    //注意，这里删除了init()，否则又初始化了，状态就丢失

    findViews();

    setListensers();

}

//布局为不按照layout-land和layout-port目录，而自定义名字时

@Override

public void onConfigurationChanged (Configuration newConfig){

    super.onConfigurationChanged(newConfig);

    int mCurrentOrientation = getResources().getConfiguration().orientation;

    if ( mCurrentOrientation == Configuration.ORIENTATION\_PORTRAIT ) {

        // If current screen is portrait

        setContentView(R.layout.mainP);

        //注意，这里删除了init()，否则又初始化了，状态就丢失

        findViews();

        setListensers();

    } else if ( mCurrentOrientation == Configuration.ORIENTATION\_LANDSCAPE ) {

        //If current screen is landscape

        setContentView(R.layout.mainL);

        //注意，这里删除了init()，否则又初始化了，状态就丢失

        findViews();

        setListensers();

    }

}

当然有时候连布局都不用更改的话，就可以直接对原有控件进行调用操作了，比如：

public class MainActivity extends Activity {

    private TextView textView;

    @Override

    public void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.main);

        Log.i("--Main--", "onCreate");

        textView=(TextView)findViewById(R.id.tv\_id);

    }

    @Override

    public void onConfigurationChanged(Configuration newConfig) {

        super.onConfigurationChanged(newConfig);

        Log.i("--Main--", "onConfigurationChanged");

        if(newConfig.orientation==Configuration.ORIENTATION\_LANDSCAPE){

            textView.setText("当前屏幕为横屏");

        }else{

            textView.setText("当前屏幕为竖屏");

        }

    }

}

需要注意的是，onConfigurationChanged函数中只能获得横竖屏切换后的参数，在该函数中获取不到新的Layout和控件的尺寸位置信息，如果要处理尺寸和位置信息，必须通过消息异步或者延时调用，下面是一个App在横竖屏切换时需要重新设置popupWindow位置的代码：

@Override

protected void onConfigurationChanged(Configuration newConfig) {

    super.onConfigurationChanged(newConfig);

    //View中不用创建Handler，可直接调用post操作

    //new Handler().postDelayed(new Runnable() {

    //    @Override

    //    public void run() {

    //        updatePopup();

    //    }

    //}, 500);

    postDelayed(new Runnable() {

        @Override

        public void run() {

            updatePopup();      //

        }

    }, 500);//如果不在post中，而是直接调用，那么弹出位置就会有问题

}

虽然上面没有看到对布局的显式调用进行重新布局，照理控件的对象没有被销毁，但是控件在横竖屏切换时应该是需要进行重新layout和measure，然后再进行重绘的，否则不会引发弹出框位置的变化，至于如何调用重新layout、measure和Draw操作，在这里就不多展开了。

**五、对于AndroidManifest.xml设置的补充**

经过上面代码演示，我们可以看到具体实现涉及到了Manifest工程配置里面具体Activity的screenOrientation和configChanges两个参数，这两个参数screenOrientation的优先级是高于configChanges，即假如screenOrientation设置为固定横竖屏时，那么configChanges参数无论怎么设都没有办法引发横竖屏切换，除非在代码中手动调用setRequestedOrientation函数进行修改。

screenOrientation属性在前面已经讲过了，而关于configChanges属性设置有如下选项：

|  |  |
| --- | --- |
| 值 | 描述 |
| mcc | IMSI移动台的国家代码（MCC）发生变化——一个SIM被探测到并且更新MCC |
| mnc | IMSI移动台的网络代码（MNC）发生变化——一个SIM被探测到并且更新MNC |
| locale | 区域发生变化——用户选择了一个文本需要显示的新语言 |
| touchscreen | 触摸屏发生变化。（这个通常不会发生。） |
| keyboard | 键盘类型发生变化——例如：用户插入了外接键盘。 |
| keyboardHidden | 键盘的可访问性发生变化——例如：用户发现了硬件键盘。 |
| navigation | 导航类型（轨迹球或dpad）发生变化。（通常不会发生。） |
| screenLayout | 屏幕布局发生变化——这个会导致显示不同的Activity。 |
| fontScale | 字体缩放因子发生变化——用户选择了新的字体大小。 |
| uiMode | 当UI模式发生改变的时候——当用户放置设备到桌子或/汽车或夜间模式改变的时候可以引起UI模式变化。阅读UiModeManager。在API级别8时引入。 |
| orientation | 屏幕方向发生变化——用户旋转了屏幕。注意：如果应用程序的目标API级别是13或更高（通过属性minSdkVersion和属性targetSdkVersion声明），你也需要声明配置项screenSize，因为这将在设备选择肖像和屏幕方向时发生改变。 |
| screenSize | 当前可用屏幕大小发生变化。这代表一个当前可用大小的变化，和当前的比率相关，因此当用户选择不同的画面和图像，会发生变化。然而，如果你的程序目标API级别是12或更低，你的Activity总是会自己处理这个配置变化（这个变化不会引起Activity的重启，甚至在Android 3.2或更新的设备上）。在API级别13里加入的。 |
| smallestScreenSize | 物理屏幕大小的变化。不管方向的变化，仅仅在实际物理屏幕打包变化的时候，如：外接显示器。这个配置项的变化引起在smallestWidth configuration里的变化。然而，如果你的程序目标API级别是12或更低，你的Activity将自己处理这个变化（这个变化不会引起Activity的重启，甚至在Android 3.2或更新的设备上）在API级别13里加入的。 |
| layoutDirection | 布局方向变化。例如书写方式从左向右（LTR）转换为从右向左（RTL） |

从上述这个表我们可以看到除了横竖屏，包括语言、网络、键盘和外设等变化都可以被onConfigurationChanged函数监控到，具体的内容和释义还是查看官方英文文档吧，详见如下链接

<http://developer.android.com/guide/topics/manifest/activity-element.html>

中文翻译可以查阅 <http://wiki.eoe.cn/page/Activity.html>

结合网上的整理，小结跟这几配置相关的情景：

1、不设置Activity的android:configChanges时，切屏会重新调用各个生命周期，切横屏时会执行一次，切竖屏时会执行两次（我在三星4.0设备上发现切横屏和竖屏都是执行一次，而并非这里说的有执行两次的情况，不知道是否以前版本手机会这样？）；

2、设置Activity的android:configChanges="orientation"时，切屏还是会重新调用各个生命周期，切横、竖屏时只会执行一次；

3、设置Activity的android:configChanges="orientation|keyboardHidden"时，切屏不会重新调用各个生命周期，只会执行onConfigurationChanged方法。

注：上述描述是在Android3.2以前，如果缺少了keyboardHidden选项，不能防止Activity的销毁重启，也就不能执行onConfigurationChanged方法了。在3.2之后，必须加上screenSize属性才可以屏蔽调用Activity的生命周期（我在一些设备上亲测可以不需要keyboardHidden，只要screenSize就可以了，但是保险起见还是继续保留keyboardHidden吧）。

**六、对于setRequestedOrientation函数的补充说明**

在上述（二）对于手动触发横竖屏切换的时候，我们用到了setRequestedOrientation，那时只是简单做了下演示，后来发现还是需要做下补充说明的：

首先在非重启Activity模式下

手动调用setRequestedOrientation之后，假如会引发横竖屏切换（即请求的横竖屏要求与当前的横竖屏情况不一致，就会引发切换），那么会立即调用onConfigurationChanged函数；假如不会引发横竖屏切换（请求前后一致），那么也就不会调用到onConfigurationChanged函数。

这个手动调用setRequestedOrientation的地方也可以在Activity中的任何地方，即也可以在onConfigurationChanged中调用，但是一旦指定为横屏或竖屏完成这个变换之后，后面不论屏幕如何进行怎么翻转变化，都不会再触发横竖屏切换了，也即等同于在manifest中设置了android:screenOrientation属性为横屏或竖屏。如果要恢复为响应横竖屏随物理传感器设备变换，那么就需要手动调用类似如下代码进行恢复：

setRequestedOrientation(ActivityInfo.SCREEN\_ORIENTATION\_SENSOR);

其次在重启Activity模式下

手动调用setRequestedOrientation发出横竖屏设定请求之后，假如需要进行横竖屏切换（即请求前后横竖屏状态不一致），则会对Activity进行销毁并重启；假如不需要需要进行横竖屏切换，则Activity维持现状不变；

手动调用setRequestedOrientation一次，完成变换之后，也跟上面非重启一样，相当于在manifest中设置了android:screenOrientation属性为横屏或竖屏。要想恢复也需要重新调用类似上面非重启的调用。

在这样一个原理下，就有了对如下一种需求的解决方案：

让App启动的时候是横屏的话就横屏表示，纵屏的话就纵屏表示，然后手机切换横竖屏就不能用了该怎么解决呢？

网上给出了一个例子代码，这里就不做摘抄了，有兴趣可以试一下，然后对比一下人家的实现方式，具体见如下链接

<http://blog.csdn.net/yimo29/article/details/6030445>

另外再给出几个我做整理时参考的帖子，觉得对我帮助很大，分别如下

Android横屏竖屏切换的问题（一个总结帖，还是不错的）

<http://blog.sina.com.cn/s/blog_77c632410101790w.html>

解决Android手机屏幕横竖屏切换（一个真实测试过的小结）

<http://www.cnblogs.com/zhangkai281/archive/2011/07/06/2099277.html>

Android 处理横竖屏切换事件

<http://ipjmc.iteye.com/blog/1265991>