align  v. 成一条线;排整齐;

**安卓四大组件：**

Activity(活动)，

其四个状态:激活，暂停(可见状态但失去焦点)，停止，…

方法:

oncreate()

activity生命周期的开始函数。

注:

新创建的activity要在mainfest中声明

Service(服务)，

BroadcastReceive(广播接收器)，

ContentProvide(内容提供者)

注：组件都得注册才可使用，组件有静态注册方式和动态注册方式。

**service:**

介绍:第9-10？

服务是一个可在后台执行长时间操作而不提供用户界面的组件。

创建服务:

父类:Service

service类生命周期:

onBind(Intent )：

onUnbind()：

onCreate()：

onDestroy()：

onStartCommand()：

注：

MyBinder:

介绍:

service类的内部类，该类用来允许activity能调用service的那些方法，

在服务的onBind()周期返回该类的对象。

注册服务:

在mainfest文件中以<service android:name="">标签声明service。

启动service方式:

介绍：

服务可由其他组件启动。

注:组件可以绑定到服务，以与之交互，

甚至是进行进程间通信（IPC）。

startService()

注:该方式启动的service，当前activity与service无任何联系，activity无法调用该服务的方法。?

调用stopService()停止服务

bindService()

介绍:

把activity与服务绑定，然后来启动service。一般要调用service中方法时用该函数启动服务。

参数：

Intent 意图

ServiceConnection

这是个接口，即用匿名内部类实例化该接口。当绑定服务后，该接口可根据绑定状态执行相应操作，如可在服务绑定成功后获取服务暴露出来的方法。

函数：

onServiceDisconnected()

onServiceConnected():

绑定成功（即服务连接成功），返回一个IBinder对象，

通过IBinder对象获取服务暴漏给调用者的方法。

这样我们可以去使用service里的方法。

注：

为解决解绑服务后，服务还在问题，

末尾一般把boolean类型变量设为true。

int flags

服务启动流程：

调用bindService()启动服务后，

1.会触发服务类的onBind()，

onBind()可返回IBinder，

注：

IBinder类要继承Binder,

实现getLocalService方法，

1.该方法决定了要返回服务的那些方法，

即要把服务的那些方法暴漏给服务调用者。

2.该方法返回LocalService类型参数。

注：

把整个服务暴漏给activity

return LocalService.this;

2.接着触发bindService方法的ServiceConnection参数 ，

该参数用来接收服务的状态。

注:通过Binder来实现activity能操作service中方法

注:

以上均需要intent参数。

解除服务绑定：

unbindService();

解绑服务后，服务还在：

即解绑服务后，组件还能调用到服务方法。

例：解绑后，点击控件还能调用服务方法。

解决方案：

设置个boolean变量，

组件启动成功变量设为true，

其余时候设为false，

当控件点击时判断该变量值，

是true点击事件才可使用。

可解决解绑服务后，还可调用服务的问题。

两种方法混合使用:?

远程服务:

介绍：

1.可实现进程间通信。

2.android进程之间彼此是隔离的，进程之间传递数据和对象，需要使用Android支持的

进程间通信（Inter-Process Communication，IPC）机制。

3.Android采用Intent和远程服务的方式实现IPC。

4.Android中，进程之间不能直接访问相互的内存控件，

因此为了使数据能够在不同进程间传递，数据必须转换成能够穿越进程边界的系统级原语，

同时，在数据完成进程边界穿越后，还需要转换回原有的格式。

使用AIDL创建绑定服务:

介绍：

AIDL（Android Interface Definition Language）

是Android系统自定义的接口描述语言，

可以简化进程间数据格式转换和数据交换的代码。

1.使用AIDL语言定义远程服务的接口（.aidl文件）。

interface IRemoteService{

//声明方法

}

2.在具体的远程服务中实现接口中的方法，将接口暴露给客户端。

private IRemoteService.Stub mBinder = new IRemoteService.Stub() {

    //实现接口中的方法

};

3.绑定和使用远程服务

在另一个app中加入远程服务的接口（.aidl文件），并在需要服务的地方绑定远程服务。

对ServiceConnection的onServiceConnected方法的处理：

在该方法中通过如下代码

来获取远程服务的onbind方法返回的IBinder对象的代理：

IRemoteService.Stub.asInterface(service)

注：样例为实验11

**BroadCastReceiver:**

介绍:

Android中，有一些操作完成以后，会发送广播，

广播接收器是一种用于响应系统范围广播通知的组件。

创建:

继承自BroadCastReceiver

方法:

onReceive()接收到广播时触发该函数。

注册:

静态:

实现：

<!-- 注册广播接收器-->

<receiver android:name="cn.hecaigui.twelvebroadcastreceiver.BroadCastReceiver">

<!-- 设置过滤器，案例中是接收手机重启广播。注：要获取允许接收重启信息的权限-->

<intent-filter>

<action android:name="android.intent.action.BOOT\_COMPLETED"/>

</intent-filter>

</receiver>

优点：即使应用程序关闭了，依然可以收到广播。

缺点：对于比较频繁的广播消息，不能使用这种方式。

动态:

实现：

代码中定义并设置好一个IntentFilter对象，

然后在需要注册的地方调用registerReceiver()方法。取消时调用unregisterReceiver()。

优点：可以接收比较频繁的广播消息。

缺点：当注册它的对象被销毁时，它也会自动取消注册。

发送广播:

方法:

sendBroadcast(intent);//该方法在activity类中就有

**Intent:**

介绍：

轻量级的消息传递机制，用来启动其他组件可携带信息。

即使用Intent启动Activity和Service，同时Intent可以传递数据。

启动activity的两种方式:

显示意图:

明确指出要启动的组件。

简易使用:

实例化时Intent时传递两个参数。

例：new Intent(MainActivity.this,HelloActivity.class);

隐式意图:

介绍:

给出条件，由系统去查找满足条件的组件，效率低。一般用来调用Android系统自带的组件。

注:这种方式启动的activity，activit要在配置文件中指明action和catalogue

简直使用:

...

方法:

startActivityForResult():

介绍:

启动一个activity后可返回一个值。

该方法为activity拥有。

setResult()：

介绍：

用来设置返回给上个activity的消息。

该方法为activity拥有。

注：调用finish()可结束当前activity，可返回上个activity。？

onActivityResult():

介绍:

用来处理从其他activity返回的信息。

该方法为activity拥有。

该方法参数：

requestCode 用来区分result来自那个activity

注：这个requestCode应该指的是

最初的activity的requestCode 。

即1跳2，2再跳1。则requestCode 为1的。

resultCode 用来判断返回值是否有意义

data 用来存放返回的intent。

getIntent()

用来获取从另一个组件传递过来的intent，可读取intent携带的消息。

该方法为activity拥有。

**Layout(Android布局)：**

小知识:

@+id    新建一个ID    @id    一个ID

dp安卓特有的尺度，推荐使用因为它会根据不同尺寸设备自动调节。

RelativeLayoutv(相对布局):

听说不咋好用!

LinearLayout(线性布局)的标签:

android:orientation=

"horizontal"表示水平方式排列

"vertical"表示布局中控件以竖直方式排列

layout\_weight(权重)：

用来使控件平分屏幕，但是layout\_width要设为0dp,仅在LinearLayout起作用。

layout\_gravity:

控件相对布局的对齐方式

gravity:

控件中文本对齐方式

FrameLayput(帧布局)

AbsoluteLayout(绝对布局)

GridLayout(网格布局)

TableLayout(表格布局)

控件属性:

<refocus/>:自动聚焦

spinner：

函数：

.setOnItemSelectedListener();

//当框中选项改变时，会触发该监听器

**点击事件:**

介绍：

事件处理机制是应用程序组件级别的，

比如一个按钮的OnClickListener事件，只能够在一个应用程序中处理。

1.

view.setOnclickListener(参数为实现了onClickListener接口的匿名内部类)

2.Activity实现OnClickListener接口:

介绍：

该方式适用于布局中点击事件较多。

接口函数：

onClick(View v) {

//当点击设置监听器并绑定了该类的控件时，会调用该函数，

//通过v.getId()可获得点击控件的id

//

}

实现步骤:

1.控件绑定监听器：

例：button.setOnClickListener(this);

2.activity实现OnClickListener接口。

**适配器:**

(我认为是一个容器，即用来指定内容如何显示，适配器要加载到控件中才能生效)

ArrayAdapt:

只能显示字符串

SimpleAdapt:

显示图片和文字但是位置被固定死。

BaseAdapt:

用来制定自己的适配器

列表布局等用adapt显示内容步骤:

1.准备数据

2.新建一个适配器

3.设置适配器(即可把适配器与spinner或listView或其他视图绑定)

BaseAdapt使用:

// 返回数据集中数据总个数

@Override

public int getCount() {

return list.size();

}

// 获取数据集中与指定索引对应的数据项

@Override

public Object getItem(int position) {

return list.get(position);

}

// 获取指定行(指定索引)对应的ID

@Override

public long getItemId(int position) {

return position;

}

// 获取指定索引的Item视图

@Override

public View getView(int position, View convertView, ViewGroup parent) {

View item = mInflater.inflate(R.layout.list\_item, null);

获取item的控件;

对获取的控件设置值;

(用这种方式设值list.get(position).get\*\*\*())

return item;

}

注：

一个listview（adapt容器）中有很多item（数量可以从0-N），

而adapt中只是设置单条item的值，只是listview会把当前的item的position传递给adapt，而adapt负责返回指定位置的item的具体视图。错???

adapt的getCount负责设置总数据数，而adapt的其余3个函数负责设置指定位置的item的具体视图。

函数介绍:

.notifyDataSetChange();

//提示适配器更新数据

**菜单：**

菜单布局设置:

创建一个menu文件夹，在该文件夹中创建.xml。在.xml进行菜单布局设置。

onCreateOptionsMenu(){}：

OptionsMenu是属于一个Activity

菜单布局的添加：

getMenuInflater().inflate(菜单xml文件, menu);

onOptionsItemSelected(){}

OptionsMenu的item(条目)选中事件处理

onCreateContextMenu(){}

ContextMenu（上下文菜单）是一个针对View的菜单，它绑定在某一个View上.

菜单布局的添加：

getMenuInflater().inflate(R.menu.\*,menu);

onContextItemSelected(){}

上下文菜单条目选中事件处理

为控件注册上下文菜单：

registerForContextMenu(textView);

注：

不注册，上下文菜单无法生效。

为listview注册上下文菜单：

为listview注册上下文菜单时，

上下文菜单是绑定到整个listview中，

还是listview会把上下文菜单绑定到每个item中。？

答：

可能是绑定到listview中。

绑定上下文菜单后，在listview中点击一个item时会自动调用adapt的那些方法？

getItemId（）一定被调用了。

答：

只要点击item就会调用adapt的getItemId（）等方法，不需要绑定上下文菜单。

注：

listview注册上下文菜单后点击item，应该是先调用adapt的方法，然后在调用listview上下文菜单。

**对话框:**

AlertDialog.Builder();对话框设置

.setView();给对话框添加视图

.setPositiveButton(，"这里写点击确认后执行操作");

//给对话框添加确认按钮

.creat().show();显示对话框

View.flate();布局转换为视图

**视图:**

介绍：

创建新的视图，可用来加载到布局（Layout）中。

注：Android自带视图有TextViw等

画板视图:

创建画图的视图，父类view。

实现画板的关键函数：

onDraw(Canvas canvas)：

介绍：Implement this to do your drawing.实现该函数就可进行绘图。

该方法具体内容:

canvas.drawPath(path, paint);

介绍：

canvas - the canvas on which the background will be drawn

一个已经画好背景的画布，最后该画布会被显示出来。

drawPath()在该画布上画图

Paint画笔:

画出的线或图的样式。

Path路径:

moveTo(x, y)图从哪开始画

lineTo(x, y)图画到那

reset()即清空画布上图

注：

如过未重新调用moveTo()，下面图的起点将从上个图停下来的点

onTouchEvent()：

介绍：触摸事件,当触碰屏幕时会调动该函数

返回值:

true 表明后续的移动，抬起事件处理都由该函数处理。

false 后续事件传递下去!

即返回true该函数只能处理触摸时那瞬间的事件。

invalidate()：

通知系统重调用onDraw(),即重新加载画布。

在布局中加载新建的视图：

1.可在design中找到自定义的视图托进去

2. 在布局代码中添加自定义视图，即添加包名.类名

<cn.hecaigui.eightthmydrawview.MyDrawView

....

/>

**注：**

界面事件:

点击:

长按:

触摸:

与关闭程序有关:?

基本用个activitymangager

kill掉所有的activity就行了把

加到一个集合 在杀死

finish()?

数据库:

父类:SQLiteOpenHelper

声明周期:

构造函数中创建数据库。

onCreate…才会调用

**错误:**

启动eclipseForAndroid出现adt出错:

解决方案:

命令行中输入 eclipse.exe -clean