1. 控件
2. Menu控件

当用户点击菜单按钮的时候调用

复写onCreateOptionsMenu方法

例子：

Public Boolean onCreateOptionsMenu(Menu menu)

{

menu.add(0,1,1,R.string.exit);//（组、编号【ItemId】、排序、名称）

menu.add(0,2,2,R.string.help);;

}

当用户点击某一个菜单项时调用

复写onOptonsItemSelected方法

例子：

Public void onOptionItemSelected(MenuItem item)

{

If(item.getItemId == 1)

{

Finish();//结束应用程序

}

}

1. RadioGroup和RadioButton
   1. 布局

<RadioGroup

android:id=”@+id/genderGroup”

android:layout\_width=”wrap\_content”

android:layout\_height=”wrap\_content”

android:orientation=”vertical”>

<RadioButton></RadioButton>

<RadioButton></RadioButton>

</RadioGroup>

* 1. 代码

Private RadioGroup genderGroup;

Private RadioButton femaleButton;

Private RadioButton maleButton;

genderGroup.setOnCheckedChangeListener(new RadioGroup.OnCheckedChangeListener(){

@Override

Public void onCheckedChanged(RadioGroup group, int checkId)

{

If(femaleButton.getId == checkId)

{

System.out.Println(“female”);

}else if(maleButton.getId == checkId){}

}

});

1. CheckBox
   1. 布局

<CheckBox

Android:id=”@+id/swim” />

<CheckBox

Android:id=”@+id/read” />

<CheckBox

Android:id=”@+id/run” />

* 1. 代码

CheckBox继承了CompoundButton

swimBox.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangedListener(){

@Override

Public void onCheckedChanged(CompoundButton buttonView,Boolean isChecked){

If(isChecked){};

}

});

1. Toast

提示信息

Toast.makeText(RadioTest.this,”female”,Toast.LENGTH\_SHORT).show();//（当前类，提示信息，长度）

1. ProgressBar
   1. 进度条
   2. <ProgressBar

android:id=”@+id/firstBar”

style=”?android:attr/progressBarStyleHorizontal” //水平方向的风格

android:visibility=”gone” /> //是否可视,gone表示不可视

<ProgressBar

Android:id=”@+id/secondBar”

Style=”?android:attr/progressBarStyle” //默认的风格，圆圈央视

Android:visibility=”gone”

Android:max=”200”> //设置进度条的最大值

* 1. firstBar.setVisibility(View.VISIBLE);//设置进度条处于可见状态

firstBar.setPaogress(i);//设置进度条的当前值

firstBar.setSeconddaryProgress(i+10);//设置第二进度条的当前值

firstBar.setMax(150);//设置进度条最大值

1. ListView
   1. 列表
   2. 继承ListAvtivity

<ListView

android:id=”@id/android:list”

android:drawSelectorOnTop=”false”

android:scrollbars=”vertical” />//滚动条方向

ArrayList<HashMap<String,String>> list = new ArrayList<HashMap<String,String>>();

SimpleAdapter listAdapter = new SimpleAdapter(this,list,R.layout.user,new String(“user\_name”,”user\_ip”),new int[]{R.id.user\_name,R.id.user\_ip});//适配器

setListAdapter(listAdapter);

@Override

Protected void onListItemClick(ListView l, View v, int position, long id)

{

Super.onListItemClick(l,v,postion,id);

}

1. 创建Android Project
   1. 向上兼容
   2. 选择1.6的SDK
   3. Application Name ——应用程序的名称
   4. Create Activity —— 显示的名称
   5. Min SDK Version ——兼容的最低版本
2. Project 目录介绍

gen 目录，插件自动生成，R.java不能编辑

assets 可以防止任意的人间

res 其中的人间会在R.java中生成一个Id

layout main.xml——布局文件

values string.xml——键值对，会在R.java中生成Id

1. AndroidManifest 整个应用程序的配置
2. Activity
   1. Activity的作用
      1. 是一个用户接口，和用户进行界面的交互，空间的容器；
   2. Activity的步骤
      1. 一个Activity就是一个类，并且要继承Activity
      2. 需要复写onCreate()方法
      3. 每个Activity都要在Manifest.xml文件中配置，配置了<intent-activity>就是第一启动画面
      4. 为Activity添加控件
3. Activity和Intent的关系

Activity01 ——Intent intent——Activity02

Intent对象包括：

* 1. Component name ——启动哪一个组件
  2. Action——下一个Activity要做的动作
  3. Data——数据
  4. Category
  5. Extras——额外的一些信息、键值对
  6. Flags

生成一个Intent对象

Intent intent = new Intent();

Intent.setClass(Activity02.this,OtherActivity.class);

Activity02.this.startActivity(intent);

1. Activity的生命周期
2. 生命周期函数

启动一个新的Activity的时候会调用下面三个生命周期函数：

onCreat()——》当Activity第一个被创建的时候就会调用这个方法

onStart()——》当Activity能被户用看到的时候会调用这个方法

onResume()——》当Activity能获得用户焦点的时候调用这个方法

启动第二个Activity的时候：

第一个Activity的onPause()——》当应用程序启动了另外一个Activity的时候调用

onPause()当应该写保存当前数据的代码；

第二个Activity的onCreate() onStart() onResume()方法

第一个Activity的onStop()方法——》当这个Activity处于不可见的状态的时候，调用这个方法（需要全部遮挡），处于停止状态

点击Back之后

第二个Activity的onPause()

第一个Activity的onRestart()

第一个Activity的onStart()

第一个Activity的onResume()

第二个Activity的onStop()

第二个Activity的onDestory()//第二个被销毁，有两种情况会调用，（1）调用了 finish()方法；（2）系统资源不够用

1. Task

Task就是一个栈（先进后出），里面存放了很多Activity，压栈，弹栈

当调用了finish方法之后就会从栈里面消失

3. 窗口风格的Activity的实现

其他没有区别，区别在AndroidManifest.xml文件中

<activity android:name=”.SecondActivity”

android:label=”SecondActivity”

android:theme=”@android:style/Theme.Dialog”/>

指定android:theme=”@android:style/Theme.Dialog”

1. Activity 布局
2. LinearLayout——线性布局

android:orientation=”vertical”//线性布局的方向，vertical为垂直，horizontal为水平

android:layout\_width=”fill\_parent”//宽度填充父控件

android:layout\_height=”wrap\_parent”//高度显示内容即可

android:id //为控件指定相应的Id

android:text //指定控件当中显示的文字，尽量使用string.xml

android:gravity //指定控件里面的内容的基本位置，比如说居中、居右等位置

android:textSize //指定控件当中字体的大小

android:background //指定控件所使用的背景颜色，RGB命名法

android:width //指定控件的宽度

android:height //指定控件的哦高度

android:padding\* //指定控件内容的内边距，也就是说控件当中的内容

android:sigleLine //如果设置为真的话，则将控件的内容在同一行进行显示

android:weight //控件的占的比例

1. TableLayout——表格布局

<TableRow>行

android:stretchColumns=”1” //使用第一列作为拉升的一列以便填充父控件（从0位计数）

3．RelativeLayout——相对布局

（1）两个控件之间的上下左右对齐

Android:layout\_above——将控件的底部至于给定Id的控件之上

android:layout\_above=”@id/xxx”

android:layout\_below——将控件的顶部至于给定Id的控件之下

android:layout\_below=”@id/xxx”

android:layout\_toLeftOf——将控件的右边缘与给定Id的控件的左边缘对齐（即在左边）

android:layout\_toLeftOf=”@id/xxx”

android:layout\_toRightOf——

（2）对齐

android:layout\_alignBaseline

android:layout\_alignBottom

android:layout\_alignRight

android:layout\_alignLeft

android:layout\_alignTop

（2）和父控件进行对齐

Android:alignParentBottom ——如果为True，则将该控件的底部和父控件的底部对齐

android:layout\_alignParentLeft——如果该值为true，则该控件的左边与父控件的左边对齐

android:layout\_alignParentRight

android:layout\_alignParentTop

android:layout\_centerHorizontal ——如果为True，该控件将被至于水平方向的中央

android:layout\_centerInParent ——如果为true,该控件将被至于父控件水平方向和垂直方向

android:layout\_centerVertical——如果为true，该控件将被置于垂直方向的中央

1. Handler
   1. handler.postDelayer(Thread.class,3000);//隔3秒把线程加入到消息队列中
   2. 一个handler就是一个消息队列，消息队列都是先进先出的顺序
   3. handler.post(Thread.class);//马上加入到消息队列中
   4. handler.removeCallbacks(Thread.class);//删除线程
   5. 步骤：创建一个Handler对象；调用Handler的Post方法，将要执行的线程对象添加到队列当中；将要执行的操作写在线程对象的run方法中；在run方法内部，执行postDelayer或者是post方法。
   6. 提供了异步线程处理方案

Handler handler = new Handler();

Runnable updateThreead = new Runnable(){

@Override

Public void run(){

System.out.println(“Update Thread”);

handler.postDelayer(updateThread,3000);

}

}

1. Bundle
   1. 是一个以String为Key的Map，类似于HashMap
   2. Key的类型必须是String，Value也是限于几种数据类型

十一、HandlerThread

生成一个HandlerTHread对象，实现了使用Looper来处理消息队列的功能，这个类由Android应用成框架提供

HandlerThread handlerThread = new HandlerThread(“Handler\_Thread”);

handlerThread.start();

MyHandler myHandler = new MyHandler(handlerThread.getLooper());

Message msg = myHandler.obtainMessage();

Msg.sendToTarget();//发Msg发送到目标（哪一个Handler生成的Msg）

Class MyHandler extends Handler{

Public MyHandler(){}

Public MyHandler(Looper looper){super(looper);}

}

十二、SQLite

1. 关系型数据库
2. SQLiteOpenHelper使用
   1. getReadableDataBase()——得到一个可读状态的数据库
   2. gteWriteableDataBase()——得到一个可写状态的数据库
   3. onCreate();——回调函数，当创建数据库的时候
   4. onOpen();——当打开数据库的时候
   5. onUpdate();——当更新数据库的时候
3. 代码

DataBaseHelper extends SQLiteOpenHelper

构造函数

插入数据

ContentValues values = new ContentValues();

values.put(“id”,1);

values.put(“name”,”zhangsan”);

DatabaseHelper dbHelper = new DatabaseHelper(SQLiteActivity.this,”test”);

SQLiteDatabase db = dbHelper.getWriteableDatabase();

db.insert(“user”,null,values);//（表名、null、值）

更新数据

//update table\_name set XXCOL=XXX where XXCOL=XXX…

DatabaseHelper dbHelper = new DatabaseHelper(S QLiteActivity.this,”test”);

SQLiteDatabase db = dbHelper.getWriteableDatabase();

ContentValues values = new ContentValues();

values.put(“name”,”lisi”);

//第一个参数是要更新的表名

//第二个参数是一个ContentValues对象

//第三个参数是where子句

db.update(“user”,values,”id=?”,new String[]{“1”});

查询数据

DatabaseHelper dbHelper = new DatabaseHelper(S QLiteActivity.this,”test”);

SQLiteDatabase db = dbHelper.getReadableDatabase();

Cursor cursor = db.query(“user”,new String[]{“id”,”name”},”id=?”,new String[]{“1”});

While(cursor.moveToNext())

{

//先获取列的编号

String name = cursor.getString(cursor.getColumnIndex(“name”));

System.out.println(“query-----🡪” + name);

}

十三、HTTP协议下载文件

1. Http协议下载
   1. 创建httpURLConnection对象
   2. 获取InputStream
      1. urlConn.getInputSteam();
   3. 访问网络权限
      1. android.permission.INTERNET
2. 下载的文件写入SDCARD
   1. 得到SD卡的目录

Environment.getExternalStorageDirectory();

* 1. 访问SD卡的权限

android.permission.WRITE\_EXTERNAL\_STORAGE

十四、Content Provider

1. 提供为存储和获取数据提供了统一的接口
2. 可以在不同应用程序之间共享数据
3. 音频、视频、图片、通讯录等等
4. 常用函数：
   1. Query();//查询
   2. Insert();//插入
   3. Update();//更新
   4. Delete();//删除
   5. getType();//得到数据类型
   6. onCreate();//创建时的回调函数
5. 每一个ContentProvider都拥有一个公共的URI，这个URI用于表示这个ContentProvider所提供的数据
6. 实现一个Content Provider的过程
   1. 定义一个CONTENT\_URI常量
   2. 定义一个类，继承ContentProvider
   3. 实现Query、Insert、Update、Delete、GetType和Oncreate方法
   4. 在AndroidManifest.xml当中进行申明

十五、 广播机制

1. 继承BroadcastReceiver
2. <!—决定receiver接收什么事件-->

<receiver android:name=*".TestReceiver"*>

<intent-filter>

<action android:name=*"android.intent.action.EDIT"*/>

</intent-filter>

</receiver>

1. 代码
   1. Intent intent = new Intent();

intent.setAction(Intent.ACTION\_EDIT);//设置Intent的Action

TestActivity.this.sendBroadcast(intent);//发送广播到感兴趣的接收receiver

1. 两种注册方式
   1. 在代码中注册
      1. 注册——registerReceiver(reveiver,filter);

IntentFilter filter = new IntentFilter();

Filter.addAction(SMS\_ACTION);

TestActivity.this.registerReceiver(smsReceive,filter);

* + 1. 取消注册——unregisterReceiver(receiver);
  1. 在AndroidManifest.xml文件中注册，如上（2），区别在于如果应用程序关闭了，还是会收到这个广播事件；

Android开发规范之编码规范

1，java代码中不出现中文，最多注释中可以出现中文，中文统一写在strings.xml中；

2，局部变量命名、静态成员变量命名 只能包含字母，单词首字母大写，其他字母都为小写

3，常量命名只能包含字母和\_，字母全部大写，单词之间用\_隔开

4，布局文件中的id命名

　　命名模式为：view缩写\_模块名称\_view的逻辑名称

　　常用控件的缩写如下

　　LayoutView：lv\_

　　RelativeView : rv\_

　　TextView : tv\_

　　ImageView : iv\_

　　ImageButton : im\_

　　Button : btn\_

5，strings.xml中的id命名

　　命名模式：activity名称\_功能模块名称\_逻辑名称/activity名称\_逻辑名称/common\_逻辑名称

　　strings.xml中，使用activity名称注释，将文件内容区分开来

6，styles.xml：将layout中不断重现的style提炼出通用的style通用组件，放到styles.xml中；

7，服务端可以实现的，就不要放在客户端；

8，图片尽量分拆成多个可重用的图片

9，引用第三方库要慎重，避免应用大容量的第三方库，导致客户端包非常大；

10，图片要.9.png处理

11，使用静态变量方式实现界面间共享要慎重；

12，Log(系统名称 模块名称 接口名称，详细描述)；

13，不要重用父类的handler，对应一个类的handler也不应该让其子类用到，否则会导致message.what冲突

14，strings.xml中使用%1$s实现字符串的通配

15，如果多个Activity中包含共同的UI处理，那么可以提炼一个CommonActivity，把通用部分叫由它来处理，其他activity只要继承它即可

16，数据一定要效验，例如字符型转数字型，如果转换失败一定要有缺省值；服务端响应数据是否有效判断；

----------------------------------------------------------------------------------------------

如何提高Android的效率，也就是如何来优化Android的性能：

1，http用gzip压缩，设置连接超时时间和响应超时时间

　　http请求按照业务需求，分为是否可以缓存和不可缓存，那么在无网络的环境中，仍然通过缓存的httpresponse浏览部分数据，实现离线阅读。

2.listview 性能优化

　　1).复用convertView

　　　　在getItemView中，判断convertView是否为空，如果不为空，可复用。如果couvertview中的view需要添加listerner，代码一定要在if(convertView==null){}之外。

　　2).异步加载图片

　　　　item中如果包含有webimage，那么最好异步加载

　　3).快速滑动时不显示图片

　　　　当快速滑动列表时（SCROLL\_STATE\_FLING），item中的图片或获取需要消耗资源的view，可以不显示出来；而处于其他两种状态（SCROLL\_STATE\_IDLE 和　　　　　　　　　　　　　　　　　　  　　   SCROLL\_STATE\_TOUCH\_SCROLL），则将那些view显示出来

　　4).BaseAdapter避免内存溢出

　　　　如果BaseAdapter的实体类有属性非常消耗内存，可以将保存到文件；为提高性能，可以进行缓存，并限制缓存大小。

3.使用线程池，分为核心线程池和普通线程池，下载图片等耗时任务放置在普通线程池，避免耗时任务阻塞线程池后，导致所有异步任务都必须等待

4.异步任务，分为核心任务和普通任务，只有核心任务中出现的系统级错误才会报错，异步任务的ui操作需要判断原activity是否处于激活状态

5.尽量避免static成员变量引用资源耗费过多的实例,比如Context

6.使用WeakReference代替强引用，弱引用可以让您保持对对象的引用，同时允许GC在必要时释放对象，回收内存。对于那些创建便宜但耗费大量内存的对象，即希望保持该对象，又要在应　　　　  　　用程序需要时使用，同时希望GC必要时回收时，可以考虑使用弱引用。

7.超级大胖子Bitmap

　　及时的销毁(Activity的onDestroy时将bitmap回收，在被UI组件使用后马上进行回收会抛 RuntimeException: Canvas: trying to use a recycled bitmap android.graphics.Bitmap) 设置一定的采样率(有开发者提供的图片无需进行采样，对于有用户上传或第三方的大小不可控图片，可进行采样减少图片所占的内存)，从服务端返回图片，建议同时反馈图片的size 巧妙的运用软引用 drawable对应resid的资源，bitmap对应其他资源 任何类型的图片，如果获取不到（例如文件不存在，或者读取文件时跑OutOfMemory异常），应该有对应的默认图片（默认图片放在在apk中，通过resid获取）； 8.保证Cursor 占用的内存被及时的释放掉，而不是等待GC来处理。并且 Android明显是倾向于编 程者手动的将Cursor close掉

9.线程也是造成内存泄露的一个重要的源头。线程产生内存泄露的主要原因在于线程 生命周期的不可控

10.如果ImageView的图片是来自网络，进行异步加载

11.应用开发中自定义View的时候，交互部分，千万不要写成线程不断刷新界面显示，而是根据TouchListener事件主动触发界面的更新

12.Drawable

　　ui组件需要用到的图片是apk包自带的，那么一律用setImageResource或者setBackgroundResource，而不要根据resourceid

注意：get(getResources(), R.drawable.btn\_achievement\_normal)该方法通过resid转换为drawable，需要考虑回收的问题，如果drawable是对象私有对象，在对象销毁前是肯定不会释放内存的。

13.复用、回收Activity对象临时的activity及时finish主界面设置为singleTask一般界面设置为singleTop

14.位置信息获取用户的地理位置信息时，在需要获取数据的时候打开GPS，之后及时关闭掉

15.在onResume时设置该界面的电源管理，在onPause时取消设置