## 对象的初始化顺序

父类的静态代码块、子类的静态代码块

main方法

父类的成员变量初始化、父类的构造代码块、父类的构造函数

子类的成员变量初始化、子类的构造代码块、子类的构造函数

## Socket

服务端

1. ServerSocket server = new ServerSocket(端口);
2. Socket cliend = server.accept();
3. 接受客户端发送过来的数据

BufferedReader reader = new BufferedReader(new InputStreamReader(cliend.getInputStream()));

1. 发送给客户端

printStream out = new PrintStream(cliend.getOutputStream());

客户端

1. 建立连接 Sokcet cliend = new Scoket(ip,端口);
2. 设置超时时间 cliend.setSoTimeout(10000);
3. 发送到服务器上

PirntStream print = new PrintStream(cliend.getOutputStream());

1. 接受服务器的数据

BufferedReader readerServer = new BufferedReader(new InputStreamReader(cliend.getInputStream()));

1. 获取从键盘输入的数据

BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));

Android Context

1. 通过它可以获取应用程序的资源和类，包括应用级别的操作，例如启动Activity，发送广播，接受Intent信息，创建Application对象，创建Service对象等。
2. ContextIml.java类 该函数的大部分功能都是通过调用其属性mPackageInfo去完成
3. ContextThemeWrapper包含了主题相关的接口android:theme属性指定的。只有Activity需要主题，service不需要主题，所以service继承自ContextWrapper，Activity继承自ContextThemeWrapper
4. 每个应用程序在第一次启动时都会创建一个Application对象

## 广播

用户自定义广播

1. 注册广播

BroadcastReceiverA receiver = New BroadcastReceiverA();

IntentFilter filter = new IntentFilter();

Filter.addAction(“action”);

registerReceiver(receiver,filter);

1. 发送广播

Intent intent = new Intent(“action”);

sendBroadcast(intent);

1. 广播

Class BroadcastReceiverA{

onReceive(){

收到广播要做的事情

}

}

系统广播

1. 注册广播

BroadcastReceiverA receiver = New BroadcastReceiverA();

IntentFilter filter = new IntentFilter();

Filter.addAction(Intent.ACTION\_...);

registerReceiver(receiver,filter);

**不用发送广播，不允许用户发送系统广播**

1. 广播

Class BroadcastReceiverA{

onReceive(){

收到广播要做的事情

}

}

## Adapter

ArrayAdapter 最简单只能展示一行字

public class MainActivity extends AppCompatActivity {  
  
 ListView listView;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*list\_view*);  
 listView = new ListView(this);  
 listView.setAdapter(new ArrayAdapter<String>(this,android.R.layout.*simple\_expandable\_list\_item\_1*,getData()));  
 setContentView(listView);  
 }  
  
 public List<String> getData(){  
 List<String> data = new ArrayList<String>();  
 data.add("testone");  
 data.add("testtwo");  
 data.add("testthree");  
 data.add("testfoure");  
 data.add("five");  
 return data;  
 }

Simpleadapter一般都是HashMap构成的List，list的每一节对应ListView的每一行，扩展性好

public class MainActivity extends ListActivity {  
  
 ListView listView;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
  
 SimpleAdapter adapter = new SimpleAdapter(this,getData(),R.layout.*list\_view*,new String[]{"ssid","bssid"},new int[]{R.id.*ssid*,R.id.*bssid*});  
 setListAdapter(adapter);  
 }  
  
 public List<Map<String,String>> getData(){  
  
 List<Map<String,String>> data = new ArrayList<Map<String,String>>();  
 Map<String,String> map1 = new HashMap<String,String>();  
 map1.put("ssid","wifi1");  
 map1.put("bssid","123456");  
 data.add(map1);  
  
 Map<String,String> map2 = new HashMap<String,String>();  
 map2.put("ssid", "wifi2");  
 map2.put("bssid", "1234567");  
 data.add(map2);

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="vertical" android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/ssid"/>  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/bssid"/>  
  
</LinearLayout>

Simplecursoradapter 一定要以数据库为数据源的时候才使用simplecursoradapter

Baseadapter 有按钮需要点击，因为按钮是无法映射的，即使你成功的用布局文件显示出了按钮也无法添加按钮的响应，使用baseadapter处理点击事件。

public class MainActivity extends ListActivity {  
  
 public List<Map<String,String>> data;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
  
 data = getData();  
 MyAdapter myAdapter = new MyAdapter(this);  
 setListAdapter(myAdapter);  
  
 }  
  
 @Override  
 protected void onListItemClick(ListView l, View v, int position, long id) {  
 Log.*e*("click",(String)data.get(position).get("name") + "is clicked");  
 }  
  
 public List<Map<String,String>> getData(){  
  
 List<Map<String,String>> data = new ArrayList<Map<String,String>>();  
 Map<String,String> map1 = new HashMap<String,String>();  
 map1.put("name","one");  
 map1.put("text","test1");  
 data.add(map1);  
  
 Map<String,String> map2 = new HashMap<String,String>();  
 map2.put("name","two");  
 map2.put("text", "test2");  
 data.add(map2);  
  
 Map<String,String> map3 = new HashMap<String,String>();  
 map3.put("name","three");  
 map3.put("text","test3");  
 data.add(map3);  
  
 Map<String,String> map4 = new HashMap<String,String>();  
 map4.put("name","foure");  
 map4.put("text","teset4");  
 data.add(map4);  
  
 return data;  
 }

public class MyAdapter extends BaseAdapter {  
 private LayoutInflater flater;  
 MainActivity mainActivity;  
  
 MyAdapter(Context context){  
 mainActivity = new MainActivity();  
 this.flater = LayoutInflater.*from*(context);  
 }  
  
 @Override  
 public int getCount() {  
 return mainActivity.getData().size();  
 }  
  
 @Override  
 public Object getItem(int position) {  
 return null;  
 }  
  
 @Override  
 public long getItemId(int position) {  
 return position;  
 }  
  
 @Override  
 public View getView(final int position, View convertView, ViewGroup parent) {  
 ViewHolder holder = null;  
 if(holder == null){  
 holder = new ViewHolder();  
 convertView = flater.inflate(R.layout.*list\_view*,null);  
 holder.name = (TextView)convertView.findViewById(R.id.*name*);  
 holder.text = (TextView)convertView.findViewById(R.id.*text*);  
 holder.button = (Button)convertView.findViewById(R.id.*button*);  
 convertView.setTag(holder);  
 }  
 else{  
 holder = (ViewHolder) convertView.getTag();  
 }  
  
 holder.name.setText((String)mainActivity.getData().get(position).get("name"));  
 holder.text.setText((String)mainActivity.getData().get(position).get("text"));  
 holder.button.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Log.*e*("click","click ~~~~~~~~~`" + getItemId(position));  
 }  
 });  
  
 return convertView;  
 }  
  
 public class ViewHolder{  
 public TextView name;  
 public TextView text;  
 public Button button;  
 }  
}

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="vertical" android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
  
 <LinearLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content">  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/name"/>  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/text"/>  
  
 </LinearLayout>  
  
 <Button  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/button"/>  
  
</LinearLayout>

## LayoutInflater

用来加载布局，setContentView（）其实也是用LayoutInflater加载布局的