SmallTips

- && 与 & 的区别：&&前面为false则不执行后面，&前后都执行
- 加密算法：
- Java中，允许父类引用(即定义的变量)，指向子类对象。eg：`Person p = new Man()`。类的多态性，允许向上转型。
- **观察者模式**适合解决对象之间的依赖关系
- 更新UI的两种方式：handler机制，AsyncTask，Runnable
- handler的两种用途：更新UI，线程间通信
- Handler发送两种消息：`handler.sendMessage(new Message())` 或 `handler.post(new Runnable())`,即表面handler发送 `Message` 和 `Runnable`,实际会将 `Runnable` 封装成 `Message`

知识点

设计模式

- 工厂模式
- 外观模式
- 单例模式
- 观察者模式：类似一种**通知机制**，定义了对象间的一种一对多的依赖关系，以便当一个对象状态发生改变时，所有依赖于他的对象都得到通知并自动刷新。eg：

> 回调：
>
> 子：定义接口&调用接口方法；主：实现多个接口；
>
> 一个类实现多个接口；
>
> 一个接口，被多个类实现

volatile 与 sychroized 关键字

volatile ['vɒl.ə.taɪl]能保证数据可见性，不能完全保证数据原子性；

synchroized [sin k raz] 能保证数据的可见性和原子性

Intent 传递数据的方式

```
// 1.传递Uri
Intent intent = new Intent();
intent.sendData(new Uri());
Uri uri =intent.getData();

// 2.通过Extra传递 (k,v)
intent.putExtra(k,v); // 基本数据类型，数组，Bundle,Serializable,Parcelable

// 3.通过Extra 传递Bundle
intent.putExtras(new Bundle());
intent.putExtra(key,new Bundle());
```

UI更新与数据更新

android采用，UI单线程模型，即只能在主线程中更新UI。当数据运算较大，将数据更新放在主线程中会阻塞主线程。**即子线程中数据更新，使用Handler进行子线程-主线程通信，主线程UI更新。**

```
import android.os.Handler

public class MainActivity extends Activity(){

    Handler mHandler = new Handler(){
        @override
        public void handleMessage(msg){
            super.handlerMessage();

            // 获取数据
            // msg.what/arg1/agr2 -> int,简单消息节省资源
            // msg.obj
            // msg.getData -> Bundle
            // ...
            // UI更新

        }
    }

    // 子线程数据更新
    Thread dataUpdateThread = new Thread(new Runnable(){
        @override
        public void run(){
            // 1.更新数据..
            // 2.从消息池中获取msg，节省资源
            // 3.发送消息
            Message msg = Message.obtain();
            mHandler.sendMessage(msg);


        }
    })

    @override
    protected void onCreate(Bundel saveInstanceState){
        dataUpdateThread().start();
    }

}
```

Handler机制

用途：主要用于实现线程间通信，UI更新

更新UI的三种方式

1.handler机制，子线程中发送消息，主线程在处理消息；

2.AsyncTask，

3.Runnable，子线程中使用 `runOnUiThread(new Runnable())` ；类似的

```
ProgressDialog dialog = new ProgressDialog();
dialog.setProgress(); // 在子线程中执行
```

Parcelable，Serializable是什么东西

Drawable是个抽象类

```
Drawable是个抽象类
Bitmap implements Parcelable
```

```
Drawable
```