实验编号：5 **四川师大Android高级开发实验报告 2020**年 **4**月 **17**日

**计算机科学学院** 级 3 班 实验名称： 多线程

姓名： 吴文娇 学号： 2017110543 指导老师： 樊相奎 实验成绩:\_\_\_\_\_

**实验 五 多线程**

1. 实验目的及要求

目的：

(1) 验证多线程下同步 UI 的实现方式

要求：

(1) 认真填写实验报告，要求附加部分运行界面和主要代码;

(2) 对设计好的程序，检查输出是否符合预期，如有错请分析错误原因并解决;

1. 实验内容

(1) 在子线程中进行模拟下载，下载过程中发送通知消息;

(2) 在主线程的 Handler 中处理消息，更新界面;

(3) 采用 AsyncTask 派生类实现多线程异步下载;

1. 实验主要流程、基本操作或核心代码、算法片段（该部分如不够填写，请另加附页）
2. 在子线程中进行模拟下载，下载过程中发送通知消息;
3. **package** com.example.fairy.myapplication;  
   **import** android.nfc.Tag;  
   **import** android.os.AsyncTask;  
   **import** android.os.Handler;  
   **import** android.os.Message;  
   **import** android.support.v7.app.AppCompatActivity;  
   **import** android.os.Bundle;  
   **import** android.util.Log;  
   **import** android.view.View;  
   **import** android.widget.ProgressBar;  
   **import** android.widget.TextView;  
     
   **import** org.w3c.dom.Text;  
     
   **import** java.lang.ref.WeakReference;  
     
   **public class** MainActivity **extends** AppCompatActivity {  
     
    **private static final** String ***TAG*** = **"MainActivity"**;  
    TextView **textView**;  
    **private boolean isStarted** = **false**;*//控制一次只产生一个线程* ProgressBar **progressBar**;  
    MyAsyncTask **asyncTask**;  
     
    @Override  
    **protected void** onCreate(Bundle savedInstanceState) {  
    **super**.onCreate(savedInstanceState);  
    setContentView(R.layout.***activity\_main***);  
    **textView** = findViewById(R.id.***textView***);  
    **progressBar** = findViewById(R.id.***progressBar***);  
    Log.*d*(***TAG***,**"onCreate: "**+Thread.*currentThread*().getName()); *//onCreate: main(主线程内运行)  
    //asyncTask = new MyAsyncTask(this);* }  
     
    **static class** MyAsyncTask **extends** AsyncTask<Integer,Integer,Integer>{  
    *//传的参数，中间进度，结果* WeakReference<MainActivity> **weakReference**;  
    MyAsyncTask(MainActivity activity){  
    **this**.**weakReference** = **new** WeakReference<MainActivity>(activity);  
    }  
     
    @Override  
    **protected void** onPreExecute() {  
    **super**.onPreExecute();  
    Log.*d*(***TAG***, **"onPreExecute: "**);  
    }  
     
    @Override  
    **protected** Integer doInBackground(Integer... integers) {  
    Log.*d*(***TAG***,**"doInBackground"**+Thread.*currentThread*().getName());  
    **int** sum = 0;  
    **for** (**int** i = 0; i < 100; i++) {  
    sum += i;  
    **try** {  
    Thread.*sleep*(100);  
    } **catch** (InterruptedException e) {  
    e.printStackTrace();  
    }  
    **if** (isCancelled()) **break**;  
    publishProgress(i);*//传进度* }  
    **return null**;  
    }  
     
    @Override  
    **protected void** onCancelled() {  
    **super**.onCancelled();  
    MainActivity activity = **weakReference**.get();  
    **if** (activity!=**null**) activity.**textView**.setText(**"Canceled!"**);  
    }  
     
    @Override  
    **protected void** onProgressUpdate(Integer... values) {  
    MainActivity activity = **weakReference**.get();  
    **if** (activity!=**null**) activity.**textView**.setText(**"progress"**+values[0]);  
    **if** (activity!=**null**) activity.**progressBar**.setProgress(values[0]);  
    **super**.onProgressUpdate(values);  
    }  
     
    @Override  
    **protected void** onPostExecute(Integer integer) {  
    MainActivity activity = **weakReference**.get();  
    **if** (activity!=**null**) activity.**textView**.setText(**"Sucess!"**);  
    **super**.onPostExecute(integer);  
    }  
    }  
     
    **public void** start(View v) {  
    **asyncTask** = **new** MyAsyncTask(**this**);  
    *//AayncTask不能完全取代线程，Task只能被执行一次，一些逻辑较为复杂或者需要后台反复执行的逻辑可能就需要线程来完成了  
   // 所以本题cancel之后再次start会报异常  
   //修改；每次cancel完之后又start的时候构造一个新的  
     
   //问题：内存泄漏  
   //修改:static class MyAsyncTask , 弱引用 ，构造函数.....* **asyncTask**.execute();*//execute方法必须在UI线程中被调用* }  
     
    **public void** stop(View v){  
    **asyncTask**.cancel(**true**);  
    }  
   }

(2) 在主线程的 Handler 中处理消息，更新界面;

**package** com.example.fairy.myapplication;  
  
**import** android.nfc.Tag;  
**import** android.os.Handler;  
**import** android.os.Message;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.util.Log;  
**import** android.view.View;  
**import** android.widget.ProgressBar;  
**import** android.widget.TextView;  
  
**import** org.w3c.dom.Text;  
  
**import** java.lang.ref.WeakReference;  
  
*//1.从handler派生一个类（不内存泄露）  
//2.弱引用  
//3.要用中间终止状态（考虑加锁）***public class** MainActivity **extends** AppCompatActivity {  
  
 **private static final** String ***TAG*** = **"MainActivity"**;  
 TextView **textView**;  
 **private** Myhandler **handler**;  
 **private boolean isStarted** = **false**;*//控制一次只产生一个线程* ProgressBar **progressBar**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 **textView** = findViewById(R.id.***textView***);  
 **progressBar** = findViewById(R.id.***progressBar***);  
 Log.*d*(***TAG***,**"onCreate: "**+Thread.*currentThread*().getName()); *//onCreate: main(主线程内运行)* **handler** = **new** Myhandler(**this**);  
 }  
  
*//内存泄漏  
// private Handler handler = new Handler(){  
// @Override  
// public void handleMessage(Message msg) {  
// super.handleMessage(msg);  
// switch(msg.what){  
// case 0:  
// activity.textView.setText("Process"+msg.arg1);  
// break;  
// case 1:  
// activity.textView.setText("Sucess!");  
// break;  
// }  
// }  
// };  
  
 //静态内部类* **static class** Myhandler **extends** Handler{  
 *//弱引用* WeakReference<MainActivity> **weakReference**;  
 Myhandler(MainActivity activity){  
 **this**.**weakReference** = **new** WeakReference<MainActivity>(activity);  
 }  
  
 @Override  
 **public void** handleMessage(Message msg) {  
 **super**.handleMessage(msg);  
 MainActivity activity = **weakReference**.get();  
 **if**(activity == **null**) **return**;  
 **switch**(msg.**what**){  
 **case** 0:  
 activity.**textView**.setText(**"Process"**+msg.**arg1**);  
 activity.**progressBar**.setProgress(msg.**arg1**);  
 **break**;  
 **case** 1:  
 activity.**textView**.setText(**"Sucess!"**);  
 activity.**isStarted** = **false**;  
 **break**;  
 **case** 2:  
 activity.**textView**.setText(**"Cancel!"**);  
 **break**;  
 }  
  
 }  
 }  
 *//同一个进程里的不同线程共享变量  
 //存在问题：现在主线程和子线程都在访问同一个变量textView（a在访问，b又来了，就中断，b修改，a也修改，就会报错）  
 //操作系统上的解决措施：加锁(生产者-消费者问题)，但是如果在两个线程都加锁，需要繁杂的加锁解锁操作，并且界面每1s普遍刷屏60次（资源负担重）  
 //优化方案：子线程不访问：子线程给主线程发消息，让主线程帮自己做事情（间接访问）  
 //Handler（专门处理多线程问题）的使用场合：  
 //(一) 安排messages和runnables在将来的某个时间点执行。  
 //(二) 将action入队以备在一个不同的线程中执行。即可以实现线程间通信。  
 // 比如当你创建子线程时，你可以再你的子线程中拿到父线程中创建的Handler对象，就可以通过该对象向父线程的消息队列发送消息了。  
 // 由于Android要求在UI线程中更新界面，因此，可以通过该方法在其它线程中更新界面。  
 // 通过Handler更新UI实例：步骤：  
 // 1、创建Handler对象（此处创建于主线程中便于更新UI）。  
 // 2、构建Runnable对象，在Runnable中更新界面。  
 // 3、在子线程的run方法中向UI线程post，runnable对象来更新UI。  
  
 //不能在其他线程上访问界面上的元素* **public void** start(View v) {  
 **synchronized** (**this**) {  
 *//加锁* **if** (**isStarted**) **return**;  
 }  
 *//isStrateed一开始定义的是false，所以需要先置成true* **isStarted** = **true**;  
 **new** Thread(**new** Runnable() {  
 @Override  
 **public void** run() {  
 **int** sum = 0;  
 Log.*d*(***TAG***,**"run: "**+Thread.*currentThread*().getName()); *//run: Thread-4（子线程内执行按键事件）  
 //only the original thread that created a view hierarchy can touch its views.  
 //android中相关的view和控件不是线程安全的，我们必须单独做处理。这里借此引出Handler的使用。  
 //android规定：1.不阻塞UI线程 2.不在UI线程之外访问Android Ui工具包* **for**(**int** i = 0; i < 100; i++){  
 *//for(int i = 0; i < 10000; i++){  
 //大耗时的如果不放在线程里面将会造成button点了之后不计算并且不会在点动按键的操作* sum += i;  
 *//捕获异常* **try**{  
 Thread.*sleep*(100);  
 } **catch** (InterruptedException e) {  
 e.printStackTrace();  
 }  
 *//期待在中间终止  
 //判断是否结束* **synchronized** (**this**){  
 **if**(!**isStarted**) {  
 *//结束之前发消息通知* Message msg = **handler**.obtainMessage();  
 msg.**what** = 2;  
 **handler**.sendMessage(msg);  
 **return**;  
 *//里面要返回两个消息，所以不用break，直接用return* }  
 }  
 *//下面这段注释掉的代码和what=0的那一段代码选择一个留下，两种传参方法 ）  
 //一般还是建议发消息（不过简单应用可以用本方法）（避免内存泄露）  
// final int j = i;  
// handler.post(new Runnable(){ //post也是sendmessage （拿到消息之后调用callback回调函数（容易内存泄露））  
// @Override  
// public void run(){  
// textView.setText("Process"+j);  
// progressBar.setProgress(j);  
// }  
// });  
 //textView.setText("Progress:" + i);* Message msg = **handler**.obtainMessage();  
 msg.**what** = 0;  
 msg.**arg1** = i;  
 **handler**.sendMessage(msg);  
 }  
 *//textView.setText("sum=" + sum);* Message msg = **handler**.obtainMessage();  
 msg.**what** = 1;  
 **handler**.sendMessage(msg);  
*// //在子线程里面结束（也可在主线程（MyHandler类里面）里面结束，就不再需要加锁）  
// synchronized (this) {  
// isStarted = false; //又产生子线程祝线程都访问同一个变量（虽然布尔值产生内存泄漏威胁小） ／完善：加锁（操作系统机制）  
// }* }  
 }).start();  
 }  
  
 **public void** stop(View v){  
 **synchronized** (**this**) {  
 **isStarted** = **false**;  
 }  
 }  
}

(3) 采用 AsyncTask 派生类实现多线程异步下载;

***activity\_main.xml***

*<?***xml version="1.0" encoding="utf-8"***?>*<**android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context="com.example.fairy.myapplication.MainActivity"**>  
  
 <**EditText  
 android:id="@+id/editText\_url"  
 android:layout\_width="394dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="8dp"  
 android:layout\_marginEnd="8dp"  
 android:layout\_marginLeft="16dp"  
 android:layout\_marginTop="8dp"  
 android:ems="10"  
 android:hint="input image url"  
 android:inputType="textPersonName"  
 app:layout\_constraintBottom\_toTopOf="@+id/button2"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/progressBar"** />  
  
 <**Button  
 android:id="@+id/button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="16dp"  
 android:layout\_marginLeft="28dp"  
 android:onClick="add"  
 android:text="add"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"** />  
  
 <**Button  
 android:id="@+id/button2"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="16dp"  
 android:layout\_marginStart="36dp"  
 android:onClick="download"  
 android:text="download"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintStart\_toEndOf="@+id/button"** />  
  
 <**Button  
 android:id="@+id/button3"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="16dp"  
 android:layout\_marginEnd="8dp"  
 android:layout\_marginStart="8dp"  
 android:layout\_marginTop="8dp"  
 android:onClick="showimages"  
 android:text="Button"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.461"  
 app:layout\_constraintStart\_toEndOf="@+id/button2"  
 app:layout\_constraintTop\_toBottomOf="@+id/editText\_url"  
 app:layout\_constraintVertical\_bias="1.0"** />  
  
 <**ProgressBar  
 android:id="@+id/progressBar"  
 style="?android:attr/progressBarStyleHorizontal"  
 android:layout\_width="385dp"  
 android:layout\_height="20dp"  
 android:layout\_marginEnd="16dp"  
 android:layout\_marginStart="16dp"  
 android:layout\_marginTop="8dp"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/textView"** />  
  
 <**TextView  
 android:id="@+id/textView"  
 android:layout\_width="394dp"  
 android:layout\_height="16dp"  
 android:layout\_marginBottom="154dp"  
 android:layout\_marginEnd="8dp"  
 android:layout\_marginLeft="16dp"  
 android:text="TextView"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/listView"** />  
  
 <**ListView  
 android:id="@+id/listView"  
 android:layout\_width="418dp"  
 android:layout\_height="0dp"  
 android:layout\_marginBottom="9dp"  
 android:layout\_marginTop="16dp"  
 app:layout\_constraintBottom\_toTopOf="@+id/textView"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintLeft\_toLeftOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"** />  
  
</**android.support.constraint.ConstraintLayout**>

***MainActivity.java***

**package** com.example.fairy.myapplication;  
  
**import** android.content.Intent;  
**import** android.os.AsyncTask;  
**import** android.os.Environment;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.ArrayAdapter;  
**import** android.widget.EditText;  
**import** android.widget.ListView;  
**import** android.widget.ProgressBar;  
**import** android.widget.TextView;  
  
**import** java.io.File;  
**import** java.io.FileOutputStream;  
**import** java.io.IOException;  
**import** java.io.InputStream;  
**import** java.net.MalformedURLException;  
**import** java.net.URL;  
**import** java.net.URLConnection;  
**import** java.util.ArrayList;  
  
*//访问网络，需要做权限声明（在AndroidManifest.xml中）***public class** MainActivity **extends** AppCompatActivity {  
  
 ListView **listView**; *//heap堆，可以被整个对象所有方法访问* EditText **editText\_url**;  
 TextView **textView**;  
 ProgressBar **progressBar**;  
  
 ArrayList<String> **arrayList** = **new** ArrayList<>();  
 ArrayList<String> **loacalImages** = **new** ArrayList<>();  
 ArrayAdapter<String> **arrayAdapter**;  
  
 MyAsyncTask **asyncTask**;*// 异步任务初始化* @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 **arrayList**.add(**"https://ss1.bdstatic.com/70cFuXSh\_Q1YnxGkpoWK1HF6hhy/it/u=2768994405,2861612898&fm=27&gp=0.jpg"**);  
 **arrayList**.add(**"https://ss0.bdstatic.com/70cFuHSh\_Q1YnxGkpoWK1HF6hhy/it/u=1324594007,2656156541&fm=27&gp=0.jpg"**);  
 **arrayList**.add(**"https://ss1.bdstatic.com/70cFuXSh\_Q1YnxGkpoWK1HF6hhy/it/u=3820748528,1139722339&fm=27&gp=0.jpg"**);  
 **arrayList**.add(**"https://ss2.bdstatic.com/70cFvnSh\_Q1YnxGkpoWK1HF6hhy/it/u=4292181273,3945252903&fm=27&gp=0.jpg"**);  
 **arrayList**.add(**"https://ss0.bdstatic.com/70cFuHSh\_Q1YnxGkpoWK1HF6hhy/it/u=1033621088,3281922354&fm=27&gp=0.jpg"**);  
 **arrayList**.add(**"https://ss1.bdstatic.com/70cFvXSh\_Q1YnxGkpoWK1HF6hhy/it/u=2884322920,434592651&fm=27&gp=0.jpg"**);  
  
  
 **listView** = findViewById(R.id.***listView***);  
 **editText\_url** = findViewById(R.id.***editText\_url***);  
 **textView** = findViewById(R.id.***textView***);  
 **progressBar** = findViewById(R.id.***progressBar***);  
  
 **arrayAdapter** = **new** ArrayAdapter<String>(**this**,android.R.layout.***simple\_list\_item\_1***,**arrayList**);  
 **listView**.setAdapter(**arrayAdapter**);  
  
 **asyncTask** = **new** MyAsyncTask();  
 }  
 **public void** add(View view){  
 **arrayList**.add(**editText\_url**.getText().toString());  
 **arrayAdapter**.notifyDataSetChanged();  
 }  
  
 **public void** download(View view){  
 *//下载过程中，多次点击download程序崩溃问题 改善方法* **switch** (**asyncTask**.getStatus()){  
 **case *PENDING***:*//准备好了，但是还没有执行* **asyncTask**.execute(0); *//启动执行异步任务* **break**;  
 **case *RUNNING***:*//* **break**;  
 **case *FINISHED***:*//* **asyncTask** = **new** MyAsyncTask();  
 **asyncTask**.execute(0);  
 **break**;  
 }  
  
 *//asyncTask.execute(0);//启动异步任务* }  
  
 *// 异步任务派生（暂不考虑内存泄漏）* **class** MyAsyncTask **extends** AsyncTask<Integer,Integer,Integer>{  
  
 @Override  
 **protected void** onPreExecute() {  
 **super**.onPreExecute();  
 **loacalImages**.clear();*//每次执行前清空* }  
  
 @Override  
 **protected** Integer doInBackground(Integer... integers) {  
 *//子线程（其他的方法在主线程）  
 //一个线程里面把文件按顺序下载（多个线程里面下载更快，缓冲区大也会让下载速度加快 ）* **for**(**int** i = 0; i<**arrayList**.size(); i++){  
 String s = **arrayList**.get(i);  
 *//转成URL（自动补全异常处理：万一转不成url）* **try** {  
 URL url = **new** URL(s);  
 *//openconnection:可以拿到文件大小  
 //openSteam:也可以下载图片，但这里要用进度条，不能拿到文件大小* URLConnection connection = url.openConnection();  
 **int** size = connection.getContentLength();*//拿到文件大小（进度条用）* publishProgress(0,i+1, size);*//通知主界面更新（让操作系统去帮助你间接调用）(里面是一个数组，可以传很多数进来)  
 //三个参数含义：开始下载，下载第几个文件，文件大小  
  
 //下载图片  
 //网络输入流* InputStream inputStream = connection.getInputStream();  
  
 *//本地文件输出流(外部私有存储)(不用做权限声明（公有的public才需要）)* File file = getExternalFilesDir(Environment.*DIRECTORY\_PICTURES*);*//等同 拿到 文件路径* String filename = file.getAbsolutePath()+**"/"**+i+**".jpg"**;*//路径／文件名* FileOutputStream fileOutputStream = **new** FileOutputStream(filename ); *//输出流  
  
 //图片是一个二进制流，我们要一个字节一个字节的读取（所以我们要准备一个缓冲区）* **byte**[] bytes = **new byte**[1000];  
 *//加while循环：图片字节数不定* **int** len;*// 文件读完之后返回 -1* **while**((len = inputStream.read(bytes))!= -1){  
 *//没读完  
 //读完的存入文件输出流* fileOutputStream.write(bytes,0,len);  
 *//参数含义；字节数组，偏移值（从0开始写), 长度  
  
 //Thread.sleep(10); // 休息10毫秒* publishProgress(1,len);*//通知进度条* }  
  
 *//关闭输入输出流* inputStream.close();  
 fileOutputStream.close();  
  
 **loacalImages**.add(filename);  
  
 } **catch** (MalformedURLException e) {  
 e.printStackTrace();  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 }  
*// catch (InterruptedException e) {  
// e.printStackTrace();  
// }* }  
 **return null**;  
 }  
  
 @Override  
 **protected void** onProgressUpdate(Integer... values) {  
 **super**.onProgressUpdate(values);  
 **switch** (values[0]){  
 **case** 0:  
 **textView**.setText(**"正在下载第"**+values[1]+**"个图片"**);  
 **progressBar**.setMax(values[2]);*//最大值为图片大小* **progressBar**.setProgress(0);*//清空* **break**;  
 **case** 1:  
 **progressBar**.incrementProgressBy(values[1]);*//增加长度（比setprocessbar更加直观）* **break**;  
 }  
 *//textView.setText("开始下载");* }  
  
 @Override  
 **protected void** onPostExecute(Integer integer) {  
 **super**.onPostExecute(integer);  
 **textView**.setText(**"下载完成！"**);  
 **progressBar**.setProgress(0 );  
  
  
*// //下载完成后自动显示  
// Intent intent = new Intent(MainActivity.this,Main2Activity.class);  
// //传参：有哪些文件 (第二个参数是文件数据(构造一个arraylist))  
// intent.putExtra("images",loacalImages);  
// startActivity( intent);* }  
 }  
  
 **public void** showimages(View view){  
 Intent intent = **new** Intent(**this**,Main2Activity.**class**);  
 *//传参：有哪些文件 (第二个参数是文件数据(构造一个arraylist))* intent.putExtra(**"images"**,**loacalImages**);  
 startActivity(intent);  
 }  
）

***activity\_main2.xml***

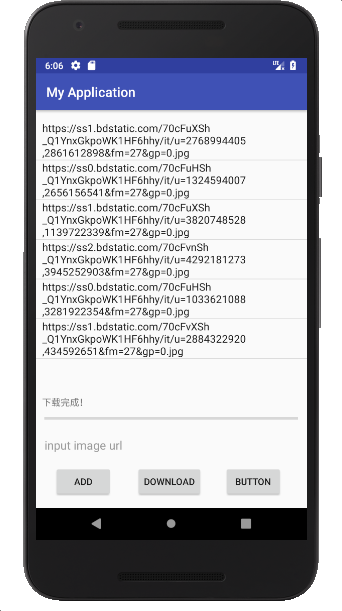
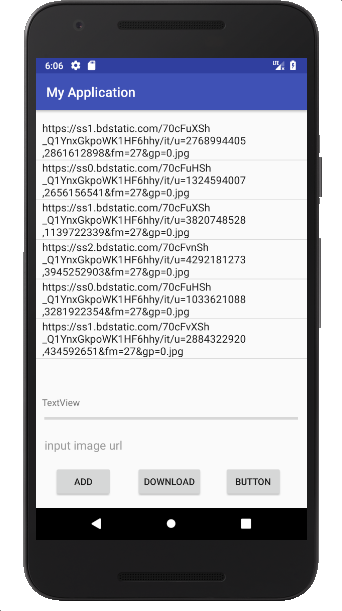
*<?***xml version="1.0" encoding="utf-8"***?>*<**android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context="com.example.fairy.myapplication.Main2Activity"**>  
  
 <**ScrollView  
 android:id="@+id/scrollView2"  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
 android:layout\_marginBottom="8dp"  
 android:layout\_marginEnd="8dp"  
 android:layout\_marginStart="8dp"  
 android:layout\_marginTop="8dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"**>  
  
 <**LinearLayout  
 android:id="@+id/layout"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"** />  
 </**ScrollView**>  
</**android.support.constraint.ConstraintLayout**>

***Main2Activity.java***

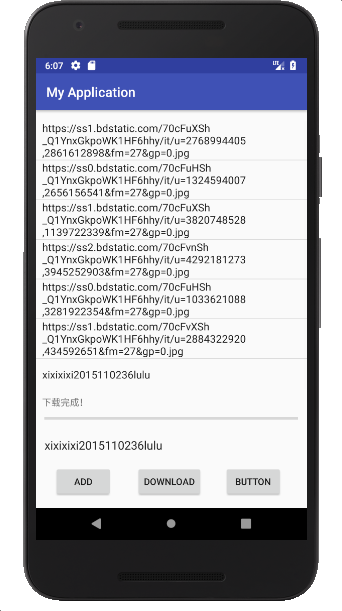
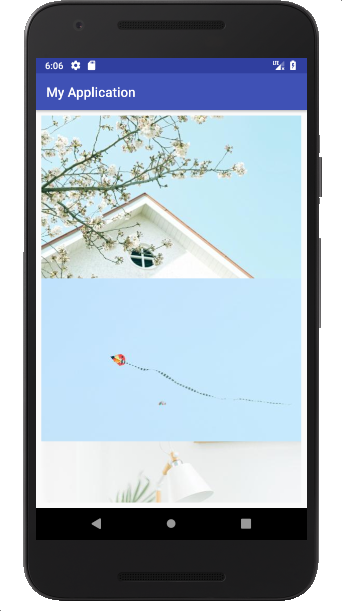
**package** com.example.fairy.myapplication;  
  
**import** android.content.Intent;  
**import** android.graphics.Bitmap;  
**import** android.graphics.BitmapFactory;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.widget.ImageView;  
**import** android.widget.LinearLayout;  
  
**import** java.util.ArrayList;  
  
*//显示图片***public class** Main2Activity **extends** AppCompatActivity {  
  
 LinearLayout **layout**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main2***);  
 Intent intent = getIntent();  
  
 **layout** = findViewById(R.id.***layout***);  
  
 ArrayList<String> localImages = (ArrayList<String>) intent.getSerializableExtra(**"images"**); *//取的时候，序列化 识别不了->强制转化  
  
 //读文件* **for**(**int** i = 0; i< localImages.size(); i++){  
 String imagePath = localImages.get(i);*//找到路径* Bitmap bitmap = BitmapFactory.*decodeFile*(imagePath);*//把拿到的文件恢复成一个图片  
  
 //在界面上显示图片* ImageView imageView = **new** ImageView(**this**);  
 imageView.setImageBitmap(bitmap);*//把转化得到的图片放到imageview（不用imagesource是因为这个图片不是资源文件，而是我们自己下载得到的文件 ）  
  
 //把图片放大成最优的效果（界面更加美观）* imageView.setScaleType(ImageView.ScaleType.***CENTER\_INSIDE***);  
 imageView.setAdjustViewBounds(**true**);*//自动调节大小* **layout**.addView(imageView);  
 }  
 }  
}

1. 实验结果的分析与评价（该部分如不够填写，请另加附页）

1.界面： 2.开始下载 3.下载完成



4.显示图片 5.添加



注：实验成绩等级分为（90－100分）优，（80－89分）良，(70-79分)中，（60－69分）及格，（59分）不及格。