

[首页](#) [ANDROID](#) [互联网](#) [杂乱无章](#) [科技资讯](#) [程序员人生](#) [程序员笑话](#) [编程技术](#) [网址导航](#)

## 7.3.2 Android 文件下载 (1)

分类 **Android 基础入门教程**

**Android 基础入门教程(Q群号 : 153836263)**

### 本节引言：



又是一个深坑，初学者慎入...本节将从普通的单线程下载 -> 普通多线程下载 -> -> 以及一个很实用的例子：利用Android那只DownloadManager更新apk 并覆盖安装的实现代码！好的，这样看上去，本节还是蛮有趣的，开始本节内容！PS:我们把整个完整的多线程断点续传放到下一节中！

### 1.普通单线程下载文件：

直接使用URLConnection.openStream()打开网络输入流,然后将流写入到文件中！

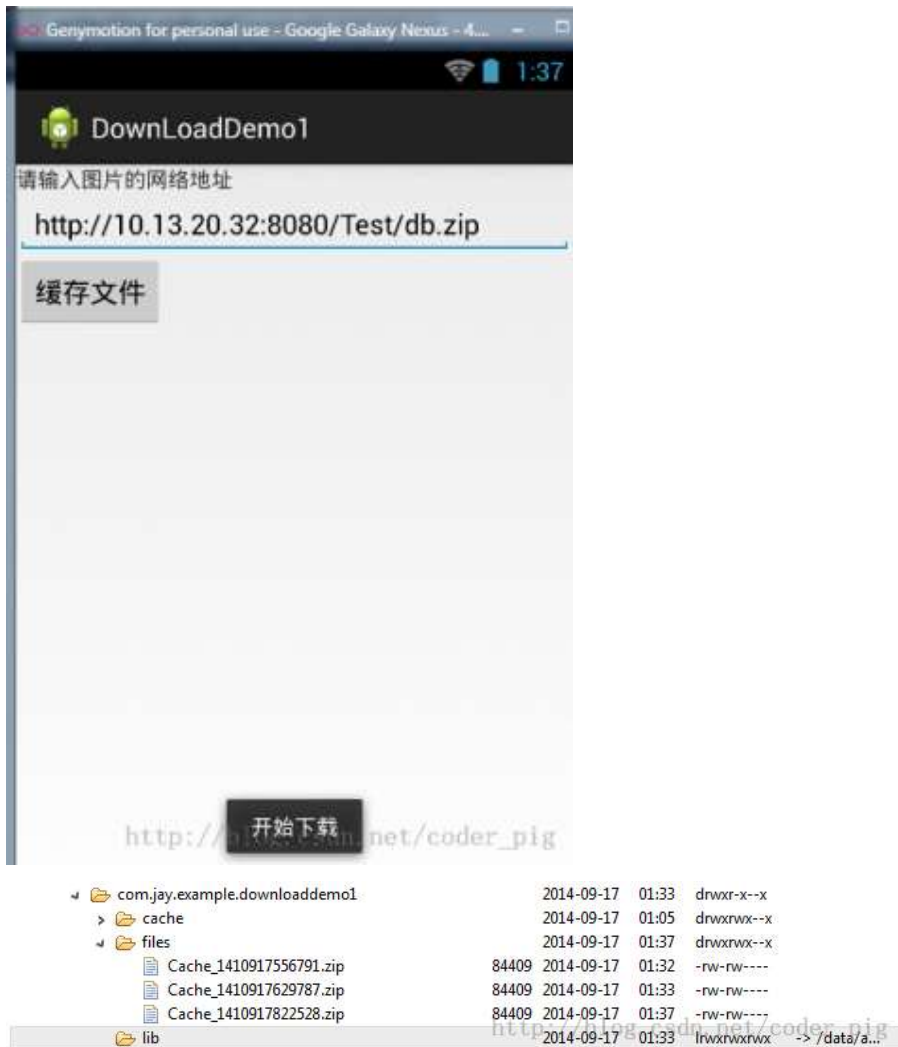
#### 核心方法：

```
public static void download(String path, Context context) throws Exception
{
    URL url = new URL(path);
    InputStream is = url.openStream();
    //截取最后的文件名
    String end = path.substring(path.lastIndexOf("."));
    //打开手机对应的输出流,输出到文件中
    OutputStream os = context.openFileOutput("Cache_"+System.currentTimeMillis()+end, Context.MODE_PRIVATE);
    byte[] buffer = new byte[1024];
    int len = 0;
    //从输入六中读取数据,读到缓冲区中
    while((len = is.read(buffer)) > 0)
    {
        os.write(buffer, 0, len);
    }
}
```

- 1.0 Android基础入门教程
- 1.0.1 2015年最新Android基...
- 1.1 背景相关与系统架构分析
- 1.2 开发环境搭建
- 1.2.1 使用Eclipse + ADT + S...
- 1.2.2 使用Android Studio开...
- 1.3 SDK更新不了问题解决
- 1.4 Genymotion模拟器安装
- 1.5.1 Git使用教程之本地仓...
- 1.5.2 Git之使用GitHub搭建...
- 1.6 .9(九妹)图片怎么玩
- 1.7 界面原型设计
- 1.8 工程相关解析(各种文件...
- 1.9 Android程序签名打包
- 1.11 反编译APK获取代码&...
- 2.1 View与ViewGroup的概念
- 2.2.1 LinearLayout(线性布局)
- 2.2.2 RelativeLayout(相对布...
- 2.2.3 TableLayout(表格布局)
- 2.2.4 FrameLayout(帧布局)
- 2.2.5 GridLayout(网格布局)
- 2.2.6 AbsoluteLayout(绝对...
- 2.3.1 TextView(文本框)详解
- 2.3.2 EditText(输入框)详解
- 2.3.3 Button(按钮)与ImageB...
- 2.3.4 ImageView(图像视图)
- 2.3.5.RadioButton(单选按钮...
- 2.3.6 开关按钮ToggleButton...
- 2.3.7 ProgressBar(进度条)
- 2.3.8 SeekBar(拖动条)
- 2.3.9 RatingBar(星级评分条)
- 2.4.1 ScrollView(滚动条)

```
//关闭输入输出流
is.close();
os.close();
}
```

运行结果：



## 2. 普通多线程下载：

我们都知道使用多线程下载文件可以更快地完成文件的下载,但是为什么呢?

答：因为抢占的服务器资源多,假设服务器最多服务100个用户,服务器中的一个线程 对应一个用户100条线程在计算机中并发执行,由CPU划分时间片轮流执行,加入a有99条线程 下载文件,那么相当于占用了99个用户资源,自然就有用较快的下载速度

**PS:**当然不是线程越多就越好,开启过多线程的话,app需要维护和同步每条线程的开销, 这些开销反而会导致下载速度的降低,另外还和你的网速有关!

**多线程下载的流程：**

获取网络连接

本地磁盘创建相同大小的空文件

计算每条线程需从文件哪个部分开始下载，结束

2.4.2 Date & Time组件(上)

2.4.3 Date & Time组件(下)

2.4.4 Adapter基础讲解

2.4.5 ListView简单实用

2.4.6 BaseAdapter优化

2.4.7 ListView的焦点问题

2.4.8 ListView之checkbox错...

2.4.9 ListView的数据更新问题

2.5.0 构建一个可复用的自定...

2.5.1 ListView Item多布局的...

2.5.2 GridView(网格视图)的...

2.5.3 Spinner(列表选项框)...

2.5.4 AutoCompleteTextVie...

2.5.5 ExpandableListView(...

2.5.6 ViewPager(翻转视图)...

2.5.7 Toast(吐司)的基本使用

2.5.8 Notification(状态栏通...

2.5.9 AlertDialog(对话框)详解

2.6.0 其他几种常用对话框基...

2.6.1 PopupWindow(悬浮框...

2.6.2 菜单(Menu)

2.6.3 ViewPager的简单使用

2.6.4 DrawerLayout(官方侧...

3.1.1 基于监听的事件处理机制

3.2 基于回调的事件处理机制

3.3 Handler消息传递机制浅析

3.4 TouchListener PK OnTo...

3.5 监听EditText的内容变化

3.6 响应系统设置的事件(Co...

3.7 AsyncTask异步任务

3.8 Gestures(手势)

4.1.1 Activity初学乍练

4.1.2 Activity初窥门径

4.1.3 Activity登堂入室

4.2.1 Service初涉

4.2.2 Service进阶

4.2.3 Service精通

4.3.1 BroadcastReceiver牛...

4.3.2 BroadcastReceiver庖...

4.4.1 ContentProvider初探

依次创建，启动多条线程来下载网络资源的指定部分

#### 普通多线程下载的流程

①根据要访问的URL路径去调用openConnection(),得到HttpConnection对象,接着用getContentLength();获得下载的文件长度,最后设置本地文件的长度

ps:RandomAccessFile随机访问类,同时整合了FileOutputStream和FileInputStream,支持从文件的任何字节处读写数据,而File只支持将文件作为整体来处理,不能读写文件

```
int filesize = HttpURLConnection.getContentLength();
RandomAccessFile file = new RandomAccessFile("xxx.apk", "rwd");
file.setLength(filesize);
```

②根据文件长度以及线程数计算每条线程的下载长度,以及每条线程下载的起始位置

计算公式:加入N条线程下载大小为M个字节的文件:  
每个线程下载的数据量:  $M \% N == 0 ? M/N : M/N + 1$   
比如:大小为10个字节的,开3条线程下载,那么每个线程的下载量为  $10/3 + 1 = 4$ ,即三条线程的下载量分别为:4,4,2

下载起始位置的计算:假设线程id分别为0,1,2;  
那么开始位置 =  $id * \text{下载量}$   
结束位置 =  $(id + 1) * \text{下载量} - 1$   
(最后一条线程不用计算结束位置的!)

③保存文件,使用RandomAccessFile类指定从文件的什么位置写入数据

```
RandomAccessFile threadFile = new
RandomAccessFile("xxx.apk", "rwd");
threadFile.seek(2097152);
```

另外,在下载时,怎么指定每条线程开始下载的位置呢?

答:HTTP协议为我们提供了一个Range头,使用下面方法指定从什么位置开始下载:  
`HttpURLConnection.setRequestProperty("Range", "bytes=2097152-4194303");`

PS:这里直接创建一个Java项目，然后在JUnit里运行指定方法即可，

核心代码如下：

```
public class Downloader {
    //添加@Test标记是表示该方法是JUnit测试的方法,就可以直接运行该方法了
    @Test
    public void download() throws Exception
    {
        //设置URL的地址和下载后的文件名
        String filename = "meitu.exe";
        String path = "http://10.13.20.32:8080/Test/XiuXiu_Green.exe";
        URL url = new URL(path);
        HttpURLConnection conn = (HttpURLConnection) url.openConnection();
        conn.setConnectTimeout(5000);
        conn.setRequestMethod("GET");
        //获得需要下载的文件长度(大小)
        int filelength = conn.getContentLength();
        System.out.println("要下载的文件长度"+filelength);
        //生成一个大小相同的本地文件
        RandomAccessFile file = new RandomAccessFile(filename, "rwd");
        file.setLength(filelength);
        file.close();
        conn.disconnect();
        //设置有多少条线程下载
        int threadsize = 3;
        //计算每个线程下载的量
        int threadlength = filelength % 3 == 0 ? filelength/3:filelength+1;
        for(int i = 0;i < threadsize;i++)
        {
            //设置每条线程从哪个位置开始下载
            int startposition = i * threadlength;
            //从文件的什么位置开始写入数据
            RandomAccessFile threadfile = new RandomAccessFile(filename, "rwd");
            threadfile.seek(startposition);
            //开始下载
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            //下载
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            //结束
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            threadfile.close();
        }
    }
}
```

4.4.2 ContentProvider再探...

4.5.1 Intent的基本使用

4.5.2 Intent之复杂数据的传递

5.1 Fragment基本概述

5.2.1 Fragment实例精讲一...

5.2.2 Fragment实例精讲一...

5.2.3 Fragment实例精讲一...

5.2.4 Fragment实例精讲一...

5.2.5 Fragment实例精讲一...

6.1 数据存储与访问之——文...

6.2 数据存储与访问之——S...

6.3.1 数据存储与访问之——...

6.3.2 数据存储与访问之——...

7.1.1 Android网络编程要学...

7.1.2 Android Http请求头与...

7.1.3 Android HTTP请求方...

7.1.4 Android HTTP请求方...

7.2.1 Android XML数据解析

7.2.2 Android JSON数据解析

7.3.1 Android 文件上传

7.3.2 Android 文件下载 (1)

7.3.3 Android 文件下载 (2)

7.4 Android 调用 Webservice

7.5.1 WebView(网页视图)基...

7.5.2 WebView和JavaScrip...

7.5.3 Android 4.4后WebVie...

7.5.4 WebView文件下载

7.5.5 WebView缓存问题

7.5.6 WebView处理网页返...

7.6.1 Socket学习网络基础准备

7.6.2 基于TCP协议的Socket...

7.6.3 基于TCP协议的Socket...

7.6.4 基于UDP协议的Socke...

8.1.1 Android中的13种Draw...

8.1.2 Android中的13种Draw...

8.1.3 Android中的13种Draw...

8.2.1 Bitmap(位图)全解析 P...

8.2.2 Bitmap引起的OOM问题

8.3.1 三个绘图工具类详解

8.3.2 绘图类实战示例

```

        threadfile.seek(startposition);
        //启动三条线程分别从startposition位置开始下载文件
        new DownloadThread(i, startposition, threadfile, threadlength, path).start();
    }
    int quit = System.in.read();
    while('q' != quit)
    {
        Thread.sleep(2000);
    }
}

private class DownloadThread extends Thread {
    private int threadid;
    private int startposition;
    private RandomAccessFile threadfile;
    private int threadlength;
    private String path;
    public DownloadThread(int threadid, int startposition,
        RandomAccessFile threadfile, int threadlength, String path) {
        this.threadid = threadid;
        this.startposition = startposition;
        this.threadfile = threadfile;
        this.threadlength = threadlength;
        this.path = path;
    }
    public DownloadThread() {}
    @Override
    public void run() {
        try
        {
            URL url = new URL(path);
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            conn.setConnectTimeout(5000);
            conn.setRequestMethod("GET");
            //指定从什么位置开始下载
            conn.setRequestProperty("Range", "bytes="+startposition+"-");
            //System.out.println(conn.getResponseCode());
            if(conn.getResponseCode() == 206)
            {
                InputStream is = conn.getInputStream();
                byte[] buffer = new byte[1024];
                int len = -1;
                int length = 0;
                while(length < threadlength && (len = is.read(buffer)) != -1)
                {
                    threadfile.write(buffer, 0, len);
                    //计算累计下载的长度
                    length += len;
                }
            }
        }
    }
}

```



反馈

8.3.3 Paint API之—— Mask...

8.3.4 Paint API之—— Xferm...

8.3.5 Paint API之—— Xferm...

8.3.6 Paint API之—— Xferm...

8.3.7 Paint API之—— Xferm...

8.3.8 Paint API之—— Xferm...

8.3.9 Paint API之—— Color...

8.3.10 Paint API之—— Colo...

8.3.11 Paint API之—— Colo...

8.3.12 Paint API之—— Path...

8.3.13 Paint API之—— Sha...

8.3.14 Paint几个枚举/常量值...

8.3.15 Paint API之——Type...

8.3.16 Canvas API详解(Part 1)

8.3.17 Canvas API详解(Part...

8.3.18 Canvas API详解(Part...

8.4.1 Android动画合集之帧...

8.4.2 Android动画合集之补...

8.4.3 Android动画合集之属...

8.4.4 Android动画合集之属...

9.1 使用SoundPool播放音...

9.2 MediaPlayer播放音频与...

9.3 使用Camera拍照

9.4 使用MediaRecord录音

10.1 TelephonyManager(电...

10.2 SmsManager(短信管理...

10.3 AudioManager(音频管...

10.4 Vibrator(振动器)

10.5 AlarmManager(闹钟服务)

10.6 PowerManager(电源服...

10.7 WindowManager(窗口...

10.8 LayoutInflater(布局服务)

10.9 WallpaperManager(壁...

10.10 传感器专题(1)——相...

10.11 传感器专题(2)——方...

10.12 传感器专题(3)——加...

10.12 传感器专题(4)——其...

10.14 Android GPS初涉

11.0 《2015最新Android基...

```

    }
    threadfile.close();
    is.close();
    System.out.println("线程"+(threadid+1) + "已下载完成");
}
} catch (Exception ex) {System.out.println("线程"+(threadid+
1) + "下载出错"+ ex);}
}

}
}

```

### 运行截图：

如图,使用多线程完成了对文件的下载!双击exe文件可运行,说明文件并没有损坏!



### 注意事项：

`int filelength = conn.getContentLength();` //获得下载文件的长度(大小)

`RandomAccessFile file = new RandomAccessFile(filename, "rwd");` //该类运行对文件进行读写,是多线程下载的核心

`int threadlength = filelength % 3 == 0 ? filelength/3:filelength+1;` //计算每个线程要下载的量

`conn.setRequestProperty("Range", "bytes="+startposition+"-");` //指定从哪个位置开始读写,这个是URLConnection提供的方法

`//System.out.println(conn.getResponseCode());` //这个注释了的代码是用来查看conn的返回码的,我们前面用的都是200,而针对多线程的话,通常是206,必要时我们可以通过调用该方法查看返回码!

`int quit = System.in.read();while('q' != quit){Thread.sleep(2000);}` //这段代码是做延时操作的,因为我们用的是本地下载,可能该方法运行完了,而我们的线程还没有开启,这样会引发异常,这里的话,让用户输入一个字符,如果是'q'的话就退出



### 3.使用DownloadManager更新应用并覆盖安装：

下面的代码可以直接用，加入到项目后，记得为这个内部广播注册一个过滤器：

#### AndroidManifest.xml

:

```
import android.app.DownloadManager;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.pm.ApplicationInfo;
import android.content.pm.PackageManager;
import android.database.Cursor;
import android.net.Uri;
import android.os.Bundle;
import android.os.Environment;
import android.support.v7.app.AppCompatActivity;

/**
 * Created by Jay on 2015/9/9 0009.
 */
public class UpdateAct extends AppCompatActivity {
    //这个更新的APK的版本部分，我们是这样命名的:xxx_v1.0.0_xxxxxxxxxx
    .apk
    //这里我们用的是git提交版本的前九位作为表示
    private static final String FILE_NAME = "ABCDEFGHI";

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        String endpoint = "";
        try {
            //这部分是获取AndroidManifest.xml里的配置信息的，包名，
            以及Meta_data里保存的东西
            ApplicationInfo info = getPackageManager().getApplicationInfo(
                getPackageName(), PackageManager.GET_META_D
                ATA);
            //我们在meta_data保存了xxx.xxx这样一个数据，是https开头
            的一个链接，这里替换成http
            endpoint = info.metaData.getString("xxxx.xxxx").rep
            lace("https",
                "http");
        } catch (PackageManager.NameNotFoundException e) {
            e.printStackTrace();
        }
        //下面的都是拼接apk更新下载url的，path是保存的文件夹路径
        final String _Path = this.getIntent().getStringExtra("p
        ath");
        final String _Url = endpoint + _Path;
        final DownloadManager _DownloadManager = (DownloadManag
```

```

er) getSystemService(DOWNLOAD_SERVICE);

        DownloadManager.Request _Request = new DownloadManager.
Request(

            Uri.parse(_Url));
        _Request.setDestinationInExternalPublicDir(
            Environment.DIRECTORY_DOWNLOADS, FILE_NAME + ".
apk");
        _Request.setTitle(this.getString(R.string.app_name));
        //是否显示下载对话框
        _Request.setShowRunningNotification(true);
        _Request.setMimeType("application/com.trinea.download.f
ile");
        //将下载请求放入队列
        _DownloadManager.enqueue(_Request);
        this.finish();
    }

    //注册一个广播接收器, 当下载完毕后会收到一个android.intent.action
.DOWNLOAD_COMPLETE
    //的广播, 在这里取出队列里下载任务, 进行安装
    public static class Receiver extends BroadcastReceiver {
        public void onReceive(Context context, Intent intent) {
            final DownloadManager _DownloadManager = (DownloadM
anager) context
                .getSystemService(Context.DOWNLOAD_SERVICE)
;

            final long _DownloadId = intent.getLongExtra(
                DownloadManager.EXTRA_DOWNLOAD_ID, 0);
            final DownloadManager.Query _Query = new DownloadMa
nager.Query();
            _Query.setFilterById(_DownloadId);
            final Cursor _Cursor = _DownloadManager.query(_Quer
y);

            if (_Cursor.moveToFirst()) {
                final int _Status = _Cursor.getInt(_Cursor
                    .getColumnIndexOrThrow(DownloadManager.
COLUMN_STATUS));
                final String _Name = _Cursor.getString(_Cursor
                    .getColumnIndexOrThrow("local_filename"
));
                if (_Status == DownloadManager.STATUS_SUCCESSFU
L
                    && _Name.indexOf(FILE_NAME) != 0) {

                    Intent _Intent = new Intent(Intent.ACTION_V
IEW);

                    _Intent.setDataAndType(
                        Uri.parse(_Cursor.getString(_Cursor
                            .getColumnIndexOrThrow(Down
loadManager.COLUMN_LOCAL_URI))),
                        "application/vnd.android.package-ar

```

```
chive");

        _Intent.addFlags(Intent.FLAG_ACTIVITY_NEW_T
ASK);

        context.startActivity(_Intent);
    }
}

    _Cursor.close();
}
}
}
```

4.参考代码下载：

普通单线程下载文件：[DownloadDemo1.zip](#) 普通多线程下载文件：[J2SEMulDownloader.zip](#)

本节小结：

好的，本节给大家介绍了普通单线程以及多线程下载文件，还有利用Android自带DownManager来 下载更新APK，然后覆盖的实现！相信会对大家的实际开发带来便利，好的，就说这么多，谢谢~

大家都在玩手机而我却用手机学英语

邮箱订阅《每日英语》

每天坚持五分钟, 随时随地学英语!

立即行动



	<div>在线实例</div> <ul style="list-style-type: none"><li>· <a href="#">HTML 实例</a></li><li>· <a href="#">CSS 实例</a></li><li>· <a href="#">JavaScript 实例</a></li><li>· <a href="#">Ajax 实例</a></li><li>· <a href="#">jQuery 实例</a></li><li>· <a href="#">XML 实例</a></li><li>· <a href="#">Java 实例</a></li></ul>	<div>字符集&amp;工具</div> <ul style="list-style-type: none"><li>· <a href="#">HTML 字符集设置</a></li><li>· <a href="#">HTML ASCII 字符集</a></li><li>· <a href="#">HTML ISO-8859-1</a></li><li>· <a href="#">HTML 实体符号</a></li><li>· <a href="#">HTML 拾色器</a></li><li>· <a href="#">JSON 格式化工具</a></li></ul>	<div>最新更新</div> <ul style="list-style-type: none"><li>· <a href="#">JavaScript 查找...</a></li><li>· <a href="#">JavaScript 判断...</a></li><li>· <a href="#">设置 SSH 通过密...</a></li><li>· <a href="#">CSS all 属性</a></li><li>· <a href="#">Px、Em 换算工具</a></li><li>· <a href="#">px,pt,em换算表</a></li><li>· <a href="#">px、em、rem 区别...</a></li></ul>	<div>站点信息</div> <ul style="list-style-type: none"><li>· <a href="#">意见反馈</a></li><li>· <a href="#">免责声明</a></li><li>· <a href="#">关于我们</a></li><li>· <a href="#">文章归档</a></li></ul>
				<div>关注微信</div> <div></div> <div>Copyright © 2013-2015 菜鸟教程 <a href="#">runoob.com</a> All Rights Reserved. 备案号：闽ICP备15012807号-1</div>



