ISSN 1009-3044
Computer Knowledge and Technology 电脑知识与技术
Vol.9, No.18, June 2013.
E-mail: xsjl@dnzs.net.cn
http://www.dnzs.net.cn
Tel:+86-551-65690963 65690964

基于Android的多线程处理技术

杨 杰

(中国地质大学 地球物理与信息技术学院,北京)

摘要: 在Android程序的开发过程中,程序运行的流畅性是十分重要的。如果主线程处理的事件耗时过长将会出现ANR(应用程序无响应),导致程序崩溃。这就有必要将耗时较多的事件交给后台线程处理,从而来提升用户体验,改善应用程序性能。该文就Android中多线程技术的使用进行讲解。

关键词: Android;多线程;Handler;AsyncTask

中图分类号:TN929 文献标识码:A 文章编号:1009-3044(2013)18-4251-04

Android-based Multi-threaded Processing Technology

YANG Jie

(Institute of Geophysics and Information Technology, China University of Geosciences, Beijing 100083, China)

Abstract: Android application development process, the program runs smoothly is very important. If the main thread processing the event took too long there will be ANR (Application not responding) that crashes the program. This will take more event to a background thread processing to enhance the user experience, improve application performance. This article is to explain the use of the Android multi-threading technology.

Key words: Android; Multi-threaded; Handler; AsyncTask

Android是Google公司历经数年和投资数亿美元开发处出来的的智能手机操作系统,经过多年的发展,Android系统市场份额不断增大,目前已跃居世界第一。基于Android平台的各类人才逐渐成为各大企业争抢的对象。对于Android应用程序的开发有着很大的市场前景。要开发一款好的应用程序,多线程技术的使用是必不可少的。多线程技术可以很好地提升程序的执行效率,提升用户体验。因而很有必要掌握多线程技术。

1 Android系统组成

Android的系统架构和其他操作系统一样,采用了分层的架构。从架构图看(如图1),android分为四个层,从高层到低层分别是应用程序层(Applications)、应用程序框架层(Application Framework)、系统运行库层(Libraries、Android Runtime)和linux核心层(Linux Kernel)。

1)应用程序层:Android应用程序层中,所有的应用程序都是使用JAVA语言编写的。我们可以编写自己的核心应用程序。

2)应用程序框架:在android中,应用程序所使用的API,开发者可以完全访问。android应用程序的架构设计使得所有开发组件可以非常方便的复用和修改。

3)函数库层(Libraries):Android包含了一些C/C++库,Android系统中的不同的组件都可以使用这些类库。这些类库通过Android应用程序框架为开发者提供开发服务,进行应用程序的开发。

4)Android运行时库层(Android Runtime):Android使用了自有的Android Runtime来执行。

5) linux核心层(Linux Kernel):Android系统用的是为2.6的Linux内核版本,主要包含的功能有内存管理(Memory Management)机制、安全控制(Security)机制、进程管理(Process Managemen)机制、硬件驱动(Driver Model)机制、网络协议栈(Network Stack)机制等,同时Linux内核也作为硬件和软件栈之间的抽象层(HAL)。

2 Android开发基本组件

1)活动(Activity)是Android中最基本的应用程序组件,在应用程序中,通常一个活动就是一个单独的屏幕。每一个活动都是从活动基类中继承而来,Activity类将会显示由几个Views控件组成的用户接口,并对用户事件做出响应。

2)服务(Service)是一段长生命周期的,没有用户界面的程序,在后台一直运行。

3)内容提供者(Content Provider):内容提供者是用来管理和共享应用数据类库,它可以实现多个程序之间的数据共享。

收稿日期:2013-05-09
项目基金:中国地质大学(北京)大学生创新创业训练计划项目(目编号为2012AX0157)

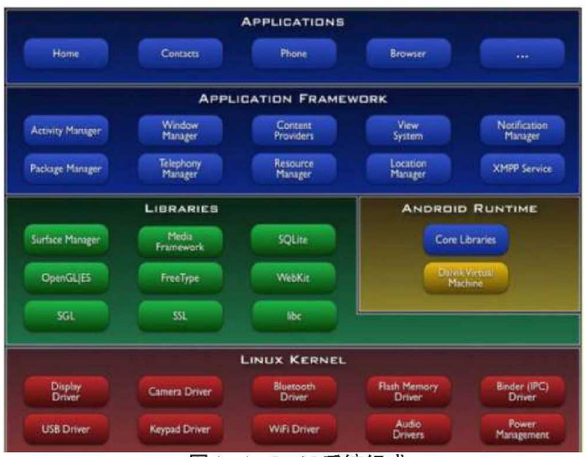

图1 Android 系统组成

4)Intent(意图):Intent在Android起着一个中介的作用,专门用于提供应用组件相互调用时的相关信息,可以实现调用者与被调用者之间的解耦合。

5)广播接收器(Broadcast receiver):可以使用它对外部事件进行过滤,使得应用程序只处理感兴趣的外部事件。

3 Android中多线程技术的使用

在android应用中,UI线程(主线程)超过5秒没响应的话就会抛出无响应异常(ANR)。在通过网络获取下载的大资源文件时,如果处理不当,是很容易出现异常问题的。Android开发框架中提供两种方法来处理这种问题:

1)首先,启动一个新的线程来获取下载资源文件,资源获取结束后通过Handler机制发送消息(Message),并同时在UI线程中处理消息,从而达到在异步线程中处理事件(event)的效果,然后通过Handler Message方法来更新UI线程;(实现过程见图2)

2)使用Android中提供的AsyncTask方式来完成异步操作。AsyncTask是使用java.util.concurrent框架来管理线程以及任务的执行的,concurrent框架是一个非常成熟,高效的框架,经过了严格的测试。这说明AsyncTask的设计很好的解决了匿名线程存在的问题。

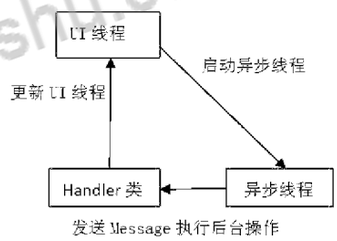

图2 异步线程处理

4 多线程技术具体实现

4.1 使用Handler实现多线程

我们都知道,在Android中主线程是线程不安全的,也就是说,更新UI界面只能在应用程序的主线程(UI)中进行更新,在子线程中进行UI更新操作是十分危险的(及容易造成线程崩溃)。android为了解决这样的问题,让我们就使用Handler机制来进行处理。由于Handler运行在主线程中(UI线程中),它与子线程可以通过Message对象来传递数据进行通信,这个时候,Handler就承担着接受子线程传过来的(子线程用sedMessage()方法)传递Message对象,(里面包含数据),然后,把这些消息放入主线程队列中,进而配合主线程进行UI的更新操作。

实例代码如下:

```
//主线程
public class HandlerDemo extends Activity {
private String TAG="HandlerTESTActivity";
//当Handler被创建会关联到创建它的当前线程的消息队列。
private Handler mHandler = new Handler(){
//接收子线程发送的消息
public void handleMessage(Message msg) {
```

```
switch (msg.what) {
case 2:
Log.i(TAG ,"测试使用");
break;
}
};
};
Public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.main);
}
//开启新线程执行
Thread thread = new Thread()
{
public void run()
{
Log.i(TAG, "start Thread");
Message message = new Message();
message.what = 2;
mHandler.sendMessage(message);
//发送消息给Handler
};
};
}
```

Handler可以说是线程和Activity交互的桥梁,我们只要在run方法中发送Message,而在Handler里,通过不同的Message执行不同的任务。从而可以很好地降低异步线程和主线程之间的耦合度。

4.2 使用AsyncTask类实现多线程

为了要使用AsyncTask,首先需要定义下面的几个方法

1)onPreExecute()

该方法将在执行后台操作前被UI线程调用。我们可以在该方法中做一些事先准备工作,当然,这个方法也可以不用实现。

2)doInBackground(Params...)

在onPreExecute方法执行后马上执行,该方法运行在后台线程中。将负责执行那些很耗时的后台处理工作。该方法是抽象方法,根据java语言的规则,该方法必须实现。

3)onProgressUpdate(Progress...)

在publishProgress方法被调用后,UI线程将调用这个方法从而在界面上展示任务的进展情况,

4) onPostExecute(Result)

在doInBackground执行完成后,onPostExecute方法将被UI线程回调,后台的计算结果将通过该方法传递到UI线程中,并在界面上进行显示。

5) onCancelled()

应用程序取消线程操作的时候调用。

实例代码如下:(用于更新进度条)

```
public class ProgressTask extends Activity {
ProgressBar pb;//进度条
TextView tv;  //文本显示
Button dButton;//下载按钮
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.test);        tv=(TextView)findViewById(R.id.tv);
pb=(ProgressBar)findViewById(R.id.pb);
dButton = (Button)findViewById(R.id.download);  //给下载按钮设置监听器
download.setOnClickListener(new View.OnClickListener() {
public void onClick(View v) {
```

```
DownloadTask dTask = new DownloadTask();
dTask.execute(100);  //运行AnyncTask实例
}
});
}
class DownloadTask extends AsyncTask<Integer, Integer, String>{
protected void onPreExecute() {
//第一个执行方法
super.onPreExecute();
}
//开启异步线程
protected String doInBackground(Integer... params) {
//第二个执行方法
for(int i=0;i<=100;i++){
pb.setProgress(i); //设置进度条
publishProgress(i);//更新进度条
try {
Thread.sleep(params[0]);
} catch (InterruptedException e) {
e.printStackTrace();
} }
return "测试异步线程结束"; }
protected  void onProgressUpdate(Integer... progress) {
//这个函数在doInBackground调用publishProgress时触发。
//进行文本显示
tv.setText(progress[0]+"%");            super.onProgressUpdate(progress);
}
protected void onPostExecute(String result) {
//doInBackground返回时触发,换句话说,就是doInBackground执行完后触发
setTitle(result);
super.onPostExecute(result);
} } }
```

实现效果见图3。

图3 异步下载实验图

AsyncTask是Android为开发者提供的用于处理一些比较耗时的后台任务的基类,通过使用AsyncTask可以优化应用程序,可以给用户带来良好用户体验的。

5 结束语

本文对Android系统的整体架构进行了简单的介绍,对Android中的开发组件进行了大致的分析,提出了使用多线程技术来避免应用出现ANR(应用程序无响应)的方法,通过Handler和AsyncTask机制把耗时较多的任务放在子线程中执行,来保持UI主线程的流畅顺滑,从而获得良好的用户体验。我们很难想象没有使用的多线程技术的应用程序,多线程技术的使用对于提高应用程序性能和用户体验是十分重要的。具有一定的研究价值。

参考文献:

[1] Dave MacLean.精通Android3[M].杨越,译.北京:人民邮电出版社,2011:255-260,324-328.
[2] 余志龙,陈小凤.AndroidSDK开发范例大[M].北京:清华大学出版社,2010.
[3] 闫伟,叶建榜.多线程技术在android手机开发中的应用[J].信息通信,2012(1):46-47.