# **Android 多线程：手把手教你使用AsyncTask**

# **前言**

1. 多线程的应用在Android开发中是非常常见的，常用方法主要有：

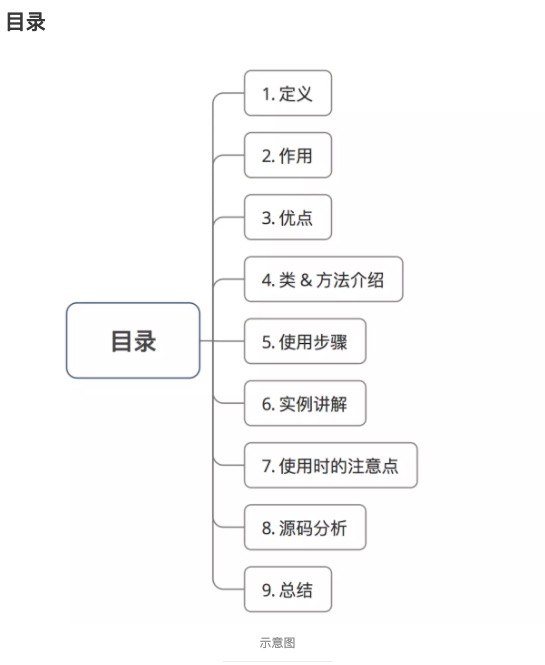
2. 继承Thread类

3. 实现Runnable接口

4. Handler

5. AsyncTask

6. HandlerThread



# 1. 定义

* 一个Android 已封装好的轻量级异步类
* 属于抽象类，即使用时需 实现子类

public abstract class AsyncTask<Params, Progress, Result> {

...

}

# 2.作用

1. 实现所线程，在工作线程中执行任务，如耗时任务。

2. 异步通信，消息传递，实现主线程（UI线程）&工作线程之间的通信，即将工作线程的执行结果传递给主线程，从而在主线程中执行相应的UI操作。保证线程安全。

3. 优点

1.方便实现异步通信  
 不需使用 “任务线程（如继承Thread类） + Handler”的复杂组合

节省资源。  
 2. 采用线程池的缓存线程 + 复用线程，避免了频繁创建 & 销毁线程所带来的系统资源开销。

4.类&方法介绍

4.1 定义

AsyncTask类属于抽象类，即使用时需 实现子类。

public abstract class AsyncTask<Params, Progress, Result> {

... }

// 类中参数为3种泛型类型// 整体作用：控制AsyncTask子类执行线程任务时各个阶段的返回类型// 具体说明：

// a. Params：开始异步任务执行时传入的参数类型，对应excute（）中传递的参数

// b. Progress：异步任务执行过程中，返回下载进度值的类型

// c. Result：异步任务执行完成后，返回的结果类型，与doInBackground()的返回值类型保持一致// 注：

// a. 使用时并不是所有类型都被使用

// b. 若无被使用，可用java.lang.Void类型代替

// c. 若有不同业务，需额外再写1个AsyncTask的子类}

4.2 核心方法

AsyncTask 核心 & 常用的方法如下：





# 5. 使用步骤

AsyncTask的使用步骤有3个：

1. 创建 AsyncTask 子类 & 根据需求实现核心方法

2. 创建 AsyncTask子类的实例对象（即 任务实例）

3. 手动调用execute(（）从而执行异步线程任务

具体介绍如下

/\*\*

\* 步骤1：创建AsyncTask子类

\* 注：

\* a. 继承AsyncTask类

\* b. 为3个泛型参数指定类型；若不使用，可用java.lang.Void类型代替

\* c. 根据需求，在AsyncTask子类内实现核心方法

\*/

private class MyTask extends AsyncTask<Params, Progress, Result> {

....

// 方法1：onPreExecute（）

// 作用：执行 线程任务前的操作

// 注：根据需求复写

@Override

protected void onPreExecute() {

...

}

// 方法2：doInBackground（）

// 作用：接收输入参数、执行任务中的耗时操作、返回 线程任务执行的结果

// 注：必须复写，从而自定义线程任务

@Override

protected String doInBackground(String... params) {

...// 自定义的线程任务

// 可调用publishProgress（）显示进度, 之后将执行onProgressUpdate（）

publishProgress(count);

}

// 方法3：onProgressUpdate（）

// 作用：在主线程 显示线程任务执行的进度

// 注：根据需求复写

@Override

protected void onProgressUpdate(Integer... progresses) {

...

}

// 方法4：onPostExecute（）

// 作用：接收线程任务执行结果、将执行结果显示到UI组件

// 注：必须复写，从而自定义UI操作

@Override

protected void onPostExecute(String result) {

...// UI操作

}

// 方法5：onCancelled()

// 作用：将异步任务设置为：取消状态

@Override

protected void onCancelled() {

...

}

}

/\*\*

\* 步骤2：创建AsyncTask子类的实例对象（即 任务实例）

\* 注：AsyncTask子类的实例必须在UI线程中创建

\*/

MyTask mTask = new MyTask();

/\*\*

\* 步骤3：手动调用execute(Params... params) 从而执行异步线程任务

\* 注：

\* a. 必须在UI线程中调用

\* b. 同一个AsyncTask实例对象只能执行1次，若执行第2次将会抛出异常

\* c. 执行任务中，系统会自动调用AsyncTask的一系列方法：onPreExecute() 、doInBackground()、onProgressUpdate() 、onPostExecute()

\* d. 不能手动调用上述方法

\*/

mTask.execute()；