https://www.jianshu.com/p/669ee2aec526

https://www.jianshu.com/p/263fe326d0bc

定义：

一个基本的CPU执行单元 & 程序执行流的最小单元。

特点：

- 比进程更小的可独立运行的基本单位，可理解为：轻量级进程；
- 组成：线程ID + 程序计数器 + 寄存器集合 + 堆栈；
- 线程自己不拥有系统资源，与其他线程共享进程所拥有的全部资源。

作用：

减少程序在并发执行时所付出的时空开销，提高操作系统的并发性能。

分类：

1.守护线程

守护用户线程的线程，即在程序运行时为其他线程提供一种通用服务，如垃圾回收线程。设置该线程为守护线程的方式如下：

```
thread.setDaemon(true);
```

2.非守护线程（用户线程）

主要包括：主线程 & 子线程。

a. 主线程（UI线程）

定义：Android系统在程序启动时会自动启动一条主线程
作用：处理四大组件与用户进行交互的事情（如UI、界面交互相关）
注：因为用户随时会与界面发生交互，因此主线程任何时候都必须保持很高的响应速度，所以主线程不允许进行耗时操作，否则会出现ANR

b. 子线程（工作线程）

手动创建的线程，主要用于耗时的操作（网络请求、I/O操作等）

3.守护线程与非守护线程的区别：

1.当所有用户线程结束时，因为没有守护的必要，所以守护线程也会终止，虚拟机也同样退出；
2.反过来，只要任何用户线程还在运行，守护线程就不会终止，虚拟机就不会退出

> 补充：子线程能不能更新UI?
> 不能，不过也的看情况，在界面刚刚开启的时候，子线程更新UI这个机制还没来得及开启，这个监测机制还没上班

点击查看详情：https://www.jianshu.com/p/7a8cb20cfd80

一. 创建子线程

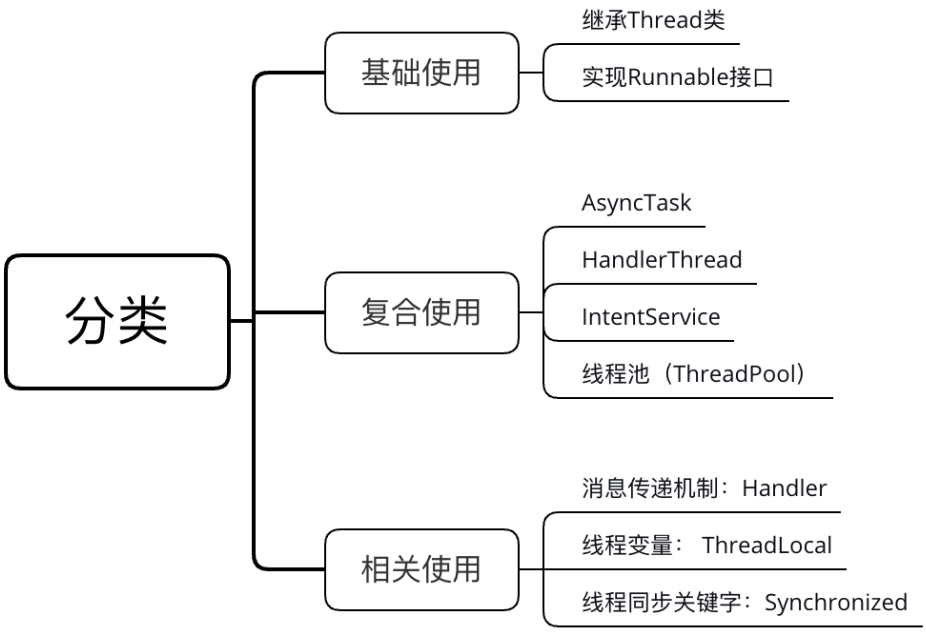
```
//1.创建 Thread的匿名内部类的形式（重写run方法）
new Thread(){
    public void run() {

        };
}.start();


//2.实现Runnalbe接口，重载Runnalbe接口中的run()方法
public class MyThread implements Runnable{
    public void run(){
    }
}
```

二. 更新UI的方法

```
//1.直接在子线程更新UI（runOnUiThread），逻辑在Runnable的run方法中(要重写run方法)，内部封装了Handler发消息的机制（内部会做出判断）
runOnUiThread(new Runnable(){
    @Override
    public void run() {

        Looper.prepare();
        Toast.makeText(ctx, "在子线程中弹吐司", 0).show();
        Looper.loop();
    }}
);


//2.消息机制的写法，创建一个成员变量 Handler ，  并复写方法 handleMessage(Message msg)
private Handler handler=new Handler(){
    @Override
    handleMessage(Message msg){

        //得到发送过来的消息（图片：Bitmap_强转）,然后直接显示更新UI
        Bitmap bitmap =  (Bitmap) msg.obj;
        iv_imasetImage(bitmap );
    }
}

//当子线程有更新UI需求时，创建一个message对象,把要更新的数据设置给msg,用handler.sendMessage()发送
Message msg=Message.obtain();
msg.obj=bitmap;
handler.sendMessage(msg);

// 2秒之后再出来
handler.postDelayed(new Runnable() {
    @Override
    public void run() {

    }
}, 2000);
```

点击查看详情：Android多线程继承Thread类 使用解析

点击查看详情：Android多线程实现Runnable接口 使用解析



点击查看详情：关于线程的总结归档篇