1. NSRunLoop的实现机制,及在多线程中如何使用

NSRunLoop是IOS消息机制的处理模式

>1.NSRunLoop的主要作用：控制NSRunLoop里面线程的执行和休眠，在有事情做的时候使当前NSRunLoop控制的线程工作，没有事情做让当前NSRunLoop的控制的线程休眠。

>2.NSRunLoop 就是一直在循环检测，从线程start到线程end，检测inputsource(如点击，双击等操作)同步事件，检测timesource同步事件，检测到输入源会执行处理函数，首先会产生通知，corefunction向线程添加runloop observers来监听事件，意在监听事件发生时来做处理。

>3.runloopmode是一个集合，包括监听：事件源，定时器，以及需通知的runloop observers

>1. 只有在为你的程序创建次线程的时候，才需要运行run loop。对于程序的主线程而言，run loop是关键部分。Cocoa提供了运行主线程run loop的代码同时也会自动运行run loop。IOS程序UIApplication中的run方法在程序正常启动的时候就会启动run loop。如果你使用xcode提供的模板创建的程序，那你永远不需要自己去启动run loop

>2. 在多线程中，你需要判断是否需要run loop。如果需要run loop，那么你要负责配置run loop并启动。你不需要在任何情况下都去启动run loop。比如，你使用线程去处理一个预先定义好的耗时极长的任务时，你就可以毋需启动run loop。Run loop只在你要和线程有交互时才需要

2．IOS7之前,后台执行内容有几种形式,都是什么

一般的应用在进入后台的时候可以获取一定时间来运行相关任务，也就是说可以在后台运行一小段时间(10S左右)。

1. 后台播放音乐

2. 后台GPS跟踪

3. 后台voip支持

3简单说一下APP的启动过程,从main文件开始说起

程序启动分为两类:1.有storyboard 2.没有storyboard

有storyboard情况下:

1.main函数

2.UIApplicationMain

\* 创建UIApplication对象

\* 创建UIApplication的delegate对象

3.根据Info.plist获得最主要storyboard的文件名,加载最主要的storyboard(有storyboard)

\* 创建UIWindow

\* 创建和设置UIWindow的rootViewController

\* 显示窗口

没有storyboard情况下:

1.main函数

2.UIApplicationMain

\* 创建UIApplication对象

\* 创建UIApplication的delegate对象

3.delegate对象开始处理(监听)系统事件(没有storyboard)

\* 程序启动完毕的时候, 就会调用代理的application:didFinishLaunchingWithOptions:方法

\* 在application:didFinishLaunchingWithOptions:中创建UIWindow

\* 创建和设置UIWindow的rootViewController

\* 显示窗口

4把程序自己关掉和程序进入后台,远程推送的区别

1. 关掉后不执行任何代码,不能处理事件

2. 应用程序进入后台状态不久后转入挂起状态。在这种状态下，应用程序不执行任何代码，并有可能在任意时候从内存中删除。只有当用户再次运行此应用，应用才会从挂起状态唤醒，代码得以继续执行.或者进入后台时开启多任务状态，保留在内存中，这样就可以执行系统允许的动作

3. 远程推送是由远程服务器上的程序发送到APNS,再由APNS把消息推送至设备上的程序,当应用程序收到推送的消息会自动调用特定的方法执行事先写好的代码

5本地通知和远程推送通知对基本概念和用法？

本地通知和远程推送通知都可以向不在前台运行的应用发送消息,这种消息既可能是即将发生的事件,也可能是服务器的新数据.不管是本地通知还是远程通知,他们在程序界面的显示效果相同,都可能显示为一段警告信息或应用程序图标上的微章.

本地通知和远程推送通知的基本目的都是让应用程序能够通知用户某些事情, 而且不需要应用程序在前台运行.二者的区别在于本地通知由本应用负责调用,只能从当前设备上的iOS发出, 而远程通知由远程服务器上的程序发送到APNS,再由APNS把消息推送至设备上的程序



