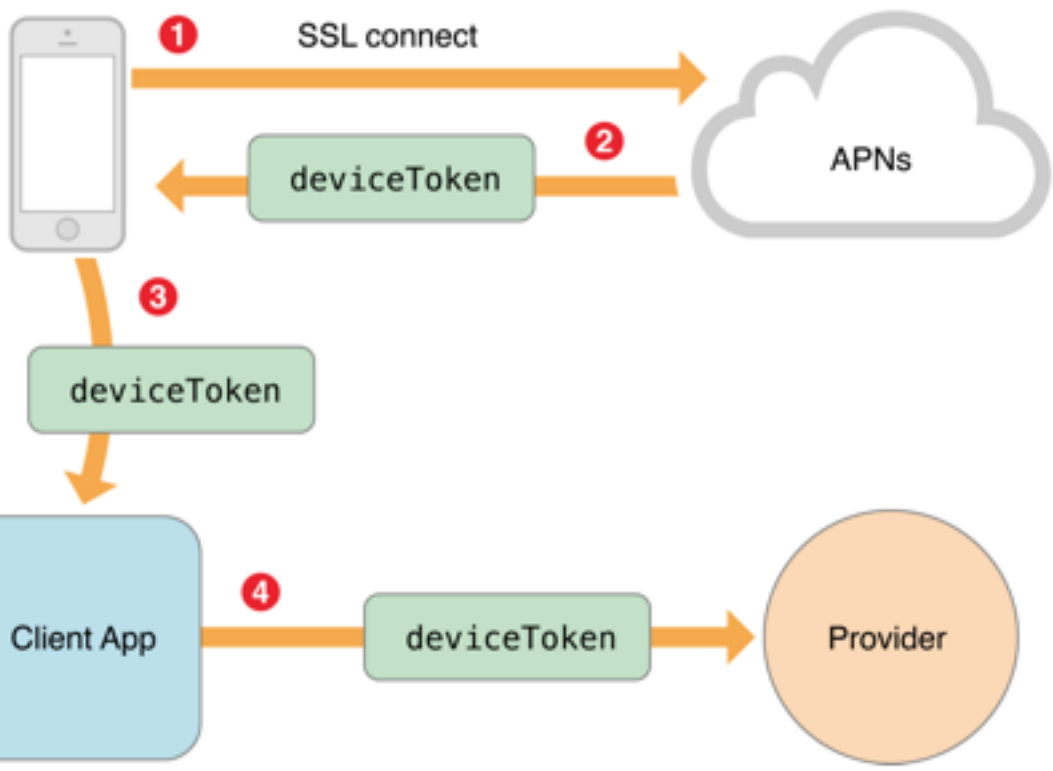
iOS远程推送处理

本文主要讲解iOS收到远程消息后客户端的一些处理方法，iOS 10开始苹果单独集成一套框架专门处理通知，可谓非常方便。至于如何集成推送，比较简单，这里不做探讨，本文主要是让大家了解收到远程消息后应用程序若要做相应的操作以及点击通知需要做的操作，该如何去处理。

先带大家大致了解一下推送的流程，大致三步：

1. 应用程序在苹果推送服务器APNS上注册deviceToken，APNS返回devicetoken给应用程序，之后发送给自己的后台服务器；
2. 后台服务器将deviceToken和要发送的消息一起打包发送给APNS；
3. APNS将消息发送给deviceToken中保存的指定设备中的指定APP



APNS流程图

好了，现在开始讲重点了。。。

首先应用程序接收到远程通知时可能处于三种状态：① 程序未启动，退出状态；② 程序处于后台，挂起状态；③ 程序处于前台，运行状态。

而程序启动也可有两种方法：①点击通知；②点击应用图标。

本文着重对点击通知进行深入探讨，至于点击应用图标进入app，受制于

apple，开发者并不能做什么，故忽略。

收到消息和点击推送消息可能会调用的5个方法如下：

```
①- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOpt
```

此方法是当应用程序处于退出状态后，点击应用图标或者点击通知栏启动程序会调用，若是点击了通知栏启动，则推送信息userInfo会保存在launchOptions这里

```
if (launchOptions) {
    NSDictionary *userInfo = [launchOptions objectForKey:UIApplicationLaunc
    self.userInfo = userInfo;
}
```

很多时候我们点击通知栏启动app需要跳转到相应的页面(有时通知栏的消息类型不一样，所需跳转的页面也不一样)，我们可以先将userInfo保存起来(self.userInfo = userInfo)，待app初始化完之后，再在main控制器里的viewDidAppear里进行跳转，跳转完之后再把userInfo清空，否则每次进入此控制器都会跳转，或者对它设置只执行一次的操作。但是点击通知栏除了会调用此方法，还会调用方法②(iOS 7.0)或者④(iOS 10.0)，若在这个方法里做了操作还需进行另外的处理，请看下面。

```
② - (void)application:(UIApplication *)application didReceiveRemoteNotifica
```

此方法为iOS 7.0之后，若实现了此方法，则下面的方法③就会被覆盖，不被调用。

收到通知，若处于前台则会调用，若处于后台并不会调用，若要想在后台收到通知调用(app想在后台收到通知之后做一些刷新UI的操作)，则需后台服务器配上content-available = 1这个字段，同时在Xcode里Capabilities中Background Modes里面勾选了Remote notifications为YES。点击通知栏则一定会调用，若想点击之后做一些跳转页面的操作，还需加一些判断条件:

```
if (application.applicationState == UIApplicationStateActive) {
```

```
} else if (application.applicationState == UIApplicationStateInactive) {
    // 在这里进行跳转操作

} else if (application.applicationState == UIApplicationStateBackground) {

}
```

为什么要加判断？上面讲了，处于前台和后台收到通知会调用此方法，不加判断处于前台和后台的话，就会自动跳转咯。并且还要判断①方法中是否有保存userInfo，若有说明是启动app同时也会调用此方法但不需在这里执行跳转(因为你前面做了跳转处理了，不然要跳两次)。

总结：①app处于前台，收到通知会调用；②app处于后台，收到通知，若配置了以上所讲则会调用，否则不调用；③点击通知栏，无论app处于后台、前台、退出都会调用

③ - (void)application:(UIApplication *)application didReceiveRemoteNotifica

此方法，只有当app处于前台收到或者点击通知才会调用。若有实现方法②，则此方法作废；若未实现，点击通知的处理与方法②一样，就不过多赘述了。

④ - (void)userNotificationCenter:(UNUserNotificationCenter *)center willPre

此方法为iOS 10独享的，将本地推送和远程推送都统一在这个方法里。只有当app处于前台时收到通知(本地和远程)会调用此方法。有点需要注意，若想在前台时收到通知不展示出来，则需在此方法中这样处理：

```
completionHandler(UNNotificationPresentationOptionBadge|UNNotificationPres
completionHandler(UNNotificationPresentationOptionBadge|UNNotificationPres
```

⑤ - (void)userNotificationCenter:(UNUserNotificationCenter *)center didRece

此方法为iOS 10独享的，将本地推送和远程推送都统一在这个方法里。当app处于后台时收到通知(本地和远程)会调用此方法，对于远程推送也需后台服务器配置content-available = 1这个字段且Remote notifications为

YES，否则处于后台收到推送不调用，点击通知则一定会调用，交互处理方法同②。

若为iOS 10，但并未实现④和⑤，则还是iOS 7 8 9之前一样,调用相应的方法。

总结（最低为iOS 7.0）：

1. app处于退出状态，所有方法都不调用，只有点击应用图标才会调用方法①，点击通知栏iOS 10 调用①和④，非iOS 10调用①和②；
2. app处于后台，若有content-available = 1和Remote notifications为YES，iOS 10 调用⑤，非iOS 10调用②，否则不调用；点击通知栏又会调用一次，iOS 10 调用⑤，非iOS 10调用②；
3. app处于前台，iOS 10调用④，非iOS 10调用②；点击通知栏又会调用一次，iOS 10 调用④，非iOS 10调用②；

以上为后台远程推送，需要后台必须配置字段badge、alert、sound，若有静默推送的需求，则必须加上content-available = 1，且必须去掉字段badge、alert、sound

以上是工作中总结出来的，希望能帮助到大家，如有不正确之出，欢迎指出。

另外，本人在处理推送时遇到了一个比较大的坑，在推送时，APNS会返回说一个无效的token，导致推送失败，配合后台花了大半天才解决，大致原因如下：

1. 在推送时，遇到失效的TOKEN导致消息推送失败。查阅很多资料，都说一旦遇到一个失效的TOKEN，同一个队列中，从失效的TOKEN往后的消息都无法推送
2. 即便通过APNS提供的方法，定时获取失效TOKEN进行删除，但是由于有一定的延迟，从失效TOKEN开始，往后一定时间内推送的消息，还是无法正常的推送到用户手机上
3. 如果一旦遇到失效TOKEN，苹果推送服务器，是否会主动断开连接。
4. 有人提供的方法是：每发送一定数量的消息，就检查一下是否有失效TOKEN，如果有就删除失效TOKEN，从新获取新的连接。就像上面

说的，因为有一定的延迟(延迟多长时间还不清楚)，即便通过这种方式，好像也没办法保证大批量消息丢失啊。