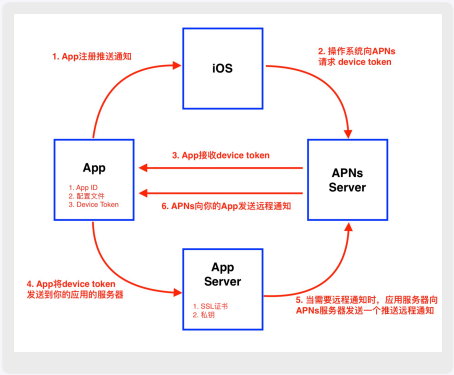
iOS远程推送

远程推送原理

- app 注册推送通知
- iOS 操作系统向 APNs 请求 device token
- app 接收 device token
- app 将 device token 发送到应用的服务器
- 当需要远程推送时，应用服务器向 APNs 服务器发送一个推送远程通知
- APNs 向 app 发送远程通知



网络状态

- 设备联网时会与苹果的APNs服务器建立一个长连接
  - 苹果的 APNs 服务器通过与设备建立的长连接，进而把通知推送到我们的设备
- 设备未联网时，APNs 服务器会保留 Provider 所推送的最后一条通知
  - 设备长时间处于非联网状态下，那么 APNs 服务器为其保存的最后一条通知也会丢失

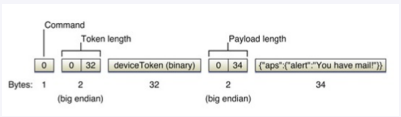
deviceToken

- 是什么
  - 注册远程通知的时候向 APNs 服务器发送 Token key，Token key 中包含了设备的 UDID 和 App 的 Bundle Identifier，然后苹果 APNs 服务器根据此 Token key 编码生成一个 deviceToken
    - APNs 服务器会留有 UDID+Bundle Identifier+deviceToken 的映射表
- 为什么
  - 推送消息的时候必须带有此 deviceToken，然后根据 deviceToken（UDID + App's Bundle Identifier）找到对应的设备以及该设备上对应的应用，从而把此推送消息推送给此应用
- 唯一性
  - 苹果 APNs 的编码技术和 deviceToken 的独特作用保证了它的唯一性。当用户升级系统时 deviceToken 是会变化的。

远程推送负载大小

根据 Provider 使用的 API 不同而不同

- 使用 HTTP/2 provider API 时，负载最大为 4096bytes，即 4kB
- 使用 legacy binary interface 时，负载最大为 2048bytes，即 2kB



远程推送负载内容

- aps
  - "content-available" :1
    - 后台接收通知
  - "mutable-content":1
    - 使推送通知是动态可变的，结合 UNNotificationServiceExtension 使用
- 自定义 key 与 aps 并列

指定用户的推送

userToken

- userToken 一般都是根据自己公司自定义的规则去生成的

第三方SDK

- 大同小异
- 在基本原理上面进行了扩展
  - JPush 提供了应用内消息推送
- 方便服务端开发
  - 服务端开发者不需要去开发、维护自己的推送服务器与 APNs 对接，不必自己维护更新 deviceToken

利用 runtime 实现收到推送消息后进行页面跳转

Notification Extension

- UNNotificationServiceExtension 通知服务扩展
- UNNotificationContentExtension 通知内容扩展
  - 自定义 UI