iOS 10 通知更新详解

时间 2016-09-20 22:59:26 ZLtunes' Connect

原文 http://zltunes.com/ios-10-tong-zhi-geng-xin-xiang-jie/

主题 iOS开发

是「通知」而不是「推送」

关于「通知」iOS 10 新增了一个框架 UserNotifications.framework ，即“用户通知框架”，推送 "Push" 只是「通知」触发的一种方式，而「通知」是操作系统层面的一种UI展示。苹果官方文档中 Notification 分为两类： - Remote (远程，即 Push 方式) - Local (本地，通知由本地事件触发，iOS 10 中有三种不同的触发 "Trigger" 方式，下文有详细说明)

所以，「推送」只是「通知」的一种触发方式，从 iOS 迭代更新的历史特征中看，「通知」一直是被苹果作为重点内容来延展的。iOS 10 中新增了独立框架（之前一直存在于 UIKit Framework 中）还有丰富的特性更新。

更新概览

原文

1. Familiar API with feature parity
2. Expanded content
3. Same code path for local and remote notification handling
4. Simplified delegate methods

5. Better notification management
6. In-app presentation option
7. Schedule and handle notifications in extensions
8. Notification Extensions

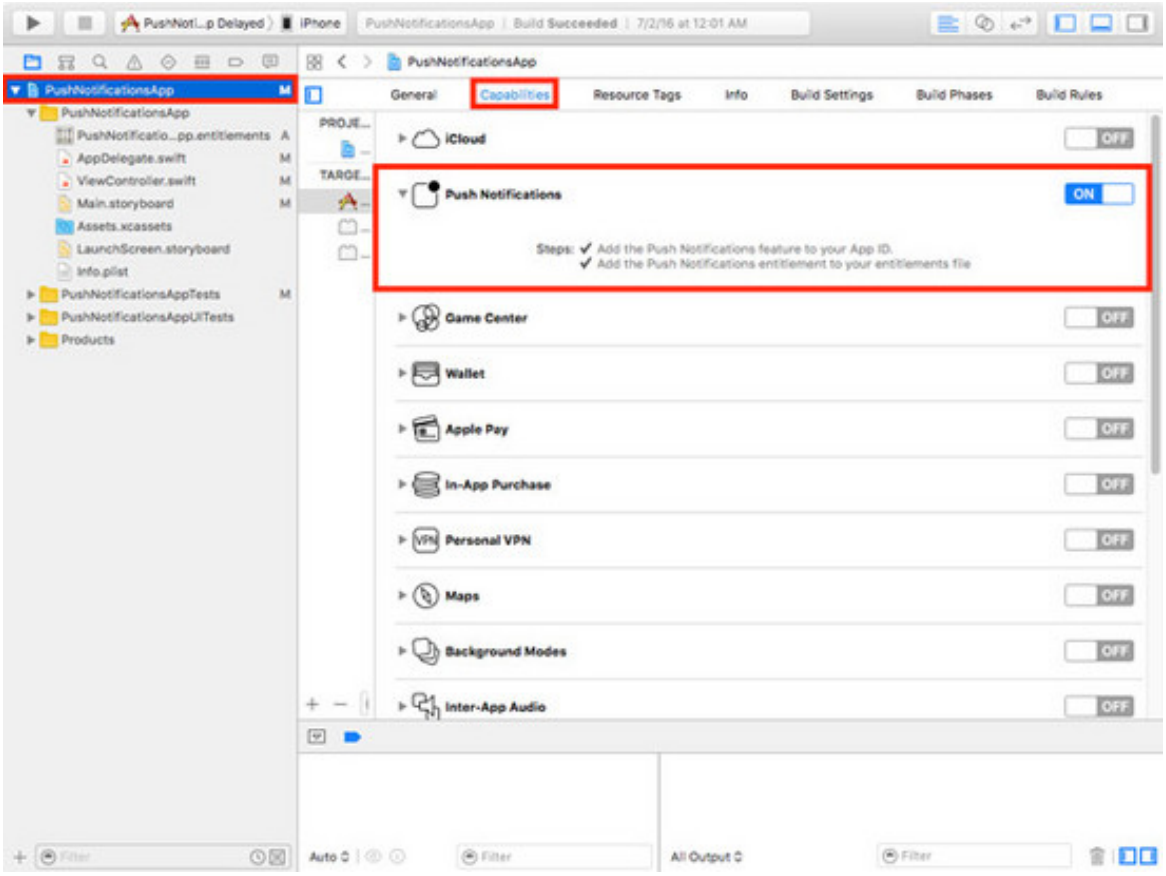
解析

1. 相同的特性使用类似API（之前的功能API使用方法类似但是还是稍有改变）
2. 内容扩展（支持附件和展示更多内容）
3. 本地通知和远程通知操作代码在相同调用路径（合并代理方法）
4. 简化代理方法
5. 更好的通知管理（支持通知查、改、删；增强本地通知管理，增加日历与地理位置事件的触发）
6. 应用内通知展示（之前App在前台的情况下收到通知不会UI展示）
7. 在Extensions中规划和操作通知（使更新通知内容和删除误发或过期的通知内容成为可能，另一个重要场景为端到端加密）
8. 引入通知Extensions

用 UserNotifications Framework 实现通知

1. 在 Xcode 中启用推送通知

要使用 UserNotifications Framework 需在 Xcode 项目中开启推送通知： Project Target --> Capabilities --> Push Notifications



2. import

```
#import <UserNotifications/UserNotifications.h>
```

3. 注册推送

```
- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)l

    // iOS 10 before
    UIUserNotificationSettings *settings = [UIUserNotificationSettings settingsForTypes:UIUserNc
    [application registerUserNotificationSettings:settings];

    // iOS 10
    UNUserNotificationCenter *center = [UNUserNotificationCenter currentNotificationCenter];
    [center requestAuthorizationWithOptions:(UNAuthorizationOptionBadge | UNAuthorizationOptionS
        if (!error) {
            NSLog(@"request authorization succeeded!");
        }
    }];

    return YES;
}
```

4. Token Registration

跟之前一样:

```
[[UIApplication sharedApplication] registerForRemoteNotifications];
```

5. Notification Settings

之前注册推送服务，首次安装 APP 后弹出授权推送通知框，用户点击了同意还是不同意，以及用户之后又做了怎样的更改我们是无从得知的，现在 apple 开放了这个 API，我们可以直接获取到用户的设定信息了。

```
[center getNotificationSettingsWithCompletionHandler:^(UNNotificationSettings * _Nonnull setting
        NSLog(@"%@",settings);
}];
```

打印 settings 如下：

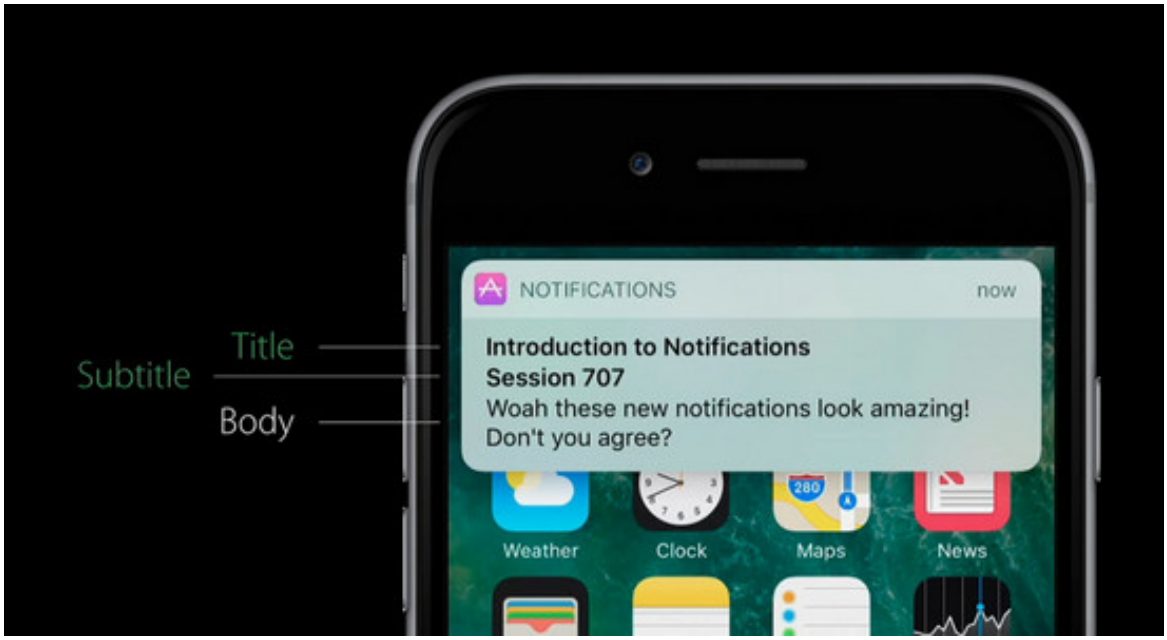
```
<UNNotificationSettings: 0x16567310;
authorizationStatus: Authorized,
notificationCenterSetting: Enabled,
soundSetting: Enabled,
badgeSetting: Enabled,
lockScreenSetting: Enabled,
alertSetting: NotSupported,
carPlaySetting: Enabled,
alertStyle: Banner>
```

settings.authorizationStatus 有三个值：

```
typedef NS_ENUM(NSInteger, UNAuthorizationStatus) {
    // 用户还没决定是否允许开启推送通知.
    UNAuthorizationStatusNotDetermined = 0,

    // 不允许开启通知.
    UNAuthorizationStatusDenied,

    // 允许开启.
    UNAuthorizationStatusAuthorized
} __IOS_AVAILABLE(10.0) __TVOS_AVAILABLE(10.0) __WATCHOS_AVAILABLE(3.0);
```

6. Content

以前只能展示一条文字，现在可以有 title 、subtitle 以及 body.

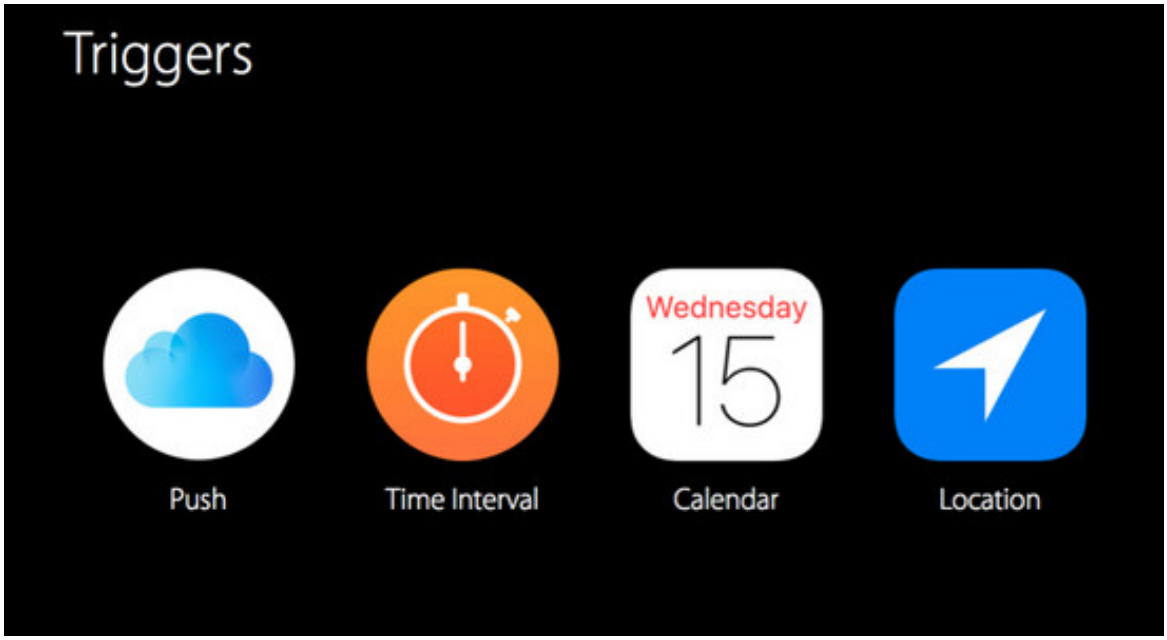


定制方法:

```
//Local Notification
UNMutableNotificationContent *content = [[UNMutableNotificationContent alloc] init];
content.title = @"Introduction to Notifications";
content.subtitle = @"Session 707";
content.body = @"Woah! These new notifications look amazing! Don't you agree?";
content.badge = @1;

//Remote Notification
{
"aps" : {
    "alert" : {
         "title" : "Introduction to Notifications",
         "subtitle" : "Session 707",
         "body" : "Woah! These new notifications look amazing! Don't you agree?"
                },
    "badge" : 1
        },
}
```

7. Triggers

本地通知新增了两种新的 Triggers —— 日历和地理位置。日历 Triggers 让开发者可以根据指定的日期和时间来展示本地通知，并且支持循环条件，如“每周二上午十一点”这种条件。地理位置 Triggers 可以在进入或者离开指定区域时触发本地通知，如“某品牌App在你进入该品牌线下店铺的范围内即展示最新优惠信息”等。



Triggers 有三种值:

- UNTimeIntervalNotificationTrigger - UNCalendarNotificationTrigger - UNLocationNotificationTrigger

```
//2 分钟后提醒
UNTimeIntervalNotificationTrigger *trigger1 = [UNTimeIntervalNotificationTrigger triggerWithTime

//每小时重复 1 次（循环）
UNTimeIntervalNotificationTrigger *trigger2 = [UNTimeIntervalNotificationTrigger triggerWithTime

//每周一早上 8：00 提醒我
NSDateComponents *components = [[NSDateComponents alloc] init];
components.weekday = 2;
components.hour = 8;
UNCalendarNotificationTrigger *trigger3 = [UNCalendarNotificationTrigger triggerWithDateMatching

//#import <CoreLocation/CoreLocation.h>
//一到麦当劳就喊我下车
CLRegion *region = [[CLRegion alloc] init];
UNLocationNotificationTrigger *trigger4 = [UNLocationNotificationTrigger triggerWithRegion:regio
```

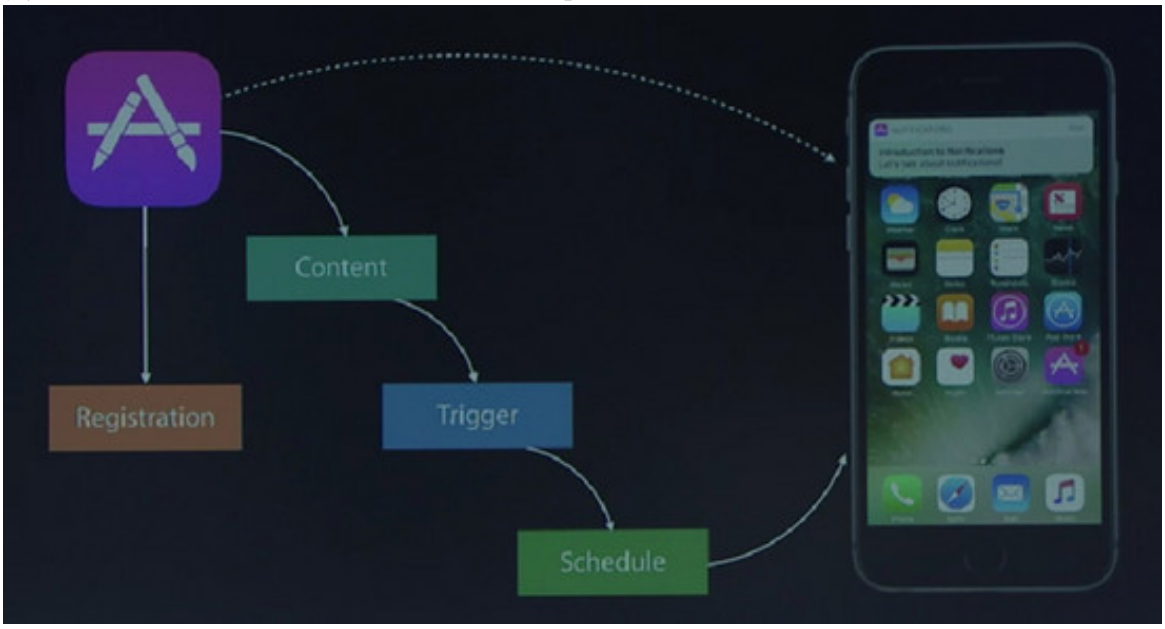
8. Add Request

```
NSString *requestIdentifier = @"sampleRequest";
UNNotificationRequest *request = [UNNotificationRequest requestWithIdentifier:requestIdentifier
                                                                       content:content
                                                                       trigger:trigger1];
[center addNotificationRequest:request withCompletionHandler:^(NSError * _Nullable error) {

}];
```

9. 通知实现小结

- Local Notifications:

1. 定义 `Content` .
2. 定义 `Trigger` .
3. 向 `UNUserNotificationCenter` 发送 `request` .



- Remote Notifications: 向 APNs 发送 `Notification Payload` .

处理通知

1. UNUserNotificationCenterDelegate

`UNUserNotificationCenterDelegate` 提供了两个方法： - 在应用内展示通知。App 处于前台时捕捉并处理即将触发的推送:

```
@interface AppDelegate () <UNUserNotificationCenterDelegate>

-(void)userNotificationCenter:(UNUserNotificationCenter *)center willPresentNotification:(UNNoti

    completionHandler(UNNotificationPresentationOptionAlert | UNNotificationPresentationOptionSc

}
```

- 收到通知响应时的处理工作。用户与你推送的通知进行交互时被调用:

```
- (void)userNotificationCenter:(UNUserNotificationCenter *)center didReceiveNotificationResponse

      completionHandler()//什么也不做
}
```

2. Notification Management

可以对通知执行查、改、删操作。实现该功能需要有一个必要参数 `identifer` ，后续的查改删操作

都是根据此参数去执行的。 - Local Notification：通过更新 request. - Remote Notification 通过新的字段 `apns-collapse-id` .

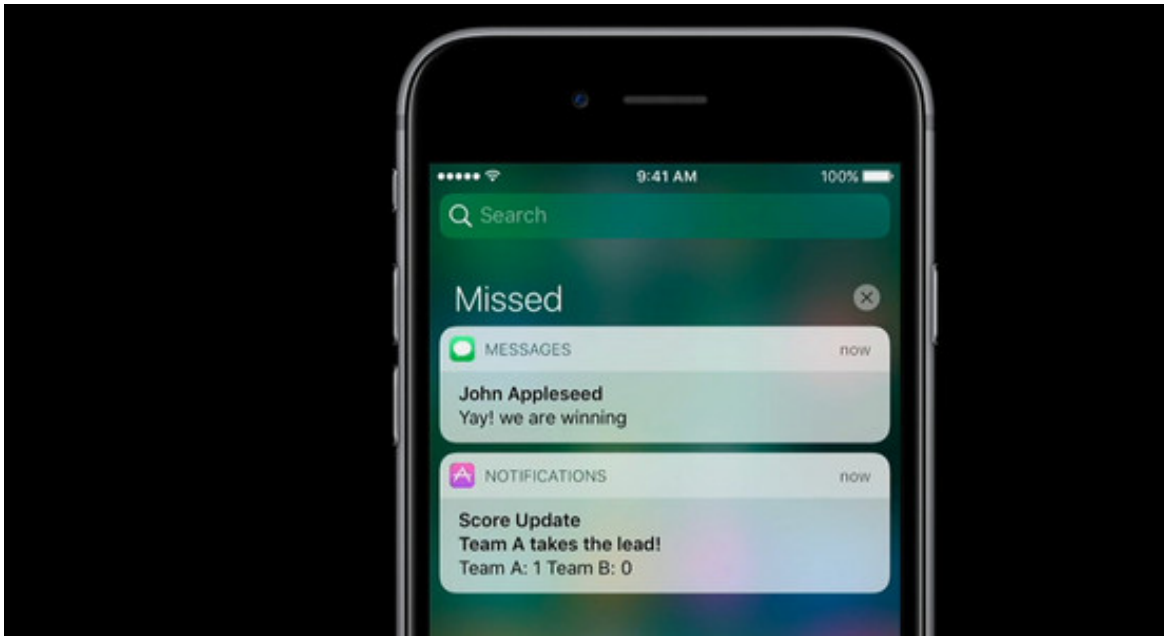
更新原有推送：

```
NSString *requestIdentifier = @"sampleRequest";
UNNotificationRequest *request = [UNNotificationRequest requestWithIdentifier:requestIdentifier
                                                                       content:newContent
                                                                       trigger:newTrigger1];
[center addNotificationRequest:request withCompletionHandler:^(NSError * _Nullable error) {

}];
```

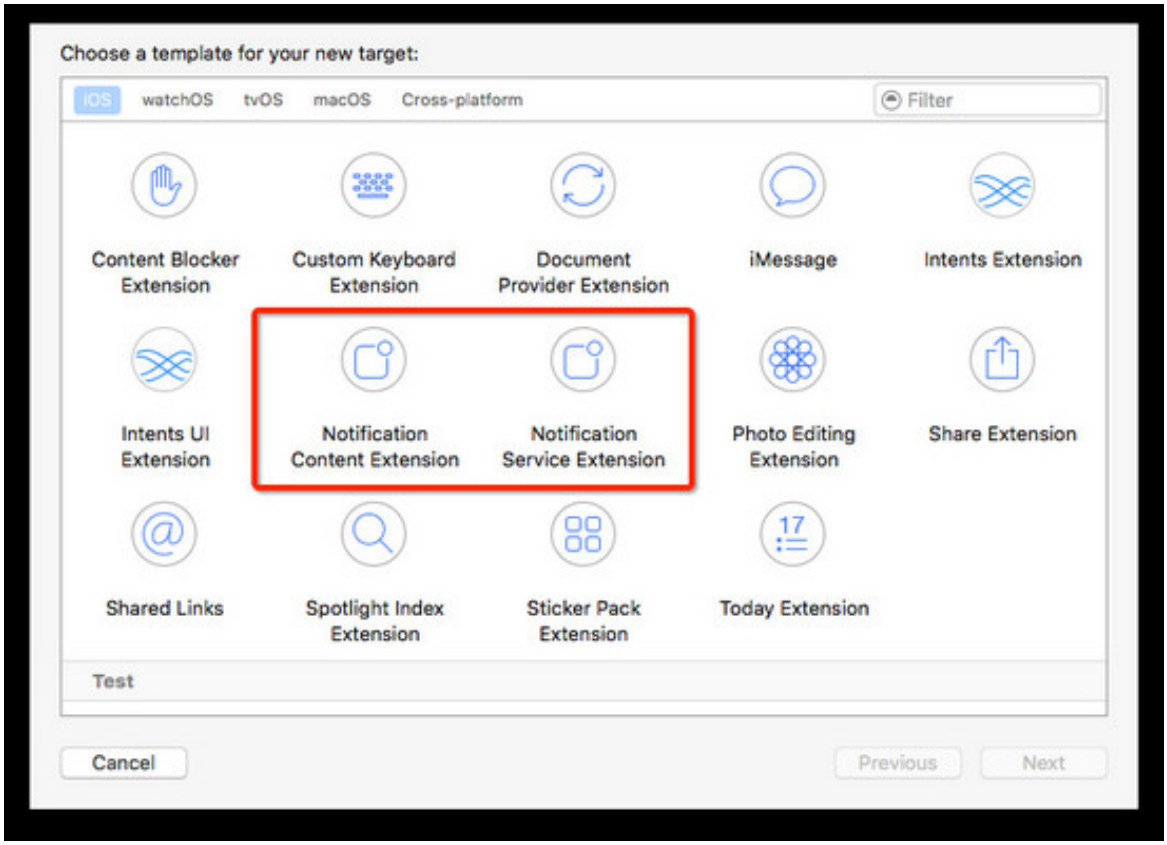
删除推送：

```
[center removePendingNotificationRequestsWithIdentifiers:@[requestIdentifier]];
```

典型应用场景： - 赛事比分更新 - 撤回通知



Notification Extension



Service Extension

Service Extension允许在收到远程推送的通知后，展示之前对通知内容进行修改。

```
class NotificationService: UNNotificationServiceExtension {

    var contentHandler: ((UNNotificationContent) -> Void)?
```

```
    var bestAttemptContent: UNMutableNotificationContent?

    override func didReceive(_ request: UNNotificationRequest, withContentHandler contentHandler
        self.contentHandler = contentHandler
        bestAttemptContent = (request.content.mutableCopy() as? UNMutableNotificationContent)

        if let bestAttemptContent = bestAttemptContent {
            if request.identifier == "mutableContent" {
                bestAttemptContent.body = "\(bestAttemptContent.body),tuneszhao"
            }
            contentHandler(bestAttemptContent)
        }
    }

    override func serviceExtensionTimeWillExpire() {
        // Called just before the extension will be terminated by the system.
        // Use this as an opportunity to deliver your "best attempt" at modified content, otherw
        if let contentHandler = contentHandler, let bestAttemptContent =  bestAttemptContent {
            contentHandler(bestAttemptContent)
        }
    }
}
```

- `didReceive:` 方法中通过修改请求的 `content` 内容，然后在限制的时间内将修改后的内容调用通过 `contentHandler` 返还给系统，就可以显示这个修改过的通知了。(本例中在原有 content 之后添加了字符串 “tuneszhao”)。
- `serviceExtensionTimeWillExpire:`：在一定时间内没有调用 `contentHandler` 的话，系统会调用这个方法。可以选择什么都不做，系统将当作什么都没发生，简单地显示原来的通知。可能你其实已经设置好了绝大部分内容，只是有很少一部分没有完成，这时你也可以像例子中这样调用 contentHandler 来显示一个变更“中途”的通知。

`Service Extension` 只对远程推送的通知有效，启用内容修改要在推送 `payload` 中设置 `mutable-content` 值为1：

```
{
  "aps":{
    "alert":{
      "title":"Greetings",
      "body":"Long time no see"
    },
    "mutable-content":1
  }
}
```

该特性可用于推送内容加密。服务器推送 `payload` 中加入加密过的文本，在客户端接到通知后使用预先定义或者获取过的密钥进行解密，然后显示。这样可以保证传递内容的安全。

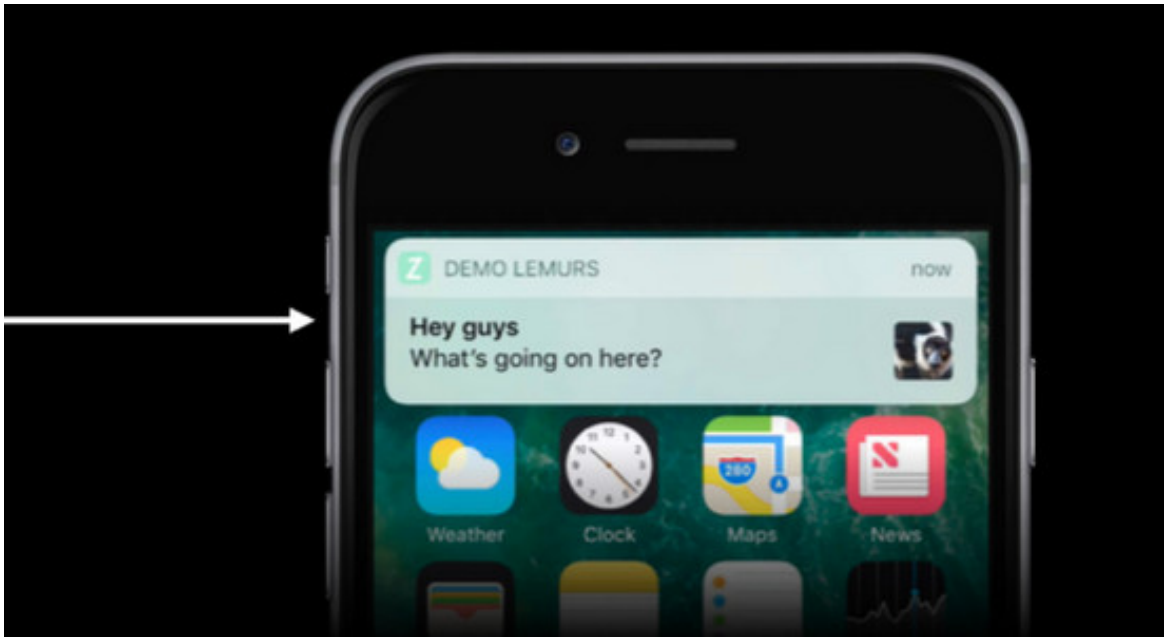
Media Attachments

iOS 10 的另一个亮眼功能是多媒体的推送，开发者可以在通知中嵌入图片或视频。

为本地通知添加图片／视频

通过本地磁盘上的文件 URL 创建一个 UNNotificationAttachment 对象，然后将这个对象放到数组中赋值给 content 的 attachments 属性：

```
let content = UNMutableNotificationContent()
  content.title = "Hey guys"
content.body = "What's going on here?"

if let imageURL = Bundle.main.url(forResource: "image", withExtension: "jpg"),
   let attachment = try? UNNotificationAttachment(identifier: "imageAttachment", url: imageURL,
{
    content.attachments = [attachment]
}
```



为远程推送添加多媒体内容

需要借助上面提到的 Notification Service Extension 。 - payload 中指定需要加载的图片资源地址，这个地址可以是应用 bundle 内已经存在的资源，也可以是网络的资源。 - 如果多媒体不在本地的话，需要先将其下载到本地。 - 创建 UNNotificationAttachment ,之后和本地通知一样，将多媒体资源设置给 attachments 属性，然后调用 contentHandler 。

```
{
  "aps":{
    "alert":{
      "title":"Image Notification",
      "body":"Show me an image from web!"
    },
    "mutable-content":1
  },
  "image": "http://ww2.sinaimg.cn/large/005tGCqhgw1f7xlo99zmsj30og0dc402.jpg"
}
```

下载图片代码：

```
private func downloadAndSave(url: URL, handler: @escaping (_ localURL: URL?) -> Void) {
    let task = URLSession.shared.dataTask(with: url, completionHandler: {
```

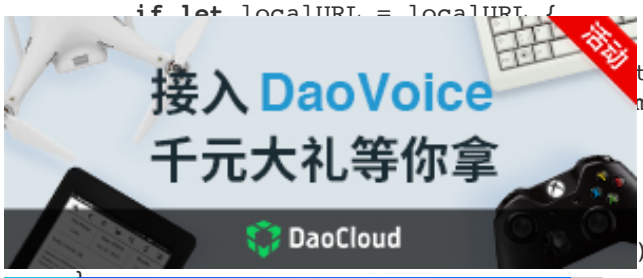
```
        data, res, error in

        var localURL: URL? = nil

        if let data = data {
            let ext = (url.absoluteString as NSString).pathExtension
            let cacheURL = URL(fileURLWithPath: FileManager.default.cachesDirectory)
            let url = cacheURL.appendingPathComponent(url.absoluteString.md5).appendingPathExter

            if let _ = try? data.write(to: url) {
                localURL = url
            }
        }

        handler(localURL)
    })

    task.resume()
}
```

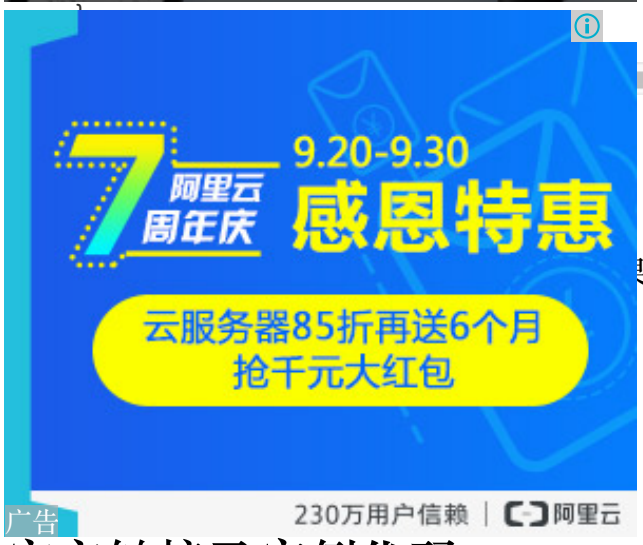
didReceive: 中接收通知获取图片地址后下载，并创建 attachment 展示：

```
if let imageURLString = bestAttemptContent.userInfo["image"] as? String,
   let URL = URL(string: imageURLString)
{
    downloadAndSave(url: URL) { localURL in
        if let localURL = localURL {
```

```
tificationAttachment(identifier: "image_downloaded", ur
ments = [attachment]
```

```
)
```





果需要支持 iOS 10 之前的系统，还是需要使用原来的
为应用通知带来更多新特性，可以使用：

官方链接及实例代码

分享

收藏 纠错

专访：听道哥聊互联网江湖

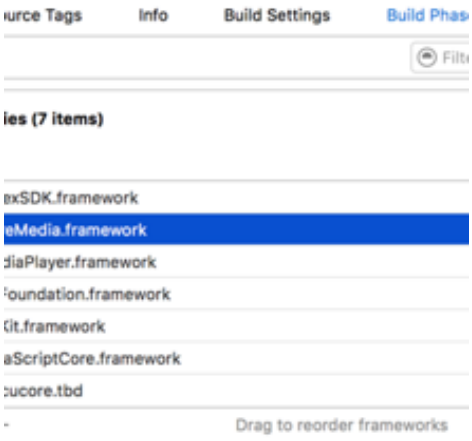
“根据判断，下一个被黑产盯上的行业很有可能是直播行业。现在来看，这件事情已经发生了。”

推荐文章

- 1. Xcode8和iOS10问题小结
- 2. 获取当前最顶层的ViewController
- 3. 开始一步一步学习Message App Extension
- 4. 独立双端App《瓦格相机》的开发过程分享
- 5. iOS 10 UserNotifications 使用说明
- 6. [iOS] mas：命令行下 Mac App Store 应用安装工具

相关推刊

- by narklon 

《默认推刊》 *1*
- by binbinweiwei 

《iOS_binbinweiwei》 *70*

- by kafka0102  《iOS开发》 *11*

我来评几句

请输入评论内容...

登录后评论

已发表评论数(0)

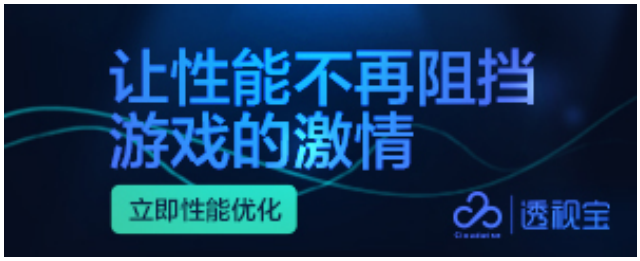
相关站点

ZLtunes' Connect

＋订阅

热门文章

- 1. Xcode8和iOS10问题小结
- 2. 获取当前最顶层的ViewController
- 3. 开始一步一步学习Message App Extension
- 4. 独立双端App《瓦格相机》的开发过程分享
- 5. iOS 10 UserNotifications 使用说明



收藏到推刊 创建推刊

收 藏 取消

推刊名(必填) 请填写推刊名

推刊描述

描述不能大于100个字符!

权限设置: 公开 仅自己可见

创建 取消

×

文章纠错

邮箱地址
错误类型 正文不准确
补充信息
提交

网站相关

- 关于我们
- 移动应用
- 建议反馈

关注我们

- 推酷网
- tuicool2012

友情链接

人人都是产品经理 PM256 移动信息化 行晓网 智城外包网 虎嗅 IT耳朵 创媒工场 经理人分享 市场部网 砍柴网 CocoaChina 北风网 云智慧 我赢职场 大数据时代 奇笛网 咕噜网 红联linux Win10之家 鸟哥笔记 爱游戏 投资潮 31会议网 极光推送 Teambition 硅谷网 leangoo ZEALER中国 OpenSNS 小牛学堂 handone Scrum中文网 比戈大牛 又拍云 更多链接>>