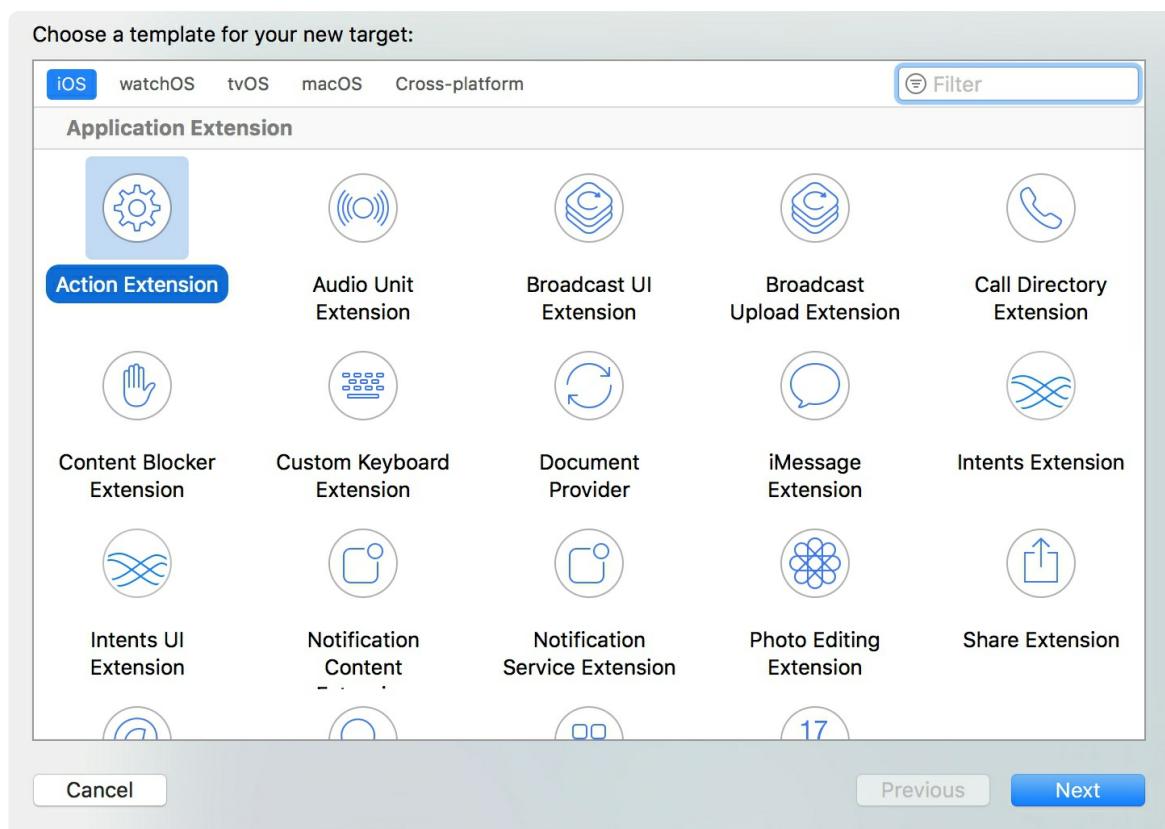


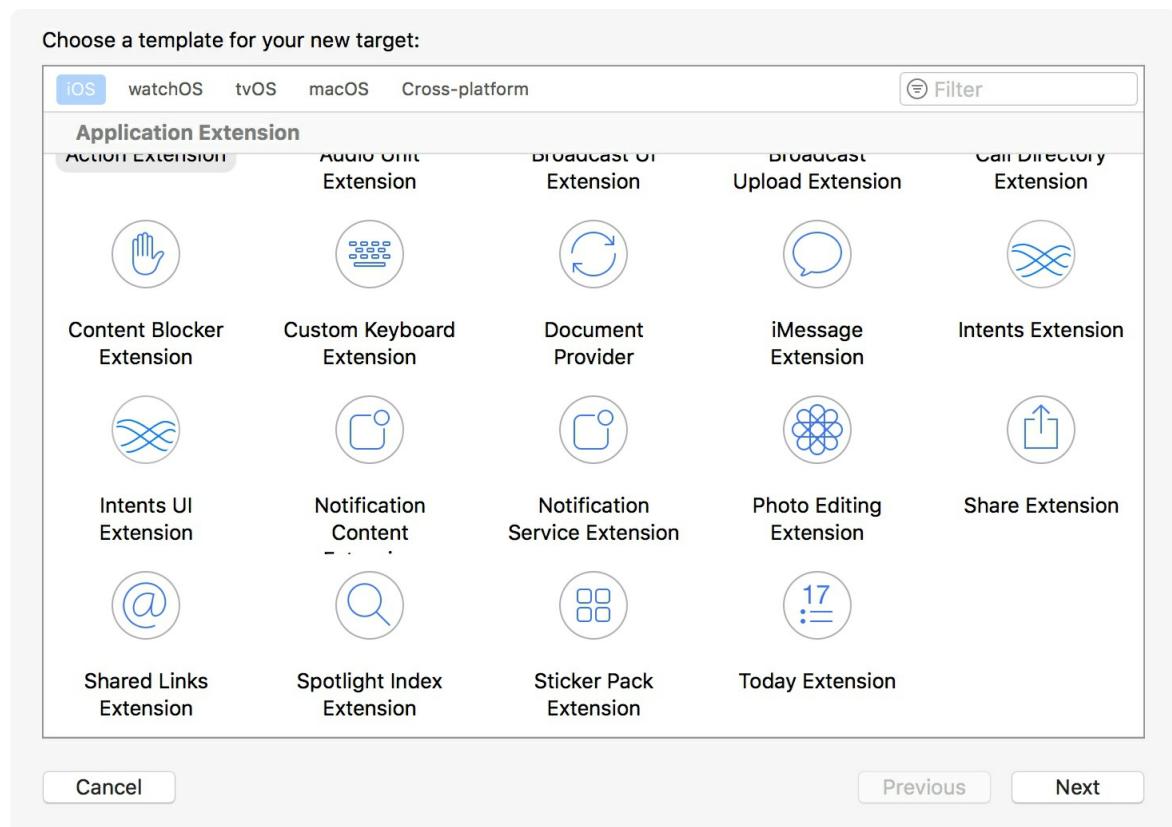
Application Extension 践坑指南

一. 什么是扩展?

扩展（Extension）是从 iOS 8 开始中引入的一个非常重要的新特性。扩展让 app 之间的数据交互成为可能。用户可以在 app 中使用其他应用提供的功能，而无需离开当前的应用。在 iOS 8 系统之前，每一个 app 在物理上都是彼此独立的，app 之间不能互访彼此的私有数据。而在引入扩展之后，其他 app 可以与扩展进行数据交换。基于安全和性能的考虑，每一个扩展运行在一个单独的进程中，它拥有自己的 bundle，bundle 后缀名是.appex。扩展 bundle 必须包含在一个普通应用的 bundle 的内部。

iOS 10 系统主要有 6 个常用支持扩展的系统区域，分别是 Today、Share、Action、Photo Editing、Storage Provider、Custom keyboard。类别总共19种Application Extension。





1. Today Widget

对于赛事比分，股票、天气、快递这类需要实时获取的信息，可以在通知中心的Today 视图中创建一个 Today 扩展实现。 Today 扩展又称为 Widget 。

2. Share

在 iOS 8 之前，用户只有 Facebook, Twitter 等有限的几个分享选项可以选择。如果希望将内容分享到 Pinterest ，开发者则需要一些额外的努力。在 iOS 8 中，开发者可以创建自定义的分享选项。

3. Action

action 在所有支持的扩展点中扩展性最强的一个。它可以实现转换另一个 app 上下文中的内容。苹果在 WWDC 大会上演示了一个 Bing 翻译动作扩展，它可以将在 Safari 中选中的文本翻译成不同的语言。

4. Photo Editing

在 iOS 8 之前，如果你想为你的照片添加一个特殊的滤镜，你需要进入第三方 app 中，这个过程是相当繁琐的。在 iOS 8 中，你可以直接在 Photos 中使用第三方 app ，如 Instagram , VSCO cam 、 Aviary 提供的 Photo Editing 扩展完成对图片的编辑，而无需离开当前的 app 。

5. Storage Provider

Storage Provider 让跨多个文件存储服务之间的管理变得更简单。类似 Dropbox 、 Google Drive 等存储提供商通过在 iOS 8 中提供一个 Storage Provider 扩展， app 直接可以使用这些扩展检索和存储文件而不再需要创建不必要的拷贝。

6. Custom Keyboard

苹果公司在 2007 年率先推出了触摸屏键盘，但一直没多大改进。在这一方面， Android 则将键盘权限开放给了第三方开发者，所以出现了许多像 Swype ， SwiftKey 等优秀的键盘输入法。在 iOS 8 中，苹果终于将键盘权限开发给了第三方开发者，自定义键盘输入法可以让用户在整个系统范围内使用。

以下是 iOS 9 中新增扩展

7. 网络扩展

开发者可以通过改扩展来实现自定义的VPN客户端、透明的网络代理客户端以及实现动态的设备端网络内容过滤。

8. Safari 扩展

该扩展可以让用户通过Safari的分享链接看到你的内容。又或者提供一个屏蔽列表，让你的用户使用你的App浏览Web内容时屏蔽指定的内容。

9. Spotlight 扩展

该扩展可以对App内的数据进行索引，并且可以在不重启App的情况下重建数据索引。

10. Audio Unit 扩展

该扩展允许App提供类似于GarageBand， Logic等App提供的乐器演奏，音频特效，声音合成功能。

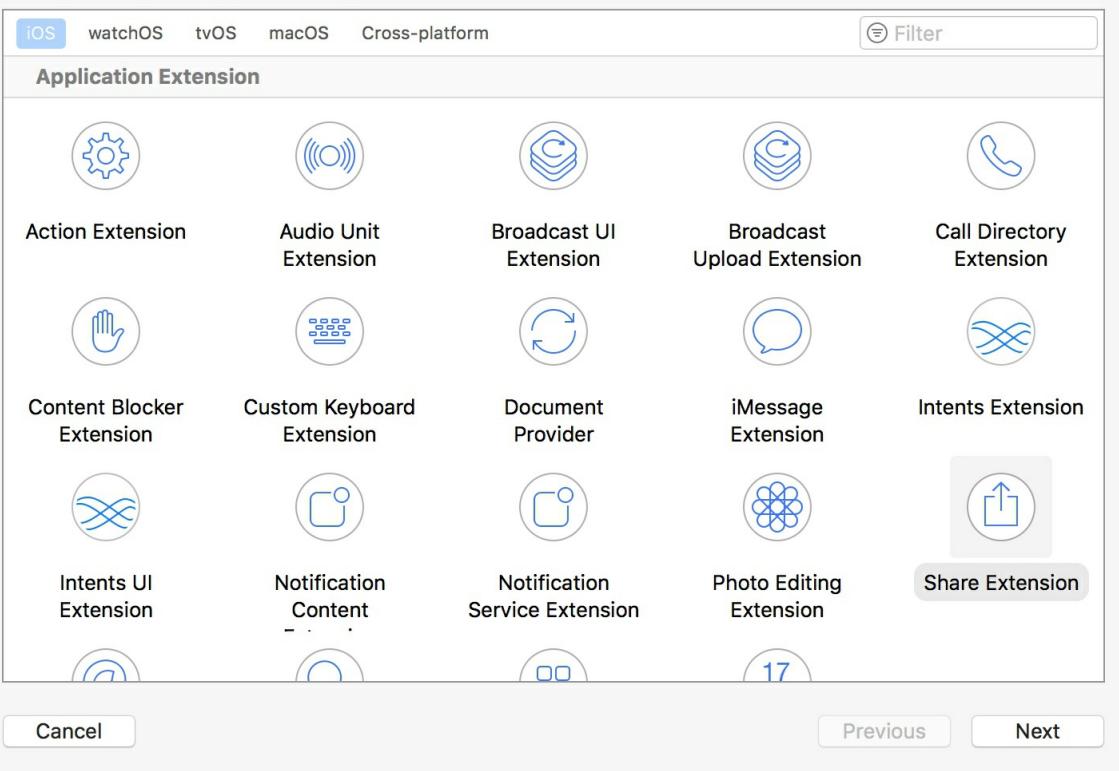
二. Share Extension

1. 创建Share Extension 扩展Target

注：扩展不能单独创建，必须依赖于应用工程项目，因此如果你还没有创建一个应用工程，先去创建一个。

1、打开项目设置，在TARGETS侧栏地下点击“+”号来创建一个新的Target，如图：

Choose a template for your new target:



2、然后选择“iOS” -> “Application Extension” -> “Share Extension”，点击“Next”。如图：

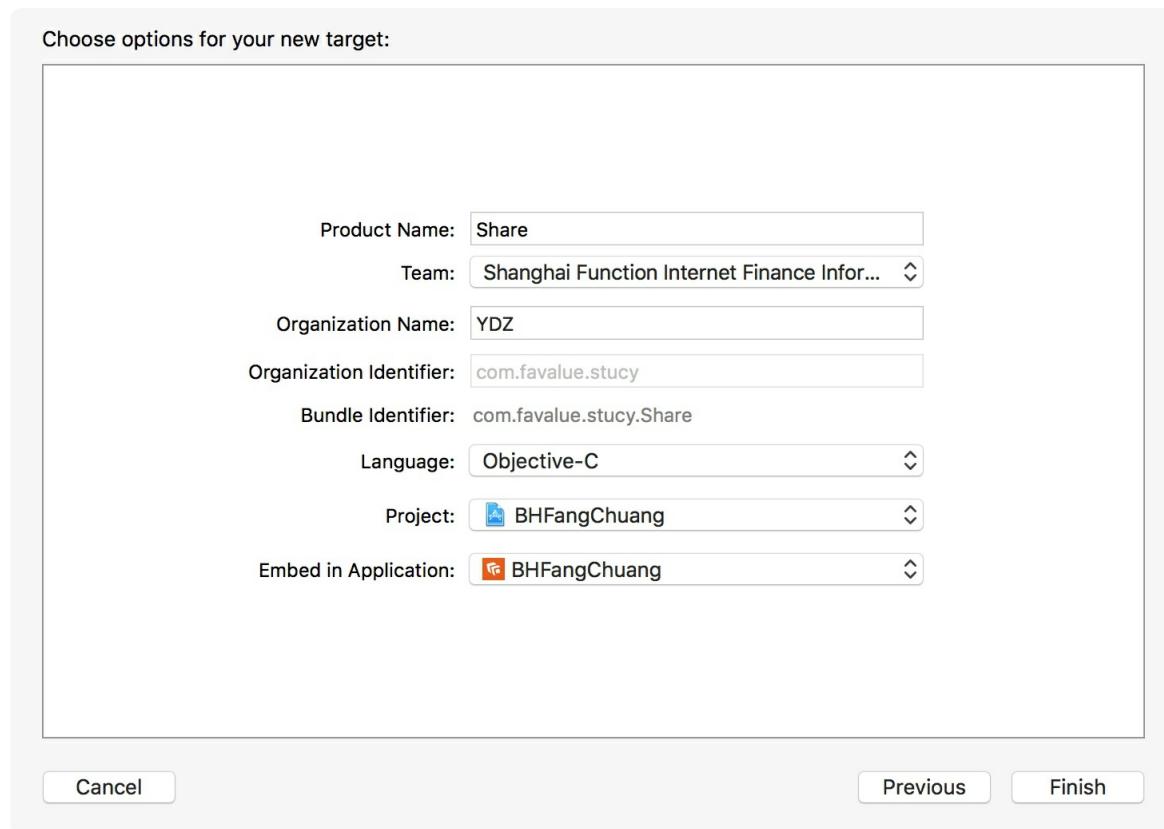
Choose options for your new target:

This screenshot shows the configuration options for a new Share Extension target. The fields are as follows:

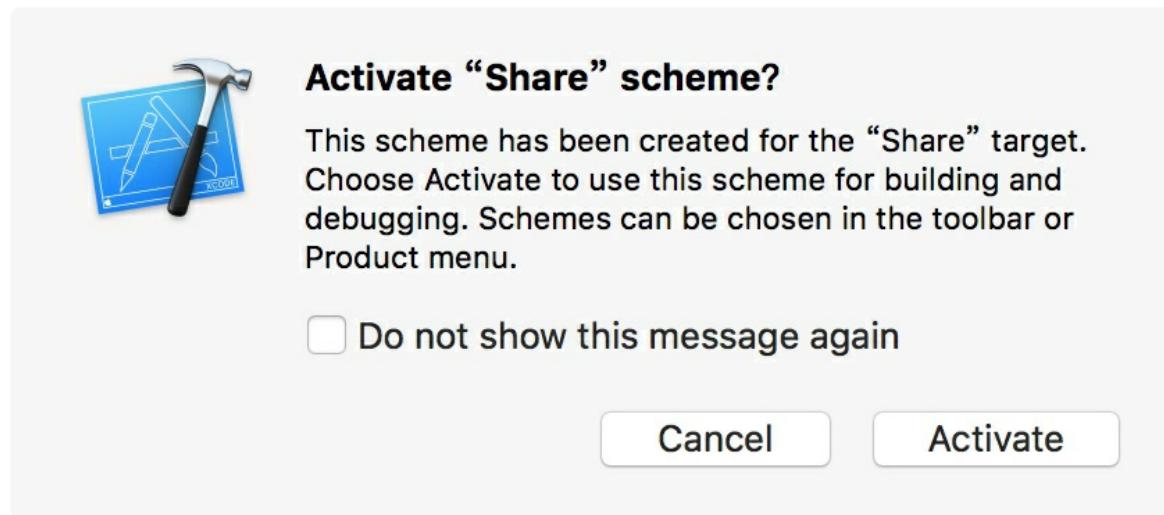
- Product Name: [Empty input field]
- Team: Shanghai Function Internet Finance Infor... [Dropdown menu]
- Organization Name: YDZ [Input field]
- Organization Identifier: com.favalue.stucy [Input field]
- Bundle Identifier: com.favalue.stucy.ProductName [Input field]
- Language: Objective-C [Dropdown menu]
- Project: BHFangChuang [Dropdown menu]
- Embed in Application: BHFangChuang [Dropdown menu]

At the bottom, there are 'Cancel', 'Previous', and 'Finish' buttons.

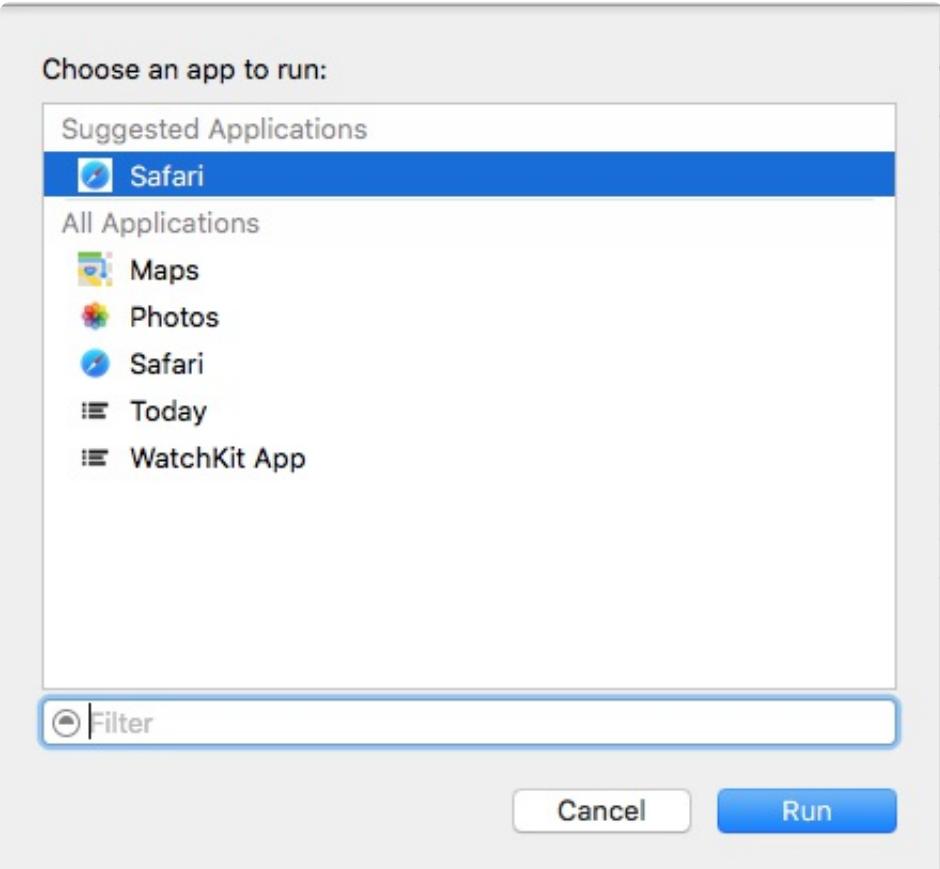
3、给扩展起个名字，这里填写了“Share”，点击“Finish”。如图：



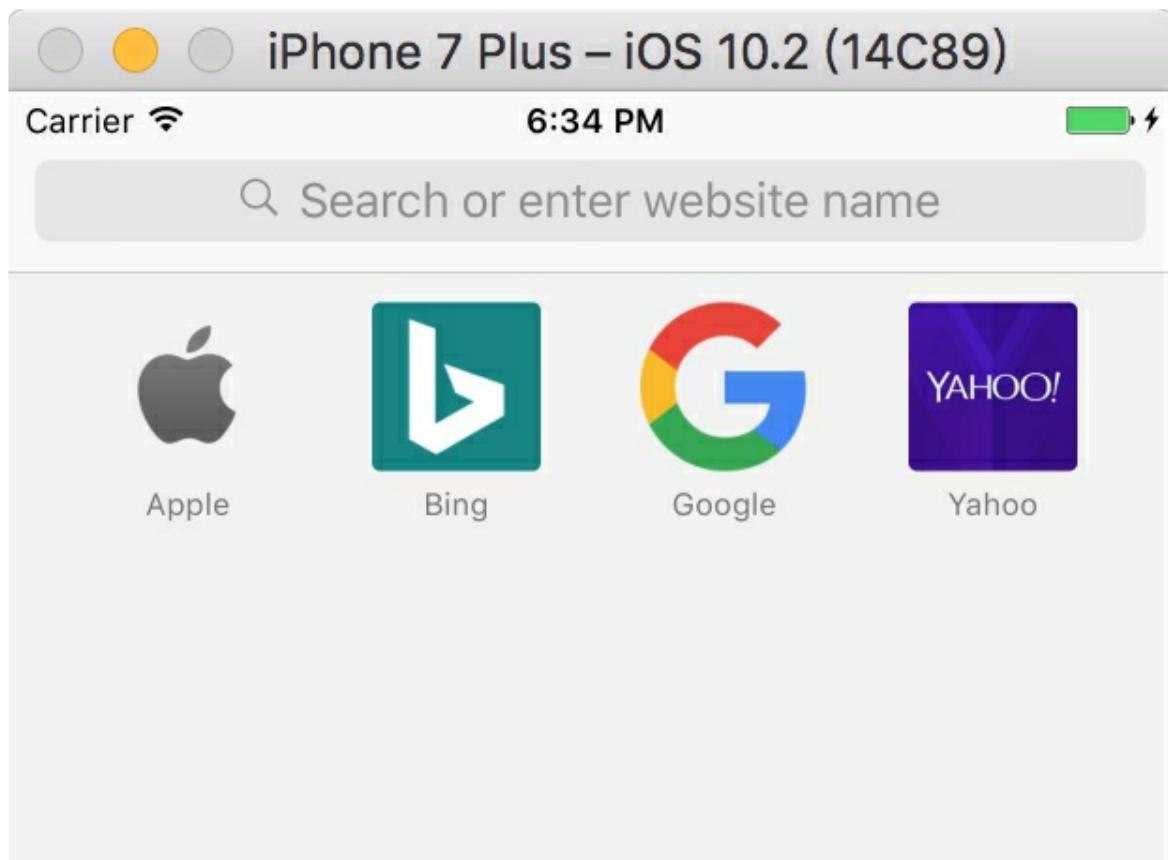
4、这时候会提示创建一个Scheme，点击“Activate”。如图：

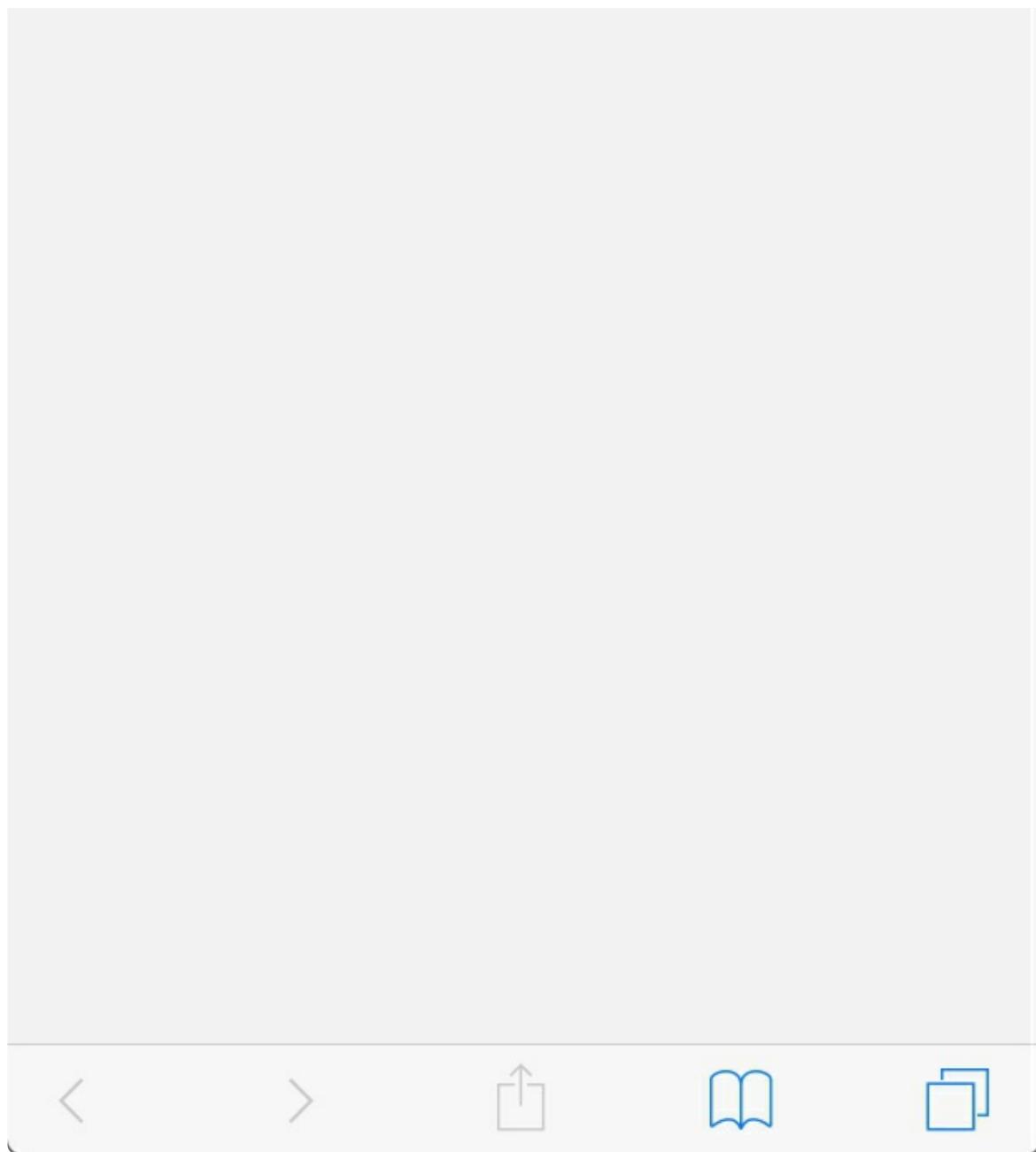


那么，直到这里创建Share Extension的工作就算是完成了。接下来可以先进行一下编译运行。这里跟做App开发的时候会稍微有点不一样。因为Extension是需要Host App（宿主应用）来运行的。所以，XCode中会弹出界面让我们选择一个iOS的App来运行Extension。如图：

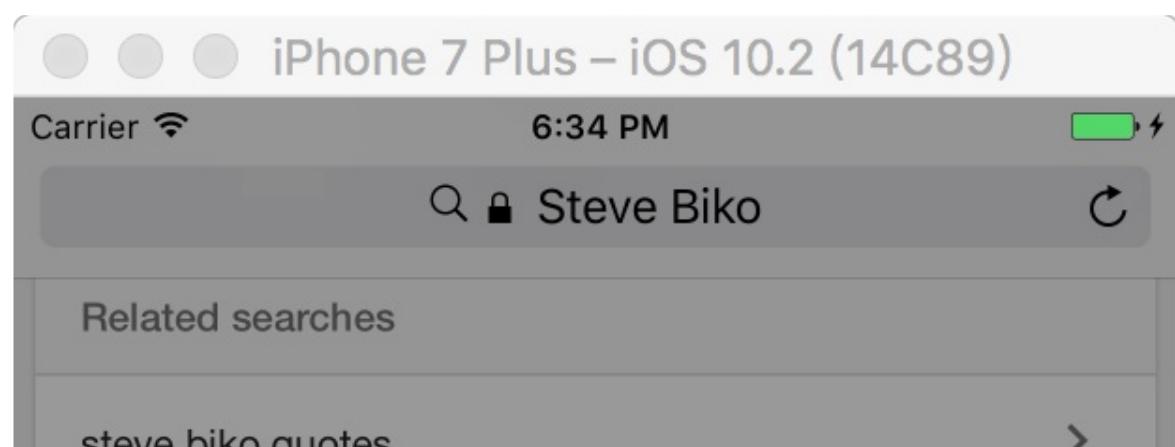


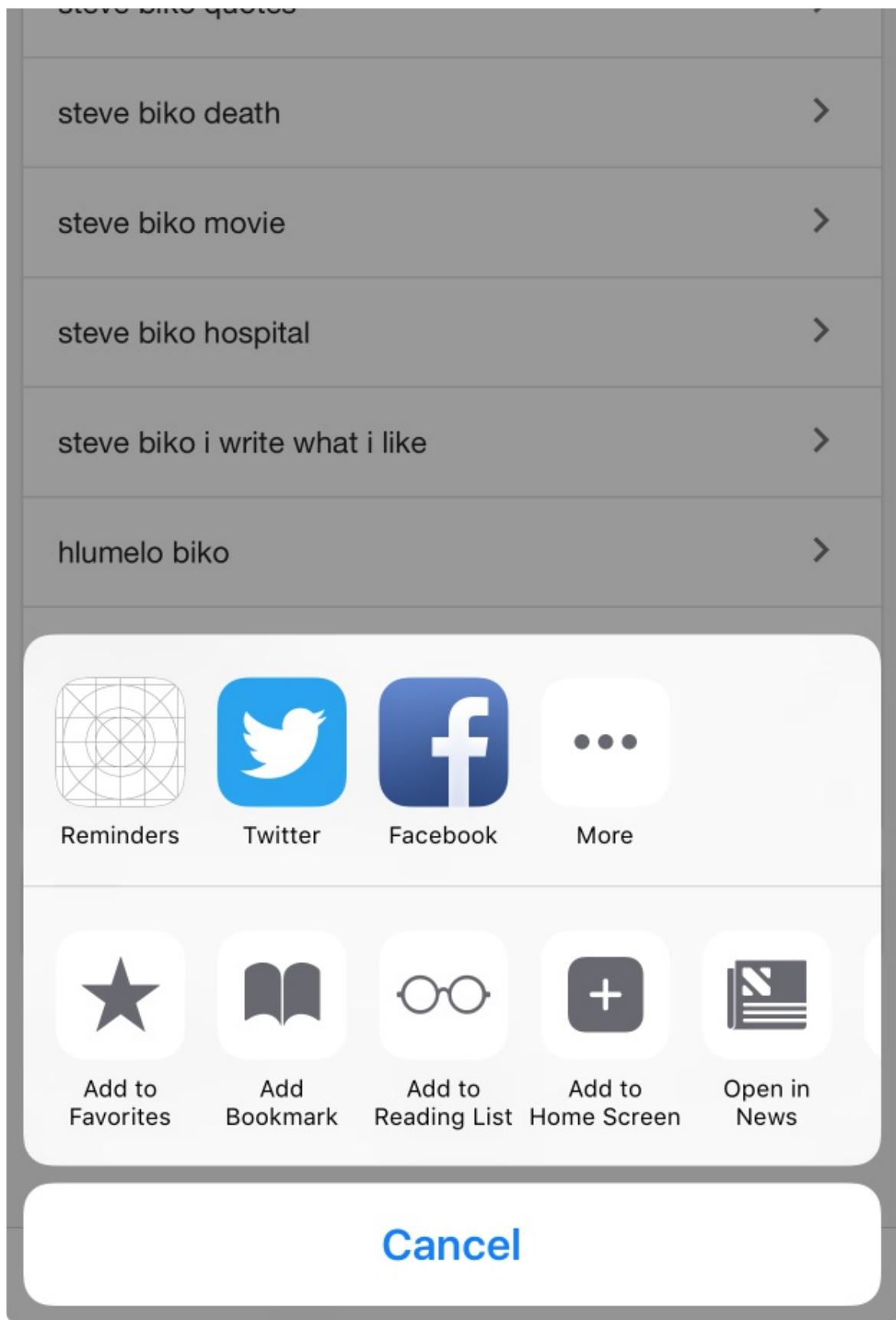
这里我选择了XCode建议的应用Safari，然后点击“Run”来进行调试运行。XCode会启动Safari，如图：





能看到Safari中间的分享按钮是灰色不可用的。我们随便点开一个网页，可以看到分享按钮变为激活状态。点击分享按钮就会弹出分享菜单，如图：





可以看到刚才建立的Share扩展已经显示在面板上了，如果你没有发现自己的扩展，那么你可以将菜单滑动到最右边，在“更多”选项中激活自己的扩展。如图：

iPhone 7 Plus – iOS 10.2 (14C89)

Carrier

6:34 PM

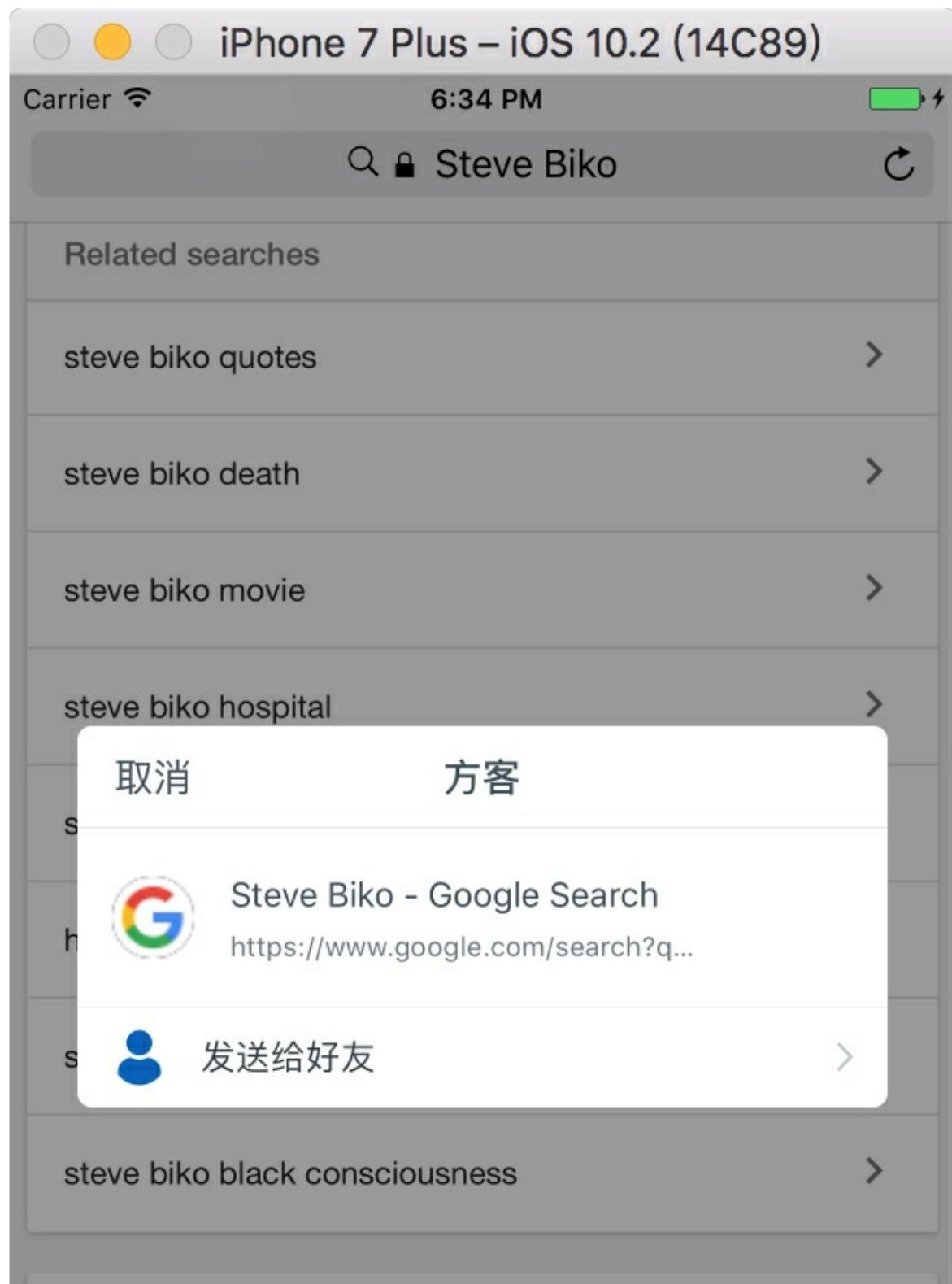


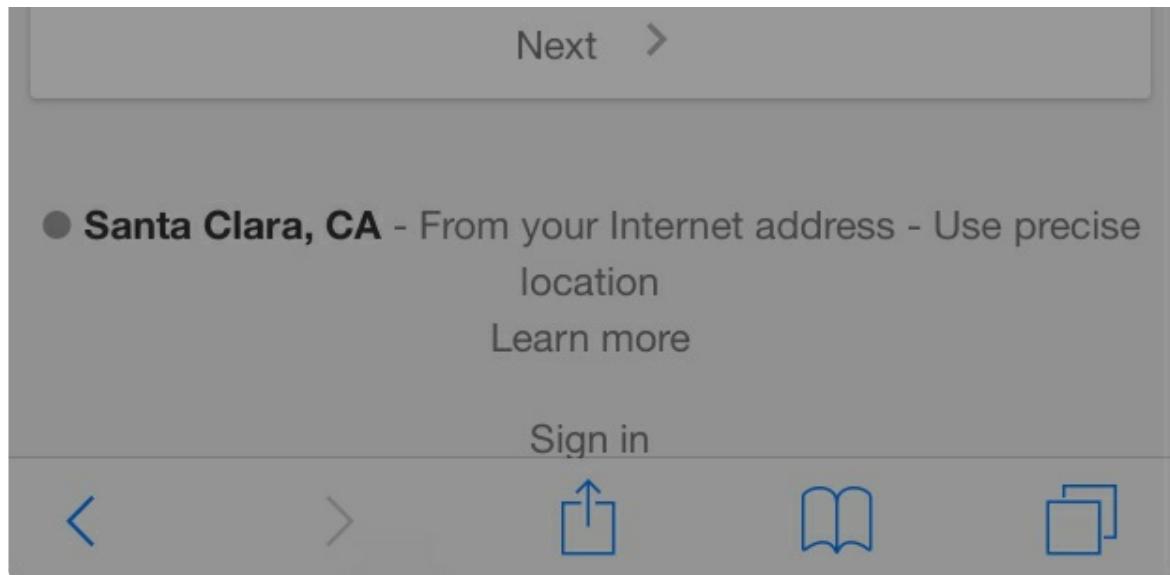
Activities

Done

	Reminders	<input checked="" type="checkbox"/>	
	Twitter	<input checked="" type="checkbox"/>	
	Facebook	<input checked="" type="checkbox"/>	
	KurtWangShareExtension	<input type="checkbox"/>	
	方客	<input checked="" type="checkbox"/>	

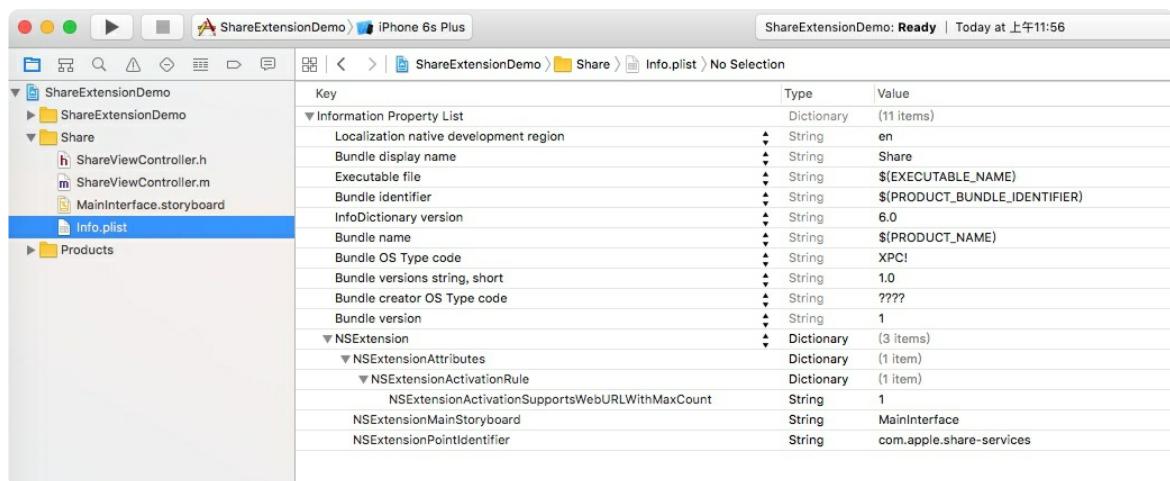
我们点击自己创建的分享项，其弹出一个分享窗口。如图：





2. 配置Share Extension

接下来我们需要给他一些设置。我们展开XCode左侧栏的Share目录，找到Info.plist文件。如：



我们只需要关注以下几个字段的设置：

名称	说明
Bundle display name	扩展的显示名称，默认跟你的项目名称相同，可以通过修改此字段来控制扩展的显示名称。
NSExtension	扩展描述字段，用于描述扩展的属性、设置等。作为一个扩展项目必须要包含此字段。
NSExtensionAttributes	扩展属性集合字段。用于描述扩展的属性。
NSExtensionActivationRule	激活扩展的规则。默认为字符串“TRUEPREDICATE”，表示在分享菜单中一直显示该扩展。可以将类型改为Dictionary类型，然后添加以下字段： NSExtensionActivationSupportsAttachmentsWithMaxCount NSExtensionActivationSupportsAttachmentsWithMinCount NSExtensionActivationSupportsImageWithMaxCount NSExtensionActivationSupportsMovieWithMaxCount NSExtensionActivationSupportsWebPageWithMaxCount NSExtensionActivationSupportsWebURLWithMaxCount
NSExtensionMainStoryboard	设置主界面的Storyboard，如果不想使用storyboard，也可以使用NSExtensionPrincipalClass指定自定义UIViewController子类名
NSExtensionPointIdentifier	扩展标识，在分享扩展中为：com.apple.share-services
NSExtensionPrincipalClass	自定义UI的类名
NSExtensionActivationSupportsAttachmentsWithMaxCount	附件最多限制，为数值类型。附件包括File、Image和Movie三大类，单一、混选总量不超过指定数量

NSExtensionActivationSupportsAttachmentsWithMinCount	附件最少限制，为数值类型。当设置NSExtensionActivationSupportsAttachmentsWithMaxCount时生效，默认至少选择1个附件，分享菜单中才显示扩展插件图标。
NSExtensionActivationSupportsFileWithMaxCount	文件最多限制，为数值类型。文件泛指除Image/Movie之外的附件，例如【邮件】附件、【语音备忘录】等。 单一、混选均不超过指定数量。
NSExtensionActivationSupportsImageWithMaxCount	图片最多限制，为数值类型。单一、混选均不超过指定数量。
NSExtensionActivationSupportsMovieWithMaxCount	视频最多限制，为数值类型。单一、混选均不超过指定数量。
NSExtensionActivationSupportsText	是否支持文本类型，布尔类型，默认不支持。如【备忘录】的分享
NSExtensionActivationSupportsWebURLWithMaxCount	Web链接最多限制，为数值类型。默认不支持分享超链接，需要自己设置一个数值。
NSExtensionActivationSupportsWebPageWithMaxCount	Web页面最多限制，为数值类型。默认不支持Web页面分享，需要自己设置一个数值。

对于不同的应用里面有可能出现只允许接受某种类型的内容，那么Share Extension就不能一直出现在分享菜单中，因为不同的应用提供的分享内容不一样，这就需要通过设置NSExtensionActivationRule字段来决定Share Extension是否显示。例如，只想接受其他应用分享链接到自己的应用，那么可以通过下面的步骤来设置：

1. 将NSExtensionActivationRule字段类型由String改为Dictionary。
2. 展开NSExtensionActivationRule字段，创建其子项
NSExtensionActivationSupportsWebURLWithMaxCount，并设置一个限制数量。

调整后如下图所示：

Key	Type	Value
▼Information Property List	Dictionary	(11 items)
Localization native development region	String	en
Bundle display name	String	Share
Executable file	String	\$(EXECUTABLE_NAME)
Bundle identifier	String	\$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	\$(PRODUCT_NAME)
Bundle OS Type code	String	XPC!
Bundle versions string, short	String	1.0
Bundle creator OS Type code	String	????
Bundle version	String	1
▼NSExtension	Dictionary	(3 items)
▼NSExtensionAttributes	Dictionary	(1 item)
▼NSExtensionActivationRule	Dictionary	(1 item)
NSExtensionActivationSupportsWebURLWithMaxCount	String	1
NSExtensionMainStoryboard	String	MainInterface
NSExtensionPointIdentifier	String	com.apple.share-services

3. 处理Share Extension中的数据

其实在Share Extension中默认都会有一个数据展现的UI界面。该界面继承SLComposeServiceViewController这个类型，如：

```
@interface ShareViewController : SLComposeServiceViewController  
@end
```

一般情况下，当用户点击提交按钮的时候，扩展要做的事情就是要把数据取出来，并且放入一个与Containing App（容器程序，尽管苹果开放了Extension，但是在iOS中extension并不能单独存在，要想提交到AppStore，必须将Extension包含在一个App中提交，并且App的实现部分不能为空，这个包含Extension的App就叫Containing app。Extension会随着Containing App的安装而安装，同时随着ContainingApp的卸载而卸载。）共享的数据介质中（包括NSUserDefaults、Sqlite、CoreData），要跟容器程序进行数据交互需要借助AppGroups服务。

在ShareExtension中，UIViewController包含一个extensionContext这样的上下文对象：

```
@interface UIViewController(NSExtensionAdditions) <NSExtensionRequestHandling>  
  
// Returns the extension context. Also acts as a convenience method for a view controller to check if it participating in an extension request.
```

```

@property (nullable, nonatomic,readonly,strong) NSExtensionContext *extensionCont
ext NS_AVAILABLE_IOS(8_0);

@end

```

通过操作它就可以获取到分享的数据，返回宿主应用的界面等操作。具体的可以查看苹果官方文档 NSExtensionContext。

NSExtensionContext的结构比较简单，包含一个属性和三个方法。其说明如下：

方法	说明
inputItems	该数组存储着容器应用传入给 NSExtensionContext的NSExtensionItem数组。其中每个NSExtensionItem标识了一种类型的数据。要获取数据就要从这个属性入手。
completeRequestReturningItems:completionHandler:	通知宿主程序的扩展已完成请求。调用此方法后，扩展UI会关闭并返回容器程序中。其中的 items就是返回宿主程序的数据项。
cancelRequestWithError:	通知宿主程序的扩展已取消请求。调用此方法后，扩展UI会关闭并返回容器程序中。其中error为错误的描述信息。
NSExtensionItemsAndErrorsKey	NSExtensionItem的userInfo属性中对应的错误信息键名。

通知名称	说明
NSExtensionHostWillEnterForegroundNotification	宿主程序将要返回前台通知
NSExtensionHostDidEnterBackgroundNotification	宿主程序进入后台通知
NSExtensionHostWillResignActiveNotification	宿主程序将要被挂起通知
NSExtensionHostDidBecomeActiveNotification	宿主程序被激活通知

4. 拿到分享的数据

要拿到分享的数据，就需要读取NSExtensionItem里面的数据。

NSExtensionItem包含四个属性

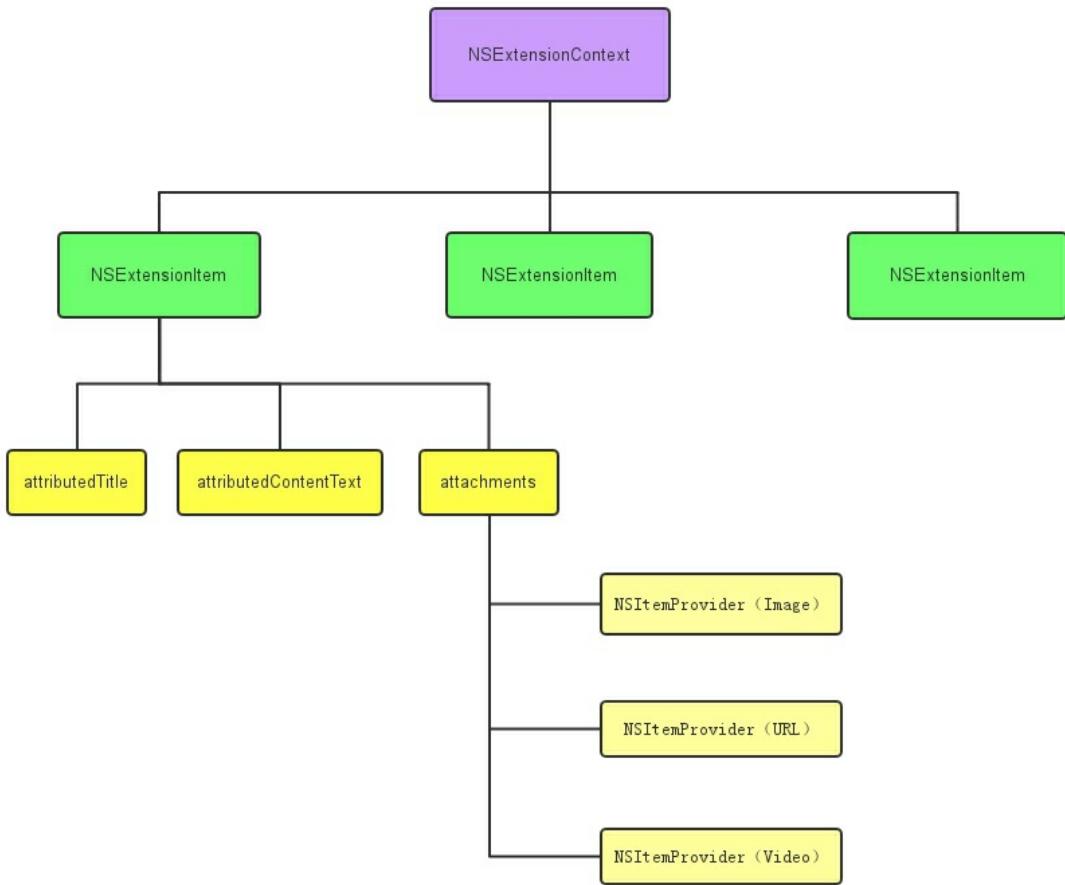
属性	说明
attributedTitle	标题。
attributedContentText	内容。
attachments	附件数组，包含图片、视频、链接等资源，封装在 NSItemProvider类型中。
userInfo	一个key-value结构的数据。NSExtensionItem中的属性都会在这个属性中一一映射。

对应userInfo结构中的NSExtensionItem属性的键名如下图：

名称	说明
NSExtensionItemAttributedTitleKey	标题的键名
NSExtensionItemAttributedContentTextKey	内容的键名
NSExtensionItemAttachmentsKey	附件的键名

附件又是封装在一个叫NSItemProvider的类型中

方法	说明
initWithItem:typelIdentifier:	初始化方法，item为附件的数据，typelIdentifier是附件对应的类型标识,对应UTI的描述。
initWithContentsOfURL:	根据制定的文件路径来初始化。
registerItemForTypelIdentifier:loadHandler:	为一种资源类型自定义加载过程。这个方法主要针对自定义资源使用，例如自己定义的类或者文件格式等。当调用 loadItemForTypelIdentifier:options:completionHandler: 方法时就会触发定义的加载过程。
hasItemConformingToTypelIdentifier:	用于判断是否有typelIdentifier(UTI)所指定的资源存在。存在则返回YES，否则返回NO。 该方法结合 loadItemForTypelIdentifier:options:completionHandler: 使用。
loadItemForTypelIdentifier:options:completionHandler:	加载typelIdentifier指定的资源。加载是一个异步过程，加载完成后会触发completionHandler。
loadPreviewImageWithOptions:completionHandler:	加载资源的预览图片。



下面举出分享Url和分享文件的获取数据的代码

```

[self.extensionContext.inputItems enumerateObjectsUsingBlock:^(NSExtensionItem * _Nonnull obj, NSUInteger idx, BOOL * _Nonnull stop) {

    weakSelf.shareItemTitle.text = [self isNilOrZeroString:obj.attributedContentText.string] ? @"来自网页分享的一条消息" : obj.attributedContentText.string;

    [obj.attachments enumerateObjectsUsingBlock:^(NSItemProvider * _Nonnull itemProvider, NSUInteger idx, BOOL * _Nonnull stop) {

        if ([itemProvider hasItemConformingToTypeIdentifier:@"public.url"])
        {
            [itemProvider loadItemForTypeIdentifier:@"public.url" options:nil completionHandler:^(id<NSSecureCoding> _Nullable item, NSError * _Nullable error) {

```

```

        if ([[NSObject *])item isKindOfClass:[NSURL class]])
        {
            dispatch_async(dispatch_get_main_queue(), ^{
                NSString *linkString = ((NSURL *)item).absoluteString
            };

            if ([((NSURL *)item).absoluteString hasPrefix:@"file://"])
            {
                NSString *fileName = [self getFileName:linkString];
                NSUserDefaults *userDefaults = [[NSUserDefaults alloc] initWithSuiteName:@"group.com.favalue.stucy"];
                NSDictionary *dic = [userDefaults objectForKey:@"FileInfo"];
                NSString *fileRealName;
                if ([dic allKeys].count != 0)
                {
                    fileRealName = [dic objectForKey:fileName];
                }

                weakSelf.shareItemTitle.text = [self isNilOrZeroString:fileRealName] ? [fileName stringByReplacingPercentEscapesUsingEncoding:NSUTF8StringEncoding] : fileRealName;

                weakSelf.shareItemFavicon.contentMode = UIViewContentModeScaleAspectFill;
                weakSelf.shareItemUrl.text = ((NSURL *)item).absoluteString;
                weakSelf.shareItemUrl.hidden = YES;
                [weakSelf.shareItemFavicon setImage:[self isNilOrZeroString:fileRealName] ? [weakSelf getImage:fileName] : [weakSelf getImage:fileRealName]];
                weakSelf.shareType = FileType;
                [weakSelf createCardWithLinkString:((NSURL *)item).absoluteString];
            }
        }
    }
}

```

```
        NSURL *url = [weakSelf getFaviconURL:(NSURL *)item].absoluteString];
        NSURLRequest *request = [NSURLRequest requestWithURL:url];
        NSURLSession *session = [NSURLSession sessionWithConfiguration:[NSURLSessionConfiguration ephemeralSessionConfiguration]];
        NSURLSessionDownloadTask *task = [session downloadTaskWithRequest:request completionHandler:^(NSURL *location, NSURLResponse *response, NSError *error) {
            NSLog(@"%@", location);
            NSLog(@"home %@", NSHomeDirectory());
            dispatch_async(dispatch_get_main_queue(), ^{
                if (error==nil) {
                    weakSelf.shareItemFavicon.image = [UIImage imageWithData:[NSData dataWithContentsOfURL:location]];
                }else{
                    weakSelf.shareItemFavicon.image = [UIImage imageNamed:@"articleIcon"];
                }
                weakSelf.shareItemUrl.text = ((NSURL *)item).absoluteString;
                weakSelf.shareItemUrl.hidden = NO;
                weakSelf.shareType = URLType;
                [weakSelf createCardWithLinkString:(NSURL *)item].absoluteString];
            });
        }];
        [task resume];
    }
});
```

```

        }else if ([itemProvider hasItemConformingToTypeIdentifier:@"public.pl
ain-text"]){
            [itemProvider loadItemForTypeIdentifier:@"public.plain-text" opti
ons:nil completionHandler:^(id<NSSecureCoding> _Nullable item, NSError * _Null_u
nspecified error) {
                if ([(NSObject *)item isKindOfClass:[NSString class]])
                {
                    dispatch_async(dispatch_get_main_queue(), ^{
                        NSString *linkString = (NSString *)item;
                        NSArray *linkArray = [self matchLinksWithString:linkS
tring];
                        if (linkArray.count > 0) {
                            NSURL *url = [weakSelf getFaviconURL:((NSURL *)it
em).absoluteString];
                            NSURLRequest *reque = [NSURLRequest requestWithUR
L:url];
                            NSURLSession *session = [NSURLSession sessionWith
Configuration:[NSURLSessionConfiguration ephemeralSessionConfiguration]];
                            NSURLSessionDownloadTask *task = [session downloa
dTaskWithRequest:reque completionHandler:^(NSURL *location, NSURLResponse *respon
se, NSError *error) {
                                NSLog(@"%@", location);
                                NSLog(@"home %@", NSHomeDirectory());
                                dispatch_async(dispatch_get_main_queue(), ^{
                                    if (error==nil) {
                                        weakSelf.shareItemFavicon.image = [UI
Image imageWithData:[NSData dataWithContentsOfURL:location]];
                                    }else{
                                        weakSelf.shareItemFavicon.image = [UI
Image imageNamed:@"articleIcon"];
                                    }
                                });
                            }];
                            [task resume];
                        }
                    }
                }
            }];
        }
    }
}

```

```

        [self createCardWithLinkString:linkArray[0]];
        *stop = YES;
    }
}
];
}];

}else {
    // 未知类型
    [self unsupportUrl];
}

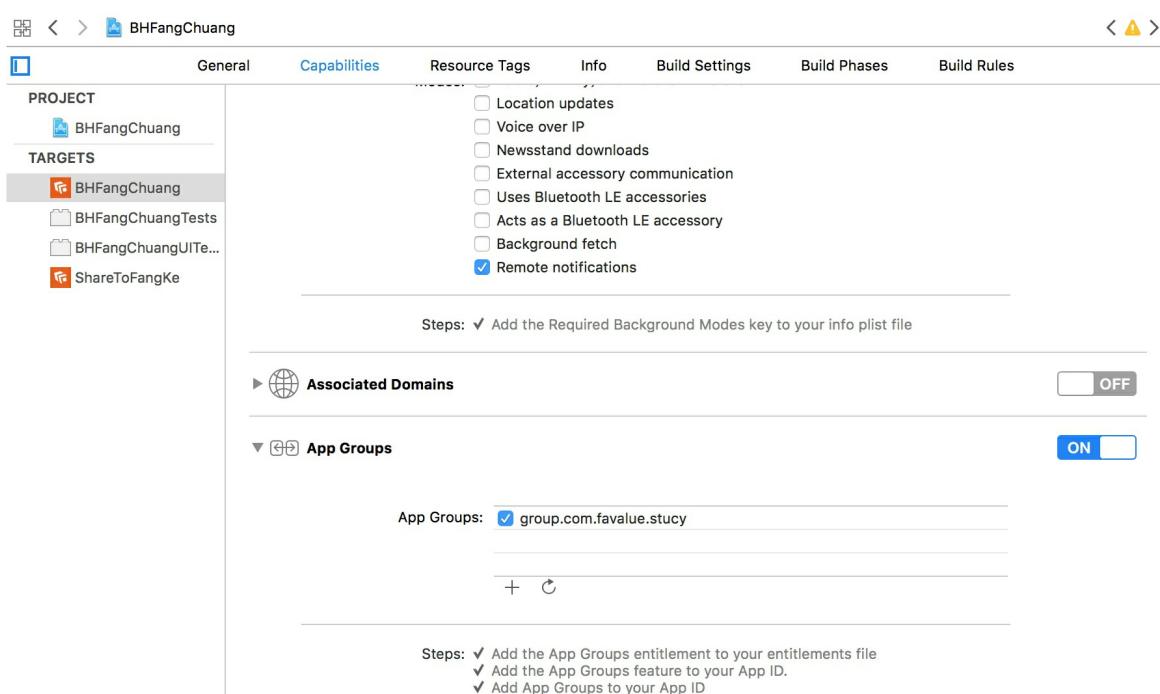
];
};

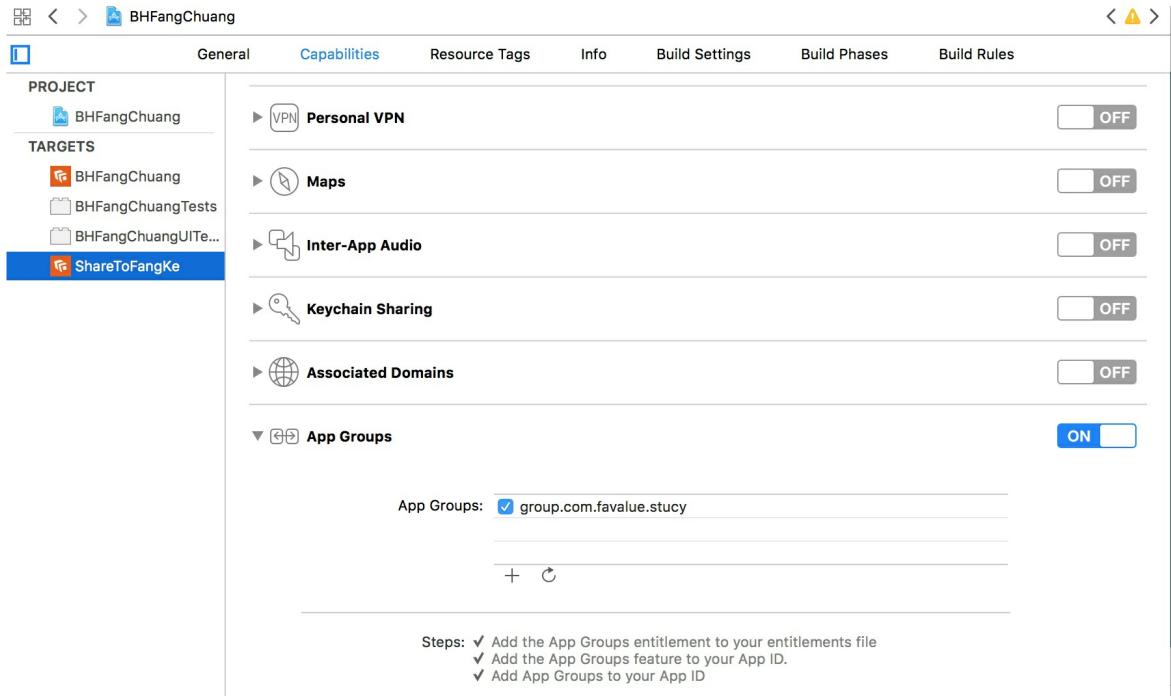
];
}

```

5. 将分享数据传递给容器程序

首先需要开启App Groups服务





要把宿主的target和新建的extension的target都添加App Groups服务，extension和主app的target要一致。

要获取到主app里面的数据主要有3种方式。NSUserDefaults、NSFileManager以及CoreData。下面介绍NSUserDefaults的方法，其他两种方法可以查看网络资料。

NSUserDefaults: 要想设置或访问Group的数据，不能在使用standardUserDefaults方法来获取一个NSUserDefaults对象了。应该使用initWithSuiteName:方法来初始化一个NSUserDefaults对象，其中的SuiteName就是创建的Group的名字，然后利用这个对象来实现，跨应用的数据读写。

首先在原app里面存储数据：

```
NSUserDefaults *userDefaults = [[NSUserDefaults alloc] initWithSuiteName:@"group.com.favalue.stucy"];

// 为extension写入用户登录数据
if ([[BHUser getCurrentUser].accid smk_isNotEmpty]) {
    [userDefaults setValue:[BHUser getCurrentUser].accid forKey:@"userAcc
id"];
}
```

在extension里面取数据

```
NSUserDefaults *userDefaults = [[NSUserDefaults alloc] initWithSuiteName:@"group.com.favalue.stucy"];  
  
NSDictionary *dic = [userDefaults objectForKey:@"FileInfo"];
```

6. 提交Appstore需要注意的问题

1. 提审的扩展和容器程序的Build Version要保持一致，否则在上传审核包的时候会提示警告，导致程序无法正常提审。
2. 一定要详细描述extension里面NSExtension的字典，否则会出现下面的错误。

```
ERROR ITMS-90362: "Invalid Info.plist value. The value for the key 'NSExtensionActivationRule' in bundle BHFFangChuang.app/PlugIns/ShareToFangKe.appex is invalid. Please refer to the App Extension Programming Guide on https://developer.apple.com"
```

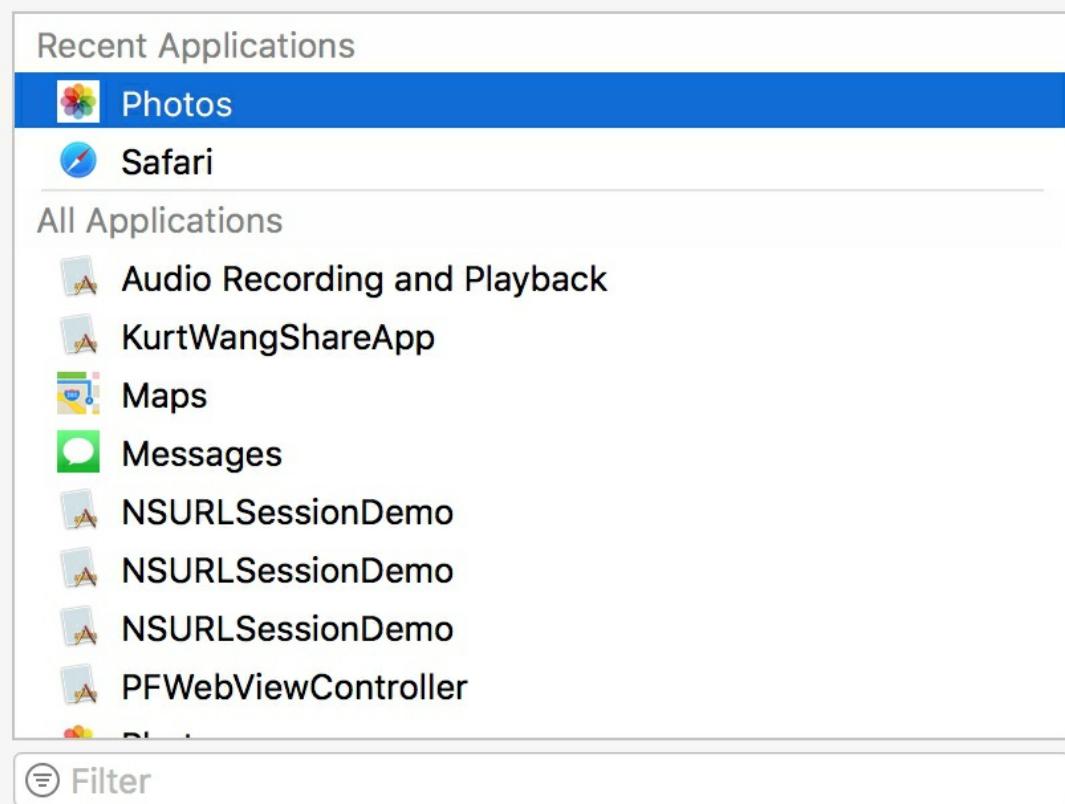
▼ NSExtension	Dictionary	(3 items)
▼ NSExtensionAttributes	Dictionary	(1 item)
▼ NSExtensionActivationRule	Dictionary	(3 items)
NSExtensionActivationSuppo...	Number	1
NSExtensionActivationSuppo...	Number	0
NSExtensionActivationSuppo...	Number	1
NSExtensionMainStoryboard	String	MainInterface
NSExtensionPointIdentifier	String	com.apple.share-services

三. 关于调试

不管是真机调试还是模拟器调试，调试的时候一定要记得选择正确“目标”。

比如说要调试Safari的分享，那么就应该运行extension，在运行之后的这个弹框中：

Choose an app to run:



Cancel

Run

选择要分享出来的程序。选择好了以后，手机就会调起这个程序，之后就可以在分享的extension里面加断点调试了。

一般容易出现的错误是运行了主app，发现extension里面不走断点，这是因为运行错了程序。

最后，在extension的target里面是可以使用pod的，只不过需要设置正确的profile文件。