WILEY

整合脸书并创建一个单登陆应用

让应用登陆任何社交媒体帐户

让你的应用整合脸书

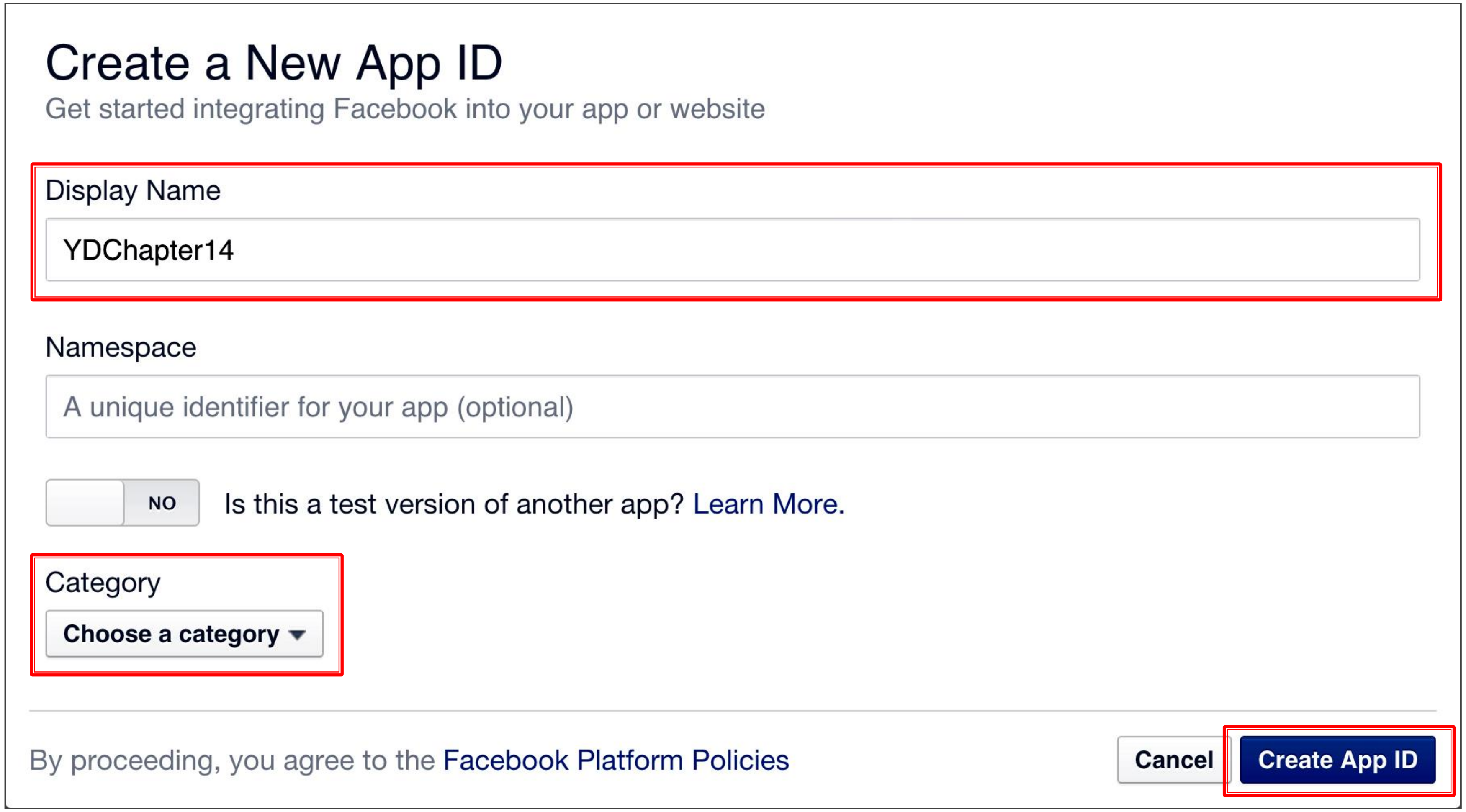
创建一个单登陆应用

让应用整合脸书

- 在 *http://developers.facebook.com* 上设置你的应用
- 点 **My Apps** 菜单，选择 **Add a New App**

让应用整合脸书

- 一个App ID和一个App Secret
- 保存这些，用于未来引用



Create a New App ID

Get started integrating Facebook into your app or website

Display Name: YDChapter14

Namespace: A unique identifier for your app (optional)

NO Is this a test version of another app? Learn More.

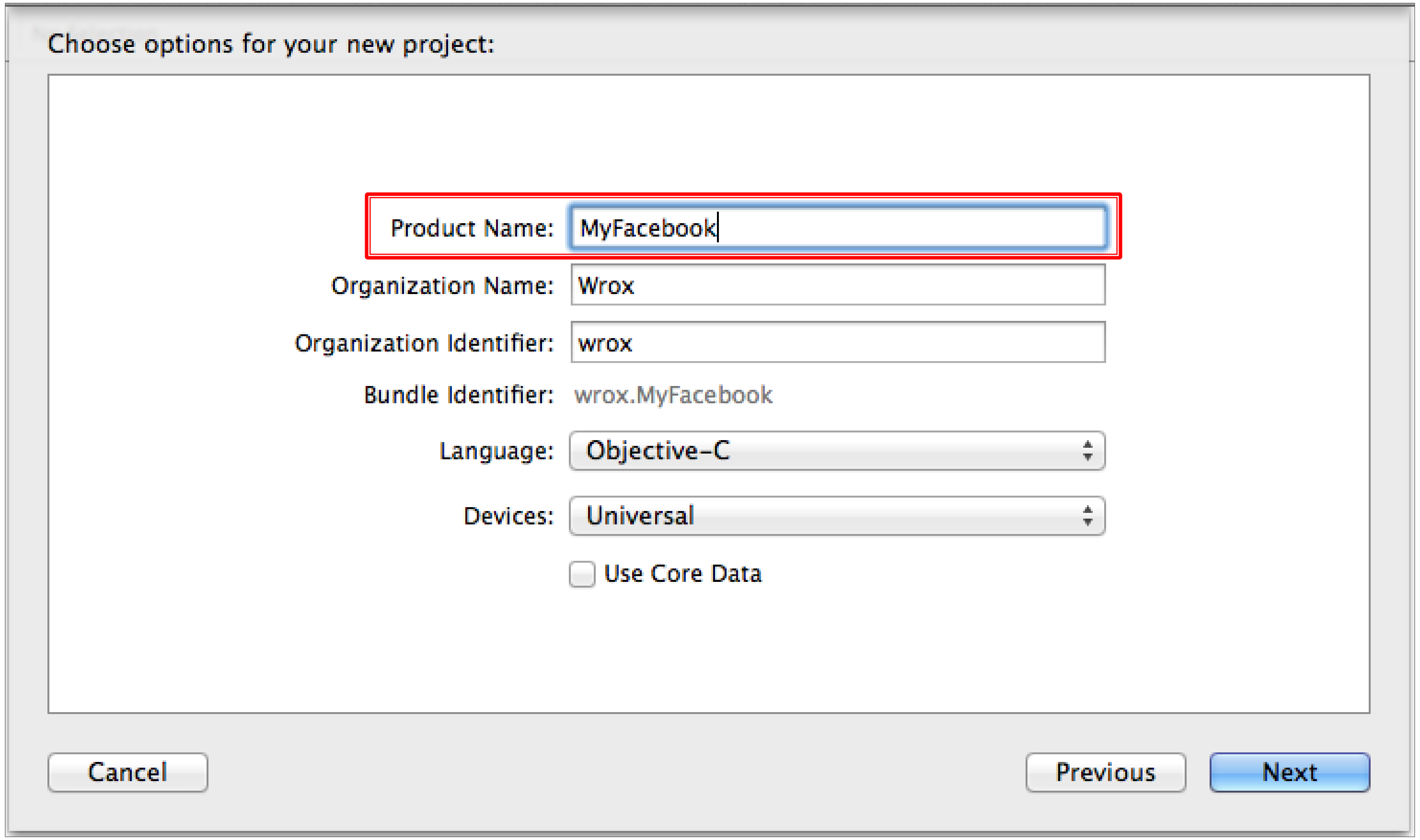
Category: Choose a category ▾

By proceeding, you agree to the Facebook Platform Policies

让应用整合脸书

- 点仪表板中的 **Getting Started** 并选择 **iOS** 作为平台
- 下载 **Facebook SDK for iOS**

让应用整合脸书



Choose options for your new project:

Product Name: MyFacebook

Organization Name: Wrox

Organization Identifier: wrox

Bundle Identifier: wrox.MyFacebook

Language: Objective-C

Devices: Universal

□ Use Core Data

让应用整合脸书

- 添加下载的 **Facebook SDK** 到你的 **Xcode** 项目；将 **FBSDKCoreKit.Framework**、**FBSDKLoginKit.Framework** 和 **FBSDKShareKit.Framework** 文件夹拖到你的Xcode 项目框架文件夹中。

- 取消选择 **Copy into destination, group folder** 以确保只有选择的文件夹，而非整组文件夹都被复制到目标文件夹。

- 找到 Xcode 项目中 **Supporting Files** 文件夹中的 **info.plist** 文件并进行配置。

让应用整合脸书

- 在 .plist 文件中创建一个 **String** 类型的键，名为 **FacebookAppID**，将你的 **AppID** 作为字符串复制到值字段。

- 创建一个 **String** 类型的键，名为 **FacebookDisplayName**，将你的**应用名YDChapter 14** 作为字符串复制到值字段。

- 创建一个 **Array** 类型的键，名为 **URL types**，带有一个字典作为其唯一的成员。

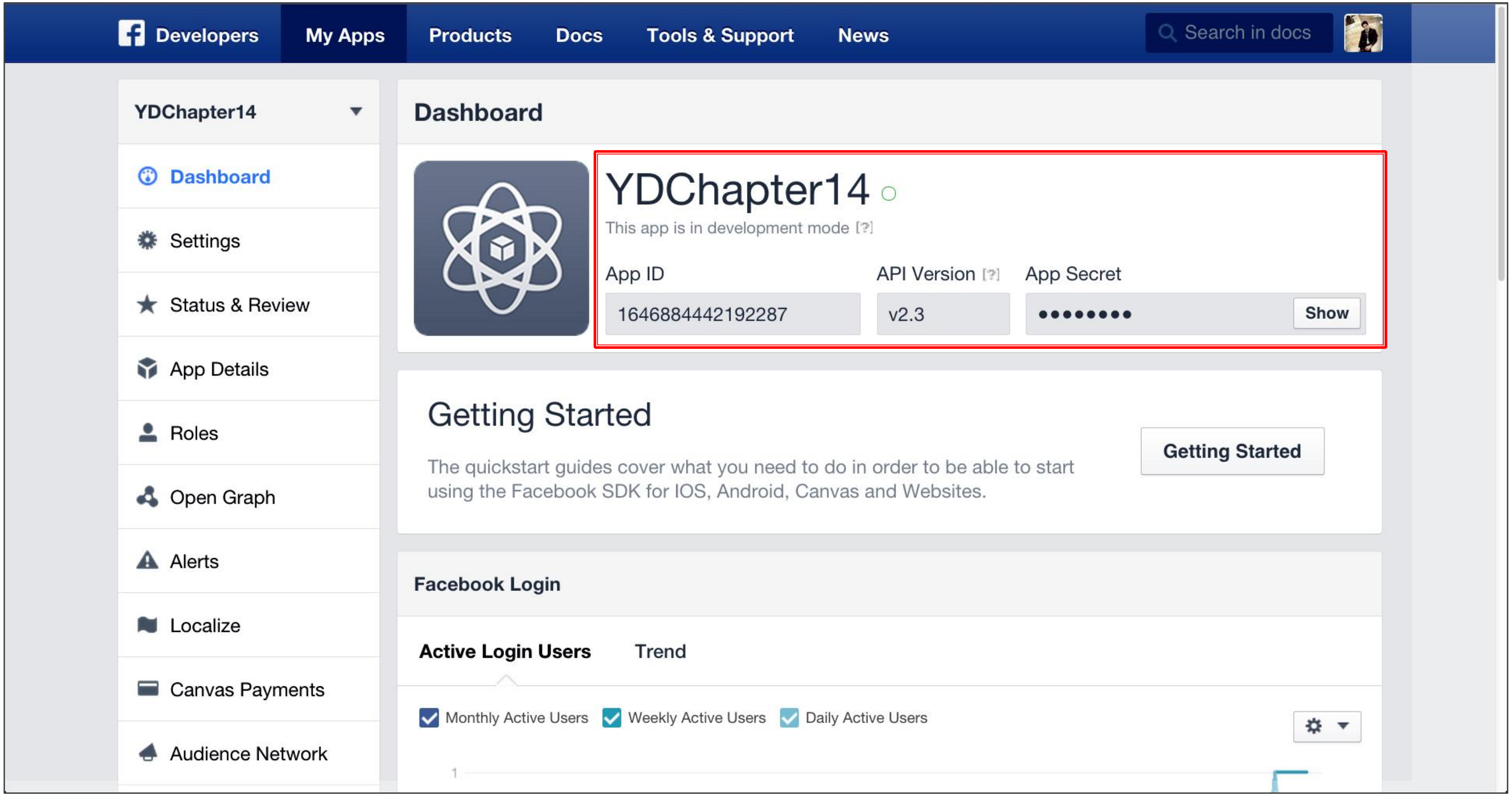
- 这个字典只需要一个数组作为其成员，名为 **URL Schemes**。这个数组的成员是一个字符串，值为 **fb<AppID>**。

- 配置 info.plist 文件后，你需要为脸书提供应用的 **Bundle ID**，形如 net.your-developer.MyFacebook。

让应用整合脸书

- 回到**脸书**
- 你需要传递**一个验证码**，然后你会看到一个页面，通过它可以配置你的应用
- 选择 **Native iOS App** 并从你的应用输入 **Bundle ID**

让应用整合脸书



Dashboard

YDChapter14 ○

This app is in development mode [?]

App ID	API Version [?]	App Secret
1646884442192287	v2.3	●●●●●●●● Show

Getting Started

The quickstart guides cover what you need to do in order to be able to start using the Facebook SDK for IOS, Android, Canvas and Websites.

Getting Started

Facebook Login

Active Login Users Trend

☑ Monthly Active Users ☑ Weekly Active Users ☑ Daily Active Users

让应用整合脸书

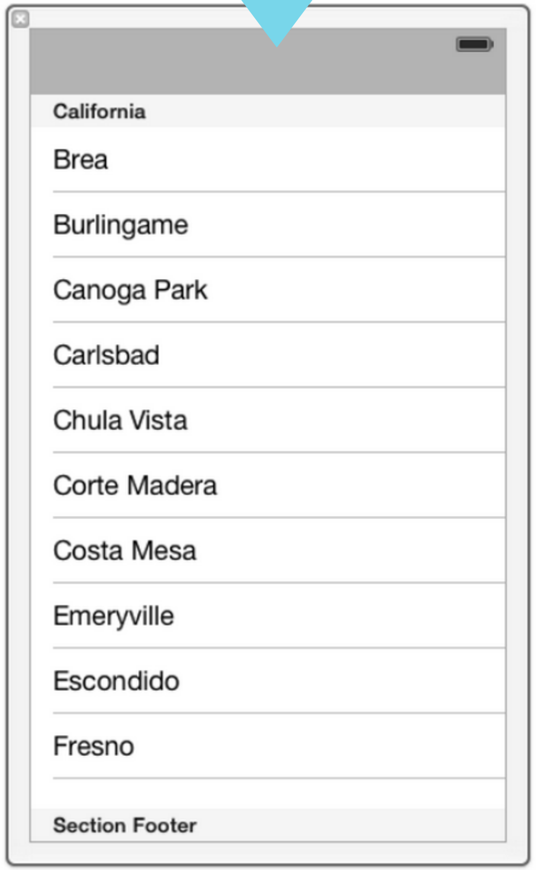
1. 打开 YDAppDelegate.h 文件并创建一个 **navigation controller**

2. 定义一个 **UINavigationController** 类型的强属性，名为 **navController**

3. 打开 YDAppDelegate.m 文件并更改 **application:didFinishWithLaunchingOptions:** 方法，来使用 navController， 以成为 **rootViewController**

4. 使用界面生成器和辅助编辑器打开 YDViewController.xib 文件，并使用 **UITableView** 创建一个简单用户界面

Table Created Using UITableView Object



让应用整合脸书

```
#import <UIKit/UIKit.h>
#import <Social/Social.h>
#import <Accounts/Accounts.h>
@interface YDViewController : UIViewController
@property (weak, nonatomic) IBOutlet UITableView *mTableView;
@property(nonatomic,strong) NSArray* facebookAccounts;
@property(nonatomic,strong) NSMutableArray* posts;
@property(nonatomic,strong) ACAccountStore* accountStore;
@property(nonatomic,strong) ACAccountType* facebookAccountType;
@property(nonatomic,strong) NSString* facebookUserID;
@property(nonatomic,strong) NSString* accessToken;
@end
```

让应用整合脸书

```
ACAccount* currentAccount =
[self.facebookAccounts objectAtIndex:0];
 NSDictionary* fbPost =
@{@"access_token":self.accessToken,@"message":
 @"My first FBpost using the SLRequest class"};
 NSURL* requestURL = [NSURL
 URLWithString:[NSString stringWithFormat:

@"https://graph.facebook.com/%@/feed",self.face
bookUserID]];
SLRequest* postRequest = [SLRequest
requestForServiceType:SLServiceTypeFacebook
requestMethod:SLRequestMethodPOST
URL:requestURL parameters:fbPost];
 postRequest.account = currentAccount;
 [postRequest
performRequestWithHandler:requestHandler];
}
else
 {
 UIAlertView* alert = [[UIAlertView alloc]
 initWithTitle:@"Error"
 message:@"Configure a Facebook account"
 delegate:self cancelButtonTitle:@"OK"
 otherButtonTitles:nil, nil];
 [alert show];
}
}
```

让应用整合脸书

```
  (void)requestAccessToAccountStore
{
 self.accountStore = [[ACAccountStore alloc] init];
 self.facebookAccountType = [self.accountStore
 accountTypeWithAccountTypeIdentifier:
 ACAccountTypeIdentifierFacebook];
 NSDictionary* options = nil;
 [self.accountStore
requestAccessToAccountsWithType:
 self.facebookAccountType options:options
 completion:^(BOOL granted, NSError *error)
 {
 if (granted == YES)
{ // do what you need to do here
```

```
dispatch_async(dispatch_get_main_queue(),^{
//call loadAccounts here now you know the user
has granted access
 [self loadAccounts];
 });
 }
 else
 {
dispatch_async(dispatch_get_main_queue(),^{
  {
  [self showAuthorizationError];
 });
 }
 }];
}
```

让应用整合脸书

```
- (IBAction)facebookSignin:(UIButton *)sender
{
self.accountStore= [[ACAccountStore alloc] init];
 self.facebookAccountType = [self.accountStore
accountTypeWithAccountTypeIdentifier:
 ACAccountTypeIdentifierFacebook];
 NSDictionary *options = @{
 ACFacebookAppIdKey: @"151575221693657",
 ACFacebookPermissionsKey: @[@"email"],
 ACFacebookAudienceKey:
ACFacebookAudienceOnlyMe
 };
```

```
 [self.accountStore
requestAccessToAccountsWithType:
 self.facebookAccountType options:options
 completion:^(BOOL granted, NSError *error)
 {
 if (granted == YES)
 {
 // Call the loadAccounts and the loadWallPosts
methods here.
dispatch_async(dispatch_get_main_queue(),^{

 if ([self.facebookAccountType accessGranted])
 }
```

让应用整合脸书

```
NSArray* facebookAccounts = [[NSArray alloc]
 initWithArray:[self.accountStore
accountsWithAccountType:self.facebookAccount
Type]];
self.currentAccount = [facebookAccounts
objectAtIndex:0];
[self signedIn];
}
```

```
else
{
 [self showAuthorizationError];
 }
 });
 }
 else
 {
dispatch_async(dispatch_get_main_queue(),^{
 [self showAuthorizationError];
 });
 }
 }];
```

让应用整合脸书

```
-(void)requestAccessToAccountStore
{
self.accountStore = [[ACAccountStore alloc] init];
self.facebookAccountType = [self.accountStore
accountTypeWithAccountTypeIdentifier:
ACAccountTypeIdentifierFacebook];
NSDictionary *options = @{
ACFacebookAppIdKey: @"151575221693657",
ACFacebookPermissionsKey: @[@"email"],
ACFacebookAudienceKey:
ACFacebookAudienceFriends
};
[self.accountStore requestAccessToAccountsWithType:
self.facebookAccountType options:options
completion:^(BOOL granted, NSError *error)
```

```
 {
if (granted == YES)
{
// do what you need to do here
dispatch_async(dispatch_get_main_queue(),^{//call
loadAccounts here now you know the user has granted
access
[self loadAccounts];
});
}
else
{
dispatch_async(dispatch_get_main_queue(),^{
[self showAuthorizationError];
});
}
}];
}
```

让应用整合脸书

```
-(void)requestAdditionalPermissions
{
self.accountStore = [[ACAccountStore alloc] init];
self.facebookAccountType = [self.accountStore
accountTypeWithAccountTypeIdentifier:
ACAccountTypeIdentifierFacebook];
NSDictionary *options = @{
ACFacebookAppIdKey: @"151575221693657",
ACFacebookPermissionsKey: @[@"publish_stream“],
ACFacebookAudienceKey:
ACFacebookAudienceFriends
};
[self.accountStore requestAccessToAccountsWithType:
self.facebookAccountType options:options
completion:^(BOOL granted, NSError *error)
```

```
{if (granted == YES)
{ // do what you need to do here
/*
dispatch_async(dispatch_get_main_queue(),^{
//call loadAccounts here now you know the user has
granted access
[self loadAccounts];
});
*/
}
else
{dispatch_async(dispatch_get_main_queue(),^{
[self showAuthorizationError];
});
}
}];
}
@end
}
```

让应用整合脸书

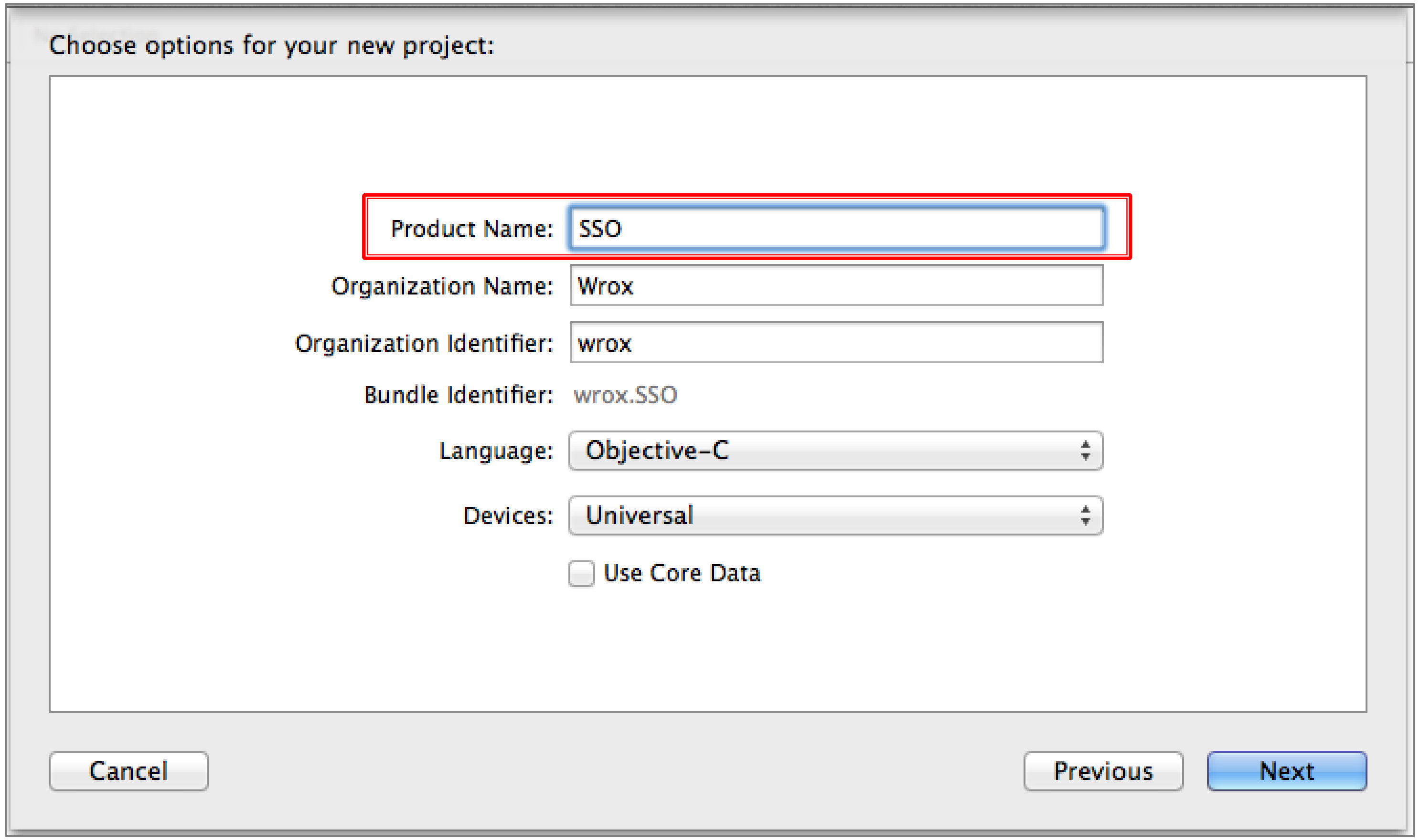
```
- (void)newPost:(id)sender
{//check if there are facebook accounts
if ([self.facebookAccounts count] >0)
{
SLRequestHandler requestHandler =
^(NSData *responseData, NSHTTPURLResponse
*urlResponse, NSError *error) {
if (responseData) {
NSInteger statusCode = urlResponse.statusCode;
if (statusCode >= 200 && statusCode < 300) {
//SUCCESS
}
else {//Error
}
}
else {//Error
}
};
```

```
ACAccount* currentAccount = [self.facebookAccounts
objectAtIndex:0];
NSDictionary* fbPost =
@{@"access_token":self.accessToken,@"message":
@"My first FBpost using the SLRequest class"};
NSURL* requestURL = [NSURL
URLWithString:[NSString stringWithFormat:
@"https://graph.facebook.com/%@/feed",self.facebook
UserID]];
```

让应用整合脸书

```
SLRequest* postRequest = [SLRequest
requestForServiceType:SLServiceTypeFacebook
requestMethod:SLRequestMethodPOST
URL:requestURL parameters:fbPost];
postRequest.account = currentAccount;
[postRequest performRequestWithHandler:requestHandler];
}
else
{ UIAlertView* alert = [[UIAlertView alloc]
initWithTitle:@"Error"
message:@"Configure a Facebook account"
delegate:self cancelButtonTitle:@"OK"
otherButtonTitles:nil, nil];
[alert show];
}
}
```

创建一个单登陆应用



Choose options for your new project:

Product Name: SSO

Organization Name: Wrox

Organization Identifier: wrox

Bundle Identifier: wrox.SSO

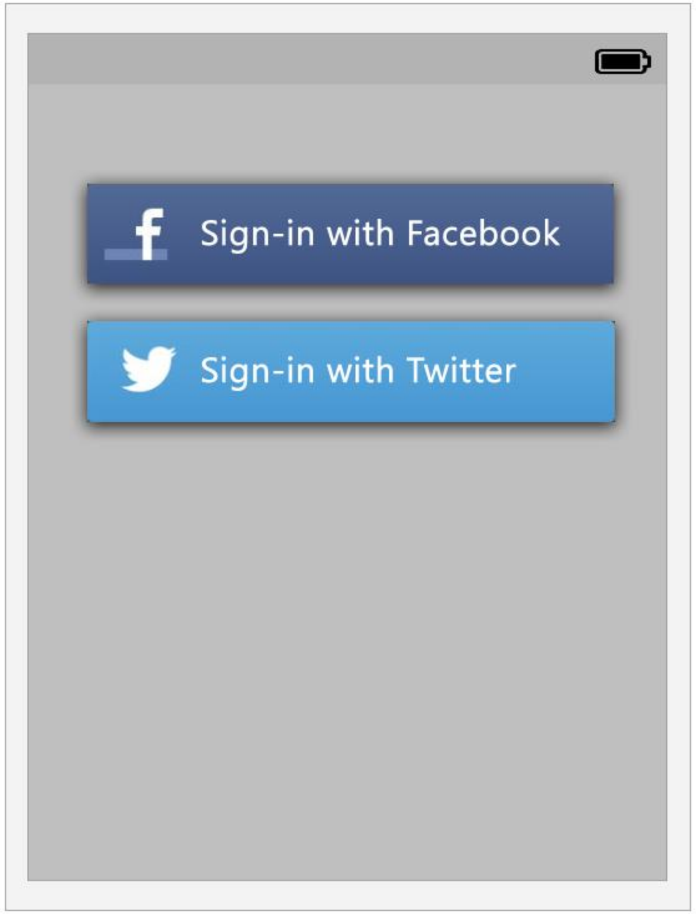
Language: Objective-C

Devices: Universal

□ Use Core Data

创建一个单登陆应用

- 使用界面生成器和辅助编辑器打开YDViewController.xib文件
- 创建一个简单用户界面，带有两个UIButton对象



创建一个单登陆应用

```
  import <UIKit/UIKit.h>
#import <Accounts/Accounts.h>
@interface YDViewController : UIViewController
- (IBAction)facebookSignin:(UIButton *)sender;
- (IBAction)twitterSignin:(UIButton *)sender;
@property(nonatomic,strong) ACAccountStore* accountStore;
@property(nonatomic,strong) ACAccount* currentAccount;
@property(nonatomic,strong) ACAccountType* twitterAccountType;
@property(nonatomic,strong) ACAccountType* facebookAccountType;
@property(nonatomic,strong) ACAccountType* selectedSigninAccountType;
@end
```

创建一个单登陆应用

```
#import "YDViewController.h"
@interface YDViewController ()
@end
@implementation YDViewController
- (void)viewDidLoad
{[super viewDidLoad];
// Do any additional setup after loading the view, typically
from a nib.}
- (IBAction)facebookSignin:(UIButton *)sender
{self.accountStore= [[ACAccountStore alloc] init];
self.facebookAccountType = [self.accountStore
accountTypeWithAccountTypeIdentifier:
ACAccountTypeIdentifierFacebook];
NSDictionary *options = @{
ACFacebookAppIdKey: @"151575221693657",
ACFacebookPermissionsKey: @[@"email"],
ACFacebookAudienceKey: ACFacebookAudienceOnlyMe
};
[self.accountStore requestAccessToAccountsWithType:
self.facebookAccountType options:options
completion:^(BOOL granted, NSError *error)
```

```
{if (granted == YES)
{// do what you need to do here
dispatch_async(dispatch_get_main_queue(),^{
if ([self.facebookAccountType accessGranted])
{NSArray* facebookAccounts = [[NSArray alloc]
initWithArray:[self.accountStore
accountsWithAccountType:self.facebookAccountType]];
self.currentAccount = [facebookAccounts objectAtIndex:0];
[self signedIn];}
else{[self showAuthorizationError];
}
});
}
else{dispatch_async(dispatch_get_main_queue(),^{
[self showAuthorizationError];
});
}
}];
}
```

创建一个单登陆应用

```
- (IBAction)twitterSignin:(UIButton *)sender
{
self.accountStore = [[ACAccountStore alloc] init];
self.twitterAccountType = [self.accountStore
accountTypeWithAccountTypeIdentifier:
ACAccountTypeIdentifierTwitter];
[self.accountStore
requestAccessToAccountsWithType:self.twitterAccountType
options:nil completion:^(BOOL granted, NSError *error)
{
if (granted == YES)
{
// do what you need to do here
dispatch_async(dispatch_get_main_queue(),^{
if ([self.twitterAccountType accessGranted])
```

```
{NSArray* twitterAccounts = [[NSArray alloc]
initWithArray:[self.accountStore
accountsWithAccountType:self.twitterAccountType]];
self.currentAccount = [twitterAccounts objectAtIndex:0];
[self signedIn];
}
else
{[self showAuthorizationError];
}
});
}
else
{dispatch_async(dispatch_get_main_queue(),^{
[self showAuthorizationError];
});
}
}];
}
```

创建一个单登陆应用

```
-(void)signedIn
{
NSLog(@"username: %@",
self.currentAccount.username);
//you can use self.currentAccount.username to
pass to your backend system
UIAlertView* alert = [[UIAlertView alloc]
initWithTitle:@"Welcome"
message:@"You have been signed-in"
delegate:self
cancelButtonTitle:@"OK" otherButtonTitles:nil,
nil];
[alert show];
}
```

```
-(void)showAuthorizationError
{
UIAlertView* alert = [[UIAlertView alloc]
initWithTitle:@"Error"
message:@"You are not authorized"
delegate:self cancelButtonTitle:@"OK"
otherButtonTitles:nil, nil];
[alert show];
}
- (void)didReceiveMemoryWarning
{
[super didReceiveMemoryWarning];
// Dispose of any resources that can be recreated.
}
@end
```

整合社交媒体

- 在单个位置整合所有社交帐户
- 多个社交帐户
- 无需创建任何登陆凭证

从用户的脸书上取回帖子，需要用到下列哪种方法?

a) **SLRequestGet:**

b) **SLServiceTypeFacebook:**

c) **SLRequestController:**

d) **SLRequest:**

从用户的脸书上取回帖子，需要用到下列哪种方法？

a) SLRequestGet:

b) SLServiceTypeFacebook:

c) SLRequestController:

d) SLRequest:

总结

你可以在应用中创建一个单登陆模式，获得用户帐户的访问授权，而无需要求用户重新验证。

你也可以在你的应用中整合脸书，让用户能够直接从你的应用往脸书发布更新。

要将你的应用同脸书整合，你需要：

1. 在*http://developers.facebook.com*上设置脸书应用
2. 在头文件中实例化各种UI对象
3. 在头文件中导入Social和Accounts框架
4. 在实现文件中实现loadWallPosts:和newPost:方法
5. 使用SLRequestGet:方法，从用户的脸书上取回帖子

对于脸书登陆凭证，使用facebookSignin:方法

对于推特凭证，使用twitterSignin:方法

WILEY

All brand names, product names, and logos represented in this course are trademarks, registered trademarks, or trade names of their respective holders. John Wiley & Sons Inc. is not associated with any product or vendor mentioned in this course.