

实验 5



创建访问地址簿并管理其数据的应用

实验结构

- » 创建一个应用来访问地址簿并管理其数据

实验目标

本实验结束后，你将能够：

- » 创建一个可以访问地址簿的应用
- » 在地址簿中添加数据
- » 编辑地址簿中的已有数据
- » 从地址簿中删除数据

导论

iOS 中提供的地址簿功能能够在数据库中存储联系人及相关信息。这种集中式数据库中存储的信息可以在其它应用中分享。AddressBookUI 框架提供了用于访问这些存储信息的用户界面。iOS 提供的通讯录应用就是访问 iOS 数据库中这一信息的一种方式。

实验：创建访问地址簿并管理其数据的应用

背景

你开发了一款应用来提醒用户，在生日和纪念日打电话给爱人。你完成了应用的全部工作，只剩下从手机联系人名录中获取联系人的机制。你需要使用 AddressBookUI 框架添加这一功能。它能从地址簿中获取数据。地址簿是联系人信息和群组信息的集合，每个人都对应有姓、名、电话号码、电子邮箱等属性。

实验中，你需要执行下面这些任务：

1. 访问地址簿，通过你的应用选择任意联系人
2. 添加新联系人
3. 编辑和删除联系人

实验准备

要执行这些任务，你需要有：

- 带有 Xcode 的 Mac
- 苹果开发者帐户

实验推荐解决方案

1. 打开 Xcode 并选择 **Create a new Xcode project** 选项，如图 1 所示：

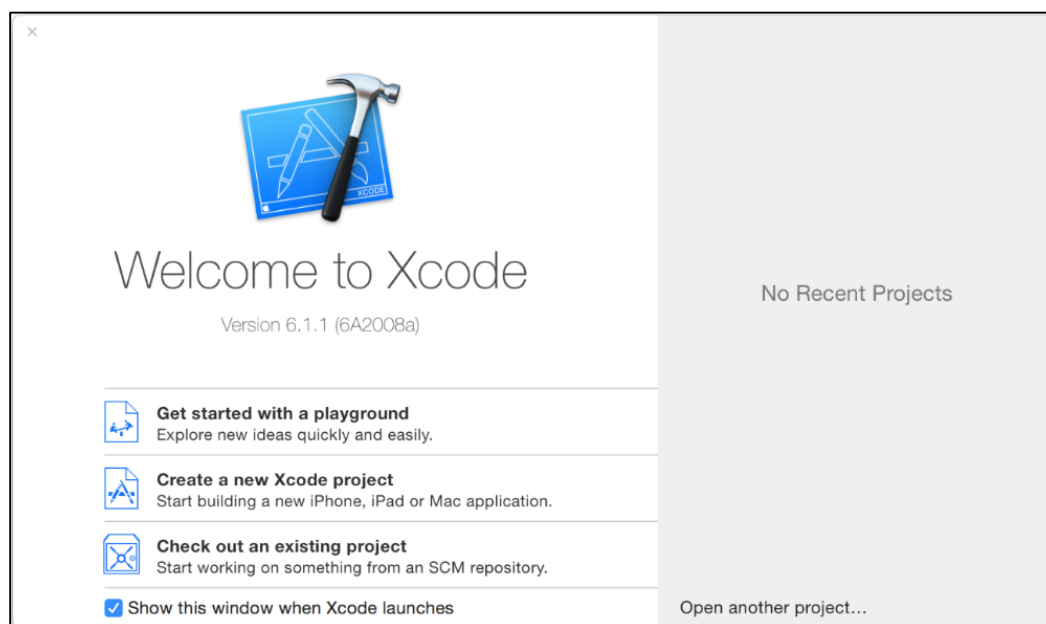


图 1: 选择创建新 Xcode 项目

2. 选择 **Single View Application** 模板，并点 **Next**，如图 2 所示：

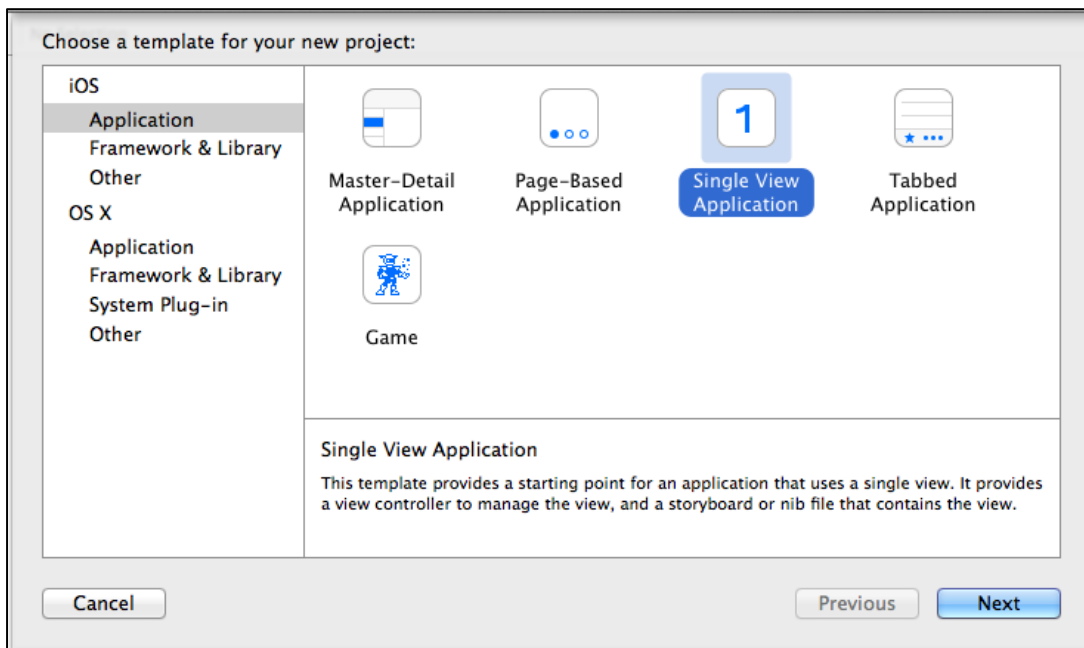


图 2：选择 Single View Application 模板

3. 下一个界面上，在 **Product Name** 文本框中键入 MyAddressBook，选择 **Swift** 作为编程语言，然后点 **Next**，如图 3 所示：

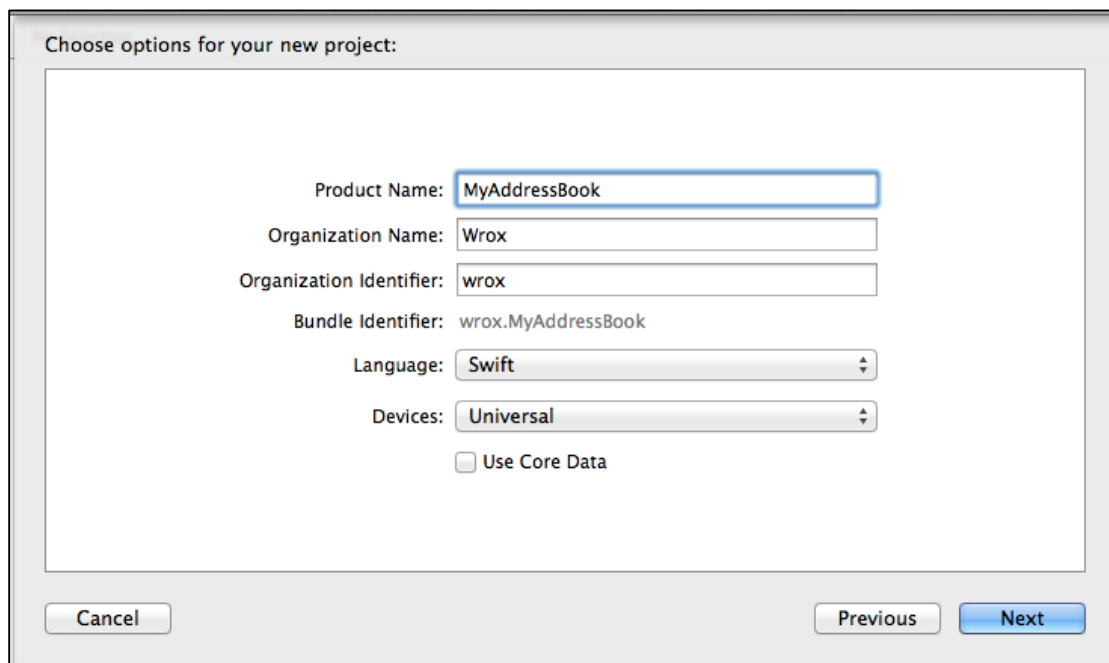


图 3：将项目命名为 MyAddressBook

4. 选择项目保存位置，然后点 **Create**，这会在 Xcode 中打开项目，如图 4 所示：

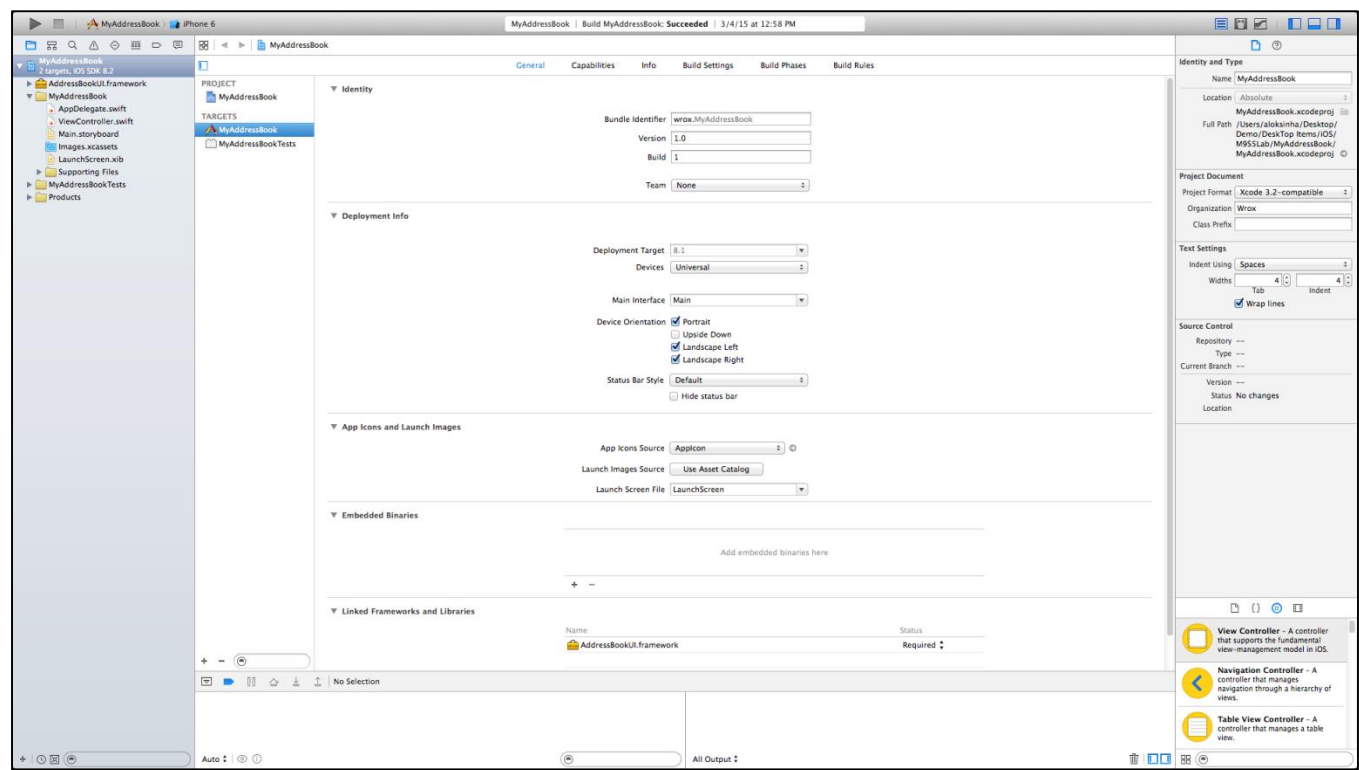


图 4: Xcode 中的 MyAddressBook 项目

- 5. Xcode 6 中无法自动获得任何核心框架。你需要在 iOS SDK 中添加 AddressBook.framework，这样才能同地址簿数据库进行交互。为此，点 **Linked Frameworks and Libraries** 部分的+图标，如图 4 所示：
- 6. 在 **Choose frameworks and libraries to add** 对话框中，选择 AddressBookUI.framework 并点 **Add**，如图 5 所示：

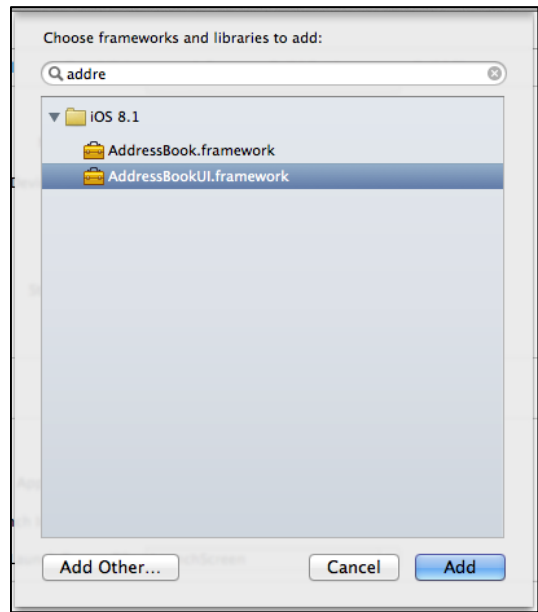


图 5: 为 MyAddressBook 选择 AddressBookUI.framework 库

- 7. 打开 Main.storyboard 文件，为 MyAddressBook 项目设计用户界面。添加标签 MyAddressBook，按钮作为动作按钮，名为 **Select Contact** 和 **Add New Contact**，如图 6 所示：

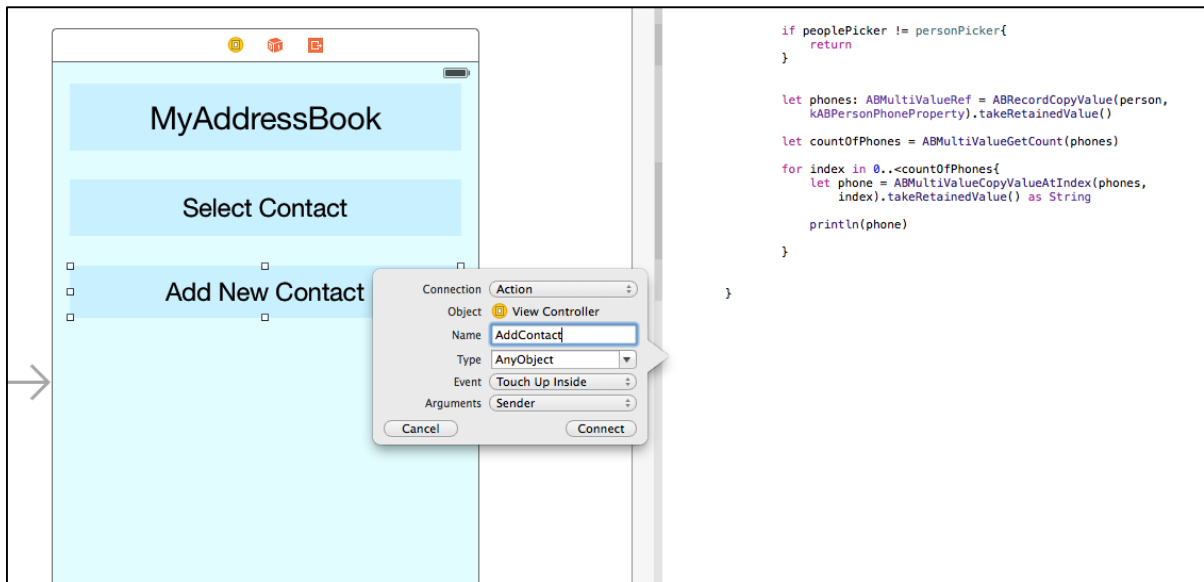


图 6: 使用标签和按钮设计简单用户界面

8. 打开 ViewController.swift 并导入 AddressBookUI，如下所示：

```
import UIKit
import AddressBook
import AddressBookUI
```

9. 苹果 iOS 不允许任何应用直接访问个人数据库。要访问用户的地址簿，你需要从 iOS 获得权限。为此，你需要创建一个用户界面来明确询问用户，是否允许应用访问数据库的特定部分，例如地址簿。要获得访问权限，需要使用 ABAddressBookGetAuthorizationStatus 函数，来向用户索取授权。为此需要在 AppDelegate.swift 中使用如下代码：

```
func application(application: UIApplication!, didFinishLaunchingWithOptions launchOptions:
[NSObject: AnyObject]?) -> Bool {
    switch ABAddressBookGetAuthorizationStatus() {
    case .Authorized:
        println("Already authorized")
        ShowContacts()
        /* Now you can use the address book */
    case .Denied:
        println("You are denied access to address book")
    case .NotDetermined:
        ShowContacts()
        if let theBook: ABAddressBookRef = addressBook {
            ABAddressBookRequestAccessWithCompletion(theBook,
                {(granted: Bool, error: CFError!) in
                    if granted {
                        println("Access granted")
                    } else {
                        println("Access not granted")
                    }
                })
        }
    }
}
```

```
    }  
    case .Restricted:  
        println("Access is restricted")  
    default:  
        println("Unhandled")  
    }  
    return true  
}
```

10. 选择 **Project** 菜单中的 **Run** 选项，或是使用 **Command+Run** 路径来运行模拟器，你会得到如图 7 所示的输出。点 **OK** 按钮提供访问。

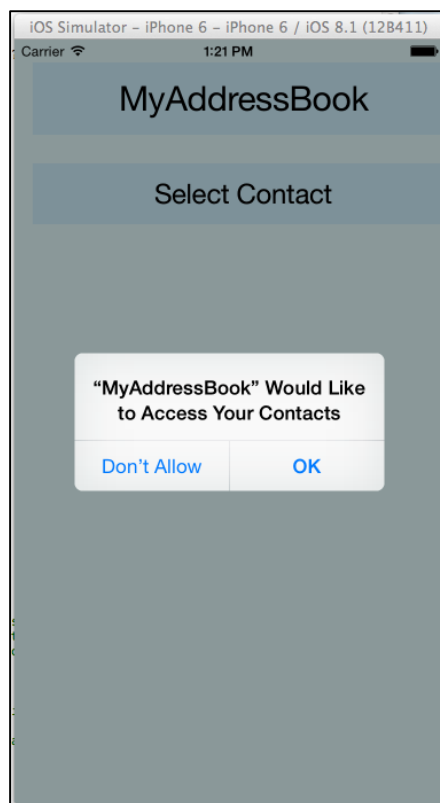


图 7: 点 OK 来提供访问

11. 在 `ViewController.swift` 中添加如下代码，让用户能够打开地址簿：

```
@IBAction func SelectContact(sender: AnyObject) {  
    let person: ABRecordRef = ABPersonCreate().takeRetainedValue()  
    self.presentViewController(personPicker, animated: true, completion: nil)  
}  
func peoplePickerNavigationControllerDidCancel(  
    peoplePicker: ABPeoplePickerNavigationController!){  
}
```

12. 运行应用并点击 **Select Contact** 按钮，如图 8 所示：

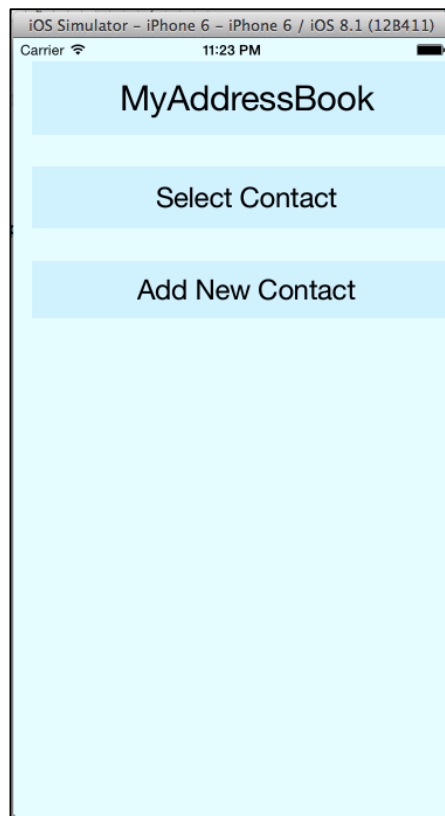


图 8：在模拟器上运行应用

13. 这样就能打开地址簿了，允许用户从地址簿中选择联系人，如图 9 所示。为此，你需要在类 `ViewController.swift` 中实现 `ABPeoplePickerNavigationControllerDelegate` 协议并添加 `peoplePickerNavigationController:didSelectPerson:property:identifier:` 方法，如下面代码所示：

```
import UIKit
import AddressBook
import AddressBookUI

class ViewController: UIViewController, ABPeoplePickerNavigationControllerDelegate,
ABNewPersonViewControllerDelegate {
    let personPicker: ABPeoplePickerNavigationController
    required init(coder aDecoder: NSCoder) {
        personPicker = ABPeoplePickerNavigationController()
        super.init(coder: aDecoder)
        personPicker.peoplePickerDelegate = self
    }
    @IBAction func SelectContact(sender: AnyObject) {
        let person: ABRecordRef = ABPersonCreate().takeRetainedValue()
        self.presentViewController(personPicker, animated: true, completion: nil)
    }
    func peoplePickerNavigationControllerDidCancel(
        peoplePicker: ABPeoplePickerNavigationController!) {
    }
}
```

```
func peoplePickerNavigationController(
    peoplePicker: ABPeoplePickerNavigationController!, didSelectPerson person:
ABRecordRef!, property: ABPropertyID,
    identifier: ABMultiValueIdentifier){
    if peoplePicker != personPicker{
        return
    }
    let phones: ABMultiValueRef = ABRecordCopyValue(person,
        kABPersonPhoneProperty).takeRetainedValue()

    let countOfPhones = ABMultiValueGetCount(phones)
    for index in 0..
```

14. 还是通过选择 **Project** 菜单中的 **Run** 选项，或是使用 Command+Run 路径来运行模拟器，你会得到如图 9 所示的输出。
选择联系人 **John Appleseed**，你会得到他的联系信息，如图 10 所示：

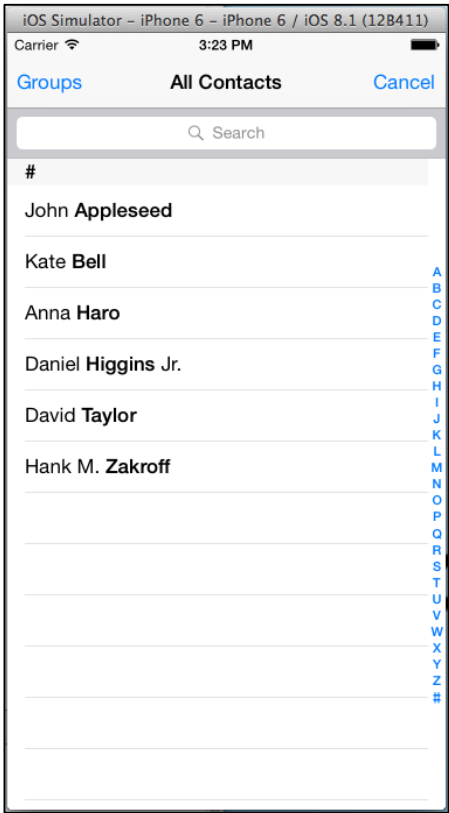


图 9: 点击 Select Contact 后的视图

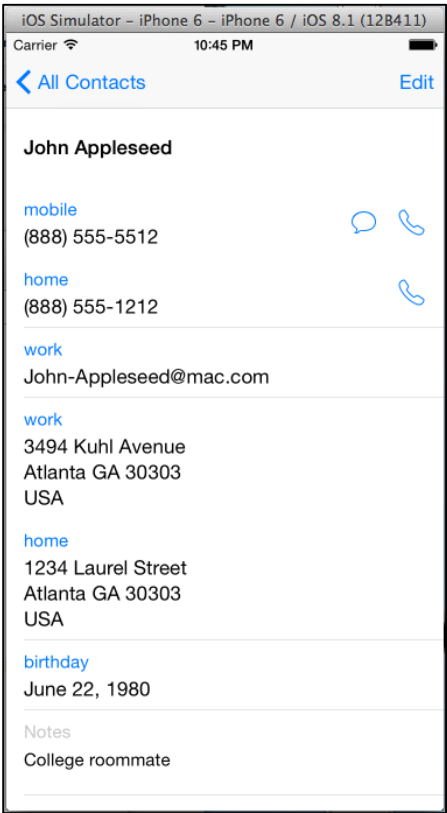


图 10: 点击第一个联系人 John Appleseed 后的视图

15. 要添加新联系人，你需要在类 ViewController 中添加 ABNewPersonViewControllerDelegate 协议，如下面代码所示。这会将代码添加到 **Add New Contact** 按钮。


```

@IBAction func AddContact(sender: AnyObject) {
    let addNew = ABNewPersonViewController()
    addNew.newPersonViewDelegate = self
    let navigationController = UINavigationController(rootViewController: addNew)
    self.presentViewController(navigationController, animated: true, completion: nil)
}

func newPersonViewController(newPersonView: ABNewPersonViewController!,
didCompleteWithNewPerson person: ABRecord!) {
    newPersonView.navigationController?.dismissViewControllerAnimated(true, completion:
nil);
}

```

16. 再次运行模拟器，你会得到如图 11 所示的输出。之后点击 **Add New Contact**，你会得到添加新联系人的界面，如图 12 所示。

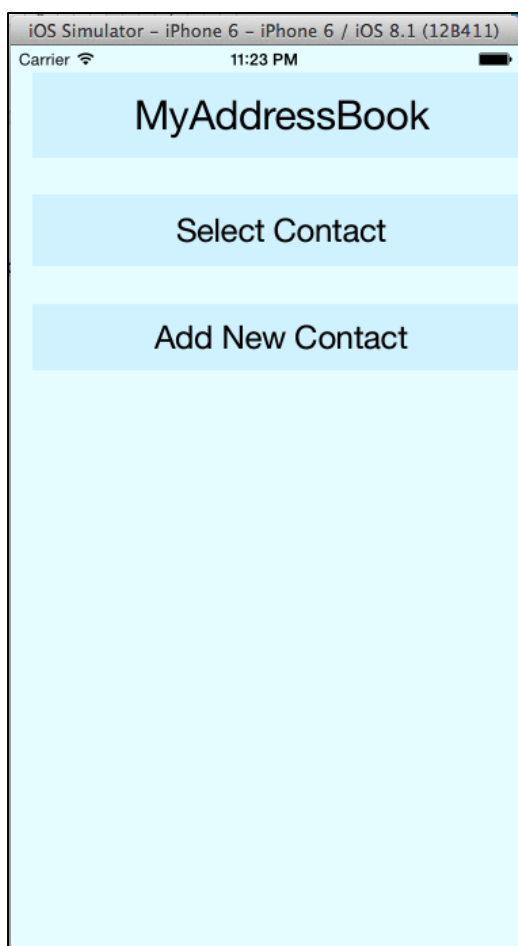


图 11: 在模拟器上运行应用

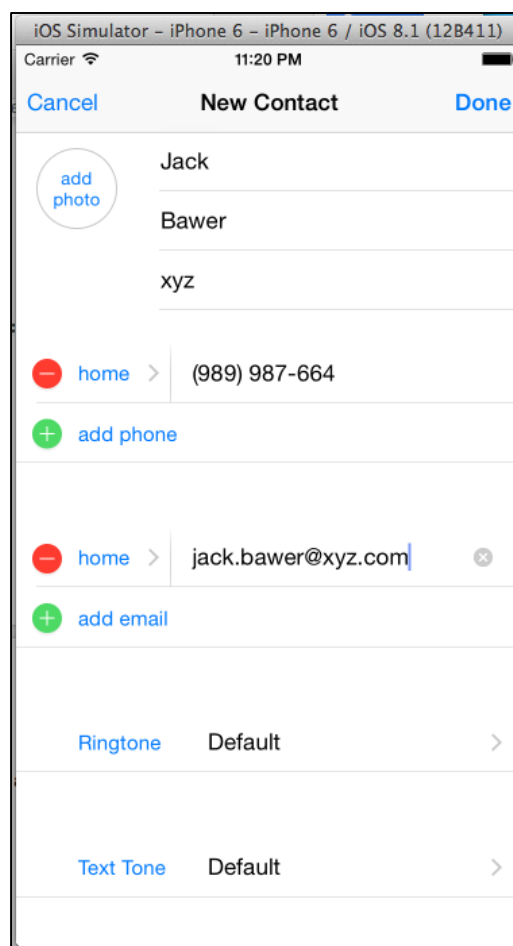


图 12: 点击 Add New Contact 后的视图

17. 要添加联系人照片，点击 **add photo** 选项，如图 12 所示，这会允许访问图库中的照片，如图 13 和 14 所示：

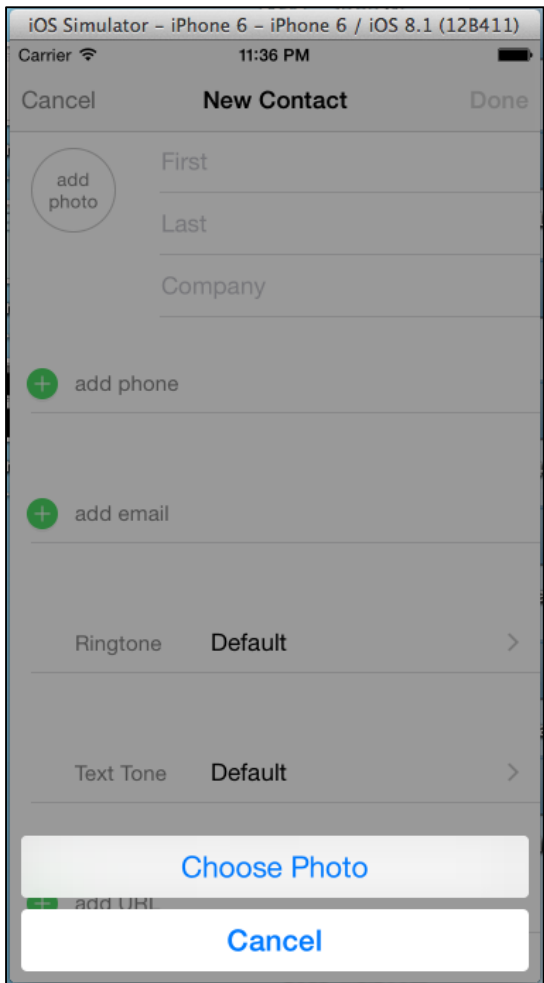


图 13: 点击 Add Photo 后的视图

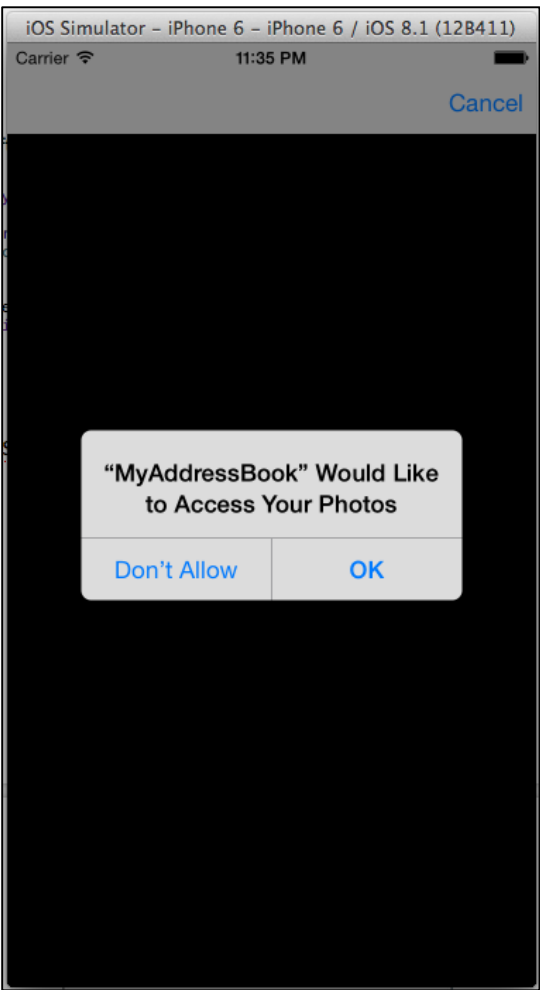


图 14: 点 OK 来允许访问

18. 要编辑和删除联系人，你需要额外添加权限，如下面代码所示：

```
ABPersonViewController *pvController = [[ABPersonViewController alloc] init];
pvController.personViewDelegate = self;
pvController.displayedPerson = person;
// Allow to edit the information
pvController.allowsEditing = YES;
[self.navigationController pushViewController:pvController animated:YES];
```