iOS9----Contact Framework

在以前iOS开发中，涉及联系人相关的编程，代码都非常繁琐，并且框架的设计也不是Objective-C风格的，这使开发者用起来非常的难受。在iOS9中，apple终于解决了这个问题，全新的Contacts Framework将完全替代AddressBookFramework,AddressBookFramework也将成为历史被弃用。至于AddressBookFramework的相关API如何繁琐，在以前的博客（OC）中有记录，地址如下：

联系人信息相关编程：[http://my.oschina.net/u/2340880/blog/407347](http://my.oschina.net/u/2340880/blog/407347" \t "_self)。

联系人UI界面相关编程：[http://my.oschina.net/u/2340880/blog/407973](http://my.oschina.net/u/2340880/blog/407973" \t "_self)。

这一新的框架是iOS9新特性中十分受欢迎的一个。apple的Objective—C体系也更加完善与强大。

**第一种：获取用户的联系人**

1. **首先我们要在头部引入 Contacts 和 ContactsUI 框架。**
2. **读取手机中的联系人并存储为数据源：**

func findContacts() -> [CNContact] {

let store = CNContactStore()

**}**

CNContactStore 是一个用来读取和保存联系人的新的类。

**3、当我们有了这个联系人数据库的引用后，我们需要创建一个指定条件的请求，通过这个 query 的请求去获取某些结果。创建一个 CNContactFetchRequest ，我们可以通过设置 contact keys 的数组，来获取我们需要的结果。我们可以通过CNContactFormatter.descriptorForRequiredKeysForStyle(.FullName) 来格式化。**

当 CNContactFormatter key 的需求发生改变，在从CNContactFormatter 生成一个字符串时，你会接到一个异常。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | var contacts = [CNContact]()  do {      try store.enumerateContactsWithFetchRequest(fetchRequest, usingBlock: { (let contact, let stop) -> Void in      contacts.append(contact)  })  }  catch let error as NSError {      print(error.localizedDescription)  }  return contacts |

我们所做的是从 CNContactStore 中遍历所有符合我们需求的联系人。这个request 没有加任何的条件，所以会返回全部的联系人，包含我们需要的 keys。我们把每一条记录都逐个保存到一个数组中，返回。形成列表展示的数据源。

**注意**

1. **由于获取联系人属于费时操作，建议放在子线程中获取回主线程刷新UI**

//开辟一条线程获取本地联系人

dispatch\_async(dispatch\_get\_global\_queue(DISPATCH\_QUEUE\_PRIORITY\_DEFAULT, 0)) { () -> Void in

self.contacts = self.findContacts()

//回主线程刷新UI（必须在主线程刷新）

dispatch\_async(dispatch\_get\_main\_queue(), { () -> Void in

self.tableView.reloadData()

})

}

**2、CNContactFormatter 需要很多不同的 keys，如果不使用 descriptorForRequiredKeysForStyle 方法，我们需要手动设置以下的 keys。**

**[CNContactGivenNameKey,**

**CNContactNamePrefixKey,**

**CNContactNameSuffixKey,**

**CNContactMiddleNameKey,**

**CNContactFamilyNameKey,**

**CNContactTypeKey...]**

**如你所见，要写一大堆代码。当 CNContactFormatter key 的需求发生改变，在从CNContactFormatter 生成一个字符串时，你会接到一个异常。所以需要**

**do{**

**try store.enumerateContactsWithFetchRequest(fetchRequest, usingBlock: { (let contact, let stop) -> Void in**

**contactsArray.append(contact)**

**})**

**}**

**catch let error as NSError {**

**print(error.localizedDescription)**

**}**

第二种：**使用 ContactsUI 选择联系人**

**也许我们希望这个应用，可以让用户自己选择联系人，并且展示详细信息给我们。正如此前你看到的，这可能要写很多代码。我们也可以使用 ContactsUI framework 的功能。他提供了一套 view controllers，我们可以用在我们的应用中，直接展示联系人的信息。**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | @IBAction func showContactsPicker(sender: UIBarButtonItem) {      let contactPicker = CNContactPickerViewController()      contactPicker.delegate = self;      contactPicker.displayedPropertyKeys = [CNContactPhoneNumbersKey]      self.presentViewController(contactPicker, animated: true, completion: nil)  } |

**我们创建了一个简单的 CNContactPickerViewController ，设置他的代理为 self.这样我们就能够响应他的请求，我们感兴趣的事电话号码，尽在选中电话号码后，展示联系人信息。CNContactPickerViewController 帮我们控制UI。**