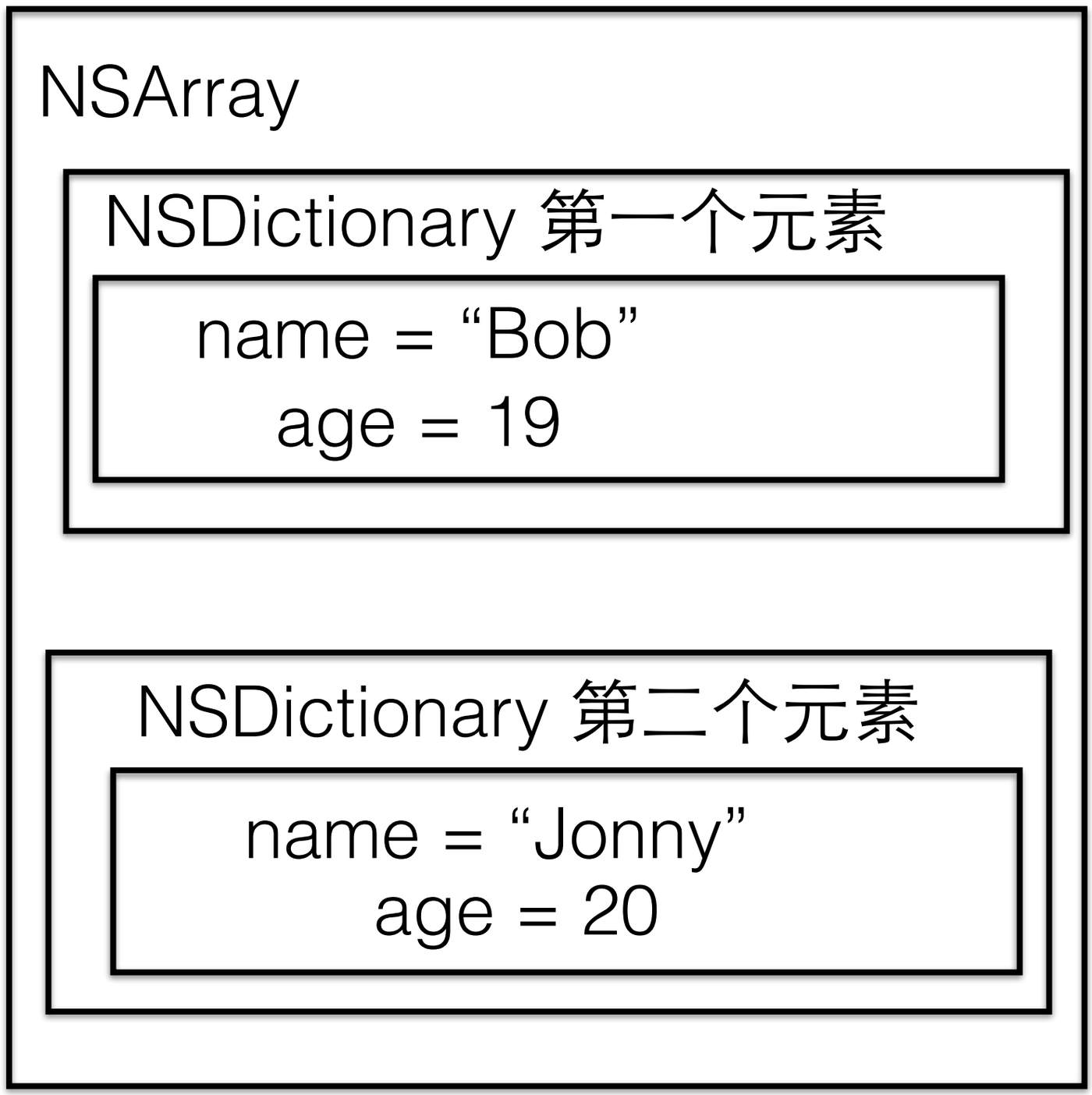
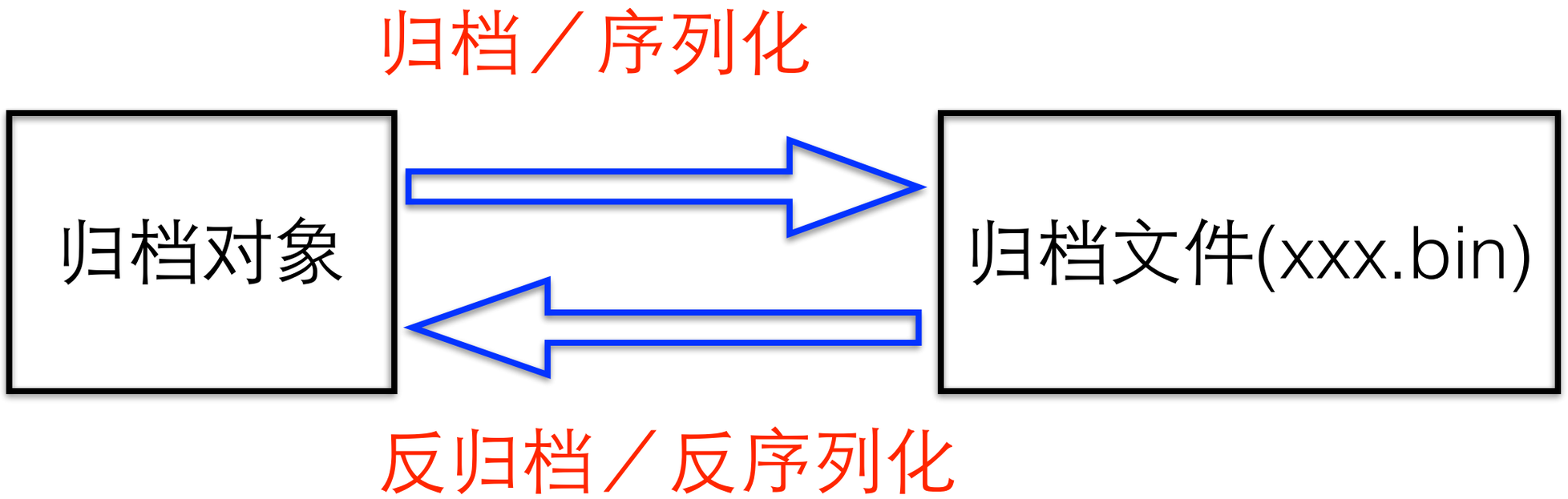

NSArray
NSDictionary 第一个元素
name = “Bob”
age = 19
NSDictionary 第二个元素
name = “Jonny”
age = 20


归档／序列化
归档对象
归档文件(xxx.bin)
反归档／反序列化

数据库：“按照数据结构来组织、存储和管理数据的仓库”

SQLite介绍：

- SQLite是一个开源的嵌入式关系数据库，它在2000年由D. Richard Hipp发布
- SQLite大量的被用于手机、MP3播放器以及机顶盒等设备

特性：

- 零配置：sqlite3不用安装、不用配置，不需要启动
- 可移植性：SQLite可移植性好，很容易使用，容量小(只需要几百K内存)，高效而且可靠

SQL介绍

- SQL是Structured Quevy Language(结构化查询语言)的缩写。
- SQL是专为数据库而建立的操作命令集，是一种功能齐全的数据库语言

SQLite支持的数据格式:

- NULL：值是NULL
- INTEGER：值是有符号整形，根据值的大小以1,2,3,4,6或8字节存放
- REAL：值是浮点型值，以8字节IEEE浮点数存放
- TEXT：值是文本字符串，使用数据库编码(UTF-8，UTF-16BE或者UTF-16LE)存储

术语:

- 表 (table): 是数据存储的最常见和最简单的形式，是构成关系型数据库的基本元素
- 字段/属性 (column): 表中的列名字
- 纪录 (record): 每行的数据

数据库 personDB.db

表一： person

id	name	age	class
1	Jonny	19	class1
2	Bob	17	class2
3	Maggie	18	class3

表二： details

id	name	father	country	birth place
1	Jonny	Tom	Amerian	New York
2	Bob	Jerry	Canada	Ottawa
3	Maggie	Jack	Australia	Sydney

表三

……

常用SQL语句：

创建表：

```
create table person (name text, age integer);
create table people (id integer primary key, name text, age integer);
```

增：插入一条数据

1. 指定id

```
insert into person (id, name, age, height) values(1,'shirley',18,1.66);
```

2. 不指定id

```
insert into people (name, age, height) values(‘shirley', 18, 1.66);
```

查：选择数据

1. 查询所有数据

```
select * from person;
select * from people;
```

2. 查询满足某个条件的数据

```
select *from people where name=‘shirley’;
```

3. 查询满足两个个条件的数据

```
select *from people where age>19 and height>1.65
```

改：更新几条数据

update people set name='jonny' where name='maggie';

```
update people set height=1.85 where id=5;
```

—> 针对主键那条纪录，更新一条数据

删：删除几条数据

1. 删除某条纪录

   delete from people where name='jonny';

2. 删除整个表

   drop table person;

iOS中libsqlite3.dylib框架方法解释

- sqlite3 *db: 数据库句柄
- sqlite3_stmt *stmt: 用于保存编译好的SQL语句
- sqlite3_open(): 打开数据库；数据库不存在时创建。
- sqlite3_exec(): 执行非查询的sql语句
- sqlite3_step(): 在调用sqlite3_prepare后，使用这个函数在记录集中移动。
- sqlite3_close(): 关闭数据库文件

- 下面的函数，用于从记录集字段中获取数据
  - sqlite3_column_text(), 取text类型的数据。
  - sqlite3_column_blob（），取blob类型的数据
  - sqlite3_column_int(), 取int类型的数据

FMDB

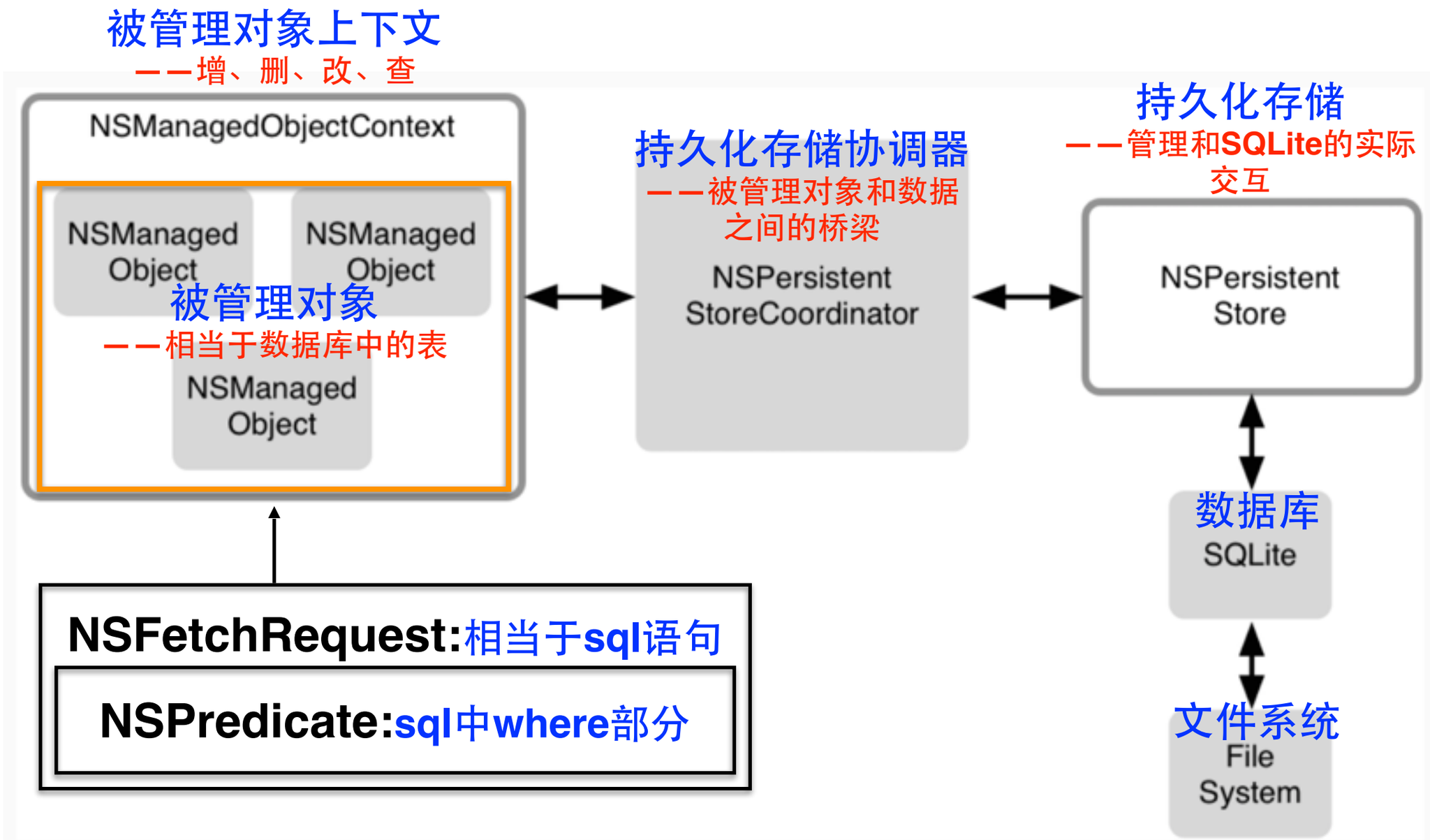
- FMDB是iOS平台的SQLite数据库第三方开源框架
- FMDB以面向对象的方式封装了SQLite的C语言API
- FMDB兼容ARC和非ARC工程
- 主页：https://github.com/ccgus/fmdb

FMDB常用类：

- FMDatabase： 单一的SQLite数据库类
- FMResultSet： 存储对FMDatabase对象的查询结果集
- FMDatabaseQueue：用于多个线程执行查询和更新的情况 (详情查看主页中的Thread Safety部分)

备注：

在FMDB中，除Query查询以外的所有操作 (create、drop、insert、update、delete等)，都称为“更新”



1. **data model(数据模型文件.xcdatamodel):相当于数据库文件;**
2. **数据模型对应OC中的NSManagedObjectModel(被管理的对象模型类):管理数据库中的所有“表”**

目标：使用Core data解决方案中的NSFetchedResultsController控制器来控制table view的数据显示

实现功能：好友的插入、查询、删除和更新

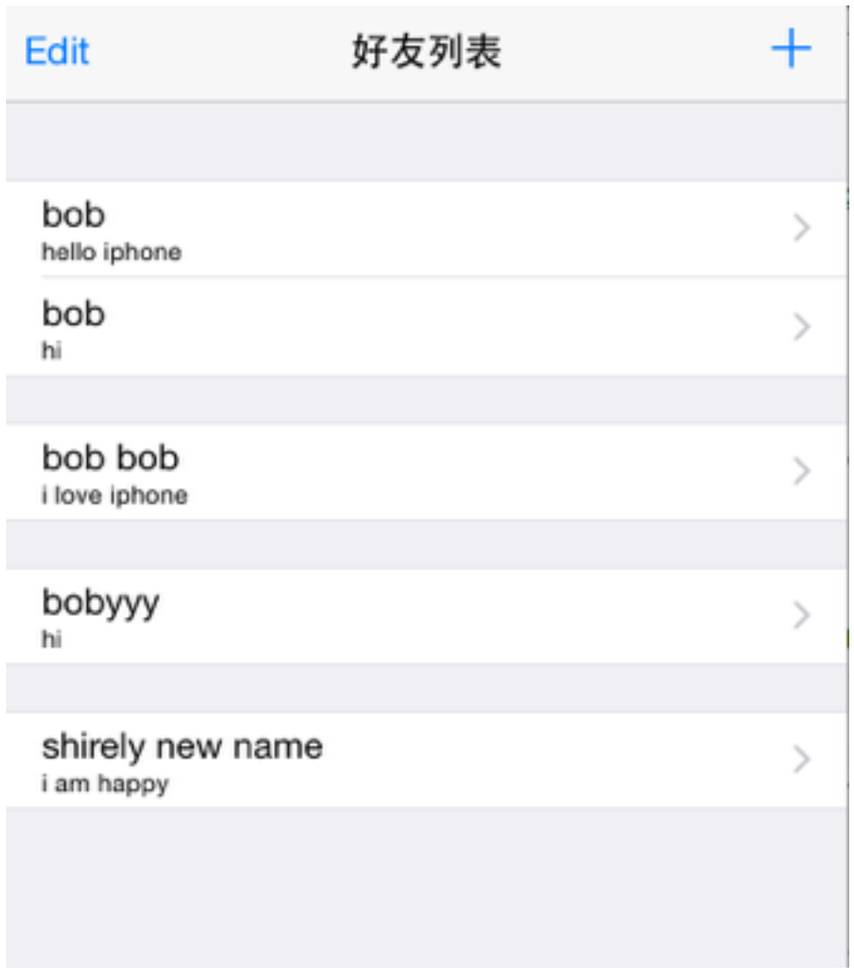
准备工作:

1. 创建工程，选择“User Core Data”选项
2. 导入<CoreData/CoreData.h>
3. 添加Entity(实体),并添加属性、设置属性的类型
4. 创建被管理对象类(NSManagedObject)

功能一：添加／插入好友记录

1. ContactTableViewController

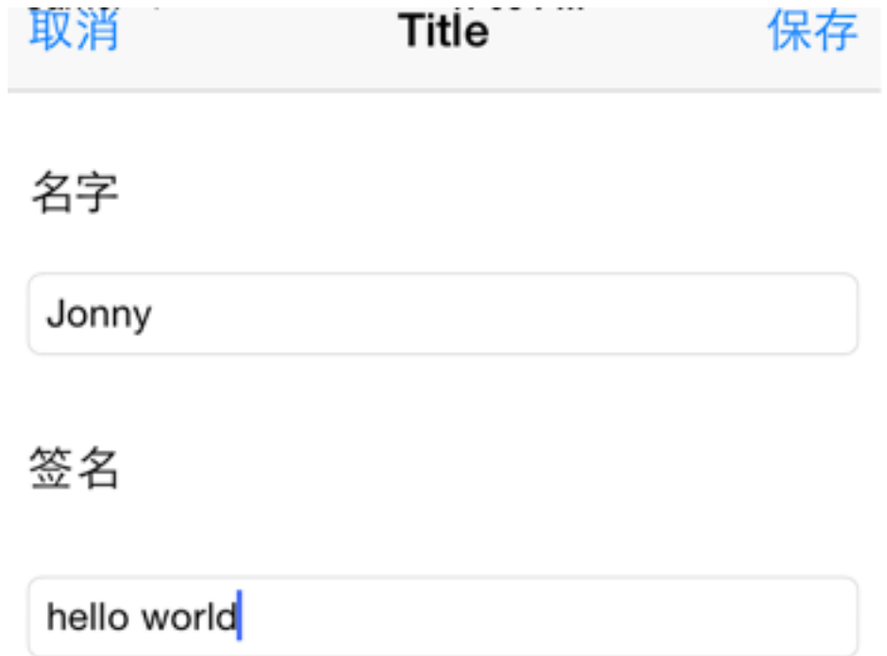
2. AddContactViewController



点击”+”按钮

功能一：添加／插入好友记录

2. AddContactViewController



点击”保存”按钮 →

1. 保存到sqlite

步骤1. 在类2中， 实现第2个类中的save点击函数

1.1 从结果控制器中获取一个空的模型对象

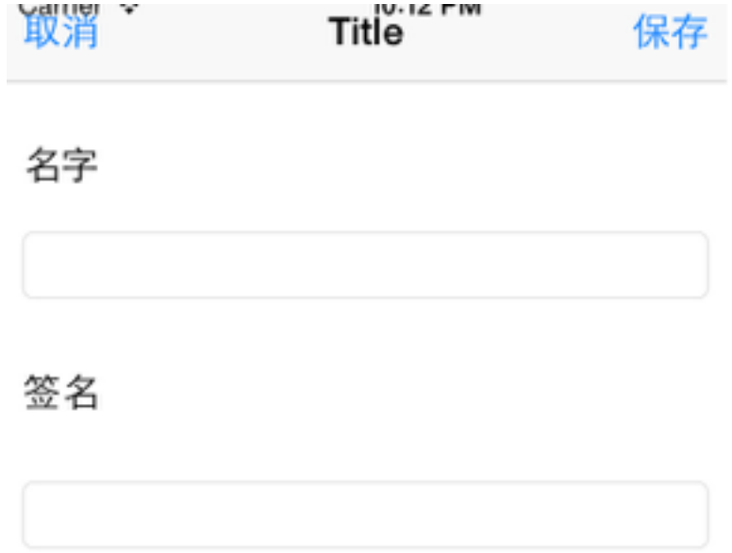
1.2 将用户输入的名字和签名赋值给该空对象

1.3 使用上下文将赋值完毕的模型对象保存到数据库文件中

1.4 退出当前控制器

功能一：添加／插入好友记录

2. AddContactViewController



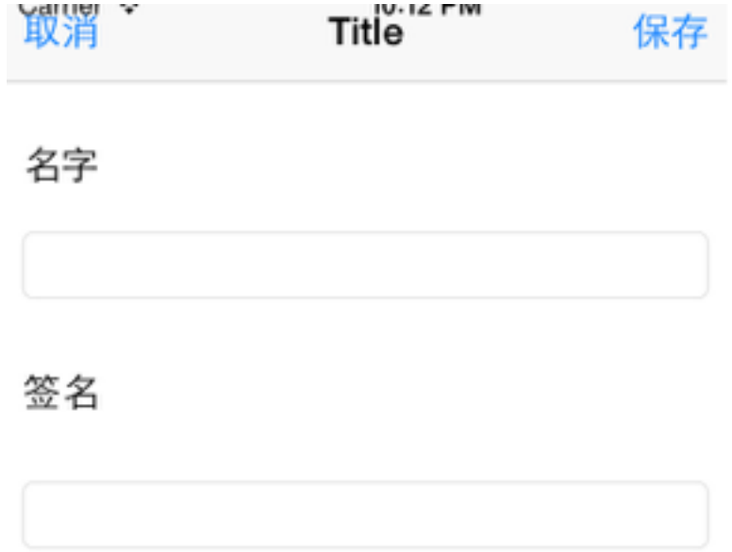
点击”保存”按钮

2. 结果控制器监听到sqlite的变化

步骤2. 在**类1**中，实现“结果控制器”中的四个协议方法
2.1 实现针对行的监听方法(此时type为insert)
2.2 实现针对section的监听方法(此时type为insert)
2.3 实现“结果控制器”将要发生变化的监听方法
2.4 实现“结果控制器”已经发生变化的监听方法

功能一：添加／插入好友记录

2. AddContactViewController



点击”保存”按钮

3. 更新table view界面

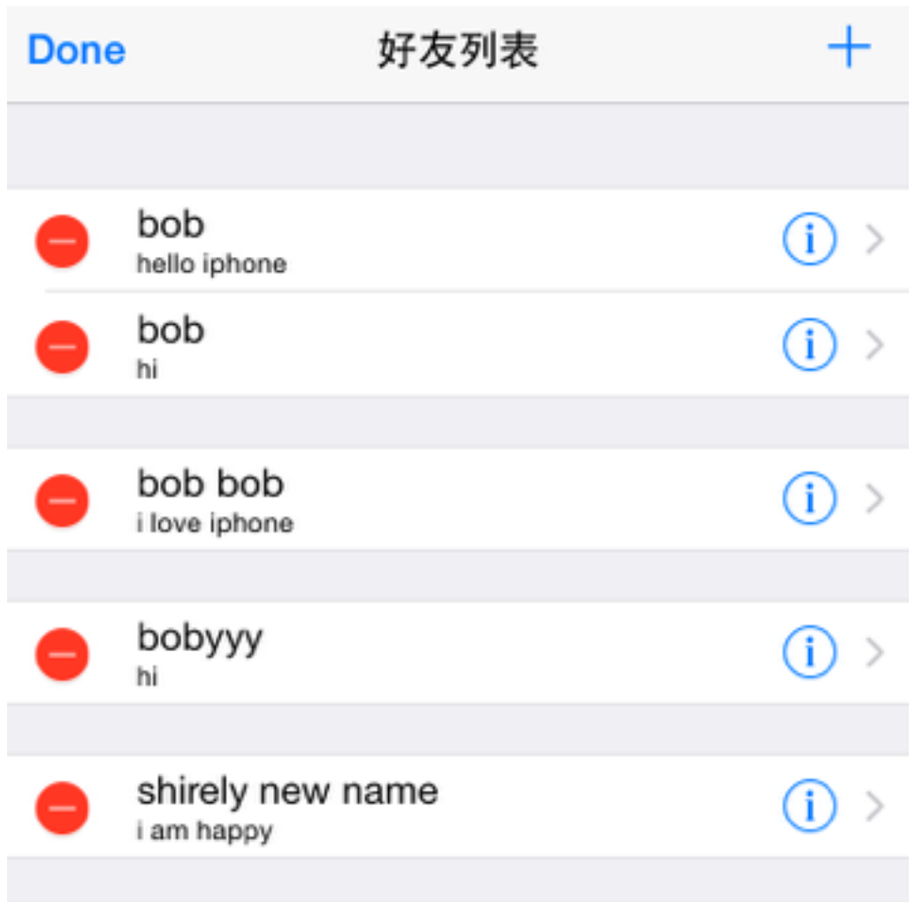
步骤3. 在**类1**中，实现table view中的三个协议方法
3.1 实现table view中返回行数的协议方法
3.2 实现table view中返回section的协议方法
3.3 实现table view中设置行的协议方法

功能二：删除好友记录

1. ContactTableViewController

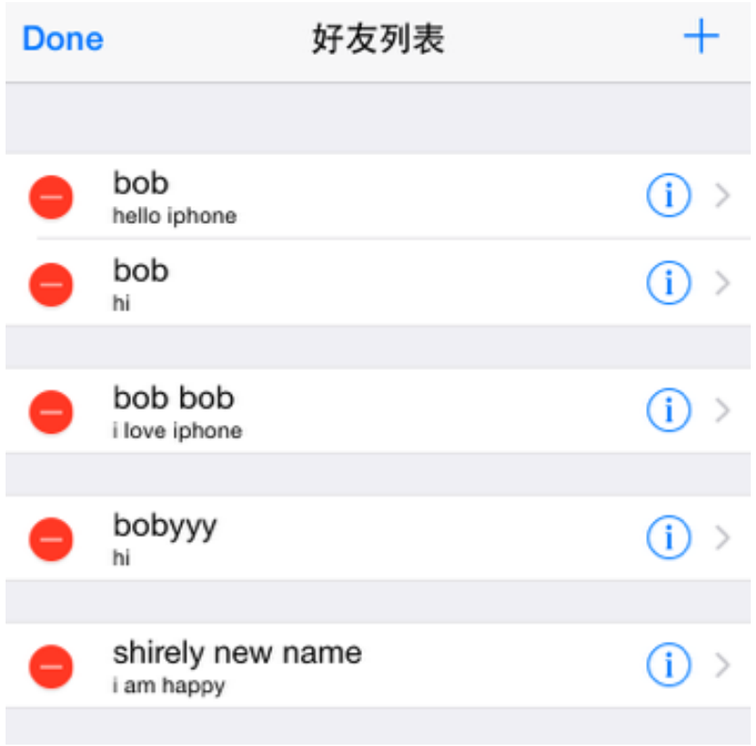
1. ContactTableViewController

点击”Edit”按钮



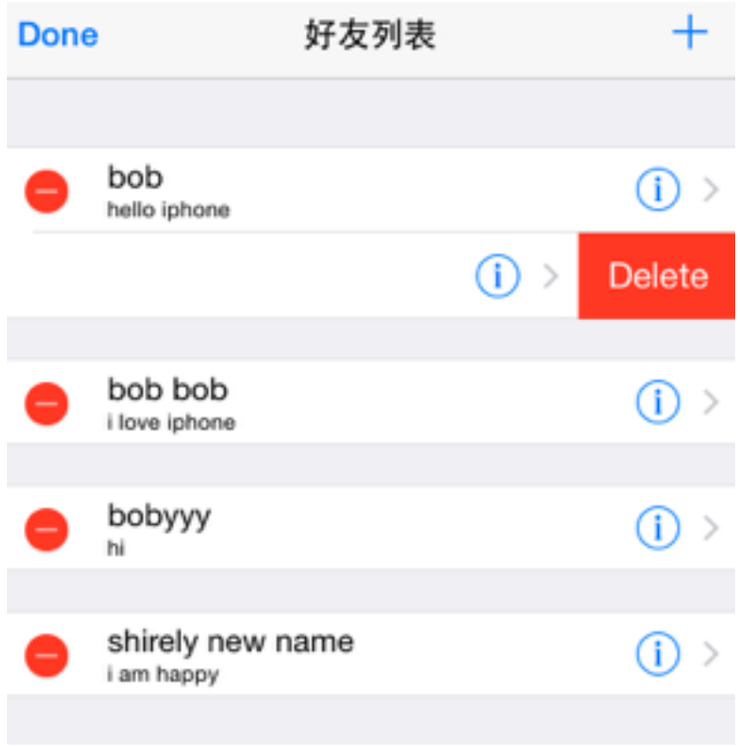
功能二：删除好友记录

1. ContactTableViewController



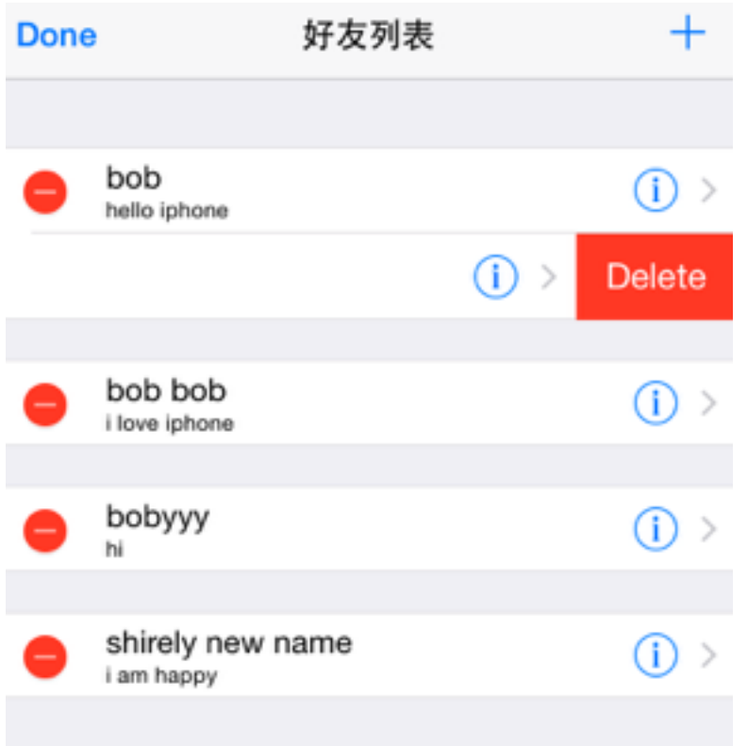
点击”－”按钮

1. ContactTableViewController



功能二：删除好友记录

1. ContactTableViewController



点击
”Delete”

1. ContactTableViewController

sqlite

步骤1: 当点击“Delete”时，在**类1**中，

1.1 实现table view的一个和编辑行有关的协议方法

1.1.1 通过”结果控制器“来获取被管理对象上下文

1.1.2 通过被管理对象上下文删除indexPath位置的对象

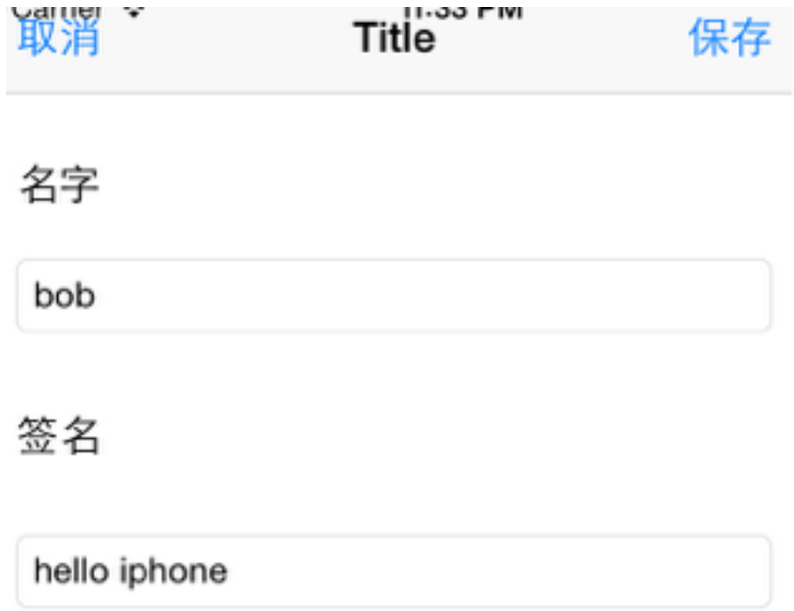
1.2.3 通过被管理对象上下文调用save函数，将数据从数据库中删除

功能三：更新好友记录

1. ContactTableViewController

2. AddContactViewController

点击”i”按钮



步骤1: 在table view编辑状态下，当点击“i”按钮时，在**类1**中，

1.1 实现table view的一个和点击accessory按钮有关的协议方法

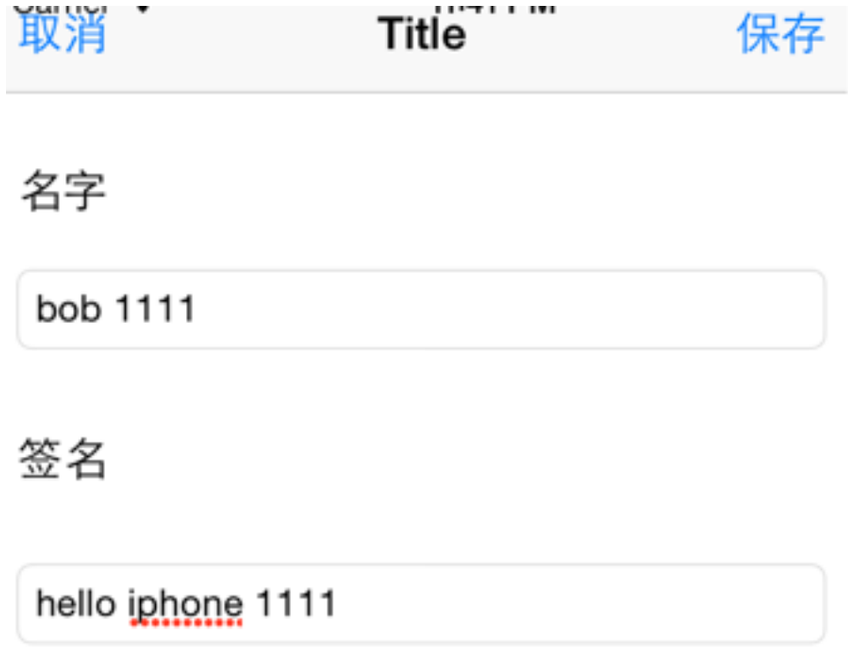
1.1.1 通过”结果控制器“来获取indexPath位置的Contact对象

1.1.2 创建类2的对象，并使用自定义的初始化方法，将上面获取的对象传给类2

1.2.3 从类1跳转到类2

功能三：更新好友记录

2. AddContactViewController



点击
”保存”

1. ContactTableViewController

sqlite

步骤2: 当点击”保存”时，在**类2**中，

2.1 添加save触发方法的逻辑

2.1.1 通过传入的contact对象是否为空，来判定是否是从编辑状态过来的； 如果contact对象为空，做插入操作；否则只做更新操作