* Core data programming guide (core data)
* **Introduction** Core Data is a schema-driven object graph management and persistence framework. Fundamentally, Core Data helps you to save model objects (in the sense of the model-view-controller design pattern) to a file and get them back again.
* **Programming Steps:**
* ***>>1、Create the Project***

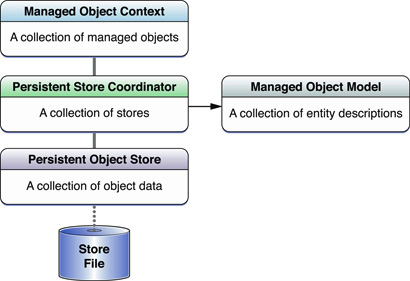
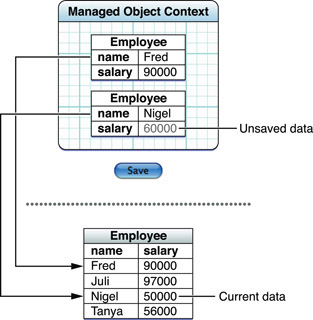
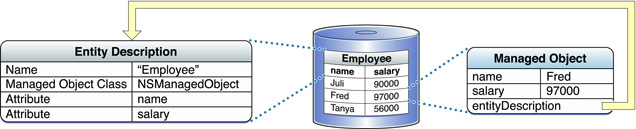
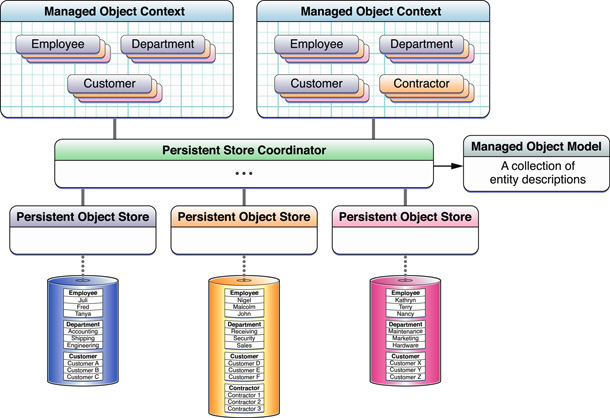
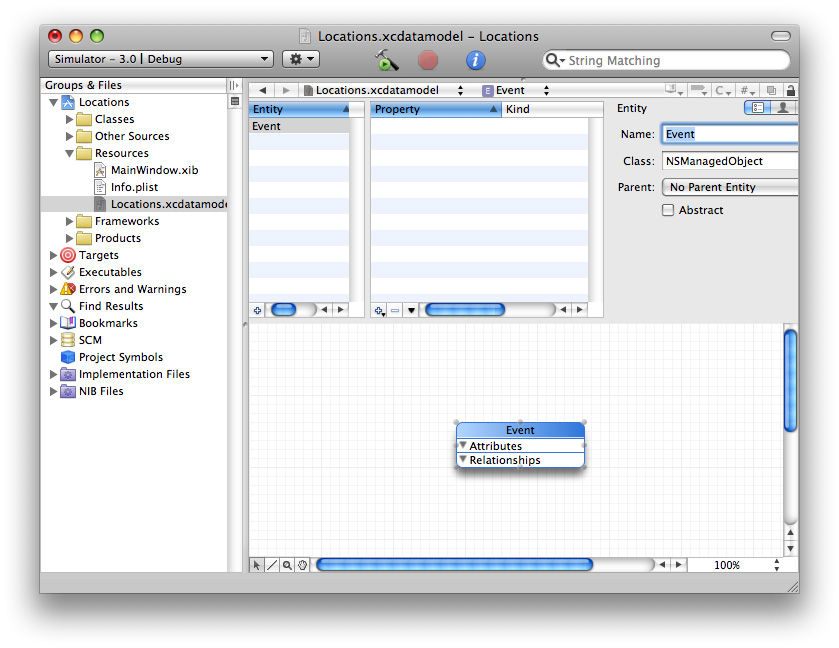
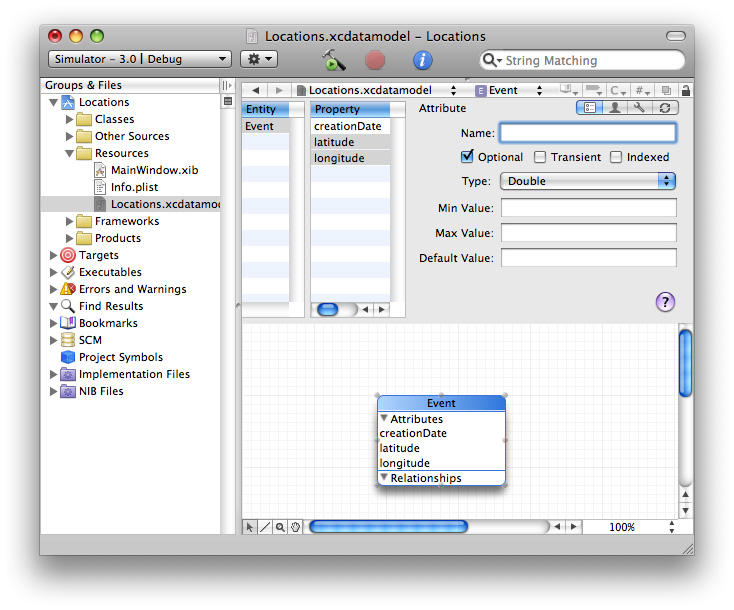
**In Xcode, create a new project using the Window-Based Application template in the iOS section. In the Options section, select the switch to use Core Data for storage.**Together with various other supporting files, the template provides you with:

* *An application delegate class*
* *A MainWindow interface (.xib) file*

*A Core Data model (.xcdatamodeld) file—typically referred to* as the managed object model

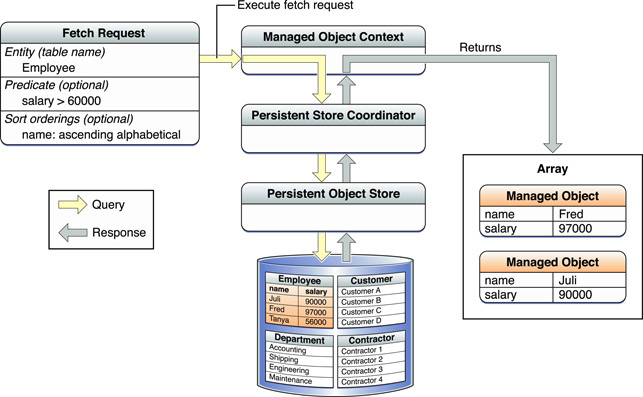
*>>2、Add properties for core data framework*

|  |
| --- |
| * *@property (nonatomic, retain, readonly) NSManagedObjectContext \*managedObjectContext;* |
| * *@property (nonatomic, retain, readonly) NSManagedObjectModel \*managedObjectModel;* |
| * *@property (nonatomic, retain, readonly) NSPersistentStoreCoordinator \*persistentStoreCoordinator;* |
|  |
| * *- (NSURL \*)applicationDocumentsDirectory;* |
| * *- (void)saveContext;* |

* *the relationship in core data framework*
* **
* **Managed objects in a context, and a table in the persistent store**
* 
* An entity description, a table in the database, and a managed object.
* 
* illustrates the role the coordinator plays. Stacks aren’t usually this complicated.
* 
* *>>3、Manage object and context programming*
* 3.1)In Xcode, in the Resources group select the model file (Locations.xcdatamodel) to display the model editor.
* 3.2)Choose Design > Data Model > Add Entity to add a new entity to the model.
* You can also use the Add button (+) at the lower left of the entity pane, or use the shortcut menu within the diagram view in the model editor.
* You should see a new entry for the entity (called “Entity”) appear in the entity pane at the top left of the document editor, and a graphical representation of the entity (a rounded rectangle) appear in the diagram view. Now you can set the name for the new entity.
* 3.3) Make sure you have the new entity selected in the entity pane so that you see information about the entity in the detail pane at the right. Change the name of the entity to Event. (Don’t change the class name.)
* Your model should look similar to this:
* 
* There’s an important difference between the name of the entity and the name of the Objective-C class used to represent instances of the entity. Core Data uses the entity description to find out about the data objects it manages, so the class name doesn’t have to be the same as the entity name. Indeed, in some cases several entities may be represented by the same class—NSManagedObject. Core Data is able to differentiate the instances on the basis of their associated entity description.
* 3.4)add attributes
* 3.4.1 Make sure you have selected Event in the entity pane, then choose Design > Data Model > Add Attribute.
* You should see a new attribute (called newAttribute) appear in the property pane. You need to set its name and type.
* 3.4.2 Make sure you have selected the new attribute in the property pane, then in the detail pane change the name of the attribute to creationDate, and select Date from the Type pop-up menu.
* You don’t need to set any of the other values.
* Now add attributes for latitude and longitude.
* 3.4.3 Make sure you have selected Event in the entity browser, then choose Design > Data Model > Add Attribute twice (to add two attributes).
* 3.4.4 Select both the new attributes in the property pane, then in the detail pane select Double from the Type pop-up menu.
* 3.4.5 Select just the first new attribute in the property pane, and in the detail pane change the Name of the attribute to latitude.
* 3.4.6 Select just the second new attribute in the property pane, and in the detail pane change the Name of the attribute to longitude.
* Your model should look similar to this:
* 
* *>>4 Custom Managed Object Class*
* use Xcode to generate the files for a custom class to represent the Event entity.
* 4.1 In Xcode, in the model, select the Event entity. (You must select the entity; the selection is used to indicate which items to create subclasses for.)
* 4.2 Choose File > New File. In the New File dialog, select Managed Object Class.
* Depending on the version of Xcode you’re using, the Managed Object Class may be available in the iOS section under Cocoa Touch Classes, or you may need to choose the template in the Mac OS X section, under Cocoa—either will work correctly.
* 4.3 Click Next. The correct location and targets should have been selected for you. Click Next to accept them.
* You should see the Entity selection pane, with the Event entity selected. The “Generate accessors” and “Generate Objective-C 2.0 properties” options should also be selected.
* 4.4 Click Finish to generate the files.
* **Notice: the core data framework handle object(use nsnumber to encapsulate double,integer,float et al.)**
* ***>>5 Add, Fetch, Delete entity***

**5.1 Add Entity :you typically create a managed object using a convenience method of NSEntityDescription** [**insertNewObjectForEntityForName:inManagedObjectContext:**](file:///Users/suyuancheng/Library/Developer/Shared/Documentation/DocSets/com.apple.adc.documentation.AppleiOS5_1.iOSLibrary.docset/Contents/Resources/Documents/documentation/Cocoa/Reference/CoreDataFramework/Classes/NSEntityDescription_Class/NSEntityDescription.html#//apple_ref/occ/clm/NSEntityDescription/insertNewObjectForEntityForName:inManagedObjectContext:) **which returns a properly initialized instance of the correct class for the entity you specify, inserted into the managed object context.**

**5.2 Fetch entity:**

****

**5.2.1 create a NSFetchRequst and set entity of it use the method : entityForName: inManagedObjectContext:to get the entity**

**5.2.2 execute the request: use method:** **executeFetchRequest: error: of manage object context. By the way it returns a array which fetched from the entity.**

**5.3 Delete entity:use the method:deleteobject of manage object context to delete a manage object**

* **Use the save method of manage object context to save the changes.**