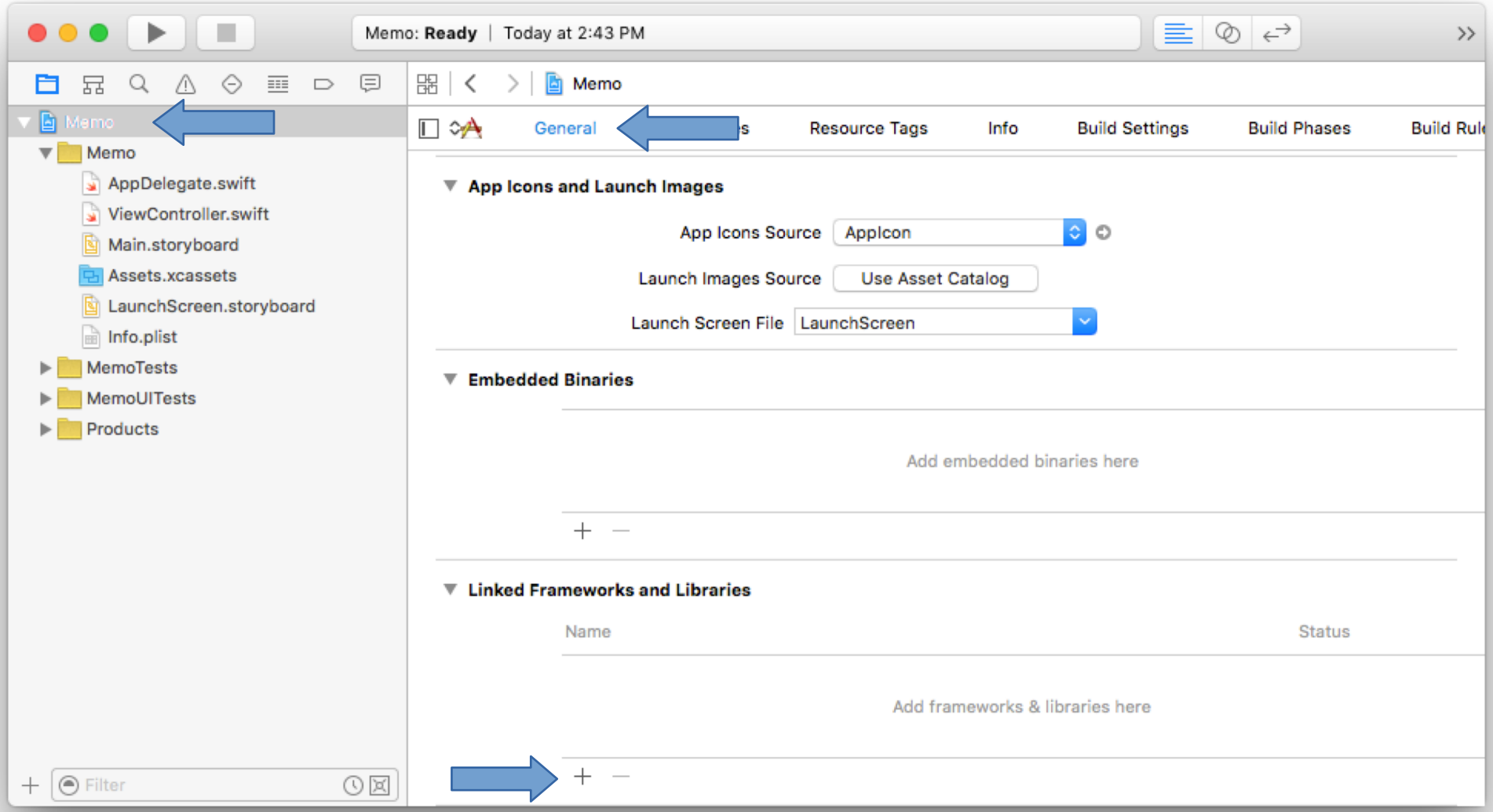
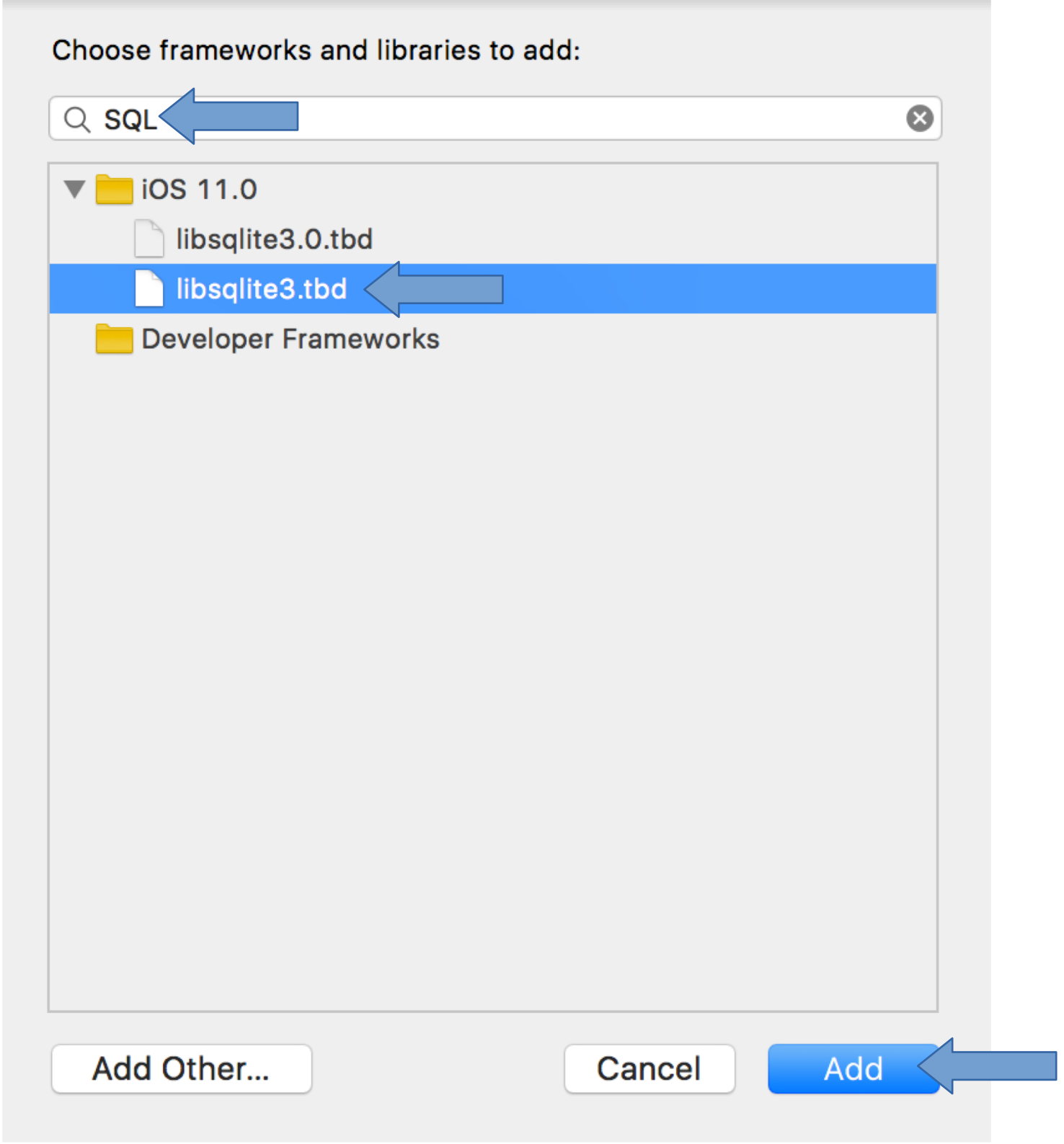
SQLite

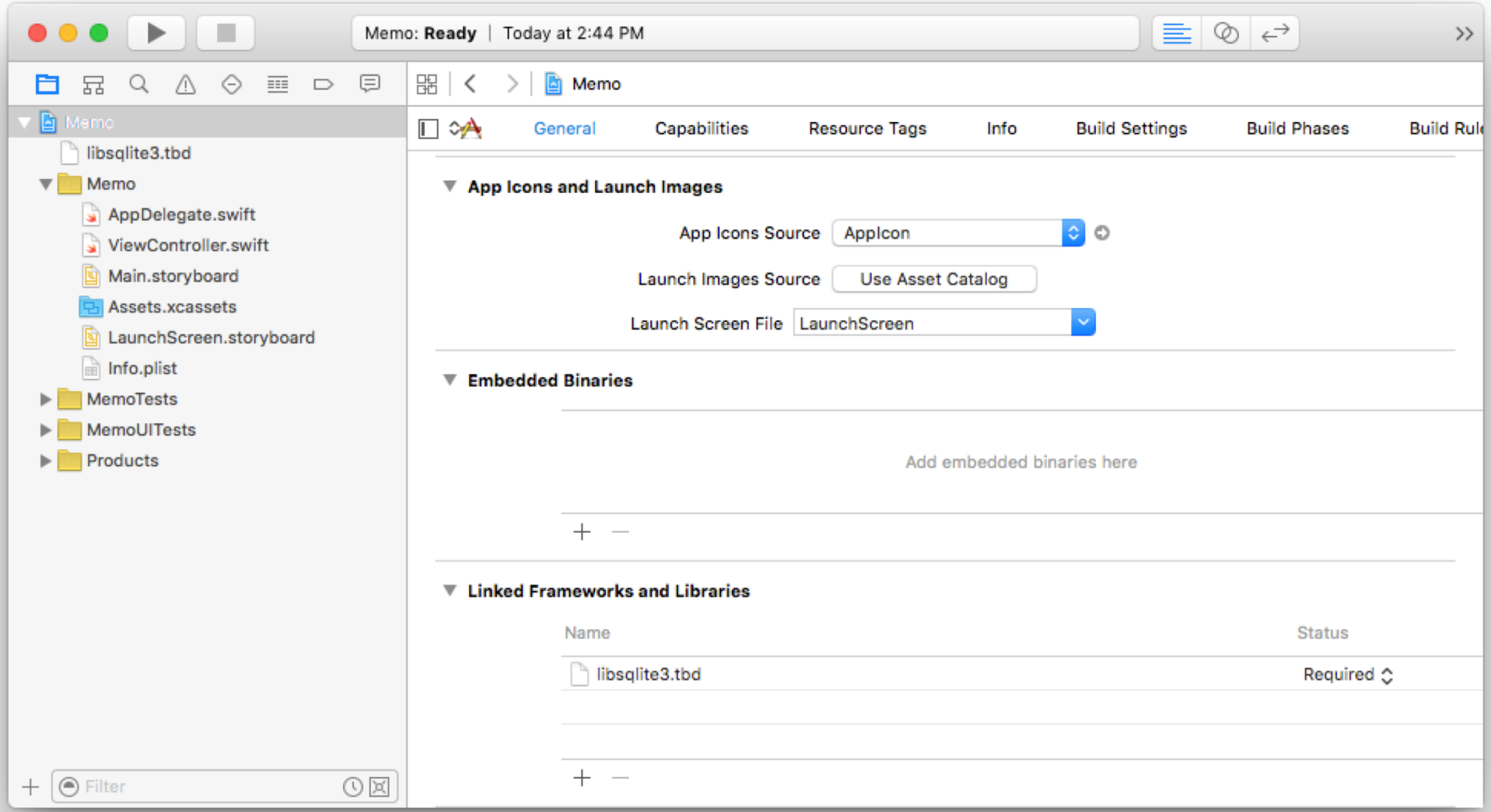
Create Memo App

- Create a “Memo” project

Add SQL library to the project.



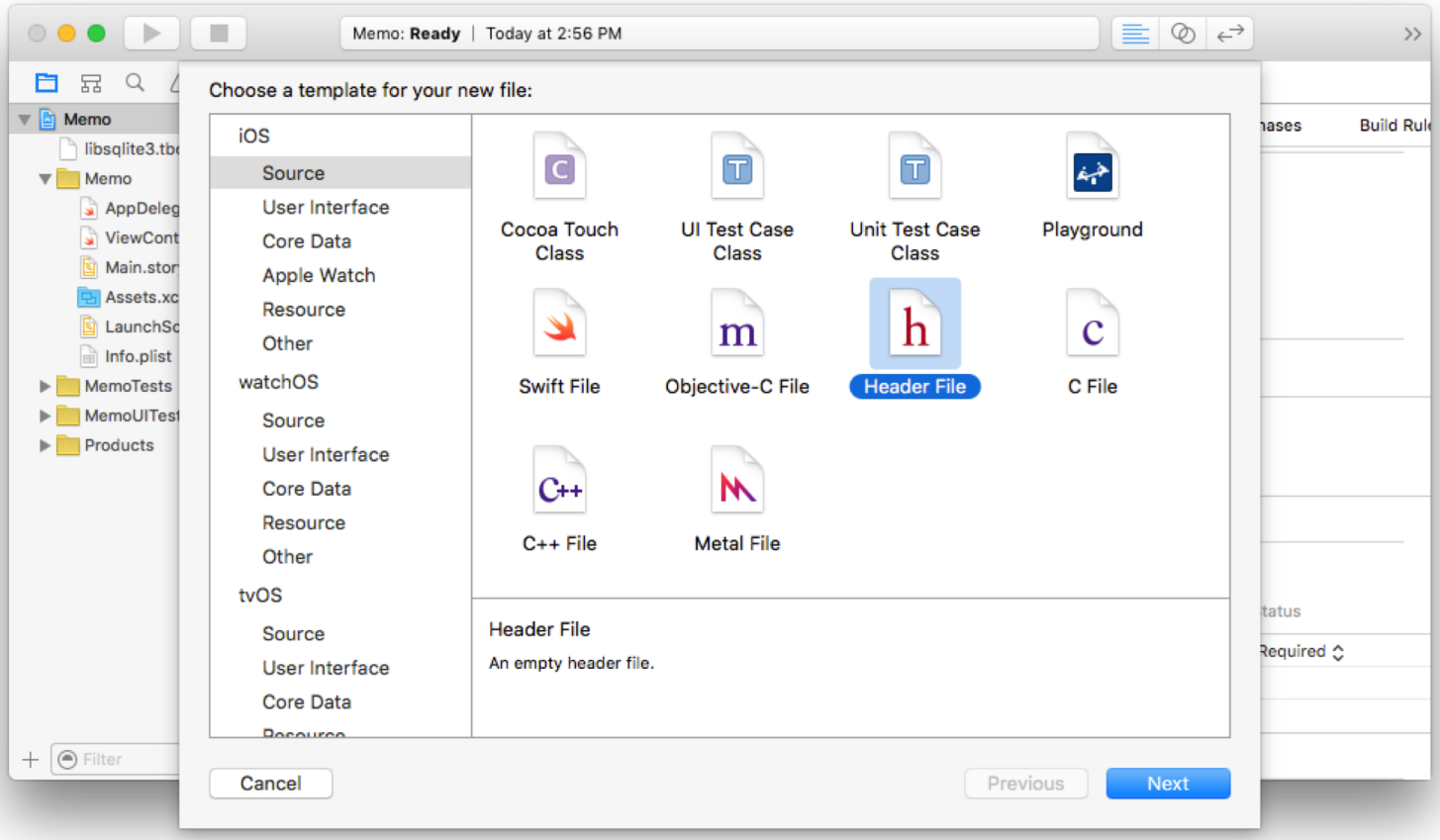

Choose frameworks and libraries to add:
SQL
iOS 11.0
libsqlite3.0.tbd
libsqlite3.tbd
Developer Frameworks
Add Other...
Cancel
Add


Memo: Ready | Today at 2:44 PM
Memo
Memo
libsqlite3.tbd
Memo
AppDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
MemoTests
MemoUITests
Products
Filter
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rul
App Icons and Launch Images
App Icons Source
AppIcon
Launch Images Source
Use Asset Catalog
Launch Screen File
LaunchScreen
Embedded Binaries
Add embedded binaries here
Linked Frameworks and Libraries
Name
Status
libsqlite3.tbd
Required

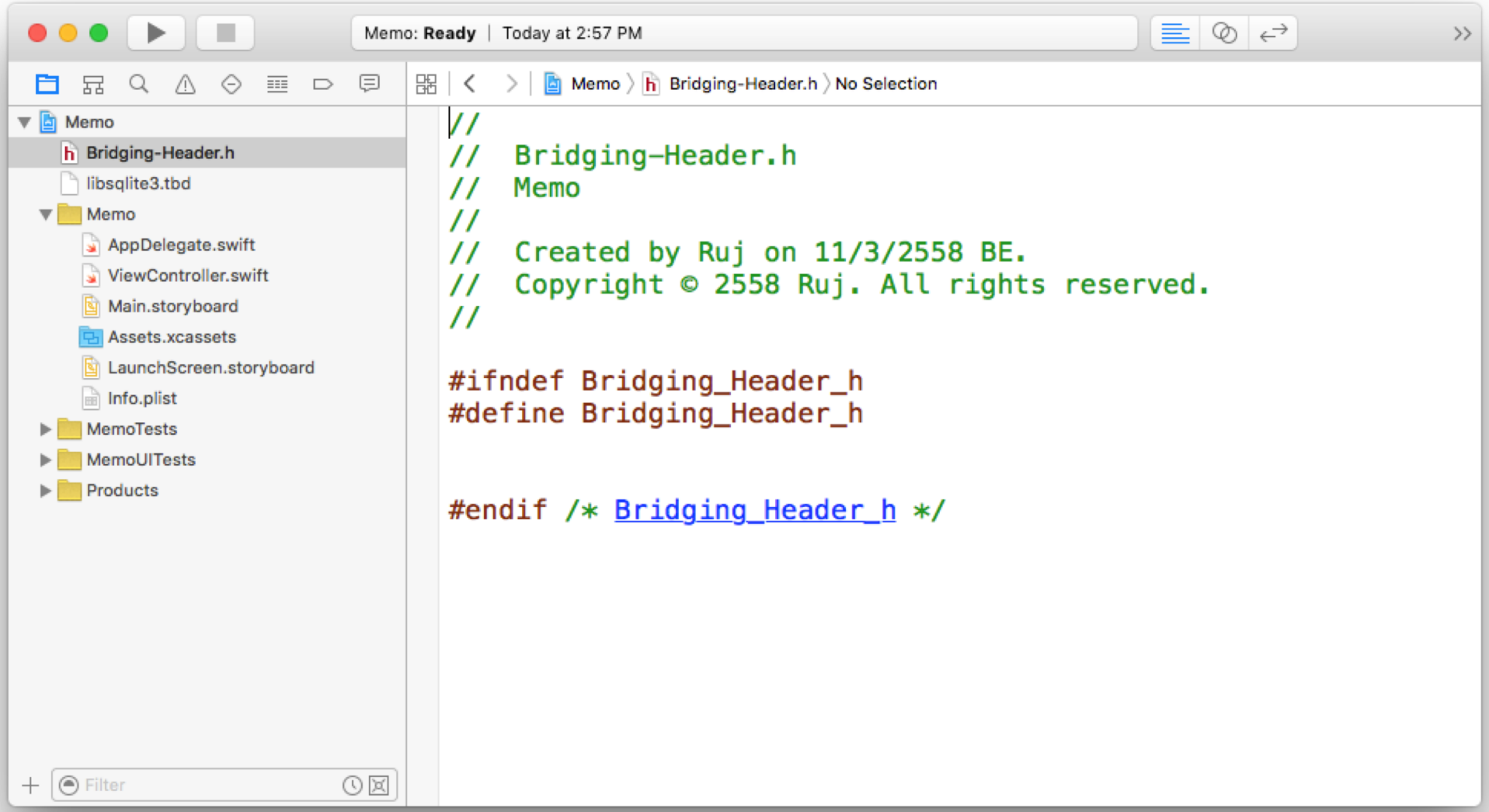
Linking SQLite

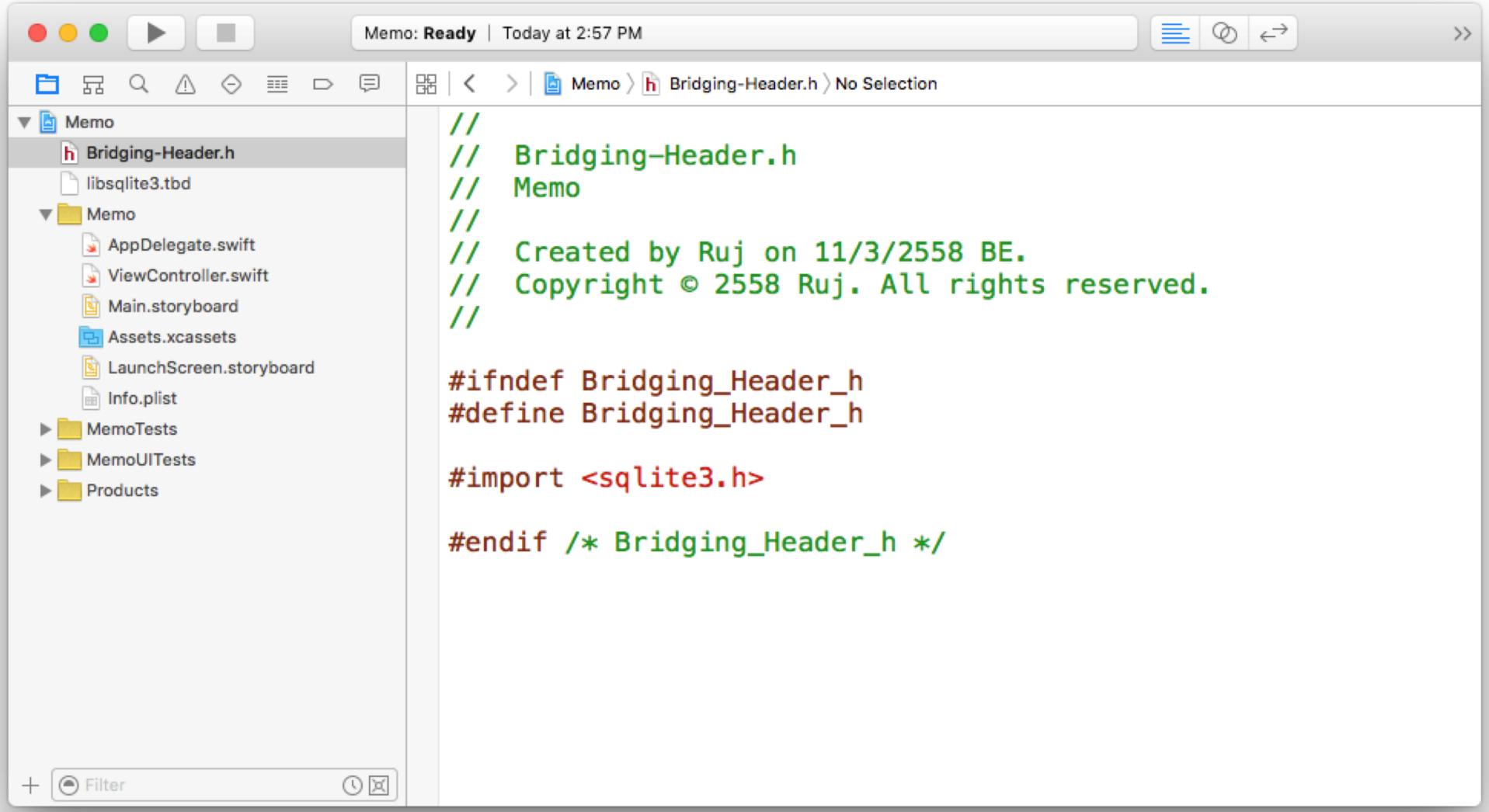
- Since SQLite is C library. You need to create a bridge file to connect Swift with SQLite.

Crate Bridge The Hard Way



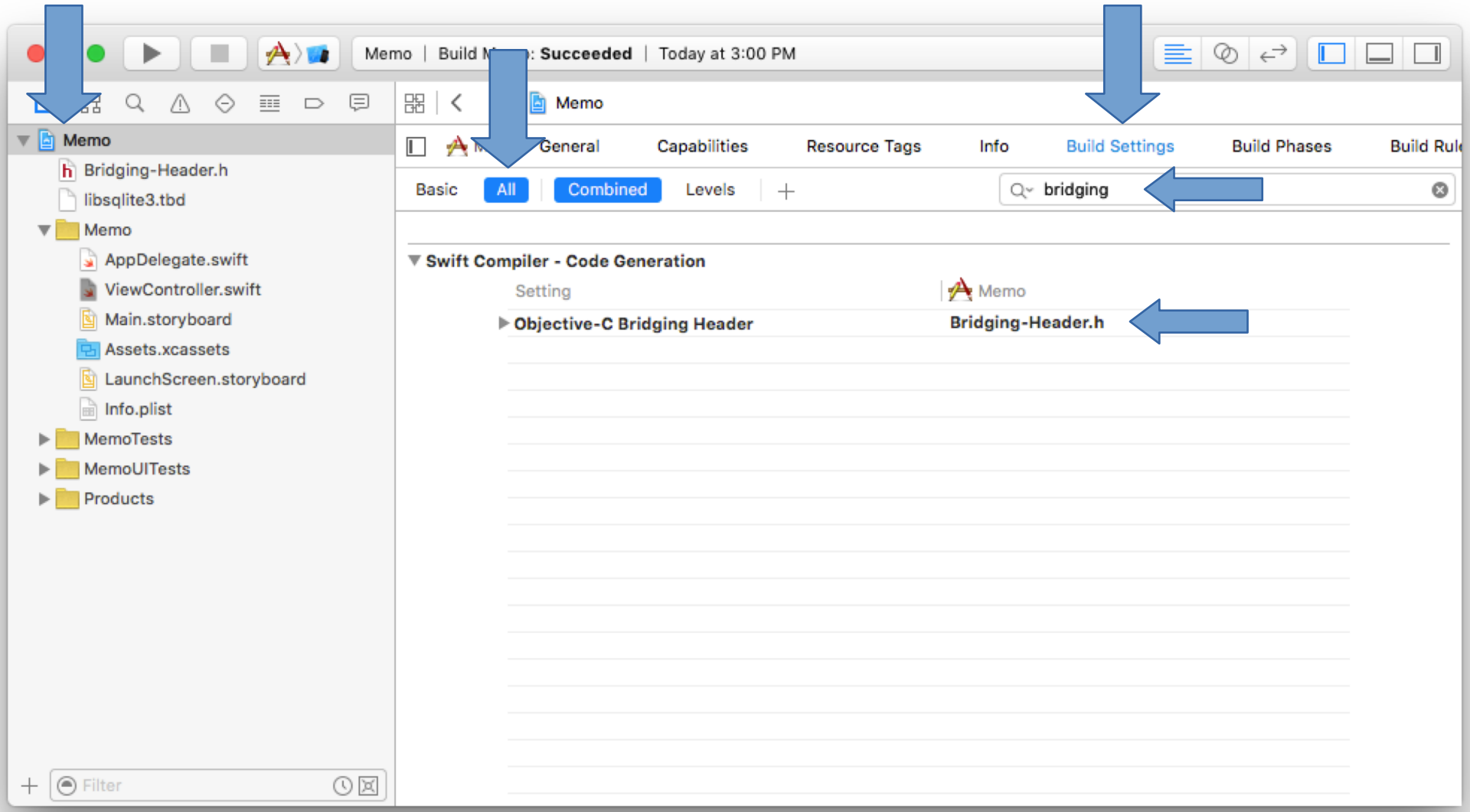
- Create file name “Bridging-Header.h” (This file is placed in project folder not source folder)


Memo: Ready | Today at 2:57 PM
Memo
Bridging-Header.h
No Selection
Memo
Bridging-Header.h
libsqlite3.tbd
Memo
AppDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
MemoTests
MemoUITests
Products
Filter
//
// Bridging-Header.h
// Memo
//
// Created by Ruj on 11/3/2558 BE.
// Copyright © 2558 Ruj. All rights reserved.
//
#ifndef Bridging_Header_h
#define Bridging_Header_h
#endif /* Bridging_Header_h */


Memo: Ready | Today at 2:57 PM
Memo
Bridging-Header.h
No Selection
Memo
Bridging-Header.h
libsqlite3.tbd
Memo
AppDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
MemoTests
MemoUITests
Products
Filter
//
// Bridging-Header.h
// Memo
//
// Created by Ruj on 11/3/2558 BE.
// Copyright © 2558 Ruj. All rights reserved.
//
#ifndef Bridging_Header_h
#define Bridging_Header_h
#import <sqlite3.h>
#endif /* Bridging_Header_h */

Linking SQLite

- In project setting add the bridging header to the build setting (See the following slide)


Memo | Build M : Succeeded | Today at 3:00 PM
Memo
Bridging-Header.h
libsqlite3.tbd
Memo
AppDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
MemoTests
MemoUITests
Products
Filter
Memo
General
Capabilities
Resource Tags
Info
Build Settings
Build Phases
Build Rul
Basic
All
Combined
Levels
bridging
Swift Compiler - Code Generation
Setting
Memo
Objective-C Bridging Header
Bridging-Header.h

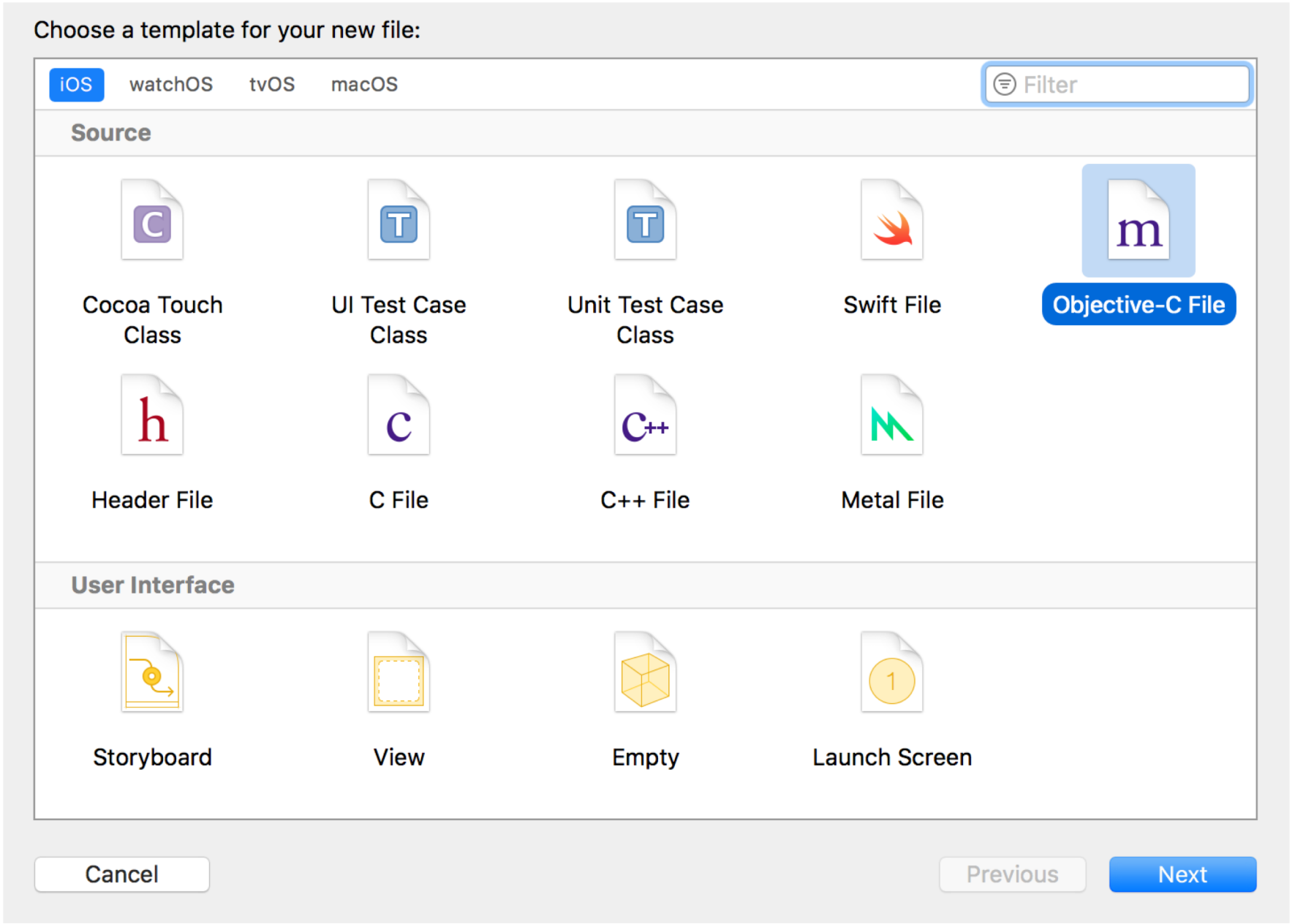
Linking SQLite

- Try building your project. If there is no error, you are doing it right. Remember, the location of the header file is important.
- Once built, you can auto-complete sqlite functions.

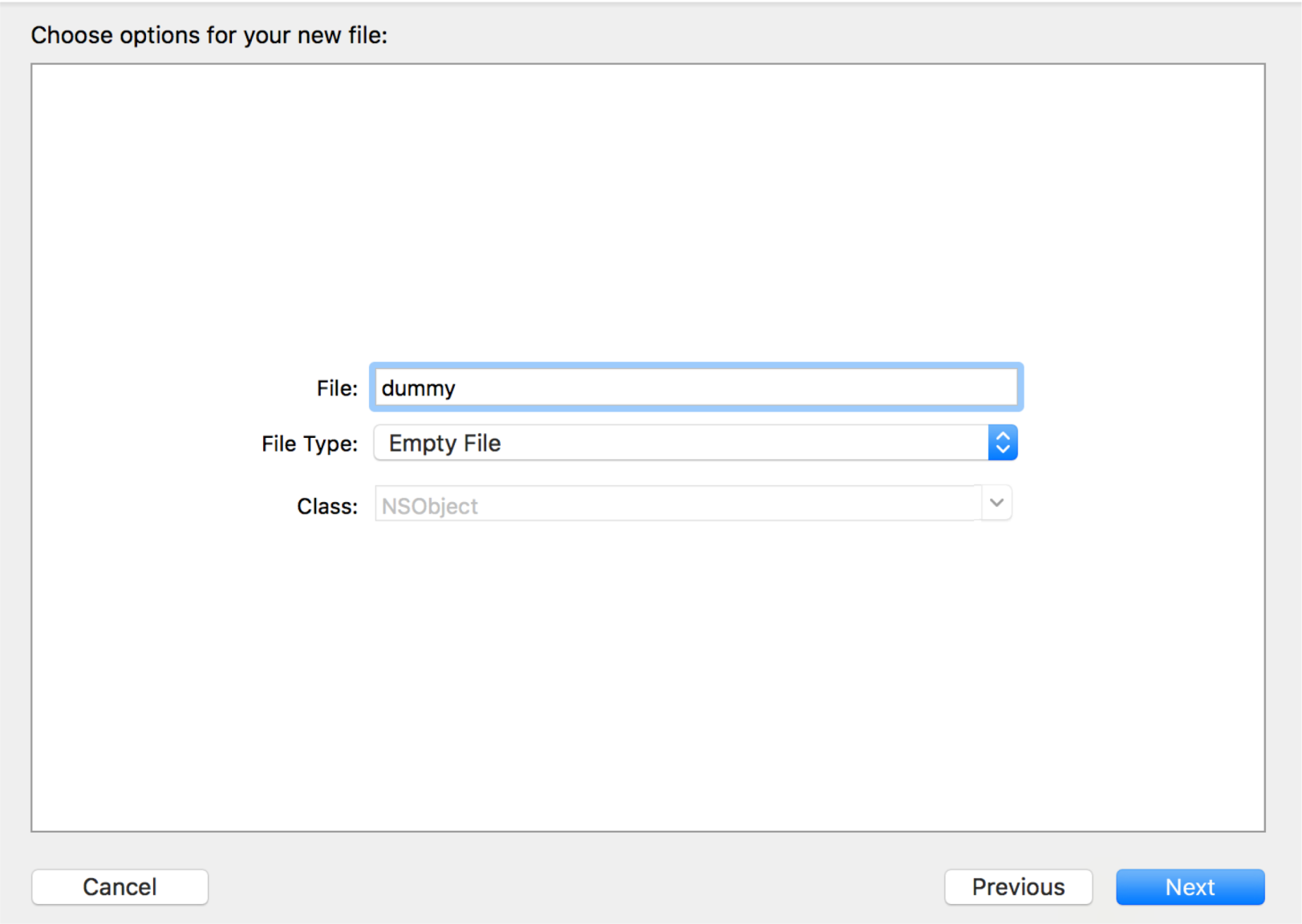
Finish The Hard Way

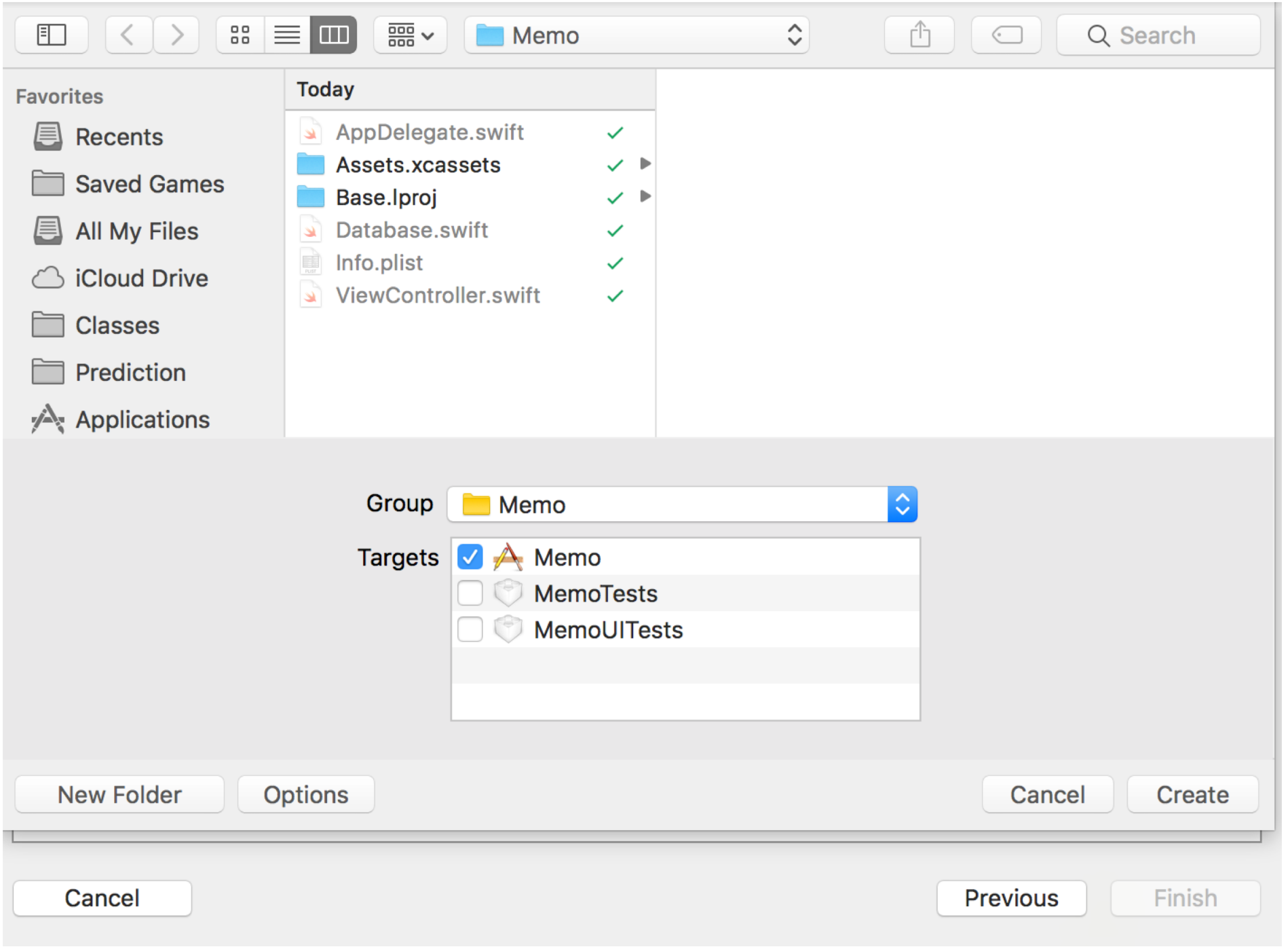
Crate Bridge The Easy Way

Create dummy Objective-C file

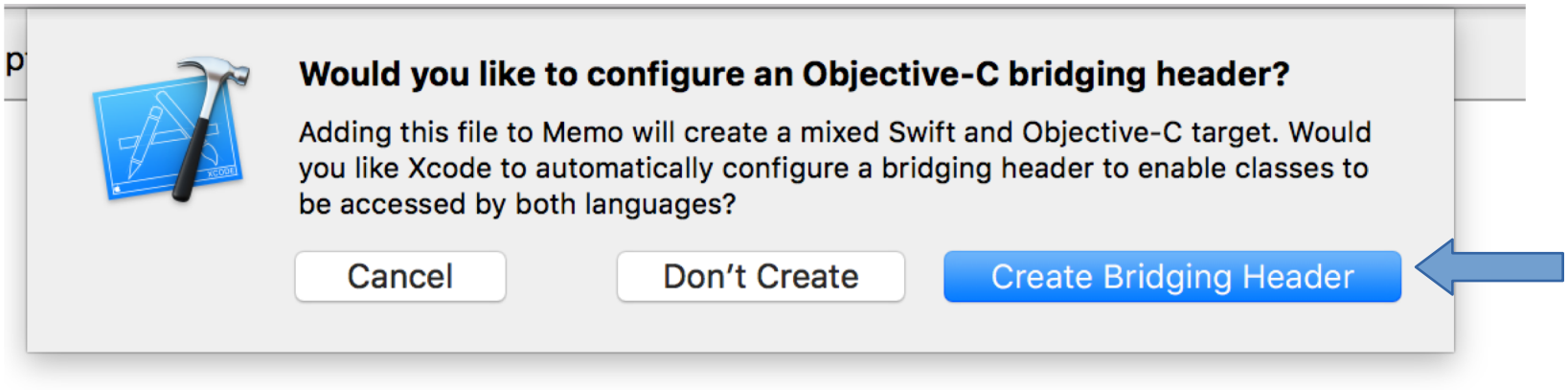


Give it a dummy name. We will delete this file later anyway.

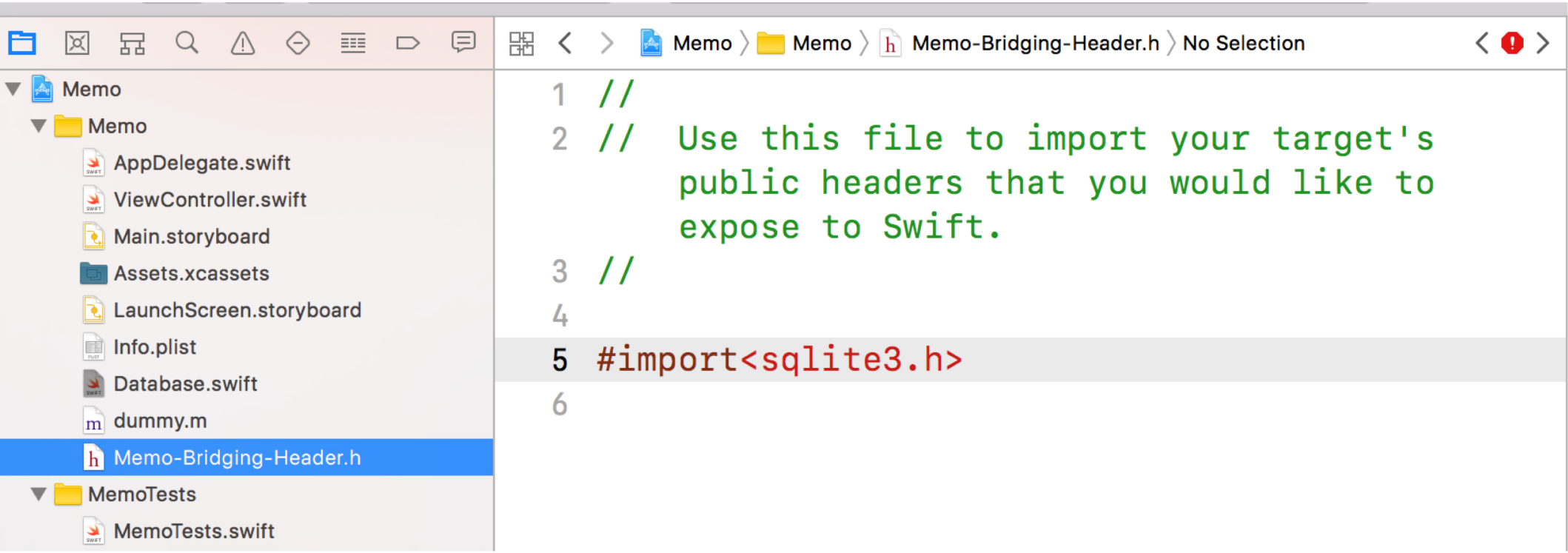



Memo
Search
Favorites
Recents
Saved Games
All My Files
iCloud Drive
Classes
Prediction
Applications
Today
AppDelegate.swift
Assets.xcassets
Base.lproj
Database.swift
Info.plist
ViewController.swift
Group
Memo
Targets
Memo
MemoTests
MemoUITests
New Folder
Options
Cancel
Create
Cancel
Previous
Finish

Xcode will offer to create header file for us and do all the hard work.



Delete dummy.m to remove clutter.



Finish The Easy Way

Database.swift

- It's better to wrap everything inside a database class.
- Create a “Database.swift” file to do just that.
- See following link for more detail of the functions.
- SQL lite http://www.sqlite.org/cintro.html

SQLite Operations

It needs to follow the sequences below:

1. sqlite3_open()
2. sqlite3_prepare_v2()
3. sqlite3_step()
4. sqlite3_column_XXX()
5. sqlite3_finalize()
6. sqlite3_close()

SQLite Operations

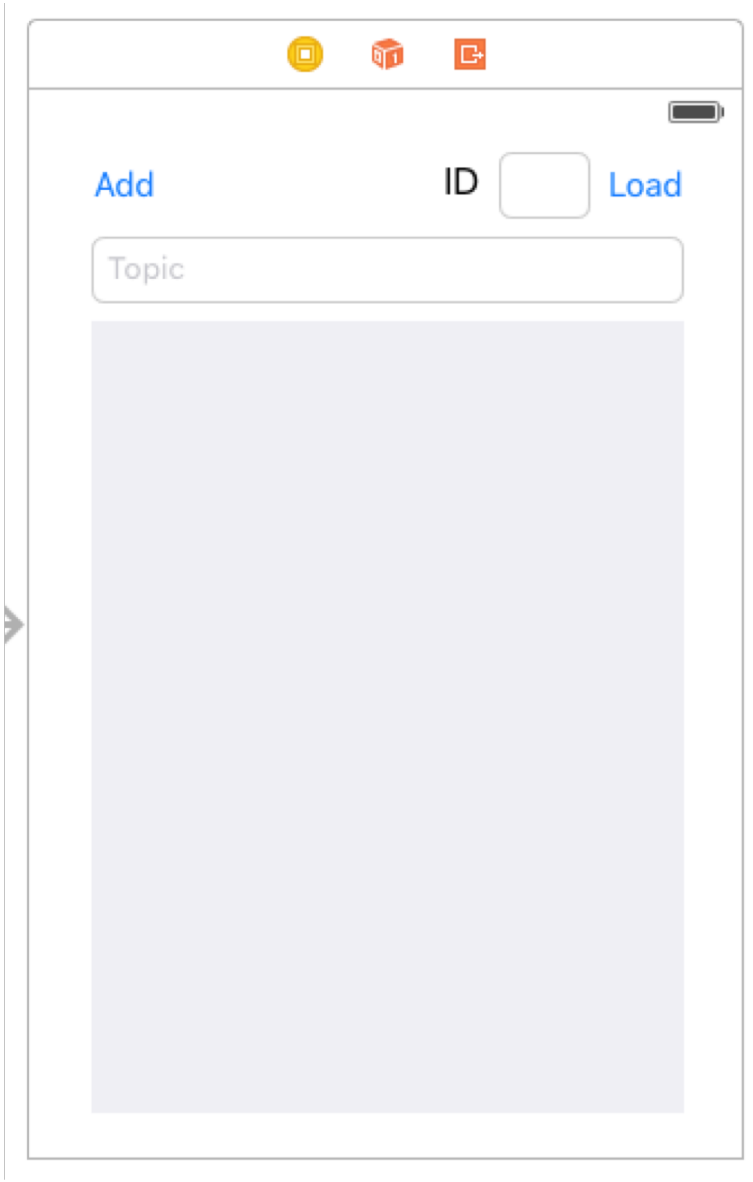
- You can take a shortcut by using sqlite3_exec() to replace 2-5 if you do not want any output.

- DONOT forget to use sqlite3_close() after sqlite3_finalize() or sqlite3_exec().

Database.swift

- init() is private to keep people from creating more than one database instance.
- Static function getSharedInstance() is used to retrieve Database instance.

Create the following page with appropriate outlets and event listeners



ViewController.swift

```
let db = Database.getSharedInstance()

 @IBOutlet weak var idTextField: UITextField!

 @IBOutlet weak var topicTextField: UITextField!

 @IBOutlet weak var contentTextView: UITextView!

 @IBAction func loadButtonTouch() {
     let result = db.getNote(recordID: Int(self.idTextField.text!)!)

     topicTextField.text = result.topic
     contentTextView.text = result.content
 }

 @IBAction func addButtonTouch() {
     let id = db.addNote(topic: topicTextField.text!,
                         content: contentTextView.text)

     self.idTextField.text = "\(id)"
 }
```

Testing

You can use Firefox to open sqlite file that you created in the program to verify the outcome.

Assignment

- Integrate SQLite to table view.