

第1回

「iOSでの画像/アニメーション/
TableViewとIBのイロハ」

toru.inoue@kissaki.tv

タグは#TechHUBJP

Agenda | /4

- ・下準備と前提の用意

この授業の前提スキル

Objective-Cの記法とかC言語のルール(メモリマネージメント)は
疑問があっても心の中にしまっておけたり後で自力で調べるなどができるレベル

iOSでの開発がすでにある程度できるが、
サクッと作りたいとか、
後で手直しが出ないように作りたいとか、
今の作り方に疑問がある、不満がある、ってレベルの人たち

→そんな皆さんを対象に、伝えて、悩みを聞くフリをするよ！

- ・コードは、gitoriousっていうところでバージョン管理とコードレビューをします。
<https://gitorious.org/gitorious>

バージョン管理の話は、今回の講義後にメールで行うよ！

(間に合ったら、Gerritというコードレビュー用のツールを動かします。)
→すまん次回から。

Agenda2/4

- ・この全4回の授業の全体像

第1回.iOSでの画像/アニメーション/TableViewとIBのイロハ

iOSでのアニメーションの仕組みとか、グラフィックリソースの持ち方とか、
テーブルについてのTipsを1日で紹介するよ！ けっこうハードだよククク

第2回.CoreDataなにそれ死ぬわ

iOSでのセーブとロードについて、一番強烈なプログラム部分。
これが出来れば大体何でも出来る。
永続化について行いたいだけなのに、フェッチとかシリアル化とかクエリとか、、わけがわからないよ！
っていうの相手に戦う。

第3回.サウンド鳴らす、メッセージングについて

サウンドは、CoreData次第ではやんないかもしないよ！ 簡単すぎて。
メッセージングは、テ戻りの少ないプログラム作りに於いて必須の、しかもお手軽なテクニック。
iOSを使うなら、是非知って欲しい。

第4回.通信してみようぜ

iOSでの通信をするよ！
簡単だけど、今後使えるかどうかはその人のスキル次第。
めちゃめちゃメモリリークするって言う人、言わない人の差

Agenda3/4

- ・この授業の日程

4/2 第1回.iOSでの画像/アニメーション/TableViewとIBのイロハ

iOSでのアニメーションの仕組みとか、グラフィックリソースの持ち方とか、
テーブルについてのTipsを1日で紹介するよ！ けっこうハードだよククク

4/16 第2回.CoreData

iOSでのセーブとロードについて、一番強烈なプログラム部分。
これが出来れば大体何でも出来る。
永続化について行いたいだけなのに、フェッチとかシリアル化とかクエリとか、、わけがわからないよ！
っていうの相手に戦う。

4/23 第3回.サウンド鳴らす、メッセージングについて(←募集時の日程とずれてるよ！！)

サウンドは、CoreData次第ではやんないかもしんないよ！ 簡単すぎて。

4/30 第4回.通信してみようぜ

iOSでの通信をちょっとだけ伝授するよ！
簡単だけど、今後使えるかどうかはその人のスキル次第。
めちゃめちゃメモリリークするって言う人、言わない人の差

Agenda4/4

本日のお題：

- 1.iOSでのグラフィックリソースの持ち方初歩、兼、IB/xibの使い方のウォーミングアップ
- 2.アニメーション無双
- 3.TableView無双

コレが出来れば相当な物

本日予想される成果：

- UIImageを使いこなす
- iOSでのアニメーションの概念を理解する
- IB(インターフェースビルダー)のxibの中級概念マスター、どんなインターフェースを作ってももう何も怖くない
- UITableViewについて、多分どこよりも正確で正しい理解が出来る、使える
- 本日のorz

地震でハアハアしてる間に、Xcode4が、でちゃった、、、
説明資料は3で作っちゃってるので、えーっと、なんだ、アレだ、スマン。

Drive 1/57

I.iOSでのグラフィックリソースの
持ち方初步、兼、
IB/xibの使い方のウォーミングアップ

Drive2/57

1-0-0.グラフィックの使い方

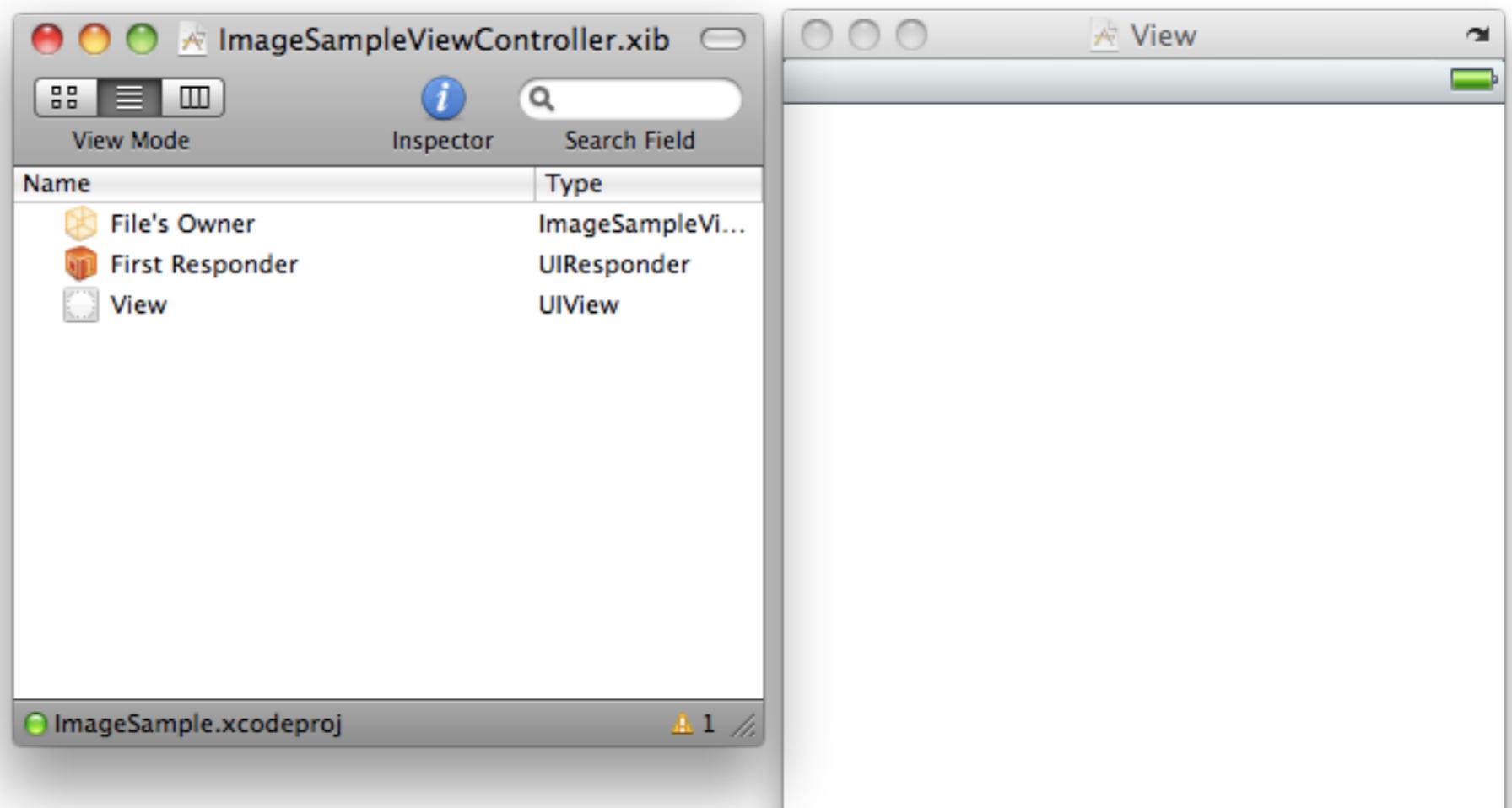
jpgとかpngとか、tiffとか、、、放り込めれば大体使える

ImageViewでもButtonでも

プロジェクトを新規作成して、xib付きでViewControllerを追加してみよう

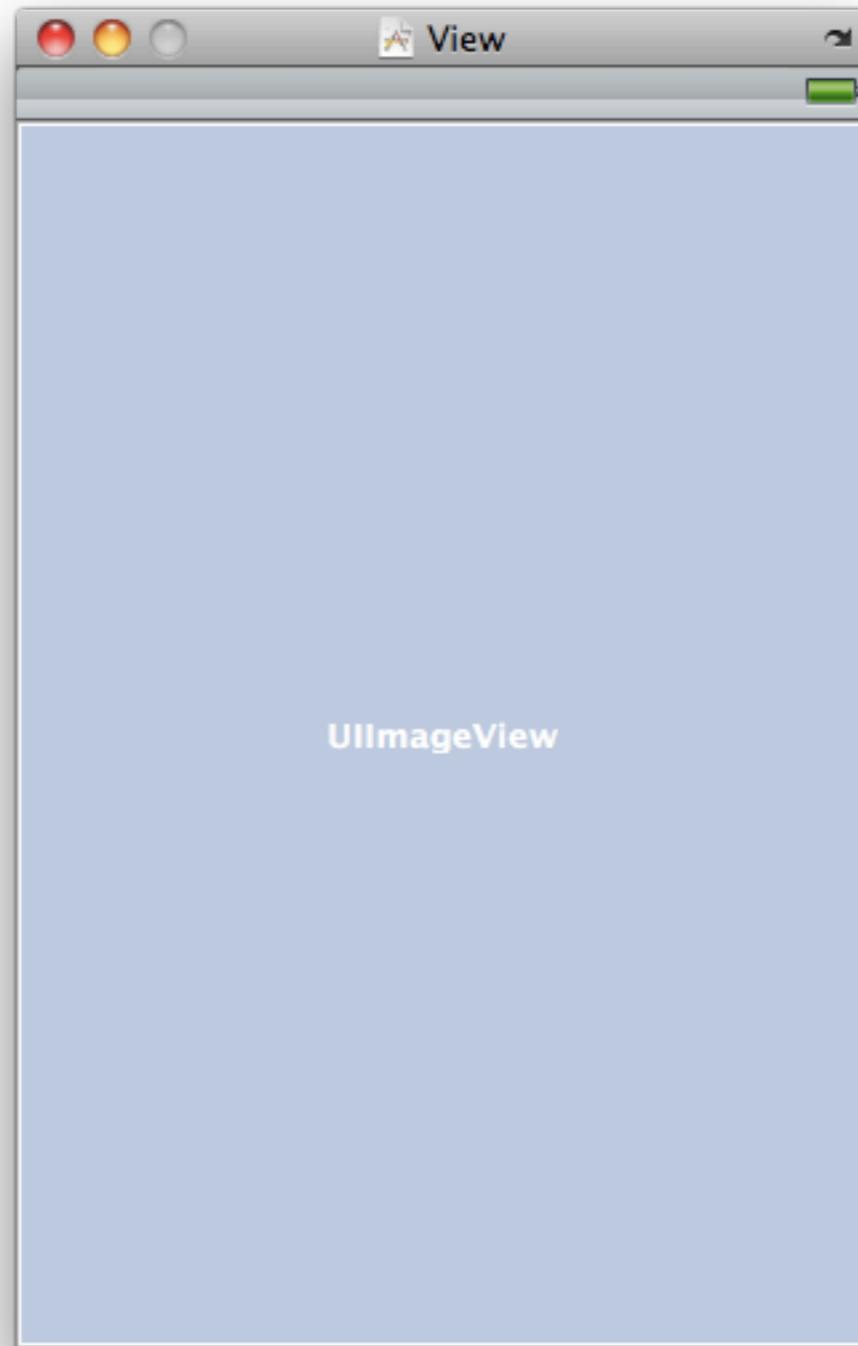
[ImageSampleViewController](#)という名前で追加

→ xibを開いてみたところ



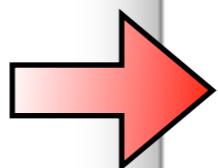
Drive3/57

1-0-1.IBのLibraryから、UIImageViewを置く

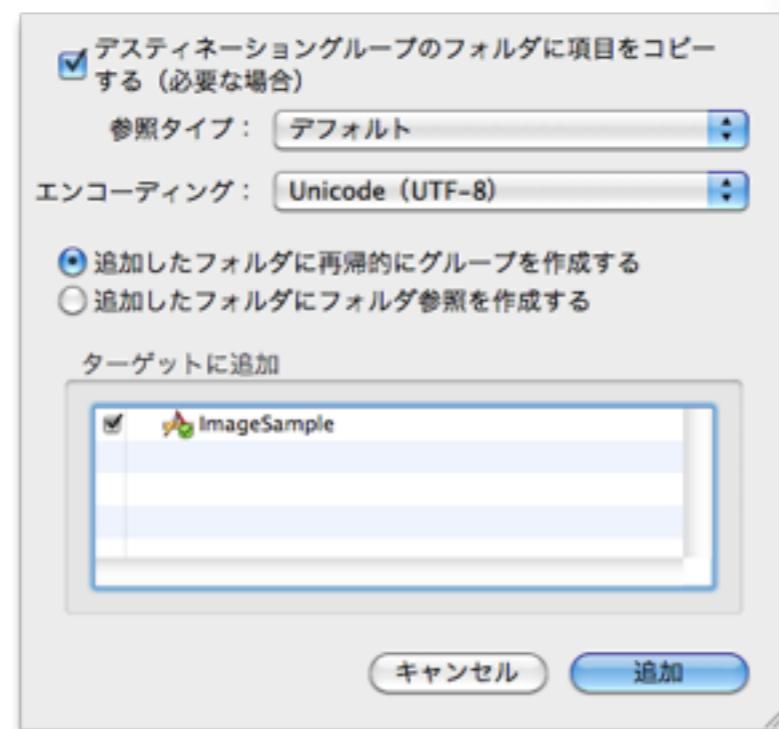


Drive4/57

1-0-2. なんでもいいから画像リソースを一枚、Xcodeに放り込む



☆コピーする、
に
チェック！



The Xcode Project Navigator shows the 'ImageSample' project structure:

- ImageSample (group)
 - Classes
 - ImageSampleAppDelegate.h
 - ImageSampleAppDelegate.m** (selected)
 - ImageSampleViewController.h
 - ImageSampleViewController.m
 - ImageSampleViewController.xib
 - Other Sources
 - Resources
 - Frameworks
 - Products
- ターゲット
- 実行可能ファイル
- 検索結果
- ブックマーク
- SCM
- プロジェクトのシンボル
- 実装ファイル
- Interface Builder ファイル

The right panel shows the code editor for 'ImageSampleAppDelegate.m' with the following content:

```
// ImageSample
// Created by Toru ...
// Copyright 2011 K ...

#import "ImageSampleAppDelegate.h"

@implementation ImageSampleAppDelegate

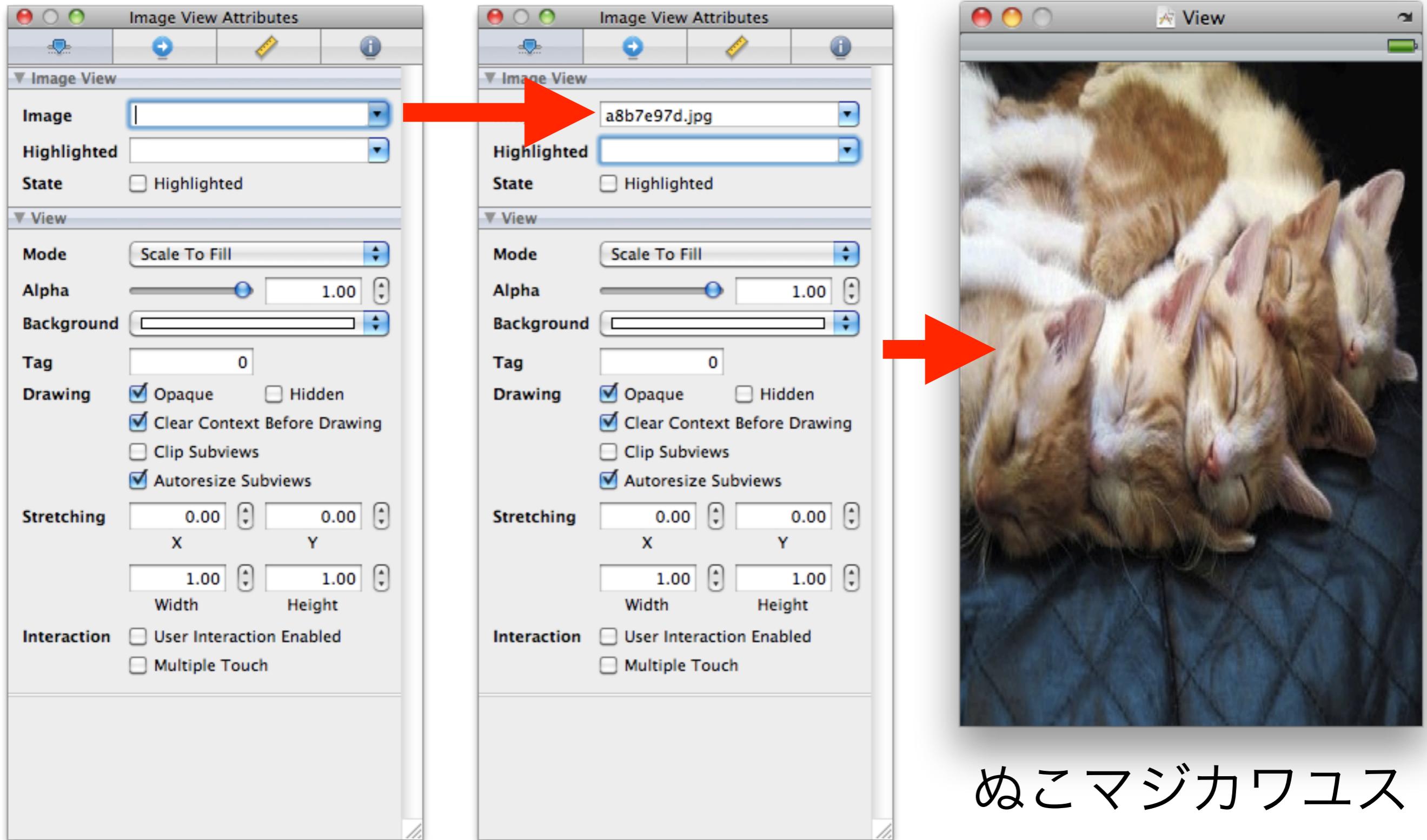
@synthesize window;

#pragma mark -
#pragma mark Application Delegate Methods

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    viewCont = [[UIImageView alloc] initWithImage:[UIImage imageNamed:@"kittens.jpg"]];
    [window addSubview:viewCont];
    NSTimer *timer = [NSTimer scheduledTimerWithTimeInterval:1.0 target:self selector:@selector(timerAction:) userInfo:nil repeats:YES];
}
```

Drive5/57

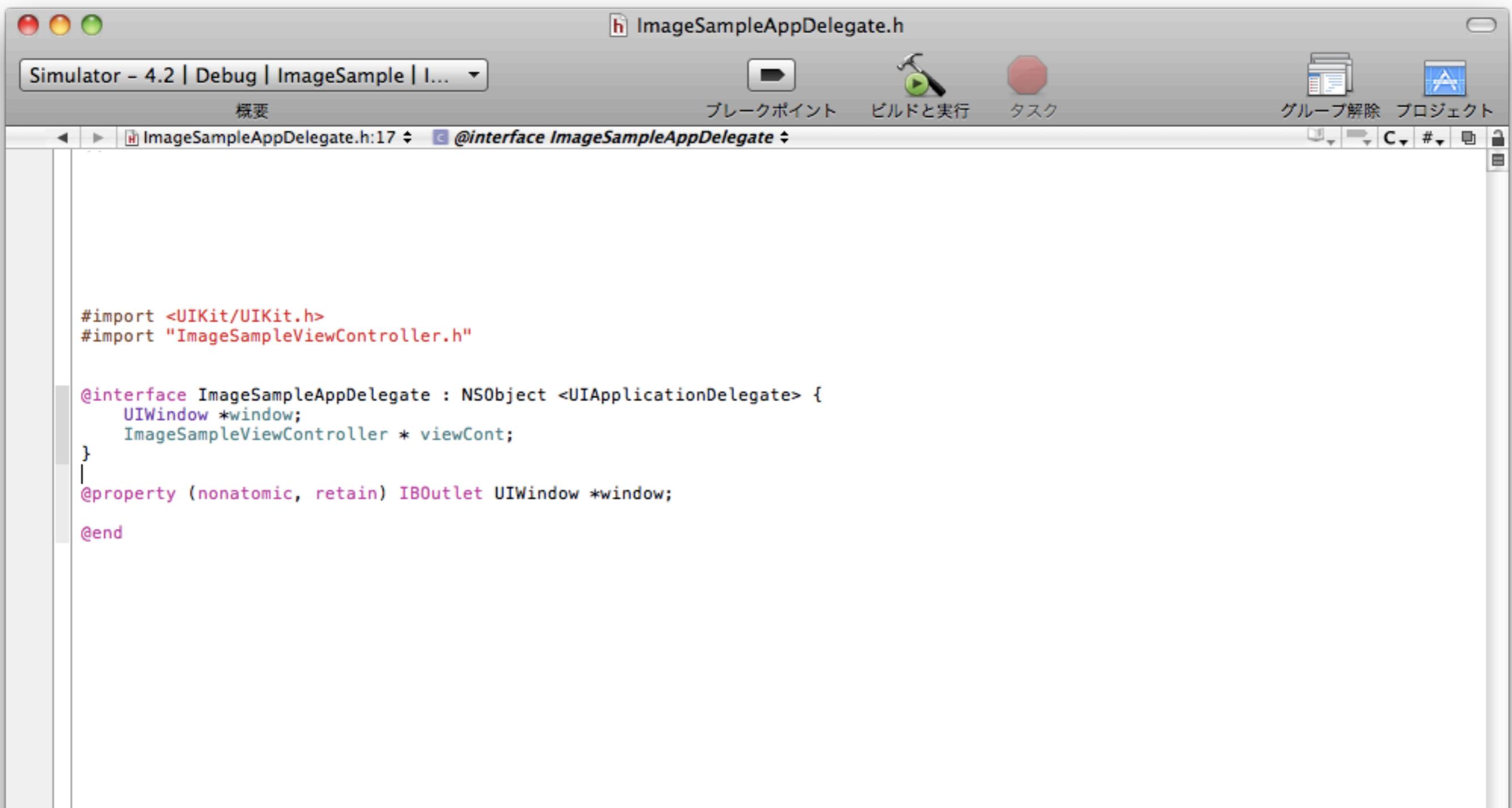
1-0-3.IBから画像を選ぶ→表示される



ぬこマジカワユス

Drive6/57

1-0-4.AppDelegateの.hとmに、windowにセットするよ！ ってコード書く
まずは.h



The screenshot shows the Xcode interface with the title bar "ImageSampleAppDelegate.h". The toolbar includes standard icons for running, stopping, building, and switching between tabs. The main editor area displays the following code:

```
#import <UIKit/UIKit.h>
#import "ImageSampleViewController.h"

@interface ImageSampleAppDelegate : NSObject <UIApplicationDelegate> {
    UIWindow *window;
    ImageSampleViewController * viewCont;
}
@property (nonatomic, retain) IBOutlet UIWindow *window;
@end
```

Drive7/57

1-0-5. .m

The screenshot shows the Xcode IDE with the file `ImageSampleAppDelegate.m` open. The window title is `ImageSampleAppDelegate.m`. The toolbar includes standard icons for running, stopping, and building. The menu bar shows "Simulator - 4.2 | Debug | ImageSample | I...". The code editor displays the following implementation:

```
#import "ImageSampleAppDelegate.h"

@implementation ImageSampleAppDelegate
@synthesize window;

#pragma mark -
#pragma mark Application lifecycle

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    viewCont = [[ImageSampleViewController alloc] init];
    [window addSubview:viewCont.view];
    [self.window makeKeyAndVisible];
    return YES;
}

- (void) applicationWillResignActive:(UIApplication *)application {
    /*
        Sent when the application is about to move from active to inactive state. This can occur for certain types of temporary interruptions, such as an incoming phone call or SMS message. Applications should not use this method to pause their execution.
    */
}
```

Drive8/57

1-0-6. 実行したら動く、で解説

画像は、

Xcodeにリソースを放り込んだ時点で、IBからの選択肢として現れる。

IBからは、Xcodeに放り込まれたデータを、拡張子とかで判断して見てる。

UIImageView以外にも、

imageを貼付けられるオブジェクトは沢山有る。

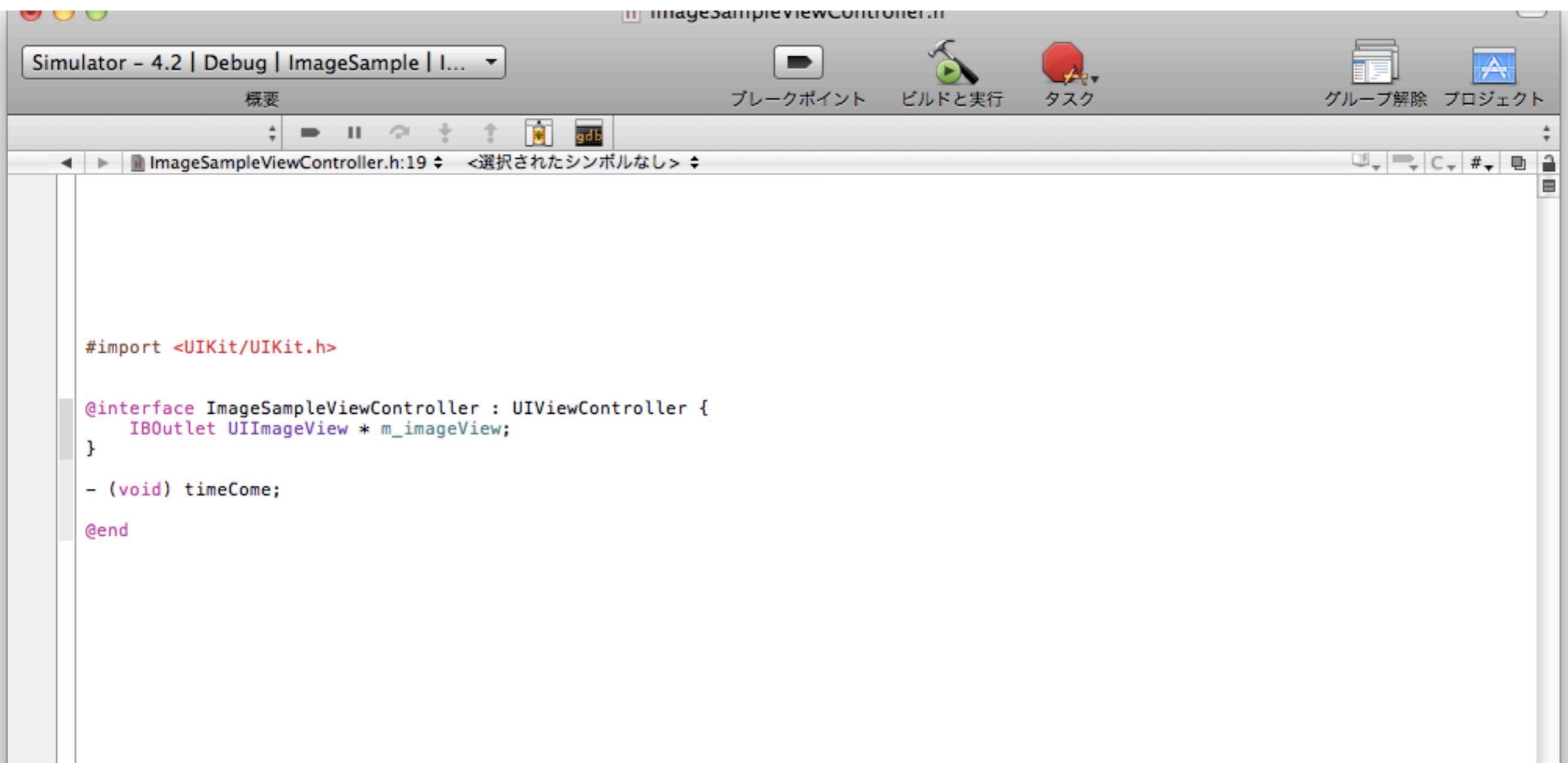
UIButtonとか、UITableViewCellとか。

これらは、UIImageViewを内包している

(内部にUIImageViewのオブジェクトを予め持っている)事がほとんど。

Drive9/57

1-0-7.さっちはIBで画像置いたが、今度は画像をコードで変化させてみよう
先ほどのxibファイルの中で、imageを表示させている要素のインスタンスを、
コード側にIBOutletとして書こう。
(IBOutletとして書く事で、IBから選択する事が出来るようになる。)



The screenshot shows the Xcode interface with the title bar "ImageSampleViewController.h" and the status bar "Simulator - 4.2 | Debug | ImageSample | ...". The code editor displays the following code:

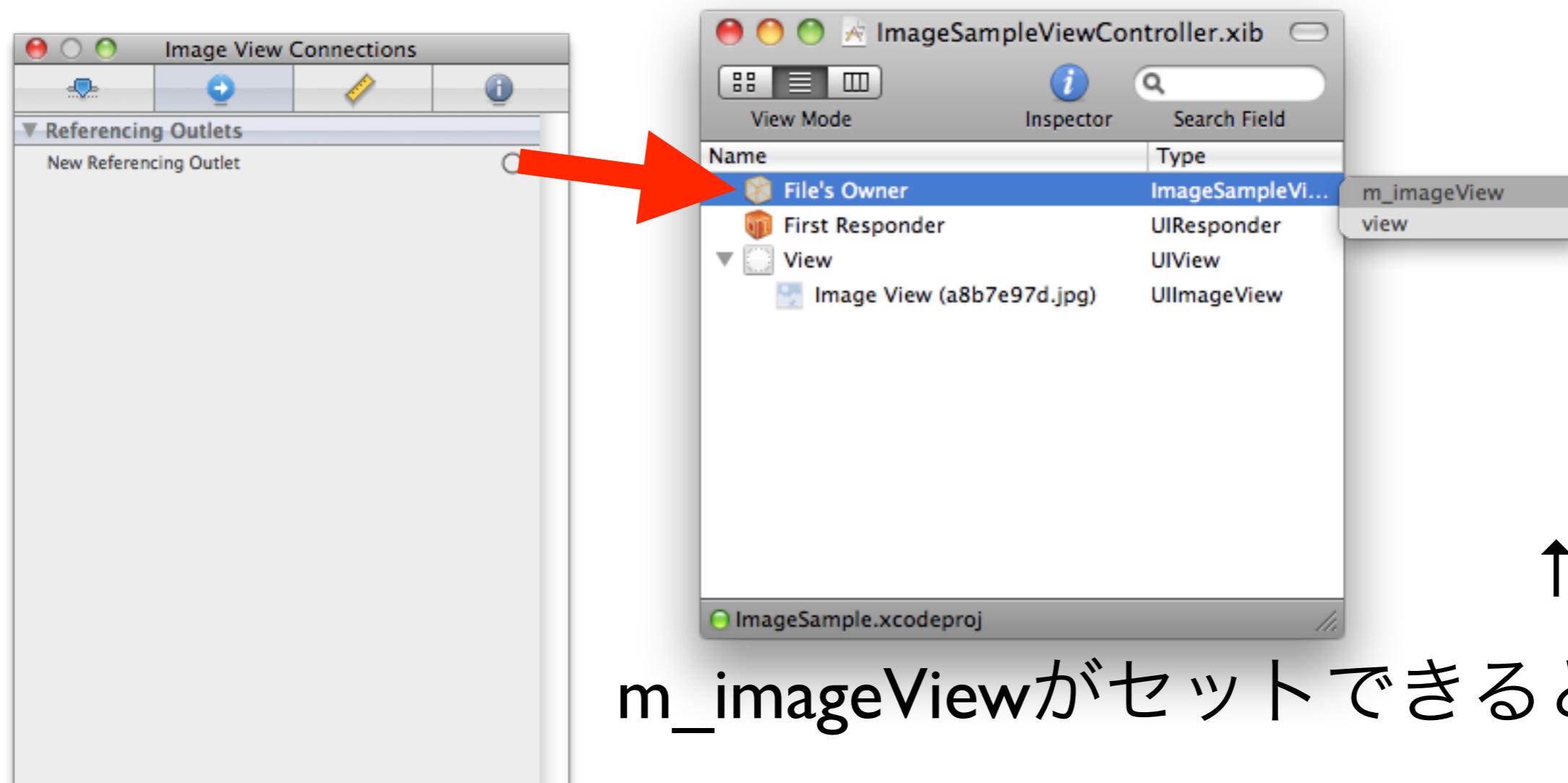
```
#import <UIKit/UIKit.h>

@interface ImageSampleViewController : UIViewController {
    IBOutlet UIImageView *m_imageView;
}
- (void) timeCome;
@end
```

Drive | 0/57

1-0-8. IBで、imageが貼ってあるオブジェクトを選択した状態で、
ウインドウ > Inspector > Referencing Outlets > New Referencing Outlet
をドラッグ、File's Ownerへと繋ごう。

これで、Xcode上のm_imageViewインスタンスと、
IB上のimageが貼ってあるインスタンスがリンク(イコールの状態)になる。



Drive ||/57

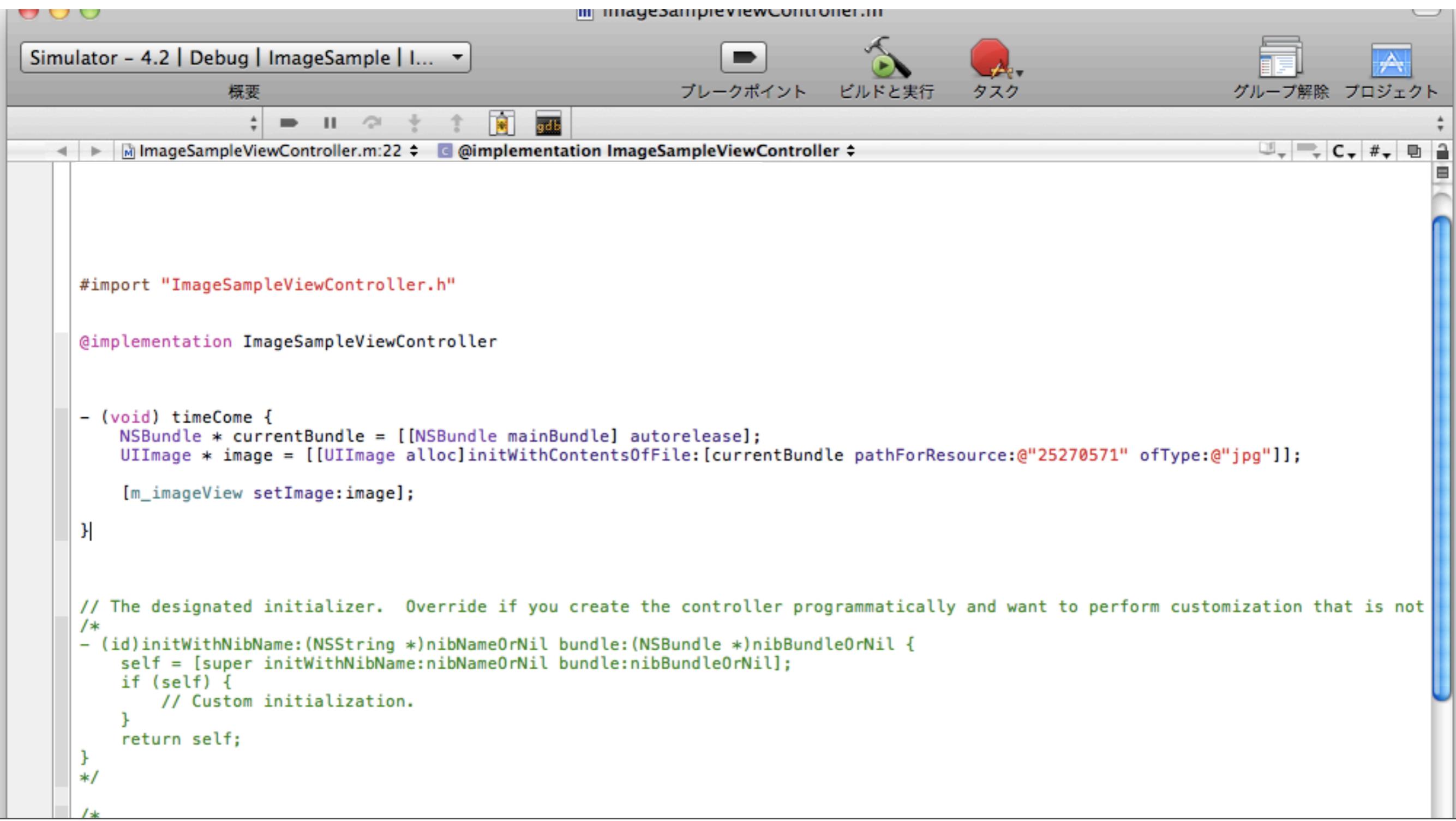
1-0-9.切り替わった後の画像を用意しよう
Xcodeにほうりこむだけ。



Drive | 2/57

1-0-10. 時間で切り替わるようにしようかね

ImageSampleViewControllerとDelegateにコード書く



The screenshot shows the Xcode IDE interface with the following details:

- Title Bar:** Displays "Simulator - 4.2 | Debug | ImageSample | I...".
- Toolbar:** Includes icons for Breakpoint (red arrow), Build & Run (green play button), Task (red octagon), Group Unwrap (blue folder), and Project (blue A).
- Code Editor:** The file "ImageSampleViewController.m" is open at line 22. The code implements a timer to change images:

```
#import "ImageSampleViewController.h"

@implementation ImageSampleViewController

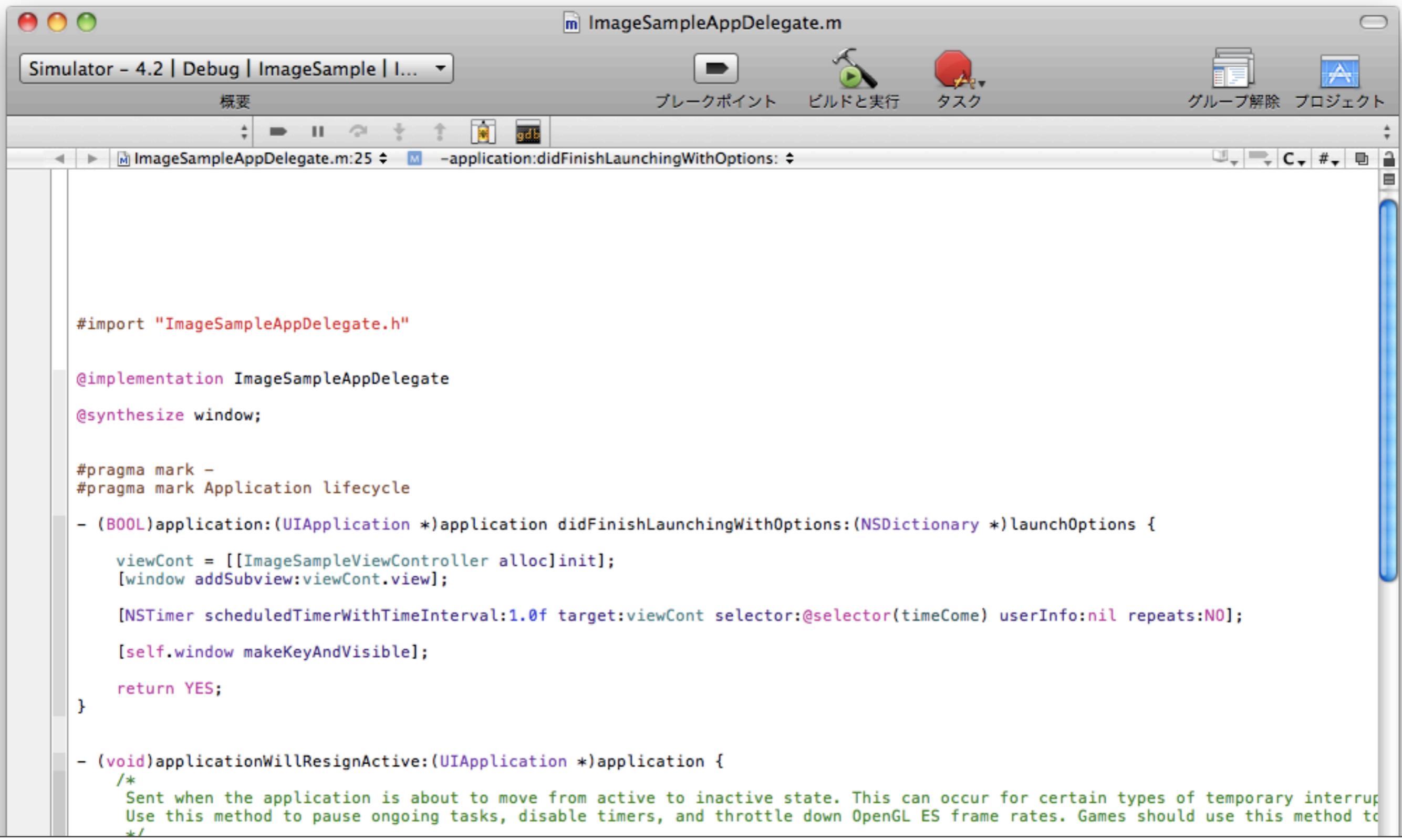
- (void) timeCome {
    NSBundle * currentBundle = [[NSBundle mainBundle] autorelease];
    UIImage * image = [[UIImage alloc] initWithContentsOfFile:[currentBundle pathForResource:@"25270571" ofType:@"jpg"]];
    [m_imageView setImage:image];
}

// The designated initializer. Override if you create the controller programmatically and want to perform customization that is not
/*-
- (id)initWithNibName:(NSString *)NibNameOrNil bundle:(NSBundle *)nibBundleOrNilOrNil {
    self = [super initWithNibName:nibNameOrNilOrNil bundle:nibBundleOrNilOrNil];
    if (self) {
        // Custom initialization.
    }
    return self;
}
*/

```

Drive | 3/57

1-1-0. Delegate.mに、Timerを書いた。1秒後にviewContのtimeCome!



```
#import "ImageSampleAppDelegate.h"

@implementation ImageSampleAppDelegate
@synthesize window;

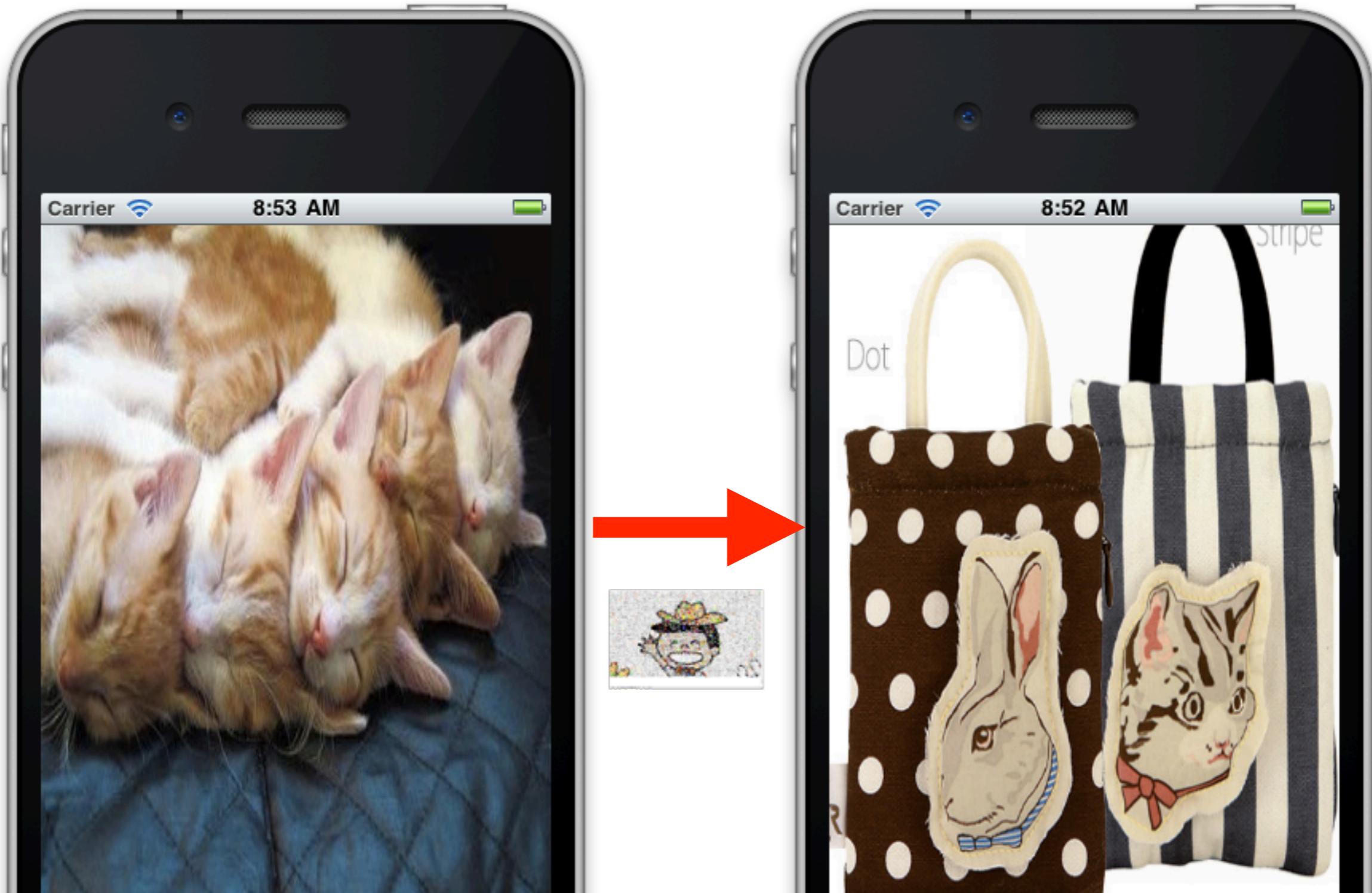
#pragma mark -
#pragma mark Application lifecycle

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    viewCont = [[ImageSampleViewController alloc] init];
    [window addSubview:viewCont.view];
    [NSTimer scheduledTimerWithTimeInterval:1.0f target:viewCont selector:@selector(timeCome) userInfo:nil repeats:NO];
    [self.window makeKeyAndVisible];
    return YES;
}

- (void)applicationWillResignActive:(UIApplication *)application {
    /*
     Sent when the application is about to move from active to inactive state. This can occur for certain types of temporary interruptions. Use this method to pause ongoing tasks, disable timers, and throttle down OpenGL ES frame rates. Games should use this method to pause.
    */
}
```

Drive | 4/57

1-1-1.メソッドで書き変わるのが判ったと思う。
イメージは、こんな風に扱える。



Drive | 5/57

1-2-0.扱えるリソースの種類については、ドキュメント参照のこと。

The screenshot shows a Mac OS X-style window titled "UIImage Class Reference". The window has standard OS X controls at the top left and a search bar at the top right. The main content area is titled "UIImage Class Reference". On the left, there's a sidebar with a "Table of Contents" section containing links like Overview, Tasks, Properties, Class Methods, Instance Methods, and Constants. Below the sidebar is a list of instance methods. The main content area contains text about drawing large images and a section titled "Supported Image Formats" with a table listing file formats and their extensions.

an image larger than 1024 x 1024 pixels by drawing it to a bitmap-backed graphics context. In fact, you may need image in this manner (or break it into several smaller images) in order to draw it to one of your views.

Supported Image Formats

Table 1 lists the file formats that can be read by the `UIImage` class.

Table 1 Supported file formats

Format	Filename extensions
Tagged Image File Format (TIFF)	.tiff, .tif
Joint Photographic Experts Group (JPEG)	.jpg, .jpeg
Graphic Interchange Format (GIF)	.gif
Portable Network Graphic (PNG)	.png
Windows Bitmap Format (DIB)	.bmp, .BMPf
Windows Icon Format	.ico
Windows Cursor	.cur
XWindow bitmap	.xbm

Drive | 6/57

2.アニメーション無双

Drive | 7/57

2-0-0.iOSでのアニメーション

☆原理の解説

☆フレームレートが無い、、だと、、?

まず解説

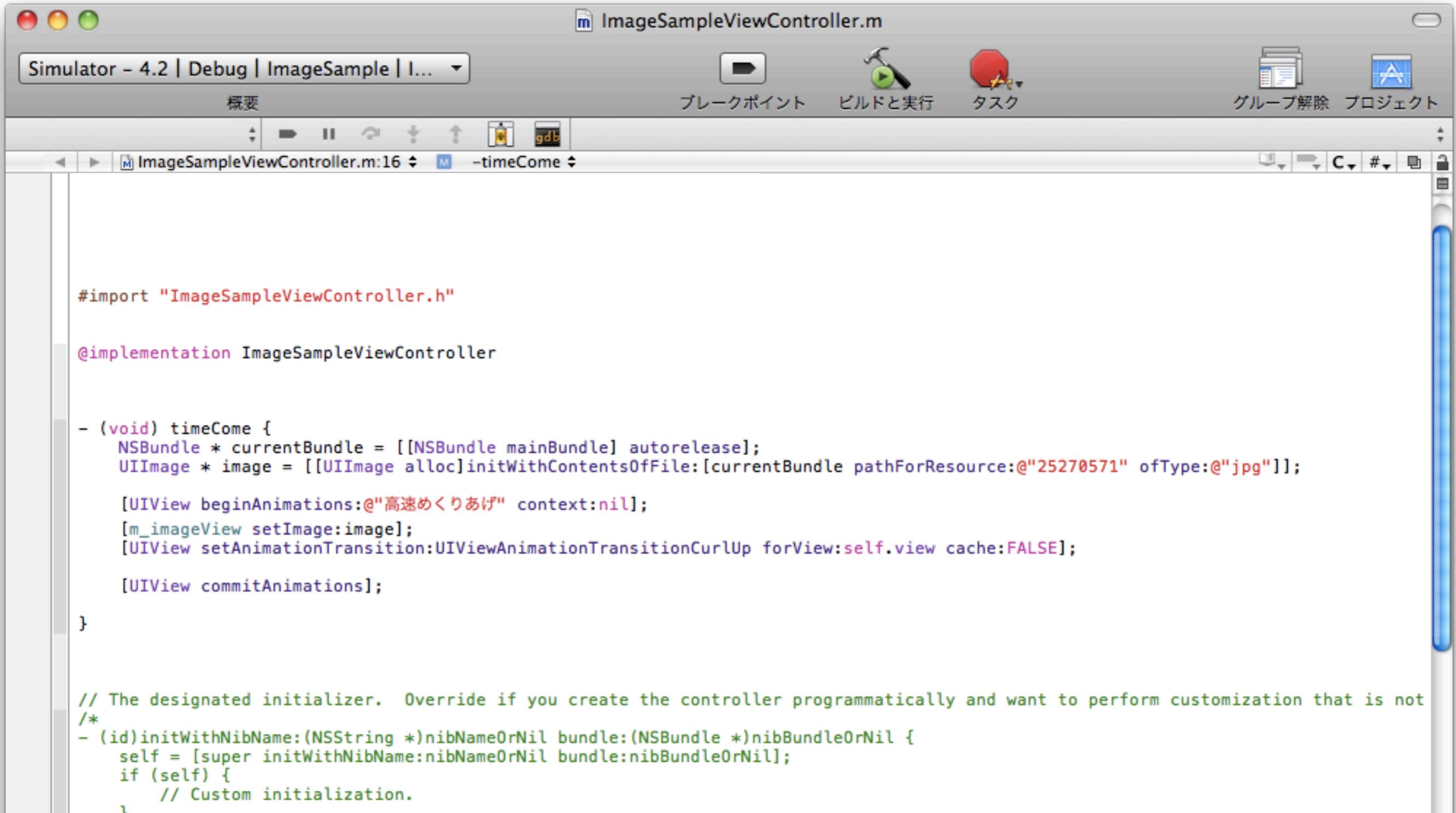
iOSでは、プログラマが明示的に「このパートを秒間何十回描画しろ」みたいな命令は、あまり頻繁に出てきません。

ゲームとかだと、「この画面のこのパートを60FPSで描画しろ！」
=一秒間に60回、画面を書け！

とかあり得るんですが、iOSでは、
画面に置かれたパートをいつ、どう描画するかは、
大体iOS任せになります。

Drive | 8/57

2-0-1. 例として、簡単なアニメーションのコードを書いてみよう。
timeComeメソッドに3行、書き加える



The screenshot shows the Xcode IDE interface with the file `ImageSampleViewController.m` open. The code editor displays the implementation of the `ImageSampleViewController` class, specifically focusing on the `- (void) timeCome` method. The code adds three lines of animation logic to the existing implementation. The Xcode toolbar at the top shows the simulator set to version 4.2 in debug mode, and the code editor shows the current line of code being edited.

```
#import "ImageSampleViewController.h"

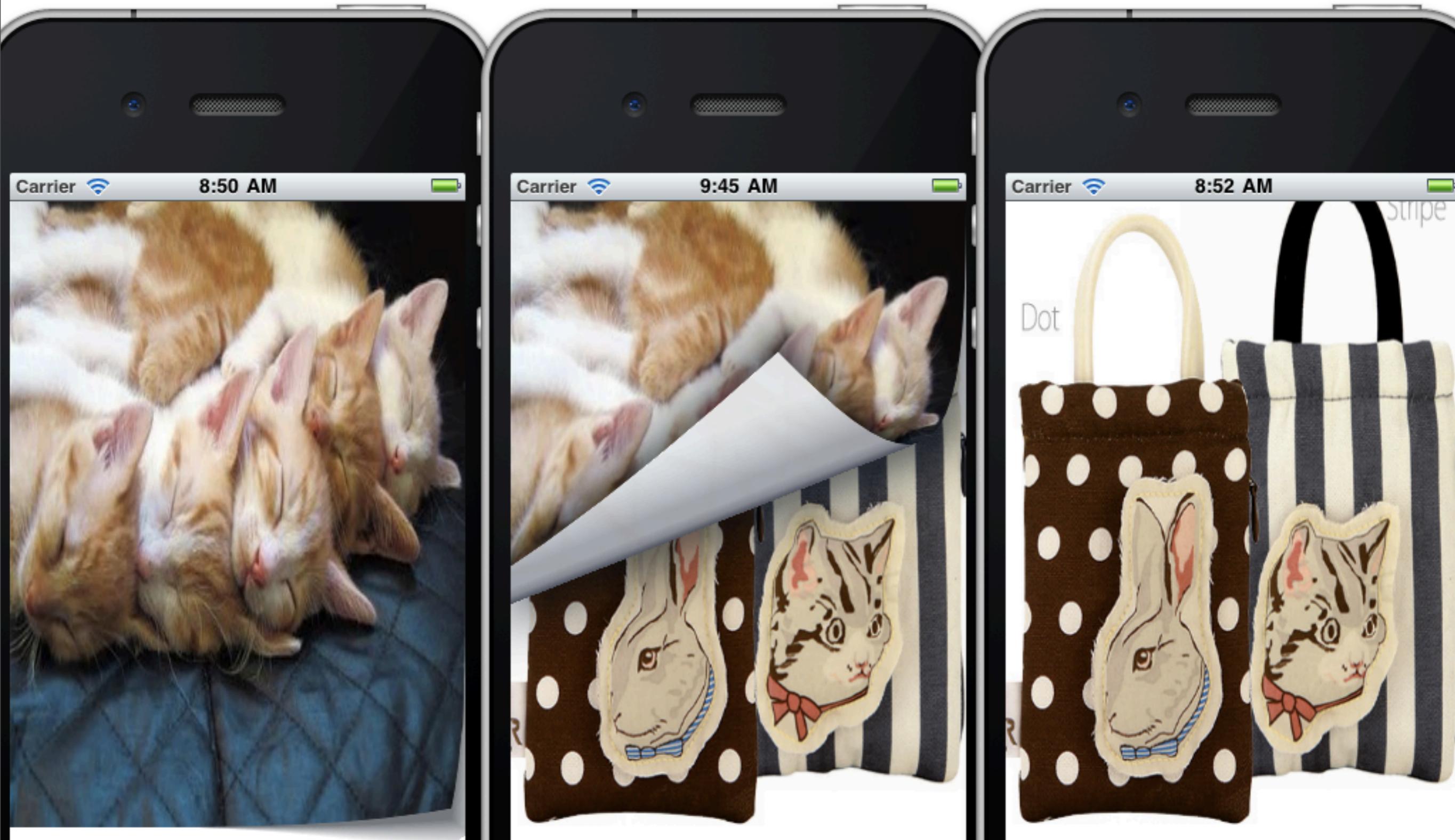
@implementation ImageSampleViewController

- (void) timeCome {
    NSBundle * currentBundle = [[NSBundle mainBundle] autorelease];
    UIImage * image = [[UIImage alloc] initWithContentsOfFile:[currentBundle pathForResource:@"25270571" ofType:@"jpg"]];
    [UIView beginAnimations:@"高速めくりあげ" context:nil];
    [m_imageView setImage:image];
    [UIView setAnimationTransition:UIViewAnimationTransitionCurlUp forView:self.view cache:FALSE];
    [UIView commitAnimations];
}

// The designated initializer. Override if you create the controller programmatically and want to perform customization that is not
/* - (id)initWithNibName:(NSString *)NibNameOrNil bundle:(NSBundle *)nibBundleOrNilOrNil {
    self = [super initWithNibName:nibNameOrNilOrNil bundle:nibBundleOrNilOrNil];
    if (self) {
        // Custom initialization.
    }
}
```

Drive | 9/57

2-0-2. UIAnimationが、あっさりと慮ってくれます



Drive20/57

2-1-0.アニメーションの解説

このアニメーション機構は、下記の前提、原理を持ったものになっています

- ・iPhone上で見えるすべての要素は、UIViewを基礎にして実現されている。
- ・UIViewに対して、

　　クラスメソッド beginAnimation と、
　　commitAnimationの間に書かれたアクションを、
　　指定されたアニメーションのビフォア/アフターのように、
　　二重に用意し、それらをエフェクトとともにに入れ替える。
　　パッと見すげー！みたいな事を実現している。

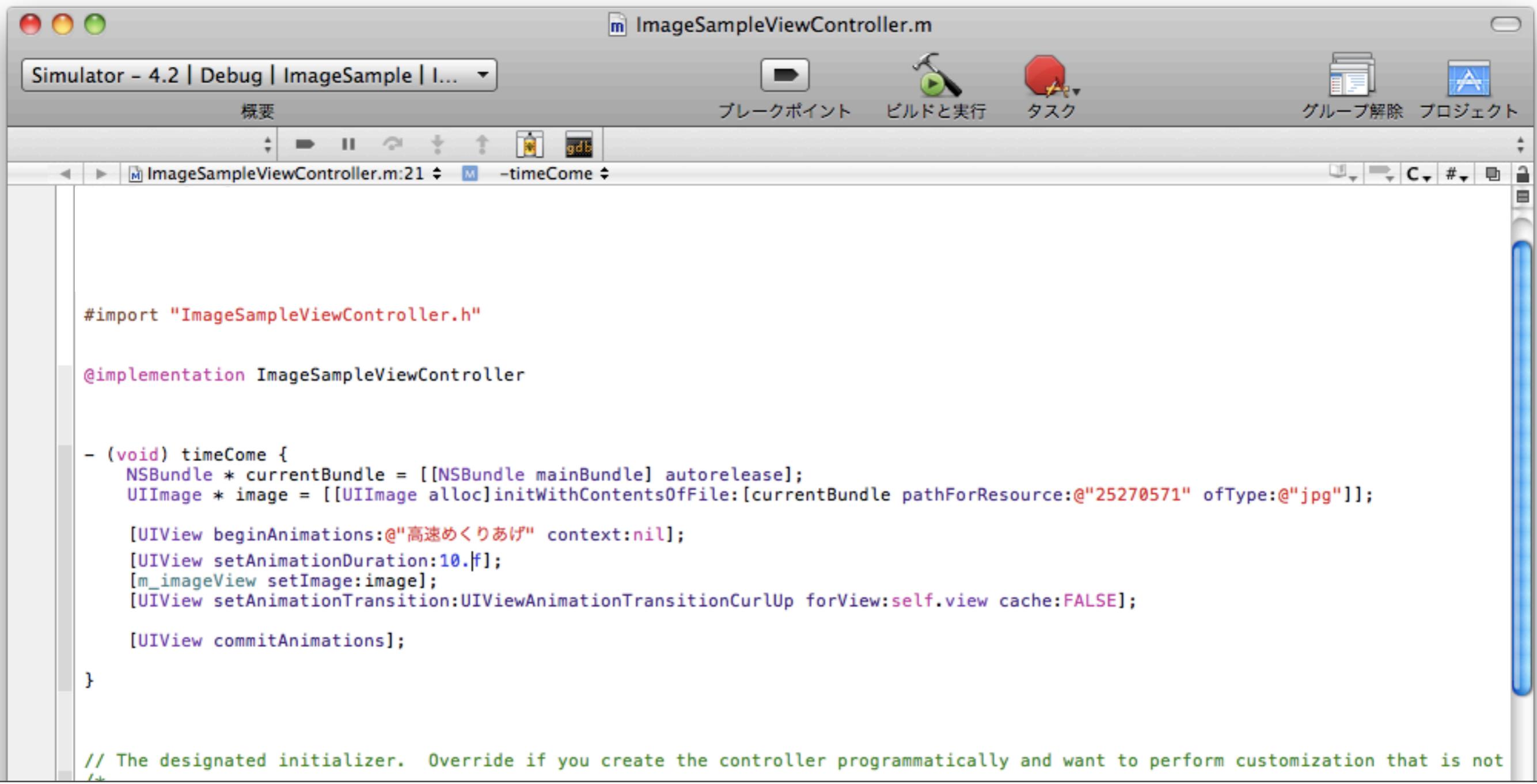
→アニメ後ビューの見た目を用意し、その上に
　　アニメ前ビューの物をアニメーションさせながら表示する、
　　というテクニックの合わせ技。

= 2枚のビューを持ってなきゃ出来ないけど、
　　とりあえず3行書くとなんとかしてくれる。

Drive2 | 57

2-1-1. アニメーションに、何秒かけるか

iOSでは、下記のような1行で、アニメーションにかかる時間を指定出来ます。



The screenshot shows the Xcode IDE interface with the file `ImageSampleViewController.m` open. The code implements a method `- (void) timeCome` which performs a view animation. The animation duration is set to 10.0 seconds using the `setAnimationDuration:` method.

```
#import "ImageSampleViewController.h"

@implementation ImageSampleViewController

- (void) timeCome {
    NSBundle * currentBundle = [[NSBundle mainBundle] autorelease];
    UIImage * image = [[UIImage alloc] initWithContentsOfFile:[currentBundle pathForResource:@"25270571" ofType:@"jpg"]];

    [UIView beginAnimations:@"高速めくりあげ" context:nil];
    [UIView setAnimationDuration:10.0f];
    [m_imageView setImage:image];
    [UIView setAnimationTransition:UIViewAnimationTransitionCurlUp forView:self.view cache:NO];
    [UIView commitAnimations];
}

// The designated initializer. Override if you create the controller programmatically and want to perform customization that is not
// needed for normal usage.
```

Drive22/57

2-1-2.アニメーションの結論として、

- ・アニメーションのタイミング、アニメーションにかかる時間などは指定できるが、その間何回描画するか、という設定は、見事に存在しない。

→美麗なアニメーションとかを実行しても、プログラマが”何回描画するか”という部分に苦心しないでもいい構造になっている。
→描画のパフォーマンスチューニングはOSがやってくれてる！

結果として、iOSでは、OS側がアニメーション等を慮ってくれる為、フレームレートという概念が通常有りません。

*但し、これはUIViewかそのinheritがあるクラスを利用している場合の前提で、OpenGLを使った描画等の場合は、フレームレートでの描画設定が必須になります。
(UIViewがサポートしてないんです。)

→GLの描画を行うビューの名称は、GLESView系。 うん、UI関連じゃない。

Drive 1/57

3. TableView 無双
コレが出来れば相当な物

Drive24/57

3-0-0.TableViewには、アニメーション、イメージ、xibがふんだんに使われています。



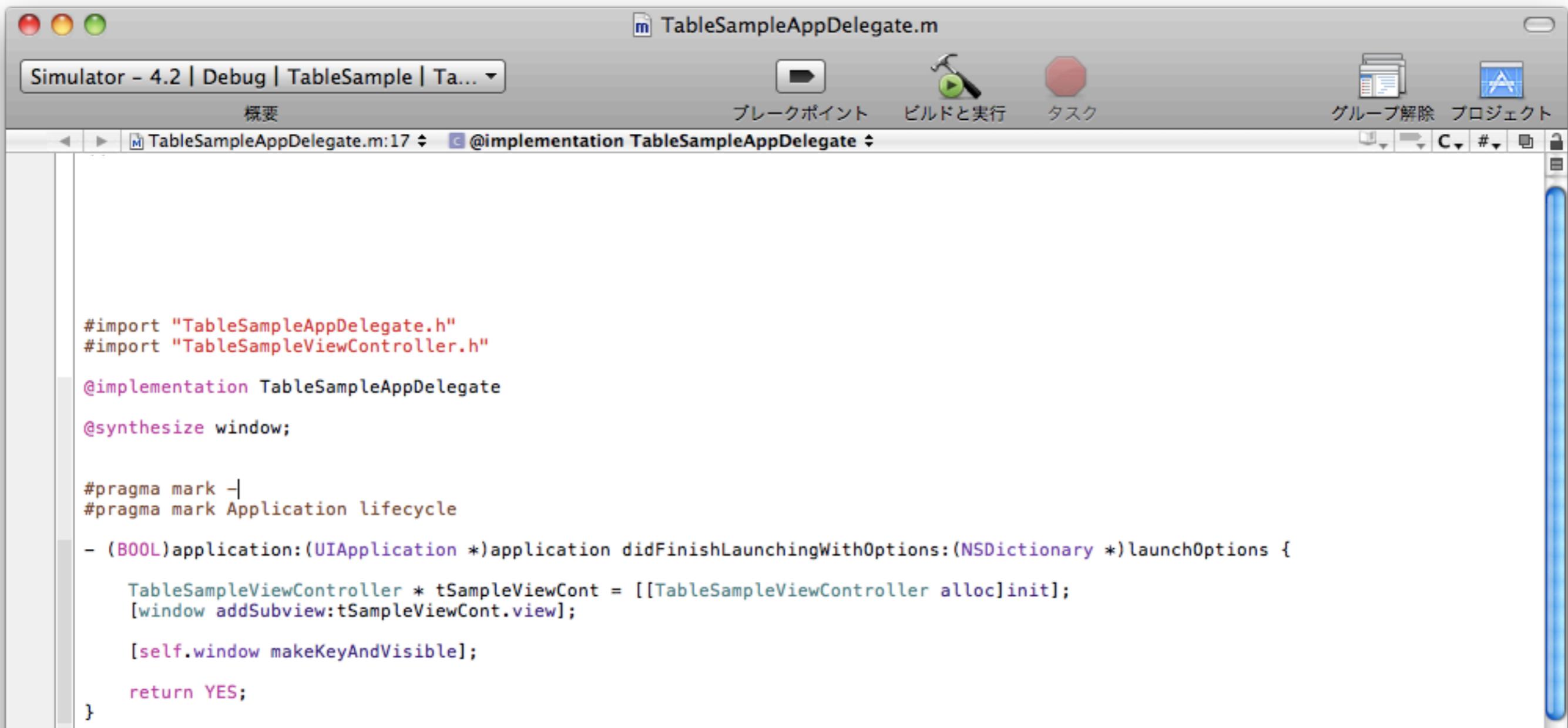
Drive25/57

3-0-1. UITableViewを作つてみよう1

TableSample プロジェクトを新規作成、

TableViewCellControllerをinheritしたクラス

TableSampleViewController を作り、このビューを画面に表示。



The screenshot shows the Xcode IDE interface with the following details:

- Title Bar:** Displays "TableSampleAppDelegate.m".
- Toolbar:** Shows icons for Breakpoint (red arrow), Build & Run (green play button), Task (red octagon), Group Unwrap (blue folder), and Project (blue A).
- Menu Bar:** Shows "Simulator - 4.2 | Debug | TableSample | Ta...".
- Editor Area:** Displays the code for TableSampleAppDelegate.m. The code implements theUIApplicationDelegate protocol and initializes a TableSampleViewController as the root view controller.

```
#import "TableSampleAppDelegate.h"
#import "TableSampleViewController.h"

@implementation TableSampleAppDelegate

@synthesize window;

#pragma mark -
#pragma mark Application lifecycle

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    TableSampleViewController * tSampleViewCont = [[TableSampleViewController alloc] init];
    [window addSubview:tSampleViewCont.view];
    [self.window makeKeyAndVisible];
    return YES;
}

@end
```

Drive26/57

3-0-2.起動する前にひと手間。このままだとエラーが出るんだわ
TableSampleViewController クラスは、

①- (**NSInteger**)numberOfSectionsInTableView:(**UITableView** *)tableView

②- (**NSInteger**)tableView:(**UITableView** *)tableView
numberOfRowsInSection:(**NSInteger**)section

メソッドで、それぞれセクション数と行数を返さないといけない。

この部分で、TableViewの中に表示されるデータ(セルという)の数がセットされる。sectionには1、Rowsには好きな数をセットすんべ、で起動すんべ。

UITableViewをドラッグしたりタップしたときのアニメーションも、勿論
UIViewがらみのテクニックで内部実装されている。
描画をOSが担当してくれるって、いいよね！

Drive27/57

3-0-2.こんな見た目



Drive28/57

3-0-3.じゃ、テーブルの中身=セルに情報を出そう。　　、、という前に解説
TableViewは、数万、数億行のデータであっても、なんとかそれを
表示出来るように実装されている。

どうやってやってるか？っていうと、

- ・画面に表示されるだけの箱をまず作り、
- ・その箱の中身を、現在表示している範囲の情報だけにする事で、
メモリの消費を最小限に抑えている。



}

実際、Rowsが何行であっても、



× 画面に表示される数+2,3個

=iPhoneの場合、

14,5個くらいのセルしかない。

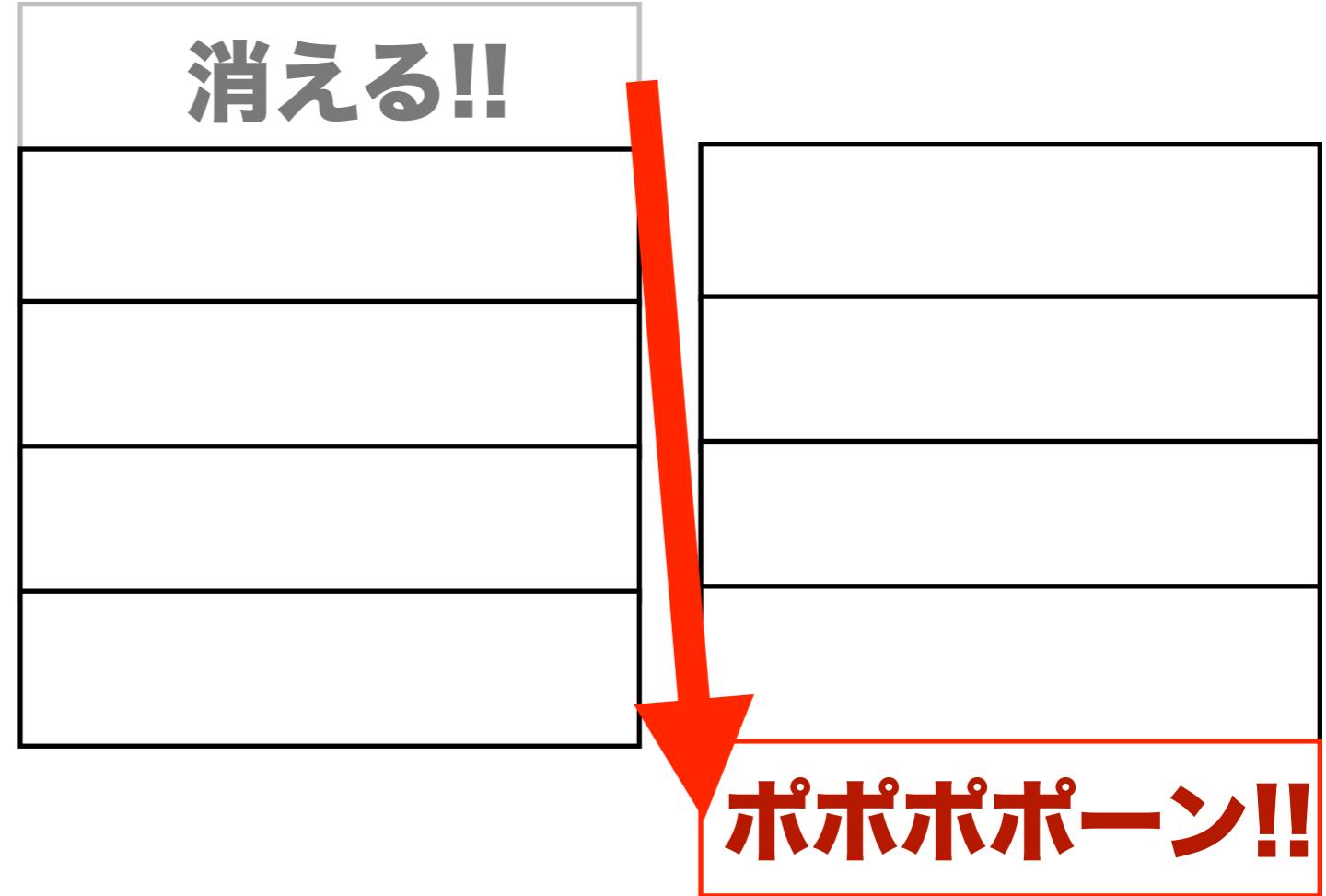
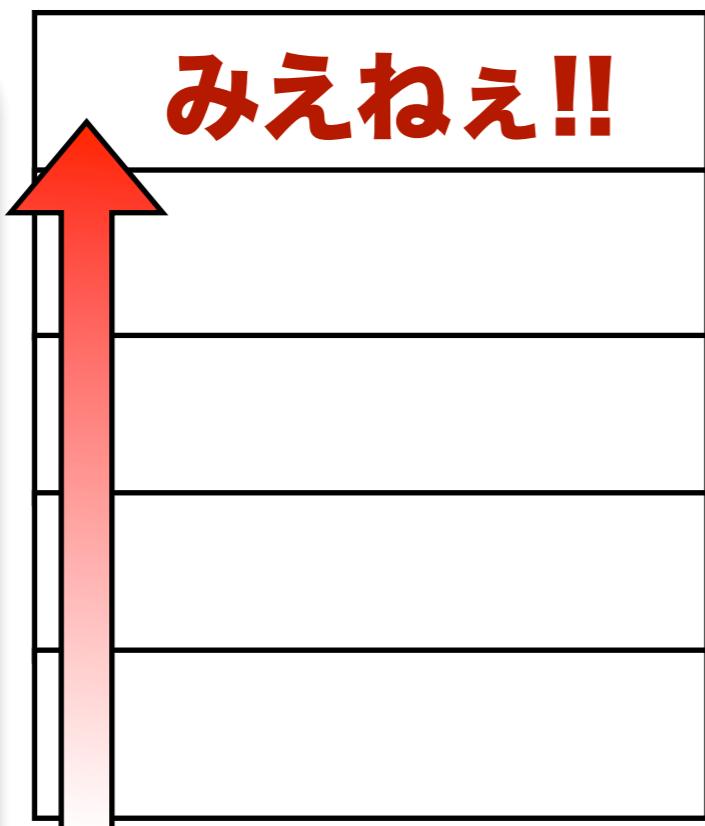
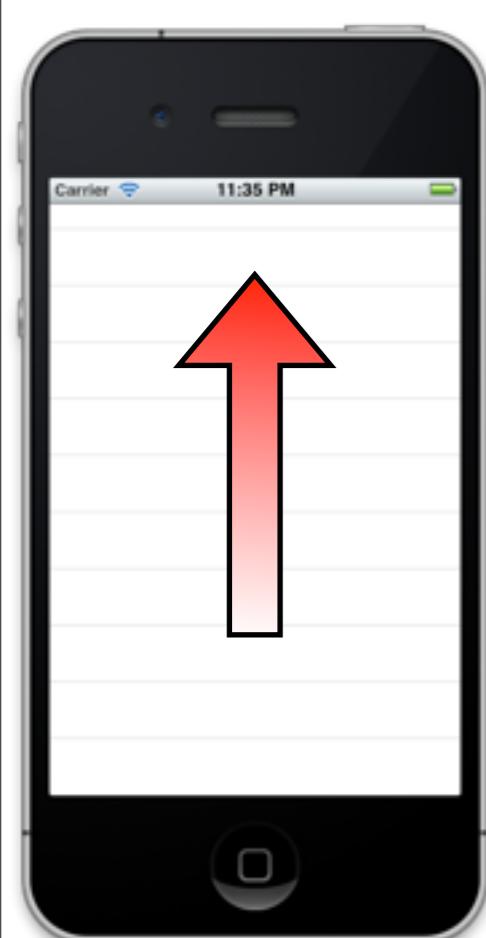
Drive29/57

3-0-4. そんな少なさで大丈夫かってさ。



これじゃない

スライドとかされて表示範囲が変わったら、
まずは移動方向と逆の位置の、**画面外に消えたセルを、**
移動方向から出すように"瞬間移動"させて、
セルの中身を、表示範囲に合わせて、変化させる、ということをやっている。



Drive30/57

3-0-5.入れ替えはこの1メソッドで全てやってる

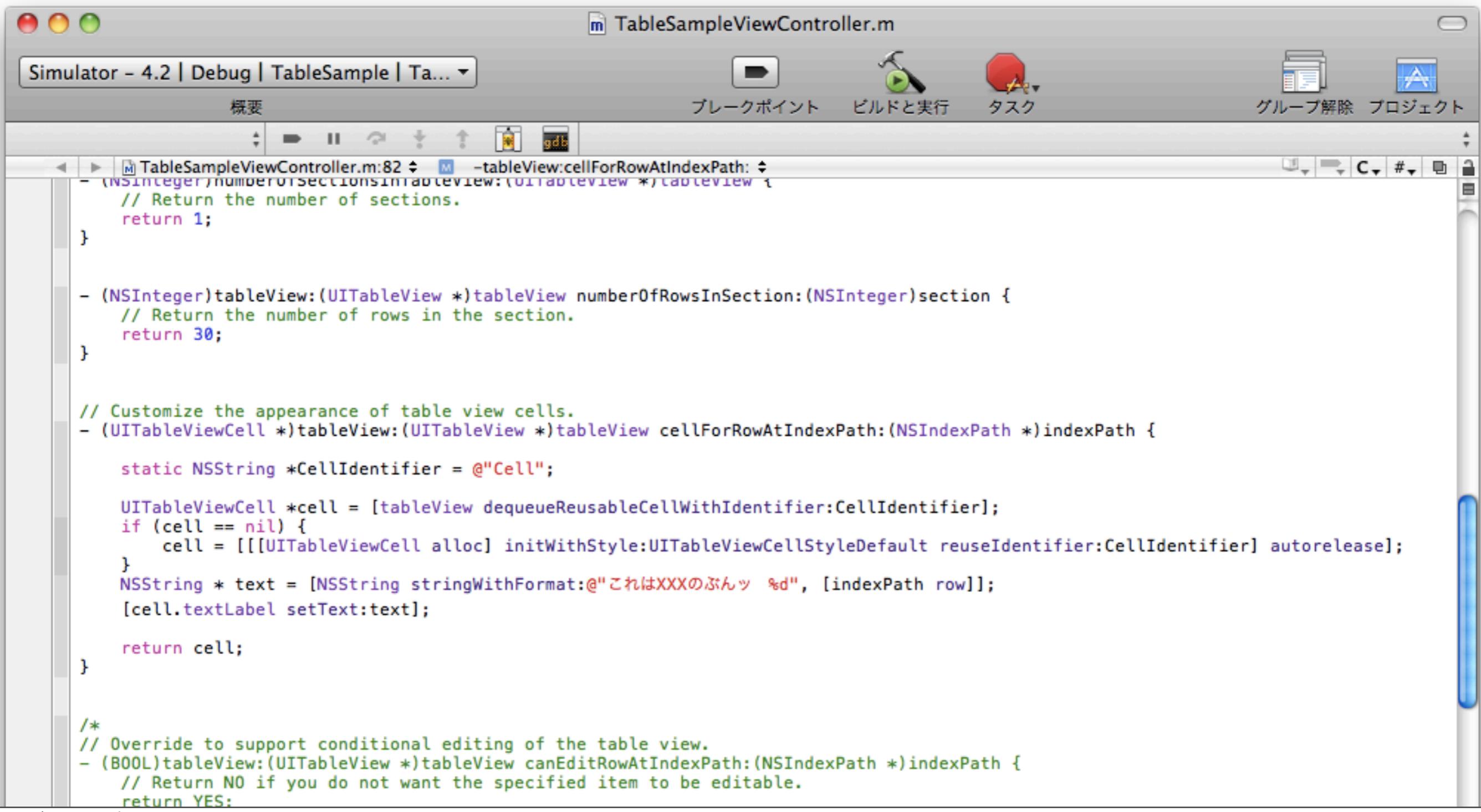
この2つの動作を一度にやっているメソッドが、

- (UITableViewCell *)tableView:(UITableView *)tableView
cellForRowAtIndexPath:(NSIndexPath *)indexPath

これ。

Drive3 | 57

3-0-6.セルの内容を変化させる
下記みたいな追記。



The screenshot shows the Xcode IDE interface with the file `TableSampleViewController.m` open. The code implements a UITableViewDataSource protocol to customize table view cells.

```
m TableSampleViewController.m
Simulator - 4.2 | Debug | TableSample | Ta...
概要 ブレークポイント ビルドと実行 タスク グループ解除 プロジェクト
TableSampleViewController.m:82 - tableView:cellForRowAtIndexPath:
- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView {
    // Return the number of sections.
    return 1;
}

- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section {
    // Return the number of rows in the section.
    return 30;
}

// Customize the appearance of table view cells.
- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath {
    static NSString *CellIdentifier = @"Cell";
    UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
    if (cell == nil) {
        cell = [[[UITableViewCell alloc] initWithStyle:UITableViewCellStyleDefault reuseIdentifier:CellIdentifier] autorelease];
    }
    NSString * text = [NSString stringWithFormat:@"これはXXXのぶん%d", [indexPath row]];
    [cell.textLabel setText:text];
    return cell;
}

/*
// Override to support conditional editing of the table view.
- (BOOL)tableView:(UITableView *)tableView canEditRowAtIndexPath:(NSIndexPath *)indexPath {
    // Return NO if you do not want the specified item to be editable.
    return YES;
}
```

Drive32/57

3-0-7.結果。



デフォルトで用意されている
UITableViewCellには、
ラベルとかイメージとかが
予め用意されています。

ま、普通ですよね。
でも、普通にCellを使うとか、
先ず無いよね。
独自の見た目にしたーい！

と思うのが人情というもの。

Drive33/57

3-1-0.TableViewを作つてみよう2 ハイパー
Appleのサンプルがマジゴイナーな件

本日の総決算。xibとImageとAnimationがドカッと出てくる。
UITableViewControllerを継承したクラスTableSample2ViewControllerを作り、

そのヘッダに、UITableViewCellのIBOutletを一つ記述する。



The screenshot shows the Xcode interface with the following details:

- Title Bar:** Shows "TableSample2ViewController.h" as the active file.
- Toolbar:** Includes icons for Simulator (4.2), Debug, TableSample2, Breakpoint, Build & Run, Task, Group Unwrap, and Project.
- Editor Area:** Displays the code for TableSample2ViewController.h. The code includes a header guard, copyright information, imports, and an interface definition that inherits from UITableViewController and includes an IBOutlet for a UITableViewCell.

```
TableSample2ViewController.h

// TableSample2ViewController.h
// TableSample2
//
// Created by Toru Inoue on 11/04/02.
// Copyright 2011 KISSAKI. All rights reserved.

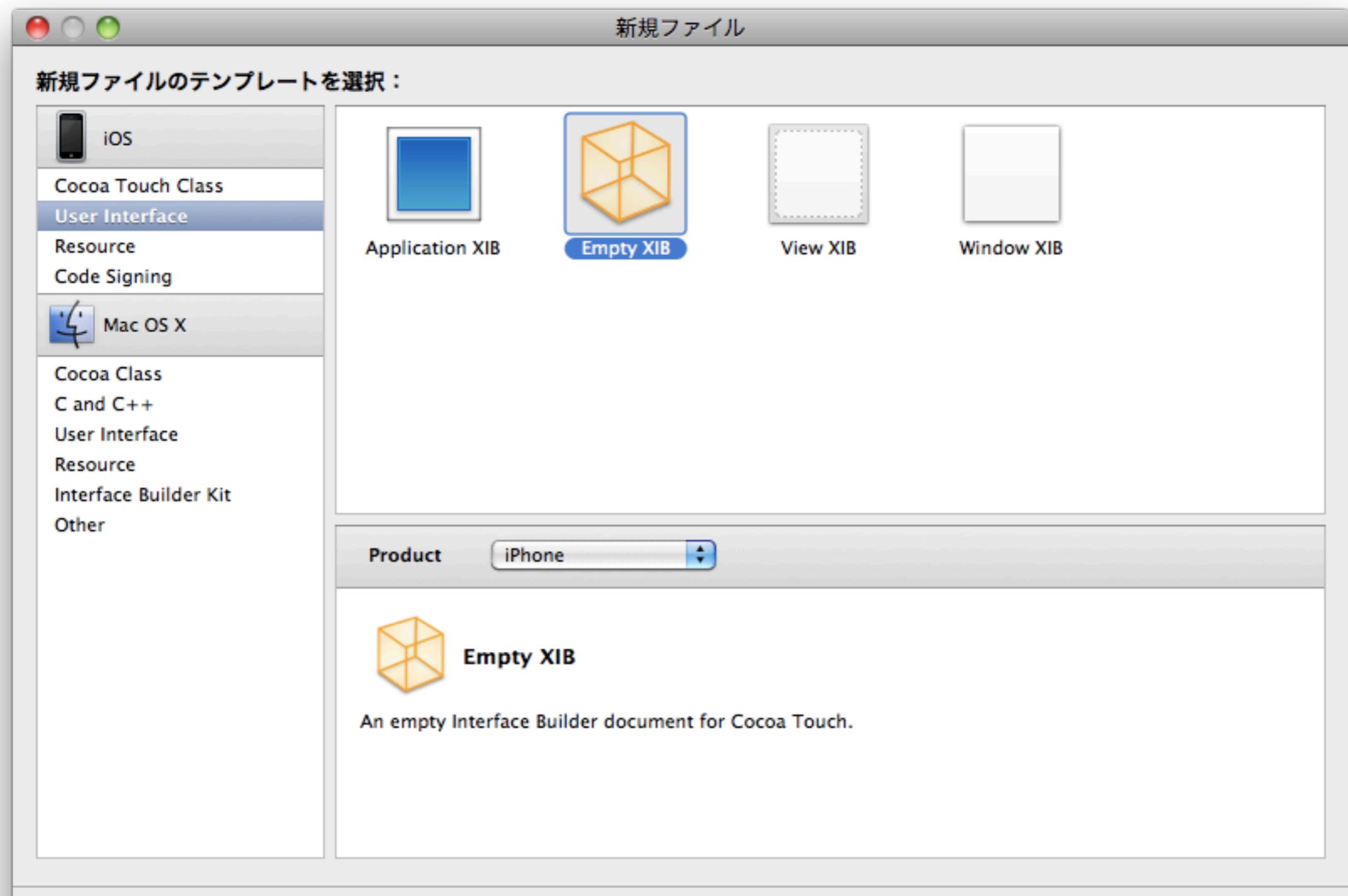
#import <UIKit/UIKit.h>

@interface TableSample2ViewController : UITableViewController {
    IBOutlet UITableViewCell * _cell;
}

@end
```

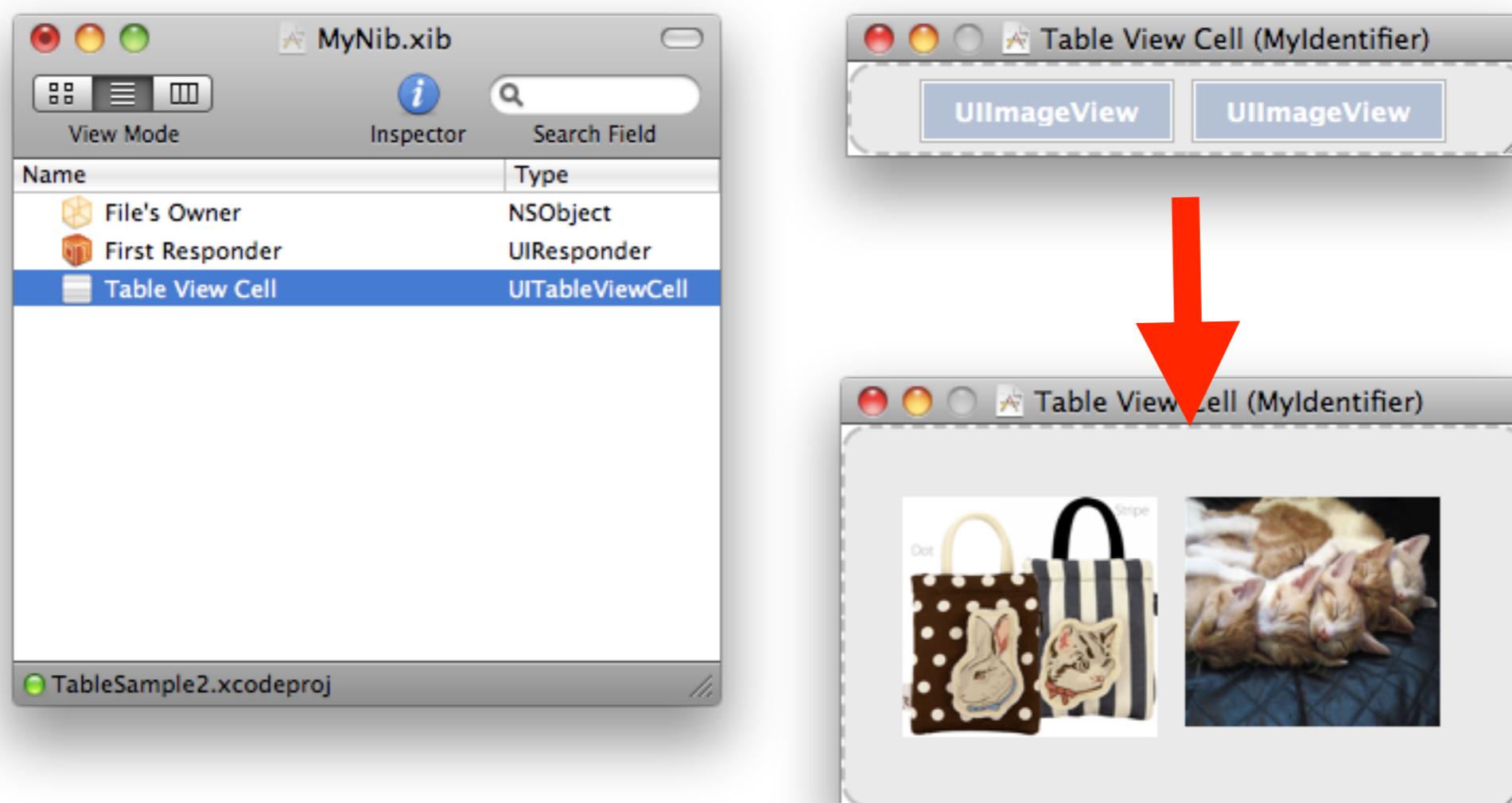
Drive34/57

3-1-1. ファイル > 新規ファイル > UserInterface → Empty XIB を作る。名前は、MyNib.xibとかにしておこう。



Drive35/57

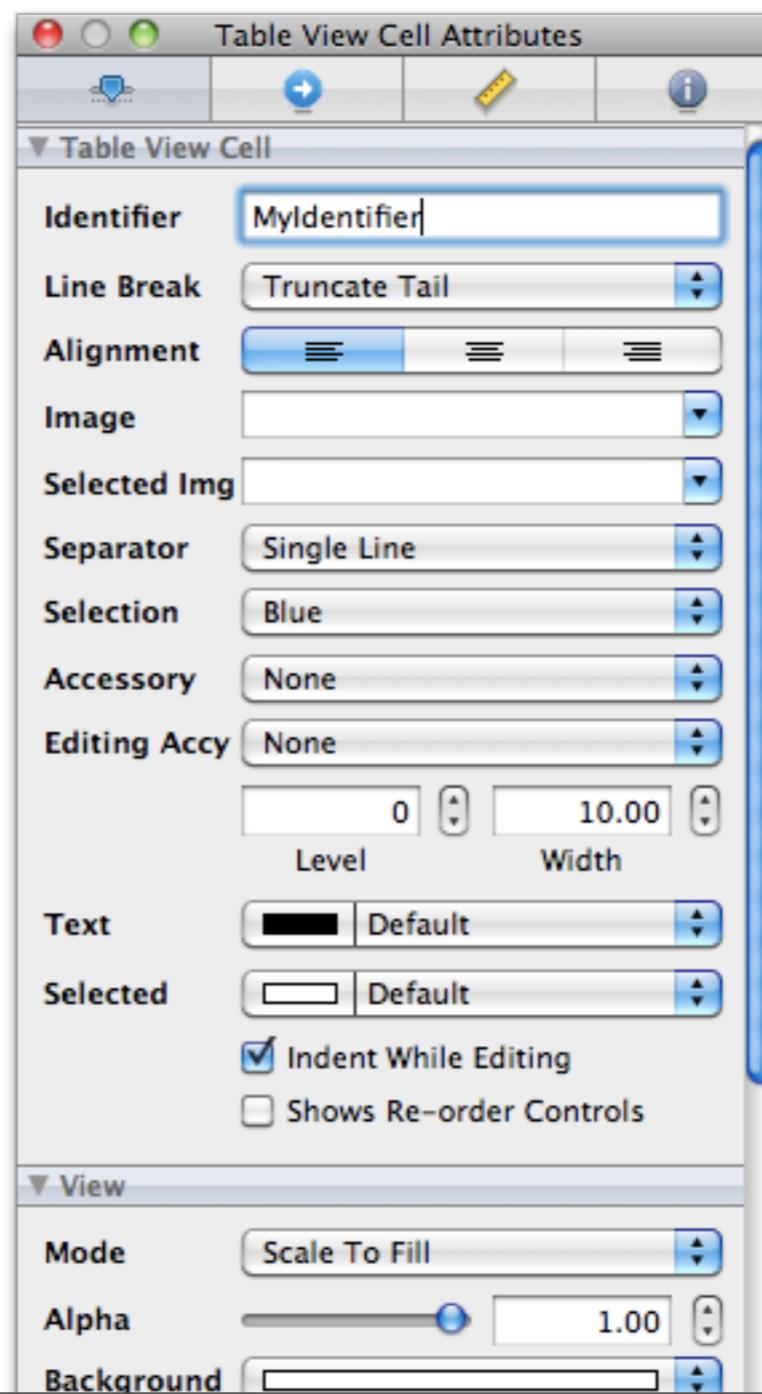
3-1-2. できたxibに、UITableViewCellを置く。サイズ弄くろうが、中に何を置こうが自由だ。
→ここでは、画像として、UIImageViewを二枚置いた。
画像を2枚、プロジェクトに放り込んで、
セルに置いたUIImageViewの中に表示しよう。



Drive36/57

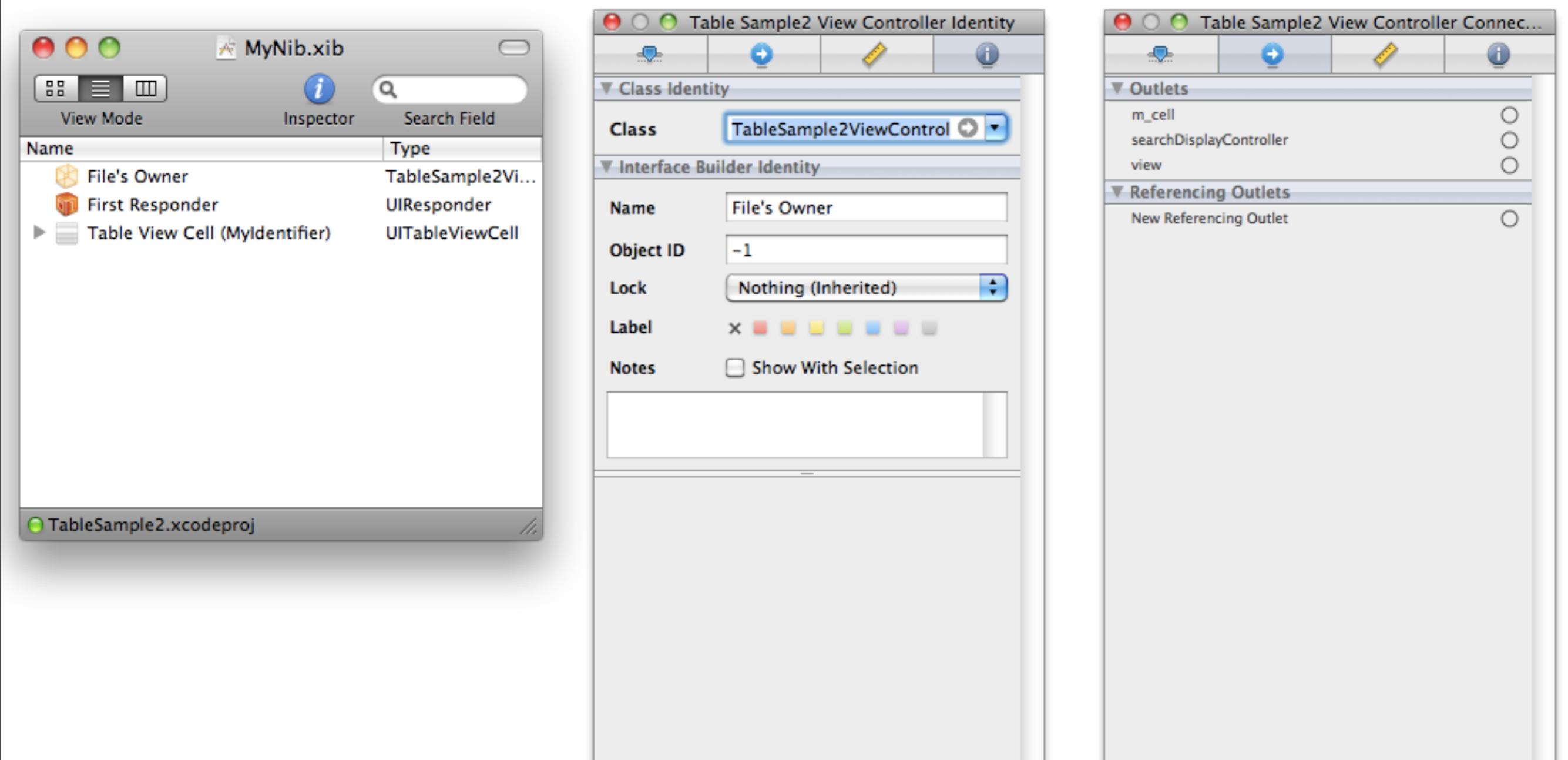
3-1-3.Identifierを入力

TableViewCellを選択した状態で、Inspector > Identifierに、 MyIdentifier と入力。



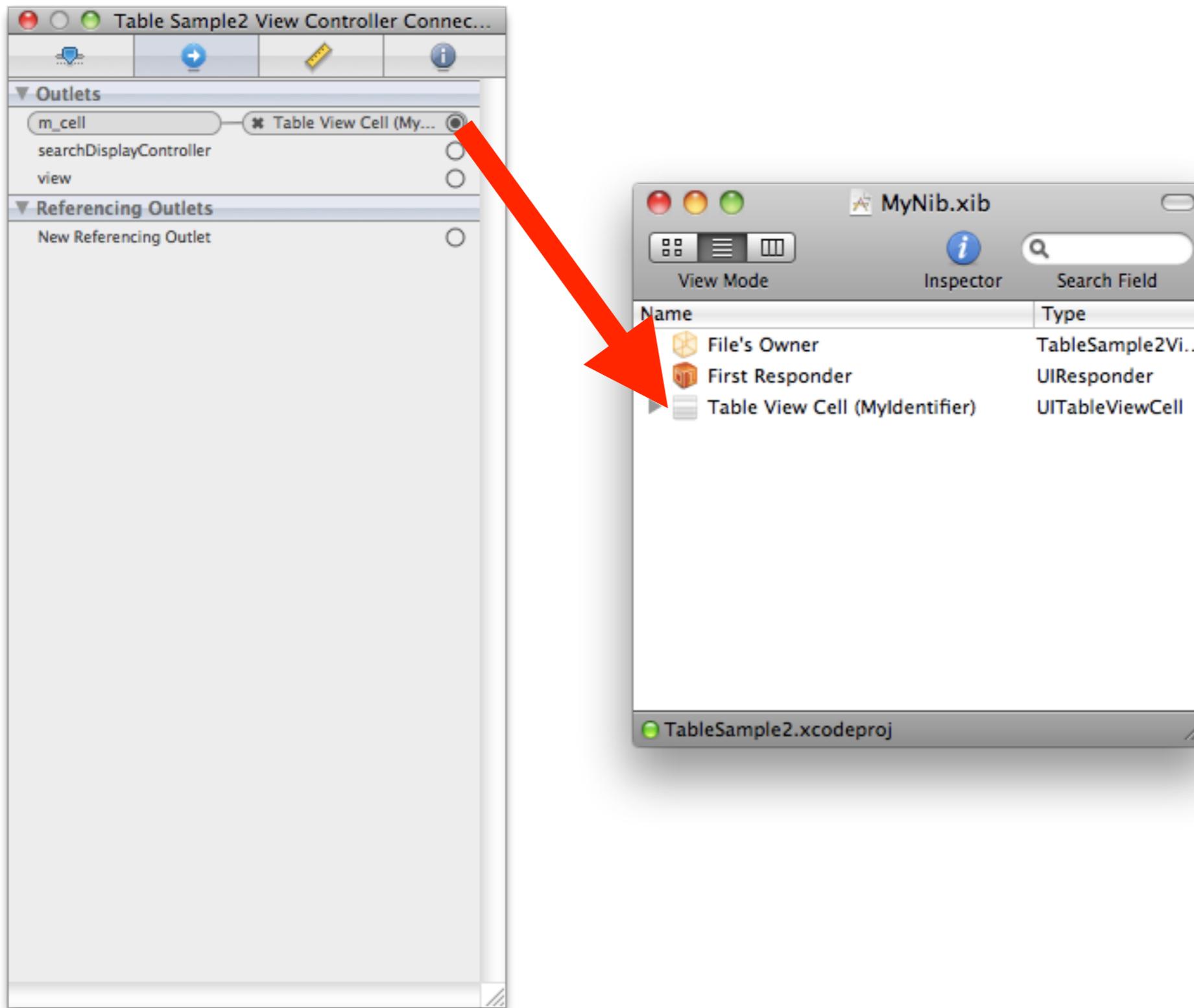
Drive37/57

3-1-4.MyNib.xibのFile's ownerに、TableSample2ViewControllerと入力。
すると、File's ownerのInspector > connectionに、
m_cellの名前があるじゃねーの。



Drive38/57

3-1-5. MyNib内のTableViewCellのReferenceと接続する。



Drive39/57

3-1-6. ここまででは、xibの使い道の一つなので、驚く事は無い、なんだけれど、

- (UITableViewCell *)tableView:(UITableView *)tableView
cellForRowAtIndexPath:(NSIndexPath *)indexPath

このメソッドの書き方で世界が一変する。

```
- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath {
    static NSString *MyIdentifier = @"MyIdentifier";

    UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:MyIdentifier];
    if (cell == nil) {
        [[NSBundle mainBundle] loadNibNamed:@"MyNib" owner:self options:nil];
        cell = m_cell;
        m_cell = nil;
    }
    return cell;
}
```

Drive40/57

3-1-7. ここまで書いたら、Delegateの中に、このTableSample2ViewControllerを表示するコードを書こう。

で、これで、、、何かが起こる筈。

ズキュウウウウン



え、なに、何が起こったの！？

MyNibにTableViewCellを置いて、高さを変えた人は、こうなる。

これは、TableViewのセルの高さみたいなものが、実は暗黙で固定されているから、縦に入りきらないぶんはハミ出す！

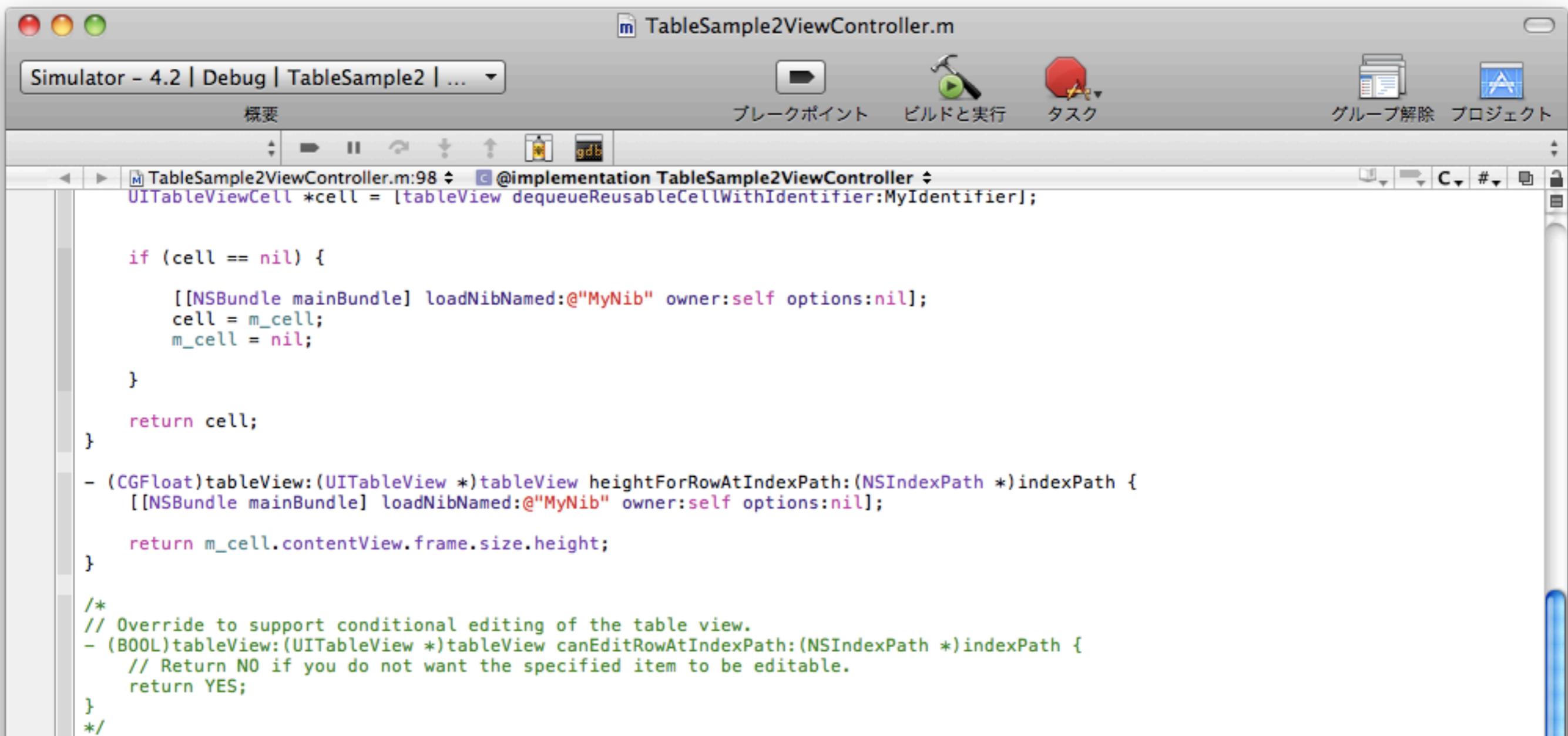
みたいな事になっているんだわ。

Drive4 | /57

3-1-8.もちろん、編集出来る

このメソッドを、TableViewのクラスの中に記述して、起動！

*このメソッドの名前の一文字でも間違えると、効果を発揮しないので、注意！



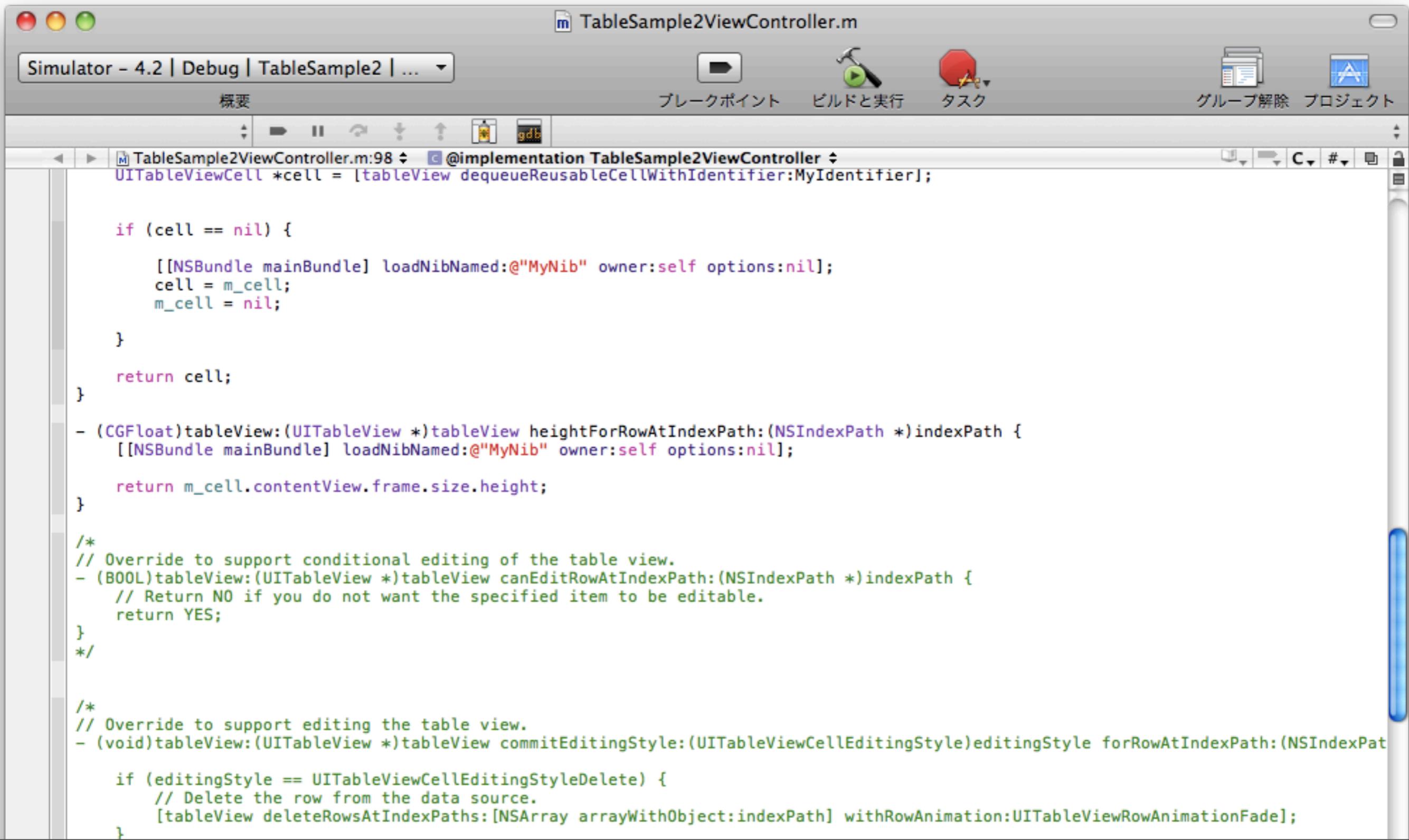
The screenshot shows the Xcode IDE interface with the following details:

- Title Bar:** Shows the file name "TableSample2ViewController.m".
- Toolbar:** Includes icons for Simulator (4.2), Debug, TableSample2, Breakpoint, Build & Run, Task, Group Unwrap, and Project.
- Editor Area:** Displays the source code for `TableSample2ViewController.m`. The code includes methods for dequeuing cells and returning their height, along with a conditional editing override.

```
m TableSample2ViewController.m
Simulator - 4.2 | Debug | TableSample2 | ...
概要 ブレークポイント ビルドと実行 タスク グループ解除 プロジェクト
TableSample2ViewController.m:98 @implementation TableSample2ViewController
UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:MyIdentifier];
if (cell == nil) {
    [[NSBundle mainBundle] loadNibNamed:@"MyNib" owner:self options:nil];
    cell = m_cell;
    m_cell = nil;
}
return cell;
- (CGFloat)tableView:(UITableView *)tableView heightForRowAtIndexPath:(NSIndexPath *)indexPath {
    [[NSBundle mainBundle] loadNibNamed:@"MyNib" owner:self options:nil];
    return m_cell.contentView.frame.size.height;
}
/*
// Override to support conditional editing of the table view.
- (BOOL)tableView:(UITableView *)tableView canEditRowAtIndexPath:(NSIndexPath *)indexPath {
    // Return NO if you do not want the specified item to be editable.
    return YES;
}
*/
```

Drive42/57

3-1-9. いちおうキャプチャ



The screenshot shows the Xcode IDE interface with the following details:

- Title Bar:** Shows the file name "TableSample2ViewController.m".
- Toolbar:** Includes icons for Simulator (4.2), Debug, TableSample2, Breakpoint, Build & Run, Task, Group Unlink, and Project.
- Code Editor:** Displays the source code for `TableSample2ViewController.m`. The code implements a table view controller, handling cell dequeuing, height calculation, and editing support.

```
m TableSample2ViewController.m
Simulator - 4.2 | Debug | TableSample2 | ...
概要 ブレークポイント ビルドと実行 タスク グループ解除 プロジェクト
TableSample2ViewController.m:98 @implementation TableSample2ViewController
UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:MyIdentifier];

if (cell == nil) {
    [[NSBundle mainBundle] loadNibNamed:@"MyNib" owner:self options:nil];
    cell = m_cell;
    m_cell = nil;
}

return cell;
}

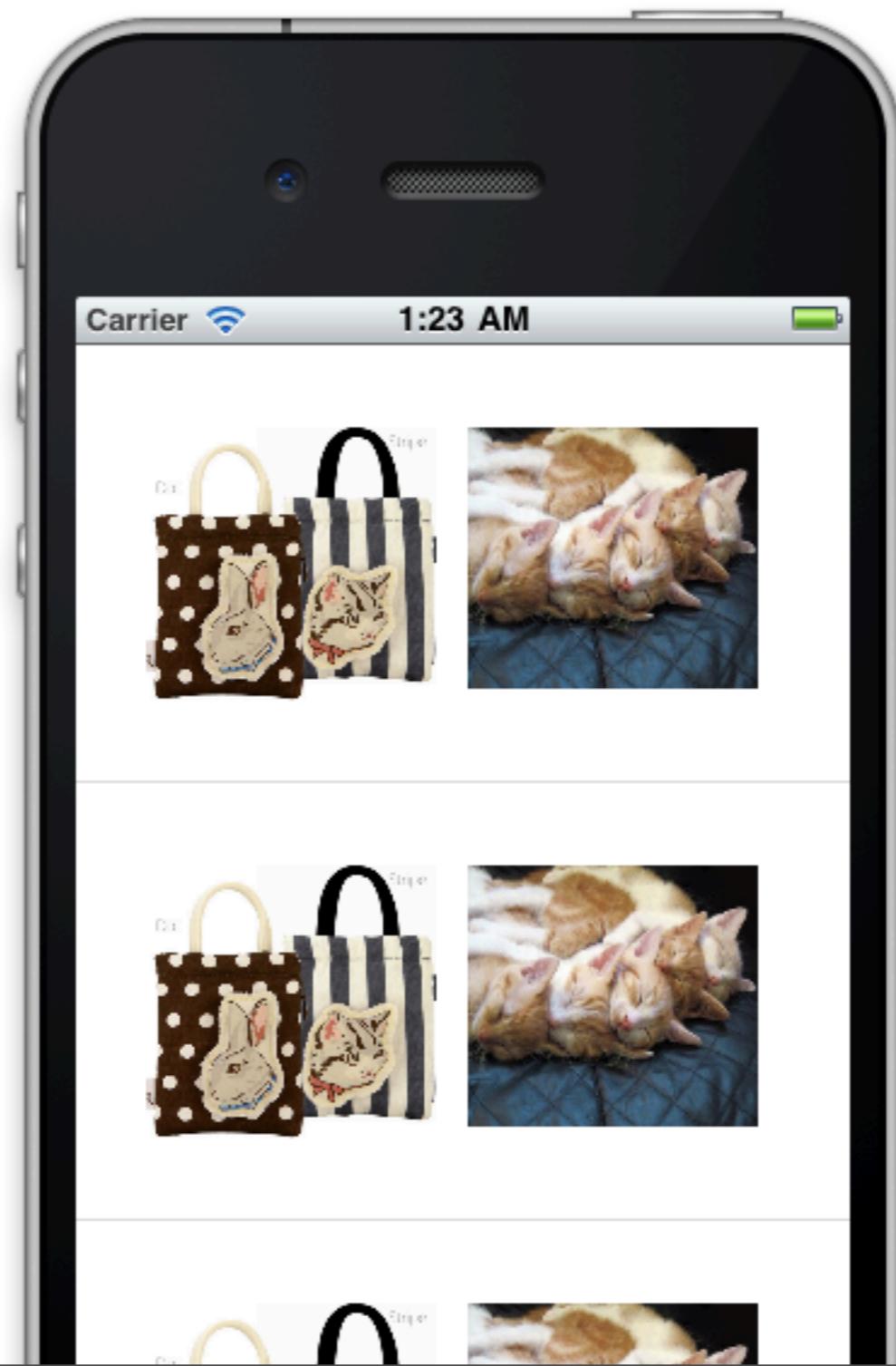
-(CGFloat)tableView:(UITableView *)tableView heightForRowAtIndexPath:(NSIndexPath *)indexPath {
    [[NSBundle mainBundle] loadNibNamed:@"MyNib" owner:self options:nil];
    return m_cell.contentView.frame.size.height;
}

/*
// Override to support conditional editing of the table view.
-(BOOL)tableView:(UITableView *)tableView canEditRowAtIndexPath:(NSIndexPath *)indexPath {
    // Return NO if you do not want the specified item to be editable.
    return YES;
}
*/

/*
// Override to support editing the table view.
-(void)tableView:(UITableView *)tableView commitEditingStyle:(UITableViewCellEditingStyle)editingStyle forRowAtIndexPath:(NSIndexPath *)
    if (editingStyle == UITableViewCellEditingStyleDelete) {
        // Delete the row from the data source.
        [tableView deleteRowsAtIndexPaths:[NSArray arrayWithObject:indexPath] withRowAnimation:UITableViewRowAnimationFade];
    }
}
```

Drive43/57

3-1-10.起動すると、、、うへっへーい完成。



Drive44/57

3-1-11.解説！ オーパーツが3つくらいある筈。
種明かしにいく。ソースコードを上から見よう。

```
UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:MyIdentifier];
```

- cellインスタンスをdequeuेから取り出す際、
`@"MyIdentifier"`をアイデンティファイに使っている。
これは、reusableなモノが有る場合の処理。

Drive45/57

3-1-10.xibの真理、超大事な事

```
if (cell == nil) {  
    [[NSBundle mainBundle] loadNibNamed:@"MyNib" owner:self options:nil];
```

- cellがnilであれば、メインバンドルから**MyNib.xib**を、
ファイル名@**"MyNib"**をトリガーにして呼び出す。
MyNib.xibがバンドルから呼ばれた瞬間、xibが展開される。ここまででは、普通の事。
なんだけど、この時、xibには、
File's Owner = TableSample2ViewController、
TableViewCell = TableSample2ViewControllerの **m_cell** という、
関係性が記述されている。

xibが読み込まれた瞬間、xibは、
TableSample2ViewControllerとの関係性を元に、瞬間で内容を構築！
m_cell にはこの瞬間にすでに、MyNibファイルの内容と同じように、画像が入っている！
なんてアクロバティック。

Drive46/57

3-1-11.xibは、実は、全てこのような機構で動いている。

- ・ 展開された瞬間、クラス名からFile's Ownerとおぼしきインスタンスを発見し、そのクラスからのつながりに属するインスタンス(IBOutlet)の位置や形状などをxibの通りに整形する。

これが、xibの最後の秘密。

Drive47/57

3-1-12.MyIdentifier は何をしているのか

- MyNib.xibの中で、TableViewCellに対して、MyIdentifierという文字列を入れた。
- - `(UITableViewCell *)tableView:(UITableView *)tableView`
`cellForRowAtIndexPath:(NSIndexPath *)indexPath`
メソッドの中にも、MyIdentifierの記述が有る。

これは、`[tableView dequeueReusableCellWithIdentifier:MyIdentifier]`と書いてある部分で、

`MyIdentifier` という記述が既に着いているTableViewCellがあれば、それらをreuseするために、非表示になった部分に召還(瞬間移動)する。

もしあなたが `cell == nil` となるので、直下のif文の中を通り、
`MyIdentifier` という名前のついたTableViewCellを
xibから取得した`m_cell`が生まれ、
`cell = m_cell` とポインタ同期されるので、`cell = MyIdentifier` から作られた、
`MyIdentifier` という名前のついたcellになる、という訳だ。

Drive48/57

3-1-13.で、もし、

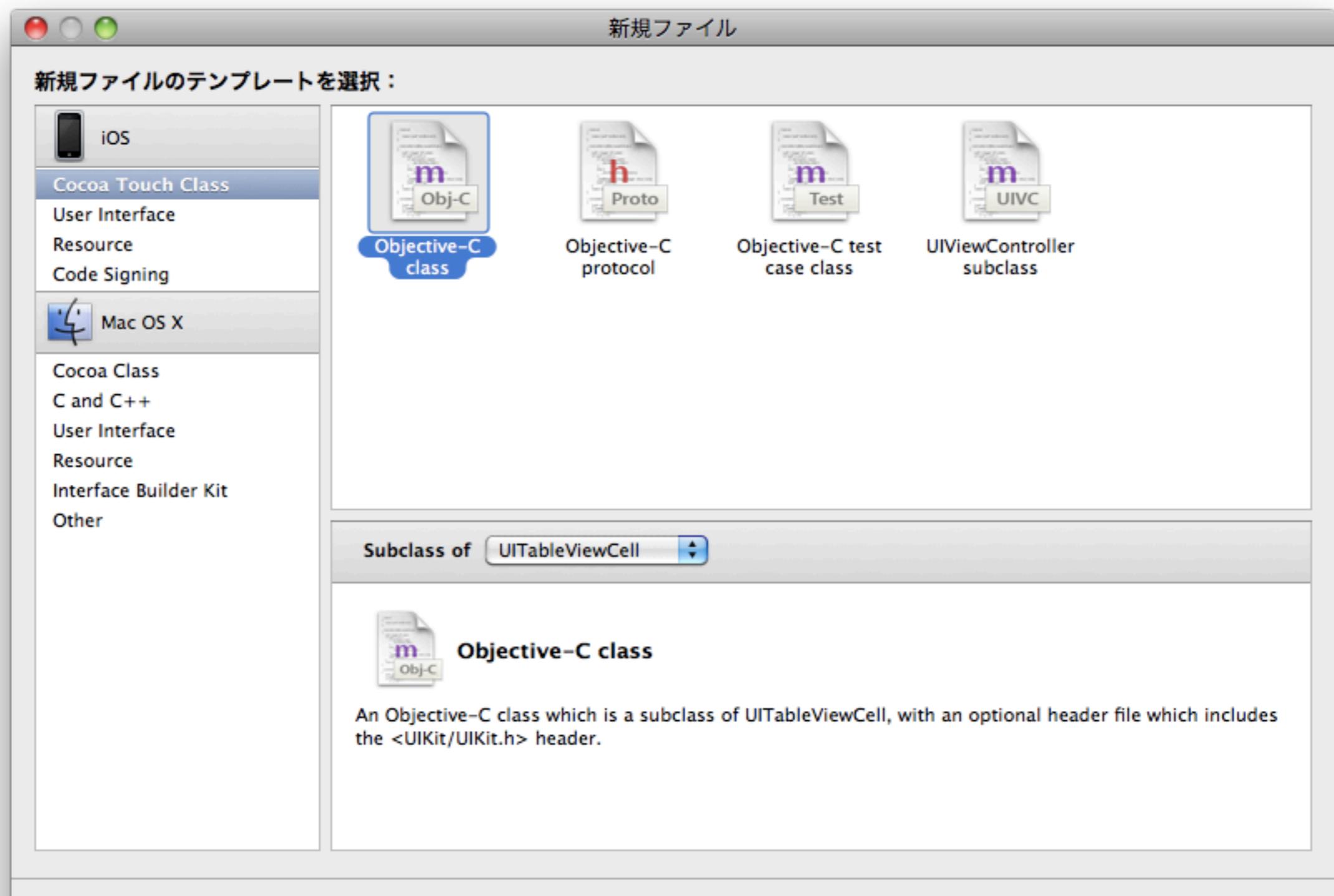
MyIdentifier という記述が既に着いているTableViewCellがあれば、
それらをreuseするために、非表示になった部分から召還(瞬間移動)する。
(赤い字の部分は繰り返しになっていることに注目！)

、、、 というように、MyNib.xibで付けた名前が、2週目以降で生きてくる。

Drive49/57

3-2-0. ラスト、こいつにアニメーションを入れてみよう

UITableViewCellをinheritしたクラス MyTableViewCell を作成。



Drive50/57

3-2-1. MyTableViewCell クラスの.hに、IBOutletでUIImageViewを2つ書こう。

左と右。

あと、アニメーションのコードも書いてしまおう。



The screenshot shows the Xcode IDE interface with the following details:

- Title Bar:** Displays "MyTableViewCell.h" as the current file.
- Toolbar:** Contains icons for "Simulator - 4.2 | Debug | TableSample2 | ...", "Breakpoint", "Build & Run", "Task", "Group Unwrap", and "Project".
- Editor Area:** Shows the code for MyTableViewCell.h. The code includes imports, an interface definition with two IBOutlet UIImageView properties named m_left and m_right, a private method animation, and an end brace.

```
#import <UIKit/UIKit.h>

@interface MyTableViewCell : UITableViewCell {
    IBOutlet UIImageView *m_left;
    IBOutlet UIImageView *m_right;
}
- (void) animation;
@end
```

Drive5 | 57

3-2-2. .mにanimationの実装を書く。

The screenshot shows the Xcode interface with the file `MyTableViewCell.m` open. The code implements an animation for selecting a table view cell. It includes loading images from the main bundle, starting an animation with a duration of 0.5f, and performing a flip transition from left to right.

```
- (void)setSelected:(BOOL)selected animated:(BOOL)animated {
    [super setSelected:selected animated:animated];
    // Configure the view for the selected state.
}

- (void)animation {
    NSBundle * currentBundle = [[NSBundle mainBundle] autorelease];
    UIImage * toLeftImage = [[UIImage alloc] initWithContentsOfFile:[currentBundle pathForResource:@"a8b7e97d" ofType:@"jpg"]];
    UIImage * toRightImage = [[UIImage alloc] initWithContentsOfFile:[currentBundle pathForResource:@"25270571" ofType:@"jpg"]];

    [UIView beginAnimations:@"高速めくりあげ" context:nil];
    [UIView setAnimationDuration:0.5f];
    [m_left setImage:toLeftImage];
    [m_right setImage:toRightImage];
    [UIView setAnimationTransition:UIViewAnimationTransitionFlipFromLeft forView:self cache:FALSE];
    [UIView commitAnimations];
}

- (void)dealloc {
    [super dealloc];
}

@end
```

TableSample2を起動

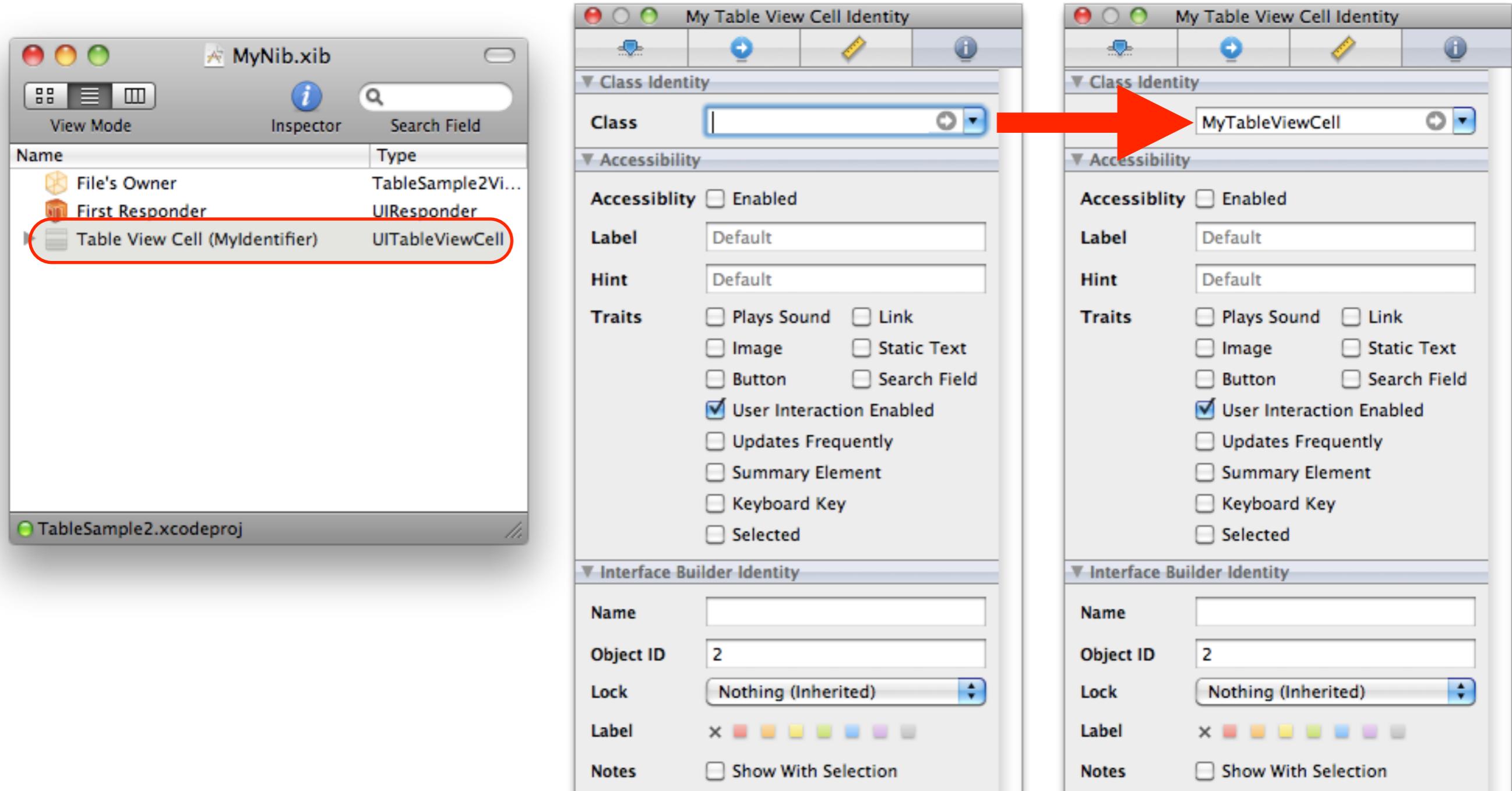
問題なく完了しました

Drive52/57

3-2-3.MyNib.xibを開いて、TableViewCellのClassを弄る

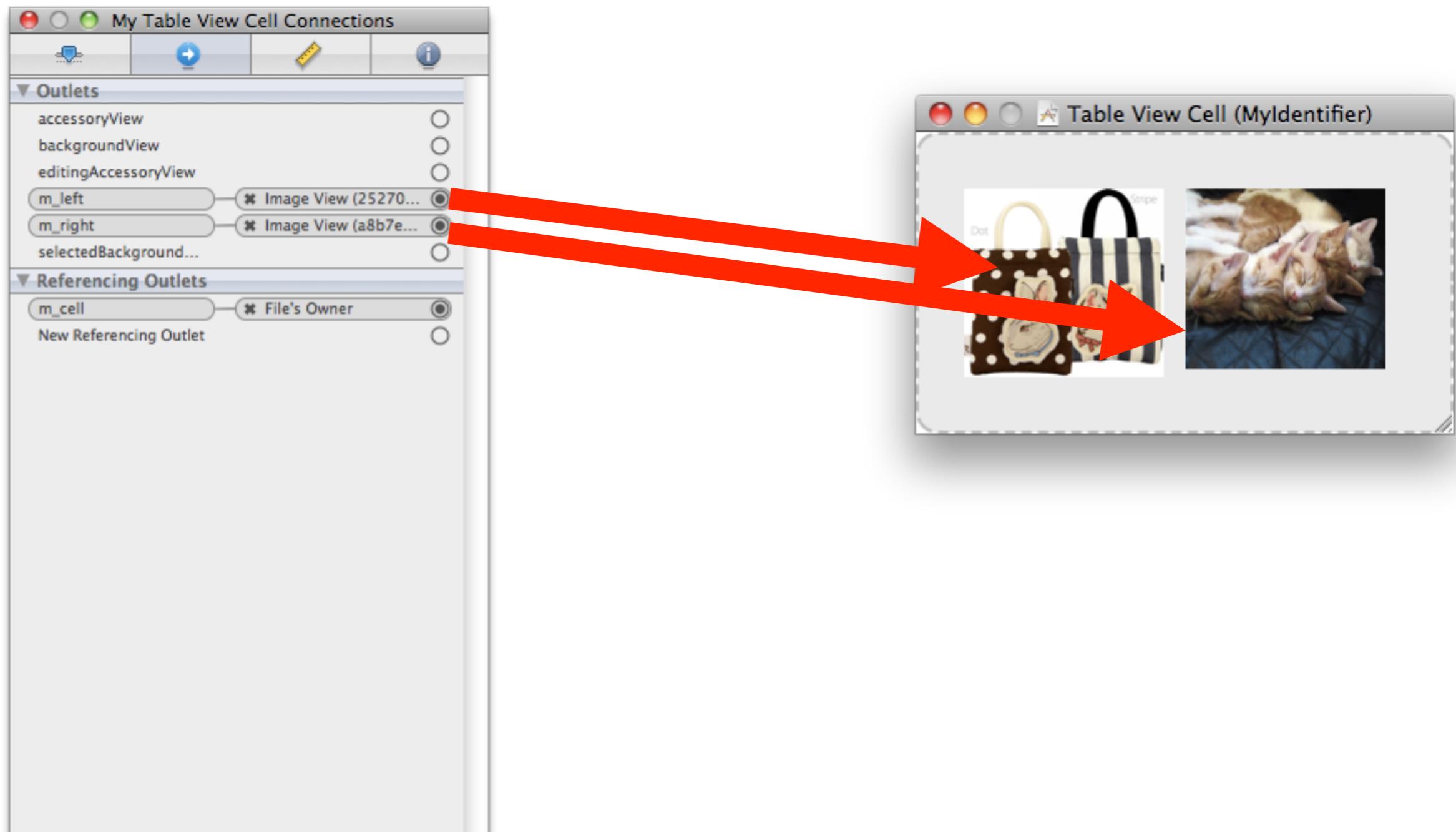
Drive37でやったみたいに、TableViewCellのClassに、MyTableViewCellって書く！

→これで、TableViewCellは MyTableViewCell のインスタンスになった。



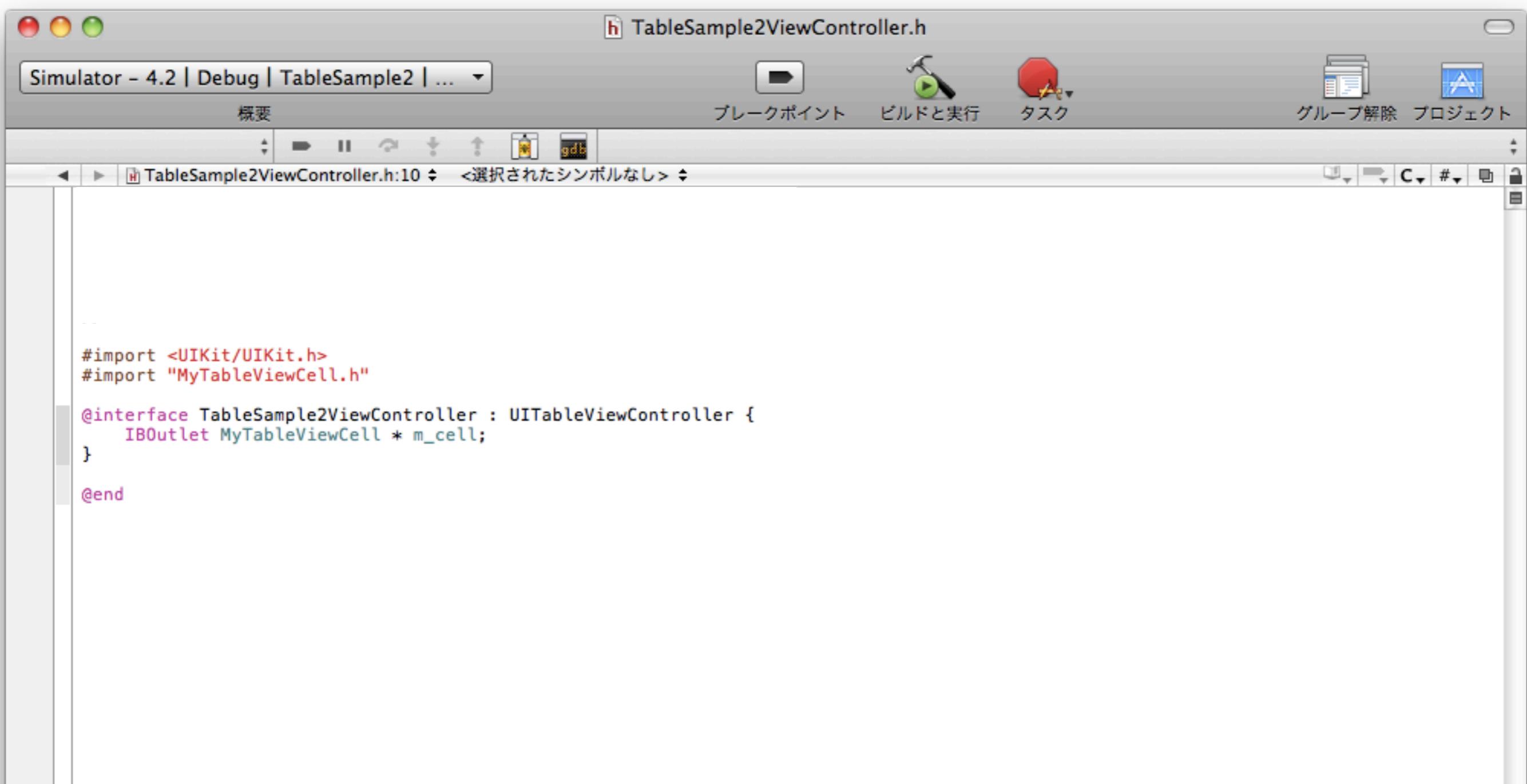
Drive53/57

3-2-4.MyNib内の画像2枚を、MyTableViewCell 内のm_right,m_leftと関連づけ設定。
Cell内の画像を、Cellに含まれているm_right,m_leftと繋ぐ。



Drive54/57

3-2-5.TableSample2ViewController 内の、 m_cellのクラスを変える
m_cellに関する記述を、次のように変える。
.hで。



The screenshot shows the Xcode IDE interface with the title bar "TableSample2ViewController.h". The menu bar includes "Simulator - 4.2 | Debug | TableSample2 | ...". The toolbar contains icons for Run, Breakpoint, Build & Run, Task, Group Unfold, and Project. The main editor area displays the following code:

```
#import <UIKit/UIKit.h>
#import "MyTableViewCell.h"

@interface TableSample2ViewController : UITableViewController {
    IBOutlet MyTableViewCell * m_cell;
}

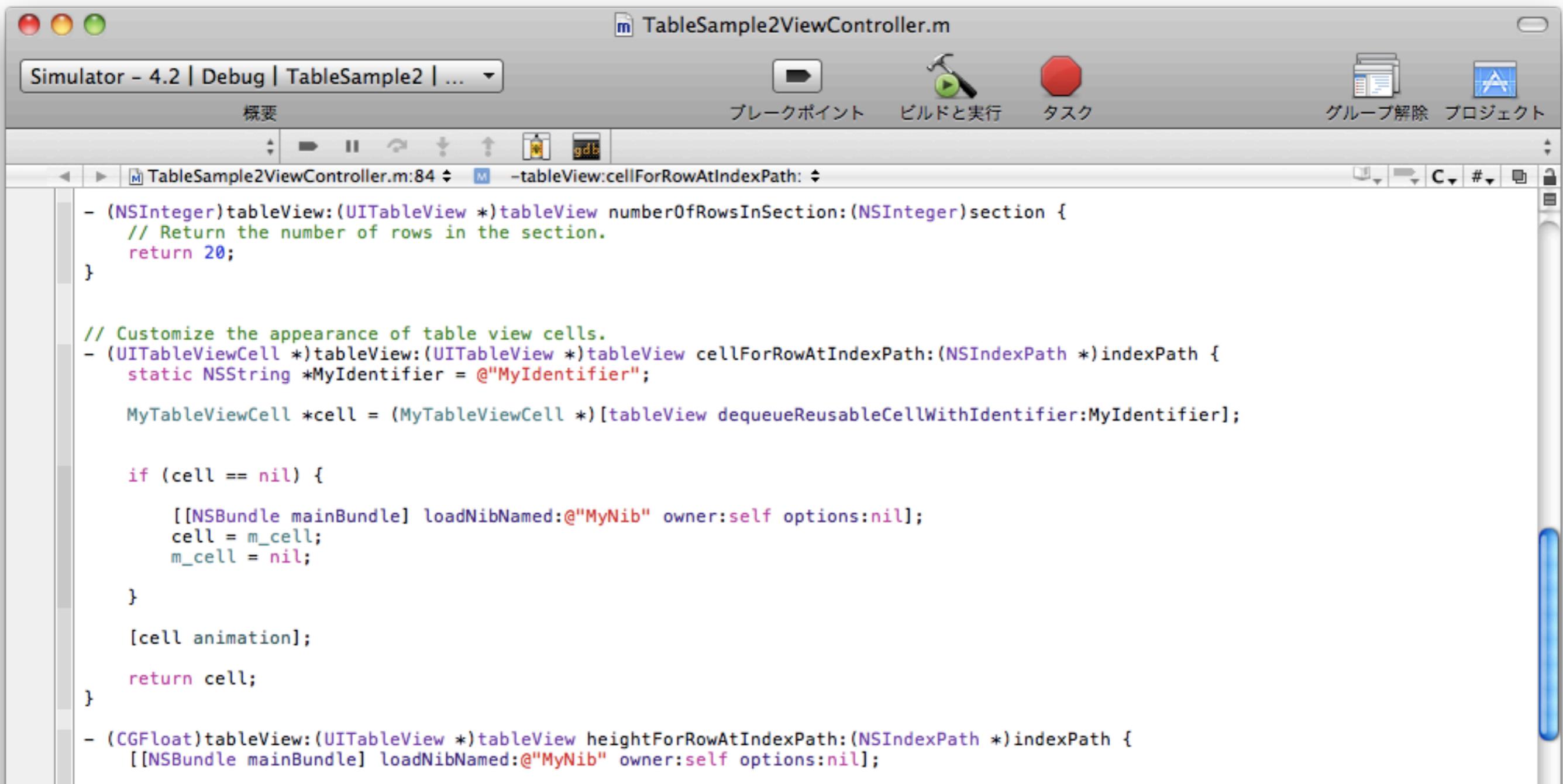
@end
```

Drive55/57

3-2-6.m_cellについて、.m で。

こちらでは、TableViewCellだった部分を、MyTableViewCell に変える。

あと、MyTableViewCell のアニメーションも、ここで呼ぼう。



The screenshot shows the Xcode interface with the file `TableSample2ViewController.m` open. The code implements the `UITableViewDataSource` and `UITableViewDelegate` protocols. It defines a section with 20 rows and customizes the appearance of each row by dequeuing a reusable cell from a nib named "MyNib". The cell is then animated before being returned.

```
- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section {
    // Return the number of rows in the section.
    return 20;
}

// Customize the appearance of table view cells.
- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath *)indexPath {
    static NSString *MyIdentifier = @"MyIdentifier";

    MyTableViewCell *cell = (MyTableViewCell *)[tableView dequeueReusableCellWithIdentifier:MyIdentifier];

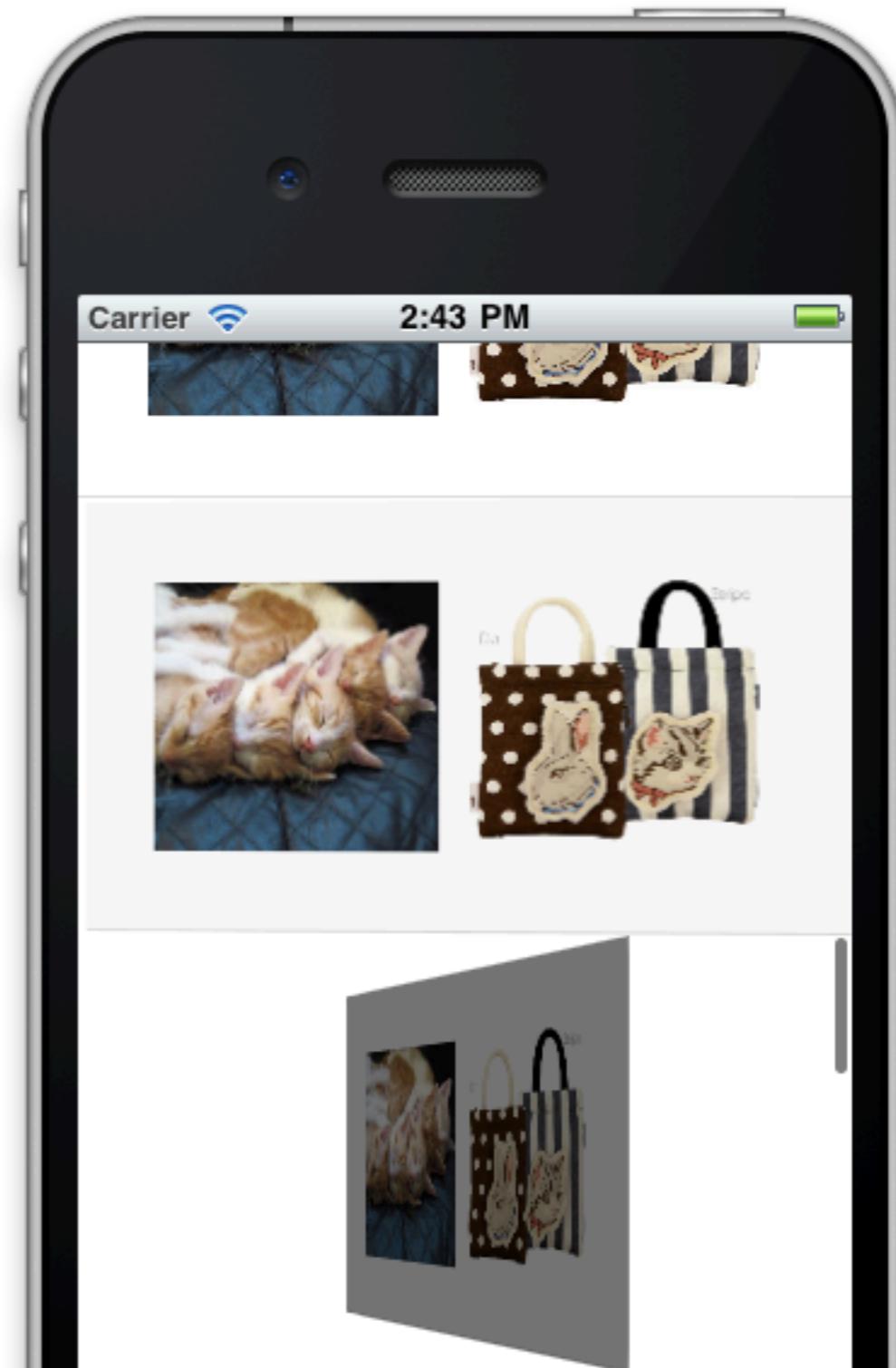
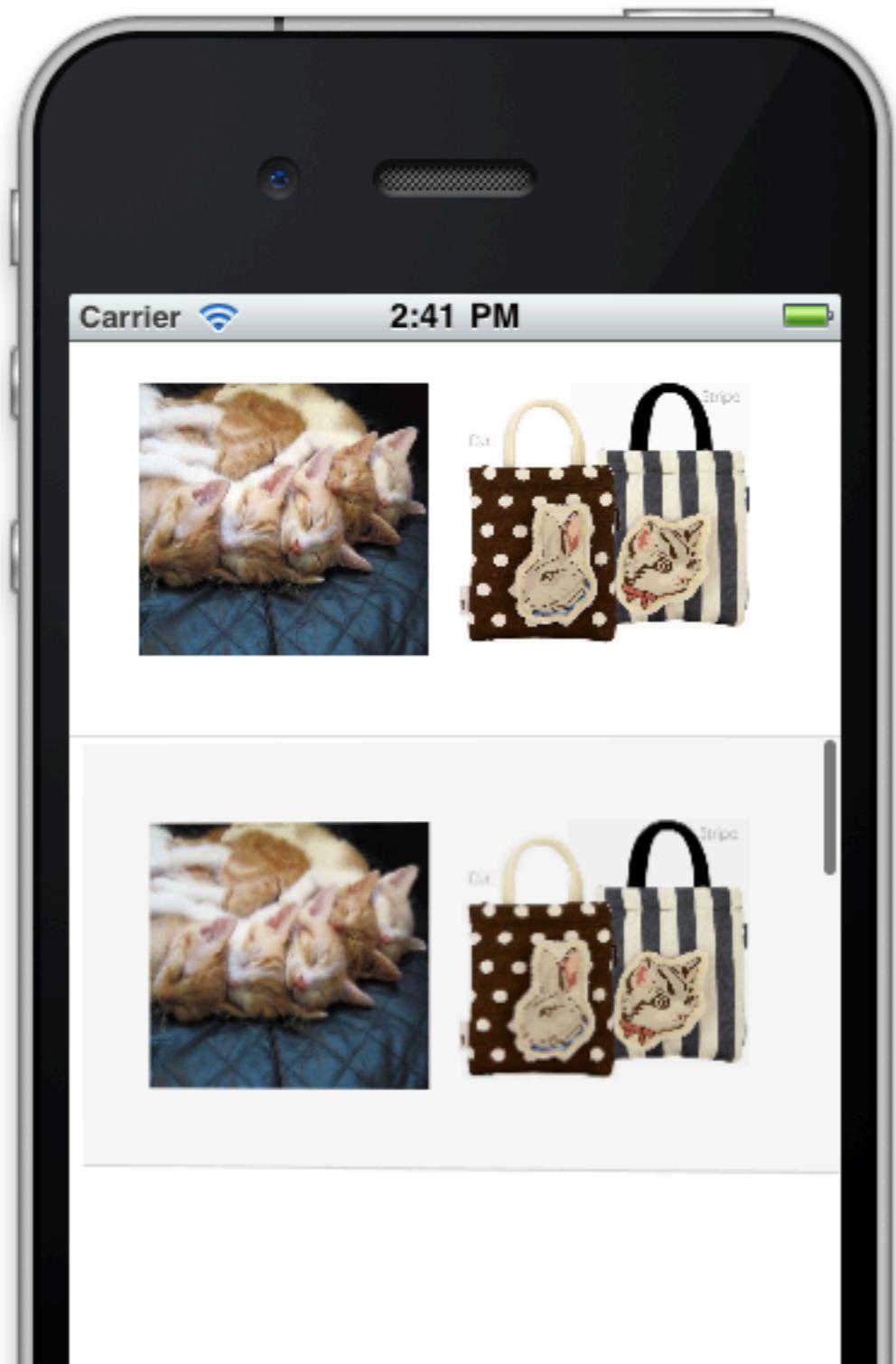
    if (cell == nil) {
        [[NSBundle mainBundle] loadNibNamed:@"MyNib" owner:self options:nil];
        cell = m_cell;
        m_cell = nil;
    }

    [cell animation];
    return cell;
}

- (CGFloat)tableView:(UITableView *)tableView heightForRowAtIndexPath:(NSIndexPath *)indexPath {
    [[NSBundle mainBundle] loadNibNamed:@"MyNib" owner:self options:nil];
}
```

Drive56/57

3-2-7.動かす。ぬこが神のようだ！！！



Drive57/57

3-2-8.xibゴイスト。 アニメーションゴイスト。 ということで、今回は以上！

LookBack

- 1.iOSでのグラフィックリソースの持ち方初歩、
- 2.アニメーション無双
- 3.TableView無双

プログラムの質問、提出はこっちな！！
自分の名前と宛名忘れんなよ！！



toru.inoue@kissaki.tv

Continue?

次回、

第2回

「CoreDataなにそれ死ぬわ」