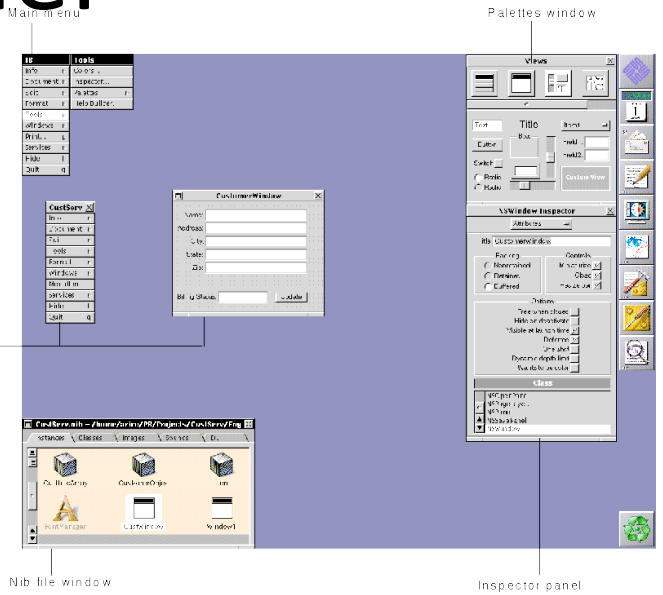
iPhone 開発における Interface Builderについて

慶應義塾大学
徳田研究室
伊藤昌毅
niya@ht.sfc.keio.ac.jp

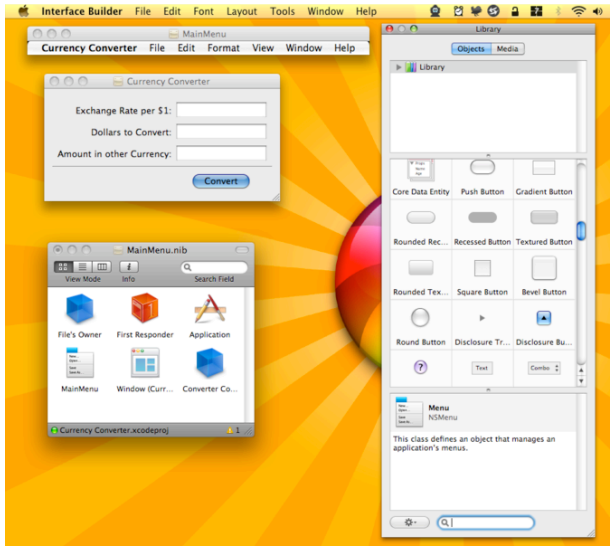
Interface Builder

- NextStep 由来のGUI開発環境
  - 1988年登場
  - 最初期のGUIによるGUI開発環境



- 使うか使わないか
  - Appleの文書では，利用を推奨
  - より素早く開発・変更できる
  - パフォーマンス的には？

  - あるのに使わないのは勿体ない！



アプリケーション開発手順

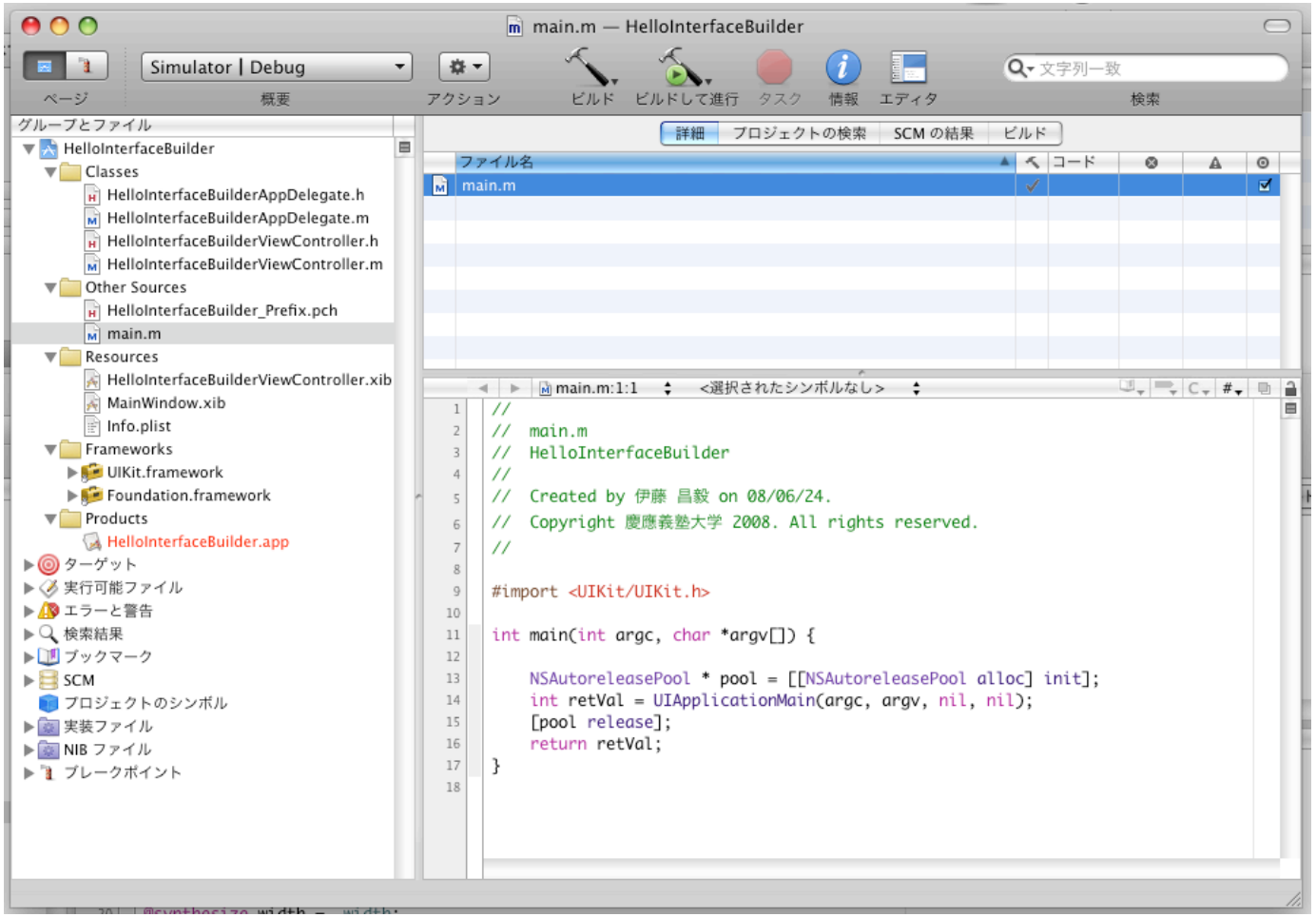
- iPhone Application Tutorial に準拠
- 以下のデザインパターンを前提
  - Delegation
  - Model View Controller
  - Target Action

プロジェクト作成

- ファイル→新規プロジェクト
- View Based Applicationを選択

作成直後のプロジェクト

- 全く白紙のアプリケーションがつくられる

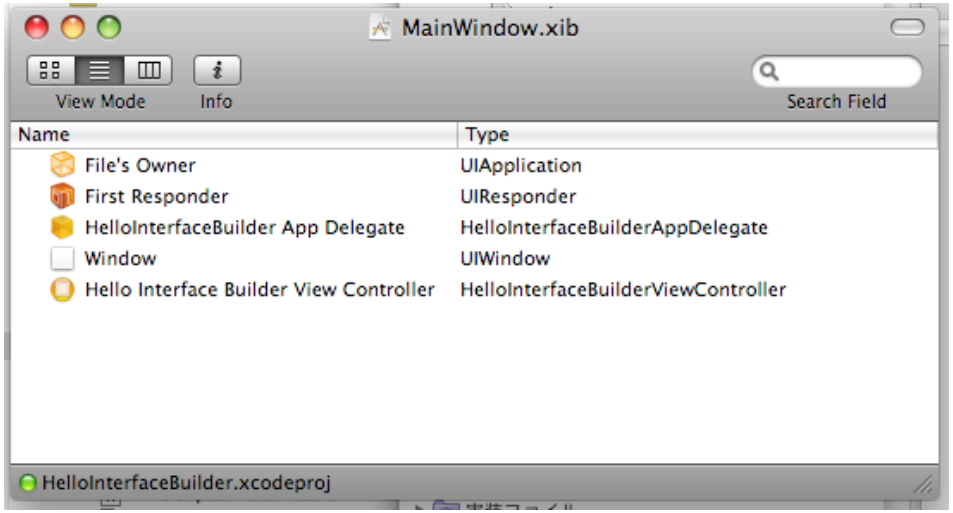


アプリケーションの起動手順 1

- main.m において
  - UIApplicationクラスをインスタンス化
  - Info.plist を参照
    - Main nib file base nameの名前からnib(xib)ファイルを探す
  - Nibファイル読み込み
    - MainWindow.xib の記述に基づきインスタンス化

Proxy Object: 既にある
オブジェクトが割り当てられる

インスタンス化するのは3つの
オブジェクト

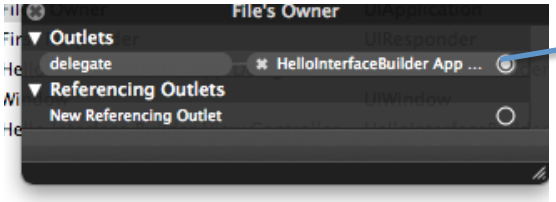


Nibファイル読み込み（前ページより）

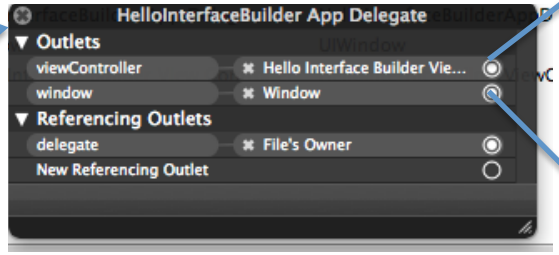
- Outletの設定
  - Outlet = クラスのインスタンス変数
  - ヘッダにおける変数宣言の前にIBOutlet と書く
- Nib中のオブジェクトには以下のOutletがある
  - HelloInterfaceBuilderViewController.h において
    - IBOutlet UIWindow *window;IBOutlet
    - HelloInterfaceBuilderViewController *viewController;
  - UIApplicationクラスにおいて
    - Id delegate（idは，すべての種類のオブジェクトを代入できる型)

GUIによるOutletの設定

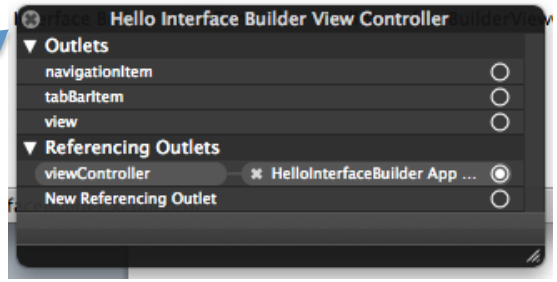
File's Owner (UIApplication)



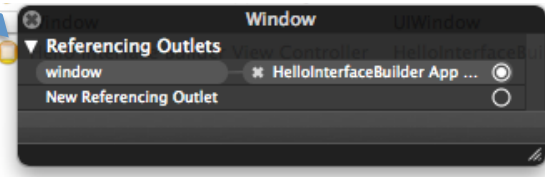
HelloInterfaceBuilderAppDelegate



HelloInterfaceBuilderViewController



UIWindow



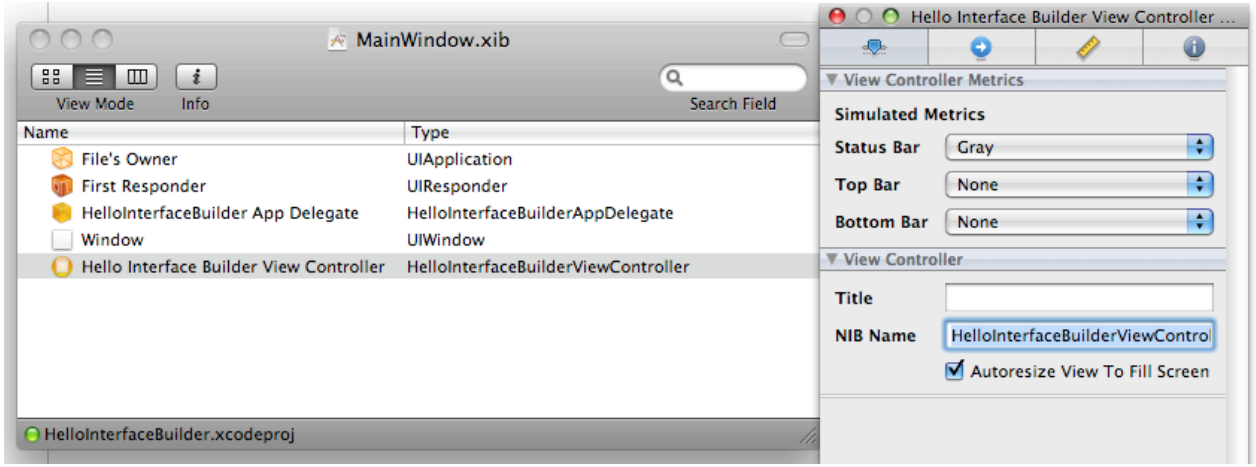
- オブジェクト上を右クリックすると上のウィンドウが出現
- Outlets
  - そのオブジェクトが持っているInterfaceBulder で設定可能なインスタンス変数
  - ○をクリックすると矢印が出てきて，変数に代入するオブジェクトを設定可能
  - ×をクリックすると代入が取り消される
  - 変数の型によって代入の可否が決まる．代入できないときは矢印を持って行っても繋げられない
- Referencing Outlets
  - そのオブジェクトが他のオブジェクトに代入されているときに表示される

Nib読み込みについて，以下は省略

- Nibが読み込まれたときのメモリリテインの話
  - Resource Programming Guide pp.20-21 参照
- Actionの接続の話
  - 後述

アプリケーションの起動手順：nib読み込み

- Hello Interface Builder View Controller のNIB Name欄に注目
  - ここにあるnibの名前とともにViewControllerが初期化される
    - initWithNibName:bundle: を呼び出す
  - 最初のnibファイルのFile's ownerはUIApplicationなので，自分のUIは別途自作ViewControllerをFile's Owner とするnibファイルを作成した方がよい
  - HelloInterfaceBuilderAppDelegate.m
    内，applicationDidFinishLaunchingで，windowのaddSubViewメソッドを呼び出し，ここで作成した画面を表示させている

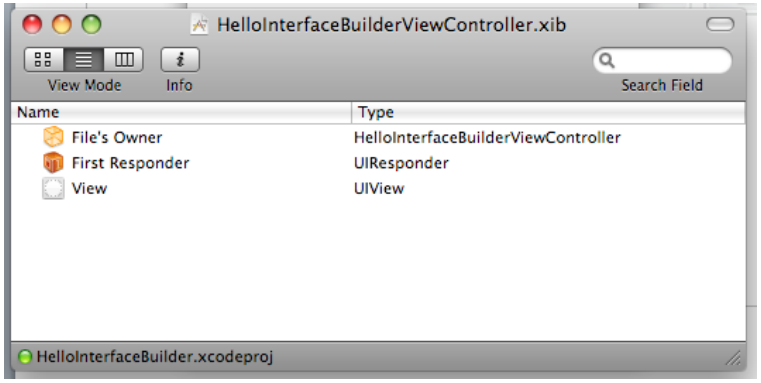


HelloInterfaceBuilderViewController.xib について

- Nibファイル読み込み
  - File's Ownwerに，自作のViewControllerが割り当てられる
  - インスタンス化されるのはUIVew
  - ViewController のView変数(Outlet)に割り当て
    - 自作ViewControllerの親クラスで定義されている

Proxy Object: 既にある
オブジェクトが割り当てられる

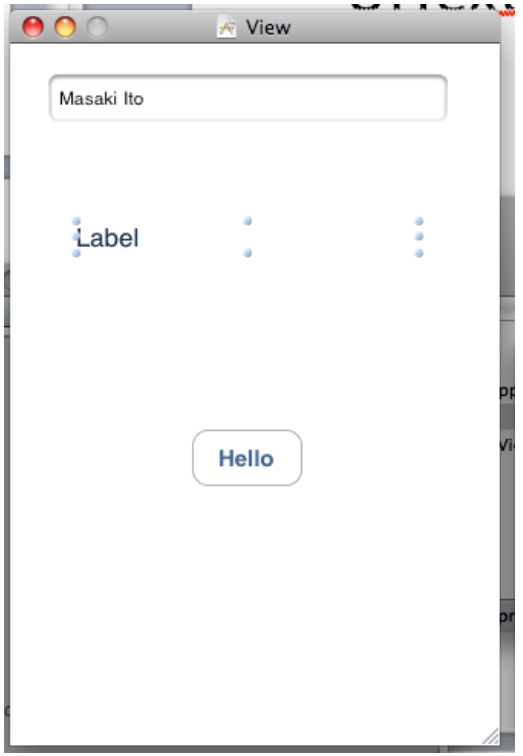
インスタンス化するのは1つの
オブジェクト



- 以上がiPhone Application Tutorial 31ページまでに相当

Interface BuilderにてViewの編集

- HelloInterfaceBuilderViewController.xibを編集
  - UITextField, UILabel, UIButtonを追加
  - 位置，文字を編集し図のように
  - Inspectorでキーボードの属性など適宜設定
    - Return キーは “done”に
- この時点で保存，起動することでテスト可能
  - キーボードを表示したら戻れないのは仕様



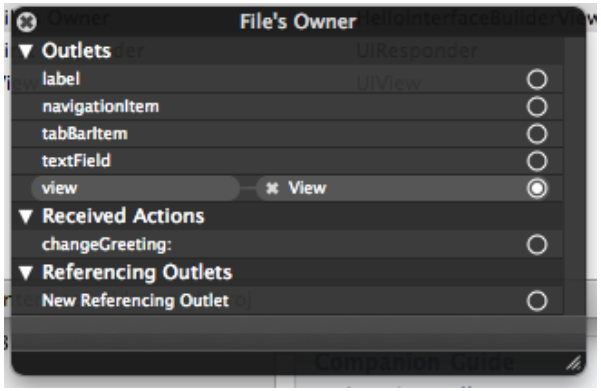
View Controllerを設定

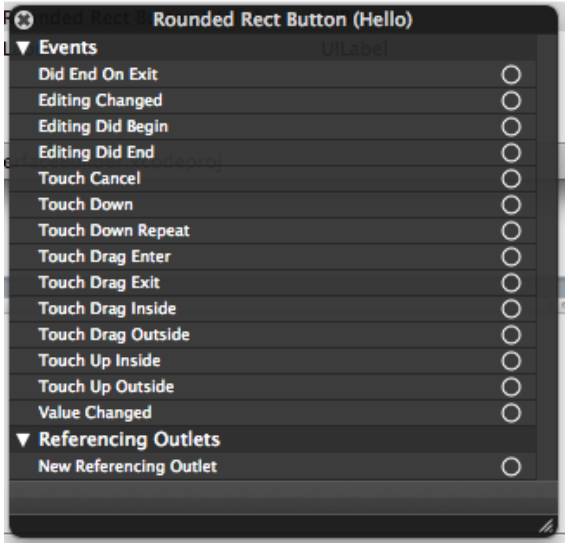
- ヘッダにOutletとなる変数を設定
  - IBOutlet UITextField *textField;
  - IBOutlet UILabel * label;
- Actionを設定
  - Action: イベント発生時に呼び出されるメソッド
  - 通常のメソッド宣言の戻り型として(IBAction)を指定
  - -(IBAction)changeGreeting: (id)sender;

```
9  #import <UIKit/UIKit.h>
10
11 @interface HelloInterfaceBuilderViewController : UIViewController {
12     IBOutlet UITextField *textField;
13     IBOutlet UILabel *label;
14     NSString *string;
15 }
16
17 @property (nonatomic, retain) UITextField* textField;
18 @property (nonatomic, retain) UILabel* label;
19 @property (nonatomic, retain) NSString* string;
20
21 -(IBAction)changeGreeting: (id) sender;
22
23 @end
```

Interface Builder 設定

- File’s Owner の属性が自動的に増えている
  - label, textFieldを設定
- Actionの設定
  - ボタンには様々なイベントが定義されている
  - Touch Up Insideと結びつける





View Controller を実装

```
#import "HelloInterfaceBuilderViewController.h"

@implementation HelloInterfaceBuilderViewController

@synthesize textField;
@synthesize label;
@synthesize string;

-(IBAction)changeGreeting: (id) sender{
    self.string = textField.text;

    NSString *nameString = string;
    if([nameString length] == 0){
        nameString = @"World";
    }

    NSString *greeting = [[NSString alloc]initWithFormat:@"Hello ,%@!", nameString];
    label.text = greeting;
    [greeting release];
}

- (void)dealloc {
    [textField release];
    [label release];
    [string release];
    [super dealloc];
}
```

キーボードの打鍵終了を検知

- ヘッダにて，UITextFieldDelegateプロトコルを採用
- textFieldShouldReturn メソッドを実装
- Interface Builder にて，UITextField のDelegateに File’s Ownerを指定

```
@interface HelloInterfaceBuilderViewController : UIViewController <UITextFieldDelegate>{
    IBOutlet UITextField *textField;
```

```
38  -(BOOL)textFieldShouldReturn:(UITextField*)theTextField{
39      if(theTextField == textField){
40          [textField resignFirstResponder];
41      }
42      return YES;
43  }
44
```

別の画面への遷移

- 前項までがAppleによるチュートリアルの内容
- 以降は，Interface Builder User’s Guide p.55 Using a View Controller as File’s Owner of a Nib File参照

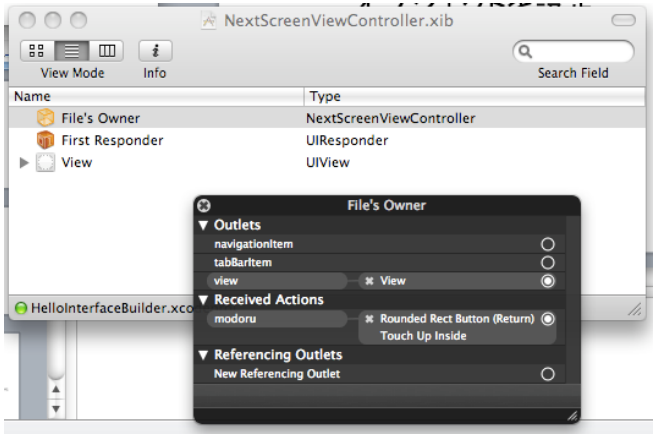
- 新しい画面をつくるためには
  - 新たにView Controllerを定義し新たなView Controller が File’s Owner となるnibファイルを作成
- さらに，画面を切り替えるアクションを定義
  - アクションの中で，View Controller をnibファイルとともに初期化する
  - Appleのインタフェースガイドラインに則った適切な方法で，画面遷移

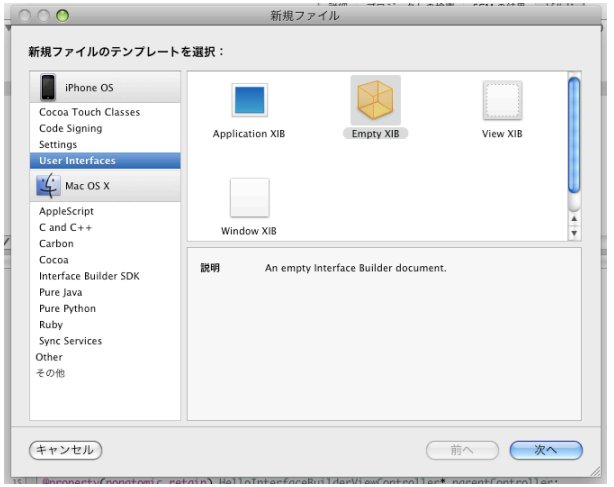
新規カスタムViewControllerの作成

- 新規ファイル→UIViewController サブクラスを作成
- スケルトンがあるのでこのままでもまずは動作可能

新規xibファイル作成

- User Interfaces → Empty XIBを選択
- 必須作業
  - UIView を追加
  - File’s OwnerのIdentity Inspectorで，クラスをNSObjectから先ほど作成したViewController へ切り替え
  - File’s Owner のviewアウトレットに，先ほど追加したUIViewオブジェクトを設定





画面遷移を起こすボタン追加

- HelloInterfaceBuilderViewController.xibに新規ボタン追加
- HelloInterfaceBuilderViewController に新規アクション追加
- 今回の例では，下からせり上がるModalViewとして切り替える
  - Appleのデザインガイドにどのような切り替えが相応しいか整理されてる
  - View Controller Programming Guide pp.11　等

画面切り替えメソッドの内部

- initWithNibName:bundle: で，nib名を指定し初期化する
  - Nib名は拡張子不要
- Bundle は，ここにある書き方でmainBundleを指定すればいいはず
- presentModalViewController で新規画面に切り替わる
- dismissModalViewControllerAnimatedで戻る

```
47  - (IBAction)showNextScreen:(id)sender{
48      NextScreenViewController* nextViewController = [[NextScreenViewController alloc]initWithNibName:@"NextScreenViewController" bundle:[NSBundle mainBundle]];
49      nextViewController.parentController = self;
50      [self presentModalViewController:nextViewController animated:YES];
51      [nextViewController release];
52
53  }
```

このあと

- 新しく追加したViewにボタンを配置する
- ボタンを押すと元の画面に戻るようにする

- ソースコード参照

Xcodeとの連携時の注意

- Interface BuilderファイルはXcodeの「新規ファイル」として作成
- Interface Builder作業中，Xcodeは開いたままにする

- Interface Builderによるクラス作成，アウトレットやアクション追加は推奨されない

アウトレット

- Interface Builder から見える要素
- 実体はインスタンス変数
- IBOutletという特別の接頭辞を付ける

- Interface Builder には自動的に反映される

アクション

- 戻り値として(IBAction)を指定したメソッド
- 必要な引数に応じて，以下のいずれかの形式
  - -(IBAction)respondToButtonClick;
  - -(IBAction)respondToButtonClick:(id)sender
  - -(IBAction)respondToButtonClick:(id)sender forEvent:(UIEvent*)event;

参考になるドキュメント

- iPhone Application Tutorial
  - アプリケーション開発について手順をおって説明
  - カスタムView Controller を作り，Interface Builder で利用する手順を解説
- Interface Builder User Guide
  - Interface Builderの使い方
  - Xcode との関連づけ方
- Resource Programming Guide
  - NIBファイルの内容や読み込み手順について
    - iPhone 独自の，Proxy Object の読み込みについて

- すべて「ヘルプ」→「製品ドキュメント」からたどれます
  - このほかに，クラスライブラリリファレンスも揃う