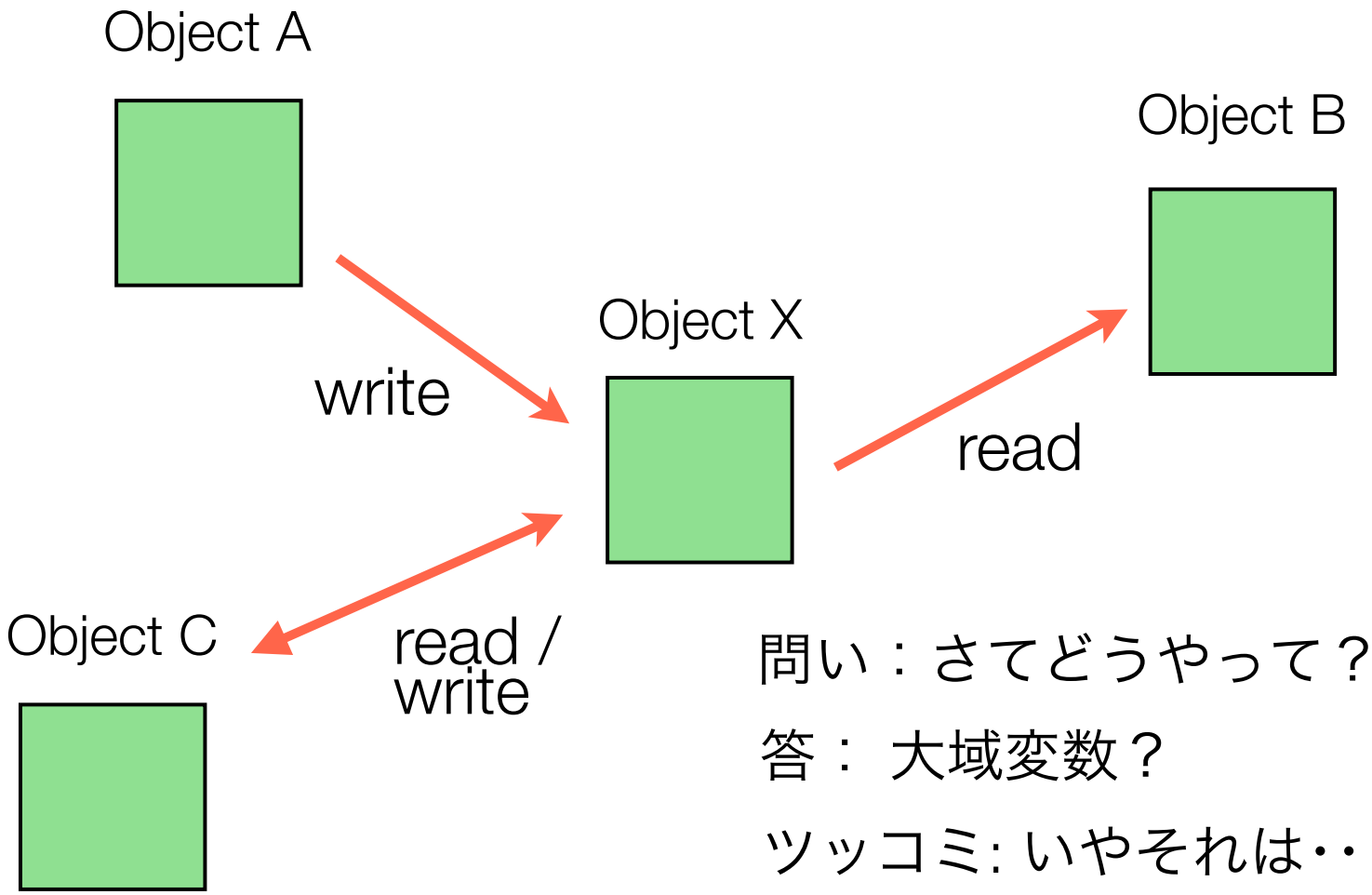
オブジェクト間でデータを共有する

Yutaka Yasuda, Kyoto Sangyo University

複数オブジェクトでデータを共有する



問い：さてどうやって？

答： 大域変数？

ツッコミ: いやそれは・・・

ここはやっぱりオブジェクトで・・・

満たすべき要件

大域変数

- どの関数からでも
- これだと名指しすれば
- 常にシステムの中で
- 特定の変数領域を読み書きできる

作るべきオブジェクト

- どのオブジェクトからでも
- そのアクセス方法でなら
- 常にシステムの中で
- 特定のインスタンス（の変数）を読み書きできる

シングルトン

- 一つのアイディア（今どきはデザインパターンの一つとされる）

  「ただ一つの」インスタンスしか持たないクラス

  「唯一の」＝「同一の」である（今回はこの性質が欲しい）

- 実装

  特別な機能は言語（ex. Objective C）には無い

  「インスタンスの作成」周りを工夫して実現

インスタンス取得メソッドに右のような仕組みを用意する。
結果的に常にシステム中唯一の（つまり同一の）インスタンスにアクセスできる。

```
if( 初回呼び出し ) {
    インスタンスを生成して返す
} else {
    過去に生成したインスタンスを返す
}
```

Singleton 管理のための SingletonManager を定義

SingletonManager.h

```
@interface SingletonManager : NSObject {
@private
  NSInteger managedParam;
}

+ (id)sharedManager;

- (void)setParam:(NSInteger)param;
- (NSInteger)getParam;

@end
```

このシングルトンが管理する値
（今回は一つだけだが多くの場合はズラッと並ぶ？）

インスタンス作成のためのクラスメソッドを一つ用意（後述）

インスタンスに値を書くメソッド

インスタンスから値を得るメソッド

SingletonManager の使い方

値のセット

```
#import "SingletonManager.h"
SingletonManager *single = [SingletonManager sharedManager];
[single setParam:100];
```

シングルトンオブジェクトを取得

シングルトンオブジェクトに値をセットしてみた

値の取得

```
#import "SingletonManager.h"
SingletonManager *single = [SingletonManager sharedManager];
NSInteger param = [single getParam];
```

シングルトンオブジェクトを取得

シングルトンオブジェクトから値を取得してみた

Singleton なインスタンスを作る実装

SingletonManager.m

クラスメソッド sharedManager によって
シングルトンとなるインスタンスを作成する

```
static id theSharedManager = nil;
```

初期化済みフラグ兼
作成したインスタンスを覚えておく変数
（Objective C でのクラス定義 static 変数はクラス変数相当）

```
@implementation SingletonManager

+ (id)sharedManager { // これはクラスメソッド（+つきで定義）
  if (theSharedManager == nil) { // もし初期化を経ていなければ
      theSharedManager = [[self alloc] init]; // 作成・初期化
  }

  return theSharedManager;
}
```

ここで返すインスタンスは
事実上シングルトンとなる

つまり常に同じインスタンスが返される

全体図

SingletonManager.h

```
@interface SingletonManager : NSObject {
   NSInteger managedParam;  (1)
}
```

SingletonManager.m

```
static id theSharedManager = nil; (4)
+ (id)sharedManager {  (2)
   if (theSharedManager == nil) {  (3)
       theSharedManager = [[self alloc] init];
   }
   return theSharedManager;
}

- (void)setParam:(NSInteger)param (5)
{  managedParam=param; }

- (NSInteger)getParam  (5)
{  return managedParam; }

- (id)init  // 内部の初期化（必要なら）
{ [self setParam:0]; return self; }

- (void)dealloc { [super dealloc]; } // 抹消
```

(1) データはインスタンス変数に格納

(2) インスタンスがシングルトンとなるよう、インスタンス作成用クラスメソッド sharedManager に工夫

(3) 工夫の実体は初回時にだけインスタンス作成し、次回からはそのインスタンスを返すメソッドを用意すること

(4) そのためにインスタンスを作った事があるか否か（と作ったインスタンスを覚える）変数を static で用意する

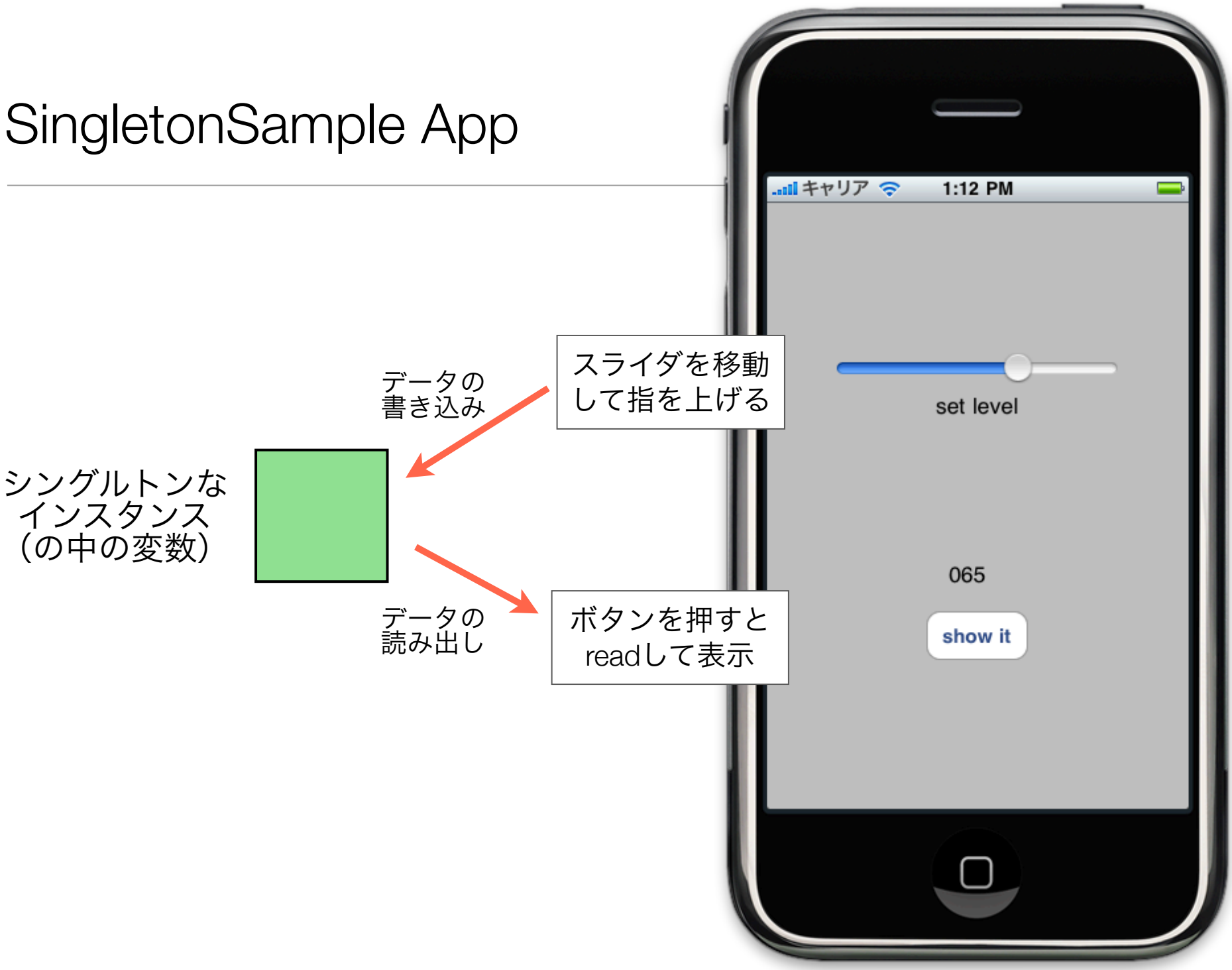
(5) インスタンス変数はメソッドを通じて読み書き

(6) 外から呼び出す時は sharedManager メソッドでインスタンスを得て、そこを通じて読み書きする

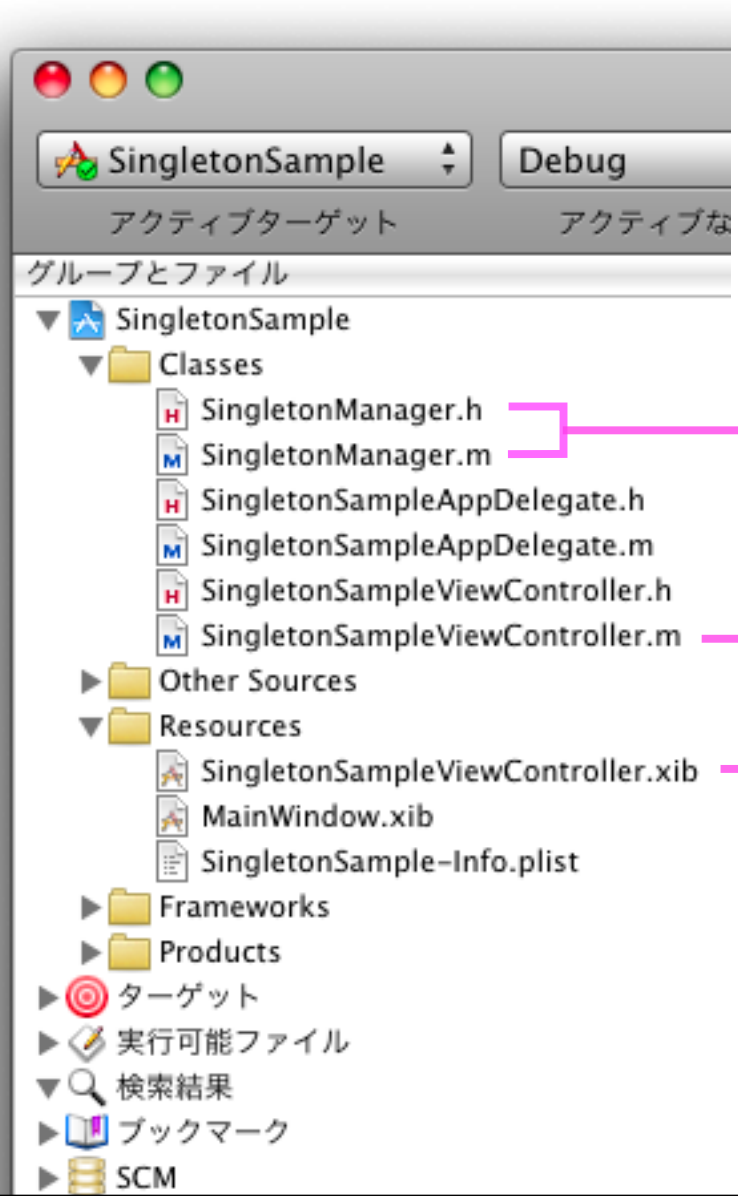
SomeClass.m

```
#import "SingletonManager.h"
.....
SingletonManager *single =
    [SingletonManager sharedManager]; (6)
[single setParam:100];
```

SingletonSample App



登場人物

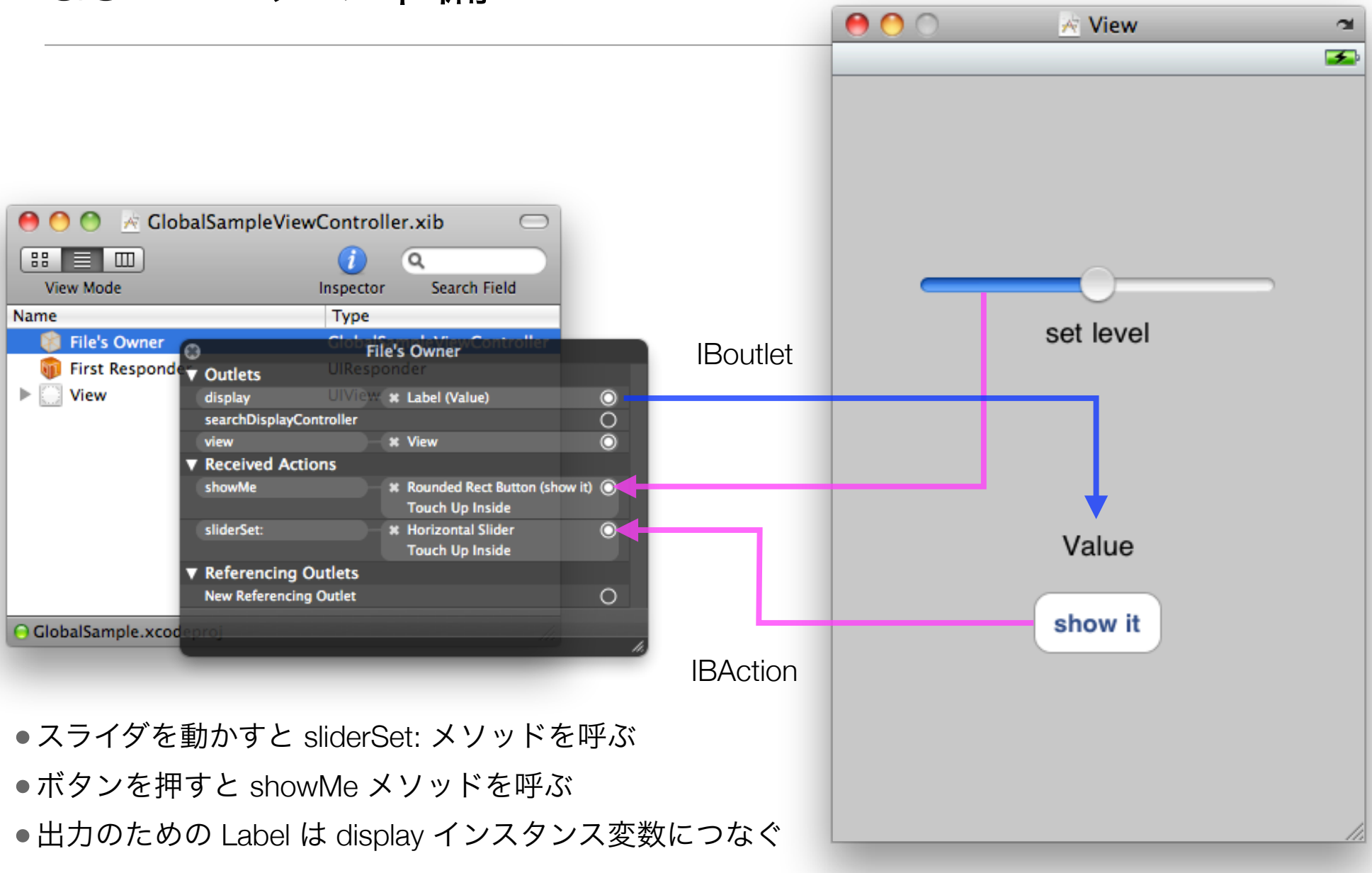


Singleton 処理を行うクラス

今回 Singleton にアクセスするコードは
ここ（ViewController）に書く

ViewController と結びつけられた GUI パーツ定義

GUIパーツの準備



- スライダを動かすと sliderSet: メソッドを呼ぶ
- ボタンを押すと showMe メソッドを呼ぶ
- 出力のための Label は display インスタンス変数につなぐ

SingletonApp2 @property, @synthesize の利用と明示的な初期化等

SingletonManager.h

```
@interface SingletonManager : NSObject {
   NSInteger managedParam;
}
@property NSInteger managedParam;   (1)
+ (id)sharedManager;
@end
```

SingletonManager.m

```
static id theSharedManager = nil;

@synthesize managedParam;(2)
+ (id)sharedManager {
   if (theSharedManager == nil) {
     theSharedManager = [[self alloc] init];
   }
   return theSharedManager;
}
- (id)init  // 内部の初期化（必要なら）
{ self.managedParam=0; return self; }

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

(1) インスタンス変数をプロパティとする

(2) 対応する実装部では @systhesize を書いて setter, getter メソッドを省略

(3) アクセスは . （ドット）演算子でも可能

(4) アプリ起動時に明示的に初期化

(5) 終了時に明示的に抹消

SomeClass.m

```
#import "SingletonManager.h"
.....
SingletonManager *single =
     [SingletonManager sharedManager];
single.managedParam=100;   (3)
```

SingletonSampleAppDelegate.m

```
#import "SingletonManager.h"
- (void)applicationDidFinishLaunching:...
  (void)[SingletonManager sharedManager]; (4)

- (void)dealloc {
   SingletonManager *single =
        [SingletonManager sharedManager];
   [single dealloc];  (5)
```

まとめ

- 大域変数のように、システムのどこからでも読み書きできる変数を用意したい
- シングルトンを用いて実現する
- 実装

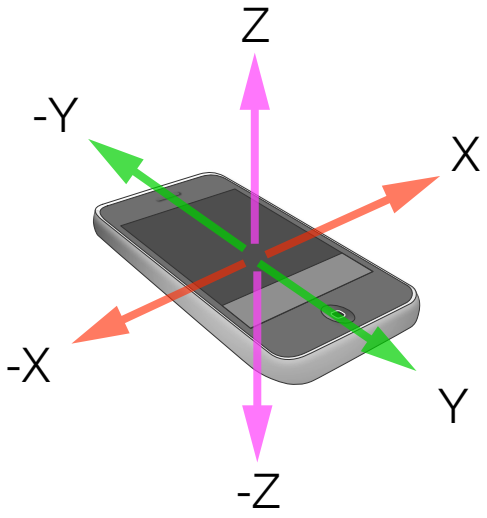
  特別な機能は言語（ex. Objective C）には無い

  「インスタンスの作成」周りを工夫して実現

インスタンス取得メソッドに右のような仕組みを用意する。
結果的に常にシステム中唯一の（つまり同一の）インスタンスにアクセスできる。

```
if( 初回呼び出し ) {
    インスタンスを生成して返す
} else {
    過去に生成したインスタンスを返す
}
```

応用例：加速度センサーの値を使う



システム側
が用意

ユーザの
プログラム

加速度センサーから値
を取って何かする処理

加速度センサーから
値を取るプログラム

何らかの方法に
よるデータの読
み取り

accelerometer
メソッド呼び出
しによる更新

UIAccelerometerDelegate の指定先