iPad&iPhone2

太 田　慎 吾
安倉　秀一郎

１．まえがき

私達は、アプリケーションの開発に興味を持っていて、今回は最近注目を集めている iPad や iPhone のアプリケーションを作ることにしました。

今回、作成したアプリは電卓で、スタンフォード大学の動画を参考にして作りました。
(iTunesU で Stanford Unviersity と検索すると無料の動画があります。)

２．原　　理

作成環境は iMac で Xcode という SDK(**Software development kit**)を使ってアプリを作成しました。

Xcode とは(図 1 参照)

ソフトウェアを開発するためのアップルの統合開発環境で主にプロジェクト管理、コード編集、デバッグを行う為のソフトです。

C、C++、Objective C++、Java、AppleScript、オブジェクト指向記述言語 Objective-C ソースコードをコンパイルする事ができます。

３．作成手順

(1)Xcode の起動

Xcode を起動し、新規プロジェクトを選択してプロジェクト名をいれます。

今回使用したプロジェクト名は Calculator です。

完了をクリックすると必要なファイルが自動的に作成されます。

ファイルの中に MainWindow.xib があるのでそれをクリックします。

クリックすると view や library が表示されます。

(2)画面設定(図 2 参照)

view に button を 16 個配列します。

配列した label に 0~9 までの数字と＋、－、×、÷、＝、sqrt を入れます。

図 1 開発中の画面

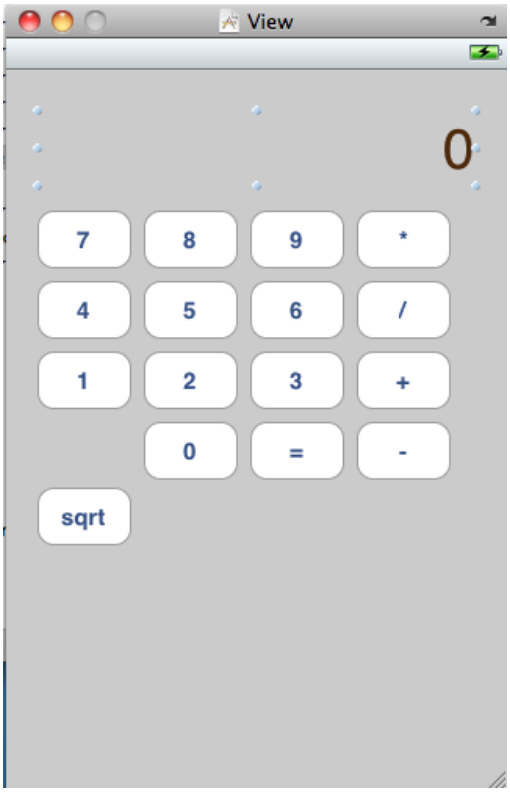

図 2　画面設定

```
//
//  CalculatorBrain.m
//  Calculator
//
//  Created by mizuko on 10/11/22.
//  Copyright 2010 __MyCompanyName__. All rights reserved.
//

#import "CalculatorBrain.h"


@implementation CalculatorBrain

-(void)setOperand:(double)anOperand
{
    operand = anOperand;
}


-(void)performWaitingOperation
{
    if ([@"+" isEqual:waitingOperation]){
        operand = waitingOperand + operand;
    } else if ([@"-" isEqual:waitingOperation]){
        operand = waitingOperand - operand;
    } else if ([@"*" isEqual:waitingOperation]){
        operand = waitingOperand * operand;
    } else if ([@"/" isEqual:waitingOperation]){
        if (operand) {
            operand = waitingOperand / operand;
    }
    }
    }

-(double)performOperation:(NSString *)operation
{
    if ([operation isEqual:@"sqrt"]) {
        operand = sqrt(operand);
    } else {
        [self performWaitingOperation];
        waitingOperand = operand;
        waitingOperation = operation;

    }
    return operand;
}

@end
```

図３　プログラムの一部

図４　完成画面

(3)プログラミング(図３参照)

Classes というディレクトリの中に CalculatorBrain.h などのヘッダーがあるのでそれを開き、図３のようなプログラムを各ヘッダーに追加・削除していきます。

(4)実行(図４参照)

プログラムの追加と削除が終了するとすべてを保存して、ビルドして実行をクリックします。

ここでプログラムミスがあると間違えている場所を自動的に表示してくれるのでその個所を探し、直します。

成功すると iPhone シミュレータが起動され(2)で作った画面が表示されます。

これで電卓が反応すれば完成となります。

電卓が反応しない場合はプログラム的には成功していますが、図２の画面設定の時点でミスがあります。

４．まとめ

今回私たちの作った電卓のアプリケーションには機能的に重大なミスがありました。それはクリアボタンを作り忘れたことです。このボタンがなかったために計算結果がクリアできず、電卓としてはとても使いにくいものになりました。

苦労した点は、使った教材のビデオ音声がすべて英語だったので耳から入る情報はあまりあてになりませんでした。しかし、見よう見まねでもアプリケーションが完成した時の達成感はとても良いものでした。

今回の経験を生かしてこれからもいろいろなアプリケーションを作成していくのが課題です。