Objective C: sending email in your app

Design Team 10
Kaichang Wang

Mar 27 2014
Summary

This document provides a guide to access Email system in an IoS application and open it with email address, subject and content in objective-c. And it shows the usage of MFMailComposeViewController class and the MessageUI framework.

Keywords

Objective-c, Email
Introduction and background

Objective-c is an object oriented programming language that adds smalltalk-style messaging in the C programming language. It is the main language for the OS X and IOS applications. After IOS 3.0, people can take the advantage of the MFMailComposeViewController class and the MessageUI framework to manage the editing and sending an email message. People can use this view controller to display a standard email view inside the application. The user can edit the initial contents and choose to send the email or not. However this interface does not provide a way to verify whether emails were actually sent.
Procedure

1. Create a single view application (figure 1).

![Choose a template for your new project](image)

Figure 1

2. Add the “MessageUI.framework”. The program links three frameworks and libraries initially. In order to access email system, we need to add the MessageUI.framework in the general tab. Click the “+” button under the “Linked Frameworks and Libraries” and find the framework we need in the popup dialog (figure 2).

![Choose frameworks and libraries to add](image)
3. Set up the user interface. In the utilities panel, there are many useful tools, such as view controller, label, button etc. In this app, we need a “Label”, a “Text Field”, a “text View” and a “Button” (figure 3).

4. Link all the components to the header file. Click “Show the Assistant editor” button in the upper right corner and select “ViewController.h” in the right editing page (figure 4). Right click the “subject” text filed and drag it to the header file. Name it “subject” (figure 5). It is similar to deal with the “contents” text view. Select Action in the connection section when drag the “send” button to the header file (figure 6). And then import the “MessageUI.h” file to the header file by “#import <MessageUI/MessageUI.h>” (figure 7).
Figure 4
5. Coding in the "ViewController.m". Select “ViewController.m”. Under the "@implementation ViewController" type "@synthesize subject; @synthesize content;". Under the "(IBAction)send:(id)sender", type:

```
MFMailComposeViewController *mailcomposer;
mailcomposer = [[MFMailComposeViewController alloc] init];
mailcomposer.mailComposeDelegate = self;
```

Go back to the "ViewController.h", type:

```
<MFMailComposeViewControllerDelegate>
```

Now return to the “ViewController.m”, type:

```
NSArray *emailaddress;
emailaddress = [[NSArray alloc] initWithObjects:@"xx@xx.com", nil];
NSString *sendsubject = [[NSString alloc] initWithFormat:@"%@", self.subject.text];
NSString *sendmessage = [[NSString alloc] initWithFormat:@"%@", self.content.text];
```
mailcomposer setToRecipients:emailaddress;
mailcomposer setSubject:sendsubject;
mailcomposer setMessageBody:sendmessage isHTML:NO];
[self presentViewController:mailcomposer animated:YES completion:nil].

Out of the “(IBAction)send:(id)sender”, type:

```
-(void)mailComposeController:(MFMailComposeViewController *)controller didFinishWithResult:(MFMailComposeResult)result error:(NSError *)error{
    [self dismissViewControllerAnimated:YES completion:nil];
}
```

Finally these are the .h file(figure 8) and .m file(figure 9).

```
// ViewController.m
// AppDelegate
// Created by RCB 480 on 3/27/14.
// Copyright (c) 2014 RCB 480. All rights reserved.
//
#import "ViewController.h"

@interface ViewController ()
@end

@implementation ViewController
@synthesize content;

-(void)viewDidLoad
{
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.
}

-(void)didReceiveMemoryWarning
{
    [super didReceiveMemoryWarning];
    // Release of any resources that can be recreated.
}

-(IBAction)send:(id)sender
{
    MFMailComposeViewController *mailcomposer = [[MFMailComposeViewController alloc] init];
    mailcomposer.mailComposeDelegate = self;
    NSArray *emailaddress = [[NSArray alloc] initWithObjects:@"xxxxxx", nil];
    NSString *sendsubject = [NSString stringWithFormat:@"subject", self, subject, text];
    NSString *sendmessage = [NSString stringWithFormat:@"content", self, content, text];
    [mailcomposer setToRecipients:emailaddress];
    [mailcomposer setSubject:sendsubject];
    [mailcomposer setMessageBody:sendmessage isHTML:YES];
    [self presentViewController:mailcomposer animated:YES completion:nil];
}
@end
```

Figure 8
6. Simulate the application. Click the “run” button. The iOS simulator would pop up (figure 10).

```objc
// ViewController.h
// Emailapp
//
// Created by ECE 480 on 3/27/14.
// Copyright (c) 2014 ECE 480. All rights reserved.
//
#import <UIKit/UIKit.h>
#import <MessageUI/MessageUI.h>
@interface ViewController : UIViewController
@property (strong, nonatomic) IBOutlet UITextField *subject;
@property (strong, nonatomic) IBOutlet UITextView *content;
-(IBAction)sender:(id)sender;
@end
```

Figure 9

Figure 10
Click the “send” button, a standard mail system would pop up. In this screen, we can choose to send or cancel the email (figure 11).

To: xx@xx.com

Cc/Bcc:

Subject: subject

contents

Sent from my iPhone Simulator

Figure 11

Conclusion

This application note demonstrate an easy way to set up email system in a particular iOS app, which is useful in our project, because our project needs to email the finalized report to a particular email address.
Reference


MFMailComposeViewController Class Reference:

iOS 5 - Sending E-Mail from Your App:
https://www.youtube.com/watch?v=ly6RpRG_Fis