# 28

# Social Media Integration

Social media integration is not something that most apps can ignore. These days, social media integration in apps is the norm rather than the exception. Fortunately for you, Apple has integrated support for Facebook Twitter, Sina Webo, and Tecent Webo into iOS 9. Posting to social media services has never been easier!

In this lesson, you learn to integrate the Social framework in your iOS apps and allow the user to share a post on Facebook and Twitter from your apps. You can build more complex clients that can access the entire Facebook/Twitter API, but this topic is beyond the scope of this book.

The Social framework is not included in any of the standard iOS project templates that you use when creating a new project. You will need to add a reference to this framework manually. You can do this from the Project Settings page in Xcode. Select the project node in the project navigator to display the settings page. On the settings page, select the build target and switch to the Build Phases tab. Click the plus (+) button under the Link Binary With Libraries category and select `Social.framework` from the list of available frameworks (see Figure 28-1).

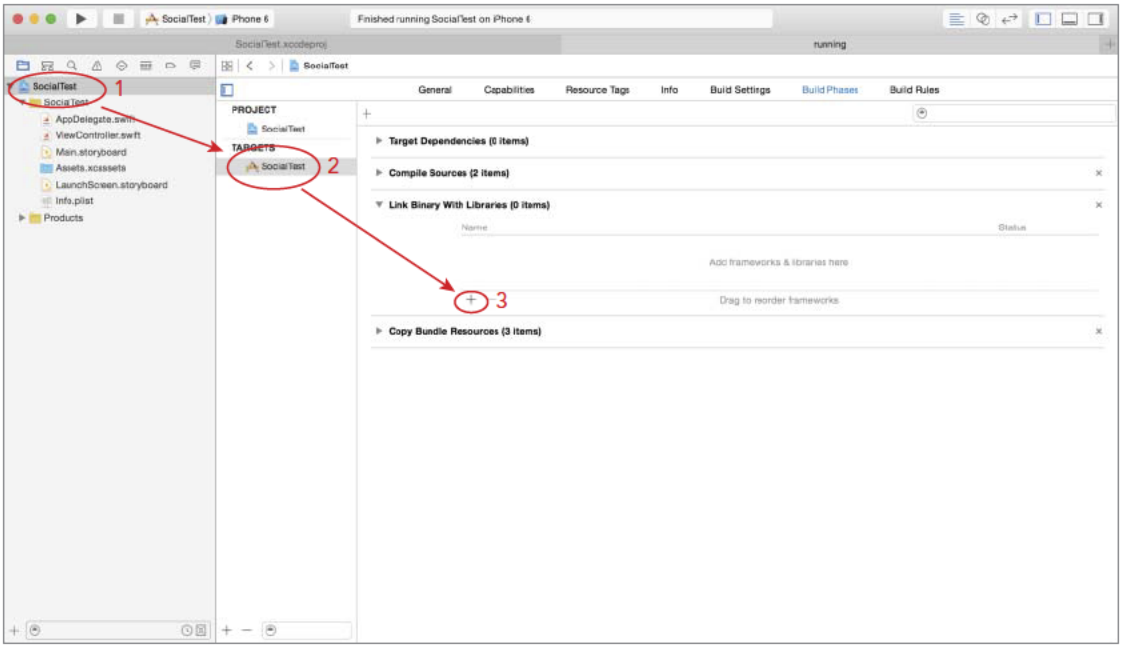


FIGURE 28-1

## THE SHARE SHEET

The Social framework provides a share sheet that you should use in your apps if all you want is a simple "share" feature. The share sheet is an instance of the `SLComposeViewController` class and provides a convenient user interface to allow the user to type a message, attach an image, and add the current location (see Figure 28-2).

The keyboard is displayed automatically when the share sheet appears and disappears automatically when the user presses the Send or Cancel buttons. Creating and displaying the share sheet configured to one of the supported services is a simple matter of instantiating it and presenting it modally:

```
let facebookMessageComposer:SLComposeViewController =
SLComposeViewController(forServiceType: SLServiceTypeFacebook)

self.presentViewController(facebookMessageComposer,
animated: true, completion: nil)
```

When creating an `SLComposeViewController` instance, you must provide a single argument that indicates what social media service you want to use. This argument can have one of four possible values:

- `SLServiceTypeTwitter`
- `SLServiceTypeFacebook`

- `SLServiceTypeSinaWeibo`
- `SLServiceTypeTencentWeibo`


**FIGURE 28-2**

The options displayed in the share sheet will vary depending on the social media service that is configured. Typically, you will want to do this in an action method that is triggered when your user taps on a button in the user interface. Before you show the share sheet for a particular service, you must check to see if the user has created an appropriate account on the system (see Figure 28-3).

For instance, if you detect that the user has not created a Twitter account on the system, you may want to hide the Tweet button from your user interface entirely, or display an alert when the user taps it.

To check the availability of a service, use the `isAvailableForServiceType(serviceType: String!)` class method of the `SLComposeViewController` class as follows:

```
if SLComposeViewController.isAvailableForServiceType(SLServiceTypeFacebook)
{
    // service is available
}
else
{
    // service is not available, perhaps show an alert to the user?
}
```

You can set up the initial text displayed in the tweet sheet prior to displaying it by calling the `setInitialText()` method on the `SLComposeViewController` instance:

```
func setInitialText(text: String!) -> Bool
```

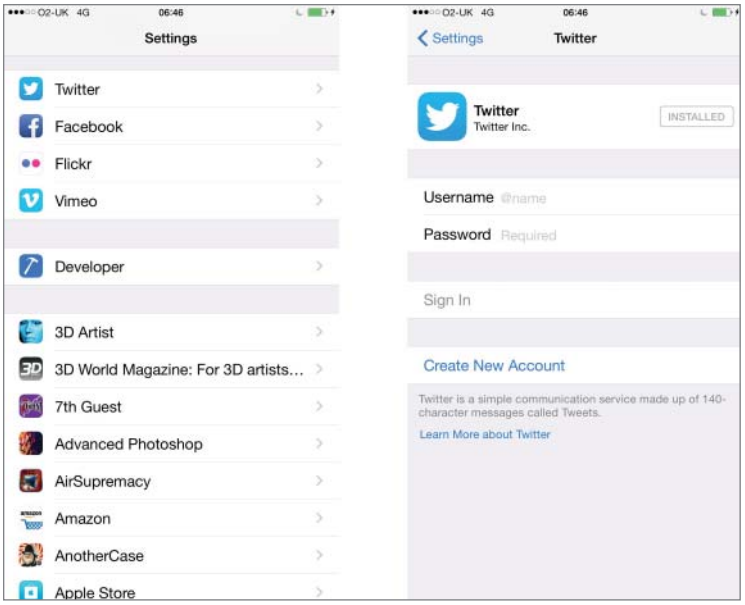


FIGURE 28-3

This method takes one `String` argument that contains the text you want to set and returns a Boolean value that contains the result of the operation. Common reasons why the operation may not be successful are:

- The length of the message is longer than the maximum character limit set by the service.
- You are trying to set the text in the share sheet after it has been displayed.
- The social media service does not allow you to pre-populate content in a share sheet because of legal reasons.

You can attach an image to the share sheet by calling the `addImage()` method on the `SLComposeViewController` instance:

```
func addImage(image: UIImage!) -> Bool
```

This method has one argument that is a `UIImage` object and returns a Boolean result. The image is automatically resized and uploaded to the appropriate social media service by the framework. You must examine the return value to determine if the operation was successful.

To add a URL to the share sheet, use the `addURL()` method:

```
func addURL(url: NSURL!) -> Bool
```

As with the `setInitialText()` and `addImage()` methods, the `addURL()` method returns a Boolean value indicating success or failure. It is important to note that images and URLs take up part of the character limit imposed by the social media service.

You can provide an optional block completion handler that will be executed when the operation has completed. Assuming `messageComposer` is an instance of an `SLComposeViewController` configured for Twitter, you can do this as follows:

```
messageComposer.completionHandler = (result:SLComposeViewControllerResult) in
            // place your code here
}
```

Within the block, you can examine the value of the `result` parameter to get more information on the result of the operation. The value of the `result` parameter depends on which button was pressed by the user, and can be either of the following:

- `Cancelled`
- `Done`

You will need to dismiss the tweet sheet by calling the `dismissModalViewControllerAnimated()` method of the presenting view controller. If you do not provide a block completion handler, the tweet sheet is dismissed automatically regardless of the result of the operation.

## TRY IT

In this Try It, you create a simple iPhone application based on the Single View Application template called `SocialTest` that displays Facebook and Twitter share sheets with pre-populated contents.

## Lesson Requirements

- Launch Xcode.
- Create a new iPhone project based on the Single View Application template.
- Add two `UIButton` instances to the default scene and appropriate action methods to the view controller class.
- Add the Social framework to the build target.
- Add code to display pre-populated share sheets.

> **REFERENCE** *The code for this Try It is available at* `www.wrox.com/go/swiftios`.

## Hints

- To use a share sheet you must add a reference to the Social framework.
- When creating a new project, you can use your website's domain name as the Company Identifier in the Project Options dialog box.

- To show the Object library, select View ➪ Utilities ➪ Show Object Library.
- To show the assistant editor, select View ➪ Assistant Editor ➪ Show Assistant Editor.

## Step-by-Step

- Create a Single View Application in Xcode called `ShareTest`.
  1. Launch Xcode and create a new application by selecting File ➪ New ➪ Project.
  2. Select the Single View Application template from the list of iOS project templates.
  3. In the project options screen, use the following values:
     - **Product Name:** ShareTest
     - **Organization Name:** your company
     - **Organization Identifier:** com.yourcompany
     - **Language:** Swift
     - **Devices:** iPhone
     - **Use Core Data:** Unchecked
     - **Include UI Tests:** Unchecked
     - **Include Unit Tests:** Unchecked
  4. Save the project onto your hard disk.
- Add image resources to your project.
  1. Ensure the project navigator is visible. To show it, select View ➪ Navigators ➪ Show Project Navigator.
  2. Open the `Assets.xcassets` file by clicking on it in the project navigator.
  3. Navigate to the `Images` folder in this chapter's resources from the website.
  4. Create a new Image set by selecting Editor ➪ New Image Set, and name this new image set `Petal`.
  5. Drag the `Petal_1x.jpg`, `Petal_2x.jpg`, and `Petal_3x.jpg` images from this chapter's resources into the appropriate placeholders in the image set.
- Add UI elements to the default scene.
  1. Open the `Main.storyboard` file in the Interface Editor.
  2. From the Object library, drag and drop two buttons onto the scene and place to resemble Figure 28-4.
  3. Select the first button and use the Attribute inspector to change its caption to Share on Facebook and the background color to a shade of gray.

4. Select the second button and use the Attribute inspector to change its caption to Share on Twitter and the background color to a shade of gray.

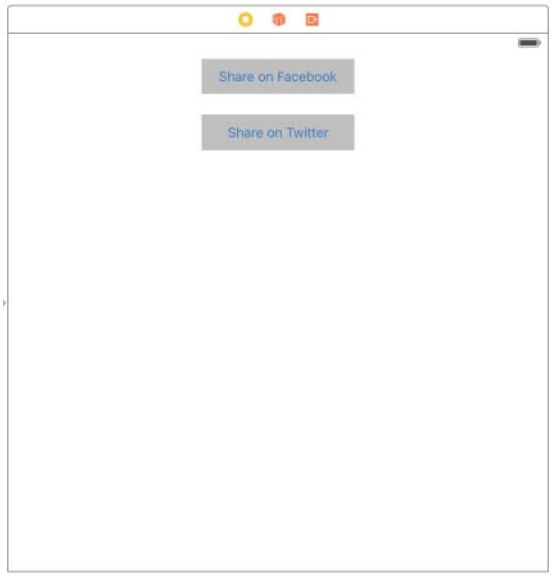


FIGURE 28-4

5. Create layout constraints for each of elements on the storyboard scene using the information in Table 28-1. When creating layout constraints using the constraints editor popover, ensure the Constrain to margins option is unchecked and Update Frames is set to Items of New Constraints.

**TABLE 28-1:** Layout Constraints

| ELEMENT | LEFT | TOP | RIGHT | HEIGHT |
|---|---|---|---|---|
| Facebook button | 20 | 20 | 20 | 40 |
| Twitter button | 20 | 20 | 20 | 40 |

6. Create an action in the view controller class and connect it with the Touch Up Inside event of the Share on Facebook button.
   - Ensure the Assistant editor is visible and the `ViewController.swift` file is loaded in it.
   - Right-click the Share on Facebook button in the scene to display its context menu, and drag from the circle beside the Touch Up Inside item to an empty line in the `ViewController.swift` file.
   - Name the new action `onFacebookShare`.

7. Create an action in the view controller class and connect it with the Touch Up Inside event of the Share on Twitter button.
    - Ensure the Assistant editor is visible and the `ViewController.swift` file is loaded in it.
    - Right-click the Share on Twitter button in the scene to display its context menu, and drag from the circle beside the Touch Up Inside item to an empty line in the `ViewController.swift` file.
    - Name the new action `onTwitterShare`.

- Import the Social framework into the project. The Social framework is not included in any of the standard iOS project templates that you use when creating a new project. You will need to add a reference to this framework manually. You can do so from the Project Settings page in Xcode.

  Ensure the following import statements are located at the top of the `ViewController` class:

```
import UIKit
import Social
```

- Add code to post a tweet.

1. Open the `ViewController.swift` file in the project explorer.
2. Update the empty implementation of the `onTwitterShare` method to resemble the following:

```
@IBAction func onTwitterShare(sender: AnyObject) {

    if
SLComposeViewController.isAvailableForServiceType(SLServiceTypeTwitter)
{
    let twitterMessageComposer:SLComposeViewController =
 SLComposeViewController(forServiceType: SLServiceTypeTwitter)

    twitterMessageComposer.setInitialText("Test Twitter Post")

 twitterMessageComposer.addURL(NSURL(string: "http://www.asmtechnology.com"))

 twitterMessageComposer.addImage(UIImage(named: "Petal"))

 self.presentViewController(twitterMessageComposer,
 animated: true, completion: nil)
}
else
{
    let twitterNotConfiguredAlert =
    UIAlertController(title: "Twitter Not Configured",
    message: "Please setup a twitter account.",
    preferredStyle: UIAlertControllerStyle.Alert)

    twitterNotConfiguredAlert.addAction(UIAlertAction(title: "OK",
    style: UIAlertActionStyle.Default, handler: nil))
```

```
        self.presentViewController(twitterNotConfiguredAlert,
        animated: true, completion: nil)
        }
    }
```

➤ Add code to post to the Facebook timeline.

1. Open the `ViewController.swift` file in the project explorer.
2. Update the empty implementation of the `onFacebookShare` method to resemble:

```
@IBAction func onFacebookShare(sender: AnyObject) {

if
SLComposeViewController.isAvailableForServiceType(SLServiceTypeFacebook)
{
    let facebookMessageComposer:SLComposeViewController =
    SLComposeViewController(forServiceType: SLServiceTypeFacebook)

    facebookMessageComposer.addURL(NSURL(string:
                                "http://www.asmtechnology.com"))

    facebookMessageComposer.addImage(UIImage(named: "Petal"))

    self.presentViewController(facebookMessageComposer,
    animated: true, completion: nil)

}
else
{
    let facebookNotConfiguredAlert =
    UIAlertController(title: "Facebook Not Configured",
    message: "Please setup a facebook account.",
    preferredStyle: UIAlertControllerStyle.Alert)

    facebookNotConfiguredAlert.addAction(UIAlertAction(title: "OK",
    style: UIAlertActionStyle.Default, handler: nil))

    self.presentViewController(facebookNotConfiguredAlert,
    animated: true, completion: nil)
}
     }
```

➤ Test your app in the iOS Simulator.

1. Click the Run button in the Xcode toolbar. Alternatively, you can select Project ➪ Run.
2. Ensure your computer has an active Internet connection.
3. Enter a numeric value for the radius field and tap the Compute Area of Circle button.

**REFERENCE** *To see some of the examples from this lesson, watch the Lesson 28 video online at* `www.wrox.com/go/swiftiosvid`.