SignaKit by InvasiveCode

The **SignaKit** framework allows you to add a handwritten signature to any single- or multi-page PDF file. Additonally, the framework offers functionalities such as PDF browsing and sharing.

Installation

To add the **SignaKit** framework to your Xcode project, go to the **General** preference tab of your **Target**. Drag the `SignaKit.framework` file and drop it onto the **Embedded Binaries**.

Usage

SignaKit exposes the `ISKSignaKitViewController` class and the `ISKSignaKitViewControllerDelegate` protocol.

You must import the framework in your classes:

In Objective-C:

```
@import SignaKit;
```

In Swift:

```
import SignaKit
```

To display a PDF and sign it, you create an instance of `ISKSignaKitViewController` using the provided designated initializer `initWithPDFDocument:`. The argument of the this initializer is a the URL to the PDF file you want to present to the user. Next, you set the SignaKit View Controller's delegate property to any object of your app. Finally, you present the instance of `ISKSignaKitViewController` onto the screen.

The following example shows how to implement these steps in Objective-C:

```
NSURL *documentURL = <the pdf file url>

// You create a new instance of the SignaKit view controller
ISKSignaKitViewController *signaKitViewController =
[[ISKSignaKitViewController alloc]
initWithPDFDocument:documentURL];

// You set its delegate
signaKitViewController.delegate = self;

// You present the view controller on screen
[self.presentViewController:signaKitViewController animated:YES
completion:nil];
```

In Swift:

```
let documentURL = <the pdf file url>

// You create a new instance of the SignaKit view controller
let signaKitViewController =
ISKSignaKitViewController(PDFDocument: documentURL)

// You set its delegate
signaKitViewController.delegate = self

// You present the view controller on screen
presentViewController(signaKitViewController, animated: true,
completion: nil)
```

Implementing the delegate methods

The object that will act as the delegate of the Signa Kit View Controller must conform to the `ISKSignaKitViewControllerDelegate` protocol:

In Objective-C:

```
@import UIKit;
@import SignaKit;

@interface YourViewController : UIViewController
<ISKSignaKitViewControllerDelegate>
```

In Swift:

```
import UIKit
import SignaKit

class YourViewController: UIViewController,
ISKSignaKitViewControllerDelegate
```

The delegate object can now implement the two methods provided by the protocol.

The first method `signaKitViewController:didSignPDFDocumentAtURL:` notifies the delegate that the PDF file has been signed. The second argument is a URL to the newly created PDF containing the handwritten signature (or multiple signatures). The original PDF document remains untouched. The new PDF file is created in the cache folder and you will have to move or copy that document to another location, depending on what you want to do with the new PDF that has been created (save it permanently, send it, etc...).

The protocol provides also an additional method `signaKitViewControllerDidCancel:` in response to the user cancelling the signature.

You are responsible of dismissing the SignaKitViewController.

In Objective-C:

```
- (void)signaKitViewController:(ISKSignaKitViewController
*)signaKitViewController didSignPDFDocumentAtURL:(NSURL
*)documentURL {

    // Do here what you need with documentURL

    // Dismiss the SignaKit view controller
    [self dismissViewControllerAnimated:YES completion:nil];
}

- (void)signaKitViewControllerDidCancel:
(ISKSignaKitViewController *)signaKitViewController
{
    // Dismiss the SignaKit view controller
    [self dismissViewControllerAnimated:YES completion:nil];
}
```

In Swift:

```
func signaKitViewController(signaKitViewController:
ISKSignaKitViewController, didSignPDFDocumentAtURL documentURL:
NSURL?) {

    // Do here what you need with documentURL

    // Dismiss the SignaKit view controller
    self.dismissViewControllerAnimated(true, completion: nil)
}

func signaKitViewControllerDidCancel(signaKitViewController:
ISKSignaKitViewController) {
    // Dismiss the SignaKit view controller
    self.dismissViewControllerAnimated(true, completion: nil)
}
```

IMPORTANT: App Store

The **SignaKit** framework is a universal framework and you can use it in development with both Simulators and Devices.

Before shipping an app with our framework, you need to follow these instructions:

1. Edit the scheme that you use for archiving your App.
2. Select **Archive** and then **Post-actions**.
3. Add a **New Run Script Action**.
4. In the field **Provide build settings from** select the target of your

App.

5. Copy and paste the following code in the script editor.

```
"${SRCROOT}"/SignaKit.framework/INVASIVECODE/archivepostaction.sh
```

This code assumes that `SignaKit.framework` is located at the root directory of Xcode project. If that is not the case, please, change `${SRCROOT}` with the path to the framework location.

With this setup, each time you archive your app with that `Scheme`, after the process of archiving, the post action will run the script and **SignaKit** framework will be ready for the App Store with Bitcode enabled.

License

To use **SignaKit** framework in your App you will need a valid `SignaKit.license` file. This license is only mandatory if you compile your App for distribution, that is, when your App is generated for being deployed to the App Store or Ad Hoc. You can still use **SignaKit** in development without the license file.

Use SignaKit in development

If you are using your App in development, the **SignaKit** framework works with no license. The first time you present the SignaKit View Controller, an alert appears on the screen, and a message is logged on the Xcode console. This alert appears at each App launch. You can develop and test your App freely.

Use SignaKit in production

Before you ship your App to the App Store or do an Ad Hoc deployment, you need to obtain a valid license. Contact **INVASIVECODE** for obtaining a valid license.

To install a license, you need to add a provided license file to your Xcode project.

If you ship an App to the App Store or Ad Hoc without a valid license file, your application will show a message to the user each time the App tries to use the **SignaKit** framework. Please, make sure that you have obtained and installed a valid license file for your App before shipping it.

For more information about license files, please contact INVASIVECODE at hello@invasivecode.com