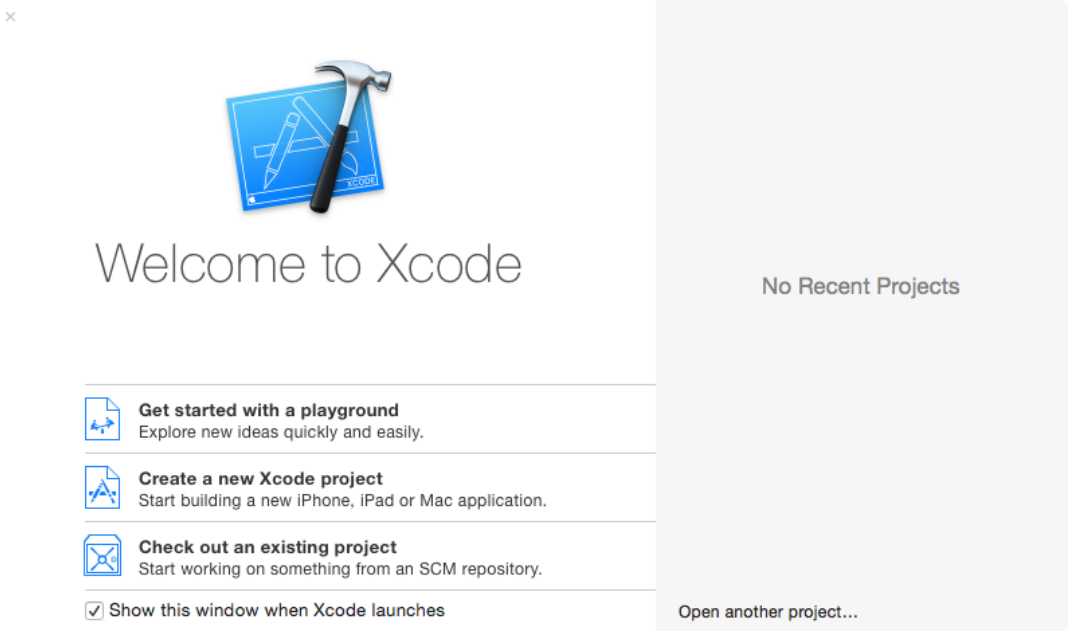
LOG IN WITH RAZER

IOS DEVELOPMENT DOCUMENTATION

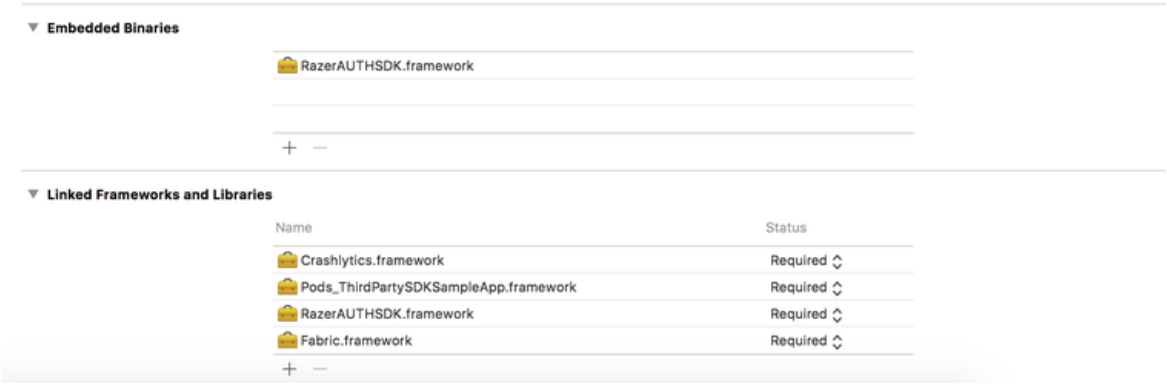
CONTENTS

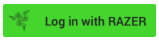
APPLICATION SETUP USING THE RAZERAUTHSDK FRAMEWORK

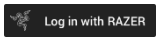
1. Create or open your existing project in Xcode.



2. Download the **RazerAUTHSDK.framework** at github.com/razerofficial/razer_sdk_ios_sample
3. Add the **RazerAUTHSDK.framework** inside the Framework Folder in your app.
4. Go to your App Targets and add the **RazerAUTHSDK.framework** in both the Embedded Binaries and Linked Frameworks and Libraries.



5. Create a Razer app in the Razer Developer Portal.
6. Add 

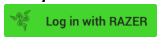
or 

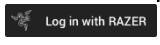
to your project.

7. Import RazerAuthSDK in your AppDelegate and add a callback manager instance for it.

AppDelegate.swift

```
import UIKit
import Fabric
import Crashlytics
import RazerAUTHSDK

@UIApplicationMain

/**
This class is designed and implemented to receive any callbacks from another
App.
*/

class AppDelegate: UIResponder, UIApplicationDelegate {
    let callBackManager = RzLoginView()
    var window: UIWindow?

    /**
    Tells the delegate that the launch process is almost done and the app is
almost ready to run.
    */
    func application(_ application: UIApplication,
didFinishLaunchingWithOptions launchOptions: [UIApplicationLaunchOptionKey:
Any]?)
        -> Bool {
            // Override point for customization after application launch.
            Fabric.with([Crashlytics.self])
            callBackManager.setup(isDebug: true)
            callBackManager.YPrint(value: "working fine.")
            return true
        }
}
```

8. In your View controller, import **RazerAUTHSDK** and create an instance for RzLoginView(). On 

or 

trigger the following API , where **rzAuthSDK** is an instance :

```
        let rzAuthSDK = RzLoginView()
```

The appURLScheme will be your URL Scheme added in your Plist.

```
rzAuthSDK.loginRazerID(urlScheme: appURLScheme, clientID: "CLIENT_ID",
clientSecret: "CLIENT_SECRET", scope: "openid+profile", {
    }) {(error) in
         print(error.debugDescription)
    }
```

9. In iOS, we can send data from one app to another using Query Schemes only. Please make sure to add your URL scheme in your app.

   When a user taps the Authorize / Deny Button in the client app, a call back will be triggered with parameters such as access token. The expiry time will come to your AppDelegate where you need to add the following API to redirect users to the Main Screen (if a user denies the app) and Detail Screen (if a user authorizes the App):

```
func application (_ app: UIApplication, open url: URL, options:
[UIApplicationOpenURLOptionsKey: Any] = [:]) -> Bool
{
    callBackManager.application(url: url, {(params) in
        Params will be access token info from another app
        /**
        After Authorize and receive the data from other app, user will
redirect to Detail View Screen.
        */
         // Redirect to Detail Screen

    }) {(error) in

        /**
         After Deny authorization from another app, user will redirect to
Main Login View.
         */
         //Redirect to Login Screen
    }
    return true
}
```

REFERENCES

We have included a sample demo app using the RazerAUTHSDK.framework which will illustrate how the sample app will work with the authorization screen.

You may refer to the sample GitHub package at github.com/razerofficial/razer_sdk_ios_sample to see how this will be implemented on the iOS platform.

LEGALESE

COPYRIGHT AND INTELLECTUAL PROPERTY INFORMATION