Using Alchemer Mobile with SwiftUI

While Alchemer Mobile (ApptentiveKit) makes heavy use of UIKit internally, it can be used from apps written in SwiftUI.

Registering Alchemer Mobile

If your SwiftUI app already has an app delegate, you can add the code to register ApptentiveKit described in our Quick Start guide.

If your SwiftUI app uses an object that conforms to SwiftUI's `App` protocol, you will have to add the following to your object:

```
    @UIApplicationDelegateAdaptor(AppDelegate.self) var appDelegate
```

You will also have to create an object that conforms to the `UIApplicationDelegate` protocol, and call register in the `application(_ :didFinishLaunchingWithOptions:)` method:

```
 import UIKit
import ApptentiveKit

class AppDelegate: NSObject, UIApplicationDelegate {
    func application(_ application: UIApplication, didFinishLaunchingWithOptions launchOptions: [UIApplication.LaunchOptionsKey : Any]? = nil) -> Bool {

        Apptentive.shared.register(with: .init(key: "<#Your Apptentive App Key#>", signature: "<#Your Apptentive App Signature#>"))

        return true
    }
}
```

Calling ApptentiveKit's `register(with:completion:)` method from an `App` -conforming object's `init` method is not currently supported and will cause ApptentiveKit to fail to initialize properly.

Engaging Events

You can engage events as you would normally from a UIKit-based application using ApptentiveKit, but your calls to the `engage(event:)` method will omit the second (`viewController`) argument.

If engaging the event triggers an Alchemer Mobile interaction, ApptentiveKit will attempt to automatically find the currently-visible view controller and present the interaction using UIKit methods from that view controller.

Related Articles