Symbiote

Swift Auto Analytics Framework for iOS Apps

iOS

Symbiote’s Features

Automatic Analytics

- Getting started takes one line of code
- Symbiote uses Swizzling to attach to all UI ViewControllers and UIButtons
- Important events are automatically detected and sent to the analytics backend(s)
- Symbiote keeps track of the view hierarchy tracking the users journey through the app

Multi-Platform Support

- A variety of analytics backends are supported out of the box
- Enabling or swapping an analytics backend takes one line of code
- Analytics SDKs vendor specific functionality is not disabled or limited
- Additional Analytics Backend Providers can easily be integrated (abstraction layer)

Easily Extendable

- All events run through a processing pipeline before getting sent out to the Analytics Backend(s)
- Events can be filtered and manipulated easily with custom rules and conditions
- Smart Features can be added such as measuring screen time or keeping track of user behavior over multiple screens and scenarios

Supported Analytics Providers

- AWS Mobile Analytics
- Answers Analytics
- Flurry Analytics
- Google Analytics
- More to come …

Components

Core

Swizzler

Analytics Compatible

Analytics Provider

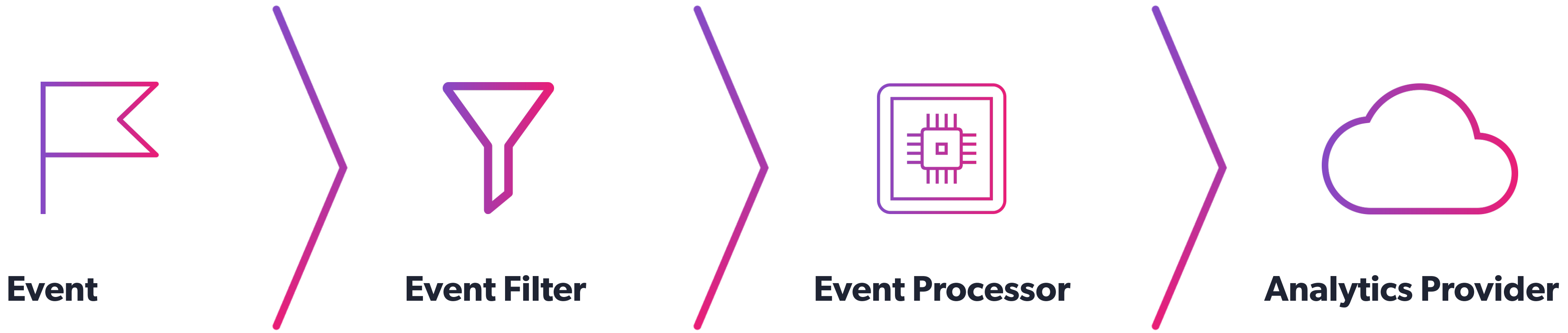
Event

Event Filter

Event Processor

VECTORFORM OPEN SOURCE: SYMBIOTE

Event Processing



Getting Started

```swift
import UIKit
import Symbiote

@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {
    var window: UIWindow?

    override init() {
        super.init()

        // Enable simple log provider to print all output.
        // TODO: Disable for production build!
        Symbiote.SharedInstance.register(analyticsProvider: DebugLogProvider());
    }

}
```

Getting Started

```
import UIKit
import Symbiote

@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {
    var window: UIWindow?

    override init() {
        super.init()

        Symbiote.SharedInstance.register(analyticsProvider: AnswersAnalyticsProvider());
    }

}
```

Auto Analytics: First Results

```
AppDelegate:App/Start          [[ : ]]

AppDelegate:App/Active         [[ : ]]

Swizzle:View/Appear            [[ "ViewName":     ”TestViewController” ]]

Swizzle:Button/TargetSelector [[ "SelectorName":  "presentNavController",
                                 "ViewName":      "UIButton” ]]

Swizzle:View/Disappear         [[ "ViewName":     ”TestViewController",
                                  "ScreenTime":   "8.79127103090286” ]]
```

VECTORFORM OPEN SOURCE: SYMBIOTE

Using Advanced Auto Analytics for UIViewControllers

```
import UIKit
import Symbiote

class SampleViewController: AnalyticsEnabledViewController {

    override func viewDidLoad() {
        super.viewDidLoad()
        analyticsDescription = "SampleViewController"
    }

}
```

Using Advanced Analytics for UIButtons

```
let sampleEventButton:AnalyticsEnabledButton = AnalyticsEnabledButton()
sampleEventButton.setTitle("Sample Button Event", forState: UIControlState.Normal)
sampleEventButton.parentViewController = self
sampleEventButton.analyticsDescription = "SampleEventButton”
```

```
sampleEventButton.customEvent = Event(sender: AnalyticsExtensions.SampleSender,
                                      action: AnalyticsExtensions.SampleAction)
```

Advanced Auto Analytics

```
Subclass:View/Appear      [[ "ViewName": "FirstViewController",
                             "Path": "/FirstViewController” ]]

Subclass:Button/Press     [[ "ViewName": "PresentNavControllerButton",
                             "ControlEvent": "TouchUpInside",
                             "Path": "/FirstViewController/PresentNavControllerButton” ]]

Subclass:View/Disappear   [[ "ViewName": "FirstViewController",
                             "ScreenTime": "3.38345295190811",
                             "Path": "/FirstViewController” ]]

Subclass:View/Appear      [[ "ViewName": "SecondViewController",
                             "Path": "/SampleNavigationController/SecondViewController” ]]

Subclass:Button/Press     [[ "ViewName": "DismissButton",
                             "ControlEvent": "TouchUpInside",
                             "Path":
                               "/SampleNavigationController/SecondViewController/DismissButton”
                            ]]
```

Custom Events

```
let AnalyticsSenderMap = Event.Sender("Map") // Map components that log/send events
let AnalyticsActionLocated = Event.Action("Located")
let AnalyticsDataDescriptorsLocationAccuracy = Event.DataDescriptor("LocationAccuracy")

let locationAccuracy = "10m”

Symbiote.SharedInstance.log(
     event: Event(  sender: AnalyticsSenderMap,
                    action: AnalyticsActionLocated,
                    data: [
                              AnalyticsDataDescriptorsLocationAccuracy: locationAccuracy
                         ])
          )
```

Event Filters & Processors

```
// Sample of how to prohibit all events with a .App sender
Symbiote.SharedInstance.register(
                                   eventProcessor: ProhibitAllProcessor(),
                                   filter: SimpleGenericFilter(
                                                  filterSenders: [Event.Senders.App]
                                           )
                                 )
```

Get up and running in no time!

1. CocoaPods

Just add it to your Podfile:

```
pod 'Symbiote', '0.3.0'
```

2. Open Source on Github

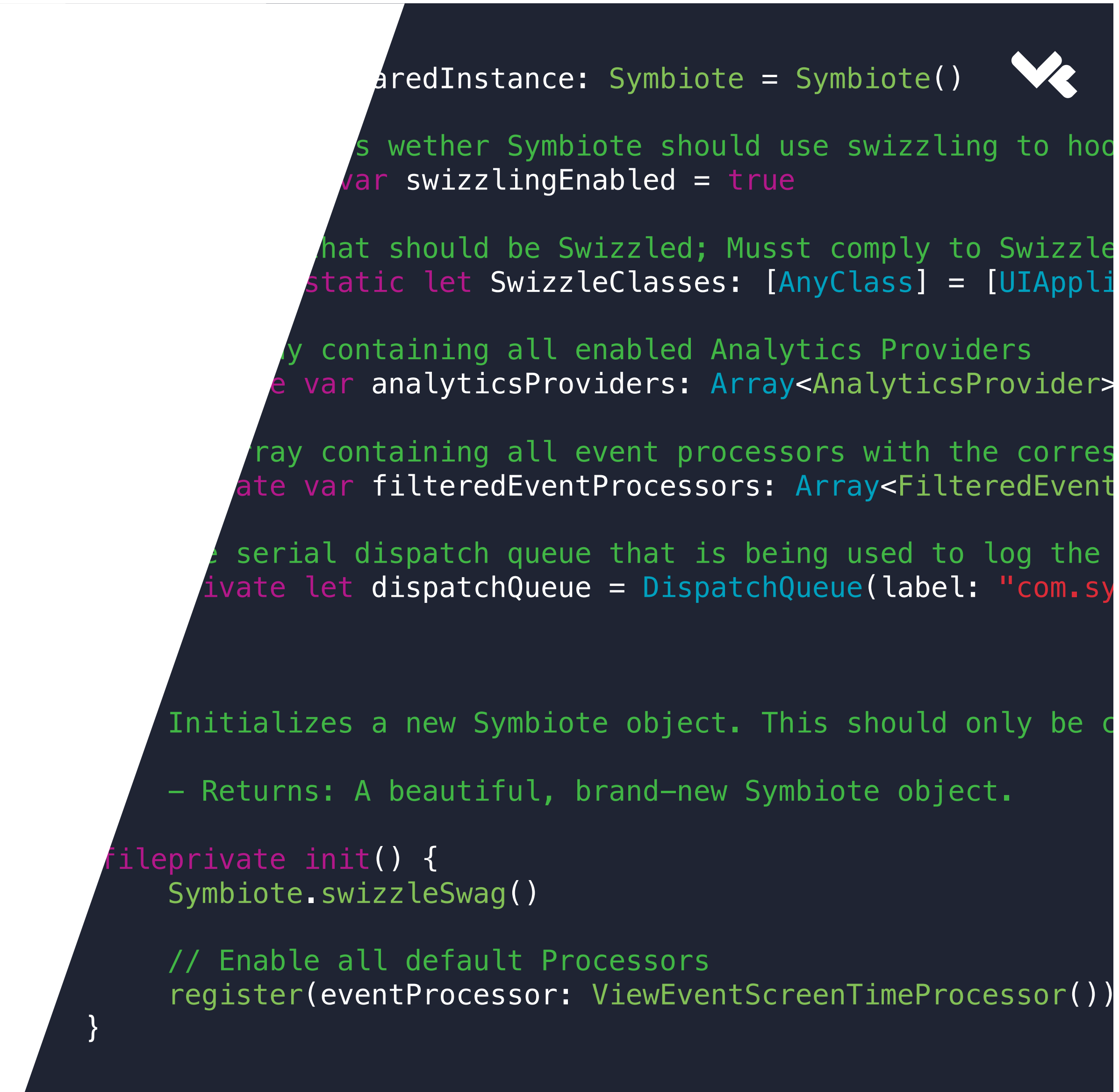
Find docs and source code on Github:

```
https://github.com/vectorform/Symbiote
```

3. Contribute

Want to add a feature?

`We love Pull Requests ♥`



```
aredInstance: Symbiote = Symbiote()

s wether Symbiote should use swizzling to hoo
var swizzlingEnabled = true

hat should be Swizzled; Musst comply to Swizzle
static let SwizzleClasses: [AnyClass] = [UIAppli

y containing all enabled Analytics Providers
e var analyticsProviders: Array<AnalyticsProvider>

ray containing all event processors with the corres
ate var filteredEventProcessors: Array<FilteredEvent

 serial dispatch queue that is being used to log the
ivate let dispatchQueue = DispatchQueue(label: "com.sy


Initializes a new Symbiote object. This should only be c

- Returns: A beautiful, brand-new Symbiote object.

fileprivate init() {
    Symbiote.swizzleSwag()

    // Enable all default Processors
    register(eventProcessor: ViewEventScreenTimeProcessor())
}
```

Thank You

Vectorform Invent with us.