## **Introduction**

OrderCloudSwifty is an SDK for OrderCloud.io's API written in Swift and compatible with Objective-C. All of the methods for the most part are a 1:1 reflection of the API. This guide will cover how to install and get started with OrderCloudSwifty

## **Requirements**

* iOS 8.0+ / Mac OS X 10.9 + / tvOS 9.0+ / watchOS 2.0+
* Xcode 7.2+

## **Installation**

The easiest way to install the SDK is by using [CocoaPods](https://cocoapods.org/), a dependency manager for Swift and Objective-C.

1. Install cocoapods:
2. sudo gem install cocoapods
3. Create your Podfile. In your Xcode project directory, run:
4. pod init
5. Open the Podfile and edit it to include pod 'OrderCloudSwifty'
6. Install OrderCloudSwifty:
7. pod install --verbose
8. Finally, open the .xcworskpace file to build your project

## **Configuration**

You will need to provide the ClientID which identifies the Organization or Buyer Company you will be interacting with. The ClientID can be found in the application tab of your Dashboard. Check out [this](https://devcenter.ordercloud.io/docs/guides/getting-started/dashboard-and-api-console#Applications) guide for more information. Once you have the ClientID you can configure it with the following function.

import OrderCloudSwifty  
OrderCloud.setupClientId("Your\_Client\_ID")

## **Authentication**

To authenticate you will need the username and a password of a user created in the [API Console](https://devcenter.ordercloud.io/docs/guides/getting-started/dashboard-and-api-console#APIConsole). This function will retrieve the access token and set it in the header of every subsequent request

OrderCloud.Auth.authenticate("exampleUsername", password:"examplePassword")

### **OPTIONAL PARAMETERS**

You can also specify the scope of the user as well as a set of callback functions.

OrderCloud.Auth.authenticate("exampleUsername", password:"examplePassword", scope:"FullAccess",   
completionHandler:{exampleSuccessFunction, jsonError: ExampleFunction2, requestFailure: ExampleFunction3  
})

The scope parameter accepts a + delimited list of roles that the user has access to. This will default toFullAccess if none is specified. For more information about roles please check out our guide on [Security Profiles](https://devcenter.ordercloud.io/base.docs.guides(%7BsectionID:%27authentication%27,guideID:%27security-profiles%27%7D))

The completionHandler parameter accepts up to three functions to handle responses: A success function, a function to call when there is an error parsing JSON and finally a function to call when there is an error with the request.

## **Handling Responses**

As we just saw with the authentication function, most functions that call the API will give you the ability to define your own closure with the completionHandler parameter. This ensures that your application stays asynchronous, especially in the context of a UI. They do a lot of the heavy lifting of handling the HTTP response and calling the proper code depending on the response received.

If you prefer not to use the completionHandler parameter you will need to define your own way to handle responses. Check out ApiObject.swift in the source code as a guide. This includes the definition for the three callback functions.

## **SDK Organization**

You’ll notice that there are two different namespaces: OrderCloud and OC. OrderCloud’s namespace is inherent to shared code among all API objects and resources. OC is an indicator that you are working with a class related to the API content itself

## **Conclusion**

You should now have enough information to get you started building on our platform! Remember that the [API reference](https://devcenter.ordercloud.io/docs/api) should be your go-to guide for working with the API and of course check out our API guides as well.

If you notice anything wrong or have any questions please don't hesitate to reach out to us. You can also ask us questions on [StackOverflow](http://stackoverflow.com/questions/tagged/ordercloud) just use the tag ordercloud