**Integrating Zoom Sdk in Project-ios**

### **Requirements**

* Xcode 10
* iOS 8.3+
* armv7, arm64

### **Integration steps :**

1. Download and extract the ZoOm iOS SDK.(can download From [https://dev.zoomlogin.com/zoomsdk/#/downloads](https://dev.zoomlogin.com/zoomsdk/" \l "/downloads))
2. Copy ZoomAuthenticationHybrid.framework into your Xcode project.
3. Import ZoomAuthenticationHybrid
4. Go to Targets → Build Phases and add a New Run Script command (after Embed Frameworks phase):
   1. bash "${BUILT\_PRODUCTS\_DIR}/${FRAMEWORKS\_FOLDER\_PATH}/ZoomAuthenticationHybrid.framework/strip-unused-architectures-from-target.sh"
5. Declare camera access by adding the NSCameraUsageDescription key to the Info.plist along with usage description string.
6. Set up your encryption keys
   1. To create your RSA private key, run: openssl genpkey -algorithm RSA -pkeyopt rsa\_keygen\_bits:2048 | openssl pkcs8 -topk8 -nocrypt > ZoomHybrid\_private.pem
   2. To create your RSA public key, run: openssl rsa -pubout -in ZoomHybrid\_private.pem -out ZoomHybrid.pub
   3. Add ZoomHybrid.pub file to your project.
7. Add [Zoom.sdk preload] (obj-c) Zoom.sdk preload() (swift) to prepare prepares the ZoOm engine to be used by loading portions of the library so that it does not need to occur at the same time the user or app needs to invoke ZoOm functionality, improving UI responsiveness. It is highly recommended to call preload() at the earliest possible moment the application knows it will utilize ZoOm you can add it in - (void)viewDidAppear:(BOOL)animated method(obj – c) **override** **func** viewDidAppear(**\_** animated: Bool) (swift)
8. Zoom initialization
   1. ZoOm must be initialized with a valid Device SDK License before it will function.
   2. Copy your Device SDK License Key from your **[Account Page](https://dev.zoomlogin.com/zoomsdk/" \l "/account)**.
   3. Check the below function for initialization ……

**For Obje-C**

[Zoom.sdk initializeWithAppToken: MY\_ZOOM\_DEVELOPER\_APP\_TOKEN

completion: ^ void (BOOL initializationSuccessful)

{

if(!initializationSuccessful)

{

[self handleUnsuccessfulInit];

return;

}

// Ready to launch ZoOm!

}];

**For Swift**

// Initialize ZoomSDK using a Device SDK License - HTTPS Log mode

Zoom.sdk.initialize(

appToken: MY\_ZOOM\_DEVELOPER\_APP\_TOKEN,

completion:

{

initializationSuccessful in

if(!initializationSuccessful) {

handleUnsuccessfulInit()

return

}

// Ready to launch ZoOm!

}

)

1. Zoom Launching

**For Obje-C**

// Make sure the ZoOm SDK is initialized before launching ZoOm

if([Zoom.sdk getStatus] == ZoomSDKStatusInitialized) {

// Core function calls that create and launch the ZoOm interface

UIViewController \*verificationVC = [Zoom.sdk createVerificationVCWithDelegate:self];

// When presenting the ZoOm interface over your own application, you can keep your application showing in the background by using this presentation style

[verificationVC setModalPresentationStyle:UIModalPresentationOverCurrentContext];

// Present ZoOm's view controller and capture a session

[self presentViewController:verificationVC animated:true completion:nil];

}

**For Swift**

// Make sure the ZoOm SDK is initialized before launching ZoOm

if(Zoom.sdk.getStatus() == .initialized) {

// Core function calls that create and launch the ZoOm interface

let verificationVC = Zoom.sdk.createVerificationVC(delegate: self)

// When presenting the ZoOm interface over your own application, you can keep your application showing in the background by using this presentation style

verificationVC.modalPresentationStyle = UIModalPresentationStyle.overCurrentContext

// Present ZoOm's view controller and capture a session

self.present(verificationVC, animated: true, completion: nil)

}

1. Handle Result

**For Obje-C**

- (void) onZoomVerificationResultWithResult:(id<ZoomVerificationResult>)result {

// CASE: you did not set a public key before attempting to retrieve a facemap.

// Retrieving facemaps requires that you generate a public/private key pair per the instructions at https://dev.zoomlogin.com/zoomsdk/#/zoom-server-guide

if([result status] == ZoomVerificationStatusFailedBecauseEncryptionKeyInvalid) {

UIAlertController \*alertController = [UIAlertController alertControllerWithTitle:@"Public Key Not Set"

message:@"Retrieving facemaps requires that you generate a public/private key pair per the instructions at https://dev.zoomlogin.com/zoomsdk/#/zoom-server-guide"

preferredStyle:UIAlertControllerStyleAlert];

UIAlertAction \*cancelAction = [UIAlertAction actionWithTitle:@"OK" style:UIAlertActionStyleCancel handler:nil];

[alertController addAction:cancelAction];

[self presentViewController:alertController animated:true completion:nil];

}

// CASE: user performed a ZoOm and passed the liveness check

else if([result status] == ZoomVerificationStatusUserProcessedSuccessfully)

{

// ZoOm has completed, pass the facemap to your desired API for processing

[self handleVerificationSuccessResult:result];

}

else {

// Handle other error

}

}

**For Swift**

// handle ZoOm result

func onZoomVerificationResult(result: ZoomVerificationResult) {

// get the status the captured session's result

let resultStatus = result.status

// capture failed

if resultStatus != .UserProcessedSuccessfully {

switch resultStatus {

// handle reason for failed capture

}

return

}

// get the unique sessionID

let sessionID: NSString = result.sessionId

// get the count of the ZoOm sessions performed

let countOfZoomSessionsPerformed: NSInteger = result.countOfZoomSessionsPerformed

if result.faceMetrics != nil {

// get the ZoOm 3D FaceMap

let zoomFacemap: NSData = result.faceMetrics?.zoomFacemap

// get the device partial liveness result

let devicePartialLivenessResult: ZoomDevicePartialLivenessResult = result.faceMetrics?.devicePartialLivenessResult

}

// ZoOm has completed, pass the facemap to your desired API for processing

handleResultFromFaceTecManagedRESTAPICall(result)

}

1. ZoomAPI
   1. For Liveness:
      1. [https://api.zoomauth.com/api/v1/biometrics/liveness](https://accurascan.com/dev_v1/biometrics/liveness)

Method type : post

Request:

sessionId = out put from zoom ios sdk

facemap = out put from zoom ios sdk Image (/result.faceMetrics.zoomFacemap)

Response :

{

"meta": {

"ok": true,

"code": 200,

"mode": "dev",

"message": "The facemap exhibited liveness"

},

"data": {

"livenessResult": "passed",

"livenessScore": 85.1,

"glassesScore": 87.3,

"glassesDecision": true,

"retryFeedbackSuggestion": 0,

"creationStatusFromZoomServer": "The facemap was created successfully.",

"errorFromZoomServer": null

}

}