Initialize Web3auth CoreKit tKey Android SDK

Once you have installed the Web3Auth Core Kit tKey SDK, the next step is to initialize it. This involves a few steps, such as initiating the tKey SDK with the service provider and modules.

- Configuration of Service Provider
- Initializing Service Provider
- Instantiation of tKey

Configuring Service Providerâ

Service Provider intKey generates aShare A , i.e., the private key share managed by a wallet service provider via their authentication flows. This share in ourwallet infrastructure refers to the social login aspect, where we associate a private key share with the user's social login, enabling the seamless login experience.

To configure your service provider, you must useCustomAuth Android SDK . Please note that this SDK is not automatically installed with tKey Android SDK, so you must install it first.

Usage of CustomAuth Swift SDK

Installationâ

Add CustomAuth to Gradleâ

In your project-levelbuild.gradle orsettings.gradle file, add JitPack repository:

dependencyResolutionManagement { repositoriesMode . set (RepositoriesMode . FAIL_ON_PROJECT_REPOS) repositories { google ( ) mavenCentral ( ) maven { url "https://jitpack.io"

}

// <-- Add this line } } Then, in your app-levelbuild.gradle dependencies section, add the following:

dependencies { // ... implementation 'org.torusresearch:customauth-android-sdk:5.0.2' } Latest-SDK Check thelatest version of Web3Auth's CustomAuth Android SDK and update accordingly.

Initializationâ

Initalize the SDK depending on the login you require.

import

org . torusresearch . customauth . CustomAuth ;

private

CustomAuth torusSdk ; MainActivity activity =

((MainActivity)

requireActivity ( )) ;

CustomAuthArgs args =

new

CustomAuthArgs ("https://scripts.toruswallet.io/redirect.html" , TorusNetwork . TESTNET , "torusapp://org.torusresearch.customauthandroid/redirect") ;

this . torusSdk =

new

CustomAuth (args , activity) ;

selectedLoginVerifier

new

LoginVerifier (name :

"Google" , typeOfLogin :

LoginType . GOOGLE , clientId :

GOOGLE_CLIENT_ID , verifier :

GOOGLE_VERIFIER) ;

SubVerifierDetails

â

Parameter Type Mandatory Description typeOfLogin LoginType Yes loginProvider to be used. [google ,facebook ,twitch ,reddit ,discord ,apple ,github ,linkedin ,kakao ,twitter ,weibo ,line ,wechat ,email_password , andjwt] verifier String Yes Web3Auth verifier name clientId String Yes login provider's client Id. jwtParams String No Additional JWT parameters to be passed. isNewActivity boolean No isNewActivity Boolean preferCustomTabs boolean No preferCustomTabs allowedBrowsers String[] No String[] array

CustomAuth

â

CustomAuth(CustomAuthArgs, activity)

CustomAuthArgs

Parameter Type Mandatory Description browserRedirectUri String Yes It refers to a page that the browser should use in the login flow, it should have a http or https scheme. e.g.https://scripts.toruswallet.io/redirect.html redirectUri String Yes It refers to a url for the login flow to redirect into your app, it should have a scheme that is registered by your app, for examplecom.mycompany.myapp://redirect network Network Yes Network to be used. [MAINNET ,TESTNET ,CYAN ,AQUA]

Initializing Service Providerâ

1. triggerLogin()
2. returns a promise that resolve with a Dictionary that contain at leastprivateKey
3. andpublicAddress
4. field.
5. Initialize the activity's postboxKey with the privateKey retrived by the result oftriggerLogin()
6. , that to be used in the next step.

torusLoginResponseCf

torusSdk . triggerLogin (new

SubVerifierDetails (selectedLoginVerifier . getTypeOfLogin ( ) , selectedLoginVerifier . getVerifier ( ) , selectedLoginVerifier . getClientId ( )) . setPreferCustomTabs (true) . setAllowedBrowsers (allowedBrowsers)) ;

torusLoginResponseCf . whenCompleteAsync ((torusLoginResponse , error)

->

{ if

(error !=

null)

{ renderError (error) ; }

else

{ activity . runOnUiThread (( )

->

{ String publicAddress = torusLoginResponse . getPublicAddress ( ) ; activity . postboxKey = torusLoginResponse .

getPrivateKey ( ) . toString (16) ; binding . resultView . append ("publicAddress: "

+ publicAddress) ; }) ; } }) ;

Instantiating tKeyâ

activity . appKey =

new

ThresholdKey (metadata :

null , shares :

null , storage : activity . tkeyStorage , provider : activity . tkeyProvider , transitions :

null , lastFetchedCloudMetadata :

null , enableLogging :

false , manualSync :

false) ;

Parametersâ

Parameter Type Description Mandatory metadata Metadata Metadata object containing the metadata details of tKey. No shares ShareStorePolyIdIndexMap Array of ShareStore with PolyId. No storage StorageLayer Takes in the Storage Provider Instance No provider ServiceProvider Takes in the Service Provider Instance No transitions LocalMetadataTransitions Local metadata transitions No lastFetchedCloudMetadata Metadata lastFetchedCloudMetadata No enableLogging boolean This option is used to specify whether to enable logging or not. No manualSync boolean manual sync provides atomicity to your tkey share. If manualSync is true, you should sync your local metadata transitions manually to your storageLayer, which means your storage layer doesnât know the local changes of your tkey unless you manually sync, gives atomicity. Otherwise, If manualSync is false, then your local metadata changes will be synced automatically to your storage layer. If manualSync = true and want to synchronize manually. No Usage activity . postboxKey = torusLoginResponse . getPrivateKey ( ) . toString (16) ;

activity . tkeyStorage =

new

StorageLayer (enableLogging :

false , hostUrl :

"https://metadata.tor.us" , serverTimeOffset :

2) ;

activity . tkeyProvider =

new

ServiceProvider (enableLogging :

false , postboxKey : activity . postboxKey) ;

activity . appKey =

new

ThresholdKey (metadata :

null , shares :

null , storage : activity . tkeyStorage , provider : activity . tkeyProvider , transitions :

null , lastFetchedCloudMetadata :

null , enableLogging :

false , manualSync :

false) ; Edit this page Previous Install Next Usage