

Initializing PnP Android SDK

After installation, the next step to use Web3Auth is to initialize the SDK.

However, Initialization is a two-step process:

1. [Creating a Web3Auth Instance](#)
2. [Setting a Result URL](#)

Please note that these are the most critical steps where you must pass on different parameters according to the preference of your project. Additionally, You must configure Whitelabeling and Custom Authentication within this step if you wish to customize your Web3Auth Instance.

Create Web3Auth Instance

In your activity, create aWeb3Auth instance with your Web3Auth project's configurations.

web3Auth

Web3Auth (Web3AuthOptions)

Arguments

Web3AuthOptions

The Web3Auth Constructor takes an object withWeb3AuthOptions as input.

- Table
- Interface

Parameter Description context Android context to launch Web-based authentication, usually is the current activity. It's a mandatory field, and acceptsandroid.content.Context as a value. clientId Your Web3Auth Client ID. You can get it from Web3Auth[Dashboard](#) under project details. It's a mandatory field of typeString network Defines the Web3Auth network. It's a mandatory field of type Network. redirectUrl URL that Web3Auth will redirect API responses upon successful authentication from browser. It's a mandatory field of typeUri . whiteLabel? WhiteLabel options for web3auth. It helps you define custom UI, branding, and translations for your brand app. It takesWhiteLabelData as a value. loginConfig? Login config for the custom verifiers. It takesHashMap as a value. useCoreKitKey? Use CoreKit Key to get core kit key. It's an optional field with default value asfalse . chainNamespace? Chain Namespace [EIP155 andSOLANA]. It takesChainNamespace as a value. mfaSettings? Allows developers to configure the Mfa settings for authentication. It takesMfaSettings as a value. sessionTime? It allows developers to configure the session management time. Session Time is in seconds, default is 86400 seconds which is 1 day.sessionTime can be max 7 days data

class

Web3AuthOptions (var context : Context , val clientId : String , val network : Network , var buildEnv : BuildEnv ?

= BuildEnv . PRODUCTION , @Transient

var redirectUrl : Uri ?

=

null , var sdkUrl : String =

getSdkUrl (buildEnv) , val whiteLabel : WhiteLabelData ?

=

null , val loginConfig : HashMap < String , LoginConfigItem

?

=

null , val useCoreKitKey : Boolean ?

```

=
false , val chainNamespace : ChainNamespace ?
= ChainNamespace . EIP155 , val mfaSettings : MfaSettings ?
=
null , val sessionTime : Int ?
=
86400 )

```

Instance[â](#)

web3Auth

```

Web3Auth ( Web3AuthOptions ( context =
this , clientId =
getString ( R . string . web3auth_project_id ) ,
// pass over your Web3Auth Client ID from Developer Dashboard network = Network . MAINNET ,
// pass over the network you want to use (MAINNET or TESTNET or CYAN or AQUA or SAPPHIRE_MAINNET or
SAPPHIRE_TESTNET) redirectUrl = Uri . parse ( "{YOUR_APP_PACKAGE_NAME}://auth" ) ,
// your app's redirect URL ) )
// Handle user signing in when app is not alive web3Auth . setResultUrl ( intent ? . data ) Add the below line to
yourapp/res/values/strings.xml file:

```

```

< resources
< string

```

name

```

" web3auth_project_id "
CLIENT_ID_FROM_WEB3AUTH_DASHBOARD </ string
</ resources

```

Session Management[â](#)

The Session Management feature allows you to check the existing sessions with Web3Auth. The `sessionResponse()` will allow users to remain authenticated with Web3Auth for up to 1 day default, or a maximum of 7 days, or until they logout or session data is cleared. To change the default session time, you can pass `sessionTime` in `Web3AuthOptions` .

```

// Call sessionResponse() in onCreate() to check for any existing session. val sessionResponse : CompletableFuture < Void
= web3Auth . initialize ( ) sessionResponse . whenComplete
{ _ , error -> if
( error ==
null )
{ Log . d ( "MainActivity_Web3Auth" ,
"User already logged in" ) // Add your logic }
else
{ Log . d ( "MainActivity_Web3Auth" , error . message ? :

```

"Something went wrong") // Ideally, you should initiate the login function here. } } note If you're looking forrefreshToken to maintain the JWT session, use this method instead to extend the session.

Set Result URL

```
override  
  
fun  
  
onNewIntent ( intent : Intent ? )  
  
{ super . onNewIntent ( intent )  
  
// Handle user signing in when app is active web3Auth . setResultUrl ( intent ? . data ) }
```

Example

```
class MainActivity :  
    AppCompatActivity ( )  
{ // ... private  
  
lateinit  
  
var web3Auth : Web3Auth  
  
override  
  
fun  
  
onCreate ( savedInstanceState : Bundle ? )  
  
{ super . onCreate ( savedInstanceState ) setContentView ( R . layout . activity_main )
```

web3Auth

```
Web3Auth ( Web3AuthOptions ( context =  
    this , clientId =  
  
    getString ( R . string . web3auth_project_id ) , network = Network . MAINNET , redirectUrl = Uri . parse ( "  
{YOUR_APP_PACKAGE_NAME}://auth" ) ) )  
  
// Handle user signing in when app is not alive web3Auth . setResultUrl ( intent ? . data )  
  
// Calls sessionResponse() to check for any existing session. val sessionResponse : CompletableFuture < Void  
= web3Auth . initialize ( ) sessionResponse . whenComplete  
  
{ _ , error -> if  
  
( error ==  
  
null )  
  
{ Log . d ( "MainActivity_Web3Auth" ,  
  
"User already logged in" ) // Add your logic }  
  
else  
  
{ Log . d ( "MainActivity_Web3Auth" , error . message ?:  
  
"Something went wrong" ) // Ideally, you should initiate the login function here. } }  
  
// Setup UI and event handlers val signInButton = findViewById < Button  
  
    ( R . id . signInButton ) signInButton . setOnClickListener  
  
{
```

```

signIn ( )
} }

override
fun
onNewIntent ( intent : Intent ? )
{ super . onNewIntent ( intent )

// Handle user signing in when app is active web3Auth . setResultUrl ( intent ? . data ) }

private
fun
signIn ( )

{ val selectedLoginProvider = Provider . GOOGLE // Can be GOOGLE, FACEBOOK, TWITCH etc. val
loginCompletableFuture : CompletableFuture < Web3AuthResponse

= web3Auth . login ( LoginParams ( selectedLoginProvider ) )

loginCompletableFuture . whenComplete
{ loginResponse , error -> if
( error ==
null )

{ // render logged in UI println ( loginResponse ) }

else

{ // render error UI } } //... } sample app Get started with sample application found here . Edit this page Previous Install
Next Usage

```