Multi Factor Authentication in PnP Unity SDK

MFA is an extra layer of protection that verifies your identity when accessing your account. To ensure ownership, you must provide two or more different backup factors. You have the option to choose from the device, social, backup factor (seed phrase), and password factors to guarantee access to your Web3 account. Once you create a recovery factor, MFA is enabled, and your keys are divided into three shares for off-chain multi-sig, making the key self-custodial. With backup factors, you can easily recover your account if you lose access to your original device or help login into a new device.

For a dApp, we provide various options to set up Multi-Factor Authentication. You can customize the MFA screen by setting themfaLevel argument. You can enable or disable a backup factor and change their order. Currently, there are four values formfaLevel :

- default
- : presents the MFA screen every third login
- optional
- : presents the MFA screen on every login, but you can skip it
- mandatory
- : make it mandatory to set up MFA after login
- none
- : skips the MFA setup screen

Note If you are using default verifiers, your users may have set up MFA on other dApps that also use default Web3Auth verifiers. In this case, the MFA screen will continue to appear if the user has enabled MFA on other dApps. This is because MFA cannot be turned off once it is enabled. We offer the following backup factors undermfaSettings :

- deviceShareFactor
- ,
- backUpShareFactor
- ,
- socialBackupFactor
- , and
- passwordFactor
- .

Choose the best options that suit your needs to ensure a safe and secure Web3 experience.

Minimum Growth plan required This is a paid feature and the minimumpricing plan to use this SDK in a production environment is theGrowth Plan . You can use this feature in the development environment for free.

Argumentsâ

MFALevel

â

public

enum

MFALevel { [EnumMember (Value =

"default")] DEFAULT , [EnumMember (Value =

"optional")] OPTIONAL , [EnumMember (Value =

"mandatory")] MANDATORY , [EnumMember (Value =

"none")] NONE } Usage public

void

login ( ) { var selectedProvider = Provider . GOOGLE ; var options =

new

LoginParams ( ) { loginProvider = selectedProvider , mfaLevel = MFALevel . MANDATORY } ; web3Auth . login (options) ; }

MFASettings

Minimum SCALE plan required This is a paid feature and the minimum pricing plan to use this SDK in a production environment is the SCALE Plan . You can use this feature in the development environment for free. public

```
class

MfaSetting { public

bool enable {

get ;

set ;

} public

int ? priority {

get ;

set ;

} public

bool ? mandatory {

get ;

set ;

}
// Constructor public

MfaSetting ( bool enable ,

int ? priority ,

bool ? mandatory ) { this . enable = enable ; this . priority = priority ; this . mandatory = mandatory ; } } Usage using

System ; using

System . Linq ; using

System . Collections . Generic ; using

UnityEngine ; using

UnityEngine . UI ; using

Newtonsoft . Json ;

public

class

Web3custom

:

MonoBehaviour { Web3Auth web3Auth ;

// Start is called before the first frame update void

Start ( ) { web3Auth =

GetComponent < Web3Auth

( ) ; web3Auth . setOptions ( new

Web3AuthOptions ( ) { redirectUrl =

new
```

```
Uri ( "torusapp://com.torus.Web3AuthUnity/auth" ) , clientId =

"BAwFgL-r7wzQKmtcdiz2uHJKNZdK7gzEf2q-m55xfzSZOw8jLOyIi4AVvvzaEQO5nv2dFLEmf9LBkF8kaq3aErg" , network =
Web3Auth . Network . TESTNET , mfaSettings =

new

MfaSettings ( new

MfaSetting ( true ,

1 ,

false ) , new

MfaSetting ( true ,

1 ,

true ) , new

MfaSetting ( true ,

1 ,

false ) , new

MfaSetting ( true ,

1 ,

true ) ) } ) ; web3Auth . onLogin += onLogin ; web3Auth . onLogout += onLogout ; } public

void

login ( )

{ } private

void

onLogin ( Web3AuthResponse response )

{ } public

void

logout ( )

{ } private

void

onLogout ( )

{ } }
```