thirdweb SDK

Create Application

thirdweb offers SDKs for a range of programming languages, such as React, React Native, TypeScript, Python, Go, and Unity.

1. In your CLI run the following command:
2. npx thirdweb create
3. --app
4. Input your preferences for the command line prompts:
5.
    1. Give your project a name
6.
    1. Choose your network: We will choose EVM for Moonbeam
7.
    1. Choose your preferred framework: Next.js, CRA, Vite, React Native, Node.js, or Express
8.
    1. Choose your preferred language: JavaScript or TypeScript
9. Use the React or TypeScript SDK to interact with your application's functions. See section on "interact with your contract"

Interact With a Contract

Initialize SDK On Celo

Wrap your application in theThirdwebProvider component and change theactiveChain to Celo

import

{

ThirdwebProvider

}

from

"@thirdweb-dev/react" ; import

{

Celo

}

from

"@thirdweb-dev/chains" ;

const

App

=

( )

=>

{ return

(< ThirdwebProvider

activeChain

{ Celo }

< YourApp

/> </ ThirdwebProvider

) ; } ;

Get Contract

To connect to your contract, use the SDK's getContract method.

import

{ useContract }

from

"@thirdweb-dev/react" ;

function

App ( )

{ const

{ contract , isLoading , error }

=

useContract ("{{contract_address}}") ; }

Calling Contract Functions

- For extension based functions, use the built-in supported hooks. The following is an example using the NFTs extension to access a list of NFTs owned by an address-useOwnedNFTs
- import
- {
- useOwnedNFTs
- ,
- useContract
- ,
- useAddress
- }
- from
- "@thirdweb-dev/react"
- ;
- // Your smart contract address
- const
- contractAddress
- =
- "{{contract_address}}"
- ;
- function
- App
- (
-)
- {
- const
- address
- =
- useAddress
- (
-)
- ;
- const
- {
- contract
- }
- =
- useContract
- (

- contractAddress
-)
- ;
- const
- {
- data
- ,
- isLoading
- ,
- error
- }
- =
- useOwnedNFTs
- (
- contract
- ,
- address
-)
- ;
- }
- Full reference:https://portal.thirdweb.com/react/react.usenft
- Use theuseContractRead
- hook to call any read functions on your contract by passing in the name of the function you want to use.
- import
- {
- useContractRead
- ,
- useContract
- }
- from
- "@thirdweb-dev/react"
- ;
- // Your smart contract address
- const
- contractAddress
- =
- "{{contract_address}}"
- ;
- function
- App
- (
-)
- {
- const
- {
- contract
- }
- =
- useContract
- (
- contractAddress
-)
- ;
- const
- {
- data
- ,
- isLoading
- ,
- error
- }
- =
- useContractRead
- (
- contract
- ,
- "getName"
-)

- ;
- }
- Full reference:https://portal.thirdweb.com/react/react.usecontractread
- Use theuseContractWrite
- hook to call any writefunctions on your contract by passing in the name of the function you want to use.
- import
- {
- useContractWrite
- ,
- useContract
- ,
- Web3Button
- ,
- }
- from
- "@thirdweb-dev/react"
- ;
- // Your smart contract address
- const
- contractAddress
- =
- "{{contract_address}}"
- ;
- function
- App
- (
-)
- {
- const
- {
- contract
- }
- =
- useContract
- (
- contractAddress
-)
- ;
- const
- {
- mutateAsync
- ,
- isLoading
- ,
- error
- }
- =
- useContractWrite
- (
- contract
- ,
- "setName"
-)
- ;
- return
- (
- <
- Web3Button
- contractAddress
- =
- {
- contractAddress
- }
- // Calls the "setName" function on your smart contract with "My Name" as the first argument
- action
- =
- {
- (

-)
- =>
- mutateAsync
- (
- {
- args
- :
- [
- "My Name"
-]
- }
-)
- }
-

- Send Transaction
- <!--
- Web3Button
- -->
-)
- ;
- }
- Full reference:https://portal.thirdweb.com/react/react.usecontractwrite

Connect Wallet

Create a custom connect wallet experience by declaring supported wallets passed to your provider.

import

{ ThirdwebProvider , metamaskWallet , coinbaseWallet , walletConnectV1 , walletConnect , safeWallet , paperWallet , }

from

"@thirdweb-dev/react" ;

function

MyApp ( )

{ return

(< ThirdwebProvider supportedWallets = { [metamaskWallet ( ) , coinbaseWallet ( ) , walletConnect ({ projectId :

"YOUR_PROJECT_ID" ,

// optional }) , walletConnectV1 ( ) , safeWallet ( ) , paperWallet ({ clientId :

"YOUR_CLIENT_ID" ,

// required }) ,] } activeChain = { Celo }

< App

/> </ ThirdwebProvider

) ; } Add in a connect wallet button to prompt end-users to login with any of the above supported wallets.

import

{

ConnectWallet

}

from

"@thirdweb-dev/react" ;

function

App ( )

{ return

< ConnectWallet

/> ; } Full reference:https://portal.thirdweb.com/react/connecting-wallets

Deploy Application

To host your static web application on decentralized storage, run:

npx thirdweb deploy

app By running this command, your application is built for production and stored using Storage. The built application is uploaded to IPFS, a decentralized storage network, and a unique URL is generated for your application.

This URL serves as a permanent hosting location for your application on the web.

If you have any further questions or encounter any issues during the process, please reach out to thirdweb support atsupport.thirdweb.com . Edit this page Previous Celo Libraries & SDKs Next Celo ContractKit