Testing

To wrap up this tutorial, we'll set up a simple automated test for our dapp contracts. We will be usingjest , but any nodejs test runner works fine.

Here we'll only test the happy path for atransfer on our private token contract, but in a real application you should be testing both happy and unhappy paths, as well as both your contracts and application logic. Refer to the fulltesting guide for more info on testing and assertions.

Dependencies

Start by installing our test runner, in this case jest:

yarn add -D jest We'll need toinstall and run the Sandbox .

Test setup

Create a new filesrc/index.test.mjs with the imports we'll be using and an empty test suite to begin with:

import

{ Contract , ExtendedNote , Fr , Note , computeMessageSecretHash , createPXEClient , waitForPXE , }

from

"@aztec/aztec.js" ; import

{ createAccount }

from

"@aztec/accounts/testing" ; import

{ TokenContractArtifact }

from

"@aztec/noir-contracts.js/Token" ;

const

{ PXE_URL

=

"http://localhost:8080" , ETHEREUM_HOST

=

"http://localhost:8545" , }

= process . env ;

describe ("token contract" ,

( )

=>

{ }) ; Let's set up our test suite. We'll make sure the Sandbox is running, create two fresh accounts to test with, and deploy an instance of our contract.aztec.js provides the helper functions we need to do this:

setup let owner , recipient , token ; beforeAll (async

( )

=>

{ const pxe =

createPXEClient (PXE_URL) ; await

waitForPXE (pxe) ; owner =

await

createAccount (pxe) ; recipient =

await

createAccount (pxe) ;

token

await Contract . deploy (owner , TokenContractArtifact ,

[owner . getCompleteAddress ( ) ,

'TokenName' ,

'TKN' ,

18]) . send ( ) . deployed ( ) ;

const initialBalance =

20n ; const secret = Fr . random ( ) ; const secretHash =

await

computeMessageSecretHash (secret) ; const receipt =

await token . methods . mint_private (initialBalance , secretHash) . send ( ) . wait ( ) ;

const storageSlot =

new

Fr (5) ; const noteTypeId =

new

Fr (84114971101151129711410111011678111116101n) ;

// TransparentNote const note =

new

Note ([new

Fr (initialBalance) , secretHash]) ; const extendedNote =

new

ExtendedNote (note , owner . getAddress ( ) , token . address , storageSlot , noteTypeId , receipt . txHash ,) ; await pxe . addNote (extendedNote) ;

await token . methods . redeem_shield ({

address : owner . getAddress ( )

} , initialBalance , secret) . send ( ) . wait ( ) ; } ,

120_000) ;Source code: yarn-project/end-to-end/src/sample-dapp/index.test.mjs#L16-L48 tip Instead of creating new accounts in our test suite, we can use the ones already initialized by the Sandbox upon startup. This can provide a speed boost to your tests setup. However, bear in mind that you may accidentally introduce an interdependency across test suites by reusing the same accounts. Read morehere .

Writing our test

Now that we have a working test environment, we can write our first test for exercising thetransfer function on the token contract. We will use the sameaztec.js methods we used when building our dapp:

test it ('increases recipient funds on transfer' ,

async

( )

=>

{ expect (await token . methods . balance_of_private (recipient . getAddress ( )) . view ( )) . toEqual (0n) ; await token . methods . transfer (owner . getAddress ( ) , recipient . getAddress ( ) ,

20n ,

0) . send ( ) . wait ( ) ; expect (await token . methods . balance_of_private (recipient . getAddress ( )) . view ( )) . toEqual (20n) ; } ,

30_000) ;Source code: yarn-project/end-to-end/src/sample-dapp/index.test.mjs#L50-L56 In this example, we assert that therecipient 's balance is increased by the amount transferred. We could also test that theowner 's funds are decremented by the same amount, or that a transaction that attempts to send more funds than those available would fail. Check out thetesting guide for more ideas.

Running our tests

We can run ourjest tests usingyarn . The quirky syntax is due tojest limitations in ESM support , as well as not picking upmjs file by default:

yarn node --experimental-vm-modules (yarn bin jest) --testRegex '.*.test.mjs'

Next steps

Now that you have finished the tutorial, you can learn more aboutwriting contracts with Noir or read about thefundamental concepts behind Aztec Network . Edit this page

Previous Contract interactions Next Build a Token Bridge