Deploying

Now that we have our completed WrappingERC20 token, the next step is to see if our code actually works!

To do this, we'll be writing tests in typescript using hardhat, and deploying them on ouLocalFhenix environment which we set up earlier.

Note At this stage, using hardhat network is not supported, as Fhenix uses custom extensions to the EVM that enable FHE operations

Compiling the Contractâ

Compiling your contractsâ

First, let's see that our current contract is even valid. Let's run the following:

- npm
- yarn
- pnpm

npm run compile yarn compile pnpm compile This will compile our Solidity contract into bytecode, and generate helper components that we'll be able to use for testing and deployment. If everything worked, you should see something like:

> cross-env TS_NODE_TRANSPILE_ONLY=true hardhat compile

Generating typings for: 5 artifacts in dir: types for target: ethers-v6 Successfully generated 28 typings! Compiled 5 Solidity files successfully

Deploying the Contractâ

Tasksâ

To help us deploy and perform actions you can make use of tasks. We'll add deployment and usage tasks for our new contract. We'll replace the deployment of the default contract with our WrappedERC20. Notice that the differences are mostly just the naming and constructor arguments that are different. Replace thedeploy/deploy.ts with the following content:

import

{

DeployFunction

}

from

"hardhat-deploy/types" ; const hre =

require ("hardhat") ;

const

func :

DeployFunction

=

async

function

( )

{ const

{ ethers }

= hre ; const

```
{ deploy }

= hre . deployments ; const

[ signer ]

=

await ethers . getSigners ( ) ;

const counter =

await

deploy ( "WrappingERC20" ,

{ from : signer . address , args :

[ "Test Token" ,

"TST" ] , log :

true , skipIfAlreadyDeployed :

false } ) ;

console . log (Counter contract: , counter . address ) ; } ;

export

default func ; func . id

=

"deploy_counter" ; func . tags

=
```

["WrappingERC20"] ; Now we can use this task to deploy our contract to either LocalFhenix, or the Devnet chain.

- LocalFhenix
- Fhenix Frontier
- npm
- yarn
- pnpm

get tokens from localfhenix faucet

npm run faucet

if this doesn't work, try running it directly

with "node getFromFaucet"

deploy the contract

npm run deploy:contracts

get tokens from localfhenix faucet

yarn faucet

if this doesn't work, try running it directly

with "node getFromFaucet"

deploy the contract

yarn deploy:contracts

get tokens from localfhenix faucet

pnpm faucet

if this doesn't work, try running it directly

with "node getFromFaucet"

deploy the contract

pnpm deploy:contracts Make sure your deployer address has some tokens, which you can get from []

- npm
- yarn
- pnpm

npm run deploy:contracts --network devnet yarn deploy:contracts --network devnet pnpm deploy:contracts --network devnet Okay, now we know how to create programmatic actions. You can find a few other examples of tasks that interact with the deployed contract in thetasks folder.

Making Changes? When deploying a contract hardhat creates a static deployment. If you want to make changes and redeploy using this method run

- npm
- yarn
- pnpm

npm run clean yarn clean pnpm clean Let's move on to writing a few unit testsEdit this page

Previous Adding View Functions Next Testing on Fhenix