

# End-to-End Example

In this section, we'll explore how to use fhenix.js to send transactions on the Fhenix blockchain.

To send transactions with fhenix.js, we'll first establish a connection to the blockchain, then interact with it using a contract method. For this process, we'll also need to encrypt the transaction data.

Here's a step-by-step explanation, using ethers, though other libraries like web3 can also be used in a similar way.

Let's assume we have a deployed ERC20 contract, only this one uses encrypted inputs and outputs (you can find the solidity code [here](#)). Let's see how we can transfer some of our tokens to another address, while keeping the amount hidden.

## 1. Import fhenixjs and ethers

danger OUTDATED import

```
{
  FhenixClient
}
from
"fhenixjs" ; import
{
  BrowserProvider
}
from
```

"ethers" ; 1. Define the Smart Contract Address and Provider: 2. The smart contract address is the Ethereum address of the deployed contract.provider 3. allows you to interact with the Ethereum blockchain.

```
const
CONTRACT_ADDRESS
=
"0x1c786b8ca49D932AFaDCEc00827352B503edf16c" ; const provider =
new
```

BrowserProvider ( window . ethereum ) ; 1. Create a Client to Interact With Fhenix: 2. The constructor of FhenixClient is used to create an instance of the client with the given provider.

```
const client =
new
```

FhenixClient ( { provider } ) ; 1. Create the Transfer Function: 2. The transfer 3. function is used to send a transaction on the blockchain. It requires the recipient address and the amount to be sent as parameters.

```
const
transfer
=
async
( to , amount )
=>
{ // Create client const client =
new
```

```
FhenixClient ( { provider } ) ;  
  
// get contract const contract =  
await ethers . getContractAt ( CONTRACT_NAME ,  
CONTRACT_ADDRESS ) ; const encryptedAmount =  
await client . encrypt ( number ,  
EncryptionTypes . uint32 ) ;  
  
const response =  
await contract . connect ( SENDER_ACCOUNT ) . transfer ( address , encryptedAmount ) ; return response ; }Edit this page
```

[Previous Permits](#) [Next Network Keys](#)