Set Up

We'll be writing this bot in Typescript. Client libraries and examples for other languages will be available soon.

Starting a new bot project

First, some boilerplate project setup. Run these commands to set up a new typescript project.

mkdir simple-limit-order-bot && cd simple-limit-order-bot yarn init

install typescript & eslint dev dependencies

yarn add -D @types/node @typescript-eslint/eslint-plugin @typescript-eslint/parser dotenv eslint eslint-plugin-tsdoc ts-node typescript

install ethers & mev-share client

yarn add ethers @flashbots/mev-share-client Now in your editor, make asrc directory and add a new file calledindex.ts

Then, import the required dependencies.

src/index.ts

import

MevShareClient ,

{ IPendingTransaction }

from

'@flashbots/mev-share-client' import

{

Contract ,

JsonRpcProvider ,

Wallet

}

from

'ethers' We'll use the mev-share-client library to listen for new pending transactions, and we'll use ethers to create and sign our own transactions, and query the blockchain.

Lastly, we'll create a file called.env in the project root directory to store our private variables, such as private keys and RPC endpoints.

.env

RPC_URL= EXECUTOR_KEY= FB_REPUTATION_KEY= Fill this in with your own values.

- RPC_URL
- is the Ethereum RPC endpoint we'll use to query smart contract values and account balancesAlchemy
- ,Quicknode
- , andInfura
- are popular options for free RPC endpoints.
- EXECUTOR_KEY
- is the private key that will send transactions; it should have at least 0.05 ETH in it to pay for our trade (or more, if you want to make a larger trade than our example).
- FB_REPUTATION_KEY
- is the private key used to sign the payload sent to Flashbots, and is used for tracking searcher reputation. If you earn a high reputation, you may be placed in a high-priority queue, which is prioritized during periods of high traffic. This accountshould not

- have any ETH in it.

Your project should now look like this:

Usecast to generate private keys for cool addresses like this: cast wallet vanity --starts-with babe Starting to generate vanity address... Successfully found vanity address in 0 seconds. Address: 0xbabe32A9112Dc37a0A9274c86CAD0D1676fEA55a Private Key: 😉 ::: info

Next we'll read in the variables from our .env file withdotenv .

Add the following code to your project:

src/index.ts

import

dotenv

from

"dotenv" dotenv . config ( )

const

RPC_URL

= process . env . RPC_URL

||

'http://127.0.0.1:8545' const

EXECUTOR_KEY

= process . env . EXECUTOR_KEY

||

Wallet . createRandom ( ) . privateKey const

FB_REPUTATION_PRIVATE_KEY

= process . env . FB_REPUTATION_KEY

||

Wallet . createRandom ( ) . privateKey Notice we set default values with the|| operator. You can omit these if you prefer the variables to remain undefined when they're not set in the .env file.

Connecting to smart contracts to get prices and make trades

To get the price of our trading pair and make trades, we'll need to interact with a few smart contracts: the Uniswap V2 Router, the ERC20 token contracts, and the factory contract.

- Uniswap V2 Router contract
- : smart contract to trade tokens on Uniswap V2. We also use it to get the market price of the trading pair, by simulating a small trade.
- factory contract
- : this is where Uniswap trading pairs are created. We use it to find the pair address for the tokens we want to trade (e.g. WETH/DAI).
- ERC20 token contract
- : the tokens themselves; in our example, we use the WETH contract to callapprove
- , so that the router can transfer our WETH tokens for us.

To do this in our code, we'll create contract instances using ethers. We instantiate contracts with the ABI and contract address of each contract we want to use. The ABI specifies the functions that can be called on the contract.

For convenience, we've gathered the ABIs required to create ethers contracts for Uniswap V2. Copy these into a new filesrc/abi.ts .

src/abi.ts

export

const

UNISWAP_V2_ABI

=

```
[ { "inputs" : [ { "internalType" : "address" , "name" : "_factory" , "type" : "address" } , { "internalType" : "address" , "name" :
"_WETH" , "type" : "address" } ] , "stateMutability" : "nonpayable" , "type" : "constructor" } , { "inputs" : [ ] , "name" : "WETH" ,
"outputs" : [ { "internalType" : "address" , "name" : "" , "type" : "address" } ] , "stateMutability" : "view" , "type" : "function" } , {
"inputs" : [ { "internalType" : "address" , "name" : "tokenA" , "type" : "address" } , { "internalType" : "address" , "name" :
"tokenB" , "type" : "address" } , { "internalType" : "uint256" , "name" : "amountADesired" , "type" : "uint256" } , {
"internalType" : "uint256" , "name" : "amountBDesired" , "type" : "uint256" } , { "internalType" : "uint256" , "name" :
"amountAMin" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountBMin" , "type" : "uint256" } , {
"internalType" : "address" , "name" : "to" , "type" : "address" } , { "internalType" : "uint256" , "name" : "deadline" , "type" :
"uint256" } ] , "name" : "addLiquidity" , "outputs" : [ { "internalType" : "uint256" , "name" : "amountA" , "type" : "uint256" } , {
"internalType" : "uint256" , "name" : "amountB" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "liquidity" , "type" :
"uint256" } ] , "stateMutability" : "nonpayable" , "type" : "function" } , { "inputs" : [ { "internalType" : "address" , "name" :
"token" , "type" : "address" } , { "internalType" : "uint256" , "name" : "amountTokenDesired" , "type" : "uint256" } , {
"internalType" : "uint256" , "name" : "amountTokenMin" , "type" : "uint256" } , { "internalType" : "uint256" , "name" :
"amountETHMin" , "type" : "uint256" } , { "internalType" : "address" , "name" : "to" , "type" : "address" } , { "internalType" :
"uint256" , "name" : "deadline" , "type" : "uint256" } ] , "name" : "addLiquidityETH" , "outputs" : [ { "internalType" : "uint256" ,
"name" : "amountToken" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountETH" , "type" : "uint256" } , {
"internalType" : "uint256" , "name" : "liquidity" , "type" : "uint256" } ] , "stateMutability" : "payable" , "type" : "function" } , {
"inputs" : [ ] , "name" : "factory" , "outputs" : [ { "internalType" : "address" , "name" : "" , "type" : "address" } ] ,
"stateMutability" : "view" , "type" : "function" } , { "inputs" : [ { "internalType" : "uint256" , "name" : "amountOut" , "type" :
"uint256" } , { "internalType" : "uint256" , "name" : "reserveIn" , "type" : "uint256" } , { "internalType" : "uint256" , "name" :
"reserveOut" , "type" : "uint256" } ] , "name" : "getAmountIn" , "outputs" : [ { "internalType" : "uint256" , "name" : "amountIn" ,
"type" : "uint256" } ] , "stateMutability" : "pure" , "type" : "function" } , { "inputs" : [ { "internalType" : "uint256" , "name" :
"amountIn" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "reserveIn" , "type" : "uint256" } , { "internalType" :
"uint256" , "name" : "reserveOut" , "type" : "uint256" } ] , "name" : "getAmountOut" , "outputs" : [ { "internalType" : "uint256" ,
"name" : "amountOut" , "type" : "uint256" } ] , "stateMutability" : "pure" , "type" : "function" } , { "inputs" : [ { "internalType" :
"uint256" , "name" : "amountOut" , "type" : "uint256" } , { "internalType" : "address[]" , "name" : "path" , "type" : "address[]" } ]
, "name" : "getAmountsIn" , "outputs" : [ { "internalType" : "uint256[]" , "name" : "amounts" , "type" : "uint256[]" } ] ,
"stateMutability" : "view" , "type" : "function" } , { "inputs" : [ { "internalType" : "uint256" , "name" : "amountIn" , "type" :
"uint256" } , { "internalType" : "address[]" , "name" : "path" , "type" : "address[]" } ] , "name" : "getAmountsOut" , "outputs" : [ {
"internalType" : "uint256[]" , "name" : "amounts" , "type" : "uint256[]" } ] , "stateMutability" : "view" , "type" : "function" } , {
"inputs" : [ { "internalType" : "uint256" , "name" : "amountA" , "type" : "uint256" } , { "internalType" : "uint256" , "name" :
"reserveA" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "reserveB" , "type" : "uint256" } ] , "name" : "quote" ,
"outputs" : [ { "internalType" : "uint256" , "name" : "amountB" , "type" : "uint256" } ] , "stateMutability" : "pure" , "type" :
"function" } , { "inputs" : [ { "internalType" : "address" , "name" : "tokenA" , "type" : "address" } , { "internalType" : "address" ,
"name" : "tokenB" , "type" : "address" } , { "internalType" : "uint256" , "name" : "liquidity" , "type" : "uint256" } , {
"internalType" : "uint256" , "name" : "amountAMin" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountBMin"
, "type" : "uint256" } , { "internalType" : "address" , "name" : "to" , "type" : "address" } , { "internalType" : "uint256" , "name" :
"deadline" , "type" : "uint256" } ] , "name" : "removeLiquidity" , "outputs" : [ { "internalType" : "uint256" , "name" : "amountA" ,
"type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountB" , "type" : "uint256" } ] , "stateMutability" : "nonpayable" ,
"type" : "function" } , { "inputs" : [ { "internalType" : "address" , "name" : "token" , "type" : "address" } , { "internalType" :
"uint256" , "name" : "liquidity" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountTokenMin" , "type" :
"uint256" } , { "internalType" : "uint256" , "name" : "amountETHMin" , "type" : "uint256" } , { "internalType" : "address" ,
"name" : "to" , "type" : "address" } , { "internalType" : "uint256" , "name" : "deadline" , "type" : "uint256" } ] , "name" :
"removeLiquidityETH" , "outputs" : [ { "internalType" : "uint256" , "name" : "amountToken" , "type" : "uint256" } , {
"internalType" : "uint256" , "name" : "amountETH" , "type" : "uint256" } ] , "stateMutability" : "nonpayable" , "type" : "function"
} , { "inputs" : [ { "internalType" : "address" , "name" : "token" , "type" : "address" } , { "internalType" : "uint256" , "name" :
"liquidity" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountTokenMin" , "type" : "uint256" } , {
"internalType" : "uint256" , "name" : "amountETHMin" , "type" : "uint256" } , { "internalType" : "address" , "name" : "to" ,
"type" : "address" } , { "internalType" : "uint256" , "name" : "deadline" , "type" : "uint256" } ] , "name" :
"removeLiquidityETHSupportingFeeOnTransferTokens" , "outputs" : [ { "internalType" : "uint256" , "name" : "amountETH" ,
"type" : "uint256" } ] , "stateMutability" : "nonpayable" , "type" : "function" } , { "inputs" : [ { "internalType" : "address" , "name"
: "token" , "type" : "address" } , { "internalType" : "uint256" , "name" : "liquidity" , "type" : "uint256" } , { "internalType" :
"uint256" , "name" : "amountTokenMin" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountETHMin" , "type"
: "uint256" } , { "internalType" : "address" , "name" : "to" , "type" : "address" } , { "internalType" : "uint256" , "name" :
"deadline" , "type" : "uint256" } , { "internalType" : "bool" , "name" : "approveMax" , "type" : "bool" } , { "internalType" : "uint8"
, "name" : "v" , "type" : "uint8" } , { "internalType" : "bytes32" , "name" : "r" , "type" : "bytes32" } , { "internalType" : "bytes32" ,
"name" : "s" , "type" : "bytes32" } ] , "name" : "removeLiquidityETHWithPermit" , "outputs" : [ { "internalType" : "uint256" ,
"name" : "amountToken" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountETH" , "type" : "uint256" } ] ,
"stateMutability" : "nonpayable" , "type" : "function" } , { "inputs" : [ { "internalType" : "address" , "name" : "token" , "type" :
"address" } , { "internalType" : "uint256" , "name" : "liquidity" , "type" : "uint256" } , { "internalType" : "uint256" , "name" :
"amountTokenMin" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountETHMin" , "type" : "uint256" } , {
```

"internalType" : "address" , "name" : "to" , "type" : "address" } , { "internalType" : "uint256" , "name" : "deadline" , "type" : "uint256" } , { "internalType" : "bool" , "name" : "approveMax" , "type" : "bool" } , { "internalType" : "uint8" , "name" : "v" , "type" : "uint8" } , { "internalType" : "bytes32" , "name" : "r" , "type" : "bytes32" } , { "internalType" : "bytes32" , "name" : "s" , "type" : "bytes32" }] , "name" : "removeLiquidityETHWithPermitSupportingFeeOnTransferTokens" , "outputs" : [{ "internalType" : "uint256" , "name" : "amountETH" , "type" : "uint256" }] , "stateMutability" : "nonpayable" , "type" : "function" } , { "inputs" : [{ "internalType" : "address" , "name" : "tokenA" , "type" : "address" } , { "internalType" : "address" , "name" : "tokenB" , "type" : "address" } , { "internalType" : "uint256" , "name" : "liquidity" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountAMin" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountBMin" , "type" : "uint256" } , { "internalType" : "address" , "name" : "to" , "type" : "address" } , { "internalType" : "uint256" , "name" : "deadline" , "type" : "uint256" } , { "internalType" : "bool" , "name" : "approveMax" , "type" : "bool" } , { "internalType" : "uint8" , "name" : "v" , "type" : "uint8" } , { "internalType" : "bytes32" , "name" : "r" , "type" : "bytes32" } , { "internalType" : "bytes32" , "name" : "s" , "type" : "bytes32" }] , "name" : "removeLiquidityWithPermit" , "outputs" : [{ "internalType" : "uint256" , "name" : "amountA" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountB" , "type" : "uint256" }] , "stateMutability" : "nonpayable" , "type" : "function" } , { "inputs" : [{ "internalType" : "uint256" , "name" : "amountOut" , "type" : "uint256" } , { "internalType" : "address[]" , "name" : "path" , "type" : "address[]" } , { "internalType" : "address" , "name" : "to" , "type" : "address" } , { "internalType" : "uint256" , "name" : "deadline" , "type" : "uint256" }] , "name" : "swapETHForExactTokens" , "outputs" : [{ "internalType" : "uint256[]" , "name" : "amounts" , "type" : "uint256[]" }] , "stateMutability" : "payable" , "type" : "function" } , { "inputs" : [{ "internalType" : "uint256" , "name" : "amountOutMin" , "type" : "uint256" } , { "internalType" : "address[]" , "name" : "path" , "type" : "address[]" } , { "internalType" : "address" , "name" : "to" , "type" : "address" } , { "internalType" : "uint256" , "name" : "deadline" , "type" : "uint256" }] , "name" : "swapExactETHForTokens" , "outputs" : [{ "internalType" : "uint256[]" , "name" : "amounts" , "type" : "uint256[]" }] , "stateMutability" : "payable" , "type" : "function" } , { "inputs" : [{ "internalType" : "uint256" , "name" : "amountOutMin" , "type" : "uint256" } , { "internalType" : "address[]" , "name" : "path" , "type" : "address[]" } , { "internalType" : "address" , "name" : "to" , "type" : "address" } , { "internalType" : "uint256" , "name" : "deadline" , "type" : "uint256" }] , "name" : "swapExactETHForTokensSupportingFeeOnTransferTokens" , "outputs" : [ ] , "stateMutability" : "payable" , "type" : "function" } , { "inputs" : [{ "internalType" : "uint256" , "name" : "amountIn" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountOutMin" , "type" : "uint256" } , { "internalType" : "address[]" , "name" : "path" , "type" : "address[]" } , { "internalType" : "address" , "name" : "to" , "type" : "address" } , { "internalType" : "uint256" , "name" : "deadline" , "type" : "uint256" }] , "name" : "swapExactTokensForETH" , "outputs" : [{ "internalType" : "uint256[]" , "name" : "amounts" , "type" : "uint256[]" }] , "stateMutability" : "nonpayable" , "type" : "function" } , { "inputs" : [{ "internalType" : "uint256" , "name" : "amountIn" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountOutMin" , "type" : "uint256" } , { "internalType" : "address[]" , "name" : "path" , "type" : "address[]" } , { "internalType" : "address" , "name" : "to" , "type" : "address" } , { "internalType" : "uint256" , "name" : "deadline" , "type" : "uint256" }] , "name" : "swapExactTokensForETHSupportingFeeOnTransferTokens" , "outputs" : [ ] , "stateMutability" : "nonpayable" , "type" : "function" } , { "inputs" : [{ "internalType" : "uint256" , "name" : "amountIn" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountOutMin" , "type" : "uint256" } , { "internalType" : "address[]" , "name" : "path" , "type" : "address[]" } , { "internalType" : "address" , "name" : "to" , "type" : "address" } , { "internalType" : "uint256" , "name" : "deadline" , "type" : "uint256" }] , "name" : "swapExactTokensForTokens" , "outputs" : [{ "internalType" : "uint256[]" , "name" : "amounts" , "type" : "uint256[]" }] , "stateMutability" : "nonpayable" , "type" : "function" } , { "inputs" : [{ "internalType" : "uint256" , "name" : "amountIn" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountOutMin" , "type" : "uint256" } , { "internalType" : "address[]" , "name" : "path" , "type" : "address[]" } , { "internalType" : "address" , "name" : "to" , "type" : "address" } , { "internalType" : "uint256" , "name" : "deadline" , "type" : "uint256" }] , "name" : "swapExactTokensForTokensSupportingFeeOnTransferTokens" , "outputs" : [ ] , "stateMutability" : "nonpayable" , "type" : "function" } , { "inputs" : [{ "internalType" : "uint256" , "name" : "amountOut" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountInMax" , "type" : "uint256" } , { "internalType" : "address[]" , "name" : "path" , "type" : "address[]" } , { "internalType" : "address" , "name" : "to" , "type" : "address" } , { "internalType" : "uint256" , "name" : "deadline" , "type" : "uint256" }] , "name" : "swapTokensForExactETH" , "outputs" : [{ "internalType" : "uint256[]" , "name" : "amounts" , "type" : "uint256[]" }] , "stateMutability" : "nonpayable" , "type" : "function" } , { "inputs" : [{ "internalType" : "uint256" , "name" : "amountOut" , "type" : "uint256" } , { "internalType" : "uint256" , "name" : "amountInMax" , "type" : "uint256" } , { "internalType" : "address[]" , "name" : "path" , "type" : "address[]" } , { "internalType" : "address" , "name" : "to" , "type" : "address" } , { "internalType" : "uint256" , "name" : "deadline" , "type" : "uint256" }] , "name" : "swapTokensForExactTokens" , "outputs" : [{ "internalType" : "uint256[]" , "name" : "amounts" , "type" : "uint256[]" }] , "stateMutability" : "nonpayable" , "type" : "function" } , { "stateMutability" : "payable" , "type" : "receive" }] export

const

UNISWAP_FACTORY_ABI

=

[{ "inputs" : [{ "internalType" : "address" , "name" : "_feeToSetter" , "type" : "address" }] , "payable" : false , "stateMutability" : "nonpayable" , "type" : "constructor" } , { "anonymous" : false , "inputs" : [{ "indexed" : true , "internalType" : "address" , "name" : "token0" , "type" : "address" } , { "indexed" : true , "internalType" : "address" , "name" : "token1" , "type" : "address" } , { "indexed" : false , "internalType" : "address" , "name" : "pair" , "type" : "address" } , { "indexed" : false , "internalType" : "uint256" , "name" : "" , "type" : "uint256" }] , "name" : "PairCreated" , "type" : "event" } , { "constant" : true , "inputs" : [{ "internalType" : "uint256" , "name" : "" , "type" : "uint256" }] , "name" : "allPairs" , "outputs" : [{ "internalType" : "address" , "name" : "" , "type" : "address" }] , "payable" : false , "stateMutability" : "view" , "type" : "function" } , { "constant" : true , "inputs" : [ ] , "name" : "allPairsLength" , "outputs" : [{ "internalType" : "uint256" , "name" : "" , "type" : "uint256" }] , "payable" : false , "stateMutability" : "view" , "type" : "function" } , { "constant" : false , "inputs" : [{ "internalType" : "address" , "name" : "tokenA" , "type" : "address" } , { "internalType" : "address" , "name" : "tokenB" , "type" : "address" }] , "name" : "createPair"

```
, "outputs" : [ { "internalType" : "address" , "name" : "pair" , "type" : "address" } ] , "payable" : false , "stateMutability" :
"nonpayable" , "type" : "function" } , { "constant" : true , "inputs" : [ ] , "name" : "feeTo" , "outputs" : [ { "internalType" :
"address" , "name" : "" , "type" : "address" } ] , "payable" : false , "stateMutability" : "view" , "type" : "function" } , { "constant" :
true , "inputs" : [ ] , "name" : "feeToSetter" , "outputs" : [ { "internalType" : "address" , "name" : "" , "type" : "address" } ] ,
"payable" : false , "stateMutability" : "view" , "type" : "function" } , { "constant" : true , "inputs" : [ { "internalType" : "address" ,
"name" : "" , "type" : "address" } , { "internalType" : "address" , "name" : "" , "type" : "address" } ] , "name" : "getPair" ,
"outputs" : [ { "internalType" : "address" , "name" : "" , "type" : "address" } ] , "payable" : false , "stateMutability" : "view" ,
"type" : "function" } , { "constant" : false , "inputs" : [ { "internalType" : "address" , "name" : "_feeTo" , "type" : "address" } ] ,
"name" : "setFeeTo" , "outputs" : [ ] , "payable" : false , "stateMutability" : "nonpayable" , "type" : "function" } , { "constant" :
false , "inputs" : [ { "internalType" : "address" , "name" : "_feeToSetter" , "type" : "address" } ] , "name" : "setFeeToSetter" ,
"outputs" : [ ] , "payable" : false , "stateMutability" : "nonpayable" , "type" : "function" } ] export

const

ERC20_ABI

=

[ {

"constant" :

true ,

"inputs" :

[ ] ,

"name" :

"name" ,

"outputs" :

[ {

"name" :

"" ,

"type" :

"string"

} ] ,

"payable" :

false ,

"stateMutability" :

"view" ,

"type" :

"function"

} ,

{

"constant" :

false ,

"inputs" :

[ {

"name" :

"_spender" ,
```

"type" :

"address"

} ,

{

"name" :

"_value" ,

"type" :

"uint256"

}] ,

"name" :

"approve" ,

"outputs" :

[{

"name" :

"" ,

"type" :

"bool"

}] ,

"payable" :

false ,

"stateMutability" :

"nonpayable" ,

"type" :

"function"

} ,

{

"constant" :

true ,

"inputs" :

[ ] ,

"name" :

"totalSupply" ,

"outputs" :

[{

"name" :

"" ,

"type" :

"uint256"

}] ,

"payable" :

false ,

"stateMutability" :

"view" ,

"type" :

"function"

} ,

{

"constant" :

false ,

"inputs" :

[{

"name" :

"_from" ,

"type" :

"address"

} ,

{

"name" :

"_to" ,

"type" :

"address"

} ,

{

"name" :

"_value" ,

"type" :

"uint256"

}] ,

"name" :

"transferFrom" ,

"outputs" :

[{

"name" :

"" ,

"type" :
"bool"
}] ,
"payable" :
false ,
"stateMutability" :
"nonpayable" ,
"type" :
"function"
} ,
{
"constant" :
true ,
"inputs" :
[ ] ,
"name" :
"decimals" ,
"outputs" :
[{
"name" :
"" ,
"type" :
"uint8"
}] ,
"payable" :
false ,
"stateMutability" :
"view" ,
"type" :
"function"
} ,
{
"constant" :
true ,
"inputs" :
[{
"name" :

"_owner" ,
"type" :
"address"
}] ,
"name" :
"balanceOf" ,
"outputs" :
[{
"name" :
"balance" ,
"type" :
"uint256"
}] ,
"payable" :
false ,
"stateMutability" :
"view" ,
"type" :
"function"
} ,
{
"constant" :
true ,
"inputs" :
[ ] ,
"name" :
"symbol" ,
"outputs" :
[{
"name" :
"" ,
"type" :
"string"
}] ,
"payable" :
false ,
"stateMutability" :

"view" ,
"type" :
"function"
} ,
{
"constant" :
false ,
"inputs" :
[{
"name" :
"_to" ,
"type" :
"address"
} ,
{
"name" :
"_value" ,
"type" :
"uint256"
}] ,
"name" :
"transfer" ,
"outputs" :
[{
"name" :
"" ,
"type" :
"bool"
}] ,
"payable" :
false ,
"stateMutability" :
"nonpayable" ,
"type" :
"function"
} ,
{

"constant" :

true ,

"inputs" :

[{

"name" :

"_owner" ,

"type" :

"address"

} ,

{

"name" :

"_spender" ,

"type" :

"address"

}] ,

"name" :

"allowance" ,

"outputs" :

[{

"name" :

"" ,

"type" :

"uint256"

}] ,

"payable" :

false ,

"stateMutability" :

"view" ,

"type" :

"function"

} ,

{

"payable" :

true ,

"stateMutability" :

"payable" ,

"type" :

"fallback"
} ,
{
"anonymous" :
false ,
"inputs" :
[{
"indexed" :
true ,
"name" :
"owner" ,
"type" :
"address"
} ,
{
"indexed" :
true ,
"name" :
"spender" ,
"type" :
"address"
} ,
{
"indexed" :
false ,
"name" :
"value" ,
"type" :
"uint256"
}] ,
"name" :
"Approval" ,
"type" :
"event"
} ,
{
"anonymous" :

```
false ,
"inputs" :
[ {
"indexed" :
true ,
"name" :
"from" ,
"type" :
"address"
} ,
{
"indexed" :
true ,
"name" :
"to" ,
"type" :
"address"
} ,
{
"indexed" :
false ,
"name" :
"value" ,
"type" :
"uint256"
} ] ,
"name" :
"Transfer" ,
"type" :
"event"
```

}] We need to add this import insrc/index.ts , and then write just a little more boilerplate code. It should look like this all together:

```
import
MevShareClient ,
{ IPendingTransaction }
from
'@flashbots/mev-share-client' import
{
```

```
Contract ,
JsonRpcProvider ,
Wallet
}
from
'ethers' import
{
UNISWAP_V2_ABI ,
UNISWAP_FACTORY_ABI ,
ERC20_ABI
}
from
'./abi'
// <-- new import import
dotenv
from
"dotenv" dotenv . config ( )
const
RPC_URL
= process . env . RPC_URL
||
'http://127.0.0.1:8545' const
EXECUTOR_KEY
= process . env . EXECUTOR_KEY
||
Wallet . createRandom ( ) . privateKey const
FB_REPUTATION_PRIVATE_KEY
= process . env . FB_REPUTATION_KEY
||
Wallet . createRandom ( ) . privateKey
// create web3 provider & wallets, connect to mev-share const provider =
new
JsonRpcProvider ( RPC_URL ) const executorWallet =
new
Wallet ( EXECUTOR_KEY , provider ) const authSigner =
new
Wallet ( FB_REPUTATION_PRIVATE_KEY , provider ) const mevshare =
```

MevShareClient . useEthereumGoerli (authSigner) // if you want to connect to mainnet instead: // const mevshare = MevShareClient.useEthereumMainnet(authSigner)

// create contract instances const

UNISWAP_V2_ADDRESS

=

'0x7a250d5630b4cf539739df2c5dacb4c659f2488d' const

UNISWAP_FACTORY_ADDRESS

=

'0x5C69bEe701ef814a2B6a3EDD4B1652CB9cc5aA6f' const uniswapRouterContract =

new

Contract (UNISWAP_V2_ADDRESS ,

UNISWAP_V2_ABI , executorWallet) const uniswapFactoryContract =

new

Contract (UNISWAP_FACTORY_ADDRESS ,

UNISWAP_FACTORY_ABI , provider)

/ *While we're here, let's also set some useful constants we'll use later*/ // discount we expect from the backrun trade (basis points): const

DISCOUNT_IN_BPS

=

40n // try sending a backrun bundle for this many blocks: const

BLOCKS_TO_TRY

=

24 // WETH: const

SELL_TOKEN_ADDRESS

=

'0xc02aaa39b223fe8d0a0e5c4f27ead9083c756cc2' const

SELL_TOKEN_AMOUNT

=

100000000n // DAI: const

BUY_TOKEN_ADDRESS

=

'0x6b175474e89094c44da98b954eedeac495271d0f' const

BUY_TOKEN_AMOUNT_CUTOFF

=

SELL_TOKEN_AMOUNT

*

1800n

const

TX_GAS_LIMIT

=

400000 const

MAX_GAS_PRICE

=

20n const

MAX_PRIORITY_FEE

=

5n const

GWEI

=

10n

**

9n uniswapRouterContract is the contract we use to execute trades.

uniswapFactoryContract is used to find the contract address of the token pair we trade on (e.g.WETH/DAI).

SELL_TOKEN_ADDRESS is the token we want to sell (in this case, WETH). The token we're buying in this example is DAI . Choose whichever tokens you want to trade if you're following along — you can check to see if your tokens have a pair by calling thegetPair function with your token addresses on the Uniswapfactory contract .

SELL_TOKEN_AMOUNT specifies 0.1 gwei of WETH to spend, and inBUY_TOKEN_AMOUNT_CUTOFF we specify that we want to buy when the token price is1800 DAI/WETH.

We set our gas fees (seeMAX_GAS_PRICE andMAX_PRIORITY_FEE) to constant values for the example. If you don't mind possibly paying more gas, we recommend setting your gas parameters so that they follow the base fee. Ethers has a function for this calledgetFeeData .