RSK Next JS Box¶

This box comes with everything you need to start using smart contracts from a react app andNext JS onRSK Blockchain . It includes network configurations for Mainnet, Testnet and the SimpleStorage contract as an example to deploy.

Requirements¶

1. NPM (Node Package Manager)
2. Node.js and NPM are needed, though both are usually installed at once.

Go toNode.js if you need to install it.

1. Truffle

Install Truffle globally:

npm install -g truffle

Installation¶

1. Create a new folder. For example, create the folderrsk-next
2. . Navigate to the folder in the terminal.

mkdir rsk-nextcd

rsk-next 1. Run the unbox command. It can take some time, as this will install all necessary dependencies.

truffle unbox rsksmart/rsk-next-box A Create-React-Next-App is generated in theapp directory.

This is the result using Windows OS:

Development console¶

Truffle has an interactive console that also spawns a development blockchain. This is very useful for compiling, deploying and testing locally.

1. Run the development console. This command is successful if you see a list of 10 accounts, a mnemonic and the command prompt is nowtruffle(develop)>

truffle develop You will now be in the truffle develop REPL with seeded accounts and their associated private keys listed.

C:\RSK\rsk-next>truffle develop

Truffle Develop started at http://127.0.0.1:8545/

Accounts: (0) 0x1056f747cf4bc7710e178b2aeed4eb8c8506c728 (1) 0x45a71c00382c2898b5d6fae69a6f7bfe6edab80c (2) 0x1596384706dc9ac4cca7f50279a4abe591d6c3fe (3) 0x9576d0a496b645baa64f22aceb2328e7468d4113 (4) 0xd431572eef7d77584d944c1809398a155e89f830 (5) 0x92c111839718fe0800fadccc67068b40b8524a0f (6) 0x6da22b5a027146619bfe6704957f7f36ff029c48 (7) 0x2c3a82d8c3993f8c80dcaf91025437bd057df867 (8) 0xc43ae7a44f7deb759177b7093f06512a0a9ff5d7 (9) 0xe61bf00cd7dce248449cfe58f23a4ef7d542bc0b

Private Keys: (0) f32f32839fe27ad906b63eafb326f26fed95c231e3c5e33c7cdd08f62db63167 (1) ebef990088f27f6ef13b5e52a77d5dcc5a76862a701908c586d01b6fe93562b3 (2) 598ccae5e4436fedeb0e798c0d254789c55a63401ebfc3ae8ddde29634ddfcde (3) 09934b80f391e0024b8cb00cd73790fdf64c4d0509e144766414fee317cd3f4e (4) ac745b84b6574b5738d364b43e0d471c9d5107504acc709c90f6f091b78c751b (5) 449654cde095f2349113ef12a93e139b4302bc95adb3619d08adf53dde9b8847 (6) c217f12a89c352fc70b5f1bd5742314b4fb1bb1e35cb779fdb3c2390106355db (7) 1d4c74dfa4e99e161130c18cc63938bb120a128cefbf1b9188efc678bf5722cb (8) 0f44e0becf2e090db498a1b747d2a758fcc81fb0241f350d61117a9c6b1fa82e (9) 85218c5eec657470dafeb09e6f7101f91d21bfe822fbeeecfc9275f798662a63

Mnemonic: virtual valve razor retreat either turn possible student grief engage attract fiber

⚠ Important ⚠ : This mnemonic was created for you by Truffle. It is not secure. Ensure you do not use it on production blockchains, or else you risk losing funds.

truffle(develop)> 1. Take a look at the smart contractSimpleStorage.sol 2. . You can check it out in foldercontracts 3. .

This smart contract has:

- A variablestoredData
- to store a number
- A functionget()
- to return the number stored at variablestoredData
- A functionset()
- to change the number stored at variablestoredData
- Compile and migrate the smart contract. Note inside the development console we don't preface commands with truffle.

To make sure you're in the development console, the command prompt must betruffle(develop)>

compile Thecompile output should be similar to:

migrate And themigrate output should be similar to:

1. Running contract tests.

Our box also comes with the fileTestSimpleStorage.js for testing the smart contract. You can check it out in thetest folder.

Run this command in the development console:

test Thistest output should be similar to:

Note the command varies slightly if you're in or outside of the development console.

// inside the development console. test // outside the development console. truffle

test

Client side application¶

Our box has done a front end to interact with the smart contract, built using React app and Next JS.

1. In another terminal (i.e. not in the truffle develop prompt), go to theapp
2. directory and run the React app.

Do not close the other terminal, which is running the Truffle development console, because it is our Blockchain simulator.

If you close it and then open it again, you need to deploy / migrate the smart contract again too! cd

app npm run dev

Then go to your browser athttp://localhost:3000/

Smart contract changes must be manually recompiled and migrated! NOTE : This box is the starting point for the RSK tutorialUsing rsk-next-box .

Using RSK networks¶

Setup an account & get R-BTC¶

- Get an address, learning how works theaccount based RSK addresses
- .
- For the RSK Testnet, get tR-BTC fromour faucet
- .
- For the RSK Mainnet, get R-BTC fromon an exchange
- .

Setup the gas price¶

Gas is the internal pricing for running a transaction or contract. When you send tokens, interact with a contract, send R-BTC, or do anything else on the blockchain, you must pay for that computation. That payment is calculated as gas. In RSK, this is paid inR-BTC . TheminimumGasPrice is written in the block header by miners and establishes the minimum gas price that a transaction should have in order to be included in that block.

To update theminimumGasPrice in our project run this query using cURL:

Testnet

curl https://public-node.testnet.rsk.co/ -X POST -H "Content-Type: application/json"

\

--data '{"jsonrpc":"2.0","method":"eth_getBlockByNumber","params":["latest",false],"id":1}'

\

.minimum-gas-price-testnet.json Mainnet

curl https://public-node.rsk.co/ -X POST -H "Content-Type: application/json"

\

--data '{"jsonrpc":"2.0","method":"eth_getBlockByNumber","params":["latest",false],"id":1}'

\

.minimum-gas-price-mainnet.json This query saved the details of latest block to file .minimum-gas-price-testnet.json or .minimum-gas-price-mainnet.json, respectively.

In thetruffle-config.js , we are reading the parameterminimumGasPrice in each json file.

For more information about theGas andminimumGasPrice please go togas page .

Connect to RSK¶

1. Copy your mnemonic totruffle-config.js

//Put your mnemonic here, be careful not to deploy your mnemonic into production! const

mnemonic

=

'A_MNEMONIC' ; Please be aware that we are usingHDWalletProvider with RSK Networks derivations path: - RSK Mainnet dpath:m/44'/137'/0'/0 - RSK Testnet dpath:m/44'/37310'/0'/0

For more information checkRSKIP57 .

1. Run the development console for any RSK network.

Console for Testnet

truffle console --network testnet# Console for Mainnet truffle console --network mainnet 1. Migrate the smart contracts. We will do it running the below commands directly in the terminal, without using the truffle console now to show to you this alternative.

truffle migrate 1. Update Express JS component.

The component located in the fileweb3-util.js uses theweb3.js library to interact with the blockchain - writing code that reads and writes data from the blockchain with smart contracts.

Choose which network you would like to connect the server to RSK Network and update line 15 of the fileapp/utils/web3-util.js

Testnet

const

provider

=

new

Web3 . providers . HttpProvider ("https://public-node.testnet.rsk.co");

Mainnet

const

provider

=

new

Web3 . providers . HttpProvider ("https://public-node.rsk.co");

1. In a terminal, go to theapp
2. directory and run the React app.

cd

app npm run dev Go to your browser athttp://localhost:3000/

Note that when you are connected to an RSK network, you do not need to leave open the Truffle console, because the app is connected via a public node, directly to the network.

Next steps¶

- Go to tutorial

Go to the tutorialusing rsk-next-box to learn more about this project. We covered all the steps with more details, explanations, and images.

- Find more documentation

Check out theRSK developers portal .

- Do you have questions?

Ask inRSK chat .