Truffle NFT Box¶

Requirements¶

The NFT Box has the following requirements:

- Node.js
- 10.x or later
- NPM
- version 5.2 or later
- Windows, Linux or MacOS

Helpful, but optional: - AnInfura account and Project ID - AMetaMask account

Setup¶

Installation¶

First ensure you are in a new and empty directory.

1. Run theunbox
2. command vianpx
3. and skip to step 3. This will install all necessary dependencies. A Create-React-App is generated in theclient
4. directory.
5. npx
6. truffle
7. unbox
8. nft
9. -
10. box
11. Alternatively, you can install Truffle globally and run theunbox
12. command.
13. npm
14. install
15. -
16. g
17. truffle
18. truffle
19. unbox
20. nft
21. -
22. box
23. In the root directory install the required dependices. This will install a few things along with Open Zeppelin Contracts
24. check them outhere
25. for more info.javascript npm install @openzeppelin/contracts
26. Add the contructor arguments todeployer
27. function in2_deploy_contracts.js
28. . The URI for your NFT images must be passed as the first argument. Thename
29. andsymbol
30. of the token will be passed as the second and third respectively. ```javascript module.exports = function(deployer) { deployer.deploy(NFTCollection,"YOUR URI","TOKEN NAME","TOKEN SYMBOL"); }; ``````
31. In the root directory install the required dependices. This will install a few things along with Open Zeppelin Contracts
32. check them outhere
33. for more info.javascript npm install
34. Run the development console.
35. truffle
36. develop
37. Compile and migrate the smart contracts. Note inside the development console we don't preface commands withtruffle
38. . ```javascript compile migrate
39. In theclient

40. directory, we run the React app. Smart contract changes must be manually recompiled and migrated.
41. // in another terminal (i.e. not in the truffle develop prompt)
42. cd
43. client
44. npm
45. install
46. npm
47. run
48. start
49. After migrating your contracts head to theclient
50. directory and runnpm run start
51. to view the application in yourhttp://localhost:3000/
52. . Connect your wallet and mint your first NFT!
53. You can check the developer console to see the transaction has and if you deploy torinkeby
54. you will be able to see the NFT on their test-net site[here](#)
55. .
56. To build the application for production, use the build script. A production build will be in theclient/build
57. folder.
58. // ensure you are inside the client directory when running this
59. npm
60. run
61. build

Deployment¶

To deploy your contracts to a public network (such as a testnet or mainnet) there are two approaches. The first uses[Truffle Dashboard](#) which provides "an easy way to use your existing MetaMask wallet for your deployments". The second, requires copying your private key or mnemonic into your project so the deployment transactions can be signed prior to submission to the network.

Using Truffle Dashboard (recommended)¶

Truffle Dashboard ships with Truffle and can be started withtruffle dashboard . This in turn loads the dashboard at http://localhost:24012 and beyond that you'll just need to run your migration (truffle migrate). A more detailed guide to using Truffle Dashboard is available[here](#) .

Using the env File¶

You will need at least one mnemonic to use with the network. The.dotenv npm package has been installed for you, and you will need to create a.env file for storing your mnemonic and any other needed private information.

The.env file is ignored by git in this project, to help protect your private data. In general, it is good security practice to avoid committing information about your private keys to github. Thetruffle-config.js file expects aMNEMONIC value to exist in.env for running commands on each of these networks, as well as a defaultMNEMONIC for the Arbitrum network we will run locally.

If you are unfamiliar with using.env for managing your mnemonics and other keys, the basic steps for doing so are below:

1) Usetouch .env in the command line to create a.env file at the root of your project. 2) Open the.env file in your preferred IDE 3) Add the following, filling in your own Infura project key and mnemonics:

MNEMONIC="" INFURA_KEY="" RINKEBY_MNEMONIC="" MAINNET_MNEMONIC="" 4) As you develop your project, you can put any other sensitive information in this file. You can access it from other files withrequire('dotenv').config() and refer to the variable you need withprocess.env['] .

Support¶

Support for this box is available via the Truffle community available[here](#) .