Index

SvelteJS Truffle Box¶

A Truffle box usingSvelteJS andRollup .

This box contains everything you need to start building a smart-contract app.

Project Goal¶

To provide the simplest, cleanest seed for building an Ethereum dapp usingTruffle , with the minimum possible dependencies, meaning that beginners and pros a like have the most transparent possible method for developingEthereum contracts.

Truffle Box¶

Atruffle box is a seed project for building a truffle dapp.

Why Svelte?¶

Svelte was chosen as it is a rich, state-model based, ES6, component framework with very few dependencies, which is nothing more than html, javascript, and css. Once compiled via svelte, there are no clientside dependencies at all - simply vanilla JS.

Svelte is basically a simple DSL (domain specific language) for building a reactive, stateful, dependency-free web-application in pure javascript.

Additionally, theSvelte API is so simple and well-designed, you can learn the whole thing from scratch in less than an hour!

Why Rollup?¶

Originally this project used ParcelJS but sadly Parcel's support for Svelte is currently broken, and has been for a while. I've switched to RollupJS in order to upgrade to Svelte 3.

Currently, we load web3 from UNPKG, since it appears to be borderline impossible to bundle successfully. If anybody wants to open a PR to bundle Web3, it would be greatly appreciated.

Setting up¶

1. Install truffle and an ethereum client. For local development, try Ethereum TestRPC.
2. npm
3. install
4. -
5. g
6. truffle
7. // Version 3.0.5+ required.
8. npm
9. install
10. -
11. g
12. ganache
13. -
14. cli
15. // Or the ganache GUI will work too.
16. Download box.
17. truffle
18. unbox
19. antony
20. /
21. svelte
22. -
23. box
24. Run an Ethereum RPC. For simplicity and development we will be using Ethereum TestRPC.
25. ganache
26. -
27. cli

28. Compile and migrate the contracts after authenticating your account on the blockchain (i.e. restoring from seed in MetaMask).
29. truffle
30. compile
31. truffle
32. migrate

You're ready to go!

Usage¶

Components are insrc/components/*.html . Everything else is in the usual placeaccording to the docs

Run the testrpc so that you have a blockchain to work with, and deploy your contracts:

testrpc truffle deploy Log in to metamask by importing the HD Wallet that testrpc gave you, and do the same for one of the accounts by entering its private key. Then, run the dev task to have the code updated in realtime as you develop:

truffle compile npm run dev

Publishing¶

To produce your production dApp, run the build task:

npm run build This will publish your completed dApp to the folder./dist

Testing¶

Testing works much the same way as it does in any web-application, with an additionaltruffle test command for testing smart contracts.

Be sure you've compiled your contracts before running the tests, or you'll get file not found errors.

```
javascript npm run test:unit // for dApp tests npm run test:contract // for contract tests
```

Releasing¶

To build the application for production, use the build command. A production build will be in the./dist folder.

```
javascript npm run build
```

FAQ¶

- Why is there both a truffle.js file and a truffle-config.js file?
- Truffle requires the truffle.js file be named truffle-config on Windows machines. Feel free to delete the file that doesn't correspond to your platform.