# Etherplate¶

### What is this?¶

This is an example project showing how you can hook up your Ethereum Non-Fungible Tokens (NFTs, ERC721, similar to CryptoKitties) contract in a DApp. It demos web3 events, and is highly opinionated in that it uses Redux, React, React Router, and Bulma. You can rip out of any these or replace them with your favourites (ie. Skeleton.css instead of Bulma, etc.).

Etherplate uses OpenZeppelin's fantastic community-audited contracts as a base to implement the ERC721 standard.

### View Demo¶

View Demo on Netlify (Uses Ropsten testnet)

# Setup¶

### Requires NPM & Direnv¶

Homebrew on Mac OSX:

brew install node npm direnv

Apt on Linux:

apt-get install node npm direnv

### Install truffle globally:¶

npm install truffle -g

### Install the local NPM packages:¶

npm install

### Environment Variables¶

1. cp .envrc.example .envrc
2. Enter your own twelve random words in the .envrc.
3. Also, we'll leverage Infura's Ethereum Ropsten testnet node, so make sure to set up an account and paste your private key in your .envrc.
4. Usedirenv allow
5. to export the env vars into your current terminal shell.

### Run the Ganache CLI (local EVM)¶

Start up the local Ethereum test node with:

npm run ganache

(You may need to loosen up the permissions on the file, trychmod 755 scripts/ganache.sh )

### Compile the Solidity code¶

Once Ganache is running, in another terminal window compile and migrate the contracts:

truffle compile

This will deploy the compiled contracts to the network (tip: use --network=ropsten to deploy to Ethereum's Ropsten Testnet, --network=rinkeby for rinkeby, etc)

truffle migrate

# Run the Project¶

Make sure the truffle contracts are compiled and migrated, and ganache is running.

Start the Webpack dev server.

npm run dev

Your server should now be running at http://127.0.0.1:8080

### truffle.js & truffle-config.js¶

Why is there both a truffle and truffle-config file?

- On Windows, truffle-config.js is required. You can safely delete the one you don't need (ie on Mac/Linux you can delete truffle-config.js)

# Building the Project¶

npm run build

Note: Currently we are manually migrating contracts against the Ropsten & Rinkeby testnets, and checking the build into the repo. This is less than ideal. It would be better to use a build script such as thenetlify-build.sh file and compile contracts on the server.

# Running the Tests¶

For the Solidity contract's truffle test suite:

truffle test

To run the DApp test suite (React components, etc):

npm test

## Toast Messages¶

Examples of a bunch of different looking toast messages to show on an error message, success, info, etc.:

toastr.light('test', 'message', { icon: 'info', status: 'info' }) toastr.light('test', 'message', { icon: 'success', status: 'success' }) toastr.light('test', 'message', { icon: 'warning', status: 'warning' })

toastr.success('test', 'message') toastr.info('test', 'message') toastr.warning('test', 'message') toastr.error('test', 'message')

## TODO:¶

- Test w/ Web3-integrated browsers such as Trust
- Set up a basic server to respond to tokenURI requests and store the tokenURI in the contract (buyToken()
- function)
- Refactor React components to have both presentation and container components
- Add deepFreeze() to test immutability of Redux reducers
- Fix issues where we should be unsubscribing / canceling server requests incomponentWillUnmount()
- Rename all services to have-service
- in the filename
- Rename all components to use standard React naming scheme:EtherscanButton.jsx
- instead ofetherscan-button.jsx
- Follow a standard ES6 export pattern (use TokenListItem component code as an example)

## Nice-to-haves:¶

- Offline.js or react-detect-offline to let users know when their network connection is dead
- Find a way to preventtruffle test
- from recompiling the contracts each time it is run
- Store transactionHash in localStorage and call info on it after page refreshes if it isn't instore.getState().tokens
- pool
- Remove the build directory from the repo, build server-side on each deploy and possibly use truffle-migrate-off-chain (https://github.com/asselstine/truffle-migrate-off-chain)
- Test the happy path of filling out the form and purchasing a token via (jest/enzyme)

- Getcircleci
- branch up and running, put a badge on the README for test runs
- Demo how ERC721 expects you to store data (such as the JSON response when the tokenURI is requested) as per https://eips.ethereum.org/EIPS/eip-721 (For instance, OpenSea has a server which takes a contract address and tokenID, which then does a GET request to the tokenURI to pull more info (as JSON) about the token (images, name, etc), for example: https://opensea-api.herokuapp.com/events/?asset_contract_address=0x06012c8cf97bead5deae237070f9587f8e7a266d&token_id=389343)

# Done:¶

- ~~Make into a truffle box and submit to trufflesuite~~
- ~~BUG: Purchase History only showing some purchases while My Tokens shows more ... ?~~
- ~~Show token ID / transaction ID on purchase history and Tokens#show page~~
- ~~Make sure 'Purchase History' page works~~
- ~~Implement Redux for web3 events~~
- ~~Make all React prop types required (isRequired) and provide defaultProps for those that are not~~
- ~~Convert all css to scss~~
- ~~Improve mobile styling / media query support~~
- ~~Deploy to Netlify & Ropsten, use Infura~~
- ~~Use a local web3 (1.0.0.beta?) instead of the current MetaMask/browser's web3 instance (which is deprecated)~~
- ~~Fix getting duplicate entries when Ropsten returns the BoughtToken event (active subscriber listening for events in browser)~~
- ~~New token updated from transaction receipt event is not being added to state properly in realtime~~
- ~~Get DApp tests working again~~
- ~~On successful purchase, show a message about the new purchase and how it needs to be confirmed by the network, and redirect to show the now confirming token on Purchase History or My Tokens page~~
- ~~Instead of 'Loading ...' should say confirming (show # of confirmations?)~~
- ~~Race condition: sometimes we do not have the list of accounts from MetaMask on time when page loads (google for onPageLoad code)~~
- ~~Mock out a web3 object in the specs~~
- ~~Clean up JS in Header.jsx for controlling Bulma link active states~~
- ~~Show account balance, network and account address / avatar~~
- ~~Finish upgrade path by removingoldNfTokenFactory
- and in turn get.events.BoughtToken()
- working again: (Error: The current provider doesn't support subscriptions: MetamaskInpageProvider)~~
- ~~Add a price for each token (say 0.03 eth)~~
- ~~Toast message to say token purchase was broadcast~~
- ~~Link to view on on etherscan~~
- ~~If the user switches their MetaMask account or logs out of MetaMask, need to refresh the page or stop/restart event listeners with new wallet address (Long Polling?)~~

### Gratitude¶

Big thanks to all of the fantastic open source developers who have made this technology actual, and to Brendan Asselstine for helping kickstart my development with blockchain technologies.

### Etherplate Wordmark¶

The Etherplate Wordmark is set in Sign Painter: https://typekit.com/fonts/signpainter