Truffle Suite

Update: While this tutorial was originally written for Truffle 3, we have updated it to be fully compatible with newer versions of Truffle. We have kept some of the language unaltered, except where necessary.

Truffle 3 is out and switched to less opinionated build process that allows any build pipeline to be plugged in. We're going to take advantage of that feature today, and build a frontend app with a custom pipeline.

Intended audience¶

This is written for those familiar with Truffle and Ethereum, who want a sense of how to structure a frontend application for easy development and testing. This example useswebpack , but familiarity with other build tools is enough background.

What does a build process need to do in Truffle?¶

In general, simple: Turn all higher level code (like ES6, SASS, JSX, templates, etc.) into vanilla JavaScript/HTML/CSS artifacts, and then move those artifacts into thebuild/ folder alongside our contract artifacts.

What this tutorial does¶

Useswebpack to compile the application's frontend code and move the artifacts into thebuild/ folder.

Getting started¶

Once you have Truffle installed, runtruffle unbox webpack in anempty directory to pull down theTruffle Box for this tutorial. If you're familiar with theMetaCoin Truffle Box), you'll notice your old friendsMetacoin.sol andConvertLib.sol are there. But now, runningtruffle build does this:

Error building:

No build configuration specified. Can' t build.

Build failed. See above. That's ok! Truffle is getting out of your way and letting you control the build process. Our new process is specified inwebpack.config.js . More on this later.

Compile, migrate, and ...¶

In order to interact with contracts, we need them deployed on a network! The default network is configured intruffle.js :

networks :

{

development :

{

host :

'127.0.0.1' ,

port :

7545 ,

network_id :

'*'

// Match any network id

} } This is the default setting forGanache , though you can change this to use any connection you'd like.

Note: Read more aboutsetting up Ganache. You can also useTruffle Develop, which launches a personal blockchain for testing and is built directly into Truffle, no separate installation required.

Let's get the contracts on the network:

First runtruffle compile . This will compile the.sol contracts into.json artifacts (specified in the@truffle/contract library). They will appear inbuild/contracts/*.json . Now we can include contracts in our app with a simpleimport orrequire statement:

// Import our contract artifacts and turn them into usable abstractions. import

metacoin_artifacts

from

'../../build/contracts/MetaCoin.json' Next, make sure you have a blockchain running, and then runtruffle migrate . This will deploy the contracts onto the default network running at127.0.0.1:7545 .

... (webpack) build¶

All that's left is to use webpack to compile the app and place it in thebuild/ folder. A simplenpm run dev and we're done! This will build the app and serve it onhttp://127.0.0.1:8080 . Relevant configs here:

// file: package.json ...

"scripts" :

{

"lint" :

"eslint ./" ,

"build" :

"webpack" ,

"dev" :

"webpack-dev-server"

}, ... // file: webpack.config.js ... entry :

'./app/javascripts/app.js' , output :

{

path :

path . resolve (__dirname ,

'build'),

filename :

'app.js' }, plugins :

[

// Copy our app's index.html to the build folder.

new

CopyWebpackPlugin ([

{

from :

'./app/index.html' ,

to :

'index.html'

}

])], ... You can find more information on webpack concepts onwebpack's website . Notice we didn'thave to use webpack here, but Truffle comes with a handy webpack demo that gets us started quickly. Wecould replace the webpack config with aGruntfile , for instance, and use Grunt instead. Truffle don't care no mo'.

The app¶

As previously mentioned, the commandnpm run dev builds the app and serves it with a local web server. (This is equivalent to runningnpm run build and thentruffle serve , but we've done it in one step for convenience.) To see it, navigate tohttp://127.0.0.1:8080 . You should see:

That's it, all done! Happy Truffling!

Extra: Webpack + Docker + Travis CI¶

If you want to see how to deploy a frontend app using Docker and Travis build pipeline, check outmy frontend example for added goodness.