2019-03-28 工作记录-next框架使用

本文记录了第一次安装使用next框架。

Next是一个路由框架，官网：<https://github.com/zeit/next.js>

第一部分：

1 进入工程目录，安装相关包：

全部命令指定版本安装，保证通过。

npm init

npm install --save solc@0.4.25

npm install --save web3@1.0.0-beta.36

npm install --save [mocha@5.2.0](mailto:mocha@5.2.0)

npm install --save ganache-cli@6.1.8

npm install --save [fs-extra@7.0.1](mailto:fs-extra@7.0.1)

npm install --save react

npm install --save react-dom

npm install --save next

npm install --save [truffle-hdwallet-provider@1.0.5](mailto:truffle-hdwallet-provider@1.0.5)

npm install --save [semantic-ui-react@0.86.0](mailto:semantic-ui-react@0.86.0)

npm install --save [semantic-ui-css@^2.4.1](mailto:semantic-ui-css@%5e2.4.1)

npm install --save next-routes

可以一次性安装全部：

D:\code-nodejs\fullstackDapp>npm install --save next react react-dom

也可以分开安装，安装next时比较慢，在公司里用上代理才安装成功了，在家中用不用代理都没能安装成功。

D:\code-nodejs\fullstackDapp>npm install --save react

D:\code-nodejs\fullstackDapp>npm install --save react-dom

D:\code-nodejs\fullstackDapp>npm install --save next

npm WARN registry Using stale data from https://registry.npmjs.org/ because the host is inaccessible -- are you offline?

npm WARN registry Using stale package data from https://registry.npmjs.org/ due to a request error during revalidation.

npm WARN unistore@3.2.1 requires a peer of preact@\* but none is installed. You must install peer dependencies yourself.

npm WARN fullstackdapp@1.0.0 No description

npm WARN fullstackdapp@1.0.0 No repository field.

npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.7 (node\_modules\fsevents):

npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.7: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

+ next@8.0.3

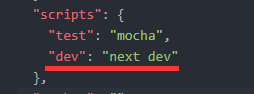
added 350 packages from 185 contributors and audited 118118 packages in 24101.706s

found 1 moderate severity vulnerability

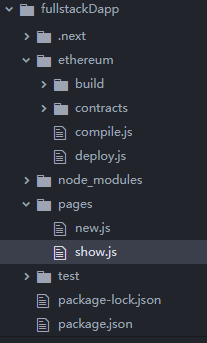
run `npm audit fix` to fix them, or `npm audit` for details

D:\code-nodejs\fullstackDapp>

2 修改package.json: 把dev更改成next dev，这样运行dev时会自动启动next dev。



3 编写网页。工程中新建pages目录，里面建立两个文件：



new.js：

import React from 'react';

export default ()=>{

return <h1>hi Tom!</h1>;

}

show.js:

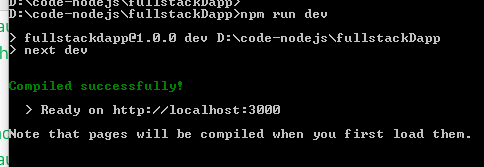
import React from 'react';

export default ()=>{

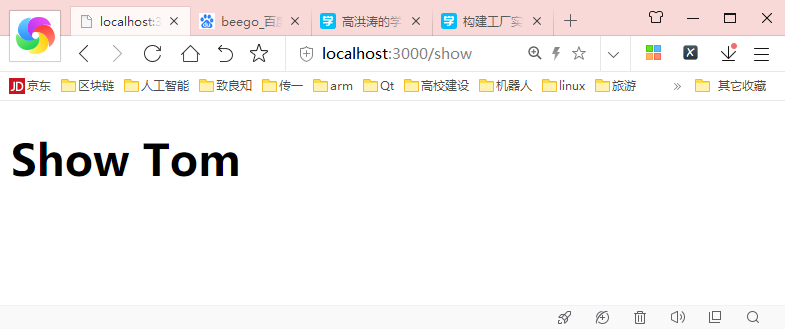
return <h1>Show Tom</h1>;

}

4 运行dev：在工程目录下执行 npm run dev, 成功启动后显示本地访问地址<http://localhost:3000>。

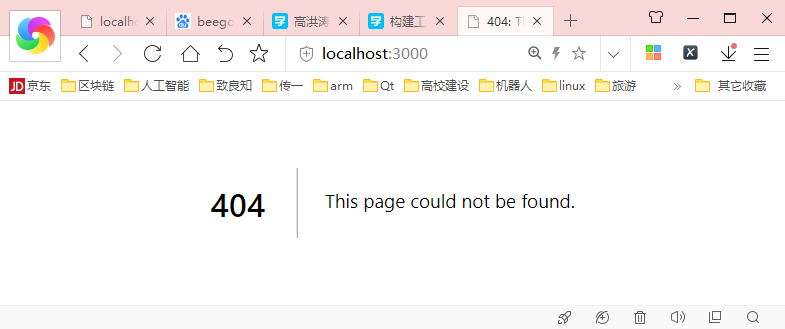


5 打开浏览器，输入<http://localhost:3000/new>,或者<http://localhost:3000/show>,就显示之前两个文件的内容



第二部分：

上述内容如果打开http://localhost:3000，就会显示404错误， 如何让next显示我们定义的首页呢？



1 在创建pages/index.js文件，内容如下：

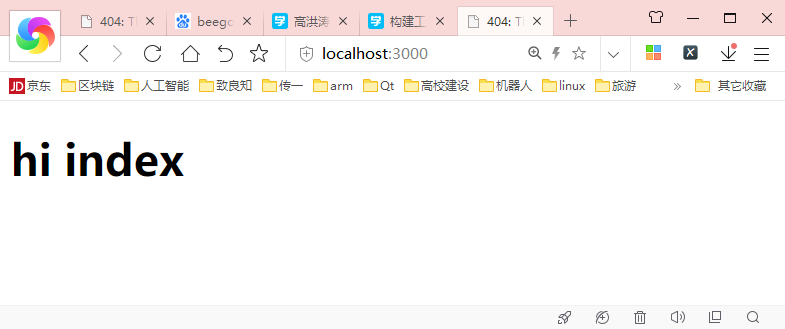
import React from 'react';

export default ()=>{

return <h1>hi index</h1>;

}

当访问<http://localhost:3000，next>会自动调用index.js,显示内容。



第三部分 真正实现工厂web3访问的首页。使用工厂实例

1. 创建ethereum/web3.js。 注意中间的if写法，避免next框架下找不到window对象。

import Web3 from 'web3';

var web3;

if(typeof window !='undefined' && window.web3 != 'undefined'){

web3 = new Web3(window.web3.currentProvider);

}else{

const provider = new Web3.providers.HttpProvider(

'https://ropsten.infura.io/v3/5391d9a764884afbb4bbd7d0994e68ec'

);

web3 = new Web3(provider);

}

export default web3;

1. 创建ethereum/factory.js。注意创建工厂实例的地址参数，必须是执行deploy后得到的已经部署的工厂合约地址。

import web3 from './web3';

import CompileFactory from './build/CampaignFactory.json';

//得到工厂合约对象

const instance = new web3.eth.Contract(

JSON.parse(CompileFactory.interface),

'0xcfb6977AE2302Ba3B350e24AefbE249B4c468b10'

);

export default instance;

1. 修改首页文件 pages/index.js。

import React, {Component} from 'react';

import factory from '../ethereum/factory';

class CampaignIndex extends Component{

async componentDidMount(){

const campaign = factory.methods.getDeployedCampaign().call();

console.log(campaign);

}

render(){

return <div>hello index</div>;

}

}

export default CampaignIndex;

1. 使用安装了web3扩展插件的google浏览器访问<http://localhost:3000>，运行正常。如果用360浏览器会运行错误。

