2019-03-29工作记录-动态页面路由跳转

本文记录了如何配置next服务器，如何跳转到动态生成的页面

第一部分 自定义next server

1 在工程目录下安装 next-routes库。

具体方法：https://github.com/fridays/next-routes

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

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-routes@1.4.2

added 2 packages from 1 contributor and audited 118142 packages in 34.313s

found 1 moderate severity vulnerability

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

D:\code-nodejs\fullstackDapp>

2 工程中创建routes.js, 实现了动态路由功能。

内容编写：

//require('next-routes') 是导入的函数，添加上（）才是调用函数，生成对象routes

const routes = require('next-routes')();

module.exports = routes;

注意这里的 exports单词，千万别少写一个s，会导致server.js运行失败， 服务器启动失败。

3 工程中创建server.js, 通过require('./routes')导入toutes.js, 内容编写：

const next = require('next')

const routes = require('./routes')

const app = next({dev: process.env.NODE\_ENV !== 'production'})

const handler = routes.getRequestHandler(app)

// Without express

const {createServer} = require('http')

app.prepare().then(() => {

createServer(handler).listen(3000)

})

4 修改package.json，配置为启动server.js

"scripts": {

"test": "mocha",

"dev": "node server.js"

},

5 修改pages/campaign/new.js，当完成创建众筹活动后返回首页

导入 Router：

import {Router} from '../../routes';



6 重新启动node：

D:\code-nodejs\fullstackDapp>npm run dev

7 实际运行结果正常，从<http://localhost:3000/campaigns/new>变成了<http://localhost:3000/>。

第二部分 添加链接，实现首页和二级页面的跳转

目标： 标题行的首页、众筹、+都能点击跳转，众筹活动的“查看众筹”点击调到动态页面（url链接是一串地址）,

1 实现标题行的首页、众筹、+都能点击跳转。

修改components/Header.js, 导入Link标签，<Link route='/'>这个就是跳到首页。

import React from 'react';

import { Menu } from 'semantic-ui-react'

import { Link } from '../routes';

export default props =>{

return (

<Menu style={{marginTop:'15px'}}>

<Menu.Item>

<Link route='/'>

首页

</Link>

</Menu.Item>

<Menu.Menu position='right'>

<Menu.Item>

<Link route='/'>

众筹

</Link>

</Menu.Item>

<Menu.Item>

<Link route='/campaigns/new'>

+

</Link>

</Menu.Item>

</Menu.Menu>

</Menu>

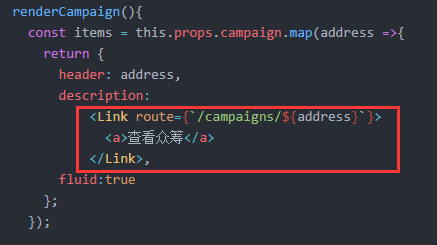
);

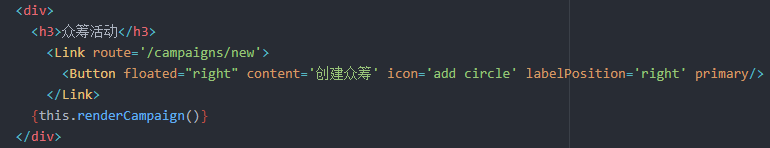
}

2 修改首页pages/index.js，关键代码：

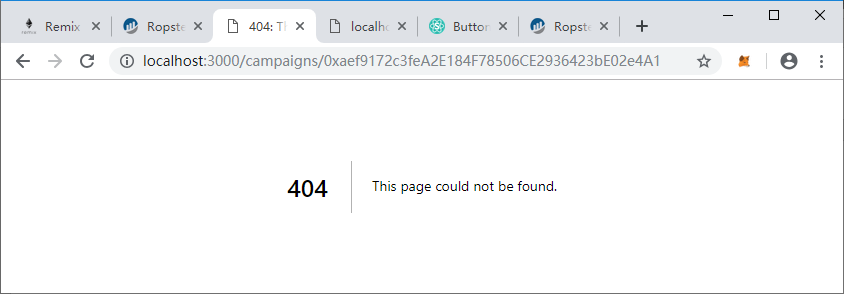
{`/campaigns/${address}`} 这个代码格式实现了字符串+变量组合。

import { Link } from '../routes';





现在点击“查看众筹”后切换到地址界面，显示404， 这是因为该页面还不存在。接下来我们将设计这个动态页面。



第三部分 编写最基本的动态跳转页面。

1 创建文件pages/compaigns/show.js，内容：

import React from 'react';

class CampaignShow extends React.Component{

render(){

return (

<Layout>

<h1>Hello, Tom</h1>

</Layout>

);

}

}

export default CampaignShow;

2 在routes.js中添加路由规则：

//require('next-routes') 是导入的函数，添加上（）才是调用函数，生成对象routes

const routes = require('next-routes')();

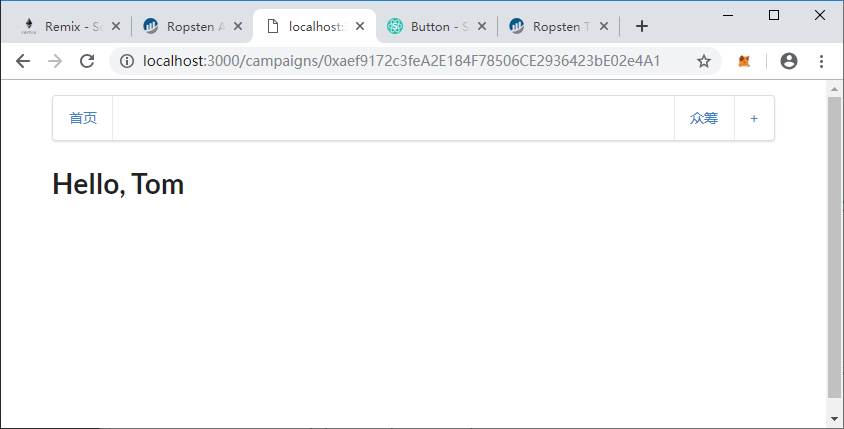
routes.add('/campaigns/new','/campaigns/new')

.add('/campaigns/:address', '/campaigns/show');

//注意这里的 exports单词，千万别少写一个s，会导致server.js运行失败， 服务器启动失败。

module.exports = routes;

3 现在点击“查看众筹”后切换到地址界面。



第4部分 把跳转的地址作为参数传递给show.js

更改show.js, 添加：

//这是生命周期函数，参数props由route转换时传递。

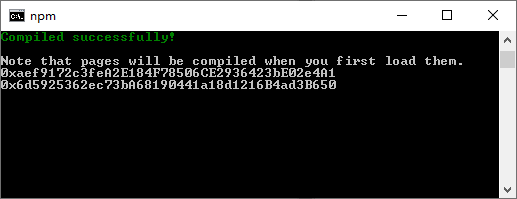
static async getInitialProps(props){

console.log(props.query.address);

return {};

};

这会在命令行终端打印出来页面地址：秘诀是跳转到地址界面后点击工具栏的刷新图标，不然打印不出来。



第5 部分 ，消除浏览器控制台的警告，为link标签添加<a>标签

components/Header.js:

index.js 类似下面的link。

