link: null
title: 珠峰架构师成长计划
description: 放在服务器进行就是服务器渲染，放在浏览器进行就是浏览器渲染
keywords: null
author: null
date: null
publisher: 珠峰架构师成长计划
stats: paragraph=64 sentences=154, words=1164

什么是服务端渲染(Server Side Render)

放在服务器进行就是服务器渲染，放在浏览器进行就是浏览器渲染

- 客户端渲染不利于 SEO 搜索引擎优化
- 服务端渲染是可以被爬虫抓取到的，客户端异步渲染是很难被爬虫抓取到的
- SSR直接将HTML字符串传递给浏览器。大大加快了首屏加载时间。
- SSR占用更多的CPU和内存资源
- 一些常用的浏览器API可能无法正常使用
- 在vue中只支持beforeCreate和created两个生命周期

开始vue-ssr之旅

```
yarn add vue-server-renderer vue
yarn add express
```

createRenderer,创建一个渲染函数 renderToString, 渲染出一个字符串

```
let Vue = require('vue');
let render = require('vue-server-renderer');
let vm = new Vue({
    data:{
        msg:'jw',
    },
    template:('<h1>{{msg}}</h1>')
})
let express = require('express');
let app = express();
app.get('/',async (req,res)=>{
    let code = await render.createRenderer().renderToString(vm);
    res.send(`
        <!DOCTYPE html>
        <html lang="en">
        <head>
            <meta charset="UTF-8">
            <meta name="viewport" content="width=device-width, initial-scale=1.0">
            <meta http-equiv="X-UA-Compatible" content="ie=edge">
            <title>Document</title>
        </head>
        <body>
            ${code}
        </body>
        </html>
    `)
});
app.listen(3000);
```

采用模板渲染

```
<!DOCTYPE html>
<html lang="en">
  <head><title>Hello</title></head>
  <body>
    <!--vue-ssr-outlet-->
  </body>
</html>
```

传入template 替换掉注释标签

```
llet Vue = require('vue');
let fs = require('fs');
let template = fs.readFileSync('template.html','utf8')
let render = require('vue-server-renderer');
let vm = new Vue({
    data:{
        msg:'jw',
    },
    template:('<h1>{{msg}}</h1>')
})
let express = require('express');
let app = express();
app.get('/',async (req,res)=>{
    let code = await render.createRenderer({
        template
    }).renderToString(vm);
    res.send(code);
});
app.listen(3000);
```

目录创建

```
&#x251C;&#x2500;&#x2500; config
&#x2502;   &#x251C;&#x2500;&#x2500; webpack.base.js
&#x2502;   &#x251C;&#x2500;&#x2500; webpack.client.js
&#x2502;   &#x2514;&#x2500;&#x2500; webpack.server.js
&#x251C;&#x2500;&#x2500; dist
&#x2502;   &#x251C;&#x2500;&#x2500; client.bundle.js
&#x2502;   &#x251C;&#x2500;&#x2500; index.html
&#x2502;   &#x251C;&#x2500;&#x2500; index.ssr.html
&#x2502;   &#x251C;&#x2500;&#x2500; server.bundle.js
&#x2502;   &#x251C;&#x2500;&#x2500; vue-ssr-client-manifest.json
&#x2502;   &#x2514;&#x2500;&#x2500; vue-ssr-server-bundle.json
&#x251C;&#x2500;&#x2500; package.json
&#x251C;&#x2500;&#x2500; public
&#x2502;   &#x251C;&#x2500;&#x2500; index.html
&#x2502;   &#x2514;&#x2500;&#x2500; index.ssr.html
&#x251C;&#x2500;&#x2500; server.js
&#x251C;&#x2500;&#x2500; src
&#x2502;   &#x251C;&#x2500;&#x2500; App.vue
&#x2502;   &#x251C;&#x2500;&#x2500; components
&#x2502;   &#x2502;   &#x251C;&#x2500;&#x2500; Bar.vue
&#x2502;   &#x2502;   &#x2514;&#x2500;&#x2500; Foo.vue
&#x2502;   &#x251C;&#x2500;&#x2500; entry-client.js
&#x2502;   &#x251C;&#x2500;&#x2500; entry-server.js
&#x2502;   &#x251C;&#x2500;&#x2500; main.js
&#x2502;   &#x251C;&#x2500;&#x2500; router.js
&#x2502;   &#x2514;&#x2500;&#x2500; store.js
&#x251C;&#x2500;&#x2500; webpack.config.js
```

通过webpack实现编译vue项目

安装插件

```
yarn add webpack webpack-cli webpack-dev-server vue-loader vue-style-loader css-loader html-webpack-plugin @babel/core @babel/preset-env babel-loader vue-template-compiler webpack-merge
```

```
let path = require('path');
let HtmlWebpackPlugin = require('html-webpack-plugin');
let VueLoaderPlugin = require('vue-loader/lib/plugin')
module.exports = {
    entry:'./src/main.js',
    output:{
        filename:'bundle.js',
        path:path.resolve(__dirname)
    },
    module:{
        rules:[
            {test:/\.css/,use:['vue-style-loader','css-loader']},
            {
                test:/\.js/,
                use:{
                    loader:'babel-loader',
                    options:{
                        presets:['@babel/preset-env']
                     },
                },
                exclude:/node_modules/,
            },
            {test:/\.vue/,use:'vue-loader'}
        ]
    },
    plugins:[
        new VueLoaderPlugin(),
        new HtmlWebpackPlugin({
            template:'./src/index.html'
        })
    ]
}
```

配置客户端大包和服务端打包

- webpack.base.js

```
let path = require('path');
let VueLoaderPlugin = require('vue-loader/lib/plugin')
module.exports = {
  entry:'./src/main.js',
  output:{
      filename:'[name].bundle.js',
      path:path.resolve(__dirname)
  },
  module:{
      rules:[
          {test:/\.css/,use:['vue-style-loader','css-loader']},
          {
              test:/\.js/,
              use:{
                  loader:'babel-loader',
                  options:{
                      presets:['@babel/preset-env']
                   },
              },
              exclude:/node_modules/,
          },
          {test:/\.vue/,use:'vue-loader'}
      ]
  },
  plugins:[
      new VueLoaderPlugin()
  ]
}
```

- webpack.client.js

```
let path = require('path');
let HtmlWebpackPlugin = require('html-webpack-plugin');
let merge  = require('webpack-merge');
let base = require('./webpack.base');
module.exports = merge(base, {
    entry:{
        'client':path.resolve(__dirname,'../src/client.js')
    },
    plugins:[
        new HtmlWebpackPlugin({
            template:'./src/index.html'
        })
    ],
});
```

- webpack.server.js

```
let path = require('path');
let HtmlWebpackPlugin = require('html-webpack-plugin');
let merge  = require('webpack-merge');
let base = require('./webpack.base');

module.exports =merge(base,{
    target:'node', // &#x6253;&#x5305;&#x7C7B;&#x578B;node
    entry:{
        server:path.resolve(__dirname,'../src/server.js')
    },
    output:{
        libraryTarget: 'commonjs2' // &#x4EE5;commonjs&#x89C4;&#x8303;&#x5BFC;&#x51FA;
    },
    plugins:[
        new HtmlWebpackPlugin({
            template:'./src/index.html',
            excludeChunks:['server']
        })
    ]
})
```

服务端配置

在App.vue上增加id="app"可以保证元素被正常激活

```
let express = require('express');
let vueServerRenderer = require('vue-server-renderer');
let path = require('path')
let app = express();
let fs = require('fs');

let render = vueServerRenderer.createBundleRenderer(fs.readFileSync('./dist/server.bundle.js','utf8'),{
    template:fs.readFileSync('./dist/index.ssr.html','utf8')
});

app.get('/',(req,res)=>{
    render.renderToString((err,html)=>{
        res.send(html);
    })
});

app.use(express.static(path.join(__dirname,'dist')))
app.listen(3000,()=>{
    console.log('server start 3000')
});
```

集成路由

```
import Vue from 'vue';
import VueRouter from 'vue-router';
Vue.use(VueRouter);
import Bar from './components/Bar.vue';
import Foo from './components/Foo.vue';
export default ()=>{
    let router = new VueRouter({
        mode:'history',
        routes:[
            {
                path:'/', component:Bar,
            },
            {
                path:'/foo',component:Foo
            }
        ]
    });
    return router
}
```

配置入口文件

```
import createRouter from './router';
export default ()=>{
    let router = createRouter(); // &#x589E;&#x52A0;&#x8DEF;&#x7531;
    let app = new Vue({
        router,
        render:(h)=>h(App);
    })
    return {app,router}
}
```

配置组件信息

```
<template>
    <div id="app">
        <router-link to="/"> bar</router-link>
        <router-link to="/foo"> foo</router-link>
        <router-view></router-view>
        {{$store.state.username}}
    </div>
</template>
```

防止刷新页面不存在 #

```
app.get('*',(req,res)=>{
    render.renderToString({url:req.url},(err,html)=>{
        res.send(html);
    })
});
```

vuex配置

```
import Vue from 'vue';
import Vuex from 'vuex';
Vue.use(Vuex);

export default ()=>{
    let store = new Vuex.Store({
        state:{
            username:'zf'
        },
        mutations:{
            set_user(state){
                state.username = 'hello;
            }
        },
        actions:{
            set_user({commit}){
                return new Promise((resolve,reject)=>{
                    setTimeout(() => {
                        commit('set_user');
                        resolve();
                    }, 1000);
                })
            }
        }
    });
    return store
}

// &#x5F15;&#x7528;vuex
import createRouter from './router';
import createStore from './store'
export default ()=>{
    let router = createRouter();
    let store = createStore();
    let app = new Vue({
        router,
        store,
        render:(h)=>h(App)
    })
    return {app,router,store}
}
```

在后端更新vuex #

```
import createApp from './main';
export default (context)=>{
    return new Promise((resolve)=>{
        let {app,router,store} = createApp();
        router.push(context.url); // &#x9ED8;&#x8BA4;&#x8BBF;&#x95EE;&#x5230;/a&#x5C31;&#x8DF3;&#x8F6C;&#x5230;/a
        router.onReady(()=>{
            let matchs = router.getMatchedComponents(); // &#x83B7;&#x53D6;&#x8DEF;&#x7531;&#x5339;&#x914D;&#x5230;&#x7684;&#x7EC4;&#x4EF6;

            Promise.all(matchs.map(component=>{
                if(component.asyncData){
                    return component.asyncData(store);
                }
            })).then(()=>{
                context.state = store.state; // &#x5C06;store&#x6302;&#x8F7D;&#x5728;window.__INITIAL_STATE__
                resolve(app);

            });
        })
    })
}
```

在浏览器运行时替换store #

```
// &#x5728;&#x6D4F;&#x89C8;&#x5668;&#x8FD0;&#x884C;&#x4EE3;&#x7801;
if(typeof window !== 'undefined' && window.__INITIAL_STATE__){
    store.replaceState(window.__INITIAL_STATE__);
}
```

通过json配置createBundleRenderer方法

实现热更新,自动增加preload和prefetch,以及可以使用sourceMap

```
const VueSSRClientPlugin = require('vue-server-renderer/client-plugin');
const VueSSRServerPlugin = require('vue-server-renderer/server-plugin');
const nodeExternals = require('webpack-node-externals');
let template = fs.readFileSync('./dist/index.ssr.html','utf8');
let manifest = require('./dist/vue-ssr-client-manifest.json');
let bundle = require('./dist/vue-ssr-server-bundle.json');

let render = vueServerRenderer.createBundleRenderer(bundle,{
    template,
    clientManifest:manifest
})
```

自己更方便的管理vue的meta标签

```
import Vue from 'vue'
import Meta from 'vue-meta';

Vue.use(Meta);

// &#x5C06;meta&#x6302;&#x8F7D;&#x5728;&#x4E0A;&#x4E0B;&#x6587;&#x4E2D;
const meta = app.$meta()
context.meta = meta;

// &#x7EC4;&#x4EF6;&#x4E2D;&#x914D;&#x7F6E;
metaInfo: {
    title:'&#x563F;&#x563F;'
}

// &#x5728;&#x6A21;&#x677F;&#x4E2D;&#x53D6;&#x503C;
{{{ meta.inject().title.text() }}}
```