一. SSR

> https://ssr.vuejs.org/zh/guide/#%E5%AE%89%E8%A3%85

```
// 第 1 步：创建一个 Vue 实例
const Vue = require('vue')
const app = new Vue({
 template: `<div>Hello World</div>`
})

// 第 2 步：创建一个 renderer
const renderer = require('vue-server-renderer').createRenderer()

// 第 3 步：将 Vue 实例渲染为 HTML
renderer.renderToString(app).then(html => {
 console.log(html)
}).catch(err => {
 console.error(err)
})

```

二. Nuxt.js

1. Nuxt.js介绍与安装

> https://zh.nuxtjs.org/guide

npx create-nuxt-app <项目名>

服务端渲染， 解决首屏加载速度， 和 seo问题

```
//错误 HTMLElement is not define

修改nuxt.config.js 中plugins
plugins: [
 // '@/plugins/element-ui',
 { src: '@/plugins/element-ui', ssr: false}
 ]
 //不要复制 ， 格式有问题
```

2. Nuxt.js的配置

> https://zh.nuxtjs.org/guide/configuration

3. 路由

Nuxt.js 依据 pages 目录结构自动生成 vue-router 模块的路由配置。

(1) 要在页面之间使用路由，我们建议使用<nuxt-link> 标签。 支持activeClass,tag

(2)

```
pages/
--| user/
-----| index.vue
-----| one.vue
--| index.vue


那么，Nuxt.js 自动生成的路由配置如下：

 router: {
  routes: [
  {
  name: 'index',
  path: '/',
  component: 'pages/index.vue'
  },
  {
  name: 'user',
  path: '/user',
  component: 'pages/user/index.vue'
  },
  {
  name: 'user-one',
  path: '/user/one',
  component: 'pages/user/one.vue'
  }
  ]
 }
```

(3)嵌套路由

创建内嵌子路由，你需要添加一个 Vue 文件，同时添加一个与该文件同名的目录用来存放子视图组件。

Warning: 别忘了在父组件(.vue文件) 内增加 <nuxt-child/> 用于显示子视图内容。

```
pages/
--| film/
-----| nowplaying.vue
-----| comingsoon.vue
--| film.vue
```

(4) 重定向

a. nuxt.config.js

```
router:{
extendRoutes (routes, resolve) {
routes.push({
path: '/',
redirect: '/film'
})
}
}
```

b. 利用中间件来处理

```
// 中间件 middle/ redirect.js
export default function({ isHMR, app, store, route, params, error, redirect }) {
if (isHMR) return
// 页面均放在_lang文件夹下，即lang为动态路由参数
/*if (!params.lang) { //此写法会出现路由重定向次数过多的问题
return redirect('/' + defaultLocale + '' + route.fullPath)
}
*/
if(route.fullPath == '/film') {
 return redirect('/film/nowplaying')
 }
 }
 router: {
 middleware: 'redirect' // 即每次路由跳转会调用该中间件
 //多个中间件写法
 // middleware: ['redirect']
 }
```

（5） 动态路由

必须加下划线 (文件夹也可以加下划线(多级嵌套)， 文件也可以加下划线)

```
pages/
--| detail/
-----| _id.vue


//编程式跳转 this.$router.push("/detail");
```

（6） 获取动态路由参数

```
 asyncData({params}){
 console.log(params.id);
 }
```

4. 视图

在layout 里面 写好default.vue 可以认为这是根组件的模板了，

所有的组件都加在里面， 但是有些页面 可能不一样，就可以使用 个性化定制页面。

举个例子 layouts/template.vue:

```
<template>
 <div>
 <div>这个页面不需要导航栏</div>
 <nuxt/>
 </div>
</template>

在 pages/detail.vue 里， 可以指定页面组件使用 template 布局。

<script>
export default {
 layout: 'template'
}
</script>
```

5. 异步数据与资源文件

(1) 如果组件的数据不需要异步获取或处理，可以直接返回指定的字面对象作为组件的数据。

```
export default {
 data () {
 return { foo: 'bar' }
 }
}
```

(2)使用 req/res(request/response) 对象

```
在服务器端调用asyncData时，您可以访问用户请求的req和res对象。
在当前页面刷新， 服务端执行此函数
从其他页面跳转过来，客户端执行此函数

export default {
 async asyncData ({ req, res }) {
 // 请检查您是否在服务器端
 // 使用 req 和 res
 if (process.server) { //判断是否在服务器被调用
  //process.client 判断是否在客户端被调用
  return { host: req.headers.host }
 }

  return {}
 }
}
```

(3)错误处理

Nuxt.js 在上下文对象context中提供了一个 error(params) 方法，

你可以通过调用该方法来显示错误信息页面。params.statusCode 可用于指定服务端返回的请求状态码。

以返回 Promise 的方式举个例子：

```
export default {
 asyncData ({ params, error }) {
 return axios.get(`https://my-api/posts/${params.id}`)
 .then((res) => {
 return { title: res.data.title }
 })
```

```
  .catch((e) => {
  error({ statusCode: 404, message: 'Post not found' })
  })
   }
}
```

(4)反向代理的配置 （重启服务器）

```
npm i @nuxtjs/proxy -D
在 nuxt.config.js 配置文件中添加对应的模块，并设置代理

 modules: [
 '@nuxtjs/axios', //添加axios
 '@nuxtjs/proxy' //添加proxy模块
 ],
 axios: {
 proxy: true
  },
  proxy: {
  '/api': {
  target: 'http://example.com',
  pathRewrite: {
  '^/api' : '/'
  }
  }
  }

这样就配置好了webpack的反向代理。
为了在服务端和客户端都工作， 需要

axios.get((process.server?'https://h5.ele.me':'')+"/restapi/shop......e&terminal=h5").then(res=>{
  console.log(res.data)
})
```

如果上线了， 需要在node中配置好 http-proxy-middleware 就工作了。

（5）轮播，下拉刷新， 是必须在浏览器环境中运行

```
// plugins/vue-swipe.js
import { Swipe, SwipeItem } from 'vue-swipe'
```

```
import Vue from 'vue'
import 'vue-swipe/dist/vue-swipe.css'

Vue.component('swipe', Swipe);
Vue.component('swipe-item', SwipeItem);

// nuxt.config.js
 module.exports = {
  plugins: [
  { src: '~/plugins/vue-swipe', ssr: false }
  ]
 }
 直接限制这个插件只在浏览器环境下被引入。(但是会报个警告)

 或者:只要在mounted()中初始化,
 因为服务端渲染只会执行beforeCreate 以及created两个生命周期。
```