## Vue小纪

1. vue是个渐进式的框架。数据驱动，不需要操作dom，是reactive，响应式的。
2. Vue通过Object.defineProperty方法来实现数据劫持或者监听
3. Vue里不允许通过修改数组的下标改变数组的内容，需通过Vue.set（this.arr，1（数组的哪一位），8（改成8））
4. vue的单页面应用是基于路由和组件的，路由用于设定访问路径，并将路径和组件映射起来；单页面应用(SPA)的核心之一是: 更新视图而不重新请求页面；
5. **关于各种指令**
6. v-if：直接删除节点，若为false，在dom节点中不存在；若true，会在dom中新插入 一个节点，影响网页的性能

v-show：控制display：none来切换是否显示（频繁切换显示与不显示时用）

（2）v-for：想遍历哪个元素，就写在哪个元素上面

遍历数组时（item，index）中的index是数组的唯一标识索引

遍历对象是（item，index）中的index是对象的唯一标识key值

（3）{{ msg }}刷新页面的时候会先显示{{ msg }；

V-text会把标签当成字文本输出显示

V-html会把标签解析成真正对应的html标签显示出来，更新部分文本

（4）v-on：简写成@

修饰符①“.”例：@keyup.13 || @keyup.enter 都是绑定的按下回车键

②“.stop”阻止事件冒泡 || js中处理事件冒泡的方法e.stopPropagation（）

③“.prevent”阻止浏览器的默认行为||js中阻止默认行为的方法e.preventDefault

1. V-model：双向数据绑定，主要用于表单提交

修饰符①.lazy（懒惰的）当焦点离开的时候完成数据绑定

②.number 配合text=“number”使用，取到number型的数字

③.trim 去掉当前输入内容的前后空格

（6）v-bind：简写成“：”

（7）v-pre：跳过编译过程，直接显示原来的值{{msg}}

V-cloak：等到变异结束之后使用，:需要与css属性联合使用

V-once：只渲染一次

**6、MVC || MVVM**

1. View（视图）：用户界面
2. Controller（控制器）：业务逻辑 ----------->V-C-M-C-V
3. Model（模型）：数据保存
4. 通过ajax和form表单提交跟后端进行交互
5. MVVM（双向数据绑定）；M-V-ViewModel

**7、计算属性（computed）：**

1. data里面跟computed里面不能同时存在属性）当前我的属性可能依赖其他的值， 当其他值修改时，他就会自动修改当前的值；一个数据受到多个数据的影响。
2. 会缓存属性，用方法（methods）计算的时候会重新计算，性能不够好

**8、侦听属性（watch）：**

1. 监听已有的属性，在data中定义

**9、computed和watch的区别：**

Computed: 一个数据受多个数据的影响；例：商品的总价受单价的影响

Watch: 一个属性的变化影响其他数据的变化； 例： 监听百度的搜索框

**10、生命周期（钩子函数）：被自动调用**

1. beforeCreate：实例创建之前自动调用
2. Created：实例创建成功，data的数据显示出来，但是获取不到$el挂在元素（可以在 此阶段给data赋值）代码没有被编译成对应的数据，还是{{ msg }}
3. beforeMounted：el挂载成功，data数据显示，但是实例没被编译成正确的值 {{msg}}-->dom没有渲染完成
4. Mounted:el挂载成功，模板中的data显示,dom元素渲染完成
5. beforeUpdate：当数据发生变化之前调用，虚拟dom被更新但是真实得dom没有更新
6. Updated：数据发生变化之后调用，虚拟dom和真实dom都被更新
7. beforeDestroy：实例被销毁之前调用 ------------>销毁的都是虚拟的dom------>vue实例 （销毁实例之前的代码放在这个函数中）
8. Destroy：实例被销毁之后调用。所有的时间监听器被移除，所有的子实例也会被销毁

**11、todolist的小例子**

1. forEach（）循环，想循环谁，就把谁放在forEach前面，this.productList。forEach（） 没有返回值
2. Filter（）返回一个符合结果的数组

**12、虚拟dom是在内存中，真实dom是在页面上；**

当虚拟dom树上的节点发生变化的时候，他会把虚拟dom更改的变化更新到真实的 dom上。但是没有发生变化的数据不会更改，这样有利于网页的性能；

频繁操作dom影响性能（虚拟dom有变化的时候才会改变真实dom，才会触发 beforeUpdate（虚拟dom的值被更新）和updated（虚拟dom更新到真实得 dom上）方法）

**13、件（全局组件component || 局部组件components）**

1. 组件要有唯一的根元素
2. 父组件向子组件传值： props传值传的是属性（props同时能检测传值的类型）
3. 子组件向父组件传值： $emit 在定义组件的地方抛出事件，在调用组件的地方监 听相应的事件

$emit（‘’）传的值必须是一个字符串

1. 自定义组件绑定事件的时候需要@click.native
2. Data的值传递给组件：v-bind
3. .is 切换组件

示例：<table>

      <tr is="my-row"></tr>

     </table>

1. slot插槽----->相当于一个占位
2. 组件内部的data必须是一个方法，不能直接定义成对象；.vue结尾的文件里面的data 必须是个方法

**14、vue-cli（全局安装）**

（1）安装npm install -g @vue/cli 查看版本vue -V

（2）nrm：镜像管理工具

**15、vue-router：（vue是个单页面应用，把不同的url对应到加载不同的组件）**

（1）params(传参)，通过传⼀一个对象去传递参数，页⾯面时参数就为空

（2）url(传参):通过to=‘/path/backend/21/java' ，通过在to后⾯面追加参数的⽅方法传参， 但是需要在router⽂文件夹下的index.js下先进⾏行行配置。刷新页⾯面的时候参数 仍然保留留;通过这 种⽅方式传参⽐比较普遍

（3）重定向(双十⼀有个活动的⻚页⾯面，但是活动过时了了，点击的时候会重定向到首页)

配置⼀一个redirect: ‘/’，跳转到⾸首⻚页;要是重定向时需要参数，那么就在配置重 定向时 配置对应的参数。redirect直接改变了了url的值，把url变成了了真实的path 路路径

（4）alias(别名):url路路径没有改变，这种情况更更友好，让⽤用户知道⾃自⼰己访问的路 路径，只是改变了了<router-view>中的内容 。例例:path: ‘/alias’,component: Alias,alias: ‘/abc’,这个 路路径上显示的就是http://。。。。。/abc

（5）过渡动画: <transition name=”fade“>. <router-view> </transiton>;通过.fade去 设置样 式来改变过渡的效果

（6）配置子路由事需要注意，在path： ‘’中不需要加‘/’，页面引用的时候需要写‘/path/backend’

（7）路由跳转以及传参（传参时记得绑定动态属性：to=“”）

①通过name跳转，通过params（刷新页面时数据重载）传参：

<router-link to='{name: "FrontEnd", params: {name: "zxf", age: 25}}'>前端</router-link>

②通过url传参：需要现在index.js里面先配置：path: 'backend/:classId/:className',

然后在页面上传参：<router-link to='/path/backend/21/java'>后端</router-link>，

然后再使用：

<h1>课程ID是：{{$route.params.classId}} 课程名称： {{$route.params.className}} </h1>

③通过query传参：<router-link :to="{ name:'Query',query: { queryId: status }}" >

router-link跳转Query</router-link>

然后在index.js中配置就ok了

获取：this.$route.query.queryId

1. $router.push和$router.replace的区别：

①使用push方法的跳转会向 history 栈添加一个新的记录，当我们点击浏览器的返回 按钮时可以看到之前的页面。

②使用replace方法不会向 history 添加新记录，而是替换掉当前的 history 记录，即 当replace跳转到的网页后，‘后退’按钮不能查看之前的页面。

1. history || hash

①histroy需要在后端服务器上额外配置，会去掉url上的‘#’号

②hash不需要额外配置，会有‘#’号

1. 关于404丢失页面

配置Error.vue页面，并配置path：‘\*’，component：Error

1. 导航守卫（前三个是在router/index.js里面配置）

①全局前置守卫：router . beforeEach，接收3个参数，to（进到哪个路由），from（到 哪个路由），next（false/true）false是禁止跳转，true是允许跳转；next({ ‘ /path’ })， 另一个作用是是直接跳转到指定的地址。因为是全局守卫，所以不管哪个路由跳转 都能够检测的到

②全局后置钩子：router . afterEach，接收2个参数，to（进到哪个路由），from（到 哪个路由），因为已经进入到页面了，所以不需要next去判断了

③路由独享守卫：router . beforeEnter，接收3个参数，同上，可以在组件内部定义， 未定义组件不生效

④组件内部的守卫：

befreRouteEnter：接收3个参数，在渲染该组件的对应路由被confirm前调用，不 能获取组件实例‘this’，因为当守卫执行前，组件实例还没被创建

beforeRouterUpdata：接收3个参数，在当前路由改变，但是该组件被复用时调用，

参数改变也会触发此导航守卫可以访问组件实例‘this’

beforeRouteLeave：接收3个参数，导航离开该组件的对应路由时调用，可以访问 组件实例‘this’

1. 编程式导航：在js里面控制路由的跳转。

后退：this.$router.go(-1) 后退一下

前进：this.$router.go(2) 前进两下

回到首页：this.$router.push(‘/’) push直接回到指定页面首页

1. **vuex（状态管理）：单向数据流**
2. State：驱动应用的数据源，存储数据每个页面中都能够取到这个数据；页面刷 新时数据回到初始状态
3. Mutation（监测改变）：更改state中的值只能通过mutation中的值，这里面的方 法是commit；可以传参，在store.index.js中可以传参，add(state, n),这里面的n代 表形参，点击的时候加n，实参在@click=“add（2）”
4. （mapState，mapMutation， mapGetters）映射state和mutation，getters

使用方法，在组件中:

<h1>计数器：{{ count }} </h1>

<button @click="add">加一</button>

<button @click="reduce">减一</button>

<script>

import {mapState, mapMutations} from 'vuex';

export default {

  computed: {

     ...mapState(['count']) **...扩展运算符**

   },

   methods: {

     ...mapMutations(['add', 'reduce'])

   }

}

</script>

1. getter，相当于vuex中的computed（计算属性），依赖于其他值得变化，其他值发生改变时，getters的值会重新计算
2. Action（类似于mutation，不是必须的），不同的是：
   1. Action提交的是mutation，而不是直接变更状态，通过调用mutation中的方法去改变state的数据，改变state中数据的唯一方法是通过mutation去改
   2. Action可以包含任意异步操作
3. module（模块），每个模块都是自己的state，mutation，getter，action
4. 命名空间。默认情况下，模块内部的action、mutation和geter是注册在全局命名空间的----这样使得多个模块能够对同一mutation或action作出相应；若希望模块具有更高的封装度和复用性，可以通过namespaced: true的方式使其成为带命名空间的模块
5. **nuxt（服务端渲染）**

**（1）优点**

* + 1. 有利于索索引擎优化（SEO）
    2. 更快的到达时间，我们常用的客户端渲染需要先向服务端发送请求，然后再返回回来，一共两次请求，nuxt方式直接向服务端发送请求，一次请求，返回一次

**缺点**

①开发条件艰难

②设计构建设置和部署的更多要求

③更多的服务端负载

**（2）安装(npx在NPM版本5.2.0默认安装了)**

若npm不是5.2.0以上版本，可以通过npm install -g npm升级到npm的最新版本

安装nuxt Npx create-nuxt-app nuxt（项目名称）

（3）目录结构

Assets：资源文件，会被webpack打包到dist文件夹中，但是static中的文件就会 直接复制，不会被webpack打包

1. 配置页面的时候可以用一个单独的文件，也可以用一个文件夹，不需要配置路由
2. **编程式导航**

[使用的时候是@click=“$router.push(](mailto:使用的时候是@click=\“$router.push() 传进来路径 )”