Vue 相关

1.Vue 的核心是什么

Vue 是一套构建用户界面的渐进式自底向上增量开发的 MVVM 框架，vue 的核心只关注视图层，

核心思想：

数据驱动（视图的内容随着数据的改变而改变）

组件化（可以增加代码的复用性，可维护性，可测试性，提高开发效率，方便重复使用，体现了高内聚低耦合）

2.请简述你对 vue 的理解

Vue 是一套构建用户界面的渐进式的自底向上增量开发的 MVVM 框架，核心是关注视图层，vue 的核心是为了解决数据的绑定问题，为了开发大型单页面应用和组件化，所以 vue 的核心思想是数据驱动和组件化，这里也说一下 MVVM 思想，MVVM 思想是 模型 视图 vm 是 v 和 m 连接的桥梁，当模型层数据修改时，VM 层会检测到，并通知视图层进行相应修改

3.请简述 vue 的单向数据流

父级 prop 的更新会向下流动到子组件中，每次父组件发生更新，子组件所有的 prop 都会刷新为最新的值

数据从父组件传递给子组件，只能单向绑定，子组件内部不能直接修改父组件传递过来的数据，（可以使用 data 和 computed 解决）

Vue 常用的修饰符有哪些
修饰符：.lazy 改变后触发，光标离开 input 输入框的时候值才会改变

.number 将输出字符串转为 number 类型

.trim 自动过滤用户输入的首尾空格

事件修饰符：

.stop 阻止点击事件冒泡，相当于原生 js 中的 event.stopPropagation()

.prevent 防止执行预设的行为，相当于原生 js 中 event.preventDefault()

.capture 添加事件侦听器时使用事件捕获模式，就是谁有该事件修饰符，就先触发谁

.self　只会触发自己范围内的事件，不包括子元素

.once 只执行一次

键盘修饰符：

.enter 回车键　　.tab 制表键　　.esc 返回键　　.space 空格键

.up 向上键　　.down 向下键　　.left 向左建　　.right 向右键

系统修饰符：.ctrl .alt　.shift　.meta

5.v-text 与{{}}与 v-html 区别

{{}} 将数据解析为纯文本，不能显示输出 html

v-html 可以渲染输出 html

v-text 将数据解析为纯文本，不能输出真正的 html，与花括号的区别是在页面加载时不显示双花括号

v-text 指令：

作用：操作网页元素中的纯文本内容。{{}}是他的另外一种写法

v-text 与{{}}区别：

v-text 与{{}}等价，{{}}叫模板插值，v-text 叫指令。

有一点区别就是，在渲染的数据比较多的时候，可能会把大括号显示出来，俗称屏幕闪动：

6.v-on 可以绑定多个方法吗

可以　如果绑定多个事件，可以用键值对的形式 事件类型：事件名

如果绑定是多个相同事件，直接用逗号分隔就行

7.Vue 循环的 key 作用

Key 值的存在保证了唯一性，Vue 在执行时，会对节点进行检查，如果没有 key 值，那么 vue 检查到这里有 dom 节点，就会对内容清空并赋新值，如果有 key 值存在，那么会对新老节点进行对比，比较两者 key 是否相同，进行调换位置或删除操作

8.什么是计算属性

计算属性是用来声明式的描述一个值依赖了其他的值，当它依赖的这个值发生改变时，就更新 DOM

当在模板中把数据绑定到一个计算属性上时，vue 会在它依赖的任何值导致该计算属性改变时更新 DOM

每个计算属性都包括一个 getter 和 setter，读取时触发 getter，修改时触发 setter

Vue 单页面的优缺点
单页面 spa

优点：前后端分离　用户体验好 一个字　快　内容改变不需要重新加载整个页面

缺点：不利于 seo，　初次加载时耗长（浏览器一开始就要加载 html　css js ，所有的页面内容都包含在主页面中），页面复杂度提高了，导航不可用

Vuex 是什么？怎么使用？在那种场景下使用
Vuex 是一个专为 vue.js 应用程序开发的状态管理模式，通过创建一个集中的数据存储，方便程序中的所有组件进行访问，简单来说 vuex 就是 vue 的状态管理工具

Vuex 有五个属性 state　getters　mutations　actions　modules

State 就是数据源存放地，对应一般 vue 对象的 data，state 里面存放的数据是响应式的，state 数据发生改变，对应这个数据的组件也会发生改变　用 this.$store.state.xxx 调用

Getters 相当于 store 的计算属性，主要是对 state 中数据的过滤，用 this.$store.getters.xxx 调用

Mutations 处理数据逻辑的方法全部放在 mutations 中，当触发事件想改变 state 数据的时候使用 mutations，用 this.$store.commit 调用，给这个方法添加一个参数，就是 mutation 的载荷（payload）

Actions 异步操作数据，但是是通过 mutation 来操作 用 this.$store.dispatch 来触发，actions 也支持载荷

使用场景：组件之间的状态，登录状态，加入购物车，音乐播放

Vuex 使用流程：

下载 vuex

在 src 下创建 store 以及 index.js

引入 vue 和 vuex， 使用 vuex ，导出实例对象

在 main.js 中引入，在.vue 文件中使用

Vue 中路由跳转方式（声明式/编程式）
Vue 中路由跳转有两种，分别是声明式和编程式

用 js 方式进行跳转的叫编程式导航 this.$router.push（）

用 router-link 进行跳转的叫声明式 router-view 路由出口，路由模板显示的位置

路由中 name 属性有什么作用？
在 router-link 中使用 name 导航到对应路由
使用 name 导航的同时，给子路由传递参数
12.vue 跨域的解决方式

1.后台更改 header

2.使用 jq 提供 jsonp

3.用 http-proxy-middleware（配置代理服务器的中间件）

13.Vue 的生命周期请简述

vue 的生命周期就是 vue 实例创建到实例销毁的过程。期间会有 8 个钩子函数的调用。

beforeCreate（创建实例）

created（创建完成）、

beforeMount（开始创建模板）

mounted（创建完成）、

beforeUpdate（开始更新）

updated（更新完成）、

beforeDestroy（开始销毁）

destroyed（销毁完成）

Vue 生命周期的作用

给了用户在不同阶段添加自己的代码的机会

15.DOM 渲染在那个生命周期阶段内完成

DOM 渲染在 mounted 周期中就已经完成

Vue 路由的实现
前端路由就是更新视图但不请求页面，

利用锚点完成切换，页面不会刷新

官网推荐用 vue-router.js 来引入路由模块

定义路由组件
定义路由，使用 component 进行路由映射组件，用 name 导航到对应路由
创建 router 实例，传入 routes 配置
创建和挂载根实例
用 router-link 设置路由跳转
17.Vue 路由模式 hash 和 history，简单讲一下

Hash 模式地址栏中有#，history 没有，history 模式下刷新，会出现 404 情况，需要后台配置

使用 JavaScript 来对 loaction.hash 进行赋值，改变 URL 的 hash 值

可以使用 hashchange 事件来监听 hash 值的变化

HTML5 提供了 History API 来实现 URL 的变化。其中最主要的 API 有以下两个：history.pushState() 和 history.repalceState()。这两个 API 可以在不进行刷新的情况下，操作浏览器的历史纪录。唯一不同的是，前者是新增一个历史记录，后者是直接替换当前的历史记录。

18.Vue 路由传参的两种方式，params 和 query 方式与区别

动态路由也可以叫路由传参，就是根据不同的选择在同一个组件渲染不同的内容

用法上：query 用 path 引入，params 用 name 引入，接收参数都是类似的，分别是 this.$route.query.name 和 this.$route.params.name

url 展示上:params 类似于 post，query 类似于 get，也就是安全问题，params 传值相对更安全点，query 通过 url 传参，刷新页面还在，params 刷新页面不在了

19.Vue 数据绑定的几种方式

1.单向绑定　双大括号 {{}}　html 内字符串绑定

2.v-bind 绑定　html 属性绑定

3.双向绑定 v-model

4.一次性绑定　v-once 依赖于 v-model

20.Vue 注册一个全局组件

Vue.componnet（“组件的名字”{对象　　template 组建的内容 }）

21.Vue 的路由钩子函数/路由守卫有哪些

全局守卫：beforeEach（to，from，next）和 afterEach（to，from）

路由独享守卫：beforeEnter

组件内的守卫：路由进入/更新/离开之前 beforeRouterEnter/update/leave

22.Vue 中如何进行动态路由设置？有哪些方式？怎么获取传递过来的数据？

动态路由也可以叫路由传参，

动态路由有 query 和 prrams 两种方式传参

query 用 path 引入，params 用 name 引入，query 用 this.$route.query.name 接收参数

params 用 this.$route.params.name 接收参数

23.Elementui 中的常用组件有哪些？请简述你经常使用的 并且他们的属性有哪些？

Container 布局容器

<el-container>外层容器

<el-header>顶栏容器

<el-aside>侧边栏容器

<el-main>主要内容容器

<el-footer>底栏容器

Dropdown　下拉菜单

<el-container split-buton> 下拉按钮

<el-container-menu> 下拉菜单

<el-container-item> 下拉项

Table　表格

Tabs　标签页

Form　表单

Pagination 分页

Message　消息提示

24.Vue-cli 中如何自定义指令

25.Vue 中指令有哪些

v-for：循环数组，对象，字符串，数字

v-on：绑定事件监听

v-bind：动态绑定一个或者多个属性

v-model：表单控件或者组件上创建双向绑定

v-if　v-else　v-else-if　条件渲染

v-show　根据表达式真假，切换元素的 display

v-html 更新元素的 innerhtml

v-text 更新元素的 textcontent

v-pre 跳过这个元素和子元素的编译过程

v-clock 这个指令保持在元素上知道关联实例结束编译

v-once　只渲染一次

26.Vue 如何定义一个过滤器

过滤器本质就是一个有参数有返回值的方法

```
new Vue({
        filters:{
            myCurrency:function(myInput){
                return  处理后的数据

            }

        }

})
```

使用方法：<h1>{{表达式 | 过滤器}}</h1>

过滤器高级用法：可以指定参数，告诉过滤器按照参数进行数据的过滤

27.对 vue 中 keep-alive 的理解

概念：keep-alive 是 vue 的内置组件，当它动态包裹组件时，会缓存不活动的组件实例，它自身不会渲染成一个 DOM 元素也不会出现在父组件链中

作用：在组件切换过程中将状态保留在内存中，防止重复渲染 DOM，减少加载时间以及性能消耗，提高用户体验。

生命周期函数：Activated 在 keep-alive 组件激活时调用，deactivated 在 keep-alive 组件停用时调用

28.如何让组件中的 css 在当前组件生效

在 styled 中加上 scoped

29.Vue 生命周期一共几个阶段

创建 加载 更新 销毁

Beforecreate 创建前

Created 创建后

Beforemount 加载前

Mounted 加载后

Beforeupdate 更新前

Updated 更新后

Beforedestroy 销毁前

Destroyed 销毁后

页面第一次加载会触发 beforecreate created beforemount mounted

DOM 渲染在 mounted 周期中就已经完成

30.Mvvm 与 mvc 的区别

Mvc 模型视图控制器，视图是可以直接访问模型，所以，视图里面会包含模型信息，mvc 关注的是模型不变，所以，在 mvc 中，模型不依赖视图，但是视图依赖模型

Mvvm 模型 视图 和 vm vm 是作为模型和视图的桥梁，当模型层数据改变，vm 会检测到并通知视图层进行相应的修改

31.Vue 组件中的 data 为什么是函数

Data 是一个函数时，每个组件实例都有自己的作用域，每个实例相互独立，不会相互影响

如果是引用类型（对象），当多个组件共用一个数据源时，一处数据改变，所有的组件数据都会改变，所以要利用函数通过 return 返回对象的拷贝，（返回一个新数据），让每个实例都有自己的作用域，相互不影响。

32.Vue 双向绑定的原理

Vue 双向绑定就是：数据变化更新视图，视图变化更新数据

Vue 数据双向绑定是通过数据劫持和观察者模式来实现的，

数据劫持，object.defineproperty 它的目的是：当给属性赋值的时候，程序可以感知到，就可以控制改变属性值

观察者模式 当属性发生改变的时候，使用该数据的地方也发生改变

33.Vue 中组件怎么传值

正向：父传子 父组件把要传递的数据绑定在属性上，发送，子组件通过 props 接收

逆向：子传父　　子组件通过 this.$emit（自定义事件名，要发送的数据），父组件设置一个监听事件来接收，然后拿到数据

兄弟：eventbus　中央事件总线

通过 Vuex

34.Bootstrap 的原理

网格系统的实现原理，通过定义容器大小，平分 12 份，（24 份或者 32 份），再调整内外边距，结合媒体查询，就成了强大的响应式网格系统。

比如　row　col-xs-4

36.如果一个组件在多个项目中使用怎么办

37.槽口请简述

大概分这几点，首先槽口（插槽）可以放什么内容？放在哪？什么作用？可以放任意内容，在子组件中使用，是为了将父组件中的子组件模板数据正常显示。

具名插槽和匿名插槽，作用域插槽，说白了就是在组件上的属性，可以在组件元素内使用，

可以在父组件中使用 slot-scope 从子组件获取数据

38.Watch 请简述

Watch 的作用是监控一个值的变化，并调用因为变化需要执行的方法

39.Vant Ui 请简述下

轻量、可靠的移动端 Vue 组件库

40.计算属性与 watch 区别

Computed　watch　区别就是 computed 的缓存功能，当无关数据数据改变时，不会重新计算，直接使用缓存中的值。计算属性是用来声明式的描述一个值依赖了其他的值，当所依赖的值后者变量发生变化时，计算属性也跟着改变，

Watch 监听的是在 data 中定义的变量，当该变量变化时，会触发 watch 中的方法

41.mvvm 框架是什么？它和其它框架（jquery）的区别是什么？哪些场景适合？

Mvvm 和其他框架的区别是　vue 数据驱动　通过数据来显示视图而不是节点操作

适用于数据操作比较多的场景

42.Vue 首屏加载慢的原因，怎么解决的，白屏时间怎么检测，怎么解决白屏问题

首屏加载慢的原因：

第一次加载页面有很多组件数据需要渲染

解决方法：

1.路由懒加载　component：（）=>import(“路由地址”)

2.ui 框架按需加载

3.gzip 压缩

白屏时间检测：

？？？？

解决白屏问题：

①使用 v-text 渲染数据

②使用{{}}语法渲染数据，但是同时使用 v-cloak 指令（用来保持在元素上直到关联实例结束时候进行编译），v-cloak 要放在什么位置呢，v-cloak 并不需要添加到每个标签，只要在 el 挂载的标签上添加就可以

43.Vue 双数据绑定过程中，这边儿数据改变了怎么通知另一边改变

数据劫持和观察者模式

Vue 数据双向绑定是通过数据劫持和观察者模式来实现的，

数据劫持，object.defineproperty 它的目的是：当给属性赋值的时候，程序可以感知到，就可以控制属性值的有效范围，可以改变其他属性的值

观察者模式它的目的是当属性发生改变的时候，使用该数据的地方也发生改变

44.Vuex 流程

在 vue 组件里面，通过 dispatch 来触发 actions 提交修改数据的操作，然后通过 actions 的

commit 触发 mutations 来修改数据，mutations 接收到 commit 的请求，就会自动通过 mutate 来修改 state，最后由 store 触发每一个调用它的组件的更新

45.Vuex 怎么请求异步数据

1.首先在 state 中创建变量

2.然后在 action 中调用封装好的 axios 请求，异步接收数据，commit 提交给 mutations

Mutations 中改变 state 中的状态，将从 action 中获取到的值赋值给 state

46.Vuex 中 action 如何提交给 mutation 的

Action 函数接收一个与 store 实例具有相同方法和属性的 context 对象，可以调用 context.commit 提交一个 mutation，或者通过 context.state 和 context.getters 获取 state 和 getters

47.Route 与 router 区别

1. router 是 VueRouter 的一个对象，通过 Vue.use(VueRouter)和 VueRouter 构造函数得到一个 router 的实例对象，这个对象中是一个全局的对象，他包含了所有的路由包含了许多关键的对象和属性。

2.route 是一个跳转的路由对象，每一个路由都会有一个 route 对象，是一个局部的对象，可以获取对应的 name,path,params,query 等

49.vuex 的 State 特性是？

State 就是数据源的存放地

State 里面的数据是响应式的，state 中的数据改变，对应这个数据的组件也会发生改变

State 通过 mapstate 把全局的 state 和 getters 映射到当前组件的计算属性中

50.vuex 的 Getter 特性是？

Getter 可以对 state 进行计算操作，它就是 store 的计算属性

Getter 可以在多组件之间复用

如果一个状态只在一个组件内使用，可以不用 getters

51.vuex 的 Mutation 特性是？

更改 vuex　store 中修改状态的唯一办法就是提交 mutation，可以在回调函数中修改 store 中的状态

52.vuex 的 actions 特性是？

Action 类似于 mutation，不同的是 action 提交的是 mutation，不是直接变更状态，可以包含任意异步操作

54.vuex 的优势

优点：解决了非父子组件的通信，减少了 ajax 请求次数，有些可以直接从 state 中获取

缺点：刷新浏览器，vuex 中的 state 会重新变为初始状态，解决办法是 vuex-along，得配合计算属性和 sessionstorage 来实现

55.Vue 路由懒加载（按需加载路由）
56.v-for 与 v-if 优先级

首先不要把 v-if 与 v-for 用在同一个元素上，原因：v-for 比 v-if 优先，如果每一次都需要遍历整个数组，将会影响速度，尤其是当之需要渲染很小一部分的时候。

v-for 比 v-if 具有更高的优先级

请写出饿了么 5 个组件
<el-alert>弹窗</el-alert>
<el-dialog>对话</el-dialog>
<el-calender>日历表</el-calender>
<el-progress:percentage="0">进度条<el-progrees>
<el-switch>开关</el-switch>