vue全家桶：vue+vue-cli+vue-router+vuex+axios

（1）vue-cli：Vue的脚手架工具，用于自动生成Vue项目的目录及文件。

（2）vue-router： Vue提供的前端路由工具，利用其我们实现页面的路由控制，局部刷新及按需加载，构建单页应用，实现前后端分离。

（3）vuex：Vue提供的状态管理工具，用于同一管理我们项目中各种数据的交互和重用，存储我们需要用到数据对象。

（4）ES6：Javascript的新版本，ECMAScript6的简称。利用ES6我们可以简化我们的JS代码，同时利用其提供的强大功能来快速实现JS逻辑。

（5）NPM：node.js的包管理工具，用于同一管理我们前端项目中需要用到的包、插件、工具、命令等，便于开发和维护。

（6）webpack：一款强大的文件打包工具，可以将我们的前端项目文件同一打包压缩至js中，并且可以通过vue-loader等加载器实现语法转化与加载。

（7）Babel：一款将ES6代码转化为浏览器兼容的ES5代码的插件

vue全局组件、局部组件

全局组件注册方式：Vue.component(组件名,{方法})

1.全局组件必须写在Vue实例创建之前，才在该根元素下面生效;

2.模板里面第一级只能有一个标签，不能并行;

局部组件注册方式，直接在Vue实例里面注册

1.属性名为components，s千万别忘了;

2.模板标签比较多的时候，可以定义在一个全局变量里

两个组件内data属性必须都必须是一个函数

Vue中所谓的全局指的是该挂载下的区域，组件处于全局下不可以添加默认事件，要用全局的事件函数，必须父子通信，函数return后面必须跟返回的内容，不能换行写，data里面this指的window，在methods里面this才是Vue实例

组件封装

```
export default {
  install: ( Vue )=>{
    Vue.component(组件名称， 组件的配置)
  }
```

}

vue之watch用法

对应一个对象，键是观察表达式，值是对应回调。值也可以是方法名，或者是对象，包含选项。在实例化时为每个键调用 $watch() ；
//观察数据为字符串或数组
//当单观察数据examples2为对象时，如果键值发生变化，为了监听到数据变化，需要添加deep:true参数

常见面试题：

请详细说下你对vue生命周期的理解？

答：总共分为8个阶段创建前/后，载入前/后，更新前/后，销毁前/后。

- 创建前/后： 在beforeCreate阶段，vue实例的挂载元素el和数据对象data都为undefined，还未初始化。在created阶段，vue实例的数据对象data有了，el还没有。
- 载入前/后：在beforeMount阶段，vue实例的$el和data都初始化了，但还是挂载之前为虚拟的dom节点，data.message还未替换。在mounted阶段，vue实例挂载完成，data.message成功渲染。
- 更新前/后：当data变化时，会触发beforeUpdate和updated方法。
- 销毁前/后：在执行destroy方法后，对data的改变不会再触发周期函数，说明此时vue实例已经解除了事件监听以及和dom的绑定，但是dom结构依然存在

组件的使用和自己创建公用组件？

第一步：在components目录新建你的组件文件（indexPage.vue），script一定要export default {}
第二步：在需要用的页面（组件）中导入：import indexPage from '@/components/indexPage.vue'
第三步：注入到vue的子组件的components属性上面,components:{indexPage}
第四步：在template视图view中使用，
问题有indexPage命名，使用的时候则index-page。

vue如何实现按需加载配合webpack设置?

webpack中提供了require.ensure()来实现按需加载。以前引入路由是通过import 这样的方式引入，改为const定义的方式进行引入。
不进行页面按需加载引入方式：import home from '../../common/home.vue'
进行页面按需加载的引入方式：const home = r => require.ensure([], () => r (require('../../common/home.vue')))

vuex是什么？怎么使用？哪种功能场景使用它？

vue框架中状态管理。在main.js引入store，注入。新建了一个目录store，…..
export 。场景有：单页应用中，组件之间的状态。音乐播放、登录状态、加入购物车

vue请求数据写在哪一步

看实际情况，一般在created里面就可以，如果涉及到需要页面加载完成之后的话就用mounted
数据请求是异步的

vue解决一个方法同时发送多个请求

lodash

```vue
<template>
  <div>
    <div class="bindBtn">
      <button class="bindDataBtn" @click="postAction">提交</button>
    </div>
  </div>
</template>
<script>
import _ from 'lodash'
export default {
  data() {
    return {

    }
  },
  mounted() {

  },
  methods: {
    sendAjax(){
      /*这里是请求的接口、参数以及回调函数等*/
    },
    postAction(){
      this.doPostAction()
    }
  },
  created(){
    this.doPostAction = _.debounce(this.sendAjax,500);
  }
}
</script>
```

我们首先将发送请求的ajax方法写在一个函数里面，在这里就是sendAjax函数，其次，我们引入lodash，然后将sendAjax这个函数用一个方法自定义一下，在这里就是doPostAction，其中_是我们引入的lodash，_.debounce是一个限制操作频率的函数，里面的是500是毫秒单位。

当执行点击事件的时候，也就是postAction函数，我们只需要调用doPostAction这个函数就可以了，而那个500的功能就是你在这个时间段里，无论执行了多少次这个点击事件，它都只会执行一次。

active-class是哪个组件的属性？嵌套路由怎么定义？

答：vue-router模块的router-link组件。

怎么定义vue-router的动态路由？怎么获取传过来的动态参数？

答：在router目录下的index.js文件中，对path属性加上/:id。　　使用router对象的params.id

vue-router有哪几种导航钩子？

答：三种，
一种是全局导航钩子：router.beforeEach(to,from,next)，作用：跳转前进行判断拦截。
参数：有to（去的那个路由）、from（离开的路由）、next（一定要用这个函数才能去到下一个路由，如果不用就拦截）

第二种：组件内的钩子；

- beforeRouteEnter
- beforeRouteUpdate (2.2 新增)
- beforeRouteLeave

```
const Foo = {
  template: `...`,
  beforeRouteEnter (to, from, next) {
    // 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
  },
  beforeRouteUpdate (to, from, next) {
    // 在当前路由改变，但是该组件被复用时调用
    // 举例来说，对于一个带有动态参数的路径 /foo/:id，在 /foo/1 和 /foo/2 之间跳转的时候，
    // 由于会渲染同样的 Foo 组件，因此组件实例会被复用。而这个钩子就会在这个情况下被调用。
    // 可以访问组件实例 `this`
  },
  beforeRouteLeave (to, from, next) {
    // 导航离开该组件的对应路由时调用
    // 可以访问组件实例 `this`
  }
}
```

第三种：单独路由独享组件（router.beforeEnter）

每个守卫方法接收三个参数：

- to: Route: 即将要进入的目标 路由对象
- from: Route: 当前导航正要离开的路由
- next: Function: 一定要调用该方法来 resolve 这个钩子。执行效果依赖 next 方法的调用参数。
    - next(): 进行管道中的下一个钩子。如果全部钩子执行完了，则导航的状态就是 confirmed (确认的)。
    - next(false): 中断当前的导航。如果浏览器的 URL 改变了 (可能是用户手动或者浏览器后退按钮)，那么 URL 地址会重置到 from 路由对应的地址。
    - next('/') 或者 next({ path: '/' }): 跳转到一个不同的地址。当前的导航被中断，然后进行一个新的导航。你可以向 next 传递任意位置对象，且允许设置诸如 replace: true、name: 'home' 之类的选项以及任何用在 router-link 的 to prop 或 router.push 中的选项。

- next(error): (2.4.0+) 如果传入 next 的参数是一个 Error 实例，则导航会被终止且该错误会被传递给 router.onError() 注册过的回调。

确保要调用 next 方法，否则钩子就不会被 resolved。

scss是什么？安装使用的步骤是？有哪几大特性？

答：预处理css，把css当前函数编写，定义变量,嵌套。 先装css-loader、node-loader、sass-loader等加载器模块，在webpack-base.config.js配置文件中加多一个拓展:extenstion，再加多一个模块：module里面test、loader

scss是什么？在vue.cli中的安装使用步骤是？有哪几大特性？

答：css的预编译。

使用步骤：

第一步：用npm 下三个loader（sass-loader、css-loader、node-sass）

第二步：在build目录找到webpack.base.config.js，在那个extends属性中加一个拓展.scss

第三步：还是在同一个文件，配置一个module属性

第四步：然后在组件的style标签加上lang属性 ，例如：lang=”scss”

有哪几大特性:

1、可以用变量，例如（$变量名称=值）；

2、可以用混合器，例如（）

3、可以嵌套

mint-ui是什么？怎么使用？说出至少三个组件使用方法？

答：基于vue的前端组件库。npm安装，然后import样式和js，vue.use（mintUi）全局引入。在单个组件局部引入：import {Toast} from ‘mint-ui’。组件一：Toast(‘登录成功’)；组件二：mint-header；组件三：mint-swiper

v-model是什么？怎么使用？ vue中标签怎么绑定事件？

答：可以实现双向绑定，指令（v-class、v-for、v-if、v-show、v-on）。vue的model层的data属性。绑定事件：<input @click=doLog() />

axios是什么？怎么使用？描述使用它实现登录功能的流程？

答：请求后台资源的模块。npm install axios -S装好，然后发送的是跨域，需在配置文件中config/index.js进行设置。后台如果是Tp5则定义一个资源路由。js中使用import进来，然后.get或.post。返回在.then函数中如果成功，失败则是在.catch函数中

axios+tp5（ThinkPHP5）进阶中，调用axios.post(‘api/user’)是进行的什么操作？axios.put(‘api/user/8′)呢？

答：跨域，添加用户操作，更新操作。

什么是RESTful API？怎么使用?

答：是一个api的标准，无状态请求。请求的路由地址是固定的，如果是tp5则先路由配置中把资源路由配置好。标准有：.post .put .delete

vuex是什么？怎么使用？哪种功能场景使用它？

component ---dispatch---> actions ---commit--->mutations---state <----getters----component

vuex主要由五部分组成：state　action、mutation、getters、mudle组成。

答：vue框架中状态管理。在main.js引入store，注入。新建了一个目录store，…..

export 。场景有：单页应用中，组件之间的状态。音乐播放、登录状态、加入购物车

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？

答：一个model+view+viewModel框架，数据模型model，viewModel连接两个

区别：vue数据驱动，通过数据来显示视图层而不是节点操作。

场景：数据操作比较多的场景，更加便捷

自定义指令（v-check、v-focus）的方法有哪些？它有哪些钩子函数？还有哪些钩子函数参数？

答：全局定义指令：在vue对象的directive方法里面有两个参数，一个是指令名称，另外一个是函数。组件内定义指令：directives

钩子函数：bind（绑定事件触发）、inserted(节点插入的时候触发)、update（组件内相关更新）、

componentUpdated（指令所在组件的 VNode 及其子 VNode 全部更新后调用）、

unbind（只调用一次，指令与元素解绑时调用）

钩子函数参数：el、binding、vnode、oldVnode

钩子函数参数

指令钩子函数会被传入以下参数：

- `el`：指令所绑定的元素，可以用来直接操作 DOM。
- `binding`：一个对象，包含以下属性：
  - `name`：指令名，不包括 `v-` 前缀。
  - `value`：指令的绑定值，例如：`v-my-directive="1 + 1"` 中，绑定值为 `2`。
  - `oldValue`：指令绑定的前一个值，仅在 `update` 和 `componentUpdated` 钩子中可用。无论值是否改变都可用。
  - `expression`：字符串形式的指令表达式。例如 `v-my-directive="1 + 1"` 中，表达式为 `"1 + 1"`。
  - `arg`：传给指令的参数，可选。例如 `v-my-directive:foo` 中，参数为 `"foo"`。
  - `modifiers`：一个包含修饰符的对象。例如：`v-my-directive.foo.bar` 中，修饰符对象为 `{ foo: true, bar: true }`。
- `vnode`：Vue 编译生成的虚拟节点。移步 VNode API 来了解更多详情。
- `oldVnode`：上一个虚拟节点，仅在 `update` 和 `componentUpdated` 钩子中可用。

说出至少4种vue当中的指令和它的用法？

答：v-if：判断是否隐藏；v-for：数据循环出来；v-bind:class：绑定一个属性；v-model：实现双向绑定

vue-router是什么？它有哪些组件？

答：vue用来写路由一个插件。router-link、router-view

Vue的双向数据绑定原理是什么？

答：vue.js 是采用数据劫持结合发布者-订阅者模式的方式，通过Object.defineProperty()来劫持各个属性的setter，getter，在数据变动时发布消息给订阅者，触发相应的监听回调。

具体步骤：

第一步：需要observe的数据对象进行递归遍历，包括子属性对象的属性，都加上 setter和getter

这样的话，给这个对象的某个值赋值，就会触发setter，那么就能监听到了数据变化

第二步：compile解析模板指令，将模板中的变量替换成数据，然后初始化渲染页面视图，并将每个指令对应的节点绑定更新函数，添加监听数据的订阅者，一旦数据有变动，收到通知，更新视图

第三步：Watcher订阅者是Observer和Compile之间通信的桥梁，主要做的事情是:

1、在自身实例化时往属性订阅器(dep)里面添加自己

2、自身必须有一个update()方法

3、待属性变动dep.notice()通知时，能调用自身的update()方法，并触发Compile中绑定的回调，则功成身退。

第四步：MVVM作为数据绑定的入口，整合Observer、Compile和Watcher三者，通过Observer来监听自己的model数据变化，通过Compile来解析编译模板指令，最终利用Watcher搭起Observer和Compile之间的通信桥梁，达到数据变化 -> 视图更新；视图交互变化(input) -> 数据model变更的双向绑定效果。

请说下封装 vue 组件的过程？

答：首先，组件可以提升整个项目的开发效率。能够把页面抽象成多个相对独立的模块，解决了我们传统项目开发：效率低、难维护、复用性等问题。

然后，使用Vue.extend方法创建一个组件，然后使用Vue.component方法注册组件。子组件需要数据，可以在props中接受定义。而子组件修改好数据后，想把数据传递给父组件。可以采用emit方法。

你是怎么认识vuex的？

答：vuex可以理解为一种开发模式或框架。比如PHP有thinkphp，java有spring等。

通过状态（数据源）集中管理驱动组件的变化（好比spring的IOC容器对bean进行集中管理）。

应用级的状态集中放在store中； 改变状态的方式是提交mutations，这是个同步的事物；异步逻辑应该封装在action中。

vue-loader是什么？使用它的用途有哪些？

答：解析.vue文件的一个加载器，跟template/js/style转换成js模块。

用途：js可以写es6、style样式可以scss或less、template可以加jade等

请说出vue.cli项目中src目录每个文件夹和文件的用法？

答：assets文件夹是放静态资源；components是放组件；router是定义路由相关的配置;view视图；app.vue是一个应用主组件；main.js是入口文件

Vue.cli中怎样使用自定义的组件？有遇到过哪些问题吗？

答：第一步：在components目录新建你的组件文件（smithButton.vue），script一定要export default {

第二步：在需要用的页面（组件）中导入：import smithButton from '../components/smithButton.vue'

第三步：注入到vue的子组件的components属性上面,components:{smithButton}

第四步：在template视图view中使用，<smith-button> </smith-button>

问题有：smithButton命名，使用的时候则smith-button。

聊聊你对Vue.js的template编译的理解？

答：简而言之，就是先转化成AST树，再得到的render函数返回VNode（Vue的虚拟DOM节点）

详情步骤：

首先，通过compile编译器把template编译成AST语法树（abstract syntax tree 即 源代码的抽象语法结构的树状表现形式），compile是createCompiler的返回值，createCompiler是用以创建编译器的。另外compile还负责合并option。

然后，AST会经过generate（将AST语法树转化成render funtion字符串的过程）得到render函数，render的返回值是VNode，VNode是Vue的虚拟DOM节点，里面有（标签名、子节点、文本等等）

什么是vue的计算属性？

答：在模板中放入太多的逻辑会让模板过重且难以维护，在需要对数据进行复杂处理，且可能多次使用的情况下，尽量采取计算属性的方式。好处：①使得数据处理结构清晰；②依赖于数据，数据更新，处理结果自动更新；③计算属性内部this指向vm实例；④在template调用时，直接写计算属性名即可；⑤常用的是getter方法，获取

数据，也可以使用set方法改变数据；⑥相较于methods，不管依赖的数据变不变，methods都会重新计算，但是依赖数据不变的时候computed从缓存中获取，不会重新计算

vue.js的两个核心是什么？

答：数据驱动、组件系统

Vue的路由实现：hash模式 和 history模式

hash模式：在浏览器中符号“#”，#以及#后面的字符称之为hash，用window.location.hash读取；

特点：hash虽然在URL中，但不被包括在HTTP请求中；用来指导浏览器动作，对服务端安全无用，hash不会重加载页面。

hash 模式下，仅 hash 符号之前的内容会被包含在请求中，

如 http://www.xxx.com，因此对于后端来说，即使没有做到对路由的全覆盖，也不会返回 404 错误。

history模式：history采用HTML5的新特性；且提供了两个新方法：pushState（），replaceState（）可以对浏览器历史记录栈进行修改，以及popState事件的监听到状态变更。

history 模式下，前端的 URL 必须和实际向后端发起请求的 URL 一致，

如 http://www.xxx.com/items/id。后端如果缺少对 /items/id 的路由处理，将返回 404 错误。**Vue-Router 官网里如此描述：**“不过这种模式要玩好，还需要后台配置支持……所以呢，你要在服务端增加一个覆盖所有情况的候选资源：如果 URL 匹配不到任何静态资源，则应该返回同一个 index.html 页面，这个页面就是你 app 依赖的页面。

chunk、bundle的区别

bundle，chunk，module是什么

bundle：是由webpack打包出来的文件，

chunk：代码块，一个chunk由多个模块组合而成，用于代码的合并和分割。

module：是开发中的单个模块，在webpack的世界，一切皆模块，一个模块对应一个文件，webpack会从配置的entry中递归开始找出所有依赖的模块