# vue面试题-4

**1、对于Vue是一套渐进式框架的理解**  
答：Vue是渐进的，没有强主张，是个轻量视图。它只做了自己应该做的事，没有做多余的事。

**2、vue.js的两个核心是什么？**  
答：数据驱动和组件化。

**3、vue中的模板编译原理**  
答：模板指的就是template。如果我们传了一个template，我们会把template转换成一个render函数，然后通过render函数返回虚拟DOM，再把虚拟的DOM变成真正的DOM。

**4、 响应式数据的原理**答：响应式就是当数据变化的时候，可以让视图也同步更新。核心是Object.defineProperty，vue初始化的时候，Object.defineProperty依次会给data的属性上增加get和set方法，并对依赖进行收集，如果数据发生变化，就会去通知相关的依赖做出对应的更新。

**5、vue生命周期钩子函数有哪些？**  
(vue2.0)

答：① 创建期间的生命周期函数：  
beforeCreate()：此时，实例在内存中刚刚创建出来，data和methods没 有被初始化。  
created()：此时，实例已经在内存中创建完成，data和methods已经被初始化完成。模板还没有编译。  
beforeMount（）：此时，模板已经编译成功，还没有挂载到页面上。  
mounted()：此时，编译好的模板已经挂载到了指定的位置上去。  
② 运行期间的生命周期函数：  
beforeUpdate()：此时，data数据发生改变后，还没有重新渲染DOM树，data的数据是最新的，但是页面上展示的还是旧数据。  
updated()：此时，data中的数据和页面中的渲染是一致的。  
③ 销毁期间的生命周期函数：  
beforeDestroy()：此时，实例的方法、指令都还可以使用，实例销毁之前调用。  
destroyed()：此时，vue实例上的所有指令、绑定、监听都会被销毁，子实例也全部被销毁。

**6、 vue的生命周期钩子是如何实现的？**答：vue的生命周期钩子实际上就是一个回调函数。当我们传入一个钩子函数时，vue内部会帮我们调用，并将生命周期钩子转换成数组，调用的时候，就又会把数组遍历一遍，类似一个发布订阅的模式。

**8、 vue为什么要操作虚拟DOM?**答：虚拟DOM就是用一个对象，来描述真实DOM。一个真实的DOM上面有非常多的属性，操作起来非常不便，为了减少DOM操作，我们在更新的时候就把需要更新的DOM先记录下来，然后更新这些需要更新的DOM，最后再根据diff算法比对，更新DOM。（vue里的diff算法是平级比较，不考虑跨级比较）虚拟DOM不依赖真实的平台环境，可以实现跨平台。

**9、v-if 和 v-show 有什么区别？**答：这两个指令都是在判断DOM节点是否要显示。区别是：  
①.实现方式： v-if是根据后面数据的真假值判断直接从Dom树上删除或重建元素节点。 v-show只是在修改元素的display的属性值，元素始终在Dom树上。  
②.编译过程：v-if切换有一个局部编译/卸载的过程，切换过程中合适地销毁和重建内部的事件监听和子组件； v-show只是简单的基于css切换；  
③.编译条件：v-if是惰性的，如果初始条件为假，则什么也不做；只有在条件第一次变为真时才开始局部编译； v-show是在任何条件下（首次条件是否为真）都被编译，然后被缓存，而且DOM元素始终被保留；  
④.性能消耗：v-if有更高的切换消耗，不适合做频繁的切换； v-show有更高的初始渲染消耗，适合做频繁的切换；

**10、vue常用的修饰符**答：vue修饰符主要有：  
①事件修饰符  
.stop：和原生JavaScript中的event.stopPropagation()一致，阻止事件冒泡  
.prevent：和原生JavaScript中的event.preventDefault()一致，阻止默认事件  
.capture：与事件冒泡的方向相反，事件捕获由外到内  
.self：只会触发自己范围内的事件，不包含子元素  
.once：只会触发一次。  
②按键修饰符  
.delete（捕获“删除”和”退格“键） 用法上和事件修饰符一样，挂载在v-on:后面：

v-on:keyup.xxx=’yyy’ <input @keyup.delete="onKey"/>

* 1

③系统修饰符  
.ctrl .alt .shift .meta这些修饰符可以实现按下相应按键触发鼠标或键盘事件的监听器。

**11、v-on可以监听多个方法吗？**  
答：可以

<button v-on="{mouseenter: onEnter,mouseleave: onLeave}">点我</button>

**12、 vue.mixin的使用场景和原理是什么？**答：vue.mixin可以增加公共方法，当组件初始化调用的时候，mergeOptions方法会进行合并，并针对不同的属性进行合并。vue.mixin也有很多缺点，比如依赖问题、命名问题、数据不能共享、数据来源等问题。

**13、vue中 key 值的作用**答： key的作用主要是为了高效的更新虚拟DOM。在vue中，当使用相同标签名元素的过渡切换时，也会使用到key属性，这样是为了让vue区分它们，否则vue只会替换其内部属性而不会触发过渡效果。

**14、Vue 组件中 data 为什么必须是函数？**答：在 new Vue() 中，data 是可以作为一个对象进行操作的，然而在 component 中，data 只能以函数的形式存在，不能直接将对象赋值给它。 当data选项是一个函数的时候，每个实例可以维护一份被返回对象的独立的拷贝，这样各个实例中的data不会相互影响，是独立的。

**15、v-for 与 v-if 的优先级**答：v-for优先级高于v-if。

**16、 nextTick的实现原理是什么？**答：nextTick中的回调函数是在下一次DOM更新结束之后执行的，延迟了回调，从而防止多次更新。nextTick里面就是一个异步方法（promise）。

**17、说出至少 4 种 vue 当中的指令和它的用法**答：  
① v-if(判断是否隐藏)  
② v-for(遍历)  
③ v-bind(属性绑定)  
⑤ v-model(双向数据绑定)

**18、vue中子组件调用父组件的方法**答：  
① 直接在子组件中通过this.p a r e n t . e v e n t 来 调 用 父 组 件 的 方 法 。 ② 在 子 组 件 里 用 parent.event来调用父组件的方法。 ② 在子组件里用*parent*.*event*来调用父组件的方法。②在子组件里用emit向父组件触发一个事件，父组件监听这个事件就行了。  
③ 父组件把方法传入子组件中，在子组件里直接调用这个方法。

**19、vue中父组件调用子组件的方法**答：  
父组件利用ref属性操作子组件方法。

父：  
  
子：

test() {

console.log('你好')

}}5

在父组件里调用test使用this.$refs.childMethod.test()

1. **vue组件之间传值**答：  
   (1)父子组件传值：  
   ① 父组件调用子组件的时候动态绑定属性  
   ② 子组件定义props接收动态绑定的属性props: [‘dataList’]  
   ③ 子组件获取父子间的属性和方法：  
   在子组件中使用this.p a r e n t . 属 性 名 / t h i s . parent.属性名 /this.*parent*.属性名/*this*.parent.方法名。  
   ④ 子组件给父组件传值：  
   a. 使用ref属性  
   父组件调用子组件时绑定属性ref  
   在父组件中使用this.r e f s . p a r e n t . 属 性 / t h i s . refs.parent.属性/this.*refs*.*parent*.属

*this*.refs.parent.方法  
b. 使用e m i t 方 法 子 组 件 调 用 t h i s . emit方法 子组件调用this.*emit*方法子组件调用*this*.emit('方法名‘,传值)  
父组件通过子组件绑定的’方法名’获取传值  
（2）vue页面级组件之间传值  
① 使用vue-router通过跳转链接带参数传参。  
② 使用本地缓存localStorge。  
③ 使用vuex数据管理传值。

**21、说说vue的动态组件。**答：多个组件通过同一个挂载点进行组件的切换，is的值是哪个组件的名称，那么页面就会显示哪个组件。

**22、keep-alive内置组件的作用**答：keep-alive是vue内置的一个组件，而这个组件的作用就是能够缓存不活动的组件，组件进行切换的时候，默认会进行销毁，如果有需求，某个组件切换后不进行销毁，而是保存之前的状态，那么就可以利用keep-alive来实现。

**23、vue中是如何检测数组变化的**答：vue将数组原型上的方法进行了重新编写，更改了一些数组的方法，比如push、shift、pop、splice、unshift、sort、reverse，这些方法都有一个特点，就是可以改变数组原来的值。当我们用了这些方法来操作数组时，就会把原来的方法进行劫持，可以在函数内部添加自己的功能。如果想跟新数组的索引，需要使用vue.$set方法来实现。

**24、 vue.set方法**答：vue不允许在已经创建的实例上动态添加新的根级响应式属性，$set可以触发更新，当对象新增不存在的属性时，会触发对象依赖的watcher去更新，当更改数组索引时，我们调用数组的splice方法去更新数组。  
操作数组示列：

this.$set(arr, index, val)

* 1

操作对象示例：

this.$set( obj, key, val)

*25、nextTick的实现原理是什么*答：nextTick中的回调函数是在下一次DOM更新结束之后执行的，延迟了回调，从而防止多次更新。nextTick里面就是一个异步方法（promise）。

在下次 DOM 更新循环结束之后执行延迟回调。在修改数据之后立即使用这个方法，获取更新后的 DOM

**26、递归组件的用法**

答：在export default中，有一个属性是name。这属性对递归组件来说非常重要。递归组件只能通过 name 选项来做事。递归组件一定要有一个结束的条件，否则就会使组件循环引用，最终出现“max stack size exceeded”的错误，也就是栈溢出。那么，我们可以使用v-if="false"作为递归组件的结束条件。当遇到v-if为false时，组件将不会再进行渲染。

**27、怎么定义vue-router的动态路由？怎么获取传过来的值？**答：动态路由的创建,使用path属性，使用动态路径参数，以冒号开头:

{

path: '/test/:id'

name: 'Test'

components: Test

}

访问test目录下的所有文件，test上的所有属性都会映射到Test组件上。  
当读取/test下的路由时，参数会被放到this.r o u t e . p a r a m s 里 面 。 可 以 通 过 t h i s . route.params里面。可以通过 this.*route*.*params*里面。可以通过*this*.route.params.id动态获取参数。

**28、vue-router有哪几种路由守卫?**答：路由守卫为  
全局守卫：beforeEach  
后置守卫：afterEach  
全局解析守卫：beforeResolve  
路由独享守卫：beforeEnter

**29、beforeEach的原理**答：router.beforeEach（）用来做一些进入页面的限制。比如登录，若没有登录就不能进入页面，只有登录了之后才有权限查看某些页面。也就是路由拦截。  
原理：  
当一个导航触发时，全局前置守卫按照创建顺序调用。守卫是异步执行，此时导航在所有守卫 resolve 完之前都一直处 等待状态中。  
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。执行依赖 next 方法的调用参数。  
next(): 进行队列中的下一个钩子。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。  
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。

**30、$route和 r o u t e r 的 区 别 是 什 么 ？ 答 ： router的区别是什么？ 答：*router*的区别是什么？**

答：router为VueRouter的实例，是一个全局路由对象，包含了路由跳转的方法、钩子函数等。  
$route是路由信息对象跳转的路由对象，每一个路由都会有一个route对象，是一个局部对象，里面包含：path、params、hash、query、fullPath、matched、name等路由信息参数。

**31、vue-router响应路由参数的变化**答：  
① 用watch 监听  
② 使用组件内beforeRouteUpdate(to,from,next)导航钩子函数，to表示将要跳转的路由对象，from表示从哪个路由跳转过来，next是进行下一个钩子函数

**32、 vue-router 传参**答：① 使用Params:只能使用name属性，不能使用path，参数不会显示在路径上，浏览器强制刷新参数会被清空  
② 使用Query：参数会显示在路径上，刷新不会被清空，name 可以使用path路径

**33、不用Vuex会带来什么问题？**答：  
① 可维护性下降，修改数据，需要维护好个地方  
② 可读性下降，组件内的数据来源不明确  
③增加耦合，Vue用Component的初衷在于降低耦合性，如果大量的上传分发，反而会增加耦合度。

**34、vuex有哪几种属性？**答：State、 Getter、Mutation 、Action、 Module。

**35、vuex的State特性是？**答：Vuex就好比是一个仓库，仓库里面放了很多对象。其中state就是存放数据源的地方，对应Vue对象里面的data。state里面存放的数据是响应式的，Vue组件从store中读取数据，若是store中的数据发生改变，依赖这个数据的组件也会发生更新。它通过mapState把全局的 state 和 getters 映射到当前组件的 computed 计算属性中

**36、vuex的Getter特性是？**答：getters 可以对State进行计算操作，它就是Store的计算属性。是getters 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

**37、vuex的Mutation特性是？**答：Action 类似于 mutation，不同在于：Action 提交的是 mutation，而不是直接变更状态。Action 可以包含任意异步操作

**38、Vue.js中ajax请求代码应该写在组件的methods中还是vuex的actions中？**答：  
① 如果请求来的数据不需要被其他组件复用，只在请求的组件内使用，就不需要放入vuex 的state里。  
② 如果被其他地方复用，就可以把请求放入action里，包装成promise返回，在调用的地方用async await处理返回的数据。

**39. 简述Vue的响应式原理**

当一个Vue实例创建时，vue会遍历data选项的属性，用 Object.defineProperty 将它们转为 getter/setter并且在内部追踪相关依赖，在属性被访问和修改时通知变化。

每个组件实例都有相应的 watcher 程序实例，它会在组件渲染的过程中把属性记录为依赖，之后当依赖项的 setter 被调用时，会通知 watcher 重新计算，从而致使它关联的组件得以更新。