



Vue 2 押题

Vue 2 的生命周期钩子有哪些？数据请求放在哪个钩子？

Vue 2 组件间通信方式有哪些？

Vuex 用过吗？怎么理解？

VueRouter 用过吗？怎么理解？

Vue 2 是如何实现双向绑定的？



📱 扫码购买 [《前端押题》视频课程](#)

😁 让您面试无忧

😊 绝对物超所值

Vue 2 的生命周期钩子有哪些？数据请求放在哪个钩子？

Vue 2 文档里有一张图说得很清楚，红色空心框中的文字皆为生命周期钩子：

1. create x 2 (before + ed) - SSR
2. mount x 2
3. update x 2

4. destroy x 2

还有三个写在钩子列表里：

1. activated
2. deactivated
3. errorCaptured

请求放在 mounted 里面，因为放在其他地方都不合适（xjb扯）。

Vue 2 组件间通信方式有哪些？

1. 父子组件：使用「props 和事件」进行通信
2. 爷孙组件：
 - a. 使用两次父子组件间通信来实现
 - b. 使用「provide + inject」来通信
3. 任意组件：使用 eventBus = new Vue() 来通信
 - a. 主要API 是 eventBus.\$on 和 eventBus.\$emit
 - b. 缺点是事件多了就很乱，难以维护
4. 任意组件：使用 Vuex 通信（Vue 3 可用 Pinia 代替 Vuex）

Vuex 用过吗？怎么理解？

1. 背下文档第一句：Vuex 是一个专为 Vue.js 应用程序开发的状态管理模式 + 库
2. 说出核心概念的名字和作用：store/State/Getter/Mutation/Action/Module
 - a. store 是个大容器，包含以下所有内容
 - b. State 用来读取状态，带有一个 mapState 辅助函数
 - c. Getter 用来读取派生状态，附有一个 mapGetters 辅助函数
 - d. Mutation 用于同步提交状态变更，附有一个 mapMutations 辅助函数
 - e. Action 用于异步变更状态，但它提交的是 mutation，而不是直接变更状态。
 - f. Module 用来给 store 划分模块，方便维护代码

常见追问：Mutation 和 Action 为什么要分开？

答案：为了让代码更易于维护。（可是 Pinia 就把 Mutation 和 Action 合并了呀）
完。

VueRouter 用过吗？怎么理解？


1. 背下文档第一句：Vue Router 是 Vue.js 的官方路由。它与 Vue.js 核心深度集成，让用 Vue.js 构建单页应用变得轻而易举。
2. 说出核心概念的名字和作用：`router-link` `router-view` 嵌套路由、Hash 模式和 History 模式、导航守卫、懒加载
3. 常见追问：
 - a. Hash 模式和 History 模式的区别？
 - i. 一个用的 Hash，一个用的 History API
 - ii. 一个不需要后端 nginx 配合，一个需要
 - b. 导航守卫如何实现登录控制？

```
router.beforeEach((to, from, next) => {  
  if (to.path === '/login') return next()  
  if (to是受控页面 && 没有登录) return next('/login')  
  next()  
})
```

推荐阅读：

路由守卫

在系统路由跳转前做权限校验，是经常遇到的需求。本文将使用 Vue-Router 中的路由守卫功能实现权限控制和加载进度。Vue-

 https://blog.csdn.net/sinat_36521655/article/details/10612...

原创

Vue 2 是如何实现双向绑定的？

vue的双向绑定原理及实现 - canfoo#! - 博客园

使用vue也好有一段时间了，虽然对其双向绑定原理也有了解个大概，但也没好好探究下其原理实现，所以这次特意花了几晚时间

 <https://www.cnblogs.com/canfoo/p/6891868.html>

```
<script src="js/watcher.js"></script>  
<script src="js/compile.js"></script>  
<script src="js/index.js"></script>  
<script type="text/javascript">  
  
  new SelfVue({  
    el: '#app',  
    data: {  
      title: 'hello world',  
      name: 'canfoo'  
    },  
    methods: {  
      clickMe: function () {  
        this.title = 'hello world';  
      }  
    }  
  })  
</script>
```

1. 说明一般使用 `v-model` / `.sync` 实现, ``v-model`` 是 `v-bind:value` 和 `v-on:input` 的语法糖
 - a. `v-bind:value` 实现了 `data ⇒ UI` 的单向绑定
 - b. `v-on:input` 实现了 `UI ⇒ data` 的单向绑定
 - c. 加起来就是双向绑定了
2. 这两个单向绑定是如何实现的呢?
 - a. 前者通过 `Object.defineProperty` API 给 `data` 创建 `getter` 和 `setter`, 用于监听 `data` 的改变, `data` 一变就会安排改变 `UI`
 - b. 后者通过 `template compiler` 给 `DOM` 添加事件监听, `DOM input` 的值变了就会去修改 `data`。

网上的博客讲得很绕, 你可以尝试理解看看。