



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

第三版：Vue 23 道

为什么 Vue 被称为“渐进框架”？

使用渐进式框架的代价很小，从而使现有项目（使用其他技术构建的项目）更容易采用并迁移到新框架。Vue.js 是一个渐进式框架，因为你可以逐步将其引入现有应用，而不必从头开始重写整个程序。

Vue 的最基本和核心的部分涉及“视图”层，因此可以通过逐步将 Vue 引入程序并替换“视图”实现来开始你的旅程。

由于其不断发展的性质，Vue 与其他库配合使用非常好，并且非常容易上手。这与 Angular.js 之类的框架相反，后者要求将现有程序完全重构并在该框架中实现。

Vue.js 中的声明式渲染是什么？

Vue.js 使渲染数据变得容易，并隐藏了内部实现。例如下面的代码：

HTML

```
<div id=" app" ></div>
```

JavaScript

```
const greeting = "Hello there!";  
const appDiv = document.getElementById( "app" );  
appDiv.innerText = greeting;
```

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

上面的代码段将在 ID 为 “app” 的 div 中显示短语 “Hello there! ”。代码包含实现结果所需的所有步骤。首先选择 ID 为 “app” 的 DOM 元素，然后用 innerText 属性手动设置 div 的内容。

现在，让我们看看在 Vue 中是怎么做的。

Template

```
<div id=" app" >{{ greeting }}</div>
```

App

```
new Vue({
  data: {
    greeting: 'Hello There!'
  },
  el: '#app'
});
```

我们在 Vue 程序中创建了一个名为 “greeting” 的数据属性，但是只需要在 div 中用 mustache 语法输入 “greeting” 即可，而不必关心内部实现。我们声明了 “greeting” 变量，其余的由 Vue 完成。这就是声明式渲染的样子。Vue 隐藏并管理内部信息。

你用哪个指令遍历对象的属性？

要遍历对象或数组，可以使用 **v-for** 指令。下面是一个例子：

Template

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

```
<div id="app">
  <ul>
    <li v-for="(value, key) in card">{{ key }} - {{ value }}</li>
  </ul>
</div>
```

App

```
new Vue({
  el: '#app',
  data: {
    card: {
      name: 'John Doe',
      age: 25,
      city: 'New York',
      country: 'US'
    }
  }
});
```

输出

1. name - John Doe
2. age - 25
3. city - New York
4. country - US

给出模板，描述 Vue 程序的输出。

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

模板：

```
{{title}}
```

App:

```
new Vue({
  el: '#app',
  data: {
    title: 'Vue.js'
  }
})
```

上面的代码将在 `div` 中输出字符串 `<h1 style="color: green;">Vue.js</h1>`。之所以将整个标签渲染为字符串，是因为 `mustache` 模板标签 `{{title}}` 将传入的数据视为字符串，而不将其解析为可执行代码。这也有助于缓解把恶意代码注入到页面的 `XSS` 相关的问题。这类似于在 `JavaScript` 中使用 `elementSelector.innerText = text` 语句。

如何在输入框和数据属性之间实现双向数据绑定？

要实现双向数据绑定，可以使用 `v-model` 指令。`v-model` 指令主要是语法糖：

```
<input type="text" :value="nameInput" @keyup="nameInput =
$event.target.value">
```

在上面的语句中，每当观察到 `"keyup"` 事件时，就会将名为 `"nameInput"` 的数据属性设置为输入框的值。同时，将输入框的 `value` 属性绑定到 `"nameInput"` 数据属性。这样在表单字段和数据属性之间建立了双向数据关系。

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

v-model 可以做到这一点，并且比手动设置更有效地。要使用 v-model 复制上述效果，请再次在同一输入框中输入以下内容：

```
<input type="text" v-model="nameInput">
```

需要注意的是，当实现双向数据绑定时，使用的数据属性被认为是事实上的来源。在 data 属性上所做的任何更改都将优先于 form 字段上的用户输入事件。

你如何捕获元素上的点击事件？

可以使用 v-on:click 指令捕获 Click 事件。该指令也可以用缩写符号 @click 表示。这是一个例子：

v-on:click 符号

```
<a v-on:click="clickHandler" >Launch!</a>
```

@click 符号

```
<a @click="clickHandler" >Launch!</a>
```

什么是动态 prop？

当使用 v-bind 指令为 prop 分配值作为绑定到属性的函数时，被称为动态 prop。例如以下组件的 tweet 属性绑定到名为 tweetText 的数据属性。这与静态硬编码值相反。这种绑定始终是单向的，这意味着数据可以从父组件流到子组件，而绝不会反过来。

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

```
<TweetBox :tweet=" tweetText" >
```

Vue.js 中的指令是什么？

指令是一系列特殊属性，你可以通过将其添加到模板 HTML 标记中来赋予它们特殊的响应功能。指令允许模板中的元素使用数据属性、方法、计算或监视的属性和内联表达式根据定义的逻辑对更改做出反应。例如以下代码使用 `v-on` 指令在组件上实现 `click` 事件侦听器。

```
<SignUpButton v-on:click=" doSignup" />
```

或者

```
<SignUpButton @click=" doSignup" />
```

在这个例子中，我们使用 `v-if` 指令基于名为 `showButton` 的数据属性显示或删除元素与组件。指令以 `v-` 开头来指示 Vue 特定的属性。此规则的例外是 `v-on` 和 `v-bind` 的简写形式。

```
<SignUpButton v-if=" showButton" />
```

Vue 还允许定义自己的自定义指令。

v-show 指令的用途是什么？

`v-show` 指令允许有条件地显示元素。在下面的代码中，仅当 `isDisplayed` 数据属性为 `true` 时，才会显示该元素。

```
<TweetBox v-show=" isDisplayed" >
```

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

使用 `v-show` 指令时，可使用 CSS 的 `display` 属性切换元素的可见性。

v-show 与 v-if 指令有何不同？

`v-show` 和 `v-if` 都用于有条件地显示元素，而后者提供了条件渲染的真正实现。`v-show` 只需切换 CSS 的 `display` 属性即可显示或隐藏元素，而 `v-if` 指令可创建或销毁组件。每次显示状态更改时，代价通常会更大。

另一方面，`v-show` 成本较低，因为它仅切换元素的 CSS 显示属性。所以如果必须经常切换元素，则 `v-show` 会提供比 `v-if` 更好，更优化的结果。

就加载元素的初始渲染成本而言，`v-if` 不会渲染最初隐藏的元素节点，而 `v-show` 会渲染其 CSS `display` 属性被设置为 `none` 的元素。

对于作为元素实现的注释框，我们希望使用户能够按下键盘上的

Enter 键，来将内容提交给名为 “storeComment” 的方法。

在代码中对此进行演示。

可以在任何元素上使用 `v-on` 指令来实现事件侦听器。此外，`v-on` 还允许我们将按键修饰符用于 “enter”，“tab”，“esc”，“space” 等常见按键。这是一个例子：

模板

```
<textarea @keyup.enter="storeComment"> </textarea>
```

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

App

```
new Vue({
  el: '#app',
  methods: {
    storeComment(event) {
      // access the value of the textarea box using event.target.value or
      // use v-model to bind to a data property
    }
  }
});
```

如何在单页 Vue 应用 (SPA) 中实现路由?

可以通过官方的 `vue-router` 库在用 `Vue` 构建的 SPA 中进行路由。该库提供了全面的功能集，其中包括嵌套路线、路线参数和通配符、过渡、HTML5 历史与哈希模式和自定义滚动行为等功能。`Vue` 还支持某些第三方路由器包。

使用 `Vue` 时调用 `event.preventDefault()` 的最佳方式是什么?

在事件侦听器上调用 `event.preventDefault()` 的最佳方式是将 `.prevent` 修饰符与 `v-on` 指令一起使用。这是一个例子：

```
<a @click.prevent="doSomethingWhenClicked" >Do Something</a>
```

什么是过滤器?

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

过滤器是在 Vue 程序中实现自定义文本格式的一种非常简单的方法。它们就像可以在表达式中通过管道传递（使用管道字符）以取得结果的运算符。下面是一个可以反转文本字符串的过滤器示例：

模板

```
<div id="app">{{ title | reverseText }}</div>
```

App

```
new Vue({
  el: '#app',
  data: {
    title: 'This is a title'
  },
  filters: {
    reverseText(text) {
      return text.split("").reverse().join("");
    }
  }
});
```

输出

```
elTit a si sihT
```

在上面的例子中，我们创建了一个名为 `reverseText` 的过滤器，该过滤器反转文本字符串并返回。这是一个简单的函数，接受输入并返回处理后的输出。通过在过滤器下声明，它就可以成为可以在模板中使用的过滤器。

关注公众号：磊哥聊编程，回复：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

在模板中，我们只是将 `reverseText` 过滤器通过管道传递到了想要在 `mustache` 标签中显示的数据变量。这样可以将多个过滤器管道连接在一起。因此过滤器提供了一种非常优雅的方式来处理文本。

如何动态地在元素上切换 CSS 类？

Vue 允许我们绑定到 `class` 属性。在下面的例子中，我们将 `class` 属性绑定到一个对象，该对象允许使用 `data` 属性切换类。

模板

```
<div :class=" { divStyle : showDiv }" ></div>
```

App

```
new Vue({
  el: '#app',
  data: {
    showDiv: true
  }
});
```

在上面的代码中，只要数据属性 `showDiv` 为 `true`，类名 `divStyle` 将应用于 `div`。



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

绑定 HTML 类时，该如何连接类？假设存在一个元素：

Process。我们只希望使用名为 **“isActive”** 的数据属性动态地切换 **btnActive** 类。

这可以在绑定类时用 **Array** 来实现。以下是实现方法：

模板

```
<Button :class=" [ 'btn' , 'btnRed' , { btnActive :  
isActive }]" >Process</button>
```

App

```
new Vue({  
  el: '#app',  
  data: {  
    isActive: true  
  }  
});
```

在上面的代码段中，将串联各个类的数组，并基于 **isActive** 数据属性的值对对象中的表达式进行响应式评估。

什么是计算属性？

关注公众号：磊哥聊编程，回复：¹¹面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

```
computed: {
  isValid() {
    return !emailRegex.test(this.email);
  }
});
```

在上面的代码示例中，如果正则表达式测试针对电子邮件输入框失败，则 `isValid` 计算属性将返回 `true`。如果电子邮件验证程序认为输入的值无效，就会看到文本框变为红色（你必须创建一个名为 `.invalid` 的类，并将背景颜色属性设置为红色）。当用户键入内容时，将重新执行计算的方法，并且在验证格式之后，动态删除无效的类。

如何确保在单文件组件中定义的 CSS 样式仅应用于该组件，而不被用于其他组件？

这可以通过样式标签上的 `scoped` 属性来实现。在内部 Vue 使用 PostCSS 插件为所有样式元素分配唯一的数据属性，然后使样式针对这些唯一的元素。举个例子：

```
<template>
  <div class=" title" >This is a title</div>
</template>
<style scoped>
  .title {
    font-family: sans-serif;
    font-size: 20px;
  }
</style>
```

关注公众号：磊哥聊编程，回复¹³：面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

如何将数据从父组件传递到子组件？

可以作为组件中单向入口的 `prop` 把数据向下传递到子组件。这是一个例子：

```
<template>
  <div>
    <contact-list-item v-for="contact in contacts" :contact="contact"
  />
  </div>
</template>
<script>
  import ContactListItem from './ContactListItem' ;
  export default {
name: 'address-book' ,
data: function() {
return {
contacts: [.....]
}
},
components: {
ContactListItem
}
}
</script>
```

`contact-list-item` 上绑定的 `prop` “`contact`” 是一个入口，用于从用作子项的父组件接收数据。在 `contact-list-item` 组件中：

```
<template>
  <div>
```

关注公众号：磊哥聊编程，回复：¹⁴面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

```
<span>{{ contact.name }}</span>
<span>{{ contact.email }}</span>
</div>
</template>

<script>
  export default {
    name: 'contact-list-item' ,
    props: [ 'contact' ]
  }
</script>
```

在这里声明了一个名为 “contact” 的 prop 引入数据，然后可以直接在模板部分中显示。

什么是组件？

组件本质上是 Vue 实例，它们封装模板、逻辑和可选的本地响应性数据属性，能够提供可重新使用的自定义构建元素。可重用性是构建组件的核心。

使用单文件组件构建应用程序时，组件在扩展名为 .vue 的文件中定义。单文件组件包含三个部分：模板部分定义了该组件的 HTML 布局；脚本部分定义了数据、属性和逻辑单元（如方法）并将内容导出为 Vue 组件；还有一个样式部分，用于定义组件的样式表。单文件组件使用 Webpack 等模块捆绑器进行编译。

什么是生命周期 hook？列出一些生命周期 hook。

关注公众号：磊哥聊编程，回复：¹⁵面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

Vue 实例（组件）从其初始化到销毁和删除都经历生命周期。在整个过程中，Vue 允许开发人员运行自定义函数的几个阶段。这些函数称为生命周期 hook。以下是一些生命周期 hook 的列表：

- created
- mounted
- updated
- destroyed

什么是插槽 (slot) ?

插槽允许你定义可以封装和接受子 DOM 元素的元素。组件模板中的 `<slot>` `</slot>` 元素作为通过组件标签捕获的所有 DOM 元素的出口。这是一个例子：

Post.vue | 实现插槽的组件

```
<template>
  <div class="hello">
    <h3 class="title">{{title}}</h3>
    <div class="content">
      <slot> </slot>
    </div>
  </div>
</template>
```

App.vue | 使用 Post 组件的 App 组件

关注公众号：磊哥聊编程，回复：¹⁶面试题，获取最新版面试题



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

```
<template>
  <div id="app">
    <Post title="Hello from Vue!">
```

Vue 是用于构建用户界面的渐进框架。与其他框架不同，Vue 从头开始设计以逐渐采用。核心库仅集中在视图层，并且很容易与其他库或现有项目集成。另一方面，当与现代工具和支持库结合使用时，Vue 也完全能够为复杂的单页应用程序提供支持。

```
    </Post>
  </div>
</template>
```

在上面的示例中，斜体文本显示在 `Post` 组件中标有 `<slot>` 元素的区域内。

什么是观察者？

观察者允许我们观察更改的特定属性，并执行定义为函数的自定义操作。尽管它们的用例与计算的属性相交叉，但是当某些数据属性发生改变时，有时需要观察者执行自定义操作或运行代价昂贵的操作。

如何从子组件发出自定义事件？

可以用 `$emit('event-name', eventPayload)` 发出自定义事件。然后可以像其他事件一样，用 `v-on` 指令在父组件上拦截。



微信搜一搜

磊哥聊编程

扫码关注



回复：面试题 获取最新版面试题

开发人员经常使用字母 “vm” 作为变量名来声明根 Vue 实例。例如 `const vm = new Vue()`。在这种情况下，“vm”指的是什么？

虽然这不是约定，但是开发人员经常使用变量名称 'vm' 来命名根 Vue 实例，该变量名称代表 'ViewModel'，因为 Vue 本质上负责视图层，并且部分受到了 MVVM 模式的启发 (Model-View-View-Model)。但是，根本没有必要将根实例命名为 “vm”。