

Vue 核心技术与实战

day01



黑马程序员
www.itheima.com

传智教育旗下
高端IT教育品牌



目录

Contents

◆ Vue 快速上手

Vue 概念 / 创建实例 / 插值表达式 / 响应式特性 / 开发者工具

◆ Vue 指令

v-html / v-show / v-if / v-else / v-on / v-bind / v-for / v-model

◆ 综合案例 - 小黑记事本

列表渲染 / 删除功能 / 添加功能 / 底部统计 / 清空

综合案例：

小黑记事本

请输入任务

添加任务


1. 跑步一公里

2. 跳绳200个

3. 游泳100米

合计: 3

清空任务



目录

Contents

◆ Vue 快速上手

Vue 概念 / 创建实例 / 插值表达式 / 响应式特性 / 开发者工具

◆ Vue 指令

v-html / v-show / v-if / v-else / v-on / v-bind / v-for / v-model

◆ 综合案例 - 小黑记事本

列表渲染 / 删除功能 / 添加功能 / 底部统计 / 清空

基于数据渲染出用户看到的页面

Vue 是什么

概念：Vue 是一个用于 **构建用户界面**^① 的 **渐进式**^② 框架^③

```
title: '清仓大促',  
products: [  
  { id: 1, name: '手机', price: '1999元' },  
  { id: 2, name: '平板', price: '2999元' },  
  { id: 3, name: '电脑', price: '3999元' },  
]
```

数据



清仓大促

- 手机 - 1999元
- 平板 - 2999元
- 电脑 - 3999元

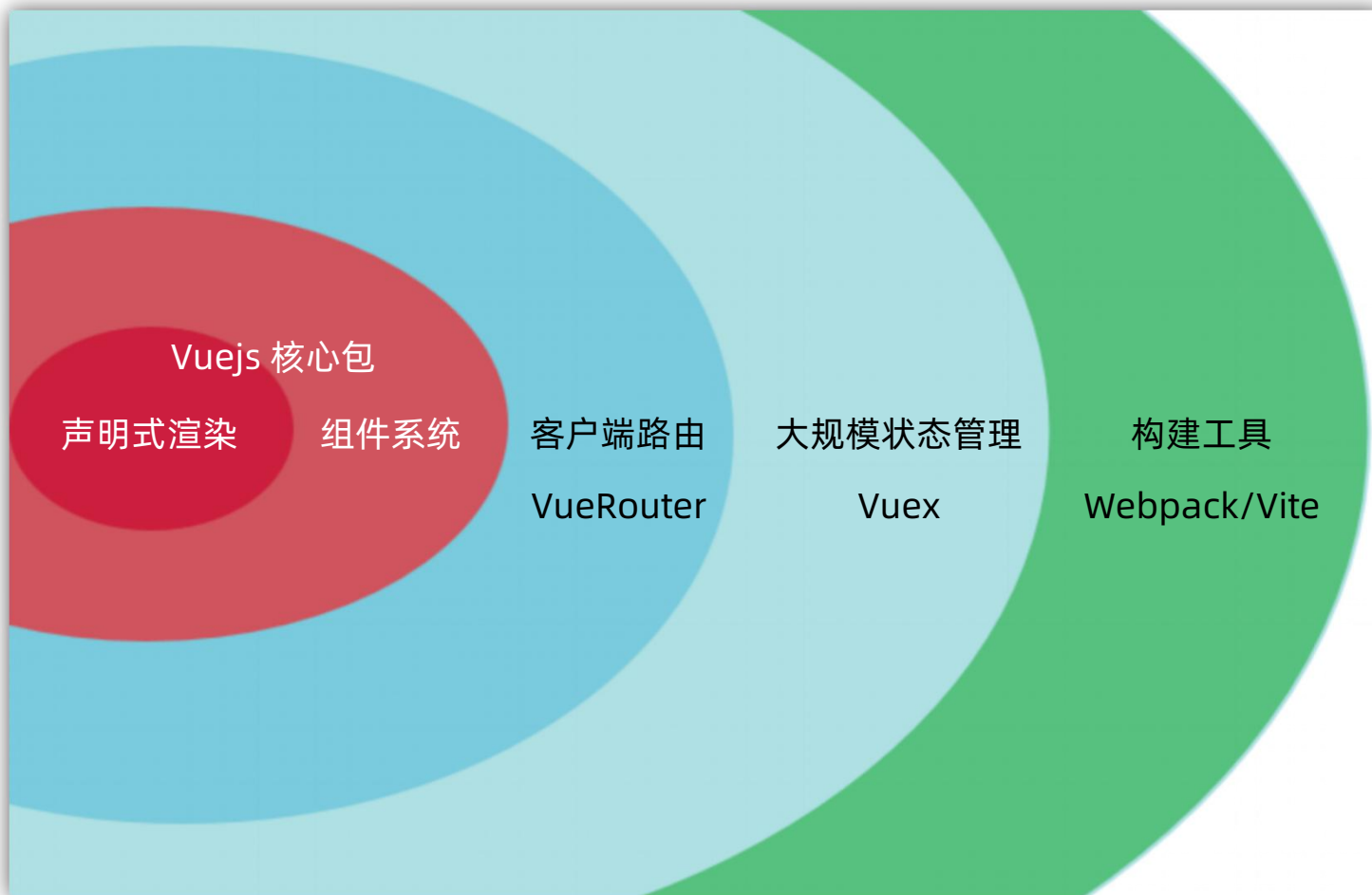
用户页面（视图）



Vue 是什么

概念：Vue 是一个用于 构建用户界面^①的 渐进式^② 框架^③

循序渐进



Vue 的两种使用方式：

① Vue 核心包开发

场景：局部 模块改造

② Vue 核心包 & Vue 插件 工程化开发

场景：整站 开发

Vue 是什么

概念：Vue 是一个用于 构建用户界面^①的 渐进式^② 框架^③

优点：大大提升开发效率 (70% ↑)

缺点：需要理解记忆规则 → 官网

一套完整的项目解决方案



渐进式
JavaScript 框架



WHY VUE.JS?

起步



GITHUB

特别赞助



稀土掘金

来掘金与Vue同好者交流

成为特别赞助商

易用

已经会了 HTML、CSS、
JavaScript? 即刻阅读指南开始构
建应用!

灵活

不断繁荣的生态系统，可以在一
个库和一套完整框架之间自如伸
缩。

高效

20kB min+gzip 运行大小
超快虚拟 DOM
最省心的优化



总结

Vue是什么?

Vue 是一个用于 构建用户界面 的 渐进式 框架

1. 构建用户界面：基于 数据 动态 渲染 页面
2. 渐进式：循序渐进的学习
3. 框架：一套完整的项目解决方案，提升开发效率↑ (理解记忆规则)

规则 → 官网



目录

Contents

◆ Vue 快速上手

Vue 概念 / 创建实例 / 插值表达式 / 响应式特性 / 开发者工具

◆ Vue 指令

v-html / v-show / v-if / v-else / v-on / v-bind / v-for / v-model

◆ 综合案例 - 小黑记事本

列表渲染 / 删除功能 / 添加功能 / 底部统计 / 清空

创建 Vue 实例，初始化渲染

构建用户界面

创建 Vue 实例
初始化渲染

核心步骤 4步:

1. 准备容器
2. 引包 (官网) - 开发版本 / 生产版本
3. 创建 Vue 实例 new Vue()
4. 指定配置项 → 渲染数据
 - ① el 指定挂载点
 - ② data 提供数据

Hello 黑马

msg: 'Hello 黑马'

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

<script src="https://cdn.jsdelivr.net/npm/vue@2.7.10/dist/vue.js"></script>


<script>
  const app = new Vue({
    el: '#app',
    data: {
      msg: 'Hello 黑马'
    }
  })
</script>
```



总结

创建 Vue 实例，初始化渲染的核心步骤：

1. 准备容器
2. 引包 (官网) - 开发版本 / 生产版本
3. 创建 Vue 实例 `new Vue()`
4. 指定配置项 `el data =>` 渲染数据
 - ① `el` 指定挂载点，选择器指定控制的是哪个盒子
 - ② `data` 提供数据



目录

Contents

◆ Vue 快速上手

Vue 概念 / 创建实例 / 插值表达式 / 响应式特性 / 开发者工具

◆ Vue 指令

v-html / v-show / v-if / v-else / v-on / v-bind / v-for / v-model

◆ 综合案例 - 小黑记事本

列表渲染 / 删除功能 / 添加功能 / 底部统计 / 清空

插值表达式 {{ }}

插值表达式是一种 Vue 的模板语法

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

```
data: {  
  msg: 'Hello 黑马'  
}
```



Hello 黑马

插值表达式 {{ }}

插值表达式是一种 Vue 的模板语法

1. 作用：利用表达式进行插值，渲染到页面中

表达式：是可以被求值的代码，JS引擎会将其计算出一个结果

```
money + 100  
money - 100  
money * 10  
money / 10
```

```
price >= 100 ? '真贵' : '还行'  
obj.name  
arr[0]  
fn()  
obj.fn()
```

插值表达式 {{ }}

插值表达式是一种 Vue 的模板语法

1. 作用：利用表达式进行插值，渲染到页面中

表达式：是可以被求值的代码，JS引擎会将其计算出一个结果

2. 语法：{{ 表达式 }}

```
<h3>{{ title }}</h3>
<p>{{ nickname.toUpperCase() }}</p>
<p>{{ age >= 18 ? '成年' : '未成年' }}</p>
<p>{{ obj.name }}</p>
```

3. 注意点：

(1) 使用的数据必须存在 (data)

```
<p>{{ hobby }}</p>
```

(2) 支持的是表达式，而非语句，比如：if for ...

```
<p>{{ if }}</p>
```

(3) 不能在标签属性中使用 {{ }} 插值

```
<p title="{{ username }}">我是p标签</p>
```



总结

1. 插值表达式的作用是什么?

利用表达式进行插值，将数据渲染页面中

2. 语法格式?

`{{ 表达式 }}`

3. 插值表达式的注意点:

- ① 使用的数据要存在 (data)
- ② 支持的是表达式，而非语句 if ... for
- ③ 不能在标签属性里面使用



目录

Contents

◆ Vue 快速上手

Vue 概念 / 创建实例 / 插值表达式 / 响应式特性 / 开发者工具

◆ Vue 指令

v-html / v-show / v-if / v-else / v-on / v-bind / v-for / v-model

◆ 综合案例 - 小黑记事本

列表渲染 / 删除功能 / 添加功能 / 底部统计 / 清空

Vue 核心特性：响应式

我们已经掌握了基础的模板渲染，其实除了基本的模板渲染，Vue背后还做了大量工作。

比如：数据的响应式处理 → 响应式：数据变化，视图自动更新

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

```
const app = new Vue({
  el: '#app',
  data: {
    msg: 'Hello 黑马'
  }
})
```

响应式数据



Hello Vue
~~Hello 黑马~~

如何访问 or 修改？data中的数据，最终会被添加到实例上

① 访问数据： "实例.属性名"

② 修改数据： "实例.属性名" = "值"

Vue 核心特性：响应式

数据改变，视图会自动更新



聚焦于数据 → 数据驱动视图

使用 Vue 开发，关注业务的核心逻辑，根据业务修改数据即可



总结

1. 什么是响应式呢?

数据改变，视图自动更新

使用 Vue 开发 → 专注于业务核心逻辑 即可

2. 如何访问或修改数据呢?

data中的数据, 最终会被添加到实例上

① 访问数据: "实例.属性名"

② 修改数据: "实例.属性名" = "值"



目录

Contents

◆ Vue 快速上手

Vue 概念 / 创建实例 / 插值表达式 / 响应式特性 / **开发者工具**

◆ Vue 指令

v-html / v-show / v-if / v-else / v-on / v-bind / v-for / v-model

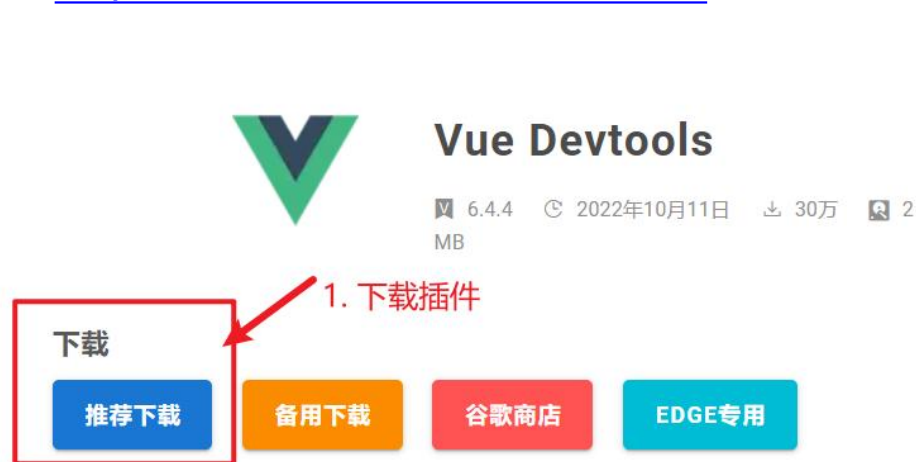
◆ 综合案例 - 小黑记事本

列表渲染 / 删除功能 / 添加功能 / 底部统计 / 清空

安装 Vue 开发者工具：装插件调试 Vue 应用

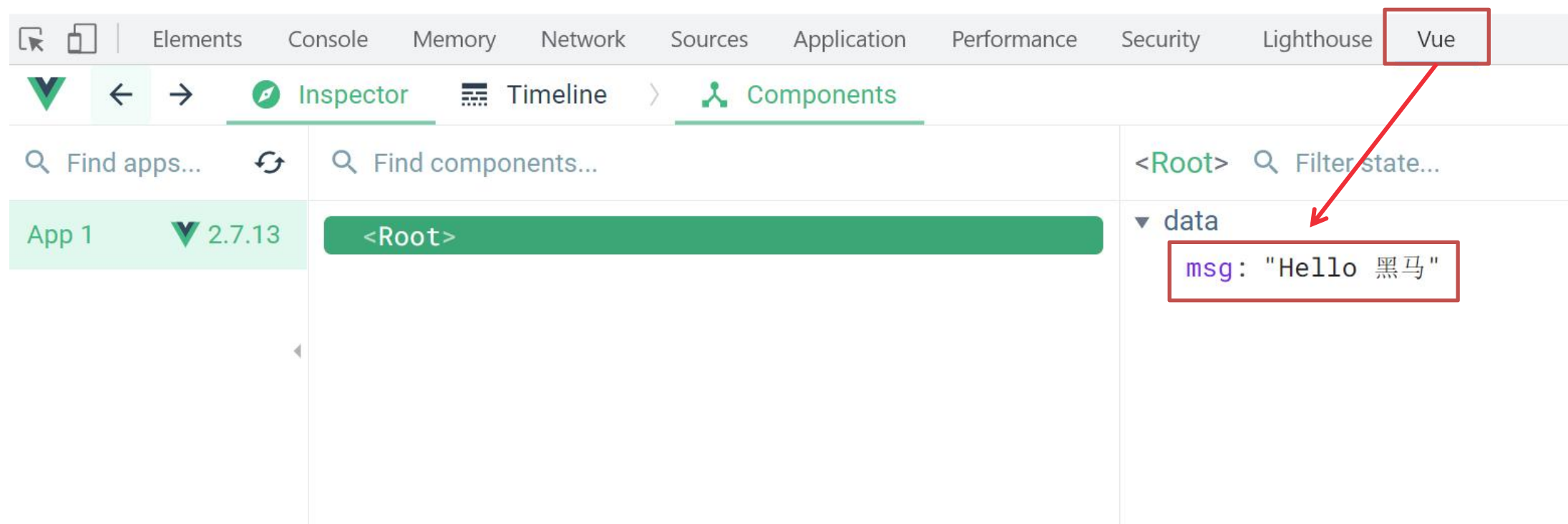
- (1) 通过谷歌应用商店安装（国外网站）
- (2) 极简插件: 下载 → 开发者模式 → 拖拽安装 → 插件详情允许访问文件

<https://chrome.zzzmh.cn/index>



安装 Vue 开发者工具：装插件调试 Vue 应用

打开 Vue 运行的页面，调试工具中 **Vue** 栏，即可查看修改数据，进行调试。





目录

Contents

◆ Vue 快速上手

Vue 概念 / 创建实例 / 插值表达式 / 响应式特性 / 开发者工具

◆ Vue 指令

v-html / v-show / v-if / v-else / v-on / v-bind / v-for / v-model

◆ 综合案例 - 小黑记事本

列表渲染 / 删除功能 / 添加功能 / 底部统计 / 清空

Vue 指令

Vue 会根据不同的【指令】，针对标签实现不同的【功能】

指令：带有 v- 前缀 的特殊 标签属性

```
<!-- Vue 指令: v- 前缀的标签属性 -->
<div v-html="str"></div>
```

v-html:

作用：设置元素的 innerHTML

语法：v-html = "表达式"

```
<!-- 普通标签属性 -->
<div class="box"> ... </div>
<div title="小张"> ... </div>
```



总结

1. 什么是 Vue 指令呢?

指令就是带有 **v- 前缀** 的特殊 **属性**，不同属性 **对应** 不同的功能

学习不同指令 → **解决不同业务场景需求**

2. 如果需要动态解析标签，可以用哪个指令？语法？

`v-html = "表达式"` → 动态设置元素 **innerHTML**

Vue 指令 v-show vs v-if

v-show

1. 作用： 控制元素显示隐藏
2. 语法： v-show = "表达式" 表达式值 true 显示， false 隐藏
3. 原理： 切换 display:none 控制显示隐藏
4. 场景： 频繁切换显示隐藏的场景



v-if

1. 作用： 控制元素显示隐藏（条件渲染）
2. 语法： v-if = "表达式" 表达式值 true 显示， false 隐藏
3. 原理： 基于条件判断，是否 创建 或 移除 元素节点
4. 场景： 要么显示，要么隐藏，不频繁切换的场景



Vue 指令 v-else v-else-if

1. 作用：辅助 v-if 进行判断渲染
2. 语法：v-else v-else-if = "表达式"
3. 注意：需要紧挨着 v-if 一起使用

♂ 男

♀ 女

成绩评定A：奖励电脑一台

成绩评定B：奖励周末郊游

成绩评定C：奖励零食礼包

成绩评定D：惩罚一周不能玩手机

性别	性别标识
男	1
女	2

序号	成绩等级	考试分数
1	A	90分+
2	B	70-90分
3	C	60-70分
4	D	60分以下

Vue 指令 v-on

1. 作用： 注册事件 = 添加监听 + 提供处理逻辑

2. 语法：

① v-on:事件名 = "内联语句"

② v-on:事件名 = "methods中的函数名"

3. 简写：@事件名

```
<button v-on:click="count++">按钮</button>
```

```
<button @click="count++">按钮</button>
```

Vue 指令 v-on

1. 作用： 注册事件 = 添加监听 + 提供处理逻辑

2. 语法：

① v-on:事件名 = "内联语句"

② v-on:事件名 = "methods中的函数名"

3. 简写：@事件名

```
<button @click="fn">-</button>
```

4. 注意：methods函数内的 this 指向 Vue 实例

```
const app = new Vue({  
  el: '#app',  
  data: {  
    // 提供数据  
    count: 100  
  },  
  methods: {  
    // 提供处理逻辑函数  
    fn () {  
      console.log('提供逻辑代码')  
    }  
  }  
})
```

Vue 指令 v-on 调用传参

```
<button @click="fn">  
  按钮  
</button>
```

```
const app = new Vue({  
  el: '#app',  
  methods: {  
    fn () {  
      console.log('这是一个fn函数')  
    }  
  }  
})
```

Vue 指令 v-on 调用传参

```
<button @click="fn()">  
  按钮  
</button>
```

```
const app = new Vue({  
  el: '#app',  
  methods: {  
    fn () {  
      console.log('这是一个fn函数')  
    }  
  }  
})
```


Vue 指令 v-on 调用传参

```
<button @click="fn(参数1,参数2)">  
  按钮  
</button>
```



```
const app = new Vue({  
  el: '#app',  
  methods: {  
    fn () {  
      console.log('这是一个fn函数')  
    }  
  }  
})
```

Vue 指令 v-on 调用传参

```
<button @click="fn(参数1, 参数2)">  
  按钮  
</button>
```

```
const app = new Vue({  
  el: '#app',  
  methods: {  
    fn (a, b) {  
      console.log('这是一个fn函数')  
    }  
  }  
})
```

小黑自动售货机

可乐5元

咖啡10元

银行卡余额: 100元

Vue 指令 v-bind

1. 作用： 动态的设置html的**标签属性** → src url title ...
2. 语法： v-bind:**属性名**="表达式"
3. 注意： 简写形式 **:属性名**="表达式"

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

```
const app = new Vue({  
  el: '#app',  
  data: {  
    url: '图片路径'  
  }  
})
```



图片切换案例-波仔学习之旅

上一页



图片切换案例-波仔学习之旅



核心思路分析:

① 数组存储图片路径 → [图片1, 图片2, 图片3, ...]

② 准备下标 index, 数组[下标] → v-bind 设置 src 展示图片 → 修改下标切换图片

Vue 指令 v-for

1. 作用： 基于数据循环，多次渲染整个元素 → 数组、对象、数字...

```
<p v-for="...">我是一个内容</p>
```

```
<p>我是一个内容</p>
<p>我是一个内容</p>
<p>我是一个内容</p>
```

2. 遍历数组语法：

v-for = "(item, index) in 数组"

➤ item 每一项，index 下标

➤ 省略 index: v-for = "item in 数组"

图书管理案例 - 小黑的书架

明确需求：

- ① 基本渲染 → v-for
- ② 删除功能 → 用 filter 根据 id 从数组中删除对应项

```
booksList: [  
  { id: 1, name: '《红楼梦》', author: '曹雪芹' },  
  { id: 2, name: '《西游记》', author: '吴承恩' },  
  { id: 3, name: '《水浒传》', author: '施耐庵' },  
  { id: 4, name: '《三国演义》', author: '罗贯中' }  
]
```

小黑的书架

- 《红楼梦》 曹雪芹
- 《西游记》 吴承恩
- 《水浒传》 施耐庵
- 《三国演义》 罗贯中

v-for 中的 key

语法：key属性 = "唯一标识"

作用：给列表项添加的**唯一标识**。便于Vue进行列表项的**正确排序复用**。

```
<ul>
  <li v-for="(item, index) in booksList" :key="item.id">
    <span>{{ item.name }}</span>
    <span>{{ item.author }}</span>
    <button @click="del(item.id)">删除</button>
  </li>
</ul>
```

小黑的书架

- 《红楼梦》 曹雪芹 删除
- 《西游记》 吴承恩 删除
- 《水浒传》 施耐庵 删除
- 《三国演义》 罗贯中 删除

v-for 中的 key

key作用：给元素添加的**唯一标识**。

```
booksList: [  
  { id: 1, name: '《红楼梦》', author: '曹雪芹' },  
  { id: 2, name: '《西游记》', author: '吴承恩' },  
  { id: 3, name: '《水浒传》', author: '施耐庵' },  
  { id: 4, name: '《三国演义》', author: '罗贯中' }  
]
```

```
:key="item.id"
```

 红楼梦 曹雪芹  key = "1"

 西游记 吴承恩 key = "2"

 水浒传 施耐庵 key = "3"

三国演义 罗贯中 key = "4"

v-for 中的 key

key作用：给元素添加的**唯一标识**。

```
booksList: [  
  { id: 2, name: '《西游记》', author: '吴承恩' },  
  { id: 3, name: '《水浒传》', author: '施耐庵' },  
  { id: 4, name: '《三国演义》', author: '罗贯中' }  
]
```

```
:key="item.id"
```

 红楼梦 曹雪芹  key = "1"

 西游记 吴承恩 key = "2"

 水浒传 施耐庵 key = "3"

三国演义 罗贯中 key = "4"

v-for 中的 key

key作用：给元素添加的**唯一标识**。

```
booksList: [  
  { id: 2, name: '《西游记》', author: '吴承恩' },  
  { id: 3, name: '《水浒传》', author: '施耐庵' },  
  { id: 4, name: '《三国演义》', author: '罗贯中' }  
]
```

```
:key="item.id"
```

```
<li> 西游记 吴承恩 </li>
```

 key = "2"

```
<li> 水浒传 施耐庵 </li>
```

 key = "3"

```
<li>三国演义 罗贯中</li>
```

 key = "4"

v-for 中的 key - 不加 key

v-for 的默认行为会尝试 **原地修改元素**（就地复用）

```
booksList: [  
  { id: 2, name: '《西游记》', author: '吴承恩' },  
  { id: 3, name: '《水浒传》', author: '施耐庵' },  
  { id: 4, name: '《三国演义》', author: '罗贯中' }  
]
```

 红楼梦 曹雪芹 ❌

 西游记 吴承恩

 水浒传 施耐庵

 三国演义 罗贯中

v-for 中的 key - 不加 key

v-for 的默认行为会尝试 原地修改元素（就地复用）

```
booksList: [  
  { id: 2, name: '《西游记》', author: '吴承恩' },  
  { id: 3, name: '《水浒传》', author: '施耐庵' },  
  { id: 4, name: '《三国演义》', author: '罗贯中' }  
]
```

 西游记 吴承恩

 水浒传 施耐庵

 三国演义 罗贯中

v-for 中的 key

key作用：

给元素添加的**唯一标识**，便于Vue进行列表项的**正确排序复用**。

注意点：

1. key 的值只能是 **字符串** 或 **数字类型**
2. key 的值必须具有 **唯一性**
3. 推荐使用 **id** 作为 key（唯一），不推荐使用 **index** 作为 key（会变化，不对应）

```
<li v-for="(item, index) in xxx" :key="唯一值">
```

Vue 指令 v-model

1. 作用：给 表单元素 使用，双向数据绑定 → 可以快速 获取 或 设置 表单元素内容

① 数据变化 → 视图自动更新

② 视图变化 → 数据自动更新

2. 语法：v-model = '变量'

```
data: {  
  username: '',  
  password: '',  
},
```



用户:

密码:

登录

重置



目录

Contents

◆ Vue 快速上手

Vue 概念 / 创建实例 / 插值表达式 / 响应式特性 / 开发者工具

◆ Vue 指令

v-html / v-show / v-if / v-else / v-on / v-bind / v-for / v-model

◆ 综合案例 - 小黑记事本

列表渲染 / 删除功能 / 添加功能 / 底部统计 / 清空

综合案例 - 小黑记事本

功能需求:

- ① 列表渲染
- ② 删除功能
- ③ 添加功能
- ④ 底部统计 和 清空

小黑记事本

添加任务

1. 跑步锻炼20分钟

×

2. 复习数组语法

合计: 2

清空任务



总结

功能总结：

① 列表渲染：

v-for **key** 的设置 {{ }} 插值表达式

② 删除功能

v-on 调用传参 **filter** 过滤 覆盖修改原数组

③ 添加功能

v-model 绑定 **unshift** 修改原数组添加

④ 底部统计 和 清空

数组.length 累计长度

覆盖数组清空列表

v-show 控制隐藏



传智教育旗下高端IT教育品牌