

Vue 进阶属性

directives、mixins、extends、provide、inject

版权声明

本内容版权属杭州饥人谷教育科技有限公司（简称饥人谷）所有。

任何媒体、网站或个人未经本网协议授权不得转载、链接、转贴，或以其他方式复制、发布和发表。

已获得饥人谷授权的媒体、网站或个人在使用时须注明「资料来源：饥人谷」。

对于违反者，饥人谷将依法追究 responsibility。

联系方式

如果你想要购买本课程

请微信联系 [xiedaimala02](#) 或 [xiedaimala03](#)

如果你发现有人盗用本课程

请微信联系 [xiedaimala02](#) 或 [xiedaimala03](#)

回顾之前的一张 PPT

上面罗列了 options 里的所有属性

一、options 里面有什么

- 文档

- ✓ 英文文档里搜 options, 中文文档里搜选项
- ✓ 即可得到所有相关文档

- options 的五类属性

今天学习这五个黄色属性

- ✓ 数据: `data`、`props`、`propsData`、`computed`、`methods`、`watch`
- ✓ DOM: `el`、`template`、`render`、`renderError`
- ✓ 生命周期钩子: `beforeCreate`、`created`、`beforeMount`、`mounted`、`beforeUpdate`、`updated`、`activated`、`deactivated`、`beforeDestroy`、`destroyed`、`errorCaptured`
- ✓ 资源: `directives`、`filters`、`components`
- ✓ 组合: `parent`、`mixins`、`extends`、`provide`、`inject`
- ✓ 其他: 先不看

Directives

指令

自定义指令

- 我们已经学了一些内置指令
 - ✓ v-if v-for v-show v-html 等
- 今天学习如何自己造一个指令
 - ✓ 目标：造出 v-x，点击即出现一个 x

两种写法

- 一、声明一个全局指令

`Vue.directive('x', directiveOptions)`
你就可以在任何组件里用 `v-x` 了，[动手试试](#)

- 二、声明一个局部指令

```
new Vue({  
  ...,  
  directives: {  
    "x": directiveOptions  
  }  
})
```

`v-x` 只能用在该实例中，[动手试试](#)

- `directiveOptions` 里有哪些属性？

directiveOptions

- 五个函数属性

- ✓ bind(el, info, vnode, oldVnode) - 类似 created
- ✓ inserted(参数同上) - 类似 mounted
- ✓ update(参数同上) - 类似 updated
- ✓ componentUpdated(参数同上) - 用得不多, [见文档](#)
- ✓ unbind(参数同上) - 类似 destroyed

- 举例

- ✓ [自制 v-on2 指令, 模仿 v-on](#)

- 缩写

- ✓ directiveOptions 在某些条件下可以缩写为函数, 用得不多, 可以自行[看文档](#)

指令的作用

- 主要用于 DOM 操作

- ✓ Vue 实例/组件用于数据绑定、事件监听、DOM更新
- ✓ Vue 指令主要目的就是原生 DOM 操作

- 减少重复

- ✓ 如果某个 DOM 操作你经常使用，就可以封装为指令
- ✓ 如果某个 DOM 操作比较复杂，也可以封装为指令

mixins

混入

混入就是™复制

前端就喜欢把简单概念搞复杂

减少重复

- 类比

- ✓ directives 的作用是减少 DOM 操作的重复
- ✓ mixins 的作用是减少 data、methods、钩子的重复

- 场景描述

- ✓ 假设我们需要在每个组件上添加 name 和 time
- ✓ 在 created、destroyed 时，打出提示，并报出存活时间
- ✓ 一共有五个组件，请问你怎么做？
- ✓ 一、给每个组件添加 data 和钩子，共五次
- ✓ 二、或者使用 mixins 减少重复
- ✓ 未完成的示例
- ✓ 已完成的示例

mixins 技巧

- 选项智能合并

- ✓ 见文档

- Vue.mixin

- ✓ 见文档，不推荐使用

extends

继承

减少重复

- 与 mixins 同样的需求

- ✓ 不想要每次都写一个 mixins, 咋办?

- ✓ 你可以使用 Vue.extend 或 options.extends

```
const MyVue = Vue.extend({
  data(){ return {name:'',time:undefined} },
  created(){
    if(!this.name){console.error('no name!')}
    this.time = new Date()
  },
  beforeDestroy(){
    const duration = (new Date()) - this.time
    console.log(`${this.name}存活时间${duration}`)
  }
})
```

- ✓ 然后我们就可以使用 new MyVue(options) 了

extends 是比 mixins 更抽象一点的封装

如果你嫌写五次 mixins 麻烦，可以考虑 extends 一次
不过实际工作中用得很少

provide 和 inject

提供和注入

使用举例

- 需求

- ✓ 一键换肤功能：默认蓝色，可以切换为红色
- ✓ 文字大小：默认正常，可以改成大或小
- ✓ 未完成的示例
- ✓ 已完成的示例
- ✓ 祖先栽树(provide)，后人乘凉(inject)

- 总结

- ✓ 作用：大范围的 data 和 method 等共用
- ✓ 注意：不能只传 themeName 不传 changeTheme，因为 themeName 的值是被复制给 provide 的
- ✓ 思考题：传引用可以吗？可以，但是不推荐，因为容易失控

总结

- directives 指令

- ✓ 全局用 `Vue.directive('x', {...})`
- ✓ 局部用 `options.directives`
- ✓ 作用是减少 DOM 操作相关重复代码

- mixins 混入

- ✓ 全局用 `Vue.mixin({...})`
- ✓ 局部用 `options.mixins: [mixin1, mixin2]`
- ✓ 作用是减少 options 里的重复

- extends 继承/扩展

- ✓ 全局用 `Vue.extend({...})`
- ✓ 局部用 `options.extends: {...}`
- ✓ 作用跟 mixins 差不多，只是形式不同

- provide / inject 提供和注入

- ✓ 祖先提供东西，后代注入东西
- ✓ 作用是大范围、隔 N 代共享信息

options 基本讲完

Vue 基础知识告一段落，之后我们开始做点东西

如果你觉得 Vue 不适合你，也可以开始学习 React