

computed 和 watch

从入门到工作 - Vue 全解

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，饥人谷将依法追究法律责任。

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](#)

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

- ✓ 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

入门属性

- **el - 挂载点**

- ✓ 可以用 `$mount` 代替

- **data - 内部数据**

- ✓ 支持对象和函数，优先用函数

- **methods - 方法**

- ✓ 事件处理函数或者是普通函数

- **components**

- ✓ Vue 组件，注意大小写
- ✓ 三种引入方式，推荐最后一种

- **四个钩子**

- ✓ `created` - 实例出现在内存中
- ✓ `mounted` - 实例出现在页面中
- ✓ `updated` - 实例更新了
- ✓ `destroyed` - 实例从页面和内存中消亡了

- **props - 外部数据**

- ✓ 也叫属性
- ✓ `message="n"` 传入字符串
- ✓ `:message="n"` 传入 `this.n` 数据
- ✓ `:fn="add"` 传入 `this.add` 函数

进阶属性

- **computed - 计算属性**

- ✓ 不需要加括号
- ✓ 它会根据依赖是否变化来缓存

- **watch - 侦听**

- ✓ 一旦 data 变化，就执行的函数
- ✓ options.watch 用法
- ✓ this.\$watch 用法
- ✓ deep, immediate 含义

- **directives - 指令**

- ✓ 内置指令 v-if / v-for / v-bind / v-on
- ✓ 自定义指令，如 v-focus
- ✓ 指令是为了减少重复的 DOM 操作

- **mixin - 混入**

- ✓ 重复三次之后的出路
- ✓ 混入 v.s. 全局混入
- ✓ 选项自动合并
- ✓ 混入就是为了减少重复的构造选项

- **extends - 继承**

- ✓ 先了解一下 Vue.extend
- ✓ 你觉得用了 mixin 还是重复
- ✓ 于是你自己写了一个 View，它继承 Vue
- ✓ 你还可以预先定义其他构造选项
- ✓ 继承就是为了减少重复的构造选项
- ✓ 那为什么不用 ES 6 的 extends 呢？

- **provide/inject**

- ✓ 爷爷想和孙子讲话怎么办
- ✓ 祖宗想跟它的所有后代讲话怎么办
- ✓ 答案是全局变量，但是全局变量太 low
- ✓ 所以我们需要局部的全局变量

复习一下响应式原理

- options.data

- ✓ 会被 Vue 监听
- ✓ 会被 Vue 实例代理
- ✓ 每次对 data 的读写都会被 Vue 监控
- ✓ Vue 会在 data 变化时更新 UI

- 这节课讲

- ✓ data 变化时除了更新 UI，还能做点啥？

Computed

计算属性

computed - 计算属性

- 用途

- ✓ 被计算出来的属性就是计算属性
- ✓ 例1: 用户名展示
- ✓ 例2: 列表展示

- 缓存

- ✓ 如果依赖的属性没有变化，就不会重新计算
- ✓ getter / setter 默认不会做缓存，Vue 做了特殊处理
- ✓ 如何缓存？看示例。这是示例，不代表 Vue 这样实现

Watch

监听 / 侦听

watch - 侦听

- 用途

- ✓ 当数据变化时，执行一个函数
- ✓ 例1: 撤销
- ✓ 例2: 模拟computed，说实话，这样做很傻

- 何谓变化？

- ✓ 看示例
- ✓ obj 原本是 {a:'a'}，现在 obj = {a:'a'}，请问
- ✓ obj 变了没有？ obj.a 变了没有
- ✓ 简单类型看值，复杂类型（对象）看地址
- ✓ 这其实就是 === 的规则

watch - 侦听 (续)

- 语法1 (推荐看[文档](#))

```
watch: {  
  o1: () => {}, // 别用这种, 这里的 this 是全局对象  
  // 很多前端学 JS 几年, 都没搞清楚 this, 忘了的同学  
  写博客总结, 我视频里讲过。必须搞清楚 this  
  o2: function(value, oldValue){},  
  o3(){},  
  o4: [f1, f2],  
  o5: 'methodName',  
  o6: {handler:fn, deep:true, immediate:true},  
  'object.a': function(){}  
}
```

- 语法2

```
vm.$watch('xxx', fn, {deep: .., immediate: ..})  
其中 'xxx' 可以改为一个返回字符串的函数
```

watch - 侦听 (续2)

- **deep: true 是干什么的?**

- ✓ 如果 `object.a` 变了, 请问 `object` 算不算也变了
- ✓ 如果你需要答案是「也变了」, 那么就用 `deep:true`
- ✓ 如果你需要答案是「没有变」, 那么就用 `deep:false`
- ✓ `deep` 的意思是, 监听 `object` 的时候是否往深了看

options 还没讲完

下节课见