第三课 计算属性

第三章 计算属性

3.1 什么是计算属性

> 我们已经可以搭建出一个简单的 Vue 应用，在模板中双向绑定一些数据或表达式了。但是表达式如果过长，或逻辑更为复杂时，就会变得臃肿甚至难以阅读和维护

```
<div>
    {{ text.split ( ',' ) .reverse () . join (',')}}
</div>
```

这里的表达式包含 3 个操作，并不是很清晰，所以在遇到复杂的逻辑时应该使用 计算属性

> 所有的计算属性都以函数的形式写在 Vue 实例内的**computed** 选项内，最终返回计算后的结果。
>
> ### 3.2 计算属性用法
>
> 在一个计算属性里可以完成各种复杂的逻辑，包括运算、函数调用等，只要最终返回一个结果就可以。除了上例简单的用法，计算属性还可以依赖多个 Vue 实例的数据，**只要其中任一数据变化，计算属性就会重新执行，视图也会更新**
>
> 实例 :展示两个购物车的物品总价

getter和setter

每一个计算属性都包含一个 getter 和一个 setter，我们上面的两个示例都是计算属性的默认用法 ， 只是利用了 getter 来读取。在你需要时，也可以提供一个 setter 函数 ， 当手动修改计算属性的值就像修改一个普通数据那样时，就会触发 setter函数，执行一些自定义的操作

计算属性除了上述简单的文本插值外，还经常用于动态地设置元素的样式名称 class 和内联样式 style,后边会

> **小技巧**：计算属性还有两个很实用的小技巧容易被忽略：一是计算属性可以依赖其他计算属性；
> 二是计算属性不仅可以依赖当前 Vue 实例的数据，还可以依赖其他实例的数据

3.3计算属性缓存

调用 methods 里的方法也可以与计算属性起到同样的作用

页面中的方法： 如果是调用方法，只要页面重新渲染。方法就会重新执行，不需要渲染，则不需要重新执行

计算属性：不管渲染不渲染，只要计算属性依赖的数据未发生变化，就永远不变

结论：没有使用计算属性，在 methods 里定义了一个方法实现了相同的效果，甚至该方法还可以接受参数，使用起来更灵活。既然使用 methods 就可以实现，那么为什么还需要计算属性呢？原因就是

计算属性是基于它的依赖缓存的。 一个计算属性所依赖的数据发生变化时，它才会重新取值，所以text 只要不改变，计算属性也就不更新

何时使用：----------使用计算属性还是 methods 取决于你是否需要缓存，当遍历大数组和做大量计算时，应当使用

计算属性，除非你不希望得到缓存。