文档说明

2019-9-8 21:37:46

目录

回顾上节课内容

2019-9-11 23:20:22

- 第一章课件

https://github.com/username2099/vue-course

- vue 的特点
  - P2
- 复习课本 MVVM
  - 第一章课件

Vue 的模板语法

预习 vue 模板语法（8）

- 书本
- 官方文档
  - https://cn.vuejs.org/v2/guide/syntax.html
- 回答
  - 什么是模板语法?

模板语法

将 {{}} 中的文本数据放入 DOM 中

在底层的实现上，Vue 将模板编译成虚拟 DOM 渲染函数。结合响应系统，Vue 能够智能地计算出最少需要重新渲染多少组件，并把 DOM 操作次数减到最少。

使用表达式

```
<!DOCTYPE html>
<!--声明语言-->
<html lang="en">
<!--头-->
<head>
    <!-- 声明字符编码 -->
    <meta charset="UTF-8">
    <!--  标题 显示位置  -->
    <title>vue</title>
```

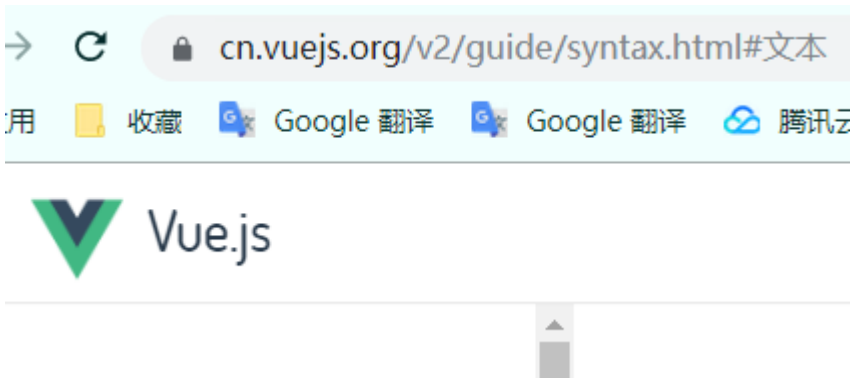
```
</head>
<!--内容-->
<body>

<!--视图  view-->
<div id="hello">

    <!--    v-model 双向数据绑定 -->
    <input type="text"  v-model="username" />
    <br/>
    <h1>花括号</h1>
    {{username}}
    <br/>
    <h1>V-HTML</h1>
    <p v-html="username"></p>
    <h1>表达式</h1>
    <p>转换为大写: {{username.toUpperCase()}}</p>
    <p>三元表达式: {{""==username ? "NULL" : username}}</p>
    <!-- 解释每一步 .split('') - reverse() - join('')  -->
    <p>字符串反转: {{username.split('').reverse().join('')}}</p>

</div>
<!--视图  view 模板页面 end -->

</body>

<!--引入 vue-->
<script src="https://cdn.jsdelivr.net/npm/vue"></script>
<script type="text/javascript">

    <!--    视图模型 vm viewmodel -->
    let vm=new Vue({
        // element 元素选择器 # id
        el:"#hello",
        // 数据对象 模型
        data:{
            username:"username"
        }
    })

</script>

</html>
```

练习

- 思考 如何把 username 原样放到 中

属性绑定

- 如何提取到图片的地址？



Chrome 开发者工具

有没有更简单的方法

```
<img src="https://cn.vuejs.org/images/logo.png"/>
<img src="imgUrl"/>
```

- 图片为什么不能正常显示（reason），怎么解决？

  - v-bind:

✧ ：

缩写

`v-` 前缀作为一种视觉提示，用来识别模板中 Vue 特定的特性。当你在使用 Vue.js 为现有标签添加动态行为 (dynamic behavior) 时， `v-` 前缀很有帮助，然而，对于一些频繁用到的指令来说，就会感到使用繁琐。同时，在构建由 Vue 管理所有模板的**单页面应用程序 (SPA - single page application)** 时， `v-` 前缀也变得没那么重要了。因此，Vue 为 `v-bind` 和 `v-on` 这两个最常用的指令，提供了特定简写：

`v-bind` 缩写

```HTML
<!-- 完整语法 -->
<a v-bind:href="url">...</a>

<!-- 缩写 -->
<a :href="url">...</a>
```

自己敲代码练习（5）

发布原代码（5）

提升练习 1（8）

class1

输入

class1

字体颜色

初始源代码

```
<!DOCTYPE html>
<!--声明语言-->
<html lang="en">
<!--头-->
<head>
    <!-- 声明字符编码 -->
    <meta charset="UTF-8">
    <!--  标题 显示位置  -->
    <title>vue</title>

</head>
<!--内容-->
<body>

<!--视图  view-->
<div id="hello">


    <!--    v-model 双向数据绑定 -->
    <input type="text"  v-model="myClass" />
    <br/>
    <h1>输入</h1>
    {{myClass}}
    <br/>
    <p>字体颜色</p>


</div>
<!--视图  view 模板页面 end -->

</body>

<!--引入vue-->
<script src="https://cdn.jsdelivr.net/npm/vue"></script>
<script type="text/javascript">

    <!--    视图模型 vm viewmodel -->
    let vm=new Vue({
        // element 元素选择器 # id
        el:"#hello",
        // 数据对象 模型
        data:{
            myClass:""
        }
    })
```

```
</script>

</html>
```

提升练习 2（8）


```
<!DOCTYPE html>
<!--声明语言-->
<html lang="en">
<!--头-->
<head>
    <!-- 声明字符编码 -->
    <meta charset="UTF-8">
    <!--  标题 显示位置  -->
    <title>vue</title>
    <style>
        .class1{
            color:red;
        }
        .class2{
            color: green;
        }

        .class3{
            font-size: 30px;
        }

    </style>
</head>
<!--内容-->
<body>

<!--视图  view-->
<div id="hello">
```

```
    <!--    v-model 双向数据绑定 -->
    <input type="text"  v-model="myClass" />
    <br/>
    <h1>输入</h1>
    {{myClass}}
    <br/>
    <p v-bind:class="myClass" >字体颜色</p>

</div>
<!--视图  view 模板页面 end -->

</body>

<!--引入 vue-->
<script src="https://cdn.jsdelivr.net/npm/vue"></script>
<script type="text/javascript">

    <!--    视图模型 vm viewmodel -->
    let vm=new Vue({
        // element 元素选择器 # id
        el:"#hello",
        // 数据对象 模型
        data:{
            myClass:""
        }
    })
</script>
</html>
```

计算属性

- 预习计算属性
  - https://cn.vuejs.org/v2/guide/computed.html
  - 书上 P24-27

回到第一个问题，如何计算出输入的字符串的长度？

不能使用 `<p>字符串长度：{{username.length}}</p>`

11111111111

花括号

11111111111

V-HTML

11111111111

表达式

转换为大写：11111111111

三元表达式：11111111111

字符串反转：11111111111

输入字符串的长度：11

练习（8）

- 发布需要修改的代码

```
<!DOCTYPE html>
<!--声明语言-->
<html lang="en">
<!--头-->
<head>
    <!-- 声明字符编码 -->
    <meta charset="UTF-8">
    <!--  标题 显示位置  -->
    <title>vue 计算属性</title>
</head>
<!--内容-->
<body>

<!--视图  view-->
<div id="hello">

    <!--    v-model 双向数据绑定 -->
    <input type="text"  v-model="username" />
```

```
<br/>
<h1>花括号</h1>
{{username}}
<br/>
<h1>V-HTML</h1>
<p v-html="username"></p>

<h1>表达式</h1>
<p>转换为大写: {{username.toUpperCase()}}</p>
<p>三元表达式: {{""==username ? "NULL" : username}}</p>
<!-- 解释每一步 .split('') - reverse() - join('')  -->
<p>字符串反转: {{username.split('').reverse().join('')}}</p>
<p>输入字符串的长度[username.length]: {{username.length}}</p>
    <p>输入字符串的长度[计算属性]: {{  }}</p>
</div>
<!--视图  view 模板页面 end -->
</body>

<!--引入 vue-->
<script src="https://cdn.jsdelivr.net/npm/vue"></script>
<script type="text/javascript">
    <!--    视图模型 vm viewmodel -->
    let vm=new Vue({
        // element 元素选择器 # id
        el:"#hello",
        // 数据对象 模型
        data:{
            username:"username"
        }
    })
</script>
</html>
```

练习（5）

讲解源代码

Vue02-04.html

方法 methods

预习 methods（8）

- https://cn.vuejs.org/v2/guide/computed.html#%E8%AE%A1%E7%AE%97%E5%B1%9E%E6%80%A7%E7%BC%93%E5%AD%98-vs-%E6%96%B9%E6%B3%95
- 书上 P27-28 页

把上面的计算属性修改为 methods，实现计算字符串长度

讲解源代码

Vue02-05.html

方法属性 vs 计算属性

代码

```
// 计算属性
  computed:{
      usernameLength(){
          return Date.now();
      }
  },
  methods:{
      usernameLen:function(){
```

```
            return Date.now();
        }
    }
```

111

花括号

111

V-HTML

111

表达式

输入字符串的长度[username.length]：3

当前时间戳[计算属性]：1568559998613

当前时间戳[方法属性]：1568560006092

解释现象（5）

- 第一次执行时结果相同，后面更新时计算属性不发生改变（WHY？）

原因

- 计算属性是基于它们的响应式依赖进行缓存的
- 每当触发重新渲染时，调用方法将总会再次执行函数。

总结

- 计算属性是根据依赖自动执行的，methods 需要事件调用。

- 使用计算属性还是 methods 取决于是否需要缓存，当遍历大数组和做大量计算时，应当使用计算属性，除非不希望得到缓存。
- 将同一函数定义为一个方法或是一个计算属性，不同的是计算属性是基于它们的依赖进行缓存的。只在相关依赖发生改变时它们才会重新求值。相比之下，每当触发重新渲染时，调用方法将总会再次执行函数。

计算属性的 Watch 与 setter

自学并修改源代码（15）

- 要求
  - 使用 watch
  - 使用计算属性里的 get 和 set

```
<!DOCTYPE html>
<!--声明语言-->
<html lang="en">
<!--头-->
<head>
    <!-- 声明字符编码 -->
    <meta charset="UTF-8">
    <!--  标题 显示位置  -->
    <title>vue 计算属性</title>
</head>
<!--内容-->
<body>

<!--视图  view-->
<div id="hello">

    <h1>vue 计算属性</h1>

    姓名: <input type="text"  v-model="username" />
```

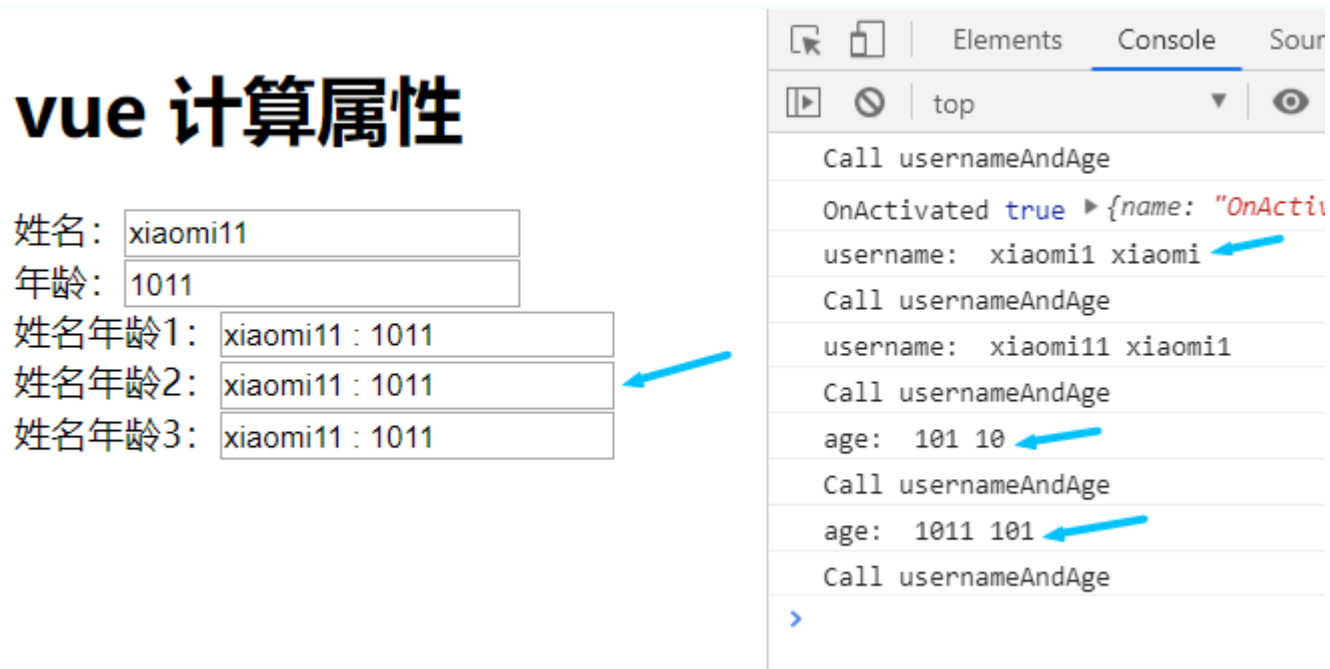
```
<br/>
年龄: <input type="text"  v-model="age" />
<br/>
姓名年龄 1: <input type="text"  v-model="usernameAndAge" />
<br/>

<!-- 参考 https://cn.vuejs.org/v2/guide/computed.html -->

<!-- 使用 watch -->
姓名年龄 2: <input type="text"  v-model="usernameAndAge" />
<br/>
<!-- 使用 计算属性的 setter 和 getter -->
姓名年龄 3: <input type="text"  v-model="usernameAndAge" />
```

```
</div>
<!--视图  view 模板页面 end -->

</body>

<!--引入 vue-->
<script src="https://cdn.jsdelivr.net/npm/vue"></script>
<script type="text/javascript">

    <!--    视图模型 vm viewmodel -->
    let vm=new Vue({
        // element 元素选择器 # id
        el:"#hello",
        // 数据对象 模型
    data:{
        username:"xiaomi",
        age:"10"
    },
    // 计算属性
    computed:{
        // 单向绑定
        // call usernameAndAge
        // 初始化时 or 相关的 data 发生改变
        usernameAndAge(){
            console.log("Call usernameAndAge");
            return this.username+" : "+this.age;
        }
    }

})
```

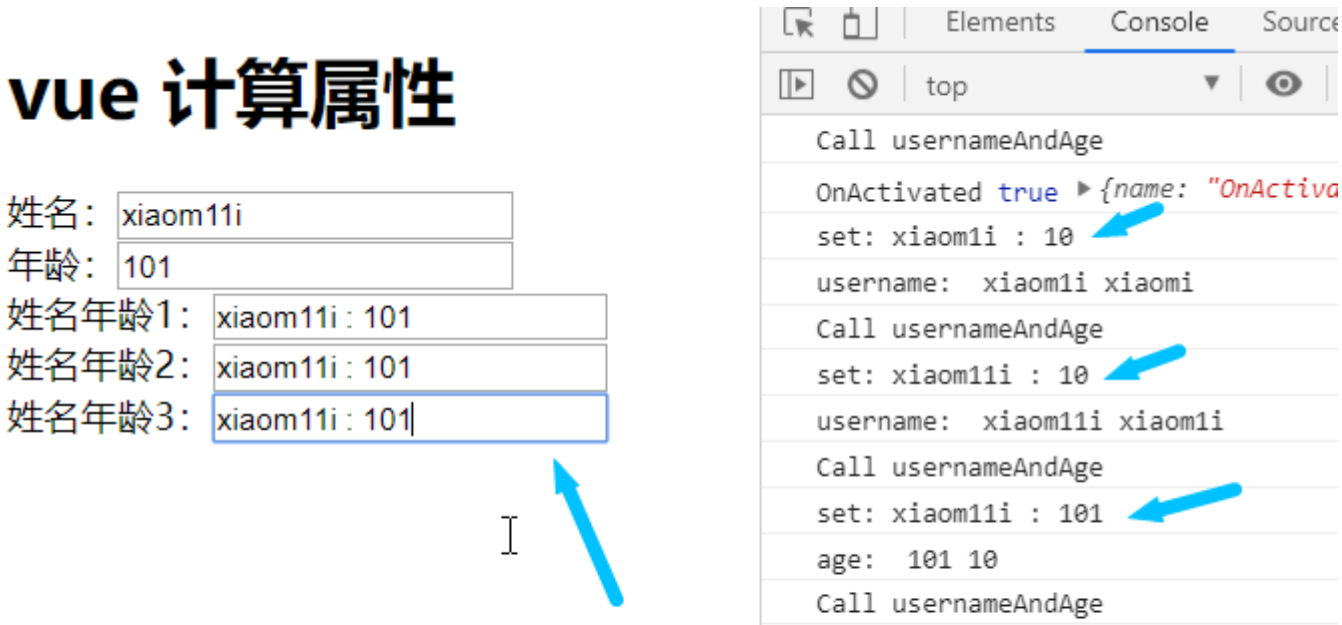
```
</script>

</html>
```

初步效果

Watch



Setter & getter



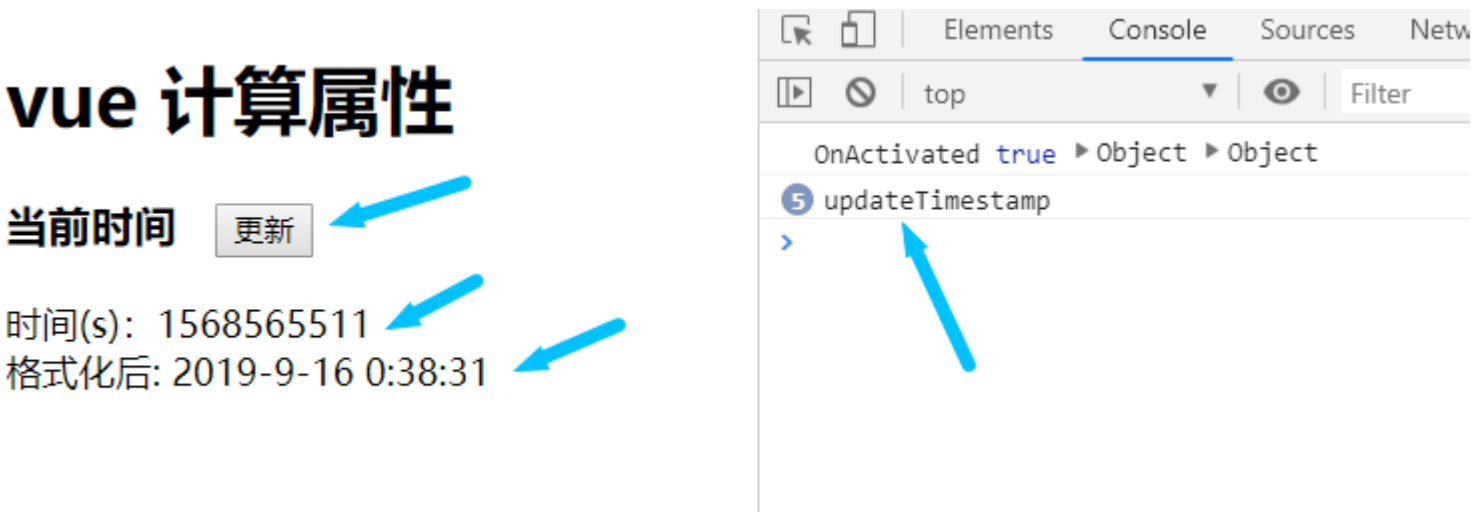
发布源代码，练习（8）

练习

使用计算属性，综合所学知识，实现日期格式化

- 发布 vue02-08.html

实现效果（10）



讲解

- vue02-08-1.html
- 发布 Vue02-08-2.html

提升 1（8）

在源代码的基础上，实现点击一次更新，接下来时间自己会主动更新。(演示)

- （4）提示使用 setTimeout
- 参考 https://www.w3school.com.cn/tiy/t.asp?f=js_timing_settimeout_2

讲解

Vue02-08-2.html

- 发布源代码 Vue02-08-2.html

提升 2（8）

在原来代码的基础上，打开页面后无需任何点击，自动实现时间更新。

（4）示范执行时打出 console.log(),所以只需要在初始化时调用到此函数即可。

讲解