Vue键盘修饰符&自定义指令

软通大学
ISOFTSTONE UNIVERSITY

本节目标

- 掌握常用键盘修饰符的使用
- 掌握自定义vue指令

目录 CONTENTS

01 vue键盘修饰符

vue键盘修饰符

在网页中，有的时候会有这种需求，监听键盘按键抬起响应事件，比如：上述例子中，当用户输入了id和name之后，不通过添加按钮添加一个品牌，而是name输入框按下enter键完成添加;

Vue.js官网中：https://cn.vuejs.org/v2/guide/events.html#按键修饰符， 对于按键修饰符有了说明

```
<label>
      Name:
       <!-- keyup监听键盘抬起事件:enter表示监听enter键抬起 -->
        <input type="text" class="form-control" v-model="name" @keyup.enter="add()">
</label>
```

*:Vue为键盘的常用按键设置了别名, 在keyup事件中，不需要记住这个键的keycode，直接使用别名就可以

.enter . tab
.delete .esc .space
.up .down .left . right

vue键盘修饰符

自定义键盘:如果默认没有我们想要监听的按键，也可以使用它对应的keycode或者为它自定义一个别名

- <!-- 113对应键盘F2 -->
  <input type="text" class="form-control" v-model="name" @keyup.113="add()">

- //自定义键盘修饰符
  Vue.config.keyCodes.f2=113
  //<!-- 113对应键盘F2 -->
  <input type="text" class="form-control" v-model="name" @keyup.f2="add()">

*: 注意的是，有些按键可能是会被系统响应的，比如浏览器的F1, Tab键，那么当定义这些按键的时候，事件会被捕获，也就是先响应系统的，即使是vue内置的键也是一样,所以当真有自定义键盘按键的时候，对于按键的选择就需要有一些甄别了
键盘key Code对照表: 自行百度即可

02 vue自定义指令

vue自定义指令

有的情况下，vue提供的指令不能实现需求，比如：当页面加载时，该元素自动获得焦点 ，Vue允许我们自定义指令，同样分为全局和私有

全局自定义指令： Vue.directive('指令名称',{}) , 比如: v-focus 指令: Vue.directive("focus",{...})

参数1:指令的名称，在定义的时候，指令的名称前面，不需要加 v- 前缀，在调用的时候，必须 在指令名称前 加上 v- 前缀来进行调用
参数2:是一个对象，这个对象身上，有一些指令相关的钩子函数，这些函数在特定的阶段执行, 要记住的是，每一个钩子函数的第一个参数表示绑定了这个指令的dom元素，**是一个原生js对象**

```
Vue.directive("focus", {
    bind:function(el){...},
    inserted:function(el){...},
    update: function(el){...},
    componentUpdated:function(el){...},
    unbind:function(el){...}
})
```

- **bind**:每当指令绑定到元素上的时候，会立即执行这个 bind 函数，只执行一次，一般用来做一些准备初始化工作
- **inserted**:表示元素 插入到DOM完成之后，会立即调用，执行一次
- **update**:当VNode更新的时候，会执行 update， 可能会触发多次，也就是绑定了Vue指令的dom元素发生改变时,这个改变包括很多，比如：样式、值、位置等等，只要这个元素和vue进行了数据绑定
- **componentUpdated**: 同update一样，当VNode更新时, 但在它的回调里面，el是改变后的dom
- **unbind**：元素从DOM删除时触发.(仅仅是删除)

vue自定义指令

bind与inserted钩子函数说明:

bind函数里调用focus是无效的，这个钩子函数表示指令被帮到元素上，但是还没有被插入到dom树中去，任何元素，之后再插入到dom之后，才能获取焦点.在实际运用中，通常我们用到最多的钩子函数只有bind和inserted, 一个是样式操作bind,一个是js操作inserted

结论:不需要操作dom，不需要等待dom完成插入，所以和样式相关的操作放在bind内,要操作dom元素，必须dom完成插入后，所以和js相关的操作放在inserted内

案例: 自定义指令v-color,完成指定元素内容颜色的功能?

```
Vue.directive("color", {
   bind: function (el, binding) {
      el.style.color = binding.value
   }
})
// 这里要注意指令值的，因为style.color='red'
<input type="text" class="form-control" v-model="keywords" id="search" v-focus  v-color="'green'">
```

vue自定义指令

Binding对象，包含以下属性

- name：指令名，不包括 v- 前缀
- **value**：指令的绑定值
- oldValue：指令绑定的前一个值
- expression：字符串形式的指令表达式，没啥用
- arg：传给指令的修饰参数，可选。例如 v-my-directive:foo 中，参数为 "foo"，
  比如： v-color.dc=“”“”,当没有指定颜色的时候，通过它可以设置默认颜色
- modifiers：一个包含修饰符的对象。例如：v-my-directive.foo.bar 中，修饰符对象为 { foo: true, bar: true }
  它和arg类似，将修饰参数封装成了一个对象，开发者可以为指令添加多个修饰符参数，并自定义它们的作用，通过这个属性获取。

vue自定义指令

私有自定义指令:

私有的就是在vm实例上添加directives属性即可。比如: 如下完成自定义指令指定元素字体粗细和字体大小?

```
directives: { // 自定义私有指令
    'fontweight': { // 设置字体粗细的
      bind: function (el, binding) {
        el.style.fontWeight = binding.value
      }
    },
    'fontsize': function (el, binding) { // 注意： 这个 function 等同于 把 代码写到了 bind 和 update 中去
      el.style.fontSize = parseInt(binding.value) + 'px'
    }
  }

  <p v-fontweight="800" v-fontsize="28" v-color="'yellow'">这是自定义的私有指令，改变字体大小和粗细</p>
```

本节总结

- Vue键盘修饰符、自定义键盘修饰符
- Vue自定义指令的定义
- Vue自定义指令的bind和inserted钩子函数

本节练习

◆ 课堂案例：完成自定义指令v-focus光标聚焦、v-color指定文本颜色、v-fontSize指定字体大小?


THANK YOU
软通大学
ISOFTSTONE UNIVERSITY