1、关于v-on事件传值：

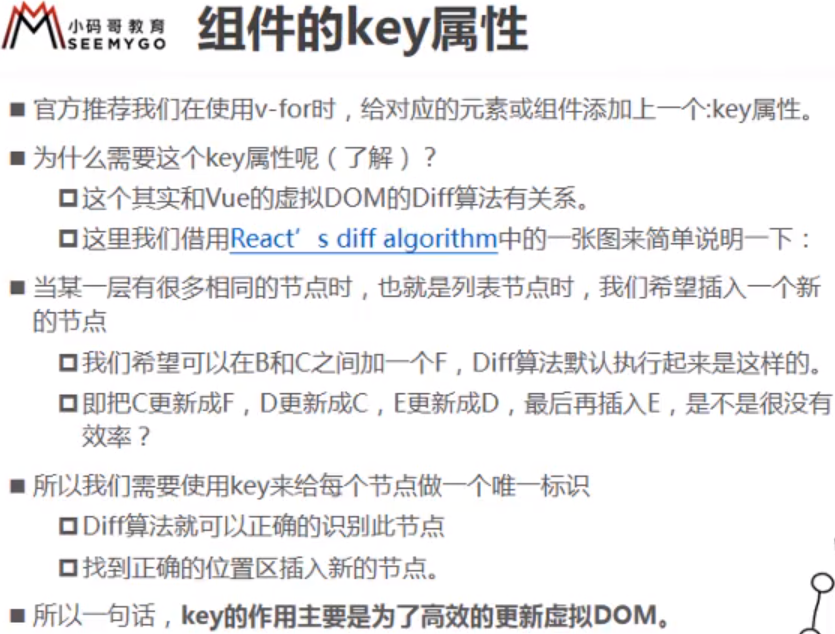
|  |  |
| --- | --- |
| html | js(点击按钮后) |
| @click="btn1Click() | btn1Click(){console.log(event);}  // MouseEvent {…} |
| @click="btn1Click() | btn1Click($event){console.log(event);}  // MouseEvent {…} |
| @click="btn1Click('abc') | btn1Click(a){ console.log(a,event); }  //abc MouseEvent {…} |
| @click="btn1Click($event) | btn1Click(e){ console.log(e); }  // MouseEvent {…} |
| @click="btn1Click('abc',$event) | btn1Click(a,e){ console.log(a,e); }  //abc MouseEvent {…} |
| @click="btn1Click('abc') | btn1Click($event){ console.log(event); }  // MouseEvent {…} |
|  |  |

总结：一般显式的调用event或者$event，都会把事件传进去。

2、关于input标签已输入内容的问题：

两个输入框切换的时候，已输入的内容是不会清空的，这时候如果想清空就加入一个key属性，让vue知道这是两个不一样的输入框。

3、性能优化：



4、vue中数组的响应式方法：

pop()

shift()

unshift()

splice()

sort()

reverse()

但是通过数组下标修改数组内元素时，不会响应

5、组件里模板的分离写法：

1. <script type="text/x-template" id="c">

<h2>qwe</h2>

</script>

2.<template id="c">

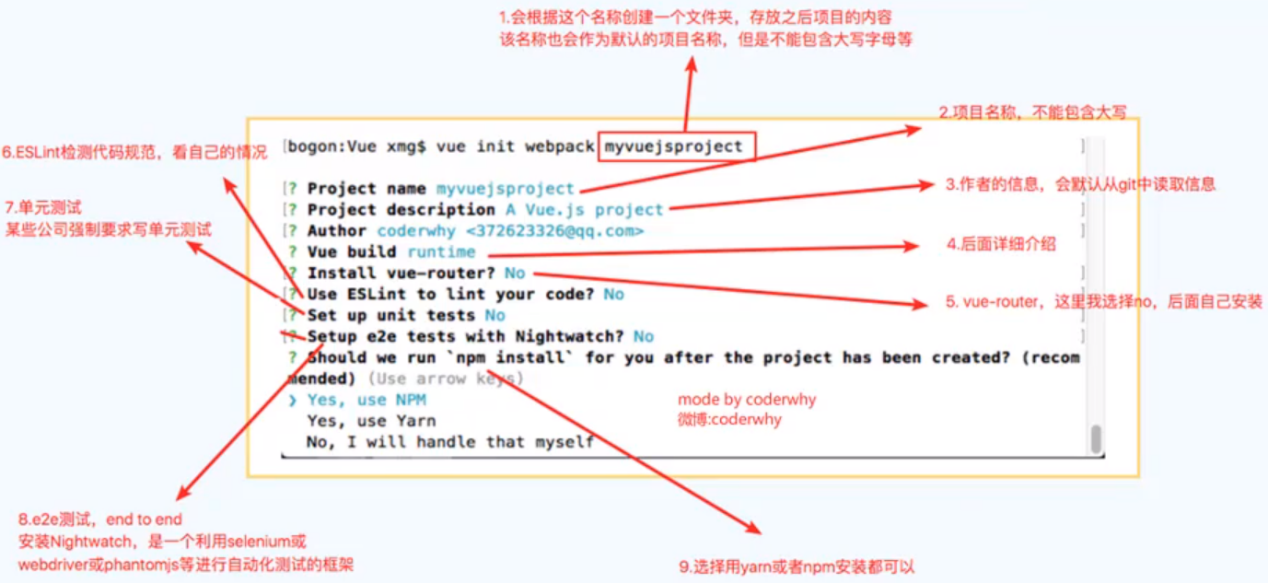
<h2>qwe</h2>

</template>

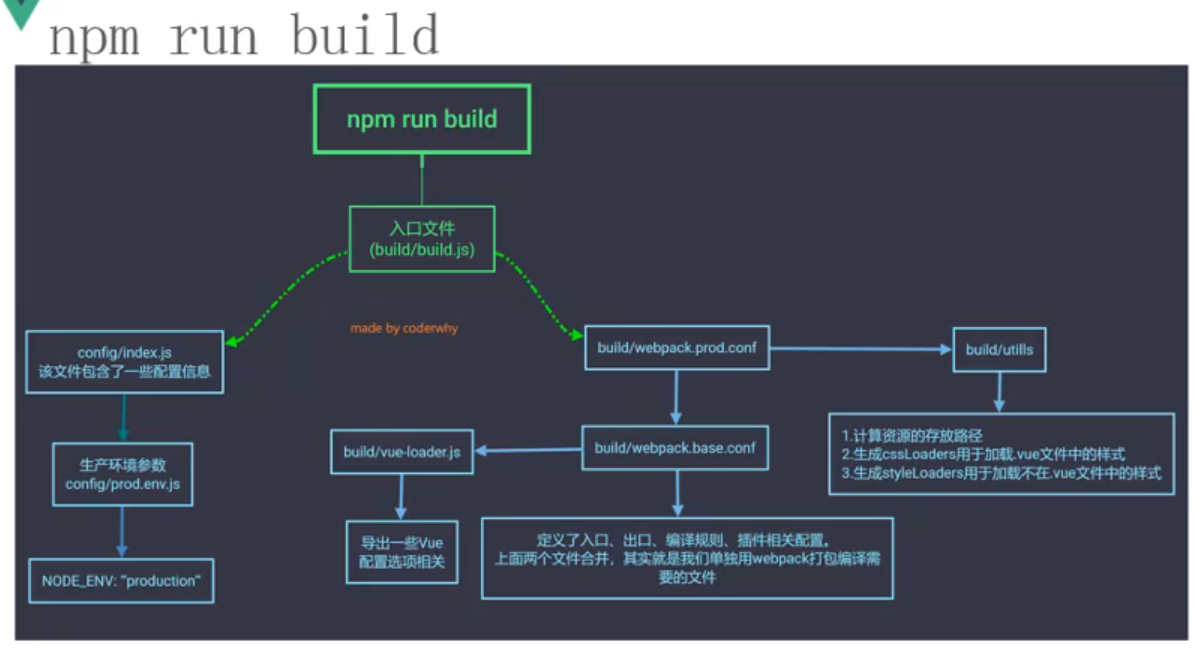
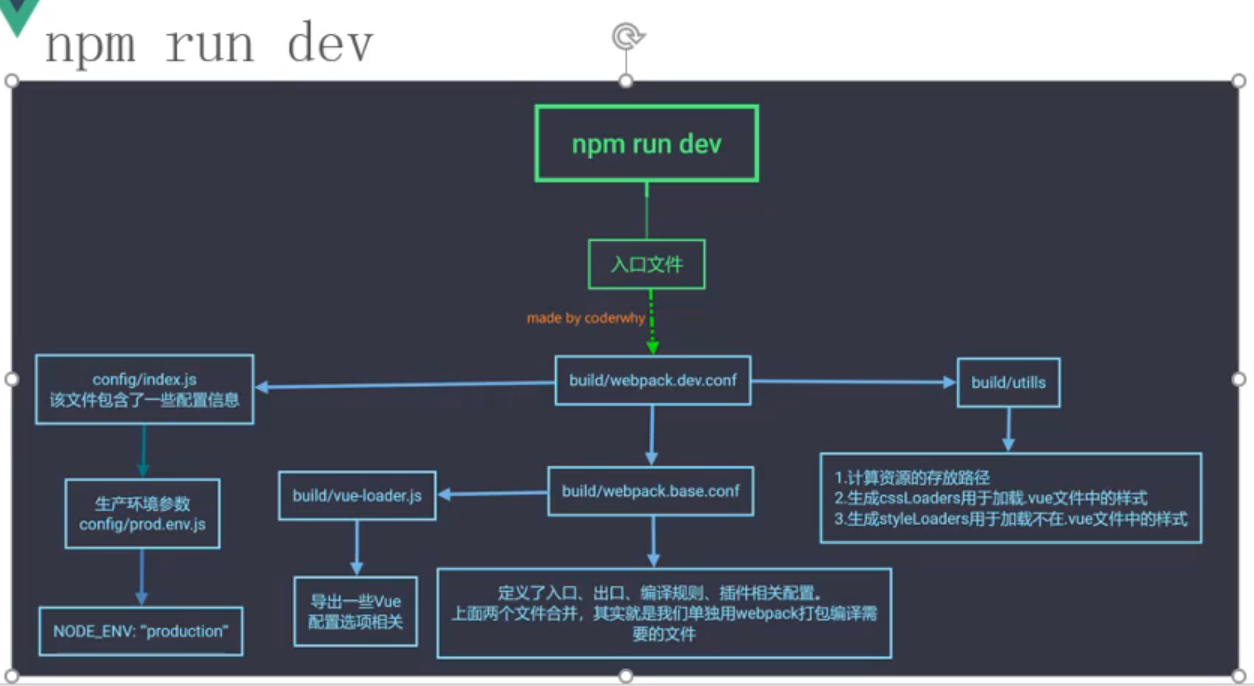
使用：

{template: '#c'}

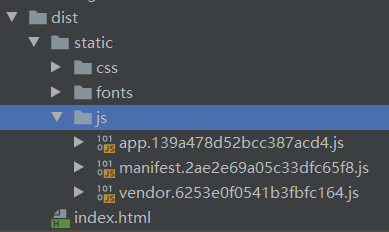
6、脚手架相关：



7、开发/生产的步骤：



8、dist目录结构含义：



app.js：自己写的业务代码

manifest.js：底层支撑代码

vendor.js：第三方代码（即各种插件等提供的代码：vue、vue-router、element-ui等等）

9、所有的组件都继承自Vue的原型：即

Vue.prototype.name = ‘qwe’;

this.name // qwe

Vue.prototype.$router

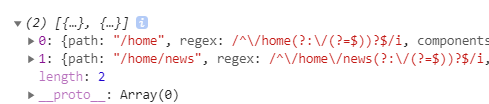
this.$router

Vue.prototype.$route

this.$route

10、使用$route.matched就可以找到各级的route，在各级的route上定义meta:{}就能保存一些数据，然后用$route.matched[ [num] ].meta.xxx取得，[num]为0时，就是父级路径。

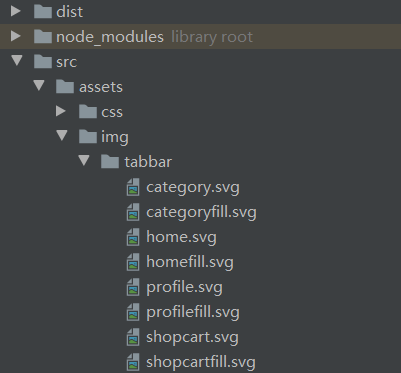
例如：/home/message



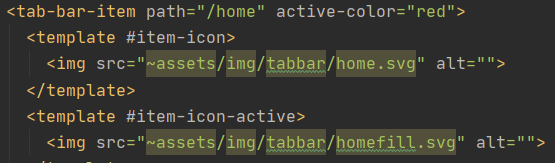


11、起别名：

定义：



使用：



12、父子组件传递的方法：

父->子：（子访问父）

1、在父中定义provide，子定义inject

<https://cn.vuejs.org/v2/guide/components-edge-cases.html#%E4%BE%9D%E8%B5%96%E6%B3%A8%E5%85%A5>

2、$parent

3、$root

4、props