上节聊了vue的条件渲染

这节谈谈vue的列表渲染

v-for

v-for是vue的一个指令，可以理解为就是为了把后台的多条数据循环遍历渲染到前端页面中 相当于for循环

形如

<ul>

<li v-for='item in arr'>

{{item.cor----}}

</li>

</ul>

data:{

arr:[

{cor:1},

{cor:2}

]

}

在v-for块中，我们拥有对父作用域属性的完全访问权限。v-for还支持第二个参数为当前项的索引

形如

<ul>

<li v-for='（item，index） in arr'>

{{item.cor}}--------{{index}}

</li>

</ul>

data:{

arr:[

{cor:1},

{cor:2}

]

}

就像js中的for in 和for of 循环遍历 vue中也可以用v-for=“item of arr”来代替v-for：“item in arr”

当然 v-for 也可以拿来遍历对象的属性

形如

<ul>

<li v-for="（val,key）in arr">{{val}}-----{{key}}<li>

</ul>

data:{

arr:{

cor:"1",

txt:"2"

}

}

此处语义化的很清楚 val 就是对象的值，key就是对象的健 单独遍历对象的值也是可以的

形如

<ul>

<li v-for="val in arr">{{val}}-----{{key}}<li>

</ul>

data:{

arr:{

cor:"1",

txt:"2"

}

}

很清楚的可以理解

遍历数组

一个参数 遍历的是数组中的每一个元素

两个参数 第一个是元素本身，第二个是索引

遍历对象

一个参数 遍历的对象的值

两个参数 第一个参数是值，第二和参数是键

三个参数 第一个是值，第二个是键，第三个是索引

当vue.js用v-for正在更新已经渲染过得元素列表时，它默认用就地复用策略，如果数据项的顺序被改变，vue将不会移动Dom元素

来匹配数据项的顺序，而是简单的复用此处每个元素，并且保证它在特定索引下显示已被渲染过得每一个元素，

这个默认的模式是高效的，但是只适用于不依赖子组件状态或临时DOM状态的列表渲染输出

为了给vue一个提示，以便他能跟踪每个节点的身份，从而重用和重新排序现有元素，你需要为每项提供一个唯一key属性。

理想的key值是每项都有的且唯一的id，并且需要利用v-bind绑定动态值

所以在我们使用v-for时尽可能的提供key，除非遍历输出的DOM内容非常简单，或者刻意依赖默认行为以获取性能上的提升