**Vue slot-scope的理解（适合初学者）**

         先抛例程：

1. <!DOCTYPE html>
2. <html lang="en">
3. <head>
4. <meta charset="UTF-8">
5. <title>组件之通过插槽(slot)分发内容</title>
6. <script src="你的vue地址，请下载最新版的"></script>
7. <script>
8. Vue.component("test-slot",{
9. // 插槽允许有默认内容
10. template:
11. `<div>
12. <strong>Error!</strong>
13. <slot></slot>
14. </div>
15. `,
16. data:function () {
17. return {
18. name:"perry"
19. }
20. }
21. });
22. // 具名插槽
23. Vue.component("slot-name",{
24. template:
25. `<div>
26. <header>
27. <slot name="header"></slot>
28. </header>
29. <main>
30. <slot ></slot>
31. </main>
32. <footer>
33. <slot name="footer"></slot>
34. </footer>
36. </div>
37. `
38. });
40. // 作用域插槽
41. Vue.component("todo-list",{
42. inheritAttrs:false,
43. props:{
44. todos:[Array,Object]
45. },
46. template:
47. `<ul>
48. <li v-for="todo in todos" :key="todo.id" style="display: block;" >
49. <slot :data="todo">{{todo.text}}</slot>
50. </li>
51. </ul>
52. `
53. });
55. </script>
57. <style>
58. li{
59. list-style: none;
60. display: inline-block;
61. font-size: 25px;
62. }
63. </style>
64. </head>
65. <body>
66. <div id="app">
67. *<!--1、插槽内容 slot可以接收任何内容、代码，包括组件-->*
68. <test-slot>Something bad happened.你是谁啊，我认识你吗</test-slot>
69. <test-slot>
70. Something bad happened.
71. *<!--允许使用变量，但该变量必须是本作用域的才会生效，这里name的值将会是"岔道口"-->*
72. <h3>标题{{name}}</h3>
73. <img src="img/flower.jfif" alt="图片无法显示">
75. </test-slot>
76. *<!--2、具名插槽 slo有一个特殊的属性name,可以使插槽具名-->*
77. <slot-name>
78. <h3>劝学</h3>
79. <p>少壮不努力，老大徒伤悲</p>
80. <p>光阴似箭，日月如梭</p>
81. <template slot="header">
82. <ul >
83. <li>主页</li>
84. <li>关于时间</li>
85. <li>关于人生</li>
86. <li>关于未来</li>
87. </ul>
88. </template>
89. <p>你好啊</p>
90. <p slot="footer">
91. <p>结尾</p>
92. <p>粤ICP备6545646456415</p>
93. </p>
94. </slot-name>
96. *<!--使用作用域插槽，通过slot-scope实现-->*
97. <todo-list :todos="todos">
98. <template slot-scope="slotProps">
99. <span v-if="slotProps.data.isTrue">√</span>
100. {{slotProps.data.text}}
101. </template>
102. </todo-list>
104. *<!--使用ES2015结构语法-->*
105. <todo-list :todos="todos">
106. <template slot-scope="{data}">
107. <span v-if="data.isTrue">√</span>
108. {{data.text}}
109. </template>
110. </todo-list>
111. </div>
113. <script>
114. new Vue({
115. el:"#app",
116. data:{
117. name:"岔道口",
118. todos:[
119. {text:"A",id:1,isTrue:true},
120. {text:"B",id:2,isTrue:true},
121. {text:"C",id:3,isTrue:false},
122. {text:"D",id:4,isTrue:true},
123. ],
124. *// slotProps:""*
125. }
126. })
128. </script>
129. </body>
130. </html>

       例程中也写了普通插槽和具名插槽，这个我就不说，我就说一下作用域插槽。首先从“作用域插槽“”这个名字说起，本来你父组件的模板是无法使用到子组件模板中的数据，官网中有一句特别强调的话：父组件模板的所有东西都会在父级作用域内编译；子组件模板的所有东西都会在子级作用域内编译。这句话其实挺难理解，举个例子就知道了，

    <test-slot>

          Something bad happened.

         <!--允许使用变量，但该变量必须是本作用域的才会生效，这里name的值将会是"岔道口"-->

           <h3>标题{{name}}</h3>

           <img src="img/flower.jfif" alt="图片无法显示">

</test-slot>

     这是我上面的例子，也就是说这里name必须是当前组件里的，不能是test-slot这个组件内部的数据，这是它作用域限制，name这个数据属于父组件的作用域，所以只能是父组件的数据。但是slot-scope的出现却实现了父组件调用子组件内部的数据，子组件的数据通过slot-scope属性传递到了父组件

1. // 作用域插槽
2. Vue.component("todo-list",{
3. inheritAttrs:false,
4. props:{
5. todos:[Array,Object]
6. },
7. template:
8. `<ul>
9. <li v-for="todo in todos" :key="todo.id" style="display: block;" >
10. //这里的：data=“todo”就是讲子组件todo这个数据传给了父组件
11. <slot :data="todo">{{todo.text}}</slot>//todo.text是默认值，父组件会把它覆盖掉
12. </li>
13. </ul>
14. `
15. });
16. *<!--使用作用域插槽，通过slot-scope实现-->*
17. <todo-list :todos="todos">
18. <template slot-scope="slotProps">
19. <span v-if="slotProps.data.isTrue">√</span>
20. {{slotProps.data.text}}
21. </template>
22. </todo-list>

使用的时候就是这样，从子组件传过来的名为data值为todo的数据将会被slot-scope接收，需要注意的是，slot-scope接收的是一个对象，这里命名为slotProps，也就是说你传过来的data将会作为slotProps的一个属性，所以需要slotProps.data才能调用到data。前面说了，data的值为todo，todo是什么呢？todo是todos遍历出来的，所以是一个对象。

1. new Vue({
2. el:"#app",
3. data:{
4. name:"岔道口",
5. todos:[
6. {text:"A",id:1,isTrue:true},
7. {text:"B",id:2,isTrue:true},
8. {text:"C",id:3,isTrue:false},
9. {text:"D",id:4,isTrue:true},
10. ],
11. // slotProps:""
12. }
13. })；

       我们已经理解了slot-scope特性，那么他有什么运用场景呢？我们可以试想一下，别人写好了一个组件，该组件已经封装好了，但他觉得，数据的显示风格让用户自己定义，就像我们上面，我们希望列表的样式由用户定义，比如我就加了一个√号。一开始最初的数据是我们传进去的，但数据必须进过子组件处理后，写组件那个人肯定希望显示的是处理后的数据。这时候slot-scope就无比重要了。