**[vue父子组件的传值总结](https://www.cnblogs.com/phermis/p/10710894.html)**

**情况一：父组件给子组件传值方法，使用props**

父页面：parent.vue

<template>

<div class="sidebar\_contianer">

<sidebar-item :routerData="transmitData"></sidebar-item>

</div>

</template>

<script>

import sidebarItem from './sidebarItem'

export default {

data(){

return{

transmitData:{

title:'首页',

uuid:'123'

}

}

},

components:{

sidebarItem

}

}

</script>

第一步：用import引入子组件

第二步：在components中注入子组件

第三步：在子组件中定义一个指令:routerData="transmitData"  （routerData名称需要和子组件中的props保持一致,transmitData是要传递给子组件的数据）

子页面：child.vue

<script>

export default {

name:'sidebarItem',

props:{

routerData:{

type:Object,

default:null

}

}

}

</script>

<script>

export default {

name:'sidebarItem',

props:[routerData]

}

第四步：在子组件中定义props属性，transmitData的type可以自定义类型（但是必须和父组件中的数据类型一致，否则会报错），default是默认值。

**情况二：子组件要给父组件传值方法，使用$emit**

子页面：a.vue

<template>

<div class="testCom">

<input type="text" v-model="message" />

<button @click="click">发送消息给爸爸</button>

</div>

</template>

<script>

export default {

data() {

return {

message: '我是来自子组件的消息'

}

},

methods: {

click() {

1、childFn 组件方法名，父组件中用childFn方法接收子组件中的数据；

2、message是传递给父组件的数据

this.$emit('childFn', this.message);

}

}

}

</script>

第一步：this.$emit("方法名","传递给父组件的数据")

父页面：b.vue

<template>

<div>

<child-com @childFn="parentFn"></child-com>

</div>

</template>

<script>

export default {

data: {

message: ''

},

methods: {

parentFn(childData) {

this.message = childData;

}

}

}

</script>

第二步：父组件中注入子组件（此处代码中省略，如要参考请看情况一）

第三步：定义方法childFn（必须和子组件中方法名一致），parentFn定义可随意。

注： 大型的项目如果用以上两种方法传递值，耦合性大。所以大型项目还是的用vuex对状态进行管理。