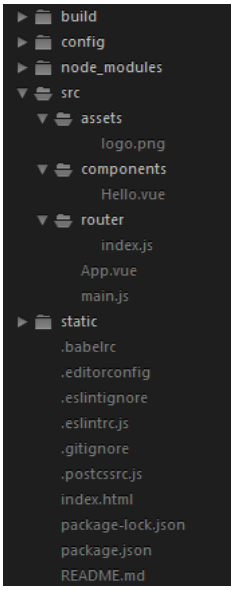
转 Vue2.0子同级组件之间数据交互

2017年07月23日 22:03:41 wjq_smile 阅读数：22097 标签： vue组件

熟悉了Vue.js的同级组件之间通信，写此文章，以便记录。

Vue是一个轻量级的渐进式框架，对于它的一些特性和优点，请在官网上进行查看，不再赘述。

使用NPM及相关命令行工具初始化的Vue工程，目录结构如下



接着我们进入Demo，首先我们可以删除掉模板项目中src/components/Hello.vue,然后在App.vue中删除对于Hello子组件的注册和使用他无关紧要的东西，此时的App.vue应为这样

App.vue

```
<template>
  <div id="app">

  </div>
</template>

<script>

export default {
  name: 'app',
}
</script>

<style>

</style>

```

一 .我们先来创建中央事件总线，在src/assets/下创建一个eventBus.js,内容如下
（eventBus中我们只创建了一个新的Vue实例，以后它就承担起了组件之间通信的桥梁了，也就是中央事件总线。）

二．创建一个firstChild组件，引入eventBus这个事件总线，接着添加一个按钮并绑定一个点击事件

```
App.vue | firstChild.vue | eventBus.js

<template>
    <div id="firstChild">
        <h2>firstChild组件</h2>
        <button v-on:click="sendMsg">向组件传值</button>
    </div>
</template>
<script>
    import bus from '../assets/eventBus';
    export default{
        methods:{
            sendMsg:function(){
                bus.$emit("userDefinedEvent","this message is from
                    firstChild");
            }
        }
    }
</script>
<style>
#firstChild{
}
</style>
```

1、我们在响应点击事件的sendMsg函数中用$emit触发了一个自定义的userDefinedEvent事件，并传递了一个字符串参数
2、$emit实例方法触发当前实例(这里的当前实例就是bus)上的事件,附加参数都会传给监听器回调。

三．我们再创建一个secondChild组件，引入eventBus事件总线，并用一个p标签来显示传递过来的值

App.vue | firstChild.vue | secondChild.vue | eventBus.js

```vue
<template>
    <div id="secondChild">
        <h2>secondChild组件</h2>
        <p>从firstChild接收的字符串参数：{{msg}}</p>
    </div>
</template>
<script>
    import bus from '../assets/eventBus';
    export default{
        data(){
            return {
                msg:"默认值"
            }
        },
        mounted(){
            var self=this;
            bus.$on("userDefinedEvent",function(msg){
                self.msg=msg;
            });
        }
    }
</script>
<style>
    #secondChild{
    }
</style>
```

1、我们在mounted中，监听了userDefinedEvent,并把传递过来的字符串参数传递给了$on监听器的回调函数

2、mounted:是一个Vue生命周期中的钩子函数，简单点说就类似于jQuery的ready，Vue会在文档加载完毕后调用mounted函数。

3、$on:监听当前实例上的自定义事件(此处当前实例为bus)。事件可以由$emit触发，回调函数会接收所有传入事件触发函数($emit)的

四 . 在父组件中，注册这两个组件，并添加这两个组件的标签

App.vue | firstChild.vue | secondChild.vue | eventBus.js

```vue
<template>
  <div id="app">
    <firstChild></firstChild>
    <secondChild></secondChild>
  </div>
</template>

<script>
import firstChild from './components/firstChild';
import secondChild from './components/secondChild';
export default {
  name: 'app',
  components:{
    firstChild,
    secondChild
  }
}
</script>
<style>
</style>

```

保存所有修改的文件，然后打开浏览器窗口，内容如下(css请自行处理)

点击向组件传值按钮，我们可以看到传值成功

总结：

1、创建一个事件总线，例如demo中的eventBus，用它作为通信桥梁

2、在需要传值的组件中用bus.$emit触发一个自定义事件，并传递参数

3、在需要接收数据的组件中用bus.$on监听自定义事件，并在回调函数中处理传递过来的参数

另外：

1、兄弟组件之间与父子组件之间的数据交互，两者相比较，兄弟组件之间的通信其实和子组件向父组件传值有些类似，其实他们的通信原理都是相同的，父传值也是$emit和$on的形式，只是没有eventBus，但若我们仔细想想，此时父组件其实就充当了bus这个事件总线的角色。

2、这种用一个Vue实例来作为中央事件总线来管理组件通信的方法只适用于通信需求简单一点的项目，对于更复杂的情况，Vue也有提供更复杂的状态Vuex来进行处理。

vue2.0中的组件通信3——兄弟组件之间通信

上一篇我写了子组件通过事件向父组件发送数据，在这里我们就来聊聊同级组件之间数据通信的问题。同级组件之间的通信需要依赖一个vue实例作为中介者，（类似邮差）的作用。 我的代码依然沿用上面两...