Vue 弹窗组件封装 命令式组件封装

目录 disc

教程 Tutorial

弹窗属于通用型的功能，在各个场景中都可能用到。开发仅需要一次，使用却需要很多次，因此着重考虑使用时的成本。

组件式

```
<!-- MessageBox.vue -->
<template>
    <div id="modal">
        <div id="box">
            <div id="text">{{ msg }}</div>
            <Button @click="emit('click')"></Button>
        </div>
    </div>
</template>
<script setup>
import Button from "~/components/Button.vue";
const emit = defineEmits(['click']);
defineProps({
    msg:{
        type:String,
        required:true
    }
})
</script>
<style scoped>
.modal{
    ...
}
.box{
    ...
}
.text{
    ...
}
</style>
```

开发十分简单，但是使用呢?

```
<!-- App.vue -->
<template>
    <div>
        <Button @click="clickHandler">显示弹窗</Button>
        <MessageBox v-if="showMsg" :msg="msg" @click="clickHandler"></MessageBox>
```

```
    </div>
</template>
<script setup>
import { ref } from 'vue';
import Button from "~/components/Button.vue";
import MessageBox from "~/components/MessageBox.vue";
const showMsg=ref(false);
const msg=ref('提示消息');
const clickHandler = () => {
    showMsg.value=!showMsg.value
};
</script>
```

是否十分甚至九分麻烦?

那么能否不导入任何组件使用呢?

能。

```
<!-- App.vue -->
<script setup>
- import { ref } from 'vue';
import Button from "~/components/Button.vue";
+ import showMsg from "~/commons/showMsg"
- import MessageBox from "~/components/MessageBox.vue";
- const showMsg=ref(false);
- const msg=ref('提示消息');
const clickHandler = () => {
-    showMsg.value=!showMsg.value;
+    showMsg('欲显示的消息',(close)=>{
+    console.log('点击了确定按钮');
+    close();
+ })
};
</script>
```

稍作修改

这样过后 开发就不会那么容易了。

但是，这是**通用型的功能**，因此不用太多地考虑开发成本。

所以在js文件里定义一个函数

```
// ~/commons/showMsg.js
function showMsg() {

}
export default showMsg;
```

在函数里接受两个参数，一个是消息内容，另一个是事件.

```
// ~/commons/showMsg.js
- function showMsg() {
-
- }
+ function showMsg(msg,clickHandler) {
+
+ }
export default showMsg;
```

这个方法是为了渲染MessageBox。

导入刚才的MessageBox.vue

```
// ~/commons/showMsg.js
+ import Message from "~/components/MessageBox.vue"
function showMsg(msg,clickHandler) {

}
export default showMsg;
```

怎么渲染呢?

我们看看Vue怎么渲染的

```
// main.js
import { createApp } from 'vue'
import App from './App.vue'

const app = createApp(App)

app.mount('#app')
```

创建一个Vue的实例，把组件传进去，再挂载到页面上的某个区域。

所以我们把这个事情也做一遍。

```
// ~/commons/showMsg.js
+ import { createApp } from 'vue'
import Message from "~/components/MessageBox.vue"
function showMsg(msg,clickHandler) {
+    // 渲染MessageBox组件
+    const app =createApp(MessageBox);
}
export default showMsg;
```

但是app已经被主页面挂载了，所以我们要换一个。

```
// ~/commons/showMsg.js
import { createApp } from 'vue'
import Message from "~/components/MessageBox.vue"
function showMsg(msg,clickHandler) {
+    const div =document.createElement('div');
   // 渲染MessageBox组件
+    document.body.appendChild(div);
   const app =createApp(MessageBox);
+    app.mount(div);
}
export default showMsg;
```

刷新页面，发现弹窗已经出来了，但是没有内容。

所以现在我们要接受两个属性

```
// ~/commons/showMsg.js
import { createApp } from 'vue'
import Message from "~/components/MessageBox.vue"
function showMsg(msg,clickHandler) {
    const div =document.createElement('div');
    // 渲染MessageBox组件
    document.body.appendChild(div);
-    const app =createApp(MessageBox);
+    const app =createApp(MessageBox,{
+        msg,
+        onClick(){
+            clickHandler & clickHandler(()=>{
+                app.unmount();
+                div.remove();
+            });
+        }
+    });
    app.mount(div);
}
export default showMsg;
```

完事

现在我们使用方法就可以调用了

```
<!-- App.vue -->
<script setup>
import Button from "~/components/Button.vue";
import showMsg from "~/commons/showMsg"
const clickHandler = () => {
   showMsg.value=!showMsg.value;
   showMsg('欲显示的消息',(close)=>{
   console.log('点击了确定按钮');
   close();
})
};
</script>
```

目前这种写法并不是很好，因为他拆开成了几个文件`showMsg.js`和`MessageBox.js`。

所以我们要合并成一个文件。

```
// ~/commons/showMsg.js
import { createApp } from 'vue'
+ import Button from "~/components/Button.vue";
- import Message from "~/components/MessageBox.vue"
+
+ const MessageBox = {
+     props: {
+         msg: {
+             type: String,
+             required: true,
+         },
+     },
+     render(ctx) {
+         const { $props, $emit } = ctx;
+         return (
+             <div class="modal">
+                 <div class="box">
+                     <div class="text">{$props.msg}</div>
+                     <Button click={$emit('onClick')}></Button>
+                 </div>
+             </div>);
+     }
+ }
+
function showMsg(msg,clickHandler) {
    const div =document.createElement('div');
    // 渲染MessageBox组件
    document.body.appendChild(div);
    const app =createApp(MessageBox);
    const app =createApp(MessageBox,{
        msg,
        onClick(){
            clickHandler & clickHandler(()=>{
                app.unmount();
                div.remove();
            });
        }
    });
    app.mount(div);
}
export default showMsg;
```

接下来就是添加样式。

这里使用`@styils/vue`

```
$ npm instll @styils/vue
```

导入`@styils/vue`

```
// showMsg.js
import { styled } from "@styils/vue";
```

写样式

```
// showMsg.js
const DivModal = styled('div', {
    ...
});
const DivBox = styled('div', {
    ...
});
const DivText = styled('div', {
    ...
});
```

注意带横线的样式转换为驼峰写法。

修改显示

```
// showMsg.js
...
- return (
-     <div class="modal">
-         <div class="box">
-             <div class="text">{$props.msg}</div>
-             <Button click={$emit('onClick')}></Button>
-         </div>
-     </div>
- );
+ render(ctx) {
+     const { $props, $emit } = ctx;
+     return (
+         <DivModal>
+             <DivBox>
+                 <DivText>{$props.msg}</DivText>
+                 <Button click={$emit('onClick')}></Button>
+             </DivBox>
+         </DivModal>);
+ }
...
```

最终文件

```
// showMsg.js
import { createApp } from 'vue'
import Button from "~/components/Button.vue";
import { styled } from "@styils/vue";
const DivModal = styled('div', {
    ...
});
const DivBox = styled('div', {
    ...
});
```

```
const DivText = styled('div', {
    ...
});

const MessageBox = {
    props: {
        msg: {
            type: String,
            required: true,
        },
    },
    render(ctx) {
        const { $props, $emit } = ctx;
        return (
            <DivModal>
                <DivBox>
                    <DivText>{$props.msg}</DivText>
                    <Button click={$emit('onClick')}></Button>
                </DivBox>
            </DivModal>);
    }
}

function showMsg(msg, clickHandler) {
    const div = document.createElement('div');
    document.body.appendChild(div);
    const app = createApp(MessageBox, {
        msg,
        onClick() {
            clickHandler & clickHandler(() => {
                app.unmount(div);
                div.remove();
            });
        }
    });
    app.mount(div);
}
export default showMsg;
```

效果演示 Effect Show

演示地址：首页按钮