# Vite+Vue



Docx文档测试了标题和图片，以下是大量的文本在内容

Vue3 中加入了响应式API，书写一个组件的方式又增加了几种。看看有什么不一样

### `vue2` 和 `vue3`都适用

1. 熟悉的模板`.vue`单文件

自定义语法、插值、指令都可以使用。按照官网书写就不会有什么问题。

```js

<template>

<div></div>

</template>

<script>

export default {

name:"",

data(){

return {}

}

}

</script>

<style></style>

```

2. 不使用`template`,改用`render`函数

改用render函数后，以前是`createElement`函数，简写为h，这种方式太繁琐了，代码完全无法维护。支持JSX语法后，好太多。

但随之而来的一些vue语法特性、指令等使用方式发生更改，需要个人对这些有一些了解。个人喜好更喜欢JSX的书写方式。

```js

<script>

export default {

name:"",

data(){

return {}

},

methods:{},

render(){

return <></>

}

}

</script>

<style></style>

```

3. 分离JS，一是还采用`template`,另一种采用`render`函数。

和1、2 写法一致，只是把script提取到外部单文件中。

同级目录下创建`index.js`

```js

<template>

<div></div>

</template>

<script src="./index.js">

</script>

<style></style>

```

`index.js`

```js

export default {

data(){

return {}

},

render(){

return <></>

}

}

```

以上都是vue2中也可以这样写的，自从vue3新增响应式API，又多了不同方式的书写。

\* 一是在1,2,3基础是使用响应式API

\* 而是完全使用响应式API。

### vue3有了响应式API的书写

4. 在\*\*1,2,3\*\*基础上使用响应式API

`script` 部分通用,现有的非响应式配置仍然有效

```js

import { ref ,computed,onMounted} from 'vue'

export default {

setup(){

// 生命变量

const name = ref('')

// 计算值

// 变量在内部使用时，需要 .value

const firstName = computed(()=>name.value.toUpperCase())

// 定义方法

const getData = ()=>{

/\*\*

\*

\*\*/

}

// 生命周期

onMounted(()=>{

// 挂载

})

return {

name,

}

},

methods:{

// 定义方法

},

render(){

return <>

<p>{this.name}</p>

</>

}

}

```

提供了`defineComponent` 语法提示、类型推断

```js

import { defineComponent} from 'vue'

export default defineComponent({

// ...

})

```

5. `.vue` 模板文件中，`script` 增加setup标识

```js

<template>

<div>

<p>{{name}}</p>

</div>

</template>

<script setup>

// 必须调用响应式API

import { ref ,computed,onMounted} from 'vue'

// 生命变量

const name = ref('')

// 计算值

// 变量在内部使用时，需要 .value

const firstName = computed(()=>name.value.toUpperCase())

// 定义方法

const getData = ()=>{

/\*\*

\*

\*\*/

}

// 生命周期

onMounted(()=>{

// 挂载

})

</script>

<style></style>

```

6. 当然`setup` 也可以不使用`template`

```js

<script setup>

// 必须调用响应式API

import { ref ,computed,onMounted,h} from 'vue'

// 生命变量

const name = ref('')

// 生命周期

onMounted(()=>{

//

})

</script>

<script>

// 普通的script

export default {

render() {

return h("p", this.name);

},

};

</script>

<style></style>

```

`setup`模式下不能使用`render`函数，但是我们可以使用普通的`script`.定义render函数。

7. `defineComponent` 直接返回匿名函数，或者具名函数，函数名作为组件名称

```js

import { ref ,computed,onMounted,defineComponent} from 'vue'

export default defineComponent(()=>{

// 生命变量

const name = ref('')

// 计算值

// 变量在内部使用时，需要 .value

const firstName = computed(()=>name.value.toUpperCase())

// 定义方法

const getData = ()=>{

/\*\*

\*

\*\*/

}

// 生命周期

onMounted(()=>{

// 挂载

})

// return 直接渲染试图

return ()=><>

<p></p>

</>

})

```

测试项目，代码仓库[vite-vue3(https://gitee.com/ngd\_b/vue3-vite)](https://gitee.com/ngd\_b/vue3-vite)