**兼容性**

Vue **支持** IE9 及上版本，因为 Vue 使用了 IE8 无法模拟的 ECMAScript 5 特性。但它支持所有[**兼容 ECMAScript 5 的浏览器**](https://caniuse.com/#feat=es5)。

因为它使用的是对象的getter,setter。这些特性ie9以上才支持，其他非ie系浏览器基本都支持

**数据绑定**

**绑定文本**：{{绑定键名}}

**绑定属性**：v-bind:属性名=” 绑定键名” ，它有一种缩写：“:属性名 = 绑定键名”

**绑定DOM**：v-if=”绑定键名(必须是boolean类型)”

**组件化**

在一个大型的应用中，不同的页面中肯定有重复的内容，那么将这些重复的内容封装成组件，也就更便于维护了。以下是创建一个组件的例子：

JS文件：

Vue.component('todo-item', {

props: ['todo'], //todo是传入这个组件的参数

template: '<li>{{ todo }}</li>' //模板

})

var app7 = new Vue({

el: '#app-7',

data: {

groceryList: [

'蔬菜' ,

'奶酪' ,

'随便其它什么人吃的东西',

]

}

})

HTML文件：

<!--

v-bind:todo传递参数

-->

<todo-item v-for="item in groceryList"

v-bind:todo="item">

</todo-item>

**选项/数据**

当我们创建一个Vue实例时，你可以传入一个选项对象，下面对它们一一介绍：

**#data**

**类型：Object | Function**

data是Vue实例的数据对象。Vue会将data属性中的值转成getter/setter，从而让data的值能够响应数据的变化。值得注意的是只有当实例被创建时data中存在的属性才是响应式的。也就是说如果在页面加载完毕后新增一个属性就不会触发更新。解决方法是，将那些一开始值为空的属性设置为初始值。

实例创建之后，可以通过 vm.$data 访问原始数据对象。Vue 实例也代理了 data 对象上所有的属性，因此访问 vm.a 等价于访问 vm.$data.a。

当一个**组件**被定义，data 必须声明为返回一个初始数据对象的函数，因为组件可能被用来创建多个实例。如果 data 仍然是一个纯粹的对象，则所有的实例将**共享引用**同一个数据对象！通过提供 data 函数，每次创建一个新实例后，我们能够调用 data 函数，从而返回初始数据的一个全新副本数据对象。

**#props**

**类型：Array<string> | Object**

props 可以是数组或对象，用于接收来自父组件的数据。props 可以是简单的数组，或者使用对象作为替代，对象允许配置高级选项，如类型检测、自定义校验和设置默认值。

**#methods**

**类型：{key<string> : Function}**

可以用v-on指令监听DOM事件，该函数的第一个参数为event对象。

**计算属性**

当我们需要对普通属性中值进行一些复杂的逻辑运算时，可以使用computed(计算属性)来绑定普通属性。

计算属性的结果会被缓存，除非普通属性发生改变需要重新计算。计算属性相当于声明了一个getter函数，使它更易于维护和理解。

例如：HTML文件：

<tr>

<td>用户名：</td>

<td><input type="text" v-model="UserName"/></td>

<td :class="ValidUserName.font">**{{**ValidUserName.msg**}}**</td>

</tr>

JS文件：

data: {

UserName: "",

},

computed: {

ValidUserName: function () {

if (this.UserName.length <= 0) {

return {

state: 0,

font:"font-red",

msg : "用户名不能为空"

};

}}

当需要在数据变化时执行异步或开销较大的操作时，我们需要自定义一个监听器，Vue通过watch选项提供了一个通用的方法来响应数据。