**一、引入文件并使用**

1. 新建一个JS，可直接定义多个函数，后将函数定义在window上直接导出

如function dj () {} window.dj = dj main.js中导入后通过dj直接引用

或 ( function (window) { formatImg(){} window.formatImg = formatImg }(window) )

通过window调用

1. 定义一个js，const tools = { //多个通用函数 } 后， export default tools;

或直接export default { //多个通用函数 }

main.js中import tools from './dj’后通过tools.dj ( ) 使用；(//Vue.prototype.tools = tools; )

1. **mixin**( 混合机制 )：

多个组件可以共享数据和方法，在使用mixin的组件中引入后，mixin中的方法和属性也就并入到该组件中，可以直接使用。钩子函数会两个都被调用，mixin中的钩子首先执行。

mixins是在引入组件之后，将组件内部的内容如data等方法、method等属性与父组件相应内容进行合并。相当于在引入后，父组件的各种属性方法都被扩充了

单纯组件引用： 父组件 + 子组件 >>> 父组件 + 子组件

mixins： 父组件 + 子组件 >>> new父组件

用法：

//1、新定义一个mixin.js文件

export default {

data() {

return {

name: 'mixin'

}

},

created() {

console.log('mixin...', this.name);

},

mounted() {},

methods: {}

}

2、在vue文件中使用mixin

import mixin from './mixin'; // 先引入mixin文件

export default {

mixins: [mixin] // 再在data层次引入mixin

}

3、在main.js中全局混入（不建议全局混入）

import Vue from 'vue'

import mixin from './mixin' // 先引入mixin文件

Vue.mixin(mixin) //全局混入是.mixin没有s

二、全局或部分组件注册

1、Vue.component 注册的 【全局组件】或 【单独组件】（components: { },）

2、Vue.filter 注册的 【全局过滤器】或【单独过滤器】（filters:{ }）

Vue.js 允许你自定义过滤器，可被用于一些常见的文本格式化。过滤器可以用在两个地方：双花括号插值和 v-bind 表达式

//局部组件定义

filters: {

capitalize: function (value) {

if (!value) return ''

value = value.toString()

return value.charAt(0).toUpperCase() + value.slice(1)

}

} //使用

<!-- 在双花括号中 -->

{{ message | capitalize }}

<!-- 在 `v-bind` 中 -->

<div v-bind:id="rawId | formatId"></div>

//或者在创建 Vue 实例之前全局定义过滤器：

Vue.filter('capitalize', function (value) {

if (!value) return ''

value = value.toString()

return value.charAt(0).toUpperCase() + value.slice(1)

})

new Vue({// ...})

或者全局生成一个filter.js文件：（同directive）

import Vue from 'vue';

Vue.filter('priceValue', function (value) { });

Vue.filter(filterTime, function (value) { });

在main.js中引入： import '@js/filters.js';

3、Vue.directive 注册的 【全局指令】或【单独指令】（directive：{ }）

Vue 也允许注册自定义指令

// 注册一个全局自定义指令 `v-focus`

Vue.directive('focus', {

inserted: function (el) { // 当被绑定的元素插入到 DOM 中时……

el.focus()// 聚焦元素

}

})

//如果想注册局部指令，组件中也接受一个 directives 的选项：

directives: {

focus: {

inserted: function (el) { // 指令的定义

el.focus()

}

}

} //然后你可以在模板中任何元素上使用新的 v-focus 属性，如下：

<input v-focus>

1. Vue.mixin 注册的 【全局mixin】或【单独mixin】（mixins: [mixin]）（参考前面）

//处理时间方法

setFormTime(val, type = 1){

if (val) {

let date = new Date(val);

const year = date.getFullYear();

const month = `0${date.getMonth() + 1}`.slice(-2); //月补0操作

const day = `0${date.getDate()}`.slice(-2); //日补0操作

const hour= `0${date.getHours ()}`.slice(-2); //小时补0操作

const minute = `0${date.getMinutes ()}`.slice(-2); //分钟补0操作

const second = `0${date.getSeconds ()}`.slice(-2); //秒补0操作

if (type == 1) {

return year+"-"+month+"-"+day+" "+hour+":"+minute+":"+second;

} else {

return year+"-"+month+"-"+day

}

} else { return "" }

}

//对象重新排序

changeSort(data) {

var newkey = Object.keys(data).sort()

var newObj = {}

for (var i = 0; i < newkey.length; i++) {

newObj[newkey[i]] = data[newkey[i]]

}

return newObj // 返回排好序的新对象

};

//动态class的几种绑定方式

:class="{'className': true/false, 'sort': isSort}"

:class=" true/false ? 'active': ''"

:class="[ true/false ? 'blackBg' : '', 'CongestionShow']"

:class="[{ 'className': true/false }, 'sort']"