Vue.js

<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>

(會自動抓取 Vue.js 最新版本)

--🡪 @import

import'./'*// 同一層目錄開始找*

import'../'*// 上一層目錄開始找*

import'@/'*// src目錄下開始找*

import''*// 直接打名稱則是node\_modules*

|  |  |
| --- | --- |
| v\_if | <div id="app">  <p v-if="seen">you look me.</p>  </div>  new Vue({  el:"#app" ,  data() {  return {  seen : true  }  },  }) |
| v\_for | <div id="app">  <p v-for="(item, index) in items" :key="index">  {{ item.id }}  </p>  </div>  new Vue({  el:"#app" ,  data() {  return {  items : [  {id:"pudding" , "number":1}  ]  }  },  })  (建議最好每個 for 都設定 key，假如 array 更動時，資料不會更著陣列一起更新，如果設定 key 時，資料就會自動更動到該欄位) |
| List | * push()：新增元素。 * pop()：刪除最新加入的元素。 * shift()：刪除第一個 (即最舊的) 元素。 * unshift()：加入元素至第一個位置。 * splice()：加入或移除元素。 * sort()：由小至大排序。 * reverse()：元素反序排列。 * filter()：過濾陣列的元素，並將符合條件的元素傳回成為一個新陣列。 * concat()：連接陣列，會返回一個新的陣列。 * slice()：切割陣列，會返回一個新的陣列。 |
| is vs :is | is (指定一個 component)  <div id="app">  <div is="com"></div>  </div>  Vue.component('com', {  template:'<p> title </p>'  })  new Vue({  el:"#app"  })  : is (需指定一個變數，並用變數去呼叫compoent)  <div id="app">  <div :is="view"></div>  </div>  var vm = new Vue({  el:"#app" ,  data() {  return {  view:''  }  },  comments:{  view:{  template:'<h2> title </h2>',  }  }  }) |
| v\_model  (雙向綁定) | <div id="app">  <p>{{ message }}</p>  <input v-model="message">  </div>  new Vue({  el:"#app" ,  data() {  return {  message : 'Hello'  }  },  }) |
| v\_bind | 1. 可放css 指令      1. Class      1. Style |
| Vue Compoent | # ( props 可以動態配置 template 裡的變數) |
| Vue On | <button v-on:click.prevent="handle"></button>  <button @click.prevent="">  <a href="#"> click </a>  </button>  # prevent 頁面轉換如果是同個頁面時，url 後會出現 #，若不想 url 出現 # ，則使用 |
| vue 頁面轉換 | <router-link :to="'HelloWorld'"></router-link> |
| (Props資料傳遞)  Parent  and  Child | * child.vue (將變數 message 藉由 props 方式傳遞給父物件)   <template>  <h5>{{ message }}</h5>  </template>  <script>  export default {  props:['message']  }  </script>   * parent.vue   <template>  <div>  <p>This is Parent.</p>  <child message='Text'></child>  <child :message='a'></child>  </div>  </template>  <script>  import Child from '../components/child.vue'  export default {  components:{Child} ,  data(){  return{  a:10,  b:10,  }  }  }  </script>  Props 型態、資料設定  prop=(型態) : 限制使用者只能塞特定型態  required : 一定要用此 prop  default : 當使用者沒有輸入值時，就設定默認值  validator : 條件限定，若判斷錯誤則會error   * child.vue   <template>  <div>  <span>Message : {{propA}}</span>  <span>Message : {{propB}}</span>  <span>Message : {{propF}}</span>  </div>  </template>  <script>  export default {  props:{  propA:Number,  propB:[String , Number],  propC:{  type:String ,  //required: true  },  propD:{  type:Number,  default:100  },  propE:{  type:Object,  default()  {  return {message: 'hello'}  }  },  propF:{  validator(value)  {  return value > 10  }  }  }  }  </script>   * parent.vue   <template>  <div id="app">  <child :propA='10' propB="20" :propF="100"></child>  </div>  </template>  <script>  import Child from './Child.vue'  export default {  name: 'app',  components:{Child},    }  </script>  <https://www.jianshu.com/p/91416e11f012>  (\* 注意 : 只能單向傳輸。(單:父物件抓子物件，不可雙向傳輸))  **(props 與 $ref的區別 :**  **props 擅長傳遞變數(不傳屬系、方法)、$ref 擅長傳方法(不傳變數))** |
| (ref 資料傳遞)  $ref | * child.vue (一般的寫法。)   <template>  <h5>{{ message }}</h5>  </template>  <script>  export default {  data(){  return{  message : ''  }  },  methods:{  getMessage(m)  {  this.message = m;  }  }  }  </script>   * parent.vue   <template>  <div>  <p> 我是父組件 !! </p>  <child ref="msg"></child>  </div>  </template>  <script>  import Child from './Child.vue'  export default {  components:{Child},  mounted:function()  {  console.log(this.$refs.msg);  this.$refs.msg.getMessage('我是子組件 !!');  }  }  </script>  <https://www.jianshu.com/p/91416e11f012>  \* 藉由 ref = msg 指定 msg 為 child.vue  (\* 注意 : 只能單向傳輸。(單:父物件抓子物件，不可雙向傳輸))  **(props 與 $ref的區別 :**  **props 擅長傳遞變數(不傳屬系、方法)、$ref 擅長傳方法(不傳變數))** |
|  |  |
|  |  |
| ***套件載入 :*** | |
| bootStrap | <https://bootstrap-vue.js.org/docs/> |
|  |  |