2206Web 笔记

数据库笔记地址:

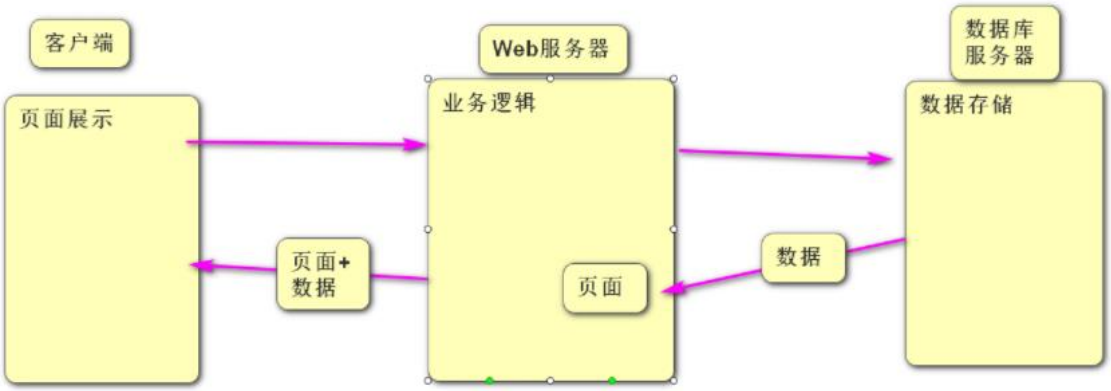
https://docs.qq.com/doc/DTEFicm10TWxWTGlQ

自我介绍

刘国斌

课程介绍：

- Web 前端： 学习如何搭建页面
- 数据库： 学习如何对数据进行增删改查
- SpringBoot： 学习如何接收客户端请求以及如何对请求做出响应



HTML

- HyperTextMarkupLanguage： 超文本标记语言
- 超文本： 不仅仅是纯文本还包括字体效果和多媒体（音频视频图片）
- 标记语言特点：

```
<开始标签 属性名="属性值">标签体</结束标签>
```

- 作用： 搭建页面结构和内容 ， 相当于盖房子（毛坯房）
- 学习 HTML 主要学习的就是有哪些标签，以及标签的使用方式。

文本相关标签

1. 内容标题： h1-h6

   独占一行, 自带上下间距, 字体加粗
2. 换行: br
3. 段落标签 p

   独占一行, 自带上下间距
4. 水平分割线:hr
5. 加粗 b
6. 斜体 i
7. 下划线 u
8. 删除线 s

列表标签

1. 无序列表: ul 和 li 组合
2. 有序列表: ol 和 li 组合
3. 列表嵌套: 有序和无序可以任意无限嵌套

图片标签

- src: 资源路径
  - 相对路径:访问站内资源使用
    - 和页面同级目录: 直接写图片名
    - 在页面的上级目录: ../图片名
    - 在页面的下级目录: 文件夹名/图片名
  - 绝对路径:访问站外资源使用, 称为图片盗链,有找不到图片的风险
- alt: 当图片不能正常显示时显示的文本
- title:图片标题 当鼠标悬停时显示的文本
- width/height:设置宽高, 只设置宽度高度会自动等比例缩放， 两种赋值方式:1.像素 2.百分比

- a 标签包裹文本为文本超链接 , 包裹图片为图片超链接
- 页面内部跳转, 在目的地元素里面添加 id 属性, 然后在超链接的 href 属性中写#id 这样点击时就可以跳转到目的地元素的位置

表格标签 table

- 相关标签: table 表格 tr 表示行　td 表示列 th 表头 caption 表格标题
- 相关属性:border 边框　　colspan 跨列　 rowspan 跨行

表单 form

- 作用: 获取用户输入的内容并提交给服务器
- 学习表单主要学习的就是表单中有哪些控件, 包括:文本框,密码框,单选,多选,下拉选,日期,文件等
- 相关代码:

```
<form action="http://www.baidu.com">
    <!--placeholder 占位文本
    maxlength 最大字符长度
    value 设置控件的值
    readonly 只读-->
    用户名:<input type="text" name="username" maxlength="5"
value="tom"
                readonly placeholder="请输入用户名"><br>
    密码:<input type="password" name="password" placeholder="请输入
密码"><br>
    <!--value 设置提交的值,如果不设置则提交 on checked 设置默认选中-->
    性别:<input type="radio" name="gender" value="m" id="r1">
    <label for="r1">男</label>
    <input type="radio" name="gender" checked value="w" id="r2">
    <label for="r2">女</label><br>
    兴趣爱好:<input type="checkbox" name="hobby" value="cy">抽烟
    <input type="checkbox" name="hobby" checked value="hj">喝酒
    <input type="checkbox" name="hobby" value="tt">烫头<br>
    生日:<input type="date" name="birthday"><br>
    靓照:<input type="file" name="pic"><br>
    所在地:
    <select name="city">
        <!--value 设置提交的内容,如果不写则提交标签体的内容
        selected 设置默认选中-->
        <option value="bj">北京</option>
        <option value="sh" selected>上海</option>
        <option value="gz">广州</option>
    </select><br>
    <input type="submit" value="注册">
    <!--重置按钮-->
    <input type="reset" value="重置按钮">
    <!--自定义按钮-->
    <input type="button" value="自定义按钮">
    <hr>
    <button type="submit">注册</button>
    <button type="reset">重置按钮</button>
    <button type="button">自定义按钮</button>
</form>
```

分区标签

- 作用: 可以理解为一个容器, 对多个有相关性的标签进行统一管理
- 常见的分区标签包括:
  - div: 独占一行
  - span: 共占一行
- HTML5 标准中新增的专门用于做页面布局的分区标签作用和 div 一样都是独占一行的:
  - header 头
  - main 主体
  - footer 脚
  - section 区域
  - nav 导航

CSS 层叠样式表

- Cascading Style Sheet, 作用: 美化页面, 相当于装修

如何在 HTML 页面中添加 CSS 样式代码

- 三种引入方式:
  - 内联: 在标签的 style 属性中添加样式代码, 弊端:不能复用
  - 内部: 在 head 标签里面添加 style 标签,在标签体内通过选择器找到需要添加样式的元素,然后再添加样式代码, 这种用法可以复用,但是只能当前页面复用,不能多页面复用
  - 外部: 在单独的 css 样式文件中写样式代码,在 HTML 页面中通过 link 标签引入, 可以实现多页面复用

选择器

- 作用: 用来查找元素的, 找到之后才能添加样式

1. 标签名选择器: 通过标签的名称选择页面中所有同名元素
   - 格式: 标签名{样式代码}
2. id 选择器: 通过页面中元素的 id 选择元素, 元素的 id 是唯一标识不能重复
   - 格式: #id{样式代码}
3. 类选择器: 如果需要选择多个不相关的元素, 可以给多个元素添加相同的 class 属性值,然后通过类选择器进行选择

- 格式: .class{样式代码}

4. 分组选择器: 将多个选择器合并成一个选择器
   - 格式: div,#id,.class{样式代码}
5. 属性选择器: 通过元素的属性选择元素
   - 格式: 标签名[属性名='值']{样式代码}
6. 任意元素选择器: 选取页面中所有标签
   - 格式: *{样式代码}

选择器练习

- 通过内部样式实现以下功能

1. 把张学友改成绿色
2. 把刘德华和悟空改成蓝色
3. 修改取经 3 人和刘备的背景为黄色
4. 修改提交按钮的背景为红色,
自定义按钮的字为粉色
5. 给所有元素添加红色的边框

day02

选择器续

1. 子孙后代选择器: 通过元素和元素之间的关系匹配元素
- 格式: body div div p{样式代码} 匹配 body 里面的 div 里面的 div 里面的所有 p(包括后代)
2. 子元素选择器:通过元素和元素之间的关系匹配元素
- 格式: body>div>div>p{样式代码}匹配 body 里面的 div 里面的 div 里面的 p 子元素(不包括后代)
3. 伪类选择器: 此选择器选择的是元素的状态,元素状态包括: 未访问,访问过状态,悬停状态,点击/激活状态
- 格式: a:link/visited/hover/active{样式代码}

通过内部样式实现以下效果
1. 关羽绿色
2. 张飞和苹果黄色

3. 文本框和所有水果背景红色
4. p2 字体粉色
5. p2 和 p3 背景黄色
6. 腾讯官网未访问绿色,访问过红色

悬停黄色,点击粉色

颜色赋值

- 三原色 RGB RedGreenBlue ，每种颜色的取值范围 0-255
- 五种赋值方式:
  - 颜色单词赋值: red/green/blue/yellow/pink....
  - 6 位 16 进制: #ff 00 00
  - 3 位 16 进制: #f00
  - 3 位 10 进制: rgb(255,0,0)
  - 4 位 10 进制: rgba(255,0,0,0-1) a=alpha 透明度 值越小越透明

背景图片

- background-image:url("路径") 设置背景图片
- background-size:100px 200px; 设置背景图片尺寸
- background-repeat: no-repeat; 禁止重复
- background-position:200px 100px; 设置背景图片位置
- background-position:50% 100%; 设置背景图片位置

文本和字体相关样式

- text-align:left/right/center; 水平对齐方式
- text-decoration:overline/underline/line-through/none; 文本修饰
- line-height:20px; 设置行高
- text-shadow:颜色 x 偏移值 y 偏移值 模糊度; 设置阴影
- font-size:20px; 字体大小
- font-weight:bold 加粗/normal 去掉加粗;
- font-style:italic; 斜体
- font-family:xxx,xxx,xxx; 设置字体
- font: 20px xxx,xxx,xxx; 字体大小+字体设置

文本和字体相关练习

1. 刘德华 宽度 100 高度 30 绿色背景红色字体

横向和纵向居中

2. 苹果和香蕉 字体大小 25px 斜体

3. 冰箱去掉加粗 蓝色阴影

方向是左下 浓度 3

4. 洗衣机添加下划线

5. 百度去掉下划线字体加粗

字体大小 20px

元素的显示方式 display

- block: 块级元素的默认值, 特点: 独占一行,可以修改宽高,包括: h1-h6,p, div
- inline: 行内元素的默认值, 特点: 共占一行,不能修改宽高, 包括: span,b 加粗,i 斜体,u 下划线,s 删除线,a 超链接等
- inline-block:行内块元素的默认值, 特点:共占一行,并且可以修改宽高, 包括:input,img
- none: 隐藏元素
- 行内元素不能修宽高,如必须要修改则需要设置为 block 块级元素或 inline-block 行内块元素

盒子模型

- 盒子模型= content 内容+margin 外边距+padding 内边距+border 边框
- 作用: 控制元素的显示效果
  - content 内容: 控制元素的显示大小
  - margin 外边距: 控制元素的显示位置
  - padding 内边距: 控制元素内容的位置
  - border 边框: 控制元素的边框效果

盒子模型之 Content 内容

- 通过 width 和 height 设置元素的内容大小
- 两种赋值方式
  - 像素
  - 上级元素的百分比
- 行内元素不能修改宽高,如必须要修改则需要设置为 block 块级元素或 inline-block 行内块元

素

盒子模型之 Margin 外边距

- 作用: 控制元素的显示位置
- 赋值方式:
  - margin-left/right/top/bottom:10px; 单独某一个方向赋值
  - margin:20px; 四个方向赋值
  - margin:10px 20px; 上下和左右赋值
  - margin:10px 20px 30px 40px; 上右下左顺时针赋值
- 行内元素上下外边距无效
- 上下相邻彼此添加外边距 取最大值，左右相邻 两者相加
- 粘连问题: 当元素的上边缘和上级元素的上边缘重叠时,给元素添加上外边距会出现粘连问题,给上级元素添加 overflow:hidden 解决
- 部分标签自带外边距,比如: h1-h6 内容标题, p 段落标签, 列表标签,body

盒子模型之 border 边框

- 作用: 控制元素边框的效果
- 赋值方式:
  - border:粗细 样式 颜色;　　四个方向添加边框
  - border-left/right/top/bottom:粗细 样式 颜色;　单独某一个方向添加边框
- border-radius:10px;　设置圆角　值越大越圆，超过宽高的一半时为正圆

盒子模型之 padding 内边距

- 作用: 控制元素内容的位置
- 赋值方式:　和外边距类似
  - padding-left/right/top/bottom:10px; 单独某个方向添加
  - padding:10px;　四个方向添加
  - padding:10px 20px;　上下和左右
  - padding:10px 20p 30px 40px;　上右下左顺时针添加
- 给元素添加内边距会影响元素的宽高,box-sizing: border-box;给元素添加此样式后边框和内边距则不再影响宽高
- 列表标签自带内边距

CSS 的三大特性

- 继承性: 元素可以继承上级元素文本和字体相关的样式, 部分标签自带效果不受继承影响,比如超链接字体颜色
- 层叠性: 多个选择器可以选择同一个元素, 如果添加的样式不同,则全部层叠生效, 如果添加的样式相同则由优先级决定哪个生效
- 优先级: 指选择器的优先级, 作用范围越小优先级越高

  !important>id 选择器>类选择器>标签名选择器>继承(因为继承属于间接选中)

day03

居中相关

- text-align:center; 让元素的文本内容和行内子元素居中 ,不能让块级子元素居中
- margin:0 auto; 让块级元素自身居中

定位方式

- 包括:
  - 静态定位
  - 相对定位
  - 绝对定位
  - 固定定位
  - 浮动定位

静态定位

- 元素默认的定位方式
- 特点: 元素以左上为基准,行内元素从左向右依次排列, 块级元素从上往下依次排列,一般情况下不能实现元素层叠效果. 通过外边距控制元素的位置
- 格式: position:static;

相对定位

- 格式: position:relative;
- 特点: 元素不脱离文档流(不管元素显示到什么位置都占着原来的位置),通过 left/right/top/bottom 控制元素的位置, 让元素相对于初始位置做偏移.

- 应用场景: 当需要对某个元素位置进行调整,并且不影响其它元素时使用

绝对定位

- 格式: position:absolute
- 特点: 元素脱离文档流(不占原来的位置) ,通过 left/right/top/bottom 控制元素的位置,让元素相对于窗口(默认)或某一个上级元素做偏移,如果需要相对于某一个上级元素则必须把上级元素改成相对定位作为参照物.
- 应用场景: 当需要实现层叠效果,让元素以页面中某一个上级元素为参照物时使用绝对定位

固定定位

- 格式: position:fixed;
- 特点: 元素脱离文档流, 通过 left/right/top/bottom 控制元素的位置,让元素相对于窗口做位置偏移.
- 应用场景: 当需要让某个元素固定在窗口某个位置时使用.

浮动定位

- 格式: float:left/right;
- 特点: 元素脱离文档流, 从当前所在行向左或向右浮动, 当撞到上级元素边缘或其它浮动元素时停止
- 浮动元素一行装不下时会自动折行, 折行时有可能被卡住.
- 当某个元素的所有子元素全部浮动时, 元素自动识别的高度为 0,后面的元素会顶上来并且会把文本内容挤到旁边位置显示, 通过给元素添加 overflow:hidden 解决此问题
- 应用场景: 当需要将纵向排列的元素改成横向排列时使用.

day04

溢出设置 overflow

- visible 显示(默认)
- hidden 隐藏
- scroll 滚动显示

行内元素垂直对其方式 vertical-align

- top 上对齐
- middle 中间对齐

- bottom 下对齐
- baseline 基线对齐

显示层级 z-index

- 当元素脱离文档流出现层叠显示时,可以通过 z-index 设置显示层级 z-index 的值越大显示约靠前
- 此样式只能添加给非静态定位的元素,静态定位添加无效

JavaScript

- 作用: 给页面添加动态效果
- 语言特点:
  - 基于面向对象的语言
  - 属于弱类型语言
  - 属于脚本语言, 不需要编译由浏览器解析执行.
  - 安全性强:JS 语言只能访问浏览器内部的数据,浏览器以外的禁止访问
  - 交互性强: 由于 JS 语言是嵌入到 HTML 页面中,最终执行在客户端的浏览器中的语言,和用户是直接接触, Java 语言是运行在服务器的语言, 用户需要进行交互的话必须通过网络才可以,所以 JS 语言的交互性会更强.

变量,数据类型,运算符,各种语句,数组,方法,

变量

- JS 语言属于弱类型语言,声明变量的时候不需要指定类型
- 通过 let 或 var 声明变量
  - let 声明的变量,作用域和 java 语言类似
  - var 声明的变量,作用域是全局的
  - 举例:
    - java:

      for(int i=0;i<10;i++){

      int j = i+1;

```
}
int x = i+j;    //编译报错, 因为 i 和 j 超出作用域
```

- JS-let:

```
for(let i=0;i<10;i++){
    let j = i+1;
}
let x = i+j;    // 运行时访问不到 i 和 j  因为超出作用域
```

- JS-var:

```
for(var i=0;i<10;i++){
    var j = i+1;
}
var x = i+j;   // 此时可以访问到 i 和 j 的值,因为 var 声明的变量相当于是全局的
```

数据类型

- JavaScript 中只有对象类型
- 常见的对象类型:
  - number 数值: 相当于 Java 中所有数值类型的总和
  - string 字符串: 可以用单引号或双引号修饰
  - boolean 布尔值: true/false
  - undefined 未定义: 当变量只声明不赋值的时候变量为未定义类型
- typeof 变量; 获取变量类型

运算符

- 算术运算符+ - * / %
  - 除法和 java 不一样, JS 语言会自动根据结果转换整数或小数

  - java : int x = 5; int y = 2; int z = x/y; z=2

  - JS : let x =5; let y = 2; let z = x/y; z=2.5 x=4 z=2
- 关系运算符 > < >= <= != == ===
  - ==和===区别: ==先统一等号两边变量的类型 再比较值, ===先比较两个变量的类型,类型一致后再比较值. "666"==666 true "666"===666 false
- 逻辑运算符: && || !
- 赋值运算符: = += -= *= /= %=

- 三目运算符:　　条件?值 1:值 2

各种语句

- if else
- while
- for
- switch　case

如何在 HTML 页面中添加 JS 代码

- 三种添加方式:
  - 内联: 在标签的事件属性中添加 js 代码, 事件触发时执行
  - 内部: 在页面的任意位置写 script 标签,标签体内写 js 代码
  - 外部: 在单独的 js 文件中写 js 代码, 在 html 页面中通过 script 标签的 src 属性引入

方法

- java: public void 方法名(参数列表){方法体}
- JavaScript:
  - function 方法名(参数列表){方法体}
  - let 方法名 = function(参数列表){方法体}
  - let 方法名 = new Function("参数 1","参数 2","参数 3","方法体");

和页面相关的方法

1. 通过选择器找到页面中的元素对象

   let 元素对象 = document.querySelector("选择器");
2. 获取和修改元素的文本内容

   元素对象.innerText = "xxxx";　　修改

   元素对象.innerText　　获取
3. 获取和修改控件的值

   文本框对象.value = "xxx"; 修改

   文本框对象.value　　获取

NaN

- Not a Number, 代表不是一个数字

- isNaN(变量)　返回值 true 代表是 NaN false 代表不是 NaN
- NaN 和任何数值进行任何运算 结果都是 NaN

晚课上课时间 7:40

day05

JavaScript 对象分类

- 内置对象: 包括 string , number,boolean 等
- BOM: 浏览器对象模型
- DOM: 文档对象模型

BOM

- Browser Object Model: 浏览器对象模型, 包含和浏览器相关的内容.
- window 对象: 此对象里面的属性和方法称为全局属性和全局方法, 访问时可以省略 window.
- window 中常见的方法:
  - isNaN(); 判断变量是否是 NaN
  - parseInt()/parseFloat(); 将字符串转成整数或小数, 还可以将小数转成整数
  - alert(); 弹出提示框
  - confirm();弹出确认框
  - prompt(); 弹出文本框
  - let timer = setInterval(方法,时间间隔); 开启定时器
  - clearInterval(timer); 停止定时器
  - setTimeout(方法,时间间隔); 开启只执行一次的定时器
- window 中常见的属性:
  - location: 位置
    - location.href　获取和修改浏览器的请求地址
    - location.reload()　　重新加载/刷新

- history: 历史
  - history.length 得到历史页面数量
  - history.back() 返回上一页面
  - history.forward() 前往下一页面

DOM

- Document Object Model 文档对象模型, 包含和页面相关的内容

1. 通过选择器找到页面中的元素对象

   let 元素对象 = document.querySelector("选择器");

2. 获取和修改元素的文本内容

   元素对象.innerText = "xxxx"; 修改

   元素对象.innerText 获取

3. 获取和修改控件的值

   文本框对象.value = "xxx"; 修改

   文本框对象.value 获取

4. 创建元素对象

   let 元素对象 = document.createElement("标签名");

5. 添加元素对象

   已经在页面中显示的元素对象.append(元素对象);

6. 获取页面 body 元素对象

   document.body

前端 MVC 设计模式

- MVC 设计模式就是将实现一个业务的所有代码划分为 3 部分.
- M: Model 模型, 数据模型 指和数据相关的代码
- V:View 视图, 指和页面相关的代码
- C:Controller 控制器, 指将数据展示到页面中的过程代码
- MVC 设计模式的 Controller 中需要频繁的进 DOM 操作(查找页面中元素,创建元素等),会浪费资源,MVVM 设计模式可以解决此问题

前端 MVVM 设计模式

- MVVM 设计模式也是将实现一个业务的所有代码划分为 3 部分.
- M:Model 模型, 数据模型 指和数据相关的代码

- V:View 视图，指和页面相关的代码
- VM: ViewModel 视图模型，负责将页面中可能发生改变的元素和变量在内存中进行绑定，当需要改变页面中元素时,只需要修改变量，视图模型会不断监听变量值的改变，当值发生改变时会从内存中找到对应元素让其跟着改变

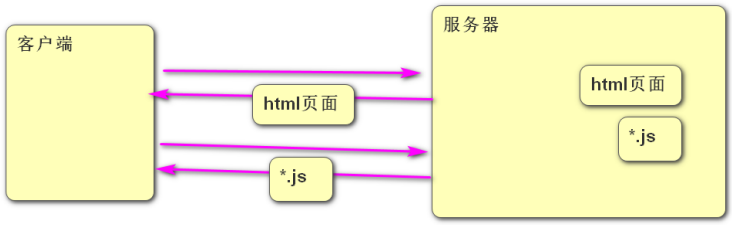
VUE

- 目前最流行的前端框架，此框架基于 MVVM 设计模式
- 两种使用方式:
  - vue.js 文件引入 html 页面中使用
  - 脚手架环境下使用

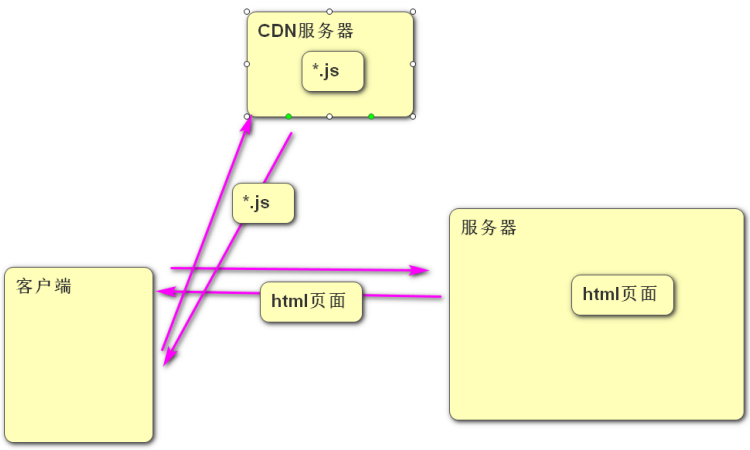
以下推荐国外比较稳定的两个 CDN，国内还没发现哪一家比较好，目前还是建议下载到本地。

- **Staticfile CDN（国内）**：https://cdn.staticfile.org/vue/2.2.2/vue.min.js
- **unpkg**：https://unpkg.com/vue@2.6.14/dist/vue.min.js
- **cdnjs**：https://cdnjs.cloudflare.com/ajax/libs/vue/2.1.8/vue.min.js

- 引入外部的 js 文件两种方式:
  - 把 js 文件下载到本地 然后引入的到自己的页面中



  - 直接引入一个 CDN 服务的地址



VUE 框架的工作流程

- VUE 框架中 创建的 Vue 对象相当于是 MVVM 设计模式中的 VM 视图模型, 该对象会将页面中可能发生改变的元素和变量在内存中进行绑定, 并且会一直监听变量值的改变, 当值发生改变时会从内存中的对应关系找到对应的页面元素,并对其进行修改, 以后需要改变页面时只需要找到页面对应的变量,让变量改动 页面会自动跟着改变

VUE 框架中的各种指令

- {{变量}} , 插值: 让此处的文本和变量进行绑定 , 不写在标签里面也可以使用
- v-text="变量", 让元素的文本内容和变量进行绑定
- v-html="变量", 让元素的 html 标签内容 和变量进行绑定
- v-bind:属性名="变量", 让元素某个属性的值和变量进行绑定 可以省略 v-bind 直接写:属性名进行绑定

day06

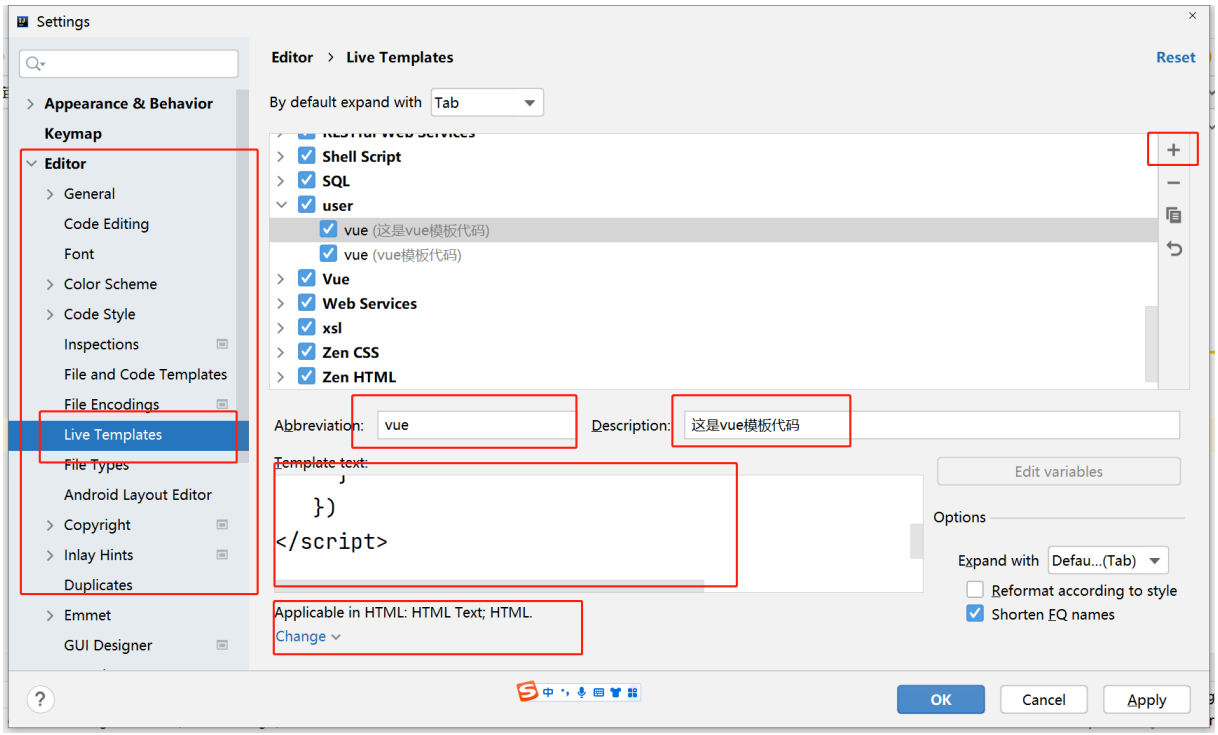
Vue 框架中的各种指令(续)

- v-model="变量", 让控件的值和变量进行双向绑定, 控件的值改变变量的值会跟着改变,同时变量的值改变也会影响控件的值, 应用场景: 当需要获取控件的值的时候使用双向绑定
- v-for="(对象,下标) in 数组" 让显示的元素内容和数组进行绑定
- v-if="变量" 让元素是否显示和变量进行绑定, true 显示, false 不显示(删除元素)
- v-else 让元素的显示状态和 v-if 元素的状态相反
- v-show="变量" 让元素是否显示和变量进行绑定, true 显示, false 不显示(隐藏)
- v-on:事件名="方法" 事件绑定, 绑定的方法必须写在 Vue 对象中的 methods 属性里面, @事件名="方法" 简写

ElementUI

- 目前比较流行的前端框架, 基于 HTML,CSS,JavaScript,Vue 的一款前端框架, 可以让程序员高效的开发出前端页面

自定义模板代码


Settings
Editor › Live Templates
Reset
By default expand with Tab
Appearance & Behavior
Keymap
Editor
General
Code Editing
Font
Color Scheme
Code Style
Inspections
File and Code Templates
File Encodings
Live Templates
File Types
Android Layout Editor
Copyright
Inlay Hints
Duplicates
Emmet
GUI Designer
Shell Script
SQL
user
vue (这是vue模板代码)
vue (vue模板代码)
Vue
Web Services
xsl
Zen CSS
Zen HTML
Abbreviation: vue
Description: 这是vue模板代码
Template text:
})
</script>
Edit variables
Options
Expand with Defau...(Tab)
Reformat according to style
Shorten FQ names
Applicable in HTML: HTML Text; HTML.
Change
OK
Cancel
Apply