目录

Chapter 1

CSS

1.1 CSS颜色

1.2 元素显示模式

元素显示模式就是元素（标签）以什么方式进行显示，比如说div自己占一行，而span却可以一行内写多个。

HTML元素一般分为块元素和行内元素两种类型。

1.2.1 块元素

常见的块元素有h1-h6，p，div，ul，ol，li，其中标签是最典型的块元素。（似乎暗示自带换行的功能？）

块级元素的特点：

- 自己独占一行；
- 高度、宽度、外边距预计内边距都可以控制；
- 宽度默认是容器（父级宽度）的100%；
- 是一个容器及盒子，里面可以放行内或者块级元素；

注意：

- 文字类的元素不能使用块级元素；
- p标签主要用于存放文字，因此里面不能放块级元素；
- h1-h6都是文字类块级标签，里面也不能放其他块级元素；

1.2.2 行内元素（内联元素）

常见的行内元素有a、strong、b、em、i、del、s、ins、u、span等，其中span标签是最典型的行内元素。

行内元素的特点：

- 相邻行内元素在一行上可以显示多个；
- 高、宽直接设置是无效的；
- 默认宽度就是元素本身内容的宽度；
- 行内元素只能容纳文本或其他行内元素；

注意：

- 链接里面不能再放链接；
- 特殊情况链接a里面可以放跨级元素，但是给a转换一下块级模式最安全；

1.3 CSS背景

1.3.1 背景图像固定（背景附着）

background-attachment属性设置背景图像是否是固定或者随着页面

1.3.2 背景复合写法

当使用简写属性时，没有特定的书写顺序，一般习惯按照约定顺序为：

`backgroud: color url repeat attachment position`

1.3.3 背景色半透明

CSS3为我们提供了背景颜色半透明的效果。

```
1    backgroud: rgba(0, 0, 0, .3);
```

- 最后一个参数时alpha透明度，取值范围在0 – 1之间；
- 背景半透明是指盒子背景半透明，盒子里面的内容不受影响；

表 1.1: 选择器权重

选择器	选择器权重
继承或者*	0,0,0,0
元素选择器	0,0,0,1
类选择器、伪类选择器	0,0,1,0
ID选择器	0,1,0,0
行内样式style=""	1,0,0,0
!important	无穷大

1.3.4 背景总结

属性	作用	值
background-color	背景颜色	预定义的颜色值或十六进制或RGB值
background-image	背景图片	url(path)
background-repeat	是否平铺	repeat/no-repeat/repeat-x/repeat-y
background-position	背景位置	length/position 分别是x和y的坐标
background-attachment	背景附着	scroll（背景滚动） fixed（背景固定）
背景简写	书写更简单	背景颜色背景图片背景平铺背景滚动背景位置
背景色半透明	背景颜色半透明	background: rgba(red, green, blue, alpha)

1.4 CSS三大样式

CSS有三个非常重要的三个特性：层叠性、继承性、优先级。

1.4.1 层叠性

相同选择器设置相同的样式时，此时新的样式就会覆盖另一个冲突的样式，层叠性主要解决样式冲突的问题。

层叠性原则：

- 样式冲突，遵顼的原则是就近原则，哪个样式离结构近，就执行哪个样式；
- 样式不冲突，不会层叠；

1.4.2 继承性

1.4.3 优先级

??显示了选择器的权重

1. 继承的权重是0，如果该元素没有直接选中，不管父元素权重多高，子元素得到的权重都是0。

2. a链接浏览器默认指定列一个样式：蓝色的 有下划线

权重叠加

1.5 盒子模型

1.5.1 盒子模型的组成

所谓盒子模型，就是把HTML页面中的布局元素看作是一个矩形的盒子，也就是一个盛装内容的容器。

CSS盒子模型本质是一个盒子，封装周围的HTML元素，它包括：边框（border），外边距（margin），内边距（padding）和实际内容（content）。

边框（**border**）

表格的细线边框 border-collapse属性控制连六七绘制表格边框的方式，它控制相邻单元格的边框。

语法格式为：

```
1     border-collapse: collapse;
```

边框会影响盒子的实际大小

内边距（**padding**）

padding属性用于设置内边距，即边框与内容之间的距离。

如果盒子本身没有指定width/height属性，则此时padding不会撑开盒子大小。

外边距（**margin**）

margin属性用于设置外边距，即控制盒子与盒子之间的距离。

属性	作用
margin-left	左外边距
margin-right	右外边距
margin-top	上外边距
margin-bottom	下外边距

清除外边距

网页元素很多都带有默认的内外边距，而且不同浏览器默认的值也不一样，因此我们在布局前，首先要清除网页元素的内外边距。

```
* {
    padding: 0;
    margin: 0;
}
```

注意：行内元素尽量只设置左右的内外边距，不要设置上下内外边距。但是转换为块级或者块元素就可以了。

1.5.2 圆角边框

border-radius属性可以设置元素的外边框圆角。
语法格式为：

```
border-radius: length|pct;
```

- 参数值可以为数值或者百分比的形式；
- 如果是正方形，想要一个圆，课数值修改为高度或宽度的一般即可，或者直接写50%；
- 如果是一个矩形，设置为高度的一般即可；
- 该属性是一个简写属性，可以写四个值，分别表示左上角、右上角、右下角、左下角；
- 分开写：border-top-left-radius、border-top-right-radius、border-bottom-right-radius、border-bottom-left-radius；

1.5.3 盒子阴影（重点）

box-shadow属性可以为盒子添加阴影。
语法格式为：

```
box-shadow: h-shadow v-shadow blur spread color inset;
```

属性	描述
h-shadow	必需，水平阴影的位置，允许负值
v-shadow	必需，垂直阴影的位置，允许负值
blur	可选，模糊的清晰度
spread	可选，阴影的尺寸
color	可选，阴影的颜色，参见??
inset	可选，将外部阴影（默认）改为内部阴影

- 默认为外部阴影（outset），这个值不写，如果写了将导致阴影无效；
- 盒子阴影不占用控件，不会印象其他盒子排列；

1.6 文字阴影（应用不多）

text-shadow属性将阴影应用于文本。语法格式为：

```
box-shadow: h-shadow v-shadow blur color;
```

属性	描述
h-shadow	必需，水平阴影的位置，允许负值
v-shadow	必需，垂直阴影的位置，允许负值
blur	可选，模糊的清晰度
color	可选，阴影的颜色，参见??

1.7 CSS浮动

1.7.1 标准流（普通流/文档流）

标准流就是标签按照规定好默认的方式排序。

1.7.2 浮动

网页布局第一准则：多个块级元素纵向排列找标准流，多个块级元素横向排列找浮动。

float属性用于创建浮动框，将其移动到一边，直到左边缘或右边缘触及包含块或者另一个浮动框的边缘。

语法格式为：

```
selector {
float: value;
}
```

属性值	描述
none	元素不浮动（默认值）
left	元素向左浮动
right	元素向右浮动

1.7.3 浮动特性

1. 浮动元素会脱离标准流（拖标），浮动的盒子不再保留原先的位置；
2. 浮动元素会一行内显示并且元素顶部对齐；
3. 浮动元素会具有行内块元素的特性

- 如果行内元素有了浮动，则不需要转换为行内块元素就可以直接指定高度和宽度；
- 如果块级盒子没有设置宽度，默认宽度和父级宽度一样宽，但是添加浮动后，它的大小根据内容来决定；
- 浮动的盒子中间是没有缝隙的，是紧挨在一起的；
- 行内元素同理；

为了约束浮动元素的位置，网页布局一般采取的策略是：先用标准流的父元素排列上下位置，之后内部子元素采取浮动排列左右位置，符合网页布局第一准则。

1.8 常见网页布局

1.9 清除浮动

清除浮动的本质就是清除浮动元素造成的影响。如果父盒子本身有高度，则不需要清除浮动。清除浮动之后，父级就会根据浮动的子盒子自动检测高度，父级指定高度之后，就不会影响下面的标准流了。

语法格式为：

```
selector{
clear: value;
}
```

属性值	描述
left	元素向左浮动
right	元素向右浮动
both	同时清除左右两侧浮动的影响

清除浮动的策略是：闭合浮动。

1.9.1 清除浮动的方法

1. 额外标签法也称为隔墙法，是W3C推荐的做法；
2. 父级添加overflow属性；
3. 父级添加after伪元素；
4. 父级添加双伪元素；

父级添加overflow属性 可以给父级添加overflow属性，将其属性值设置为hidden、auto和scroll。代码简洁，但是无法显示溢出内容。

:after伪元素法

添加双伪元素

1.10 CSS属性书写顺序（重点）

1. 布局定位属性：display/position/float/clear/visibility/overflow；
2. 自身属性：width/height/margin/padding/border/background；
3. 文本属性：color/font/text-decoration/text-align/vertical-align/white-space/break-word；
4. 其他属性(CSS3): content/cursor/border-radius/box-shaow/text-shadow/background: linear-gradient；

1.11 页面布局整体思路

为了提高网页制作的效率，布局时通常有以下的整体思路：

1. 必须确定页面的版心（可视区）；
2. 分析页面中的行模块以及每个行模块中的列模块；
3. 一行中的列模块经常浮动布局，先确定每个列的大小，之后确定列的位置；
4. 制作HTML结构，遵循先有结构，后有样式的原则，结构最重要；

Chapter 2

CSS定位

浮动可以让多个块级盒子在一行内没有缝隙排列显示，经常用于横向排列盒子。定位则是可以让盒子自由的在某个盒子内移动位置或者固定屏幕中某个位置，并且可以压住其他盒子。

2.1 定位组成

定位是将盒子定在某个位置，所以定位也是摆放盒子，按照定位的方式移动盒子。
定位由定位模式和边偏移两部分组成：

- 定位模式用于指定一个元素在文档中的定位方式；
- 边偏移决定改元素的最终位置；

2.1.1 定位模式

定位模式决定元素的定位方式，通过CSS的`position`属性来设置，可取以下四个值：

表 2.1: 定位模式属性值

属性值	描述
`static`	静态定位
`relative`	相对定位
`absolute`	绝对定位
`fixed`	固定定位

静态定位static

静态定位是元素的默认定位方式，无定位的意思。
语法格式为：

```
selector {
position: static;
}
```

静态定位按照标准流特性拜访位置，它没有边偏移，在布局中很少用到。

相对定位relative

相对定位是元素在移动位置的时候，是相对于它原来的位置来参考（自恋型）。
语法格式为：

```
selector {
position: relative;
}
```

特点：

- 位置移动参考自己原来所在的位置；
- 原来在标准流中的位置继续占有，后面的盒子仍然以标准流的方式对待它（不脱标，继续保留原来位置）；

绝对定位absolute

绝对定位是元素在移动位置的时候，相对于它祖先元素来说的。
语法格式为：

```
selector {
position: absolute;
}
```

特点：

- 如果没有祖先元素或者祖先元素没有定位，则以浏览器为准定位（Document文档）；
- 如果祖先元素有定位，则以最近一级的有定位祖先元素为参考点移动位置；
- 绝对定位不占有原先的位置（脱标）；

子绝父相口诀

子绝父相的意思是子级是绝对定位的话，父级要用相对定位。

子级绝对定位，不会占用位置，可以放到父盒子里面的任何一个地方，不会影响其他的兄弟盒子。父盒子需要加定位限制子盒子在父盒子内显示。父盒子布局时，需要占用位置，因此父盒子只能是相对定位。

固定定位fixed

固定定位是元素固定于浏览器可视区的位置，主要使用场景是可以在浏览器页面滚动时元素的位置不会改变。语法格式为：

```
selector {
position: fixed;
}
```

固定定位特点：

- 以浏览器的可视窗口为参照点移动元素；
- 与父元素没有任何关系；
- 不随滚动条滚动；
- 固定定位不再占有原先的位置；

固定定位是脱标的，其实固定定位也可以看作是一种特殊的绝对定位。

如果想要在版心右侧使用固定定位，首先让固定定位的盒子`left: 50%`，移动到浏览器可视区域的一般位置，再让固定定位的盒子`margin-left: 50% of core`。

粘性定位sticky

粘性定位可以被认为是相对定位和固定定位的混合。

粘性定位语法格式为：

```
selector {
position: sticky;
}
```

粘性定位的特点：

- 以浏览器的可视窗口为参考点移动元素（固定定位特点）；
- 粘性定位占有原先的位置（相对定位特点）；
- 必须添加**??**中的一个；

2.1.2 边偏移

边偏移决定定位的盒子移动的最终位置，主要由top、bottom、left、right4个属性：

表 2.2: 边偏移属性

属性值	示例	描述
top	top: 80px	顶端偏移量，定义元素相对于父元素上边线的距离
bottom	bottom: 80px	底部偏移量，定义元素相对于父元素上边线的距离
left	left: 80px	左侧偏移量，定义元素相对于父元素左边线的距离
right	right: 80px	右侧偏移量，定义元素相对于父元素右边线的距离

2.1.3 定位的总结

表 2.3: 定位总结

定位模式	是否脱标	移动位置	是否常用
静态定位static	否	不能使用边偏移	很少
相对定位relative	否（占有位置）	相对于自身位置移动	常用
绝对定位absolute	是（不占位置）	带有定位的父级	常用
固定定位fixed	是（不占位置）	浏览器可视区	常用
粘性定位sticky	否（占有位置）	浏览器可视区	当前阶段少

2.1.4 定位叠放次序z-index

在使用定位布局时，可能会出现盒子重叠的情况，此时可以使用z-index来控制盒子的前后次序（z轴）。

语法格式为：

```
selector {
z-index: 1;
}
```

数值可以是正整数、负整数或者0，默认是auto，数值越大、盒子越靠上。如果属性值相同，则按照章节书写顺序，后来居上，数字后面不能加单位，并且只有有定位属性的盒子才有z-index属性。

2.1.5 定位的拓展

绝对定位盒子的水平垂直居中方法

如果想要在版心右侧使用固定定位，首先让固定定位的盒子`left: 50%`，移动到浏览器可视区域的一般位置，再让固定定位的盒子`margin-left: -50% of width`。

定位特殊特性

- 行内元素添加绝对或固定定位，可以直接设置高度和宽度；

- 块级元素添加绝对或者固定定位，如果不该宽度或者高度，默认大小是内容的大小；

绝对定位/固定定位会完全压住盒子

浮动元素是不同的，尽会压住它下面标准流的盒子，但是不会压住下面标准流盒子里面的文字以及图片，但是绝对定位（固定定位）会压住下面标准流所有的内容。

浮动之所以不会压住文字，是因为浮动产生的最初目的是为了左文字环绕效果的，文字会围绕浮动元素。

2.2 元素的显示与隐藏

类似网站中的广告，当点击叉号关闭后，广告消失，但是重新刷新页面，则会重新出现广告，本质就是一个元素在页面中的隐藏于显示。

2.2.1 display显示隐藏

display属性用于设置一个元素应如何显示。display的取值很多，但是最常用的取值为none（隐藏对象）、block（除了转换为块级元素之外，同时还有显示元素的意思）。

display隐藏元素之后，不再占有原来的位置。

2.2.2 visibility显示隐藏

visibility元素用于指定一个元素应可见还是隐藏，常见参数为visible（元素可见）、hidden（元素隐藏）。

visibility隐藏元素后，继续占有原来的位置。如果隐藏元素想要继续占有原来的位置，使用`visibility: hidden`，不想继续占有原来的位置，使用`display: none`。

2.2.3 overflow溢出显示隐藏

overflow属性规定当内容溢出元素框时发生的事情。

表 2.4: overflow属性值

值	描述
visible	默认值。内容不会被修剪，会呈现在元素框之外。
hidden	内容会被修剪，并且其余内容是不可见的。
scroll	内容会被修剪，但是浏览器会显示滚动条以便查看其余的内容。没有溢出时，滚动条也显示。
auto	如果内容被修剪，则浏览器会显示滚动条以便查看其余的内容。没有溢出时，滚动条不显示。
inherit	规定应该从父元素继承 overflow 属性的值。

如果有定位的盒子，慎用`overflow: hidden`，因为它会隐藏多余的部分。

Chapter 3

CSS高级技巧

3.1 精灵图

一个页面中往往会应用很多小的背景图像作为装饰，当页面中的图像过多时，服务器就会频繁地接收和发送请求图片，造成服务器压力过大，这将打打降低页面的加载速度。因此，为了有效地减少服务器接收和请求的次数，提高页面的加载速度，出现了CSS精灵技术（也称CSS Sprites、CSS雪碧）。

使用精灵图的核心：

- 精灵技术主要针对于背景图片使用，就是把多个小背景图片整合到一张大图片中；
- 移动背景图片位置，使用`background-position`；
- 移动的距离就是这个目标图片的x（从left向right为正）和y（从top向bottom为正）坐标，注意也网页中的坐标有所不同；
- 使用精灵图的时候需要精确测量每个小背景图片的大小和位置；

精灵图使用的图片是多张图片组合而成的图片，将图片位置不断移动，将所在位置的图片内容显示出来。

精灵图缺点：

- 图片文件仍然比较大；
- 图片本身放大或缩小还是会存在失真；
- 一旦图片制作完毕想要更换非常复杂；

3.2 字体图标

字体图标主要用于显示网页中通用、常用的一些小图标。虽然精灵图也可以实现该技术，但是精灵图的缺陷使得其不方便完成，字体图标的出现很好的解决了上面的问题。字体图标可以为前端工程师提供一种方便高效的图标使用方式，展示的效果是图标，但是本质上属于字体。

字体图标是一些网页常见的小图标，可以直接从网上下载，因此使用中需要先下载后引入HTML页面中，对于没有的图标可以进行追加。

推荐下载的网站：

- icomoon;
- 阿里iconfont字体;

3.3 CSS三角

3.4 CSS用户界面样式

3.5

Chapter 4

项目笔记

```
    lintOnSave: false,
```

4.1 Element-UI

4.1.1 Form表单

表单验证 在防止用户犯错的前提下，尽可能让用户更早地发现并纠正错误。

表 4.1: 表单方法

方法名	说明	参数
validate	对整个表单进行校验的方法，参数为一个回调函数。该回调函数会在校验结束后被调用，并传入两个参数：是否校验成功和未通过校验的字段。若不传入回调函数，则会返回一个 promise	validate

Form Methods

4.2 登录和退出功能

4.2.1 登录功能实现

路由导航守卫控制访问权限

如果用户没有登录，但是直接通过URL方位特定页面，需要重新导航到登录页面。

```
// Mount route navigation guard
// to: the path to be accessed
// from: Represents which path it jumped from
// next(): let go, next('/path'): Force jump to path page
router.beforeEach((to, from, next) => {
if (to.path === '/login') return next()
// get token
const tokenStr = window.sessionStorage.getItem('token')
if (!tokenStr) return next('/login')
next()
})
```

4.2.2 退出功能

退出功能原理

基于token的方式实现退出比较简单，只需要销毁本地的token即可，这样后续的请求就不会携带token，必须重新登录生成一个token之后才可以访问页面。

```
// clear token
window.sessionStorage.clear()
// jump to new path
this.$router.push("/path")
```

4.3 主页布局

4.3.1 通过接口获取菜单数据

通过axios请求拦截器添加token，保证拥有获取数据的权限。

4.4 项目处理步骤

1. 用户列表界面的创建（第三天）
   - 添加用户：添加表单、表单内容自定义规则检验、表单重置功能、预验证功能、发起添加用户请求、
2. •
3. •

Part I

Element UI

Chapter 5

开发指南

Chapter 6

组件

6.1 Basic

Switch开关

6.2 Form

6.3 Data

6.3.1 Pagination分页

当数据量过多时，使用分页分解数据。

6.4 Notice

6.4.1 Message消息提示

常用于主动操作后的反馈提示。与 Notification 的区别是后者更多用于系统级通知的被动提醒。

6.5 Navigation

6.6 Others

6.6.1 Dialog对话框

在保留当前页面状态的情况下，告知用户并承载相关操作。

6.6.2 Tooltip文字提示

常用于展示鼠标 hover 时的提示信息。