

# 前端基础笔记

曹思远

2020 年 12 月 30 日

## 目录

<b>1</b>	<b>软件安装</b>	<b>3</b>
1.1	必备软件列表	3
1.1.1	必备软件配置	3
<b>2</b>	<b>Git 入门</b>	<b>5</b>
2.1	命令行入门	5
2.2	本地仓库	5
2.3	远程仓库	5
<b>3</b>	<b>HTML</b>	<b>5</b>
3.1	概览	5
3.2	标签	5
3.3	重难点	5
3.4	实践和手机调试	5
<b>4</b>	<b>CSS</b>	<b>5</b>
4.1	基础	5
4.1.1	语法	5
4.1.2	Border 调试法	5
4.1.3	文档流	5
4.1.4	盒模型	6
4.1.5	margin 合并	6
4.1.6	基本单位	7
4.1.7	练手项目	7
4.2	布局	7
4.2.1	布局分类	7
4.2.2	两种布局思路	8
4.2.3	float 布局	8
4.2.4	Flex 布局	9
4.2.5	Grid 布局	9
4.3	定位	10
4.3.1	一个 div 的分层	10
4.3.2	position 的五个取值	10
4.3.3	层叠上下文	10
4.4	动画	10
4.4.1	动画的原理	10

4.4.2	浏览器渲染的原理 . . . . .	10
4.4.3	CSS 动画优化 . . . . .	11
4.4.4	transition . . . . .	11
4.4.5	红心实践 . . . . .	13
<b>5</b>	<b>HTTP</b>	<b>13</b>
<b>6</b>	<b>JS</b>	<b>13</b>
<b>7</b>	<b>算法与数据结构</b>	<b>13</b>
<b>8</b>	<b>JS 编程接口</b>	<b>13</b>
<b>9</b>	<b>项目</b>	<b>13</b>
<b>10</b>	<b>MVC</b>	<b>13</b>
<b>11</b>	<b>Webpack</b>	<b>13</b>
<b>12</b>	<b>虚拟 DOM 与 DOM diff</b>	<b>13</b>
<b>13</b>	<b>Vue</b>	<b>13</b>
<b>14</b>	<b>React</b>	<b>13</b>
<b>15</b>	<b>NodeJS</b>	<b>13</b>
<b>16</b>	<b>Vue3 造轮子</b>	<b>13</b>

# 1 软件安装

## 1.1 必备软件列表

1. vscode
2. cmdr
3. chrome
4. clash
5. NodeJS, Yarn

### 1.1.1 必备软件配置

任何软件都需要配置

#### vscode

1. 必备插件
  - Chinese (Simplified) Language Pack for Visual Studio Code
  - Code Spell Checker
  - Git Easy
  - Latex Workshop
  - Markdown All in One
  - Prettier - Code formatter
2. 环境和快捷键配置
  - 具体看我知乎保存的 setting.json 和 keybinding.json
3. 快捷键
  - ctrl + p - 找文件
  - ctrl + shift + p 或 F1 - 输命令
  - alt + 单击 - 多位置输入

#### cmdr

1. 配置内容较多，直接看我保存的配置文件。
2. 将 cmdr 加入右键菜单，加入环境变量。

## chrome

### 1. 可选插件

- Proxy SwitchOmega
- uBlock

### 2. 高级用户配置

- 在开发者工具里面按 ESC 可以新建控制台
- Sources 面板可以保存代码片段 (Snippets)
- Network 关掉 show overview, filter 可以搜索, 右击勾选 Method
- Network 可模拟慢网速/断网
- Preserve log 不会清空当前请求数据
- Disable cache 清除缓存

### 3. chrome 常用快捷键

- 鼠标中键单击 - 打开或关闭
- ctrl + T - 新开标签
- ctrl + shift + T - 撤销关闭
- ctrl + 点击 - 在新标签打开
- ctrl + W - 关闭当前标签
- ctrl + Reload 或者 F5 - 刷新
- ctrl + Location - 输入网址
- ctrl + shift + Inspector 或 F12 - 打开开发者工具
- alt + 左右 - 前进后退
- alt + 回车 - 在新标签打开
- shift + 回车 - 在新窗口打开
- ctrl + shift + delete - 删除历史浏览数据

## windows

1. 关掉任务栏无用标签, 卸载无用软件。(如搜索框, 任务视图, 微软小娜 Cortana, 开始菜单里的各种贴图)
2. 可安装 TranslucentTB 使任务栏透明
3. 理解用户目录, 即 C:\Users\Jony, 分别右击用户目录里的下载和文件, 属性-> 位置-> 移动到 E 盘。
4. 显示文件后缀, 打开查看-> 选项-> 查看, 去掉隐藏已知文件扩展名, 勾选显示已知文件, 文件夹和驱动器。
5. 记住组合键

- Win 组合键
  - (a) win + Desktop - 展示桌面
  - (b) win + 方向键 - 移动窗口
  - (c) alt + tab - 切换窗口
  - (d) win + tab - 不怎么常用的切换窗口

(e) win + ctrl + 方向键 - 切换桌面

- Ctrl 组合键

(a) ctrl + All/ctrl + Copy/ctrl + V/ctrl + Z/ctrl + Y

(b) ctrl + Reload/F5

(c) ctrl + P - 打印

## NodeJS, Yarn

### 1. 开发必装的东西

- nrm
- tldr

## 2 Git 入门

### 2.1 命令行入门

### 2.2 本地仓库

### 2.3 远程仓库

## 3 HTML

### 3.1 概览

### 3.2 标签

### 3.3 重难点

### 3.4 实践和手机调试

## 4 CSS

### 4.1 基础

#### 4.1.1 语法

#### 4.1.2 Border 调试法

#### 4.1.3 文档流

##### 1. 文档流的基本概念

- 流动方向
  - inline 元素从左到右，到达最右边才会换行
  - block 元素从上到下，每一个都另起一行
  - inline-block 也是从左到右
- 宽度
  - inline 宽度为内部 inline 元素的和，不能用 width 指定
  - block 默认自动计算宽度，可用 width 指定
  - inline-block 结合前两者特点，可用 width

- 高度
  - inline 高度由 line-height 间接确定，跟 height 无关
  - block 高度由文档流元素决定，可以设 height
  - inline-block 跟 block 类似，可以设置 height

## 2. overflow 溢出

- 当内容大于容器
  - 等内容的宽度或高度大于容器，会溢出
  - 可用 overflow 来设置是否显示滚动条
  - auto 是灵活设置
  - scroll 是永远显示
  - hidden 是直接隐藏溢出部分
  - visible 是直接显示溢出部分
  - overflow 可以分为 overflow-x 和 overflow-y

## 3. 脱离文档流

- 有些元素可不在文档流
  - 原因是 block 高度由内部文档流元素决定，可以设 height
- 以下元素脱离文档流
  - float
  - position:absolute/fixed
- 不用以上属性就不脱离文档流

### 4.1.4 盒模型

#### 1. 两种

- content-box 内容盒 - 内容就是盒子的边界
- border-box 边框盒 - 边框才是盒子的边界

#### 2. 公式

- content-box width = 内容宽度
- border-box width = 内容宽度 + padding + border

#### 3. 哪个好用？

- border-box 好用
- 因为可以同时指定 padding, width, border

### 4.1.5 margin 合并

#### 1. 哪些情况会合并

- 父子 margin 合并
- 兄弟 margin 合并

#### 2. 如何阻止合并

- 父子合并用 padding/border 挡住
- 父子合并用 overflow:hidden 挡住
- 父子合并用 display:flex
- 兄弟合并是符合预期的
- 兄弟合并可以用 inline-block 消除
- css 属性逐年增多，每年都有新的，死记就完事了

#### 4.1.6 基本单位

##### 1. 长度单位

- px 像素
- em 相对于自身 font-size 的倍数
- 百分数
- 整数
- rem
- vw 和 vh

##### 2. 颜色

- 十六进制 #FF6600 或者 #F60
- RGBA 颜色 rgb(0,0,0) 或者 rgba(0,0,0,1)
- hsl 颜色 hsl(360,100%, 100%)

#### 4.1.7 练手项目

彩虹 demo

<http://js.jirengu.com/xacet/1/edit?html,css,output>

## 4.2 布局

### 4.2.1 布局分类

#### 1. 两种

- 固定宽度布局，一般宽度为 960/1000/1024px
- 不固定宽度布局，主要靠文档流的原来布局

#### 2. 回顾

- 文档流本来就是自适应的，不需要加额外的样式

#### 3. 响应式布局

- PC 上固定宽度，手机上不固定宽度
- 也就是一种混合布局

### 4.2.2 两种布局思路

#### 1. 从大到小

- 先定下大局
- 然后完善每个部分的小布局

#### 2. 从小到大

- 先完成小布局
- 然后组合成大布局

#### 3. 两种均可

- 新人推荐第二种，因为小的简单
- 老手一般用第一种，因为熟练有大局观

### 4.2.3 float 布局

一图流 (图片以后贴出)

#### 1. float 布局

- 子元素加上 float:left 和 width
- 在父元素上加.clearfix

#### 2. float 布局经验

- 留一些空间或者最后一个不设 width
- 不需要做响应式，因为手机上没有 IE，而这个布局是专门为 IE 准备的
- 解决 IE6/7 存在的双倍 margin bug 如下
- 一是将错就错，针对 IE6/7 把 margin 减半
- 二是神来一笔，再加一个 display:inline-block

### float 布局实践

#### 1. 不同布局

- 用 float 做两栏布局 (如顶部条)
- 用 float 做三栏布局 (如内容区)
- 用 float 做四栏布局 (如导航)
- 用 float 做平均布局 (如产品展示区)
- <http://js.jirengu.com/wiwuh/3/edit?html,css,output>

#### 2. 实践经验

- 加上头尾，即可满足所有 PC 页面需求
- 手机页面傻子采用 float
- float 要程序员自己计算宽度，不灵活
- float 用来应付 IE 足矣



#### 4.2.4 Flex 布局

##### 1. 重点

- display: flex
- flex-direction: row/column
- flex-wrap: wrap
- justify-content: center/space-between
- align-items: center

##### 2. 颜色

- 十六进制 #FF6600 或者 #F60
- RGBA 颜色 rgb(0,0,0) 或者 rgba(0,0,0,1)
- hsl 颜色 hsl(360,100%, 100%)

##### 3. 实践

- 用 flex 做两栏布局
- 用 flex 做三栏布局
- 用 flex 做四栏布局
- 用 flex 做平均布局
- 用 flex 组合使用，做更复杂的布局
- <http://js.jirengu.com/pakez/2/edit?html,css,output>

##### 4. 经验

- 永远不要把 width 和 height 写死，除非特殊说明
- 用 min-width/max-width/min-height/max-height
- flex 可以基本满足所有需求
- flex 和 margin-xxx: auto 配合有意外的效果

##### 5. 什么是写死

- width: 100px

##### 6. 不写死

- width: 50%
- max-width: 100px
- width: 30vw
- min-width: 80%
- 特点：不使用 px，或者加 min max 前缀

#### 4.2.5 Grid 布局

二维布局用 Grid，一维布局用 Flex

#### 语法

## 例子和语法

### 4.3 定位

布局与定位的区别是：布局是屏幕平面上的，定位是垂直于屏幕的

#### 4.3.1 一个 div 的分层

#### 4.3.2 position 的五个取值

#### 4.3.3 层叠上下文

### 4.4 动画

#### 4.4.1 动画的原理

#### 4.4.2 浏览器渲染的原理

#### 浏览器渲染过程

1. 根据 HTML 构建 HTML 树 (DOM)
2. 根据 CSS 构建 CSS 树 (CSSOM)
3. 将两颗树合并成一颗渲染树 (render tree)
4. Layout 布局 (文档流，盒模型，计算大小和位置)
5. Paint 绘制 (把边框颜色，文字颜色，阴影等画出来)
6. Compose 合成 (根据层叠关系展示画面)

#### 三棵树 图片以后放

如何更新样式 一般我们采用 JS 来更新样式

1. 比如 `div.style.background='red'`
2. 比如 `div.style.display='none'`
3. 比如 `div.classList.add('red')`
4. 比如 `div.remove()` 直接删掉节点

#### 三种更新方式

1. JS/CSS > 样式 > 布局 > 绘制 > 合成
2. JS/CSS > 样式 > 绘制 > 合成
3. JS/CSS > 样式 > 合成

### 三种更新方式区别

#### 1. 第一种，全走

- `div.remove()` 会触发当前消失，其他元素 `relayout`
- <http://js.jirengu.com/jagel/1/edit?html,css,js,output>

#### 2. 第二种，跳过 layout

- 改变背景颜色，直接 `repaint+composite`
- <http://js.jirengu.com/jidam/1/edit?html,css,js,output>

#### 3. 第三种，跳过 layout 和 paint

- 改变 `transform`，只需 `composite`
- 注意必须全屏查看效果，在 `iframe` 里看有问题
- <http://js.jirengu.com/wusew/1>

### 4.4.3 CSS 动画优化

#### JS 优化

1. 使用 `requestAnimationFrame` 代替 `setTimeout` 或 `setInterval`

#### JS 优化

1. 使用 `will-change` 或 `translate`

#### 参考文章

### 4.4.4 transition

位移 `translate` 缩放 `scale` 旋转 `rotate` 倾斜 `skew`

#### 经验

1. 一般都不需要配合 `transition` 过度
2. `inline` 元素不支持 `transform`，需要先变成 `block`

#### translate

##### 1. 常用写法

- `translateX(<length-percentage>)`
- `translateY(<length-percentage>)`
- `translate(<length-percentage>, <length-percentage>?)`
- `translateZ(<length>)` 且父容器 `perspective`
- `translate3d(x,y,z)`
- 演示<http://js.jirengu.com/xidiy/1/edit?html,css,output>

##### 2. 经验

- 看懂 MDN 语法示例
- `translate(-50%, -50%)` 可做绝对定位元素的居中

## scale

### 1. 常用写法

- `scaleX(<number>)`
- `scaleX(<number>)`
- `scaleX(<number>, <number>?)`
- 演示<http://js.jirengu.com/jucal/1/edit?html,css,output>

### 2. 经验

- 用的少

## rotate

### 1. 常用写法

- `rotate([<angle>|<zero>])`
- `rotateZ([<angle>|<zero>])`
- `rotateX([<angle>|<zero>])`
- `rotateY([<angle>|<zero>])`
- `rotate3d` 太复杂
- 演示<http://js.jirengu.com/jiquq/1/edit?html,css,output>

### 2. 经验

- 一般用于 360 度选择制作 loading
- 用到的时候查 rotate MDN 文档

## skew

### 1. 常用写法

- `skewX([<angle>|<zero>])`
- `skewY([<angle>|<zero>])`
- `skew([<angle>|<zero>],[<angle>|<zero>]?)`
- 演示<http://js.jirengu.com/tazer/1/edit?html,css,output>

### 2. 经验

- 用的较少
- 用到的时候查 skew MDN 文档

## transform 多重效果

### 1. 组合使用

- `transform:scale(0.5) translate(-100%, -100%);`
- `transform:none;` 取消所有

## 参考文章

#### 4.4.5 transition 过渡

作用是补充中间帧

#### 4.4.6 红心实践

css 需要想象力

<http://js.jirengu.com/nonud/1/edit?html,css,output>

## 5 HTTP

## 6 JS

## 7 算法与数据结构

## 8 JS 编程接口

## 9 项目

## 10 MVC

## 11 Webpack

## 12 虚拟 DOM 与 DOM diff

## 13 Vue

## 14 React

## 15 NodeJS

## 16 Vue3 造轮子