

《新媒体交互设计》

第一讲 课程介绍

近几年，交互设计（User Experience design）因其不菲的薪资成为大家转行的首选。

交互设计不仅在艺术领域有所发展，也在商业和科技领域中发挥着重要作用，交互设计专业拥有着广阔的就业前景和发展空间。

“学习交互设计毕业后可以做什么”也成为大家关心的热门话题。

什么是交互设计？

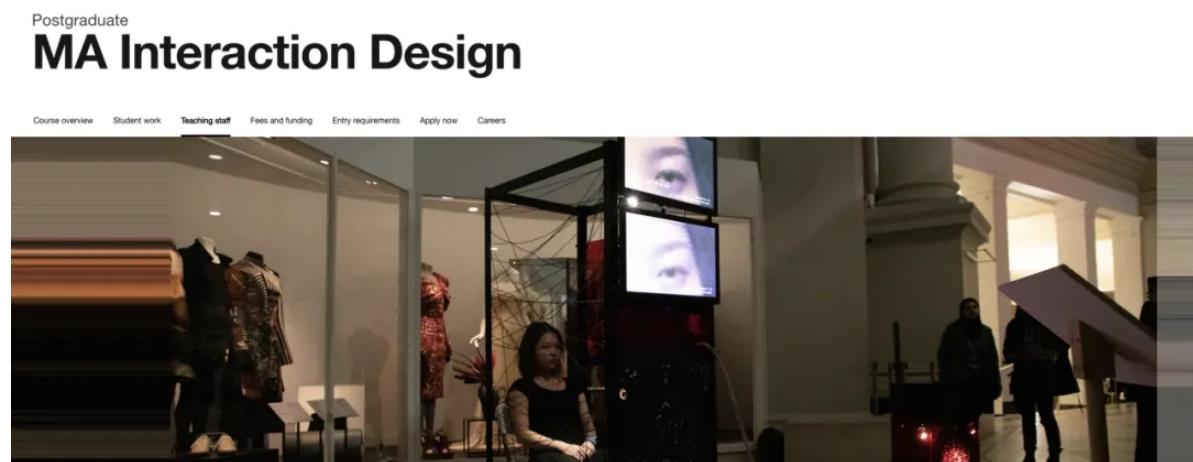
交互设计，是一种关注人与产品、系统、环境之间交互关系的设计领域。

这个专业的目标是创造出用户体验更好的产品和服务。

核心：解决问题

交互设计努力去创造和建立人与产品及服务之间有意义的关系，关注以人为本的用户需求，以“**在充满社会复杂性的物质世界中嵌入信息技术**”为中心。

交互系统设计的目标可以从“**可用性**”和“**用户体验**”两个层面上进行分析。



图片来源UAL (University of the Arts London, 伦敦艺术大学) 官网

交互设计可以分为三个层次：

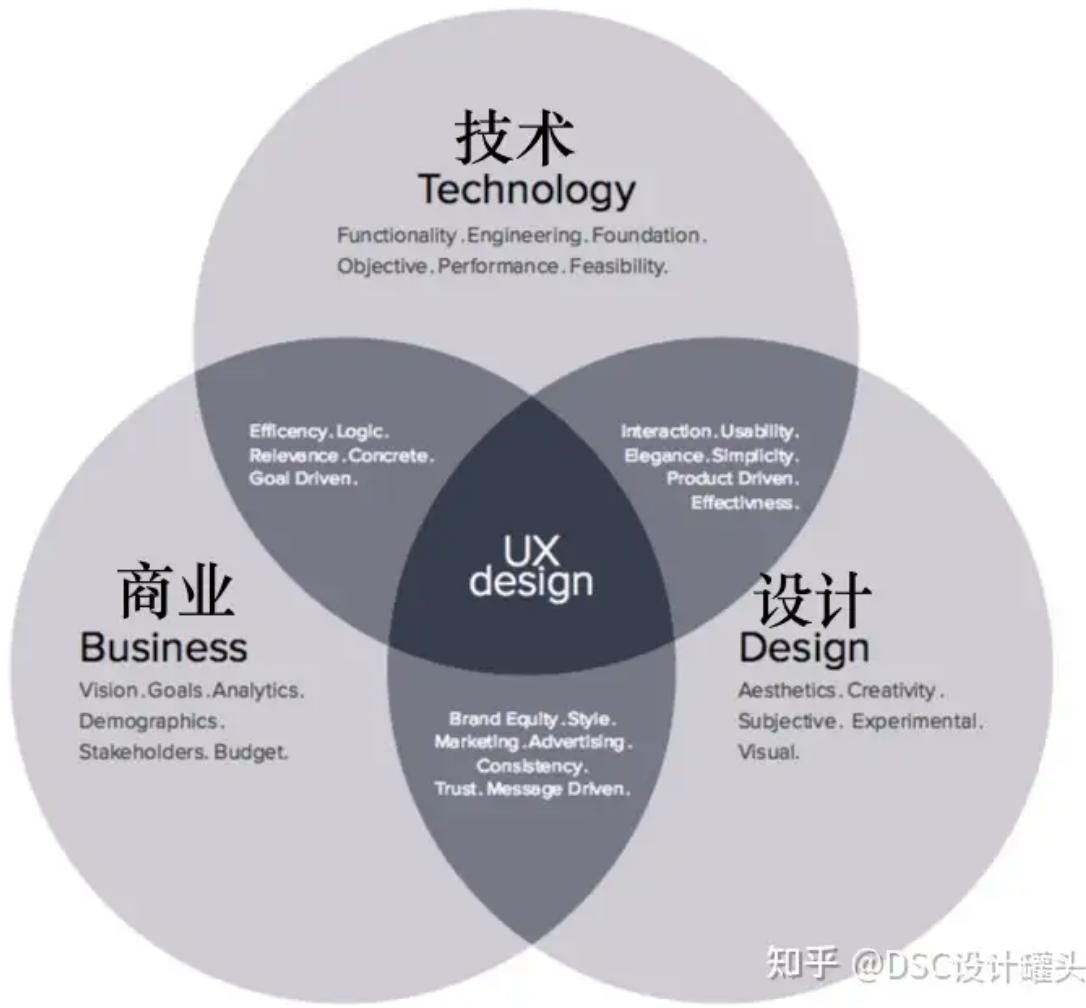
第一层是信息的传递，即把信息有序、合理、高效地传递给用户；

第二层是互动的设计，涉及到用户的选择操作及结果反馈等动态行为；

第三层是设计人与人的交互，通过一种服务、一种系统来创造人与人的交流。

与传统的设计学科相比，交互设计是一个融合了多个学科知识的新兴学科，是互联网时代、智能化时代的产物。

交互设计是一个跨学科的多元化项目，要求大家有较强的沟通能力、团队合作能力，强调团队分工。根据团队分工，交互设计可以大致分为技术、设计、商业三大方向。



© startupolic.com

+01. 技术路线

- Web App Developer 网页App设计

如果你对于 Web 网页或者APP移动端应用技术感兴趣，并且喜欢钻研 HTML、CSS 和 JavaScript等前端基础语言，那么你可以从事前端 Web 开发相关岗位。你还可以学习 Python、Ruby 和 PHP 等新语言，从而过渡到全栈开发。



前端设计师的技能点 © medium.com

- Video Game Designer 视频游戏设计

如果你愿意学习C++或C#相关语言，你也可以投身于视频游戏设计，从事游戏情节、界面设计，游戏程序编写、测试等相关工作。



知乎 @DSC设计罐头

Essential tools for game developers 游戏设计师需要哪些技能 © freelancermap.com

- Multimedia Artist and Animator 多媒体与动画

如果你还是想继续从事设计方向的工作，但是对技术又比较感兴趣，也可以从事多媒体、动画、[图形用户界面](#)GUI设计、计算机生成艺术设计等相关行业。





黑神话，悟空在游戏制作时使用猫咪来进行动态捕捉模拟情节中的四足怪兽的动作神态 ©
tw.news.yahoo.com

- Web analytics specialists 网站分析

网站分析师是当下热门的电商或者网络媒体运营必不可少的岗位之一。其日常工作除统计网站访问量以外，还要检索分析关键字、分析客户、分析客户行为、分析流量导入、分析广告、个性化服务等。从而对网站内容进行优化，使网站构架、内容、[网络营销](#)方面更具吸引、更出色，提升网站的功能。

WEB ANALYTICS



网站分析师的工作内容 © de.webmasters-europe.org

+02. 设计路线

- UI Designer 用户界面设计

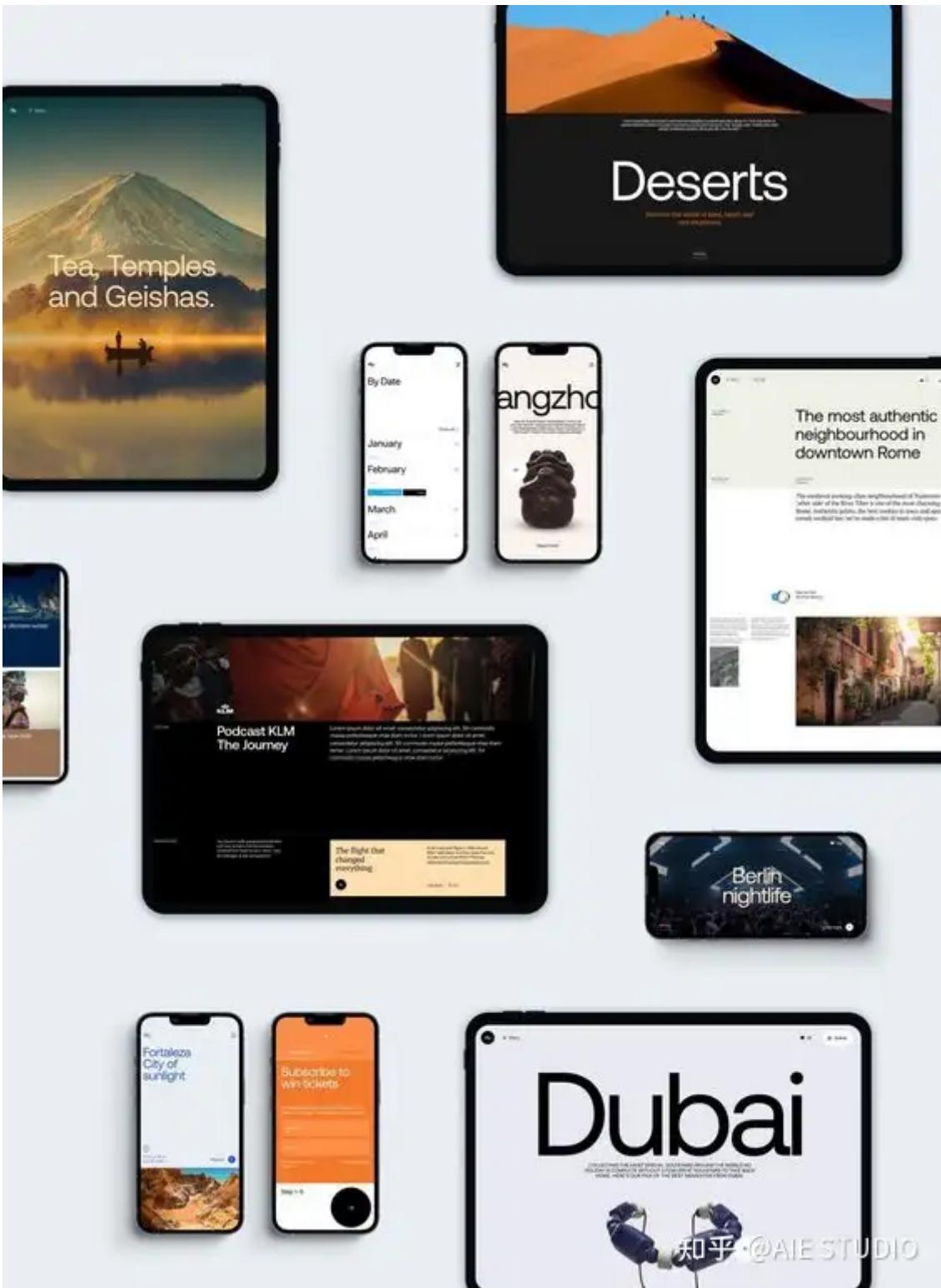
如果你有良好的审美和设计能力，可以从事用户界面设计，包括按钮、菜单、搜索栏和导航等的[视觉设计](#)和组合。

· 案例：

涉及到各种产品上的界面及产品本身的使用体验。



App



App



知乎 @AIE STUDIO

网站



appwatch



知乎 @AIE STUDIO

其他界面设计



知乎 @AIE STUDIO

其他界面设计



知乎 @AIE STUDIO

其他界面设计

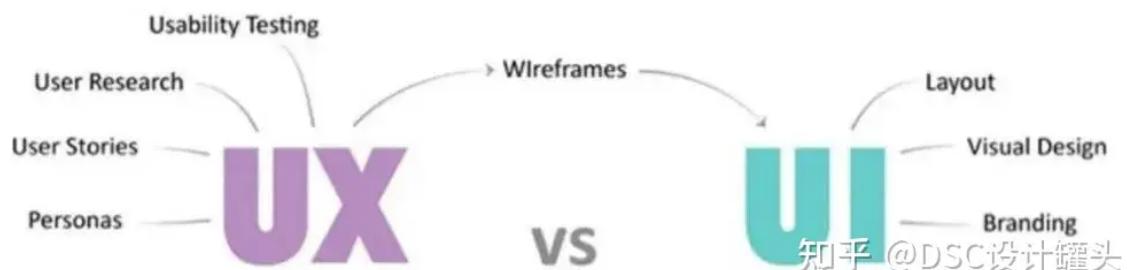


知乎 @AIE STUDIO

其他界面设计

- UX Designer [用户体验设计](#)

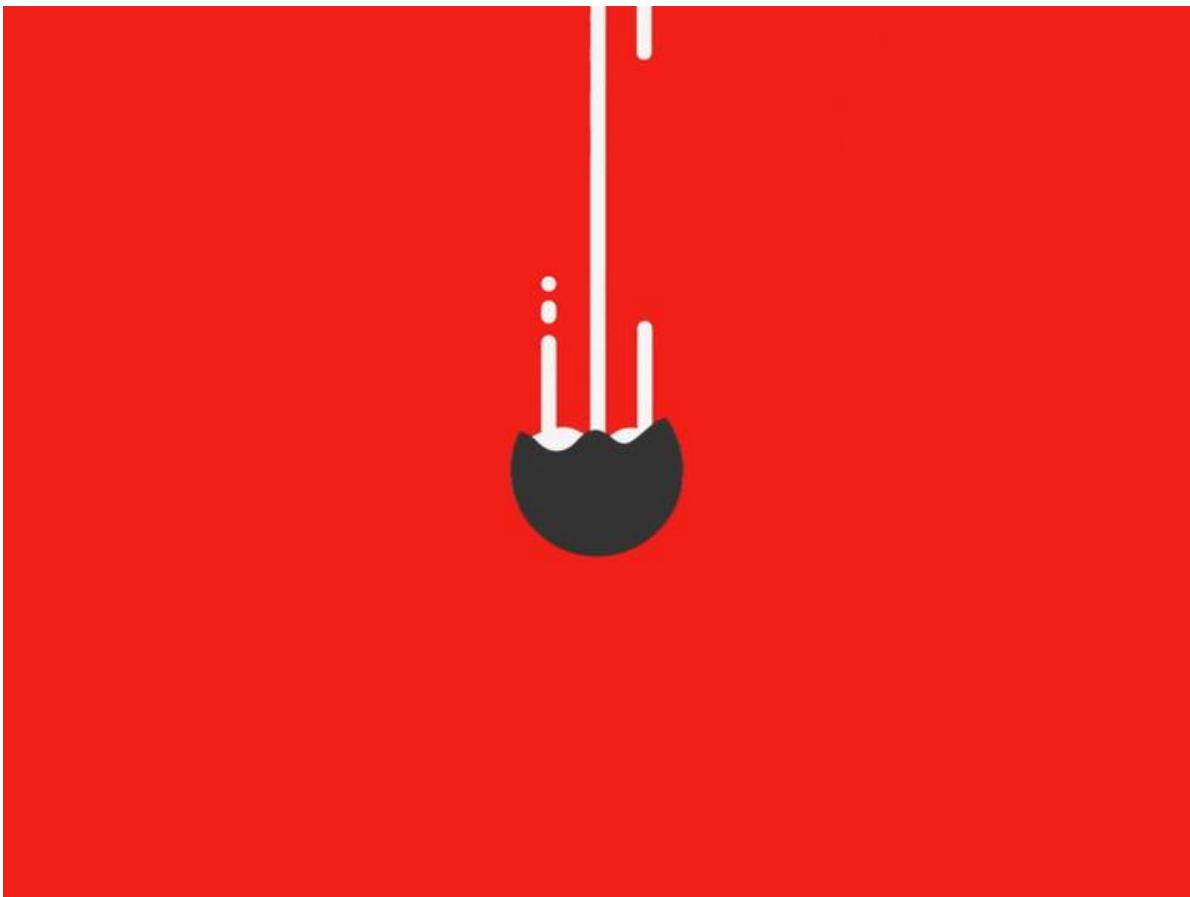
如果你逻辑能力够强，又善于了解客户心态和需求，可以从事用户体验设计，通过分析用户对产品的使用情绪，建立逻辑流程，创建产品原型、开发和测试等来提升用户体验。



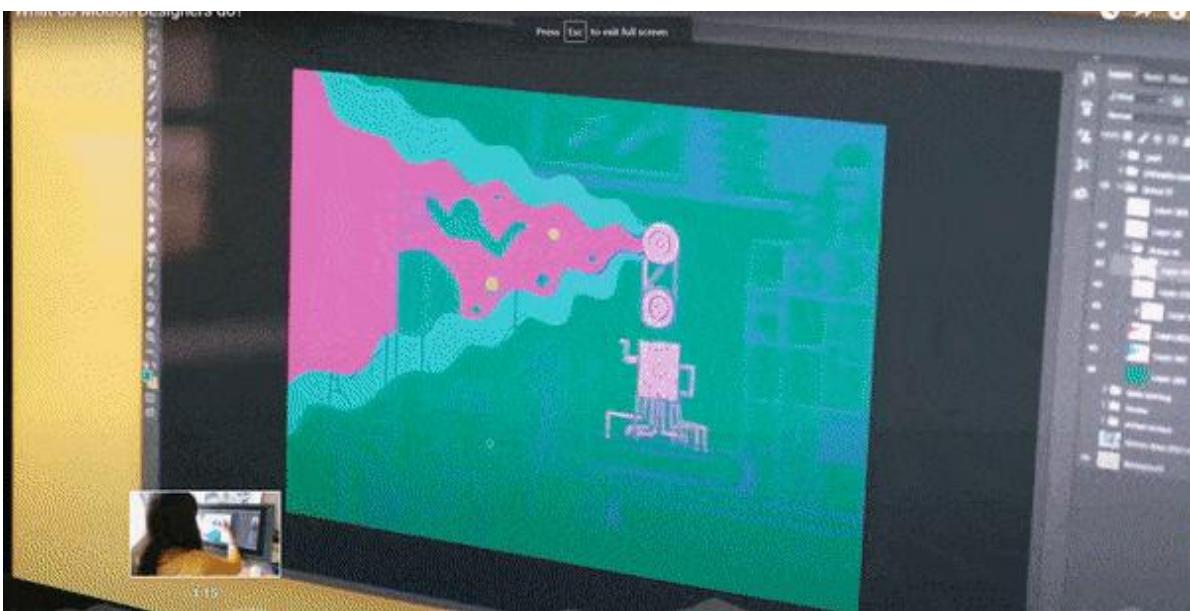
UI vs UX 设计师技能对比 © buzzp.net

- Motion Designer 动效设计

关注用户与屏幕交互时的动态体验。通过构建产品运动原型，创建动态图形、动画和移动元素来提升动画效果，同时需测试动画并确保流畅的性能。



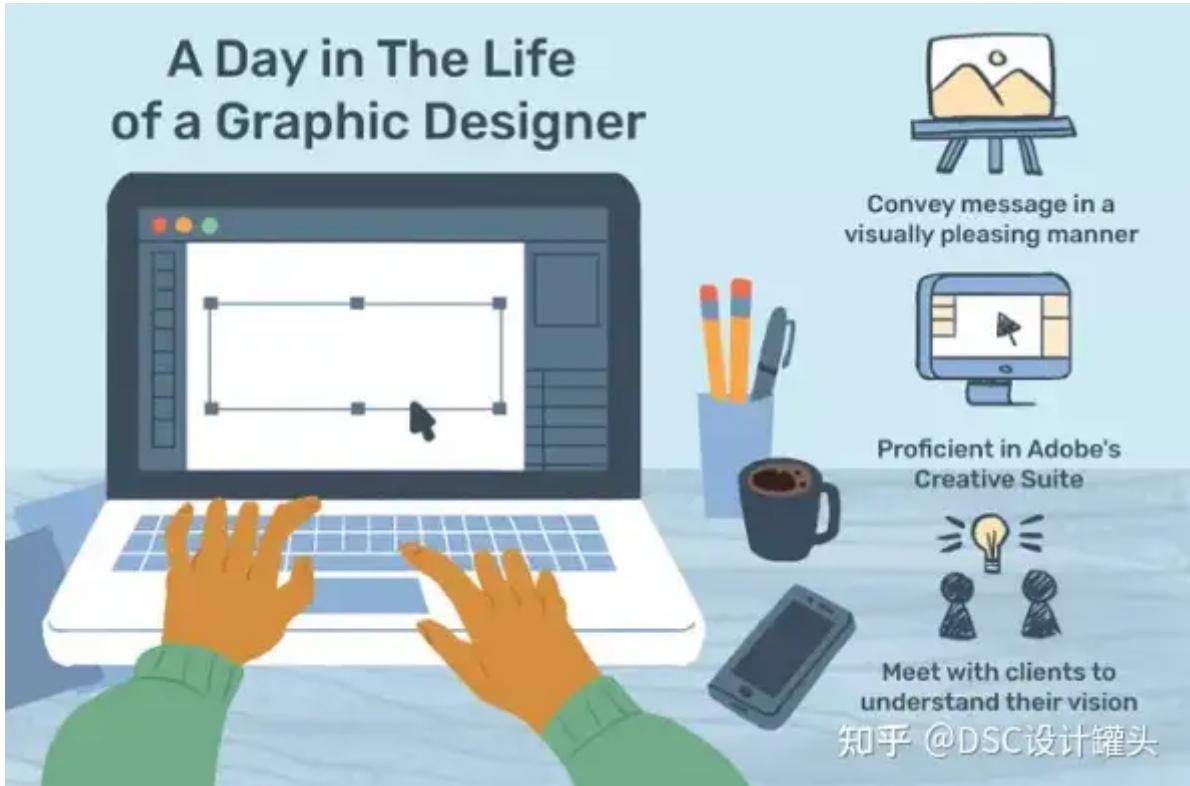
"Ideas are the fuel we run on." Logo动效设计 © dribbble.com



动效设计师的团队合作 © youtube.com

- Graphic Designer and Artist 图形设计

A Day in The Life of a Graphic Designer



图形设计师每天做什么 © thebalancecareers.com



图形设计师的工作流 ©youtube.com

+03. 商业管理路线

- Digital Marketing 数字营销

使用各种数字渠道来产生潜在客户并建立品牌知名度。数字营销在这里只是一个泛称，根据负责数字渠道的不同，可以细分成各种专业方向包括公司网站、Facebook、YouTube 和 Instagram 等社交媒体网络、搜索引擎、[电子邮件营销](#)、移动端应用、在线广告、博客论坛等。数字营销需要具备一定的分析及商业产品思维来制定高效的营销战略。



知乎 @DSC设计罐头

数字营销是做什么的 © pnetform.com

- Product Manager 产品经理

如果你擅长管理和领导团队，你可以尝试产品经理等相关岗位。你将担任更加面向业务的角色，包括项目规划、团队协调、资源分配等。这条道路需要你的领导才能和强大的人际交往能力。



知乎 @DSC设计罐头

产品经理需要做什么 ©productcoalition.com

- User Researcher 用户调研员

如果你对于交互设计的前期调研感兴趣，用户体验研员会比较适合你。该岗位需要充分了解用户的需求，通过彻底的研究包括前期调查、问卷调查、用户访谈、竞品分析、创建用户旅程、[可用性测试](#)等方法获得对客户心态的有用见解。并将调研结论呈现给设计团队，以帮助产品的外观和定位制定科学且有竞争力的策略。



知乎 @DSC设计罐头

用户研究院需要掌握哪些用户调研的方法 ©bootcamp.berkeley.edu

· 案例：

针对路怒情况，设计了一个系统，检测司机开车时的情绪，给出相应的驾驶建议，情绪越稳定车险越低

(我们的服务体系如何运作)



知乎@AIE STUDIO

交互设计及相关项目具有多学科交叉性，但它的多元背景包容性也会对人的沟通、团队合作等综合要求更高。交互设计更像是一个跳板，为你的职业生涯开辟更多可能性。有了新技术的武装，你既可以在设计领域继续深造而不可替代，又可以跳出设计的圈子去寻找新的兴趣点和职业发展方向！

同学们，试想一下，过去、现在和未来，交互设计给我们生活带来了哪些改变？

什么又是新媒体交互？

新媒体交互是在传统多媒体的基础上加入了交互功能，通过交互行为并以视觉、触觉、听觉等呈现信息，达到信息传递的目的。目前主要基于各种智能设备的窗口、菜单、图标、可视化图形界面与人实现交互。大家每天都在通过手机交互完成接受反馈各种信息：点击一个按钮、观看/暂停一个视频、用手机阅读一篇文章，用手指上下左右滑动/翻页，这些动作都是交互，而与图片、视频、音频等媒介交互统称为新媒体交互。

现在，我们确定了学习的方向，**新媒体交互设计——H5**

作业一：整理思路，用MarkDown写一篇关于我的新媒体交互设计作品的构想/构思（技术方向不限）？

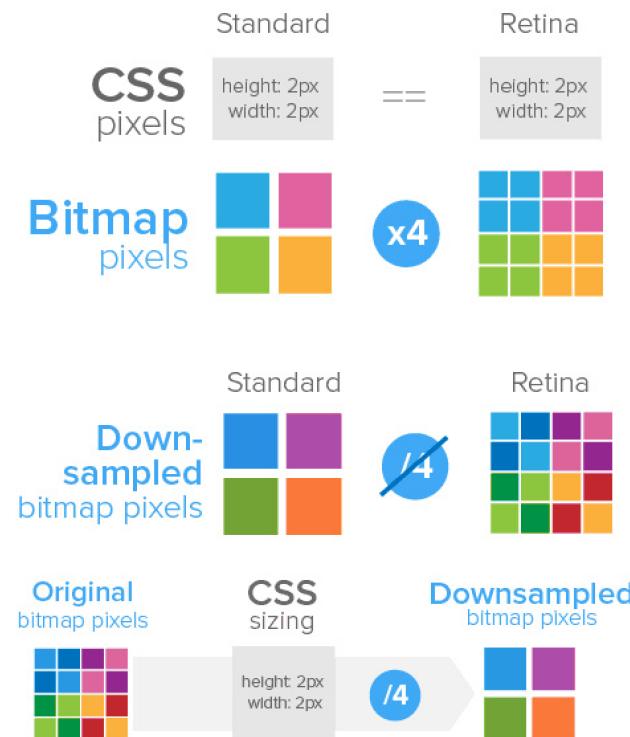
1.为什么学？

互维网使得全世界的人们以史无前例的巨大规模相互交流。相距遥远的人们，甚至是不同年代的人们可以通过网络来发展亲密的关系或者使彼此思想境界得到升华，甚至改变他们对待小事的态度以及精神。情感经历、政治观点、文化习惯、表达方式、商业建议、艺术、摄影、文学都可以以人类历史上从来没有过的低投入实现数据共享。

数据渲染—>网页（输出）—>Screen

- 我们从生活中无处不在的屏幕（Screen）中去了解世界！

- PC屏幕
- Mobile屏幕
- 自定义屏幕



Web¹ 标准大潮已经席卷了国内互联网的各个领域。

Web3.0² 是指第三代互联网，也称为智能互联网。一个运行在“区块链”技术之上的“去中心化”的互联网。



Web标准简介³

Web发展史年鉴⁴

2. 学什么？

本课程是**网络新媒体专业**的选修课，是一门偏实践的计算机课程。

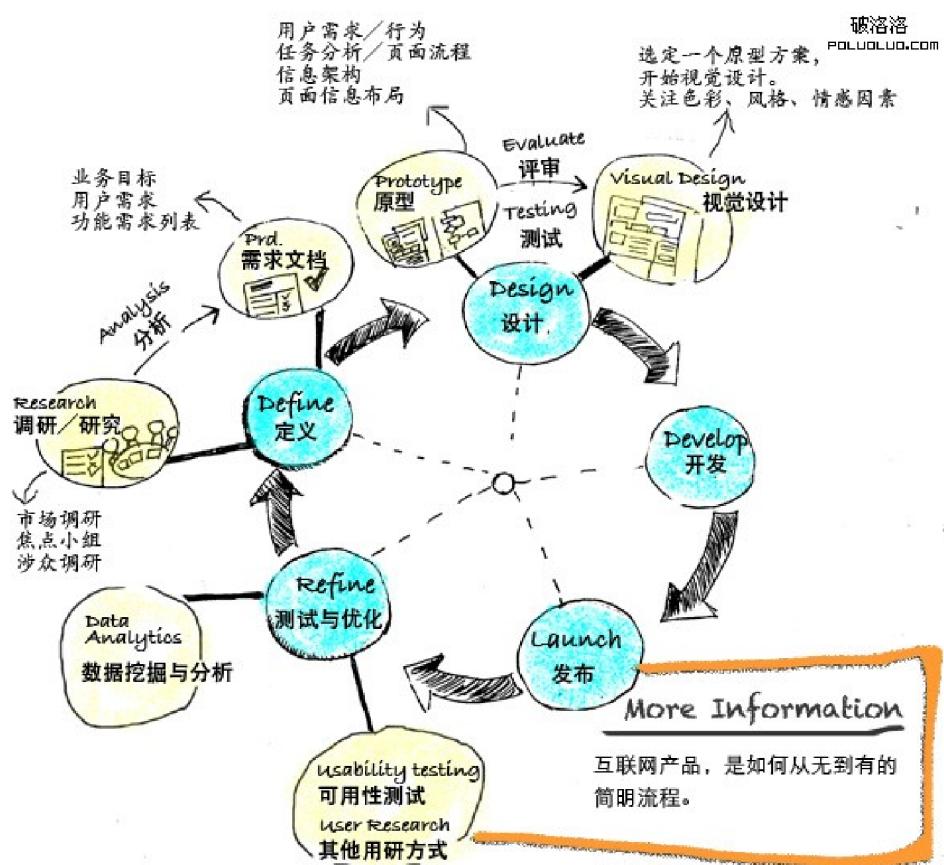
- 力求以简单高效地方法生产出与新闻内容的交互式网页页面，为后续的课程提供可视化展示的平台。

- 网络编辑
- 新媒体数据分析
- 数据新闻等

- 重点培养学生的动手能力，激发学生的学习兴趣，具备举一反三、自主创新的能力。

- 案例展示

- H5交互页面
- 响应式首页
- 引导页
- Echarts数据大屏页面
- Map可视化页面
- Slidev PPT展示页



• 设计方法

- 从无到有的网页生产流程（三步走）
 - 抽象（原型图、草稿）
 - 映射（效果图、设计稿）
 - 实现（布局）
- 网页的构成

- 内容
- 结构
- 表现 (渲染)

- 角色

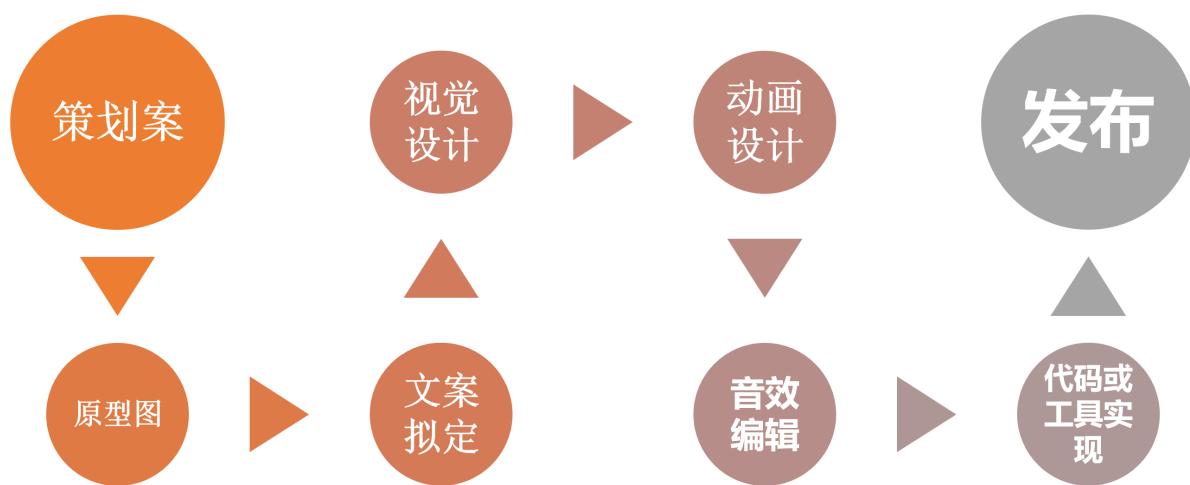
- 网络编辑
- 设计师
- 美工
- 程序员

- 技术语言

- Html
- CSS
- JavaScript
- Markdown等

- 工具

- Dreamweaver
- Photoshop
- Typora
- 木疙瘩
- Nginx(IIS)等



3.如何学？(从无到有)

登录名 密码 选择去向 注册通行证 找回密码 客服帮助 企业邮箱 手机新浪网 设为主页 新浪导航

sina新浪网 sina.com.cn
成都 多云 29~21℃

新闻 军事 社会 体育 中超 NBA 博客 微博 草根 读书 教育 健康 空间 邮箱 出国 城市 四川 上海 高尔夫 下载 导航
财经 股票 基金 娱乐 明星 音乐 视频 插客 大片 女性 星座 育儿 论坛 SHOW UC 生活 旅游 电商 商城 天气 爱问
科技 手机 数码 汽车 图库 车型 房产 地产 家居 乐库 尚品 宠物 游戏 玩玩 交友 短信 彩铃 彩信 彩票 公益 世博

热销
提价或引发严厉调控
楼市面临阶段性回暖
四川筹建天府新区
东山副总微博赞乐居
盗梦空间趴生活奢想
二线城市房价稳升
80后小窝涂出彩(图)
6-12日楼市开盘汇总

滨水御院 潘白独栋 财富公馆 北京三亞 格林公馆 应市公映 PARK北京 天然豪宅 380万水岸城市别墅
中国景观魔王 大国世家·长安太和 国奥双轨 55-80m² 燕莎官邸 阳量优惠 榻榻房精装新品
新浪微博

中国航母横空出世
房地产大腕齐道贺
百度强势开拓新市场
二手房搜索全新体验
CRIC开创新局面
安家首选楼盘和小区
乐山土地继续疯狂
11日无优惠不发团
拿11日团购优惠报名

品牌 房买 卖房 租房
百度乐居二手房屋隆重上线
新浪二手房 百度乐居售房

爱问搜索 网页 新闻 音乐 图片 地图 知识人 博客 资料 汽车 楼盘 铃声 彩铃
请输入关键词 搜索

热门品牌专区 ▶ 惠活动
云南白药 男士动能 香翼 忧抗力
京东商城 搜索

教育 培训 招生 出国
留美申请时间及专业大全
上海交大EMBA
人大商学院MBA研修课程
港理大品质管理硕士热招
北大光华总裁班正在招生
北大创办千人私募班

视频 博客 播客 大片 电视台 新闻 上传视频
M6
尊贵·安稳
旗舰尊享价：23.98万
BYD 比亚迪汽车 高品质好车

新闻 四川新闻 北京时间:2010.9.8
· 中央政策助推经济特区30年发展 胡锦涛讲话反响强烈
· 我国使馆要求日方立即放人放船 派员探视被扣渔民
外交部召见日本大使提出强烈抗议 ▶ 视频 专题 滚动
· 湖南回应茶油秘密召回质疑 评论：应启动问责 专题
· 教育部网站刊登倡议书倡导教师不收礼

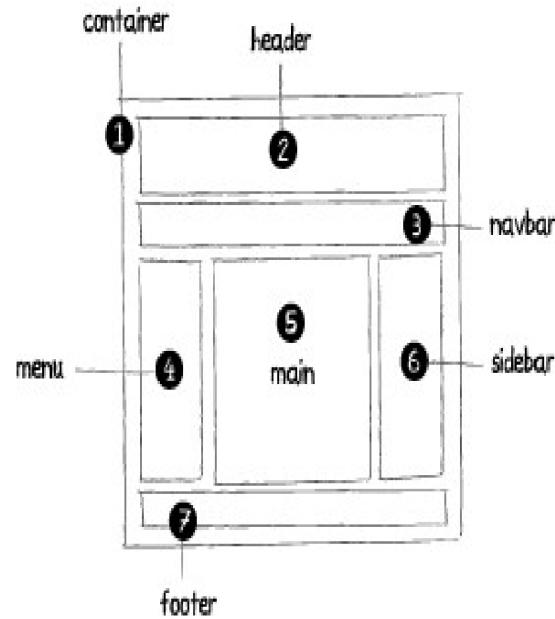
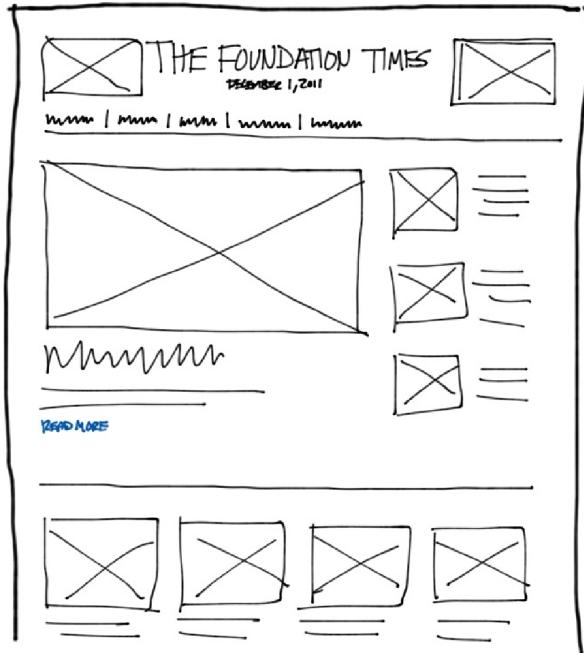




图 1.14 网站效果图

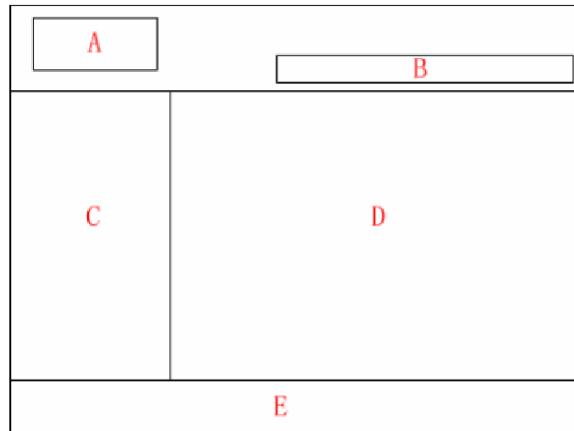
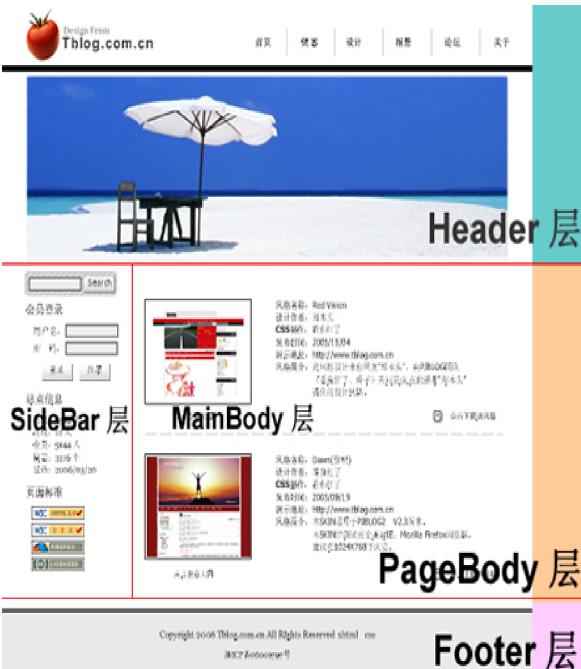


图 1.15 布局



- 网站设计及基本框架结构（布局命名）

Container

- “container”就是将页面中的所有元素包在一起的部分，这部分还可以命名为：“wrapper”, “wrap”, “page”

Header

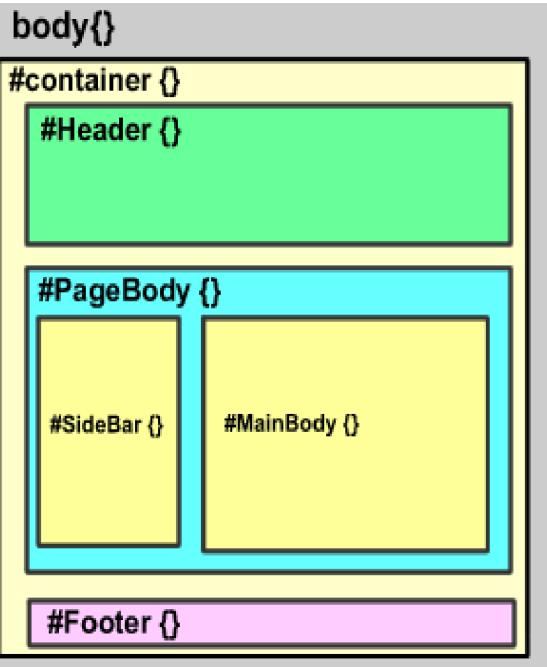
- “header”是网站页面的头部区域，一般来讲，它包含网站的logo和一些其他元素。这部分还可以命名为：“page-header”（或 pageHeader）。

Logo

- 通常网站为体现其特色与内涵，涉及并制作一个LOGO图像放置在网站的左上角或其他醒目的位置。企业网站常常使用企业的标志或者注册商标。

Banner

- “banner”是横幅，Banner的内容通常为网页中的广告。在网页布局中，大部分网页将Banner放置在与导航条相邻处，或者其他醒目的位置以吸引浏览者浏览。



Navbar

- “navbar”等同于横向的导航栏，是最典型的网页元素。这部分还可以命名为：“nav”, “navigation”, “nav-wrapper”。

Menu

- “Menu”区域包含一般的链接和菜单，这部分还可以命名为：“subNav”, “links”, “sidebar-main”。

Main

- “Main”是网站的主要区域，如果是博客的话它将包含的日志。这部分还可以命名为：“content”, “main-content”(或“mainContent”)。

Sidebar

- “Sidebar”部分可以包含网站的次要内容，比如最近更新内容列表、关于网站的介绍或广告元素等...这部分还可以命名为：“subNav”, “side-panel”, “secondary-content”。

Footer

- “Footer”包含网站的一些附加信息，这部分还可以命名为：“copyright”。 © 5

- 学习资源

- [W3Cschool](#)

- [菜鸟教程](#)

- [计算机的编年史](#)

4. 我的第一个H5 “Hello Wrold!”

- 我的一个人网页(index.html)

- 内容 (文字)

- 结构 (Html)

- 表现 (CSS无)

- Html源代码：

```
<html>
  <head>
    <title>第一个网页</title>
  </head>
  <body>这是我的第一个网页</body>
</html>
```

- 三种Html编辑器

- Notepad

- W3C在线

- DW/VScode

- 思考一下？：我们是如何浏览网页的？

- 例子：背包客旅行札记

- 内容 (文案)

- [结构 \(Html\)](#)

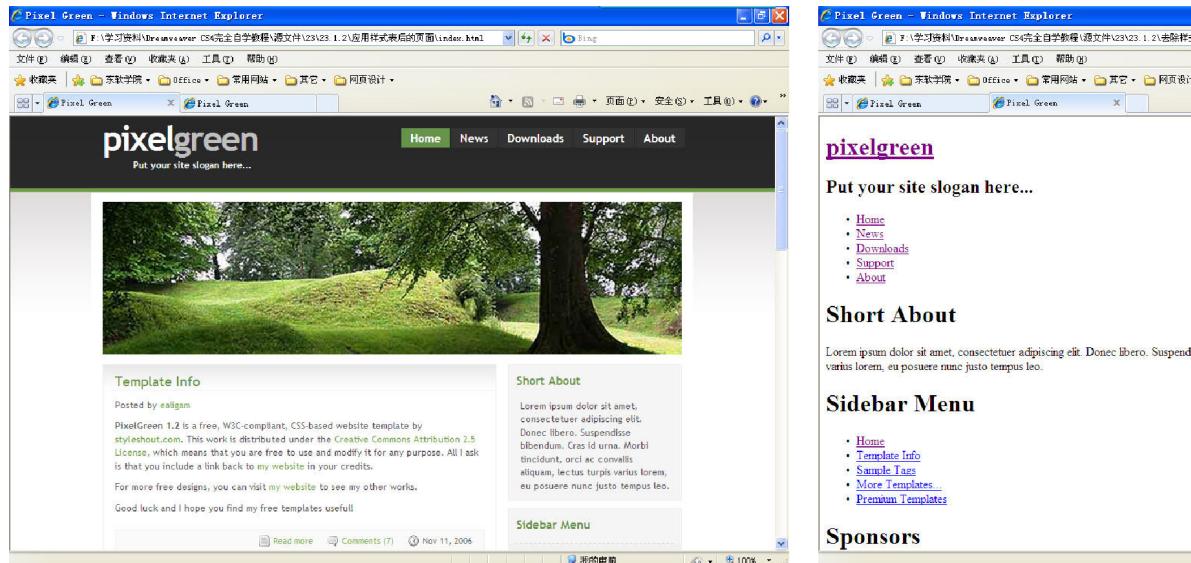
表现 (CSS)

- 从无到有网页生产流程：

- 第一步：头脑风暴（笔纸绘制草稿）
- 第二步：设计稿（PS创作）
- 第三步：观察、拆分设计稿（PS切图功能）
 - 分离颜色：基本配色、超链接配色和导航链接配色
 - 提取尺寸：网页分辨率是72ppi（像素/英寸），画布尺寸1920px*1080px，内容显示的区域为996px
 - 分离背景图：CSS Sprites（精灵图）
 - 分离图标及特殊边框
 - 分离图片
- 第四步：网页设计实现
 - Table布局
 - Div+CSS布局
 - HTML

- CSS禅意花园

CSS 禅意花园: CSS 设计之美 (csszengarden.com)



- 网页的基本元素：

- 文本
- 表格
- 图像
- 超链接
- 动画
- 表单交互
- 音/视频等

5.如何发布自己的网页?

- 从网页到网站 (整体性原则)
 - 主页或首页: Index Page
 - 登录页: Login Page
 - 着陆页 (引导页) : Landing Page
 - 导航链接: Navigation Page
- Nginx⁶ (IIS)
 - 下载
 - 安装
 - 配置
 - 发布
 - 测试

课后实践1

提交时间: 第3周前。

- 从无到有一—制作自己的["个人主页"](#)
 - 抽象 (大脑)
 - 映射 (PS的效果图)
 - 实现 (布局、代码)
- 认识浏览器 (Browser)

★ ★ ★ 作业提交规范★ ★ ★

1. 目录命名:
 - 每人每次作业一个独立的文件目录。
 - 作业XX+学号+姓名, 注意顺序、中间无空格无+号, 例如: 作业一20211060001张三。
2. 网页命名:
 - 不可以用中文或其他英文命名。
 - 首页: index.html
 - 一级页面: index01.html
 - 二级页面: index0101.html
3. 图片目录: /image
4. CSS文件命名: layout.css
5. 注释:
 - html:
 - CSS: //和/* 注释 */

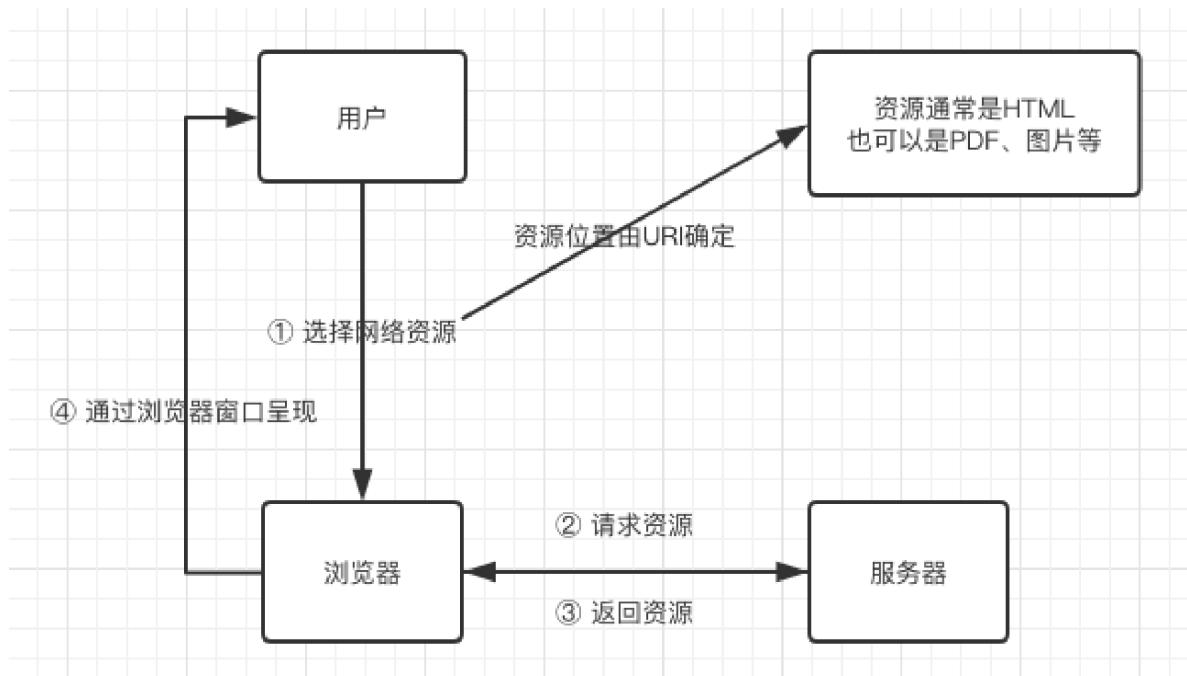
第二讲 认识浏览器

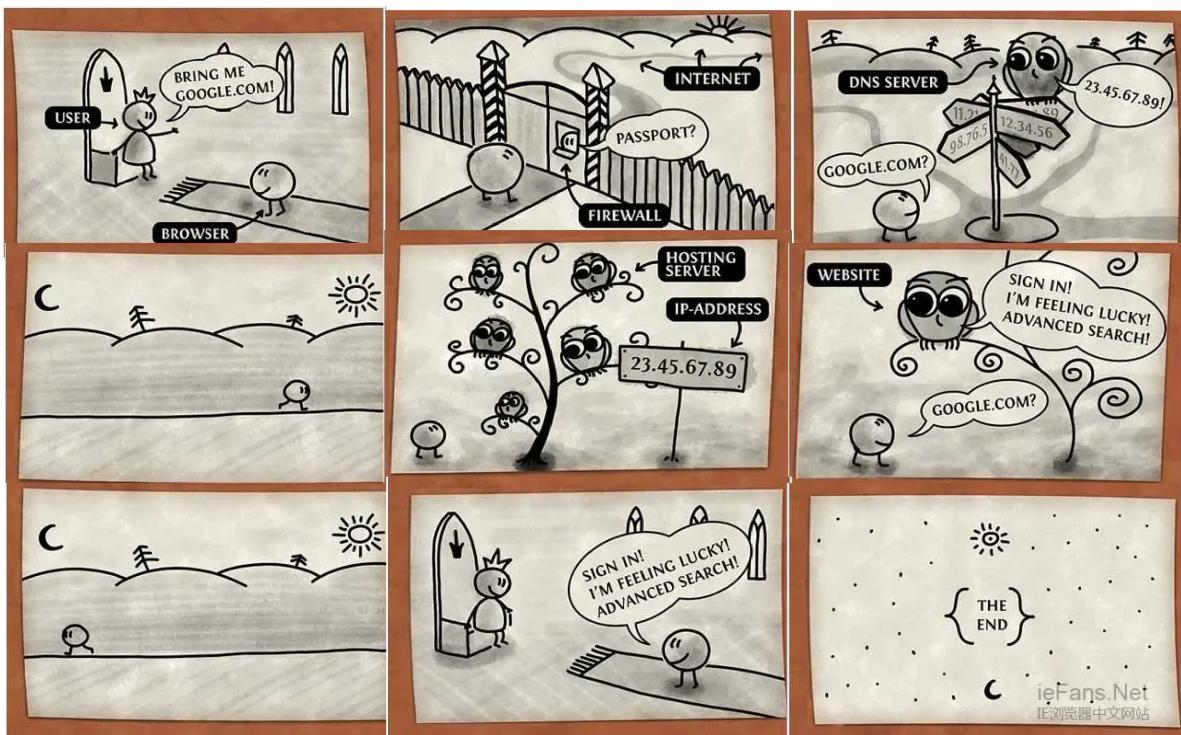
今天，有五种主流浏览器——IE、Firefox、Safari、Chrome及Opera。



© @老虎的新世界

浏览器的主要功能是将用户选择的web资源呈现出来，它需要从服务器请求资源，并将其显示在浏览器窗口中，资源的格式通常是HTML，也包括PDF、image及其他格式。用户用URI (Uniform Resource Identifier统一资源标识符) 来指定所请求资源的位置。





1. 浏览器的前世今生⁷

1.1 Web时代的序幕

1990年，英国计算机工程师蒂姆·伯纳斯·李（Tim Berners-Lee）开发出首个Web服务器与图形化Web浏览器“World Wide Web”（即“万维网”）。软件把Web互联网数据呈现在我们的眼前。互联网世界就此打开。

到了1993年，网络发展突飞猛进。大学、政府机关、企业都看到了开放互联网中的机会，每个人都需要新的计算机程序来访问网络。这一年，美国国家超级计算机应用中心（NCSA）的计算机科学家马克·安德森（Marc Andreessen）开发了Mosaic，这是第一款流行的Web浏览器。

NCSA Mosaic易于使用，又可以在Windows计算机运行，让任何拥有PC的人都可以浏览早期的网页、聊天室、图库。Mosaic是Mozilla Firefox的早期版本，同时也是网景浏览器和微软IE浏览器的起源。

1.2 第一次浏览器大战

网景兴起



1994年，安德森创立Netscape并公开发行了Netscape 0.9，网络浏览器从试验室走向用户，取得了巨大成功。这也是第一次浏览器大战的开端。

微软应战：IE浏览器



@老虎的新世界

微软直接购买了网景导航者浏览器原型版本Mosaic的源代码（也就是马克·安德森在大学时期做的第一个版本的Mosaic浏览器）。然后，稍作修改后就发布了。从IE3.0开始，微软把IE浏览器强行内置在了Windows的操作系统中。使用IE浏览器，你等于免费不掏钱。同时，微软也在商业市场发力，联合大型互联网服务提供商，美国在线、AT&T等，让他们放弃网景导航者浏览器而改用IE作为首选浏览器。

网景失误，IE获胜



1997年，微软IE更新至4.0版本后，性能已经与Navigator不相上下。四年内，IE获得了75%的市场份额，到1999年甚至达到了99%。

微软上下其手，明争暗截，各种黑手阴招频出。同时，网景自身也犯了一系列错误，终于彻底被微软挤死了，最终在1998年被美国在线（AOL）以42亿美元收购。第一次浏览器大战以IE后来居上，网景的惨淡收场告终。

1.3 第二次浏览器大战

网景失败，IE躺平



2003年，IE的市场份额超过了八成，处于绝对垄断地位。

在IE6之后，IE团队事实上就解散了。因为那时候的IE，打到了竞争对手网景，微软可以歇歇气了，滋生了懒惰情绪。IE5之前，IE每年进行一次大版本更新，而IE5到IE6用了两年的时间。更过分的是，从IE6到IE7居然过了5年的时间。

IE市场占有率非常高，从商业上来不值得再继续投入。IE属于捆绑消费，不能直接卖钱。继续投资IE没有直接对应的效益可言。所以我们不能刻意拔高那些所牛皮的公司，没有钱，也无所谓所愿景和使命感。

就像黄石公园的群鹿一样，没有了狼，鹿群就没有什么奋斗精神了。微软在浏览器方面，滋生了懒惰躺平精神，在IE6.0版本之后长达五年的时间里，微软对IE没有任何作为。

然而市场此时在悄悄孕育着强大的竞争对手。终究有一天，微软自食其果，IE全面落败，被迫退出历史长河。

网景涅槃重生，firefox虎口夺食



前面说过，一个强大的网景公司被微软明里暗里搞死了。它肯定是心有不甘的，临死之前，网景垂死挣扎了一下：**把网景导航者浏览器的核心源代码开源了！**

开源，也就意味着，浏览器的源代码，你有我有大家都有了。谁都可以依据它来改进和开发。

1998年，一帮网景前老员工们一起成立了一个组织，其中就有Netscape创始人之一Marc Andreessen 和 Netscape 高级工程师 Eric Bina。他们合谋开发下一代的浏览器，这个组织就是 Mozilla.org，叫做 Mozilla。

前面说过，网景浏览器的代码存在诸多问题，技术架构陈旧，比如代码复杂、不稳定、安全性较差，难以满足当时互联网快速发展的需求。这些问题也让 Mozilla 组织决定采用新的代码架构和技术，重新设计和开发一个更加安全、稳定、快速和功能丰富的浏览器。

因此，Mozilla 组织开发了 Firefox 浏览器，其采用了全新的 Gecko 渲染引擎和 XUL 用户界面语言，使其在性能、安全性和可扩展性方面都有了显著的提升。网景浏览器的源代码几乎被全部抛弃。

2004 年 11 月 9 日发布了 Firefox 的第一个正式版本 Firefox1.0，即火狐浏览器。在随后的 4 年时间里，尽管 IE 的份额依然领先，Firefox 也稳扎稳打，凭借几次大版本更新提升了浏览器的功能与先进性，又有 Google 的神助攻，它从 IE 手中夺下了两成多的市场份额。Firefox 迅速成长，成为了当时市场份额排名第二的浏览器。

1.4 第三次浏览器大战

chrome 浏览器崛起



2006 年，搜索引擎已经成熟，苹果在研发手机，谷歌安卓正在开发。各种趋势表明，**互联网世界处于 PC 互联网向手机移动互联网的变革趋势中。这一点微软还在睡梦里面，毫不察觉。**

[问题1]

Chrome 与 Chromium 浏览器的区别在哪里？



2006 年，谷歌邀请拉斯·巴克 (Lars Bak) 领导 JavaScript 引擎开发。

我们都知道，浏览器核心代码是渲染引擎和 JS 引擎。搭载 V8 引擎的 chrome 浏览器，据说当时 JavaScript 解析速度是 IE 的 65 倍之多，一经问世，顿时让 IE 成为了下载 chrome 浏览器的入口浏览器。

微软如梦方醒，一朝醒来，发现处于全面劣势。因为长江后浪推前浪，不仅 chrome 技术大幅领先，玩起谋略，chrome 浏览器的称霸过程，比微软当年的策略更胜一筹。

- 首先，谷歌把Chromium内核开源，谁都可以基于这个内核开发自己的浏览器，且不用公布自己做了哪些改动。这个策略惠及了许多开发者，比如国内的比如QQ、360、UC大部分浏览器，都是基于Chromium内核开发出来的。
- 其次，谷歌拉拢扶植火狐的公司来打压IE，让fireFox浏览器把google成为了自己默认的搜索引擎。Mozilla有90%的收入都来自谷歌，当然愿意干了。这妙计有三国“联吴抗曹”之功效。
- **Chrome与Google账号捆绑**，所有Android系统的手机也都内置了Chrome。这个是最重要的致命手段，善玩捆绑的微软，终于遇到了一个更会玩捆绑的。

IE浏览器倒下



互联网技术是飞速发展的，没有了专业团队和继续投资，IE开始爆出各种问题。

首先就是安全问题，IE不更新，各种安全漏洞越积越多。最后连微软自家的安全专家都建议大家不要把IE作为默认浏览器。

其次是IE的代码架构和技术也落后了，像是上个世纪的老爷车一样越跑越慢，还没法和最新的技术兼容，启动慢、易卡死之类的问题也就成了家常便饭。

逐渐地，用户越来越厌恶IE。2006年，IE6被国外机构评为“有史以来第八糟糕的科技产品”，2011年，IE6再次进入“科技史上50种最糟糕科技产品”的榜单第11名。**短短几年，在firefox和chrome的映衬之下，IE就彻底变成了人厌狗嫌的产品，唯一的作用，就是提供各种网络笑话。**

IE也不是没有挣扎过，后续推出多个升级版本，但是都失败了。2022年6月15日，是个值得互联网纪念的日子。这一天，全世界互联网用户的噩梦——IE浏览器，正式宣告退出历史舞台。

为了纪念这个服役27年的浏览器正式死亡，一位韩国工程师甚至花了两千多元给它立了一块墓碑，上面写着的墓志铭是：“**他是下载其他浏览器的好工具**”。



Edge浏览器兴起



✿@老虎的新世界

2015年4月，微软新开发的Edge浏览器，随着Win10登场了。最初的Edge是在EdgeHTML渲染引擎上构建的，性能不高，且由于开发环境薄弱和各种兼容性问题显得很鸡肋。

2018年，微软下定决心放弃自家引擎，转而采用更成熟、开源的Chromium内核。事实上代表微软服软，不再封闭。2020年1月15日，微软发布了基于Chromium的新版Microsoft Edge。采用Chromium内核之后，Edge浏览器将直接兼容支持Chrome现有的所有扩展。

这一改变有了立竿见影的效果，微软有着Windows平台的天然优势，Edge在今年4月的桌面端市场份额开始快速提高。借助自家windows平台的优势，微软稳住了浏览器市场，份额开始逐步上升。

firefox份额下降



✿@老虎的新世界

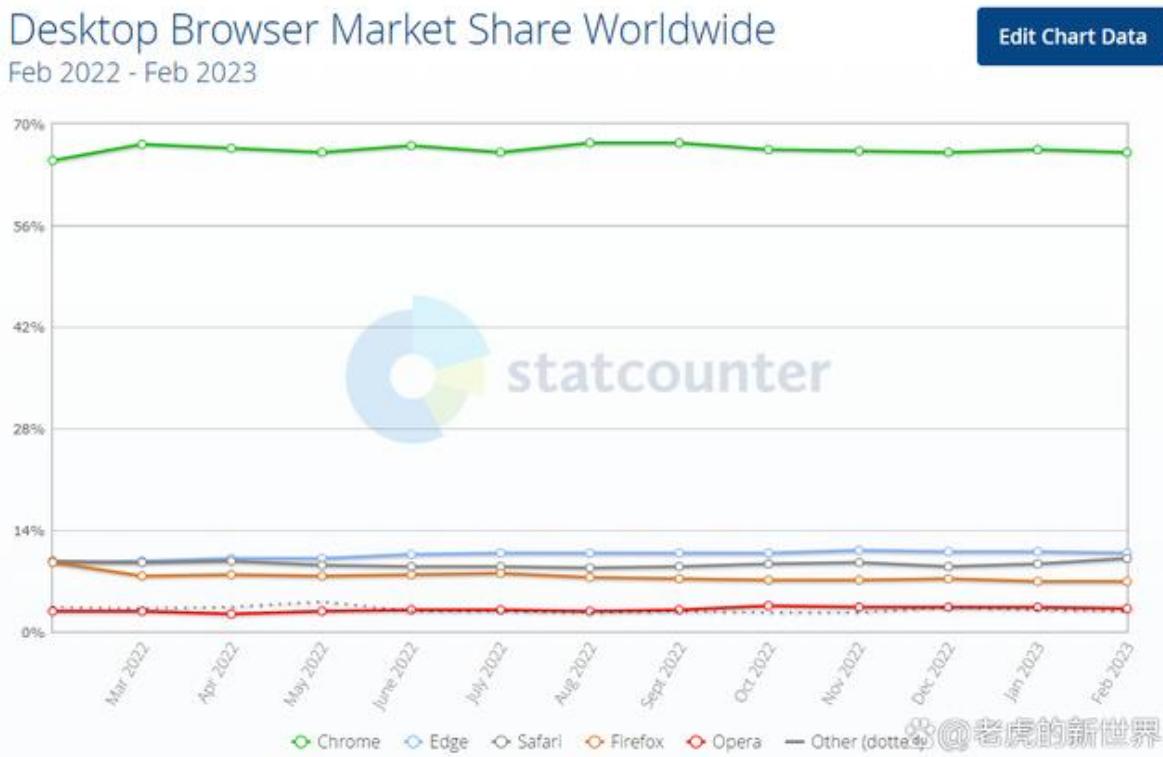
前面说过，当初google的chrome浏览器没出来之前，让fireFox浏览器把google成为了自己默认的搜索引擎。

差不多十年中，火狐浏览器一直是谷歌的忠实盟友。依靠给谷歌送去搜索流量，Mozilla每年获得近三亿美元的收入，而谷歌依靠火狐浏览器的鼎力协助，后来一度斩获了全球搜索市场的九成。

既然google有了自己的chrome浏览器，那么这样就不太划算了。于是firefox与百度也有了一定的合作，将在全部中文版火狐互联网浏览器中内置百度搜索引擎。

chrome和edge浏览器的竞争，让firefox的份额开始下降。但firefox作为开源浏览器，广受开源社区的好评，在专业人士、数码爱好者和极客中用户很多，绝不会再走网景浏览器的灭亡的老路了。

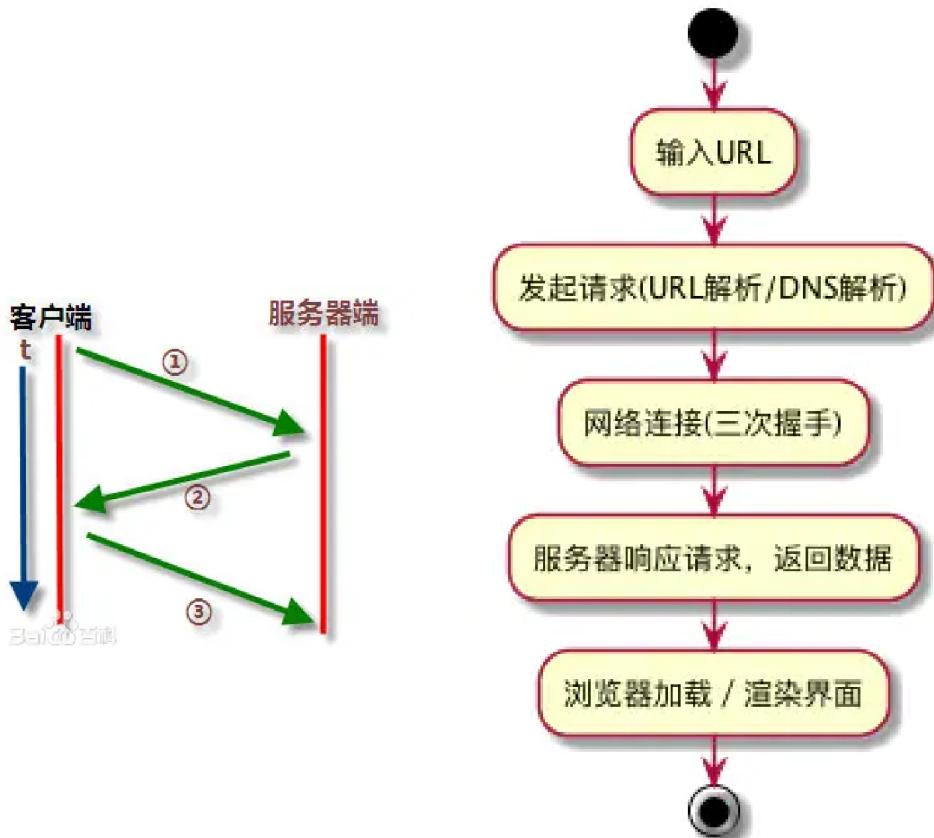
1.5 浏览器现状



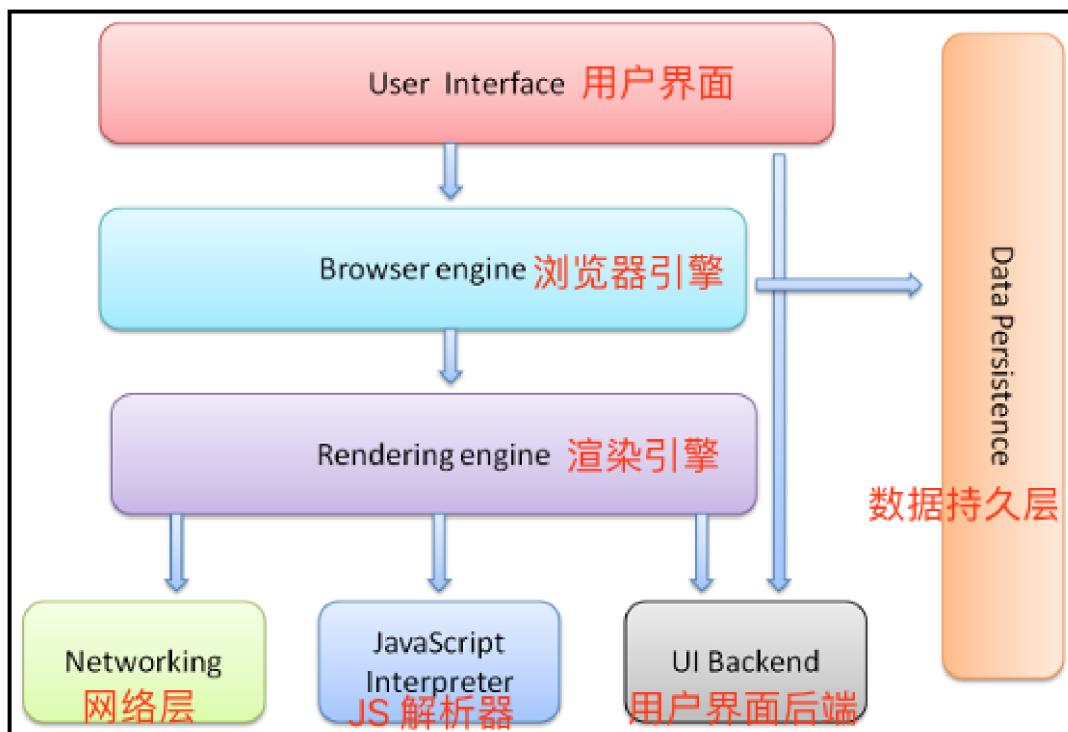
Statcounter 近日发布了 2022 年 2 月到 2023 年 2 月的浏览器份额报告，Chrome 占 66% 左右，Edge 占 10% 左右，fireFox 接近 7%，safari 占 10%。

2. 页面展示 timeline

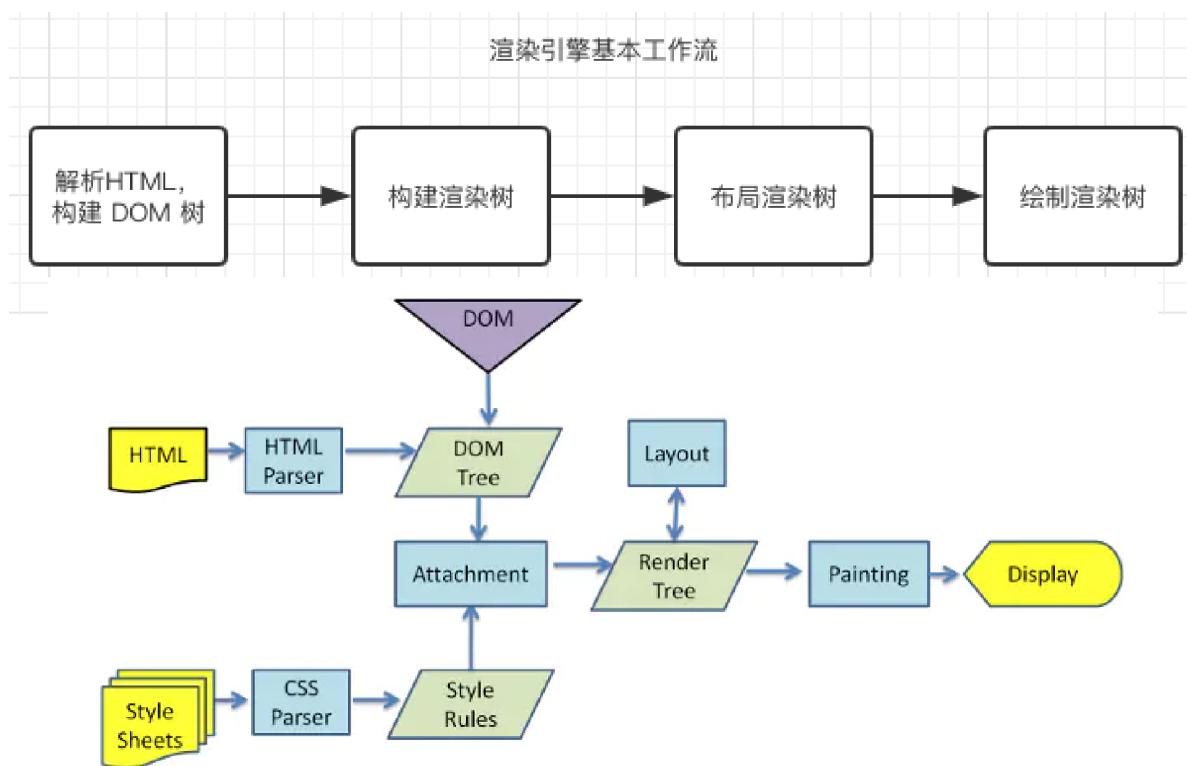
浏览器加载和渲染界面这个过程。



3. 浏览器的结构



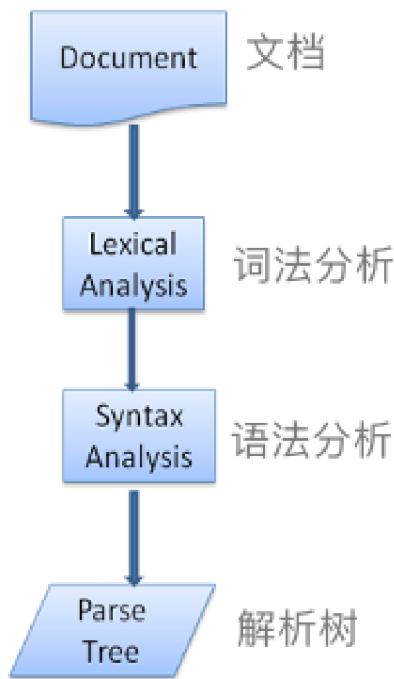
4. 浏览器渲染过程 (工作原理)



解析html 为DOM 树

- HTML字节流解码变为字符流。
- 根据不同编码方式，如UTF-8 GBK来解码
- 词法分析：将字符流解析为一个个词语
- 语法分析：通过不同标签，生成node节点

构建DOM树：将node节点组织成DOM树



5. 浏览器内核

浏览器最重要或者说核心的部分是“Rendering Engine”，可大概译为“渲染引擎”，不过我们一般习惯将之称为“浏览器内核”。负责对网页语法的解释（如标准通用标记语言下的一个应用HTML、JavaScript）并渲染（显示）网页。

所以，通常所谓的浏览器内核也就是浏览器所采用的渲染引擎，渲染引擎决定了浏览器如何显示网页的内容以及页面的格式信息。不同的浏览器内核对网页编写语法的解释也有不同，因此同一网页在不同的内核的浏览器里的渲染（显示）效果也可能不同，这也是网页编写者需要在不同内核的浏览器中测试网页显示效果的原因。

名称	前缀	示例	代表浏览器
Gecko内核	-moz-	-moz-animation	Firefox
Webkit内核	-webkit-	-webkit-animation	Safari、曾经的Chrome
Trident内核	-ms-	-ms-animation	IE
Presto内核	-o-	-o-animation	Opera (挪威)
Blink	-webkit-	-webkit-animation	Chrome/Chromium、Edge

课后实践2

提交时间：第6周前。

策划设计并制作一个以浏览器的“前世今生”为题材的网站（要求：小而完整）。

课堂实践：[网页切片案例（畅游古诗乐园）](#)

第三讲 Markdown教程



- **Markdown**是一种轻量级标记语言，它允许人们使用易读易写的纯文本格式编写文档。
 - **Markdown**语言在 2004 由约翰·格鲁伯（英语：John Gruber）创建。
 - **Markdown**编写的文档可以导出 HTML、Word、图像、PDF、Epub 等多种格式的文档。
 - **Markdown**编写的文档后缀为 **.md, .markdown**。
-

1. Markdown应用

- **Markdown**能被使用来撰写电子书，如：Gitbook。
- 当前许多网站都广泛使用**Markdown**来撰写帮助文档或是用于论坛上发表消息。例如：GitHub、简书、reddit、Diaspora、Stack Exchange、OpenStreetMap、SourceForge等。

2.What is Markdown?

在讲**Markdown**的使用前，我觉得还是很有必要讲讲啥是**Markdown**。

Markdown在我看来其实有两重含义：

- 第一重：文档格式，用**Markdown**编写的文档保存会有两种格式，**.md , .markdown**。
- 第二重：语言，我相信你一定或多或少了解过计算机语言，编程语言，像C, Python这些，而如果你的见闻再广泛一些，你可能知道现在很多的网页都是采用 html 来编写，而 html 是一种标记语言，巧了 **Markdown**也是一门轻量级标记语言，它允许使用易读易写的纯文本格式编写文档，也可以导出Html、word、图像、pdf等格式的文档。

3.为什么要使用markdown?

排版

相信，在你看完我前面对**Markdown**的介绍后，可能已经有一个初映象了。但是你可能会疑惑，**Markdown**看起来确实蛮方便，但是如果编辑文本，写文档，我不是有word, wps，甚至是记事本，我整个txt文件就行了，为什么还要用**Markdown**? 但是，在你一通编辑后，如果你和我一样是个分享狂的话，在你打算将你编写的内容输出为博客到一些开源社区像CSDN、掘金、博客园、GitHub等网站，你会发现即使你在Word, WPS已经排版好了，一旦传输到上面的那些网站上，你会发现格式，排版全不乱了。为什么会这样，如果你细心看本文且记忆力不错的情况下一定知道我前面提到的现在的网站基本上都是采用html来编写。而html的排版，逻辑其实和纯文本或者word的doc文档的格式差异还是比较大的，这样就造成一个问题——排版逻辑差异过大导致格式错乱!

键盘

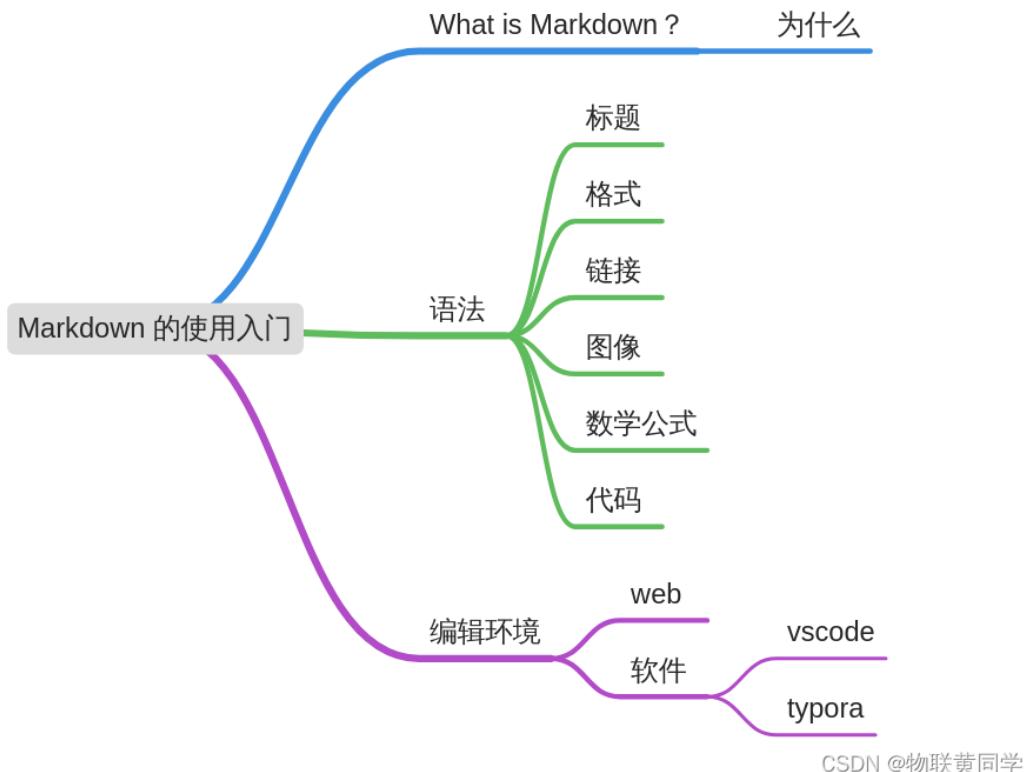
除此之外，对于很多程序员而言，使用键盘的频率要远高于使用鼠标，如果经常使用鼠标，其实会降低内容的输出效率，他们其实非常需要一套较少使用鼠标的排版方式。

格式

而这世界很简单，有需求就会有人去满足需求，于是乎，就有人发明了**Markdown**，一方面**Markdown**本身作为标记语言，在逻辑上也比较接近html，且由于各种**Markdown**的优秀特性，很多网站都具备了**Markdown**文档的接口；另一方面**Markdown**本身的代码量极低，接近记事本的纯文本，却可以实现多种格式和排版，且几乎不需要使用到鼠标，因此深受程序员等群体的喜爱。

4.开始学习MD

[Markdown 教程 | 菜鸟教程 \(runoob.com\)](#)



脚注

[Google](#)

test¹

多选框

PC屏幕

Mobile屏幕

图片

表格

表头	表头
单元格	单元格
单元格	单元格

表头	表头
-----	-----
单元格	单元格
单元格	单元格

red

值班人员	星期一	星期二	星期三
	李强	张明	王平

上下标

H~2~O

- H~2~O
- H₂O

x^10^

- x^10^
- x¹⁰

公式

本节重中之重，我知道，该篇文档的阅读对象又不少是对数学感兴趣的同學。我不知道你们在word或者wps有没有写过公式，如果写过你一定会觉得比较麻烦，而且能写的比较有限。那么现在，可能就需要扯到另外一个知识了，那就是**Latex**，关于这个，我不想过多描述，如果后面有机会，会出一期关于**Latex**的使用的文档。之所以说这个，是因为**Markdown**的数学公式语法其实就是**Latex**语法，虽然有些说法说其实并不是，但在我的个人感觉中，二者的语法是几乎一模一样的，当然，**Markdown**中其实也就只支持**Latex**的一小部分，但我个人感觉也相当够用了。

Markdown的数学公式有两种：

多行公式

也称为公式块，我们用\$\$ \$\$，在中间填入**Latex**语法实现公式的输出。

行内公式

如果我们想要在文本段落中显示公式，且公式单行，那我们可以用 \$\$，在中间填写**Latex**语法表达式输出数学公式。

下面的就是一个多行公式，用**Markdown**编写出来的。

$$f(x) = \sin(x) + 12$$

$$2x=f(x)$$

$$2^x=f(x)$$

$$2^x=f(x)$$

$$2^x=f(x)$$

$$2^x = f(x)$$

$$\begin{Bmatrix} a & b \\ c & d \end{Bmatrix}$$

$$\begin{array}{ccc} A & \xrightarrow{a} & B \\ b \downarrow & & \uparrow c \\ C & \xlongequal{\quad} & D \end{array}$$

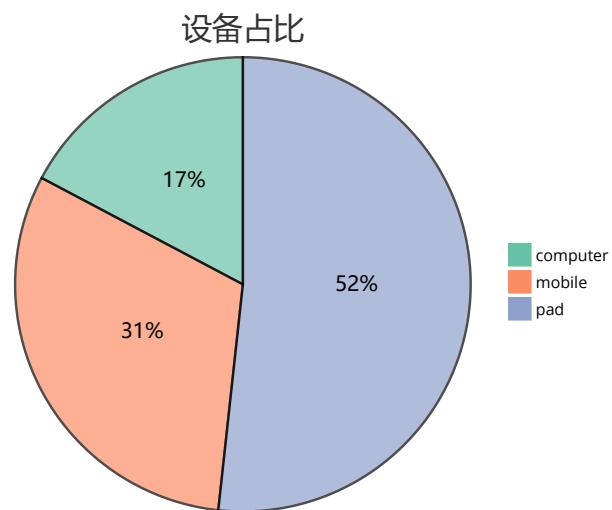
```
\begin{Bmatrix}
a & b \\
c & d
\end{Bmatrix}
```

<Empty Math Block>

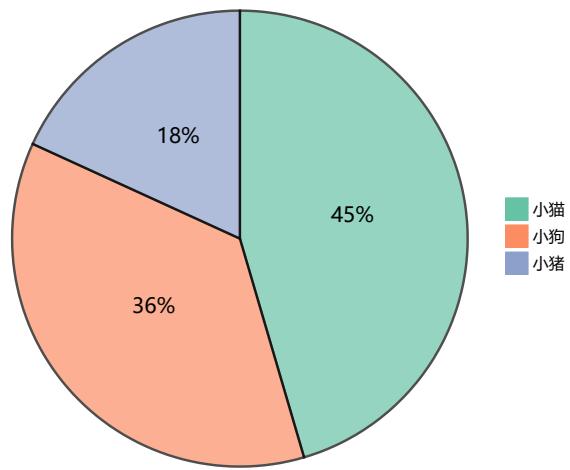
```
\begin{CD}
A @>a>> B \\
@VbVV @AAcA \\
C @= D
\end{CD}
```

$$a^2 + b^2 = c^2$$

绘图

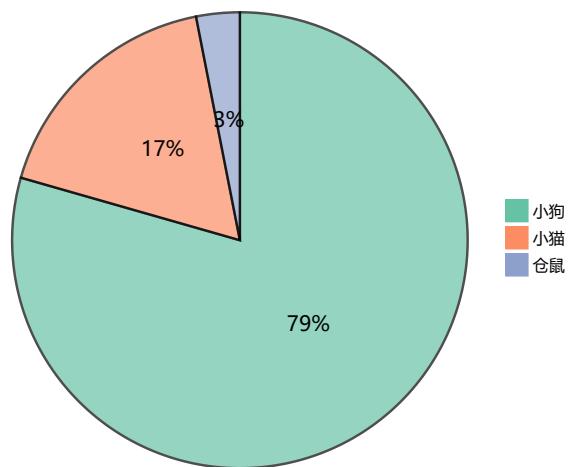


志愿者领养的占比



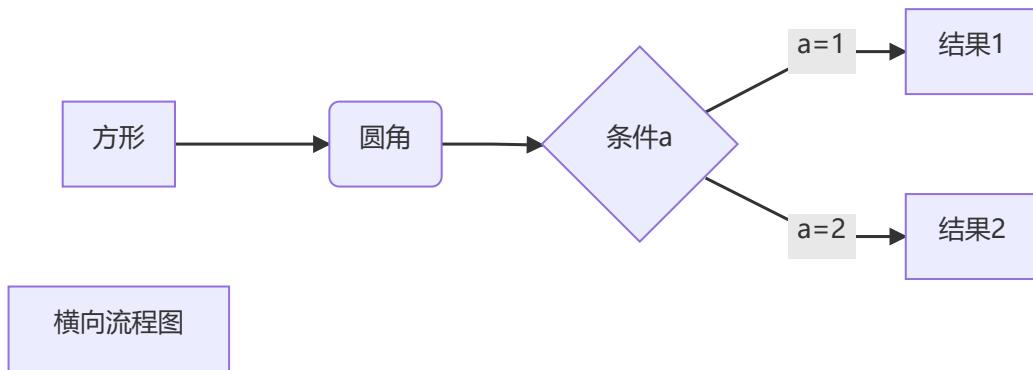
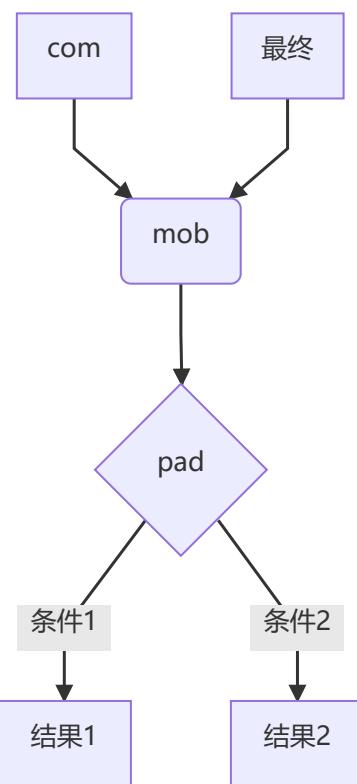
饼图

志愿者领养的宠物占比

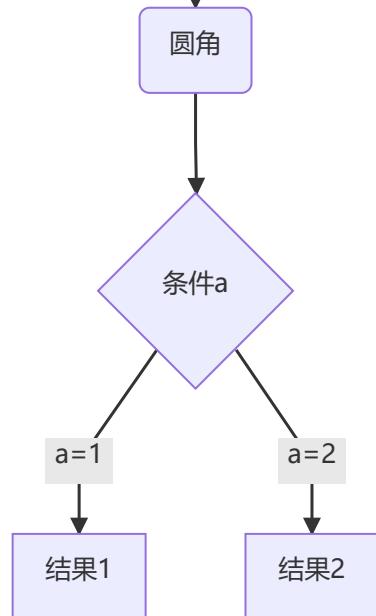


流程图

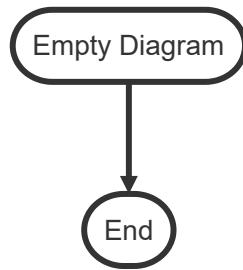
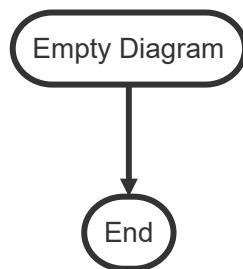
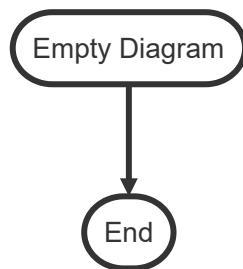
横向流程图

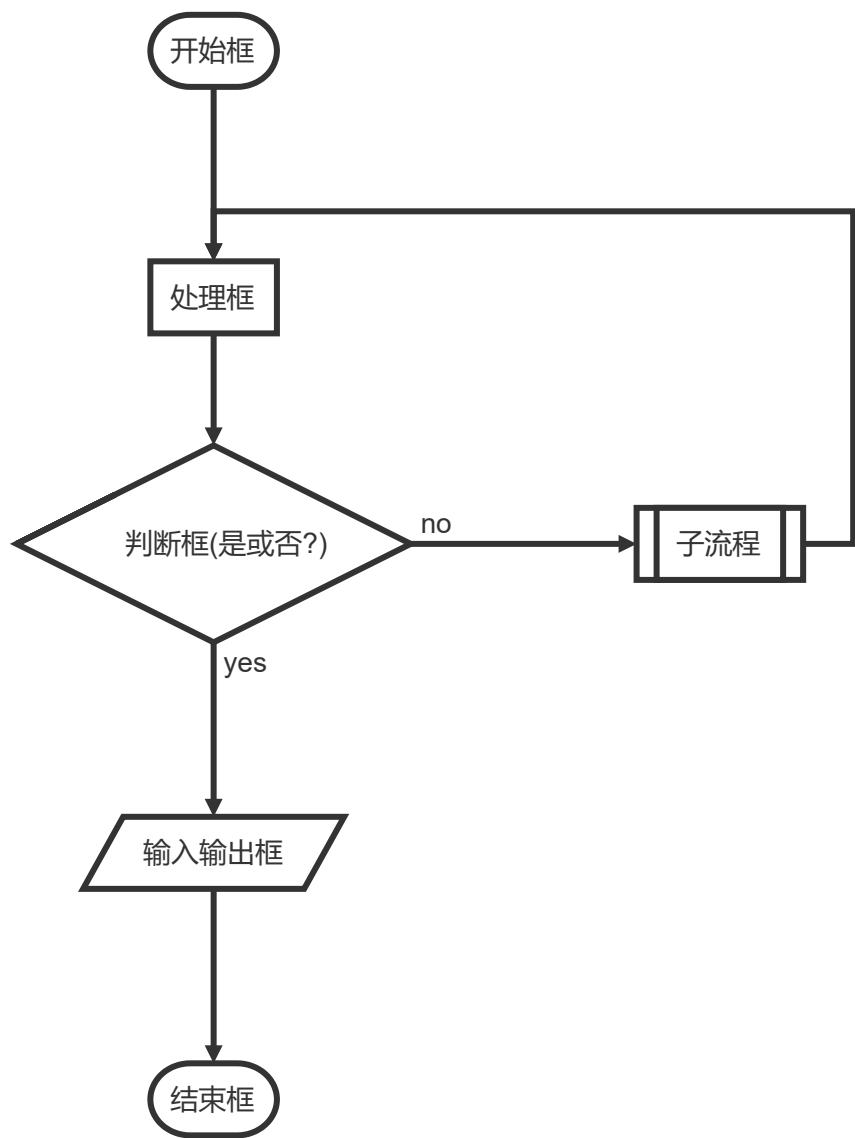


横向流程图

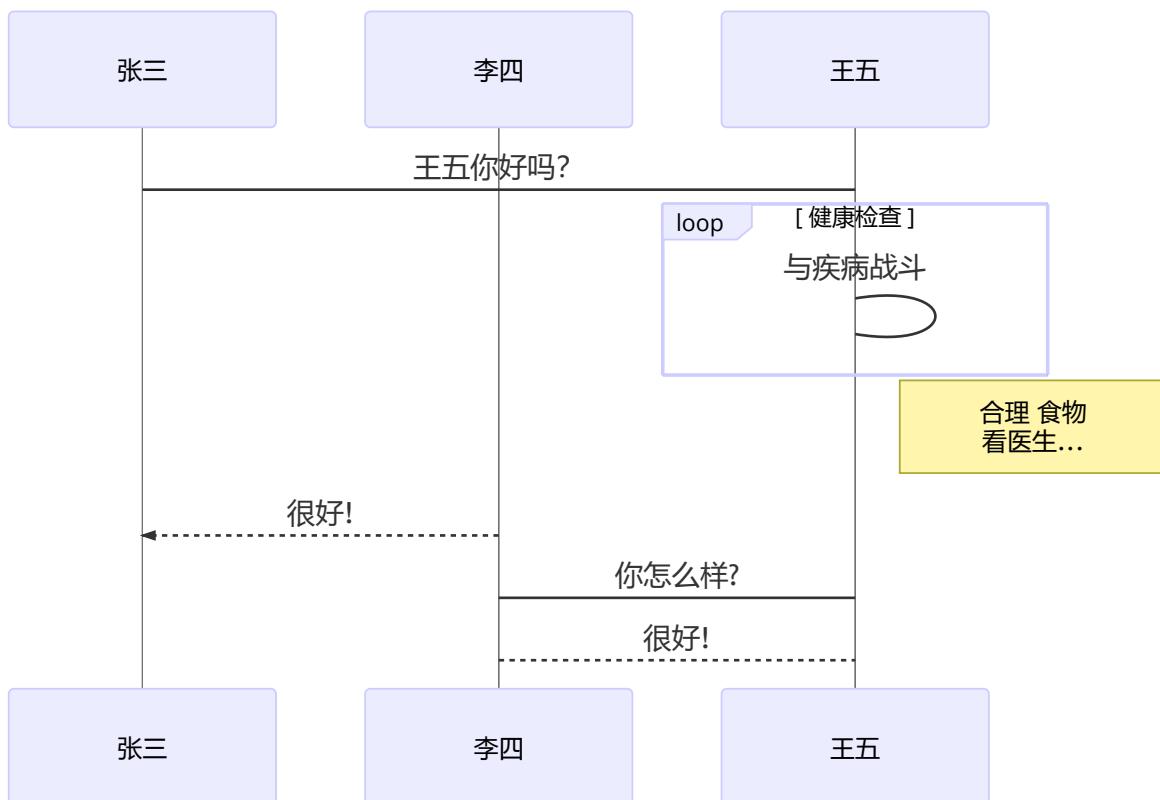


标准流程图

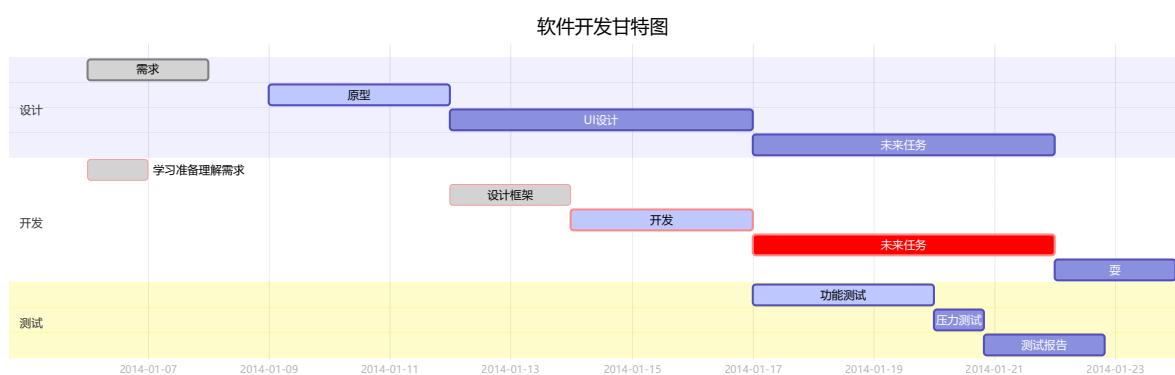




UML时序图

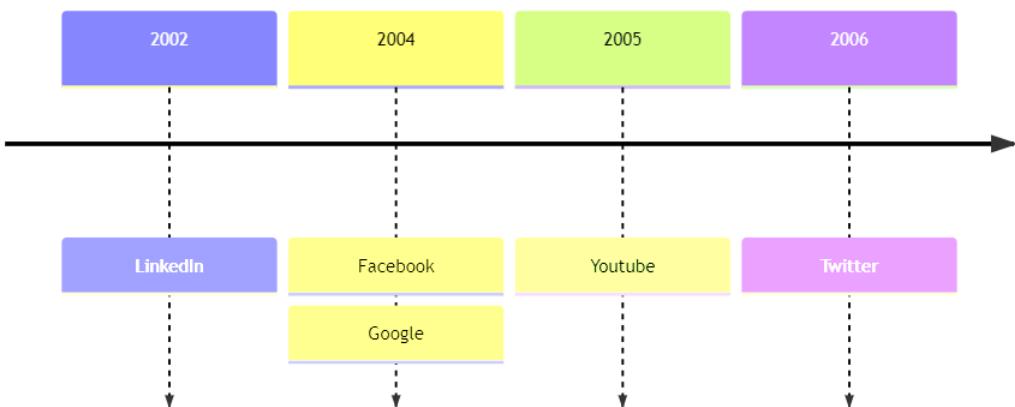


甘特图



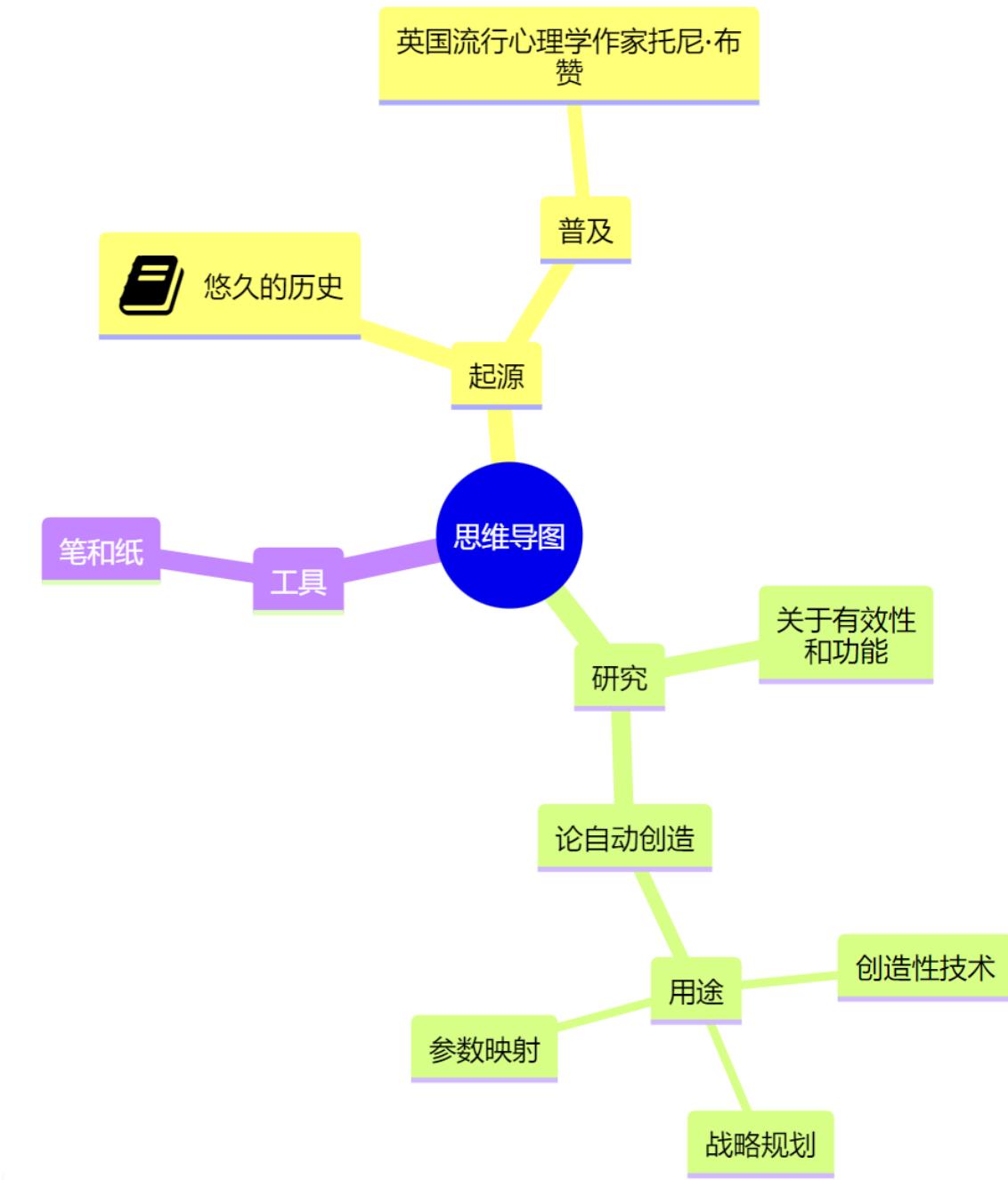
时间线

History of Social Media Platform



Mindmap(VS code)

```
```mermaid
mindmap
root((思维导图))
 起源
 悠久的历史
 ::icon(fa fa-book)
 普及
 • 英国流行心理学作家托尼·布赞
 研究
 关于有效性和功能
 论自动创造
 • 用途
 • 创造性技术
 • 战略规划
 • 参数映射
 工具
 笔和纸
````
```



```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Mermaid 思维导图</title>
    <script src="https://cdn.jsdelivr.net/npm/mermaid/dist/mermaid.min.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/mermaid/dist/mermaid.js"></script>
  </head>
  <body>
    <pre class="mermaid">
mindmap
root((思维导图))
起源
悠久的历史
::icon(fa fa-book)
普及

```

```

英国流行心理学作家托尼·布赞
研究
关于有效性<br/>和功能
论自动创造
用途
创造性技术
战略规划
参数映射
工具
笔和纸
Mermaid
</pre>
<script>
const config = {
  startOnLoad: false,
  securityLevel: 'loose',
};
mermaid.initialize(config);
</script>
</body>
</html>

```

MarkMap(VS code插件)

Markdown的使用入门**

What is MarkDown? 为什么

语法

\- [标题]()

\- 格式

\- 链接

\- 图像

\- 数学公式

\- 代码

\---

编辑环境

web

软件

VS code

插件



5.MD编辑器

Web

[CSDN](#)

Typora

很好用，而且如果你不熟悉Markdown语法，其实你使用该软件的来上手其实会比纯写Markdown会更快，因为它集成了很多快捷键，使用起来还是很丝滑的。

美中不足：

新版本收费了，如果支持正版，可能要有所破费了。

在文档篇幅过大的情况下，其实卡顿会比较明显。

VS code

你没看错，也没听错，VS code可以用来写Markdown！

我愿意称VS code为宇宙第一文本编辑器！它几乎可以用来写一切文档，程序！

当然，熟悉VS code的小伙伴肯定知道，VS code需要一些插件来支撑我们的程序环境，Markdown也是如此，我们需要下面这几个小插件：

Markdown All in One：其实Markdown的编辑语法的支持插件。

Markdown Preview Enhanced：就是将Markdown语法的内容渲染到成为可视化文本界面，就是我们常看到的文本界面。

Markdown PDF：我个人有时候需要把md文档输出为pdf格式，所以这是一个不错的插件。

对比Typora：

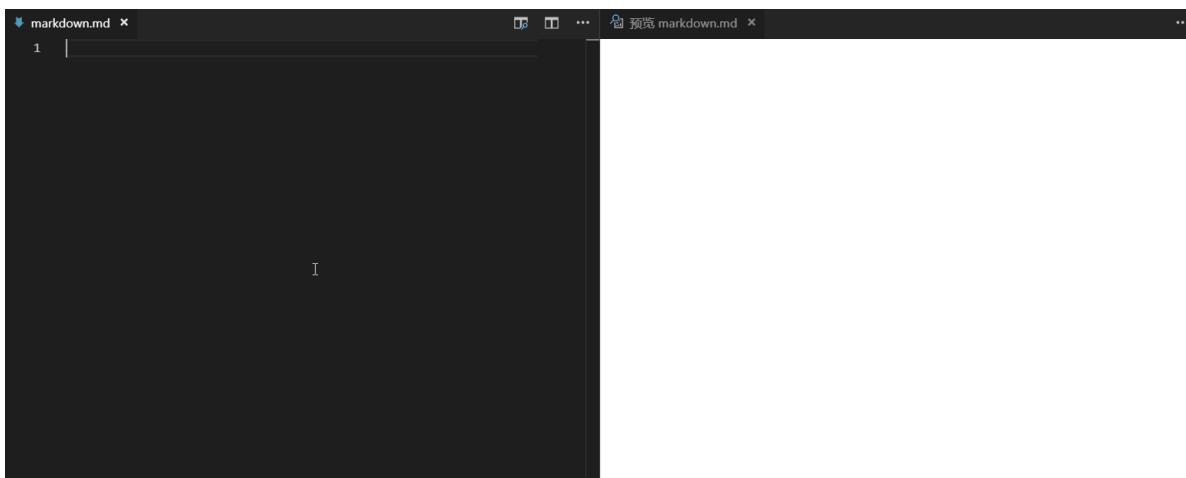
1. 开源免费，且比Typora更快。
2. 使用纯Markdown语法，所以上手需要多一点时间。

6.举例

这个链接用1作为网址变量 [RUNOOB](#).

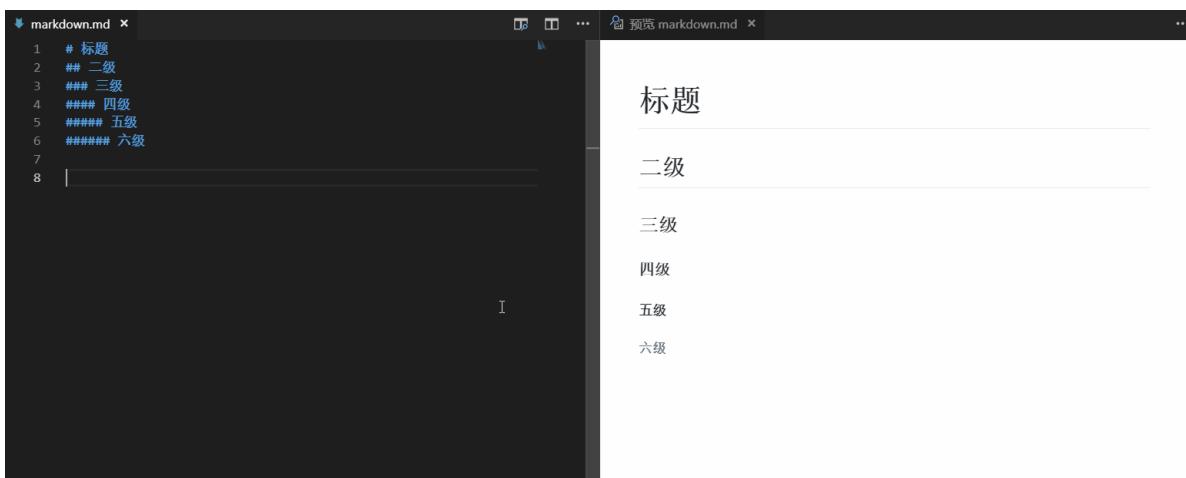
然后在文档的结尾为变量赋值（网址）

标题编辑



A screenshot of a dark-themed code editor window. The title bar shows the file name "markdown.md". The main area is mostly empty, with a single character "I" at the top center.

正文



A screenshot of a dark-themed code editor window. The title bar shows "markdown.md". The left pane contains the following text:

```
1 # 标题
2 ## 二级
3 ### 三级
4 #### 四级
5 ##### 五级
6 ##### 六级
7
8 |
```

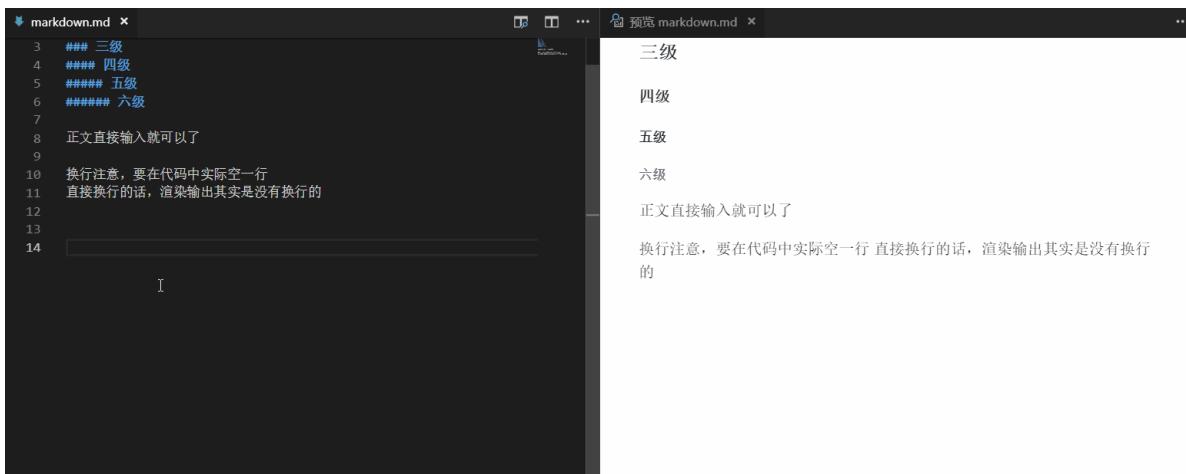
The right pane shows the rendered output:

标题
二级
三级
四级
五级
六级

代码块

干IT的少不了和各种代码，命令行接触；Markdown的代码块功能就非常有用了——

代码块通过两行 ``` 符号框出，如果你写的代码是某种语言，那么可以在第一行末尾加上这个语言的名字，代码块内的代码就会执行对应的高亮语法，例如python



A screenshot of a dark-themed code editor window. The title bar shows "markdown.md". The left pane contains the following text:

```
3 ### 三级
4 #### 四级
5 ##### 五级
6 ##### 六级
7
8 正文直接输入就可以了
9
10 换行注意，要在代码中实际空一行
11 直接换行的话，渲染输出其实是没有换行的
12
13
14 |
```

The right pane shows the rendered output:

三级
四级
五级
六级
正文直接输入就可以了
换行注意，要在代码中实际空一行 直接换行的话，渲染输出其实是没有换行的

行内代码

正文中的代码，则通过输入```框出

The screenshot shows a comparison between a code editor and its preview pane. In the editor (markdown.md), lines 13-26 contain code blocks:

```
13 ...
14 ...
15 show ip interface brief
16 router ospf 1
17 ...
18 ...
19 ...
20 ```python
21 print (
22     'hello, world'
23 )
24 ...
25 ...
26 |
```

In the preview pane (预览 markdown.md), the code is rendered:

```
show ip interface brief
router ospf 1

print (
    'hello, world'
)
```

列表

有序列表

输入数字，加一个句点，然后空格即可；可以缩进空置多级列表；

The screenshot shows a comparison between a code editor and its preview pane. In the editor (markdown.md), lines 13-26 contain an ordered list:

```
13 ...
14 ...
15 show ip interface brief
16 router ospf 1
17 ...
18 ...
19 ...
20 ```python
21 print (
22     'hello, world'
23 )
24 ...
25 ...
26 正文中的代码 [show ip interface brief] |
```

In the preview pane (预览 markdown.md), the list is rendered:

```
show ip interface brief
router ospf 1

print (
    'hello, world'
)

正文中的代码 show ip interface brief
```

无序列表

输入 - ,然后空格

The screenshot shows a comparison between a code editor and its preview pane. In the editor (markdown.md), lines 26-37 contain both ordered and unordered lists:

```
26 正文中的代码 [show ip interface brief]
27 ...
28 有序列表
29 1. 123
30 2. 456
31 3. 789
32   1. 123
33     1. 123
34   1. 123
35 ...
36 无序列表
37 |
```

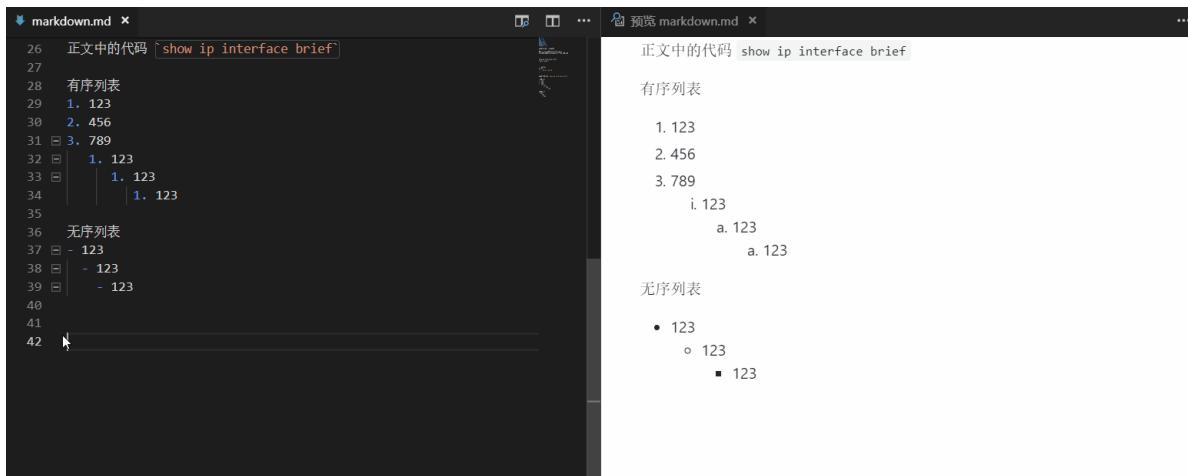
In the preview pane (预览 markdown.md), the lists are rendered:

```
正文中的代码 show ip interface brief

有序列表
1. 123
2. 456
3. 789
  1. 123
    1. 123
  1. 123
a. 123
  a. 123

无序列表
```

加粗(Ctrl+b)和倾斜



The screenshot shows two windows side-by-side. The left window is titled 'markdown.md' and contains raw Markdown code. The right window is titled '预览 markdown.md' and shows the rendered output. The rendered output includes:

- 有序列表
 - 1. 123
 - 2. 456
 - 3. 789
 - 4. 123
 - 5. 123
- 无序列表
 - 123
 - 123
 - 123
- 代码块

```
show ip interface brief
```

课后实践3(2023.09.25)

整理完成一篇有关Markdown的文章。

要求：

- Markdown工具与语法
- 文字排版
- 图片
- 参考文献引用
- Graph或flow绘制流程图
- MindMap或MarkMap绘制思维脑图
- 导出文件格式PDF和HTML

第四讲 CSS入门

1.样式来源（CSS）

CSS

网页主要由三部分构成：

- HTML：网页的骨架，网页的结构设计
- CSS：网页的样式，用于美化网页，有了css就可以实现像素级的还原
- JavaScript：动态脚本，控制页面中的动态效果、数据交互等

基本概念

- 概念：CSS (Cascading Style Sheets)，主要用于控制网页的样式，不能单独使用，它需要作用在HTML
标签上，控制标签的样式。
- 层叠：css样式可以写在多个地方，首先检查代码中有没有样式，有的话会将内联样式、内部样式、外部样式、浏览器默认的样式、浏览器用户自定义的样式叠加在一起，最新形成一套样式，相同的属性的样式层叠掉，不同的属性的样式直接作用在标签上。
- 样式表：指的是css样式代码，页面上通过css编写的代码渲染的样式。页面中通过标签上style属性书写的样式、在style标签中书写的样式代码，在外部css文件中书写的css代码。

网页中样式的来源

所有的标签默认都没有样式

浏览器默认的样式

- h标签、p标签等默认没有任何样式，浏览器在渲染页面时，默认给不同的标签添加不同的样式

浏览器用户自定义的样式

- 浏览器默认给每个标签添加样式，还提供了用户可以自己更改对应的一些样式，比如，字号，字体样式等

内联样式

- 通过在标签上添加style属性，在属性值中书写css样式代码
- 语法：

```
<p style="css属性名1:css属性值1;css属性名2:css属性值2; css属性名3:css属性值3;">文本</p>
```

- 注意：每个html标签上都有一个style属性
- 好处：
 - 哪儿需要添加就在哪个标签上即可
- 缺点：
 - 多个标签需要设置相同的样式时，需要重复添加

内部样式

- 通过在head标签中，添加style标签，在style标签中通过选择器来书写css代码
- 语法：

```
<head>
  <style>
    选择器{
      css属性名1:css属性值1;
      css属性名2:css属性值2;
      css属性名3:css属性值3;
    }
  </style>
</head>
```

- 好处：
 - 结构代码和样式分离
 - 结构代码更加清晰，便于后期维护
 - 多个标签使用相同的样式时，可以提取公共样式，可以同时设置多个标签的样式
- 缺点：
 - 结构代码和样式没有完全分离

外部样式

- 先创建一个后缀名为.css的文件，在css文件中通过选择器书写代码，再在head标签中通过link标签引入的css文件
- 语法：

```
<link rel="stylesheet" href="需要引入的css文件的文件路径">
```

- 好处：
 - 结构代码和样式代码完全分离
 - 便于后期维护，维护根据方便，一处改处处改
 - 以后我们可以将css文件进行压缩，减少文件的体积，优化网页

注意：

- 相同选择器时，优先级：内联样式的优先级（权重）最高，内部和外部样式采用的就近原则，离标签最近的优先作用
- 同一个标签上作用相同的css样式时，后面的样式盖住前面的样式

2.选择器语法

选择器

- 用来找到页面中满足条件的标签，作用的css样式

标签选择器

- 通过标签名找到满足条件的标签，默认找到页面中所有的标签
- 语法：

```
标签名{  
    CSS属性名1:css属性值1;  
    CSS属性名2:css属性值2;  
    CSS属性名3:css属性值3;  
}
```

类选择器（class选择器）

- 通过标签上的class名找到满足条件的标签
- class可以重复，多个标签可以使用同一个class名，可以用于提取公共样式
- 同一个标签上，可以设置多个class名，中间使用空格隔开
- 语法：

```
.class名{  
    CSS属性名1:css属性值1;  
    CSS属性名2:css属性值2;  
    CSS属性名3:css属性值3;  
}
```

- 注意：
 - 选择器可以组合在一起使用，中间没有空格，代表并且的意思

```
/*找到标签上的class名中既有aa又有bb的标签作用样式*/
.aa.bb{
    css样式代码
}
```

id选择器

- 通过标签上的id名找到对应的标签
- id是唯一的，一个页面同名的id只有一个，id选择器只能找到页面中的一个标签
- id命名规则：
 - 是以数字、字母、_和-构成
 - 不能以数字开头，不能包含特殊的符号
- 语法：

```
#id名{
    css属性名1:css属性值1;
    css属性名2:css属性值2;
    css属性名3:css属性值3;
}
```

- 注意：id是唯一的，以后的js会通过id名来找到页面中标签，id选择器慎用。

优先级（权重大小）

- 相同的选择器（相同权重）时，内联的权重最大，内部和外部样式采用就近原则，离标签最近的样式优先作用
- 相同选择器时，后面的样式会盖住前面的
- 不同选择器时，优先级：id选择器>类选择器>标签选择器

3. 背景样式

- `background-color`：设置背景颜色
 - 颜色单词
 - #十六进制 #FF0011 FF代表rgb中red, 00代表rgb的green, 11代表rgb中blue
 - rgb(0~255,0~255,0~255)
 - rgba(0~255,0~255,0~255,0-1) (颜色透明) 引申一下：整体透明度用 `opacity:value` 写法，value取值为0-1。
- `background-image`：设置背景图片

```
background-image: url(背景图片的文件路径);
```

- 如果背景颜色和背景图片同时设置，背景图片会盖住背景颜色
- `background-repeat`：设置背景图片是否平铺
 - `repeat`：默认值，沿着x轴和y轴都平铺
 - `repeat-x`：沿着x轴进行平铺
 - `repeat-y`：沿着y轴进行平铺
 - `no-repeat`：不平铺
 - 应用：像比较规则的渐变背景，可以使用1px的渐变背景平铺，减少图片的体积，优化网页
- `background-size`：设置背景图片的大小
 - `contain`：宽度和高度铺满一边，另一边不管

- `cover`: 两边都铺满, 超出部分隐藏
- 宽度 高度

`宽度 高度`: 如果只设置一个值, 代表宽度, 高度等比例缩放

- 像素
- 百分比: 参考当前盒子的宽度和高度

- `background-position`: 设置背景图片的显示位置

`background-position:x轴 y轴;`

- 方位单词: 两两搭配使用
`left` (左) 、`right` (右) 、`center` (居中) 、`top` (上) 、`bottom` (下)
- 像素: `0px 0px`相当于左上角
- 百分比: `50% 50%`相当于`center center`
 - 注意: 默认是参考的当前盒子的宽高, 如果一旦设置了背景图片固定, 那么参考的是整个显示区域
- `background-attachment`: 设置背景图片是否固定在页面上
 - `scroll`: 默认值, 随着滚动条滚动
 - `fixed`: 固定在页面上
- `background`: 复合属性

```
background: pink url(img/img-3.jpg) no-repeat 100px 100px /50px 50px;
/*前面代表显示位置, /后面代表背景图片的大小*/
```

- 注意: 设置两个值代表背景图片的显示位置, 如果需要设置背景图片的大小, 需要使用`/`将两组值分隔开, `/`前面代表显示位置, `/`后面代表背景图片的大小

应用：雪碧图实现

- 概念: 将多张图片有序地组合在一起, 形成一张图片, 可以减少请求服务器的次数, 优化网页
- 原理: 通过 `background-image` 引入雪碧图, 再通过 `background-position` 调整背景图片的显示位置
- 步骤:
 1. 创建一个标签, 设置宽高, 宽高刚好是需要显示的图标的大小
 2. 通过 `background-image` 引入背景图片 (雪碧图)
 3. 使用 `background-position` 调整雪碧图的显示位置
- 注意: 在开发中, `x轴`的正方向是水平向右, `y轴`的正方向垂直向下



```
img.home{
  width:46px;
  height:44px;
  background:url(img_navsprites.gif) 0 0;
}
```

4.文本样式

- `color`: 设置文本的颜色
 - 单词
 - 十六进制
 - `rgb(0~255,0~255,0~255)`
- `text-align`: 设置文本的对齐方式
 - `left`: 左对齐
 - `center`: 居中对齐
 - `right`: 右对齐
 - 注意: 设置在标签上可以控制文本的对齐方式, 也可以控制标签中行级元素的对齐方式
- `line-height`: 设置行高
 - 像素
 - 百分比或数字: 参考的是当前标签的字体大小 `font-size`
 - 注意: 针对单行文本的水平垂直居中, 设置 `text-align:center;` 和 `line-height:盒子的高度;`
- `text-decoration`: 设置文本的修饰
 - `none`: 无, 可以用于取消a标签的下划线
 - `underline`: 下划线
 - `line-through`: 中划线, 删除线
 - `overline`: 上划线
- `letter-spacing`: 设置字符间距, 一个中文就是一个字符, 一个英文字母就是一个字符
- `word-spacing`: 设置字间距, 以空格来区分
- `text-transform`: 设置字母的大小写
 - `none`: 默认值, 无
 - `capitalize`: 首字母大写
 - `uppercase`: 全大写
 - `lowercase`: 全小写
- `text-indent`: 设置首行的缩进

字体样式

`serif`和`sans-serif`字体之间的区别



💡 在计算机屏幕上, `sans-serif`字体被认为是比`serif`字体容易阅读

- `font-family`: 设置字体类型
 - 注意:
 - 可以设置多种字体, 中间使用逗号隔开, 浏览器先在系统中找第一种字体, 如果有直接作用, 如果没有找第二种字体, 依次类推, 直到找到最后一种字体。一般最后的`serif`字

- 体或者其他字体都是通用字体
 - 多种字体风格比较类似，为了保证页面效果
 - 字体名称如果是中文或者多个英文单词，需要使用引号包裹
- `font-size`：设置字体大小，值越大，字号越大
- `font-weight`：设置字体的粗细
 - `100~900`：数字越大，加粗效果越好，不能带单位
 - `normal`：默认值，正常
 - `lighter`：更细
 - `bold`：加粗
 - `bolder`：更粗
- `font-style`：设置字体的风格
 - `normal`：默认值，正常
 - `italic`：斜体
 - `oblique`：倾斜
- `font`：复合属性

```
p.ex2{
  font:italic bold 12px/30px Georgia, serif;
}
```

字体引用技术

- 在网页中引用自己的字体文件
- 语法：

```
@font-face{
  font-family:字体的名称;
  src: url(引入的字体文件格式1的文件路径) , url(引入的字体文件格式2的文件路径);
}
div{
  font-family: 字体的名称;
}
```

- 注意：
 - 一个`@font-face`只能引入一种字体
 - 为了解决不同浏览器对字体文件格式的兼容问题，可以引入多种文件格式，中间使用逗号隔开

行和块分类

行内标签

- 一般用于组织页面中文本等信息
- 特点：
 - 可以同行显示，排列不下就会换行显示
 - 不支持宽高，由内容撑开
- 常见的标签：`span`, `label`, `a`, `b/strong`, `i/em`等

行内块级标签

- 介于行内元素和块级元素之间，既有行内元素的特点，又有块级元素的一些特点
- 特点：
 - 可以同行显示，排列不下就会换行显示
 - 支持宽高
- 常见的标签：img、input（text）、button、td等
- 注意：行内块级元素之间存在5px的间距

块级标签

- 一般用于页面布局或者组织行级标签
- 特点：
 - 独占一行
 - 支持宽高
- 常见的标签：h、p、table、tr、ul、ol、li、div等

相互转换

- 通过 `display` 属性可以实现相互转换
- 取值：
 - `display:inline;`：转为行内元素
 - `display:inline-block;`：转为行内块级元素
 - `display:block;`：转为块级元素
 - `display:none;`：隐藏元素，页面进行重新绘制

超链接

- 给超链接添加样式
- 语法：

```
/* 向未被访问过的超链接添加样式 */
a:link {
    color: green;
}

/* 向访问过的超链接添加样式 */
a:visited {
    color: yellow;
}

/* 向鼠标悬停的超链接添加样式 */
/* :hover必须放在:link和:visited之后 */
a:hover {
    color: red;
}

/* 向获取焦点的超链接添加样式 */
/* :active必须放在:hover之后 */
a:active {
    color: blue;
}
```

- 注意：`:hover` 必须放在 `:link` 和 `:visited` 之后，`:active` 必须放在 `:hover` 之后，顺序：link-visited-hover-active (L-V-H-A)

列表样式

- 设置列表项的标志的效果
- 属性：
 - `list-style-type` : 设置列表项标志的类型
 - `none` : 无
 - `list-style-position` : 设置列表项标志的显示位置
 - `outside` : 默认值，标志显示在li标签内容之外
 - `inside` : 标志现在li标签内容区域里面
 - `list-style-image` : 设置图片作为列表项标志，该图片不能通过css来控制大小，只能通过图像软件更改原图的大小，不建议使用
 - `list-style` : 复合属性
 - `list-style:none`; 取消列表项的标志
- 注意：可以在ul或ol标签上，设置list-style属性，li标签上可以自动继承来使用

表格样式

将以下代码设置在table标签中

- `border-collapse: collapse;` : 将表格边框合并为一 (推荐使用)
- `boder-spacing: 1px` : 设置单元格与单元格之间的间距
- `border-spacing` : 设置单元格与单元格之间的间距

字体图标设计

字体图标

- 图标是以字体的形式显示，可以使用css字体相关的属性进行控制
- 设计师在设计字体时，字体是以图标的形状进行设计
- font awesome官网：[Font Awesome，一套绝佳的图标字体库和CSS框架](#)
- font awesome是通过设置不同的class名来使用不同的图标，图标是矢量图

网页优化的方案

1. 合理使用图片格式，可以减少图片的体积，优化网页
2. 使用雪碧图技术，减少请求服务器的次数，优化网页
3. 1px渐变背景平铺，可以减少图片的体积，优化网页
4. 使用外部样式，可以将css文件进行压缩，减少文件的体积，优化网页
5. 使用字体图标也可以优化网页

应用：导航设计

垂直



水平



导航栏

熟练使用导航栏，对于任何网站都非常重要。

使用CSS你可以转换成好看的导航栏而不是枯燥的HTML菜单。

导航栏=链接列表

作为标准的 HTML 基础一个导航栏是必须的。

在我们的例子中我们将建立一个标准的 HTML 列表导航栏。

导航条基本上是一个链接列表，所以使用

和

- 元素非常有意义：

垂直导航



```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title></title>
<style>
    ul {
        list-style-type: none;
        margin: 0;
        padding: 0;
        width: 200px;
        background-color: #f1f1f1;
    }
    li a {
        display: block;
        color: #000;
        padding: 8px 16px;
        text-decoration: none;
    }
    li a.active {
        background-color: #4CAF50;
        color: white;
    }
    li a:hover:not(.active) {
        background-color: #555;
        color: white;
    }
</style>
</head>

<body>
    <ul>
        <li><a href="#home">主页</a></li>
        <li><a href="#news">新闻</a></li>
        <li><a href="#contact">联系</a></li>
        <li><a href="#about">关于</a></li>
    </ul>
</body>
</html>
```

导航图标



```
<!DOCTYPE html>
<html>
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
<style>
body {margin:0}

.icon-bar {
    width: 90px;
    background-color: #555;
}

.icon-bar a {
    display: block;
    text-align: center;
    padding: 16px;
    transition: all 0.3s ease;
    color: white;
    font-size: 36px;
}

.icon-bar a:hover {
    background-color: #000;
}

.active {
    background-color: #04AA6D;
}
</style>
<body>

<div class="icon-bar">
    <a class="active" href="#"><i class="fa fa-home"></i></a>
    <a href="#"><i class="fa fa-search"></i></a>
    <a href="#"><i class="fa fa-envelope"></i></a>
    <a href="#"><i class="fa fa-globe"></i></a>
    <a href="#"><i class="fa fa-trash"></i></a>
</div>

</body>
</html>
```

水平导航



```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title></title>
<style>
```

```

ul {
    list-style-type: none;
    margin: 0;
    padding: 0;
    overflow: hidden;
    background-color: #333;
}

li {
    float: left;
}

li a {
    display: block;
    color: white;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none;
}

li a:hover:not(.active) {
    background-color: #111;
}

.active {
    background-color: #4CAF50;
}
</style>
</head>
<body>

<ul>
    <li><a href="#home">主页</a></li>
    <li><a href="#news">新闻</a></li>
    <li><a href="#contact">联系</a></li>
    <li style="float:right"><a class="active" href="#about">关于</a></li>
</ul>

</body>
</html>

```

下拉菜单导航



```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title></title>
<style>
ul {
    list-style-type: none;
    margin: 0;
    padding: 0;
    background-color: #999;
    overflow: hidden;
}

```

```
/*
注意: overflow:hidden 添加到 ul 元素,以防止 li 元素列表的外出(当
li{float:left}时)。 来源链接: https://www.runoob.com/try/try.php?
filename=trycss_navbar_horizontal_float&basepath=0 */
}

li {
    float: left;
}

li a {
    color: white;
    padding: 14px 16px;
    display: inline-block;
    text-decoration: none;
}

li>a.active {
    background-color: green;
}

li>a:hover:not(a.active),
.dropbtn:hover {
    background-color: #555;
}

.dropdown-content {
    display: none;
    position: absolute;
    /* 默认相对于<html>进行绝对定位 */
    background-color: #f9f9f9;
    min-width: 100px;
    box-shadow: 0 8px 16px 0 rgba(0, 0, 0, .2);
}

.dropdown-content a {
    color: black;
    display: block;
    /* 因为<a>标签不是块元素, min-width:100px不会生效 */
}

.dropdown-content a:hover {
    background-color: #f1f1f1;
    color: deepskyblue;
}

.dropdown:hover .dropdown-content {
    display: block;
}
/* 中间的空格表示, 鼠标悬在.dropdown上时, dropdown的子元素.dropdown-content变为块
元素 */
</style>
</head>
<body>
<ul>
    <li><a href="#home" class="active">主页</a></li>
    <li><a href="#news">新闻</a></li>
    <li>
        <div class="dropdown">
```

```
<a href="#" class="dropbtn">下拉菜单</a>
<div class="dropdown-content">
    <a href="#">链接 1</a>
    <a href="#">链接 2</a>
    <a href="#">链接 3</a>
</div>
</div>
</li>
<li><a href="#about">关于</a></li>
</ul>
<h3>导航栏上的下拉菜单</h3>
<p>鼠标移动到 "下拉菜单" 链接先显示下拉菜单。</p>
</body>
</html>
```

课后实践5(2023.10.23)



要求：

- Div+CSS左中右布局结构
- 水平导航菜单
- 含三种类型的菜单

改进一下：导航 (2023.10.29)



图片导航

- li:nth-child(n)
- li:nth-child(n):hover
- li:first-child
- li:last-child
- cursor: pointer;

Bootstrap框架创建导航

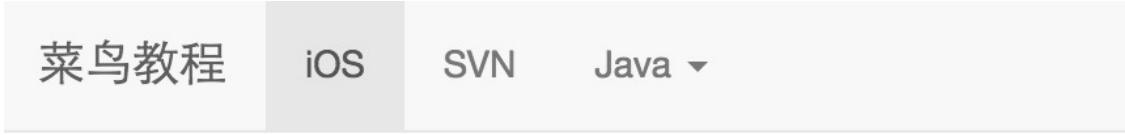
- 引入Bootstrap

```
<!-- 引入 Bootstrap -->
<link
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"
rel="stylesheet">
```

- 下载Bootstrap
 - 从 <http://getbootstrap.com/> 上下载 Bootstrap 的最新版本

```
bootstrap/
  └── css/
    ├── bootstrap.css
    ├── bootstrap.min.css
    ├── bootstrap-theme.css
    └── bootstrap-theme.min.css
  └── js/
    ├── bootstrap.js
    └── bootstrap.min.js
  └── fonts/
    ├── glyphicon-halflings-regular.eot
    ├── glyphicon-halflings-regular.svg
    ├── glyphicon-halflings-regular.ttf
    └── glyphicon-halflings-regular.woff
```

简单导航



```
<nav class="navbar navbar-default" role="navigation">
  <div class="container-fluid">
    <div class="navbar-header">
      <a class="navbar-brand" href="#">菜鸟教程</a>
    </div>
    <div>
      <ul class="nav navbar-nav">
        <li class="active"><a href="#">iOS</a></li>
        <li><a href="#">SVN</a></li>
        <li class="dropdown">
          <a href="#" class="dropdown-toggle" data-toggle="dropdown">
            Java
            <b class="caret"></b>
          </a>
          <ul class="dropdown-menu">
            <li><a href="#">jmeter</a></li>
            <li><a href="#">EJB</a></li>
            <li><a href="#">Jasper Report</a></li>
            <li class="divider"></li>
            <li><a href="#">分离的链接</a></li>
            <li class="divider"></li>
            <li><a href="#">另一个分离的链接</a></li>
          </ul>
        </li>
      </ul>
    </div>
  </div>
</nav>
```

响应式导航



iOS

SVN

Java ▼

```
<nav class="navbar navbar-default" role="navigation">
    <div class="container-fluid">
        <div class="navbar-header">
            <button type="button" class="navbar-toggle" data-toggle="collapse"
                    data-target="#example-navbar-collapse">
                <span class="sr-only">切换导航</span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
            </button>
            <a class="navbar-brand" href="#">菜鸟教程</a>
        </div>
        <div class="collapse navbar-collapse" id="example-navbar-collapse">
            <ul class="nav navbar-nav">
                <li class="active"><a href="#">iOS</a></li>
                <li><a href="#">SVN</a></li>
                <li class="dropdown">
                    <a href="#" class="dropdown-toggle" data-toggle="dropdown">
                        Java <b class="caret"></b>
                    </a>
                    <ul class="dropdown-menu">
                        <li><a href="#">jmeter</a></li>
                        <li><a href="#">EJB</a></li>
                        <li><a href="#">Jasper Report</a></li>
                        <li class="divider"></li>
                        <li><a href="#">分离的链接</a></li>
                        <li class="divider"></li>
                        <li><a href="#">另一个分离的链接</a></li>
                    </ul>
                </li>
            </ul>
        </div>
    </div>
</nav>
```

面包屑导航

- 面包屑导航 (Breadcrumbs) 是一种基于网站层次信息的显示方式。以博客为例，面包屑导航可以显示发布日期、类别或标签。它们表示当前页面在导航层次结构内的位置。

Home / 2013 / 十一月

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Bootstrap 实例 - 面包屑导航</title>
    <link rel="stylesheet" href="https://cdn.staticfile.org/twitter-
bootstrap/3.3.7/css/bootstrap.min.css">
    <script src="https://cdn.staticfile.org/jquery/2.1.1/jquery.min.js">
    </script>
    <script src="https://cdn.staticfile.org/twitter-
bootstrap/3.3.7/js/bootstrap.min.js"></script>
</head>
<body>
    <ul class="breadcrumb">
        <li><a href="#">Home</a></li>
        <li><a href="#">2013</a></li>
        <li class="active">十一月</li>
    </ul>
</body>
</html>
```

面包屑导航

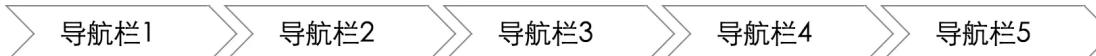
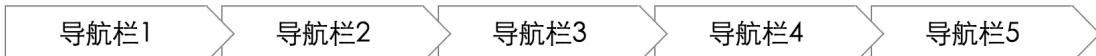
导航栏1

导航栏2

导航栏3

导航栏4

导航栏5



标准盒模型

标准盒模型



上图就是一个盒模型：

- content: 代表内容区域，存放内容的空间，存放文本或者其他的盒子
- padding: 内边距，内容到边框之间的间距，相当于快递中泡沫
- border: 盒子的边框，四周边框可以分别设置

- margin: 盒子的外边距, 盒子与盒子之间的间距 (兄弟关系、父子关系)

边框

边框 border

- 设置盒子四周的边框
- 语法:

```
边框三要素:  
border-方位-width: 宽度;  
border-方位-style: 类型;  
border-方位-color: 颜色;  
复合属性:  
设置一条边框:  
border-方位: 宽度 类型 颜色;  
同时设置四条边框的样式:  
border: 宽度 类型 颜色;  
虚线: dashed
```

- 注意:
 - 边框在设置时至少设置两个值, 宽度和类型, 默认颜色为黑色
- 设置边框会撑大盒子, 会改变盒子的大小
 - **边框的渲染原理:** 如果只设置一条边框, 显示为矩形形状, 如果设置了相邻的边框, 相接的部分是沿着对角线均分

内边距

内边距padding

- 设置内容到边框之间的间距
- 语法:

```
分别设置四周的内边距:  
padding-left  
padding-right  
padding-top  
padding-bottom  
复合属性:  
padding  
一个值: 上下左右  
两个值: 上下 左右  
三个值: 上 左右 下  
四个值: 上 右 下 左
```

- 注意:
 - padding会撑大盒子, 会更改盒子的大小
 - 默认会添加背景的颜色, padding去会填充背景颜色
- 应用:
 - 可以使用padding撑大盒子, 让内容居中
 - 设置内容到边框的显示位置

外边距

外边距margin

- 设置盒子与盒子之间的间距
- 语法：

分别设置四周的外边距：

`margin-left`
`margin-right`
`margin-top`
`margin-bottom`

复合属性：

`margin`

一个值：上下左右

两个值：上下 左右

三个值：上 左右 下

四个值：上 右 下 左

margin重叠性

- 盒子与盒子之间是**兄弟关系**时，margin在垂直方向会发生**重叠**，以值大的为准
- **注意：**水平方向的margin是叠加在一起，值相加

margin-top传递性

- 盒子与盒子之间是父子关系，子元素设置margin-top后，会传递给父元素显示
- 原因：子元素设置margin-top时会去找父元素的边界（参考位置），找不到父元素的边界就会传递父元素显示
- 解决方案
 - 给父元素设置border，会改变父元素的盒子的大小
 - 给父元素设置padding，会改变父元素盒子的大小
 - 给父元素设置`overflow:hidden;`
 - `overflow:hidden;`：盒子中内容溢出隐藏掉，设置了该属性后，会产生一个BFC容器，该容器里面的元素不会影响盒子外面的元素的排列

IE盒模型

怪异盒模型

又称为ie盒模型，css中设置width包含了content+padding+border



目前浏览器默认都是标准盒模型，需要使用怪异盒模型，需要转化

盒模型转化：`box-sizing`

- `border-box`：设置为怪异盒模型
- `content-box`：默认值，设置为标准盒模型

怪异盒子大小的计算

真正的大小的计算

- 宽度 = width (包含了content+ padding+border)
- 高度 = height (包含了content+padding+border)

所占空间的大小的计算

- 宽度 = width (包含了content+ padding+border) + margin * 2 (左右)
- 高度 = height (包含了content+padding+border) + margin * 2 (上下)

IE盒模型-拓展知识

拓展

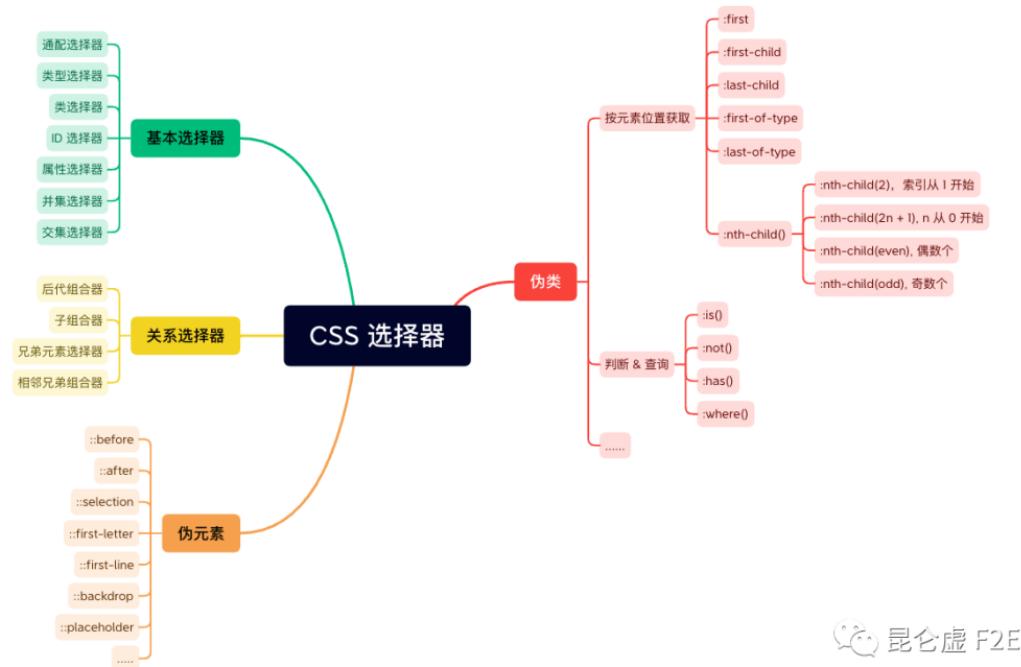
margin:0 auto;

- 让块级元素在父标签中水平居中
- 设置上下为0，左右为auto
- auto：代表自适应的意思，可以将父盒子水平方向的剩余空间均分子元素的左右两边

padding&margin设置百分比

- padding&margin设置百分比时，无论设置哪个方向的百分比，都是参考父元素的宽度

组合选择器



选择器

css主要由两部分构成：

1. 选择器：通过选择器去找到页面中的标签
2. css样式 (多个css样式中间使用分号隔开)

对于选择器来说，我们丰富选择器的写法，种类非常多，能够更加的精确找到页面中标签

选择器

注意：选择器只能从父标签找到子标签，或者从前面对的标签找到后面的标签（从外到内，或从前到后找标签）



id选择器

- 通过标签上的id名找到满足条件的标签
- 语法：

```
#id名{  
    CSS代码  
}
```

- 注意：id是唯一，只能找到页面中唯一的一个标签，以后js会通过id控制页面中标签，id选择器慎用

类选择器

- 又称为class选择器，通过标签上的class名找到满足条件的标签
- 一般用于提取公共样式，作用在不同的标签
- 语法：

```
.className{  
    CSS代码  
}
```

标签选择器

- 通过标签名找到满足条件的标签，如果没有规定范围时，默认会找到页面中所有满足条件的标签
- 语法：

```
标签名{  
    CSS代码  
}
```

后代选择器（派生选择器）

- 通过找到指定标签里面所有满足条件的后代的标签，可以找到儿子、孙子、重孙子...
- 中间使用空格隔开
- 语法：

找到**class**名为**box1**的标签里面的所有后代**div**标签

```
.box1 div{  
}
```

子元素选择器

- 找到指定标签里面满足条件的直接子标签，只找儿子，不找孙子、重孙子...
- 中间使用 > 隔开
- 语法：

找到**class**名为**box1**里面的直接子元素**div**标签

```
.box1>div{  
}
```

兄弟选择器

相邻兄弟选择器

- 找到满足条件的标签后面相邻的第一个满足条件的兄弟标签
- 中间使用 + 隔开
- 语法：

找到**class**名为**box**的标签后面相邻的第一个兄弟标签

```
.box+p{  
}
```

后续兄弟选择器

- 找到满足条件的标签后面所有的满足条件的兄弟标签
- 中间使用 ~ 隔开
- 语法：

找到**class**名为**box**的标签后面所有的兄弟**p**标签

```
.box ~ p{  
}
```

选择器分组

- 中间使用逗号隔开，逗号前后的选择器找到对应的标签分别作用一次css样式
- 语法：

```

p{
    color:red;
}
span{
    color:red;
}
label{
    color:red;
}
等价于
p,span,label{
    color:red;
}

```

属性选择器

属性选择器

- 根据标签上属性名及属性值找到满足条件的标签

选择器	描述
[属性名]	用于选取带有指定属性的元素。
[属性名=属性值]	用于选取带有指定属性和值的元素。 (精确匹配)
[属性名*=属性值]	匹配属性值中包含指定值的某个元素。 (模糊匹配)
[属性名~=属性值]	用于选取属性值中包含指定词汇的元素。
[属性名^=属性值]	匹配属性值以指定值开头的每个元素。
[属性名\$=属性值]	匹配属性值以指定值结尾的每个元素。

[属性名 |= 属性值] : 用于选取带有以指定值开头的属性值的元素，该值必须是整个单词。

- 属性选择器一般用于提取公共样式

伪类选择器

伪类选择器

- 给标签添加一些功能性的效果

选择器	描述
:link	向未被访问过的超链接添加样式
:visited	向访问过的超链接添加样式
:hover	向鼠标悬停的标签添加样式
:active	向被激活的标签添加样式
:focus	向拥有键盘输入焦点的标签添加样式

- 注意：
 - `:hover` 必须放在 `:link` 和 `:visited` 之后，`:active` 必须放在 `:hover` 之后，才有效果
 - `outline`：设置边框外面的一圈轮廓线，使用方式 border一样

```
outline: 宽度  类型  颜色;
取消input获取键盘输入焦点时默认样式
input:focus{
    outline: none;
}
```

```
//sass中hover案例: hover    .header-right2.info显示
.header-right2 {
    &:hover .info{
        display: block;
    }
    .info {
        display: none;
    }
}
```

伪元素选择器

伪元素选择器

- 概念：可以在HTML标签中通过css代码添加一块渲染区域，该区域可以设置样式

选择器	描述
<code>::first-letter</code>	找到指定的标签中的第一个字符添加样式
<code>::first-line</code>	找到指定标签中的第一行添加样式
<code>::selection</code>	向鼠标选中的区域添加样式
<code>::before</code>	在指定的标签的内容之前添加的一块渲染区域
<code>::after</code>	在指定的标签的内容之后添加一块渲染区域

- 注意：
 - `::before` 和 `::after` 都必须搭配 `content: "";`一起使用
 - `::before` 和 `::after` 渲染出来的是一个**行内元素**，如果需要设置宽高，需要转化为行内块级元素或块级元素
 - 可以 `::before` 和 `::after` 来渲染三角形

通配符选择器

- 找到页面中所有的标签添加样式
- 语法：

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

选择器权重

选择器分类

- 基础选择器：id选择器、类选择器、标签选择器、属性选择器、伪类选择器、伪元素选择器、通配符选择器
- 组合选择器：后代选择器、子元素选择器、相邻兄弟选择器、后续兄弟选择器

选择器权重（优先级问题）

- 在相同的选择器下，内联的权重最大，内部和外部采用就近原则
- 基础选择器：id选择器 > 类选择器（属性选择器）> 标签选择器 > 通配符选择器 > 继承的样式
- 组合选择器的权重需要将所有的基础选择器的权重加起来一起比较

选择器权重的计算方法

加法运算

- 内联样式默认为1000
- id选择器默认权重为100
- 类（属性）选择器默认权重为10
- 标签选择器默认权重为1

将所有的标签的权重进行相加，比较权重值的大小，值大的权重越高，如果权重值一样，后面的盖住前面

注意：不满足满10进1的规则，十几个标签选择器都没有一个类选择器的权重高

4个0

(0,0,0,0)

- 第一个0代表是否有内联样式，有则为1，无则为0
- 第二个0代表id选择器的个数
- 第三个0代表类（属性）选择器的个数
- 第四个0代表标签选择器的个数

计算出每种选择器的个数，从第一个0开始对比，值大的权重最大，值相同，对比第二个0，依次类推，如果最后一个0页相同，说明选择器权重值一样，后面样式盖住前面的样式。

拓展：

`!important`：比较暴力，会将css样式直接作用标签上，无视选择器的权重大小，比内联样式的权重都大，在开发中尽量少用，权重值的规则被破坏了，用多了样式没有规律，计算权重值失效，后期维护起来比较麻烦

- 语法：

```
div{  
    /* !important 比较暴力，会将css样式直接作用在标签上，无视选择器的权重 */  
    color: blue !important;  
}
```

样式继承

CSS继承

继承的概念更多出现在编程语言中，js学习继承

css的继承是指子标签可以从父标签上将css样式继承过来，css样式可以作用在子标签上

CSS的继承分为两类：

- 自动继承：子元素不需要任何的操作，可以从父元素上继承css样式
- 手动继承：子元素需要敲代码，指定子元素某个属性从父元素上继承样式

`inherit`:可以控制子元素的属性从父元素上继承该属性的值

```
margin: inherit;
```

哪些属性可以被自动继承

- 文本样式可以被自动继承
 - `color`: 设置文本颜色
 - `text-align`: 设置文本的对齐方式
 - `line-height`: 设置行高
 - `text-decoration`: 设置文本修饰
 - `letter-spacing`: 设置字符间距
 - `word-spacing`: 设置字间距
 - `text-indent`: 设置首行缩进
 - `text-transform`: 设置大小写
- font系列的字体样式也可以被自动继承
 - `font-size`: 设置字体的大小
 - `font-family`: 设置字体的类型
 - `font-weight`: 设置字体的粗细
 - `font-style`: 设置字体的风格
- list-style属性，li标签可以从ul或ol标签上自动继承
- cursor鼠标样式可以自动继承 `pointer` 抓手，(`cursor: pointer;`可以修改鼠标为手指)

哪些属性不可以被自动继承

- `width`、`height`、`margin`、`padding`、`background`、`border`等等

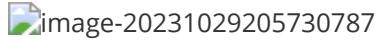
课后实践6(2023.10.29)

观察下面的5个网页

分析结构，找出异同点！！！

任选其一制作，以后在此网页基础上不断改进完善功能，最终的期末大作业！！！





浮动特点

块级元素同行显示

1. display:inline-block

- 可以让块级元素同行显示，存在兼容问题
- 盒子之间存在5px的间距

2. 浮动

浮动

- 可以让元素同行显示，排列不下会自动换行显示，不存在兼容问题
- 语法：

```
float:left | right | none;
```

- left: 设置左浮动
- right: 设置右浮动
- none: 默认值，不浮动
- 特点：
 - 可以让元素同行显示，排列不下时会自动换行显示，不存在兼容问题
 - 浮动元素会脱离文档流，在标准文档流之上
 - 浮动元素不再占用原来的空间
 - 行内元素设置浮动后，行内元素支持宽高
- 注意：
 - 多个元素同时设置浮动时，后面浮动的元素会找到前面浮动元素的边界或第一个浮动元素找到父元素浮动的边界就停止下来
 - 只有设置了浮动的元素才可以同行显示，
 - 块级元素没设置宽度时，如果设置浮动，块级元素的宽度由内容决定

标准文档流

- 在页面布局过程中，页面中元素会按照从上到下，从左到右，块级元素独占一行，行级元素共享一行的排列规范

脱离文档流

- 在页面布局过程中，元素不再遵循标准文档流的规范，按照自己的排列规范进行排列

清除浮动

浮动元素对非浮动元素的影响

- 非浮动的元素会挤占浮动元素原来的空间
- 非浮动元素中如果有文本，文本会被浮动元素挤开
 - 实现图文混排

```
<style>
  img{
    float:left;
  }
</style>

<p>
  文本
</p>
```

- 子元素浮动时，父元素高度会塌陷

清除浮动

- 清除浮动元素对非浮动元素的影响
- 语法：

```
clear: left | right | both;
```

- left：清除左浮动的影响
- right：清除右浮动的影响
- both：清除左浮动和右浮动的影响

清除浮动的方法

1. 给受影响的元素添加clear样式
2. 使用 `<br clear="all">` 将浮动元素和非浮动元素隔开
3. 使用空白的div添加 `clear:both;` 将浮动元素和非浮动元素隔开
4. 给父元素添加伪元素选择器可以清除浮动的影响(推荐使用)

```
.clearfix::after{
  content: "";
  clear: both;
  display: block;
}
```

注意：`overflow:hidden`：可以解决子元素浮动，父元素高度塌陷的问题，借助是BFC容器的特点，不是清除浮动的方法

定位

如何让盒子移动到指定的位置

- margin-top设置负值
 - 会破坏文档流进行移动，移动完成之后会回到标准文档流之中
 - 应用：内容和前面部分重叠
- 定位
 - 是css提出一个比较重要的概念，可以让盒子按照指定的方向进行移动

标准文档流

- 在页面布局过程中，元素默认按照从上到下，从左到右，块级元素独占一行，行级元素共享一行的排列规范

脱离文档流

- 在页面布局过程中，元素不再遵循标准文档流的规范，按照自己的排列规范进行排列

破坏文档流

- 在页面布局过程中，元素在移动过程中不遵循标准文档流的规范，移动完成之后仍然在标准文档流之中

定位

静态定位

相对定位

绝对定位

固定定位

粘性定位

相对定位

相对定位

- 概念：元素参考原来的位置，按照指定的方向进行移动
- 语法：

```
position: relative;  
top: ;  
bottom: ;  
left: ;  
right: ;
```

- 定义偏移量：可以设置正值或负值
 - top：设置盒子与参考位置上边缘的距离
 - bottom：设置盒子与参考位置下边缘的距离
 - left：设置盒子与参考位置左边缘的距离
 - right：设置盒子与参考位置右边缘的距离
- 特点：

- 设置了相对定位的元素，不会脱离文档流，会破坏文档流进行移动
 - 原来的空间还在
 - 只设置相对定位的元素，元素没有任何变化，一旦设置了偏移量，元素参考**原来的位置**进行移动
- 注意：
 - 子元素设置相对定位，如果父元素进行了移动，子元素会跟着移动，原因在于：父元素进行移动后，子元素原来的位置也会跟着移动，而相对定位的元素是参加原来的位置进行移动的，所以也会跟着移动
 - 相对定位一般不会单独使用，一般会配合绝对定位一起使用

绝对定位

绝对定位

- 设置绝对定位的元素会默认参考整个文档进行移动，按照指定的方向进行移动
- 语法：

```
position: absolute;
```
- 设置偏移量：可以正值也可以负值
 - `top`：设置盒子与参考位置上边缘的距离
 - `bottom`：设置盒子与参考位置下边缘的距离
 - `left`：设置盒子与参考位置左边缘的距离
 - `right`：设置盒子与参考位置右边缘的距离
- 特点：
 - 绝对定位的元素会脱离文档流，在标准文档流之上
 - 原来的空间不存在
 - 只设置绝对定位的元素，只在当前位置脱离文档流，一旦设置偏移量，绝对定位的元素默认是参考整个文档进行移动或者参考最近的定位父级进行移动
- 注意：
 - 后面绝对定位的元素会盖住前面定位的元素
 - 行内元素设置绝对定位后，行内元素支持宽高

结构父级

- 标签在html代码结构中的父元素

定位父级

- 绝对定位的元素参考某个满足条件的父级标签进行移动，这个父元素就是定位父级，父级标签可以设置相对定位、绝对定位、固定定位作为定位父级
- 注意：
 - 可以设置相对定位、绝对定位、固定定位作为绝对定位的定位父级，但是一般会采用相对定位（子绝父相）
 - 多个定位父级时，绝对定位的元素是参考最近的定位父级进行移动
 - 定位父级可以父亲、爷爷、曾祖父.....

固定定位

固定定位

- 概念：固定定位的元素可以固定在页面上，不会随着滚动条而滚动
- 语法：

```
position:fixed;
```

- 设置偏移量：可以正值也可以负值
 - top：设置盒子与参考位置上边缘的距离
 - bottom：设置盒子与参考位置下边缘的距离
 - left：设置盒子与参考位置左边缘的距离
 - right：设置盒子与参考位置右边缘的距离
- 特点：
 - 固定定位的元素会脱离文档流，在标准文档流之上
 - 原来的位置不再占用
 - 只设置固定定位的元素，元素会在当前位置脱离文档流，会固定页面上，不会随着滚动条而滚动，一旦设置了偏移量，固定定位的元素会参考整个文档进行移动

z-index

- 设置定位元素（相对定位、绝对定位、固定定位）的层级关系，显示的顺序
- 语法：

```
z-index: 数字;
```

- 所有元素相当于默认为0，数字越大，定位层级越高，会显示在上方，数字相同时，按照代码结构显示顺序依次显示
- 注意：
 - 相对定位、绝对定位、固定定位的元素默认后面定位的元素会盖住前面定位的元素
 - z-index只针对与相对定位、绝对定位、固定定位的元素才有效

拓展：绝对定位元素宽度设置百分比

- 没有设置绝对定位的元素的宽度设置百分比时，宽度是参考结构父级的宽度
- 设置绝对定位的元素，宽度设置百分比时，绝对定位的元素的宽度默认是参考整个文档的宽度，如果设置了定位父级，参考最近的定位父级的宽度

粘性定位

粘性定位 `position: sticky` 可以被认为是相对定位(`position: relative`)和固定定位(`position: fixed`)的混合。元素在跨越特定阈值前为相对定位，之后为固定定位。例如：

```
.sticky-header {  
    position: sticky;  
    top: 10px;  
}
```

在视口滚动到元素 top 距离小于 10px 之前，元素为相对定位。之后，元素将固定在与顶部距离 10px 的位置，直到视口回滚到阈值以下。

粘性定位常作用在导航和概览信息(标题，表头，操作栏，底部评论等)上。这样，用户在浏览详细信息时，也能看到信息的概览和做一些操作，给用户带来便捷的使用体验。



粘性定位看着很简单，但也很容易出现不生效的情况。为帮助大家彻底理解粘性定位，本文会从 3 个方面来介绍：

- 粘性定位的原理。
- 不生效的情况。
- 具体的例子。

原理

为便于理解粘性定位，这里引入四个元素：视口元素，容器元素，粘性约束元素 和 sticky 元素。它们的关系如下：



视口元素：显示内容的区域。会设置宽，高。一般会设置 `overflow:hidden`。容器元素：离 sticky 元素最近的能滚动的祖先元素。粘性约束元素：粘性定位的父元素。有时，也会出现粘性约束元素就是容器元素的情况。sticky 元素：设置了 `position: sticky;` 的元素。

滚动时，sticky 元素设置的 left, right, top, bottom 的值相对的是容器元素。当粘性约束元素滚出视口时，sticky 元素也会滚出视口。

不生效的情况

情况1: 未指定 top, right, top 和 bottom 中的任何一个值

此时，设置 `position: sticky` 相当于设置 `position: relative`。

要生效，要指定 top, right, top 或 bottom 中的任何一个值。

情况2: 垂直滚动时，粘性约束元素高度小于等于 sticky 元素高度

不生效的原因：当粘性约束元素滚出视口时，sticky 元素也会滚出视口。粘性约束元素比 sticky 元素还小，sticky 元素没有显示固定定位状态的机会。

同样的，水平滚动时，粘性约束元素宽度小于等于 sticky 元素宽度时，也不会生效。

情况3: 粘性约束元素和容器元素之间存在 overflow: hidden 的元素

该情况的示例代码：

```
<div class="viewport">
  <div class="container"> <!-- 容器元素 -->
    <div style="overflow: hidden">
      <div> <!-- 粘性约束元素 -->
        <div class="stick-elem">...</div> <!-- sticky 元素 -->
        ...
      </div>
    </div>
  </div>
</div>
```

要生效，要把 `overflow: hidden` 的元素移除。

具体的例子

例子1: 页面滚动

这个例子，我们来实现页面滚动到文章内容时，文章标题吸顶的效果。



HTML 结构如下：

```
<div class="header">网站头部</div>
<!-- 粘性约束元素 -->
<div class="article">
    <!-- sticky 元素 -->
    <h2 class="title">彻底理解粘性定位 - position: sticky</h2>
    <div class="content">...</div>
</div>
<div class="footer">网站底部</div>
```

在这个例子中，视口元素和容器元素都是 `body`。sticky 元素是 `.title`，因此只要在 sticky 元素上设置如下样式即可：

```
.title {
    position: sticky;
    top: 0;
    background-color: #fff;
```

例子2: 文章列表

这个例子，我们来实现一块区域下有多篇文章，区域滚动到文章内容时，对应的标题和操作按钮吸顶的效果。



HTML 结构如下：

```
<!-- 视口元素 -->
<div class="viewport">
    <!-- 容器元素 -->
    <div class="container">
        <!-- 文章:粘性约束元素 -->
        <div class="article">
            <div class="sticky-header">
                <h2>彻底理解粘性定位 - position: sticky</h2>
                <div class="options">
                    <button>转发</button>
                    <button>点赞</button>
                </div>
            </div>
            <div class="article-content">...</div>
        </div>
        <!-- 文章 -->
        <div class="article">...</div>
        <div class="article">...</div>
    </div>
```

在这个例子中，视口元素是 `.viewport`，容器元素是 `.container`。sticky 元素是 `.sticky-header`。核心样式如下：

```
/* 视口元素 */
.viewport {
    width: 50%;
    overflow: hidden;
    height: 200px;
}

/* 容器元素 */
.container {
    overflow: auto;
    height: 100%;
}

/* 粘性约束元素 */
.article {
    margin-bottom: 10px;
}

/* sticky 元素 */
.sticky-header {
    position: sticky;
    top: 0;
    padding: 5px 0;
    background-color:#fff;
}
```

例子3：甘特图

最后，我们来做个复杂点的例子：甘特图。如下图所示：



需要实现：

- 左侧事项菜单总在最左侧。
- 菜单的头部和时间轴吸顶。
- 时间轴的年总在月的最左边。

实现代码有点多，就不在这里贴了。

BFC基础

FC(Formatting Context)：格式化上下文。它是css 2.1提出一个视觉渲染的概念。

它是页面中一块渲染区域，并且有一套渲染规则，它决定了其子元素如何排列，和其他的元素相关之间的关系。

BFC和IFC是常见的FC。分别叫做 `Block Formatting Context` 和 `Inline Formatting Context`。

BFC

Block Formatting Context：块级格式化上下文

它是指页面上的一块渲染区域，它有自己的渲染规则，它其实就是一个标签，标签不一定是块级标签，它只有满足条件，它就是一个BFC区域，决定了BFC的内部的HTML标签如何进行排列以及不同的BFC区域如何进行显示。

BFC区域的特点



1. 内部的Box会在垂直方向，一个接一个地放置。 (标准文档流)
2. Box垂直方向的距离由margin决定。属于同一个BFC的两个相邻Box的margin会发生重叠
 - 如果说不想让垂直方向上的margin发生重叠，我们可以形成一个新的BFC区域
3. 每个元素的左外边缘 (margin-left)，与包含块的左边 (contain box left) 相接触(对于从左往右的格式化，否则相反)。即使存在浮动也是如此。除非这个元素自己形成了一个新的BFC。
4. BFC的区域不会与float box重叠。
 - 利于这点，可以实现非浮动元素 (BFC) 和浮动元素同行显示，可以实现三列布局
5. BFC就是页面上的一个隔离的独立容器，容器里面的子元素不会影响到外面的元素。反之也如此。
6. 计算BFC的高度时，浮动元素也参与计算
 - 可以解决子元素浮动父元素高度塌陷的问题

如何成为BFC容器（如何升级为BFC）

1. 根标签 (html标签就是一个BFC容器)
2. float 不为 none
3. position 为 absolute 或 fixed
4. display 为 inline-block、table-cell、table-caption、flex (弹性盒子)
5. overflow 不为 visible;

BFC应用场景

应用场景

场景一：

每个元素的左外边缘 (margin-left)，与包含块的左边 (contain box left) 相接触(对于从左往右的格式化，否则相反)。即使存在浮动也是如此。除非这个元素形成一个新的BFC区域

场景二：

BFC的区域不会与float box重叠。

- 实现三列自适应布局：左右两边固定宽度，中间自适应

```
ss:  
<style>  
/* BFC的区域不会与float box重叠。 */  
.container{  
    width: 100%;  
    height: 500px;  
    border: 1px solid red;  
}  
.left{  
    width: 200px;  
    height: 200px;  
    background-color: pink;  
    float: left;  
}  
.right{  
    width: 200px;
```

```

        height: 200px;
        background-color: tomato;
        float: right;
    }
    .center{
        height: 400px;
        background-color: orange;
        /* 设置为BFC */
        overflow: hidden;
    }

```

</style>

html:

```

<div class="container">
    <div class="left"></div>
    <div class="right"></div>
    <div class="center">center</div>
</div>

```

场景三:

Box垂直方向的距离由margin决定。属于同一个BFC的两个相邻Box的margin会发生重叠。

- margin重叠性问题解决方案：将其中一个盒子放进一个新的BFC容器中

```

html:
<div class="box1"></div>
<!-- 形成一个新的BFC区域 -->
<div class="container">
    <div class="box2"></div>
</div>
css:
<style>
    /* Box垂直方向的距离由margin决定。属于同一个BFC的两个相邻Box的margin会发生重叠 */
    /* 属于不同的BFC区域就不会发生margin的重叠 */
    .box1{
        width: 100px;
        height: 100px;
        background-color: pink;
        margin-bottom: 50px;
    }
    .box2{
        width: 100px;
        height: 100px;
        background-color: tomato;
        margin-top: 80px;
    }
    .container{
        /* 设置为BFC容器 */
        /* overflow: hidden; */
        /* float: left; */
        /* position: absolute; */
        position: fixed;
    }

```

</style>

场景四:

计算BFC的高度时，浮动元素也参与计算

- 用来解决子元素浮动，父元素高度塌陷

IFC基础

IFC

IFC(Inline Formatting Context): 行内格式化上下文，和块级格式上下文一样，都是页面进行css渲染时一个视觉概念。

指的是一行区域的渲染规则，确定了一行中行级元素如何进行排列以及对齐。

IFC区域的特点

- 在IFC中，盒子水平放置，一个接着一个，从包含块的顶部开始
- 在盒子间margin和padding在水平方向有效
 - 行内元素在垂直方向上的margin和padding是没有效果
- 这些盒子会通过不同的方式进行对齐，有些底部和顶部对齐，底部和文本的基线进行对齐等等
- IFC负责的矩形区域称为行盒 (line-box)
 - 行盒的宽度由内容决定
 - 行盒的高度由line-box的最高点和最低点决定

主要影响行盒高度的css属性 (影响行盒的布局)

1. font-size：字体大小会影响行盒高度
2. font-family:字体类型会影响行盒的高度
3. height | line-height: 高度和行高也会影响行盒的高度
4. vertical-align：设置垂直方向上的对齐的方式，也会影响行盒的高度

IFC应用场景

主要影响行盒高度的css属性 (影响行盒的布局)

1. font-size：字体大小会影响行盒高度
2. font-family:字体类型会影响行盒的高度
3. height | line-height: 高度和行高也会影响行盒的高度
4. vertical-align：设置垂直方向上的对齐的方式，也会影响行盒的高度

font-size

- 不同的字体大小会影响行盒的高度
- 字体越大，行盒的高度越高

font-family

- 不同的字体类型会影响行盒的高度
- 原因：设计师在设计字体时，不同的字体类型占据的高度时不一样

height | line-height

- 不同的高度和行高会影响行盒的高度
- 行高是指文本的最顶部到文本最底部，文字在当前的行高中垂直居中

img

vertical-align

- 设置行级元素垂直方向上对齐方式，也会影响行盒的高度

img

绿色：顶线

蓝色：中线

红色：基线

紫色：底线

- 取值：

- baseline：默认值，基线对齐
- middle：中线对齐
- top：顶部对齐
- bottom：底部对齐



文本之间

- 文本与文本之间对齐方式，文本默认是以基线与其他元素对齐
- 设置vertical-align控制当前标签的文本的参考位置

文本与图片之间

- 文本默认是以基线和图片的底部进行对齐
- 给文本设置vertical-align：文本是以参考位置和图片进行对齐
- 给图片设置vertical-align：图片是以参考位置和文本进行对齐

表格

- 设置单元格中内容垂直方向上的对齐方式
- 取值：

- middle：默认值，居中对齐
- top：顶部对齐
- bottom：底部对齐

H5标签

H5媒体标签

音频 audio

- 用于在页面中引入音频文件
- 语法：

写法一：

```
<audio src="连接音频文件的文件路径" controls loop muted autoplay></audio>
```

写法二：为了保证不同的浏览器都可以正常播放音频文件，需要引入多种文件格式的音频文件

```
<audio controls loop muted autoplay>
    <!-- 资源标签，用于引入多媒体的资源 -->
    <source src="连接音频文件格式1的文件路径" type="audio/mp3">
    <source src="连接音频文件格式2的文件路径" type="audio/ogg">
</audio>
```

- `controls`：这是一个控制器，控制音频文件的播放暂停等等
- `loop`：循环播放，默认音频只播放一次
- `muted`：静音
- `autoplay`：自动播放

视频 video

- 用于在页面中引入视频文件
- 语法：

写法一：

```
<video src="连接视频文件的文件路径" controls loop muted autoplay></video>
```

写法二：为了保证不同浏览器都能正常播放音频文件，需要引入多种文件格式的视频文件

```
<video controls loop muted autoplay>
  <source src="连接视频文件格式1的文件路径">
  <source src="连接视频文件格式2的文件路径">
</video>
```

- `controls`：这是一个控制器，控制视频文件的播放暂停等等
- `loop`：循环播放，默认视频只播放一次
- `muted`：静音播放
- `autoplay`：自动播放

H5新增控件

H5表单标签

form、input (text) 、input (password) 、input (radio) 、input (checkbox) 、select、textarea

按钮：input和button

邮箱输入框

- 在提交数据时提供了邮箱格式的验证，验证不准确，以后通过js进行验证
- 语法：

```
<input type="email">
```

数字输入框

- 只能输入数字，提供了数字的加减操作
- 语法：

```
<input type="number">
```

网址输入框

- 可以进行网址格式的验证，必须是完整的网址，是以http或者https开头的网址
- 语法：

```
<input type="url">
```

搜索框

- 在页面中加入搜索框，提供了清空内容的操作
- 语法：

```
<input type="search">
```

选取颜色

- 可以在页面中选择颜色色值
- 语法：

```
<input type="color">
```

选取范围

- 语法：

```
<input type="range">
```

文件上传

- 提供文件上传的功能
- 语法：

```
<input type="file" multiple>
```

- 注意：默认只能上传一个文件，如果需要上传多个文件，需要添加multiple属性

时间控件

- 提供时间的选择
- 语法：

```
<!-- 某年某月某日 -->
<input type="date">
<!-- 某年某月 -->
<input type="month">
<!-- 某年第几周 -->
<input type="week">
<!-- 时/分 -->
<input type="time">
```

选项列表

- 当在输入框中输入内容时，下方的提示相关开头的内容
- 语法：

```
<input type="text" list="datalist的id名">
<datalist id="id名">
    <option value="">小猪佩奇</option>
    <option value="">小猪乔治</option>
    <option value="">小米手机</option>
    <option value="">小米电脑</option>
    <option value="">大米先生</option>
    <option value="">大米小姐</option>
</datalist>
```

- 注意：输入框需要设置list属性指定datalist的id名进行绑定

表单属性

- `required`：必填项，必须填写才可以提交
- `autofocus`：自动获取焦点
- `readonly`：只读，不能修改，可以复制，可以提交
- `disabled`：禁用，不能修改，可以复制，不可以提交

语义化标签

语义化标签

- 通过标签名可以合理正确的表达标签中内容的含义

语义化标签的好处

- 易于用户阅读，样式丢失的时候让页面结构更加清晰明了
- 有利于SEO优化，搜索引擎可以根据根据标签名确定上下文和各个关键字之间的关系
- 方便其他设备的解析，比如盲人阅读器
- 有利于开发和维护，语义化具有可读性，代码更加好维护

常用的语义化标签

标签名	描述
<header>	代表网页的头部
<main>	代表网页的主体内容，页面上有且仅有一个
<footer>	代表网页的尾部
<nav>	代表导航
<aside>	代表侧边栏，可以放侧边导航或者推荐信息等等
<article>	代表一个独立的区域，存放内容
<section>	代表网页中一个模块
<thead>	代表表格的头部
<tbody>	代表表格的内容
<tfoot>	代表表格的尾部
<audio>	音频
<video>	视频
<source>	资源标签，用于引入多媒体资源
<canvas>	定义形状，在里面绘制图形

网页中标签的选择

- 最外层的标签尽量使用语义化标签，比如header、main、footer等等
- 标题尽量使用标题标签，主要为了SEO优化
- 对于网页中的内容区域，该用什么标签就用什么标签，超链接使用a标签，图片使用img等等

CSS兼容性问题处理

兼容性来源

内核是浏览器最底层、最核心的代码。决定了网页如何进行解析的，页面是如何被加载的，脚本如何执行。

不同的浏览器的内核是不一样，每种浏览器对相同的页面有不同的解析方式，最终的效果不一样，将这种情况称为兼容性问题。

同一个页面在不同的浏览器上运行的结果是不一样，就是兼容性。

主流的浏览器：Chrome、firfox、IE、safira

国产浏览器：QQ浏览器、360浏览器，UC浏览器，猎豹等等

目前国内是没有自己的内核，都用的国外的内核。360支持双核浏览器：IE Trident 和 safira 的 webkit内核一起集成。

浏览器的内核

作用：用于解析网页

解析网页：

- html内容解析
- css内容解析
- JavaScript内容解析

内核中有个比较重要的概念：渲染引擎

渲染引擎

主要包含一些几部分：

- HTML解析器：会将html代码解析成一棵DOM数，每个标签就是dom树的一个节点
- CSS解析器：会将css样式计算出来，内存会跟着一起工作，会将每个标签上的最终样式计算出来
- JavaScript引擎：允许js脚本运行环境
- 布局：主要网页中模块定位、排列、浮动等等



网页是如何解析出来？



1. 加载网页内容
 - 判断是否为网络资源，是就利用网络模块从网页中加载代码
 - 如果是本地的资源，就直接从本地加载网页文件
2. 经过HTML解释器，对代码进行解析
 - 浏览器从代码的第一行进行解析，把不同类型的代码会交给对应的解释器进行解析
 - css代码——css解释器
 - html代码——html解释器
 - js代码——JavaScript引擎
3. 将各个解释器的结果进行综合梳理（内部表示）
 - 会把标签和对应的css样式结合起来，每个标签都有自己的渲染结果
4. 布局和绘图
 - 会把每个标签的样式、位置等绘制在页面上，如果有图片或视频音频，再利用对应的模块进行处理，处理完成之后直接显示在页面上。

兼容性的处理方案

hack代码

专门针对特点的浏览器设置的css代码

css hack：针对不同的浏览器或浏览器不同的版本写不同的css代码，书写css代码的这个过程就是css hack。

主要学习IE hack代码。

css hack分类

属性前缀法

- 针对不同的浏览器，在css属性的前面添加一些特点的浏览器的版本才能识别的前缀代码
- 多个属性需要设置时，每个属性前面都添加前缀



选择器前缀法

- 在选择器前面添加一些特定浏览器才能识别的代码



条件注释法

- 在引入css文件时规定作业的浏览器版本或类型

gte 大于等于 gt 大于 lte 小于等于 lt 小于 ! 不是

只在IE下生效

<!--[if IE]>

这段文字只在IE浏览器显示

<![endif]-->

只在IE6下生效

<!--[if IE 6]>

这段文字只在IE6浏览器显示

<![endif]-->

只在IE6以上版本生效

<!--[if gte IE 6]>

这段文字只在IE6以上(包括)版本IE浏览器显示

<![endif]-->

只在IE8上不生效

<!--[if ! IE 8]>

这段文字在非IE8浏览器显示

<![endif]-->

非IE浏览器生效

<!--[if !IE]>

这段文字只在非IE浏览器显示

<![endif]-->

兼容性自查网址

[Can I use... Support tables for HTML5, CSS3, etc](#)

css兼容指导思想

渐进增强

- 先考虑大多数浏览器能够正常显示网页，针对高版本的浏览器单独设置样式，添加页面效果，浏览器的版本越高，页面效果越好

优雅降级（推荐）

- 先不管兼容问题，直接以目前主流的浏览器为基础，实现最佳效果，然后再考虑低版本的浏览器书写css hack代码来保证低版本的浏览器能够正常显示

圆角原理

圆角

- CSS3提出的，盒子默认四个角都是直角，通过border-radius设置圆角的效果
- 语法：

分别设置四个角的元素：

`border-top-left-radius`
`border-top-right-radius`
`border-bottom-left-radius`
`border-bottom-right-radius`

复合属性：

`border-radius`

一个值：左上右上右下左下

两个值：左上右下 右上左下

三个值：左上 右上左下 右下

四个值：左上 右上 右下 左下

标准写法：八个值 /前面代表每个角的水平半径，/后面代表每个角的垂直半径

`border-radius: 30px 30px 30px 30px / 60px 60px 60px 60px;`

- 圆角的形成：一个椭圆是由水平半径和垂直半径构成，正圆是特殊的椭圆，圆角是一个椭圆或正圆的一段圆弧构成。
- 如果水平半径和垂直半径相等时，圆角取得是正圆的圆弧，如果不相等，取得是椭圆的圆弧



内半径和外半径

- 当边框足够宽时，设置border-radius小于边框的宽度时，边框内边缘没有圆角，是直角，外边缘是圆角
- 当边框足够宽时，设置border-radius大于边框的宽度时，边框外边缘的圆角的半径就是外半径，边框内边缘的圆角的半径就是内半径
- 内半径=外半径-边框的宽度
 - 外半径就是设置border-radius的值
- 注意：
 - 当border-radius小于或等于边框的宽度时，内半径=外半径-边框，内半径为负或0，直接为0，显示为直角
 - 当border-radius大于边框的宽度，内半径=外半径-边框，内半径为正，边框内边缘就是圆角

弹性布局 (弹性盒模型)

传统的布局

- 基于盒模型，依赖display+float+position进行页面布局
- 这种方式兼容性比较好，但是布局比较繁琐
- 各种不便：
 - 各种盒子居中问题
 - 盒子之间的间距margin调整

- 块级元素同行显示设置浮动——清除浮动

针对这些不便，css3提出一种自适应的布局方式——弹性布局，它用来替代或辅助传统布局

弹性盒模型

- 概念：是一种当页面需要适应不同的屏幕大小时，确保元素拥有恰当行为的布局方式。
- 目的：提供了一种布局方式，这种布局方式可以进行元素排列、空白空间分配等等

弹性盒模型结构



弹性盒子是由弹性容器和弹性项目构成。

- 弹性容器：包含弹性项目的父元素。
- 弹性项目：弹性容器里面每个子元素都是弹性项目。
- 主轴：弹性项目一行显示的方向就是主轴的方向
- 侧轴：与主轴垂直的方向就是侧轴的方向

给父元素（弹性容器）设置的属性

display:flex | inline-flex

- 设置盒子为弹性容器，里面子元素就是弹性项目
- `display:flex;`：设置块级弹性容器，对其他的兄弟元素来说，它就是一个普通的块级元素，对其子元素来说，它就是一个弹性容器，子元素按照弹性项目的方式进行显示
- `display:inline-flex;`：设置行级弹性容器，对其他的兄弟元素来说，它就是一个普通的行级元素，对其子元素来说，它就是一个弹性容器，子元素按照弹性项目的方式进行显示

弹性盒子特点

- 弹性容器里面的弹性项目可以同行显示，显示不下时按比例压缩显示
- 弹性容器没有设置高度时，高度可以自适应
- 弹性容器对兄弟元素没有影响
- 弹性容器对直接子元素有影响

flex-wrap

- 设置弹性项目是否可以换行
- 语法：

```
flex-wrap: nowrap | wrap;
```

- `nowrap`：默认值，不换行
- `wrap`：换行

flex-direction

- 设置弹性容器中主轴的方向和弹性项目的排列方式
- 主轴：弹性项目同行显示的方向 侧轴：与主轴垂直的方向就是侧轴的方向
- 取值：
 - `row`：默认值，设置水平方向为主轴，弹性项目从左到右依次排列
 - `row-reverse`：设置水平方向为主轴，弹性项目从右到左依次排列
 - `column`：设置垂直方向为主轴，弹性项目从上到下依次排列
 - `column-reverse`：设置垂直方向为主轴，弹性项目从下到上依次排列

富裕空间

给父元素（弹性容器）设置的属性

justify-content

- 处理主轴上的富裕空间的分配
- 取值：
 - `flex-start`：默认值，将主轴上的富裕空间放在弹性项目之后
 - `flex-end`：将主轴上的富裕空间放在弹性项目之前
 - `center`：将弹性项目在主轴上居中
 - `space-between`：首尾弹性项目紧靠弹性容器，中间均分
 - `space-around`：将主轴上的富裕空间均分到每个弹性项目两边
 - `space-evenly`：每个弹性项目之间的间距均分，包括首尾和弹性容器的间距

align-items

- 处理侧轴上的富裕空间的分配
- 取值：
 - `flex-start`：默认值，当弹性项目设置了高度时，将侧轴上的富裕空间均分到每行弹性项目之后
 - `flex-end`：将侧轴上的富裕空间均分到每行弹性项目之前
 - `center`：每行弹性项目上下均分富裕空间，在每行弹性项目居中
 - `stretch`：默认值，当弹性项目没有设置高度时，默认将弹性项目拉伸撑满弹性容器

给子元素（弹性项目）设置的属性

align-self

- 设置当前的弹性项目在侧轴上的富裕空间分配
- 取值：
 - `flex-start`：默认值，当弹性项目设置了高度时，将侧轴上的富裕空间均分到每行弹性项目之后
 - `flex-end`：将侧轴上的富裕空间均分到每行弹性项目之前
 - `center`：每行弹性项目上下均分富裕空间，在每行弹性项目居中
 - `stretch`：默认值，当弹性项目没有设置高度时，默认将弹性项目拉伸撑满弹性容器

富裕空间

- 概念：在弹性容器中除弹性项目之外的剩余空间，就称为富裕空间。主要分为主轴上的富裕空间和侧轴上的富裕空间
- 分配：
 - 主轴上的富裕空间分配：justify-content
 - 侧轴上的富裕空间分配：align-items 和 align-self

弹性空间

给子元素（弹性项目）设置的属性

flex-shrink

- 设置弹性项目的压缩因子，压缩比例
- 原理：当弹性容器宽度不够时并且弹性项目在一行显示不换行，为了保证不换行，通过 `flex-shrink` 属性将弹性项目进行按比例压缩显示，以达到不换行的目的。
- 语法：

```
flex-shrink: 数字;
```

- 默认值为1，数字越大，压缩比例越大，盒子的大小越小
- 计算公式：

总压缩空间 = 弹性项目总宽度 - 弹性容器的宽度

每个弹性项目的压缩空间 = 总压缩空间 / 压缩因子总数 * 弹性项目的压缩因子

盒子的大小 = 盒子原来的宽度 - 压缩空间宽度

flex-grow

- 设置弹性项目的弹性因子
- 语法：

```
flex-grow: 数字;
```

- 默认值为0；数字越大，弹性空间越大，盒子的大小越大
- 计算公式

富裕空间 = 弹性容器的宽度 - 弹性项目的总宽度

每个弹性项目的弹性空间 = 富裕空间 / 弹性因子总数 * 弹性项目的弹性因子

盒子的大小 = 盒子原来的宽度 + 弹性项目的弹性空间

弹性空间

- 概念：弹性项目消化掉的富裕空间，
- 目的：设置弹性因子，弹性项目可以撑满弹性容器，消化掉主轴上的富裕空间，通过 `flex-grow` 属性可以设置每个弹性项目消化空间的比例

弹性扩展属性

给子元素（弹性项目）设置的属性

order

- 能够更改某个弹性项目的显示的位置
- 语法：

```
order: 数字;
```

- 默认值为0，可以设置负值，数字越大，元素显示位置越靠后，数字相同时，按照代码顺序依次显示
- 应用场景：
 - js控制拖拽或点击让元素排列到最后

flex-basis

- 设置弹性项目的默认的宽度
- 语法：

```
flex-basis: 300px;
```

- 注意：
 - 优先级：flex-basis > width > 由内容撑开的

flex

- 复合属性，是flex-grow、flex-shrink、flex-basis的复合属性
- 语法：

```
flex: flex-grow flex-shrink flex-basis;
```

- 默认值：0 1 auto

线性渐变 (CSS3基础)

渐变背景

像比较规则的渐变背景，使用1px的渐变背景的平铺，像不规则的渐变背景，则直接使用整张图片作为背景，图片开发维护性比较低，影响网页加载速率。

css3提出了渐变背景：线性渐变和径向渐变

渐变背景可以看作一张“图像”，使用css属性中的 `background-image`

线性渐变

- 概念：线性渐变可以看作沿着一条直线进行颜色渐变。
- 注意：渐变背景至少设置2种颜色
- 语法：

```
background-image: linear-gradient( 渐变方向 , 颜色1  位置1, 颜色2  位置2, 颜色2  位置2)
```

色标

- 是由一个颜色和一个位置构成的。控制渐变的范围
- 颜色：单词，#十六进制，rgb(),rgba(0-255, 0-255, 0-255, 0.5)
- 位置：像素，百分比
 - 至设置颜色时，渐变默认为均匀渐变
 - 如果首尾色标的颜色不在0%或100%时，是以纯色进行填充

```
0%~20%是红色，20%~60%是红色到绿色的渐变，60%~80%是绿色到黄色的渐变，  
80%~100%是黄色  
background-image: linear-gradient(red 20%, green 60%, yellow 80%);
```

- 如果首尾色标的位置不设置时，默认0%或100%
- 如果两个色标的位罝相同时，中间的渐变的范围为0，会出现断层的效果。

渐变线

- 可以用来控制渐变方向
- to 结束方向
 - to bottom ——默认值，180deg或者-180deg
 - to top——0deg
 - to left——270deg 或-90deg
 - to right——90deg 或-270deg
 - to top left——315deg 或-45deg
 - to top right——45deg或-315deg
 - to bottom left——225deg或-135deg
 - to bottom right——135deg 或 -225deg
- 角度：必须带deg的单位，0deg的方向是垂直向上，顺时针是正方向，逆时针是负方向



重复线性渐变

- 将线性渐变进行重复铺设，使用 repeating-linear-gradient
- 注意：只有首尾不在0%或100%时，重复线性渐变才有效
- 语法：

```
background-image: repeating-linear-gradient(pink 0px, pink 20px, yellow  
20px, yellow 40px);
```

径向渐变

径向渐变

- 概念：径向渐变指的就是椭圆渐变，渐变效果就是沿着椭圆的半径方向进行渐变，正圆就是一种特殊的椭圆。
- 分为两部分：
 - 色标：是由一个颜色和一个位置构成，用来控制渐变过渡的颜色变化范围（参考线性渐变的用法）
 - 椭圆：用来控制径向渐变位置、大小、形状。
- 语法：

```
background-image: radial-gradient(大小 形状 at x轴 y轴, 颜色1 位置  
1, 颜色2 位置2, 颜色3 位置3);
```

- 大小：
 - farthest-corner：默认值，半径从圆心到最远的角
 - closest-corner：半径从圆心到最近的角
 - farthest-side：半径从圆心到最远的边

- `closest-side`: 半径从圆心到最近的边
- 形状:
 - `ellipse`: 默认值, 椭圆
 - `circle`: 正圆
- 圆心: at x轴 y轴

默认在盒子的中心点, 相当于center center

 - 单词: top bottom left right center 两两搭配使用
 - 像素: 0px 0px 相当于盒子的左上角
 - 百分比: 参考盒子的宽度和高度, 50% 50% 相当于center center

重复径向渐变

- 将径向渐变进行重复铺设
- 注意: 渐变首尾不在0% 或100%时, 重复径向渐变才有效

盒子阴影

盒子阴影

- 给盒子设置阴影, 通过box-shadow设置盒子阴影
- 语法:

```
box-shadow: x轴偏移量 y轴偏移量 模糊程度 阴影大小 颜色;
inset将外阴影设置为内阴影
box-shadow: x轴偏移量 y轴偏移量 模糊程度 阴影大小 颜色 inset;
可以设置多层阴影, 中间使用逗号隔开
box-shadow: x轴偏移量 y轴偏移量 模糊程度 阴影大小 颜色 , x轴偏移量 y轴偏
移量 模糊程度 阴影大小 颜色 , x轴偏移量 y轴偏移量 模糊程度 阴影大小 颜色;
```

- 注意: 模糊程度和阴影大小可以不写

文字阴影

- 给文本设置阴影, 通过text-shadow设置阴影
- 语法:

```
text-shadow: x轴偏移量 y轴偏移量 模糊程度 颜色;
```

结构选择器

结构选择器

- 根据HTML代码结构找到满足条件的标签, 也是一种伪类选择器

`:first-child`

- 找到满足条件的标签, 并且这个标签是某个标签的第一个子标签
- 语法:

找到class名为box的标签里面所有的后代span标签，并且这个span标签是某个标签的第一个子标签

```
.box span:first-child{  
}
```

:last-child

- 找到满足条件的标签，并且这个标签是某个标签的倒数第一个子标签（最后一个子标签）

- 语法：

找到class名为box的标签里面所有的后代span标签，并且这个span标签是某个标签的倒数第一个子标签

```
.box span:last-child{  
}
```

:nth-child(n)

- 找到满足条件的标签，并且这个标签是某个标签的第n个子标签

- 语法：

找到class名为box的标签里面所有的后代span标签，并且这个span标签是某个标签的第n个子标签

```
.box span:nth-child(n){  
}
```

■ n代表数字

- 注意：

- :nth-child(2n-1) 或 :nth-child(2n+1)：找到奇数的子标签
- :nth-child(2n)：找到偶数的子标签

:nth-last-child(n)

- 找到满足条件的标签，并且这个标签是某个标签的倒数第n个子标签

- 语法：

找到class名为box的标签里面所有的后代span标签，并且这个span标签是某个标签的倒数第n个子标签

```
.box span:nth-last-child(n){  
}
```

■ n代表数字

:nth-of-type(n)

- 将满足条件的同类型的标签，将其筛选出来，重新排序，找到其中的第n个标签

- 语法

找到class名为box的标签里面所有的后代span标签，将span标签筛选出来，重新排序，找到其中的第n个标签

```
.box span:nth-of-type(n){  
}
```

:nth-last-of-type(n)

- 将满足条件的同类型的标签，将其筛选出来，重新排序，找到其中的倒数第n个标签
- 语法

找到class名为box的标签里面所有的后代span标签，将span标签筛选出来，重新排序，找到其中的倒数第n个标签

```
.box span:nth-last-of-type(n){  
}
```

SASS基础

原生css的不便

- 书写重复的选择器
- 修改属性，每次都需要修改，不能一处改处处改
- css代码重复编写

SASS

- 概念：less和sass都是一个css预处理器，可以为网站启用可定义、可管理和可重用的样式表，就是动态的样式表语言。
 - css预处理器其实是一种脚本语言，可以扩展css语法并将其编译成常规的css代码，方便的浏览器正常渲染。
 - sass是css的扩展，通过提供嵌套、变量、混合等操作，通过编译都可以变成常规的css代码
- 作用：极大的提高开发者开发的效率
- 官网：[Sass世界上最成熟、稳定和强大的CSS扩展语言 | Sass中文网](#)

SASS基本使用

使用vs code的easy sass插件来完成sass文件的编写

- 按照easy sass插件
- 新建以.sass或.scss为后缀的文件，在里面书写css代码
 - .sass：是老版本的sass文件
 - .scss是新版本的sass文件（推荐）
- easy sass插件可以将sass文件编译为css文件，在页面中引入css文件即可。

手动更改编译后的css文件存储路径

- 找到设置，搜索easy sass
- 注意：存储路径是从工作区出发，工作区必须有名字



嵌套

嵌套

- 概念：SASS支持选择器里面嵌套子选择器
- 作用：让sass代码对应css代码层级结构清晰明了，不会出现父子标签之间权重
- 语法：

```
父选择器{
    父选择器的css代码;
    css属性: css属性值;
    子选择器{
        子选择器的css代码
        子选择器{
            子选择器的css代码
        }
    }
}

示例:
.header{
    width: 100%;
    height: 70px;
    background-color: pink;
    .container{
        width: 1200px;
        height: 70px;
        border: 1px solid red;
        margin: 0 auto;
        .logo{
            width: 200px;
            height: 70px;
            background-color: tomato;
        }
    }
    .nav{
        width: 400px;
        height: 70px;
        background-color: yellowgreen;
        >ul{
            list-style: none;
            li{
                background-color: red;
                // &代表父选择器
                &:hover{
                    background-color: blue;
                }
                &::before{
                    content: "";
                }
                &::after{
                    content: "after";
                }
            }
        }
        +.box{
            border: 1px solid red;
        }
    ~div{
        width: 300px;
    }
}
```

```
        height: 200px;
    }
}
}
}
```

变量

变量

- 概念：变量可以看作一个保存数据的容器，可以在代码中重复使用
- 作用：可以实现一处改处处改
- 语法：

```
定义变量：  
$变量名: 数据;  
使用变量：  
.box{  
    color: $变量名;  
}
```

- 数据可以是任意的属性值，比如：100px, red，也可以复合属性的属性值，比如：1px solid red
- 变量名规范：
 - 是以数字、字母、_和-构成
 - 不能以数字开头，不能包含特殊的符号
 - 多个单词之间尽量使用-分开
- 注意：
 - 变量需要先在文件的开头定义变量，再使用变量
 - sass语法中提供数学运算，符号前后加空格

CSS过渡

基本概念

- `transition` 提供了一种在更改css属性时控制动画执行的速率或者时间，可以在指定的时间内执行动画效果，而不是立即执行。

属性

`transition-property`

- 设置过渡的css属性名
- 语法：

```
transition-property: css属性名;
```

- 默认值为all，代表所有的css属性
- 注意：

- 仅仅设置过渡的属性名没有效果，需要配合过渡的执行时长一起使用
- 多个属性同时设置过渡时，中间使用逗号隔开，时间和属性名一一对应

transition-duration

- 设置过渡的执行时长，控制整个过渡的动画在多长时间内执行完成
- 语法：

```
transition-duration: 时间;
```

- 默认为0s，s代表秒，ms代码毫秒，1s=1000ms
- 注意：
 - 如果过渡的属性名的个数多于执行时长的个数，时间一一对应，时间重复一遍

transition-timing-function

- 设置过渡的执行的速率
- 属性值：
 - `ease`：默认值，以低速开始，然后变快，在结束前变慢
 - `linear`：匀速
 - `ease-in`：以低速开始
 - `ease-out`：以低速结束
 - `ease-in-out`：以低速开始和结束
 - `cubic-bezier(x1,y1,x2,y2)`：贝塞尔曲线

贝塞尔曲线

- 概念：是一种构建二维图形的速度曲线，本质上是一个数学曲线
- 作用：在css中，可以用来控制动画的执行速率
- 语法：

```
cubic-bezier(x1,y1,x2,y2)
```

- 这两个坐标决定了曲线形状，不同的形状对应的速率不一样，可以为负
- 构建贝塞尔曲线的网址：cubic-bezier.com

transition-delay

- 设置过渡的延迟时长，就是过渡的效果从何时开始执行
- 语法：

```
transition-delay: 时间;
```

- 默认值为0s，

transition

- 复合属性
- 语法：

```
transition: 过渡属性名 执行时长 延迟时长 执行速率;
多个属性同时过渡时，中间使用逗号隔开
transition: 过渡属性名1 执行时长 延迟时长 执行速率 , 过渡属性名2 执行时长
延迟时长 执行速率 , 过渡属性名3 执行时长 延迟时长 执行速率;
```

- 注意：并不是所有的css属性都可以设置过渡，比如display

2D转换

基本概念

- CSS3 提出了一个 `transform` 属性来处理盒子（html标签）的变化或转换，包含移动、旋转、缩放和倾斜。
- 2D转换一般配合过渡的一起使用
- 注意：2D转换不会影响其他盒子的排列

位移 translate

- 让盒子进行移动
- 语法：

```
transform: translate(x轴移动量 , y轴移动量);
transform: translate(x轴移动量);
沿着x轴进行移动
transform: translateX(x轴移动量);
沿着y轴进行移动
transform: translateY(y轴移动量);
```

- 盒子是以原来的位置为起点进行移动

旋转 rotate

- 可以让盒子进行旋转
- 语法：

```
围绕着盒子的中心点进行旋转
transform: rotate(角度);
围绕着x轴进行旋转
transform: rotateX(角度);
围绕着y轴进行旋转
transform: rotateY(角度);
```

- `rotate(角度)`：围绕着盒子的中心点进行旋转，设置正值，顺时针旋转，设置负值，逆时针旋转

缩放 scale

- 设置盒子进行放大或缩小
- 语法：

```
transform: scale(宽度的倍数, 高度的倍数);  
等比例缩放  
transform: scale(倍数);
```

- 0: 标签会缩放为原来的0倍, 标签会从页面上消失不见
- 0~1: 盒子缩小
- 1: 盒子原来的大小
- 1以上: 盒子放大
- 注意: 默认缩放会参考盒子的中心点进行缩放

倾斜 skew

- 让盒子沿着x轴或y轴进行倾斜
- 语法:

```
沿着x轴进行倾斜  
transform: skew(角度);  
transform: skewX(角度);  
沿着y轴进行倾斜  
transform: skewY(角度);
```

- `skewX(角度)`: 沿着x轴进行倾斜, 角度越大, 越接近x轴, 当角度为90deg时, 会和x轴平行, 消失不见
- `skewY(角度)`: 沿着Y轴进行倾斜, 角度越大, 越接近y轴, 当角度为90deg时, 会和y轴平行, 消失不见

组合变换

组合变换

- 多个2D转换组合在一起, 作用在同一个标签上, 中间使用空格隔开
- 语法:

```
transform: rotate(90deg) translateX(200px) scale(0.5);
```

- 注意:
 - 旋转会改变坐标轴的方向

基点设置

转换基点

- 基点: 进行css的转换时的参考点
- 通过`transform-origin`属性可以设置转换基点的位置, 盒子默认的转换基点是盒子的中心点, , 默认值为center center
- 语法:

```
transform-origin: x y;
```

- 单词: top bottom left right center 两两搭配使用

- 像素：左上角相当于0px 0px
- 百分比：参考的是盒子的宽度和高度，50% 50% 相当于center center
- 注意：
 - 对于旋转和缩放来说，因为其转换和盒子的中心点有关，改变了转换基点的位置，旋转和缩放会受影响
 - 对于位移来说，因为位移参考的盒子原来的位置，改变了转换基点的位置，原来的位置不会发生改变，位移不会受到影响
 - 如果盒子需要设置转换基点，需要设置在盒子样式未被更改时的位置，因为转换基点的默认值为盒子的中心点center center，如果在改变盒子样式时设置转换基点，转换基点相当于会从center center 改变为指定的位置，那么转换的效果容易错乱

3D空间介绍

3D转换

3D转换是2D转换的一种补充，基于css样式来完成页面中3D效果

在2D平面上，新增了一条z轴，形成了三维坐标系

三维坐标系

- 概念：在x轴和y轴的基础上，新增了z轴，形成了三维坐标系
- 三维坐标系：x轴的正方向默认水平向右，y轴的正方向是垂直向下，z轴的正方向是垂直屏幕向外
- 左手系统：食指是水平向右是x轴的正方向，中指垂直向下是y轴的正方向，大拇指垂直屏幕向外是z轴的正方向



有了三维坐标系，我们可以用(x,y,z)就表示三维空间中任意一个点



景深

- 概念：在摄像中，景深是指相机对焦点的相对清晰的成像范围，即物体和镜头之间的距离。
在开发中，我们将景深转化物体在z轴上的显示距离。
- 作用：css可以通过perspective属性来设置盒子和屏幕之间的距离，即景深，景深越大，物体距离屏幕越远，景深越小，物体距离屏幕越近（一旦设置景深，盒子会呈现近大远小的效果）
- 语法：

```
perspective: 距离;
```

- 注意：
 - 景深是给父盒子设置景深，子元素会在3D空间呈现效果，实现近大远小的效果
 - 景深越大，3D效果越小，景深越小，3D效果越明显
 - 景深配合3D转换一起使用

3D基础属性

3D转换

位移

- 让盒子进行移动
- 语法：

```
沿着x轴进行移动  
transform: translateX(x轴移动量);  
沿着y轴进行移动  
transform: translateY(y轴移动量);  
沿着z轴进行移动  
transform: translateZ(z轴移动量);  
同时设置三条轴进行移动  
transform: translate3d(x轴移动量, y轴移动量, z轴移动量);
```

- 注意：沿着z轴进行移动，会呈现近大远小的效果

缩放

- 让盒子进行放大和缩小
- 语法：

```
沿着x轴进行缩放，宽度的缩放  
transform: scaleX(倍数);  
沿着y轴进行缩放，高度的缩放  
transform: scaleY(倍数);  
沿着z轴进行缩放，厚度的缩放，看不到效果  
transform: scaleZ(倍数);
```

- 注意：沿着z轴进行缩放，缩放的是盒子的厚度，一般看不到效果，标签本身没有厚度

旋转

- 让盒子进行旋转
- 先确定旋转轴的方向，再设置旋转角度
- 语法：

```
沿着x轴的方向进行旋转  
transform: rotateX(角度);  
沿着y轴的方向进行旋转  
transform: rotateY(角度);  
沿着z轴的方向进行旋转  
transform: rotateZ(角度);  
自定义旋转轴进行旋转  
transform: rotate3d(x, y, z, 角度);
```

- x,y,z确定了三维坐标系中的一个点
- 旋转轴：是由坐标原点 (0, 0, 0) 到指定的点的连线的方向就是旋转轴的方向
- 注意：无论旋转轴在哪个方向，旋转轴都会过转换基点

灭点

perspective-origin

- 设置视线的灭点，设置3D盒子的观察的位置，默认舞台的中心点，默认正对着舞台
- 作用：以不同的角度观察盒子的变换
- 语法：

```
perspective-origin: x轴 y轴;xxxxxxxxx perspective-origin: x轴 y轴;  
perspective-origin: x轴 y轴;css
```

- 默认值为center center， 默认观察者正对着舞台的中心点观察盒子的变换
- 单词：top bottom left right center 两两搭配使用
- 像素：0px 0px相当于左上角
- 百分比：50% 50%相当于center center
- 注意：
 - 这个属性是设置在设置景深的盒子上
 - 如果视线灭点在舞台的中心点，并且盒子也在舞台的正中心，盒子显示的效果是对称的
 - 一个舞台只有一个视线灭点

transform-style

- 定义里面嵌套元素是如何在3D空间中显示的，设置子元素如何在父元素中显示
- 语法：

```
transform-style: flat | preserve-3d;
```

- flat：默认值，代表所有子元素在2D平面上显示
- preserve-3d：代表所有子元素在3D空间中显示
- 注意：
 - 需要结合景深的效果一起使用

Animation动画

完成自动播放的动画

- 使用flash技术可以完成动画的设计
- 编写JavaScript的脚本完成动画
- 借助某些动画标签，比如marquee
- h5提供了canvas技术（绘图技术），完成动画的设计
- css3提供了Animation动画可以完成自动持续播放

动画

- 概念：快速切换连续图片而达到流畅的画面，即动画
- 帧数：每秒切换图片的数量，数量越多，画面越流畅，60hz即每秒60帧，
- 关键帧：就是指每个关键的动作



关键帧

- 概念：是指盒子状态要发生改变的帧，就是将需要播放的动画拆分为多个步骤，每个步骤就是一个关键帧，将多个步骤连接起来，形成完整的动画
- 例子：盒子右移500px 关键帧有2个，开始状态和结束状态
- 例子：盒子变圆后，右移500px，关键帧有3个，开始状态——变圆——右移500px结束状态
- css动画中，关键帧需要定义

Animation动画基本使用

- 通过Animation的相关属性调用关键帧，实现动画的自动播放
- 步骤：
 - 利用 `@keyframes` 属性来定义动画的所有关键帧，给动画取名
 - 给需要设置动画的盒子添加 `animation` 相关属性调用动画的关键帧

定义关键帧

- 利用 `@keyframes` 属性来定义动画的所有的关键帧
- 注意：这个关键帧属于css的代码，不需要写在任何选择器中
- 语法：

```
先定义关键帧
@keyframes 动画名称{
    from{
        开始状态
    }
    to{
        结束 状态
    }
}

还可以使用百分比来定义关键帧
@keyframes 动画名称{
    0%{
    }
    10%{
    }
    20%{
    }
    100%{
    }
}

再在选择器中调用的关键帧
div{
    animation-name: 动画名称;
    animation-duration: 动画的执行时长;
}
```

- 关键帧在定义过程中，`from`相当于0%，`to`相当于100%
- 使用百分比定义关键帧时，每一帧的执行时长和两帧之间的百分比有关系，跨度越大，分配的时间越多
- 公式：分配时间 = 总时间 * 两帧之间的百分比之差

动画属性

动画属性

- CSS3提供了很多CSS属性来控制动画的播放速率、次数等等
- 属性：
 - `animation-name`：设置动画的名称，需要配合动画的执行时长一起使用
 - `animation-duration`：设置动画的执行时长，默认为0s
 - `animation-delay`：设置动画的延迟时长，默认为0s
 - `animation-timing-function`：设置动画的执行速率，原理是贝塞尔曲线
 - `ease`：默认值，以低速开始，然后变快，在结束前变慢
 - `linear`：匀速
 - `ease-in`：是以低速开始
 - `ease-out`：以低速结束
 - `ease-in-out`：以低速开始和结束
 - `cubic-bezier(x1,y1,x2,y2)`：贝塞尔曲线
 - `animation-iteration-count`：设置动画的播放次数
 - `n`：代表数字，动画执行几次，就写几，默认是1
 - `infinite`：无限次
 - `animation-direction`：设置动画的播放方向
 - `normal`：默认值，正常播放，正向播放动画
 - `reverse`：反向播放动画
 - `alternate`：奇数次正向播放动画，偶数次方向播放动画
 - `alternate-reverse`：奇数次反向播放动画，偶数次正向播放动画
 - `animation-fill-mode`：将第一帧或者最后一帧作用在元素上，
 - `backwards`：将第一帧作用在元素上，保持开始状态
 - `forwards`：将最后一帧作用在元素上，保持结束状态
 - `both`：同时将第一帧和最后一帧作用在元素上，保持开始和结束状态
 - `animation-play-state`：设置动画的播放状态
 - `running`：默认值，播放
 - `pause`：暂停
 - `animation`：复合属性

```
animation: 动画名称 执行时长 执行速率 延迟时长 执行次数 执行方向 是否保持开始或者结束状态;  
animation: name duration timing-function delay iteration-count  
         direction fill-mode;
```

- 至少设置动画的名称和执行时长，动画才能正常执行

布局分类

常见的布局方式

固定布局

- 原理：主要内容区域是采用固定的像素进行布局（固定尺寸比如960px~1200px等等），再水平居中
- 一般用于pc端，一般采用中间内容固定，两边自适应
- 好处：
 - 布局简单：大盒子设置固定宽度，在设置 `margin:0 auto;`
- 缺点：
 - 无适应性（对移动端不友好）
 - 会产生滚动条

流式布局

- 原理：为了适应不同的屏幕，主要内容区域采用百分比进行布局，采用百分比替代固定像素，不同的屏幕的有一定适应性，无论宽度如何改变，页面的布局不会发生改变
- 好处：
 - 对不同的屏幕有一定的适应性
- 缺点：
 - 当屏幕足够小时，导致内容无法查看

响应式布局

- 原理：利用媒体查询技术，实现当使用不同的屏幕时采用不同的css代码，即一套HTML代码多套css样式代码，达到不同的屏幕有不同的用户体验
- 好处：
 - 有很好用户体验（对移动端友好）
- 缺点：
 - 大量的css代码，开发难道比较大，维护性下降

移动端布局

- 专门针对移动端单独开发一套HTML和css代码，移动端代码和pc端的代码完全独立开
- 好处：
 - 移动端有极好的用户体验
- 缺点：
 - pc端和移动端的代码是独立，代码量比较大

媒体查询

响应式布局

- 概念：是指利用媒体查询技术，实现不同的屏幕采用不同的css代码，实现不同的屏幕都有良好用户体验。核心在于媒体查询。

媒体查询

- 媒体：在网页中是指各种设备，比如：电脑、手机、电视等等
- 查询：检测当前的屏幕属于哪种设备，以及属于哪种类型，根据这些信息来使用对应的css代码，实现不同的屏幕有不同页面效果。

基本使用

- 在引入css样式文件时规定样式的作用的范围

```
only 代表仅仅    print代表打印机    screen代表彩色屏幕  
<link rel="stylesheet" href="css文件文件路径" media="only print">
```

- 在css样式中规定样式的作用范围

```
@media 媒体类型{  
    选择器{  
        css代码  
    }  
}
```

媒体类型

- 可以将不同的设备进行分类，通过针对不同的设备写不同的css代码

值	设备类型
All	所有设备
Print	打印设备
Screen	电脑显示器，彩色屏幕
Braille	盲人用点字法触觉回馈设备
Embossed	盲文打印机
Handheld	便携设备
Projection	投影设备
Speech	语音或者音频合成器
Tv	电视类型设备
Try	电传打印机或者终端

媒体特性

- 概念：媒体设备的特性，用来描述设备的特点，包含设备的宽度、高度、最大最小宽度、最大最小高度、或者横屏竖屏等
- 可以通过媒体的特性来区分pc端还是移动

值	描述	
width	网页显示区域完全等于设备的宽度	
height	网页显示区域完全等于设备的高度	
max-width	网页显示区域小于等于设备宽度	
max-height	网页显示区域小于等于设备高度	
min-width	网页显示区域大于等于设备的宽度	
min-height	网页显示区域大于等于设备的高度	
orientation	portrait (竖屏模式) \	landscape (横屏模式)

媒体查询关键字

- 用于连接媒体特点的连接词，可以将多个媒体特性连接在一起
- 关键字：
 - `and`：代表并且的意思，既要满足前面的条件，又有后面的条件，同时满足才能作用css的样式

```
/* 375px~750px */
@media screen and (min-width:375px) and (max-width:750px) {
  .box{
    background-color: pink;
  }
}
```

- `,`：代表或者的意思，满足其中一个条件即可

```
/* 375以下或者750以上的屏幕 */
@media screen and (max-width:375px),screen and (min-width:750px){
  .box{
    background-color: tomato;
  }
}
```

- 注意：逗号前后媒体类型是单独的
- `not`：代表否定、不是、非的意思

```
/* 除了320px之外 */
@media not screen and (width: 320px) {
  .box{
    background-color: tomato;
  }
}
```

- 注意：not否定的是媒体的特性
- `only`：代表仅仅的意思，强调

```
@media only screen and (width: 320px) {}
```

- 代表唯一，目前新版本的浏览器没有区别，在老版本中，用来排除一些不兼容的版本。

Bootstrap

基本概念

- Bootstrap就是可以快速构建响应式网站的一款前端框架
- 框架：指的是其他的开发人员或公司为了简化某一领域的开发流程，而设计出来的有序代码。简称第三方开发工具。
- Bootstrap内置了很多组件，只需要添加class名就可以使用样式
- Bootstrap内置很多js插件，比如轮播图
- Bootstrap3和Bootstrap4的区别：
| BootStrap 3 | BootStrap 4 | | ----- |
----- | | less语言编写 | sass语言编写 | |
4种栅格类 | 5种栅格类 | | 使用px为单位 | 使用rem和em为单位（除部分margin和padding
使用px） | | 使用push和pull向左右移动 | 偏移列通过offset-类设置 | | 使用float的布局方
式 | 选择弹性盒模型（flexbox） |
 - Bootstrap3的4种栅格：
 1. 特小 (col-xs-) 适配手机(<768px)
 2. 小 (col-sm-) 适配平板(≥768px)
 3. 中 (col-md-) 适配电脑(≥992px)
 4. 大 (col-lg-) 适配宽屏电脑(≥1200px)
 - Bootstrap4的5种栅格：
 1. 特小 (col-) (<576px)
 2. 小 (col-sm-) (≥576px)
 3. 中 (col-md-) (≥768px)
 4. 大 (col-lg-) (≥992px)
 5. 特大 (col-xl-) (≥1200px)
 - Bootstrap4特点
 1. 新增网格层适配了移动端；
 2. 全面引入ES6新特性（重写所有JavaScript插件）；
 3. CSS文件减少了至少40%；
 4. 所有文档都用Markdown编辑器重写；
 5. 放弃对IE8的支持
- Bootstrap的官网：[Bootstrap中文网](#)

基本使用

- 步骤：
 - 下载Bootstrap的生成文件夹
 - 在页面中引入Bootstrap.css或者Bootstrap.min.css
 - 先引入jQuery的js文件
 - 再引入Bootstrap.bundle.js或者Bootstrap.bundle.min.js
- 语法：

```
<!-- 使用本地的Bootstrap文件 -->
<link rel="stylesheet" href="css/bootstrap.min.css">
<!-- 先引入jQuery的js文件 -->
<script src="js/JQuery2.1.4.js"></script>
<!-- 引入Bootstrap的js文件 -->
<script src="js/bootstrap.bundle.min.js"></script>
```

- 注意：必须先引入jQuery的js文件，再引入Bootstrap的js文件

js插件-拓展： hightchars图表

图表 框架

highcharts

- 实例丰富
- 文档是英文
- 稳定性比Echarts好
- [Highcharts 演示 | Highcharts](#)

Echarts

- 百度地图团队开发
- 能够结合百度地图一起使用
- 内部使用vue语法来写数据部分代码

页面布局

HTML

```
<body>
    <!-- 头部 -->
    <header>
        <!-- 顶部登录注册按钮 -->
        <div class="header-top">
            <div class="center-container">
                <a href="#">下载 APP</a>
                <div>
                    <a href="#">登录</a>
                    <a href="#">注册</a>
                </div>
            </div>
        </div>
        <!-- 导航栏 -->
        <div class="header-nav">
            <div class="center-container">
                <!-- logo -->
            </div>
        </div>
        <!-- 轮播图 -->
        <div class="header-banner">
            <div class="center-container">
                </div>
        </div>
    </header>
```

```
</header>

<!-- 主体内容 -->
<main>
    <div class="center-container">
        <!-- 下拉列表 -->
        <div></div>
        <!-- 电影相关信息 -->
        <div>
            <!-- 电影列表 -->
            <div class="movies-list">
                <!-- 正在热映 -->
                <div>
                    <!-- 标题 -->
                    <div></div>
                    <!-- 正在热映的电影列表 -->
                    <div class="moviesList" id="hotMoviesList">
                        <!-- 每一个电影项 -->
                        <!-- <div class="movie-item">
                            
                            <div class="movie-name">
                                <span>怒火重案</span>
                                <span>9.0</span>
                            </div>
                        </div> -->
                    </div>
                </div>
                <!-- 即将上映 -->
                <div>
                    <div></div>
                    <div class="moviesList" id="comingMoviesList"></div>
                </div>
            </div>
            <!-- 电影活动 -->
            <div></div>
        </div>
    </div>
</main>
<!-- 底部 -->
<footer>
    <div class="center-container">
        1212124124124
    </div>
</footer>

<script src=".//utils/jquery-3.6.0.min.js"></script>
<script src=".//utils/utils.js"></script>
<script src=".//js/getData.js"></script>
<script src=".//js/home.js"></script>
</body>
```

CSS

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

.header-top {
    height: 40px;
    background-color: #f0f0f0;
}

.center-container {
    width: 1200px;
    margin: 0 auto;
    /* outline: 1px solid red; */
}

.header-top .center-container {
    display: flex;
    justify-content: space-between;
    line-height: 40px;
    /* align-items: center; */
}

.header-banner {
    height: 360px;
    background-image: url('../images/banner02.png');
    background-position: center;
}

.movies-list {
    width: 730px;
}

.moviesList {
    display: flex;
    flex-wrap: wrap;
    justify-content: space-around;
}

.movie-item {
    width: 160px;
    border: 1px solid #ddd;
    margin: 20px 0;
}

.movie-item > img {
    width: 160px;
    height: 220px;
}

.movie-item > .movie-name {
    display: flex;
    justify-content: space-between;
    line-height: 40px;
    padding: 0 10px;
}
```

```
}
```

```
footer {  
    height: 290px;  
    background-color: #262426;  
}
```

解决margin塌陷问题

一：什么是margin塌陷？

在标准文档流中，竖直方向（左右方向的不会出现塌陷现象）的margin会出现叠加现象，即较大的margin会覆盖掉较小的margin，竖直方向的两个盒子中间只有一个较大的margin值，这就是margin的塌陷现象。

二：margin塌陷分类？

- (1) 兄弟盒子之间的塌陷问题。
- (2) 父子盒子之间的塌陷问题。

解决方案：

- (1) 给父元素father添加一个border边框，比如给box1添加样式：border: 1px solid transparent这种方式造成贴合的样子；
- (2) 给父元素添加一个overflow: hidden;
- (3) 给父元素添加一个position: fixed;
- (4) 给父元素添加一个display: table;
- (5) 将子元素都margin改为父元素的padding
- (6) 给子元素son添加一个兄弟元素属性为content:"";overflow: hidden;

思考：边框上有文字的方法。

```
<fieldset>  
    <legend>账号</legend>  
</fieldset>
```

调整legend的margin实现

关联点击

描述：当点击.close时，相当于点击了.cancel

案例代码如下：

```

<div class="close" onclick="cancelBtn.click()">
    
</div>
<button class="cancel" id="cancelBtn">取消</button>

```

单行、多行文本溢出显示省略号 (...)

一、单行超出显示省略号

描述：如果文字超出父元素指定宽度，文字会自动换行，而连续不间断数字和英文字母（没有其他字符）不会自动换行； 详细步骤：

第一步(不换行): white-space: nowrap; (对于连续的数字或者英文字母可省略) 第二步 (溢出隐藏) overflow: hidden; 第三步 (文本溢出显示省略号) text-overflow: ellipsis;(省略号)

二、多行超出显示省略号

a.对于内核是webkit的浏览器 (Google/Safari) ,可以直接用CSS样式;

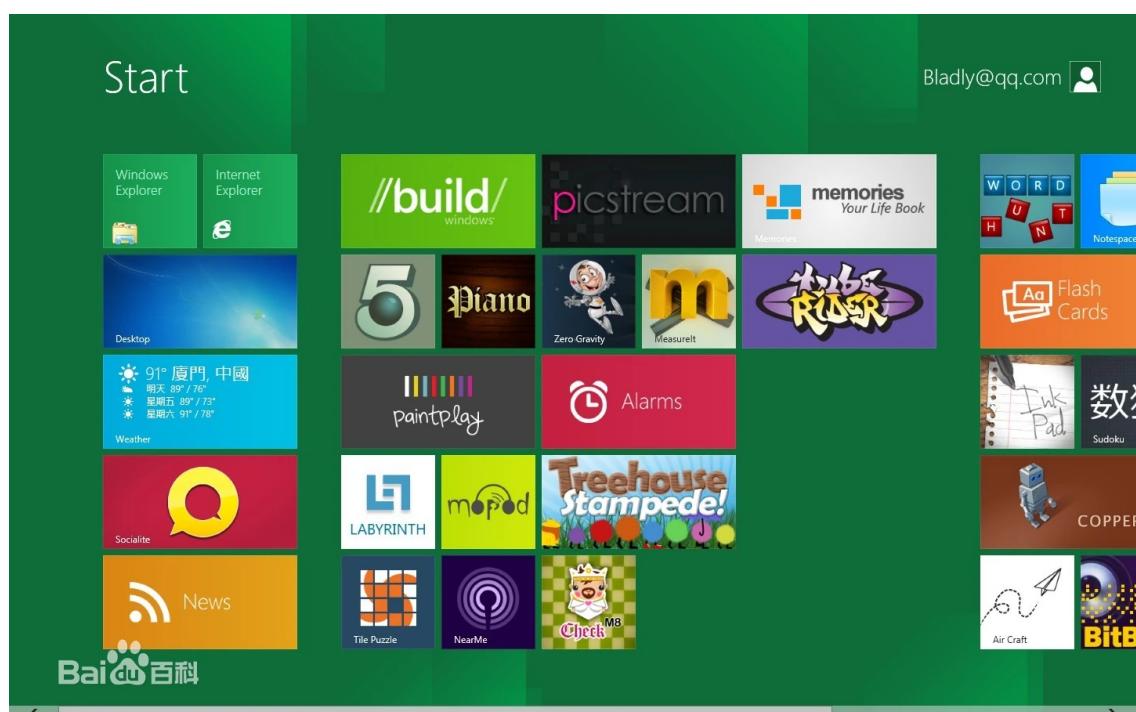
溢出隐藏: overflow: hidden; 省略号: text-overflow: ellipsis; display: -webkit-box; 弹性盒模型：设置弹性盒子的子元素的排列方式 : -webkit-box-orient: vertical; 设置显示文本的行数： -webkit-line-clamp: 3; (最多显示3行)

1.Metro UI

Metro是由[微软公司](#)开发的内部名称为“typography-based design language” (基于排版的设计语言)。

强调信息本身的Metro UI是一种界面展示技术。

Metro基于瑞士平面设计的设计原则，有利于以文字为主的界面导航。



2.Table布局

3.Grid布局

4.flex布局

5.float布局

6.CSS引用

课后实践4(2023.10.09)

The page features a top banner with a pink teacup and saucer, green leaves, and the brand name "JARDINES DEL TÉ" in Spanish and Chinese. Below the banner are three product sections: "百佳妍迷迭香茶" (Baijiajian Rosemary Tea) with a globe icon; "七彩云南醇香袋泡茶" (Seven Colors Yunnan Pure Fragrance Bagged Tea) with a row of tea canisters; and "中茶大益茶园滇红茶" (Chungcha Da Yi Tea Garden Dianhong Tea) with a green tea canister. Each section includes a detailed product description.

百佳妍迷迭香茶
用途广泛，其强烈香气会随着热水的漫溢而扩散到茶雾之中，有令人清醒的功效。身心疲劳、头脑沉闷时，泡上这样一杯茶，迷迭香将会在体内释放原有的活力，精神也为之振奋起来。迷迭香草中含有有助于人体健康的物质能促进血液循环，强化血管，所以迷迭香又有“回春茶”之称，它还活化脑部，提高记忆力，早上喝会觉得神清气爽，充满活力。

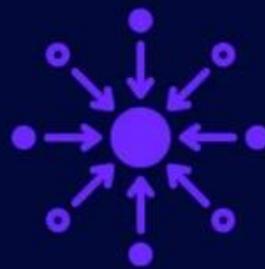
七彩云南醇香袋泡茶
七彩云南醇香普洱茶，汤色绝妙，芳香诱人，醇香回甘，美味悠长；醇香普洱精选上等普洱，提取一叶一芽之精华，以丰富品质带来浓郁滋味，非常健康！地道普洱熟茶，底端温润柔和，散发熟茶特有的悠悠陈香，将普洱茶大叶肥大的条索切细部分为颗粒状，使得品质优异，香郁持久，味美纯然。现代商务精英首选，简约品位，在享受醇香四溢的同时获得纯正普洱茶的喜爱。

中茶大益茶园滇红茶
滇红的品种众多，加糖加奶调味和饮用为主，加奶后的香气滋味依然浓烈。冲泡后的滇红茶汤红艳明亮，唇齿留香，茶汤与茶杯接触处常挂金圈，冷却后立即出现乳凝现象，冷却后浑浊出现香气品质优美的表现。滇红茶最大的特征为茸毫显露，毫色分淡黄、箭黄、金黄、凤庆、云昌等地生产的滇红工夫茶，毫色呈黄绿；临沧、勐海等地所产，毫色则多为金黄。

关于我买网 | 送货范围 | 退换货政策 | 媒体报道 | 友情链接 | 联系我们 | 招聘信息
Copyright © 2022 All Rights Reserved XXXXX Group 版权所有
技术支持：西安市sky技术有限公司

1. web (World Wide Web) 即全球广域网，也称为万维网。 [上一节](#)

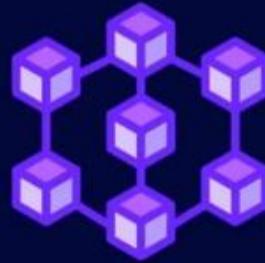
Web 1.0



Web 2.0



Web 3.0



WHAT ARE WEB1, WEB2, WEB3?

1

3.

简介

- web (World Wide Web) 即“万维网”，它是一种基于文本和HTTP的、全球性的、动态交互的、跨平台的分布式信息/信息系统。
- 定义：不是某一个标准，而是一系列标准的集合。
- 主要由三部分组成：结构（Structure）、表现（Presentation）和行为（Behavior）。

标准

- 结构化标记语言
 - XML
 - HTML
- 表现标准语言
 - CSS
- 行为脚本语言
 - ECMAScript

Standard Generalized Markup language

SGML或SGML

简介

一种定义电子文档结构和描述其内容的国际标准语言。

所有新的文字符号记述的规范，早在万维网兴起之前，“开放标准”就已经存在了。

1986年国际标准化组织出版发布的多个信息管理方面的国际标准（ISO/IEC）。

标记语言

在电子计算机中，标记计算机所能理解的任意符号，通过此种标记语言，可以用以标记数据，定义数据类型，是一种允许用户对自己的标记语言进行自定义的标记语言。

优点

- 高稳定性
- 高可读性
- 高实用性
- 高兼容性
- 高复用性

缺点

- 无法限制数据，可读性差，限制时间长。

衍生

1989年，HTML诞生。它抛弃了SGML复杂大的缺点。

HTML最大的优点是简单性与通用性。

HTML是一种简化的技术，它只使用了SGML中很少的一部分。

为了便于计算机实现，HTML和它的标记是固定的。

HTML

缺点：
无法限制数据，可读性差，限制时间长。

衍生

1998年7月10日，W3C(World Wide Web Consortium)标准，HTML诞生。

XML

可扩展标记语言，标记通用语句的语言。简称XML。

一种用于表示电子文件使其具有结构性的标记语言。

Hyper Text Markup Language

简介

超文本标记语言，是一种标识语言的通用。

超文本是一种信息组织方式，它通过链接方法把文本中的文字、图形与其他信息连接起来。

HTML 1.0：在1993年3月为微软的工程小组（IETT）工具发布。

HTML 2.0：1995年1月为BFC 1865发布，于2000年6月发布之后将宣布已弃用。

HTML 3.2：1997年1月14日，W3C推出标准。

HTML 4.01：1997年12月18日，W3C推荐标准。

HTML 4.01（微小版）：1999年12月24日，W3C推荐标准。

HTML 5：HTML5是迄今为止的下一代标准，极大地提升了Web客户端、富客户端（如移动设备）和服务器端的能力，被看作是将改变行业巨擘们所要期待的。

历史

特点

- 可读性
- 平行优先级
- 通用性

2000年W3C成员决定停止对HTML，发现 XHTML，因此WHATWG于2004年7月10日，Mozilla Foundation, Apple, Opera Software 建议 W3C跳过WHATWG，使用HTML5，将新的HTML（标准通用语句语言下的一个应用）命名为“HTML5”。2007-4-7，新的HTML工作坊采纳了他们的建议。

HTML5

HTML5是面向未来的下一代标准，是构建以及呈现互联网内容的一种语言方式，被业界称为“语义化”。

简介

层叠样式表(Cascading Style Sheets)是一种用来声明HTML或XML等文件在计算机屏幕上显示效果的风格指南。

CSS不仅可以静态地声明风格，还可以配合各种脚本动态地对网页元素进行个性化。

1994年第一个实现了CSS的浏览器。

历史

1996年，CSS1（层叠样式表1级）发布，同年12月，层叠样式的第一次正式标准（Cascading Style Sheets Level 1）发布，成为W3C的推荐标准。

CSS2 发布于 1999 年 1 月 1 日，CSS2 增加了媒体（打印和屏幕设备设置）和可下载字体支持。

CSS3 和 CSS2 的区别在于模块化。

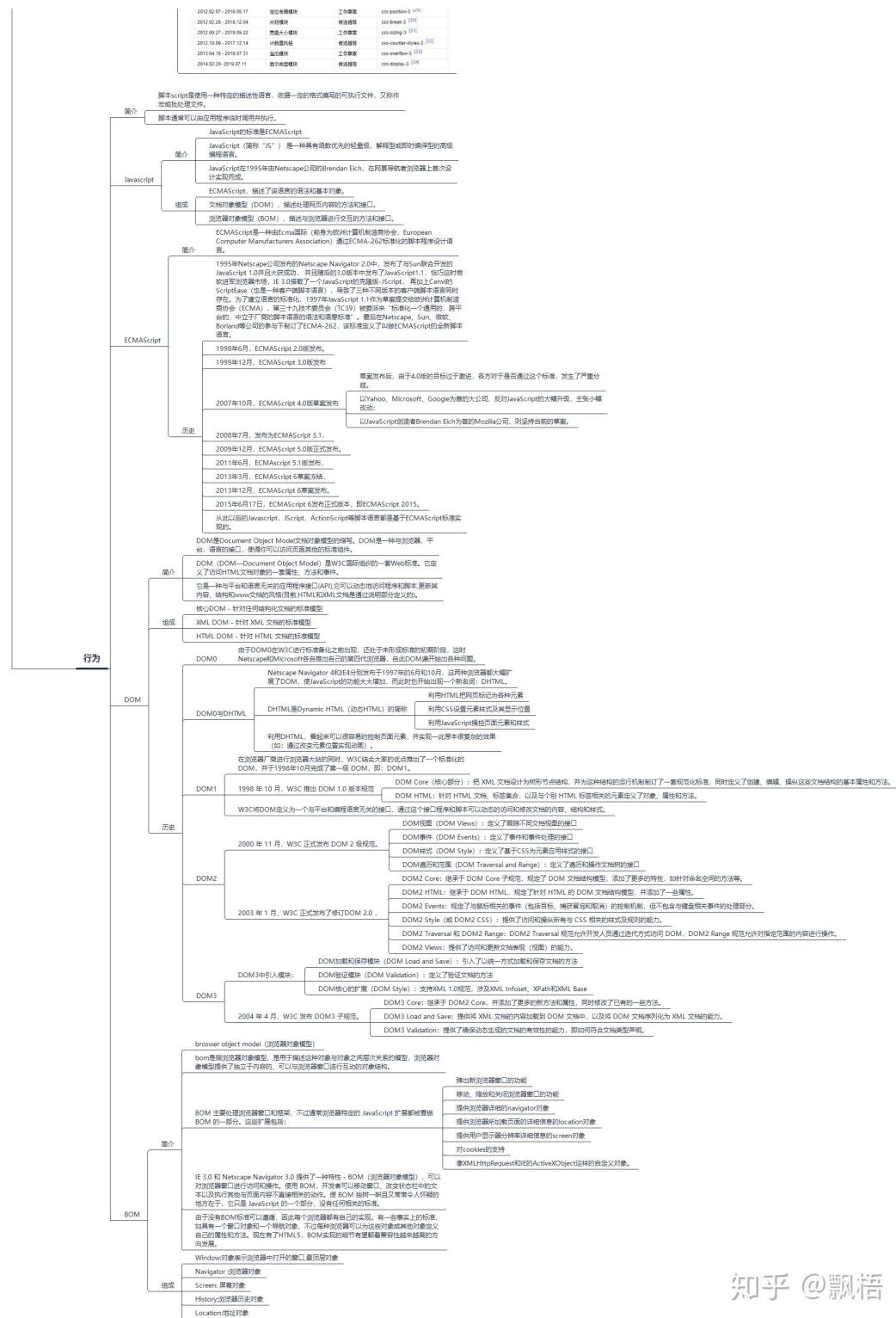
表现

CSS3 是 CSS 的一个主要变化是将CSS分为三个不同的模块。

模块	功能	兼容性
1996.01.27 - 2010.08.13	文本排版模块	IE6-IE9, FF3-FF10, CH10
2000.02.22 - 2010.08.13	盒子模型模块	IE6-IE9, FF3-FF10, CH10
1999.03.23 - 2010.05.15	尺寸与位置	IE6-IE9, FF3-FF10, CH10
1999.06.22 - 2010.08.13	颜色模块	IE6-IE9, FF3-FF10, CH10
1999.09.25 - 2010.04.20	变形与过渡效果	IE6-IE9, FF3-FF10, CH10
2000.03.03 - 2010.08.13	动画模块	IE6-IE9, FF3-FF10, CH10
2001.04.04 - 2012.02.19	视口属性	IE6-IE9, FF3-FF10, CH10
2000.09.17 - 2010.09.13	文本装饰	IE6-IE9, FF3-FF10, CH10
2001.09.26 - 2010.08.13	背景与边框	IE6-IE9, FF3-FF10, CH10
2001.07.13 - 2010.08.13	垂直滚动轴	IE6-IE9, FF3-FF10, CH10
2001.07.26 - 2010.08.13	基本尺寸属性	IE6-IE9, FF3-FF10, CH10
2001.07.31 - 2010.08.20	线性滤镜	IE6-IE9, FF3-FF10, CH10
2001.09.24 - 2017.10.17	背景与边框	IE6-IE9, FF3-FF10, CH10
2002.02.20 - 2010.08.17	列表项目	IE6-IE9, FF3-FF10, CH10
2002.05.15 - 2010.08.18	内联样表模块	IE6-IE9, FF3-FF10, CH10
2002.06.02 - 2010.08.21	基本尺寸与线性滤镜	IE6-IE9, FF3-FF10, CH10
2010.09.17 - 2010.09.13	背景与边框	IE6-IE9, FF3-FF10, CH10
2010.09.13 - 2010.09.16	透明度	IE6-IE9, FF3-FF10, CH10
2010.03.24 - 2014.01.14	颜色与单位模块	IE6-IE9, FF3-FF10, CH10
2012.12.15 - 2015.09.29	叠放与混合	IE6-IE9, FF3-FF10, CH10
2006.01.02 - 2014.01.13	分离式样表与生成内容	IE6-IE9, FF3-FF10, CH10
2006.01.01 - 2014.10.14	Margin和负值	IE6-IE9, FF3-FF10, CH10
2009.07.23 - 2010.09.19	图像效果	IE6-IE9, FF3-FF10, CH10
2012.02.12 - 2010.09.19	视口属性	IE6-IE9, FF3-FF10, CH10

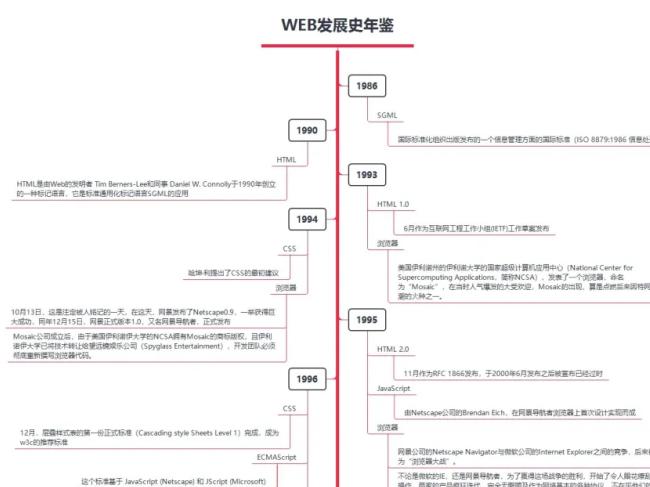
CSS3

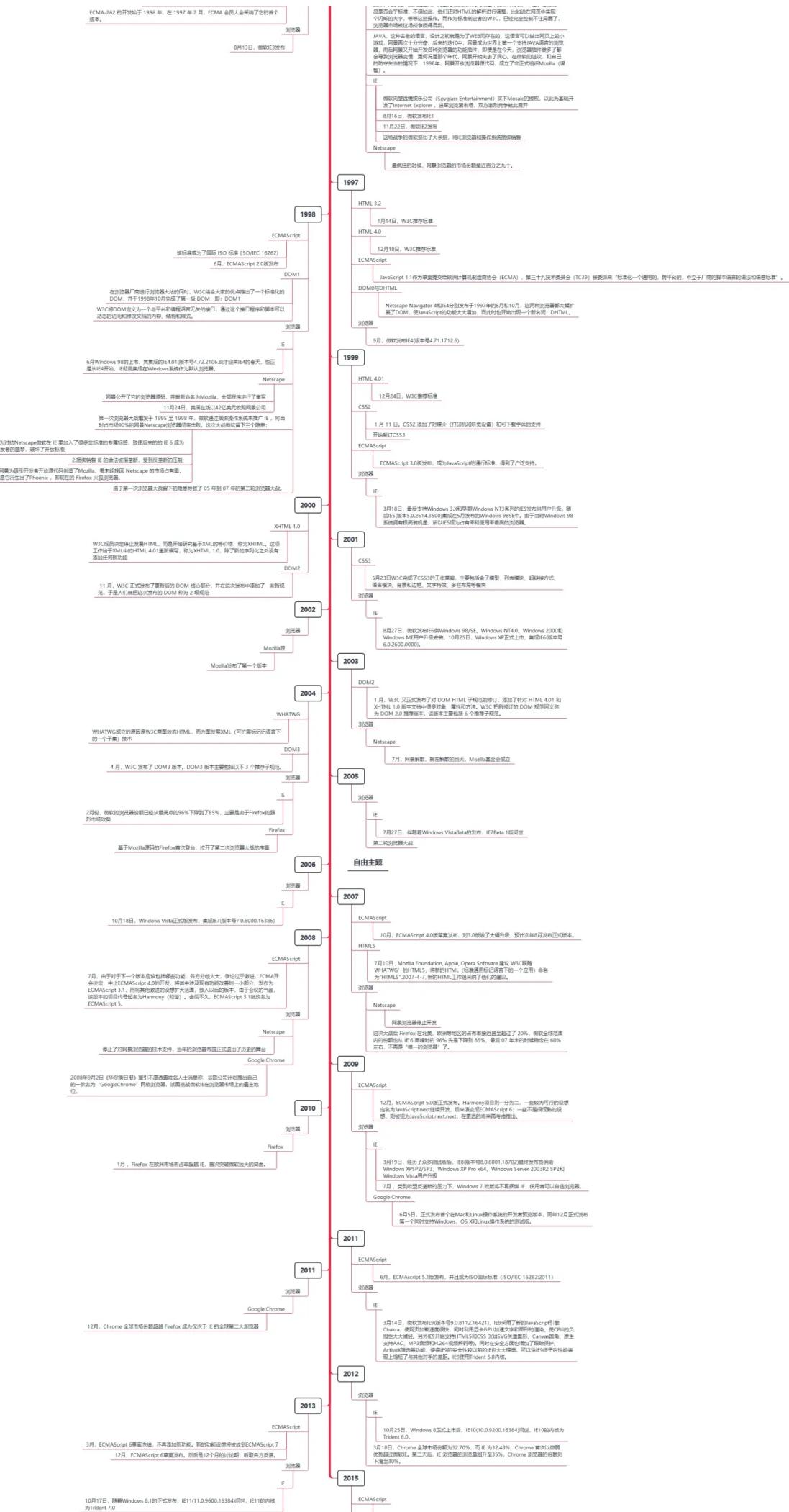
CSS3演进的一个主要变化是将CSS分为三个不同的模块。



知乎 @飘梧

1
4.





^

5. 转义字符，ASCII码字符集HTML字符集。 ↵

6. Web服务器。 ↵

