

全栈开发 & 系统设计串讲

小王子, 韩立, Lance 等FLAG老师

课程主要分为三个章节和部分。

实现一个项目来帮助理解全栈开发

了解全栈开发以及基础知识准备

系统设计以及云服务实践和思考

版权归上岸教育所有，任何不经同意的录象，转发行为将被视为侵犯版权。

我假设你们已经知道了...

JS的基础语法

网络及http传输的基础知识

Model-View-Controller

IDE, IntelliJ, Ultimate

使用mac的同学已经安装Homebrew

已经安装Node.JS

带上了一颗工程师的心

知道什么是 git 并会简单操作

已经阅读Angular 环境配置

已经安装最新的java JDK

什么是前端和框架

Angular, html, css, layout



PART 1

懒

一件事我要做两次以上，我就得考虑Automation

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>This is a Heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

新建一个文件，把这段代码复制进去并保存为.html


```
<!DOCTYPE html>
<html>
<body>

<p>Click the button to open a new browser window.</p>

<button onclick="myFunction()">Try it</button>

<script>
function myFunction() {
  window.open("https://shangan.us");
}
</script>

</body>
</html>
```

你要是加点花样还能跟浏览器互动下

原生html缺点

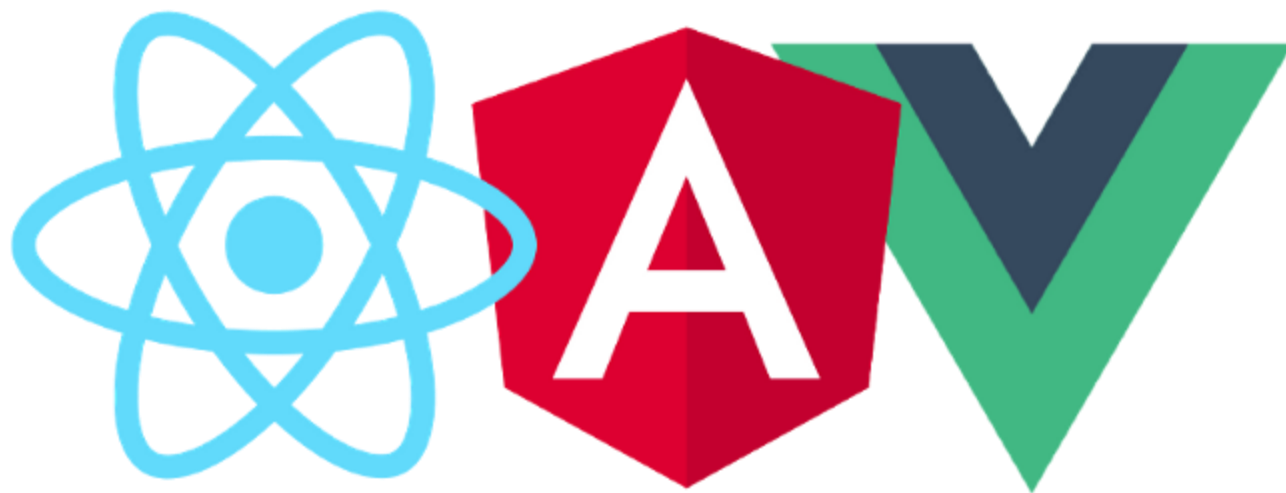


页面千千万，每个都重写营养跟不上。

框架年年变，原生总不变。以不变应万变。

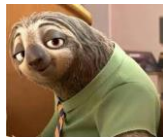
---某老鸟





主要还是因为

上岸



懒

一件事我要做两次以上，我就得考虑Automation

还有就是

上岸



钱

西方世界什么都讲究效率速度，框架就像机器，程序员就像工人。

2019-今

亚马逊谷歌这类公司招聘毕业生超过一万人

如何保障这些毕业生能在代码的生产流水线上快速产出？

框架就是秘诀

掌握一种框架就和二十年前会开车是一回事

我们为什么选



好学不劝退。

教完不退费。

<https://github.com/wang2510/angularTutor>

三 上岸



上岸一对一服务



由北美一线IT公司并修改过百份简历以上的面试官亲自线上与您语音沟通交流并代笔您的简历，一次修改，让您的简历达到北美一线IT公司的筛选标准

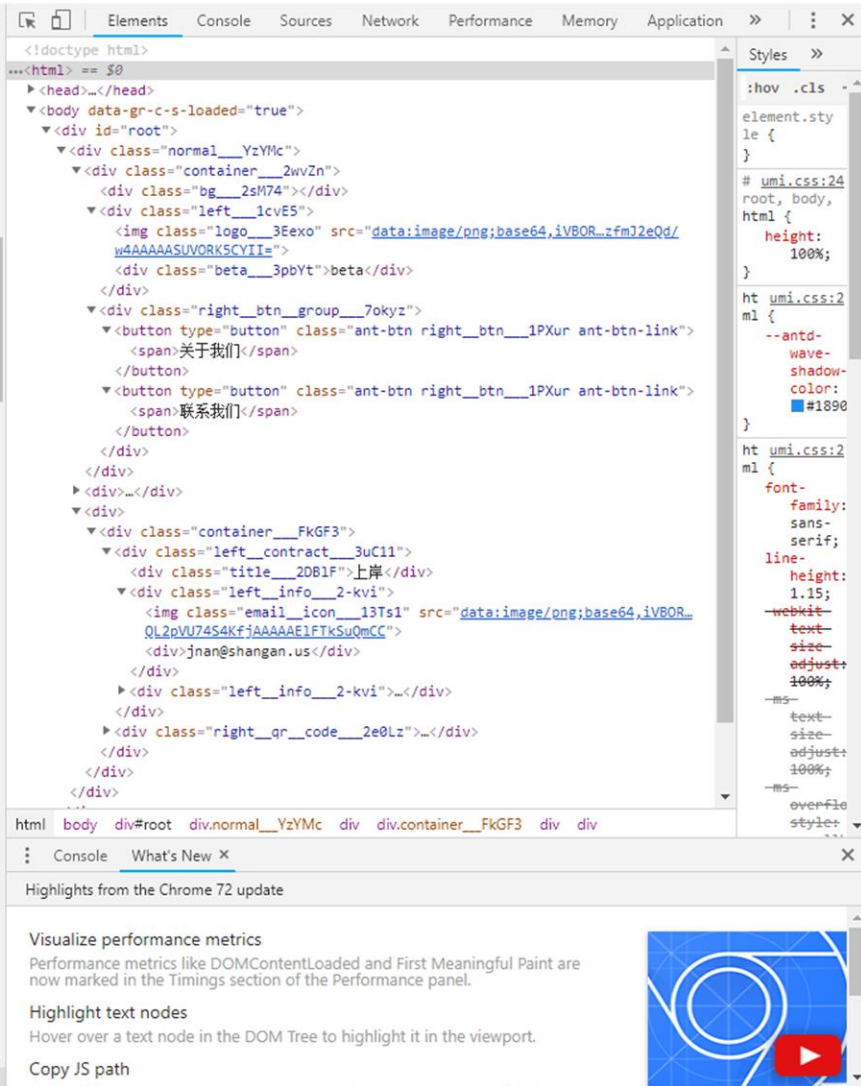


\$200/次

由北美一线IT公司面试官操刀，为您准备不同公司难度的面试，获得最真实的面试体验，并提供以一线IT公司招聘为标准的面试反馈。



服务保障



网格 + 数据 + 样式 = 页面

Html

Objects/Json

css

Page

我们先来看一下如何制作网格

Html结构是由以下一对格式表示的

```
<head> </head>
```

Html有三种基本

行状元素

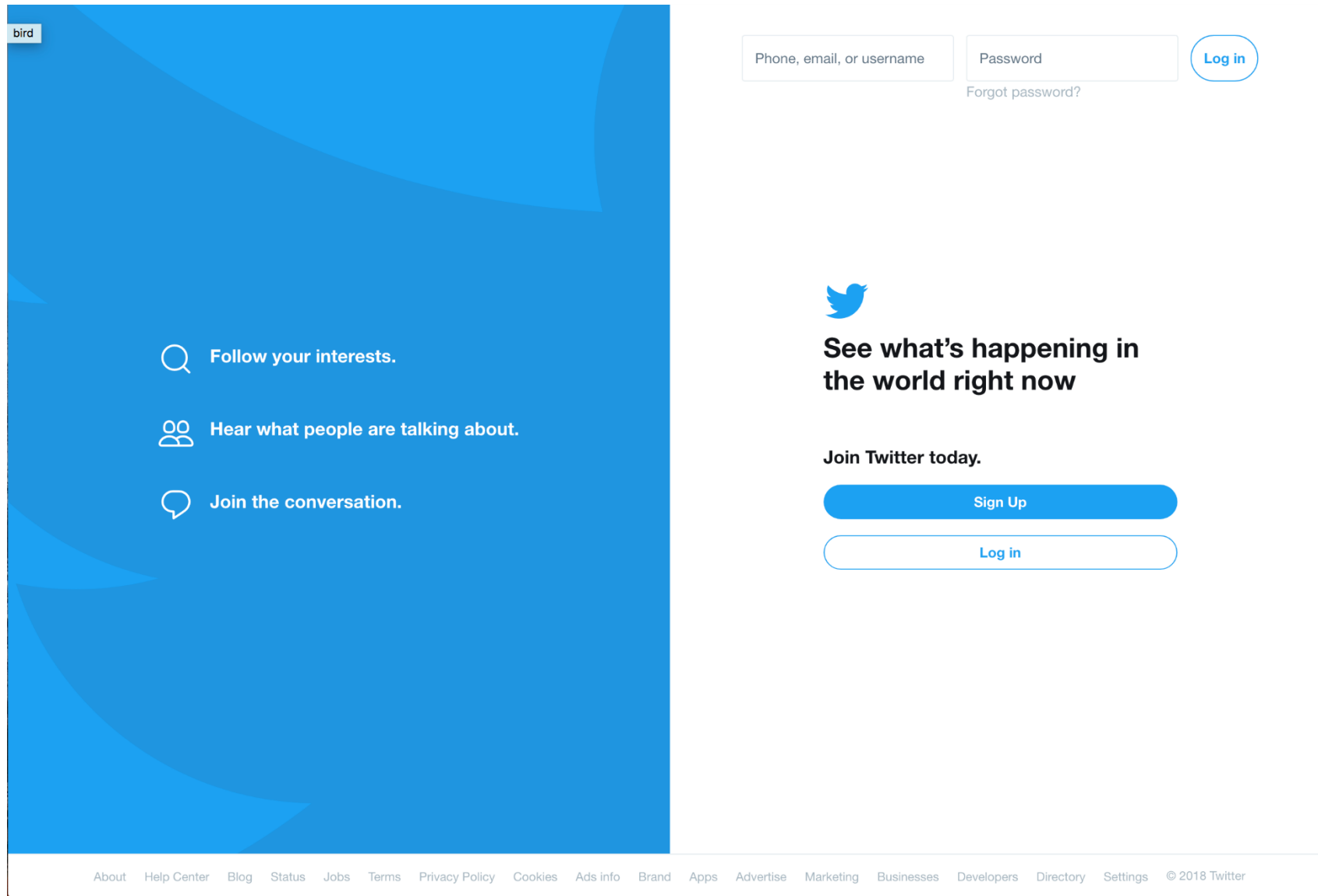
01 | 02

块状元素
<div> </div>

03

可变元素
<button> </button>

如果我们要做一个twitter的页面



如果我们要做一个twitter的页面

Div放了个图 →



Div放了几个字 →

**See what's happening in
the world right now**

Div放了几个字 →

Join Twitter today.

Div放了个按钮 →

Sign Up

Div放了个按钮 →

Log in

网格 + 数据 + 样式 = 页面

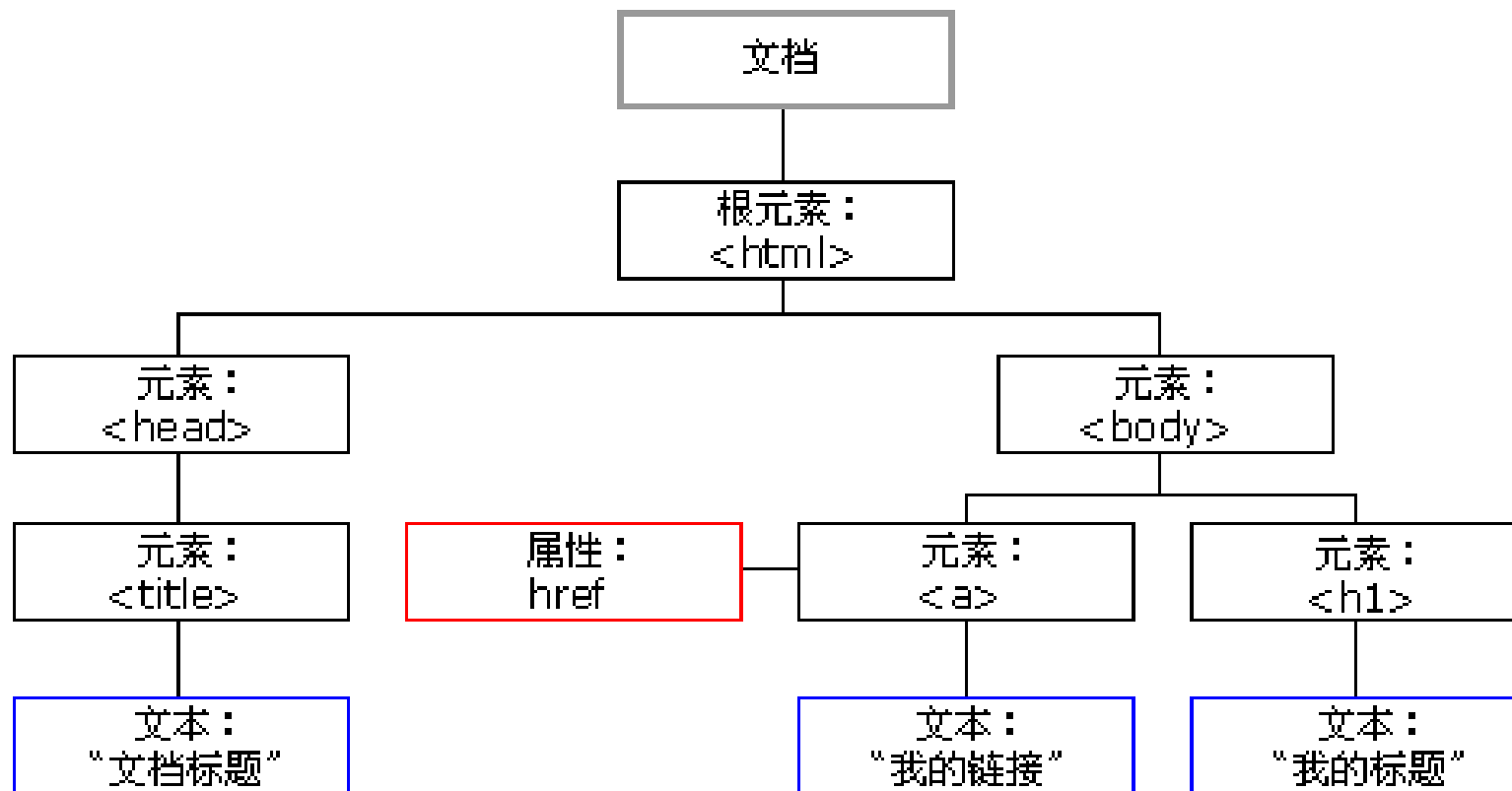
Html

Objects/Json

css

Page

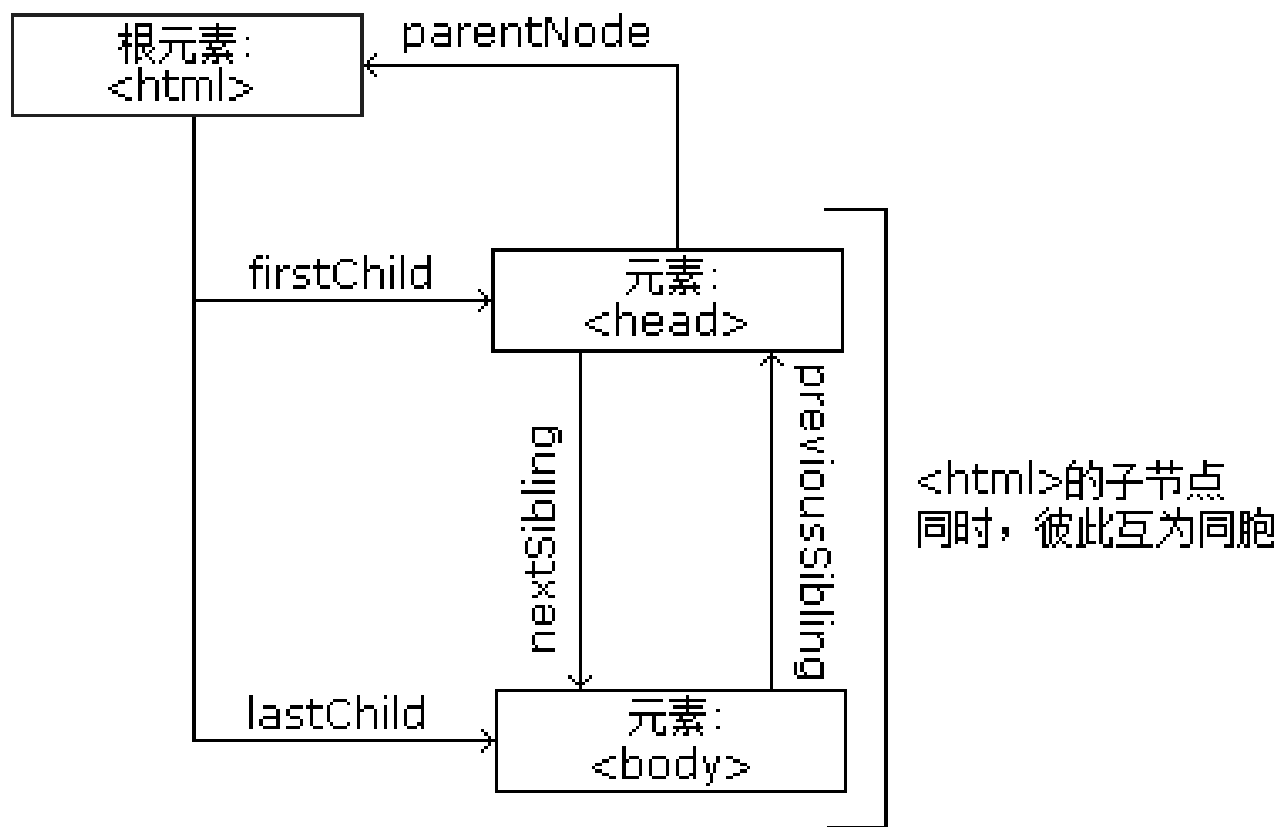
HTML DOM (Document Object Module)



父 (parent)、子 (child) 和同胞 (sibling) 等术语用于描述这些关系。父节点拥有子节点。同级的子节点被称为同胞 (兄弟姐妹)：

- 在节点树中，顶端节点被称为根 (root)
- 每个节点都有父节点、除了根 (它没有父节点)
- 一个节点可拥有任意数量的子
- 同胞是拥有相同父节点的节点

HTML DOM 的关系



根据 W3C 的 HTML DOM 标准，HTML 文档中的所有内容都是节点：

- A. 整个文档是一个文档节点
- B. 每个 HTML 元素是元素节点
- C. HTML 元素内的文本是文本节点
- D. 每个 HTML 属性是属性节点
- E. 注释是注释节点

数据和网格都有了

上岸



布局 and 美化

盒模型, 布局, Flex, css



PART 2

我们来聊一聊布局

传统方法



Flex布局



- 在我们讨论宽度的时候，我们应该讲下与它相关的另外一个重点知识：盒模型。当你设置了元素的宽度，实际展现的元素却超出你的设置：这是因为元素的边框和内边距会撑开元素。看下面的例子，两个相同宽度的元素显示的实际宽度却不一样。

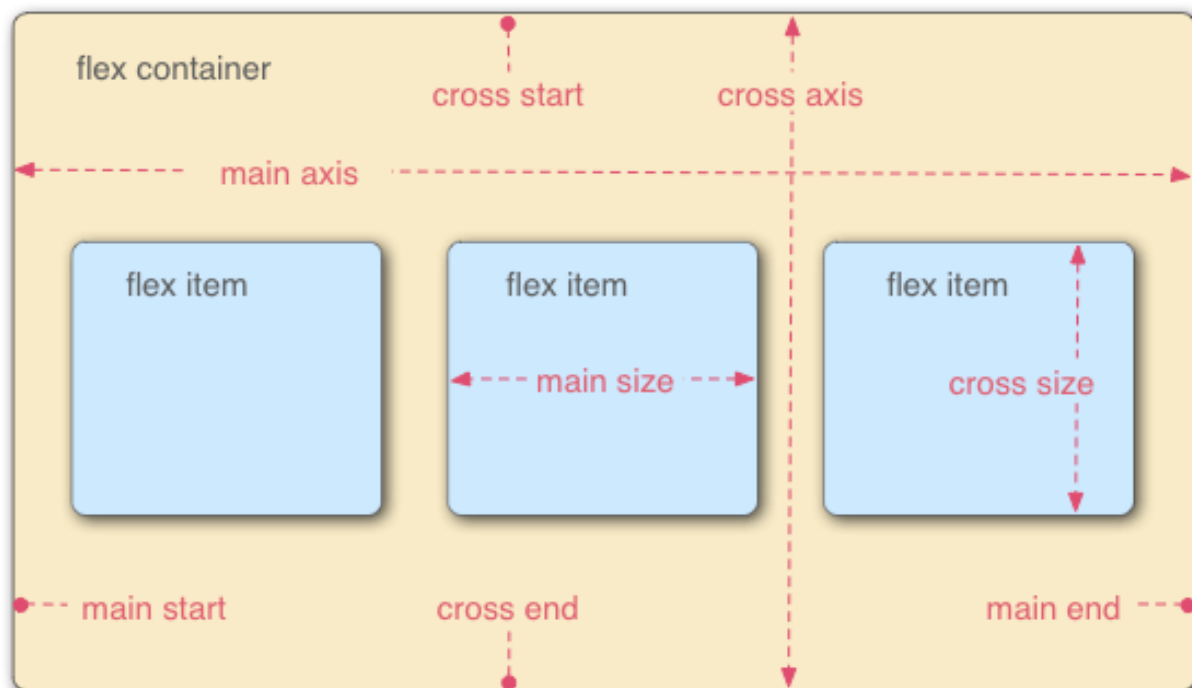
以前有一个代代相传的解决方案是通过数学计算。CSS开发者需要用比他们实际想要的宽度小一点的宽度，需要减去内边距和边框的宽度。值得庆幸地是你不需要再这么做了...



```
.simple {  
  width: 500px;  
  margin: 20px auto;  
}  
  
.fancy {  
  width: 500px;  
  margin: 20px auto;  
  padding: 50px;  
  border-width: 10px;  
}
```

```
<div class="simple">  
  我小一些...  
</div>
```

```
<div class="fancy">  
  我比它大!  
</div>
```



flex-direction
flex-wrap
flex-flow
justify-content
align-items
align-content

```
.box { flex-direction: row | row-  
reverse | column | column-  
reverse; }
```



- 传统布局

- 核心就是盒模型
- 依赖 display 属性 + position属性 + float属性。
- 垂直居中太难做

2009年，W3C 提出了一种新的方案----Flex 布局，可以简便、完整、响应式地实现各种页面布局。目前，它已经得到了所有浏览器的支持，这意味着，现在就能很安全地使用这项功能。

01. CSS背景属性(Background)

属性与描述

background : 在一个声明中设置所有的背景插件1

background-attachment : 设置背景图像是否固定或者随着页面的其余部分滚动1

background-color : 设置元素的背景颜色1

background-image : 设置元素的背景图像1

background-position : 设置背景图像的起始位置1

background-repeat : 设置是否及如何重复背景图像

02. CSS边框属性(Border和Outline)

属性与描述

border：在一个声明中设置所有的边框属性1

border-bottom：在一个声明中设置所有的下边框属性1

border-bottom-color：设置下边框的颜色2

border-bottom-style：设置下边框的样式2

border-bottom-width：设置下边框的宽度1

border-color：设置四条边框的颜色1

border-left：在一个声明中设置所有的左边框属性1

border-left-color：设置左边框的颜色2

border-left-style：设置左边框的样式2

border-left-width：设置左边框的宽度1

border-right：在一个声明中设置所有右边框的属性1

border-right-color：设置右边框的颜色2

border-right-style：设置右边框的样式2

border-right-width：设置右边框的宽度1

border-style：设置四条边框的样式1

border-top：在一个声明中设置所有上边框的属性1

border-top-color：设置上边框的颜色2

border-top-style：设置上边框的样式2

border-top-width：设置上边框的宽度1

border-width：设置四条边框的宽度1

outline：在一个声明中设置所有的轮廓属性2

outline-color：设置轮廓的颜色2

outline-style：设置轮廓的样式2

outline-width：设置轮廓的宽度

03. CSS文本属性(Text)

属性与描述

color : 设置文本的颜色1

direction : 规定文本的方向/书写方向2

letter-spacing : 设置字符间距1

line-height : 设置行高1

text-align : 规定文本的水平对齐方式1

text-decoration : 规定添加到文本的装饰效果1

text-indent : 规定文本块首行的缩进1

text-shadow : 规定添加到文本的阴影效果2

text-transform : 控制文本的大小写1

unicode-bidi : 设置文本方向2

white-space : 规定如何处理元素中的空白1

word-spacing : 设置单词间距1

04. CSS字体属性(Font)

属性与描述

font : 在一个声明中设置所有字体属性1

font-family : 规定文本的字体系列1

font-size : 规定文本的字体尺寸1

font-size-adjust : 为元素规定aspect值2

font-stretch : 收缩或拉伸当前的字体系列2

font-style : 规定文本的字体压实1

font-variant : 规定文本的字体样式1

font-weight : 规定字体的粗细

05. CSS外边距属性(Margin)

属性与描述

margin : 在一个声明中设置所有的外边距属性1

margin-bottom:设置元素的下外边距1

margin-left:设置元素的左外边距1

margin-right:设置元素的右外边距1

margin-top:设置元素的上外边距

06. CSS内边距属性(Padding)

属性与描述

padding: 在一个声明中设置所有的内边距属性1

padding-bottom: 设置元素的下内边距1

padding-left: 设置元素的左内边距1

padding-right: 设置元素的右内边距1

padding-top: 设置元素的上内边距

07. CSS列表属性(List)

属性与描述

list-style:在一个声明中设置所有的列表属性1

list-style-image:将图像设置为列表项标记1

list-position:设置列表项标记的放置位置1

list-style-type:设置列表项标记的类型

08. CSS尺寸属性(Dimension)

属性与描述

height:设置元素高度1

max-height:设置元素的最大高度2

max-width:设置元素的最大宽度2

min-height:设置元素的最小高度2

min-width:设置元素的最小宽度2

width:设置元素的宽度1

09. CSS定位属性(Positioning)

属性与描述

bottom: 设置定位元素下外边距边界与其包含块下边界之间的偏移²

clear: 规定元素的哪一侧不允许其他浮动元素¹

clip: 剪裁绝对定位元素²

cursor: 规定要显示的光标类型(形状)²

display: 规定元素应该生成的框的类型¹

float: 规定框是否应该浮动¹

left: 设置定位元素左外边距边界与其包含块左边界之间的偏移²

overflow: 规定当内容溢出元素框时发生的事情²

position: 规定元素定位类型²

right: 设置定位元素右外边距边界与其包含块右边之间的偏移²

top: 设置定位元素上外边距边界与其包含块上边之间的偏移²

vertical-align: 设置元素的垂直对其方式¹

visibility: 规定元素是否可见²

z-index: 设置元素的堆叠顺序

10. CSS表格属性(Table)

属性与描述

border-collapse:规定是否合并表格边框²

border-spacing:规定相邻单元格边框之间的距离²

caption-side:规定表格标题的位置²

empty-cells:规定是否显示表格中的空单元格上的边框和背景²

table-layout:设置用于表格的布局算法

W3cshool

Block大小调整

```
#main {  
  width: 600px;  
  margin: 0 auto;  
}
```

CSS

<div id="main">

设置块级元素的 `width` 可以防止它从左到右撑满整个容器。然后你就可以设置左右外边距为 `auto` 来使其水平居中。元素会占据你所指定的宽度，然后剩余的宽度会一分为二成为左右外边距。

唯一的问题是，当浏览器窗口比元素的宽度还要窄时，浏览器会显示一个水平滚动条来容纳页面。让我们再来改进下这个方案...

</div>

这样就不会充满整个容器了

max-width

```
#main {  
  max-width: 600px;  
  margin: 0 auto;  
}
```

CSS

这样能处理窗口小的时候

<div id="main">

在这种情况下使用 `max-width` 替代 `width` 可以使浏览器更好地处理小窗口的情况。这点在移动设备上显得尤为重要，调整下浏览器窗口大小检查下吧！

顺便提下，**所有的主流浏览器**包括IE7+在内都支持 `max-width`，所以放心大胆的用吧。

</div>

- Absolute

- absolute 是最棘手的position值。absolute 与 fixed 的表现类似，但是它不是相对于视窗而是相对于最近的“positioned”祖先元素。如果绝对定位（position属性的值为absolute）的元素没有“positioned”祖先元素，那么它是相对于文档的 body 元素，并且它会随着页面滚动而移动。记住一个“positioned”元素是指 position 值不是 static 的元素。

```
.relative {  
  position: relative;  
  width: 600px;  
  height: 400px;  
}  
.absolute {  
  position: absolute;  
  top: 120px;  
  right: 0;  
  width: 300px;  
  height: 200px;  
}
```

CSS

`<div class="relative">`

这个元素是相对定位的。如果它是 `position: static;`，那么它的绝对定位子元素会跳过它直接相对于body元素定位。

`<div class="absolute">`

这个元素是绝对定位的。它相对于它的父元素定位。

`</div>`

`</div>`

Position

```
.container {
  position: relative;
}
nav {
  position: absolute;
  left: 0px;
  width: 200px;
}
section {
  /* position is static by default */
  margin-left: 200px;
}
footer {
  position: fixed;
  bottom: 0;
  left: 0;
  height: 70px;
  background-color: white;
  width: 100%;
}
body {
  margin-bottom: 120px;
}
```

CSS

一个合格的前端工程师，能够看到css和html就想象出布局位置~

```
<nav class="container">
```

- Home
- Taco Menu
- Draft List
- Hours
- Directions
- Contact

```
</nav>
```

```
<section>
```

`section` 的 `margin-left` 样式确保了有足够的空间容纳 `nav` 元素。

```
</section>
```

```
<section>
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum, nisi lorem egestas odio, vitae scelerisque enim ligula venenatis dolor. Maecenas nisl est, ultrices nec congue eget, auctor vitae massa. Fusce luctus vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in odio. Praesent convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis dictum nisi, sed ullamcorper ipsum dignissim ac. In at libero sed nunc venenatis imperdiet sed ornare turpis. Donec vitae dui eget tellus gravida venenatis. Integer fringilla congue eros non fermentum. Sed dapibus pulvinar nibh tempor porta. Cras ac leo purus. Mauris quis diam velit.

```
</section>
```

```
<section>
```

注意观察当你调整浏览器窗口时发生了什么。效果很赞！

```
</section>
```

Inline Block

- Inline - block

```
.box2 {  
  display: inline-block;  
  width: 200px;  
  height: 100px;  
  margin: 1em;  
}
```

CSS

<div class="box2">
我是一个行内块!
</div>

<div class="box2">
我是一个行内块!
</div>

<div class="box2">
我是一个行内块!
</div>

<div class="box2">
我是一个行内块!
</div>

<div class="box2">
我是一个行内块!
</div>

<div class="box2">
我是一个行内块!
</div>

<div class="box2">
我是一个行内块!
</div>

<div class="box2">
我是一个行内块!
</div>

<div class="box2">
我是一个行内块!
</div>

<div class="box2">
我是一个行内块!
</div>

<div>
这次我可没有用 `clear` 。太棒了!

</div>

- 你可以使用 inline-block 来布局。有一些事情需要你牢记：
 - vertical-align 属性会影响到 inline-block 元素，你可能会把它的值设置为 top.
 - 你需要设置每一列的宽度 如果HTML源代码中元素之间有空格，那么列与列之间会产生空隙.

```
nav {  
  display: inline-block;  
  vertical-align: top;  
  width: 25%;  
}  
.column {  
  display: inline-block;  
  vertical-align: top;  
  width: 75%;  
}
```

CSS

- “响应式设计 (Responsive Design)” 是一种让网站针对不同的浏览器和设备 “呈现” 不同显示效果的策略，这样可以让网站在任何情况下显示的很棒！
- 媒体查询是做此事所需的最强大的工具。让我们使用百分比宽度来布局，然后在浏览器变窄到无法容纳侧边栏中的菜单时，把布局显示成一行：

```
@media screen and (min-width:600px) {  
  nav {  
    float: left;  
    width: 25%;  
  }  
  section {  
    margin-left: 25%;  
  }  
}  
@media screen and (max-width:599px) {  
  nav li {  
    display: inline;  
  }  
}
```

CSS

调整窗口大小，你会发现显示更智能

外边距叠加是一个相当简单的概念。但是，在实践中对网页进行布局时，它会造成许多混淆。简单的说，当两个或更多个垂直边距相遇时，它们将形成一个外边距。这个外边距的高度等于两个发生叠加的外边距的高度中的较大者。但是注意只有普通文档流中块框的垂直外边距才会发生外边距叠加。行内框、浮动框或绝对定位框之间的外边距不会叠加。

一般来说，垂直外边距叠加有三种情况：

- A. 元素自身叠加 当元素没有内容（即空元素）、内边距、边框时，它的上下边距就相遇了，即会产生叠加（垂直方向）。当为元素添加内容、内边距、边框任何一项，就会取消叠加。
- B. 相邻元素叠加 相邻的两个元素，如果它们的上下边距相遇，即会产生叠加。
- C. 包含（父子）元素叠加 包含元素的外边距隔着父元素的内边距和边框，当这两项都不存在的时候，父子元素垂直外边距相邻，产生叠加。添加任何一项即会取消叠加。

到这里呢..

上岸

你的html功夫也就差不多了

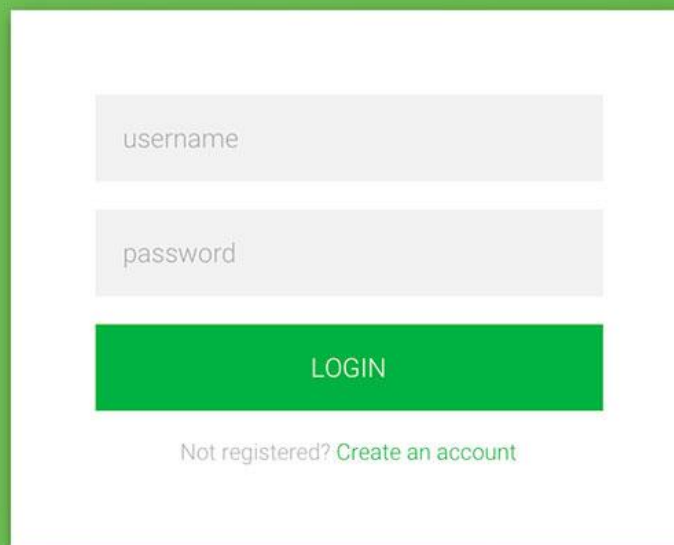


有几个作业:



<https://www.w3schools.com/html/default.asp>

有几个作业:



A login form centered on a green background. The form is a white rectangle containing two input fields and a button. The first input field is labeled 'username' and the second is labeled 'password'. Below these fields is a green button with the text 'LOGIN'. At the bottom of the form, there is a link that says 'Not registered? Create an account'.

username

password

LOGIN

Not registered? [Create an account](#)

有几个作业:

- 请根据下面的HTML和CSS代码，画出布局示意图

```
<div id="page">
  <div class="main">
    <div class="sub"></div>
  </div>
  <div class="nav"></div>
</div>
<style type="text/css">
  #page { width: 520px; }
  .nav { width: 200px; float: right; }
  .main { width: 200px; float: left; padding-left: 110px; }
  .sub { width: 100px; float: left; margin: 10px 0 10px -100px; }
  .main { border: 1px solid #000; }
  .nav, .sub { border: 1px dashed #000; height: 300px; }
  .sub { height: 280px; }
</style>
```

Questions?



THANK YOU

The image features a horizontal banner. The left portion of the banner is a solid teal color, while the right portion is filled with a complex geometric pattern of overlapping triangles in various shades of yellow, orange, and light green. The text 'THANK YOU' is written in a bold, white, sans-serif font across the teal section.