13 | 浏览器：一个浏览器是如何工作的？（阶段四）

winter 2019-02-16

00:00 09:58
讲述：winter 大小：9.14M

你好，我是 winter。

我们书接上文。浏览器进行到这一步，我们已经给 DOM 元素添加了用于展现的 CSS 属性，接下来，浏览器的工作就是确定每一个元素的位置了。我们的基本原则仍然不变，就是尽可能流式地处理上一步骤的输出。

在构建 DOM 树和计算 CSS 属性这两个步骤，我们的产出都是一个一个的元素，但是在排版这个步骤中，有些情况下，我们就没法做到这样了。

尤其是表格相关排版、flex 排版和 grid 排版，它们有一个显著的特点，那就是子元素之间具有关联性。

基本概念

首先我们先来介绍一些基本概念，使你可以感性地认识一下我们平常说的各种术语。

“排版”这个概念最初来自活字印刷，是指我们把一个一个的铅字根据文章顺序，放入板框当中的步骤，排版的意思是确定每一个字的位置。

在现代浏览器中，仍然借用了这个概念，但是排版的内容更加复杂，包括文字、图片、图形、表格等等，我们把浏览器确定它们位置的过程，叫作排版。

浏览器最基本的排版方案是**正常流排版**，它包含了顺次排布和折行等规则，这是一个跟我们提到的印刷排版类似的排版方案，也跟我们平时书写文字的方式一致，所以我们把它叫做正常流。

浏览器的文字排版遵循公认的文字排版规范，文字排版是一个复杂的系统，它规定了行模型和文字在行模型中的排布。行模型规定了行顶、行底、文字区域、基线等对齐方式。（你还记得小时候写英语的英语本吗？英语本上的四条线就是一个简单的行模型）

此外，浏览器支持不同语言，因为不同语言的书写顺序不一致，所以浏览器的文本排版还支持双向文字系统。

浏览器又可以支持元素和文字的混排，元素被定义为占据长方形的区域，还允许边框、边距和留白，这个就是所谓的**盒模型**。

在正常流的基础上，浏览器还支持两类元素：绝对定位元素和浮动元素。

- 绝对定位元素把自身从正常流抽出，直接由 top 和 left 等属性确定自身的位置，不参加排版计算，也不影响其它元素。绝对定位元素由 position 属性控制。
- 浮动元素则是使得自己在正常流的位置向左或者向右移动到边界，并且占据一块排版空间。浮动元素由 float 属性控制。

除了正常流，浏览器还支持其它排版方式，比如现在非常常用的 flex 排版，这些排版方式由外部元素的 display 属性来控制（注意，display 同时还控制元素在正常流中属于 inline 等级还是 block 等级）。

正常流文字排版

我们会在 CSS 部分详细介绍正常流排版的行为，我们这里主要介绍浏览器中的正常流。正常流是唯一一个文字和盒混排的排版方式，我们先从文字来讲起。

要想理解正常流，我们首先要回忆一下自己如何在纸上写文章。

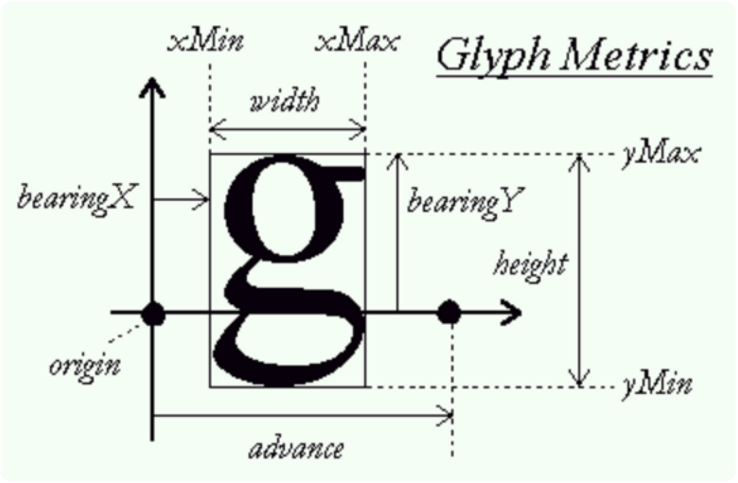
首先，纸是有固定宽度的，虽然纸有固定高度，但是我们可以通过下一页纸的方式来接续，因此我们不存在写不下的场景。

我们书写文字的时候，是从左到右依次书写，每一个字跟上一个字都不重叠，文字之间有一定间距，当写满一行时，我们换到下一行去继续写。

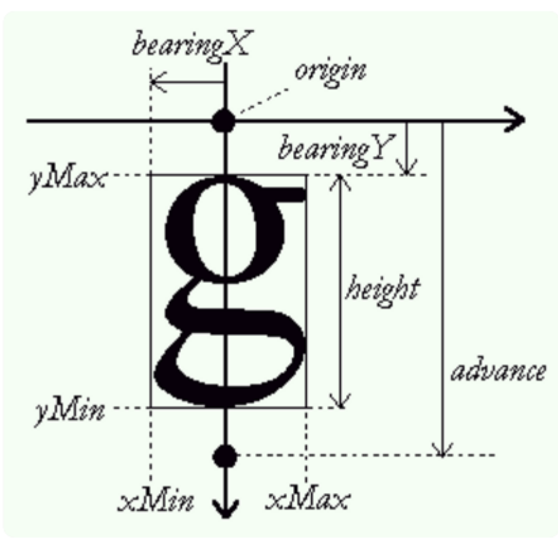
书写中文时，文字的上、下、中轴线都对齐，书写英文时，不同字母的高度不同，但是有一条基线对齐。

实际上浏览器环境也很类似。但是因为浏览器支持改变排版方向，不一定是从左到右从上到下，所以我们把文字依次书写的延伸方向称为主轴或者主方向，换行延伸的方向，跟主轴垂直交叉，称为交叉轴或者交叉方向。

我们一般会从某个字体文件中获取某个特定文字的相关信息。我们获取到的信息大概类似下面：



纵向版本：



这两张图片来自著名开源字体解析库 freetype，实际上，各个库对字体的理解大同小异，我们注意一下，advance 代表每一个文字排布后在主轴上的前进距离，它跟文字的宽 / 高不相等，是字体中最重要的属性。

除了字体提供的字形本身包含的信息，文字排版还受到一些 CSS 属性影响，如 line-height、letter-spacing、word-spacing 等。

在正常流的文字排版中，多数元素被当作长方形盒来排版，而只有 display 为 inline 的元素，是被拆成文本来排版的（还有一种 run-in 元素，它有时作为盒，有时作为文字，不太常用，这里不详细讲了）。

display 值为 inline 的元素中的文字排版时会被直接排入文字流中，inline 元素主轴方向的 margin 属性和 border 属性（例如主轴为横向时的 margin-left 和 margin-right）也会被计算进排版前进距离当中。

注意，当没有强制指定文字书写方向时，在左到右文字中插入右到左向文字，会形成一个双向文字盒，反之亦然。

这样，即使没有元素包裹，混合书写方向的文字也可以形成一个盒结构，我们在排版时，遇到这样的双向文字盒，会先排完盒内再排盒外。

正常流中的盒

在正常流中，display 不为 inline 的元素或者伪元素，会以盒的形式跟文字一起排版。多数 display 属性都可以分成两部分：内部的排版和是否 inline，带有 inline- 前缀的盒，被称作行内级盒。

根据盒模型，一个盒具有 margin、border、padding、width/height 等属性，它在主轴方向占据的空间是由对应方向的这几个属性之和决定的，而 vertical-align 属性决定了盒在交叉轴方向的位置，同时也会影响实际行高。

所以，浏览器对行的排版，一般是先行内布局，再确定行的位置，根据行的位置计算出行内盒和文字的排版位置。

块级盒比较简单，它总是单独占据一整行，计算出交叉轴方向的高度即可。

绝对定位元素

position 属性为 absolute 的元素，我们需要根据它的包含块来确定位置，这是完全跟正常流无关的一种独立排版模式，逐层找到其父级的 position 非 static 元素即可。

浮动元素排版

float 元素非常特别，浏览器对 float 的处理是先排入正常流，再移动到排版宽度的最左 / 最右（这里实际上是主轴的最前和最后）。

移动之后，float 元素占据了一块排版的空间，因此，在数行之内，主轴方向的排版距离发生了变化，直到交叉轴方向的尺寸超过了浮动元素的交叉轴尺寸范围，主轴排版尺寸才会恢复。
float 元素排布完成后，float 元素所在的行需要重新确定位置。

其它的排版

CSS 的每一种排版都有一个很复杂的规定，实际实现形式也各不相同。比如如 flex 排版，支持了 flex 属性，flex 属性将每一行排版后的剩余空间平均分配给主轴方向的 width/height 属性。浏览器支持的每一种排版方式，都是按照对应的标准来实现的。

总结

这一部分，我们以正常流为主，介绍了浏览器的排版基本概念及一些算法。这里，我主要介绍了正常流中的文字排版、正常流中的盒、绝对定位元素、浮动元素排版这几种情况。最后，我还简单介绍了一下 flex 排版。这属于进阶版的排版方式了，你可以了解一下。

你平时喜欢使用方式排版呢，欢迎留言告诉我。