Web前端开发

第12章 布局

阮晓龙

13938213680 / rxl@hactcm.edu.cn
http://web.book.51xueweb.cn
http://www.51xueweb.cn

河南中医药大学管理科学与工程学科

2018.5

本章主要内容

- 定位与布局的基本属性
- 多列布局
- 盒布局
- 自适应布局
- 案例：网页布局

1.定位与布局的基本属性

- 布局是指对网页中各个构成要素的合理编排，是呈现页面内容的基础。合理的布局将有效的提高页面的可读性，提升用户体验。
- 通过定位与布局的基本属性，可以确定元素的位置，并实现页面多种多样的布局。

1.定位与布局的基本属性

表 12-1　定位与布局的基本属性

属性	描述
margin	在一个声明中设置所有的外边距属性
margin-bottom	设置元素的下外边距
margin-left	设置元素的左外边距
margin-right	设置元素的右外边距
margin-top	设置元素的上外边距
padding	在一个声明中设置所有的内边距属性
padding-bottom	设置元素的下内边距
padding-left	设置元素的左内边距
padding-right	设置元素的右内边距
padding-top	设置元素的上内边距
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
clear	规定元素的哪一侧不允许有其他浮动元素

1.定位与布局的基本属性

表 12-1　定位与布局的基本属性

属性	描述
clip	剪裁绝对定位元素
cursor	规定要显示的光标的类型（形状）
display	规定元素应该生成的框的类型
float	规定框是否应该浮动
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
overflow	规定当内容溢出元素框时发生的事情
position	规定元素的定位类型
right	设置定位元素右外边距边界与其包含块右边界之间的偏移
top	设置定位元素上外边距边界与其包含块上边界之间的偏移
vertical-align	设置元素的垂直对齐方式
visibility	规定元素是否可见
z-index	设置元素的堆叠顺序

1.定位与布局的基本属性

- 外边距属性
  - margin属性可以设定元素的所有外边距。
    - 该属性可以通过1-4个正负值进行设置，其值可以为长度、百分比等单位，也可从父元素继承外边距。
  - 可以通过margin-top、margin-right、margin-bottom、margin-left属性分别设置。

1.定位与布局的基本属性

- 具体语法：

```
/*快速定义盒子的外边距都为 10 像素*/
margin:10px;
/*定义上下外边距为 5 像素，左右外边距为 10 像素*/
margin:5px 10px;
/*定义上外边距为 5 像素，左右外边距为 10 像素，下外边距为 15 像素*/
margin:5px 10px 15px;
/*定义上外边距为 5 像素，右外边距为 10 像素，下外边距为 15 像素，左边外距为 20 像素*/
margin:5px 10px 15px 20px;
/*单独定义上外边距为 5 像素*/
margin-top:5px;
/*单独定义左外边距为 10 像素*/
margin-left:10px;
/*单独定义右外边距为 15 像素*/
margin-right:15px;
/*单独定义下外边距为 20 像素*/
margin-bottom:20px;
```

1.定位与布局的基本属性

- 行内元素的外边距
  - 当行内元素定义外边距时，只能看到左右外边距对布局的影响，但是上下外边距犹如不存在一般，不会对周围元素产生影响。
- 块级元素的外边距
  - 对于块级元素来说，外边距都能够很好的被解析。可以用“display”属性改变元素的表现形式来保证元素对外边距的支持。

1.定位与布局的基本属性

- 内边距属性
  - padding属性可以设定元素的所有内边距。该属性可以通过1-4个正负值进行设置，其值可以为像素、百分比等单位，也可从父元素继承外边距。
  - 可以通过padding-top、padding-right、padding-bottom、padding-left属性分别设置。

1.定位与布局的基本属性

- 具体语法：

```
/*快速定义盒子的内边距都为 10 像素*/
padding:10px;
/*定义上下内边距为 5 像素，左右内边距为 10 像素*/
padding:5px 10px;
/*定义上内边距为 5 像素，左右内边距为 10 像素，下内边距为 15 像素*/
padding:5px 10px 15px;
/*定义上内边距为 5 像素，右内边距为 10 像素，下内边距为 15 像素，左内边距为 20 像素*/
padding:5px 10px 15px 20px;
/*单独定义上内边距为 5 像素*/
padding-top:5px;
/*单独定义左内边距为 10 像素*/
padding-left:10px;
/*单独定义右内边距为 15 像素*/
padding-right:15px;
/*单独定义下内边距为 20 像素*/
padding-bottom:20px;
```

1.定位与布局的基本属性

- float
  - 网页的布局主要通过float属性来实现，float 属性定义元素在哪个方向浮动。
  - 属性值有以下四种情况：
    - left定义向左浮动。
    - right定义向右浮动。
    - none为float属性的默认值，表示元素不浮动，并会显示其在页面中出现的位置。
    - inherit规定应该从父元素继承float属性的值。

1.定位与布局的基本属性

- 浮动元素特性
  - 当一个元素被设置为浮动元素后，元素本身的属性会发生一些改变，具体如下：
    - 空间的改变。
    - 位置的改变。
    - 布局环绕。
  - 当元素被定义为浮动显示时，它会自动成为一个块状元素，相当于定义了“display:block;”。但是块级元素会自动伸张宽度，占据一行位置，且块级元素会附加换行符，所以在同一行内只能显示一个块级元素。而浮动元素虽然拥有块级元素的特性，但是它并没有上述表现，这时它更像行内元素那样收缩显示。

现场演示：

- 案例12-01：浮动元素的空间
- 案例12-02：浮动元素的位置
- 案例12-03：浮动元素的环绕
- 案例12-04：图文混排

1.定位与布局的基本属性

- clear属性规定元素的哪一侧不允许存在其他浮动元素，属性值有五种情况：
  - left、right、both
    - left、right、both分别规定在左侧不允许浮动元素、在右侧不允许浮动元素和在左右两侧均不允许浮动元素。
  - none
    - none为clear元素的默认值，允许浮动元素出现在两侧。
  - inherit
    - inherit规定应该从父元素继承clear属性的值。

1.定位与布局的基本属性

- 元素浮动以后，其所在的位置会被下方不浮动的元素填充掉，而有些时候这样的填充会破坏页面布局，clear元素可以解决这个问题。在不浮动元素中添加与浮动元素float属性值相同的clear属性值，会使不浮动元素显示在浮动元素的下边距边界之下。
- 浮动元素也可以添加clear属性，添加的clear属性的属性值只有和float属性的属性值相同时才能起作用，即当元素向左浮动时只能清除元素的左浮动，而不能将属性值设为清除右浮动。

现场演示：

- 案例12-05：清除浮动

1.定位与布局的基本属性

- 定位坐标值
  - 为了灵活的定位页面元素，CSS定义了4个坐标属性：top、right、bottom和left。通过这些属性的联合使用，可包含块的4个内顶角来定位元素在页面中的位置。
    - top属性表示定位元素顶边外壁到包含块元素顶部内壁的距离。
    - right属性表示定位元素右边外壁到包含块元素右侧内壁的距离。
    - left属性表示定位元素左边外壁到包含块元素左侧内壁的距离。
    - bottom属性表示定位元素底边外壁到包含块元素底部内壁的距离。

1.定位与布局的基本属性

- position
  - position属性用于确定元素的位置，该属性可将图片放置到任何位置，也可以使导航始终显示于页面最上方。CSS的定位核心正是基于这个属性实现的。属性值有五种情况。
  - static
    - static为position的默认属性值，没有定位，元素出现在正常流中（忽略 top, bottom, left, right 或者 z-index 声明）。
    - 任何元素在默认的状态下都会以静态定位来确定自己的位置，所以当没有定义position时，并不说明该元素没有自己的位置，它会遵循默认值显示为静态位置。在静态位置下，开发人员无法通过坐标值（top、bottom、left和right）来改变它的位置。

1.定位与布局的基本属性

- position
  - absolute
    - absolute可用于生成绝对定位的元素，相对于static定位以外的第一个父元素进行定位。元素的位置通过left、top、right、bottom属性进行设置。
    - 当position属性取值为absolute时，程序就会把元素从文档流中拖出来，根据某个参照物坐标来确定显示位置。绝对定位是网页精准定位的基本方法。如果结合left、right、top、bottom坐标属性进行精确定位，结合z-index属性排列元素覆盖顺序，同时通过clip和visiblity属性裁切、显示或隐藏元素对象或部分区域，就可以设计出丰富多样的网页布局效果。

现场演示：

- 案例12-06：绝对定位

1.定位与布局的基本属性

- position
  - fixed
    - fixed可用于生成固定定位的元素，相对于浏览器窗口进行定位。元素的位置通过top、right、bottom、left属性进行定义。
    - 固定定位是绝对定位的一种特殊形式，它是以浏览器作为参照物来定义网页元素的。如果定义某个元素固定显示而不受文档流的影响，也不受包含块的位置影响，它始终以浏览器窗口来定位自己的显示位置。不管浏览器的滚动条如何滚动，也不管浏览器窗口大小如何变化，该元素都会显示在浏览器窗口内。

现场演示：

- 案例12-07：fixed属性值

1.定位与布局的基本属性

- position
  - relative
    - relative可用于生成相对定位的元素，相对于其正常位置进行定位。例如，"left:20px"会向元素的左侧位置添加20像素。
    - 相对定位是一种折中的定位方法，是在静态定位和绝对定位之间取的一个平衡点。所谓相对定位就是使被应用元素不脱离文档流，却能通过坐标值以原始位置为参照物进行偏移。
  - inherit
    - inherit用于从父元素继承 position 属性的值。

现场演示：

- 案例12-08：相对定位

1.定位与布局的基本属性

- 定位层叠
  - CSS可通过z-index属性来排列不同定位元素之间的层叠顺序。
  - 该属性可以设置为任意的整数值，数值越大，所排列的顺序就越靠上（前）。

现场演示：

- 案例12-09：定位层叠

1.定位与布局的基本属性

- 定位与参照
  - 定位是网页布局的重中之重，为页面中每个构成要素找到它应该待的位置是页面布局的基础。
  - position属性专门用于页面布局，但因其复杂度较高很难被初学者掌握，下面会通过一个案例来讲解position属性中的定位与参照。
  - 在绝对定位的父级没有设定position属性时，将以浏览器左上角为参照点进行定位；当父级设定position属性值时，将以父级为参照点进行定位。

现场演示：

- 案例12-10：定位与参照

2.多列布局

- 使用float属性或position属性进行页面布局时有一个比较明显的缺点，就是多列的div元素间是各自独立的。
- 如果在第一列div元素中加入一些内容，将会使得两列元素底部不能对齐，多出一块空白的区域。这种情况在多列文章排版时显得极为明显。

2.多列布局

- 基本属性：

表 12-2　多列布局的基本属性

属性	描述
column-count	规定元素应该被分隔的列数
column-fill	规定如何填充列
column-gap	规定列之间的间隔
column-rule	设置所有 column-rule-*属性的简写属性
column-rule-color	规定列之间规则的颜色
column-rule-style	规定列之间规则的样式
column-rule-width	规定列之间规则的宽度
column-span	规定元素应该横跨的列数
column-width	规定列的宽度
columns	设置 column-width 和 column-count 的简写属性

2.多列布局

- columns是多列布局的基本属性。该属性可以同时定义列数和每列的宽度。相当于同时指定了column-width、column-count属性。目前，Webkit引擎支持-webkit-columns私有属性，Mozilla Gecko引擎支持-moz-columns私有属性。
- 具体语法：

```
div{
columns:100px 3;
-moz-columns:100px 3;          /* Firefox */
-webkit-columns:100px 3;       /* Safari 和 Chrome */
}
```

2.多列布局

- column-width属性可以定义单列显示的宽度。该属性可以与其他多列布局属性配合使用，也可以单独使用。
- 具体语法：

```
div{
column-width:100px;
-moz-column-width:100px;       /* Firefox */
-webkit-column-width:100px;    /* Safari 和 Chrome */
}
```

  - column-width可以与column-count属性配合使用，设定固定列数、列宽的布局效果，也可以单独使用，限制模块的单列宽度，当超出宽度时，则会自动以多列进行显示。

2.多列布局

- 具体语法：
  - column-count属性可以定义显示的列数，取值为大于0的整数。如果column-width和column-count属性没有明确的值，则默认为最大列数。
  - column-width、column-count这两个属性可以相互影响，指定的栏目宽度、栏目数并不是绝对的。
  - 当分栏内容所在容器的宽度大于column-width*column-count+间距时，有的浏览器会增加栏目数，有的浏览器会增加栏目宽度。

2.多列布局

- 具体语法：
  - column-gap属性可以定义两列之间的间距，其默认值为normal，用于规定列间间隔为一个常规的间隔。W3C建议的值是1em。
  - column-rule属性用于指定栏目之间的分割条。该属性可同时指定分割条的宽度、样式、颜色。
  - column-rule-width属性的值为一个长度值，用于指定栏目之间分割条的宽度。
  - column-rule-style属性用于设置分割条的线型。该属性支持的属性值有none、dotted、dashed、solid、double、groove、ridge、inset、outset，这些属性值与前面介绍的边框线型的各属性值的意义完全相同。
  - column-rule-color属性用于设置分隔条的颜色。

2.多列布局

- 具体语法：
  - 在报刊杂志中，经常会看到文章标题跨列居中显示。
  - column-span属性可以定义跨列显示，也可以设置单列显示，其属性值默认为1，适用于静态的、非浮动元素，代表只能在本栏中显示。
  - all属性值则表示横跨所有的列，并定位在列的Z轴上。

2.多列布局

- 具体语法：
  - column-fill属性可以定义栏目的高度是否统一。
  - 其属性值有两种情况：
    - auto属性值可以设置各列高度随其内容的变化而变化。
    - balance属性值是column-fill默认值，设置各列的高度根据内容最多的那一列的高度进行统一。

现场演示：

- 案例12-11：多列布局

3.盒布局

- CSS3引入了新的盒模型：弹性盒模型。
- 该模型决定一个盒子在其他盒子中的分布方式以及如何处理可用空间。使用该模型可以很轻松地创建自适应浏览器窗口的流动布局，或自适应字体大小的弹性布局。
- 传统的是基于HTML流在垂直方向上排列盒子，使用弹性盒布局可以规定特定的顺序，也可以将其反转。要开启弹性盒布局，只需设置盒子的display属性值为box（或inline-box）即可。
- 在CSS3中，除了多列布局之外，还可以用盒布局解决float属性或position属性布局存在的问题。

3.盒布局

- 基本属性：

表 12-3　盒布局的基本属性

属性	描述
column-count	规定元素应该被分隔的列数
box-align	规定如何对齐框的子元素
box-direction	规定框的子元素的显示方向
box-flex	规定框的子元素是否可伸缩
box-flex-group	将可伸缩元素分配到柔性分组
box-lines	规定当超出父元素框的空间时，是否换行显示
box-ordinal-group	规定框的子元素的显示次序
box-orient	规定框的子元素是否应水平或垂直排列
box-pack	规定水平框中的水平位置或者垂直框中的垂直位置

3.盒布局

- box-flex属性可将盒布局设置为弹性盒布局。
- Webkit引擎支持-webkit-box-flex私有属性，Mozilla Gecko引擎支持-moz-box-flex私有属性。
- 默认情况下，盒子不具备弹性，如果box-flex的属性值至少为1时，则变得富有弹性。如果盒子不具有弹性，它将尽可能的宽以使内容可见且没有任何溢出，其大小由width和height属性值，或者min-height、min-width、max-height、max-width属性值来决定。

3.盒布局

- 如果盒子是弹性的，其大小将按照下面的方式进行计算：
  - 具体大小声明（width、height、min-height、min-width、max-height、max-width）。
  - 父盒子的大小和所有余下的可利用的内部空间。
- 如果盒子没有任何大小声明，那么其大小将完全取决于父盒子的大小，其公式如下所示：

$$\text{子盒子的大小} = \text{父盒子的大小} \times \frac{\text{子盒子的}box-flex}{\text{所有子盒子的}box-flex\text{值的和}}$$

3.盒布局

- 如果一个或更多个盒子有一个具体的大小声明，那么其大小将计算到其中，余下的弹性盒子将按照上面原则分享剩下可利用的空间。

子盒子的大小 =

$$(\text{父盒子的大小} - \text{已定义子盒子大小}) \times \frac{\text{子盒子的}box-flex}{\text{所有子盒子的}box-flex\text{值的和}}$$

现场演示：

- 案例12-12：自适应宽度的弹性盒布局

3.盒布局

- 使用弹性盒布局时，可以使用box-ordinal-group属性来改变各元素的显示顺序。可在每个元素中加入box-ordinal-group属性，该属性使用一个表示序号的整数属性值，浏览器在显示的时候根据该序号从小到大显示这些元素。
- 目前Webkit引擎支持-webkit-box-ordinal-group私有属性，Mozilla Gecko引擎支持-moz-box-ordinal-group私有属性。

现场演示：

- 案例12-13：改变元素的显示顺序

3.盒布局

- box-direction可以简单地将多个元素的排列方向从水平方向修改为垂直方向，或者从垂直方向修改为水平方向。
- 目前Webkit引擎支持-webkit-box-direction私有属性，Mozilla Gecko引擎支持-moz-box-direction私有属性。
- 其属性值有三种情况：
  - normal：以默认方向显示子元素。
  - reverse：以反方向显示子元素。
  - inherit：设定从父元素继承box-direction属性的值。

现场演示：

- 案例12-14：改变元素排列方向
- 从案例中可以看到盒子并没有自适应于整个有边框的div，box-flex属性可以设置弹性的盒布局，使其充满整个div。

现场演示：

- 案例12-15：使用弹性盒布局消除空白

3.盒布局

- 如果box-flex属性只对一个元素使用，可以使其宽度、高度自动扩大，让浏览器或容器中所有元素的总宽度/总高度等于浏览器或容器的宽度/高度。在CSS 3中也可以对多个元素使用box-flex属性。
- 案例12-16中box-flex的属性值设为1，如果把box-flex的属性值设为其他的整数，例如2，页面结构将发生变化。
- 案例12-16中多个元素设置box-flex的属性值均为1，元素将等分空白区域。使用浏览器的开发者模式对案例12-16进行调试，将box-flex的属性值进行调整，元素将按比例填充空白区域。

现场演示：

- 案例12-16：多个元素使用box-flex值
- 讨论：使用浏览器调试工具进行参数调试，以观察响应的变化。

3.盒布局

- 使用盒布局时，可以使用box-pack属性及box-align属性来指定元素中文字、图像及子元素水平方向或垂直方向的对齐方式。
- 目前Webkit引擎支持-webkit-box-pack和-webkit-box-align私有属性，Mozilla Gecko引擎支持-moz-pack和-webkit-box-align私有属性。

3.盒布局

- box-pack属性可以用于设置子容器在水平轴上的空间分配方式，它共有四种可能值：start、end、justify、center。
- 具体含义：
  - start：所有子容器都分布在父容器的左侧，右侧留空。
  - end：所有子容器都分布在父容器的右侧，左侧留空。
  - justify：所有子容器平均分布（默认值）。
  - center：平均分配父容器剩余的空间（能压缩子容器的大小，并且有全局居中的效果）。

3.盒布局

- box-align 属性用于管理子容器在竖轴上的空间分配方式，共有五个属性值：start、end、center、baseline、stretch。
- 具体含义：
  - start：子容器从父容器顶部开始排列。
  - end：子容器从父容器底部开始排列。
  - center：子容器横向居中。
  - baseline：所有子容器沿同一基线排列。
  - stretch：所有子容器和父容器保持同一高度（默认值）。

现场演示：

- 案例12-17：定位布局使图片居中
- 案例12-18：盒布局使图片居中

3.盒布局

- 通过使用传统的浮动布局、CSS3新增的多列布局和CSS3新增的盒布局来实现简单的三列布局，进行布局方式对比说明。

现场演示：

- 案例12-19：浮动布局
- 案例12-20：盒布局
- 案例12-21：多列布局

3.盒布局

- 使用float属性或position属性进行页面布局时各列的div元素间是独立的，不能统一的定义div的各种属性。
- 盒布局与多列布局的区别在于使用多列布局时，各列宽度必须是相等的，在指定每列宽度时，也只能为所有列指定一个统一的宽度。
- 使用多列布局时，也不可能具体指定什么列显示什么内容，因此比较适用于显示文章内容，不适合用于安排整个网页中各个元素组成的网页结构。

4.自适应布局

- 2010年，Ethan Marcotte提出了“自适应网页设计”（Responsive Web Design）这个名词，指可以自动识别屏幕宽度、并做出相应调整的网页设计。
- 自适应布局的特点是分别为不同的屏幕分辨率定义布局，即创建多个静态布局，每个静态布局对应一个屏幕分辨率范围。改变屏幕分辨率可以切换不同的静态局部（页面元素位置发生改变），但在每个静态布局中，页面元素不随窗口大小的调整发生变化。

4.自适应布局

- 基本属性：

表 12-4　自适应布局的基本属性

属性	描述
horizontal	规定水平方向布局
vertical	规定垂直方向布局
DHorizontalLayoutZero	规定水平方向 L、R、W、C 清零
DVerticalLayoutZero	规定垂直方向 T、B、H、C 清零
DLayoutZero	规定 Layout L、R、W、T、B、H、C 清零
DHorizontalLayoutFill	规定水平方向塞满
DVerticalLayoutFill	规定垂直方向塞满
DLayoutFill	规定塞满

4.自适应布局

- 首先，在网页代码的头部，加入一行viewport元标签。
  <meta name="viewport" content="width=device-width, initial-scale=1">
- viewport是网页默认的宽度和高度，上面这行代码的意思是：网页宽度默认等于屏幕宽度（width=device-width），原始缩放比例（initial-scale=1）为1.0，即网页初始大小占屏幕面积的100%。
- 所有主流浏览器都支持这个设置，包括IE9。对于那些老式浏览器（主要是IE6、7、8），需要使用css3-mediaqueries.js。

```
<!--[if lte IE 9]>
<script src="http://css3-mediaqueries-js.googlecode.com/svn/trunk/css3-mediaqueries.js"></script>
<![endif]-->
```

4.自适应布局

- 由于网页会根据屏幕宽度调整布局，所以不能使用绝对宽度的布局，也不能使用具有绝对宽度的元素。CSS代码不能指定像素宽度，只能指定百分比宽度。

```
.css{
width:xxxx px;
} /*不能指定像素宽度*/
.css{
width: xx%;
} /*只能指定百分比宽度*/
```

4.自适应布局

- 字体也不能使用绝对大小（px），而只能使用相对大小（em），字体大小是页面默认大小的100%，即16像素，h1的大小是默认大小的1.5倍，即24像素（24/16=1.5），small元素的大小是默认大小的0.875倍，即14像素（14/16=0.875）。

```
. body {
font: normal 100% Helvetica, Arial, sans-serif;
} /*不能使用绝对大小，只能使用相对大小，字体默认大小 16 像素*/
.h1 {
font-size: 1.5em;
} /*h1 元素大小是默认大小的 1.5 倍，即 16 像素*/
.small {
font-size: 0.875em;
} /* small 元素的大小是默认大小的 0.875 倍，即 14 像素*/
```

4.自适应布局

- “流动布局”的含义是，各个区块的位置都是浮动的，不是固定不变的。

```
.main{
float: right;
width: 70%;
 }
.leftBar {
float: left;
width: 25%;
}
```

- float的好处是，如果宽度太小，放不下两个元素，后面的元素会自动滚动到前面元素的下方，不会在水平方向overflow（溢出），避免了水平滚动条的出现。

4.自适应布局

- “自适应网页设计”的核心，就是CSS3引入的Media Query模块，其自动探测屏幕宽度，然后加载相应的CSS文件。
- 如果屏幕宽度小于400像素（max-device-width: 400px），就加载tinyScreen.css文件，代码如下：

```
<link rel="stylesheet" type="text/css"media="screen and (max-device-width: 400px)" href="tinyScreen.css" >
```

- 如果屏幕宽度在400像素到600像素之间，则加载smallScreen.css文件，代码如下：

```
<link rel="stylesheet" type="text/css"media="screen and (min-width: 400px) and (max-device-width: 600px)" href="smallScreen.css">
```

- 除了用html标签加载CSS文件，还可以在现有CSS文件中加载：

```
@import url("tinyScreen.css") screen and (max-device-width: 400px);
```

4.自适应布局

- 同一个CSS文件中，也可以根据不同的屏幕分辨率，选择应用不同的CSS规则。

```
.column {
float: none;
width:auto;
}
#sidebar {
display:none;
}
}
```

- 如果屏幕宽度小于400像素，则column块取消浮动（float:none）、宽度自动调节（width:auto）、sidebar块不显示（display:none）。

4.自适应布局

- 除了布局和文本，“自适应网页设计”还必须实现图片的自动缩放，这只要一行CSS代码：

```
img {
max-width: 100%;
}
```

- 由于老版本的IE不支持max-width，所以需要写成：

```
img {
width: 100%;
}
```

- 此外windows平台缩放图片时，可能会出现失真现象，这是可以使用IE的专用命令，或者Ethan Marcotte的imgSizer.js：

```
img {
-ms-interpolation-mode: bicubic;
}
```

```
i addLoadEvent(function() {
var imgs = document.getElementById("content").getElementsByTagName("img");
imgSizer.collate(imgs);
});
```

现场演示：

- 案例12-22：自适应布局

5.案例：网页布局

- Web前端开发并不仅仅是代码层面的编写，还需要对页面内容进行合理的编排，用更好的方式呈现页面内容。
- 综合使用五种布局方式，对一个完整的网页进行布局，了解网页的结构与信息展示方式。

现场演示：

- 案例12-23：网页布局

Thanks.