CSS3 Tutorial
《CSS3 教程》

waylau

Published
with GitBook

目錄

CSS3 Tutorial 《CSS3 教程》

CSS3 Tutorial takes you to learn CSS3 step by step with a large number of samples. There is also a GitBook version of the book: http://www.gitbook.com/book/waylau/css3-tutorial. Let's READ!

CSS3 Tutorial 是一本关于 CSS3 的开源书。利用业余时间写了本书，图文并茂，用大量实例带你一步一步走进 CSS3 的世界。如有疏漏欢迎指正，欢迎提问。感谢您的参与！

Get Started 如何开始阅读

选择下面入口之一：

- https://github.com/waylau/css3-tutorial/ 的 SUMMARY.md（源码）
- http://waylau.gitbooks.io/css3-tutorial/ 点击 Read 按钮（同步更新，国内访问速度一般）

Code 源码

书中所有示例源码，移步至https://github.com/waylau/css3-tutorial `samples` 目录下

Issue 意见、建议

如有勘误、意见或建议欢迎拍砖 https://github.com/waylau/css3-tutorial/issues

Contact 联系作者:

- Blog: waylau.com
- Gmail: waylau521(at)gmail.com
- Weibo: waylau521
- Twitter: waylau521
- Github : waylau

介绍

什么是CSS

CSS 是 Cascading Style Sheet（层叠样式表）的缩写。是用于（增强）控制网页样式并允许将样式信息与网页内容分离的一种标记性语言。CSS 不需要编译,可以直接由浏览器执行(属于浏览器解释型语言)。

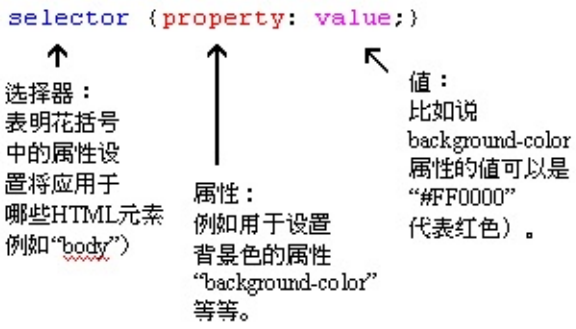
历史

- CSS 最早被提议是在1994年；
- 最早被浏览器支持是1996年；
- 1996年 W3C 正式推出了CSS1；
- 1998年 W3C 正式推出了CSS2；
- CSS2.1 是 W3C 现在正在推荐使用的；
- CSS3 现在还处于开发中；
- CSS 3 在包含了所有 CSS 2 所支持的基础上更有所改进，所以不必担心兼容问题。

CSS 支持多种设备，例如手机、电视、打印机、幻灯片等。但是 CSS 在浏览器上得到了更好的推广。

语法

selector {property: value;}



引入方式

三种方式将样式表加入您的网页：

内联方式 Inline Styles

内联定义即是在对象的标记内使用对象的 style 属性定义适用其的样式表属性。 示例代码：

```
<p style="color:#f00">这一行的字体颜色将显示为红色</p>
```

内部样式块对象 Embedding a Style Block

你可以在你的 HTML 文档的 `<head>` 标记里插入一个 `<style>` 块对象。 示例代码：

```
<style>
    .test2 {
        color: #000;
    }
</style>
```

外部样式表 Linking to a Style Sheet

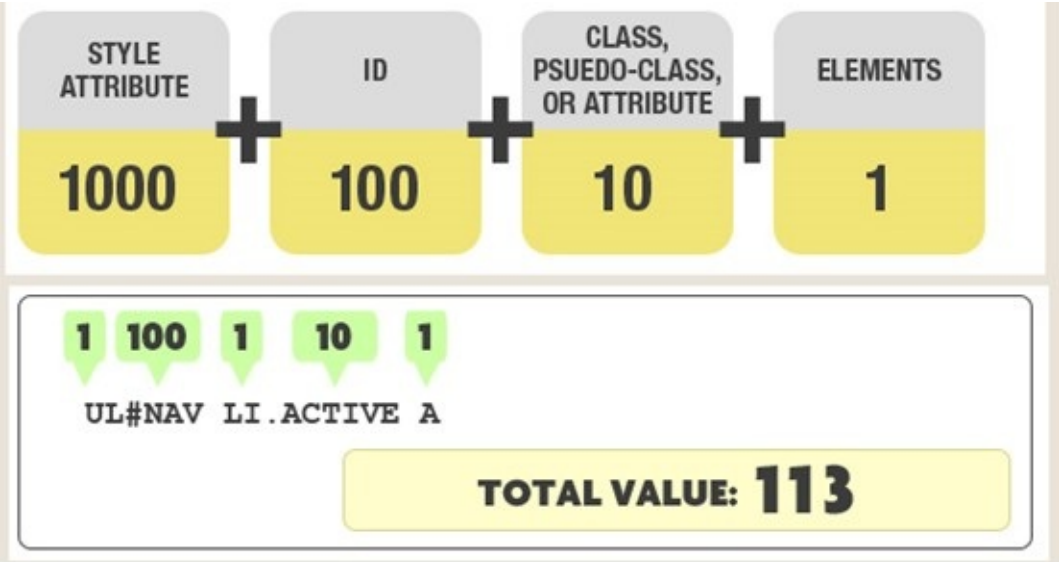
你可以先建立外部样式表文件 `*.css`，然后使用 HTML 的 link 对象。或者使用 `@import` 来引入。 示例代码：

```
<!-- Use link elements -->
<link rel="stylesheet" href="core.css">

<!-- Use @imports -->
<style>
  @import url("more.css");
</style>
```

注意：在实际开发中，推荐使用 HTML 的 link 对象来引入。详细内容可以参见 http://www.waylau.com/css-code-guide/#css-miscellaneous

选择器权重



权重主要分为 4 个等级：

- 第一等：代表内联样式，如: `style=""`，权值为1000
- 第二等：代表ID选择器，如： `#content`，权值为100
- 第三等：代表类，伪类和属性选择器，如 `.content`，权值为10
- 第四等：代表类型选择器和伪元素选择器，如 `div p`，权值为1

例子如下

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>选择器权重 | www.waylau.com</title>
    <meta name="description" content="选择器权重">
    <meta name="author" content="Way Lau, www.waylau.com"/>
    <meta name="viewport" content="width=device-width">
    <link rel="shortcut icon" href="/favicon.ico">

    <style type="text/css">
        #redP p {
            /* 权值 = 100+1=101 */
            color: #F00; /* 红色 */
        }

        #redP .red em {
            /* 权值 = 100+10+1=111 */
            color: #00F; /* 蓝色 */

        }

        #redP p span em {
            /* 权值 = 100+1+1+1=103 */
            color: #FF0; /*黄色*/
        }
    </style>
</head>
<body>
<div id="redP">
    <p class="red">red
        <span><em>em red</em></span>
    </p>

    <p>red</p>
</div>
</body>
</html>
```

最终页面效果如下：

优先级

遵循如下法则：

- 选择器都有一个权值，权值越大越优先；
- 当权值相等时，后出现的样式表设置要优于先出现的样式表设置；
- 创作者的规则高于浏览者：即网页编写者设置的 CSS 样式的优先权高于浏览器所设置的样式；
- 继承的 CSS 样式不如后来指定的 CSS 样式；
- 在同一组属性设置中标有 `!important` 规则的优先级最大

例子如下：

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>!important 用法 | www.waylau.com</title>
    <meta name="description" content="!important 用法">
    <meta name="author" content="Way Lau, www.waylau.com"/>
    <meta name="viewport" content="width=device-width">
    <link rel="shortcut icon" href="/favicon.ico">

    <style>
        .test {
            color: #f00 !important;
            color: #000;
        }

        .test2 {
            color: #f00 !important;
        }

        .test2 {
            color: #000;
        }

        .test3 {
            color: #000;
        }

        .test3 {
            color: #f00;
        }
    </style>
</head>
<body>
<div class="test">同一条样式内，!important 优先级高</div>
<div class="test2">在分散的样式条目内，!important 优先级高</div>
<div class="test3">没有被覆盖</div>
</body>
</html>
```

源码

本文中所用例子源码参见 https://github.com/waylau/css3-tutorial 中 `samples` 目录下的 important.html、priority_rules.html

参考

- http://www.nowamagic.net/csszone/css_SeletorPriorityRules.php
- http://www.cnblogs.com/xugang/archive/2010/09/24/1833760.html

动画

CSS3，我们可以创建动画，它可以取代许多网页动画图像，Flash 动画，和 Javascripts。

CSS3 @keyframes 规则

要创建CSS3动画，你将不得不了解 `@keyframes` 规则。

`@keyframes` 规则是用来创建动画。 `@keyframes` 规则内指定一个 CSS样式和动画将逐步从目前的样式更改为新的样式。

注意：Internet Explorer 10、Firefox 以及 Opera 支持 `@keyframes` 规则和 animation 属性。Chrome 和 Safari 需要前缀 -webkit-。

CSS3 动画

当在 `@keyframe` 创建动画，把它绑定到一个选择器，否则动画不会有任何效果。

指定至少这两个 CSS3 的动画属性绑定向一个选择器：

- 规定动画的名称
- 规定动画的时长

例子：

```
#animated_div {
    animation: animated_div 5s infinite;
    -moz-animation: animated_div 5s infinite;
    -webkit-animation: animated_div 5s infinite;
}
```

CSS3动画是什么？

- 动画是使元素从一种样式逐渐变化为另一种样式的效果。
- 您可以改变任意多的样式任意多的次数。
- 请用百分比来规定变化发生的时间，或用关键词 "from" 和 "to"，等同于 0% 和 100%。
- 0% 是动画的开始，100% 是动画的完成。
- 为了得到最佳的浏览器支持，您应该始终定义 0% 和 100% 选择器。

例子：

```
@keyframes animated_div {
    0% {
        left: 0px;
    }
    20% {
        left: 50px;
        background-color: green;
    }
    40% {
        left: 140px;
        background-color: red;
    }
    60% {
        left: 280px;
        background-color: yellow;
    }
    80% {
        left: 425px;
        background-color: blue;
    }
    100% {
        left: 0px;
        background-color: pink;
    }
}
```

常用属性

属性	描述	CSS
@keyframes	规定动画。	3
animation	所有动画属性的简写属性，除了 animation-play-state 属性。	3
animation-name	规定 @keyframes 动画的名称。	3
animation-duration	规定动画完成一个周期所花费的秒或毫秒。默认是 0。	3
animation-timing-function	规定动画的速度曲线。默认是 "ease"。	3
animation-delay	规定动画何时开始。默认是 0。	3
animation-iteration-count	规定动画被播放的次数。默认是 1。	3
animation-direction	规定动画是否在下一周期逆向地播放。默认是 "normal"。	3
animation-play-state	规定动画是否正在运行或暂停。默认是 "running"。	3

源码

本文中所用例子源码参见 https://github.com/waylau/css3-tutorial 中 `samples` 目录下的 animation.html、animation_2.html

边框

CSS3 Border（边框）主要有以下属性：

- border-radius
- box-shadow
- border-image

注意：Internet Explorer 9+ 支持 border-radius 和 box-shadow 属性。Firefox、Chrome 以及 Safari 支持所有新的边框属性。 对于 border-image，Safari 5 以及更老的版本需要前缀 -webkit-。 Opera 支持 border-radius 和 box-shadow 属性，但是对于 border-image 需要前缀 -o-

border-radius （圆角边框）

在 CSS3 中，border-radius 属性用于创建圆角

```
.border_radius {
    border: 2px solid;
    font-size: 14px;
    color: #ffffff;
    font-weight: bold;
    padding: 10px;
    background: #6AAFCF;
    border-radius: 25px;
    -moz-border-radius: 25px; /* For Firefox Browser */
    -webkit-border-radius: 25px; /* For Safari and Google Chrome Browser */
    -o-border-radius: 25px /* For Opera Browser */
}
```

box-shadow （边框阴影）

在 CSS3 中，box-shadow 用于向方框添加阴影：

```
.box_shadow {
    font-size: 14px;
    color: #ffffff;
    font-weight: bold;
    padding: 10px;
    background: #6AAFCF;
    -moz-box-shadow: 15px 15px 5px #888245; /* For Firefox/Mozilla */
    -webkit-box-shadow: 15px 15px 5px #888245; /* For Google Chrome and Safari */
    -o-box-shadow: 15px 15px 5px #888245; /* For Opera */
    box-shadow: 15px 15px 5px #888245;
}
```

border-image （边框图片）

通过 CSS3 的 border-image 属性，您可以使用图片来创建边框：

```
.border_image {
    border-width: 15px;
    -moz-border-image: url(/images/border.png) 30 30 round; /* Firefox */
    -webkit-border-image: url(/images/border.png) 30 30 round; /* Safari and Chrome */
    -o-border-image: url(/images/border.png) 30 30 round; /* Opera */
    border-image: url(/images/border.png) 30 30 round;
}
```

源码

本文中所用例子源码参见 https://github.com/waylau/css3-tutorial 中 `samples` 目录下的 border.html

背景

CSS3 Background 中包含几个新的背景属性，提供更大背景元素控制。

主要是2个背景属性：

- background-size
- background-origin

您还将学习如何使用多重背景图像。

background-size

该属性规定背景图片的尺寸。

在 CSS3 之前，背景图片的尺寸是由图片的实际尺寸决定的。在 CSS3 中，可以规定背景图片的尺寸，这就允许我们在不同的环境中重复使用背景图片。

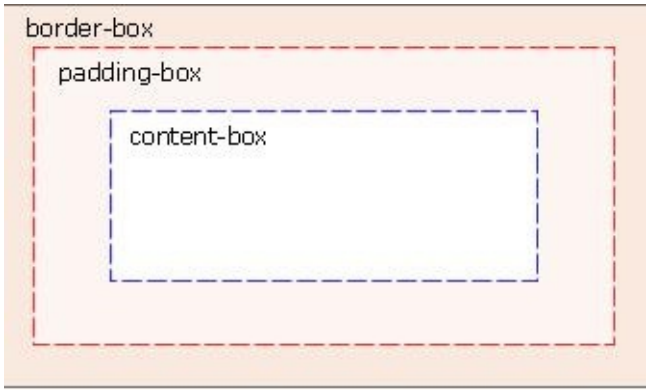
您能够以像素或百分比规定尺寸。如果以百分比规定尺寸，那么尺寸相对于父元素的宽度和高度。

```
.background-size {
    background: url(images/wl_white.png);
    background-size: 100px 40px;
    -moz-background-size: 100px 40px; /* Firefox 3.6 */
    -webkit-background-size: 100px 40px;
    background-repeat: no-repeat;
    padding-top: 40px;
}
```

background-origin

该属性指定了背景图像的位置区域。

content-box, padding-box,和 border-box 区域内可以放置背景图像。




```
.background-origin-border {
    width: 250px;
    height: 250px;
    border: 1px dotted green;
    padding: 25px;
    background-image: url('images/border.png');
    background-repeat: no-repeat;
    background-position: left;
    background-origin: border-box;
}

.background-origin-content {
    width: 250px;
    height: 250px;
    border: 1px dotted green;
    padding: 25px;
    background-image: url('images/border.png');
    background-repeat: no-repeat;
    background-position: left;
    background-origin: content-box;
}
```

源码

本文中所用例子源码参见 https://github.com/waylau/css3-tutorial 中 `samples` 目录下的 background_size.html、background_origin.html

字体

以前 CSS3 的版本，网页设计师不得不使用用户计算机上已经安装的字体。

使用 CSS3，网页设计师可以使用他/她喜欢的任何字体。

当你发现您要使用的字体文件时，只需简单的将字体文件包含在网站中，它会自动下载给需要的用户。

您所选择的字体在新的 CSS3 版本有关于 `@font-face` 规则描述。

您"自己的"的字体是在 CSS3 `@font-face` 规则中定义的。

注意：Internet Explorer 9+, Firefox, Chrome, Safari, 和 Opera 支持 WOFF (Web Open Font Format) 字体。

Firefox, Chrome, Safari, 和 Opera 支持 .ttf(True Type字体)和.otf(OpenType)字体字体类型）。

Chrome, Safari 和 Opera 也支持 SVG 字体/折叠。

Internet Explorer 同样支持 EOT (Embedded OpenType) 字体。

在 `@font-face` 规则中，您必须首先定义字体的名称（比如 FontAwesome ），然后指向该字体文件 fontawesome-webfont.woff 。

```
@font-face {
    font-family: 'FontAwesome';
    src: url('fonts/fontawesome-webfont.woff');
}

.font6 {
    font-family: 'FontAwesome', sans-serif;
    font-size: 14px;
    color: pink;
    line-height: 1.3em;
}
```

源码

本文中所用例子源码参见 https://github.com/waylau/css3-tutorial 中 `samples` 目录下的 font.html

多列

CSS3 多列再需要设计多个布局时是非常有用的。比如，报纸布局。

主要属性如下：

- column-count ： 指定元素的列数
- column-rule ： 指定的列之间的差距
- column-gap ： 设置列之间的宽度，样式和颜色

例子：

```
.newspaper {
    column-count: 3;
    -moz-column-count: 3; /* Firefox */
    -webkit-column-count: 3; /* Safari and Chrome */

    column-gap: 40px;
    -moz-column-gap: 40px; /* Firefox */
    -webkit-column-gap: 40px; /* Safari and Chrome */

    column-rule: 4px outset #ff00ff;
    -moz-column-rule: 4px outset #ff00ff; /* Firefox */
    -webkit-column-rule: 4px outset #ff00ff; /* Safari and Chrome */
}
```

属性

属性	说明	CSS
column-count	指定元素应分为的列数	3
column-fill	指定如何填充列	3
column-gap	指定列之间差距	3
column-rule	一个用于设置所有列规则的简写属性	3
column-rule-color	指定的列之间颜色规则	3
column-rule-style	指定的列之间的样式规则	3
column-rule-width	指定的列之间的宽度规则	3
column-span	指定一个元素应该横跨多少列	3
column-width	指定列的宽度	3
columns	缩写属性设置列宽和列数	3

源码

本文中所用例子源码参见 https://github.com/waylau/css3-tutorial 中 `samples` 目录下的 multiple_columns.html

UI

增加了一些新的用户界面特性来调整元素尺寸，框尺寸和外边框。

在本章中，您将了解以下的用户界面属性：

- resize
- box-sizing
- outline-offset

注意：Firefox、Chrome 以及 Safari 支持 resize 属性。 Internet Explorer、Chrome、Safari 以及 Opera 支持 box-sizing 属性。Firefox 需要前缀 -moz-。 所有主流浏览器都支持 outline-offset 属性，除了 Internet Explorer。

resize

resize 属性指定一个元素是否应该由用户去调整大小。可以使用 `resize:both`，`resize:vertical` 或者 `resize:horizontal`，用来分别设置元素是可以水平、垂直调整，垂直调整，水平调整。

例子

```
.div-both {
    border: 1px solid green;
    margin-top: 20px;
    padding: 15px 30px;
    width: 250px;
    resize: both;
    overflow: auto;
}

.div-horizontal {
    border: 1px solid green;
    margin-top: 20px;
    padding: 15px 30px;
    width: 250px;
    resize: horizontal;
    overflow: auto;
}

.div-vertical {
    border: 1px solid green;
    margin-top: 20px;
    padding: 15px 30px;
    width: 250px;
    resize: vertical;
    overflow: auto;
}
```

box-sizing

box-sizing 允许您以确切的方式定义适应某个区域的具体内容

例子

```
.box-sizing {
    box-sizing: border-box;
    -moz-box-sizing: border-box; /* Firefox */
    width: 50%;
    border: 1em solid red;
    float: left;
}
```

outline-offset

outline-offset 属性对轮廓进行偏移，并在超出边框边缘的位置绘制轮廓。

轮廓与边框有两点不同：

- 轮廓不占用空间
- 轮廓可能是非矩形

例子

```
.outline-offset {
    width: 180px;
    height: 80px;
    border: 1px solid red;
    outline: 1px solid green;
    outline-offset: 20px;
}
```

属性

属性	说明	CSS
appearance	允许您使一个元素的外观像一个标准的用户界面元素	3
box-sizing	允许你以适应区域而用某种方式定义某些元素	3
icon	Provides the author the ability to style an element with an iconic equivalent	3
nav-down	指定在何处使用箭头向下导航键时进行导航	3
nav-index	指定一个元素的Tab的顺序	3
nav-left	指定在何处使用左侧的箭头导航键进行导航	3
nav-right	指定在何处使用右侧的箭头导航键进行导航	3
nav-up	指定在何处使用箭头向上导航键时进行导航	3
outline-offset	外轮廓修饰并绘制超出边框的边缘	3
resize	指定一个元素是否是由用户调整大小	3

源码

本文中所用例子源码参见 https://github.com/waylau/css3-tutorial 中 `samples` 目录下的 userinterface.html

过渡效果

CSS3 过渡是元素从一种样式逐渐改变为另一种的效果。

尽管 CSS3 过渡效果是足够的过渡的一个元素,但是 text-transform 属性可以提高 CSS3 过渡效果的风格。

主要有四个属性的CSS3转换效果,已被描述如下:

- transition-property
- transition-duration
- transition-timing-function
- transition-delay

注意：Internet Explorer 10, Firefox, Opera, Chrome, 和Opera 支持transition 属性。Safari 需要前缀 -webkit-。Internet Explorer 9 以及更早的版本，不支持 transition 属性。Chrome 25 以及更早的版本，需要前缀 -webkit-

transition-property

规定应用过渡的 CSS 属性的名称。

```
transition-property: all;
transition-property: none;
transition-property: background-color;
transition-property: background-color, height, width;
```

transition-duration

定义过渡效果花费的时间。默认是 0。 时间单位可以是秒/毫秒。

```
transition-duration: 2s;
transition-duration: 1000ms;
transition-duration: 1000ms, 2000ms;
```

transition-timing-function

规定过渡效果的时间曲线。默认是 "ease"。

```
transition-timing-function: ease;
transition-timing-function: ease-in;
transition-timing-function: ease-in-out;
transition-timing-function: ease, linear;
transition-timing-function: cubic-bezier(1.000, 0.835, 0.000, 0.945);
```

其中：

- linear：线性过渡。等同于贝塞尔曲线(0.0, 0.0, 1.0, 1.0)
- ease：平滑过渡。等同于贝塞尔曲线(0.25, 0.1, 0.25, 1.0)
- ease-in：由慢到快。等同于贝塞尔曲线(0.42, 0, 1.0, 1.0)
- ease-out：由快到慢。等同于贝塞尔曲线(0, 0, 0.58, 1.0)
- ease-in-out：由慢到快再到慢。等同于贝塞尔曲线(0.42, 0, 0.58, 1.0)
- step-start：等同于 steps(1, start)
- step-end：等同于 steps(1, end)
- steps(`<integer>[, [ start | end ] ]?`)：接受两个参数的步进函数。第一个参数必须为正整数，指定函数的步数。第二个参数取值可以是start或end，指定每一步的值发生变化的时间点。第二个参数是可选的，默认值为end。
- cubic-bezier(`<number>, <number>, <number>, <number>`)：特定的贝塞尔曲线类型，4个数值需在[0, 1]区间内

transition-delay

规定过渡效果何时开始。默认是 0。

```
transition-delay: 2s;
transition-delay: 1000ms, 2000ms;
transition-delay: -2s;
```

源码

本文中所用例子源码参见 https://github.com/waylau/css3-tutorial 中 `samples` 目录下的 transitions.html

文本效果

CSS3 文本效果是这样一个术语用来在正常的文本中实现一些额外的特性。

主要是两个属性的 CSS3 文本效果,如下:

- text-shadow
- word-wrap

注意:Internet Explorer 10, Firefox,Chrome, Safari, 和 Opera支持text-shadow 属性。所有的主流浏览器支持自动换行（word-wrap）属性。Internet Explorer 9及更早IE版本不支持 text-shadow 属性

text-shadow

文本阴影。 您指定了水平阴影，垂直阴影，模糊的距离，以及阴影的颜色：

```
.text-shadow {
    text-shadow: 10px 10px 10px #6AAFCF;
}
```

word-wrap

换行。 CSS3中，自动换行属性允许您强制文本换行 - 即使这意味着分裂它中间的一个字：

```
.word-wrap {
    word-wrap: break-word;
    width: 150px;
    border: 1px solid #ff0000;
}
```

属性

属性	描述	CSS
hanging-punctuation	规定标点字符是否位于线框之外。	3
punctuation-trim	规定是否对标点字符进行修剪。	3
text-align-last	设置如何对齐最后一行或紧挨着强制换行符之前的行。	3
text-emphasis	向元素的文本应用重点标记以及重点标记的前景色。	3
text-justify	规定当 text-align 设置为 "justify" 时所使用的对齐方法。	3
text-outline	规定文本的轮廓。	3
text-overflow	规定当文本溢出包含元素时发生的事情。	3
text-shadow	向文本添加阴影。	3
text-wrap	规定文本的换行规则。	3
word-break	规定非中日韩文本的换行规则。	3
word-wrap	允许对长的不可分割的单词进行分割并换行到下一行。	3

源码

本文中所用例子源码参见 https://github.com/waylau/css3-tutorial 中 `samples` 目录下的 texteffects.html

2D 转换

CSS3 2D转换，我们可以斜拉(skew)，缩放(scale)，旋转(rotate)以及位移(translate)元素。

注意： Internet Explorer 10, Firefox, 和 Opera支持transform 属性。Chrome 和 Safari 要求前缀 -webkit- 版本。 Internet Explorer 9 要求前缀 -ms- 版本.

常用 2D 变换方法：

- translate()
- rotate()
- scale()
- skew()
- matrix()

translate()

translate()方法，根据左(X轴)和顶部(Y轴)位置给定的参数，从当前元素位置移动

```
.translate {
    transform: translate(50px, 100px);
    -ms-transform: translate(50px, 100px); /* IE 9 */
    -webkit-transform: translate(50px, 100px); /* Safari and Chrome */
}
```

rotate()

rotate()方法，在一个给定度数沿着元素中心顺时针旋转的元素。负值是允许的，这样是元素逆时针旋转。

```
.rotate
{
    transform:rotate(30deg);
    -ms-transform:rotate(30deg); /* IE 9 */
    -webkit-transform:rotate(30deg); /* Safari and Chrome */
}
```

scale()

scale()方法，该元素增加或减少的大小，取决于宽度（X轴）和高度（Y轴）的参数：

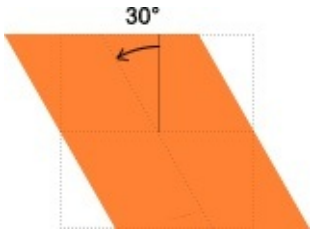
```
.scale
{
    transform: scale(2,4);
    -ms-transform: scale(2,4); /* IE 9 */
    -webkit-transform: scale(2,4); /* Safari and Chrome */
}
```

skew()

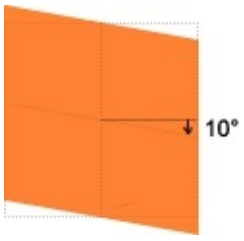
skew()方法，该元素会根据横向（X轴）和垂直（Y轴）线参数给定角度：

```
.skew
{
    transform:skew(30deg,20deg);
    -ms-transform:skew(30deg,20deg); /* IE 9 */
    -webkit-transform:skew(30deg,20deg); /* Safari and Chrome */
}
```

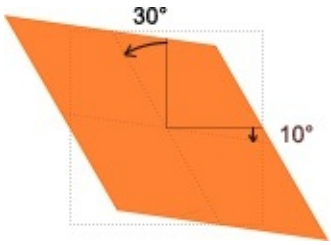
skewX(30deg) 如下图：



skewY(10deg) 如下图：



skew(30deg, 10deg) 如下图：



matrix()

matrix()方法和2D变换方法合并成一个。

matrix 方法有六个参数，包含旋转，缩放，移动（平移）和倾斜功能。

```
.matrix
{
    transform:matrix(0.866,0.5,-0.5,0.866,0,0);
    -ms-transform:matrix(0.866,0.5,-0.5,0.866,0,0); /* IE 9 */
    -webkit-transform:matrix(0.866,0.5,-0.5,0.866,0,0); /* Safari and Chrome */
}
```

源码

本文中所用例子源码参见 https://github.com/waylau/css3-tutorial 中 `samples` 目录下的 2d_transform.html

参考

- http://www.zhangxinxu.com/wordpress/2012/06/css3-transform-matrix-%E7%9F%A9%E9%98%B5/

3D 转换

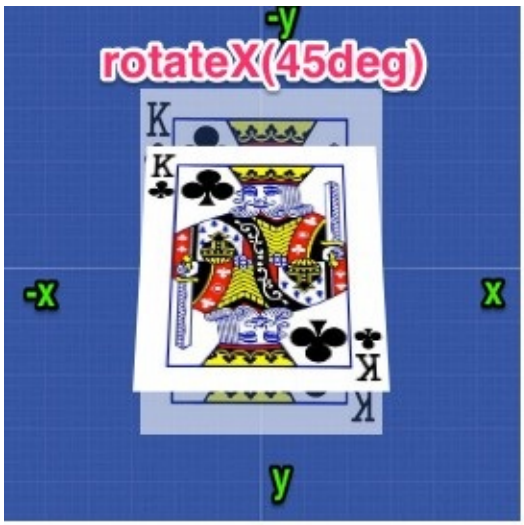
CSS3 3D Transform,用于 3 维动画或旋转。

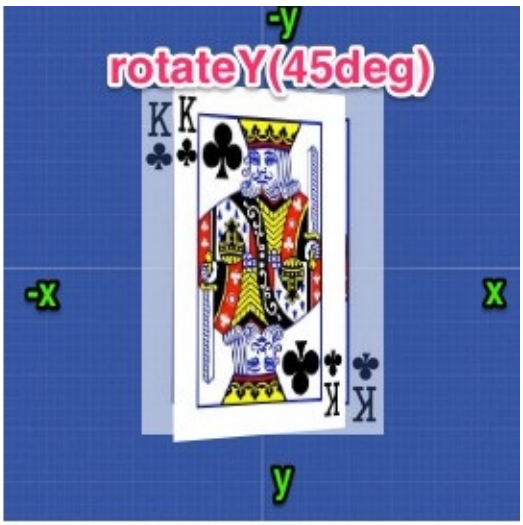
CSS3 3D 转换是一个属性,用于改变元素的实际形式。这个特性可以改变元素的形状、大小和位置。

主要有下列方法：

- rotateX()
- rotateY()
- rotateZ()

注意：Internet Explorer 10 和 Firefox 支持 3D 转换； Chrome 和 Safari 必须添加前缀 -webkit-； Opera 还不支持 3D 转换(支持 2D 转换)





rotateX()

rotateX()方法，围绕其在一个给定度数X轴旋转的元素。

```
.rotate-x {
    transform: rotateX(60deg);
    -webkit-transform: rotateX(120deg); /* Safari and Chrome */
}
```

rotateY()

rotateY()方法，围绕其在一个给定度数Y轴旋转的元素。

```
.rotate-y {
    transform: rotateY(60deg);
    -webkit-transform: rotateY(120deg); /* Safari and Chrome */
}
```

rotateZ()

rotateZ()方法，围绕其在一个给定度数Z轴旋转的元素。

```
.rotate-z {
    transform: rotateZ(60deg);
    -webkit-transform: rotateY(120deg); /* Safari and Chrome */
}
```

rotate3d()

otate3d(x,y,z,a)中取值说明：

- x：是一个0到 1 之间的数值，主要用来描述元素围绕X轴旋转的矢量值；
- y：是一个 0 到 1 之间的数值，主要用来描述元素围绕Y轴旋转的矢量值；
- z：是一个 0 到 1 之间的数值，主要用来描述元素围绕Z轴旋转的矢量值；
- a：是一个角度值，主要用来指定元素在3D空间旋转的角度，如果其值为正值，元素顺时针旋转，反之元素逆时针旋转。 面介绍的三个旋转函数功能等同：
- rotateX(a)函数功能等同于rotate3d(1,0,0,a)
- rotateY(a)函数功能等同于rotate3d(0,1,0,a)

- rotateZ(a)函数功能等同于rotate3d(0,0,1,a)

例子

```
.rotate-3d {
    transform: rotate3d(0,0.6,0.2,60deg);
    -webkit-transform: rotate3d(0.6,0.6,0.2,60deg); /* Safari and Chrome */
}
```

转换方法

函数	描述
matrix3d(*n*,*n*,*n*,*n*,*n*,*n*,*n*,*n*,*n*,*n*,*n*,*n*,*n*,*n*,*n*,*n*)	定义 3D 转换，使用 16 个值的 4x4 矩阵。
translate3d(*x*,*y*,*z*)	定义 3D 转化。
translateX(*x*)	定义 3D 转化，仅使用用于 X 轴的值。
translateY(*y*)	定义 3D 转化，仅使用用于 Y 轴的值。
translateZ(*z*)	定义 3D 转化，仅使用用于 Z 轴的值。
scale3d(*x*,*y*,*z*)	定义 3D 缩放转换。
scaleX(*x*)	定义 3D 缩放转换，通过给定一个 X 轴的值。
scaleY(*y*)	定义 3D 缩放转换，通过给定一个 Y 轴的值。
scaleZ(*z*)	定义 3D 缩放转换，通过给定一个 Z 轴的值。
rotate3d(*x*,*y*,*z*,*angle*)	定义 3D 旋转。
rotateX(*angle*)	定义沿 X 轴的 3D 旋转。
rotateY(*angle*)	定义沿 Y 轴的 3D 旋转。
rotateZ(*angle*)	定义沿 Z 轴的 3D 旋转。
perspective(*n*)	定义 3D 转换元素的透视视图。

源码

本文中所用例子源码参见 https://github.com/waylau/css3-tutorial 中 `samples` 目录下的 3d_transform.html

参考

- http://www.zhangxinxu.com/wordpress/2012/09/css3-3d-transform-perspective-animate-transition/
- http://www.w3cplus.com/css3/css3-3d-transform.html