www.SoftGozar.com

به نام خدا

آموزش کاربردی CSS (لایه دوم از طراحی وب)

Cascading Style Sheets (برگه های سبک آبشاری)

برگرفته از کتاب :

The Ultimate CSS Reference

BY TOMMY OLSSON & PAUL O'BRIEN

و

WWW.W3SCHOOLS.COM

آشنایی با طراحی سه لایه ای وب

آشنایی با تمامی اجزای CSS

بررسی انواع روش های استفاده از دستورات CSS در صفحات HTML

معرفی قواعد و قوانین CSS

بررسی انواع Selector ها

آموزش طراحی قالب های دو و سه ستونه

✓ همراه با پوشش کامل CSS 3

نویسنده : احمد بادپی

دانشگاه پیام نور مرکز آران و بیدگل

انتقادات و پیشنهادات خود را از طریق ایمیل زیر مطرح نمایید :

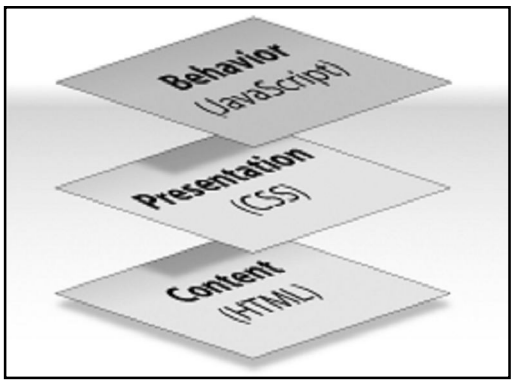
ahmadbadpey@gmail.com

www.SoftGozar.com

CSS چیست ؟

اجزای یک صفحه وب می توانند به سه لایه تقسیم شوند :

- **محتوا** – یا همان Content که بوسیله HTML ایجاد می شود .
- **نمایش** – یا همان Presentation که به وسیله CSS ایجاد می شود .
- **رفتار** – یا همان Behavior که به وسیله Javascript ایجاد می شود .



لایه محتوا همیشه وجود دارد . این لایه حاوی اطلاعاتی است که سازنده صفحه وب قصد دارد آن ها را به خواننده نشان دهد . این اطلاعات بین تگ های HTML و XHTML قرار داده می شود . این تگ ها ساختار صفحه وب را مشخص می کنند .
وظیفه لایه نمایش ، مشخص کردن ظاهر صفحه برای کاربر است . در واقع این لایه چگونگی نمایش لایه محتوا (عناصر صفحه) را مشخص می کند .

وظیفه لایه رفتار ، کنترل نحوه تعامل کاربر با صفحه وب است . این وظیفه معمولا به وسیله زبان جاوااسکریپت انجام می پذیرد . درواقع از طریق این لایه می توانیم به کنش های کاربر در صفحه واکنش نشان دهیم .
هر سه لایه مذکور را می توان در یک صفحه پیاده سازی کرد اما جداسازی آن ها ، یک مزیت بسیار بزرگ به شمار می رود : **می توان لایه ای را بدون نیاز به تغییر در لایه دیگر تغییر داد .**

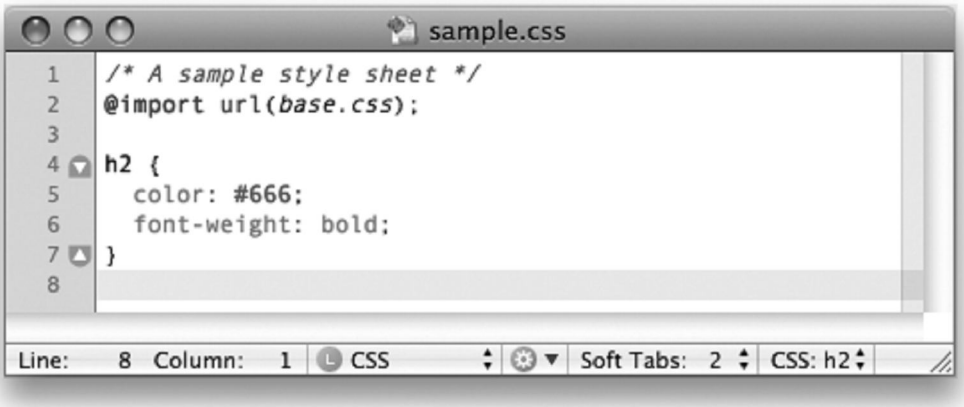
همانطور که می دانید HTML تگ هایی دارد که می توان به وسیله آن ها ، ظاهر و ساختار یک صفحه وب را ایجاد کرد . به عنوان مثال می توان از تگ های **<b>** و **<i>** برای کنترل ظاهر یک متن و از تگ **<hr />** برای قرار دادن یک خط افقی در صفحه استفاده نمود . با این وجود به این دلیل که این تگ ها لایه نمایش و محتوا را با یکدیگر ادغام می کنند ، مزیتی که پیشتر به آن اشاره شده از بین می رود .

Cascading Style Sheets یا همان **CSS** تکنیکی است که برای کنترل لایه نمایش در یک صفحه وب استفاده می شود.
مزیت اصلی CSS نسبت به عناصری از HTML که وظیفه ایجاد ظاهر صفحه را بر عهده دارند این است که می توان قالب و فرمت عناصر صفحه را از محتویات آن جدا کرد . به عنوان مثال می توان تمامی قالب های مربوط به تگ های **10000** صفحه وب را در یک فایل CSS نگهداری کرد . با CSS کنترل بیشتری بر روی ظاهر تگ های HTML خواهید داشت .

www.SoftGozar.com

سیستکس CSS و اصطلاحات :

برای اینکه با اجزای مختلف تشکیل دهنده **CSS** آشنا شوید شکل زیر را در نظر بگیرید .



سینتکس با یک Comment (توضیح) آغاز شده است :

```
/* A sample style sheet */
```

✓ **نکته : توضیحات برای تشریح کدها استفاده می شوند و می توانند به شما در ویرایش آنها پس از گذشت مدت زمانی کمک کنند . مرورگر ها comment ها را نادیده می گیرند . یک comment در CSS با یک */ شروع و به /* ختم می شود و می توان از آن ها برای ایجاد توضیحات یک خطی و چند خطی استفاده کرد . مثال های بیشتر :**

```
/*This is a comment*/

p{
      text-align:center;
      /*This is another comment*/
      color:black;
      font-family:arial
}

/*this is a
multiple line comment*/
```

در ادامه دو دستور ذکر شده است . دستور اول یک قانون است :

```
@import url(base.css);
```

دومین دستور مجموعه ای از قواعد را در بر می گیرد :

```
h2 {
  color: #666;
  font-weight: bold;
}
```

در ابتدای مجموعه قواعد فوق یک selector (گزینشگر) – عبارت سمت چپ کاراکتر {- آمده است . سپس بلوک تعریف قواعد نوشته می شود و در انتهای بلوک ، کاراکتر } قرار می گیرد . این مجموعه از دو قاعده تشکیل شده است . قواعد در بلوک تعریف ، با کاراکتر ";" از یکدیگر جدا می شوند :

```
color: #666;
font-weight: bold;
```

هر تعریف از یک نام و یک مقدار که با کاراکتر ":" از هم تفکیک شده اند تشکیل شده است .

www.SoftGozar.com

دستورات :

قواعد CSS از مجموعه ای از دستورات تشکیل شده اند . یک دستور ، یا یک **قانون** است یا **مجموعه ای از قواعد** . مثال زیر از دو دستور تشکیل شده است . اولین دستور ، یک قانون است که با کاراکتر ";" که در انتهای خط اول قرار دارد از دستور دوم جدا شده است و دومین دستور ، مجموعه ای از قواعد است که انتهای آن با کاراکتر "}" مشخص شده است :

```
@import url(base.css);
h2 {
	color: #666;
	font-weight: bold;
}
```

قوانین :

یک قانون با کاراکتر @ آغاز می شود و در ادامه ، یک مشخصه قرار می گیرد . یک قانون می تواند شامل یک متن با کاراکتر ";" در انتهای آن یا یک بلوک از تعاریف که بین کاراکتر های "{" و "}" محصور شده اند باشد . در حقیقت ، مشخصه ای که بعد از کاراکتر @ قرار می گیرد . مشخص می کند که آیا نیاز به یک بلوک از تعاریف بعد از آن است یا فقط یک متن ساده کفایت می کند .

در زیر مثالی از یک قانون CSS را ملاحظه می کنید که نیاز به بلوکی از تعاریف دارد . قانون @media را در زیر می بینید :

```
@media print {
	body {
		font-size: 12pt;
	}
}
```

و همچنین مثالی در مورد یک قانون که فقط یک متن ساده در ادامه آن آمده است و پایان آن با کاراکتر ";" مشخص شده است . قانون @import را در ذیل می بینید :

```
@import url(base.css);
```

مجموعه ای از قواعد :

یک مجموعه از قواعد ، از یک selector و مجموعه ای از قواعد CSS در ادامه آن تشکیل شده است . این مجموعه قواعد به تمامی تگ های هم نام با selector در صفحه اعمال می شوند . مثال را در این مورد ملاحظه می کنید :

```
h2 {
	color: #666;
	font-weight: bold;
}
```

Selector (گزینشگر) ها :

Selector ها همانطور که از نام آن ها پیداست به معنی "انتخاب کننده" هستند . هر آنچه قبل از کاراکتر "{" قرار می گیرد یک selector است . به عبارت دیگر ، selector قبل از تعریف بدنه مجموعه قوانین CSS قرار می گیرد و تمامی قواعد تعریف شده در بلوک ، به تگ هایی که از الگوی selector پیروی می کنند اعمال می شوند . در مثال زیر h2 یک selector است :

```
h2 {
	color: #666;
	font-weight: bold;
}
```

بلوک های تعریف قواعد :

بلوک تعریف قواعد CSS با کاراکتر "{" آغاز و با کاراکتر "}" پایان می یابند و می تواند شامل صفر یا تعداد بیشتری قواعد CSS باشد .

www.SoftGozar.com

تعریف قواعد ، نام و مقدار آن ها :

یک قاعده از یک نام و یک مقدار که با ":" از هم جدا شده اند تشکیل شده است . تعداد زیادی قواعد وجود دارد که می توان از آن ها استفاده نمود بسته به نوع قاعده ای که استفاده می شود نوع مقداردهی به آن نیز متفاوت است . این مقادیر می توانند از نوع متن ساده ، رنگ ، اعداد ، درصد ، آدرس های اینترنتی و ... باشند .

- ✓ **نکته : اگر مقداری که به یک قاعده می دهید چند کلمه ای است باید آن را داخل کوتیشن ها قرار دهید . به عنوان مثال دستور زیر فونت نمایشی تمامی پاراگراف های صفحه را به sans serif تغییر می دهد :**

```
p {font-family:"sans serif"}
```

- ✓ **نکته : اگر می خواهید بیش از یک قاعده را تعریف کنید می بایست آن ها را با یک ; از هم جدا کنید . دستور زیر نشان می دهد که چگونه می توان تمامی پاراگراف های صفحه را وسط چین و قرمز رنگ کرد :**

```
p {text-align:center;color:red}
```

برای خوانایی بهتر دستورات می توان هر قاعده را در یک خط به صورت زیر نوشت :

```
p
{
text-align:center;
color:black;
font-family:arial
}
```

اندازه ها و واحد ها :

یکی از مقادیری که برخی از قواعد **CSS** می پذیرند ، مقادیر طولی و اندازه هستند . این اندازه ها بر حسب واحدی معین بیان می شوند . به عنوان مثال 10 پیکسل یا 20 سانتیمتر و ... . واحد ها به دو دسته نسبی و قطعی تقسیم می شوند :

واحد های نسبی :

در جدول زیر لیستی از واحد های نسبی را مشاهده می کنید :

نام واحد	توضیحات
em	بر مبنای ارتفاع فونت پدر تگ جاری
ex	بر مبنای ارتفاع حرف **x** کوچک فونت پدر تگ جاری
px	پیکسل (1 پیکسل برابر با 0,26 میلی متر یا 1,96 اینچ است)

em واحدی است که اندازه آن به اندازه فونت پدر تگ جاری بستگی دارد . به عنوان مثال اگر اندازه فونت یک تگ p برابر با ۱۰ پیکسل باشد و تگی با اندازه فونت 2em درون تگ p قرار دهید ، اندازه فونت تگ درون p ، دو برابر اندازه فونت تگ p یعنی ۲۰ پیکسل خواهد بود .

ex واحدی است که اندازه آن به اندازه حرف x کوچک فونت پدر تگ جاری بستگی دارد . به عنوان مثال اگر اندازه فونت یک تگ p برابر با 10 پیکسل باشد و تگی را با اندازه فونت 2ex داخل آن قرار دهید اندازه فونت تگ درون تگ p ، دو برابر اندازه حرف x فونت تگ p در نظر گرفته خواهد شد .

www.SoftGozar.com

واحد های قطعی :

د رجدول زیر لیستی از واحد های قطعی آمده است :

نام واحد	توضیحات
mm	میلی متر
cm	سانتی متر
in	اینچ
pt	نقطه
pc	پیکا

استفاده از URI (آدرس های اینترنتی) در مقداردهی به قواعد :

بعضی از قواعد موجود در CSS مقادیری از نوع آدرس های اینترنتی را اختیار می کنند که می توانیم آن ها را با سینتکس url(*path*) مقداردهی کنیم . path همان مسیر به منبعی است که قصد داریم به آن دسترسی داشته باشیم . path را می توان بین کوتیشن ها (یعنی کاراکتر های ' یا ") هم قرار داد .

چگونه از دستورات CSS در صفحات HTML استفاده کنیم :

موقعی که مرورگر یک برگه سبک را می خواند صفحه را بر حسب آن نمایش می دهد . سه راه برای قراردادن دستورات CSS در صفحات HTML وجود دارد :

1- برگه های سبک خارجی (External Style Sheet) :

این نوع برگه های سبک برای زمانی مناسب اند که می خواهیم دستورات را به تعداد زیادی صفحات HTML اعمال کنیم . با این روش می توان نما (چهره) ی کل سایت را با تغییر تنها یک فایل عوض کرد . در این روش دستورات CSS (همان برگه های سبک) در یک فایل خارجی و با پسوند css. ذخیره می شوند . این صفحه باید به تمامی صفحات HTML مورد نظر لینک شود .
ما می توانیم برای این کار از تگی به نام link که جزء تگ های تهی است و حتما باید در قسمت head صفحه قرار بگیرد به صورت زیر استفاده کنیم :

```
<head>
     <link rel="stylesheet" type="text/css" href="mystyle.css" />
</head>
```

- ✓ **نکته : در تگ link فوق صفت rel نوع رابطه صفحه css با صفحه HTML (که همیشه باید برابر stylesheet باشد) ، صفت type ، MIME TYPE یک فایل با فرمت CSS و صفت href هم آدرس فایل CSS خارجی را نشان می دهند .**

- ✓ **نکته : در فایل css نباید از هیچ گونه تگی استفاده شود . به عنوان مثال فایل mystyle.css فوق فقط می تواند شامل دستورات css به صورت زیر باشد :**

```
hr {color:sienna}
p {margin-left:20px}
body {background-image:url("images/back40.gif")}
```

روش دیگری نیز برای لینک کردن یک فایل CSS به صفحات وب وجود دارد و آن استفاده از قانون @import به همراه تگی به نام Style است . به عنوان مثال فایل mystyle.css فوق را به صورت زیر هم می توان به صفحات HTML الحاق کرد :

```
<style type='text/css'>
     @import url(mystyle.css);
</style>
```

تگ Style هم مانند تگ link باید در قسمت head صفحه قرار بگیرد .

2- برگه های سبک داخلی (Internal Style Sheet) :

معمولا از این روش زمانی استفاده می شود که می خواهیم هر صفحه Style نمایشی اختصاصی خود را داشته باشد . در این روش برای تعریف قواعد از تگ style در داخل تگ head صفحه به صورت زیر استفاده می شود :

```
<head>
     <style type="text/css">
          hr {color:sienna}
          p {margin-left:20px}
          body {background-image:url("images/back40.gif")}
     </style>
</head>
```

- ✓ **نکته : احتمالا متوجه شده اید که تگ style دو کاربرد متفاوت دارد . اولی اینکه می توان از آن به همراه قانون @import برای الحاق فایل های خارجی CSS به صفحات HTML استفاده کرد . ضمن اینکه می توان ا ز آن به تنهایی برای تعریف برگه های سبک به روش داخلی بهره برد . اما معمولا کاربرد دوم آن بسیار بیشتر از کاربرد اول است !!!**

- ✓ **نکته : به صفت type در تگ style فوق که حتما باید ذکر شود و مقدارش هم چیزی جز text/css نمی تواند باشد دقت کنید .**

3- برگه های سبک درون خطی (Inline Style Sheet) :

این روش که استفاده از آن (مگر در مواقع خاص) اصلا پیشنهاد نمی شود بسیاری از امتیازات اصل جداسازی لایه های محتوا و نمایش را نقض می کند . معمولا از این روش وقتی می خواهیم تگی خاصی در صفحه style نمایشی خاصی داشته باشند استفاده می کنیم . در این روش می بایست در داخل تگ باز مربوطه از صفتی به نام style استفاده کنیم . در داخل این صفت هر تعداد قاعده CSS می تواند قرار بگیرد . مثال زیر نحوه تغییر رنگ متن و پس زمینه یک پاراگراف خاص (و نه همه پاراگراف ها) در صفحه را نمایش می دهد :

```
<p style="background-color:red;color:yellow">This is a paragraph.</p>
```

www.SoftGozar.com

انواع Selector (گزینشگر) ها :

Selector یک الگو و بخشی از دستور CSS محسوب می شود که بر روی تگ های صفحه وب تاثیر می گذارد . تمامی قواعدی که بعد از selector ذکر می شود تا زمانی که به وسیله قاعده دیگری نقض نشود بر روی تمامی تگ های صفحه که با الگوی selector تطبیق کند اعمال می شوند .

گزینشگر های ساده انواع مختلفی دارند :

- گزینشگر های نوع دار مانند "**h1**"
- گزینشگر سراسری یعنی "*****"
- گزینشگر های کلاس مثلا : "**.warning**"
- گزینشگر های نام ، مانند : "**#menu**"
- گزینشگر های صفت ، مانند "**[type='submit']**"
- کلاس های کاذب (شبه کلاس) ، مثل "**:hover**" : این کلاس ها بر روی رفتار یک گزینشگر نوع دار تاثیر می گذارد .

Selector سراسری :

Selector های سراسری برای انتخاب تمامی تگ های صفحه یا قسمتی از آن ها استفاده می شود . در صورتی که این **selector** به صورت ترکیبی همراه با **selector** های ساده دیگر استفاده شود می تواند نوشته نشود . به عنوان مثال ، دو دستور زیر معادل هم هستند :

syntax

* {
declaration
}

```
*.warning {
      declarations
}

.warning {
      declarations
}
```

به عنوان مثال برای اینکه قواعد margin و padding را برای تمامی تگ های صفحه تعیین کنید می توان دستوری به شکل زیر نوشت :

```
* {
  Margin:0;
  Padding:0;
}
```

تکه کد زیر را در نظر بگیرید :

```
<body>
      <div>
         <h1>The <em>Universal</em> Selector</h1>
         <p>We must <em>emphasize</em> the following:</p>
            <ul>
                  <li>It's <em>not</em> a wildcard.</li>
                  <li>It matches elements regardless of <em>type</em>.</li>
            </ul>
         This is an <em>immediate</em> child of the division.
      </div>
</body>
```

www.SoftGozar.com

می توان selector ی مانند "**div * em**" را به این صورت تر جمه کرد :

"بر روی تمامی تگ های emی که زیر مجموعه ای از تگ های div هستند عمل کن " .

در نتیجه selector فوق ، متون زیر را تحت تاثیر قرار میدهد :

- کلمه "Universal" در تگ h1 (* تگ h1 را تطبیق می دهد)
- کلمه "emphasize" در تگ p (* تگ p را تطبیق میدهد)
- کلمه not در اولین تگ li (* تگ ul و li را تطبیق میدهد)
- کلمه type در دومین تگ li (* تگ ul و li را تطبیق میدهد)

با این وجود کلمه immediate با وجود اینکه در قرار گرفته است تحت تاثیر selector قرار نمی گیرد . چون این تگ زیرتگی از تگ div است نه زیر تگی از زیرتگ های تگ div. به عبارتی دیگر ، زیر تگی برای آن که توسط کاراکتر * تطبیق شود بین تگ div و وجود ندارد .

Selector نوع دار :

syntax
Element { declaration }

Element در سینتکس این selector بیانگر عنوان تگ است . دقت داشته باشید که عنوان تگ با نام آن متفاوت است . این نوع selector ها فقط بر روی تگ هایی که عنوان آن ها با عنوان Element یکسان است اعمال می شوند . selector های نوع دار در صفحات HTML به بزرگ یا کوچک بودن عنوان تگ ها حساس نیستند .

مثالی از کاربرد این نوع selector :

```
li {
 color:blue;
 font-weight:bold;
}
```

در این مثال رنگ تمامی عناصر li صفحه آبی و به حالت bold در خواهند آمد .

Selector کلاس :

syntax
.ClassName { declaration }

className در سینتکس این نوع selector نام دلخواهی است که به کلاس نسبت داده می شود . قبل از نام کلاس ، کاراکتر "." قرار می گیرد . اعمال فرمت به تگ هایی که دارای صفت class هستند متداولترین روش تعیین فرمت برای تگ هاست . تگ هایی که دارای صفت class هستند ، نام کلاسی را که برای قواعد تعریف شده است به عنوان مقدار صفت class می پذیرند .

به عنوان مثالی ساده ، قواعد ذیل ، به تمامی تگ هایی در صفحه که مقدار صفت class شان برابر با warning باشد اعمال می شوند :

```
.warning {
      Font-weight:bold;
      Color:red;
}
```

کلاس warning. را می توان با استفاده از selector سراسری به صورت "*.warning" نیز نوشت . دقت داشته باشید که کاراکتر های * و . و کلمه warning به یکدیگر چسبیده اند و فاصله ای نباید بین آن ها وجود داشته باشد .

www.SoftGozar.com

به عنوان مثالی دیگر ، selector زیر بر روی تمامی تگ های p صفحه که صفت class آنها برابر با کلمه intro است اعمال می شود :

```
p.intro {
      Font-weight:bold;
      Color:green;
}
```

معمولا از selector کلاس هنگامی که می خواهیم یک نوع تگ خاص در صفحه style های متفاوتی داشته باشند استفاده می کنیم .
به عنوان مثال فرض کنید که می خواهید دو نوع پاراگراف در صفحه داشته باشید : یکی وسط چین و دیگری راست چین .
برای این کار می توان دو کلاس زیر را برای تگ های p تعریف نمود :

```
p.right {text-align:right}
p.center {text-align:center}
```

حا ل می توانیم این کلاس ها را با استفاده از صفت class بر روی دو تگ p اعمال کنیم :

```
<p class="right">This paragraph will be right-aligned.</p>
<p class="center">This paragraph will be center-aligned.</p>
```

- ✓ **نکته : ما می توانیم بیش از یک class را نیز به یک تگ اعمال کنیم . برای اینکار می بایست نام کلاس ها را با یک فاصله در صفت class تگ مربوطه ذکر کنیم . به عنوان مثال پاراگراف زیر از قواعد کلاس هایی به نام های center و bold پیروی می کند :**

```
<p class="center bold">This is a paragraph.</p>
```

- ✓ **نکته : در انتخاب نام کلاس دقت کنید . این نام نباید با عدد شروع شود زیرا مرورگر Firefox آن را نادیده می گیرد . همچنین این نام نباید با کاراکتر های _ (زیر خط) یا – (خط تیره) شروع شود در این صورت مرورگر Internet Explorer آن را نادیده می گیرد .**

Selector نام (ID):

syntax
#ID declaration }

Selector های نام همانند selector های class هستند . این selector ها بر روی تگ هایی تاثیر می گذارند که دارای صفت ID هستند . در یک صفحه وب ، از یک ID فقط می توان برای یک تگ استفاده نمود و دو تگ نمی توانند دارای مقادیر یکسانی برای صفت ID باشند . در صورتی که تگ ها می توانند دارای مقدار یکسانی برای صفت class شان باشند .
در سینتکس این selector ، ID مقداری است که به صفت ID تگ در صفحه نسبت داده شده است . قبل از ID کاراکتر # قرار می گیرد .
قاعده دستور زیر بر روی هر تگی که مقدار صفت ID آن برابر با nav باشد اعمال می شود :

```
#nav {
 Color:lue;
}
```

امکان ترکیب selector نوع دار با selector نام نیز وجود دارد . البته این حالت بسیار کم استفاده می شود چون دو یا چندین تگ در یک صفحه نمی توانند دارای مقدار ID یکسانی باشند .
در مثال زیر ، قواعد فقط به تگ ul ی اعمال می شوند که ID آن برابر با nav باشد .

```
ul#navigation {
      color:lue;
}
```

www.SoftGozar.com

Selector صفت :

syntax
[{ attribute | attribute { = | |= | ~= } attribute value }] { **declaration** **}**

از این نوع selector برای انتخاب تگ ها بر اساس صفت ها و مقادیر صفت هایشان استفاده می شود .

این selector بر روی تگ هایی که دارای صفت مشخص یا صفتی با مقدار تعیین شده هستند تاثیر می گذارد . مقدار صقت یا باید دقیقا وجود داشته باشد یا وجود قسمتی از آن نیز کفایت می کند .selector های صفت بین "[" و "]" محصور شده اند .

ساده ترین شکل استفاده از این selector از قرارگیری نام خاصیت بین "[" و "]" به وجود می آید :

```
[href]{
  مجموعه قواعد
}
```

در این مثال قواعد بر روی تمامی تگ هایی که خاصیت href دارند اعمال می شوند . selector فوق معادل با "*[href]" است .

مثال دیگری را بررسی می کنیم :

```
a[href]{
  مجموعه قواعد
}
```

Selector بالا تمامی تگ های a که دارای خاصیت href هستند را تحت تاثیر قرار می دهد .

از عملگر = می توان برای بررسی برابر بودن مقداری مشخص از خاصیت یک تگ استفاده نمود :

```
input[type="submit"] {
  قواعد
}
```

Selector فوق تمامی تگ های input ی که مقدار صفت type آنها برابر با submit است را تحت تاثیر قرار می دهد .

- ✓ **نکته : selector صفت فقط بر روی تگ هایی تاثیر می گذارد که نام صفت و مقدار تعریف شده در این selector حتما برای تگ مربوطه ذکر شده باشد . به عنوان مثال در HTML مقدار پیش فرض صفت method تگ form عبارت get است اما نمی توان با تکیه بر selectorی مانند form[method='get'] اطمینان داشت که قواعد تعیین شده حتما به تگ فرم اعمال می شوند چون اگر تگ form به صورت <form action='ahmadbadpey.php'> باشد چون صفت method ی صریحا برای آن ذکر نشده است پس selector بر روی این تگ تاثیری نمی گذارد .**

از عملگر =| برای تطبیق تگ هایی استفاده می شود که مقدار خاصیتی مشخص از آن ها با عبارتی معین آغاز می شود و در ادامه لیتسی از عبارت ها می آیند که با کاراکتر "-" از یکدیگر جدا شده اند . به عنوان مثال :

```
[hreflang|="en"] {
  قواعد
}
```

Selector فوق بر روی تمامی تگ هایی که مقدار خاصیت hreflang آنها با عبارت en یا en- آغاز می شود اثر می گذارد . به عبارت دیگر ، مقادیری همانند "en" ، "en-US" و "en-GB" و همانند آن توسط این دستور تطبیق داده می شود .

www.SoftGozar.com

ا زعملگر "=~" در selector های صفت می توان برای تطبیق تگ هایی که در مقدار خاصیت معینی از آن ها یک کلمه خاص وجود دارد استفاده کرد . دستور زیر را در نظر بگیرید :

```
[class~="warning"] {
  قواعد
}
```

اگر این دستور بر روی صفحه وبی با تگ های زیر اعمال شود :

```
<p class="warning"></p>
<strong class="important warning"></strong>
<div class="warning highlight"></div>
<p class="my-warning"></p>
<ul class="warnings"></ul>
```

فقط سه تگ اول تحت تاثیر قرار می گیرند . تگ چهارم بدین دلیل که کلمه "warning" در آن یک کلمه مستقل نیست (چون با کاراکتر "-" به کلمه "my" چسبیده است) و تگ پنجم به دلیل وجود حرف s ی که در انتهای کلمه "warning" قرار دارد تطبیق نمی شود .

Selector صفت در CSS نسخه 3 :

در CSS نسخه 3 ، سه نوع دیگر از selector های صفت معرفی شده اند . این selector ها می توانند جستجو را در بخشی از مقدار یک صفت انجام دهند .

syntax
[attribute { ^= | $= | *= } attribute value] { **declaration** **}**

از عملگر "=^" برای تطبیق شروع مقداری از یک صفت با مقداری مشخص استفاده می شود . مثال :

```
a[href^="http:"] {
  قواعد
}
```

قواعد selector فوق بر روی تمامی تگ های a که صفت href آنها با عبارت "http:" آغاز می شود اعمال می شوند .

عملگر "=$" دقیقا بالعکس عملگر قبل رفتار می کند . این عملگر برای تطبیق پایان مقداری از یک صفت با مقداری مشخص استفاده می شود . به عنوان مثال :

```
img[src$=".png"] {
  قواعد
}
```

قواعد selector فوق بر روی تمامی تگ های img که صفت src آنها با عبارت png. پایان می پذیرد اعمال می شوند .

و در نهایت از عملگر "=*" برای تطبیق وجود مقداری در هر جای مقداری از یک صفت استفاده می شود . مثال :

```
div[id*="foo"] {
  قواعد
}
```

قواعد selector فوق بر روی تمامی تگ های div که در خاصیت id آنها عبارت "foo" وجود دارد اعمال می شوند . "foo" می تواند قسمتی از یک کلمه دیگر باشد .

✓ **نکته : selector های صفت در مرورگر Internet Explorer نسخه 6 پشتیبانی نمی شوند .**

www.SoftGozar.com

گروه بندی selector ها :

Selector ها را می توان با قرار دادن کاراکتر "," بین آنها گروه بندی و یکی کرد . به عنوان مثال ، قواعد ذیل به تمامی تگ های td و th صفحه اعمال می شوند :

```
td, th {
  declarations
}
```

می توان به کاراکتر "," به عنوان عبارت "یا" نگاه کرد در نتیجه دستور فوق بدین شکل بیان می شود :

"قواعد را به تمامی تگ های td یا td صفحه اعمال کن ."

نکته مهم در گروه بندی selector ها این است که هر قسمت از آن ها به صورت مستقل عمل می کنند به عنوان مثال یکی از برداشتهای اشتباهی که فرد تازه کاری ممکن است درگروه بندی selector ها مرتکب شود به صورت ذیل است :

```
#foo td, th {
      declarations
}
```

ممکن است این طور به نظر برسد که قواعد فوق به تمامی تگ های td و th که زیر تگ هایی از تگی به نام "#foo" هستند اعمال می شوند اما این طور نیست ! در حقیقت دستور فوق معادل با دستورات زیر است :

```
#foo td {
      declarations
}
th {
      declarations
}
```

برای اینکه به مقصودی که در پاراگراف قبل به آن اشاره شد برسیم ، دستور قبل را می توان به شکل زیر نوشت :

```
#foo td, #foo th {
      declarations
}
```

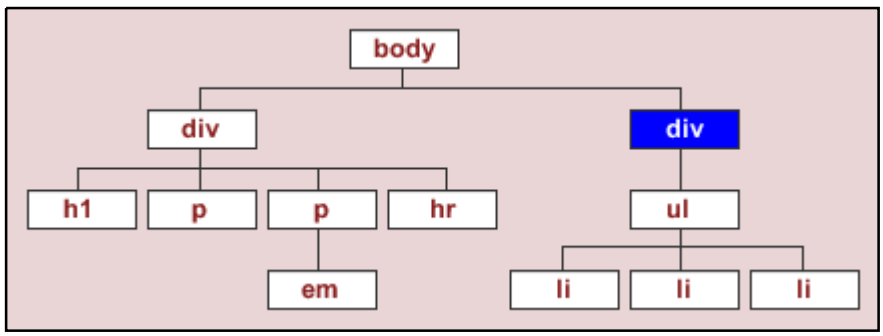
تمام صفحات HTML یک درخت هستند !!!

قبل از اینکه وارد مبحث بعدی شویم لازم است یکی از مهمترین مفاهیم موجود در CSS را بیان کنیم و آن نمایش تگ های موجود در صفحات HTML در قالب یک درخت و بیان روابط خانوادگی موجود بین آن هاست !

ما می توانیم تگ های موجود در یک صفحه را به صورت یک درخت تصور کنیم . به عنوان مثال کد زیر را در نظر بگیرید . توجه داشته باشید که در کد زیر برای اختصار از تگ های html و head چشم پوشی شده است :

```
<body>
     <div id="content">
          <h1>Heading here</h1>
          <p>Lorem ipsum dolor sit amet.</p>
          <p>Lorem ipsum dolor <em>sit</em> amet.</p>
          <hr>
      </div>
      <div id="nav">
          <ul>
              <li>item 1</li>
              <li>item 2</li>
              <li>item 3</li>
          </ul>
     </div>
</body>
```

کد فوق را می توان به صورت درخت زیر تصور کرد :



بین عناصر موجود در این درخت می توان روابط خانوادگی زیر را در نظر گرفت :

- ✓ Ansector (جد)
- ✓ Descendants (اولاد)
- ✓ Parents (پدر)
- ✓ children (فرزند)
- ✓ Siblings (برادر)

به عنوان مثال در درخت فوق می توان تگ div مشخص شده با پس زمینه تیره تر را به صورت زیر بیان کرد :

- ✓ فرزندی از عنصر <body>
- ✓ پدر عنصر
- ✓ جد عناصر <ul> و
- ✓ برادر عنصر دیگر **(به این دلیل که پدر آن ها یکسان است (<body>)**

درک درست از این مفاهیم در ادامه برای یادگیری بهتر مطالب به ما کمک خواهد کرد .

Combinator (ترکیب کننده) ها:

یک selector می تواند از بیش از یک selector ساده تشکیل شده باشد . بین selector های ساده یک نماد ترکیب کننده قرار می گیرد . نماد های ترکیب کننده نحوه ارتباط بین selector ها را تشریح می کنند . این نماد ها ، رفتار پیش فرض selector ها را تغییر می دهند و باعث بوجود آمدن چهار نوع جدید selector می شوند :

- **Selector اولاد**
- **Selector فرزند**
- **Selector هم نوع مجاور**
- **Selector هم نوع مجاور عمومی (معرفی شده در نسحه CSS 3)**

Selector اولاد

این نوع selector تمامی تگ هایی مشخص را که اولاد تگی دیگر محسوب می شوند تحت تاثیر قرار می دهد . حرف E در سینتکس این selector نمایانگر تگ پدر است . حرف F نیز تگ فرزند را مشخص می کند .
نماد ترکیب کننده ای که برای selector اولاد استفاده می شود ، یک فاصله (space) یا کلید TAB است . کد زیر را در نظر بگیرید :

syntax

E F{
declaration block
}

```
<ul>
      <li>Item 1</li>
      <li>
            <ol>
                  <li>Sub-item 2A</li>
                  <li>Sub-item 2B</li>
            </ol>
      </li>
</ul>
```

اگر selector زیر را بر روی کد های فوق اعمال کنیم :

```
ul li {
      declarations
}
```

تمامی قواعد selector بالا بر روی تمامی تگ های li موجود در کد HTML اعمال می شوند . چون تگ های li همگی زیر مجموعه ای از تگ ul هستند .
از selector اولاد نمی توان برای تطبیق "**فقط**" تگ های li مختص تگ ul استفاده نمود و از تطبیق تگ های li درون تگ ol جلوگیری کرد . بدین منظور باید از selector **فرزند** استفاده نمود .

Selector فرزند :

این selector فقط تگ های فرزند یک تگ پدر را انتخاب می کند نه تگ های فرزند فرزندان تگ دیگری را . حرف E در سینتکس این selector نمایانگر تگ پدر و F نمایانگر تگ فرزند است . نماد ">" باید به تگ پدر و فرزند چسبیده باشد . کد ذیل را در نظر بگیرید :

syntax

E>F{
declaration block
}

```
<ul>
      <li>Item 1</li>
      <li>
            <ol>
                  <li>Subitem 2A</li>
                  <li>Subitem 2B</li>
            </ol>
      </li>
</ul>
```

اگر selector زیر را بر روی کد های فوق اعمال کنیم :

```
ul>li {
      declarations
}
```

تنها تگ های li مختص تگ ul تحت تاثیر قرار می گیرند نه تگ های li تگ ol . به عبارت دیگر ، قواعد فقط بر روی اولین سطح فرزندان تگ پدر اعمال می شوند .

Selector هم نوع مجاور :

این selector فقط تگ بعد از تگ دیگری را تحت تاثیر قرار می دهد . نماد ترکیب کننده ، کاراکتر + است .

syntax

E+F{
declaration block
}

این selector را می توان به گونه ای دیگر نیز تعریف کرد :

این selector ، عناصر منطبق با F را که اولین برادر بعد از عناصر منطبق با E هستند را تحت تاثیر قرار می دهد .

کد زیر را در نظر بگیرید :

```
<h2>Heading</h2>
<p>The selector above matches this paragraph.</p>
<p>The selector above does not match this paragraph.</p>
```

اگر selector زیر را بر روی کد های بالا اعمال کنیم :

```
h2+p {
      declarations
}
```

فقط اولین تگ p بعد از تگ h2 تحت تاثیر قواعد فوق قرارمی گیرد . قواعد بر روی دومین تگ p اعمال نمی شوند چون این تگ بلافاصله پس از تگ h2 نیامده است . (در واقع اولین برادر آن نیست)

اگرselector قبل را بر روی کد های زیر اعمال کنیم :

```
<h2>Heading</h2>
<div>
      <p>The selector above does not match this paragraph.</p>
</div>
```

تگ p چون بلافاصله بعد از تگ h2 قرار نگرفته است تحت تاثیر قرار نمی گیرد .

و همچنین اگر selector قبل را بر روی کد های زیر اعمال کنیم :

```
<h2>Heading</h2>
Lorem ipsum dolor sit amet.
<p>The selector above matches this paragraph.</p>
```

قواعد بر روی تگ p اعمال خواهند شد . مشاهده می کنید که در مثال فوق بعد از پایان تگ p متنی آمده است اما عبارت های متنی در حوزه عملکرد این selector تاثیری نمی گذارد .

Selector هم نوع مجاور عمومی :

syntax
E~F{ **declaration block** **}**

این selector در CSS نسخه 3 معرفی شده است و نماد ترکیب کننده آن کاراکتر "~" است . این selector تمامی تگ های هم نوع بعد از تگی مشخص را تطبیق می دهد .
این selector را به شکل دیگری نیز تعریف می کنند :

این selector تمامی تگ های منطبق با F که برادر های بعد از تگ های منطبق با E هستند را تحت تاثیر قرار می دهد .

در مثال ذیل تمام تگ های P بعد از تگ h2 تطبیق داده می شوند توجه داشته باشید که نیاز نیست که این تگ ها بلافاصله بعد از تگ پدر یا پست سرهم بیایند بلکه فقط تگ پدر همه آن ها باید یکسان باشد . منظور از تگ پدر در اینجا تگی است که قبل از آن ها می آید :

```
h2~p {
      declarations
}
```

کد زیر را در نظر بگیرید :

```
<h2>Heading</h2>
<p>The selector above matches this paragraph.</p>
<p>The selector above matches this paragraph.</p>
```

در کد فوق هر دو تگ p توسط selector فوق تحت تاثیر قرار می گیرند .

اما در کد زیر تگ p توسط selector در نظر گرفته نمی شود چون پدر آن تگ div است و نه تگ h2 .

```
<h2>Heading</h2>
<div>
     <p>The selector above does not match this paragraph.</p>
</div>
```

در کد های زیر تنها دومین تگ p توسط selector تحت تاثیر قرار می گیرد . اولین تگ p چون قبل از h2 آمده تحت تاثیر قرار نمی گیرد :

```
<p>The selector above does not match this paragraph.</p>
<h2>Heading</h2>
<p>The selector above matches this paragraph.</p>
```

کلاس های کاذب (pseudo Class) :

کلاس های کاذب رفتار تگ های دیگر را تغییر می دهند . این کلاس ها نمی توانند همانند قواعد CSS به صورت مستقیم در صفت style تگ ها نوشته شوند . آنها حتما باید در فایل CSS یا بین تگ style در صفحه وب تعریف شوند .
یک کلاس کاذب با ":" آغاز می شود و بعد از آن نام کلاس قرار می گیرد . قبل از ":" نیز نام selector ی که قرار است کلاس کاذب ، رفتار آن را اصلاح کند نوشته می شود . دقت داشته باشید که بین نام selector ، کاراکتر ":" و کلاس کاذب هیچ فاصله ای نباشد .

در CSS نسخه 1 سه کلاس کاذب معرفی شدند : کلاس های ":link" ، ":visited" و ":active" . این کلاس ها فقط برای اصلاح رفتار تگ a که نمایانگر یک لینک است ارائه شده بودند و نمایانگر وضعیت لینک در حالت های مختلف از جمله لینک کلیک نشده (بازدید نشده) ، کلیک شده (بازدید شده) و لینک های فعال که با رنگ دیگری نشان داده می شوند بودند . البته کلاس های ":link" و ":visited" اصطلاحا نسبت به هم انحصار دوگانه دارند و نمی توانند همزمان با یکدیگر در یک selector به کار روند .

- ✓ **نکته : انحصار دوگانه به حالتی اطلاق می شود که در آن ، دو شیء نسبت به هم تضاد رفتاری دارند . به عبارت دیگر ، دو شیء نمی توانند همزمان با یکدیگر استفاده شوند .**

در CSS نسخه 2 تعدادی دیگر از کلاس های کاذب به نام های ":focus" و ":hover" معرفی شدند . **همچنین در این نسخه امکان اعمال کلاس های کاذب به هر نوع تگی فراهم شد .**
دو کلاس کاذب دیگر نیز در این نسخه وجود دارند . کلاس ":lang" که اعمال رفتار بر روی تگ را منوط به نوع زبانی که برای آن تگ در نظر گرفته شده است می کند . و کلاس ":first-child" برای اعمال قواعد بر روی تگی که اولین فرزند پدرش هست .
در CSS نسخه 3 هم تعداد دیگری کلاس کاذب معرفی شدند .

یک selector ساده می تواند از بیش از یک کلاس کاذب تشکیل شده باشد . البته این گفته در حالی است که آن دو کلاس کاذب نسبت به یکدیگر انحصار دوگانه نداشته باشند . به عنوان مثال استفاده همزمان از کلاس های ":link" و ":hover" به صورت "a:link:hover" مجاز است اما استفاده همزمان از دو کلاس ":link" و ":visited" به صورت "a:link:visited" بی معناست و نسبت به یکدیگر انحصار دوگانه دارند . چون بر روی یک لینک کلیک شده است یا کلیک نشده است !

ترتیب تعریف کلاس های کاذب مختص لینک ها بسیار مهم است . این ترتیب باید همیشه رعایت شود و به شکل زیر باشد :

```
a:link {
      declarations
}
a:visited {
      declarations
}
a:focus {
      declarations
}
a:hover {
      declarations
}
a:active {
      declarations
}
```

اگر ترتیب فوق رعایت نشود و به عنوان مثال کلاس ":focus" بعد از کلاس ":hover" بیاید قواعدی که برای کلاس ":hover" تعیین شده است . برای کلاس ":focus" در نظر گرفته می شوند !!!

اینک به بررسی برخی از این کلاس های کاذب پرداخته و مثال هایی در مورد آن ها را بررسی خواهیم کرد.

:link

syntax

:link {
declaration block
}

قواعد این کلاس به لینک هایی در صفحه وب که بر روی آن ها کلیک نشده است (قبلا بازدید نشده اند) اعمال می شوند . در مثال زیر رنگ متن تمامی لینک هایی که بر روی آنها کلیک نشده است green خواهد شد :

```
a:link {
      color: green;
}
```

:visited

syntax

:visited {
declaration block
}

قواعد این کلاس به لینک هایی در صفحه وب که بر روی آن ها کلیک شده است (قبلا بازدید شده اند) اعمال می شوند . در مثال زیر رنگ متن تمامی لینک هایی که بر روی آنها کلیک شده است red خواهد شد :

```
a:visited {
      color: red;
}
```

:focus

syntax

:focus {
declaration block
}

این کلاس هنگامی عمل می کند که بر روی تگ مورد نظر فوکاس شده باشد . به عنوان مثال با کلید **TAB** انتخاب شده باشد . در مثال زیر حاشیه ای با فرمتی که درقاعده border تعریف شده است در هنگام قرار گرفتن فوکاس بر روی یک تگ textarea برای آن ایجاد می شود :

```
textarea:focus {
      border: 2px solid #F2F2F2;
}
```

- ✓ **نکته : کلاس های link,visited,hover,active در مرورگر IE فقط بر تگ های a که دارای صفت href هستند اعمال می شود . اما در مرورگر های دیگر این کلاس های را بر هر نوع تگی قابل اعمال است .**
- ✓ **نکته : مرورگر IE از شبه کلاس focus برای هیچ نوع تگی پشتیبانی نمی کند .**

:hover

syntax

:hover {
declaration block
}

کلاس hover مختص سیستم هایی همچون کامپیوتر است که در آن از اشاره گر برای کار با صفحه وب و المان های آن استفاده می شود .
رفتار این کلاس هنگامی است که اشاره گر ماوس بر روی یک عنصر برود .در مثال زیر حاشیه ای با فرمتی که درقاعده border تعریف شده است در هنگام قرار گرفتن ماوس بر روی یک تگ img برای آن ایجاد می شود :

```
img:hover {
      border: 5px solid #F2F2F2;
}
```

:active

syntax

:active {
declaration block
}

قواعد این کلاس به تگ هایی که فرآیند فعال شدن / بودن در مورد آن ها صدق می کند فعال شدن میتواند در مرورگر های مختلف تعابیر متفاوتی داشته باشد .
حالت active را می توان از زمانی که بر روی یک عنصر کلیک می شود تا زمانی که دکمه ماوس رها می شود در نظر گرفت .

```
a:active {
      color: blue;
}
```

:first-child

syntax

:first-child {
declaration block
}

این کلاس در صورتی بر روی تگ اثر می گذارد که آن تگ اولین تگ فرزند تگ پدر خود باشد . به عنوان مثال selector ی مانند "li:first-child" فقط بر روی اولین زیر تگ li تگ های ol یا ul تاثیر می گذارد .مثال زیر را در نظر بگیرید :

```
li:first-child {
      color:green;
}
```

اگر این selector را بر روی کد زیر اجرا کنیم :

```
<ul>
      <li>This item matches the selector li:first-child.</li>
      <li>This item does not match that selector.</li>
      <li>Neither does this one.</li>
</ul>
```

فقط اولین تگ li تحت تاثیر قرار می گیرد و رنگش سبز خواهد شد .

:last-child

syntax

:last-child {
declaration block
}

این کلاس دقیقا عکس کلاس فوق عمل می کند و در صورتی بر روی تگ اثر می گذارد که آن تگ آخرین تگ فرزند تگ پدر خود باشد .

کلاس های کاذب CSS نسخه 3 :

:nth-child(N)

syntax

:nth-child({ number expression | odd | even }) {
declaration block
}

این کلاس ، تگ ها را بر اساس مکان آن ها در تگ پدرشان تطبیق می دهد . حرف N در جلوی این کلاس معرف یک کلمه کلیدی (odd یا even) ، یک عدد یا یک عبارت عددی به فرم an+b است . n در فرم این عبارت عددی ، یک عدد تصاعدی خودکار است که از 0 شروع می شود . در a ضرب می شود و با b جمع می شود . دقت داشته باشید که به جای n نباید چیزی نوشته شود و خود حرف n باید قرار داده شود . هم a و هم b می توانند مقادیر منفی را بپذیرند . در این حالت ، فقط در صورتی تگ ها تطبیق می شوند که مقدار نهایی N یک عدد مثبت شود . اگر مقدار b منفی است قبل از آن به جای عملگر + از عملگر – استفاده کنید .
این selector تطبیق را از اولین تگ فرزند آغاز می کند :
دو selector ی که در مثال زیر مشاهده می کنید معادل هم هستند و ردیف های فرد تگ های table صفحه را تحت تاثیر قرار میدهد.

```
tr:nth-child(2n+1) {
      declarations
}
tr:nth-child(odd) {
      declarations
}
```

Selector زیر سه ردیف اول تمامی تگ های table را تحت تاثیر قرار میدهد :

```
tr:nth-child(-n+3) {
      declarations
}
```

Selector زیر تمامی تگ های p که اولین تگ فرزند پدرشان هستند را تحت تاثیر قرار می دهد :

```
p:nth-child(1) {
      declarations
}
```

Selector فوق معادل با p:first-child است .

:nth-last-child(N)

syntax
:nth-last-child({ *number expression* | odd | even }) **{** **declaration block** **}**

این selector بر خلاف selector قبل ، تطبیق را از آخرین تگ فرزند آغاز می کند .
Selector ذیل ، چهار تگ li آخر هر تگ ul یا ol را تحت تاثیر قرار می دهد :

```
li:nth-last-child(-n+4) {
      declarations
}
```

Selector زیر تمامی تگ های p که آخرین تگ فرزند پدرشان هستند را تحت تاثیر قرار می دهد .

```
p:nth-last-child(1) {
      declarations
}
```

:nth-of-type(N)

syntax
:nth-of-type({ number expression | odd | even }) **{** **declaration block** **}**

این کلاس ، تگ ها را بر اساس مکان آن ها در تگ پدرشان و بر اساس "نوع آنها" تطبیق می دهد . این selector تطبیق را از اولین تگ فرزند آغاز می کند .
Selector زیر دومین ، پنجمین ، هشتمین و .. تگ p را در هر تگ div تحت تاثیر قرار می دهد و با بقیه تگ های درون تگ div کاری ندارد :

```
div>p:nth-of-type(3n-1) {
      declarations
}
```

:nth-last-of-type(N)

syntax
:nth-last-of-type({ *number expression* | odd | even }) **{** **declaration block** **}**

این selector بر خلاف selector قبل ، تطبیق را از آخرین تگ فرزند آغاز می کند .
Selector زیر سه تگ img آخر تگی که نام آن gallery است راتحت تاثیر قرار می دهد :

```
#gallery>img:nth-of-type(odd) {
      declarations
}
```

:first-of-type

syntax

:first-of-type {
declaration block
}

این کلاس کاذب ، اولین تگ فرزند تگی با نوعی خاص را تحت تاثیر قرار می دهد و معادل با "nth-of-type(1)" است .

قواعد selector زیر بر روی اولین تگ p هر تگ div اعمال می شوند :

```
div>p:first-of-type {
	declarations
}
```

:last-of-type

syntax

:last-of-type {
declaration block
}

این کلاس ، عکس کلاس قبل رفتار می کند و آخرین تگ فرزند تگی با نوعی خاص را تحت تاثیر قرار می دهد .

:only-child

syntax

:only-child {
declaration block
}

این کلاس در صورتی یک زیر تگ را تحت تاثیر قرار می دهد که آن زیر تگ ، تنها تگ فرزند تگ پدرش باشد و تگ پدر ، هیچ تگ فرزند دیگری نداشته باشد .

قواعد selector زیر ، تنها در صورتی بر روی تگ li اعمال می شوند که آن تگ li تنها تگ فرزند تگ های ol و ul باشد .

```
li:only-child {
	declarations
}
```

:only-of-type

syntax

:only-of-child {
declaration block
}

این کلاس در صورتی یک زیر تگ را تحت تاثیر قرار می دهد که آن زیر تگ ، تنها تگ فرزند با آن نوع خاص در تگ پدرش باشد و تگ پدر ، هیچ تگ فرزند دیگری از آن نوع نداشته باشد .

قواعدselector زیر تنها در صورتی بر روی تگ img اعمال می شوند که آن تگ img تنها تگ img تگ پدرش باشد :

```
img:only-of-type {
	declarations
}
```

:empty

syntax

:empty {
declaration block
}

این کلاس ، تگ هایی که هیچ تگ فرزندی ندارند را تحت تاثیر قرار می دهد توجه داشته باشید که متن بین یک تگ نیز یک نوع تگ متنی محسوب و موجب می شود که این کلاس بر روی چنین تگ هایی نیز تاثیر نداشته باشد . حتی یک فاصله بین تگ آغاز و تگ پایان نیز از اعمال این کلاس به تگ جلوگیری می کند selectorی مانند "p:empty" موجب می شود که در کد های HTML زیر فقط اولین تگ p تحت تاثیر قرار گیرد .

```
<p></p>
<p> </p>
<p>Hello, World!</p>
```

:enabled

syntax

:enabled {
declaration block
}

این کلاس تگ هایی را تحت تاثیر قرار می دهد که اصطلاحا enable یا فعال هستند . تگ های فعال تگ هایی هستند که می توان بر روی آن ها فوکاس نمود ، کلیک کرد یا متنی را در آن ها وارد نمود .

قواعد selector زیر به تمامی تگ هایی از نوع input (همانند textbox ، button و...) که فعال هستند اعمال می شوند :

```
input:enabled {
      declarations
}
```

:disabled

syntax

:disabled {
declaration block
}

این کلاس تگ هایی را که disable (غیر فعال) هستند تحت تاثیر قرار می دهد .

قواعد selector زیر به تمامی تگ هایی از نوع input (همانند Textbox ، Button و...) که فعال نیستند اعمال می شوند :

```
input:disabled {
      declarations
}
```

:checked

syntax

:checked {
declaration block
}

این کلاس فقط به المان هایی اعمال می شوند که حالت تیک خوردن دارند . المان هایی مانند checkbox و radiobutton از این دست هستند . کلا تگ هایی که خاصیت selected یا checked دارند توسط این کلاس تحت تاثیر قرار می گیرند . دقت داسته باشید که تنها در صورت تیک خوردن آن المان ، قواعد این کلاس بر روی آن اعمال می شوند .

به عنوان مثال قواعد selector زیر بر روی تگی اعمال می شوند که نام آن confirm است . این تگ می تواند یک checkbox یا radiobutton باشد و حتما باید تیک خورده باشد :

```
#confirm:checked{
        قواعد
}
```

:not(s)

syntax

:not(simple selector) {
declaration block
}

این کلاس پارامتری را که می پذیرد نقض می کند . عبارت simple selector در سینتکس این کلاس می تواند یک selector ساده یا کلاس کاذب باشد اما نمی توان از خود این کلاس در پارامترش استفاده کرد . به عنوان مثال ، selector ی مانند input:not([type='submit']) تمامی تگ های صفحه به جز تگ هایی از نوع input را که از نوع submit هستند تحت تاثیر قرار می دهد .

قواعد selector زیر بر روی تمامی تگ های صفحه ، به جز تگ های table اعمال می شوند :

```
:not(table){
        قواعد
}
```

عناصر کاذب (pseudo Element) :

عناصر کاذب نیز به نوعی همانند کلا س های کاذب هستند . تنها تفاوتی که عناصر کاذب با کلاس های کاذب دارند که عناصر کاذب با :: شروع می شوند . در ادامه چند عنصر کاذب را بررسی خواهیم کرد .

::first-letter

syntax

::first-letter {
declaration block
}

این عنصر کاذب ، قالب متفاوتی را برای **اولین حرف** یک رشته در یک تگ تعیین می کند . توجه داشته باشید که این حرف یک عدد نیز می تواند باشد . تکه کد زیر را در نظر بگیرید :

```
<p>Hello, World!</p>
```

اگر selector زیر را بر روی کد قبل اعمال شود :

```
p::first-letter {
     color:red;
     background-color:yellow;
}
```

حرف H تحت تاثیر قواعد فوق قرار می گیرد و رنگ آن قرمز و پس زمینه آن زرد رنگ خواهد شد .

اگر تگ p به عنوان تگ فرزند تگ دیگری باشد و عنصر "::first-letter" برای تگ پدر آن تعیین شود باز هم حرف H تحت تاثیر قرار می گیرد و تفاوتی در این بین وجود ندارد . تکه کد زیر را در نظر بگیرید :

```
<div>
      <p>Hello World !</p>
</div>
```

هم selector ی مانند "p::first-letter" و هم selectorی همچون "div::first-line" حرف H را در تکه کد قبل تحت تاثیر قرار می دهد .

::first-line

syntax

::first-line {
declaration block
}

این عنصر کاذب ، قالب متفاوتی برای اولین خط رشته در یک تعیین می کند . این که این عنصر به چه شکل یک خط را تشخیص دهد بستگی به نحوه نمایش متن دارد . فاکتور هایی از جمله اندازه فونت متن و پهنای پنجره مرورگر از این جمله اند . به عنوان مثال اگر کاربر ، فونت متن یا پهنای پنجره مرورگر را تغییر دهد ، کاراکتر هایی از متن که به عنوان یک خط در نظر گرفته و قواعد این عنصر به آن ها اعمال می شود تغییر می کنند .

::before

syntax

::before {
declaration block
}

این عنصر برای قرار دادن متن یا تصویری قبل از یک تگ استفاده می شود . این عنصر همراه با خاصیت content استفاده می شود . توجه داشته باشید که متن یا تصویری که با استفاده از این عنصر ایجاد می شود ، به عنوان یکی از اجزای صفحه همانند سایر تگ ها د رنظر گرفته نمی شود . در مثال زیر عبارت "You Are Here" قبل از تگی با id='breadcrums' قرار می گیرد . همچنین فاصله ی این عبارت از سمت چپ تگ مذبور ، 0.5em تعیین شده است :

```
#breadcrums::before{
      Content:"You Are Here";
      Margin-rihgt : 0.5em
}
```

::after

syntax

::after {
declaration block
}

این عنصر برای قرار دادن متن یا تصویری بعد از یک تگ استفاده می شود . این عنصر همراه با خاصیت content استفاده می شود . توجه داشته باشید که متن یا تصویری که با استفاده از این عنصر ایجاد می شود ، به عنوان یکی از اجزای صفحه همانند سایر تگ ها د رنظر گرفته نمی شود .

در مثال زیر عبارت "cm" با رنگی با کد #cccccc بعد از یک تگ span که مقدار صفت class آن centimeters است قرار می گیرد :

```
Span.centimeters::after{
       Content:"cm";
      Color: #cccccc;
}
```

::selection

syntax

::selection {
declaration block
}

این عنصر که در CSS3 معرفی شده است نمایانگر متنی است که کاربر در قسمتی از صفحه انتخاب کرده است . این متن می تواند عبارتی در یک textbox هم باشد . فقط تعداد محدودی از قواعد می توانند با این عنصر استفاده شوند . مرورگر ها باید اجازه تغییر رنگ فونت (قاعده color) یا زمینه (قاعده background) متن انتخاب شده را بدهند . اما اجازه اعمال قواعد cursor و outline به متن انتخاب شده اختیاری است .

به عنوان مثال ، selector ای مانند textarea::selection ارجاعی است به تمامی متن انتخاب شده توسط کاربر در یک المان از نوع textarea .

مرجع قواعد در CSS

در این بخش با اکثر قواعد موجود در CSS به صورت دسته بندی شده خواهیم پرداخت و هر یک از مقادیر آن ها را نیز بررسی می کنیم .

قواعد کادر

قبل از اینکه به توضیح قواعد کادر بپردازیم لازم است با مفهوم کادر در **CSS** آشنا شویم .

Box Model (مدل کادری) چیست ؟

جالب است بدانید تمامی عناصر موجود در HTML در CSS به صورت یک کادر (جعبه) چهار گوش تصور می شوند . این مفهوم در مبحث طراحی و طرح بندی (قالب) صفحات استفاده می شود .

مدل کادری در اصل کادری است که در اطراف عناصر صفحه می تواند قرار بگیرد و شامل ویژگی های زیر باشد :

- Margin
- Border
- Padding
- Content

در تصویر زیر مدل کادری به وضوح نشان داده شده است :

اینک به توضیح هر یک از این بخش ها می پردازیم :

- Margin : فضای اطراف border را پوشش می دهد . این قسمت نمی تواند رنگ پس زمینه داشته باشد و کاملا شفاف است.
- border : این بخش فضای اطراف content و padding را پوشش می دهد و می تواند رنگ ، سبک و ضخامت خاص خود را داشته باشد .
- Padding : فضای اطراف content را پوشش می دهد . این بخش می تواند تحت تاثیر رنگ پس زمینه عنصر قرار بگیرد .
- Content : محتوای عنصر را در بر می گیرد . همان جایی که متن و عکس قرار می گیرد .

✓ **نکته : برای تنظیم عرض و ارتفاع حقیقی یک عنصر درک مفهوم مدل کادری بسیار مهم است .**

قواعد مربوطه به تنظیم ابعاد عنصر :

- Height **:** این قاعده ارتفاع فضای اشغال شده توسط تگ را مشخص می کند . این فاصله شامل فاصله محتویات درون تگ از حاشیه آن ، اندازه حاشیه های تگ و فاصله حاشیه های آن از تگ های اطراف نمی شود .

- min-height **:** این قاعده ، حداقل ارتفاع فضای اشغالی توسط تگ را تعیین می کند . ارتفاع یک تگ هیچ گاه کمتر از میزان تعیین شده از طریق قاعده min-height نمی شود. اما حداکثر ارتفاع می تواند بر اساس محتویات درون تگ متغیر باشد . از این قاعده معمولا برای مطمئن شدن از اینکه حداقل ارتفاع تگی حتی در صورت نداشتن محتویاتی از مقداری کمتر نشود استفاده می شود .

- max-height **:** این قاعده دقیقا بالعکس قاعده min-height رفتار می کند یعنی حداکثر ارتفاع فضای اشغالی توسط تگ را تعیین می کند .

- width **:** این قاعده ، پهنای فضای اشغال شده توسط تگ را مشخص می کند .

- min-width **:** این قاعده حداقل پهنای فضای اشغال شده توسط تگ را تعیین می کند .

- max-width : این قاعده حداکثر پهنای فضای اشغال شده توسط تگ را تعیین می کند .

فاصله از اطراف :

- margin-top **:** این قاعده میزان فاصله حاشیه بالای تگ را از تگ پدرش یا تگی که در بالای آن قرار دارد تعیین می کند .

- margin-right **:** این قاعده میزان فاصله حاشیه سمت راست تگ را از تگ پدرش یا تگی که دربالای آن قرار دارد تعیین می کند .
- margin-left **:** این قاعده میزان فاصله حاشیه سمت چپ تگ را از تگ پدرش یا تگی که دربالای آن قرار دارد تعیین می کند .
- margin-bottom **:** این قاعده میزان فاصله حاشیه سمت پایین تگ را از تگ پدرش یا تگی که دربالای آن قرار دارد تعیین می کند .

- margin : این قاعده ، میانبری برای چهار قاعده فوق می باشد . در واقع از این قاعده می توان برای تنظیم هر چهار قاعده فوق به یکباره استفاده کرد . به مثال زیر توجه کنید :

```
img {
      margin-top : 20px;
      margin-right : 30px;
      margin-bottom : 10px;
      margin-left : 15px;
}
```

این دستور میزان فاصله هر یک از چهار طرف عکس های صفحه را به صورت جداگانه تعیین می کند . ما می توانیم به جای این دستور از margin به صورت زیر استفاده کنیم :

```
img {
      margin : 20px 30px 10px 15px ;
}
```

- ✓ **نکته : به نحوه مقدار دهی به قاعده margin که در جهت عقربه های ساعت (یعنی ابتدا بالا ، راست ، پایین و چپ) است و بین هر مقدار فاصله ای وجود دارد دقت کنید .**

فاصله از داخل :

- padding-top
- padding-right
- padding-bottom
- padding-left
- padding

این قواعد هم میزان فاصله محتویات داخل یک تگ (content) را از حاشیه های بالا ، راست ، پایین و چپ کنترل می کنند .
padding هم میانبری برای هر ۴ قاعده است و نحوه مقدار دهی آن همانند قاعده margin خواهد بود .

نکته بسیار مهم :

موقعی که ما از قواعد width و height در CSS برای یک عنصر استفاده می کنیم در واقع فقط عرض و ارتفاع بخش content آن را تعیین کرده ایم . برای بدست آوردن عرض و ارتفاع حقیقی یک عنصر (یعنی آنچه در صفحه نمایش داده می شود) باید مقادیر تعیین شده برای قواعد border , padding , margin را نیز به آن ها اضافه کنیم . به عنوان مثال قواعد زیر را که برای یک عنصر تعیین شده است در نظر بگیرید :

```
width:250px;
padding:10px;
border:5px solid gray;
margin:10px;
```

عرض حقیقی این عنصر در صفحه 300px خواهد بود .

برای بدست آوردن عرض حقیقی یک عنصر باید به صورت زیر عمل کنیم :

```
element width = width + left padding + right padding + left border + right
border + left margin + right margin
```

این موضوع برای ارتفاع یک عنصر نیز صادق است . ارتفاع حقیقی یک عنصر باید به صورت زیر محاسبه شود :

```
element height = height + top padding + bottom padding + top border + bottom
border + top margin + bottom margin
```

www.SoftGozar.com

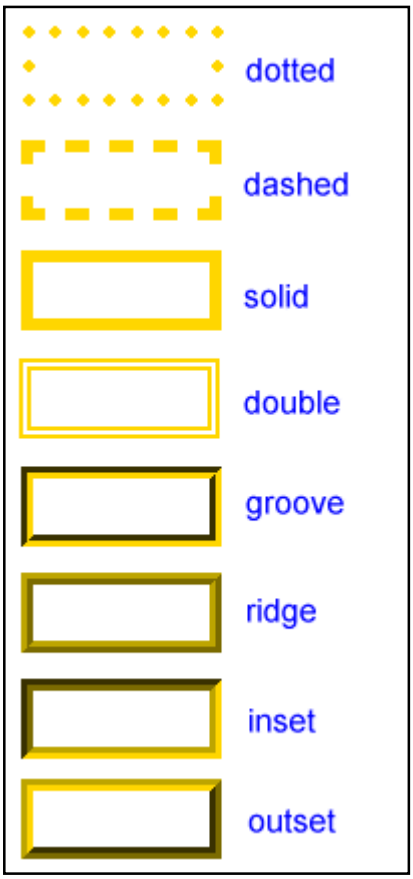
حاشیه ها و Outline ها

قواعد حاشیه ها ، اجازه کنترل حاشیه های تگ را می دهند . حاشیه تگ مکانی است میان padding و margin . به عبارتی دیگر مکانی بین محتویات داخل یک تگ و مرز آن ها با تگ های دیگر .

Outline ها به دور حاشیه ها کشیده می شوند و در حقیقت یک حاشیه دیگر را برای تگ بوجود می آورند . تفاوت Outline ها با حاشیه ها این است که Outline ها بر خلاف حاشیه ها ، فضایی که اشغال می کنند جزء فضای تگ محسوب نمی شود و در صورتی که قسمتی از تگ مجاور با تگ جاری تلاقی داشته باشد بر روی تگ مجاور قرار می گیرد .

- border-top-color : این قاعده رنگ حاشیه بالای تگ را تعیین می کند و یک رنگ را به عنوان مقدار می پذیرد . یکی از مقادیری که این قاعده می پذیرد ، کلمه کلیدی transparent است که در این حالت برای حاشیه تگ ، رنگی در نظر گرفته نمی شود اما همچنان فضایی را به میزانی که با قاعده border-width برای تگ تعیین شده است اشغال می کند .

- border-top-style : این قاعده طرح حاشیه بالای تگ را تعیین می کند . در تصویر زیر مقادیر این قاعده و طرح هر یک از آن ها نشان داده شده است :

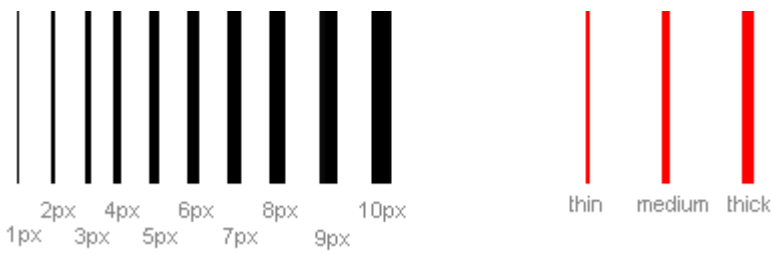


به غیر از مقادیری که در تصویر نشان داده شده است این خاصیت دو مقدار زیر را با کارکرد های متفاوت به عنوان مقدار می پذیرد :

- none : با این مقدار هیچ حاشیه ای برای تگ در نظر گرفته نمی شود . در این حالت مقدار قاعده border-width نیز برابر با صفر در نظر گرفته خواهد شد .
- hidden : این مقدار موجب می شود هیچ حاشیه ای برای تگ در نظر گرفته نشود . اما ضخامت border که در قاعده border-width تعیین شده است در نظر گرفته خواهد شد .

- border-top-width : این قاعده ، اندازه پهنای حاشیه بالا را تعیین می کند و به دو شکل مقدار دهی می شود . یا با تعیین یکی از واحد های اندازه گیری همانند px,pt,em و ... یا با قبول یکی از کلمات کلیدی thin , medium , thick .

مقادیر درصدی و منفی مجاز نیستند . در تصویر زیر ضخامت بعضی از مقادیر فوق نشان داده شده است :



www.SoftGozar.com

- border-top : این قاعده ، میانبری برای سه قاعده پیشین است و می توان به یکباره آن ها را با این قاعده مشخص نمود .
سینتکس کاربرد این قاعده به صورت زیر است :

```
border-top: { [border-width] [border-style][border-color] | inherit } ;
```

✓ **توجه : برای هر چهار طرف یک تگ ، هر چهار قاعده فوق تعریف شده است که به منظور جلوگیری از تکرار از توضیح آنها خودداری می کنیم .**

- border-color : این قاعده میانبر ، رنگ حاشیه هر چهار سمت تگ را به یکباره تعیین می کند و همان مقادیر قاعده border-top-color را می پذیرد .
- border-style : این قاعده میانبر ، طرح حاشیه هر چهار سمت تگ را به یکباره کنترل می کند و همان مقادیر قاعده border-top-style را می پذیرد .
- border-width : این قاعده میانبر ، اندازه پهنای حاشیه هر چهار سمت تگ را به یکباره تعیین می کند و همان مقادیر قاعده border-top-width را می پذیرد .
- border : این قاعده میانبری برای هر سه قاعده border-width ، border-style و border-color است و با استفاده از آن می توان این سه قاعده را یکجا به چهار طرف عنصر اعمال نمود . نحوه استفاده از این قاعده به صورت زیر است :

```
border: { [border-width] [border-style][border-color] | inherit } ;
```

نکته : قبل از وارد شدن به بحث Outline ها ذکر این نکته ضروری است که Outline ها نمی توانند به حاشیه ای خاص اعمال شوند . Outline ی که تعریف می شود به حاشیه های هر چهار سمت تگ اعمال خواهد شد .

- outline-color : این قاعده ، رنگ Outline ی که به دور حاشیه کشیده می شود را تعیین می کند
- outline-style : این قاعده طرح Outline ی را که به دور حاشیه کشیده می شود تعیین می کند و همان مقادیر قاعده border-top-style را می پذیرد .
- outline-width : این قاعده اندازه پهنای Outline ی را که به دور حاشیه کشیده می شود تعیین می کند و همان مقادیر قاعده border-top-width را می پذیرد .
- outline : این قاعده میانبری برای سه قاعده فوق است و می بایست به صورت زیر مقدار دهی شود :

```
Outline : { [outline-width] [outline-style][outline-color] | inherit } ;
```

قواعد طرح بندی :

در این قسمت با قواعدی آشنا می شویم که اجازه کنترل رویت ، مکان و رفتار کادر ها را برای تگ هایی که درون آنها قرار می گیرند مشخص می کنند . قواعد این بخش جزء مهمترین قواعد برای طراحی قالب در سایت ها به شمار می روند .

display

این قاعده نوع کادری را که ایجاد می شود مشخص می کند . در مثال زیر لینک هایی که در تگی قرار دارند که مقدار صفت class آنها برابر menu است به جای آنکه inline در نظر گرفته شوند ، همانند یک بلوک درنظر گرفته می شوند :

```
.menu a{
     display : block;
}
```

پارامتر هایی که این خاصیت می گیرد در زیر توضیح داده شده است :

- ✓ block : طرح کادر به صورت یک بلوک خواهد بود . معمولا در فضای بالا و پایین تگ های نوع block کمی فضای خالی وجود دارد و همیشه از یک خط جدید شروع می شوند .
- ✓ inline : مقصود از این مقدار ، قرار گرفتن تگ ، درون بلوک جاری و در همان خطی است که بلوک مجاور قرار دارد .

نکته : برای آگاهی از تفاوت های عناصر block-level و inline-level می توانید به جزوه "مرجع آموزش HTML و XHTML" مراجعه نمایید .

- ✓ inline-block : همانند مقدار قبل است با این تفاوت که به عنوان یک بلوک با آن رفتار می شود بیشترین کاربرد این مقدار هنگامی است که می خواهید به تگی که درون تگ دیگر قرار دارد پهنایی اختصاص دهید .
- ✓ list-item : این مقدار موجب می شود تا تگ فرزند به صورت یک گزینه از لیست درون تگ پدرش نمایش داده شود .
- ✓ none : این مورد باعث می شود که تگ به عنوان یک کادر در نظر گرفته نشود . درواقع این مقدار باعث **عدم نمایش تگ و تمامی فرزندان آن** می شود . از این قاعده برای حذف عناصر از صفحه مخصوصا در جاوااسکریپت زیاد استفاده می شود .

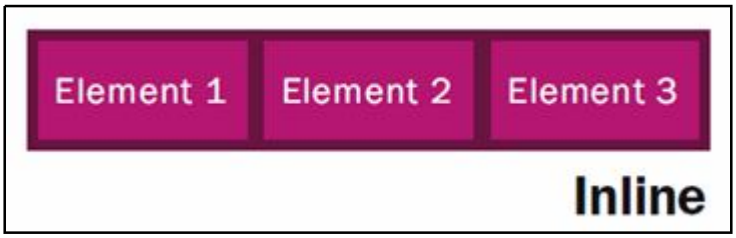
نکته : این قاعده مقادیر دیگری نیز می پذیرد که به دلیل عدم پشتیبانی درست و همگانی مرورگر های موجود از بررسی آن ها خودداری می کنیم .

اینک چند مثال از بعضی از مقادیر این قاعده را بررسی می کنیم . فرض کنید ما تگ های زیر را در یک صفحه HTML داریم :

```
<span id="e1">Element 1</span>
<span id="e2">Element 2</span>
<span id="e3">Element 3</span>
```

دستور **#e1, #e2, #e3 { display: block; }** موجب می شود این عناصر به صورت زیر نمایش داده شوند :

همچنین دستور #e1, #e2, #e3 { display: inline; } عناصر فوق را به صورت زیر نمایش خواهد داد :



اما دستورات زیر موجب می شود عنصر e2 نمایش داده نشود و در واقع از صفحه حذف شود :

```
#e1, #e3 { display: block; }
#e2 { display: none; }
```

حاصل کد فوق به صورت زیر خواهد بود :

position

این قاعده همانند قاعده float ، نحوه قرار گیری کادر را مشخص می کند . به کادر هایی که مقادیری غیر از static برای این قاعده آنها تعیین شده باشد ، اصطلاحا **کادر های مکان دار** گفته می شود .

در مثال زیر تگی با نام sidebar ، دقیقا در گوشه بالا و سمت راست کادری که در آن وجود دارد قرار می گیرد :

```
#sidebar {
      position: absolute;
      top: 0;
      right: 0;
}
```

پارامتر ها :

- ✓ absolute : این مقدار مکان قرارگیری کادر را به طور مطلق نسبت به **کادر دربر گیرنده** آن تعیین می کند . مکان کادر می تواند با استفاده از یکی از چهار قاعده top ، right ، bottom و left تعیین شود .
- ✓ fixed : این مقدار مکان قرار گیری کادر را به طور مطلق نسبت به **فضای کل صفحه** تعیین میکند .
- ✓ relative : کادر هایی که این مقدار را دارند ابتدا مکان قرارگیری آنها بر طبق نحوه چیدمانشان درکنار تگ های دیگر به طور معمول تعیین می شود و سپس از آن مکان ، قواعد top ، right ، bottom و left مکان دقیق کادر را مشخص می کند .
- ✓ static : کادر هایی که مکان آن ها به طور معمول با توجه به نحوه قرارگیری آن ها درکنار کادر ها یا تگ های دیگر مشخص می شود این مقدار را می پذیرند .

float

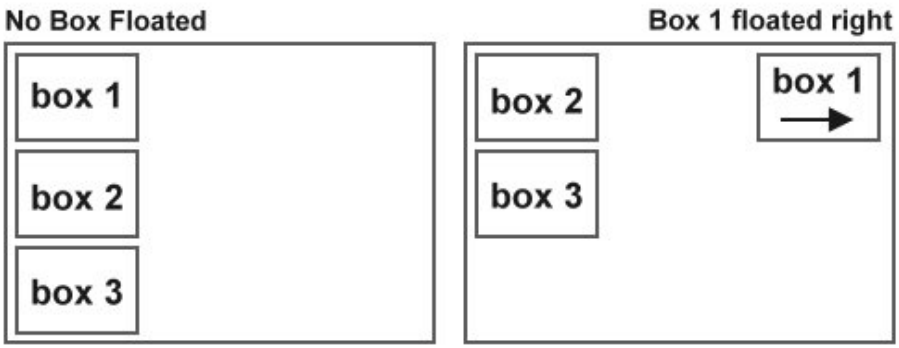
float به حالتی گفته می شود که تگی حالت شناور به خود می گیرد و نسبت به جهتی متمایل می شود . این قاعده اولا مشخص می کند که آیا تگی حالت شناور داشته باشد یا خیر و دوما در صورت شناور بودن آیا به سمت چپ متمایل شود یا به سمت راست .

پارامتر ها :

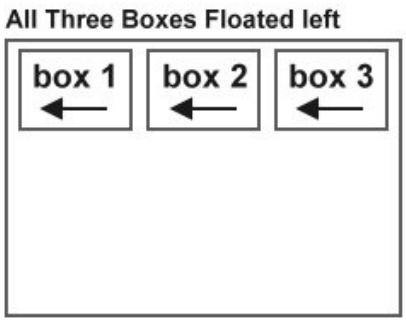
- ✓ left : تگ را در سمت چپ پدرش قرار می دهد .
- ✓ right : تگ را در سمت راست تگ پدرش قرار می دهد .
- ✓ none : تگ را بدون حالت شناور در نظر می گیرد.

این قاعده یکی از مهمترین قواعدی است که این روز ها در طرح بندی صفحات نیز کاربرد داشته و ارزش بررسی بیشتر را دارد : شناور ساختن عناصر (که در اصطلاح به آن **floating** می گویند) یکی از روش های تعیین موقعیت عناصر داخل عناصر دیگر است . یک عنصر شناور می تواند به سمت چپ یا راست عنصر در برگیرنده خود برود تا لبه های آن با لبه های عنصر پدر (همان عنصر در برگیرنده) یا عناصر شناور دیگر مماس شود .

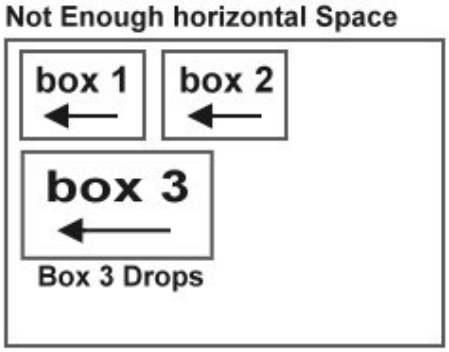
تصویر زیر نشان می دهد که در صورتی که قاعده float را برای box 1 برابر right قرار دهیم این عنصر به سمت راست عنصر پدر خود می رود و لبه سمت راست آن با لبه سمت راست پدرش مماس می شود :



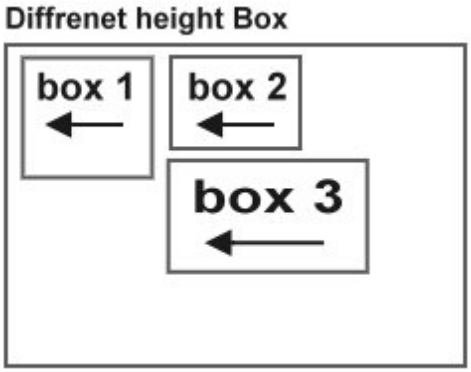
در صورتی که ما هر سه box را به یک جهت شناور کنیم هر سه عنصر تا جایی که در جهت افقی در پدر خود فضا وجود داشته باشد در کنار سایر box ها قرار خواهد گرفت . در تصویر زیر هر سه box به سمت چپ شناور شده اند در نتیجه box 1 به سمت چپ متمایل می شود تا لبه چپ آن با عنصر پدر مماس شود و دو box دیگر هم چون فضا وجود دارد به سمت چپ شناور می شوند تالبه چپ آن ها با box های قبلی مماس شود. (از این مورد در طراحی منو های افقی در بسیاری از موارد استفاده می شود .) در تصویر زیر قاعده float هر سه box برابر left تعیین شده است :



در صورتی که در عنصر پدر فضای کافی برای اینکه که سه box در جهت افقی قرار بگیرند وجود نداشته باشد عناصر بعدی تا جایی پایین خواهند رفت که فضای کافی بدست آورند مانند شکل ذیل :



در صورتی که box ها ارتفاع های متفاوتی داشته باشند ممکن است در هنگام پایین آمدن به لبه های box های قبلی بچسبند . به تصویر زیر که گویای موضوع است توجه کنید :



Clear

با استفاده از این قاعده می توان تعیین کرد که آیا یک تگ می تواند در کنار تگی که قاعده float را دارد قرار بگیرد یا خیر .

پارامتر ها :

- ✓ left : هیچ تگ شناوری نمی تواند در سمت چپ تگ مورد نظر قرار گیرد .
- ✓ right : هیچ تگ شناوری نمی تواند در سمت راست تگ مورد نظر قرار گیرد .
- ✓ both : هیچ تگ شناوری نمی تواند در سمت راست یا چپ تگ مورد نظر قرار گیرد .
- ✓ none : تگ های شناور می توانند درکنار تگ مورد نظر قرار بگیرند .

Visibility

این قاعده مشخص می کند که یک تگ نمایش داده شود یا نه . توجه داشته باشید که اگر حتی تگی با استفاده از این قاعده نمایش داده نشود ، بازهم فضا را اشغال می کند و بر روی عناصر دیگر صفحه نیز اثر خواهد گذاشت . دقت داشته باشید که اگر تگ هایی به عنوان فرزندان تگ دیگری باشندکه از این تگ با استفاده از مقدار hidden برای قاعده visibility خود مقدار دهی شده باشند که موجب می شود تگ نمایش داده نشود اما تگ های فرزند درون این تگ ، مقدار visible را برای این قاعده داشته باشند تگ های فرزند نمایش داده خواهند شد .

پارامتر ها :

- ✓ visible : تگ نمایش داده می شود .
- ✓ hidden : تگ نمایش داده نمی شود .
- ✓ collapse : این مقدار فقط برای برخی از اجزای تشکیل دهنده تگ table کاربرد دارد . اجزایی همانند ردیف ها ، گروهی از ردیف ها ، ستون ها و گروهی از ستون ها . استفاده از این مقدار برای این اجزا موجب خواهد شد تا نمایش داده نشوند و فضای آنها بوسیله دیگر تگ های مجاورشان پر خواهد شد .

top , right , bottom , left

این قواعد برای تگ هایی که مقدار قاعده position آن ها برابر با absolute یا relative است به ترتیب میزان فاصله حاشیه بالا ، راست ، پایین و چپ را کنترل می کند .

z-index

اگر چه که به نظرمی رسد که تگ ها ی یک صفحه حالتی دو بعدی (شامل محور x و y) را دارند اما در حقیقت این عناصر ، سه بعدی هستند و دارای محور z نیز هستند . مقدار z یک عنصر با استفاده از قاعده z-index تعیین می شود .

از آنجا که مکان دو کادر ممکن است با هم تداخل داشته باشد ، حق تقدم قرار گیری کادری بر روی کادر دیگر با استفاده از قاعده z-index مشخص می شود . در مثال زیر تگی با نام warning حق تقدم بیشتری برای نمایش نسبت به تگ های مجاورش خواهد شد :

```
#warning {
      position: absolute;
      z-index: 1;
}
```

پارامتر ها :

یک عدد صحیح که منفی نیز می تواند باشد و نمایانگر سطح تقدم کادر نسبت به کادر های دیگر است . مقدار auto برای قاعده z-index مشخص می کند که عنصر دارای همان سطح z ایست که تگ پدر آن دارد و مقدار جدیدی را نخواهد پذیرفت .

Overflow

این قاعده نحوه رفتار یک تگ را در زمانی که اندازه محتویات درونش از ابعاد آن تجاوز می کند مشخص می کند .

پارامتر ها :

- ✓ auto : مرورگر ها با این مقدار ، در زمانی که نیاز باشد به تگ scrollbar اضافه می کنند و تا زمانی که محتویات درون تگ از ابعاد آن تجاوز نکرده اند scrollbar ظاهر نمی شود .
- ✓ hidden : موجب می شود تا محتویاتی که از ابعاد تگ پدرشان تجاوز می کنند نمایش داده نشوند .
- ✓ scroll : این مقدار هم مانند hidden عمل می کند اما در عین حا ل یک scrollbar عمودی نیز به تگ اضافه می کند تا کاربر بتواند با پیمایش آن ، محتویاتی که نمایش داده نشده اند را نیز ببیند .
- ✓ visible : موجب می شود تا محتویات متجاوز از ابعاد تگ پدر کاملا نمایش داده شوند .

کنترل میزان شفافیت عناصر با استفاده از قاعده Opacity :

یکی از جالب ترین و پرکاربردترین قواعدی که در CSS 3 ارائه شده است قاعده opacity است که از آن برای کنترل میزان شفافیت عناصر استفاده می شود . این قاعده می تواند مقداری بین 0.0 (شفاف ترین حالت) تا 1.0 (واضح ترین حالت) را به عنوان مقدار بپذیرد. سینتکس استفاده از این قاعده باید به صورت opacity : x است که x می تواند یکی از مقادیر فوق باشد . به عنوان مثال در دستور زیر میزان شفافیت تمامی عکس های صفحه 0.8 در نظر گرفته شده است :

```
img {
      opacity : 0.8;
}
```

نکته : این نوع استفاده از این قاعده در مرورگر Internet Explorer پشتیبانی نمی شود و به جای آن باید از دستور filter:alpha(opacity=x) استفاده کنیم که در این قاعده x می تواند بین 0 (شفاف ترین) تا 100 (واضح ترین) باشد . چنانچه می خواهید این قاعده در تمامی مرورگر ها به درستی کار کند باید از هر دو روش به صورت همزمان استفاده کنیم . مثال بالا به روش زیر در تمامی مرورگر ها کار می کند :

```
img {
      opacity : 0.8;
      filter:alpha(opacity=80);
}
```

قواعد مربوط به انواع لیست ها :

قواعد این دسته اجازه تعیین نحوه نمایش لیست ها را در صفحه می دهند .

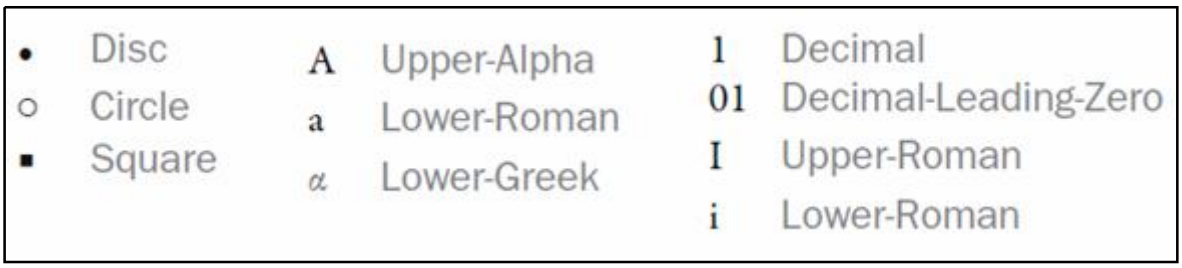
list-style-type

این قاعده شکل کنار هر یک از گزینه های لیست را تعیین می کند .این شکل برای کادر هایی که مقدار قاعده display ان ها برابر با list-item است نیز نمایش داده می شود . این قاعده تنها در صورتی به یک گزینه از لیست اعمال می شود که قاعده list-style-image آن گزینه برابر با none یا آدرسی باشد که وجود ندارد .

رنگ شکل ها همان رنگی است که برای قاعده color هر گزینه از لیست تعیین می شود .

پارامتر ها :

این قاعده مقادیری را که در شکل زیر نمایش داده شده است را به عنوان مقدار می پذیرد . البته دقت کنید که همه مرورگر ها از همه این مقادیر پشتیبانی نمی کنند . اما عموما سه مقدار disc , square , circle در تمام مرورگرها به خوبی پشتیبانی می شود :



یکی دیگر از مقادیری که این قاعده می پذیرد مقدار none است که موجب می شود هیچ شکلی کنار آیتم های لیست نمایش داده نشود . این مورد در ساخت منو های افقی و عمودی کاربرد زیادی دارد .

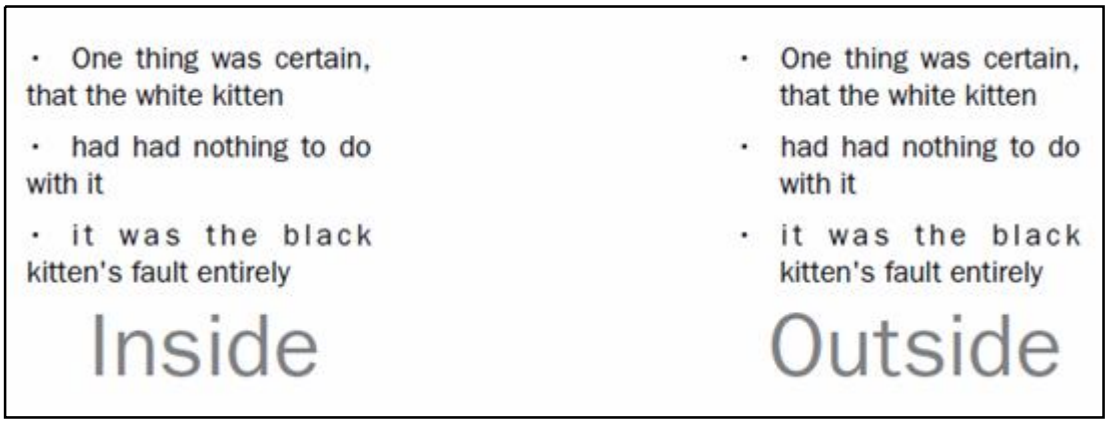
list-style-position

این قاعده مکان قرار گیری شکل کنار هر گزینه در لیست را تعیین می کند .

پارامتر ها :

- ✓ Outside : این مقدار موجب می شود تا شکل های کنار هر گزینه ، خارج از کادری که لیست در آن قرار گرفته است ظاهر شوند .
- ✓ Inside : این مقدار موجب می شود تا شکل های کنار هر گزینه ، داخل از کادری که لیست در آن قرار گرفته است ظاهر شوند.

نحوه عملکرد هر یک از مقادیر فوق در شکل زیر نشان داده شده است :



list-style-image

با استفاه از این قاعده می توان تصویری را به عنوان شکلی که در کنار هر یک از گزینه های لیست ظاهر می شود تعیین کرد . در صورت تعیین تصویر با این قاعده ، قاعده list-style-type بی اثر خواهد بود.

در مثال زیر تصویری در کنار تمامی گزینه هایی که درون تگی به نام links هستند قرار می گیرد :

```
#links li {
      list-style-image: url("/images/link.png");
}
```

یکی از مقادیری که این قاعده می گیرد مقدار none است که موجب می شود هیچ تصویری در نظر کرفته نمی شود و از شکلی که در قاعده list-style-type تعیین شده است استفاده خواهد شد .

list-style

این قاعده میانبری برای سه قاعده پیشین است است در صورتی که یکی از مقادیر پارامتر های این قاعده ذکر نشود ، مقدار پیش فرض آن در نظر گرفته خواهد شد . سینتکس کلی این قاعده به شکل زیر است :

```
list-style: { list-style-type list-style-position list-style-image | inherit } ;
```

قواعد مربوط به تعیین رنگ متن و پس زمینه عناصر :

در این بخش با قواعدی آشنا خواهیم شد که بر نحوه تعیین رنگ پیش زمینه و پس زمینه تگ ها و کنترل مکان تصویر پس زمینه نقش دارند .

background-color

این قاعده رنگ پس زمینه یک تگ را مشخص می کند . پس زمینه یک تگ ، ناحیه ای است که یا صریحا توسط قواعد width و height برای آن تعیین شده است و یا این محدوده به طور ضمنی توسط محتویات درون تگ تعیین می شوند و از قبل قابل تعیین نیستند .

پارامتر ها :

- ✓ پارامتر color این قاعده رنگی را به عنوان پس زمینه تعیین می کند .
- ✓ پارامتر transparent در حقیقت رنگی را برای پس زمینه در نظر نمی گیرد . با این مقدار می توان تگی را که در زیر تگ دیگری قرار دارد مشاهده نمود . همانند اینکه از پشت شیشه به جسمی نگاه کنید !!!

background-image

این قاعده ، تصویر پس زمینه محیط احاطه شده توسط تگ را تعیین می کند . تصویر پس زمینه بر روی رنگ پس زمینه قرار می گیرد .
در مثال زیر تصویری به عنوان پس زمینه به تگی با نام example نسبت داده می شود :

```
#example {
      background-image:url(images/bg.gif);
}
```

پارامتر ها :

- ✓ url : آدرسی است که تصویر مورد نظر در آنجا قرار دارد
- ✓ none : هیچ تصویری را به عنوان پس زمینه در نظر نمی گیرد.none ، مقدار پیش فرض است .
- ✓ inherit : موجب می شود تا تصویر پس زمینه از پدر تگ به آن ارث برسد .

نکته : پیشنهاد می شود که در صورت تعیین عکس پس زمینه برای یک تگ ، رنگ پس زمینه ای متناسب با تصویر آن تگ نیز انتخاب نید تا در صورتی که تصویر پس زمینه به هر علتی نمایش داده نشد ، رنگ پس زمینه ای که نمایش داده می شود ، متناسب با رنگ های دیگر استفاده شده در سایت باشد .

background-repeat

این قاعده در ابتدا مشخص می کند که آیا تصویر پس زمینه ی تگ باید تکرار شود ؟ و اگر باید تکرار شود در کدام جهت ؟ د رجهت x ، در جهت y یا هر دو ؟
پارامتر ها :

- ✓ repeat : تصویر زمینه هم در جهت x و هم در جهت y تکرار می شود . (مقدار پیش فرض)
- ✓ repeat-x : تکرار فقط در جهت x
- ✓ repeat-y : تکرار فقط در جهت y
- ✓ no-repeat : هیچ تکراری برای پس زمینه درنظر نمی گیرد . و فقط یک مرتبه تصویر را در مکانی که به وسیله قاعده background-position تعیین شده است قرار می دهد .

background-position

این قاعده مکان اولیه تصویر پس زمینه تگ را تعیین می کند . اگر این قاعده در نظرگرفته نشود ، تصویر به طور پیش فرض در بالا و ست چپ محوطه احاطه شده توسط تگ قرار می گیرد . در مثال زیر مکان اولیه تصویر پس زمینه تگی با نام example ، 100px از سمت چپ و 200px از بالای محدوده تگ تعیین شده است :

```
#example{
      background-position: 100px 200px;
}
```

پارامتر ها :

این قاعده یک یا دو مقدار را که می تواند بر حسب درصد ، عدد صحیح همراه با نوع واحد آن یا کلمه کلیدی باشند می پذیرد . اگر فقط یک مقدار تعیین شود ، مقدار دوم center در نظر گرفته می شود . اگر دو مقدار تعیین شده باشد و حداقل یکی از آنها کلمه کلیدی (top , right , bottom , left , center) نباشد ، اولین مقدار مکان تصویر در محور افقی و دومین مقدار ، مکان تصویر در محور عمودی است .

برای تعیین مکان تصویر در محور افقی ، از کلمات کلیدی left,center,right و در محور عمودی از top,center,bottom استفاده کنید .

background-attachment

این قاعده مشخص می کند که آیا تصویر پس زمینه همراه با پیمایش تگ با scrollbars جابجا شود یا در جای خود ثابت بماند . این قاعده برای تصویر پس زمینه صفحه که به تگ body اعمال می شود و همچنین تگ هایی که مقدار قاعده overflow آنها برابر با scroll است کاربرد دارد .

پارامتر ها :

مقدار scroll برای این قاعده در هنگامی که برای تگ body در نظر گرفته شود موجب می شود تا تصویر همراه با پیمایش صفحه نیز جابجا شود . به این نکته توجه داشته باشید که که مقدار scroll برای این قاعده در زمانی که برای تگی استفاده شود که مقدار قاعده overflow آن برابر با scroll است موجب می شود که تصویر پس زمینه ی آن تگ ، با پیمایش تگ جابجا نشود و به جای آن ، با پیمایش صفحه درون محوطه تگ جابجا شود .

مقدار fixed موجب می شود تا با پیمایش صفحه یا تگ ، تصویر پس زمینه آن در جای خود ثابت باقی بماند .

background

این قاعده میانبری برای قواعد ذکر شده فوق است . و با استفاده از آن می توان تمامی این قواعد را یکجا تعیین کرد . نحوه استفاده از این قاعده باید به ترتیب زیر باشد :

```
background: { background-color background-image background-repeat background-
attachment background-position} ;
```

در مثال زیر تمامی مقادیر تعیین شده برای قاعده background به تگی با نام example و تمامی تگ های فرزند آن اعمال می شوند :

```
#example{
      Background : #fff url(image.gif) no-repeat fixed left top;
}
```

Color

این قاعده رنگ پیش زمینه تگ را تعیین می کند . این قاعده در حقیقت رنگ متون نوشته شده در تگ را مشخص می کند .
در مثال زیر ، رنگ متونی که درون تگی با نام example قرار دارند به رنگ قرمز خواهد بود :

```
#example{
      color: red;
}
```

قواعد چاپ :

با استفاده از قواعد این دسته می توان ظاهر متون و نوشته های صفحه را تنظیم کرد .

font-family

این قاعده ، فونت نمایشی مورد استفاده برای متن را مشخص می کند . لیستی از نام فونت ها می توانند به عنوان ورودی به این قاعده پاس داده شوند . فونت باید بر روی سیستم کاربر وجود داشته باشد . مرورگر اولین فونت در لیست را که بر روی سیستم پیدا کرد ، آن را به عنوان فونت متن در نظر می گیرد .
در مثال زیر تمامی تگ های صفحه که زیر مجموعه ای از تگ html هستند به جز آنهایی که نوع فونتشان صریحا مشخص شده است در ابتدا فونتی با نام helvetica را می پذیرد ، گر این فونت بر روی سیستم کاربر وجو د نداشت ، فونت arial برای آنها درنظر گرفته می شود و ...

```
html {
      font-family: Helvetica, Arial, "Luxi Sans", sans-serif;
}
```

- ✓ **نکته : دقت داشته باشید که بعضی از مقادیری که می توان به این قاعده داد چند کلمه ای است و باید حتما آن ها را داخل کوتیشن ها قرار بدهید . همان طور که در مثال فوق هم می بینید .**

font-size: { absolute-size | relative-size | length| percentage | inherit }

این قاعده اندازه فونت متن تگ را مشخص میکند .

پارامتر ها :
پارامتر absolute-size در سینتکس این قاعده مقداری ثابت است که با استفاده از تعدادی کلمات کلیدی مشخص می شود . این کلمات عبارتند از :

xx-small , x-small , small , medium , large , x-large , xx-large

اندازه دقیق مقادیر فوق در مستندات CSS مشخص نشده است اما اندازه هر یک به ترتیب از بالا به پایین باید مساوی یا بزرگتر از مقدار قبل خود باشد .

پارامتر relative-size مقداری نسبی است که با استفاده از کلمات کلیدی زیر مشخص می شود :

- ☒ smaller : اندازه فونت متن را کوچکتر از مقدار قاعده font-size به ارث رسیده به تگ فرزند در نظر می گیرد .
- ☒ larger : اندازه فونت متن را کوچکتر از مقدار قاعده font-size به ارث رسیده به تگ فرزند در نظر می گیرد .

اگر مقدار ، بر حسب عددی صحیح با نوع واحد em یا ex باشد ، مقدار واقعی بر حسب اندازه فونت تگ پدر به دست می آید . این مورد برای مقادیر درصدی نیز صدق می کند . در ضمن مقادیر منفی مجاز نیستند .

font-weight: { 100 … 900 | bold | bolder | lighter | normal}

این قاعده میزان ضخامت فونت متن تگ را مشخص می کند .

توجه داشته باشید که اکثر فونت ها ، تنها از تعداد محدودی مقادیر ضخامت ها که در این قاعده تعریف شده است پشتیبانی می کنند . معمولا دو مقدار normal و bold پشتیبانی می شوند .

پارامتر ها :

در مقادیر ۱۰۰ تا ۹۰۰ هر مقدار از مقدار قبلی خود ضخامت بیشتری دارد . کلمات کلیدی زیر نیز می توانند به این قاعده اعمال شوند :

- ✓ Bold : معادل با 700
- ✓ Bolder :ضخامت بیش از ضخامت فونت تگ پدر در نظر گرفته می شود .
- ✓ Lighter : ضخامت کمتر از ضخامت فونت تگ پدر در نظر گرفته می شود .
- ✓ Normal : معادل با 400

font-style

با استفاده از این قاعده می توان ظاهر فونت متن تگ را تعیین نمود .

پارامتر ها :

- ✓ italic : حروف متن به صورت کج نمایش داده می شود .
- ✓ normal : حروف متن به صورت معمولی نمایش داده می شود .
- ✓ oblique : همان مقدار italic است و حروف متن را به صورت کج نمایش می دهد .

font-variant

این قاعده حروف کوچک متن را به حروف بزرگ تبدیل می کند اما این تبدیل ، با اندازه فونت متن ارتباط مستقیمی ندارد .مثال زیر را در نظر بگیرید :

```
<div style='font-size:40px ; font-variant:small-caps'>
     this is a test
</div>

<div style='font-size:40px'>
     THIS IS A TEST
</div>
```

تنظیم مقدار قاعده font-variant به small-caps موجب می شود تا حروف کوچک متن به حروف بزگ تبدیل شوند . در پاراگراف اول مثال فوق ، این مورد پیش بینی شده است . همان طور که ملاحظه میکنید مقدار قاعده font-size برای هر دو پاراگراف برابر با 40 پیکسل در نظر گرفته شده است اما در صورت اجرا خواهید دید که اندازه عبارت THIS IS A TEST در هر دو حالت متفاوت است .

پارامتر ها :

- ✓ normal : حالت معمول را برای متن د رنظر می گیرد و افکتی را بر روی آن اعمال نمی کند .
- ✓ small-caps : حروف کوچک متن را به مقیاسی بزرگتر از حروف کوچک تبدیل می کند . دقت داشته باشید که این مقدار ، بر روی حروف بزرگ متن تاثیر نمی گذارد .

font

این قاعده یک قاعده میانبر است و با استفاده از آن می توان تمام خواص فونت که در بالا ذکر شد را یک جا تعیین کرد .
پارامتر های font-size و font-family در این قاعده مقادیر اجباری هستند که حتما باید ذکر شوند .
سینتکس کلی استفاده از این قاعده به صورت زیر باید باشد :

```
font: { [font-style] [font-variant] [font-weight] font-size [ /line-height ]
font-family } ;
```

در مثال زیر قاعده font را با مقدار دهی به برخی پارامتر های آن به تگی با نام sidebar اعمال می شود :

```
#sidebar {
      font: bold small-caps 0.8em/1.4 Helvetica, Arial, "Luxi Sans", sans-
      serif;
}
```

letter-spacing

با استفاده از این قاعده می توان فاصله بین حروف متن را کاهش یا افزایش داد . این مقدار به میزان فاصله معمول کم یا به آن اضافه می شود. در مثال زیر فاصله بین حروف متن قرار گرفته در تمام تگ های div ، برابر با 1- پیکسل در نظرگرفته شده است که موجب می شود فاصله بین حروف از حد معمول کمتر شود :

```
div  {
      letter-spacing: -1px;
}
```

پارامتر ها :

یک عدد همراه با نوع واحد می تواند به عنوان پارامتر معرفی شود . مقادیر منفی مجاز هستند . مقادیر منفی موجب کاهش فاصله بین حروف و مقدار ثابت باعث افزایش فاصله بین حروف می شوند .
مقدار normal مقدار پیش فرض است و حد معمول فاصله بین حروف را در نظر می گیرد .

word-spacing

با استفاده از این قاعده می توان فاصله بین کلمات متن را کاهش یا افزایش داد . این مقدار به میزان فاصله معمول اضافه یا از آن کم می شود .

پارامتر ها :
همانند خاصیت Letter-spacing است .

line-height{ number | percentage | length }

با استفاده از این می توان میزان فاصله بین خطوط یک متن چند خطی را مشخص کرد . در مثال زیر از پارامتر number این قاعده استفاده شده و عددی که در مقدار font-size تگ ضرب می شود برابر 1.5 تعیین شده است :

```
html {
      line-height: 1.5;
}
```

پارامتر ها :

پارامتر length یک عدد همراه با نوع واحد آن است .

پارامتر number عددی ثابت یا اعشاری بدون تعیین نوع واحد آن است که در مقدار قاعده font-size تگ ضرب می شود تا مقدار نهایی برای میزان فاصله بین خطوط به دست آید .

پارامتر percentage نیز یک مقدار درصدی است که همانند پارامتر number در مقدار قاعده font-size تگ ضرب می شود .

مقادیر منفی برای این قاعده مجاز نیستند .

text-align

با استفاده از این قاعده می توان تراز محتویات یک تگ را مشخص نمود . پارامتر های این قاعده یکی از مقادیر left , right , center , justify است .

text-decoration

با استفاده از این قاعده می توان افکتی را به متن تگ اعمال کرد . رنگ افکت از قاعده color به ارث برده می شود .

پارامتر ها :

- ✓ blink : این مقدار موجب می شود تا متن حالت چشمک زن داشته باشد . مرورگر ها می توانند این مقدار را در نظر نگیرند .
- ✓ line-through : خطی بر روی متن ایجاد می کند .
- ✓ none : مقدار پیش فرض است و افکتی به متن نمی دهد .
- ✓ overline : خطی در بالای متن ایجاد می کند .
- ✓ underline : متن را زیر خط می کند .

text-indent

این قاعده میزان تورفتگی اولین خط یک متن را مشخص می کند . تورفتگی از سمت چپ یا راست بستگی به جهت صفحه وب دارد که این مورد معمولا با استفاده از خاصیت dir تگ html مشخص می شود .

نکته : تنظیم مقدار قاعده text-indent به یک مقدار عددی منفی بزرگ ، یکی از ترفند هایی است که می توان برای مخفی کردن اولین خط یک متن استفاده نمود . به عنوان مثال ، دستوری همانند text-indent : -9999px موجب می شود تا حتی بزرگترین مانیتور ها هم نتوانند اولین خط یک متن را نمایش دهند .

text-transform

با استفاده از این قاعده می توان بر روی بزرگ یا کوچک نمایش داده شدن حروف متن کنترل داشت .

پارامتر ها :

- ✓ capitalize : اولین حرف تمامی کلمات متن را به حروف بزرگ تبدیل می کند . بقیه حروف متن به همان شکلی که هستند نمایش داده می شوند .
- ✓ lowercase : تمامی حروف متن را به حروف کوچک تبدیل می کند .
- ✓ none : هیچ تغییری بر روی متن ایجاد نمی کند .
- ✓ uppercase : تمامی حروف متن را به حروف بزرگ تبدیل می کند .

text-shadow

با استفاده از این قاعده که در نسخه CSS 3 اراه شده است می توان چند افکت سایه ای مختلف به متن اعمال کرد . در مثال زیر افکتی سایه ای با رنگ سیاه که به اندازه 2 پیکسل درسمت راست و پایین متن امتداد و دارای 2 پیکسل ماتی است به تگی به نام title اعمال می شود :

```
#title{
      text-shadow : 2px 2px 2px #000;
}
```

پارامتر ها :

این قاعده دو مقدار اجباری که شامل جهت کشیده شدن سایه هستند و دو مقدار اختیاری که شامل میزان ماتی و رنگ سایه هستند را می پذیرد . جهت سایه با استفاده از دو مقدار عددی همراه با نوع واحد آن ها تعیین می شود . مقدار اول در صورتی که عددی مثبت باشد میزان کشیده شدن سایه را از سمت راست عنصر مشخص می کند و در صورتی که عددی منفی باشد میزان کشیده شدن از سمت چپ را تعیین می کند . مقدار دوم در صورتی که عددی مثبت باشد میزان کشیده شدن سایه را از پایین عنصر و در صورتی که عددی منفی باشد میزان کشیده شدن سایه را از بالای عنصر تعیین می کند .

میزان ماتی سایه بعد از تعیین جهت سایه مشخص می شود و مقداری عددی است که همراه با نوع واحد آن تعیین می شود. اگر میزان ماتی تعیین نشود ، سایه به صورت واضح نمایش داده می شود .

رنگ سایه می تواند قبل و یا بعد از مقادیر جهت سایه و میزان ماتی ذکر شود . اگر رنگی برای سایه تعیین نشود ، رنگ آن از قاعده color تگ به ارث برده می شود .

- ✓ **نکته : این قاعده فقط در مرورگر های Firefox , Safari , Opera , Chrome به خوبی پشتیانی می شود .**

vertival-align

با استفاده از این قاعده می توان تراز عمودی محتویات داخل تگ ها یا تراز عمودی محتویات خانه های جدول را مشخص می کند . در مثال زیر تمامی تصاویر درون خانه های جداول ، در پایین محوطه اشغال شده توسط آن خانه قرار خواهند گرفت :

```
td img {
      vertical-align: bottom;
}
```

کلمات کلیدی زیر می توانند به عنوان مقدار برای این پارامتر اعمال شوند :

- ✓ baseline : اصطلاح baseline به خطی فرضی گفته می شود که حروف بر روی آن نوشته می شوند . این خطوط را می توانید همانند خطوط یک دفتر یادداشت در نظر بگیرید . این مقدار موجب می شود تا تگ فرزند در زیر baseline تگ پدرش قرار گیرد .
- ✓ bottom : حاشیه پایین تگ فرزند در بالای پایینترین سطح هر عنصری در سطر جاری قرار خواهد گرفت .
- ✓ middle : تگ فرزند در وسط baseline تگ پدر قرار خواهد گرفت .
- ✓ sub : تگ فرزند را کمی پایین تر از امتداد متن تگ پدر قرار می دهد .
- ✓ super : تگ فرزند را کمی بالاتر از امتداد متن تگ پدر قرار می دهد .
- ✓ text-bottom : متون تگ پدر در امتداد زیرین تگ فرزند قرار خواهند گرفت .
- ✓ text-top : متون تگ پدر در امتداد بالای تگ فرزند قرار خواهند گرفت .
- ✓ top : حاشیه بالای تگ فرزند با حاشیه بالای بلندترین سطح هر عنصری در سطر جاری همتراز خواهد شد .

white-space

با استفاده از این قاعده می توان نحوه برخورد با کاراکتر های نامرئی برای متون چند خطی را کنترل کرد.

پارامتر ها :

- ✓ normal : این مقدار ، رفتار معمول HTML برای برخورد با کارکتر های نامرئی را در نظر میگیرد .
- ✓ nowrap : این مقدار نیز همانند بالا رفتار می کند اما خطوط را نمی شکند .
- ✓ pre : موجب می شود تمامی کاراکتر های نامرئی بدون تغییر در متن نمایش داده شوند . در حقیقت حالت اولیه متن حفظ می شود . خطوط ، تنها در حالتی شکسته می شوند که صراحتا با استفاده از تگ
 این مورد لحاظ شده باشد .
- ✓ pre-line : همانند normal عمل میکند .
- ✓ pre-wrap : این مقدار نیز همانند مقدار pre است . تنها تفاوت این دو مقدار در این است که در مقدار pre-wrap ، در صورتی که طول متن از پهنای تگ بیشتر شود ، آن متن شکسته می شود و به خط بعد می رود .

direction

با استفاده از این قاعده می توان جهت متون و محتوای یک تگ را مشخص نمود .

پارامتر ها :

- ✓ ltr : تراز محتویات تگ ، از سمت چپ به راست خواهد بود .
- ✓ rtl : تراز محتویات تگ ، از سمت راست به چپ خواهد بود .

چگونگی تغییر نشانگر ماوس در CSS ؟

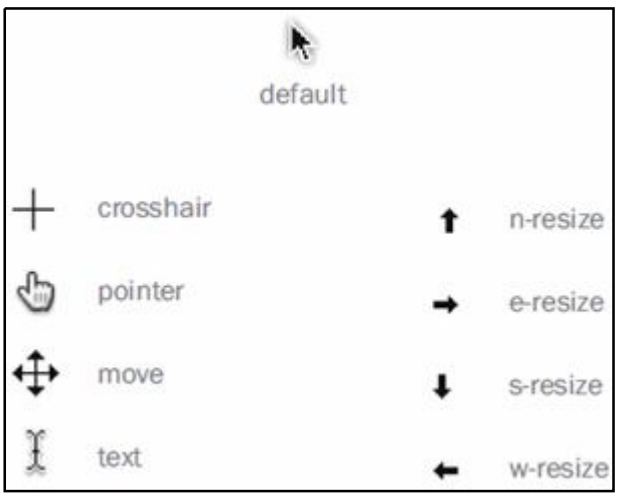
برای این کار می بایست از قاعده ای به نام cursor استفاده کنیم .

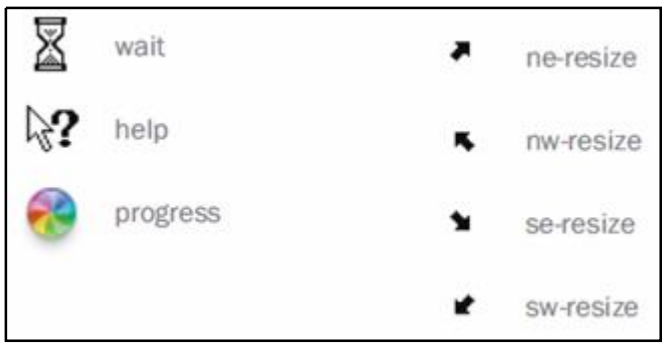
می توان لیستی از مقادیر را که با استفاده از کاراکتر "," از هم جدا شده اند برای این قاعده تعیین کرد . اگر مرورگر از اولین مقدار پشتیبانی نکند ، مقادیر بعدی را بررسی خواهد کرد . می توان یک عکس دلخواه را هم برای نشانگر ماوس تعریف کرد . برای اینکار باید آدرس عکس را حتما ذکر کنیم . این عکس دلخواه برای IE حتما باید پسوند cur. داشته باشد . حداکثر ابعادی که این تصویر می تواند داشته باشد در IE باید 32*32 باشد .

در مثل زیر ، شکل اشاره گر ماوس د رحالتی که بر روی لینک هایی که قبلا بازدید نشده اند قرار می گیرد از یک مسیر خوانده می شود . در صورتی که مرورگر نتواند به این مسیر دسترسی پیدا کند ، مقدار دوم را که pointer است در نظر می گیرد :

```
a:link{
     cursor:url(hyper.cur),pointer;
}
```

مقادیری که این قاعده می تواند بپذیرد به همراه تصویر هر یک از آن ها در شکل زیر نشان داده شده است :





قواعد مربوط به جداول :

قواعد این دسته اجازه تعیین نحوه نمایش جدول و اجزای آن را در صفحه می دهند .

table-layout: { auto | fixed | inherit }

این قاعده نحوه محاسبه پهنای تگ table و اجزای آنها را کنترل می کند . توجه داشته باشید پهنای تگ table بر اساس محتویات درون آن محاسبه می شود .

دو الگوریتم مختلف برای تعیین پهنای ستون های جدول وجود دارد . الگوریتم fixed و auto . اگر مقدار قاعده width تگ table برابر با auto باشد به طور معمول از الگوریتم auto استفاده می شود . مرورگر ها می توانند برای عناصری که مقدار قاعده display آنها برابر با table است از الگوریتم fixed استفاده کنند .

هنگامی که از الگوریتم fixed استفاده می شود پهنای ستون های جدول و خود آن از محتویات درون خانه های جدول پیروی نمی کنند و پهنای هر ستون به شکل زیر محاسبه می شوند :

- ستون هایی که مقدار قاعده width آنها برابر با auto نیست ، پهنای ستون خود را تعیین می کنند .
- خانه ای که در ردیف اول جدول قرار دارد و مقدار قاعده width آن auto نیست ، پهنای ستونی که در آن قرار دارد را تعیین می کند . اگر آن خانه خود نیز از چند ستون تشکیل شده بود ، مقدار پهنا بین این ستون ها تقسیم می شود .
- مقدار پهنای باقیمانده به طور مساوی بین ستون های باقیمانده تقسیم می شود . البته پهنای حاشیه ها و فضای بین خانه ها در محاسبه پهنا در نظر گرفته نمی شود .

پارامتر ها :

- ✓ auto : الگوریتم auto را اعمال می کند .
- ✓ fixed : الگوریتم fixed را اعمال می کند .

border-collapse: { collapse | separate | inherit }

این قاعده مدل حاشیه را برای تگ table مشخص می کند .

دو نوع مدل نمایش وجود دارد :

- **Separate** : در این مدل فقط خانه های درون جدول و خود جدول می توانند حاشیه داسته باشند . ردیف ها ، گروهی از ردیف ها ، ستون ها و گروهی از ستون ها نمی توانند دارای حاشیه باشند . در این مدل حاشیه ها به دور خانه ها کشیده می شوند و بین خانه ها ، فضای افقی و عمودی که مقدار آن به وسیله قاعده border-spacing تعیین می شود قرار می گیرد .
- **Collapsing** : در این مدل فاصله ای بین خانه های جدول و حاشیه های آنها قرار نمی گیرد .

✓ **نکته : مقدار پیش فرض این قاعده برابر separate است .**

www.SoftGozar.com

border-spacing

این قاعده فقط در هنگامی که از مد separate برای جدول استفاده می شود کاربرد دارد و میزان فاصله بین خانه های جدول را کنترل می کند . در مثال زیر مقدار 1em برای فاصله افقی و 0.5em برای فاصله عمودی بین خانه های جدولی با نام results تعیین شده است :

```
#results {
      border-collapse: separate;
      border-spacing: 1em 0.5em;
}
```

پارامتر ها :

مقادیر منفی نیز مجاز هستند .

این قاعده می تواند یک یا دو مقدار را بپذیرد . اگر دو مقدار برای این قاعده تعیین شود ، مقدار اول به عنوان فاصله افقی و مقدار دوم به عنوان فاصله عمودی بین خانه های جدول درنظر گرفته می شوند . اگر فقط یک مقدار تعیین شود این مقدار هم برای فاصله افقی و هم برای فاصله عمودی در نظر گرفته می شود .

empty-cells

با استفاده از این قاعده می توان مشخص نمود که آیا رنگ پس زمینه و حاشیه های خانه ای از جدول که محتویاتی ندارد نمایش داده شوند یا خیر . این قاعده فقط در هنگامی که از مدل separate برای جدول استفاده می شود کاربرد دارد و در مدل collapsing در نظرگرفته نمی شود .

پارامتر ها :

- show : حاشیه ها و رنگ پس زمینه خانه های خالی را نمایش می دهد .
- hide : حاشیه ها و رنگ پس زمینه خانه های خالی را نمایش نمی دهد . اگر مقدار قاعده empty-cells تمامی خانه های موجود در یک ردیف جدول برابر با hide باشد و همچنین تمامی این خانه ها نیز خالی باشند آن ردیف نمایش داده نمی شود .

caption-side: { bottom | top | inherit }

این قاعده مکان عنوان جدول را در جهت عمودی مشخص می کند .

برای تعیین مکان افقی عنوان جدول می توان از خاصیت text-align استفاده نمود .

این خاصیت دو مقدار bottom و top را که به ترتیب برای نمایش عنوان جدول در پایین و بالا استفاده می شوند را می پذیرد .

www.SoftGozar.com

وراثت در CSS به چه معناست ؟

وراثت فرآیندی است که در آن خصوصیات تگ پدر در صورتی که برای تگ فرزندش آن خصوصیت تعریف نشده باشد به تگ فرزند به ارث می رسند . به عنوان مثال اگر قاعده "font-size:20px" برای یک تگ div تعریف شده باشد و برای تگ های فرزند آن تعریف نشده باشد تمامی فرزندان این قاعده را از پدرشان یعنی همان div به ارث می برند .

البته به این نکته توجه داشته باشید که تنها تعدادی از قواعد تگ پدر توسط تگ فرزند به ارث برده می شوند . به عنوان مثال رنگ متن بین تگ ها به طور خودکار از تگ پدر به تگ فرزند به ارث می رسند . دلیل این امر واضح است : دراکثر مواقع نیاز دارید تا رنگ متن توسط تگ های فرزند به ارث برده شوند اما اگر تگ پدر تصویری را به عنوان عکس پس زمینه داشته باشد تمایل ندارید تا تگ های فرزند نیز همان تصویر را به عنوان تصویر پس زمینه شان داشته باشند .

گاهی اوقات ممکن است نیاز داشته باشید که برخی قواعد حتما توسط تگ های فرزند به ارث برده شوند . در این حالت باید از کلمه کلیدی inherit استفاده کنید .

گروه بندی عناصر صفحه با استفاده از تگ های div و span :

ما می توانیم از تگ های و برای گروه بندی سایر تگ ها به منظور ایجاد بخش ها و زیر بخش های داخل یک صفحه HTML استفاده کنیم . از این دو عنصر معمولا به همراه CSS برای اینکه به گروهی از عناصر style خاصی نسبت دهیم استفاده می شود .

برای مثال فرض کنید می خواهید تعدادی عنصر در بخشی از صفحه همیشه از کلاس خاصی پیروی کنند. احتمالا کلاس مورد نظر را به روش زیر به تک تک آن ها نسبت می دهید :

```
<h3 class='footer'>This is a header</h3>
<p class='footer'>This is a paragraph.</p>
<h2 class='footer'>This is a paragraph.</h2>
<strong class='footer'>this text is strong</strong>
```

اما ما می توانیم به جای این کار آن ها را با استفاده از تگ div داخل یک گروه قرار داده و کلاس مورد نظر را فقط به تگ div اعمال کنیم . به روش زیر :

```
<div class='footer'>
     <h3>This is a header</h3>
     <p>This is a paragraph.</p>
     <h2>This is a paragraph.</h2>
     <strong>this text is strong</strong>
</div>
```

یکی از ویژگی های مهم تگ div این است که از آن برای گروه بندی عناصر block-level و inline می توان استفاده نمود .

از طرف دیگر ما می توانیم از تگ span فقط برای گروه بندی عناصر از نوع inline استفاده کنیم . بنابراین اگر می خواهید حداقل به یک جمله و یا قسمتی از یک پاراگراف style خاصی نسبت دهید بهتر است از span استفاده کنید . به عنوان مثال :

```
<p>My mother has <span class="blue">light blue</span> eyes.</p>
```

یکی دیگر از تفاوت های تگ های span و div در این است که تگ div خود جزء عناصر block-level و span جزء عناصر inline-level است .

www.SoftGozar.com

آشنایی با نحوه طرح بندی و طراحی قالب سایت

در این بخش قصد داریم به بررسی روش های طرح بندی (Layout) که این روز ها از آن با عنوان **طراحی قالب** یاد می شود آشنا شویم .

راه های گوناگونی برای این امر وجود دارد که استفاده از عنصر table یکی از آن هاست . استفاده از جداول برای طرح بندی کلی صفحات به دلیل مشکلاتی همچون سخت بودن کار با آن ها (مخصوصا هنگام کار با جداول تو در تو) ، سرعت بارگذاری پایین و عدم کنترل دقیق چگونگی نمایش ، دیگر این روز ها پیشنهاد نمی شود و به جای آن استفاده از ترکیب عناصر ساده تری همچون div به همراه قواعد CSS برای کنترل خصوصیات آن ها توصیه می شود .

در این قسمت سعی می کنیم به اختصار به نحوه طراحی قالب های دو ستونه و سه ستونه پرداخته شود. بدیهی است مباحث به تفصیل در کلاس آموزشی مورد بررسی قرار خواهند گرفت .

طرح بندی صفحات float-based :

راه های گوناگونی برای طرح بندی صفحات از طریق CSS وجود دارد که از آن می توان به استفاده از روش تعیین مکان مطلق عناصر با کمک قواعد position و margin اشاره نمود . اما ما از روش دیگری که بسیار ساده تر و البته پر کاربردتر است استفاده خواهیم کرد و آن استفاده از قواعدی همچون float و clear برای تعیین موقعیت عناصر است . در این روش ما به سادگی عرض و ارتفاع عناصر را تعیین کرده و سپس مشخص می کنیم که در چپ یا راست عناصر در برگیرنده خود قرار بگیرند . (در واقع به مشخص می کنیم که کدام جهت شناور شوند .)

ایجاد طرح (قالب) های شناور دو ستونه :

برای ایجاد یک طرح دو ستونه ساده می بایست کار را از یک صفحه HTML شروع کنیم .در صفحه HTML مثالی که در ادامه بررسی خواهیم کرد بخش های اصلی زیر وجود دارد .

- content : بخش محتوا که قرار است محتوای اصلی سایت در آن قرار بگیرد .
- mainNav : بخش پیمایش اصلی (main Navigation) که قرار است در سمت چپ قرار گرفته و حاوی لینک ها و عناصری برای دسترسی ساده تر کاربر به صفحات دیگر سایت باشد .
- footer : بخش پانویس که قرار است اطلاعاتی اضافی از سایت را در آن قرار دهیم .

هر کدام از این عناصر را با یک عنصر div ایجاد خواهیم کرد و از طریق selector نام (id) قواعد CSS مربوطه را به آن ها اعمال می کنیم .

تمامی این بخش ها در داخل یک عنصر div با نام wrapper پوشانده شده اند . به کد زیر دقت کنید :

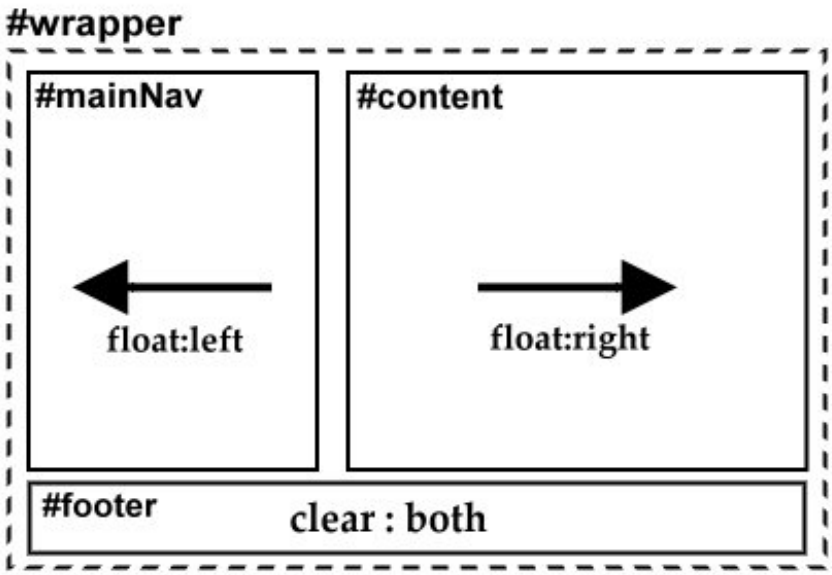
```
<div id="wrapper">

    <div id="content">
    ...
    </div>

    <div id="mainNav">
    ...
    </div>

    <div id="footer">
    ...
    </div>

</div>
```

www.SoftGozar.com

در این مثال قرار است content به سمت راست و mainNav به سمت چپ بروند . به این نکته دقت کنید چون بخش content ما حاوی اطلاعات مهمتری است کد آن نیز باید قبل از بخش mainNav بیاید تا دسترسی و استفاده از آن نیز آسان تر باشد .
حال به سراغ تعیین قواعد عناصر فوق در CSS می رویم . ما از selector نام برای تعریف قواعد برای هر بخش استفاده می کنیم .
ابتدا به صورت زیر برای هر یک از بخش های content و mainNav عرض تعیین می کنیم :

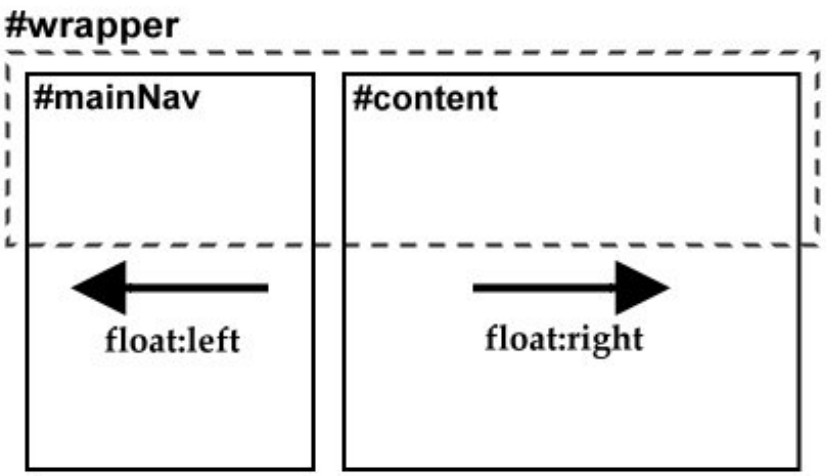
```
#content {
      width: 520px;
      float: right;
}
#mainNav {
      width: 180px;
      float: left;
}
```

همانطور که می بینید چون content باید در سمت راست قرار بگیرد قاعده float آن را برابر right و برای mainNav آن را برابر left قرار می دهیم . به تصویر زیر نگاه کنید:



✓ **نکته بسیار مهم :**

از آنجایی که عناصری از نوع شناور (یعنی عناصری که قاعده float آنها برابر right یا left باشد) معمولا از جریان عادی صفحه خارج شده و بر روی عناصر دیگر نمی توانند تاثیر بگذارند احتمالا به صورت کامل در عنصر در برگیرنده خود (در مثال ما منظور از عنصر در برگیرنده wrapper است) قرار نگرفته و آن چیزی که انتظار داریم انجام نمی شود . در واقع طرح ما به صورت زیر در خواهد آمد :



www.SoftGozar.com

برای حل این مشکل دو راه حل وجود دارد :

1. یا اینکه عنصر دربرگیرنده (wrapper) خود حالت float داشته باشد . (یعنی به چپ یا راست متمایل شود)- این مورد را امتحان کنید !
2. یا اینکه عنصری با قاعده clear : both را در آن قرار دهیم .

ما از روش دوم استفاده خواهیم کرد . برای این کار از بخش footer استفاده می کنیم و قاعده clear آن را برابر both قرار می دهیم . با این کار مطمئن می شویم که footer دقیقا در پایین دو بخش قبلی قرار خواهد کرد و چون خود ، حالت float ندارد در نتیجه به عنصر wrapper ارتفاع خواهد داد و به این شکل مشکل قبلی اصلاح خواهد شد :

```
#footer {
      clear: both;
}
```

ایجاد طرح های سه ستونه شناور :

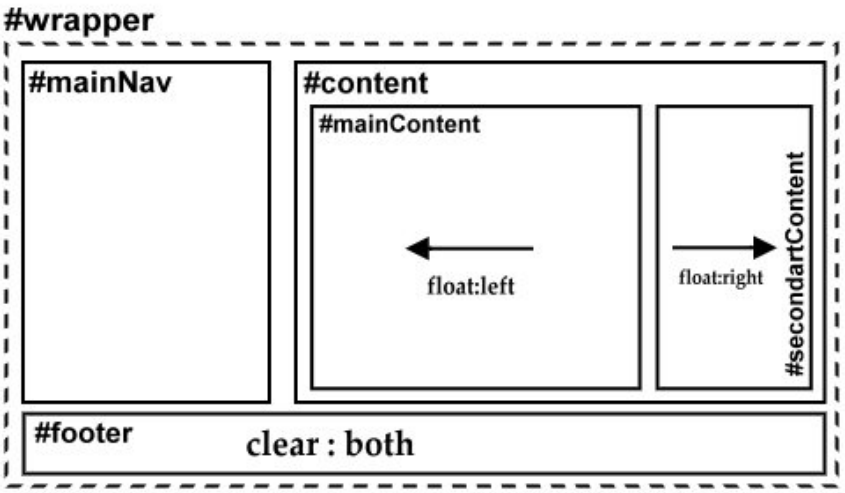
ساختار صفحه HTML مورد استفاده برای ایجاد طرح های سه ستونه بسیار شبیه به ساختار طرح های دو ستونه است . تنها تفاوت ، اضافه شدن دو عنصر div جدید در بخش content است : یکی با نام mainContent برای محتوای اصلی و دیگری با نام secondaryContent برای محتوای ستون سوم که قرار است در سمت راست نمایش داده شود :

```
<div id="content">
      <div id="mainContent">
      …
      </div>

      <div id="secondaryContent">
      …
      </div>
</div>
```

دقیقا با همان روشی که قبلا اشاره شد می توان عنصر mainContent را به چپ و عنصر secondaryContent را به سمت راست عنصر content هدایت کرد :

```
#mainContent {
      width: 320px;
      float: left;
}
#secondaryContent {
      width: 180px;
      float: right;
}
```

در واقع با این کار ما عنصر content را به دو قسمت تقسیم کردیم که با عنصر mainNav جمعا تشکیل سه ستون می دهند . تصویر زیر گویای موضوع خواهد بود :



www.SoftGozar.com