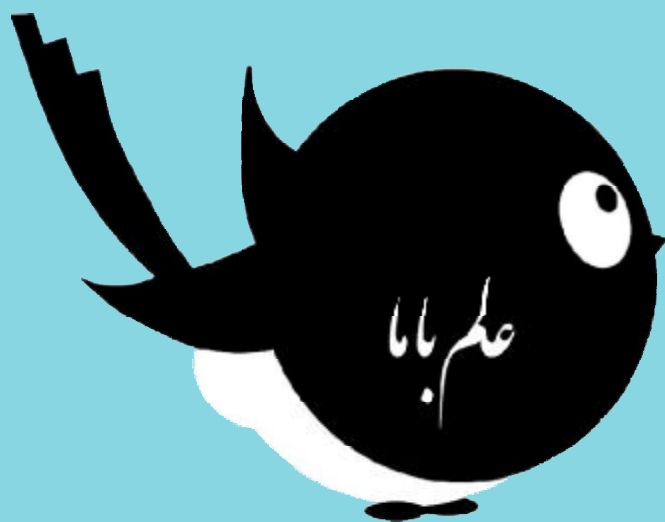


آموزش سریع و آسان CSS در یک روز



www.elmbama.i



فهرست

۳	مقدمه
۴	ساعت اول
۶	آشنایی با واژه های معمول html
۸	ساختار کلی یک صفحه html
۱۰	فرمت کلی دستورات CSS
۱۱	انتخابگرها در CSS
۱۶	روش های درج CSS
۱۸	ساعت دوم
۱۹	معرفی Divisions و Spans
۲۰	اسکلت صفحه وب
۲۷	پس زمینه
۳۱	ساعت سوم
۳۲	FONTS
۳۹	LINKS
۴۳	LISTS
۵۰	ساعت چهارم
۵۱	Tables
۵۸	Display
۶۲	Width,Max-width
۶۴	ساعت پنجم
۶۴	Position
۶۹	OverFlow



مقدمه

سلام به تو دوست خوبم و خیلی خوشحالم و **بهت تبریک می گم** که این کتاب رو برای یادگیری **html&css** انتخاب کردی و من بهت قول می دم که نهایت سعیم رو بکنم تا تو بتونی در **یک روز CSS** رو کامل یاد بگیری و به راحتی سایت های زیبایی رو طراحی کنی. توی این کتاب من به صورت کامل **html&css** رو توضیح دادم که بتونی به راحتی درکی از **html&css** بدست بیاری و او نو به صورت رایگان در اختیار قرار دادم. اما اگه خواستی که بازم به صورت راحت تر مطلب رو یادگیری و در یک محیط واقعی همراه بامن پیش بری ، تصمیم گرفتم که به **صورت تصویری** این آموزش رو هم برات آماده کنم که می تونی بزودی از سایت [علم با ما](http://www.ElmBama.ir) دانلود کنی .



ساعت اول



در اولین ساعت یه سری کلیات هستش که باید با هم کار کنیم، که به صورت کامل من در این قسمت بهش می پردازم و توضیح می دم

اجازه بده با یک مثال بحث رو شروع کنیم.



فرض کن وارد یک فروشگاه ماشین شدی و می خواهی یک ماشین بخری اولین چیزی که توجهت رو جلب میکنه حتما ظاهر ماشیناس، بعد از اینکه ظاهر یک ماشین رو پسندیدی حالا مشخصات فنی اون رو بررسی می کنی. سایت هم شباهت زیادی به یک ماشین داره که در اینجا **html** میشه شاسی ماشین که ساختمان اصلی ماشین روی شاسی سوار میشه و **CSS** ظاهر ماشینه که شامل رنگ و اتاق و داشبورد میشه و موتور ماشین هم میشه یکی از زبان های برنامه نویسی دیگه مثل **php** یا **C#** یا **jquery** و ...

html که توی این مثال موتور ماشین بودش یک زبان برنامه نویسی هست که برای اولین بار بشر تونست محتوای وب رو به صورت خیلی ساده طراحی کنه و متن و تصویر رو داخل شبکه برای بازدید عموم قرار بده ، اما این ظاهر زیاد دلچسب و زیبا نبود و واسه همین یه مقدار افکت و رنگ داخل این زبان استفاده شد ، اما سرعت بارگذاری سایت رو خیلی زیاد کرد ، برای رفع این مشکل **CSS** اختراع شد.



زبان **html** از یک سری المنت تشکیل شده و با استفاده از این المنت ها می تونیم بدنه سایت رو طراحی کنیم. به عنوان مثال می تونیم **header** سایت رو ایجاد کنیم ، می تونیم صفحه وب رو قسمت بندی کنیم و یا پاراگراف ایجاد کنیم و یا تصویری رو به صفحه اضافه کنیم.

حالا با استفاده از **CSS** می تونیم بگیم که این المنت ها چجوری تو صفحه ، نمایش داده بشن. یعنی با چه رنگی یا با چه وضعیتی در صفحه نمایش داده بشن.

یکی از قابلیت های مهم **CSS** این هستش که می تونیم با یک بار کدنویسی ، هر تعداد صفحه **html** که داشتیم رو قالب دهی کنیم و نیازی نیست که برای هر صفحه، **CSS** ایجاد کنیم. پس **CSS** باعث میشه که حجم زیادی از کارها کاهش پیدا کنه و این برنامه نویسی رو خیلی هیجان انگیز و ساده می کنه.

آشنایی با واژه های معمول html

برای شروع حتما باید با سه اصطلاح رایج و پر کاربرد **html** آشنا بشی که به مراتب توی آموزش ازشون استفاده میشه چونکه کلا ما وابسته به این اصطلاحات هستیم و با درک اینها هستش که می تونیم به راحتی کد بنویسیم. البته اگه الان زیاد این مفاهیم قدرتمند رو درک نکردین اشکال نداره چون در آینده بدلیل اینکه ما زیاد این اصطلاحات رو تکرار می کنیم شما به راحتی متوجه میشین:

المنت ها (Elements) :

المنت ها از مهمترین عناصر **html** هستن که به صورت کلی طراحان و تعریف کنندگان یک صفحه **html** هستند. از مهمترین اونها میشه **<h1>** تا **<h6>** که برای نمایش عنوان سایت در سایزهای مختلف استفاده میشه اشاره کرد و پاراگراف **<p>** که برای تعریف یک پاراگراف ساده استفاده می شه و همچنین **<a>** و **<div>** و **** و **** و ...

المنت ها بین براکت های بزرگتر و کوچکتر قرار می گیرند به شکل زیر

<a>



تگ ها (Tags):

با قرار گرفتن المنت میان براکت های بزرگتر و کوچکتر تگ شکل می گیرد . تگ ها نقطه شروع و پایان دارند یعنی ما یک تگ شروع و تگ پایان داریم. تگ شروع شامل المنتی است که میان براکتهای بزرگتر و کوچکتر قرار گرفته و تگ پایان المنتی است که میان براکت های بزرگتر و کوچکتر به همراه یک اسلش قرار گرفته است. به عنوان مثال تگ **div** که به شکل زیر نمایش داده استفاده می‌شود.

`<div>هرچیزی می‌تونیم این وسط استفاده کنیم</div>`

خواص (Attributes):

خواص یا ویژگی‌های خاصی هستند که برای ارائه اطلاعات بیشتر در خصوص المنت استفاده می‌شوند. یکی از خصوصیت های معمول **id** هستش که جهت شناسایی المنت استفاده می‌شود ، **class** یکی دیگه از خصوصیت ها هستش که برای کلاس بندی المنت های استفاده می‌شود و یا **src** که خصوصیتی هستش که برای درج محتوای خارجی به یک المنت استفاده می‌شود. مثل عکس و ...

خواص معمولاً بعد از نام المنت آورده می‌شوند و شامل نام خواص و مقدار هستند که در مثال زیر به وضوح این مورد نشون داده شده

``

در این مثال همونطور که می بینید یک المنت لینک هستش که به خصوصیت **href** داره و مقدار این خصوصیت یک آدرس وب هستش.

www.elmbama.ir

خروجی این دستور به صورت زیر هستش:

شکل کلی دستورات **html** به صورت زیر هستش





خب تا حالا به شکل کلی دستورات **html** آشنا شدیم و حالا قصد داریم که شکل کلی یک صفحه **html** رو بررسی کنیم . اگه تا اینجا جایی رو متوجه نشدین اصلا نگران نباش ، چون در ادامه خیلی بهتر با این موارد آشنا میشی . البته یادت باشه که تمرین برای یادگیری این مطالب خیلی خیلی مهمه .

ساختار کلی یک صفحه html

یک صفحه **html** داخل یک فایل متنی با پسوند **html** ذخیره میشه . برای اینکه شما بتونین یک صفحه **html** رو کد نویسی کنین فقط نیاز به یک ویرایشگر متن دارین . حالا این ویرایشگر متن می تونه **notepad** باشه و یا نرم افزارهای حرفه ای تر مثل **Dreamweaver** یا **Sublime** یا **Notepad++** و یا هر نرم افزار دیگه ای که بشه یک متن رو ویرایش کرد البته بجز **word** .

همه صفحه های **html** نیاز به ساختاری دارن که المنت های زیر رو شامل میشه:

```
<DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<body>
```

<!DOCTYPE html>: معرف نوع **html** هستش . در این خط ورژن **html** مشخص میشه . چون ما آخرین نسخه **html** رو استفاده می کنیم این تگ به تنهایی استفاده میشه و نیازی به سایر کدهای این خط نیست . این تعریف همیشه در اولین خط قرار داده میشه .

در ادامه وبعد از کد تعریف **html** المنت **<html>** قرار میگیره که نشون دهنده شروع کدهای **html** هستش و همه کد ها داخل این تگ قرار میگیره . یعنی به صورت روبرو

```
<html>سایر کد های مربوطه</html>
```

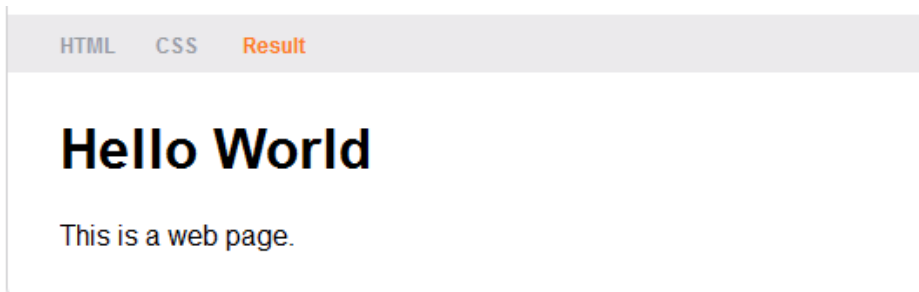
داخل تگ **html** تگ **head** قرار میگیره که قسمت بالای صفحه رو تعریف میکنه و شامل متادیتاها (اطلاعات همراه در مورد صفحه) هستش . اطلاعاتی که داخل تگ **head** قرار می گیره داخل صفحه نمایش داده نمیشه ، البته ممکنه شامل تگ **title** باشه که عنوان صفحه رو در مرور گر نمایش میده و سایر متادیتاهایی که فایلهایی رو از خارج به صفحه لینک می کنن .



تمامی محتوای قابل رویت در صفحه داخل تگ **body** قرار می گیرند. یک شکل ساده از صفحه **html** تو مثال زیر رو نشون دادم که بهتر با داستان آشنا بشین.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Hello World</title>
  </head>
  <body>
    <h1>Hello World</h1>
    <p>This is a web page.</p>
  </body>
</html>
```

خروجی دستورات بالا به شکل زیر هستش:



تو این مثال اولین خط که نشون میده ورژن **html** ما آخرین نسخه هستش و بعد تگ **html** شروع میشه که نشون می ده صفحه ما یک **html** هست و بعد تگ **head** اومده. داخل تگ **head** دو تا تگ آورده شده که تگ اول یعنی تگ **meta** مشخص کرده که کدینگ صفحه ما چجوری باشه و برای پشتیبانی از حروف فارسی فرمت **utf-8** استفاده میشه. البته تو تگ **html** یک خصوصیت **lang** هم اضافه شده که زبان صفحه رو نشون میده و برای فارسی کافیه که قرار بدین. بعد تگ **title** اومده که عنوان صفحه رو مشخص کرده و این عنوان توی صفحه نمایش داده نمیشه بلکه در قسمت نوار عنوان مرورگر نشون داده میشه. و داخل تگ **body** یک تگ **h1** اومده که نشون میده تیتل صفحه چی باشه و اونو با سایز درشت نمایش میده و بعد یک تگ پاراگراف اومده که **p** هستش.

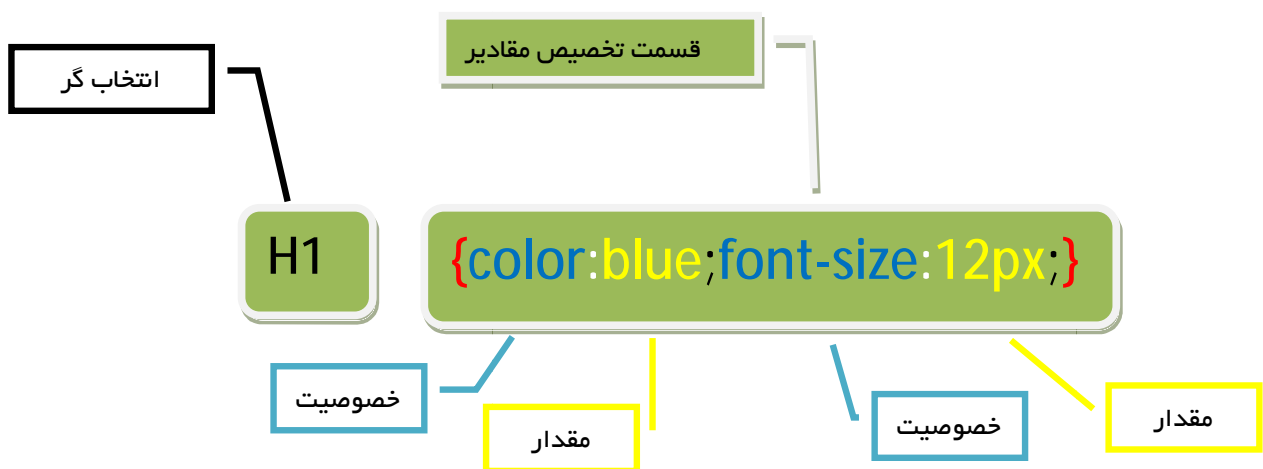


توی این مثال همونطور که میبینی همه تگ ها یک تگ بستن هم دارن مثلا تگ `<body></body>` که با تگ `<body>` شروع شده و با تگ `</body>` خاتمه پیدا کرده. اما بعضی از تگ ها هستن که تگ بستن ندارن و به تنهایی استفاده میشن. مثل تگ هایی که در قسمت زیر براتون لیست کردم:

- `
`
- ``
- `<meta>`
- `<whf>`
- `<embed>`
- `<input>`
- `<par am>`
- `<hr>`
- `<link>`
- `<source>`

فرمت کلی دستورات CSS

در فرمت دستور CSS زیر همانطور که می بینی برای تگ `h1` رنگ آبی و سایز فونت اون رو برابر با 12 پیکسل تعریف کردم.



انتخابگر: انتخابگر به یکی از المنتهای `html` اشاره می کنه که می خواهیم برای اون استایل تعریف کنیم.



قسمت تخصیص مقادیر: این قسمت شامل یک یا چندتا از دستورات CSS هستش که با یک سمیکالن از هم جدا میشن و داخل آکولاد قرار می گیرن و به المنتی که در قسمت انتخابگر قرار گرفته تخصیص داده میشن.

خصوصیت: دستورات CSS هستن که با یک دونقطه مقدار می گیرن

مثال

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

در این مثال به تگ p رنگ خاکستری و متن رو وسط چین قرار دادیم.

```
<html>  
<head>  
<style>  
p {  
  color: gray;  
  text-align: center;  
}  
</style>  
</head>  
<body>  
<p>سلام به شما</p>  
<p>من یک تگ پاراگراف هستم</p>  
</body>  
</html>
```

خروجی

سلام به شما
من یک تگ پاراگراف هستم

انتخابگرها در CSS

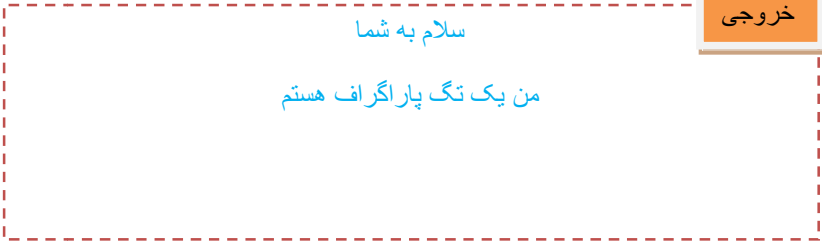
انتخابگرها در CSS برای یافتن و یا انتخاب یک المنت html استفاده میشه که می تونه بر مبنای نام المنت و یا id و یا کلاس مربوطه باشه. یعنی وقتی که می خواهیم برای یک تگ CSS بنویسیم و به طور مثال می خواهیم رنگ یک تگ رو تغییر بدیم باید این تگ رو انتخاب کنیم و به سیستم بفهمونیم که هدف و منظور ما این تگ هستش که به شرح زیر این حالت ها رو توضیح می دم:

الف - بر اساس نام المنت :



در این روش با استفاده از نام المنت می شه یک المنت رو انتخاب کرد به عنوان مثال برای انتخاب کلیه تگ های **p** در **html** کافیه که به روش زیر این کار رو انجام بدیم.

```
<html>
<head>
<style>
p {
  color: blue;
  text-align: center;
}
</style>
</head>
<body>
<p>سلام به شما</p>
<p>من یک تگ پاراگراف هستم</p>
</body>
</html>
```



ب- بر اساس ID

در این روش کافیه که به یک المنت **id** تخصیص بدیم و بر اساس این **id** میشه داخل **CSS** اون رو انتخاب کنیم. برای تخصیص **id** چند مورد رو باید حتما مد نظر داشته باشیم.

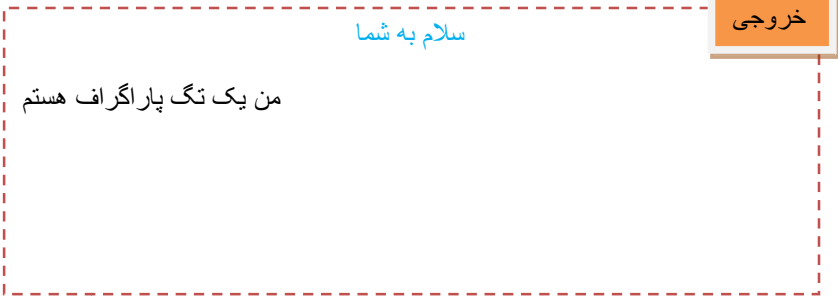
- یکی اینکه **id** حتما باید منحصر به فرد باشه و نمی شه که اونو به چند تا المنت تخصیص بدیم
- برای اینکه بتونیم بر اساس **id** یک المنت رو انتخاب کنیم باید حتما قبل از نام **id** علامت **#** رو قرار بدیم تا **CSS** متوجه بشه که هدف ما انتخاب بر اساس **id** هستش.

```
#name {
  color: blue ;
  text-align: center ;
}
```

مثال

در این مثال تگ **p** بر اساس **id** انتخاب شده و رنگ آبی و متن رو وسط جین قرار دادم.

```
<html>
<head>
<style>
#name {
  color: blue;
  text-align: center;
}
</style>
</head>
<body>
<p id="name">سلام به شما</p>
<p>من یک تگ پاراگراف هستم</p>
</body>
</html>
```





ج- بر اساس کلاس

در این روش میتونیم المنت ها رو بر اساس خاصیت کلاس انتخاب کنیم و مزیت این روش اینه که شما کافیه که یک مرتبه کد CSS رو بنویسید و چندین مرتبه و برای المنت های متفاوت ازون استفاده کنید.

برای استفاده از این روش کافیه که داخل CSS قبل از نام کلاس یک علامت (.) قرار بدین. برای درک بهتر میتونی به مثال زیر توجه کنی.

```
.myclass {  
  color: blue ;  
  text-align: center ;  
}
```

مثال

در این مثال تگ p بر اساس id انتخاب شده و رنگ آبی و متن رو وسط چین قرار دادم.

```
<html>  
<head>  
<style>  
.myclass {  
  color: blue;  
  text-align: center;  
}  
</style>  
</head>  
<body>  
<p>سلام به شما</p>  
<p class="myclass">من یک تگ پاراگراف هستم</p>  
</body>  
</html>
```

سلام به شما

من یک تگ پاراگراف هستم

خروجی

کلاس رو میشه به المنت های متفاوتی نسبت داد.

فرض کن که دوتا المنت داری با کلاس یکسان و میخوای فقط به یک المنت کلاس اختصاص پیدا کنه در این صورت میتونی در ابتدای نام کلاس نام المنت رو بیاری . مثل مثال زیر

```
<html>  
<head>  
<style>  
H1.myclass {  
  color: blue;  
  text-align: center;  
}  
</style>  
</head>  
<body>  
<p class="myclass">سلام به شما</p>  
<h1 class="myclass">من یک تگ هدر هستم</h1>  
</body>  
</html>
```

سلام به شما

من یک تگ هدر هستم

خروجی



همچنین همیشه به یک المنت چند تا کلاس رو اختصاص بدی مثل مثال زیر

مثال

```
.myclass {  
  color: blue ;  
}  
.yourclass{  
  text-align: center ;  
}
```

در این مثال تگ h1 براساس دو کلاس انتخاب شده، توجه کنین که بین کلاس ها فقط باید فاصله باشه

```
<html>  
<head>  
<style>  
  . myclass {  
    color: blue ;  
  }  
  .yourclass{  
    text-align: center ;  
  }  
</style>  
</head>  
<body>  
<p>سلام به شما</p>  
<p class="myclass yourclass">من یک تگ پاراگراف هستم</p>  
</body>  
</html>
```

سلام به شما

من یک تگ پاراگراف هستم

خروجی

د- انتخاب گروهی

اگه شما چند تا تگ دارین که استایل یکسانی قراره روی اونا اعمال بشه کافیه که همه اونا رو باهم انتخاب کنی. به

روش زیر:

```
<html>  
<head>  
<style>  
h1, h2, p {  
  text-align: center;  
  color: red;  
}  
</style>  
</head>  
<body>  
  
<h1>سلام به شما</h1>  
<h2>من یک هدر هستم</h2>  
<p>من یک پاراگراف هستم</p>  
  
</body>  
</html>
```

سلام به شما

من یک هدر هستم

من یک تگ پاراگراف هستم

خروجی



توضیحات

اگر بخواهیم به توضیحاتی رو تو کدهاتون قرار بدین باید چکاری انجام بدین؟ اصلا توضیحات به چه دردی می خوره توی CSS .

توضیحات برای اینه که داخل کدها به سری توضیحاتی رو در رابطه با کدهایی که گذاشتین قرار بدین تا هر موقع خواستین ویرایشی رو روی کدهاتون انجام بدین و یا اینکه کس دیگه ای خواسته باشه روی اونا کاری انجام بده به راحتی با توضیحاتی که شما نوشتین بتونه این کار رو انجام بده. برای قرار دادن توضیحات توی یک CSS کافیه که ابتدای توضیحات رو با علامت `/*` و آخر توضیحات رو با علامت `*/` خاتمه بدین. به همین راحتی!

خروجی

```
p {  
  color: red;  
  /* رنگ این تگ قرمز و وسط چین میشه */  
  text-align: center;  
}  
  
/* این استایل باعث میشه که همه پاراگراف  
های شما به صورت قرمز رنگ و وسط چین باشه */
```

سلام به شما

حتما کدهایی که به صورت مثال برات می ذارم رو توی یکه ویرایشگر مثل notepad و یا Dreamweaver تمرین کن و خروجی بگیر. تا مطلب رو کاملا متوجه بشی و خوب یادگیری و بتونی به راحتی سایت هایی با گرافیک عالی طراحی کنی. دوست من توجه داشته باش که هیچ موفقیتی با تمرین و تلاش بدست نمی یاد. خود من هم با تمرین زیاد تونستم یاد بگیرم و هنوز هم در حال یاد گرفتن هستم. پس با اراده و پشتکار می تونیم موفق بشیم. (اگر سوالی داشتی می تونی برام ایمیل بزنی و یا توی سایت مطرح کنی. من حتما کمکت می کنم) elmbama.ir@gmail.com



روش های درج CSS

سه روش برای درج CSS وجود دارد که شامل موارد زیر هستند:

- 1- روش خارجی
- 2- روش داخلی
- 3- روش درون خطی

روش خارجی: در این روش شما می توانید به راحتی تنها با تغییر یک فایل تمامی صفحاتی را که به این فایل CSS لینک شده اند را تغییر استایل بدهید.

در این روش یک فایل با پسوند CSS ایجاد می کنیم و همه کدهای CSS رو توی این فایل قرار میدیم و وقتی که بخواهیم صفحه ای را به این فایل قالب بدیم کافیه که یک لینک به این فایل در صفحه قرار بدیم. توجه داشته باشیم که داخل فایل CSS هیچ کد html ی قرار نمیگیره .

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

روش داخلی: این روش تنها زمانی استفاده میشه که بخواهین فقط برای یک صفحه استایل طراحی کنین . برای استفاده از این روش کافیه که داخل تگ <Head> از تگ <style></style> استفاده کنین و کدهای CSS رو بین این دو تا تگ قرار بدین. مثل روش زیر



```
<head>
<style>
body {
  background-color: linen;
}
h1 {
  color: maroon;
  margin-left: 40px;
}
</style>
</head>
```

روش درون خطی: در این روش می شه فقط به یک تگ استایل بدیم و می تونیم از همه دستورات CSS در این روش نیز استفاده بکنیم. برای انجام این کار کافیه که از دستور **style** داخل تگ استفاده کنیم . و بعد از مساوی داخل دوتا دابل کوتیشن دستورات CSS رو قرار میدیم . توجه داشته باشیم که بین دستورات CSS حتما سمیکالن قرار بدین. شیوه استفاده از این روش رو تو مثال زیر قرار دادم.

```
<h1 style="color:blue;margin-left:30px;">این یک عنوان است</h1>
```

توجه: اگه شما به یک المنت با سه روشی که گفتم استایل بدین ، مثلا به تگ **p** به سه روش بالا رنگ قرمز، زرد و آبی بدین . همیشه در همه حال استایلی که در روش درون خطی لحاظ کردی اولویت بالاتری داره و جایگزین سایر استایلها میشه ، بعد از اون اولویت بین خارجی و داخلی ، اونی بالاتره که زودتر تعریف شده باشی. یعنی اگه شما اول لینک CSS خارجی رو قرار بدین و بعد از تگ استایل برای تعریف CSS داخلی استفاده بکنین استایلی که داخل فایل هستش اولویت بالاتری داره و اگر این روش رو برعکس کنیم یعنی اول استایل تگ داخلی رو قرار بدیم و بعد لینک استایل خارجی رو اولویت با استایل تگ داخلی میشه.



ساعت دوم



در این ساعت می خواهیم به مقدار عمیق تر وارد **html** بشیم و زوایا و دستورات بیشتری از این زبان رو یاد بگیریم. به عبارت دیگه برای ساخت یک سایت ما باید المنت های **html** رو تا حدی یاد داشته باشیم و بتونیم ازش استفاده کنیم تا بعد شروع کنیم **CSS** رو یاد بگیریم و سایتمون رو زیبا کنیم.

معرفی Spans و Divisions

Divisions یا همان تگ **<div>** و **Spans** یا همان تگ **** صرفاً جهت تقسیم بندی صفحه و ظاهر صفحه وب ایجاد شده اند. این دو تگ در واقع به تنهایی هیچ معنی خاصی ندارند و اگر بخواهیم در دنیای واقعی این دو تگ رو نگاه کنیم می تونیم به ظروف اشاره کنیم که داخل این ظروف که در سایزهای مختلف طراحی شده اند، اشیا مختلفی قرار می گیرند و این اشیا نمی تونن خارج از ظرف قرار بگیرند و مجبورند که در داخل این ظروف خودشون رو تطبیق بدن.

بلوک در مقابل المنت های درون خطی

بیشتر المنت ها در **html** به صورت بلوک هستند و برخی هم به صورت درون خطی هستند. خب حالا چه تفاوتی بین این المنت ها وجود داره؟ در المنت های بلوک می توان چندین خط اطلاعات را داشته باشیم مثل پاراگراف. اما در المنت های درون خطی فقط می توان یک خط اطلاعات را داشته باشیم.

<div> و **** المنت هایی هستند که بسیار در طراحی صفحات وب ارزشمند هستند و بسیار پرکاربرند که در ادامه خود شما متوجه این موضوع می شوید.

<div> یک المنت بلوک هست که به ما این امکان رو میده که بتونیم محتویات با سایزهای بزرگ رو گروه بندی کنیم.

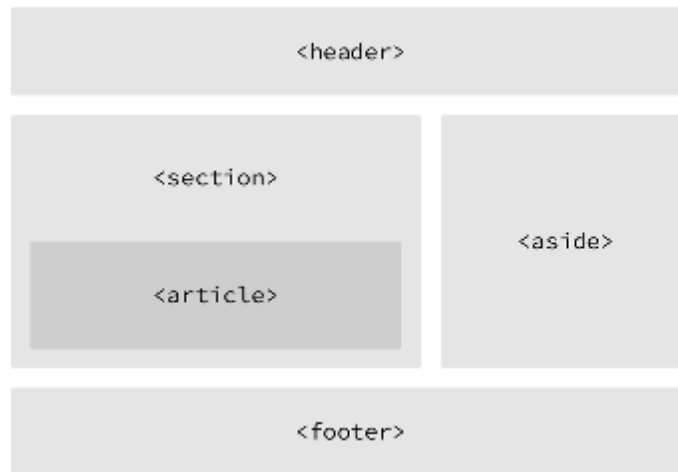
**** یک المنت درون خطی هست که این امکان رو به ما میده که گروه بندی های کوچکتري رو بدون بلوک ایجاد کنیم.

ما معمولاً از **id** و **class** و یا نام المنت برای انتخاب و طراحی این المنت ها استفاده می کنیم که در ساعت پیش با این موضوع آشنا شدین.



اسکلت صفحه وب

برای مدت خیلی زیادی برای اینکه اسکلت یک صفحه وب رو که منظورم همون عنوان و منوهای سمت راست و متن سایت هست رو طراحی کنیم از تگ `<div>` استفاده می شد. اما با اومدن **html5** دیگه این کار خیلی خیلی راحت تر شده و ما می تونیم با استفاده از تگ هایی که اضافه شدن در **html** به شرح `<header><article><section><aside><nav>` و `<footer>` هست و در شکل زیر من این تگ ها رو به شما به صورت واضح در یک صفحه وب نشون دادم



همه این المنت هایی که به شما معرفی شد به صورت کاملا واضح و مشخص با معنی که دارن توی وب نشون داده می شن. و همشون به صورت بلوک هستند و می تونن چندین بار در یک صفحه وب استفاده بشن.

خب اجازه بدین که بصورت جزئی تر به هرکدوم از این المنت ها بپردازیم:

Header

از این المنت برای تعریف عنوان و یا سربرگ سایت استفاده می شود که معمولا می تواند شامل یک عنوان و یا یک منو باشد.

Navigation



از این تگ برای دسته بندی و کنار هم قرار دادن گروهی از لینک ها استفاده می شه . لینک هایی که درون این تگ تعیین شوند به ترتیب در کنار هم قرار می گیرند . تمام لینک های یک صفحه را نباید در تگ تعریف نمود . فقط لینک های اصلی و پایه صفحه در این بخش تعریف می شوند .

شکل کلی استفاده از یک تگ `<nav>` به صورت زیره :

```
<nav>
  <a href="Link 1"> </a>
  <a href="Link 2"> </a>
  <a href="Link 3"> </a>
  .....
</nav>
```

Article

تگ `<article>` نشان دهنده یک ترکیب مستقل در یک سند، صفحه، برنامه، و یا سایت است، به عنوان مثال می تونه شامل یک پست انجمن، مقاله یک مجله یا روزنامه ، یک پست وبلاگ، داستانهای خبری ، نظرات یک کاربر ثبت نام کرده ، ویجت ها و یا ابزارهای تعاملی و یا هر مورد دیگر مستقل از محتوا باشه

هنگامی که یک عنصر `<article>` به صورت تو در تو قرار بگیره، عنصر درونی نشان دهنده یک مقاله مربوط به عنصر خارجی است. به عنوان مثال، نظرات یک پست وبلاگ می تواند در یک عنصر `<article>` و در داخل عنصر `<article>` دیگر که نشاندهنده پست وبلاگ است ، قرار بگیرد .

```
<body>
  <article>
    .....
  </article>
</body>
```



یک عنصر `<article>` علاوه بر محتوا، معمولاً دارای یک عنوان (اغلب در یک عنصر هدر) و گاهی اوقات یک پاصفحه میباشد. ساده ترین راه برای درک `<article>` استفاده از آن در وبلاگ است، هر ورودی وبلاگ را نیز میتوان در داخل یک عنصر `<article>` قرار داد. از این عنصر برای محتوای صفحات استاتیک نیز میتوان استفاده کرد مانند صفحه درباره ما، تماس با ما و ...

Section

تگ `section` از تگ های مفهومی HTML5 است. بر خلاف سایر تگ های مفهومی ارائه شده در HTML5 کاربرد این تگ برای طراحان کمی سخت است. شاید هر زمانی که از این تگ استفاده می کنید به این فکر بیفتید که آیا از این تگ به درستی استفاده کرده اید یا خیر.

تگ `section` جایگزینی برای `div` نیست. هرچند سایر تگ های HTML5 مانند `header`، `footer`، `main` و ... با `div`هایی که آن قسمت ها را در بر میگیرند جایگزین شدند؛ ولی برای `section` اینگونه نیست. اگر به تعریف این تگ در مستندات مربوط به HTML5 را بخوانید، به این موضوع پی خواهید برد. تگ `section` ظرفی است برای محتوایی که ارتباط معنایی با هم دارند و دارای عنوان (`heading`) هستند. اگر نمی توانید برای `section` عنوانی تعریف کنید پس می توانید بگویید که درست از `section` استفاده نکرده اید و باید از `div` به جای آن استفاده کنید. دلیل استفاده از عنوان برای `section` این است که اطلاعات مفهومی راجع به آن `section` را بیان کند.

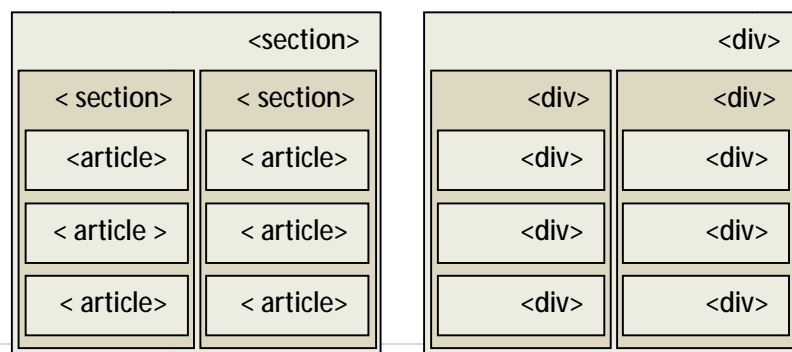
در کاربرد این تگ در طراحی وب سایت به این نکات توجه کنید:

-از `section` هنگامی که فقط نیاز به استایل دادن و اسکرپت نویسی دارید استفاده نکنید.

-هنگامی از این تگ استفاده کنید که محتوای شما از نظر مفهومی دارای تم مشخصی است

-حتماً به تگ خود عنوان بدهید.

به عکس زیر توجه کنید تا به تفاوت این تگ با تگ `div` و `article` پی ببرید





Aside

تگ `<aside>` نشان دهنده یک بخش از صفحه می باشد که شامل محتوایی است که در حاشیه محتوای اصلی مربوط به آن قرار میگیرد و می تواند جدا از محتوای اصلی در نظر گرفته شود.

این بخش اغلب به عنوان ستون های فرعی نشان داده میشود. آنها اغلب حاوی توضیحات جانبی، مانند تعریف لغات یا چیزهایی که کمتر به محتوای اصلی مرتبط هستند مانند تبلیغات. زندگینامه نویسنده و یا برنامه های کاربردی وب سایت ها ، اطلاعات پروفایل و یا لینک وبلاگ های مرتبط

```
<aside>
  <h4>Epcot Center</h4>
  <p>The Epcot Center is a theme park in Disney World, Florida.</p>
</aside>
```

Footer

تگ `<footer>` برای یک سند و یا بخش، یک پا صفحه تعریف می کند و شامل اطلاعات عناصری است که در داخل آن قرار دارند. **Footer** یا پا صفحه معمولا شامل اطلاعات نویسنده ی سند، اطلاعات کپی رایت، لینک شرایط استفاده ، اطلاعات تماس و غیره میباشد.

در یک سند **html** ممکن است چند عنصر **footer** وجود داشته باشد. و این مطلب رو در نظر داشته باشید که اطلاعات تماس داخل یک المنت **footer** باید در داخل المنت **address** قرار بگیرد.

اگرچه این المنت میتواند در داخل یک المنت **header** یا یک المنت **footer** دیگر قرار بگیرد اما نمیتواند حاوی المنت **header** باشد .



خب تا اینجا به صورت کلی با html آشنا شدیم و از اینجا به بعد مجدد می ریم سراغ CSS
و به صورت جزئی و کامل CSS رو بررسی می کنیم.



رنگ ها

رنگ توی CSS اغلب به سه روش استفاده میشن

- اولین روش برای اختصاص رنگ به المنت خاص استفاده از نام خود رنگ هستش مثلا برای رنگ قرمز کافیه که `red` رو جلوی خصیصه قرار بدین

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

مثال

در این مثال به تگ `p` رنگ قرمز تخصیص دادم.

```
<html>  
<head>  
<style>  
p {  
  color: gray;  
  text-align: center;  
}  
</style>  
</head>  
<body>  
<p>سلام به شما</p>  
<p>من یک تگ پاراگراف هستم</p>  
</body>  
</html>
```

خروجی

سلام به شما
من یک تگ پاراگراف هستم

- دومین روش برای اختصاص رنگ به المنت استفاده از دستور `RGB` به روش زیر هستش
`RGB(255,255,255)`

که عدد اول داخل این دستور برای رنگ قرمز و عدد دوم برای رنگ سبز و عدد سوم برای رنگ آبی استفاده میشه که مقدارش از 0 هستش تا 255 و هرچه به صفر نزدیکتر بشیم رنگ تیره تر میشه

```
<html>  
<head>  
<style>  
p {  
  color: RGB(255,0,0);  
  text-align: center;  
}  
</style>  
</head>  
<body>  
<p>سلام به شما</p>  
<p>من یک تگ پاراگراف هستم</p>  
</body>  
</html>
```

خروجی

سلام به شما
من یک تگ پاراگراف هستم



- سومین روش استفاده از کد هگز هشتش که با قرار گرفتن علامت # معرف این روش همیشه و به صورت زیر استفاده میشه

#ff00ff

در این روش دو رقم اول بیانگر رنگ قرمز و دو رقم دوم بیانگر رنگ سبز و دو رقم آخر برابر رنگ آبی هشتش که از 0 تا 9 و از a تا f هرکدوم از ارقام می تونن مقدار بگیرن. و هرچه به f نزدیکتر بشن رنگ ها روشن تر میشه.

```
<html>
<head>
<style>
p {
  color: #ff0000;
  text-align: center;
}
</style>
</head>
<body>
<p>سلام به شما</p>
<p>من یک تگ پاراگراف هستم</p>
</body>
</html>
```

در قسمت زیر چند رنگ رو به صورت مثال قرار دادم که با مفهوم رنگ بیشتر آشنا بشین

Color	Name
	Red
	Green
	Blue
	Orange
	Yellow
	Cyan
	Black



Color	RGB
	rgb(255,0,0)
	rgb(0,255,0)
	rgb(0,0,255)
	rgb(255,165,0)
	rgb(255,255,0)
	rgb(0,255,255)

Color	HEX
	#FF0000
	#00FF00
	#0000FF
	#FFA500
	#FFFF00
	#00FFFF



در CSS از دستور **background** برای تخصیص رنگ به المنت ها استفاده میشه که مجموعه دستورات **background** به شرح زیره:

background-color

background-image

background-repeat

background-attachment

background-position

حالا هر کدام از این دستورات رو به صورت کامل توضیح می دم

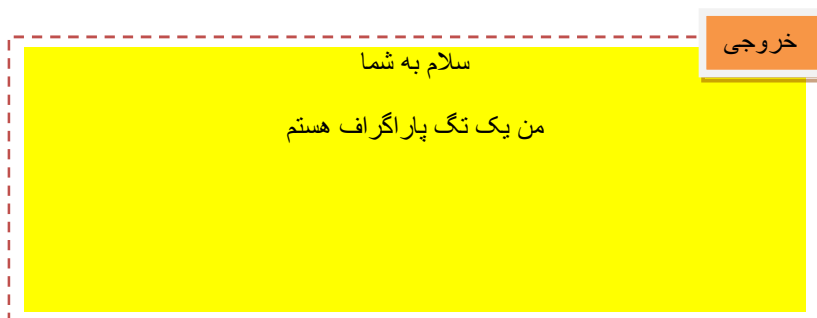
background-color



از این دستور برای درج رنگ برای پس زمینه المنت استفاده میشه

خب برای اینکه رنگ پس زمینه صفحه رو تغییر بدیم از روش زیر استفاده می کنیم

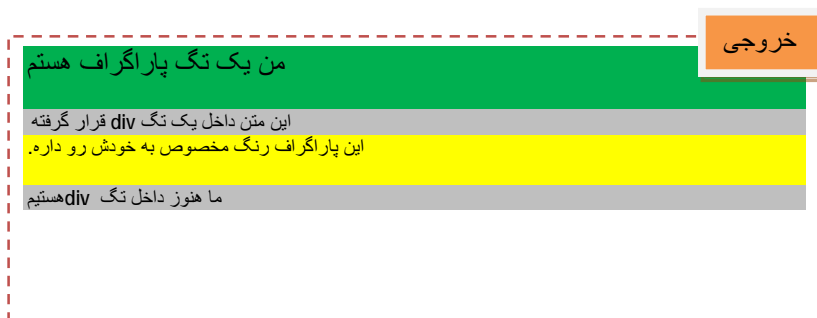
```
<html>
<head>
<style>
body {
  background-color:yellow;
}
</style>
</head>
<body>
<p>سلام به شما</p>
<p>من یک تگ پاراگراف هستم</p>
</body>
</html>
```



برای تخصیص رنگ به پس زمینه می تونیم از همون شرایطی که تو قسمت رنگ ها توضیح دادم استفاده کنیم یعنی اینکه می تونیم از نام خود رنگ استفاده کنیم و یا از دستور **rgb** استفاده کنیم و یا اینکه از روش هگز که با # می اومد استفاده کنیم.

در مثال زیر اومدیم برای چندتا تگ رنگ متفاوتی رو لحاظ کردیم:

```
<html>
<head>
<style>
h1 {
  background-color: green;
}
div {
  background-color: gray;
}
p {
  background-color: yellow;
}
</style>
</head>
<body>
<h1>یک مثال ساده از رنگ پس زمینه</h1>
<div>
این متن داخل یک تگ div قرار گرفته
<p>این پاراگراف رنگ مخصوص به خودش رو داره.
ما هنوز داخل تگ div هستیم
</div></body>
</html>
```





background-image

با استفاده از این دستور می توانیم برای یک المنت عکس لحاظ کنیم . به صورت پیش فرض عکسی که توی یک المنت قرار میدیم اگه کوچیک باشه توی کل المنت تکرار میشه. برای اینکه یک عکس رو پس زمینه صفحه کنیم به شکل زیر انجام میدیم:

```
<html>
<head>
<style>
body {
  background-image: url("mypic.jpg");
}
</style>
</head>
<body>
<p>سلام به شما</p>
</body>
</html>
```



به صورت پیش فرض عکسی رو که شما تو پس زمینه یک المنت قرار می دین به صورت افقی و عمودی تکرار میشه . حالا شما با این دستور میتونین این وضعیت رو کنترل کنین.

فرض کنین که یک عکس رو می خواهیم به صورت افقی فقط تکرار کنیم به روش زیر انجام میدیم. من یک عکس رو با نرم افزار فوتوشاپ طراحی کردم و حالا می خوام اونو به صورت افقی تکرار کنم. روش کار ساده است ، نگاه کنین:

عکسی که من طراحی کردم به شکل زیره

اسم عکس من r.jpg هستش

حالا اینو تو CSS تعریف می کنم.



```
<html>
<head>
<style>
body {
  background-image: url("r.jpg");
  background-repeat: repeat-x;
}
</style>
</head>
<body>
<p>سلام به شما</p>
</body>
</html>
```

پس اگر مقدار **repeat-x** رو برای این دستور لحاظ کنیم عکس ما به صورت افقی تکرار میشه.

واگه بخواهیم که عکس به صورت عمودی تکرار بشه کافیه از مقدار **repeat-y** برای این دستور استفاده کنیم

و حالا اگه بخواهیم که عکس تکرار نشه کافیه از دستور **no-repeat** استفاده کنیم.



ساعت سوم



Fonts

خب برای اینکه با مفهوم فونت‌ها بیشتر آشنا بشین دوتا فونت **Sans-serif** و **Serif** رو تو شکل زیر مقایسه کردیم که چه تفاوتی با هم دیگه دارن



تفاوت با رنگ قرم نمایش داده شده است

Font Family

با استفاده از دستور **font-family** می‌توانیم چندین فونت را برای مرورگر تعریف کنیم، در صورتی که مرورگر فونت اول رو نتونست پشتیبانی بکنه فونت بعدی رو در نظر می‌گیره و همینطور به ترتیب. در ضمن هیچ محدودیتی هم وجود نداره که چندتا فونت رو ما توی این دستور بکار ببریم، و نکته آخر اینکه اگه اسم فونت از چند کلمه تشکیل شده باشه باید حتما اون فونت رو داخل دابل کوتیشن قرار بدیم و فونت‌ها رو باید با استفاده از کاما از همدیگه جدا بکنیم. به مثال زیر توجه کنید

```
p {  
  font-family: "Times New Roman", Times, serif;  
}
```

مثال

در این دوتا فونت برای فونت فامیلی در نظر گرفتیم که شرایطی رو که ذکر کردیم رو لحاظ کردیم.

```
<html>  
<head>  
<style>  
p {  
  font-family: "Times New Roman", Times, serif;  
}  
</style>  
</head>  
<body>  
<p>Hello my dear!</p>  
</body>  
</html>
```

Hello my dear!

خروجی



Font Style

از دستور **font-style** برای اینکه متن رو به صورت مورب نمایش دهیم استفاده میشه و سه خصوصیت هم داره که به شرح زیره:

normal: این خصوصیت متن رو به صورت معمولی نمایش می دهد

italic: این خصوصیت متن رو به صورت مورب نمایش می دهد

oblique: این خصوصیت متن رو به صورت لم داده نمایش می دهد (این خصوصیت خیلی شبیه به **italic** هستش و خیلی به ندرت استفاده میشه)

```
p.normal{
  font-style: normal;
}
p.italic{
  font-style: italic;
}
p.oblique{
  font-style: oblique;
}
```

مثال

در این مثال هر سه حالت رو در سه کلاس قرار دادیم

```
<html>
<head>
<style>
p.normal {
  font-style: normal;
}
p.italic {
  font-style: italic;
}
p.oblique {
  font-style: oblique;
}</style>
</head>
<body>
<p class=" normal">Hello my dear!</p><br/>
<p class=" italic">Hello my dear!</p><br/>
<p class=" oblique">Hello my dear!</p>
</body>
</html>
```

خروجی

Hello my dear!

Hello my dear!

Hello my dear!



Font Size

خصوصیت **font-size** برای تعیین اندازه متن به کار می رود

مدیریت اندازه متن توی طراحی وبسایت خیلی با اهمیت هستش و خیلی مهمه که اندازه متنون یک تناسبی با هم دیگه داشته باشند ، مسلما اندازه متن عنوان و متن پاراگراف باید با همدیگه متفاوت باشند.

این موضوع رو همیشه توجه داشته باشین که برای عناوین حتما از تگ های `<h1>`-`<h6>` استفاده کنید و برای پاراگراف ها از تگ `<p>`

اندازه متون می توانند هم به صورت مطلق و ثابت باشند و هم به صورت وابسته و متغیر

اندازه ثابت:

- در این روش اندازه متن ثابت هستش و هیچگونه تغییری نمی کنه
- به کاربر اجازه نمی ده که بتونه اندازه متن رو تغییر بده.
- این روش وقتی مفید هست که اندازه خروجی کاملا مشخص باشه

اندازه متغیر:

- در این روش اندازه متن به المنت های اطراف متن وابسته هستش
- به کاربر اجازه داده می شه که بتونه اندازه متن رو تغییر بده

اگر شما مقداری رو برای اندازه متن در نظر نگیرید به صورت پیشفرض مقدار **16px** برای متن در نظر گرفته میشه

(16px=1em).

تعیین اندازه متن با استفاده از پیکسل:

استفاده از پیکسل برای تعیین اندازه متن می تونه کنترل زیادی رو به کاربر برای مدیریت متن بده



```
h1{
  font-size: 40px;
}
h2{
  font-size: 30px;
}
p{
  font-size: 14px;
}
```

مثال

```
<html>
<head>
<style>
h1 {
  font-size: 40px;
}
h2 {
  font-size: 30px;
}
p {
  font-size: 14px;
}
</style>
</head>
<body>
<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
</body>
</html>
```

خروجی

This is heading 1

This is heading 2

This is a paragraph.

This is another paragraph.

• اگر شما از پیکسل برای اندازه فونت استفاده کنید می توانید از طریق زوم مرورگر اندازه متن را تغییر دهید.

تعیین اندازه متن با استفاده از EM:

برای اینکه به کاربر بتوانید این اجازه را بدهید که بتواند از طریق تنظیمات مرورگر اندازه متن را تغییر دهد از em به جای پیکسل برای تغییر اندازه متن استفاده کنید.



اگر شما سایزی را برای متن در نظر نگیرید به صورت پیش فرض $1em$ در نظر گرفته می شود که $1em$ برابر با 16 پیکسل می باشد .

برای تبدیل em به پیکسل کافی است که از فرمول زیر استفاده کنید:

$$\text{Pixels}/16=1em$$

```
h1 {
  font-size: 2.5em; /* 40px/16=2.5em */
}
h2 {
  font-size: 1.875em; /* 30px/16=1.875em */
}
p {
  font-size: 0.875em; /* 14px/16=0.875em */
}
```

مثال

```
<html>
<head>
<style>
h1 {
  font-size: 2.5em; /* 40px/16=2.5em */
}
h2 {
  font-size: 1.875em; /* 30px/16=1.875em */
}
p {
  font-size: 0.875em; /* 14px/16=0.875em */
}
</style>
</head>
<body>
<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<p>This is a paragraph.</p>
</body>
</html>
```

This is heading 1

This is heading 2

This is a paragraph.

برابر با $1em$ در نظر گرفته می شود که $1em$ اگر شما سایزی را برای متن در نظر نگیرید به صورت پیش فرض 16 پیکسل می باشد .

خروجی

این مثالی که ارائه شد مشابه مثالی قبلی بود که با پیکسل مقدار دهی کرده بودیم، تفاوتی که این مثال داره اینه که باعث می شه اندازه متن توی سایر مرورگرها تطبیق داده بشه. اما متاسفانه اندازه متن توی ورژن های قدیمی تر IE ممکنه که بزرگتر و یا کوچکتر از اندازه متنی هستش که باید باشه و ما تعریف کردیم.



ترکیب درصد و em در تعیین سایز متن

در این روش برای body به درصد اندازه متن را مشخص می کنیم ، این روش در همه مرورگرها کارایی دارد و متن رو با همون اندازه ای که تعیین می کنیم نمایش می دهد

```
body {
  font-size: 100%;
}
h1 {
  font-size: 2.5em;
}
h2 {
  font-size: 1.875em;
}
p {
  font-size: 0.875em;
}
```

مثال

```
<html>
<head>
<style>
body {
  font-size: 100%;
}
h1 {
  font-size: 2.5em;
}
h2 {
  font-size: 1.875em;
}
p {
  font-size: 0.875em;
}
</style>
</head>
<body>
<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<p>This is a paragraph.</p>
<p>برابر با 16 پیکسل می باشد . 1em در نظر گرفته می شود که 1em اگر شما سایزی را برای متن در نظر نگیرید به صورت پیش فرض
</p>
</body>
</html>
```

This is heading 1

This is heading 2

This is a paragraph.

برابر با 1em در نظر گرفته می شود که 1em اگر شما سایزی را برای متن در نظر نگیرید به صورت پیش فرض 16 پیکسل می باشد .

خروجی



در این روش خیلی عالی می تونین متن رو نمایش بدین و توی همه مرورگرها به راحتی کاربر می تونه سایز متن رو تغییر بده و یا صفحه رو زوم کنه و متن مطابق با اون تغییر میکنه.

Font Weight

با استفاده از خاصیت **font-weight** می تونیم ضخامت متن رو تعیین کنیم

```
p.normal {
  font-weight: normal;
}
p.light {
  font-weight: lighter;
}
p.thick {
  font-weight: bold;
}
p.thicker {
  font-weight: 900;
}
```

مثال

```
<html>
<head>
<style>
p.normal {
  font-weight: normal;
}
p.light {
  font-weight: lighter;
}
p.thick {
  font-weight: bold;
}
p.thicker {
  font-weight: 900;
}
</style>
</head>
<body>
<p class="normal">This is a paragraph.</p>
<p class="light">This is a paragraph.</p>
<p class="thick">This is a paragraph.</p>
<p class="thicker">This is a paragraph.</p>
</body>
</html>
```

This is a paragraph.

This is a paragraph.

This is a paragraph.

This is a paragraph.

خروجی



Font Variant

با استفاده از خاصیت **font-variant** می توان متن را با فونت **small-caps** نمایش داد

در فونت **small-caps** کلیه حروف کوچک به حروف بزرگ تبدیل می شوند و حروف بزرگ تغییری نمی کنند. البته با این تفاوت که حروفی کوچکی که تبدیل به حروف بزرگ انگلیسی شده اند در همان اندازه حروف کوچک نمایش داده می شوند. این خاصیت کاربردی برای زبان فارسی ندارد.

```
p.normal {  
    font-variant: normal;  
}  
  
p.small {  
    font-variant: small-caps;  
}
```

مثال

```
<html>  
<head>  
<style>  
p.normal {  
    font-variant: normal;  
}  
p.small {  
    font-variant: small-caps;  
}  
</style>  
</head>  
<body>  
<p class="normal">My name is Hege Refsnes.</p>  
<p class="small">My name is Hege Refsnes.</p>  
</body>  
</html>
```

My name is Hege Refsnes.

MY NAME IS HEGE REFSNES.

خروجی



با استفاده از CSS می توان استایل های متفاوتی رو به لینک ها داد، در قسمت زیر چند نمونه استایلی که به لینک داده ایم رو نشون دادم

Text Link

Text Link

Link Button

Link Button

لینک ها رو میشه با خصوصیت های مختلف CSS استایل بدیم . یعنی می تونیم با font فونتشنون رو تغییر بدیم با color رنگ لینک رو تغییر بدیم و یا با background رنگ پس زمینه لینک رو تغییر بدیم. در قسمت زیر یک مثال برای لینک زدیم

```
a {  
  color: hotpink;  
}
```

مثال

در این مثال به تگ لینک با رنگ صورتی ایجاد کردیم.

```
<html>  
<head>  
<style>  
a {  
  color: hotpink;  
}  
</style>  
</head>  
<body>  
<p><b><a href="default.asp" target="_blank">This is a link</a></b></p>  
</body>  
</html>
```

This is a link

خروجی

همچنین لینک رو میشه بنا به موقعیتی که داره استایل های مختلفی بهش بدیم.

لینک کلا می تونه توی 4 حالت قرار بگیره که شامل

a:link - لینک در حالت معمول که مشاهده نشده



a:visited - لینک در حالتی که کاربر اونو مشاهده کرده

a:hover - لینک در حالتی که کاربر موس رو روی اون قرار داده

a:active - لینک در حالتی که کاربر روی اون کلیک کرده

```
/* حالت مشاهده نشده */
```

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

```
/* حالت مشاهده شده */
```

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

```
/* حالتی که موس روی اون قرار گرفته */
```

```
a:hover {  
  color: hotpink;  
}
```

```
/* حالت انتخاب */
```

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

مثال

وقتی برای حالت های مختلف لینک استایل دهی می کنید این نکته رو در نظر داشته باشید که لازم هستش ترتیب این استایل دهی رو رعایت کنید

- **a:hover** باید بعد از **a:link** و **a:visited** استایل دهی بشه
- **a:active** باید بعد از **a:hover** استایل دهی بشه

Text Decoration

خصوصیت **text-decoration** برای حذف زیرخط از لینک ها مورد استفاده قرار می گیره



مثال

```
a:link {
  text-decoration: none;
}
a:visited {
  text-decoration: none;
}
a:hover {
  text-decoration: underline;
}

a:active {
  text-decoration: underline;
}
```

Background Color

از خصوصیت **background-color** برای تخصیص رنگ به پس زمینه لینک می توان استفاده کرد

مثال

```
a:link {
  background-color: yellow;
}
a:visited {
  background-color: cyan;
}
a:hover {
  background-color: lightgreen;
}
a:active {
  background-color: hotpink;
}
```

در اینجا یک مثال برای شما آماده کرده ایم برای اینکه ببینید می توانید از خصوصیت های CSS برای تبدیل یک لینک ساده به یک **button** و یا یک **box** استفاده کنید



مثال

```
<html>
<head>
<style>
a:link, a:visited {
  background-color: #f44336;
  color: white;
  padding: 14px 25px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
}
a:hover, a:active {
  background-color: red;
}
</style>
</head>
<body>
<a href="default.asp" target="_blank">This is a link</a>
</body>
</html>
```

خروجی

This is a link



1. Coffee
2. Tea
3. Coca Cola

- Coffee
- Tea
- Coca Cola

لیست ها در Html و خصوصیات لیست در CSS

در html دو نوع لیست اصلی وجود دارد

- لیست های مرتب () - آیتم های این لیست با البت ها نشانه گذاری می شوند مانند همین مطلب
- لیست های نامرتب () - آیتم های این لیست با اعداد و یا حروف نشانه گذاری می شوند

در CSS این خصوصیات برای لیست ها وجود دارد

- علامت ها و یا نشانه های لیست های مرتب را می توان تعیین کرد
- علامت ها و یا نشانه های لیست های نامرتب را می توان تعیین کرد
- به جای علامت ها و یا نشانه های لیست ها می توان تصویر تخصیص داد
- می توان به لیست ها و یا آیتم های لیست رنگ پس زمینه تعیین کرد.

تخصیص مارکرهاى متفاوت به آیتم های یک لیست

با استفاده از خصوصیت **list-style-type** می توان انواع مارکرها را به لیست تخصیص داد. در مثال زیر به چند مورد از این تخصیص مارکرها اشاره می کنیم



```
<html>
<head>
<style>
ul.a {
  list-style-type: circle;
}
ul.b {
  list-style-type: square;
}
ol.c {
  list-style-type: upper-roman;
}
ol.d {
  list-style-type: lower-alpha;
}
</style>
</head>
<body>
<p>Example of unordered lists:</p>
<ul class="a">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ul>
<ul class="b">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ul>
<p>Example of ordered lists:</p>
<ol class="c">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ol>
<ol class="d">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ol>

</body>
</html>
```

Example of unordered lists:

- Coffee
- Tea
- Coca Cola

- Coffee
- Tea
- Coca Cola

Example of ordered lists:

- I. Coffee
- II. Tea
- III. Coca Cola

- a. Coffee
- b. Tea
- c. Coca Cola



تخصیص تصویر برای مارکر در لیست

با استفاده از خصوصیت `list-style-image` می توان یک تصویر را برای مارکر آیتم های لیست در نظر گرفت

```
<html>
<head>
<style>
ul {
  list-style-image: url('sqpurple.gif');
}
</style>
</head>
<body>
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ul>
</body>
</html>
```

مثال

خروجی

- Coffee
- Tea
- Coca Cola

تعیین موقعیت آیتم های لیست

با استفاده از خاصیت `list-style-position` می توان موقعیت آیتم های لیست را در متن مشخص کرد. به مثال زیر توجه کنید تا با نحوه کارکرد این دستور به خوبی آشنا شوید



مثال

خروجی

```
<html>
<head>
<style>
ul.a {list-style-position:inside;}
ul.b {list-style-position:outside;}
</style>
</head>

<body>
<p>The following list has list-style-position: inside:</p>
<ul class="a">
<li>Coffee</li>
<li>Tea</li>
<li>Coca Cola</li>
</ul>

<p>The following list has list-style-position: outside:</p>
<ul class="b">
<li>Coffee</li>
<li>Tea</li>
<li>Coca Cola</li>
</ul>

<p>"list-style-position: outside" is the default setting.</p>
</body>
</html>
```

The following list has list-style-position: inside:

- Coffee
- Tea
- Coca Cola

The following list has list-style-position: outside:

- Coffee
- Tea
- Coca Cola

"list-style-position: outside" is the default setting.

حذف تنظیمات پیش فرض

با استفاده از خصوصیت **list-style-type:none** می توانید مارکرها یا بالت ها را حذف کنید . همچنین در داخل لیست معمولاً از **padding** و **margin** برای ایجاد فاصله استفاده می شود که با صفر قرار دادن آنها در **ul** و **ol** می توانید این فاصله ها را لحاظ نکنید



مثال

```
ul {  
  list-style-type: none;  
  margin: 0;  
  padding: 0;  
}
```

کوتاه سازی خصوصیت های لیست

با استفاده از خصوصیت **list-style** می توان تمامی خصوصیت ها را یکجا تعریف کرد

مثال

```
ul {  
  list-style: square inside url("sqpurple.gif");  
}
```

وقتی که از این خصوصیت برای کوتاه سازی کلیه دستورات استفاده می کنید ، حتما باید ترتیب آنها را رعایت کنید. ترتیب استفاده این خصوصیت ها اینجا برای شما لیست شده:

- **list-style-type**: اگر تصویر به مارکر اختصاص بدیم مقداری که برای این خصوصیت در نظر بگیریم نمایش داده میشه و تصویر نادیده گرفته میشه
- **list-style-position**: این خصوصیت می تونه دو مقدار **inside** و **outside** داشته باشه
- **list-style-image**: این خصوصیت یک تصویر به مارکر اختصاص می ده

تخصیص رنگ به لیست ها

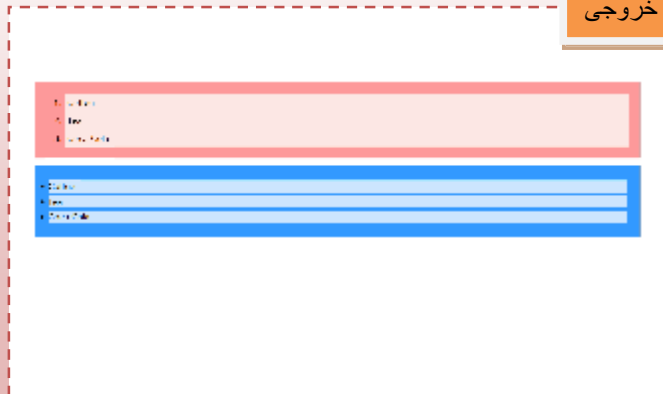
برای اینکه لیست هایی جذاب تر و زیباتری داشته باشیم می تونیم به اونها رنگ بدیم ، البته دقت کنید که رنگ هایی که انتخاب می کنین باهمدیگه همخوانی داشته باشه چه از لحاظ سردی و گرمی رنگ و اینکه رنگها در کنار همدیگه جلوه ای زیبایی به لیست شما بده. برای اینکه به یک لیست رنگ بدیم کافیه که به **** یا **** رنگ تخصیص بدیم و اگه بخواهیم به آیتم های لیست رنگ بدیم کافیه به **** رنگ تخصیص بدیم. به مثال زیر توجه کنید.



مثال

خروجی

```
ol {  
  background: #ff9999;  
  padding: 20px;  
}  
ul {  
  background: #3399ff;  
  padding: 20px;  
}  
ol li {  
  background: #ffe5e5;  
  padding: 5px;  
  margin-left: 35px;  
}  
ul li {  
  background: #cce5ff;  
  margin: 5px;  
}
```





ساعت چهارم



Tables

جدول ایجاد شده توسط **html** را می شود توسط **CSS** به شکلی جذاب و زیبا طراحی کرد.

شرح کالا	قیمت	موجودی
کیف چرم	1500000	20
ساعت مچی	2500000	10
کفش مردانه	700000	50
کفش زنانه	550000	10
پیراهن مردانه	450000	50
تیشرت مردانه	880000	45
توال	250000	23

Table Borders

برای اینکه بتوانیم به یک جدول حاشیه بدهیم کافیست که از خصوصیت **border** استفاده کنیم

در مثال زیر به یک جدول ، برای تگ های `<table>`، `<th>` و `<td>` حاشیه تخصیص داده ایم.

```
table, th, td {
```

```
border: 1px solid black;
```

```
}
```

خروجی

مثال

سن	نام و نام خانوادگی
25	علی جعفری
36	حامد قاسمی

توجه داشته باشیم که در مثال بالا جدول ما دوتا حاشیه داره ، این به خاطر اینه که ما برای تگ `<th>` و تگ `<td>` حاشیه اختصاص دادیم. یعنی این دو تا تگ هرکدوم حاشیه مجزایی دارند.



Collapse Table Borders

خاصیت **border-collapse** حاشیه های جدول رو به یک حاشیه تبدیل می کند. طریق استفاده از این خاصیت رو در مثال زیر توضیح داده شده :

```
table {  
  border-collapse: collapse;  
}  
table, th, td {  
  border: 1px solid black;  
}
```

خروجی

سن نام و نام خانوادگی	
علی جعفری	25
حامد قاسمی	36

مثال

اگه شما می خواهید که فقط دورتادور جدول حاشیه داشته باشه کافیه که دستورات مثال بالا رو داخل تگ **table** قرار بدین. اینکارو تو مثال پایین نشون دادم

```
table {  
  border-collapse: collapse;  
  border: 1px solid black;  
}
```

خروجی

سن نام و نام خانوادگی	
علی جعفری	25
حامد قاسمی	36

مثال

Table Width and Height

برای تنظیم عرض و ارتفاع جدول از خصوصیت های **width** و **height** استفاده می کنیم. در مثال زیر ما عرض جدول رو برابر 100 درصد قرار دادیم و ارتفاع تگ **<th>** رو برابر 50 درصد قرار دادیم. به مثال توجه کنید:



```
table {  
  width: 100%;  
}  
th {  
  height: 50px;  
}
```

نام	فامیل	هزینه
علی	جعفری	\$100
حامد	قاسمی	\$150
حجت	قاسم آبادی	\$300
رضا	حسینی	\$250

خروجی

مثال

Horizontal Alignment

با استفاده از خصوصیت **text-align** می‌توانیم محتویات `<th>` و `<td>` رو به صورت افقی چیدمان کنید یعنی می‌توانید محتویاتشان را در سمت چپ یا راست و یا وسط قرار دهید. به صورت پیش فرض محتویات `<th>` وسط چین است و محتویات `<td>` چپ چین است. در مثال زیر من محتویات تگ `<th>` رو چپ چین می‌کنم. به مثال توجه کنید:

```
table, td, th {  
  border: 1px solid black;  
}  
table {  
  border-collapse: collapse;  
  width: 100%;  
}  
th {  
  text-align: left;  
}
```

نام	فامیل	هزینه
علی	جعفری	\$100
حامد	قاسمی	\$150
حجت	قاسم آبادی	\$300
رضا	حسینی	\$250

خروجی

مثال

Vertical Alignment

خاصیت **vertical-align** محتویات `<th>` و `<td>` رو به صورت عمودی چیدمان می‌کند. یعنی می‌توانید محتویاتشان را در سمت بالا یا پایین و یا وسط قرار دهید. به صورت پیش فرض محتویات `<th>` و محتویات `<td>` وسط چین است. در مثال زیر من محتویات تگ `<td>` رو پایین قرار می‌دهم. به مثال توجه کنید:



```
table, td, th {  
  border: 1px solid black;  
}  
table {  
  border-collapse: collapse;  
  width: 100%;  
}  
td {  
  height: 50px;  
  vertical-align: bottom;  
}
```

مثال

خروجی

نام	فامیل	میزبانه
علی	بختیاری	\$100
حامد	قاسمی	\$150
حجت	قاسم آبادی	\$300
رضا	حسینی	\$250

Table Padding

برای اینکه فاصله بین حاشیه و محتویات تگ های `<th>` و `<td>` را بتوانیم مدیریت کنیم می بایست از خاصیت `padding` استفاده کنیم. در مثال زیر من به محتویات این تگ ها 15 پیکسل تا حاشیه فاصله داده ام. به مثال توجه کنید

```
table, td, th {  
  border: 1px solid black;  
  text-align: left;  
}  
table {  
  border-collapse: collapse;  
  width: 100%;  
}  
th, td {  
  padding: 15px;  
}
```

مثال

خروجی

نام	فامیل	میزبانه
علی	بختیاری	\$100
حامد	قاسمی	\$150
حجت	قاسم آبادی	\$300
رضا	حسینی	\$250



Horizontal Dividers

با استفاده از خصوصیت **border-bottom** می توانیم حاشیه پایین **<th>** و **<td>** را مدیریت کنیم و مابقی حاشیه ها را از بین ببریم. و تنها حاشیه پایینی این تگ ها با توجه به خصوصیت هایی که برای آن در نظر بگیریم ایجاد می شود. به مثال زیر توجه کنید با مطلب خیلی بهتر آشنا می شوید:

مثال

```
table {  
  border-collapse: collapse;  
  width: 100%;  
}  
th, td {  
  padding: 8px;  
  text-align: left;  
  border-bottom: 1px solid #ddd;  
}
```

نام	فامیل	هزینه
طهم	جعفری	\$100
حامد	قاسمی	\$150
حجت	قاسم آبادی	\$300
رضا	حسیناد	\$250

خروجی

Hoverable Table

با استفاده از انتخاب گر **hover**: روی **<tr>** می توانیم با حرکت موس روی این تگ رنگ آن را تغییر دهیم

مثال

```
table {  
  border-collapse: collapse;  
  width: 100%;  
}  
th, td {  
  padding: 8px;  
  text-align: left;  
  border-bottom: 1px solid #ddd;  
}  
tr:hover{background-color:#f5f5f5}
```

نام	فامیل	هزینه
طهم	جعفری	\$100
حامد	قاسمی	\$150
حجت	قاسم آبادی	\$300
رضا	حسیناد	\$250

خروجی



Striped Tables

با استفاده از انتخاب گر **nth-child()** می توان سطرهای جدول را به صورت یکی در میان رنگ بدیم و این باعث میشه که سطرهای جدول خواناتر و زیباتر بشه. و اگه مقدار **even** رو توی این انتخابگر استفاده کنید سطرهای زوج تغییر رنگ پیدا می کنند و اگه مقدار **odd** رو استفاده کنید سطرهای فرد تغییر رنگ پیدا می کنند. توی مثال زیر این خصوصیت رو توضیح دادم: به مثال توجه کنید

مثال

```
table {  
  border-collapse: collapse;  
  width: 100%;  
}  
th, td {  
  text-align: left;  
  padding: 8px;  
}  
tr:nth-child(even){background-color: #f2f2f2}
```

خروجی		
نام	فصل	قیمت
طاهر	حطرت	\$100
حاجت	قاسم	\$150
حجت	قاسم آبادی	\$200
رضا	حمیده	\$250

Table Color

در جداول می توانیم به راحتی روی رنگ ها کار کنیم و به سطرها و ستون ها و محتویات جدول با استفاده از CSS رنگ بدیم. در مثال زیر من به تگ **<th>** رنگ بدیم. به مثال توجه کنید

مثال

```
table {  
  border-collapse: collapse;  
  width: 100%;  
}  
th, td {  
  text-align: left;  
  padding: 8px;  
}  
tr:nth-child(even){background-color: #f2f2f2}  
th {  
  background-color: #4CAF50;  
  color: white;  
}
```

خروجی		
نام	فصل	قیمت
حجت	حطرت	\$100
حاجت	قاسم	\$150
حجت	قاسم آبادی	\$200
رضا	حمیده	\$250



Responsive Table

یک جدول واکنش گرا وقتی که سایز صفحه نمایش خیلی کوچکتر از جدول باشد به جدول اسکرول می دهد و کاربر با اسکرول کردن می تواند سایر محتویات جدول را ببیند. توی مثال زیر این قضیه روشن میشه براتون:

```
<div style="overflow-x:auto;">  
<table>  
... محتویات جدول ...  
</table>  
</div>
```

مثال

مقدار `overflow-x:auto` باعث می شود که جدول در حالت افقی اسکرول شود. و اگر مقدار برابر با `overflow-y:auto` جدول در حالت عمودی اسکرول می شود و اگر از دستور `overflow` به تنهایی استفاده کنیم جدول در حالت افقی و عمودی اسکرول می شود.



display

با استفاده از خصیصه **display** که یکی از مهمترین خصیصیت های CSS می باشد می توان لایه ها را مدیریت کرد.

خصیصیت **display** بیان می کند که یک المنت چطور و چگونه نمایش داده شود.

هر المنت **html** به رویه پیش فرض برای نمایش دارد و مقدار پیش فرض برای اکثر المنت ها مقدار **block** یا **inline** است.

: Block

وقتی برای خصیصیت **display** مقدار **Block** را برای المنت در نظر می گیریم. باعث می شود که المنت در یک خط قرار بگیرد و توانایی استفاده کامل از یک خط را دارد.

این یک المنت **div** هستش که مقدار خصیصیت **display** آن برابر **block** است

در زیر لیست نمونه المنت هایی که خصیصیت **display** برای آن ها **block** است. آورده شده است.

```
<div>  
<h1> - <h6>  
<p>  
<form>  
<header>  
<footer>  
<section>
```

InLine

وقتی برای خصیصیت **display** مقدار **inline** را برای المنت در نظر می گیریم. باعث می شود که المنت به اندازه ای که نیاز دارد از فضای یک خط را اشغال کند و سایر المنت های می توانند بعد از این المنت قرار بگیرند

این یک **span** با مقدار خصیصیت **display** آن برابر **inline** است **پاراگراف است**

در زیر لیست نمونه المنت هایی که خصیصیت **display** برای آن ها **inline** است. آورده شده است.



```
<span>  
<a>  
<img>
```

Display: none;

Display: none این خصوصیت بیشتر با دستورات **jquery** به کار می رود و باعث می شود که یک المنت نمایش داده شود و یا نمایش داده نشود.

مقدار پیش فرض را نادیده بگیرید

بسیاری از المنت ها مقدار پیش فرضی برای خصوصیت **display** دارند اما شما می توانید به راحتی مقدار این خصوصیت را برای المنت تغییر بدهید. این روش یعنی تغییر مقدار خصوصیت **display** به **block** یا **inline** می تواند برای طراحی صفحات وب و ساختن صفحات کارآمد و استاندارد بسیار مفید باشد. یک مثال در قسمت زیر ارائه شده که مقدار خصوصیت **display** تگ **li** را به **inline** تبدیل می کند.

مثال

```
<html>  
<head>  
<style>  
li {  
  display: inline;  
}  
</style>  
</head>  
<body>  
  
<p>نمایش یک منو به صورت افقی</p>  
  
<ul>  
<li><a href="/html/default.asp" target="_blank">HTML</a></li>  
<li><a href="/css/default.asp" target="_blank">CSS</a></li>  
<li><a href="/js/default.asp" target="_blank">JavaScript</a></li>  
</ul>  
  
</body>  
</html>
```



استفاده از خصوصیت **display** برای المنت ها توجهی به نوع المنت نمی کند که این المنت دارای مقدار پیشفرض چه مقداری برای **display** است. و هر المنتی که باشد قابل تغییر است. به مثال زیر توجه کنید. تگ `` دارای خصوصیت **inline** است اما به راحتی خصوصیت **display** را می توانیم **block** کنیم و خاصیت آن را تغییر بدهیم.

مثال

```
<html>
<head>
<style>
span {
  display: block;
}
</style>
</head>
<body>
```

`` مقدار این `span` برابر `block` شده است `` `` یک خط بین این دو `span` فاصله ایجاد شده است ``.

```
</body>
</html>
```

مثال زیر تگ `a` را در حالت **block** نمایش می دهد.

مثال

```
<html>
<head>
<style>
a {
  display: block;
}
</style>
</head>
<body>
```

```
<p>نمایش لینک ها در حالت بلاک</p>
<a href="/html/default.asp" target="_blank">HTML</a>
<a href="/css/default.asp" target="_blank">CSS</a>
<a href="/js/default.asp" target="_blank">JavaScript</a>
```

```
</body>
</html>
```



با استفاده از خصوصیت **display** با مقدار **none** می توانیم یک المنت را مخفی کنیم ، در این روش المنت مورد نظر مخفی می شود و جای آن در وب باقی نمی ماند یعنی سایر المنت ها جای المنت مخفی شده را پر می کنند.

مثال

```
<html>
<head>
<style>
h1.hidden {
  display: none;
}
</style>
</head>
<body>

<h1></h1> این یک عنوانی است که نمایش داده می شود</h1>
<h1 class="hidden">این عنوان نمایش داده نمی شود</h1>
<p>همانطور که در این مثال می بینید خط دوم نمایش داده نمی شود و جای آن نیز باقی نمی ماند</p>

</body>
</html>
```

خصوصیت **visibility:hidden** المنت را مخفی می کند با این تفاوت که جای آن در صفحه وب خالی می ماند و سایر المنت ها جای آن را پر نمی کنند.

مثال

```
<html>
<head>
<style>
h1.hidden {
  visibility: hidden;
}
</style>
</head>
<body>

<h1></h1> این یک عنوانی است که نمایش داده می شود</h1>
<h1 class="hidden">این عنوان نمایش داده نمی شود</h1>
<p>همانطور که در این مثال می بینید خط دوم نمایش داده نمی شود و جای آن نیز باقی می ماند</p>

</body>
</html>
```



width , max-width

استفاده از `margin: auto` و `width, max-width`

همانطور که در قسمت قبلی شرح داده شد ، المنتی که خاصیت `display` آن برابر با `block` بود عرض کامل یک خط را در اختیار می گرفت به نحوی که کل خط رو می تونست پر بکنه و اجازه نمی داد که المنت دیگه ای توی اون خط قرار بگیره.

خب حالا می تونیم برای این عنصر مقداری رو برای `width` در نظر بگیریم و عرض اون رو مشخص کنیم. با اینکار و قرار دادن دستور `margin` با مقدار `auto` باعث میشه که المنت در وسط خط قرار بگیره و فضای اضافی به طور مساوی بین دو طرف المنت تقسیم بشه.

این یک المنت `div` با مقدار خصوصیت `width` آن برابر `500px` است و `margin` آن برابر با `auto` شده است

مشکلی که در این حالت به وجود می آید اینه که اگر سایز صفحه صفحه مرورگر کمتر از مقدار در نظر گرفته شده برای `div` بالا باشه ، مرورگر یک اسکرول به صفحه اضافه می کنه. برای حل این مشکل از دستور `max-width` استفاده می کنیم. با استفاده از این دستور در صورتی که اندازه صفحه مرورگر کمتر از اندازه `div` باشه اسکرول ایجاد نمی شه و کل متن داخل `div` یک جا نمایش داده میشه.

این یک المنت `div` با مقدار خصوصیت `max-width` آن برابر `500px` است و `margin` آن برابر با `auto` شده است

خب حالا توی مثال زیر مبحث بالا آورده شده است که می تونین به راحتی تست کنین



مثال

```
<html>
<head>
<style>
div.ex1 {
  width:500px;
  margin: auto;
  border: 3px solid #73AD21;
}
div.ex2 {
  max-width:500px;
  margin: auto;
  border: 3px solid #73AD21;
}
</style>
</head>
<body>
<div class="ex1">برای این المنت در نظر گرفته شده است width: 500px;</div>
<br>
<div class="ex2">برای این المنت در نظر گرفته شده است max-width: 500px;</div>
</body>
</html>
```



ساعت پنجم



Position

از خصوصیت **Position** برای موقعیت دهی به یک المنت استفاده می کنیم که می تونه این موقعیت ایستا ، وابسته ، ثابت یا مطلق باشه.

با توجه به توضیح بالا پس ما 4 نوع موقعیت دهی برای المنت داریم که شامل:

static
relative
fixed
absolute

المنت ها با استفاده از خصوصیت های بالا، پایین، سمت چپ و راست موقعیت دهی می شوند اما این خواص کار نمی کنند مگر اینکه خصوصیت **position** برای اولین بار تنظیم شود. آنها نیز بسته به مقدار **position**، متفاوت عمل می کنند.

position: static;

المنت های **html** به صورت پیش فرض با این مقدار موقعیت دهی شده اند، این المنت ها که به صورت ایستا موقعیت دهی شده اند توسط خواص **top, left, right, bottom** تحت تاثیر قرار نمی گیرند. المنتی که با خصوصیت **position: static** مقدار دهی شوند نمی توانند در موقعیت های متفاوتی قرار بگیرند بلکه فقط می توانند در موقعیت معمول صفحه قرار بگیرند. به مثال زیر توجه کنید

مثال

```
<html>
<head>
<style>
div.static {
  position: static;
  border: 3px solid #73AD21;
}
</style>
</head>
<body>
<h2>position: static;</h2>
<p>المنتی که با خصوصیت position: static مقدار دهی شوند نمی توانند در موقعیت های متفاوتی قرار بگیرند بلکه فقط می توانند در موقعیت معمول صفحه قرار بگیرند.</p>
<div class="static">
المنتی که با خصوصیت position: static برای این المنت در نظر گرفته شده است
</div>
</body>
</html>
```



position: relative

عنصر با **position: relative** باعث می شود که المنت مطابق با موقعیت فعلی آن موقعیت دهی شود. تنظیم خصوصیت های **top, left, right, bottom** باعث میشه که المنتی که با خصوصیت **position: relative** مقدار دهی شده از موقعیت معمولی خودش تغییر کنه و سایر المنت ها کاری به تغییرات این المنت ندارند و آنها ثابت هستند.

مثال زیر را انجام دهید و مقادیر مختلفی برای خصوصیت های **left, top, right** امتحان کنید و نتایج را مشاهده کنید تا با این مفهوم به خوبی آشنا بشوید.

```
<html>
<head>
<style>
div.relative {
  position: relative;
  left: 30px;
  border: 3px solid #73AD21;
}
</style>
</head>
<body>
<h2>position: relative;</h2>
<p></p>:المنتی که با این خصوصیت مقدار دهی بشه، مطابق با موقعیت فعلیش قابل تغییر هست.
<div class="relative">
position: relative;
</div>
</body>
</html>
```

مثال

position: fixed

المنتی که با خصوصیت **position: fixed** مقدار دهی شده باشد وابسته به صفحه نمایش می شود به این معنی که همیشه در یک مکان ثابت قرار می گیرد و حتی اگه صفحه اسکرول بشه باز هم مکان این المنت تغییر نمی کنه. خصوصیت های **top, left, right, bottom** برای موقعیت دهی به المنت همراه با این خصوصیت استفاده می شوند.



در مثال زیر ما المنت **div** را خصوصیت دهی کرده ایم که در پایین صفحه سمت راست نمایش داده شود.

مثال

```
<html>
<head>
<style>
div.fixed {
  position: fixed;
  bottom: 0;
  right: 0;
  width: 300px;
  border: 3px solid #73AD21;
}
</style>
</head>
<body>
<h2>position: fixed;</h2>
<div class="fixed">
This div element has position: fixed;
</div>
</body>
</html>
```

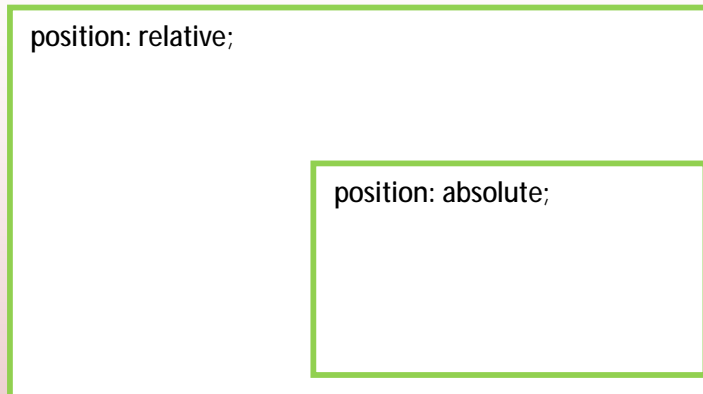
position: absolute

یک المنت با خصوصیت **position: absolute** موقعیتش وابسته به موقعیت المنت خارجی اش می شود. یعنی آگه دو تا المنت داشته باشیم که یکی در داخل دیگری قرار داشته باشد، موقعیت المنت داخلی وابسته به المنت خارجی می شود. مانند خصوصیت **fixed** که وابسته به صفحه نمایش بود، با این تفاوت که این المنت وابسته به المنت پدر می شود. اگر المنت هیچ پدری نداشته باشد مطابق با المنت **body** موقعیت دهی می شود. در قسمت زیر یک مثال ارائه شده که به خوبی این خصوصیت را توضیح می دهد:



مثال

```
<html>
<head>
<style>
div.relative {
  position: relative;
  width: 400px;
  height: 200px;
  border: 3px solid #73AD21;
}
div.absolute {
  position: absolute;
  top: 80px;
  right: 0;
  width: 200px;
  height: 100px;
  border: 3px solid #73AD21;
}
</style>
</head>
<body>
<h2>position: absolute;</h2>
<div class="relative"> position: relative;
  <div class="absolute"> position: absolute;</div>
</div>
</body>
</html>
```



Overlapping Elements

وقتی که المنت ها را موقعیت دهی می کنیم ممکن است بعضی از المنت ها روی یکدیگر قرار بگیرند که با استفاده از خصوصیت **z-index** می توانیم چیدمان این المنت ها را مدیریت کنیم. یک المنت می تواند یک پشته مثبت یا منفی داشته باشد. به مثال زیر توجه کنید



مثال

```
<head>
<style>
img {
  position: absolute;
  left: 0px;
  top: 0px;
  z-index: -1;
}
</style>
</head>
<body>
<h1>This is a heading</h1>

<p>چون تصویر مقدار 1- گرفته است زیر متن قرار می گیرد.</p>
</body>
</html>
```



یک المنت با مقدار بیشتر همیشه روی المنتی با مقدار کمتر قرار می گیرد. اگه دو المنت خاصیت **z-index** نداشته باشند، المنتی که در آخر تعریف شده باشد همیشه روی سایر المنت ها قرار می گیرد.

Overflow

با استفاده از خصوصیت **Overflow** می توانیم محتویات یک بلوک را در صورتی که اندازه محتویات زیاد باشد اسکرول بدهیم.

خصوصیت **Overflow** مقدارهای زیر را می تواند بگیرد:

- **visible** حالت پیش فرضه و باعث میشه که محتویات بیشتر از بلوک از اون خارج بشه
- **Hidden** محتویات بیشتر از بلوک نمایش داده نمی شود
- **Scroll** محتویات بلوک قابل اسکرول می شود
- **Auto** اگر محتویات بلوک بیشتر از بلوک باشد ، بلوک باید اسکرول شود.



overflow: visible

این حالت پیش فرضه و در این حالت اطلاعاتی که بیشتر از بلوک هستند نمایش داده می شوند اما به صورت خارج از بلوک به مثال زیر توجه کنید

```
<html>
<head>
<style>
div {
  background-color: #eee;
  width: 200px;
  height: 50px;
  border: 1px dotted black;
  overflow: visible;
}
</style>
</head>
<body>
<div> این حالت پیش فرضه و در این حالت اطلاعاتی که بیشتر از بلوک هستند نمایش داده می شوند اما به صورت خارج از بلوک
</div>
</body>
</html>
```

مثال

این حالت پیش فرضه و در این حالت اطلاعاتی که بیشتر از بلوک هستند نمایش داده می شوند اما به صورت خارج از بلوک

overflow: hidden

در این حالت مقدار محتوایی که بیشتر از بلوک باشد مخفی می شود. به مثال زیر توجه کنید

```
<html>
<head>
<style>
div {
  background-color: #eee;
  width: 200px;
  height: 50px;
  border: 1px dotted black;
  overflow: hidden;
}
</style>
</head>
<body>
<div> در این حالت اطلاعاتی که بیشتر از بلوک هستند نمایش داده نمی شوند و کاربر نمی تواند اطلاعاتی که در بلوک جای نمی گیرند را
</div> مشاهده کند
</body>
</html>
```

مثال

در این حالت اطلاعاتی که بیشتر از بلوک هستند نمایش داده نمی شوند و

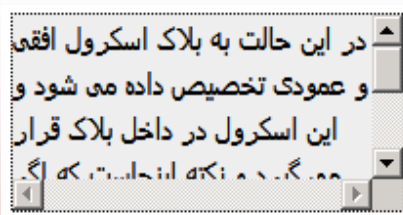


overflow: scroll

در این حالت به بلاک اسکرول افقی و عمودی تخصیص داده می شود و این اسکرول در داخل بلاک قرار می گیرد و نکته اینجاست که اگر متن نیاز به اسکرول هم نداشته باشد باز هم اسکرول نمایش داده می شود. به مثال زیر توجه کنید:

مثال

```
<html>
<head>
<style>
div {
  background-color: #eee;
  width: 200px;
  height: 50px;
  border: 1px dotted black;
  overflow: scroll;
}
</style>
</head>
<body>
<div> در این حالت به بلاک اسکرول افقی و عمودی تخصیص داده می شود و این اسکرول در داخل بلاک قرار می گیرد و نکته اینجاست که اگر متن نیاز به اسکرول هم نداشته باشد </div>
</body>
</html>
```



overflow: auto

این مقدار هم تقریباً شبیه به حالت scroll هستش ولی با این تفاوت که اسکرول وقتی اضافه می شه که نیاز هست

مثال

```
<html>
<head>
<style>
div {
  background-color: #eee;
  width: 200px;
  height: 50px;
  border: 1px dotted black;
  overflow: auto;
}
</style>
</head>
<body>
<div> در این حالت اسکرول زمانی به بلوک اضافه می شود که نیاز باشد . یعنی اطلاعات اگر فقط در سطح عمودی بیشتر از بلاک باشند دیگر اسکرول افقی </div>
</body>
</html>
```



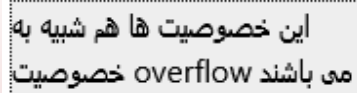


overflow-x and overflow-y

این خصوصیت ها هم شبیه به خصوصیت `overflow` می باشند با این تفاوت که `overflow-x` فقط در محور افقی اعمال می شود و `overflow-y` فقط در محور عمودی اعمال می شود. به مثال زیر توجه کنید:

مثال

```
<html>
<head>
<style>
div {
  background-color: #eee;
  width: 200px;
  height: 50px;
  border: 1px dotted black;
  overflow-x: hidden;
  overflow-y: scroll;
}
```



این خصوصیت ها هم شبیه به
می باشند overflow خصوصیت

```
</style>
</head>
<body>
```

```
<div> فقط در محور افقی اعمال overflow-x می باشند با این تفاوت که overflow این خصوصیت ها هم شبیه به خصوصیت  
در div فقط در محور عمودی اعمال می شود overflow-y می شود و  
</body>
```




Float

float and clear

خصوصیت **float** برای المنت هایی استفاده می شود که قرار است شناور باشند
خصوصیت **clear** برای المنت هایی استفاده می شود که قبلا شناور شده اند و باید از شناوری خارج شوند.

یک مثال خیلی ساده برای **float** این است که بخواهیم متن را دور تا دور تصویر قرار دهیم. در مثال زیر یک تصویر را در سمت راست متن قرار داده ایم. به مثال توجه کنید

```
<html>
<head>
<style>
img {
  float: right;
  margin: 0 0 10px 10px;
}
</style>
</head>
<body>
```

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



مثال

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

</body>
</html>
```



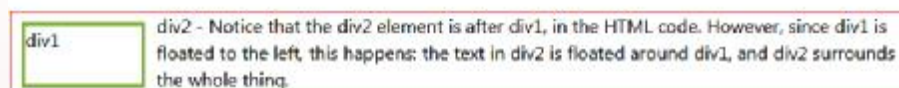
خصوصیت clear

از این خصوصیت برای کنترل رفتار **float** بر روی المنت استفاده می شود. المنت هایی که بعد از المنتی که خصوصیت **float** روی آن اعمال شده قرار می گیرند تحت تاثیر قرار می گیرند. خصوصیت **clear** باعث می شود که این المنت ها **float** نشوند. به مثال زیر توجه کنید:

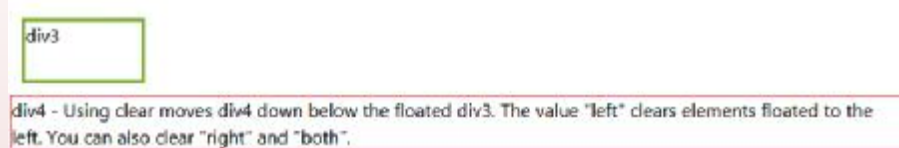
مثال

```
<html>
<head>
<style>
.div1 {
  float: left;
  width: 100px;
  height: 50px;
  margin: 10px;
  border: 3px solid #73AD21;
}
.div2 {
  border: 1px solid red;
}
.div3 {
  float: left;
  width: 100px;
  height: 50px;
  margin: 10px;
  border: 3px solid #73AD21;
}
.div4 {
  border: 1px solid red;
  clear: left;
}
</style>
</head>
<body>
<h2>بدون clear</h2>
<div class="div1">div1</div>
<div class="div2">div2 - Notice that the div2 element is after div1, in the HTML code. However, since div1 is floated to the left, this happens: the text in div2 is floated around div1, and div2 surrounds the whole thing.</div>
<h2>با clear</h2>
<div class="div3">div3</div>
<div class="div4">div4 - Using clear moves div4 down below the floated div3. The value "left" clears elements floated to the left. You can also clear "right" and "both".</div>
</body>
</html>
```

بدون clear



با clear





اگر یک المنتی از المنت پدر خود بزرگتر باشد و **float** باشد. در این حالت از فضای المنت پدر خارج می شود. در این صورت برای حل مشکل کافی است که المنت پدر خصوصیت **overflow: auto** را اعمال کنیم. با این روش المنت پدر به اندازه المنت فرزند می شود. به مثال زیر توجه کنید.

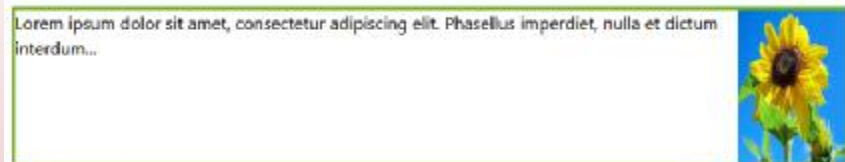
مثال

```
<html>
<head>
<style>
div {
  border: 3px solid #73AD21;
}
.img1 {
  float: right;
}
.clearfix {
  overflow: auto;
}
.img2 {
  float: right;
}
</style>
</head>
<body>
<p>In this example, the image is taller than the element containing it, and it is floated, so it overflows outside of its container:</p>
<div>
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum...</div>
<p style="clear:right">Add a clearfix class with overflow: auto; to the containing element, to fix this problem:</p>
<div class="clearfix">
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum...</div>
</body>
</html>
```

In this example, the image is taller than the element containing it, and it is floated, so it overflows outside of its container:



Add a clearfix class with overflow: auto; to the containing element, to fix this problem:





Inline-block

ما می‌توانیم چندین ردیف و ستون از باکس‌ها را در صفحه وب ایجاد کنیم و به آنها عرض و ارتفاع بدهیم و با استفاده از دستور **float** آنها را کنار یکدیگر قرار بدهیم. به صورتی که اگر صفحه وب تغییر اندازه داده بشه ترتیب این باکس‌ها بهم نریزه و صفحه وب ما شکل باقی بمونه.

با استفاده از مقدار **inline-block** برای خصوصیت **display** این کار رو راحت‌تر انجام بدهیم.

المنتی که **Inline-block** می‌گیره شبیه به المنتی است که به صورت **inline** هستش با این تفاوت که می‌تونیم به المنت عرض و ارتفاع بدهیم.

در زیر یک مثال ارائه شده که در یک صفحه وب تعدادی باکس با استفاده از دستور **float** کنار هم قرار گرفته‌اند.

مثال

```
<html>
<head>
<style>
.floating-box {
  float: left;
  width: 150px;
  height: 75px;
  margin: 10px;
  border: 3px solid #73AD21;
}
.after-box {
  clear: left;
  border: 3px solid red;
}
</style>
</head>
<body>
<h2>The Old Way - using float</h2>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="after-box">Another box, after the floating boxes...</div>
</body>
</html>
```

The Old Way - using float





خب حالا توی مثال زیر مثال بالا رو با استفاده از `inline-block` طراحی می کنیم. البته این نکته رو هم مد نظر داشته باشید که نیازی به خصوصیت `clear` وجود نداره دیگه.

مثال

```
<html>
<head>
<style>
.floating-box {
  display: inline-block;
  width: 150px;
  height: 75px;
  margin: 10px;
  border: 3px solid #73AD21;
}
.after-box {
  border: 3px solid red;
}
</style>
</head>
<body>
<h2>The New Way - using inline-block</h2>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="floating-box">Floating box</div>
<div class="after-box">Another box, after the floating boxes...</div>
</body>
</html>
```

The Old Way - using float





Align

وسط چین کردن المنت ها

برای وسط چین کردن یک المنت مانند **div** می توانید از **margin:auto** استفاده کنید. تنظیم عرض المنت باعث میشه که کشیدگی لبه های المنت از ابتدا و انتهای المنت به صورتی باشه که المنت وسط چین بودن خودشو حفظ کنه. در زیر یک مثال ارائه شده که این مفهوم رو به خوبی توضیح می ده

مثال

```
<html>
<head>
<style>
.center {
  margin: auto;
  width: 60%;
  border: 3px solid #73AD21;
  padding: 10px;
}
</style>
</head>
<body>

<h2>Center Align Elements</h2>

<div class="center">
این یک المنتی هست که وسط چین شده است
</div>

</body>
</html>
```

Center Align Elements

این یک المنتی هست که وسط چین شده است

توجه داشته باشید که اگه شما خصوصیت **width** رو برای المنت تنظیم نکنین این وسط چین کار نمی کنه. نکته دیگه اینکه **margin:auto** روی مرورگر اینترنت اکسپلورر 8 کار نمی کنه.



وسط چین کردن متن در درون المنت

برای اینکه یک متن رو که داخل المنت قرار داره رو وسط چین کنیم از دستور **text-align** استفاده می کنیم. در مثال زیر ما متن داخل یک المنت رو وسط چین کردیم. به مثال توجه کنید

مثال

```
<html>
<head>
<style>
.center {
  text-align: center;
  border: 3px solid green;
}
</style>
</head>
<body>

<h2>Center Text</h2>

<div class="center">
  <p>This text is centered.</p>
</div>

</body>
</html>
```

Center Text

This text is centered.

وسط چین کردن یک تصویر

برای اینکه یک تصویر را وسط چین کنیم کافی است که خصوصیت **display** آنرا برابر با **block** کنیم و از دستور **margin:auto** استفاده کنید. به مثال زیر توجه کنید:



مثال

```
<html>
<head>
<style>
img {
  display: block;
  margin: 0 auto;
}
</style>
</head>
<body>
<h2>Center an Image</h2>

</body>
</html>
```

Center an Image



چپ چین و راست چین – با استفاده از خصوصیت position

یک روش دیگر برای چیدمان کردن المنت ها استفاده از خصوصیت position با مقدار absolute است.

مثال

```
<html>
<head>
<style>
.right {
  position: absolute;
  right: 0px;
  width: 300px;
  border: 3px solid #73AD21;
  padding: 10px;
}
</style>
</head>
<body>
<h2>Right Align</h2>
<div class="right">
  <p>این یک متن هست که داخل المنت قرار گرفته است و این المنت راست چین شده است.</p>
</div>
</body>
</html>
```

Right Align

این یک متن هست که داخل المنت قرار گرفته است و این المنت راست چین شده است.



المنتی که خصوصیت **position** آن برابر با **absolute** شده است از حالت نرمال خارج می شود و می تواند روی سایر المنت ها قرار بگیرد.

وقتی که از خصوصیت **position** استفاده می کنید حتما برای المنت **<body>** خصوصیت **margin** و **padding** رو تعریف کنید. این باعث میشه که صفحه وب شما روی سایر مرورگرها درست نمایش داده بشه. به مثال زیر توجه کنید:

مثال

```
<html>
<head>
<style>
body {
  margin: 0;
  padding: 0;
}
.container {
  position: relative;
  width: 100%;
}
.right {
  position: absolute;
  right: 0px;
  width: 300px;
  background-color: #b0e0e6;
}
</style>
</head>
<body>
<h2>Right Align</h2>
<div class="container">
  <div class="right">
    <p><b>Note: </b>When aligning using the position property, always include the !DOCTYPE declaration! If missing, it can produce strange results in IE browsers.</p>
  </div>
</div>
</body>
</html>
```

Right Align

Note: When aligning using the position property, always include the !DOCTYPE declaration! If missing, it can produce strange results in IE browsers.



چپ چین و راست چین با استفاده از float

روش دیگری که برای چیدمان المنت ها وجود دارد استفاده از خصوصیت float است. به مثال زیر توجه کنید

مثال

```
<html>
<head>
<style>
.right {
  float: right;
  width: 300px;
  border: 3px solid #73AD21;
  padding: 10px;
}
</style>
</head>
<body>

<h2>Right Align</h2>

<div class="right">
  <p>In my younger and more vulnerable years my father gave me some advice that I've been turning over in my mind ever since.</p>
</div>

</body>
</html>
```

Right Align

In my younger and more vulnerable years my father gave me some advice that I've been turning over in my mind ever since.

وقتی که از خصوصیت float استفاده می کنید حتما برای المنت <body> خصوصیت padding و margin رو تعریف کنید. این باعث میشه که صفحه وب شما روی سایر مرورگرها درست نمایش داده بشه. به مثال زیر توجه کنید:



مثال

```
<html>
<head>
<style>
body {
  margin: 0;
  padding: 0;
}
.right {
  float: right;
  width: 300px;
  background-color: #b0e0e6;
}
</style>
</head>
<body>
<h2>Right Align</h2>
<div class="right">
  <p><b>Note: </b>When aligning using the float property, always include the !DOCTYPE declaration! If missing,
  it can produce strange results in IE browsers.</p>
</div>
</body>
</html>
```

Right Align

Note: When aligning using the float property, always include the !DOCTYPE declaration! If missing, it can produce strange results in IE browsers.

عمودی وسط چین کردن با استفاده از padding

روش های زیادی وجود دارد برای اینکه بتونیم یک المنت رو در CSS به صورت عمودی وسط چین کنیم ، یک روش خیلی ساده استفاده از padding هستش. به مثال زیر توجه کنید:



مثال

```
<html>
<head>
<style>
.center {
  padding: 70px 0;
  border: 3px solid green;
}
</style>
</head>
<body>
<h2>Center Vertically</h2>
<div class="center">
  <p>I am vertically centered.</p>
</div>
</body>
</html>
```

Center Vertically

I am vertically centered.

برای اینکه بتوانیم هم به صورت عمودی و هم به صورت افقی وسط چین کنیم کافیست که از خصوصیت **text-align** با مقدار **center** در کنار **padding** استفاده کنیم. به مثال زیر توجه کنید:

مثال

```
<html>
<head>
<style>
.center {
  padding: 70px 0;
  border: 3px solid green;
  text-align: center;
}
</style>
</head>
<body>
<h2>Centering</h2>
<div class="center">
  <p>I am vertically and horizontally centered.</p>
</div>
</body>
</html>
```

Centering

I am vertically and horizontally centered.



عمودی وسط چین کردن با استفاده از **line-height**

ترفند دیگری ای که برای وسط چین کردن به صورت عمودی وجود دارد استفاده از خصوصیت **line-height** هستش که توی مثال زیر توضیح داده شده :

مثال

```
<html>
<head>
<style>
.center {
  line-height: 200px;
  height: 200px;
  border: 3px solid green;
  text-align: center;
}
.center p {
  line-height: 1.5;
  display: inline-block;
  vertical-align: middle;
}
</style>
</head>
<body>
<h2>Centering</h2>
<div class="center">
  <p>I am vertically and horizontally centered.</p>
</div>
</body>
</html>
```

عمودی وسط چین کردن با استفاده از **transform** و **position**

اگه دو روش قبلی رو نخواستین حالا به هر دلیلی استفاده کنید می تونید از این روش استفاده کنید. در این روش می توانید از خصوصیت **transform** استفاده کنید. به مثال زیر توجه کنید



مثال

```
<html>
<head>
<style>
.center {
  height: 200px;
  position: relative;
  border: 3px solid green;
}
.center p {
  margin: 0;
  position: absolute;
  top: 50%;
  left: 50%;
  -ms-transform: translate(-50%, -50%);
  transform: translate(-50%, -50%);
}
</style>
</head>
<body>
<h2>Centering</h2>
<div class="center">
  <p>I am vertically and horizontally centered.</p>
</div>
</body>
</html>
```

در مورد خصوصیت transform در درس CSS3 توضیح کامل تری خواهیم داد.



Combinators