

2 CSS3의 기초

1) 스타일을 만드는 기본 CSS

CSS(Cascading Style Sheet)는 스타일 시트(Style Sheet)의 나열(Cascading)을 의미합니다. CSS는 HTML 요소를 보기 좋게 디자인하는 역할을 담당합니다.

HTML 요소만 가지고는 스타일을 만들 수 없습니다. HTML 요소는 단순히 문서의 내용을 정의하는 역할을 담당합니다. HTML 요소를 디자인할 때는 CSS를 이용해야 합니다. 대부분 CSS 작업은 외부 스타일 시트(External style sheet)인 CSS 파일(**.css)로 저장하여 사용합니다.

이전에 사용되었던 font 요소와 같은 경우에는, 폰트에 대한 정보와 컬러 정보를 전달하기 위해 불필요하게 추가되는 코드가 많았습니다.

```
<p><font size="3" color="red">text</font></p>
<p><font size="2" color="blue">text</font></p>
<p><font face="verdana" color="green">text</font></p>
```

위의 표와 같이 HTML의 내용이 복잡해지는 문제를 해결하기 위해 W3C(World Wide Web Consortium)에서는 CSS를 고안해 내었습니다. HTML 4.0에서부터는 HTML 문서의 스타일을 위해서 문서로부터 별도의 CSS로 분리했습니다.

스타일 정의는 일반적으로 외부 스타일 파일로 저장합니다. 외부 스타일을 사용하면 하나의 파일을 변경하여 전체 웹 페이지의 모양을 변경할 수 있습니다.

그리고 CSS3은 CSS의 가장 최신 기준입니다.

1-1) 기초적인 CSS 작성

1-1-1) Chrome 개발자 도구(Chrome Developer Tools)

CSS를 작성하기에 앞서서 실습할 브라우저를 선택해 봅시다. 일반적으로 웹 퍼블리셔들이나 프론트엔드 개발자들은 Chrome 브라우저를 선호합니다. Chrome 브라우저는 최신 웹에 최적화된 빠르고 안전한 무료 웹 브라우저입니다.

아래의 URL에서 Chrome 브라우저를 설치할 수 있습니다.

참고 URL

<https://www.google.co.kr/chrome/browser/desktop/>

Chrome 개발자 도구(Chrome Developer Tools)는 Google Chrome 브라우저에 내장된 웹 제작 및 디버깅을 위한 도구로서, 웹을 작업하는 사람들에게 매우 유용한 기능을 제공합니다. 또한 레이아웃에 대한 관련 CSS와 JavaScript에 대한 디버깅 기능을 포함하여 코드 최적화에 기능을 제공합니다.

참고 URL


<https://developer.chrome.com/devtools>

debugging(디버깅)

디버그(debug), 디버깅(debugging) 혹은 수정은 컴퓨터 프로그램의 정확성이나 논리적인 오류(버그)를 찾아내는 테스트 과정을 뜻한다. 디버깅(debugging), 수정이라고도 한다.

출처 : <https://ko.wikipedia.org/wiki/디버그>

Chrome 개발자 도구를 여는 두 가지 방법이 있습니다.

첫 번째 방법은 Chrome 브라우저 창의 오른쪽 윗부분에 있는 Chrome 메뉴 버튼()을 클릭하여 표시되는 메뉴에서 [도구 더보기] - [개발자 도구(D)]를 선택합니다.

두 번째 방법은 브라우저에 표시된 페이지 요소를 마우스 오른쪽 버튼으로 클릭하고 나타나는 단축 메뉴에서 [검사(N)] 메뉴를 선택합니다. 둘 중 하나의 방식으로 Chrome 개발자 도구를 확인할 수 있을 것입니다.

Chrome 개발자 도구는 도구 모음에서 작업 영역이 나뉘게 됩니다.



각 도구 모음 항목과 해당 패널을 이용하면 페이지에 대한 정보와 DOM 요소에 대한 정보를 쉽게 알 수 있습니다. 또한 리소스 및 소스를 포함하는 응용 프로그램에 대한 정보를 얻을 수 있을 뿐만 아니라, 간단하게나마 소스를 수정해서 확인할 수도 있습니다. 예를 들어 간단하게나마 Chrome 브라우저 창에서 직접 스타일을 변경할 수 있습니다.

아래의 내용은 Chrome 브라우저를 통해서 예제를 확인하는 방법입니다.

Console 작업(Working with the Console) :

JavaScript Console은 웹 페이지와 응용 프로그램에 대하여 유용한 기능을 제공합니다. Console API에서 제공하는 `console.log()`나 `console.profile()`과 같은 방법을 사용하여 로그 정보를 기록할 수 있습니다.

JavaScript 디버깅(Debugging JavaScript) :

JavaScript 애플리케이션이 늘어남에 따라 웹 개발자는 JavaScript 애플리케이션에서 발생하는 문제의 원인을 신속하게 발견하고 효율적으로 해결하기 위해 강력한 디버깅 도구가 필요합니다. Chrome 개발자 도구는 JavaScript 애플리케이션을 디버깅하는 간단한 방법을 제공합니다.

네트워크 성능 향상(Improving network performance) :

Network 패널은 요청과 실시간으로 네트워크를 통해 다운로드된 자원에 대한 자료를 제공합니다. 예상보다 오래 걸리는 요청을 해결하는 방법으로 페이지 최적화할 수 있습니다.

1-1-2) CSS 내용을 통한 스타일 구분

tag :

가장 기본적인 HTML 태그에 대한 스타일을 정해줍니다.

```
div {  
  width: 800px;  
}
```

class :

클래스 요소, 즉 점(dot)으로 시작되는 스타일을 정해줍니다.

```
.container {  
  width: 800px;  
}
```

id :

id 요소, 즉 '#'으로 시작되는 스타일을 정해줍니다.

```
#user {  
  width: 800px;  
}
```

1-1-3) CSS 형식을 통한 스타일 구분

참고 URL

http://www.w3schools.com/css/css_howto.asp

CSS를 HTML 문서에 적용하는 3가지 방법은 다음과 같습니다. :

- 외부 스타일 시트(External Style Sheet)
- 내부 스타일 시트(Internal Style Sheet)
- 인라인 스타일(Inline Style)

각각의 스타일의 방법을 어떻게 사용하는지 살펴보기로 합니다.

외부 스타일 시트(External Style Sheet) :

외부 스타일 시트를 사용하면 하나의 CSS 파일을 변경하여 전체 웹 페이지의 모양을 변경할 수 있습니다. 외부 스타일 시트를 사용하는 각각의 HTML 페이지에서는 다음과 같이 link 요소를 사용하여 외부 스타일 시트 파일에 대한 참조를 포함해야 합니다.(rel="stylesheet") 참고로 link 요소는 HTML 문서의 head 영역에 포함됩니다.

```
<head>
<link rel="stylesheet" href="style.css">
</head>
```

내부 스타일 시트(Internal Style Sheet) :

하나의 페이지에만 적용해야 하는 고유한 스타일이 필요한 경우, 내부 스타일 시트가 사용될 수 있습니다. 내부 스타일은 HTML 문서에서 head 영역에서 style 요소 내에 정의됩니다.

```
<style>
body {
    background-color: #f1f1f1;
}
h1 {
    margin-left: 40px;
    color: #555;
}
</style>
```

인라인 스타일(Inline Style) :

인라인 스타일은 단일 요소에 고유한 스타일을 적용하기 위해 사용할 수 있습니다. 인라인 스타일이 많이 사용된 HTML 문서는 CSS가 갖는 큰 장점인 통합적인 관리가 되지 않으므로, 될 수 있으면 사용하지 않는 것이 좋습니다.

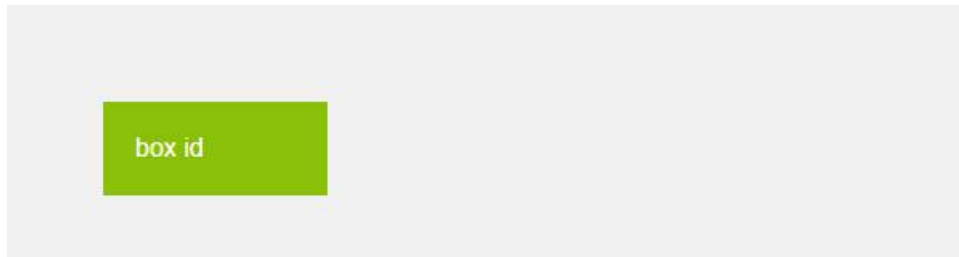
인라인 스타일을 사용하려면, 해당 요소에 스타일 속성을 추가하여 사용합니다.

```
<h1 style="margin-left:40px; color:#555;">타이틀입니다.</h1>
```

이제 좀 그럴 듯한 HTML 문서를 만들어 보겠습니다.

시작 파일

sample/basic_css/basic_css_start1.html



HTML

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
6 <meta name="viewport" content="width=device-width, initial-scale=1.0">
7 <title>basic :: CSS Start</title>
8 <style type="text/css">
9 body {
10     margin: 20px;
11     padding: 20px;
12     font-family: Arial, Helvetica, sans-serif;
13     font-size: 1em;
14     background-color: #f1f1f1;
15     color: #555;
16 }
17 .container {
18     width: 800px;
19 }
20 #box {
21     margin: 20px;
22     padding: 20px;
23     width: 100px;
24     background-color: #8ac007;
25     color: #fff;
26 }
27 </style>
28 </head>
```

```

29
30 <body>
31 <div class="container">
32     <div id="box">box id</div>
33 </div>
34 </body>
35 </html>

```

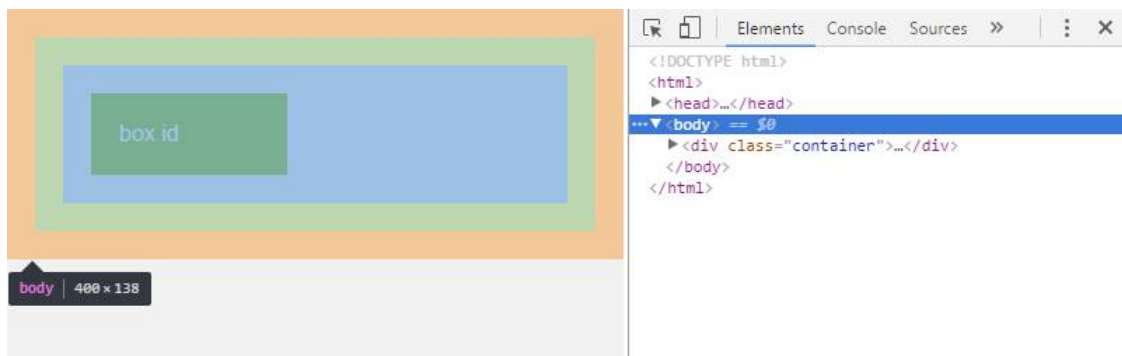
현재 문서에 적용된 스타일은 head 부분에 적용된 내부 스타일 시트입니다. 하지만 이 교재에서는 대부분의 예제에서는 외부 스타일 시트를 사용하여 설명합니다.

9행 : body {

body 요소에 대한 스타일 선언입니다. 가장 빈번하게 등장하는 속성은 margin 속성과 padding 속성일 것입니다. margin은 외부 여백을 의미하며, padding은 내부 여백을 의미합니다.

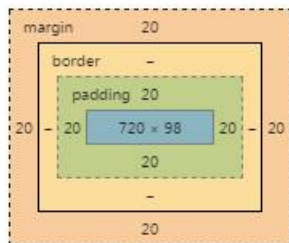
Chrome 디버깅 창을 활용해서 margin과 padding 속성을 알아봅시다.

[Select an element in the page to inspect it] 버튼을 클릭하고 body 요소를 클릭합니다.



[Style] 탭에 관련된 속성이 진열됩니다.

```
Styles Computed Event Listeners >>
Filter :hov .cls +
element.style {
}
body { basic css start1.html:9
margin: 20px;
padding: 20px;
font-family: Arial, Helvetica, sans-serif;
font-size: 1em;
background-color: #f1f1f1;
color: #555;
}
body { user agent stylesheet
display: block;
margin: 8px;
}
```



[Computed] 탭을 누르면 관련된 스타일이 도식화된 그림으로 표현됩니다.



이제까지 body 요소에 대한 스타일을 확인해 보았습니다. body 요소 하위의 요소 스타일도 확인해 봅시다.

17행 : .container {

container 클래스에 대한 스타일 선언입니다.

클래스는 여러 번 중복 사용할 수 있는 요소입니다. width는 가로 크기를 정해주는 CSS 속성입니다.

20행 : #box {

box 아이디에 대한 스타일 선언입니다.

아이디는 클래스 요소와 달리 중복 사용될 수 없는 요소입니다.

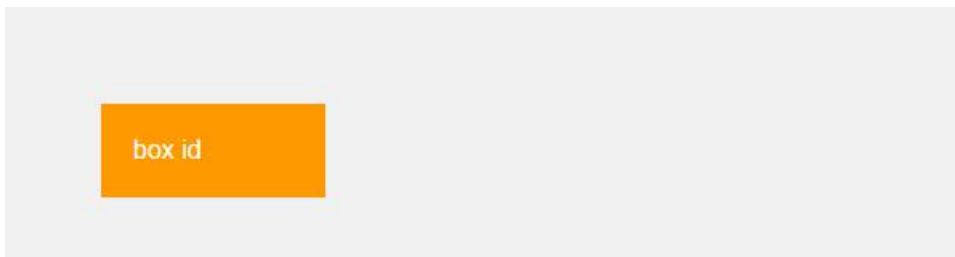
다중 스타일 시트(Multiple Style Sheet) :

다중 스타일 시트 방식으로 하나의 요소에 대해 스타일이 된 경우에는, 속성 값은 보다 구체적으로 지정된 스타일에 의해 정해집니다. 예를 들어 외부 스타일보다는 내부 스타일로 지정된 스타일이 우선 적용되며, 내부 스타일보다는 인라인 스타일이 우선 적용됩니다.

다음 예제는 다중 스타일을 적용하는 방법을 보여 줍니다.

시작 파일

sample/basic_css/basic_css_start2.html



HTML

```
1      <!DOCTYPE html>
2      <html>
3      <head>
4      <meta charset="utf-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>basic :: CSS Start</title>
8      <!-- 외부 스타일 시트 사용 -->
9      <link rel="stylesheet" href="css/style.css">
10     <!-- 내부 스타일 시트 사용 -->
11     <style>
12     #box {
13         background-color: #555;
14     }
15     </style>
16     </head>
17
18     <body>
19     <div class="container">
20         <!-- div 요소에 대한 인라인 스타일 시트 사용 -->
21         <div id="box" style="background-color:#f90;">box id</div>
22     </div>
```

```
23     </body>
24 </html>
```

다음과 같이 외부 스타일 시트로 style.css 파일을 연결합니다.

```
<link rel="stylesheet" href="css/style.css">
```

다음은 style.css 파일의 내용입니다.

```
#box {
    margin: 20px;
    padding: 20px;
    width: 100px;
    background-color: #8ac007;
    color: #fff;
}
```

외부 스타일 시트인 style.css 파일에서는 box id 요소의 바탕 색상을 정의하기 위해 background-color 속성의 값을 #8ac007로 하였습니다,

이번에는 내부 스타일 시트를 사용하여 box 요소의 바탕 색상을 #555로 정의합니다.

```
#box {
    background-color: #555;
}
```

마지막으로 인라인 스타일을 사용하여 box 요소의 바탕 색상을 #f90으로 정의합니다.

```
<div id="box" style="background-color:#f90;">box id</div>
```

브라우저를 통해 예제의 결과를 확인해 보면 box 요소의 바탕 색상은 오렌지 색상(#f90)으로 스타일이 정해집니다. 인라인 스타일은 head 요소에 정의된 내부 스타일 또는 외부 스타일보다 가장 우선 적용됩니다.

다음의 표는 같은 요소를 참조하였을 경우, 우선 적용되는 스타일입니다.

Inline Style	>	Internal Style Sheet	>	External Style Sheet
--------------	---	----------------------	---	----------------------

1-2) 기본적인 HTML 문서를 스타일하자!

1-2-1) 기본 HTML 템플릿의 제작

이제 본격적으로 본문 예제를 만들기 위한 기본 HTML 형식을 준비해 봅시다.

시작 파일

sample/basic_css/basic_sample_template.html

HTML

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
6  <meta name="viewport" content="width=device-width, initial-scale=1.0">
7  <title>basic :: basic Cascading Style Sheets</title>
8  <link rel="stylesheet"
  href="http://fonts.googleapis.com/css?family=Open+Sans:400,300,300italic,400italic,600,6
  00italic,700,700italic,800,800italic">
9  <style>
10 body {
11     margin: 20px;
12     padding: 20px;
13     line-height: 1;
14     font-family: "Open Sans", sans-serif;
15     font-size: 1em;
16     background-color: #f1f1f1;
17 }
18 .title {
19     margin: 0;
20     margin-bottom: 35px;
21     padding: 0;
22     font-size: 1.5em;
23     font-weight: 300;
24     color: #555;
25 }
26 </style>
27 </head>
28
29 <body>
```

```
30 <h1 class="title">basic :: basic Cascading Style Sheets</h1>
31 </body>
32 </html>
```

4행 : `<meta charset="utf-8">`

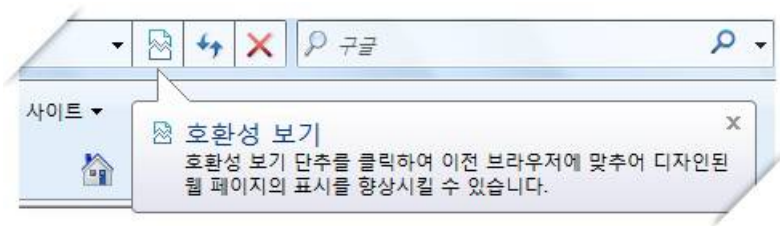
charset 속성은 HTML 문서의 문자 인코딩을 지정합니다.

5행 : `<meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">`

최신 모드로 지정된 DOCTYPE에 상관없이 Internet Explorer 8 이상 버전에서 항상 최신 표준 모드로 렌더링됩니다

호환성 보기

이전에 Internet Explorer 8 버전이 등장하면서 새로운 기능 한 가지를 추가했습니다.
바로 호환성 보기 버튼입니다.



호환성 보기 버튼은 Internet Explorer 8 버전에서 웹 페이지에 접근했을 때, 어떤 렌더링 엔진을 사용할 것인지를 선택하게 하는 용도의 버튼이었습니다.

Internet Explorer의 브라우저 버전이 올라가면서 해당 오류들을 웹 표준에 맞게 수정되었고 이에 따라 기존의 구형 브라우저와는 다르게 해석되는 부분이 발생하게 됩니다.

과거 Internet Explorer 6과 Internet Explorer 7이 브라우저 점유율의 대부분을 차지하고 있던 시기에 제작된 웹 페이지들은 이러한 오류들을 피하면서 제작되었기 때문에 현재의 웹표준과 다른 방식으로 렌더링이 됩니다. 이로 인해, 최신의 브라우저에서 웹표준을 지키지 않던 브라우저를 기준으로 제작된 웹 페이지를 방문하게 되면 레이아웃이 깨지거나 작동하지 않는 기능이 발생하게 됩니다. 따라서 아래와 같은 meta 정보를 추가하게 된 것입니다.

```
<meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
```

IE=edge 속성을 적용하면, 최신 모드로 지정된 DOCTYPE에 상관없이 IE8 이상 버전에서 항상 최신 표준 모드로 렌더링이 됩니다.

Microsoft는 실험적인 프로젝트가 아닌 이상 'IE=edge' 모드를 지정할 것을 권장합니다.

6행 : `<meta name="viewport" content="width=device-width, initial-scale=1.0">`

참고 URL

http://www.w3schools.com/css/css_rwd_viewport.asp

뷰포트(viewport)란 사용자가 보고 있는 웹 페이지의 영역입니다.

Tablet이나 Mobile이 등장하기 전에는, 웹 페이지는 컴퓨터 스크린을 위해서만 디자인되었습니다. 따라서 웹 페이지 구현에 있어서 정적인 디자인(static design)과 고정 크기(fixed size)를 가지는 웹 페이지가 일반적이었던 것이죠.

그러나 Tablet과 Mobile에서 보면 이전에 만들어진 고정된 웹 페이지는 자체 해상도가 너무 컸습니다. 이러한 디바이스에서의 브라우저 화면에 맞게 전체 웹 페이지를 축소하기 때문에 여러 가지 문제점이 발생되었습니다.

따라서 여러 가지 디바이스 뷰포트를 맞추기 위해서는 뭔가의 수정이 필요하게 되었습니다. HTML5는 웹 개발자들에게 `<meta>` 태그를 통해 뷰포트를 제어하는 방법을 제안하였습니다.

모든 웹 페이지에 기본적으로 아래와 같은 `<meta>` 태그를 삽입하는 방식입니다.

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

'width=device-width' 부분은 서로 다른 디바이스의 가로 크기에 따라 페이지의 폭을 설정합니다. width 값은 디바이스의 해상도가 아닌 웹 브라우저의 해상도임을 잊지 말아야 합니다.

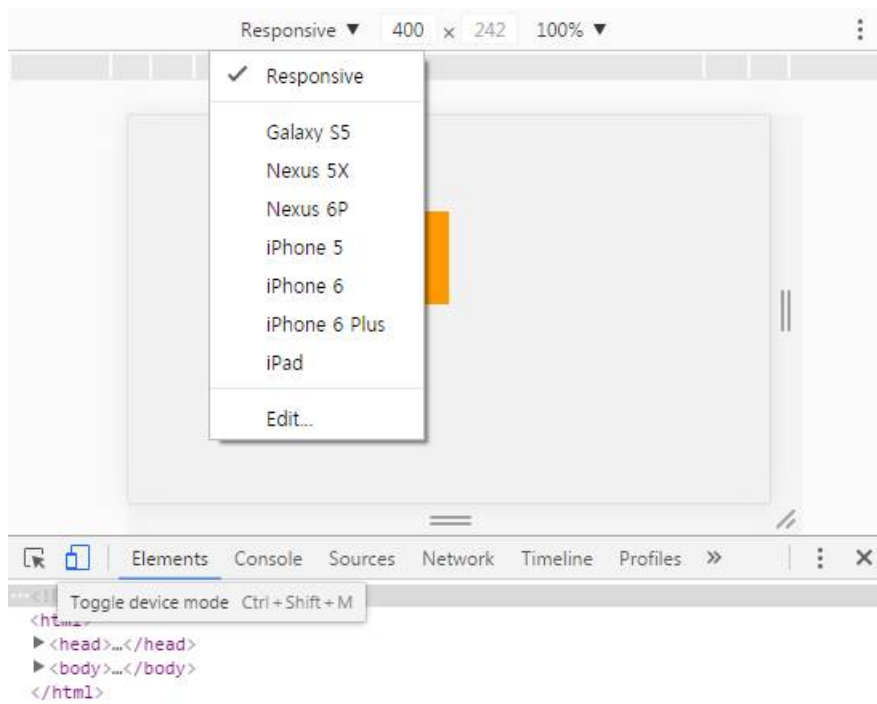
'initial-scale=1.0' 부분은 웹 페이지가 브라우저에 처음 로드되었을 경우 1:1 레벨로 설정합니다.

요약해 보면, 가로 값(width)을 디바이스의 기본 가로 값(device-width)으로 유지하고 비율은 1:1로 설정해 디바이스로 접속했을 때 디바이스의 웹 브라우저 해상도로 보이게 합니다.

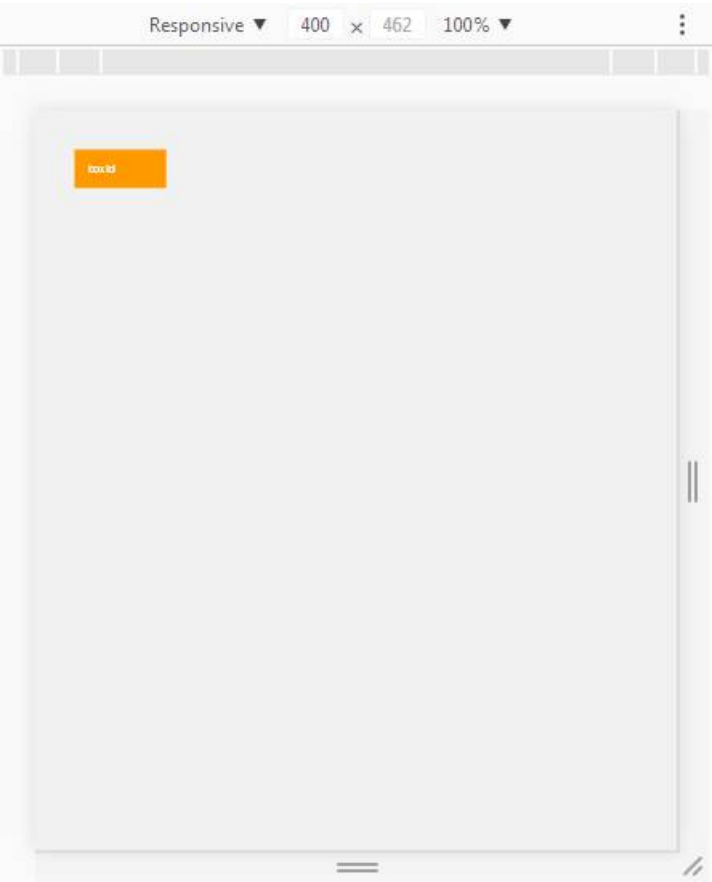
뷰포트를 사용하지 않는다면 스마트폰의 웹 브라우저는 웹 사이트를 자동으로 전체 화면에서 보여줍니다. 뷰포트를 설정해 정확히 스마트폰의 해상도로 보일 때의 모습으로 정확히 픽셀 크기만큼 보이게 됩니다. 대신 PC 사이트로 제작된 경우에는 스크롤 영역이 생기게 됩니다. 이런 스크롤 영역도 없애고 최적화된 사이트를 제작하려면 Mobile 웹을 따로 제작하거나 반응형 웹디자인을 이용하기도 합니다.

아래의 이미지는 위의 예제에 대해서 meta 요소가 없는 경우와 있는 경우의 웹 페이지의 예입니다. 테스트를 하는 방식은 Chrome 브라우저에서 [검사(N)] > [Toggle Device Mode]를 사용해서 가상으로 Mobile 테스트합니다.

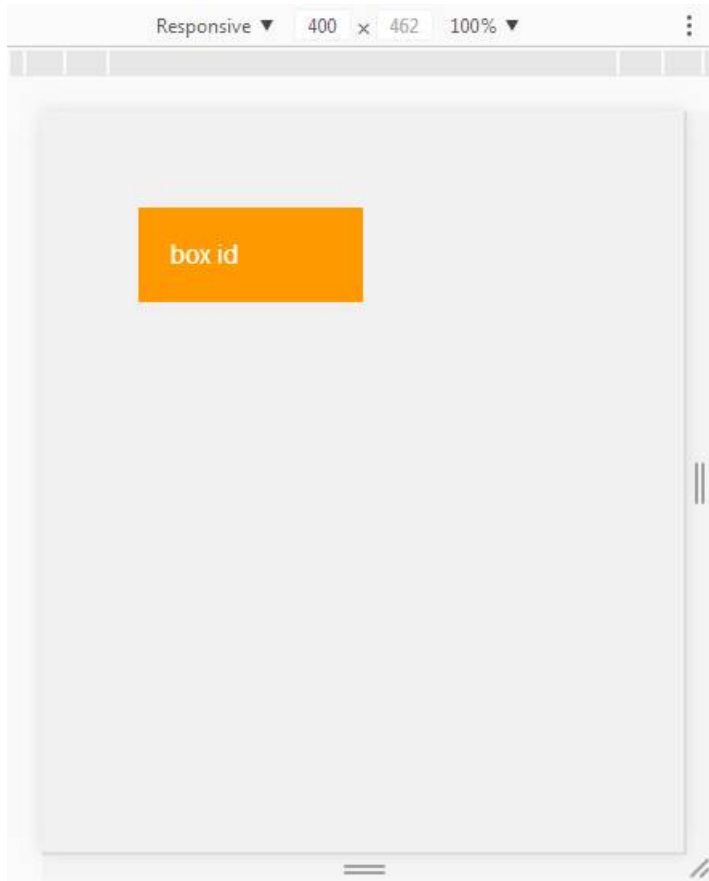
가상 Mobile 테스트입니다.



뷰포트 코드가 없는 Mobile 테스트 화면입니다. 화면 전체가 축소되어서 보입니다.



뷰포트 코드가 추가된 Mobile 테스트 화면입니다. Mobile 해상도만큼 축소되어 화면이 보이는 것을 알 수 있습니다.



9행 : <link rel="stylesheet"

href="http://fonts.googleapis.com/css?family=Open+Sans:400,300,300italic,400italic,600,600italic,700,700italic,800,800italic">

CSS3에서 새로 등장한 폰트 적용 방식(Web Font)입니다.

참고 URL

http://www.w3schools.com/css/css3_fonts.asp

@font-face 규칙 :

자신의 PC에 폰트가 설치되어 있어 제대로 보이지만 폰트가 설치되지 않은 환경에서 보면 글꼴이 표현되지 않아 레이아웃을 망치게 됩니다. 해결 방법은 문서 내에 폰트를 저장시키는 방식, 즉 웹 폰트를 문서 내에 포함시키는 것입니다.

CSS3에서의 새로운 기능인 웹 폰트(Web Font)를 통해 사용자의 컴퓨터에 설치되지 않은 글꼴을 웹 페이지에 사용할 수 있습니다. 만일 사용하고자 하는 폰트가 있다면, 웹 서버에 글꼴 파일을 포함하고,

필요할 경우 사용자에게 다운로드 되도록 합니다.

웹 폰트는 @font-face 규칙에 의해 정해집니다.

다양한 폰트 포맷 :

- TrueType Fonts(TTF)
TrueType은 1980년대 후반에 Apple과 Microsoft에서 개발된 글꼴에 표준입니다.
TrueType은 Mac OS와 Microsoft Windows 운영체제에서 가장 기본적으로 사용하는 폰트 포맷입니다.
- OpenType Fonts(OTF)
OpenType은 확장 가능한 컴퓨터 글꼴 형식입니다. TrueType에 기반을 두고 있으며 Microsoft의 등록 상표입니다. OpenType 글꼴은 현재 컴퓨터 플랫폼으로 일반적으로 많이 사용되고 있습니다.
- The Web Open Font Format(WOFF)
WOFF는 웹 페이지에 사용되는 2009년에 개발된 W3C 추천 글꼴 형식입니다.
- The Web Open Font Format(WOFF 2.0)
WOFF 1.0보다 더 압축을 제공하는 트루 타입이며 오픈 타입 글꼴입니다.
- SVG Fonts/Shapes
텍스트를 표시할 때 SVG 폰트는 글리프로서 SVG를 사용할 수 있도록 합니다.
- Embedded OpenType Fonts(EOT)
EOT 글꼴은 웹 페이지에 포함된 글꼴을 사용하기 위해 Microsoft에서 설계된 오픈 타입 글꼴의 형태입니다.

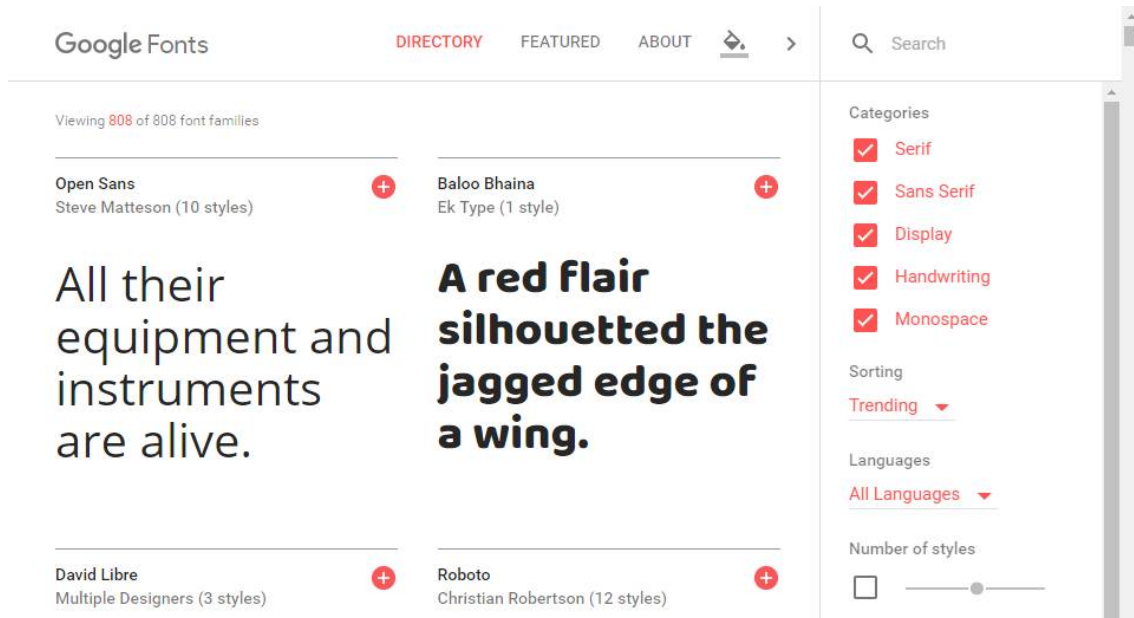
Web Font 사용 규칙 :

CSS3의 @font-face 규칙에서는 먼저 글꼴의 이름을 정의해야 합니다.

```
@font-face {  
    font-family: "Open Sans", sans-serif;  
    src: url(font/opensans.woff);  
}
```

외부 CDN 폰트의 사용(Google Fonts) :

Google 폰트는 웹 폰트를 사용하고자 할 때, 손쉽고 빠른 방법이며 공개된 글꼴로 상업적인 이용이 가능합니다.



CDN(Contents Delivery Network)

콘텐츠 전송 네트워크(Content delivery network 또는 Content distribution network)는 콘텐츠를 효율적으로 전달하기 위해 여러 노드를 가진 네트워크에 데이터를 저장하여 제공하는 시스템을 말한다. 인터넷 서비스 제공자에 직접 연결되어 데이터를 전송하므로, 콘텐츠 병목을 피할 수 있는 장점이 있다.

CDN의 목적은 높은 사용성과 효율로 사용자에게 콘텐츠를 전달함에 있다. CDN은 오늘날 인터넷에 존재하는 콘텐츠의 상당수를 서비스하고 있는데, 이에에는 웹 요소(텍스트, 그래픽, 스크립트), 다운로드 가능한 요소(미디어 파일, 소프트웨어, 문서), 애플리케이션(전자상거래, 포털), 실시간 미디어, 주문형 스트리밍, 그리고 소셜 네트워크 등이 있다.

출처 : https://ko.wikipedia.org/wiki/콘텐츠_전송_네트워크

자, 이제 처음으로 Google 웹 폰트를 사용해 봅시다. 아래 URL에 들어가서, 맘에 드는 폰트를 고르세요.

참고 URL

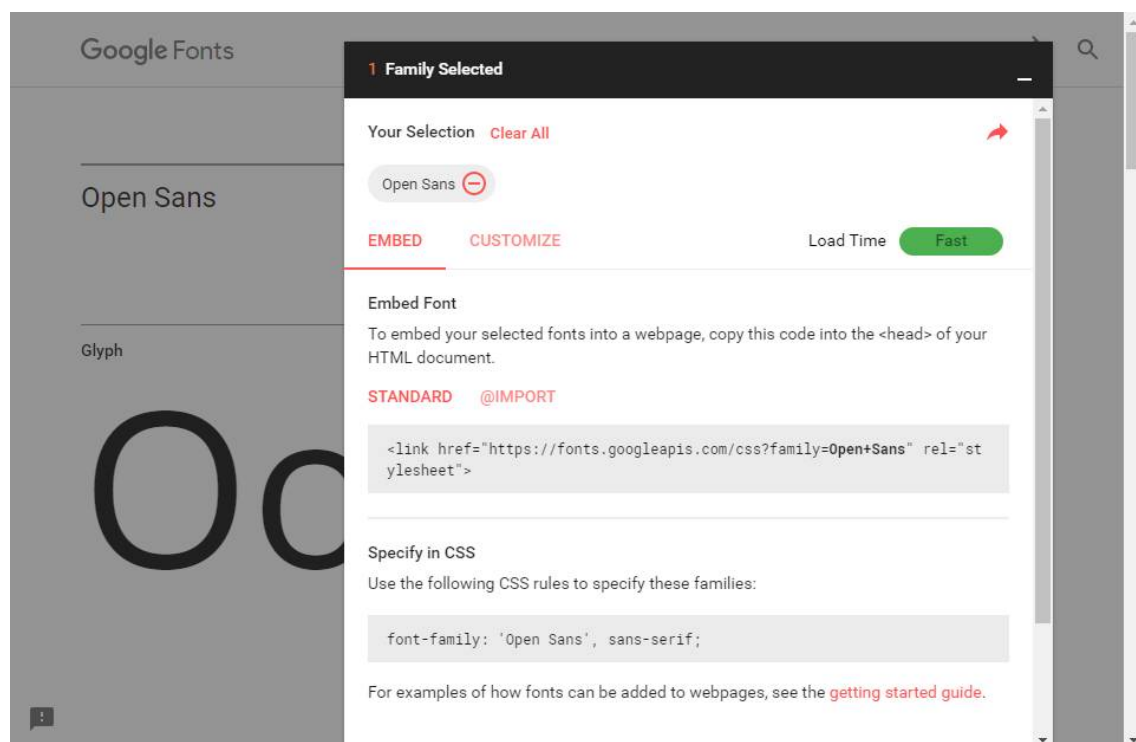
<https://www.google.com/fonts>

맘에 드는 폰트를 고르셨나요? 그렇다면 그 폰트를 사용해 봅시다.

원하는 폰트를 지정한 후에 'SELECT THIS FONT'를 클릭한 후에 폰트의 사용법을 확인할 수 있습니다.



폰트를 지정한다면, 하단 영역에 '1 Family Select'라고 활성화된 영역이 보일 것입니다. 그 활성화된 회색 영역을 클릭하면 아래 그림과 같은 정보 창을 확인할 수 있습니다.



[CUSTOMIZE] 버튼을 누르면 폰트의 다양한 두께나 스타일을 고를 수 있습니다.

1 Family Selected

Your Selection Clear All

Open Sans

EMBED

CUSTOMIZE

Load Time Fast

Open Sans

☐ light 300

☐ *light 300 Italic*

☒ **regular 400**

☐ *regular 400 Italic*

☐ semi-bold 600

☐ *semi-bold 600 Italic*

☐ bold 700

☐ *bold 700 Italic*

☐ extra-bold 800

☐ *extra-bold 800 Italic*

다시 [EMBED] 버튼을 눌러서 정보 창의 내용을 바꾸어 봅시다.

폰트 관련 CSS를 추가하는 방식은 [STANDARD] 방식과 [@IMPORT] 방식이 있습니다.

EMBED

CUSTOMIZE

Load Time Fast

Embed Font

To embed your selected fonts into a webpage, copy this code into the <head> of your HTML document.

STANDARD

@IMPORT

```
<link href="https://fonts.googleapis.com/css?family=Open+Sans" rel="stylesheet">
```

[STANDARD] 방식은 head 요소에 폰트 CSS를 링크하는 방법입니다.

```
<link href="https://fonts.googleapis.com/css?family=Open+Sans" rel="stylesheet">
```

[@IMPORT] 방식은 외부 스타일 파일에서 폰트 CSS를 링크하는 방법입니다.

```
@import 'https://fonts.googleapis.com/css?family=Open+Sans';
```

'Specify in CSS'은 실제 요소에 적용하기 위한 폰트의 이름입니다.

```
font-family: 'Open Sans', sans-serif;
```

위의 예제에서 font-size에 관련된 크기 단위를 em 단위로 사용했습니다.
1em이 16px로 정해져 있다면, 0.875em은 14px입니다.

보통 PC를 기준으로 웹 사이트를 작업할 때 주로 px 단위로 작업해 왔을 것입니다.
반응형 웹에서는 px보다는 퍼센트(%)나 em 단위 혹은 rem 단위로 작업합니다.

픽셀을 em으로 변환해주는 pxtoem 사이트입니다.

참고 URL

<http://pxtoem.com/>

The screenshot shows the interface of the pxtoem.com website. It is divided into three main sections: 1. Enter a base pixel size, 2. Convert, and 3. Result. In the '1. Enter a base pixel size' section, the number '16' is entered in a text box next to the unit 'px'. The '2. Convert' section has two tabs: 'PX to EM' and 'EM to PX'. The 'PX to EM' tab is active, showing '16' in a text box next to 'px'. There is an 'or' option and an empty text box next to 'em'. Below these is a 'Convert' button. The '3. Result' section shows the output '1.000em'.

1-2-2) 링크에 관련된 예제

다음 예제는 링크된 텍스트에 관련된 예제입니다.

참고 URL

http://www.w3schools.com/css/css_link.asp

링크된 텍스트는 CSS 속성으로 스타일이 될 수 있습니다.

추가적으로 상태가 어떤 상태인지에 따라서 다르게 스타일이 될 수 있습니다.

4가지의 링크 스타일로 나눌 수 있습니다. :

- a:link 한 번도 방문하지 않은 링크입니다.
- a:visited 방문한 링크입니다.
- a:hover 해당 요소에 마우스를 올린 상태입니다.
- a:active 해당 요소에 마우스를 클릭한 상태입니다.

여러 링크 상태에 대한 스타일을 설정하는 경우, 순서에 대한 규칙이 있습니다.

hover는 반드시 link나 visited보다 뒤에 배치되어야 하고, active는 반드시 hover보다 뒤에 배치되어야 합니다.

시작 파일

sample/basic_css/start/basic_text_css.html



HTML

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <meta http-equiv="X-UA-Compatible" content="IE=edge">
6 <meta name="viewport" content="width=device-width, initial-scale=1.0,
  minimum-scale=1.0">
7 <title>basic :: link style</title>
8 <link rel="stylesheet"
  href="http://fonts.googleapis.com/css?family=Open+Sans:400,300,300italic,400italic,600,6
  00italic,700,700italic,800,800italic">
9 <style>
10 body {
11     margin: 20px;
12     padding: 20px;
13     line-height: 1;
14     font-family: "Open Sans", sans-serif;
15     font-size: 1em;
16     background-color: #555;
17 }
18 .title {
19     margin: 0;
20     margin-bottom: 35px;
21     padding: 0;
22     font-size: 1.5em;
23     font-weight: 300;
24     color: #000;
25 }
26 .container {
27     margin-top: 35px;
28 }
29 .container a {
30     text-decoration: none;
31     color: #8ac007;
32 }
33 .container a:hover {
34     text-decoration: underline;
35 }
36 .container .orange {
37     color: #f90;
38 }
39 .container .orange a {
40     color: #f90;
```

```

41     }
42     .container .orange a:hover {
43         text-decoration: underline;
44         color: #f60;
45     }
46 </style>
47 </head>
48
49 <body>
50 <h1 class="title">basic :: link style</h1>
51 <div class="container">
52     text<br><br><br><br>
53     <a href="#">link text</a><br><br><br><br>
54     <span class="orange">orange text</span><br><br><br><br>
55     <span class="orange"><a href="#">orange link text</a></span>
56 </div>
57 </html>

```

26행 : .container {

텍스트를 감싸는 컨테이너 영역입니다.

31행 : .container a {color: #8ac007;}

클래스가 적용되지 않은 a 요소입니다. a 요소의 색상은 #8ac007입니다.

HTML 영역입니다.

```
<a href="#">link text</a>
```

37행 : .container .orange {color: #f90;}

orange 클래스입니다. body 요소의 색상은 #555이지만, orange 클래스 적용으로 #f90로 정해집니다.

HTML 영역입니다.

```
<span class="orange">orange text</span>
```

40행 : `.container .orange a {color: #f90;}`

orange 클래스의 색상을 지정해 주었지만, 기본 a 요소의 색상인 #8ac007 색상으로 정해집니다.
따라서 orange 클래스 하위에 a 요소도 특별히 색상을 지정합니다.

HTML 영역입니다.

```
<span class="orange"> <a href="#">orange link text</a></span>
```

44행 : `.container .orange a:hover {color: #f60;}`

orange 클래스 하위 a 요소에 마우스를 올렸을 경우에는, 밑줄이 생기면서 색상이 변경됩니다.

1-2-3) 이해하기 어려운 display 속성

CSS 속성은 사실 그 속성명만 가지고 어느 스타일이 어떤 것인지를 알 수 있습니다.

color는 글자 색상에 관련된 스타일이고, background-color는 바탕 색상에 관련된 스타일이겠죠.
따라서 이런 내용을 처음 접하는 초보자도 쉽게 이해가 될 것입니다.
하지만 가장 기본적인 내용이면서 이해하기 어려운 CSS 속성도 있습니다.

바로 display 속성과 position 속성과 같은 것들입니다.

다음 예제는 가장 중요한 속성 중에 하나인 CSS 요소의 display 속성에 대해 알아보겠습니다.

참고 URL

http://www.w3schools.com/css/css_display_visibility.asp

display 속성은 레이아웃을 위한 가장 중요한 CSS 속성으로 display 속성은 요소가 표시되는 스타일을 지정합니다.

모든 HTML 요소는 기본적으로 고유의 display 속성을 가지고 있습니다. 가장 기본적인 display에 관련된 속성 값은 'block'과 'inline'입니다.

block 요소 :

block 요소는 항상 새로운 행에 시작되며, 사용할 수 있는 전체 가로 크기를 차지합니다.

다음과 같은 HTML 요소가 해당됩니다. :

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

Inline 요소 :

inline 요소는 새로운 행에서 시작되지 않으며, 영역이 차지한 만큼의 가로 크기를 차지합니다.

다음과 같은 HTML 요소가 해당됩니다. :

-
- <a>

display: none; :

display: none; 속성은 일반적으로 해당 요소를 삭제시킬 때 사용합니다.

HTML 요소가 가지는 고유한 display 속성 값의 변경 :

HTML 요소가 가지는 고유한 display 속성을 변경할 수 있습니다.

다음의 예는 li 요소를 통해서 가로 메뉴를 만들 때 사용하는 방식일 것입니다.

```
li {  
    display: inline;  
}
```

다음의 예는 span 요소의 가로, 세로 크기를 적용하고 싶을 때 display 속성을 block으로 변경합니다.

```
span {  
    display: block;  
}
```

요소를 감추는 방식(display: none; 혹은 visibility: hidden;) :

display: none; 방식의 스타일은 해당 요소를 가릴 때 사용하며, 화면에는 없는 요소처럼 보입니다.

```
h1.hidden {  
    display: none;  
}
```

visibility: hidden; 방식의 스타일도 마찬가지로 요소가 가려질 것입니다.

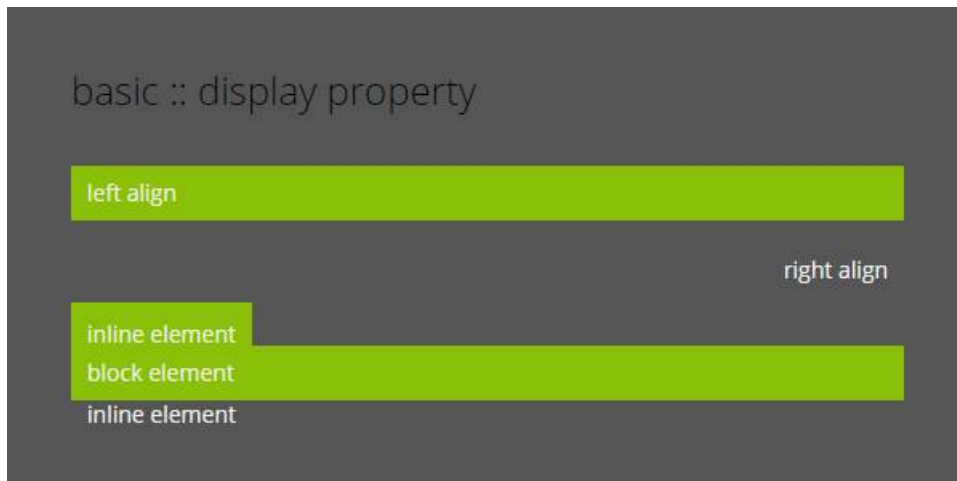
하지만 자신의 영역을 차지하고 있습니다. 요소는 없어지는 것이 아닌 가려진다고 할 수 있습니다.

```
h1.hidden {  
    visibility: hidden;  
}
```

이제까지 배운 display 속성을 정리할 예제를 하나 준비했습니다.

시작 파일

sample/basic_css/start/basic_display.html



HTML

```
1 <!DOCTYPE html>  
2 <html>  
3 <head>
```

```
4 <meta charset="utf-8">
5 <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
6 <meta name="viewport" content="width=device-width, initial-scale=1.0,
  minimum-scale=1.0">
7 <title>basic :: display property</title>
8 <link rel="stylesheet"
  href="http://fonts.googleapis.com/css?family=Open+Sans:400,300,300italic,400italic,600,6
  00italic,700,700italic,800,800italic">
9 <style>
10 body {
11     margin: 20px;
12     padding: 20px;
13     line-height: 1;
14     font-family: "Open Sans", sans-serif;
15     font-size: 1em;
16     background-color: #555;
17     color: #f1f1f1;
18 }
19 .title {
20     margin: 0;
21     margin-bottom: 35px;
22     padding: 0;
23     font-size: 1.5em;
24     font-weight: 300;
25     color: #000;
26 }
27 .container {
28     margin-top: 35px;
29 }
30 .container p,
31 .container span {
32     padding: 10px;
33     font-size: 0.875em;
34 }
35 .container .left {
36     text-align: left;
37     background-color: #8ac007;
38 }
39 .container .right {
40     text-align: right;
41 }
42 .container .span1 {
43     display: inline;
44     background-color: #8ac007;
```



```

45     }
46     .container .span2 {
47         display: block;
48         background-color: #8ac007;
49     }
50 </style>
51 </head>
52
53 <body>
54 <h1 class="title">basic :: display property</h1>
55 <div class="container">
56     <p class="left">left align</p>
57     <p class="right">right align</p>
58     <span class="span1">inline element</span>
59     <span class="span2">block element</span>
60     <span>inline element</span>
61 </div>
62 </body>
63 </html>

```

36행 : .container .left {text-align: left;}

p 요소는 기본적으로 block 요소입니다.

text-align 속성은 텍스트의 정렬에 관련된 CSS 속성입니다.

HTML 영역입니다.

```
<p class="left">left align</p>
```

47행 : .container .span2 {display: block;}

span 요소는 기본적으로 inline 요소입니다.

display의 속성을 block으로 변경합니다.

HTML 영역입니다.

```
<span class="span2">block element</span>
```

1-2-4) 요소의 위치하는 방식에 관련된 position 속성

position은 요소가 위치하는 방식을 결정합니다.(static, relative, absolute, fixed)
기본 속성 값은 static입니다.

간단히 내용을 표로 정리해 보면 다음과 같습니다.

속성 값	설명
static	기본 속성 값입니다. 나타나는 순서대로 화면에 표시됩니다.
absolute	relative 요소를 기준으로 위치됩니다.
fixed	브라우저를 기준으로 위치됩니다.
relative	static과 같은 방식으로 나타나는 순서대로 화면에 표시됩니다. 단 위치된 요소는 하위 absolute 요소의 기준이 됩니다.
initial	위치 속성을 기본 값으로 스타일합니다.
inherit	상위 요소의 position 속성 그대로 적용합니다.

간단한 예제로써 알아보도록 하겠습니다.

시작 파일

sample/basic_css/start/basic_position.html

HTML

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4  <meta charset="utf-8">
5  <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
6  <meta name="viewport" content="width=device-width, initial-scale=1.0,
  minimum-scale=1.0">
7  <title>basic :: position property</title>
8  <link rel="stylesheet"
  href="http://fonts.googleapis.com/css?family=Open+Sans:400,300,300italic,400italic,600,6
  00italic,700,700italic,800,800italic">
9  <style>
10 body {
11     margin: 20px;
12     padding: 20px;
13     line-height: 1;
14     font-family: "Open Sans", sans-serif;
15     font-size: 1em;
16     background-color: #555;
17     color: #f1f1f1;
18 }
19 .title {
20     margin: 0;
21     margin-bottom: 35px;
22     padding: 0;
23     font-size: 1.5em;
24     font-weight: 300;
25     color: #000;
26 }
27 .container {
28     margin-top: 35px;
29 }
30 .box {
31     margin-top: 35px;
32     width: 400px;
```

```
33     height: 400px;
34     border: 1px solid #000;
35 }
36 #position {
37     position: static;
38     padding: 10px;
39     width: 80px;
40     height: 80px;
41     font-size: 0.875em;
42     background-color: #8ac007;
43     color: #f1f1f1;
44 }
45 </style>
46 </head>
47
48 <body>
49 <h1 class="title">basic :: position property</h1>
50 <div class="container">
51     <div class="box">
52         <div id="position">#position</div>
53     </div>
54 </div>
55 </html>
```

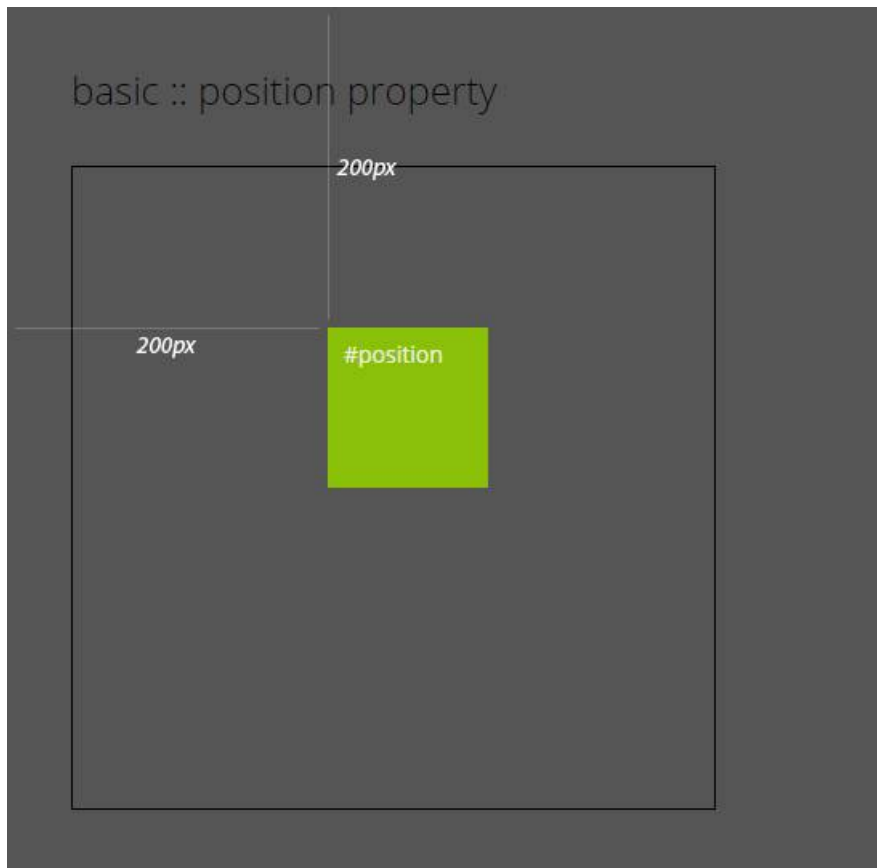
static 속성 :



37행 : #position {position: static;}

position 속성의 기본 속성 값은 static입니다.

absolute 속성 :



이전 예제의 CSS 속성을 변경해 보세요.

CSS

```
30 .box {
31     margin-top: 35px;
32     width: 400px;
33     height: 400px;
34     border: 1px solid #000;
35 }
36 #position {
37     position: absolute;
38     left: 200px;
39     top: 200px;
40     padding: 10px;
41     width: 80px;
42     height: 80px;
```

```
43     font-size: 0.875em;  
44     background-color: #8ac007;  
45     color: #f1f1f1;  
46 }
```

37행 : #position {position: absolute; left: 200px; top: 200px;}

position 속성에 대한 absolute 속성 값은 left, right 속성으로 좌표가 정해집니다.
브라우저에서 좌측과 상단에 200px 여백이 있는 사각형이 표시됩니다.

relative 속성 :



이전 예제의 CSS 속성을 변경해 보세요.

CSS

```
30 .box {
31     position: relative;
32     margin-top: 35px;
33     width: 400px;
34     height: 400px;
35     border: 1px solid #000;
36 }
37 #position {
38     position: absolute;
39     left: 50px;
40     top: 50px;
41     padding: 10px;
42     width: 80px;
```



```
43     height: 80px;
44     font-size: 0.875em;
45     background-color: #8ac007;
46     color: #f1f1f1;
47 }
```

31행 : `.box {position: relative;}`

부모 요소인 box 클래스에 position 속성을 relative 속성 값으로 적용하면 position id 요소는 위의 그림과 같이 달라집니다.

relative 부모 요소는 absolute 자식 요소의 위치를 정하는 기준이 됩니다.

38행 : `#position {position: absolute; left: 50px; top: 50px;}`

position id 요소는 box 클래스의 위치를 기준으로 left 50px, top 50px 영역으로 위치합니다.

fixed 속성 :



이전 예제의 CSS 속성을 변경해 보세요.

CSS

```
30 .box {
31     position: relative;
32     margin-top: 35px;
33     width: 400px;
34     height: 2000px;
35     border: 1px solid #000;
36 }
37 #position {
38     position: fixed;
39     left: 100px;
40     top: 100px;
41     padding: 10px;
42     width: 80px;
```

```
43     height: 80px;
44     font-size: 0.875em;
45     background-color: #8ac007;
46     color: #f1f1f1;
47 }
```

38행 : #position {position: fixed;}

position 속성을 fixed 속성 값으로 지정하면 브라우저를 기준으로 위치합니다.

반면 absolute 속성 값으로 지정하면, 부모 요소의 position 속성이 relative인 경우에는 부모 요소를 기준으로 위치합니다.

1-2-5) clearfix 클래스의 활용

참고 URL

<http://nicolasgallagher.com/micro-clearfix-hack/>

<https://drafts.csswg.org/css-sizing/#the-contain-floats-value>

레이아웃을 만들 때에 대부분의 경우에는, 하위 div 요소와 그것을 둘러싼 또 하나의 div 요소로 구성합니다. 하위 div 요소가 좌우로 float 속성이 지정된다면, 상위 div 요소는 몇 가지 필요한 스타일이 있습니다.

간단한 테스트를 해봅시다.

시작 파일

sample/basic_css/start/basic_clearfix.html

HTML

```
1      <!DOCTYPE html>
2      <html>
3      <head>
4      <meta charset="utf-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0,
7      minimum-scale=1.0">
8      <title>basic :: clearfix class</title>
9      <link rel="stylesheet"
10      href="http://fonts.googleapis.com/css?family=Open+Sans:400,300,300italic,400italic,600,6
11      00italic,700,700italic,800,800italic">
12      <style>
13      body {
14          margin: 20px;
15          padding: 20px;
16          line-height: 1;
17          font-family: "Open Sans", sans-serif;
18          font-size: 1em;
19          background-color: #555;
20      }
21      .title {
22          margin: 0;
23          margin-bottom: 35px;
24          padding: 0;
```

```

22     font-size: 1.5em;
23     font-weight: 300;
24     color: #000;
25 }
26 .container1 {
27     margin-top: 35px;
28     width: 545px;
29     font-size: 0.875em;
30     border: 4px solid #8ac007;
31 }
32 .container1 > div {
33     float: left;
34     padding: 10px;
35     width: 250px;
36     height: 250px;
37     border: 1px solid #000;
38 }
39 .container2 {
40     display: inline-block;
41     padding: 10px;
42     width: 500px;
43     font-size: 0.875em;
44     border: 5px solid #8ac007;
45 }
46 </style>
47 </head>
48
49 <body>
50 <h1 class="title">basic :: clearfix class</h1>
51 <div class="container1 clearfix">
52     <div class="left">left element</div>
53     <div class="right">right element</div>
54 </div>
55 <div class="container2">etc</div>
56 </body>
57 </html>

```

26행 : .container1 {

container1 클래스는 float된 하위 요소를 담고 있는 컨테이너입니다.

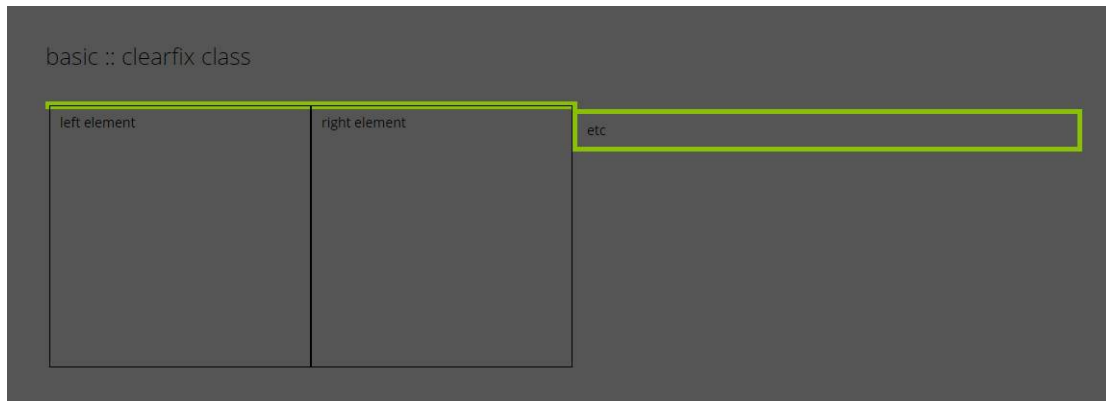
33행 : .container1 > div {float: left;}

container1 클래스 하위의 자식 요소는 float: left; 스타일로 왼쪽 정렬됩니다.

39행 : .container2 {

container2 클래스는 display 속성 값은 inline-block입니다. inline-block은 가로 정렬되도록 정해진 block 요소입니다.

결과를 보니 레이아웃이 뒤죽박죽되었습니다.



float 속성을 사용할 때에는 float 속성이 적용된 컨테이너에 대한 처리가 중요합니다. 더 쉽게 설명한다면, container2 클래스는 수직 아래의 위치로 이동되어야 합니다. 위의 그림처럼 container1 클래스에 배치되면 안 됩니다.

float된 요소의 상위 컨테이너 요소에 처리되는 스타일 방식을 'clearing floats' 또는 'clearfix methods'라고 합니다. 가장 많이 쓰는 방식은 ::before와 ::after 가상 요소를 이용해서 해결될 수 있습니다.

CSS 부분만 다시 작성해 봅시다.

CSS

```
9 <style>
10 body {
11     margin: 20px;
12     padding: 20px;
13     line-height: 1;
14     font-family: "Open Sans", sans-serif;
15     font-size: 1em;
16     background-color: #555;
17 }
18 .title {
19     margin: 0;
20     margin-bottom: 35px;
```

```

21     padding: 0;
22     font-size: 1.5em;
23     font-weight: 300;
24     color: #000;
25 }
26 .container1 {
27     margin-top: 35px;
28     width: 545px;
29     font-size: 0.875em;
30     border: 4px solid #8ac007;
31 }
32 .container1 > div {
33     float: left;
34     padding: 10px;
35     width: 250px;
36     height: 250px;
37     border: 1px solid #000;
38 }
39 .container2 {
40     display: inline-block;
41     padding: 10px;
42     width: 500px;
43     font-size: 0.875em;
44     border: 5px solid #8ac007;
45 }
46 .clearfix::before,
47 .clearfix::after {
48     content: "";
49     display: block;
50 }
51 .clearfix::after {
52     clear: both;
53 }
54 </style>

```

48행 : .clearfix::before, .clearfix::after {content: ""; display: block;}

clearfix 클래스 이전 가상 요소와 다음 가상 요소에 빈 block 속성으로 적용합니다.

50행 : .clearfix::after {clear: both;}

clearfix 클래스 다음 가상 요소에 대해서, float에 대한 속성을 제거합니다.

basic :: clearfix class



float을 사용한 레이아웃을 구현할 경우 생각해 보아야 할 것은, 부모 요소가 float 요소를 담아낼 수 있고 다음에 나타날 요소가 float에 영향을 받지 않도록 float을 none 속성 값으로 지정해야 합니다.

웹 페이지 작업을 하다 보면, float 속성을 사용할 경우가 많이 있을 것입니다. 따라서 중복된 스타일은 clearfix 클래스로 만들어서 사용하는 것이 좋습니다.

이번에는 기초적인 레이아웃을 통해서 HTML5의 새로운 요소와 CSS를 연습해 봅시다.

시작 파일

sample/basic_css/start/basic_layout.html



HTML

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
6 <meta name="viewport" content="width=device-width, initial-scale=1.0,
  minimum-scale=1.0">
7 <title>basic :: basic layout</title>
8 <link rel="stylesheet"
  href="http://fonts.googleapis.com/css?family=Open+Sans:400,300,300italic,400italic,600,6
  00italic,700,700italic,800,800italic">
9 <style>
10 body {
11     margin: 20px;
12     padding: 20px;
13     line-height: 1;
14     font-family: "Open Sans", sans-serif;
15     font-size: 1em;
```

```
16     background: #555;
17     color: #f1f1f1;
18 }
19 .title {
20     margin: 0;
21     margin-bottom: 35px;
22     padding: 0;
23     font-size: 1.5em;
24     font-weight: 300;
25     color: #000;
26 }
27 .container {
28     margin-top: 35px;
29     width: 800px;
30     font-size: 0.875em;
31     color: #333;
32 }
33 header {
34     padding: 10px;
35     height: 40px;
36     background: #e8e7e5;
37 }
38 section .left {
39     float: left;
40     padding: 10px;
41     width: 100px;
42     height: 200px;
43     background: #f0e0a2;
44 }
45 section .center {
46     float: left;
47     padding: 10px;
48     width: 540px;
49     height: 200px;
50     background: #fed1be;
51 }
52 section .right {
53     float: right;
54     padding: 10px;
55     width: 100px;
56     height: 200px;
57     background: #abe1fd;
58 }
59 footer {
```

```

60     clear: both;
61     padding: 10px;
62     height: 40px;
63     background-color: #bfb1d5;
64 }
65 .clearfix::before,
66 .clearfix::after {
67     content: "";
68     display: table;
69 }
70 .clearfix::after {
71     clear: both;
72 }
73 </style>
74 <!--[if lt IE 9]>
75 <script src="http://html5shim.googlecode.com/svn/trunk/html5.js"></script>
76 <![endif]-->
77 </head>
78
79 <body>
80 <h1 class="title">basic :: basic layout</h1>
81 <div class="container">
82     <header>header</header>
83     <section class="clearfix">
84         <div class="left">left</div>
85         <div class="center">center</div>
86         <div class="right">right</div>
87     </section>
88     <footer>footer</footer>
89 </div>
90 </body>
91 </html>

```

33행 : header {

상단 영역의 스타일을 지정합니다. 1단으로 구성되며, 바탕 색상은 #e8e7e5입니다.

38행 : section .left {

두 번째 행에서의 왼쪽 영역을 지정합니다. float: left; 방식으로 정렬되며, 바탕 색상은 #f0e0a2입니다.

45행 : section .center {

두 번째 행에서의 중앙 영역을 지정합니다. left 클래스와 마찬가지로 float: left; 방식으로 정렬되며, 바탕 색상은 #fed1be입니다.

52행 : `section .right {`

두 번째 행에서의 오른쪽 영역을 지정합니다. `float: right;` 방식으로 정렬되며, 바탕 색상은 `#abe1fd`입니다. `right` 클래스의 가로 크기는 100px입니다.

두 번째 행에서 `float`된 요소들의 크기와 `container` 클래스의 크기가 맞지 않는다면, 레이아웃이 틀어지게 될 것입니다.

`container` 클래스의 가로 크기는 800px이며, `left`, `center`, `right` 클래스의 `padding` 속성은 각각 10px입니다. `padding` 속성은 실제 너비를 계산할 때 추가가 되는 영역이므로 다음과 같이 크기 계산이 됩니다.

left 클래스의 가로 크기 : $120\text{px}(10+10+100)$

center 클래스의 가로 크기 : $560\text{px}(10+10+540)$

right 클래스의 가로 크기 : $120\text{px}(10+10+100)$

따라서 전체 컨테이너인 `container` 클래스의 가로 크기는 800px입니다.

요소에 대한 `padding`이나 `border` 속성은 실제 크기를 변화시키는 스타일 속성입니다.

예제에서와 같이 실제 크기를 꼼꼼하게 계산해서 레이아웃이 틀어지는 것을 방지하는 방법도 있고, `box-sizing` 속성을 사용할 수도 있습니다.

다음의 예제는 box-sizing: border-box;를 사용한 기본 레이아웃 예제입니다.

CSS

```
38  section > div {
39      -webkit-box-sizing: border-box;
40      box-sizing: border-box;
41  }
42  section .left {
43      float: left;
44      padding: 10px;
45      width: 100px;
46      height: 200px;
47      background: #f0e0a2;
48  }
49  section .center {
50      float: left;
51      padding: 10px;
52      width: 600px;
53      height: 200px;
54      background: #fed1be;
55  }
56  section .right {
57      float: right;
58      padding: 10px;
59      width: 100px;
60      height: 200px;
61      background: #abe1fd;
62  }
```

사실 이전에 사용하던 CSS를 통한 가로 크기를 정하는 방법은 그다지 만족스럽지 않았습니다. 하지만 CSS3 요건에서는 box-sizing과 같은 새로운 CSS 속성을 추가했습니다. 해당 요소에 box-sizing: border-box; 속성을 적용하면, padding과 border 속성은 실제 요소에 대한 부피와는 상관이 없습니다.

따라서 요소에 지정된 가로 크기 그대로 정해지는 것입니다.

참고 URL

http://www.w3schools.com/cssref/css3_pr_box-sizing.asp

box-sizing 속성은 CSS3의 요건이기 때문에 브라우저에 따른 처리를 위해서 '-webkit'과 같은 접두사를 붙여야 합니다. 이를 프리픽스(prefix)라고 합니다.

또한 box-sizing 속성을 사용하는 데에 유의할 점은 Internet Explorer 9 이상 버전에서부터 적용된다는 점입니다. 따라서 하위 Internet Explorer 버전에 대한 대안도 필요할 것입니다.

38행 : `section > div {`

section 요소 하위의 자식 요소(children)를 참조합니다.

39행 : `-webkit-box-sizing: border-box;`

box-sizing: border-box; 속성으로 padding이나 border 속성으로 크기에 영향 받지 않도록 합니다.

'-webkit' 접두사 방식으로 Chrome 4.0 이하의 브라우저에 적용합니다.

52행 : `width: 600px;`

box-sizing: border-box; 속성으로 실제 크기가 바뀌게 됩니다. 이전에는 540px이었습니다.

69행 : `.clearfix::before, .clearfix::after {`

clearfix 클래스를 통한 float된 요소에 처리 방법입니다.

관련 HTML 요소는 다음과 같습니다.

```
<section class="clearfix">
```

78행 : `<!--[if lt IE 9]>`

```
<script src="http://html5shim.googlecode.com/svn/trunk/html5.js"></script>
```

```
<![endif]-->
```

오래된 브라우저에서도 제대로 된 화면을 확인하기 위해서는 다음의 JavaScript 파일을 링크합니다.

'lt'란 조건부 주석으로서, 크로스 브라우징을 할 경우에 HTML의 주석을 이용해서 Internet Explorer에 대해 적용하기 위한 특수한 형식의 주석입니다.

조건부 주석을 이용하면 주석 내부에 있는 내용은 지원하는 브라우저의 조건이 참이면 마크업 된 내용은 브라우저에 렌더링 됩니다.

조건부 주석은 세부적인 조건을 사용할 수 있습니다. 간단한 예로서 설명한 표입니다.

조건의 내용	예제	조건에 대한 설명
!	[if !IE]	IE가 아닐 경우에 참 값을 가집니다.
lt	[if lt IE 9]	해당 버전 미만 버전의 IE에서 항상 참 값을 가집니다.(lt = less than)

lte	[If lte IE 9]	해당 버전 이하의 버전에서 항상 참 값을 가집니다.(lte = less than or equal)
gt	[If gt IE 6]	해당 버전을 초과하는 버전의 Internet Explorer에서 항상 참 값을 가집니다.(gt = greater than)
gte	[If gte IE 6]	해당 버전 이상의 버전의 Internet Explorer에서 항상 참 값을 가집니다.(gte = greater than or equal)
()	[if !(IE 7)]	세부적인 표현을 위한 형식입니다. 수학의 계산과 프로그래밍과 동일하게 괄호 안의 표현식부터 먼저 처리됩니다.
&	[if (gt IE 5)&(lt IE 7)]	프로그래밍에서의 '&&'와 동일하게 'and'를 의미합니다.
	[if (IE 6) (IE 8)]	프로그래밍에서의 ' '와 동일하게 'or'를 의미합니다.

2) CSS Reset

CSS 초기화(CSS Reset)란 브라우저마다 동일한 스타일을 적용하기 위해 기본적인 CSS를 정해주는 것을 의미합니다. CSS normalize, default CSS와 같은 스타일들도 CSS Reset과 같은 용도로 사용됩니다.

웹 브라우저는 HTML 요소에 대한 기본 모양을 각 브라우저마다 다르게 정의하고 있습니다. 따라서 웹 브라우저에 상관없이 항상 같은 모양을 나타내도록 CSS를 초기화할 필요가 있는 것입니다.

CSS를 초기화하는 방법 중 가장 일반적으로 쓰이는 방법은 에릭 마이어(Eric Meyer)의 CSS Reset입니다.

모든 모양을 다시 설정해야 하는 불편함과 CSS 초기화에 대한 다양한 논란이 있기도 하지만, 세밀한 설정을 위해서는 CSS를 초기화하는 것도 좋은 방법입니다.

에릭 마이어(Eric Meyer)의 CSS Reset

참고 URL

<http://meyerweb.com/eric/tools/css/reset/reset.css>

```
/* http://meyerweb.com/eric/tools/css/reset/  
v2.0 | 20110126  
License: none(public domain)  
*/
```

```
html, body, div, span, applet, object, iframe,  
h1, h2, h3, h4, h5, h6, p, blockquote, pre,  
a, abbr, acronym, address, big, cite, code,  
del, dfn, em, img, ins, kbd, q, s, samp,  
small, strike, strong, sub, sup, tt, var,  
b, u, i, center,  
dl, dt, dd, ol, ul, li,  
fieldset, form, label, legend,  
table, caption, tbody, tfoot, thead, tr, th, td,  
article, aside, canvas, details, embed,  
figure, figcaption, footer, header, hgroup,  
menu, nav, output, ruby, section, summary,  
time, mark, audio, video {  
    margin: 0;  
    padding: 0;  
    border: 0;
```



```
font-size: 100%;
font: inherit;
vertical-align: baseline;
}

/* HTML5 display-role reset for older browsers */
article, aside, details, figcaption, figure,
footer, header, hgroup, menu, nav, section {
    display: block;
}
body {
    line-height: 1;
}
ol, ul {
    list-style: none;
}
blockquote, q {
    quotes: none;
}
blockquote:before, blockquote:after,
q:before, q:after {
    content: "";
    content: none;
}
table {
    border-collapse: collapse;
    border-spacing: 0;
}
```

CSS 스타일 제작의 기본기를 알고 싶은 분들은 위의 언급된 에릭 마이어의 CSS 스타일을 공부하는 것이 좋을 것입니다.

하지만 실무 작업의 경우에는, 개발하는 웹 페이지마다 스타일이 다르기 때문에 자신만의 스타일을 만들어 내기를 권장합니다.

아래의 URL은 최신 CSS Reset 사이트입니다.

참고 URL

<http://www.cssreset.com/>

교재에서 사용할 CSS Reset 스타일을 설명하겠습니다.

```
<link rel="stylesheet" href="css/reset.css">
```

아래의 내용은 세부 스타일 요소에 대한 설명입니다.

```
html, body, div, span, applet, object, iframe,
h1, h2, h3, h4, h5, h6, p, blockquote, pre,
a, abbr, acronym, address, big, cite, code,
del, dfn, em, img, ins, kbd, q, s, samp,
small, strike, strong, sub, sup, tt, var,
b, u, i, center,
dl, dt, dd, ol, ul, li,
fieldset, form, label, legend,
table, caption, tbody, tfoot, thead, tr, th, td,
article, aside, canvas, details, embed,
figure, figcaption, footer, header, hgroup,
menu, nav, output, ruby, section, summary,
time, mark, audio, video {
    margin: 0;
    padding: 0;
    vertical-align: baseline;
    font: inherit;
    font-size: 100%;
    border: 0;
}
```

대부분의 요소의 margin, padding, border 속성을 0으로 지정합니다.

각각의 요소들은 서로 다른 크기의 margin, padding, border를 가지고 있으며, 여러 브라우저마다 다르게 보일 수 있습니다. 이런 문제점을 해결하기 위해 0으로 초기화시켜줍니다.

또한 세로 정렬(vertical-align)에 관련된 속성 값은 baseline으로 지정합니다.

글꼴(font, font-size)은 정해진 스타일을 따르도록 합니다.

HTML5에서 새로 나온 시멘틱 요소의 속성을 div 요소와 같은 block으로 지정합니다.

```
article, aside, details, figcaption, figure, footer, header, hgroup, menu, nav, section {  
    display: block;  
}
```

body의 행간(line-height)은 1로 지정합니다. 숫자 1은 현재의 폰트 크기에 맞춰진 스타일입니다.

```
body {  
    line-height: 1;  
}
```

img 요소의 크기는 담고 있는 영역의 가로 크기에 맞춰지도록 지정합니다. height 값은 width 값에 맞춰 비율대로 늘어나거나 줄어듭니다. 반응형 사이트를 만들 때에 적합한 속성입니다.

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

리스트 요소가 기본적으로 가진 리스트 스타일을 없애 줍니다.

```
ol, ul {  
    list-style: none;  
}
```

서로 이웃하는 셀의 테두리를 겹쳐서 하나의 선으로 표현합니다. 이는 이중으로 테두리 선이 나타나는 것을 방지해 줍니다.

```
table {  
    border-collapse: collapse;  
    border-spacing: 0;  
}
```

hr 요소를 단순한 모양을 가진 가로 선으로 지정합니다.

```
hr {  
  display: block;  
  margin: 16px 0;  
  padding: 0;  
  height: 1px;  
  border: 0;  
  border-top: 1px solid #ccc;  
}
```

input 요소와 select 요소는 다른 요소에 비해 vertical-align이 높게 설정되어 있으므로 middle로 지정합니다.

```
input, select {  
  vertical-align: middle;  
}
```

한글 폰트를 웹 폰트로 활용하기

Google early access에서는 한글 폰트를 지원해주기 때문에 쉽게 웹 사이트에 적용할 수 있습니다.

참고 URL

구글 웹 폰트 서비스 :

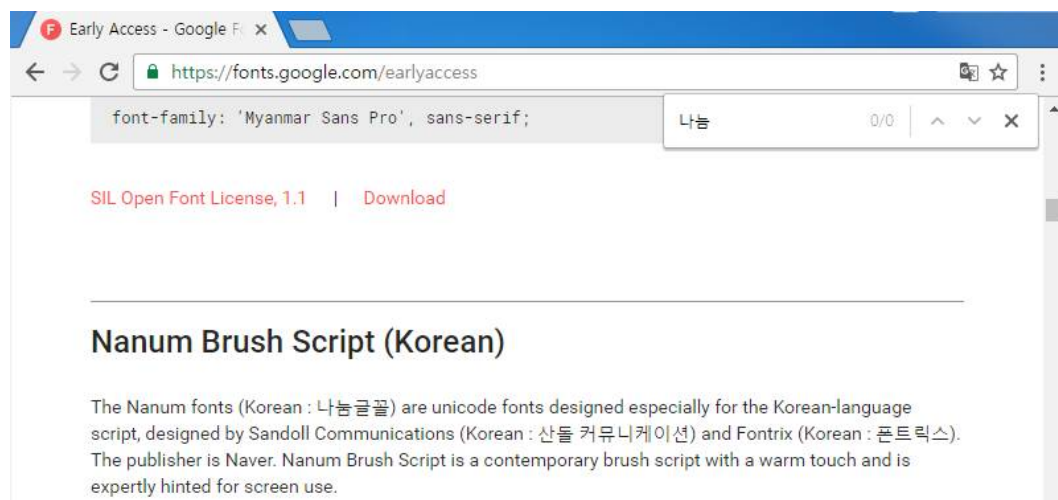
<http://www.google.com/fonts/earlyaccess>

<http://deminoth.github.io/google-font-kor/>

Chrome 브라우저에서 페이지 접속 후 찾기 기능(Ctrl+F)를 누르고 원하는 폰트를 찾습니다. 이후의 과정은 구글 웹 폰트의 사용 방법과 같습니다.

예를 들어 '나눔 고딕(Nanum Gothic)' 폰트를 사용하고자 한다면,

① 나눔으로 검색합니다.



② CSS 링크 방법은 다음과 같습니다.

```
@import url(http://fonts.googleapis.com/earlyaccess/nanumgothic.css);
```

위에 제시된 방법은 CSS에서 불러오는 방법이며, HTML head 영역에서의 링크 방법은 다음과 같습니다.

```
<link rel="stylesheet" href="http://fonts.googleapis.com/earlyaccess/nanumgothic.css">
```

③ 사용하고자 하는 텍스트에 폰트 이름을 지정합니다.

font-family: "Nanum Gothic", sans-serif;

3) 추가된 CSS3 요건

CSS3은 CSS에 대한 최신 표준입니다.

CSS3은 CSS의 이전 버전과 완전히 호환되므로. 이전 CSS 내용은 모두 사용할 수 있다고 보면 될 것입니다.

특별히 이전 버전과 차별화되는 점은 코드만으로 만들어진 디자인 효과, 웹 폰트의 사용, 동적인 기능 등을 들 수 있습니다.

코드로 만들어진 디자인 효과(Border Radius, Text Shadow, Box Shadow, Gradient) :

Adobe Photoshop과 같은 그래픽 프로그램을 통해 만들 수 있었던 둥근 모서리, 그림자 효과, 그라데이션을 코드만으로 만들 수 있습니다.

웹 폰트(Web Font) :

웹 폰트를 활용하면 사용자 환경과 상관없이 다양한 디자인 폰트를 사용할 수 있습니다.

애니메이션(Animation) :

이전에는 GIF 이미지나, Adobe Flash 프로그램을 통해 구현된 애니메이션을 'animation', 'transition'과 같은 CSS3 속성을 사용하여 구현할 수 있습니다.

밴더 프리픽스(Vendor Prefix) :

CSS3 요건들은 대부분 최종 규격이 승인되지 않았습니다. 이 때문에 다양한 브라우저에서 동일한 스타일을 적용하도록 여러 가지의 접두사를 제공하고 있습니다.

밴더 프리픽스는 CSS3의 속성마다 적용되는 내용이 달라지기 때문에 항상 관련 사이트를 확인할 필요가 있습니다.

참고 URL

http://www.w3schools.com/css/css3_intro.asp

벤더 프리픽스를 붙여주는 사이트

CSS3 표준 속성에 알아서 벤더 프리픽스를 붙여줍니다.

참고 URL

<http://cssprefixer.appspot.com>

또한 자바스크립트를 통해 CSS3 표준 속성을 알아서 벤더 프리픽스를 붙여주는 스크립트도 있습니다.

참고 URL

<http://leaverou.github.io/prefixfree/>

3-1) CSS3 Rounded Corner

border-radius 속성을 통해서, 해당 요소에 둥근 모서리를 표현할 수 있습니다.

지원 브라우저는 다음과 같습니다.

Property	IE	Chrome	Firefox	Safari	Opera
border-radius	9.0	5.0 4.0 -webkit-	4.0 3.0 -moz-	5.0 3.1 -webkit-	10.5

시작 파일

sample/basic_css3/start/rounded_corner.html

CSS

```
9      <style>
30      .example li {
31          margin-top: 60px;
32          font-size: 0.875em;
33      }
34      .example li:first-child {
35          margin-top: 35px;
36      }
37      .example .rcorner1 {
38          padding: 20px;
39          width: 200px;
40          height: 150px;
41          background: #8ac007;
42          border-radius: 25px;
43      }
44      .example .rcorner2 {
45          margin-top: 10px;
46          padding: 20px;
47          width: 200px;
48          height: 150px;
49          border: 2px solid #8ac007;
50          border-radius: 25px;
51      }
52      .example .rcorner3 {
53          padding: 20px;
```

```

54     width: 200px;
55     height: 150px;
56     background: #8ac007;
57     border-radius: 15px 50px 30px 5px;
58 }
59 .example .rcorner4 {
60     padding: 20px;
61     width: 200px;
62     height: 150px;
63     background: #8ac007;
64     border-radius: 15px 50px 30px;
65 }
66 </style>

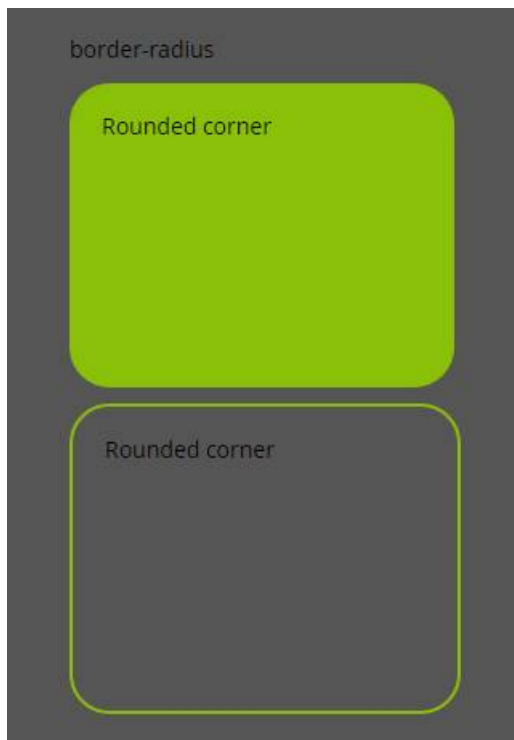
```

border-radius 속성은 border-top-left-radius, border-top-right-radius, border-bottom-right-radius 그리고 border-bottom-left-radius 속성을 간략하게 사용한 CSS 속성입니다.

별도로 각 모서리를 지정할 수 있습니다. 규칙은 다음과 같습니다. :

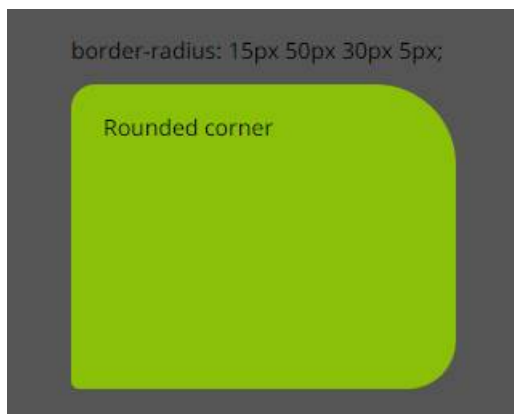
- 4개의 값 : top-left, top-right, bottom-right, bottom-left 값을 지정합니다.
- 3개의 값 : top-left, top-right와 bottom-left, 그리고 bottom-right 값을 지정합니다.
- 2개의 값 : top-left와 bottom-right, top-right와 bottom-left 값을 지정합니다.
- 1개의 값 : 모든 모서리에 균등한 값을 지정합니다.

42행 : `border-radius: 25px;`



`border-radius` 속성 값에 대해 하나의 속성 값을 지정하는 경우, 4개의 모서리에 정해집니다. 4개의 모서리에 둥근 모서리를 25px의 반지름 원으로 설정합니다.

57행 : `border-radius: 15px 50px 30px 5px;`



4개의 모서리에 둥근 모서리 반지름을 개별적으로 설정합니다. 순서대로 top-left, top-right, bottom-right, bottom-left 순입니다.

64행 : `border-radius: 15px 50px 30px;`



마찬가지로 4개의 모서리에 둥근 모서리 반지름을 개별적으로 설정합니다.

단 top-right와 bottom-left 반지름이 같을 경우, 마지막 bottom-left 반지름은 생략합니다. 아래와 같은 표현해도 됩니다.

```
border-radius: 15px 50px 30px 50px;
```

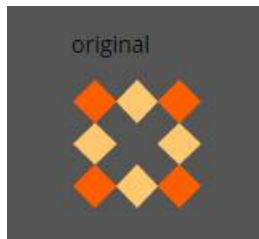
3-2) CSS3 Border Image

border-image 속성은 요소를 감싸는 일반적인 테두리 선 대신에 정해진 이미지로 처리될 수 있습니다.

속성은 3가지 요소로 구성됩니다. :

- 테두리 선으로 사용될 이미지
- 이미지의 어느 부분이 사용될 것인지 지정
- 중간 부분이 반복되거나 뺄어야 할 지를 지정

border-image 속성은 이미지에 대해서 9부분으로 등분화합니다.
원본 이미지는 다음과 같습니다.()



지원 브라우저는 다음과 같습니다.

Property	IE	Chrome	Firefox	Safari	Opera
border-image	11.0	16.0 4.0 -webkit-	15.0 3.5 -moz-	6.0 3.1 -webkit-	15 11.0 -o-

시작 파일

start/basic_css3/start/border_image.html

CSS

```
9      <style>

36      .example li {
37          margin-top: 80px;
38      }
39      .example .description,
```

```

40 .example li {
41     font-size: 0.875em;
42 }
43 .example .border_img1 {
44     padding: 15px;
45     border: 10px solid transparent;
46     -webkit-border-image: url(images/border.png) 30 round;
47     border-image: url(images/border.png) 30 round;
48 }
49 .example .border_img2 {
50     padding: 15px;
51     border: 10px solid transparent;
52     -webkit-border-image: url(images/border.png) 30 stretch;
53     border-image: url(images/border.png) 30 stretch;
54 }
55 .example .border_img3 {
56     padding: 15px;
57     border: 10px solid transparent;
58     -webkit-border-image: url(images/border.png) 50 round;
59     border-image: url(images/border.png) 50 round;
60 }
61 .example .border_img4 {
62     padding: 15px;
63     border: 10px solid transparent;
64     -webkit-border-image: url(images/border.png) 20% round;
65     border-image: url(images/border.png) 20% round;
66 }
67 .example .border_img5 {
68     padding: 15px;
69     border: 10px solid transparent;
70     -webkit-border-image: url(images/border.png) 30% round;
71     border-image: url(images/border.png) 30% round;
72 }
73 </style>

```

45행 : border: 10px solid transparent;

border의 두께는 10px로 두껍게 정하며, 투명한 단선(solid)으로 지정합니다.

46행 : `-webkit-border-image: url(images/border.png) 30 round;`

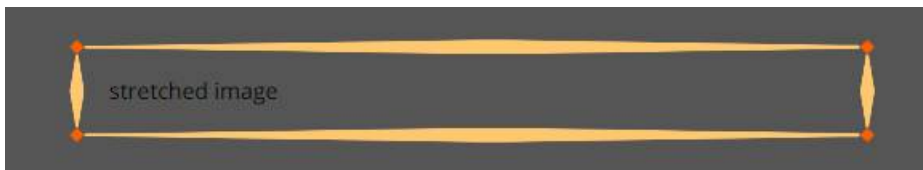


`border-image` 속성은 기본적으로 `border-image-source`, `border-image-slice`, `border-image-width`, `border-image-outset`과 `border-image-repeat` 속성을 간략하게 작성한 것입니다.

접두사 `'-webkit'`은 Chrome 4.0 이하의 버전에 적용되는 속성입니다.
30px의 분할된 영역으로 이미지가 정해집니다.

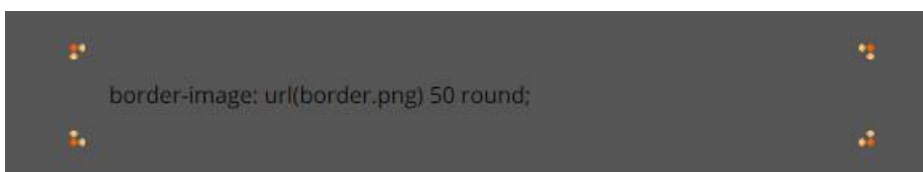


53행 : `border-image: url(images/border.png) 30 stretch;`



`stretch` 속성 값은 이미지의 중간 영역을 테두리를 위해 길게 만듭니다.

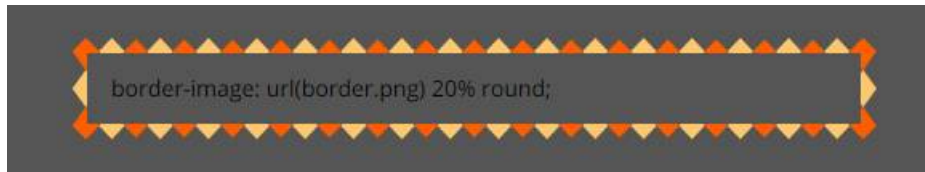
59행 : `border-image: url(images/border.png) 50 round;`



50px의 분할된 영역으로 이미지가 정해집니다.



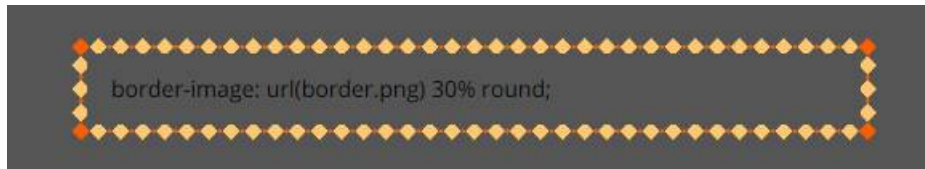
65행 : border-image: url(images/border.png) 20% round;



20%의 분할된 영역으로 이미지가 정해집니다.



71행 : border-image: url(images/border.png) 30% round;



30%의 분할된 영역으로 이미지가 정해집니다.



슬라이스 값을 바꿈으로써 테두리의 느낌을 완전히 바꿀 수 있습니다.

3-3) CSS3 Background

CSS3 요건에는 몇 가지의 새로운 background 속성을 포함합니다.
하나의 요소에 두 가지 이상의 배경 이미지도 포함할 수 있습니다.

또한 다음과 같은 CSS3의 속성도 예제를 통해서 배울 수 있습니다. :

- background-size
- background-origin
- background-clip

지원 브라우저는 다음과 같습니다.

Property	IE	Chrome	Firefox	Safari	Opera
background-image(with multiple backgrounds)	9.0	4.0	3.6	3.1	11.5
background-size	9.0	4.0 1.0 -webkit	4.0 3.6 -moz	4.1 3.0 -webkit	10.5 10.0 -o-
background-origin	9.0	1.0	4.0	3.0	10.5
background-clip	9.0	4.0	4.0	3.0	10.5

시작 파일

sample/basic_css3/start/background1.html

CSS

```
9      <style>
47      .example .background1 {
48          padding: 15px;
49          height: 150px;
50          background: url(images/round.png) right bottom no-repeat, url(images/pattern.gif)
              left top repeat;
51      }
52      .example .background2 {
53          padding: 15px;
54          height: 150px;
55          text-align: right;
56          border: 1px solid #8ac007;
57          background: url(images/round.png);
58          background-repeat: no-repeat;
59      }
60      .example .background3 {
61          padding: 15px;
62          height: 150px;
63          text-align: right;
64          border: 1px solid #8ac007;
65          background: url(images/round.png);
66          background-repeat: no-repeat;
67          background-size: 72px 72px;
68      }
69      .example .background4 {
70          width: 180px;
71          height: 150px;
72          background: url(images/round.png);
73          background-repeat: no-repeat;
74          border: 1px solid #8ac007;
75          background-size: auto;
76      }
77      .example .background5 {
78          width: 180px;
79          height: 150px;
80          background: url(images/round.png);
81          background-repeat: no-repeat;
```

```
82     border: 1px solid #8ac007;
83     background-size: contain;
84 }
85 .example .background6 {
86     width: 180px;
87     height: 150px;
88     background: url(images/round.png);
89     background-repeat: no-repeat;
90     border: 1px solid #8ac007;
91     background-size: cover;
92 }
93 .example .background7 {
94     padding: 15px;
95     height: 150px;
96     background: url(images/round.png) left top no-repeat, url(images/round.png)
           right bottom no-repeat, url(images/pattern.gif) left top repeat;
97     background-size: 50px, 130px, auto;
98 }
99 .example .background8 {
100     padding: 35px;
101     height: 150px;
102     background: url(images/round.png);
103     background-repeat: no-repeat;
104     border: 10px solid #8ac007;
105     background-origin: padding-box;
106 }
107 .example .background9 {
108     padding: 35px;
109     height: 150px;
110     background: url(images/round.png);
111     background-repeat: no-repeat;
112     border: 10px solid #8ac007;
113     background-origin: border-box;
114 }
115 .example .background10 {
116     padding: 35px;
117     height: 150px;
118     background: url(images/round.png);
119     background-repeat: no-repeat;
120     border: 10px solid #8ac007;
121     background-origin: content-box;
122 }
123 .example .background11 {
124     padding: 35px;
```

```

125     height: 150px;
126     background: #fff;
127     border: 10px dotted #8ac007;
128     background-clip: border-box;
129 }
130 .example .background12 {
131     padding: 35px;
132     height: 150px;
133     background: #fff;
134     border: 10px dotted #8ac007;
135     background-clip: padding-box;
136 }
137 .example .background13 {
138     padding: 35px;
139     height: 150px;
140     background: #fff;
141     border: 10px dotted #8ac007;
142     background-clip: content-box;
143 }
144 </style>

```

CSS3을 사용하면 배경 이미지 속성을 통해 하나의 요소에 두 가지 이상의 배경 이미지를 추가할 수 있습니다.

50행 : `background: url(images/round.png) right bottom no-repeat, url(images/pattern.gif) left top repeat;`



coma(,)를 사용해 관련 이미지를 구분합니다. rount1 클래스에 사용된 첫 번째 배경 이미지는 원형의 png 이미지(round.png)이고 두 번째 배경 이미지는 배경에 깔리는 이미지(pattern.gif)입니다.

57행 : `background: url(images/round.png); background-repeat: no-repeat;`



background2 클래스에는 'round.png' 이미지로 반복되지 않도록 지정합니다.

67행 : `background-size: 72px 72px;`



background3 클래스는 배경 이미지의 크기를 가로 크기 72px, 세로 크기를 72px로 정합니다.

75행 : `background-size: auto;`



background-size 속성에 대한 기본 키워드는 auto입니다. 배경 이미지의 크기는 실제 이미지 크기만큼 표시됩니다.

83행 : `background-size: contain;`



`background-size` 속성에 대한 `contain` 키워드는 컨테이너 영역에 맞게 배경 이미지의 폭과 높이 모두 영역 안에 비율적으로 맞춰집니다.

91행 : `background-size: cover;`



`background-size` 속성에 대한 `cover` 키워드는 배경 이미지가 컨테이너에 포함되도록 크기 조정합니다. 특히 배경 이미지의 폭이 영역 안에 맞춰집니다.

97행 : `background-size: 50px, 130px, auto;`



두 개 이상의 이미지의 `background-size` 속성은 콤마(,)로 구분된 배경 이미지 크기 값에 의해 정해집니다.

`background-origin` 속성은 배경 이미지가 위치하는 곳을 지정합니다.

이 속성은 `border-box`, `padding-box`, `content-box` 3가지 다른 속성 값들이 있습니다.

105행 : background-origin: padding-box;



background-origin 속성에 대한 기본 키워드는 padding-box입니다.
padding-box 키워드는 컨테이너의 테두리 영역이 포함된 왼쪽 상단 가장자리로부터 시작됩니다.

113행 : background-origin: border-box;



border-box 키워드는 컨테이너의 왼쪽 상단 가장자리로부터 시작됩니다.

121행 : background-origin: content-box;



content-box 키워드의 배경 이미지는 컨테이너의 테두리 영역과 내부 영역이 포함된 왼쪽 상단 가장자리로부터 시작됩니다.

background-clip 속성은 테두리에 그림 영역을 지정할 수 있습니다.

이 속성은 border-box, padding-box, content-box 3가지 다른 속성 값들이 있습니다.

128행 : background-clip: border-box;



border-box 키워드는 컨테이너 영역 가장자리에 흰 배경색이 지정됩니다.

135행 : background-clip: padding-box;



padding-box 키워드는 컨테이너 영역에서 테두리가 가장자리를 제외한 내부 영역에 흰 배경색이 지정됩니다.

142행 : background-clip: content-box;



content-box 키워드는 컨테이너 영역에서 테두리와 내부 영역을 제외한 내부 영역에 흰 배경색이 지정됩니다.

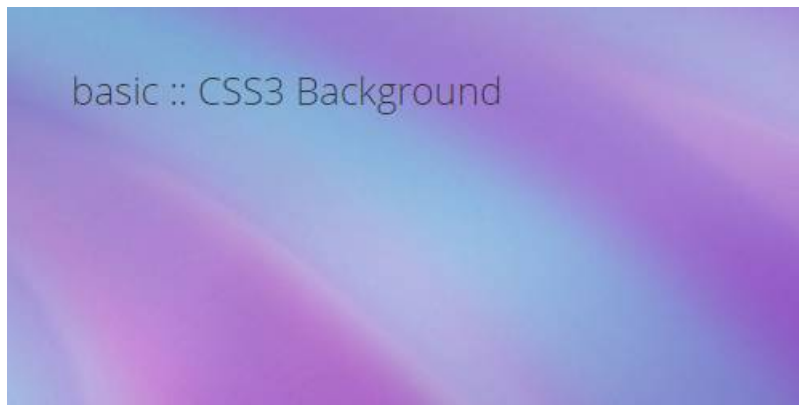
이번에는 웹 사이트의 배경으로 전체 브라우저를 감싸는 배경 이미지를 만들어 봅시다.

전체 브라우저에 대한 배경 이미지를 만들기 위해서는 아래와 같은 요구사항이 필요합니다. :

- 전체 영역을 채울만한 이미지를 준비합니다.
- 이미지의 크기를 필요에 따라 유동적으로 변형합니다.
- 이미지를 페이지 중간 영역에 배치합니다.
- 이미지 때문에 스크롤바가 생성되지 않도록 합니다.

시작 파일

sample/basic_css3/start/background2.html



CSS

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <meta http-equiv="X-UA-Compatible" content="IE=edge">
6 <meta name="viewport" content="width=device-width, initial-scale=1.0,
  minimum-scale=1.0">
7 <title>basic :: CSS 3 Background</title>
8 <link rel="stylesheet"
  href="http://fonts.googleapis.com/css?family=Open+Sans:400,300,300italic,400italic,600,6
  00italic,700,700italic,800,800italic">
9 <style>
10 html {
11     background: url(images/rainbow.jpg) no-repeat center center fixed;
12     background-size: cover;
13 }
14 body {
15     margin: 20px;
16     padding: 20px;
17     line-height: 1;
18     font-family: "Open Sans", sans-serif;
19     font-size: 1em;
20     color: #333;
21 }
22 .title {
23     margin: 0;
24     padding: 0;
25     font-size: 1.5em;
26     font-weight: 300;
27 }
28 .container {
29     margin-top: 35px;
30 }
31 </style>
32 <!--[if lt IE 9]>
33 <script src="http://html5shim.googlecode.com/svn/trunk/html5.js"> </script>
34 <![endif]-->
35 </head>
36
37 <body>
38 <h1 class="title">basic :: CSS 3 Background</h1>
39 <div class="container"> </div>
40 </body>
```

11행 : `background: url(images/rainbow.jpg) no-repeat center center fixed;`

`background` 이미지를 반복시키지 않습니다.(no-repeat)

그리고 가로, 세로 방향으로 중앙 정렬하며(center center), 브라우저를 기준(fixed)으로 정렬합니다.

12행 : `background-size: cover;`

`background-size` 속성에 대한 `cover` 키워드는 컨테이너 영역에 배경 이미지가 포함되도록 크기 조정합니다. 특히 배경 이미지의 폭이 영역 안에 맞춰집니다.

3-4) CSS3 Color

CSS는 색상명, hex 값, 그리고 RGB 색상 값을 지원합니다. :

- RGBA color
- HSL color
- HSLA color
- opacity

RGBA 색상 값은 RGB 색상 값에서 alpha 속성이 추가된 스타일입니다.

```
rgba(red, green, blue, alpha);
```

지원 브라우저는 다음과 같습니다.

Property	IE	Chrome	Firefox	Safari	Opera
RGBA, HSL, and HSLA	9.0	4.0	3.0	3.1	10.1
opacity	9.0	4.0	2.0	3.1	10.1

시작 파일

sample/basic_css3/start/color.html

CSS

```
9      <style>
34      .rgba {
35          margin-top: 35px;
36      }
37      .rgba .rgba1 {
38          background-color: rgba(255,0,0,.3);
39      }
40      .rgba .rgba2 {
41          background-color: rgba(0,255,0,.3);
42      }
43      .rgba .rgba3 {
44          background-color: rgba(0,0,255,.3);
```

```

45     }
46     .hsl {
47         margin-top: 80px;
48     }
49     .hsl .hsl1 {
50         background-color: hsl(120,100%,50%);
51     }
52     .hsl .hsl2 {
53         background-color: hsl(120,100%,75%);
54     }
55     .hsl .hsl3 {
56         background-color: hsl(120,60%,70%);
57     }
58     .hsla {
59         margin-top: 80px;
60     }
61     .hsla .hsla1 {
62         background-color: hsla(120,100%,50%,.3);
63     }
64     .hsla .hsla2 {
65         background-color: hsla(120,100%,75%,.3);
66     }
67     .hsla .hsla3 {
68         background-color: hsla(120,60%,70%,.3);
69     }
70     .opacity {
71         margin-top: 80px;
72     }
73     .opacity .opacity1 {
74         background-color: rgb(255,0,0);
75         opacity: 0.6;
76     }
77     .opacity .opacity2 {
78         background-color: rgb(0,255,0);
79         opacity: 0.6;
80     }
81     .opacity .opacity3 {
82         background-color: rgb(0,0,255);
83         opacity: 0.6;
84     }
85 </style>

```

41행 : background-color: rgba(255,0,0,.3);



rgba1 클래스는 rgb(255,0,0)로 지정되고, alpha 속성은 0.3으로 지정됩니다.

alpha 속성은 0.0에서 1.0까지의 값을 가질 수 있습니다. 0.0은 완전 투명한 상태이고 1.0은 완전 불투명한 상태입니다.

53행 : background-color: hsl(120,100%,50%);

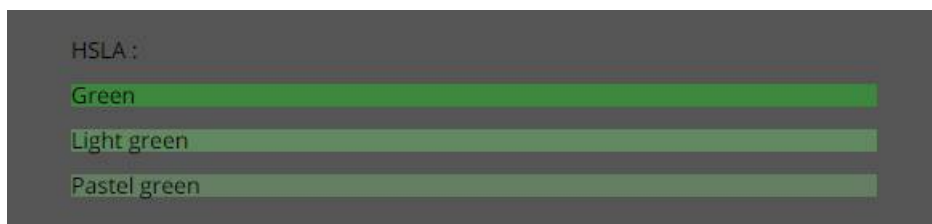


hsl1 클래스는 색상의 값은 120, 채도의 값은 100%로 순색이며 밝기는 50%로 증명도입니다.

HSL은 색상, 채도 및 밝기를 의미합니다.(Hue, Saturation, Lightness) :

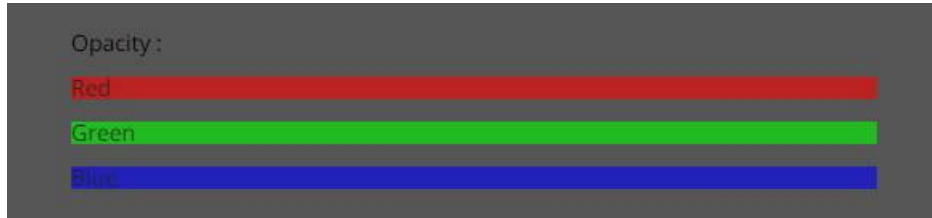
- Hue는 색상환으로 정해집니다.(0 ~ 360)
0이나 360은 red이며, 120은 green이고 240은 blue입니다.
- Saturation은 퍼센트로 값을 표시합니다. 100%는 순색입니다.
- Lightness도 퍼센트로 값을 표시합니다. 0%는 black, 100%는 white입니다.

62행 : background-color: hsla(120,100%,50%,.3);



HSLA 색상 값은 HSL 색상 값에 대해서 alpha 속성이 추가된 스타일입니다. hsla1 클래스는 hsl(120,100,50%) 색상에서 alpha 속성을 0.3으로 지정합니다.

75행 : opacity: 0.6;



opacity 속성은 불투명도에 관련된 속성입니다. opacity 속성 값은 0.0에서 1.0까지 적용할 수 있습니다. opacity1 클래스는 불투명도를 0.6으로 지정합니다.

3-5) CSS3 Gradient

CSS3 그라데이션은 두 개 이상의 지정된 색상 사이의 부드러운 전환을 표시할 수 있습니다.

이전에는 이러한 효과를 위해서 이미지를 준비해야 했지만, CSS3 그라데이션을 이용하면 사이트 사용에 있어서 이미지 다운로드 시간을 줄일 수 있습니다.

CSS3 그라데이션은 2가지 유형으로 정의할 수 있습니다. :

- 직선형 그라디언트(위 / 아래 / 좌 / 우 / 대각선)
- 원형 그라디언트(중심에서 시작)

지원 브라우저는 다음과 같습니다.

Property	IE	Chrome	Firefox	Safari	Opera
linear-gradient	10.0	26.0 10.0 -webkit-	16.0 3.6 -moz-	6.1 5.1 -webkit-	12.1 11.1 -o-
radial-gradient	10.0	26.0 10.0 -webkit-	16.0 3.6 -moz-	6.1 5.1 -webkit-	12.1 11.1 -o-
repeating-linear-gradient	10.0	26.0 10.0 -webkit-	16.0 3.6 -moz-	6.1 5.1 -webkit-	12.1 11.1 -o-
repeating-radial-gradient	10.0	26.0 10.0 -webkit-	16.0 3.6 -moz-	6.1 5.1 -webkit-	12.1 11.1 -o-

직선형 그라데이션을 만들기 위해서는, 적어도 2가지 지정 색상이 필요합니다. 지정 색상 사이에서는 부드러운 전환으로 렌더링이 될 것입니다. 또한 출발점의 지정과 함께 그라디언트 효과에 대한 방향 또는 각도를 설정할 수 있습니다.

용법은 아래의 표와 같습니다.

```
background: linear-gradient(방향, 색상1, 색상2, ... );
```

시작 파일

sample/basic_css3/start/gradient.html

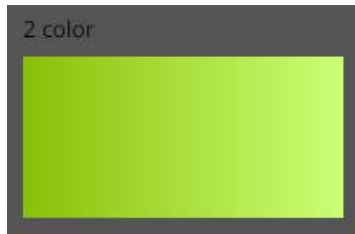
CSS

```
9      <style>

48      .example .gradient1 {
49          background: -webkit-linear-gradient(left, #8ac007, #cbff75);
50          background: -o-linear-gradient(right, #8ac007, #cbff75);
51          background: -moz-linear-gradient(right, #8ac007, #cbff75);
52          background: linear-gradient(to right, #8ac007, #cbff75);
53      }
54      .example .gradient2 {
55          background: -webkit-linear-gradient(left, #8ac007, #cbff75 , #ffb400);
56          background: -o-linear-gradient(right, #8ac007, #cbff75, #ffb400);
57          background: -moz-linear-gradient(right, #8ac007, #cbff75, #ffb400);
58          background: linear-gradient(to right, #8ac007, #cbff75, #ffb400);
59      }
60      .example .gradient3 {
61          background: -webkit-linear-gradient(left top, #8ac007, #cbff75 , #ffb400);
62          background: -o-linear-gradient(bottom right, #8ac007, #cbff75 , #ffb400);
63          background: -moz-linear-gradient(bottom right, #8ac007, #cbff75 , #ffb400);
64          background: linear-gradient(to bottom right, #8ac007, #cbff75 , #ffb400);
65      }
66      .example .gradient4 {
67          background: -webkit-linear-gradient(0deg, #8ac007, #cbff75);
68          background: -o-linear-gradient(0deg, #8ac007, #cbff75);
69          background: -moz-linear-gradient(0deg, #8ac007, #cbff75);
70          background: linear-gradient(0deg, #8ac007, #cbff75);
71      }
72      .example .gradient5 {
73          background: -webkit-linear-gradient(90deg, #8ac007, #cbff75);
74          background: -o-linear-gradient(90deg, #8ac007, #cbff75);
75          background: -moz-linear-gradient(90deg, #8ac007, #cbff75);
76          background: linear-gradient(90deg, #8ac007, #cbff75);
77      }
78      .example .gradient6 {
79          background: -webkit-linear-gradient(180deg, #8ac007, #cbff75);
80          background: -o-linear-gradient(180deg, #8ac007, #cbff75);
81          background: -moz-linear-gradient(180deg, #8ac007, #cbff75);
82          background: linear-gradient(180deg, #8ac007, #cbff75);
83      }
```

```
84 .example .gradient7 {
85     background: -webkit-linear-gradient(-90deg, #8ac007, #cbff75);
86     background: -o-linear-gradient(-90deg, #8ac007, #cbff75);
87     background: -moz-linear-gradient(-90deg, #8ac007, #cbff75);
88     background: linear-gradient(-90deg, #8ac007, #cbff75);
89 }
90 .example .gradient8 {
91     background: -webkit-linear-gradient(left, #8ac007 10%, #cbff75 85% , #ffb400
92                                     90%);
93     background: -o-linear-gradient(right, #8ac007 10%, #cbff75 85% , #ffb400 90%);
94     background: -moz-linear-gradient(right, #8ac007 10%, #cbff75 85% , #ffb400
95                                     90%);
96     background: linear-gradient(to right, #8ac007 10%, #cbff75 85% , #ffb400 90%);
97 }
98 .example .gradient9 {
99     background: -webkit-repeating-linear-gradient(left, #8ac007, #cbff75 10%, #ffb400
100                                     20%);
101     background: -o-repeating-linear-gradient(right, #8ac007, #cbff75 10%, #ffb400
102                                     20%);
103     background: -moz-repeating-linear-gradient(right, #8ac007, #cbff75 10%, #ffb400
104                                     20%);
105     background: repeating-linear-gradient(to right, #8ac007, #cbff75 10%, #ffb400
106                                     20%);
107 }
108 .example .gradient10 {
109     background: -webkit-radial-gradient(#8ac007, #cbff75);
110     background: -o-radial-gradient(#8ac007, #cbff75);
111     background: -moz-radial-gradient(#8ac007, #cbff75);
112     background: radial-gradient(#8ac007, #cbff75);
113 }
114 .example .gradient11 {
115     background: -webkit-radial-gradient(circle, #8ac007, #cbff75);
116     background: -o-radial-gradient(circle, #8ac007, #cbff75);
117     background: -moz-radial-gradient(circle, #8ac007, #cbff75);
118     background: radial-gradient(circle, #8ac007, #cbff75);
119 }
120 .example .gradient12 {
121     background: -webkit-radial-gradient(ellipse, #8ac007, #cbff75);
122     background: -o-radial-gradient(ellipse, #8ac007, #cbff75);
123     background: -moz-radial-gradient(ellipse, #8ac007, #cbff75);
124     background: radial-gradient(ellipse, #8ac007, #cbff75);
125 }
126 </style>
```

49행 : background: -webkit-linear-gradient(left, #8ac007, #cbff75);



최소한 두 개의 색상 포인트를 지정하여 그라데이션을 만들 수 있습니다.

'-webkit' 접두사는 Safari 브라우저 5.1 버전에서 6.0 버전 사이에서 그라데이션을 지정합니다.

50행 : background: -o-linear-gradient(right, #8ac007, #cbff75);

'-o-' 접두사는 Opera 브라우저 11.1 버전에서 12.0 버전 사이에서 그라데이션을 지정합니다.

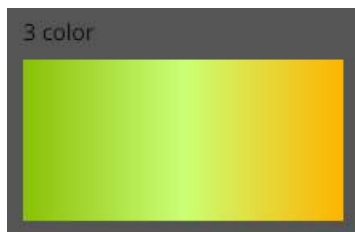
51행 : background: -moz-linear-gradient(right, #8ac007, #cbff75);

'-moz-' 접두사는 Firefox 브라우저 3.6 버전에서 15 버전 사이에서 그라데이션을 지정합니다.

52행 : background: linear-gradient(to right, #8ac007, #cbff75);

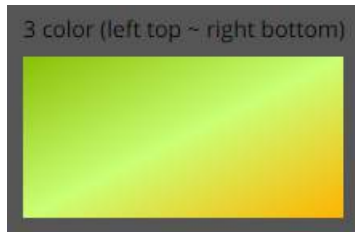
접두사가 없는 경우가 상위 브라우저에 맞춰진 그라데이션입니다. 유의할 점은 접두사가 없는 CSS 속성이 제일 마지막에 배치되어야 한다는 것입니다.

58행 : background: linear-gradient(to right, #8ac007, #cbff75, #ffb400);



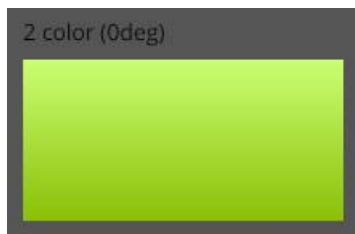
세 개 이상의 색상 포인트를 지정하여 다양한 효과의 그라데이션을 만들 수 있습니다. gradient2 클래스는 3개의 색상 포인트를 지정하고 있습니다.

64행 : background: linear-gradient(to bottom right, #8ac007, #cbff75 , #ffb400);



두 대각선의 수평 및 수직 시작 위치를 지정하여 그라데이션을 만들 수 있습니다.
gradient3 클래스의 그라데이션 방향은 하단(bottom)에서 오른쪽(right)으로 지정하고 있습니다.

70행 : background: linear-gradient(0deg, #8ac007, #cbff75);



각도를 이용해서 그라데이션의 방향을 정해줄 수도 있습니다. gradient4 클래스는 0도 방향으로, 즉 수직 방향으로 그라데이션을 지정하고 있습니다.

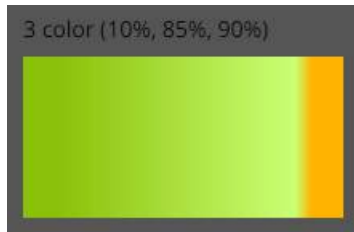
```
background: linear-gradient(각도, 색상1, 색상2, ... );
```

76행 : background: linear-gradient(90deg, #8ac007, #cbff75);



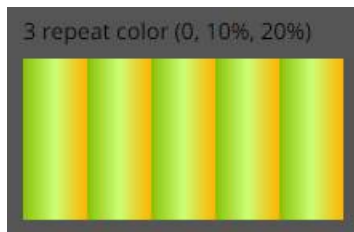
gradient5 클래스는 그라데이션의 방향이 90도 방향으로 지정되어 수평 방향으로 그라데이션을 지정하고 있습니다.

94행 : background: linear-gradient(to right, #8ac007 10%, #cbff75 85%, #ffb400 90%);



gradient8 클래스는 3개의 그레데이션 포인트에 범위를 퍼센트로 지정하는 방식입니다.

100행 : background: repeating-linear-gradient(to right, #8ac007, #cbff75 10%, #ffb400 20%);



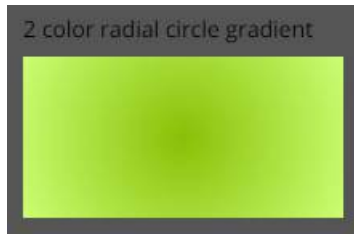
gradient9 클래스는 반복적인 그레디언트를 지정하고 있습니다. repeating-linear-gradient 속성 값으로 반복적인 그레데이션을 만들 수 있습니다.

106행 : background: radial-gradient(#8ac007, #cbff75);



gradient10 클래스는 타원형 그레데이션으로 모양에 대한 인자는 생략되었으며, 생략되었을 경우 지정되는 기본적인 값은 ellipse(타원형)입니다.

112행 : background: radial-gradient(circle, #8ac007, #cbff75);



gradient11 클래스의 그라데이션 모양은 원형(circle)입니다.

114행 : background: radial-gradient(ellipse, #8ac007, #cbff75);



gradient12 클래스의 그라데이션 모양은 타원형(ellipse)입니다. 타원형 모양이 기본 값입니다.

3-6) CSS3 Shadow

3-6-1) CSS3 Text Shadow

CSS3의 `text-shadow` 속성은 텍스트에 그림자를 추가할 수 있습니다. :

- 일반적인 그림자 효과
- 그림자에 색상 지정
- 그림자에 블러 효과 지정
- 흰 색 텍스트에, 검정색 블러 그림자 효과 지정
- 그림자에 네온 효과 지정

지원 브라우저는 다음과 같습니다.

Property	IE	Chrome	Firefox	Safari	Opera
text-shadow	10.0	4.0	3.5	4.0	9.5

시작 파일

sample/basic_css3/start/text_shadow.html

CSS

```
9      <style>
44     .example .text_shadow1 {
45         text-shadow: 2px 2px;
46     }
47     .example .text_shadow2 {
48         text-shadow: 2px 2px #333;
49     }
50     .example .text_shadow3 {
51         text-shadow: 2px 2px 5px #333;
52     }
53     .example .text_shadow4 {
54         text-shadow: 0 0 6px #8ac007;
55     }
56     .example .text_shadow5 {
57         text-shadow: 1px 1px 2px #000, 0 0 25px #8ac007, 0 0 5px #cbff75;
58     }
59     </style>
```


45행 : text-shadow: 2px 2px;

```
text-shadow: 2px 2px;
```

text_shadow1 클래스는 가로로 2px, 세로로 2px 떨어진, 검은 색의 그림자로 지정합니다.

text-shadow: (가로 그림자 위치) (세로 그림자 위치) (퍼짐 정도) (그림자 색상);

48행 : text-shadow: 2px 2px #333;

```
text-shadow: 2px 2px #333;
```

text_shadow2 클래스는 가로로 2px, 세로로 2px 떨어진 #555 색의 그림자로 지정합니다.

51행 : text-shadow: 2px 2px 5px #333;

```
text-shadow: 2px 2px 5px #333;
```

text_shadow3 클래스의 그림자 퍼짐 정도는 5px입니다.

54행 : text-shadow: 0 0 5px #8ac007;

```
text-shadow: 0 0 5px #8ac007;
```

text_shadow4 클래스와 같은 속성 값들의 조정으로 네온 효과를 낼 수 있습니다.

57행 : text-shadow: 1px 1px 2px #000, 0 0 25px #8ac007, 0 0 5px #cbff75;

```
text-shadow: 1px 1px 2px #000, 0 0 25px #8ac007, 0 0 5px #cbff75;
```

텍스트에 하나 이상의 그림자를 추가하기 위해서는 콤마(,)를 분리자로 사용합니다.

3-6-2) CSS3 Box Shadow

CSS3의 box-shadow 속성은 text-shadow와 같은 방법으로 덩어리 요소에 그림자를 추가할 수 있습니다.

단순한 사용으로는, 가로 그림자와 세로 그림자를 지정하는 것입니다. :

- 단순한 그림자 효과
- 그림자에 색상 지정
- 그림자에 블러 효과 지정

지원 브라우저는 다음과 같습니다.

Property	IE	Chrome	Firefox	Safari	Opera
box-shadow	9.0	10.0 4.0 -webkit-	4.0 3.5 -moz-	5.1 3.1 -webkit-	10.5

시작 파일

sample/basic_css3/start/box_shadow.html

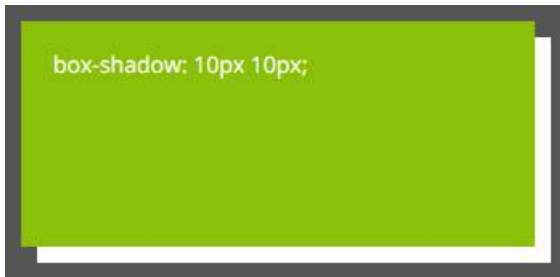
CSS

```
9      <style>
48      .example .box_shadow1 {
49          box-shadow: 10px 10px;
50      }
51      .example .box_shadow2 {
52          box-shadow: 10px 10px #333;
53      }
54      .example .box_shadow3 {
55          box-shadow: 10px 10px 5px #333;
56      }
      </style>
```

box-shadow 속성은 text-shadow 속성과 마찬가지로 속성 값을 지정할 수 있습니다.

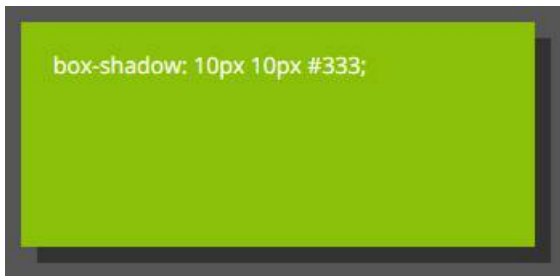
box-shadow: (가로 그림자 위치) (세로 그림자 위치) (퍼짐 정도) (그림자 색상);

48행 : `box-shadow: 10px 10px;`



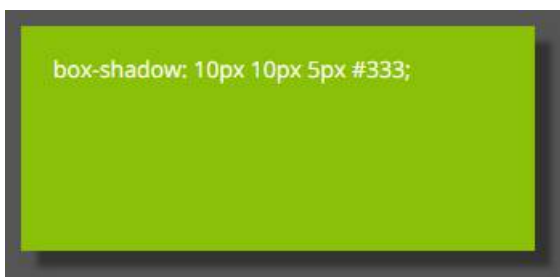
box_shadow1 클래스는 가로로 10px, 세로로 10px 떨어진 흰색의 그림자로 지정합니다.
그림자 색상이 정해지지 않을 때 기본 그림자 색상은 흰색으로 정해집니다.

51행 : `box-shadow: 10px 10px #333;`



box_shadow2 클래스는 가로로 10px, 세로로 10px 떨어진 #333 색의 그림자로 지정합니다.

54행 : `box-shadow: 10px 10px 5px #333;`



box_shadow3 클래스의 그림자 퍼짐 정도는 5px입니다.

3-7) CSS3 Text

CSS3에서는 몇 가지 새로운 텍스트 기능이 포함되어 있습니다. :

- text-overflow
- word-wrap
- word-break

지원 브라우저는 다음과 같습니다.

Property	IE	Chrome	Firefox	Safari	Opera
text-overflow	6.0	4.0	7.0	3.1	11.0 9.0 -o-
word-wrap	5.5	23.0	3.5	6.1	12.1
word-break	5.5	4.0	15.0	3.1	15.0

시작 파일

sample/basic_css3/start/text.html

CSS

```
9      <style>

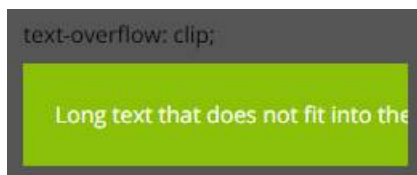
52     .example .text1 {
53         overflow: hidden;
54         white-space: nowrap;
55         text-overflow: clip;
56     }
57     .example .text2,
58     .example .text3 {
59         overflow: hidden;
60         white-space: nowrap;
61         text-overflow: ellipsis;
62     }
63     .example .text3:hover {
64         width: 260px;
65         overflow: visible;
66         text-overflow: inherit;
67     }
```

```

68 .example .text4 {
69     word-wrap: initial;
70 }
71 .example .text5 {
72     word-wrap: break-word;
73 }
74 .example .text6 {
75     word-break: break-all;
76 }
77 </style>

```

53행 : `overflow: hidden; white-space: nowrap; text-overflow: clip;`

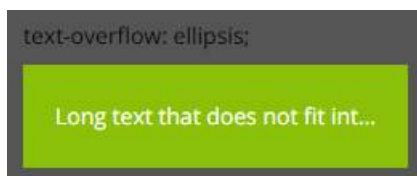


`text-overflow` 속성은 보이지 않을 콘텐츠를 어떻게 처리할 지를 결정합니다.

`text1` 클래스를 요목 조목 따져보면 다음과 같습니다. :

- `overflow: hidden;` : 영역을 넘치는 텍스트는 가려줍니다.
- `white-space: nowrap;` : 영역을 넘치는 텍스트에 대해서는 줄 바꿈을 하지 않습니다.
- `text-overflow: clip;` : `text-overflow: clip;` 방식은 영역을 넘치는 텍스트를 그냥 잘라줍니다.
이는 기본 값입니다.

61행 : `text-overflow: ellipsis;`

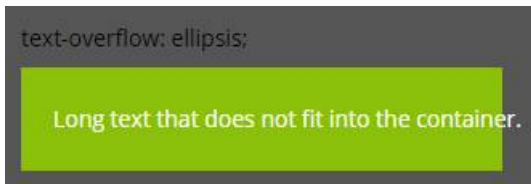


`text2` 클래스에 적용된 `text-overflow: ellipsis;` 방식은 외부로 벗어나는 텍스트는 말 줄임(...) 처리해 줍니다.

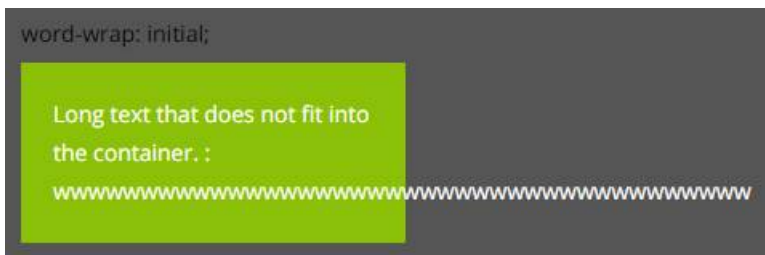
64행 : width: 260px; overflow: visible; text-overflow: inherit;



특별한 이벤트(hover)로 텍스트를 보이도록 처리할 수 있습니다.
아래의 그림은 text3 클래스에 마우스를 올렸을 경우입니다.



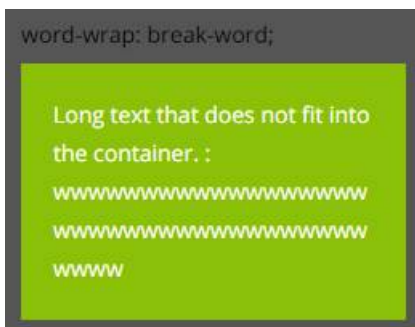
69행 : word-wrap: initial;



word-wrap 속성은 영역에 포함되기 힘든 긴 단어를 잘라내기 할 지, 다음 라인으로 처리할 지를 결정합니다.

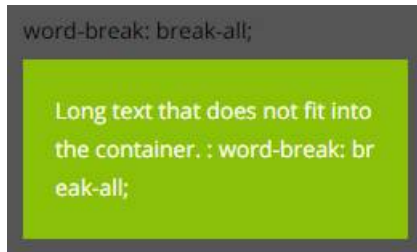
word-wrap: initial; 방식은 기본 속성 값으로써 단어를 기준으로 개행 처리가 됩니다. 이는 normal 키워드도 마찬가지입니다.

72행 : word-wrap: break-word;

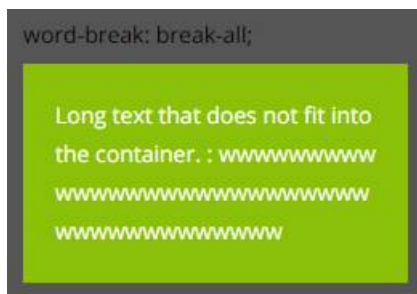


word-wrap: break-word; 방식은 영역을 기준으로 단어를 잘라내기 합니다.

73행 : .example .text6 {word-break: break-all;}



word-break: break-all; 방식은 모든 글자에서 분리됩니다. 따라서 단어의 의미가 모호해질 수 있습니다.



결과적으로 영역을 기준으로 단어를 잘라내기 위한 방식은 word-wrap: break-word;나 word-break: break-all; 속성을 사용하면 됩니다.

3-8) CSS3 Web Font

웹 폰트는 웹 디자이너들이 사용자의 컴퓨터에 설치되지 않은 폰트를 적용하기 위해 사용합니다. 폰트를 사용하기 위해 웹 서버에 폰트 파일을 포함시키는 것입니다.

웹 폰트를 적용하기 위해서는 CSS3의 @font-face 규칙에 의해 정의됩니다.

지원 브라우저는 다음과 같습니다.

Font Format	IE	Chrome	Firefox	Safari	Opera
TTF / OTF	9.0*	4.0	3.5	3.1	10.0
WOFF	9.0	5.0	3.6	5.1	11.1
WOFF2	Not supported	36.0	35.0*	Not supported	26.0
SVG	Not supported	4.0	Not supported	3.2	9.0
EOT	6.0	Not supported	Not supported	Not supported	Not supported

Internet Explorer 9에서 TTF/OTF 폰트 포맷은 'installable'로 설정해야만 글꼴을 활용할 수 있습니다. Firefox에서 WOFF2 폰트 포맷은 기본적으로 지원하지 않지만 WOFF2 사용하는 옵션을 설정하면 글꼴을 활용할 수 있습니다.

3-8-1) Fonts Format

TrueType Fonts(TTF) :

TrueType은 1980년대 후반에 개발된 글꼴 표준입니다. TrueType은 Mac OS와 Window 운영 체제 모두 사용할 수 있는 가장 보편화된 포맷입니다.

OpenType Fonts(OTF) :

OpenType은 확장 컴퓨터 글꼴 형식입니다. TrueType을 기반으로, Microsoft의 등록 상표입니다. OpenType 글꼴은 주요 컴퓨터 플랫폼에서 현재 일반적으로 사용됩니다.

The Web Open Font Format(WOFF) :

WOFF는 웹 페이지에 사용되는 글꼴 형식입니다. 2009년에 개발되었으며, 현재 W3C에서 추천하는 포맷입니다. WOFF는 기본적으로 압축 및 추가 메타 데이터를 포함하는 압축 오픈 타입 또는 트루 타입입니다.

The Web Open Font Format(WOFF 2.0) :

WOFF 1.0보다 더 압축 효율이 좋은 TrueType이면서 OpenType 글꼴입니다.

SVG Fonts/Shapes :

텍스트를 표시할 때 SVG 글리프(glyph)로서 사용될 수 있도록 합니다. 글리프는 '자체'를 의미하며, '문자의 구체적인 모양, 디자인, 표현'을 가리킵니다.

Embedded OpenType Fonts(EOT) :

EOT 글꼴은 웹 페이지에 포함된 글꼴로 사용하기 위해 Microsoft가 설계한 오픈 타입 글꼴의 최적화된 형태입니다.

3-8-2) Fonts의 활용

CSS3의 @font-face 규칙에서 먼저 글꼴 이름을 정의해야 합니다.

주의할 점은, 항상 폰트 URL에 대한 소문자를 사용합니다. 대문자는 Internet Explorer에서는 예기치 않은 결과가 생길 수 있기 때문입니다. 또한 웹 폰트는 다운로드용 폰트로의 변환이기 때문에 저작권에 유의해야 합니다.

시작 파일

sample/basic_css3/start/web_font.html



CSS

```
9      <style>
10     @font-face {
11         font-family: "sansation_light";
12         src: url(fonts/sansation_light.woff);
13     }

51     .example .fonts1 {
52         font-family: "sansation_light";
53     }
54     </style>
```

11행 : font-family: "sansation_light";

CSS3에서는 @font-face 규칙으로 먼저 글꼴 이름을 정해야 합니다.

12행 : src: url(fonts/sansation_light.woff);

사용할 글꼴의 경로입니다.

52행 : font-family: "sansation_light";

fonts1 클래스는 'sansation_light'라는 이름의 폰트를 사용하고 있습니다. 웹 사용자들은 해당 폰트가 시스템에 설치되어 있지 않더라도 'sansation_light.woff' 폰트를 다운로드하여 사용할 수 있는 것입니다.

사실 이제까지 예제에서 구글 웹 폰트를 외부 경로(CDN)를 통해서 사용해 왔습니다.

구글 웹 폰트의 경로입니다.

```
<link rel="stylesheet"
href="http://fonts.googleapis.com/css?family=Open+Sans:400,300,300italic,400italic,600,600italic,
700,700italic,800,800italic">
```

19행 : font-family: "Open Sans", sans-serif;

이처럼 외부 경로가 로딩이 된다면, 'Open Sans' 폰트를 사용할 수 있습니다.

3-9) CSS3 2D Transform

CSS3 transform 속성은 해당 요소에 대해서 2D 또는 3D 변환이 가능합니다. 즉 요소를 변환, 회전, 크기 조절 및 왜곡시킬 수 있도록 합니다.

2D 변환 방법에 대한 내용입니다. :

- translate()
- rotate()
- scale()
- skewX()
- skewY()
- matrix()

지원 브라우저는 다음과 같습니다.

Property	IE	Chrome	Firefox	Safari	Opera
transform	10.0 9.0 -ms-	36.0 4.0 -webkit-	16.0 3.5 -moz-	3.2 -webkit-	23.0 15.0 -webkit- 12.1 10.5 -o-
transform-origin	10.0 9.0 -ms-	36.0 4.0 -webkit-	16.0 3.5 -moz-	3.2 -webkit-	23.0 15.0 -webkit- 12.1 10.5 -o-

시작 파일

sample/basic_css3/start/2d_transform.html

CSS

```
9      <style>
47      .example .translate {
48          -ms-transform: translate(100px,20px);
49          -webkit-transform: translate(100px,20px);
50          transform: translate(100px,20px);
51      }
52      .example .rotate1 {
53          -ms-transform: rotate(20deg);
54          -webkit-transform: rotate(20deg);
55          transform: rotate(20deg);
56      }
57      .example .rotate2 {
58          -ms-transform: rotate(-20deg);
59          -webkit-transform: rotate(-20deg);
60          transform: rotate(-20deg);
61      }
62      .example .scale {
63          -ms-transform: scale(1,2);
64          -webkit-transform: scale(.1,2);
65          transform: scale(1,2);
66      }
67      .example .skewx {
68          -ms-transform: skewX(20deg);
69          -webkit-transform: skewX(20deg);
70          transform: skewX(20deg);
71      }
72      .example .skewy {
73          -ms-transform: skewY(20deg);
74          -webkit-transform: skewY(20deg);
75          transform: skewY(20deg);
76      }
77      .example .skew1 {
78          -ms-transform: skew(20deg);
79          -webkit-transform: skew(20deg);
80          transform: skew(20deg);
81      }
82      .example .skew2 {
```

```

83     -ms-transform: skew(20deg,10deg);
84     -webkit-transform: skew(20deg,10deg);
85     transform: skew(20deg,10deg);
86 }
87 .example .matrix1 {
88     -ms-transform: matrix(1,-.3,0,1,0,0);
89     -webkit-transform: matrix(1,-.3,0,1,0,0);
90     transform: matrix(1,-.3,0,1,0,0);
91 }
92 .example .matrix2 {
93     -ms-transform: matrix(1,0,.5,1,150,0);
94     -webkit-transform: matrix(1,0,.5,1,150,0);
95     transform: matrix(1,0,.5,1,150,0);
96 }
97 </style>

```

50행 : transform: translate(100px,20px);



translate 클래스는 가로로는 100px 세로로는 20px 이동합니다.

55행 : transform: rotate(20deg);



rotate() 키워드는 요소를 시계 방향이나 시계 반대 방향으로 주어진 각도만큼 회전시킵니다.
rotate1 클래스는 시계 방향으로 20° 회전합니다.

60행 : transform: rotate(-20deg);



rotate2 클래스는 시계 반대 방향으로 20° 회전합니다. 음수 값을 사용하면 요소를 시계 반대 방향으로 회전합니다.

65행 : transform: scale(1,2);



scale() 키워드는 요소 크기를 조절합니다.(width, height)
scale 클래스는 세로로 2배 확대됩니다.

70행 : transform: skewX(20deg);



skewX() 키워드는 요소를 x 축을 기준으로 주어진 각도만큼 기울입니다.
skewx 클래스는 x 축을 기준으로 20° 기울입니다.

75행 : transform: skewY(20deg);



skewY() 키워드는 y 축을 기준으로 주어진 각도만큼 기울입니다.
skewY 클래스는 y 축을 기준으로 20° 기울입니다.

80행 : transform: skew(20deg);



skew() 키워드는 x 축과 y 축을 기준으로 주어진 각도만큼 기울입니다. 두 번째 인자가 지정되어 있지 않다면, 0°로 자동으로 정해집니다.
skew1 클래스는 x 축을 기준으로 20° 기울입니다.

85행 : transform: skew(20deg,10deg);



skew2 클래스는 x 축을 기준으로 20° 기울이고, y 축을 기준으로 10° 기울입니다.

90행 : transform: matrix(1,-.3,0,1,0,0);



matrix() 키워드는 2D 변환 기능을 하나의 방식으로 결합합니다. matrix() 키워드는 6개의 기능을 포함한 인자가 필요합니다.

인자에 대한 부분은 다음과 같습니다.

```
matrix(scaleX(), skewY(), skewX(), scaleY(), translateX(), translateY());
```

matrix에 대해 테스트해봅시다.

아래의 예시 스타일은 matrix1 클래스에 적용한 결과입니다.

```
.example .matrix1 {  
  -ms-transform: matrix(1, 0, 0, 1, 0, 0);  
  -webkit-transform: matrix(1, 0, 0, 1, 0, 0);  
  transform: matrix(1, 0, 0, 1, 0, 0);  
}
```

변화가 생기지 않았습니다. scaleX(), scaleY() 값이 1이면 정해진 가로 세로 크기이기 때문입니다.

```
.example .matrix1 {  
  -ms-transform: matrix(1, 0, 0, 1, 50, 50);  
  -webkit-transform: matrix(1, 0, 0, 1, 50, 50);  
  transform: matrix(1, 0, 0, 1, 50, 50);  
}
```

translateX(), translateY() 좌표를 50px씩 이동시켜줍니다.

```
.example .matrix1 {  
  -ms-transform: matrix(1, .5, 0, 1, 50, 50);  
  -webkit-transform: matrix(1, .5, 0, 1, 50, 50);  
  transform: matrix(1, .5, 0, 1, 50, 50);  
}
```

skewY()에 0.5를 적용합니다.

matrix() 키워드에 대한 자세한 설명을 위해서, 아래의 URL을 공유합니다.

참고 URL

<http://peterned.home.xs4all.nl/matrices/#0.923,0,0,0.845,0,0>

3-10) CSS3 3D Transform

CSS3은 해당 요소에 대해서 2D 또는 3D 변환이 가능합니다.

3D 변환 방법에 대한 내용입니다. :

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

지원 브라우저는 다음과 같습니다.

Property	IE	Chrome	Firefox	Safari	Opera
transform	10.0	36.0 12.0 -webkit-	16.0 10.0 -moz-	4.0 -webkit-	23.0 15.0 -webkit-
transform-origin	10.0	36.0 12.0 -webkit-	16.0 10.0 -moz-	4.0 -webkit-	23.0 15.0 -webkit-
transform-style	11.0	36.0 12.0 -webkit-	16.0 10.0 -moz-	4.0 -webkit-	23.0 15.0 -webkit-
perspective	10.0	36.0 12.0 -webkit-	16.0 10.0 -moz-	4.0 -webkit-	23.0 15.0 -webkit-
perspective-origin	10.0	36.0 12.0 -webkit-	16.0 10.0 -moz-	4.0 -webkit-	23.0 15.0 -webkit-
backface-visibility	10.0	36.0 12.0 -webkit-	16.0 10.0 -moz-	4.0 -webkit-	23.0 15.0 -webkit-

시작 파일

sample/basic_css3/start/3d_transform.html

CSS

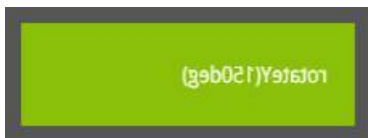
```
9      <style>
47      .example .rotatex {
48          -webkit-transform: rotateX(150deg);
49          transform: rotateX(150deg);
50      }
51      .example .rotatey {
52          -webkit-transform: rotateY(150deg);
53          transform: rotateY(150deg);
54      }
55      .example .rotatez {
56          -webkit-transform: rotateZ(10deg);
57          transform: rotateZ(10deg);
58      }
59      </style>
```

49행 : transform: rotateX(150deg);



rotateX() 명령어는 3D x 축으로 주어진 각도만큼 3D 회전합니다. rotatex 클래스는 3D x 축으로 150° 회전합니다.

53행 : transform: rotateY(150deg);



rotateY() 명령어는 3D y 축으로 주어진 각도만큼 3D 회전합니다. rotatey 클래스는 3D y 축으로 150° 회전합니다.

57행 : transform: rotateZ(10deg);



rotateZ() 메서드는 3D z 축으로 주어진 각도만큼 3D 회전합니다. rotatez 클래스는 3D z 축으로 10° 회전합니다.

3-11) CSS3 Transition

CSS3 Transitions 속성은 해당 요소에 대한 시간에 따른 변환을 가능하게 합니다.
적용 시간이 지정되어 있지 않다면, 기본 값은 0이기 때문에 변환되는 과정이 생략됩니다.

변환 효과를 만들기 위해서는 2가지 요소가 반드시 있어야 합니다. :

- 효과를 적용할 CSS 속성
- 효과에 대한 적용 시간

지원 브라우저는 다음과 같습니다.

Property	IE	Chrome	Firefox	Safari	Opera
transition	10.0	26.0 4.0 -webkit-	16.0 4.0 -moz-	6.1 3.1 -webkit-	12.1 10.5 -o-
transition-delay	10.0	26.0 4.0 -webkit-	16.0 4.0 -moz-	6.1 3.1 -webkit-	12.1 10.5 -o-
transition-duration	10.0	26.0 4.0 -webkit-	16.0 4.0 -moz-	6.1 3.1 -webkit-	12.1 10.5 -o-
transition-property	10.0	26.0 4.0 -webkit-	16.0 4.0 -moz-	6.1 3.1 -webkit-	12.1 10.5 -o-
transition-timing-function	10.0	26.0 4.0 -webkit-	16.0 4.0 -moz-	6.1 3.1 -webkit-	12.1 10.5 -o-

시작 파일

sample/basic_css3/start/transition.html

CSS

```
9      <style>
49      /* transitions */
50      .example .transition1 {
51          cursor: pointer;
52          -webkit-transition: width 2s;
53          transition: width 2s;
```

```
54     }
55     .example .transition1:hover {
56         width: 300px;
57     }
58     .example .transition2 {
59         cursor: pointer;
60         -webkit-transition: width 2s, height 4s;
61         transition: width 2s, height 4s;
62     }
63     .example .transition2:hover {
64         width: 300px;
65         height: 200px;
66     }
67
68     /* speed curve */
69     .example .speed_curve {
70         font-family: "Open Sans", sans-serif;
71         cursor: pointer;
72         -webkit-transition: width 2s;
73         transition: width 2s;
74     }
75     .example .linear {
76         -webkit-transition-timing-function: linear;
77         transition-timing-function: linear;
78     }
79     .example .ease {
80         -webkit-transition-timing-function: ease;
81         transition-timing-function: ease;
82     }
83     .example .ease_in {
84         -webkit-transition-timing-function: ease-in;
85         transition-timing-function: ease-in;
86     }
87     .example .ease_out {
88         -webkit-transition-timing-function: ease-out;
89         transition-timing-function: ease-out;
90     }
91     .example .ease_in_out {
92         -webkit-transition-timing-function: ease-in-out;
93         transition-timing-function: ease-in-out;
94     }
95     .example .speed_curve:hover {
96         width: 300px;
97     }
```

```

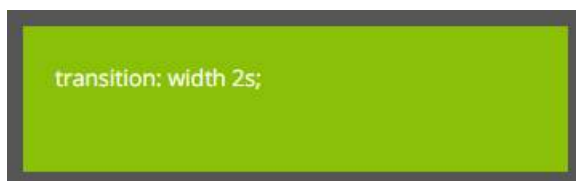
98
99  /* delay */
100 .example .delay {
101     -webkit-transition: width 3s;
102     -webkit-transition-delay: 1s;
103     transition: width 3s;
104     transition-delay: 1s;
105 }
106 .example .delay:hover {
107     width: 300px;
108 }
109 .example .transform_transition {
110     -webkit-transition: -webkit-transform 2s;
111     transition: transform 2s;
112 }
113 .example .transform_transition:hover {
114     -webkit-transform: rotate(180deg);
115     transform: rotate(180deg);
116 }
117 </style>

```

53행 : `transition: width 2s;`

transition1 클래스에 대해서 가로 크기에 대한 변환이 지정됩니다. 변환되는 시간은 2초(2s)입니다.

56행 : `width: 300px;`

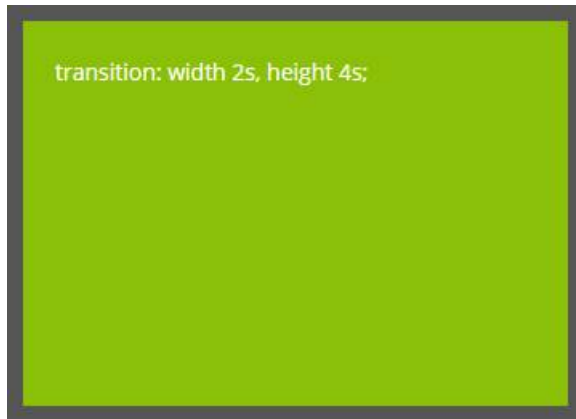


transition1 클래스에 마우스를 올리면 가로 크기가 300px로 변환됩니다. 원래의 크기는 200px이었습니다.

61행 : `transition: width 2s, height 4s;`

transition2 클래스에 대해서 가로와 세로의 크기에 대한 변환이 지정됩니다.

64행 : width: 300px; height: 200px;



transition2 클래스에 마우스를 올리면 가로 크기 300px, 세로 크기 200px로 변환됩니다.

transition-timing-function 속성은 아래와 같은 속성 값을 가질 수가 있습니다. :

- ease : 느리게 시작해서 빨라지고, 서서히 끝나는 변환 효과를 지정합니다. 기본 값입니다.
- linear : 처음부터 끝까지 같은 속도로 변환 효과를 지정합니다.
- ease-in : 느리게 시작하는 변환 효과를 지정합니다.
- ease-out : 느리게 끝나는 변환 효과를 지정합니다.
- ease-in-out : 느린 시작과 끝으로 변환 효과를 지정합니다.
- cubic-bezier(n,n,n,n) : cubic-bezier() 키워드를 이용해서 자신만의 변환 값을 지정합니다.

75행 : transition-timing-function: linear;

linear 클래스는 transition-timing-function 값을 linear로 지정합니다.

linear 값은 처음부터 끝까지 같은 속도로 변환을 지속합니다.

또한 transition-delay 속성은 변환 효과에 대해서 지정된 시간동안 지연시킬 수 있습니다.

104행 : transition-delay: 1s;

delay 클래스는 1초간 지연되었다가 변환 효과가 발생합니다.

마우스를 올리면 가로 크기가 300px로 변형됩니다.

CSS3 transition 속성은 width나 height와 같은 속성과 마찬가지로, transform과 같은 변형도 변환시킬 수 있습니다.

111행 : transition: transform 2s;

transform_transition 클래스는 transform 속성에 대하여 2초 동안 변환시킵니다.

114행 : transform: rotate(180deg);

transform_transition 클래스에 마우스를 올리면 180° 회전합니다.

CSS3 transition 속성은 하나하나를 따로 지정할 수 있습니다.

```
.elements {
  width: 100px;
  height: 100px;

  /* Safari 3.1 to 6.0 */
  -webkit-transition-property: width;
  -webkit-transition-duration: 2s;
  -webkit-transition-timing-function: linear;
  -webkit-transition-delay: 1s;
  transition-property: width;
  transition-duration: 2s;
  transition-timing-function: linear;
  transition-delay: 1s;
}
.elements:hover {
  width: 300px;
}
```

예제에서 확인했던 방식은 아래의 표와 같이 단축 속성이었습니다.

```
.elements {
  width: 100px;
  height: 100px;
  background: red;
  -webkit-transition: width 2s linear 1s; /* Safari 3.1 to 6.0 */
  transition: width 2s linear 1s;
}
.elements:hover {
  width: 300px;
}
```

3-12) CSS3 Animation

CSS3 애니메이션을 사용하면 JavaScript나 Flash와 같은 외부 프로그램을 사용하지 않아도 애니메이션을 구현할 수 있습니다.

3-12-1) CSS3 Transition과 Animation

애니메이션은 점차적으로 다른 스타일로 변경할 수 있습니다. 이 점은 변환 효과와 같다고 할 수 있죠. 하지만 변환 효과는 단 한 번의 변화인데 비해, 애니메이션은 여러 번을 통해서 속성을 변경시킬 수 있는 것처럼 복잡한 동적이 구현이 가능합니다.

CSS3 애니메이션을 사용하기 위해서는, 먼저 애니메이션에 대한 키 프레임(keyframes) 규칙을 정의해야 합니다. 키 프레임 규칙은 정해진 시간 동안에 스타일을 변경시킬 것입니다.

3-12-2) 키 프레임 규칙

키 프레임 규칙에서 CSS 스타일을 지정하는 경우, 애니메이션은 특정한 시간동안 새로운 스타일로 점차 변경됩니다.

애니메이션을 만들려고 한다면, 요소를 결합해야 합니다. 애니메이션에 대한 선언 부분입니다. Chrome이나 Safari, Opera 브라우저를 위해 prefix를 지정합니다.(@-webkit)

```
/* Chrome, Safari, Opera */
@-webkit-keyframes color_change {
  from {
    background: #8ac007;
  } to {
    background: #333;
  }
}
@keyframes color_change {
  from {
    background: #8ac007;
  } to {
    background: #333;
  }
}
```

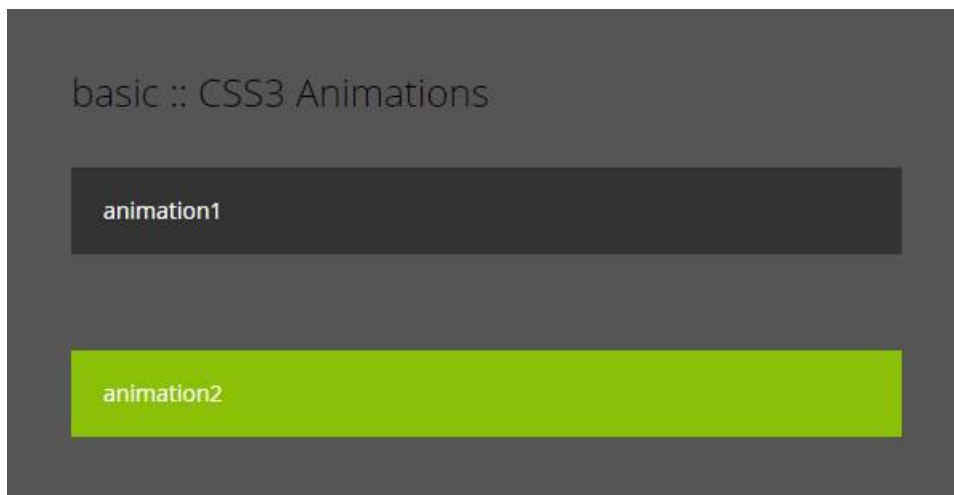
애니메이션을 호출하는 부분입니다.

먼저 `animation-name` 속성으로 애니메이션의 이름을 지정합니다. 이는 키 프레임 규칙에 따른 이름입니다. 또한 `animation-duration` 속성으로 애니메이션이 지속될 시간을 지정합니다. `animation-duration` 속성을 지정하지 않는다면, 기본 값이 0이기 때문에 애니메이션은 아무런 변화가 없는 것처럼 보일 것입니다.

```
-webkit-animation-name: color_change; /* Chrome, Safari, Opera */
-webkit-animation-duration: 1s; /* Chrome, Safari, Opera */
animation-name: color_change;
animation-duration: 1s;
```

시작 파일

sample/basic_css3/start/animation1.html



CSS

```
9      <style>
46      .example .animation1:hover {
47          background: #333;
48          -webkit-animation-name: color_change1; /* Chrome, Safari, Opera */
49          -webkit-animation-duration: 1s; /* Chrome, Safari, Opera */
50          animation-name: color_change1;
51          animation-duration: 1s;
52      }
53      .example .animation2:hover {
54          background: #333;
```

```
55     -webkit-animation-name: color_change2;
56     -webkit-animation-duration: 4s;
57     animation-name: color_change2;
58     animation-duration: 4s;
59 }
60
61 /* Chrome, Safari, Opera */
62 @-webkit-keyframes color_change1 {
63     from {
64         background: #8ac007;
65     } to {
66         background: #333;
67     }
68 }
69 @-webkit-keyframes color_change2 {
70     0% {
71         background: #8ac007;
72     } 25% {
73         background: #07bec0;
74     } 50% {
75         background: #c07d07;
76     } 100% {
77         background: #8ac007;
78     }
79 }
80 @keyframes color_change1 {
81     from {
82         background: #8ac007;
83     } to {
84         background: #333;
85     }
86 }
87 @keyframes color_change2 {
88     0% {
89         background: #8ac007;
90     } 25% {
91         background: #07bec0;
92     } 50% {
93         background: #c07d07;
94     } 100% {
95         background: #8ac007;
96     }
97 }
98 </style>
```

50행 : `animation-name: color_change1; animation-duration: 1s;`

animation1 클래스에 마우스를 올리면, 호출되는 애니메이션 이름은 color_change1입니다. 동작은 1초(1s)간 진행됩니다.

58행 : `.example .animation2:hover {animation-name: color_change2; animation-duration: 4s;}`

animation2 클래스에 마우스를 올리면, 호출되는 애니메이션 이름은 color_change2입니다. 동작은 4초(4s)간 진행됩니다.

62행 : `@-webkit-keyframes color_change1 {`

color_change1 애니메이션에 대한 선언부입니다. 애니메이션을 선언할 때에는 @keyframes 규칙을 통해서 작성합니다. '-webkit-'은 Chrome, Safari, Opera 브라우저에 대한 prefix입니다.

color_change1 애니메이션은 from, to 방식으로 2개의 애니메이션 지점을 정하고 있습니다.

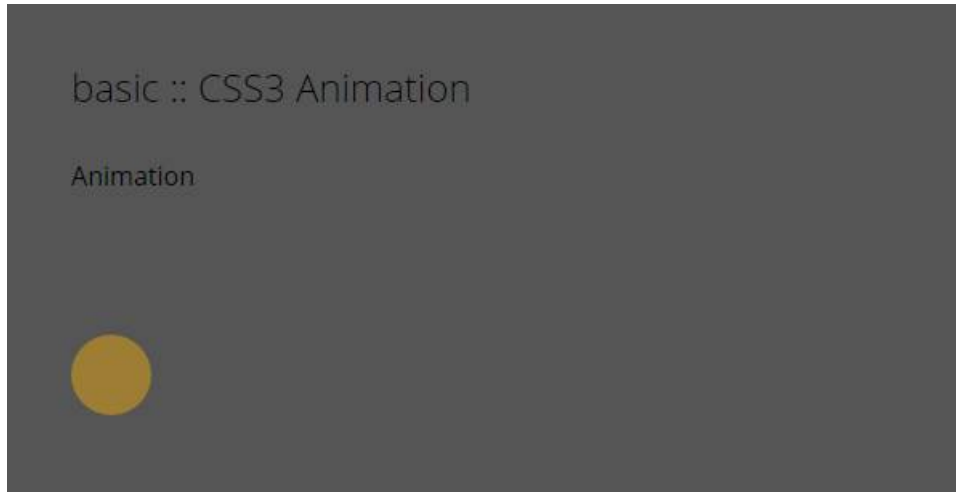
69행 : `@-webkit-keyframes color_change2 {`

color_change2 애니메이션에 대한 선언부입니다.

color_chagnge2 애니메이션은 0, 25%, 50%, 100%와 같은 4개의 애니메이션 지점을 정하고 있습니다. 즉 4개의 색상 변화를 지정하고 있습니다.

시작 파일

start/basic_css3/start/animation2.html



CSS

```
9      <style>
43      .example .ball {
44          position: relative;
45          width: 50px;
46          height: 50px;
47          background-color: #8ac007;
48          border-radius: 50%;
49          -webkit-animation-name: example;
50          -webkit-animation-duration: 4s;
51          animation-name: example;
52          animation-duration: 4s;
53      }
54
55      @-webkit-keyframes example {
56          0% {
57              left: 0;
58              top: 0;
59              background-color: #8ac007;
60          } 25% {
61              left: 200px;
62              top: 0;
63              background-color: #c0a407;
64          } 50% {
```

```

65         left: 200px;
66         top: 200px;
67         background-color: #07c0a8;
68     } 75% {
69         left: 0;
70         top: 200px;
71         background-color: #c00785;
72     } 100% {
73         left: 0;
74         top: 0;
75         background-color: #8ac007;
76     }
77 }
78 @keyframes example {
79     0% {
80         left: 0;
81         top: 0;
82         background-color: #8ac007;
83     } 25% {
84         left: 200px;
85         top: 0;
86         background-color: #c0a407;
87     } 50% {
88         left: 200px;
89         top: 200px;
90         background-color: #07c0a8;
91     } 75% {
92         left: 0;
93         top: 200px;
94         background-color: #c00785;
95     } 100% {
96         left: 0;
97         top: 0;
98         background-color: #8ac007;
99     }
100 }
101 </style>

```

51행 : `animation-name: example; animation-duration: 4s;`

ball 클래스의 애니메이션 이름은 example입니다. 그리고 동작은 4초(4s)간 진행됩니다.

78행 : `@keyframes example {`

example 애니메이션에 대한 선언부입니다.

example 애니메이션은 0, 25%, 50%, 75% 100%와 같은 5개의 애니메이션 지점을 정하고 있습니다.

애니메이션 지점에 대한 상세한 속성입니다. :

- 0% : left: 0; top: 0; background-color: #8ac007;
- 25% : left: 200px; top: 0; background-color: #c0a407;
- 50% : left: 200px; top: 200px; background-color: #07c0a8;
- 75% : left: 0; top: 200px; background-color: #c00785;
- 100% : left: 0; top: 0; background-color: #8ac007;

animation-iteration-count 속성은 애니메이션이 실행해야하는 횟수를 지정하며, animation-direction 속성은 애니메이션이 실행되는 방향(circle)을 지정합니다.

시작 파일

sample/basic_css3/start/animation3.html

CSS

```
9      <style>

43      .example .ball1 {
44          position: relative;
45          width: 50px;
46          height: 50px;
47          background-color: #8ac007;
48          border-radius: 50%;
49          -webkit-animation-name: example;
50          -webkit-animation-duration: 4s;
51          -webkit-animation-iteration-count: 3;
52          animation-name: example;
53          animation-duration: 4s;
54          animation-iteration-count: 3;
55      }
56      .example .ball2 {
57          position: relative;
58          width: 50px;
59          height: 50px;
60          background-color: #8ac007;
61          border-radius: 50%;
62          -webkit-animation-name: example;
63          -webkit-animation-duration: 4s;
64          -webkit-animation-iteration-count: infinite;
65          animation-name: example;
66          animation-duration: 4s;
67          animation-iteration-count: infinite;
68      }
69      .example .ball3 {
70          position: relative;
71          width: 50px;
72          height: 50px;
73          background-color: #8ac007;
74          border-radius: 50%;
75          -webkit-animation-name: example;
76          -webkit-animation-duration: 4s;
77          -webkit-animation-iteration-count: 3;
78          -webkit-animation-direction: reverse;
```

```
79     animation-name: example;
80     animation-duration: 4s;
81     animation-iteration-count: 3;
82     animation-direction: reverse;
83 }
84 .example .ball4 {
85     position: relative;
86     width: 50px;
87     height: 50px;
88     background-color: #8ac007;
89     border-radius: 50%;
90     -webkit-animation-name: example;
91     -webkit-animation-duration: 4s;
92     -webkit-animation-iteration-count: 3;
93     -webkit-animation-direction: alternate;
94     animation-name: example;
95     animation-duration: 4s;
96     animation-iteration-count: 3;
97     animation-direction: alternate;
98 }
99
100 @-webkit-keyframes example {
101     0% {
102         left: 0;
103         top: 0;
104         background-color: #8ac007;
105     } 25% {
106         left: 200px;
107         top: 0;
108         background-color: #c0a407;
109     } 50% {
110         left: 200px;
111         top: 200px;
112         background-color: #07c0a8;
113     } 75% {
114         left: 0;
115         top: 200px;
116         background-color: #c00785;
117     } 100% {
118         left: 0;
119         top: 0;
120         background-color: #8ac007;
121     }
122 }
```

```

123 @keyframes example {
124     0% {
125         left: 0;
126         top: 0;
127         background-color: #8ac007;
128     } 25% {
129         left: 200px;
130         top: 0;
131         background-color: #c0a407;
132     } 50% {
133         left: 200px;
134         top: 200px;
135         background-color: #07c0a8;
136     } 75% {
137         left: 0;
138         top: 200px;
139         background-color: #c00785;
140     } 100% {
141         left: 0;
142         top: 0;
143         background-color: #8ac007;
144     }
145 }
146 </style>

```

54행 : `animation-iteration-count: 3;`

round1 클래스는 적용되는 example 애니메이션을 3번 실행합니다.

67행 : `animation-iteration-count: infinite;`

round2 클래스는 적용되는 example 애니메이션을 연속적으로 무한 반복합니다.

81행 : `animation-iteration-count: 3; animation-direction: reverse;`

round3 클래스는 적용되는 example 애니메이션을 반대방향으로 3번 실행합니다.

85행 : `animation-iteration-count: 3; animation-direction: alternate;`

round4 클래스는 적용되는 example 애니메이션을 한 번은 정 방향으로 실행하고, 다음번은 반대방향으로 실행합니다.

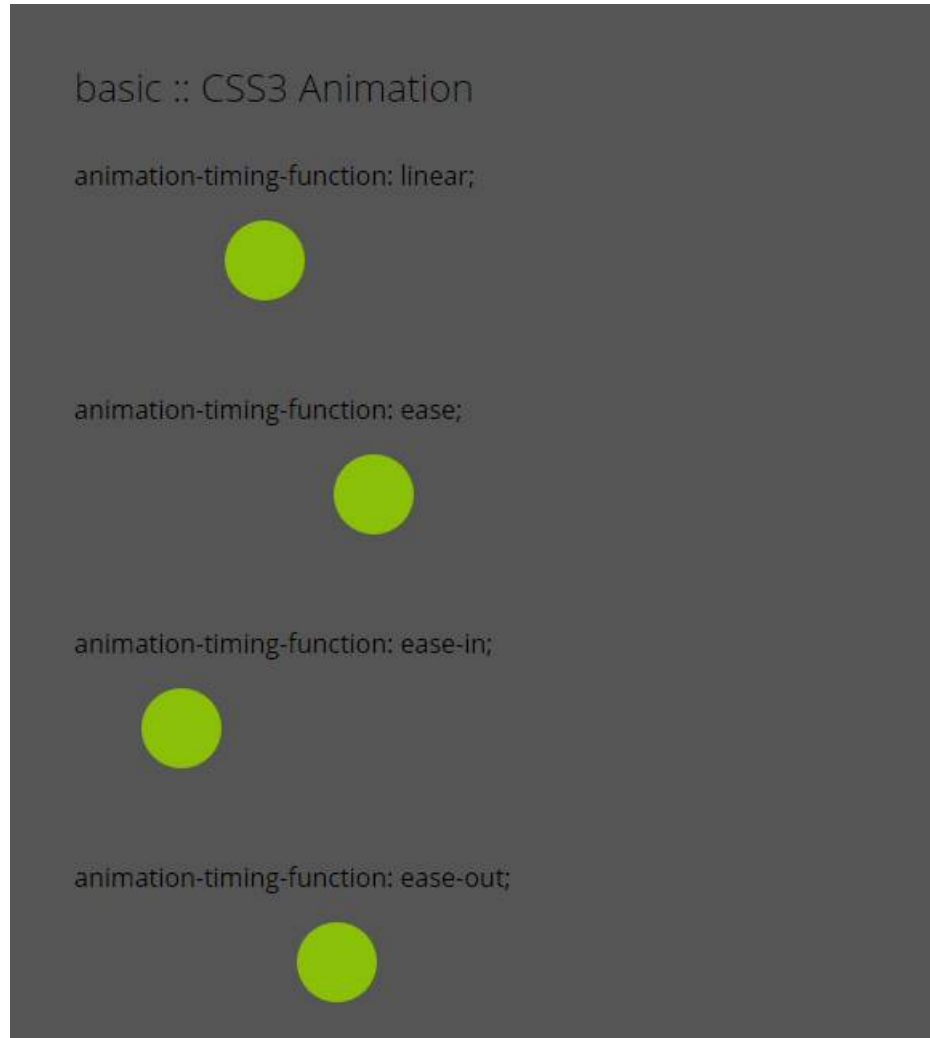
다음은 애니메이션의 속도 곡선의 지정방법에 대해 알아보시다. animation-timing-function 속성은 애니메이션의 속도 곡선을 지정합니다.

animation-timing-function 속성은 아래와 같은 속성 값을 가질 수 있습니다. :

- ease : 느리게 시작해서 빨라지고, 서서히 끝나는 변화 효과를 지정합니다. 기본 값입니다.
- linear : 처음부터 끝까지 같은 속도로 변화 효과를 지정합니다.
- ease-in : 느리게 시작하는 변화 효과를 지정합니다.
- ease-out : 느리게 끝나는 변화 효과를 지정합니다.
- ease-in-out : 느린 시작과 끝으로 변화 효과를 지정합니다.
- cubic-bezier(n,n,n,n) : cubic-bezier() 키워드를 이용해서 자신만의 변화 값을 지정합니다.

시작 파일

sample/basic_css3/start/animations4.html



CSS

```
9      <style>
43      .example div {
44          position: relative;
45          width: 50px;
46          height: 50px;
47          background-color: #8ac007;
48          border-radius: 50%;
49          -webkit-animation-name: move;
50          -webkit-animation-duration: 6s;
```

```
51     -webkit-animation-delay: 2s;
52     -webkit-animation-iteration-count: infinite;
53     animation-name: move;
54     animation-duration: 6s;
55     animation-delay: 2s;
56     animation-iteration-count: infinite;
57 }
58 .ball1 {
59     -webkit-animation-timing-function: linear;
60     animation-timing-function: linear;
61 }
62 .ball2 {
63     -webkit-animation-timing-function: ease;
64     animation-timing-function: ease;
65 }
66 .ball3 {
67     -webkit-animation-timing-function: ease-in;
68     animation-timing-function: ease-in;
69 }
70 .ball4 {
71     -webkit-animation-timing-function: ease-out;
72     animation-timing-function: ease-out;
73 }
74 .ball5 {
75     -webkit-animation-timing-function: ease-in-out;
76     animation-timing-function: ease-in-out;
77 }
78
79 @-webkit-keyframes move {
80     from {
81         left: 0;
82     } to {
83         left: 300px;
84     }
85 }
86 @keyframes move {
87     from {
88         left: 0;
89     } to {
90         left: 300px;
91     }
92 }
93 </style>
```

43행 : .example div {

div 요소는 .round1, .round2, .round3, .round4, .round5 요소를 의미합니다.

호출되는 애니메이션에 대한 속성은 아래와 같습니다. :

- animation-name: move;
- animation-duration: 6s;
- animation-delay: 2s;
- animation-iteration-count: infinite;

move 애니메이션은 left 속성을 0에서 300px 위치로 6초 동안에 이동합니다.

다만 .round1, .round2, .round3, .round4, .round5 요소의 animation-timing-function 속성 값이 다른 점을 확인하는 것입니다.