

반응형 웹



학습 목표

웹의 형태가 다양하게 변화하면서, 이를 표시해주는 디바이스들도 다양해지고 HTML 요소의 배치 또한 이에 맞도록 배치해야 할 필요성이 생기게 되었다. CSS3에서는 보다 복잡한 응용 프로그램과 웹 문서의 레이아웃을 위해 설계된 새로운 레이아웃 모드인 유연한 레이아웃(flex layout) 기능과 그리드 레이아웃(grid layout) 기능을 추가하였다. 이번 장에서는 이렇게 새롭고 유용한 고급 레이아웃 기능들에 대하여 자세히 살펴보고 사용자 인터페이스 관련하여 새롭게 추가된 기능에 대하여 살펴보도록 하겠다.

Section

- 1 유연한 박스 레이아웃(Flexible Box Layout)
- 2 그리드 레이아웃(Grid Layout)
- 3 사용자 인터페이스(User Interface)



표준화 문서	CSS Flexible Box Layout Module Level 1 - http://dev.w3.org/csswg/css-flexbox/
표준화 단계	W3C Editor's Draft (2014-04-17)

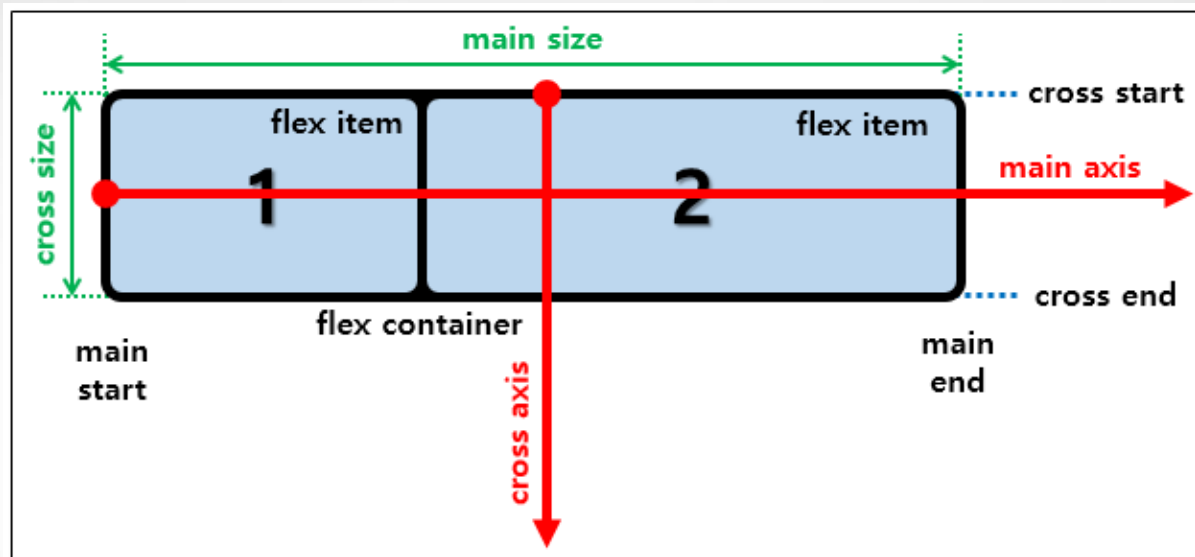
■ 유연한 박스 레이아웃

■ 화면과 브라우저 창이 크기가 변경되더라도 요소의 상대적 위치 및 크기를 일정하게 유지하는 데 유용하기 때문에 복잡한 웹 문서의 레이아웃에 사용된다.

- 서로 다른 화면 및 브라우저 창 크기에 대해 상대적인 위치와 크기를 유지하는 요소(이미지 또는 컨트롤 등)를 포함하는 레이아웃을 만들 수 있다.
- 요소의 레이아웃 축(가로 또는 세로)에서 여분의 공간을 비례적으로 할당하여 주어진 요소의 크기를 늘리는 방법을 지정할 수 있다.
- 요소의 레이아웃 축(가로 또는 세로) 주변에 있는 여분의 공간을 할당하여 요소의 앞, 뒤 또는 사이에 배치하는 방법을 지정할 수 있다.
- 요소의 레이아웃 축과 수직인 여분의 공간(예를 들어, 나란히 배치된 버튼 위 또는 아래의 추가 공간)을 요소 주위에 배치하는 방법을 지정할 수 있다.
- 요소가 페이지에 배치되는 방향(예를 들어, 위에서 아래, 아래에서 위, 왼쪽에서 오른쪽, 오른쪽에서 왼쪽)을 제어할 수 있다.
- 화면에 있는 요소를 DOM(문서 개체 모델)에 지정된 방식과 다른 순서로 다시 정렬할 수 있다.



유연한 박스 모델과 용어



main axis main dimension	플렉스 컨테이너의 기본 축(main axis)은 플렉스 항목이 배치되는 축으로써, 기본 크기(main dimension) 내에서 연장된다.
main-start main-end	플렉스 항목들은 플렉스 컨테이너 내의 기본-시작(main-start)에서 시작하여 기본-끝(main-end) 방향으로 배치된다.
main size main size property	플렉스 항목의 폭 또는 높이는 기본 크기(main size)에서의 값 또는 항목의 기본 크기가 되기 때문에 플렉스 항목의 기본 크기 속성(main size property)은 폭(width) 또는 높이(height) 속성 중에 하나가 된다.
cross axis cross dimension	기본 축에 수직인 축을 교차 축(cross axis)이라 하고 교차 크기(cross dimension)내에서 연장된다.
cross-start cross-end	플렉스 라인(flex lines)은 플렉스 항목들로 채워지고 플렉스 컨테이너의 교차-시작(cross-start) 측면에서 시작하여 교차-끝(cross-end) 측면 방향으로 배치된다.
cross size cross size property	플렉스 항목의 폭이나 높이는 교차 크기(cross size) 내에서 값 또는 항목의 기본 크기가 되기 때문에 교차 크기 속성(cross size property)은 교차 크기 내에서의 폭 또는 높이가 된다.



유연한 컨테이너



속성

display - 박스 모델 지정

CSS3

■ CSS3에서 추가된 flex 또는 inline-flex 값을 지정해서 사용

- ✓ 블록 형식의 컨텍스트를 설정하는 것과 동일하지만, 레이아웃의 형식은 블록 레이아웃과는 다르다. 즉, float은 플렉스 컨테이너에 강제로 침범할 수 없고, 플렉스 컨테이너의 여백 또한 콘텐츠의 여백과 통합(collapse)되지 않는다.

속성 값	flex	inline-flex
JavaScript	<code>object.style.display = "flex"</code>	<code>object.style.display = "inline-flex"</code>
사용 예제	<code>div { display: flex }</code>	<code>div { display: inline-flex }</code>

속성 값	설명
flex	블록 수준의 유연한 플렉스 컨테이너를 생성하도록 지정한다.
inline-flex	인라인 수준의 플렉스 컨테이너를 생성하도록 지정한다.



유연한 박스의 순서 및 방향



속성

flex-direction – 유연한 박스의 배치 방법 지정하기

CSS3

플렉스 컨테이너의 기본 축(main axis) 방향을 설정

속성 값	row row-reverse column column-reverse					
속성 특징	초기값	row	적용	플렉스 컨테이너들	상속	X
JavaScript	<code>object.style.flexDirection = "column-reverse"</code>					
사용 예제	<code>div { flex-direction: column-reverse }</code>					

속성 값	설명
<i>row</i>	플렉스 컨테이너의 기본 축을 현재 쓰기 모드의 인라인 축과 동일한 방향으로 지정한다. 이는 main-start와 main-end 방향이 각각 inline-start와 inline-end 방향에 해당된다. 따라서 자식 요소는 원본 문서에서 선언한 것과 같은 순서인 왼쪽에서 오른쪽으로 표시된다.
row-reverse	방향이 반대인 것을 제외하고는 row 속성 값과 동일하다.
column	플렉스 컨테이너의 기본 축을 현재 쓰기 모드의 블록 축과 동일한 방향으로 지정한다. 이는 main-start와 main-end 방향이 각각 block-start와 block-end 방향에 해당된다. 따라서 자식 요소는 원본 문서에서 선언한 것과 같은 순서인 위에서 아래쪽으로 표시된다.
column-reverse	방향이 반대인 것을 제외하고는 column 속성 값과 동일하다



유연한 박스의 순서 및 방향



속성

flex-direction – 유연한 박스의 배치 방법 지정하기

CSS3

- 플렉스 컨테이너의 기본 축(main axis) 방향을 설정



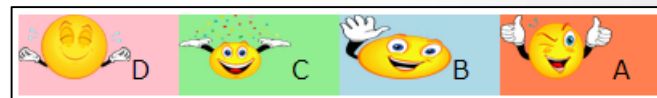
예제 살펴보기

예제

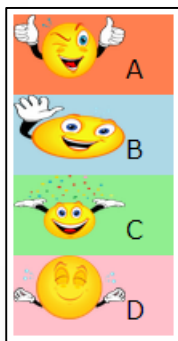
CSS3_08-01_FlexDirection.html



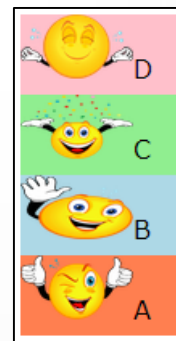
flex-direction: row



flex-direction: row-reverse



flex-direction: column



flex-direction: column-reverse



유연한 박스의 순서 및 방향



속성

flex-wrap – 유연한 박스의 넘김 지정하기

CSS3

- 플렉스 컨테이너를 한 줄 또는 여러 줄에 배치할 것인지를 제어하고 새로운 줄이 교차 축에 어떠한 방향으로 진행되는지를 결정

- ✓ 플렉스 항목이 플렉스 컨테이너에서 사용 가능한 공간을 기준으로 여러 줄 또는 열에 줄 바꿈 될지를 지정

속성 값	nowrap wrap wrap-reverse					
속성 특징	초기값	nowrap	적용	플렉스 컨테이너들	상속	X
JavaScript	<code>object.style.flexWrap = "wrap"</code>					
사용 예제	<code>div { flex-wrap: wrap }</code>					

속성 값	설명
<i>nowrap</i>	플렉스 컨테이너가 한 줄에 배치하도록 지정한다. 교차-시작(cross-start) 방향은 inline-start 또는 block-start 중에 하나와 동일.
wrap	플렉스 컨테이너가 여러 줄에 배치하도록 지정한다. 교차-시작(cross-start) 방향은 inline-start 또는 block-start 중에 하나와 동일. 플렉스 항목들이 줄 바꿈되고 연속된 행이나 열에 표시된다. 이때, 행이나 열이 추가될 수 있도록, 현재 쓰기 모드 속성에서 정의한 축에 수직으로 플렉스 컨테이너의 높이 또는 폭이 확장될 수 있다.
wrap-reverse	cross-start와 cross-end 방향이 반대인 것을 제외하고는 wrap과 동일.



유연한 박스의 순서 및 방향



속성

flex-wrap – 유연한 박스의 넘김 지정하기

CSS3

- 플렉스 컨테이너를 한 줄 또는 여러 줄에 배치할 것인지를 제어하고 새로운 줄이 교차 축에 어떠한 방향으로 진행되는지를 결정

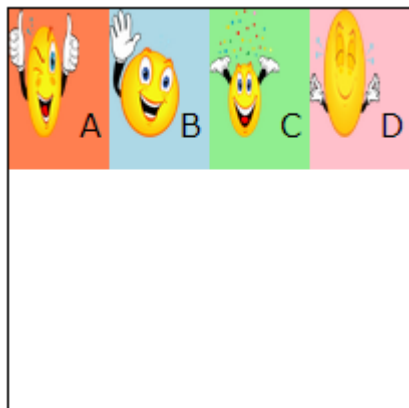
- ✓ 플렉스 항목이 플렉스 컨테이너에서 사용 가능한 공간을 기준으로 여러 줄 또는 열에 줄 바꿈 될지를 지정



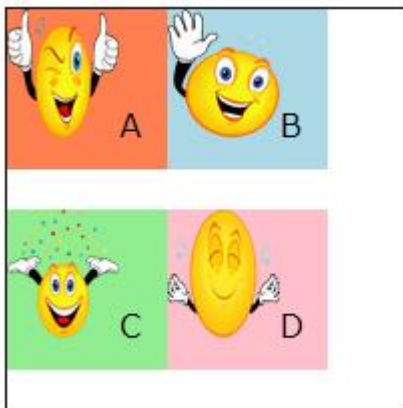
예제 살펴보기

예제

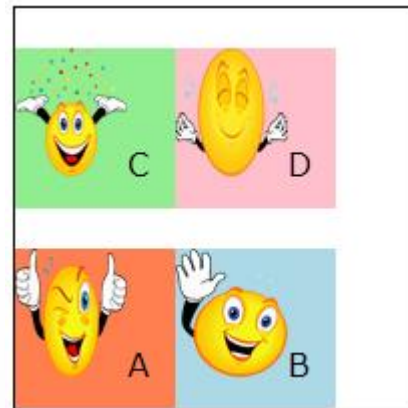
CSS3_08-01_FlexWrap.html



flex-wrap: nowrap



flex-wrap: wrap



flex-wrap: wrap-reverse



유연한 박스의 순서 및 방향



속성

flex-flow – 유연한 박스의 배치 및 넘김 지정하기


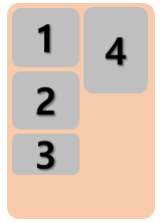
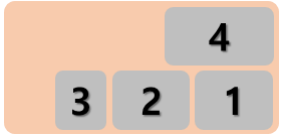
CSS3

- flex-direction 속성과 flex-flow 속성을 한번에 지정할 수 있는 대표 속성

속성 값	<flex-direction> <flex-wrap>					
속성 특징	초기값	개별 속성 참조	적용	플렉스 컨테이너들	상속	X
JavaScript	object.style.flexFlow = "column wrap"					
사용 예제	div { flex-flow: column wrap }					



예제 살펴보기

div { flex-flow: row } /* 기본 축(main-axis)은 인라인, 한 줄에 배치하도록 지정 */	
div { flex-flow: column wrap } /* 기본 축(main-axis)은 블록 방향(위에서 아래 방향) 여러 줄에 배치하도록 지정(왼쪽에서 오른쪽으로) */	
div { flex-flow: row-reverse wrap-reverse } /* 기본 축(main-axis)은 인라인 방향의 반대(오른쪽에서 왼쪽), 여러 줄에 배치하도록 지정(아래에서 위쪽으로) */	



유연한 박스의 순서 및 방향



속성

order – 유연한 박스의 배치 순서 지정하기









CSS3

- 플렉스 항목들이 배치하게 되는 순서를 변경

속성 값	<integer>					
속성 특징	초기값	0	적용	플렉스 항목들과 절대값으로 위치한 플렉스 컨테이너의 자손들	상속	X
JavaScript	<code>object.style.order = "2"</code>					
사용 예제	<code>div { order: 2 }</code>					



예제 살펴보기

예제	CSS3_08-01_FlexOrder.html		
<div><div> order: 2</div><div> order: 기준</div><div> order: 4</div><div> order: 6</div></div>		<div><div> order: 2</div><div> order: 4</div><div> order: 기준</div><div> order: 6</div></div>	
order: 3		order: 5	



유연성



속성

flex-grow – 유연한 박스 폭의 확대 비율 지정

CSS3

플렉스 항목들의 전체 폭이 플렉스 컨테이너의 폭보다 작을 경우에

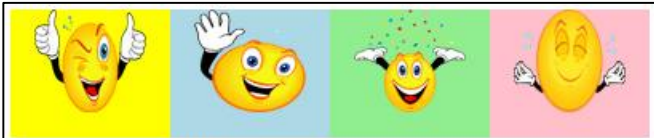
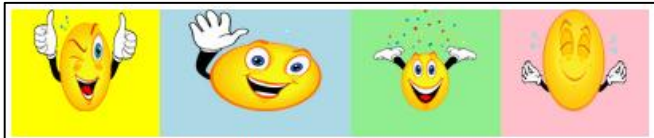
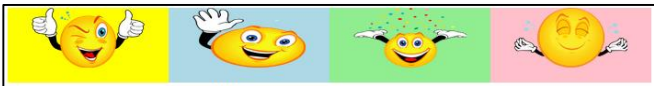
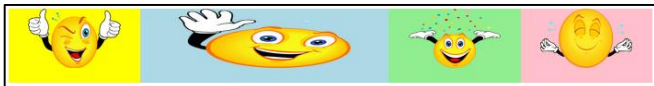
- ✓ 플렉스 항목들 폭의 확대 비율을 지정하여 플렉스 컨테이너 폭에 맞게 자동으로 크기를 조절

속성 값에 따라서 플렉스 컨테이너의 여백 공간 안에서 자동으로 크기 조절

속성 값	<number>					
속성 특징	초기값	0	적용	플렉스 항목들	상속	X
JavaScript	object.style.flexGrow = "3"					
사용 예제	div { flex-grow: 3 }					



예제 살펴보기

예제	CSS3_08-01_FlexGrow.html	
<div> grow 항목</div>		
flex-grow: 1 (width: 500px)		
<div> grow 항목</div>		
flex-grow: 3 (width: 500px)		
<div> grow 항목</div>		
flex-grow: 1 (width: 700px)		
<div> grow 항목</div>		
flex-grow: 3 (width: 700px)		



유연성



속성

flex-shrink – 유연한 박스 폭의 축소 비율 지정하기

CSS3

플렉스 항목들의 전체 폭이 플렉스 컨테이너의 폭보다 큰 경우에

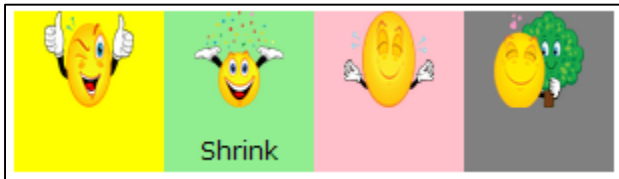

- ✓ 플렉스 항목들 폭의 축소 비율을 지정하여 플렉스 컨테이너 폭에 맞게 자동으로 크기를 조절

속성 값에 따라서 플렉스 컨테이너의 여유 공간 내에서 자동으로 크기 조절

속성 값	<number>				
속성 특징	초기값	1	적용	플렉스 항목들	상속 X
JavaScript	<code>object.style.flexShrink = "3"</code>				
사용 예제	<code>div { flex-shrink: 3 }</code>				



예제 살펴보기

예제	CSS3_08-01_FlexShrink.html	
		
flex-shrink: 1		flex-shrink: 2



유연성



속성

flex-basis – 유연한 박스의 기준 폭 지정하기

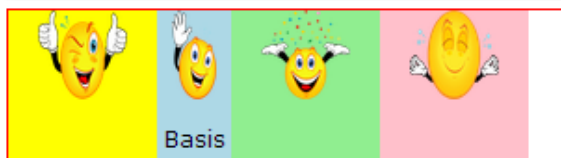
CSS3

속성 값	auto <'width'>					
속성 특징	초기값	auto	적용	플렉스 항목들	상속	X
JavaScript	<code>object.style.flexBasis = "80px"</code>					
사용 예제	<code>div { flex-basis: 80px }</code>					

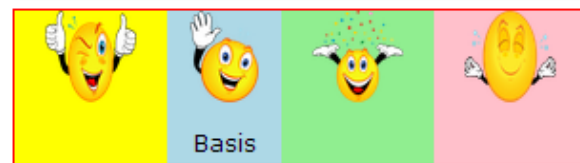
속성 값	설명
<i>auto</i>	기준 폭을 플렉스 항목의 기본 크기(main size) 속성의 값이 되도록 지정한다. 이는 콘텐츠에 기초한 플렉스 항목의 크기를 의미한다.
<i>width</i>	width 속성에서 사용할 수 있는 값과 동일하게 플렉스 항목의 기준 폭을 지정한다. 만일, width 속성 값도 함께 지정되어 있다면 flex로 지정된 요소는 flex-basis 속성 값이 적용된다. 그러나 flex-basis 속성 값이 지정되어 있지 않으면 0%로 지정된다.

예제

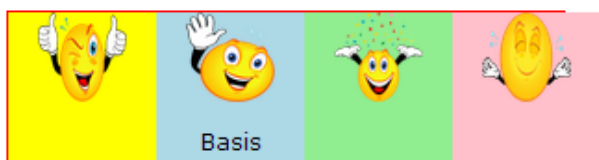
CSS3_08-01_FlexBasis.html



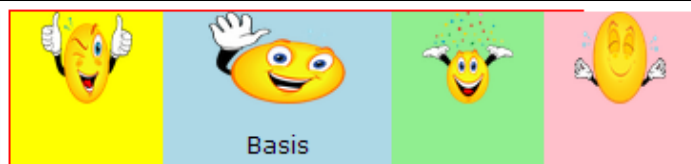
flex-basis: 40px



flex-basis: 60px



flex-basis: 80px



flex-basis: 120px



유연성



속성

flex – 유연한 박스 지정하기

CSS3

- flex-grow, flex-shrink, 그리고 flex-basis 속성을 한번에 지정할 수 있는 대표 소서

속성 값	none [<'flex-grow'> <'flex-shrink'>? <'flex-basis'>]					
속성 특징	초기값	개별 속성 참조	적용	플렉스 항목들	상속	X
JavaScript	<code>object.style.flex = "3 2 80px"</code>					
사용 예제	<code>div { flex: 3 2 80px } /* flex: flex-grow flex-shrink flex-basis */</code>					



정렬 지정하기




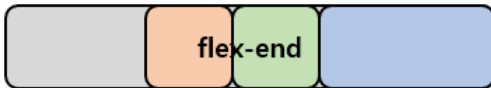

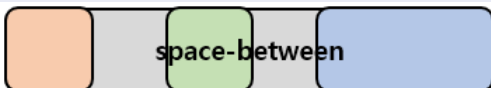
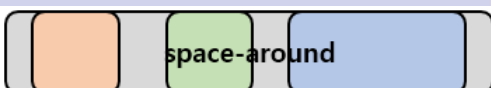
속성

justify-content – 유연한 박스의 정렬 방식 지정하기

CSS3

- 플렉스 컨테이너 안에서 현재 줄의 기본 축을 따라서 플렉스 항목들의 정렬

속성 값	flex-start flex-end center space-between space-around				
속성 특징	초기값	flex-start	적용	플렉스 항목들	상속
JavaScript	<code>object.style.justifyContent = "center"</code>				
사용 예제	<code>div { justify-content: center }</code>				

속성 값	설명	
<i>flex-start</i>	첫 번째 플렉스 항목이 줄의 시작으로 배치되고 나머지 항목들은 가로로 배치된다.	
<i>flex-end</i>	마지막 플렉스 항목이 줄의 마지막으로 배치되고 앞의 항목들이 이어서 가로로 배치된다.	
<i>center</i>	플렉스 항목들이 가운데 가로로 배치된다.	
<i>space-between</i>	플렉스 항목들이 가로로 균일하게 배치되기 때문에 항목간에는 공백이 자동으로 들어간다.	
<i>space-around</i>	플렉스 항목들이 가로로 균일하게 배치하되, 양끝의 여백은 균일한 공백의 절반이 자동으로 들어간다.	



정렬 지정하기



속성

justify-content – 유연한 박스의 정렬 방식 지정하기

CSS3

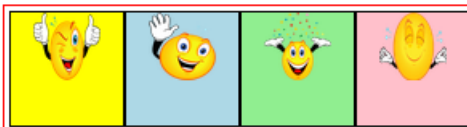
- 플렉스 컨테이너 안에서 현재 줄의 기본 축을 따라서 플렉스 항목들의 정렬
방향을 지정



예제 살펴보기

예제

CSS3_08-01_JustifyContent.html



justify-content: flex-start



justify-content: flex-end



justify-content: center



justify-content: space-between



justify-content: space-around



정렬 지정하기



속성

align-content – 유연한 박스의 콘텐츠 정렬 방식 지정하기

CSS3

- 플렉스 컨테이너 안에서 교차 축의 여분의 공간이 있을 때, 컨테이너 줄의 정렬

속성 값	flex-start flex-end center space-between space-around stretch					
속성 특징	초기값	stretch	적용	플렉스 항목들	상속	X
JavaScript	<code>object.style.alignContent = "center"</code>					
사용 예제	<code>div { align-content: center }</code>					

속성 값	설명
<i>stretch</i>	줄이 나머지 공간을 차지하도록 플렉스 항목들의 높이를 늘린다. 여유 공간이 부족할 경우에는 flex-start와 동일하지만, 그렇지 않으면, 여유 공간은 모든 줄 사이에 균등하게 분할된다.
<i>flex-start</i>	플렉스 컨테이너의 첫 번째 줄이 cross-start 측에 배치되고 그 이후의 플렉스 컨테이너 줄들은 연속적으로 배치된다.
<i>flex-end</i>	플렉스 컨테이너의 마지막 번째 줄이 cross-end 측에 배치되고 그 이전의 플렉스 컨테이너 줄들은 연속적으로 배치된다.
<i>center</i>	플렉스 컨테이너 줄들이 세로 가운데로 배치된다.
<i>space-between</i>	플렉스 컨테이너 줄들이 세로로 균일하게 배치되고 줄간의 공백이 자동으로 들어간다.
<i>space-around</i>	플렉스 컨테이너 줄들이 세로로 균일하게 배치하되, 상단과 하단의 여백은 줄간 공백의 절반이 자동으로 들어간다. 만일, 남은 여유공간이 모자라면 이 값은 center와 동일하다.



정렬 지정하기



속성

align-content – 유연한 박스의 콘텐츠 정렬 방식 지정하기

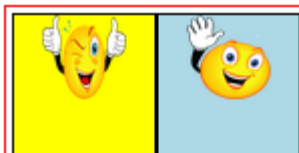
CSS3

예제

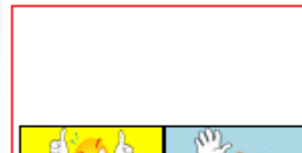
CSS3_08-01_AlignContent.html



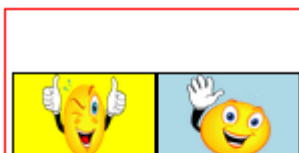
align-content: stretch



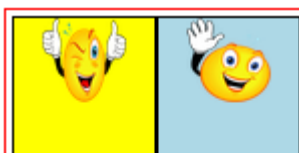
align-content: flex-start



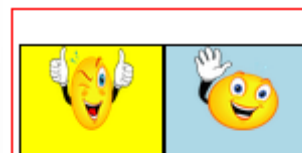
align-content: flex-end



align-content: center



align-content: space-between



align-content: space-around



유연한 박스의 교차 축 지정하기

- 플렉스 컨테이너 줄의 높이에 따라 플렉스 항목들을 배치하는 교차 축 지정

✓ 속성	align-items					CSS3
속성 값	flex-start flex-end center baseline stretch					
속성 특징	초기값	stretch	적용	플렉스 항목들	상속	X
JavaScript	<code>object.style.alignItems = "center"</code>					
사용 예제	<code>div { align-items: center }</code>					

✓ 속성	align-self					CSS3
속성 값	auto flex-start flex-end center baseline stretch					
속성 특징	초기값	auto	적용	플렉스 항목들	상속	X
JavaScript	<code>object.style.alignSelf = "center"</code>					
사용 예제	<code>div { align-self: center }</code>					

속성 값	설명
auto	align-self 속성의 기본값으로, 부모 요소의 align-item 값으로 계산되거나 부모가 없는 요소를 교차 축의 높이만큼 늘린다.
stretch	교차 크기에 공간을 차지하도록 플렉스 항목들의 높이를 늘리도록 지정한다.
flex-start	플렉스 항목이 cross-start 측에 배치되도록 지정한다.
flex-end	플렉스 항목이 cross-end 측에 배치되도록 지정한다.
center	플렉스 항목이 교차 축(cross axis)의 세로 가운데로 배치되도록 지정한다.
baseline	플렉스 항목의 인라인 축이 교차 축과 동일하면 flex-start와 동일하지만, 그렇지 않으면 baseline 정렬이 되도록 한다.



유연한 박스의 교차 축 지정하기

- 플렉스 컨테이너 줄의 높이에 따라 플렉스 항목들을 배치하는 교차 축 지정



속성

align-items

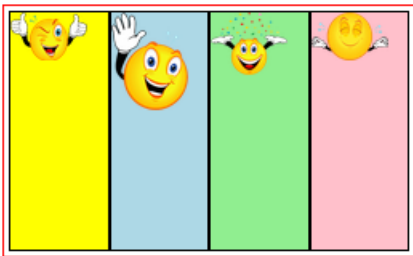
CSS3



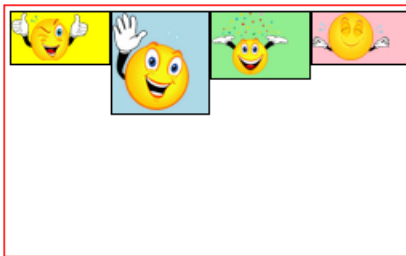
예제 살펴보기

예제

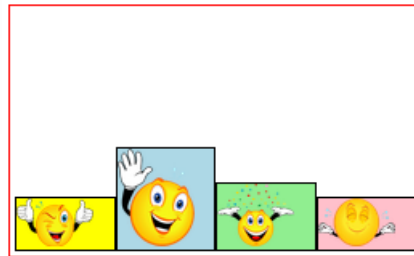
CSS3_08-01_AlignItems.html



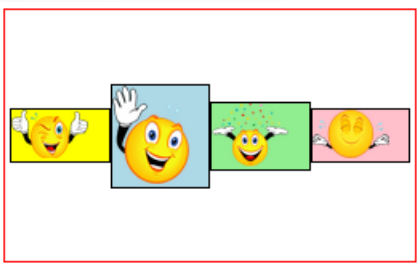
align-items: stretch



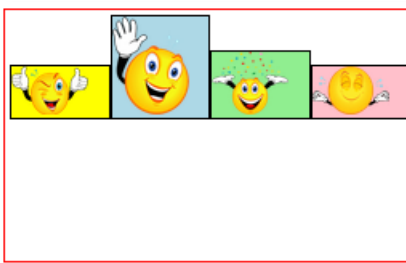
align-items: flex-start



align-items: flex-end



align-items: center



align-items: baseline



유연한 박스의 교차 축 지정하기

- 플렉스 컨테이너 줄의 높이에 따라 플렉스 항목들을 배치하는 교차 축 지정



속성

align-self

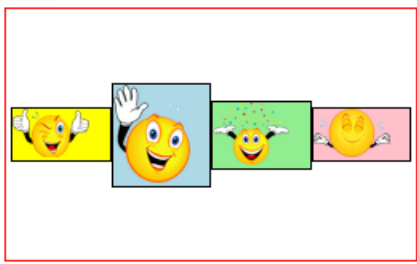
CSS3



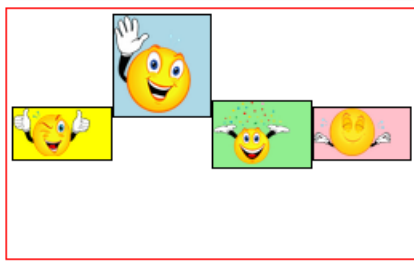
예제 살펴보기

예제

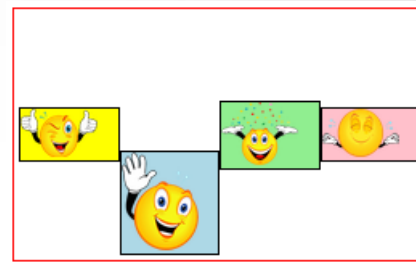
CSS3_08-01_AlignSelf.html



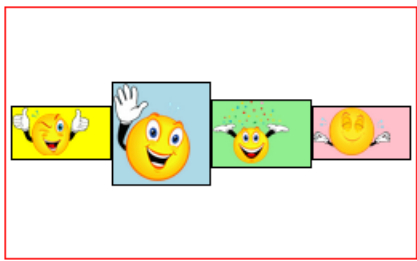
align-self: auto



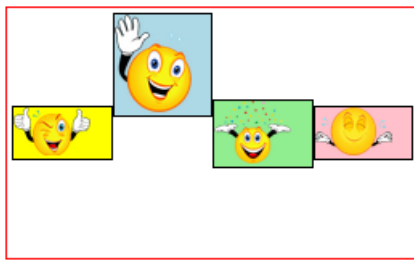
align-self: flex-start



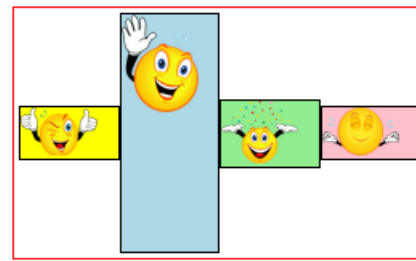
align-self: flex-end



align-self: center



align-self: baseline



align-self: stretch



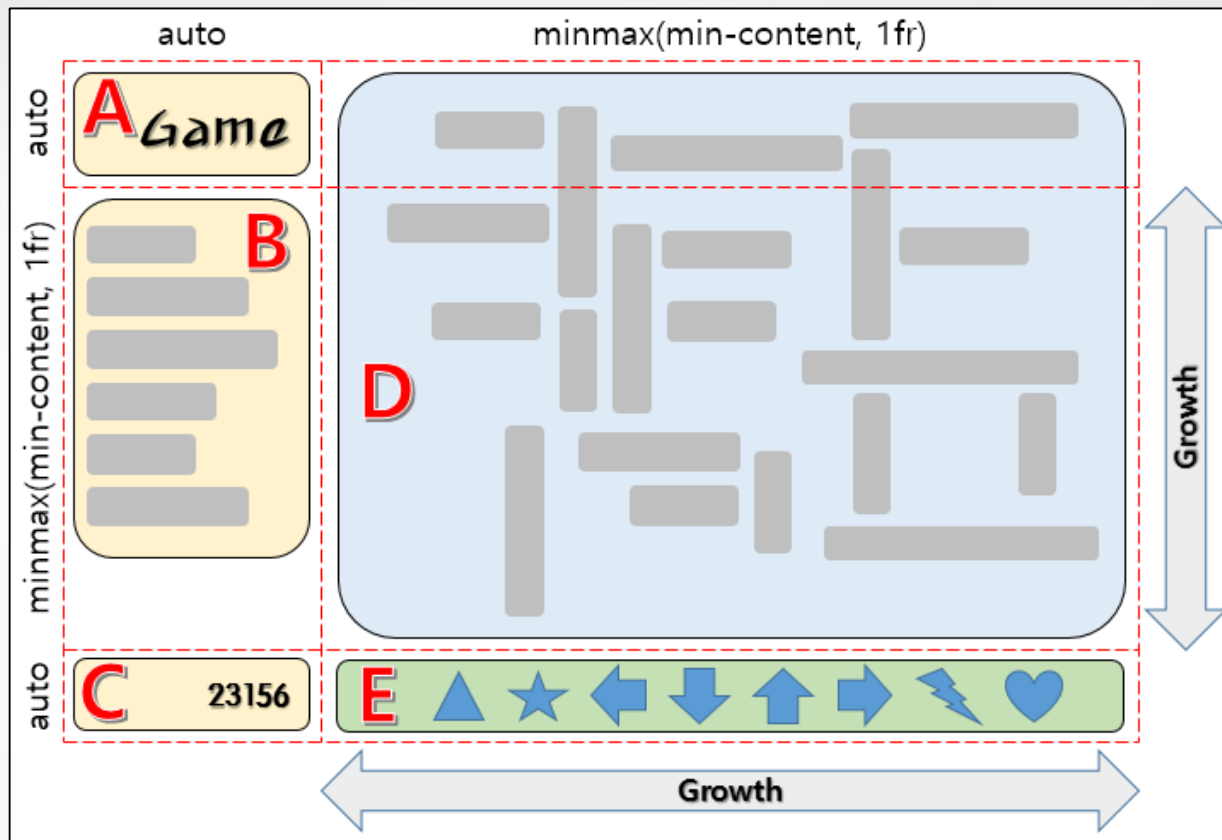
표준화 문서	CSS Grid Layout Module Level 1 - http://dev.w3.org/csswg/css-grid/
표준화 단계	W3C Editor's draft (2014-05-20)

■ 그리드 레이아웃

- 브라우저 창 내에서 사용 가능한 공간이나 고정된 공간, 또는 이들의 조합을 기반으로 웹 문서의 주 영역 공간을 분할하는 새로운 레이아웃
 - 그리드 레이아웃에는 콘텐츠 구조가 없기 때문에 HTML의 표 또는 CSS를 통해서는 불가능하거나 수행하기 아주 어려운 레이아웃을 만들 수 있다.
 - ✓ 그리드 컨테이너의 하위 요소들을 겹치도록 배치할 수 있다.
 - ✓ 그리고 미디어 쿼리와 함께 사용하면 장치에 대한 폼 팩터(factors), 방향, 사용 가능한 공간 등의 변경에 맞추어 레이아웃을 더욱 원활하게 조정할 수 있다.



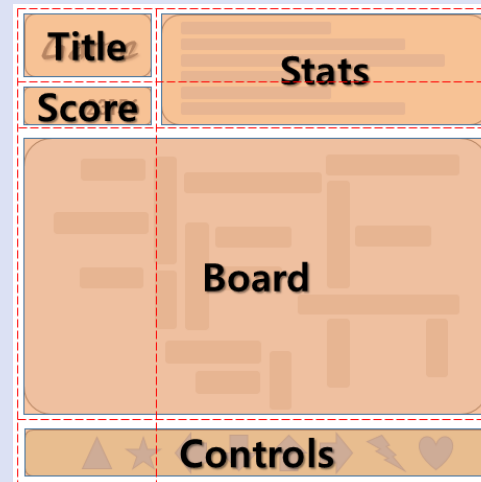
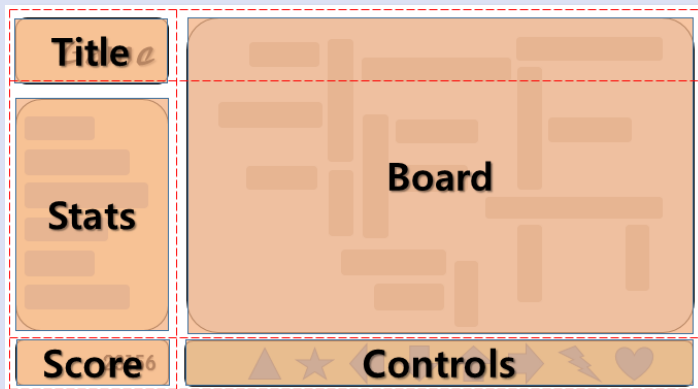
■ 그리드 레이아웃 배경



사용 가능한 공간의 변경에 따른 D와 E 부분의 크기 변화를 보여주는 그리드 레이아웃



그리드 레이아웃 배경



가로 방향(Landscape)에 적절한 배치

세로 방향(Portrait)에 적절한 배치

CSS

```
#grid { display: grid;
  grid-template-areas: "title stats"
                      "score stats"
                      "board board"
                      "ctrls ctrls";

  grid-template-columns: auto minmax(min-content, 1fr);
  grid-template-rows: auto auto minmax(min-content, 1fr) auto;
}

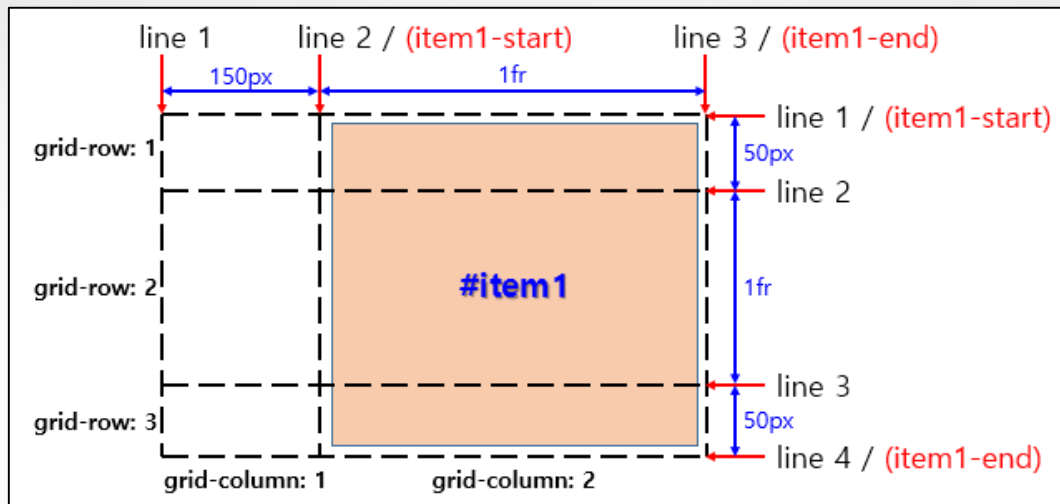
@media (orientation: landscape) {
  #grid { display: grid;
    grid-template-areas: "title board"
                      "stats board"
                      "score ctrls";

    grid-template-columns: auto minmax(min-content, 1fr);
    grid-template-rows: auto minmax(min-content, 1fr) auto;
  }
}
```

```
#title { grid-area: title; }
#score { grid-area: score; }
#stats { grid-area: stats; }
#board { grid-area: board; }
#ctrls { grid-area: ctrls; }
```



■ 그리드 레이아웃 개념 및 용어



🚩 그리드 라인(Line)

➤ 그리드의 가로 및 세로의 분할 선을 나타내는 것/ 열(grid-column), 행(grid-row)

- ✓ 1번째 열(grid-column: 1)은 왼쪽에 line1, 오른쪽에 line2가 위치
- ✓ 그리드 컨테이너의 공간을 그리드 영역으로 분할하는 역할

✓ 그리드 항목(그리드 컨테이너의 콘텐츠)이 위치하고 정렬되어 배치

사용 예제

```
#grid { display: grid;
  grid-template-columns: 150px 1fr; /* 2열, 1열의 크기는 150px, 2열의 크기는 유연한(flexible) 크기 */
  grid-template-rows: 50px 1fr 50px /* 3행 */
}
#item1 { grid-column: 2; grid-row: 1 / span 2; } /* 2번째 열, 1번째 행에서 3번째 행까지 확장 */
```

※ 그리드 라인의 숫자 인덱스 사용



■ 그리드 레이아웃 개념 및 용어

🚩 그리드 라인의 이름 사용: **<custom-ident>***

➤ 그리드 라인 인덱스 대신에 지정한 이름 사용(item-start 및 item1-end 등)

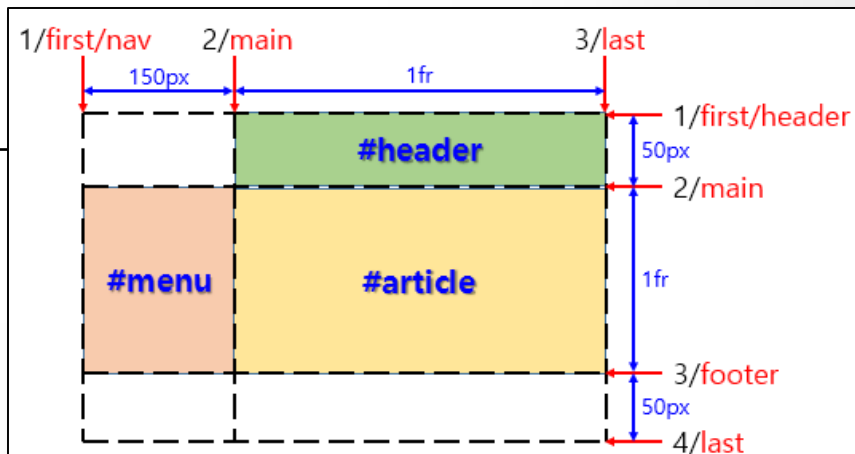
사용 예제

※ 그리드 라인의 이름 사용

```
#grid { display: grid;  
  /* 두 번째 열(1fr)의 왼쪽과 오른쪽의 그리드 라인의 이름을 각각 item1-start와 item1-end로 명시하여 지정 */  
  grid-template-columns: 150px (item1-start) 1fr (item1-end);  
  grid-template-rows: (item1-start) 50px 1fr 50px (item1-end);  
}  
#item1 { grid-column: item1-start / item1-end; grid-row: item1-start / item1-end }
```

사용 예제

```
#grid { display: grid;  
  grid-template-columns: (first nav) 150px (main) 1fr (last);  
  grid-template-rows: (first header) 50px (main) 1fr (footer) 50px (last);  
}  
#menu { grid-column: nav; grid-row: main; }  
#header { grid-column: main; grid-row: header; }  
#article { grid-column: main; grid-row: main; }
```





■ 그리드 레이아웃 개념 및 용어

🚩 그리드 트랙(Track)

- 두 개의 인접하는 그리드 라인 사이의 공간을 의미하며 지정한 요소들이 위치하게 될 가상의 그리드
 - ✓ 그리드 열과 행에 대한 또 다른 이름의 일반적인 용어
 - ✓ 앞의 예제에서, grid-template-columns 및 grid-template-rows가 트랙에

🚩 그리드 셀(Cell)

- 두 개의 인접한 행이나 열의 그리드 라인 사이의 공간
 - ✓ 그리드 항목을 위치할 때 참조할 수 있는 최소 단위(엑셀에서의 셀과 동일한 의미)

🚩 그리드 영역(Areas)

- 하나 이상의 그리드 항목을 배치하는데 사용되는 논리적인 공간

사용 예제

```
#grid { display: grid;  
  grid-template-areas: ". a"  
                      "b a"  
                      ". a";  
  grid-template-columns: 150px 1fr;  
  grid-template-rows: 50px 1fr 50px;  
}
```

```
#item1 { grid-area: a }  
#item2 { grid-area: b }  
#item3 { grid-area: b }  
  
#item2 { align-self: start }  
#item3 { align-self: end; justify-self: end; }
```

- item2와 item3는 동일한 영역에 할당되지만, 서로 다른 위치에 정렬됨을 볼 수 있다.



■ 그리드 컨테이너 및 항목



속성

display - 그리드 컨테이너 지정하기

CSS3

- 새로운 그리드 형식의 컨테이너를 만든다.

속성 값	grid	inline-grid
JavaScript	<code>object.style.display = "grid"</code>	<code>object.style.display = "inline-grid"</code>
사용 예제	<code>div { display: grid }</code>	<code>div { display: inline-grid }</code>

속성 값	설명
grid	블록 레벨의 그리드 컨테이너를 생성하는 요소를 지정한다.
inline-grid	인라인 레벨의 그리드 컨테이너를 생성하는 요소를 지정한다.



그리드 항목(Items)

- 그리드 컨테이너의 각 자식들은 그리드 레벨의 박스로서, 그리드 항목이 된다.
 - ✓ 그리드 컨테이너의 콘텐츠는 하나 이상의 그리드 항목으로 구성되어 있다.
 - ✓ order 속성을 그리드 항목에 적용시킬 수 있기 때문에 그리드 항목들이 배치되는 순서를 정할 수 있다.
 - ✓ 그리드 컨테이너 자식의 위치는 justify-self 및 align-self 값에 의해 영향을 받지만, 대부분의 다른 레이아웃 모델에서와 같이 절대적으로 위치한 자식은 포함 블록이나 콘텐츠 레이아웃의 크기에는 영향을 주지 않는다.



■ 그리드 컨테이너 및 항목



속성

display - 그리드 컨테이너 지정하기

CSS3

■ 그리드 컨테이너의 특징

- float 속성은 그리드 컨테이너에 침범하지 않는다.
- 그리드 컨테이너의 여백은 콘텐츠간의 여백을 통합(collapse)하지 않는다.
- overflow 속성은 그리드 컨테이너에 적용된다.
- 그리드 컨테이너는 블록 컨테이너가 아니기 때문에, 블록 레이아웃으로 가정하여 디자인된 일부 속성들은 그리드 레이아웃에 적용되지 않는다.
- 다단(multicol_ 모듈에 있는 'column-*' 와 관련된 모든 속성들은 그리드 컨테이너에서는 효과가 없다.
- float 및 clear 는 그리드 항목에 영향을 주지 않지만, float 속성은 여전히 그리드 항목이 결정되기 전에 발생한 그리드 컨테이너의 자식 요소에 대한 display 속성의 계산된 값에는 영향을 준다.
- vertical-align 속성은 그리드 항목에 영향을 주지 않는다.
- ::first-line과 ::first-letter 가상 요소는 그리드 컨테이너에는 적용되지 않는다.



■ 그리드 레이아웃 만들기

➤ 각 그리드 열의 폭 및 행의 높이를 지정한다.



속성

grid-template-columns, grid-template-rows

CSS3

속성 값	none <track-list> subgrid <line-name-list>?
	<track-list> = [<line-names>? [<track-size> <repeat()>]]+ <line-names>? <track-size> = minmax(<track-breadth> , <track-breadth>) auto <track-breadth> <track-breadth> = <length> <percentage> <flex> min-content max-content <line-names> = (<custom-ident>*) <line-name-list> = [<line-names> repeat(<positive-integer>, <line-names>+)]+

속성 값	설명
length	표준 길이 단위의 음수가 아닌 길이를 나타낸다.
percentage	음수가 아닌 백분율을 나타낸다.
flex	트랙의 flex 계수(factor)로 지정되는 단위 'fr'를 사용하여 양수 값의 크기로 지정한다. <flex> 크기로 지정된 트랙은 flex 계수에 비례하여 나머지 공간을 유연하게(flexible) 공유한다.
max-content	그리드 트랙을 차지하고 있는 그리드 항목의 최대 크기(폭 또는 높이)를 나타낸다.
min-content	그리드 트랙을 차지하고 있는 그리드 항목의 최소 크기(폭 또는 높이)를 나타낸다.
minmax (min, max)	사용 가능한 공간이 허용하는 대로, 폭 또는 높이의 크기 범위가 min과 max 사이에서 지정
auto	내부 항목을 기준으로 열의 폭 또는 행의 높이가 자동으로 지정된다.
none	명시적인 그리드가 없음을 나타낸다.
subgrid	그리드가 그 축에서 상위 그리드에 정렬됨을 나타내는 것으로, 명시적으로 행 및 열의 크기를 지정하는 대신에 부모 그리드의 정의에서 값을 취하도록 지정한다.



■ 그리드 레이아웃 만들기

- 각 그리드 열의 폭 및 행의 높이를 지정한다.



속성

grid-template-columns, grid-template-rows

CSS3

사용 예제

grid-template-columns: 100px 1fr max-content minmax(min-content, 1fr);

grid-template-rows: 1fr minmax(min-content, 1fr);

grid-template-rows: 10px repeat(2, 1fr auto minmax(30%, 1fr));

grid-template-rows: calc(4em - 5px)



유연한 크기를 나타내는 'fr' 단위

- 유연한 크기 또는 `<flex>`는 'fr' 단위를 갖는 크기으로써, 그리드 컨테이너 내의 자유 공간의 비율을 분수(fraction)로 나타낸다.
- ✓ 사용 가능한 공간을 분수 값에 따라 열 또는 행 사이에 나누어야 함을 나타내는 분수 단위

열 또는 행의 **< flex >** 자유 공간
 —————
 모든 **flex** 계수들의 합



■ 그리드 레이아웃 만들기

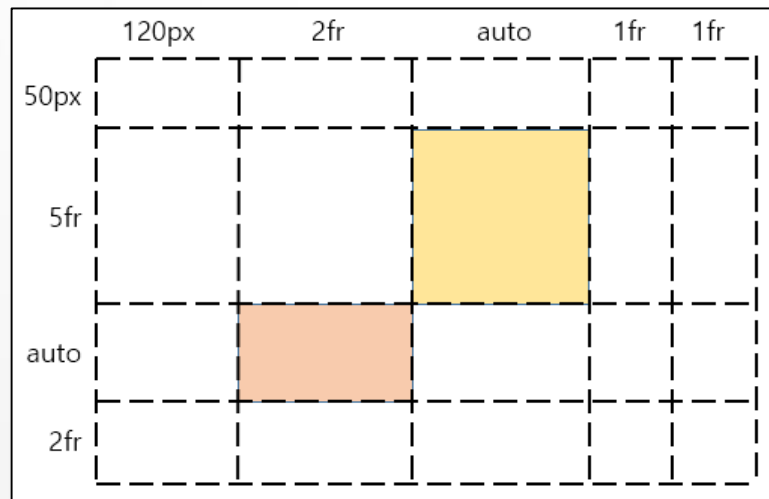


유연한 크기를 나타내는 'fr' 단위

사용 예제

```
#grid { display: grid;  
  grid-template-columns: auto 100px 1fr 2fr;  
  grid-template-rows: 1fr }
```

- 1번째 열(auto 키워드)의 크기는 콘텐츠에 맞게 자동으로 조정된다.
- 2번째 열("100px")의 크기는 100픽셀 크기만큼의 폭을 차지한다.
- 3번째 열("1fr")의 크기는 남은 자유 공간의 1/3만큼의 크기를 차지한다.
- 4번째 열("2fr")의 크기는 남은 자유 공간의 2/3만큼의 크기를 차지한다.



- 3개의 열(2fr, 1fr, 1fr)이 2:1:1의 비율 크기로 지정되며,
- 2개의 행(5fr, 2fr)이 5:2의 비율 크기로 지정된다.



■ 그리드 레이아웃 만들기



트랙(행과 열)의 반복을 나타내는 ‘repeat()’ 표기법

repeat() = repeat(<positive-integer> , [<line-names>? <track-size>]+ <line-names>?)

사용 예제

```
#grid { display: grid;
  grid-template-columns: 10px 250px 10px 250px 10px 250px 10px 250px 10px 5px;
  grid-template-rows: 1fr }
```

```
#grid { display: grid;
  grid-template-columns: 10px repeat(4, 250px 10px) 5px;
  grid-template-rows: 1fr }
```

그리드 라인의 이름 병합

만일, 서로 다른 2개의 그리드 라인의 이름이 인접하여 있다면, 이름들은 병합되어 동일한 그리드 라인의 이름들로 사용된다. 다음 2개의 예는 완전히 동일하다.

```
repeat(2, (a) 1fr (b))
```

```
(a) 1fr (b a) 1fr (b)
```



■ 그리드 레이아웃 만들기



subgrid 키워드

- Subgrid 키워드를 사용하면, 부모 그리드 컨테이너를 기준으로 그리드의 행 또는 열의 크기를 조정할 수 있도록 참여하여, subgrid로 지정된 그리드 항목들의 그리드 콘텐츠가 정렬할 수 있도록 한다

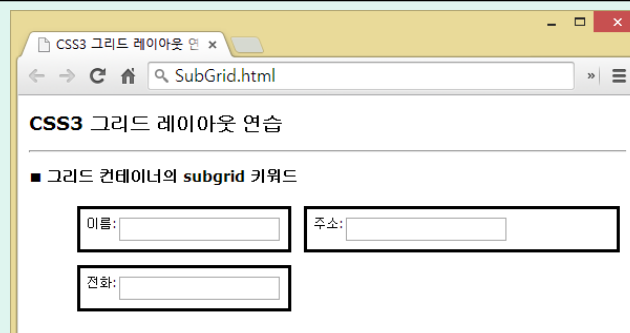
사용 예제	
CSS	<pre>ul { display: grid; grid-auto-flow: row; grid-template-columns: auto 1fr; } li { display: grid; grid: subgrid; border: solid; margin: 0.5em; padding: 0.5em; } label { grid-column: 1; } input { grid-column: 2; }</pre>
HTML	<pre> <label>이름:</label> <input name=fn> <label>주소:</label> <input name=address> <label>전화:</label> <input name=phone> </pre>

예제

CSS3_08-02_SubGrid.html



초기 화면



창의 크기 변화로, 그리드 컨테이너의 2열 크기가 변경됨



■ 그리드 레이아웃 만들기

- 특정 그리드 항목과 관련되지 않은 그리드 영역에 이름을 지정



속성

grid-template-areas

CSS3

속성 값	none <string> +					
속성 특징	초기값	none	적용	그리드 컨테이너들	상속	X

속성 값	설명
none	그리드 컨테이너가 그리드 영역의 이름을 정의하지 않도록 지정한다.
<string> +	행은 속성에 나열된 모든 문자열을 분리하여 생성하고, 열은 각 문자열의 각 셀에 대하여 생성된다.

사용 예제

4개의 그리드 영역 이름을 가진 3개의 행과 2개의 열을 생성하는 예제

```
#grid { display: grid;
  grid-template-areas: "head head"
                      "nav main"
                      "foot . "
}
```

```
#grid > header { grid-area: head; }
#grid > nav    { grid-area: nav; }
#grid > main   { grid-area: main; }
#grid > footer { grid-area: foot; }
```

- head 영역은 1개의 행과 2개의 영역에 걸쳐 있는 것을 볼 수 있다



■ 그리드 레이아웃 만들기

- `grid-template-rows`, `grid-template-columns`, 그리고 `grid-template-areas` 속성을 동시에 모두 설정하는 속성



속성

grid-template

CSS3

속성 값	none subgrid <'grid-template-columns'> / <'grid-template-rows'> [<track-list> /]? [<line-names>? <string> <track-size>? <line-names>?]+					
속성 특징	초기값	개별 속성 참조	적용	그리드 컨테이너들	상속	개별 속성 참조

속성 값	설명
none	3가지 각 속성들의 값을 none으로 설정한다.
subgrid	grid-template-rows와 grid-template-columns 속성의 값을 subgrid로 설정하고 grid-template-areas 속성의 값을 초기값(none)으로 설정한다.

<'grid-template-columns'> / <'grid-template-rows'>

grid-template: **auto 1fr auto** / *auto 1fr*;

grid-template-columns: **auto 1fr auto**;
grid-template-rows: **auto 1fr**;
grid-template-areas: none;

[<track-list> /]? [<line-names>? <string> <track-size>? <line-names>?]+

grid-template: auto 1fr auto /
(header-top) "a a a" (header-bottom)
(main-top) "b b b" 1fr (main-bottom);

grid-template-columns: auto 1fr auto;
grid-template-rows: (header-top) auto (header-bottom)
main-top) 1fr (main-bottom);
grid-template-areas: "a a a"
"b b b";



■ 그리드 레이아웃 만들기

➤ 자동 생성된 행 및 열의 크기 조정



속성

grid-auto-columns, grid-auto-rows

CSS3

속성 값	<track-size>					
속성 특징	초기값	auto	적용	그리드 컨테이너들	상속	X

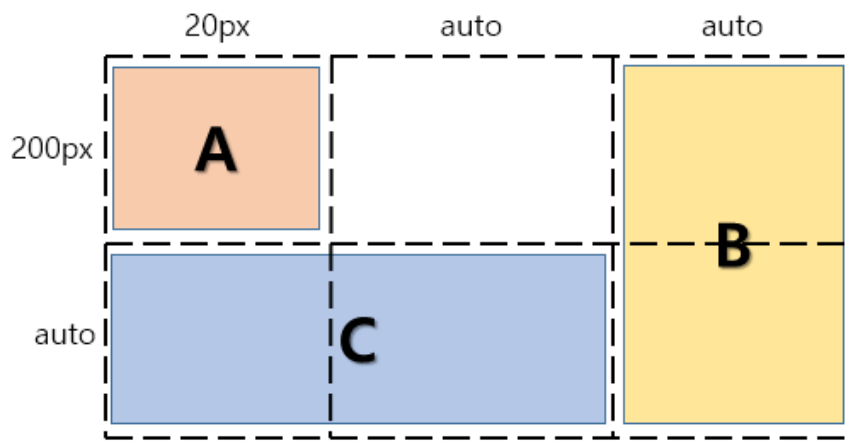
예제

CSS3_08-02_GridAutoSpan.html

※ 아직은 주요 브라우저에서 정상적으로 동작하지 않는다.

```
#grid { display: grid;
        grid-template-columns: 20px;
        grid-template-rows: 20px;
      }
#A { grid-column: 1;      grid-row: 1 }
#B { grid-column: 5;      grid-row: 1 / span 2 }
#C { grid-column: 1 / span 2; grid-row: 2 }
```

```
<div id="grid">
  <div id="A">A</div>
  <div id="B">B</div>
  <div id="C">C</div>
</div>
```





■ 그리드 레이아웃 만들기

➤ 그리드 항목을 자동으로 배치할 수 있도록 하는 속성



속성

grid-auto-flow

CSS3

속성 값	[row column] && dense? stack && [row column]?					
속성 특징	초기값	auto	적용	그리드 컨테이너들	상속	X

속성 값	설명
row	항목을 배치할 수 있도록 필요에 따라서 자동으로 행을 추가하도록 지정한다.
column	항목을 배치할 수 있도록 필요에 따라서 자동으로 열을 추가하도록 지정한다.
dense	항목을 자동으로 배치하도록 하는 "dense" 패킹 알고리즘을 사용하도록 지정한다.
stack	모든 항목들이 겹쳐져서 배치하도록 지정한다. 이 값은 자동 배치 알고리즘을 하나의 가상 1x1 그리드 항목으로 실행하기 때문에 속성 값이 row 또는 column이 지정되어 있다면, 항목은 행이나 열에 따라서 그 옆에 지정된다. row와 column이 모두 지정되어 있다면 기본적으로 row가 지정된 것으로 처리된다.



■ 그리드 레이아웃 만들기

- 그리드 항목을 자동으로 배치할 수 있도록 하는 속성



속성

grid-auto-flow

CSS3

예제

CSS3_08-02_GridForm.html

```

form { display: grid;
      grid-template-columns: (labels) auto (controls) auto (oversized) auto;
      grid-auto-flow: row } /* 자동으로 행이 추가될 수 있도록 한다 */
form > label { grid-column: labels; grid-row: auto }
form > input, form > select { grid-column: controls; grid-row: auto }
#department { grid-column: oversized; grid-row: span 3; }
#buttons { grid-row: auto; grid-column: 1 / -1; }

<form>
  <label for="name">이 름:</label> <input type="text" id="name" name="name" />
  <label for="address">주 소:</label> <input type="text" id="address" name="address" />
  <label for="city">도 시:</label> <input type="text" id="city" name="city" />
  <label for="tel">전 화:</label> <input type="text" id="tel" name="tel" />
  <div id="department">
    <label for="department">부 서:</label>
    <select id="department" name="department" multiple>
      <option value="finance">총무부</option>
      <option value="humanresources">관리부</option>
      <option value="marketing">홍보부</option>
    </select>
  </div>
  <div id="buttons">
    <button id="cancel">취 소</button>
    <button id="back">이 전</button>
    <button id="next">다 음</button>
  </div>
</form>

```

이 름:	<input type="text"/>	부 서:
주 소:	<input type="text"/>	총무부 관리부 홍보부
도 시:	<input type="text"/>	
전 화:	<input type="text"/>	
<input type="button" value="이 전"/> <input type="button" value="취 소"/> <input type="button" value="다 음"/>		



■ 그리드 레이아웃 만들기

➤ 그리드 레이아웃 속성 한번에 지정



속성

grid

CSS3

속성 값	<'grid-template'> [<'grid-auto-flow'> [<'grid-auto-columns'> [/ <'grid-auto-rows'>]?]?]					
속성 특징	초기값	개별 속성 참조	적용	그리드 컨테이너들	상속	개별 속성 참조

사용 예제

'grid: rows 1fr '는 다음과 동일한 내용이다.	'grid: columns 1fr / auto '는 다음과 동일한 내용이다.
<pre>grid-template: none; grid-auto-columns: 1fr; grid-auto-rows: 1fr; grid-auto-flow: row;</pre>	<pre>grid-template: none; grid-auto-columns: 1fr; grid-auto-rows: auto; grid-auto-flow: column;</pre>



■ 그리드 항목 배치하기

- 그리드 항목들의 배치는 다음과 같이 위치(position)와 확장(span)으로

그려진다.

그리드의 위치	그리드 내에서의 그리드 항목의 위치를 나타내는 것으로, 위치는 명시적인 지정(특정 위치 지정) 및 암시적인 지정((자동 배치)에 의해 결정될 수 있다.
그리드의 확장(또는 병합)	그리드 항목이 얼마나 많은 그리드 트랙을 차지하는지를 나타내는 것으로, table 요소의 colspan 및 rowspan과 동일한 의미를 갖는다.

	행(row)	열(column)
Start	row-start 라인	column-start 라인
End	row-end 라인	column-end 라인
Span	row span	column span



그리드 배치에 대한 공통적인 패턴

grid-area			
grid-column		grid-row	
grid-column-start	grid-column-end	grid-row-start	grid-row-end



■ 그리드 항목 배치하기



그리드 배치에 대한 공통적인 패턴



영역 이름을 통한 배치

사용 예제

```
article { grid-area: main }
```

```
.one { grid-row-start: main }
```



수치 인덱스 및 확장을 통한 배치

사용 예제

※ 다음의 예제는 'grid-area: 2 / 3' 와 동일한 의미를 갖는다

```
.two {  
  grid-row: 2; /* 그리드 항목을 2번째 행에 위치시킨다. */  
  grid-column: 3; /* 그리드 항목을 3번째 열에 위치시킨다. */  
}
```

사용 예제

```
.three { grid-row: 2 / span 5 }
```

```
.four { grid-row: span 5 / 7 }
```



■ 그리드 항목 배치하기



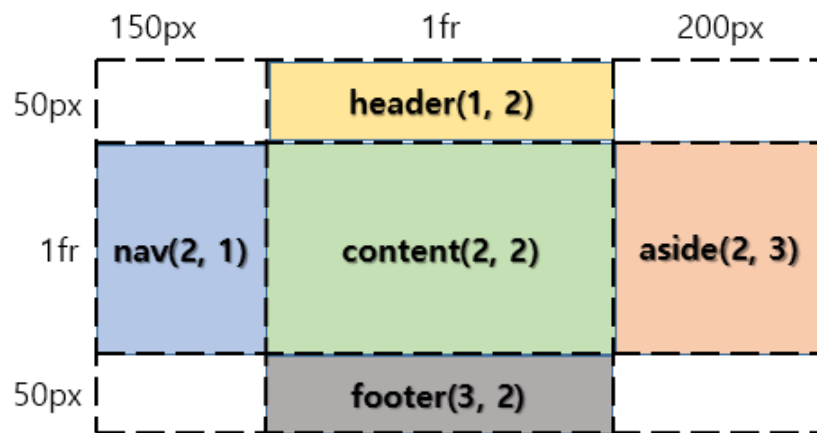
그리드 배치에 대한 공통적인 패턴



수치 인덱스 및 확장을 통한

배치

예제	CSS3_08-02_GridPosition.html
<pre>section { display: grid; grid-template-columns: 150px 1fr 200px; /* 3열 정의 */ grid-template-rows: 50px 1fr 50px; /* 3행 정의 */ }</pre>	<pre>section header { grid-row: 1; grid-column: 2 } section nav { grid-row: 2; grid-column: 1 } section content { grid-row: 2; grid-column: 2 } section aside { grid-row: 2; grid-column: 3 } section footer { grid-row: 3; grid-column: 2 }</pre>





■ 그리드 항목 배치하기



그리드 배치에 대한 공통적인 패턴



그리드 라인 이름 및 확장을 통한 배치

사용 예제

<code>.five { grid-column: first / middle }</code>	<code>.six { grid-row: text 5 / text 7 }</code>
	<code>.six { grid-row: text 5 / span text 2 }</code>



그리드 항목의 자동 배치

사용 예제

<code>.eight { grid-area: auto } /* 초기 값 */</code>	<code>.nine { grid-area: span 2 / span 3 }</code>
--	---



자동으로 subgrids 크기 조정

사용 예제


<code>.adverts { grid: subgrid; grid-column: span 2 }</code>
--

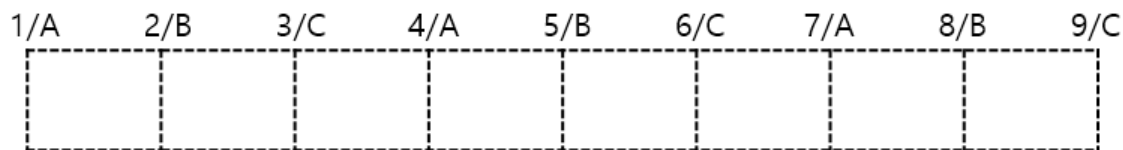


그리드 항목 배치하기



그리드 라인을 기준으로 한 배치

 속성	grid-row-start, grid-column-start, grid-row-end, grid-column-end					CSS3
속성 값	<grid-line>					
	< grid-line > = auto <custom-ident> [<integer> && <custom-ident>?] [span && [<integer> <custom-ident>]]					
속성 특징	초기값	auto	적용	그리드 항목들	상속	X



사용 예제

grid-column-start: 4; grid-column-end: auto;	/* 라인 4에서 5까지 */
grid-column-start: auto; grid-column-end: 6;	/* 라인 5에서 6까지 */
grid-column-start: C; grid-column-end: C -1;	/* 라인 3에서 9까지 */
grid-column-start: C; grid-column-end: span C;	/* 라인 3에서 6까지 */
grid-column-start: span C; grid-column-end: C -1;	/* 라인 6에서 9까지 */
grid-column-start: 5; grid-column-end: C -1;	/* 라인 5에서 9까지 */
grid-column-start: 5; grid-column-end: span C;	/* 라인 5에서 6까지 */
grid-column-start: 8; grid-column-end: 8;	/* 오류(라인 8에서 9까지) */
grid-column-start: B 2; grid-column-end: span 1;	/* 라인 5에서 6까지 */



■ 그리드 항목 배치하기



단축 속성에 의한 배치

✓ 속성	grid-row, grid-column					CSS3
속성 값	<grid-line> [/ <grid-line>] ?					
속성 특징	초기값	개별 속성 참조	적용	그리드 항목들	상속	개별 속성 참조

- grid-row-start/grid-row-end 속성과 grid-column-start/grid-column-end 속성들을 각각 한번에 지정할 수 있는 단축 속성

✓ 속성	grid-area					CSS3
속성 값	<grid-line> [/ <grid-line>] { 0, 3 }					
속성 특징	초기값	개별 속성 참조	적용	그리드 항목들	상속	개별 속성 참조

- grid-row-start/grid-row-end 속성과 grid-column-start/grid-column-end 속성들 4개 모두를 한번에 지정할 수 있는 단축 속성



■ 그리드 항목 배치하기



절대 위치에 의한 배치

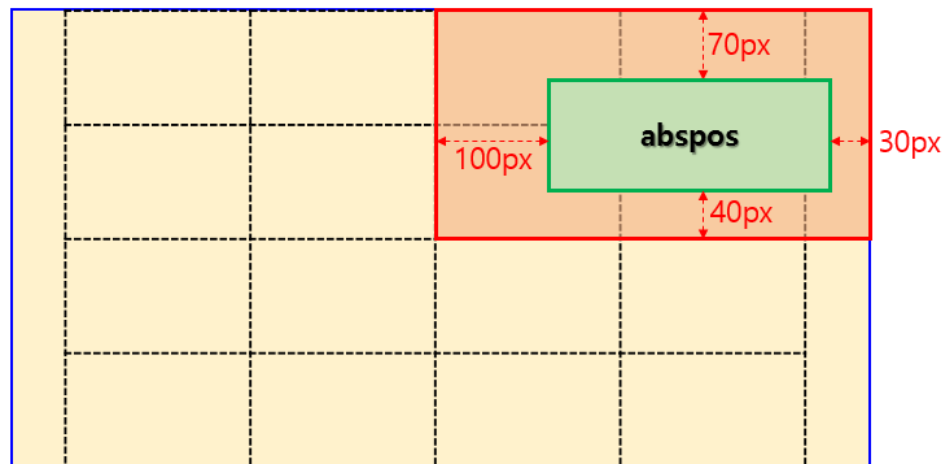
- 절대 위치에 대한 오프셋 속성들(top/right/bottom/left)은 일반적으로 포함 블록의 경계로부터 안쪽 방향으로의 오프셋을 의미한다.

사용 예제

```
.grid {  
  grid: 10rem 10rem 10rem 10rem / 1fr 1fr 1fr 1fr;  
  justify-content: center;  
  position: relative;  
}
```

```
.abspos {  
  grid-row-start: 1; grid-row-end: span 2;  
  grid-column-start: 3; grid-column-end: auto;  
  position: absolute;  
  top: 70px; bottom: 40px; left: 100px; right: 30px;  
}
```

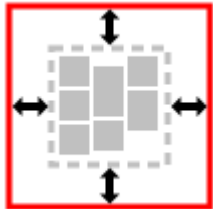
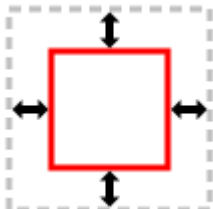
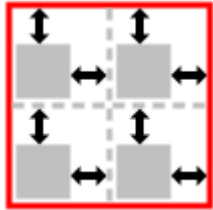
그리드 컨테이너





■ 그리드 정렬하기

표준화 문서	CSS Box Alignment Module Level 3 - http://dev.w3.org/csswg/css-align/
표준화 단계	W3C Editor's draft, (2014-05-19)

공통 속성	축(Axis)	정렬		적용 범위
justify-content	inline	요소 안에서의 콘텐츠 (패딩이 효과적으로 적용된다)		블록(Block) 컨테이너 유연한(Flexible) 컨테이너 그리드 컨테이너
align-content	stacking			
justify-self	inline	부모 안에서의 요소 (여백이 효과적으로 적용된다)		블록-레벨(Block-level) 요소 절대 위치에 의한 박스 그리드 항목
align-self	stacking			블록-레벨(Block-level) 박스
justify-items	inline	요소 내부에서의 항목들(자식 항목들을 제어한다 - 'align/justify-self:auto')		블록(Block) 컨테이너 유연한(Flexible) 컨테이너 그리드 컨테이너
align-items	stacking			블록-레벨(Block-level) 요소



■ 그리드 정렬하기

키워드	내용	적용
center	정렬할 내용을 중앙으로 위치시킨다.	항목 위치 (item-position) 및 콘텐츠 위치 (content-position)
start	정렬할 컨테이너의 시작 경계와 같은 높이로 위치 시킨다..	
end	정렬할 컨테이너의 마지막 경계와 같은 높이로 위치 시킨다.	
flex-start	유연한 레이아웃에만 적용되는 값으로, 그 이외의 레이아웃에서는 start로 계산된다. 유연한 컨테이너의 main-start 또는 cross-start 측면에 대응하는 정렬 컨테이너의 경계와 같은 높이로 위치 시킨다.	
flex-end	유연한 레이아웃에만 적용되는 값으로, 그 이외의 레이아웃에서는 end로 계산된다. 유연한 컨테이너의 main-end 또는 cross-end 측면에 대응하는 정렬 컨테이너의 경계와 같은 높이로 위치 시킨다.	
left	정렬 컨테이너의 라인-왼쪽(line-left) 측면과 같은 높이로 위치 시킨다. 만일, 정렬하는 축과 인라인 축이 평행하지 않다면 이 값은 start로 계산된다.	
right	정렬 컨테이너의 라인-오른쪽(line-right) 측면과 같은 높이로 위치 시킨다. 만일, 정렬하는 축과 인라인 축이 평행하지 않다면 이 값은 end로 계산된다.	항목 위치 (item-position)
self-start	정렬할 내용의 시작 측면에 대응하는 정렬 컨테이너의 경계와 같은 높이로 위치 시킨다. 만일, 정렬할 내용의 쓰기 모드와 정렬할 컨테이너가 직교한다면 이 값은 start로 계산된다.	
self-end	정렬할 내용의 끝 측면에 대응하는 정렬 컨테이너의 경계와 같은 높이로 위치 시킨다. 만일, 정렬할 내용의 쓰기 모드와 정렬할 컨테이너가 직교한다면 이 값은 end로 계산된다.	



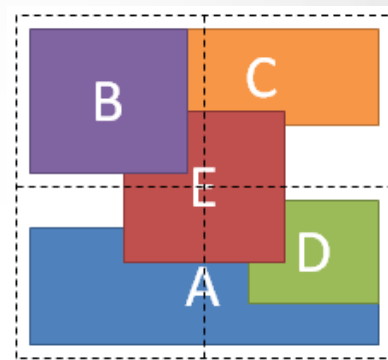
■ 그리드 항목의 겹침 우선 순위

- z-index 속성은 그리드 셀 내의 그리드 항목의 계층(z-index)을 지정

사용 예제

```
#grid { display: grid;
  grid-template-columns: 1fr 1fr; grid-template-rows: 1fr 1fr }
#A { grid-column: 1 / span 2; grid-row: 2; align-self: end }
#B { grid-column: 1; grid-row: 1; z-index: 10 }
#C { grid-column: 2; grid-row: 1; align-self: start; margin-left: -20px }
#D { grid-column: 2; grid-row: 2; justify-self: end; align-self: start; }
#E { grid-column: 1 / span 2; grid-row: 1 / span 2; z-index: 5;
  justify-self: center; align-self: center; }
```

```
<div id="grid">
  <div id="A">A</div>
  <div id="B">B</div>
  <div id="C">C</div>
  <div id="D">D</div>
  <div id="E">E</div>
</div>
```





표준화 문서	CSS Basic User Interface Module Level 3(CSS3 UI) - http://dev.w3.org/csswg/css-ui/
표준화 단계	W3C Editor's draft (2013-11-28)

■ 요소 아이콘(Element Icons)



속성

content – 콘텐츠 추가 지정하기

CSS2/CSS3

- 선택자 가상 요소인 **:before** 및 **:after**를 이용하여 특정 요소의 앞이나 뒤에 콘텐츠를 추가할 수 있도록 지정
 - ✓ 텍스트를 포함하여 이미지 또는 특정 규칙에 따른 값을 지정할 수도 있고,
 - ✓ CSS3에서 새롭게 추가된 icon을 사용할 수도 있다.

속성 값	[<string> <uri> <counter> attr(X) open-quote close-quote no-open-quote no-close-quote] icon] + inherit					
속성 특징	초기값	빈(empty) 문자열	적용	가상 요소(:before 및 :after)	상속	X
사용 예제	h1:before { content : "Read me –" }					



■ 요소 아이콘(Element Icons)



속성

content – 콘텐츠 추가 지정하기

CSS2/CSS3

속성 값	설명
<i>string</i>	문자열 내용으로, 이중 따옴표 또는 단일 따옴표들과 함께 쓰여질 수 있다
<i>uri</i>	외부(external) 자원을 지정하는 URI를 이용하여 콘텐츠를 추가하도록 한다
<i>counter</i>	카운터 값을 콘텐츠로 추가하도록 한다. 자세한 내용은 counter-reset 및 counter-increment 속성을 참고하도록 한다.
<i>attr(X)</i>	선택자(selector)의 맞는 제목(subject)을 위한 속성(attribute) X의 값을 문자열로 돌려받아 콘텐츠로 추가한다. 문자열은 CSS 처리자(processor)에 의해 해석(parse)되지 않으며, 선택자의 맞는 제목이 속성 X를 가지고 있지 않다면 빈 문자열이 된다.
<i>open-quote</i> <i>close-quote</i>	따옴표('quotes') 속성으로부터의 적당한 문자열로 대체되도록 한다.
<i>no-open-quote</i> <i>no-close-quote</i>	빈(empty)을 삽입하는 것과 같이 아무것도 삽입하지 않으나, 내포된(nest) 따옴표의 레벨을 증가 혹은 감소 시킨다.
CSS3 <i>icon</i>	뒤에서 언급할 icon 속성에 의해 참조된 자원(resource)에 의해 대체되고 대체된 요소로써 처리된다.

➤ 장(Chapter)들과 항목들의 번호를 "Chapter 1", "1.1", "1.2" 등으로 번호 붙이는 것을 나타낸 예제

```
H1:before {
  content: "Chapter " counter(chapter) ". ";
  counter-increment: chapter; /* chapter를 1 증가 */
  counter-reset: section; /* 항목을 0 으로 설정*/
}
H2:before {
  content: counter(chapter) "." counter(section) " ";
  counter-increment: section; /* section을 1 증가 */
}
```



■ 요소 아이콘(Element Icons)



속성

content – 콘텐츠 추가 지정하기

CSS3

- 콘텐츠 저작자가 임의의 요소에 스타일링을(iconic) 할 수 있는 기능을 제공

속성 값	auto <uri> [, <uri>]* inherit				
속성 특징	초기값	auto	적용	모든 요소들	상속
JavaScript	<code>object.style.icon= "url(image.png)"</code>				
사용 예제	<pre>img { content: icon; icon: url(img_icon.png) }</pre>				

속성 값	설명
<i>auto</i>	웹 브라우저(UA)가 제공하는 기본적인 일반 아이콘을 사용하도록 지정한다.
<i>uri</i>	외부(external) 자원을 지정하는 URI를 이용하여 쉼표(,)로 구분된 목록에서 하나 이상의 아이콘을 참조하도록 지정한다.

```
img,object { content: icon }
img { icon:url(img_icon.png) } /* 이미지에 대한 사용자 아이콘을 지정한다 */
object { icon:url(obj_icon.png) } /* 객체에 대한 다른 사용자 아이콘을 지정한다 */
```



■ 추가적인 박스 모델



속성

box-sizing - 박스 크기 결정 방식 지정하기

CSS3

■ 해당 요소의 박스 크기를 결정하는 방식을 지정

속성 값	content-box padding-box border-box inherit					
속성 특징	초기값	content-box	적용	폭(width)과 높이(height)를 허용하는 모든 요소들	상속	X
JavaScript	<code>object.style.boxSizing = "border-box"</code>					
사용 예제	<code>div { box-sizing: border-box }</code>					

속성 값	설명
<i>content-box</i>	CSS2.1에 지정된 폭과 높이로써, 요소 박스에 지정된 패딩 및 테두리의 크기 값을 더해서 실제 박스의 크기를 결정하도록 지정한다.
<i>padding-box</i>	패딩 영역까지의 크기를 실제 박스 크기로 결정하도록 지정한다(명세서에서 제외될 예정).
<i>border-box</i>	테두리 영역까지의 크기를 실제 박스 크기로 결정하도록 지정한다.



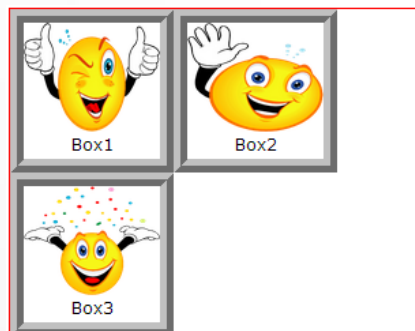
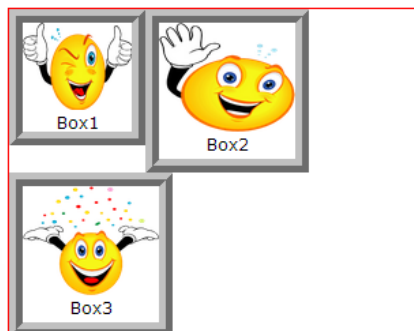

■ 추가적인 박스 모델



속성

box-sizing - 박스 크기 결정 방식 지정하기

CSS3

예제	CSS3_08-03_BoxSizing.html		
Style	#myDIV { width: 300px; border:1px solid red; } .Box { width: 100px; height: 100px; text-align: center; float: left; padding 10px; border: 10px silver ridge; }		
Body	<div id="myDIV"> <div class="Box" id="Box1"> Box1 </div> <div class="Box" id="Box2"> Box2 </div> <div class="Box" id="Box3"> Box3 </div> <div style="clear:both;"> </div> </div>		
			
			
box-sizing: content-box		Box1 - box-sizing: border-box	
		Box1,2,3 - box-sizing: border-box	



■ 박스의 윤곽선 지정하기

✓ 속성 **outline-width** – 윤곽선의 굵기 지정 **CSS2**

속성 값	<border-width> inherit					
속성 특징	초기값	medium	적용	모든 요소들	상속	X
JavaScript	<i>object.style.outlineWidth</i> = "10px"					
사용 예제	div { outline-width : 10px }					

✓ 속성 **outline-style** – 윤곽선의 모양을 지정 **CSS2**

속성 값	auto <border-style> inherit					
속성 특징	초기값	none	적용	모든 요소들	상속	X
JavaScript	<i>object.style.outlineStyle</i> = "double"					
사용 예제	div { outline-style : double }					

✓ 속성 **outline-color** – 윤곽선의 색상을 지정 **CSS2**

속성 값	<color> invert inherit					
속성 특징	초기값	invert	적용	모든 요소들	상속	X
JavaScript	<i>object.style.outlineColor</i> = "rgba(255, 0, 0, 0.7)"					
사용 예제	div { outline-color : rgba(255, 0, 0, 0.7) }					



■ 박스의 윤곽선 지정하기



속성

outline - 3개의 윤곽선 속성을 지정하는 대표 속성

CSS2

속성 값	[<'outline-color'> <'outline-style'> <'outline-width'>] inherit					
속성 특징	초기값	개별 속성 참조	적용	모든 요소들	상속	X
JavaScript	<i>object.style.outline</i> = "10px dashed rgba(255, 0, 0, 0.7)"					
사용 예제	div { outline : 10px dashed rgba(255, 0, 0, 0.7) }					

속성 값	설명
border-width	border-width 속성과 동일한 값들을 지정할 수 있다.
border-style	border-style 속성과 동일한 값들을 지정할 수 있다(단, hidden 값은 지정할 수 없다).
color	모든 색상들 및 색상 키워드를 지정할 수 있다.
invert	윤곽선의 색상에 지정하는 값으로, 스크린 상에서의 색상을 역(inversion)으로 만들게 한다. 이는 초점(focus) 테두리가 어떤 색상 배경이던 관계없이 확실히 보이게 한다.



■ 박스의 윤곽선 지정하기

예제

CSS3_08-03_OutlineWidth.html

**outline-width:** thin**outline-width:** thick**outline-width:** medium**outline-width:** 10px

예제

CSS3_08-03_OutlineStyle.html

**outline-style:** dotted**outline-style:** thick**outline-style:** medium**outline-style:** 10px



■ 박스의 윤곽선 지정하기



속성

outline-offset – 윤곽선 간격 지정하기

CSS3

속성 값	<length> inherit				
속성 특징	초기값	0	적용	모든 요소들	상속 X
JavaScript	<code>object.style.outlineOffset= "10px"</code>				
사용 예제	<code>div { outline-offset: 10px }</code>				

예제

CSS3_08-03_OutlineOffset.html



outline-offset: 10px



outline-offset: 20px



outline-offset: -10px



outline-offset: -20px



■ 박스 크기 재조정(Resizing) 및 넘침(Overflow)



속성

resize – 요소 박스 크기 재조정하기

CSS3

속성 값	none both horizontal vertical inherit					
속성 특징	초기값	none	적용	overflow 속성(visible이 아닌)을 가진 요소들	상속	X
JavaScript	<code>object.style.resize= "10px"</code>					
사용 예제	div { resize : both }					

속성 값	설명
none	사용자는 요소의 크기를 조절할 수 없도록 지정한다. 브라우저는 요소의 크기를 조절하는 도구를 제공하지 않는다.
both	사용자가 박스의 가로 및 세로 크기를 조절하도록 허용한다.
horizontal	사용자가 박스의 가로 크기를 조절하도록 허용한다.
vertical	사용자가 박스의 세로 크기를 조절하도록 허용한다.



■ 박스 크기 재조정(Resizing) 및 넘침(Overflow)



속성

resize - 요소 박스 크기 재조정하기

CSS3

예제

CSS3_08-03_Resize.html



resize 속성은 사용자로 하여금 요소 박스의 크기를 조절하게 할 것인지를 지정하는 속성으로, overflow 속성이 지정되어 있어야 한다.



resize 속성은 사용자로 하여금 요소 박스의 크기를 조절하게 할 것인지를 지정하는 속성으로, overflow 속성이 지정되어 있어야 한다.



resize 속성은 사용자로 하여금 요소 박스의 크기를 조절하게 할 것인지를 지정하는 속성으로, overflow 속성이 지정되어 있어야 한다.



resize 속성은 사용자로 하여금 요소 박스의 크기를 조절하게 할 것인지를 지정하는 속성으로, overflow 속성이 지정되어 있어야 한다.

resize: none

resize: horizontal

resize: horizontal (크기 조절 후)



resize 속성은 사용자로 하여금 요소 박스의 크기를 조절하게 할 것인지를 지정하는 속성으로, overflow 속성이 지정되어 있어야 한다.



resize 속성은 사용자로 하여금 요소 박스의 크기를 조절하게 할 것인지를 지정하는 속성으로, overflow 속성이 지정되어 있어야 한다.



resize 속성은 사용자로 하여금 요소 박스의 크기를 조절하게 할 것인지를 지정하는 속성으로, overflow 속성이 지정되어 있어야 한다.



resize 속성은 사용자로 하여금 요소 박스의 크기를 조절하게 할 것인지를 지정하는 속성으로, overflow 속성이 지정되어 있어야 한다.

resize: both

resize: vertical

resize: vertical (크기 조절 후)



■ 박스 크기 재조정(Resizing) 및 넘침(Overflow)



속성

text-overflow – 텍스트의 줄임 지정하기

CSS3

속성 값	(clip ellipsis <string> {1,2} inherit				
속성 특징	초기값	clip	적용	블록 요소들	상속
JavaScript	object.style.textOverflow= "ellipsis"				
사용 예제	div { text-overflow: ellipsis }				

속성 값	설명
clip	박스 영역에 맞추어 텍스트의 글자를 자르도록 지정한다.
ellipsis	박스 영역 외곽으로 벗어나서 잘리는 끝 부분에 자동으로 "...로 표시하여 말 줄임을 표시하도록 지정한다.
string	박스 영역을 벗어나 텍스트가 잘린 부분에 지정한 문자열이 대신 나타나도록 지정한다.

예제	CSS3_08-03_TextOverflow.html	
Style	#myDIV { overflow: hidden; white-space: nowrap; text-overflow: ellipsis }	
Body	<div id="myDIV">text-overflow 속성은 요소 박스의 폭이 고정되어 있고 요소 박스 내의 텍스트가 박스 영역에서 벗어났을 때 어떤 방식으로 보여줄지를 지정한다.</div>	
text-overflow 속성은 요소 박스의 폭이		text-overflow 속성은 요소 박스의 폭...
text-overflow: clip		text-overflow: ellipsis



■ 박스 크기 재조정(Resizing) 및 넘침(Overflow)



속성

text-overflow - 텍스트의 줄임 지정하기

CSS3

CSS 속성 값	direction: ltr		direction: rtl	
	기대한 결과	실제 결과	기대한 결과	실제 결과
visible overflow	1234567890	1234567890	0987654321	0987654321
text-overflow: clip	123456	123456	654321	7654321
text-overflow: clip clip	123456	1234567	654321	7654321
text-overflow: clip ellipsis	1234...	1234567	6543...	7654321
text-overflow: clip .	1234.	1234567	6543...	7654321
text-overflow: ellipsis	1234...	1234...	...4321	...4321
text-overflow: ellipsis ellipsis	...34...	1234567	...43...	7654321
text-overflow: ellipsis clip	...3456	1234567	...4321	7654321
text-overflow: ellipsis '	...34.	1234567	...43.	7654321
text-overflow: ', clip	,3456	1234567	,4321	7654321
text-overflow: ', ellipsis	,34...	1234567	,43...	7654321
text-overflow: ', '	,34.	1234567	,53.	7654321

(참고 - <https://developer.mozilla.org/en-US/docs/Web/CSS/text-overflow>)



■ 포인팅 디바이스와 키보드






속성

cursor – 포인터 상호작용하기

CSS2/CSS3

속성 값	[[<uri> [<x> <y>]?,* [auto default none context-menu help pointer progress wait cell crosshair text vertical-text alias copy move no-drop not-allowed e-resize n-resize ne-resize nw-resize s-resize se-resize sw-resize w-resize ew-resize ns-resize nesw-resize nwse-resize col-resize row-resize all-scroll zoom-in zoom-out]] inherit				
속성 특징	초기값	auto	적용	모든 요소들	상속 O
JavaScript	<code>object.style.cursor= "url('images/smile_s.png'), url('images/myCursor.cur')"</code>				
사용 예제	<code>div { cursor: url(images/smile_s.png), url('images/myCursor.cur') }</code>				

커서 종류	Level	속성 값	설명
image 커서	CSS2	uri	URI에서 지정한 커서로 표시되도록 지정한다. 
	CSS3	x, y	커서의 좌표 체계(왼쪽 상단부터 상대적인)에 있는 x좌표 및 y좌표를 나타낸다.
일반 커서	CSS2	auto	현재의 문맥에 기초하여 브라우저가 표시될 커서를 결정하도록 한다. 
	CSS2	default	플랫폼에 따른 기본 커서로써, 일반적인 화살표로 표시된다. 
	CSS3	none	요소에 대한 커서를 보이지 않게 지정한다.



■ 포인팅 디바이스와 키보드



속성

cursor – 포인터 상호작용하기

CSS2/CSS3

커서 종류	Level	속성 값	설명	
연결 및 상태 커서	CSS2	wait	프로그램이 작동 중이어서 사용자가 기다려야 함을 나타내는 모양으로 지정. 일반적으로 시계 또는 모래시계로 표시된다.	
	CSS2	help	커서에 객체(object)가 있을 때 나타나는 도움말 모양으로 지정. 일반적으로 물음표 또는 풍선으로 표시된다.	
	CSS2	pointer	연결(link)을 가리키는 모양으로 지정.	
	CSS3	progress	진행 표시기가 나타나도록 지정. 일반적으로 시계 또는 모래시계를 가진 화살표나 회전하는 비치볼 모양이 표시된다.	
	CSS3	context-menu	객체에 대한 상황에 맞는 메뉴가 있을 때 나타나는 모양으로 지정	
선택 커서	CSS2	crosshair	단순한 십자가 형태(예들 들어, "+" 부호와 비슷한 짧은 선)	
	CSS2	text	선택될 수 있는 텍스트를 나타내는 모양으로 표시된다. 일반적으로 I-막대로 표시된다.	
	CSS3	cell	셀 또는 셀 세트가 선택될 수 있음을 표시하는 모양으로 지정한다. 일반적으로 중간에 점이 있는 두꺼운 더하기 기호로 표시된다.	
	CSS3	vertical-text	수직 텍스트가 선택될 수 있음을 표시하는 모양으로 지정. 종종 수평 I-beam으로 표시된다.	
드래그 앤 드롭 커서	CSS2	move	무엇이 움직이는 것을 나타내는 모양으로 지정된다.	
	CSS3	copy	무엇인가 복사할 때 나타내는 모양으로 지정. 일반적으로 일반적으로 옆에 작은 더하기 기호가 있는 화살표로 표시된다.	
	CSS3	alias	무엇인가 만들어질 수 있는 바로 가기의 별칭을 나타내는 모양으로 지정. 일반적으로 옆에 작은 곡선의 화살표가 있도록 표시된다.	
	CSS3	no-drop	드래그 된 항목을 현재 커서 위치에 놓을 수 없음을 나타내는 모양으로 지정. 일반적으로 작은 원을 대각선으로 가로지르는 선의 모양으로 표시된다.	
	CSS3	not-allowed	요청된 작업이 수행되지 않음을 나타내는 모양으로 지정. 일반적으로 원을 대각선으로 가로지른 선의 모양으로 표시된다.	





■ 포인팅 디바이스와 키보드



속성

cursor – 포인터 상호작용하기

CSS2/CSS3

커서 종류	Level	속성 값	설명
크기 재조정 및 스크롤링 커서	CSS2	e-resize, ne-resize, nw-resize, n-resize, se-resize , sw-resize, s-resize, w-resize	일부 모서리가 움직이는 것을 나타내는 모양으로 표시된다. 
	CSS3	ew-resize, ns-resize, nesw-resize, nwse- resize	양방향 크기를 조절할 수 있는 모양으로 표시된다. 
	CSS3	col-resize	항목/열의 가로 크기를 조절할 수 있음을 나타내는 모양으로 표시된다. 일반적으로 항목을 구분하는 왼쪽과 오른쪽 화살표가 있는 세로 막대로 표시된다.
	CSS3	row-resize	항목/행의 세로 크기를 조절할 수 있음을 나타내는 모양으로 표시된다. 일반적으로 항목을 구분하는 위와 아래쪽 화살표가 있는 가로 막대로 표시된다.
	CSS3	all-scroll	어떤 방향으로 스크롤될 수 있음을 나타내는 모양으로 표시된다. 일반적으로 중간에 점이 있는 상하좌우를 가리키는 화살표로 표시된다.
확대 및 축소 커서	CSS3	zoom-in, zoom-out	무엇인가의 크기가 확대 및 축소될 수 있음을 나타내는 모양으로 표시된다. 일반적으로 돋보기 모양의 가운데에 "+" 또는 "-" 기호가 있는 모양으로 표시된다.





■ 포인팅 디바이스와 키보드



속성

nav-index – 요소의 탐색 순서 지정하기

CSS3

속성 값	auto <number> inherit					
속성 특징	초기값	auto	적용	사용 가능한 모든 요소들	상속	O
JavaScript	<i>object.style.navIndex</i> = "1"					
사용 예제	div { nav-index : 1 }					

속성 값	설명
<i>auto</i>	요소의 순차 탐색 순서가 브라우저(사용자 에이전트)에 의해서 자동으로 결정된다.
<i>number</i>	지정한 양수 값으로(0 이상) 요소의 순차 탐색 순서를 지정하도록 한다. 만일 동일한 탐색 순서 값을 가진 요소들이 있다면, 문서의 순서에 의해 탐색된다.



■ 포인팅 디바이스와 키보드



포커스 탐색 방향 지정

- 요소가 포커스를 이동하여 반응하는 순서를 지정



속성	nav-up, nav-right, nav-down, nav-left					CSS3
속성 값	auto <id> [current root <target-name>]? inherit					
속성 특징	초기값	auto	적용	사용 가능한 모든 요소들	상속	X
JavaScript	<code>object.style.navUp= "#up"</code>			<code>object.style.navRight= "#right"</code>		
	<code>object.style.navDown= "#down"</code>			<code>object.style.navLeft= "#left"</code>		
사용 예제	div { nav-up : #up }			div { nav-right : #right }		
	div { nav-down : #down }			div { nav-left : #left }		

속성 값	설명
<i>auto</i>	브라우저의 자동 탐색 입력 방향성에 대한 응답으로 포커스가 이동하도록 지정한다.
id	특정 속성에 각각의 방향 탐색 입력에 대한 응답으로, 탐색되는 요소를 나타내도록 아이디(#) 뒤에 순서를 지정한다. <id>가 현재 포커스를 가진 요소를 참조하는 경우에는 동일한 요소를 포커싱할 필요가 없기 때문에 지정된 값은 무시된다.
target-name	포커스 탐색의 대상 프레임을 지정한다. 밑줄(_)이 아닌 문자열로 값을 지정해야 한다. 지정한 대상 프레임이 없을 경우, 현재 프레임을 의미하는 키워드 <code>current</code> 값으로 취급되고 전체 창을 대상 프레임으로 하려는 경우에는 키워드 <code>root</code> 값을 사용한다.



■ 포인팅 디바이스와 키보드



포커스 탐색 방향 지정

- 요소가 포커스를 이동하여 반응하는 순서를 지정

예제	CSS3_08-03_Nav.html
Style	<pre> button { position:absolute } button#b1 { top:0; left:50%; nav-index:1; nav-right:#b2; nav-left:#b4; nav-down:#b2; nav-up:#b4; } button#b2 { top:50%; left:100%; nav-index:2; nav-right:#b3; nav-left:#b1; nav-down:#b3; nav-up:#b1; } button#b3 { top:100%; left:50%; nav-index:3; nav-right:#b4; nav-left:#b2; nav-down:#b4; nav-up:#b2; } button#b4 { top:50%; left:0; nav-index:4; nav-right:#b1; nav-left:#b3; nav-down:#b1; nav-up:#b3; } </pre>
Body	<pre> <button id="b1">Button 1</button> <button id="b2">Button 2</button> <button id="b3">Button 3</button> <button id="b4">Button 4</button> </pre>
<div> <div>Button 1</div> <div>Button 4</div> <div>Button 2</div> <div>Button 3</div> </div> <div> <div>Button 1</div> <div>Button 4</div> <div>Button 2</div> <div>Button 3</div> </div>	



■ 포인팅 디바이스와 키보드



속성

ime-mode – 입력기 제어하기

CSS3

- 텍스트 필드에 대한 입력기(Input method editor)의 상태를 제어

속성 값	auto normal active inactive disabled inherit					
속성 특징	초기값	auto	적용	텍스트 항목(fields)	상속	O
JavaScript	<code>object.style.imeMode= "active"</code>					
사용 예제	<code>div { ime-mode: active }</code>					

속성 값	설명
<i>auto</i>	기본값으로, 현재의 입력기 상태가 바뀌지 않도록 한다.
<i>normal</i>	페이지 설정을 대체하는 사용자 스타일시트에 사용될 수 있도록 지정한다(IME 상태가 normal 이어야 한다).
<i>active</i>	초기에 입력기를 활성화 상태로 설정하여 텍스트 입력이 가능하도록 지정한다.
<i>inactive</i>	초기에 입력기를 비활성화 상태로 지정하지만, 사용자가 원할 경우에는 활성화 상태로 변경할 수 있다.
<i>disabled</i>	입력기를 사용 불가능하도록 지정한다.

```
<input type="text" name="name" value="initial value" style="ime-mode: disabled">
```