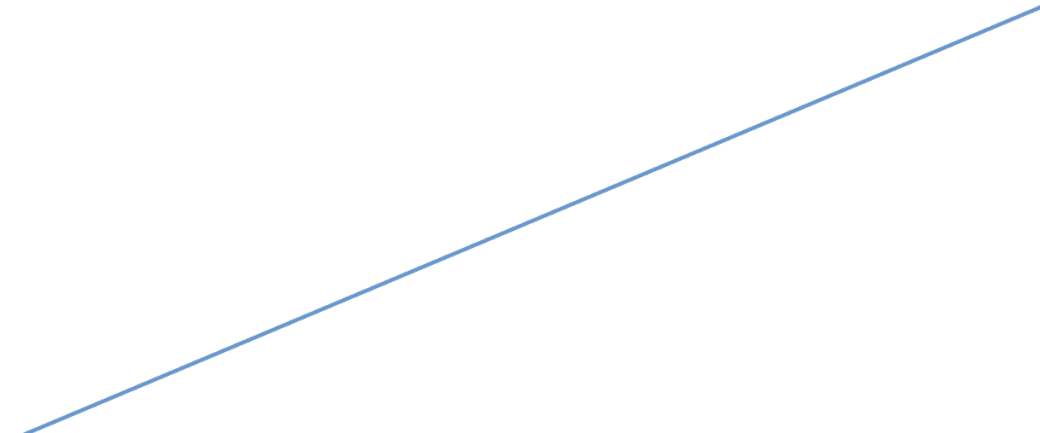


프론트엔드 개발 실무

[5장] CSS 심화

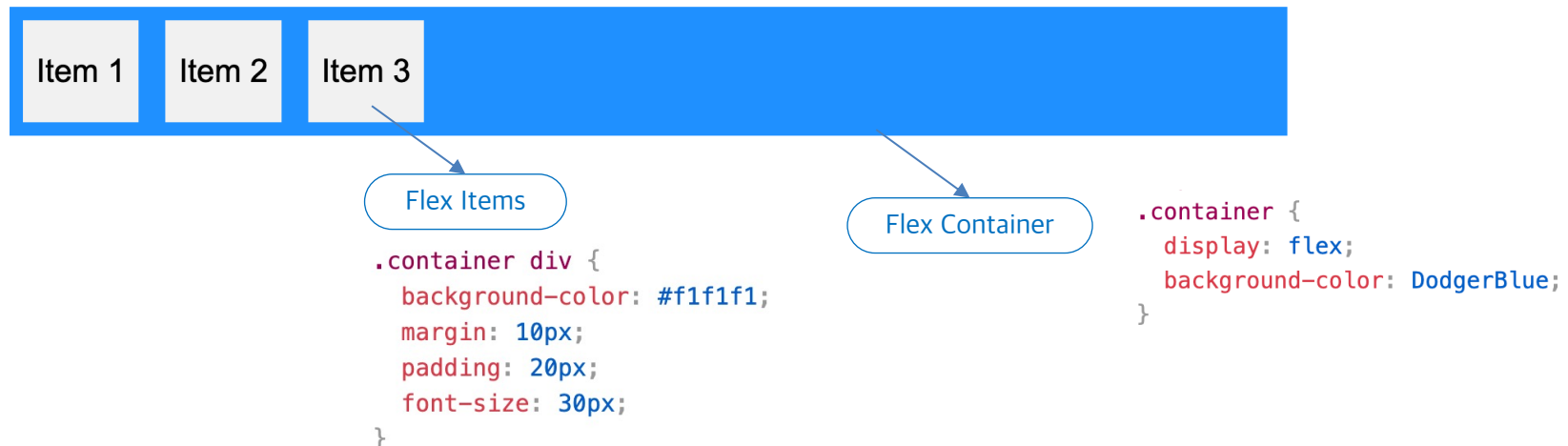
디노웍스
이 시 영

목 차

- Flexbox
 - Grid
 - Media Query
 - Transition
 - Animation
 - CSS 실습
- 

➤ Flexbox 개요

- Flexible Box Layout Module 을 의미
- 컨테이너 내에서 항목을 수평 또는 수직으로 유연한 방식으로 배열하기 위한 레이아웃 모델
- float이나 position 사용 없이 유연하고 반응성이 뛰어난 레이아웃을 쉽게 디자인
- Grid와의 차이점
 - Flexbox는 행 또는 열이 있는 1차원 레이아웃에 사용되며, Grid는 2차원 레이아웃에 사용
- Flexbox의 구성
 - Flex Container : display 속성이 flex 혹은 inline-flex 로 지정된 부모(container) 요소
 - Flex Items : Flex container의 직계 자식 요소가 자동으로 지정됨



➤ Flex Container

○ 속성(Property)

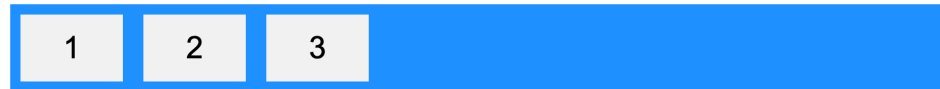
속성명	설 명	비 고
display	flex 혹은 inline-flex로 지정되어야 함	
flex-direction	Flex 항목의 표시 방향을 설정	
flex-wrap	Flex 항목을 래핑할지 여부를 결정	
flex-flow	flex-direction과 flex-wrap 속성을 동시에 지정할 수 있는 축약 표현	
justify-content	주축(수평)에서 사용 가능한 모든 공간을 사용하지 않을 때 flex item을 정렬	
align-items	교차 축(수직)에서 사용 가능한 모든 공간을 사용하지 않을 때 flex item을 정렬 (한 줄의 내부)	
align-content	교차 축에 여유 공간이 있고 flex item이 래핑되는 경우 flex 라인을 정렬 (여러 줄을 포함한 그룹)	

➤ Flex Container

○ flex-direction

- row(default)
- column
- row-reverse
- column-reverse

row



column



row-reverse



column-reverse



```
.flex-container {  
  display: flex;  
  flex-direction: row;  
}
```

➤ Flex Container

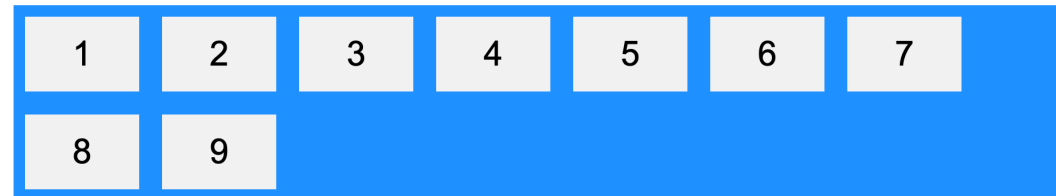
○ flex-wrap

- nowrap(default)
- wrap
- wrap-reverse

nowrap



wrap



wrap-reverse



```
.flex-container {  
  display: flex;  
  flex-wrap: wrap;  
}
```

➤ Flex Container

○ flex-flow

- 첫번째 속성(flex-direction) : row(default), column, row-reverse, column-reverse
- 두번째 속성(flex-wrap) : nowrap(default), wrap, wrap-reverse

```
.flex-container {  
  display: flex;  
  flex-flow: row wrap;  
}
```

➤ Flex Container

○ justify-content

- center
- flex-start(default)
- flex-end
- space-around
- space-between
- space-evenly

center

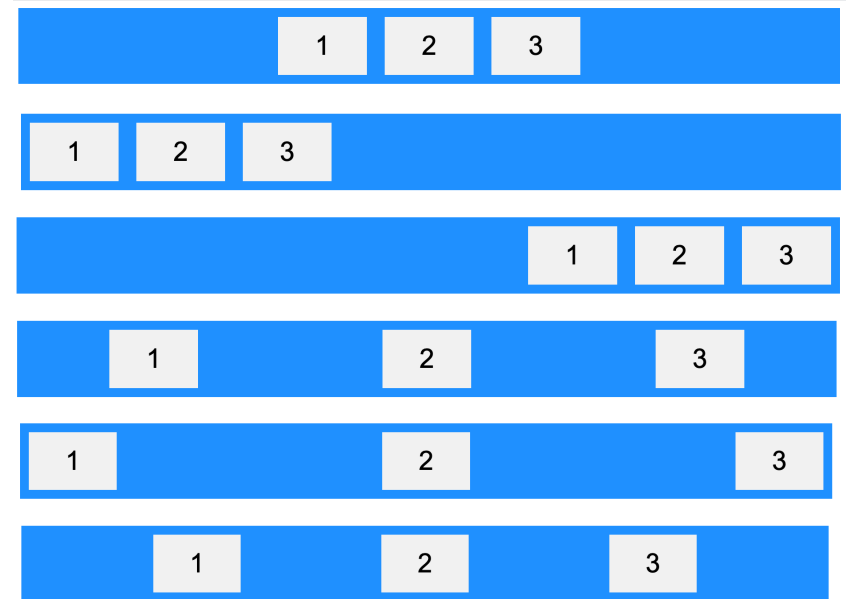
flex-start

flex-end

space-around

space-between

space-evenly



```
.flex-container {  
  display: flex;  
  justify-content: space-evenly;  
}
```

※ flex-direction이 column으로 지정된 경우에도 y축 방향으로 동일하게 적용됨

➤ Flex Container

○ align-items

- center
- flex-start
- flex-end
- stretch
- baseline
- normal(default)

center



flex-start



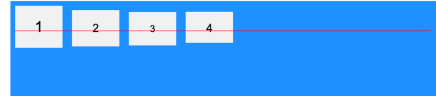
flex-end



stretch



baseline



※ flex-direction이 column으로 지정된 경우에도 x축(교차축) 방향으로 동일하게 적용됨

➤ Flex Container

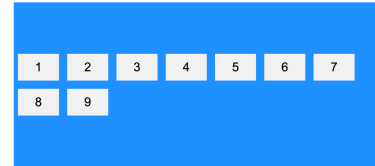
○ align-content

- center
- stretch(default)
- flex-start
- flex-end
- space-around
- space-between
- space-evenly

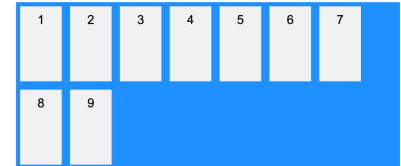
```
.flex-container {  
  display: flex;  
  height: 400px;  
  flex-wrap: wrap;  
  align-content: center;  
}
```

※ flex-direction이 column으로 지정된 경우에도 x축(교차축) 방향으로 동일하게 적용됨

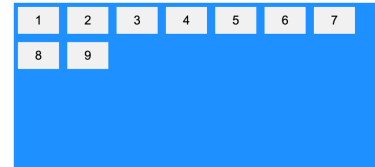
center



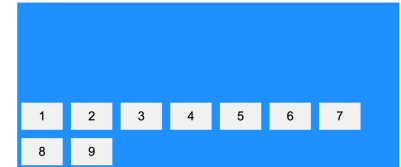
stretch



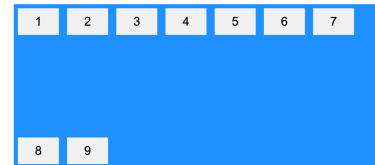
flex-start



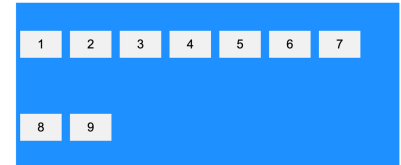
flex-end



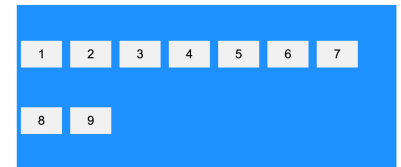
space-between



space-around



space-evenly



➤ Flex Container

- 중앙정렬



```
.flex-container {  
  display: flex;  
  height: 400px;  
  justify-content: center;  
  align-items: center;  
}
```

코드 예제

➤ Flex Items

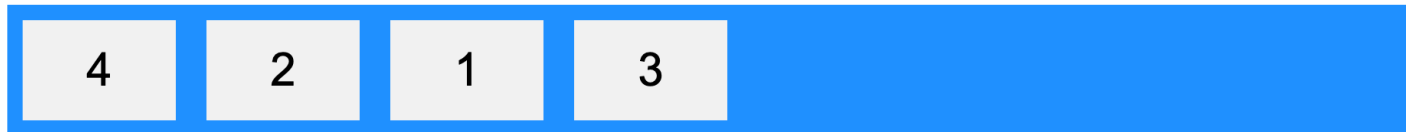
○ 속성(property)

속성명	설 명	비 고
order	flex item의 표시 순서를 지정	
flex-grow	flex item이 다른 item에 비해 얼마나 커질지 지정	
flex-shrink	flex item이 다른 item에 비해 얼마나 줄어들지 지정	
flex-basis	flex item의 초기 길이를 지정	
flex	flex-grow, flex-shrink, flex-basis 를 한번에 표시하는 축약 표현	
align-self	flex item에 대한 정렬을 지정	

➤ Flex Items

- order

```
<div class="flex-container">  
  <div style="order: 3">1</div>  
  <div style="order: 2">2</div>  
  <div style="order: 4">3</div>  
  <div style="order: 1">4</div>  
</div>
```



➤ Flex Items

○ flex-grow

- 다른 아이템에 비해 얼마나 클지 비율을 정하는 것으로 기본값은 0
- 해당 속성이 0이 아닌 값으로 지정되면 원래의 지정된 폭과는 상관없이 빈공간을 모두 활용하여 확장됨

```
<div class="flex-container">  
  <div style="flex-grow: 1">1</div>  
  <div style="flex-grow: 1">2</div>  
  <div style="flex-grow: 4">3</div>  
</div>
```

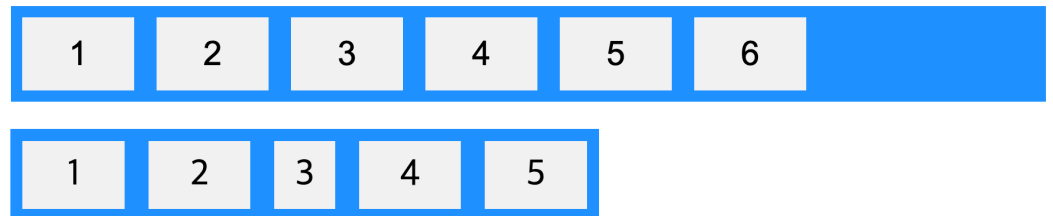


➤ Flex Items

○ flex-shrink

- 아이템들의 총 너비가 부모 컨테이너의 너비보다 커서 공간이 부족할 때에만 동작
- 다른 아이템들과 함께 설정된 비율대로 줄어들며 기본값은 1 (값이 클 수록 더 많이 줄어 듦)
 - 2로 지정된 경우 1인 아이템보다 2배 더 많이 줄어든다는 의미이나, 정확히는 flex-basis를 함께 고려해야 함
 - 기본크기(flex-basis → width(+padding) → 콘텐츠의 크기 순으로 고려)의 비율대로 부족한 공간을 나누며, 여기에 flex-shrink로 지정된 비율이 적용됨
- 0으로 지정된 경우 줄어들지 않음

```
<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div style="flex-shrink: 2">3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
</div>
```

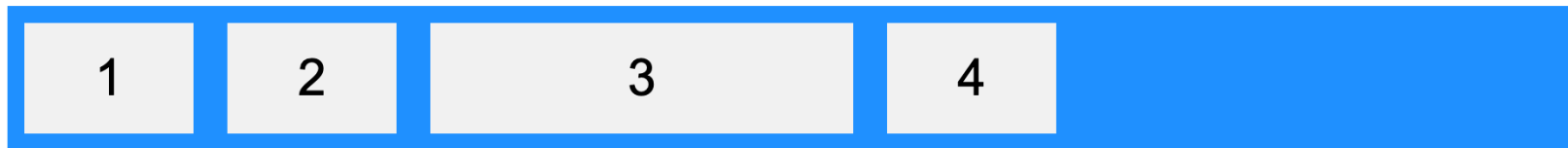


➤ Flex Items

○ flex-basis

- flex item의 초기 크기를 지정
- flex-direction이 row일 때는 width처럼 동작하고, column일 때는 height처럼 동작
- width보다 flex-basis의 우선순위가 높음(height도 동일)
- 기본값은 auto이며 이는 width나 height를 기본 크기로 사용하고, 이것도 없으면 콘텐츠 고유의 크기로 사용하라는 의미

```
<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div style="flex-basis: 250px">3</div>  
  <div>4</div>  
</div>
```

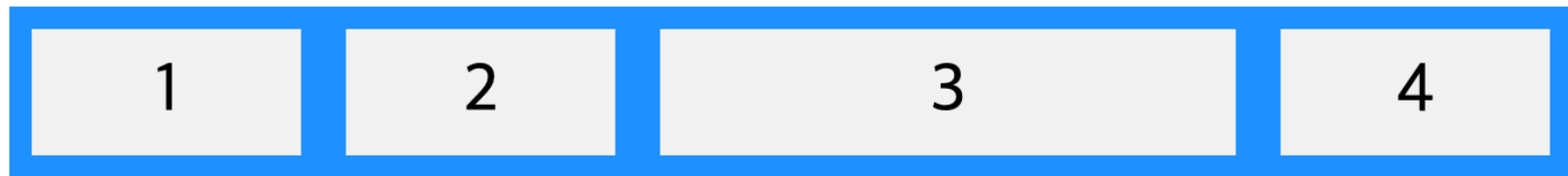


➤ Flex Items

○ flex

- flex-grow, flex-shrink, flex-basis 속성을 한번에 작성하는 축약 표현

```
<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div style="flex: 1 0 150px">3</div>  
  <div>4</div>  
</div>
```

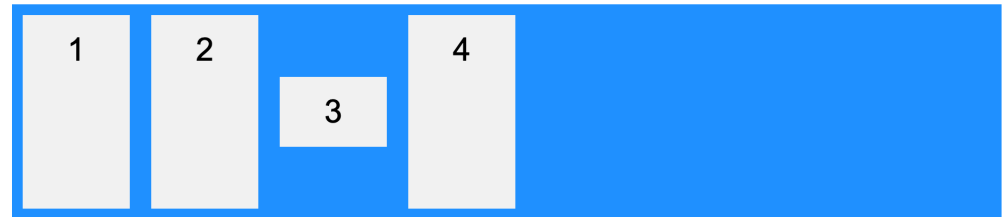


➤ Flex Items

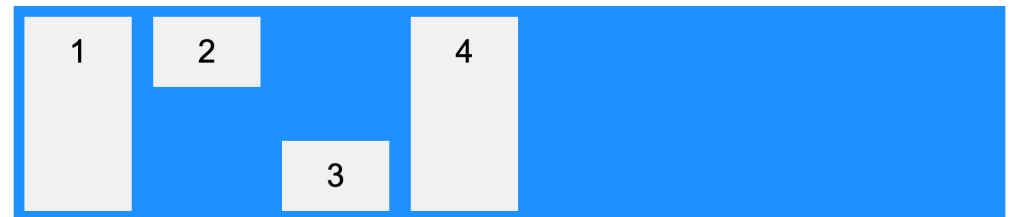
○ align-self

- flex item의 정렬방식을 지정
- 부모 컨테이너에서 지정한 align-items의 룰을 따르지 않고 각 개별 item에 지정하고자 할 때 사용

```
<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div style="align-self: center">3</div>  
  <div>4</div>  
</div>
```



```
<div class="flex-container">  
  <div>1</div>  
  <div style="align-self: flex-start">2</div>  
  <div style="align-self: flex-end">3</div>  
  <div>4</div>  
</div>
```



➤ Grid Layout Module

- 행과 열로 구성된 그리드 기반 레이아웃 시스템을 제공
- 반응형 레이아웃 구조를 쉽게 디자인 할 수 있음
- 구성요소
 - Grid Container : `display` 속성이 `grid` 혹은 `inline-grid`로 설정된 부모(컨테이너 요소)
 - Grid Items : 그리드 컨테이너의 바로 자식 요소는 자동으로 `grid item`이 됨

```
.container {  
  display: grid;  
  grid-template-columns: auto auto auto;  
  background-color: dodgerblue;  
  padding: 10px;  
}  
  
.container div {  
  background-color: #f1f1f1;  
  border: 1px solid black;  
  padding: 10px;  
  font-size: 30px;  
  text-align: center;  
}
```

1	2	3
4	5	

➤ Grid Container

○ 속성(property)

속성명	설 명	비 고
<code>align-content</code>	(전체 그리드 크기가 컨테이너보다 작을 때) 컨테이너 내부의 전체 그리드를 수직 정렬	
<code>align-items</code>	flexbox 또는 grid 컨테이너 내부 아이템의 기본 정렬을 지정	
<code>display</code>	요소의 표시 동작(렌더링 상자 유형)을 지정	
<code>column-gap</code>	열(column) 사이의 간격을 지정	
<code>gap</code>	<code>row-gap</code> 과 <code>column-gap</code> 속성의 단축 속성입니다.	
<code>grid</code>	<code>grid-template-rows</code> , <code>grid-template-columns</code> , <code>grid-template-areas</code> , <code>grid-auto-rows</code> , <code>grid-auto-columns</code> , <code>grid-auto-flow</code> 속성의 단축 속성	
<code>grid-auto-columns</code>	기본 열(column) 크기를 지정	
<code>grid-auto-flow</code>	자동 배치된 아이템이 그리드에 삽입되는 방식을 지정	
<code>grid-auto-rows</code>	기본 행(row) 크기를 지정	
<code>grid-template</code>	<code>grid-template-rows</code> , <code>grid-template-columns</code> , <code>grid-areas</code> 속성의 단축 속성	

➤ Grid Container

○ 속성(property)

속성명	설 명	비 고
grid-template-areas	이름이 지정된 그리드 아이템을 사용하여 열과 행을 표시하는 방법을 지정	
grid-template-columns	그리드 레이아웃의 열 크기와 열 개수를 지정	
grid-template-rows	그리드 레이아웃의 행 크기를 지정	
justify-content	(전체 그리드 크기가 컨테이너보다 작을 때) 컨테이너 내부의 전체 그리드를 수평 정렬	
place-content	align-content와 justify-content 속성의 단축 속성	
row-gap	행(row) 사이의 간격을 지정	

➤ Grid Tracks

- 그리드 컨테이너 내부에서 그리드 열과 행의 개수와 크기를 정의
 - `grid-template-columns` : 열의 개수와 너비를 정의
 - `grid-template-rows` : 행의 개수와 높이를 정의
 - `grid-template-areas` : 이름이 부여된 그리드 아이템을 사용하여 행과 열을 표시하는 방법을 정의
- 사용 단위
 - 고정 길이 : px, em, rem 등
 - 비율 : %
 - fr unit : Fraction의 약자로 고정크기를 제외한 사용 가능한 공간을 지정된 비율로 나눔
 - auto : 기본값으로 콘텐츠의 크기에 따라 자동으로 조정
 - min-content : 콘텐츠가 잘리거나 넘치지 않는 선에서 최소한으로 필요한 크기(보통 가장 긴 단어)를 차지
 - max-content : 콘텐츠가 줄바꿈 없이 최대한으로 차지하려는 크기(텍스트가 한 줄로 쭉 퍼졌을 때)를 차지
 - minmax(min, max) : 최소 크기와 최대 크기를 지정하여, 그 범위 안에서 유연하게 크기가 조절
 - fit-content(value) : max-content 처럼 콘텐츠에 맞춰 크기를 정하지만, value로 지정한 값보다는 커지지 않음
 - repeat() : repeat(3, 1fr), repeat(auto-fit, minmax(0, 1fr))

※ `grid-template-rows`에서도 동일하게 동작하기 위해서는 그리드 전체의 높이가 정확하게 지정되어 있어야 함.

➤ Grid Tracks

```
.container {  
  display: grid;  
  grid-template-columns: auto auto auto;  
}
```

1	2	3
4	5	6

```
.container {  
  display: grid;  
  grid-template-columns: 1fr 2fr 1fr;  
}
```

1	2	3
4	5	6

```
.container {  
  display: grid;  
  grid-template-columns: 80px 200px auto;  
}
```

1	2	3
4	5	6

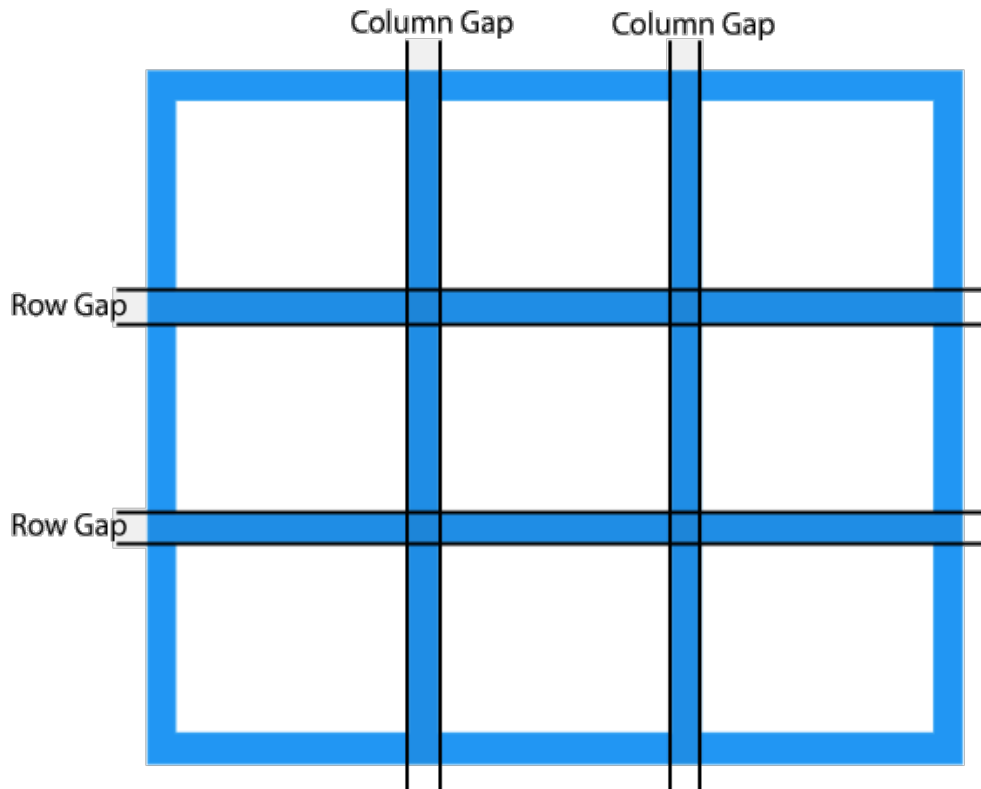
```
.container {  
  display: grid;  
  grid-template-columns: minmax(80px, 1fr) 150px 150px;  
}
```

1	2	3
4	5	6

```
.container {  
  display: grid;  
  grid-template-rows: 80px 200px;  
}
```

1	2	3
4	5	6
7	8	

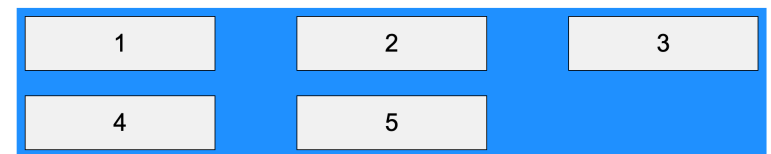
➤ Grid Gaps



```
.container {  
  display: grid;  
  column-gap: 50px;  
}
```

```
.container {  
  display: grid;  
  row-gap: 50px;  
}
```

```
.container {  
  display: grid;  
  gap: 30px 100px;  
}
```



➤ Grid Align

- justify-content
- align-content
- place-content

※ https://www.w3schools.com/css/css_grid_align.asp

➤ Grid Items

○ Item의 크기 조정

속성명	설 명	비 고
grid-column-start	시작될 열 구분선(column-line)을 선택	
grid-column-end	종료될 열 구분선(column-line)을 선택	
grid-column	grid-column-start, grid-column-end의 축약 표현	
grid-row-start	시작될 행 구분선(row-line)을 선택	
grid-row-end	종료될 행 구분선(row-line)을 선택	
grid-row	grid-row-start, grid-row-end의 축약 표현	

➤ Grid Items

```
.item1 {  
  grid-column-start: 1;  
  grid-column-end: 3;  
}
```

```
.item1 {  
  grid-column: 1 / span 2;  
}
```

1		2
3	4	5

```
.item1 {  
  grid-row-start: 1;  
  grid-row-end: 3;  
}
```

```
.item1 {  
  grid-row: 1 / span 2;  
}
```

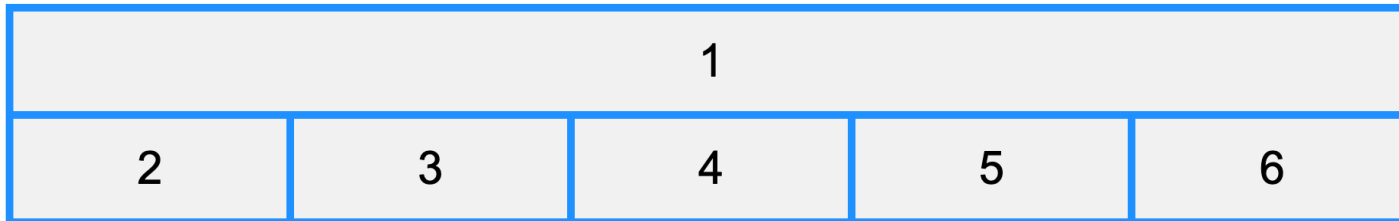
1	2	3
	4	5

```
.item1 {  
  grid-column: 1 / span 2;  
  grid-row: 1 / span 2;  
}
```

1		2
		3
4	5	6

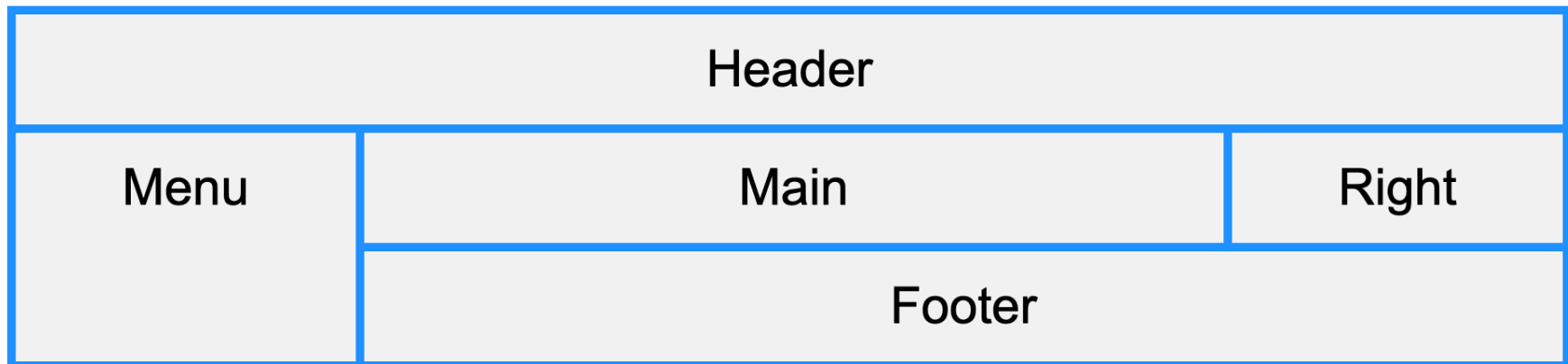
➤ Naming Grid Items

```
.container {  
  display: grid;  
  grid-template-areas: 'myHeader myHeader myHeader myHeader myHeader';  
}  
  
.item1 {  
  grid-area: myHeader;  
}
```



➤ Naming Grid Items

```
.item1 { grid-area: header; }  
.item2 { grid-area: menu; }  
.item3 { grid-area: main; }  
.item4 { grid-area: right; }  
.item5 { grid-area: footer; }  
  
.container {  
  display: grid;  
  grid-template-areas:  
    'header header header header header header'  
    'menu main main main main right'  
    'menu footer footer footer footer footer';  
}
```



➤ CSS Media Query

- 웹 페이지를 표시하는 장치나 환경의 특성에 따라 스타일을 적용
 - 쿼리 내부의 CSS 코드는 해당 조건이 만족하는 경우에만 동작

```
/* Hide element if the viewport width is 600px or less */  
@media screen and (max-width: 600px) {  
  #div1 {  
    display: none;  
  }  
}
```



데스크톱



태블릿

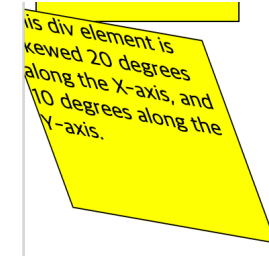


핸드폰

➤ 2D Transforms

○ 요소에 대한 회전, 확대/축소, 이동 등의 변형 가능

- translate
- rotate
- scale
- skew
- matrix



```
div {  
  transform: translate(50px, 100px);  
}
```

```
div {  
  transform: rotate(20deg);  
}
```

```
div {  
  transform: scale(2, 3);  
}
```

```
div {  
  transform: skew(20deg);  
}
```

➤ Transition

- 지정된 시간 동안 속성 값을 부드럽게 변경

```
div {  
  width: 100px;  
  height: 100px;  
  background-color: red;  
  transition: width 2s;  
}
```

```
div {  
  transition: width 2s, height 4s, background-color 3s;  
}
```

The transition Property

Hover over the div element below, to see the transition effect:



The transition Property

Hover over the div element below, to see the transition effect:



➤ Animation

- JavaScript를 사용하지 않고도 HTML 요소의 애니메이션 가능
- 관련 속성(property)

속성명	설 명	비 고
@keyframes	애니메이션 코드 정의	
animation	애니메이션 속성 축약 표현	name, duration, timing-func, delay, iteration, direction
animation-delay	애니메이션 시작의 지연 시간 설정	
animation-direction	애니메이션 재생 방향(forward, backward, alternate) 설정	normal, reverse, alternate, alternate-reverse
animation-duration	애니메이션 한 사이클 진행 시간을 지정	
animation-fill-mode	애니메이션이 진행되지 않을 때의 요소 스타일 지정(시작 전, 종료 후, 둘 다)	none, forward, backward, both
animation-iteration-count	애니메이션 반복 횟수 지정	
animation-name	애니메이션을 정의한 @keyframes 이름을 지정	
animation-play-state	애니메이션의 진행 혹은 정지 상태를 지정	
animation-timing-function	애니메이션 속도 커브 지정	ease, linear, ease-in, ease-out, ease-in-out, cubic-bezier

➤ Animation

○ @keyframes

```
/* The animation code */
@keyframes myAnimation {
  from {background-color: red;}
  to {background-color: yellow;}
}

/* The element to apply the animation to */
div {
  width: 100px;
  height: 100px;
  background-color: red;
  animation-name: myAnimation;
  animation-duration: 4s;
}
```

```
div {
  width: 100px;
  height: 100px;
  background: red;
  position: relative;
  animation-name: myAnimation;
  animation-duration: 3s;
  animation-delay: 2s;
  animation-fill-mode: backwards;
}

div {
  animation: myAnimation 5s linear 2s infinite alternate;
}
```

➤ header



학회소개

회원안내

학회지

최고위 과정 

NEWS

➤ card design

2025 방공-미사일방어 포럼

세션 II

순항미사일 위협 대응을 위한
요격체계 발전 방향



김 대 준

소장

LIG넥스원

대공체계 연구소

2025.09.25. 대한민국공군, 한국방위산
업학회 공동주최 2025 방공-미사일방어

관리자

2025-10-09

25

2025 방공-미사일방어 포럼

세션 II

저 RCS · 저고도 위협 표적의
탐지 / 추적 기술



홍 윤 식

소장

한화시스템

레이다 연구소

2025.09.25. 대한민국공군, 한국방위산
업학회 공동주최 2025 방공-미사일방어

관리자

2025-10-09

43

2025 방공-미사일방어 포럼

세션 II

라-우전 무인기 · 복합 무기체계
위협사례와 교훈



Herald Mannheim

대표

AIRBUS

Defense and Space

2025.09.25. 대한민국공군, 한국방위산
업학회 공동주최 2025 방공-미사일방어

관리자

2025-10-09

24

2025 방공-미사일방어 포럼

세션 I

공중위협 변화에 따른
방공전력 발전방안



정 성 순

과장

육군 방공학교

전력발전과

2025.09.25. 대한민국공군, 한국방위산
업학회 공동주최 2025 방공-미사일방어

관리자

2025-10-09

24

2025 방공-미사일방어 포럼

세션 I

방공의 시선을 낮추다.



주 성 근

과장

공군 미사일방어사령부

전력계획과

2025.09.25. 대한민국공군, 한국방위산
업학회 공동주최 2025 방공-미사일방어

관리자

2025-10-09

28

2025 방공-미사일방어 포럼

기 조 강 연



윤 용 현

교수

국민대학교

안보융합기술연구소장

2025.09.25. 대한민국공군, 한국방위산
업학회 공동주최 2025 방공-미사일방어

관리자

2025-10-09

16

2025 방공-미사일방어 포럼

축 사



권 진 회

총장

경상대학교

2025.09.25. 대한민국공군, 한국방위산
업학회 공동주최 2025 방공-미사일방어

관리자

2025-10-09

25

2025 방공-미사일방어 포럼

환 영 사



손 석 락

공군 참모총장

2025.09.25. 대한민국공군, 한국방위산
업학회 공동주최 2025 방공-미사일방어

관리자

2025-10-09

16

➤ 이미지 슬라이드

이미지 슬라이드 UI



강사 연락처

➤ 디노웍스 대표 이시영

➤ 문의

- E-mail : lsy@dinoworks.kr
 - Mobile : 010-5179-6455
- 