

14. CSS grid

코드는 짧고 사연은 길다.



CSS grid

격자를 이용하여 내용의 크기와 위치를 제어하는 방법.

flex는 단일 축을 중심으로 배치, grid는 두 개의 축을 이용.
특히 셀 병합 기능을 제공. 다른 수단에 비해 짧은 코드로 자유도 높은 배치를 구현.

CSS grid

- **grid** ← 트랙의 수와 크기, 컨테이너에 적용.
 - grid-template
 - grid-template-rows(행 트랙의 수량과 크기) ☆
 - grid-template-columns(열 트랙의 수량과 크기) ☆
 - grid-template-areas(셀 이름 명시) ?
 - grid-auto-flow(흐름 방향과 밀집) ☆
 - grid-auto-rows(암시적 행 트랙의 크기) ☆
 - grid-auto-columns(암시적 열 트랙의 크기) ☆
- **grid-area** ← 아이탬의 배치와 병합, 아이탬에 적용.
 - grid-row
 - grid-row-start ☆(행 시작)
 - grid-row-end ☆(행 끝)
 - grid-column
 - grid-column-start ☆(열 시작)
 - grid-column-end ☆(열 끝)

“

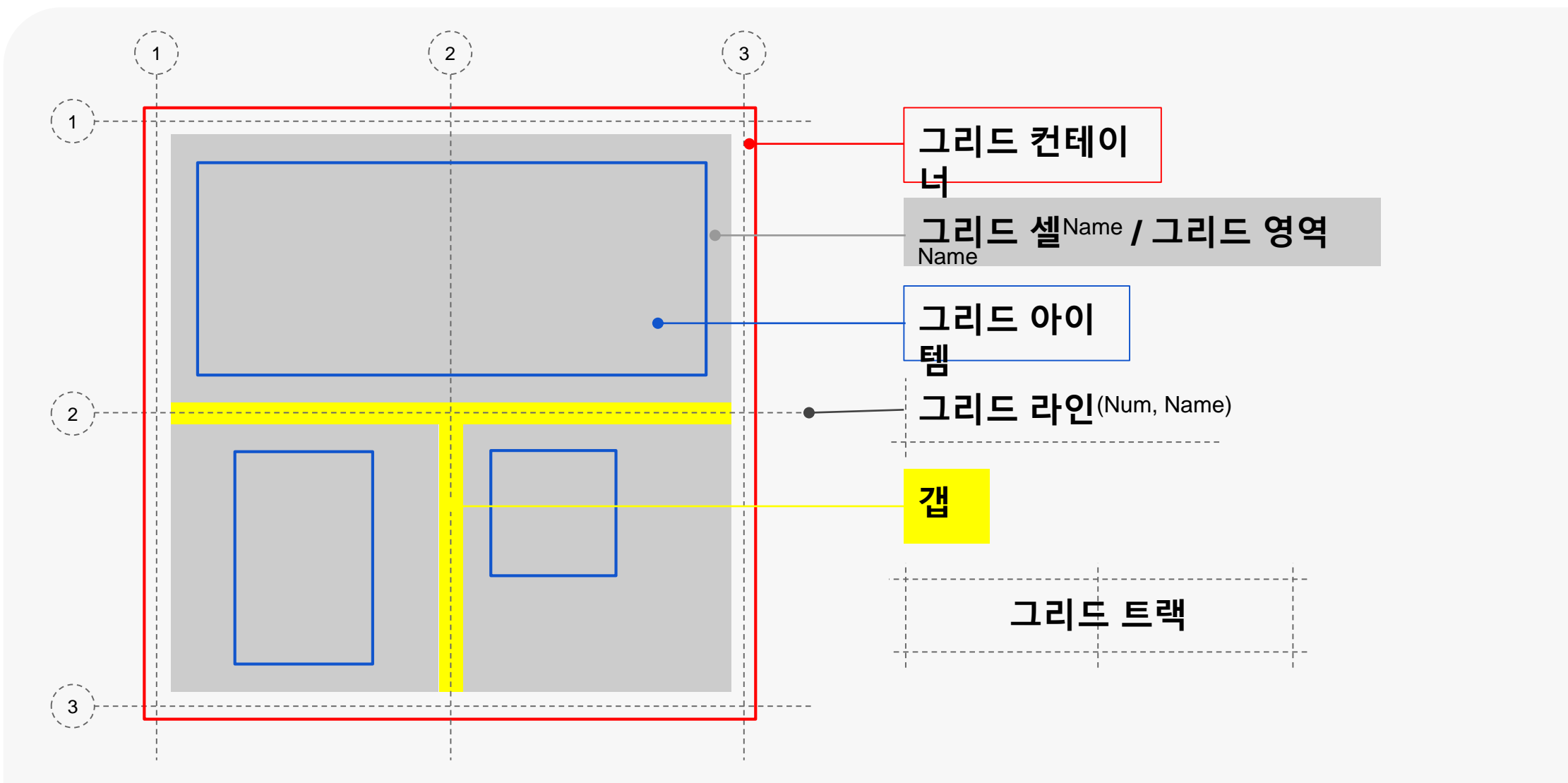
Grid 용어



Grid 용어

- grid container(그리드 컨테이너)
- grid item(그리드 아이템)
- grid line(그리드 라인)
- grid track(그리드 트랙)
- grid cell(그리드 셀), grid area(그리드 영역)
- gap(갭)

Grid 용어



Grid 용어

명시적 그리드

트랙의 크기와 수량을 분명하게 선언한 그리드 `grid-template-`
`rows/columns/areas`

속성으로 제어.

암시적 그리드

명시적 그리드 외부에 배치되어 `grid-auto-`
`flow/rows/columns` 속성으로
흐름 방향과 크기를 결정하는 그리드.

gap: ☆

다중 컬럼, 플렉스, 그리드 아이템 사이의 간격.

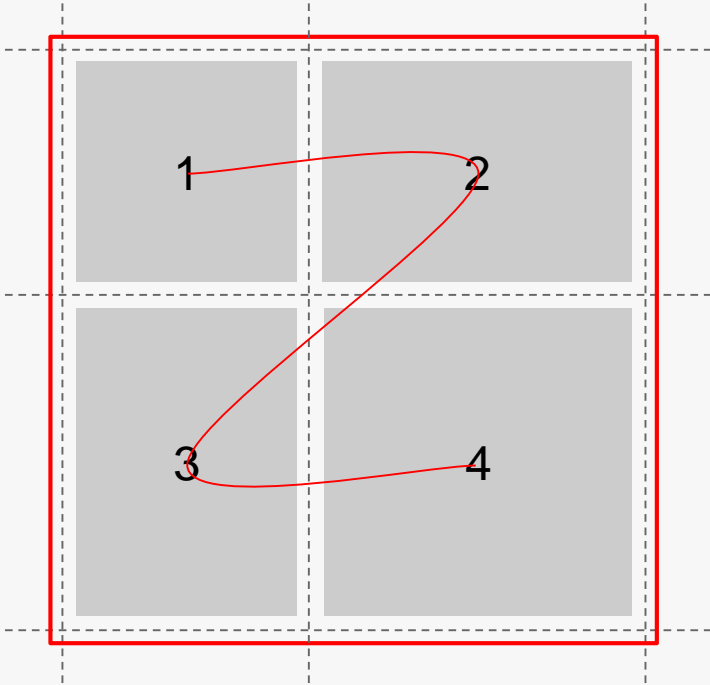
- 값: <'row-gap'> <'column-gap'>?
- 초기 값: normal // == 다중 컬럼에서 1em 그렇지 않으면 0
- 적용: 컬럼/플렉스/그리드 컨테이너.

“

Grid container / item 역할

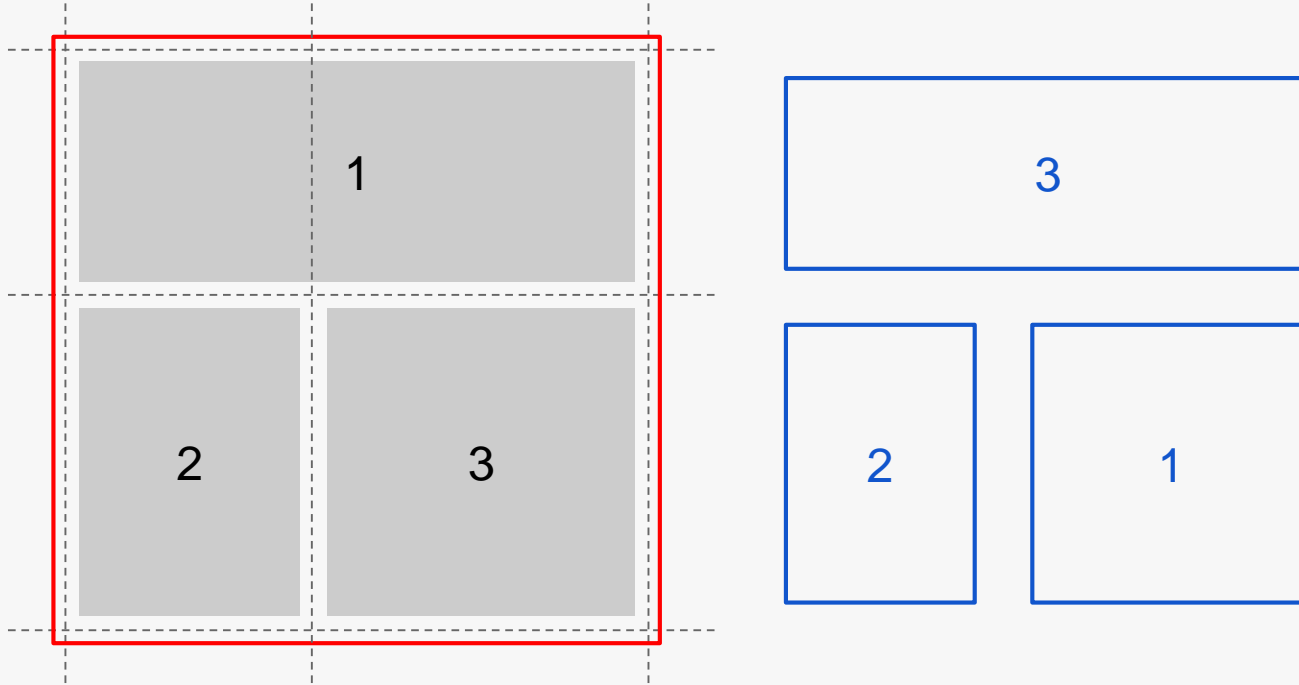


grid container 역할



- 트랙의 수량과 크기를 명시. (grid-template-*)
- 아이템 배치 방향. (grid-auto-flow)
- 암시적 트랙 크기. (grid-auto-*)

grid item 역할



→ 아이템의 배치와 병합. (grid-row-start/-column-start/-row-end/-column-end, grid-area)

“

Grid container 생성

display: grid | inline-grid



Grid container 생성

Test template: <https://t.ly/hefq>

```
.container {  
  display: grid | inline-grid;  
}
```

그리드 컨테이너/아이템에 선언하지 않은 모든 grid-* 속성은 초기값으로 설정된다. grid 속성으로 컨테이너의 트랙(행/열) 수와 크기를 설정하고 grid-area 속성으로 아이템의 배치와 병합을 설정하기 전까지는 그리드 컨테이너를 생성한 것 만으로 특별한 효용이 없다.

“

Grid track 생성

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



Grid track 생성(균등)

Test template: <https://t.ly/hefq>

```
.container {  
  display: grid;  
  grid: ' . . .';  
}
```

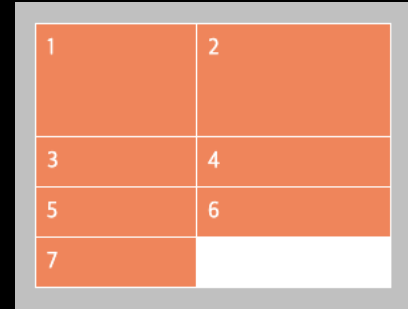
1	2	3
4	5	6
7		

그리드 컨테이너와 3열의 익명 트랙 단축 문법. 마침표(.)와 공백() 구분자로 익명 셀을 생성. 셀 크기는 내용에 따라 자동. 트랙의 크기를 제어하지 않기 때문에 실무에서 유용하지 않은 패턴.

Grid track 생성(제어) ★ 실무 사용 확률 99%

Test template: <https://t.ly/hefq>

```
.container {  
  display: grid;  
  grid: 80px 1fr / 120px 1fr; ★  
  grid: <'grid-template-rows'> / <'grid-template-columns'>  
}
```



1	2
3	4
5	6
7	

2열 2행 트랙 단축 문법. 트랙의 크기와 수량을 명시적으로 제어.
실무에서 가장 빈번하게 사용하는 패턴.
명시적으로 선언하지 않는 나머지 트랙은 자동.

Grid track 생성(반복)

Test template: <https://t.ly/hefq>

```
.container {  
  display: grid;  
  grid: auto / 40px 1fr 2fr 1fr 2fr;  
  grid: auto / 40px repeat(2, 1fr 2fr);  
  repeat( <integer [1,∞]> , <track-list> )  
}
```

1	2	3	4	5
6	7			

auto 값으로 트랙의 크기 임의 지정 가능.

repeat() 함수로 크기 값을 반복할 수 있다.

함수의 첫 번째 인자는 트랙의 수량, 두 번째 인자는 트랙의 크기.

Grid track 생성(방향)

Test template: <https://t.ly/hefq>

```
.container {  
  display: grid;  
  grid: 1fr 2fr / auto-flow; ↓  
  <'grid-template-rows'> / [ auto-flow && dense? ] <'grid-auto-columns'>?  
  grid: auto-flow / 1fr 2fr; →  
  [ auto-flow && dense? ] <'grid-auto-rows'>? / <'grid-template-columns'>  
}
```

1	3	5	7
2	4	6	

배치 방향 설정.

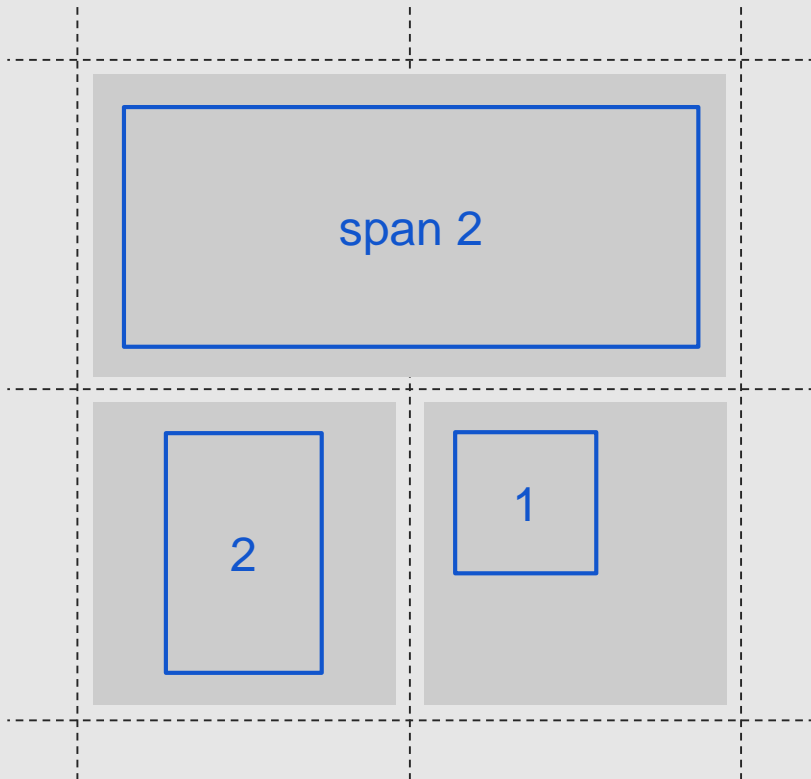
auto-flow 값은 grid 단축 속성에서만 사용하는 값의 형태로 grid-auto-flow 속성 값의 다른 표기법.

슬래시(/)와 함께 교차 축 grid-template-rows/columns 값의 명시가 필수.

“

Grid item 배치/병합

grid-area: <grid-line> [/ <grid-line>]{0,3}



Grid item 배치

Test template: <https://t.ly/hefq>

```
.container {  
  display: grid;  
  grid: auto / repeat(3, 1fr);  
}  
  
.item1 { grid-area: 2 / 3; }  
<grid-line> [ / <grid-line> ]{0,3}  
<'grid-row-start'> / <'grid-column-start'> / <'grid-row-end'> / <'grid-column-end'>
```

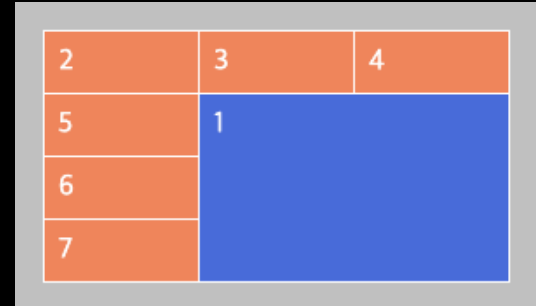
2	3	4
5	6	1
7		

행 배치 시점 / 열 배치 시점 / 행 배치 종점 / 열 배치 종점 값을 선언하여
아이템의
배치 위치를 결정할 수 있다.
값은 시계 반대 방향으로 순환하고 슬래시(/) 구분자로 분리한다.
생략한 값은 auto와 같다.

Grid item 배치/병합 ✨

Test template: <https://t.ly/hefq>

```
.container {  
  display: grid;  
  grid: auto / repeat(3, 1fr);  
}  
  
.item1 { grid-area: 2 / 2 / span 3 / span 2; }  
  
<integer> | span && <integer>
```



span 키워드와 병합할 트랙의 수량을 조합하면 셀을 병합할 수 있다.

“

Grid track/item 정렬

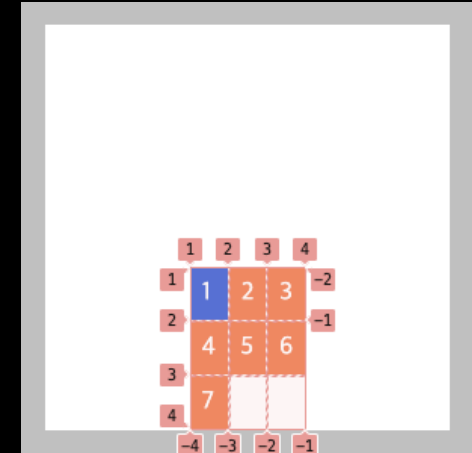
place-content: <'align-content'> <'justify-content'>?
place-items: <'align-items'> <'justify-items'>?
place-self: <'align-self'> <'justify-self'>?



Grid item 정렬(트랙)

Test template: <https://t.ly/hefq>

```
.container {  
  display: grid;  
  grid: auto / repeat(3, auto);  
  place-content: end center;  
  <content-position> = center | start | end  
}
```

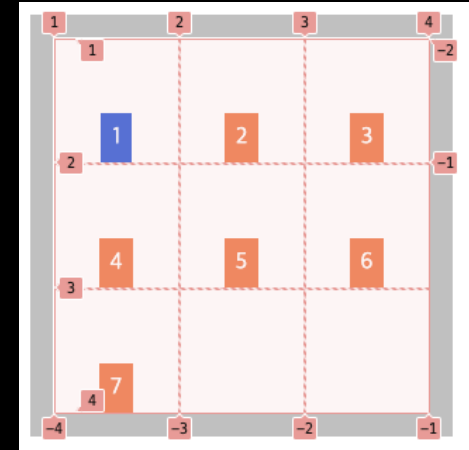


트랙의 크기가 auto인 상태로 컨테이너를 가득 채우지 않는다면
트랙의 위치를 정렬할 수 있다.

Grid item 정렬(아이템 복수)

Test template: <https://t.ly/hefq>

```
.container {  
  display: grid;  
  grid: auto / repeat(3, 1fr);  
  place-items: end center;  
  <self-position> = center | start | end  
}
```



셀(복수)의 위치를 정렬할 수 있다.

Grid item 정렬(아이템 단수)

Test template: <https://t.ly/hefq>

“

Grid dense

auto-flow dense



Grid dense(밀집)

Test template: <https://t.ly/hefq>

1 { grid-area: auto / span 2 }		
2 { grid-area: auto / span 2 }	3	
4	5	6
7		

☒ grid: auto-flow / repeat(3, 1fr);
☐ grid: auto-flow **dense** / repeat(3, 1fr);

1 { grid-area: auto / span 2 }	3	
2 { grid-area: auto / span 2 }	4	
5	6	7

☐ grid: auto-flow / repeat(3, 1fr);
☒ grid: auto-flow **dense** / repeat(3, 1fr);

채우지 못한 빈 영역이 있으면 흐름 방향을 거슬러 올라 트랙을 채운다.

Demo - <https://t.ly/ZSsq>

Grid dense(밀집)

Test template: <https://t.ly/hefq>

```
.container {  
  display: grid;  
  grid: auto-flow dense / repeat(3, 1fr);  
}  
.item1, .item2 {  
  grid-area: auto / span 2;  
}  
[ auto-flow && dense? ]
```

dense는 흐름 방향을 설정하는 auto-flow 값과 항상 함께 쓴다.
채우지 못한 빈 영역이 있으면 흐름 방향을 거슬러 올라 트랙을 채운다.

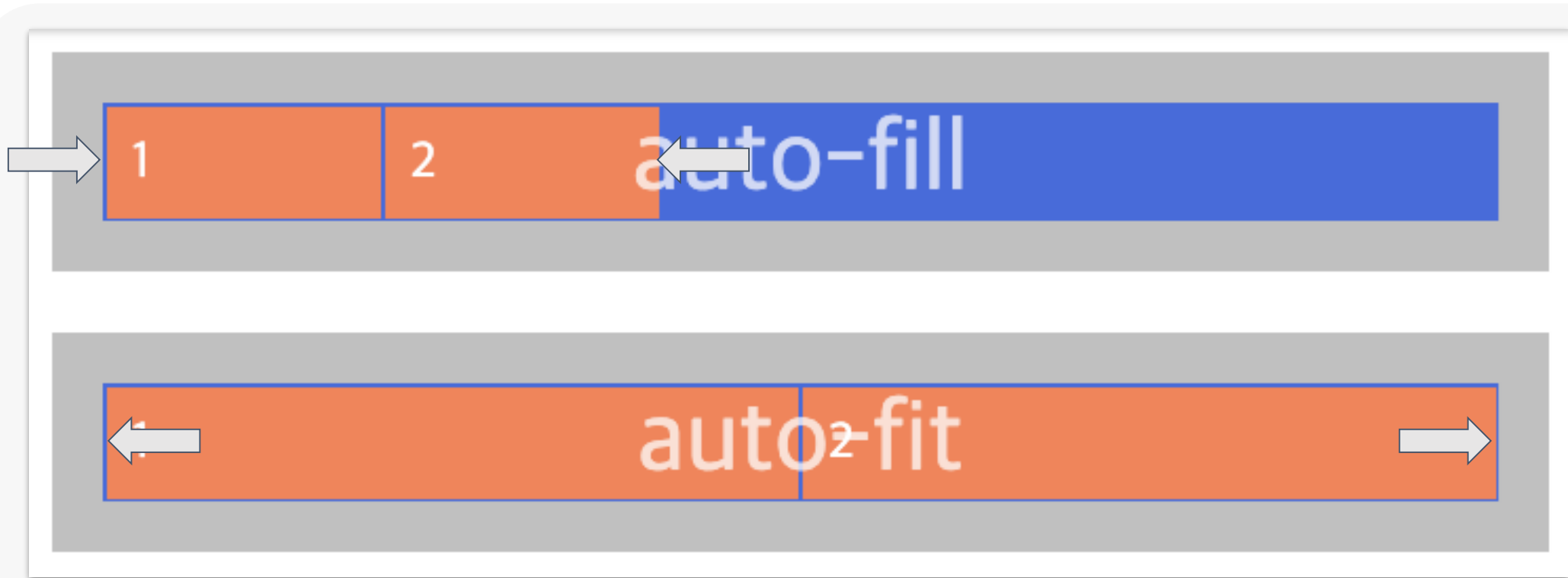
“

Grid auto-fill/fit

`repeat([auto-fill | auto-fit] , <fixed-size>)`



Grid auto-fill/fit (동적 트랙)



트랙을 채우지 못한 상황에서 트랙의 최대 크기가 auto이면 auto-fill 또는 auto-fit 방식으로 트랙의 크기와 수량을 자동으로 결정한다.

Grid auto-fill/fit (동적 트랙)

Test template: <https://t.ly/hefq>

```
.container {  
  display: grid;  
  grid: auto / repeat(auto-fill, minmax(80px, auto));  
  grid: auto / repeat(auto-fit, minmax(80px, auto));  
  repeat( [ auto-fill | auto-fit ] , <fixed-size> )  
  repeat(auto-fill, auto) ✗  
  repeat(auto-fill, 1fr) ✗  
  auto repeat(auto-fill, ...) ✗  
  1fr repeat(auto-fill, ...) ✗  
}
```

트랙을 채우지 못한 상황에서 트랙의 최대 크기가 auto이면 auto-fill 또는 auto-fit 방식으로 트랙의 크기와 수량을 자동으로 결정한다.

“

실습 과제



과제

CSS grid 레이아웃 문법과 예제.

<https://naradesign.github.io/css-grid-layout.html>

- CSS 속성 값(value:) 정의 구문 해설
- 그리드 컨테이너 생성하기
- `gap` ← 격자 사이의 간격, 컨테이너에 적용.
 - `row-gap`
 - `column-gap`
- `grid` ← 트랙의 수와 크기, 컨테이너에 적용.
 - `grid-template`
 - `grid-template-rows`
 - `grid-template-columns`
 - `grid-template-areas`
 - `grid-auto-flow`
 - `grid-auto-rows`
 - `grid-auto-columns`
- `grid-area` ← 아이템의 배치와 병합, 아이템에 적용.
 - `grid-row`
 - `grid-row-start`
 - `grid-row-end`
 - `grid-column`
 - `grid-column-start`
 - `grid-column-end`

Thank you !

