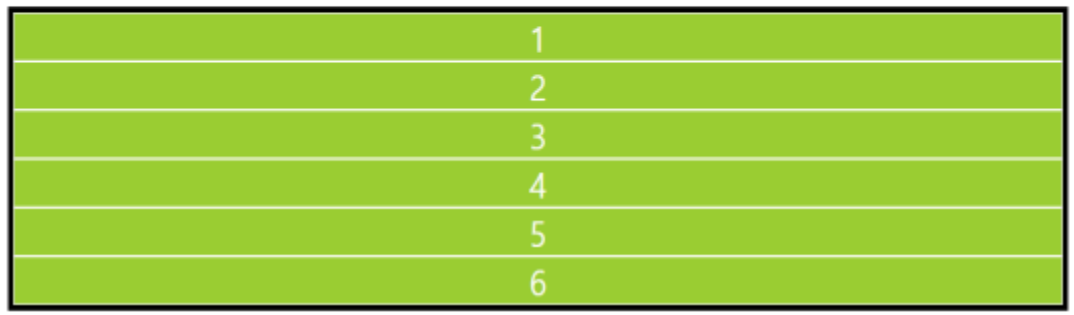
그리드 부모요소를 정의 곧, 해당 선택자에 그리드를 적용합니다.

1) 부모요소에 사용하는 Grid 속성 display: grid; display: inline-grid;



HTML

```
<div class="parent">
   <div>1</div>
   <div>2</div>
   <div>3</div>
   <div>4</div>
   <div>5</div>
   <div>6</div>
</div>
```

```
.parent {
    border: 5px solid #000;
    display: grid;
}
.parent div {
    color: #fff;
    text-align: center;
    font-size: 30px; padding: 0 20px;
    border: 1px solid #fff;
    background-color: yellowgreen;
}
```

플렉스의 경우 display: flex 만으로도 자식요소가 가로로 배치됩니다. 왜냐면 플렉스를 시작하는 것 만으로 flex-direction: row를 기본 값으로 갖게 됩니다. 하지만 그리드는 display: grid 만으로 어떤 변화도 줄 수 없습니다.

display: inline-grid는 자식요소의 내용에 맞게 줄어듭니다.
플렉스 부모요소 속성에서 display: inline-flex 와 동일합니다.

2-1) 부모요소에 사용하는 Grid 속성 grid-template-columns: auto 또는 단위

자식요소 컬럼(열, 세로)을 크기 또는 단위로 정의

1	2
3	4
5	6

HTML

```
<div class="parent">
    <div>1</div>
    <div>2</div>
    <div>3</div>
    <div>4</div>
    <div>5</div>
    <div>6</div>
</div>
```

```
.parent {
    border: 5px solid #000;
    display: grid;
    grid-template-columns: auto auto;
}
.parent div {
    color: #fff;
    text-align: center;
    font-size: 30px;
    border: 1px solid #fff;
    background-color: yellowgreen;
}
```

자식요소를 컬럼 곧, 세로로 2분할 한다 하는 의미입니다. 만약 auto 가 3개이면 3분할 4개이면 4분할을 의미합니다.
단위 대신에 auto를 사용했기 때문에 알아서 자동으로 1/2 또는 1/3로 나눕니다.

auto 대신 특정 단위를 넣으면 그 곳은 고정하고 나머지들은 부모요소의 너비에 꽉 차게 자동으로(auto) 배분합니다.

CSS 레이아웃에서 기억하실 것!
컬럼(column), 열, 세로 다 같은 말입니다.
로우(row), 행, 가로 다 같은 말입니다.

1	2	3
4	5	6

▲ grid-template-column: auto auto auto;

auto 200px auto

1	2	3
4	5	6

▲ grid-template-column: auto 200px auto;

fraction 일부, 파편이란 뜻으로 flex-grow와 같은 개념

2-2) 부모요소에 사용하는 Grid 속성 grid-template-columns: fr 단위

1	2
3	4
5	6

HTML

```
<div class="parent">
    <div>1</div>
    <div>2</div>
    <div>3</div>
    <div>4</div>
    <div>5</div>
    <div>6</div>
</div>
```

```
.parent {
    border: 5px solid #000;
    display: grid;
    grid-template-columns: 1fr 1fr;
}
.parent div {
    color: #fff;
    text-align: center;
    font-size: 30px;
    border: 1px solid #fff;
    background-color: yellowgreen;
}
```

fr의 개수가 auto의 개수처럼 컬럼의 수를 결정하므로, 왼쪽 예시처럼 2개의 컬럼을 만들려면 fr이 2개 있어야 합니다.

또는 아래 예시처럼 px % auto fr 단위를 섞어서 사용해도 됩니다.

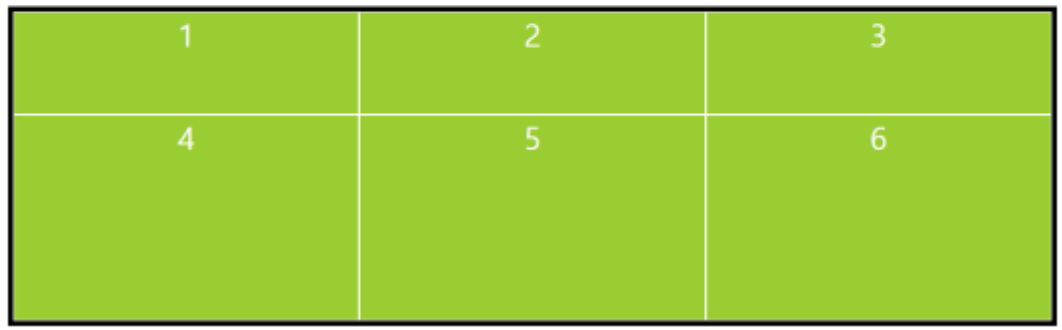
1	2	3
4	5	6

▲ grid-template-column: 1fr 1fr 1fr

▲ grid-template-columns: 100px 1fr 2fr;

자식요소 로우(열, 가로)을 크기 또는 단위로 정의

3-1) 부모요소에 사용하는 Grid 속성 grid-template-rows: auto 또는 단위



HTML

```
<div class="parent">
    <div>1</div>
    <div>2</div>
    <div>3</div>
    <div>4</div>
    <div>5</div>
    <div>6</div>
</div>
```

```
.parent {
    border: 5px solid #000; background-color: pink;
    display: grid;
    grid-template-columns: auto auto auto;
    grid-template-rows: 100px 200px;
}
```

- grid-template-rows의 첫번째 가로의 높이, 두번째 값은 두번째 가로의 높이

```
.parent div {
    color: #fff; text-align: center;
    font-size: 30px; border: 1px solid #fff;
    background-color: yellowgreen;
}
```

- grid-template-rows: 100px;
  첫번째 가로가 100px, 두번째 가로는 auto
  (auto로 적용하지 않으려면 가로의 개수만큼 단위를 적어줘야 함)

▲ grid-template-rows: 100px

그리드 템플릿을 가로 / 세로 축약형으로 표시

3-2) 부모요소에 사용하는 Grid 축약 속성 grid: [grid-template-rows] / [grid-template-columns]

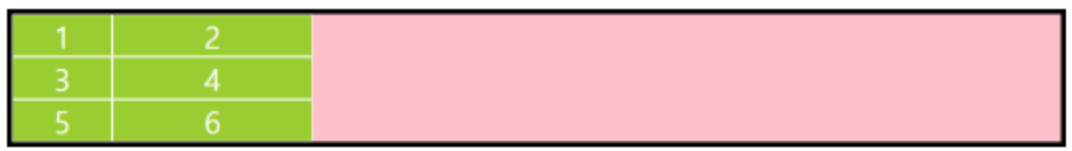
HTML

```
<div class="parent">
    <div>1</div>
    <div>2</div>
    <div>3</div>
    <div>4</div>
    <div>5</div>
    <div>6</div>
</div>
```

```
.parent {
    border: 5px solid #000; background-color: pink;
    display: grid;
    /* grid-template-columns: auto auto auto; */
    /* grid-template-rows: 100px 200px; */
    grid: 100px 200px / auto auto auto;
}
.parent div {
    color: #fff; text-align: center; font-size: 30px;
    border: 1px solid #fff; background-color: yellowgreen;
}
```

그리드 템플릿 로우와 그리드 템플릿 컬럼을 동시에 축약형으로 사용 가능.

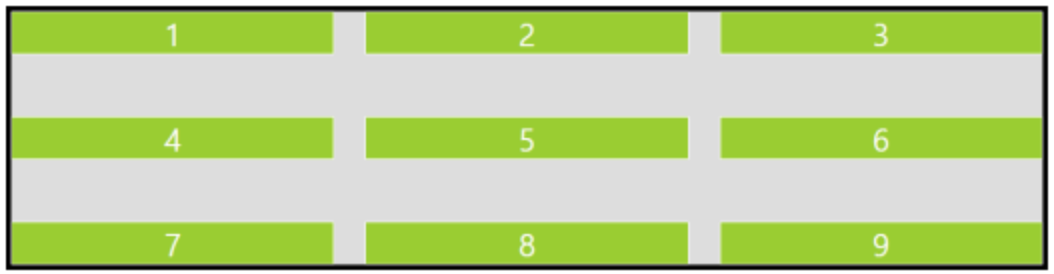
순서는 가로(row) / 세로(column)



▲ 가로와 세로 순서가 바뀌는 경우 grid: auto auto auto / 100px 200px;

4) 부모요소에 사용하는 Grid 속성 grid-column-gap, grid-row-gap, grid-gap

가로 또는 세로 사이의 간격을 정의



HTML

```
<div class="parent">
   <div>1</div>
   <div>2</div>
   <div>3</div>
   <div>4</div>
   <div>5</div>
   <div>6</div>
   <div>7</div>
   <div>8</div>
   <div>9</div>
</div>
```

```
.parent {
    border: 5px solid #000;
    background-color: #ddd;
    display: grid;
    grid-template-columns: auto auto auto;
    grid-column-gap: 30px;
    grid-row-gap: 60px;
    /* grid-gap: 30px 60px; */
    /* grid-gap: 30px */
}
.parent div {
    color: #fff;
    text-align: center; font-size: 30px;
    border: 1px solid #fff;
    background-color: yellowgreen;
```

- grid-column-gap
  좌우 여백을 조절
- grid-row-gap
  상하 여백을 조절

- grid-gap: 좌우 여백 상하 여백

- grid-gap: 좌우 상하 여백 동일하게

※ Grid 레이아웃을 코딩하기 전에 부모요소에 사용하는 Grid 정렬 속성에 대한 이해

그리드(Grid) 자식요소 정렬 방식은 플렉스(Flex) 자식요소 정렬하는 것도 비슷합니다. Grid 정렬은 수직 정렬과 수평 정렬 그리고 개별 정렬 3가지가 있습니다. 수평 정렬과 수직 정렬에 사용하는 CSS 속성 이름에서 유추하시면 크게 어렵지는 않습니다.

justify-content는 수평 정렬을 할 때 사용하고, align-content와 align-items는 수직 정렬입니다. 이건 수평 정렬 이건 수직 정렬 이렇게 외우시면 실수하실 수 있습니다. 그래서 아래의 개념을 생각하시면 혼란스럽지 않으실 거에요.

첫번째 팁(Tip)

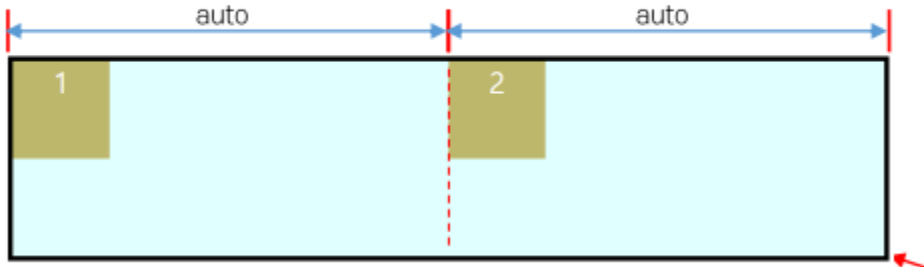
- CSS 속성 중에 텍스트를 수평으로 양쪽 정렬하는 속성이 text-align: justify 입니다. 곧 justify가 들어가면 수평 정렬이라고 생각하시면 됩니다.
- CSS 속성 중에 인라인 요소를 수직으로 중앙 정렬하는 속성이 vertical-align: middle 입니다. 곧 align이 들어가면 수직 정렬이라고 생각하시면 됩니다.

두번째 팁(Tip)

- Grid 속성에서 수평 정렬을 하는 justify-content 가 있습니다. content 가 들어가면 자식 요소들이 2줄 이상으로 줄이 바뀐 상태를 수평 정렬합니다.
- Grid 속성에서 수직 정렬을 하는 align-items가 있습니다. items 가 들어가면 자식 요소들이 줄이 바뀌지 않은 상태의 1줄일 경우를 수직 정렬합니다.

자식요소를 grid-template-column 안에서 각자의 컬럼 내에서 수평으로 정렬

5) 부모요소에 사용하는 Grid 속성 justify-items: start | center | end | stretch;



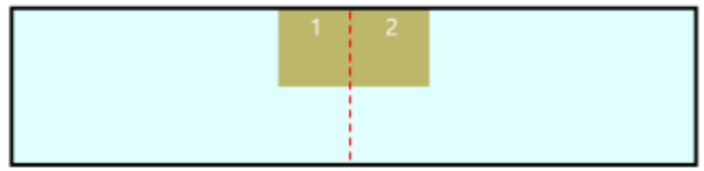
▲ justify-items: start;

▲ justify-items: center;

▲ justify-items: end;

```
.parent {
    border: 5px solid #000; width: 900px; height: 300px;
    background-color: LightCyan;
    display: grid;
    grid-template-columns: auto auto;
}
.parent div {
    color: #fff; text-align: center; font-size: 30px;
    background-color: DarkKhaki; width: 100px; height: 100px;
}
```

justify-items 속성에서
- start는 left로 사용 가능
- end는 right로 사용 가능



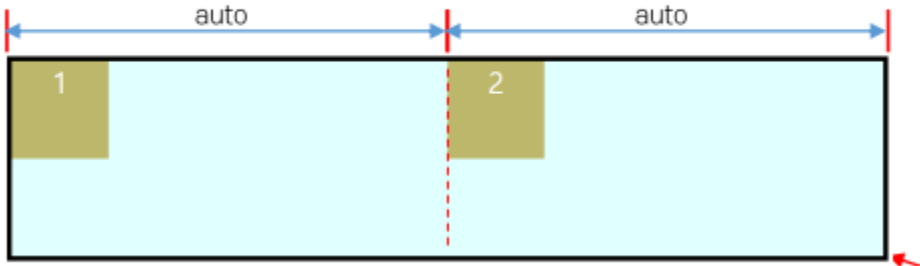
▲ justify-content: center;

HTML

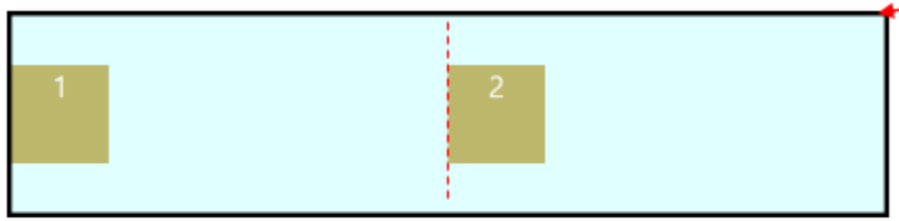
```
<div class="parent">
  <div>1</div>
  <div>2</div>
</div>
```

자식요소를 grid-template-column 안에서 각자의 컬럼 내에서 수직으로 정렬

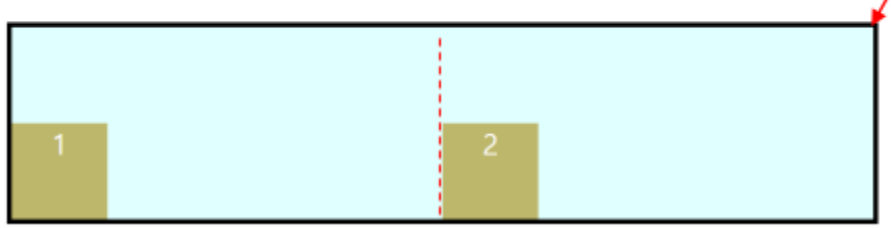
6) 부모요소에 사용하는 Grid 속성 align-items: start | center | end | stretch;



▲ align-items: start;



▲ align-items: center; ▲ align-content: center;



▲ align-items: end;

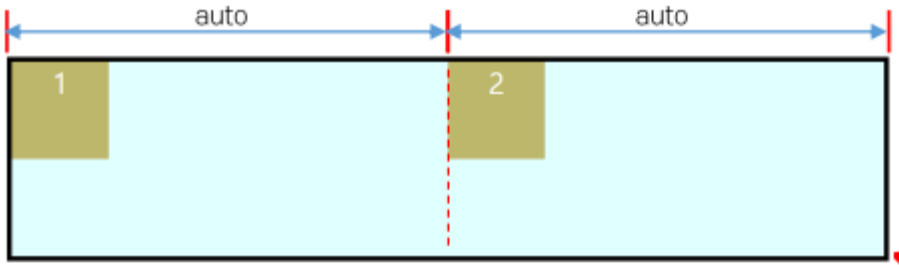
```
.parent {
    border: 5px solid #000; width: 900px; height: 300px;
    background-color: LightCyan;
    display: grid;
    grid-template-columns: auto auto;
}
.parent div {
    color: #fff; text-align: center; font-size: 30px;
    background-color: DarkKhaki; width: 100px; height: 100px;
}
```

HTML

```
<div class="parent">
  <div>1</div>
  <div>2</div>
</div>
```

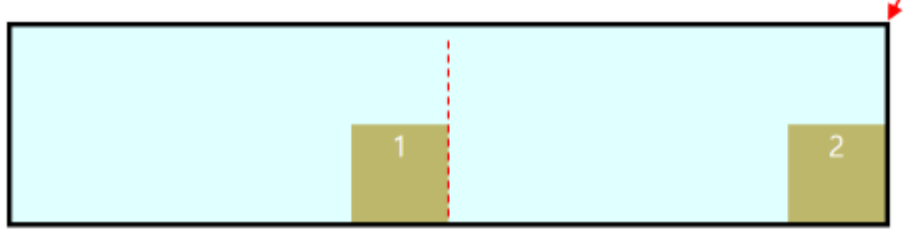
자식요소를 grid-template-column 안에서 각자의 컬럼 내에서 수평 수직으로 정렬

6-1) 부모요소에 사용하는 Grid 속성 justify-items와 align-items



▲ justify-items: start; align-items: start;

▲ justify-items: center; align-items: center;



▲ justify-items: end; align-items: end;

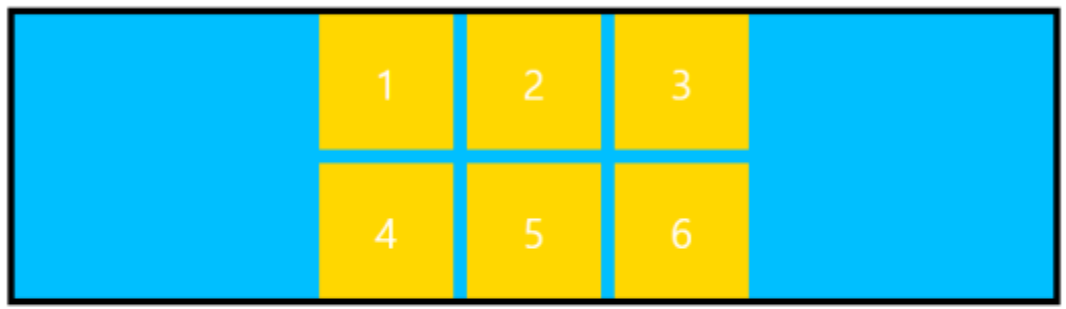
```
.parent {
    border: 5px solid #000; width: 900px; height: 300px;
    background-color: LightCyan;
    display: grid;
    grid-template-columns: auto auto;
}
.parent div {
    color: #fff; text-align: center; font-size: 30px;
    background-color: DarkKhaki; width: 100px; height: 100px;
}
```

HTML

```
<div class="parent">
  <div>1</div>
  <div>2</div>
</div>
```

7-1) 부모요소에 사용하는 Grid 속성 justify-content: center;

자식요소를 수평으로 정렬



HTML

```
<div class="parent">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
</div>
```

```
.parent {
    border: 5px solid #000;
    background-color: deepskyblue;
    display: grid;
    grid-template-columns: auto auto auto;
    grid-gap: 10px;
    justify-content: center;
}

  .parent div {
    color: #fff; text-align: center;
    font-size: 30px;
    background-color: gold;
    width: 100px;
    padding: 20px 0;
}
```

수평을 기준으로 자식요소를 중앙 정렬

(플렉스 부모요소 속성 justfy-content 의 개념과 모두 동일함)

7-2) 부모요소에 사용하는 Grid 속성 justify-content: start;

자식요소를 수평으로 정렬

HTML

```
<div class="parent">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
</div>
```

```
.parent {
    border: 5px solid #000;
    background-color: deepskyblue;
    display: grid;
    grid-template-columns: auto auto auto;
    grid-gap: 10px;
    justify-content: start;
}
  .parent div {
    color: #fff; text-align: center;
    font-size: 30px;
    background-color: gold;
    width: 100px;
    padding: 20px 0;
}
```

수평을 기준으로 자식요소를 왼쪽 정렬

(플렉스 부모요소 속성 justfy-content의 개념과 모두 동일함)

7-3) 부모요소에 사용하는 Grid 속성 justify-content: end;

자식요소를 수평으로 정렬

HTML

```
<div class="parent">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
</div>
```

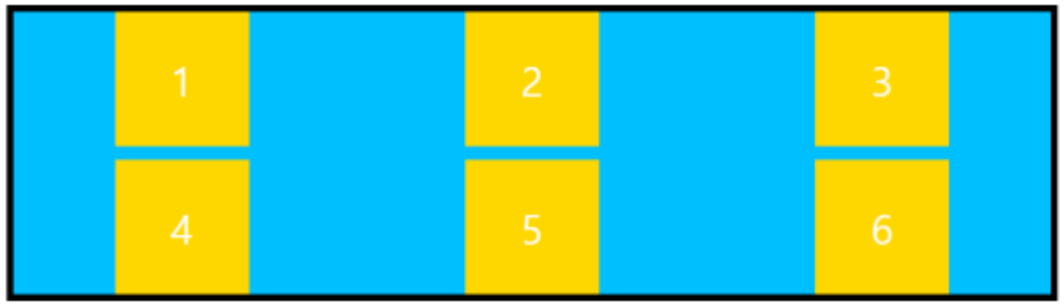
```
.parent {
    border: 5px solid #000;
    background-color: deepskyblue;
    display: grid;
    grid-template-columns: auto auto auto;
    grid-gap: 10px;
    justify-content: end;
}
  .parent div {
    color: #fff; text-align: center;
    font-size: 30px;
    background-color: gold;
    width: 100px;
    padding: 20px 0;
}
```

수평을 기준으로 자식요소를 오른쪽 정렬

(플렉스 부모요소 속성 justfy-content의 개념과 모두 동일함)

자식요소를 수평으로 정렬

7-4) 부모요소에 사용하는 Grid 속성 justify-content: space-around;



HTML

```
<div class="parent">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
</div>
```

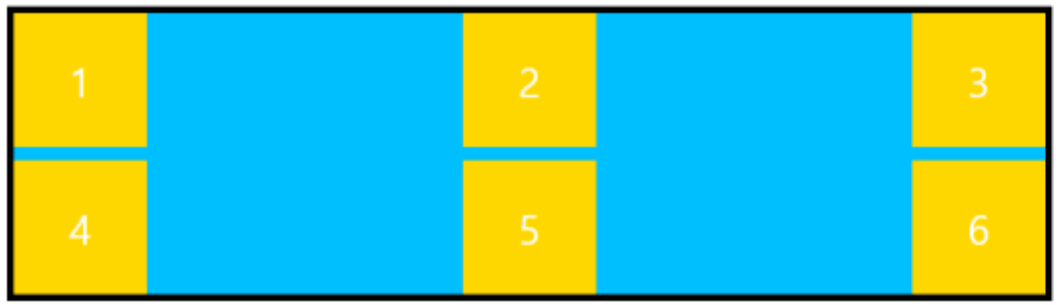
```
.parent {
    border: 5px solid #000;
    background-color: deepskyblue;
    display: grid;
    grid-template-columns: auto auto auto;
    grid-gap: 10px;
    justify-content: space-around;
}
  .parent div {
    color: #fff; text-align: center;
    font-size: 30px;
    background-color: gold;
    width: 100px;
    padding: 20px 0;
}
```

자식요소를 수평으로 양쪽 정렬

(플렉스 부모요소 속성 justfy-content의 개념과 모두 동일함)

7-5) 부모요소에 사용하는 Grid 속성 justify-content: space-between;

↗ 자식요소를 수평으로 정렬



HTML

```
<div class="parent">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
</div>
```

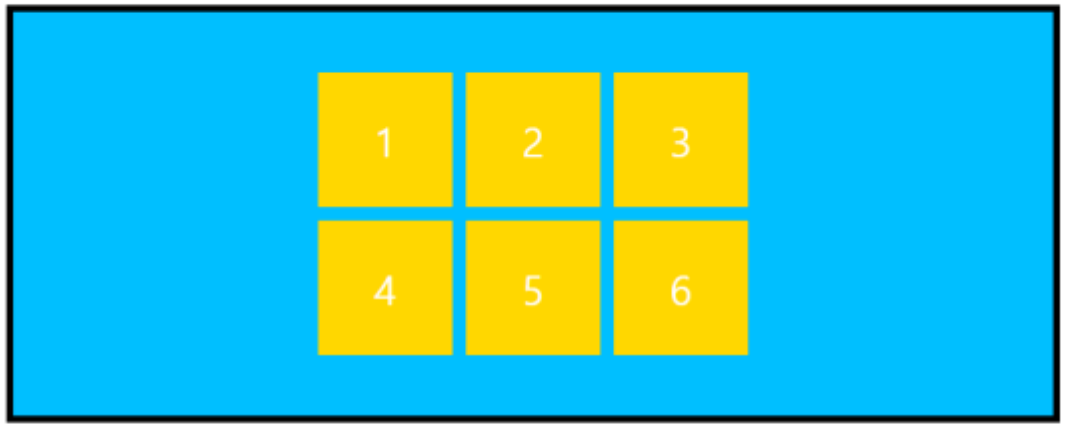
```
.parent {
    border: 5px solid #000;
    background-color: deepskyblue;
    display: grid;
    grid-template-columns: auto auto auto;
    grid-gap: 10px;
    justify-content: space-between;
}
  .parent div {
    color: #fff; text-align: center;
    font-size: 30px;
    background-color: gold;
    width: 100px;
    padding: 20px 0;
}
```

→ 자식요소를 수평으로 양쪽 정렬(첫번째 컬럼과 마지막 컬럼은 부모요소 양끝에 여백 없이 붙음)

(플렉스 부모요소 속성 justfy-content의 개념과 모두 동일함)

8-1) 부모요소에 사용하는 Grid 속성 align-content: center;

자식요소를 수직으로 정렬



HTML

```
<div class="parent">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
</div>
```

```
.parent {
    border: 5px solid #000;
    background-color: deepskyblue;
    height: 300px;
    display: grid;
    grid-template-columns: auto auto auto;
    grid-gap: 10px;
    justify-content: center;
    align-content: center;
}
  .parent div {
    color: #fff; text-align: center;
    font-size: 30px; background-color: gold;
    width: 100px; padding: 30px 0;
}
```

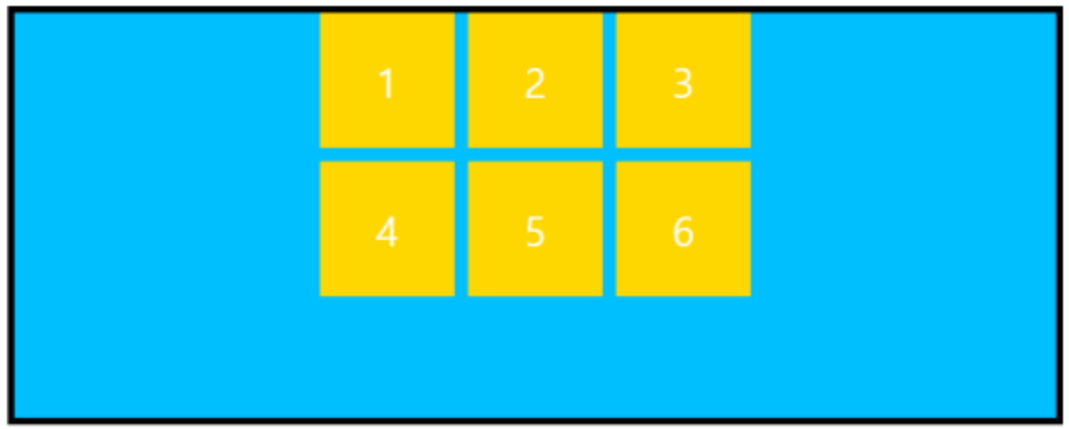
height: 300px; → 수직 정렬이기 때문에 부모요소의 높이 값이 필요

align-content: center; → 수직을 기준으로 자식요소를 중앙 정렬

(플렉스 부모요소 속성 align-items의 개념과 모두 동일함)

자식요소를 수직으로 정렬

8-2) 부모요소에 사용하는 Grid 속성 align-content: start;



HTML

```
<div class="parent">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
</div>
```

```
.parent {
    border: 5px solid #000;
    background-color: deepskyblue;
    height: 300px;
    display: grid;
    grid-template-columns: auto auto auto;
    grid-gap: 10px;
    justify-content: center;
    align-content: start;
}
  .parent div {
    color: #fff; text-align: center;
    font-size: 30px; background-color: gold;
    width: 100px; padding: 30px 0;
}
```

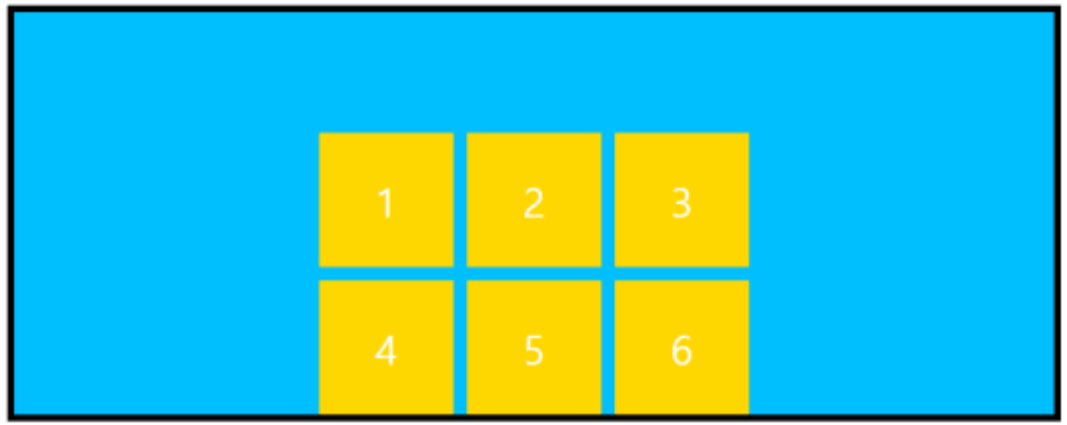
height: 300px; → 수직 정렬이기 때문에 부모요소의 높이 값이 필요

align-content: start; → 수직을 기준으로 자식요소를 상단 정렬

(플렉스 부모요소 속성 align-items의 개념과 모두 동일함)

자식요소를 수직으로 정렬

8-3) 부모요소에 사용하는 Grid 속성 align-content: end;



HTML

```
<div class="parent">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
</div>
```

```
.parent {
    border: 5px solid #000;
    background-color: deepskyblue;
    height: 300px;
    display: grid;
    grid-template-columns: auto auto auto;
    grid-gap: 10px;
    justify-content: center;
    align-content: end;
}
  .parent div {
    color: #fff; text-align: center;
    font-size: 30px; background-color: gold;
    width: 100px; padding: 30px 0;
}
```

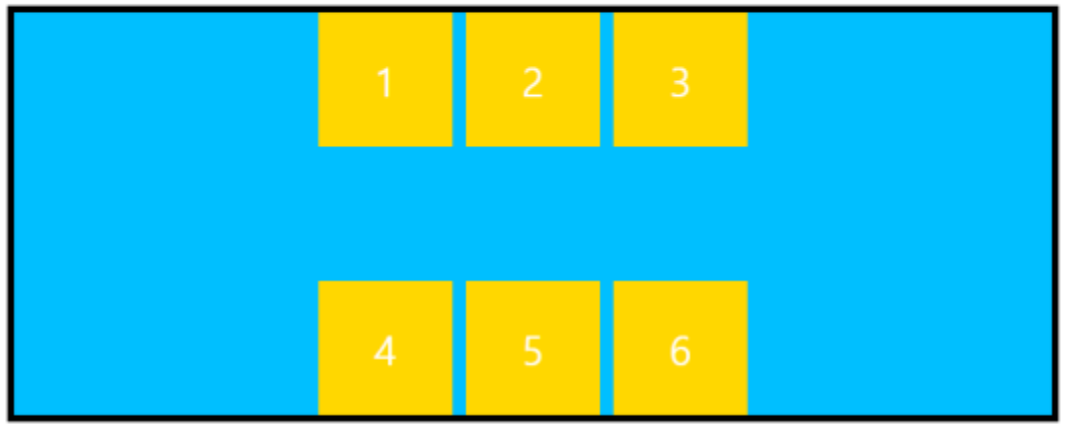
수직 정렬이기 때문에 부모요소의 높이 값이 필요

수직을 기준으로 자식요소를 하단 정렬

(플렉스 부모요소 속성 align-items의 개념과 모두 동일함)

8-4) 부모요소에 사용하는 Grid 속성 align-content: space-between;

자식요소를 수직으로 정렬



HTML

```
<div class="parent">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
</div>
```

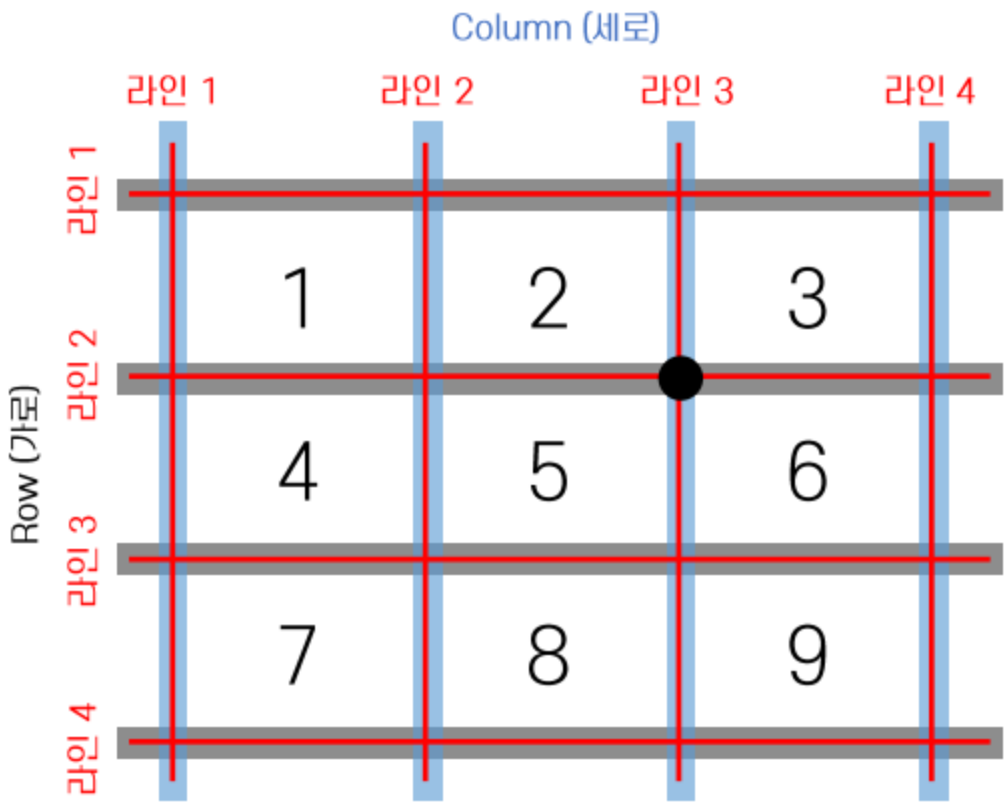
```
.parent {
    border: 5px solid #000;
    background-color: deepskyblue;
    height: 300px;
    display: grid;
    grid-template-columns: auto auto auto;
    grid-gap: 10px;
    justify-content: center;
    align-content: space-between;
}

  .parent div {
    color: #fff; text-align: center;
    font-size: 30px; background-color: gold;
    width: 100px; padding: 30px 0;
}
```

height: 300px; → 수직 정렬이기 때문에 부모요소의 높이 값이 필요

align-content: space-between; → 수직을 기준으로 자식요소를 상하 양쪽 정렬(첫번째 컬럼과 마지막 컬럼은 부모요소 상하 양끝에 여백 없이 붙음)

(플렉스 부모요소 속성 align-items의 개념과 모두 동일함)

자식요소에 사용하는 Grid 속성을 위한 필수 개념



그리드 레이아웃의 특이한 점은 라인을 기준으로 크기를 결정합니다. 왼쪽 그림을 보면서 이해하시면 됩니다.

- 컬럼(세로)이 3개라면 컬럼 라인은 4개가 있습니다.
- 로우(가로)가 3개라면 로우 라인은 4개가 있습니다.
- 라인의 숫자는 컬럼과 로우의 숫자 +1
- 왼쪽 검은색 동그라미(●)의 grid-column은 3, grid-row는 2 입니다.
- 왼쪽 검은색 동그라미(●)의 grid-area는 가로 2와 세로 3입니다.

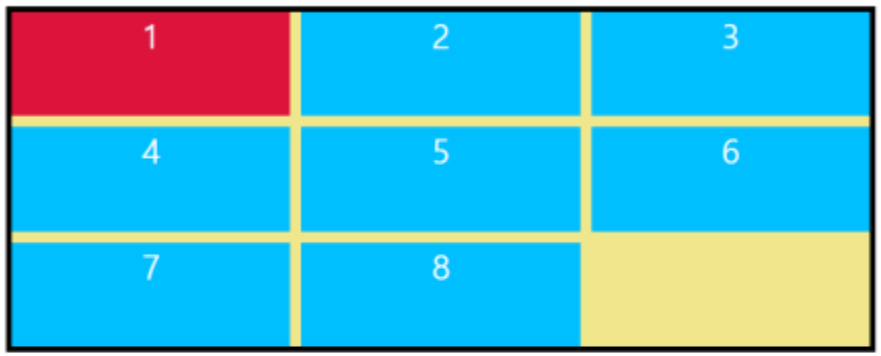
HTML

```
<div class="parent">
  <div class="child">1</div>
  <div class="child">2</div>
  <div class="child">3</div>
  <div class="child">4</div>
  <div class="child">5</div>
  <div class="child">6</div>
  <div class="child">7</div>
  <div class="child">8</div>
  <div class="child">9</div>
</div>
```

2-1) 자식요소에 사용하는 Grid 속성 grid-column-start, grid-column-end

자식요소 세로 기준 시작 위치

자식요소 세로 기준 끝 위치

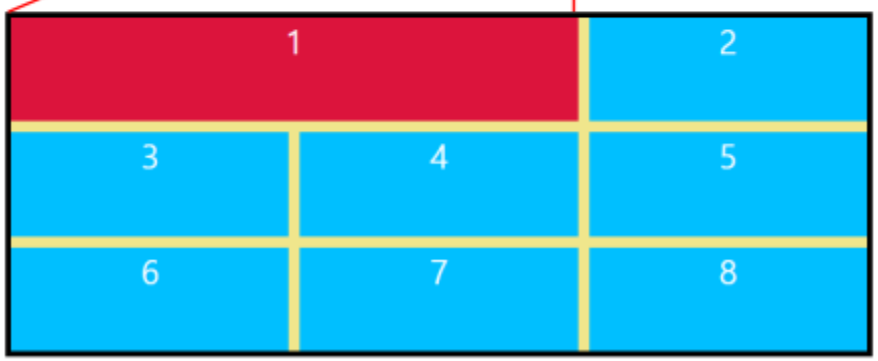


HTML

```
<div class="parent">
  <div class="item">1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
</div>
```

컬럼 기준 라인1에서 시작(start)

컬럼 기준 라인3에서 끝(end)

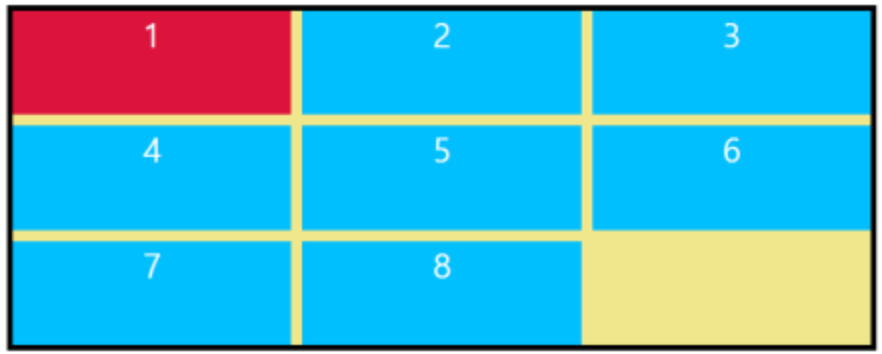



```
.parent {
    border: 5px solid #000;
    background-color: khaki; height: 300px;
    display: grid;
    grid-template-columns: auto auto auto;
    grid-gap: 10px;
}
  .parent div {
    color: #fff; text-align: center;
    font-size: 30px; background-color: deepskyblue;
}
  .item {
     background-color: crimson;
     grid-column-start: 1;
     grid-column-end: 3;
}
```

2-2) 자식요소에 사용하는 Grid 속성 grid-column: 정수 / 정수

자식요소 세로 기준 시작 위치

자식요소 세로 기준 끝 위치

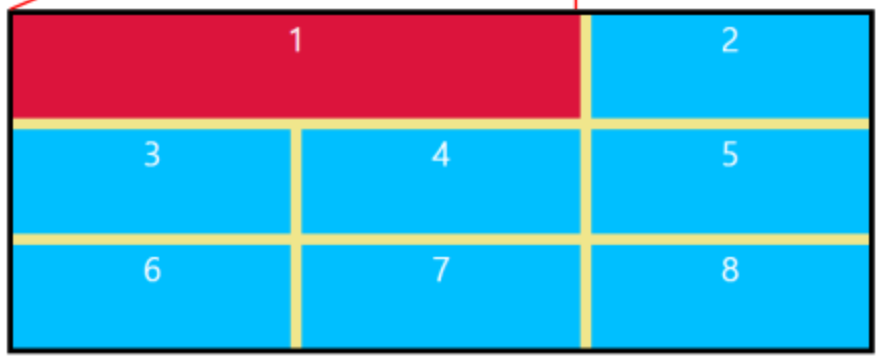


HTML

```
<div class="parent">
  <div class="item">1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
</div>
```

컬럼 기준 라인1에서 시작(start)

컬럼 기준 라인3에서 끝(end)



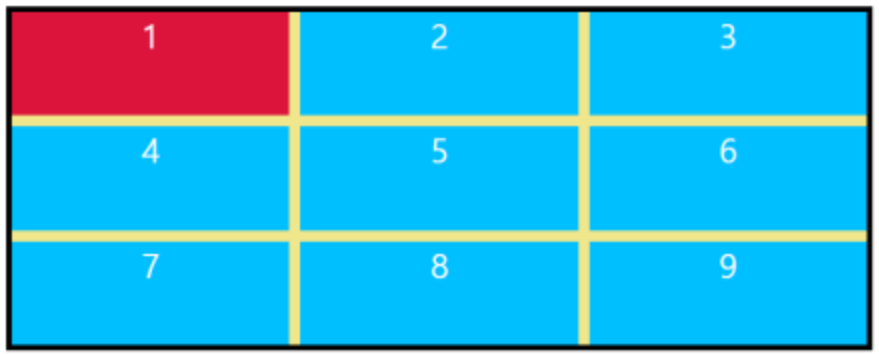

```
.parent {
    border: 5px solid #000;
    background-color: khaki; height: 300px;
    display: grid;
    grid-template-columns: auto auto auto;
    grid-gap: 10px;
}
.parent div {
    color: #fff; text-align: center;
    font-size: 30px; background-color: deepskyblue;
}
. Item {
    background-color: crimson;
    grid-column: 1 / 3;
}
```

정수/정수 처럼 슬래시(/) 좌우로 띄어쓰기를 하지 않아도 되지만 가독성을 위해서 띄어쓰기를 사용하는 것이 좋습니다.

3-1) 자식요소에 사용하는 Grid 속성 grid-row-start, grid-row-end

자식요소 가로 기준 시작 위치

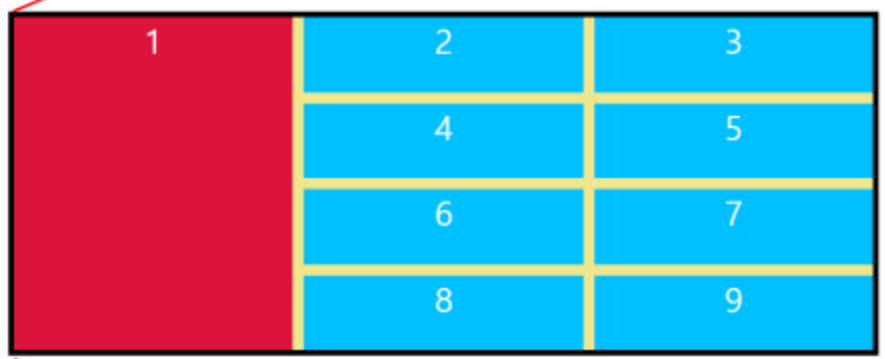
자식요소 가로 기준 끝 위치



HTML

```
<div class="parent">
  <div class="item">1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
  <div>9</div>
```

로우 기준 라인1에서 시작(start)



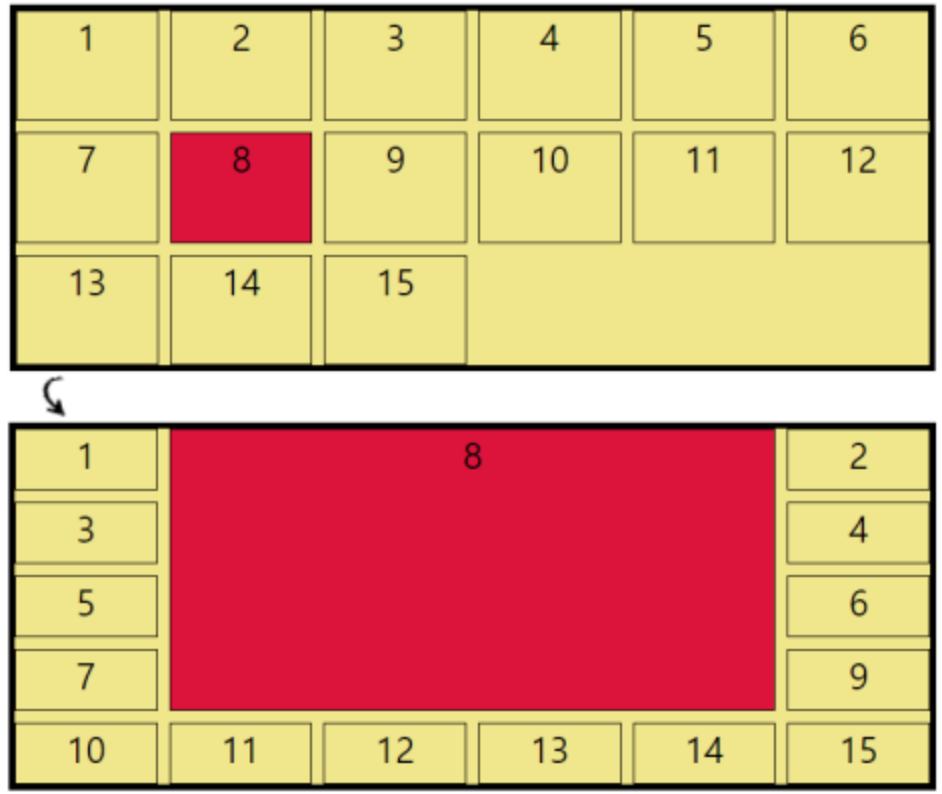

```
.parent {
    border: 5px solid #000;
    background-color: khaki; height: 300px;
    display: grid; grid-gap: 10px;
    grid-template-columns: auto auto auto;
}
.parent div {
    color: #fff; text-align: center;
    font-size: 30px; background-color: deepskyblue;
}
.item {
     background-color: crimson;
     grid-row-start: 1;
     grid-row-end: 5;
     /* grid-row: 1 / 5 */
}
```

4-1) 자식요소에 사용하는 Grid 속성 grid-area

자식요소 가로 세로 시작위치에서 가로 세로 종료 위치까지의 영역 지정

HTML

```
<div class="parent">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div class="item">8</div>
  <div>9</div>
  <div>10</div>
  <div>11</div>
  <div>12</div>
  <div>13</div>
  <div>14</div>
  <div>15</div>
</div>
```




```
.parent {
    border: 5px solid #000;
    background-color: khaki; height: 300px;
    display: grid; grid-gap: 10px;
    grid-template-columns: auto auto auto auto auto auto;
}
```

세로로 6개 나눔

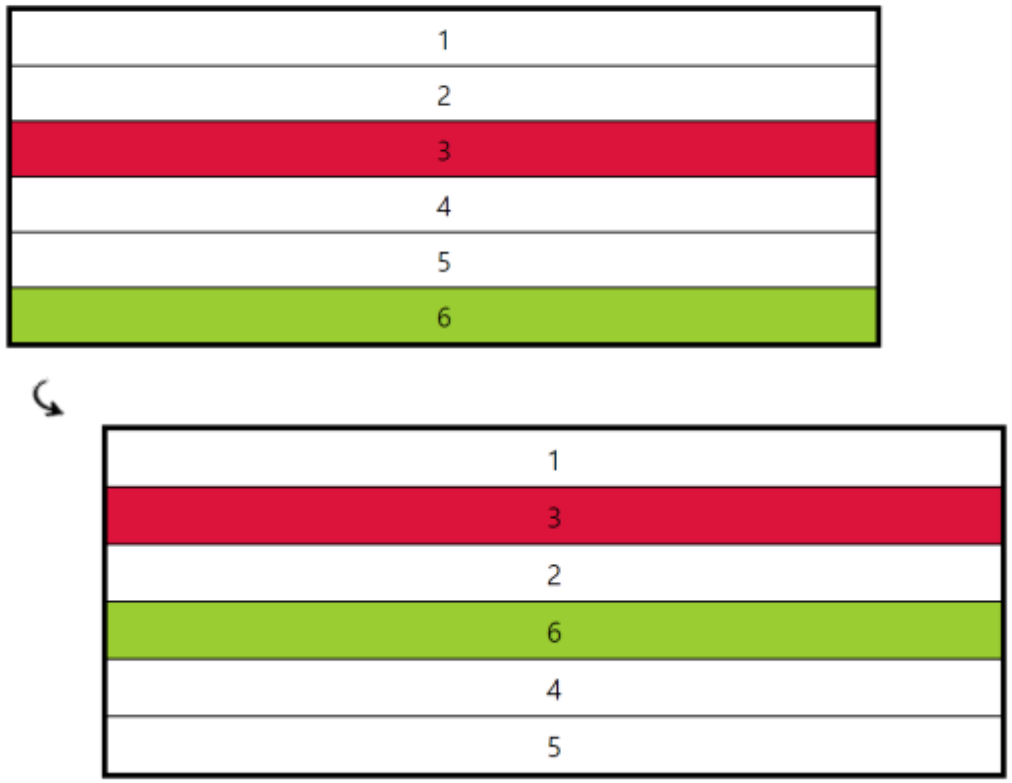
```
.parent div {
    text-align: center; font-size: 30px;
    border: 1px solid #000;
}
.item {
    background-color: crimson;
    grid-area: 1 / 2 / 5 / 6;
}
```

가로 / 세로 순서를 세로 / 가로 순서로 입력하면 안됩니다.

영역 종료 위치 (가로 종료 라인 / 세로 종료 라인)

영역 시작 위치 (가로 시작 라인 / 세로 시작라인)

6) 자식요소에 사용하는 Grid 속성 grid-row 속성으로 Row 순서 변경




```css
.parent div {
    border: 5px solid #000;
    display: grid;
    text-align: center;
    font-size: 30px;
    border: 1px solid #000;
    padding: 10px;
}
.item1 {
    background-color: crimson;
    grid-row: 2;
}
.item2 {
    background-color: yellowgreen;
    grid-row: 4;
}
```

grid-row: 2; → 가로의 출연 순서를 2번째로 변경

grid-row: 4; → 가로의 출연 순서를 2번째로 변경

HTML

```html
<div class="parent">
    <div>1</div>
    <div>2</div>
    <div class="item1">3</div>
    <div>4</div>
    <div>5</div>
    <div class="item2">6</div>
</div>
```