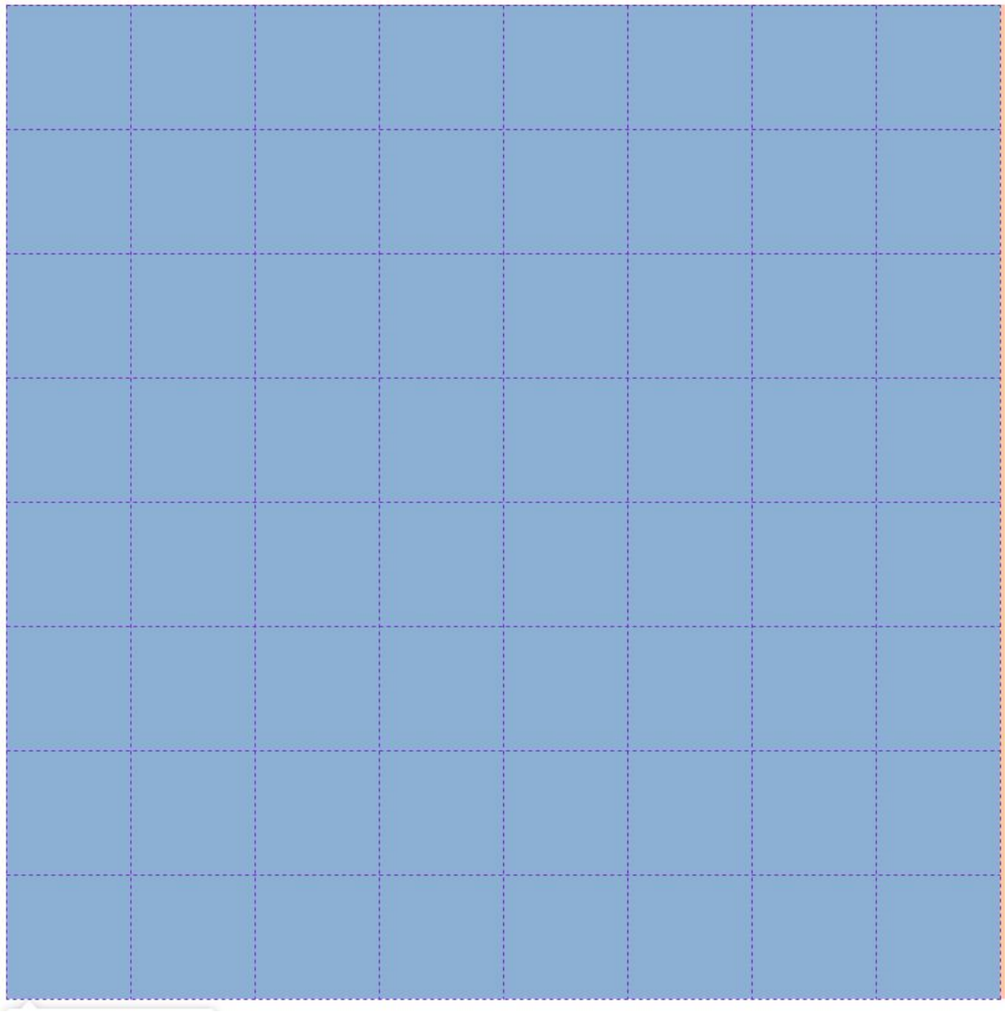
Grid Layout

Дозволяє поділити контейнер на N кількість рядків та стовбців, для майбутнього їх заповнення дочірніми елементами



```
.target {
    width: 800px;
    height: 800px;
    background: silver;
    display: grid;
    grid-template-columns:repeat(8, 100px);
    grid-template-rows: repeat(8, 100px);
}
```

Columns & rows

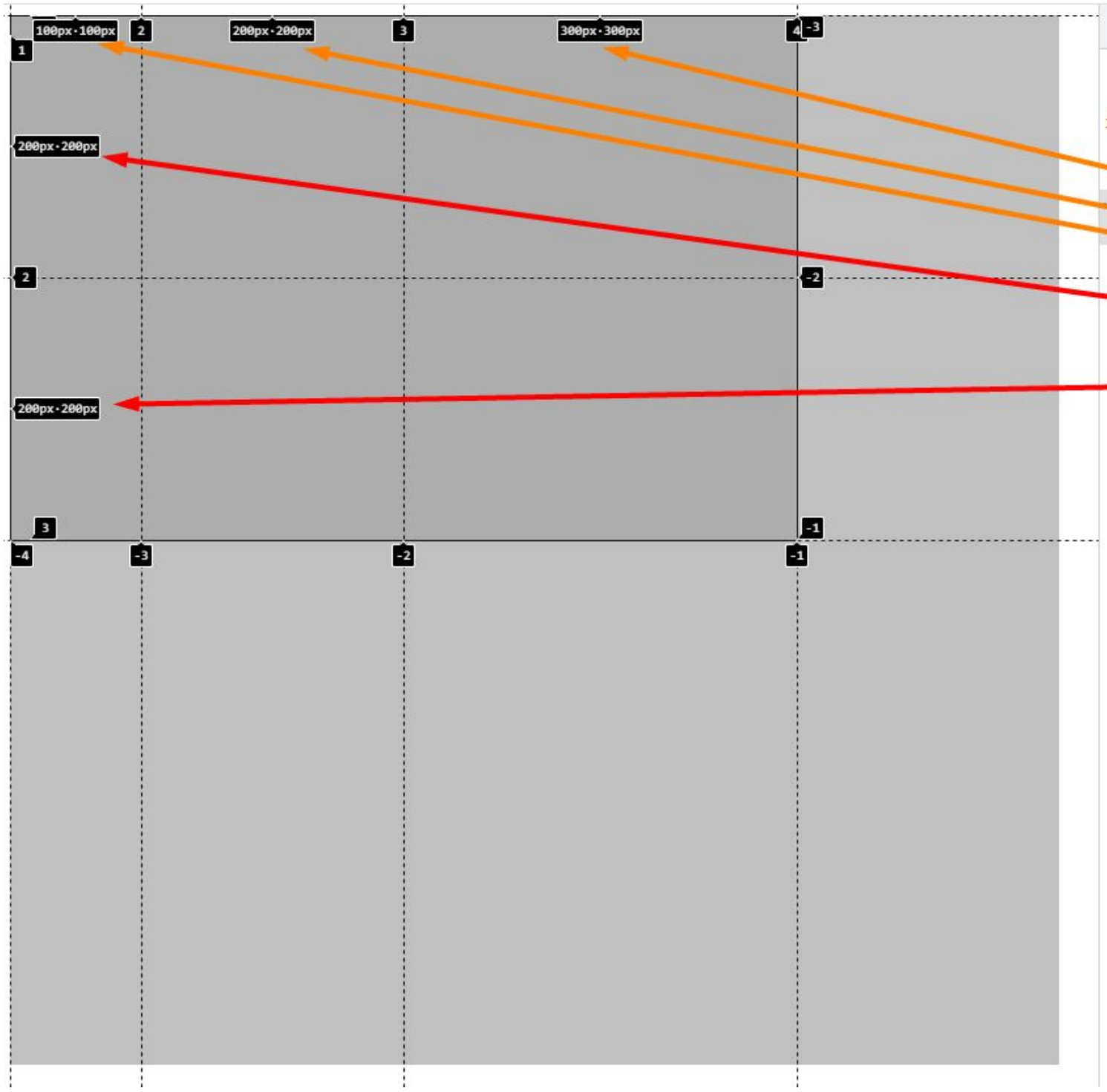
grid-template-columns - визначає кількість колонок, та яку ширину вони займають.

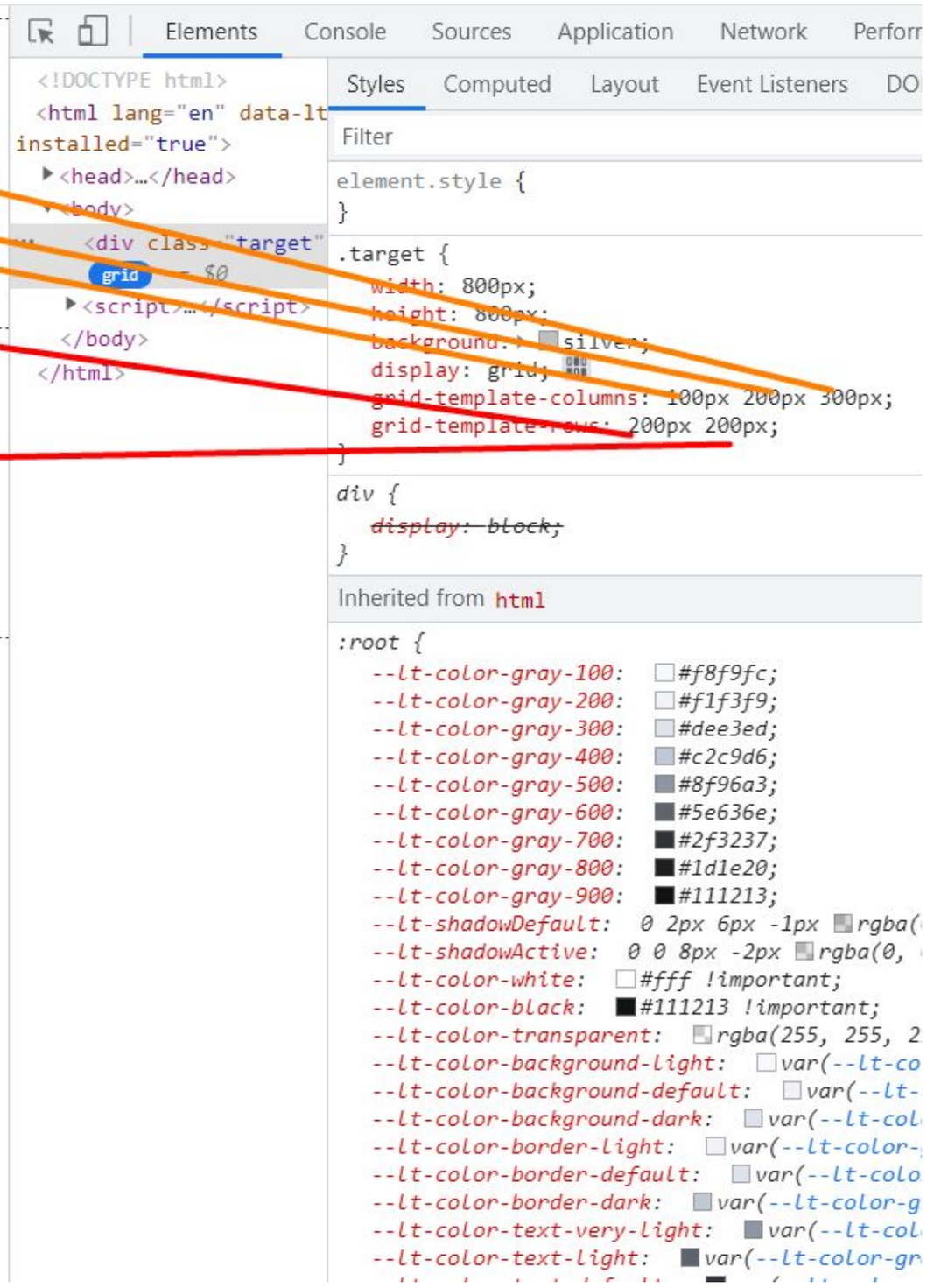
grid-template-rows - визначає кількість колонок та висоту яку вони займають.
Розміри можуть бути задані в будь-яких дозволених одиницях вимірювання, а також в нових одиницях fraction (fr)

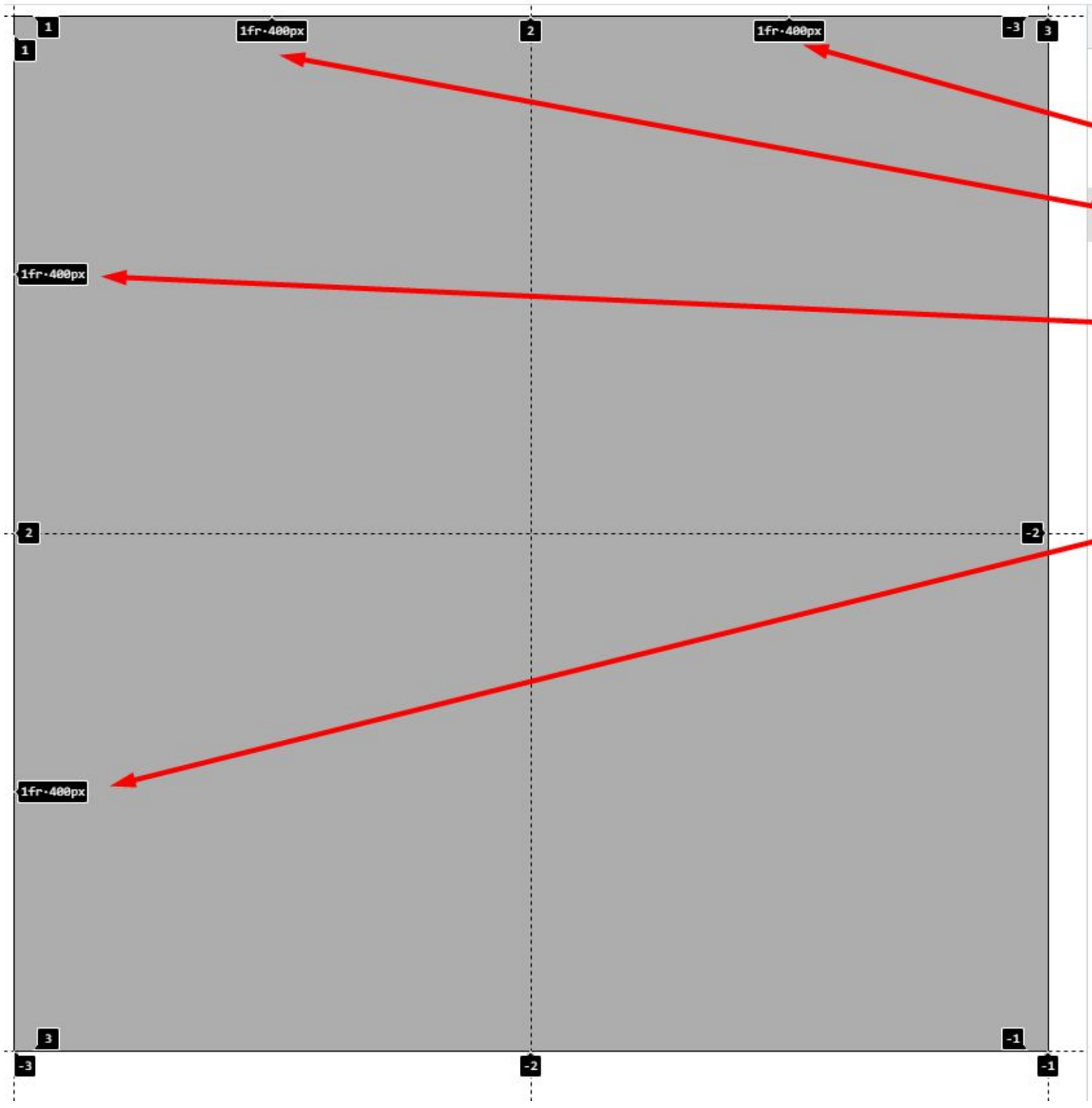
Fractions (fr)- відносна одиниця вимірювання яка застосовується для визначення ширини блоку. Для одного блоку 1rf буде дорівнювати 100% ширини. Для двох блоків по 1fr, це буде по 50% ширини. Для трьох блоків , значення яких , наприклад, буде 1fr 2fr 1fr - це буде (25%,50%,25%).
Ми чітко бачимо, що це - просто частки. Загальна кількість часток в останньому прикладі - 4. Відповідно 1/4 - 25%, 2/4 - 50%, 1/4 - 25%;

grid-auto-rows - дозволяє визначити висоту колонки, а за допомоги функції minmax(minvalue,maxvalue) (minvalue та maxvalue - значення у px,% і тд) можливо задати мінімальну та максимальну висоту колонки у випадку якщо в комірці є багато тексту.

grid-auto-columns - виконує ту саму функцію, але впливає на ширину колонки.


100px·100px
200px·200px
300px·300px
200px·200px
200px·200px
1
2
3
-3
-2
-1
-4
-3
-2
-1


Elements
Console
Sources
Application
Network
Perforr
<!DOCTYPE html>
<html lang="en" data-lt
installed="true">
<head>…</head>
<body>
<div class="target"
grid
$0
<script>…</script>
</body>
</html>
Styles
Computed
Layout
Event Listeners
DO
Filter
element.style {
}
.target {
width: 800px;
height: 800px;
background: silver;
display: grid;
grid-template-columns: 100px 200px 300px;
grid-template-rows: 200px 200px;
}
div {
display: block;
}
Inherited from html
:root {
--lt-color-gray-100: #f8f9fc;
--lt-color-gray-200: #f1f3f9;
--lt-color-gray-300: #dee3ed;
--lt-color-gray-400: #c2c9d6;
--lt-color-gray-500: #8f96a3;
--lt-color-gray-600: #5e636e;
--lt-color-gray-700: #2f3237;
--lt-color-gray-800: #1d1e20;
--lt-color-gray-900: #111213;
--lt-shadowDefault: 0 2px 6px -1px rgba(
--lt-shadowActive: 0 0 8px -2px rgba(0,
--lt-color-white: #fff !important;
--lt-color-black: #111213 !important;
--lt-color-transparent: rgba(255, 255, 2
--lt-color-background-light: var(--lt-co
--lt-color-background-default: var(--lt-
--lt-color-background-dark: var(--lt-col
--lt-color-border-light: var(--lt-color-
--lt-color-border-default: var(--lt-colo
--lt-color-border-dark: var(--lt-color-g
--lt-color-text-very-light: var(--lt-col
--lt-color-text-light: var(--lt-color-gr



Elements Console Sources Application Network Performance

```
<!DOCTYPE html>
<html lang="en" data-lt-installed="true">
▶ <head>…</head>
▼ <body>
    <div class="target">…</div>
      grid  == $0
  ▶ <script>…</script>
  </body>
</html>
```

Styles Computed Layout Event Listeners

Filter

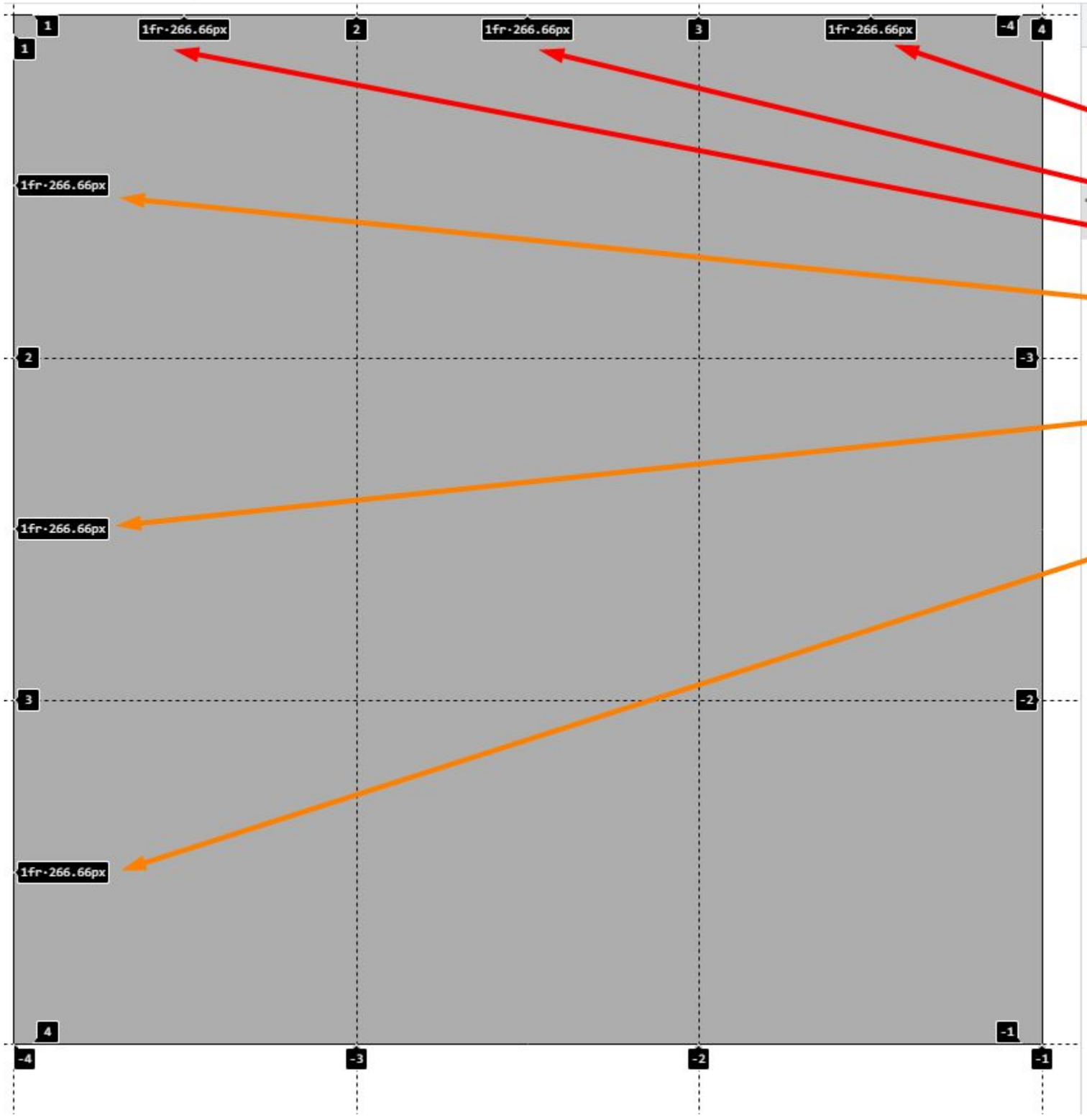
```
element.style {
}

.target {
  width: 800px;
  height: 800px;
  background: ▶ silver;
  display: grid;
  grid-template-columns: 1fr 1fr;
  grid-template-rows: 1fr 1fr;
}

div {
  display: block;
}
```

Inherited from `html`

```
:root {
  --lt-color-gray-100: #f8f9fc;
  --lt-color-gray-200: #f1f3f9;
  --lt-color-gray-300: #dee3ed;
  --lt-color-gray-400: #c2c9d6;
  --lt-color-gray-500: #8f96a3;
  --lt-color-gray-600: #5e636e;
  --lt-color-gray-700: #2f3237;
  --lt-color-gray-800: #1d1e20;
  --lt-color-gray-900: #111213;
  --lt-shadowDefault: 0 2px 6px -1px
    rgba(0, 0, 0, 0.04);
  --lt-shadowActive: 0 0 8px -2px rgb
    rgba(0, 0, 0, 0.2);
  --lt-color-white: #fff !important;
  --lt-color-black: #111213 !importan
  --lt-color-transparent: rgba(255, 2
  --lt-color-background-light: var(--
  --lt-color-background-default: var(
  --lt-color-background-dark: var(--l
  --lt-color-border-light: var(--lt-c
  --lt-color-border-default: var(--lt
  --lt-color-border-dark: var(--lt-co
  --lt-color-text-very-light: var(--l
```



Elements Console Sources Application Network

```
<!DOCTYPE html>
<html lang="en" data-lt
installed="true">
  ▶ <head>…</head>
  ▼ <body>
    ⋯ <div class="target"
        grid == $0
    ▶ <script>…</script>
    </body>
  </html>
```

Styles Computed Layout Event Listene

Filter

```
element.style {
}

.target {
    width: 800px;
    height: 800px;
    background: silver;
    display: grid;
    grid-template-columns: 1fr 1fr 1fr;
    grid-template-rows: 1fr 1fr 1fr;
}

div {
    display: block;
}
```

Inherited from `html`

```
:root {
    --lt-color-gray-100: #f8f9fc;
    --lt-color-gray-200: #f1f3f9;
    --lt-color-gray-300: #dee3ed;
    --lt-color-gray-400: #c2c9d6;
    --lt-color-gray-500: #8f96a3;
    --lt-color-gray-600: #5e636e;
    --lt-color-gray-700: #2f3237;
    --lt-color-gray-800: #1d1e20;
    --lt-color-gray-900: #111213;
    --lt-shadowDefault: 0 2px 6px -1px
    --lt-shadowActive: 0 0 8px -2px r
    --lt-color-white: #fff !important
    --lt-color-black: #111213 !import
    --lt-color-transparent: rgba(255,
    --lt-color-background-light: var(
    --lt-color-background-default: va
    --lt-color-background-dark: var(-
    --lt-color-border-light: var(--lt
    --lt-color-border-default: var(--
    --lt-color-border-dark: var(--lt-
    --lt-color-text-very-light: var(-
    --lt-color-text-light: var(--lt-c
    --lt-color-text-default: var(--lt
    --lt-color-text-dark: var(--lt-co
```

grid-gap

grid-column-gap - Дозволяє надавати відступи між колонками, які будуть враховані у вже існуючі розміри колонок.

grid-row-gap - дозволяє надавати відступи між рядками

grid-gap - композитна характеристика, дозволяє одночасно задати відступи між колонками та рядками

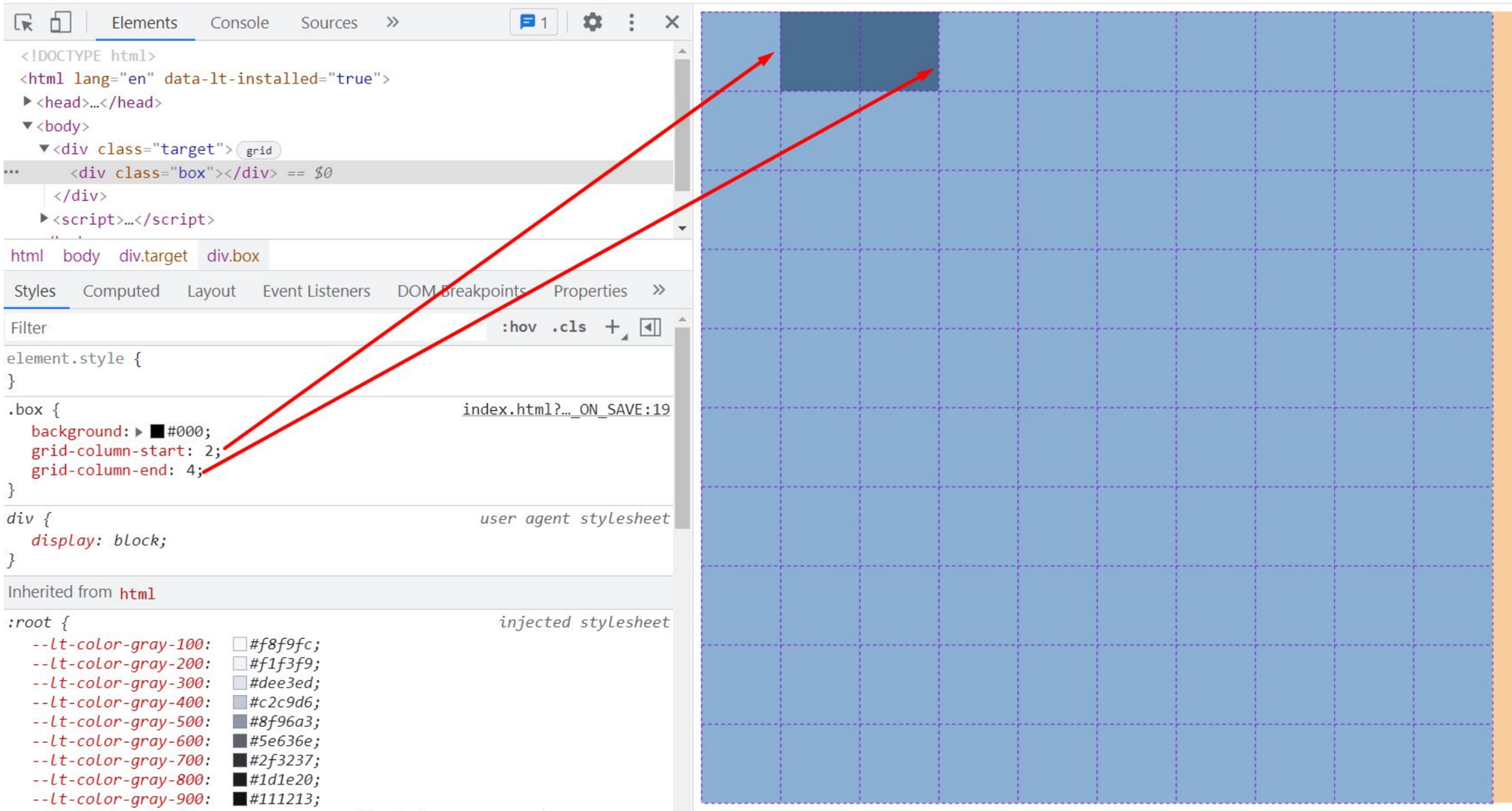
grid-column & grid-row

Уявимо собі, що наш контейнер поділений на 10 рядків та 10 колонок. та в середині є блок із класом box

```
.target {
    width: 800px;
    height: 800px;
    background: silver;
    display: grid;
    grid-template-columns:repeat(10, 1fr);
    grid-template-rows:repeat(10, 1fr);
}
.box{
    background: #000;

}
```

Використовуючи властивість grid-column-start ми можемо вказати з якої колонки починає розташовуватись блок, а за допомоги властивості grid-column-end - вказати де закінчується блок.

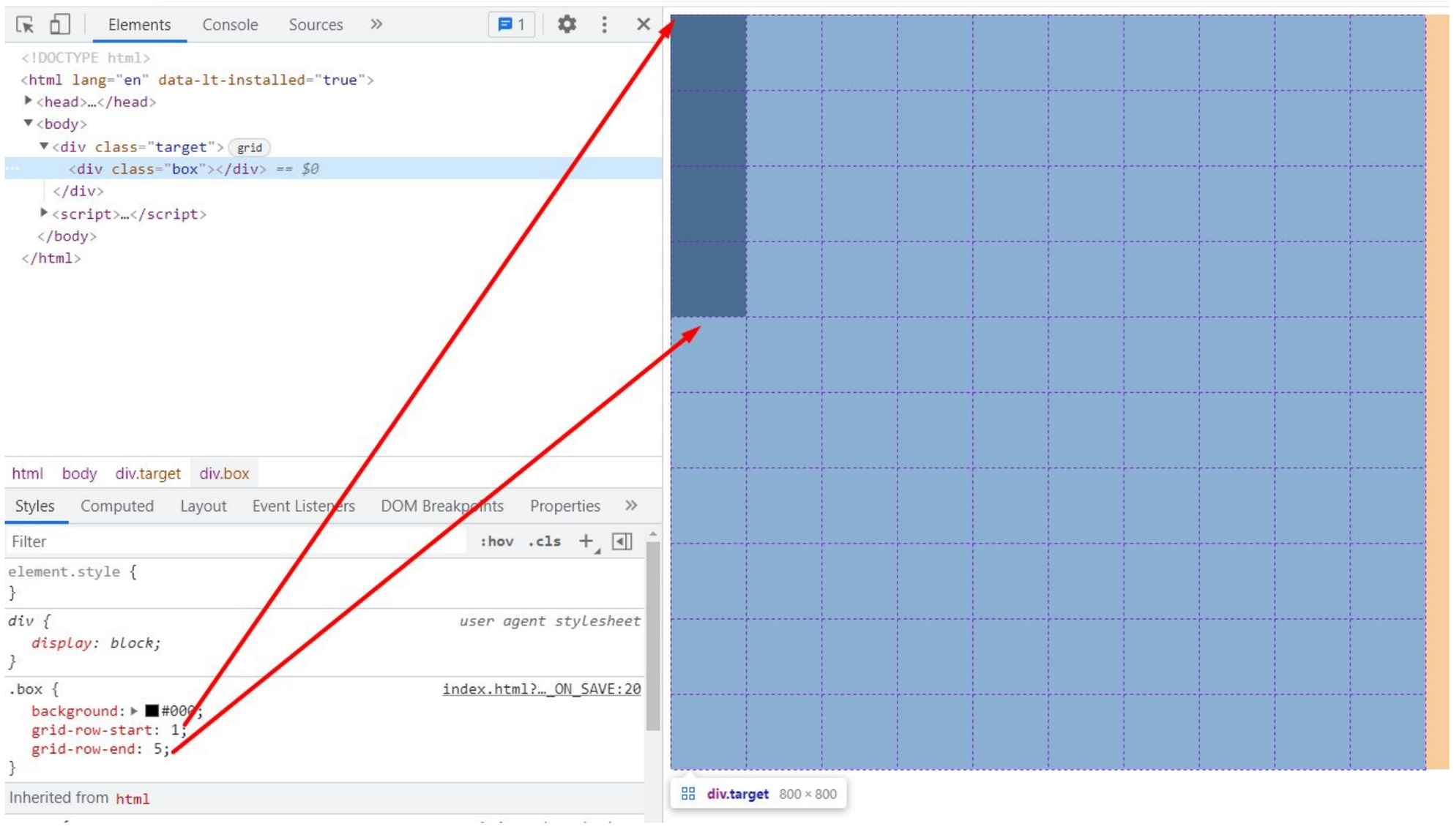


Важливо пам’ятати,що значення start & end (2 та 4 в прикладі) характеризують лнії grid системи, а не номери колонок чи рядків

подібна характеристика спроможна впливати на розмір об’єднання рядків.

```
.box {
    background: #000;
    grid-row-start: 1;
    grid-row-end: 5;

}
```



Для скороченого формату запису можна використовувати наступны властивост:

```
grid-row: 1/100;
grid-column: 2/10;
```

Де перша цирфа - значення start , друга - end

grid-template-areas

Дозволяэ визначати розташування та пропорцію блоків використовуючи тільки css.

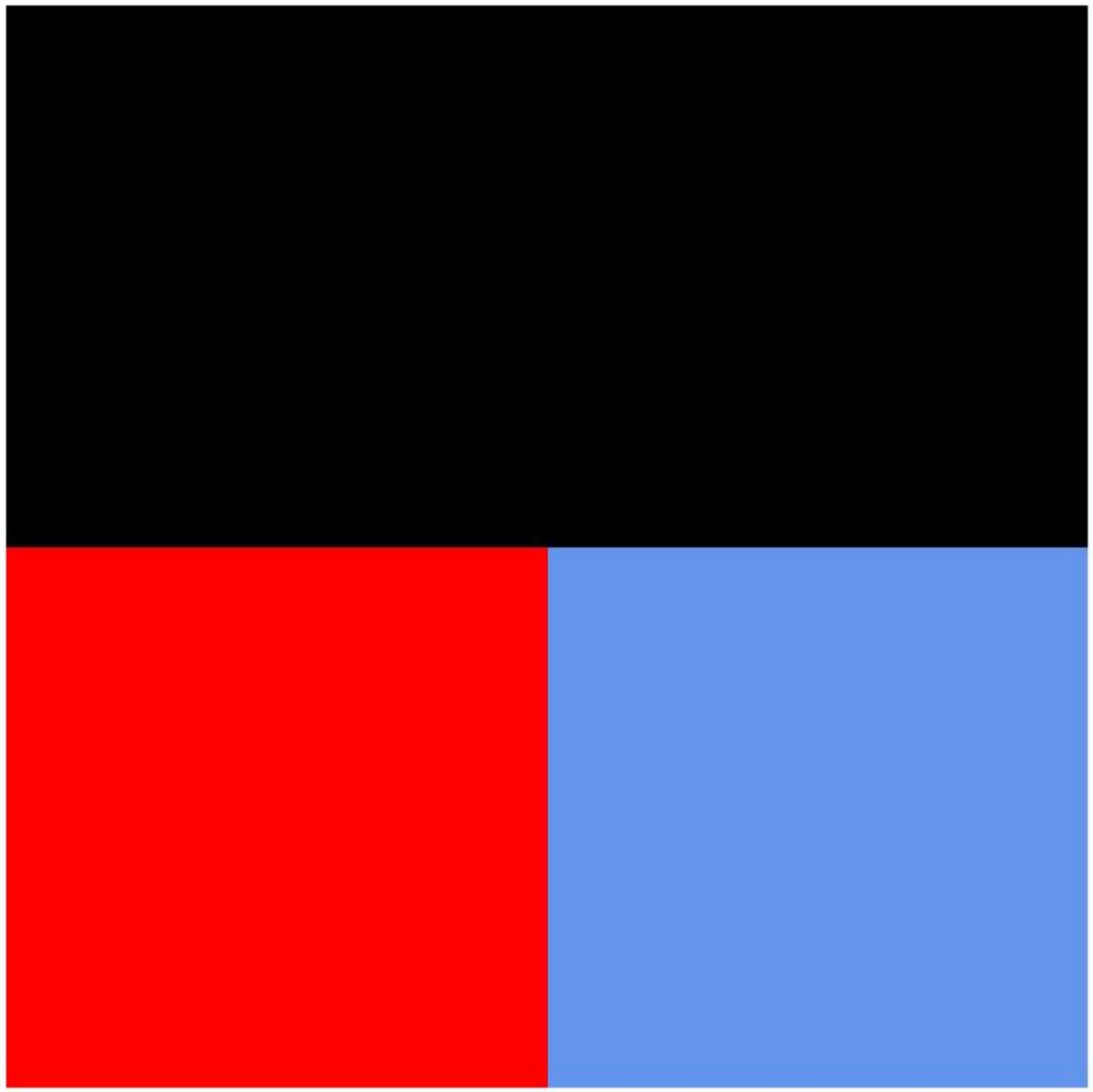
```css
.target {
    width: 600px;
    height: 600px;
    background: silver;
    display: grid;
    grid-template-columns: 1fr 1fr;
    grid-template-rows: 1fr 1fr;
    grid-template-areas:
            "a a"
            "b c"
}

.a {
    background: #000;
    grid-area: a;
}

.b {
    background: red;
    grid-area: b;
}

.c {
    background: cornflowerblue;
    grid-area: c;
}
```

```html
<div class="target">
    <div class="a"></div>
    <div class="b"></div>
    <div class="c"></div>
</div>
```



Не забувати, що при розподілі на сітку, дочірні блоки можуть мати розміри відмінні від ячейки сітки. Приклад. Є бабтьківський блок розміром 800*800 пікселів. Він розподілен на 10 фракцій вертикально та горизонтально. Відповідно кожна фракція має розмір 80*80. Але блок який знаходиться в середині може бути 10*10 пікселів.
Маніпулювати таким блоком можливо за допомоги justifiy-content & align-items і подібних характеристик, але зсовуватись такций блок буде лише в межах своєї фракції