CSS: Grid

AIT
Hi-tech team

НАШИ ПРАВИЛА

Включенная камера

Вопросы по поднятой руке

Не перебиваем друг друга

Все вопросы, не связанные с тематикой курса (орг-вопросы и т. д.), должны быть направлены куратору

Подготовьте свое рабочее окружение для возможной демонстрации экрана (закройте лишние соцсети и прочие приложения)

Повторим;)

Какой тип для элемента input применяется по умолчанию

Как в input задать значение в поле по умолчанию

Чем textarea отличается от обычного input

Как в select добавить варианты выбора

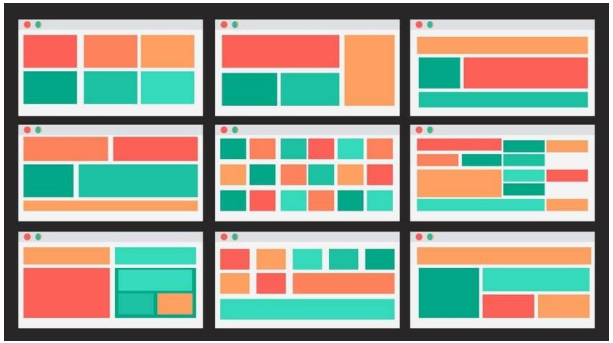
ЦЕЛЬ

Изучить концепцию Grid. Узнать какие свойства отвечают за выравнивание элементов.

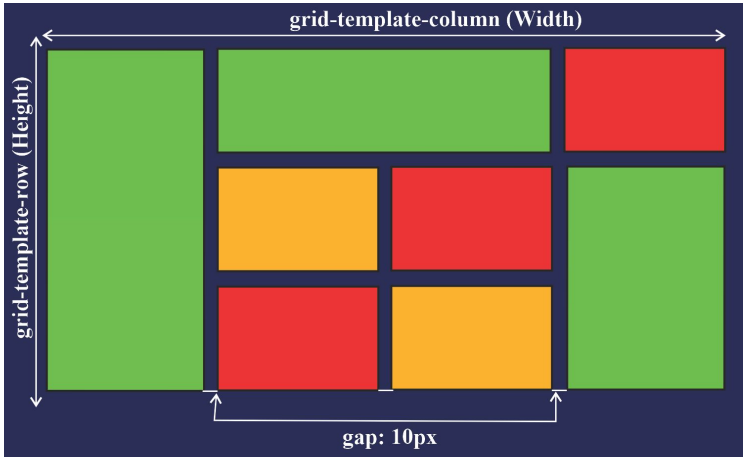
ПЛАН ЗАНЯТИЯ

1. Определение grid
2. Свойства grid контейнера
3. Свойства grid элементов

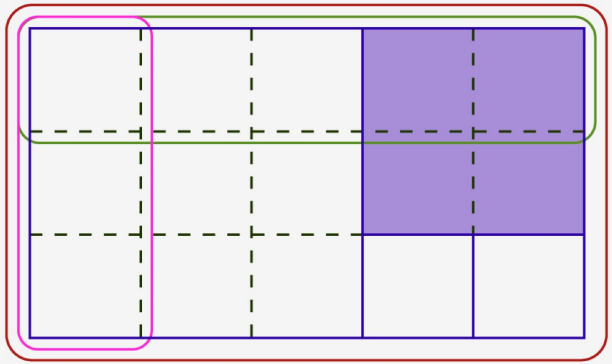
Grid



Grid (сетка) — это вид разметки, в котором элементы на сайте расположены в виде таблицы. Главная особенность этой таблицы в гибкости — можно объединять отдельные ячейки, менять размеры строк и столбцов, регулировать отступы между ними. А ещё гриды хорошо приспосабливаются к разным размерам экрана, что делает их адаптивными.



Структура Grid-разметки



— grid-ячейка

— grid-линия

— grid-элемент

— grid-контейнер

— grid-область

— grid-ряд

— grid-колонка

grid-контейнер — самый главный элемент во всей разметке, в нём хранится всё содержимое сетки;

grid-ячейка — единица грид-сетки, сюда можно положить один или несколько блоков кода;

grid-линия — горизонтальная или вертикальная линия, разделяющая столбцы и колонки;

grid-строка (row) — ряд ячеек;

grid-столбец (column) — колонка ячеек;

grid-элемент — какой-либо элемент сайта;

grid-область (area) — пространство из ячеек, в CSS можно объединить несколько ячеек в одну и работать с ними как с единым целым.

Грид-контейнер

```
    <div class="container">
        <div class="A">A</div>
        <div class="B">B</div>
        <div class="C">C</div>
        <div class="D">D</div>
        <div class="E">E</div>
    </div>
</body>
```

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

Свойства Grid контейнера

Свойства грид-контейнера - **grid-template-columns**

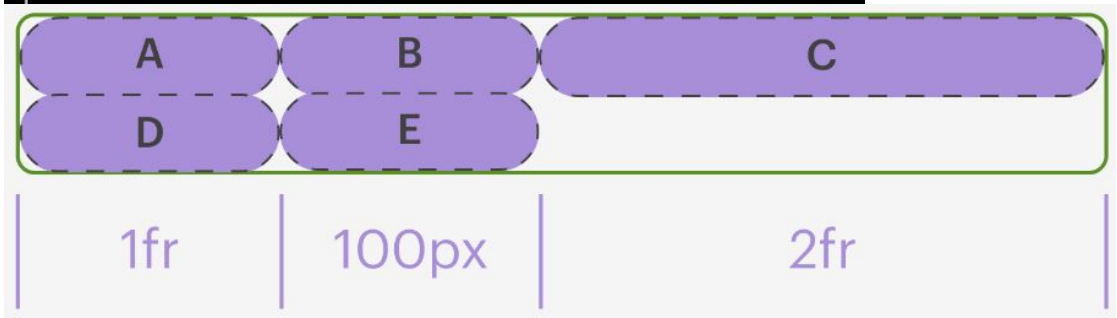
Создаём три колонки: первая занимает одну треть экрана, вторая — ровно 100px, а третья — всё оставшееся место

```
.container {
    display: grid;
    grid-template-columns: 30% 100px 70%;
}
```

Свойства грид-контейнера - **grid-template-columns**

Чтобы избегать ситуаций с переполнением, гридам придумали новую единицу измерения — фракцию (fr). Она позволяет разделить свободное пространство экрана на несколько частей

```
.container {
    display: grid;
    grid-template-columns: 1fr 100px 2fr;
}
```



Свойства грид-контейнера - **grid-template-rows**

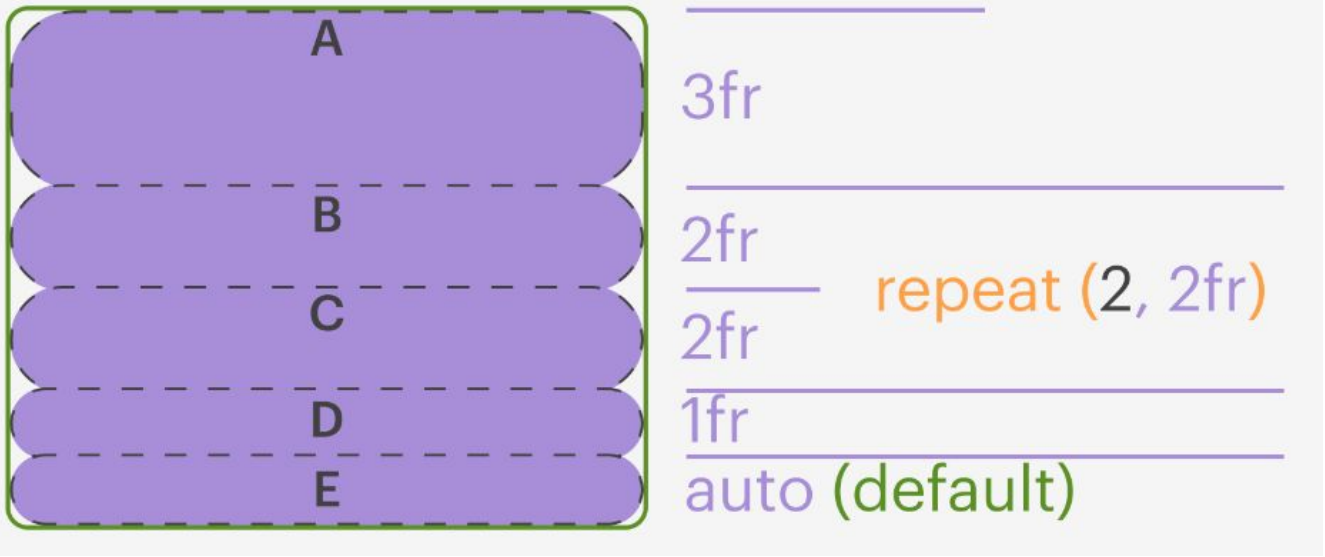
Свойство **grid-template-rows** задаёт таблице размеры строк

```
.container {
    display: grid;
    grid-template-rows: 3fr 2fr 2fr 1fr;
}
```

Чтобы избежать повторений можно записать одинаковые значения в функцию repeat

```
grid-template-rows: 3fr repeat(2, 2fr) 1fr;
```

Свойства грид-контейнера - **grid-template-rows**



Свойства грид-контейнера - **grid-template**

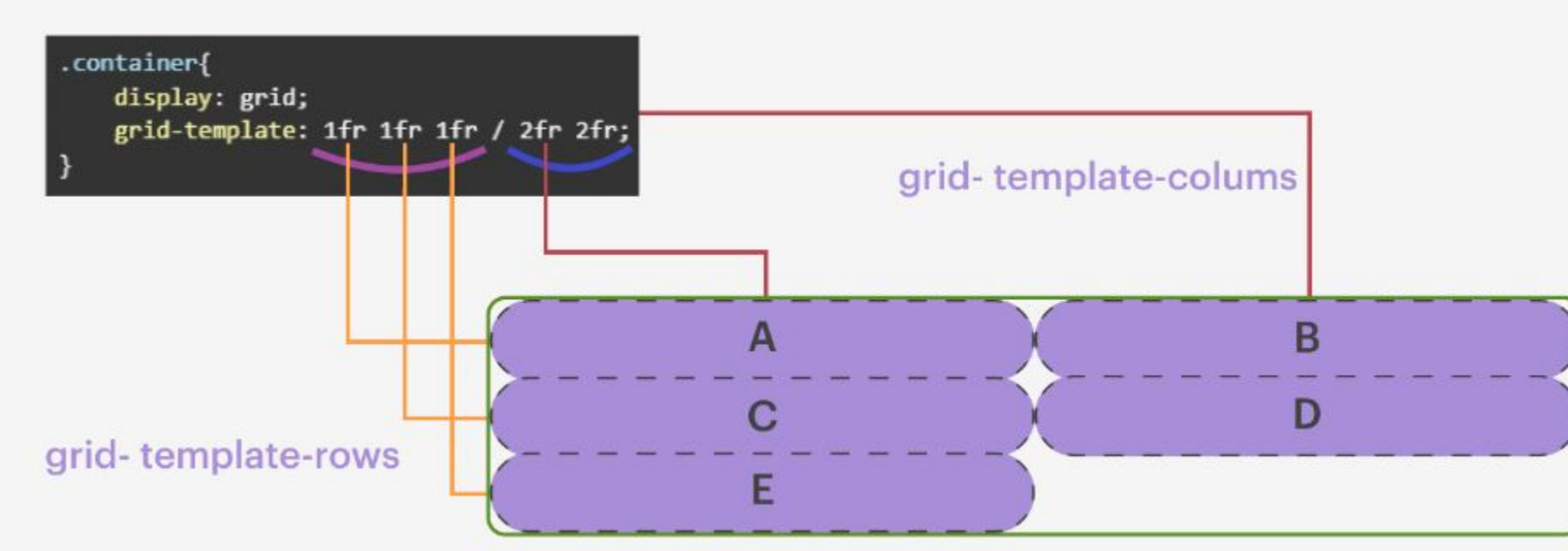
Свойство grid-template позволяет в краткой форме определить или столбцы со строками, или целые области.

```
/* Сетка из строк и столбцов */
grid-template: grid-template-rows / grid-template-columns;
```

Мы всего одной строчкой создаём целую таблицу — просто записываем через слеш количество рядов и столбцов

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

Свойства грид-контейнера - **grid-template**



Свойства грид-контейнера - **justify-items**

Выравнивает элементы в пределах области по горизонтали.

```
justify-items: stretch
```

Растягивает элементы на всю ширину грид-ячейки

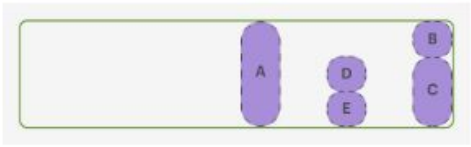
```
justify-items: start;
```

Располагает элементы в левой части грид-ячейки или области

```
justify-items: center;
```

Располагает элементы в центре грид-ячейки или области

```
justify-items: end;
```



Располагает элементы в правой части грид-ячейки или области

Свойства грид-контейнера **- align-items**

Выравнивает элементы в пределах области по вертикали.

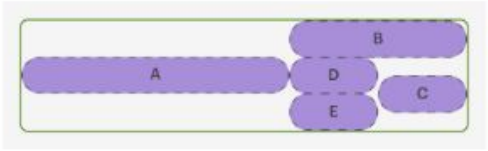
```
align-items: stretch;
```

Растягивает элементы на всю длину грид-ячейки

```
align-items: start;
```

Располагает элементы в верхней части грид-ячейки

```
align-items: center;
```



Располагает элементы в центре грид-ячейки

```
align-items: end;
```

Располагает элементы в нижней части грид-ячейки

Свойства грид-контейнера - **grid-template-areas**

Одна из особенностей грид-разметки — возможность создавать области и гибко регулировать их размеры.

Объявим для каждого элемента в CSS-файле свойство **grid-area** — его параметром будет любое имя, какое захотите. Можно просто добавить в любое место CSS-файла этот код:

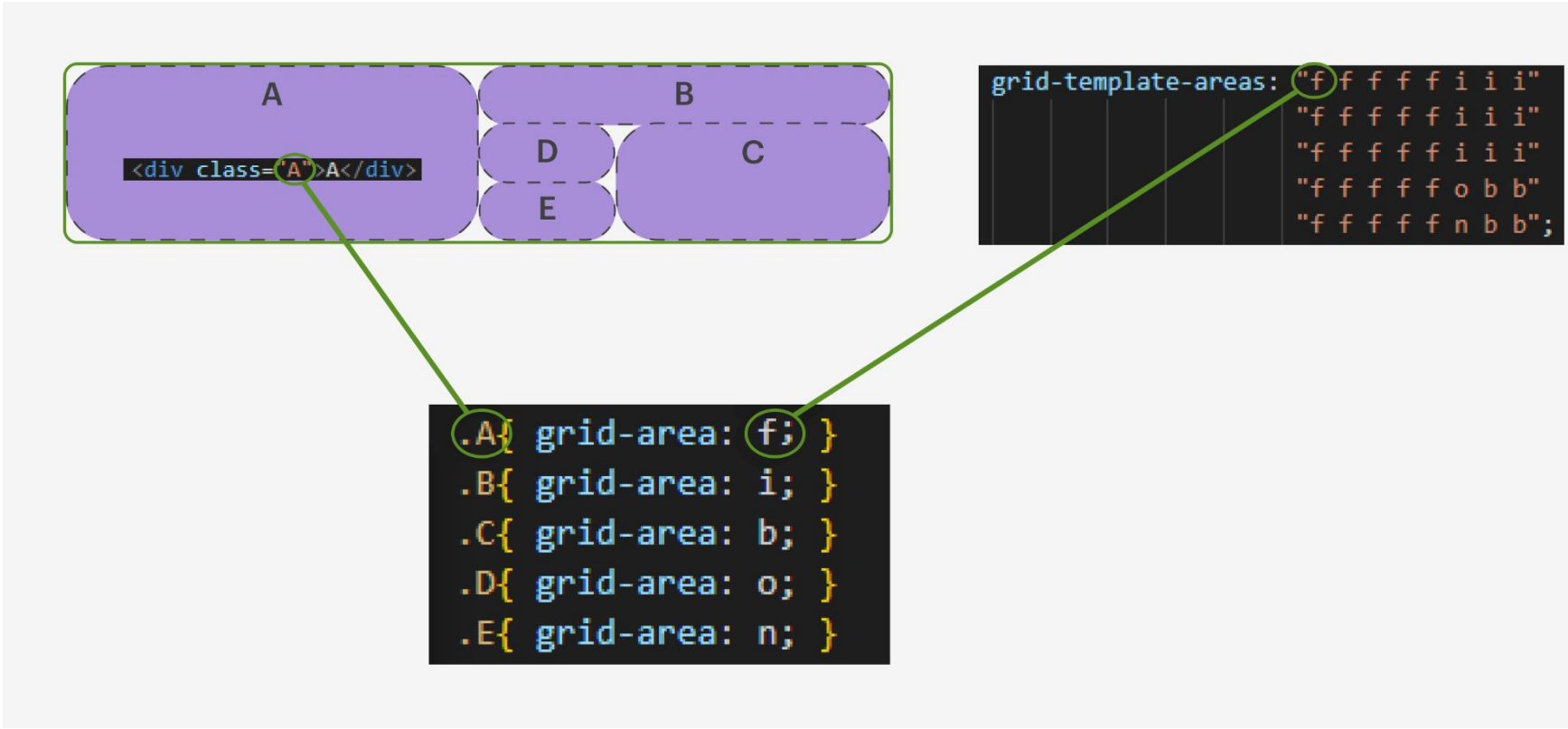
```
.A { grid-area: f; }

.B { grid-area: i; }

.C { grid-area: b; }

.D { grid-area: o; }

.E { grid-area: n; }
```

Свойства грид-контейнера - **grid-template-areas**

Затем в грид-контейнере создаём «матрицу» из этих имён:

```
.container {
    display: grid;
    grid-template-areas: "f f f f f i i i"
                         "f f f f f i i i"
                         "f f f f f i i i"
                         "f f f f f o b b"
                         "f f f f f n b b";
}
```

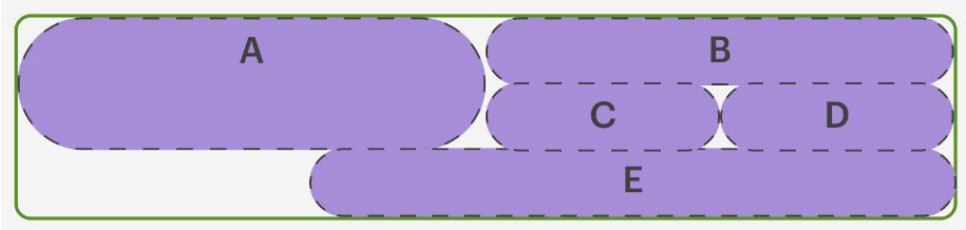
Свойства грид-контейнера - **grid-template-areas**



Свойства грид-контейнера - **grid-template**

```
.container {
    display: grid;
    grid-template: "f f f i i"
                   "f f f i i"
                   "f f f b o"
                   ". . n n n";
}

.A { grid-area: f; }
.B { grid-area: i; }
.C { grid-area: b; }
.D { grid-area: o; }
.E { grid-area: n; }
```

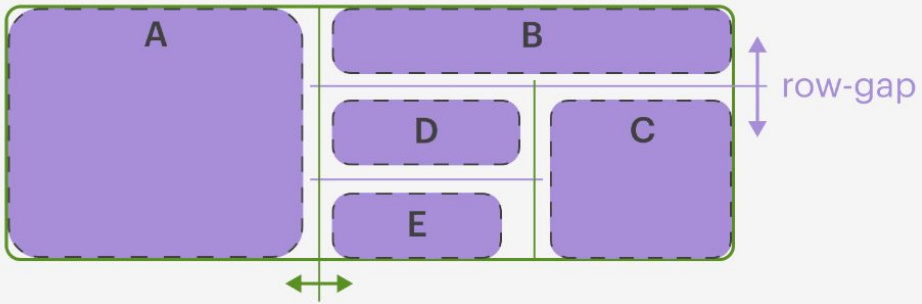
Короткая форма для грид-областей.



Свойства грид-контейнера - **grid-gap**

column-gap — расстояние между колонками;
row-gap — расстояние между строками.

```
.container {
    column-gap: 1%;
    row-gap: 1%;
}
```



gap: row-gap column-gap

```
gap: 1% 1%;
```

Свойства Grid элемента

Свойства грид-элементов - **grid-column**

С помощью этих свойств можно прямо указать браузеру, где должны находиться элементы. Для этого нужно указать две точки: с какой линии элемент начинается и какой заканчивается — это работает как по горизонтали, так и по вертикали.

```
grid-column-start: 1;
```

Задаёт начальную позицию (линию) столбцов

```
grid-column-end: 3;
```

Задаёт конечную позицию (линию) столбцов

```
grid-column: grid-column-start / grid-column-end;
```

Объединённое свойство грид-линий столбцов: начало и конец

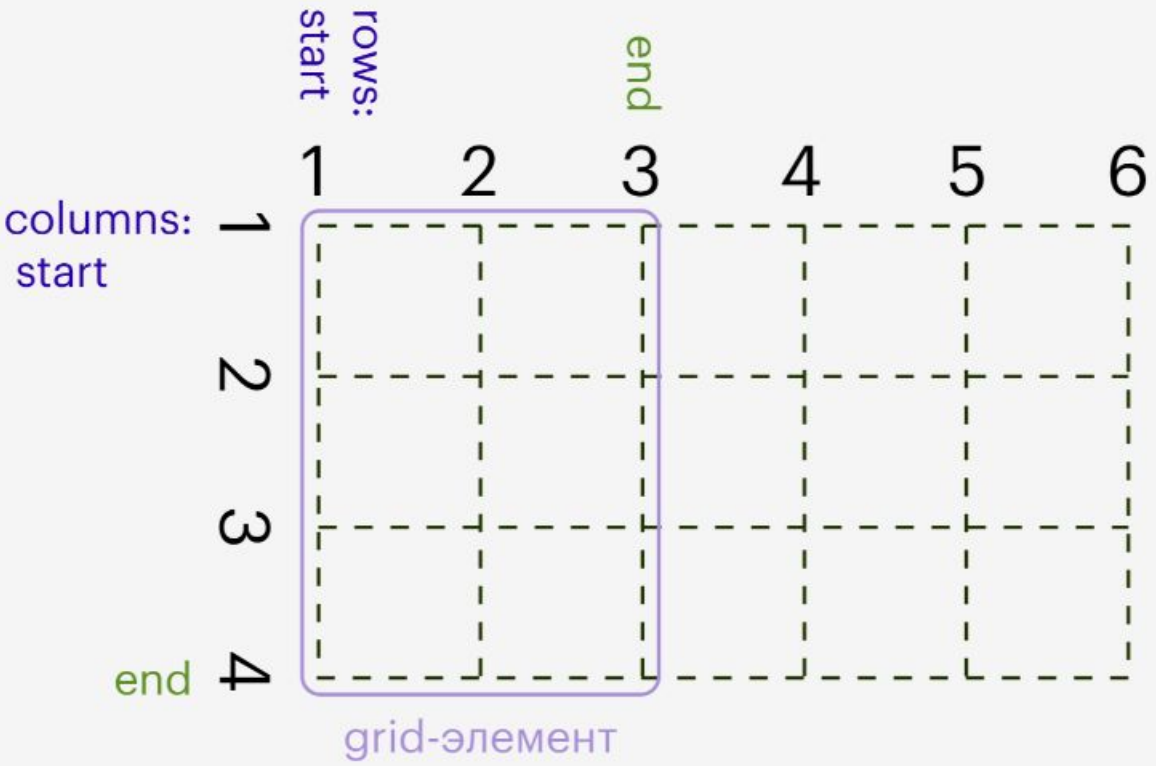
Свойства грид-элементов - **grid-row**

Вертикальная позиция

`grid-row-start: 1;`	Задаёт начальную позицию (линию) строк
`grid-row-end: 4;`	Задаёт конечную позицию (линию) строк
`grid-row: grid-row-start / grid-row-end;`	Объединённое свойство грид-линий строк: начало и конец

Общее свойство

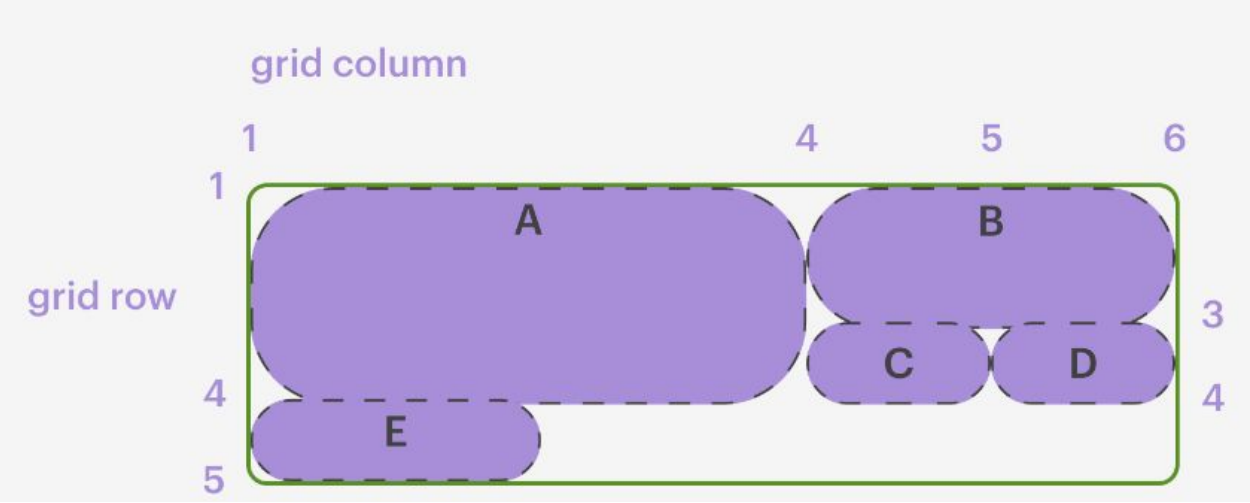
```
grid-area: grid-row-start / grid-column-start / grid-row-end / grid-column-end;
```


rows:
start
end
1
2
3
4
5
6
columns:
start
1
2
3
4
end
grid-элемент

```css
/* Объявляем грид 5 × 3 */
.container {
    display: grid;
    grid-template: repeat(3, 1fr) / repeat(5, 1fr) ;
}

/* Расставляем первые три элемента: A, B и C */
.A {
    grid-area: 1 / 1 / 4 / 4;
}

.B {
    grid-column-start: 4;
    grid-column-end: 6;
    grid-row-start: 1;
    grid-row-end: 3;
}

.C {
    grid-column: 4 / 5;
    grid-row: 3 / 4;
}
```

Этот код указывает нашим буквам, какие позиции занять в сетке:

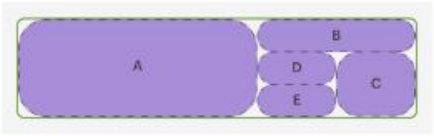
- Элемент A занял область от первой до четвёртой линии — и по горизонтали, и по вертикали.
- Элемент B занял область от четвёртой до шестой линии по горизонтали и от первой до третьей по вертикали.
- Элемент C занял область от четвёртой до пятой линии по горизонтали и от третьей до четвёртой линии по вертикали.
- Ну а остальные расположились по дефолту



Выравнивание грид-элементов - **justify-self**

Выравнивает грид-элементы по горизонтали.

```
justify-self: stretch;
```



Элемент A растянулся по всей длине грид-области

```
justify-self: center;
```

Элемент A расположился в центре грид-области

```
justify-self: start;
```

Элемент A расположился в начале грид-области

```
justify-self: end;
```

Элемент A расположился в конце грид-области

Выравнивание грид-элементов - **align-self**

Выравнивает грид-элементы по вертикали

```
align-self: stretch;
```

Элемент A растянулся по всей ширине грид-области

```
align-self: center;
```

Элемент A расположился в середине грид-области

```
align-self: start;
```

Элемент A расположился в начале грид-области

```
align-self: end;
```

Элемент A расположился в конце грид-области

Ваша новая IT-профессия – Ваш новый уровень жизни

Программирование с нуля в
немецкой школе AIT TR GmbH

AIT
Hi-tech team