**Grid (сетка)** — это вид разметки, в котором элементы на сайте расположены в виде таблицы. Главная фишка 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.

|  |  |
| --- | --- |
| **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. |  |
| **display: grid** |  |
| **Колонки**  **grid-template-columns: 1fr 100px 2fr**-  первому столбцу отдаём одну часть свободного пространства (1fr), третьему — две (2fr), а тот, что посередине, пускай забирает свои 100px |  |
| **Ряды**  **grid-template-rows: 3fr repeat(2, 2fr) 1fr;**  Обратите внимание: высоту последней строчки E мы нигде не указывали, поэтому она остаётся дефолтной. |  |
| **Области (grid-template-areas)**  Одна из особенностей грид-разметки — возможность создавать области и гибко регулировать их размеры.  Чтобы создать область, 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(класс) { grid-area: f; }  .B { grid-area: i; }  .C { grid-area: b; }  .D { grid-area: o; }  .E { grid-area: n; }  Затем в грид-контейнере создаём «матрицу» из этих имён:  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"; |  |
| **Сокращённое свойство**  **(grid-template)**  Свойство grid-template позволяет в краткой форме определить или столбцы со строками, или целые области.  1. **Короткая форма для грид-строк и грид-колонок:**  grid-template: grid-template-rows / grid-template-columns;  **grid-template: repeat(3, 1fr) / repeat(2, 2fr);**  2. **Короткая форма для грид-областей.**  Также сокращённая запись поддерживает объявление грид-областей, нужно только объявить свойство grid-area для каждого объекта:  grid-template: "f f f i i"  "f f f i i"  "f f f b o"  ". . n n n"; . – пустое место  }  .A { grid-area: f; }  .B { grid-area: i; }  .C { grid-area: b; }  .D { grid-area: o; }  .E { grid-area: n; } |  |
| **Свойства грид-элементов**  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.  По горизонтали  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;  По вертикали  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;  **grid-area: grid-row-start / grid-column-start / grid-row-end / grid-column-end;**   |  |  | | --- | --- | | **Пример:**  **.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 А 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 С занял область от четвёртой до пятой линии по горизонтали и от третьей до четвёртой линии по вертикали. * Ну а остальные расположились по дефолту :) | |  |
| **Grid-gap (отступы)** Используя grid-gap, можно регулировать интервалы между элементами в контейнере. |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Сегодня мы узнали, что такое грид в CSS: как он работает, из чего состоит и как настраивается. Повторим ключевые моменты:**

**Grid Layout в CSS** — это гибкая разметка, при которой элементы располагаются в виде таблицы. Таблица состоит из ячеек, которые можно собирать в целые области.

**Чтобы создать грид**, нужно в файле с CSS-разметкой добавить строчку display: grid.

**Грид состоит** из родительского контейнера и вложенных в него элементов. Элементы могут занимать несколько ячеек, которые складываются в столбцы, колонки или области.

**Фракция** — единица измерения в CSS, созданная для удобства — чтобы можно было задавать соотношения между элементами и не подгонять вручную проценты.

**Грид может** содержать в себе другой грид — такая конструкция называется вложенным subgrid.

**Основными свойствами** грид-контейнера являются: grid-template, grid-template-columns, grid-template-rows, grid-template-areas. Они позволяют задавать размеры колонок, строк или целых областей.

**За отступы между** колонками грида отвечает свойство column-gap, а между рядами — row-gap.

**Горизонтально выровнять** элементы контейнера можно с помощью justify-items и justify-content, а вертикально — с align-items и align-content.

**Настроить расположение элементов** от определённой линии грид-сетки можно с помощью свойств:

по горизонтальной оси: grid-column-start, grid-column-end, grid-column;

по вертикальной оси: grid-row-start, grid-row-end, grid-row;

или общим свойством grid-area.

**Выровнять конкретный элемент** грида можно с помощью justify-self (по горизонтали) и align-self (по вертикали).