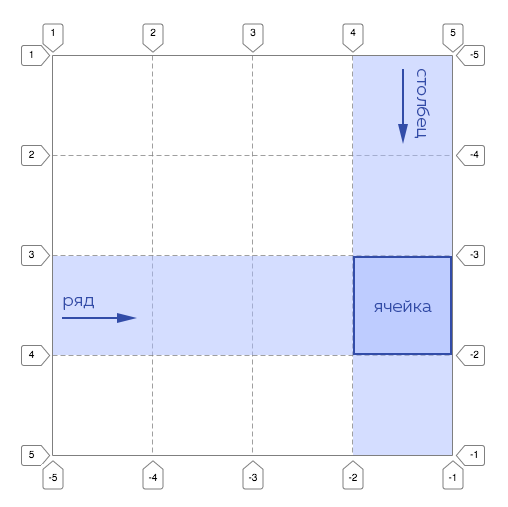
# «Гриды: расположение грид-элементов».

Чтобы сделать элемент грид-контейнером, нужно задать ему соответствующее значение свойства display:   
.container { display: grid; }

В гриде элементы располагаются по двумерной сетке. То есть грид состоит из рядов и столбцов, располагающихся между линий, которые нумеруются по порядку. Одно «деление» грида называют ячейкой.



Чтобы расположить элемент по сетке внутри грида, нужно задать ему координаты по столбцам и по рядам: с какой линии столбцов и рядов грид-область будет начинаться, на какой линии столбцов и рядов будет заканчиваться. Координаты грид-области на иллюстрации выше в коде описываются так:

/\*

Область

начинается с 4 линии столбцов,

заканчивается на 5 линии столбцов,

начинается на 3 линии рядов,

заканчивается на 4 линии рядов.

\*/

.element {

grid-column-start: 4;

grid-column-end: 5;

grid-row-start: 3;

grid-row-end: 4;

}

Координаты можно отсчитывать не только от начала, но и от конца грида. При этом к индексу линии, от которой ведётся отсчёт, добавляет знак «минус». Координаты той же грид-области можно описать следующим образом:

/\*

Область

начинается со 2 линии столбцов с конца грида,

заканчивается на 1 линии столбцов с конца грида,

начинается на 3 линии рядов с конца грида,

заканчивается на 2 линии рядов с конца грида.

\*/

.element {

grid-column-start: -2;

grid-column-end: -1;

grid-row-start: -3;

grid-row-end: -2;

}

Существует также сокращённый синтаксис для этих свойств. Свойство grid-column объединяет в себе сразу два свойства: grid-column-start/grid-column-end.

Пример:

grid-column: 1 / 3;

/\* Это то же самое, что: \*/

grid-column-start: 1;

grid-column-end: 3;

Аналогично, свойство grid-row — это сокращение для задания пары свойств: grid-row-start/grid-row-end.

Пример:

grid-row: 1 / -2;

/\* Это то же самое, что: \*/

grid-row-start: 1;

grid-row-end: -2;

Если в свойстве grid-row или grid-column не задать второй параметр, то значение останется валидным, но применится только первый параметр.

Грид-элементы могут наслаиваться друг на друга, при этом они начинают себя вести *как будто* абсолютно спозиционированные, при этом на них так же действует свойство z-index. Чем больше значение z-index, тем выше грид-элемент в «стопке».

# «Гриды: создание раскладки».

Чтобы задать гриду определённое количество столбцов и рядов, существуют свойства grid-template-columns и grid-template-rows.

Свойство grid-template-columns перечисляет количество и ширину будущих столбцов грида:

/\*

Задаём гриду три столбца,

первый шириной 100px,

второй шириной 200px,

третий — 300px.

\*/

.element {

grid-template-columns: 100px 200px 300px;

}

Аналогично grid-template-columns работает и свойство grid-template-rows, только оно сообщает гриду, сколько рядов он будет содержать и какой они будут высоты:

/\*

Задаём гриду три ряда,

первый высотой 100px,

второй высотой 200px,

третий — 300px.

\*/

.element {

grid-template-rows: 100px 200px 300px;

}

Также есть возможность задавать нефиксированный размер ячейкам. Для этого существует значение auto:

/\*

Задаём гриду два столбца,

первый с нефиксированной шириной,

второй шириной 100px.

\*/

.element {

grid-template-columns: auto 100px;

}

/\*

Задаём гриду три ряда,

первый высотой 100px,

второй с нефиксированной высотой,

третий высотой 200px.

\*/

.element {

grid-template-rows: 100px auto 200px; }

При заданных свойствах grid-template-columns и grid-template-rows грид-элементы вписываются в заданную сетку автоматически. При этом часть грид-элементов также может иметь чёткие координаты в гриде. Комбинируя задание явного расположения грид-элементов и их автоматическое распределение, можно строить сложные и одновременно гибкие сетки.

Ещё один механизм создания раскладки грида заключается в использовании свойств grid-template-areas и grid-area. В значении свойства grid-template-areas визуально «по клеточкам» описывается структура грида.

Пример:

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

HTML:

<div class="grid-container">

<div class="grid-element-1"></div>

<div class="grid-element-2"></div>

<div class="grid-element-3"></div>

</div>

CSS:

.grid-container {

grid-template-areas:

"red yellow green"

"red yellow green"

"red yellow green";

}

.grid-element-1 {

grid-area: red;

}

.grid-element-2 {

grid-area: yellow;

}

.grid-element-3 {

grid-area: green;

}

Свойство grid-template-areas позволяет некоторые ячейки грида помечать как пустые. Для этого вместо буквенного именования области используется символ точки .

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

.grid-container {

grid-template-areas:

". . green"

"red yellow green"

"red yellow green";

}

Свойство gap позволяет добавлять равномерный интервал между рядами и столбцами. Чтобы добавить интервал только между рядами, используется свойство row-gap, а только между столбцами — column-gap.

.grid-container {

gap: 10px; /\* Между рядами и столбцами интервал 10px \*/

column-gap: 20px; /\* Между столбцами интервал 20px \*/

row-gap: 30px; /\* Между рядами интервал 30px \*/}

# Ресурсы:

1. Гриды: <https://tpverstak.ru/grid/>
2. Гриды: <https://doka.guide/css/grid-guide/>
3. Флексы: <https://doka.guide/css/flexbox-guide/>
4. Флексы: <https://tpverstak.ru/flex-cheatsheet/>
5. БЭМ: <https://tpverstak.ru/common-css-class-names/>
6. БЭМ: <https://bem-cheat-sheet.9elements.com/>