Теория

Модуль CSS Grid Layout - система двумерного макета, оптимизированного для дизайна пользовательского интерфейса. Главная идея, лежащая в основе макета сетки, заключается в разделении веб-страницы на столбцы и строки. В образовавшиеся области сетки можно помещать элементы сетки, а управлять их размерами и расположением можно с помощью специальных свойств модуля.

Кроме того, благодаря своей способности явно размещать элементы в сетке, Grid Layout позволяет кардинально преобразовывать структуру визуального макета (отображаемого на экране), не требуя соответствующих изменений разметки.

Хотя многие макеты могут быть отображены с помощью Grid или Flexbox, у каждого есть свои особенности. Grid обеспечивает двухмерное выравнивание, использует нисходящий подход к макету, допускает явное перекрытие элементов и обладает более мощными связующими возможностями. Flexbox фокусируется на распределении пространства по оси, использует более простой восходящий подход к макету, может использовать систему переноса строк на основе размера контента для управления своей вторичной осью и опирается на базовую иерархию разметки для построения более сложных макетов.

Концепция сетки и основные понятия

Сетка (grid) представляет собой набор пересекающихся горизонтальных и вертикальных линий, делящих пространство grid-контейнера на области сетки, в которые могут быть помещены элементы сетки.

Линии сетки (grid lines) — это невидимые горизонтальные и вертикальные разделительные линии, они существуют по обе стороны от строки и столбца. На них можно ссылаться по числовому индексу (используя свойства `grid-column-start`, `grid-column-end`, `grid-row-start` и `grid-row-end`) или имени, заданному в CSS-коде. Числовые индексы сетки зависят от стиля языка, поэтому первым столбцом может быть как самый левый, так и самый правый столбец.

Выделяют две группы линий сетки: одна группа определяет столбцы, которые проходят вдоль оси блока (ось столбцов), и перпендикулярная группа, определяющая строки, простирающиеся вдоль линейной оси.

Дорожка сетки (grid track) — пространство между двумя соседними линиями сетки, используется для определения либо столбца, либо строки сетки. Дорожка идет от одного края контейнера к другому, размер зависит от расположения линий сетки, которые ее определяют. Дорожки сетки аналогичны столбцам и строкам таблицы. По умолчанию

смежные дорожки плотно прилегают друг к другу, задать расстояние между ними можно с помощью свойств `row-gap`, `column-gap` и `gap`.

Ячейка сетки (grid cell) — пространство, ограниченное четырьмя линиями сетки, аналогично ячейке таблицы. Ячейка сетки — это область, в которой можно разместить контент. Это наименьшая единица сетки, на которую можно ссылаться при позиционировании элементов сетки. К ячейкам сетки нельзя обращаться напрямую с помощью CSS-свойств.

Область сетки (grid area) — прямоугольная область, ограниченная четырьмя линиями сетки и состоящая из одной или нескольких соседних ячеек. Область может быть такой же маленькой, как одна ячейка, или такой же большой, как все ячейки сетки. Область сетки может быть задана явно с помощью свойства `grid-template-areas`, по умолчанию на нее ссылаются ограничивающие линии сетки.

Элементы сетки (grid items) — отдельные элементы, которые назначаются области сетки (или ячейке сетки). Каждый контейнер-сетка включает ноль и более элементов сетки; каждый дочерний элемент контейнера-сетки автоматически становится элементом сетки. Дорожки, ячейки и области сетки построены из линий сетки. Тем не менее не требуется, чтобы все области сетки были заполнены элементами, вполне возможно, что некоторые или даже большинство ячеек сетки будут пустыми от любого содержимого. Также возможно, что элементы сетки будут перекрывать друг друга, либо определять перекрывающиеся области сетки.

Контейнер-сетка

Для создания макета на основе сетки необходимо определить контейнер-сетку.

Контейнер-сетка (grid container) — это блок, который устанавливает контекст форматирования по типу сетки, то есть создает область с сеткой, а дочерние элементы располагаются в соответствии с правилами компоновки сетки, а не блочной компоновки. Когда вы определяете контейнер сетки с помощью `display: grid` или `display: inline-grid`, вы создаете новый контекст форматирования для содержимого этого контейнера, который влияет только на дочерние элементы сетки.

Контейнер-сетка бывает двух видов: обычный `display: grid` и встроенный `display: inline-grid`. Первый генерирует grid-контейнер уровня блока, второй — grid-контейнер уровня строки. Контейнеры-сетки не являются блочными контейнерами, поэтому некоторые CSS-свойства не работают в контексте макета сетки:

- `float` и `clear` игнорируются элементами сетки (но не самим контейнером-сеткой).
- `vertical-align` не влияет на элементы сетки.
- Псевдоэлементы `::first-line` и `::first-letter` не применяются к контейнеру-сетке и его потомкам.
- Если контейнер-сетка является контейнером уровня строки `display: inline-grid` и для него заданы обтекание или абсолютное позиционирование, то вычисляемое значение свойства `display` будет `grid`.

Определение сетки

Когда вы создаете контейнер-сетку, сетка по умолчанию имеет один столбец и одну строку, которые занимают полный размер контейнера. Для разделения контейнера-сетки на столбцы и строки используются свойства `grid-template-columns`, `grid-template-rows` и `grid-template-areas`. С помощью этих свойств можно определить сетку явно.

Окончательная сетка может оказаться больше из-за элементов сетки, размещенных вне явной сетки; в этом случае будут созданы неявные дорожки, размер этих неявных дорожек будет определяться свойствами `grid-auto-rows` и `grid-auto-columns`.
Свойства `grid` и `grid-template` — это сокращенные обозначения, которые можно использовать для одновременной установки всех трех явных свойств сетки `grid-template-columns`, `grid-template-rows` и `grid-template-areas`. `grid` сбрасывает свойства, управляющие неявной сеткой, тогда как свойство `grid-template` оставляет их без изменений.

Строки и столбцы

Количество строк / столбцов определяется с помощью свойств `grid-template-rows` и `grid-template-columns`. Свойства не наследуются.

grid-template-rows, grid-template-columns

Значения:

`none`	Указывает, что свойство не создает явных дорожек сетки (хотя явные дорожки сетки все еще могут создаваться свойством `grid-template-areas`). При отсутствии явной сетки любые строки/столбцы будут генерироваться неявно, а их размер будет определяться свойствами `grid-auto-rows` и `grid-auto-columns`. Значение по умолчанию.
список дорожек / автоматический список дорожек	Устанавливает список дорожек в виде последовательности функций определения размера дорожек и названий линий сетки. Каждая функция определения размера дорожки может быть задана в единицах длины, как процент от размера контейнера-сетки или доля свободного пространства в сетке. Размер также может быть указан как диапазон с помощью нотации `minmax()`.

Относительные, абсолютные единицы и процентные значения для определения дорожек сетки (длина)

Размеры дорожек сетки можно задавать с помощью положительных значений, используя относительные единицы длины — например, `em`, `vh`, `vw`; абсолютные единицы длины — `px`; и проценты `%`. Размеры в `%` вычисляются от ширины или высоты контейнера-сетки.

Синтаксис

```
.grid-container {
  display: grid;
  grid-template-rows:  5em 200px 200px; /* 3 строки */
  grid-template-columns: 200px 5em 50%; /* 3 столбца */
}
CSS
```

Гибкие размеры дорожек: единица измерения fr

`fr` — единица длины, которая позволяет создавать гибкие дорожки. Не является единицей измерения в обычном ее понимании, поэтому не может быть представлена или объединена с другими типами единиц в выражениях `calc()`. Общий размер фиксированных строк или столбцов вычитается из доступного пространства контейнера-сетки. Оставшееся пространство делится между строками и столбцами с гибкими размерами пропорционально их коэффициенту, например:

Синтаксис

```
.grid-container {
  display: grid;
  grid-template-columns: 1fr 1fr 1fr 1fr; /* эквивалентно grid-template-columns: 25% 25% 25% 25%; */
}
.grid-container {
  display: grid;
  grid-template-columns: 1fr 2fr 1fr; /* эквивалентно grid-template-columns: 25% 50% 25%; */
}
```

CSS

Если сумма всех гибких размеров дорожек меньше `1`, они будут занимать только соответствующую долю оставшегося пространства, а не расширяться, чтобы заполнить его полностью.

Если доступное пространство бесконечно (то есть, ширина или высота контейнера-сетки не заданы), дорожки сетки гибкого размера масштабируются по своему содержимому, сохраняя при этом их соответствующие пропорции.

Минимальные и максимальные размеры дорожек

Ключевое слово `max-content` устанавливает для дорожки размер, который занимает максимально необходимое пространство с учетом содержимого элемента сетки.

`min-content` позволяет занимать минимальное пространство, необходимое для этого содержимого, при этом ширина элемента ориентируется на самое длинное слово или на самое широкое изображение.

Функция `minmax(min,max)` определяет диапазон размеров, больше или равный `min` и меньше или равный `max`. Если `max < min`, то `max` игнорируется, а `minmax(min,max)` обрабатывается как `min`. Значения в `fr` можно устанавливать только как максимальное.

Варианты

```
minmax(длина или min-content или max-content или auto, длина или fr или min-content или max-content или auto)
minmax(длина, длина или fr или min-content или max-content или auto)
minmax(длина или min-content или max-content или auto, длина)
```

CSS

Синтаксис

```
.grid-container {
  display: grid;
  grid-template-rows: 200px minmax(100px, 1fr);
}
```

CSS

Автоматические размеры

Значение `auto` ориентируется на содержимое элементов сетки одной дорожки. Как минимум, рассматривается как минимальный размер элемента сетки, как определено `min-width` или `min-height`. Как максимум, обрабатывается так же, как и `max-content`. Может растягиваться за счет свойств `align-content` и `justify-content`.

Синтаксис

```
.grid-container {
  display: grid;
  grid-template-rows: auto 1fr;
  grid-template-columns: auto 1fr auto;
}
```

Соответствие содержимому

Размеры дорожек можно задавать с помощью значения `fit-content(длина или %)`, представляющее собой формулу `min(maximum size, max(minimum size, argument))`, которая вычисляется как `minmax(auto, max-content)`, то есть `auto`. При этом, размер дорожки ограничивается значением, указанным в скобках, и если оно больше, чем автоматический минимум.

Синтаксис

```
.grid-container {
  display: grid;
  grid-template-columns: fit-content(50%) fit-content(300px) 1fr;
}
```

Повтор строк и столбцов

Нотация `repeat()` представляет повторяющийся фрагмент списка дорожек, что позволяет записать в более компактной форме большое количество одинаковых по размерам столбцов или строк. Общая форма синтаксиса следующая:

```
repeat(число или auto-fill или auto-fit, повторяющаяся дорожка)
CSS
```

Синтаксис

```
.grid-container {
  display: grid;
  grid-template-rows: repeat(3, 200px);
}
```

Первый аргумент задает количество повторений, которое может быть задано с помощью положительного целого числа или ключевых слов. Второй аргумент - размер повторяющейся дорожки. Однако, существуют некоторые ограничения:

- Нотация `repeat()` не может быть вложенной.
- Значения `auto-fill` или `auto-fit` не могут быть совмещены с `min-content`, `max-content`, `auto`, `fit-content()` или `fr`.

Синтаксис `repeat()` имеет несколько форм:

```
/*повтор дорожки*/
repeat(количество повторений, имя дорожки? размер дорожки + имя дорожки?)
/*автозаполнение дорожек сетки*/
repeat(auto-fill или auto-fit, имя дорожки? фиксированный размер дорожки + имя дорожки?)
/*фиксированный повтор дорожки*/
```

```
repeat(количество повторений, имя дорожки? фиксированный размер дорожки + имя дорожки?)
```
CSS

Используя значение `auto-fill`, вы всегда получите хотя бы один столбец, даже если он по какой-то причине не помещается в контейнер-сетку. Если вы используете `auto-fit`, то дорожки, которые не содержат элементы сетки, будут сброшены.

Именованные области

Свойство `grid-template-areas` определяет именованные области сетки, которые не связаны с каким-либо конкретным элементом сетки, но на которые можно ссылаться из свойств размещения сетки. Синтаксис свойства обеспечивает визуализацию структуры сетки, облегчая понимание общего макета контейнера-сетки. Свойство не наследуется.

grid-template-areas

Значения:

`none`	Указывает, что никакие именованные области сетки и никакие явные дорожки сетки не определены этим свойством (хотя явные дорожки сетки все еще могут быть созданы с помощью `grid-template-columns` или `grid-template-rows`). При отсутствии явной сетки любые строки/столбцы будут генерироваться неявно, а их размер будет определяться свойствами `grid-auto-rows` и `grid-auto-columns`. Значение по умолчанию.
строка +	Последовательность идентификаторов, определяющая, как должны отображаться строки и столбцы.

Синтаксис

```
.grid-container {
  display: grid;
  grid-template-areas: "header header"
                       "sidebar content"
                       "sidebar content";
  grid-template-columns: 150px 1fr;
  grid-template-rows: 50px 1fr 50px;
}
header {
  grid-area: header;
}
aside {
  grid-area: sidebar;
}
main {
  grid-area: content;
}
```
CSS

Каждый идентификатор сетки в значении `grid-template-areas` соответствует ячейке сетки. Как только все ячейки идентифицированы, браузер объединяет все смежные ячейки с одинаковыми именами в одну область, которая охватывает все их, при условии, что они описывают область прямоугольной формы. Если вы попытаетесь настроить более сложные области, весь шаблон будет недействительным и области сетки не будут определены.
Все строки должны содержать одинаковое количество столбцов. Если вы хотите определить только некоторые ячейки как часть области сетки, вы можете использовать одну или несколько `.` для заполнения этих безымянных ячеек. При определении областей сетки идентификаторы можно перечислять через единичный пробел, без разрыва строки. Или же выровнять с помощью пробелов/табуляции и перевода строки для большей наглядности.
Области сетки полезны для определения семантических отношений между различными частями макета страницы, позволяя указать, какая часть страницы включает в себя верхний колонтитул, боковую панель, область содержимого и нижний колонтитул.
После того, как вы создали области сетки, элементы сетки могут быть назначены непосредственно, чтобы занимать эти области, используя свойство `grid-area`.

Неявная сетка

Автоматические дорожки сетки

Если элемент сетки расположен в строке или столбце, размер которых не определен явно `grid-template-rows` или `grid-template-columns`, создаются неявные дорожки сетки для его хранения. Это может произойти в случае, если строка или столбец оказались за пределами установленных размеров сетки.
По умолчанию эти автоматически добавляемые дорожки имеют минимальный необходимый размер. Свойства `grid-auto-rows` и `grid-auto-columns` позволяют контролировать размер неявных дорожек сетки. Если дано несколько размеров дорожек, шаблон повторяется по мере необходимости, чтобы найти размер неявных дорожек. Первая неявная дорожка сетки после явной сетки получает первый заданный размер и так далее. Свойства не наследуются.

grid-auto-columns, grid-auto-rows	
Значения:	
`auto`	Значение по умолчанию.
размер дорожки +	В качестве размера дорожки может использоваться любое значение, допустимое для задания размеров дорожек сетки.

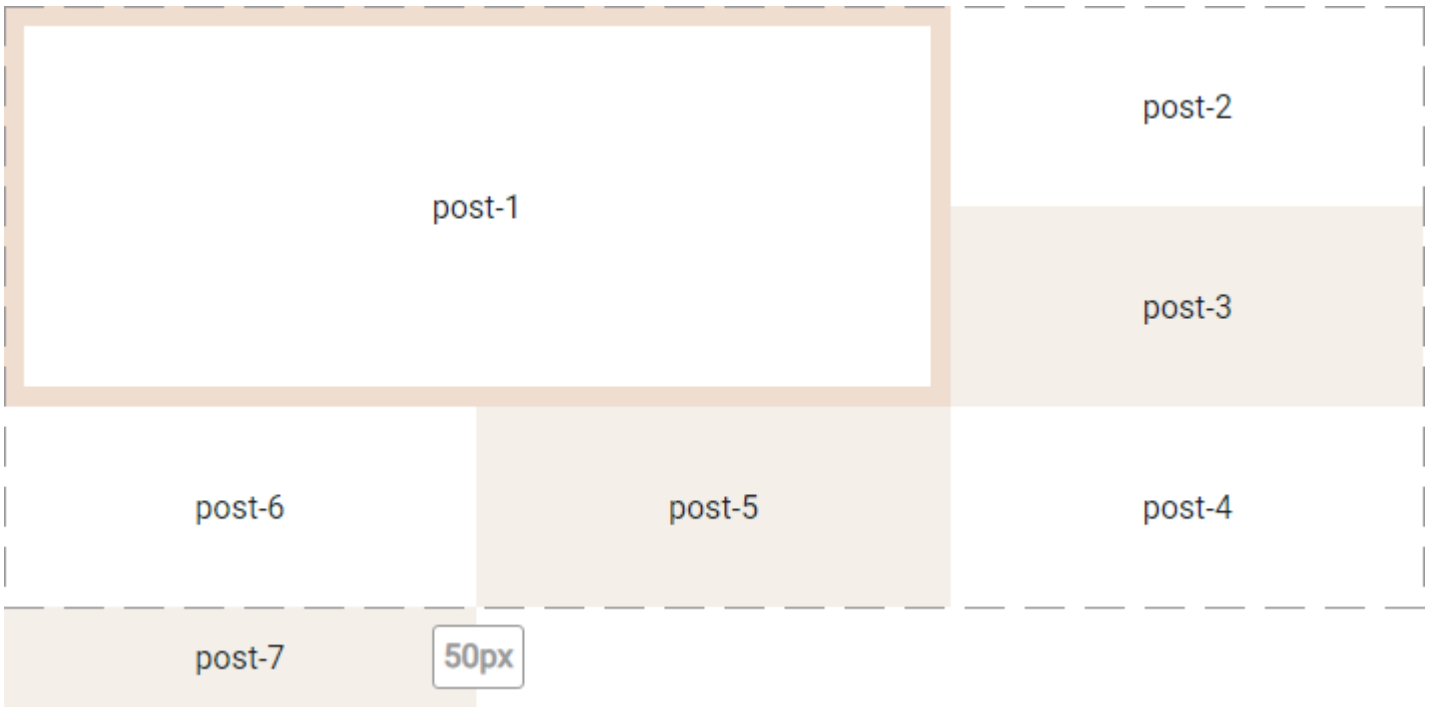


РИС. 3. АВТОМАТИЧЕСКИЕ ДОРОЖКИ СЕТКИ

```
.grid-container {
  max-width: 710px;
  display: grid;
  grid-template-columns: repeat(3,1fr);
  grid-template-rows: repeat(3,100px);
  grid-auto-rows: 50px;
}
.post-1 {
  grid-column: 1/3;
  grid-row: 1/3;
}
.post-2 {
  grid-column: 3;
  grid-row: 1;
}
.post-3 {
  grid-column: 3;
  grid-row: 2;
}
.post-4 {
  grid-column: 3;
  grid-row: 3;
}
.post-5 {
  grid-column: 2;
  grid-row: 3;
}
.post-6 {
  grid-column: 1;
  grid-row: 3;
}
```

Автоматическое размещение
Элементы сетки, которые не размещены явно, автоматически помещаются в незанятое пространство в контейнере-сетке с помощью алгоритма автоматического размещения. Свойство `grid-auto-flow` управляет автоматическим размещением элементов сетки без явного положения. После заполнения явной сетки (или если явной сетки нет) автоматическое размещение также приведет к генерации неявных дорожек сетки. Свойство не наследуется.

grid-auto-flow	
Значения:	
`row`	Алгоритм автоматического размещения размещает элементы, заполняя каждую строку по очереди слева-направо (для LTR-языков), добавляя новые строки по мере необходимости. Значение по умолчанию.
`column`	Алгоритм размещает элементы, заполняя каждый столбец по очереди сверху-вниз, добавляя новые столбцы по мере необходимости.
`dense`	Алгоритм "плотной" укладки элементов. При необходимости может менять порядок следования элементов, заполняя пустые места более крупными элементами.

Свойство будет полезным при создании компактных галерей, если для изображений не задан порядок, в котором они должны быть расположены. Для каждого элемента сетки браузер сканирует всю сетку в заданном направлении потока (строка или столбец), начиная от начальной точки потока (верхний левый угол, на языках LTR - слева направо), пока не найдет место, куда поместится этот элемент сетки.

Элементы сетки
Контейнер-сетка устанавливает новый контекст форматирования для элементов сетки, который обуславливает следующие особенности:

- Для элементов сетки блокируется их значение свойства `display`. Значение `display: inline-block` вычисляется в `display: block`, анонимные блоки текста также занимают всю ширину контейнера и образуют разрыв строки.
- Размер элемента сетки в пределах содержащего блока определяется его областью сетки.
- Расчеты элементов сетки для `width: auto` и `height: auto` зависят от их значений `align-self`:
  `align-self: normal;` - незамещаемые элементы заполняют область сетки, замещаемые элементы используют собственные размеры;
  `align-self: stretch;` - обе категории элементов заполняют область сетки;
  `align-self: start/center и т.д.` - незамещаемые элементы устанавливают размеры в

соответствии со своим содержимым, замещаемые элементы используют собственные размеры.

- Поскольку соседние элементы сетки находятся в независимых областях сетки, то поля соседних элементов сетки `margin` не схлопываются.
- Браузеры по-разному обрабатывают процентные значения свойств `margin` и `padding`, поэтому не рекомендуется использовать их при задании значений этих свойств.
- Поля `margin: auto;` расширяются, поглощая свободное пространство в соответствующем измерении, поэтому могут использоваться для выравнивания элемента.

Размещение и переупорядочивание элементов сетки
Свойства размещения позволяют свободно упорядочивать и переупорядочивать содержимое сетки таким образом, что визуальное представление может значительно отличаться от порядка элементов в html-документе.

Размещение с помощью линий сетки

Каждый элемент сетки связан с областью сетки, которая определяет содержащий блок для элемента сетки. Положение элементов сетки определяется расположением линий сетки и диапазоном сетки - количеством занимаемых дорожек сетки. По умолчанию элемент сетки занимает одну дорожку на каждой оси. Поэтому можно опустить значение `grid-column-end` или `grid-row-end`.

Свойства размещения на сетке - `grid-row-start`, `grid-row-end`, `grid-column-start` и `grid-column-end` и их краткая запись `grid-row`, `grid-column` и `grid-area` позволяют определить размещение элемента сетки, предоставив любую (или ноль) из следующих шести частей информации:

	Строка	**Столбец**
Начало	Начальная линия строки	Начальная линия столбца
Конец	Конечная линия строки	Конечная линия столбца
Диапазон	Диапазон строк	Диапазон столбцов

grid-row-start, grid-column-start, grid-row-end, grid-column-end

Значения:

`auto`	Свойство не влияет на размещение элемента сетки, указывая на автоматическое размещение или диапазон по умолчанию, равный единице.
имя линии	Начальная и конечная линия строки/столбца задаются в именованных линий сетки.
целое число и имя линии?	Начальная и конечная линия строки/столбца задаются с помощью целого числа (отрицательное порядковый номер линии сетки будет отсчитываться с противоположного края явной сетки) и (необязательно) имени линии.

`span` и целое число или имя линии	Ключевое слово `span` и целое положительное число/имя линии задают диапазон ячеек для размещения элемента сетки.

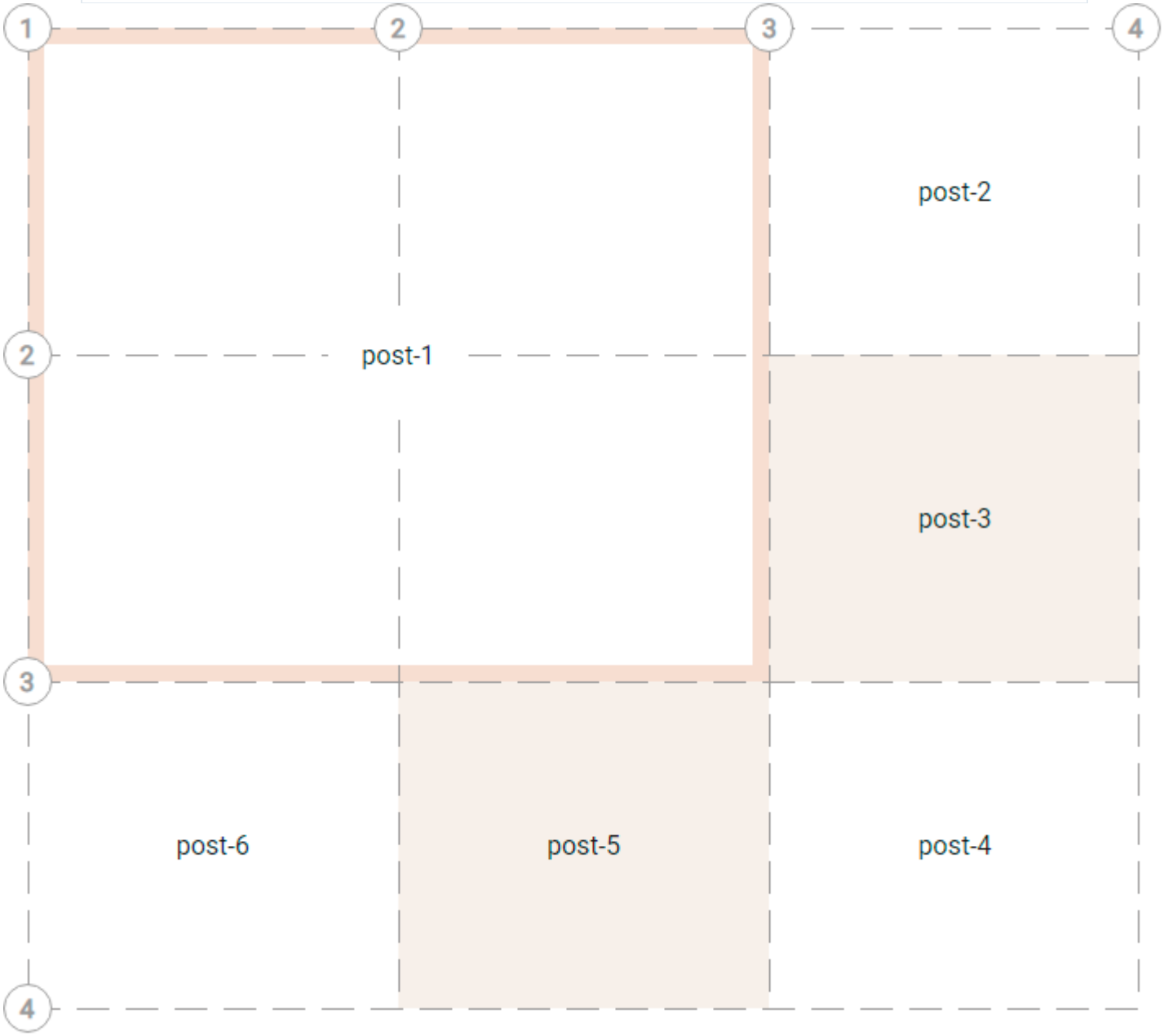


РИС. 5. РАЗМЕЩЕНИЕ ЭЛЕМЕНТОВ СЕТКИ С ПОМОЩЬЮ ЧИСЛОВЫХ ИНДЕКСОВ ЛИНИЙ

```
.grid-container {
  display: grid;
  grid-template-rows: 200px 200px 200px;
  grid-template-columns: 1fr 1fr 1fr;
}
.post-1 {
  grid-row-start: 1;
  grid-row-end: 3;
  grid-column-start: 1;
  grid-column-end: 3;
```

```
}
.post-2 {
  grid-row-start: 1;
  grid-column-start: 3;
}
.post-3 {
  grid-row-start: 2;
  grid-column-start: 3;
}
.post-4 {
  grid-row-start: 3;
  grid-column-start: 3;
}
.post-5 {
  grid-row-start: 3;
  grid-column-start: 2;
}
.post-6 {
  grid-row-start: 3;
  grid-column-start: 1;
}
```
CSS

Именованные линии сетки

Хотя на линии сетки можно ссылаться по их числовому индексу, именованные линии облегчают понимание и использование свойств размещения сетки. Линии могут быть названы явно в свойствах `grid-template-rows` и `grid-template-columns` или неявно путем создания именованных областей сетки в свойстве `grid-template-areas`.

Имя линии может быть любым, при указании в значении свойства оно заключается в квадратные скобки. В качестве имени линии нельзя использовать слово `span`.

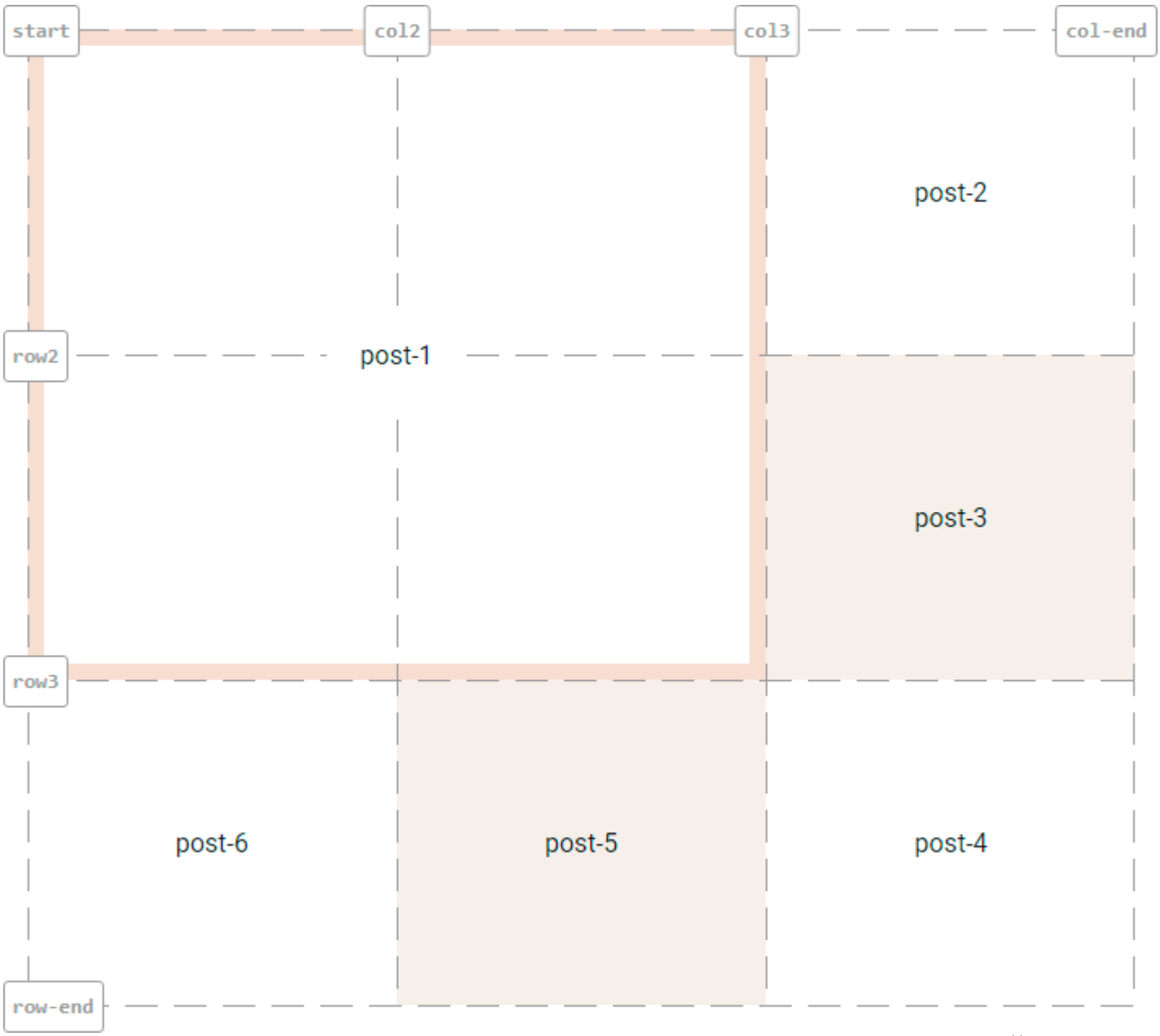


РИС. 6. РАЗМЕЩЕНИЕ ЭЛЕМЕНТОВ СЕТКИ С ПОМОЩЬЮ ИМЕНОВАННЫХ ЛИНИЙ

```
.grid-container {
  display: grid;
  grid-template-rows: [start] 200px [row2] 200px [row3] 200px [row-end];
  grid-template-columns: [start] 1fr [col2] 1fr [col3] 1fr [col-end];
}
.post-1 {
  grid-row-start: span 2;
  grid-column-start: span 2;
}
.post-2 {
  grid-row-start: start;
  grid-column-start: col3;
}
.post-3 {
  grid-row-start: row2;
  grid-column-start: col3;
```

```
}
.post-4 {
  grid-row-start: row3;
  grid-column-start: col3;
}
.post-5 {
  grid-row-start: row3;
  grid-column-start: col2;
}
.post-6 {
 grid-row-start: row3;
 grid-column-start: start;
}
```
CSS

Имена линий добавляются к неявным именам линий сетки, созданным свойством `grid-template-areas`, принимая вид `name-start` и `name-end`. Имена линий сетки никогда не заменяют другие имена линий сетки. Вместо этого они просто накапливаются.

Краткая запись свойств размещения элементов сетки

Свойства `grid-row` и `grid-column` являются сокращенными именами для свойств `grid-row-start` / `grid-row-end` и `grid-column-start` / `grid-column-end` соответственно.

Если заданы два значения, первое (до косой черты) устанавливается для параметра `grid-row-start/grid-column-start`, второе - для `grid-row-end/grid-column-end`. Если второе значение опущено, а первое указано в формате пользовательского идентификатора, то `grid-row-end/grid-column-end` также устанавливается в пользовательское имя сетки. В противном случае, оно вычисляется в `auto`.

Для свойства `grid-area` если указано четыре значения, первое устанавливается для `grid-row-start`, второе - для `grid-column-start`, третье - для `grid-row-end`, четвертое - для `grid-column-end`.

Если `grid-column-end/grid-row-end` не указан, а `grid-column-start/grid-row-start` указан в форме пользовательского имени, то для `grid-column-end/grid-row-end` также устанавливается значение пользовательского имени линии; в противном случае он установлен на `auto`.

Когда `grid-column-start` опущен, а значение `grid-row-start` указан в форме пользовательского имени, оно устанавливается для всех четырех значений. В противном случае оно устанавливается на `auto`.

<table>
<tr><td colspan="4">grid-area</td></tr>
<tr><td colspan="2">grid-row</td><td colspan="2">grid-column</td></tr>
<tr><td>grid-row-start</td><td>grid-row-end</td><td>grid-column-start</td><td>grid-column-end</td></tr>
</table>

Синтаксис

```
grid-row: a;
```

```
grid-row: auto;
grid-column: 2;
grid-row: 1 / -1;
grid-column: sidebar-start / footer-end;
```
CSS
```
grid-area: a;
grid-area: auto;
grid-area: 2 / 4;
grid-area: 1 / 3 / -1;
grid-area: header-start / sidebar-start / footer-end / sidebar-start;
```
Переупорядочивание элементов сетки

Свойство `order` также применяется к элементам сетки. Это влияет на их автоматическое размещение и порядок отрисовки. Свойство должно использоваться только для визуального, а не логического переупорядочения контента.

Выравнивание элементов сетки и промежутки между элементами

Для выравнивания элементов сетки можно использовать свойство `margin`, аналогично, как работает это свойство для блочных элементов.

По умолчанию элементы сетки растягиваются, чтобы заполнить свою область сетки. Тем не менее, если `justify-self` или `align-self` вычисляют значение, отличное от `stretch` или задано `margin: auto`, элементы сетки будут автоматически изменяться в соответствии с их содержимым.

Выравнивание с помощью margin: auto

При расчете размеров дорожек сетки `margin: auto` обрабатываются как `0`. Они поглощают положительное свободное пространство, предшествующее выравниванием с помощью свойств выравнивания. Переполняющиеся элементы игнорируют свои автоматические поля и переполнение, как указано в их свойствах выравнивания блоков.

Выравнивание по оси строки

Элементы сетки могут быть выровнены в направлении оси строки (по горизонтали для LTR-языков) с помощью свойства `justify-self` или свойства `justify-items` (заданного для контейнера-сетки).

Выравнивание по оси столбца

Элементы сетки могут выровнены в направлении, перпендикулярном оси строки с помощью свойства `align-self` или свойства `align-item`, заданного для контейнера-сетки.

Промежутки между элементами сетки

Свойства `row-gap` и `column-gap` (и их сокращенная запись `gap`), если они указаны в контейнере сетки, определяют промежутки между строками и столбцами сетки. При определении размера дорожки каждый промежуток рассматривается как дополнительная пустая дорожка указанного размера. Дополнительный промежуток также может быть добавлен между дорожками за счет свойств `justify-content` и `align-content`.

Промежутки добавляются только между двумя дорожками сетки, то есть они не добавляются перед первой и после последней дорожки.

row-gap, column-gap
Значения:

`normal`	Вычисляется как `0px`. Значение по умолчанию.
длина или `%`	Процентное значение вычисляется относительно размеров области сетки. Отрицательные значения не используются.

Синтаксис

```
row-gap: 1.5em;
column-gap: 10px;
gap: 1%;
```

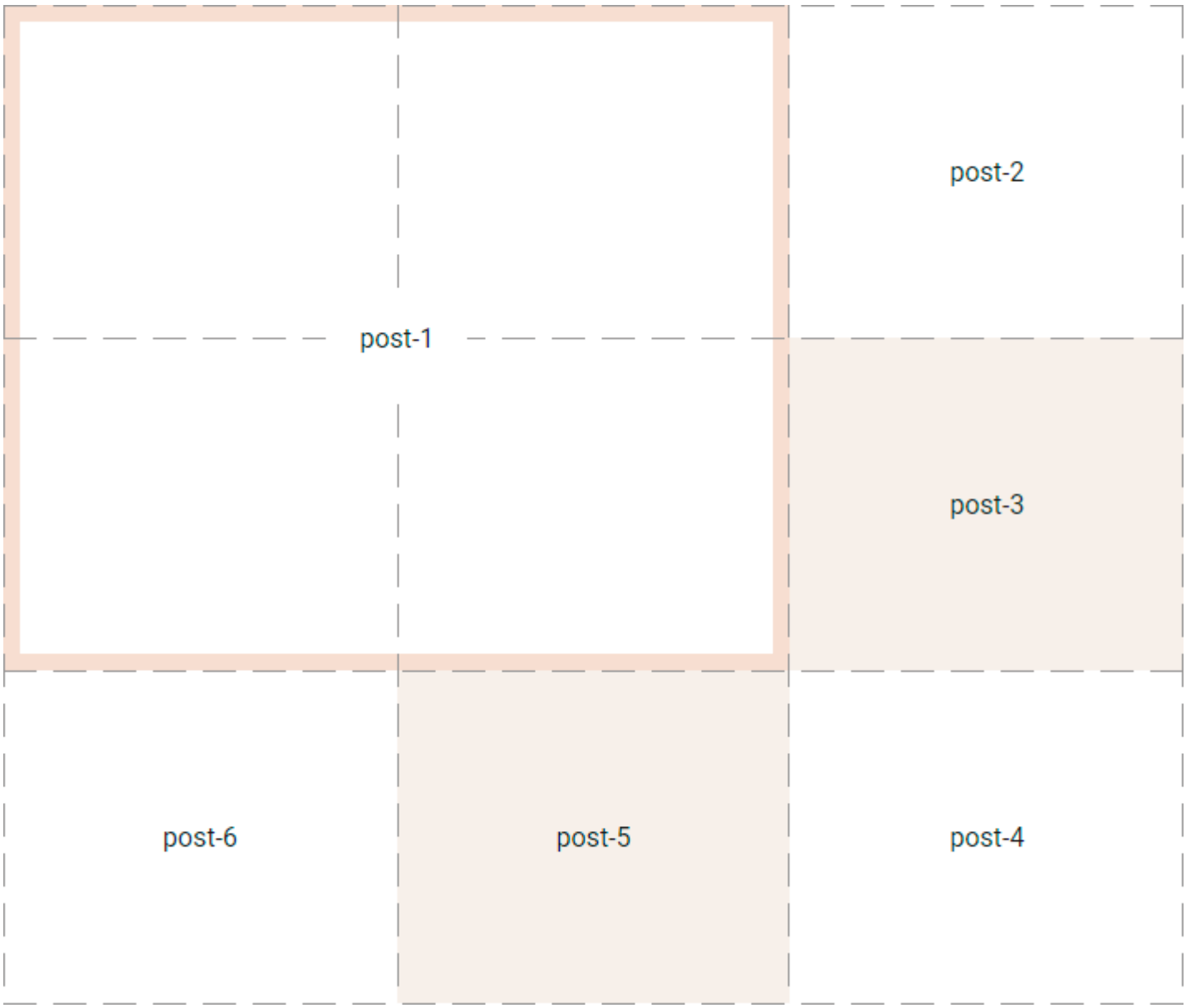


РИС. 2. РАЗМЕЩЕНИЕ ЭЛЕМЕНТОВ СЕТКИ С ПОМОЩЬЮ ИМЕНОВАННЫХ ОБЛАСТЕЙ

```
.grid-container {
  display: grid;
  grid-template-areas: "post-1 post-1 post-2"
                       "post-1 post-1 post-3"
                       "post-6 post-5 post-4";
  grid-template-rows: repeat(3, 200px);
```

```
  grid-template-columns: repeat(3, 1fr);
}
.post-1 {
  grid-area: post-1;
}
.post-2 {
  grid-area: post-2;
}
.post-3 {
  grid-area: post-3;
}
.post-4 {
  grid-area: post-4;
}
.post-5 {
  grid-area: post-5;
}
.post-6 {
  grid-area: post-6;
}
```
CSS

3.3. Краткая запись явной сетки

Свойство `grid-template` является сокращением для установки `grid-template-rows`, `grid-template-columns` и `grid-template-areas` в одном объявлении.

grid-template

Значения:

`none`	Устанавливает для всех трех свойств начальные значения `none`
значение `grid-template-rows` / `grid-template-columns`	Устанавливает `grid-template-rows` и значение `grid-template-columns` в указанные значения, а `grid-template-areas` в значение `none`.
имена линий сетки *или* последовательность идентификаторов, заключенная в кавычки и размер дорожки *или* именованные линии сетки *или* + /явный список дорожек	Устанавливает `grid-template-areas` для перечисленных последовательностей идентификаторов. Устанавливает для `grid-template-rows` указанные значения размеров дорожек, следующие за каждой последовательностью идентификаторов (выставляя `auto` для любых отсутствующих размеров), и объединяет в именованных линиях сетки, определенных до / после каждого размера. Устанавливает `grid-template-columns` в список дорожек, указанный после косой черты (или ни одного, если не указан).

Варианты

```
.grid-container {
  display: grid;
  grid-template: repeat(3, 200px)/repeat(3, 1fr);
}
.post-1 {
  grid-row-start: 1;
  grid-row-end: 3;
  grid-column-start: 1;
  grid-column-end: 3;
}
.post-2 {
  grid-row-start: 1;
  grid-row-end: 2;
  grid-column-start: 3;
  grid-column-end: 4;
}
...
```

CSS

```
.grid-container {
  display: grid;
  grid-template: [start] "post-1 post-1 post-2" 200px [row2]
                 [row2] "post-1 post-1 post-3"  200px [row3]
                 [row3] "post-6 post-5 post-4" 200px [row-end] / 1fr 1fr 1fr;
}
.post-1 {
  grid-area: post-1;
}
.post-2 {
  grid-area: post-2;
}
...
```

CSS

Функция `repeat()` не разрешена для определения списка дорожек в этом свойстве, если используются именованные области сетки (сетка просто не будет отрисована).

1. С помощью CSS Grid Layout определите на странице столбцы и строки: 2 строки и 3 столбца. (**grid-template-column** и **grid-template-row**). Строки задают ячейкам ширину в 50px, а столбцы – длину в 100px. В качестве контента используйте цифры от 1 до 6.

Результат

1 2 3

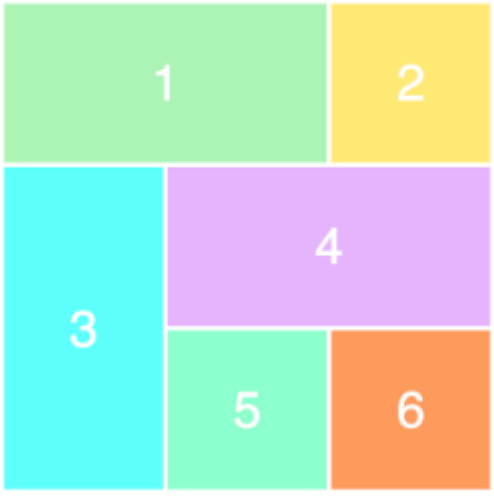
4 5 6

Пример html

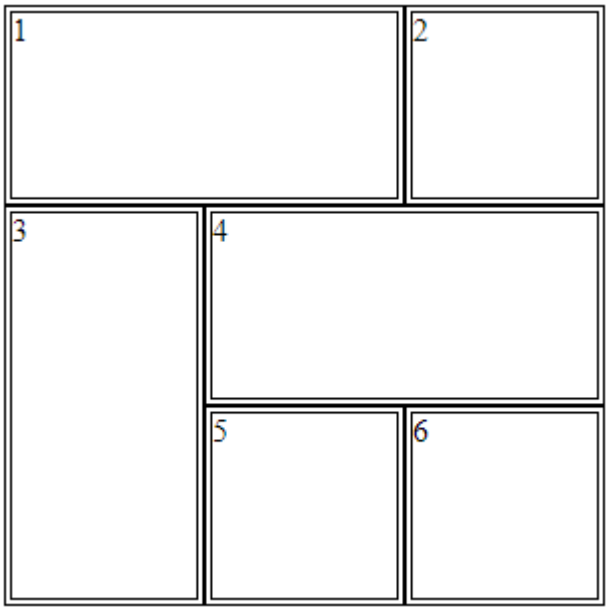
```
<div class="wrapper">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
</div>
```

2. Измените размеры и отпозиционируйте элементы (**grid-row** и **grid-column**) **таким образом, чтобы получилось как на макете.**

Макет



Результат



Пример html

```
<div class="wrapper">
   <div class= "item1">1</div>
   <div class= "item">2</div>
   <div class= "item3">3</div>
   <div class= "item4">4</div>
   <div class= "item">5</div>
   <div class= "item">6</div>
</div>
```

Пример css

```
.item {

    border: 4px double black;
}
```

3. Замените в макете числовые значения на картинки.

Результат