

## Построение сеток на гридах по макету

Выполнен на 0%

Введение

Теория

Базовая теория

Описание сетки

Явные и неявные координаты грид-элементов

Грид-области

Именованные грид-линии

Грид-интервал

Размеры в гриде

Выравнивание в гриде

Углублённая теория

Многослойность в гриде

Повторы в гриде

Сокращённая запись

Продвинутая теория

Поведение автоматически размещаемых элементов

Определение диапазона размеров

Автоматическое заполнение грида

Дополнительный материал

Инструменты для работы с гридами

Баги и ограничения

Методика построения сеток на гридах

Кейс 1, лёгкий уровень

Кейс 2, лёгкий уровень

Кейс 3, лёгкий уровень

Кейс 4, лёгкий уровень

Кейс 5, средний уровень

Кейс 6, средний уровень

Кейс 7, средний уровень

Кейс 8, средний уровень

Кейс 9, сложный уровень

Кейс 10, сложный уровень

Главная / Мое обучение / Построение сеток на гридах по макету / Теория / Базовая теория /

# Именованные грид-линии

Вернёмся к грид-линиям. Когда мы задавали координаты для грид-элементов, мы обращались к грид-линиям по номерам:

```
.header {
  grid-column: 1 / 5; /* элемент располагается с первой грид-линии до пятой */
}
```

Это не всегда удобно. Когда рядов и колонок становится больше, хочется большей явности, например, задать определённой линии не безликий номер, а явное имя. И такая возможность в гридах есть — грид-линиям можно задавать имена.

Давайте возьмём для примера раскладку, которую рассматривали раньше.

```
<div class="grid">
  <header class="header">
  </header>

  <main class="main">
  </main>

  <aside class="aside">
  </aside>

  <footer class="footer">
  </footer>
</div>
```

```
.grid {
  display: grid;

  grid-template-columns: 100px 100px 100px 100px;
  grid-template-rows: 100px 100px 100px 100px;

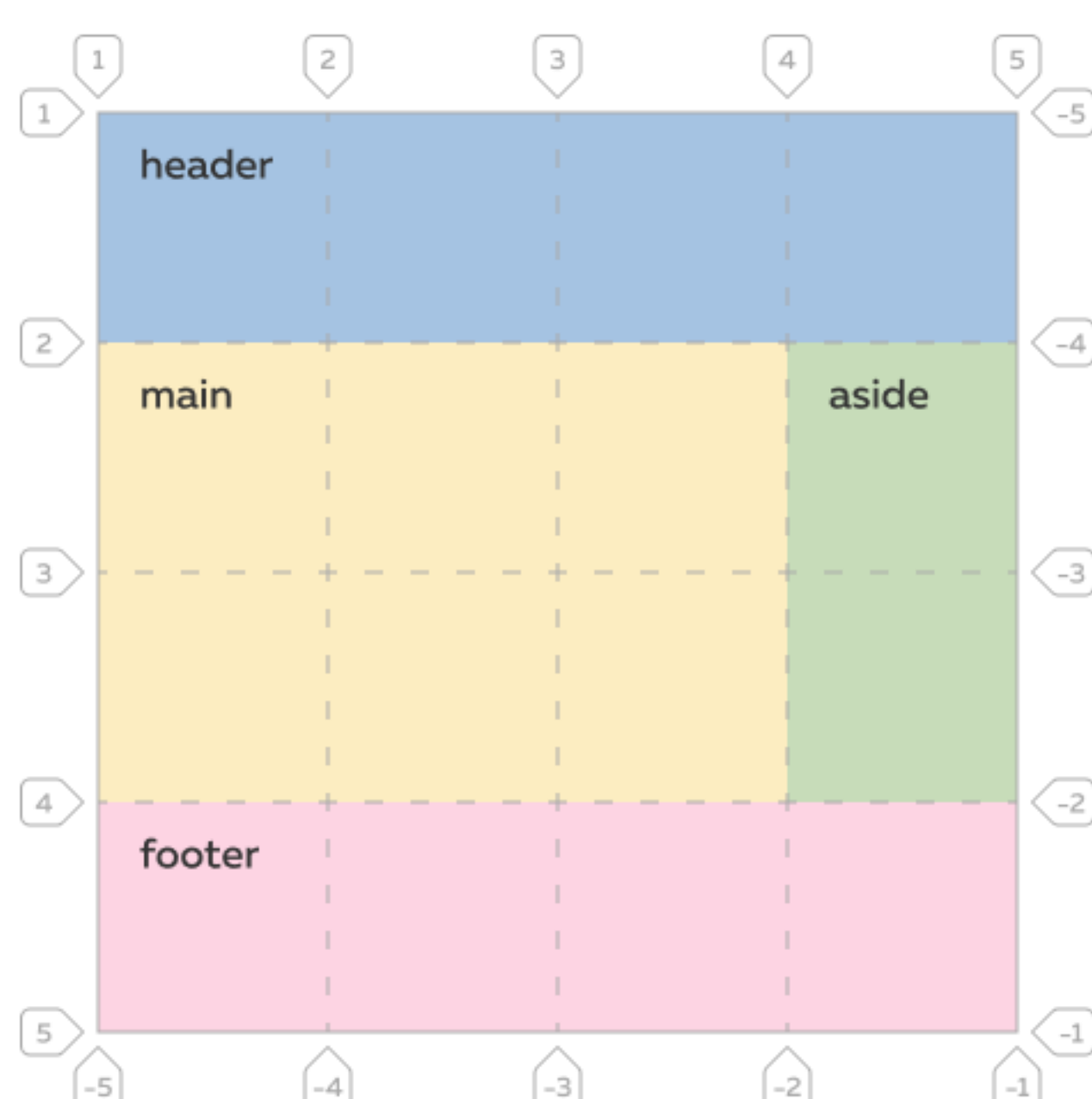
  width: 400px;
  height: 400px;
}

.header {
  grid-column: 1 / 5;
}

.main {
  grid-column: 1 / 4;
  grid-row: 2 / -2;
}

.aside {
  grid-row: 2 / -2;
}

.footer {
  grid-column: 1 / 5;
}
```



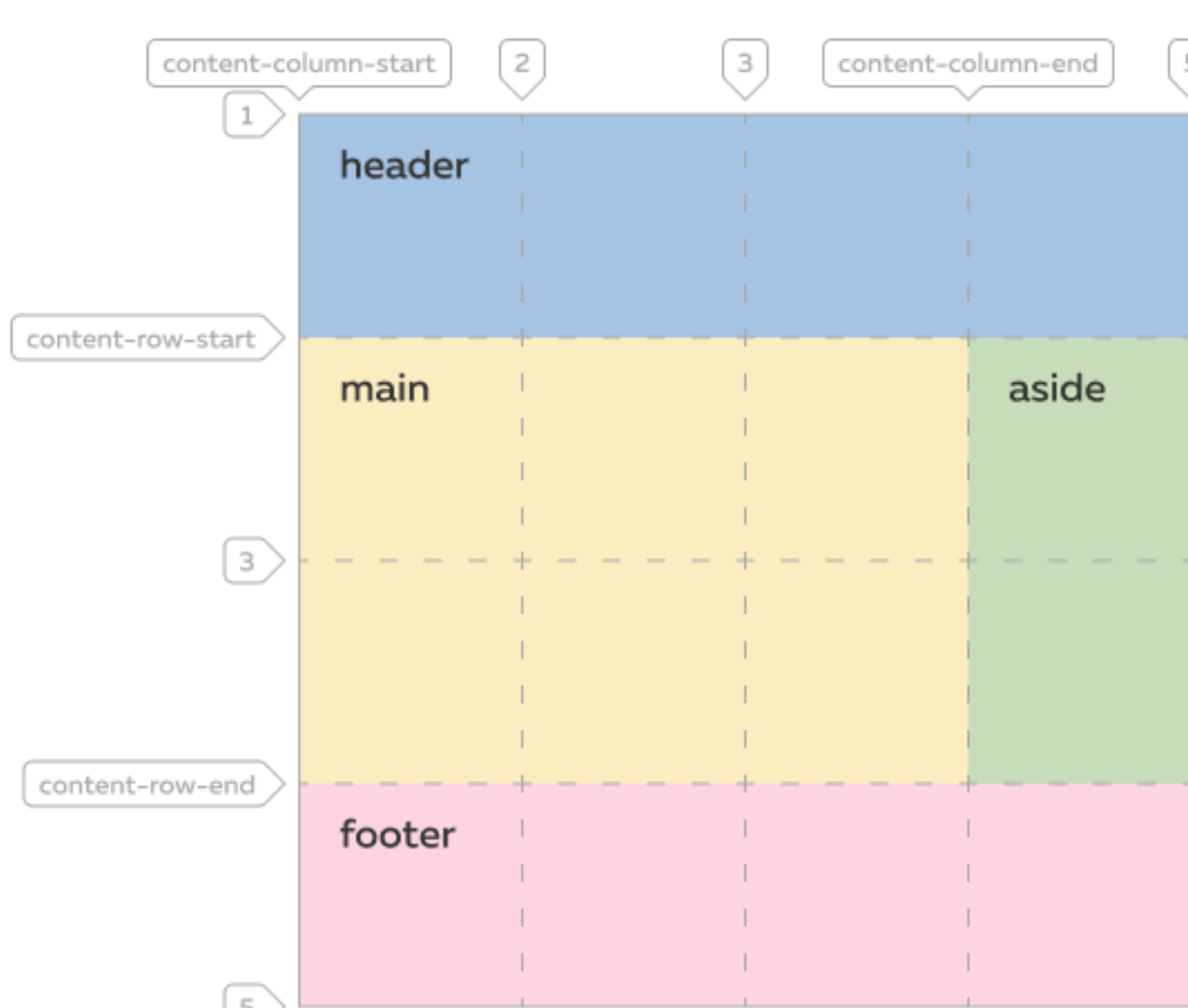
Пример раскладки

В раскладке мы задаём координаты грид-элементам по номерам грид-линий, например так:

```
.main {
  grid-column: 1 / 4; /* элемент с первой линии до четвёртой */
  grid-row: 2 / -2; /* элемент со второй линии до предпоследней */
}
```

Давайте теперь изменим пример и в изначальной раскладке грида пропишем также названия грид-линий для области грид-элемента `.main`. В имени могут использоваться любые буквы, латиница, кириллица, символы юникода и так далее. Имена записываются в квадратных скобках через дефис внутри свойств `grid-template-columns` и `grid-template-rows`. «Стартовые» линии рядов назовём `[content-column-start]` и `[content-row-start]`, а «конечные» — `[content-column-end]` и `[content-row-end]`.

```
.grid {
  --
  grid-template-columns: [content-column-start] 100px 100px 100px [content-column-end] 100px;
  grid-template-rows:
    100px
    [content-row-start]
    100px
    100px
    [content-row-end]
    100px;
  --
}
```



Назвали линии

Теперь при прописывании координат грид-элементов мы можем опираться не только на номера грид-линий, но ещё и на их имена:

```
.main {
  /* элемент начинается с линии «content-column-start» и идёт до линии «content-column-end» */
  grid-column: content-column-start / content-column-end;

  /* элемент начинается с линии «content-row-start» и идёт до линии «content-row-end» */
  grid-row: content-row-start / content-row-end;
}
```

Именованными линиями мы можем пользоваться не только для задания координат грид-элемента `.main`, но и для остальных грид-элементов:

```
.header {
  grid-column: 1 / 5;

  /* элемент идёт с первой линии до линии «content-row-start» */
  grid-row: 1 / content-row-start;
}

.aside {
  /* элемент идёт с линии «content-row-start» до линии «content-row-end» */
  grid-row: content-row-start / content-row-end;
}

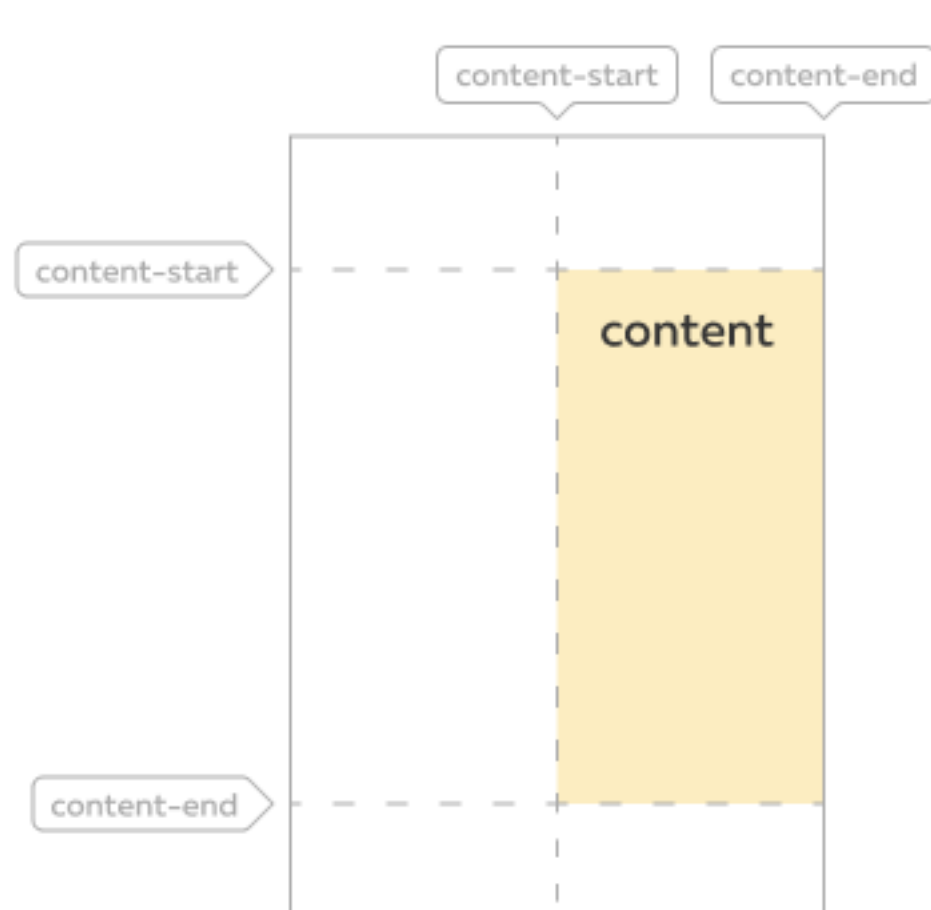
.footer {
  /* элемент идёт с линии «content-row-end» до пятой линии */
  grid-row: content-row-end / 5;
  grid-column: 1 / 5;
}
```

Именованными линиями также можно пользоваться не только в свойствах `grid-row-start/grid-row-end/grid-row` и `grid-column-start/grid-column-end/grid-column`. Рассмотрим пример. Здесь обозначены линии по обоим осям, которые «открывают» и «закрывают» контент — они называются `[content-start]` и `[content-end]`.

```
.grid {
  display: grid;

  grid-template-columns: 100px [content-start] 100px [content-end];
  grid-template-rows:
    50px
    [content-start]
    200px
    [content-end]
    50px;
}

.content {
  /* мы назвали линии единообразно и между линий автоматически появилась грид-область с соответствующим
  линиям названием */
  grid-area: content;
}
```



Грид-область, которая появилась из-за правильно названных линий

## Ознакомились со статьёй?

Сохранить прогресс

Грид-области

Грид-интервал

### Практикум

Тренажёры  
Подписка  
Для команд и компаний  
Учебник по PHP

### Профессии

Фронтенд-разработчик  
React-разработчик  
Фулстек-разработчик  
Бэкенд-разработчик

### Услуги

Работа наставником  
Для учителей  
Стать автором

### Курсы

HTML и CSS. Профессиональная верстка сайтов  
HTML и CSS. Адаптивная верстка и автоматизация  
JavaScript. Профессиональная разработка веб-интерфейсов  
JavaScript. Архитектура клиентских приложений  
React. Разработка сложных клиентских приложений  
PHP. Профессиональная веб-разработка  
PHP и Yii. Архитектура сложных веб-сервисов  
Node.js. Разработка серверов приложений и API  
Анимация для фронтендеров  
Верстка email-рассылок  
Vue.js для опытных разработчиков  
Регулярные выражения для фронтендеров  
Шаблонизаторы HTML  
Алгоритмы и структуры данных  
Анатомия CSS-каскада

### Блог

С чего начать  
Шпаргалки для разработчиков  
Отчеты о курсах

### Информация

Об Академии  
О центре карьеры

### Остальное

Написать нам  
Мероприятия  
Форум