Шаг 1. Создание общего контейнера для сетки

В обоих случаях (приложение или контентный сайт) нам нужно первым делом создать общий контейнер (контейнер-центровщик), который будет ограничивать сетку по ширине и размещать её по центру страницы.

Ширину контейнера определяют по макету. Чтобы разместить контейнер по центру, можно использовать автоматический внешний отступ.

```
.container {
  width: 1140;
  margin: 0 auto;
}
```

Как правило, в макете предусмотрены поля. За счёт них дизайнеры выделяют контентную часть. Можно установить поля и у контейнера-центровщика, а их размеры определить на своё усмотрение (если они чётко не определены в техническом задании к макету). К примеру, так:

```
.container {
  width: 1140;
  margin: 0 auto;
  padding-left: 15px;
  padding-right: 15px;
}
```

После того как мы определили стили для контейнера-центровщика, нужно решить, какой элемент будет выполнять эту роль. А именно: нужно ли создать дополнительный элемент в разметке или достаточно добавить ещё один класс для существующего элемента (например, секции или раздела сайта). Это определяется также особенностями макета. Представим, что в качестве оформления для блока в макете предполагается фон или нижняя граница, которые тянутся по всей ширине окна браузера, в то время как контент ограничен центровщиком. В этом случае понадобится отдельный элемент, так как фон будет «прикреплён» к верхнему элементу разметки. Если речь о шапке, всё содержимое которой является навигацией, таким контейнером-центровщиком может быть тег `<nav>`. В других случаях может понадобиться дополнительный `<div>`.

```
<header class="header">
  <div class="container">
    …
  </div>
</header>
```

При этом стоит ожидать, что если мы создали отдельный элемент для центровщика, то сеточные стили для блока придётся привязывать ко второму классу у этого нового элемента, ведь именно от будет родительским элементом для отдельных элементов сетки.

```
<header class="header">
  <nav class="header-navigation container">
    …
  </nav>
</header>
```

Если ширина нижней границы совпадает с шириной контента в шапке, то достаточно добавить дополнительный класс для `<header class="header">`.

```
<header class="header container">
  …
</header>
```

Возможен и другой вариант — совместить стили для `<header class="header">` и контейнер-центровщика в одном CSS-правиле, но этот вариант не универсален, так как мы не сможем переиспользовать элемент `container` там, где нужен только контейнер-центровщик.

Разберём, как добавить контейнер-центровщик в раскладку приложения и раскладку контентной страницы.

Раскладка приложения

В случае приложения, которое старается уместиться в доступную высоту экрана, мы будем делать один общий контейнер-центровщик. В роли такого контейнера-центровщика может выступать `<div>`, который идёт первым в `<body>` и оборачивает всю остальную разметку, или же сам `<body>`.

```
<body class="container">
  …
</body>
```

```
.container {
  width: 1440;
  margin: 0 auto;
}
```


Контейнер-центровщик ограничивает размер «рабочей области»

Раскладка контентной страницы

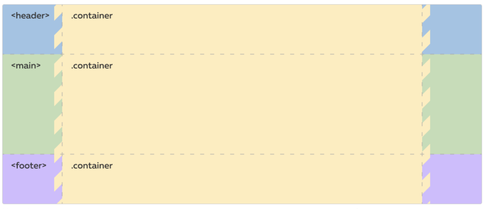
На контентной странице в `<body>` могут находиться секции и разделы документа, а контейнеры-центровщики будут располагаться внутри каждого из разделов.

```
<body>
  <header>
    <div class="container"></div>
  </header>

  <main>
    <div class="container"></div>
  </main>

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

```
.container {
  width: 1440;
  margin: 0 auto;
}
```



Контейнеры-центровщики для каждого смыслового раздела ограничивают ширину блоков

Замечание

Для фиксированной сетки желательно также задать ширину, при которой сайт нормально смотрится на минимальной для макета ширине. Как правило, для этого устанавливают свойство `min-width` у тега `<body>`. Значение для свойства можно получить из макета.

```
body {
    margin: 0;
    padding: 0;
    min-width: 1440px;

    /* … */
}
```

Ознакомились со статьёй?

Сохранить прогресс

Методика построения сеток на гридах
Шаг 2. Строим сетку для блоков первого уровня