????????

- Минимальный шаблон
- Контейнеры
- Сетки
- Колонки

?????????? ??????

```
<!doctype html>
<html lang="ru">
<head>
    <title>Minimal HTML5 page</title>
    <meta  charset="utf-8">
    <meta  name="viewport" content="width=device-width, initial-scale=1">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css"
rel="stylesheet" crossorigin="anonymous">
</head>
<body>

  <!-- Вариант 1: Набор Bootstrap с Popper -->
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js"
crossorigin="anonymous"></script>
  <!-- Вариант 2: Отдельные Popper и Bootstrap -->
  <!--
  <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.8/dist/umd/popper.min.js"
crossorigin="anonymous"></script>
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.min.js"
crossorigin="anonymous"></script>
  -->
</body>
</html>
```

??????????

Общая информация

- необходимы при использовании сеточной системы
- используются для содержания, заполнения и (иногда) центрирования содержимого
- могут быть вложенными, но для большинства макетов это не требуется.

Контрольные точки

Extra small		<576px
Small	-sm	≥576px
Medium	-md	≥768px
Large	-lg	≥992px
Extra large	-xl	≥1200px
Extra Extra large	-xxl	≥1400px

Типы контейнеров

- .container - адаптивный контейнер фиксированной ширины. При изменении размеров браузера его ширина изменяется до новой ширины контрольной точки. При extra small выставляется 100% ширины.
- .container-{breakpoint}, который равен width: 100% до указанной контрольной точки
- .container-fluid, который равен width: 100% во всех контрольных точках

	<576px	**≥576px**	**≥768px**	**≥992px**	**≥1200px**	**≥1400px**
.container	100%	540px	720px	960px	1140px	1320px
.container-sm	100%	540px	720px	960px	1140px	1320px
.container-md	100%	100%	720px	960px	1140px	1320px
.container-lg	100%	100%	100%	960px	1140px	1320px
.container-xl	100%	100%	100%	100%	1140px	1320px
.container-xxl	100%	100%	100%	100%	100%	1320px
.container-fluid	100%	100%	100%	100%	100%	100%

?????

- Сетка внутри контейнера, относительно ширины которого рассчитываются размеры колонок.
- Для строки - класс row, для столбца - класс col.
- По документации в строке может быть до 12 ячеек, но если их больше, и текст внутри короткий - визуально в строку помещается больше столбцов.
- Визуально внутри row может быть несколько строк при превышении 12.
- Одна строка всегда находится под другой.

Минимальный пример сетки:

```
<div class="container">
  <div class="row">
    <div class="col">
      Колонка
    </div>
    <div class="col">
      Колонка
    </div>
    <div class="col">
      Колонка
    </div>
  </div>
</div>
```

- Модификаторы относительно размера окна, контрольные точки аналогичные контрольным точкам для сетки.
- Для контрольной точки может быть дополнительный параметр - кол-во занимаемых ячеек. Например col-md-6.
- Поведение: до размера окна, указанного в контрольной точке ширина 12, после - в соответствии с цифрой. Но за счет комбинации классов можно сделать, чтобы например на большом было 4 столбца один за одним, меньше - 2 и 2, еще меньше - 1,1,1,1

```
<div class="row">
   <div class="col-md-6 col-xl-3">one</div>
   <div class="col-md-6 col-xl-3">two</div>
```

```
   <div class="col-md-6 col-xl-3">three</div>
   <div class="col-md-6 col-xl-3">four</div>
</div>
```

- Вместо цифры может быть -auto в этом случае ширина зависит от контента
- Может быть без контрольной точки

```
<div class="row">
    <div class="col-8">col-8</div>
    <div class="col-4">col-4</div>
</div>
```

- Может быть без доп. параметра, но тогда должен быть предшествующий с полным набором параметров

```
<div class="container text-center">
  <div class="row">
    <div class="col-sm-8">col-sm-8</div>
    <div class="col-sm-4">col-sm-4</div>
  </div>
  <div class="row">
    <div class="col-sm">col-sm</div>
    <div class="col-sm">col-sm</div>
    <div class="col-sm">col-sm</div>
  </div>
</div>
```

- В этом случае берется предыдущее распределение и применяется к неуказанному

За счет применения класса row-cols-* у div row фиксируется визуальное количество столбцов в одной строке (по-моему это повтор функционала). Хотя поведение данного класса несколько неясно: для такой настройки

```
<div class="row row-cols-4">
    <div class="col">Колонка</div>
    <div class="col">Колонка</div>
    <div class="col-6">Колонка</div>
    <div class="col">Колонка</div>
  </div>
```

последняя колонка будет визуально на следующей строке, что странно. Установленное количество строк фиксируется для всех размеров. Однако реальное назначение данного

функционал несколько непонятно.

Строки могут быть вложенными.

???????

Вертикальное выравнивание

Единый класс для строки

Единый класс для строки с фиксированной высотой строки настраивается общее выравнивание

align-items-start	вверху
align-items-center	посередине
align-items-end	снизу

Например, устанавливается высота в 100px и выравнивание посередине для всех колонок

```
<style>
  .middle-row{
    height: 100px;
    }
</style>
<div class="container">
  <div class="row align-items-center middle-row">
    <div class="col-sm-8">col-sm-8</div>
    <div class="col-sm-4">col-sm-4</div>
  </div>
</div>
```

Индивидуальное выравнивание каждого столбца

Класс align-self-* который применяется для класса столбца

```
<style>
  .middle-row{
    height: 100px;
    }
</style>
<div class="container">
  <div class="row align-items-center middle-row">
    <div class="col-sm-8">col-sm-8</div>
```

```
    <div class="col-sm-4 align-self-end">col-sm-4</div>
  </div>
</div>
```

Горизонтальное выравнивание

justify-content-* для выравнивания всех столбцов в классе row

justify-content-start	Колонки смещены в начало, между колонками нет промежутка
justify-content-center	Колонки в центре, между колонками нет промежутка
justify-content-end	Колонки смещены в конец, между колонками нет промежутка
justify-content-around	Последующие три класса с промежутками (считая промежуток как колонку) и выравниванием относительно этого. +/- для использования.
justify-content-between	
justify-content-evenly	

Порядок отображения

Классы order- , поддерживают контрольные точки. Элементы без порядка имеют приоритет выше.

```
<div class="row">
    <div class="col">
      Первый в DOM, первый в отображении
    </div>
    <div class="col order-5">
      Второй в DOM, последний в отображении
    </div>
    <div class="col order-1">
      Третий в DOM, третий в отображении
    </div>
    <div class="col">
      Последний в DOM, второй в отображении
    </div>
  </div>
```

Однако классы order-first и order-last имеют наивысший приоритет.

Смещение колонок

Классы offset-контр_точка-* смещают колонку влево на указанную ширину.

```
<div class="row">
    <div class="col-md-4">.col-md-4</div>
    <div class="col-md-4 offset-md-4">.col-md-4 .offset-md-4</div>
</div>
```

Класс offset-контр_точка-0 сбрасывает смещение в контрольной точке.

Автономное использование колонок

Классы `.col-*` также могут использоваться вне `.row`, чтобы дать элементу определенную ширину. Когда классы колонок используются как непрямые дочерние элементы строки, отступы опускаются. Могут использоваться контрольные точки.

```
<div class="col-3 p-3 mb-2">
  .col-3: ширина 25%
</div>

<div class="col-sm-9 p-3">
  .col-sm-9: ширина 75% выше контрольной точки sm
</div>
```

Плавающее изображение

```
<div class="clearfix">
  <img src="..." class="col-md-6 float-md-end mb-3 ms-md-3" alt="...">
  <p>
    Абзац текста-заполнителя. Мы используем его здесь, чтобы показать использование класса clearfix. Мы добавляем здесь довольно много бессмысленных фраз, чтобы продемонстрировать, как столбцы здесь взаимодействуют с плавающим изображением.
  </p>
  <p>
    Как видите, абзацы изящно обтекают плавающее изображение. А теперь представьте, как это будет выглядеть с фактическим содержанием здесь, а не просто с этим скучным текстом-заполнителем, который продолжается и продолжается, но на самом деле не передает никакой осязаемой информации. Он просто занимает место, и его не стоит читать.
  </p>
  <p>
    И тем не менее, здесь вы, по-прежнему настойчив в чтении этот текст-заполнитель, в надежде еще несколько прозрений, или некоторые скрытые пасхальные яйца содержания. Возможно, шутка. К
```

```
сожалению, здесь ничего этого нет.
  </p>
</div>
```