**💎Вопросы**

1. **Почему в большинстве ситуаций стоит использовать flexbox-позиционирование?**

**Flexbox позиционирование позволяет сделать адаптивный дизайн, т.е. распределять и выравнивать элементы таким образом, чтобы их можно было видеть на разных типах устройств с различными размерами дисплея.**

1. **Самостоятельно изучи способ позиционирования через display: table и ответьте на вопрос, для каких ситуаций оно лучше всего подходит?**

CSS-таблица отличается от HTML-таблицы тем, что она может перестать быть таблицей, если изменить ее CSS-свойства. А HTML-таблица на такое не способна. Это открывает широкие возможности выборочного использования в вёрстке нужных табличных особенностей. Лучше всего display: table подходит для динамического выравнивания. С его помощью можно получить, к примеру, истинное выравнивание по вертикали элементов с динамической высотой.

Однако у display: table есть ограничение – объединение ячеек colspan и rowspan не поддерживаются.

1. **Какие есть оси во флекс-верстке и как задается их направление?**

Главная и поперечная оси. Элементы флекс-контейнера выравниваются вдоль этих осей. Главная ось указывает направление потока флекс-элементов. По умолчанию он течёт по горизонтали слева направо, от начала к концу оси. Направлением главной оси можно управлять, меняя значения свойства flex-direction:

flex-direction: row — ось направлена слева направо, элементы располагаются в ряд. Это значение по умолчанию.

flex-direction: row-reverse — элементы идут в обратном направлении — справа налево.

flex-direction: column — элементы располагаются в столбец сверху вниз.

flex-direction: column-reverse — элементы идут снизу вверх.

Поперечная ось всегда находится под углом 90 градусов к главной. Если главная ось горизонтальна, поперечная идёт сверху вниз, а если главная вертикальна, то поперечная направлена слева направо.

1. **Разберитесь, как работает свойство margin: auto во флекс-верстке, приведите пример использования.**

Очень частый приём, который используется в вёрстке — выравнивание блочного элемента по центру родителя при помощи margin: 0 auto.

Если блочному элементу установить margin-left и margin-right в значении auto, блок центрируется по горизонтали. Для вертикального выравнивания это не сработает.

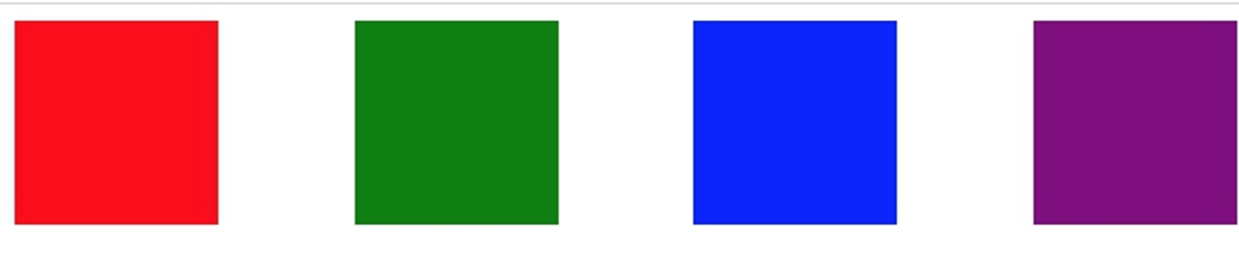
1. **В чем преимущества box-sizing?**

Свойство box-sizing позволяет изменять поведение блочных элементов, размер которых обычно увеличивается в зависимости от толщины границ, внешних или внутренних отступов. Box-sizing нивелирует эти параметры и позволяет вписать блок, не изменив размеры блока.

1. **Чем отличается flex-grow от flex-shrink?**

Если свойство flex-grow помогает распределить свободное пространство между элементами, то flex-shrink помогает распределить пространство, если места в контейнере недостаточно. Необходимая ширина рассчитывается на основании начальной ширины, которую имеет каждый flex-элемент в ней.

1. **Как можно добиться следующего позиционирования элементов:**



<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Square positioning</title>

    <!-- Сначала Normalize -->

    <link rel="stylesheet" href="/styles/normalize.css" />

    <!-- Потом свои стили -->

    <link rel="stylesheet" href="/styles/style4.css" />

  </head>

  <body>

    <div class="flexcontainer">

      <div class="block block-one">1</div>

      <div class="block block-two">2</div>

      <div class="block block-three">3</div>

      <div class="block block-four">4</div>

    </div>

  </body>

</html>

/\* Позиционирование квадратов\*/

:root {

  --square-size: 250px;

}

.flexcontainer {

  display: flex;

  justify-content: space-around;

  align-items: flex-start;

}

.block {

  padding: 30px;

  box-sizing: border-box;

  margin-top: 5px;

}

.block-one {

  width: var(--square-size);

  height: var(--square-size);

  background: red;

}

.block-two {

  width: var(--square-size);

  height: var(--square-size);

  background: green;

}

.block-three {

  width: var(--square-size);

  height: var(--square-size);

  background: blue;

}

.block-four {

  width: var(--square-size);

  height: var(--square-size);

  background: rgb(130, 47, 185);

}

Или?

Html:

<div class=”flex-container”>

<div class=”item-red”></div>

<div class=”item-green”></div>

<div class=”item-blue”></div>

<div class=”item-violet”></div>

</div>

CSS:

.flex-container {

display: flex;

justify-content: row;

align-items: stretch;

}

.item-red{

width:300px;

height:300px;

background: red;

}

.item-green{

width:300px;

height:300px;

background: green;

}

.item-blue{

width:300px;

height:300px;

background: blue;

}

.item-violet{

width:300px;

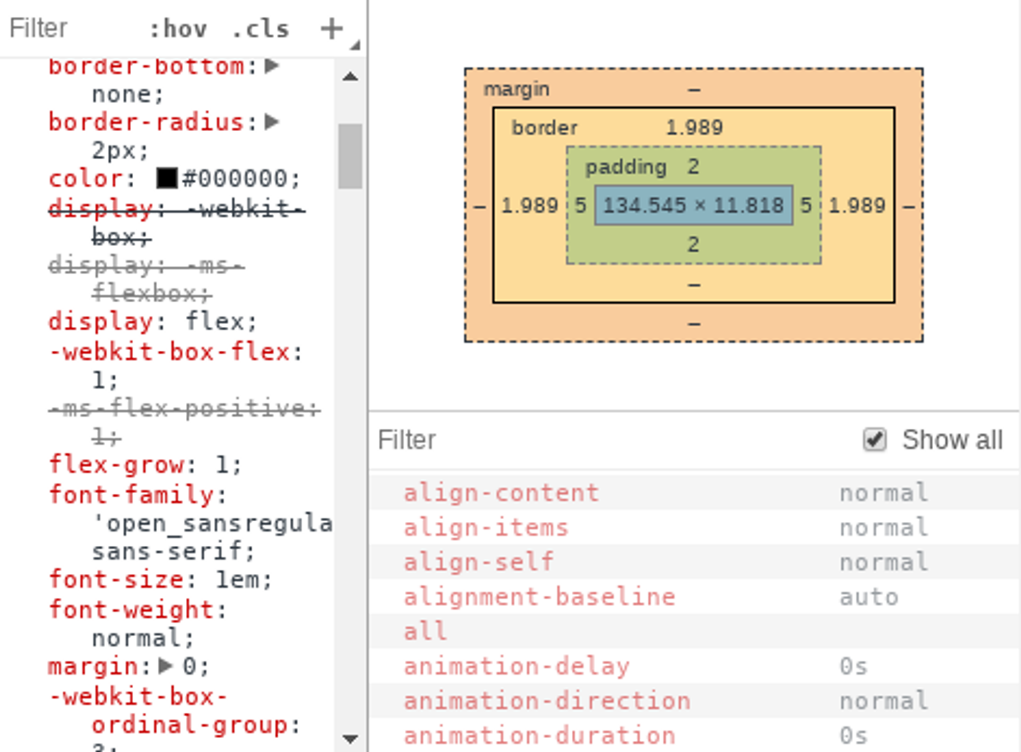
height:300px;

margin-right:150px;

background: violet;

}

1. **Какой в итоге будет размер у элемента?**



**Width: 148.523**

**Height: 17.807**

1. **Самостоятельно разберись, зачем нужно свойство order?**

Свойство order определяет порядок вывода флексов внутри флекс-контейнера. Элементы располагаются согласно значениям свойства order от меньшего к большему. При равных значениях order элементы выводятся в том порядке, в каком они появляются в исходном коде. Поскольку order предназначен только для воздействия на визуальный порядок элементов, а не на их логический порядок или порядок вкладок, order не должен использоваться для невизуальных медиа, таких как речь.

1. **Каким кодом можно сделать такую таблицу?**

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <title>Table</title>

  </head>

  <body>

    <h1>Homework</h1>

    <div class="container"></div>

    <table border="1px">

      <thead>

        Table

      </thead>

      <th>Column1</th>

      <th>Column2</th>

      <th>Column3</th>

      <tr>

        <td rowspan="2">Row 1 Cell 1</td>

        <td>Row 1 Cell 2</td>

        <td>Row 1 Cell 3</td>

      </tr>

      <tr>

        <td>Row 2 Cell 2</td>

        <td>Row 2 Cell 3</td>

      </tr>

      <td colspan="3">Row 3 Cell 1</td>

    </table>

  </body>

</html>

1. **Как рассчитывается размер flex-контейнера?**

Флекс-контейнер — родительский элемент: гибкое пространство, в котором элементы распределяются по правилам, установленным для контейнера

Флекс-элементы — дочерние элементы флекс-контейнера; подчиняются правилам, описанным как для контейнера, так и для отдельных элементов персонально.

Когда во флекс-контейнере несколько элементов, они сразу же выстраиваются в строку независимо от типа.