**Теория к практическому занятию №9**

**Модуль компоновки CSS Flexbox**

Перед модулем компоновки Flexbox было четыре режима компоновки:

* Блок, для разделов на веб-странице
* Встроенный, для текста
* Таблица для двумерных табличных данных
* Положение, для явного положения элемента

Модуль компоновки гибких коробок упрощает разработку гибкой адаптивной структуры макета без использования float или позиционирования.

### 1. Основные понятия

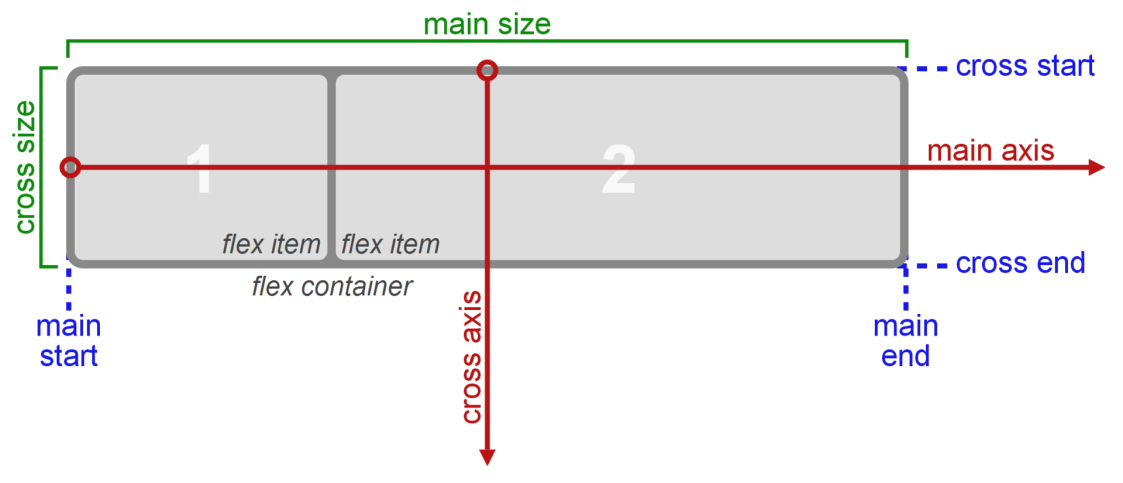


РИС. 1. МОДЕЛЬ FLEXBOX

Для описания модуля Flexbox используется определенный набор терминов. Значение flex-flow и режим записи определяют соответствие этих терминов физическим направлениям: верх / право / низ / лево, осям: вертикальная / горизонтальная и размерам: ширина / высота.

**Главная ось (main axis)** — ось, вдоль которой выкладываются flex-элементы. Она простирается в основном измерении.

**Main start и main end** — линии, которые определяют начальную и конечную стороны flex-контейнера, относительно которых выкладываются flex-элементы (начиная с main start по направлению к main end).

**Основной размер (main size**) — ширина или высота flex-контейнера или flex-элементов, в зависимости от того, что из них находится в основном измерении, определяют основной размер flex-контейнера или flex-элемента.

**Поперечная ось (cross axis)** — ось, перпендикулярная главной оси. Она простирается в поперечном измерении.

**Cross start и cross end** — линии, которые определяют начальную и конечную стороны поперечной оси, относительно которых выкладываются flex-элементы.

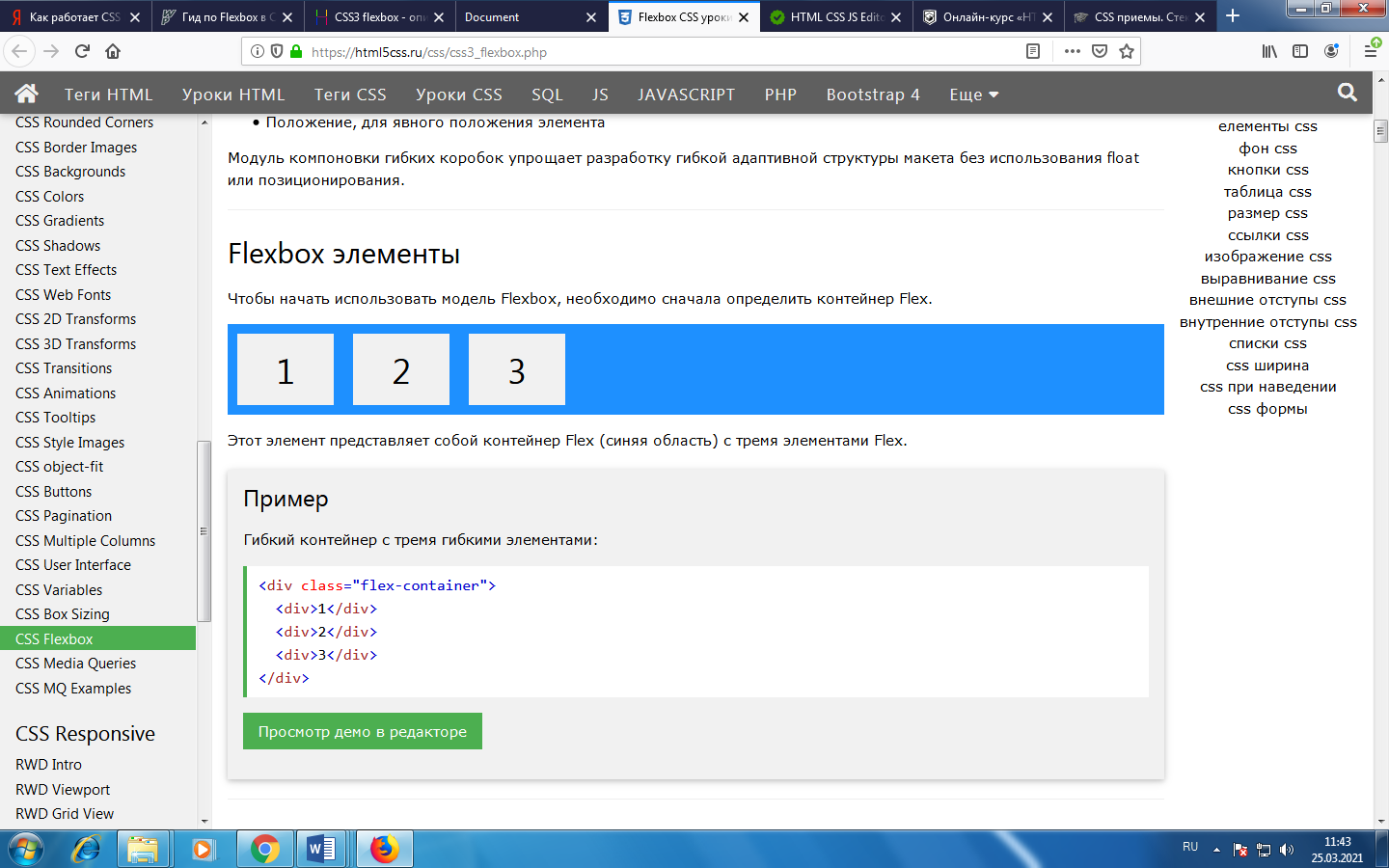
**Поперечный размер (cross size)** — ширина или высота flex-контейнера или flex-элементов, в зависимости от того, что находится в поперечном измерении, являются их поперечным размером.



РИС. 2. РЕЖИМ СТРОКИ И КОЛОНКИ

**Flexbox элементы**

Чтобы начать использовать модель Flexbox, необходимо сначала определить контейнер Flex.



Этот элемент представляет собой контейнер Flex (синяя область) с тремя элементами Flex.

<!DOCTYPE html>

<html>

<head>

<style>

.flex-container {

display: flex;

background-color: DodgerBlue;

}

.flex-container > div {

background-color: #f1f1f1;

width: 100px;

margin: 10px;

text-align: center;

line-height: 75px;

font-size: 30px;}

</style>

</head>

<body>

<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div>3</div>  
</div>

</body>

</html>

Гибкий контейнер с тремя гибкими элементами:

<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div>3</div>  
</div>

**Родительский элемент (контейнер)**

Гибкий контейнер становится гибким, установив display свойство в значение *Flex*:

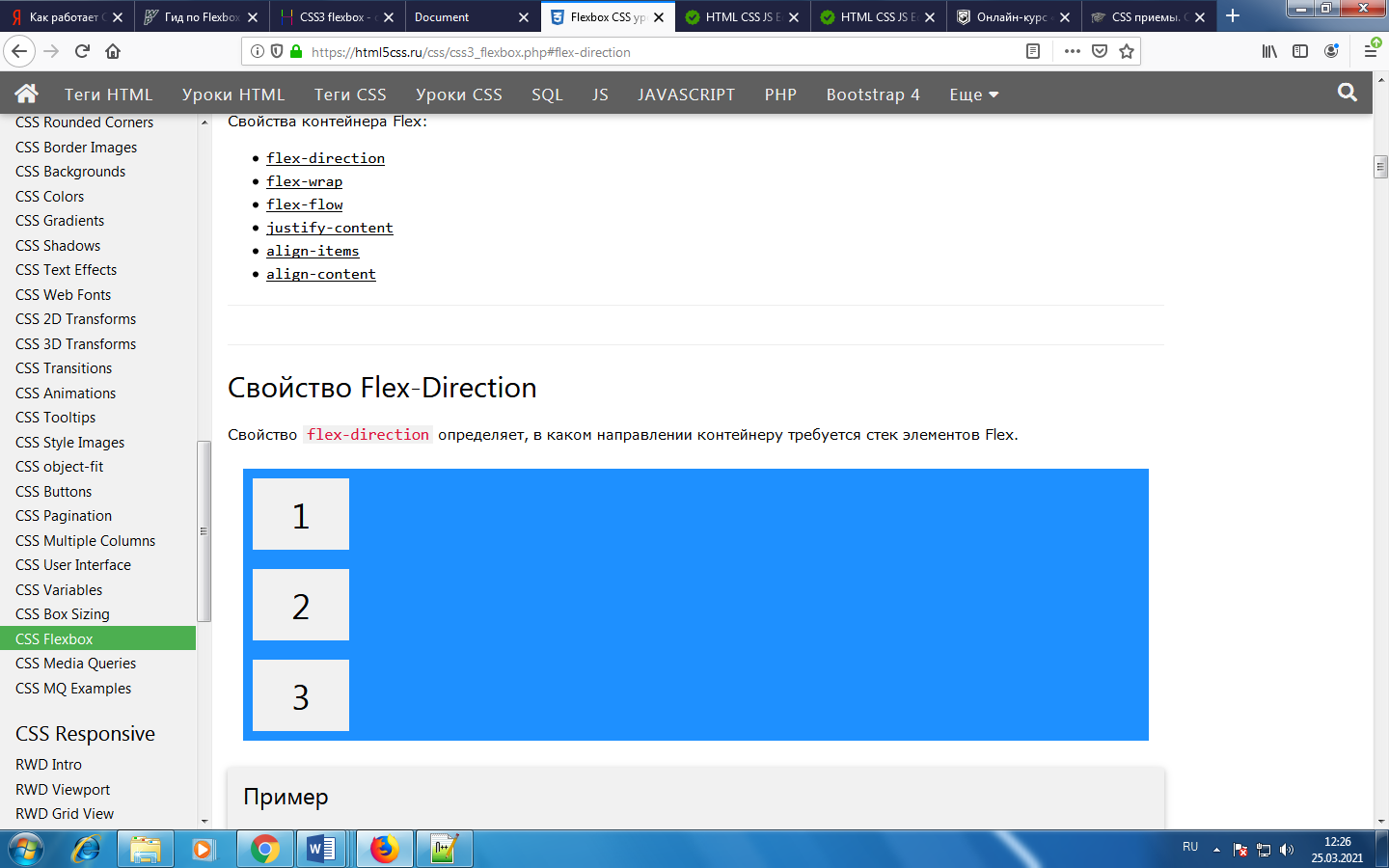
.flex-container {  
  display: flex;  
}

## Свойство Flex-Direction

Свойство flex-direction определяет, в каком направлении контейнеру требуется стек элементов Flex.

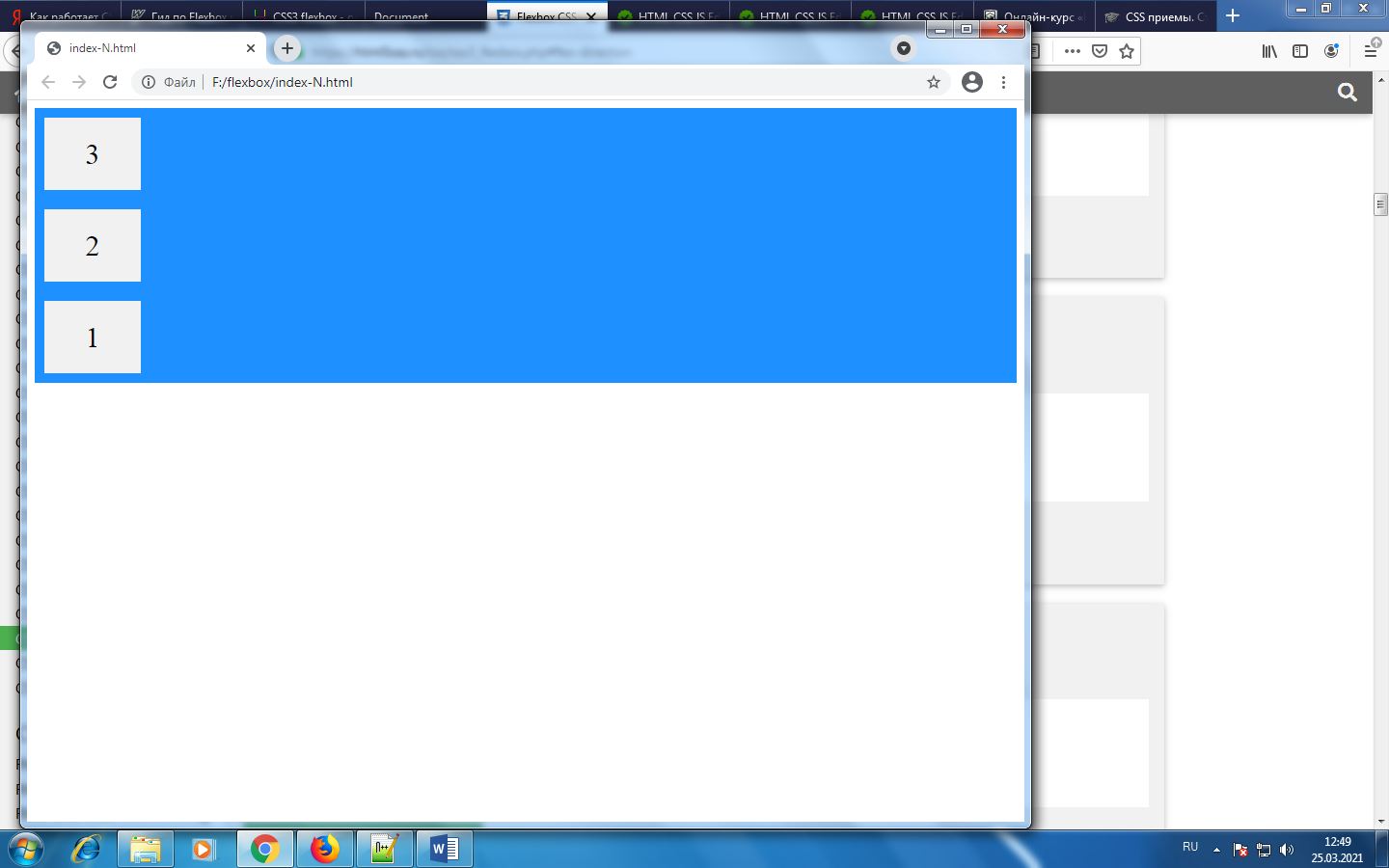
Значение column суммирует элементы Flex по вертикали (сверху вниз):

.flex-container {  
  display: flex;  
  flex-direction: column;  
}



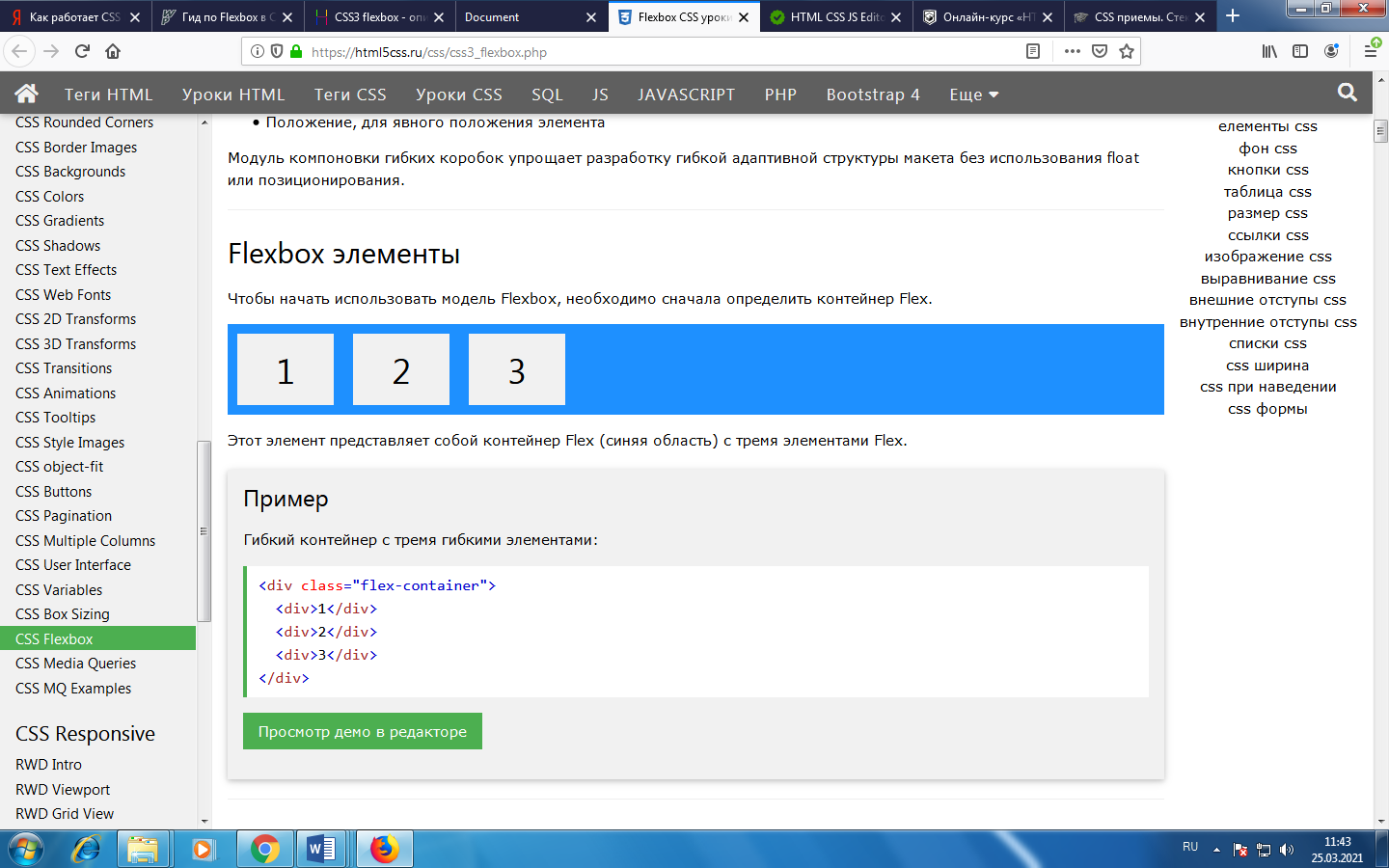
Значение column-reverse суммирует элементы Flex по вертикали (но снизу вверх):

.flex-container {  
  display: flex;  
  flex-direction: column-reverse;  
}



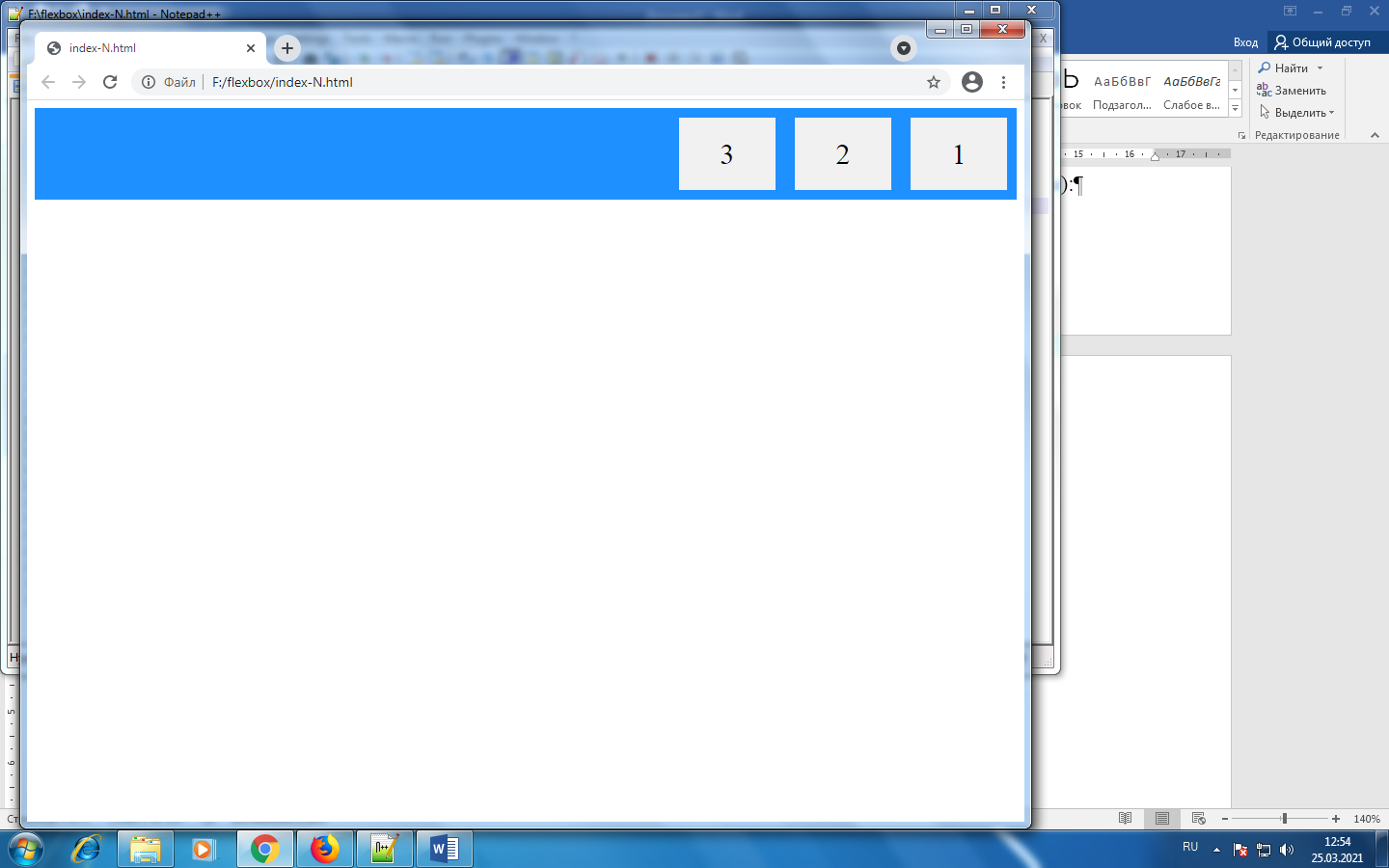
Значение row суммирует элементы Flex горизонтально (слева направо):

.flex-container {  
  display: flex;  
  flex-direction: row;  
}



Значение row-reverse суммирует элементы Flex горизонтально (но справа налево):

.flex-container {  
  display: flex;  
  flex-direction: row-reverse;  
}



## Свойство "выравнивание-содержимое"

Свойство **justify-content**используется для выравнивания элементов Flex:

Значение **center** выравнивает элементы Flex в центре контейнера:

.flex-container {  
  display: flex;  
  justify-content: center;  
}

## 

Значение **flex-start** выравнивает элементы Flex в начале контейнера (по умолчанию):

.flex-container {  
  display: flex;  
  justify-content: flex-start;  
}

## 

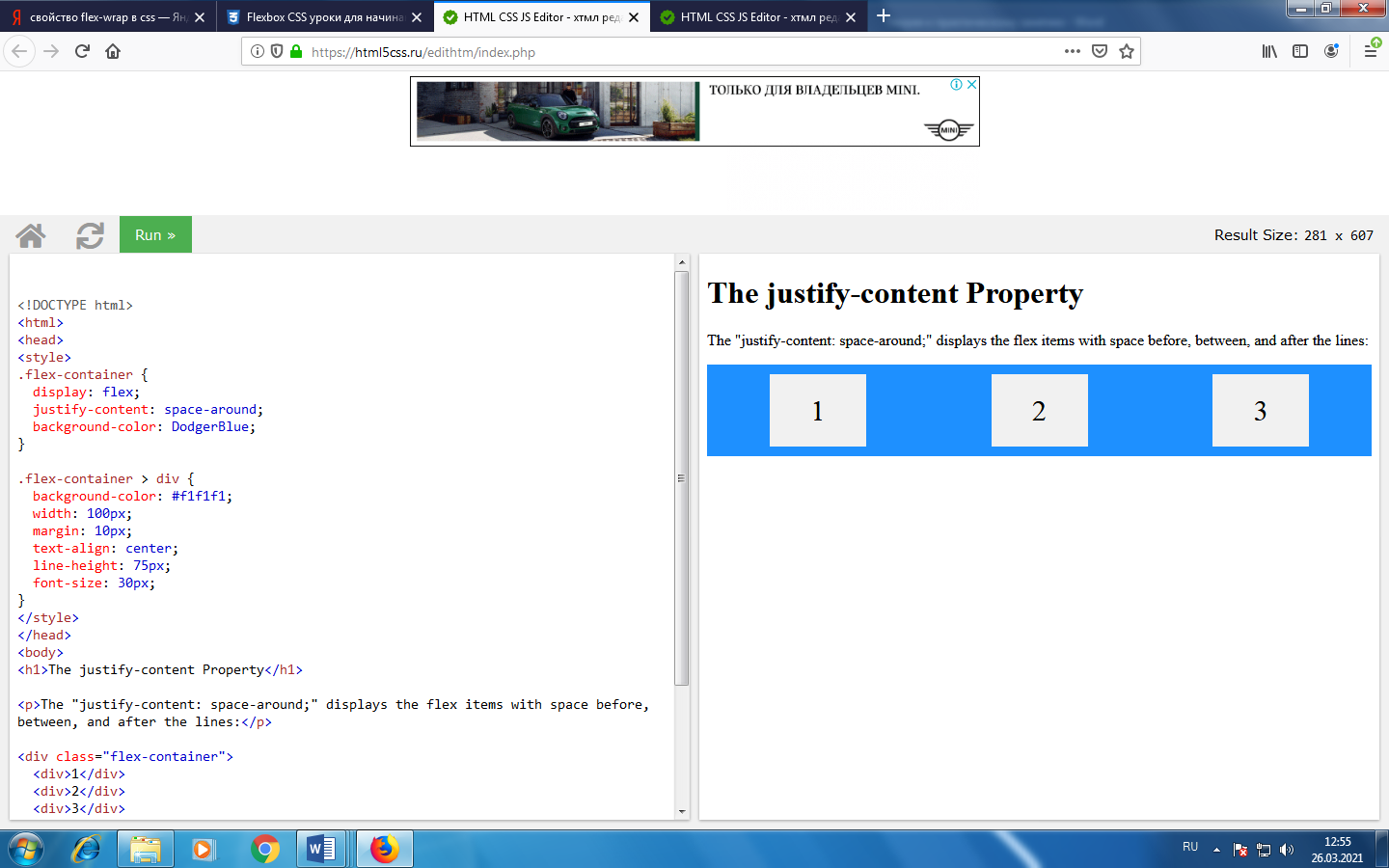
Значение **flex-end** выравнивает элементы Flex в конце контейнера:

.flex-container {  
  display: flex;  
  justify-content: flex-end;  
}

## 

Значение **space-around** отображает элементы Flex с пробелами до, между и после строк:

.flex-container {  
  display: flex;  
  justify-content: space-around;  
}



Значение **space-between** отображает элементы Flex с интервалом между строками:

.flex-container {  
  display: flex;  
  justify-content: space-between;  
}

## 

## Свойство Выравнивание-элементы

Свойство **align-items**используется для выравнивания элементов Flex по вертикали.

## .flex-container {   display: flex;   height: 200px;   align-items: center; }

## 

Значение **flex-start** выравнивает элементы Flex в верхней части контейнера:

.flex-container {  
  display: flex;  
  height: 200px;  
  align-items: flex-start;  
}

## 

Значение **flex-end** выравнивает элементы Flex в нижней части контейнера:

.flex-container {  
  display: flex;  
  height: 200px;  
  align-items: flex-end;  
}

## 

Значение **stretch** растягивает элементы Flex для заполнения контейнера (по умолчанию):

.flex-container {  
  display: flex;  
  height: 200px;  
  align-items: stretch;  
}

## 

Значение **baseline***Базовая линия* выравнивает элементы Flex, такие как Выравнивание базовых линий:

.flex-container {  
  display: flex;  
  height: 200px;  
  align-items: baseline;  
}

<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div>3</div>

<div><small>4</small></div>  
</div>

## \*<small>(от англ. small — маленький) уменьшает размер шрифта на единицу по сравнению с обычным текстом.

## 

## Свойство Flex-Wrap

Добавьте в HTMLследующие контейнеры:

<div class="flex-container">

<div>1</div>

<div>2</div>

<div>3</div>

<div>4</div>

<div>5</div>

<div>6</div>

<div>7</div>

<div>8</div>

<div>9</div>

<div>10</div>

<div>11</div>

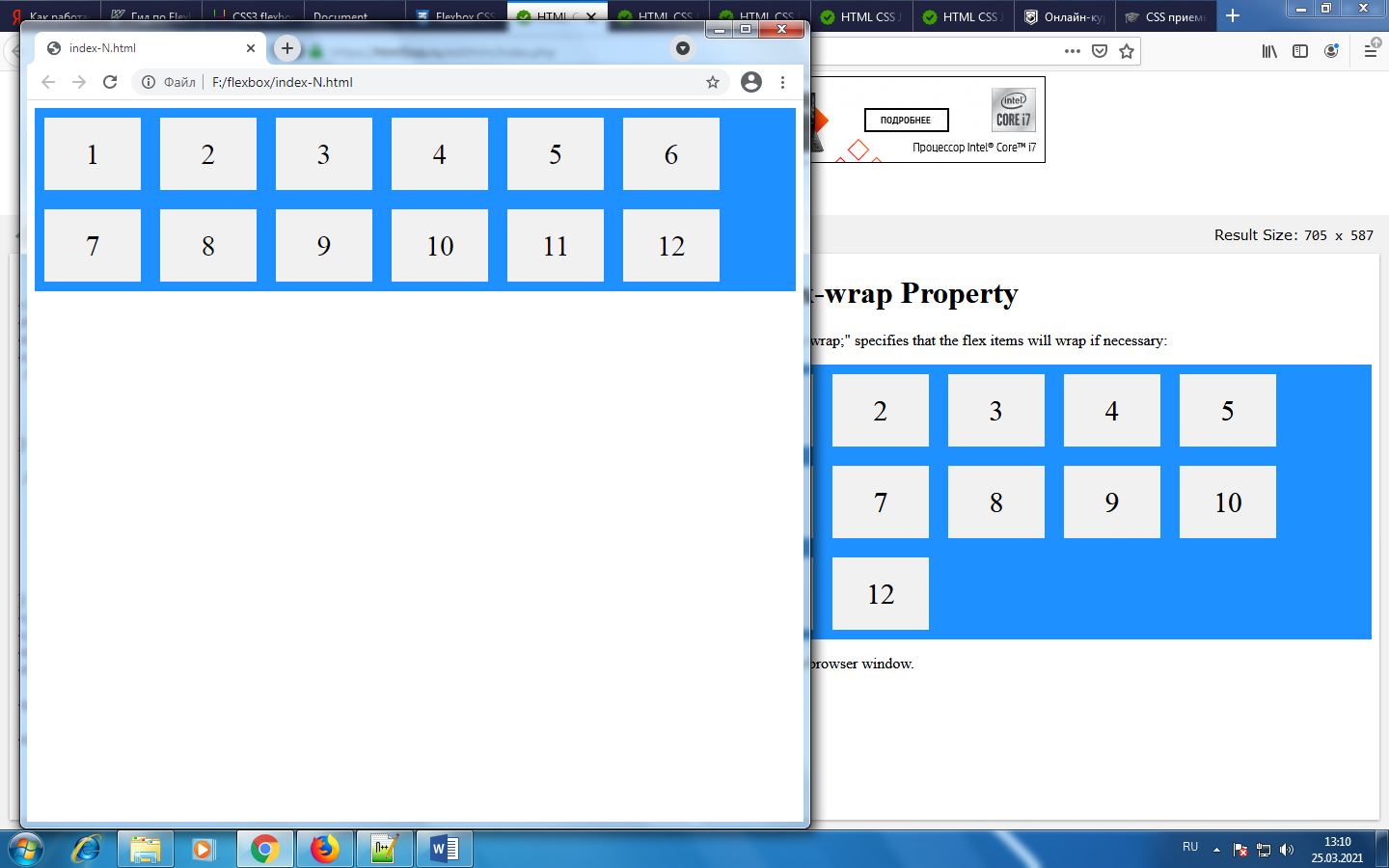
<div>12</div>

</div>

Свойство **flex-wrap** указывает, должны ли элементы Flex обернуть или нет.

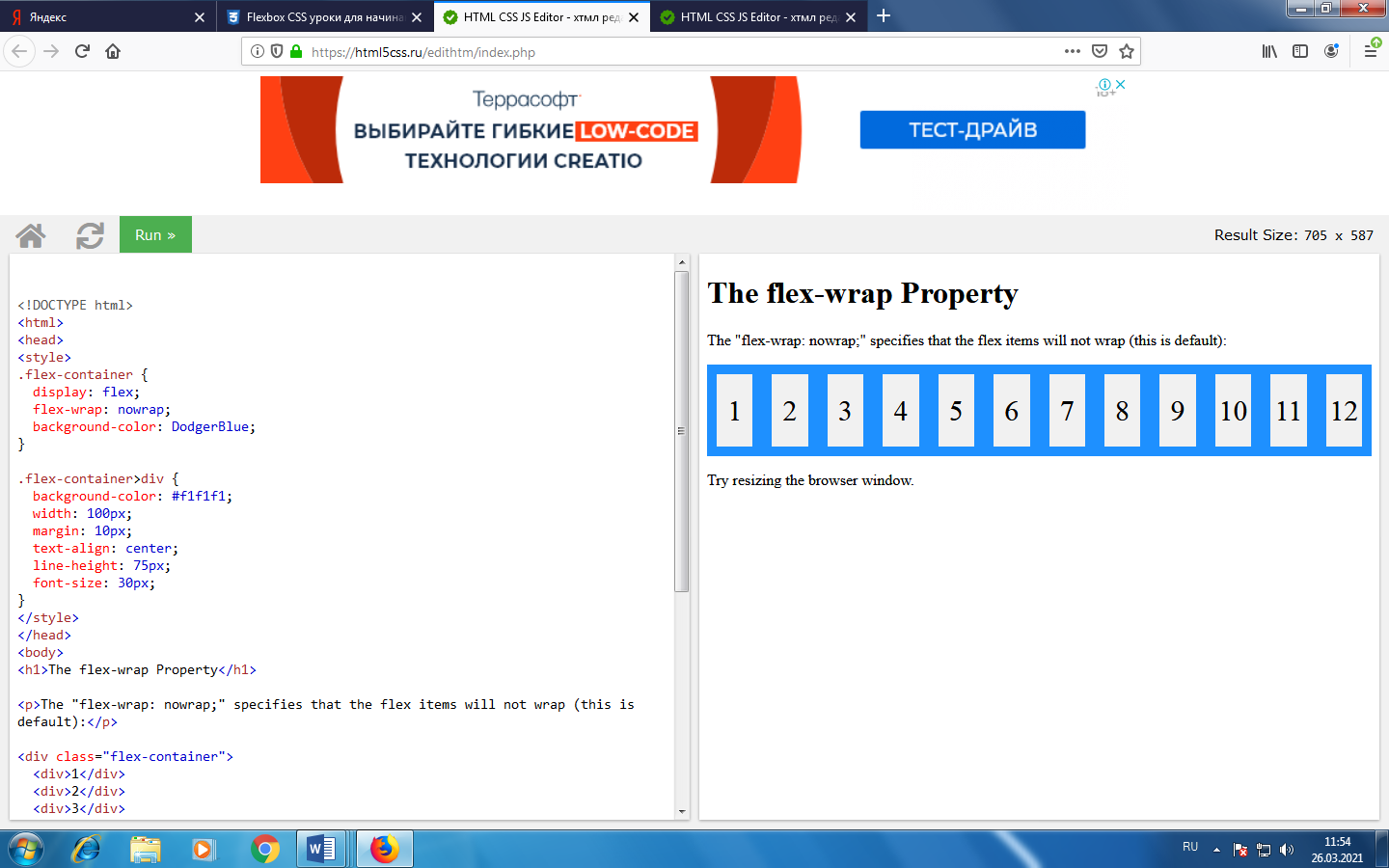
СвойствоCSS flex-wrap задаёт правила вывода flex-элементов — в одну строку или в несколько, с переносом блоков. Если перенос разрешён, то возможно задать направление, в котором выводятся блоки.

.flex-container {  
  display: flex;  
  flex-wrap: wrap;  
}



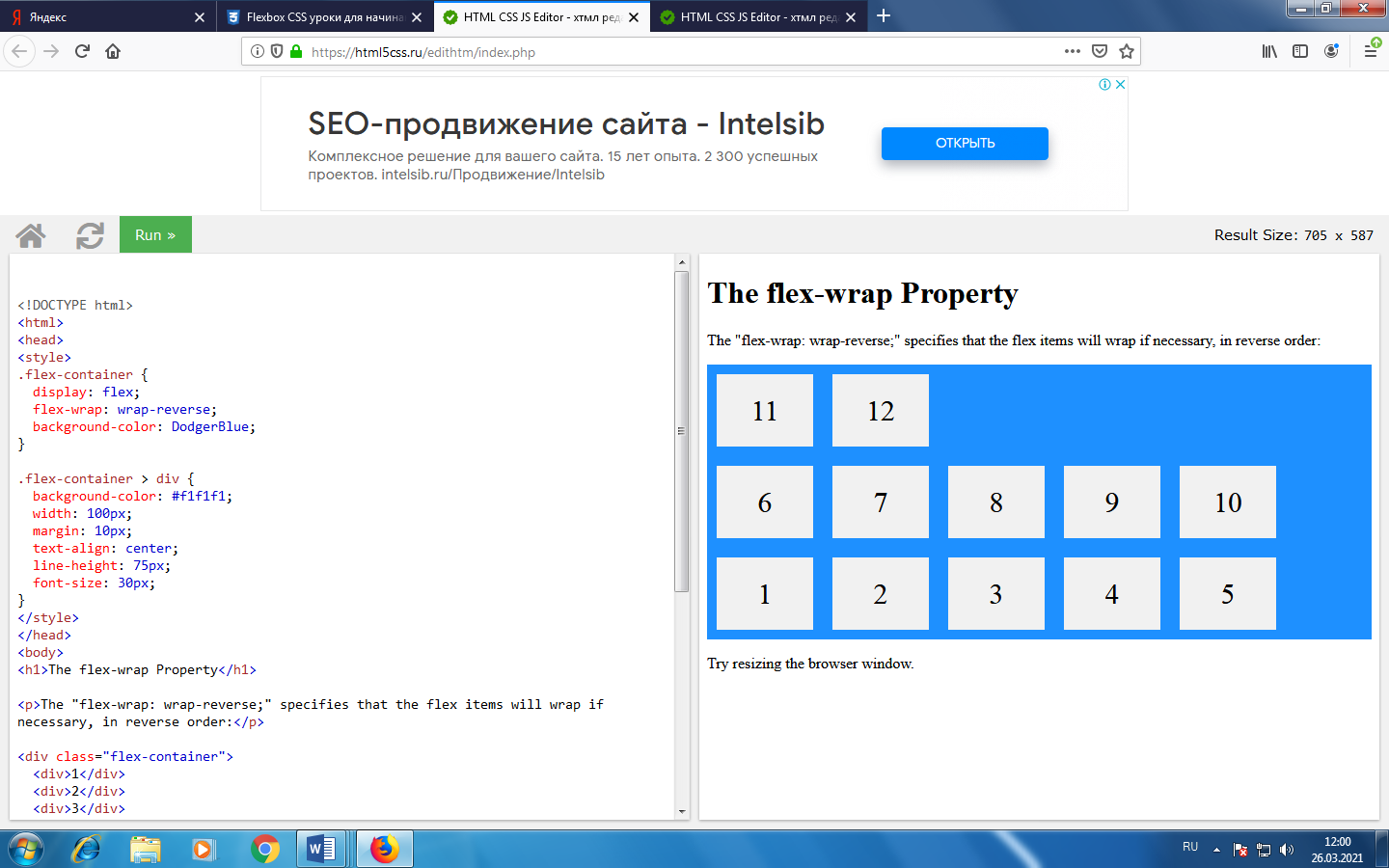
Значение **nowrap** указывает, что элементы Flex не будут обтекать (по умолчанию):

.flex-container {  
  display: flex;  
  flex-wrap:nowrap;  
}



Значение **wrap-reverse** указывает, что гибкие элементы будут при необходимости обернуты в обратном порядке:

.flex-container {  
  display: flex;  
  flex-wrap: wrap-reverse;  
}



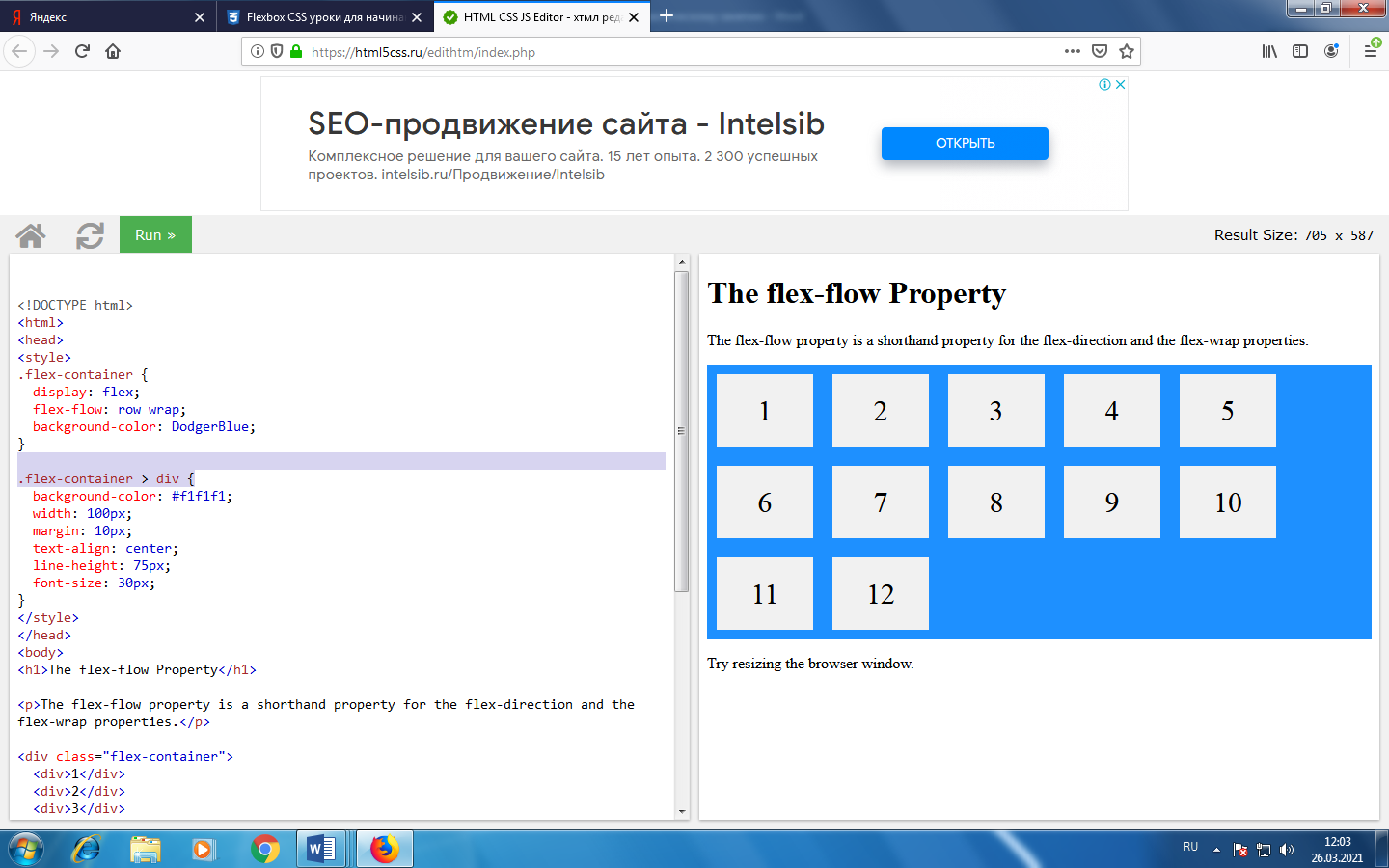
## Свойство Flex-Flow

Свойство **flex-flow** является сокращенным свойством для задания свойств **flex-direction** и **flex-wrap.**

CSS **свойство flex**-**flow** позволяет в одном объявлении указать значения **свойств flex**-direction (задает направление, в соответствии с которым располагаются флекс элементы внутри флекс контейнера), и flex-wrap (определяет, будет ли флекс контейнер однострочным, или многострочным)

### Пример

.flex-container {  
  display: flex;  
  flex-flow: row wrap;  
}



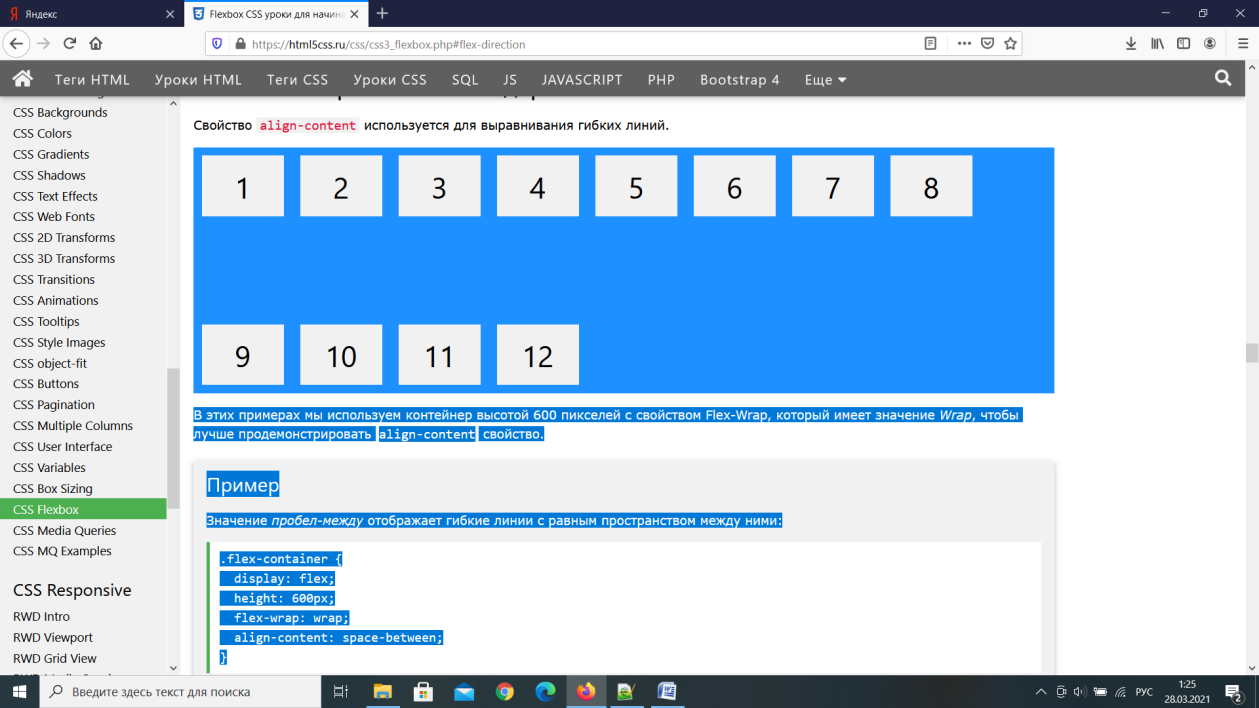
**Свойство выравнивания содержимого** space-between,space-around, stretch, center flex-start,flex-end

Свойство align-content используется для выравнивания гибких линий.

В этих примерах используется контейнер высотой 600 пикселей с свойством **Flex-Wrap**, который имеет значение **wrap**, чтобы лучше продемонстрировать **align-content** свойство.

Значение **space-between** отображает гибкие линии с равным пространством между ними:

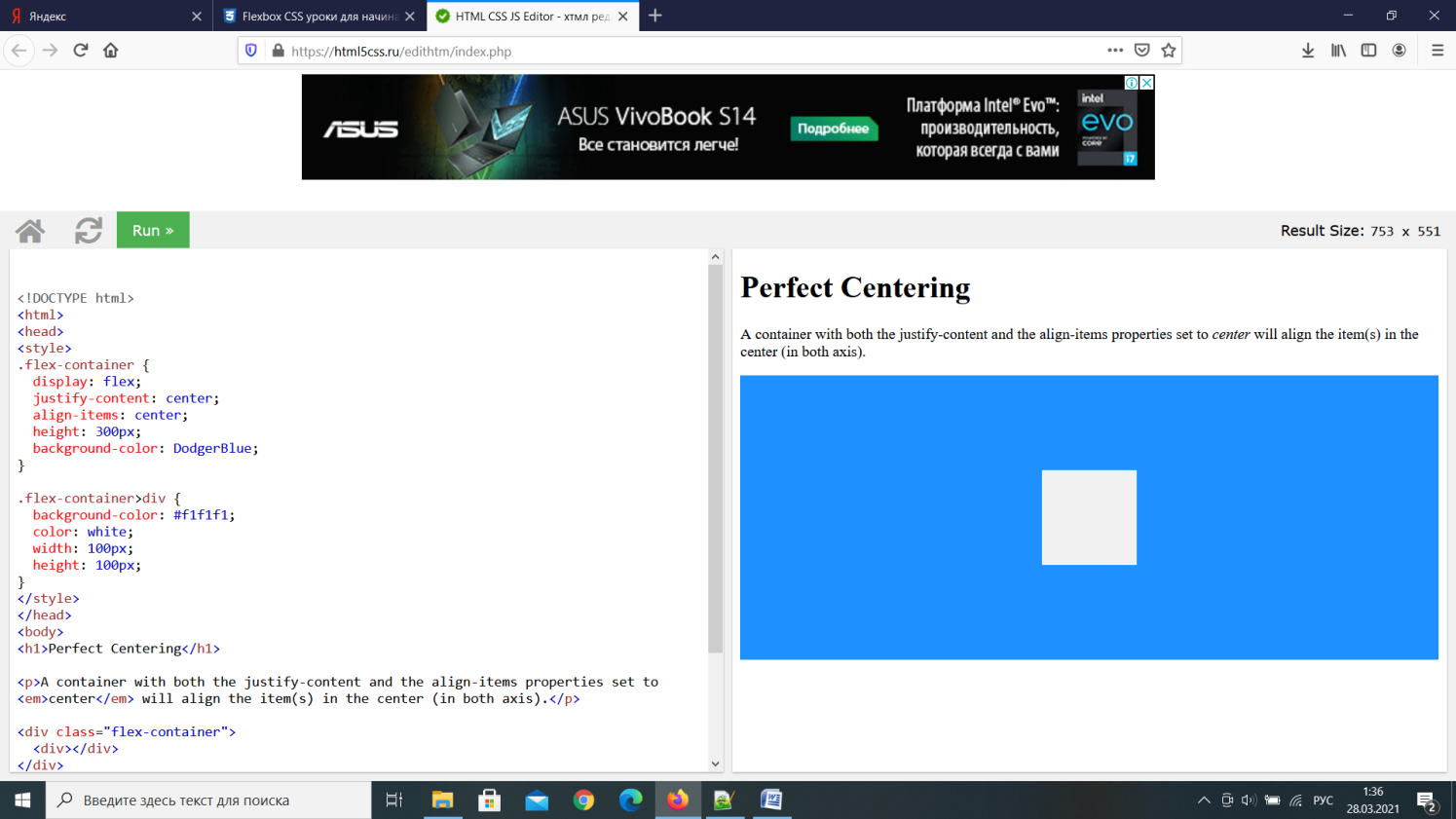
.flex-container {  
  display: flex;  
  height: 600px;  
  flex-wrap: wrap;  
  align-content: space-between;  
}



**Идеальное центрирование**

Задайте для свойств **justify-content** и **align-items** значение Center и элемент Flex будут идеально центрированы:

.flex-container {  
  display: flex;  
  height: 300px;  
**justify-content: center;  
  align-items: center;**}



## Дочерние элементы (элементы)

Прямые дочерние элементы контейнера Flex автоматически становятся гибкими (Flex) элементами.

<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div>3</div>   
  <div>4</div>  
</div>

.flex-container > div {

background-color: DodgerBlue;

color: white;

width: 100px;

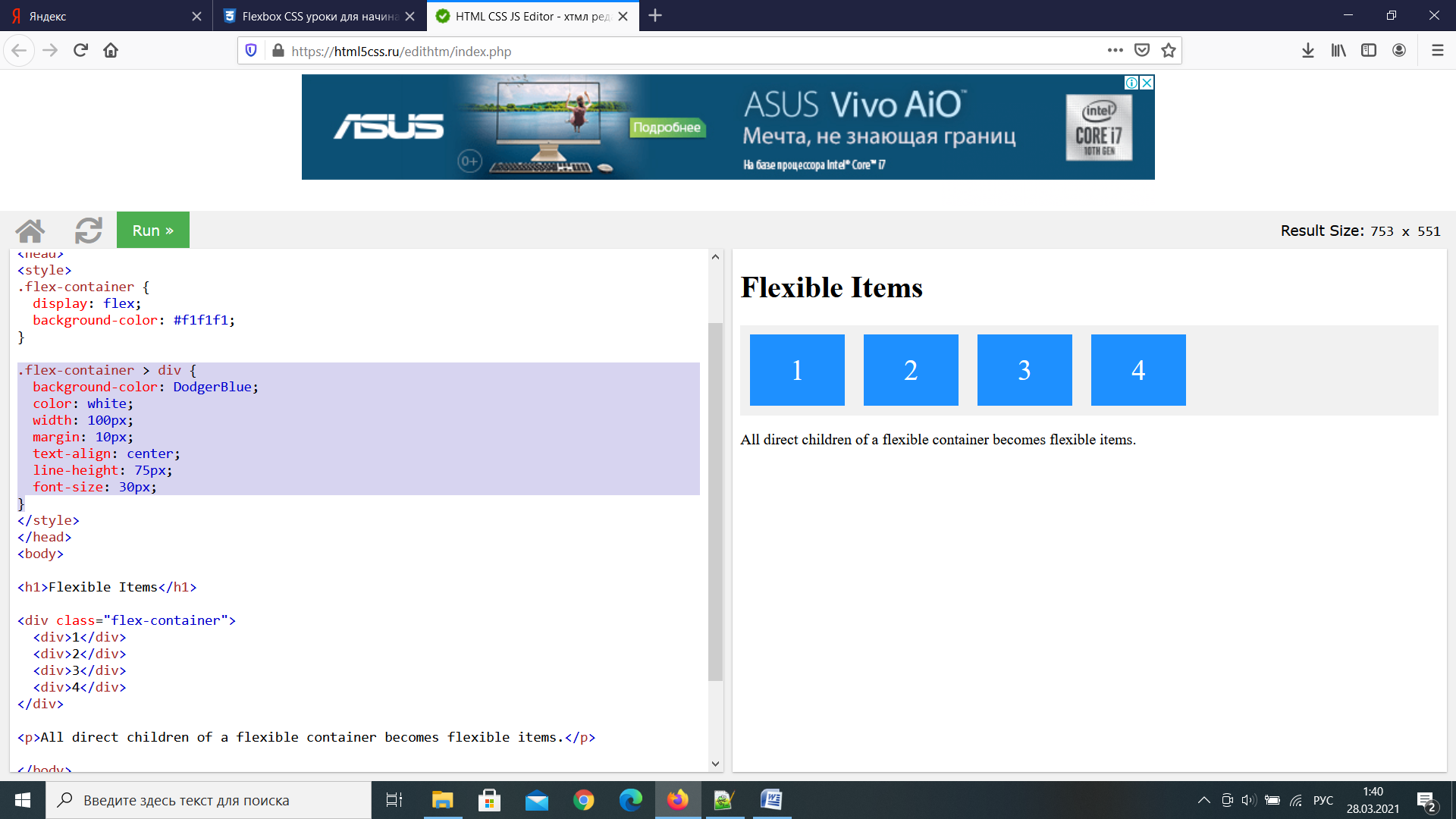
margin: 10px;

text-align: center;

line-height: 75px;

font-size: 30px;

}



**Свойства элемента Flex:**

* [order](https://html5css.ru/css/css3_flexbox.php#order)
* [flex-grow](https://html5css.ru/css/css3_flexbox.php#flex-grow)
* [flex-shrink](https://html5css.ru/css/css3_flexbox.php#flex-shrink)
* [flex-basis](https://html5css.ru/css/css3_flexbox.php#flex-basis)
* [flex](https://html5css.ru/css/css3_flexbox.php#flex)
* [align-self](https://html5css.ru/css/css3_flexbox.php#align-self)

**Свойство Order**

Первый элемент Flex в коде не должен отображаться в качестве первого элемента макета.

Значение Order должно быть числом, значением по умолчанию является 0.

Свойство Order может изменить порядок элементов Flex:

<div class="flex-container">  
  <div style="order: 3">1</div>  
  <div style="order: 2">2</div>  
  <div style="order: 4">3</div>   
  <div style="order: 1">4</div>  
</div>

.flex-container {

display: flex;

align-items: stretch;

background-color: #f1f1f1;

}

.flex-container>div {

background-color: DodgerBlue;

color: white;

width: 100px;

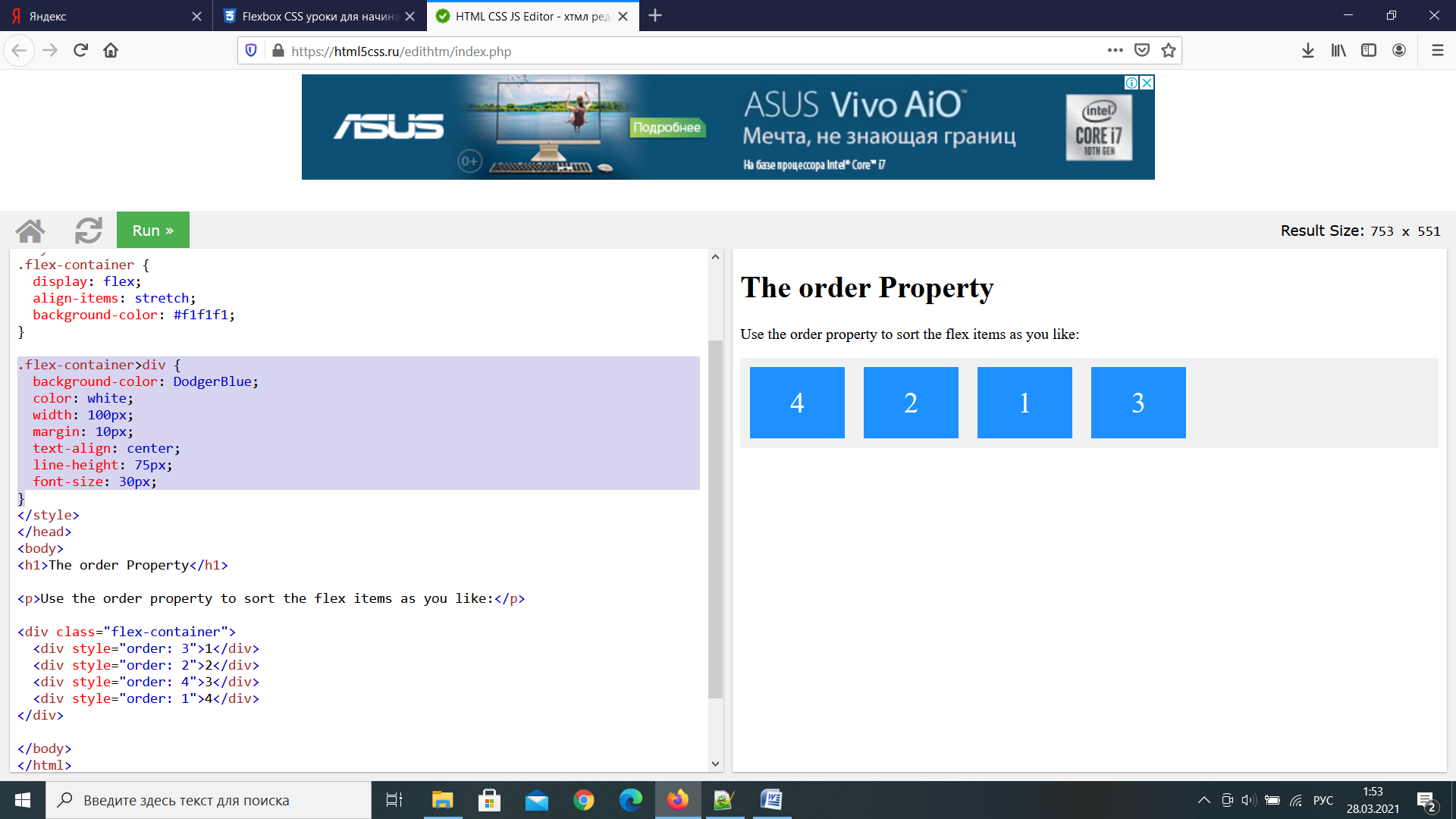
margin: 10px;

text-align: center;

line-height: 75px;

font-size: 30px;

}

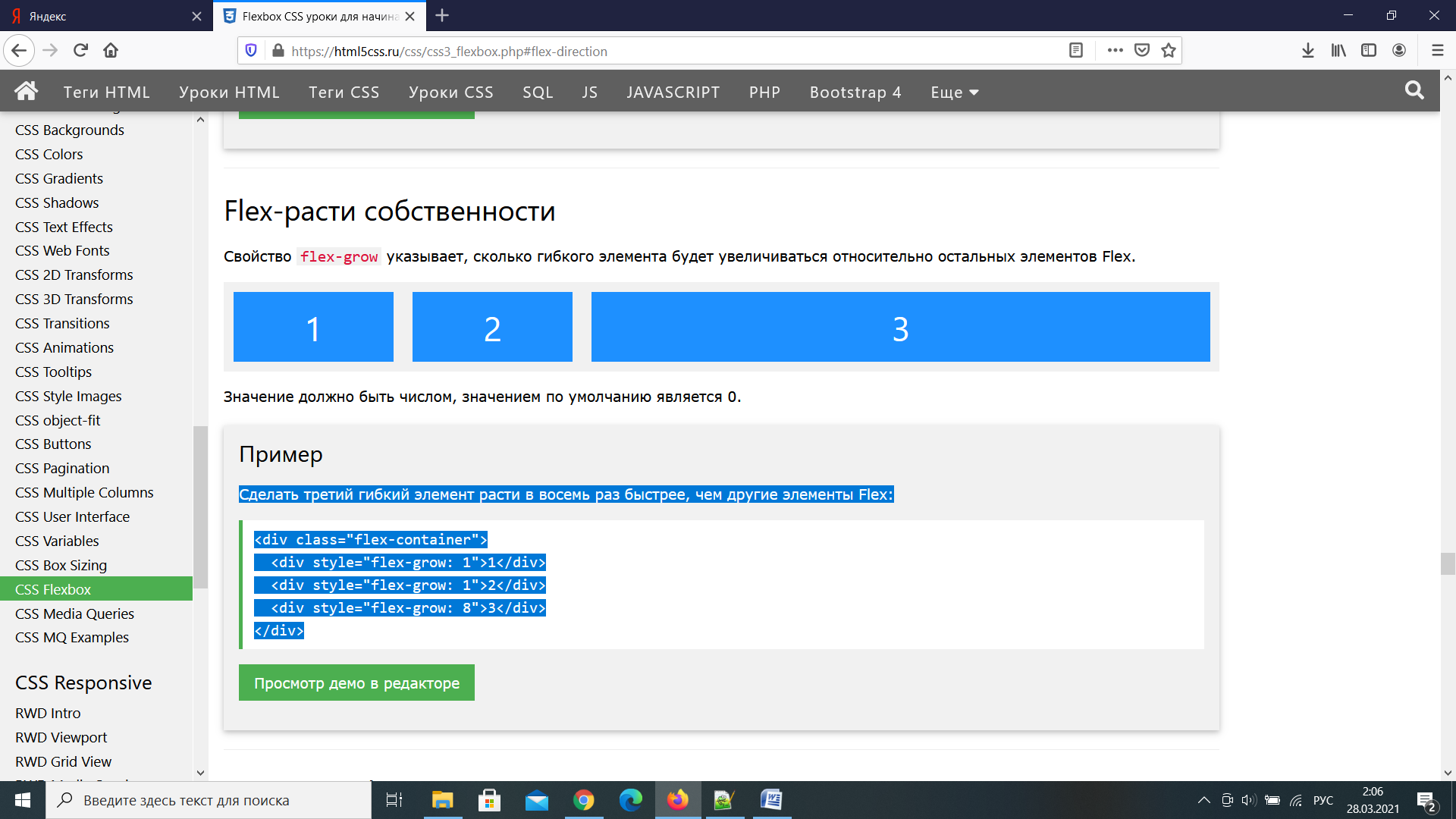


**Flex-расти собственности**

Свойство flex-grow указывает, сколько гибкого элемента будет увеличиваться относительно остальных элементов Flex.

Сделать третий гибкий элемент расти в восемь раз быстрее, чем другие элементы Flex:

<div class="flex-container">  
  <div style="flex-grow: 1">1</div>  
  <div style="flex-grow: 1">2</div>  
  <div style="flex-grow: 8">3</div>   
</div>

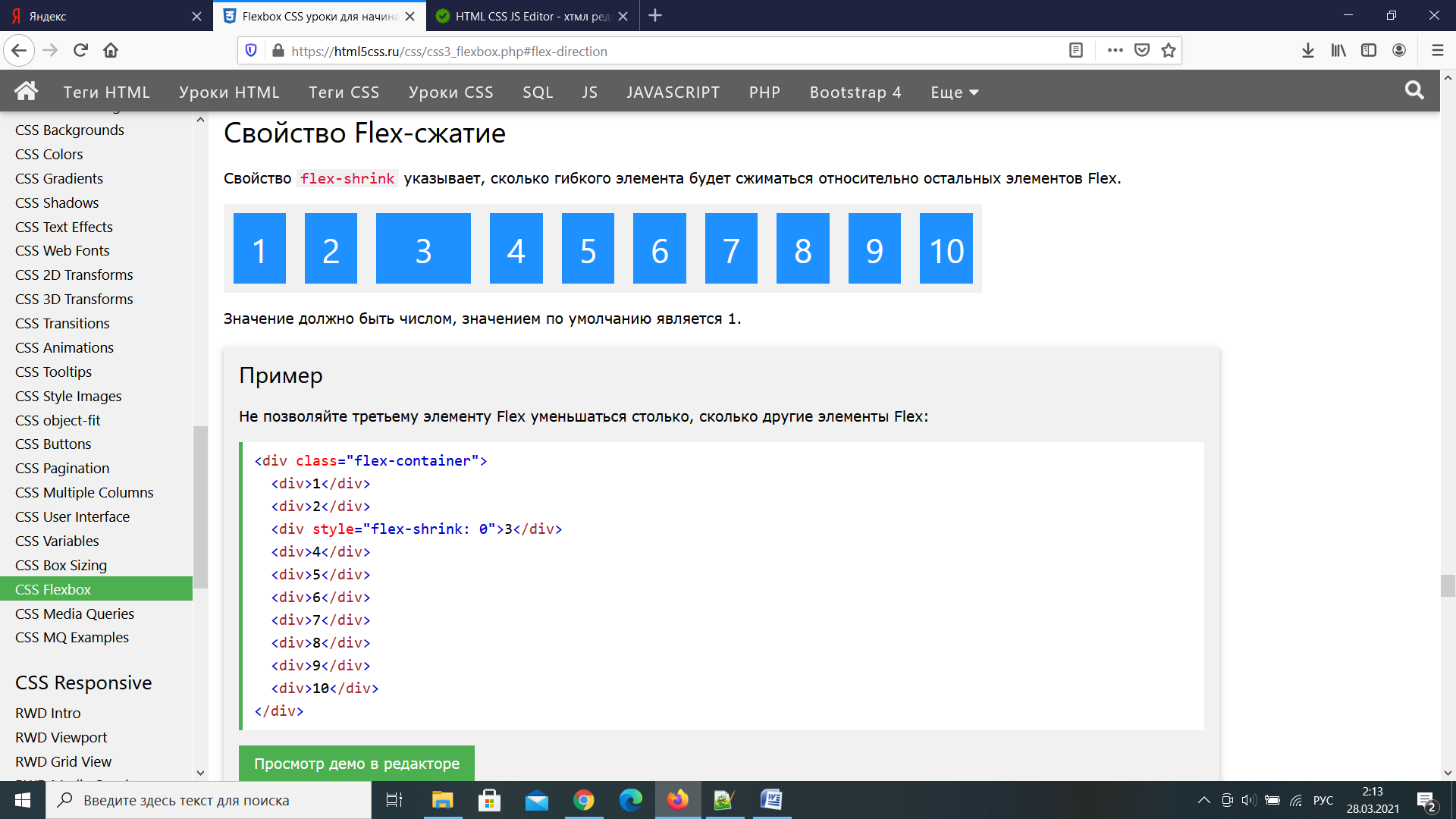


## Свойство Flex-сжатие

Свойство **flex-shrink** указывает, сколько гибкого элемента будет сжиматься относительно остальных элементов Flex.

Значение должно быть числом, значением по умолчанию является 1.

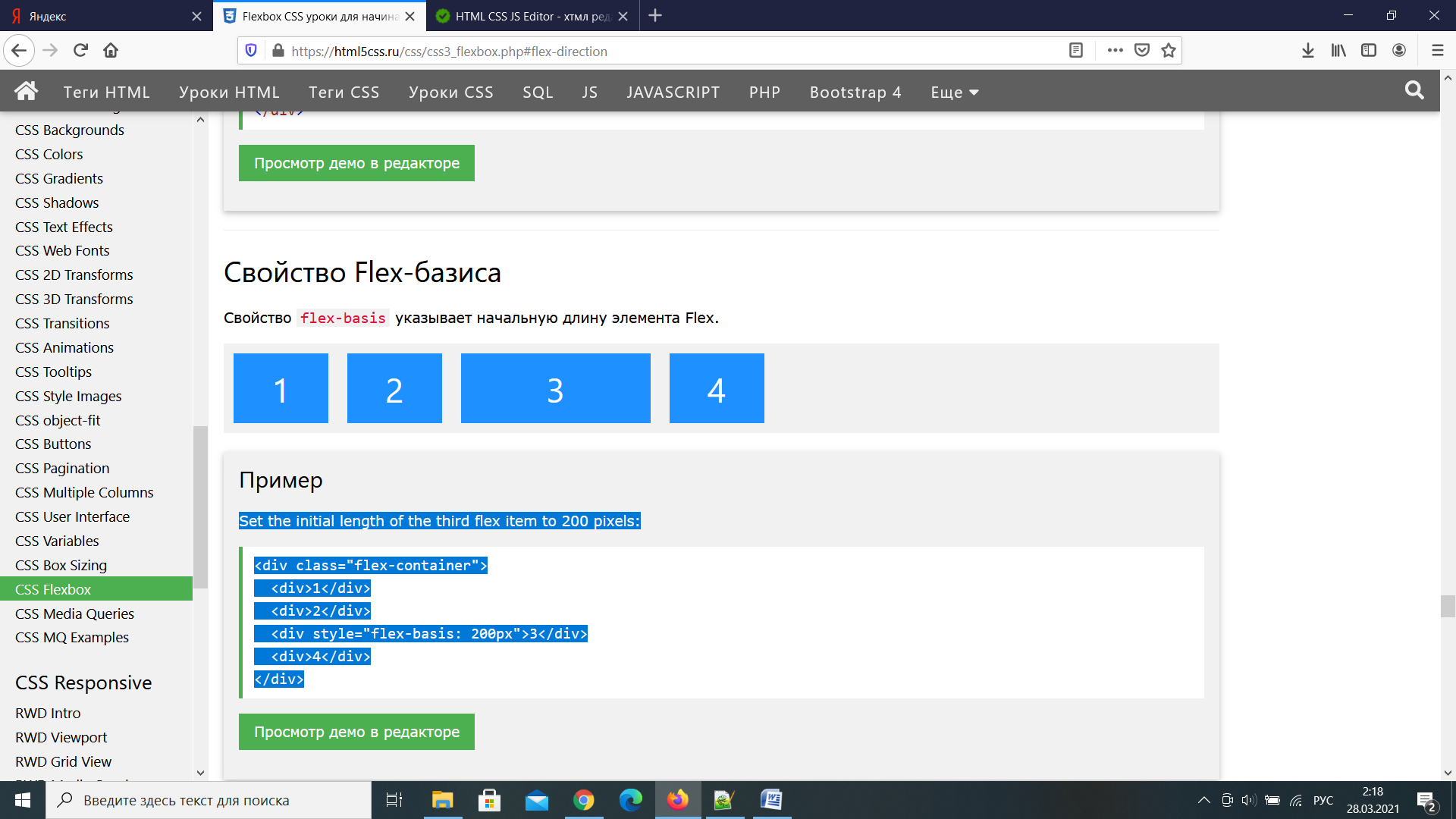
<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div style="flex-shrink: 0">3</div>  
  <div>4</div>  
  <div>5</div>  
  <div>6</div>  
  <div>7</div>  
  <div>8</div>  
  <div>9</div>  
  <div>10</div>  
</div>



**Свойство Flex-базиса**

Свойство flex-basis указывает начальную длину элемента Flex.

<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div style="flex-basis: 200px">3</div>  
  <div>4</div>  
</div>



## Свойство Flex

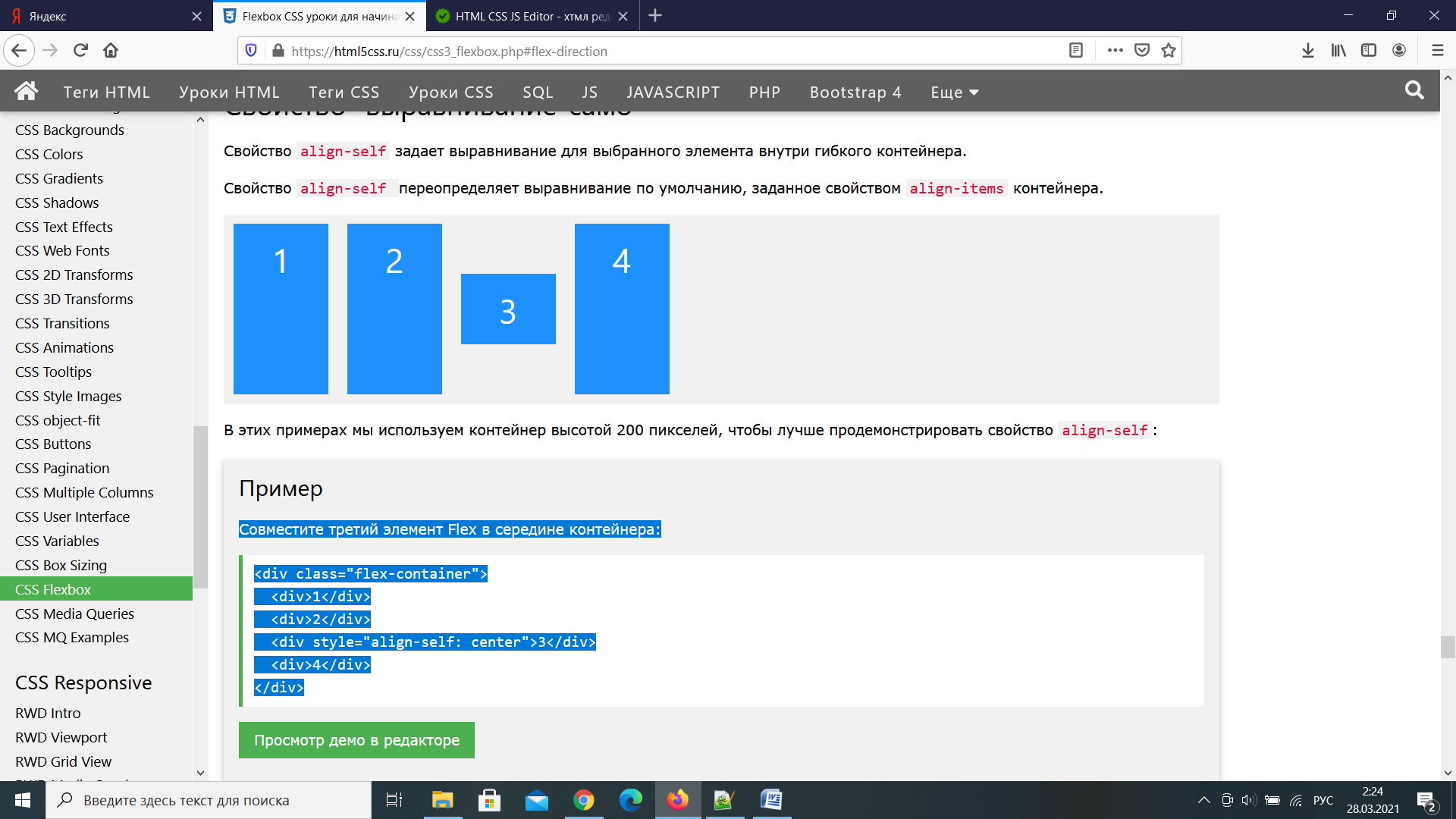
Свойство **flex** является сокращенным свойством для свойств flex-grow , flex-shrink и flex-basis.

<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div style="flex: 0 0 200px">3</div>  
  <div>4</div>  
</div>

**Свойство "выравнивание-само"**

Свойство align-self задает выравнивание для выбранного элемента внутри гибкого контейнера.

Свойство align-self переопределяет выравнивание по умолчанию, заданное свойством align-items контейнера.

Совместите третий элемент Flex в середине контейнера:

<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div style="align-self: center">3</div>  
  <div>4</div>  
</div>

Источник: <https://html5css.ru/css/css3_flexbox.php#flex-direction>

https://tpverstak.ru/flex-cheatsheet/