**Конспект**

**03\_Верстка сайтов (HTML5+CSS3)**

Тема 1. Настройка окружения

[**Научитесь стилизовать HTML с помощью CSS - Изучение веб-разработки | MDN**](https://developer.mozilla.org/ru/docs/Learn/CSS)

**CSS**

Каскадные (многоуровневые) таблицы стилей – **cascading style sheets (CSS)** – это мощный стандарт на основе текстового формата, определяющий представление данных в браузере.

/\*Это комментарий, который игнорируется при обработке\*/

Способы применения CSS свойств:

Встроенные стили: размещаются внутри элементов HTML -

<p style=”color: red”>Text Here</p>

Внутренние стили: размещаются в разделе заголовка веб-страницы –

<!DOCTYPE html>

<head>

<style type=”text/css>p {

color: red;

font-size: 14px;

}

</style>

</head>

Внешние стили: размещаются на внешней таблице стилей, которая представляет собой отдельную страницу, связанную с веб-страницей –

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

CSS, как и любой язык, имеет свой синтаксис. В нем есть правила — значения, которые определяют внешний вид элементов.

CSS-правило состоит из селектора, CSS-свойств и их значений:

* **Селекторы —** это метки, которые помогают браузеру понять, к какой части HTML-кода нужно применить заданные параметры.
* **CSS-свойства —** это определенные параметры оформления, например цвет элемента или текста (color) или цвет фона (background).
* **Значение —** это просто значение, оно выражается текстом или числом, например черный (black).

|  |  |
| --- | --- |
| *селектор { свойство: значение; }* | *p { color: black }* |

Селекторы, псевдоклассы и псевдоэлементы:

Псевдоэлементы - <https://blog.skillfactory.ru/psevdoelementy-v-css-chto-eto-i-zachem-oni-nuzhny/>

|  |  |
| --- | --- |
| \* | селектор всех элементов |
| section | селектор всех тегов с таким именем(по имени) |
| #id | селектор элемента с данным id |
| .class | селектор элементов с таким классом |
| [attribute=”value”] | селектор по атрибуту (значения атрибута) |

Селекторы можно комбинировать:

|  |  |
| --- | --- |
| .class1.class2 | селектор для элементов с одновременно двумя классами |
| a#id:visited | элемент тега <a> с данным id и псевдоклассом :visited |
| section p | элементы <p>, являющиеся потомками section |
| section > p | только непосредственные потомки section |
| section ~ p | все <p> на том же уровне вложенности, которые идут после section |
| section + p | первый <p> на том же уровне вложенности |

Псевдоклассы

для указания его места среди соседей:

|  |  |
| --- | --- |
| :first-child | первый потомок своего родителя |
| :last-child | последний потомок своего родителя |
| :only-child | единственный потомок своего родителя,  соседних элементов нет |
| :nth-child(a) | потомок номер a своего родителя, например :nth-child(2)– второй потомок. Нумерация начинается с 1 |
| :nth-child(an+b) | расширение предыдущего селектора через  указание номера потомка формулой, где a,b – константы, а под n подразумевается любое целое число. Этот псевдокласс будет  фильтровать все элементы, которые попадают под формулу при каком-  либо n. Например: -:nth-child(2n) даст элементы номер 2, 4, 6…, то есть чётные |
| :nth-child(2n+1) | даст элементы номер 1, 3…, то есть нечётные |
| :nth-child(3n+2) | даст элементы номер 2, 5, 8 и так далее |
| :nth-last-child(a)  :nth-last-child(an+b) | то же самое, но отсчёт начинается с конца, например :nth-last-child(2) – второй элемент с конца |
|  | |
| :not(селектор) | все, кроме подходящих под селектор. |
| :focus | в фокусе |
| :hover | под мышью |
| :empty | без детей (даже без текстовых) |
| :checked, :disabled, :enabled | состояния INPUT |
| :target | этот фильтр сработает для элемента, ID которого совпадает с анкором #... текущего URL |

Основное различие между псевдоклассами и псевдоэлементами заключается в том, как они влияют на элементы и их состояния.

|  |  |
| --- | --- |
| **Псевдоклассы** | **Псевдоэлементы** |
| Работают с состоянием и положением уже существующих элементов. | Работают с виртуальными частями элементов, которых нет в HTML-коде. |
| Применяются при взаимодействии пользователя. | Позволяют стилизовать определенные части элемента. |
| Не добавляют новый контент, а просто изменяют стиль элемента. | Создают новые виртуальные элементы, которые можно стилизовать. |
| Синтаксис с одним двоеточием. | Синтаксис с двойным двоеточием. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Оформление текста

правило @font-face:

<https://hcdev.ru/css/font-face/>

* позволяет авторам указывать онлайн-шрифты для отображения текста на своих веб-страницах;
* позволяет авторам предоставлять свои собственные шрифты;
* позволяет задать набор общих правил для текста.

CSS – свойства текста:

|  |  |
| --- | --- |
| font-family | устанавливает семейство шрифта, которое будет  использоваться для оформления текста содержимого. Список  шрифтов может включать одно или несколько названий,  разделённых запятой. Если в имени шрифта содержатся пробелы,  например, Comic Sans MS, оно должно заключаться в одинарные  или двойные кавычки |
| font-size | определяет размер шрифта элемента |
| font-style | определяет начертание шрифта — обычное, курсивное или наклонное. |
| text-decoration (оформление) | добавляет оформление текста в виде его подчёркивания, перечёркивания или линии над текстом |
| font-weight(начертание\_толщина) | устанавливает насыщенность шрифта. Значение указывается в виде чисел от 100 до 900 с шагом 100 или с помощью заданных ключевых слов (по умолчанию 400 или normal). |
| text-align(выравнивание) | определяет горизонтальное выравнивание текста в пределах элемента |

|  |  |
| --- | --- |
| white-space (пробел) | устанавливает, как отображать пробелы между словами. В обычных условиях любое количество пробелов в коде HTML показывается на веб-странице как один. **Исключением**  **является элемент <pre>**, помещённый в этот контейнер текст выводится со всеми пробелами, как он был отформатирован пользователем. Таким образом, white-space имитирует работу <pre>, но в отличие от него не меняет шрифт на моноширинный |
| letter-spacing | определяет интервал между символами в пределах элемента. |
| word-spacing(интервал) | устанавливает интервал между словами. Если для текста задано выравнивание через text-align со значением justify (выравнивание по ширине), то интервал между словами  будет установлен принудительно, но не меньше значения, указанного через word-spacing |
| line-height (высота) | для блочных элементов определяет  минимальную высоту строки текста. Для внедряемых строчных элементов (вроде <img>) свойство line-height не оказывает никакого эффекта. Для остальных строчных элементов line-height - задаёт высоту, которая используется для расчёта высоты строки блока |

|  |  |
| --- | --- |
| text-overflow (позволяет взять под контроль переполнение элемента ) | определяет параметры видимости текста в блоке, если текст целиком не помещается в заданную область.  Возможны два варианта: текст обрезается; текст обрезается и к  концу строки добавляется многоточие.  text-overflow работает в  том случае, если для блока значение свойства overflow установлено как auto, scroll или hidden. |
| text-shadow (тень) | добавляет тень к тексту, а также устанавливает её параметры:   * цвет тени, * смещение относительно надписи; * радиус размытия; |
|  |  |
|  |  |
|  |  |
|  |  |

Фон (background)

background – позволяет установить одновременно до пяти характеристик фона.

Свойство background позволяет установить фоновое изображение, цвет фона и другие свойства стиля для заданного элемента.

background также объединяет в себе несколько других свойств:

* background-color — задаёт цвет фона элемента;
* background-image — устанавливает фоновое изображение для элемента;
* background-position — определяет начальную позицию фонового изображения;
* background-repeat — указывает, должно ли фоновое изображение повторяться;
* background-size — устанавливает размер фонового изображения;
* background-attachment — определяет, будет ли фоновое изображение прокручиваться вместе с содержимым элемента.

|  |  |
| --- | --- |
| background-image | устанавливает фоновое изображение для  элемента.В случае наличия в рисунке прозрачных областей, через них будет проглядывать фоновый цвет. |
| background-color | определяет цвет фона элемента, цвет фона дочерних элементов совпадает с цветом фона родительского элемента. |
| background-size | масштабирует фоновое изображение  согласно заданным размерам. Значения могут быть: размером (пиксели, проценты, em и др.), auto, cover, contain. |
| background-position | задаёт начальное положение фонового  изображения, установленного с помощью свойства background-image. Допустимо указывать несколько значений для каждого  фона, перечисляя значения через запятую |
| background-repeat | определяет, как будет повторяться  фоновое изображение, установленное с помощью свойства background-image. Можно установить повторение рисунка только по горизонтали, по вертикали или в обе стороны. Допустимо указывать несколько значений для каждого фона, перечисляя значения через запятую |

|  |  |
| --- | --- |
| background-attachment | устанавливает, будет ли прокручиваться фоновое изображение вместе с содержимым элемента. Изображение может быть зафиксировано и оставаться  неподвижным, либо перемещаться совместно с документом.  Можно указать несколько значений для ряда фоновых изображений, перечисляя значения через запятую. |
| background-origin | определяет область позиционирования  фонового рисунка. Это свойство не применяется, когда значение background-attachment задано как fixed. |
| background-clip | определяет, как цвет фона или фоновая  картинка должна выводиться под границами. Эффект заметен при  прозрачных или пунктирных границах. Значения могут быть:  padding-box, border-box, content- box, text |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Позиционирование элементов

CSS свойства: **margin и** **padding**.

margin - внешний отступ от других элементов. Значение "auto" - это значение центрирует элемент по горизонтали.

padding внутренний отступ от содержимого.

У свойства padding существует проблема. Это свойство расширяет элемент на величину отступа. Чтобы такого не происходило, необходимо задать элементу свойство box-sizing в значении border-box (по умолчанию оно в значении content-box).

border-box - контент сожмётся;

content-box - ширина границ и внутренних отступов при отрисовке будет добавлена к финальной ширине, делая элемент шире.

Потоком документа в HTML называется порядок вывода

элементов на страницу. В обычном виде все блоки выводятся в

том порядке, в котором они записаны внутри HTML-документа.

Существует несколько разных типов позиционирования, которые вы можете применить к элементам HTML. Для активации специфического типа позиционирования у элемента, мы

используем свойство position.

Типы позиционирования:

Статическое позиционирование — это умолчание, которое получает каждый элемент, что всего лишь значит "поставить элемент в его нормальное положение в потоке макета документа ".

Относительное позиционирование — возможно модифицировать окончательное положение объекта, занявшего своё место в макете нормального потока, в том числе заставлять его перекрывать другие элементы на странице.

Абсолютное позиционирование — элемент больше не существует в нормальном потоке макета документа. Вместо этого он располагается на своём собственном слое отдельно от всего остального. Мы можем создавать изолированные функции пользовательского интерфейса, которые не влияют на макет других элементов страницы. Например, всплывающие информационные блоки и меню управления. Абсолютно позиционируемый элемент будет отображаться за пределами элемента <html> и будет расположен относительно исходного

окна просмотра.

Фиксированное позиционирование - обычно фиксирует элемент в месте относительно видимой части области просмотра. можете создать элементы пользовательского интерфейса, которые зафиксированы на месте, как постоянные меню навигации, которые всегда видимы вне зависимости от того,сколько прокручивается страница.

position: sticky - это гибрид относительной и фиксированной позиции, который позволяет позиционируемому элементу вести себя, как будто он относительно позиционирован, до тех пор, пока он не будет прокручен до определённой пороговой точки

(например, 10px от вершины окна просмотра), после чего он становится фиксированным. Это можно использовать, например, чтобы заставить панель навигации прокручиваться вместе со страницей до определённой точки, а затем задерживать в верхней части страницы.

Блочная верстка

Блочная модель – это способ отображения элементов браузерами, которые обрабатывают все теги как небольшие прямоугольные блоки. Для браузеров любой тег – это контейнер с

содержимым: текстом, изображениями, другими тегами и т.д.Основные понятия:

Блок – это обычная прямоугольная область, обладающая рядом свойств, таких как: рамка, поля и отступы.

Содержимым блока может быть что угодно – текст, картинки, списки, формы для заполнения, меню навигации и т.п.

Рамка (border) – это контур, для которого можно задать такие характеристики как толщина, цвет и тип (пунктирная, сплошная, точечная).

Поля (padding) – отделяют содержимое блока от его рамки, чтобы текст, например, не был «впритык» к стенкам блока.

Отступы (margin) – это пустое пространство между различными блоками, позволяющее на заданном расстоянии расположить два блока относительно друг друга.

Блочная верстка сайта включает в себя блоки. Блоки, как и таблицы – это элементы, всегда располагающиеся на странице вертикально. То есть, если в коде страницы записаны подряд два блока, то отобразятся они в браузере один под другим. Если нам нужно расположить несколько блоков горизонтально, то в их свойствах задается такой параметр как «обтекание» (float).

float - расположить несколько блоков горизонтально.

Свойство display - определяет тип отображения (display type) элемента. Имеет два основных свойства:

* внешний тип отображения определяет расположение блока в схеме потока (flow layout);
* внутренний тип отображения определяет расположение дочерних элементов блока;

Параметр как display:inline-block (придает тегу одновременно свойства блочного и строчного элемента):

* по отношению к соседним веб-элементам он будет вести себя в качестве строчного (рядом с ним могут находиться другие теги в строку);
* для вложенных в него элементов он будет блочным (возможность определять его размеры и отступы);.

Виды верстки сайтов

Фиксированная или статическая верстка:

Вне зависимости от размеров окна браузера или устройства ширина страниц будет постоянной. Все элементы занимают строго определенную ширину в пикселях на странице. Если разрешение экрана большое и окно браузера развернуто на весь экран, то, как правило, по бокам остается свободное место. И,наоборот, на мобильных устройствах при фиксированной верстке снизу на странице появляется полоса горизонтальной прокрутки.

Такая верстка считается устаревшей.

Резиновая или гибкая верстка:

Страница занимает всю ширину браузера, каких бы размеров она не была и на каком бы устройстве вы не открыли эту страницу. Ширина элементов страницы задается в процентах от ширины окна, и поэтому занимают всю доступную область. При этой верстке тяжело добиться хорошего удобного дизайна при всех возможных разрешениях экранов.

Адаптивная или отзывчивая верстка:

Ключевая особенность в том, что страницы хорошо адаптируются под любое разрешение экрана пользователя. Это достигается тем, что используют несколько таблиц стилей под разные разрешения.

CSS Flexible Box Layout Module

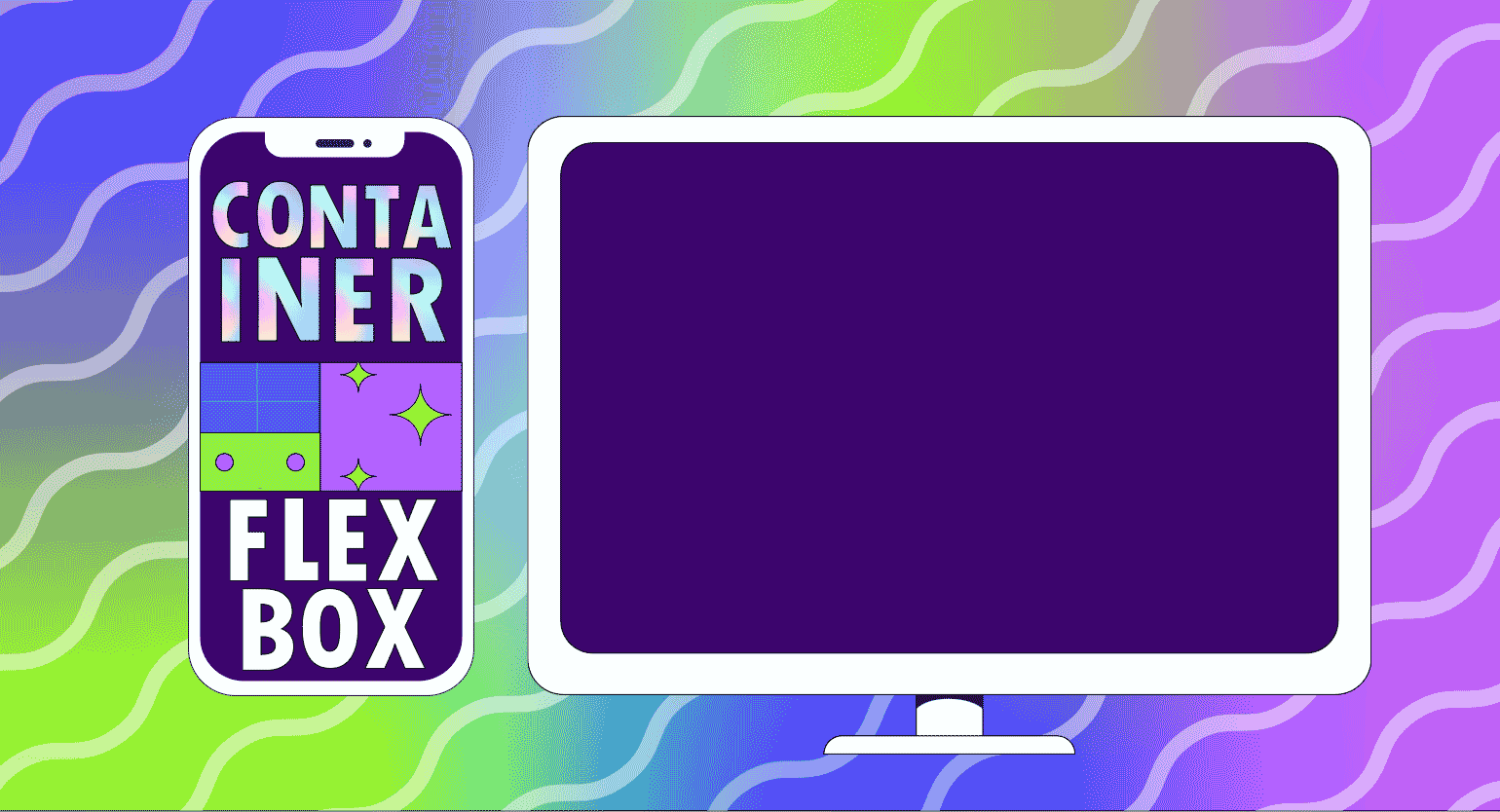
Flexbox («флексбокс») – это CSS модуль для макетов с гибкими блоками,предоставляет возможность при помощи CSS3 быстро и удобно позиционировать блоки на странице, подстраиваясь под разные варианты разрешения экрана, представляет собой гибкий вид

верстки.

Flexbox включает в себя ряд свойств и их значений, как для родителя элемента, так и для дочерних элементов.

Сильная сторона Flexbox заключается в его адаптивности под направление текста. Гибкие блоки сжимаются и растягиваются, выстраиваются в ряд или образуют столбцы, выравниваются как по вертикали, так и по горизонтали, причем их очередностью можно управлять через CSS. При этом, их размер может быть заранее неизвестен или постепенно меняться — отсюда и название flex, означающее «гибкий». Корректируя ориентацию и размерэлементов, контейнер Flexbox каждый раз подстраивается под требуемые условия.

Главная и поперечная ось flex-контейнера, flex-direction

[](https://skillbox.ru/media/code/shpargalka-po-flexbox-svoystva-primery-ispolzovaniya-i-besplatnye-trenazhyery/#stk-4)Прежде всего, это две оси, на которые нужно ориентироваться при позиционировании — главная и поперечная.

**Основные понятия**

Во Flexbox есть два вида свойств:

* одни применяются к flex-контейнеру,
* другие — к элементам, которые в нём расположены.

**Flex-контейнер** — это «коробка», в которой хранятся flex-элементы (flex item).

Чтобы превратить элемент во flex-контейнер, нужно установить ему свойство

* display: flex
* display: inline-flex.

Разница между **flex** и **inline-flex** в том, что в первом случае контейнер будет занимать всю ширину экрана (как блочный элемент), а во втором — лишь пространство, занимаемое его содержимым.

**Главная ось (main axis)** — направление, в котором располагаются flex-элементы.

**Поперечная ось (cross axis)**— ось, перпендикулярная главной оси.

**Обратите внимание:** направление главной и поперечной осей можно изменить с помощью **свойства flex-direction** .

**Главный размер (main size)** — размер, соответствующий направлению главной оси.

**Начало главной оси (main start)**— точка, в которой начинается последовательность flex-элементов, расположенных по главной оси.

**Конец главной оси (main end)**— точка, в которой заканчивается последовательность flex-элементов, расположенных по главной оси.

**Поперечный размер (cross size)**— размер, соответствующий поперечной оси.

Свойства Flexbox

flex-direction - **определяет направление** основной и поперечной осей в контейнере.

Содержит (может принимать) четыре возможных значения:

* row (по умолчанию) — элементы располагаются в строку, слева направо;
* row-reverse — элементы располагаются в строку, справа налево;
* column — элементы располагаются в столбец, сверху вниз;
* column-reverse — элементы располагаются в столбец, снизу вверх.

Запись: flex-direction: row;

flex-wrap - определяет, будут ли flex-элементы переноситься на другую строку, если им не хватит места в ряду.

* nowrap (по умолчанию) — элементы не переносятся на новую строку, а сжимаются;
* wrap — элементы переносятся на новую строку;
* wrap-reverse — элементы переносятся на новую строку в обратном порядке.

flex-flow объединяет в одно свойство flex-direction и flex-wrap, позволяя задать их значения в одной строке.

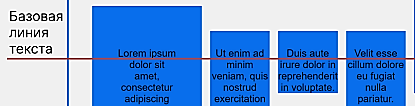
Обратите внимание: при использовании flex-flow важно соблюдать правильный порядок: сначала задать значение для flex-direction, потом для flex-wrap, иначе свойство работать не будет.

justify-content - определяет, **как будут распределены** элементы вдоль главной оси.

* flex-start (по умолчанию): выравнивает элементы по началу главной оси;
* flex-end: выравнивает элементы по концу главной оси;
* center: центрирует элементы вдоль главной оси;
* space-between: распределяет элементы по ширине, с одинаковым пространством между ними. Не добавляет отступы между крайними дочерними элементами и границей родителя;
* space-around: добавляет пространство вокруг каждого элемента;
* space-evenly: добавляет пространство не только между элементами, но и между ними и границей контейнера (см. иллюстрации ниже).

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

align-items - **выравнивает содержимое контейнера** вдоль по поперечной оси.

* flex-start: элементы выравниваются по верхнему краю контейнера;
* flex-end: элементы выравниваются по нижнему краю контейнера;
* center: содержимое контейнера выравнивается по центру;
* baseline: элементы выравниваются по базовой линии текста, который в них содержится;
* stretch: внутренние элементы растягиваются на всю высоту flex-контейнера.

align-content действует подобно justify-content — только **распределяет элементы относительно поперечной оси** и работает с **многострочными контейнерами.**

|  |  |
| --- | --- |
| .container {  height: 600px;- установим фикс. высоту  flex-wrap: wrap;  display: flex;  align-content: flex-start; | /\* align-content: flex-end; \*/  /\* align-content: center; \*/  /\* align-content: space-between; \*/  /\* align-content: space-around; \*/  /\* align-content: stretch; \*/  } |

**gap, row-gap, column-gap**

**gap** — это **разрыв между отдельными элементами**, строками или колонками внутри flex-контейнера. Значение по умолчанию — none; задаётся пикселями (px) или процентами.

* **gap: … px; -** устанавливает разрыв между строками и колонками;
* **row-gap**— устанавливает разрыв между строками;
* **column-gap**— устанавливает разрыв между колонками;
* **gap** — устанавливает разрыв между строками и колонками. При этом, если задать только значение, оно применяется и к строкам, и к колонкам.

Свойства устанавливаются в стилях контейнера и действуют на всё его содержимое.

Свойства flex-элементов

Flexbox также можно управлять **отдельными flex-элементами**.

**order** устанавливает **порядок расположения элементов** во flex-контейнере относительно друг друга.

В качестве значения order принимает любое целое число: чем оно меньше, тем ближе к началу главной оси будет расположен элемент. По умолчанию у каждого элемента значение **order равно 0.**

|  |  |
| --- | --- |
| .container {  display: flex;  }  .item-1 {  order: 2;  }  .item-2 {  order: 1;  } | Классу присваивается ордер и элемент класса перемещается ближе к началу главной оси в зависимости от числа.  **Поменять местами …** |

**flex-grow** - позволяет указать, **какую долю свободного пространства** во **flex-контейнере** может занимать элемент, или, другими словами, в **какой степени он может «расти» вдоль главной оси**. В качестве значения можно задать любое неотрицательное число, по умолчанию это 0.

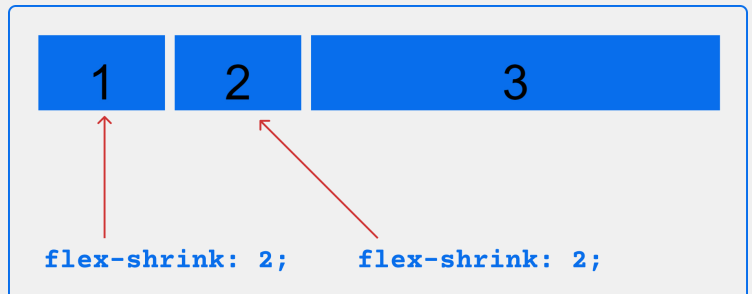
**flex-grow: 0; -** неотрицательное число



**flex-shrink** - в противоположность **flex-grow**, позволяет указать, насколько сильно **элемент должен сжиматься**, если суммарная ширина элементов превышает размер контейнера. Значение может быть любым положительным числом.

По умолчанию - **flex-shrink: 1;**

**flex-shrink: 0;** - будет сжиматься, насколько это возможно, с **учётом значений flex-shrink других элементов**.



**flex-basis** - задаёт **размер элемента по главной оси**, перед тем как будет распределено оставшееся пространство. (…px …% … с помощью ключевых слов)

* **auto** (по умолчанию) — в зависимости от значения flex-direction (направление оси) примет значение свойства width или height. Если же соответствующее свойство тоже установлено в auto, то элемент примет размер содержимого;
* **max-content**— устанавливает предпочтительную ширину;
* **min-content** — устанавливает минимальную ширину;
* **fit-content** — устанавливает максимально возможный размер элемента, ограниченный значениями max-content и min-content.

**Обратите внимание:** flex-basis действует подобно свойству width, только имеет более высокий приоритет. Поэтому если для элемента будут заданы значения обоих свойств, то width будет игнорироваться.

**align-self** - устанавливает **выравнивание** отдельного flex-элемента **вдоль поперечной оси**, переопределяя значение свойства align-items.( выравнивает содержимое контейнера)

Значения align-self задаются для каждого flex-элемента и могут быть следующими:

* **flex-start** — выравнивает элемент по началу поперечной оси (cross start);
* **flex-end** — выравнивает элемент по концу поперечной оси (cross end);
* **center** — выравнивает элемент по центру поперечной оси;
* **stretch** (по умолчанию для flex-элементов) — элементы растягиваются на всю высоту контейнера.

|  |  |
| --- | --- |
|  |  |

<https://skillbox.ru/media/code/shpargalka-po-flexbox-svoystva-primery-ispolzovaniya-i-besplatnye-trenazhyery/#stk-4>

Адаптивная верстка

**Адаптивная верстка** — это такой способ создания веб-страниц, при котором они автоматически подстраиваются под размеры и ориентацию экрана устройства, а их дизайн

варьируется в зависимости от действий пользователя.

Цель адаптивной верстки — добиться того, чтобы сайт оставался удобным при загрузке на разных устройствах.

**Медиазапросы (media queries)** – это правила CSS, которые **позволяют управлять стилями элементов** в зависимости от значений технических параметров устройств. Иными словами, это конструкции, которые позволяют определять на основании некоторых условий, какие стили необходимо использовать на веб-странице, а какие нет.