

HyperText Markup Language

HTML & CSS



Урок 3

Списки. CSS отступы и поля

Оглавление

Списки. CSS отступы и поля	4
Списки.....	4
Упорядоченный (нумерованный) список.....	4
Маркированный список.....	10
Список определений	12
Многоуровневые, или вложенные списки	16
CSS-стили для списков	21
Использование псевдоэлемента ::before для стилей списков	28
Что такое псевдоэлементы ::before и ::after?	29
Оформление многоуровневых списков.	
Вложенные селекторы	32
CSS-свойства для управления отступами и полями	41
Внутренние и внешние отступы.	
Общая информация	41

Внешние отступы — margin.....	42
Внутренние отступы — padding	47
Значения отступов по умолчанию.....	49
Домашнее задание.....	54
Внешний вид заданий.....	55

- В тексте указываются ссылки на файлы примеров, их вы можете найти в архиве прикрепленном к PDF-файлу данного урока.

Списки. CSS ОТСТУПЫ И ПОЛЯ

Списки

На множестве страниц в интернете используют перечисление каких-либо вариантов услуг или товаров. Для такого перечисления отлично подходят списки. В HTML их существует 3 вида: списки определений, упорядоченные (нумерованные) и неупорядоченные (маркированные). С помощью списков можно упорядочить и систематизировать различные виды данных и представить их в удобном для пользователя виде.

Упорядоченный (нумерованный) список

Упорядоченный (нумерованный) список представляет собой блок с элементами, каждый из которых по умолчанию имеет номер в виде арабской цифры:

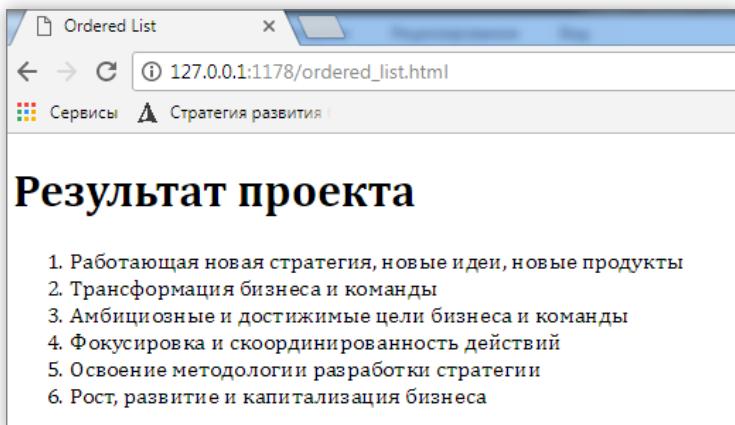


Рисунок 1

В html-разметке нумерованный список формируется с помощью тега `` (от англ. *ordered list*) и вложенных в него тегов `` (от англ. *list item*)

```
<ol>
  <li> Работающая новая стратегия, новые идеи,
    новые продукты </li>
  <li> Трансформация бизнеса и команды </li>
  <li> Амбициозные и достижимые цели бизнеса
    и команды </li>
  <li> Фокусировка и скоординированность
    действий </li>
  <li> Освоение методологии разработки
    стратегии </li>
  <li> Рост, развитие и капитализация бизнеса </li>
</ol>
```

Файл примера — *ordered_list.html*.

Вместо арабских цифр мы можем указать в качестве нумерации римские цифры или буквы латинского алфавита, причем как строчные, так и прописные, а также указать нумерацию на основе греческого, армянского или грузинского языка. Для этого в HTML4 существовал атрибут `type`, который на данный момент является устаревшим. В HTML5 вместо него принято указывать тип списка с помощью стилей, а именно с помощью CSS-свойства `list-style-type`:

```
list-style-type: decimal (по умолчанию, можно не
  указывать) | decimal-leading-zero | armenian |
  georgian | lower-alpha | lower-greek | lower-latin |
  lower-roman | upper-alpha | upper-latin |
  upper-roman;
```

Визуально эти стили будут представлять собой такую нумерацию:

```
list-style-type: decimal-leading-zero;
```

- 01. Работающая новая стратегия, новые идеи, новые продукты
- 02. Трансформация бизнеса и команды
- 03. Амбициозные и достижимые цели бизнеса и команды
- 04. Фокусировка и координированность действий
- 05. Освоение методологии разработки стратегии
- 06. Рост, развитие и капитализация бизнеса

Наша компания предоставляет такие услуги

```
list-style-type: lower-alpha; (lower-latin)
```

- a. Зеленый офис
- b. Зоны отдыха
- c. Медицинское страхование
- d. Мультиспортивный абонемент
- e. Спортивзал
- f. Дисконтная программа
- g. Корпоративные праздники
- h. Корпоративные тренинги
- i. Кухня

Формат разработки

```
list-style-type: upper-alpha;
```

- A. Стратегическая сессия
- B. Стратегический аудит
- C. Межсессионная разработка
- D. Серия интервью с командой
- E. Рыночные исследования

```
list-style-type: upper-roman;
```

- I. Куда стремиться?
- II. На каком поле играть?
- III. Каким образом добиться победы?
- IV. В чем нужно быть сильнее?
- V. Какие управленческие системы нужно внедрить?

Почему стоит путешествовать с нами по Грузии

```
list-style-type: georgian;
```

- ა. Готовые туры, разработанные знатоками Грузии
- ბ. Уже более 10 лет мы работаем в сфере туризма
- გ. Грузинско-русская команда
- დ. Свой автопарк и свои водители
- ე. Своя команда обученных гидов
- ვ. Дружелюбный и лояльный персонал
- ზ. Мы на связи всегда (24/7)

Города и популярные курорты Армении

```
list-style-type: armenian;
```

- Ա. Ереван, столица Армении
- Բ. Гюмри, культурная столица Армении
- Չ. Цахкадзор, горнолыжный курорт Армении
- Ղ. Шушин, культурный центр Арцаха
- Ճ. Джермук, высокогорный курорт

Почему стоит купить недвижимость в компании D

```
list-style-type: lower-greek;
```

- ο. Более 20 лет на рынке недвижимости. Эти годы позволили нам приобрести множество объектов и их законности.
- β. Наша компания является застройщиком с огромным опытом работы в сфере жилых объектов и их законности.
- γ. Большой выбор недвижимости в Греции в сегменте эконом-класса.
- δ. Весь персонал компании говорит на английском языке.
- ε. Сопровождение клиента на всех этапах приобретения недвижимости, как в Греции, так и в Европе.
- ζ. Возможность размещения в собственных апартаментах компании в Дионах.

Рисунок 2

- арабские цифры (1, 2, 3, ...) — decimal;
- арабские цифры с ведущим нулём для цифр первой десятки (01, 02, 03, ..., 09) — decimal-leading-zero;
- армянская нумерация — armenian;
- грузинская нумерация — georgian;
- прописные латинские буквы (A, B, C, ...) — upper-alpha или upper-latin;
- строчные латинские буквы (a, b, c, ...) — lower-alpha или lower-latin;
- римские цифры в верхнем регистре (I, II, III, ...) — upper-roman;

- римские цифры в нижнем регистре (i, ii, iii, ...) — lower-roman;
- строчные греческие буквы — lower-greek.

Посмотреть на внешний вид списков с различными стилями можно в файле *ordered-list-style.html* (рис. 2).

Нумерованные списки имеют также дополнительные атрибуты: *start* для тега ** и *value* для тега **. В этом случае список начинается не с цифры 1, а с той, которая указана в атрибуте *start*. Как правило, этот атрибут бывает необходим, если список нужно разорвать каким-либо пояснением или картинкой. Пример (рис. 3).

Требования к кандидату на должность в компании IT-Softwear

На собеседовании с кандидатом на вакантную должность в первую очередь оцениваются такие показатели:

1. Знания и умения кандидата
2. Образование
3. Опыт работы в ИТ
4. Навыки работы в команде

После успешного прохождения собеседования, кандидату будет предложены следующие тесты:

5. Тест на определение уровня владения английским языком
6. Внешняя сертификация
7. Оценка знаний
8. Оценка эффективности работы в условиях установки сжатых сроков

Рисунок 3

```
<p>На собеседовании с кандидатом на вакантную должность  
в первую очередь оцениваются такие показатели:</p>
```

```
<ol>  
    <li>Знания и умения кандидата</li>  
    <li>Образование</li>  
    <li>Опыт работы в ИТ</li>  
    <li>Навыки работы в команде</li>  
</ol>
```

```
<p>После успешного прохождения собеседования,  
кандидату будет предложены следующие тесты:  
</p>  
  
<ol start="5">  
    <li>Тест на определение уровня владения английским  
        языком</li>  
    <li>Внешняя сертификация</li>  
    <li>Оценка знаний</li>  
    <li>Оценка эффективности работы в условиях  
        установки сжатых сроков</li>  
</ol>
```

Следует отметить, что используется он крайне редко. Так же редко используется атрибут `value` для элемента `li`, который меняет порядок нумерации, начиная с той цифры, которая указана в качестве значения атрибута. Этот атрибут имеет приоритет перед атрибутом `start`, если они указаны в одном и том же списке. Кроме того, эти 2 атрибута применяются к любой нумерации (латинскими буквами, римскими цифрами и т.п.). В примере выделены красным цветом те элементы `li`, у которых есть атрибут `value`:

Акции, бонусные программы, мероприятия

1. система раннего бронирования (РБ);
2. «горячие туры» по спецпредложениям (СП1-СП4);
3. специальные предложения от партнеров (СПП);
- 10. система скидок постоянным и корпоративным клиентам;**
11. помочь в оформлении банковского кредита на путешествие;
12. регулярно проводятся акции и бонусные программы;
- 30. «наши друзья – ваши друзья» - мы предлагаем вам лояльность и различные льготные условия при приобретении услуг у наших партнеров;**
31. регулярные мероприятия и дни открытых дверей, посвященные туристическим новинкам и открытию туристического сезона!
32. поощрение клиентов подарочными сертификатами.

Рисунок 4

```
<ol>
    <li>система раннего бронирования (РБ);</li>
    <li>"горячие туры" по спецпредложениям
        (СП1-СП4);</li>
    <li>специальные предложения от партнеров (СПП);</li>
    <li value="10">система скидок постоянным
        и корпоративным клиентам;</li>
    <li>помощь в оформлении банковского кредита
        на путешествие;</li>
    <li>регулярно проводятся акции и бонусные
        программы;</li>
    <li value="30">"наши друзья - ваши друзья" -
        мы предлагаем вам лояльность ...;</li>
    <li>регулярные мероприятия и дни открытых дверей,
        посвященные туристическим новинкам и открытию
        туристического сезона! </li>
    <li>поощрение клиентов подарочными сертификатами.</li>
</ol>
```

Еще один малоиспользуемый атрибут для тега `` — `reversed` — изменяет порядок нумерации списка на обратный. Т.е. вместо 1, 2, 3, 4 будет 4, 3, 2, 1. Например, так:

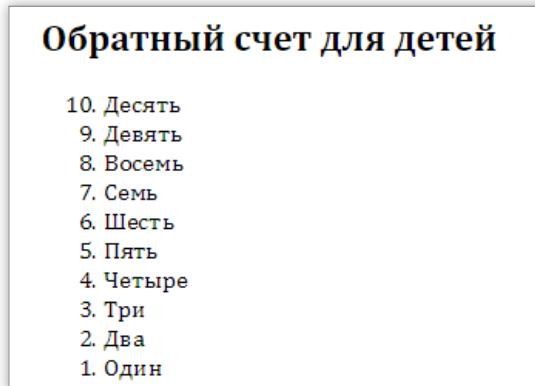


Рисунок 5

```
<ol reversed>
    <li>Десять</li>
    <li>Девять</li>
    <li>Восемь</li>
    <li>Семь</li>
    <li>Шесть</li>
    <li>Пять</li>
    <li>Четыре</li>
    <li>Три</li>
    <li>два</li>
    <li>Один</li>
</ol>
```

Примечание: *Internet Explorer не поддерживает этот атрибут.*

Посмотреть код примеров можно в файле *ordered-list-attributes.html*.

Маркированный список

Маркированный список отличается от нумерованного тем, что вместо цифр имеет маркеры (по умолчанию в виде черных кружочков). Также маркированный список называется неупорядоченным (*unordered list*), т.к. нумерация в нем не нужна (рис. 6).

Формируется маркированный список аналогично нумерованному двумя тегами: ** (от англ. *unordered list*) и уже известным вам ** (от англ. *list item*).

```
<ul>
    <li>Биографии (255)</li>
    <li>Боевики (1581)</li>
    <li>Вестерны (76)</li>
```

```
<li>Военные (237)</li>
<li>Детективы (627)</li>
<li>Документ (64)</li>
<li>Драмы (2814)</li>
<li>Исторические (237)</li>
<li>Комедии (1983)</li>
</ul>
```

Категории

- Биографии (255)
- Боевики (1581)
- Вестерны (76)
- Военные (237)
- Детективы (627)
- Документ (64)
- Драмы (2814)
- Исторические (237)
- Комедии (1983)

Рисунок 6

Файл примера — *unordered_list.html*.

С помощью устаревшего атрибута *type* ранее определялся вид маркера, который на данный момент заменяется CSS-свойством *list-style-type*:

```
list-style-type: disc | circle | square | none
```

- **disc** — темный кружок (значение по умолчанию, можно не указывать);
- **circle** — пустой кружок;
- **square** — темный квадрат;
- **none** — маркеры (нумерация) отсутствует.

Урок 3. Списки. CSS отступы и поля

Responsibilities:

```
list-style-type: disc; (default)
```

- Lead in the development of hardening the web application in protecting PII (Personally Identifying Information)
- Work side-by-side with product owners to create products and features to help protect users information while remaining seamless to the experience
- Find and build creative solutions to satisfy general audiences and weather enthusiasts while building in compliance for GDPR
- Something about integrating with external data sources/CMS to generate consent flow screens
- Write fluid/liquid/responsive HTML and CSS that works across all three screens (phone, tablet, and desktop)
- Designing and implementing solutions collaboratively to add value to software development efforts in the most cost efficient way. Must be able to effectively communicate JavaScript/HTML/CSS designs to other team members for they could implement the design.
- Ability to define, document and write Unit Test for front-end web technologies
- Create custom JavaScript libraries and utilize APIs
- Knowledge and experience in any JavaScript frameworks, including Angular
- Strong aptitude for both giving and taking direction
- Strong documentation skills to work properly with the general audience (not necessarily technical)

Requirements:

```
list-style-type: circle;
```

- Core aspects of software development using HTML5, CSS and JavaScript
- 3+ years experience in web applications development
- Strong OOP and Functional software design ability. Able to develop systems that meet architectural objectives including reusable, scalable code
- Experience with Angular + NodeJS
- Exposure to build tools like Webpack and Jenkins
- Working knowledge of client and server-side Javascript
- Ability to care about his/her products and the people who use them
- Ability to follow the timeline and meet deadlines
- Analytical and detail oriented, with the ability to prioritize, execute and deliver projects on time

As a Plus:

```
list-style-type: none;
```

Github profile and contributed projects
Development projects and code samples you can demo

Рисунок 7

Benefits:

```
list-style-type: square;
```

- Challenging and interesting projects for Fortune 500 companies
- Competitive salary & bonus system
- Your personal Career Advisor & individual career map
- Corporate doctors available 24/7
- Social package
- Certification center and language school
- SoftServ discount program
- A truly flexible working schedule
- Fully equipped gyms and relax zones
- Unforgettable corporate events and even more

Рисунок 8

Файл примера — *unordered-list-style.html*.

Список определений

Списки определений нужны в тех сайтах, которые посвящены, как правило, обучению. Например, отлич-

ный сайт [webref.ru](#) использует списки определений для указания атрибутов тегов или вариантов свойств.

Атрибуты

<code>type</code>	Устанавливает вид маркера списка.
<code>reversed</code>	Нумерация в списке становится по убыванию (3,2,1). HTML5
<code>start</code>	Задаёт число, с которого будет начинаться нумерованный список. HTML5

Также для этого элемента доступны [универсальные атрибуты и события](#).

```

ts Console Sources Network Performance Memory Application Security Audits
▼ <dl class="param">
  ▼ <dt>
    <a href="/html/o1/type">type</a> == $0
  </dt>
  <dd>Устанавливает вид маркера списка.</dd>
  ▼ <dt>
    <a href="/html/o1/reversed">reversed</a>
  </dt>
  ▼ <dd>
    "Нумерация в списке становится по убыванию (3,2,1). "
    ► <span class="attr-HTML5">...</span>
  </dd>
  ▼ <dt>
    <a href="/html/o1/start">start</a>
  </dt>

```

Рисунок 9

Создается список определений с помощью 3-х тегов:

- **dl** — **definition list** — основной, или родительский тег;
- **dt** — **definition termin** — вложенный тег;
- **dd** — **definition description** — вложенный тег.

Код списка выглядит так:

```

<dl>
  <dt>Термин 1</dt>
  <dd>Описание термина 1</dd>
  <dt>Термин 2</dt>
  <dd>Описание термина 2</dd>
  <dt>Термин 3</dt>
  <dd>Описание термина 3</dd>
</dl>

```

Для примера используем определения нескольких терминов (рис. 10):

Интернет
совокупность сетей, применяющих единый протокол обмена (точнее, обширное семейство из сотен протоколов) для передачи информации
Локальная сеть
соединение нескольких компьютеров при помощи технических и программных средств, которые позволяют объединить файловые системы входящих в нее компьютеров
Глобальная сеть
такое соединение компьютеров, при котором допускается использование информации, физически находящейся на других компьютерах.
Провайдер сетевых услуг
владелец конкретной сети, с которой клиент находится в юридических отношениях

Рисунок 10

На скриншоте видно, что по сравнению с термином в элементе **dt** элемент **dd** имеет смещение от левого края окна браузера на 40px вправо. Этот отступ определяется свойством **margin-left** или в браузере Хром — **webkit-margin-start**. За счет этого определение визуально отличается от термина и привлекает к себе внимание (рис. 11).

```
dd { user agent stylesheet
      display: block;
      -webkit-margin-start: 40px;
}
```

Рисунок 11

Список определений хорошо форматировать с помощью css. Для сайта, который использует много вариантов таких списков, имеет смысл назначить css-свойства для селектора тега, не прибегая ни к использованию селекторов классов, ни селекторов **id**. Например, список на скриншоте выше можно отформатировать так (рис. 12).

Интернет

совокупность сетей, применяющих единый протокол обмена (точнее, обширное семейство из сотен протоколов) для передачи информации

Локальная сеть

соединение нескольких компьютеров при помощи технических и программных средств, которые позволяют объединить файловые системы входящих в нее компьютеров

Глобальная сеть

такое соединение компьютеров, при котором допускается использование информации, физически находящейся на других компьютерах.

Провайдер сетевых услуг

владелец конкретной сети, с которой клиент находится в юридических отношениях

Рисунок 12

Код css-стилей:

```
<style>
  dt {
    font-weight: bold;
    color: #036011;
  }
  dd {
    color: #333;
    margin-top: 10px;
    margin-bottom: 10px;
  }
</style>
```

Посмотреть данный пример вы можете в файле *definition-list.html*.

Ссылки по теме:

1. <https://webref.ru/layout/html5-css3/list>.
2. <https://html5book.ru/html-lists/>.
3. <http://htmlbook.ru/faq/theme/list>.
4. <http://html-plus.in.ua/spiski-v-html/>.

Многоуровневые, или вложенные списки

Многоуровневым списком является список, внутри элементов `` которого размещается список второго уровня. Т.е. второй список вкладывается внутрь первого, поэтому такие списки называются еще вложенными (файл `inner_list.html`).

В многоуровневом списке можно использовать как упорядоченные, так и неупорядоченные списки, а также списки определений, чередуя их в любой последовательности. Важно, чтобы вложенный список находился именно в элементе `` родительского списка. Количество уровней в списках может быть любым, хотя больше 3-х уровней вложенности бывает крайне редко.

Пример правильного кода:

```
<ol>
    <li>Площади
        <ul>
            <li>Дворцовая площадь</li>
            <li>Рыночная площадь</li>
        </ul>
    </li>

    <li>Замки
        <ul>
            <li>Королевский замок</li>
            <li>Вилянувский дворец</li>
        </ul>
    </li>
    <li>Старе Място</li>
    <li>Музеи

```

```
<ul>
    <li>Исторический музей</li>
    <li>Музей Войска Польского</li>
    <li>Музей карикатуры</li>
    <li>Музей Марии Кюри</li>
    <li>Музей Шопена</li>
    <li>Национальный музей</li>
</ul>
</li>
<li>Краковское предместье</li>
</ol>
```

Вариант ошибочного кода (файл *invalid-multilevel-list.html*):

```
<ol>
    <li>Площади</li>
    <ul>
        <li>Дворцовая площадь</li>
        <li>Рыночная площадь</li>
    </ul>

    <li>Замки</li>
    <ul>
        <li>Королевский замок</li>
        <li>Вилянувский дворец</li>
    </ul>

    <li>Старе Място</li>
    <li>Музеи</li>
    <ul>
        <li>Исторический музей</li>
```

```
<li>Музей Войска Польского</li>
<li>Музей карикатуры</li>
<li>Музей Марии Кюри</li>
<li>Музей Шопена</li>
<li>Национальный музей</li>
</ul>
<li>Краковское предместье</li>
</ol>
```

Ошибка заключается в том, что вложенный список вставлен не внутрь элемента ``, а между двумя соседними элементами, а это недопустимо с точки зрения [валидатора](#):



Рисунок 13

Визуально список выглядит так (рис. 14):

- 1. Площади
 - Дворцовая площадь
 - Рыночная площадь
- 2. Замки
 - Королевский замок
 - Вильнюсский дворец
- 3. Старе Място
- 4. Музеи
 - Исторический музей
 - Музей Войска Польского
 - Музей карикатуры
 - Музей Марии Кюри
 - Музей Шопена
 - Национальный музей
- 5. Krakowskie Przedmieście

Рисунок 14

Этот список можно дополнить и он будет значительно больше (рис. 14).

Достопримечательности известных городов мира

- 1. Достопримечательности Парижа
 1. Эйфелева башня
 2. Нотр Дам де Пари
 3. Триумфальная арка
 4. Дворец Инвалидов
 5. Сакре Кер
 6. Пантеон
 7. Мост Александра III
 8. Лувр
 9. Консержери
 10. Башня Монпарнас
 11. Елисейские поля (Champs-Elysées)
- 2. Достопримечательности Рима
 1. Аппиева дорога
 2. Арка Константина
 3. Вилла Боргезе
 4. Галерея Боргезе
 5. Испанская лестница
 6. Капитолийский холм
 7. Колизей
 8. Музейный комплекс Ватикана
 9. Палатин
 10. Пантеон

Рисунок 15 (начало)

Достопримечательности известных городов мира

1. Достопримечательности Парижа
 1. Эйфелева башня
 2. Нотр Дам де Пари
 3. Триумфальная арка
 4. Дворец Инвалидов
 5. Сакре Кер
 6. Пантеон
 7. Мост Александра III
 8. Лувр
 9. Консьержери
 10. Башня Монпарнас
 11. Елисейские поля (Champs-Elysées)
2. Достопримечательности Рима
 1. Аппиева дорога
 2. Арка Константина
 3. Вилла Боргезе
 4. Галерея Боргезе
 5. Испанская лестница
 6. Капитолийский холм
 7. Колизей
 8. Музейный комплекс Ватикана
 9. Палатин
 10. Пантеон

Рисунок 15 (*продолжение*)

Файл примера — *multilevel-list.html*.

Обратите внимание, что каждый вложенный список смещен относительно своего родителя вправо, причем при вложении друг в друга маркированных списков меняется стиль маркера — **disc** у главного списка, **circle** у первого дочернего списка и **square** — у второго дочернего списка.

Ссылки по теме:

1. <https://webref.ru/layout/html5-css3/list/nested>.
2. https://puzzleweb.ru/html/11_lists2.php.
3. <http://html-plus.in.ua/vlozhennye-spiski/>.

CSS-стили для списков

CSS-свойства, о которых пойдет речь ниже, назначают обычно для маркированных или нумерованных списков, причем указывать их можно как для родительских элементов (`ul`, `ol`), так и для вложенных (`li`).

Одно из стилевых свойств списков мы уже рассмотрели. Это свойство `list-style-type`, определяющее вид маркера или цифры:

```
list-style-type: decimal | decimal-leading-zero |
    armenian | georgian | lower-alpha | lower-greek |
    lower-latin | lower-roman | upper-alpha | upper-latin |
    upper-roman | disc | circle | square | none;
```

В предыдущих примерах мы разделяли его по значениям, которые были разными для нумерованных и маркированных списков. Вы можете назначить любое из значений для любого вида списка, однако логика HTML-разметки тогда потеряется.

Довольно часто маркеры или цифры в списках «прячут», используя значение `list-style-type: none`. Например, так делают при форматировании меню.

```
Аббревиатуры Emmet: lst, lstdlz, lstlr, lstur, lstd,
    lstc, lsts, lstn
```

Второе свойство — это позиция цифр или маркеров. Определяется свойством `list-style-position` с такими значениями:

```
list-style-position: outside | inside | inherit
```

По умолчанию у всех списков `list-style-position: outside`, поэтому использовать это значение вы будете редко. Оно подразумевает, что маркеры (цифры) размещаются слева от текста элемента списка. Значение `list-style-position: inside` размещает маркеры (цифры) внутри текста элемента списка.

1. `list-style-position: inside`

- 2. Консультативная помощь пациенту на каждом этапе реабилитационного процесса,
- 3. Жесткие стандарты проверки качества медицинских и дополнительных услуг.
- 4. Выделение бюджета на инвестиции в образование и повышение квалификации сотрудников центра,
- 5. Надежность результатов работы специалистов — основной принцип оказания медицинской услуги.

• `list-style-position: inside;`

- стойкое и долгосрочное улучшение качества медицинских услуг
- разработка терапевтических протоколов с учетом расширяющихся видов патологий пациентов центра
- гибкость и способность быстро адаптироваться к любым клинически сложным случаям
- научно-исследовательская работа и внедрение инноваций
- разработка и внедрение систем реабилитационных манипуляций и новых технологий

1. `list-style-position: outside; (default)`

- 2. Консультативная помощь пациенту на каждом этапе реабилитационного процесса,
- 3. Жесткие стандарты проверки качества медицинских и дополнительных услуг.
- 4. Выделение бюджета на инвестиции в образование и повышение квалификации сотрудников центра,
- 5. Надежность результатов работы специалистов — основной принцип оказания медицинской услуги.

• `list-style-position: outside; (default)`

- стойкое и долгосрочное улучшение качества медицинских услуг
- разработка терапевтических протоколов с учетом расширяющихся видов патологий пациентов центра
- гибкость и способность быстро адаптироваться к любым клинически сложным случаям
- научно-исследовательская работа и внедрение инноваций
- разработка и внедрение систем реабилитационных манипуляций и новых технологий

Рисунок 16

Значение `inherit` подразумевает, что вложенный список наследует это значение от родительского элемента. Поскольку вложенные элементы списков обычно имеют такие же свойства, как и родительские, поэтому для них используют это значение не так часто.

Аббревиатуры Emmet: `lsp`, `lspi`, `lspo`

Пример можно в файле `list-style-position.html`.

Третьим, достаточно популярным свойством, является `list-style-image`, которое позволяет установить изображение в качестве маркера списка:

```
list-style-type: none |  
                    url(путь к файлу изображения.расш);  
list-style-type: url(markers/arrow.gif);
```

или

```
list-style-type: url(http://mysite.com/images/ arrow.gif);
```

В первом примере путь к файлу указывается относительно местоположения html-файла и папки с изображениями для маркеров (они должны находиться в общей папке). Во втором — путь указывается с учетом доменного имени вашего сайта. Поскольку на данном этапе сайт у вас вряд ли существует, то единственным вариантом для аудиторных и домашних заданий будет первый пример.

Что касается текстового редактора Brackets, то в нем существуют очень удобные подсказки, позволяющие легко указать путь к файлу изображения-маркера, а также посмотреть при наведении на строку пути к изображению, как выглядит файл, и его размер. Важным условием для этих подсказок является открытие папки в качестве проекта в Brackets, доступное из контекстного меню проводника.

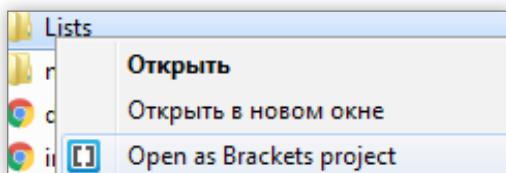


Рисунок 17

Как только вы написали свойство `list-style-type: url()` и оставили курсор в скобках, Brackets начинает подсказы-

вать вам, показывая папки, которые есть в вашем проекте и файлы в этих папках:

```
ul {  
    list-style-image: url();  
}  
</style>
```

url()

Рисунок 18

```
ul {  
    list-style-image: url();  
}  
</style>  
</head>  
  
<body>  
    <h1>Фундаментальные ценности</h1>  
    <ul>  
        <li>Ответственность</li>  
        <li>Верность принципам</li>  
        <li>Творческий подход</li>  
    </ul>  
</body>
```

markers/
definition-list.html
inner_list.html
list-style-image.html
list-style-position.html
list.html
ordered-list-attributes.html
ordered-list-style.html

Рисунок 19

В списке будут также и все файлы, которые есть в текущей папке, поэтому будьте внимательны. Свой выбор вы можете подтвердить, нажав клавишу Enter или сделав клик левой кнопкой мыши.

```
<meta charset="UTF-8">  
<title>CSS property list-style-image</title>  
<style>  
    body {  
        font-family: sans-serif;  
        width: 80%;  
        margin: auto;  
    }  
    ul {list-style-image: url(markers/);}  
</style>
```

markers/arr.gif
markers/arrb.gif
markers/arrow.gif
markers/leaf.png
markers/rain.png
markers/snow.png
markers/sun.png

Рисунок 20

При наведении на текст в скобках вы увидите, как выглядит изображение и каков его размер (на скриншоте 25×25px):

```
<style>
  body {
    font-family: sans-serif;
    width: 80%;
    margin: auto;
  }
  ul {list-style-image: url(markers/check.png);}
</style>
```

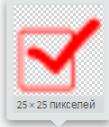


Рисунок 21

Для того чтобы поменять изображение на любое другое, нужно просто поставить курсор после слэша и нажать **Ctrl+пробел**:

```
ul {
  list-style-image: url(markers/check.png);
}
</style>
<head>
</head>
```

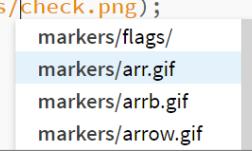


Рисунок 22

Для замены одного изображения другим нужно просто выбрать его и нажать **Enter** — и название файла будет автоматически заменено, даже если не было выделено.

При выборе различных файлов для перемещения по списку можно использовать не только мышь, но и клавиши со стрелками вверх и вниз. В результате получим следующий вид списка (рис. 23).

Файл примера *list-style-image.html*.

Фундаментальные ценности центра

- Ответственность
- Верность принципам
- Творческий подход
- Эффективность
- Качество
- Профессионализм
- Новаторство
- Командная работа

Рисунок 23

Примечание: свойство *list-style-image* имеет приоритет перед *list-style-type*, т.е. изображение всегда будет перекрывать маркер или номер элемента списка.

В случае если нужно убрать в одном из элементов списка назначенный маркер-изображение, используют значение *none*:

```
list-style-image: none;
Аббревиатура Emmet: lsi, lis, lisin
```

Универсальное свойство *list-style* позволяет задать через пробел все свойства из вышеперечисленных, т.е. указать и стиль маркера, и его позицию, и изображение. Перечисление идет в такой последовательности:

```
list-style: list-style-type list-style-position
          list-style-image;
```

Например:

```
list-style: upper-latin inside url(markers/somepic.png);
```

Фундаментальные ценности центра

- ... Ответственность
- ... Верность принципам
- ... Творческий подход
- ... Эффективность
- ... Качество
- ... Профессионализм
- ... Новаторство
- ... Командная работа

Рисунок 24

Поскольку изображение перекрывает маркер, имеет смысл ограничиться двумя свойствами:

```
list-style: upper-latin inside;
```

Комедии 2017г.

- A. Семейное ограбление
- B. Во всем виноват енот
- C. Если б я была мужчиной
- D. Не/смотря ни на что
- E. Его собачье дело
- F. Ой, мамочки
- G. Доспехи бога: В поисках сокровищ
- H. Видели ночь
- I. Гипнотизм
- J. Телохранитель киллера
- K. Завуалируй это
- L. В гостях у Элис
- M. SuperАЛИБИ
- N. Kingsman: Золотое кольцо
- O. Пока не кончилось лето
- P. Послесвадебный разгром

Рисунок 25

ИЛИ

```
list-style: inside url(markers/somepic.png);
```

Можно указать в этом CSS-свойстве только одно значение:

```
list-style: url(markers/actor-icon.png);
```

Список актеров в фильме "Лучшее предложение"

-  Джейфри Раш — Вирджил Олдман — главная роль
-  Джим Стёрджесс — Роберт — главная роль
-  Сильвия Хукс — Клара — главная роль
-  Дональд Сазерленд — Билли
-  Филип Джексон — Фрэд
-  Дермот Кроули — Ламберт
-  Лия Кебеде — Сара
-  Кируна Стамелл — девушка в баре
-  Джон Бенфилд — бармен
-  Клаус Таубер — ассистент Вирджила
-  Максимилиан Дирр — ассистент Вирджила
-  Ген Сето — домоправитель
-  Сильвия Де Фанти — менеджер галереи

Рисунок 26

Значение по умолчанию: `disc outside none;`
Можно также указать значение `inherit`.

Аббревиатуры Emmet: `lis`, `lisl`

Файл с примерами `list-style.html`.

Использование псевдоэлемента `::before` для стилей списков

Зачастую хочется изменить нумерацию в упорядоченном списке, задав для него скобки после номера, а для неупорядоченного списка — тире вместо маркера. Для

этого необходимо отключить стандартную нумерацию или маркеры с помощью свойства `list-style-type: none`. А затем использовать псевдоэлемент `::before` для элемента ``.

Что такое псевдоэлементы `::before` и `::after`?

Псевдоэлементы имеют приставку псевдо- перед знакомым вам уже словом «элемент», потому что изначально в html-коде их нет. Кстати, их не будет в коде вообще, потому как появляются они исключительно с помощью css. Псевдоэлементов существует несколько, а именно `::first-line`, `::first-letter`, `::selection`, `::before` и `::after`. Причем для них доступен синтаксис и с двумя двоеточиями перед наименованием (CSS3), и с одним (CSS2), т.к. в версии CSS3 было принято решение отделить псевдоэлементы от псевдоклассов, добавив к псевдоэлементам второе двоеточие. Поэтому не удивляйтесь, если вы увидите разное их написание.

По поводу `:before` и `:after` можно сказать, что эти псевдоэлементы позволяют добавить некий контент (текстовое или визуальное содержимое) перед текстом элемента или в конце элемента, т.е. после содержимого элемента. Причем текст псевдоэлемента выделяться не будет, а css-свойства для него можно задать такие же, как и для обычных элементов.

По умолчанию `:before` и `:after` являются строчными элементами, и имеют такой же шрифт, цвет текста и начертание, что и у основного элемента.

Например, псевдоэлементы `:before` и `:after` для абзаца, у которого в стилях указано курсивное начертание, тоже будут выведены курсивом:

```
p { font-style: italic; }
p:before, p:after { background-color: #ccc; }
p:before { content: 'Start'; }
p:after { content: "End"; }
```

Start *Lorem ipsum dolor sit amet, consectetur adipisicing elit. Illo qui quod eius, vitae accusamus eveniet! Ipusum doloribus dolore sed numquam!**End*

Start *Quas quia veniam autem delectus, non repellat consequatur, amet corporis ullam odit deserunt, ab itaque ipsam assumenda aliquid. Nulla, qui.**End*

Рисунок 27

В примере (файл *list-before.html*) псевдоэлементы выделены серым фоном. В коде в свойстве **content** указан в кавычках (двойных или одинарных) текст, который будет выведен до и после текста абзаца.

Примечание: свойство **content** является обязательным для псевдоэлементов **::before** **::after**. Если это свойство не будет указано, даже с пустыми кавычками («» или ”), то на странице эти псевдоэлементы не отобразятся, какие бы еще свойства вы не писали для них.

Ссылки по теме:

1. <https://habrahabr.ru/post/154319/>.
2. <https://webref.ru/css/before>.
3. <https://webref.ru/css/after>.
4. <http://html-plus.in.ua/psevdo-elements-what-is-before-and-after/>.
5. <http://html-plus.in.ua/stili-dlya-spiskov/>.

Что касается упорядоченных списков, то в них обычно используют только псевдоэлемент `:before` для изменения нумерации, указывая некое имя счетчика (например, `number`) в нескольких свойствах:

```
ol { list-style-type: none;
      counter-reset: number; }

ol li::before {
    content: counter(number) ') ';
    counter-increment: number; }
```

Результат:

Rotational motion

- 1) Angular momentum (Introduction)
- 2) Angular velocity
- 3) Centrifugal force
- 4) Centripetal force
- 5) Circular motion
- 6) Tangential velocity
- 7) Torque

Рисунок 28

Для неупорядоченных списков в свойстве `content` указывают, например, текст с пробелом вместо маркера (рис. 29).

```
ul { list-style-type: none;
      font-style: italic; }
ul li::before { content: '- ';
      color: #04b548; }
```

Conservation of energy and momentum

- *Conservation of energy*
- *Elastic collision*
- *Energy*
- *Inelastic collision*
- *Inertia*
- *Kinetic energy*
- *Moment of inertia*
- *Momentum*
- *Potential energy*
- *Rotational energy*

Рисунок 29

Оформление многоуровневых списков. Вложенные селекторы

Для оформления многоуровневого списка недостаточно написать css-правила только для селектора элемента (ol, ul, li). Чаще всего для них (и не только) придется применять контекстные (или вложенные) селекторы. Они подразумевают, что css-свойства назначаются для тега, вложенного в другой тег, например, селектор

```
ol ul {color: red}
```

подразумевает, что красного цвета будет только тот маркированный список (ul), которыйложен в нумерованный список (ol).

Количество селекторов может быть любым, но следует понимать, что css-правила назначаются для самого последнего селектора, но с учетом его вложенности во все предыдущие. Т.е. читается такой селектор справа налево. Например, код

```
.season ul li { font-style: italic; }
```

говорит о том, что курсивное начертание будет только у элементов `li` маркированного списка `ul`, который находится в элементе с классом `season`.



Рисунок 30

Например, необходимо оформить список с перечислением времен года так, как показано на изображении слева.

Если проанализировать его внешний вид, то станет понятно, что это нумерованный список с 2-мя уровнями вложенности, причем вложенный список может быть как нумерованным, так и маркированным, т.к. цифры или маркеры все равно будут перекрыты изображениями. Поскольку нумерация не будет видна, имеет смысл сделать вложенный список маркированным. Поэтому начальная html-разметка будет такой:

```
<h2>Времена года</h2>
<ol>
    <li>Зима
        <ul>
            <li>Декабрь</li>
            <li>Январь</li>
            <li>Февраль</li>
        </ul>
```

Времена года

1. Зима
 - Декабрь
 - Январь
 - Февраль
2. Весна
 - Март
 - Апрель
 - Май
3. Лето
 - Июнь
 - Июль
 - Август
4. Осень
 - Сентябрь
 - Октябрь
 - Ноябрь

Рисунок 31

```
</li>
<li>Весна
    <ul>
        <li>Март</li>
        <li>Апрель</li>
        <li>Май</li>
    </ul>
</li>
```

```
<li>Лето
    <ul>
        <li>Июнь</li>
        <li>Июль</li>
        <li>Август</li>
    </ul>
</li>

<li>Осень
    <ul>
        <li>Сентябрь</li>
        <li>Октябрь</li>
        <li>Ноябрь</li>
    </ul>
</li>
</ol>
```

Времена года

1. Зима
 - Декабрь
 - Январь
 - Февраль
2. Весна
 - Март
 - Апрель
 - Май
3. Лето
 - Июнь
 - Июль
 - Август
4. Осень
 - Сентябрь
 - Октябрь
 - Ноябрь

Рисунок 32

Внешний вид списка представлен на скриншоте.

Теперь нужно сделать так, чтобы шрифт родительского списка `` был жирным, а дочернего списка `` — обычным, или нормальным. Поскольку элементы `` наследуют свойства родительского элемента, назначим для селектора `ol` следующее свойство:

```
ol { font-weight: 600; }
```

И получим в результате список, в котором все элементы получили жирное начертание.



Рисунок 33

Чтобы вложенный список «вернул» нормальное начертание, необходимо указать следующее css-правило:

```
ul { font-weight: 200; }
```

Теперь список имеет следующий вид: элементы родительского списка — с жирным начертанием, элемен-

ты вложенного списка — с обычным, или нормальным начертанием.

Следующим шагом будет назначение цвета для различных блоков текста: синего — для месяцев зимы, зеленого — для весенних месяцев и т.д.

Проще всего это сделать, задав класс или **id** каждому из элементов **li** родительского списка. Поскольку значения цветов будем использовать индивидуально для каждого блока месяцев, используем атрибут **id**:

```
<ol>
    <li id="winter">Зима
        <ul>
            <li>Декабрь</li>
            <li>Январь</li>
            <li>Февраль</li>
        </ul>
    </li>

    <li id="spring">Весна
        <ul>
            <li>Март</li>
            <li>Апрель</li>
            <li>Май</li>
        </ul>
    </li>

    <li id="summer">Лето
        <ul>
            <li>Июнь</li>
            <li>Июль</li>
            <li>Август</li>
        </ul>
    </li>
```

```
<li id="autumn">Осень
  <ul>
    <li>Сентябрь</li>
    <li>Октябрь</li>
    <li>Ноябрь</li>
  </ul>
</li>
</ol>
```

- 1. Зима**
 - Декабрь
 - Январь
 - Февраль
- 2. Весна**
 - Март
 - Апрель
 - Май
- 3. Лето**
 - Июнь
 - Июль
 - Август
- 4. Осень**
 - Сентябрь
 - Октябрь
 - Ноябрь

Рисунок 34

Теперь можно назначить цвета для каждого из селекторов **id**:

```
#winter { color: rgb(0, 0, 200); }
#spring { color: rgb(0, 190, 0); }
#summer { color: #ffd24c; }
#autumn { color: #d96c00; }
```

Внешний вид блоков изменился в соответствии с назначенными цветами.

Последней является задача по определению маркеров в виде различных изображений. Можно было бы назначить каждому из 3-х элементов `` атрибут класс и задать для него свойство `list-style-image`, но проще использовать здесь контекстный селектор такого вида:

```
#winter li { list-style-image: url(markers/snow.png); }
#spring li { list-style-image: url(markers/leaf.png); }
#summer ul { list-style-image: url(markers/sun.png); }
#autumn ul { list-style-image: url(markers/rain.png); }
```

Времена года

- 1. Зима**
 - ❄ Декабрь
 - ❄ Январь
 - ❄ Февраль
- 2. Весна**
 - ◐ Март
 - ◐ Апрель
 - ◐ Май
- 3. Лето**
 - ☀ Июнь
 - ☀ Июль
 - ☀ Август
- 4. Осень**
 - ◐ Сентябрь
 - ◐ Октябрь
 - ◐ Ноябрь

Рисунок 35

В результате получим нужный нам вид списка.

Обратите внимание, что в первых двух селекторах мы назначали маркер-изображение для элемента `li`, вложенного в определенный селектор `id`, а в третьем и четвертом — для элемента `ul`. При этом визуально разница незаметна. Произошло это по той причине, что свойства,

назначенные для всего списка (`ul` в примере), наследуются его элементами `li`, а свойства для элементов `li` применяются к ним непосредственно.

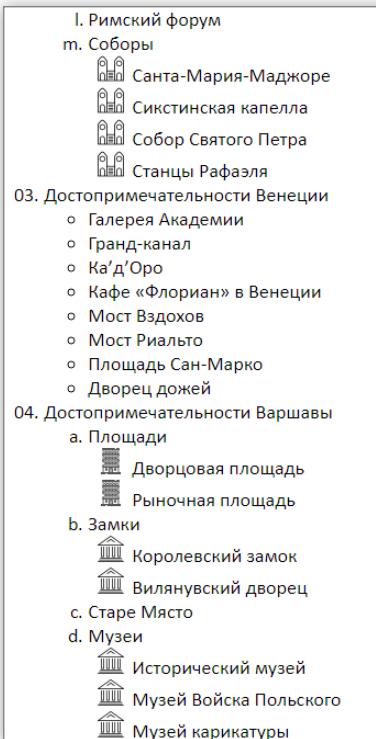


Рисунок 36

Файл примера `seasons.html`.

В файле `multilevel-list-style.html` вы найдете css-формирование для многоуровневого списка, рассмотренного ранее.

Ссылки по теме:

1. <https://webref.ru/css/selector/descendant>.
2. <http://html-plus.in.ua/vlozhennye-spiski/>.

CSS-свойства для управления отступами и полями

Отступы между блоками текста очень важны при верстке. Причем не имеет значения, подразумевается ли верстка книги или журнала или web-страницы. Невозможно читать текст, который не имеет отступов, т.к. в этом случае текст представляет собой сплошной блок, который просто пугает своим внешним видом. Поэтому в CSS предусмотрены 2 вида отступов — внешние, которые называются `margin`, и внутренние с названием `padding`.

Следует отметить, что любые виды отступов мы будем рассматривать для блочных элементов, т.к. строчные элементы отображают их несколько иначе. Эта разница будет рассмотрена в уроке «Блочная модель элементов» подробно.

Внутренние и внешние отступы. Общая информация

Свойство `padding` определяет расстояние от содержимого (контента) блочного элемента до его границы, а `margin` — расстояние между границами элементов, расположенных рядом.

Внешние отступы являются прозрачными, а внутренние можно залить фоновым цветом. Внешние отступы имеют такой эффект как схлопывание, то есть объединение значений рядом размещенных элементов. Внешние

отступы могут иметь отрицательные значения, а внутренние — только положительные. Внутренние и внешние отступы можно записывать либо в сокращённой форме, либо для каждой из сторон отдельно.

Отступы можно назначать с 4-х сторон элемента: верхней (англ. название — `top`), правой (англ. название — `right`), нижней (англ. название — `bottom`) и левой (англ. название — `left`):

- `padding-top/margin-top` — верхний отступ (внутренний/внешний);
- `padding-right /margin-right` — правый отступ (внутренний/внешний);
- `padding-bottom /margin-bottom` — нижний отступ (внутренний/внешний);
- `padding-left /margin-left` — левый отступ (внутренний/внешний);
- `padding/margin` — отступ (внутренний/внешний) (сокращённая форма).

Внешние отступы — `margin`

Внешние отступы можно задать для всех сторон элемента, например:

```
margin: 10px;
```

Можно указать отступы для верхнего и нижнего края элемента (первое значение) и через пробел — для левой и правой стороны элемента (второе значение):

```
margin: 10px 2em;
```

Три значения используют для:

1. Верхней стороны,
2. Левой и правой стороны
3. Нижней стороны элемента

```
margin: 10px 1.5% 30px;
```

Четыре значения определяют значения внешних отступов в следующей последовательности: сверху справа вниз налево

```
margin: 15px 10% 30pt 0;
```

Порядок указания величин отступов начинается с левого верхнего угла и идет по часовой стрелке:

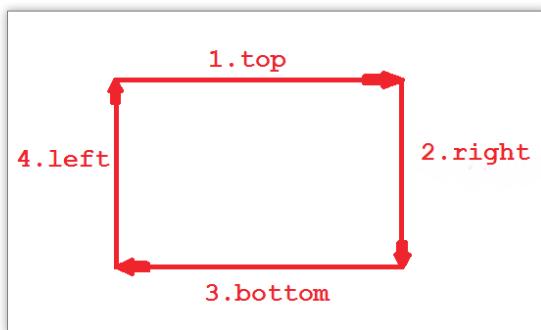


Рисунок 37

Задают отступы в любых единицах измерений (px, pt, em, rem, % др.). Можно указывать отрицательные величины, auto (браузер рассчитывает величину автоматически) или 0 (отступа нет). В последнем случае единицы измерений не указываются, т.к. 0 хоть в %, хоть в px, хоть в em — все равно 0.

Можно также назначить только один из отступов:

```
margin-right: 20px;  
margin-left: 2%;  
margin-top: 10pt;  
margin-bottom: 2em;
```

Еще можно назначить общий отступ для всех сторон, а затем отменить или переопределить его для одной из них:

```
margin: 20px;  
margin-right: 0;
```

Варианты назначения свойства `margin` вы найдете в файле *margin.html*. В нем отступы отображаются бежевым цветом, а блок, которому они применяются — белым.



Рисунок 38

Интересным является значение `auto`. Используя его вместе со свойством `width`, вы предоставляете браузеру рассчитать величину отступа самостоятельно, исходя из размера, указанного для ширины селектора.

Например, при указании для `body` таких свойств:

```
body {  
    width: 1000px;  
    margin: auto;  
}
```

при ширине браузера в 1340px браузер оставит 1000px на отображение контента, а оставшиеся 340px разделит пополам и сформирует отступы в 170px справа и слева от контента, центрировав его. Так же произойдет, если ширина указана в %. Разница состоит только в том, что сначала браузер рассчитает ширину элемента в px, исходя из ширины браузера или ширины родительского элемента, а затем рассчитает величину отступов.

Обычно таким образом указывают отступы не для `body`, а для какого-либо элемента-«обертки», который часто имеет имя класса или id `wrap`, `wrapper`, `container`, `content` и т.п. А для `body` `margin` устанавливают равными 0, т.к. по умолчанию `body` имеет внешние отступы в 8px.

```
body { margin: 0}  
.wrapper {width: 80%; margin: auto;}
```

Еще одной особенностью свойства `margin` является «схлопывание» вертикальных отступов у элементов,

идущих друг за другом. Например, если задать такие CSS-правила для элементов с классом `block`:

```
.block { margin: 40px 0; }
```

можно ожидать, что между блоками, идущими друг за другом, отступы будут 80px. Однако в реальности отступы накладываются, и общий отступ будет в 40px (файл `margin-collapse.html`).

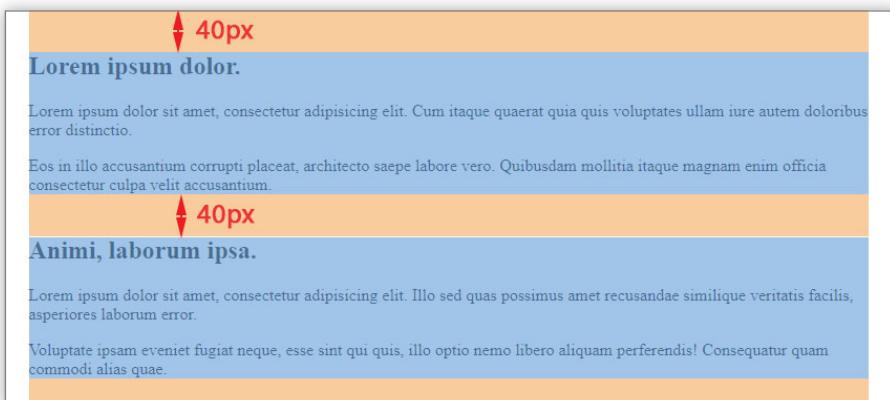


Рисунок 39

Схлопывание срабатывает только для вертикальных отступов, но не применяется для горизонтальных. Кроме того, схлопывание не срабатывает:

- для тега `<html>`;
- для строчных элементов;
- для элементов, имеющих свойство `float: left` или `right`;
- у абсолютно позиционированных элементов;
- для элементов, у которых на стороне схлопывания задано свойство `padding`;

- для элементов, у которых на стороне схлопывания задано свойство border.

Аббревиатура Emmet: m20, m2em, mb2%, mt, ml, mr:7, m:a, m20px10

Внутренние отступы — padding

Внутренние отступы распространяются от контента блочного элемента до его границы (css-свойство border), если она задана. При заливке элемента фоновым цветом padding обычно также имеет этот цвет.

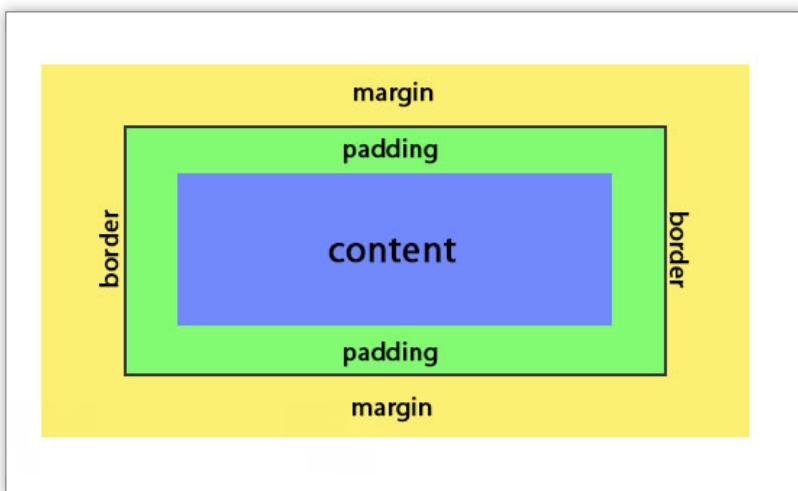


Рисунок 40

Внутренние отступы не могут быть отрицательными и они не схлопываются, как внешние. Сходство этих двух отступов в том, что padding можно задавать в px, pt, em, rem, % и т.п., а также делать это по 4-м сторонам блока так же, как и для margin. Также можно использовать зна-

чение `inherit`, которое показывает, что оно наследуется у родителя.

Одно значение для всех сторон:

```
padding: 20px;
```

Отступ по горизонтальным (сверху и снизу) и вертикальным (справа и слева) сторонам:

```
padding: 2em 10px;
```

Отступ по сверху, отступ для левой и правой стороны и отступ снизу:

```
padding: 2vw 3% 10px;
```

Отступ сверху, справа, снизу, слева:

```
padding: 1rem 5% 0 20px;
```

Отдельный отступ по каждой из сторон:

```
padding-top: 15px;  
padding-bottom: 1rem;  
padding-left: 5%;  
padding-right: 14pt;
```

Общее значение отступов с переопределением его с одной (левой) стороны:

```
padding: 20px; padding-left: 3px;
```

Визуально блоки со всеми этими значениями будут выглядеть так (рис. 41).

Все значения вы найдете в файле *padding.html*.

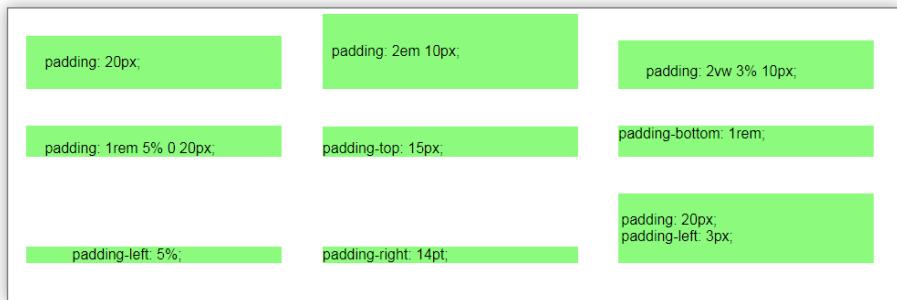


Рисунок 41

Еще одно отличие внутренних отступов от внешних состоит в том, что свойство `padding: auto` не применяется, т.к. не существует значение `auto` для `padding`.

Аббревиатура Emmet: p20, p2em, pb2%, pt, pl, pr:7, p20px6

Примечание: при назначении величин отступов в % следует понимать, что проценты рассчитываются от ширины родительского элемента, а не текущего, поэтому отступ 5% для элемента шириной 200px, который помещен внутрь блока с шириной 1000px, будет составлять 50px ($1000px * 5\% / 100\% = 50px$), а не 10 ($200px * 5\% / 100\% = 10px$).

Значения отступов по умолчанию

Для того чтобы на странице различные html-элементы хорошо смотрелись даже без css-форматирования, у них есть ряд css-свойств, назначенных по умолчанию. Они подтягиваются из таблицы стилей браузера (*user agent stylesheet*), которая создается разработчиками на

основе рекомендаций корпорации W3C. В том числе значения по умолчанию есть у таких свойств, как `margin` и `padding`.

Рассмотренные сегодня элементы списков имеют такие отступы по умолчанию:

```
ul, ol {
    margin-top: 1em;
    margin-bottom: 1em;
    padding-left: 40px;
}
```

Важно отметить, что вложенные элементы `ol` или `ul` в многоуровневых списках НЕ имеют отступов сверху и снизу.

В списках определений значения отступов по умолчанию таковы:

```
dl { margin-top: 1em;
     margin-bottom: 1em;
}
dd { margin-left: 40px; }
```

Для заголовков тоже используются внешние отступы:

```
h1 { margin-top: 0.67em;
      margin-bottom: 0.67em;
}
h2 { margin-top: 0.83em;
      margin-bottom: 0.83em;
}
h3 { margin-top: 1em;
      margin-bottom: 1em;
}
```

```

h4 { margin-top:    1.33em;
      margin-bottom: 1.33em;
}
h5 { margin-top:    1.67em;
      margin-bottom: 1.67em;
}
h6 { margin-top:    2.33em;
      margin-bottom: 2.33em;
}

```

Здесь наблюдается такая тенденция — чем больше цифра заголовка, тем меньше размер шрифта, но больше верхний и нижний внешний отступ.

Отступы у заголовков довольно часто приходится изменять, т.к. в psd-макетах дизайнеры их или увеличивают или уменьшают.

Абзацы по умолчанию также имеют внешние горизонтальные отступы такой же величины, что и у h3:

```

p {
  margin-top:    1em;
  margin-bottom: 1em;
}

```

Для блочной цитаты отступы по умолчанию назначены со всех сторон, но с разными значениями:

```

blockquote {
  margin-top:    1em;
  margin-bottom: 1em;
  margin-left:   40px;
  margin-right:  40px;
}

```

Для элемента `body`, как уже говорилось ранее, внешние отступы по умолчанию составляют 8px:

```
body { margin: 8px; }
```

Такие элементы, как `html`, `div`, `section`, `article`, `aside`, `main` по умолчанию НЕ имеют ни внешних, ни внутренних отступов, поэтому при необходимости вы можете назначать их самостоятельно.

Чтобы убрать отступы, необходимо назначить 0 в качестве значения:

```
body { margin: 0; }
```

Если нужно убрать отступы по умолчанию у всех элементов, используют «простейший сброс», устанавливая правила для универсального селектора:

```
* { margin: 0; padding: 0; }
```



Рисунок 42

Учтите, что при таком подходе маркеры списков «уедут» за пределы браузера, если в `body` или в другом родительском элементе не будет добавлен `margin-left`.

Также для приведения стилей всех элементов страницы к однообразному виду служит файл [reset.css](#), предложенный CSS-гуру [Эриком Майером](#). Это происходит за счет удаления отступов и назначения одного размера шрифта. Сейчас чаще используют другой подход — нормализацию вида элементов за счет использования файла [normalize.css](#), в котором внедрены единообразные стили для всех элементов, чье css-форматирование отличается в зависимости от браузера и его версии.

Узнать, каковы css-свойства у любого элемента страницы, вы можете с помощью инструментов разработчика, предусмотренных ныне в любом браузере, но об этом мы подробно поговорим уже в следующем уроке.

Ссылки по теме:

1. <https://meyerweb.com/eric/tools/css/reset/>.
2. <https://habrahabr.ru/post/45296/>.
3. <https://webref.ru/course/css-basics/css-reset>.
4. <https://necolas.github.io/normalize.css/>.
5. <https://htmlacademy.ru/blog/64-about-normalize-css>.

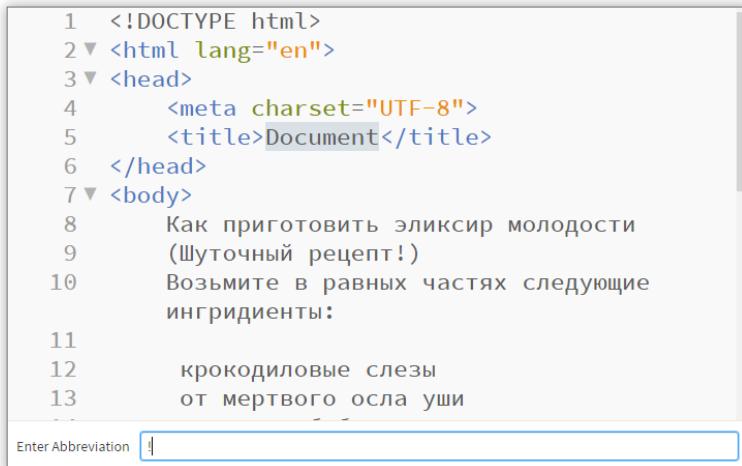
Домашнее задание

Дома вам придется создать и отформатировать с помощью css-правил несколько многоуровневых списков. В папке с домашним заданием вы найдете изображения с внешним видом списков, тексты к ним в txt-файлах и изображения-маркеры в папке markers.

Для центрирования контента используйте сочетание правил `width` и `margin: auto`.

Для того чтобы легче было помещать текст из txt-файла в html-теги, имеет смысл использовать «Emmet-обертку», а именно:

1. Выделить нужный текст.
2. Нажать `Ctrl+Shift+A`.
3. Ввести нужную аббревиатуру.
4. Нажать клавишу `Enter`.



The screenshot shows a code editor interface with a light gray background. On the left, there is a vertical scroll bar. In the main area, a snippet of HTML code is displayed with line numbers from 1 to 13 on the left. Lines 1 through 7 form the basic structure of an HTML document. Lines 8 through 13 contain placeholder text for a recipe. At the bottom of the code area, there is a text input field with the placeholder "Enter Abbreviation" and a small input icon. The entire interface is enclosed in a thin black border.

```

1  <!DOCTYPE html>
2 ▼ <html lang="en">
3 ▼ <head>
4     <meta charset="UTF-8">
5     <title>Document</title>
6   </head>
7 ▼ <body>
8     Как приготовить эликсир молодости
9     (Шуточный рецепт!)
10    Возьмите в равных частях следующие
11    ингредиенты:
12    крокодиловые слезы
13    от мертвого осла уши

```

Рисунок 43

Например, чтобы добавить с помощью такой обертки Emmet теги структуры, нужно выделить весь текст (Ctrl+A), нажать Ctrl+Shift+A, а затем ввести восклицательный знак (!) (рис. 43).

Чтобы поместить текст списка внутрь тегов списка, следует его выделить и использовать такую аббревиатуру:

```
ol>li*
```

```

8     Как приготовить эликсир молодости
9     (Шуточный рецепт!)
10    <ol>
11        <li>Возьмите в равных частях следующие
12            ингридиенты:</li>
13        <li>крокодиловые слезы</li>
14        <li>от мертвого осла уши</li>
15        <li>дырку от бублика</li>
16        <li>Все перемешайте и растолките в порошок</li>
17        <li>Далее следует:</li>
18        <li>полученное залить спиртом крепостью не
19            менее 95 градусов</li>
20        <li>слить в бутылку из непременно темного
21            стекла</li>
22        <li>тщательно взболтать</li>
23        <li>Закопайте в саду в безлунную ночь!</li>
24    </ol>
25    Ровно через год откопайте, выбросьте в море и
26    БОЛЬШЕ НЕ МОРОЧЬТЕ СЕБЕ ЭТИМ ГОЛОВУ!

```

Enter Abbreviation

Рисунок 44

В этом случае теги `ol` появятся в начале и в конце выделенного текста, а теги `` «обернут» все строки, отделенные друг от друга в txt-файле клавишей Enter.

Внешний вид заданий

Обращайте, пожалуйста, внимание на семейство шрифта и его начертание.

Для стилевого оформления используйте контекстные селекторы, селекторы классов и id.



Рисунок 45

Как приготовить эликсир молодости

(Шуточный рецепт!)

- Возьмите в равных частях следующие ингредиенты:
 - о крокодиловые слезы
 - о от мертвого осла уши
 - о дырку от бублика
- Все перемешайте и растолките в порошок
- Далее следует:
 - о полученное залить спиртом крепостью не менее 95 градусов
 - о слить в бутылку из непременно темного стекла
 - о тщательно взболтать
- Закопайте в саду в безлунную ночь!

Ровно через год откопайте, выбросьте в море и **БОЛЬШЕ НЕ МОРОЧЬТЕ СЕБЕ ЭТИМ ГОЛОВУ!**

Рисунок 46

Обратите внимание, что заголовок «Карта сайта» сдвинут вправо и имеет нормальное начертание. Используйте отступы.

Карта сайта

- Новости
- Контакты
- О компании
 - ➡ Наша миссия
 - 1. Преимущества работы с нами
 - 2. Методика
 - ➡ О нас
 - ➡ Отзывы
 - ➡ Команда
- Обучение языкам
 - ➡ Немецкий язык
 - ➡ Английский язык
 - 1. Английский по специализациям
 - 2. Подготовка к сдаче экзаменов
 - 3. Бизнес английский
 - 4. Общий английский
 - 5. Разговорный клуб
 - 6. Специальные интенсивы
 - 7. Новый конкурс "Master Mind"
- Услуги
 - ➡ Английский для детей
 - ➡ Тренинги
 - ➡ Бюро переводов
 - ➡ Обучение за рубежом
- Программы лояльности
- Наша рассылка
- Практические советы
- Вопросы и ответы
- Карта сайта

Рисунок 47



Урок 3.

Списки. CSS отступы и поля

© Слуцкая Елена.

© STEP IT Academy, www.itstep.org.

All rights to protected pictures, audio, and video belong to their authors or legal owners.

Fragments of works are used exclusively in illustration purposes to the extent justified by the purpose as part of an educational process and for educational purposes in accordance with Article 1273 Sec. 4 of the Civil Code of the Russian Federation and Articles 21 and 23 of the Law of Ukraine "On Copyright and Related Rights". The extent and method of cited works are in conformity with the standards, do not conflict with a normal exploitation of the work, and do not prejudice the legitimate interests of the authors and rightholders. Cited fragments of works can be replaced with alternative, non-protected analogs, and as such correspond the criteria of fair use.

All rights reserved. Any reproduction, in whole or in part, is prohibited. Agreement of the use of works and their fragments is carried out with the authors and other right owners. Materials from this document can be used only with resource link.

Liability for unauthorized copying and commercial use of materials is defined according to the current legislation of Ukraine.