Document Object Model (DOM)

JavaScript
Courses

www.courses.dp.ua

window.document

Хранилище HTML-документа

DOM – Document Object Model

(объектная модель документа)

Стандарт определяющий из каких объектов браузер собирает дерево документа, и какие свойства и методы есть у этих объектов.

https://learn.javascript.ru/document

Задача JavaScript – изменение HTML-документа

1. *Добавление нового элемента:*

   Создать новый элемент и присоединить его, в качестве дочернего, к одному из существующих элементов;

2. *Изменение элемента:*

   Изменение свойств элемента (в т.ч. содержимого);
   Изменение его позиции в дереве документа;

3. *Удаление элемента (из дерева документа).*

Структура HTML-документа

состоит из:

<tag attr="value”>Text data</tag>

Теги как контейнер для информации
+**атрибуты**

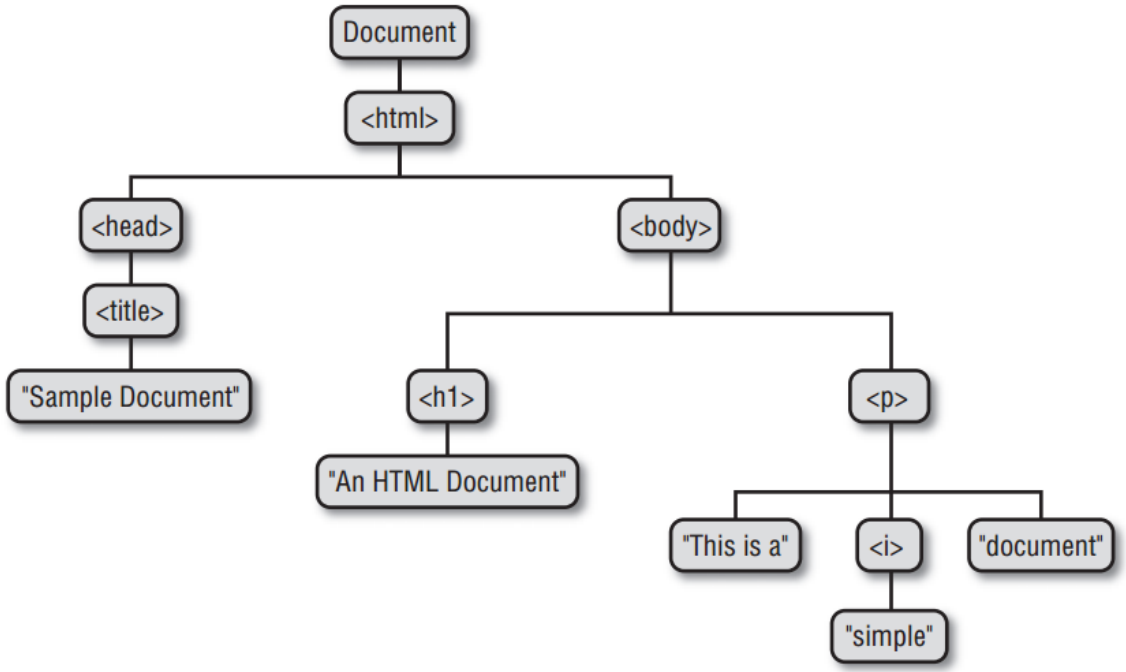
Текстовые данные (содержимое, контент)

Структура HTML-документа

```
<html>
    <head>
        <title>Sample Document</title>
    </head>
    <body>
        <h1>An HTML Document</h1>
        <p>This is a <i>simple</i> document.</p>
    </body>
</html>
```

Древовидная структура HTML-документа

Древовидная структура HTML-документа



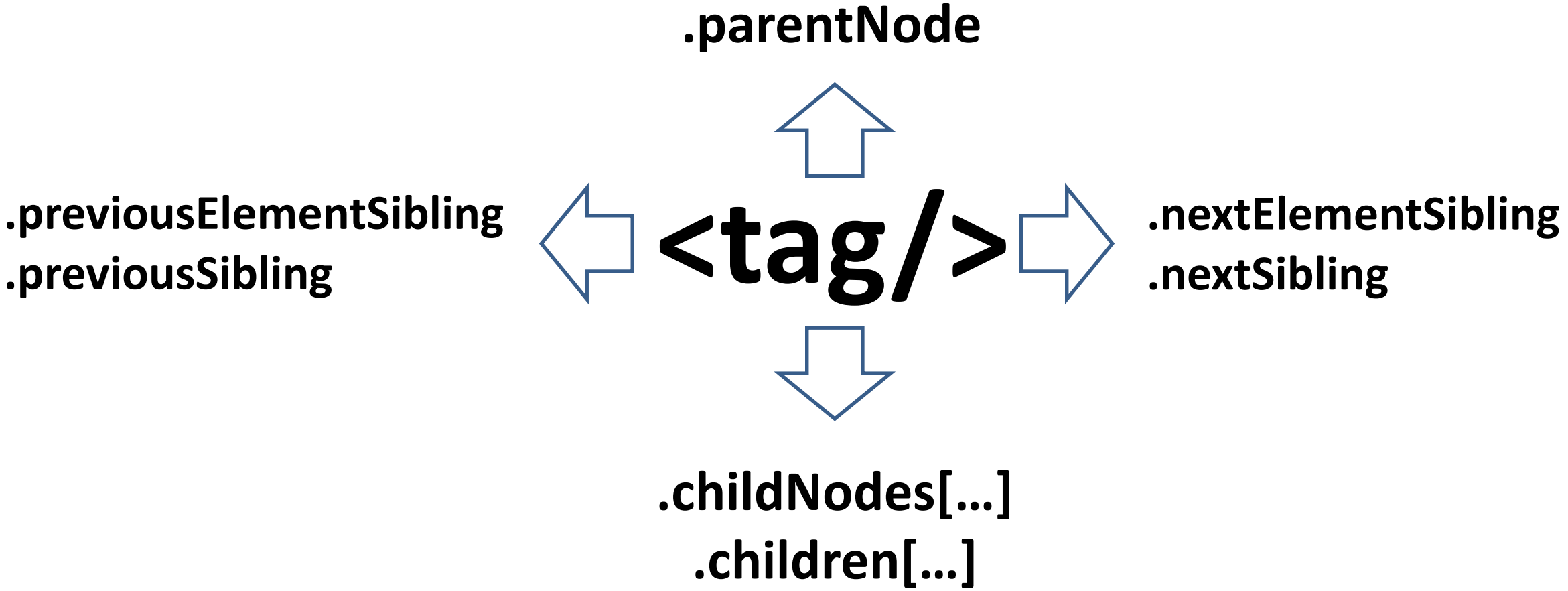
*В JavaScript **каждый тег** дерева представлен **объектом** (часто используется термин: узел, node). У каждого элемента есть один родительский элемент, и множество дочерних элементов (от 0 до ∞).*

Node/Узел/Тег/Элемент

Каждый тег представлен объектом

Воздействие на свойства и методы которого позволяют управлять внешним видом тега на странице.

Свойства элементов HTML-документа



Каждый объект (элемент, тег) имеет среди своих свойств те которые хранят ссылку на родительский элемент (**parentNode**), на соседние элементы (**previousElementSibling** и **nextElementSibling**) и на перечень потомков (**childNodes** и **children**)

Свойства элементов HTML-документа

<tag/>

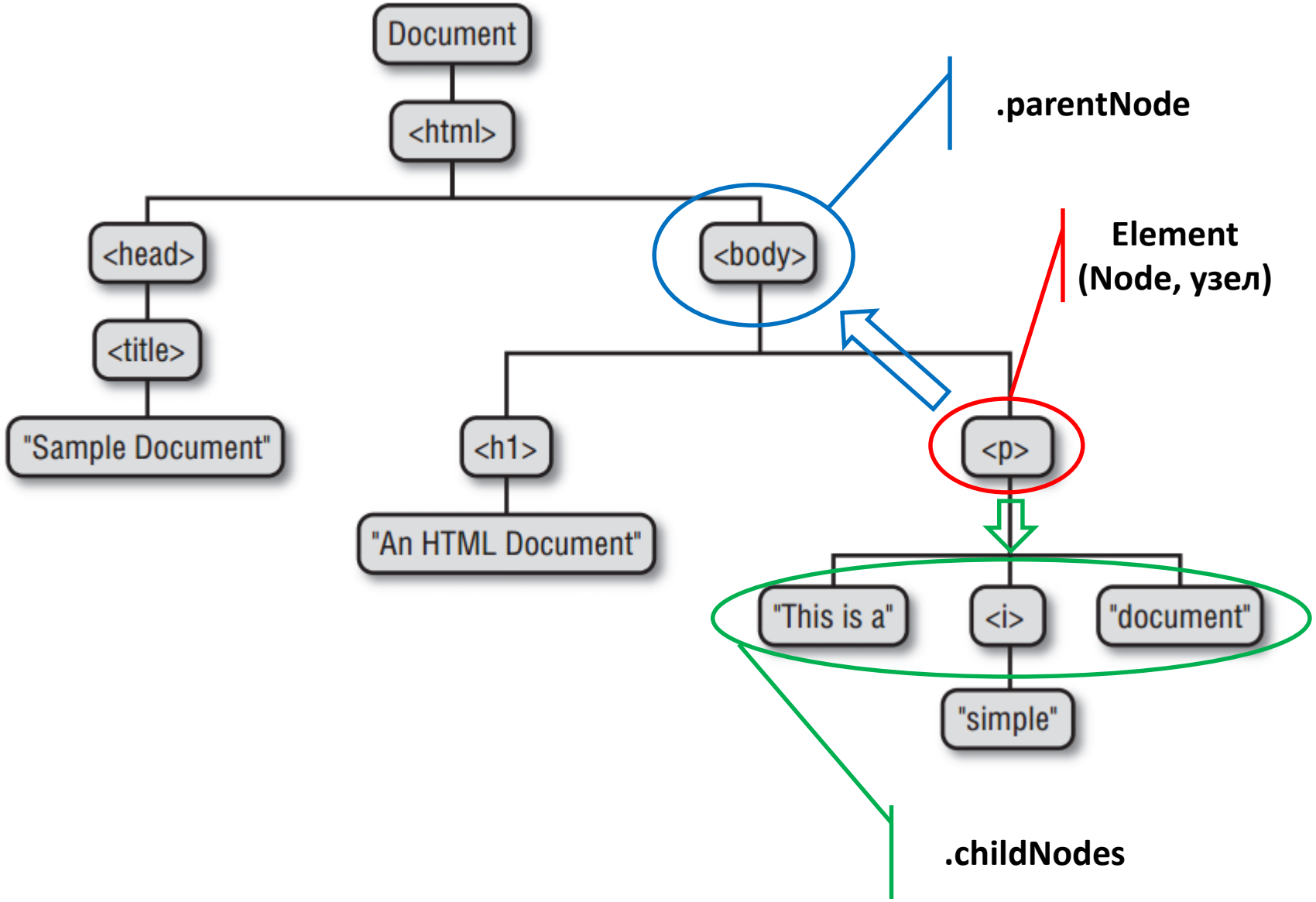
.id
.innerHTML
.className
.classList[…]
.attributes[…]
.style { … }

.onclick
.ondblclick
.onmouseenter

.appendChild()
.insertBefore()
.remove()
.insertAdjacentHTML()
.insertAdjacentElement()
.insertAdjacentText()
…

Также среди свойств объекта (элемента) есть те которые позволяют управлять содержимым (атрибутами, стилями) или подпиской на событиями, а также ряд методов позволяющих добавлять/удалять элементы, и искать вложенные элементы.

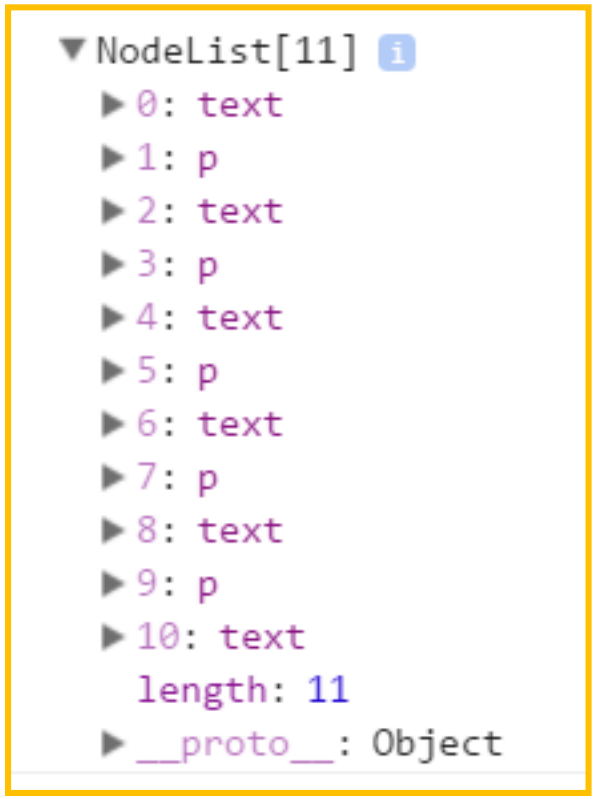
DOM – Document Object Model



window.document
корень дерева документа

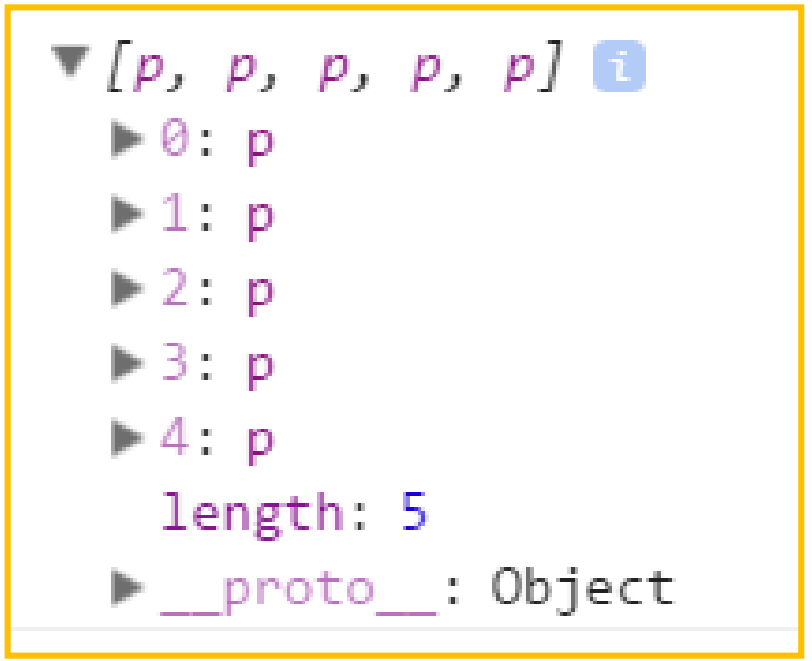
window.document.childNodes (или **children**)– массив с тегами верхнего уровня (т.е. **html** и **doctype**).

Свойство .children – тоже что и .childNodes
но без «текстовых фрагментов»



.childNodes

```
<div>
    <p>Text #1</p>
    <p>Text #2</p>
    <p>Text #3</p>
    <p>Text #4</p>
    <p>Text #5</p>
</div>
```



.children

Когда выполняется код в теге <script> ?

JavaScript в HTML

<script></script>

Тег скрипт может быть размещен в любом месте HTML-документа, с помощью него можно либо непосредственно писать JS код, либо подключать внешний файл с кодом. Однако….

JavaScript в HTML

```
<script>
    abc.style.color     = "red";
    abc.style.border    = "3px dotted green";
    abc.style.textAlign = "center";
    abc.innerHTML       = "Hello world!!!";
</script>
<h1 id="abc"></h1>
```

```
<h1 id="abc"></h1>
<script>
    abc.style.color     = "red";
    abc.style.border    = "3px dotted green";
    abc.style.textAlign = "center";
    abc.innerHTML       = "Hello world!!!";
</script>
```

Код из тега script выполняется в тот момент когда браузер дойдёт до тега, если к этому моменту браузер еще не успел обработать разметку, то нашему коду не с чем будет работать.

JavaScript в HTML

Разрешить это неудобство (с выполнением кода сразу, а не когда страница полностью загрузится) можно разными способами, например:

1. Разместить весь код в конце документа;
2. Разместить весь код во внешнем файле и подключить его с атрибутом **defer**;
3. Использовать события **onLoad** или **onDOMContentLoaded** (эти варианты мы рассмотрим детальнее когда будет говорить о событиях).

```
<script defer src="scripts/async.js"></script>
```

*Атрибут **defer** откладывает выполнение скрипта до тех пор, пока вся страница не будет загружена полностью. Работает только для внешних (подключаемых) файлов.*

Как добраться (найти) до тега?

Hello, DOM!

Praesent pharetra porttitor sem sit amet viverra. Nulla commodo feugiat orci ac dapibus. Mauris eleifend tempus tortor in imperdiet. Etiam eu iaculis justo. Nam egestas iaculis tincidunt. In rhoncus ante ipsum, vitae volutpat nulla aliquam a. Nam pretium, nulla id scelerisque bibendum, lacus enim aliquet enim, a auctor tortor turpis ut massa. In hac habitasse platea dictumst. Curabitur ante purus, luctus a felis in, vehicula feugiat sapien. Vestibulum quis sem sed erat gravida interdum a tempor odio.

1. Proin ut consectetur velit, a faucibus eros.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur posuere enim et sem interdum facilisis. Morbi augue nunc, mollis eget libero non, iaculis dictum nibh. Duis vitae ornare ex. Vestibulum est leo, mattis ac odio in, venenatis euismod est. Aenean diam augue, molestie sit amet tincidunt a, blandit at magna. Donec lacinia lectus tincidunt, lobortis tortor ac, malesuada sapien. Pellentesque fermentum finibus neque, et hendrerit enim fermentum a. Vestibulum congue purus et enim tincidunt, quis commodo libero ultrices. Quisque ornare id ipsum efficitur gravida. Nam varius tellus a libero molestie, sit amet iaculis diam suscipit. Phasellus lobortis, turpis quis consequat ultrices, mi justo venenatis eros, ac dignissim metus est in odio. Proin porttitor lacus libero, quis tincidunt elit consectetur ac.

Praesent at velit eget nunc iaculis vestibulum quis ac turpis. Praesent vehicula, quam id hendrerit faucibus, diam leo venenatis dolor, non varius velit odio sit amet ante. Vestibulum fermentum tortor id lectus viverra lobortis. Maecenas interdum risus non varius pulvinar. Phasellus posuere tincidunt blandit. Aenean ac lorem a risus malesuada faucibus. Donec lacinia lectus tincidunt, lobortis tortor ac, malesuada sapien. Pellentesque fermentum finibus neque, et hendrerit enim fermentum a. Vestibulum congue purus et enim tincidunt, quis commodo libero ultrices. Nullam ultricies est dolor, non efficitur nulla rutrum eget. In hac habitasse platea dictumst. Maecenas erat nibh, gravida a felis nec, elementum faucibus sem. In commodo, lectus quis malesuada sodales, dolor neque rutrum lectus, ut porta quam nisi ac libero. Nunc posuere ultrices pharetra.

Aliquam id ex ligula. Vestibulum est leo, mattis ac odio in, venenatis euismod est. Aenean diam augue, molestie sit amet tincidunt a, blandit at magna. Donec lacinia lectus tincidunt, lobortis tortor ac, malesuada sapien. Pellentesque fermentum finibus neque, et hendrerit enim fermentum a. Vestibulum congue purus et enim tincidunt, quis commodo libero ultrices. Duis feugiat risus sem. Morbi tincidunt risus sed vehicula mollis. Curabitur consequat pharetra urna id ornare. Vestibulum vitae nibh nibh.

2. Mauris aliquet volutpat arcu vel elementum.

Etiam blandit mauris nec ex gravida rutrum. Nam consequat diam a pretium iaculis. Donec nunc purus, semper vel nisi vitae, mattis efficitur tortor. Praesent laoreet facilisis leo, id pellentesque ante aliquam pulvinar. Sed id ligula in magna dapibus pulvinar a in lectus. Curabitur euismod bibendum rhoncus. Mauris a nibh et dolor pretium maximus lobortis sed nunc. Nam suscipit auctor commodo. Pellentesque accumsan porta risus, vel interdum nibh lacinia eu. Vestibulum nunc velit, semper non velit ut, mollis tristique tortor.

Vivamus maximus massa at blandit ultrices. Donec ac urna quis neque rutrum maximus. Pellentesque in lacus placerat enim iaculis malesuada. Cras consectetur purus in mauris imperdiet, vel elementum velit sagittis. In pulvinar sapien vitae nulla pellentesque dictum. Etiam libero metus, eleifend vel ullamcorper in, condimentum ut odio. Praesent condimentum sem quis mauris vehicula ornare pellentesque a magna. Curabitur pharetra ut urna sed molestie. Nulla quis sapien lectus. Integer sed elit aliquet erat malesuada dictum nec at arcu.

Используйте заготовку:
./source/example_1
Запустите при помощи serve

Теги у которых есть атрибут id доступны сразу как переменные ссылкой на объект

*Но только если **id** состоит из допустимых для имён переменных в **JavaScript** символов.*

Поиск элементов в документе

Выбор элемента с которым проводить манипуляции самая часто выполняемая операция в JS.

Выбор элемента по атрибуту id:

```
document.getElementById("some_id");
```

Возвращает один элемент атрибут (свойство) ***id*** *равно «**some_id**». Если такого элемента нет в документе, то возвращается **null**.*

Подробнее: https://learn.javascript.ru/searching-elements-dom

Поиск элементов в документе

Выбор элементов по названию тега:

```
document.getElementsByTagName("tag_name");
```

Выбор элементов по атрибуту name:

```
document.getElementsByName("attr_name");
```

Выбор элементов по атрибуту class:

```
document.getElementsByClassName("class_name");
```

Все эти функции возвращают псевдомассив с теми элементами которые подошли под условие.

Подробнее: https://learn.javascript.ru/searching-elements-dom

Поиск элементов в документе

Выбор всех элементов которые соответствуют CSS селектору:

```
document.querySelectorAll("css_selector");
```

Возвращает псевдомассив с теми элементами которые подошли под условие css-селектора.

```
document.querySelector("css_selector");
```

*Возвращает первый найденный элемент который подошел под условие css-селектора (или **null** если ничего не найдено).*

Подробнее: https://learn.javascript.ru/searching-elements-dom

Вложенный поиск, т.е. поиск в результатах поиска

```
54
55      let result_1 = document.querySelectorAll('p');
56
57      let columnOne = document.getElementById('column-one');
58      let result_2  = columnOne.querySelectorAll('p');
59
60      console.log('Result 1:', result_1);
61      console.log('Result 2:', result_2);
62
```

```
Result 1: ▶ NodeList(7) [p, p, p#special, p, p, p, p]
Result 2: ▶ NodeList(3) [p, p#special, p]
```

Функции поиска элементов можно применять к любому существующему элементу, а не только к документу. Когда функция поиска применяется к конкретному элементу, то поиск осуществляется среди его потомков.

«Живые» и статические коллекции

```
54
55      let result_1    = document.getElementsByTagName('p');
56      let result_2    = document.querySelectorAll('p');
57
58      special.remove();
59
60      console.log('Result 1:', result_1);
61      console.log('Result 2:', result_2);
62
```

```
Result 1: ▶ HTMLCollection(6) [p, p, p, p, p, p]
Result 2: ▶ NodeList(7) [p, p, p#special, p, p, p, p]
```

Живые (**Live**) коллекции изменяют свой состав в зависимости от изменений в документа. Статические (**Static**) коллекции не изменяют свой состав после формирования.

Живые и статические коллекции

.querySelector() и .querySelectorAll()

Возвращают статические коллекции, т.е. «слепок» на момент вызова функции.

.getElementsBy…()

Возвращают живые коллекции, которые всегда актуальны. Т.е. массив с результатом работы этих функций всегда будет содержать актуальное количество результатов, что бы не происходило с документом.

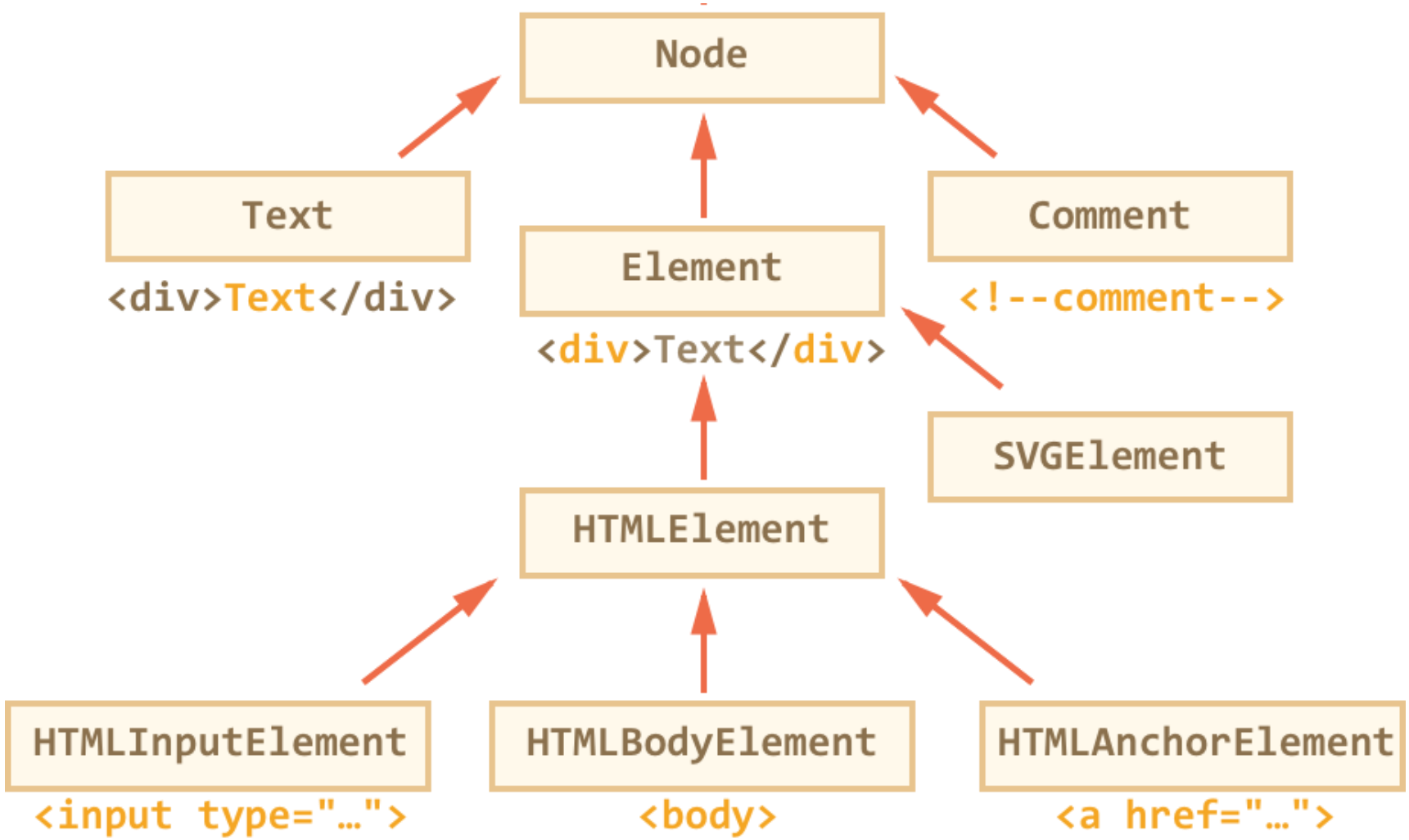
Подробнее: https://learn.javascript.ru/searching-elements-dom

С живыми коллекциями нужно быть осторожным *в том случае если вы перебираете её в цикле и изменяете её состав.*

```
54
55      let result  = document.getElementsByTagName('p');
56
57      result = [...result];
58
59      console.log('Result:', result);
60
```

Однако, и живую и статическую коллекцию можно конвертировать в классический массив.

Типы объектов в иерархии документа



Подробнее: https://learn.javascript.ru/basic-dom-node-properties

Как изменить тег?

Свойство .innerHTML хранит содержимое тега

Свойство .**innerHTML** – можно не только считывать но и устанавливать. Изменение свойства .**innerHTML** – автоматически влечёт перерисовку документа.

Полезные свойства элементов

.className *– свойство содержит полный список всех классов которые присвоены тегу (одной строкой).*

.classList *– свойство содержит список всех классов которые присвоены тегу (в виде массива).*

.classList.add('cat') *– метод добавляет класс к тегу (если есть другие классы то они остаются).*

.classList.remove('cat') *– метод удаляет класс у тегу (если есть другие классы то они не затрагиваются).*

.classList.toggle('cat') *– метод удаляет класс у тегу, если он есть, или добавляет класс, если его нет.*

.classList.contains('cat') *– метод проверяет наличие у тега заданного класса (возвращает true/false).*

.style *– свойство определяющее объект со всеми поддерживаемыми браузером стилевые свойства (CSS).*

.attributes *– хранит коллекцию с атрибутами тега.*

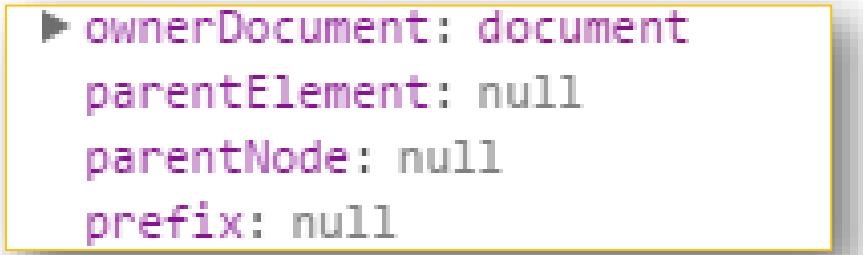
Подробнее: https://learn.javascript.ru/styles-and-classes

Как удалить тег?

Удаление элементов из дерева документа

```
54
55      let tag = document.getElementById('special');
56
57      tag.remove();
58
59      console.dir(tag);
60
```

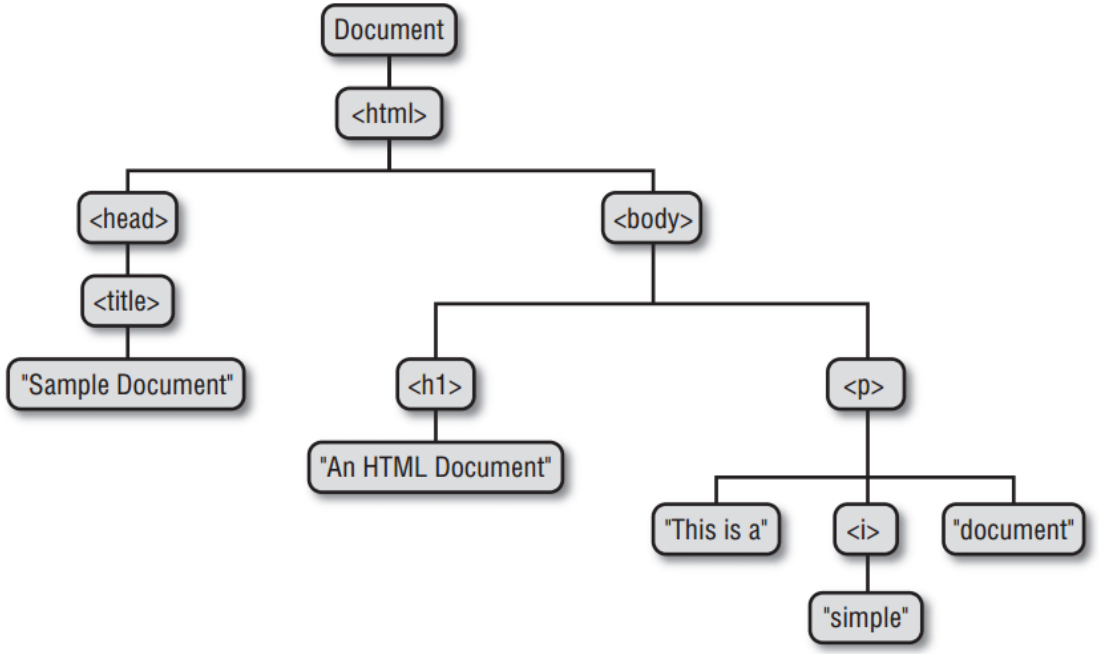
Удалить элемент из дерева документа можно вызывав у него метод **.remove()**, при этом все его дочерние элементы также исчезнут со странице. Однако сам объект-тег не уничтожается. Его можно использовать в дальнейшем.



Как создать и добавить тег?

Добавление новых элементов к дереву документа

Вставить новый элемент в документ, можно прикрепив его к какому-либо существующему элементу. Т.е. прикрепить его к родительскому элементу (другими словами: сделать его дочерним для существующего элемента).



Добавление новых элементов к дереву документа

Простейший вариант: просто добавить текстовую строку с нужными данным к свойству **.innerHTML**. Однако это не самый удобный вариант.

Добавление новых элементов к дереву документа (первое поколение)

document.createElement() – создаёт новый элемент (по имени тега). Этот элемент, после создания, еще не включен в дерево. Но его свойства уже можно изменять.

.appendChild() – добавляет элемент к существующему, в качестве последнего потомка. Может быть вызвана для любого существующего тега (даже если он не входит в дерево – другими словами можно формировать ветку еще до того как «присоединять» её к дереву).

.insertBefore() – добавляет элемент в качестве дочернего, при этом позволяет указать перед каким из, уже существующих, потомков новый элемент должен быть размещён.

Подробнее: https://learn.javascript.ru/modifying-document

Добавление новых элементов к дереву документа (второе поколение)

```
<!-- beforebegin -->
<p>
  <!-- afterbegin -->
  foo
  <!-- beforeend -->
</p>
<!-- afterend -->
```

*Варианты позиции для методов группы **.insertAdjacent**...()*

`tag.insertAdjacentElement(position, element)`

*добавляет **элемент** к **существующему**, в указанную **позицию**.*

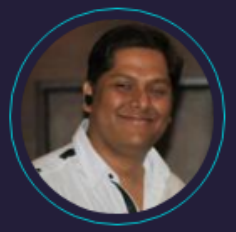
Подробнее: https://learn.javascript.ru/modifying-document

*Также существуют методы **tag.insertAdjacentHTML() и tag.insertAdjacentText()***

Немного практики

Hello, DOM!

PRO



Mr Kian Ørmen

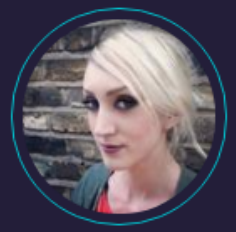
ÅKREHAMN, NORWAY

kian.ormen@example.com

27.02.1952 | 68 Y.O.

Message

PRO



Miss Silje Jørgensen

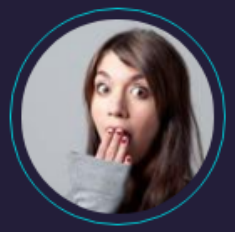
SUNDBY, DENMARK

silje.jorgensen@example.com

06.03.1997 | 23 Y.O.

Message

PRO



Mrs Romy Dubois

PARIS, FRANCE

romy.dubois@example.com

14.11.1997 | 23 Y.O.

Message

PRO

Mr Connor Rogers

CORK, IRELAND

connor.rogers@example.com

08.07.1995 | 25 Y.O.

Message

PRO

Mrs Kristin Walters

MONAGHAN, IRELAND

kristin.walters@example.com

15.07.1969 | 51 Y.O.

Message

PRO

Ms Hilde Risvik

JEVNAKER, NORWAY

hilde.risvik@example.com

21.11.1957 | 63 Y.O.

Message

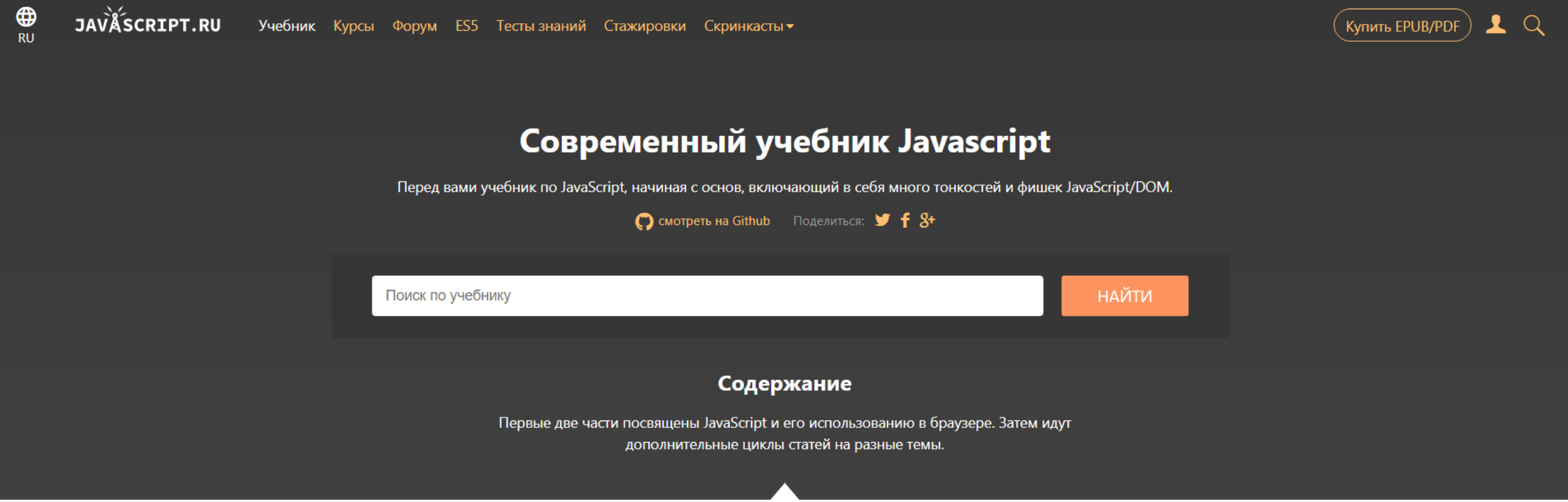
Используйте заготовку:
./source/example_2
Запустите при помощи serve

Выведем в подготовленную разметку данные пользователей полученные от сервиса **https://randomuser.me/**

Домашнее задание
/узнать

Узнать о следующих свойствах:

.firstChild;
.lastChild;
.nextSibling;
.previousSibling;
.nextElementSibling;
.previousElementSibling;



Предварительные знания – лучший помощник в обучении, поэтому к следующему занятию жду, что **пройдёте разделы 2-й части 2.1-2.5, 3.1-3.5**

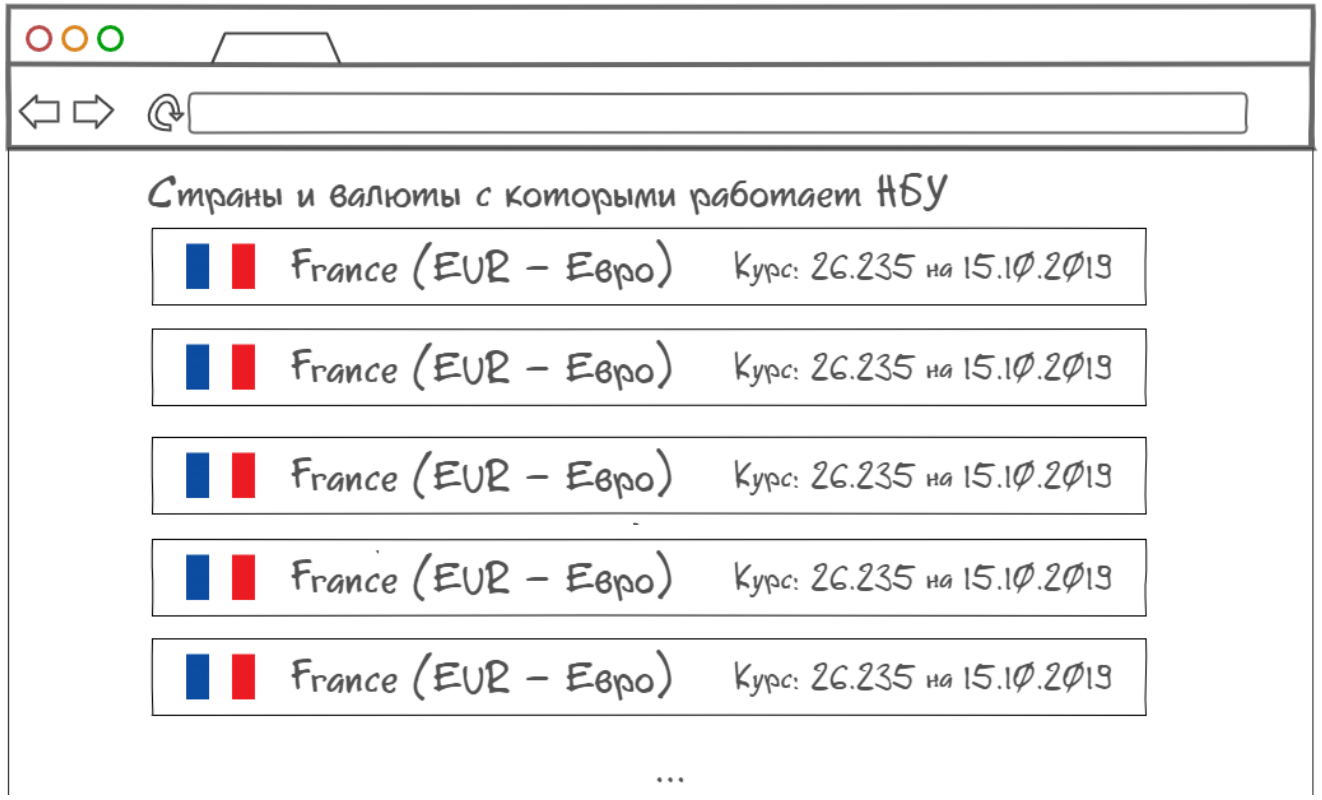
http://learn.javascript.ru/

Домашнее задание
/сделать

Домашнее задание #G.1

Воспользуйтесь API дающее информацию о странах мира:
https://restcountries.eu/rest/v2/all

Так же воспользуйтесь API НБУ по курсам валют:
https://bank.gov.ua/NBUStatService/v1/statdirectory/exchange?json



Выведите в разметку перечень стран с валютами которых работает НБУ (если несколько стран имеют общую валюту - выводите их все, пример: зона Евро, или страны использующие USD). Пример разметки на wireframe. P*азметку необходимо подготовить))*