document главная «точка входа» в DOM. Из него мы можем получить доступ к любому узлу.

Самые верхние элементы дерева доступны как свойства объекта document:

**<html> = document.documentElement**

Самый верхний узел документа: document.documentElement. В DOM он соответствует тегу <html>.

**<body> = document.body**

Другой часто используемый DOM-узел – узел тега <body>: document.body.

**<head> = document.head**

Тег <head> доступен как document.head.

**Есть одна тонкость: document.body может быть равен null**

Нельзя получить доступ к элементу, которого ещё не существует в момент выполнения скрипта.

В частности, если скрипт находится в <head>, document.body в нём недоступен, потому что браузер его ещё не прочитал.

**В мире DOM null означает «не существует»**

В DOM значение null значит «не существует» или «нет такого узла».

Здесь и далее мы будем использовать два принципиально разных термина:

* **Дочерние узлы (или дети)** – элементы, которые являются непосредственными детьми узла. Другими словами, элементы, которые лежат непосредственно внутри данного. Например, <head> и <body> являются детьми элемента <html>.
* **Потомки** – все элементы, которые лежат внутри данного, включая детей, их детей и т.д.

**childNodes –** коллекция, содержит список всех детей, **включая текстовые узлы.**

**firstChild и lastChild** – свойства обеспечивающие быстрый доступ к первому и последнему дочернему элементу.

Они, по сути, являются всего лишь сокращениями. Если у тега есть дочерние узлы, условие ниже всегда верно:



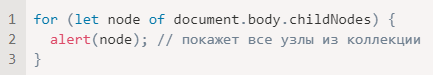
**elem.hasChildNodes() -** функция для проверки наличия дочерних узлов.

**[DOM-коллекции](https://learn.javascript.ru/dom-navigation" \l "dom-kollektsii)**

**childNodes** похож на массив. На самом деле это не массив, а коллекция – особый перебираемый объект-псевдомассив.

И есть два важных следствия из этого:

1. Для перебора *коллекции* мы можем использовать for..of:



Это работает, потому что коллекция является перебираемым объектом (есть требуемый для этого метод Symbol.iterator).

1. Методы массивов не будут работать, потому что коллекция – это не массив:



Если нам хочется использовать именно методы массива, то мы можем создать настоящий массив из коллекции, используя Array.from:



**DOM-коллекции – только для чтения**

DOM-коллекции, и даже более – *все* навигационные свойства, перечисленные в этой главе, доступны только для чтения.

Мы не можем заменить один дочерний узел на другой, просто написав childNodes[i] = ....

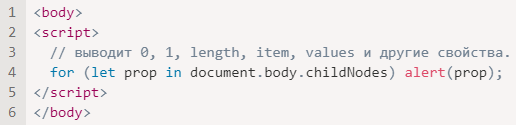
Для изменения DOM требуются другие методы.

**DOM-коллекции живые**

Почти все DOM-коллекции, за небольшим исключением, *живые*. Другими словами, они отражают текущее состояние DOM. Если мы сохраним ссылку на elem.childNodes и добавим/удалим узлы в DOM, то они появятся в сохранённой коллекции автоматически.

**Не используйте цикл for..in для перебора коллекций**

Цикл for..in перебирает все перечисляемые свойства. А у коллекций есть некоторые «лишние», редко используемые свойства, которые обычно нам не нужны:



## **[Соседи и родитель](https://learn.javascript.ru/dom-navigation" \l "sosedi-i-roditel)**

*Соседи* – это узлы, у которых один и тот же родитель.

Например, здесь <head> и <body> соседи:

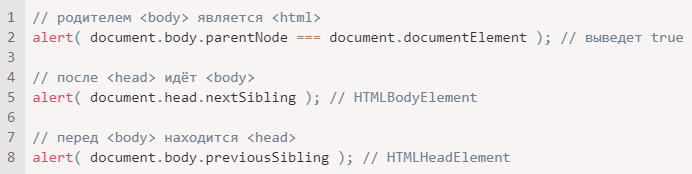


* говорят, что <body> – «следующий» или «правый» сосед <head>
* также можно сказать, что <head> «предыдущий» или «левый» сосед <body>.

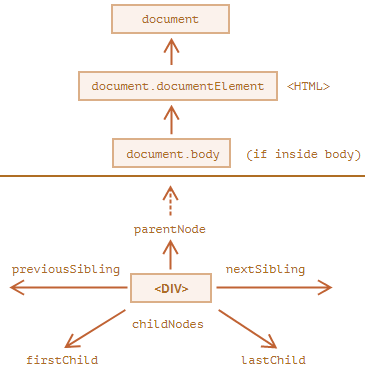
nextSibling – свойство -следующий узел того же родителя (следующий сосед);

previousSibling – предыдущий узел того же родителя

parentNode – родитель



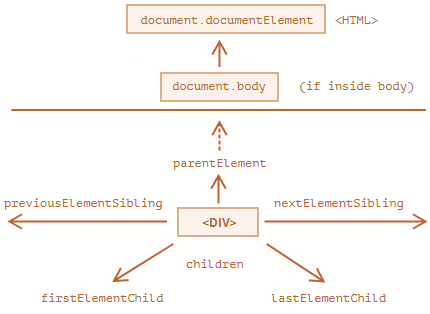
## **[Навигация только по элементам](https://learn.javascript.ru/dom-navigation" \l "navigatsiya-tolko-po-elementam)**



Все свойства выше содержат все узлы - и текстовые узлы и узлы-элементы и узлы-комментарии, если они есть.

Но для большинства задач текстовые узлы и узлы-комментарии нам не нужны. Мы хотим манипулировать узлами-элементами, которые представляют собой теги и формируют структуру страницы.

Поэтому давайте рассмотрим дополнительный набор ссылок, которые учитывают только *узлы-элементы*:



Эти ссылки похожи на те, что раньше, только в ряде мест стоит слово Element:

* children – коллекция детей, которые являются элементами.
* firstElementChild, lastElementChild – первый и последний дочерний элемент.
* previousElementSibling, nextElementSibling – соседи-элементы.
* parentElement – родитель-элемент.

**Таблицы**

Некоторые типы DOM-элементов предоставляют для удобства дополнительные свойства, специфичные для их типа.

**Элемент <table>**, в дополнение к свойствам, о которых речь шла выше, поддерживает следующие:

* table.rows – коллекция строк <tr> таблицы.
* table.caption/tHead/tFoot – ссылки на элементы таблицы <caption>, <thead>, <tfoot>.
* table.tBodies – коллекция элементов таблицы <tbody> (по спецификации их может быть больше одного).

**<thead>, <tfoot>, <tbody>** предоставляют свойство rows:

* tbody.rows – коллекция строк <tr> секции.

**<tr>:**

* tr.cells – коллекция <td> и <th> ячеек, находящихся внутри строки <tr>.
* tr.sectionRowIndex – номер строки <tr> в текущей секции <thead>/<tbody>/<tfoot>.
* tr.rowIndex – номер строки <tr> в таблице (включая все строки таблицы).

**<td> and <th>:**

* td.cellIndex – номер ячейки в строке <tr>.

Спецификация: [tabular data](https://html.spec.whatwg.org/multipage/tables.html).