Получив DOM-узел, мы можем перейти к его ближайшим соседям используя навигационные ссылки.

Есть два основных набора ссылок:

* Для всех узлов: parentNode, childNodes, firstChild, lastChild, previousSibling, nextSibling.
* Только для узлов-элементов: parentElement, children, firstElementChild, lastElementChild, previousElementSibling, nextElementSibling.

Некоторые виды DOM-элементов, например таблицы, предоставляют дополнительные ссылки и коллекции для доступа к своему содержимому.

**Таблицы всегда содержат <tbody>**

Важный «особый случай» – работа с таблицами. По стандарту DOM у них должен быть <tbody>, но в HTML их можно написать (официально) без него. В этом случае браузер добавляет <tbody> в DOM самостоятельно.

<table id="table"><tbody><tr><td>1</td></tr></tbody></table>

Есть 6 основных методов поиска элементов в DOM:

| Метод | Ищет по... | Ищет внутри элемента? | Возвращает живую коллекцию? |
| --- | --- | --- | --- |
| querySelector | CSS-selector | ✔ | - |
| querySelectorAll | CSS-selector | ✔ | - |
| getElementById | id | - | - |
| getElementsByName | name | - | ✔ |
| getElementsByTagName | tag or '\*' | ✔ | ✔ |
| getElementsByClassName | class | ✔ | ✔ |

Безусловно, наиболее часто используемыми в настоящее время являются методы querySelector и querySelectorAll, но и методы getElement(s)By\* могут быть полезны в отдельных случаях, а также встречаются в старом коде.

Кроме того:

* Есть метод elem.matches(css), который проверяет, удовлетворяет ли элемент CSS-селектору.
* Метод elem.closest(css) ищет ближайшего по иерархии предка, соответствующему данному CSS-селектору. Сам элемент также включён в поиск.

И, напоследок, давайте упомянем ещё один метод, который проверяет наличие отношений между предком и потомком:

* elemA.contains(elemB) вернёт true, если elemB находится внутри elemA (elemB потомок elemA) или когда elemA==elemB.