

5. Поиск: getElement*, querySelector*

document.getElementById или просто id

Если у элемента есть атрибут `id`, то мы можем получить его вызовом `document.getElementById(id)`, где бы он ни находился.

Значение `id` должно быть уникальным. В документе может быть только один элемент с данным `id`.

Если в документе есть несколько элементов с одинаковым значением `id`, то поведение методов поиска непредсказуемо. Браузер может вернуть любой из них случайным образом. Поэтому, пожалуйста, придерживайтесь правила сохранения уникальности `id`.

Метод `getElementById` можно вызвать только для объекта `document`. Он осуществляет поиск по `id` по всему документу.

querySelectorAll

Самый универсальный метод поиска – это `elem.querySelectorAll(css)`, он возвращает все элементы внутри `elem`, удовлетворяющие данному CSS-селектору.

Следующий запрос получает все элементы ``, которые являются последними потомками в ``

Псевдоклассы в CSS-селекторе, в частности `:hover` и `:active`, также поддерживаются. Например, `document.querySelectorAll(':hover')` вернёт коллекцию (в порядке вложенности: от внешнего к внутреннему) из текущих элементов под курсором мыши.

querySelector

Метод `elem.querySelector(css)` возвращает первый элемент, соответствующий данному CSS-селектору.

Иначе говоря, результат такой же, как при вызове `elem.querySelectorAll(css)[0]`, но он сначала найдёт все элементы, а потом возьмёт первый, в то время как `elem.querySelector` найдёт только первый и остановится. Это быстрее, кроме того, его короче писать.

matches

Предыдущие методы искали по DOM.

Метод `elem.matches(css)` ничего не ищет, а проверяет, удовлетворяет ли `elem` CSS-селектору, и возвращает `true` или `false`.

Этот метод удобен, когда мы перебираем элементы (например, в массиве или в чём-то подобном) и пытаемся выбрать те из них, которые нас интересуют.

closest

Предки элемента – родитель, родитель родителя, его родитель и так далее. Вместе они образуют цепочку иерархии от элемента до вершины.

Метод `elem.closest(css)` ищет ближайшего предка, который соответствует CSS-селектору. Сам элемент также включается в поиск.

Другими словами, метод `closest` поднимается вверх от элемента и проверяет каждого из родителей. Если он соответствует селектору, поиск прекращается. Метод возвращает либо предка, либо `null`, если такой элемент не найден.

getElementsByTagName*

Существуют также другие методы поиска элементов по тегу, классу и так далее.

На данный момент, они скорее исторические, так как `querySelector` более чем эффективен.

Здесь мы рассмотрим их для полноты картины, также вы можете встретить их в старом коде.

- `elem.getElementsByTagName(tag)` ищет элементы с данным тегом и возвращает их коллекцию. Передав "*" вместо тега, можно получить всех потомков.
- `elem.getElementsByClassName(className)` возвращает элементы, которые имеют данный CSS-класс.
- `document.getElementsByName(name)` возвращает элементы с заданным атрибутом `name`. Очень редко используется.

Безусловно, наиболее часто используемыми в настоящее время являются методы `querySelector` и `querySelectorAll`, но и методы `getElementBy*` 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.
-