Лабораторная работа № 5. Объектная модель документа. Доступ к элементам web-страницы

Цель: познакомиться с Document Object Model (DOM); изучить способы доступа к элементам web-страницы; изучить основные свойства DOM-элементов: **tagName**, **style**, **innerHTML**, **className**.

Теория

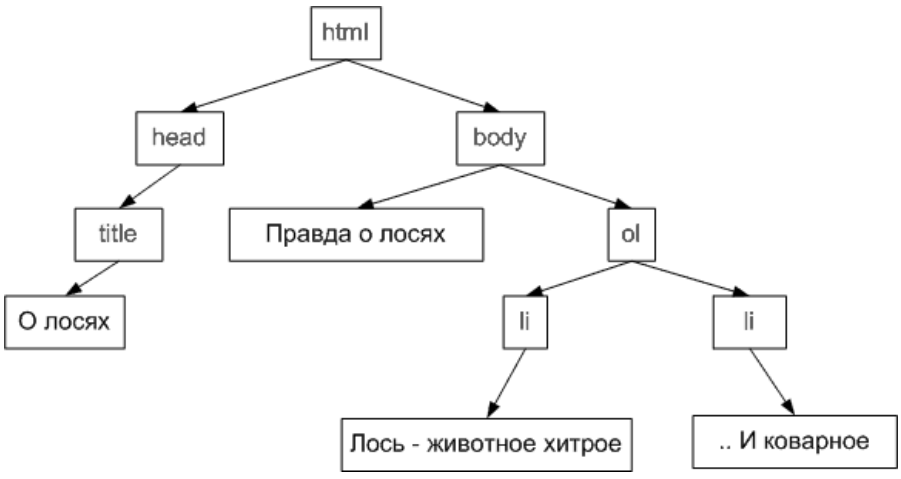
DOM (Объектная модель документа) – представление документа в виде дерева тегов.

Каждый HTML-тег образует отдельный элемент-узел, каждый фрагмент текста – текстовый элемент.

Построим дерево DOM для следующего документа.

```
<html>
    <head>
        <title>
            О лосях
        </title>
    </head>
    <body>
        Правда о лосях.
        <ol>
            <li>
                Лось - животное хитрое
            </li>
            <li>
                .. И коварное
            </li>
        </ol>
    </body>
</html>
```

Корневым элементом иерархии является html. У него есть два потомка: первый – head, второй – body. И так далее, каждый вложенный тег является потомком тега выше.



Каждый DOM-элемент является объектом и предоставляет свойства для манипуляции своим содержимым, для доступа к родителям и потомкам.

Для манипуляций с DOM используется объект **document**. Используя **document**, можно получать нужный элемент дерева и менять его содержание.

Например, следующий код получает первый элемент с тэгом **ol**, последовательно удаляет два элемента списка и затем добавляет их в обратном порядке:

```
var ol = document.getElementsByTagName('ol')[0]
var hiter = ol.removeChild(ol.firstChild)
var kovaren = ol.removeChild(ol.firstChild)
ol.appendChild(kovaren)
ol.appendChild(hiter)
```

Доступ к элементам DOM.

У каждого элемента в DOM-модели есть тип. Его номер хранится в атрибуте **elem.nodeType**. HTML-теги имеют номер типа равный 1, текстовые элементы – 3.

Следующий пример при нажатии на кнопку выведет типы **document.documentElement**, а затем тип последнего потомка узла **document.body**. Им является текстовый узел.

```
<html>
  <body>
     <script>
       function go() {
         alert(document.documentElement.nodeType)
         alert(document.body.lastChild.nodeType)
       }
     </script>
     <input type="button" onclick="go()" value="Go"/>
     Текст
  </body>
</html>
```

С вершины дерева можно пойти дальше вниз. Для этого каждый DOM-узел содержит массив всех потомков, отдельно – ссылки на первого и последнего потомка и еще ряд полезных свойств.

Все дочерние элементы, включая текстовые, находятся в массиве **childNodes**.

В следующем примере цикл перебирает всех потомков **document.body**.

```
for(var i=0; i<document.body.childNodes.length; i++) {
    var child = document.body.childNodes[i]
    alert(child.tagName)
}
```

Свойства **firstChild** и **lastChild** показывают на первый и последний дочерние элементы и равны null, если потомков нет.

Свойство **parentNode** указывает на родителя. Например, для <body> таким элементом является <html>:

Свойства **previousSibling** и **nextSibling** указывают на левого и правого братьев узла.

Свойства элементов

У DOM-элементов есть много свойств. Обычно используется максимум треть из них. Некоторые из них можно читать и устанавливать, другие – только читать.

Рассмотрим некоторые свойства элементов, полезные при работе с DOM.

Атрибут **tagName** есть у элементов-тегов и содержит имя тега в верхнем регистре, только для чтения.

Например, `alert(document.body.tagName)`.

Свойство **style** управляет стилем. Оно аналогично установке стиля в CSS. Например, можно установить **element.style.width** для кнопки:

```
<input type="button" style="width: 300px" onclick="this.style.width = parseInt(this.style.width)-10+'px'" value="Укоротить на 10px"/>
```

Обработчик события **onclick** обращается в этом примере к свойству **this.style.width**, т.к значением **this** в обработчике события является текущий элемент (т.е сама кнопка).

Есть общее правило замены – если CSS-атрибут имеет дефисы, то для установки **style** нужно заменить их на верхний регистр букв.

Например, для установки свойства z-index в 1000, нужно поставить: `element.style.zIndex = 1000`

Свойство **innerHTML** содержит весь HTML-код внутри узла, и его можно менять.

Свойство **innerHTML** применяется, в основном, для динамического изменения содержания страницы, например:

```
document.getElementById('footer').innerHTML = '<h1>БГТУ</h1><p>ФИТ</p>'
```

Пожалуй, **innerHTML** – одно из наиболее часто используемых свойств DOM-элемента.

Свойство **className** задает класс элемента. Оно полностью аналогично html-атрибуту "class".

```
elem.className = 'newclass'
```

Задания для лабораторной работы № 5

Задания выполняются со **своей** самой первой web-страницей.

Задание 1. Вывести на экран все тэги и их номера в коллекции **all** своей первой web-страницы.

Задание 2. Вывести все дочерние элементы узла **document.body** своей первой web-страницы.

Задание 3. Вывести в окно содержимое первого элемента **span**:

- используя для доступа к элементу коллекцию **all;**
- используя частную коллекцию (span);
- используя идентификатор элемента.

Задание 4. Организовать доступ к содержимому таблицы и посчитать свой средний балл. Добавить значение среднего балла к содержимому второго элемента **span**.