JS DOM

ПОДГОТОВИЛ: АЛИШЕР ХАМИДОВ

Основные моменты:

План лекции

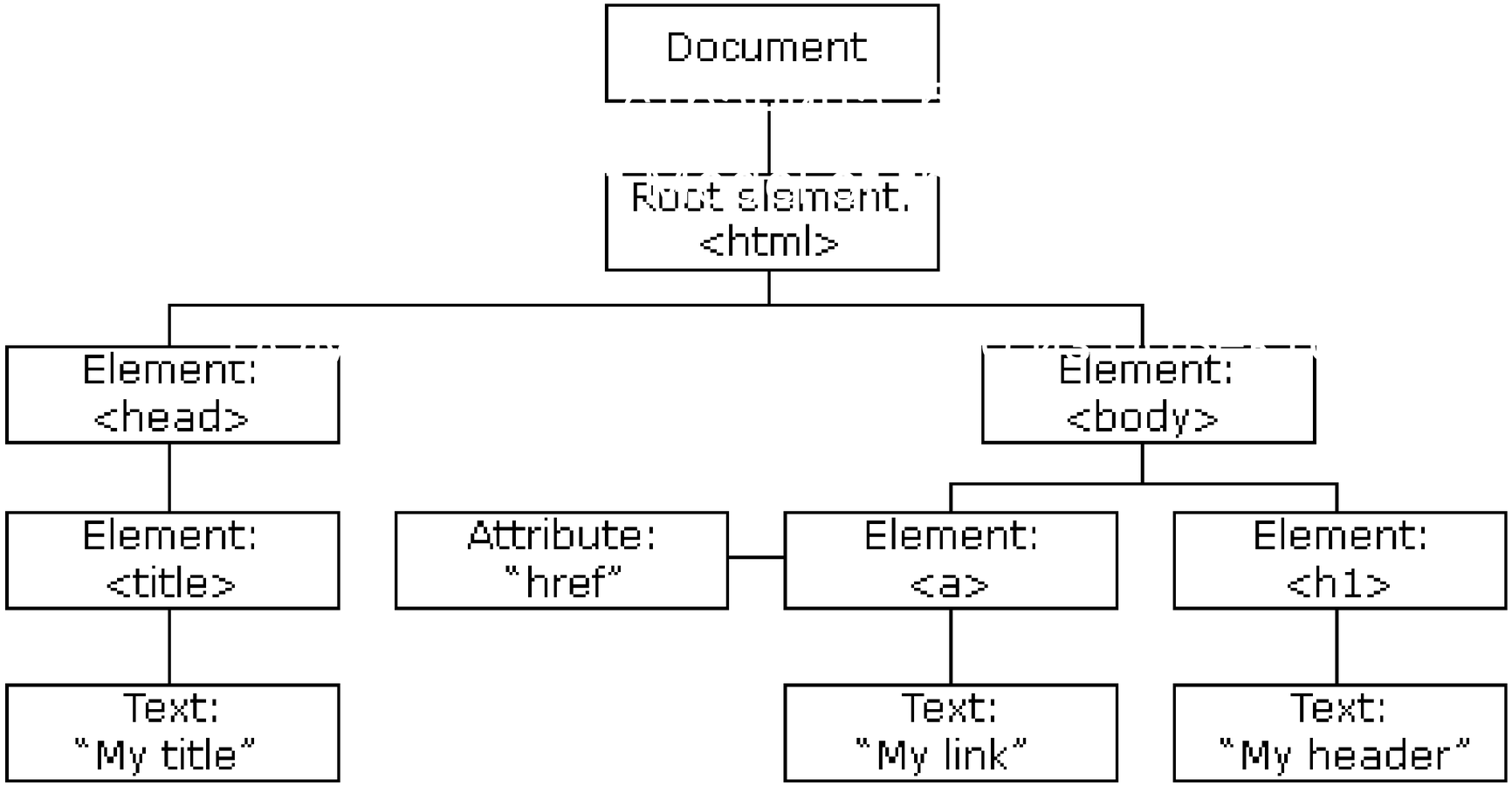
1. DOM
2. JS и DOM
3. Методы DOM

The HTML DOM (Document Object Model)

Когда загружается web страница, браузер создает Document Object Model of the page.

DOM создается как дерево из объектов.

Пример DOM-дерева



JS и DOM

Благодаря DOM JavaScript получает возможность взаимодествовать с элементами страницы и создавать динамические страницы.

JS и DOM

Появляется возможность:

- изменять элементы страницы
- изменять или удалять HTML атрибуты
- изменять CSS стили страницы
- добавлять новые HTML элементы на страницу
- реагировать на все существующие события на странице
- создавать новые HTML события на странице

HTML DOM

– это стандартная объектная модель и программный интерфейс для HTML

Он определяет:

- **Элементы HTML в качестве объектов**
- **Свойства всех HTML элементов**
- **Методы для доступа к любым HTML элементам**
- **События для всех HTML элементов**

HTML DOM

– это стандарт того, как мы можем получить(get), изменить(change) или добавить HTML элемент

DOM methods

Alisher Khamidov | 2023

getElementById

Получает доступ к элементу по id:

```
const btn = document.getElementById("btn1")

console.log(btn); // <button id="btn1">Нажми меня</button>
```

getElementByTagName

Данный метод получает доступ к элементу по тегу:

```
const text = document.getElementsByTagName("p");
console.log(text);  // HTMLCollection [p#text1, text1: p#text1]
console.log(text.text1.outerHTML); // <p id="text1">Я текст</p>
console.log(text.text1.outerText);   // Я текст
```

getElementsByClassName

Данный метод получает доступ к элементу по тегу:

```
// <span class="important-message">Важное сообщение</span>
const message = document.getElementsByClassName("important-message");
console.log(message); // HTMLCollection [span.important-message]
console.log(message[0]); // <span class="important-message">Важное
сообщение</span>
```

createElement

Создаем новый элемент:

```
const heading = document.createElement("h2");
console.log(heading); // <h2></h2>

// создали пустой заголовок, он еще никуда не помещен
```

createTextNode, appendChild

Создаем элемент и наполним текстом:

```
// создадим заголовок
const heading = document.createElement("h2");
// создадим текстовый узел
heading_text = document.createTextNode("Продолжим изучать JS");
// прикрепим к заголовку текст
heading.appendChild(heading_text);
// прикрепим  готовый заголовок к body документа
document.body.appendChild(heading);
```

appendChild

Метод appendChild() добавляет узел после последнего дочернего элемента указанного родительского узла

```
document.body.appendChild(heading);
```

Методы для изменения элементов

innerText

```
// создали элемент div
let div = document.createElement('div');
// добавили текст
div.innerText = 'We are doing good';
```

getAttribute

```
// <div id="fruit" class="apple">Апельсин</div>
const myFruit = document.getElementById("fruit");
const myAttribute = myFruit.getAttribute("class");
console.log(myAttribute); // apple
```

setAttribute

```
// <div id="fruit" class="apple">Апельсин</div>
// Зададим новое значение указанному атрибуту
myFruit.setAttribute("class", "orange");
// посмотрим, что изменилось
const myNewAttribute = myFruit.getAttribute("class");
console.log(myNewAttribute); // orange
```

innerHTML

```
// до метода: <div id="panda">

const panda = document.getElementById("panda");
panda.innerHTML = "<p> I like pandas</p>"

// элемент после применения метода
// <div id="panda"><p>I like pandas</p></div>
```

element.style.property = new style

```
// <div id="banana">Banana</div>

const banana = document.getElementById("banana");
banana.style.color="orange";

//  после применения метода текст стал желтым
```

Методы для удаления элементов

remove

```
// удаляет дочерний элемент у родителя

// <div id="garbage"> Элемент для удаления </div>
// нашли его
const garbage = document.getElementById("garbage");
// удалили
garbage.remove();
```

removeChild

```
// удаляет дочерний элемент у родителя

const myLinkList = document.getElementById(“list”);
// нашли последний элемент при помощи lastChild
const myRemovedLink = myLinkList.lastChild;
// удалили ребенка из родителя
myLinkList.removeChild(myRemoveLink);
```