



JavaScript Базовий

Модифікація DOM-дерев. CSS-стилі

JavaScript Базовий

Introduction



Охріменко Дмитро
МСТ

 _okhrimenko

 dmitriy.okhrimenko

 dokhrimenko



MCID:
9210561

JavaScript Базовий

Тема

Модифікація DOM-дерев

JavaScript Базовий

Створення вузлів

document.createElement (тег) – створення нового елемента, який вказаний у параметрах.

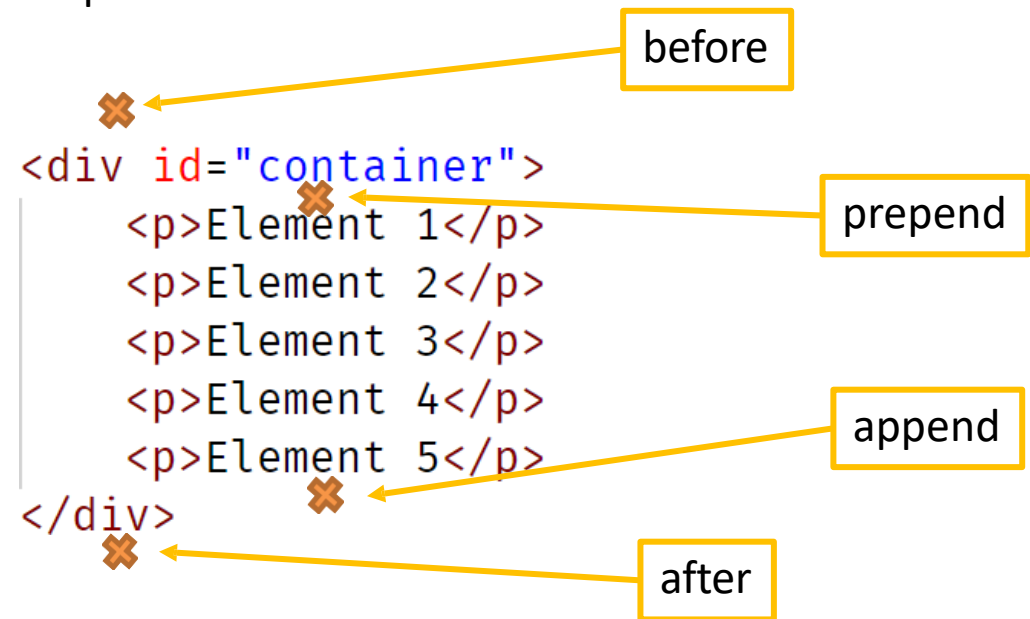
document.createTextNode(текст) – створення текстового вузла.



Щоб створені елементи стали видимими, їх потрібно додати в документ.

Методи вставлення:

```
node.append(новий_вузол)  
node.prepend(новий_вузол)  
node.before(новий_вузол)  
node.after(новий_вузол)  
node.replaceWith(новий_вузол)
```




JavaScript Базовий

Видалення вузлів


node.remove() – видаляє вузол, на якому викликаний метод із DOM-дерева.



Під час переміщення елемента в інше місце видаляти його зі старого не потрібно. Методи вставлення видаляють вузли зі старих місць.



```
<ul>
  <li>Item 01</li>
  <li>Item 02</li>
</ul>
```



```
<script>
  let lastListItem = document.querySelector("#marked");
  lastListItem.remove(); // видалення елемента з DOM
</script>
```

JavaScript Базовий

Клонування вузлів

node.cloneNode(true/false) – створення поверхневої та глибокої копії вузла, на якому було здійснено виклик.

```
<article>  
  <h1>Заголовок статті</h1>  
  <p>Lorem ipsum dolor sit...</p>  
</article>
```

`articleNode.cloneNode(true);`



```
<article>  
  <h1>Заголовок статті</h1>  
  <p>Lorem ipsum dolor sit...</p>  
</article>
```

`articleNode.cloneNode(false);`



```
<article>  
</article>
```

JavaScript Базовий

DOM-властивості

Вузол DOM – звичайний **об'єкт JavaScript**. Для DOM-вузлів можна задавати нові властивості та методи.

```
<body>
  <div></div>

  <script>
    let divElem = document.querySelector("div");

    divElem.myName = "My div";
    divElem.print = function() {
      console.log("myName = " + this.myName
        + " tagName = " + this.tagName);
    }

    divElem.print(); // myName = My div tagName = DIV
  </script>
</body>
```

JavaScript Базовий

HTML-атрибути

Під час парсингу HTML-розмітки на основі тегів створюються об'єкти, а значення **стандартних** атрибутів присвоюються відповідним властивостям створених об'єктів.

```
<div id="myDiv"  
  title="Hello world">  
</div>
```

```
<script>  
  let divElem = document.querySelector("div");  
  
  console.log(divElem.id);    // myDiv  
  console.log(divElem.title); // Hello world  
</script>
```

Список стандартних атрибутів елементів описується в специфікації. Наприклад, документація для елемента input <https://html.spec.whatwg.org/#htmlinputelement>.

Якщо атрибут не є стандартним, його не можна отримати за допомогою властивості.

JavaScript Базовий

Робота з атрибутами

- **element.hasAttribute(<ім'я_атрибута>);** – перевірка наявності атрибута.
- **element.getAttribute(<ім'я_атрибута>);** – отримання значення атрибута.
- **element.setAttribute(<ім'я_атрибута>, <значення_атрибута>);** – встановлення значення атрибута.
- **element.removeAttribute(<ім'я_атрибута>);** – видалення атрибута.

Ці методи працюють із вмістом HTML-розмітки.

Атрибути HTML мають дві особливості:

1. Їхній вміст – рядкове значення.
2. Імена атрибутів незалежні від реєстру.

JavaScript Базовий

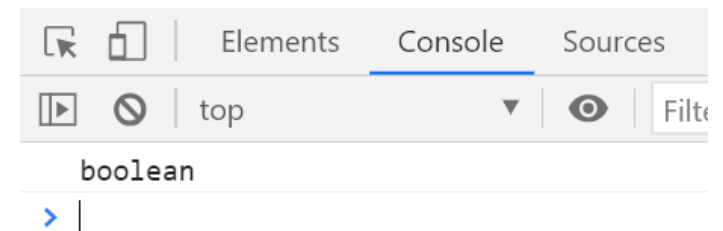
Атрибути

Між атрибутами та властивостями відбувається **автоматична синхронізація**. Якщо змінити атрибут, зміниться значення властивості. Якщо змінити властивість, зміниться значення атрибута.

Винятком є властивість та атрибут `value` для елемента `input`. Якщо змінити атрибут, значення зміниться, але не навпаки.

Властивості DOM **типізовані**, наприклад, властивість `checked` елемента `input` типу `Boolean`.

```
<input type="checkbox">
<script>
  let elem = document.querySelector("input");
  console.log(typeof (elem.checked));
</script>
```



JavaScript Базовий

Нестандартні атрибути

Атрибути, які починаються з префікса **data-**, є користувальницькими. Вони зарезервовані для використання розробником.

Атрибути data-* доступні через властивість елемента **dataset**.

Якщо атрибут складається з кількох властивостей, він потрапляє перейменованим з використанням **camelCasing**.

data-report-month в dataset буде властивістю **reportMonth**.

Зазвичай нестандартні атрибути використовуються для передання даних у JavaScript-код.

JavaScript Базовий

Тема

CSS-стилі

JavaScript Базовий


Стилі в JavaScript

Для оформлення HTML-елементів можна використовувати два підходи:

1. Використовувати **стилі inline**, які задані через атрибут style.
2. Визначати стилі за допомогою **CSS-правил** і **CSS-класів**.

elem.className
elem.classList

elem.style



```
<div class="message" style="color: red">  
  Hello world  
</div>
```

JavaScript Базовий

Властивість style

```
font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;  
border: 1px solid gray;  
border-radius: 8px;  
background-color: orange;  
font-weight: bold;  
margin: 8px;  
padding: 15px;
```



```
div.style.fontFamily = "'Segoe UI', Tahoma, Geneva, Verdana, sans-serif";  
div.style.border = "1px solid gray";  
div.style.borderRadius = "8px";  
div.style.backgroundColor = "orange";  
div.style.fontWeight = "bold";  
div.style.margin = "8px";  
div.style.padding = "15px";
```

Якщо CSS-властивість використовує префікс, то у JavaScript-кодi така властивість перейменовується з використанням **camelCasing**.

Якщо ім'я CSS-властивості починається з префікса, використовується **PascalCasing**.

CSS-властивості

color

font-family

-moz-radial-gradient

Властивість у

JavaScript

color

fontFamily

MozRadialGradient

JavaScript Базовий

Робота з CSS-класами



```
<div id="first-div" class="message">
```

```
firstDiv.className = "message";  
firstDiv.classList.add("message");
```

Методи об'єкта **classList**:

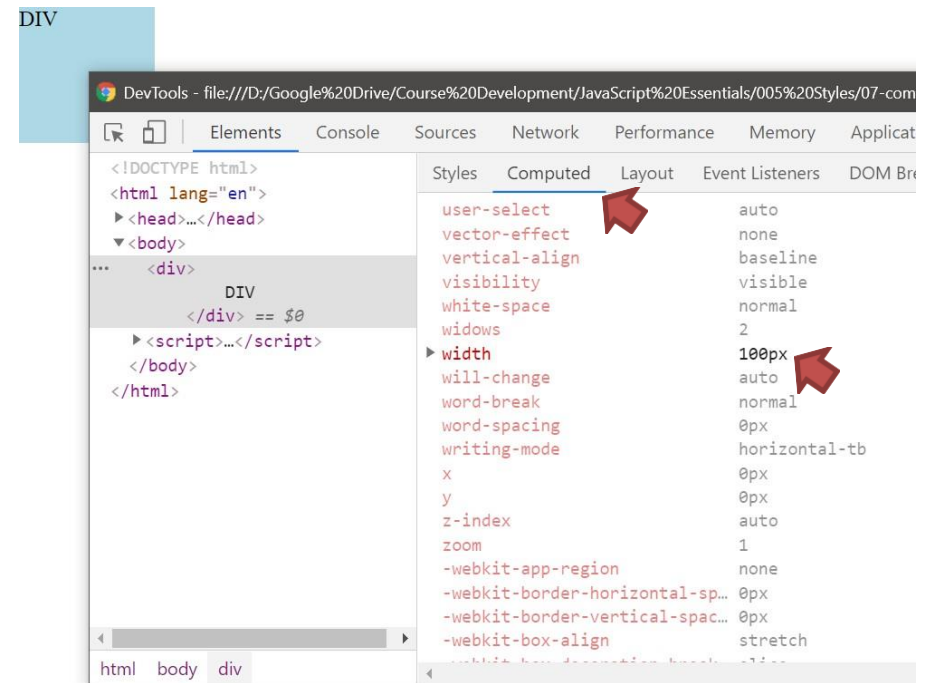
- **add** – додати клас до вже наявних;
- **remove** – видалити клас;
- **toggle** – додати, якщо класу немає, видалити, якщо клас є;
- **contains** – перевірка наявності вказаного класу у списку класів.

JavaScript Базовий

Обчислені стилі

Обчислені стилі – стилі, які отримані елементом після застосування всіх CSS-правил і дозволу всіх геометричних властивостей – висоти, ширини тощо.

Для отримання обчислених стилів використовується метод **getComputedStyle** (елемент, псевдо_елемент).



JavaScript Базовий

Висновки

- **document.createElement** – для створення елементів.
- Якщо створений елемент не буде додано до DOM, він не зобразиться.
- **node.append**, **node.prepend**, **node.before**, **node.after** – методи для вставлення створеного елемента.
- Якщо під час вставлення елемент вже є в іншій частині сторінки, його буде видалено зі старого місця.
- **node.remove** – видалення елемента з DOM.
- **node.cloneNode** – створення копії елемента.
- Елементи DOM-дерева – звичайні JS-об'єкти.
- Властивості елементів можуть ініціалізуватися на основі атрибутів HTML-розмітки.
- Між атрибутами та властивостями відбувається автоматична синхронізація.
- Користувацькі атрибути часто використовуються для передання даних у JavaScript-код.
- Для атрибутів користувача використовуйте префікс **data-**.
- Для використання атрибутів **data-*** використовуйте властивість **dataset**.

JavaScript Базовий

Висновки

- Намагайтеся використовувати **CSS-класи** для оформлення елементів.
- Властивість **style** – для роботи зі стилями inline.
- Властивість **classList** – для додавання або видалення класів CSS для елемента.
- **getComputedStyle** – для отримання стилів, які не задавалися в коді, а обчислювалися браузером.

JavaScript Базовий

Дякую за увагу! До нових зустрічей!



Охріменко Дмитро
МСТ



MCID: 9210561

Інформаційний відеосервіс для розробників програмного забезпечення

