Obiekty DOM

Czym jest model DOM?

• DOM (ang. Document Object Model)

• Model umożliwiający obiektowy opis strony internetowej oraz podstawowe API pozwalające na wykonywanie operacji na strukturze dokumentu

• W modelu tym strona przedstawiana jest w postaci drzewa, gdzie:

• Węzłami wewnętrznymi, są zawsze tagi HTML

• Liśćmi drzewa są tagi HTML, lub elementy tekstowe

Każda strona HTML składa się z elementów.

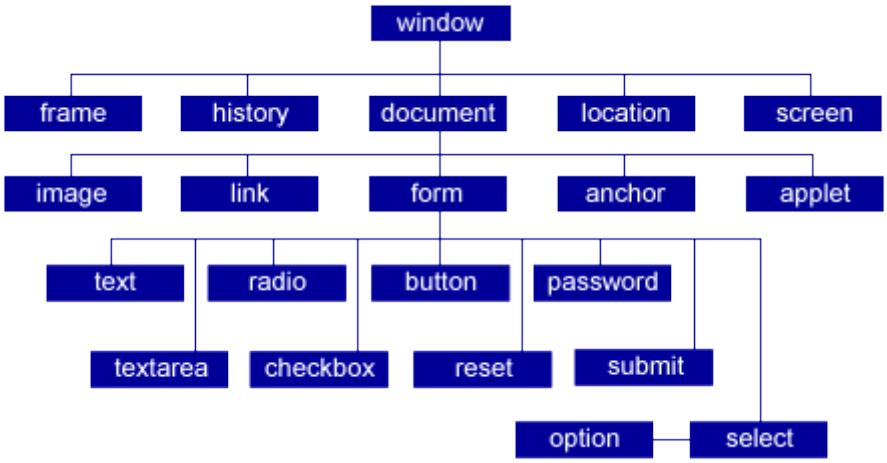
Na samej górze jest okno przeglądarki - window, które zawiera w sobie wszystkie obiekty, funkcje i właściwości. W tym oknie znajduje się obiekt **document** (czyli nasza strona). W tym obiekcuie znajdują się inne obiekty. W naszych skryptach będziemy operować na tych obiektach, nakazując im wykonywanie różnych rzeczy.

Do odzwierciedlenia ułożenia elementów JS korzysta z DOM. DOM czyli Document Object Model to stworzony przez W3C model ułożenia elementów na stronie - czyli hierarchia. DOM nie tylko opisuje ułożenie elementów, ale udostępnia mnóstwo metod, które ułatwiają poruszanie się po tych elementach i manipulowanie nimi.

Drzewo obiektów (DOM)

Obiekty są elementami języka zawierającymi właściwości, metody oraz inne obiekty. Właściwości są zmiennymi określającymi obiekt (np. nazwa). Metody natomiast są funkcjami działającymi tylko w obrębie obiektu. Dla porównania dla obiektu "człowiek" właściwościami byłby wzrost czy kolor włosów natomiast metodami chodzenie czy myślenie. Podobnie obiekty w realnym świecie składają się z innych obiektów.

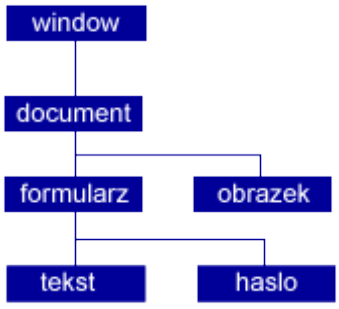
Obiekty w Java Script opisują wygląd i działanie elementów strony oraz przeglądarki. Uporządkowane są one hierarchicznie w formie drzewa. Najważniejszym obiektem "rodzicem" jest window.



W obiekcie dokument istnieje ponadto obiekt oznaczający warstwę. Jest on różnie nazywany w zależności od przeglądarki.

Obiekty można podzielić na wbudowane, których nazwa występuje w adresie jak np. window.history oraz na obiekty, których nazwa w drzewie oznacza typ obiektu. Na przykład można utworzyć obiekt typu image i nazwać go przykładowo obrazek. Będzie on miał adres window.document.obrazek (lecz nie window.document.image). Obiekty wbudowane to: window, history, document, location, screen natomiast wszystkie pozostałe to typy obiektów, które należy nazywać.

Adresowanie obiektów jest proste. Na przykład adres obiektu "obrazek" w poniższym przykładzie to window.document..obrazek



Podobnie adresuje się właściwości i metody obiektów.
Adresowanie właściwości: window.document.obrazek.src
Adresowanie metody: window.document.formularz.submit()
Przypisanie wartości do właściwości: window.document.obrazek.src="rysunek.jpg";

Wszystkie nazwane w html elementy stają się obiektami JavaSript o podanej nazwie. Dla przykładu obrazek:

< img src="plik.jpg" name="rysunek">

Będzie posiadał adres: window.document.rysunek

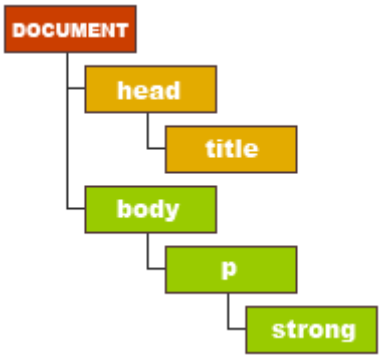
Ponieważ wszystkie obiekty znajdują się w obiekcie window czasem pomija się go w adresie, a więc można zapisać window.document.obrazek lub document.obrazek

---

Przypuśćmy, że stworzyliśmy stronę:

```
<html>
<head>
    <title>To jest tytuł strony</title>
</head>
<body>
    <p>Ten napis zawiera <strong>pogrubiony tekst</strong></p>
</body>
</html>
```

Nasz dokument możemy rozrysować jako hierarchiczne drzewo. Na samej górze jest **document** HTML, a tuż pod nim znajdują się jego "dzieci" (inna nazwa to korzenie - nody) - czyli elementy znajdujące się w document.



Odwoływanie się do obiektów

Aby odwołać się do elementów na stronie, skorzystamy z metod

getElementById,

getElementsByTagName

querySelector".

Tą pierwszą użyjemy wtedy, gdy nasz element ma atrybut **id**. Ta druga służy do pobrania kolekcji zawierającej elementy danego typu.

Przykładowo jeżeli na naszej stronie będziemy mieli akapit:

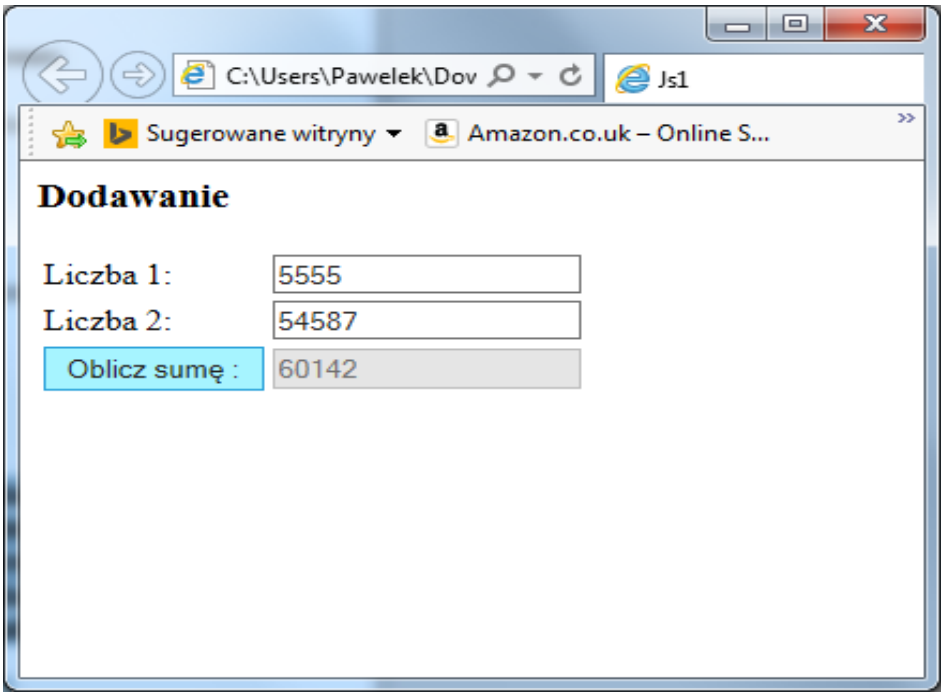
```
<p id="paragraf">Ten napis zawiera <strong id="bold">pogrubiony tekst</strong></p>
<p>Lorem ipsum</p>
```

to odwołanie się do naszych elementów z poziomu skryptu będzie miało postać:

```
document.getElementById('paragraf')       //wskazuje na nasz akapit
document.getElementById('bold')           //wskazuje na nasz znacznik strong
document.getElementsByTagName('p')        //kolekcja akapitów
document.getElemensByTagName('p')[0]             //wskazuje na pierwszy akapit

document.getElemensByTagName('p')[0].getElementsByTagName('strong')[0]
 //pobieramy pierwszy akapit, a w nim pobieramy pierwszy strong

document.getElementById('paragraf').getElementsByTagName('strong')
//kolekcja znaczników strong znajdujących się w akapicie paragraf



document.getElementById('paragraf')[1].firstChild  //pierwsze dziecko 2 akapitu
document.getElementById('cos').childNodes[1]    //drugie dziecko elementu cos
document.getElementById('cos').parentNode
 //element rodzic w którym leży element c
```

Przykład nr 1

```
<html><head>  <title>Js1</title>
<script type="text/javascript">
//funkcja
function oblicz()
{ var l1=document.getElementById('l1'); //pobierz element o id='l1' - liczba1
 l1=l1.value; //pobierz wartośc elementu l1 -tekst.
 l1=parseInt(l1); //konwertuj tekst na liczbę całkowitą
 var l2=document.getElementById('l2'); //pobierz element o id='l2' - liczba 2
 l2=parseInt(l2.value); //konwertuj wartość na liczbę całkowitą
 var s=document.getElementById('suma'); //pobierz element o id='suma'
 s.value=l1+l2; //ustaw wartość dla elementu s
}
</script>
</head>

<body>
<div> <h3>Dodawanie</h3>
<table><tbody>
 <tr><td>Liczba 1:</td><td><input id="l1" /></td> </tr>
<tr><td>Liczba 2:</td><td><input id="l2" /></td> </tr>
 <tr><td><button onclick="oblicz()">Oblicz sumę : </button></td>
 <td><input id="suma" disabled="disabled" value=""/></td>
</tr></tbody></table>
</div></body></html>
```



Rysunek 1 Skrypt nr 1 po uruchomieniu w IE

Zadania

Zad. 1.

Na podstawie ćwiczenia 1 opracować skrypt JS do obliczenia różnicy, iloczynu, ilorazu.

Zad. 2

Na podstawie ćwiczenia 1 opracować skrypt JS, który sprawdzi, czy liczba 1 jest równa liczbie 2 wyświetlając odpowiedni komunikat (komunikat w trzecim okienku tekstowym).

Zad. 3

Na podstawie ćwiczenia 1 opracować skrypt JS, który wczyta w jednym polu tekstowym login, w drugim hasło, a następnie wyświetli komunikat czy login i hasło są zgodne z zaplanowanymi w skrypcie.

Zad. 4

Na podstawie ćwiczenia 1 zaprojektować formularz i opracować skrypt JS do obliczenia sumy kolejnych liczby z przedziału <a, b>.

Zad. 5

Na podstawie ćwiczenia 1 zaprojektować formularz i opracować skrypt JS do wyznaczenia kolejnych liczby ciągu Fibonacciego.

Zad. 6

Na podstawie ćwiczenia 1 zaprojektować formularz i opracować skrypt JS do obliczenia wartości silni z liczby n (n jest liczbą naturalna większą od zera).

Zad. 7

Na podstawie ćwiczenia 1 zaprojektować formularz i opracować skrypt JS do obliczenia sumy kolejnych liczby z przedziału <a, b>.