Instrukcja JavaScript

Wprowadzenie

Obiektowy Model Dokumentu (DOM - Document Object Model) jest strukturą drzewiastą przechowującą obiekty reprezentujące wszystkie elementy strony. Korzeniem części drzewa odpowiedzialnej za sam dokument HTML jest obiekt globalny document. JavaScript oferuje metody obiektu document pozwalające na dotarcie do konkretnych obiektów w drzewie DOM:

- getElementById - obiekt o określonym identyfikatorze,
- getElementsByName - kolekcja obiektów o określonej wartości parametru name,
- getElementsByClassName - kolekcja obiektów danej klasy,
- getElementsByTagName - kolekcja obiektów danego typu.

Każdy obiekt w DOM posiada składową style, przechowującą jego wszystkie parametry CSS. Przykładowo, kolor czcionki w obiekcie o identyfikatorze mojID można ustawić instrukcją:

document.getElementById('mojID').style.color="blue";

Rozważmy następujący prosty przykład:

```
1. <html>
2. <head>
3.     <script type="text/javascript">
4.         function ustawCzcionke()
5.         {
6.             rozmiar=document.getElementById("poleTekstowe").value;
7.             element=document.getElementById("akapit");
8.             element.style.fontSize=rozmiar+"px";
9.         }
10.    </script>
11. </head>
12. <body>
13.     <input type="text" id="poleTekstowe"/>
14.     <p id="akapit">A łyżka na to:"To niemożliwe..."</p>
15.     <input type="button" value="Kliknij mnie"
onclick="ustawCzcionke()"/>
16.</body>
17.</html>
```

Po kliknięciu na przycisk "Kliknij mnie" (linia 15 kodu) wywoływana jest funkcja **ustawCzcionke()**. Funkcja ta pobiera najpierw wartość wpisaną przez użytkownika w polu tekstowym o identyfikatorze **poleTekstowe** (linia 6). Do pobrania tej wartości wykorzystana jest metoda **getElementById**. Następnie (linia 7) lokalizowany jest element o identyfikatorze **akapit**, a referencja do tego dokumentu jest zapisana w zmiennej **element**. W końcu (linia 8) modyfikowany jest styl zlokalizowanego elementu - ustawiana jest wartość rozmiaru czcionki. Pola własności CSS występujące w skryptach JavaScript po polu **style** są niemal identyczne jak własności wpisywane w plikach CSS. Własności CSS zapisywane z myślnikiem w plikach CSS są w skryptach JS zastępowane ciągiem bez myślnika, z podmianą pierwszej zawsze w CSS małej litery po myślniku na dużą (np. background-color zastępujemy

backgroundColor).
Przy ustawianiu wartości dla własności CSS trzeba koniecznie zadbać o to, aby wartość wpisana po prawej stronie była poprawną wartością CSS. Dlatego w linii8 po prawej stronie instrukcji znalazło się wyrażenie **rozmiar+"px"**. Jeżeli użytkownik wpisał w polu tekstowym liczbę 14, to wartością tego wyrażenia jest 14px, czyli prawidłowa wartość dla rozmiaru czcionki.

ZADANIE 1

Przetestuj działanie kodu z przykładu.

ZADANIE 2

Na podstawie zadania 1, zapisz stronę HTML i skrypt JS w osobnych plikach. Odpowiednio je zlinkuj.

ZADANIE 3

Proszę stworzyć aplikację, która czyta liczbę z przedziału od 0 do 100, wpisaną przez użytkownika w polu tekstowym, a następnie ustawia kolor w skali szarości odpowiadający wprowadzonej liczbie dla sekcji [*div1*], w kształcie prostokątnego obszaru, po kliknięciu na przycisk **Ustaw kolor**.
Np. Dla liczby 4 kolor tła tej sekcji ma uzyskać wartość rgb(4%,4%,4%).

Wskazówki:

- aby pobrać wartość z pola tekstowego można użyć na przykład metody getElementById obiektu document;
- aby wykonać skrypt JavaScript po kliknięciu na przycisk **Ustaw kolor** należy ten skrypt napisać;
- ... a następnie odpowiednio określić wartość atrybutu onclick dla przycisku;
- style w JavaScript ustawia się korzystając z pola style elementu wybranego np. metodą getElementById:

```
document.getElementById("mySection").style.fontSize="15px";
```

- operator dodawania "+" jest w JS przeładowany – można go użyć to tworzenia złożonych napisów, np napisów definiujących wartość koloru tła.

ZADANIE 4

Proszę zmodyfikować aplikację z poprzedniego zadania tak, aby ustawiała losowy kolor tła w skali szarości dla nowej prostokątnej sekcji [*div2*] (poniżej obecnej) po kliknięciu na nowy przycisk **Ustaw losowy kolor**.

ZADANIE 5

Proszę rozbudować aplikację z poprzedniego zadania tak, aby użytkownik mógł dopasować (klikając na przyciski **Jaśniej** i **Ciemniej**) kolor w sekcji #div1, tak aby kolor ten był jak najbliższy kolorowi ustawionemu losowo w sekcji #div2.

Wskazówki:

- Przyciski **Jaśniej** i **Ciemniej** mogą (i mają) być obsługiwane przez tą samą funkcję, której przekazujemy parametr o wartości zależnej od tego, na którym przycisku kliknięto;
- Zmienna globalna jest deklarowana poza ciałem jakiejkolwiek funkcji, we wnętrzu znacznika skrypt, powinna być inicjowana i jest widoczna przez wszystkie funkcje.

ZADANIE 6*

Proszę rozbudować aplikację z poprzedniego zadania w taki sposób, aby po kliknięciu przycisku **OK** w tabeli pojawiało się podsumowanie zawierające numer próby, losowy kolor, ustawiony kolor i błąd wyrażony jako stosunek różnicy koloru wybranego i losowego do koloru losowego (dokładność dwóch miejsc po przecinku). Po kliknięciu przycisku **Czyść** tabela ma być usuwana - oprócz wiersza nagłówkowego.

Wskazówki:

- Wykorzystać metody appendChild w celu dodania wierszy do tabeli i dodania komórek do wiersza;
- Wykorzystać pole innerHTML do ustawienia zawartości komórek;
- Wykorzystać metodę removeChild do usunięcia wierszy tabeli.