

Lekcja 1

Temat: Wprowadzenie do języka JavaScript

Czym jest JavaScript?

Język ten umożliwia tworzenie zagnieżdżonych bezpośrednio w kodzie HTML krótkich programów, które mogą wykonywać najrozmaitsze zadania, np. potrafią rozpoznać zdarzenia powodowane przez użytkownika i odpowiednio na nie zareagować. Zdarzenia te to np. kliknięcie myszą, zmiana rozmiaru okna czy przesunięcie elementu witryny. JavaScript to też podstawa tak popularnej ostatnimi czasy techniki Ajax (dawniej: AJAX), dzięki której strona WWW może zachowywać się tak jak zwyczajna aplikacja uruchamiana z poziomu systemu operacyjnego.

Obecnie trudno już spotkać witrynę, która nie korzystałaby z JavaScriptu. Każdy twórca nowoczesnych stron internetowych musi więc, oprócz HTML-a i stylów CSS, znać również i ten język, a przynajmniej jego podstawy. Zatem jeśli chcesz wiedzieć, jak obsługiwać na stronie elementy interfejsu użytkownika, budować dynamiczne menu i przyciski, dynamicznie generować treści i tworzyć rozmaitego rodzaju efekty, musisz poznać JavaScript.

Trochę historii

- JavaScript stworzony został przez firmę Netscape i pojawił się początkowo w 1995 roku.
- Organizacja ECMA zajmuje się standaryzacją technologii
- Powstał standard ECMAScript - JavaScript to implementacja tego standardu

Jakie jest znaczenie nazwy

- ECMAScript 2015 - standard z roku 2015
- ES6 - jest to 6 z kolei wariant standardu

ECMAScript 2015 = ES6

Obecne, przeszłe i przyszłe wersje

- ES5 - dotychczasowy standard obecny od 2009 roku
- ES6/ECMAScript 2015 - standard najnowszy
- ECMAScript 2016 - ewentualny przyszły standard

Dlaczego nowy standard

- JavaScript początkowo był prostym językiem skryptowym
- Obecnie często używamy JavaScript do tworzenia złożonych dynamicznych aplikacji webowych
- Node.js oraz inne elementy ekosystemu
- Dotychczasowe standardy nie odpowiadały dzisiejszym potrzebom

Najpopularniejsze języki programowania 2020:

<https://bulldogjob.pl/news/1065-najpopularniejsze-jezyki-programowania-2020>

Praca:

https://justjoin.it/?gclid=EAlaIQobChMIgpeUwcrt8gIVLwWiAx1IVQTBAAAYASAAEgJbS_D_BwE

Nauka:

<https://www.programiz.com/javascript/get-started>

<https://www.programiz.com/javascript>

<https://javascript.info/>

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference>

<https://kursjs.pl/>

Narzędzia deweloperskie

Przycisk F12 lub wykorzystanie narzędzia **Zbadaj** na stronie pozwalają na wejście do konsoli w przeglądarce internetowej. Możemy w niej wykonać proste działania, stworzyć różne obiekty JS i przede wszystkim zobaczyć błędy w skryptach zapisanych na stronie w plikach HTML.

The screenshot shows a web browser window. On the left, a page titled "Pierwszy skrypt" (First script) is displayed. It contains a list of five numbered items in red text:

- 1 Umieszczanie skryptów w oddzielnym pliku
- 2 Która metoda jest lepsza?
- 3 Gdzie wstawiać nasze skrypty
- 4 Atrybuty async i defer
- 5 Type module

Below the list, there is a text block: "Do umieszczenia skryptów na stronie służy znacznik `<script>`:".

On the right side of the browser window, the developer console is open. It shows the following log entries:

- `> 5+6` followed by `< 11`
- `> console.log("testujemy")` followed by `testujemy` and `< undefined`
- `> [3,4,5]` followed by `< (3) [3, 4, 5]`

The last entry, `(3) [3, 4, 5]`, is expanded, showing the array structure:

- `0: 3`
- `1: 4`
- `2: 5`
- `length: 3`
- `[[Prototype]]: Array(0)`
- Methods: `at`, `concat`, `constructor`, `copyWithin`, `entries`, `every`, `fill`, `filter`

Skrypty w kodzie strony

W celu umieszczenia skryptu w kodzie strony stosuje się znacznik

<script>,

który może zawierać następujące atrybuty:

- type — określa typ skryptu (wg standardu MIME).
- src — wskazuje plik zewnętrzny z kodem skryptu.
- charset — określa standard kodowania znaków w skrypcie zewnętrznym.
- defer — wskazuje, że skrypt (zewnętrzny) powinien być wykonany dopiero po przetworzeniu struktury dokumentu.
- async — wskazuje, że skrypt może być przetwarzany asynchronicznie (równoległe z kodem strony; nie zawiera odwołań do struktury dokumentu). Parametr dostępny w HTML5.

Parametr type jest obligatoryjny w HTML 4.01, nie musi natomiast występować w HTML5. Dla JavaScriptu wartością tego atrybutu powinien być ciąg text/javascript.

Jedna strona może zawierać wiele skryptów, które można umieszczać zarówno **w sekcji head, jak i w sekcji body**. Przyjmuje się, że o ile to możliwe, należy korzystać z tej pierwszej możliwości i nie przeplatać kodu HTML z kodem JavaScript.

<https://kursjs.pl/kurs/super-podstawy/pierwszy-skrypt.php>

Użycie znacznika <script>

Przykład 1.

```
1  <!DOCTYPE HTML>
2  <html lang="pl">
3  <head>
4      <meta charset="utf-8"/>
5      <script type="text/javascript">
6          //Tutaj można umieścić kod pierwszego skryptu. - komentarz JS
7      </script>
8      <title>JavaScript -l1p1</title>
9  </head>
10 <body>
11     <script type="text/javascript">
12         //Tutaj można umieścić kod drugiego skryptu.
13     </script>
14 </body>
15 </html>
```

Skrypt może być również umieszczony w osobnym pliku; mówi się wtedy o **skrypcie zewnętrznym**. W takim przypadku do znacznika `<script>` trzeba dodać atrybut `src` wskazujący lokalizację pliku. Taki plik może mieć dowolną nazwę, zwykle przyjmuje się jednak, że ma ona rozszerzenie `js`. Można też dodać atrybut `charset` określający sposób kodowania znaków w skrypcie (w szczególności gdyby miał być użyty niestandardowy typ kodowania). Przy stosowaniu atrybutu `src` treść znacznika musi być pominięta, należy jednak stosować wersję pełną ze znacznikiem zamykającym `</script>`.

Skrypty zewnętrzne

Przykład 2. Umieść w kodzie HTML odwołanie do skryptu znajdującego się w osobnym pliku niż kod HTML.

```
1  <!DOCTYPE HTML>
2  <html lang="pl">
3  <head>
4      <meta charset="utf-8"/>
5      <script type="text/javascript"
6          src="scripts/myscript1.js"
7          charset="utf-8"></script>
8      <title>Skrypt zewnętrzny -11p2</title>
9  </head>
10 <body>
11     <!-- tutaj treść strony HTML - komentarz HTML-->
12 </body>
13 </html>
```

```
myscript1.js
1  console.log("Witaj");
```

Atrybut `src` o wartości `scripts/myscript1.js` wskazuje lokalizację skryptu, czyli podkatalog i znajdujący się w nim plik. Użyty został również atrybut `charset` wskazujący, że w skrypcie użyto kodowania znaków UTF-8.

UWAGA: Z reguły w HTML umieszczane są tylko najprostsze skrypty. Bardziej złożone znajdują się w osobnych plikach.

Zaletą oddzielnego pliku jest to, że przeglądarka pobierze go i zapisze w swojej pamięci podręcznej. Inne strony, które odwołują się do tego samego skryptu, wezmą go z pamięci podręcznej zamiast go pobierać, więc plik jest w rzeczywistości pobierany tylko raz. To zmniejsza ruch i przyspiesza strony.

Wyświetlanie informacji

Możliwości wyświetlania JavaScript

JavaScript może "wyświetlać" dane na różne sposoby:

- Zapis do elementu HTML za pomocą `innerHTML`, `innerText`, `textContent`.
- Pisanie do wyjścia HTML za pomocą `document.write()` . – Metoda przestarzała, powinna być używana tylko do testowania na początku pracy z JS.
- Pisanie w polu alertu za pomocą `window.alert()` .
- Pisanie do konsoli przeglądarki za pomocą `console.log()` .

Pisanie w polu alertu za pomocą `window.alert()` .

Skrypty działające w przeglądarce muszą mieć możliwość interakcji z elementami witryny. **Jedną z możliwości jest wyświetlenie okna dialogowego.** Okno takie służy zwykle do poinformowania użytkownika o wystąpieniu jakiegoś zdarzenia; najczęściej chodzi o sytuację, w której wystąpił błąd. Na taki charakter prezentowanej metody wskazuje już sama nazwa — `alert`. Schematyczne wywołanie ma postać:

```
alert("ciąg znaków do wyświetlenia");
```

Tekst do wyświetlenia musi być ujęty w znaki cudzysłowu (cudzysłowy proste górne, jak powyżej) lub apostrofu (cudzysłowy proste górne: `'tekst'`).

Możesz pominąć słowo kluczowe **window**.

W JavaScript obiekt **window** jest obiektem zasięgu globalnego, co oznacza, że zmienne, właściwości i metody domyślnie należą do obiektu `window`. Oznacza to również, że określenie słowa kluczowego `window` jest opcjonalne.

Wyświetlenie okna dialogowego

Przykład 3. Napisz skrypt, który wyświetli w przeglądarce okno dialogowe z dowolną informacją.

Przygotuj stronę html wykorzystując poniższy kod. Zapisz stronę pod nazwą `l1p3.html`. Uruchom ją w dowolnej przeglądarce. Okno dialogowe uzyskane w ten sposób będzie miało różną postać, w zależności od użytej przeglądarki.

```
1 <!DOCTYPE HTML>
2 <html lang="pl">
3 <head>
4   <meta charset="utf-8"/>
5   <title>Mój pierwszy skrypt JS-l1p3</title>
6   <script>
7     //komentarz: w starszej wersji js możesz spotkać się z zapisem:
8     // <script type = "text / javascript">
9     //obecnie atrybut typu nie jest wymagany
10
11     alert("witaj");
12   </script>
13 </head>
14 <body>
15   <script>
16     // kod możesz też umieścić w body
17     alert("witaj ponownie");
18   </script>
19 </body>
20 </html>
```

Instrukcja `document.write()`

UWAGA: Metoda przestarzała, powinna być używana tylko do testowania na początku pracy z JS

Jeżeli informacja ma się pojawić bezpośrednio na stronie (a nie w osobnym oknie), można użyć instrukcji `document.write`. Tekst, który ma zostać wyświetlony, należy umieścić w nawiasie okrągłym i ująć w cudzysłów (lub znaki apostrofu):

```
document.write("tekst");
```

Ten tekst będzie wprowadzony bezpośrednio do struktury dokumentu, skrypt powinien więc znajdować się w sekcji `body` w miejscu, gdzie ma się pojawić dany napis.

Przykład 4. Umieść na stronie akapit tekstowy. Napisz skrypt, który umieści w tym akapicie dowolny tekst. Przygotuj stronę html wykorzystując poniższy kod. Zapisz stronę pod nazwą l1p4.html. Uruchom ją w dowolnej przeglądarce.

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8"/>
5     <title>Generowanie treści w skrypcie -l1p4</title>
6   </head>
7   <body>
8     <p>
9       <script type="text/javascript">
10        document.write("To jest treść akapitu tekstowego.");
11      </script>
12    </p>
13  </body>
14 </html>
```

Tak przygotowany kod spowoduje, że na ekranie pojawi się pożądaný napis. O tym, że tekst przekazany instrukcji `document.write` faktycznie został wprowadzony do treści znacznika `<p>`, można się przekonać, podglądając źródło strony. Sprawdź.

Używanie znaczników formatujących dane

Treść wyświetlana za pomocą metody `document.write` przedstawionej w poprzednim przykładzie jest traktowana tak jak kod HTML i tak samo interpretowana przez przeglądarkę. Oznacza to, że można stosować tu znaczniki HTML, w tym takie, które wpływają na wygląd tekstu. Nic nie stoi więc na przeszkodzie, aby utworzyć np. akapit tekstowy z pogrubioną czcionką.

Przykład 5. Umieść na stronie akapit tekstowy. Napisz skrypt, dzięki któremu w warstwie pojawi się tekst częściowo pogrubiony, a częściowo pochylony. Przygotuj stronę html wykorzystując poniższy kod. Zapisz stronę pod nazwą l1p5.html. Uruchom ją w dowolnej przeglądarce.

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8"/>
5     <title>Formatowanie tekstu -l1p5</title>
6   </head>
7   <body>
8     <p>
9       <script type="text/javascript">
10        document.write("Ten tekst <b>częściowo pogrubiony</b>, ");
11        document.write("a <i>częściowo pochylony</i>.");
12      </script>
13    </p>
14  </body>
15 </html>
```


UWAGA: Użycie `document.write()` po załadowaniu dokumentu HTML spowoduje **usunięcie całego istniejącego HTML**

Przykład 6.

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<button type="button" onclick="document.write(5 + 6)">Try it</button>

</body>
</html>
```

Zobacz działanie przykładu na stronie: https://www.w3schools.com/js/js_output.asp

Zapis do elementu HTML za pomocą `innerHTML`, `innerText`, `textContent`

Aby uzyskać dostęp do elementu HTML, JavaScript możesz użyć np. metody `document.getElementById(id)` lub innych (poznasz je na kolejnych lekcjach).

Atrybut `id` definiuje element HTML. Właściwość `innerHTML` określa zawartość HTML.

Przykład 7.

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8"/>
    <title>Generowanie treści w skrypcie -llp7</title>
  </head>
  <body>
    <h3>
      Generowanie treści na stronie
    </h3>

    <p id="demo"></p>
    <script>
      document.getElementById("demo").innerHTML = 5 + 6;
    </script>
  </body>
</html>
```

Pisanie do konsoli przeglądarki za pomocą `console.log()`.

Przykład 8. Użyj w kodzie skryptu komentarza blokowego. Przygotuj stronę html wykorzystując poniższy kod. Zapisz stronę pod nazwą `l1p8.html`.

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8"/>
    <title>Generowanie treści w skrypcie -l1p8</title>
  </head>
  <body>
    <h3>
      Generowanie treści
    </h3>
    <script type="text/javascript">
      console.log("Cześć");
    </script>
  </body>
</html>
```

Komentarze

W treści skryptu można umieszczać jedynie taką treść, która będzie interpretowana przez przeglądarkę jako instrukcje JavaScriptu. W przypadku bardziej skomplikowanego kodu warto jednak dodać komentarze objaśniające działanie poszczególnych jego fragmentów. Już z nich korzystałeś/łaś. Do dyspozycji są dwa rodzaje komentarzy: wierszowy i blokowy.

Komentarz wierszowy (liniowy) zaczyna się od znaków `//` i obowiązuje do końca danej linii skryptu. Wszystko, co występuje po tych dwóch znakach, aż do końca bieżącej linii, jest ignorowane przez przeglądarkę przetwarzającą skrypt.

Komentarz blokowy rozpoczyna się od znaków `/*` i kończy znakami `*/`. Wszystko, co znajduje się pomiędzy, jest pomijane przy przetwarzaniu kodu przez przeglądarkę. Komentarz blokowy można umieścić praktycznie w dowolnym miejscu skryptu, może on się nawet znaleźć w środku instrukcji (pod warunkiem że nie zostanie przedzielone żadne słowo).

Wykorzystanie komentarza blokowego

Przykład 9. Użyj w kodzie skryptu komentarza blokowego. Przygotuj stronę html wykorzystując poniższy kod. Zapisz stronę pod nazwą l1p9.html.

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8"/>
    <title>Komentarz blokowy - l1p9</title>
  </head>
  <body>
    <script type="text/javascript">
      /*
      Przykładowy skrypt wyświetlający treść strony.
      Powstanie warstwa wraz z akapitem tekstowym.
      */
      document.write("<div>");
      document.write("<p>To jest treść akapitu tekstowego.</p>");
      document.write("</div>");
    </script>
  </body>
</html>
```