Ćwiczenie 8

JavaScript – skrypty klient-side

Czas trwania: 90 min.

Materiały opracowane na podstawie ćwiczeń dr. Tomasza Bajorka

Zadania do wykonania:

1. **W oparciu o instrukcję przećwiczyć kilka przykładów zastosowania języka JavaScript.**
   (Czas trwania: ok. 45 min.)
   W instrukcji zamieszczono przykłady kilku skryptów napisanych w technologii JavaScript. Należy je przećwiczyć sprawdzając ich działanie jako elementów dodanych do strony WWW.
2. **Na podstawie przeprowadzonych ćwiczeń wykonać zadania przedstawione na końcu instrukcji**
   (Czas trwania: ok. 45 min.)

Objaśnienia

JavaScript – jest językiem skryptowym typu klient-side, czyli uruchamianym przez przeglądarkę internetową po stronie klienta. Język doskonale współdziała w zakresie współpracy z kodem PHP oraz HTML. Za jego pomocą można wykonywać czynności pomocnicze, komplementarne dla budowanego serwisu. Jest składnikiem technologii dynamicznych.

Ćwiczenia praktyczne

1. Komentowanie treści opisowych w skryptach
   Komentarz o kilku wierszach,

```
<SCRIPT LANGUAGE="JavaScript">
/*
treść komentarza
 */
//  Jednowierszowy komentarz
</SCRIPT>
```

2. Wstawienie skryptu do dokumentu HTML
   Skrypty *JavaScript* są zagnieżdżane w dokumentach HTML między znacznikami *<SCRIPT> i </SCRIPT>.*

```
<SCRIPT LANGUAGE="JavaScript">
 treść skryptu
```

</SCRIPT>

Przykłady zawarte w skrypcie należy wykonywać sukcesywnie dopisując poszczególne linijki i sprawdzając ich działanie. NIE należy kopiować całości skryptu od razu na stronę ze względu na to, że wtedy w trakcie poznawania technologii nie wiadomo dlaczego powstały poszczególne efekty.

```
<HTML>
<meta http-equiv="Content-Type" content="text/html;
charset=Windows-1250"/>
<HEAD>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="JavaScript">
// obiekt document i jego metoda write pozwalają na
wypisanie tekstu w dokumencie html
document.write ("To jest zwykły tekst<BR>");
document.write("wiersz <BR> kolejny wiersz");
document.write ("<BR>"); // wysyłamy znacznik html
document.write ("<B>NAVIGATOR</B>:");
document.write (navigator.appName);
document.write ("<BR>");
alert("Zmienna x wynosi:"+x);
x=5; //przypisujemy wartość zmiennej
alert("Wartość zmiennej x wynosi:"+x);
document.write("Wartość zmiennej <I>x</I> : "+x); // ... i
wyświetlamy jej wartość
document.write ("<BR>To jest liczba PI:"+Math.PI);
</SCRIPT>
<P><B> <FONT color=navy size=7>A to jest zwykły
HTML</FONT></B></P>
</BODY>
</HTML>
```

efekt : wyświetlenie treści oraz okienek, które prezentują poszczególne informacje zawarte w skrypcie.

3. Instrukcje podstawowe, dodawanie instrukcji, operatory
   Instrukcje języka oddzielamy średnikami (jeśli zapisujemy w tym samym wierszu). Blok instrukcji otaczamy klamrami { } – zazwyczaj w ciele funkcji, instrukcjach warunkowych i iteracyjnych (np. if, for, while) – to samo co begin end w Pascalu. Duże i małe litery są istotne.

   Operatory
   Przypisania: =

+=	x+=5	odpowiada	x=x+5,
–=	x–=5	odpowiada	x=x–5,
*=	x*=5	odpowiada	x=x*5,

/= x/=5 odpowiada x=x/5,
%= x%=5 odpowiada x=x%5 (reszta z dzielenia)

```
x = 7; x += 4; x %= 10;
```

Porównania: == !=(nierówne) <= < > >=

```
x>10        2*y== -3        3!=="3"     3<=10
```

Arytmetyczne: +, -. /, *, ++, --, % (reszta z dzielenia)

```
x++;
--x;
b=x%4;
```

Logiczne: koniunkcja &&, alternatywa ||, negacja !
true && false daje false
!false daje true

Zadanie:
Sprawdzić w skrypcie efekt działania instrukcji:

```
x=2;y=4;
document.write(  (x==y)  +"<BR>");
document.write(  (x>y)   +"<BR>");
document.write(  (x<=y)  +"<BR>");
document.write(  !(y!=4)     +"<BR>");
document.write( (x*=5)   +"<BR>");
```

efekt: wynikiem działania będą linie prezentujące na stronie poszczególne działania instrukcji JavaScriptu.

4. Istrukcja if...else
Instrukcja *if* spełnia warunek jeśli jest on prawdziwy; używamy opcjonalnie klauzuli *else* aby wykonać inną instrukcję jeśli warunek jest fałszywy. Instrukcja if ma postać:

```
if (warunek)
{
   instrukcja1
}
[else {
   instrukcja2
} ]
```

Przykład zastosowania instrukcji if

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
function cB(x){
if (x==0) x++;
```

```
if (x==5) x--;
return x;}
</SCRIPT>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="JavaScript">
document.write(cB(0));
document.write(cB(5));
</SCRIPT>
</BODY>
</HTML>
```

efekt: wynikiem działania powyższego skryptu jest zastosowanie funkcji CD(), która w zależności od wprowadzonej zmiennej sterującej zwraca wartość tej zmiennej inkrementowanej lub dekrementowanej.

5. Instrukcja switch

   Instrukcja switch pozwala programowi na sprawdzenie ciągu warunków i próbuje wykonać wartość wyrażenia przypisaną do odpowiedniej etykiety case. Jeśli wartość wyrażenia jest znaleziona, program wykonuje załączoną instrukcję. Schemat instrukcji switch:

```
switch (expression){
   case label :
      instrukcja1;
      break;
   case label :
      instrukcja2;
      break;
   ...
   default : instrukcja3;
}
```

   Przykład zastosowania instrukcji switch:

```
<SCRIPT LANGUAGE="JavaScript">
x="cytryny";
switch (x)
{
case "cytryny" :
document.write("Cena cytryn 2,50");
break;
case "pomarańcze" :
document.write("Cena pomarańczy 3,50");
break;
default : document.write("Nie podano towaru");
}
</SCRIPT>
```

   efekt: wynikiem działania powyższego skryptu będzie wyświetlenie komentarza o cenie produktu w zależności od tego czym dany produkt jest – zmienna x.

6. Instrukcja for

Pętla for jest powtarzana aż do momentu, kiedy testowany warunek staje się fałszywy. JavaScript pętla for jest składniowo podobna do pętli w Java i C. Instrukcja pętli for wygląda następująco:

```
for ([przypisanie]; [warunek]; [zmiana])
{
   Instrukcje
}
```

Wykonywane są kolejno:

a. Inicjalizacja wyrażenia przypisanie.
b. Obliczenie wyrażenia warunek. Jeśli wartość warunek jest prawdziwa, instrukcja jest wykonana. Jeśli wartość warunek jest fałszywa, pętla for jest przerwana.
c. Wykonywanie instrukcji wewnętrznych.
d. Aktualizacja wyrażenia zmiana, i przejście do kroku drugiego.

Zastanowić się a następnie wykonać ponizszy skrypt uzupełniając go o brakujące frazy i wprowadzając do treści strony www:

```
for (i=1;i<11;i++)
document.write( (i%3)  +"<BR>");
```

efekt: wynikime działania ma być wyświetlenie na stronie poszczególnych wartości licznika

7. Instrukcja do...while

Analogia do pętli repeat w Pascalu. Instrukcja do...while powtarza warunek, aż do momentu, kiedy uzna go za fałszywy. Instrukcja do...while wygląda następująco:

```
do
   instrukcje
while (warunek);
```

Instrukcje zostaną wykonane raz, zanim warunek zostanie sprawdzony. Jeśli warunek jest prawdziwy (true), instrukcje zostaną wykonane ponownie. Warunek jest sprawdzany na końcu każdego wykonania. Kiedy warunek jest fałszywy (false), wykonanie zostaje zatrzymane i kontrola jest przekazywana do instrukcji następującej po pętli do...while. Pętla iteracyjna wykonuje się co najmniej raz – wykonanie pętli odbywa się wielokrotnie dopóki wartość i jest mniejsza niż 5.

Wprowadzić poniższy kod do strony www a następnie sprawdzić efekt jego działania.

```
x=0;
do {
   i += 1;
   document.write(i);
} while (i < 5);
```

efekt: wynikime działania skryptu jest wyświetlenie poszczególnych wartości licznika w pętli

8. Instrukcja while
Działa identycznie jak pętla while w Pascalu. Instrukcja while wykonuje instrukcje tak długo, dopóki warunek będzie prawdziwy. Instrukcja pętli while wygląda następująco:

```
while (warunek) {
   instrukcje
}
```

Test warunku ma miejsce przed wykonaniem instrukcji. Jeśli warunek będzie prawdziwy, instrukcje są wykonywane i ponownie jest badany warunek. Jeśli warunek będzie fałszywy, wykonywanie pętli jest zatrzymanie i następuje przejście o następnej instrukcji po instrukcji while.
Następująca pętla while wykonuje iterację (powtarza) gdy warunek jest spełniony (n jest mniejsze od 10):

Wprowadzić okreslony przykład do strony www a następnie sprawdzić jego działanie

```
n = 0;
x = 0;
while( n < 10 ) {
   n ++;
   x += n;
   document.write(x+"<BR>");
}
```

efekt: wynikiem działania jest wyprowadzenie na ekran wartości zmiennej x

9. Obiekt Math
Obiekt Math przechowuje wartości matematyczne, gromadzone jako właściwości i metody. Są tutaj przechowywane pewne stałe i funkcje matematyczne.
Składnia: Math.property lub Math.method, gdzie property lub method jest jednym z podanych niżej elementów.
Właściwości (stałe):

E	e - stała Eulera, która wynosi ok. 2.718
LN2	logarytm naturalny dwóch, tj. ok. 0.693
LN10	logarytm naturalny z dziesięciu, tj. ok. 2.302
LOG2E	logarytm o podstawie 2 z liczby Eulera, czyli ok. 1.442
LOG10E	logarytm o podstawie 10 z liczby Eulera, czyli ok. 0.434
PI	wartość liczby π, czyli ok. 3.14159
SQRT1_2	pierwiastek kwadratowy z 0.5, czyli ok. 0.707
SQRT2	pierwiastek kwadratowy z 2, czyli ok. 1.414

Metody:

abs(liczba)	wartość bezwzględna liczby
acos(liczba)	arcus cosinus z liczby (podanej w radianach)
asin(liczba)	arcus sinus z liczby (podanej w radianach)
atan(liczba)	arcus tangens z liczby (podanej w radianach)

ceil(liczba)	najmniejsza liczba całkowita, większa lub równa podanej liczbie
cos(liczba)	cosinus liczby (podanej w radianach)
exp(liczba)	wartość e podniesionej do potęgi liczba
floor(liczba)	największa liczba całkowita mniejsza lub równa podanej liczbie
log(liczba)	logarytm naturalny liczby
max(liczba1,liczba2)	większa z dwóch liczb
min(liczba1,liczba2)	mniejsza z dwóch liczb
pow(liczba1,liczba2)	wartość liczby1 podniesionej do potęgi liczby2
random()	wartość pseudolosowa z przedziału (0,1)
round(liczba)	zaokrąglenie danej liczby do najbliższej liczby całkowitej
sin(liczba)	sinus liczby (podanej w radianach)
sqrt(liczba)	pierwiastek kwadratowy liczby
tan(liczba)	tangens liczby (podanej w radianach)

Ponizszy przykład wprowadzić na stronę i sprawdzić jego działanie

```
<SCRIPT LANGUAGE="JavaScript">
for (var i=0; i<91; i++)
     {
     document.write(i," ",Math.sin(i*PI/180)+"<br>");
     }
</SCRIPT>
```

efekt: wynikiem działania będzie wizualizacja działania funcji sinus.

10. Przykłady funkcji

Funkcje są definiowane między znacznikami <HEAD> i </HEAD>. Pozwala to na załadowanie ich na samym początku, aby dowolny skrypt na stronie mógł rozpocząć pracę, gdy użytkownik zacznie przeglądać stronę. Funkcje są definiowane przez określenie ich nazwy.

Przykład 1: FunkcjaDodaj15 dodaje 15 do liczby, która jest argumentem funkcji

```
<script language="JavaScript">
function Dodaj15(i)
{ return(i + "+15 daje " + (i+15)) }
</script>
```

To, co funkcja ma zrobić po wywołaniu, jest zdefiniowane między nawiasami { }
Aby wywołać funkcję, trzeba umieścić wykonanie funkcji:

```
Dodaj15(5)
```

w tekście skryptu, w ramach znaczników <SCRIPT> i </SCRIPT>. Liczba 5 może być zmieniona na dowolną inną.
Skrypt z wykorzystaną funkcją:

```
<script>
document.write(Dodaj15(5));
</script>
```

JavaScript rozróżnia wielkość liter. Dodaj15() nie jest tym samym co dodaj15(), a document.write() nie jest tym samym co Document.WRITE().

Przykład 2: Sprawdzić działanie skryptu wykorzystującego własną *funkcję nacisnij*. alert jest metodą dla obiektu window, tworzącą okienko dialogowe z napisem informacyjnym.

```
<html>
<head>
<script language="JavaScript"> //definicja funkcji
   function nacisnij()
      { alert("Witaj!"); }
</script>
</head>
<body>
<form>
<input type="button"
   name="Button1"
   value="Nacisnij mnie"
   onclick="nacisnij()//wykorzystanie funkcji">
</form>
</body>
</html>
```

Zwróćmy uwagę na przypisanie wykonania funkcji do atrybutu onclick obiektu INPUT. Można wykorzystać również inne atrybuty:

a. onblur - atrybut definiuje reakcję na opuszczenie aktywnego elementu,
b. ondblclick - atrybut definiuje reakcję na podwójne kliknięcie myszy,
c. onfocus - atrybut definiuje reakcję na uaktywnienie elementu,
d. onkeydown - atrybut definiuje reakcję na wciśnięcie klawisza.

Przykład 3:

```
<html>
<head>
<script language="JavaScript">
   function nacisnij(x)
      { alert(x); }
</script>
</head>
<body>
<form>
<input type="text"
name="Button1" onfocus='nacisnij("Edycja")'
onblur='nacisnij("Kliknales poza polem")'>
</form>
</body>
</html>
```

Pobieranie danych z pól edycyjnych i operacje na ich wartościach

Przykład pokazuje możliwość wykonania funkcji pobierającej daną z pola edycyjnego w powiązaniu ze zdarzeniem kliknięcia przycisku i wyświetlenie wartości w oknie alert:

```
<HTML>
<META http-equiv="Content-Type" content="text/html;
charset=Windows-1250"/>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
function pobierz(s) {
    alert("Cześć  "+ s+"!");
  }
</SCRIPT>
<BODY>
Wpisz swoje imię:
<form>
  <input type="text" name="pole">
  <input type="button" name="Button1" value="Naciśnij"
onclick="pobierz(pole.value)">
  </form>
</BODY>
</HTML>
```

Można wykonywać operacje obliczeniowe na danych wpisywanych do pól edycyjnych. Wystarczy wewnątrz ciała funkcji umieścić przykładowy kod:

```
s++;
alert("Wynik zwiększenia o 1: "+ s);
```

A co pokaże okno alert, gdy zapiszemy tak:

```
s=s+1;
alert("Wynik zwiększenia o 1: "+ s);
```

Funkcje Number i String przekonwertują na obiekt numeryczny lub łańcuch znaków. Można zatem zapisać tak:

```
alert(Number(str)+0.5);
```

Zadania do wykonania

1. Wykonać przykłady z instrukcji, umieścić je pod odpowiednimi hiperłączami na swojej stronie internetowej.
2. Wstawić na stronie pole edycyjne do wpisania imienia odwiedzającego oraz spowodować pojawienie się okna powitalnego po opuszczeniu pola edycyjnego.
3. Wykonać kalkulator, którego zadaniem będzie wpisywanie wartości do dwóch pól edycyjnych, sumowanie ich i wyświetlanie sumy w okienku alertu.
4. Sprawdzić wykonywanie różnych działań dla różnych technik obsługi myszki – przypisanie do odpowiednich atrybutów: onblur, onkeydown itp
5. Umieścić na stronie odnośnik do pliku html wykonującego dodawanie dwóch liczb całkowitych podawanych w dwóch polach edycyjnych i wyświetlenie sumy w okienku alert. To samo dla liczb dziesiętnych.