JavaScript - wykład 1 Podstawy

Beata Pańczyk

Plan wykładu

- Historia JavaScript
- Wersje języka
- Zastosowania
- Sposoby osadzania skryptów na stronie HTML
- Podstawy języka
- Obiekty JavaScript

2

Język JavaScript

- JavaScript – wprowadzony przez firmy Netscape i Sun język skryptowy (nie wymaga kompilacji) wspomagający tworzenie dokumentów HTML
- Wykonywanie poleceń JavaScript - przeglądarka WWW (interpretacja **po stronie klienta**)
- JavaScript dodaje do stron WWW interaktywność – możliwość reagowania na polecenia użytkownika
- Język zorientowany obiektowo, składnia i podstawowa struktura wzorowana na C/C++
- Jeden ze składników DHTML, czyli dynamicznego HTML (z ang. *Dynamic HTML*) .
- DHTML - technika tworzenia i modyfikacji elementów strony za pomocą języka skryptowego

3

Historia

- połowa lat 90 - w firmie Netscape powstał język *LiveScript*
- 1995 - przemianowany na *JavaScript*
- główny twórca - Brendan Eich
- kolejne lata - kolejne wersje
- ustandaryzowany przez ECMA (międzynarodowa organizacja standaryzacyjna, skrót od dawnej nazwy *European Computer Manufacturers Association*)
- zamieszanie w nazewnictwie - używane są nazwy *JavaScript*, *ECMAScript* oraz (dla produktów firmy Microsoft) *JScript*.
- wersje najpopularniejszych przeglądarek (od FireFox 1.5, Internet Explorer 6, Netscape 8, Opera 8) są zgodne z ECMAScript v3, czyli JavaScript 1.5 oraz JScript 5.5 i 5.6

4

Wersje

ECMA Script	JavaScript Netscape (1996-2008)	Jscript Microsoft (1996-2006)
	1.0	1.1
	1.1	2.0
	1.2	-
V1	1.3	3.0
-	1.4	-
V3	1.5	5.5, 5.6
V4	2.0	5.7., .NET

5

Zastosowania

- Budowa stron reagujących na działania użytkownika
- Sprawdzanie poprawności formularzy
- Obsługa elementów interfejsu strony
- Dynamiczne generowanie treści oraz realizacja wielu innych efektów

6

Umieszczanie skryptów w kodzie HTML

- **Skrypty osadzone** - umieszczane wewnątrz kodu HMTL przy użyciu znacznika <script> z obligatoryjnym parametrem type, o ile dokument ma być zgodny ze specyfikacją HTML 4
- **Skrypty zewnętrzne** - umieszczenie skryptu w oddzielnym pliku i użycie znacznika <script> z parametrem src

7

Skrypty osadzone

- <script type="text/javascript">
  //tutaj kod skryptu
  </script>
- większość przeglądarek akceptuje również znacznik <script> nie zawierający argumentu type
- language – parametr dawniej określający język skryptu (np. language="javascript"); nie występuje w ścisłej (*strict*) wersji standardu HTML 4 i obecnie nie należy go stosować (z drugiej strony - starsze przeglądarki, obsługujące standard HTML wcześniejszy niż 4.0, rozpoznawały tylko parametr language, nie rozpoznając natomiast parametru type)

8

Kod JavaScript w HTML<4

```
<html>
<head>
   <script language="javascript">
   <!-- ukrycie skryptu
         function fun() { ... }
    -->
   </script>
</head>
```

9

Kod JavaScript w HTML 4.x

- kod w bloku znaczników <script>...</script>
- wymagany parametr:
  - type (dla JavaScriptu "text/javascript")
  - opcjonalnie parametr language oznaczający minimalny numer wersji JavaScript
- Np.:

```
<script type="text/javascript">
   function fun() { ... }
</script>
```

10

Kod JavaScript w XHTML

- nie występuje parametr language
- specyfikacja XML wymaga, by umieścić treść skryptu wewnątrz sekcji CDATA, jeśli stosowane są znaki "<" czy ">"
- <script type="text/javascript">
  // <![CDATA[
  function fun() { ... }
  //]]>
  </script>
  Znaki komentarza "//" zapobiegają błędnemu zinterpretowaniu tej sekcji przez starsze przeglądarki

11

Kod JavaScript w HTML5

- <script>
  function myFunction() {
  document.getElementById("demo").innerHTML = "Paragraph changed.";
  }
  </script>
- <script src="myScript.js"></script>

12

Skrypty zewnętrzne

- oddzielny plik zwykle z rozszerzeniem *.js*
- <script type="text/javascript"
  src="lokalizacja skryptu">
  </script>
  <script type="text/javascript"
  src="http://nazwa.domeny/skrypty/skrypt.js">
  </script>
- znacznik zamykający </script> jest niezbędny i nie należy go pomijać.

13

Pełny dokument ze skryptem

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="utf-8" >
   <title>Moja strona WWW</title>
   <script src="http://nazwa.domeny/skrypty/skrypt.js">
   </script>
</head>
<body>
   <script >
      //kod JS
   </script>
</body>
</html>
```

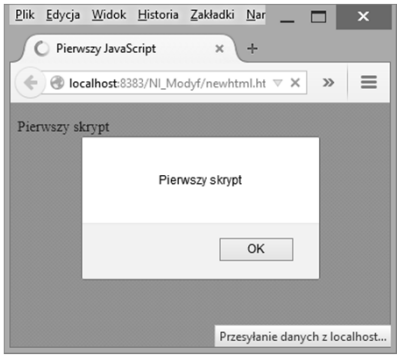
14

Prosty skrypt w HTML5

```
<html>
  <head>
    <title>Pierwszy Java Script</title>
    <meta charset="UTF-8">
    <meta      name="viewport"
               content="width=device-width, initial-scale=1.0">
    <script>
      function f1(){
        window.alert("Pierwszy skrypt");
        }
    </script>
  </head>
  <body onload="f1()" >
       <p>Pierwszy skrypt</p>
</body>
 </html>
```

15

Wynik działania skryptu



16

Blok <script>

- Dowolna liczba bloków <script> na stronie
- Kolejność osadzania skryptów ma znaczenie - będą wykonywane w kolejności umieszczenia na stronie
- Przeglądarka może wstrzymywać ładowanie i wyświetlanie strony póki nie wykona instrukcji skryptu

17

Komentarze w kodzie skryptu

- **Komentarz blokowy**
  <script>
  / ... treść komentarza blokowego... */*
  </script>
- **Komentarz wierszowy**
  <script>
  //treść komentarza liniowego...
  </script>
- Komentarz wierszowy może się znaleźć w środku komentarza blokowego:
  */**
  //treść komentarza
  **/*

18

JavaScript w przeglądarkach

- Implementacje JavaScriptu w przeglądarkach dostarczają obiektów pozwalających na:
  - modyfikowanie zawartości dokumentu (obiekt document)
  - manipulowanie oknami i wyświetlanie okien dialogowych (obiekt window)
  - pobieraniem informacji o przeglądarce (obiekt navigator)
  - obsługę zdarzeń (np. kliknięcia na przycisk)

19

Podstawy języka JavaScript

- Rozróżnianie wielkości liter (domyślna pisownia małymi literami)
- Zakończenie instrukcji - opcjonalny znak średnika
- Bloki instrukcji – w nawiasach klamrowych { }
- Nazwy zmiennych:
  - mogą zaczynać się od litery, znaku '_'
  - nie mogą zawierać spacji i polskich liter
  - nie mogą być słowami kluczowymi języka JavaScript

20

Typy zmiennych

- liczbowy (*number*) - nie ma rozróżnienia na typy całkowitoliczbowe i rzeczywiste (zmiennopozycyjne). Liczby zapisywane są za pomocą literałów liczbowych
- łańcuchowy (*string*) - w pojedynczym lub podwójnym cudzysłowie
- logiczny (*boolean*) : true,false
- specjalny typ null – wartość pusta
- obiektowy

21

Literały

- Literały – wartości zmiennych, np.
  - literał znakowy "złoty", 'kolor'
  - literał liczbowy 2.35e-4, 0.1E-1
- Znaki specjalne w literałach znakowych (stałe znakowe) - jak w C \b , \f, \n, \r, \t, \', \\
- Znak w szesnastkowym kodzie Unicode \uXXXX np. "\u0041" - znak"A"
- Przykłady literałów liczbowych
  - 3.142 liczba zmiennoprzecinkowa
  - 314E-2 liczba zmiennoprzecinkowa
  - 26 liczba całkowita
  - 076 liczba ósemkowa
  - 0x9F liczba szesnastkowa

22

Definiowanie zmiennych

- Definicja zmiennej – słowo kluczowe var (opcjonalne)
- Przykłady definiowania zmiennych

```
var liczba;          var cena=10;
var liczba=7;        var waluta="$";
liczba=7;            var napis=cena+waluta;
```

- Brak konieczności definiowania zmiennych na początku programu!

23

Operatory

- Operatory arytmetyczne: +, -, /, *, %
- Operatory zwiększania i zmniejszania: - - , ++
- Operatory porównania: ==, !=, >, <, >=, <=
  === równe wartością i typem
  !== różne wartością i typem
- Operatory logiczne: &&, | |, !
- Operatory przypisania: =, +=, -=, *=, /=, %=
- Operatory bitowe: >>, <<, &, |, ^, ~

24

Konwersja typów i konkatenacja

- JavaScript - język o łagodnej kontroli typów
- Zmienne i literały różnych typów można bez potrzeby wcześniejszego uzgadniania typów porównywać, łączyć itp.
- Łączenie typu znakowego i liczbowego operatorem "+" – wynik typu znakowego
- Przykład - najprostszy przypadek konwersji typów:

```
t="2000";
t+=10;                    //Wynik:   t="200010"
```

25

Tablice

- var nazwa_tab=Array(rozmiar_tablicy);
  np.:
  var tablica=Array(10);
- tablice o niezdefiniowanej długości można tworzyć za pomocą konstruktora **new** np.

```
var tab1=new Array();
var tab2=new Array("Rok","2000");
var tablica = new Array("wart1", "wart2", "wart3");
```

- odwołania do elementów tablicy np.: tab2[0]
- numerowanie elementów tablicy od **0**

26

Instrukcja warunkowa i switch

- if (warunek) {instrukcje}
  else if (warunek)
  {instrukcje}
  else {instrukcje}
- (warunek)? wartość1:wartość2
- switch (zmienna)

```
{   case wartość_stała1 : instrukcje;break;
    case wartość_stała2 : instrukcje;break;
    //.....................................
    default:instrukcje;
}
```

- np. if (liczbaKsiazek >= 5)

```
            alert("Prezent!");
    else    alert("Dziękujemy!");
```

27

Iteracje

- while (warunek)
  {instrukcje}
- for(licz=pocz;licz<=koniec;licz++)
  {instrukcje}
- do { instrukcje)
  while (warunek)
- for dla obiektów (przechodzi przez wszystkie pola danego obiektu - w tym elementy tablicy):
  for (wlasnosc in obiekt)
  {instrukcje}

28

Inne instrukcje

- Przerwanie wykonywania pętli i przekazanie sterowania na jej początek continue
- Przerwanie wykonywania pętli i prze-kazanie sterowania do instrukcji za pętlą break
- Instrukcja with
  with (nazwa_obiektu)
  {instrukcje}

29

Funkcje

- Umieszczanie funkcji - w nagłówku dokumentu
- Funkcje (zmienne lokalne funkcji deklaruje się jako **var**, zwracane rezultaty poprzedza się słowem kluczowym **return**)

```
function nazwa_funkcji(arg1,arg2,...)
{ instrukcje;
  return zmienna
}
function nazwa_funkcji()
{ instrukcje;
  return zmienna
}
```

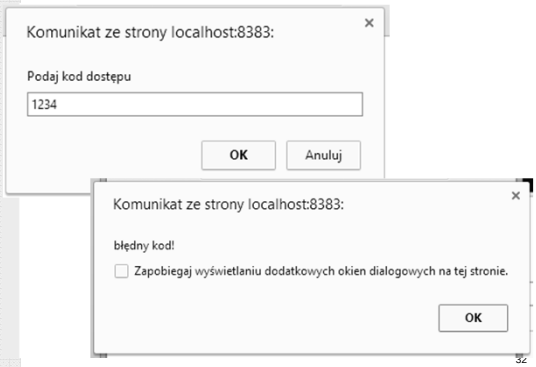
30

Przykład zastosowania funkcji

```
<html><head>
<script >
    function sprawdzkod(kod)
    {       if(kod==1313)return true;
            return false;
    }
</script>
<script >
    var kod=window.prompt("Podaj kod dostępu"," ");
    if(sprawdzkod(kod))
     window.alert("właściwy kod!");
    else
     window.alert("błędny kod!");
</script> </head>
<body></body>
</html>
```

31

Przykładowy wynik działania



32

Zmienne lokalne i globalne

- Zmienne lokalne - występujące w funkcjach (lokalne dla danej funkcji)
- Zmienne globalne – występujące poza funkcjami
- Zmiennej lokalnej nie można wywołać poza funkcją
- np.

```
function piszLublin(tekst)
{
        var nazwa="Lublin: ";     //lokalna
        window.alert(nazwa + tekst);
        return true;
}
```

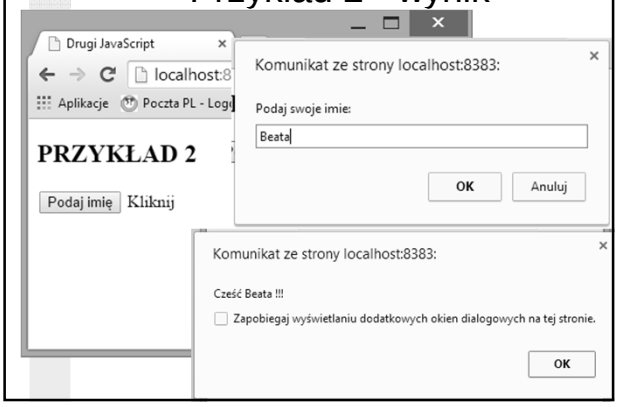
33

JavaScript - Przykład 2

```
<head>
<title>Drugi przkład JavaScript</title>
<script>
 function imie()
 {  w=window.prompt("Podaj swoje imie:","");
    if (w=="") { window.alert("Cześć ANONYMOUS?"); }
    else  { window.alert("Cześć "+w+" !!!"); }
 }
</script> </head>
<body><div>
  <h2> PRZYKŁAD 2 </h2>
  <p><input name="imie" type="button" value="Podaj  imię"
    onclick="imie()" /> Kliknij
  </p>
</div></body>
```

34

Przykład 2 - wynik



Obiekty

- **obiekty** definiuje się dwuetapowo - najpierw strukturę obiektu (prototyp) a na podstawie prototypu tworzy się faktyczny obiekt
- w JavaScript wszystko jest obiektem,
- Object - podstawowy obiekt
- Obiekty JavaScript:
  - Array (tablica)
  - String (łańcuch tekstowy)
  - Number (liczba całkowita lub rzeczywista)
  - Boolean (wartość logiczna)
  - Function (funkcja JavaScriptu)
  - Date (data)
  - Math (operacje matematyczne)
  - RegExp (wyrażenia regularne)

36

Dostęp do pól i metod

- Obiekty JavaScriptu są tablicami asocjacyjnymi (słownikami)
- Dostęp do pól obiektów przy użyciu dwóch równoważnych notacji:
  obiekt.pole
  obiekt["pole"]
- metody obiektu (funkcje) są jego polami - dostęp jest możliwy przy użyciu notacji z kropką lub notacji z nawiasami kwadratowymi
- np.

```
m.metoda1();
m["metoda1"]();
```

37

Instrukcja wiążąca with

- Instrukcje:

```
obiekt.pole1=wartość;
obiekt.pole2=wartość;
```

- Równoważne:

```
with (obiekt)
{     pole1=wartość;
      pole2=wartość;
}
```

38