

Laboratorul 6

JavaScript - partea I

1. Introducere

JavaScript este unul dintre cele mai folosite limbaje de programare ce se bazează pe "script"-uri. Termenul script este folosit pentru a defini un program ce poate însoți un document HTML sau poate fi încorporat într-un fișier HTML. Programul este executat pe mașina client atunci când documentul este încărcat sau la alt moment de timp (când o legătură este activată sau a apărut un anumit eveniment). Script-urile permit adăugarea de elemente de interactivitate unui document HTML.

JavaScript și Java sunt limbaje complet diferite. Spre deosebire de JavaScript, Java este un limbaj complex și mult mai puternic.

Un JavaScript poate fi folosit pentru a realiza următoarele operații:

- adăugarea de text într-un document HTML în mod dinamic;
- reacționarea la diferite evenimente;
- citirea sau scrierea de elemente HTML;
- validarea datelor dintr-un formular, permițând degrevarea server-ului de această operație;
- detectarea "browser"-ului folosit pe mașina client, în vederea încărcării unei pagini create special pentru tipul respectiv de "browser".

2. JavaScript și documente HTML

2.1. JavaScript în documente HTML

Pentru a insera un JavaScript într-un document HTML se folosește eticheta **<script>**. Această eticheta folosește atributul *type* ce permite definirea limbajului folosit. Astfel, un JavaScript este introdus într-un document HTML prin intermediul următoarei construcții:

```
<script type="text/javascript">  
.....  
.....  
</script>
```

Momentul în care un JavaScript este executat depinde de poziția acestuia în cadrul documentului HTML.

Un JavaScript poziționat în secțiunea **<body>** va fi executat la încărcarea documentului HTML, el contribuind la generarea paginii HTML.

```

<html>
<body>

<script type="text/javascript">
document.write("<b>Acest text a fost generat cu ajutorul unui
JavaScript!!!</b>");
</script>

</body>
</html>

```

Cand un JavaScript este pozitionat in sectiunea **<head>**, el va fi executat atunci cand este apelat sau la aparitia unui anumit eveniment. Pozitionarea sa in sectiunea **<head>** asigura faptul ca va fi incarcat inainte de a fi utilizat iar executarea sa va fi realizata prin intermediul funcțiilor.

```

<html>
<head>
<script type="text/javascript">
function mesaj()
{
alert("Acest mesaj a fost apelat prin intermediul
evenimentului onload");
}
</script>
</head>

<body onload="mesaj()">
</body>
</html>

```

2.2. JavaScript extern

Atunci cand se doreste utilizarea unui JavaScript in mai multe documente HTML solutia este folosirea unui JavaScript extern.

JavaScript-ul va fi salvat intr-un fisier separate cu extensia **.js**. Este de remarcat faptul ca in cazul utilizarii unui JavaScript extern eticheta **<script>** nu apare in fisierul **.js** ci in documentul HTML, in locul unde in mod normal s-ar scrie script-ul. Pentru a indica script-ul extern ce va fi folosit se foloseste atributul **src** al etichetei **<script>**.

```
<script type="text/javascript" src="nume.js"></script>
```

```

script_ext.js
function mesaj()
{
alert("Acest mesaj a fost apelat prin intermediul
evenimentului onload");
}

```

index.html

```
<html>
<head>
<script type="text/javascript" src="script_ext.js"></script>
</head>

<body onload="mesaj()" >
</body>
</html>
```

3. Sintaxa JavaScript

3.1. Comenzi JavaScript

Un JavaScript este format dintr-o secventa de comenzi ce sunt executate de catre "browser".

- **Atentie:** JavaScript face diferenta intre minuscule si majuscule!
- Folosirea caracterului ";" la sfarsitul unei comenzi nu este obligatorie, sfarsitul de linie fiind interpretat drept sfarsit de comanda. Acest caracter permite scrierea mai multor comenzi JavaScript pe aceeasi linie.
- Comenzile JavaScript pot fi grupate in blocuri de comenzi prin intermediul caracterelor "{" si "}".
- Comentariile in JavaScript pot fi pe o singura linie ("//")sau pe mai multe linii ("/*" si "*/").

O comanda foarte folosita in JavaScript este **document.write**. Ea permite scrierea unui anumit text intr-o pagina HTML.

```
<html><body>
.....
<script type="text/javascript">
document.write("<p>Acesta este primul paragraf.</p>");
document.write("<p>Acesta este al doilea paragraf.</p>");
</script>
.....
</body></html>
```

Continutul paginii HTML afisate va contine si cele doua paragrafe introduse prin intermediul JavaScript-ului.

3.2. Variabile

Numele variabilelor trebuie sa inceapa cu o litera sau cu caracterul "_". Valoarea unei variabile se poate modifica pe parcursul executiei unui JavaScript. Numele variabilei este folosit pentru a afisa sau modifica valoarea sa.

Variabilele pot fi declarate cu comanda **var**. Dupa declarare, variabilele sunt goale, nu au valori.

```
var x;
```

Exista si posibilitatea de a atribui o valoare variabilei inca de la declarare:

```
var x=2;
```

Atribuirea unei valori unei variabile ce nu a fost declarata provoaca declararea automata a variabilei. De asemenea, redeclararea unei variabile intr-un JavaScript nu modifica valoarea variabilei.

Cand se doreste atribuirea unei valori text unei variabile se folosesc caracterele "" pentru a incadra valoarea text.

3.3. Operatorii aritmetici, de comparare, logici si de atribuire

Operatorii aritmetici utilizati in JavaScript sunt identici cu cei din C.

Operatorii de atribuire utilizati in JavaScript sunt prezentati impreuna cu niste exemple in tabela urmatoare (se considera ca $x=10$ si $y=5$):

Operator	Exemplu	Identic cu	Rezultat
=	$x=y$		$x=5$
+=	$x+=y$	$x=x+y$	$x=15$
-=	$x-=y$	$x=x-y$	$x=5$
=	$x=y$	$x=x*y$	$x=50$
/=	$x/=y$	$x=x/y$	$x=2$
%=	$x%=y$	$x=x\%y$	$x=0$

Operatorul "+" poate fi folosit si in cazul sirurilor, in acest caz el fiind operator de concatenare. Daca se aduna o variabila de tip sir de caractere cu o variabila numerica, rezultatul va fi un sir de caractere.

Operatorii de comparare utilizati in JavaScript sunt identici cu cei din C, diferit fiind operatorul "===" ce este prezentat intr-un exemplu mai jos (se considera ca $x=5$):

Operator	Descriere	Exemplu
===	este egal exact cu (tip si valoare!)	$x===5$ este A $x===\text{"5"}$ este F

Operatorii logici utilizati in JavaScript sunt identici cu cei din C. De asemenea, in JavaScript exista un operator conditional ce are aceeasi sintaxa ca operatorul conditional din C.

3.4. Comenzi conditionale

Comenzile conditionale sunt identice cu cele din C. Sintaxa lor este identical cu cea a comenzilor conditionale din C. De notat este faptul ca toate instructiunile sunt scrise cu minuscule. Utilizarea majusculilor genereaza eroare JavaScript.

```
<script type="text/javascript">
var data = new Date();
var ora = data.getHours();

if (ora < 10) document.write("Buna dimineata!");
else document.write("Buna ziua!");
</script>
```

Exercitiu

Pornind de la JavaScript-ul prezentat drept exemplu, sa se implementeze un JavaScript ce va afisa "Buna dimineata", "Buna ziua", "Buna seara" si "Noapte buna" in functie de ora sistemului. Se vor implementa si varianta cu JavaScript inclus in documentul HTML dar si varianta cu JavaScript extern.

3.5. Instructiuni repetitive

Instructiunile repetitive **for**, **while do** si **do while** au aceeasi sintaxa cu cea a instructiunilor din C. De asemenea, instructiunile **break** si **continue** pot fi folosite la fel ca in limbajul C.

Instructiunea repetitiva **For In** este utilizata pentru a parcurge toate elementele unui vector sau pentru a parcurge toate proprietatile unui obiect. Sintaxa sa este identica cu sintaxa instructiunii din Java.

3.6. Functii

Daca se doreste ca un JavaScript sa nu fie executat atunci cand pagina este incarcata se poate introduce codul respective intr-o functie. Functia va fi executata atunci cand este apelata sau la aparitia unui eveniment. O functie poate fi apelata din orice loc din document sau chiar din alte pagini (daca functia este inclusa intr-un JavaScript extern). Desi o functie poate fi definita in sectiunea **<head>** sau in sectiunea **<body>**, pentru a avea siguranta ca functia este incarcata de catre "browser" inainte de a fi apelata este de preferat ca functia sa fie definita in sectiunea **<head>**.

```
<html><head>
<script type="text/javascript">
function displaymessage()
{
alert("Hello World!");
}
</script></head>
<body>
<form>
<input type="button" value="Click me!"
onclick="displaymessage()" >
</form>
</body></html>
```

Sintaxa crearii unei functii este urmatoarea:

```
function nume_functie(var1,var2,...,varN)
{
.....
}
```

Daca o functie returneaza o anumita valoare, returnarea valorii se face cu instructiunea **return**. Functia prezentata mai jos calculeaza si returneaza valoarea produsului a doua numere.

```
<html>
<head>
<script type="text/javascript">
function produs(a,b)
{
return a*b;
}
</script>
</head>

<body>
<script type="text/javascript">
document.write(produs(4,3));
</script>

</body>
</html>
```

3.7. Durata de viata a variabilelor JavaScript

Daca o variabila este declarata intr-o functie, variabila poate fi accesata doar pe durata functiei. La terminarea functiei variabila este distrusa (variabila locala). Daca o variabila este declarata globala (in afara functiilor), ea va putea fi accesata de catre toate functiile din pagina. Aceste variabile sunt distruse la inchiderea paginii HTML.

3.8. Ferestre de dialog in JavaScript

JavaScript permite crearea a 3 tipuri de ferestre de dialog:

a. de alertare

Fereastra de dialog de alertare are rolul de a transmite informatii catre utilizator. Utilizatorul trebuie sa actioneze butonul **OK** pentru a continua executia. Comanda ce produce aparitia unei astfel de ferestre este:

```
alert("Mesaj de atentionare");
```

b. de confirmare

Fereastra de dialog de confirmare este folosita in cazurile in care se doreste ca utilizatorul sa accepte sa sa verifice un anumit lucru. Optiunile oferite de acest tip de fereastra sunt **OK** si **Cancel**. Daca este selectat butonul **OK**, valoarea returnata este TRUE. Daca este selectat butonul **Cancel**, valoarea returnata este FALSE. Comanda ce produce aparitia unei ferestre de dialog de confirmare este:

```
confirm("Mesaj de confirmare");
```

c. de interogare

Fereastra de dialog de interogare este folosita in cazurile in care se doreste ca utilizatorul sa introduca o valoare inainte de a incarca un document HTML. Dupa introducerea unei valori, optiunile oferite de acest tip de fereastra sunt **OK** si **Cancel**. Daca este selectat butonul **OK**, valoarea returnata este valoarea introdusa de utilizator. Daca este selectat butonul **Cancel**, valoarea

returnata este NULL. Comanda ce produce aparitia unei ferestre de dialog de confirmare este:

```
prompt("Mesaj de interogare", "valoare initiala");
```

Exercitiu

Pornind de la JavaScript-ul realizat la exercitiul anterior, sa se modifice documentul astfel incat sa fie solicitat numele utilizatorului si, in functie de ora system, sa se afiseze "Buna dimineata, X!", "Buna ziua, X!", "Buna seara, X!" si "Noapte buna, X!" in functie de ora sistemului.

4. Evenimente

4.1. Evenimentele *onLoad* și *onUnload*

Evenimentul *onLoad* este folosit adesea pentru a verifica tipul și versiunea navigatorului, pentru a încărca versiunea potrivită de document html.

4.2. Evenimntele *onFocus*, *onBlur* și *onChange*

Evenimentele *onFocus*, *onBlur* și *onChange* sunt adesea utilizate în validarea câmpurilor formularelor.

```
<input type="text" size="30" id="email" onchange="checkEmail()">
```

4.3. Evenimentul *onSubmit*

Evenimentul *onSubmit* este utilizat pentru validarea câmpurilor unui formular înainte de transmiterea acestuia.

```
<form method="post" action="actiune.htm" onsubmit="return verif_campuri()">
```

4.4. Evenimentele *onMouseOver* și *onMouseOut*

Evenimentele *onMouseOver* și *onMouseOut* sunt utilizate de multe ori pentru a crea butoane animate.

```
<a href="http://ai.ac.tuiasi.ro" onmouseover="alert('A aparut un eveniment de tip onmouseover')">
```

```
Pagina disciplinei Aplicatii Internet
```

```
</a>
```