**Info Generale**

* **Script WEB** – cod pentru a face pagina WEB dinamica.
* Java si JavaScript sunt total diferite.
* JavaScript este si pentru front end si back end.
* JavaScript foloseste si el HTML pentru a afisa pe pagina chestii.
* JavaScript nu necesita compilari
* JavScript e orientat pe obiect, dar nu e bazat pe clase.

**Modul de adaugare in proiect**

1. Plasam scriptul in <head> </head> cu <script> </script>. Il plasam aici atunci cand avem nevoie sa il apelam sau are loc un eveniment. Asa el va fi incarcat si executat odata cu pagina.
2. Plasam codul in <body> </body> - atunci cand scriptul trebuie sa fie executat direct la incarcarea paginii.
3. Utilizam un fisier extern – atunci cand scripturile se pot repeta in cod. Cream un fisier .js si nu scrim <script> </script> in el, dar vom folosi in HTML <script src=” “> </script> pentru a-l incarca. E ca si cum scripurile din fisier .js ar fi puse in documentul HTML
4. Crearea unei gestionar de evenimente. Putem folosi direct anumite functii JS, in taguri, fara a mai defini scriptul. Gen:

**<p onclick="alert('Hello')">A</p>**

**Async**

Async inseamna ca codul JS se incarca fara a afecta incarcarea codului ce urmeaza din documentul HTML. Adica, pagina nu va astepta ca scriptul sa fie incarcat pentru a executa codul HTML de mai departe, ci o va face simultan.

Asa se foloseste:

<script async source=””></script>

**Reguli de sintaxa**

* // - comenatriu pe o linie
* /\* \*/ - comentariu pe mai multe linii
* Este case sensitive.
* Orice instructiune se termina cu ;
* “ “ sau ‘ ‘ sunt identice
* Primul element al variabilei poate fi o litera, \_ , sau $
* /f – indica o pagina noua

**Afisarea informatiei**

1. window.alert(“mesaj”) – deschide o fereastra de alertare mica
2. document.write(“mesaj”) – direct pe pagina. Dupa ce se afiseaza acest text, toate elementele HTML sunt sterse. De ex, daca se foloseste aceasta actiune la apasarea butonului, cand va fi apasat, va fi afisat un text, dar butonul va disparea.
3. innerHTML – adica asigura ca un text sa fie inlocuit cu altul.De e:

<p id="id">Hello</p>

<script>

**document.getElementById("id").innerHTML=4;**

</script>

1. In consola la browser

<script>

console.log(“hello”);

</script>

Mesajul “Hello” nu va fi afisat, ci va fi inlocuit cu 4.

<p id="click">Mesaj aparut pana la click doar!</p>

<button onclick='document.getElementById("click").

innerHTML="Salut"';>Click</button>

**Atentie! Cand adaugam direct scriptul in tag, il incepem si termiam cu ‘ dar stringurile din el vor fi intre “”.**

**Var si let**

* var – defineste variabila globala.

var x = 5;

Poate fi folosita pentru a spune ca variabila dintr-un alt scope se va referi la cea globala, de ex:

var x = 2;

if(a==10)

var x=20;

Document.write(x) – va fi afisat 20

* let – variabila locala disponibila doar intr-un scope.

let x = 2;

if(a==10)

let x=20;

Document.write(x) – va fi afisat 2

**Tipuri de date**

* string
* number
* boolean
* array
* object
* function – o variabila poate stoca o functie:

var a = alert(“Hello”)

* undefined
* null

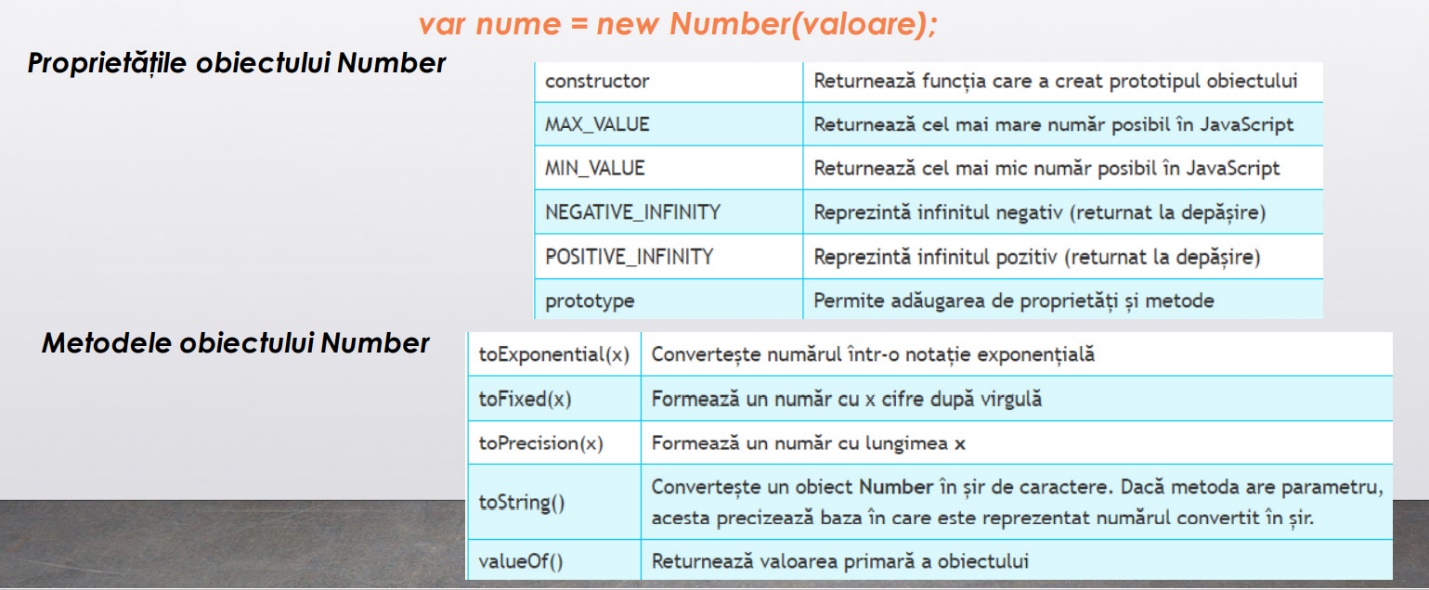
**variabile**

* Tipul variabilei poate fi modificat pe parcursul programului
* const – reprezinta constante

**Operatorul ===**

Este exact ca ==, doar ca compara si tipul varibilei.

**Number**



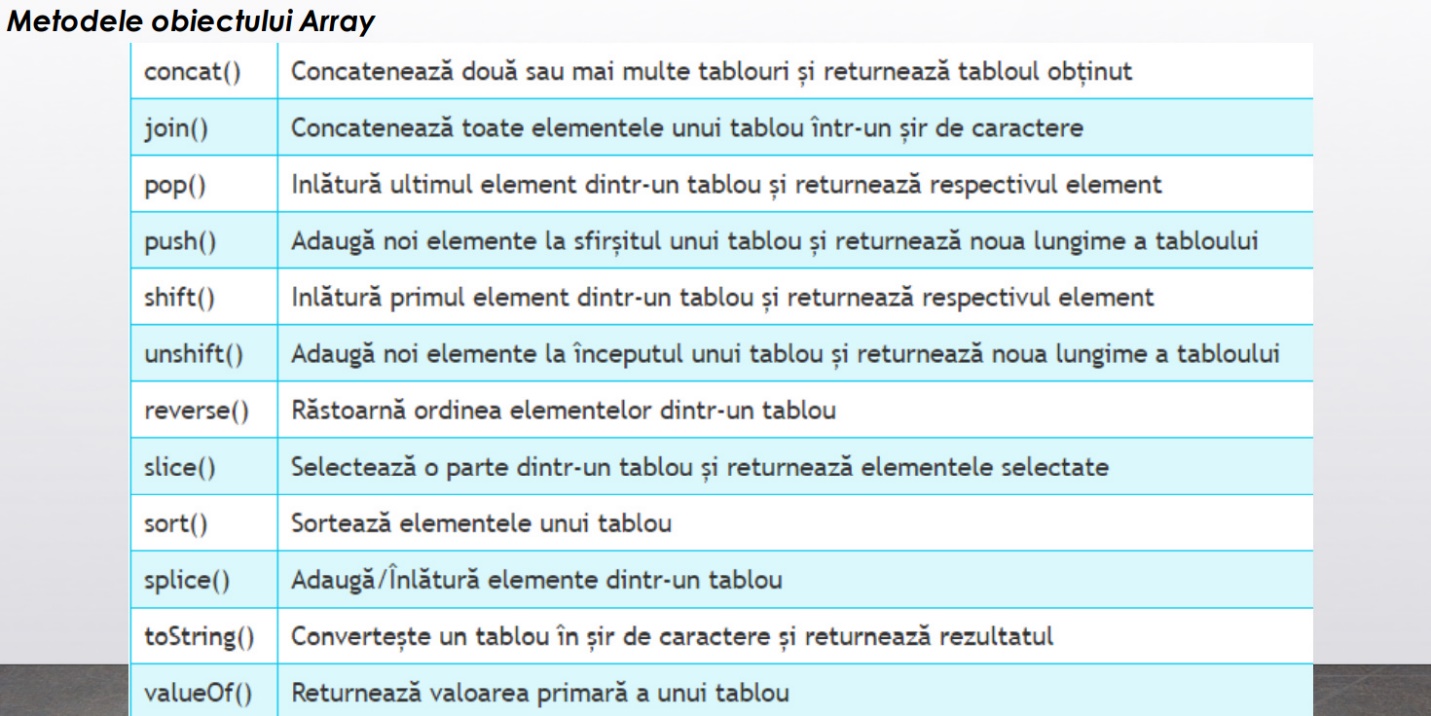
**Array**

* Folosim [] pentru elementele tabloului nu {}
* Putem crea un tablou si sa-i atribuim direct valori la definire:

var tablou = [1,2,3,4]

Nu punem [] la definirea variabilei cand dam tabloul direct!

* var tablou[]; - tablou gol
* var tablou = new Array() – echivalent cu var tablou[]
* var tablou=new Array(1,2,3) – echivalent cu var tablou = [1,2,3];
* Un tablou poate contine diferite tipuri de date



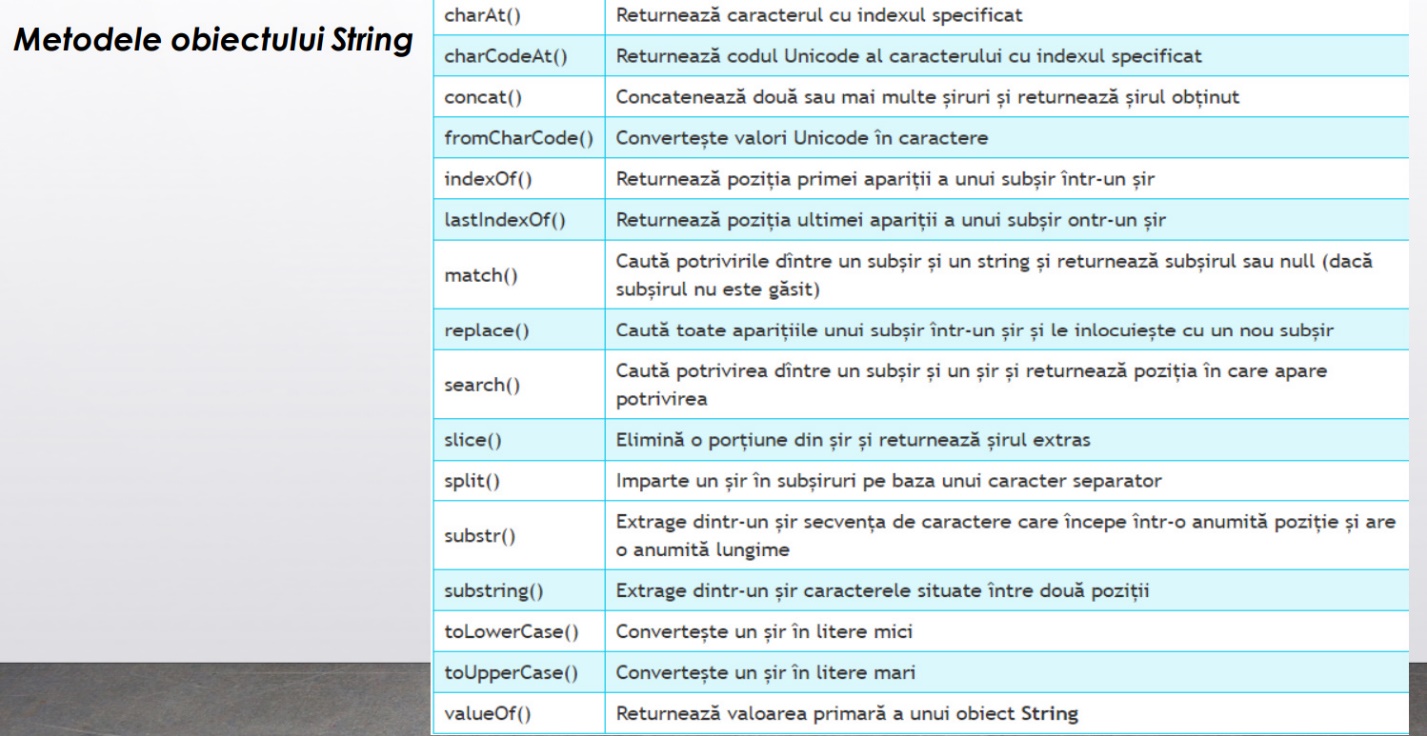
**for in**

In JavaScript exista for in

for(a in tablou)

**String**

In javascript la fel exista obiectul String



**Casete**

* alert(„text”); - apare o mini pagina de alert, cu butonul ok doar.

<!DOCTYPE html>

<html>

<title>Online HTML Editor</title>

<head>

<script>

function here(){

alert("Click ok to accept");

}

</script>

</head>

<body>

<button onclick="here()">Click here</button>

</body>

</html>

ATENTIE! TOT ce este intre <script> </script> la head se executa cand pagina porneste, evident cu exceptie functiilor.

* confirm(„Text de pe pagina”) – apare o pagina unde trebuie sa se apese OK sau cancel. Cand se apasa ok, se returneaza true, cand cancel, false.

<html>

<title>Online HTML Editor</title>

<script>

function apasa(){

var x = confirm("Apasa orice");

if(x==true){

document.write("Ati apasat ok");

}

else

document.write("Ati apasat cancel");

}

</script>

<head>

</head>

<body>

<input type="button" onclick="apasa()" value="Apasa"/>

</body>

</html>

* prompt(„text”) – afiseaza o fereastra inainte ca utilizatorul sa poata vizualiza pagina, si va cere introducerea unor date. Ok returneaza true, cancel false. Daca utilizatorul scrie ceva in casuta, se returneaza textul introdus.

<!DOCTYPE html>

<html>

<title>Online HTML Editor</title>

<script>

function func(){

var x = prompt("Introduceti numele:");

if(x!=null && x!=""){

document.write("Salut "+x);

}

}

</script>

<head>

</head>

<body>

<input type="button" onclick="func()" value="Apasa"/>

</body>

</html>

**Evenimente**



* Orice eveniment incepe cu prefixul **on**
* La functia apelata, pentru gestionarea evenimentului, se trimite argumentul **this** pentru a arata asupra carui element se aplica ea.

**Evenimente de mouse**

* onclick
* ondbclick
* onmousedown – cand este apasat clickul
* onmousemove – cand se misca cursorul in interiorul unui element
* onmouseover – cand se pune cursrul deasupra unui element
* onmouseout – cand cursorul iese din interiorul unui element
* inmouseup – cand se elibereaza clickul.
* onchange – cand se modifica starea butonului. De ex, cand se bifeaza un check box, si apoi se debifeaza, de fiecare data se apeleaza o metoda.

<html>

<title>Online HTML Editor</title>

<style>

.buton{

background-color: red;

}

</style>

<script>

function laClick(elem){

elem.style.background = 'blue';

elem.innerHTML = 'click';

}

function laHover(elem){

elem.style.background = 'green';

elem.innerHTML = 'hover';

}

</script>

<head>

</head>

<body>

<div class="buton" onclick='laClick(this)'

onmouseover='laHover(this)'>Click</input>

</body>

</html>

**Evenimente de tastatura**

* onkeydown – se apasa o tasta
* onkeypress – cand este apasata si eliberata o tasta. Daca apasam o tasta, intai se intampla evenimentul si apoi apare ce am scris noi.
* onkeyup – este eliberata o tasta

<input type="text" onkeydown="alert('eveniment la onkeydown')" />

<p><input type="text" onkeyup="alert('eveniment la onkeyup')" />

<p><input type="text" onkeypress="alert('eveniment la onkeypress')" />

**Evenimente de ferestre**

* onabort – cand se inrerupe incarcarea ferestrei
* onerror – cand o imagine nu se incarca corect sau alte tipuri de errori apar
* onload – cand se incarca fereastra
* onresize -
* onscroll -
* onunload – cand nu se incarca fereastra.
* oninput – cand se introduc caractere

**Evenimente provocate de formulare**

* onblur – cand cursorul e scos din casuta respectiva
* onchange – cand se schimba continutul din casuta
* onfocus – cand se pune cursorul in casuta
* onreset – cand se apasa butonul rese
* onselect – cand se selecteaza un text din camp
* onsubmit – cand se trimite formularul

<html>

<head>

<title> onBlur </title>

<script>

function verificare(form)

{ var x=0;

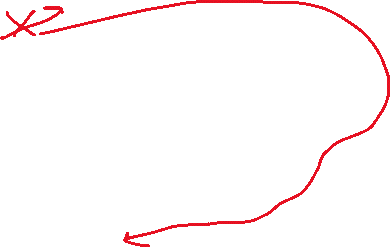
x=document.formular.date.value; y=isNaN(x);

if (y==true) alert("Introduceti un numar!");

else if (x<0)

{ alert("Va rog sa introduceti un numar pozitiv!"); } }

</script>



</head>

<body>

<h3>Introducetiun numar pozitiv:</h3>

<p>Pentru verificare, puteti introduce altceva

<br /> <form name="formular">

<input type="text" name="date" value="" size="20"

onBlur='verificare(this.form)'>

</form>

</body>

</html>

* elem.value – returneaza valoarea elementului
* document.getElementById(‚id’) – ia elementul dupa idul din tag
* elem.innerHTML – seteaza textul afisat de element

<html>

<title>Online HTML Editor</title>

<head>

<script>

function numara(elem){

var x = document.getElementById('show');

var y = elem.value;

x.innerHTML = '<p>Nr de caractere:'+y.length+ '</p>';

}

</script>

</head>

<body>

<input type="text" oninput='numara(this)'/>

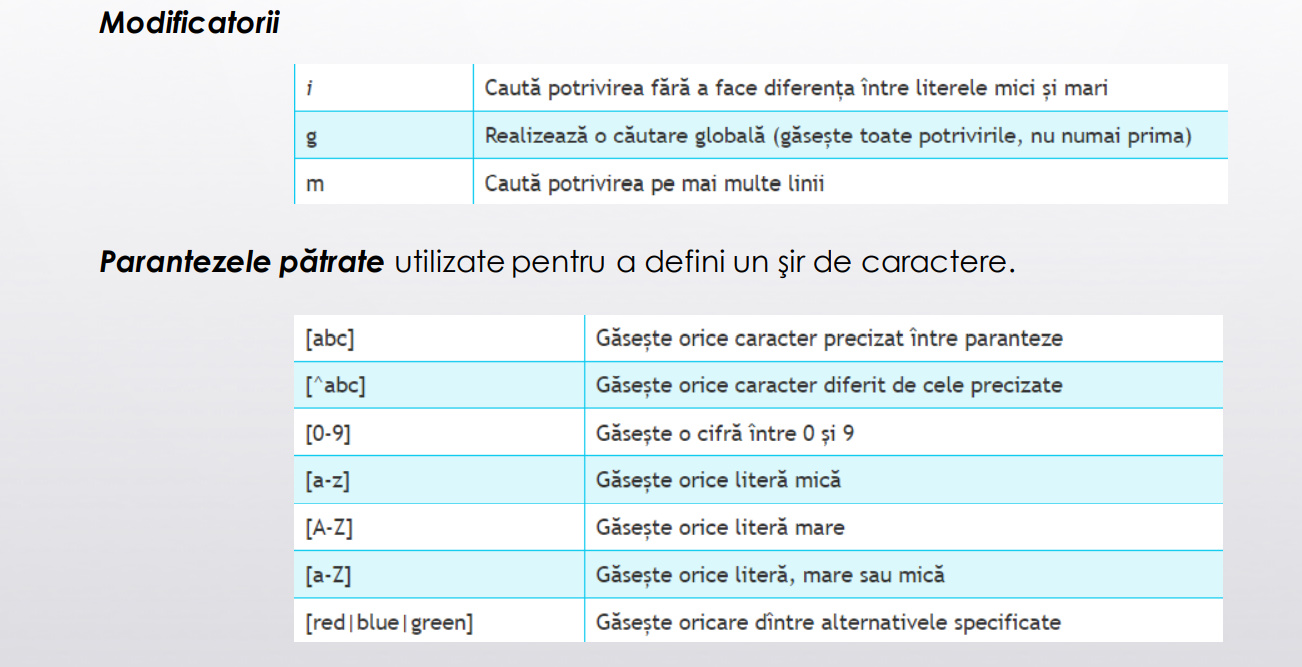
<h3 id="show">Nr de caractere:</h3>

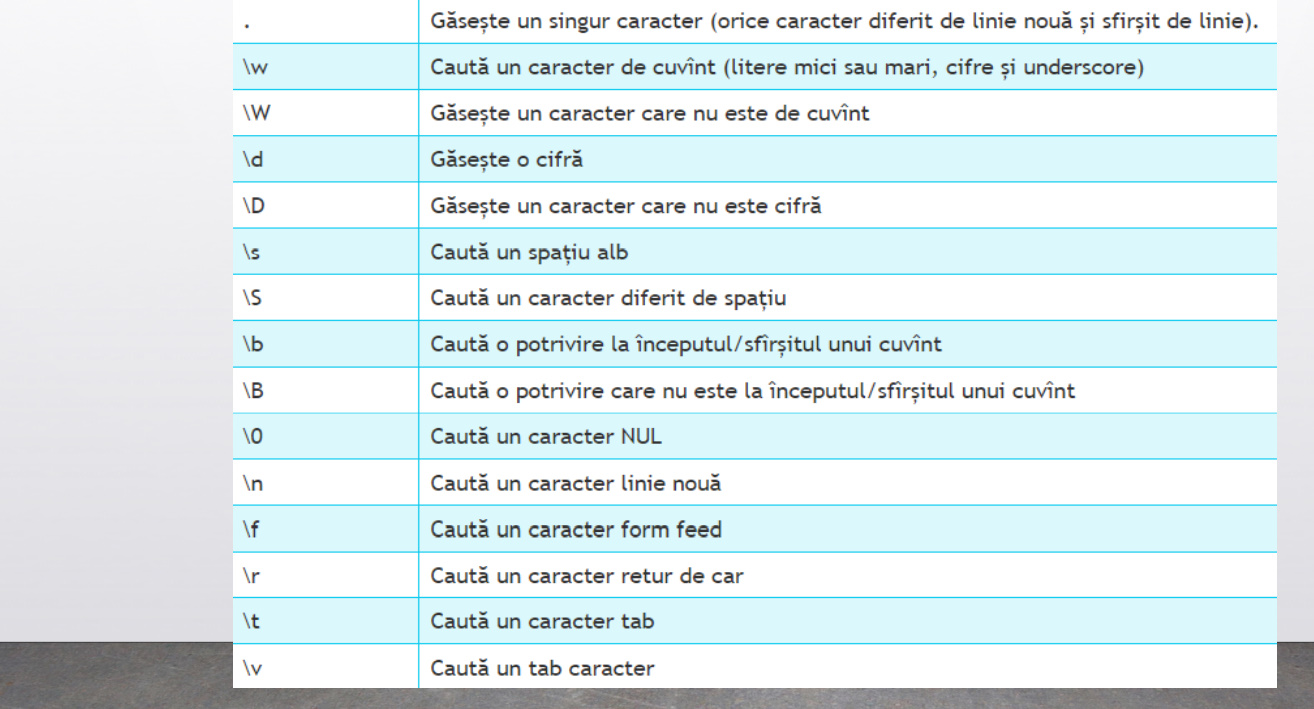
</body>

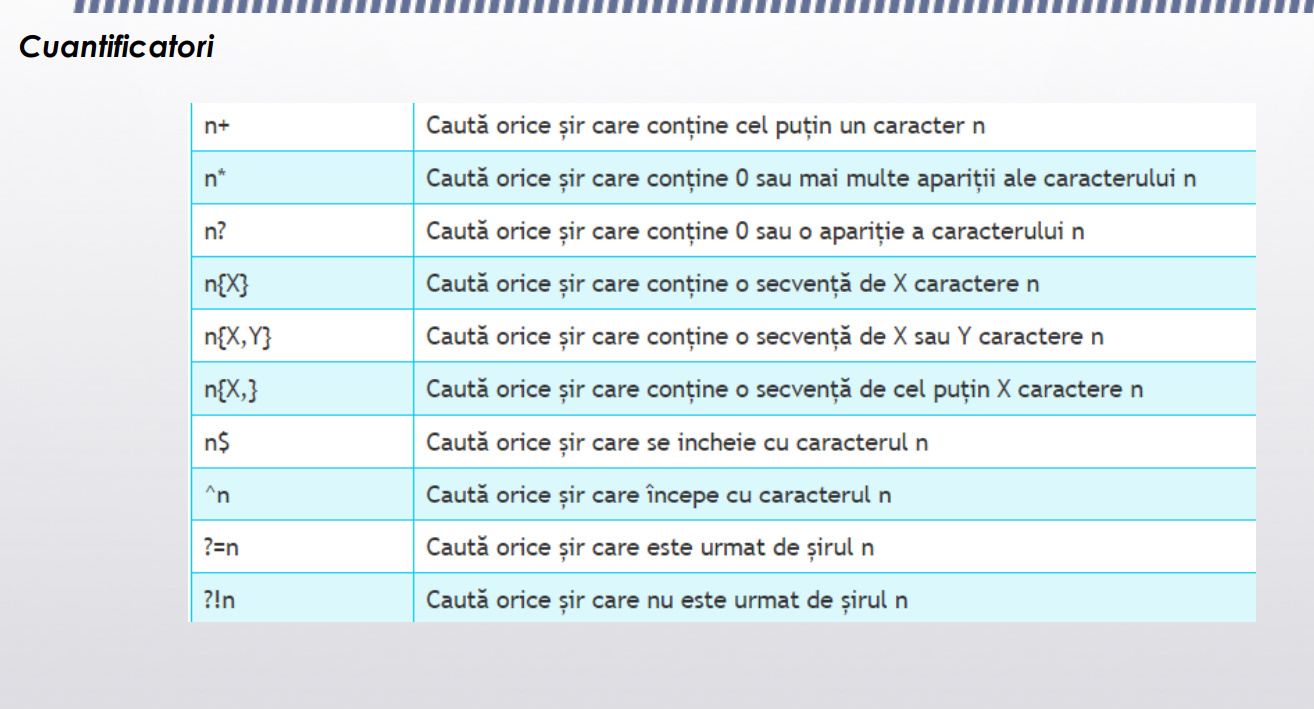
</html>

**RegExpression**

* vat txt = new RegExp(„pattern”);







**Metode**:

* .test(„text”) – returneaza true sau false, daca exista un sub sir gasit
* .exec(„text”) – returneaza un obiect cu substringurile gasite
* stringObj.search(regex) – returneaza locatiei primei aparitie a unui substring din stringObj ce corespunde la regex
* stringObj.match(regex) – returneaza substringurile gasite in array
* stringObj.replace(regex,sir) – inlocuieste sir unde gaseste substringuri ce corespuns cu regex.

**setInterval**

setInterval(function, {functiaMea()},timp in milisecunde) – asigura executarea repetata a functiei in intervalul stabilit