Introducere în limbajul JavaScript (II)

Limbajul JavaScript poate servi la:

- generarea paginilor Web personalizate şi modificarea dinamică a prezentării lor;
- realizarea calculelor matematice;
- validarea conţinutului unui formular;
- crearea animaţiilor personalizate;
- afişarea unor mesaje care defilează în bara de stare a navigatorului;
- afişarea unor mesaje într-o pagină Web sau într-o casetă de dialog;
- crearea unor butoane animate;
- identificarea navigatorului în care se afişează pagina Web
- executarea funcţiilor clasice ale unui limbaj de programare

- Utilizarea limbajului JavaScript se reduce în principal la două concepte de bază:

➢ sintaxa JavaScript

Sintaxa defineşte un ansamblu de reguli care trebuie respectate când se scrie cod JavaScript.

➢ DOM-ul (Document Object Model – modelul obiectelor documentului).

DOM-ul se referă la componentul paginii Web, obiectele care se pot accesa şi manipula cu ajutorul limbajului JavaScript.

Toate elementele dintr-o pagină sunt văzute de JavaScript ca fiind obiecte.
Obiectele sunt structuri compacte de date care pot să conțină mai multe proprietăți și funcții (denumite Metode).
Apelarea proprietăților și metodelor unui obiect se face cu operatorul punct (.)

Sintaxa

obiect.proprietate

obiect.metoda()

Obiecte

1 - Obiectul String
2 - Obiectul Array
3 - Obiectul Date
4 - Obiectul Math

- ## Obiectul Boolean

Obiectul **Boolean** este utilizat pentru a converti o valoare ne-booleană într-o valoare booleană (cu valoarea **true** sau **false**).

Crearea unui obiect boolean poate fi realizată ca în secvenţa de cod următoare:

var sem=new Boolean();

- ## Obiectul RegExp

Obiectul **RegExp** este folosit pentru a realiza căutări şi înlocuiri într-un text. **RegExp** - *expresie regulată*

Un obiect **RegExp** poate fi definit prin:

var txt=new RegExp(pattern,modifiers);

sau

var txt=/pattern/modifiers;

unde

pattern specifică modelul căutat

modifiers specifică dacă căutarea este globală, case-senzitivă etc.

Proprietăţile obiectului RegExp

Proprietate	Descriere
global	Specifică dacă modificatorul **"g"** este setat
ignoreCase	Specifică dacă modificatorul **"i"** este setat
lastIndex	Specifică indexul de la care începe căutarea următoarei potriviri
multiline	Specifică dacă modificatorul **"m"** este setat
source	Textul din modelul **RegExp**

- **Metodele obiectului RegExp**

Obiectul **RegExp** are trei metode: **test()**, **exec()** şi **compile()**.

Metoda test()

Caută într-un şir un model şi returnează **true** sau **false**.

Metoda exec()

Caută în text un model şi returnează modelul, dacă acesta este găsit sau valoarea **null**, dacă modelul nu apare în text.

Metoda compile()

Este utilizată pentru a modifica conţinutul obiectului **RegExp**.

Metoda poate schimba modelul căutat şi poate adăuga sau elimina al doilea parametru.

Reguli Expresii_regulate.pdf

```
<html>  <body>
<script>
var m1=new RegExp("e");
var str= m1.test("Cele mai frumoase carti le pastrez in amintire");
document.write(str);
</script>
</body>  </html>
```

Programul va afişa rezultatul: **true**

```
<html>  <body>
<script>
var m1=new RegExp("e");
document.write(m1.test("Cele mai frumoase carti"));
m1.compile("d");
document.write(m1.test("Cele mai frumoase carti"));
</script>
</body>  </html>
```

Programul va afişa rezultatul: truefalse

```
<html>  <body>
<script>
var str="Tu creezi pagini web interesante!";
var m1=/e+/g;
do
{ result=m1.exec(str);
document.write(result); document.write("  ");
}
while(result!=null)
</script>
</body>  </html>
```

Programul va determina toate secvenţele din text în care caracterul **„e”** apare cel puţin o dată (în poziţii consecutive).

Rezultatul afişat de program este:

ee e e e e null

HTML DOM

- DOM
  - **Document Object Model**
  - standard W3C (World Wide Web Consortium)
  - permite programelor și scripturilor accesarea dinamică a documentelor, actualizarea conținutului, structurii și stilurilor acestora.

La încărcarea unei pagini web, browser-ul construiește modelul DOM al acesteia. Modelul HTML DOM este construit ca un arbore de obiecte.

HTML DOM

Pe baza modelului DOM, JavaScript poate genera conținut HTML dinamic și de asemenea:

- JS poate schimba toate elementele HTML din pagină
- JS poate schimba toate atributele HTML în pagină
- JS poate schimba toate stilurile CSS în pagină
- JS poate elimina elemente HTML și atribute existente
- JS poate adăuga noi elemente HTML și atribute
- JS poate reacționa la toate evenimentele HTML existente în pagină
- JS poate genera noi evenimente HTML în pagină

HTML DOM

- Modificarea conţinutului unui element HTML

document.getElementById(“*id*”).innerHTML = *continut_nou_html*

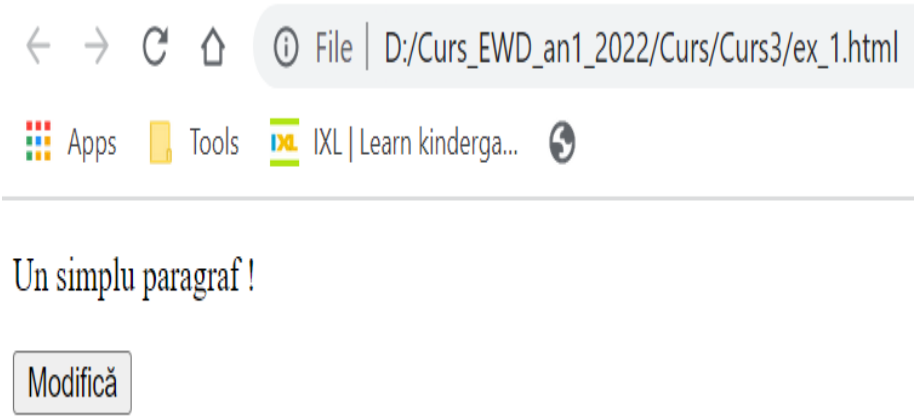
- Modificarea atributelor unui element HTML

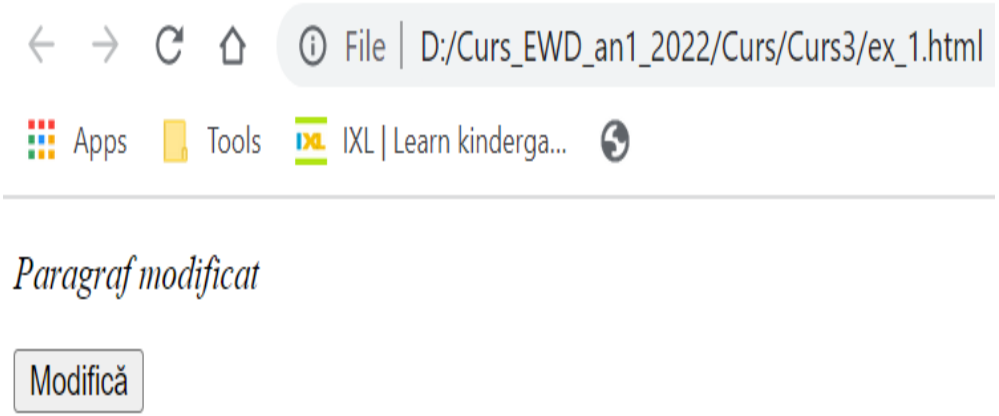
document.getElementById(“*id*”).*nume_atribut*=*noul_continut*

Modificarea conţinutului unui element HTML

document.getElementById. innerHTML= *continut_nou_html*

```
<html> <head>
<title>Modifică conținut</title>
</head>
<body >
<p id="p1">Un simplu paragraf!
<button
onclick="modifica()">Modifică
</button>
<script>
function modifica(){
      var elem=
document.getElementById("p1");
elem.innerHTML="<em>
Paragraf modificat </em>" ;
}
</script>
</body> </html>
```

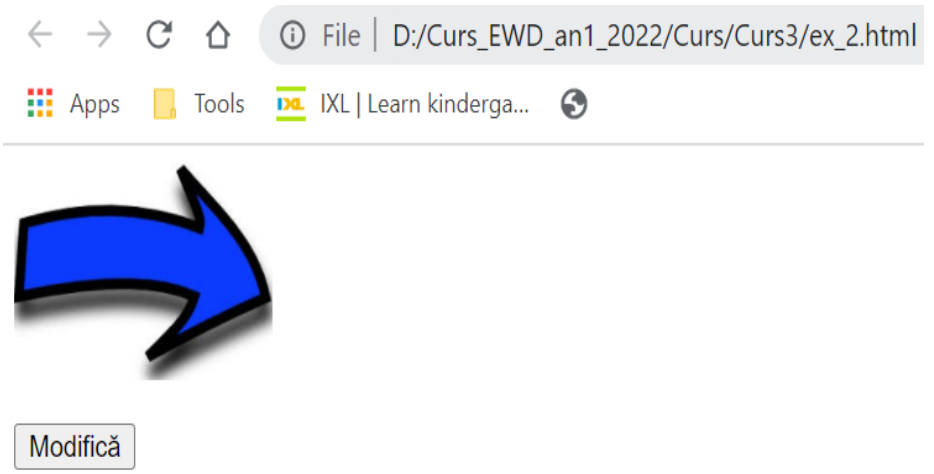


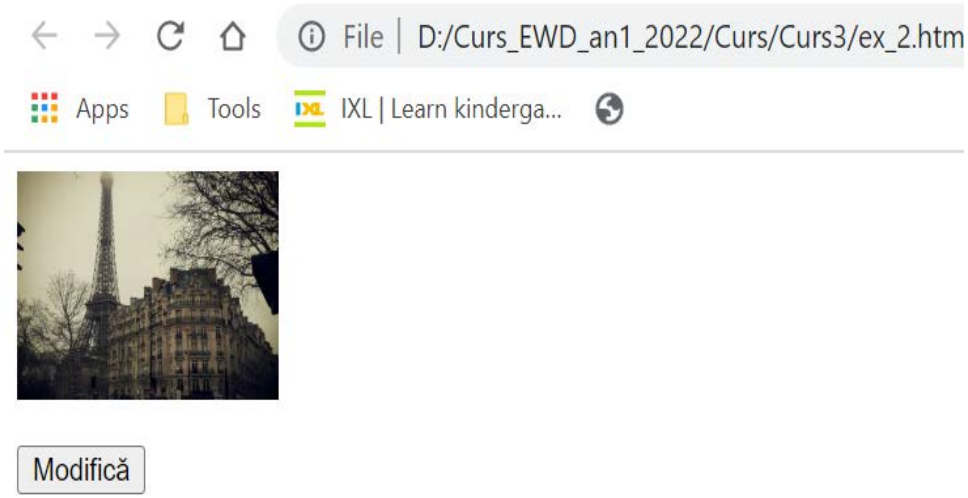


Modificarea atributelor unui element HTML

document.getElementById.nume_atribut=*noul_continut*

```
<html> <head>
<title>Modifica atribut</title> </head>
<body >
<img id="img1" src="images.jpg"
height="100" />
<p><button
onclick="modifica()">Modifica
</button></p>
<script>
function modifica(){
var elem=
document.getElementById("img1");
elem.src="paris.jpg" ;
}
</script>
</body> </html>
```





Accesul la elemente HTML folosind JS

Metodă

document.getElementById("*id*")

document.getElementsByName("nume")

document.getElementsByTagName("*nume*")

document.getElementsByClassName("*nume*")

document.getElementById("*id*")

- Metoda **returnează** un element HTML ce are valoarea atributului "id" egală cu valoarea "*id*" specificată.
- Este printre cele mai folosite metode de a manipula elementele unui document HTML.
- Id-ul trebuie să fie unic. (dacă sunt mai multe id-uri cu aceași valoare, metoda va accesa primul element cu acest id)
- Dacă nu exista id-ul, va returna “null”.

document.getElementsByName("nume")

- Metoda **returnează** o colecţie de elemente, dintr-un document HTML, ce au valoarea atributului "name" egală cu valoarea "nume" specificată
- Colecţia de obiecte poate fi accesată prin index, numărătoarea începând de la 0.
- Proprietatea “length” returnează numărul de elemente din colecţie.
- Ultimul element are indexul: **length-1**

- Exemplu

```
...
<p name="paragraf">para1</p>
<p name="paragraf">para2</p>
...
```

var x = document.getElementsByName("paragraf");

x este o listă cu elemente HTML ce au valoarea atributului name=“paragraf”’

```
for (i = 0; i < x.length; i++) {
        window.alert(x[i].innerHTML);
//afişează de două ori alert cu ceea ce contine fiecare paragraf
  }
```

document.getElementsByTagName("*nume*")

- Metoda **returnează** o colecţie de elemente, dintr-un document HTML, ce au denumirea etichetei egală cu valoarea "nume" specificată
- Accesarea datelor se face în mod asemănător cu cele prezentate la metoda document.getElementsByName()

Ex.

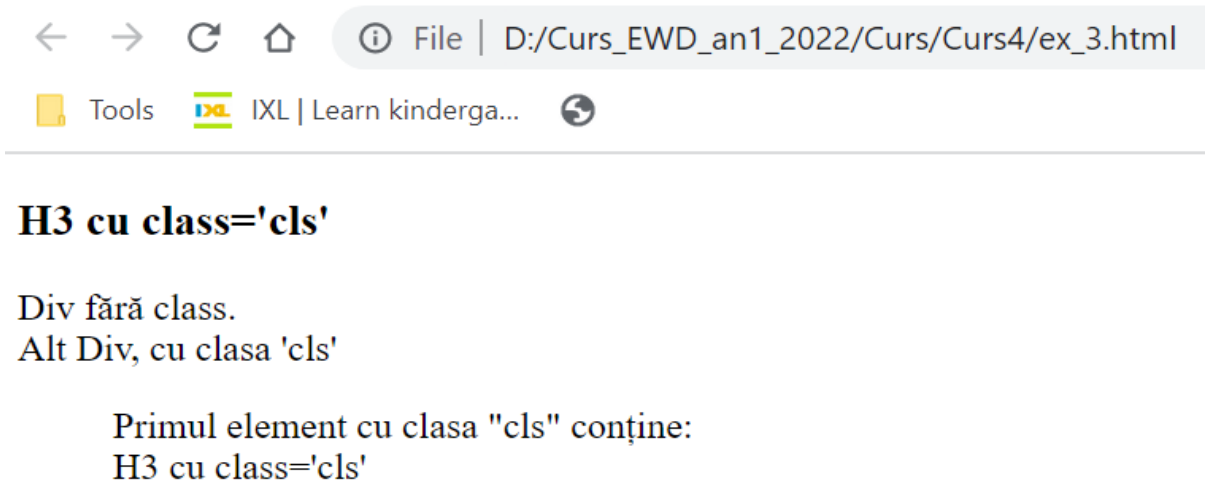
```
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>

...
var x = document.getElementsByTagName("li");
// accesează toate elementele HTML cu eticheta "li" (list item). Astfel, x.length va avea valoarea 3. Accesul la elementele colecţiei se face prin indexare.
```

document.getElementsByClassName("*nume*")

- Metoda **returnează** o colecţie de elemente, dintr-un document HTML, ce au valoarea atributului "class" egală cu valoarea "nume" specificată

```
<!doctype html> <html> <head>
<meta charset="utf-8" /> </head>
<body>
<h3 class='cls'>H3 cu class='cls'</h3>
<div>Div fără class.</div>
<div class='cls'>Alt Div, cu clasa 'cls'</div>
<blockquote id='resp'>getElementsByClassName()</blockquote>
<script>
//array cu elementele care au clasa .cls
var ecls = document.getElementsByClassName('cls');
document.getElementById('resp').innerHTML ='Primul element cu clasa "cls" conține:<br>'+ ecls[0].innerHTML;
</script>
</body> </html>
```



Schimbarea de elemente HTML

Metodă	Descriere
element .innerHTML = *conţinut HTML*	Schimbarea conținutului HTML al unui element
element . *attribute* = *valoare nouă*	Modificarea valorii *attribute* al unui element HTML
element .setAttribute *(attribute, value)*	Modificarea valorii *attribute* al unui element HTML cu *value*
element .style. *property* = *new style*	Schimbarea stilului unui element HTML

Schimbarea de elemente HTML

```
<p id="demo">Continut HTML initial</p>
document.getElementById("demo").innerHTML = "Continut HTML modif";
```

```
<a id=“id_ancora" href="http://www.microsoft.com">Microsoft</a>
 document.getElementById("id_ancora").href = "https://www.uvt.ro";
```

```
<a id=“id_ancora" href="http://www.microsoft.com">Microsoft</a>
document.getElementById(“id_ancora").setAttribute("href",
"https://www.w3schools.com");
```

```
<a id=“id_ancora" href="http://www.microsoft.com">Microsoft</a>
document.getElementById(“id_ancora”).style.color = "red";
      //efect: ”Microsoft” se va colora in rosu
```

Adăugarea și ștergerea elementelor

Metodă	Descriere
document. createElement(denumire_*element*)	Crearea unui element HTML
document. removeChild(*element*)	Eliminarea unui element HTML
document. appendChild(*element*)	Adăugarea un element HTML
document. replaceChild(*element*)	Înlocuirea unui element HTML
document. write(*text*)	Scrierea în fluxul de ieșire HTML

- Exemplu.

```
var btn =document.createElement("BUTTON");
```

Creeaza buton

```
btn.innerHTML = "CLICK ";
```

Are denumirea CLICK

```
document.body.appendChild(btn);
```

Inserează butonul creat in secţiunea “body”

Adăugarea de Evenimente

Metodă	Descriere
document. getElementById(*id*).onclick = function() { *code* }	Adăugarea de cod de tratare a evenimentului la un eveniment *onclick*

Un **eveniment** este o acţiune care se produce în raport cu un element (fereastră, document, buton, etc.) el poate fi detectat şi prelucrat de către un script care va declanşa o acţiune.
Reacţia la un eveniment este cunoscută sub numele de prelucrarea evenimentului, iar codul JavaScript corespunzător este cunoscut sub numele de **gestionar de evenimente**.
Un gestionar de evenimente este o metodă care va fi apelată în mod automat de către browser ori de câte ori va surveni un eveniment particular.
Gestionarii de evenimente sunt funcţii JavaScript.

JavaScript HTML DOM
(Modelul Obiectului Document)

- HTML DOM este un model de obiecte și o interfață de programare standard pentru HTML

Definește:

- Elementele HTML ca **obiecte**

Cuprinde:

- **Proprietățile** tuturor elementelor HTML
- **Metodele** pentru a avea acces la toate elementele HTML
- **Evenimentele** pentru toate elementele HTML

- **HTML DOM este un standard pentru a obține, a schimba, a adăuga sau a șterge elemente HTML.**

Obiectul “document”

- **Obiectul document** reprezintă pagina web, în el se găsesc elementele (tag-urile HTML), atributele și conținutul dintr-o pagină web. Acesta conține proprietăți și metode prin care se pot adăuga, accesa, edita, șterge in mod dinamic elemente și conținut in pagină.

document*.*nume_proprietate/metoda

sau

window.document*.*nume_proprietate/metoda

Toate proprietăţile și metodele HTML DOM pentru obiectul Document sunt disponibile la adresa:

https://www.w3schools.com/jsref/dom_obj_document.asp

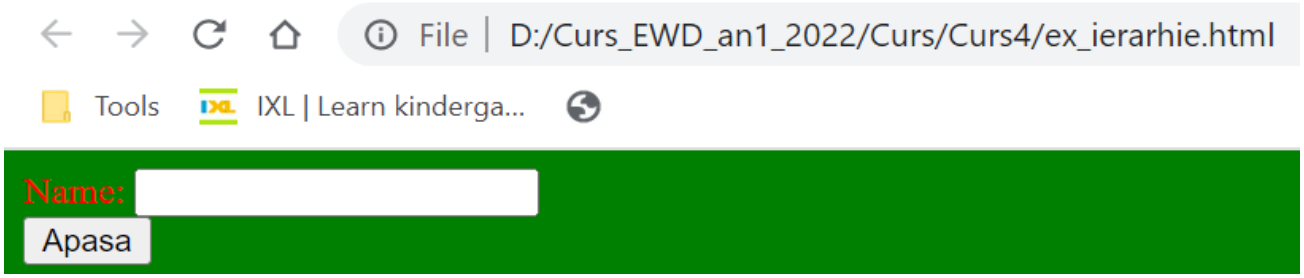
- **Obiectul document şi formulare**

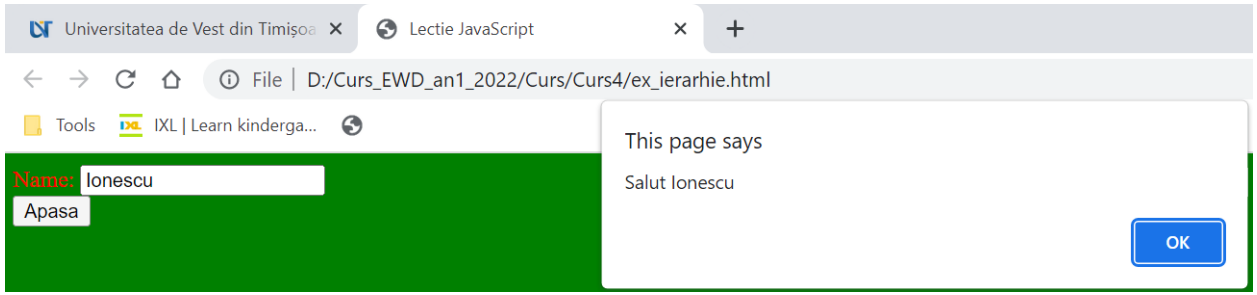
Formularele pot fi considerate şi tratate ca obiecte dintr-un document HTML, acestea fiind sub-obiecte ale obiectului "document“.

- Obiectul **form** reprezintă elementele

HTML <form>

```
<!DOCTYPE html>
<html> <head>
<script>
function afisare()
{ nume = document.nume_form.nume_camp.value;
   return nume
}
</script> </head>
<body >
<script>
    document.bgColor = 'green'
    document.fgColor = 'red'
    document.title = "Lectie JavaScript"
</script>
<form name="nume_form">
Name: <input type="text" name="nume_camp" value=""><br>
<input type="button" value="Apasa" name="Buton" onClick="alert('Salut '
+afisare())">
</form> </html>
```





Obiectul părinte:	**Document**
Subobiecte:	button, checkbox, fileupload, hidden, input, password, select, option, radio, reset, text, textarea, submit
Proprietăți:	acceptCharset, action, elements[], encoding, enctype, length, method, name, target
Metode:	reset(), submit(), tags()
Gestionarii de evenimente:	onContextMenu, onControlSelect, onCopy, onCut, onDblClick, onDeActivate, onDrag, onDragend, onDragenter, onDragLeave, onDragOver, onDragStart, onFocusOut, onHelp, onKeyDown, onKeyPress, onKeyUp, onMouseDown, onMouseEnter, onMouseLeave, onMouseMove, onMouseOut, onMouseOver, onMouseUp, onMousewheel, onMove, onMoveend, onMovestart, onPaste, onPropertyChange, onReset, onResize, onResizeEnd, onResizeStart, onSelectStart, onSubmit

Preluare element <form> în JavaScript

- Se folosesc metodele

getElementById(), **querySelector()** sau proprietatea: **document.forms**

- Dacă elementul <form> are un ID, se poate accesa prin

document.getElementById('id_form')
document.querySelector('#id_form')
document.forms['id_form']

- Dacă elementul <form> are un atribut ***name***, se poate accesa prin

document.forms['name_form']

- Dacă elementul <form> are o clasă css (atribut ***class***), se poate accesa prin

document.querySelector('form.class_form')

In afară de proprietățile și metodele specifice elementelor html, obiectul **form** conține proprietăți și metode proprii.

Accesare elemente din form

- elementele HTML dintr-un <form> pot fi preluate ca obiecte in JS prin:

metodele **getElementById(), querySelector(),**
formula: **eform['elm_name']** sau proprietatea: **eform.elements[index]**

Dacă elementul din <form> are un ID, se poate accesa prin

document.getElementById('id_elm')
document.querySelector('#id_elm')

Dacă elementul din <form> are un atribut ***name***, se poate accesa prin

eform['elm_name']

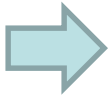
unde ***eform*** reprezintă obiectul formularului, iar '***elm_name***' numele elementului din acel formular

Dacă elementul din <form> are o clasă css (atribut ***class***), se poate accesa prin

document.querySelector('#id_form .class_elm')

eform.querySelector('.class_elm')

```
<body>
<h4>Accesare input dupa class</h4>
<form id='frm1' method='post'>
Text: <input type='text' value='some-val' class='ftxt'/>
</form>
<p>La clic pe buton, acceseaza dupa 'class' caseta text din Form si afiseaza la #resp valoarea de la 'value'.</p>
<button id='btn1'>Click</button>
<blockquote id='resp'>#resp</blockquote>
<script>
document.getElementById('btn1').addEventListener('click', (ev)=>{
 var ftxt = document.querySelector('#frm1 .ftxt');
 document.getElementById('resp').innerHTML = ftxt.value;
});
</script></body>
```



Accesare input dupa class

Text: some-val

La clic pe buton, acceseaza dupa 'class' caseta text din Form si afiseaza la #resp valoarea de la 'value'.

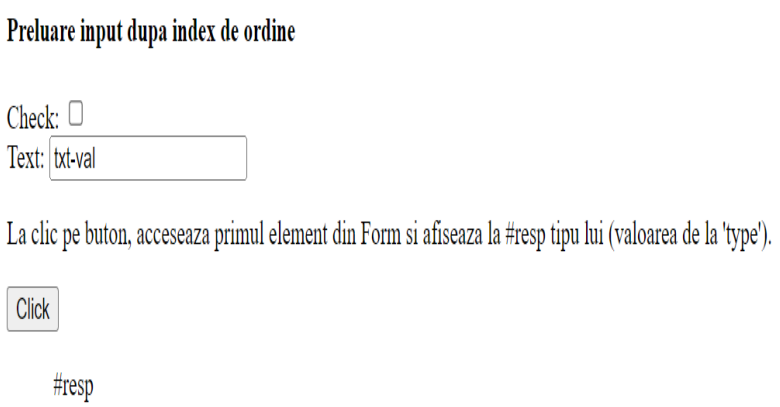
Click

#resp

Pentru accesarea unui element din <form> in funcție de numărul de ordine (index care incepe de la 0)

eform[index]
eform.elements[index]
eform.item(index)

unde '***eform***' reprezintă formularul in care este acel element,
iar ***index*** e numărul de ordine (in funcție de ordinea adăugării elementelor în <form>)




```
<body>
<h4>Preluare input dupa index de ordine</h4>
<form id='frm1' method='post'>
Check: <input type='checkbox' value='some-val'/><br>
Text: <input type='text' value='txt-val'/>
</form>
<p>La clic pe buton, acceseaza primul element din Form si afiseaza la #resp tipul lui (valoarea de la 'type').</p>
<button id='btn1'>Click</button>
<blockquote id='resp'>#resp</blockquote>
<script>
document.getElementById('btn1').addEventListener('click', (ev)=>{
 var ftxt = document.getElementById('frm1');
document.getElementById('resp').innerHTML = ftxt[0].type;
});
</script></body>
```

Evenimente JavaScript

- sunt acțiuni provocate de cele mai multe ori de vizitatorul paginii.
- JavaScript poate reacționa la unele evenimente. Aceasta se poate realiza cu ajutorul "**event-handlers**" (manageri de evenimente sau gestionari de evenimente).

Handlerele de evenimente se adaugă ca atribute ale etichetelor HTML.

<tag event='cod-JS'>

- **Evenimentele JavaScript**
  Evenimente_JS.pdf
- **Gestionarii de evenimente**
  Gestionari_ev.pdf

https://www.w3schools.com/jsref/dom_obj_event.asp

Pentru a nu amesteca tag-urile HTML cu coduri JS, evenimentele se pot adăuga in codul JavaScript, asociate la elementul HTML preluat in JS; folosind sintaxa.

```
function funcName(ev){
// ev reprezinta obiectul cu evenimentul declansat
//codul functiei
 }
elm.event = funcName;
```

sau cu funcție anonimă

```
elm.event = function(ev){
// ev reprezinta obiectul cu evenimentul declansat
//codul functiei
};
```

ex_5.html

Obiectul image

- In JavaScript toate imaginile sunt reprezentate intr-o matrice numită **images (document.images[0], document.images[1],…)**

Declararea imaginii

- ****
- folosirea proprietății **"getElementById("id_element"),** unde **"id_element"** este id-ul imaginii dat cu atributul id="... "

Ex.

- **document.nume_img.height** sau
- **document.getElementById("id_img").height**

Obiectul "image" are o singură metodă:

- **handleEvent()** - reprezintă evenimentul specificat pentru executarea unei anumite acțiuni

Proprietățile obiectului image propr_image.pdf

Border - poate fi doar citită- este lățimea marginii din jurul imaginii specificată în pixeli

Alt - specifică textul care va fi afișat in locul imaginii (dacă nu poate fi vizualizată de browser)

Align - unde să fie plasată imaginea

Complete - poate fi doar citită- este o valoare booleană care arată dacă imaginea a fost descarcată complet

Height – inălțimea in pixeli a imaginii

Hspace – spațiul din dreapta și stânga imaginii (in pixeli)

Lowsrc - specifică un URL al unei imagini ce va fi afișată la o rezoluție scăzută

Name - se folosește pentru a da nume unei imagini

Src - specifică URL-ul (adresa și numele) imaginii

Vspace - spatiul dintre partea de sus și de jos a imaginii

Width – lățimea in pixeli a imaginii

- Exemplu de pagina HTML cu două link-uri care, printr-un script JS schimbă afișarea mai multor imagini in același loc. Imaginile sunt declarate și stocate intr-o variabila tablou "imagini". Pentru schimbarea imaginilor se folosesc 2 funcții: "gonext" și "goback”

ex_imag.html

```
<!DOCTYPE html><html>
<head><meta charset="utf-8">
<script>
imagini = new
Array("img/imag1.jpg","img/imag2.jpg","im
g/imag3.jpg","img/imag4.jpg");
nr=0;
lung=imagini.length;
document.write(lung+" imagini: <br />");
function goback() {
   if ((document.images) && (nr > 0)) {
    nr--;
    document.imgs.src=imagini[nr];
   }
 }
function gonext() {
   if ((document.images) && (nr < (lung-1)))
{      nr++;
       document.imgs.src=imagini[nr];
     }
   }
</script> </head>
<body>
<h4 align=center>Imagini<br>
<img src="img/imag1.jpg" name="imgs"
width="200" height="200"><br>
<a href="javascript:goback()"><<-
Precedenta</a> |
<a href="javascript:gonext()">Următoarea
->></a>
</h4>
</body></html>
```

- Preîncărcare imagini

var obimg = new Image();
obimg.src ='adresa_img.jpg'; //preincarca imaginea

Acest cod incarcă imaginea in JavaScript, fiind valabilă spre afișare rapidă din obiectul 'obimg‘.

(Când mouse-ul intră pe suprafața unei imagini se schimbă imaginea, când mouse-ul iese din suprafața ei se adaugă o altă imagine)

ex_6.html

Validarea formularelor

– validarea datelor introduse de utilizator. Se pot valida câmpuri de intrare, grupuri de câmpuri sau întregul formular, utilizând gestionari de evenimente şi funcţii JavaScript.
– construirea formularelor interactive, în care o parte sau întreaga prelucrare are loc pe parte de client.
– a testa conformitatea datelor introduse de utilizator cu politicile de procedură impuse (exemplu de politică de procedură: data de livrare a unei comenzi nu poate fi în ziua de sâmbătă/duminică a săptămânii).
– a testa prezenţa datelor în câmpurile obligatorii ale unui formular (un câmp este prezent dacă nu este vid).

- Scrieţi un program JavaScript care verifică dacă conţinutul celor trei câmpuri ale unui formular: nume, prenume, codcard este vid. În caz de eroare afişaţi unul din mesajele de mai jos:
  – „Aţi uitat să introduceţi numele dumneavoastră!”
  – „Aţi uitat să introduceţi prenumele dumneavoastră!”
  – „Aţi uitat să introduceţi numărul dumneavoastră de card!”

- ex_7.html

```
<!DOCTYPE html>
<html> <head><meta charset="utf-8">
<title> Validare formular</title>
<script>
  function verif() {
    if (document.comanda.nume.value=="")
        alert("Ați uitat să introduceti numele dumneavoastră!");
      else if (document.comanda.prenume.value=="")
           alert("Ați uitat să introduceți prenumele dumneavoastră!");
                          else if (document.comanda.card.value=="")
                 alert("Ați uitat să introduceți numărul dumneavoastră de card!");
                     else return true;
     return false;
    }
</script>
</head>
<body>
<h3> Validare formular</h3>
<form name="comanda" action=" " method="post" onSubmit="return verif()">
  <table border="1">
    <tr><td> Nume: </td>
        <td> <input type="text" name="nume"></td>
        <td> Prenume: </td>
        <td> <input type="text" name="prenume"></td></tr>
    <tr><td> Numar card: </td>
        <td> <input type="text" name="card"></td>
        <td> <input type="submit" value="OK"
                 onSubmit="return verif()"></td>
        <td> <input type="reset" value="resetare"></td></tr>
   </table>
</form> </body>
</html>
```

jQuery

- *jQuery* este o librărie de funcții JavaScript creată de John Resig (**cea mai utilizată și stabilă librărie JavaScript**)

- jQuery este centrat pe lucrul și manipularea elementelor HTML și CSS in pagina web, concepută pentru a ușura și îmbunătăți procese precum traversarea arborelui DOM în HTML
- se pot crea cereri Ajax pentru transmiterea de date la server, funcții pentru lucru cu obiecte, array și evenimente.
  Aproape toate scripturile făcute cu jQuery funcționează la fel in principalele navigatoare web.

– Sloganul jQuery este „**Write less, do more**” deci **scrierea** este cu mult **comprimată** in comparație cu utilizarea limbajului de programare JavaScript pur;

- librăria jQuery este una dintre cele mai active in ceea ce privește producerea de cod nou și de plugin-uri
- in industria IT utilizează jQuery: IBM, Netflix, Nokia, Wikipedia, Google și Microsoft;

– Librăria jQuery are dimensiuni reduse ~ 96 KB pentru întreaga librărie în varianta comprimată.

Pentru utilizarea jQuery avem două opțiuni:

1. Descărcarea librăriei pe calculator și includerea în pagina html

Se descarcă ultima versiune de la pagina Download jQuery. Se salvează librăria jQuery pe server, intr-un fișier cu extensia ".js", apoi se include in documentul HTML, folosind următoarea sintaxă (*in sectiunea HEAD sau BODY*):

<script src="jquery_file.js"></script> (jquery-3.6.0.js)

2. Includerea librăriei prin legătura la un server CDN (Content Distribution Network).

Această opțiune ne dă posibilitatea de a utiliza librăria prin rețeaua **CDN – Content Delivery Network** pusă la dispoziție atât de jQuery, cât și de Google sau Microsoft.

Google găzduieste pe rețeaua sa numeroase librării open source, printre care și multe librării JavaScript și jQuery. **Avantajele utilizării librăriei jQuery prin CDN-ul Google:** viteza și utilizarea celei mai noi versiuni încărcate de Google. (mulți vizitatori au descărcat deja librăria de la CDN-ul Google prin vizualizarea altor site-uri web și deci viteza de încărcare a site-ului va fi astfel favorizată.)

<script src="""https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"**>**
</script>

- In jQuery pentru a chema funcțiile disponibile in librărie utilizăm ca o prescurtare simbolul $.
- Instructiunea jQuery = $.
- In interiorul codului nostru jQuery putem folosi atât funcția completă jQuery, cât și prescurtarea acesteia, simbolul $.

Semantica instrucțiunilor jQuery

Orice comandă jQuery este compusă din 4 elemente:

- începe cu **jQuery** sau cu ajutorul alias-ului **$**;
- **selectorii** – pentru indicarea elementelor din pagina web asupra cărora se aplică funcția;
- **acțiunile** sau metoda – acțiunea care se aplică asupra selectorilor specificați;
- **parametrii** – indicarea metodei exacte pentru aplicarea acțiunii

Rezultatul final este că textele h1 sunt setate pe culoarea roșie.

- Pentru a putea interacționa cu elementele HTML in pagină, instrucțiunile script-ului trebuie executate după încărcarea paginii, astfel tot codul jQuery se scrie in interiorul unei funcții speciale "*document ready*", aceasta execută codul din ea după incărcarea paginii.

……

```
<script src="JS/jquery-3.6.0.js"></script>
<script>
$(document).ready(function() {
// tot codul jQuery se scrie aici
}) ;
</script>
</body>
```

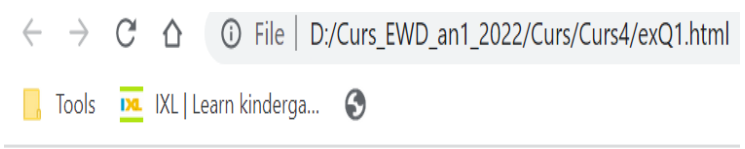
Obs. Tag-ul folosit și in acest caz este <script>.

Regulă: poziționarea acestuia inainte de inchiderea tag-ului </body>

```
$(document).ready(function()
{
$("h1").css("color", "red");
});
```

sau

```
$(function() {
$("h1").css("color", "red");
});
```



The first title is selected.

The second title is selected.

The third title is selected.

```
<!DOCTYPE html>
<html>
<head> <meta charset="utf-8" />
<title>jQuery Ex1</title>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"> </script>
 <script>
 $(function() {
 $('h1').css('color', 'red');
 });
 </script>
 </head>
<body>
<h1>The first title is selected.</h1>
<h1>The second title is selected.</h1>
<h1>The third title is selected.</h1>
</body>  </html>
```

exQ1.html

- Elementele HTML se selectează cu jQuery prin adăugarea lor in **$(" ")** sau ***jQuery(" ")*** (se pot folosi ghilimele duble sau simple).

- Adăugarea unui selector jQuery în $(), va returna un obiect jQuery ce conține un set cu elementele selectate, potrivite cu acel selector.

- Cu selectoarele jQuery, se pot găsi sau selecta elemente HTML pe baza id-ului, claselor, atributelor, tipurilor dintr-un DOM.

Selector	Descriere
Nume:	Selectează toate elementele care se potrivesc cu numele elementului dat.
#id:	Selectează un singur element care se potrivește cu id-ul dat.
.class:	Selectează toate elementele care se potrivesc cu clasa dată.
(*)	Selectează toate elementele disponibile într-un DOM.
Un șir de selectori A,B,C	Selectează rezultatele combinate ale selectorilor specificați A, B și C.

$('*') - selectează toate elementele.
$('div') - selectează toate tag-urile .
$('#un_id') - selectează un tag HTML cu id="un_id".
$('.a_class') - selectează toate tag-urile HTML cu class="a_class".
$('p#un_id') - selectează tagul <p> cu id="un_id".
$('li.a_class') - toate tag-urile <li> cu class="a_class".
$('li a') - toate tag-urile <a> din elementele <li>.
$('div a.a_class') - tag-urile <a> cu class="a_class", care sunt adăugate in DIV-uri.
$('div.a_class p span') - toate tag-urile din <p>-uri care sunt adăugate in -uri cu class="a_class".
$('a:first') - selectează primul tag <a>.
$('h3:last') - selectează ultimul <h3> din pagină.
$('input[type=text]') - selectează elementele input care au tipul (type) specificat la text.
$('p:odd') - selectează toate paragrafele cu număr de ordine impar.
$('p:even') - selectează toate paragrafele cu număr de ordine par.
$('li:first-child') - selectează primul <li> din fiecare listă cu tag-uri <li>.

jQuery are selectori proprii:

$(':button') - selectează elementele de tip buton (input sau button)
$(':radio') - selectează butoanele tip radio
$(':checkbox') - selectează checkbox
$(':checked') - selectează elementele checkbox sau radio care sunt selectate
$(':header') - selectează elementele de tip Header (h1, h2, h3, etc.)
$(':contains("String")') - selectează elementele care conțin textul specificat la "String"

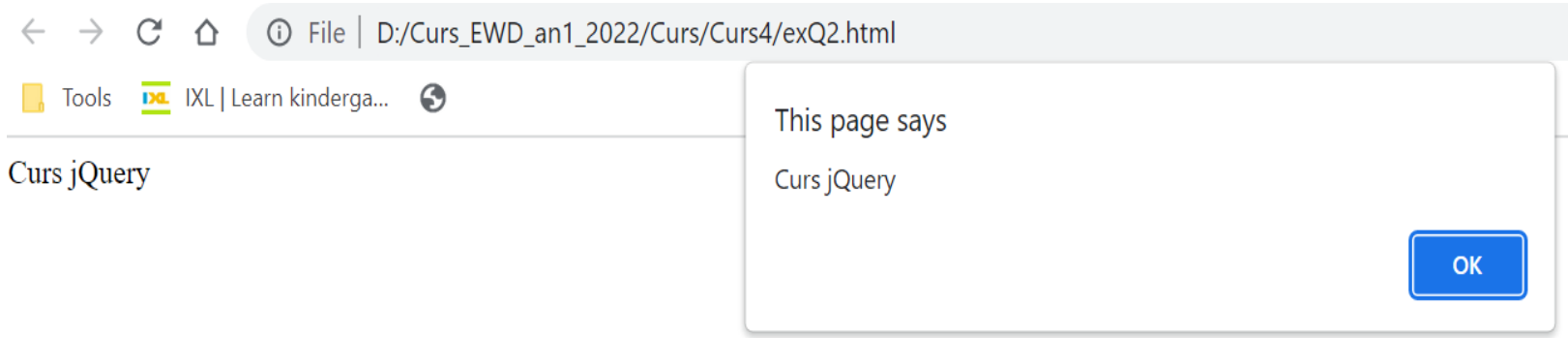
Lista completă cu selectorii folosiți in jQuery se găsește pe site-ul oficial, la pagina

https://api.jquery.com/category/selectors/

```
<!doctype html>
<html>
<head>
<meta charset="utf-8" />
<title>Simplu select</title>
<script src="JS/jquery-3.6.0.js"></script>
<script><!--
$(document).ready(function() {
  var ctn = $('#div1').html();
  alert(ctn);
});
--></script>
</head>
<body>
<div id="div1">Curs jQuery</div>
</body>
</html>
```

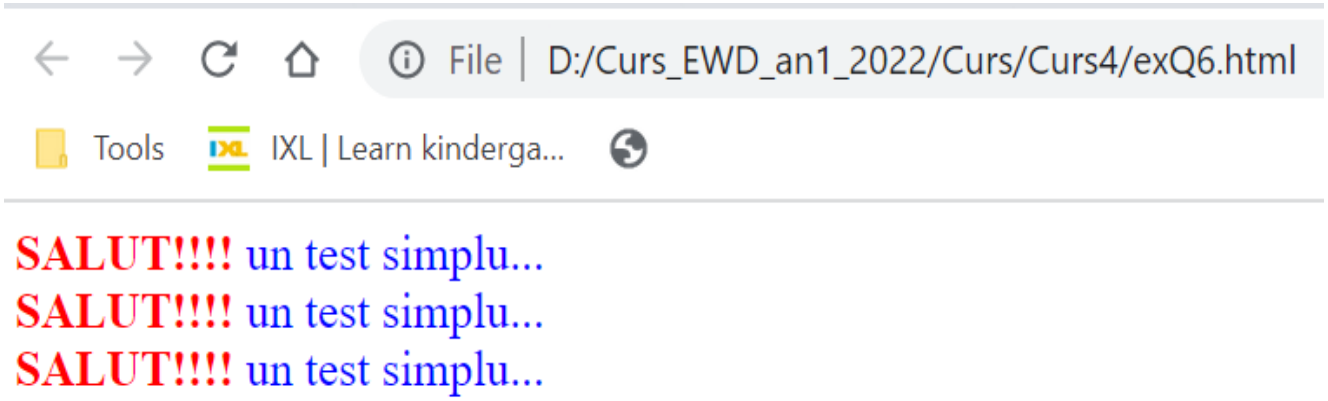
- *exemplu afisează o fereastră Alert cu conținutul unui DIV cu id="div1"*
- *Funcția **html()** este o metodă jQuery care returnează conținutul HTML din elementul la care este aplicată.*

exQ2.html



```
<!doctype html>
<head>  <meta charset="utf-8">
  <title>html demo</title>
  <style>
  div {
    color: blue;
    font-size: 18px;
  }
  </style>
<script src=" JS/jquery-3.6.0.js ">
</script>
</head>
<body>
<div></div>
<div></div>
<div></div>
```

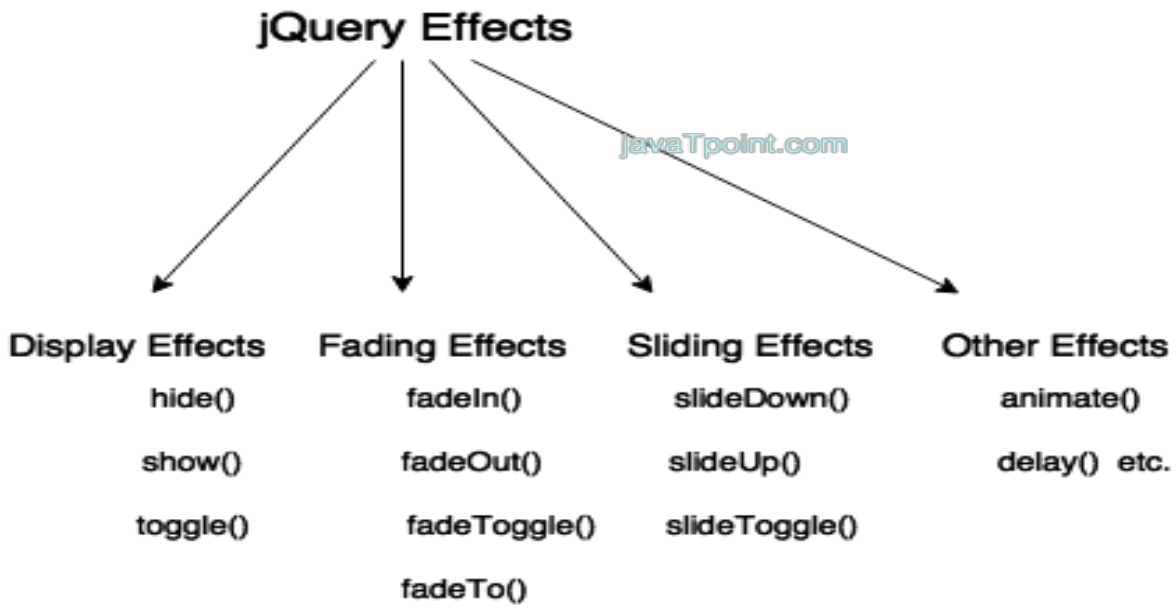
```
<script>
$( "div" ).html( "<b>SALUT!</b> un test
simplu..." );
$( "div b" )
  .append( document.createTextNode( "!!!" ) )
  .css( "color", "red" );
</script>
</body>
</html>
```



https://learn.jquery.com/

jQuery ne permite să adăugăm efecte pe o pagină web.

Efectele jQuery pot fi clasificate în efecte care estompează elemente (fading), care glisează, care ascund / afișează elemente și efecte de animație.



https://www.javatpoint.com/jquery-effects

- Pentru a ascunde un element in pagină, se poate folosi metoda **hide()**

$('selector').hide('durata');

"*durata*" - (optional) determină viteza animației de ascundere. Poate fi unul din șirurile "**fast**" și "**slow**" sau un număr care indică durata in milisecunde

- Pentru a face vizibil un element ascuns se folosește metoda **show()**

$('selector').show('durata');

- Metoda **toggle()** inversează starea obiectului, îl ascunde dacă este vizibil, și-l afișează dacă este ascuns

$('selector').toggle('durata');

- Dacă se doresc a fi executate anumite instrucțiuni după ce efectele de ascundere/ afișare sunt complete, se pot folosi variantele:

$('selector').metoda('durata', function() {

// cod ce trebuie executat

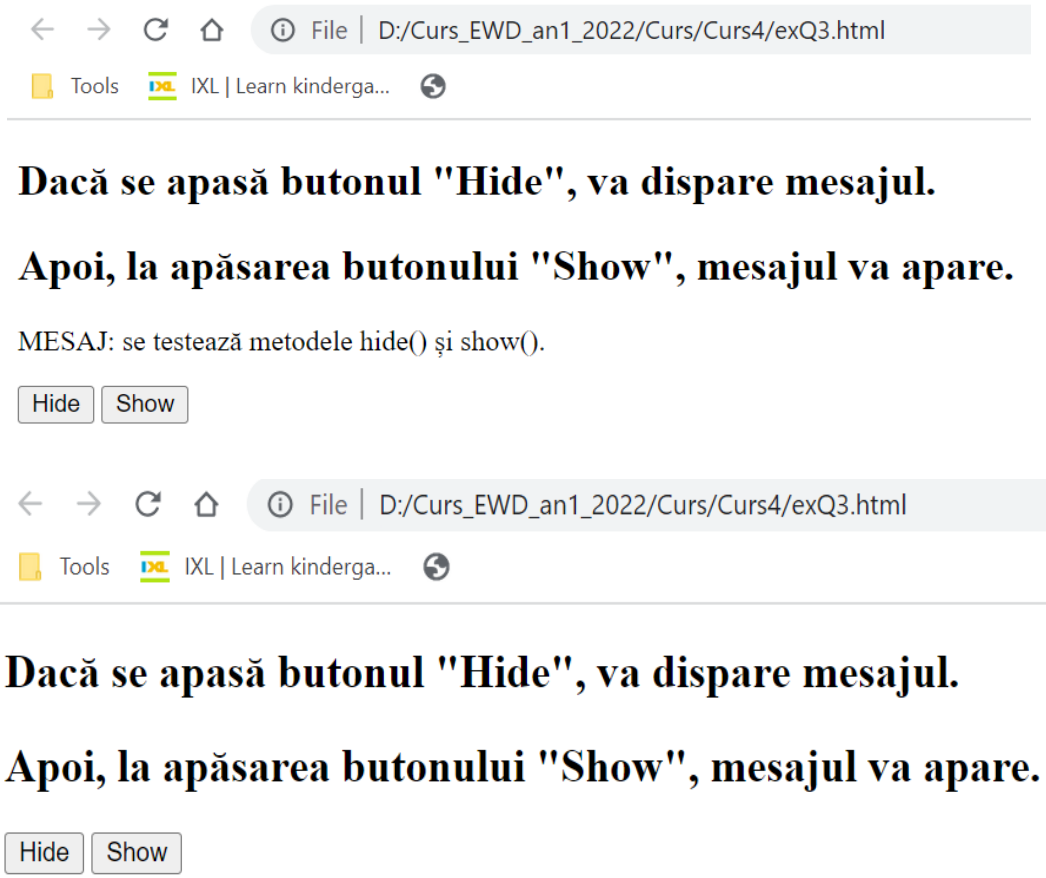
});

unde **metoda** poate fi **hide, show, toggle**

```
<!DOCTYPE html>
<html> <head> <meta charset="utf-8" />
<script src="JS/jquery-3.6.0.js"></script>
<script>
$(document).ready(function(){
  $("#hide").click(function(){
    $("p").hide();
  });
  $("#show").click(function(){
    $("p").show();
  });
});
</script>
</head>
<body>

<h2>Dacă se apasă butonul "Hide", va dispare mesajul.</h2>
<h2>Apoi, la apăsarea butonului "Show", mesajul va apare. </h2>
<p>MESAJ: se testează metodele hide() și show(). </p>
<button id="hide">Hide</button>
<button id="show">Show</button>
</body> </html>
```

exQ3.html



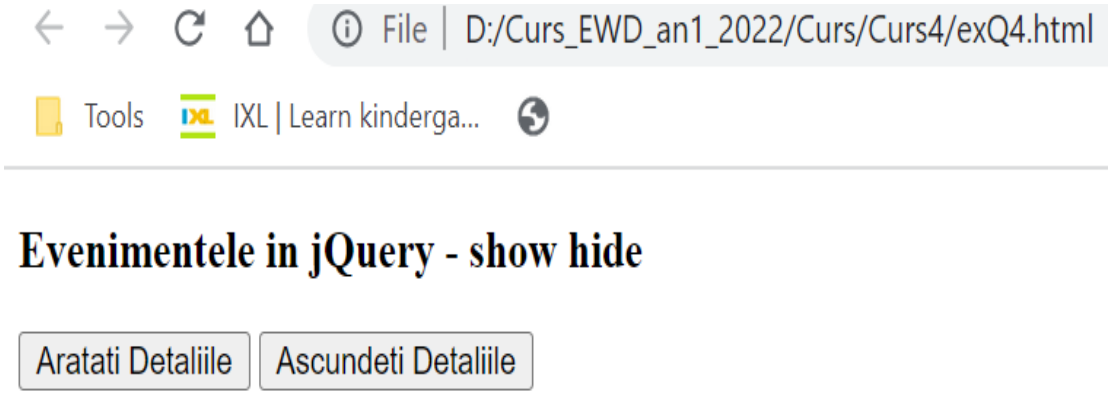
- **Evenimentele sunt acțiuni realizate de utilizatori sau de browser.**

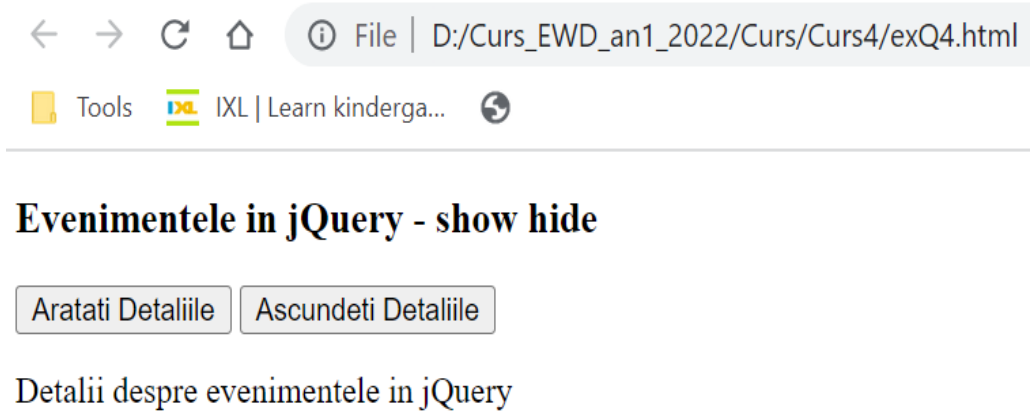
- jQuery - răspunde la evenimente într-o pagină HTML

```
<!DOCTYPE html>
<html>
<head><meta charset="utf-8" />
<script src="JS/jquery-3.6.0.js">
</script>
</head>
<body>
<div id="d1">

<h3>Evenimentele in jQuery - show hide</h3>
<input type="button" id="button-show" value="Aratati Detaliile" />
<input type="button" id="button-hide" value="Ascundeti Detaliile" />
<div id="detalii" style="display: none;"><p>Detalii despre evenimentele in jQuery</p></div>
</div>
<script>
$('#button-show').click(function() {
$("#detalii").show();
});
$('#button-hide').click(function() {
$("#detalii").hide();
});
</script>
</body> </html>
```

exQ4.html

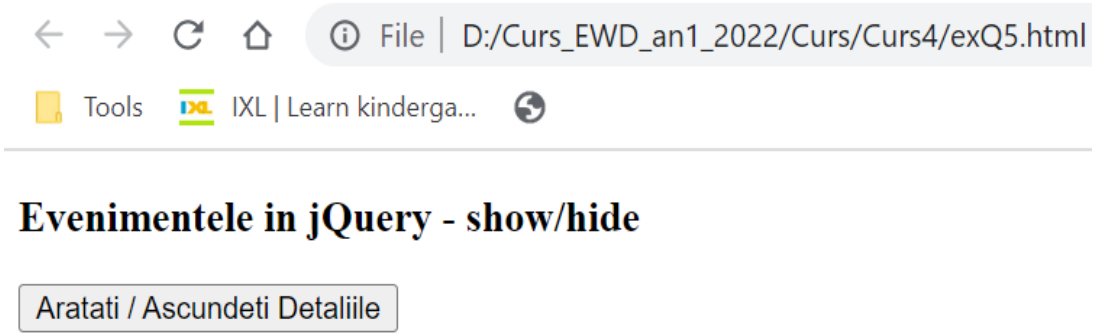


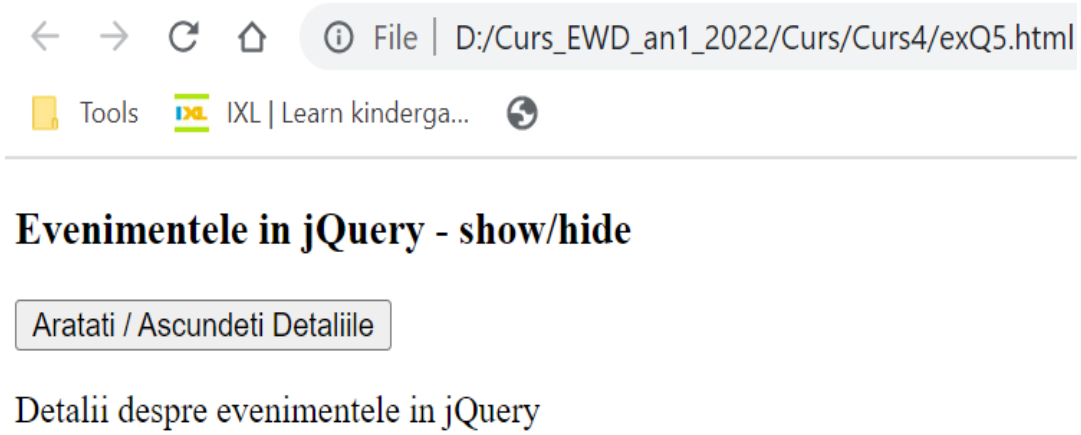


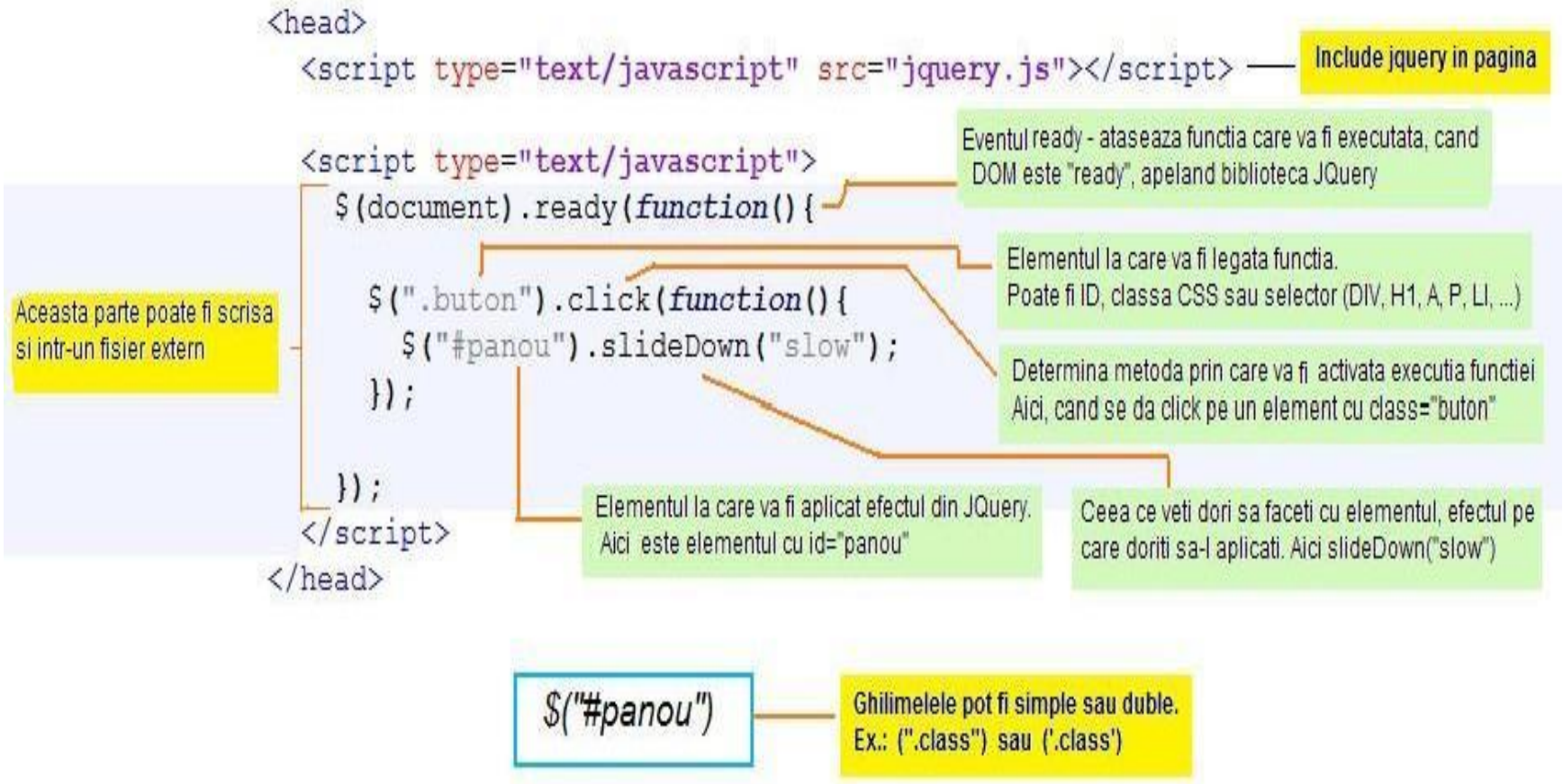
```
<!DOCTYPE html>
<html>
<head><meta charset="utf-8" />
<script src="JS/jquery-3.6.0.js"></script>
</head>
<body>

<div id="d1">
<h3>Evenimentele in jQuery - show/hide</h3>
<input type="button" id="sh" value="Aratati / Ascundeti Detaliile" />
<div id="detalii" style="display: none;" ><p>Detalii despre evenimentele in jQuery</p></div>
</div>
<script>
$('#sh').click(function() {
if ($("#detalii").is(":visible")) {
$("#detalii").hide();
} else {
$("#detalii").show();
}
});
</script>
</body> </html>
```

exQ5.html






<head>
<script type="text/javascript" src="jquery.js"></script>
Include jquery in pagina
<script type="text/javascript">
$(document).ready(function(){
Eventul ready - ataseaza functia care va fi executata, cand DOM este "ready", apeland biblioteca JQuery
$(".buton").click(function(){
Elementul la care va fi legata functia.
Poate fi ID, classa CSS sau selector (DIV, H1, A, P, LI, ...)
Determina metoda prin care va fi activata executia functiei
Aici, cand se da click pe un element cu class="buton"
$("#panou").slideDown("slow");
});
});
</script>
</head>
Aceasta parte poate fi scrisa si intr-un fisier extern
Elementul la care va fi aplicat efectul din JQuery.
Aici este elementul cu id="panou"
Ceea ce veti dori sa faceti cu elementul, efectul pe care doriti sa-l aplicati. Aici slideDown("slow")
$("#panou")
Ghilimelele pot fi simple sau duble.
Ex.: (".class") sau ('.class')