

Tehnici Web CURSUL 7

Semestrul II, 2021-2022
Carmen Chirita

<https://sites.google.com/site/fmitehnicweb/>

Evenimente

Atribut eveniment (in HTML)

```
<tag onclick="codJavascript">
```

Proprietate eveniment (in JavaScript)

```
element.onclick= numeFunctie;
```

```
element.onclick=function(){}
```

Gresit : element.onclick=numeFunctie();

Exemplu: la click pe buton se va afisa un mesaj intr-o fereastră alert

În HTML

```
<button id="bt" onclick="alert('Hello!');">Click me </button>
```

În JavaScript

```
var b = document.getElementById("bt");  
  
b.onclick = function(){alert("Hello!");}
```

Obiectele de tip Event

La aparitia unui eveniment este creat un obiect de tip Event care poate fi referit prin identificatorul event

-se poate referi in HTML : event

<p onclick="f(event)"> sau

-e parametrul implicit al functiei apelate de eveniment

element.onclick=function(e){}

Proprietati

event.target,
event.currentTarget,
event.type, event.timeStamp

Metode

event.preventDefault()
event.stopPropagation()
event.stopImmediate
Propagation()

Proprietati ale obiectului event:

event.type: numele evenimentului

event.target: tinta **initiala** a evenimentului

event.currentTarget: tinta **curenta** a evenimentului
(la capturare sau propagare)

event.timeStamp: numarul de milisecunde de la
incarcarea documentului pana la crearea
evenimentului

Exemplu

```
<script>
window.onload=function()
{
document.body.onclick=myFunction;
function myFunction(event) {
    alert("event.type: " + event.type + '\n' +
        "event.target: " + event.target.nodeName + '\n' +
        "event.currentTarget: " + event.currentTarget.nodeName + '\n' +
        "event.timeStamp: " + event.timeStamp);
}
}
</script>
</head>

<body>
<p>Paragraf 1</p>
<p>Paragraf 2</p>

</body>
```

Paragraf 1

Paragraf 2



Click pe paragraf

event.type: click
event.target: P
event.currentTarget: BODY
event.timeStamp: 28154.962256000003

Ok

Click în afara elementelor <p> și

event.type: click
event.target: BODY
event.currentTarget: BODY
event.timeStamp: 1151.4223690000001

Ok

Click pe imagine

event.type: click
event.target: IMG
event.currentTarget: BODY
event.timeStamp: 1264.551086

Ok

Exemple de evenimente

Evenimente determinate de mouse:

click
dblclick
mouseover
mouseout
mousedown
mouseup
mouseleave

Sunt reprezentate prin obiecte de tipul
[MouseEvent](#)

Evenimente determinate de tastatura:

keydown
keyup
keypress (deprecated)

Sunt reprezentate prin obiecte
[KeyboardEvent](#)

Obiectele MouseEvent au proprietati speciale

`event.button` // 0(stanga) 1(mijloc) 2(dreapta)

`event.clientX`, `event.clientY` // pozitia in fereastra (zona vizibila)

`event.pageX`, `event.pageY` //pozitia in document

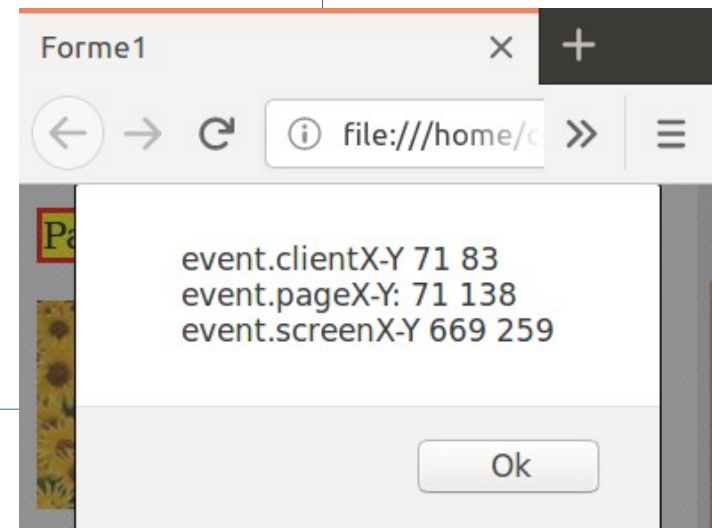
`event.screenX`, `event.screenY` //pozitia în ecran

`event.relatedTarget` //elementul legat de elementul care a declanșat evenimentul mouse-ului.

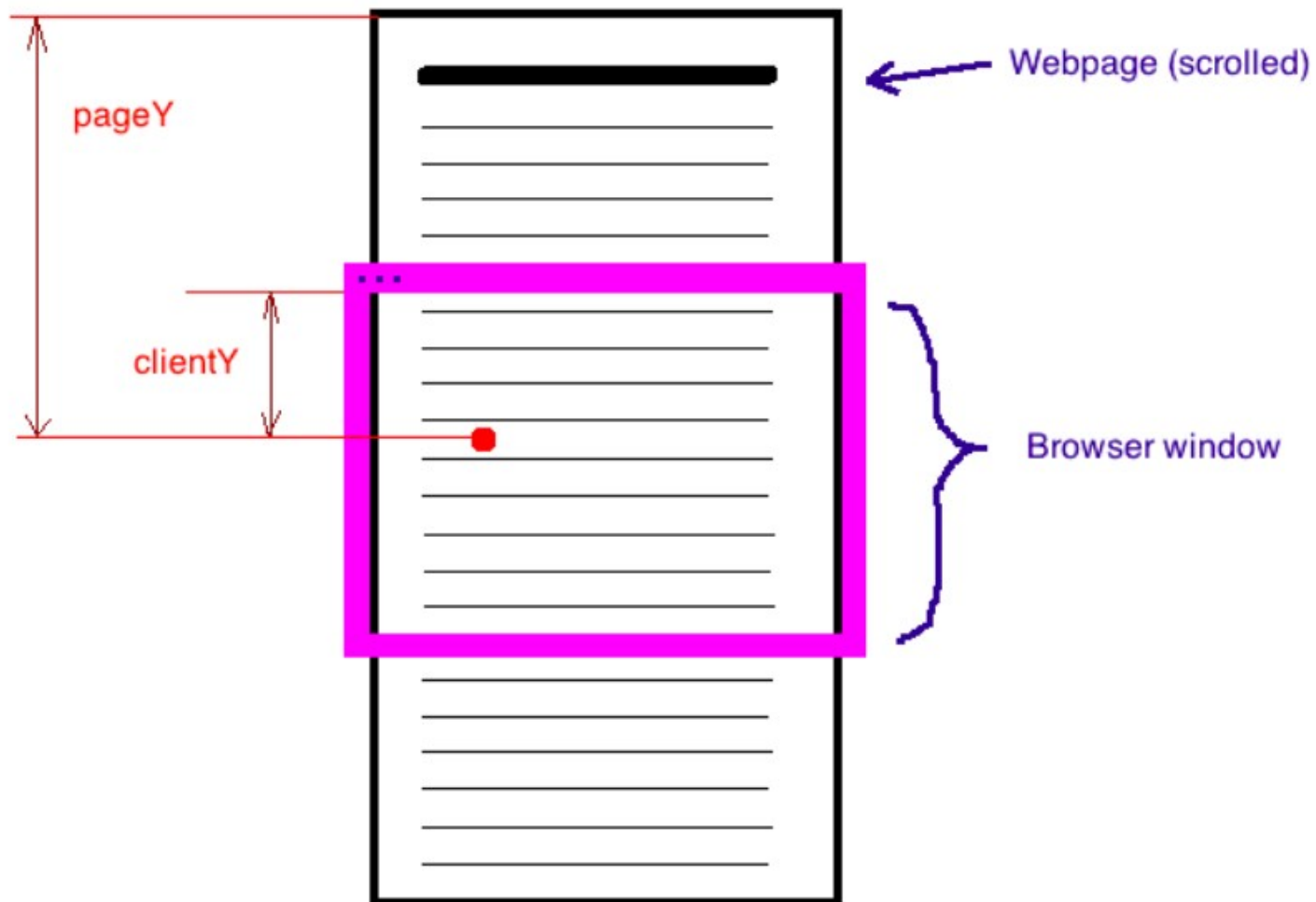
Exemplu

```
<script>
window.onload=function()
{
document.body.onclick=Poz;
function Poz(event) {
    alert("event.clientX-Y " + event.clientX + ' ' + event.clientY + '\n' +
        "event.pageX-Y: " + event.pageX + ' ' + event.pageY + '\n' +
        "event.screenX-Y " + event.screenX + ' ' + event.screenY);
}
}
</script>
</head>
<body>
<p>Paragraf 1</p>
<p>Paragraf 2</p>

</body>
```



Diferența dintre pageX-Y și clientX-Y



Obiectele KeyboardEvent au proprietati speciale

Proprietăți

`event.key` //tasta apasata
`event.altKey` // boolean
`event.ctrlKey` // boolean

Evenimente determinate
de tasta:

`keydown`
`keyup`

Sunt reprezentate prin obiecte
[KeyboardEvent](#)

deprecated
`event.keyCode`
`event.which`

```
<script>
window.onload=function()
{
var par=document.getElementById("par");
document.body.onkeyup=function(event)
{
switch (event.key) {
case "r":
par.style.color="red";
break;
case "g":
par.style.color="green";
break;
case "b":
par.style.color="blue";
break;
default:
alert("Alta tasta"); return;
}
document.getElementById("tasta").innerHTML=event.key;

}
}
</script>
</head>
<body>
<p id="par">Ati apasat tasta:</p>
<p id="tasta"><p>
</div>
</body>
```

Ati apasat tasta:

r

Ati apasat tasta:

g

Ati apasat tasta:

b

Alta tasta

Ok

Event listeners

sunt metode care inregistreaza functii handler pentru evenimente;

un eveniment poate avea inregistrate mai multe functii handler;

```
el.addEventListener("click", handleClick, false)
```

nume eveniment

functie

faza de
executie

```
el.removeEventListener("click",handleClick,true)
```

Event listeners

```
<button id="buton">Click me </button>
```

```
var b = document.getElementById("buton");
```

```
b.onclick = function(){alert("Hello!");}
```

forme echivalente

```
b.addEventListener("click", function(){alert("Good Bye!");}, false)
```

Captarea evenimentelor: obiectul **event** este transmis ca primul argument al handler-ului

```
element.onclick = myfct;  
  
function myfct (event) {alert(event.type);}
```

```
element.addEventListener("click", myfct);  
  
function myfct (event) {alert(event.type);}
```

```
element.onclick = function (event) {  
    alert(event.type);}
```

```
element.addEventListener("click",  
    function (event) {alert(event.type);} )
```

La aparitia unui eveniment, functiile handler sunt executate in ordinea in care sunt definite.

```
<script >
  window.onload=main;

  function main(){
    el=document.getElementById("buton");

    el.addEventListener("click", handle1, false);
    el.addEventListener("click", handle2, false);

    function handle1(event) {alert(1)};
    function handle2(event) {alert(2)};
  }
</script>
```

```
<body>
  <button type="button" id="buton"> click </button>
</body>
```


Setare faza_exec pentru PARINTE

- false - BUBLLING (implicit)
- true - CAPTURING

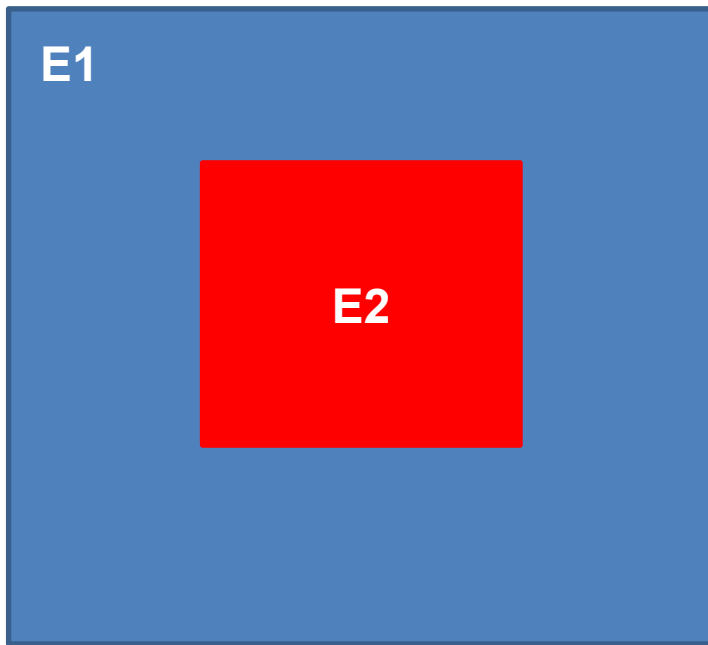
Modele de executie:

Parinte true (capturing):

intai handler **parinte** apoi handler **copil**

Parinte false(bubbling) :

intai handler **copil** apoi handler **parinte**



Event order

E1, E2 au handler la click

Evenimentul: click pe E2



Modele de executie:

CAPTURING: handler E1, handler E2

BUBBLING: handler E2, handler E1

La aparitia unui eveniment se executa:

1. handlerele stramosilor tinte setate pe CAPTURE
2. handlerul tinte
3. handlerele stramosilor tinte setate pe BUBBLE

Exemplu (addEvent.html)

```
<script>
window.onload=function()
{
var bunic=document.body;
var parinte=document.getElementById("parinte");
var copil=document.getElementById("copil");
bunic.addEventListener("click",function(){alert("BODY");bunic.className="rosu";},true);
parinte.addEventListener("click",function(){alert("DIV");parinte.className="galben";},false);
copil.addEventListener("click",function(){alert("P");copil.className="verde";});
}
</script>
</head>
<body id="bunic">
<div id="parinte">Div
<p id="copil">Paragraf</p>
</div>
</body>
```

```
<style>
.rosu{background-color:red;}
.galben{background-color:yellow;}
.verde{background-color:green;}
```

Se va executa:

1. handler BODY
2. handler P
3. handler DIV

```
div{border:2px solid green;
width:300px; height:300px;
}
p {
border: 3px solid black;
}
</style>
```

Metode obiect event:

`event.stopPropagation()`

opreste propagarea evenimentului in DOM;

`event.stopImmediatePropagation()`

daca mai multe functii listener sunt atasate aceluiasi element iar una contine `event.stopImmediatePropagation()` functiile listener urmatoare nu mai sunt apelate

`event.preventDefault()`

se anuleaza actiunea implicita a elementului

```
document.getElementById("link").addEventListener("click", function(event){
    event.preventDefault()
});
```

`Facultatea de Matematica și Informatica`

Exemplu (event.stopPropagation())

stopP.html

```
<script>
window.onload=function()
{
var bunic=document.body;
var parinte=document.getElementById("parinte");
var copil=document.getElementById("copil");
bunic.addEventListener("click",function(event){alert("BODY");bunic.className="rosu";});
parinte.addEventListener("click",function(event){event.stopPropagation();
alert("DIV");parinte.className="galben";});
copil.addEventListener("click",function(event){alert("P");copil.className="verde";});
}
</script>
</head>
<body id="bunic">
<div id="parinte">Div
<p id="copil">Paragraf</p>
</div>
</body>
```

Se va executa:

1. handler P
2. handler DIV

```
<style>
.rosu{background-color:red;}
.galben{background-color:yellow;}
.verde{background-color:green;}

div{border:2px solid green;
width:300px; height:300px;
}
p {
border: 3px solid black;
}
</style>
```

Exemplu (event.stopImmediatePropagation())

stopImP.html

```
<script>
window.onload=function()
{
var bunic=document.body;
var parinte=document.getElementById("parinte");
var copil=document.getElementById("copil");
bunic.addEventListener("click",function(event){alert("BODY");bunic.className="rosu";},true);
parinte.addEventListener("click",function(event){alert("DIV");parinte.className="galben";});
copil.addEventListener("click",function(event){alert("Prima functie");copil.className="verde";});
copil.addEventListener("click",function(event){event.stopImmediatePropagation();
alert("A doua functie");});
copil.addEventListener("click",function(event){alert("A treia functie");});
}
</script>
</head>
<body id="bunic">
<div id="parinte">Div
<p id="copil">Paragraf</p>
</div>
</body>
```

Se va executa:

1. handler BODY
2. handler1 P
3. handler2 P

```
<style>
.rosu{background-color:red;}
.galben{background-color:yellow;}
.verde{background-color:green;}

div{border:2px solid green;
width:300px; height:300px;
}
p {
border: 3px solid black;
}
</style>
```

Metode ale obiectului window: timers

window.setTimeout

apelează o funcție sau execută un fragment de cod după un anumit timp (milisecunde).

```
vt=setTimeout(numeFuncție,intarziere,parametri);  
vt=setTimeout(function(){}, intarziere,parametri);  
clearTimeout(vt) // anulare funcție lansată
```

vt –variabila globală pentru a fi văzută de clearTimeout
parametrii sunt ai funcției care se va executa (numeFuncție sau anonimă)

Metode ale obiectului window: timers

window.setInterval

executa functia in mod repetat, la un anumit interval de timp.

```
vt=setInterval(numeFunctie, interval, parametri);  
vt=setInterval(function(){}, interval, parametri);  
clearInterval(vt) // anulare functie lansata
```


Exemplul 1

```
var hello; var end = 15000;  
function sayHello() {alert("hello");}  
  
function sayHelloMany()  
    { hello = setInterval(sayHello, 3000);}  
  
function stopHello(){clearInterval(hello);}  
sayHelloMany();  
setTimeout(stopHello,end);
```

Let


```
var i=0;  
el.onclick=function()  
    {alert(i); /* evaluate la click -va afisa 1*/ }  
i=1;
```

```
var i=0;  
{let il=i;// il se creaza acum  
el.onclick=function(){alert(il); /*va afisa 0*/ }  
}// se elibereaza zona il=0  
i=1;
```

Exemplu cu let:

La click pe fiecare imagine din document sa se afiseze sursa imaginii

```
var imagini=document.getElementsByTagName("img");  
  
for(var i=0;i<imagini.length;i++)  
imagini[i].onclick=function(){alert(imagini[i].src);}  
    // nu functioneaza;  
    // i va fi egal cu imagini.length
```



Soluție:

```
for(let i=0;i<imagini.length;i++)  
imagini[i].onclick=function(){alert(imagini[i].src);}
```

Exemplul 2 (setTimeout cu parametri) time.html

```
<script>
window.onload=function()
{
var pl=document.getElementsByTagName("p");
for(var i=0;i<pl.length;i++)
    setTimeout(colorare,3000*(i+1),"red",pl[i]);

function colorare(culoare,ob)
{
    ob.style.color=culoare;
}
}
</script>
</head>
<body>
<p>Paragraful 1</p>
<p>Paragraful 2</p>
<p>Paragraful 3</p>
<p>Paragraful 4</p>
<p>Paragraful 5</p>
<p>Paragraful 6</p>
<p>Paragraful 7</p>
<p>Paragraful 8</p>
</body>
```

Varianta fără parametri

```
for(let i=0;i<pl.length;i++)
    setTimeout(function(){colorare("red",pl[i]);},3000*(i+1));
```

window.getComputedStyle

determina stilul efectiv aplicat unui element

window.getComputedStyle(ob, ":after")

este obiect din clasa CSSStyleDeclaration
este **read-only**

pseudo-element sau null
optional

```
var oStil = window.getComputedStyle(ob,null) ;  
var x=oStil.color; // proprietatea css ob.style.color
```

sau

```
window.getComputedStyle(ob,null).getPropertyValue("font-size")
```

valoarea proprietatii CSS

```
<style>
div{border:2px solid green;
  background-color:red;
  width:200px; height:100px;
}
</style>
<script>
window.onload=function()
{
var p=document.getElementById("stil");
var div=document.getElementById("div");
var stil=window.getComputedStyle(div);
  p.innerHTML+= '<br>' +
    "background-color: " + stil.getPropertyValue("background-color") + '<br>' +
    "width:" + stil.getPropertyValue("width") + '<br>' +
    "height:" + stil.getPropertyValue("height") + '<br>' +
    "border:" + stil.getPropertyValue("border-left-color");
}
</script>
```

```
<body>
<p id="stil"><b>Stilizarea:</b></p>
<div id="div">Div
</div>
</body>
```

Stilizarea:

background-color: rgb(255, 0, 0)
width:200px
height:100px
border:rgb(0, 128, 0)

Div



Obiectul window: proprietăți și metode

Obiectul window reprezintă fereastra browserului care conține DOM asociat documentului (window.document).

Obiectul window asociat unui anumit document poate fi selectat prin document.defaultView

Dacă un document HTML conține elemente <iframe>, browserul crează un obiect window pentru documentul HTML și un obiect window suplimentar pentru fiecare <iframe>

Multiple windows: ferestrele **iframe** au urmatoarele proprietati:

contentWindow // obiectul window corespunzator iframe-ului
parent // fereastra parinte

```
var x = document.getElementsByTagName("iframe")[0].contentWindow;  
  
//x = obiectul window corespunzător primului iframe  
  
x.document.getElementsByTagName("body")[0].style.backgroundColor = "blue";
```


Proprietati ale obiectului window:

window.location

- folosit pentru a determina URL-ul paginii curente și pentru a redirecționa browserul către o pagină nouă.
- este un obiect din clasa Location

Proprietăți și metode

window.location.href // adresa URL a documentului
window.location.hostname // numele de domeniu al gazdei web
window.location.pathname // calea și numele fișierului paginii curente
window.location.protocol // protocolul folosit (http: sau https:)
window.location.hash // partea din URL care începe cu #
window.location.assign(url) // încarca un nou document

<protocol>//<hostname>:<port>/<pathname><search><hash>

Proprietati ale obiectului window

Schimbarea locatiei

```
window.location.href="http://fmi.unibuc.ro"  
sau  
window.location="http://fmi.unibuc.ro"  
sau  
location="http://fmi.unibuc.ro"
```

Proprietati similare

document.location si document.location.href

Proprietati ale obiectului window:

window.history

- conține istoria browserului (adresele URL accesate de utilizator)
- este un obiect din clasa History

Proprietăți și metode

history.length //nr de URL-uri din lista de istoric

history.back() // încarcă adresa URL anterioară din lista de istoric

history.forward() // încarcă următoarea adresă URL din lista de istoric

history.go(number/url) //încarca o adresa URL specificata din lista de istoric

history.go(1) // history.go(-1)

Proprietati ale obiectului window

`window.screen`

- conține informații despre ecranul utilizatorului.
- este un obiect din clasa `Screen`

Proprietăți și metode

`screen.availHeight` // înălțimea ecranului (excluzând Windows Taskbar)
`screen.availWidth` // lățimea ecranului (excluzând Windows Taskbar)
`screen.height` // înălțimea totală a ecranului
`screen.width` // lățimea totală a ecranului
`screen.colorDepth`
`screen.pixelDepth` //nr de biți folosiți pentru culoarea unui pixel

```

<script type="text/javascript" >
window.onload = function () {
document.getElementById("b1").addEventListener("click", function()
{ document.getElementById("p1").innerHTML += "<br>" +
                                window.location.href;});
document.getElementById("b2").addEventListener("click", function(){location.href="http://fmi.unibuc.ro";});
document.getElementById("b3").addEventListener("click", function(){ history.go(1);});
document.getElementById("b4").addEventListener("click", function(){history.go(-1);});
document.getElementById("b5").addEventListener("click", function() {
document.getElementById("p2").innerHTML += "<br>" + screen.availWidth + " " + screen.availHeight +
                                '<br>' + screen.width + " " + screen.height +
                                '<br>' + screen.colorDepth + " " + screen.pixelDepth;
                                });
}
</script>
</head>
<body>
<button id="b1"> Arata locatia </button>
<button id="b2"> Schimba locatia</button>
<button id="b3"> History -> </button>
<button id="b4">History <- </button>
<button id="b5"> Screen </button>
<p id="p1"><b>Location:</b></p>
<p id="p2"><b>Screen:</b></p>
</body>

```

Arata locatia

Schimba locatia

History ->

History <-

Screen

Location:

file:///home/carmen/TEHNICI_WEB_CURSURI/EXEMPLE/ex-window-location-history-screen.html

Screen:

1416 807

1472 828

24 24

Metode ale obiectului window

`window.open(URL, loc, caracteristici, replace)`

- deschide o noua fereastră de browser
- toți parametrii sunt optionali
- dacă nu se specifica URL se va deschide o fereastră cu about-blank

Exemplu

```
window.open("https://www.w3schools.com", "_blank",  
"toolbar=yes,scrollbars=yes,resizable=yes,top=500,left=500,width=400  
,height=400");
```

Metode ale obiectului window

```
var MyWindow=window.open();  
MyWindow.close();
```

inchide fereastra deschisă cu metoda open()

```
<script>  
var myWindow; // globala pt a fi vazuta si la apelul close()  
function openWin()  
{ myWindow = window.open("http://fmi.unibuc.ro/ro/");  
}  
function closeWin()  
{ myWindow.close();  
}  
</script>  
</head>  
<body>  
<button onclick="openWin()">Open "myWindow"</button>  
<button onclick="closeWin()">Close "myWindow"</button>  
</body>
```