Tehnici Web
CURSUL 8

Semestrul I, 2017-2018

Carmen Chirita

https://sites.google.com/site/fmitehniciweb/

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 fereastra 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(event){}

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)

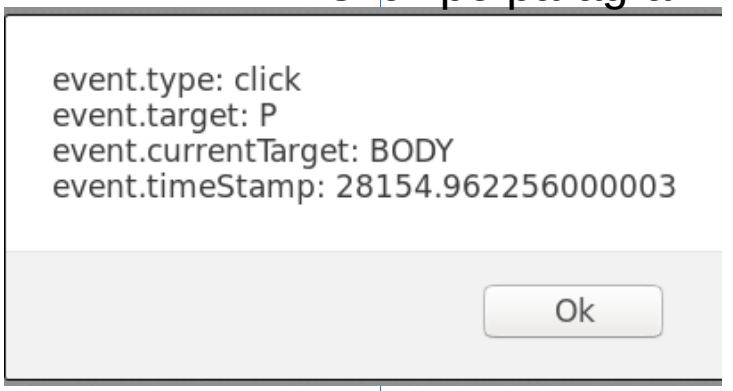
event.timeStamp: timpul (in milisecunde)

de la incarcarea paginii pana la inceputul evenimentului.

Exemplu

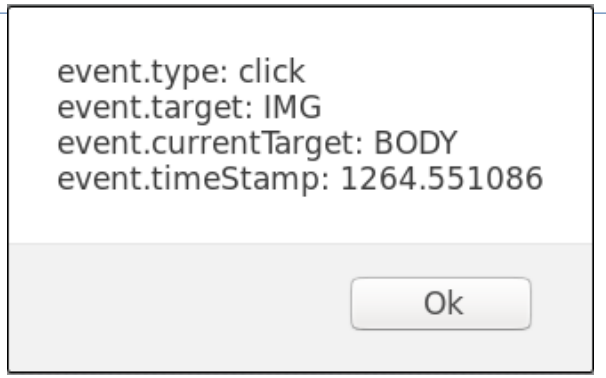
```
<script>
window.onload=function()
{
document.body.onclick=myFunction;
function myFunction(event) {
    alert("event.type: " + event.type + '\n' +
          "event.target: " + event.target.nodeName + '\n' +
          "event.currentTarget: " + event.currentTarget.nodeName + '\n' +
          "event.timeStamp: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

<body>
<p>Paragraf 1</p>
<p>Paragraf 2</p>
<img src="vara.jpg" alt="vara">
</body>
```

Click pe paragraf



Click în afara elementelor <p> și <img>

Click pe imagine



Exemple de evenimente

Evenimente determinate
de mouse:

click
dblclick
mouseup
mousedown
mouseover
mouseout
mouseleave

Sunt reprezentate prin obiecte de tipul
MouseEvent

Evenimente determinate
de tastaura:

keydown
keypress
keyup

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)

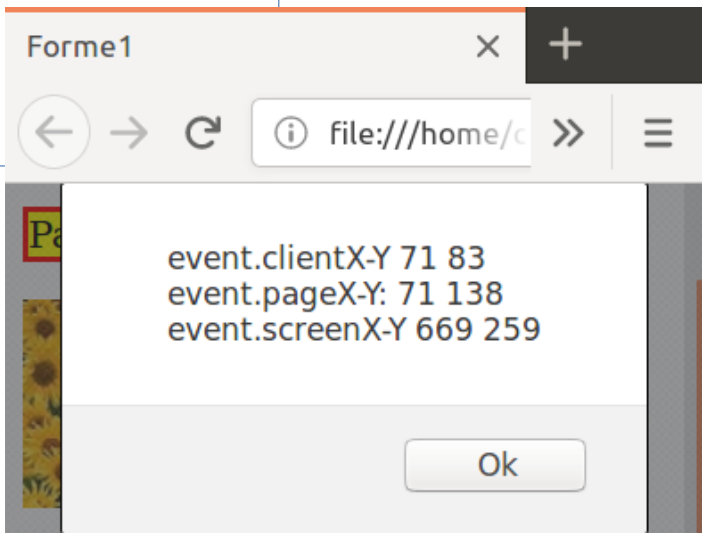
event.pageX, event.pageY //pozitia in document

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

event.relatedTarget //elementul legat de elementul care a decla
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>
<img src="vara.jpg" alt="vara">
</body>
```



Obiectele KeyboardEvent au proprietati speciale

Proprietăți

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

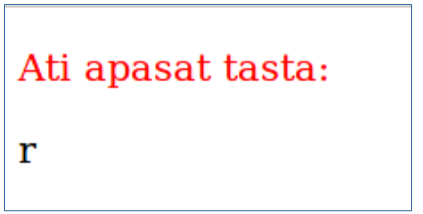
deprecated
event.keyCode
event.which

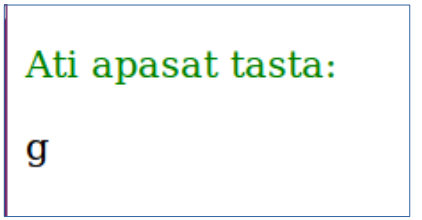
Evenimente determinate
de tastaura:

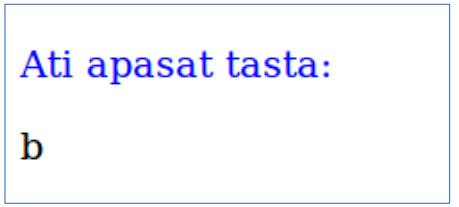
keydown
keypress
keyup

Sunt reprezentate prin obiecte
KeyboardEvent

```
<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>
```



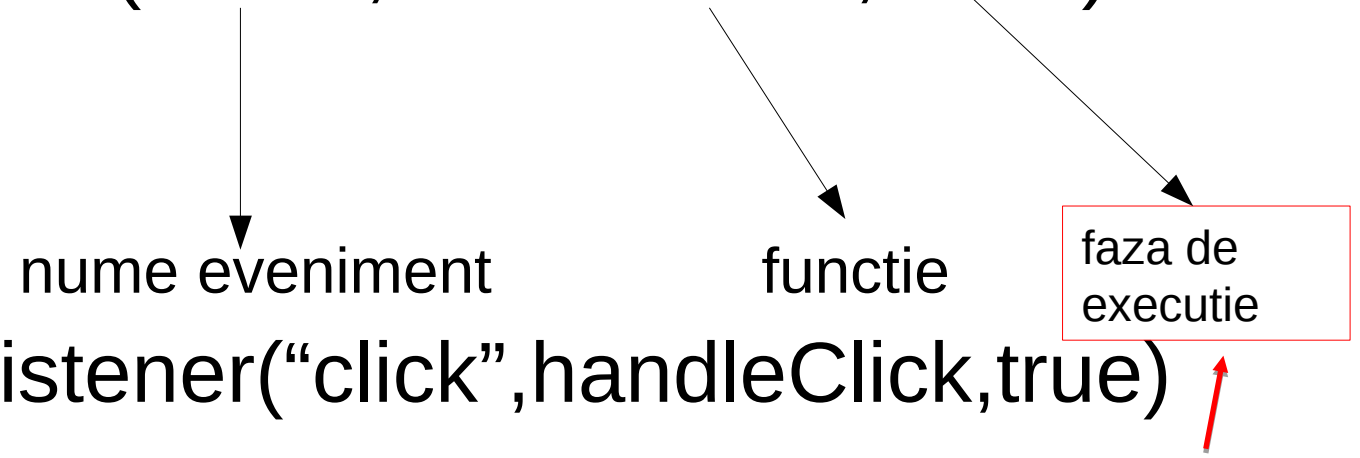




Event listeners

sunt metode care inregistreaza functii handler pentru evenimente;
un eveniment poate avea inregistrate mai multe functii handler;

el.addEventListener(“click”, handleClick, false)



el.removeEventListener(“click”,handleClick,true)

https://developer.mozilla.org/en-US/docs/Web/API/Event

Event listeners

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

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

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

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

forme echivalente

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

```
window onload = function (){

var b = document.getElementById('bt');

b.onclick = myfct;

function myfct (event) {alert(event.type);}

}
```

```
window onload = function (){

var b = document.getElementById('bt');

b.addEventListener("click", myfct);

function myfct (event) {alert(event.type);}

}
```

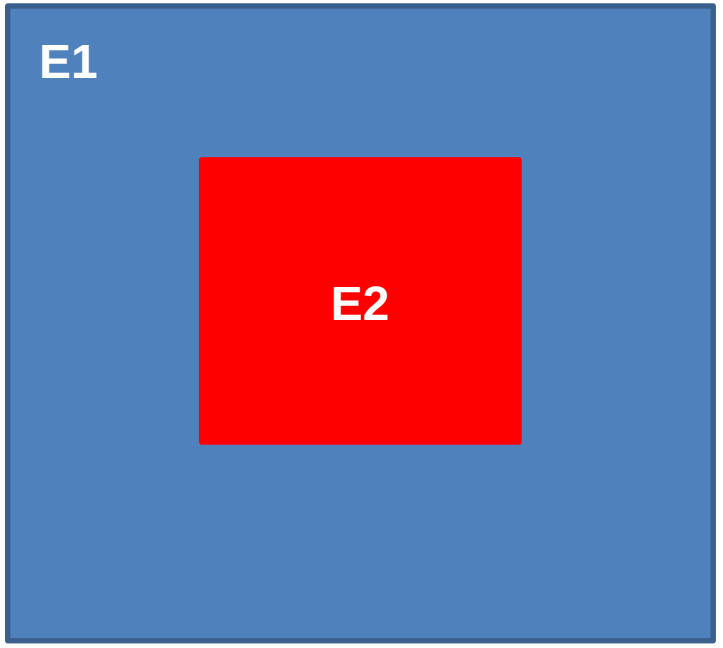
```
window onload = function (){

var b = document.getElementById('bt');

b.onclick = function (event) {

                alert(event.type);}

}
```

```
window onload = function (){

var b = document.getElementById('bt');

b.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>
```



E1, E2 au handlere la click

Evenimentul: click pe E2

Modele de executie:

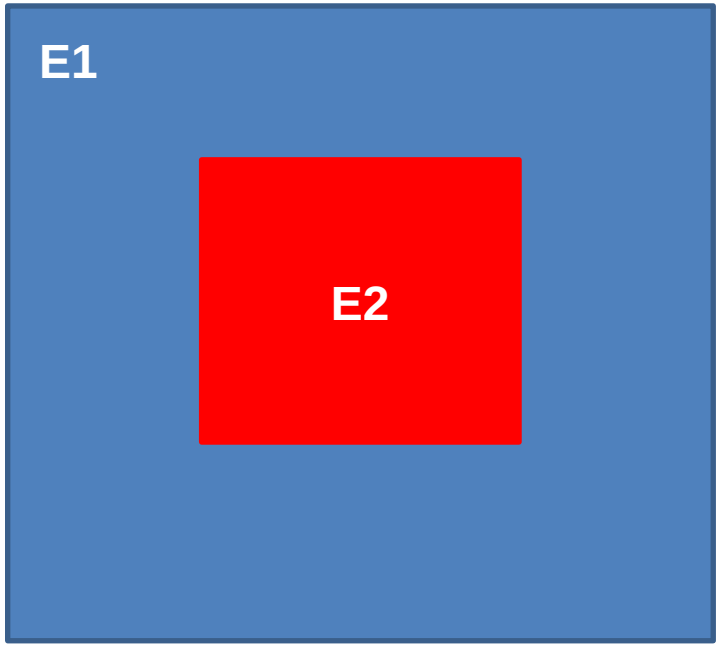
CAPTURING: handler E1, handler E2
BUBBLING: handler E2, handler E1

[Event order]

W3C:

- orice eveniment are doua faze: CAPTURE si BUBBLE
- un handler asociat evenimentului poate fi setat

pentru CAPTURE cand faza de executie este true
pentru BUBBLE cand faza de executie este false



E1, E2 au handlere la click

Evenimentul: click pe E2

Modele de executie:

CAPTURING: handler E1, handler E2
BUBBLING: handler E2, handler E1

[Event order]

La aparitia unui eveniment se executa:

1. handlerele stramosilor tintei setate pe CAPTURE
2. handlerul tintei
3. handlerele stramosilor tintei 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;}

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

Se va executa:

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

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()
});

<a id=”link” href=”fmi.unibuc.ro”>Facultatea de Matematica și Informatica</a>
```

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(numeFunctie,intarziere,parametri);
vt=setTimeout(function(){}, intarziere, parametri);
clearTimeout(vt) // anulare functie lansata
```

vt –variabila globala pentru a fi vazuta de clearTimeout

parametrii sunt ai functiei care se va executa (numeFunctie sau anonima)

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);
```

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

```
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