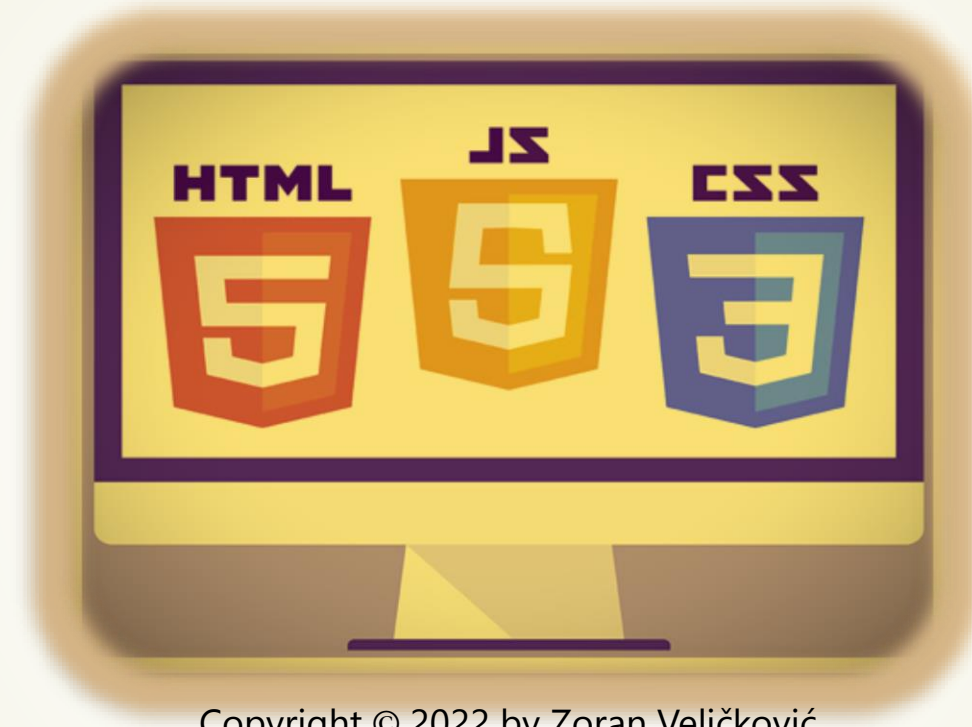




# Akademija tehničko-vaspitačkih strukovnih studija



Copyright © 2022 by Zoran Veličković

## INTERNET TEHNOLOGIJE

Prof. dr Zoran Veličković, dipl. inž. el.

2022/23.

Prof. dr Zoran Veličković, dipl. inž. el.

# Internet tehnologije



**Funkcionalnost Web stranica – JavaScript**

(7)



# Sadržaj

- ▶ Funkcionalnost Web stranica
- ▶ Osnove JavaScript-a
  - ▶ Programski kod: JavaScript
  - ▶ Web stranica: JavaScript
- ▶ JavaScript funkcije
  - ▶ Izracunaj\_IQ: JavaScript funkcija
  - ▶ Anonimne JavaScript funkcije
- ▶ JavaScript: službene reči
- ▶ JavaScript: tipovi promenljivih
- ▶ JavaScript operatori
- ▶ Obrada događaja u JavaScriptu
  - ▶ Atributi obrada događaja u JavaScriptu
- ▶ Poziv funkcije JavaScripta
  - ▶ Metoda addEventListener
- ▶ jQuery i JavaScript

# Funkcionalnost Web stranica

- ▶ Od savremenih Web strana se zahteva da izgledaju **ATRAKTIVNO**, a da pri tome poseduju određenu dozu **INTERAKTIVNOSTI** i **FUNKCIONALNOSTI**.
- ▶ Da bi se ispunili pomenuti zahtevi, neophodno je integrisati **TRI** NEZAVISNE Web TEHNOLOGIJE:
  - ▶ **HTML**,
  - ▶ **CSS**,
  - ▶ **JavaScript**.
- ▶ Već znate da je **HTML** zadužen za **ORGANIZACIJU STRUKTURE** Web dokumenta, a da je **CSS** zadužen za **ATRAKTIVAN IZGLED** Web stranice.
- ▶ Osnovna uloga **JavaScript**-a na Web stranici je da obezbedi **FUNKCIONALNOST** i **INTERAKTIVNOST** sa korisnikom i obezbedi **DINAMIKU** na Web stranici.
- ▶ Za primenu **JavaScript**-a - treće komponente Web tehnologija, potrebno OBEZBEDITI **PODRŠKU** Web čitača, što je i osnovni **IZVOR NEKOMPATIBILNOSTI** između njih.

# Osnove JavaScript-a

- ▶ Generalno, **JavaScript** je **OO PROGRAMSKI JEZIK** koji zahteva **IZVRŠNO OKRUŽENJE** i **NE MOŽE** se koristiti nezavisno od Web čitača.
- ▶ Ovaj **NEDOSTATAK** (ako se tako može nazvati) je Google pokušao da reši razvojem novih INTERPRETERA i KOMPAJLERA (primer V8 - ) za JavaScript, čime ga je učinio **NEZAVISNIM JEZIKOM** koji radi u većini okruženja.
- ▶ **JavaScript** je takođe i **SKRIPT JEZIK** (vidi se iz samog imena!) jer omogućava pisanje skriptova, odnosno, **niza naredbi** koje će Web **ČITAČ** (okruženje) **IZVRŠITI**.
- ▶ Najčešće se programski iskazi JavaScript-a **UGRAĐUJU** u **HTML kôd** Web strana.
- ▶ Primer primene JavaScript-a je **PROVERA** unetih podataka u Web obrasce koje popunjava klijent (lab. vežba).
- ▶ Na ovaj način JavaScript obezbeđuje **KOMUNIKACIJU** HTML-a sa **OBRASCEM**.
- ▶ Danas je moguće obaviti **OSNOVNU** proveru obrazaca korišćenjem HTML5 ili CSS3.

# Osnove JavaScripta

- Može se reći da je JavaScript programski jezik koji **PRETVARA Web stranice** u **Web APLIKACIJE**.
- Zapravo, tek **POTPUNO FUNKCIONALNA** Web stranica koja pruža korisniku komfor **DESKTOP aplikacije** se može nazvati **Web APLIKACIJOM**.
- **INTERAKTIVNOST** Web aplikacije se odnosi na mogućnost interakcije korisnika sa sadržajem Web stranice bez njenog **PONOVOG UČITAVANJA**.
- Pomoću JavaScripta se Web čitač, koji uglavnom služi za **PRIKAZIVANJE DOKUMENATA**, pretvara u platform za **KREIRANJE APLIKACIJA**.
- JavaScript obezbeđuje **FUNKCIONALNOST** Web stranice korišćenjem **programiranja VOĐENOG DOGAĐAJIMA**.
- **DOGAĐAJIMA** na Web stranici upravlja Web čitač (izvršno okruženje) koji ima mogućnost **AUTOMATSKOG POZIVANJA** Javascript funkcija na pojavu događaja.

# Umetanje JavaScripta u HTML kod

- Programski iskazi JavaScript-a se umeću između početne i završne **HTML OZNAKE** za script u **<head>** ili **<body>** oznaci (kod HTML5):

```
<script>
```

```
... // JavaScript programski iskazi
```

```
</script>
```

- Da bi se omogućilo **PONOVNO KORIŠĆENJE** napisanih skriptova i skratilo vreme učitavanja, preporučuje se da se JavaScript kôd smesti u **JEDNU** ili **VIŠE SPOLJNIH DATOTEKA** (nešto slično kao i CSS) koje se referenciraju atributom "**src**".

```
<html>
```

```
<head>
```

```
<script src = "mycode.js"> </script>
```

```
</head>
```

```
...
```

```
</html>
```

Atribut **src** definiše spoljašnju datoteku sa JavaScriptom **mycode.js**

# Umetanje JavaScripta u HTML 5

- **PRE** pojave HTML 5 JavaScript se postavljao **ISKJUČIVO** u zaglavlju Web stranice (kako je to prethodno pokazano), dakle na početku kodiranja same Web stranice.
- Sa pojavom HTML 5 preporučuje se postavljanje oznake `<script>` i `</script>` **NA DNU** Web stranice zbog bržeg učitavanja Web stranice.
- **KOMENTAR** u kodu JavaScript-a se označava „//“ na početku programskog reda - iskaza.

```
<html>
  <head> <title> JS u HTML 5 </title> </head>
  <body>
  ...
  ...
  <script src = "mycode.js">
    // JS kod
  </script>
</body>
</html>
```

Postavljanje veze sa  
eksternim JavaScript fajlom  
na dnu Web stranice.

Komentar unutar  
JavaScripta.



# Programski kod: JavaScript

```
<html>
```

```
<body>
```

```
<h1>HTML i Web dizajn kroz praktične primer</h1>
```

```
<p>Ovaj tekst se ispisuje DIREKTNO na Web stranici.</p>
```

```
<script language = "JavaScript">
```

```
<!--
```

```
document.write("<p>Ovo je linija teksta ubačena"  
                +"JavaScript komandom.</p>");
```

```
alert ("Moj prvi JavaScript skript glasi:\"Zdravo svete\"!");
```

```
//-->
```

```
</script>
```

```
....
```

```
</html>
```

HTML kod

Stare verzije  
čitača  
zahtevaju  
„komentar“

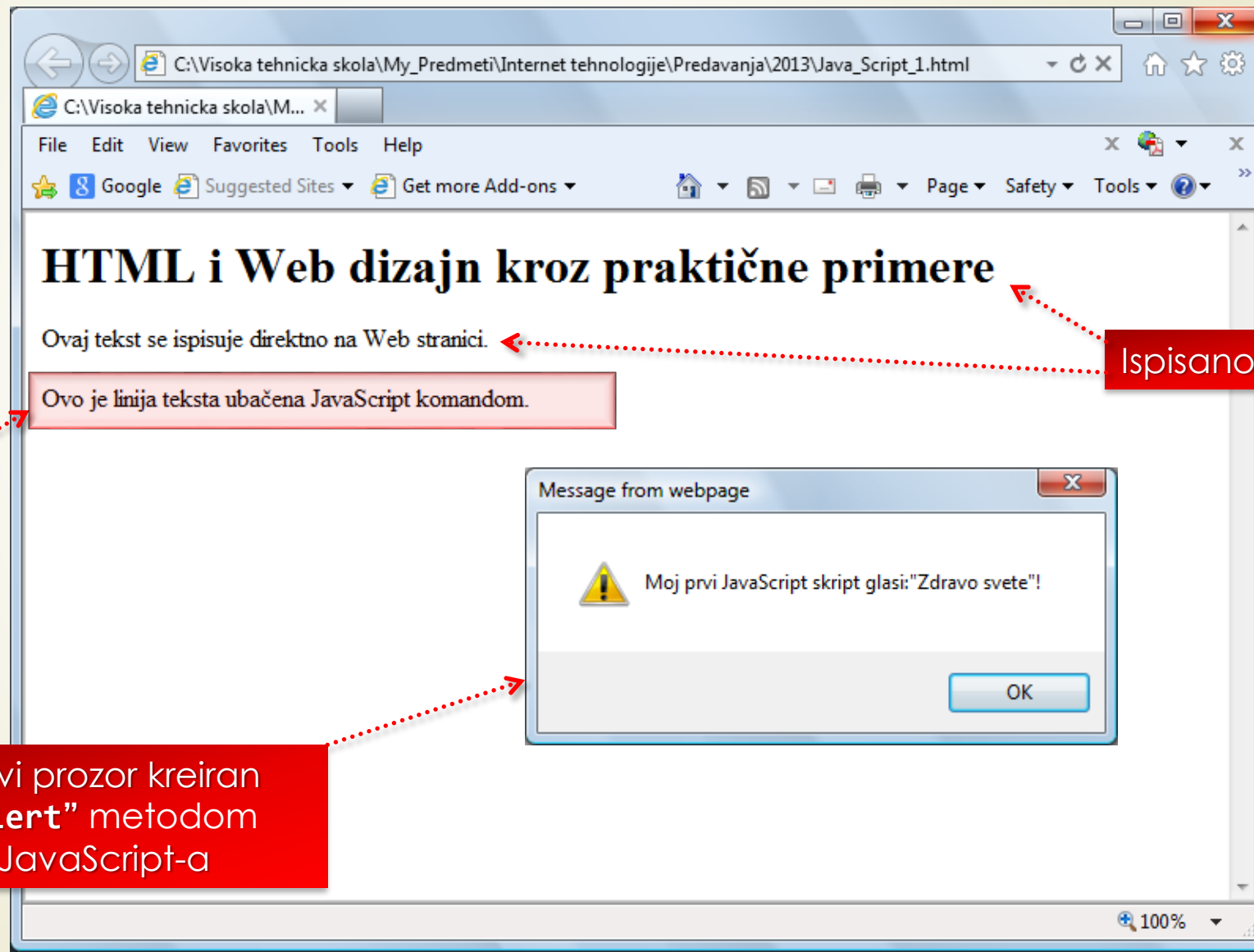
Programski  
kod  
JavaScript-a

# JS objekt document

- Sve redove HTML kôda od oznake **<script>** do završne oznake **</script>** čitač **NE PRIKAZUJE** na ekranu, već ih **IZVRŠAVA!**
- Programski iskaz **document.write()** označava da će čitač **IZVRŠITI METODU** „write()“ povezanu sa **OBJEKTOM** „document“ koji predstavlja TEKUĆU WEB STRANICU (ovaj način upotrebe JavaScript-a je adekvatan njegovom objektno-orijentisanom konceptu).
- Zapravo, JavaScript je **KLIJENTSKI - OBJEKTNO-ORIJENTISANI** programski jezik (podrazumeva mogućnost nasleđivanja i formiranja objekata – novije verzije nasleđivanje se realizuju tzv. **prototipovima**).
- Obzirom da je svaki JavaScript objekt nasto iz neke klase, JavaScript ima SPOSOBNOST **MANIPULACIJE OBJEKTIMA** na Web stranici i objektima **Web čitača**.
- **OSNOVNI OBJEKAT** JavaScript-a je **SAMA WEB STRANICA** (**document**), dok se ostali objekti na Web stranici postavljaju HTML kôdom.
- U te objekte možemo ubrojati: pasuse, tabele, slike, Aplete, ....

# Web stranica i JavaScript

U Web čitaču treba biti **dozvoljen rad** sa JavaScriptom!



Ispisano  
**JavaScript-om**

Novi prozor kreiran  
**"alert"** metodom  
JavaScript-a

Ispisano iz **HTML-a**

# JavaScript i Development tools u Chrome-u

F12

The screenshot shows a web browser window displaying a page titled "HTML i Web dizajn kroz praktične primer". The page content includes a heading and two paragraphs. The first paragraph states "Ovaj tekst se ispisuje DIREKTNO na Web stranici." and the second paragraph states "Ovo je linija teksta ubačenaJavaScript komandom." Below the page content, the Chrome Developer Tools interface is visible, showing the Elements panel with the HTML structure, the Console panel with the JavaScript code, and the Console panel with the output of the code.

```
<!doctype html>
<html>
  <head></head>
  <body>
    <h1>HTML i Web dizajn kroz praktične primer</h1>
    <p>Ovaj tekst se ispisuje DIREKTNO na Web stranici.</p>
    <script language="JavaScript">
      <!--
        document.write("<p>Ovo je linija teksta ubačena
      +"JavaScript komandom.</p>");
      alert ("Moj prvi JavaScript skript glasi:\\"Zdravo svete\\"!");
      <!-->
    </script>
    <p>Ovo je linija teksta ubačenaJavaScript komandom.</p>
  </body>
</html>
```

```
> 1+11
< 12
```

Izgled Web stranice

Meni Chrome Development tools

Izvorni kod Web stranice i JavaScripta

Prozor konzole

# JavaScript funkcije (1)

- ▶ Jedino JavaScript metode MOGU **PRISTUPITI**, a u nekim slučajevima i **IZMENITI ATRIBUTE** pojedinih HTML oznaka, čime direktno **UTIČU NA IZGLED** WEB STARANICE!
- ▶ Programski iskazi JavaScripta se kao i kod savremenih programskih jezika mogu **GRUPISATI U FUNKCIJE** i kao takve se smeštaju u Web dokument.
- ▶ Primer funkcije sa nazivom "**PozdravPosetiocu**" koja prikazuje **ZASEBAN PROZOR** sa porukom "Dobro došli na moju Web stranicu" sledi:

```
function PozdravPosetiocu()  
{  
    alert("Dobro došli na moju Web stranicu");  
    return;  
}
```

Službena reč **function** JavaScript-a.

Službena reč **return** JavaScript-a, vraća upravljanje Web čitaču.

## JavaScript funkcije (2)

- ▶ Primetite, definicija funkcije počinje ključnom rečju „**function**“, zatim sledi **IME FUNKCIJE** i opciono **NIZ PARAMETARA** između zagrada.
- ▶ Nakon **IMENA FUNKCIJE**, sledi izvorni kod između leve i desne vitičaste zagrade

```
{  
    ... // JavaScript iskazi.  
}
```

- ▶ JavaScript iskazi se moraju završavati tačkom i zarezom (;).
- ▶ Na kraju svake funkcije (pre zatvaranja desne vitičaste zagrade „}“), treba umetnuti ključnu reč „**return**“ (u novim verzijama nije neophodna).
- ▶ Iskaz **return** ili kraj programskog koda vraća upravljanje čitača Weba iskazu koji je pozvao funkciju.
- ▶ JavaScript funkcija se prevažodno poziva preko (svog) **IMENA**, a može se naći i kao **ARGUMENT IZRAZA** ili kao iskaz unutar nekog skripta.

# JavaScript funkcije (3)

- ▶ Ako se poziva funkciju **IZ TELA** Web stranice, čitač prekida učitavanje HTML-a i **IZVRŠAVA** iskaze iz JavaScript funkcije.
- ▶ Web čitač se **VRAĆA** do mesta na stranici na kojem je pozvana funkcija kada:
  - ▶ **izvrši** iskaz return, ili
  - ▶ dođe do zatvorene vitičaste zagrade („}“) na kraj JavaScript funkcije.
- ▶ Kao i kod svih programskih jezika, kroz poziv JavaScript funkcije, mogu se **PROSLEDITI** vrednosti promenljivih.
- ▶ Kao kod većine viših programskih jezika, **NIZ PARAMETARA** funkcije nalazi se u zagradama, a posle imena u deklaraciji funkcije.
- ▶ Primer:

`function izracunaj_IQ (Dob, Broj_cipela)`



# Izračunaj\_IQ: JavaScript funkcija

```
function izracunaj_IQ(Dob, Broj_cipela)
{
    Dob = Dob * 2;
    alert ("2 x Dob = " + Dob + "; Broj cipela= " + Broj_cipela);
    return (Dob * Broj_cipela) / 2;
}
```

Povratna vrednost.

➤ Proverite: šta se vraća pozivnoj funkciji u sledećem primeru?

...

```
IQ = izracunaj_IQ(16, 10);
```

➤ Odgovor: 160!



# Anonimne JavaScript funkcije

- Već smo naučili, kroz deklaraciju, funkciji dajemo **IME** koje se kasnije koristi da se **POZOVE** na izvršenje po potrebi.
- Funkciju možemo deklarirati i pomoću **ANONIMNE FUNKCIJE** koja se dodeljuje **PROMENLJIVOJ** - ime promenljive se kasnije koristi za poziv funkcije.
- Anonimne funkcije **NEMAJU** ni **IME** ni **IDENTIFIKATOR**!

```
<script>
  var myvariable = 5;
  var myfunction = function() {
    myvariable = myvariable *2;
  }
  for(var f=0; f<10; f++) {
    myfunction();
  } alert("Vrednost promenljive myvariable je " + myvariable);
</script>
```

Anonimna funkcija dodeljena promenljivoj **myfunction**

Korišćenje promenljive **myfunction** za poziv anonimne funkcije

# JavaScript: službene reči

abstract	continue	finally	instanceof	protected	throws
boolean	default	float	int	public	transient
break	delete	for	interface	return	true
byte	do	function	long	short	try
case	double	goto	native	static	typeof
catch	else	if	new	super switch	var
char	extends	implements	null	synchronized	void
class	false	import	package	this	while
const	final	in	private	throw	with

- Da li prepoznajete karakteristične naredbe za objektno-orijentisane (OO) programske jezike?

# JavaScript: tipovi promenljivih

- ▶ JavaScript određuje **TIP PROMENLJIVE** na osnovu **PODATKA** koji se smešta u promenljivu (**NIJE NEOPHODNA DEKLARACIJA** tipa promenljive).
- ▶ Tip promenljive **MOŽE DA SE PROMENI**, zavisno od operacije – konteksta u kojoj koristite promenljivu!
- ▶ U nazivima i rezervisanim rečima JavaScripta **TREBA** voditi računa o **VELIČINI SLOVA** (“**case sensitive**”).
- ▶ Aritmetički JavaScript (binarni) **OPERATORI** koji zahtevaju dve promenljive su: **+**, **-**, **\***, **/**, **%**.
- ▶ Operatori dvostruki plus (**++**) i dvostruki minus (**--**) se stavljaju ispred/iza promenljive, kao **PREFIKS/POSTFIX**, tako da se operacija obavlja **PRE/POSLE** dodele vrednosti:
- ▶ **i = 10; j = i++; j = --i ;**

Odgovori: (j=10, i=11) (j=9)

# JavaScript: operatori poredjenja

Operator	Namena	Primer
<	Manje od	<code>izraz &lt; izraz</code>
>	Veće od	<code>izraz &gt; izraz</code>
<=	Manje ili jednako	<code>izraz &lt;= izraz</code>
>=	Veće ili jednako	<code>izraz &gt;= izraz</code>
==	Jednako	<code>izraz == izraz</code>
!=	Različito od	<code>izraz != izraz</code>
?:	Uslovni (if-else) operator	<code>(logički izraz) ? izraz TRUE : izraz FALSE</code>

# JavaScript: logički operatori

Operator	Namena	Primer
&&	Logicko I	izraz && izraz
	Logicko II	izraz    izraz

➤ Operator **KOMPLEMENT** (negacija) - znak uzvika (!) - vraća negaciju logičke vrednosti promenljive (tačno ili pogrešno) pri čemu se **NE MENJA** vrednost promenljive.

➤ Primer:

```
TestResult = true; // istina, tačno
```

```
document.write(!TestResult); // netačno, laž
```

# Uslovno izvršavanje i petlje

```
function check18( )
{
    var Age = 25;
    if (Age >= 18)
        alert("Reg. www.vtsnis.edu.rs");
    return;
}
```

```
function OdJedanDoDeset( )
{
    var Count;
    for (Count = 1; Count <= 10 Count++)
        alert("Count = " + Count);
}
```

```
function Proveri_18( )
{
    var Age = 15;
    if (Age >= 18)
        alert("Nemojte zaboraviti da glasate!");
    else
        alert("Učite vredno!");
    return;
}
```

```
function getPassword( )
{
    var Password = ""; //Inic.vr. value!= "Secret"
    while (Password != "Secret")
        Password = prompt("Password", "");
    alert("You're in!");
}
```

# Obrada događaja u JavaScriptu

- ▶ **SPECIJALNI ATRIBUTI** HTML oznaka se koriste za **POVEZIVANJE DOGAĐAJA** na Web stranici i JavaScript iskaza (metoda) koji ih obrađuju.
- ▶ Ako se želi POZIV i IZVRŠENJE JavaScript funkcije u Web čitaču na **POJAVU NEKOG DOGAĐAJA**, onda se funkcija mora **POVEZATI** sa određenim **ATRIBUTOM OBRADU DOGAĐAJA**.
- ▶ Sledeća lista prikazuje skup **SPECIJALNIH ATRIBUTA** koji se odnose na **OBRADU DOGAĐAJA** u JavaScriptu.
- ▶ Evo izvoda iz spiska **ATRIBUTA DOGAĐAJA** koji su vezani za Web stranicu:

**onAbort, onBlur, onChange, onClick, onDoubleClick, onDragDrop, onError, onFocus, onKeyDown, onMouseUp, onMove, onReset, onResize, onSelect, onSubmit, OnUnload, OnKeyPress, onKeyUp, onLoad, onMouseDown, onMouseOut, onMouseOver, onMouseUp, ....**

# Atributi obrada događaja u JavaScriptu (1)

Atribut O.D.	Od događaja nastaje kada
onAbort	Posetilac prekine učitavanje slike (na primer, pritiskanjem hiperveze ili dugmeta Stop čitača). Oznaka: <code>&lt;img&gt;</code> .
onBlur	Prozor, okvir ili element obrasca izade iz fokusa. Oznake: <code>&lt;body&gt;</code> , <code>&lt;frame&gt;</code> , <code>&lt;frameset&gt;</code> , <code>&lt;input&gt;</code> , <code>&lt;select&gt;</code> .
OnChange	Posetilac menja vrednost u elementu obrasca kao što je jednorodno ili višeredno polje za unos teksta, ili bira novi element u padajućoj listi. Oznake: <code>&lt;input&gt;</code> , <code>&lt;select&gt;</code> .
OnClick	Posetilac pritisne hipervezu ili element obrasca kao što je polje za potvrdu, dugme, radio-dugme, dugme Reset ili dugme Submit. Oznake: <code>&lt;a&gt;</code> , <code>&lt;area&gt;</code> , <code>&lt;body&gt;</code> , <code>&lt;input&gt;</code> .
onDoubleClick	Posetilac dvaput pritisne hipervezu ili element obrasca kao što je polje za potvrdu, dugme, radio-dugme, dugme Reset ili dugme Submit. Oznake: <code>&lt;a&gt;</code> , <code>&lt;area&gt;</code> , <code>&lt;body&gt;</code> , <code>&lt;input&gt;</code> .
onDragDrop	Posetilac povlači objekat i pušta ga u prozor čitača Weba. Oznake: <code>&lt;body&gt;</code> , <code>&lt;frame&gt;</code> , <code>&lt;frameset&gt;</code> .
onError	Čitač naiđe na grešku kada učitava sliku ili Web stranicu. Oznake: <code>&lt;body&gt;</code> , <code>&lt;frame&gt;</code> , <code>&lt;frameset&gt;</code> , <code>&lt;img&gt;</code> .



## Atributi obrada događaja u JavaScriptu (2)

Atribut O.D.	Dodogađaja nastaje kada:
onMouseUp	Posetilac otpusti taster miša. Oznake: <code>&lt;a&gt;</code> , <code>&lt;body&gt;</code> , <code>&lt;input&gt;</code> .
onMove	Posetilac ili skript pomere prozor čitača. Oznake: <code>&lt;body&gt;</code> , <code>&lt;frame&gt;</code> , <code>&lt;frame set&gt;</code> .
onReset	Posetilac pritisne dugme Reset u obrascu. Oznaka: <code>&lt;form&gt;</code> .
onResize	Posetilac ili skript promene veličinu prozora čitača. Oznake: <code>&lt;body&gt;</code> , <code>&lt;frame&gt;</code> , <code>&lt;frameset&gt;</code> .
onSelect	Posetilac izabere tekst u višerednom ili jednorednom polju za unos teksta. Oznaka: <code>&lt;input&gt;</code> .
onSubmit	Posetilac pritisne dugme Submit u obrascu. Oznaka: <code>&lt;form&gt;</code> .
OnUnload	Pre nego što čitač zameni jednu Web stranicu drugom. Oznake: <code>&lt;body&gt;</code> , <code>&lt;frame&gt;</code> , <code>&lt;frameset&gt;</code> .
onFocus	Prozor ili element obrasca uđe u fokus. Oznake: <code>&lt;body&gt;</code> , <code>&lt;frame&gt;</code> , <code>&lt;frameset&gt;</code> , <code>&lt;input&gt;</code> , <code>&lt;select&gt;</code> .
onKeyDown	Posetilac pritisne taster na tastaturi. Oznake: <code>&lt;a&gt;</code> , <code>&lt;body&gt;</code> , <code>&lt;frame&gt;</code> , <code>&lt;frameset&gt;</code> , <code>&lt;img&gt;</code> , <code>&lt;input&gt;</code> .

## Atributi obrada događaja u JavaScriptu (3)

Atribut O.D.	Dodogađaja nastaje kada:
OnKeyPress	Posetilac pritisne i drži pritisnut taster u polju za unos teksta elementa obrasca. Oznake: <a>, <body>, <frame>, <frameset>, <img>, <input>.
onKeyUp	Posetilac otpusti taster u polju za unos teksta elementa obrasca. Oznake: <a>, <body>, <frame>, <frameset>, <img>, <input>.
onLoad	Čitač učitava Web stranicu ili sliku. Oznake: <body>, <frame>, <frameset>, <img>.
onMouseDown	Posetilac pritisne taster miša. Oznake: <a>, <body>, <input>.
onMouseOut	Posetilac pomeri pokazivač miša van područja mapirane slike ili sa hiperveze. Oznake: <a>.
onMouseOver	Posetilac pomeri pokazivač miša u područje mapirane slike ili na hipervezu. Oznake: <a>.
onMouseUp	Posetilac otpusti taster miša. Oznake: <a>, <body>, <input>.

Pogledajte HTML 5 događaje na:

[https://www.tutorialspoint.com/javascript/javascript\\_events.htm](https://www.tutorialspoint.com/javascript/javascript_events.htm)

# Pridruživanje događaja HTML oznaci

- ▶ Da bi se nekoj JavaScript funkciji **PRIDRUŽIO** neki događaj potrebno je da se:
  1. Odabere odgovarajući **ATRIBUT OBRADJE DOGAĐAJA**.
  2. Definiše se **KOJE OBJEKTE** Web stranice atribut obrade događaja treba da prati.
  3. Atribut obrade događaja se **UMEĆE** (npr. **onClick**) u **HTML** oznaku koja izaziva događaj (**<input>**, **<button>**, ...), a zatim **ZADATI METODU** za obradu događaja **onClick** atributu.
- ▶ Primer: Kada posetilac pritisne dugme, atribut obrade događaja zahteva od čitača Weba da izvrši funkciju **youClickedIt()** kao u sledećem primeru:

```
<form> <input type="button" value="Pitisni Me!" onClick="youClickedIt()"> </form>
```

# Događaj onMouseover

- ▶ Ako se hoće da čitač pozove funkciju kada posetilac **POMERI POKAZIVAČ MIŠA PREKO HIPERVEZE**, treba umetnuti atribut obrade događaja **onMouseover** u oznaku hiperveze **<a>**:



```
<a href= "http://www.vtsnis.edu.rs" onMouseover = "youAreOnIt()">  
Internet tehnologije.com </a>
```

- ▶ U JavaScriptu na raspolaganju su **KORISNIČKI** definisane i **UGRAĐENE** METODE.
- ▶ OO model JavaScript-a u HTML-u podrazumeva sledeće OBJEKTE: sama Web **stranica** (**document**), **obrazac**, **dugme** ili **hiperveza** na njoj.
- ▶ Pozivanje metode nekog objekta vrši se preko **IMENA OBJEKTA** iza čega sledi operator tačka (**.**) a onda ime metode, primer:

```
document.write("Poruka");
```

# Izmrna HTML koda write metodom

- ▶ Kada se koristi metoda **document.write()** radi ispisa teksta na Web stranici, obavezno treba uključiti **HTML OZNAKE** za formatiranje izgleda.
- ▶ Da bi se prelomio dugačak tekst, treba podeliti tekst na nekoliko znakovnih nizova (celina) i nadovezati **OPERATOROM "+"**:



```
document.write ("<p> <b>Welcome to the Web page" +  
"with my first + JavaScriptScript.</b></p>");
```

HTML oznake za  
formatiranje teksta

# \*Poziv funkcije JavaScripta (1)

- Pozivanje KORISNIČKE JavaScript funkcije - funkcije koje je definisao korisnik pozivaju se navođenjem imena funkcije, **BEZ ZADAVANJA IMENA OBJEKTA!**

```
<script>
```

```
<! --
```

```
//ispiši red teksta na Web stranici
```

```
document.write("<p>Click the mouse on the OK button to continue.</p>")
```

```
//pozovi korisničku JavaScript funkciju "helloWorld"– NEMA navođenja objekta
```

```
helloWorld();
```

```
// -->
```

```
</script>
```

## \*Poziv funkcije JavaScripta (2)

- Kad čitač naiđe na ključnu reč "**function**" u skriptu, očekuje **DEKLARACIJU FUNKCIJE**, a ne iskaz za izvršavanje!
- Čitač učitava u **MEMORIJU** ime funkcije, parametre i iskaze funkcije.
- Funkcija se poziva preko njenog **IMENA** u iskazu skripta koji se umeće između oznake za skript u telu Web stranice.
- Već je pokazano da čitač može da poziva funkcije na **IZVRŠENJE** zavisno od **DOGAĐAJA** (pritiskanje hiperveze) ili **ELEMENTA** (polje za unos).
- Ukoliko se deklariraju sve funkcije u **ZAGLAVLJU DOKUMENTA**, one će biti **UČITANE U MEMORIJU PRE** nego što ih čitač prvi put pozove iz tela dokumenta.
- Ovo će **UBRZATI RAD** JavaScript funkcija jer će sve biti **PRVO UČITANE**, a tek onda pozivane na izvršenje.

## \*Poziv funkcije JavaScripta (3)

- ▶ Pozivanje JavaScript funkcija iz atributa za obradu događaja se obavlja na sledeći način:
  - ▶ Čitač Weba **AUTOMATSKI IZVRŠAVA** JavaScript funkciju kao **ODZIV** na neki događaj.
- ▶ Primer: ako se želi da čitač Weba "uradi nešto" pre nego što ukloni Web stranicu iz svoje memorije, oznaci **<body>** u Web stranicu treba dodati atribut "**onUnload**".
- ▶ Evo primera za oznaku body:

```
<body onUnload = "alert ("Hvala na poseti!")">
```

- ▶ Nedostatak ovog koncepta poziva JavaScript metoda je **PROGRAMSKI MIX** načinjen od **HTML-a** i **JavaScript-a**, što se protivi savremenom konceptu Web programiranja i **RAZDVAJANJU SADRŽAJA OD IZGLEDA!**



# \*Poziv funkcije JavaScripta (4)

- ▶ Da li se funkcija **doSomething()** može ponovo koristiti na nekim drugim Web stranicama?

```
<input id = "doSomethingButton" type="button" value="Do Something"  
      onclick="javascript:doSomething();" />
```

```
<script type="text/javascript">  
  function doSomething() {  
    var x = 5;  
    var y = 3;  
    var z = x * y;  
    alert(z);  
  }  
</script>
```

Pozivanje metoda  
**doSomething**  
odgovarajućim atributom  
događaja iz HTML-a

JavaScript  
metoda  
**doSomething**

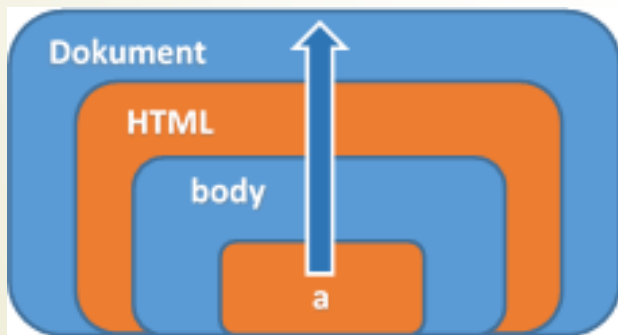
# Implementacija događaja: `addEventListener`

- ▶ Metoda `addEventListener()` je NOVI NAČIN za **IMPLEMENTACIJU DOGAĐAJA** u JavaScriptu koja se pojavljuje sa **HTML5**.
- ▶ Dodavanjem specifične metode `addEventListener()` se sprečava programski **MIX** različitih jezika.
- ▶ Sintaksa ove metode je sledeća:

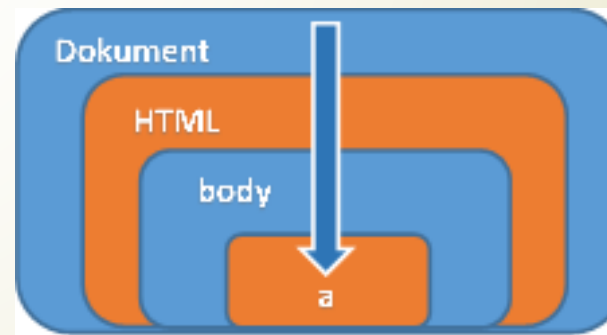


# \*Opsluživanje događaja u JavaScriptu

- ▶ Treći parametar „**aktiviranje**“ je opcioni i predstavlja Bulovu promenljivu kojom se definiše način opsluživanja događaja kod **UGNEŽDENIH** HTML elemenata.
- ▶ U osnovi se razlikuju dva načina opsluživanja ugnežđenih događaja: engl. **Capturing** i engl. **Bubbling**.
- ▶ Ako treći parametar uzme vrednost "**true**", obrađivač događaja se izvršava u "**Capturing**" fazi.
- ▶ Podrazumevana vrednost ovog parametra je "**false**", tako da se obrađivač događaja izvršava u "**Bubbling**" fazi.



„Bubbling“ faza



„Capturing“ faza

# \*jQuery i JavaScript

- **jQuery** je jedna od najpopularnijih JavaScript **BIBLIOTEKA** koja se jednostavno može integrisati u Web stranicu.
- Za pristup JavaScript funkcijama koje se nalaze u jQuery biblioteci neophodno je **UKLJUČITI U PROJEKAT** na sledeći način:

```
<script src="~/Scripts/jquery-1.9.1.min.js"> </script>
```

- **POZIV DOGAĐAJA** se u jQuery-ju može obaviti na sledći način:

```
$(document).ready(function() {  
    // ... Dodaj JavaScript kod ovde  
});
```

Ili samo:

```
$(function() {  
    // ... Dodaj JavaScript kod ovde  
});
```