



Интернет програмирање предавање 05

Проф. др Мирослав Лутовац
mlutovac@viser.edu.rs



while petlja

- while petlja funkcioniše tako što se blok instrukcija unutar petlje ponovljeno izvršava sve dok je uslov za ostanak u petlji ispunjen
- uslov se nalazi na ulasku u petlju
- while (uslov) {
 blok instrukcija unutar petlje
}

Kod u petlji se izvršava ponovo i ponovo,
sve dok je uslov istinit



i1 = 0 + 2;
i2 = ... + 4;

petlja se izvršava dok je i1 < i2

primer09a1.js

```
1 var mojiPrijatelji = ["Nenad", "Marko", "Branko",  
2   "Milan", "Predrag", "Zoran", "Jovan", "Dragan"];  
3 var pozivPrijateljima = "Na zabavu su pozvani "  
4 var ipocetak = 0;  
5 var ikraj = mojiPrijatelji.length;  
6  
7 document.write(pozivPrijateljima);  
8 var i1 = ipocetak+2;  
9 var i2 = ikraj+4;  
10 while (i1 < i2)  
11 {  
12     document.write("<br>", i1, " ", mojiPrijatelji[i1]);  
13     i1++;  
14     // if (typeof mojiPrijatelji[i1]=="undefined") {break;}  
15     // if (i1>9) {break;}  
16 }
```

Na zabavu su pozvani

2 Branko

3 Milan

4 Predrag

5 Zoran

6 Jovan

7 Dragan

8 undefined

9 undefined

10 undefined

11 undefined

Ako se zaboravi promena uslova u petlji, petlja se nikada neće završiti



```
i1 = 0 + 2;  
i2 = ... + 4;  
if (i1>9) {break;}
```

Prekid u petlji i kada je $i1 > 9$

primer09a2.js

```
1 var mojiPrijatelji = ["Nenad", "Marko  
2   "Milan", "Predrag", "Zoran", "Jova  
3 var pozivPrijateljima = "Na zabavu su  
4 var ipocetak = 0;  
5 var ikraj = mojiPrijatelji.length;  
6  
7 document.write(pozivPrijateljima);  
8 var i1 = ipocetak+2;  
9 var i2 = ikraj+4;  
10 while (i1 < i2)  
11 {  
12     document.write("<br>", i1, " ", mojiPrijatelji[i1]);  
13     i1++;  
14     // if (typeof mojiPrijatelji[i1]=="undefined") {break;}  
15     if (i1>9) {break;}  
16 }
```

Na zabavu su pozvani

2 Branko

3 Milan

4 Predrag

5 Zoran

6 Jovan

7 Dragan

8 undefined

9 undefined



Prekid u petlji i kada je ...

if (typeof mojiPrijatelji[i1]=="undefined")

primer09a3.js

```
1 var mojiPrijatelji = ["Nenad", "Marko", "Branko",  
2   "Milan", "Predrag", "Zoran", "Jovan", "Dragan"];  
3 var pozivPrijateljima = "Na zabavu su pozvani "  
4 var ipocetak = 0;  
5 var ikraj = mojiPrijatelji.length;  
6  
7 document.write(pozivPrijateljima);  
8 var i1 = ipocetak+2;  
9 var i2 = ikraj+4;  
10 while (i1 < i2)  
11 {  
12     document.write("<br>",i1," ",mojiPrijatelji[i1]);  
13     i1++;  
14     if (typeof mojiPrijatelji[i1]=="undefined") {break;}  
15     if (i1>9) {break;}  
16 }
```

Na zabavu su pozvani
2 Branko
3 Milan
4 Predrag
5 Zoran
6 Jovan
7 Dragan



Važan je i redosled u petlji,
pre ili posle promene uslova

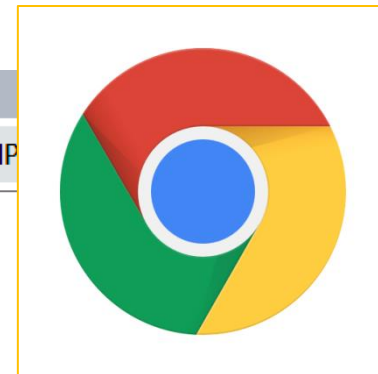
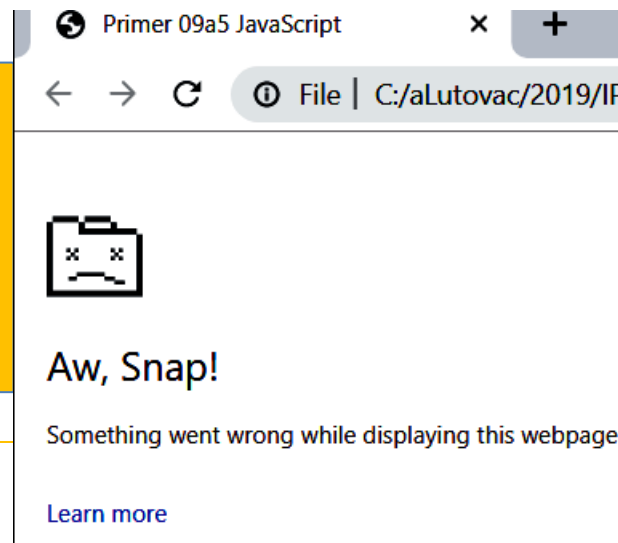
primer09a4.js

```
1 var mojiPrijatelji = ["Nenad", "Marko", "Branko",  
2   "Milan", "Predrag", "Zoran", "Jovan", "Dragan"];  
3 var pozivPrijateljima = "Na zabavu su pozvani "  
4 var ipocetak = 0;  
5 var ikraj = mojiPrijatelji.length;  
6  
7 document.write(pozivPrijateljima);  
8 var i1 = ipocetak+2;  
9 var i2 = ikraj+4;  
10 while (i1 < i2)  
11 {  
12     document.write("<br>", i1, " ", mojiPrijatelji[i1]);  
13     if (typeof mojiPrijatelji[i1] == "undefined") {break;}  
14     i1++;  
15     if (i1 > 9) {break;}  
16 }  
17
```

Na zabavu su pozvani
2 Branko
3 Milan
4 Predrag
5 Zoran
6 Jovan
7 Dragan
8 undefined



Ako se zaboravi
promena uslova u
petlji, **petlja se nikada
neće završiti**



```
10 while (i1 < i2)
11 {
12     document.write("<br>", i1, " ", mojiPrijetelji[i1]);
13     // i1++;
14     if (typeof mojiPrijetelji[i1] == "undefined") {break;}
15     if (i1 > 9) {break;}
16 }
```



Page Unresponsive

You can wait for it to become responsive or exit the page.

Primer 09a5 JavaScript

Wait

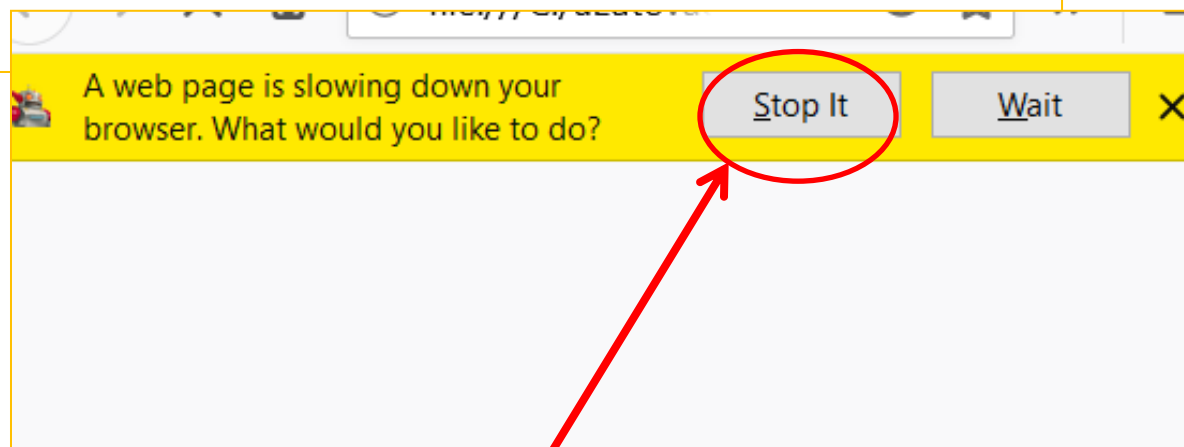
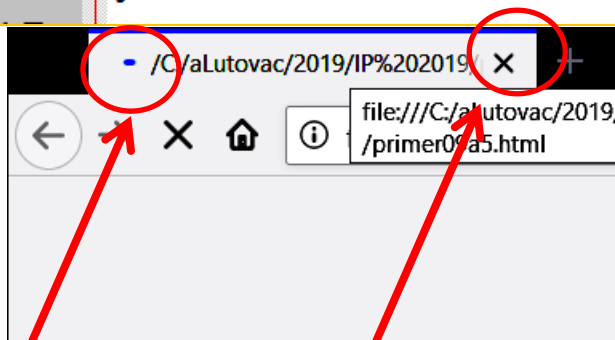
Exit page



Ako se zaboravi
promena uslova u
petlji, **petlja se nikada
neće završiti**



```
10 while (i1 < i2)
11 {
12     document.write("<br>", i1, " ", mojiPrijetelji[i1]);
13     // i1++;
14     if (typeof mojiPrijetelji[i1] == "undefined") {break;}
15     if (i1 > 9) {break;}
16 }
```





do-while petlja

- do-while petlja funkcionije tako što se blok instrukcija unutar petlje ponovljeno izvršava sve dok je uslov za ostanak u petlji ispunjen
- uslov se nalazi na kraju petlje
- ```
do {
 blok instrukcija unutar petlje
}
while (uslov)
```



**while i do**

$i1 = 0 + 2;$   
 $i2 = \dots + 4;$

petlja se izvršava dok je  $i1 < i2$

primer09a.js x

```
1 var mojiPriatelji = ["Nenad", "Marko", "Branko",
2 "Milan", "Predrag", "Zoran", "Jovan", "Dragan"];
3 var pozivPriateljima = "Na zabavu s
4 var ipocetak = 0;
5 var ikraj = mojiPriatelji.length;
6
7 document.write(pozivPriateljima);
8 var i1 = ipocetak+2;
9 var i2 = ikraj+4;
10 do {
11 document.write("
", i1, " ", mojiPriatelji[i1]);
12 i1++;
13 if (typeof mojiPriatelji[i1]=="undefined") {break;}
14 if (i1>9) {break;}
15 }
16 while (i1 < i2)
```

Na zabavu su pozvani  
2 Branko  
3 Milan  
4 Predrag  
5 Zoran  
6 Jovan  
7 Dragan



**while i do**

**petlja se izvršava barem jednom**

**i1 = 0 + 2;  
i2 = ... -10;**

primer09a6.js

```
1 var mojiPriatelji = ["Nenad", "Marko", "Branko",
2 "Milan", "Predrag", "Zoran", "Jovan", "Dragan"];
3 var pozivPriateljima = "Na zabavu su pozvani "
4 var ipocetak = 0;
5 var ikraj = mojiPriatelji.length;
6
7 document.write(pozivPriateljima);
8 var i1 = ipocetak+2;
9 var i2 = ikraj-10;
10 do {
11 document.write("
", i1, " ", mojiPriatelji[i1]);
12 i1++;
13 if (typeof mojiPriatelji[i1] == "undefined") {break;}
14 if (i1 > 9) {break;}
15 }
16 while (i1 < i2)
```

**Na zabavu su pozvani  
2 Branko**



## while petlja se koristi da se prikažu svi članovi niza

primer09b2.js

```
1 var mojiPrijetelji = ["Nenad", "Marko", "Branko",
2 "Milan", "Predrag", "Zoran", "Jovan", "Dragan"];
3 var pozivPrijeteljima = "Na zabavu su pozvani "
4 var i = 0;
5
6 document.write(pozivPrijeteljima);
7 while (mojiPrijetelji[i]) {
8 document.write("
", i, " ", mojiPrijetelji[i]);
9 i++;
10 }
11
12 /*
13 for (;mojiPrijetelji[i-1];) {
14 while (mojiPrijetelji[i]) {
15 document.write("
", i, " ", mojiP
16 i++;
17 }
18 */
```

while ili for?

Sve dok ima članova niza,  
izvršava se petlja

Na zabavu su pozvani  
0 Nenad  
1 Marko  
2 Branko  
3 Milan  
4 Predrag  
5 Zoran  
6 Jovan  
7 Dragan

# while ili for?



primer09b.js

```
1 var mojiPrijatelji = ["Nenad", "Marko", "Branko",
2 "Milan", "Predrag", "Zoran", "Jovan", "Dragan"];
3 var pozivPrijateljima = "Na zabavu su pozvani "
4 var i = 0+1;
5
6 document.write(pozivPrijateljima);
7 while (mojiPrijatelji[i+2]) {
8 document.write("
", i, " ", mojiPrijatelji[i]);
9 i++;
10 }
11
12 /*
13 for (;mojiPrijatelji[i-1];) {
14 while (mojiPrijatelji[i]) {
15 document.write("
", i, " ", mojiPrijatelji[i]);
16 i++;
17 }
18 */
```

Na zabavu su pozvani  
1 Marko  
2 Branko  
3 Milan  
4 Predrag  
5 Zoran



**for** ili **while**?

petlja počinje od 1, završava se sa 4

Na zabavu su pozvani  
1 Marko  
2 Branko  
3 Milan  
4 Predrag

primer09b3.js x

```
1 var mojiPrijatelji = ["Nenad", "Marko", "Branko",
2 "Milan", "Predrag", "Zoran", "Jovan", "Dragan"];
3 var pozivPrijateljima = "Na zabavu su pozvani "
4 var i = 0+1;
5
6 document.write(pozivPrijateljima);
7 for (;mojiPrijatelji[i+3];) {
8 document.write("
",i," ",mojiPrijatelji[i]);
9 i++;
10 }
```



**for** ili **while**?

petlja počinje od 1, završava se sa 8

Na zabavu su pozvani  
1 Marko  
2 Branko  
3 Milan  
4 Predrag  
5 Zoran  
6 Jovan  
7 Dragan  
8 undefined

primer09b4.js x

```
1 var mojiPrijatelji = ["Nenad", "Marko", "Branko",
2 "Milan", "Predrag", "Zoran", "Jovan", "Dragan"];
3 var pozivPrijateljima = "Na zabavu su pozvani "
4 var i = 0+1;
5
6 document.write(pozivPrijateljima);
7 for (;mojiPrijatelji[i-1];) {
8 document.write("
",i," ",mojiPrijatelji[i]);
9 i++;
10 }
```



for ili **while**?

petlja počinje od 1, završava se sa 5

Na zabavu su pozvani

1 Marko

2 Branko

3 Milan

4 Predrag

5 Zoran

primer09b5.js

```
1 var mojiPriatelji = ["Nenad", "Marko", "Branko",
2 "Milan", "Predrag", "Zoran", "Jovan", "Dragan"];
3 var pozivPriateljima = "Na zabavu su pozvani "
4 var i = 0+1;
5
6 document.write(pozivPriateljima);
7 for (;mojiPriatelji[i+2];) {
8 document.write("
",i," ",mojiPriatelji[i]);
9 i++;
10 }
```

# break i continue

- break se koristi da se izađe iz petlje
- continue se koristi da se prskoči jedna iteracija

## Page Unresponsive

You can wait for it to become responsive or exit the page.

Primer 09c JavaScript

Wait

Exit page



## Aw, Snap!

Something went wrong while displaying this webpage.

[Learn more](#)

Reload

if (i==5) {continue;}

primer09c.js

```
1 var mojiPrijatelji = ["Nenad", "Marko", "Branko",
2 "Milan", "Predrag", "Zoran", "Jovan", "Dragan"];
3 var pozivPrijateljima = "Na zabavu su pozvani "
4 var i = 0;
5
6 document.write(pozivPrijateljima);
7 for (;mojiPrijatelji[i];) {
8 if (i==5) {continue;}
9 document.write("
",i," ",mojiPrijatelji[i]);
10 i++;
11 }
```





if (i==5) {continue;}

Na zabavu su pozvani  
0 Nenad  
1 Marko  
2 Branko  
3 Milan  
4 Predrag  
6 Jovan  
7 Dragan

primer09c2.js

```
1 var mojiPriatelji = ["Nenad", "Marko", "Branko",
2 "Milan", "Predrag", "Zoran", "Jovan", "Dragan"];
3 var pozivPriateljima = "Na zabavu su pozvani "
4 var i = 0;
5
6 document.write(pozivPriateljima);
7 for (;mojiPriatelji[i];i++) {
8 if (i===5) {continue;}
9 document.write("
",i," ",mojiPriatelji[i]);
10 //i++;
11 }
```





Na zabavu su pozvani  
0 Nenad  
1 Marko  
2 Branko  
3 Milan  
4 Predrag

if (i==5) {break;}

primer09c3.js x

```
1 var mojiPriatelji = ["Nenad", "Marko", "Branko",
2 "Milan", "Predrag", "Zoran", "Jovan", "Dragan"];
3 var pozivPriateljima = "Na zabavu su pozvani "
4 var i = 0;
5
6 document.write(pozivPriateljima);
7 for (;mojiPriatelji[i];i++) {
8 if (i===5) {break;}
9 document.write("
",i," ",mojiPriatelji[i]);
10 //i++;
11 }
```



```
//if (i==5) {break;}
```

Na zabavu su pozvani

0 Nenad

1 Marko

2 Branko

3 Milan

4 Predrag

5 Zoran

6 Jovan

7 Dragan

primer09c4.js

```
1 var mojiPrijetelji = ["Nenad", "Marko", "Branko",
2 "Milan", "Predrag", "Zoran", "Jovan", "Dragan"];
3 var pozivPrijeteljima = "Na zabavu su pozvani "
4 var i = 0;
5
6 document.write(pozivPrijeteljima);
7 for (;mojiPrijetelji[i];i++) {
8 //if (i==5) {break;}
9 document.write("
",i," ",mojiPrijetelji[i]);
10 }
```

# JavaScript label

- label se koristi sa simbolim dvotačka:

**label:**

**statements**

- break i continue mogu da imaju label za izlazak iz bloka

**break labelname;**

**continue labelname;**

# JavaScript label

- **continue** se koristi da se preskoči jedna iteracija u petlji (sa ili bez **label**)
- **break** može da se koristi da se izađe iz petlje ili **switch** (sa ili bez **label**)
- **break** može da se koristi da se izađe iz bilo kog bloka
- Blok koda je uvek između zagrada { i }



Ne radi

continue samo u petlji

primer09d.js

```
1 var mojiPriatelji = ["Nena",
2 "Milan", "Predrag", "Zoran", "Ovan", "Dragan"],
3 var pozivPriateljima = "Na zabavu su pozvani "
4 //var i = 0;
5 document.write(pozivPriateljima);
6 mojaOznaka: {
7 i=0;document.write("
",i," ",mojiPriatelji[i]);
8 i=1;document.write("
",i," ",mojiPriatelji[i]);
9 i=2;document.write("
",i," ",mojiPriatelji[i]);
10 continue mojaOznaka; ←
11 i=3;document.write("
",i," ",mojiPriatelji[i]);
12 i=4;document.write("
",i," ",mojiPriatelji[i]);
13 i=5;document.write("
",i," ",mojiPriatelji[i]);
14 }
15 //break mojaOznaka;
16
15 break mojaOznaka;
```

Elements Console » 1

top Filter Def: ⚙

✖ Uncaught SyntaxError: Illegal `continue` statement: 'mojaOznaka' does not denote an iteration statement

>

Console What's New



mojaOznaka: { ... }



Primer 09d3 JavaScript



File | C:/aLutova

Na zabavu su pozvani

0 Nenad

1 Marko

2 Branko

primer09d3.js

```
1 var mojiPrijatelji = ["Nenad", "Marko", "Branko",
2 "Milan", "Predrag", "Zoran", "Jovan", "Dragan"];
3 var pozivPrijateljima = "Na zabavu su pozvani "
4 //var i = 0;
5 document.write(pozivPrijateljima);
6 mojaOznaka: {
7 i=0;document.write("
",i," ",mojiPrijatelji[i]);
8 i=1;document.write("
",i," ",mojiPrijatelji[i]);
9 i=2;document.write("
",i," ",mojiPrijatelji[i]);
10 break mojaOznaka;
11 i=3;document.write("
",i," ",mojiPrijatelji[i]);
12 i=4;document.write("
",i," ",mojiPrijatelji[i]);
13 i=5;document.write("
",i," ",mojiPrijatelji[i]);
14 }
```

# Primer



```

1 <!DOCTYPE html>
2 <html>
3 <head> <title>Primer 09e JavaScript</title> </head>
4 <body>
5 <input id="myInput" type="text" value="Branko">
6 <button onclick="proveriPrijatelja()">Proveri prijatelja</button>
7 <p id="demo"></p>
8 <script language="JavaScript" src="primer09e.js">
9 </script>
10 </body>
11 </html>

```

Primer 09e JavaScript x +

← → ↻ ⓘ File | C:/aLutovac/2019/

Branko Proveri prijatelja

```

1 function proveriPrijatelja() {
2 var tekst;
3 var mojPrijatelj = document.getElementById("myInput").value;
4 switch(mojPrijatelj) {
5 case "Nenad":
6 tekst = "Moj prijatelj"; break;
7 case "Marko":
8 tekst = "Jovanov prijatelj"; break;
9 case "Branko":
10 tekst = "Anin prijatelj"; break;
11 default:
12 tekst = "nepoznato ko je";
13 }
14 document.getElementById("demo").innerHTML = tekst;
15 }

```

Branko Proveri prijatelja

Anin prijatelj

```
1 function proveriPrijatelja() {
2 var tekst;
3 var mojPrijatelj = document.getElementById("myInput").value;
4 switch(mojPrijatelj) {
5 case "Nenad":
6 tekst = "Moj prijatelj"; break;
7 case "Marko":
8 tekst = "Jovanov prijatelj"; break;
9 case "Branko":
10 tekst = "Anin prijatelj"; break;
11 default:
12 tekst = "nepoznato ko je";
13 }
14 document.getElementById("demo").innerHTML = tekst;
15 }
```

Moj prijatelj

Jovanov prijatelj

nepoznato ko je





```
1 <!DOCTYPE html>
2 <html>
3 <head> <title>Primer 09f JavaScript</title> </head>
4 <body>
5 <input id="myInput" type="text" value="Branko">
6 <button onclick="proveriPrijetelja()">Proveri prijatelja</button>
7 <p id="demo"></p>
8 <script>
9 function proveriPrijetelja() {
10 var tekst;
11 var mojPrijetelj = document.getElementById("myInput").value;
12 switch(mojPrijetelj) {
13 case "Nenad":
14 tekst = "Moj prijatelj"; break;
15 case "Marko":
16 tekst = "Jovanov prijatelj"; break;
17 case "Branko":
18 tekst = "Anin prijatelj"; break;
19 default:
20 tekst = "nepoznato ko je";
21 }
22 document.getElementById("demo").innerHTML = tekst;
23 }
24 </script>
25 </body>
26 </html>
```

Nenad

Moj prijatelj

Marko

Jovanov prijatelj

Ivan

nepoznato ko je

Branko

Branko

Anin prijatelj



# HTML methods getElementById()

- metod da se "pronađe" HTML element (korišćenjem id="demo") i promeni sadržaj elementa (innerHTML)

```
primer09g.html x
1 <!DOCTYPE html>
2 <html>
3 <head> <title>Primer 09g JavaScript</title> </head>
4 <body>
5 <h2>Neko zvoni na vratima</h2>
6 <p id="demo">Kliknite na dugme</p>
7 <button type="button"
8 onclick='document.getElementById("demo").innerHTML =
9 "Dobrodošli!"'>Klikni ovo dugme</button>
10 </body>
11 </html>
```



## Neko zvoni na vratima

Kliknite na dugme

Klikni ovo dugme

## Neko zvoni na vratima

Dobrodošli!

Klikni ovo dugme

primer09g.html

```
1 <!DOCTYPE html>
2 <html>
3 <head> <title>Primer 09g JavaScript</title> </head>
4 <body>
5 <h2>Neko zvoni na vratima</h2>
6 <p id="demo">Kliknite na dugme</p>
7 <button type="button"
8 onclick='document.getElementById("demo").innerHTML =
9 "Dobrodošli!"'>Klikni ovo dugme</button>
10 </body>
11 </html>
```



' i " mogu da zamene mesta

**Neko zvoni na vratima**

Kliknite na dugme ispod

Klikni ovde

**Neko zvoni na vratima**

Dobrodošli!

Klikni ovde

primer09g2.html x

```
1 <!DOCTYPE html>
2 <html>
3 <head> <title>Primer 09g2 JavaScript</title> </head>
4 <body>
5 <h2>Neko zvoni na vratima</h2>
6 <p id="demo">Kliknite na dugme ispod</p>
7 <button type="button"
8 onclick="document.getElementById('demo').innerHTML =
9 'Dobrodošli!'">Klikni ovde</button>
10 </body>
11 </html>
```

## Kako da menjate atribute slike?

promena src (source) atributa slike.



Upali sijalicu

Ugasi sijalicu

## Kako da menjate atribute slike?

promena src (source) atributa slike.

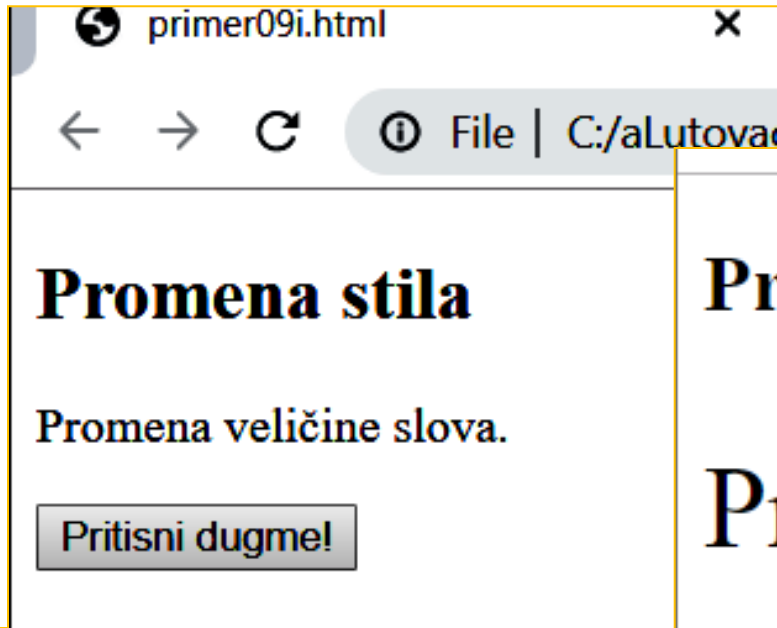


Upali sijalicu

Ugasi sijalicu

primer09h.html

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <h2>Kako da menjate atribute slike?</h2>
5 <p>promena src (source) atributa slike.</p>
6 <button onclick="document.getElementById('myImage').src='pic_bulbon.gif'"
7 ">Upali sijalicu</button>
8
9 <button onclick="document.getElementById('myImage').src='pic_bulboff.gif'"
10 ">Ugasi sijalicu</button>
11 </body>
12 </html>
```

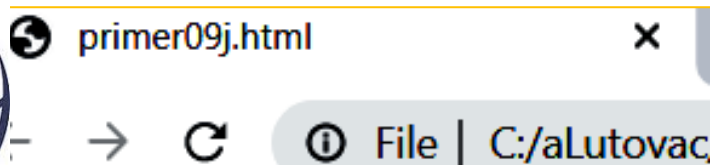


Promena stila

Promena veličine slova.

Pritisni dugme!

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <h2>Promena stila</h2>
5 <p id="demo">Promena veličine slova.</p>
6 <button type="button" onclick=
7 "document.getElementById('demo').style.fontSize='35px'
8 ">Pritisni dugme!</button>
9 </body>
10 </html>
```



## Modifikovanje prikaza

Kako da sakrijete deo teksta?

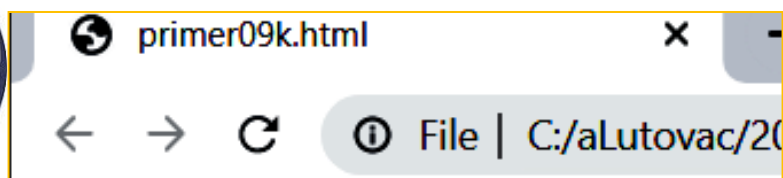
Click Me!

## Modifikovanje prikaza

Click Me!

primer09j.html

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <h2>Modifikovanje prikaza</h2>
5 <p id="demo">Kako da sakrijete deo teksta?</p>
6 <button type="button" onclick=
7 "document.getElementById('demo').style.display='none'
8 ">Click Me!</button>
9 </body>
10 </html>
```



## Modifikovanje prikaza

Kako da prikazete sakrijete deo teksta?



## Modifikovanje prikaza

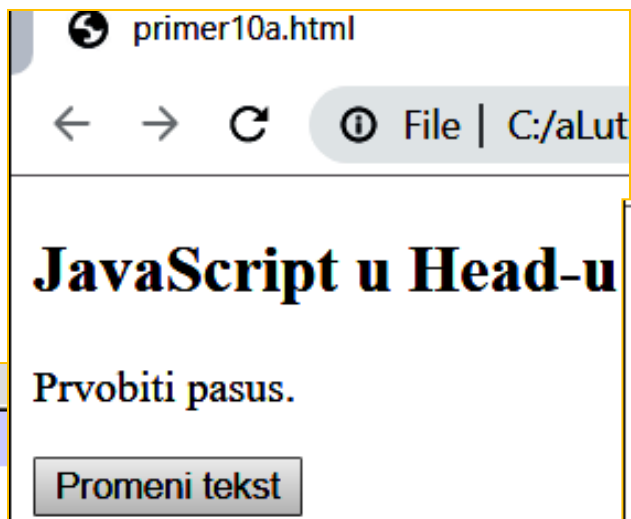
Kako da prikazete sakrijete deo teksta?

Evo me!



primer09k.html

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <h2>Modifikovanje prikaza</h2>
5 <p>Kako da prikazete sakrijete deo teksta?</p>
6 <p id="demo" style="display:none">Evo me!</p>
7 <button type="button" onclick=
8 "document.getElementById('demo').style.display='block'
9 "Pritisni dugme</button>
10 </body>
11 </html>
```



```
primer10a.html x
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <script>
5 function myHeadFunction() {
6 document.getElementById("demo").innerHTML =
7 "Novi pasus.";
8 }
9 </script>
10 </head>
11
12 <body>
13 <h2>JavaScript u Head-u</h2>
14 <p id="demo">Prvobiti pasus.</p>
15 <button type="button" onclick=
16 "myHeadFunction()">Promeni tekst</button>
17 </body>
18 </html>
```

Scrip može da bude u body ili head



```
primer10b.html x
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <h1>JS u body</h1>
5 <p id="demo">Neki pasus</p>
6 <button type="button" onclick=
7 "myBodyFunction()">Promeni tekst</button>
8 <script>
9 function myBodyFunction() {
10 document.getElementById("demo").innerHTML =
11 "Novi pasus";
12 }
13 </script>
14 </body>
15 </html>
```

## JS u body

Neki pasus

Promeni tekst

## JS u body

Novi pasus

Promeni tekst

Scrip može da bude u **body** ili **head**



# Eksterni .js

- External JS kada se isti kod koristi više puta na više web strana
- JavaScript fajlovi imaju ekstenziju .js
- U HTML se eksterni JS poziva sa src (source) atributom `<script>` taga
- Efekat je isti kao da je na mestu gde je `<script>`
- Eksterni .js **ne sme da sadrži `<script>` tag**



# Prednosti eksternih .js

- Odvojeni su HTML i .js kod
- HTML i JS su čitljiviji i jednostavniji za održavanje
- Keširani JS može da ubrza učitavanje strane
- Može da ima više JS fajlova – nekoliko script tagova

```
<script src="myScript1.js"></script>
```

```
<script src="myScript2.js"></script>
```



# External JS može da se referencira

1. sa punom URL adresom
2. sa putanjom relativnom u odnosu na trenutnu web stranu
3. samo imenom fajla ako je tamo gde je HTML
  1. `<script src="https://www.w3schools.com/js/myJS1.js"></script>`
  2. `<script src="/primeriJS/myJS1.js"></script>`
  3. `<script src="myJS1.js"></script>`



# Mogućnosti prikaza u JS

- Korišćenjem innerHTML
- Korišćenjem document.write()
- Korišćenjem window.alert()
- Korišćenjem console.log()



## Korišćenjem innerHTML

primer10c.html

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <h1>Naslov strane</h1>
5 <p>Tekst na strani, sa rezultatom JS</p>
6 <p id="demo"></p>
7 <script>
8 document.getElementById("demo").innerHTML =
9 "koliko je 5 + " + 6;
10 </script>
11 <p>Rezultat je</p>
12 <p id="demo2"></p>
13 <script>
14 document.getElementById("demo2").innerHTML = 5 + 6;
15 </script>
16 </body>
17 </html>
```

primer10c.html

← → ↻ ⓘ File | C:/aLuto

## Naslov strane

Tekst na strani, sa rezultatom JS

**koliko je** 5 + 6

Rezultat je

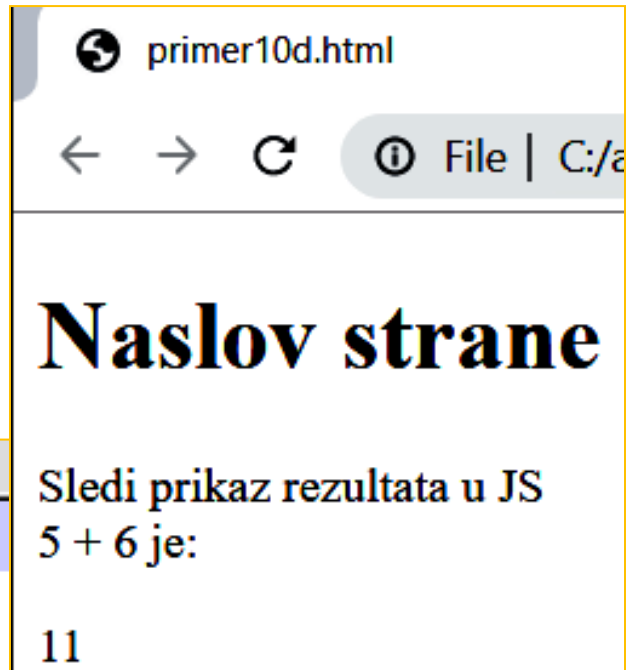
11

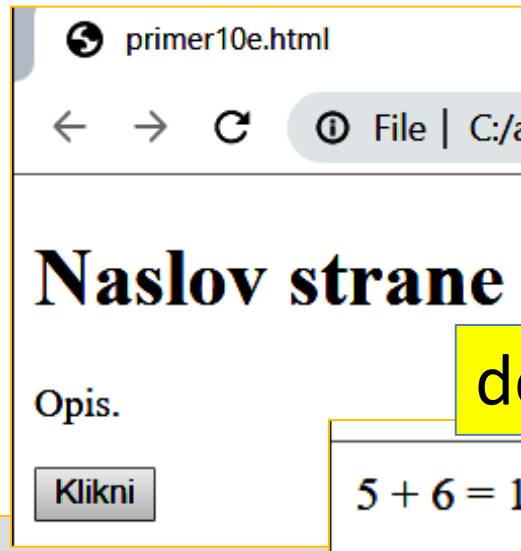


document.write()

```
primer10d.html x
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <h1>Naslov strane</h1>
5 <p>Sledi prikaz rezultata u JS

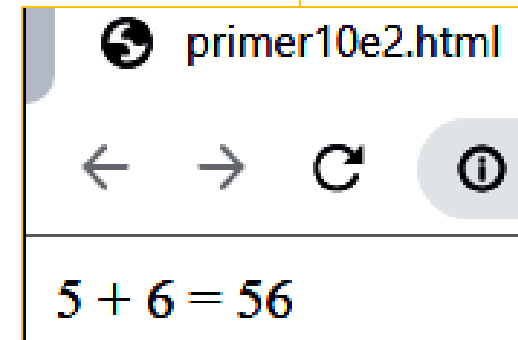
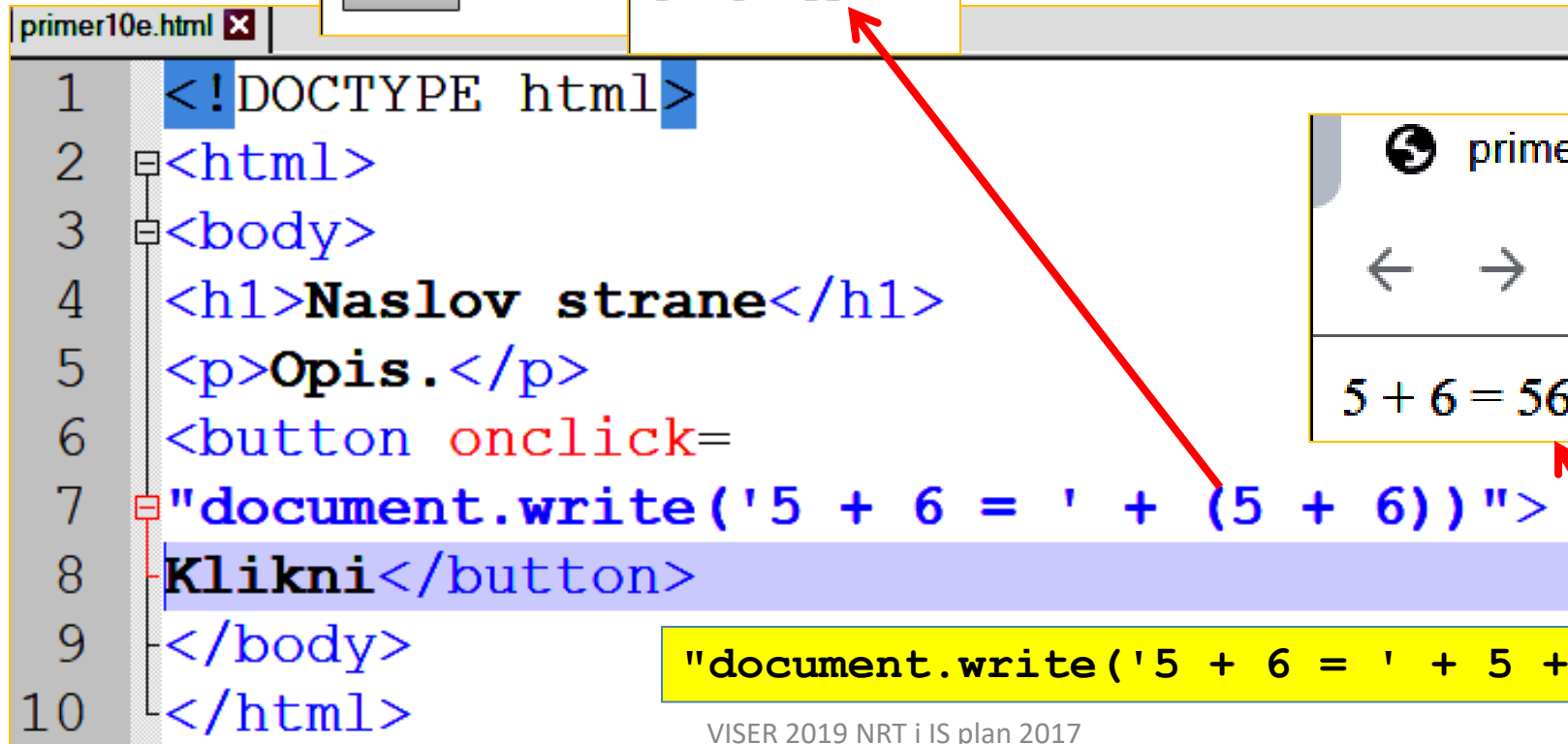
6 5 + 6 je:</p>
7 <script>
8 document.write(5 + 6);
9 </script>
10 </body>
11 </html>
```



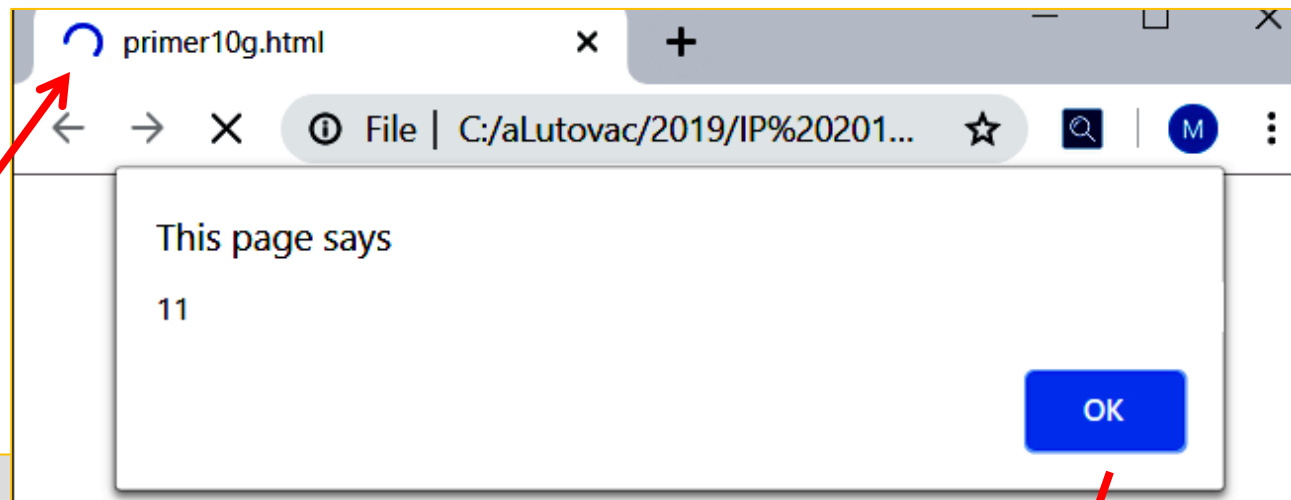


document.write()  
treba koristiti za testiranje,  
jer briše prethodni sadržaj

document.write() i klik na dugme



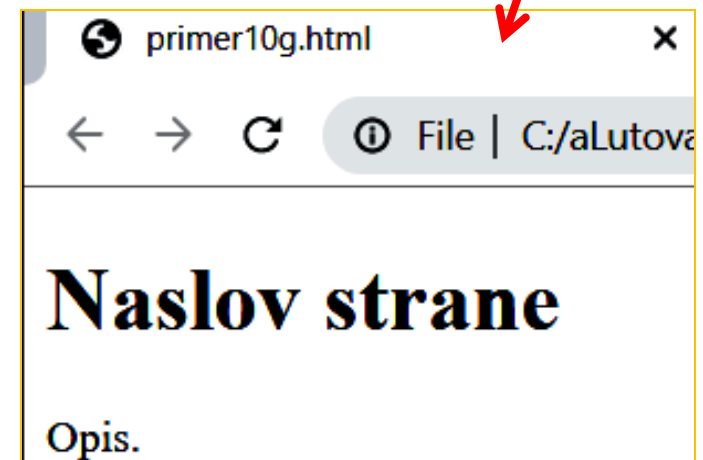
"document.write('5 + 6 = ' + 5 + 6)"

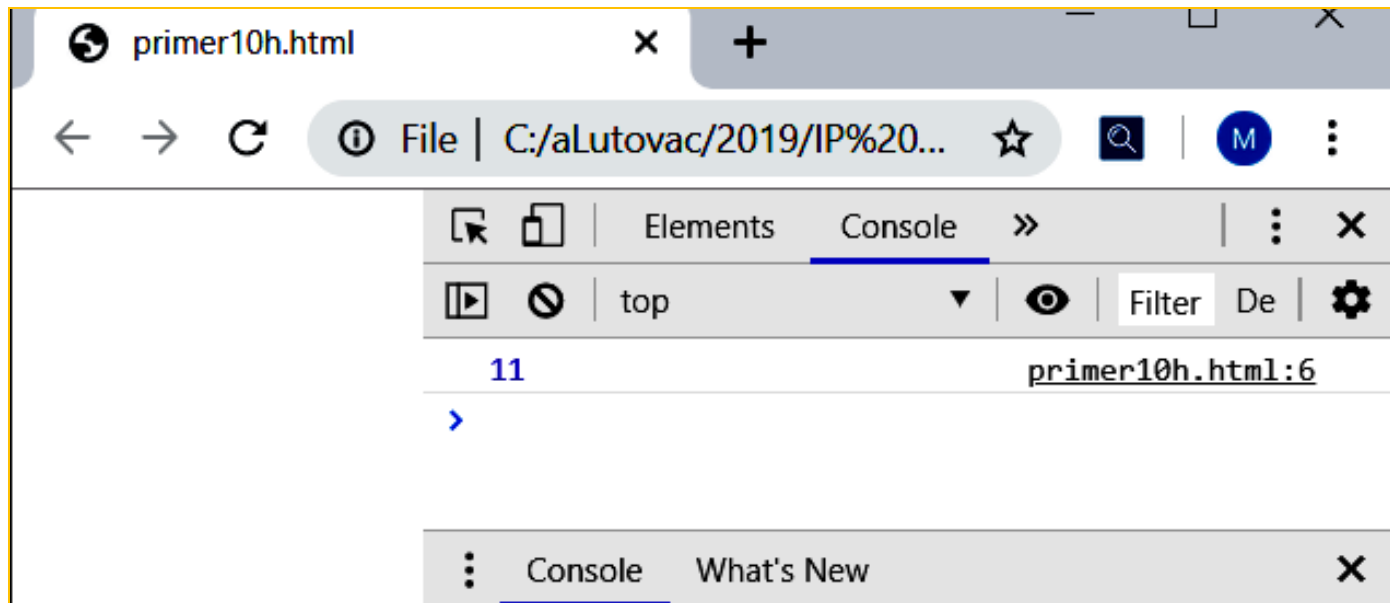


primer10g.html

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4
5 <h1>Naslov strane</h1>
6 <p>Opis.</p>
7
8 <script>
9 window.alert (5 + 6) ;
10 </script>
11
12 </body>
13 </html>
```

Korišćenjem window.alert()





```
primer10h.html x
1 <!DOCTYPE html>
2 <html>
3 <body>
4
5 <script>
6 console.log(5 + 6);
7 </script>
8
9 </body>
10 </html>
```

console.log(),  
F12, izaberi "Console" u debageru.  
click Run



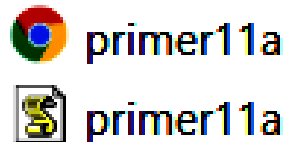
# JS sintaksa

- JS sintaksa je skup pravila po kojima je napravljena
- Računarski program je lista "instrukcija" koje izvršava računar
- U računarskoj terminologiji ove instrukcije se nazivaju **programski kod**, kod ili na eng. **statements**
- JavaScript je programski jezik
- JS statements se razdvajaju sa tačka-zarez oznakom ;



```
primer11a.js x
1 var x, y, z;
2 x = 5;
3 y = 6;
4 z = x + y;
5 document.write("x = ",x);
6 document.write("
y = ",y);
7 document.write("
z = ",z);
```

Name



Primer 11a JavaScript

← → ↻ ⓘ File

```
x = 5
y = 6
z = 11
```

Definisanje promenljivih,  
dodela vrednosti i  
prikaz u pregledaču

```
1 <html>
2 <head>
3 <title>Primer 11a JavaScript</title>
4 </head>
5 <body>
6 <script language="JavaScript" src="primer11a.js">
7 </script>
8 </body>
9 </html>
```



# JS sintaksa

- Programski kod JS se sastoji od
  - ✓ Vrednosti
  - ✓ Operatora
  - ✓ Izraza
  - ✓ Ključnih reči  
(da se identifikuje akcija koja treba da se izvrši)
  - ✓ Komentara  
(sve u vrsti posle `//` ili između `/* ... */` se ne izvršava)



# JS vrednosti

- JS sintaksa definiše dva tipa vrednosti
- ✓ Konstantne vrednosti (eng. **literals**)
  - ☐ Brojevi se pišu sa ili bez decimalne tačke
  - ☐ Stringovi su tekst između dvostrukih ili jednostrukih navodnika
- ✓ Promenljive vrednosti (eng. **variables**)



# Promenljive vrednosti (variables)

- U programskim jezicima se promenljive koriste da se zauzme prostor u memoriji u kome će se smestiti neka vrednost
- Za zauzimanje memorijskog prostora se koristi ključna reč **var**, a onda se deklariše promenljiva
- Znak jednakosti se koristi za dodelu vrednosti promenljivoj

```
var x;
```

```
x = 6;
```

```
var podaciGosti= "Marko",
 auto = "Volvo",
 cena = 200k;
```

```
var auto = "Volvo";
var auto;
```

Ostaje da je auto "Volvo"

```
var podaciGosti= "Marko", auto = "Volvo", cena = 200k;
```



# Operatori

- Koristi se operator dodele (znak jednakosti, =)
- Operatori izračunavanja aritmetičkih operacija  
( + - \* / )
- Izrazi se koriste da se, korišćenjem vrednosti, promenljivih i operatora, izračuna nova vrednost

5 \* 10

x \* 6

"Jovan," + " " + "Dule"

typeof tip promenljive

Instanceof je true ako objekat  
instanca tipa object



# Imena

- Identifikatori se koriste kao imena promenljivih
- Za prvi karakter imena može da bude slovo, znak podvučeno (\_) ili dolar znak (\$)
- Ostali karakteri imena mogu da budu slova, brojevi, znak podvučeno ili dolar znak
- JavaScript identifikatori su case sensitive (VAR, Var i var program različito koristi)
- Znak minus (-) nije odzvoljen za imena (oduzimanje)
- Programeri imaju praksu da koriste camel-case firstName, lastName, masterCard, interCity



# Imena

| Character codes                  | Decimal    | Hexadecimal |
|----------------------------------|------------|-------------|
| C0 Controls & Basic Latin        | 0-127      | 0000-007F   |
| C1 Controls & Latin-1 Supplement | 128-255    | 0080-00FF   |
| Latin Extended-A                 | 256-383    | 0100-017F   |
| Latin Extended-B                 | 384-591    | 0180-024F   |
| Spacing Modifiers                | 688-767    | 02B0-02FF   |
| Diacritical Marks                | 768-879    | 0300-036F   |
| Greek and Coptic                 | 880-1023   | 0370-03FF   |
| Cyrillic Basic                   | 1024-1279  | 0400-04FF   |
| Cyrillic Supplement              | 1280-1327  | 0500-052F   |
| General Punctuation              | 8192-8303  | 2000-206F   |
| Currency Symbols                 | 8352-8399  | 20A0-20CF   |
| Letterlike Symbols               | 8448-8527  | 2100-214F   |
| Arrows                           | 8592-8703  | 2190-21FF   |
| Mathematical Operators           | 8704-8959  | 2200-22FF   |
| Box Drawings                     | 9472-9599  | 2500-257F   |
| Block Elements                   | 9600-9631  | 2580-259F   |
| Geometric Shapes                 | 9632-9727  | 25A0-25FF   |
| Miscellaneous Symbols            | 9728-9983  | 2600-26FF   |
| Dingbats                         | 9984-10175 | 2700-27BF   |



# JS Statements i **id**

- U HTML, JS statements (programski kodovi) su programske instrukcije koje treba da budu izvršene u pregledaču
- **id** je jedinstveni identifikator za neki element  
`document.getElementById("demo").innerHTML = "Zdravo!";`
- Ovaj kod daje instrukcije pregledaču da napiše "Zdravo!" u okviru HTML elementa sa identifikatorom "demo"



Definisanje promenljivih,  
dodela vrednosti i  
prikaz u pregledaču

primer11b.js

```
1 var myName = "Marko Markov";
2 document.getElementById("demo").innerHTML = myName;
```

primer11b.html

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <h2>Naslov H2</h2>
5 <p>Kreiraj promenljivu, dodeli joj
6 vrednost, prikaži je:</p>
7 <p id="demo"></p>
8 <script language="JavaScript" src="primer11b.js">
9 </script>
10 </body>
11 </html>
```

## Naslov H2

Kreiraj promenljivu, dodeli joj vrednost, prikaži je:

Marko Markov



# JS tipovi podataka

- `var x = 16;` // broj
- `var y = „Marko“;` // string
- `var z = {ime1:"Jova", ime2:„Iva“};` // objekat



# JS funkcije

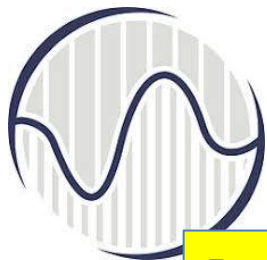
- JS function je blok komandi napravljen da izvrši specifičan zadatak
- JS function se izvršava kada je nešto prozove
- JS function koristi ključnu reč function, iza koje ide proizvoljno ime funkcije, a zatim je par zagrada (...)
- Za ime JS function su ista pravila kao i za promenljive
- Argumenti su u (...) razdvojeni zarezima (arg1, arg2, ...)
- Blok koda koji se izvršava kada je funkcija prozvana, nalazi se u vitičastim zagradama {...}
- Argumenti funkcije su realne vrednosti koje se prenose kada je funkcija prozvana
- Unutar funkcije argumenti su lokalne promenljive



## Definisanje funkcije u **body**



```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <h2>JS funkcije</h2>
5 <p>Funkcija sa dva argumenta:</p>
6 <p id="demo"></p>
7 <script>
8 function myFunction(a, b) {
9 return a * b;
10 }
11 document.getElementById("demo").innerHTML = myFunction(4, 3);
12 </script>
13 </body>
14 </html>
```



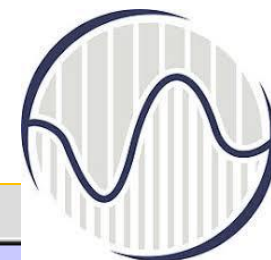
## Definisanje funkcije u head

## JS funkcije

Funkcija sa dva argumenta:

12

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <script>
5 function myFunction(a, b) {
6 return a * b;
7 }
8 </script>
9 </head>
10 <body>
11 <h2>JS funkcije</h2>
12 <p>Funkcija sa dva argumenta:</p>
13 <p id="demo"></p>
14 <script>
15 document.getElementById("demo").innerHTML = myFunction(4, 3);
16 </script>
17 </body>
18 </html>
```



```
primer11e.html x
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <h2>JS funkcije</h2>
5 <p>Funkcija sa dva argumenta:</p>
6 <p id="demo"></p>
7 <script language="JavaScript" src="primer11e.js">
8 </script>
9 </body>
10 </html>
```

Definisanje u .js fajlu

primer11e.html x

← → ↻ ⓘ File | C:/aLutova

## JS funkcije

Funkcija sa dva argumenta:

12

```
primer11e.js x
1 function myFunction(a, b) {
2 return a * b;
3 }
4 document.getElementById("demo").innerHTML = myFunction(4, 3);
```



# JS funkcije pozivanje i return

- Kod unutar JS function se izvršava kada se funkcija prozove
  - Kada se desi neki događaj (korisnik klikne dugme)
  - Kada se pozove iz JS koda
  - Automatski (samopozivanje)
- Kada JS dođe do linije koda **return**, funkcija prekida sa izvršavanjem
- Ako se funkcija pozove iz neke linije koda, JS će „vratiti” vrednost koja se koristi za izvršavanje nakon što je funkcija prozvana
- Najčešće, funkcija izračunava vrednost koja se vraća onome ko je pozvao funkciju



# Primer korišćenja funkcije

```
var x = myFunction(4, 3);
 /* funkcija je pozvana, vrednost koja se izračuna vraća se
 u x */
function myFunction(a, b) {
 return a * b;
 // funkcija vraća proizvod a * b
}
```

- Kod može da se koristi više puta
- Isti kod se koristi više puta sa različitim argumentima, što daje različite rezultate



# JavaScript Objects

- U realnom životu, automobil je object
- Automobil ima properties (težina, boja), i metode (kreni i stani)
- Svi automobili imaju iste properties, ali se osobine razlikuju od auta do auta
- Svi automobili imaju iste metode, ali se metode izvršavaju u različito vreme



**Object**  
auto

**Properties**  
ime, model, boja

**Methods**  
kreni, vozi



auto.ime = Fiat  
auto.model = 500L  
auto.tezina = 850kg  
auto.boja = crvena

auto.kreni()  
auto.vozi()  
auto.koci()  
auto.stani()



# JavaScript Objects

- Primer promenljive (**variable**) imenovana kao auto

```
var auto = "Fiat"; //dodela jedne vrednosti
```

- Objects su takođe variables
- Objects mogu da sadrže brojne vrednosti
- JavaScript može da dodeljuje veći broj vrednosti (Fiat, 500L, bela) promenljivoj (**variable**) koja je imenovana kao auto:

```
var auto = {tip: "Fiat", model: "500L", boja: "bela"};
```



## JavaScript objekti su kontejneri za **imenovane vrednosti**

Vrednosti su napisane kao parovi **ime:vrednost**  
**ime** i **vrednosti** su razdvojeni dvotačkom :

primer12a.html

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <p>Hocu da kupim:</p>
5 <p id="demo"></p>
6 <script>
7 var auto = {tip:"Fiat", model:"500L", boja:"bela"};
8 document.getElementById("demo").innerHTML =
9 ("auto marke " + auto.tip +
10 ", model " + auto.model +
11 ", boja " + auto.boja + "");
12 </script>
13
14 </body>
15 </html>
```

Hocu da kupim:

auto marke **Fiat**, model **500L**, boja **bela**



Hocu da kupim auto marke **Fiat**, model **500L**, boja **bela**

primer12b.html x

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <p id="demo"></p>
5 <script>
6 var auto = {
7 tip:"Fiat",
8 model:"500L",
9 boja:"bela"
10 };
11 document.getElementById("demo").innerHTML =
12 ("Hocu da kupim auto marke " + auto.tip +
13 ", model " + auto.model +
14 ", boja " + auto.boja + "");
15 </script>
16 </body>
17 </html>
```

Definicija objekta može da bude u više linija koda

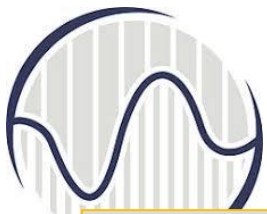


# Accessing Object Properties

- Može se pristupiti properties na 2 načina:

`imeObjekta.osobinaObjekta`

`imeObjekta["osobinaObjekta"]`



primer12c.html

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <p id="demo"></p>
5 <script>
6 var auto = {
7 tip:"Fiat",
8 model:"500L",
9 boja:"bela"
10 };
11 document.getElementById("demo").innerHTML =
12 ("Pristup objektu imeObjekta.osobinaObjekta" +
13 "
imeObjekta je auto," +
14 "
osobinaObjekta je " + auto.tip + "");
15 </script>
16 </body>
17 </html>
```

Pristup objektu imeObjekta.osobinaObjekta  
imeObjekta je **auto**,  
osobinaObjekta je **Fiat**

pristup JavaScript objektima  
**imeObjekta.osobinaObjekta**



primer12d.html

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <p id="demo"></p>
5 <script>
6 var auto = {
7 tip:"Fiat",
8 model:"500L",
9 boja:"bela"
10 };
11 document.getElementById("demo").innerHTML =
12 ('Pristup objektu imeObjekta["osobinaObjekta"] ' +
13 "
imeObjekta je auto," +
14 "
osobinaObjekta je " + auto["tip"] + "");
15 </script>
16 </body>
17 </html>
```

Pristup objektu imeObjekta["osobinaObjekta"]  
imeObjekta je **auto**,  
osobinaObjekta je **Fiat**

pristup JavaScript objektima  
imeObjekta["osobinaObjekta"]



# JavaScript Object Methods

- Metode su aktivnosti koje bi mogle da izvrše objekti
- JavaScript objekti su kontejneri za imenovane vrednosti koje nazivamo properties ili **methods**

```
function() {return auto.kreni + ... + auto.stani;}
```



# Pristup Metodi objekta

- Pristupa se metodi objekta:

`imeObjekta.imeMetode ()`

`Ime = person.fullName()`

- Ako se pristupa metodi bez zagrada (), vraća se definicija metode:

`ime = person.fullName`



```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <p id="demo"></p>
5 <script>
6 var auto = {
7 tip: "Fiat",
8 model: "500L",
9 oprema1: "klima",
10 oprema2: "alat",
11 boja : "bela ",
12 id : 5678,
13 tipModelBoja : function() {
14 return "Kupujem "+this.tip+", model "+
15 this.model+", boja "+this.boja;
16 }
17 };
18 document.getElementById("demo").innerHTML =
19 auto.tipModelBoja();
20 </script>
21 </body>
22 </html>
```

Kupujem Fiat, model 500L, boja bela

this  
pristup JavaScript objektima  
**auto.tipModelBoja();**

```

1 <!DOCTYPE html>
2 <html>
3 <body>
4 <p id="demo"></p>
5 <script>
6 var auto = {
7 tip: "Fiat",
8 model: "500L",
9 oprema1: "klima",
10 oprema2: "alat",
11 boja : "bela ",
12 id : 5678,
13 tipModelBoja : function() {
14 return "Kupujem "+this.tip+", model "+
15 this.model+", boja "+this.boja;
16 }
17 };
18 document.getElementById("demo").innerHTML =
19 auto.tipModelBoja;
20 </script>
21 </body>
22 </html>

```

pristup JavaScript objektima  
bez zagrada **auto.tipModelBoja;**

metod je u stvari definicija funkcije  
zapamćena kao vrednost property



function() { return "Kupujem "+this.tip+", model "+ this.model+", boja "+this.boja; }



# JavaScript Scope

- Scope određuje pristupačnost (vidljivost) promenljive, objekta i funkcije iz različitih delova koda
- U JavaScript-u postoje dve vrste:
  - Local scope
  - Global scope
- JavaScript ima function scope:  
Svaka funkcija stvara novi scope
- Promenljive definisane unutar funkcije nisu vidljive izvan funkcije



# Lokalne JavaScript promenljive

- Promenljive deklarisanе unutar JavaScript funkcije, postaju lokalne za funkciju
- Lokalne promenljive imaju lokalni scope: može im se pristupiti samo unutar funkcije
- Pošto se lokalne promenljive mogu prepoznati samo unutar funkcije, promenljive sa istim imenom mogu da se koriste u različitim funkcijama
- Lokalne promenljive nastaju kada se počinje neka funkcija, i nestaju kada se završi ta funkcija



Lokalna promenljiva `autoIme` **nije vidljiva** izvan funkcije:

Tip promenljive izvan funkcije je undefined

Lokalna promenljiva `autoIme` **vidljiva je** unutar funkcije:

Tip promenljive unutar funkcije je +string

primer12g.html

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <p>Lokalna promenljiva autoIme nije vidljiva izvan funkcije:</p>
5 <p id="demo"></p>
6 <p>Lokalna promenljiva autoIme vidljiva je unutar funkcije:</p>
7 <p id="demo2"></p>
8 <script>
9 myFunction();
10 document.getElementById("demo").innerHTML =
11 "Tip promenljive izvan funkcije je <u>"+typeof autoIme+"</u>";
12 function myFunction() {
13 var autoIme = "Fiat";
14 document.getElementById("demo2").innerHTML =
15 ("Tip promenljive unutar funkcije je +<u>"+typeof autoIme+"</u>");
16 }
17 </script>
18 </body>
19 </html>
```

Lokalna promenljiva



# Lokalne JavaScript promenljive

- Promenljive deklarisanе izvan funkcija postaju globalne
- Globalna promenljiva ima globalnu vidljivost
- Svi skriptovi i funkcije na veb strani mogu da pristupe promenljivoj
- U JavaScript-u, objekti i funkcije su takođe promenljive



Globalna promenljiva `autoIme` **je vidljiva** izvan funkcije:

Tip promenljive izvan funkcije je string

Globalna promenljiva `autoIme` **vidljiva je** i unutar funkcije:

Tip promenljive unutar funkcije je string

primer12h.html

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <p>Globalna promenljiva autoIme je vidljiva izvan funkcije:</p>
5 <p id="demo"></p>
6 <p>Globalna promenljiva autoIme vidljiva je i unutar funkcije:</p>
7 <p id="demo2"></p>
8 <script>
9 var autoIme = "Fiat";
10 myFunction();
11 document.getElementById("demo").innerHTML =
12 "Tip promenljive izvan funkcije je <u>" + typeof autoIme + "</u>";
13 function myFunction() {
14 document.getElementById("demo2").innerHTML =
15 ("Tip promenljive unutar funkcije je <u>" + typeof autoIme + "</u>");
16 }
17 </script>
18 </body>
19 </html>
```

Globalna promenljiva



# Automatski globalno promenljiva

- Ako se dodeljuje vrednost promenljivoj koja nije deklarirana, promenljiva će automatski postati globalna promenljiva
- Sledeći kod deklarira promenljivu kao globalnu iako je vrednost dodeljene unutar funkcije



Lokalna promenljiva **autoIme** je **vidljiva** izvan funkcije:

Tip promenljive izvan funkcije je string

Lokalna promenljiva **autoIme** **vidljiva** je i unutar funkcije:

Tip promenljive unutar funkcije je +string

```
primer12i.html x
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <p>Lokalna promenljiva autoIme je vidljiva izvan funkcije:</p>
5 <p id="demo"></p>
6 <p>Lokalna promenljiva autoIme vidljiva je i unutar funkcije:</p>
7 <p id="demo2"></p>
8 <script>
9 myFunction();
10 document.getElementById("demo").innerHTML =
11 "Tip promenljive izvan funkcije je <u>" + typeof autoIme + "</u>";
12 function myFunction() {
13 // var autoIme = "Fiat";
14 autoIme = "Fiat";
15 document.getElementById("demo2").innerHTML =
16 ("Tip promenljive unutar funkcije je +<u>" + typeof autoIme + "</u>");
17 }
18 </script>
19 </body>
20 </html>
```



Lokalna promenljiva **autoIme** je **vidljiva** izvan funkcije:

Tip promenljive izvan funkcije je string

Lokalna promenljiva **autoIme** **vidljiva** je i unutar funkcije:

Tip promenljive unutar funkcije je +string

Lokalna promenljiva **autoIme** je **vidljiva** izvan funkcije:

Tip promenljive izvan funkcije je undefined

Lokalna promenljiva **autoIme** **vidljiva** je i unutar funkcije:

Tip promenljive unutar funkcije je +string

primer12i.html

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <p>Lokalna promenljiva autoIme je vidljiva izvan funkcije:
5 <p id="demo"></p>
6 <p>Lokalna promenljiva autoIme je vidljiva i unutar funkcije:
7 <p id="demo2"></p>
8 <script>
9 myFunction();
10 document.getElementById("demo2").innerHTML =
11 "Tip promenljive izvan funkcije je <u>"+typeof autoIme+"</u>";
12 function myFunction() {
13 // var autoIme = "Fiat";
14 autoIme = "Fiat";
15 document.getElementById("demo2").innerHTML =
16 ("Tip promenljive unutar funkcije je <u>"+typeof autoIme+"</u>");
17 }
18 </script>
19 </body>
20 </html>
```



# Globalne promenljive u HTML

- Sa JavaScript-om, globalna vidljivost je celokupno JavaScript okruženje
- U HTML-u, globalna vidljivost je window objekat
- Sve globalne promenljive pripadaju window objektu
- **NEMOJTE da kreirate globalne promenljive sve dok to nije neophodno**



U HTML-u, sve globalne promenljive postaju window promenljive  
window.autoIme koristimo za prikaz Fiat

primer12j.html x

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4 <p>U HTML-u, sve globalne promenljive
5 postaju window promenljive</p>
6 <p id="demo"></p>
7 <script>
8 var autoIme = "Fiat";
9 document.getElementById("demo").innerHTML =
10 "window.autoIme koristimo za prikaz " +
11 window.autoIme;
12 </script>
13 </body>
14 </html>
```

Globalna promenljiva u HTML-u



# Upozorenja

- **NEMOJTE** da kreirate globalne promenljive sve dok to nije neophodno
- Globalne promenljive (ili funkcije) mogu da prebrišu window promenljive (ili funkcije)
- Bilo koja funkcija, uključujući window objekt, može da prebriše globalne promenljive i funkcije
- Globalne promenljive se ne kreiraju automatski u "Strict Mode"



# Životni vek JavaScript promenljivih

- Životni vek JavaScript promenljive započinje njenim deklarisanjem
- Lokalne promenljive nestaju kada se završe funkcije
- U veb pregledaču, globalne promenljive se brišu (nestaju) kada se zatvori prozor pregledača, ali ostaju dostupne novim stranama u istom prozoru
- Argumenti funkcije (parametri) funkcionišu kao lokalne promenljive unutar funkcije