

JavaScript

Skript jezici

- Izvršavaju se u čitaču
- Ugrađuju se u HTML stranice
- Omogućavaju interaktivnost
- Interpretirani jezik
 - Ne kompajliraju se
 - Izvršavaju se momentalno

<script>

- Sve između tagova **<script>** i **</script>** browser smatra elementima programskog koda
- **<script>** tag se može javiti bilo gde u HTML dokumentu
- Programski kod se ne mora nalaziti u HTML datoteci i može se nalaziti u drugoj datoteci
- Ukoliko se programski kod nalazi u drugoj datoteci mora se pozvati iz odgovarajuće HTML datoteke
- Ukoliko atribut **type** u **<script>** tagu ima vrednost „**text/javascript**“ tada se radi o JavaScript programskom jeziku

Primer JavaScript

```
<html>
  <head>
    <script type="text/javascript">
      //...
    </script>
  </head>
  <body>
    <script type="text/javascript">
      //...
    </script>
  </body>
</html>
```

Primer JavaScript u drugoj datoteci

```
<html>  
  <head>  
    <script src="skript.js"></script>  
  </head>  
  <body>  
  </body>  
</html>
```

JavaScript

- Sintaksa slična programskom jeziku Java
 - nije programski jezik Java
- Nema tipove podataka
 - kod deklaracije promenljivih se ne stavlja tip (interpreter).
- Nema kreiranja novih klasa
 - klase su zapravo „specijalne“ funkcije
 - ugrađene funkcije,
 - ugrađeni objekti
- Sistem događaja

Promenljive

- Promenljive sadrže informacije
- Deklaracija promenljivih upotrebom ključne reči **var**
- Primer:

```
var a;  
var b = 5;  
var c = "Pera";
```
- Ako se izostavi ključna reč **var** kod deklaracije promenljive unutar funkcije, ona postaje globalna
- Pored **var** može se koristiti i ključna reč **let**
 - var – function scoped
 - let – block scoped

Tipovi podataka

- Osnovni tipovi podataka:
 - Broj
 - Boolean
 - String (između navodnika: "tekst" ili 'tekst')
 - Objekat (kolekcija svojstava i funkcija)
 - Ugrađeni objekti:
 - String
 - Object
 - Array
 - Math
 - function – svaka funkcija je objekat
 - ostali
 - Korisnički definisani

Vrednosti promenljivih

- Primitivni tipovi
 - broj koji uvek se čuva kao double floating point
 - `var x = 5;`
 - true ili false za boolean tip
 - `var x = true;`
 - "tekst" ili 'tekst' za tip String
 - `var x = "pera";`
 - null – kao i u ostalim programskim jezicima
 - `var x = null;`
 - undefined – neinicijalizovana promenljiva
 - `var x;`
- Objekti
 - kolekcija svojstava i funkcija, kojoj se pristupa preko tačke ili preko uglastih zagrada
 - null
 - undefined – neinicijalizovana promenljiva

Objekti

- Objekti
 - ugrađeni
 - korisnički definisani
- Korisnički definisani objekti se kreiraju:
 - Object objektom
 - JSON sintaksom
 - pozivom konstruktora

Aritmetički i operatori dodele

- Aritmetički: + - * / % ++ --

```
x = 5;
```

```
y = x * 4;
```

```
z = y % 5;
```

- Dodele: = += -= *= /= %=

```
y += 5;      y=y+5;
```

- Operator + ima posebno značenje kada su operandi stringovi:

```
a = "Pera";
```

```
b = "Car";
```

```
c = a + b;
```

- Kada sabiramo stringove i brojeve, rezultat je string

Aritmetički operatori

- $y = 5;$

Operator	Rezultat
$x=y+2$	$x=7$
$x=y-2$	$x=3$
$x=y\%2$	$x=1$
$x=++y$	$x=6, y=6$
$x=y++$	$x=5, y=6$
$x=--y$	$x=4$

Operatori dodele

x = 10;

y = 5;

Operator	Isto kao	Rezultat
x=y		x=5
x+=y	x=x+y	x=15
x-=y	x=x-y	x=5
x*=y	x=x*y	x=50
x/=y	x=x/y	x=2
x%=y	x=x%y	x=0

Relacioni operatori

- Relacioni: `==` `===` `!=` `<` `<=` `>` `>=`

```
x = 5;
```

```
if (x == 5)
```

```
    document.write("x je jednako 5");
```

- Operator `===` će porediti i vrednost i tip:

```
var x = 5;
```

```
if (x === "5") // == bi vratilo true
```

```
    document.write("x je string sa sadržajem 5");
```

- Rezultat relacionih operatora je logička vrednost tačno (true) ili netačno (false)

Relacioni operatori

- `x = 5;`

Operator	Rezultat
<code>==</code>	<code>x == 8</code> je netačno (false)
<code>===</code>	<code>x === 5</code> je tačno (true) <code>x === "5"</code> je netačno (false)
<code>!=</code>	<code>x != 8</code> je tačno (true)
<code>></code>	<code>x > 8</code> je netačno (false)
<code><</code>	<code>x < 8</code> je tačno (true)
<code>>=</code>	<code>x >= 8</code> je netačno (false)
<code><=</code>	<code>x <= 8</code> je tačno (true)

Logički operatori

- Logički: **&&** **||** **!**
- Rezultat logičkih operatora je tačno (true) ili netačno (false)
- Operandi logičkih operatora su logički izrazi

&&	false	true
false	false	false
true	false	true

	false	true
false	false	true
true	true	true

!	
false	true
true	false

Logički operatori

x = 6;

y = 3;

Operator	Objašnjenje	Primer
&&	konjukcija (and, i)	(x < 10 && y > 1) tačno (true)
	disjunkcija (or, ili)	(x==5 y==5) netačno (false)
!	negacija (not, ne)	!(x==y) tačno (true)

Nizovi

- Slično Javi, sadrže elemente, attribute i metode
- Atribut length daje veličinu niza
- Metoda push dodaje element na kraj niza

Nizovi

```
// Kreiranje niza "JSON" sintaksom  
var niz3 = [];  
niz3[0] = 6;  
var niz4 = [1, 2, 3, 4];  
alert(niz2.length);  
// Dodavanje elementa na kraj niza  
niz1.push(33);  
// Uklanjanje jednog elementa na poziciji index  
niz1.splice(index, 1);
```

Kontrola toka

- `if else`
- `switch`
- `for`
- `while`
- `do while`
- `break`
- `continue`

if else

- Opšta sintaksa:

```
if (uslov_1)
```

```
    telo_1
```

```
else if (uslov_2)
```

```
    telo_2
```

```
else
```

```
    telo_3
```

Primer

```
// Da li nam trebaju zagrade { } ?  
  
if (poeni > 94)  
    ocena = 10;  
else if (poeni > 84)  
    ocena = 9;  
else if (poeni > 74)  
    ocena = 8;  
else if (poeni > 64)  
    ocena = 7;  
else if (poeni > 54)  
    ocena = 6;  
else ocena = 5;
```

Primer

```
var vreme = 11;  
if (time < 10) {  
    document.write("Dobro jutro!");  
} else {  
    document.write("Dobar dan!");  
}
```

switch

- Izraz u `switch()` izrazu mora da proizvede celobrojnu vrednost.
- Ako ne proizvodi celobrojnu vrednost, ne može da se koristi `switch()`, već `if()`!
- Ako se izostavi `break`; propašće u sledeći `case`.
- Kod `default` izraza ne mora `break` - to se podrazumeva.

Primer

```
switch (a)
{
    case 1:
    case 2: i = j + 6;
           break;
    case 3: i = j + 14;
           break;
    default: i = j + 8;
}
```

while

- Za cikličnu strukturu kod koje se samo zna uslov za prekid.
- Telo ciklusa ne mora ni jednom da se izvrši
- Opšta sintaksa:

```
while (uslov)
```

```
    telo
```

- Važno: izlaz iz petlje na false!

Primer

```
<html>
  <body>
    <script type="text/javascript">
      var i = 0;
      while ( i <= 10)
      {
        document.write("Trenutno je " + i);
        document.write("<br />");
        i = i + 1;
      }
    </script>
  </body>
</html>
```

do while

- Za cikličnu strukturu kod koje se samo zna uslov za prekid
- Razlika u odnosu na while petlju je u tome što se telo ciklusa izvršava makar jednom.

- Opšta sintaksa:

`do`

`telo`

`while (uslov) ;`

- Važno: izlaz iz petlje na false!

Primer

```
<html>
  <body>
    <script type="text/javascript">
      var i = 0;
      do {
        document.write("The number is " + i);
        document.write("<br />");
        i = i + 1;
      } while (i < 10);
    </script>
  </body>
</html>
```

for

- Za organizaciju petlji kod kojih se unapred zna koliko puta će se izvršiti telo ciklusa.
- Petlja sa početnom vrednošću, uslovom za kraj i blokom za korekciju.
- Opšta sintaksa:

```
for (inicijalizacija; uslov; korekcija)  
    telo
```

Primer

```
<html>
  <body>
    <script type="text/javascript">
      var i = 0;
      for (i = 0; i <= 10; i++) {
        document.write("The number is " + i);
        document.write("<br />");
      }
    </script>
  </body>
</html>
```

for ... in petlja

- Za iteriranje kroz nizove
- Opšta sintaksa:

```
for (promenljiva in niz) {  
    ...  
}
```


Primer

```
<html>
  <body>
    <script type="text/javascript">
      var x;
      var vozila = [];
      vozila[0] = "Saab";
      vozila[1] = "Volvo";
      vozila[2] = "BMW";
      for (x in vozila) {
        document.write(vozila[x] + "<br />");
      }
    </script>
  </body>
</html>
```

break i continue

- **break** – prekida telo tekuće ciklične strukture (ili **case** dela) i izlazi iz nje.
- **continue** – prekida telo tekuće ciklične strukture i otpočinje sledeću iteraciju petlje.

break i continue

```
<html>
  <body>
    <script type="text/javascript">
      var i = 0;
      for (i = 0; i <= 10; i++)
      {
        if (i == 3) {
          break;
        }

        document.write("The number is " + i);
        document.write("<br />");
      }
    </script>
  </body>
</html>
```