JAVASCRIPT

1. DEO

UVOD

- HTML , CSS
- Problem dinamičke obrade podataka unetih od strane korisnika
- Potrebna nova tehnologija za realizaciju dinamičkih delova
- Prvi pokušaj je bio pomoću serverskih komponenti, od kojih je najpopularnija bila CGI (Common Gateway Interface)
- Problem je predstavljala česta klijent-server komunikacija

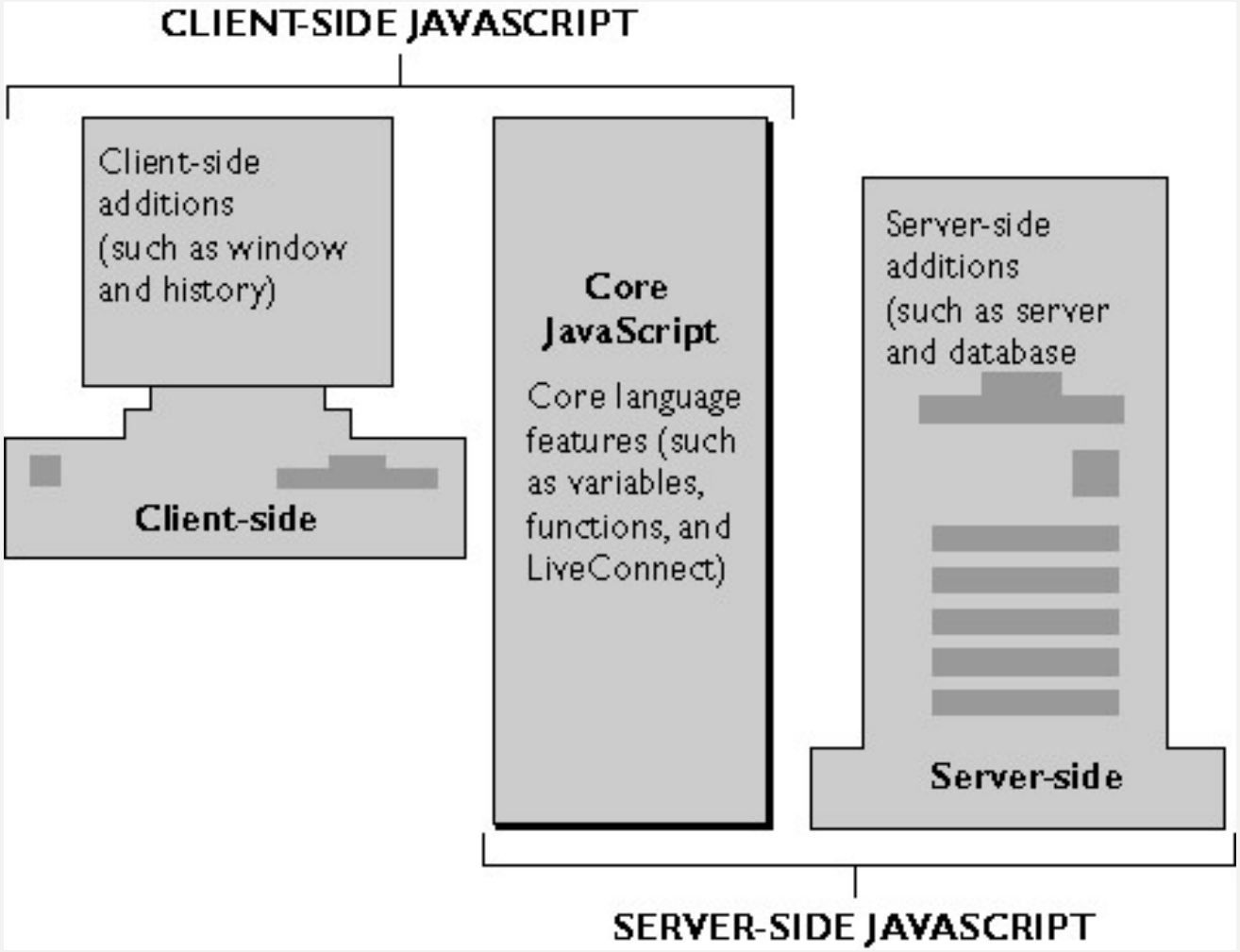
UVOD

- Decembar 1995.god, Netscape i Sun su predstavili JavaScript 1.0, originalno nazvan LiveScript
- JScript (Microsoft)
- U junu 1997. godine pojavila se ECMAScript specifikacija za skript jezike
- Danas su Netscape-ova i Microsoft-ova verzija standarda identične u preko 95% slučajeva

UVOD

- Dominantan jezik na klijentskoj strani
- Velika popularnost i na serverskoj strain (Node.js)
- JavaScript biblioteke (libraries) omogućavaju lakši razvoj JavaScript aplikacija - JQuery, Dojo Toolkit, MooTools ...
- Javascript okviri (frameworks) - AngularJS, Ember.js, Meteor.js, Vue.js

UVOD


CLIENT-SIDE JAVASCRIPT
Client-side additions (such as window and history)
Client-side
Core JavaScript
Core language features (such as variables, functions, and LiveConnect)
Server-side additions (such as server and database
Server-side
SERVER-SIDE JAVASCRIPT

OSOBINE

- Platformski neutralan
- Objektno baziran jezik
- Modularno programiranje
- Integrisanost sa HTML-om
- Nema velikih sličnosti sa Javom

UKLJUČIVANJE PROGRAMSKOG KODA

- Direktno pisanje koda u okviru stranice

```
<SCRIPT LANGUAGE="JavaScript">
...JavaScript kod…
</SCRIPT>
```

- Poziv spoljašnjeg js dokumenta

```
<SCRIPT LANGUAGE="JavaScript" SRC=" JSkod.js">
</SCRIPT>
```

- gde je JSkod.js dokument koji sadrži željene JavaScript funkcije

KOMENTARI

- Komentar jedne linije - oznaka „//” :

```
// komentar u jednoj liniji ...
```

- Komentar više redova - „/*” za početak bloka pod komentarom i „*/” za kraj bloka pod komentarom:

```
/*
komentar u više redova...
*/
```

PRIKAZ

- HTML tekst se prikazuje pomoću JavaScript koda na stranici korišćenjem metode

```
document.write("neki tekst koji seprikazuje");
```

- Argument ove metode je String koji može biti proizvoljan HTML kod

```
<script language="JavaScript">
document.write("<b>Prvi red</b><br><i>Drugi red</i>")
</script>
```

PROMENLJIVE

- Vrednosti koje program smešta u okviru memorije računara
- Da bi se koristile u okviru programa:
  - Napisati naredbu koja kreira promenljivu
  - Dodeliti joj željeno ime

IMENA PROMENLJIVIH

- Ime koje se dodeljuje promenljivoj se naziva i identifikator
- Pravila i konvencije pri imenovanju promenljivih:
  - Identifikator mora počinjati slovom, znakom dolar ($), ili donjom crtom (_)
  - U okviru imena se mogu koristiti brojevi, ali ne kao prvi karakter
  - Ne mogu se koristiti prazna mesta u okviru imena
  - Ne mogu se koristiti rezervisane reči za identifikatore

CASE SENSITIVE

- JavaScript je **case sensitive jezik**, što znači da se velika i mala slova razlikuju
- promenljiva Aaa je različita od promenljive AAA.
- Ključne reči (for, if, else, class, int,...) se ne mogu koristiti u imenu promenljivih.

DEKLARACIJA

- Viši programski jezici
  - Zahtevaju da se deklariše tip podataka promenljive
  - Statički tip podataka
    - Tip podataka se ne menja nakon same deklaracije
- Jednostavniji programski jezici - JavaScript
  - Ne zahtevaju da se deklariše tip podataka promenljive
  - Dinamički tip podataka
    - Tip podataka se može menjati nakon same deklaracije

DEFINICIJA TIPA PODATAKA

- JavaScript je jednostavniji programski jezik
  - ne zahteva da se deklariše tip podataka promenljive
- JavaScript interpreter
  - automatski prepoznaje koji tip podataka je smešten u okviru promenljive
  - dodeljuje tip podatka promenljivoj dinamički

VAR

- U okviru JavaScript jezika, var rezervisana reč se koristi za kreiranje promenljivih
- Kreiranje promenljive se naziva i deklaracija promenljive

```
var x;  //deklaracija promenljive
```

- Dodela određena vrednosti promenljivoj se naziva inicijalizacija

```
var imePromenljive = vrednost;
var ime_promenljive1 = vrednost1, ime_promenljive2 = vrednost2 ;
```

- Vrednost dodeljena promenljivoj može biti String ili numerička vrednost

```
var y = 95;  //inicijalizacija int promenljive
```

VAR

- Nije neophodno deklarisati promenljivu pre prve dodele vrednosti (automatski će se izvršiti deklarisanje)
- Predeklarisanje je dozvoljeno.

- var x; x=8;
- x=8; var x;
- var x=8;
- x=8;

TIPOVI PODATAKA

- Informacija o sadržaju promenljive
- Tipovi:
  - celobrojne vrednosti,
  - racionalni brojevi,
  - stringovi (niz karaktera),
  - logički tip (true/false).

CELOBROJNE VREDNOSTI

- Brojna osnova 10, 8 ili 16
- Najčešće se koristi osnova 10. Koriste se cifre od 0 do 9, početna cifra ne sme biti 0

- Brojevi prikazani u oktalnom brojnom sistemu, sa osnovom 8, moraju počinjati sa cifrom 0, a ostale cifre su od 0 -7.

- Brojevi prikazani u heksadecimalnom brojnom sistemu, sa osnovom 16, moraju počinjati sa 0x ili 0X, ostale cifre su od 0 - 15, s tim da se cifre 10 -15 prikazuju slovima A -F.

RACIONALNI BROJEVI

- Mogu se prikazati na dva načina:
  - pomoću decimalne tačke (3.14)
  - pomoću eksponencijalne prezentacije (314E-2 ili 314e-2)
- Brojevi su 64-bitni floating point, IEEE 754 standard
- 0-51 je broj, 52-62 eksponent, 63 je znak

STRING

- String predstavlja proizvoljan niz karaktera između navodnika

  ("neki tekst") ili između apostrofa ('neki tekst').

  x= “Maki”, y=‘Zoki’

  x = “Pera ‘Pekar’ Perić” Pera ‘Pekar’ Perić

  y = ‘Pera “Pekar” Perić’ Pera “Pekar” Perić

SPECIJALNI KARAKTERI

- U stringovima se mogu koristiti i specijalni karakteri.

- \b = jedno mesto levo (backspace)
- \f = jedan red nadole (form feed)
- \n = početak novog reda (new line character)
- \r = return (carriage return)
- \t = tabulator (tab)

KONVERZIJA U STRING

```
<script>
    x = 2+ 4;    document.write(x);  document.write("<br>");
    x = "2" + "4";  document.write(x);  document.write("<br>");
    x = 2+ "4";   document.write(x);  document.write("<br>");
    x = "2" + 4;  document.write(x);  document.write("<br>");
</script>
```

KONVERZIJA U STRING

- 6
- 24
- 24
- 24
- Broj se uvek konvertuje u String pri konkatenaciji sa Stringom.

LOGIČKI TIP

- Logički tip podataka obuhvata dve vrednosti **true (tačno)**

  i **false (netačno)**.
- Prilikom rada ako je potrebno može se izvršiti konverzija logičke vrednosti **true** u broj **1** i vrednosti **false** u broj **0**.

KONVERZIJE PODATAKA

- Kod JavaScript jezika promenljiva može da ima različite tipove podataka

  u različito vreme izvršavanja programa

```
a = 5;  //a je sada celobrojni podatak
b = 8;  //b je sada celobrojni podatak
b = "broj"+ a;
/* b je sada String podatak, zato što se na String "broj" nadovezuje ceo broj, pa se dobija String broj5   !*/
```

NULL VREDNOST

- Vrednost null je
  - vrednost koja se može dodeliti promenljivoj
  - promenljiva koja nema vrednost
  - dodeljena promenljivoj kada želimo da definišemo da promenljiva ne sadrži nikakav podatak

OPERATORI

- Operatori su specijalni karakteri, koji definišu operaciju koja treba da se izvrši nad operandima, koji mogu biti promenljive, izrazi ili konstante.
  - Aritmetički
  - Na nivou bita
  - Relacionalni
  - Logički

ARITMETIČKI

- Koriste se za matematičke operacije.
- Ukoliko je jedan od operanada tipa String za sve operatore, osim za sabiranje, pokušaće se da se izvede konverzija Stringa u broj i da se tako izvrši definisana operacija. Ako se ne uspe kao rezultat se dobija specijalna vrednost NaN (Not A Number).
- Izuzetak kod sabiranja: podatak koji nije tipa String konvertuje se u String i izvršava se sabiranje dva Stringa.
- a=24; b="broj "+ a; //dobija se da je b: broj 24

ARITMETIČKI

Operator	Opis	Operator	Opis
+	sabiranje	+=	sabiranje dodela
-	oduzimanje	-=	oduzimanje dodela
*	množenje	*=	množenje dodela
/	deljenje	/=	deljenje dodela
%	moduo	%=	moduo dodela
++	inkrement	--	dekrement

OPERATORI NA NIVOU BITA

Operator	Upotreba	Opis
Logičko I (AND)	a & b	Rezultat je 1, jedino ako su oba bita 1, u ostalim slučajevima rezultat je 0.
Logičko ILI (OR)	a | b	Rezultat je 0, jedino ako su oba bita 0, u ostalim slučajevima rezultat je 1.
Logičko ekskluzivno ILI (XOR)	a ^ b	Rezultat je 1, ako biti imaju različite vrednosti, u slučaju da imaju iste vrednosti, rezultat je 0.
Logičko NE (NOT)	~ a	Komplementira bitove operanda a.
Pomeranje ulevo	a << b	Pomera binarni sadržaj operanda a za b mesta ulevo. Prazna mesta popunjava sa vrednošću 0.
Pomeranje udesno sa znakom	a >> b	Pomera binarni sadržaj operanda a za b mesta udesno. Prazna mesta popunjavju se vrednošću najstarijeg bita.
Pomeranje udesno sa nulama	a >>> b	Pomera binarni sadržaj operanda za b mesta udesno. Prazna mesta popunjavaju se vrednošću 0.

NA NIVOU BITA

- 15 & 9 rezulat 9 (1111 & 1001 = 1001)
- 15 | 9 rezulat 15 (1111 | 1001 = 1111)
- 15 ^ 9 rezulat 6 (1111 ^ 1001 = 0110)

LOGIČKI

Operator	Upotreba	Opis
I (&&)	expr1 && expr2	Rezultat je true, jedino ako su oba operanda true, u ostalim slučajevima rezultat je false.
ILI (||)	expr1 || expr2	Rezultat je false, jedino ako su oba operanda false, u ostalim slučajevima rezultat je true.
NE (!)	!expr	Rezultat je komplement vrednosti operanada. Ako je operand true, rezultat je false, ako je operand false, rezultat je true

Imaju vrednosti: true i false

Ovi operatori imaju veliku primenu u okviru kontrola toka.

LOGIČKI

```
a = true;
b = false;
c = a || b;
d = a && b;
f = (!a && b) || (a && !b);
g = !a;
document.write( " a = " + a + "<BR>" );
document.write ( " b = " + b + "<BR>" );
document.write ( " c = " + c + "<BR>" );
document.write ( " d = " + d + "<BR>" );
document.write ( " f = " + f + "<BR>" );
document.write ( " g = " +  g);
```

Rezultat izvršavanja je:
a = true
b = false
c = true
d = false
f = true
g = false

OPERATORI POREĐENJA

Operator	Upotreba	Opis
Jednakost (==)	Rezultat je true ako su operandi jednaki	x == y rezultat je true ako su x i y jednaki.
Nejednakost (!=)	Rezultat je true ako su operandi različiti.	x != y rezultat je true ako su x i y različiti.
Veće (>)	Rezultat je true ako je levi operand veći od desnog operanda.	x > y rezultat je true ako je x veće od y.
Veće ili jednako (>=)	Rezultat je true ako je levi operand veći ili jednak desnom operandu	x >= y rezultat je true ako je x veće ili jednako y.
Manje (<)	Rezultat je true ako je levi operand manji od desnog operanda	x < y rezultat je true ako je x manje od y.
Manje ili jednako (<=)	Rezultat je true ako je levi operand manji ili jednak desnom operandu	x <= y rezultat je true ako je x manje ili jednako y.
Jednako bez konverzije tipova (===)	Rezultat je true ako su operandi jednaki bez konverzije podataka	x === y rezultat je true ako su x i y jednaki bez konverzije podataka
Različito bez konverzije tipova (!==)	Rezultat je true ako su operandi različiti bez konverzije podataka	x !== y rezultat je true ako su x i y različiti bez konverzije podataka

OPERATORI POREĐENJA

5 ==“5” true

5===“5” false

a = 4;

b = 1;

c = a < b; false

d = a == b; false

TERNARNI OPERATOR

- expression ? statement1 : statement2
- gde je izraz expression bilo koji izraz čiji rezultat je vrednost logičkog tipa. Ako je rezultat izraza true, onda se izvršava statement1, u suprotnom statement2.
- ratio = denom ==0 ? 0 : num / denom

KONTROLE TOKA - IF

```
if (boolean_izraz) blok1;
        [else blok2;]

if (x == 8) {
                    y=x;
                      }  else {   z=x;
                          y=y*x
              }
```

SWITCH

```
switch (izraz) {
  case vr1: blok1; [break];
  ...
  case vrN: blokN; [break];
  [default: blok_def]
}
```

```
if (mesec == 1)
        ime_meseca = "Januar"
else if (mesec == 2)
        ime_meseca = "Februar"
else if (mesec == 3)
        ime_meseca = "Mart"
else if (mesec == 4)
        ime_meseca = "Maj"
else
        ....
else if (mesec == 12)
        ime_meseca = "Decembar"
```

SWITCH

```
switch(mesec) {
 case 1: ime_meseca = “Januar”; break;
 case 3: ime_meseca = “ Mart”; break;
 case 5: ime_meseca = “Maj”; break;
 case 7: ime_meseca = “Jul”; break;
 case 8: ime_meseca = “Avgust”; break;
 case 10: ime_meseca = “Oktobar”; break;
 case 12: ime_meseca = “Decembar”; break;
 case 4: ime_meseca = “ April ”; break;
 case 6: ime_meseca = “Jun”; break;
 case 9: ime_meseca = “Septembar”; break;
 case 11: ime_meseca = “Novembar”; break;
 case 2:  ime_meseca = “ Februar ”;
}
```

WHILE

```
[inicijalizacija;]
while(uslov_ostanka){
    telo_petlje;
}
```

```
i=1
while(i<=10){
    document.write(i+ "<br>");
  i=i+1;
}
```

DO-WHILE

```
[inicijalizacija]
   do {
        telo_petlje
        [iteracija]
   } while (uslov);
```

```
i=1
      do {
         document.write(i+ "<br>");
          i=i+1;
   } while(i<=10)
```

FOR

```
for(inicijalizacija; uslov; iteracija) {
    telo_petlje;
}
```

```
for(i=1; i<=10; i++) {
 document.write(i+ "<br>");
}
```

BREAK

```
a:  {
        b:     {
        c:        {
            document.writeln("pre break-a");
            break b;
            document.writeln("ovo nece biti prikazano");
         }
      }
      document.writeln("posle break-a");
   }
```

Izlaz: pre break-a
posle break-a

CONTINUE

```
for(i=0; i<10; i++)  {
                document.write(i+ " ");
                        if (i%2 ==0)
                    continue;
                document.writeln(" ");
                 }
```

```
   0 1
 2 3
 4 5
 6 7
 8 9
```

RETURN

```
function kvadratBroja(x) {
return x * x;
}
x = kvadratBroja(5);
/* poziv funkcije */
document.write("Kvadrat od 5 je " + x);
```

FOR IN

```
niz = new Array (“Pera",“Mika",“Zika")
for (var i in niz) {
document.write(niz[i] + "<BR>");
}
```

FUNCTION

```
function ime([param1] [, param2] [...,paramN])
{
//izrazi
}
```

WITH

```
var a, x, y;
var r=10;
with (Math) {
   a = PI * r * r;
   x = r * cos(PI);
   y = r * sin(PI/2);
}
```

Definiše tip objekta za niz izraza. U okviru izraza dodeljuje specifične vrednosti za određene osobine objekta. Na primer, matematičkim funkcijama mora prethoditi objekat Math. Sledeći primer podrazumeva Math ispred PI, COS() i SIN():

NIZOVI

- Sadrže skup podataka definisanih u jednoj promenljivoj.
- Da bi se kreirao niz koristi se objekat **Array()**
- Poziva se konstruktor, koristi se za kreiranje instance promenljive i vraća referencu na kreiranu promenljivu.

ARRAY()

- Niz se kreira pomoću reči new i konstruktora Array() na sledeći način:

```
var arrayName = new Array();
```

- ili inicijalizovan elementima:

```
arrayObjectName = new Array(element0, element1,..., elementN);
```

- Svaki podatak u nizu se naziva element.

POZICIJA U NIZU

- Indeks je numerička pozicija u nizu.
- Brojanje elemenata u okviru niza počinje sa indeksom nula (0).
- Pojedinačnom elementu se pristupa tako što se navodi njegov indeks u srednjim zagradama.
- Dodeljivanje vrednosti pojedinačnom članu niza:

  niz[3] = “Zika"
- Veličina niza se može dinamički menjati.

PRIMER NIZA 1

```
var auto = new Array(); //definisanje niza
auto[0] = "Saab";
auto[1] = "Volvo";
auto[2] = "BMW";
for (i=0;i<auto.length;i++)
{
document.write(auto[i] + " ");
}
```

Izlaz: Saab Volvo BMW

PRIMER NIZA 2

```
var auto = new Array(2); //niz od 2 elementa
auto[0] = "Fiat";
auto[1] = "Peugeot";
auto[2] = "Citroen";  //niz ce da se prosiri dinamicki
auto[3] = "Skoda";
for (i=0; i<auto.length; i++)
{
document.write(auto[i] + " ");
}
```

Izlaz: Fiat, Peugeot, Citroen, Skoda

PRIMER NIZA 3

```
var auto = new Array("Volkswagen", "Ford", "Mercedes");
//definisemo niz od 3 elementa

auto[1] = "Opel";  //menjamo drugi element niza
for (i=0;i<auto.length;i++)
{
document.write(auto[i] + " ");
}
```

Izlaz: Volkswagen, Opel, Mercedes

FUNKCIJA SORT()

- Ova metoda uređuje (sortira) elemente niza direktno u izvornom nizu i vraća tako uređen niz.
- Kada se metoda sort() pozove bez argumenata, sortira elemente niza po abecednom redosledu.
- Ako niz sadrži nedefinisane elemente, oni se stavljaju na kraj niza.

```
var niz = new Array("Marko", "Vesna", "Ana", "Stefan", "Darija", "Ivan");
document.write(niz + "<br />")
document.write(niz.sort() + "<br />")
```

Izlaz: Marko,Vesna,Ana,Stefan,Darija,Ivan

Ana,Darija,Ivan,Marko,Stefan,Vesna

SORT() - PO NUMERIČKOM REDU

- Da bi se sortirao niz po redosledu koji nije abecedni, metodi sort() prosledjuje se kao argument funkcija za poređenje.

```
function sortNumber(a, b){
return a -b;
} //vraca vrednost <0, 0 ili >0, zavisno od redosleda
var numeric= new Array[3, 44, 1111, 222];
document.write(numeric.sort() + "<br />");
document.write(numeric.sort(sortNumber));
//prvo je abecedno, drugo numericko sortiranje!
```

Izlaz:

1111, 222, 3, 44

3, 44, 222, 1111

REVERSE()

- Ova metoda obrće redosled elemenata niza i vraća niz sa obrnuto raspoređenim elementima. Da bi to uradila, ne pravi novi niz s preuređenim elementima, već menja redosled direktno u postojećem nizu.

  a[0] postaje a[n], a[1] postaje a[n-1],...

  ```
  var niz = new Array("Marko", "Vesna", "Ana", "Stefan", "Darija", "Ivan");
  document.write(niz + "<br />")
  document.write(niz.reverse() + "<br />")
  ```

  Izlaz:

  Marko,Vesna,Ana,Stefan,Darija,Ivan

  Ivan,Darija,Stefan,Ana,Vesna,Marko

CONCAT()

- Metoda concat() pravi i vraća nov niz koji sadrži elemente izvornog niza, s pridodatim argumentima te funkcije.
- Ako je neki od ovih argumenata niz, on se razlaže na svoje elemente koji se zasebno pridodaju rezultujećem nizu.

```
var brojevi=[1,2,3];
brojevi.concat(4,5); //Rezultat: 1,2,3,4,5
brojevi.concat([4,5]);//Rezultat: 1,2,3,4,5
brojevi.concat([4,5],[6,7]);  //Rezultat: 1,2,3,4,5,6,7
brojevi.concat(4, [5,[6,7]]); //Rezultat: 1,2,3,4,5,6,7
```

JOIN()

- Metoda join() konvertuje sve elemente niza u znakovne nizove i nadovezuje ih.
- Ukoliko se ne navede nijedan graničnik u obliku znakovnog niza, za razdvajanje se koristi zarez.

```
var brojevi =[1,2,3]; // Pravi novi niz sa ova 3 elem.
var s = brojevi.join(); // Rezultat: s=1,2,3
s = brojevi.join(" | "); // Rezultat: s = 1 | 2 | 3
s = brojevi.join("#");  // Rezultat: s = 1#2#3
```

SLICE()

- Metoda slice() vraća isečak, odnosno podniz navedenog niza. Ima dva argumenta koja određuju početak i kraj isečka koji se dobija.
- Rezultujući niz sadrži element određen prvim argumentom, i sve naredne elemente sve do elementa (ali ne i njega) određenog drugim argumentom.
- Ako je naveden samo jedan argument, rezultujući niz sadrži sve elemente počev od onog predviđenog tim argumentom, do kraja niza. Ako je negativan, gleda se od poslednjeg.

```
var brojevi =[1,2,3,4,5];
brojevi.slice(0,3);//Rezultat: 1,2,3
brojevi.slice(3);//Rezultat: 4,5
brojevi.slice(1, -1);//Rezultat: 2,3,4
```