Uvod u JavaScript

JavaScript varijante

- JavaScript je prvenstveno korišćen za klijentsko programiranje
  - Client-side JavaScript
  - Izvršava se u browseru klijenta
- Postoji i JavaScript kod koji se izvršava na serveru
  - Server-side JavaScript , izvršava se na serveru i kreira dinamički sadržaj koji se šalje do browsera
  - **Node.js** serverski Java script, nastao 2009. godine
  - **Node.js** je serverska platforma bazirana na script mašini Google chrome browsera
  - Godine 2010 napravljen je menadžer paketa za Node.js pod nazivom **npm**

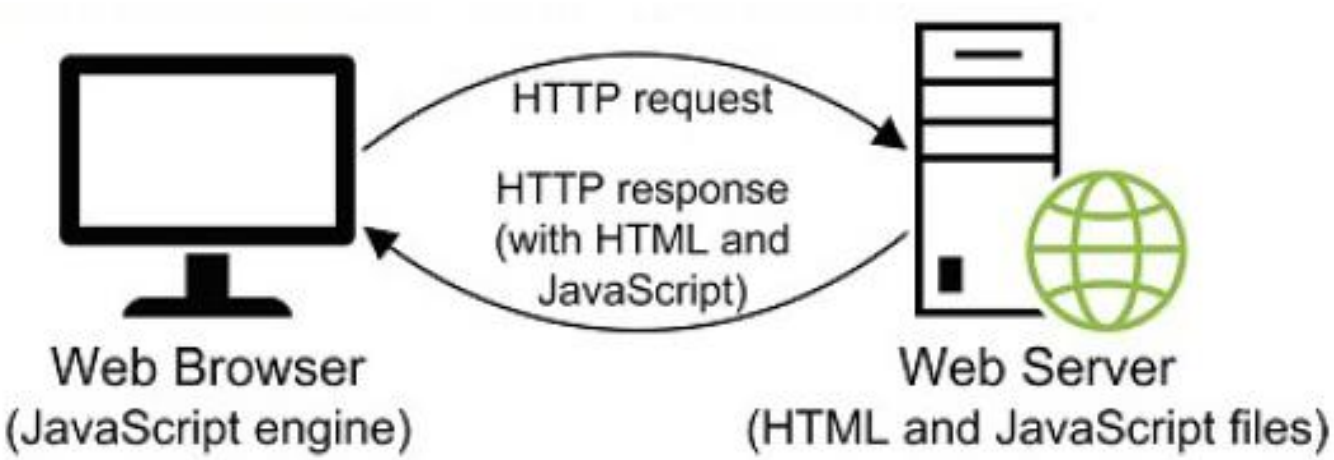
TypeScript

- TypeScript je programski jezik otvorenog koda razvijen od strane Microsofta, 2012. godine
- Kompajlira se u čist JavaScript
- Tekuća verzija TypeScript 5.9
- TypeScript je programski jezik koji predstavlja proširenje javascripta, dodaje tipove podataka, klase i module u JavaScript
- Koristi se za kreiranje javascript koda koji se može izvršavati i na klijentu i na serveru

Klijentski JavaScript

- Koristi se za izradu klijentskih delova web aplikacija
- JavaScript koristi sintaksu sličnu jeziku C
- JavaScript kod se izvršava interpretacijom i delimičnom JIT kompajlacijom (za brže izvođenje)
- JavaScript je programski jezik koji podržava:
  - Promenljive za čuvanje informacija
  - Operatore za izvođenje operacija i poređenje
  - Funkcije za grupisanje koda u celine koje se mogu više puta pozivati
  - Uslovne izraze i programske petlje za kontrolu programskog toka
- Mogućnost kreiranja objekata sa svojstvima, metodama i događajima

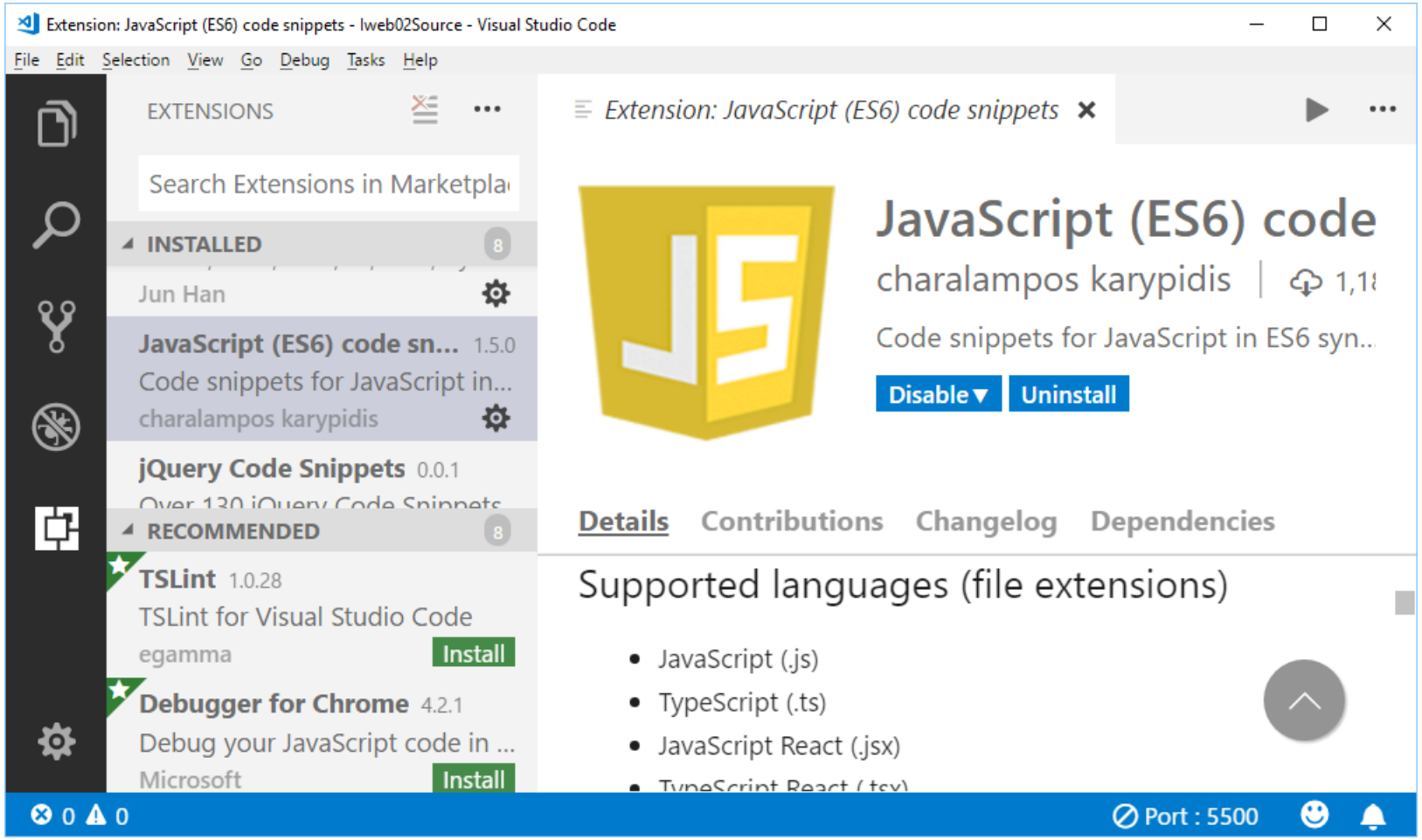
Izvršavanje klijentskog javascript koda



Funkcionisanje JavaScripta

- JavaScript se koristi u kombinaciji sa objektnim modelom dokumenta (DOM) da bi se web strana učinila dinamičkom
- Kada se HTML dokument učita u browser kreira se njegov objektni model, odnosno objekat pod nazivom ***document***
- Objekat ***document*** je deo objekta ***window*** koji predstavlja prozor browsera
- JavaScript može modifikovati web stranu kada se ona učitava u browser ili kao odgovor na akciju korisnika

Ekstenzija za JavaScript



JavaScript sintaksa

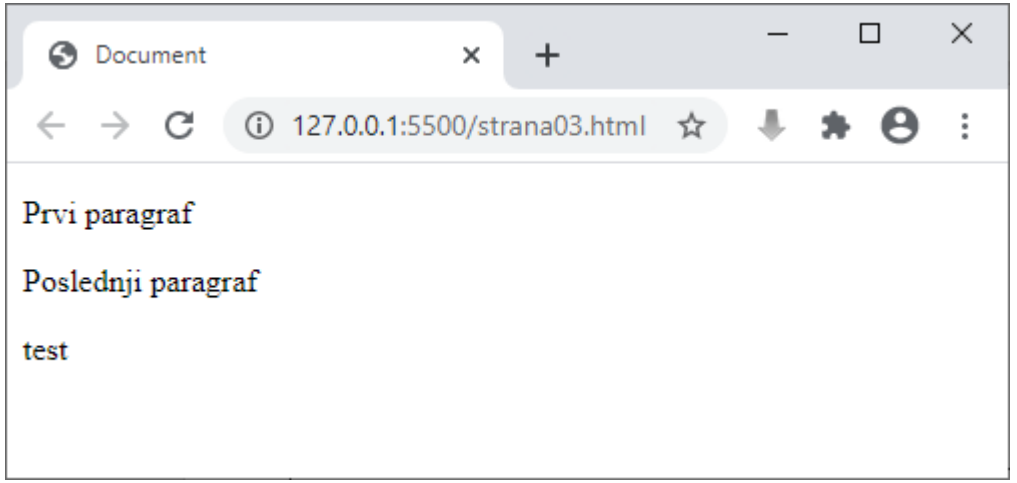
- JavaScript naredba predstavlja liniju JavaScript koda koji treba da se izvrši
- Naredba u JavaScriptu treba biti napisana u jednoj liniji i treba da se završava operatorom operatorom tačka-zarez(;)
- JavaScript naredbe se grupišu unutar bloka { }
- JavaScript je osetljiv na velika i mala slova

Pisanje JavaScript koda

- Ubacivanje JavaScript koda u HTML dokument
  - u **<body>** delu HTML dokumenta
  - u **<head>** delu HTML dokumenta
- U zasebnom **.js** dokumentu i njegovo referenciranje upotrebom src atributa
- Uzasebnom **.js** dokumentu na mreži i njegovo referenciranje upotrebom **src** atributa
- Preferira se pisanje JavaScript koda neposredno pre završnog **<body>** taga radi bržeg učitavanja strane
- HTML oznaka <script> koristi se za ubacivanje i izvršavanje JavaScript koda u okviru HTML dokumenta

JavaScript kod u **<body>** delu strane

```
<body>
    <p>Prvi paragraf</p>
    <p>Poslednji paragraf</p>
    <script>
        document.write("test");
    </script>
</body>
```



Napomena: document.write() se koristi u jednostavnim primerima i edukativne svrhe, ali se u savremenom razvoju **retko upotrebljava** jer može prebrisati sadržaj stranice nakon učitavanja

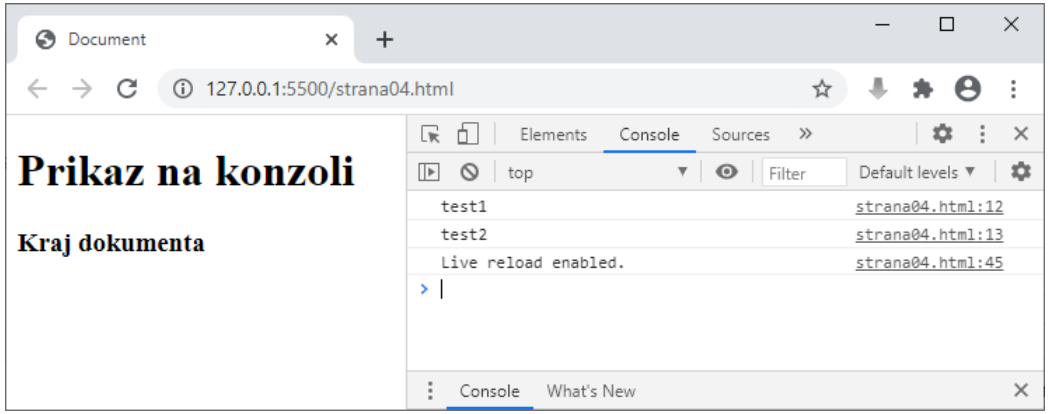
Objekat console

- Objekat console omogućava pristup developerskoj konzoli
- Konzola se otvara pomoću tastera Ctrl + Shift + I (ili F12), zatim izborom kartice **Console**
- Metoda **log**() ovog objekta ispisuje podatke u konzoli
- Metoda **log**() može primiti više argumenata odvojenih zarezom

```
console.log(vrednost1, vrednost2, ...);
```

Prikaz rezultata u konzoli browsera

```
<body>
    <h1>Prikaz na konzoli</h1>
    <script>
        //clg
        console.log("test1");
        console.log("test2");
    </script>
    <h3>Kraj dokumenta</h3>
</body>
```



Referenciranje eksternog .js fajla

- Prednost korišćenja eksternog fajla je bolja organizacija koda i mogućnost ponovne upotrebe u više HTML dokumenata
- Oznaka <script> sa atributom src se koristi za povezivanje eksternog JavaScript fajla
- Eksterni fajl **"primer01.js"**

```
console.log("Skripta uspešno učitana!");
```

- HTML deo

```
<body>
    <script src="primer01.js"></script>
    <p>Kraj dokumenta</p>
</body>
```

Komentari

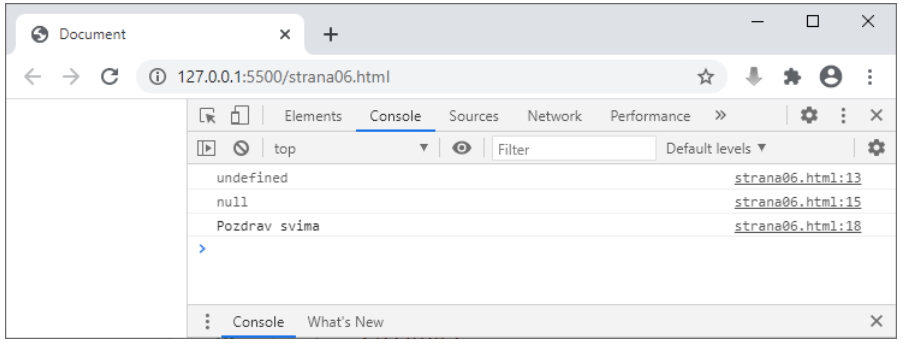
```
<script>
    // prikaz poruke korisniku
    console.log("Dobar dan svima");

    /*
        Možete koristiti multi-line komentar
        da biste dodali više informacija
    */

    alert("Dobar dan svima, također!");
</script>
```

Promenljive

- Promenljiva čuva podatke u memoriji
- Promenljive u JavaScript-u počinju slovom ili donjom crtom (_)
- Koriste se ključne reči **let** i **const** za deklarisanje promenljivih
- **let** se koristi kada se vrednost promenljive može menjati
- **const** se koristi kada promenljiva ne treba da dobije novu vrednost
- Kod promenljive deklarisane pomoću const, vrednost se ne može ponovo dodeliti, ali se sadržaj promenljive (ako je objekat ili niz) može menjati
- Promenljive deklarisane pomoću let i const imaju blokovski opseg
- Starija sintaksa koristi ključnu reč var (i dalje funkcioniše, ali se danas ređe koristi)
- Imena promenljivih su case-sensitive (razlikuju velika i mala slova)
- Promenljiva može biti nedefinisana (undefined) ili imati vrednost null

Upotreba kljune reči **let**

```
<script>
  let x;
  console.log(x); // undefined
  let y = null;
  console.log(y); // null
  let poruka = "Pozdrav svima";
  console.log(poruka);
</script>
```



Razlika između **let** i const

```
<script>
  let broj = 5;
  const PI = 3.14;

  broj = 10; // Dozvoljeno
  // PI = 3.14159; // Greška – const ne može da promeni vrednost

  console.log(broj);
  console.log(PI);
</script>
```

Tipovi podataka

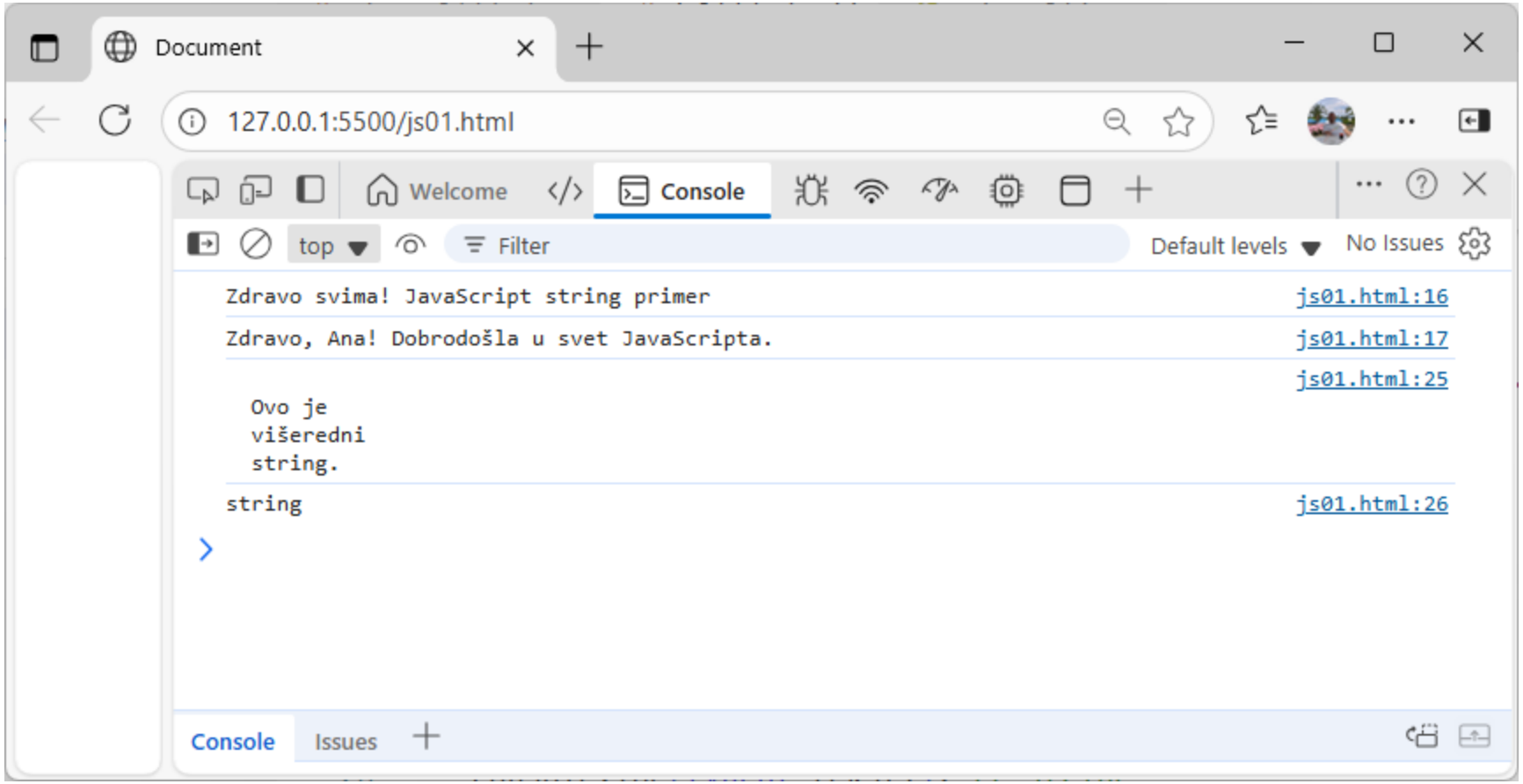
- JavaScript koristi dinamičko tipiziranje – tip promenljive se određuje tokom izvršavanja
- Promenljive mogu čuvati različite tipove podataka
- Osnovni (primitivni) tipovi podataka su:
  - **Number** – brojevi (celi i decimalni)
  - **String** – tekstualne vrednosti
  - **Boolean** – true / false
  - Undefined – promenljiva je deklarisana, ali nema vrednost
  - Null – prazna ili nepostojeća vrednost
  - BigInt – vrlo veliki brojevi
  - Symbol – jedinstveni identifikatori
  - Object (nizovi, funkcije, datumi itd.)

JavaScript stringovi

- String se može definisati pomoću dvostrukih (") ili jednostrukih (') navodnika
- Operator + se koristi za spajanje stringova
- Template literal (označava se pomoću backtick znaka `) omogućava:
  - Umetanje promenljivih pomoću ${ }
  - Višeredne stringove
- Tip promenljive se može proveriti pomoću operatora typeof

```
<script>
  let tekst1 = "Zdravo svima!";
  let tekst2 = 'JavaScript string primer';
  let ime = "Ana";

  console.log(tekst1 + " " + tekst2);
  console.log(`Zdravo, ${ime}! Dobrodošla u svet JavaScripta.`);

  // Višeredni string pomoću backtick-a
  let poruka = `
  Ovo je
  višeredni
  string.`;

  console.log(poruka);
  console.log(typeof tekst1); // string
</script>
```

JavaScript stringovi

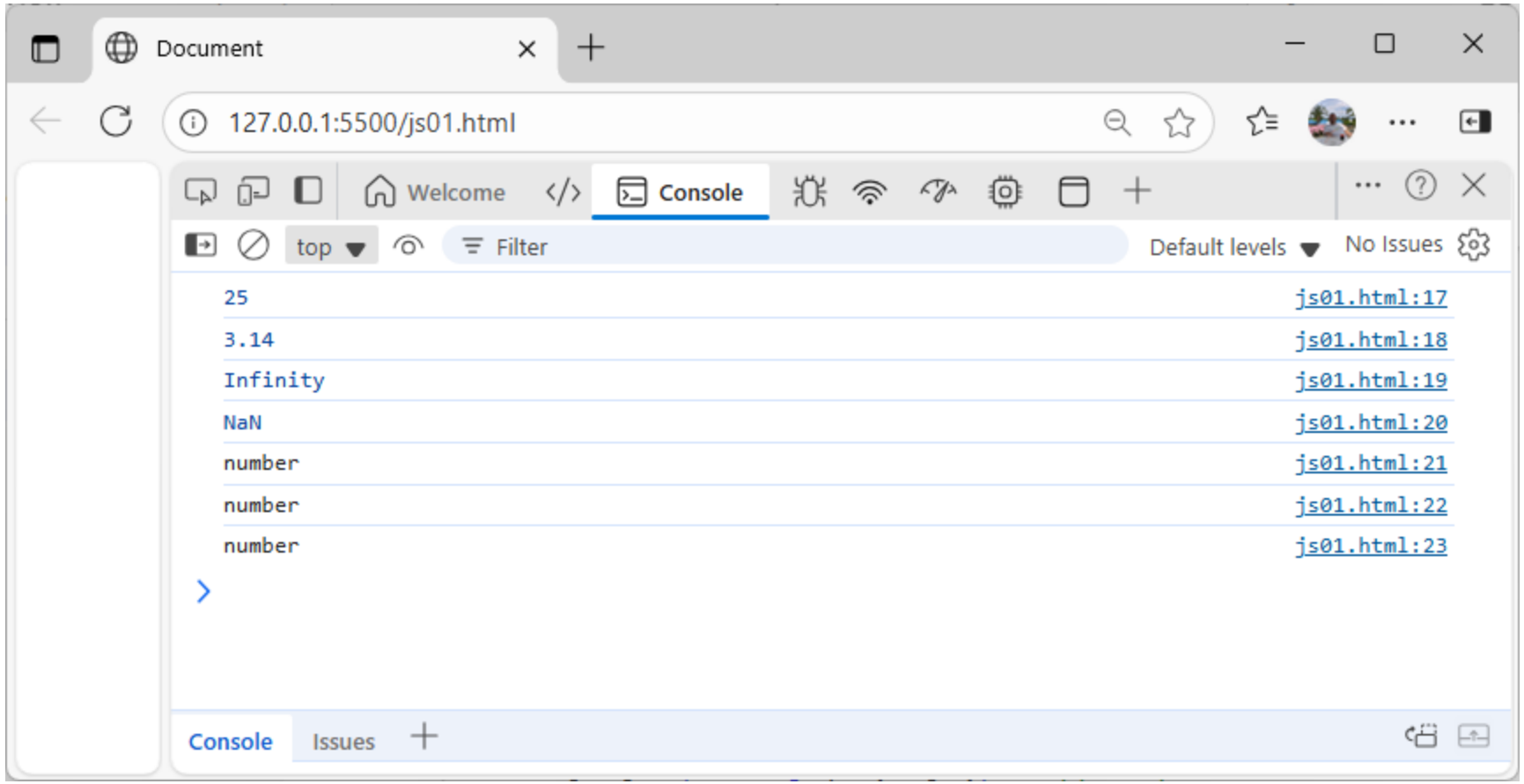


JavaScript brojevi

- JavaScript koristi samo jedan tip za sve brojeve – **Number**
- Brojevi mogu biti celi ili decimalni
- Decimalni deo se odvaja tačkom (.), ne zarezom
- Posebne numeričke vrednosti su: **Infinity**, **-Infinity** i **NaN** (Not a Number)
- **Infinity** je posebna numerička vrednost koja predstavlja beskonačnost
- **NaN** označava neispravnu numeričku vrednost (Not a Number), ali je i dalje tipa number

```
<script>
  let ceo = 25;
  let decimalni = 3.14;
  let besk = Infinity;
  let neBroj = "tekst" / 2;

  console.log(ceo);
  console.log(decimalni);
  console.log(besk);
  console.log(neBroj);
  console.log(typeof ceo);         // number
  console.log(typeof decimalni);   // number
  console.log(typeof neBroj);      // number (iako nije broj, rezultat je NaN)
</script>
```

JavaScript brojevi

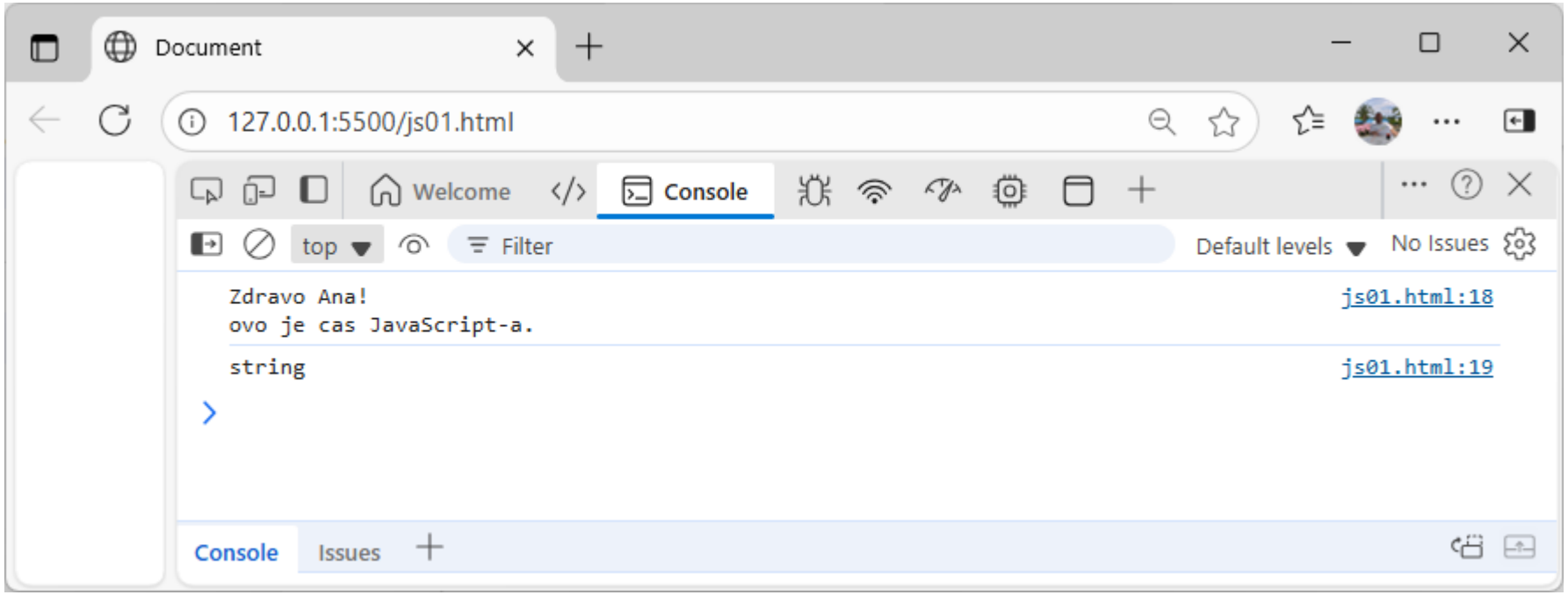


Templejt literali(šablonski stringovi)

- Označavaju se pomoću backtick znaka (`)
- Omogućavaju umetanje promenljivih unutar stringa pomoću ${ }

```
<script>
  let ime = "Ana";
  let predmet = "JavaScript";

  let poruka = `Zdravo ${ime}!
                ovo je cas ${predmet}-a.`;

  console.log(poruka);
  console.log(typeof poruka); // string
</script>
```

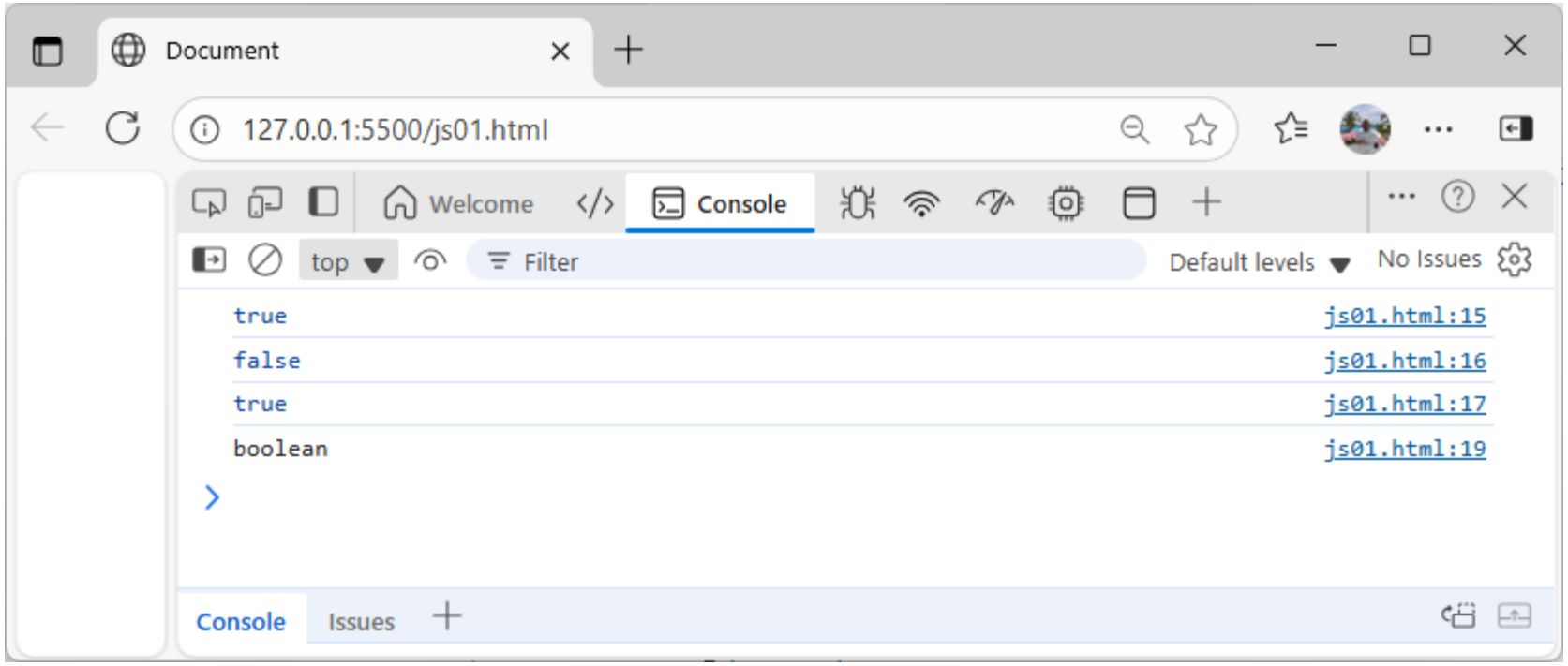
Templejt literali(šablonski stringovi)



JavaScript logičke vrednosti (Booleans)

- Logički tip podataka ima samo dve vrednosti: true i false
- Koristi se za logička poređenja i uslove u grananju (if, while itd.)
- Rezultat poređenja (>, <, ==, ===, !=, !==) uvek je true ili false

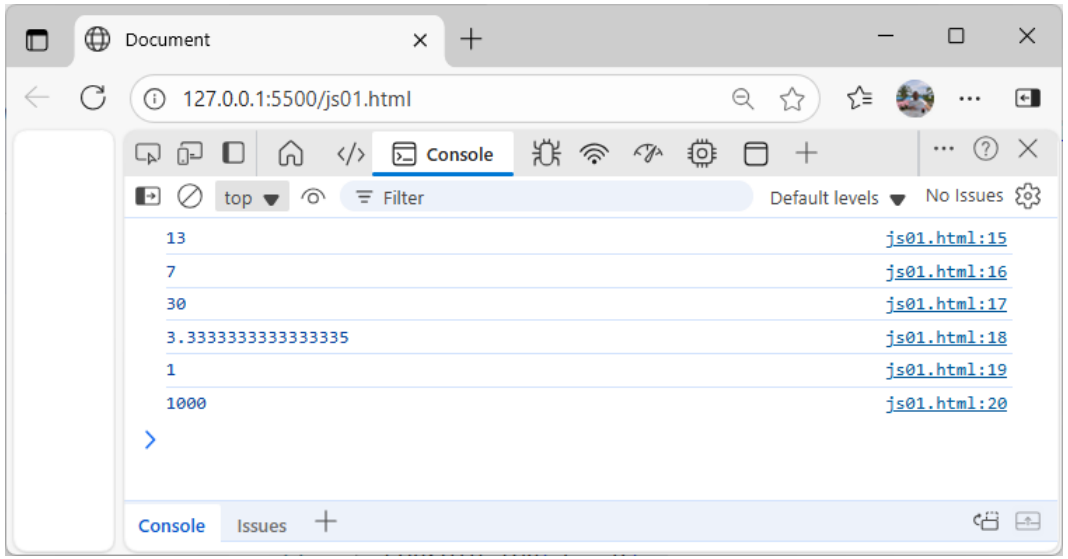
```
<script>
  let a = 10;
  let b = 5;

  console.log(a > b);   // true
  console.log(a < b);   // false
  console.log(a == 10); // true
  console.log(typeof (a > b)); // boolean
</script>
```



Aritmetički operatori

- Sabiranje (+)
- Oduzimanje (-)
- Množenje (*)
- Deljenje (/)
- Celobrojni ostatak (%)
- Inkerementiranje (++)
- Dekrementiranje (--)
- Operator ** se koristi za stepenovanje (npr. 2 ** 3 = 8)

```
<script>
  let a = 10;
  let b = 3;

  console.log(a + b);
  console.log(a - b);
  console.log(a * b);
  console.log(a / b);
  console.log(a % b);
  console.log(a ** b);
</script>
```



Operatori dodeljivanja

- Operator dodele se koristi za dodeljivanje vrednosti promenljivoj.
- Osnovni operator je **=**
- Postoje i kombinovani oblici koji spajaju dodelu i aritmetičku operaciju

Operator	Značenje	Primer	Rezultat
=	Dodela vrednosti	x = 5	x postaje 5
+=	Saberi i dodeli	x += 2	isto kao x = x + 2
-=	Oduzmi i dodeli	x -= 2	isto kao x = x - 2
*=	Pomnoži i dodeli	x *= 3	isto kao x = x * 3
/=	Podeli i dodeli	x /= 2	isto kao x = x / 2
%=	Ostatak i dodeli	x %= 2	isto kao x = x % 2
**=	Stepenovanje i dodela	x **= 2	isto kao x = x ** 2

Operatori poređenja

- Koriste se za poređenje dve vrednosti.Rezultat poređenja je uvek logička vrednost: true ili false
- Uglavnom se koriste u uslovima (if, while, for)

Operator	Značenje	Primer	Rezultat
==	Poređenje po vrednosti (ignoriše tip)	`5 == "5"`	`true`
===	Poređenje po vrednosti i tipu	`5 === "5"`	`false`
!=	Različito po vrednosti	`5 != "5"`	`false`
!==	Različito po vrednosti i tipu	`5 !== "5"`	`true`
>	Veće od	`8 > 5`	`true`
<	Manje od	`3 < 2`	`false`
>=	Veće ili jednako	`7 >= 7`	`true`
<=	Manje ili jednako	`4 <= 5`	`true`

Logički operatori

- Koriste se za kombinovanje logičkih izraza
- Rezultat je uvek true ili false
- Najčešće se koriste u uslovnim izrazima (if, while, for)

Operator	Naziv	Opis	Primer	Rezultat
`&&`	**i** (AND)	Tačno ako su oba izraza tačna	`(5 > 3 && 8 > 6)`	`true`
`||`	**ili** (OR)	Tačno ako je bar jedan izraz tačan	`(5 > 3 || 8 < 6)`	`true`
!	**ne** (NOT)	Inverzija logičke vrednosti izraza	`!(5 > 3)`	`false`

Rad sa operandom tipa string

- Ako se koristi operator + između stringa i broja, broj se automatski pretvara u string (konverzija tipa)
- Operacija + sa stringovima uvek znači spajanje (konkatenacija), a ne sabiranje
- Redosled operanada je važan — ako je prvi operand string, ostali se takođe konvertuju u string
- Kada se koriste aritmetički operatori razliciti od +, JavaScript pokušava da konvertuje string u broj
  - Ako konverzija uspe – izvršava se matematička operacija
  - Ako konverzija ne uspe – rezultat je NaN (Not a Number)

```
<script>
  console.log("5" + 2);   // "52"  (broj 2 je pretvoren u string)
  console.log(5 + "2");   // "52"  (isti rezultat)
  console.log("5" + 2 + 3); // "523" (sve postaje string)
  console.log(5 + 2 + "3"); // "73"  (prvo sabiranje, zatim spajanje)
</script>
```

```
<script>
  console.log("10" - 2);   // 8
  console.log("10" * 2);   // 20
  console.log("10" / 2);   // 5
  console.log("10a" - 2);  // NaN (nije moguće konvertovati)
</script>
```

Uslovni (ternarni) operator

- Koristi se kao kraći zapis if...else izraza.
- Ima oblik:
  uslov ? izraz_ako_je_tačno : izraz_ako_je_netačno
- Prvo se proverava uslov:
  - ako je true, izvršava se prvi izraz,
  - ako je false, izvršava se drugi izraz
- Vraća vrednost — može se dodeliti promenljivoj

```
<script>
  let broj = Math.floor(Math.random() * 21) - 10;
  let rezultat = (broj > 0) ? "Pozitivan broj" : "Negativan broj ili nula";
  console.log("Broj:", broj);
  console.log(rezultat);
</script>
```

Math.random()vraća nasumičan broj između 0 (uključivo) i 1 (isključivo)

Math.floor(x)zaokružuje broj nadole (broj koji nije veći) na najbliži ceo broj

Math.floor(3.9) → 3
Math.floor(-2.1) → -3

Naredba if - uslovno grananje

```
<script>
  let broj = Math.floor(Math.random() * 21) - 10;

  if (broj > 0) {
    console.log("Broj je pozitivan");
  }
</script>
```

if...else – uslovno grananje

```
<script>
  let broj = Math.floor(Math.random() * 21) - 10;

  if (broj >= 0) {
    console.log("Broj je pozitivan ili nula");
  } else {
    console.log("Broj je negativan");
  }
</script>
```

if-else if- else - višestruko grananje

```
<script>
  let broj = Math.floor(Math.random() * 21) - 10;

  if (broj > 0) {
    console.log("Broj je pozitivan");
  } else if (broj < 0) {
    console.log("Broj je negativan");
  } else {
    console.log("Broj je nula");
  }
</script>
```

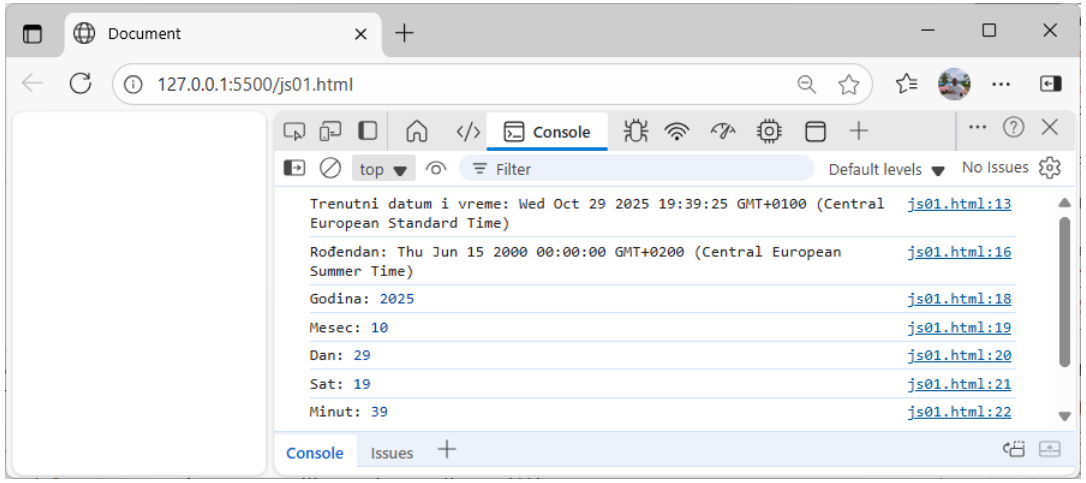
Switch selekcija

```
<script>
  let dan = Math.floor(Math.random() * 7) + 1;
  switch (dan) {
    case 1:
      console.log("Ponedeljak");
      break;
    case 2:
      console.log("Utorak");
      break;
    case 3:
      console.log("Sreda");
      break;
    case 4:
      console.log("Četvrtak");
      break;
    case 5:
      console.log("Petak");
      break;
    default:
      console.log("Vikend");
  }
</script>
```

Rad sa datumom

- JavaScript koristi **Date** objekat za rad sa datumima i vremenom
- Može se kreirati pomoću new Date()
- Kada se pozove bez argumenata, vraća trenutni datum i vreme
- Funkcija Date() može da primi argumente:(godina, mesec, dan, sat, minut, sekund, milisekund)
- Postoje metode za čitanje i podešavanje vrednosti datuma

```
<script>
  let danas = new Date();
  console.log("Trenutni datum i vreme:", danas);

  let rodjendan = new Date(2000, 5, 15); // meseci: 0-11
  console.log("Rođendan:", rodjendan);

  console.log("Godina:", danas.getFullYear());
  console.log("Mesec:", danas.getMonth() + 1); // dodaje se 1 jer meseci kreću od 0
  console.log("Dan:", danas.getDate());
  console.log("Sat:", danas.getHours());
  console.log("Minut:", danas.getMinutes());
</script>
```



While petlja

```
<script>
  let i = 1;

  while (i <= 5) {
    console.log("Broj:", i);
    i++; // uvećavanje brojača
  }
</script>
```

Do-while petlja

```
<script>
let x = 10;
do {
  console.log("Izvršeno bar jednom");
} while (x < 5);
</script>
```

For petlja

```
<script>
  for (let i = 0; i <5; i++) {
    console.log("Broj:", i);
  }
</script>
```

Definisanje funkcije

- Funkcija je blok koda koji se može pozivati više puta sa različitim vrednostima argumenata
- Argumentima funkcije se može pristupiti samo unutar tela funkcije
- Može da prima argumente i vraća rezultat pomoću return naredbe
- Funkcija može definisati lokalnu promenljivu
- Globalne promenljive definisane izvan funkcije vidljive su u svim funkcijama skript sekcije

```
function aName( argument1, argument2, …,argumentN ) {

    statement1;
    statement2;
    ...
    statementN;
}
```

Primer funkcije

```
<script>
  function pomnozi(a, b) {
    console.log(`a = ${a}`);
    console.log(`b = ${b}`);

    return a * b;
  }


  let a1 = 5;
  let b1 = 8;


  let rezultat = pomnozi(a1, b1);
  console.log("Rezultat:", rezultat);


  a1 = 7;
  b1 = 12;
  rezultat = pomnozi(a1, b1);
  console.log("Rezultat:", rezultat);
</script>
```

Oblast važenja promenljivih

- Oblast važenja (scope) određuje gde je promenljiva dostupna u programu.
- U JavaScript-u postoje tri osnovne oblasti važenja:
  - **Globalni** scope – promenljive definisane van funkcija važe svuda.
  - **Funkcijski** scope – promenljive definisane pomoću **var** važe samo unutar funkcije (ne koriste se)
  - **Blokovski** scope – promenljive definisane pomoću **let** i **const** važe samo unutar bloka { }

```
<script>
  let x = 10; // globalna promenljiva

  function test() {
    let y = 5; // lokalna (funkcijska)
    console.log(x); // 10
    console.log(y); // 5
  }

  test();
  console.log(y); // Greška – y nije definisana van funkcije
</script>
```

Funkcija alert()

Prikazuje prozor sa porukom i dugmetom OK
Mora se zatvoriti prozir da bi se koristili drugi delovi dokumenta

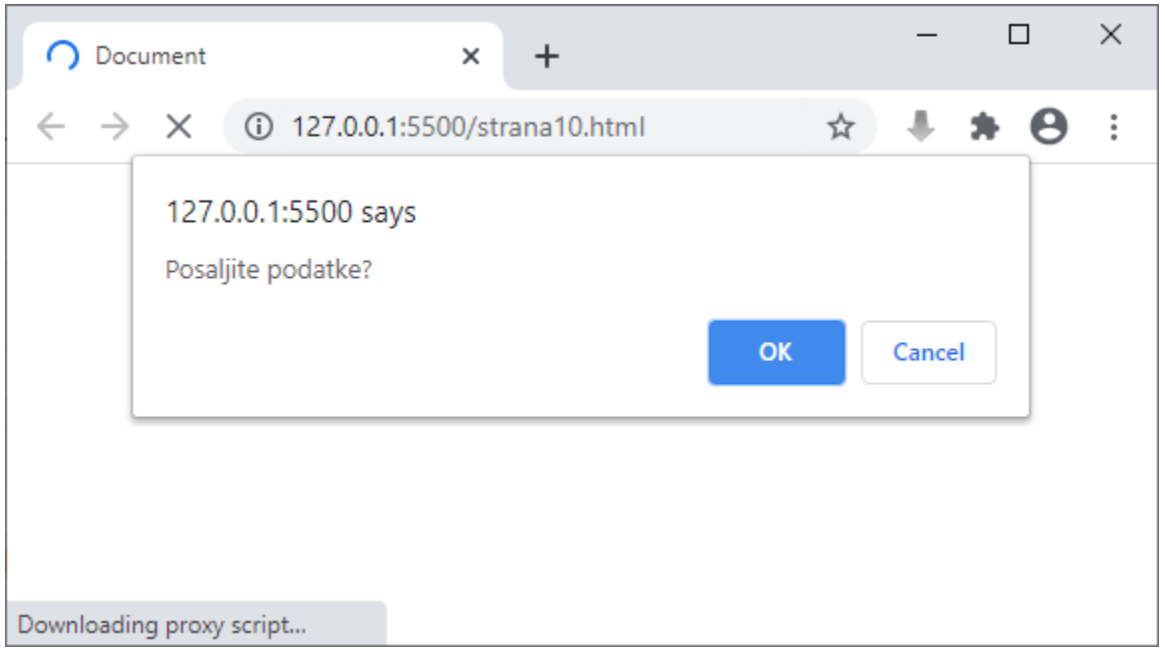
```
<script>
        alert("Pozdrav svima");
        //window.alert("Pozdrav svima");
</script>
```

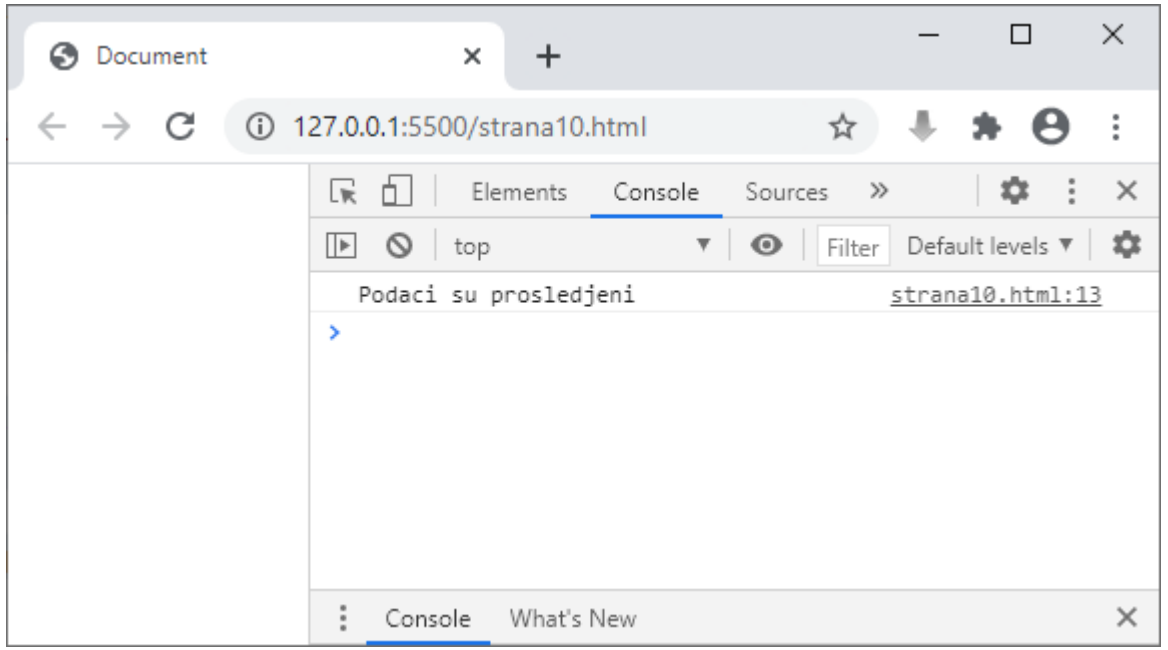
Funkcija confirm()

- Uzima u obzir akciju korisnika
- vraća rezultat (true ili false)

```
<script>
    var rezultat = confirm("Posaljite podatke?");

    if (rezultat) {
        console.log("Podaci su prosledjeni");
    }
    else{
        console.log("Nisu prosledjeni podaci");
    }
</script>
```

Prozor confirm()

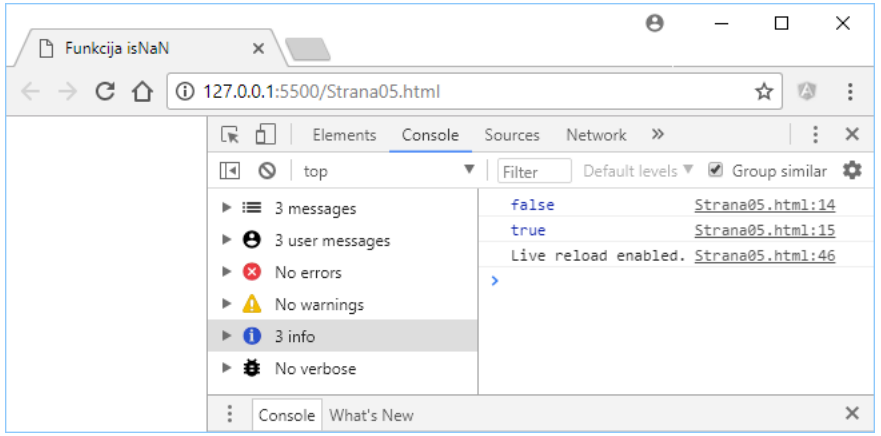




Funkcija isNaN()

- Funkcija isNaN() proverava da li vrednost nije broj (Not a Number)
- Vraća:
  - true → ako vrednost nije broj
  - false → ako vrednost je broj ili može biti konvertovana u broj

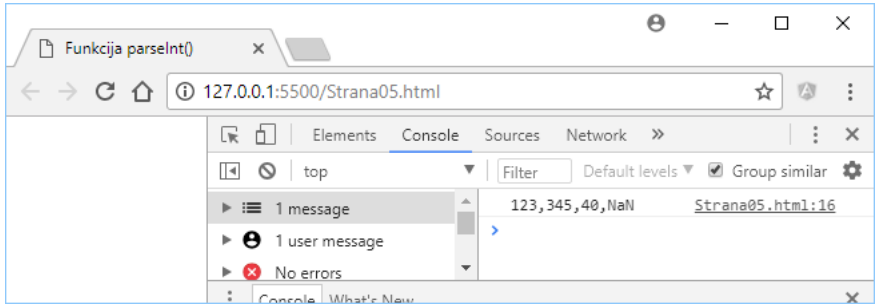
```
<script>
    var a = isNaN(123);
    var b = isNaN("Hello");

    console.log(a);
    console.log(b);
</script>
```



Funkcije parseInt() , parseFloat()

- Koriste se za pretvaranje stringova u brojeve
- parseInt() – konvertuje string u ceo broj (integer)
- parseFloat() – konvertuje string u decimalni broj (float)
- Parsiranje se zaustavlja na prvom znaku koji nije broj
- Ako konverzija nije moguća → rezultat je NaN (Not a Number)

```
<script>
    var c1 = parseInt("123");
    var c2 = parseInt("345.567");
    var c3 = parseInt("40godina");
    var c4 = parseInt("godina40");

    console.log(`${c1},${c2},${c3},${c4}`);

</script>
```



Pitanje 1

Ukoliko html strana u svojoj **<body>** sekciji sadrži javascript kod tada se on izvršava :

a) na serveru
b) na klijentu
c) i na serveru i na klijentu

Odgovor: b

Pitanje 2

Koji od navedenih tipova podataka ne postoji u JavaScript jeziku?

a. float
b. number
c. boolean

Odgovor: a

Pitanje 3

Ključna reč let u JavaScript-u se koristi:

a. za deklarisanje promenljive koja je dostupna samo unutar bloka koda
b. za deklarisanje konstantne vrednosti
c. za deklarisanje promenljive koja je dostupna na nivou funkcija

Odgovor: a

Pitanje 4

Šta se prikazuje na konzoli nakon izvršavanja sledećeg javascript koda:

```
<script>
    let a = 2 * "3";
    console.log("a=",a);
</script>
```

a. a= Nan
b. a= 2*3
c. a= 6

Odgovor: c

Pitanje 5

Koja JavaScript funkcija vraća nasumičan broj između 0 i 1 (isključujući 1)?

a. Math.round()
b. Math.random()
c. Math.floor()

Odgovor: b

Pitanje 6

Koji tipovi podataka spadaju u proste tipove u JavaScript-u?

a. string, number, boolean
b. int, float, char
c. text, numeric, logical

Odgovor: a