Web dizajn i programiranje

Prof. dr.sc. Dragutin Kermek
Sveučilište u Zagrebu
Fakultet organizacije i informatike
Pavlinska 2, Varaždin 42000
dkermek@foi.hr

09. dio

Programiranje na strani korisnika

Uvod u Javascript jezik i koncept programiranja.

Objektni model dokumenta (Document Object Model - DOM).

Programiranje događaja.

Razvoj dinamičkih osobina dokumenata.

Dozvoljeni izrazi (Regular Expressions - RegExp).

Web 2.0., Ajax

jQuery

Sadržaj stranica

Prva verzija HTML jezika temeljila se na jednostavnim hipertekst/hipermedija osobinama. Stranice su bile kreirane na poslužitelju i njihov sadržaj jedino je bio mijenjan radnjama čovjeka – webmastera. Uskoro se uočavaju potrebe za promjenama postojećeg načina i dešava se bitan napredak tako da se omogućava dinamičko sastavljanje sadržaja stranica po zahtjevu korisnika.

Sadržaj stranica može se sastavljati:

- statički – unaprijed pripremljeni i jednaki sadržaj za sve korisnike
- dinamički – sadržaj se sastavlja u ovisnosti o:
  - Lokalno pohranjenom identifikatoru korisniku (cookie) koji zahtijeva stranicu – personalizacija stranica
  - Upisanim podacima formulara koji poziva zahtjevanu stranica
  - Podacima iz baze podataka

Statički sadržaj stranica

To je tzv. poznati dio Web-a koji ulazi u baze podataka pretraživača.
Danas je glavni problem gubitak veza prema stranicima jer veliki broj stranica nestaje, a stranice koje sadrže veze prema tim stranicima nisu ažurirane pa upućuju na nepostojeće stranice.

Prema rezultatima istraživanja radi se o 10% u razdoblju od 12 mjeseci dok je u ranijem razdoblju bilo 7%.

Utvrđeno je da 25% Web stranica ima barem jednu izgubljenu vezu.

Dinamički sadržaj stranica

To je tzv. nepoznati dio Web-a koji ne može ući u potpunosti u baze podataka pretraživača jer postoji neograničeni broj različitih mogućnosti sadržaja stranica.

Najveći dio sadržaj stranice uglavnom se sastavlja na strani poslužitelja, neovisno o kojoj se vrsti sastavljanja sadržaja radi.

Poslužiteljska strana mora imati programske mogućnosti kojima se sadržaj generira prema zahtjevima korisnika

Korisnička strana također je dobila programske mogućnosti kojima se sadržaj koji je dobiven od poslužitelja prilagođava zahtjevima korisnika.
Efikasnost rada podiže se smanjenjem nepotrebnih interakcija korisnik-poslužitelj kada se radi o unosu i kontroli podataka. Sve kontrole koje se mogu realizirati na strani korisnika treba ih provesti na tom mjestu, a poslužitelj se kontaktira nakon upisivanja ispravnih podataka.

Aktivnost stranica

Osnovni elementi prve verzije HTML jezika uglavnom su bili usmjereni na praćenje sadržaja stranica putem ugrađenih veza. Dalji razvoj HTML jezika trebao je omogućiti dodatne osobine kojima bi stranice postale aktivne.

Stranica prema tome može biti:

- pasivna/statička – ne prati rad korisnika
- aktivna/dinamička – prati rad korisnika i prema njemu se može mijenjati izgled i sadržaj stranice.

Pasivne/statičke stranice

Pasivne stranice i dalje ostaju za potrebe jednostavnih sadržaja kao npr. informacije koje ne očekuju interakciju i sl.

Aktivne/dinamičke stranice

Aktivnost stranica postiže se ugradnjom programskim mogućnosti na korisničkoj strana, kojima se sadržaj dobiven od poslužitelja prilagođava zahtjevima korisnika i njegovom radu.

Brojnost preglednika i njihova raznolikost u izvršavanju standarda HTML jezika uvjetovala je ugradnju programskih sposobnosti u preglednike kojima će se moći prilagođavati osobine pojedinih preglednika prema očekivanom načinu izvršavanja.

Sve veća potreba za prikupljanjem podataka putem Weba imala je utjecaj na omogućavanje kontrole rada korisnika i podataka koje on onosi.

Efikasnost rada podiže se smanjenjem nepotrebnih interakcija korisnik-poslužitelj kada se radi o unosu i kontroli podataka. Sve kontrole koje se mogu realizirati na strani korisnika treba ih provesti na tom mjestu.

Jezici skriptiranja

HTML jezik omogućuje izbor između više jezika skriptiranja (skriptnih jezika) pri čemu se najčešće radi o sljedećim:

- JavaScript (JScript)
- VBScript
- a rijeđe:
  - Tcl
  - Perl

Preglednik treba imati ugrađenu potporu zahtjevanom jeziku skriptiranja.

U nastavku objašnjavaju se principi skriptiranja na primjeru JavaScript (JScript) jezika.

Osobine skriptnih jezika

Skriptni jezici obično imaju sljedeće osobine:

- interpreterski su pa je potreban interpreter koji će interpretirati izvorni kod i izvršavati ga
- imaju slabo povezivanje tipova (nemaju deklaraciju tipa podatka) zbog čega varijable nemaju stalni tip podatka pa mogu mijenjati tip podatka prema potrebi tijekom izvršavanja. Interpreter ne može otkriti pogreške kod pridruživanja vrijednosti pojedinoj varijabli. Misli se na drugačiji tip podatka od trenutnog tipa kojem pripada varijabla.
- koriste se za pisanje manjih programskih cjelina, koje se nazivaju skripte
- postoje razne vrste skriptnih jezika s obzirom na područje primjene (nemaju svi gore spomenute osobine):
  - web preglednik - JavaScript
  - web poslužitelj - PHP, Perl, Python, ASP, JSP
  - ljuska operacijskog sustava - UNIX shell
  - multimedija - ActionScript
  - obrada teksta - AWK, sed.

Definiranje skriptnog dijela i jezika

Skripta se definira oznakom `<script> ... </script>`

U jednom dokumentu može biti više odvojenih skripata i svaka od njih može biti napisana u drugom jeziku skriptiranja.

Postoji mogućnost da preglednik ne podržava skriptiranje ili je onemogućeno izvršavanje skripata pa se u tom slučaju može koristiti oznaku `<noscript> ... </noscript>` u kojem se objašnjava da nije moguće postići predviđenu funkcionalnost.

Za određivanje jezika skriptiranja koristi se atribut `"type", a` može se upisati:

- `"text/javascript"`
- `"text/vbscript"`
- `"text/tcl"`
- i dr.

Primjer skripte

```
<!DOCTYPE html>
<html>
    <head>
        <title>FOI - Web dizajn i programiranje : JavaScript - Primjer broj
                  01</title>
        <meta charset="utf-8">
    </head>
    <body>

        <script type="text/javascript">
            document.write("<P>JavaScript program");
        </script>

        <noscript>
        Preglednik NE može izvršti JavaScript!
        </noscript>

    </body>
</html>
```

Uvod/Primjer01 – Primjer skripte

Smještaj skripata

Skripte mogu biti sastavni dio sadržaja dokumenta kao kod prethodnog primjera, ali mogu biti pohranjene i kao zasebne datoteke i vezom uključene u dokument.

```
<!DOCTYPE html>
<html>
    <head>
        <title>FOI – Web dizajn i programiranje : JavaScript – Primjer broj
                02</title>
        <meta charset="utf-8">
    </head>
    <body>

        <script type="text/javascript" src="Primjer02.js">
        </script>

        <noscript>
        Preglednik NE može izvršti JavaScript!
        </noscript>

    </body>
</html>
```


Smještaj skripata / 1.

Sadržaj datoteke `Primjer02.js`:

```
document.write("<P>JavaScript program");
```

Uvod/Primjer02 – Smještaj skripata

JavaScript jezik

JavaScript jezik je objektno orijentirani (preciznije prototipski orijentirani) programski jezik skriptiranja koji se može izvršava unutar različitih preglednika zbog čega nije ovisan o platformi pa se može izvršavati na različitim platformama.

Razvila ga je tvrtka Netscape za preglednik Navigator 2.0 po uzoru na programski jezik Java tvrtke Sun Microsystems.

Razvijen pod nazivom Mocha, od autora Brendan Eich-a, a predstavljen je u rujnu 1995. godine kao LiveScript. Ubrzo je promijenjen naziv u JavaScript.

U studenom 1996. godine predan je zahtjev organizaciji ECMA (European Computer Manufacturers Association) da JavaScript postane industrijski standard. Kasnije (1997. godime) standardzirana verzija dobila je naziv ECMAScript (ECMA-262).

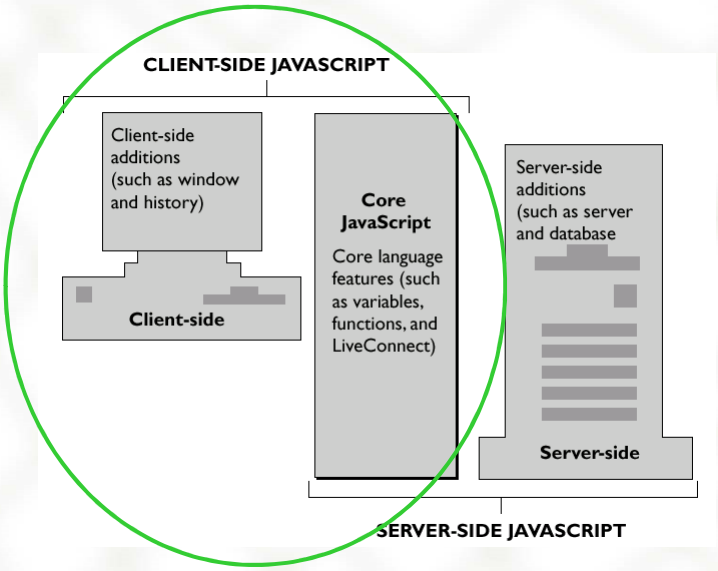
Kroz razvoj JavaScript je prošao mnoge verzije, od 1.0 do 1.8.x kao zadnje.

JavaScript jezik

Na početku su predvidene 3 razine JavaScript jezika:

- **središnji dio**
- **za programiranje na strani klijenta**
- **za programiranje na strani poslužitelja**

Korištenje JavaScript jezika



JavaScript jezik

Poslužiteljska strana nije najbolje primljena od strane struke te je dulji niz godina bila zapostavljena.

Unatrag zadnjih nekoliko godina ponovno se ulažu značajni napori kako bi se razvila uspješna verzija poslužitelja koji bi izvršavao JavaScript programe.

Trenutno je najpopularnija implementacija node.js - temelji se na V8 JavaScript Engine (stroj) koji je razvio Google. Postoji još nekoliko implementacija a one se najčešće temelje na JavaScript strojevima:

- **SpiderMonkey kodno ime prvog JavaScript stroja, otvorenog koda, razvijen u C/C++, održava ga Mozilla Foundation.**
- **Rhino JavaScript stroj otvorenog koda, razvijen u Java-i, održava ga Mozilla Foundation.**

Središnji dio JavaScript jezika

Središnji dio JavaScript jezika čine:

- **vrijednosti**
- **varijable**
- **literali – doslovne vrijednosti**
- **izrazi**
- **operacije**
- **dozvoljeni izrazi**
- **instrukcije**
- **funkcije**
- **klase/objekti**

Vrijednosti

Vrijednosti mogu biti:

`brojčane` - `34` ili `8383.3434`
`logičke` - `true` ili `false`
`stringovi` - `"Programiranje za Web"`
`null` - **posebna ključna riječ kojom se određuje nul vrijednost**
`undefined` - **svojstvo najviše razine čija vrijednost nije definirana**

JavaScript jezik razlikuje velika i mala slova!

Varijable

Naziv odnosno identifikator varijable može se sastojati od :

- **prvi znak može biti slovo ili _**
- **ostali mogu biti sastavljeni od slova, brojaka i znaka _**

Varijable se mogu deklarirati na 2 načina:

- **korištenjem ključne riječi `var`**
  - **primjer** `var dan = 29;`

  koristi se unutar funkcija za definiranje lokalne varijable
- **pridruživanjem vrijednosti**
  - **primjer** `dan = 29;`

Područje primjene varijable može biti:

- **globalno – vrijedi za sve funkcije kada je varijabla deklarirana izvan funkcija**
- **lokalno – vrijedi unutar funkcije/bloka u kojoj je deklarirana.**

Prevođenje tipa podataka

JavaScript jezik je jezik dinamičkog tipa zbog čega nije potrebno specificirati tip varijable kada se deklarira. Tipovi podataka prevode se automatski prema potrebi za vrijeme izvršavanja.

```
var mjesec = 4;
mjesec = "travanj ";
```

Pojavljivanje dviju brojčanih vrijednosti kod operatora + za JavaScript jezik znači zbrajanje brojčanih vrijednosti.

```
var dan = 20;
dan = dan + 1;
```

daje `21`

Pojavljivanje stringa i brojčane vrijednosti kod operatora + za JavaScript jezik znači prevođenje brojčane vrijednosti u string.

```
datum = mjesec + 12;
```

daje `"travanj 12"`

Literali – doslovne vrijednosti

Postoje sljedeće vrste literala:

- **polje/niz** - `prog_jezici = ["JavaScript", "VBScript", "C++"]`
- **logički** - `true` i `false`
- **klizni zarez** - `25.4343 -5.11E11`
- **cijeli broj** - `44 -343 0xFF7`
- **objekt** - `PC = {mp: "Pentium III 600", ram: "128 MB", disk:"10 GB"}`
- **string** - `"Danas je ... ".`

Izrazi

Izraz je svaki pravilan skup literala, varijabli, operatora i izraza koji daje pojedinačnu vrijednosti. Vrste izraza:

- **aritmetički**
- **string**
- **logički**

Operatori

Vrste operatora:

- **pridruživanja** (kao C/C++)
- **uspoređivanja** (kao C/C++) uz
  - **identičnost** `===` **ista vrijednosti i tip**
  - **neidentičnost** `!==` **nije ista vrijednosti i/ili isti tip**
- **aritmetički** (kao C/C++)
- **razina bitova** (kao C/C++)
- **logički** (kao C/C++)
- **string** `+` **dodaje drugi string na kraj prvog**
- **posebni**
  - **uvjetni** `? :` (kao C/C++)
  - **zarez** (kao C/C++)
  - **delete** (kao C/C++) **i brisanje elementa u polju**
  - **new** (kao C/C++)
  - **this** (kao C/C++)
  - **instanceof** (kao C/C++)
  - **typeof** (kao C/C++)
  - **void** `void (izraz)` **izraz se izvršava bez vraćanja vrijednosti**

Instrukcije

Vrste instrukcija:

- **uvjet** (kao C/C++)
- **petlja** (kao C/C++)
- **manipulacija objektom**
- **komentar** (kao C/C++)

Instrukcije uvjeta

```
if (uvjet) {
        instrukcije1
}
[else {
        instrukcije2
}]


switch (izraz) {
        case labela1:
                instrukcije1
                break;
        case labela2:
                instrukcije2
                break;
        ...
        default:
                instrukcije
}
```


Instrukcije petlje

```
for ([početni izraz]; [uvjet]; [izraz povećavanja]) {
        instrukcije1
}

do {
        instrukcije
} while (uvjet);


while (uvjet) {
        instrukcije
}

label:

break [label];

continue [label];
```


Instrukcije petlje / 1.

```
<body>

    <script type="text/javascript">
        for (i = 0; i < 10; i++) {
            document.write("i=" + i + "<br>");
        }
        while (i > 0) {
            document.write("i=" + i + "<br>");
            i--;
        }
    </script>

    <noscript>
    Preglednik NE može izvršti JavaScript!
    </noscript>

</body>
```

document.write - funkcija kojom se dinamički piše/dodaje u sadržaj važećeg dokumenta

Uvod/Primjer03 – Petlje

Instrukcije manipulacije s objektom

```
for (varijabla in objekt) {
        instrukcije
}


with (objekt) {
        instrukcije
}
```


Instrukcije obrade pogrešaka

Određeni dijelovi programskog koda mogu biti izloženi različitim situacijama uslijed čega može doći do pogrešaka u njihovom radu. Takve dijelove programskog koda (instrukcije) potrebno je ohuhvatiti instrukcijom try-catch koja će u slučaju pogreške preusmjeriti izvršavanje na obradu pogreške/iznimke. Postoji mogućnost da se u određenoj situaciji okine/generira prikladna pogreška.

```
try {
        instrukcije
} catch (errorInfo) {
        obrada pogreške
}

throw // okidanje uvjeta pogreške
```


Instrukcije obrade pogrešaka / 1.

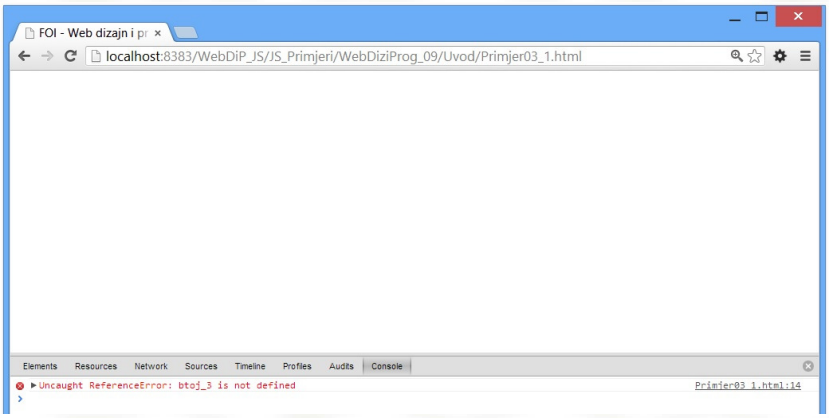
```
    <body>

        <script type="text/javascript">
            var broj_1 = 2;
            var broj_2 = 0;
            var broj_3 = broj_1 / broj_2;

            alert("Rezultat: " + broj_1 + "/" + broj_2 + "=" + btoj_3);
        </script>

...

    </body>
```

alert - funkcija za ispis poruke u dijaloškom okviru

Korištenje nedefinirane varijable

Uvod/Primjer03_1 – Obrada pogreške

Instrukcije obrade pogrešaka / 2.



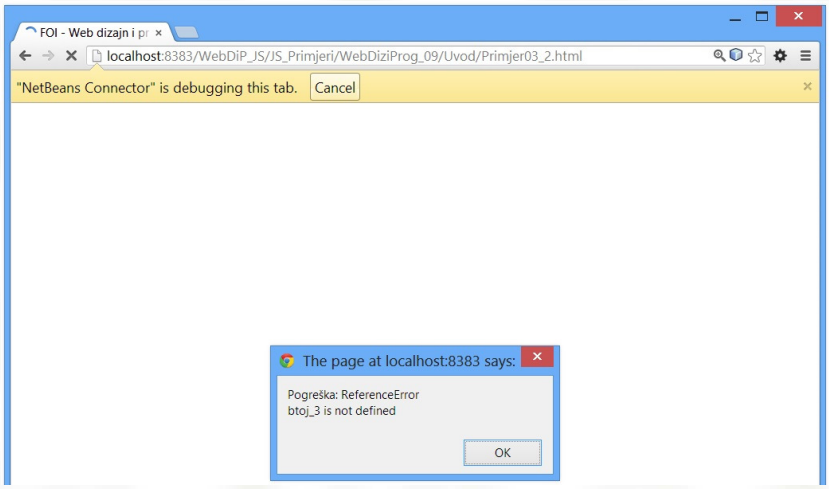
Prikaz opisa pogreške u JavaScript konzoli

Instrukcije obrade pogrešaka / 3.

```
<script type="text/javascript">
          var broj_1 = 2;
          var broj_2 = 1.3;
          var broj_3 = broj_1 / broj_2;

          try {
              alert("Rezultat: " + broj_1 + "/" + broj_2 + "=" +
                    btoj_3);
          } catch (e) {
              alert("Pogreška: " + e.name + " \n" + e.message);
          }
      </script>
```

Uvod/Primjer03_2 – Smještaj skripata

Instrukcije obrade pogrešaka / 4.



Prikaz opisa pogreške u dijaloškom okviru

Instrukcije obrade pogrešaka / 5.

```
<script type="text/javascript">
    try {
        var broj_1 = 2;
        var broj_2 = 0;
        if (broj_2 == 0) {
            var pogreska = new Error("Djeljenje s nulom!");
            throw pogreska;
        }
        var broj_3 = broj_1 / broj_2;

        alert("Rezultat: " + broj_1 + "/" + broj_2 + "=" +
              broj_3);
    } catch (e) {
        alert("Pogreška: " + e.name + " \n" + e.message);
    }
</script>
```

Klasa za pogrešku

Okidanje iznimke

Uvod/Primjer03_3 – Obrada pogreške

Funkcije

Definiranje vlastite funkcije:

```
function naziv ([argumenti]) {
        instrukcije
}
```

Predefinirane funkcije:

- `eval` - izvrši string kao dio koda
- `isFinite` - ispituje je li konačan broj
- `isNaN` - ispituje je li ne broj
- `parseInt` - vraća int od stringa
- `parseFloat` - vraća float od stringa
- `escape` - pretvara tekst u oblik za URL
- `unescape` - vraća tekst iz URL oblika u narmalan oblik

Predefinirani objekti u JavaScript jeziku

To su objekti koji postoje u središnjem dijelu JavaScript jezika. Svaki objekt ima svoj skup atributa i metoda.

To su:

- `Object`
- `Array`
- `Boolean`
- `Date`
- `Function`
- `Math`
- `Number`
- `String`
- `RegExp` - posebno se obrađuje

Objekti u JavaScript jeziku

Objekti su složeni tipovi podataka koji imaju attribute i metode koje djeluju nad atributima.

Kreiranje objekta:

```
function osoba(ime, prezime, visina, spol) {
        this.ime = ime;
        this.prezime = prezime;
        this.visina = visina;
        this.spol = spol;
 }
 var pero = new osoba("Pero", "Kos", 180, 1};
```

```
var osoba = {ime: "Pero", prezime: "Kos", visina: 180,
spol: 1};
```

Do atributa u objektu možemo pristupiti:

- `osoba["ime"]`
- `osoba.ime`

Object

Atributi:

Funkcije:

- `toSource` - **vraća objekt koji predstavlja specificirani objekt.**
- `toString` - **vraća string koji predstavlja specificirani objekt.**
- `valueOf` - **vraća primitivnu vrijednosti specificiranog objekta.**

Array - samo indeksirani nizovi

Atributi:

- `length` - broj elemenata u polju

Funkcije:

- `concat` - spaja 2 polja i vraća novo polje.
- `join` - spaja sve elemente polja u string.
- `pop` - briše posljednji element u polju i vraća taj element.
- `push` - dodaje jedan ili više elemenata na kraj polja i vraća novu veličinu polja.
- `reverse` - zamjenjuje elemente polja, prvi postaje posljednji, a posljednji prvi.
- `shift` - briše prvi element iz polja i vraća taj element.
- `slice` - izdvaja dio polja i vraća novo polje.
- `splice` - dodaje i/ili briše elemente iz polja.
- `sort` - sortira elemente polja.
- `toString` - vraća string koji predstavlja polje i njegove elemente.
- `valueOf` - vraća primitivnu vrijednost polja.

Boolean

Atributi:

Funkcije:

- `toString` - vraća string koji predstavlja specificirani objekt.
- `valueOf` - vraća primitivnu vrijednost Boolean objekta.

Date

Atributi:

Funkcije:

- `getDate` - vraća dan u mjesecu.
- `getDay` - vraća dan u tjednu.
- `getFullYear` - vraća godinu.
- `getHours` - vraća sat.
- `getMilliseconds` - vraća milisekunde.
- `getMinutes` - vraća minute.
- `getMonth` - vraća mjesec.
- `getSeconds` - vraća sekunde.
- `getTime` - vraća brojčanu vrijednosti koja odgovara vremenu.
- `getYear` - vraća godinu.
- `toSource` - vraća objekt koji predstavlja Date objekt.
- `toString` - vraća string koji predstavlja Date objekt.
- `valueOf` - vraća primitivnu vrijednosti Date objekta.

Date / 1.

Atributi:

Funkcije:

- `setDate` - postavlja dan u mjesecu.
- `setDay` - postavlja dan u tjednu.
- `setFullYear` - postavlja godinu.
- `setHours` - postavlja sat.
- `setMilliseconds` - postavlja milisekunde.
- `setMinutes` - postavlja minute.
- `setMonth` - postavlja mjesec.
- `setSeconds` - postavlja sekunde.
- `setTime` - postavlja brojčanu vrijednosti koja odgovara vremenu.
- `setYear` - postavlja godinu.

Math

Atributi:

- `E` - Eulerova konstanta i baza prirodnog, aproks. 2.718.
- `LN10` - prirodni logaritam od 10, aproks. 2.302.
- `LN2` - prirodni logaritam od 2, aproks. 0.693.
- `LOG10E` - baza 10 logaritma od E (aproks. 0.434).
- `LOG2E` - baza 2 logaritma od E (aproks. 1.442).
- `PI` - aproks. 3.14159.

Funkcije:

- `abs` - vraća apsolutnu vrijednost broja.
- `acos` - vraća arc cos vrijednosti broja.
- `asin` - vraća arc sin vrijednosti broja.
- `atan` - vraća arc tan vrijednosti broja.
- `ceil` - vraća najmanji cijeli broj veći ili jednak broju.
- `cos` - vraća cos vrijednosti broja.

Math / 1.

Funkcije:

- `floor` - vraća najveći broj manji ili jednak broju.
- `log` - vraća prirodni logaritam broja.
- `max` - vraća veći broj od dva broja.
- `min` - vraća manji broj od dva broja.
- `random` - vraća pseudo slučajni broj u intervalu 0 i 1.
- `round` - vraća vrijednost broja zaokruženu na najbliži cijeli broj.
- `sin` - vraća sin vrijednost broja.
- `sqrt` - vraća drugi korijen vrijednost broja.
- `tan` - vraća tan vrijednost broja.

Number

Atributi:

- `MAX_VALUE` - najveći broj koji se može predstaviti.
- `MIN_VALUE` - najmanji broj koji se može predstaviti.
- `NaN` - posebna "nije broj" vrijednost.
- `NEGATIVE_INFINITY` - posebna vrijednost koja predstavlja negativnu beskonačnost, vraća se kod prekoraćenja.
- `POSITIVE_INFINITY` - posebna vrijednost koja predstavlja beskonačnost, vraća se kod prekoraćenja

Funkcije:

- `toString` - vraća string koji predstavlja specificirani objekt.
- `valueOf` - vraća primitivnu vrijednosti specificiranog objekta.

String

Atributi:

- `length` - duljina stringa.

Funkcije:

- `anchor` - kreira HTML vezu koja se koristi kao hipertekstni cilj
- `charAt` - vraća znak na određenom indeksu.
- `concat` - združuje dva stringa i vraća novi string.
- `indexOf` - vraća indeks u stringu za prvo pojavljivanje tražene vrijednosti, ili -1 ako nije pronađeno.
- `lastIndexOf` - vraća indeks u stringu za posljednje pojavljivanje tražene vrijednosti, ili -1 ako nije pronađeno.
- `slice` - izdvaja dio stringa i vraća novi string.
- `split` - cijepa string u polje stringova prema izabranom znaku

String / 1.

Funkcije:

- `substr` - vraća znakove u stringu koji počinju od određene lokacije do određenog broja znakova.
- `substring` - vraća znakove u stringu između dva indeksa u stringu.
- `toLowerCase` - vraća string pretvoren u mala slova.
- `toSource` - vraća objekt koji predstavlja specificirani objekt.
- `toString` - vraća string koji predstavlja specificirani objekt.
- `toUpperCase` - vraća string pretvoren u velika slova.
- `valueOf` - vraća primitivnu vrijednosti specificiranog objekta.

Primjer skripte

```
<script type="text/javascript">
          var imena = ["Pero", "Mato", "Ivo", "Biba", "Lena", "Mia"];
          var gradovi = new Array("Varaždin", "Čakovec", "Zagreb", "Split",
               "Rijeka", "Osijek");
          ispisi(imena);
          imena.sort();
          ispisi(imena);
          gradovi.reverse();
          ispisi(gradovi);
          var povezani = gradovi.join(";");
          document.write("<br>" + povezani + "<br>");
          var mjesano = imena.concat(gradovi);
          mjesano.sort();
          ispisi(mjesano);

      function ispisi(niz) {
            document.write("<br>");
            for (n in niz) {
                document.write(n + ". => " + niz[n] + "<br>");
            }
       }
```

indeksirani nizovi string podataka

Primjer skripte

indeksirani niz objekata - JSON

```
var osobe = [{ime: "Pero", prezime: "Kos", visina: 180, spol: 1},
             {ime: "Mato", prezime: "Zec", visina: 183, spol: 1},
             {ime: "Ivo", prezime: "Vrana", visina: 173, spol: 1},
             {ime: "Biba", prezime: "Zec", visina: 165, spol: 0},
             {ime: "Lena", prezime: "Vrana", visina: 178, spol: 0},
             {ime: "Mia", prezime: "Kos", visina: 168, spol: 0}
];
```

1. način dohvata atributa u objektu

```
ispisiOsobe(osobe);
osobe.sort(function(a,b) {return a["visina"] - b["visina"];});
 ispisiOsobe(osobe);

function ispisiOsobe(niz) {
    document.write("<br><pre>");
    for (n in niz) {
        document.write(n + ". => " + niz[n].prezime + " \t" +
              niz[n].ime + " \t" + niz[n].visina + "\n");
    }
    document.write("</pre>");
}
```

2. način dohvata atributa u objektu

Primjer skripte

```
var sada = new Date();
document.write(sada.toString() + "<br>");
document.write(sada.getTime() + "<br>");
nulti = new Date(sada.setTime(0));
document.write(sada.toString() + "<br>");
var broj = 5.5;
zaokruzeno = Math.round(broj);
document.write(broj + " zaokruzeno = " + zaokruzeno + "<br>");
var min_vr = Number.MIN_VALUE;
var max_vr = Number.MAX_VALUE;
var poz_neo = Number.POSITIVE_INFINITY;
document.write(min_vr + " " + max_vr + " " + poz_neo + "<br>");
var manje = Number.MAX_VALUE < Number.POSITIVE_INFINITY;
var jednako = Number.MAX_VALUE == Number.POSITIVE_INFINITY;
document.write(manje + " " + jednako + "<br>");
```


Primjer skripte

```
var kolegij = "Web dizajn i programiranje";
document.write("'" + kolegij + "' broj znakova=" + kolegij.length +
      "<br>");
document.write("'" + kolegij + "' znak na 7 mjestu=" +
     kolegij.charAt(7) + "<br>");
document.write("'" + kolegij + "' 'Web' na mjestu=" +
     kolegij.indexOf("Web") + "<br>");
document.write("'" + kolegij + "' 'j' na mjestu=" +
     kolegij.indexOf("j") + "<br>");
document.write("'" + kolegij + "' 'j' na posljednjem mjestu=" +
     kolegij.lastIndexOf("j") + "<br>");
```


Primjer skripte

```
var kraj = false;
var tekst = kolegij;
var trazi = "a";
var i = 1;
var abspoz = 0;
do {
    var poz = tekst.indexOf(trazi);
    if (poz != -1) {
        abspoz = abspoz + poz;
        document.write(i + ". pojavljivanje '" + trazi + "' u '" +
              kolegij + "' na mjestu " + abspoz + "<br>");
        tekst = tekst.substr(poz + 1);
        i++;
    } else {
        kraj = true;
    }
} while (!kraj);
```


Primjer skripte

```
</script>
    </body>
</html>
```

Uvod/Primjer04_3 – Primjer skripte

Programiranje na strani korisnika

Jedan od glavnih problema odnosi se na postojanje više verzija preglednika koji podržavaju različite verzije JavaScript jezika.

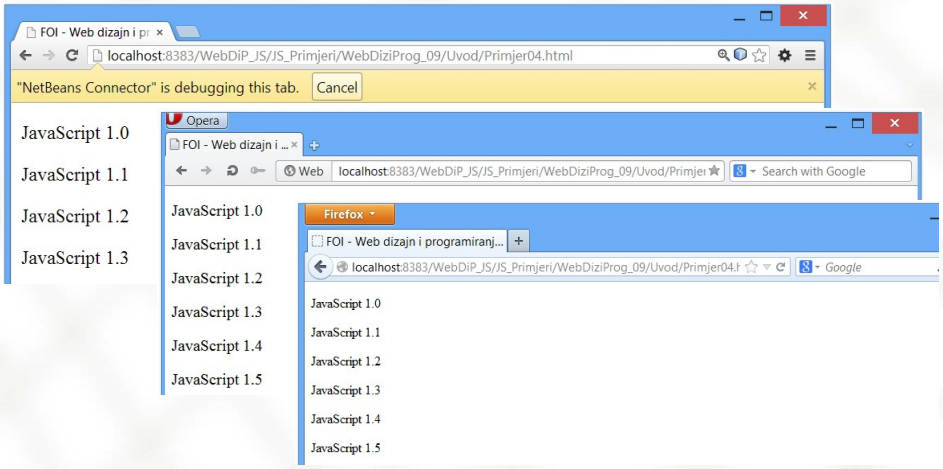
To je jedan od glavnih razloga za primjenu proširene vrijednosti atributa type kojom se određuje minimalna verzija JavaScript jezika koja će se koristiti za izvršavanje koda ukoliko ju web preglednik podržava.

```
<script type="text/javascript1.2">
...
</script>
```


Određivanje verzije

```
<script type="text/javascript">
    document.write("<P>JavaScript 1.0");
</script>

<script type="text/javascript1.1">
    document.write("<P>JavaScript 1.1");
</script>

<script type="text/javascript1.2">
    document.write("<P>JavaScript 1.2");
</script>
...
<script type="text/javascript1.8">
    document.write("<P>JavaScript 1.8");
</script>

<script type="text/javascript1.9">
    document.write("<P>JavaScript 1.9");
</script>

<script type="text/javascript2.0">
    document.write("<P>JavaScript 2.0");
</script>
```


Određivanje verzije / 1.



Uvod/Primjer04 – Određivanje verzije

Korištena i dodatna literatura

- http://www.w3.org/standards/webdesign/script
- http://www.w3schools.com/js/default.asp
- http://www.angelfire.com/tx4/cus/notes/javascript.html
- http://dir.yahoo.com/Computers_and_Internet/Programming_and_Development/Languages/JavaScript/
- http://webreference.com/programming/javascript/
- http://javascript.internet.com/
- http://www.jsworkshop.com/js15examples.html