

Oživljavanje prezentacije s JavaScript-om

Programiranje s JavaScript-om

Staviti disketu u disketni uređaj. Aktivirati program *Notepad*. U programu otkucati sledeći sadržaj:

```
<HTML>
<BODY>
<SCRIPT LANGUAGE="JavaScript">
document.write("<h2>Tabela faktorijela</h2>");
    for(i=1, fact=1; i<=10; i++, fact *= i) {
        document.write(i +"!=" + fact);
        document.write("<br>");
    }
</SCRIPT>
</BODY>
</HTML>
```

Otkucani tekst zapamtiti u datoteci pod imenom **java-cas1.htm** u katalogu **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:** (Opcija *Save as* menija *File*). Spustiti aplikaciju (a ne zatvoriti!) pritiskom na dugme **_** u gornjem desnom uglu pruge naslova. Aktivirati program *Netscape*. Otvoriti pripremljenu prezentaciju (Opcija *Open* menija *File*). Na ekranu treba da se pojavi rezultat rada JavaScript programa, a to je lista vrednosti faktorijela brojeva od 1 do 10. Program koristi **for** repetativni iskaz jezika JavaScript. Osim toga, korisiti se i metod **write()** objekta **document** koji dodaje podatke na dokument. Podaci koji se dodaju su argument funkcije **write()**.

Naslov liste faktorijela ne mora da bude deo JavaScript programa. U datoteci **java-cas1.htm** izbaciti prvi **document.write** iskaz, a ispred etikete **<SCRIPT ...** dodati **html** tekst: **<H2>Tabela faktorijela</H2>**. Zapamtiti sadržaj datoteke, aktivirati program *Netscape* i u njemu osvežiti sadržaj stranice (dugme *Reload*). Izgled stranice se nije promenio jer se JavaScript program izvršava na onom mestu u **html** tekstu na kome se pojavljuje.

Napomena: Ako se JavaScript program sadrži neku grešku, on se neće izvršiti, što znači da se prelistač neće prikazati očekivani sadržaj. Statusna linija prelistača nas obično obavještava da se neka greška pojavila u toku izvršavanja JavaScript programa. U tom slučaju treba u polju *Location* prelistača okucati `javascript:` i tada se otvara novi prozor koji sadrži poruke o greškama u toku do kojih je došlo u toku izvršavanja JavaScript programa. Te poruke obično sadrže broj reda u datoteci koja sadrži JavaScript programa koji nije mogao da se izvrši zbog greške i kratak opis greške.

Aktivirati program *Notepad* i u njemu izabrati opciju *New*. Otkucati sledeći sadržaj:

```
<HTML>
<BODY>
<H2>Najveci zajednicki delilac brojeva 438 i 213 je:</H2>
<SCRIPT LANGUAGE="JavaScript">
  m = 438;
  n = 213;
  r = m % n; // Racuna se ostatak pri deljenju
  while ( r != 0 ) { // Dok je ostatak razlicit od 0
    m = n;
    n = r;
    r = m % n;
  }
  document.write("Broj " + n);
  document.write("<br>");
</SCRIPT>
</BODY>
</HTML>
```

Otkucani tekst zapamtiti u datoteci pod imenom **java-cas2.htm** u katalogu **gr1-r1** na disketnoj jedinici **A:**. Spustiti aplikaciju (a ne zatvoriti!) i aktivirati program *Netscape*. Otvoriti pripremljenu prezentaciju (Opcija *Open* menija *File*). Na ekranu treba da se pojavi rezultat rada JavaScript programa, a to je najveći zajednički delilac brojeva 438 i 213. Program koristi `while` repetitivni iskaz jezika JavaScript. Iza dve kose crte `//` nalazi se *komentar* koji opisuje šta program radi, a koji se ne izvršava.

Računanje najvećeg zajedničkog delioca može da se izdvoji u zasebnu funkciju `nzd`. Prirodno mesto za zapis funkcije je unutar `<head>` i `</head>` etiketa `html` teksta. Rezultat rada funkcije dobija se na metu njenog poziva. Preurediti (ne prekucavati!) sadržaj datoteke **java-cas2.htm** tako da sada izgleda na sledeći način:

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript">
// Funkcija za racunanje najveceg zajednickog delioca
function nzd(m,n) {
  var r; // promenljiva koja se koristi samo u toku rada funkcije
  r = m % n; // Racuna se ostatak pri deljenju
  while ( r != 0 ) { // Dok je ostatak razlicit od 0
    m = n;
    n = r;
    r = m % n;
  }
}
```

```

        return n; // Rezultat rada funkcije koji koristi progra
                // (ili skript) koji ga poziva
    }
</SCRIPT>
</HEAD>
<BODY>
<H2>Najveci zajednicki delilac brojeva 438 i 213 je:</H2>
<SCRIPT LANGUAGE="JavaScript">
// Telo funkcije se izvorsava samo prilikom poziva funkcije
    document.write("Broj " + nzd(438,213) );
    document.write("<br>");
</SCRIPT>
</BODY>
</HTML>

```

Napomena: Programski jezik JavaScript je osetljiv na veličinu slova, što znači da ime `nzd` nije isto što i ime `NZD`. Isto važi i za imena ugrađenih funkcija, metoda, objekata i svojstava. Tako `document.write()` nije isto što i `document.Write()`.

Aktivirati program *Netscape* i u njemu osvežiti sadržaj stranice (dugme *Reload*). Izgled stranice se nije promenio jer se iskazi funkcije ne izvršavaju sve dok se funkcija ne pozove. Ova funkcija, `nzd`, ima dva argumenta: `m` i `n`. Kakve su stvarne vrednosti ovih promenljivih zavisi od toga sa kojim se vrednostima funkcija poziva. Promeniti sadržaj datoteke **java-cas2.htm** tako da se iz nje izbaci tekst između etiketa `<h2>` i `</h2>`, prvi `document.write` iskaz se menja a iza njega dodaje još jedan `document.write` iskaz, na sledeći način:

```

document.write("NZD brojeva 438 i 213 je " + nzd(438,213) + "<br>");
document.write("NZD brojeva 1025 i 1353 je " + nzd(1025,1353) );

```

Aktivirati program *Netscape* i u njemu osvežiti sadržaj stranice (dugme *Reload*). Na ekranu se sada pojavljuje rezultat dva poziva funkcije `nzd`: `nzd(438,213)` i `nzd(1025,1353)`. JavaScript iskazi se interpretiraju onim redom kojim su zapisani u `html` tekstu. S toga je neophodno da definicija funkcije prethodi njenom pozivu. Aktivirati program *Notepad* i promeniti sadržaj datoteke **java-cas2.htm** tako da se prvi blok teksta između etiketa `<SCRIPT>` i `</SCRIPT>` premesti iza drugog takvog bloka. Sada poziv funkcije prethodi njenoj definiciji. Zapamtiti promenjeni sadržaj datoteke, aktivirati program *Netscape* i u njemu osvežiti sadržaj stranice (dugme *Reload*). Na ekranu se pojavljuje prazna stranica, kao rezultat greške u zapisu JavaScript koda. Aktivirati program *Notepad* i vratiti prethodan sadržaj datoteke **java-cas2.htm**. Koristeći JavaScript moguće je programirati izračunavanje najvećeg zajedničkog delioca za dva proizvoljna broja koja korisnik sam unosi u ulaznom polju. Promeniti sadržaj između etiketa `<BODY>` i `</BODY>` tako da sada sadrži:

```

<H2>Racunanje najveceg zajednickog delioca:</H2>
<SCRIPT LANGUAGE="JavaScript">
    m=prompt("Unesi prvi broj:", "");
    n=prompt("Unesi drugi broj:", "");

```

```

        document.write("NZD brojeva ", m, " i ", n, " je ", nzd(m,n),
"<br>");
</SCRIPT>

```

Aktivirati program *Netscape* i u njemu osvežiti sadržaj stranice (dugme *Reload*). Na stranici se pojavljuje dijaloški prozor u kome se korisnik poziva da unese prvi broj u prazno ulazno polje. Uneti podatak treba potvrditi pritiskom na taster OK. Odmah se pojavljuje i drugi dijaloški prozor u kome se korisnik poziva da unese drugi broj u prazno ulazno polje. Odmah po unošenju broja i pritisku na taster OK na ekranu se ispisuje rezultat. Program koristi `prompt()` metod jezika JavaScript koja se koristi za unošenje niski preko dijaloškog prozora. Ova funkcija ima dva argumenta: prvi je poruka koja se ispisuje u dijaloškom prozoru, a drugi predefinisana vrednost ulaznog polja. U našem primeru predefinsana vrednost je prazna niska ("") i zato je na početku ulazno polje prazno.

Ako želite da izračunate najveći zajednički delilac neka druga dva broja treba osvežiti sadržaj stranice (dugme *Reload*) i na stranici će se opet pojaviti prvi dijaloški prozor. Aktivirati program *Netscape* i isprobati. Ovo se može rešiti i na drugi način. Aktivirati program *Notepad* i sadržaj datoteke **java-cas2.htm** i promeniti tako da tekst između etiketa `<SCRIPT>` i `</SCRIPT>` bude:

```

<SCRIPT LANGUAGE="JavaScript">
    dane = "D";
    while ( (dane=="D") || (dane=="d") ) {
        m=prompt("Unesi prvi broj:", "");
        n=prompt("Unesi drugi broj:", "");
        document.write("NZD brojeva ", m, " i ", n, " je ",
            nzd(m,n), "<br>");
        dane=prompt("Zelis li da nastavis (D/N)?:", "");
    }
</SCRIPT>

```

Aktivirati program *Netscape* i u njemu osvežiti sadržaj stranice (dugme *Reload*). Sada se posle unosa brojeva u prva dva dijaloška prozora i ispisa rezultata pojavljuje i treći dijaloški prozor u kome unosimo malo ili veliko slovo D ako želimo i dalje da računamo. Ako tako odgovorimo na postavljeno pitanje, ponovo će se pojaviti prvi dijaloški prozor. Ako u treći dijaloški prozor unesemo malo ili veliko slovo N (ili, uostalom, bilo koje drugo slovo osim slova D) program će prestati s radom i stranica će postati statična. Sličan efekat proizvešće i pritisak na taster **Cancel**. U slučaju pritiska tog tastera, funkcija `prompt()` vraća vrednost `null` (ništa, ili bez vrednosti), a ta 'vrednost' se razlikuje i od malog i od velikog slova D. Ovaj deo JavaScript programa koristi `while` repetitivni iskaz čiji uslov ponavljanja zavisi od vrednosti logičkog izraza koji korsiti logički 'ili' operator `||`. U ovom delu programa važno je uočiti da promenljiva `dane` mora dobiti početnu vrednost `D`, da bi izvršavanje repetitivnog iskaza uopšte otpočelo.

Objekti u JavaScript-u

Staviti disketu u disketni uređaj. Aktivirati program *Notepad*. U programu otkucati sledeći sadržaj:

```

<HTML>
<HEAD>
<TITLE>Danasnji datum</TITLE>
  <SCRIPT LANGUAGE="JavaScript">
    // Definisemo funkciju koju koristimo kasnije
    function stampaj_danasnji_datum()
    {
      var d = new Date(); // danasnji datum i vreme
      document.write(d.toLocaleString());
    }
  </SCRIPT>
</HEAD>
<BODY>
<HR>Datum i vreme su:<BR><B>
  <SCRIPT LANGUAGE="JavaScript">
    // Sada se poziva funkcija koja je gore definisana
    stampaj_danasnji_datum();
  </SCRIPT>
</B><HR>
</BODY>
</HTML>

```

Otkucani tekst zapamtiti u datoteci pod imenom **java-cas3.htm** u katalogu **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:** (Opcija *Save as* menija *File*). Spustiti aplikaciju (a ne zatvoriti!) pritiskom na dugme _ u gornjem desnom uglu pruge naslova. Aktivirati program *Netscape*. Otvoriti pripremljenu prezentaciju (Opcija *Open* menija *File*). Na ekranu treba da se pojavi rezultat rada JavaScript programa, a to je tekući datum i tekuće vreme. Ovaj program koristi objekat *Date* koji je ugrađen u JavaScript jezik. Da bi se kreirao ovaj objekat, koristi se operator *new* s konstruktorom (funkcijom) *Date()*. Konstruktorom *Date()* može se koristiti i kao funkcija koja tada vraća nisku koja predstavlja tekući datum i vreme za lokalnu vremensku zonu. Aktivirati program *Notepad* i promeniti definiciju funkcije *stampaj_danasnji_datum()* na sledeći način:

```

function stampaj_danasnji_datum()
{
  document.write( Date() );
}

```

Aktivirati program *Netscape* i u njemu osvežiti sadržaj stranice (dugme *Reload*). Na stranici se ponovo pojavljuje tekući datum i vreme ali u nešto izmenjenog izgleda. Datum i vreme se mogu prikazati i na niz drugih načina, ali za to je potrebno da se koristi objekat *Date* uz koji se može primeniti niz ugrađenih metoda. Aktivirati program *Notepad* i promeniti definiciju funkcije *stampaj_danasnji_datum()* na sledeći način:

```

function stampaj_danasnji_datum()
{
  var d = new Date(); // danasnji datum i vreme
  dan_u_m= d.getDate();
  m = d.getMonth() + 1;
  g = d.getYear() + 1900;
  document.write(dan_u_m + "." + m + "." + g); }

```

Zapamtiti sadržaj datoteke, aktivirati program *Netscape* i u njemu osvežiti sadržaj stranice (dugme *Reload*). Na stranici se ponovo pojavljuje tekući datum ali u obliku koji je kod nas uobičajeniji. Funkcija `stampaj_danasnji_datum()` koristi ugrađene metode `getDate()`, `getMonth()` i `getYear()` objekta tipa `Date` (u ovom slučaju to je objekat `d`). Izdvajanje dana, meseca i godine iz objekta `d` omogućava nam prikazivanje datuma u obliku koji nam više odgovara. Izgled prikazanog datuma može se i dalje poboljšavati. Aktivirati program *Notepad* i dopuniti definiciju funkcije `stampaj_danasnji_datum()` na sledeći način:

```
function stampaj_danasnji_datum()
{
    var d = new Date(); // danasnji datum i vreme
    var a = new Array("Ned", "Pon", "Uto", "Sre", "Cet", "Pet", "Sub");
    i = d.getDay();
    dan_u_n = a[i];
    dan_u_m= d.getDate();
    m = d.getMonth() + 1;
    g = d.getYear() + 1900;
    document.write(dan_u_n + ", " + dan_u_m + "." + m + " " + g);
}
```

Zapamtiti sadržaj datoteke, aktivirati program *Netscape* i u njemu osvežiti sadržaj stranice (dugme *Reload*). Na stranici se ponovo pojavljuje tekući datum, a uz njega i naziv dana u nedelji na srpskom jeziku. Funkcija pored već pomenutih koristi i metodu `getDay()` objekta tipa `Date` koja vraća redni broj dana u nedelji. Osim ovog objekta, funkcija koristi i objekat `a` tipa `Array` (niz) koji se kreira pomoću operatora `new` i konstruktora `Array()`. Vrednosti navedene kao argumenti funkcije `Array()` su vrednosti prvih 7 elemenata niza. Elementu niza se pristupa preko indeksa što omogućava da se redni broj dana u nedelji pretvori njegov naziv.

Staviti disketu u disketni uređaj. Aktivirati program *Notepad*. Otkucati sledeći sadržaj:

```
<HTML>
<HEAD><TITLE>Koliko jos do Nove godine</TITLE></HEAD>
<BODY>
<HR>
    <SCRIPT LANGUAGE="JavaScript">
        danas = new Date(); // danasnje vreme i datum
        nova_god = new Date(); // nova godina
        nova_god.setMonth(11); // je u decembru
        nova_god.setDate(31); // 31. dan
        // Koliko ima milisekundi izmedju Nove godine i danasnjeg dana
        razlika = nova_god.getTime() - danas.getTime();
        // A koliko je to dana
        razlika = Math.floor( razlika/ (1000*60*60*24) );
        document.write("<h2 align=center>Do Nove godine samo jos " +
                        razlika + " dana!</h2>");
    </SCRIPT>
<HR>
<HR>
</BODY>
</HTML>
```

Otkucani tekst zapamtiti u datoteci pod imenom **java-cas4.htm** ukatalogu **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:** (Opcija *Save as* menija *File*). Spustiti aplikaciju (a ne zatvoriti!) pritiskom na dugme **_** u gornjem desnom uglu pruge naslova. Aktivirati program *Netscape*. Otvoriti pripremljenu prezentaciju (Opcija *Open* menija *File*). Na ekranu treba da se pojavi koliko je još dana ostalo da Nove godine. JavaScript program koristi dva objekta tipa **Date**: **dan**as i **nova_god**. Za postavljanje datuma Nove godine koriste se ugrađeni metodi ovog objekta, **setMonth()** i **setDate()**. Metod **getTime()** vraća vrednost objekta tipa **Date** izraženu u milisekundama. Ona se može koristiti za izračunavanje intervala između dva datuma ili vremena koji su u istoj vremenskoj zoni. Ovaj JavaScript program koristi i specijalni objekat **Math** koji se koristi kao mesto na kome se čuvaju različite matematičke konstante i funkcije. Funkcija **floor()** vrši zaokruživanje realnog broja na prvi manji ceo broj.

Osobine i metode objekata

Objekat je podatkovni tip koji sadrži imenovane podatke. Svaki takav imenovani podatak je svojstvo objekta. Svako svojstvo ima ime, a objekat pridružuje vrednost svakom takvom svojstvu. Vrednost svojstva može biti bilo kog tipa.

Staviti disketu u disketni uređaj. Aktivirati program *Notepad*. U programu otkucati sledeći sadržaj:

```
<HTML>
<HEAD>
<TITLE>Svojstva objekta Windows</TITLE>
</HEAD>
<BODY>
<HR>
<H2>Svojstva tekuceg prozora su:</H2>
  <SCRIPT LANGUAGE="JavaScript">
    document.write("<b>Prozor je otvorio: </b>" + window.opener
      + "<br>");
    if(window.closed)
      document.write("Prozor je zatvoren<br>");
    document.write("<b>Tekuca vrednost statusne linije: </b>"
      + window.status + "<br>");
    document.write("<b>Predefinisana vrednost statusne linije: </b>"
      + window.defaultStatus + "<br>");
    document.write("<b>Ime prozor je: </b>" + window.name +
      "<br>");
  </SCRIPT>
<HR>
</BODY>
</HTML>
```

Otkucani tekst zapamtiti u datoteci pod imenom **java-cas5.htm** u katalogu **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:** (Opcija *Save as* menija *File*). Spustiti aplikaciju (a ne zatvoriti!) pritiskom na dugme **_** u gornjem desnom uglu pruge naslova. Aktivirati program *Netscape*. Otvoriti pripremljenu prezentaciju (Opcija *Open* menija *File*). Na ekranu se pojavljuju vrednosti svojstava tekućeg prozora (objekat **window**, čije

je alternativno ime `self`). Prikazane vrednosti, na prvi pogled, nemaju mnogo smisla. Pretvorimo, pre svega, ovaj JavaScript program u funkciju na sledeći način:

```
<HTML>
<HEAD>
<TITLE>Svojstva objekta Windows</TITLE>
<SCRIPT LANGUAGE="JavaScript">
// Funkcija za prikazivanje svojstava objekata
function svojstva_prozora(w)
{
    document.write("<b>Prozor je otvorio: </b>" + w.opener
                  + "<br>");
    if(w.closed)
        document.write("Prozor je zatvoren<br>");
    document.write("<b>Tekuca vrednost statusne linije: </b>"
                  + w.status + "<br>");
    document.write("<b>Predefinisana vrednost statusne linije: "
                  + "</b>" + w.defaultStatus + "<br>");
    document.write("<b>Ime prozor je: </b>" + w.name +
                  "<br>");
    return true;
}
</SCRIPT>
</HEAD>
<BODY>
<HR>
<H2>Svojstva tekuceg prozora su:</H2>
<SCRIPT LANGUAGE="JavaScript">
    svojstva_prozora(window);
</SCRIPT>
<HR>
</BODY>
</HTML>
```

Zapamtiti promenjeni sadržaj u datoteci **java-cas5a.htm** u katalogu **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:**, aktivirati program *Netscape* i u njemu osvežiti sadržaj stranice (dugme *Reload*). Na stranici se ponovo pojavljuje isti sadržaj, vrednosti nekih svojstava objekta `window`. Aktivirati program *Notepad* i promeniti sadržaj datoteke **java-cas5a.htm** dodavanjem sledećih redova iza poziva funkcije `svojstva_prozora(window)`:

```
win = window.open("", "mali",
                  "status, resizable,width=500,height=300");
```

Zapamtiti sadržaj datoteke, aktivirati program *Netscape* i osvežiti sadržaj stranice. Na ekranu se pojavljuje još jedan *Netscape* prozor, kao rezultat poziva metode `open` objekta `window`. (**Napomena:** Ako se novi ekran ne vidi na ekranu, možda je spušten u liniju poslavlja i terba ga aktivirati.) Ova funkcija ima tri argumenta: prvi je URL dokumenta kojim će se ispuniti prozor (ako je prazna niska, onda je prozor prazan), drugi je ime prozora, a treći je niz karakteristika prozora (ako je prazna niska dobija se uobičajeni *Netscape* prozor). U ovom slučaju otvoreni prozor ima statusnu liniju, veličina mu se može menjati, širok je 500 piksela, a visok 300 piksela. Niska koja sadrži niz

karakteristika prozora, ne sme da sadrži blanko karaktere, tabulatore ili karaktere koji označavaju kraj reda.

Aktivirati program *Notepad* i u datoteci **java-cas5a.htm** promeniti prvi argument u metodu `open` objekta `window`, tako da sadrži URL dokumenta koji treba da ispuni novi prozor. Neka taj prvi argument bude:

```
"file:///A|/gr1-r1/java-cas5.htm"
```

Ovaj argument je URL već formirane datoteke, one koja je sadržala JavaScript program u okviru tela html-dokumenta (za prvu grupu, prvi računar). Zapamtiti sadržaj datoteke, aktivirati program *Netscape* i osvežiti sadržaj stranice. Na ekranu se sada pojavljuje novi prozor koji sada nije prazan, već su u njemu ispisane karakteristike tekućeg prozora, što je u ovom slučaju taj novi prozor. Karakteristike ovog prozora nešto drugačije izgledaju od osnovnog prozora: njega je otvorio neki drugi objekat tipa `Window` (prva karakteristika), a ime mu je `mali` (poslednja karakteristika). Karakteristike novog prozora možemo dobiti i u originalnom prozoru, novim pozivanjem funkcije `svojstva_prozora` s drugim argumentom. Aktivirati program *Notepad* i u datoteci **java-cas5a.htm**, na kraj JavaScript programa (ispred druge `</SCRIPT>` etikete) uneti sledeće programske iskaze:

```
document.write("<h3>Svojstva novog prozora:</h3>");
svojstva_prozora(win);
```

Argument funkcije `svojstva_prozora` je sada promenljiva `win` koja referiše novootvoreni prozor. Zapamtiti sadržaj datoteke, aktivirati program *Netscape* i osvežiti sadržaj stranice. U originalnom prozoru pojavljuju se svojstva novog prozora koja su ista sa svojstvima novog prozora koja su u njemu i prikazana.

Aktivirati ponovo program *Notepad* i u datoteci **java-cas5a.htm** promeniti, pre svega funkciju `svojstva_prozora` tako što će se uvesti novi argument funkcije `d` koji označava dokument dok će se u telu funkcije poziv metode `write` sada primeniti na dokument `d`. Ovo znači da u telu funkcije treba zameniti svako pojavljivanje `document.write` sa `d.write` (primeniti opciju *Replace* menija *Search*). Sadržaj JavaScript programa između drugog para etiketa `<SCRIPT>` i `</SCRIPT>` treba promeniti tako da glasi:

```
<SCRIPT LANGUAGE="JavaScript">
    var d_stari = window.document;
    svojstva_prozora(d_stari, window);
    win = window.open("",
        "mali",
        "status,resizable,width=500,height=300");
    document.write("<h3>Svojstva novog prozora:</h3>");
    svojstva_prozora(d_stari, win);
    var d_novi = win.document; // dokument u novom prozoru
    d_novi.write("<h3>Svojstva prozora koji me je " +
        "otvorio:</h3>");
    svojstva_prozora(d_novi, window);
    d_novi.write("<h3>Svojstva tekućeg prozora:</h3>");
    svojstva_prozora(d_novi, win);
</SCRIPT>
```

Ovako promenjen sadržaj datoteke **java-cas5a.htm** zapamtiti u datoteci **java-cas5b.htm**. U programu su sada uvedene dve promenljive `d_stari` i `d_novi` koje referišu dokument u starom, originalnom prozoru, odnosno, dokument u novom prozoru. Dva poziva funkcije `svojstva_prozora` menjaju se tako da sada uključuju i novi argument, `d_stari` što znači da se svojstva originalnog i novog prozora ispisuju u dokumentu originalnog prozora. U program su dodata i dva nova poziva funkcije `svojstva_prozora` kod kojih je prvi argument `d_novi` a drugi `window` odnosno `win`, što znači svojstva originalnog i novog prozora treba ispisati u dokumentu novog prozora.

Aktivirati program *Netscape*. Otvoriti prezentaciju pripremljenu u datoteci **java-cas5b.htm**. (Opcija *Open* menija *File*). Na ekranu se pojavljuju podaci o starom i novom prozoru u starom prozoru i isti ti podaci u novom prozoru.

Rukovanje događajima u JavaScript-u

Rukovanje događajima obezbeđuju funkcije specijalne namene koje se koriste samo u klijentskom JavaScript-u. Njihov način korišćenja se razlikuje od uobičajenih funkcija: njih ne poziva JavaScript program već sam prelistač kad kod se dogodi neki "događaj" unutar html- elementa sa kojim je pridružen.

Staviti disketu u disketni uređaj. Aktivirati program *Notepad*. U programu otkucati sledeći sadržaj:

```
<HTML>
<HEAD>
<TITLE>Rukovanje dogadjajima - primer</TITLE>
<BODY>
<HR>
<FORM>
<INPUT TYPE='submit' VALUE='Pritisni ovde'
      onClick='alert("Vi ste to izabrali");'>
</FORM>
<HR>
</BODY>
</HTML>
```

Otkucani tekst zapamtiti u datoteci pod imenom **java-cas6.htm** u katalogu **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:** (Opcija *Save as* menija *File*). Spustiti aplikaciju (a ne zatvoriti!) pritiskom na dugme `_` u gornjem desnom uglu pruge naslova. Aktivirati program *Netscape*. Otvoriti pripremljenu prezentaciju (Opcija *Open* menija *File*). Na ekranu treba da se pojavi jedno dugme na kome piše: **Pritisni ovde**. Ako kliknete mišem na to dugme, pojavljuje se poruka **Vi ste to izabrali** u dijaloškom prozoru. Dugme koristi `onClick` rukovanje događajima u okviru etikete `<INPUT>` html-formulara. U ovom slučaju događaj je klik (`Click`) levim tasterom miša. Funkcija koja se u slučaju tog događaja izvršava ne mora da bude zadata na uobičajen način, korišćenjem ključne reči `function`, već se može zadati i navođenjem jednog, ili više, JavaScript iskaza koji se izvršavaju ako nastupi odgovarajući događaj. Iskazi se u tom slučaju završavaju obavezno sa `;`.

Aktivirati program *Notepad* i ponovo otvoriti datoteku **java-cas5b.htm**. Iza JavaScript programa (iza druge `</SCRIPT>` etikete) dodati sledeće programske iskaze:

```
<HR>
<b>Dugme za zatvaranje prozora:</b>
<FORM>
<INPUT TYPE='button' VALUE='Zatvori'
      onClick='win.close();'>
</FORM>
<HR>
```

Ovako promenjen sadržaj datoteke **java-cas5b.htm** zapamtiti u datoteci **java-cas5c.htm**. U HTML dokument je dodat obrazac sa samo jednim dugmetom, čija je vrednost **Zatvori**. I ovo dugme koristi `onClick` rukovanje događajima, događaj je klik levim tasterom miša, a u slučaju tog događaja poziva se metod `win.close()` objekta prozor (u ovom slučaju, objekat prozor je novootvoreni prozor `win`). Aktivirati program *Netscape*. Otvoriti prezentaciju pripremljenu u datoteci **java-cas5c.htm**. (Opcija *Open menija File*). U originalnom prozoru se sada pojavljuje i dugme **Zatvori**. Klikom mišem na ovo dugme, zatvara se novi prozor.

Osim događaja kakav je klik mišem postoje i mnogi drugi događaji, kakav je, na primer, prelazak mišem preko hipertekstualne veze (`MouseOver`) i pomeranje miša sa hipertekstualne veze (`MouseOut`). Staviti disketu u disketni uređaj. Aktivirati program *Notepad*. U programu otkucati sledeći sadržaj:

```
<HTML>
<HEAD>
<TITLE>Rukovanje događajima - prelazak mišem preko veze</TITLE>
<SCRIPT LANGUAGE="JavaScript">
// Funkcija otvara prozor objasnjenja
function otvori_prozor_obj()
{
    win = open("tekstovi/okrem.htm",
              "obj",
              "width=50,height=50");
    return true;
}
// Funkcija za zatvaranje prozora
function zatvori_prozor_obj()
{
    win.close();
    return true;
}
</SCRIPT>
</HEAD>
<BODY>
<HR>
<H3>Ko je ovo?
<a href="Slike/kamengrad.jpg" onMouseOver="otvori_prozor_obj();"
onMouseOut="zatvori_prozor_obj();">
</a></H3>
</BODY>
</HTML>
```

Otkucani tekst zapamtiti u datoteci pod imenom **java-cas7.htm** u katalogu **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:** (Opcija *Save as* menija *File*). Otvoriti novi dokument (Opcija *New* menija *File*) i u njemu okucati sledeći sadržaj:

```
<HTML>
<BODY>
<small>
Ako ovde kliknete, doćicete sliku svih glavnih junaka iz Kamengrada!
</small>
</BODY>
</HTML>
```

Otkucani tekst zapamtiti u datoteci pod imenom **okrem.htm** u podkatalogu **tekstovi** kataloga **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:** (Opcija *Save as* menija *File*). Spustiti *Notepad* (a ne zatvoriti!) pritiskom na dugme _ u gornjem desnom uglu pruge naslova. Aktivirati program *Netscape*. Otvoriti pripremljenu prezentaciju (Opcija *Open* menija *File*). Na ekranu se pojavljuje mala slika Freda Kremenka. Ona je u isto vreme veza polaska koja vodi ka drugom dokumentu, većoj slici svih kamengradskih junaka. Ako se mišem prede preko ove veze otvoriće se novi, mali *Netscape* prozor kao rezultat rukovanja događajem (*MouseOver*), a kada se miš skloni s hipertekstuelne veze ovaj mali prozor se zatvara kao rezultat rukovanja događajem (*MouseOut*). Događajem *MouseOver* rukuje atribut *onMouseOver* etikete sidro (*<a>*) čija vrednost je, u ovom slučaju, poziv funkcije *otvori_prozor_obj()*. Ova funkcija je definisana u zaglavlju html-dokumenta i ona otvara prozor u koji se smešta dokument čiji je sadržaj na URL-adresi *tekstovi/okrem.html*. Na sličan način se rukuje i događajem *MouseOut*.

Umesto da se otvara mali prozor u kome je sadržanao kratko objašnjenje kuda vodi neka hipertekstuelna veza, to objašnjenje se može postaviti i u statusnu liniju prelistača. Svaki objekat tipa *window* ima sledeća dva svojstva: *status* i *defaultStatus*. Prvi sadrži sadržaj statusne linije prelistača, a drugi daje osnovni sadržaj statusne linije koji se, eventualno, menja pod uticajem nekih događaja. Aktivirati program *Notepad* i u njemu opet otvoriti dokument **java-cas7.htm** iz kataloga **gr1-r1**. Sadržaj dokumenta između etiketa *<BODY>* i *</BODY>* promeniti na sledeći način:

```
<BODY>
<HR>
<H3>Ko je ovo?
<a href="Slike/kamengrad.jpg"
onMouseOver="status='Slika svih kamengradskih junaka'; return true;"
onMouseOut="status='';">
</a></H3>
</BODY>
```

U odnosu na prethodni sadržaj, izmenjene se vrednosti atributa *onMouseOver* i *onMouseOut* etikete sidro (*<a>*): događaj prelaska mišem preko veze postavlja vrednost niske 'Slika svih kamengradskih junaka' u statusnu liniju, a izmicanje miša sa veze postavlja praznu nisku u statusnu liniju. Definicije funkcija date

između etiketa `<HEAD>` i `</HEAD>` treba izbaciti, a između ovih etiketa pripremiti samo sledeći sadržaj:

```
<SCRIPT>
  defaultStatus="Fred Kremenko i drustvo";
</SCRIPT>
```

Ovaj kratki JavaScript program postavlja definiše nisku "Fred Kremenko i drustvo" kao osnovni sadržaj statusne linije. Umesto `defaultStatus` u programu može da stoji i eksplicitno `window.defaultStatus`, što znači da je `defaultStatus` svojstvo prozora `window`, ali to je i tekući i jedini otvoreni prozor pa se on ne mora navoditi. Izmenjeni sadržaj zapamtiti u datoteci **java-cas7a.htm** u katalogu **gr1-r1**. Spustiti *Notepad* (a ne zatvoriti!) pritiskom na dugme _ u gornjem desnom uglu pruge naslova. Aktivirati program *Netscape*. Otvoriti prezentaciju iz datoteke **java-cas7a.htm** (Opcija *Open* menija *File*). Slika na ekranu je ista kao i malopre, samo što statusna linija nije prazna i što se prilikom prelaska mišem preko hipertekstuelne veze sadržaj statusne linije menja.

Okviri kao objekti i rukovanje njima

Pošto su okviri vrsta prozora, imaju ista svojstva i koriste iste metode kao i prozori. Staviti disketu u disketni uređaj. Aktivirati program *Notepad*. U programu otkucati sledeći sadržaj:

```
<HTML>
<HEAD>
<TITLE>Okviri i rukovanje njima</TITLE>
</HEAD>
<FRAMESET COLS="20%,80%">
  <FRAME SRC="tekstovi/sadrzaj.htm" NAME="sadrzaj">
  <FRAME SRC="slike/kandid.jpg" NAME="tekst">
</FRAMESET>
</HTML>
```

Otkucani tekst zapamtiti u datoteci pod imenom **java-cas8.htm** u katalogu **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:** (Opcija *Save as* menija *File*). Otvoriti novi dokument (Opcija *New* menija *File*) i u njemu okucati sledeći sadržaj:

```
<HTML>
<BODY bgcolor="lavenderblush">
<H2>Sadrzaj</H2>
<a onClick="parent.tekst.location.replace('java-cas1.htm');"
  href="sadrzaj.html" >Prvi primer</a><br>
<a onClick="parent.tekst.location.replace('java-cas2.htm');"
  href="sadrzaj.html" >Drugi primer</a><br>
<a onClick="parent.tekst.location.replace('java-cas3.htm');"
  href="sadrzaj.html" >Treci primer</a><br>
</BODY>
</HTML>
```

Otkucani tekst zapamtiti u datoteci pod imenom **sadrzaj2.htm** u podkatalogu **tekstovi** kataloga **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:** (Opcija *Save as* menija *File*). Spustiti *Notepad* (a ne zatvoriti!) pritiskom na dugme **_** u gornjem desnom uglu pruge naslova. Aktivirati program *Netscape*. Otvoriti prezentaciju pripremljenu u datoteci **java-cas8.htm** (Opcija *Open* menija *File*). Na ekranu se pojavljuju dva okvira: u prvom je sadržaj prezentacije, a u drugom jedna slika — ilustracija. Ova dva okvira su definisana u okviru etiketa `<FRAMESET>` i `</FRAMESET>`. Prvi okvir popunjen je tekстом iz datoteke **sadrzaj2.htm** iz podkataloga **tekstovi**, dok je drugi okvir popunjen slikom iz datoteke **kandid.jpg** iz podkataloga **slike**. U prvom okviru je neka vrsta sadržaja onoga što se može pojaviti u drugom, većem, okviru. Sadržaj se, praktično, sastoji od liste veza polaska (između etiketa `<A>` i `</A>`), od kojih svaka sadrži dva atributa: **HREF** koji upućuje na tu istu datoteku (klikom se sadržaj levog okvira ne menja) i **onClick** koji JavaScriptom definiše akciju koja se preduzima u slučaju kada je događaj klik mišem na hipertekstuelnu vezu. U ovom slučaju akcija se definiše samo jednim JavaScript iskazom koji poziva metod `replace()` objekta `locate` koji zamenjuje jedan prikazani dokument drugim. Ovaj metod ima jedan argument, a to je URL tog novog dokumenta koji treba prikazati umesto starog. U našem primeru, objekat `locate` je svojstvo okvira `tekst` koji je tako nazvan prilikom njegovog definisanja u datoteci **java-cas8.htm**, kao vrednost atributa **NAME** u okviru etikete `<FRAME>`. Objekat `locate` predstavlja i kontroliše lokaciju prelistača u tekućem prozoru, a to je u ovom slučaju okvir. Kako je okvir `tekst` definisan u okviru glavnog prozora, u datoteci **java-cas8.htm**, njegovo ime nije poznato u okviru drugog, njemu susednog, okvira `sadrzaj` (ti okviri su, na neki način, braća). Zato se mora navesti da je ime okvira `tekst` poznato u okviru nadređenog prozora koji se uvek referiše preko svojstva `parent`.

Umesto imenom, okvir se može referisti i preko niza `frames` koji je svojstvo svakog prozora. Aktivirati program *Notepad* i u njemu opet otvoriti dokument **sadrzaj2.htm** iz kataloga **tekstovi**. Zameniti u okviru svake vrednosti `onClick` atributa, referencu drugog okvira `tekst` referencom `frames[1]`. Recimo, vrednost `onClick` atributa u okviru prve etikete `<a>` postaje:

```
onClick="parent.frames[1].location.replace('java-cas1.htm');"
```

Kako se menja sadržaj drugog okvira on se referiše sa `frames[1]`, dok bi se prvi okvir referisao sa `frames[0]`. Zapamtiti sadržaj datoteke **sadrzaj2.htm**, spustiti *Notepad*, aktivirati *Netscape* i osvežiti stranicu (dugme *Reload+Shift*, tako se postiže da se drugi okvir ponovo napuni originalnim sadržajem, slikom). Isprobati hipertekstuelne veze iz levog okvira. One funkcionišu na isti način kao i pre izmene. Ako isprobate hipertekstuelne veze više puta i pokušate da se vratite na prethodno stanje (prethodni primer) dugmetom *Back*, videćete da ono ne funkcioniše. Razlog je korišćenje metoda `replace()` objekta `location` koji zamenjuje tekući dokument u istoriji prelistača, umesto da u nju doda novi element. Zamena prikazanog dokumenta u prozoru i njegovo uvršćivanje u istoriju prelistača, postiže se direktnom dodelom URL-a novog dokumenta svojstvu `location` prozora.

Aktivirati program *Notepad*, u njemu otvoriti datoteku **sadrzaj.htm** i podkataloga **tekstovi** i zameniti vrednost `onClick` atributa u okviru svih etiketa `<a>` na sledeći način (primer je za prvu etiketu):

```
onClick="parent.frames[1].location='primer1.htm';"
```

Zapamtiti sadržaj datoteke **java-cas8.htm**, spustiti *Notepad*, aktivirati *Netscape* i osvežiti stranicu (dugme *Reload+Shift*). Isprobati hipertekstuelne veze iz levog okvira. Sada dugme *Back* opet funkcioniše na uobičajeni način i moguće je šetanje kroz istoriju prelistača sve do prvog dokumenta u desnom prozoru, a to je slika.

Dokument kao objekat

Dokument je objekat koji predstavlja sadržaj prozora ili okvira. Ovaj objekat ima svojstva koja specifikuju informacije o dokumentu: njegov URL, njegove boje, datum kada je poslednji put modifikovan, URL dokumenta koji ga je pozvao, i slično. Ovaj objekat ima i više metoda koji podržavaju dinamičko pisanje u dokument (metod `document.write()` koji je više puta korišćen), otvaranje novog dokumenta. Objekat `document` sadrži i više nizova svojstava koji specifikuju informacije o sadržaju dokumenta, kao što su veze, sidra, HTML formulari, apleti i slično.

Staviti disketu u disketni uređaj. Aktivirati program *Notepad*. Otvoriti datoteku **java-cas7a.htm** iz kataloga **gr1-r1** (opcija *Open* iz menija *File*). Na kraj datoteke (ispred završne etikete `</BODY>` otkucati sledeći sadržaj:

```
<HR><BR><BR><BR>
<FONT SIZE=1>
Dokument: <I><SCRIPT>document.write(document.title);</SCRIPT></I><BR>
URL: <I><SCRIPT>document.write(document.URL);</SCRIPT></I><BR>
Poslednja izmena:
<I><SCRIPT>document.write(document.lastModified);</SCRIPT></I><BR>
</FONT>
```

Zapamtiti izmenu u datoteci pod imenom **java-cas7b.htm** (opcija *Save As* menija *File*). Spustiti program *Notepad* i aktivirati program *Netscape*. Otvoriti pripremljenu prezentaciju (opcija *Open Page* menija *File*). Na kraju dokumenta izlistani su podaci o dokumentu kao što su njegov naslov koji se inače javlja u pruzi naslova programa *Netscape*, URL dokumenta (u ovom slučaju to će biti datoteka na lokalnom računaru) i datum poslednje izmene. Za ovaj prikaz koristi se metod `document.write()` koji je i u dosadašnjim primerima često korišćen i svojstva `title`, `URL` i `lastModified` objekta `document` koja sadrže tražne vrednosti.

Svojstva boje objekta dokument

Staviti disketu u disketni uređaj. Aktivirati program *Notepad*. U programu otkucati sledeći sadržaj:

```
<HTML>
<HEAD>
```

```

<TITLE>Izbor boje pozadine</TITLE>
<SCRIPT LANGUAGE="JavaScript">
    function DecUHexa(DecBr) {
        var hexa = "0123465789abcdef";
        x = Math.floor(DecBr / 16); h = hexa.charAt(x);
        x = DecBr % 16;             h += hexa.charAt(x);
        return h;    }
function HexaUDec(HexaBr) {
    var hexa = "0123465789abcdef";
    h = HexaBr.charAt(0); i = hexa.indexOf(h);  x = i * 16;
    h = HexaBr.charAt(1); i = hexa.indexOf(h);  x += i
    return x;
}
function menjaj_boju() {
    cr = document.bgColor.substring(1,3);
    ze = document.bgColor.substring(3,5);
    pl = document.bgColor.substring(5);
    cr = HexaUDec(cr)+8; ze = HexaUDec(ze)+8; pl = HexaUDec(pl)+8;
    if ((pl >= 255) && (ze >=255)) {
        ze = 0; pl = 0;
    }
    else if (pl >= 255) {
        pl = 0; cr = cr - 8;
    }
    else {
        cr -= 8; ze -= 8;
    }
    cr = DecUHexa(cr); ze = DecUHexa(ze); pl = DecUHexa(pl);
    document.bgColor= "#" + cr + ze + pl;
    window.setTimeout("menjaj_boju()",400)
}
defaultStatus="Posmatraj kako se menja boja pozadine";
</SCRIPT>
</HEAD>
<BODY BGCOLOR="#000000" onLoad="menjaj_boju();">
<BR><BR>
<HR>
<H2>Koja boja pozadine je najboljsa?</H2>
<HR>
</BODY>
</HTML>

```

Otkucani tekst zapamtiti u datoteci pod imenom **java-cas9.htm** u katalogu **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:** (Opcija *Save as* menija *File*). Spustiti *Notepad* (a ne zatvoriti!) pritiskom na dugme _ u gornjem desnom uglu pruge naslova. Aktivirati program *Netscape*. Otvoriti pripremljenu prezentaciju (Opcija *Open* menija *File*). Pojavljuje se crni ekran koji postepeno menja boju. Gornji program ilustruje više mogućnosti JavaScript jezika. Pre svega, program koristi rukovanje događajima **onLoad** koje se može koristiti kao atribut etikete **<BODY>** ili **<FRAME>**. Njegova vrednost je JavaScript program koji se izvršava prilikom punjenja prozora dokumentom. U ovom slučaju program se sastoji od samo jednog iskaza kojim se poziva funkcija **menjaj_boju()** koja je definisana u između etiketa **<HEAD>** i **</HEAD>**.

Funkcija `menjaj_boju()` menja vrednost boje pozadine dokumenta koja je svojstvo `bgColor` objekta `document`. Početna vrednost ovog svojstva zadaje se u okviru etikete `<BODY>` kao vrednost atributa `BGCOLOR`. Vrednost se zadaje kao niska oblika `#RRGGBB` u kojoj se iza numeričkog znaka navodi šest heksadekadnih cifara, od kojih prve dve specifikuju crveni spektar, druge dve zeleni a poslednje dve plavi. U okviru funkcije `menjaj_boju()` boja pozadine se shvata kao jedan trocifreni broj u brojnom sistemu sa osnovom 256. Vrednost svojstva `document.bgColor` uzima redom svaku osmu vrednosti u ovom brojnom sistemu (ako bi se uzimala svaka vrednost, prikazivanje različitih boja išlo bi isuviše sporo).

Funkcija `menjaj_boju()`, pre svega ilustruje korišćenje metode `substr()` objekta tipa niska (`String`). Ovde se metod primenjuje na nisku `document.bgColor`. Ovaj metod iz niske vadi podnisku. Metod ima dva argumenta: prvi je početna pozicija podniske u nisci, a drugi prva pozicija **iza** završne pozicije podniske u nisci. Odbrojavanje pozicija u nisci počinje brojem 0. U nisci koja označava RGB zapis boje, na poziciji 0 nalazi se numerički znak, a na pozicijama 1 i 2 heksadecimalni zapis crvenog spektra. Stoga su argumenti funkcije `substr()` za izdvajanje heksadecimalnog zapisa crvenog spektra 1 i 3.

Niske koja predstavlja heksadecimalni zapis boje, `cr`, `ze` i `pl`, prevode se u decimalne vrednosti s kojima se može računati (sabirati i oduzimati), korišćenjem funkcije `HexaUDec()`. Kako se svaki spektar boje zapisuje dvocifrenom heksadecimalnom vrednošću, on se pretvara u decimalnu vrednost, na primer u slučaju crvenog spektra, na sledeći način:

$$cr_{10} = cr^0 * 16 + cr^1$$

Ova funkcija ilustruje korišćenje još dve metode objekata tipa niska: to su `charAt()` i `indexOf()`. Prva metoda izdvaja karakter iz niske na datoj poziciji koja je argument metode, a druga sravnjuje nisku sa datim karakterom ili podnisom. U funkciji `HexaUDec()` prvo se iz niske argumenta izdvaja karakter na nultoj poziciji a zatim se određuje njegova pozicija u nisci koju čine sve cifre heksadecimalnog brojnog sistema, uređene po svojim rastućim vrednostima. Na taj način se, u stvari, dobija bročana vrednost cifre. Ova vrednost se množi s osnovom brojnog sistema 16, jer je to cifra na poziciji 16^1 . Sličan postupak se ponavlja i za cifru na prvoj poziciji u nisci, koja predstavlja cifru na poziciji 16^0 u heksadecimalnom brojnom sistemu.

Jednom prevedene u brojeve, promenljive koje predstavljaju tekuće boje osnovnog spektra uvećavaju se po koraku 8 — uvećavanje po manjem koraku menja neznatno boja i promena se teško uočava. Boje se menjaju tako što se prvo menjaju boje samo plavog spektra, i tek kad se sve one iscrpe (t.j., kada se premaši vrednost 256), menja se boja zelenog spektra s time da se boja plavog spektra vraća na početak (vrednost 0). Ceo postupak se ponavlja, sve dok se ne iscrpu sve kombinacije boja plavog i zelenog spektra (t.j., sve dok i jedna i druga vrednost premaše vrednost 256) kada se menja boja crvenog spektra s tim da se boje i plavog i zelenog spektra vraćaju na početak.

Kada se u okviru `if-else` iskaza odrede nove decimalne vrednosti boja osnovnog spektra treba ih ponovo pretvoriti u heksadecimalne vrednosti. To obavlja funkcija `DecUHex()`. Cifra na poziciji 16^1 dobija se kao celobrojni deo količnika

decimalne vrednosti i broja 16 (za određivanje celobrojne vrednosti koristi se matematička funkcija `Math.floor()`), dok se cifra na poziciji 16^0 dobija kao ostatak pri deljenju decimalne vrednosti i broja 16 (za određivanje ostatka pri deljenju koristi se operator %). Ovako određene vrednosti heksadecimalnih cifara pretvaraju se u uobičajeni heksadecimalni zapis primenom metode `charAt()` na nisku `hexa`, koja sadrži sve cifre heksadecimalnog brojnog sistema, uređene u rastućem redosledu. Tek iskaz `document.bgColor= "#" + cr + ze + pl;` stvarno menja boju pozadine dokumenta.

Da se boja pozadine ne bi promenila samo jednom, već da bi se stalno menjala u konstantnim vremenskim intervalima koristi se metod `setTimeout()` objekta `window` koja izvršava neki JavaScript kod posle isteka nekog specifikovanog vremena. Prvi argument funkcije je niska koja predstavlja JavaScript kod koji treba izvršiti, a drugi vremenski interval, zadat u milisekundama, posle čijeg isteka taj kod treba izvršiti. U našem primeru, kod koji treba izvršiti je sama funkcija `menjaj_boju()` u okviru koje se i poziva metod `setTimeout()`. Dakle, funkcija `menjaj_boju()` poziva samu sebe *iterativno*.

Bilo bi lepo da se ovaj vatromet boja može prekinuti. Aktivirati program *Notepad* i u JavaScript kod sadržan u dokumentu uneti izmene. Kraj funkcije `menjaj_boju()` modifikovati na sledeći način:

```
id = window.setTimeout("menjaj_boju()", 400);
return id;
```

Ova promena uvodi promenljivu `id` koja identifikuje postavljeni tajm-aut (jedan prozor ih može imati više, za razne događaje). Identifikacija tajm-auta je vrednost koju vraća poziv funkcije `setTimeout()`. Ovom izmenom tu istu vrednost vraća i poziv funkcije `menjaj_boju()`. Na kraj dokumenta, ispred etikete `</BODY>` dodati sledeći sadržaj:

```
<h3>Klikni kada zelis da prkines vatromet</h3>
<FORM NAME='pozadina'>
<INPUT TYPE='button' VALUE='Zaustavi se'
      onClick="window.clearTimeout(id);">
</FORM>
```

Zapamtiti izmene u datoteci **java-cas9a.htm** u katalogu **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:** (Opcija *Save as* menija *File*). Spustiti *Notepad* (a ne zatvoriti!) pritiskom na dugme `_` u gornjem desnom uglu pruge naslova. Aktivirati program *Netscape*. Otvoriti pripremljenu prezentaciju (Opcija *Open* menija *File*). Na ekranu na kome se menjaju boje pozadine uočljivo je dugme **Zaustavi se**. Ako se klikne na ovo dugme, vatromet boja prestaje. Dugme za zaustavljanje je ulazno polje formulara. Ovo ulazno polje je tipa dugme (`button`) i za njega je definisano atributom `onClick` upravljanje događajem — klikom mišem. Vrednost ovog atributa je poziv metode `clearTimeout()` objekta `window`. Ova funkcija prekida izvršavanje koda koje je odloženo pozivom metode `setTimeout()`. Jedini argument funkcije `clearTimeout()` je identifikacija koda čije je izvršavanje odgođeno, a to je ona vrednost koju vraća poziv funkcije `setTimeout()`.

Bilo bi korisno kada bi se u trenutku zaustavljanja vatrometa, ispisala heksadecimalna vrednost tekuće boje pozadine. Aktivirati program *Notepad* i u JavaScript kod sadržan u dokumentu uneti izmene. Na kraj odeljka <HEAD>, dodati još jednu funkciju:

```
function ispisi_boju(id) {
    document.pozadina.boja.value=document.bgColor;
    window.clearTimeout(id);
}
```

U formular koji je već uveden na kraju dokumenta, uneti još jedno ulazno polje, iza postojećeg dugmeta:

```
<BR><INPUT TYPE='text' NAME='boja' VALUE=''>
```

Promeniti vrednost atributa `onClick` ulaznog polja **Zaustavi** se tako da sada dobije vrednost `"ispisi_boju(id);"`. Zapamtiti izmene u datoteci **java-cas9a.htm** (opcija *Save* menija *File*) i ponovo aktivirati *Netscape*. Osvežiti starnicu (dugme *Reload*). Sada se na stranici uz dugme **Zaustavi** me pojavljuje i jedno ulazno polje formulara koje je sada prazno. Kada prvi put kliknete na dugme **Zaustavi** se ovo polje će se popuniti i u njemu će biti ispisana RGB vrednost tekuće boje pozadine. Ovo je rezultat rada funkcije `ispisi_boju()`. Ova funkcija ima samo dva iskaza: drugi je poziv metode `clearTimeout()` objekta `window`, na isti način kao i ranije, a prvi je postavljanje vrednosti ulaznog tekstuelnog polja formulara. Formular je nazvan **pozadina** (to je vrednost atributa `NAME` etikete <FORM>), a ovo novo ulazno tekstuelno polje je nazvano **boja** (to je vrednost atributa `NAME` etikete <INPUT>). Da bi se postavila vrednost tog polja koristi se njegovo svojstvo `value`, a samo polje se referiše hijerarhijski: polje **boja** unutar formulara **pozadina** koji je u tekućem dokumentu `document`.

Ova kreirana stranica bila bi još korisnija ako bi posle prekida vatromet mogao da se nastavi. Aktivirati program *Notepad* i u JavaScript kod sadržan u dokumentu uneti izmene. Na kraj dokumenta, ispred etikete </BODY>, dodati sledeći sadržaj:

```
<h3>Klikni kada zelis da nastavis vatromet</h3>
<INPUT TYPE='button' VALUE='Nastavi'
    onClick="menjaj_boju();"><BR>
```

Zapamtiti izmene u datoteci **java-cas9a.htm** (opcija *Save* menija *File*) i ponovo aktivirati *Netscape*. Osvežiti starnicu (dugme *Reload*). Sada se na stranici pojavljuje još jedno dugme — **Nastavi**. I za njega je definisano rukovanje događajem u slučaju klika mišom koje se sastoji u ponovnom pozivu funkcije `menjaj_boju()`. Isprobati naizmenično dugmad **Zaustavi** se i **Nastavi**.

Ovaj program bi bio još korisniji ako bi se u nekom manjem prozoru čuvala prethodno izabrana boja, tako da se može uporediti sa drugim bojama koje se pojavljuju na ekranu. Aktivirati program *Notepad* i u JavaScript kod sadržan u dokumentu uneti izmene. U funkciju `ispisi_boju()`, ispred iskaza kojim se tajm-aut anulira, dodati sledeće iskaze:

```

win = window.open("",
                  "mali",
                  "status,resizable,width=300,height=200");
var d = win.document;
d.write("<BODY BGCOLOR='" + document.bgColor + "'>");
d.write("<B>RGB = " + document.bgColor + "</B>");
d.close();

```

Otkucani tekst zapamtiti u datoteci pod imenom **java-cas9b.htm** u katalogu **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:** (Opcija *Save as* menija *File*). Spustiti *Notepad* (a ne zatvoriti!) pritiskom na dugme _ u gornjem desnom uglu pruge naslova. Aktivirati program *Netscape*. Otvoriti pripremljenu prezentaciju (Opcija *Open* menija *File*). Izgled ekrana je na početku nepromenjen. Kada se prvi put klikne na dugme **Zaustavi** me otvoriće se novi, mali prozor. (Ovaj prozor će možda odmah biti spušten u liniju poslova i da bi ga videli na ekranu treba ga podići.) Boja pozadine tog prozora biće ista kao na prvobitnom prozoru — to je izabrana boja. Ako se pritisnete na dugme **Nastavi** boja na velikom prozoru počinje da se menja, dok boja malog prozora ostaje nepromenjena. Pored već poznatih stvari, ovaj deo programa koristi metod `close()` objekta `document`. Ovaj metod prikazuje ceo sadržaj dokumenta koji je već napisan, a još nije prikazan. Kada se ceo html-dokument ispisuje koriseći metod `document.write()`, neophodno je koristiti `document.close()` da bi njegov sadržaj bio prikazan.

Još korisnije bi bilo ako bi se u manjem prozoru prikazivalo više izabranih boja, a ne samo jedna. U tom cilju mogu se koristiti okviri. Aktivirati program *Notepad* i u JavaScript kod sadržan u dokumentu uneti izmene. Na sam početak dokumenta, odmah iza **<HEAD>** etikete dodati sledeće iskaze:

```

<SCRIPT>
// otvori novi prozor
win = window.open("",
                  "mali",
                  "status,resizable,width=300,height=200");
// dinamički kreiraj okvire u tom novom prozoru
// koristi se specijalni about:blank URL da bi se dobili prazni
// okviri
win.document.write('<FRAMESET rows="50%,50%"
                  cols="50%,50%">');
win.document.write('<FRAME NAME="f1" SRC="about:blank">');
win.document.write('<FRAME NAME="f2" SRC="about:blank">');
win.document.write('<FRAME NAME="f3" SRC="about:blank">');
win.document.write('<FRAME NAME="f4" SRC="about:blank">');
win.document.write('</FRAMESET>');
// niz okvira kroz koji se kruži
okviri = new Array(win.f1, win.f2, win.f3, win.f4);
// tekuci broj brojac okvira
f = 0;
</SCRIPT>

```

Uneti izmene i u funkciju `ispisi_boju()`. Izbaciti iskaz kojim se otvara novi prozor (iskazi kojima se otvara novi, mali, prozor su sada odmah na početku dokumenta). Iskaze

koji ispisuju sadržaj dokumenta malog prozora (to su iskazi `d.write()`) zameniti sledećim iskazima:

```
okviri[f].document.write("<BODY BGCOLOR='" + document.bgColor +
                           "'>");
okviri[f].document.write("<B>RGB = " + document.bgColor + "</B>");
okviri[f].document.close();
f = (f + 1) % 4; // Uvecaj brojac okvira
```

Otkucani tekst zapamtiti u datoteci pod imenom **java-cas9c.htm** u katalogu **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:** (Opcija *Save as* menija *File*). Spustiti *Notepad* (a ne zatvoriti!) pritiskom na dugme _ u gornjem desnom uglu pruge naslova. Aktivirati program *Netscape*. Otvoriti pripremljenu prezentaciju (Opcija *Open* menija *File*). Izgled ekrana je na početku nepromenjen, osim što se mali prozor izdelfen na četiri okvira pojavljuje odmah na početku (to je rezultat premeštanja iskaza za otvaranje prozora na početak dokumenta). Okviri su na početku prazni i beli. Kada se prvi put klikne na dugme **Zaustavi** se prvi okvir će dobiti izabranu boju i u njemu će se ispisati RGB vrednost te boje. Kada se sledeći put klikne na ovo dugme drugi okvir će ispisati vrednost tekuće boje i tako redom. Prikazivanje boja u okvirima vrti se u krug, tako da kada se peti put klikne na dugme **Zaustavi** se boja se prikazuje u prvom okviru, šesti put u drugom i tako dalje.

Ovaj dokument koristi JavaScript program koji dinamiški generiše okvire u novom, malom prozoru. To se postiže višestrukom primenom metode `write` za dokument (`document`) u novom prozoru (`win`). Svaki od novogenerisanih okvira dobija ime kroz vrednost atributa `NAME`: `f1`, `f2`, `f3`, `f4`. Okviri su na početku prazni, to jest ne sadrže nikakav dokument, jer je vrednost atributa `SRC` specijalni URL `about:blank`. U programu se zatim uvodi niz `okviri` koji sadrži reference otvorenih okvira: `win.f1`, `win.f2`, `win.f3` i `win.f4`. Specijalna promenljiva `f` sadržaće u toku rada programa indeks tekućeg okvira u nizu `okviri`.

U funkciji `ispisi_boju()` se sada metod `write` primenjuje na dokument iz tekućeg okvira koji se referiše preko odgovarajućeg elementa niza `okviri[f]`. Brojač okvira uvećava se iskazom `f = (f + 1) % 4`. U ovom iskazu primenjuje se operator `%` koji računa ostatak pri celobrojnom deljenju. Tako kada brojač `f` dostigne vrednost 3, a to je indeks četvrtog okvira u nizu `okviri`, ovaj iskaz izračunava kao novu vrednost brojača 0.

Svojstva objekta dokument

Osim pomenutih svojstava, dokument kao objekat poseduje svojstva koja su nizovi veza (`links`) ili sidra (`anchors`) unutar dokumenta. Staviti disketu u disketni uređaj. Aktivirati program *Notepad*. Otvoriti datoteku **java-cas7b.htm** iz kataloga **gr1-r1** (opcija *Open* iz menija *File*). Iza prvog elementa `H3` (iza završne etikete `</H3>` otkucati sledeći sadržaj:

```
<H3>Ko je ovo?
<a href="Slike/gdja_stres.gif"
onMouseOver="status='Slika Fredove progoniteljke'; return true"
```

```
onMouseOut="status=' '>
</a></H3>
```

Zapamtiti izmenu u datoteci pod imenom **java-cas7c.htm** (opcija *Save As* menija *File*). Spustiti program *Notepad* i aktivirati program *Netscape*. Otvoriti pripremljenu prezentaciju (opcija *Open Page* menija *File*). U dokumentu se sada uz postojeću vezu pojavljuje još jedna. Aktivirati program *Notepad*. Na kraj elementa **SCRIPT** koji se nalazi unutar elementa **HEAD** uneti definiciju sledeće funkcije:

```
// Kreira se novi prozor i izlistavaju destinacije svih veza u
// dokumentu d. Za to se koristi text/plain dokument.
function prikazi_veze(d) {
    var novi = window.open("", "listaveza",
        "menubar,scrollbars,resizable,width=600,height=300");
    novi.document.open("text/plain");
    for (var i = 0; i < d.links.length; i++)
        novi.document.writeln(d.links[i]);
    novi.document.close();
}
```

Osim toga, u samom dokumentu, iza prethodno unete izmene dodati sledeći sadržaj:

```
<HR>
<h3>Klikni kada zelis da prikazes sve veze u dokumentu</h3>
<FORM>
<INPUT TYPE='button' VALUE='Prikazi'
    onClick="prikazi_veze(document);">
</FORM>
```

Zapamtiti izmenu (opcija *Save* menija *File*). Spustiti program *Notepad* i aktivirati program *Netscape*. Osvežiti prezentaciju (opcija *Reload* menija *File*). U dokumentu se sada pojavljuje i jedno dugme, **Prikazi**. Klikom miša na to dugme otvara se novi prozor u kome se prikazuju URL svih veza koje postoje u dokumentu. Otvaranje prozora i prikazivanje veza obavlja funkcija **prikazi_veze()** koja se poziva prilikom klika mišem na dugme formulara. U ovoj funkciji se, pre svega otvara novi prozor, a zatim se u okviru tog prozora otvara novi dokument korišćenjem metode **open()**. Ova metoda ima jedan argument i to je tip podataka koje dokument sadrži. Ovaj tip se zadaje kao jedan od standardnih MIME tipova koje prelistač razume ("text/html", "text/plain", "image/gif", i tako dalje). U našem primeru je izabran tip "text/plain" i stoga ima smisla da se za ispisivanje sadržaja dokumenta koristi metoda **writeln()** objekta dokument koja se ponaša isto kao i metoda **write()** samo što posle ispisanog sadržaja dodaje i karaktere za kraj reda (ti karakteri se u dokumentu tipa "text/html" ignorišu, u html-dokumentu su od značaja samo interpretacije odgovarajućih etiketa).

Svaki html-dokument ima svojstva **links[]** koji je niz **Link** objekata od kojih je svaki hipertekstuelna veza u dokumentu i **links.length** koji je ceo broj koji specifikuje broj elemenata u nizu **links[]**. Ova dva svojstva se koriste u funkciji **prikazi_veze()** za ispisivanje u novom prozoru svih veza dokumenta iz starog prozora.

Formular kao objekat

Formulari, kao i pojedinačni elementi formulara, su objekti koji imaju svojstva, metode i rukovaoce događajima. Staviti disketu u disketni uređaj. Aktivirati program *Notepad*. U programu otkucati sledeći sadržaj:

```
<HTML><HEAD><TITLE>Slova koja setaju</TITLE>
<SCRIPT LANGUAGE="JavaScript">
  var niska="Pozdrav svima! ~~~~ Pozdrav svima! ~~~~ ";
  var prazno=" "; var poz=0;
  function defile() {
    document.obrazac.rec.value=niska.substring(poz,niska.length)+
                                   niska.substring(0,poz);

    poz++;
    if(poz>niska.length) poz=0;
    window.setTimeout("defile()",200)
  }
</SCRIPT>
</HEAD>
<BODY BGCOLOR="#000033" ONLOAD="defile()">
<DIV ALIGN=CENTER>
<TABLE CELLPADDING="30">
<TR><TD BGCOLOR="#FFFF00">
<FORM NAME="obrazac">
  <INPUT TYPE="TEXT" NAME="rec" SIZE=30>
</FORM>
</TD></TR>
</TABLE>
</DIV>
</BODY>
</HTML>
```

Otkucani tekst zapamtiti u datoteci pod imenom **java-cas10.htm** u katalogu **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:** (Opcija *Save as* menija *File*). Spustiti aplikaciju (a ne zatvoriti!) pritiskom na dugme _ u gornjem desnom uglu pruge naslova. Aktivirati program *Netscape*. Otvoriti pripremljenu prezentaciju (Opcija *Open* menija *File*). Na ekranu se pojavljuje na teget pozadini, jedna žuta ćelija tabele, a u okviru nje jedno ulazno polje u okviru koga stalno promiče jedan te isti tekst: Pozdrav svima! Obrazac ima samo jedno ulazno polje koje je tipa **TEXT**. Vrednost tog polja je svojstvo **value** ulaznog polja čije ime **rec** (određeno kao vrednost atributa **NAME**) je svojstvo formulara čije ime **obrazac** (takode određeno kao vrednost atributa **NAME**) je svojstvo tekućeg dokumenta **document**. Ova vrednost se postavlja kao konkatencija dve podniskne niske **niska** u okviru funkcije **defile()**. Prva podniska čini kraj niske, iza pozicije **poz**, a druga njen početak do pozicije **poz**. Promenljiva **poz** na početku ima vrednost 0, a prilikom svakog poziva funkcije **defile()** uvećava se za jedan (iskaz **poz++**). Kada vrednost **poz** premaši dužinu niske, ona se ponovo vraća na 0. (Ovi koraci imaju isti smisao kao korišćenje operatora **%** u primeru prikazivanja boja u četiri okvira.) Funkcija **defile()** samu sebe poziva kroz pozivanje metode **setTimeout** objekta **window**, koja je već korišćena.

Staviti disketu u disketni uređaj. Aktivirati program *Notepad*. U programu otkucati sledeći sadržaj:

```
<HTML>
<HEAD>
<TITLE>Mali kalkulator</TITLE>
  <SCRIPT LANGUAGE="JavaScript">
    // Definisemo funkciju koju koristimo kasnije
    function izracunaj(f)
    {
      if (confirm("Jeste li sigurni?"))
        f.rezultat.value = eval(f.izraz.value)
      else
        alert("Dodjite nam opet")
    }
  </SCRIPT>
</HEAD>
<BODY BGCOLOR='#00FF7F'>
<H2 ALIGN=CENTER>Mali kalkulator</H2>
<FORM>
<TABLE ALIGN=CENTER BORDER=0 CELLSPACING=0 CELLPADDING=10>
<TR><TD BGCOLOR='#FFA500'>Unesi neki izraz:
<TD BGCOLOR='#FFA500'><INPUT TYPE="text" NAME="izraz" SIZE=15>
<TD BGCOLOR='#FFA500' ROWSPAN=2><INPUT TYPE="button" VALUE="Izracunaj"
onClick="izracunaj(this.form)">
</TR>
<TR><TD BGCOLOR='#FFA500'>Rezultat:
<TD BGCOLOR='#FFA500'><INPUT TYPE="text" NAME="rezultat" SIZE=15>
</TR>
</TABLE>
</FORM>
</BODY>
</HTML>
```

Otkucani tekst zapamtiti u datoteci pod imenom **java-cas11.htm** u katalogu **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:** (Opcija *Save as* menija *File*). Spustiti aplikaciju (a ne zatvoriti!) pritiskom na dugme _ u gornjem desnom uglu pruge naslova. Aktivirati program *Netscape*. Otvoriti pripremljenu prezentaciju (Opcija *Open* menija *File*). Na ekranu se pojavljuju na zelenoj pozadini dva ulazna polja i jedno dugme. U prvo ulazno polje treba uneti bilo kakav matematički izraz. Na primer, ako želite da izračunati koliko će iznositi plata od 1730.00 dinara pri povećanju od 25%, u ulazno polje treba uneti izraz $1730 * 1.25$ (ili, što je isto, $1730 + 1730 * 0.25$ ili $1730 + 1730 * 25 / 100$). Zatim treba pritisnuti dugme **Izracunaj**. Otvara se dijaloški prozor, pa, ako u njemu potvrdimo izračunavanje, u drugom ulaznom polju će se prikazati rezultat — 2162.5.

U JavaScript programu je za dugme **Izracunaj** definisano rukovanje događajem za klik mišem i to pozivom funkcije `izracunaj()` koja ima jedan argument `this.form` koji označava tekući objekat tipa `Form`. I prvo i drugo ulazno polje imaju imena, `izraz` i `rezultat`, koja su vrednosti atributa `NAME` odgovarajućih ulaznih polja. Funkcija `izracunaj` poziva metod `confirm()` objekta tipa `Window` u okiru `if-then-else` izraza, pa ako ova funkcija vrati vrednost `true` izračunava se vrednost izraza iz prvog ulaznog polja i ta vrednost postaje vrednost drugog ulaznog polja. Vrednost prvog ulaznog polja se referiše sa `f.izraz.value`, a vrednost drugog polja

sa `f.rezultat.value`. Za računanje vrednosti izraza koristi se u JavaScript ugrađena funkcija `eval()` čiji argument može biti bilo koja niska koja sadrži JavaScript iraz koji treba izračunati ili JavaScript iskaze koje treba izvršiti.

Koristeći ovaj mali kalkulator možete poželeti da izračunate vrednosti više izraza. Tada bi bilo pogodno uvesti još jedno dugme `Izbrisi` koje posle jednog izračunavanja briše oba ulazna polja. Aktivirati program *Notepad*. Pre svega, izbaciti atribut `ROWSPAN` u ćeliji tabele koja sadrži dugme `Izracunaj`, jer se sada uvodi još jedno dugme koje će zauzeti treću ćeliju u drugom redu tabele. Poslednju ćeliju u tabeli ubaciti ispred etikete `</TABLE>`:

```
<TD BGCOLOR='#FFA500'><INPUT TYPE="button" VALUE="Izbrisi"
    onClick="this.form.izraz.value='';this.form.rezultat.value='';">
```

Zapamtiti izmenu (opcija *Save* menija *File*). Spustiti program *Notepad* i aktivirati program *Netscape*. Osvežiti prezentaciju (opcija *Reload* menija *File*). U dokumentu se sada pojavljuje još jedno dugme, `Izbrisi`. Klikom miša na to dugme briše se tekući sadržaj ulaznih polja. Za ovo dugme je takođe definisano rukovanje događajem za klik mišem i to sa dva JavaScript iskaza koja postavljaju vrednosti ulazni polja na praznu nisku.

Rukovanje događajima se često koriste uz html-formulare radi provere unosa, kad god se ona može izvršiti na klijentskoj strani. Staviti disketu u disketni uređaj. Aktivirati program *Notepad*. U programu otkucati sledeći sadržaj:

```
<HTML>
<HEAD>
<TITLE>Rukovanje dogadjajima - provera podataka u formularu</TITLE>
<SCRIPT LANGUAGE="JavaScript">
    function jeliPozBr(s) {
        // Funkcija vraca true ako je broj pozitivan
        return ( parseInt(s) > 0 )
    }
    function jeliPrazno(s) {
        // Funkcija vraca true ako je niska prazna
        // (sadrzi samo blanko karaktere)
        vrati = true;
        for( var i=0; i<s.length; i++ ) {
            if (s.charAt(i) != " ")
                vrati = false;
            break;
        }
        return ( vrati )
    }
    function prover_i_prazno(i) {
        // Funkcija vraca true ako polje nije prazno
        if ( jeliPrazno (i.value) ) {
            alert("Molim Vas popunite Vase ime!");
            vrati = false;
        }
        else
            vrati = true;
        return vrati;
    }
}
```

```

function prover_i_kol(kol, min, max) {
// Proveri unetu vrednost
  var vrati = false;
  if ( !jeliPozBr(kol.value) )
    alert("Molim Vas unesite pozitivan broj")
  else if (parseInt(kol.value) < min)
    alert("Morate poruciti bar " + min + " zveckih")
  else if (parseInt(kol.value) > max)
    alert("Ne mozete poruciti vise od " + max + " zveckih")
  else
    vrati = true;
  return vrati;
}
function prover_i_posalji(formular) {
// Samo ako je sve u redu posalji formular
if( (proveri_kol(formular.kolicina,10,999)) &&
    (proveri_prazno(formular.ime)) ) {
  alert("Narudbina je poslata");
  formular.submit();
  return true;
} else {
  alert("Izvinite, ovu narudzbenu ne mozete poslati");
  return false;
}
}
}
</SCRIPT>
</HEAD>
<BODY BGCOLOR="#FFF0F5" TEXT="#000080">
<H2 ALIGN=CENTER>Kupovina zveckih</H2>
<HR>
<TABLE ALIGN=CENTER BORDER=0 CELLSPACING=0 CELLPADDING=20>
<TR><TD BGCOLOR="E6E6FA">
<FORM NAME="zvecke" ACTION="lwapp.html" METHOD="post">
<B>Kako se zovete?</B><BR>
<INPUT TYPE='text' NAME='ime' onChange="proveri_prazno(this)">
<br>
<B>Koliko zveckih narucujete danas?</B><BR>
<INPUT TYPE="text" NAME="kolicina"
onChange="proveri_kol(this,10,999)"><BR>
<INPUT TYPE="submit" Value="Posalji narudzbu"
onSubmit="proveri_i_posalji(zvecke)"><br>
</FORM>
</TD></TR></TABLE>
<HR>
</BODY>
</HTML>

```

Otkucani tekst zapamtiti u datoteci pod imenom **java-cas12.htm** u katalogu **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:** (Opcija *Save as* menija *File*). Spustiti aplikaciju (a ne zatvoriti!) pritiskom na dugme **_** u gornjem desnom uglu pruge naslova. Aktivirati program *Netscape*. Otvoriti pripremljenu prezentaciju (Opcija *Open* menija *File*). Na ekranu se pojavljuje jedan formular-narudžbenica koji služi za naručivanje zveckih. Narudžbenica se obrađuje na serverskoj strani, a na klijentskoj se može proveriti ispravnost popunjavanja formulara. Ne prihvataju se narudžbenice kod kojih je nepopunjeno ime naručioca i one koje naručuju nedovoljnu količinu proizvoda (manju od

10) ili preveliku količinu proizvoda (više od 999). Isprobati formular unošenjem i ispravnih i neispravnih vrednosti polja.

Za oba ulazna polja, `ime` i `kolicina`, definisano je rukovanje događajima i to za promenu sadržaja polja (`onChange`). U slučaju imena naručioca, pri promeni sadržaja polja poziva se funkcija `proveri_prazno()` koja proverava da li je ulazno polje prazno. Argument ove funkcije je samo ulazno polje formulara (`this`). Ako jeste (što proverava funkcija `jeliPrazno()`), poziva se metod `alert()` koji u dijaloškom prozoru upozorava korisnika da popuni polje na ispravan način. U slučaju količine naručenih proizvoda, pri promeni sadržaja polja poziva se funkcija `proveri_kol()`. Ova funkcija ima tri argumenta: samo ulazno polje formulara (`this`), najmanju dozvoljenu vrednost i najveću dozvoljenu vrednost. Pri pozivu funkcije proverava se da li je unet pozitivan broj (poziv funkcije `jeliPozBr()`), da li je taj broj veći od najmanje dozvoljene vrednosti i da li je manji od najmanje dozvoljene vrednosti. Ako je bilo koja od ovih provera neuspešna, funkcija poziva metod `alert()` koji u dijaloškom prozoru upozorava korisnika da popuni polje na ispravan način. Funkcije `proveri_kol()` i `jeliPozBr()` koriste funkcije ugrađenu funkciju `parseInt()` koja nisku, njen argument, a u ovom slučaju sadržaj ulaznog polja `kolicina`, konvertuje u ceo broj. Za dugme `Posalji narudzbu` takođe je definisano rukovanje događajem za klik mišem. U ovom slučaju poziva se funkcija `proveri_i_posalji()` koja proverava formular kao celinu. Argument ove funkcije je sam formular — `zveck`, što je ime formulara (isto tako, argument bi mogao da bude i `this.form`). Formular se šalje serveru (to jest, narudžbenica se prosleđuje) samo ako su oba ulazna polja ispravna, to jest, i funkcija `proveri_prazno()` i funkcija `proveri_kol()` vrate vrednost `true`. Da bi formular bi prosleđen serveru poziva se metod `submit()` objekta tipa `Form`.

Naravno, formular će moći da se prosledi samo ako stvarno postoji veza sa serverom. Umesto da se direktno poziva metod `submit()` može se dugme `Posalji narudzbu` definisati kao ulazno polje tipa `submit` i za to ulazno polje može se definisati rukovanje događajem u slučaju prosleđivanja formulara — `onSubmit`. Ovaj događaj bi se obrađivao opet pozivom funkcije `proveri_i_posalji()`.

Veliki primer — formulari i obrada događaja

Staviti disketu u disketni uređaj. Aktivirati program *Notepad*. U programu otkucati sledeći sadržaj:¹

```
<HTML>
<HEAD>
<TITLE>Kako radi formular sa obradom dogadjaja</TITLE>
<SCRIPT LANGUAGE="JavaScript1.1">
// Ova funkcija dodaje detalje o jednom dogadjaju u veliku tekstalnu //
zonu koja je definisana u tabeli. Ona se poziva prilikom rukovanja
// razlicitim dogadjajima
function izvestaj(element, dogadjaj)
{
```

¹ Ovaj program je preuzet i modifikovan primer iz knjige *JavaScript—the Definitive Guide* od Davida Flanagana u izdanju O'Reilly&Associates, Inc.

```

var t = element.form.tekstzona;
var ime = element.name;
if ((element.type == "select-one") ||
    (element.type == "select-multiple")) {
    vrednost = ' ';
    for (var i = 0; i < element.options.length; i++)
        if (element.options[i].selected)
            vrednost += element.options[i].value + " ";
}
else if (element.type == "textarea") vrednost = "...";
else vrednost = element.value;
var poruka = dogadjaj + ": " + ime + " (" + vrednost + ")\n";
t.value=t.value + poruka;
}

// Ova funkcija dodaje gomilu rukovanja dogadjajima svakom elementu
// u formularu. Ona ne proverava da li taj element podrzava neku obradu
// dogadjaja - ona ih dodaje sve. Funkcije rukovanja dogadjajima
// pozivaju funkciju *izvestaj*.
function dodaj(f)
{
    for (var i=0; i < f.elements.length; i++) {
        var e=f.elements[i];
        e.onclick = new Function("izvestaj(this, 'Klik')");
        e.onchange = new Function("izvestaj(this, 'Izmena')");
        e.onfocus = new Function("izvestaj(this, 'Focus')");
        e.onblur = new Function("izvestaj(this, 'Povlacenje')");
        e.onselect = new Function("izvestaj(this, 'Izbor')");
    }
    // Specijalno rukovanje za dugmice:
    f.brisi.onclick=
        new Function(
            "this.form.tekstzona.value=''; izvestaj(this, 'Klik');");
    f.salji.onclick=
        new Function("izvestaj(this, 'Klik'); return false;");
    f.pocetak.onclick=
        new Function(
            "this.form.reset(); izvestaj(this, 'Klik'); return false");
}
</SCRIPT>
</HEAD>
<BODY>
<!-- Jedan veliki formular sa po jednim ulaznim poljem za
svaki tip ulaznih polja -->
<h3 align=center><FONT COLOR=red>Kako radi obrada dogadjaja</FONT></h3>
<FORM NAME="sve">
<!-- Ulazna polja smestaju se u veliku tabelu -->
<TABLE BORDER CELLPADDING=5>
    <TR>
        <TD>Ime korisnika (tip <FONT COLOR=red>text</FONT>):<BR>
            [1]<INPUT TYPE=text NAME='ime' SIZE=15></TD>
        <TD>Lozinka (tip <FONT COLOR=red>password</FONT>):<BR>
            [2]<INPUT TYPE=password NAME='lozinka' SIZE=15></TD>
        <TD ROWSPAN=4>Ulazni dogadjaji (tip
            <FONT COLOR=red>textarea</FONT>):<BR>
            [3]<TEXTAREA NAME="tekstzona" ROWS=20 COLS=25>
            </TEXTAREA></TD>

```

```

        <TD ROWSPAN=4 ALIGNE=right VALIGN=center>Brisanje (tip
            <FONT COLOR=red>button</FONT>):<BR>
            [9]<INPUT TYPE=button VALUE='brisi' NAME='brisi' SIZE=15><BR>
            Slanje (tip <FONT COLOR=red>submit</FONT>):<BR>
            [10]<INPUT TYPE=submit VALUE='salji' NAME='salji'
                SIZE=15></BR>
            Iz pocetka (tip <FONT COLOR=red>reset</FONT>):<BR>
            [11]<INPUT TYPE=reset VALUE='pocetak' NAME='pocetak'
                SIZE=15></BR></TD></TR>
    <TR>
        <TD COLSPAN=2>Ime datoteke (tip
            <FONT COLOR=red>file</FONT>)</BR>
        [4]<INPUT TYPE=file NAME='datoteka' SIZE=15></TD></TR>
    <TR>
        <TD>Moj racunar ima (<FONT COLOR=red>checkbox</FONT>)<BR>
        [5]<INPUT TYPE=checkbox NAME='uredjaj' VALUE='modem'>
            28.8K Modem<BR>
        [5]<INPUT TYPE=checkbox NAME='uredjaj' VALUE='stampac'>
            HP LaserJet<BR>
        [5]<INPUT TYPE=checkbox NAME='uredjaj' VALUE='kompakt'>
            CD ROM</TD>
        <TD>Moj prelistac je (<FONT COLOR=red>radio</FONT>)<BR>
        [5]<INPUT TYPE=radio NAME='prelistac' VALUE='nn'>
            Netscape Navigator<BR>
        [5]<INPUT TYPE=radio NAME='prelistac' VALUE='ie'>
            Internet Explorer<BR>
        [5]<INPUT TYPE=radio NAME='prelistac' VALUE='drugi'>
            Neki drugi</TD></TR>
    <TR>
        <TD>Moj hobi su [7]: (<FONT COLOR=red>select</FONT>)<BR>
        <SELECT multiple NAME='hobi' SIZE=4>
            <OPTION VALUE='prog'>Programiranje JavaScriptom
            <OPTION VALUE='surf'>Surfanje po Internetu
            <OPTION VALUE='kafa'>Ispijanje kafe
            <OPTION VALUE='igra'>Igranje igrice
        </SELECT></TD>
        <TD ALIGN=center VALIGN=center>Moja omiljena boja je [8]:<BR>
        <SELECT NAME="boja">
            <OPTION VALUE="cr">Crvena          <OPTION VALUE="za">Zelena
            <OPTION VALUE="pl">Plava           <OPTION VALUE="be">Bela
            <OPTION VALUE="lj">Ljubicasta    <OPTION VALUE="zu">Zuta
        </SELECT></TD></TR>
</TABLE>
</FORM>
<SCRIPT LANGUAGE="JavaScript1.1">
// Aktiviranje formulara dodavanjem obrada dogadjaja
dodaj(document.sve);
</SCRIPT>
</BODY>
</HTML>

```

Otkucani tekst zapamtiti u datoteci pod imenom **java-cas13.htm** u katalogu **gr1-r1** (za prvu grupu, prvi računar) na disketnoj jedinici **A:** (Opcija *Save as* menija *File*). Spustiti aplikaciju (a ne zatvoriti!) pritiskom na dugme **_** u gornjem desnom uglu pruge naslova. Aktivirati program *Netscape*. Otvoriti pripremljenu prezentaciju (Opcija *Open* menija *File*). Na ekranu se pojavljuje jedan složeni formular koji se sastoji od skoro svih

mogućih tipova elemenata formulara. Popunjavajte formular redom, ili preko reda, a u velikom ulaznom polju (element tipa `<TEXTAREA>`) biće verno ocrtane sve akcije koje preduzimate. Pritiskom na dugme `brisi` briše se sadržaj ovog velikog ulaznog polja (to je smisao funkcija `onclick` koji se pridružuje elementu `brisi` fromulara `f` a čija je vrednost funkcija definisana konstruktorom

```
new Function("this.form.tekstzona.value='';  
izvestaj(this, 'Klik');");).
```

Pritiskom na dugme `pocetak` briše se sadržaj celog formulara (to je smisao funkcije `onclick` koji se pridružuje elementu `pocetak` fromulara `f` a čija je vrednost funkcija definisana konstruktorom `new Function(("izvestaj(this, 'Klik');
return false;"))`).

Svaki objekat tipa `Form` ima svojstvo `elements[]` koji je niz JavaScript objekata koji predstavljaju ulazne elemente formulara. Postoji više mogućih HTML elemenata formulara i odgovarajućih Java Script objekata.

Dok svaki tip JavaScript elementa formulara ima svoje osobenosti postoji određen broj osobina koje su svojstvene svim elementima formulara. Jedna od njih je da je za skoro sve elemente formulara definisana obrada događaja, to jest, definisane su funkcije koje se pozivaju kada korisnik sa njima komunicira, kao na primer, `onClick()` i `onChange()`. Osim njih, svi elementi formulara podržavaju i obradu događaja sa `onFocus()` i `onBlur()`, koji se pozivaju kada ulazno polje dobije, odnosno, izgubi ulazni fokus sa tastature. Osim njih, svaki element formulara ima i odgovarajuće metode `focus()` i `blur()` koji usmeravaju ili uklanjaju fokus sa nekog ulaznog polja.

Sledeća sličnost između objekata koji odgovaraju elementima formulara je da svi oni imaju svojstvo `type` koji identifikuje tip odgovarajućeg elementa. Kako niz `elements[]` objekta `Form` sadrži različite tipove elemenata formulara, svojstvo `type` omogućava programeru da prolazi kroz sve elemente tog niza i da ih obrađuje na način koji zavisi od njihovog tipa. Svi elementi formulara imaju takođe i svojstvo `form` koje referiše objekat `Form` koji sadrži određeni element. To je pogodan način da funkcije za obradu događaja jednog formulara referišu elemente drugog formulara. Unutar funkcije za obradu događaja jednog elementa ključna reč `this` referiše sam element. Prema tome `this.form` uvek referiše formular koji ga sadrži.

Dodaci

1. Pregled operatora

P ²	A ³	Operator	Tipovi operatora	Operacija
0	L	.	objekat, svojstvo	pristup svojstvu
	L	[]	niz, ceo broj	indeks niza
	L	()	funkcija, srgumenti	poziv funkcije
1	R	++	broj	pre ili post inkrement (unarna)
	R	--	broj	pre ili post inkrement (unarna)
	R	-	Broj	negativna vrednost (unarna)
	R	~	Ceo broj	negacija bit po bit (unarna)
	R	!	logička	bulovska negacija
	R	typeof	bilo koji	tip podatka (unarna)
	R	new	poziv konstrukora	kreira novi objekat (unarna)
	R	void	bilo koji	vraće nedefinisanu vrednost
2	L	*,/,%	brojevi	množenje, deljenje, ostatak
3	L	+, -	brojevi	sabiranje, oduzimanje
	L	+	niske	konkatenacija
4	L	<<	celi brojevi	pomeranje bitova ulevo
	L	>>	celi brojevi	pomeranje bitova udesno sa popunjavanjem znaka
	L	>>>	celi brojevi	pomeranje bitova ulevo sa popunjavanjem nulama
5	L	<,<=	brojevi ili niske	manje od, manje ili jednako
	L	>,>=	brojevi ili niske	veće od, više ili jednako
6	L	==	primitivni tip	jednako (ima istu vrednost)
	L	!=	primitivni tip	nejednako (nema istu vrednost)
	L	==	referentni tip	jednako (referiše isti objekat)
	L	!=	referentni tip	nejednako (ne referiše isti objekat)
7	L	&	ceo broj	AND bit po bit
8	L	^	ceo broj	XOR bit po bit
9	L		ceo broj	OR bit po bit
10	R	&&	logička	bulovsko AND
11	R		logička	bulovsko OR
12	R	?:	logička, bilo koji, bilo koji	uslovno izvršavanje (ternarna)
13	R	=	promenljiva, bilo koji	dodela
	R	*,/,%=	promenljiva, bilo koji	dodela sa operacijom
		+=, -=,		
		<<=, >>=,		
		&=, ^=, =		
14	L	,	bilo koji	sekvencija izvršavanja

² Prioritet operatora

³ Asocijativnost operatora: **L** - s leva u desno, **R** - s desna u levo

2. Sintaksa JavaScript iskaza

Iskaz	Sintaksa	Namena
break	break;	Izlazak iz unutrašnje petlje
continue	continue;	Skok na početak unutrašnje petlje
prazan	;	Ništa ne radi
for	for (<i>inicijalizacija; test; inkrement</i>) <i>iskaz</i>	Jednostavna petlja
for/in	for (<i>promenljiva in objekat</i>) <i>iskaz</i>	Petlja kroz svojstva objekta
function	function <i>imefunk</i> ([<i>arg1</i> [... <i>argn</i>]]) { <i>iskaz</i> }	Definicija funkcije
if/else	if (<i>izraz</i>) <i>iskaz 1</i> else <i>iskaz 2</i>	Uslovno izvršavanje koda
return	return <i>izraz</i>	Vraća vrednost pri izlasku iz funkcije
var	var <i>ime_1</i> [=vred_1] [... <i>ime_n</i> [=vred_n]];	Deklariše i inicijalizuje vrednost
while	while (<i>izraz</i>) <i>iskaz</i>	Osnovna petlja
with	with (<i>objekat</i>) <i>iskaz</i>	Specifikuje tekući prostor imena

3. Obrada događaja

Objekat	Podržana obrada događaja		
Area	onClick()	onMouseOut()	onMouseOver()
Button	onBlur()	onClick()	onFocus()
Checkbox	onBlur()	onClick()	onFocus()
FileUpload	onBlur()	onChange()	onFocus()
Form	onReset()	onSubmit()	
Frame	onLoad()	onUnload()	
Image	onAbort()	onError()	onLoad()
Link	onClick()	onMouseOut()	onMouseOver()
Radio	onBlur()	onClick()	onFocus()
Reset	onBlur()	onClick()	onFocus()
Select	onBlur()	onChange()	onFocus()
Submit	onBlur()	onClick()	onFocus()
Text	onBlur()	onChange()	onFocus()
Textarea	onBlur()	onChange()	onFocus()
Window	onBlur()	onError()	onFocus()
	onLoad()	onUnload()	

4. Elementi formulara

Objekat	HTML etiketa	Svojstvo type	Opis i događaji
Objekat	<INPUT TYPE=button>	“button”	Dugme na pritiskanje; onClick()
Checkbox	<INPUT TYPE=checkbox>	“checkbox”	Dugme uključno/isključeno; onClick()
FileUpload	<INPUT TYPE=file>	“file”	Ulazno polje u koje se unosi ime datoteke koja se postavlja na web server; onChange()
Hidden	<INPUT TYPE=hidden>	“hidden”	Podaci koji se šalju sa formularom ali nisu vidljivi za korisnika; nema obrade događaja
Option	<OPTION>	none	Jedna stavka unutar Select objekta; obrada događaja ide uz Select objekat
Password	<INPUT TYPE=password>	“password”	Ulazno polje u koje se unosi lozinka — ukucani karakteristi se ne vide; onChange()
Radio	<INPUT TYPE=radio>	“radio”	Dugme uključno/isključeno od koji samo jedno može biti uključeno; onClick()
Reset	<INPUT TYPE=reset>	“reset”	Dugme na pritiskanje koje poništava formular; onClick()
Select	<SELECT>	“select-one”	Padajući meni iz koga se može izabrati jedna stavka; onChange()
Select	<SELECT MULTIPLE>	“select-multiple”	Padajući meni iz koga se može izabrati više stavki; onChange()
Submit	<INPUT TYPE=submit>	“submit”	Dugme za pritiskanje koje šalje formular; onClick()
Text	<INPUT TYPE=text>	“text”	Tekstualno ulazno polje od jednog reda; onChange()
Textarea	<TEXTAREA>	“textarea”	Tekstualno ulazno polje od više redova; onChange()

5. JavaScript objekti i metodi

Array objekat — ugrađena podrška nizovima

Konstruktori

```
new Array()  
new Array(vel)  
new Array(element0, element1, ..., elementn)
```

Argumenti

vel Potreban broj elemenata u nizu. Vrednost svojstva `length` konstruisanog niza postavlja se na `size`.

element0, element1, ..., elementn Lista od dve ili više proizvoljnih vrednosti. Kada se konstruktor `Array` poziva sa ovakvim argumentima onda će elementi novokreiranog niza imati redom vrednosti koje su zadate listom argumenata a svojstvo `length` će imati vrednost broja argumenata u listi.

Povratna vrednost

Novokreiran i inicijalizovan niz. Kada se `Array()` poziva bez argumenata svojstvo `length` niza koji se kreira postavlja se na 0.

Svojstva

`length` Ceo broj koji se može čitati i u koji se može pisati koji specifikuje broj elemenata niza, a kada niz nema kontingentnih elemenata, onda broj za jedan veći od indeksa poslednjeg elementa niza.

Metode

`join` Konvertuje sve lemente niza u nisku i vrši njihovu konkatenciju.

`reverse` Obrće na licu mesta sve elemente niza.

`sort` Sortira na licu mesta sve elemente niza.

Boolean objekat — ugrađena podrška Bulovskim vrednostima

Konstruktori

```
new Boolean(vred)
```

Argumenti

vred Vrednost koju treba da ima Bulovski objekat. Argument se po potrebi konvertuje u Bulovsku vrednost: 0, null i prazne niske se konvertuju u false, a sve ostale vrednosti u true.

Povratna vrednost

Novokreiran i inicijalizovan niz.

Metode

`toString()` Vraća niske "true" ili "false" u zavisnosti od toga kakvu vrednost reprezentuje Bulovski objekat.
`valueOf()` Vraća vrednost koju reprezentuje Bulovski objekat.

Napomena

Bulovske vrednosti su fundamentalni tip podataka u JavaScriptu. Bulovski objekat je omotač oko Bulovske vrednosti koji služi samo da obezbedio metodi `toString()` da konvertuje Bulovsku vrednost u nisku.

Button element— grafičko dugme na pritiskanje

Korišćenje

```
form.elements[i]
form.ime_dugmeta
```

Svojstva

`form` Referenca koja može samo da se čita objekta Form koji sadrži ovo dugme.
`name` Niska koja može samo da se čita, koju postavlja HTML atribut NAME, a koja specifikuje ime dugmeta.
`type` Niska koja može samo da se čita a koja specifikuje tip elementa formulara. Za dugme ima vrednost "button".
`value` Niska koja može samo da se čita a koje specifikuje HTML atribut VALUE, a koji specifikuje vrednost prikazanu elementom Button.

Metode

`blur()` Uklanjanje fokusa sa dugmeta.
`click()` Simuliranje klika mišem na dugme.
`focus()` Fokusiranje dugmeta.

Obrada događaja

<code>onblur()</code>	Poziva se kada korisnik ukloni fokus sa dugmeta.
<code>onclick()</code>	Poziva se kada se klikne na dugme.
<code>onfocus()</code>	Poziva se kada se dugme fokusira.

Checkbox element— grafička kućica za čekiranje

Korišćenje

```
form.elements[i]  
form.ime_kucice
```

Svojstva

<code>checked</code>	Bulovska vrednost koja može da se čita i da se menja i koja specifikuje da li je kućica čekirana.
<code>defaultChecked</code>	Bulovska vrednost koja može samo da se čita i koja specifikuje početno stanje kućice. Može se specifikovati HTML atributom <code>CHECKED</code> .
<code>form</code>	Referenca koja može samo da se čita objekta <code>Form</code> koji sadrži ovu kućicu.
<code>name</code>	Niska koja može samo da se čita, koju postavlja HTML atribut <code>NAME</code> , a koja specifikuje ime kućice.
<code>type</code>	Niska koja može samo da se čita a koja specifikuje tip elementa formulara. Za kućicu za čekiranje ima vrednost <code>"checkbox"</code> .
<code>value</code>	Niska koja može da se čita i da se menja a koju na početku može da specifikuje HTML atribut <code>VALUE</code> , i koja specifikuje vrednost prikazanu elementom <code>Checkbox</code> , ako je izabran.

Metode

<code>blur()</code>	Uklanjanje fokusa sa kućice.
<code>click()</code>	Simuliranje klika mišem na kućicu.
<code>focus()</code>	Fokusiranje kućice.

Obrada događaja

<code>onblur()</code>	Poziva se kada korisnik ukloni fokus sa kućice.
<code>onclick()</code>	Poziva se kada se klikne na kućicu.
<code>onfocus()</code>	Poziva se kada korisnik fokusira kućicu.

Date objekat — manipulacija datumom i vremenom

Konstruktori

```
new Date()
new Date(milisekunde)
new Date(datumkaoniska)
new Date(godina, mesec, dan)
new Date(godina, mesec, dan, cas, minuti, sekunde)
```

Kada je bez argumenata, konstruktor `Date()` kreira objekat tipa `Date` koji je postavljen na tekuće vreme i na tekući datum (odnosno, sistemsko vreme i sistemski datum).

`Date()` se može koristiti i kao obična funkcija (bez `new` konstruktora). Tada, ako se koristi bez argumenata, vraća nisku koja predstavlja lokalni datum i vreme.

Argumenti

milisekunde
Broj milisekundi između željenog datuma i ponoći 1. januara 1970. po griničkom vremenu.

datumkaoniska
Argument koji specifikuje datum i, eventualno, vreme kao nisku u obliku *mesec dan, godina cas:minuti:sekunde*. Na primer: "March 24, 1999 20:05:33"

<i>godina</i>	Godina kao četvorocifreni ceo broj.
<i>mesec</i>	Mesec specifikovan kao ceo broj od 0 (januar) do 11(decembar).
<i>dan</i>	Dan u mesecu specifikovan kao ceo broj od 1 do 31.
<i>cas</i>	čas specifikovan kao ceo broj od 0 (ponoć) do 23 (11 uveče).
<i>minuti</i>	Minuti specifikovani kao ceo broj od 0 do 59.
<i>sekunde</i>	Sekunde specifikovane kao ceo broj od 0 do 59.

Povratna vrednost

Novokreirani objekat tipa `Date`.

Metode

<code>getDate()</code>	Vraća dan u mesecu objekta tipa <code>Date</code> .
<code>getDay()</code>	Vraća dan u nedelji objekta tipa <code>Date</code> .
<code>getHours()</code>	Vraća polje časova objekta tipa <code>Date</code> .
<code>getMinutes()</code>	Vraća polje minuta objekta tipa <code>Date</code> .
<code>getMonth()</code>	Vraća polje meseca objekta tipa <code>Date</code> .
<code>getSeconds()</code>	Vraća polje sekundi objekta tipa <code>Date</code> .

<code>getTime()</code>	Vraća internu reprezentaciju u milisekundama objekta tipa <code>Date</code> .
<code>getTimezoneOffset()</code>	Vraća vremensku razliku u minutama između datuma i GMT (griničkog vremena).
<code>getYear()</code>	Vraća polje godine objekta tipa <code>Date</code> .
<code>parse()</code>	Parsira reprezentaciju datuma kao niske i vraća isti datum u obliku milisekundi
<code>setDate()</code>	Postavlja polje dana u mesecu objekta tipa <code>Date</code> .
<code>setHours()</code>	Postavlja polje časova objekta tipa <code>Date</code> .
<code>setMinutes()</code>	Postavlja polje minuta objekta tipa <code>Date</code> .
<code>setMonth()</code>	Postavlja polje meseca objekta tipa <code>Date</code> .
<code>setSeconds()</code>	Postavlja polje sekundi objekta tipa <code>Date</code> .
<code>setTime()</code>	Postavlja polja objekta tipa <code>Date</code> korišćenjem formata u obliku milisekundi.
<code>setYear()</code>	Postavlja polje godine objekta tipa <code>Date</code> .
<code>toGMTString()</code>	Konvertuje datum u nisku koristeći GMT vremensku zonu (griničko vreme).
<code>toLocaleString()</code>	Konvertuje datum u nisku koristeći lokalnu vremensku zonu.
<code>UTC()</code>	Konvertuje numeričku specifikaciju datuma i vremena u format u obliku milisekundi.

Document objekat — trenutno prikazani HTML dokument

Korišćenje

```
prozor.document
document
```

Primerak objekta tipa `Dokument` je smešten u polje `document` objekta tipa `Window`. U specijalnom slučaju, kada se referiše objekta tipa `Dokument` tekućeg prozora (onog u kome se izvršava JavaScript program) može se izostaviti referisanje objekta `window`.

Svojstva

<code>alinkColor</code>	Niska koja specifikuje boju aktivirane veze. Može biti postavljena u zaglavlju HTML dokumenta ili kao vrednost atributa <code>ALINK</code> elementa <code><BODY></code> .
<code>anchors[]</code> <code>anchors.length</code>	Niza sidra u dokumentu; po jedan element za svaku vezu polaska. Ceo broj koji može samo da se čita i koji specifikuje broj elemenata u nizu <code>anchors[]</code> .

<code>applets[]</code>	Niza Java objekata u dokumentu; po jedan element za svaki <code><APPLET></code> koji se pojavljuje u dokumentu.
<code>applets.length</code>	Ceo broj koji može samo da se čita i koji specifikuje broj elemenata u nizu <code>applets[]</code> .
<code>bgColor</code>	Niska koja specifikuje boju pozadine dokumenta. Na početku se postavlja kao vrednost atributa <code>BGColor</code> elementa <code><BODY></code> .
<code>cookie</code>	Niska čija je vrednost 'cookie' koji je pridružen dokumentu.
<code>domain</code>	Niska koja specifikuje Internet domen sa koga je dokument. Koristi se iz razloga bezbednosti.
<code>embeds[]</code>	Niz Java objekata, po jedan element za svaki <code><EMBED></code> koji se pojavljuje u dokumentu.
<code>embeds.length</code>	Ceo broj koji može samo da se čita i koji specifikuje broj elemenata u nizu <code>embeds[]</code> .
<code>fgcolor</code>	Niska koja specifikuje boju teksta dokumenta. Može biti postavljena u zaglavlju HTML dokumenta ili kao vrednost atributa <code>TEXT</code> elementa <code><BODY></code> .
<code>forms[]</code>	Niz Form objekata, po jedan element za svaki <code><FORM></code> koji se pojavljuje u dokumentu.
<code>forms.length</code>	Ceo broj koji može samo da se čita i koji specifikuje broj elemenata u nizu <code>forms[]</code> .
<code>images[]</code>	Niz Image objekata, po jedan element za svaki <code></code> koji se pojavljuje u dokumentu.
<code>images.length</code>	Broj elemenata u nizu <code>images[]</code> .
<code>lastModified</code>	Niska koja može samo da se čita i koja specifikuje datum poslednje izmene dokumenta (onaj koji prijavljuje web server).
<code>linkColor</code>	Niska koja specifikuje boju neposećenih veza dokumenta. Može biti postavljena u zaglavlju HTML dokumenta ili kao vrednost atributa <code>LINK</code> elementa <code><BODY></code> .
<code>links[]</code>	Niz Link objekata, jedan za svaku hipertekstualnu vezu u dokumentu.
<code>links.length</code>	Ceo broj koji može samo da se čita i koji specifikuje broj elemenata u nizu <code>links[]</code> .
<code>location</code>	Sinonim za svojstvo URL. <u>Nije isto</u> što i Objekat <code>Location</code> .
<code>plugins[]</code>	Sinonim za niz <code>embeds[]</code> .
<code>plugins.length</code>	Broj elemenata u nizu <code>plugins[]</code> ili <code>embeds[]</code> .
<code>referrer</code>	Niska koja može samo da se čita i koja specifikuje URL dokumenta koji je sadržao vezu koja je referisala tekući dokument.
<code>title</code>	Niska koja može samo da se čita, a koja specifikuje sadržaj etikete

	<TITLE> dokumenta.
URL	Niska koja može samo da se čita, a koja specifikuje URL dokumenta.
vlinkColor	Niska koja specifikuje boju posećene veze. Može biti postavljena u zaglavlju HTML dokumenta ili kao vrednost atributa VLINK elementa <BODY>.

Metode

clear()	Metod koji briše sadržaj dokumenta.
close()	Metod koji zatvara tok dokumenta koji je otvoren metodom open().
open()	Metod koji otvara tok u koji se može upisivati sadržaj dokumenta.
write()	Metod koji ubacuje specifikovanu nisku ili niske u dokument koji se trenutno parsira ili u tok dokumenta koji otvoren metodom open().
writeln()	Metod koji je identičan sa write() samo što dodaje izlazu karakter za novi red.

Obrada događaja

onload	Poziva se kada je dokument u potpunosti napunjen i specifikuje ga atribut onLoad etikete <BODY>. Iako se on specifikuje kao atribut etikete <BODY> on je svojstvo Window objekta, a ne Document objekta.
onunload	Poziva se kada je dokument povlači i specifikuje ga atribut onUnload etikete <BODY>. Iako se on specifikuje kao atribut etikete <BODY> on je svojstvo Window objekta, a ne Document objekta.

Element objekat— element formulara

Korišćenje

```
form.elements[i]
form.name
```

Svojstva

checked	Bulovska vrednost koja može da se čita i menja i koja ukazuje na stanje elementa formulara koji može da bude “čekiran” i “nečekiran”.
defaultChecked	

Bulovska vrednost koja može samo da se čita i koja specifikuje pretpostavljenu vrednost elementa formulara koji može da bude “čekiran”.

<code>defaultValue</code>	Bulovska vrednost koja može samo da se čita i koja specifikuje pretpostavljenu vrednost elementa formulara. Za elemente koji prikazuju tekst, ovo svojstvo specifikuje početnu vrednost koja se prikazuje i onu koja se prikazuje kada se formular poništava. Ovo svojstvo odgovara HTML atributu <code>VALUE</code> .
<code>form</code>	Referenca koja može samo da se čita objekta <code>Form</code> koji sadrži ovaj element.
<code>length</code>	Za element formulara <code>Select</code> , ovo svojstvo specifikuje broj opcija ili izbora (od kojih je svaki reprezentovan objektom <code>Option</code>) koji su sadržani unutar niza <code>options[]</code> elementa.
<code>name</code>	Niska koja može samo da se čita i koja specifikuje ime elementa. Ovo svojstvo odgovara HTML atributu <code>NAME</code> . Ovo ime može da se koristi za referisanje tog elementa.
<code>options[]</code>	Za element formulara <code>Select</code> , ovaj niz sarži objekte <code>Options</code> koji predstavljaju opcije ili izbore koje prikazuje <code>Select</code> objekat. Broj elemenata ovog niza je specifikovan svojstvom <code>length</code> elementa <code>Select</code> .
<code>selectedIndex</code>	Za element formulara <code>Select</code> , ovaj ceo broj specifikuje koja je od opcija koju nudi <code>Select</code> objekt izabrana. Ovaj ceo broj može da se čita i menja.
<code>type</code>	Niska koja samo može da se čita i koja specifikuje tip elementa formulara
<code>value</code>	Niska koja specifikuje vrednost koja se šalje serveru kao vrednost ovog elementa kada se formular koji ga sadrži šalje.

Metode

<code>blur()</code>	Uklanjanje fokusa sa elementa.
<code>click()</code>	Simuliranje klika mišem na element formulara.
<code>focus()</code>	Fokusiranje elementa.
<code>select()</code>	Za elemente koji prikazuju tekst koji se može menjati, izaberi tekst koji se pojavljuje u elementu.

Obrada događaja

<code>onblur()</code>	Poziva se kada korisnik ukloni fokus sa elementa.
<code>onchange()</code>	Za elemente koji nisu dugmići, poziva se kada korisnik ukuca ili promeni vrednost.
<code>onclick()</code>	Za elemente koji su dugmići, poziva se kada korisnik klikne ili izabere dugmence.

`onfocus()` Poziva se kada korisnik fokusira element.

element	Svojstvo	checked	defaultchecked	defaultValue	form	length	name	options	selectedIndex	type	value	blur()	click()	focus()	select()	onblur()	onchange()	onclick()	onfocus()
button					•		•			•	•	•	•	•		•		•	•
checkbox		•	•		•		•			•	•	•	•	•		•		•	•
radio		•	•		•		•			•	•	•	•	•		•		•	•
reset					•		•			•	•	•	•	•		•		•	•
submit					•		•			•	•	•	•	•		•		•	•
text				•	•		•			•	•	•		•	•	•	•		•
textarea				•	•		•			•	•	•		•	•	•	•		•
password				•	•		•			•	•	•		•	•	•	•		•
fileUpload				•	•		•			•	•	•		•	•	•	•		•
select					•	•	•	•	•	•		•	•	•		•	•		•
hidden					•		•			•	•								

escape() funkcija — kodira nisku za transmisiju

Korišćenje

`escape(s)`

Argumenti

s Niska koju treba kodirati

Povratna vrednost

Kodirana kopija niske *s*.

Napomena

Funkcija `escape()` je ugrađena u JavaScript i nije metod nijednog objekta. `escape()` kreira i vraća novu nisku koja je kodirana verzija niske *s*, na sledeći način: sve praznine, interpunkcijski zanci, akcentovani karakteri i uopšte svi karakteri koji nisu ASCII slova i brojevi konvertuju se u oblik `%xx`, u kome su *xx* dve heksadecimalne cifre koje predstavljaju ISO 8859-1 (Latin 1) kodiranje karaktera. Na primer, `escape("Zdravo svima!")`; vraća nisku `Zdravo%20svima%21`. Namena ove funkcije je da obezbedi prenosivost niske na sve računare i preko svih mreža, bez obzira na karakterski skup koji računar ili mreža podržavaju.

***eval()* funkcija** — izvršava JavaScript kod sadržan u niski

Korišćenje

```
eval(kod)
```

Argumenti

<i>kod</i>	Niska koja sadrži JavaScript izraz koji treba izračunati ili iskaze koje treba izvršiti.
------------	--

Povratna vrednost

Vrednost izvršenog koda, ako postoji.

FileUpload element— ulazno polje za punjenje datoteke formulara

Korišćenje

```
form.elements[i]  
form.ime
```

Svojstva

<code>form</code>	Referenca koja može samo da se čita objekta <code>Form</code> koji sadrži ovo polje za punjenje datoteke.
<code>name</code>	Niska koja može samo da se čita i koju postavlja HTML atribut <code>NAME</code> , a koja specifikuje ime <code>FileUpload</code> elementa.
<code>type</code>	Niska koja može samo da se čita a koja specifikuje tip elementa formulara. Za polje za punjenje datoteke ima vrednost "file".
<code>value</code>	Niska koja može samo da se čita i koja specifikuje vrednost koja je

sadržana u ulaznom polju. To je vrednost koja se šalje serveru prilikom slanja formulara.

Metode

<code>blur()</code>	Uklanjanje fokusa sa ulaznog polja za punjenje datoteke.
<code>focus()</code>	Fokusiranje ulaznog polja za punjenje datoteke.
<code>select()</code>	Isticanje celog teksta u ulaznom polju za punjenje datoteke, čime se ulazi u specijalni režim u kome sledeći unos zamenjuje istaknut tekst.

Obrada događaja

<code>onblur()</code>	Poziva se kada korisnik ukloni fokus sa ulaznog polja za punjenje datoteke.
<code>onchange()</code>	Poziva se kada korisnik promeni vrednost u ulaznom polju za punjenje datoteke. On se ne poziva posle svakog unosa u polje, nego tek onda kada korisnik završi redigovanje.
<code>onfocus()</code>	Poziva se kada korisnik fokusira ulazno polje za punjenje datoteke.

Form Object — ulazni HTML formular

Korišćenje

```
document.ime_formulara
document.forms[broj_formulara]
document.forms.length
```

Svojstva

<code>action</code>	Niska koja može da se čita i menja i koja specifikuje URL na koji treba poslati formular.
<code>elements[]</code>	Niz ulaznih elemenata formulara. Svaki element je objekat Button, Checkbox, Hidden, Password, Radio, Reset, Select, Submit, Text ili Textarea.
<code>elements.length</code>	Broj stavki u nizu <code>elements[]</code> .
<code>encoding</code>	Niska koja može da se čita i menja koja specifikuje metod kodiranja koji se koristi za podatke iz formulara. Na početku se specifikuje atributom ENCTYPE etikete <code><FORM></code> . Pretpostavljena vrednost "application/x-www-form-urlencoded" je skoro uvek adekvatna.
<code>method</code>	Niska koja može da se čita i menja i koja specifikuje tehniku slanja

formulara. Može da ima vrednosti “get” i “post”. Na početku se specifikuje atributom `METHOD` etikete `<FORM>`.

`target` Niska koja može da se čita i menja koja specifikuje ime formulara ili prozora u kome treba prikazati rezultate slanja formulara. Na početku se specifikuje atributom `TARGET` etikete `<FORM>`. Specijalna imena “_top”, “_parent”, “_self” i “_blank” su takođe podržana.

Metode

`reset()` Poništavanje svih ulaznih elemenata formulara i njihovo postavljanje na pretpostavljenu vrednost.

`submit()` Slanje formulara.

Obrada događaja

`onreset()` Poziva se neposredno pre nego što se poništi vrednost svih elemenata formulara. Specifikuje se atributom `onReset` etikete `<FORM>`.

`onsubmit()` Poziva se neposredno pre slanja formulara. Specifikuje se atributom `onSubmit` etikete `<FORM>`. On omogućava proveru vrednosti formulara pre nego što se pošalje serveru.

Frame objekat — vrsta Window objekta

Korišćenje

```
prozor.frames[]
prozor.frames.length
frames[i]
frames.length
```

Napomena

Iako se govori o objektu `Frame`, precizno govoreći takav objekat ne postoji. Svi okviri unutar prozora prelistača su primerci objekta `Window` i imaju ista svojstva i podržavaju iste metode i istu obradu događaja kao objekti `Window`.

Function objekat — JavaScript funkcija

Korišćenje

```
function imefunkcije([imearg1[,...imeargn])
{
    telofunkcije
}
```

Konstruktori

```
new function([imearg1[,...,imeargn]], telofunkcije)
```

Argumenti

imearg0, imearg1, ..., imeargn

Bilo koji broj niski od kojih svaka imenuje jedan argument objekta `Function` koji se kreira.

telofunkcije

Niska koja specifikuje telo funkcije. Može da sadrži proizvoljan broj JavaScript iskaza, razdvojenih tačkom i zarezom.

Povratna vrednost

Novokreirana funkcija.

Svojstva

`arguments[]` Niz argumenata koji se prosleđuju funkciji.
`arguments.length`

Broj stavki u nizu `arguments[]`.

`caller` Referenca objekta `Function` koji je pozvao ovu funkciju, odnosno `null` ako je funkcija pozvana s najvišeg nivoa.

`prototype` Objekat koji za konstruktorske funkcije definiše svojstva i metode koje dele svi objekti kreirani s tom konstruktorskom funkcijom.

Metode

`toString()` Vraća niskovnu reprezentaciju funkcije.

Hidden element— skriveni podaci za komunikaciju između klijenta i servera

Korišćenje

```
form.elements[i]
form.ime
```

Svojstva

form	Referenca koja može samo da se čita objekta Form koji sadrži ovo skriveno polje.
name	Niska koja može samo da se čita i koju postavlja HTML atribut NAME, a koja specifikuje ime Hidden elementa.
type	Niska koja može samo da se čita a koja specifikuje tip elementa formulara. Za skriveno polje ima vrednost "hidden".
value	Niska koja može samo da se čita i koja specifikuje proizvoljne podatke koji se transmituju serveru kad se formular šalje. Ove podatke korisnik nikad ne vidi.

History objekat — URL istorija prelistača

Korišćenje

```
window.history
okvir.history
history
```

Svojstva

Current	Niska koja specifikuje URL tekućeg dokumenta.
Length	Broj URL-a koji su sačuvani u History objektu.
Next	Niska koja specifikuje URL dokumenta koji sledi iza tekućeg dokumenta u listi istorije.
Previous	Niska koja specifikuje URL dokumenta koji prethodi tekućem dokumentu u listi istorije.

Metode

back()	Povratak unazad na prethodno posećeni URL.
forward()	Povratak unapred na prethodno posećeni URL.
go()	Odlazak na prethodno posećeni URL.
toString()	Vraća HTML formatiranu tabelu koja sadrži istoriju prelistača.

Image objekat — slika uključena u HTML dokument

Korišćenje

```
dokument.images[i]
dokument.images.length
dokument.ime-slike
```

Konstruktori

```
new Image([sirina,visina])
```

Argumenti

sirina,visina
Neobavezna specifikacija širine i visine slike.

Povratna vrednost

Novokreirani Image objekat

Svojstva

Border	Ceo broj koji može samo da se čita i koji specifikuje širinu u pikselima okvira oko slike. Njegovu vrednost postavlja atribut BORDER etikete .
complete	Bulovska vrednost koja specifikuje da li je slika u potpunosti napunjena.
height	Ceo broj koji može samo da se čita i koji specifikuje visinu u pikselima slike. Njegovu vrednost postavlja atribut HEIGHT etikete .
hspace	Ceo broj koji može samo da se čita i koji specifikuje u pikselima veličinu dodatnog horizontalnog prostora desno i levo oko slike. Njegovu vrednost postavlja atribut HSPACE etikete .
lowsrc	Niska koja može da se čita i menja a koja specifikuje URL alternativne slike koja je pogodna za prikazivanje na malim rezolucijama. Njegovu vrednost postavlja atribut LOWSRC etikete .
name	Niska koja može samo da se čita a koju specifikuje HTML NAME atribut. Kada je slici dato ime, referenca slike postoji, osim u nizu <i>dokument.images[]</i> , i u <i>dokument.ime-slike</i> .
src	Niska koja može da se čita i menja a koja specifikuje URL slike koja se prikazuje. Njegovu početnu vrednost postavlja atribut SRC etikete .
vspace	Ceo broj koji može samo da se čita i koji specifikuje u pikselima

width veličinu dodatnog vertikalnog prostora iznad i ispod slike. Njegovu vrednost postavlja atribut VSPACE etikete . Ceo broj koji može samo da se čita i koji specifikuje širinu u pikselima slike. Njegovu vrednost postavlja atribut WIDTH etikete .

Obrada događaja

onabort	Poziva se kada korisnik prekine punjenje slike.
onerror	Poziva se ako dođe do greške u toku punjenja slike.
onload	Poziva se kada je punjenje slike uspešno završeno.

Link objekat — hipertekstualna veza

Korišćenje

```
dokument.links[i]
dokument.links.length
```

Svojstva

hash	Niska koja može da se čita i menja i koja specifikuje deo URL-a iz HTML HREF atributa iza znaka #, uključujući i taj znak. Ovaj deo specifikuje ime sidra unutar objekta koji URL referiše.
host	Niska koja može da se čita i menja koja specifikuje kombinaciju imena hosta i porta i URL-a iz HTML HREF atributa.
hostname	Niska koja može da se čita i menja koja specifikuje ime hosta iz URL-a iz HTML HREF atributa.
href	Niska koja može da se čita i menja koja specifikuje kompletan URL iz HTML HREF atributa.
pathname	Niska koja može da se čita i menja koja specifikuje putanju iz URL-a iz HTML HREF atributa.
port	Niska koja može da se čita i menja koja specifikuje port iz URL-a iz HTML HREF atributa.
protocol	Niska koja može da se čita i menja koja specifikuje protokol iz URL-a iz HTML HREF atributa.
search	Niska koja može da se čita i menja koja specifikuje upit iz URL-a iz HTML HREF atributa.
target	Niska koja može da se čita i menja koja specifikuje ime objekta Window (ili okvir) u kome dokument sa URL-a treba prikazati.

Obrada događaja

<code>onclick</code>	Iskazi koji se izvršavaju kada korisnik klikne na vezu. Ako pridružena funkcija vrat vrednost <code>false</code> , veza se neće ni pratiti.
<code>onmouseout</code>	Iskazi koji se izvršavaju kada korisnik pomeri kursor miša sa veze.
<code>onmouseover</code>	Iskazi koji se izvršavaju kada korisnik postavi kursor miša na vezu.

Location objekat — predstavlja lokaciju prelistača i upravlja njome

Korišćenje

```
location  
prozor.location
```

Svojstva

<code>hash</code>	Deo URL-a iz HTML <code>HREF</code> atributa iza znaka <code>#</code> , uključujući i taj znak. Ovaj deo specifikuje ime sidra unutar jedne HTML datoteke.
<code>host</code>	Kombinacija imena hosta i porta iz URL-a.
<code>hostname</code>	Ime hosta iz URL-a.
<code>href</code>	Kompletan URL.
<code>pathname</code>	Putanja iz URL-a.
<code>port</code>	Port iz URL-a.
<code>protocol</code>	Protokol iz URL-a.
<code>search</code>	Upit iz URL-a.

Metode

<code>reload()</code>	Ponovo napuni tekući dokument iz keša ili sa servera.
<code>replace()</code>	Zameni tekući dokument novim, neunoseći novu stavku u istoriju sesije prelistača.

Math objekat — obuhvata matematičke funkcije i konstante

Korišćenje

```
Math.konstanta  
Math.funkcija
```

Konstante

E	Konstanta e , osnova prirodnog logaritma
LN10	prirodni logaritam od 10
LN2	prirodni logaritam od 2
LOG10E	Logaritam od e s osnovi 10
LOG2E	Logaritam od e s osnovom 2
PI	Konstanta π
SQRT1_2	1 kroz kvadratni koren iz 2
SQRT2	kavadtratni koren iz 2

Funkcije

<code>abs()</code>	Računa apsolutnu vrednost
<code>acos()</code>	Računa <i>arc cosinus</i>
<code>asin()</code>	Računa <i>arc sinus</i>
<code>atan()</code>	Računa <i>arc tangens</i>
<code>atan2()</code>	Računa ugao između x ose i tačke
<code>ceil()</code>	Zaokružuje na prvi veći celi broj
<code>cos()</code>	Računa <i>cosinus</i>
<code>exp()</code>	Računa eksponent od e
<code>floor()</code>	Zaokružuje na prvi manji celi broj
<code>log()</code>	Računa prirodni logaritam
<code>max()</code>	Vraća veći od dva broja
<code>min()</code>	Vraća manji od dva broja
<code>pow()</code>	Računa x^y
<code>random()</code>	Računa slučajni broj
<code>round()</code>	Zaokružuje na najbliži ceo broj
<code>sin()</code>	Računa <i>sinus</i>
<code>sqrt()</code>	Računa kvadratni koren
<code>tan()</code>	Računa <i>tangens</i>

MimeType objekat — predstavlja MIME tip podataka

Korišćenje

```
navigator.mimeTypes[i]
navigator.mimeTypes["ime"]
navigator.mimeTypes.length
```

Svojstva

<code>opis</code>	Niska koja može samo da se čita i koja sadrži opis na engleskom sadržaja i kodiranja datog tipa
<code>enabledPlugin</code>	Referenca <code>Plugin</code> objekta koji podržava ovaj MIME tip, ili null ako ga ni jedan instalirani i omogućeni priključak ne podržava.
<code>suffixes</code>	Niska koja može samo da se čita i koja sadrži listu uobičajenih proširenja imena koja su pridružena ovom MIME tipu razdvojenih zarezom
<code>type</code>	Niska koja može samo da se čita koja ukazuje na ime MIME tips, na primer, “text/html” ili “video/mpeg”

Navigator objekat — informacije o prelistaću koji se koristi

Korišćenje

`navigator`

Svojstva

<code>appName</code>	Kod prelistaća
<code>appName</code>	Ime prelistaća
<code>appVersion</code>	Verzija prelistaća
<code>mimeTypes []</code>	Niz objekata <code>MimeType</code> koji opisuje koje Mime tipove prepoznaje i podržava prelistač.
<code>mimeTypes.length</code>	Broj stavki u nizu <code>mimeTypes []</code> .
<code>plugins []</code>	Niz objekata <code>Plugin</code> koji opisuje instalirane priključke.
<code>plugins [].length</code>	Broj stavki u nizu <code>plugins []</code> .
<code>userAgent</code>	Niska koju prelistač prosleđuje u zaglavlju HTTP zahteva.

Metode

<code>JavaEnabled()</code>	Testira da li prelistač podržava i da li je omogućio Javu.
<code>plugins.refresh</code>	Proverava novouključene priključke, ubacuje ih u niz <code>plugins []</code> , i po potrebi ponovo puni dokument koristeći te nove priključke.
<code>taintEnabled()</code>	

Number objekat — numeričke konstante**Korišćenje**

```
Number.konstanta
```

Konstruktori

```
new Number(vrednost)
```

Argumenti

vrednost

Brojčana vrednost Number objekta koji se kreira. Ovaj argument se konvertuje, ako je to potrebno, u broj.

Povratna vrednost

Novokreirani `Number` objekat

Konstante

<code>MAX_VALUE</code>	Najveći broj koji se može predstaviti
<code>MIN_VALUE</code>	Najmanje broj koji se može predstaviti
<code>NaN</code>	Specijalna Not-a-Number vrednost
<code>NEGATIVE_INFINITY</code>	Specijalna vrednost koja označava negativnu beskonačnu vrednost. Vraća se kod prekoračenja.

<code>POSITIVE_INFINITY</code>	Specijalna vrednost koja označava pozitivnu beskonačnu vrednost. Vraća se kod prekoračenja.
--------------------------------	---

Metode

<code>toString()</code>	Konvertuje broj u nisku korišćenjem specifikovane osnove
<code>valueOf()</code>	Vraća primitivnu numeričku vrednost koju sadrži <code>Number</code> objekat.

Option element — opcija u kućici za izbor

Korišćenje

```
form.options[i]
```

Svojstva

<code>defaultSelected</code>	Bulovska vrednost koja može samo da se čita i koja specifikuje da je ova opcija unapred selektovana.
<code>index</code>	Ceo broj koji može samo da se čita i koji specifikuje indeks ove opcije unutar niza opcija.
<code>selected</code>	Bulovska vrednost koja specifikuje da li je ova opcija trenutno selektovana. Početnu vrednost specifikuje HTML atribut <code>SELECTED</code> .
<code>text</code>	Tekst koji opisuje opciju. To je običan, a ne HTML, tekst koji sledi iza etikete <code><OPTION></code> .
<code>value</code>	Niska koja može da se čita i da se menja a koje specifikuje vrednost koja se prosleđuje serveru ako je ova opcija selektovana u trenutku slanja formulara. Početnu vrednost specifikuje HTML atribut <code>VALUE</code> .

Konstruktor

```
new Option([tekst, [vred,  
[unapredIzabran, [izabran]]]])
```

Argumenti

<i>tekst</i>	Neobavezni niskovni argument koji specifikuje <code>text</code> svojstvo <code>Option</code> objekta.
<i>Vred</i>	Neobavezni niskovni argument koji specifikuje <code>value</code> svojstvo objekta <code>Option</code> .
<i>unapredIzabran</i>	Neobavezan Bulovski argument koji specifikuje <code>defaultSelected</code> svojstvo <code>Option</code> objekta.
<i>izabran</i>	Neobavezan Bulovski argument koji specifikuje <code>selected</code> svojstvo <code>Option</code> objekta.

Povratna vrednost

Novokreirani `Option` objekat.

***parseFloat()* funkcija** — konvertuje nisku u broj

Korišćenje

```
parseFloat (s)
```

Argumenti

s Niska koju treba parsirati i konvertovati u broj

Povratna vrednost

Parsirani broj ili NaN ako *s* ne počinje valjanim brojem.

***parseInt()* funkcija** — konvertuje nisku u celi broj

Korišćenje

```
parseInt (s)
parseInt (s, osnova)
```

Argumenti

s Niska koju treba parsirati i konvertovati u broj
osnova Neobavezan celobrojni argument koji predstavlja osnovu broja koji se
 parsira

Povratna vrednost

Parsirani broj ili NaN ako *s* ne počinje valjanim celim brojem.

Password element— ulazno polje za osetljive podatke

Korišćenje

```
form.elements[i]
form.ime
```


Svojstva

<code>defaultValue</code>	Niska koja može samo da se čita koja specifikuje početnu vrednost koja se pojavljuje u polju lozinke. Ovo svojstvo se specifikuje atributom <code>VALUE</code> HTML etikete koja kreira Password element.
<code>form</code>	Referenca koja može samo da se čita objekta Form koji sadrži ovo polje lozinke.
<code>name</code>	Niska koja može samo da se čita, koju postavlja HTML atribut <code>NAME</code> , a koja specifikuje ime polja lozinke.
<code>type</code>	Niska koja može samo da se čita a koja specifikuje tip elementa formulara. Za dugme ima vrednost “password”.
<code>value</code>	Niska koja može samo da se čita a koja specifikuje vrednost lozinke koju unosi korisnik.

Metode

<code>blur()</code>	Uklanjanje fokusa sa polja lozinke.
<code>focus()</code>	Fokusiranje polja lozinke.
<code>select()</code>	Isticanje celog teksta elementa Password i ulazak u poseban režim u kome prvi otkucani karakter zamenjuje istaknuti tekst.

Obrada događaja

<code>onblur()</code>	Poziva se kada korisnik ukloni fokus sa polja lozinke.
<code>onchange()</code>	Poziva se kada se kada korisnik promeni vrednost Password elementa i ukloni fokus s njega. Ne poziva se posle svakog otkucanog karaktera, već tek onda kada korisnik završi sa redigovanjem polja.
<code>onfocus()</code>	Poziva se kada se polje lozinke fokusira.

Plugin objekat — opisuje instalirani priključak

Korišćenje

```
navigator.plugins[i]
navigator.plugins['ime']
```

Svojstva

<code>description</code>	Niska koja može samo da se čita a koja sadrži opis priključka koji je namenjen čitaocu. Ovo svojstvo može da specifikuje puno ime
--------------------------	---

	proizvoda, informaciju o prodavcu, verziju, itd.
<code>filename</code>	Niska koja može samo da se čita a koja specifikuje ime datoteke na disku koja sadrži programski kod priključka.
<code>length</code>	Broj MIME tipova koje priključak podržava. <code>MimeType</code> objekti koji opisuju ove tipove su elementi niza objekata tipa <code>Plugin</code> .
<code>name</code>	Niska koja može samo da se čita a koja specifikuje ime priključka. To je obično mnogo kraća niska od <code>description</code> . Vrednost ovog svojstva može se koristiti kao indeks niza <code>navigator.plugins[]</code> .

Radio element— grafičko radio dugme

Korišćenje

```
form.radio_ime[i]
form.radio_ime.length
form.elements[i][j]
form.elements[i].length
```

Svojstva

<code>checked</code>	Bulovska vrednost koja može da se čita i da se menja i koja specifikuje da li je dugme čekirano ili ne.
<code>defaultChecked</code>	Bulovska vrednost koja može samo da se čita i koja specifikuje početni status radio dugmeta. Može se specifikovati HTML atributom <code>CHECKED</code> .
<code>form</code>	Referenca koja može samo da se čita objekta <code>Form</code> koji sadrži ovo radio dugme.
<code>name</code>	Niska koja može samo da se čita, koju postavlja HTML atribut <code>NAME</code> , a koja specifikuje ime radio dugmeta.
<code>type</code>	Niska koja može samo da se čita a koja specifikuje tip elementa formulara. Za dugme ima vrednost "radio".
<code>value</code>	Niska koja može da se čita i da se menja a koja specifikuje vrednost koju vraća radio dugme ako je selektovano u trenutku slanja formulara. Početnu vrednost mu postavlja HTML atribut <code>VALUE</code> .

Metode

<code>blur()</code>	Uklanjanje fokusa sa radio dugmeta
<code>click()</code>	Simulira klik mišem na radio dugme.
<code>focus()</code>	Fokusiranje radio dugmeta.

Obrada događaja

<code>onblur()</code>	Poziva se kada korisnik ukloni fokus sa radio dugmeta.
<code>onclick()</code>	Poziva se kada se klikne na radio dugme.
<code>onfocus()</code>	Poziva se kada se radio dugme fokusira.

Reset element—dugme koje poništava vrednosti u formularu

Korišćenje

`form.ime`
`form.elements[i]`

Svojstva

<code>form</code>	Referenca koja može samo da se čita objekta Form koji sadrži ovo dugme za poništavanje.
<code>name</code>	Niska koja može samo da se čita, koju postavlja HTML atribut NAME, a koja specifikuje ime dugmeta za poništavanje.
<code>type</code>	Niska koja može samo da se čita a koja specifikuje tip elementa formulara. Za dugme ima vrednost “reset”.
<code>value</code>	Niska koja može samo da se čita a koja specifikuje tekst koji se pojavljuje na dugmetu. Vrednost mu postavlja HTML atribut VALUE.

Metode

<code>blur()</code>	Uklanjanje fokusa sa dugmeta za poništavanje.
<code>click()</code>	Simulira klik mišem na dugme za poništavanje.
<code>focus()</code>	Fokusiranje dugme za poništavanje.

Obrada događaja

<code>onblur()</code>	Poziva se kada korisnik ukloni fokus sa dugmeta za poništavanje.
<code>onclick()</code>	Poziva se kada se klikne na dugme za poništavanje
<code>onfocus()</code>	Poziva se kada se dugme za poništavanje fokusira.

Select element—grafička lista izbora

Korišćenje

`form.ime`
`form.elements[i]`

Svojstva

<code>form</code>	Referenca koja može samo da se čita objekta <code>Form</code> koji sadrži ovu listu izbora.
<code>length</code>	Ceo broj koji specifikuje broj elemenata u nizu <code>options []</code> .
<code>name</code>	Niska koja može samo da se čita, koju postavlja HTML atribut <code>NAME</code> , a koja specifikuje ime liste izbora.
<code>selectedIndex</code>	Ceo broj koji može samo da se čita i koja specifikuje indeks selektovane opcije iz ove liste izbora. Ako ova lista izbora ima postavljen atribut <code>MULTIPLE</code> i dozvoljava višestruki izbor onda ovo svojstvo specifikuje samo indeks prve selektovane stavke (ili <code>-1</code> ako ni jedna nije selektovana).
<code>type</code>	Niska koja može samo da se čita a koja specifikuje tip elementa formulara. Za dugme ima vrednost <code>"select-one"</code> ili <code>"select-multiple"</code> .

Metode

<code>blur()</code>	Uklanjanje fokusa sa liste izbora.
<code>click()</code>	Simulira klik mišem na listu izbora.
<code>focus()</code>	Fokusiranje liste izbora.

Obrada događaja

<code>onblur()</code>	Poziva se kada korisnik ukloni fokus sa liste izbora.
<code>onclick()</code>	Poziva se kada se klikne na listu izbora.
<code>onfocus()</code>	Poziva se kada se lista izbora fokusira.

String objekat — podrška niskama**Konstruktor**

```
new String(vrednost)
```

Argumenti

vrednost

Početna vrednost `String` objekta koji se kreira. Ovaj argument će biti pretvoren u nisku ako je to potrebno.

Povratna vrednost

Novokreirani `String` objekat.

Svojstva

`length` Broj karaktera u nisci.

Metode

<code>anchor()</code>	Vraća kopiju niske u okruženju etiketa <code></code> .
<code>big()</code>	Vraća kopiju niske u okruženju etiketa <code><BIG></code> .
<code>blink()</code>	Vraća kopiju niske u okruženju etiketa <code><BLINK></code> .
<code>bold()</code>	Vraća kopiju niske u okruženju etiketa <code></code> .
<code>charAt()</code>	Vadi karakter sa date pozicije u nisci.
<code>fixed()</code>	Vraća kopiju niske u okruženju etiketa <code><TT></code> .
<code>fontcolor()</code>	Vraća kopiju niske u okruženju etiketa <code></code> .
<code>fontsize()</code>	Vraća kopiju niske u okruženju etiketa <code></code> .
<code>indexOf()</code>	Traži karakter ili podnisku u nisci.
<code>italics()</code>	Vraća kopiju niske u okruženju etiketa <code><I></code> .
<code>lastIndexOf()</code>	Traži unazad nisku ili podnisku u nisci.
<code>link()</code>	Vraća kopiju niske u okruženju etiketa <code></code> .
<code>small()</code>	Vraća kopiju niske u okruženju etiketa <code><SMALL></code> .
<code>split()</code>	Konvertuje nisku u niz niski, koristeći neki karakter kao graničnik.
<code>sub()</code>	Vraća kopiju niske u okruženju etiketa <code><SUB></code> .
<code>substring()</code>	Vadi podnisku iz niske.
<code>sup()</code>	Vraća kopiju niske u okruženju etiketa <code><SUP></code> .
<code>toLowerCase()</code>	Vraća kopiju niske u kojoj su sva slova pretvorena u mala slova.
<code>toUpperCase()</code>	Vraća kopiju niske u kojoj su sva slova pretvorena u velika slova.

Submit element—dugme za slanje formulara

Korišćenje

```
form.ime
form.elements[i]
form.elements['ime']
```

Svojstva

`form` Referenca koja može samo da se čita objekta `Form` koji sadrži ovo dugme za slanje formulara.

<code>name</code>	Niska koja može samo da se čita, koju postavlja HTML atribut <code>NAME</code> , a koja specifikuje ime dugmeta za slanje.
<code>type</code>	Niska koja može samo da se čita a koja specifikuje tip elementa formulara. Za dugme ima vrednost <code>“submit”</code> .

Metode

<code>blur()</code>	Uklanjanje fokusa sa dugmeta za slanje.
<code>click()</code>	Simulira klik mišem na dugme za slanje.
<code>focus()</code>	Fokusiranje dugmeta za slanje.

Obrada događaja

<code>onblur()</code>	Poziva se kada korisnik ukloni fokus sa dugmeta za slanje.
<code>onclick()</code>	Poziva se kada se klikne na dugme za slanje.
<code>onfocus()</code>	Poziva se kada se dugme za slanje fokusira.

Text element— grafičko tekstualno ulazno polje

Korišćenje

`form.ime`
`form.elements[i]`

Svojstva

<code>defaultValue</code>	Niska koja može samo da se čita i koja specifikuje početnu vrednost ulaznog polja. Specifikuje se HTML atributom <code>VALUE</code> etikete <code><INPUT></code> .
<code>Form</code>	Referenca koja može samo da se čita objekta <code>Form</code> koji sadrži ovo ulazno polje.
<code>Name</code>	Niska koja može samo da se čita, koju postavlja HTML atribut <code>NAME</code> , a koja specifikuje ime ulaznog polja.
<code>Type</code>	Niska koja može samo da se čita a koja specifikuje tip elementa formulara. Za dugme ima vrednost <code>“text”</code> .
<code>Value</code>	Niska koja može da se čita i da se menja a koja specifikuje vrednost koja je sadržana u ulaznom polju, a to je i vrednost koje se šalje serveru prilikom slanja formulara. Početnu vrednost mu postavlja HTML atribut <code>VALUE</code> .

Metode

<code>blur()</code>	Uklanja fokus sa ulaznog polja.
<code>focus()</code>	Fokusira ulazno polje. Kada je fokus postavlje, sve što se kuca automatski se unosi u ovo polje.
<code>select()</code>	Isticanje svog teksta u ulaznom polju i ulazak u specijalni režim u kome sve što se bude unosilo zamenjuje istaknuti tekst.

Obrada događaja

<code>onblur()</code>	Poziva se kada korisnik ukloni fokus sa ulaznog polja.
<code>onchange()</code>	Poziva se kada korisnik promeni sadržaj ulaznog polja i ukloni fokus sa njega. Ne poziva se posle svakog unosa u polje već tek onda kada je korisnik završio sa redigovanjem sadržaja polja.
<code>onfocus()</code>	Poziva se kada se ulazno polje fokusira.

Textarea element — višelinijnsko tekstualno ulazno polje

Korišćenje

```
form.ime
form.elements[i]
```

Svojstva

<code>defaultValue</code>	Niska koja može samo da se čita i koja specifikuje početnu vrednost ulaznog polja. Ova početna vrednost je običan tekst koji se pojavljuje između etiketa <code><TEXTAREA></code> etikete <code></TEXTAREA></code> .
<code>form</code>	Referenca koja može samo da se čita objekta Form koji sadrži ovo ulazno polje.
<code>name</code>	Niska koja može samo da se čita, koju postavlja HTML atribut NAME, a koja specifikuje ime ulaznog polja.
<code>type</code>	Niska koja može samo da se čita a koja specifikuje tip elementa formulara. Za dugme ima vrednost "textarea".
<code>value</code>	Niska koja može da se čita i da se menja a koja specifikuje vrednost koja je sadržana u ulaznom polju, a to je i vrednost koje se šalje serveru prilikom slanja formulara. Početna vrednost ovog svojstva je ista sa vrednošću svojstva <code>defaultValue</code> .

Metode

<code>blur()</code>	Uklanja fokus sa ulaznog polja.
<code>focus()</code>	Fokusira ulazno polje. Kada je fokus postavlje, sve što se kuca automatski se unosi u ovo polje.
<code>select()</code>	Isticanje svog teksta u ulaznom polju i ulazak u specijalni režim u kome sve što se bude unosilo zamenjuje istaknuti tekst.

Obrada događaja

<code>onblur()</code>	Poziva se kada korisnik ukloni fokus sa ulaznog polja.
<code>onchange()</code>	Poziva se kada korisnik promeni sadržaj ulaznog polja i ukloni fokus sa njega. Ne poziva se posle svakog unosa u polje već tek onda kada je korisnik završio sa redigovanjem sadržaja polja.
<code>onfocus()</code>	Poziva se kada se ulazno polje fokusira.

***unescape()* Function** — dekodira nisku kodiranu za transmisiju

Korišćenje

```
unescape(s)
```

Argumenti

s Niska koju treba dekodirati

Povratna vrednost

Dekodirana kopija niske *s*.

Napomena

Funkcija `escape()` je ugrađena u JavaScript i nije metod nijednog objekta. `unescape()` dekodira nisku koja je kodirana sa `escape()`. Ona kreira i vraća dekodiranu kopiju od *s*. Ona dekodira nisku tako što pronalazi i zamenjuje karakterske sekvencije oblika `%xx`, u kojima su *xx* dve heksadecimalne cifre. Svaka od ovakvih sekvencija zamenjuje se jednim karakterom koji predstvaljaju heksadecimalne cifre u 8859-1 (Latin 1) kodiranju karaktera.

URL objekat — Univerzalni lokator resursa

Korišćenje

```
prozor.location  
dokument.links[i]
```


Svojstva

hash	Niska koja može da se čita i menja i koja specifikuje deo URL-a iz HTML HREF atributa iza znaka #, uključujući i taj znak. Ovaj deo specifikuje ime sidra unutar objekta koji URL referiše.
host	Niska koja može da se čita i menja koja specifikuje kombinaciju imena hosta i porta i URL-a iz HTML HREF atributa.
hostname	Niska koja može da se čita i menja koja specifikuje ime hosta iz URL-a iz HTML HREF atributa.
href	Niska koja može da se čita i menja koja specifikuje kompletan URL iz HTML HREF atributa.
pathname	Niska koja može da se čita i menja koja specifikuje putanju iz URL-a iz HTML HREF atributa.
port	Niska koja može da se čita i menja koja specifikuje port iz URL-a iz HTML HREF atributa.
protocol	Niska koja može da se čita i menja koja specifikuje protokol iz URL-a iz HTML HREF atributa.
search	Niska koja može da se čita i menja koja specifikuje upit iz URL-a iz HTML HREF atributa.

Window objekat — prozor ili okvir web-prelistača

Korišćenje

self	(tekući prozor)
window	(tekući prozor)

Svojstva

closed	Bulovska vrednost koja može samo da se čita i koja specifikuje da li je prozor zatvoren.
defaultStatus	Niska koja može da se čita i u koju može da se piše i koja specifikuje pretpostavljenu vrednost koja se pojavljuje u statusnoj liniji.
document	Referenca objekta Document koji je sadržan u prozoru.
frames[]	Niz okvira koji su sadržani u prozoru.
history	Referenca objekta History za ovaj prozor.
java	Referenca JavaPackage objekta koji je na vrhu hijerarhije imena paketa za jezgro <i>java*</i> paketa koji čine Java jezik.
length	Broj elemenata u nizu <code>frames[]</code> . Isto što i <code>frames.length</code> . Može samo da se čita.
location	Referenca objekta Location za ovaj prozor.
Math	Referenca objekta koji čuva različite matematičke funkcije i

	konstante.
<code>name</code>	Niska koja sadrži ime prozora. Ime se, po želji, specifikuje prilikom otvaranja prozora uz pomoć metode <code>open()</code> .
<code>navigator</code>	Referenca objekta <code>Navigator</code> koji se primenjuje na ovaj i sve ostale prozore.
<code>netscape</code>	Referenca <code>JavaPackage</code> objekta koji je na vrhu hijerarhije imena paketa za jezgro <i>netscape</i> * paketa od Netscape-a.
<code>opener</code>	Svojstvo koje može da se čita i u koje može da se piše a koje referiše objekat <code>Window</code> koji je pozvao <code>open()</code> metod za kreiranje ovog prozora.
<code>Packages</code>	Referenca <code>JavaPackage</code> objekta koji je na vrhu hijerarhije imena paketa.
<code>parent</code>	Referenca roditeljskog prozora ili okvira tekućeg prozora.
<code>self</code>	Referenca samog prozora. Sinonim za <code>window</code> .
<code>status</code>	Niska koja može da se čita i u koju može da se piše i koja specifikuje tekući sadržaj statusne linije.
<code>sun</code>	Referenca <code>JavaPackage</code> objekta koji je na vrhu hijerarhije imena paketa za jezgro <i>sun</i> * paketa koji čine Java jezik.
<code>top</code>	Referenca prozora na najvišem nivou koji sadrži tekući prozor. Korisno samo ako je tekući prozor okvir.
<code>window</code>	Referenca samog prozora. Sinonim za <code>self</code> .

Metode

<code>alert()</code>	Prikazuje jednostavnu poruku i dijaloškom prozoru.
<code>blur()</code>	Pomera fokus sa prozora najvišeg nivoa prelistača.
<code>clearTimeout()</code>	Prekida operaciju koja treba da se izvrši po isteku nekog vremena.
<code>close()</code>	Zatvara prozor.
<code>confirm()</code>	Postavlja u dijaloškom prozora pitanje koje zahteva odgovor “da” ili “ne”.
<code>focus()</code>	Fokusira prozor koji je na najvišem nivou prelistača.
<code>open()</code>	Kreira i otvara novi prozor.
<code>prompt()</code>	Traži da unese jednostavna niska u dijaloškom prozoru.
<code>scroll()</code>	Proizvodi klizanje dokumenta koji je prikazan u prozoru.
<code>setTimeout()</code>	Izvršava kod posle isteka nekog specifikovanog vremenskog intervala.

Obrada događaja

<code>onblur()</code>	Poziva se kada prozor više nije u fokusu.
<code>onerror()</code>	Poziva se kada dođe do greške u izvršavanju JavaScript koda.
<code>onfocus()</code>	Poziva se kada se prozor fokusira.

`onLoad()` Poziva se kada je dokument ili skup okvira potpuno napunjen.
`onUnload()`