Veb programiranje
JavaScript

Filip Marić
Vesna Marinković
Milan Banković
Jelena Graovac

Istorijat

- Nastao sredinom devedesetih u okviru Netscape Navigator-a, u cilju omogućavanja manipulacije sadržajem veb stranica na klijentskoj strani (1996)
- Rat pregledača: Microsoft iste godine uvodi sopstveni jezik JScript u okviru IE
- Standardizacija: ECMAScript (1997)

Osnovne karakteristike

- C-olik jezik (sintaksa liči na C, Javu, ...)
- Jezik se interpretira (tj. ne prevodi se na mašinski jezik)
- Interpretator je najčešće veb pregledač, ali to ne mora biti slučaj (npr. Node.js)
- JavaScript kod se u potpunosti odvija u okviru procesa Veb pregledača
- Dinamički tipiziran jezik (promenljiva može da menja tip tokom svog životnog veka)
- Razlikuje velika i mala slova
- Kombinuje proceduralnu i objektno-orijentisanu programsku paradigmu
- Objektno zasnovan (ali nije klasno zasnovan, poput Jave ili C++-a)
- Podrška za funkcionalno programiranje

Tipovi podataka i vrste operatora

- Osnovni tipovi podataka:
  - number: brojevni tip (jedan tip za celobrojne i realne brojeve),
  - string: niske karaktera
  - boolean: logički tip (vrednosti `true` i `false`)
  - object: objektni tip (svi objekti su ovog tipa, nema klasa)
  - function: funkcijski tip (podtip objektnog tipa, tzv. funkcijski objekat)
  - undefined: koristi se za neinicijalizovane promenljive
- Podržani operatori:
  - aritmetički operatori: $+$, -, *, / (ne postoji celobrojno deljenje), %, $++$, $--$ (prefiksni i postfiksni)
  - relacioni operatori: $<$, $<=$, $>$, $>=$, $!=$, $==$ (jednaki po vrednosti), $===$ (jednaki i po vrednosti i po tipu, nisu dozvoljene implicitne konverzije)
  - logički operatori: &&, ||, !
  - bitski operatori: &, |, $\sim$, ^, $<<$, $>>$, $>>>$ (upražnjena mesta se uvek popunjavaju nulama)
  - operatori dodele: $=$, $+=$, $-=$, ...
  - Ispitivanje tipa: typeof

Promenljive, naredbe, kontrolne strukture

- Deklaracija promenljive je oblika: `var ime_prom;`
- Domet promenljive: globalni domet (podrazumevano) i domet funkcije
- JavaScript je dinamički tipiziran jezik – jedna ista promenljiva može imati različit tip tokom izvršavanja programa
- Grananje je oblika:
  `if (uslov) blok naredbi [else blok naredbi]`
- Podržan je i operator ? :
- Višestruko grananje je oblika:
  `switch (vrednost) {case vr1: blok naredbi; break; ...}`
- Postoji podrška za `for` petlju, `while` petlju i `do while` petlju (sintaksa je C-ovska)
- Iteracija kroz objekte: `for (x in obj) { ... }`

Funkcije

- Definicija funkcije je oblika:
  `function ime_fje(lista parametara){ naredbe }`
- U promenljivoj je moguće sačuvati funkcijski izraz; na ovaj način dobijamo anonimne funkcije (ne navodi im se ime)
- Moguće je koristiti notaciju lambda izraza za zapis funkcije:
  `var f = (x=>3*x)`
- Funkcija može biti parametar druge funkcije:
  `function funkcijaFunkcije(f,y) { return f(y); }`
- Kao argument funkcije može se zadati anonimna funkcija:
  `console.log(funkcijaNiza(x=>3*x,[1,2,3,4,5]))`

Funkcije – zatvorenje

- Funkcije mogu biti ugnježdene: tada je iz unutrašnje funkcije moguće pristupiti lokalnim promenljivim spoljašnje funkcije

```
function a (){
  var x = "Ovo je primer zatvorenja";
  function b (){
    console.log (x);
  }
  b();
}
a(); Vraća: Ovo je primer zatvorenja
```

Funkcije – zatvorenje

- Funkcija nakon vraćanja vrednosti i dalje ima pristup lokalnim promenljivim funkcije koja ju je vratila – ova pojava naziva se zatvorenje.

```
function a () {
  var x = "Ovo je primer zatvorenja čak i nakon return naredbe";
  function b () {
      console.log(x);
  }
  return b;
}
var c = a();  //Dodeljujemo return funkciju promenjivoj "c"
c(); Vraća: Ovo je primer zatvorenja čak i nakon return naredbe
```

Objekti

- Objekti se sastoje od svojstava koje mogu biti različitog tipa
- U Javascript-u, objekat je u stvari kolekcija podataka kojima se pristupa pomoću naziva koji mogu biti proizvoljni stringovi
- Promenljivoj se može dodeliti objekat na sledeći način:

```
var student = {
    ime: "Ana",
    prezime: "Petrović",
    brojIndeksa: "56/2015",
    prosek: 9.21
};
```

- Svojstvima se pristupa na jedan od dva načina:
  - `nazivObjekta.nazivSvojstva`
  - `nazivObjekta["nazivSvojstva"]`
- Svojstva se mogu dinamički dodavati i uklanjati:

```
student.godina = 4;
delete student.prosek;
```

- Metoda se zadaje kao svojstvo koje sadrži definiciju funkcije

```
punoImeIPrezime: function(){
  return this.ime + " " + this.prezime;
}
```

- this unutar objekta označava sâm taj objekat

Objekti

```
  var KetiMaca = {
      broj_godina: 10,
      bojaKrzna: 'bela',
      mjauci: function() {
        console.log('mmmnjaaauuuuu');
      },
      jedi: function(hrana) {
        console.log('Njam, ja volim ' + hrana);
      },
      spavaj: function(brojMinuta) {
        for (var i = 0; i < brojMinuta; i++) {
          console.log('zzz');
        }
      }
};

KetiMaca.mjauci();
KetiMaca.jedi('meso');
KetiMaca.spavaj(10);
```

Konstruktori i prototipovi

- U Javascript-u ne postoji koncept klase, ali je ipak moguće kreirati jednoobrazne objekte
- Konstruktor: funkcija koja kreira objekte

```
function Point(x,y)
{
  this.x = x;
  this.y = y;
  this.norm = function () { return Math.sqrt(this.x*this.x + this.y*this.y); }
}
var p = new Point(1, 2);
var q = new Point(5, 1);
console.log(p.norm());
```

- keys() vraća svojstva nekog objekta

```
Object.keys(p) // ["x", "y", "norm"]
```

- Problem: svaki objekat ima svoju kopiju svih članova, pa i funkcija (koje su objekti)
- Rešenje – prototipovi

Prototipovi

- Pogledajmo sledeći primer:

```
function Student() {
    this.ime = 'Marko';
    this.pol = 'Muskarac';
}

var studObj1 = new Student();
studObj1.godine = 20;
alert(studObj1.godine); // 20

var studObj2 = new Student();
alert(studObj2.godine); // undefined
```

Primetimo da je svojstvo godine dodato objektu studObj1, ali ne i studObj2

Prototipovi

- Ovakav primer se rešava prototipovima:

```
function Student() {
    this.ime = 'Marko';
    this.pol = 'Muskarac';
}

Student.prototype.godine = 20;

var studObj1 = new Student();
alert(studObj1.godine); // 20

var studObj2 = new Student();
alert(studObj2.godine); // 20
```

Rad sa stringovima

- String se može zadati pod dvostrukim ili jednostrukim navodnicima
  var ime='Marko', prezime="Petrovic";
- s.length – vraća dužinu niske *s*
- s.charAt(i) – vraća karakter na *i*-toj poziciji u stringu *s* (indeksi idu od 0)
- s1.concat(s2) – nadovezuje nisku *s*2 na nisku *s*1,
  može se koristiti i s1 + s2
- s1.indexOf(s2) ili s1.indexOf(s2,k) – prvo pojavljivanje
  niske *s*2 u niski *s*1 (počev od pozicije *k*)
- s1.lastIndexOf(s2) ili s1.lastIndexOf(s2,k) – poslednje pojavljivanje
  niske *s*2 u niski *s*1 (počev od pozicije *k*)
- s.substr(k) ili s.substr(k,l) – izdvaja podnisku niske *s* počev od
  pozicije *k* do kraja (odnosno do pozicije *l*)

Rad sa stringovima

- s.toLowerCase() i s.toUpperCase() – niska *s* se prevodi u mala, odnosno velika slova
- s.split(c) ili s.split(c,k) – vrši se podela niske *s* u odnosu na separator *c* i vraćaju se sve (ili najviše *k*) dobijenih podniski
- s.charCodeAt(i) – vraća UNICODE kôd datog karaktera
- String.fromCharCode(c) – vraća karakter na osnovu datog UNICODE kôda *c*
- s.substring(i,j) i s.slice(i,j) – izdvaja deo niske *s* između pozicije *i* i pozicije *j* (ako *j* nije navedeno izdvaja se do kraja)
  mogu se koristiti i negativne vrednosti indeksa, tada se broji od kraja niske (ali se ne može koristiti 0)

Rad sa stringovima – specijalni karakteri

- Za zadavanje rezervisanih karaktera koristi se prekidački simbol '\':
  - \' – apostrof
  - \" – dvostruki navodnici
  - \\– obrnuta kosa crta
- Linija kôda koja sadrži string može se prelomiti preko reda korišćenjem znaka '\' ili konkatenacijom stringova (bolji način)
- U stringovima se mogu navoditi sekvence '\n', '\t', i sl.

Rad sa matematičkim funkcijama

- Ugrađeni objekat Math
- Postoji podrška za osam matematičkih konstanti:
  E, PI, LN2, LN10, LOG2E, LOG10E, SQRT2, SQRT1_2
- Postoji podrška za matematičke funkcije:
  - abs(x) – apsolutna vrednost
  - sin(x), cos(x), tan(x) – trigonometrijske funkcije
  - asin(x), acos(x), atan(x) – inverzne trigonometrijske funkcije
  - round(x), ceil(x), floor(x) – zaokruživanje na najbliži, prvi veći i prvi manji ceo broj
  - exp(x), log(x), pow(x) – eksponencijalna, logaritamska i stepena funkcija
  - sqrt(x) – kvadratni koren broja
  - min(x1,...,xn), max(x1,...,xn) – najmanji i najveći od nekoliko brojeva
  - random() – slučajan broj iz intervala [0,1]

Nizovi

- Nizovi se kreiraju konstruktorom Array (`var a = new Array(1, 2, 3);`), ili uglastim zagradama (`var a = [1, 2, 3];`)
- Svojstvo length sadrži broj elemenata niza
- Pristup elementima: `a[i]`
- Metode za rad sa nizovima:
  - a1.concat(a2,...,an) – na niz $a1$ nadovezuju se nizovi $a2, \ldots, an$
  - a.join(c) – od niza a kreira nisku koja se sastoji od elemenata niza a međusobno razdvojenih karakterom c
  - a.push(x) – umeće element x na kraj niza a
  - a.pop() – uklanja element sa kraja niza a
  - a.unshift(x) – umeće element x na početak niza a
  - a.shift() – uklanja element sa početka niza a
  - a.reverse() – obrće elemente niza
  - a.splice(i,br,e1,...,en) – uklanja br elemenata niza a počev od pozicije i a zatim umeće elemente $e1, \ldots, en$ na mestu uklonjenih elemenata
  - a.slice(poc,kraj) – izdvaja deo niza između indeksa *poc* i *kraj* (ili do kraja niza ako nije dat indeks *kraj*)

Nizovi

- Prilikom dodele `a = b` vrši se plitko kopiranje nizova – ove dve promenljive predstavljaju referencu na isti objekat
- Ako želimo da se iskopira samo vrednost možemo uraditi `a = b.slice();`
- Metod s.map(f) – pravi se novi niz čije se vrednosti dobijaju kada se na članove niza *s* primeni funkcija *f*: može biti ugrađena funkcija (Math.sqrt) ili korisnički definisana
- Metod s.sort() – sortira elemente niza *s* rastuće, razmatrajući elemente niza kao stringove; može se proslediti funkcija poređenja s.sort(f)

Datum i vreme

- Vremenske odrednice se mogu predstaviti objektom tipa Date
- Datum se konstruiše pozivom `new Date()` – ako se ne prosledi parametar, pravi se objekat koji predstavlja trenutno vreme na klijentskoj mašini
- Vremenske odrednice se mogu predstaviti i brojem milisekundi koje su protekle od ponoći 1.1.1970.
- Postoji konstruktor sa 7 parametara: godina, mesec, dan, sat, minut, sekund, milisekund; ako se neki parametar izostavi podrazumeva se 0
- Prilikom konstruisanja datuma, može se zadati i niska
- Moguće je porediti dva datuma operatorima $<$ i $>$
- Metode za izdvajanje komponenti: getFullYear(), getMonth(), getDate(), getHours(), getMinutes(), getSeconds(), getTime(),...
- Metode za postavljanje komponenti: setFullYear(), setMonth(), setDate(), setHours(), setMinutes(), setSeconds(), setTime(),...

JSON

- JSON (*JavaScript Object Notation*) je format za čuvanje i transport podataka
- Pravila:
  - podaci se čuvaju u obliku parova "naziv" : "vrednost" – ne mogu se koristiti jednostruki navodnici
  - vrednosti brojevnog i logičkog tipa se navode bez navodnika
  - podaci se međusobno razdvajaju zapetom
  - za čuvanje objekata koriste se vitičaste zagrade
  - za čuvanje nizova koriste se uglaste zagrade
- Ako je u promenljivoj `tekst` smešten JavaScript string zapisan u JSON sintaksi, onda se on može konvertovati u JavaScript objekat korišćenjem funkcije JSON.parse()
- Obratno, ako želimo da dati objekat konvertujemo u JSON string, možemo koristiti funkciju JSON.stringify()