Izrazi in operatorji

Kaj je izraz?

- sestavljen iz:
  - vrednosti,
  - spremenljivk in
  - operatorjev.
- Vrača določeno vrednost
- Povezani med seboj povezani na predpisane načine
- Izraz izračunamo in dobimo neko vrednost (vrednost izraza)
  - tip vrednosti je odvisen od tipov posameznih operandov in uporabljenih operatorjev

Tipi izrazov

- Izraz `3 + 4` ima vrednost `7`,
- Izraz `x = 5` pa vrednost `5`.
  - izraz še spremeni vrednost spremenljivke x

- Pri sestavljanju izrazov je potrebno poznati operatorje:
  - prireditveni,
  - primerjalni,
  - logični,
  - aritmetični,
  - bitni in
  - posebni operatorji.

Prireditveni operatorji

- Osnovni prireditveni operator je `=`
  - spremenljivki na levi strani priredi vrednost izraza na desni strani
    `x = y`
  - Drugi prireditveni operatorji so samo kratice za kombinacije operatorja `=` z drugimi operatorji.

Prireditveni operatorji

OPERATOR	OPIS
`x = y`	Prireditev vrednosti spremenljivke y spremenljivki x
`x += y`	`X = x+y`
`x -= y`	`X = x-y`
`x *= y`	`X = x*y`
`x /= y`	`X = x/y`
`x %= y`	`X = x%y`

Primerjalni operatorji

- primerja vrednosti svojih operandov in vrne logično vrednost `true` ali `false`

`x == y`	ali je x enak y (po vrednosti)
`x != y`	ali je x različen od y (po vrednosti)
`x === y`	ali sta x in y enaka po vrednosti in tipu
`x !== y`	ali sta x in y različna po vrednosti ali tipu
`x < y`	ali je x manjši od y
`x > y`	ali je x večji od y
`x <= y`	ali je x manjši ali enak y
`x >= y`	ali je x večji ali enak y

Logični operatorji

- Sestavljamo bolj zapletene primerjave
- Operandi so običajno logične vrednosti

`x && y`	logični in
`x || y`	logični ali
`!x`	logična negacija

Aritmetični operatorji

- Osnovni aritmetični operatorji so `+, -, *, /`

`x + y`	vsota
`x - y`	razlika
`x * y`	produkt
`x / y`	kvocient
`x % y`	ostanek pri deljenju
`+x`	Sprememba predznaka na pozitivno
`-x`	Sprememba predznaka na negativno
`x++`	poveča vrednost spremenljivke x za 1
`++x`	poveča vrednost spremenljivke x za 1
`x--`	zmanjša vrednost spremenljivke x za 1
`--x`	zmanjša vrednost spremenljivke x za 1

Aritmetični operatorji

- če kakšen operand aritmetičnih operatorjev ni številskega tipa, ga JavaScript poskusi pretvoriti v število
  - vrednosti `null, false` pretvori v število 0
  - vrednost `true` v število `1`
- **POZOR:**
  - Kadar je eden od operandov pri seštevanju niz, bo JavaScript tudi drugega najprej pretvoril v niz, nato pa bo oba niza staknil. `"123" + 4` je enako `"1234"`, in ne `127`

Aritmetični operatorji

- Razlika med `++x` in `x++`.
  - `++x` je nova (že povečana) vrednost spremenljivke `x`
  - `x++` pa je stara (še nepovečana) vrednost spremenljivke `x`
- Podobno velja tudi za operator `--`

Prednosti operatorjev

- Prednost operatorjev od najmanjše k največji je naslednja:
  - vejica ,
  - prirejanje = += -= *= /= %= <<= >>= >>>= &= ^= |=
  - pogojni operator ?:
  - logični ali ||
  - logični in &&
  - bitni operator ali |
  - bitni xor ^
  - bitni in &
  - enakost ali neenakost == !=
  - primerjanje < <= > >=
  - pomik << >> >>>
  - seštevanje in odštevanje + -
  - množenje in deljenje * / %
  - negacija in inkrement ! ~ - ++ --
  - klic funkcije, članstvo () [ ] .

Prednosti izrazov – najvišja do najnižja

`. []`
`() new`
`! ~ + - ++ -- typeof void delete`
`* / %`
`+ -`
`<< >> >>>`
`< <= > >= in instanceof`
`== != === !==`
`&`
`^`
`|`
`&&`
`||`
`?:`
`= += -= *= /= %= <<= >>= >>>= &= |= ^=`
`,`

Stavki

Kaj je stavek?

- Program je sestavljen iz zaporedja stavkov
  - Izvajajo se v vrstnem redu kakor so napisani
  - Tok programa lahko spremenimo samo s klici funkcij in kontrolnimi stavki, kot so pogojni stavek, zanke ...
- Več stavkov ločimo s podpičjem ali z znakom za novo vrsto
  - Podpičje ni nujno, lahko nadomestimo z pritiskom na tipko Enter
- Vsak izraz je lahko tudi stavek

- Primer:
  - Program, ki v danem štirimestnem naravnem številu x obrne vrstni red števk.

Stavčni bloki

- Stavek je kos programske kode, ki se konča s podpičjem: napoved, izraz ali klic metode.
- Zaporedje stavkov združimo v blok z { }

Polja

Kaj je polje?

- Množica elementov enakega podatkoven tipa
- Polje ustvarimo z uporabo oglatih oklepajev, med katerimi naštejemo vse njene elemente (seznam elementov je lahko tudi prazen).

```
var imena = ["Marko", "Janez", "Luka", "Jože"]
var imena = []
```

  - prvi element dobil indeks 0, naslednji indeks 1 ...

`imena[0]` izpiše `Marko`

`imena[1]` izpiše `Janez`

Polja

- Elementi polja so lahko poljubnega tipa
- Posamezen element tabele lahko pustimo nedoločen, tako da ne vpišemo ničesar.
  - `[1,2,,4]` ima štiri elemente, pri čemer je vrednost tretjega elementa nedefinirana (`undefined`).
- Uporabne vgrajene metode za polje:
  - `array.length` - dolžina tabele

Kreiranje polja

- Kreiranje polja:
  - `var polje = []`
  - `var polje = new Array(10)` -
    prazno polje velikosti 10 - ni priporočljivo.
  - `var polje = [10]` -
    polje, kjer ima prvi element vrednost 10.

Inicializacija polja

- Polje je mogoče inicializirati s podatki hkrati ko ga kreiramo `var polje = ['januar', 'februar', 'marec'];`
- Polje lahko najprej kreiramo in ga nato napolnimo s podatki

```
var polje = [];
polje[0] = 'januar';
polje[1] = 'februar';
polje[2] = 'marec';
```

- Če preskočimo element, bo le-ta imel vrednost `undefined`

```
var polje = [];
polje[0] = 'januar';
polje[1] = 'februar';
polje[5] = 'marec';
```

Asociativna polja

Kaj je asociativno polje?

- Za dostop do elementov uporabljamo ključne besede namesto indeksov (ki so števila)

```
var ime_polja = new Array();
ime_polja['kljuc1'] = vrednost1;
ime_polja['kljuc2'] = vrednost2;
```

Dostop do asociativnega polja

- Zanka `for … in …`

```
for (spremenljivka in objekt / polje) {
  izraz
  }
```