JavaScript

Funkcije

First class funkicje

- Funkcije u JavaScript-u su objekti
  - *Mutabilne kolekcije parova svojstvo-vrednost*
- Mogu da se tretiraju kao bilo koji drugi objekti
  - Varijabla može da primi funkciju kao vrednost
  - Funkcije mogu da se smeštaju u kolekcije i druge objekte
  - Mogu da se proslede drugim funkcijama kao parametri
  - Mogu da budu vraćene iz drugih funkcija

Kreiranje funkcije

- Funkcije u JavaScriptu mogu se kreirati na dva načina:
  1. Deklaracijom funkcije
  2. Funkcijskim izrazom

Deklaracija funkcije

- Deklaracija funkcije definiše imenovanu funkciju (varijablu kojoj je dodeljena funkcija kao objekat) bez potrebe da se radi eksplicitna dodela vrednosti varijabli
-

Deklaracija funkcije

- Funkcijska varijabla je dostupna u svom opsegu i opsegu svog roditelja

```
function factorial(x){
  if(x <= 1) return 1;
  return factorial(x-1)*x;
}
factorial(5)
```

Funkcijski izraz

- Funkcijskim izrazom funkcija se definiše kao deo većeg sintaktičkog konstrukta, tipično dodele vrednosti varijabli
-
- Ovako definisane funkcije mogu da budu *imenovane* i *neimenovane*
-

Funkcijski izraz

```
var a = function() {
    return 3;
}//neimenovana

var a = function bar() {
    return 3;
}//imenovana
```

Closure

- Funkcijski izraz može da se nađe svugde gde može da bude izraz
- Funkcija može da se definiše unutar druge funkcije – unutrašnja funkcija
  - Unutrašnja funkcija ima pristup:
    - Svojim parametrima i varijablama
    - Parametrima i varijablama svoje spoljašnje funkcije, čak i kada je spoljašnja funkcija prestala sa izvršavanjem
    - Zatvaranje (closure) - **Funkcijski objekat kreiran u funkcijskom izrazu može ima pristup svom spoljašnjem kontekstu**
    - Veoma moćan alat u JavaScript-u

Pozivi funkcija

- Prilikom poziva funkcije, pored prosleđenih parametara, funkcija dobija i dva dodatna
  - `arguments`
  - `this`
- Vrednost parametra `this` zavisi od paterna poziva funkcije
- Funkcija može da se pozove kao:
  - Metoda
  - Funkcija
  - Konstruktor
  - Parcijalna funkcija

Metode

- Funkcije koje su vrednosti svojstava objekata
- Prilikom poziva funkcije vrednost parametra this je objekat nad kojim je funkcija pozvana

```
var myObject = {
  value: 0,
  increment: function () {
    this.value += 1;
  }
};
myObject.increment( );
console.log(myObject.value);
```

Opseg vidljivosti varijabli

- Deo programskog koda u kom je varijabla dostupna
- Većina programskih jezika sa sintaksom izvedenom iz C ima blokovski opseg
  - Varijabla je dostupna u bloku u kom je definisana
- JavaScript ima **funkcijski opseg vidljivosti varijabli**
  - Varijable i parametri su dostupni u **čitavom telu funkcije** u kojoj su

Closure

- Unutrašnja funkcija ima pristup varijablama i parametrima spoljašnje funkcije
  - Čak i kada je spoljašnja funkcija prestala da se izvršava
- **Funkcija ima pristup kontekstu u kom je kreirana**

Opseg vidljivosti varijabli

- Ukoliko je varijabla deklarisana (var x) u funkciji, koristimo tu varijablu
- Ukoliko nije, tražimo je u spoljašnjoj funkciji
- Ukoliko nije defnisana ni u spoljašnjoj funkciji, tražimo je u spoljašnjoj funkciji za spoljašnju funkciju
- ...
- **Sve dok ne dođemo do globalnog opsega!**
- Ukoliko zaboravimo da deklarišemo

Callback

- Izvršni kod koji se kao argument prosleđuje drugom izvršnom kodu
- Funkcija kao argument funkcije

```
function mySandwich(param1,
param2, callback) {
  alert('Started eating my
```