JavaScript 2
5. predavanje

Callbacks

POWERED BY:

nsoft sportradar

Danas radimo

- Callbacks - uvod
- Sinkronost/asinkronost
- SetTimeout i setInterval funkcija

Sinkrono izvršavanje

- čeka da se svaka operacija izvrši prije nego što nastavi dalje
- blokirajući
- Primjer: |----A-----||-----B----------||-------C------|

Asinkrono izvršavanje

- Izvršava se operacija po operacija, ali se ne čeka da operacija završi, nego se nastavlja dalje
- neblokirajući
- Primjer:

```
|----A-----|
     |-----B----------|
          |-------C------|
```

Callback

- Dio koda koji se proslijedi nekoj funkciji i koji se kasnije izvrši (*call back*)
- 2 vrste :
  - blokirajući (sinkroni)
  - odgođeni (asinkroni)

Primjer

```
function test1() {
  console.log('test1');
}
function test2() {
  console.log('test2');
}
// Naravno ako želimo pozvati funkciju test1() prvo, napišemo nju prvo
test1();
test2(); // Nakon toga pozovemo drugu funkciju
```

Callbacks

- Javascript je **event** driven jezik
- Znači moguće je da se funkcija koja je kasnije napisana u kodu prije izvrši nego prije napisana funkcija
- Da bi bili sigurni da će se neka funkcija izvršiti točno iza druge, koristimo callback-e

Kako napraviti callback

- proslijedimo funkciji, kao parametar, drugu funkciju
- nakon što se izvrši kod u funkciji kojoj smo proslijedili callback, možemo pozvati proslijeđeni callback (funkciju)

Primjer

```
function test1() {
        console.log(1);
}
function test2(callback) {
        console.log(2); // 2
        callback(); // 1
        // pošto prosljeđujemo funkciju test1, ovo zapravo
        // izgleda kao da smo ovdje napisali test1()
}
test2(test1); //proslijedimo funkciju test1 kao callback
```

Zadatak 1

- Deklarirati dvije funkcije.

  Prva funkcija treba ispisati:

  console.log('Ovo je prva funkcija');

  Druga funkcija treba ispisati:

  console.log('Ovo je druga funkcija');

  Prvoj funkciji proslijediti drugu funkciju kao callback.

rješenje

Zadatak 2

- Deklarirati dvije funkcije.

  Prva funkcija prima 2 parametra: **a i callback.**

  Druga funkcija prima samo jedan parametar: **b** i ispisuje ga.

  Iz prve funkcije pozvati drugu funkciju i proslijediti joj parametar **a** koji je proslijeđen prvoj funkciji.

  Pozvati prvu funkciju, proslijediti joj neki broj i drugu funkciju kao callback.

  rješenje

Novi način deklariranja callback-a

```
function mainFunction(a, b, callback) {
  return callback(a, b);
        // Bitna je referenca, a ne ime funkcije
}
var rezultat = mainFunction(2, 2, function(c, d) {
        // Odmah definiramo funkciju, ali bez imena
  return c+d;
});`
console.log(rezultat);
```

Zadatak 3

- Uraditi prethodni zadatak koristeći novi način deklariranja callback-a.
  rješenje

Simuliranje asinkronosti

- **setTimeout()** - odgađa izvršenje koda

  setTimeout(callback, 1000);

- prvi parametar je **callback** funkcija, drugi parametar je koliko će odgoditi izvršavanje koda u **milisekundama**

Primjer

```
var arr = [1, 3];
function postpone(myCallback) {
        setTimeout(myCallback,1000)
;
}

postpone(function () {
        arr.push(2);
        arr.push(5);
});
console.log(arr);
```

Ispisat će se 1 i 3,

jer će se zbog asinkronosti prije izvesti linija ispisa nego što će se dodati novi članovi u niz.

Primjer asinkronog izvršavanja

```
var a = 2, b = 4;

setTimeout(test2, 1000);

function test2() {
    a = 10, b = 12;
    console.log(a + ' - ' + b);
//Nakon što se pozove callback a i b se update-aju i ispiše se: 10 - 12
}
console.log(a + ' - ' + b);
//Ispisati će se 2 - 4 jer se test2 funkcija još nije izvršila
```

Zadatak 4

- Deklarirati niz, zatim deklarirati funkciju koja će umetnuti dvije numeričke vrijednosti u niz.

  Koristeći **setTimeout** funkciju pozvati deklariranu funkciju nakon 3 sekunde.

  Ispisati niz nakon poziva funkcije i nakon izvršavanja callback-a.

  rješenje

Zadatak 5

- Funkciji **setTimeout** proslijediti funkciju, kao prvi parametar, ali na novi način.

  Proslijeđena funkcija treba ispisati ‘Pozvana na novi način’.

  Funkciju pozvati s odgodom od 2 sekunde.

  rješenje

Zadatak 6

- Pozvati **setTimeout** sa funkcijom koja će ispisati ‘Prvi poziv’ nakon 3 sekunde.

  Pozvati opet **setTimeout** sa funkcijom koja će ispisati ‘Drugi poziv’ nakon 2 sekunde.

  Funkcije koje će ispisati navedene stringove napisati na novi način.

rješenje

Zadatak 7

- Deklarirati funkciju koja prima jedan parametar (callback).

  Unutar deklarirane funkcije koristeći **setTimeout**, pozvati proslijeđeni callback nakon 5 sekundi.

  Pozovite prvu funkciju s tim da joj proslijedite drugu funkciju, deklariranu na novi način.

  Druga funkcija treba ispisati ‘Napokon ispis’.

  rješenje

Zadatak 8

Napraviti objekt **osoba.**

Napisati funkciju koja će primati jedan parametar (callback) i koja će nakon 1 sekunde pozvati taj callback. Callback funkcija treba objektu osoba dodati 2 property-a - **ime i prezime.**

Ispisati objekt osoba prije i poslije izvršavanja callback-a.

rješenje

Funkcija setInterval()

- **setInterval()** - ponavlja izvršavanje koda u intervalima zadane duljine

  setInterval(callback, 1000);
- prvi parametar je **callback** funkcija, drugi parametar je duljina intervala izvršavanja koda u **milisekundama**

Primjer

```
var a = 1, b = 2;
function test1() {
    a++;
    b += 2;
    console.log(a + ' - ' + b);
}
setInterval(test1, 1000);

//Ispisati će se 2 - 4,3 - 6,...
```

clearInterval(int)

- nakon što pokrenemo interval, moramo imati način da ga i zaustavimo
- trebamo spremiti poziv funkcije u neku varijablu

```
var i = setInterval(fun, 1000);
```

- funkciji **clearInterval()** proslijediti tu varijablu
- na taj način se prekida izvođenje intervala

```
clearInterval(i);
```

Primjer

```
var a = 1, b = 2;
function test1() {
        console.log(a + ' - ' + b);
        a++;
        b += 2;
        if (a > 5) {
                clearInterval(i);
        }

}
var i = setInterval(test1, 1000);
```

1 - 2

2 - 4

3 - 6

4 - 8

5 - 10

Zadatak 9

- Napisati funkciju koja simulira lotto izvlačenje.

  Funkcija nasumično odabire 5 brojeva u rasponu 1-20 svako 1 sec i ispisuje ih korisniku. (1 broj po sekundi)

  Treba se pobrinuti da nijedan izvučeni broj ne bude izvučen ponovno.

  Hint: Math.floor(Math.random() * 20); and *splice()*

rješenje

Ponavljanje

POWERED BY: nsoft sportradar

SPARK. SCHOOL

THANK YOU

for attention