JavaScript – uvod

Uvod

JavaScript (JS) je programski jezik za HTML i za Web.

Sintaksa jezika JavaScript-a je slična C jeziku, zajedno sa C#, C++, JAVA-om, PHP-om i nekim drugim jezicima spada u C-olike jezike .

Klijentski orijentisan. Jednostavan. Najpopularniji programski jezik.

JavaScript se piše unutar `body` i/ili `head` sekcije i koristeći odgovarajući tag `script`. JavaScript kod se piše u vidu odgovarajućih funkcija koje se pozivaju, na primer:

```
<script>
   function myFunction() {
      document.getElementById("div1").innerHTML = "Zdravo svete!";
   }
</script>
```

Napomena: Primeri su kompletni pa sadrže delove koji će biti naknadno objašnjeni. Treba obratiti pažnju na delove primera koji se objašnjavaju ili su već objašnjeni.

Zbog brzine učitavanja stranice programeri ubacuju script tag na dno stranice pre `</body>` taga. Kod se može pisati u posebnom fajlu koji se naknado uključuje u neku `script` sekciju.

Sadržaj HTML dokumenta

```
<!DOCTYPE html>
<html>
<body>
    <script src="ScriptFunkcije.js"></script>
</body>
</html>
```

Sadržaj fajla ScriptFunkcije.js

```
function myFunction() {
    document.getElementById("p1").innerHTML = "Izmenjeno iz JS.";
}
```

Zasebni dokumenti, fajlovi, koji sadrže JavaScript kod imaju ekstenziju **`.js`**.

Mogućnosti prikaza pomoću JavaScript-a

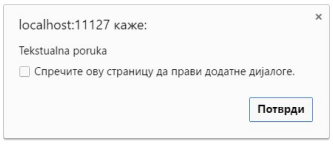
JavaScript se može koristiti za prikaz sadržaja na više načina, što je definisano mestom prikaza: pomoćni prozor, konzola čitača, unutrašnjost elementa i slično. Ispisi se vrše pomoću sledećih funkcija:

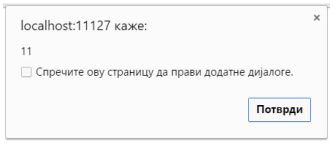
- **`window.alert()`** - koristeći pomoćni prozor za poruke
- **`document.write()`** – upisujući u HTML izlazni tok dokumenta
- **`innerHTML`** – ubacujući HTML kod unutar nekog elementa
- **`console.log()`** – upisujući u konzolu čitača

Primeri:

window.alert()

```
<!DOCTYPE html>
<html>
<body>
    <h1>Alerti</h1>
    <script>
        alert("Tekstualna poruka");
        window.alert(5 + 6);
    </script>
</body>
</html>
```



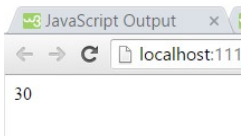


Document.write() – ubacivanje u postojći tok

```
<!DOCTYPE html>
<html>
<body>
    <h1>h1 naslov</h1>
    <script>
        document.write("<h2>ubačeni
element</h2>");
    </script>
    <p>Sledi ubačen rezultat izraza</p>
    <script>
         document.write(44 - 14);
    </script>
</body>
</html>
```

Document.write() – prepisivanje preko postojećeg toka

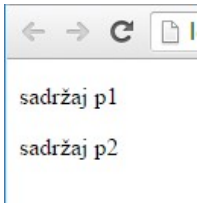
```
<!DOCTYPE html>
<html>
<head>
    <script>
        function myFunction() {
            document.write(44 - 14)
        }
    </script>
</head>
<body>
    <h1>Naslov</h1>
    <p>paragraf</p>
    <button onclick="myFunction()">Test</button>
</body>
```



```
</html>
```

innerHTML

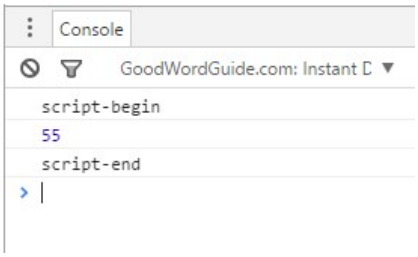
```
<!DOCTYPE html>
<html>
<body>
    <div id="div1"></div>
    <p id="p2"></p>
    <script>
      document.getElementById("div1").innerHTML =
        "<p id='p1'>Ubaceni paragraf</p>";
      document.getElementById("p2").innerHTML =
        "sadržaj p2";
      document.getElementById("p1").innerHTML =
        "sadržaj p1"; // testirajte posle
komentarisanja ove linije
    </script>
</body>
</html>
```



console.log()

```
<!DOCTYPE html>
<html>
<body>
    <div id="div1"></div>
    <script>
        console.log("script-begin");
        console.log(77 - 22);
        console.log("script-end");
    </script>
</body>
</html>
```

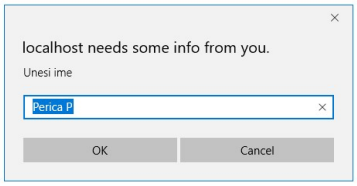
F12 –



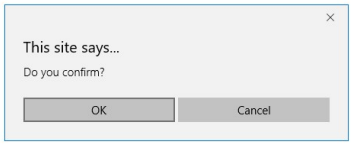
Upotreba ugrađenih prozora „prompt“ i „confirm“

Osim ugrađenog prozora „alert“ u upotrebi su: „prompt“ i „confirm“. Prozor „prompt“ omogućava unos podataka i prikazuje dva dugmeta: za potvrdu odnosno za otkazivanje unosa. Prozor „confirm“ koristi za prikaz poruka takođe sa mogućnošću potvrde ili otkazivanja. Na primer:


```
function myFunction() {
   var ime = prompt("Unesi ime",
                    "Perica P");
   if (ime != null) {
      document.getElementById("test").innerHTML =
                                  "Zdravo " + ime + "!";
      }
   }
}
```




```
function myFunction() {
    var confirmResult = confirm('Do you confirm?');
    document.getElementById("test").innerHTML =
                                      confirmResult;
}
```



JavaScript funkcije

JavaScript funkcija je blok JS naredbi tj. kod koji se poziva kao jedna celina. To se izvodi navođenjem naziva funkcije. Definisanje funkcije i poziv neke funkcije je definisano sintaksom programskog jezika. JS koristi ključnu reč **function** za definisanje funkcije, a poziv se izvodi navođenjem imena funkcije.

Na primer, neka funkcija se može izvršiti pošto se pritisne neko dugme. Trenutak pritiska na dugme nije unapred predvidiv i definiše se kao događaj vezan za neki element. Konkretno događaj pritiska (tj. klik) na dugme se naziva **click**.

```
<!DOCTYPE html>
<html>
<head>
  <script>
     function myFunction() {
       document.getElementById("p1").innerHTML = "Tekst nakon klika.";
     }
  </script>
</head>

<body>
  <p id="p1">Početni tekst</p>
  <button onclick="myFunction()">Promeni tekst</button>
</body>
</html>
```

Definisanje funkcije

Upotreba funkcije

Ukoliko funkcija vraća vrednost koristi se ključna reč return, na primer:

```
function addFunction(x, y){ return x + y; };
var c = addFunction(5, 10);
```

JavaScript tipovi promenljive

Promenljive se koriste da bi se neka vrednost čuvala ili koristila. Promenljiva može biti bilo kog tipa. Tipovi podataka JavaScripta su:

- `String` – niz karaktera, tekst. Zapisuje se između jednostrukih ili dvostrukih znakova navoda. Moguće je koristiti apostrof u stringu sa navodnicima i obratno. Specijalni znakovi su:
  - \' – apostrof,
  - \" – navodnici,
  - \\ – obrnuta kosa crta,
  - \n – novi red,

- \r - povratak na početak reda,
- \t – tab,
- \b – backspace,
- \f - nova strana.

Neke funkcije i svojstva stringova su:

- length - dužina stringa,
- charAt() - vraća znak sa određene pozicije u stringu, počev od nule,
- concat() - spaja dva ili više stringova, npr: str2 = str0.concat(" ",str1); str = "Zdravo" + " " + "svete!"; var str = " Zdravo ".concat(" ","svete!");
- indexOf() - vraća indeks, tj. poziciju, prvog pojavljivanja nekog teksta u stringu ili -1 ako ne postoji,
- lastIndexOf() - vraća indeks, tj. poziciju, poslednjeg pojavljivanja zadatog teksta u stringu,
- match() - vraća niz pogodaka za pretragu stringa korišćenjem regularnog izraza ili null ako nema pogodaka,
- replace() - zamenjuje prvo pojavljivanje traženog teksta u stringu drugim tekstom,
- search() - Kao i indexOf() pretražuje pojavljivanje zadatog stringa i vraća njegovu poziciju, ali ima mogućnost korišćenja regularnih,
- slice() - vraća deo stringa između zadatih pozicija,
- substr() - slično kao slice() samo što je drugi parametar dužina vraćenog stringa i on ne može biti negativan
- split() - deli string na osnovu zadatog stringa i pravi niz podstringova

- `Number` – tip za brojeve. Zapisuje se u formatu pokretnog zareza sa dvostrukom preciznošću u 64 bita. Ovo je jedini tip za brojeve. Brojevi se mogu zapisivati:
  - U decimalnom formatu, sa ili bez decimala. Na primer:
    ```
    tmp = 2.68;
    tmp = 365;
    ```
  - U eksponencijalnom formatu. Na primer:
    ```
    tmp = 1212e4;
    tmp = 1.11e-5;
    ```
    Što se tiče preciznosti celi brojevi su zapisani na 15 cifara, decimalni do 17 decimala.

  Promenljive koje su tipa Number mogu na različit način formatirati prikaz. Tako se heksadecimalne vrednosti zapisuju navodeći **0x** na početku, na primer - `0xbeca`. Generalno za prikaz u binarnom, oktalnom ili heksadecimalnom brojnom sistemu može se koristiti funkcija toString() sa argumentom koji je osnova brojnog sistema. Na primer:

`var a = Number(15).toString();`
`var b = Number(15).toString(2);`
`var num8 = 14;`
`var num16 = 0xbeca;`
`var c = num8.toString(8);`
`var d = num16.toString(16);`
	15
	1111
	
	16
	e

Slične metode su:

- toExponential(brojDecimala); - vraća string od zaokruženog broja na određeni broj decimala i u eksponencijalnom zapisu,
- toFixed() - vraća string od zaokruženog broja i predstavljenog sa određenim brojem decimalnih mesta,

- toPrecision() - vraća string na osnovu broja cifara i bez decimalne tačke.

Specifične vrednosti su:

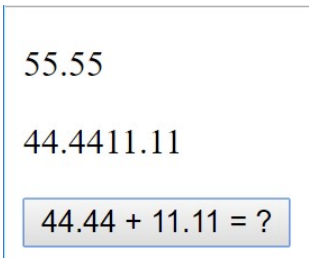
- Infinity odnosno –Infinity, predstavljaju beskonačne vrednosti, a takođe su tipa Number.
- NaN (engl. Not a Number) – vrednost koja znači da promenljiva tipa Number nije broj. Treba zapamtiti i funkciju: Number.isNaN(), koja provera da li je promenljiva broj ili ne. Neke metode vezane za brojeve su:
- Number() - Vraća broj ili NaN,
- parseFloat() - Parsira argument i vraća broj u pokretnom zarezu ili NaN,
- parseInt() - Parsira argument i vraća ceo broj ili NaN.

- Boolean – vraća jednu od dve vrednosti: true ili false. Ovaj tip se vrća kao rezultat logičkih izraza, na primer: (10 > 11), (a < b), (x == y), ... Svaka realna vrednost promenljive se može pretvoriti u logičku vrednost pomoću izraza Boolean, na primer: Boolean("Hello"); Boolean("false"); Boolean(-15);.... Za sve ove izraze logička vrednost je true.
- Array – tip objekata za niz. Ovom tipu smo posvetili posebno poglavlje. Radi se o nizu vrednosti koje čuva jedna promenljiva, na primer: var niz = ["NRT", "RT", "EPO", 44, true];
- Object – u jeziku JavaScript sve promenljive su objekti osim prostih tipova. Prosti tipovi su: string, number, boolean, null i undefined. Čak se string, boolean i number mogu kreirati i koristiti kao objekti. Jedan primer definisanog objekta je:
  var osoba={ime:"Perica", lastName:"P", starost:50};
- Null – predstavlja vrednost objekta koji nije inicijalizova tj. nepostojeću vrednost. Na primer, ako promenljiva osoba postavljena kao objekat u nekom trenutku dobije vrednost null, ona ostaje objekat i dalje. Na primer: var osoba={ime:"Perica", lastName:"P", starost:50}; osoba=null; osoba ostaje objekat sa vrednošću null. Takođe, ako se izvrši inicijalizacija promenljive var osoba=null; promenljiva je implicitno tipa object sa vrednošću null.
- Undefined – ova vrednost se u JavaScriptu dodeljuje promenljivoj koja nije inicijalizovana (ili joj je eksplicitno dodeljena vrednost undefined) , na primer: var osoba; osoba dobija vrednost `undefined` implicitno. Ovako definisana promenljiva ima vrednost i tip `undefined`.

Promenljive

Sve promenljive se deklarišu pomoću ključne reči **var**. Promenljive imaju i svoju stringovsku reprezentaciju. Ukoliko promenljiva nije string, već na primer broj, ta promeljiva se može konvertovati u string. Konverzija može biti i bez eksplicitnog zahteva. Na primer:

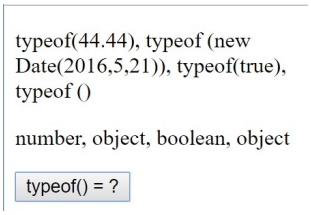
```
<!DOCTYPE html>
<html>
<head>
 <script>
  function myFunction() {
    var x = 44.44;
    var y = 11.11;
    document.getElementById("p1").innerHTML = x + y;
    document.getElementById("p2").innerHTML = "44.44" + y;
  }
  </script>
</head>
<body>
```



```
  <p id="p1">44.44 + 11.11 = </p>
  <p id="p2">"44.44" + 11.11 = </p>
  <button onclick="myFunction()">44.44 + 11.11 = ?
  </button>
</body>
</html>
```

Tip promenljive možemo prikazati primenom metode **typeof(.)**

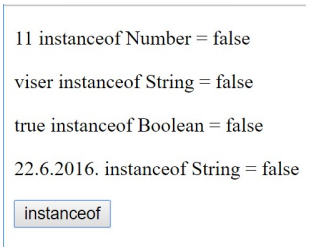
```
<!DOCTYPE html>
<html>
<head>
  <script>
    function myFunction() {
      var x = 44.44;
      var y = new Date(2016,5,21);
      var z = true;
      var o = document.getElementById("p1");
      document.getElementById("p1").innerHTML = "typeof
      (44.44), typeof (new Date(2016,5,21)), typeof(true),
      typeof (<p1>)";
      document.getElementById("p2").innerHTML = typeof (x)
       + ", " + typeof (y) + ", " + typeof (z) + ", "
       + typeof (o);
    }
  </script>
</head>
<body>
  <p id="p1">...</p><p id="p2">...</p>
  <button onclick="myFunction()">typeof() = ?</button>
</body>
</html>
```



Provera tipa neke promenljive se izvodi primenom komande **instaceof**. Sintaksa je:
`x = tmp instanceof Tip;`
gde je `tmp` promenljiva čiji tip se ispituje, a `Tip` tip podataka koji za koji se vrši ispitivanje (String, Number,...).

Na primer:

```
<!DOCTYPE html>
<html>
<head>
    <script>
    function myFunction() {
        var tmp = 11;
        document.getElementById("p1").innerHTML =
            "" + tmp + " instanceof Number = " +
            (tmp instanceof Number);
```



```
        var tmp = "viser";
        document.getElementById("p2").innerHTML =
            "" + tmp + " instanceof String = " +
            (tmp instanceof String);
        var tmp = true;
        document.getElementById("p3").innerHTML =
            "" + tmp + " instanceof Boolean = " +
            (tmp instanceof Boolean);
        var tmp = new Date(2016, 5, 22);
        document.getElementById("p4").innerHTML =
            "" + tmp.toLocaleDateString() + " instanceof
            String = " + (tmp instanceof String);
    }
    </script>
</head>
<body>
    <p id="p1">...</p>
    <p id="p2">...</p>
    <p id="p3">...</p>
    <p id="p4">...</p>
    <button onclick="myFunction()">instanceof</button>
</body>
</html>
```

Obratite pažnju na dobijeni rezultat.

Promenljive tipa Number, String i Boolean koje su u primeru korišćene kao primitivne mogu biti definisane i kao objekti. Promenite definicije promenljivih u kodu u objekte, pa proverite ishod primera, na primer:

```
var tmp = new Number(11);
var tmp = new String("asdf");
var tmp = new Boolean(true);
```