javascript დაპროგრამების ენის სახელმძღვანელო

javascript არის web დაპროგრამების ენა. ყველა თანამედროვე HTML გვერდები იყენებენ javascript. იგი არის იოლი შესასწავლად. ეს სახელმძღვანელო შეგასწავლით თქვენ დაწყებული ძირითადი ელემენტებიდან დამთავრებული თანამედროვე ელემენტებით.

შესავალი javascript-ში

javascript არის ყველაზე პოპულარული დაპროგრამების ენა მსოფლიოში. ეს გვერდი შეიცავს მაგალითებს თუ როგორ ჩავწეროთ javascrit HTML-ში.

javascript-ს შეუძლია შეცვალოს HTML ელემენტები.

javascript DOM(დოკუმენტის ობიექტ მოდელი) არის ოფიციალური W3C სტანდარტი HTML ელემენტებზე წვდომისათვის. javascript შეუძლია DOM-ზე მანიპულაციით შეცვალოს HTML შიგთავსი. შემდეგი მაგალითი ცვლის innerHTML-ის შიგთავს, რომელიც გამოცნობილია id=”DEMO” ჭდით. მაგალითი:

```
document.getElementById("demo").innerHTML = "Hello JavaScript";
```

მეთოდი **document.getElementById()** არის HTML.DOM-ის ერთ-ერთი მეთოდი.

თქვენ შეგიძლიათ გამოიყენოთ javascript, რომ:

- შეცვალოთ HTML ელემენტები;
- წაშალოთ HTML ელემენტები ;
- შექმნათ ახალი HTML ელემენტები;
- გააკეთოთ კოპირება და გაამრავლოთ HTML ელემენტები;
- და კიდევ ბევრი სხვა რამე.
  HTML-ში javascript უნდა იყოს ჩადგმული <script> და </script> ტეგებს შორის. javascript-ები შეიძლება მოვათავსოთ <body> და <head> სექციებში.
  <script> და </script> ტეგები უჩვენებს სად იწყება და სად მთავრდება javascript. სტრიქონები <script> და </script> შორის შეიცავენ javascript კოდს.
  მაგალითი:

```
<script>
function myFunction() {
   document.getElementById("demo").innerHTML = "My First JavaScript Function";
}
</script>
```

HTML -ში javascript შეიძლება იყოს გამოყენებული მხოლოდ HTML ელემენტებზე მანიპულირებისათვის.

რომ მიმართოთ HTML ელემენტს javascript-დან უნდა გამოიყენოთ document.getElementById(*id*) მეთოდი.

id ატრიბუტი გამოიყენება HTML ელემენტის მისათითებლად და innerHTMLკი ელემენტის შიგთავსის მისათითებლად.

მაგალითი:

```
<!DOCTYPE html>
<html>
<body>
<h1>My First Web Page</h1>

<p id="demo">My First Paragraph</p>
<script>
document.getElementById("demo").innerHTML = "Paragraph changed.";
</script>
</body>
</html>
```

ეკრანზე გამოტანა

მაგალითი;

```
<!DOCTYPE html>
<html>
<body>
<h1>My First Web Page</h1>
<script>
a = 5;
b = 6;
c = a + b;
console.log(c);
</script>
</body>
</html>
```

javascript სინტაქსი

ლიტერალები
ლიტერალები არიან კონსტანტები როგორიცაა, მაგალითად, რიცხვი 3.14
სტრიქონული ლიტერალები შეიძლება იყოს მოთავსებული ერთმაგ ან ორმაგ ბრჭყალებში.
რიცხვითი ლიტერალებია, მაგალითად: 3.14, 1001, 123e5
სტრიქონული ლიტერალებია: „სახლი“ ან ‘სახლი’
გამოსახულება ლიტერალების გამოთვლა ხდება;
5+6 ან 3*10
მასივი ლიტერალები განსაზღვრავენ მასივს: [3, 2, 5]
ობიექტილიტერალები განსაზღვრავენ ობიექტებს: {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"}
ფუნქცია ლიტერალები განსაზღვრავენ ფუნქციებს: function myFunction(a, b) { return a * b;}
ცვლადები
ცვლადები გამოიყენებიან მნიშვნელობების შესანახად:
var x, length
x = 5
length = 6
ცვლადი არის სახელი, ხოლო ლიტერალი არის მნიშვნელობა.

ოპერატორები
არითმეტიკული გამოსახულების მნიშვნელობის გამოსათვლელად ჯავასკრიპტი იყენებს არითმეტიკულ ოპერატორებს:
5+6*7
ცვლადისთვის მნიშვნელობის მისანიჭებლად გამოიყენება მინიჭების ოპერატორი:
x = 5
y = 6
z = (x + y) * 10
ჯავასკრიპტს აქვს მრავალი ტიპის ოპერატორი:
არითმეტიკული, შედარების, პირობითი და ლოგიკური ოპერატორები:
+, -, *, /, <, >, ==, !=, <, >
ინსტრუქციები ერთიმეორისგან გამოიყოფიან წერტილმძიმით.
ყოველი დაპროგრამების ენა განასხვავებს ცვლადებს, ფუნქციებს, ობიექტებს და სხვას უნიკალური სახელით, რომელსაც ჰქვია იდენტიფიკატორი.
იდენტიფიკატორი სახელები შეიძლება შეიცავდნენ ლათინურ ასოებს, ციფრებს, დოლარის ნიშანს ქვედა ხაზს, მაგრამ არ უნდა იწყებოდეს ციფრით. გასაღები სიტყვები არ შეიძლება გამოვიყენოთ იდენტიფიკატორად.

// დაწყებული სტრიქონი განიხილება როგორც კომენტარი და ასეთი სტრიონი უგულველყოფილია ბროუზერის მიერ.

ჯავასკრიპტის მონაცემთა ტიპები

ცვლადები შეიძლება იყვნენ სხვადასხვა ტიპის, როგორიცაა: რიცხვები, ტექსტური სტრიქონები, მასივები, ობიექტები და მრავალი სხვა.

```
var length = 16;                       // Number assigned by a number literal
var points = x * 10;                   // Number assigned by an expression literal
var lastName = "Johnson";              // String assigned by a string literal
var cars = ["Saab", "Volvo", "BMW"];   // Array assigned by an array literal
var person = {firstName:John, lastName:Doe};   // Object assigned by an object literal
```

ფუნქციები

ფუნქციის შიგნით მოთავსებული ინსტრუქციები შეიძლება გამოძახებულ იქნან მრავალჯერ.

```
function myFunction(a, b) {
    return a * b;                      // აბრუნებს  a * b
```

იდენტიფიკატორები არიან კლავიატურის რეგისტრისადმი მგრძნობიარე. მაგალითად, lastname და lastName არიან განსხვავებული იდენტიფიკატორები. ჯავასკრიპტი იყენებს unicode სიმბოლოებს.

ინსტრუქციები

ინსტრუქციები არიან ბრძანებათა სტრიქონები, რომლებსაც ითვლის ბროუზერი. ინსტრუქციები გამოიყენებიან იმისათვის, რომ ვუთხრათ ბროუზერს, თუ რა უნდა გააკეთოს.

მაგალითად: document.getElementById("demo").innerHTML = "Hello Dolly."

ფიგურულ ფრჩხილებში მოთავსებული ინსტრუქციები არის ბლოკი.

მაგალითად:

```
function myFunction() {    document.getElementById("demo").innerHTML = "Hello Dolly.";    document.getElementById("myDIV").innerHTML = "How are you?";}
```

ჯავასკრიპტის ინსტრუქციები ხშირად იწყებიან რეზერვირებული სიტყვებით იმისათვის, რომ დავაფიქსიროთ შესასრულებელი მოქმედება:

break	ამთავრებს გადამრთველს ან ციკლს.
catch	აღნიშნავს ინსტრუქციების ბლოკს, რომელიც უნდაშესრულდეს try ბლოკში, როცა შეცდომა წარმოიშვება.
continue	იმეორებს ციკლის ტანს
do ... while	იმეორებს ბლოკის გამოთვლას, სანამ პირობა ჭეშმარიტია.
for	აღნიშნავს ინსტრუქციების ბლოკს, რომელიც უნდა იყოს გამოთვლილი სანამ პირობა ჭეშმარიტია.

for ... in	აღნიშნავს ინსტრუქციების ბლოკს, რომლიც უნდა იყოს გამოთვლილი ობიეტის ან მასივის ყოველი ელემენტისათვის.
function	აცხადებს ფუნქციას.
if ... else	აღნიშნავსინსტრუქციების ბლოკს, რომელიც უნდა იყოს გამოთვლილი პირობის მიხედვით.
return	ამთავრებს ფუნქციის ტანს და აბრუნებს მნიშვნელობას.
switch	აღნიშნავს ბლოკს, რომელიც უნდა იყოს გამოთვლილი სხვადასხვა სემტხვევების ნიხედვით.
throw	აგენერირებს სეცდომას.
try	ახორციელებს შეცდომის დამუშავებას ბლოკისათვის.
var	აცხადებს ცვლადს.
while	აღნისნავს ბლოკს, რომელიც უნდა გამოითვალოს სანამ პირობა ჭეშმარიტია.

ჯავასკრიპტი უგულველყოფს ზედმეტ ხარვეზის ნიშნებს.
თუ ინსტრუქცია არ ეტევა ერთ სტრიქონზე უმჯობესია იგი გადავიტანოთ ოპერატორის ან მძიმის შემდეგ. მაგალითი:

```
document.getElementById("demo").innerHTML =
   "Hello Dolly.";
```

თქვენ შეგიძლიათ გაწყვიტოთ ტექსტი დახრილი ხაზით. მაგალითი;

```
document.getElementById("demo").innerHTML = "Hello\
   Dolly!";
```

ერთსტრიქონიანი კომენტარი იწყება ნიშნით // ხოლო მრავალსტრიქონიანი კომენტარი მოთავსებულია /* და */ სიმბოლოებს შორის. მაგალითი:

```
var x = 5;      // Declare x, give it the value of 5
var y = x + 2;  // Declare y, give it the value of x + 2
/*
The code below will change
the heading with id = "myH"
and the paragraph with id = "myp"
in my web page:
*/
document.getElementById("myH").innerHTML = "My First Page";
document.getElementById("myP").innerHTML = "My first paragraph.";
```

თუ კომენტარად ვაქცევთ ინსტრუქციებს, მაშინ ისინი არ შესრულდებიან.

მაგალითი:

```
/*
document.getElementById("myH").innerHTML = "My First Page";
document.getElementById("myP").innerHTML = "My first paragraph.";
*/
```

მონაცემთა ტიპების მაგალითი:

```
var pi = 3.14;
var person = "John Doe";
var answer = 'Yes I am!';
```

ერთ ინსტრუქციით შეგვიძლიაგამოვაცხადოთ მრავალი ცვლადი.

მაგალითი:

```
var lastName = "Doe", age = 30, job = "carpenter";
```

ან ასე:

```
var lastName = "Doe",
age = 30,
job = "carpenter";
```

ცვლადი, რომელიც გამოცხადებულია მნიშვნელობის გარეშე მას აქვს მნისვნელობა undefined.

თუ ხელმეორედ გამოვაცხდებთ ცვლადს, მას ექნება იგივე მნიშვნელობა.

მაგალითად:

```
var carName = "Volvo";
var carName;
```

```
  var y = "5";
var x = y + 2;
```

ამ შემთხვევაში x=“52”

ჯავასკრიპტს აქვს დინამიური ტიპები. ეს ნიშნავს შემდეგს:

```
var x;               // x აქვს მნიშვნელობა undefined
var x = 5;           // აქ x არის რიცხვი
var x = "John";      // აქ x არის სტრიქონი
```

ორმაგი და ერთმაგი ბრჭყალების გამოყენება:

```
var answer = "It's alright";            // ერთმაგი ბრჭყალები ორმაგი ბრჭყალების შიგნით
var answer = "He is called 'Johnny'";   // ერთმაგი ბრჭყალები ორმაგი ბრჭყალების
//შიგნით
var answer = 'He is called "Johnny"';   ორმაგი ბრჭყალები ერთმაგი ბრჭყალების
//შიგნით
```

მასივის მაგალითი:

```
var cars = ["Saab", "Volvo", "BMW"];
```

მასივის ინდექსის ათვლა იწყება ნულიდან.

ობიექტები იწერება ფიგურულ ფრჩხილებში. ობიექტის თვისებები არიან წყვილები თვისება:მნიშვნელობა გამოყოფილი არიან ერთიმეორესაგან მძიმეებით.
მაგალითად: var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};
თუ ცვლადის მნიშვნელობა არაა განსაზღვრული, მაშინ იგი არის undefined, თუ ცვლადის მნიშვნელობა ცარიელია, მაშინ მისი მნიშვნელობაა null.
რომ იპოვოთ ცვლადის ტიპი, უნდა გამოიყენოთ typeof ოპერატორი.
მაგალითი:

```
typeof "John"                 // აბრუნებსs string
typeof 3.14                   // აბრუნებს number
typeof false                  // აბრუნებს Boolean
typeof [1,2,3,4]              // აბრუნებს  object
typeof {name:'John', age:34}  // აბრუნებს  object
```

new გასაღები სიტყვით გამოცხადებული ცვლადები არიან ობიექტები.

```
var x = new String();        // Declares x as a String object
var y = new Number();        // Declares y as a Number object
var z = new Boolean();       // Declares z as a Boolean object
```

ცვლადების ობიექტებად გამოცხადება ანელებს პროგრამის შესრულებას. თვისებები არიან ობიექტებთან დაკავშირებული მნიშვნელობები. მეთოდები არიან მოქმედებები, რომლთა შესრულება შეუძლიათ ობიექტებს. მაგალითად, მანქანა არის ობიექტი, რომელსაც აქვს თვისებები, როგორიცაა: წონა და ფერი; მოქმედებები, როგორიცაა: დაძვრა და გაჩერება.
თვისებები:

```
car.name = Fiat
car.model = 500
car.weight = 850kg
car.color = white
```

მეთოდები:

```
car.start()
car.drive()
car.brake()
car.stop()
```

ყველა მანქანას აქვს ერთნაირი თვისებები, მაგრამ სხვადასხვა მანქანების თვისებების მნიშვნელობები განსხვავდებიან ერთიმეორესაგან. ყველა მანქანას აქვს ერთნაირი მეთოდები, მაგრამ ისინი შეიძლება სრულდებოდეს სხვადასხვა დროს და მეთოდების არგუმენტები შეიძლება იყოს განსხვავებული.
ობიექტები არიან ცვლადები, თვისებებითა და მეთოდებით. ჯავასკრიპტში ყველაფერი დამუშავებულია როგორც ობიექტები. მონაცემები, მასივები,სტრიქონები, ფუნქციები ...
თქვენ შეგიძლიათ შექმნათ საკუთარი ობიექტები. შემდეგი მაგალითი ქმნის „Person“ ობიექტს და უმატებს მას 4 თვისებას:

```
var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};
```
ობიექტის გამოცხადება შეიძლება შეიცავდეს მრავალ სტრიქონს:
```
var person = {
    firstName:"John",
    lastName:"Doe",
    age:50,
    eyeColor:"blue"
};
```
თქვენ შეგიძლიათ დაუმატოთ ახალი თვისებები და მეთოდები უკვე არსებულ ობიექტს. თქვენ შეგიძლიათ მიუთითოთ ობიექტის თვისებები ორნაირად:
```
var person = {
    firstName:"John",
    lastName:"Doe",
    age:50,
    eyeColor:"blue"
};
```
მეთოდზე მიმართვა ხდება ასე:
```
objectName.methodName()
name = person.fullName();
```
ფუნქციის მაგალითი:
```
function myFunction(p1, p2) {
    return p1 * p2;              // the function returns the product of p1 and p2}
```
functionName(parameter1, parameter2, parameter3) { აქ იწერება შესასრულებელი მოქმედებები}

ფუნქციის შესრულება ხდება მისი გამოძახებით შემდეგნაირად:

- როცა მოხდება რაიმე ხდომილება, რომლისათვისაც დანიშნულია ეს ფუნქცია;
- როცა კოდი გამოიძახებს ამ ფუნქციას;
- ავტომატურად.

მაგალითი:
```
var x = myFunction(4, 3);        // ფუნქციის გამოძახება

function myFunction(a, b) {
    return a * b;                // ფუნქცია აბრუნებს ნამრავლს
}
```
შედეგი იქნება:12
```
// carName არის გლობალური ცვლადი. ამიტომ ქვემოთ შეგვიძლია გამოვიყენოთ
//window.careName
```

```
function myFunction()
{
   carName = "Volvo";
}
```

HTML ხდომილება არის ნებისმიერი რამ, რასაც ბროუზერი ან მომხმარებელი აკეთებს. HTML ხდომილების მაგალითებია:

HTML ვებგვერდის ჩატვირთვის დამთავრება;

HTML შეტანის ველის შეცვლა;

HTML ღილაკზე(button) დაწკაპუნება.

მაგალითი:

```
<button onclick='getElementById("demo").innerHTML=Date()'>The time is?</button>
```

ღილაკზე დაწკაპუნების დროს გამოიტანება მიმდინარე თარიღი.

სტრიქონების დამუშავება