საქართველოს ტექნიკური უნივერსიტეტი

მ. კიკნაძე, ე. ჩიკაშუა, თ. კაიშაური,
თ. ქვანია, ლ. პეტრიაშვილი

v e b d i z a i n i

HTML & JAVA SCRIPT

დამხმარე სახელმძღვანელო

დამტკიცებულია სტუ-ს
სარედაქციო-საგამომცემლო
საბჭოს მიერ

თბილისი
2006

წიგნი წარმოადგენს დამხმარე სახელმძღვანელოს საგანში „ვებ გვერდების პროექტირება“. მასში აღწერილია HTML (Hyper Text Markup Language) ჰიპერტექსტების მარკირების ენის შესაძლებლობები, სტილების შექმნა და გამოყენება CSS (Cascading Style Sheet) ენის საშუალებით და Java script დაპროგრამების ენის საფუძვლები.

სახელმძღვანელო გათვალისწინებულია საქართველოს ტექნიკური უნივერსიტეტის ინფორმატიკისა და მართვის სისტემების ფაკულტეტის სტუდენტებისათვის და ასევე მკითხველთა ფართო წრისათვის რომლებიც დაინტერესებულნი არიან ვებ დიზაინით.

რეცენზენტები: პროფ. ზ. გასიტაშვილი
დოც. ბ. კიზირია

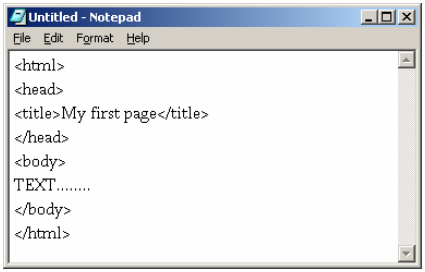
WEB გვერდების პროექტირება

შე ი

როგორც ცნობილია ინტერნეტში არსებული ნებისმიერი ინფორმაცია, რომლთა დათვალიერება შესაძლებელია ინტერნეტ-ბრაუზერების საშუალებით, ინახება რომელიმე Web სერვერზე საიტის სახით. Web საიტი შედგება Web გვერდებისაგან, რომელთა აღწერა (შექმნა) და ერთმანეთთან დაკავშირება ხდება html (Hyper Text Markup Language) ჰიპერტექსტების მარკირების ენის საშუალებით. html არ წარმოადგენს პროგრამირების ენას, მისი საშუალებით ხდება მხოლოდ Web გვერდების შექმნა და რედაქტირება.

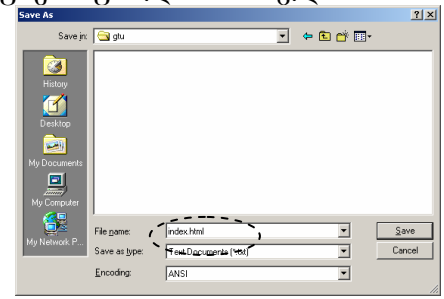
რა არის საჭირო WEB გვერდის შესაქმნელად

Web გვერდის აღწერა ხდება ნებისმიერ ტექსტურ რედაქტორში. ამისათვის ვხსნით მაგ: ტექსტურ რედაქტორს **NotePad** (Start+Programs+Accessories+**NotePad**) და ვკრიფავთ html დოკუმენტს (კონკრეტულად რას, აღწერილია შემდეგ თავებში);

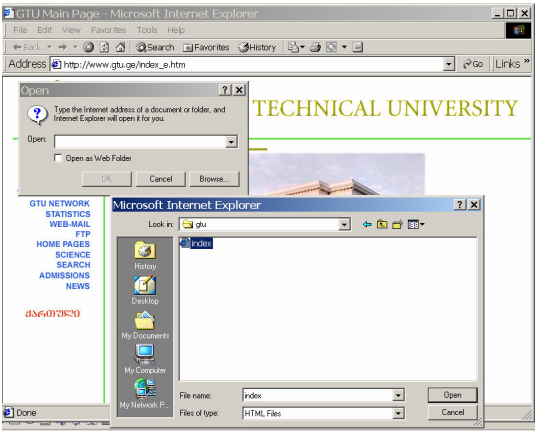


ჩვენ მიერ შექმნილი დოკუმენტი დამახსოვრების დროს ტექსტური ფაილის სახით რომ არ ჩაიწეროს დისკზე, ამისათვის ვანიჭებთ ფაილს html ან htm

გაფართოებას. ე.ი. დამახსოვრების დროს File name-ში ვკრიფავთ ფაილის_სახელს.html მაგ: **index.html**;



ჩვენ მიერ შექმნილი Web გვერდის დათვალიერება ხდება ინტერნეტ-ბრაუზერის საშუალებით. ე.ი შევდივართ Microsoft Explorer ან Netscape Navigator-ში და ვიძახებთ ფაილს File+Open+Browse და ვირჩევთ შესაბამის ფაილს.



შენიშვნა: თუ დაგვჭირდა Web გვერდის რედაქტირება ვბრუნდებით ტექსტურ რედაქტორში (NotePad) სადაც ავკრიფეთ პროგრამა, ვარედაქტირებთ შესაბამის დოკუმენტს და ვიმახსოვრებთ. (ინტერნეტ-ბრაუზერიდან html დოკუმენტის გახსნა ხდება კონტექსტური მენიუდან View Source ბრძანებით.)

ინტერნეტ-ბრაუზერში დოკუმენტის განახლება ხდება ინსტრუმენტების ველიდან  Refresh ბრძანებით.

ძირითადი მცნებები

ნებისმიერი html დოკუმენტი შედგება ინტერნეტ-ბრაუზერში გამოსატანი ტექსტისაგან და მმართველი სიმბოლოებისაგან ანუ **ტეგებისაგან**.

ტეგი ეს არის ბრძანება რომელიც მოთავსებულია < და > სიმბოლოებს შორის (მაგ. <p>) და განსაზღვრავს მის შემდეგ წარმოდგენილი ტექსტის ან გრაფიკული გამოსახულების მდებარეობას, ზომას, ფერს და ა. შ.

როგორც წესი გამოიყენება **<ტეგის_სახელი>** - გამხსნელი და **</ტეგის_სახელი>** - დამხურავი ტეგი, რომელთა საშუალებითაც ხდება ამა თუ იმ ბრძანების მოქმედების არის განსაზღვრა. (არსებობს გამონაკლისი ტეგები რომლებისთვისაც არ გამოიყენება დამხურავი ტეგი)

თითქმის ყველა ტეგი შედგება ორი ნაწილისაგან:

ტეგის სახელი - რომელიც განსაზღვრავა თუ რომელ ობიექტზე ხდება მოქმედება;

ატრიბუტი – მიუთითებს რა მოქმედება უნდა შესრულდეს აღნიშნულ ობიექტზე.

<u>ტეგის საბოლოო ფორმატი:</u>

<ტეგის_სახელი ატრიბუტი1=”მნიშვნელობა”
ატრიბუტი2=”მნიშვნელობა”>

...

მოქმედების ველი

....

</ტეგის_სახელი>

მაგ.: ****
saqarTvelos teqnikuri universiteti
****

საბაზო ტეგები

<HTML> ... </HTML> - არის Web დოკუმენტის პირველი და ბოლო ტეგი, რომელიც მიუთითებს რომ აღნიშნული ფაილი არის web გვერდი და არა უბარლო ტექსტი.

<HEAD> ... </HEAD> - სათაურის ბლოკი, სადაც იწერება ზოგადი ინფორმაცია ვებ გვერდის შესახებ.

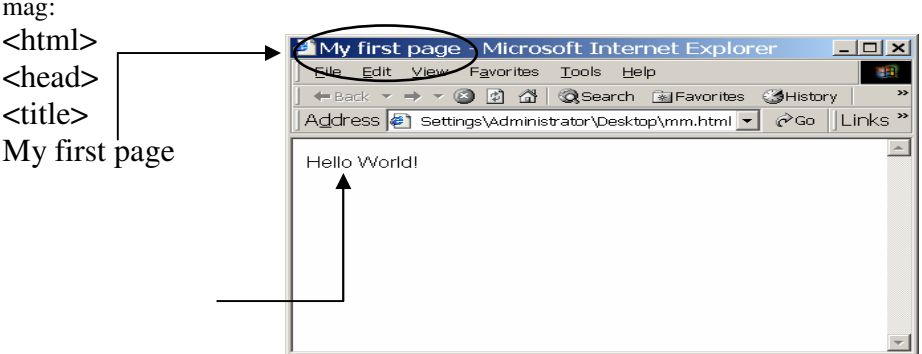
<TITLE> ... </TITLE> - ტეგებს შორის იწერება დოკუმენტის სახელი, რომელიც შემდგომ გამოისახება ბრაუზერის სათაურის ველში.

<BODY> ... </BODY> - ტეგი ხსნის დოკუმენტის ტანს, სადაც აღიწერება მთლიანად დოკუმენტი (web გვერდის შემცველობა).

<u>html დოკუმენტის ზოგადი სტრუქტურა:</u>

<HTML>
<HEAD>
<TITLE> დოკუმენტის სახელი **</TITLE>**
</HEAD>
<BODY>
... დოკუმენტის შემცველობა **...**
</BODY>
</HTML>

mag:



```
</title>
</head>
 <body>
  Hello World!
 </body>
</html>
```

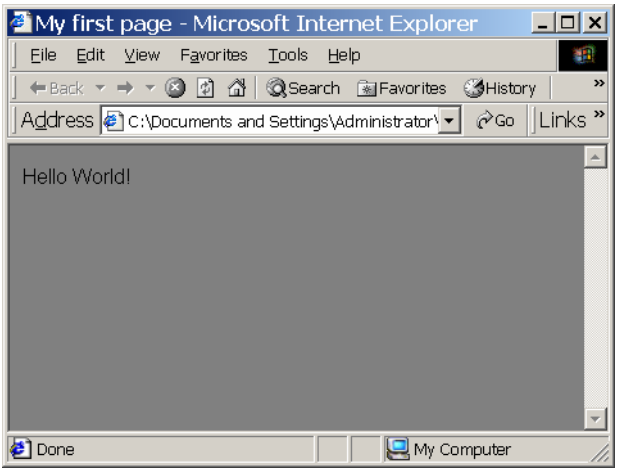
თუ BODY ტეგში არ მიუთითეთ Web გვერდის ფონის ფერი, ავტომატურად ირჩევს თეთრ ფონს. შესაძლებელია ფონის შეცვლა Bgcolor ატრიბუტით.

არსებობს ფერის მითითების ორი გზა:

1. ფერის ინგლისურენოვანი სახელის მითითებით
   <body **bgcolor=gray>**
2. ფერის ექვსციფრიანი კოდის მითითებით
   <body **bgcolor=#808080>**

მაგ:

```
<html>
 <title>My first page </title>
  <body bgcolor=gray>
   Hello World!
  </body>
</html>
```



აბზაცის ფორმატირება

<P> - ტეგი ხსნის ახალ აბზაცს, სადაც შეიძლება მოვათავსოთ როგორც ტექსტი, ასევე გრაფიკული გამოსახულება.

ALIGN ატრიბუტის საშუალებით ხდება მდებარეობის განსაზღვრა. ამ ატრიბუტს შეიძლება მიენიჭოს შემდეგი მნიშვნელობები:

LEFT-მარცხნივ**, CENTER-**ცენტრი, **RIGHT-**მარჯვნივ.

მაგ.: **<P ALIGN=CENTER>**

შენიშვნა: ALIGN ატრიბუტის საბაზო მნიშვნელობაა LEFT-მარცხნივ, ე.ი. თუ გვინდა აბზაცის მარცხნივ გასწორება, მაშინ შეგვიძლია არ მიუთითოთ ეს ატრიბუტი და <P ALIGN=LEFT>-ის ნაცვლად დავწერთ <P>.

**
 -** ტეგის საშუალებით ხდება ახალ სტრიქონზე გადასვლა. ამ დროს უცვლელი რჩება წინა აბზაცში მითითებული მდებარეობა.

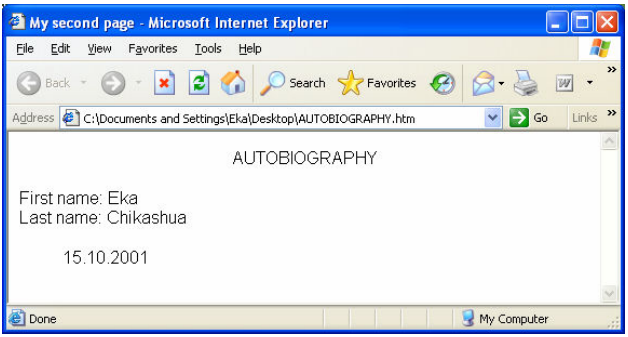
 ტეგის დახურვა არ არის საჭირო ე.ი. </BR> - არასწორია.

<BLOCKQUOTE> - ტეგის საშუალებით ხდება ტექსტის ტაბულაცია.

მაგ:

```
<html>
<title> My second page </title>
```

```
<body>
<p align=center>
AUTOBIOGRAPHY
</p>
<p align=left>
First name: Eka
<br>
 Last name: Chikashua
</p>
<blockquote>
15.10.2001
</ blockquote >
</body>
</html>
```



ტექსტის ფორმატირება

FONT – ტეგის საშუალებით ხდება შრიფტის აღწერა. ამ ტეგს გააჩნია შემდეგი ატრიბუტები:

FACE – შრიფტის დასახელება,

SIZE – ზომა,

COLOR – ფერი.

მაგ.: **<font face=acadnusx size=4 color=red>**

შრიფტის საბაზო მნიშვნელობაა - Times New Roman. ე.ი. იმ შემთხვევაში თუ არ არის მითითებული შრიფტის დასახელება ავტომატურად იწერება ინგლისურად.

ასევე შესაძლებელია ერთდროულად მიუთითოთ რამოდენიმე შრიფტი. <font face="I შრიფტი, II შრიფტი ... "> იმ შემთხვევაში თუ მომხმარებლის კომპიუტერზე არ არის დაყენებული I შრიფტი, web გვერდის შესაბამის ნაწილის გამოიტანა მოხდება შემდეგი შრიფტით და. ა.შ. მაგ.: **<font face="AcadNusx, DumbaNusx">**

შრიფტის ზომის საბაზო მნიშვნელობაა – 3.

არსებობს ზომის მითითების ორი ხერხი: აბსოლიტური <font size=4> და შედარებითი <font size=+1>.

აბსოლიტური მნიშვნელობა	1	2	3	4	5	6	7
შედარებითი მნიშვნელობა	-2	-1	-	+1	+2	+3	+4
	6pt	8pt	10pt	12pt	14pt	18pt	27pt

შრიფტის ფერის საბაზო მნიშვნელობაა – შავი.

მაგ:

```
<html>
<title> GTU </title>
<body>
<p>
Georgian Technical University <br>
<font size=5>Georgian Technical University</font> <br>
<font face=acadnusx>saqarTvelos teqnikuri universiteti </font> <br>
<font face=acadnusx size=5>saqarTvelos teqnikuri universiteti </font>
</p>
</body>
</html>
```

Georgian Technical University
Georgian Technical University
საქართველოს ტექნიკური უნივერსიტეტი
საქართველოს ტექნიკური უნივერსიტეტი

შესაძლებელია HTML დოკუმენტის საბაზო პარამეტრების შეცვლა <BaseFont> ტეგის საშუალებით, რომელიც უნდა მოთავსდეს <Body> ტეგის შემდეგ.

მაგ:

```
<html>
<title> GTU </title>
<body>
<BaseFont face=acadnusx size=4>
<p> saqarTvelos teqnikuri universiteti
 <br><Font face=arial> Georgian Technical University</font>
.....
```

განხილულ მაგალითში ვინაიდან საბაზო შრიფტად არჩეული გვაქვს acadnusx ავტომატურად ტექსტი ვებ საიტზე გამოჩნდება ქართულად და იმ შემთხვევისათვის თუ გვჭირდება ინგლისური ტექსტი უნდა მიუთითოთ შესაბამისი შრიფტი.

შრიფტის სტილები

ტეგი	დასახელება	ფუნქცია
<I> ... </I> <EM> ... </EM>	Italic	დახრილი
<B> ... </B> <STRONG> ... </STRONG>	Bold	მუქი
<U> ... </U>	Underline	ხაზგასმული
_{...}	subscript	ზედაინდექსი
^{...}	superscript	ქვედაინდექსი

ჰორიზონტალური ხაზი

საჭიროების შემთხვევაში web გვერდის ჰორიზონტალური გაყოფა შესაძლებელია „ჰორიზონტალური ხაზის“ გამოყენებით, რომლის აღწერა ხდება <**HR**> - Horizontal rule ტეგით.

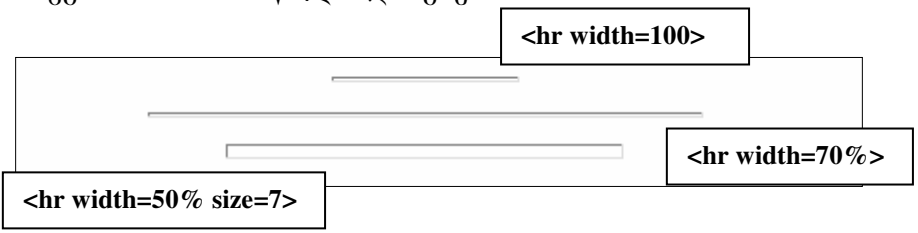
შესაძლებელია ხაზის ზომების მითითება შემდეგი ატრიბუტების დახმარებით:

SIZE - ხაზის სისქე;

WIDTH – ხაზის სიგრძე.

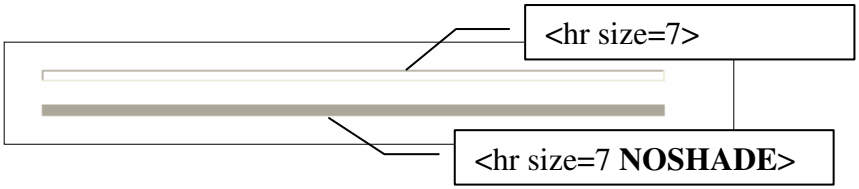
გამოიყენება ხაზის სიგრძის მითითების ორი ვარიანტი:

1. რიცხვითი მნიშვნელობა, რომელიც მიეთითება პიქსელებში (1 Pixel = 1 წერტილი ეკრანზე.
2. პროცენტული მნიშვნელობა (%), რომელიც მიუთითებს ეკრანის რა ნაწილი დაიკავოს ხაზმა.



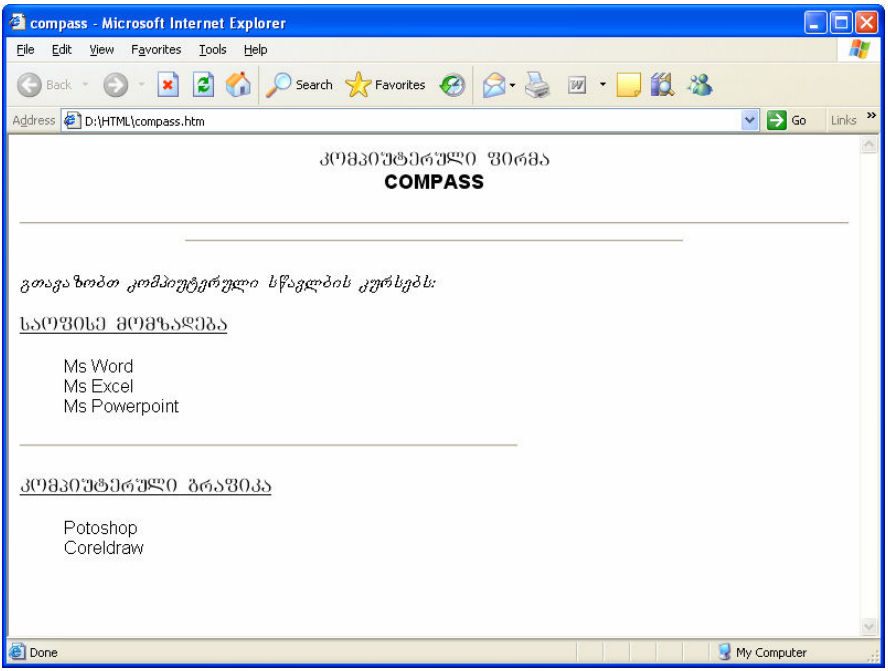
ხაზის მდებარეობის არჩევა შესაძლებელია **ALIGN=Left / Center / Right** ატრიბუტის საშუალებით. (ავტომატურად ხაზის ჩასმა ხდება ეკრანის ცენტრში).

ხაზის სტილის შეცვლა ხდება **NOSHADE** ატრიბუტით, რომელიც ხსნის ხაზის ეფექტს.



შენიშვნა: <HR> ტეგს არ გააჩნის დამხურავი ტეგი. ე.ი. </HR> - არასწორია.

ლაბორატორიული სამუშაო



გრაფიკული ობიექტის ჩასმა WEB გვერდზე

Web გვერდის სრულყოფისათვის გარდა ტექსტური ინფორმაციისა გამოიყენება გრაფიკული ობიექტები (*.gif ან *.jpg ფორმატის ნებისმიერი ფაილი).

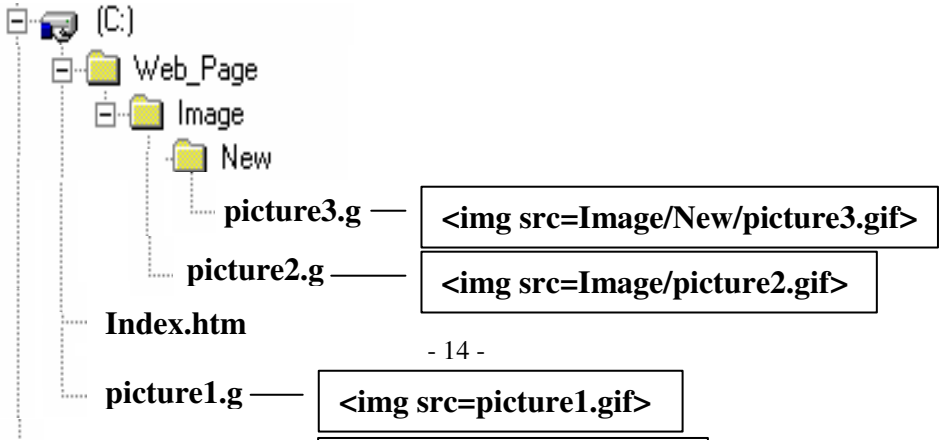
გრაფიკული ფაილის HTML დოკუმენტში ჩასმა ხდება შემდეგი ტეგით:

<IMG SRC=სრული გზა ფაილამდე>

იმ შემთხვევაში თუ HTML დოკუმენტი და გრაფიკული ფაილი მოთავსებულია ერთ საქაღალდეში ფაილის სრული გზის ნაცვლად მიეთითება მხოლოდ ფაილის სახელი.

<IMG SRC=ფაილის სახელი> ე.ი. **<IMG SRC=*.GIF >**
ან **<IMG SRC=*.JPG>**

განვიხილოთ შემთხვევეი როცა HTML დოკუმენტი და გრაფიკული ფაილი არ არის მოთავსებული ერთ საქაღალდეში.



1) თუ სურათი და HTML დოკუმენტი მოთავსებულია ერთ საქაღალდეში, მაშინ ვუთითებთ მხოლოდ სურათის სახელს:

<img src=ფაილის სახელი>

მაგ. **<img src=picture1.gif>**

2) თუ სურათი მოთავსებულია HTML დოკუმენტის საქაღალდის ქვესაქაღალდეში, მაშინ მიეთითება ქვესაქაღალდის სახელი და შემდეგ სურათის სახელი.

<img src=ქვესაქაღალდე/ფაილის სახელი>

მაგ. **<img src=Image/picture2.gif>**

ან **<img src=Image/New/picture3.gif>**

3) თუ სურათი მოთავსებულია HTML დოკუმენტის საქაღალდის გარეთ, ../ ხდება საქაღალდიდან გასვლა.

<img src=../ფაილის სახელი>

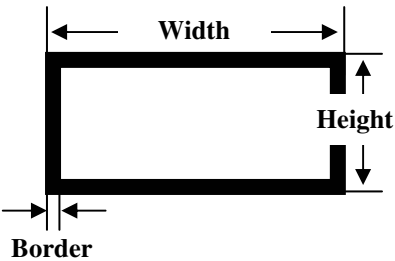
მაგ. **<img src=Image/picture4.gif>**

4) თუ სურათი მოთავსებულია HTML დოკუმენტის საქაღალდის გარეთ, სხვა საქაღალდეში, მაშინ ხდება საქაღალდიდან გასვლა (../), მერე მიეთითება იმ საქაღალდის სახელი რომელშიც სურათია მოთავსებული და ბოლოს სურათის სახელი.

<img src=../საქაღალდე/ფაილის სახელი>

მაგ. **<img src=../Old/picture5.gif>**

შესაძლებელია სურათის ზომების მითითება და



სურათის ჩასმა ჩარჩოში შემდეგი ატრიბუტების საშუალებით:

Width=რიცხვი ან % - სიგრძე;
Height=რიცხვი ან % **-** სიგანე.
Border=რიცხვი – ჩარჩოს სისქე

სურათისა და ტექსტის ურთიერთმედებარეობა

იმ შემთხვევაში თუ არ მიუთითეთ სურათის მიმართ ტექსტის მდებარეობა ავტომატურად მათი გასწორება ხდება ქვემოდან.

მაგ.

. . .

<img src=emblem.gif>
<font face=acadnusx>saqarTvelos teqnikuri universiteti


საქართველოს ტექნიკური უნივერსიტეტი

თუ გვინდა მდებარეობის შეცვლა სურათის აღმწერ ტეგში უნდა მიუთითოთ ატრიბუტი align რომელსაც შეიძლება მივანიჭოთ მნიშვნელობები middle - ცენტრი და top- ზემოთ.

მაგ.

. . .

<img src=emblem.gif **align=middle**>
<font face=acadnusx>saqarTvelos teqnikuri universiteti


ქართველოს ტექნიკური უნივერსიტეტი

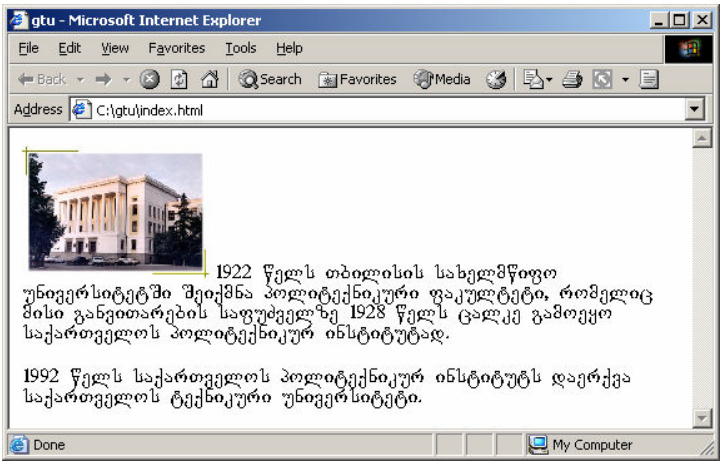
. . .
<img src=emblem.gif **align=top**>
<font face=acadnusx>saqarTvelos teqnikuri universiteti

საქართველოს ტექნიკური უნივერსიტეტი

ზემოთ აღნიშნულ სამივე შემთხვევაში სურათის გასწვრივ იწერება მხოლოდ ერთი სტრიქონი (არჩეულ ადგილას) და ტექსტის დანარჩენის სტრიქონები გადადის სურათის ქვემოთ.

მაგ.

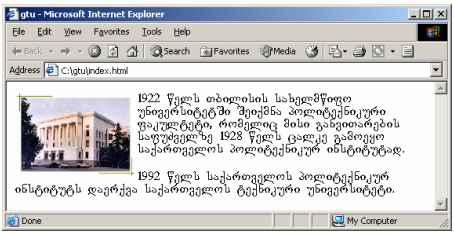
```
<html>
<title>gtu</title>
<body>
<p>
<b><img src=gtu.jpg width=150></b>
<font face=acadnusx>1922 wels Tbilisis saxelmwifo universitetSi Seiqmna politeqnikuri fakulteti, romelic misi ganviTarebis safuZvelze 1928 wels calke gamoeyo saqarTvelos politeqnikur institutad.
</p>
<p>1992 wels saqarTvelos politeqnikur instituts daerqva saqarTvelos teqnikuri universiteti.
</font>
</p>
</body>
</html>
```



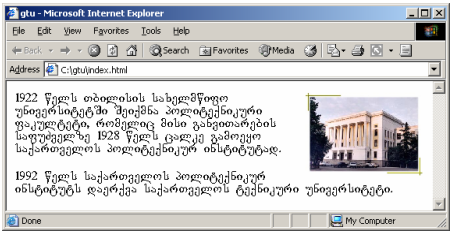
იმ შემთხვევაში თუ გვინდა სურათის გასწვრივ დავწეროთ **რამოდენიმე სტრიქონი**, მაშინ სურათის აღმწერ ტეგში უნდა მიუთითოთ ატრიბუტი Align, რომელსაც შეიძლება მიუთითოთ შემდეგი მნიშვნელობები left-მარცხნივ ან right-მარჯვნივ.

მაგ.:

```
<html>
<title>gtu</title>
<body>
<img src=gtu.jpg  align=left>
<p> <font face=acadnusx>
   1922 wels Tbilisis saxelmwifo universitetSi Seiqmna politeqnikuri fakulteti, romelic misi ganviTarebis safuZvelze 1928 wels calke gamoeyo saqarTvelos politeqnikur institutad
</p>
<p>1992 wels saqarTvelos politeqnikur instituts daerqva saqarTvelos teqnikuri universiteti.
</font>
</p>
</body>
</html>
```



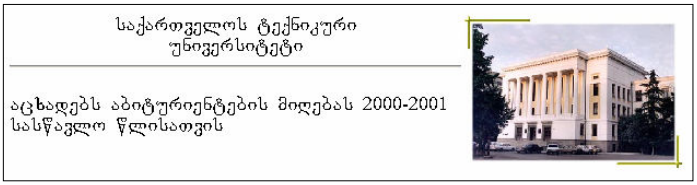
თუ ზემოთ განხილულ პროგრამაში შევცვლით სურათის მდებარეობას **<img src=gtu.jpg align=right>** მაშინ შესაბამის web გვერდს ექნება შემდეგი სახე.



როცა სურათის გვერდით არის რამოდენიმე სტრიქონი, პირველად ხდება სურათის აღწერა, შემდეგ ტექსტის მდებარეობის <p align=left ან center ან right>, შრიფტის და ა.შ.

მაგ.:

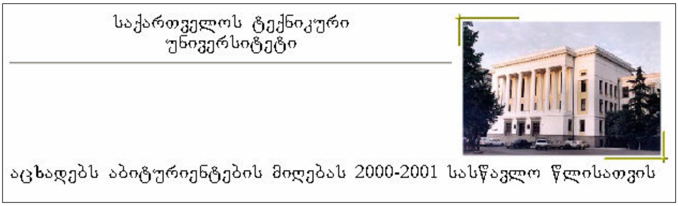
```
<html>
<title>gtu</title>
<body>
<img src=gtu.jpg width=200 align=right>
<p align=center> <font face=acadnusx size=4>
saqarTvelos teqnikuri <br> universiteti
<hr>
</p>
<p>
acxadebs abiturientebis miRebas 2000-2001 saswavlo wlisaTvis
</font>
</p>
</body>
</html>
```



თუ რამოდენიმე სტრიქონის შემდეგ გვინდა სურათის გასწვრივ დარჩეს ცარიელი ადგილი და ახალი სტრიქონი გაგრძელდეს სურათის ქვემოთ უნდა გამოვიყენოთ ტეგი **<br clear=all>.**

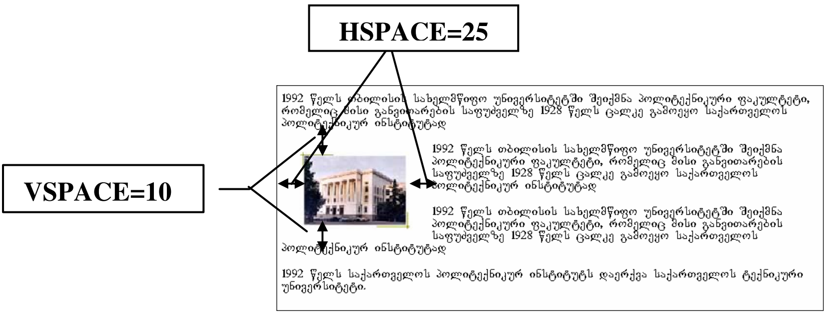
მაგ.:

```
<html>
<title>gtu</title>
<body>
<img src=gtu.jpg width=200 align=right>
<p align=center> <font face=acadnusx  size=4>
saqarTvelos teqnikuri <br> universiteti
<hr> <br clear=all>
acxadebs abiturientebis miRebas 2000-2001 saswavlo wlisaTvis
</font>
</p>
</body>
</html>
```



სურათიდან ტექსტამდე ჰორიზონტალური და ვერტიკალური დაშორების განსაზღვრა ხდება შესაბამისად HSPACE და VSPACE ატრიბუტებით

მაგ.: <img src=gtu.jpg align=left **HSPACE=25 VSPACE=10**>



ასევე შესაძლებელია სურათის ჩასმა web გვერდის ფონად. ამისათვის **BODY** ტეგში გამოიყენება ატრიბუტი **background**

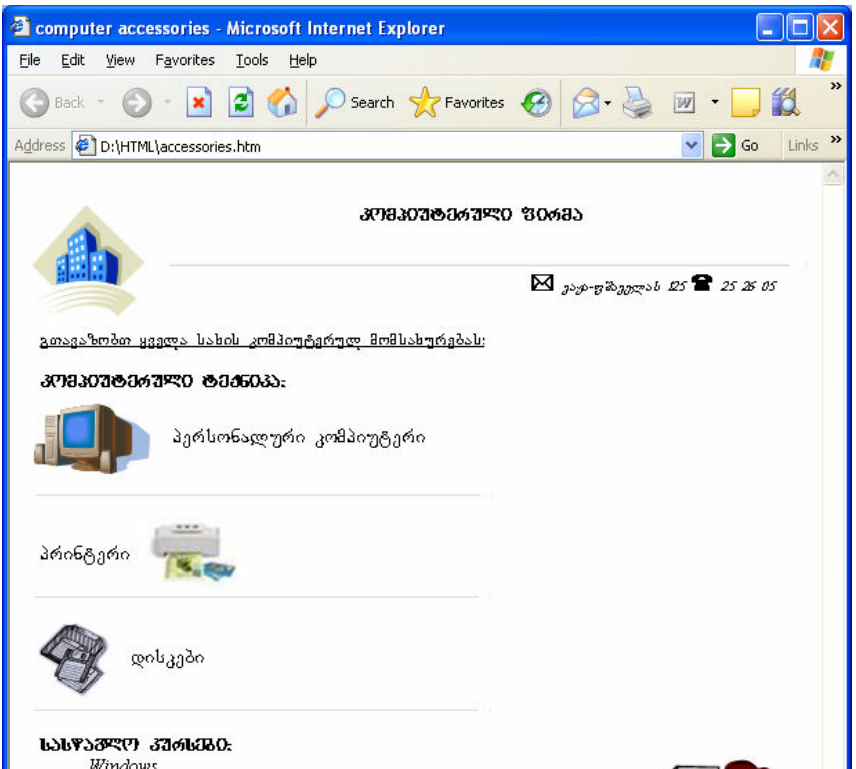
<body background=ფაილის სახელი>

ფონად შეიძლება ჩავსვათ გრაფიკული გამოსახულება jpg ან gif გაფართოებით.

მაგ.:

```
<html>
<title>gtu</title>
<body background=pic.gif>
<p>
<img src=gtu.jpg width=200 align=right>
. . .
```

ლაბორატორიული სამუშაო



ჩამონათვალი და ნუმერაცია

ჩამონათვალის აღწერა ხდება **<UL> ... </UL>** (unordered list) ტეგის საშუალებით.

ჩამონათვალის თითეული წევრი აღიწერება **<LI>** (List Item) ტეგის საშუალებით.

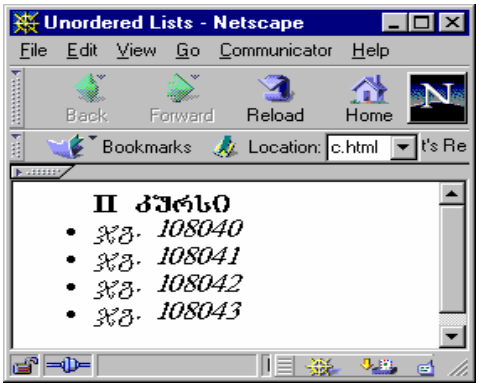
ჩამონათავლის სათაურის მითითება ხდება **<LH>** (List Head) ტეგით.

ზოგადად ჩამონათვალს აქვს შემდეგი სახე:

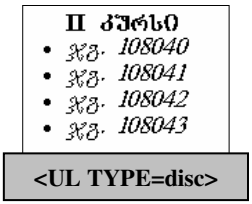
```
<UL>
<LH>სიის სათაური
<LI>I ელემენტი
<LI>II ელემენტი
<LI>III ელემენტი
</UL>
```

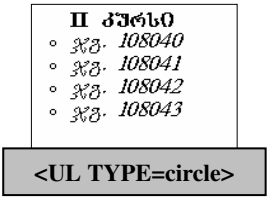
მაგ:

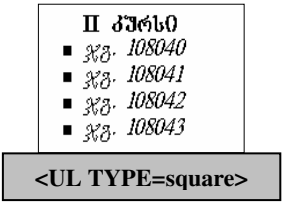
```
<HTML>
<TITLE> Unordered Lists </TITLE>
<BODY>
<UL>
<font face=acadmtavr size=4> <b>
<LH> II kursi </b> </font>
<font face=acadnusx size=4> <i>
<LI> jg. 108040
<LI> jg. 108041
<LI> jg. 108042
   <LI> jg. 108043
   </i> </font>
   </UL>
   </BODY>
   </HTML>
```



შესაძლებელია ჩამონათვალის სტილის არჩევა TYPE ატრიბუტის საშუალებით





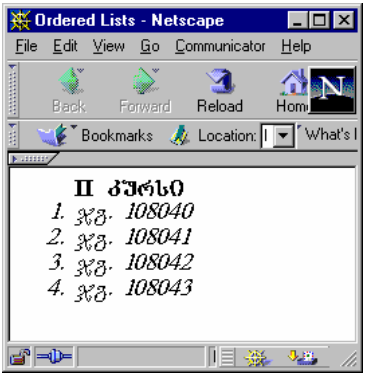


იმ შემთხვევაში თუ გვჭირდება ჩამონათავლის გადანომვრა, მის აღწერა უნდა მოვახდინოთ უნდა **<OL> ... </OL>** (ordered list) ტეგის საშუალებით.

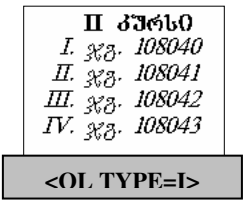
მაგ:

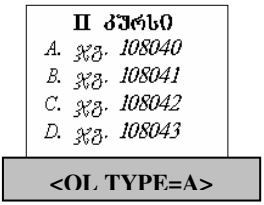
```
<HTML>
<TITLE> Unordered Lists </TITLE>
```

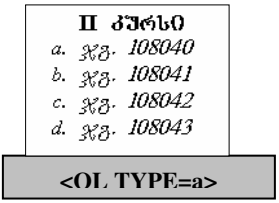
```
<BODY>
<OL>
<font face=acadmtavr size=4> <b>
<LH> II kursi </b> </font>
<font face=acadnusx size=4> <i>
<LI> jg. 108040
<LI> jg. 108041
<LI> jg. 108042
<LI> jg. 108043
</i> </font>
</OL>
</BODY>
</HTML>
```



შესაძლებელის ნუმერაციის სტილის არჩევა TYPE ატრიბუტის საშუალებით







ასევე შესაძლებელია ნუმერაციის საწყისი ციფრის არჩევა START ატრიბუტის საშუალებით

II კურსი
5. ჯგ. 108040
6. ჯგ. 108041
7. ჯგ. 108042
8. ჯგ. 108043

<OL START=5>

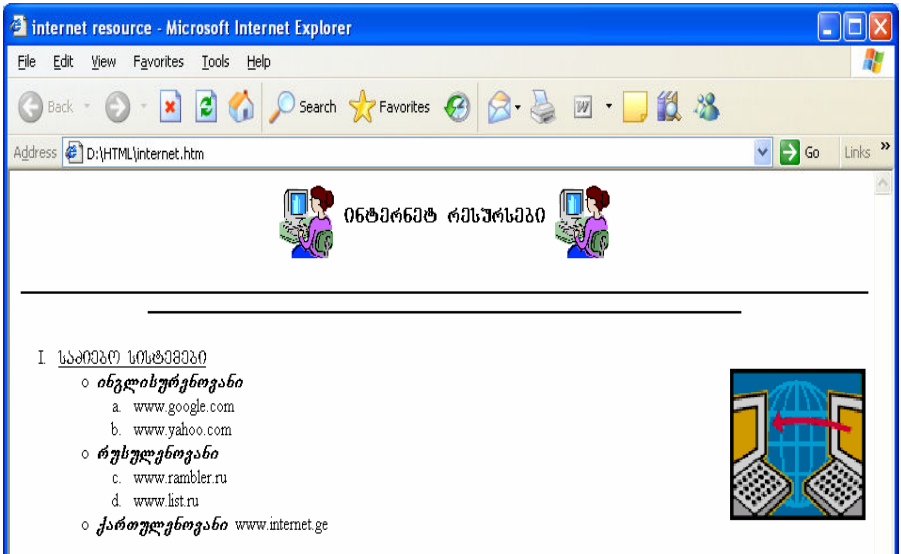
II კურსი
C. ჯგ. 108040
D. ჯგ. 108041
E. ჯგ. 108042
F. ჯგ. 108043

<OL TYPE=A START=3>

II კურსი
VII. ჯგ. 108040
VIII. ჯგ. 108041
IX. ჯგ. 108042
X. ჯგ. 108043

<OL TYPE=I START=7>

ლაბორატორიული სამუშაო



მოძრავი სტრიქონი

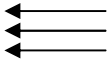
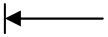
მოძრავი სტრიქონის აღწერა ხდება **<MARQUEE>** ტეგით, რომელსაც გააჩნია შემდეგი ატრიბუტები:

WIDTH:	**სამოძრაო არის სიგრძე**
HEIGHT:	**სამოძრაო არის სიმაღლე**
BGCOLOR:	**ფონის ფერი**
HSPACE:	**ჰორიზონტალური და**
VSPACE:	**ვერტიკალური დაშორება სამოძრაო არის ირგვლივ**

მაგ.:

```
<marquee width=400 height=50 bgcolor=gray>
    <font face=acadmtavr size=4 color=white>
     <b>saqarTvelos teqnikuri universiteti</b>
    </font>
</marquee>
```

მოძრავ სტრიქონში ტექსტის მოძრაობის მიმართულების, სტილის, რაოდენობის, სიჩქარის და გადაადგილების ბიჯის განსაზღვრა ხდება შემდეგი ატრიბუტებით:

DIRECTION	**მოძრაობის მიმართულება** *LEFT - მარცხნივ* *RIGHT – მარჯვნივ* *UP - ზევით* *DOWN – ქვევით*
BEHAVIOR	**მოძრაობის სტილი** *SCROLL - მოძრაობა მეორდება განუწყვეტლივ შესაბამისი ერთი მიმართულებით* 
	SLIDE – მოძრაობა სრულდება მხოლოდ ერთჯერ შესაბამისი მიმართულებით  *ALTERNATE – მოძრაობა მეორდება განუწყვეტლივ ურთიერთსაპირისპირო მიმართულებით* 
LOOP	**განსაზღვრავს მოძრაობის განმეორების რაოდენობას**
SCROLLDELAY	**განსაზღვრავს მოძრაობის სიჩქარეს**

SCROLLAMOUNT	განსაზღვრავს გადაადგილების ბიჯს.

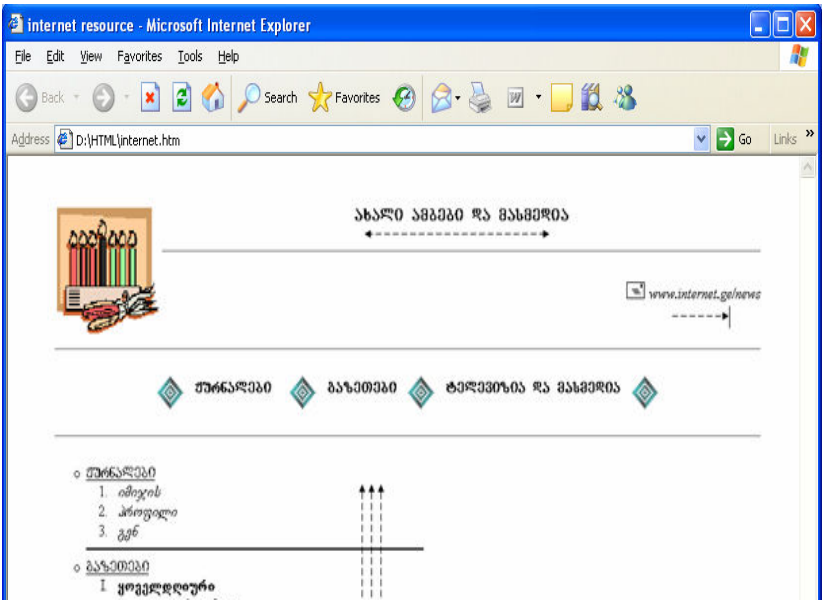
მაგ.:

<marquee width=200 height=150 bgcolor=gray hspace=50 vspace=20 direction=up behavior=scroll loop=6 scrolldelay=100 scrollamount=25>

saqarTvelo
</marquee>

ტექსტი მოძრაობს ქვევიდან ზევით და მოძრაობა განმეორდება 6-ჯერ

ან



ცხრილები

იმ შემთხვევაში თუ უკვე შექმენით რამოდენიმე html დოკუმენტი, აუცილებლად წააწყდებოდით პრობლემებს, Web გვერდზე ტექსტისა და გრაფიკული ობიექტების სასურველ ადგილზე განლაგების დროს. სწორედ ამ პრობლემის გადასაჭრელად და Web გვერდის მოსაწესრიგებლად გამოიყენება ცხრილები.

განვიხილოთ ტეგები, რომელთა სასუალებითაც ხდება ცხრილის აგება. როგორც ვიცით ცხრილი შედგება სტრიქონებისაგან და ამ სტრიქონში არსებული უჯრებისაგან. აქედან გამომდინარე სხვადასხვა ტეგის საშუალებით ხდება ცალკეული ელემენტების აღწერა:

<TABLE> - ცხრილი, <TR> - სტრიქონი, <TD> - უჯრა.

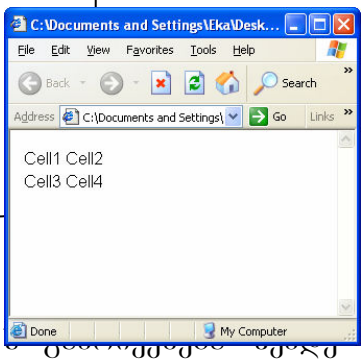
<TABLE>

<TR>	**<TD> . . . </TD>**	**<TD> . . . </TD>**	**<TD> . . . </TD>**	**</TR>**
<TR>	**<TD> . . . </TD>**	**<TD> . . . </TD>**	**<TD> . . . </TD>**	**</TR>**

</TABLE>

მაგ.:

```
<table width=80>
<tr>
 <td>Cell1</td>  <td>Cell2</td>
</tr>
<tr>
 <td>Cell3</td>  <td>Cell4</td>
</tr>
</table>
```



ცხრილის ფორმატირებისათვის [illegible] გი ატრიბუტები:

1. Border=რიცხვი - ჩარჩოს (ხაზების) სისქე

<table border=2>

Cell1	Cell2
Cell3	Cell4

<table border=7>

Cell1	Cell2
Cell3	Cell4

2. Width=რიცხვი ან % - ცხრილის სიგრძე

<table border=2 width=200>

Cell1	Cell2
Cell3	Cell4

<table border=2 width=80%>

Cell1	Cell2
Cell3	Cell4

3. Height=რიცხვი ან % - ცხრილის სიმაღლე

<table border=2 width=200 height=100>

Cell1	Cell2
Cell3	Cell4

4. Bgcolor=ფერის ნომერი ან სახელი - ფონის ფერი

<table border=2 width=200 height=100 **BGCOLOR=GRAY**>

Cell1	Cell2
Cell3	Cell4

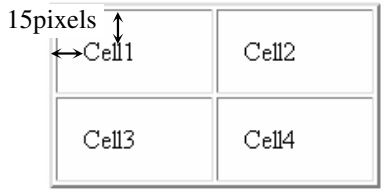
5. Background=ფაილის სახელი - გრაფიკული გამოსახულების ჩასმა ფონად

<table border=2 width=200 height=100 **BACKGROUND=PIC.JPG**>

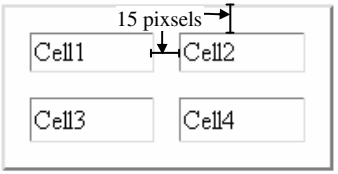
Cell1	Cell2
Cell3	Cell4

6. CellPadding=რიცხვი - მანძილი უჯრის საზღვრებიდან მასში მოთავსებულ ტექსტსა

თუ გრაფიკულ
გამოსახულებამდე
<table border=2 width=200 height=100 **CELLPADDING=15**>



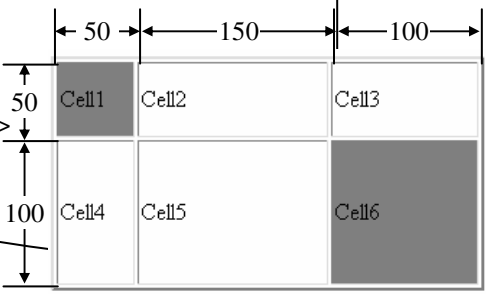
7. CellSpasing=რიცხვი - დაშორება უჯრებს შორის
<table border=2 width=200 height=100 **CELLSPACING=15**>



უჯრების ფორმატირება

იმ შემთხვევაში თუ გვინდა ცხრილის ცალკეული სვეტები და სტრიქონები იყოს სხვადასხვა ზომის ან განსხვავებული ფონი ჰქონდეს ცალკეულ უჯრებს შესაბამის ატრიბუტების გამოყენება შესაძლებელია უჯრის აღწერის დროს ე.ი. <TD> ტეგში.

```
<table border=2 width=300 height=150>
<tr>
 <td WIDTH=50 HEIGHT=50 BGCOLOR=GRAY>Cell1 </td>
 <td WIDTH=150>Cell2</td>
 <td WIDTH=100>Cell3</td>
</tr>
<tr>
 <td HEIGHT=100>Cell4</td>
 <td>Cell5</td>
 <td BGCOLOR=GRAY>Cell6</td>
</tr>
</table>
```



უჯრაში ტექსტისა და გრაფიკული გამოსახულების მდებარეობის არჩევა

- ჰორიზონტალური მდებარეობის არჩევა ხდება ატრიბუტით: **Align=** Left / Center / Right

```
<table border=2 width=80%>
<tr>
 <td align=left>Cell1</td>
 <td align=center>Cell2</td>
 <td align=right>Cell3</td>
</tr>
</table>
```

Cell1	Cell2	Cell3

- ვერტიკალური განლაგების არჩევა ხდება ატრიბუტით: **Valign=** Top / Middle / Bottom

```
<table border=2 height=150>
<tr>
 <td valign=top>Cell1</td>
 <td valign=middle>Cell2</td>
 <td valign=bottom>Cell3</td>
</tr>
</table>
```

Cell1	Cell2	Cell3

მდებარეობის საბაზო მნიშვნელობებია align=left და valign=middle

<u>მითითებები:</u>

1. ცხრილის მდებარეობის არჩევა ხდება **DIV** ტეგით:
2. ცარიელი უჯრის აღწერა ხდება შემდეგნაირად: <td> ** ** </td>
3. უჯრაში გამოყენებული ტეგები უჯრის დახურვის შემდეგ ავტომატურად წყვეტენ ფუნქციონირებას, აქედან გამომდინარე მათი დახურვა აუცილებელი არ არის.

   მაგ.: <td><font face=acadnusx><i>siaxleebi:</td>
   <td>News</td>

saკულტეტი:	kaTedrebi

4. ტექსტის შრიფტისა და სტილის აღწერა საჭიროა ყველა უჯრაში ცალცალკე;
5. იმ შემთხვევაში, თუ ერთ სტრიქონზე ყველა უჯრაში გვსურს ფონის შეცვალა, მაშინ შესაბამისი ატრიბუტი (bgcolor ან background) ყველა ცალკეული უჯრის ნაცვლად შეგვიძლია მიუთითოს სტრიქონის აღმწერ ტეგში (tr). ასევე შეიძლება მოვიქცეთ მდებარეობის არჩევისას.

უჯრების გაერთიანება

ცხრილის აგების დროს ვაწყდებით უჯრების გაერთიანების საჭიროებას. ამისათვის გამოიყენება შემდეგი ატრიბუტები:

- **Colspan=რიცხვი – სვეტების გაერთიანება;**

აღნიშნული რიცხვი განსაზღვრავს გასაერთიანებელი უჯრების რაოდენობას..

```
<table border=2 width=200>
 <tr>  <td COLSPAN=3 align=center>Cell</td> </tr>
 <tr>  <td>Cell1</td> <td>Cell2</td> <td>Cell3</td>
 </tr>
</table>
```

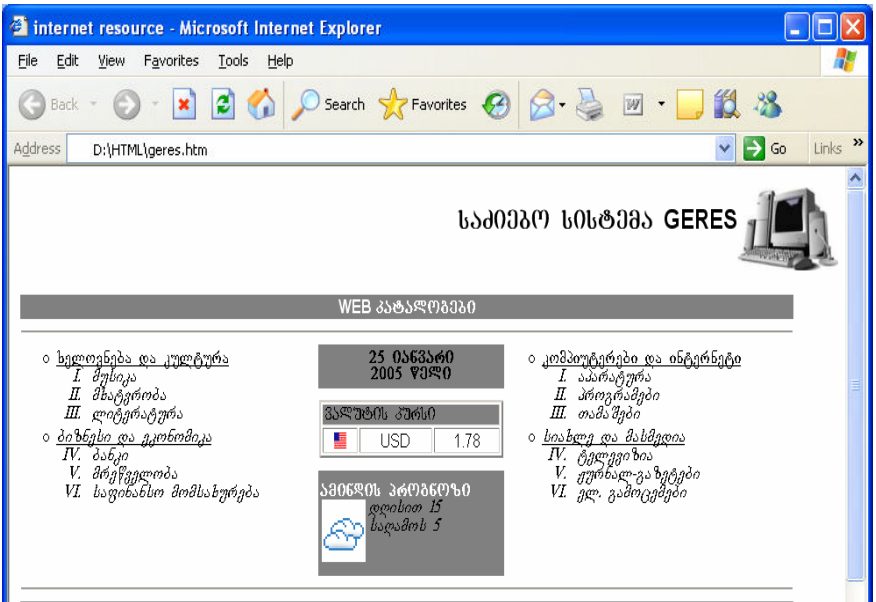
Cell		
Cell1	Cell2	Cell3

- **Rowspan=**რიცხვი – სტრიქონების გაერთიანება.

```
<table border=2 width=100>
<tr>
  <td ROWSPAN=5>Cell</td>
  <td>Cell1</td>
</tr>
<tr> <td>Cell2</td></tr>
<tr> <td>Cell3</td></tr>
<tr> <td>Cell4</td></tr>
<tr> <td>Cell5</td></tr>
</tr>
</table>
```

Cell	Cell1
	Cell2
	Cell3
	Cell4
	Cell5

სამუშაო



ჰიპერკავშირები

Web საიტის შესაქმნელად საჭიროა ცალკეული ვებ გვერდის ერთმანეთთან დაკავშირება, კავშირი შეიძლება განხორციელდეს ტექსტის გარკვეულ ნაწილზე ან გრაფიკულ გამოსახულებაზე.

გამოიყენება კავშირის განსახორციელების 4 ტიპი:

1. **რეალური მისამართის (URL) მქონე web საიტთან დაკავშირება**

<A HREF=HTTP://მისამართი> xxx **</A>**

xxx-ით აღნიშნულ ადგილზე იწერება ტექსტი, რომლის საშუალებითაც უნდა განხორციელდეს კავშირი,

ე.ი. შესაბამის ტექსტზე მაუსის დაჭერით მოხდება მითითებულ მისამართზე არსებული საიტის გამოძახება.

მაგ.: მე ვსწავლობ სტუ-ში II კურსზე.

```
<font face=acadnusx size=4> me vcwavlob
<a href=http://www.gtu.edu.ge> stu </a>
-Si II kursze </font>
```

შენიშვნა: (*სტუ-ზე მაუსის დაჭერით უნდა მოხდეს სტუ-ს web გვერდის გამოძახება, რომლის მისამართია www.gtu.edu.ge*)

2. მომხმარებლის მიერ შექმნილი web გვერდების ერთმანეთთან დაკავშირება

<A HREF=ფაილის სახელი.HTML> xxx **</A>**

მაგ.: *ჯგუფი №108040*

1. ზაზა
2. მაკა
3. ნინო

```
<font face=acadnusx size=4>
<i> jgufi #108040 </i>
<ol>
<li>zaza
<li><a href=biography.html> maka </a>
<li> nino
</ol>
```

შენიშვნა: (*სიტყვა მაკა-ზე მაუსის დაჭერით უნდა მოხდეს მაკას ავტობიოგრაფიის გამოძახება, რომელიც წინასწარ არის აღწერილი და დამახსოვრებული* biography.html *ფაილში)*

3. კავშირი, რომლის საშუალებითაც შესაძლებელია წერილის გაგზავნა აღნიშნულ საფოსტო მისამართზე

<A HREF=MAILTO:საფოსტო მისამართი> xxx **</A>**

მაგ.**:** ჩვენთან დაკავშირება შესაძლებელია ელ-ფოსტის საშუალებით

```
<font face=acadnusx size=4>
CvenTan dakavSireba SesaZlebelai
<a href=mailto:info@gtu.ge> el-fostis </a>
saSualebiT </font>
```

შენიშვნა: (სიტყვაზე *„ელ-ფოსტის“ მაუსის დაჭერით უნდა მოხდეს წერილის გაგზავნა info@gtu.ge მისამართზე).*

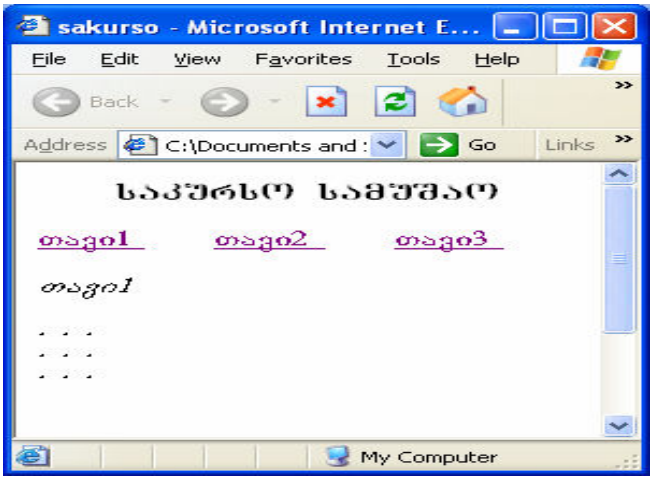
4. ერთი ფაილის ფარგლებში, მარკერით მონიშნულ ადგილთან დაკავშირება;

იმისათვის რომ, ერთი ფაილის ფარგლებში, სხვადასხვა ადგილთან (სტრიქონთან) მოხდეს კავშირის განხორციელება, წინასწარ უნდა იქნას მარკერი დაყენებული შესაბამის ადგილებზე ანუ სახელი უნდა დაერქვას აღნიშნულ ადგილებს.

მარკერის დაყენება ხდება **<A NAME=მარკერის სახელი>** ტეგის საშუალებით. აღნიშნული ტეგი იწერება იმ ტექსტის წინ სადაც გვინდა მარკერის დაყენება.

გადასვლა შესაბამის მარკერზე ხდება შემდეგი ტეგით: **<A HREF=#სანიშნის სახელი >**

მაგ.: *შევქმნათ „საკურსო სამუშაო“-ს web გვერდი, რომელიც შედგება 3 თავისაგან. ბრაუზერის ფანჯარაში ჩანს მხოლოდ თავი1-ის შემცველობა, იმისათვის რომ თავიდან ავიცილოთ დანარჩენი თავების ძებნა და ფურცვლა, თავი1-ზე მაუსის დაჭერთ ავტომატურად გამოიტანოს ეკრანზე თავი1-ის შემცველობა; თავი2-ზე მაუსის დაჭერით – თავი2-ის შემცველობა; თავი3-ზე – თავი3-ის შემცველობა*




```
<html>
<title>sakurso</title>
<body>
<p align=center><font face=acadmtavr size=4><b> sakurso
samuSao </b></font>
</p>
<p> <font face=acadnusx size=3>
<a href=#1>Tavi1 </a>
```

```
<a href=#2>Tavi2 </a>
<a href=#3>Tavi3 </a>
</p>
<p>
<a name=1> <i>Tavi1 </i></a>
. . .
. . .
. . .
</p>
<p>
<a name=2> <i>Tavi2 </i></a>
. . .
. . .
. . .
</p>
<p>
<a name=3> <i>Tavi3 </i></a>
. . .
. . .
. . .
</p>
</body>
</html>
```

ლაბორატორიული სამუშაო

სტილების შექმნა და გამოყენება

CSS - Cascading Style Sheet

CSS ენის დახმარებით შესაძლებელია ვებ გვერდის სრული კონტროლი გაცილებით მარტივად და მეტად ფუნქციონალურად, ვიდრე ეს HTML ენის საშუალებით ხდება.

არსებობს სამი სახის სტილი:

1) INLINE STYLE SHEETS – შიდა სტილი.

შიდა სტილის გამოყენებისას სპეციალური ატრიბუტები (css თვისებები) იწერება პირდაპირ HTML ტეგში.

მაგ.

HTML-ში saqarTvelos teqnikuri universiteti

CSS-ში saqarTvelos teqnikuri universiteti

როგორც ვხედავთ შიდა სტილების გამოყენებით მივიღეთ უფრო დიდი ტეგი ვიდრე ეს HTML ენის დროს იყო. ამიტომ მათი გამოყენება ხდება მხოლოდ იმ კონკრეტული შემთხვევისათვის, როცა გვინდა ისეთი თვისების გამოყენება რაც არ არის HTML-ში და არის CSS ენაში.

მაგ. *ასოებს შორის დაშორებაა 7*

 saqarTvelos teqnikuri universiteti

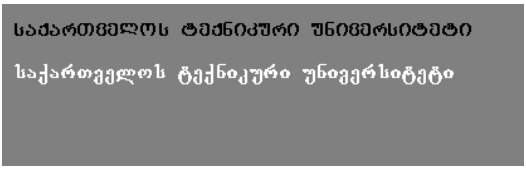
2) GLOBAL STYLE SHEETS – გლობალური სტილი.

ამ შემთხვევაში სტილის თვისებების აღწერა ხდება ვებ დოკუმენტის დასაწყისში (<body>-მდე) და გამოყენება ხდება მთლიანი დოკუმენტის ფარგლებში.

მაგ.

```
<html>
<title>gtu</title>
<style type=text/css>
   H1 {font-family: acadmtavr; font-size: 16; font-weight: bold}
   H2 {font-family: acadnusx; font-size: 16; color: white}
   body {background-color: gray}
</style>
<body>
```

```
<p align=center> <h1> saqarTvelos teqnikuri universiteti </h1>
</p>
<p align=center> <h2> saqarTvelos teqnikuri universiteti </h2>
</p>
</p>
</body>
</html>
```



3) LINKED STYLE SHEETS - დაკავშირებული სტილები.

დაკავშირებული სახის სტილის გამოყენება მიზანშეწონილია და მოსახერხებელი, როცა ვქმნით რამოდენიმე ვებ გვერდს, სადაც ვიყენებთ ერთიაიგივე სტილის ელემენტებს.

ამ შემთხვევაში სტილის აღწერა ხდება ცალკე ფაილში, რომელსაც შემდეგ ვიმახსოვრებთ css-ფორმატით ანუ file_name.css (მაგ. style.css) და შემდეგ ნებისმიერი ვებ გვერდის აღწერის დროს შეგვიძლია გამოვეყენოთ style.css ფაილში არსებული სტილები.

მაგ. თავდაპირველად ვქმნით ფაილს – **style.css** სადაც უნდა მოვახდინოთ სტილების აღწერა.

```
<STYLE type="text/css">
body {background: gray; font-size: 12pt; font-family:
      Acadnusx}
.geo {font-family: acadnusx; font-size: 16; color: blue; font-
      style: italic}
#bold {font-weight:bold}
</STYLE>
```

HTML ენაზე ვებ გვერდების შექმნის დროს სტილების აღმწერ ფაილზე (style.css) მიმართვა ხდება **<LINK rel="stylesheet" type="text/css" href="styles.css">** ტეგის საშუალებით (href ატრიბუტში მიეთითება სრული გზა).

შევქმნათ HTML ფაილი (მაგ. index.html), სადაც გამოყენებული იქნება სტილები style.css ფაილიდან

```
<html>
<title> gtu </title>
<LINK rel="stylesheet" type="text/css" href="styles.css">
<body>
<p class=geo id=bold>saqarTvelos teqnikuri universiteti </p>
</body>
</html>
```

CSS სტრუქტურა

არსებობს სტილების გამოცხადების რამოდენიმე მეთოდი:

1) **selector {property: value}**

selector არის ჩვეულებრივ HTML ელემენტი, რომელის გარკვეულ თვისებებს (property) ენიჭერა კონკრეტული მნიშვნელობები (value).

მაგ.

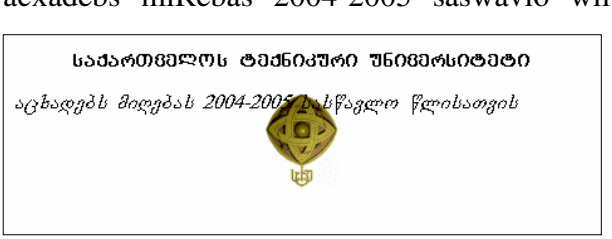
```
<html>
<title>gtu</title>
<style type=text/css>
p {text-align: center}
body {background-color: blue}
h1 {font-family: acadmtavr; font-size: 16}
</style>
<body>
<p>
<h1> saqarTvelos teqnikuri universiteti </h1>
</p>
</body>
</html>
```

2) .class_name {property: value}

კლასის სახელად შეგვიძლია მიუთითოდ ნებისმიერი სიტვა და შემდგომ HTML ელემენტში ამ კლასის გამოძახება ხდება **clas=clas_name** ატრიბუტით.

მაგ.

```
<html>
<title>gtu</title>
<style type=text/css>
.back {background-image: url(../image/pic.jpg); background-repeat: no-repaet; background-position: center center}
.title {font-family: acadmtavr; font-size: 16; font-weight:bold; text-align: center}
.geo {font-family: acadnusx; font-style: italic}
</style>
<body class=back>
<p class=title> saqarTvelos teqnikuri universiteti </p>
<p class=geo>acxadebs miRebas 2004-2005 saswavlo wlisaTvis </p>
</body>
</html>
```



3) #id_name {property: value}

თუ გვჭირდება აღწერილ კლასს დავამატოთ რომელიმე თვისება, მაშინ ვქმნით ე.წ. ინდივიდუალურ სტილს – ID, რომლის გამოყენება HTML ელემენტში ხდება **ID=id_name** ატრიბუტით.

მაგ.

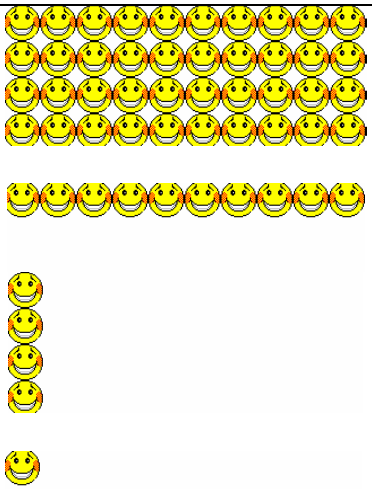
```
<html>
<title>gtu</title>
<style type=text/css>
.back {background-image: url(../image/pic.jpg); background-repeat: no-repaet; background-position: center center}
```

```
.title { font-family: acadnusx; font-size: 16; font-weight: bold; text-
align: center}
#text {text-align: left; font-weight: normal; font-style: italic}
</style>
<body class=back>
<p class=title> saqarTvelos teqnikuri universiteti </p>
<p class=title id=text>acxadebs miRebas 2004-2005 saswavlo
wlisaTvis </p>
</body>
</html>
```

CSS თვისებები

Background Properties – ფონის თვისებები

თვისება	მნიშვნელობები	აღწერა
background-color	*color name*	ფონის ფერი
background-image	url(*.jp) url(*.gif)	სურათის ჩასმა ფონად

	მაგ. url(pic.jpg)	
background-repeat	repeat repeat-x repeat-y no-repeat	
background-position	top left top center top right center left center center center right bottom left bottom center bottom right	ფონად ჩასმული სურათის მდებარეობის არჩევა  *მაგ. center center*

Fonts Properties - შრიფტის თვისებები

თვისება	მნიშვნელობები	მაგ.	აღწერა
font-family	acadusx arial ...	თბილისი Tbilisi	შრიფტია არჩევა
font-size	small medium	Tbilisi	ტექსტის

	large *რიცხვობრივი მნიშვლელობები:* მაგ: 10 12 20 30 ...	Tbilisi Tbilisi Tbilisi Tbilisi Tbilisi Tbilisi	ზომის არჩევა
font-style	normal italic	Tbilisi *Tbilisi*	ტექსტის სტილი
font-weight	normal bold bolder lighter	Tbilisi **Tbilisi** **Tbilisi** Tbilisi	ტექსტის სისქე

Text Properties - ტექსტის თვისებები

თვისება	მნიშვნელობები	მაგ.	აღწერა
color	red; #ac34bf55 ...		ტექსტის ფერი
text-decoration	none underline overline line-through	თბილისი თბილისი თბილისი ~~თბილისი~~	ხაზგასმის სტილი
text-align	left center right justify	თბილისი თბილისი თბილისი ორმხრივი გასწორება	ტექსტის მდებარეობა
text-indent	*რიცხობრივი მნიშვნელობა* *მაგ:* 20	საქართველოს ტექნიკური უნივერსიტეტი	აბზაცის დაყენება

letter-spacing	*რიცხობრივი მნიშვნელობა მაგ:* 5	თ ბ ი ლ ი ს ი	ასოებს შორის დაშორება
word-spacing	*რიცხობრივი მნიშვნელობა მაგ:* 10		სიტყვებს შორის დაშორება
text-transform	none capitalize uppercase lowercase	Georgian Technical University GEORGIAN TECHNICAL UNIVERSITY georgian technical university	

Border Properties - ჩარჩოს თვისებები

თვისება	მნიშვნელობები		აღწერა
border-color	ფერის დასახელება		ხაზის ფერი
border-width	thin medium thick *რიცხვითი მნიშვნელობა*	ვიწრო საშუალო სქელი	ხაზის სისქე
border-style	none double groove ridge inset outset		ხაზის სტილი
border-top-color border-top-width border-top-style			ზედა ხაზის ფერი სისქე სტილი
border-bottom -color border-bottom -width border-bottom –style			ქვედა ხაზის ფერი სისქე სტილი
border-left-color border-left -width			მარცხენა ხაზის ფერი სისქე

border-left –style	სტილი
border-right-color	მარჯვენა ხაზის ფერი
border-right -width	სისქე
border-right –style	სტილი

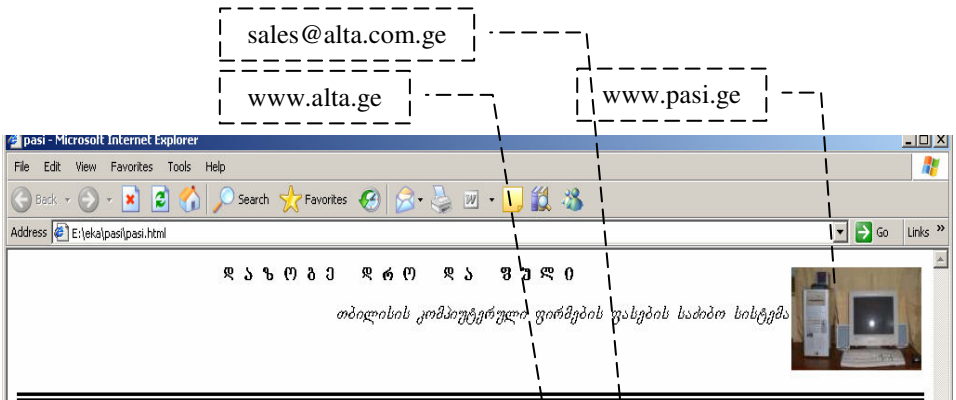
Padding Properties –უჯრის ხაზიდან ტექსტამდე დაშორება

Padding-top	*რიცხვითი მნიშვნელობა*	ზედა დაშორება
Padding-bottom	*რიცხვითი მნიშვნელობა*	ქვედა დაშორება
Padding-left	*რიცხვითი მნიშვნელობა*	მარცხენა დაშორება
Padding-right	*რიცხვითი მნიშვნელობა*	მარჯვენა დაშორება

Margin Properties - ველის თვისებები

თვისება	მნიშვნელობები	აღწერა
margin-top	*რიცხვითი მნიშვნელობა*	ზედა დაშორება
margin-bottom	*რიცხვითი მნიშვნელობა*	ქვედა დაშორება
margin-left	*რიცხვითი მნიშვნელობა*	მარცხენა დაშორება
margin-right	*რიცხვითი მნიშვნელობა*	მარჯვენა დაშორება

ლაბორატორიული სამუშაო



დაპროგრამების ენა JavaScript

JavaScript-ი დაპროგრამების ენა გამოიყენება HTML დოკუმენტების შესაქმნელად. ენის მთავარი ასპექტი არის ობიექტის ცნება.

ლიტერალები

მარტივი მონაცემები რაზედაც შეიძლება აგებული იქნას პროგრამა არის ლიტერალები. მთელი ტიპის მუდმივები შეიძლება მოცემული იქნას ათობითი თექვსმეტობითი, ან რვაობითი ფორმით.

მთელი ტიპის ათობითი მუდმივები შეიძლება წარმოდგენილი იყოს ნიშნით ან ნიშნის გარეშე მაგალითად 15,125, -156 +3546.

თექვსმეტობითი რიცხვები შეიცავენ ციფრებს 0-9-მდე და ასოებს a,b, c, d. e,f.

თექვსმეტობითი რიცხვები იწერება რიცხვის წინ სიმბოლოთი 0x, მაგალითად 0x25, 0xa1, 0xff.

რვაობითი რიცხვები შეიცავენ მხოლოდ ციფრებს 0-დან 7-ის ჩათვლით დაიწერება ასე 03, 0543. 011.

ათწილადი რიცხვების ჩასაწერად, როგორც ყველა დაპროგრამების ენაში მძიმის ნაცვლად გამოიყენება წერტილი. მაგალითად, 123. 45, -56.789.

ათწილადი რიცხვების ჩასაწერად გამოიყენება ექსპონენციალური ფორმა მაგალითად, რიცხვი 0,000000234 რომელიც შეიძლება ასე წარმოვადგინოთ $2.34x10^{-7}$ JavaScript დაპროგრამების ენაზე ჩაიწერება 2.34e-7.

ათობითი და მთელი მუდმივების გარდა არსებობს ლოგიკური მუდმივები, რომლებიც იღებენ მხოლოდ ორ მნიშვნელობას ჭეშმარიტი – true ზოგიერთ წარმოდგენაში - 1, მცდარი- false – 0.

სტრიქონული ლიტერალი წარმოადგენს სიმბოლოების თანმიმდევრობას რომელიც მოთავსებულია ‘ ‘ ან “ “ სიმბოლოებში. მაგალითად “ შედეგი” ან ’შედეგი’. ცარიელი სტრიქონული ცვლადი აღინიშნება ‘ ‘ ან “ ” სიმბოლოებით.

ცვლადი

ცვლადები გამოიყენება მონაცემების შესანახად. ცვლადების აღწერა ხდება იდენტიფიკატორის საშუალებით. იდენტიფიკატორი აუცილებლად უნდა იწყებოდეს ლათინური ალფავიტის ასოთი ან გახაზვის სიმბოლოთი , შემდეგ შეიძლება ჩაწერილი იყოს ციფრი ან გახაზვის სიმბოლო. მაგალითად test1, _my-test, test-1. ცვლადის ტიპი განისაზღვრება მათში შენახული მონაცემების ტიპით, მონაცემების ტიპის შეცვლით იცვლება ცვლადის ტიპი. ცვლადის განსაზღვრა ხდება სიტყვით var ის აუცილებლად უნდა იყოს გამოყენებული ცვლადების აღწერისას ან ინიციალიზაციისას. მაგალითად,

var t1.

აქ ცვლადის ტიპი არ არის განსაზღვრული და ის ცნობილი გახდება მას შემეგ რაც მას მიენიჭება მნიშვნელობა ეს კი ხდება მინიჭების ოპერატორის საშუალებით. მაგალითად:

var t1=345

ამ შემთხვევაში t1 ცვლადს მინიჭებული აქვს მნიშვნელობა 345.

ცვლადის მნიშვნელობა შეიძლება შეიცვალოს ასევე მინიჭების ოპერატორით. მინიჭების ოპერატორი შეიძლება გამოყენებული იყოს ნებისმიერ ადგილზე და შეუძლია შეცვალოს არა მარტო მონაცემი არამედ ცვლადის ტიპიც.

მინიჭების ოპერატორს აქვს სახე:

a=b

სადაც a- ცვლადია, b- გამოსახულება.

დასაშვებია ცვლადების შემდები აღწერა:

```
var n=3245
var x=2.89
var p=true
var s="გამოთვლა დამთავრებულია"
```

n და x ცვლადებს აქვთ რიცხვითი ტიპი, p-ს აქვს ლოგიკური ტიპი, ხოლო s-ს string.

JavaScript-ში განსაზღვრულია ასევე function ტიპი ყველა სტანდარტული ფუნქციისათვის და ასევე მომხმარებლის მიერ განსაზღვრული ფუნქციებისათვის.

JavaScript-ში ობიექტებს აქვთ მონაცემთა ტიპი object.

ცვლადების მოქმედების არე

JavaScript-ში არსებობს ორი ტიპის ცვლადები გლობალური და ლოკალური. გლობალური ცვლადი განსაზღვრულია მთელ პროგრამაში, ლოკალური ცვლადი განსაზღვრულია მხოლოდ კონკრეტული ფუნქციის შიგნით.

გამოსახულება

გამოსახულება შედგება ლიტერალებისაგან, ცვლადებისაგან, მოქმედებათა ნიშნებისაგან და ფრჩხილებისაგან. გამოსახულების გამოთვლის შემდეგ მიიღება ერთი მნიშვნელობა, რომელიც შეიძლება იყოს რიცხვი, სტრიქონი, ან ლოგიკური.

გამოსახულებაში არსებულ ყველა ცვლადი უნდა იყოს განსაზღვრული. თუ გამოსახულებაში არსებულ რომელიმე ცვლადს არ აქვს მნიშვნელობა ან არ არის განსაზღვრული მაშინ ხდება შეცდომის დაფიქსირება.

JavaScript-ში არსებობს null ლიტერალი რომელიც აღნიშნავს განუსაზღვრელ მნიშვნელობას.

ნებისმიერი გამოსახულება შედგება ოპერანდების და ოპერაციის ნიშნებისაგან. მაგალითად, გამოსახულება a*b -ში, a და b ოპერანდებია, * ოპერაციის ნიშანია.

ოპერაციები არსებობენ უნარული (ერთადგილიანი) და ბინარული (ორადგილიანი). გამოსახულება ჩაიწერება +A, თუ + - უნარუ-

ლი ოპერაციის აღნიშვნაა, ან A+B. თუ + - ბინარული ოპერაციის აღნიშვნაა.

გამოთვლილი მნიშვნელობის მიხედვით, გამოსახულება შეიძლება არსებობდეს არითმეტიკული, ლოგიკური და სტრიქონული.

არითმეტიკული გამოსახულების ჩასაწერად გამოიყენება შემდეგი ოპერატორები:

+	შეკრება
-	გამოკლება
*	გამრავლება
/	გაყოფა
%	ნაშთის მიღება გაყოფის შედეგად
++	ოპერანდის მნიშვნელობის გაზრდა 1-ით
--	ოპერანდის მნიშვნელობის შემცირება1-ით

ოპერაციების შესრულება ხდება მარცხნიდან მარჯვნივ არითმეტიკული ოპერაციების პრიორიტეტის მიხედვით.

JavaScrip-ში გამოიყენება ოპერატორების ჩაწერის შემოკლებული ფორმა:

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

JavaScript-ში გამოიყენება შედარების შემდეგი ოპერაციები. ისინი შესრულების შედეგად იღებენ მხოლოდ ორ მნიშვნელობას true თუ შედარება ჭეშმარიტია ან false წინააღმდეგ შემთხვევაში. ეს ოპერაციებია:

<	ნაკლებია
<=	ნაკლებია ან ტოლია
==	ტოლია
!=	არ არის ტოლი
>	მეტია
>=	მეტია ან ტოლია

! ოპერაცია (ლოგიკური არა) გამოიყენება ლოგიკური ტიპის ოპერანდებზე. თუ a ოპერანდის მნიშვნელობა true, მაშინ !a – მნიშვნელობა არის მცდარი, თუ a ოპერანდის მნიშვნელობა არის false , მაშინ !a - მნიშვნელობა არის ჭეშმარიტი. გარდა აღნიშნულისა ლოგიკური ოპერაციების შესასრულებლად გამოიყენება შემდეგი ლოგიკური ოპერაციები:

&& (ლოგიკური და), || (ლოგიკური ან)

მათი შესრულების შედეგი მოყვანილია ცხრილში

A	B	A&&B	A||B
true	true	true	true
true	false	false	true
false	true	false	true
false	false	false	false

სტრიქონულ მნიშვნელობებზე დასაშვებია კონკატენციის (შეერთების) ოპერაცია. ის აღინიშნება + ნიშნით.

მაგალითად,

st1= “მიმდინარე”

st2=”მდგომარეობა”

st3=st1+st2

ან

st1+=st2

მაშინ st3 მიიღებს მნიშვნელობას “მიმდინარე მდგომარეობა”

st1-იც მიიღებს მნიშვნელობას “მიმდინარე მდგომარეობა”

JavaScript-ში გამოიყენება შემდეგი ოპერატორები (ისინი მოცემულია პრიორიტეტის შემცირების მიხედვით):

ინკრემენტი	++
დეკრემენტი	--
უარყოფა	!
უნარული მინუსი	-
გამრავლება	*
გაყოფა, გაყოფა ნაშთის შენახვით	/,%
გამოკლება	-
შედარება	<,>,<=,>=
ტოლობა	==
არ არის ტოლი	!=
ლოგიკური და	&&
ლოგიკური ან	||
მინიჭება	+, +=, -=, *=, /=, %=, !=

მონაცემთა ტიპების გარდაქმნა

მონაცემთა ტიპები გავლენას ახდენს ოპერატორების შესრულებაზე, განსაკუთრებით მაშინ თუ ოპერანდებს აქვთ სხვადასხვა ტიპი. ავიღოთ მაგალითად ორი რიცხვის შეკრების მაგალითი

3+3 შედეგი იქნება 6,

მაგრამ როცა ოპერანდებს სხვადასხვა ტიპი აქვთ

3+”3” შედეგი იქნება”33”

რადგან მეორე ოპერანდის ტიპი არის სტრიქონული ე.ი შედეგი იქნება სტრიქონული ტიპის. პირველი ოპერანდი ავტომატურად გარდაიქმნება სტრიქონულ ტიპად და შედეგად მიიღება ორი სტრიქონის გაერთიანება. თუ ადგილი ექნება ასეთ შემთხვევას:
3+3+"3" მაშინ შედეგი იქნება "63".

JavaSkript ენას აქვს შესაძლებლობა სტრიქონული სიმბოლოები გარდაქმნას რიცხვით მონაცემებად. ამისათვის იყენებს ფუნქციებს: parseInt () და ParseFloat().

მაგალითად:

parseInt ("34") შედეგი იქნება 34

parseInt ("23.56") შედეგი იქნება 23

ე. ი მეორე შემთხვევაში შედეგის დამრგვალება მოხდება მთელამდე.

ParseFloat("42") შედეგი იქნება 42

ParseFloat("42.56") შედეგი იქნება 42.33

რომ გამოვიყენოთ ეს ფუნქციები იქ სადაც საჭიროა გარდაქმნა უნდა ჩავსვათ შესაბამისი ფუნქცია.

მაგალითად:

3+3+ parseInt"3" შედეგი იქნება 9.

რიცხვითი მნიშვნელობის გარდავქმნა სტრიქონულად განვიხილოთ მარტივ მაგალითებზე:

("")+2500) შედეგი იქნება2500

("")+2500). length შედეგი იქნება 4.

length ნიშნავს რომ დათვლილი იქნება სიმბოლოების რაოდენობა. მაგალითად, "2500" სიმბოლოების რაოდენობა მასში ტოლია 4-ის. length - სიგრძე , მისი ტიპი არის რიცხვი და სტრიქონი.

HTML დოკუმენტში პროგრამის ჩასმა

JavaScript-ზე დაწერილი პროგრამა შეიძლება მოთავსდეს HTML დოკუმენტში <script> და </script> ტეგებს შორის, თუ რომელიმე ტეგი გამორჩენილი იქნება პროგრამა არ გაიშვება.

<script> ტეგში შეიძლება გამოვიყენოთ სხვადასხვა პარამეტრები, ერთ-ერთი პარამეტრია language, რომელიც განსაზღვრავს პროგრამის ენას. JavaScript ენისათვის მისი მნიშვნელობაა "JavaScript". ამ ატრიბუტის გამოყენებას აქვს სახე:

<script language="JavaScript">

<script> დესკრიპტორი შეიძლება მოთავსებული იყოს ყველგან, სადაც საჭიროა. ზოგჯერ სასურველია და აზრი აქვს მის მოთავსებას შემდეგ დესკრიპტორებს შორის:

<HEAD> და </HEAD> ან <BODY> და </BODY>.

ზოგიერთი ბრაუზერი ვერ იგებებ HTML –ის ზოგიერთ ტეგებს, მაშინ ხდება მათ მიერ ასეთი ტეგების იგნორირება.

JavaScript-ზე პროგრამების დაწერისას მოხერხებულია გამოყენებული იქნას კომენტარების ოპერატორი. მას აქვს სახე <!-- . . . -->. დამხურავი ტეგის წინ ისმება // სიმბოლო.

ე.ი

```
<script language="JavaScript">
<!--
JavaScript-ენის ოპერატორები
//-->
</script>
```

შევადგინოთ მართკუთხა სამკუთხედის ფართობის გამოსათვლელი პროგრამა, როცა მოცემულია მისი კათეტები

```
<HTML>
   <HEAD>
      <TITLE> პირველი პროგრამა</TITLE>
</HEAD>
<BODY>
<P> პროგრამა      </P>
<script>
<!--
   var a=8; b= 9
document.write("მართკუთხა სამკუთხედის ფართობი
ტოლია",a*b/2,".")
//-->
</script>
</P> პროგრამის დასასრული</P>
  </BODY>
</HTML>
```

პროგრამაში გამოყენებულია ორი ცვლადი a და b. HTML გვერდზე გამოტანის ფორმირებისათვის გამოიყენება write document ობიექტში

თუ JavaScript- ს მხარს არ უჭერს ბრაუზერი, მაშინ გამოიყენება ტეგი <noscript>, ეს ტეგი მოყვება <script> და </script> ტეგებს. თუ ბრაუზერი მხარს უჭერს JavaScrip-ს მაშინ ეს ტეგი იგნორირდება.

ფუნქცია

პროგრამის შედგენის დროს გამოიყენება ლოგიკურად დამოუკიდებელი ნაწილები (ქვეპროგრამები). პროგრამის ყოველი ნაწილი შეიძლება დავყოთ ცალკეულ ქვეპროგრამად. ერთხელ შექმნილი პროგრამა შეიძლება გამოვიყენოთ მრავალჯერ, ბევრ ენაში ქვეპროგრამის გაგება რეალიზდება პროცედურების, ფუნქციის, მოდულების და ა.შ დახმარებით.

JavaScript-ში მთავარი ელემენტი არის ფუნქცია.

ფუნქცია ესაა ის სამშენებლო ბლოკი, რომელიც გამოიყენება მრავალჯერ.

ფუნქციის აღწერას აქვს სახე:

function F (V) {S}

სადაც, F ფუნქციის სახელია, რომლის საშუალებითაც ხდება მასზე მიმართვა, V- ფუნქციის პარამერტების (არგუმენტების) სიაა, ისინი ერთმანეთისგან გამოიყოფიან მძიმით; s- ფუნქციის ტანია, მისი საშუალებით მოიცემა ის შესასრულებელი მოქმედება, რომლითაც უნდა მივიღოთ შედეგი.

ფუნქციის აღწერისას უნდა გავითვალისწინოთ, რომ ის არ უნდა შეიცავდეს სხვა ფუნქციის აღწერას.

თუ ფუნქციის აღწერას აქვს სახე:

function F (v1,v2, ... vn) {s}

მაშინ ფუნქციის გამოძახებას უნდა ქონდეს სახე:

F (e1,e2, ... ,en)

სადაც, e1,e2, ... en– გამოსახულებებია რომლებიც ფაქტიურ პარამეტრებს (არგუმენტებს) განსაზღვრავენ.

v1,v2, ..., vn პარამეტრებს, რომლებიც ფუნქციის აღწერაში არის მითითებული ქვია ფორმალური პარამეტრები. თუ ფუნქციას არ აქვს პარამეტრები მაშინ მის აღწერას აქვს სახე:

function F () {s}

ფრჩხილების არსებობა გამოძახების ოპერატორში აუცილებელია ანუ ფუნქციის გამოძახებას ასეთ შემთხვევაში ექნება სახე:

F ().

შენიშვნა: ყველა განსაზღვრა და ფუნქცია უნდა შეტანილი იყოს <HEAD> განყოფილებაში.

განვიხილოთ მართკუთხა სამკუთხედის ფართობის გამოთვლის პროგრამა ფუნქციის გამოყენებით, როდესაც მოცემულია მისი კათეტები.

```
<HTML>
  <HEAD>
     <TITLE> პროგრამა ფუნქციის გამოყენებით </TITLE>
     <script language=”JavaScript”>
```

```
        <!-- //
         function ss (a,b)
         {return a*b/2 }
        // -->
      </script>
</HEAD>
<BODY>
<P> ფურცლის გამოსახვა ფუნქციით</P>
<script>
 <!- -
var a1=6; b1=9
var s= ss(a1,b1)
dokument.write ("მიიღება მნიშვნელობა ", s,".");
//-->
</script>
<P> ფურცლის ფორმირების დასასრული </P>
</BODY>
</HTML>
```

ფუნქციის ტანი შედგება მხოლოდ ერთი return ოპერატორით, რომელიც განსაზღვრავს

ფუნქციის გამოძახება ხდება ფუნქციის ტანში მინიჭების ოპერატორით :

```
s=ss(a1,b1)
```

ფორმალურ a და b პარამეტრს მიენიჭება a1-ის და b1-ის ფაქტიური მნიშვნელობა და შესრულდება ფუნქციის ტანი. მიღებული მნიშვნელობა თავსდება დოკუმენტში Writen ოპერატორით.

განხილულ შემთხვევებში მომხმარებელს არ ქონდა შესაძლებლობა შეეტანა მნიშვნელობები და მიეღო შესაბამისი შედეგი.

ფორმები

ინტერაქტიური დოკუმენტების უმრავლესობა (კავშირი Web გვერდსა და მომხმარებელს შორის) შეიძლება შექმნილი იყოს ფორმების საშუალებით. ეს ის სადგილია სადაც დამალულია ნებისმიერი ბრაუზერის ინტერაქტიური შესაძლებლობები: ტექსტური ველი, ღილაკები, გადამრთველები, სიები და ა.შ..

მომხმარებლის მოქმედება (მაგ. მაუსის ღილაკზე დაჭერა) იწვევს გარკვეულ პროცესს. ამ პროცესის დამუშავება ხდება ფორმების ელემენტებით. პროცესის დამუშავებელი პარამეტრის სახელი

იწყება on თავსართით. მაგალითად Click პროცესის დამუშავების პარამეტრს ექნება სახე onClick.

პროცესის პარამეტრები შეიძლება იყოს ოპერატორები, პროცესის მნიშვნელობები კი ფუნქციის გამოძახებები, რომლებიც უნდა შესრულდეს როდესაც ეს პროცესი გააქტიურდება.

დავუშვათ გვსურს შევქმნათ ფორმა რომელშიც ფუძე და სიმაღლის ველები გამოიყენება შესაბამისი მნიშვნელობის შესატანად. გარდა ამისა ფორმაში შევქმნათ ღილაკი გამოთვლა. ამ ღილაკზე დაჭერით მივიღებთ სამკუთხედის ფართობს. ასეთ შემთხვევაში შეიძლება გამოყენებული იყოს შემდეგი ფორმა:

```
<FORM name="form1">
ფუძე: <input type="text" size=8 name="st1"><hr>
სიმაღლე: <input type="text" size=8 name="st2"><hr>
<input tupe="button" value=გამოთვლა
                    onClick="maxval(document.form1.st1.value,
document.form1.st2.value,)">
</FORM>
```

დავწეროთ პროგრამა, რომელიც გამოთვლის სამკუთხედის ფართობს და საწყისი მნიშვნელობების შესატანად გამოვიყენოთ ფორმა.

```
<HTML>
  <HEAD>
     <TITLE> სამკუთხედის ფართობის გამოთვლა ფორმით
</TITLE>
      <script language="JavaScript">
       <!-- //
        function Fart(a,b)
        { var s=(a*b)/2;
document.write("სამკუთხედის ფართობი არის", s);
return s
}
// -->
    </script>
</HEAD>
<BODY>
<p> სამკუთხედის ფართობის გამოთვლა </P>
<FORM name="form1">
ფუძე: <input type="text" size=5 name="st1"><hr>
სიმაღლე: <input type="text" size=5 name="st2"><hr>
<input tupe="button" value=გამოთვლა
```

```
onClick="Fart (document.form1.st1.value,
document.form1.st2.value,)">
</FORM>
</BODY>
</HTML>
```

განვიხილოთ Fart ფუნქციის პარამეტრები დაწვრილებით.

HTML გვერდის ბრაუზერის მიერ ინტერპრეტაციისას იქმნება JavaScript-ის ობიექტები. ობიექტებს შორის ურთიერთკავშირს აქვს იერარქიული სტრუქტურა. იერარქიის ყველაზე მაღალ დონეზე იმყოფება Windows ობიექტი, რომელიც ბრაუზერის ფანჯარას წარმოადგენს. ის არის "მშობელი" ყველა დანარჩენი ობიექტის. ყველა ფურცელს windows ობიექტის გარდა აქვს document ობიექტი. document ობიექტის თვისებები განსაზღვრავენ თვით დოკუმენტის შემცველობას: ფონის ფერს, შრიფტის ფერს და ა.შ განხილულ მაგალითში მიღებული ფორმა წარმოადგენს document ობიექტის ნაკადებს, ხოლო ფორმის ყველა ელემენტი წარმოადგენენ ფორმა ობიექტის ნაკადებს. ობიექტზე მითითება ხორციელდება სახელით, რომელიც მოცემულია Html ტეგის **name** პარამეტრით. სამკუთხედის ფართობის მნიშვნელობის მისაღებად უნდა შესრულდეს ის კონსტრუქცია რომელიც ფორმის პირველ ველშია შეტანილი:

```
document.form1.st1.value
```

ფორმაზე მიმართვისას შეიძლება არ იყოს მითითებული დოცუმენტ ობიექტი. განხილულ მაგალითში ფორმის პირველი სტრიქონი შეიძლება ასეც იყოს ჩაწერილი:

```
form1.st1.value
```

როდესაც ფუნქციაში გადაიცემა მარტივი ტიპის მონაცემები, მაგალითად რიცხვები, როგორც განხილულ მაგალითში, პარამეტრის მნიშვნელობის გადაცემა ხორციელდება მნიშვნელობით. a ფორმალურ პარამეტრს ენიჭება ფაქტიური პარამეტრის form1.st1.value მნიშვნელობა, ხოლო b ფორმალურ პარამეტრს form1.st2.value მნიშვნელობა. ამის შემდეგ სრულდება ფუნქციის ტანი.

სიტუაცია იცვლება როდესაც ფაქტიური პარამეტრი არის ობიექტი. ამ დროს ამბობენ პარამეტრის გადაცემა ხორციელდება მითითებით ან დასახელებით

პირობითი ოპერატორი if

მას აქვს სახე

if (პირობა) {ოპერატორი ან ოპერატორები}

if საგასაღებო სიტყვაა და მისი მითითება აუცილებელია. ოპერატორი მუშაობს შემდეგნაირად :

თუ მრგვალ ფრჩხილებში მოთავსებული პირობა (მისი მნიშვნელობა ლოგიკური ტიპისაა) ჭეშმარიტია მაშინ შესრულდება ფიგურულ ფრჩხილებში ჩაწერილი ოპერატორი ან ოპერატორები, შემდეგ კი მართვა გადაეცემა ფიგურული ფრჩხილების შემდეგ ჩაწერილ ოპერატორს. თუ პირობა მცდარია მაშინ ფიგურულ ფრჩხილებში ჩაწერილი ოპერატორები იგნორირდება და მართვა გადაეცემა ფიგურული ფრჩხილების შემდეგ ჩაწერილ ოპერატორს.

პირობითი ოპერატორი if...else

მას აქვს სახე

if (პირობა) {ოპერატორი ან ოპერატორები} თუ პირობა ჭეშმარიტია

else

{ოპერატორი ან ოპერატორები} თუ პირობა მცდარია

მაგალითი: განვსაზღვროთ თებერვალში დღეების რიცხვი იმის გათვალისწინებით წელიადი ჩვეულებრივია თუ ნაკიანი.

```
var febdays
var theYear=1993
if (theYear %4==0) { febdays=29}
else
{ febdays==28}
```

განვიხილოთ JavaScript ენაზე პროგრამის შედგენის რამდენიმე მაგალითი:

1. გამოვთვალოთ სამ a, b, და c რიცხვებს შორის მაქსიმალური მნიშვნელობა.

```
<HTML>
  <HEAD>
     <TITLE> მაქსიმალური მნიშვნელობის გამოთვლა </TITLE>
     <script language="JavaScript">
      <!-- //
       function max (obj)
       {var a=Number (obj.num1.value);
var a=Number (obj.num2.value);
var a=Number (obj.num3.value);
var m=a
```

```
if (b>m)    m=b
if (c>m)     m=c
obj.res.value=m
}
// -->
    </script>
</HEAD>
<BODY>
<H4>მაქსიმალური მნიშვნელობის გამოთვლა </H4>
<FORM name="form1">
რიცხვი 1: <input type="text" size=8 name="num1"><hr>
რიცხვი 2: <input type="text" size=8 name="num2"><hr>
რიცხვი 3: <input type="text" size=8 name="num3"><hr>
მაქსიმალური მნიშვნელობა ტოლია
<input tupe="button" value=განსაზღვრა onClick="maxval(form1)">
<input type ="text" size=8 name="res"><hr>
<input type ="reset">
</FORM>
</BODY>
<HTML>
```

2. შეავდგინოთ პროგრამა რომელიც გამოითვლის მაქსიმალურ და მინიმალურ მნიშვნელობებს სამი რიცხვიდან.

```
<HTML>
  <HEAD>
     <TITLE> მაქსიმალური და მინიმალური მნიშვნელობის
გამოთვლა </TITLE>
      <script language="JavaScript">
       <!-- //
        function maxmin (obj)
  { var a=Number (obj.num1.value);
var b=Number (obj.num2.value);
var ca=Number (obj.num3.value);
var l
var t
if (a>b)   {l=b; t=a}
else
{t=b;l=a}
if(b>t) t=b
if(c<l) l=c
obj.resmax.value=t
```

```
obj.resmin.value=l

}
// -->
    </script>
</HEAD>
<BODY>
<H4>მაქსიმალური და მინიმალური მნიშვნელობის გამოთვლა
</H4>
<FORM name="form1">
რიცხვი 1: <input type="text" size=8 name="num1"><hr>
რიცხვი 2: <input type="text" size=8 name="num2"><hr>
რიცხვი 3: <input type="text" size=8 name="num3"><hr>
მაქსიმალური მნიშვნელობა ტოლია
<input tupe="button" value=gansazRvra onClick="maxminval(form1)">
<hr>
<input type ="text" size=8 name="resmax">მაქსიმალური
მნიშვნელობა<hr>
<input type ="text" size=8 name="resmin">მინიმალური
მნიშვნელობა<hr>
<input type ="reset">
</FORM>
</BODY>
<HTML>
```

3. შევადგინოთ პროგრამა რომელიც მოცემულ ოთხ a,b,c, d რიცხვებს დააალაგებს ზრდადობის მიხედვით

ალგორითმი: თავიდან ედარება ორი a და b ცვლადების მნიშვნელობები. თუ ა მეტი აღმოჩნდება b-ზე ცვლადები ცვლიან მნიშვნელობებს, შემდეგ a-ს მნიშვნელობა დარდება c-ს. თუ a>c-ზე მაშინ ცვლადები კვლავ იცვლიან მნიშვნელობებს. და ბოლოს აედარება d-ს, საჭიროების შემთხვევაში კვლავ ხდება ცვლადების მნიშვნელობების შეცვლა. ასეთნაირად მიიღება რომ a-ს მნიშვნელობა 4-დან არის მინიმალური. შემდებ b ედარება c და d-ს საჭიროებისას ცვლადები ცვლიან მნიშვნელობებს. ბოლოს c ედარება d-ს.

```
<HTML>
  <HEAD>
    <TITLE>ოთხი რიცხვის მნიშვნელობის დალაგება</TITLE>
    <script language="JavaScript">
     <!-- //
```

```
        function maxmin (obj)
  { var a=Number (obj.num1.value);
var b=Number (obj.num2.value);
var c=Number (obj.num3.value);
var d=Number (obj.num3.value);
if (a>b) {r=a; a=b; b=r}
if (a>c) {r=a; a=c;c=r}
if(a.d){r=a;a=d;d=r}
if (b>c) {r=b;b=c;c=r}
if (b>d){r=b;b=d; d=r}
if (c>d){r=c; c=d; d=r}
obj.num1.value=a
obj.num2.value=b
obj.num3.value=c
obj.num4.value=d
}
// -->
    </script>
</HEAD>
<BODY>
<H4> ოთხი რიცხვის მნიშვნელობის დალაგება ზრდადობის მიხედვით </H4>
<FORM name="form1">
რიცხვი 1: <input type="text" size=10 name="num1"><hr>
რიცხვი 2: <input type="text" size=10  name="num2"><hr>
რიცხვი 3: <input type="text" size=10 name="num3"><hr>
რიცხვი 4: <input type="text" size=10 name="num4"><hr>
მაქსიმალური მნიშვნელობა ტოლია
<input tupe="button" value=დალაგება onClick="sortval(form1)"> <hr>
<input type ="reset" value= გასუფთავება>
</FORM>
</BODY>
<HTML>
```

ციკლები

ციკლები აღნიშნავს განსაზღვრული მოქმედებების განმეორებით შესრულებას, მანამ სანამ არ შესრულდება გარკვეული პირობა.

JavaScript –ში ყველაზე ხშირად გამოყენებადი ციკლური სტრუქტურა არის for. მის სინტაქსს აქვს შემდეგი სახე:

for ([საწყისი მნიშვნელობა]; [პირობა]; [ცვლილების ბიჯი])
{ციკლის შიგნით შესასრულებელი ოპერატორები

}

კვადრატული ფრჩხილები მიუთითებს რომ მათში ჩაწერილი ოპერატორები არ არიან აუცილებელი რომ იქნან შეტანილი

საწყისი მნიშვნელობა- განსაზღვრავს ციკლის მთვლელის საწყის მნიშვნელობას;

პირობა- იგულისხმება ისეთივე გამოსახულება როგორც if ოპერატორში.

ცვლილების ბიჯი- წარმოადგენს ოპერატორს, რომელიც სრულდება ყოველჯერზე ციკლის შიგნით შესასრულებელი ოპერატორების შესრულების შემდეგ.

მასივები

JavaScript-ში მონაცემთა შენახვის და ორგანიზაციის ერთ-ერთი ყველაზე მოხერხებული ხერხია მასივი. ყველაზე მარტივი მასივი შეიძლება წარმოვიდინოთ ერთგანზომილებიანი ცხრილის სახით. ასეთი ცხრილის თითოეული სვეტი შეიცავს მონაცემებს., ამიტომ თითოეული სვეტი დანომრილია. სვეტების დანომრისათვის (მასივის ელემენტების) გამოიყენება მკაცრად განსაზღვრული რიცხვითი თანმიმდევრობა, პირველ სვეტს აქვს ნულოვანი ნომერი. სვეტების ამ ნომრებს ზოგჯერ უწოდებენ ინდექსებს. მასივის ელემენტზე მიმართვისათვის საჭიროა მასივის სახელი და მისი ინდექსის სახელი.

მასივის ელემენტები შეიძლება იყოს ნებისმიერი ტიპის, მათ შორის ობიექტებიც. JavaScript-ში ერთი მასივის ელემენტები შეიძლება იყოს სხვადასხვა ტიპის მონაცემები.

მასივი განისაზღვრება ცვლადის საშუალებით. ამიტომ მასივის შექმნისას ცვლადს ენიჭება ახალი მასივის ობიექტი. მასივები წარმოადგენენ JavaScript-ში ობიექტებს..

მასივის გამოსაცხადებლად გამოიყენება სპეციალური სიტყვა new. მისი საშუალებით მეხსიერებაში მასივის ელემენტებისათვის გამოიყოფა ადგილი. მას აქვს არააუცილებელი პარამეტრი Array (), რომელშიც შეიძლება მითითებული იყოს მასივის ელემენტების საორიენტაციო რაოდენობა. მაგალითად

var USS = new Array (51)

ამ აღწერის შედეგად მასივისათვის მეხსიერებაში გამოიყოფა ადგილი 51 ელემენტისთვის. რომ შეივსოს მასივი მის ყოველ ელემენტს უნდა მივანიჭოთ მნიშვნელობა. მასივის ელემენტზე მიმართვისათვის გამოიყენება სპეციალური ინდექსები, თავიდან იწერება მასივის სახელი, შემდეგ კვადრატულ ფრჩხილებში საჭირო ელემენტის ინდექსი. USS მასივის პირველი ელემენტი იქნება:

USS[0].

რომ მივანიჭოთ მასივის ელემენტებს კონკრეტული მნიშვნელობები უნდა გამოვიყენოთ მინიჭების ოპერატორი. მაგალითად:

USS[0]=”Arizona”
USS[1]=”Arkansas”
და ა.შ

ს ა ვ ა რ ჯ ი შ ო ე ბ ი:

1. ქვემოთ ჩამოთვლილი ცვლადებიის აღწერიდან და ინიციალიზაციიდან რომელია შეცდომა, ახსენით რატომაა შეცდომა და გაასწორეთ ის:

a. my_name =”kate”
b. var how many=25
c. var zip=document.form1.zip.value
d. var 1address=document.nameform.address1.value

2. რა მნიშვნელობებს მიიღებს ცვლადი drf ოპერატორების შესრულებისას:

var drf=2
drf=drf+2
drf=drf*10
drf=drf+ “20”
drf= “robert”

3. სადაა შეცდომა:

4. რას ნიშნავს ტერმინი გაერთიანება JavaScript -ში

სარჩევი

გამოყენებული ლიტერატურა

1. J. Neiderst "Learning Web Design" March 2001.
2. N. Heinle, M. Webb " Designing with JavaScript" June 2001.
3. М. Дмитриева JavaScript М.: 2002
4. Дэнни Гудман JavaScript М.: 2002
5. Вайк А.Р, Джиллиам Джейсон Д. JavaScript М.: 2004

კორექტორი

კომპიუტერული უზრუნველყოფა

იბეჭდება ავტორის მიერ წარმოდგენილი სახით

გადაეცა წარმოებას წ. ხელმოწერილია დასაბეჭდად

წ. ქაღალდის ზომა . პირობითი ნაბეჭდი თაბახი .

სააღრიცხვო-საგამომცემლო თაბახი ტირაჟი 100 ეგზ. შეკვეთა №

გამომცემლობა „ტექნიკური უნივერსიტეტი“, თბილისი, კოსტავას 77