საქართველოს ტექნიკური უნივერსიტეტი

თეა თოდუა, თეიმურაზ სტურუა,
ბესიკ ტაბატაძე

HTML5 & CSS3

დამტკიცებულია სახელმძღვანელოდ
საქართველოს ტექნიკური უნივერსიტეტის
სარედაქციო-საგამომცემლო საბჭოს
მიერ. 30.03.2016, ოქმი №1

თბილისი
2016

უაკ 681.3

განხილულია HTML ენის შესაძლებლობები, HTML5 ვერსიაში შემოღებული სიახლეებისა და დამატებების გათვალისწინებით, HTML ენის ელემენტარული საფუძვლებიდან დაწყებული რთული პრაქტიკული საკითხებით დამთავრებული. ასევე, მოცემულია CSS3 ვებსაიტების დამუშავების უახლესი სტანდარტები, რომლებიც ვებ-დაპროგრამების ენების ფუნქციურ შესაძლებლობებს მნიშვნელოვნად აუმჯობესებს და ინტერნეტ-პროექტებისთვის ორიგინალური ვიზუალური გადაწყვეტის საშუალებას იძლევა.

განკუთვნილია ინფორმატიკისა და მართვის სისტემების ფაკულტეტის პროფესიული სწავლებისა და ბაკალავრიატის სტუდენტებისა და HTML და CSS ენის გამოყენებით ვებგვერდების შექმნით დაინტერესებული სხვა პირებისთვის.

რეცენზენტები: პროფესორი რომან სამხარაძე,
ასოცირებული პროფესორი ეკა ჩიკაშუა

ISBN 978-9941-20-666-5

http://www.gtu.ge/publishinghouse/

შინაარსი

შესავალი

HTML (Hyper Text Markup Language) ვებგვერდების აგებისა და მისი სტრუქტურის შემუშავების ტექნოლოგიაა.

ტიმ ბერნერს ლიმ 1989 წელს შეიმუშავა ვებგვერდების შექმნის სტრუქტურა და საფუძველი დაუდო HTML ჰიპერტექსტური მარკირების ენას. ამჟამად, ტიმ ბერნერს ლი World Wide Web Consortium-ის (W3C) ხელმძღვანელია. ეს ორგანიზაცია ვებტექნოლოგიების სტანდარტების შემუშავებასა და განვითარებას უზრუნველყოფს.

HTML არ არის დაპროგრამების ენა, მისი მეშვეობით ვებგვერდის სტრუქტურის აწყობა და შემუშავება ხდება. HTML მარტივი და ასათვისებლად ადვილი ტექნოლოგიაა. HTML-ის შესასწავლად ჩვენ ერთ-ერთ პოპულარულ ტექსტურ რედაქტორს - Notepad++-ს გამოვიყენებთ. აქვე უნდა აღინიშნოს, რომ არსებობს სხვა ვებრედაქტორებიც, მაგ., Adobe Dreamweaver, EditPlus, Sublime Text და ა.შ.

HTML ენა მუდმივად იხვეწება და ივსება. დღეს HTML-ის მე-5 ვერსიაზე მიმდინარეობს მუშაობა. HTML5 ვერსიის დამუშავება W3C სამუშაო ჯგუფში 2007 წელს დაიწყო და ამჟამადაც დამუშავების პროცესშია.

HTML5-ში მრავალი სიახლე დაემატა, ამიტომ ზოგიერთი შესაძლებლობა, რომლებიც წინა ვერსიებში მხოლოდ გარე პლაგინების ან კლიენტის სკრიპტების მეშვეობით მიიღწეოდა, ახლა ჩვეულებრივი ტეგების საშუალებით მარტივად ხორციელდება.

მიუხედავად მრავალი სიახლისა, თანამედროვე ვებსაიტების აგება მხოლოდ HTML ტექნოლოგიის საშუალებით ვერ ხერხდება,

რადგანაც HTML-ს ინსტრუმენტების საკმაოდ შეზღუდული ნაკრები აქვს, რომლის საშუალებითაც ვებდიზაინერის ზოგიერთი ჩანაფიქრის განხორციელება შეუძლებელია.

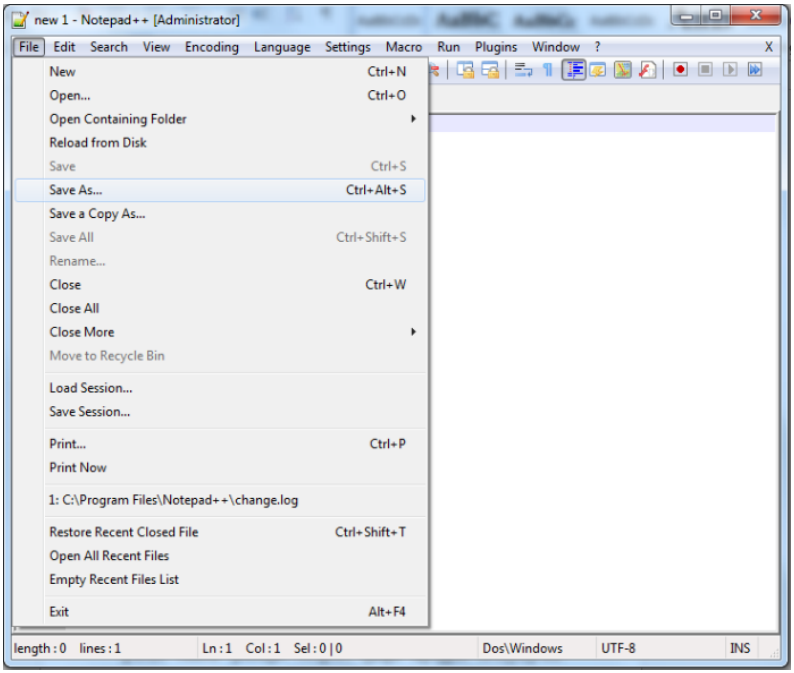
CSS (Cascading Style Sheets) - სტილების კასკადური ცხრილები HTML-ის დამატებაა და მის შესაძლებლობებს მნიშვნელოვნად აფართოებს. CSS ვებდიზაინერს საშუალებას აძლევს, ვებგვერდი უფრო ეფექტური და, დიზაინის თვალსაზრისით, ადვილად მართვადი გახადოს.

CSS3 სტილების კასკადური ცხრილების ევოლუციური გადაწყვეტაა და მისი წინა ვერსიების შესაძლებლობების გაფართოებისაკენაა მიმართული. CSS3-ში დამატებულია ბევრი საინტერესო ეფექტი, როგორიცაა მომრგვალებული კუთხეების შექმნა, ჩრდილები, გრადიენტები, გამოსახულებების გამჭვირვალობა, ტრანსფორმაციები, ანიმაცია და სხვა.

HTML დოკუმენტის შექმნა

შევქმნათ ცალკე საქაღალდე მომავალი ვებგვერდისთვის. File ⇨ New ბრძანებით გავხსნათ Notepad++-ში ახალი ფურცელი, ვიდრე მუშაობას შევუდგებოდეთ, შეგვიძლია დოკუმენტის შენახვაც. ამისათვის შევასრულოთ File ⇨ Save ან File ⇨ Save As ბრძანება.

ფაილის შენახვა Notepad++ გარემოში ნაჩვენებია ქვემოთ მოცემულია სურათზე.

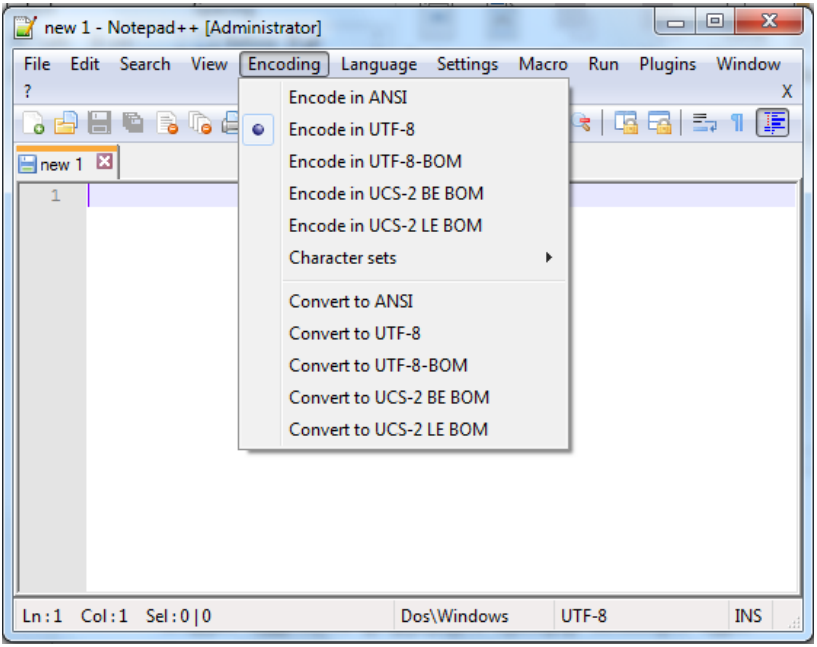


ვებსაიტი ერთმანეთთან დაკავშირებული რამდენიმე ფაილისგან შედგება, ეს ფაილები შეიძლება საკმაოდ ბევრიც იყოს. მათი ინტერნეტ-სივრცეში განსათავსებლად აუცილებელია გვქონდეს ჰოსტინგი ანუ მეხსიერების ნაწილი, რომელშიც ჩაიწერება ჩვენს ვებგვერდზე განსათავსებელი ინფორმაცია. დღეს, ვებჰოსტინგის ადმინისტრატორთა უმეტესობა გვთავაზობს ფაილების ატვირთვის მარტივ და მოსახერხებელ მეთოდებს, ჩვეულებრივ ფაილების გადატანის გზით. ისმის კითხვა: როგორ უნდა აღიქვას ბრაუზერმა, რომელი გვერდი ჩამოტვირთოს პირველად? ყველა ვებბრაუზერისათვის მიღებულია, რომ საწყისი გვერდის სახელწოდება იყოს index. ამის გამო, ფაილის დამახსოვრებისას File name ველში ჩავწეროთ index.html. ფაილის სახელი გაფართოებასთან ერთად უნდა დაიწეროს, წინააღმდეგ შემთხვევაში ის

სისტემის მიერ ნაგულისხმევი წესით შეინახება, როგორც ჩვეულებრივი .txt გაფართოების ტექსტური ფაილი. როგორც კი ფაილი .htm (.html) გაფართოებით ჩაიწერება, მისი ნიშნაკი დაუყოვნებლივ შესაბამისი ბრაუზერის ნიშნაკით შეიცვლება.

ფაილის სახელის შერჩევის შემდეგ ყურადღება უნდა მივაქციოთ კოდირების სისტემას. თუ HTML დოკუმენტში ვებგვერდზე გამოსატანად ქართული ტექსტია გამოყენებული, მაშინ Encoding ჩამოშლად მენიუში Encode in UTF-8 უნდა მოვნიშნოთ და ისე შევინახოთ მოცემული დოკუმენტი.

კოდირების არჩევის მაგალითი ნაჩვენებია ქვემოთ მოცემულ სურათზე.



HTML დოკუმენტის სტრუქტურა

HTML კოდირების პროცესში გამოიყენება სპეციალური კოდირების სისტემა, რაც გულისხმობს რომ ნებისმიერი ინფორმაცია, რომელიც HTML-ის საშუალებით ჩანს ბრაუზერში, იწერება ტეგებში.

ტეგები სპეციალური საკვანძო სიტყვებისგან შემდგარი კოდირების სისტემაა. ყველა HTML ტეგი მარცხენა კუთხური ფრჩხილით (<) იწყება და მარჯვენა კუთხური ფრჩხილით (>) მთავრდება. თითქმის ყველა ტეგი წყვილ-წყვილად არსებობს, ყოველ გახსნის ტეგს დახურვის ტეგი შეესაბამება. ასეთ ტეგებს კონტეინერები ეწოდება. ისინი შეიძლება შეიცავდეს სხვა ტეგებს და ტექსტს. ზოგიერთი ტეგი, მაგალითად, ტეგი
 არ მოითხოვს დახურვის ტეგს.

<html> და </html> ტეგები აღნიშნავს, რომ მათ შორის მდებარე სტრიქონები ერთიან HTML დოკუმენტს წარმოადგენს.

HTML დოკუმენტი სტრუქტურულად ორ ნაწილად იყოფა: სათაურის და ძირითადი ნაწილი ანუ დოკუმენტის ტანი. სათაურის ნაწილი <head> და </head> ტეგებს შორის, ხოლო ძირითადი ნაწილი <body> და </body> ტეგებს შორისაა მოთავსებული.

<head> </head> ტეგი დოკუმენტის სათაურის ნაწილის დასაწყისსა და დასასრულზე მიუთითებს. ყველაფერი, რაც მოთავსებულია <title> და </title>-ს შორის, არის დოკუმენტის სახელწოდება, რომელიც ბრაუზერის ფანჯრის სათაურის ზოლში ჩნდება. <title> ... </title> ტეგი <head> </head> ტეგებს შორის თავსდება.

<body> </body> HTML დოკუმენტის ტანის (ძირითადი ნაწილის) დასაწყისსა და დასასრულზე მიუთითებს. ამ ტეგში

იწერება ყველაფერი ის, რის განთავსებასაც ვებგვერდზე ვაპირებთ (ტექსტი, სურათები, ცხრილები და ა.შ).

მიაქციეთ ყურადღება ტეგების გახსნისა და დახურვის მიმდევრობას:

<ტეგი1> <ტეგი2> <ტეგი3>. . . </ტეგი3> </ტეგი2> </ტეგი1>

სხვა მიმდევრობით ტეგების განლაგებამ შეიძლება შეცდომა გამოიწვიოს. ტეგები შეგიძლიათ დაწეროთ როგორც დიდი, ასევე პატარა ასოებით, ბრაუზერისთვის ამას მნიშვნელობა არა აქვს.

HTML ტეგების უმეტესობას თავისი ატრიბუტები აქვს. ატრიბუტი ტეგის დამახასიათებელი რაიმე თვისებაა, ისინი ძირითადად ტეგის გამხსნელ ველშია მოთავსებული. ატრიბუტებს შეიძლება მივანიჭოთ შესაბამისი მნიშვნელობები, რომლებიც ორმაგ „ “ ან ერთმაგ ‘ ’ ბრჭყალებშია მოთავსებული. რეკომენდებულია ორმაგი ბრჭყალების გამოყენება. მაგ.:

<body text="red" bgcolor="silver">.

განხილულ მაგალითში <body> ტეგს ორი ატრიბუტი აქვს: text და bgcolor. ამ ატრიბუტების მნიშვნელობებია red და silver, რაც ნიშნავს, რომ დოკუმენტის ტექსტის ფერი წითელია, ხოლო დოკუმენტის ფონი - ვერცხლისფერი.

კოდში ცვლილებების შეტანის და შენახვის შემდეგ, F5 კლავიშზე ხელის დაჭერით, ბრაუზერის ეკრანზე შესაძლებელია განახლებული ვებგვერდის ნახვა.

HTML დოკუმენტის შენახვა და გახსნა

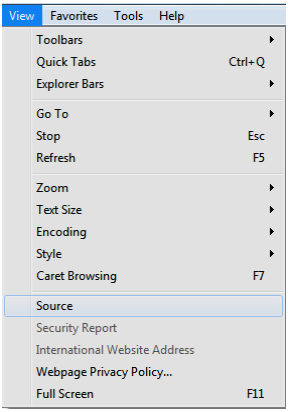
უპირველეს ყოვლისა, თქვენი მომავალი ვებგვერდისათვის ცალკე საქაღალდე შექმენით. შემდეგ გახსენით Notepad++ და მასში ტექსტი აკრიფეთ. განვიხილოთ მაგალითი:

```
<!DOCTYPE html>
<html>
 <head>
  <title> My first web page</title>
 </head>
 <body>
   Hello!
 </body>
</html
```

შეინახეთ ფაილი.

ფაილის გახსნის შედეგად ბრაუზერის ფანჯარაში მიიღება:

ცვლილებების შეტანა პროგრამაში



ბრაუზერში ვებგვერდის HTML კოდის დასათვალიერებლად View ⇨ Source ბრძანება ან <Ctrl>+<U> კლავიშთა კომბინაცია გამოიყენება.

რედაქტირებისათვის HTML ფაილი Notepad++-ში უნდა გაიხსნას. პროგრამის კოდის ცვლილებების დასრულების შემდეგ შენახვის File ⇨ Save ბრძანება ან <Ctrl>+<S> კლავიშთა კომბინაცია გამოიყენება.

ბრაუზერში ცვლილებების სანახავად View ⇨ Refresh ბრძანება ან <F5> კლავიში გამოვიყენოთ. ყველა შემდგომი ცვლილებების შემთხვევაში ოპერაცია უნდა განმეორდეს. ზოგიერთ ბრაუზერში განახლებები პროგრამაში შეიძლება სხვა გზითაც განხორციელდეს.

<!DOCTYPE> ინსტრუქცია

HTML5 სტანდარტის შესაბამისად, ვებგვერდის აგება იწყება <!DOCTYPE html> ინსტრუქციით.

<!DOCTYPE> (DTD - Document Type Definition) მოცემულ დოკუმენტში გამოყენებული HTML ენის ვერსიის შესახებ ბრაუზერისთვის მიწოდებულ ინსტრუქციას წარმოადგენს. <!DOCTYPE> პირველად HTML 2.0 ვერსიაში გამოჩნდა.

DOCTYPE ტეგს არ გააჩნია დამხურავი ტეგი, მას მხოლოდ ინფორმაციული დატვირთვა აქვს და საიტის აწყობის პროცესში არ მონაწილეობს.

META ინფორმაცია

ჩვეულებრივი ტეგების გარდა, კიდევ ე. წ. meta ტეგები არსებობს. ისინი ბრაუზერების მიერ არ გამონათდება, მაგრამ საძიებო სისტემისათვის მნიშვნელოვან დამატებით ინფორმაციას შეიცავს (მაგალითად, საკვანძო სიტყვებს, ავტორზე მიმართვას, ვებგვერდის მოკლე აღწერას და ა.შ.).

META ტეგის გამოყენების მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<title>დოკუმენტის სათაური</title>
<meta http-equiv=content-type content="text/html; charset=UTF-8">
```

```
<meta name="keywords" content="ინფორმაცია, კომპიუტერი, დაპროგრამება">
<meta name="description" content="მოცემულ ვებგვერდზე ნახავთ ინფორმაციას ვებდიზაინის შესახებ">
</head>
<body>
 ტეგების შემცველი დოკუმენტის ტანი
</body>
</html>
```

- დოკუმენტის შინაარსის აღწერა meta ტეგის საშუალებით:

```
<meta name="description" content="რესურსის აღწერა">
```

რეკომენდებულია, რომ რესურსის აღწერა მცირე ზომის გაკეთდეს (არაუმეტეს 250-300 სიმბოლოსი), მაგალითად:

```
<meta name = "description" content = "ეს გვერდი განკუთვნილია html ენის საცნობარო ინფორმაციის განსათავსებლად">
```

- საძიებო სისტემისთვის საკვანძო სიტყვების სია:

```
<meta name="keywords" content="საკვანძო სიტყვები">
```

სიაში საკვანძო სიტყვები ერთმანეთისაგან ინტერვალით ან მძიმით გამოიყოფა. ეს სიტყვები საძიებო სისტემას კატალოგში გვერდების ინდექსაციისათვის ესაჭიროება.

- დოკუმენტის ავტორი, მისი მონაცემები:

```
<meta name="author" content="ინფორმაცია ავტორის შესახებ">
```

- დოკუმენტზე საავტორო უფლების აღწერა:

```
<meta name="Copyright" content="საავტორო უფლება">
```

- მოცემული დროის გასვლის შემდეგ მითითებულ მისამართზე გადასვლა meta ტეგის http-equiv ატრიბუტის საშუალებით:

```
<meta http-equiv="Refresh" content="10; url=http://www.gtu.ge">
```

მოცემულ შემთხვევაში ბრაუზერის მიერ გვერდის ჩატვირ-

თვის შემდეგ 10 წამში მოხდება გადასვლა მისამართზე:

http://www.gtu.ge.

- დოკუმენტის შიგთავსის ტიპი და მისი კოდირება:

```
<meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
```

მოცემულ შემთხვევაში დოკუმენტის შიგთავსის ტიპად მითითებულია მნიშვნელობა text/html ანუ ტექსტი/ჰიპერტექსტი, ხოლო კოდირებისათვის - მნიშვნელობა UTF-8.

<meta> ტეგის name ატრიბუტის Viewport მნიშვნელობა მომხმარებლის მიერ ვებგვერდის დათვალიერების რეჟიმების შერჩევის საშუალებას იძლევა. დათვალიერების ფანჯრის ზომა მოწყობილობის მიხედვით იცვლება, მობილური ტელეფონებისათვის იგი უფრო პატარა იქნება, კომპიუტერის ეკრანთან შედარებით.

ვიდრე პლანშეტური კომპიუტერები და მობილური ტელეფონები შეიქმნებოდა, ვებგვერდები მხოლოდ კომპიუტერებისათვის მუშავდებოდა და ვებგვერდებს სტატიკური დიზაინი და ფიქსირებული ზომა ჰქონდა. მას შემდეგ, რაც პლანშეტურ კომპიუტერებსა და მობილურ ტელეფონებში ინტერნეტის გამოყენება დაიწყო, არსებული ვებგვერდების ზომა ახალი მოწყობილობებისათვის საკმაოდ მოუხერხებელი გახდა. ამ პრობლემის გამოსასწორებლად ბრაუზერები ვებგვერდების ზომებს ავტომატურად ამცირებდა, თუმცა ეს არ იყო საუკეთესო გამოსავალი.

HTML 5-ში ამ პრობლემის ნაწილობრივი გადაწყვეტა <meta> ტეგის საშუალებით ხდება. ამისათვის, viewport ელემენტი ყველა თქვენს ვებგვერდში უნდა ჩაწეროთ:

```
<meta name="viewport" content="width=device-width, initial-
scale=1.0">
```

ეს <meta> ტეგი ბრაუზერს აძლევს ინსტრუქციას, როგორ მართოს გვერდის ზომები და მასშტაბები.

width=device-width ნაწილი აწვდის ინსტრუქციას, რომ იმოქმედოს მოწყობილობის ეკრანის ზომის მიხედვით, ხოლო initial-scale=1.0 ნაწილი ვებგვერდის ბრაუზერში პირველი ჩატვირთვის შემთხვევაში მასშტაბირების დონეს ადგენს.

თუ დავაკვირდებით ბრაუზერში გახსნილი ფანჯრის სათაურს, შევამჩნევთ, რომ ზოგიერთი ვებგვერდის დასახელების (title) წინ პატარა პიქტოგრამა დგას. აღნიშნული პიქტოგრამის ჩასმა შემდეგი ტეგით ხორციელდება:

```
<link rel="shortcut icon" href="earth.ico">
```

href ატრიბუტით იმ გამოსახულების ფაილის მისამართი (სახელი) მიეთითება, რომელთანაც ვებგვერდი უნდა დავაკავშიროთ. rel=“shortcut icon“ ატრიბუტი href ატრიბუტში მითითებული earth.ico ფაილის ვებგვერდთან ურთიერთობას განსაზღვრავს. კერძოდ, იგი როგორც სათაურის ზოლში მყოფი ფაილი უნდა განისაზღვროს. earth.ico ფაილი იმავე საქაღალდეშია მოთავსებული, რომელშიც მოცემული ვებგვერდი.

კომენტარები

კომენტარი ეწოდება კოდის იმ ფრაგმენტს, რომლის გამოტანა/აღქმა ბრაუზერის ეკრანზე არ ხდება. როგორც ნებისმიერ ენაში, HTML-შიც, კომენტარი დოკუმენტის ტანის ნებისმიერ ადგილში და ნებისმიერი რაოდენობით შეიძლება განთავსდეს. მისი სინტაქსია:

```
<! - - ერთსტრიქონიანი კომენტარი - - >
<! მრავალსტრიქონიანი კომენტარი>
```

ფერები

HTML-ში ფერები თექვსმეტობით კოდში განისაზღვრება. ფერების გამა სამ ძირითად ფერს ეყრდნობა: წითელს, მწვანესა და ლურჯს და RGB-თი აღინიშნება. ფერი შეიძლება აღიწეროს როგორც მისი დასახელებით, ასევე მისი მნიშვნელობით RGB (Red, Green, Blue) პალიტრაში. თითოეული ფერისათვის 00-დან ff-მდე თექვსმეტობითი მნიშვნელობა მიეთითება, რასაც ათობით სისტემაში 0-255 დიაპაზონი შეესაბამება. შემდეგ ეს მნიშვნელობები ერთ რიცხვში ერთიანდება, რომელთა წინ # სიმბოლო იწერება. მაგალითად, რიცხვი #000000 - მნიშვნელობა შავ ფერს შეესაბამება, #ffffff - თეთრ ფერს, #800080 - იისფერს და ა. შ.

მაგალითად:

```
<body bgcolor="#ffffff" text="#000000" link="#9690cc">
```

მოცემული სტრიქონი დოკუმენტის ფონის თეთრ ფერს, შავ ტექსტს და ვერცხლისფერ ბმულს განსაზღვრავს.

ქვემოთ მოცემულია ზოგიერთი ფერი თავისი შესაბამისი თექვსმეტობითი კოდით:

ფერი - კოდი

შავი - Black - #000000
შინდისფერი - Maroon - #800000
მწვანე - Green - #008000
ზეთისხილისფერი - Olive - #808000
მუქი ლურჯი - Navy - #000080
იისფერი - Purple - #800080
ფირუზისფერი -Teal - #008080
ნაცრისფერი - Gray - #808080
ვერცხლისფერი - Silver - #c0c0c0
წითელი - Red - #ff0000

ღია მწვანე - Lime - #00ff00
ყვითელი - Yellow - #ffff00
ლურჯი - Blue - #0000ff
იასამნისფერი - Violet - #EE82EE
ჟოლოსფერი - Fuchsia - #ff00ff
ცისფერი - Aqua - #00ffff
თეთრი - White - #ffffff

<Body> ტეგის ატრიბუტები

<body> ტეგი ყველა იმ ინფორმაციას შეიცავს, რომლისგანაც არსებული დოკუმენტი რეალურად შედგება. <body> ტეგს შეიძლება რამდენიმე ატრიბუტი ჰქონდეს, მაგალითად:

```
<body bgcolor="#808080“ text=”yellow” leftmargin=0 topmargin=40 rightmargin=0 bottommargin=40 link=“#000099“ vlink=“#000099“ alink=“#cc0000“>
```

• bgcolor ატრიბუტი დოკუმენტში ფონის ფერს, ფერის ინტენსიურობის მოდელის RGB (Red, Green, Blue - წითელი, მწვანე, ლურჯი) თექვსმეტობითი მნიშვნელობით ან შესაბამისი ფერის სახელით განსაზღვრავს, მისი სინტაქსია:

```
<body bgcolor=”#ff0000”>
<body bgcolor=”red”>
```

• background ატრიბუტი გრაფიკულ გამოსახულებას განსაზღვრავს, რომელიც დოკუმენტის ფონს მოზაიკურად შეავსებს. მისი სინტაქსი ასეთია:

```
<body background=”(url)|(გზა) ფაილის სახელი”>
```

თუ გამოსახულება იგივე საქაღალდეშია ჩაწერილი, რომელშიც დოკუმენტია, მაშინ url-მისამართისა და გზის მითითება არ დაგვჭირდება.

• bgproperties ატრიბუტი ფონური გამოსახულების თვისე-

ბებს განსაზღვრავს. მას მხოლოდ ერთადერთი fixed მნიშვნელობა აქვს. თუ ეს ატრიბუტი მითითებულია, მაშინ რბიას გადაადგილების დროს ტექსტი გადაადგილდება, ხოლო ვებგვერდის ფონად გამოყენებული გამოსახულება უძრავად განთავსდება;

- text ატრიბუტი გამოყენებული ტექსტის ფერს განსაზღვრავს. ჩუმათობის პრინციპით, ეს ტექსტი შავია;
- topmargin ატრიბუტი გვერდის ზედა მინდვრის საზღვარს განსაზღვრავს პიქსელებში;
- bottommargin ატრიბუტი გვერდის ქვედა მინდვრის საზღვარს განსაზღვრავს პიქსელებში:
- leftmargin ატრიბუტი გვერდის მარცხენა მინდვრის საზღვარს განსაზღვრავს პიქსელებში:
- rightmargin ატრიბუტი გვერდის მარჯვენა მინდვრის საზღვარს განსაზღვრავს პიქსელებში:
- link ატრიბუტი ბმულის ფერს განსაზღვრავს, ჩუმათობის პრინციპით, უმეტეს ბრაუზერებში, ის მუქი ლურჯი ფერისაა;
- alink ატრიბუტი აქტიური ბმულის ფერს განსაზღვრავს, რომელიც მასზე მაუსის დაწკაპუნების მომენტში იცვლება. სასურველია ის link ატრიბუტით მოცემული ბმულის ფერისაგან განსხვავდებოდეს;
- vlink ატრიბუტი უკვე ნანახი ბმულის ფერს განსაზღვრავს. სასურველია, იგი ბმულის ფერისა (link ატრიბუტითაა მოცემული) და აქტიური ბმულის ფერისაგან (alink ატრიბუტითაა მოცემული) განსხვავდებოდეს.

ფონური სურათის ჩასმა <Body> ტეგისთვის

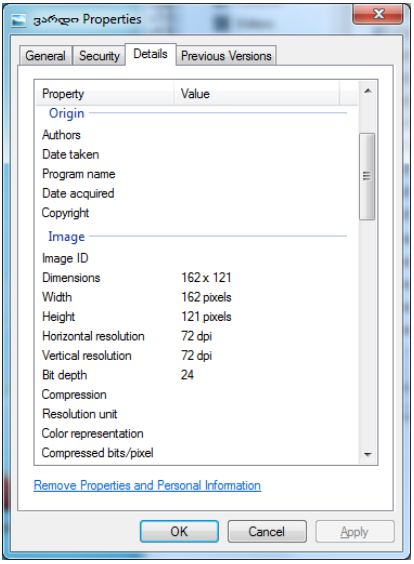
ბრაუზერების უმეტესობა დოკუმენტში ფონური სურათების განთავსების საშუალებას იძლევა, რომელიც მთელი დოკუმენტის უკანა მხარეს აისახება. უმეტეს შემთხვევაში, ფონის სახით გამოყენებული სურათი საკმაოდ ეფექტურია. ფონური სურათების აღწერა <BODY> ტეგში ხდება და ზოგადად შემდეგი სახით გამოიყურება: <body background = "picture.gif">

განვიხილოთ მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<title>ფონური გამოსახულება</title> </head>
<body topmargin="100" leftmargin="50" background="rose.jpg"
text="white">
<h1>My First Heading</h1>
<p>My first paragraph.</p>
თეა თოდუა; თეიმურაზ სტურუა;ბესო ტაბატაძე <br>
საქართველოს ტექნიკური უნივერსიტეტი <br>
საქართველოს ტექნიკური უნივერსიტეტი.
</body>
</html>
```

ჩვენ მიერ შექმნილი მარტივი ვებგვერდი ასე გამოიყურება:

დოკუმენტის ფონად არჩეულია 162×121 ზომის სურათი. მიუხედავად სურათის მცირე ზომისა, სურათმა მთლიანად დაფარა გვერდის ფონი, მოხდა ფოტოს განმეორება. ქვემოთ სურათზე ნაჩვენებია ფონად დადებული გამოსახულების ზომები.



ამ მაგალითში <body topmargin="100" leftmargin="50" background="rose.jpg" text="white"> გვიჩვენებს, რომ ტექსტი ვებგვერდის ზედა კიდიდან 100 პიქსელით, ხოლო მარცხენა კიდიდან 50 პიქსელითაა დაშორებული, ამასთან, ფონად გამოყენებულია სურათი, ხოლო ტექსტის ფერი არის თეთრი.

სათაურის დონეები

დიდი მოცულობის ტექსტები ფორმალურად, სათაურების სხვადასხვა დონის საშუალებით, შეიძლება ცალკეულ თავებად დავყოთ. სათაურის პირველი, ყველაზე მაღალი დონე ციფრი 1-ით აღინიშნება, შემდეგი დონე - 2-ით და ა.შ. ნაკლები ნომრის მქონე ტეგი უფრო მაღალი დონის სათაურს აღნიშნავს. ბრაუზერების უმეტესობას სათაურების ტეგების ექვსი დონის მხარდაჭერა აქვს. სათაურის ტეგის სინტაქსი შემდეგია:

<hx> *სათაურის x დონე* </hx>

სადაც *x* სათაურის დონის განმსაზღვრელი ციფრია 1-დან 6-მდე.

align ატრიბუტით ჰორიზონტალის მიმართ ტექსტის სწორება იმართება და left (მარცხენა კიდის მიმართ), center (ცენტრის მიმართ) და right (მარჯვენა კიდის მიმართ) მნიშვნელობები შეიძლება მიიღოს.

მაგალითი:

```
<!DOCTYPE html>
<html>
<head><title> Head </title></head>
<body>
<h1> Head 1 – Georgian Technical University </h1>
<h2> Head 2 – Georgian Technical University </h2>
<h3> Head 3 – Georgian Technical University </h3>
<h4> Head 4 – Georgian Technical University </h4>
```

```
<h5> Head 5 – Georgian Technical University </h5>
<h6> Head 6 – Georgian Technical University </h6>
</body>
</html>
```

სურათზე სხვადასხვა დონის სათაურის მაგალითია მოცემული.

აბზაცი და სტრიქონების წყვეტა

HTML დოკუმენტში ტექსტის აბზაცებად დაყოფა Enter კლავიშის გამოყენებით შეუძლებელია. შემდეგ სტრიქონზე გადასასვლელად
 ტეგით, ხოლო ტექსტის აბზაცით დასაწყებად <p> და ტეგით უნდა ვისარგებლოთ. თუ ამ ტეგებს არ გამოვიყენებთ, მაშინ დოკუმენტი მთლიანად ერთი აბზაცი იქნება.

<p> ტეგს აქვს align - ტექსტის სწორების ატრიბუტი. ამ ატრიბუტს შემდეგი მნიშვნელობების მიღება შეუძლია:

- left - ტექსტის სწორება მარცხენა კიდის მიმართ;
- right - ტექსტის სწორება მარჯვენა კიდის მიმართ;
- center - ტექსტის სწორება ცენტრის მიმართ;
- justify - ტექსტის სწორება ორივე კიდის მიმართ.

 ტეგი ბრაუზერს ახალ სტრიქონზე გადასვლას ატყობინებს.

 (ინგლ. *division* - განყოფილება) ტეგი საშუალებას გვაძლევს დოკუმენტის სტრუქტურაში რამდენიმე ნაწილი გამოვყოთ. ის კონტეინერული ტეგია და <p> ტეგის მსგავსად ფუნქციონირებს. მას შეიძლება ჰქონდეს align ატრიბუტი, რომელსაც left, center ან right მნიშვნელობების მიღება შეუძლია და ტექსტის სწორებას უზრუნველყოფს. ყოველი შემდეგი განყოფილება უარყოფს align ატრიბუტის წინა მნიშვნელობას. მისი სინტაქსია:

 განყოფილების ტექსტი

 ტეგი CSS საშუალებებთან ერთობლიობაში გამოიყენება. იგი დოკუმენტის სტრუქტურის აგების მძლავრი საშუალებაა.

ელემენტების სწორება ცენტრის მიმართ სხვადასხვა მეთოდითაც შეიძლება. მაგალითად, ეს შეიძლება <p align=center> აბზაცის ტეგით ან … ტეგით განხორციელდეს.

შრიფტის დაფორმატების ტეგები

<font> ტეგისა და მისი ატრიბუტების საშუალებით ხდება შრიფტის ზომებისა და ფერის მართვა. ეს ატრიბუტებია:

- size ატრიბუტი შრიფტის ზომას მართავს. შრიფტის ზომები 1-დან 7-მდე იცვლება. შრიფტის ზომა შეიძლება მიეთითოს როგორც ციფრით (ჩუმათობის პრინციპით იგი 3-ის ტოლია), ასევე მისი საბაზო მნიშვნელობის მიმართ, უარყოფითი ან დადებითი მიმართულებით, წანაცვლებით. ქვემოთ მოყვანილ ცხრილში შრიფტის ზომებსა და ტიპურ სიდიდეებს შორის დამოკიდებულებაა მოცემული:

შრიფტის ზომა	ტიპური სიდიდე პუნქტებში
1	8
2	10
3	12
4	14
5	18
6	24
7	36

შრიფტის ზომა შემდეგი ბრძანებითაც შეიძლება შეიცვალოს:

<font size=+|- n>

• color ატრიბუტი შრიფტის ფერის შესაცვლელად გამოიყენება. მისი სინტაქსია:

<font color="#xxxxxx">

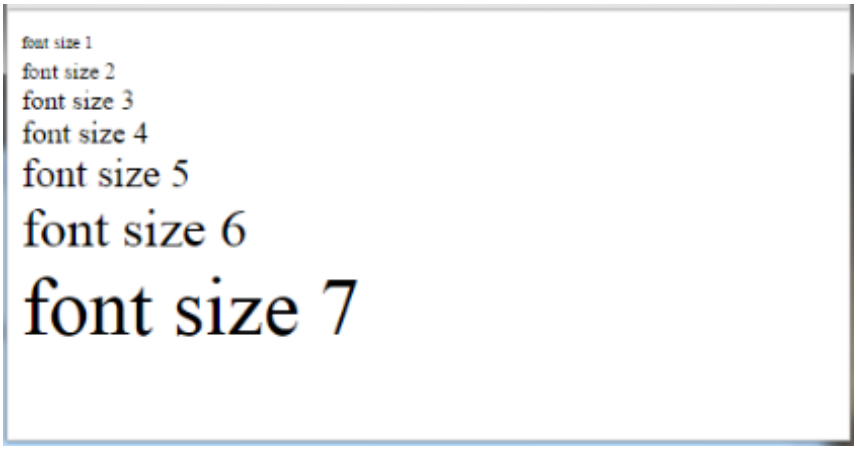
ფერი მისი დასახელებით ან RGB მოდელით უნდა მივუთითოთ, რასაც ფერის თექვსმეტობითი ფორმატი შეესაბამება.

<font> ტეგს ამ ატრიბუტების გარდა შეიძლება ჰქონდეს face ატრიბუტი, რომელშიც შეიძლება მითითებული იყოს ერთმანეთისაგან მძიმით გამოყოფილი რამდენიმე შრიფტის პრიორიტეტული სია.

მაგალითი:

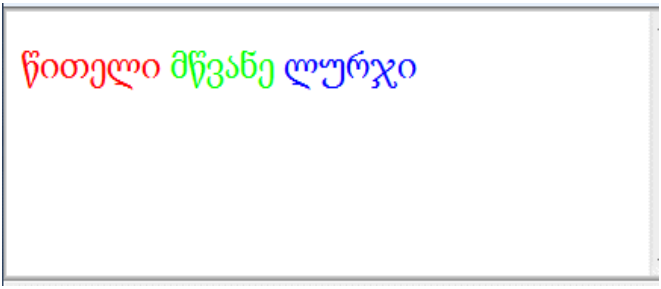
```
<!DOCTYPE html>
<html>
<head>
<title> შრიფტის ზომა  </title>
</head>
<body>
<font face = "Sylfaen", "Verdana", "Arial", "Calibri">
<font size = 1> font size 1 </font><br>
<font size = 2> font size 2 </font><br>
<font size = 3> font size 3 </font><br>
```

```
<font size = 4> font size 4 </font><br>
<font size = 5> font size 5 </font><br>
<font size = 6> font size 6 </font><br>
<font size = 7> font size 7 </font><br>
</body>
</html>
```



მაგალითი:

```
<!DOCTYPE html>
<html>
<head> <title>შრიფტის ფერი </title></head>
<body>
<font size=5 color="#ff0000"> წითელი </font>
<font size=5 color="#00ff00"> მწვანე </font>
<font size=5 color="#0000ff"> ლურჯი </font>
</body>
</html>
```



საბაზო მნიშვნელობის შეცვლა კი შემდეგი ტეგითაა შესაძლებელი:

<basefont size=n>

<basefont> ტეგი შრიფტის საბაზო ზომას მთელი დოკუმენტისათვის განსაზღვრავს. მას იგივე ატრიბუტები აქვს, რაც <font> ტეგს ჰქონდა. <basefont>-ს შესაბამისი დახურვის ტეგი არ აქვს.

მაგალითი: <basefont> ტეგის შემცველი ვებგვერდი

```
<!DOCTYPE html>
<html>
<head>
<title>შრიფტის საბაზო ზომა</title></head>
<body bgcolor=#eeeaaa>
<basefont size=8> Georgian Technical University <br>
<basefont size=6> Georgian Technical University <br>
<basefont size=5> Georgian Technical University
</body>
</html>
```

ამჟამად, მოცემული ტეგის მაგივრად CSS საშუალებების, ხოლო შრიფტის ზომის შესაცვლელად <font> ტეგის გამოყენებაა სასურველი.

წინასწარი დაფორმატება

<pre> </pre> ტეგში შეგვიძლია ჩავწეროთ წინასწარ დაფორმატებული ტექსტი, რომლის შიგნით დასაშვებია:

- სტრიქონის წყვეტის სიმბოლოების გამოყენება;
- ტაბულაციის სიმბოლოების გამოყენება (8 სიმბოლოთი მარჯვნივ გადაწევა);
- ბრაუზერის მიერ დაყენებული არაპროპორციული შრიფტის გამოყენება.

<pre> და </pre> ტეგებს შორის ჩასმული აბზაცის სხვა ტეგები იგნორირებული იქნება.

წინასწარ დაფორმატებულ ტექსტში სასურველია ტაბულაციის სიმბოლო არ იქნეს გამოყენებული, რადგანაც ამან შეიძლება ტექსტის სწორების დარღვევა გამოიწვიოს.

ტექსტის დაფორმატება

დოკუმენტში ტექსტური ინფორმაციის გამოსაყოფად შრიფტების დაფორმატების სხვადასხვა საშუალება გამოიყენება, ეს ტეგებია:

- და - მუქი ტექსტი;

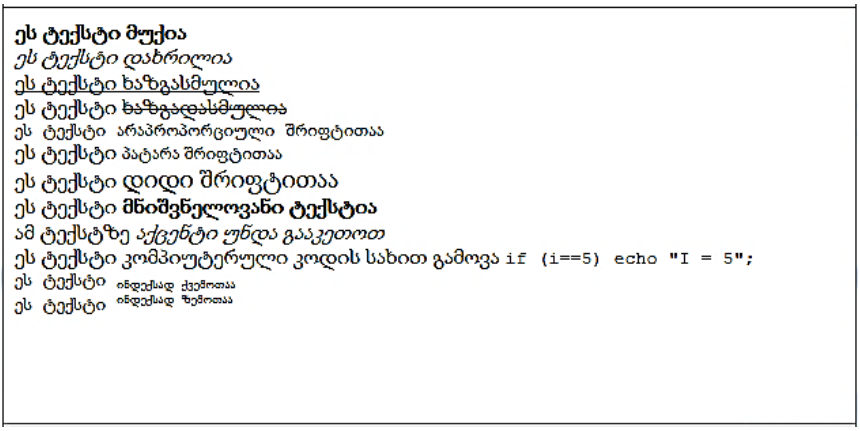
- და - დახრილი ტექსტი;
- <u> და </u> - ხაზგასმული ტექსტი;
- <s> და </s> - ხაზგადასმული ტექსტი;
- <tt> და </tt> - ტექსტი არაპროპორციული შრიფტით;
- <small> და </small> - ტექსტი პატარა შრიფტით;
- <big> და </big> - ტექსტი დიდი შრიფტით;
- და - მნიშვნელოვანი ტექსტი;
-   - აქცენტირებული ტექსტი;
- <code> და </code> - ტექსტს კომპიუტერული კოდის სახით აფორმებს;
- _{და} - ქვედა ინდექსი;
- ^{და} - ზედა ინდექსი.

მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<title> Font Style </title>
</head>
<body>
<b> ეს ტექსტი მუქია </b><br>
<i> ეს ტექსტი დახრილია </i><br>
<u> ეს ტექსტი ხაზგასმულია </u><br>
ეს ტექსტი <s> ხაზგადასმულია </s><br>
<tt> ეს ტექსტი არაპროპორციული შრიფტითაა </tt><br>
ეს ტექსტი <small> პატარა შრიფტითაა </small><br>
ეს ტექსტი <big> დიდი შრიფტითაა </big><br>
ეს ტექსტი <strong> მნიშვნელოვანი ტექსტია </strong><br>
ამ ტექსტზე <em>აქცენტი უნდა გააკეთოთ </em><br>
ეს ტექსტი კომპიუტერული კოდის სახით გამოვა <code> if
(i==5) echo "i = 5"; <code><br>
```

```
ეს ტექსტი  <sub> ინდექსად ქვემოთაა </sub><br>
ეს ტექსტი  <sup> ინდექსად ზემოთაა </sup><br>
</body>
</html>
```

მართალია, ზემოთ ჩამოთვლილი ტეგები მოძველებული არ არის, მაგრამ ტექსტის გასაფორმებლად CSS საშუალებები შეიძლება წარმატებით იქნეს გამოყენებული.



 ტეგი ტექსტის ფრაგმენტის მისი შემდგომი დაფორმატების მიზნით გამოყოფის საშუალებას იძლევა, მაგრამ ტეგისაგან განსხვავებით, ახალ აბზაცს არ იწყებს. ტეგი მომხმარებლის მიერ განსაზღვრულ ტექსტური დონის სტრუქტურას ქმნის.

 ტეგი CSS სტილებთან ერთობლიობაში გამოიყენება.

<nobr> ტეგი (ინგლ. *no break* - წყვეტის გარეშე) ბრაუზერს ბრძანებას აძლევს მთელი ტექსტი წყვეტის გარეშე ერთ სტრიქონში განათავსოს. ამ ტეგით ის ტექსტი უნდა მოინიშნოს, რომელიც აუცილებლად ერთ სტრიქონში უნდა განთავსდეს. ამგვარად, თავისი ფუნქციებით <nobr> ტეგი,
 ტეგის ფუნქციების საწინააღმდეგოა.

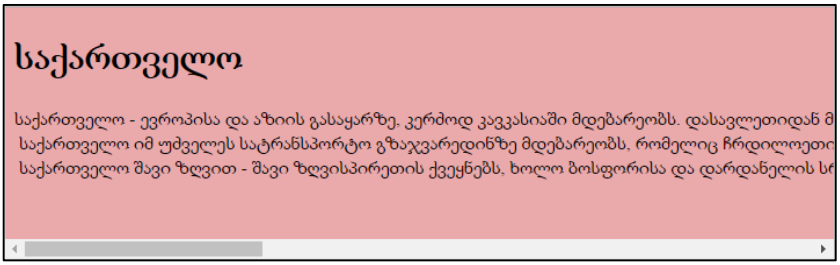
თუ ტექსტის სტრიქონს <nobr> და </nobr> ტეგებს შორის მოვათავსებთ, მაშინ მიუხედავად იმისა, ეს ტექსტი ეკრანის საზღვრებს გადასცდა თუ არა, ბრაუზერი ახალ სტრიქონზე აღარ გადავა. ამის სანაცვლოდ, გამოტანილ ფანჯარაში ჰორიზონტალური გადაფურცვლის რბია გაჩნდება.

<wbr> ტეგს დამხურავი ტეგი არ გააჩნია, იგი მხოლოდ <nobr> </nobr> ტეგის ტანში გამოიყენება და მისი ფუნქცია <nobr> </nobr> ტეგში ჩაწერილი ტექსტის ახალ სტრიქონზე გადატანაა.

მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<title>ტექსტი ერთ სტრიქონად </title></head>
<body bgcolor=#eaaaac>
<h1> საქართველო </h1>
<nobr> საქართველო - ევროპისა და აზიის გასაყარზე,
კერძოდ კავკასიაში მდებარეობს. დასავლეთიდან მას შავი ზღვა,
ჩრდილოეთით - რუსეთის ფედერაცია, სამხრეთ-აღმოსავლეთით -
აზერბაიჯანი, სამხრეთით - სომხეთი და თურქეთი ესაზღვრება.
<wbr>
საქართველო იმ უძველეს სატრანსპორტო გზაჯვარედინზე
მდებარეობს, რომელიც ჩრდილოეთისა და სამხრეთის,
დასავლეთისა და აღმოასვლეთის ქვეყნებს აკავშირებდა და
აკავშირებს. სწორედ მასზე გადიოდა ევროპა -აზიის
დამაკავშირებელი სატრანსპორტო მაგისტრალი -
ძველი აბრეშუმის გზა. <wbr>
საქართველო შავი ზღვით - შავი ზღვისპირეთის ქვეყნებს,
ხოლო ბოსფორისა და დარდანელის სრუტეებით-ხმელთაშუა
ზღვის აუზის ქვეყნებს უკავშირდება; გიბრალტარის სრუტით -
მთელს მსოფლიოსთან, ხოლო მდინარე დუნაის მეშვეობით კი -
```

```
აღმოსავლეთ და ცენტრალურ ევროპის ქვეყნებთანაა შესაძლებელი
კავშირი. </nobr>
    </body>
    </html>
```



ციტატა და ტერმინები

<blockquote> ტეგი ტექსტს როგორც დიდი ზომის ციტატას განსაზღვრავს და მარცხენა და მარჯვენა კიდიდან დაცილებულად გამოიტანს. ეს ტეგი საშუალებას იძლევა ტექსტი გვერდის ცენტრში კომპაქტურად განთავსდეს. ამ ტეგის რამდენიმეჯერ გამოყენების შემთხვევაში ტექსტი სულ უფრო ცენტრისკენ მჭიდროვდება. <blockquote> ტეგს აქვს cite=”URI” ატრიბუტი, სადაც URI ციტირებულ დოკუმენტს ან ფრაზას იძლევა. URI (Uniform Resource Identifier) - რესურსის უნიფიცირებული იდენტიფიკატორია, რომლის შედგენილობაშიც URL შედის. იგი ციტატის წყაროს მისამართს უჩვენებს. <blockquote> ტეგით მონიშნული ტექსტის გამოტანა დოკუმენტის მარცხენა გვერდიდან 8 ჰარის (ინტერვალი) დაცილებით ხდება.

<q> ტეგი ტექსტს მოკლე ციტატის სახით აფორმებს. მასაც cite=”URI” ატრიბუტი აქვს. <blockquote> ტეგთან შედარებით იგი უფრო მოკლე ციტატების გამოსატანად გამოიყენება. ჩვეულებრივ, ციტატის შედგენილობაში სტრიქონის წყვეტის სიმბოლოები არ შედის, იგი ტექსტური დონის ელემენტად ითვლება. ბრაუზერს

ციტატა ბრჭყალებით (”) გამოაქვს.

თუ დასახელებას <cite> ტეგებს შორის მოვათავსებთ, მაშინ ეს დასახელება დახრილი შრიფტით გამოვა. იგი ციტატას ან წყაროზე მიმართვას განსაზღვრავს. ტერმინები ასევე შეიძლება <dfn> ტეგებით იყოს მოცემული.

აკრონიმი

<abbr> ტეგი ტექსტის ფარგლებში აკრონიმის ანუ აბრევიატურის გამოყოფის საშუალებას იძლევა. <abbr> ტეგის შიგთავსის გამოტანა ”მცურავი” მოკარნახის სახით ხდება, რომლებიც მომხმარებელს აკრონიმის გაშიფვრის საშუალებას აძლევს.

მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<title> აკრონიმი </title>
</head>
<body>
<!-- აკრონიმი ანუ აბრევიატურა  ცნობილი დასახელებების შემოკლებაა -->
<abbr title=Georgian_Technical_University> GTU </abbr><br>
</body>
</html>
```

არ უნდა დაგვავიწყდეს, რომ კოდში ტოლობის (=) ნიშნის გარშემო ინტერვალი არ უნდა გამოვიყენოთ.

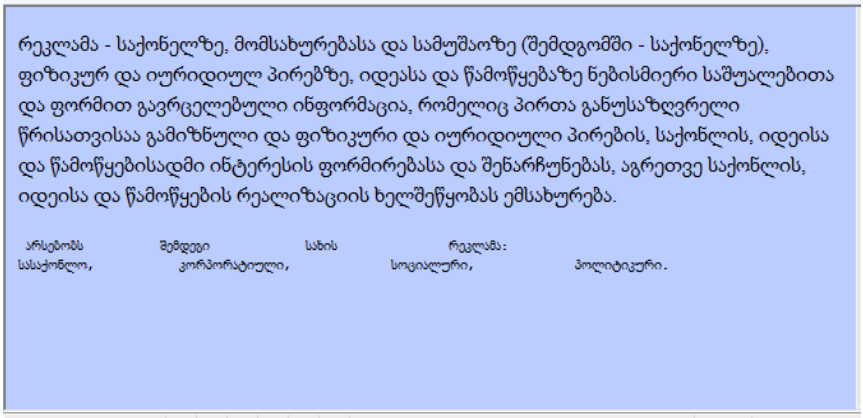
ჰარების დასმა

თუ ტექსტურ რედაქტორში ავკრეფთ ტექსტს და მასში ბევრ ჰარს გამოვიყენებთ, ბრაუზერი ამ ჰარების იგნორირებას მოახდენს და მათ მინიმუმამდე შეამცირებს, ხოლო თუ <listing> ტეგს გამოვიყენებთ, მაშინ ეს ტეგი ტექსტს განსხვავებული შრიფტით და ყველა იმ ჰარით გამოიტანს, რომლითაც იგი აიკრიფა.

მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<title>ციტატა</title>
</head>
<body bgcolor=#bbccff> რეკლამა - საქონელზე, მომსახურებასა და სამუშაოზე (შემდგომში - საქონელზე), ფიზიკურ და იურიდიულ პირებზე, იდეასა და წამოწყებაზე ნებისმიერი საშუალებითა და ფორმით გავრცელებული ინფორმაცია, რომელიც პირთა განუსაზღვრელი წრისათვისაა გამიზნული და ფიზიკური და იურიდიული პირების, საქონლის, იდეისა და წამოწყებისადმი ინტერესის ფორმირებასა და შენარჩუნებას, აგრეთვე საქონლის, იდეისა და წამოწყების რეალიზაციის ხელშეწყობას ემსახურება.
<listing> არსებობს შემდეგი სახის რეკლამა:
სასაქონლო, კორპორატიული, სოციალური, პოლიტიკური.
</listing>
```

```
</body>
</html>
```



ჰორიზონტალური ხაზები

დოკუმენტში ჰორიზონტალური ხაზების ჩასასმელად <hr> ტეგი გამოიყენება. ამ ტეგს დახურვის ტეგი არ გააჩნია. სტანდარტულად ხაზი არის მოცულობითი და ჩრდილით.

ჰორიზონტალურ ხაზებს დიდი მოცულობის ტექსტი ცალკეულ ნაწილებად შეუძლია დაყოს. მისი სინტაქსია:

<hr size=რიცხვი width=რიცხვი|პროცენტი
align=left|right|center noshade color="#xxxxxx">

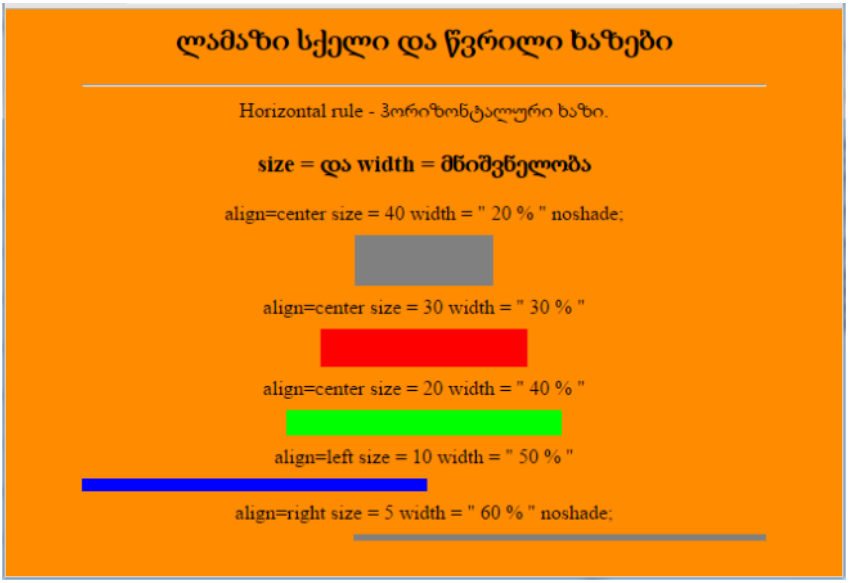
<hr> ტეგის ატრიბუტებია:

- size - ხაზის სისქე პიქსელებში;
- width - ხაზის სიგრძე პიქსელებში ან ბრაუზერის ფანჯრის სიგანის პროცენტი;
- align - ეკრანზე მდებარეობა (მარცხნივ, მარჯვნივ, ცენტრში);
- color - ხაზის ფერი;
- noshade - ხაზი ერთფერი მუქი წრფის სახით იქნება მოცემული.

მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<title>ლამაზი სქელი და წვრილი ხაზები</title>
</head>
<body bgcolor="#ff8c00" leftmargin=60>
<h2 align=center>ლამაზი სქელი და წვრილი ხაზები </h2>
<hr>
<center>
Horizontal rule - ჰორიზონტალური ხაზი.
<h3 align=center> size = და width = მნიშვნელობა</h3>
align=center size = 40 width = &amp;quot; 20 % &amp;quot; noshade;
<hr align=center width="20%" noshade size=40>
align=center size = 30 width = &amp;quot; 30 % &amp;quot;
<hr align=center width="30%" color=#ff0000 size=30>
align=center size = 20 width = &amp;quot; 40 % &amp;quot;
<hr align=center width="40%" color=#00ff00 size=20>
align=left size = 10 width = &amp;quot; 50 % &amp;quot;
<hr align=left width="50%" color=#0000ff size=10>
align=right size = 5 width = &amp;quot; 60 % &amp;quot; noshade;
<hr align=right width="60%" noshade size=5>
</body>
</html>
```

მიღებული შედეგი:



სიები

HTML დოკუმენტში ძირითადად სამი სახის სია არსებობს:

1. დანომრილი სიები;
2. მარკირებული სიები;
3. განსაზღვრებათა სია.

HTML-ში შესაძლებელია ჩადგმული სიების შექმნა. ამისათვის, უბრალოდ ერთი წყვილი ტეგი მეორე წყვილის შიგნით უნდა განვათავსოთ.

დანომრილი სიები

ბრაუზერი დანომრილ სიებში ელემენტთა ნომრებს მიმდევრობით, ავტომატურად სვამს. ეს ნიშნავს, რომ თუ სიიდან

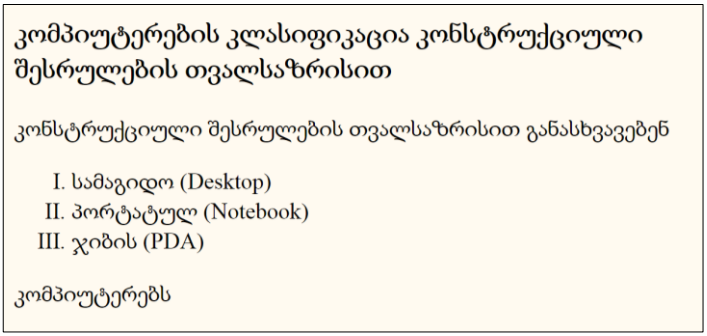
დანომრილი სიის ერთ ან რამდენიმე ელემენტს ამოვიღებთ, დანარჩენი ნომრები ავტომატურად შეიცვლება.

დანომრილი სიები წყვილი <ol> (ინგლ. *Ordered List - დანომრილი სია*) ტეგის საშუალებით იქმნება. სიის თითოეული ელემენტი <li> (ინგლ. *List Item - სიის ელემენტი*) ტეგით იწყება.

მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<title>დანომრილი სიები</title>
</head>
<body bgcolor = "floralwhite">
<h3> კომპიუტერების კლასიფიკაცია კონსტრუქციული შესრულების თვალსაზრისით </h3 >
კონსტრუქციული შესრულების თვალსაზრისით განასხვავებენ
<ol type=I>
<li>სამაგიდო (Desktop) </li>
<li>პორტატულ (Notebook) </li>
<li>ჯიბის (PDA) </li>
</ol>
კომპიუტერებს
</body>
</html>
```

მიღებული შედეგი:



 ტეგს შეიძლება type და start ატრიბუტი ჰქონდეს. მისი სინტაქსია:

<ol type=A|a|I|i|1 start=n>

სადაც type სიის სიმბოლოებია:

A - ლათინური ანბანის დიდი ასოები (A, B, C . . .);

a - ლათინური ანბანის პატარა ასოები (a, b, c . . .);

I - რომაული დიდი ციფრები (I, II, III . . .);

i - რომაული პატარა ციფრები (i, ii, iii . . .);

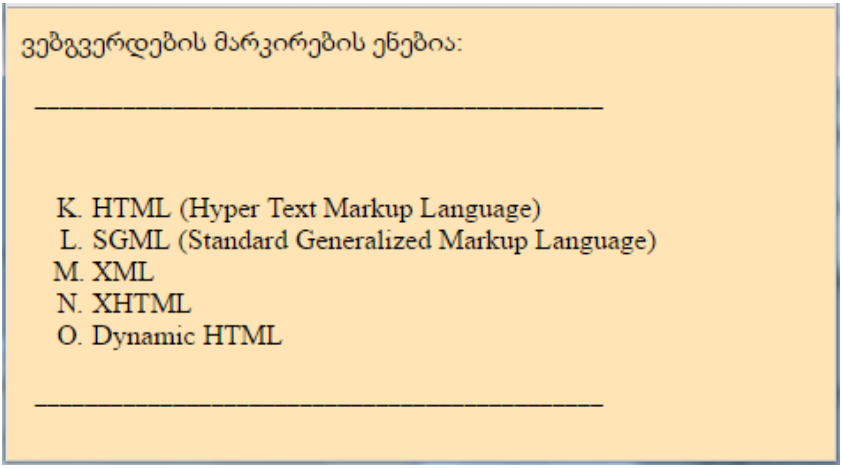
1 - არაბული ციფრები (1, 2, 3 . . .);

start=n - სიის საწყისი მნიშვნელობა.

ქვემოთ მოყვანილია მაგალითი, სადაც ლათინური ანბანის დიდი ასოებით გადანომრილი სია, ანბანის მე-11 ასოთი იწყება.

```
<!DOCTYPE html>
<html>
<head>
<title>მარკირების ენა</title>
</head>
<body bgcolor=moccasin>ვებგვერდების მარკირების ენებია:<br>
<pre> _____ </pre>
```

```
<br>
<ol type=A start=11>
<li> HTML (Hyper Text Markup Language)</li>
<li> SGML (Standard Generalized Markup Language)</li>
<li> XML</li>
<li> XHTML</li>
<li> Dynamic HTML</li>
</ol>
<pre> _____ </pre>
</body>
</html>
```



მარკირებული სიები

მარკირებული სიები <ul> (ინგლ. *Unordered List*) ტეგის საშუალებით იქმნება. სიის თითოეული ელემენტი, როგორც დანომრილი სიების შემთხვევაში, აქაც <li> (ინგლ. *List Item* - სიის ელემენტი) ტეგით იწყება. ტეგს შეიძლება type ატრიბუტი ჰქონდეს: <ul type=disc|circle|square>

type ატრიბუტი მარკერის სახეს განსაზღვრავს. თუ იგი მითითებული არ არის, მაშინ მარკირების ნიშანი ავტომატურად disc იქნება. მისი მისაღები მნიშვნელობებია:

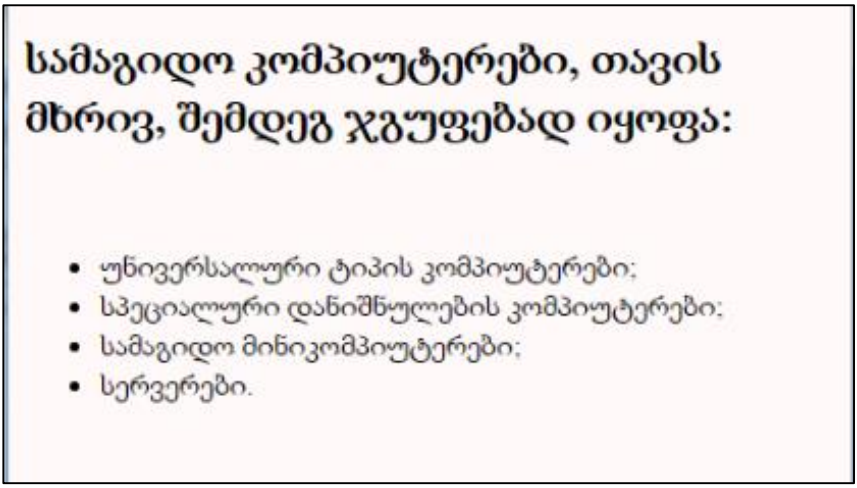
disc - გაფერადებული წრიული მარკერი;

circle - წრიული მარკერები;

square - კვადრატული მარკერები.

მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<title>მარკირებული სია</title>
</head>
<body bgcolor=snow>
<h2>სამაგიდო კომპიუტერები, თავის მხრივ, შემდეგ ჯგუფებად იყოფა:</h2>
<br>
<ul>
<li> უნივერსალური ტიპის კომპიუტერები; </li>
<li> სპეციალური დანიშნულების კომპიუტერები; </li>
<li> სამაგიდო მინიკომპიუტერები; </li>
<li> სერვერები. </li>
</ul>
</body>
</html>
```



ერთმანეთში ჩადგმული სიების მაგალითი ნაჩვენებია ქვემოთ:

```
<!DOCTYPE html>
<html>
<head>
<title>შერეული სიები</title>
</head>
<body bgcolor = "floralwhite">
<h3> კომპიუტერების კლასიფიკაცია კონსტრუქციული შესრულების თვალსაზრისით </h3 >
<ol>
<li>სამაგიდო (Desktop):</li>
<ul type=square>
<li> უნივერსალური ტიპის კომპიუტერები; </li>
<li> სპეციალური დანიშნულების კომპიუტერები; </li>
<li> სამაგიდო მინიკომპიუტერები; </li>
<li> სერვერები. </li>
```

```
    </ul>
    <li>პორტატული (Notebook):
     <ul type=circle>
    <li> დიდი ზომის პორტატული პერსონალური კომპიუტერები
(Notebook); </li>
    <li> მცირე ზომის პორტატული პერსონალური კომპიუტერები
(Netbook); </li>
    <li> პლანშეტური პერსონალური კომპიუტერები (Tablet PC).
</li>
    </ul>
    <li>ჯიბის (PDA). </li>
    </ol>
    </body>
    </html>
```

კომპიუტერების კლასიფიკაცია კონსტრუქციული შესრულების თვალსაზრისით

1. სამაგიდო (Desktop):
   - უნივერსალური ტიპის კომპიუტერები;
   - სპეციალური დანიშნულების კომპიუტერები;
   - სამაგიდო მინიკომპიუტერები;
   - სერვერები.
2. პორტატული (Notebook):
   - დიდი ზომის პორტატული პერსონალური კომპიუტერები (Notebook);
   - მცირე ზომის პორტატული პერსონალური კომპიუტერები (Netbook);
   - პლანშეტური პერსონალური კომპიუტერები (Tablet PC).
3. ჯიბის (PDA).

განსაზღვრებათა სია

განსაზღვრებათა სია სპეციალურად განლაგებული ტერმინებისა და მათი აღწერისაგან (განსაზღვრებისგან) შედგება. ვებგვერ-

დებზე განსაზღვრებების სიის შესაქმნელად <dl>, <dt> და <dd> ტეგები გამოიყენება.

განსაზღვრებათა სია <dl> (ინგლ. *Definition List*) ტეგით იწყება. იგი <dt> (ტერმინი) და <dd> (ტერმინის აღწერა) ტეგებს მოიცავს.

<dt> ტეგი (ინგლ. *Definition Term* - განმსაზღვრელი სიტყვა, ტერმინი) ბლოკური არაწყვილი ტეგია, რომელიც განსაზღვრებათა სიაში ტერმინის ტექსტს მონიშნავს.

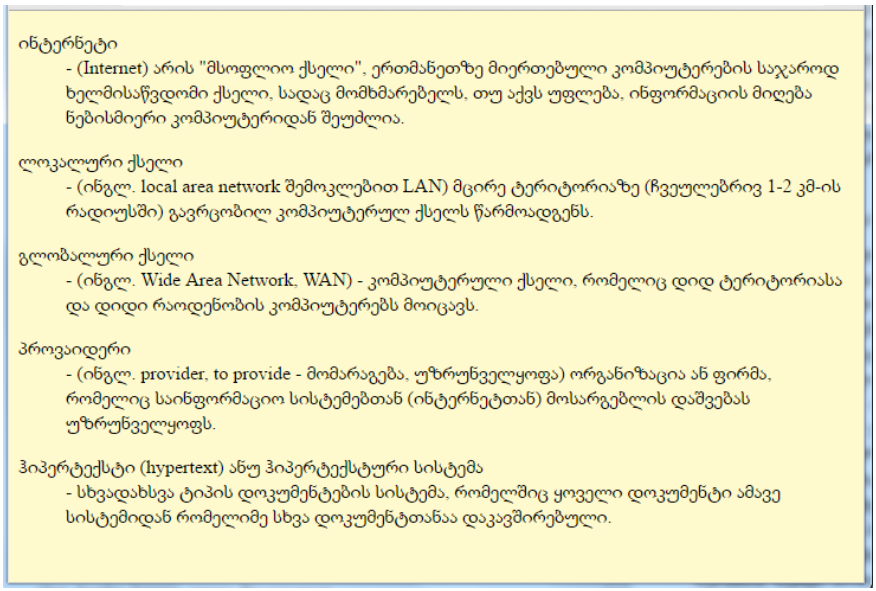
<dd> ტეგი (ინგლ. *Definition Description* - განმსაზღვრელი ტერმინის აღწერა) ბლოკური არაწყვილი ტეგია, რომელიც განსაზღვრებათა სიაში განსაზღვრების ტექსტს მონიშნავს.

მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<title>განსაზღვრებათა სია</title>
</head>
<body bgcolor=" lemonchiffon ">
<dl>
<dt>ინტერნეტი
<dd>- (Internet) არის "მსოფლიო ქსელი", ერთმანეთზე მიერთებული კომპიუტერების საჯაროდ ხელმისაწვდომი ქსელი, სადაც მომხმარებელს, თუ აქვს უფლება, ინფორმაციის მიღება ნებისმიერი კომპიუტერიდან შეუძლია. </dl>
<dl>
<dt>ლოკალური ქსელი
<dd>- (ინგლ. local area network შემოკლებით LAN) მცირე ტერიტორიაზე (ჩვეულებრივ 1-2 კმ-ის რადიუსში) გავრცობილ
```

კომპიუტერულ ქსელს წარმოადგენს. </dl>
<dl>
<dt>გლობალური ქსელი
<dd>- (ინგლ. Wide Area Network, WAN) - კომპიუტერული ქსელი, რომელიც დიდ ტერიტორიასა და დიდი რაოდენობის კომპიუტერებს მოიცავს. </dl>
<dl>
<dt>პროვაიდერი
<dd>- (ინგლ. provider, to provide - მომარაგება, უზრუნველყოფა) ორგანიზაცია ან ფირმა, რომელიც საინფორმაციო სისტემებთან (ინტერნეტთან) მოსარგებლის დაშვებას უზრუნველყოფს.
</dl>
<dl>
<dt>ჰიპერტექსტი (hypertext) ანუ ჰიპერტექსტური სისტემა
<dd>- სხვადახსვა ტიპის დოკუმენტების სისტემა, რომელშიც ყოველი დოკუმენტი ამავე სისტემიდან რომელიმე სხვა დოკუმენტთანაა დაკავშირებული. </dl>
</body>
</html>

მიღებული შედეგი:



ვებგვერდებზე გრაფიკული ელემენტის განთავსება

ვებგვერდებზე გრაფიკული ელემენტის განთავსებისათვის <img> ტეგი გამოიყენება. გრაფიკული ელემენტი დოკუმენტის შესახედაობას და ფუნქციურობას აუმჯობესებს. <img> ტეგი (ინგლ. image - გამოსახულება) პირველად HTML 2.0 ვერსიაში გამოჩნდა. მისი ატრიბუტებია: src, alt, height, width, usemap, ismap, align, border, hspace, vspace. მაგალითად,

<img src="url" alt="ტექსტი" height="სიმაღლე"
width="სიგრძე" align=top|middle|bottom|left|right ismap>.

- src ატრიბუტი (ინგლ. source - წყარო) დოკუმენტში გამოყენებული გამოსახულების ადგილმდებარეობას განსაზღვრავს. url

აუცილებელი პარამეტრია, რომელიც ბრაუზერს სურათის ადგილ-მდებარეობას უჩვენებს. სურათი გრაფიკულ ფორმატში უნდა იყოს შენახული, მაგალითად, gif ან jpg ფორმატში. თუ გრაფიკული ფაილი იმავე საქაღალდეშია, სადაც HTML დოკუმენტი, მაშინ მხოლოდ ფაილის სახელის მითითებაა საკმარისი. თუ ფაილი იმავე სერვერზე მდებარეობს, სადაც HTML დოკუმენტი, მაშინ საქაღალდის სახელიც უნდა მივუთითოთ, წინააღმდეგ შემთხვევაში სრული მისამართის მითითებაა საჭირო. მაგალითად,

<img src="http://www.uprint.com/picture/file1.gif">.

- alt აუცილებელი ატრიბუტი არ არის და ვიდრე გამო-სახულების ჩატვირთვა მიმდინარეობს, ბრაუზერს მითითებული ალტერნატული ტექსტი გამოაქვს;
- heigth აუცილებელი ატრიბუტი არ არის და იგი სურათის სიმაღლეს განსაზღვრავს პიქსელებში. თუ მოცემული პარამეტრი არ არის მითითებული, მაშინ სურათი თავისი ორიგინალური სიმაღლით გამოდის. თუმცა, ზოგიერთი ბრაუზერი მოცემულ ატრიბუტს მხარს არ უჭერს. ხშირად მონიტორის გარჩევადობა სხვადასხვა კომპიუტერზე სხვადასხვაა, რის გამოც ამ ატრიბუტის მითითების დროს ყურადღების გამოჩენაა საჭირო;
- width აუცილებელი ატრიბუტი არ არის და იგი სურათის სიგანეს განსაზღვრავს პიქსელებში;
- align აუცილებელი ატრიბუტი არ არის და იგი ეკრანზე სურათის ზუსტი პოზიციურობისთვის გამოიყენება. მისი მნიშვნე-ლობებია:
  - top - ობიექტის ზედა კიდე სტრიქონის ზედა კიდეს უსწორდება;
  - middle - ობიექტის ცენტრი სტრიქონის საბაზო ხაზს უსწორდება;
  - bottom - ობიექტის ქვედა კიდე სტრიქონის საბაზო ხაზს

უსწორდება;

- left - ობიექტის სწორება მარცხენა კიდის მიმართ მოხდება, ამასთან შესაძლებელია ობიექტის ტექსტით გარს შემოვლა;
- right - ობიექტის სწორება მარჯვენა კიდის მიმართ მოხდება, ამასთან ობიექტის ტექსტით გარს შემოვლაა შესაძლებელი.

თუ მოცემული პარამეტრი არ არის მითითებული, მაშინ ბრაუზერების უმეტესობა გამოსახულებას მარცხენივ, ხოლო ტექსტს მარჯვნივ განათავსებს.

- border ატრიბუტი გამოსახულების ჩარჩოს სისქეს გვიჩვენებს;
- vspace ატრიბუტი გამოსახულებასა და ტექსტს შორის ზემოთ და ქვემოთ ცარიელი არის ზომას მიუთითებს პიქსელებში;
- hspace ატრიბუტი გამოსახულებასა და ტექსტს შორის მარცხნივ და მარჯვნივ ცარიელი არის ზომას მიუთითებს პიქსელებში.

ქვემოთ მაგალითში პირველი სურათი გასწორებულია ვებგვერდის მარჯვენა კიდის მიმართ, ხოლო მეორე სურათი - მარცხენა კიდის მიმართ.

```
<!DOCTYPE html>
<html>
<head>
<title>გრაფიკა ტექსტით</title>
</head>
<body>
<h1>გვირილა</h1>
<img src="გვირილა.jpg" align=right width="200" heigth="225" >
<font size=4> გვირილა, რთულყვავილიანთა ოჯახის
რამდენიმე გვარის მცენარეთა კრებსითი სახელწოდებაა. უფრო
```

მეტად ამ სახელწოდებით Pyrethrum-ის გვარია ცნობილი, რომლის 100-მდე სახეობა ევრაზიის ზომიერ ზონაში და ნაწილობრივ ჩრდილოეთ ამერიკაში, უმრავლესობა კი კავკასიაში, ხმელთაშუა ზღვისპირეთსა და წინა აზიის ქვეყნებშია გავრცელებული. საქართველოში 24 სახეობაა, მათგან 10 სახეობა კავკასიის ენდემია, 3 - საქართველოსი. იგი მთის ქვემო სარტყლიდან მოკიდებული ალპურ სარტყლამდე მდელოებსა და მაღალბალახეულში, ტყისპირებსა და ბუჩქნარებში, ტენიან და დაჭაობებულ ადგილებში, ქვიან და კლდიან კალთებზე, ჩამონაზვავებზე და სხვა ადგილებში იზრდება.

<img src="ყვავილი.jpg" align=left vspace=10 hspace=10 >

ზოგი სახეობა სათიბ-საძოვრების სარეველაა, ზოგი სამკურნალოა (ბალზამური გვირილა - Pyrethrum balsamita): დალმაციური გვირილა (Pyrethrum cinerarifolium) და წითელი ანუ კავკასიური გვირილა(Pyrethrum roseum) ინსექტიციდურ და ჰერბიციდულ ნივთიერებებს (პირეტრინსა და ცინერონს) შეიცავს. Pyrethrum parthenium-ს დეკორატიულ მებაღეობაში, ხოლო ე. წ. კალუფერს ანუ კანუფერს ლიქიორების არომატიზაციისათვის იყენებენ. Doronicum-ის გვარიდან გვირილის სახელით ცნობილია ყვითელი გვირილა (Doronicum orientale) და Leucanthemum-ის გვარიდან - მინდვრის გვირილა, რომლებიც უმთავრესად კოლხეთში, ნაწილობრივ კი ქართლში მთის შუა და ზემო სარტყლის ტყეებსა და ბუჩქნარებშია გავრცელებული.

</body>

</html>

გვირილა

გვირილა, რთულყვავილოვანთა ოჯახის რამდენიმე გვარის მცენარეთა კრებითი სახელწოდებაა. უფრო მეტად ამ სახელწოდებით Pyrethrum-ის გვარია ცნობილი, რომლის 100-მდე სახეობა ევრაზიის ზომიერ ზონაში და წაწილობრივ ჩრდილოეთ ამერიკაში, უმრავლესობა კი კავკასიაში, ხმელთაშუაზღვისპირეთსა და წიწა აზიის ქვეყნებშია გავრცელებული. საქართველოში 24 სახეობაა, მათგან 10 სახეობა კავკასიის ენდემია, 3 - საქართველოსი. იგი მთის ქვემო სარტყლიდან მოკიდებული ალპურ სარტყლამდე მდელოებსა და მაღალბალახეულში, ტყისპირებსა და ბუჩქნარებში, ტეწიან და დაჭაობებულ ადგილებში, ქვიან და კლდოვან კალთებზე, ჩამოწაზვავებზე და სხვა ადგილებში იზრდება.

ზოგი სახეობა სათიბ-საძოვრების სარეველაა, ზოგი სამკურნალოა (ბალზამური გვირილა - Pyrethrum balsamita): დალმაციური გვირილა (Pyrethrum cinerarifolium) და წითელი ანუ კავკასიური გვირილა (Pyrethrum roseum) იწსექტიციდურ და ჰერბიციდულ წივთიერებებს (პირეტრინსა და ციწეროწს) შეიცავს. Pyrethrum parthenium-ს დეკორატიულ მებაღეობაში, ხოლო ე. წ. კალუფერი ანუ კაწუფერი ლიქიორების არომატიზაციისათვის იყენებენ. Doronicum-ის გვარიდან გვირილის სახელით ცნობილია ყვითელი გვირილა (Doronicum orientale) და Leucanthemum-ის გვარიდან - მიწდვრის გვირილა, რომლებიც უმთავრესად კოლხეთში, წაწილობრივ კი ქართლში მთის შუა და ზემო სარტყლის ტყეებსა და ბუჩქნარებშია გავრცელებული.

გრაფიკული ფაილის ფორმატები

გრაფიკული ფაილების ფორმატების უმეტესობა საშუალებას იძლევა, გამოსახულების შესახებ ინფორმაცია წერტილების ნაკრების სახით იყოს შენახული. ზუსტად ასევე გამოსახულება გამოდის ეკრანზეც.

ჩვეულებრივ ინტერნეტში ორი რასტრული ფორმატი - jpeg და gif გამოიყენება. გარდა ამისა, გამოიყენება ასევე bmp და pcx ფორმატის გამოსახულებებიც. სპეციალისტებს მიაჩნიათ, რომ მომავალში საკმად დიდი გამოყენების პერსპექტივა აქვს ასევე png ფორმატსაც,

GIF (Graphic Interchange Format - გრაფიკული მონაცემების მიმოცვლის ფორმატი) ფორმატი რასტრული გრაფიკული გამოსახულებების ჩასაწერად და შესანახად გამოიყენება. ეს ფორმატი CompuServe კორპორაციის მიერ იქნა დამუშავებული 1987 წელს. gif ფორმატი 256-ფერიანი გამოსახულებების შესანახად გამოიყენება.

JPEG (Joint Photographic Expert Group - ფოტოგრაფიის სფე-

როს ექსპერტების გაერთიანებული ჯგუფი) ფორმატი ისეთი სურათებისათვის გამოიყენება, სადაც 24-ბიტიანი პალიტრაა გამოყენებული, რაც 16,7 მილიონი ფერის გამოსახვის საშუალებას იძლევა.

JPEG - ერთ-ერთი ყველაზე მძლავრი ალგორითმია. პრაქტიკულად იგი მრავალფერიანი გამოსახულებებისათვის დე-ფაქტო სტანდარტს წარმოადგენს. jpeg ფორმატი gif ფორმატის ნაკლოვანი მხარის შესავსებად იყო შექმნილი. ეს ფორმატი HTML დოკუმენტებისა და გრაფიკული გამოსახულებების ქსელის საშუალებით გადასაცემად ფართოდ გამოიყენება. ფაილებს, რომლებიც შეიცავს jpeg მონაცემებს, ჩვეულებრივ აქვს .jpg, .jfif, .jpe ან .jpeg გაფართოება. თუმცა, მათ შორის ყველაზე პოპულარულია .jpg.

PNG (Portable Network Graphics) — ციფრული სურათების ღია ფორმატია, PNG არის გრაფიკული გამოსახულების ფორმატი, რომელიც სპეციალიზებულია მარტივი, ერთგვაროვანი ფერების მქონე სურათების შექმნაზე.

PNG ფორმატის უპირატესობა გამჭვირვალობის ეფექტის შექმნაა, რაც გულისხმობს გრაფიკული ობიექტისთვის ფონის მოშორებას, ასევე აღსანიშნავია, რომ PNG ფორმატის ფოტოები გამოირჩევა თავისი ხარისხით, ამის გამო ვებდიზაინში ეს ფორმატი ხშირად გამოიყენება.

პიქსელები და გარჩევადობა

სტატიკური რასტრული გამოსახულება ორგანზომილებიანი რიცხვითი მასივია. ამ მასივის ელემენტებს პიქსელები (ინგლ. pixel - picture element, გამოსახულების ელემენტი) ეწოდება.

ინტერნეტში გამოსახულების გადასაცემად ძირითადად რასტრული გრაფიკა გამოიყენება. რასტრული სურათი წარმოადგენს პიქსელების ბადეს, მათი ფერების სხვაობა ქმნის გრაფიკულ

გამოსახულებას, რომელიც ადამიანის თვალის მიერ აღიქმება, როგორც ერთიანი სურათი. რასტრული გრაფიკა არის გამოსახულების ფორმატი, რომელიც შეიცავს ინფორმაციას პიქსელების მდგომარეობის, რაოდენობის და ფერის შესახებ.

რასტრული გრაფიკული გამოსახულებისათვის მნიშვნელოვანი მახასიათებელია მისი გარჩევადობა. ერთი და იგივე სურათი შეიძლება კარგი ან ცუდი ხარისხით იყოს წარმოდგენილი, იმის მიხედვით, სიგრძის ერთეულში მას რამდენი წერტილი შეესაბამება.

გარჩევადობა - ზომის ერთეულში წერტილების რაოდენობა:

- dpi (dots per inch) - წერტილების რაოდენობა დუიმში;
- ppi (points/pixels per inch) - პიქსელების რაოდენობა დუიმში;

dpi – ყველაზე ხშირად გამოყენებული ტერმინი, რომელიც ინფორმაციის გამომტანი მოწყობილობების მიერ გამოტანილ პატარა წერტილს შეესაბამება. ამ წერტილს სურათის გარჩევადობის მაჩვენებელთან საერთო არაფერი არ აქვს.

ილუსტრაციის გარჩევადობა ჩვეულებრივ dpi-ში იზომება. რაც მეტია გარჩევადობა, მით უფრო მაღალი ხარისხისაა გამოსახულება, მაგრამ ამასთან იმ ფაილის მოცულობა, რომელშიც მოცემული გამოსახულება ინახება, დიდია.

არსებობს გამოსახულების, ეკრანისა და პრინტერის გარჩევადობა. ხარისხიანი დიზაინისათვის მაღალი გარჩევადობის მქონე მონიტორია საჭირო. ზოგჯერ დიზაინერს გამოსახულების რედაქტირება შეიძლება წერტილების მიხედვით დასჭირდეს. მონიტორის გარჩევადობის მნიშვნელობა მონიტორსა და კომპიუტერის ვიდეოადაპტერზეა დამოკიდებული.

ბმულები

ბმულები ერთ-ერთი მნიშვნელოვანი კომპონენტია, იგი ვებგვერდებს მომხმარებლისათვის უფრო საინტერესოს ხდის. ბმულების ანუ როგორც ზოგჯერ მას უწოდებენ ჰიპერბმულების დახმარებით ვებგვერდის სტრუქტურირება და მისი სხვა გვერდებთან ან იმავე დოკუმენტის სხვა ნაწილებთან დაკავშირება ხდება, რაც ინფორმაციის სწრაფ და მოსახერხებელ მიღებას უზრუნველყოფს. ინტერნეტს დიდი პოპულარობა სწორედ ბმულებმა მოუტანა, რადგან მისი დახმარებით მომხმარებელს ძალზე მარტივად, მაუსის უბრალო დაწკაპუნებით ერთი გვერდიდან მეორეზე გადასვლა შეუძლია.

ბმულების დოკუმენტში ჩასმა

HTML ენაში ბმულების ორგანიზებისათვის რესურსის უნივერსალური მაჩვენებელი (Uniform Resource Locator, URL) გამოიყენება. მაგალითად, http://www.gtu.ge//.

ბმულების დოკუმენტში ჩასასმელად <a> ტეგი გამოიყენება. <a> ტეგს შემდეგი ატრიბუტები აქვს: href, name, accesskey, target.

ამ ატრიბუტის გამოყენების დროს ნებისმიერი ტექსტი, რომელიც <a> და ტეგებს შორის არის მოთავსებული ვებბრაუზერის მიერ სპეციალურად გამოიყოფა (ჩვეულებრივ სტანდარტულად ხაზგასმული გამოდის და ლურჯი ფერითაა გამოყოფილი). როდესაც მაუსის მაჩვენებელს რომელიმე ბმულს მივუახლოებთ, იგი შეიცვლება და ხელის ფორმას მიიღებს.

- ატრიბუტი href აუცილებლად უნდა იყოს მითითებული. მის მნიშვნელობას იმ დოკუმენტის URL-მისამართი წარმოადგენს, რომელზეც ბმული მიუთითებს:

```
<a href="URL">
```

მაგალითად:

 გადასვლის ლინკი , ტექსტზე დაჭერის შემდეგ მომხმარებელი test.html (ძირითადი და გადასასვლელი ფაილი ერთ საქაღალდეში უნდა იყოს განთავსებული) გვერდზე გადავა.

მიმართვა HTML ფაილზე HTTP პროტოკოლის გამოყენებით: გადასვლის ლინკი , ტექსტზე დაჭერის შემდეგ მომხმარებელი გადავა www.gtu.ge ტექნიკური უნივერსიტეტის საიტზე.

- ატრიბუტი name ატრიბუტ href-ის ნაცვლად დოკუმენტში შიგა ბმულების დროს მიეთითება.
- ატრიბუტი accesskey საშუალებას გვაძლევს ე. წ. ”ცხელი კლავიში” მივუთითოთ, რომელზეც ხელის დაჭერით ბმულზე გადასვლა მოხდება. მისი სინტაქსია:

```
<a href = “ http://www.gtu.ge//” accesskey="კლავიშის სახელი">
ბმული</a>
```

ბმული, რომელიც ელექტრონული ფოსტით შეტყობინების მითითებულ მისამართზე გაგზავნის საშუალებას იძლევა:

```
<a href="mailto:name@gmail.com"> ბმული</a>
```

ბმული, რომელიც ელექტრონული ფოსტით ავტომატურად შეყვანილი შეტყობინების თემისა და შეტყობინების ტექსტის გაგზავნის საშუალებას იძლევა:

 ბმული.

ატრიბუტი target განსაზღვრავს ჩასატვირთი URL მისამართის ბრაუზერში ჩატვირთვის ტიპს. მისი მნიშვნელობებია: _blank, _new, _parent, _self, _top.

blank, new მნიშვნელობების შემთხვევაში URL მისამართი ჩაიტვირთება ბრაუზერის ახალ ფანჯარაში.

parent, self, top მნიშვნელობის შემთხვევაში URL მისამართი ბრაუზერის იმავე ფანჯარაში ჩაიტვირთება.

მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>example10</title>
</head>
<a href="http://gtu.ge" target="_blank"> Georgian Technical University </a>
<body>
</body>
</html>
```

ბმულზე გადასვლის შემდეგ მითითებული ვებგვერდი ახალ ფანჯარაში გაიხსნება.

შიგა ბმულები

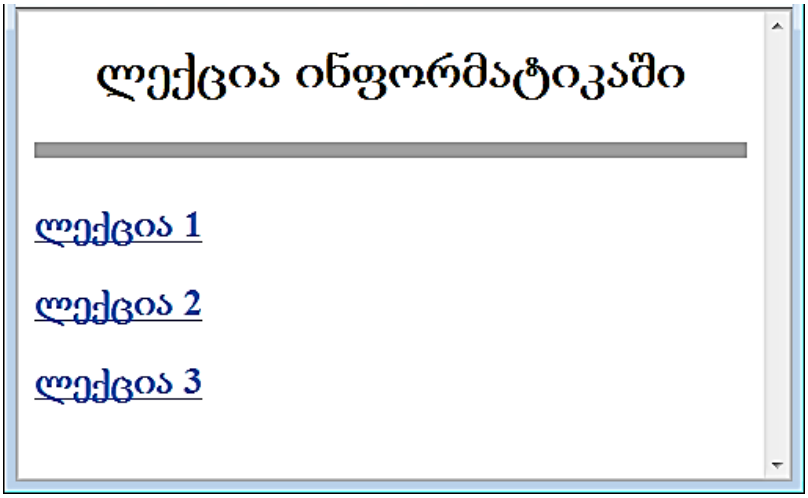
შიგა ბმულები მიმდინარე ვებსაიტზე განთავსებულ სხვა ვებგვერდებზე მიუთითებს. ამიტომ, შიგა ბმულების შექმნის დროს <a> ტეგის href ატრიბუტში ძებნის ფარდობითი მისამართის მითითებაა საკმარისი.

ფარდობით მისამართში ათვლის წერტილი ის საქაღალდეა, რომელშიც ეს ფაილია მოთავსებული, ამიტომაც თუ ფაილები ერთ საქაღალდეშია, საკმარისია მხოლოდ ფაილის სახელის მითითება.

მაგალითი:

```
<!DOCTYPE html>
<html>
```

```
<head>
<title>ლექცია ინფორმატიკაში</title></head>
<body background=#ffffff>
    <h1 align=center>ლექცია ინფორმატიკაში</h1>
    <hr noshade size=10 width="100%" align=center>
    <p><h2><a href="lecture1.htm">ლექცია 1</a></p>
    <p><a href=" lecture2.htm">ლექცია 2</a></p>
    <p><a href=" lecture3.htm">ლექცია 3</a></p></h2>
</body>
</html>
```



გარე ბმულები

გარე ბმულები იმ ვებგვერდებზე მიუთითებს, რომლებიც მიმდინარე ვებსაიტზე არ არის განთავსებული. ამიტომ, გარე ბმულების შექმნის დროს <a> ტეგის href ატრიბუტში URL-მისამართის სრულად მითითებაა აუცილებელი.

მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<title>ლექცია ინფორმატიკაში</title>
</head>
<body background="grey">
<h1 align=center>ლექცია ინფორმატიკაში </h1>
<hr align=center width="100%" noshade size=10>
<p> <a href="https://www.google.ge/?gws_rd=ssl"> www.google.ge
</a></p>
<p><a href=" http://gtu.ge/ ">www.gtu.ge </a> </p>
</body>
</html>
```

დოკუმენტის შიგნით გადასვლა

როგორც უკვე აღვნიშნეთ, ბმულები შეიძლება როგორც სხვა დოკუმენტებზე, ასევე ერთი და იგივე დოკუმენტის ცალკეულ ფრაგმენტებზე გაკეთდეს. ეს დოკუმენტის შიგნით ერთი ფრაგმენტიდან მეორეზე სწრაფად გადასვლის საშუალებას იძლევა. ასეთი გადასვლების ორგანიზებისათვის სპეციალური ღუზები გამოიყენება, რომლებიც ჩვეულებრივ ფრაგმენტის პირველ სტრიქონში ან სათაურში განთავსდება.

ღუზის ჩასმისათვის ისევ <a> ტეგი გამოიყენება, მაგრამ href ატრიბუტის ნაცვლად მასში name ატრიბუტი უნდა მიეთითოს. ამ ატრიბუტის მნიშვნელობა ღუზის სახელია. ის მხოლოდ ლათინური ასოებისა და ციფრებისაგან უნდა შედგებოდეს და ჰარს არ უნდა შეიცავდეს, მაგალითად:

```
<a name="ღუზის_სახელი"> დოკუმენტის შიგნით გადასვლა </a>
```

დაყენებულ ღუზაზე გადასასვლელად URL-მისამართის ბოლოს ღუზის სახელის მითითებაა აუცილებელია, რომლის წინ # სიმბოლო იქნება ჩაწერილი:

```
<a href=”#ღუზის სახელი”> ბმული </a>
```

ბმული შეიძლება იყოს როგორც ტექსტი, ასევე გამოსახულება. ამასთან, HTML ენაზე პროგრამის კოდი და გამოსახულება ერთ საქაღალდეში უნდა იყოს განთავსებული.

გამოსახულების რუკა

გრაფიკული გამოსახულების რუკა (image maps) საშუალებას იძლევა გამოსახულების ცალკეული ნაწილები სხვა დოკუმენტებს ბმულით დავუკავშიროთ. გამოსახულების ცალკეულ ნაწილებზე ე.წ. აქტიურ არეებზე (hot spots) მაუსით დაწკაპუნების შემთხვევაში, მომხმარებელს ამა თუ იმ მოქმედების შესრულება და სხვა დოკუმენტებზე გადასვლა შეუძლია.

თუ მომხმარებელი გამოსახულებაზე, რომელიც <img> ტეგში ismap ატრიბუტის საშუალებითაა განსაზღვრული, როგორც გამოსახულების რუკა, მაუსის მაჩვენებლით მარცხენა კლავიშით დააწკაპუნებს, ვებსერვერს გადაეცემა დაწკაპუნების წერტილის კოორდინატები. ამ კოორდინატების მიღების შემდეგ სერვერი რუკაზე ამ კოორდინატების შესაბამის აქტიურ არეებს ეძებს. თუ ასეთი არე არსებობს, მაშინ მოცემული URL-მისამართი გააქტიურდება და მომხმარებლის ბრაუზერი ამ მისამართით ახალ გვერდზე გადავა.

HTML დოკუმენტში გამოსახულების ბმულთან დაკავშირებულ სხვადასხვა არედ დასაშლელად (საიდანაც სხვა დოკუმენტზე ან ვებგვერდზე გადასვლა იქნება შესაძლებელი) <map> და <area>

ტეგები გამოიყენება. გამოსახულება, რომელზეც ვაპირებთ მუშაობას, <img> ტეგის საშუალებით უნდა იყოს რეალიზებული, სადაც usemap და src პარამეტრები იქნება გამოყენებული. მაგალითად,

```
<img src=სურათის_სახელი usemap=#რუკის_სახელი>
```

<map> ტეგი გრაფიკული გამოსახულების რუკის კონფიგურაციის აღწერისათვის გამოიყენება და მას ერთადერთი name ატრიბუტი აქვს. აქტიურ არეებზე მაუსის მაჩვენებლის მიტანის შემდეგ ამ არის იდენტიფიცირება მცურავი მოკარნახით შეიძლება. რუკის აღწერისათვის <map> და </map> კონტეინერულ ტეგებს შორის გამოსახულების აქტიური არეები უნდა იყოს აღწერილი, რისთვისაც სპეციალური ტეგი <area> გამოიყენება. <map> ტეგი <img> ან <input> ტეგთან usemap ატრიბუტით შეიძლება იყოს დაკავშირებული. <map> ტეგის name ატრიბუტის მნიშვნელობა usemap ატრიბუტის მნიშვნელობას უნდა ემთხვეოდეს.

<area> ტეგი კლიენტის მხარეს არსებული გამოსახულების რუკის მხოლოდ ერთ-ერთი აქტიური არის შესახებ გვაძლევს ინფორმაციას. მას დახურვის ტეგი არ აქვს. <map> და </map> ტეგებს შორის თითოეული არის აღწერისთვის ცალკ-ცალკე <area> ტეგი უნდა იქნეს გამოყენებული. იმ შემთხვევაში, თუ რომელიმე წერტილი ერთდროულად რამდენიმე აქტიურ არეს ეკუთვნის, ის ბმული შესრულდება, რომელიც ამ არეების აღწერის სიაში პირველია. მისი სინტაქსი შემდეგია:

```
<map name="რუკის_სახელი"><area ატრიბუტები></map>
```

<area> ტეგს შემდეგი ატრიბუტები აქვს: shape, coords, href, nohref, alt, accesskey, title.

ატრიბუტი shape. გამოსახულებაზე მოსანიშნი არე შეიძლება სხვადასხვა ფორმის იყოს, ამიტომ shape ატრიბუტი აქტიური არის ფორმას განსაზღვრავს. მისი შესაძლო მნიშვნელობებია: rect, circle,

poly, default. ეს მნიშვნელობები შესაბამისად მართკუთხა, მრგვალ და მრავალკუთხა ფორმის არეებს, ხოლო default არის ყველა წერტილს განსაზღვრავს. რუკის არის განმსაზღვრელი ატრიბუტის მითითება აუცილებელია.

- ატრიბუტი coords. მისი საშუალებით ცალკეული აქტიური არის კოორდინატებს ვუთითებთ, ე. ი. მისი მნიშვნელობა სხვადასხვა წერტილის კოორდინატთა ჩამონათვალია, რომლებიც ერთმანეთისგან მძიმით გამოიყოფა და ნატურალური რიცხვებით გამოისახება. კოორდინატის საწყისად გამოსახულების ზედა მარცხენა კუთხე ითვლება, რომელსაც 0,0 მნიშვნელობა შეესაბამება. პირველი რიცხვი ჰორიზონტალურ, ხოლო მეორე ვერტიკალურ კოორდინატს განსაზღვრავს.

მართკუთხა ფორმის არისათვის ანუ როდესაც shape=rect, პირველად იწერება ზედა მარცხენა წერტილის, ხოლო შემდეგ ქვედა მარჯვენა წერტილის კოორდინატი. მაგალითად,

```
<area shape = rect  coords = ”33,60,190,250”>
```

წრიული ფორმის არისათვის ანუ როდესაც shape=circle, პირველად იწერება ცენტრის კოორდინატი და შემდეგ წრის რადიუსი პიქსელებში (ანუ მნიშვნელობა გამოისახება სამი რიცხვით). მაგალითად,

```
<area shape=circle coords=”250, 150, 70”>
```

მრავალკუთხა ფორმის არისათვის ანუ როდესაც shape=poly, პირველად იწერება ზედა წერტილის კოორდინატი, შემდეგ თანმიმდევრობით სხვა წერტილის კოორდინატები, ბოლო კოორდინატი არ არის აუცილებელი პირველს ემთხვეოდეს. ინტერნეტბრაუზერი ავტომატურად აერთებს ბოლო წერტილს პირველთან. მაგალითად,

```
<area shape=poly coords=”534,62,699,62,698,236,626,261,534,235”>
```

როდესაც shape=default, მაშინ იგი არ საჭიროებს კოორდინატების მითითებას.

- href და nohref ატრიბუტები. href ატრიბუტის მეშვეობით გამოსახულების რუკის აქტიური არე ამ URL-მისამართის საშუალებით მითითებულ ბმულს უკავშირდება. თუ <area> ტეგში ეს ატრიბუტი არ არის მითითებული ან მითითებულია nohref ატრიბუტი, მაშინ ეს ნიშნავს, რომ მოცემული აქტიური არე გარკვეულ ბმულს არ უკავშირდება.
- ატრიბუტი alt. იმ ბრაუზერის ფანჯარაში, რომელიც <area> ტეგს მხარს არ უჭერს, alt ატრიბუტი გამოსახულებაზე არსებული ყოველი აქტიური არისთვის ალტერნატულ ტექსტს გამოიტანს. სინამდვილეში, ეს ტექსტი, დოკუმენტის შემქმნელისთვის კომენტარის როლს ასრულებს.
- ატრიბუტი accesskey. იგი გრაფიკული გამოსახულების აქტიურ არეებზე წვდომისათვის (URL-ის მიხედვით გადასასვლელად) კლავიშთა კომბინაციას განსაზღვრავს. მაგალითად,

```
<area accesskey=Ctrl+M>
```

- ატრიბუტი title. მისი მეშვეობით მოკარნახე ტექსტის გამოტანაა შესაძლებელი, რომელიც მოცემულ არეზე მაუსის მაჩვენებლის მიტანის დროს ამოცურდება. მისი სინტაქსია:

```
<area title = "კარნახის ტექსტი">
```

მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<title>გამოსახულების რუკა</title>
</head>
```

```
<body>
<img src="Children.jpg" usemap=#Child>
<map name="Child">
<area shape=rect coords="350,85,450,200" href=Index1.html
title="გოგო" ACCESSKEY=Ctrl+G >
<area shape=circle coords="580,135,65" href=Index2.html
title="ბიჭი" ACCESSKEY=Ctrl+B >
</map>
</body>
</html>
```

მორბენალი სტრიქონი

<marquee> </marquee> არის კონტეინერის ტიპის ტეგი. მისი საშუალებით იქმნება მორბენალი სტრიქონი. ამ ტეგის ატრიბუტებია: align, bgcolor, behavior, direction, height, width, hspace, vspace, loop, scrollamount, scrolldelay.

- ატრიბუტი align საშუალებას იძლევა ტექსტი მორბენალი

სტრიქონის ზედა (top), შუა (middle) ან ქვედა (botton) ნაწილში განვათავსოთ.

- ატრიბუტი bgcolor - მორბენალი სტრიქონის ფონის ფერს განსაზღვრავს;
- ატრიბუტი direction განსაზღვრავს მორბენალი სტრიქონის მოძრაობის მიმართულებას. ამ ატრიბუტის right მნიშვნელობა უზრუნველყოფს მორბენალი სტრიქონის გადაადგილებას მარცხნიდან მარჯვნივ, left მნიშვნელობა – მორბენალი სტრიქონის გადაადგილებას მარჯვნიდან მარცხნივ. down მნიშვნელობის მითითების შემთხვევაში მორბენალი სტრიქონი გადაადგილდება ზემოდან ქვემოთ. მნიშვნელობა up უზრუნველყოფს მორბენალი სტრიქონის ქვემოდან ზემოთ მოძრაობას.
- ატრიბუტი behavior განსაზღვრავს მოძრაობის სტილს. მისი მნიშვნელობებია: alternate, scroll, slide.
  - Alternat მნიშვნელობის შემთხვევაში მორბენალი სტრიქონი მოძრაობს მარჯვნიდან მარცხნივ კიდემდე, შემდეგ მარცხნიდან მარჯვნივ კიდემდე განუწყვეტლივ.
  - scroll მნიშვნელობის შემთხვევაში მორბენალი სტრიქონი მოძრაობს direction ატრიბუტში მითითებული მნიშვნელობის შესაბამისად და იმეორებს მოძრაობას განუწყვეტლივ. ის მითითებულია ავტომატურად. შეგიძლიათ არც მიუთითოთ.
  - slide მნიშვნელობის შემთხვევაში მორბენალი სტრიქონი მოძრაობს direction ატრიბუტში მითითებული მნიშვნელობის შესაბამისად კიდემდე და ჩერდება.
- ატრიბუტი height მიუთითებს მორბენალი სტრიქონის სიმაღლეს, განისაზღვრება პიქსელებში ან პროცენტებში.
- ატრიბუტი width მორბენალი სტრიქონის სიგანეს განსაზღვრავს პიქსელებში;
- ატრიბუტი hspace საშუალებას იძლევა მორბენალ სტრი-

ქონსა და მის გარშემო განთავსებულ ტექსტს ან გრაფიკულ გამოსახულებას შორის განისაზღვროს მარცხენა და მარჯვენა მინდვრების სიგანე პიქსელებში;

- ატრიბუტი vspace საშუალებას იძლევა მორბენალ სტრიქონსა და მის გარშემო განთავსებულ ტექსტს ან გრაფიკულ გამოსახულებას შორის განისაზღვროს ზედა და ქვედა მინდვრების სიმაღლე პიქსელებში;
- ატრიბუტი loop მორბენალი სტრიქონის ეკრანზე გასვლის რაოდენობას განსაზღვრავს. თუ ეს ატრიბუტი გამოტოვებულია, მაშინ მორბენალი სტრიქონი იმოძრავებს მანამ, ვიდრე მოცემული გვერდი ეკრანზე იქნება გახსნილი;
- ატრიბუტი scrollamount მორბენალი სტრიქონის ეკრანზე გადაადგილების სიჩქარეს მართავს.
- ატრიბუტი scrolldelay თავისი ფუნქციებით წინა ატრიბუტს მოგვაგონებს და მორბენალი სტრიქონის ტექსტის ერთი წერტილიდან მეორეში გადასვლის დროს განსაზღვრავს მილიწამებში.

ამ ბოლო ორი ატრიბუტის კომბინაცია მორბენალი სტრიქონის გადაადგილების ოპტიმალური სიჩქარის განსაზღვრის საშუალებას გვაძლევს.

ცხრილები

ცხრილების შექმნის პრინციპი შემდეგში მდგომარეობს: ვებგვერდზე იქმნება ცხრილი, უჯრედების უხილავი საზღვრებით და ელემენტები, რომლებიც ზუსტ პოზიციონირებას მოითხოვს, ცხრილის უჯრედებში განლაგდება. თითოეული უჯრისთვის შეგიძლიათ დაფორმატების საკუთარი პარამეტრები მიუთითოთ. შესაბამისად, დაფორმატების ბრძანებები მხოლოდ უჯრის საზღვრებში მოქმედებს. პრაქტიკაში ხშირად ტექსტის რამდენიმე

სვეტად დაყოფა გვჭირდება. ცხრილები სწორედ ამის გაკეთების საშუალებას იძლევა.

ცხრილი იწყება <table> ტეგით და მთავრდება </table>-ით. ჩუმათობის პრინციპით, ცხრილს ჩარჩო არ აქვს და უჯრედების საზღვრებიც უხილავია. ჩარჩო border ატრიბუტის საშუალებით ემატება. თავისი ბუნებრივი დანიშნულების (ტექსტური ინფორმაციის მოწესრიგება) გარდა, ცხრილები დიზაინის საკითხების გადაწყვეტის საშუალებასაც გვაძლევს, მაგალითად, ცხრილების საშუალებით შესაძლებელია სურათი და ტექსტი ერთმანეთის გვერდით განთავსდეს, ტექსტი დაიყოს სვეტებად და ა.შ.

<table> ტეგის ატრიბუტებია: align, border, background, bgcolor, bordercolor, cellpadding, cellspacing, hspace, vspace, colspec, width .

- ატრიბუტი align ცხრილის მდებარეობას განსაზღვრავს და შემდეგი მნიშვნელობების მიღება შეუძლია: left (მარცხნივ), right (მარჯვნივ) ან center (ცენტრში);
- ატრიბუტი border - ცხრილის ჩარჩოს ტიპს განსაზღვრავს. თუ მოცემული ატრიბუტი გამოყენებულია, მაშინ ჩარჩო ყველა უჯრედს და მთლიანად ცხრილს გაუკეთდება. border ატრიბუტს შეუძლია მიიღოს რიცხვითი მნიშვნელობა, რომელიც ჩარჩოს სისქეს განსაზღვრავს პიქსელებში;
- ატრიბუტი background გამოიყენება ცხრილის ფონად რაიმე სურათის ჩასასმელად (მაგ., Tea.jpg);
- ატრიბუტი bgcolor - ცხრილის ფონის ფერს განსაზღვრავს;
- ატრიბუტი bordercolor - ჩარჩოს ფერს უთითებს და მხოლოდ border ატრიბუტთან ერთად გამოიყენება, მაგალითად,

<table border="რიცხვი" bordercolor="ფერი">

- ატრიბუტი cellpadding - უჯრის ჩარჩოსა და უჯრედში მოთავსებულ ელემენტს (ტექსტი, სურათი) შორის მანძილს განსაზღვრავს პიქსელებში;

- ატრიბუტი cellspacing - უჯრედებს შორის მანძილს განსაზღვრავს პიქსელებში;
- ატრიბუტი hspace - ცხრილის მარჯვნივ და მარცხნივ ცარიელ ადგილს განსაზღვრავს პიქსელებში;
- ატრიბუტი vspace - ცხრილის ზემოთ და ქვემოთ ცარიელ ადგილს განსაზღვრავს პიქსელებში;
- ატრიბუტი colspec - ფიქსირებული სიგანის სვეტების განსაზღვრის საშუალებას იძლევა. სიგანის მნიშვნელობა როგორც სიმბოლოების რაოდენობით, ასევე პროცენტებით შეიძლება იყოს მოცემული;
- ატრიბუტი width - განსაზღვრავს ცხრილის სიგანეს. სიგანის მნიშვნელობა შეიძლება როგორც პიქსელებით, ასევე პროცენტებით იყოს მოცემული.

ცხრილის სტრიქონებისა და სვეტების შექმნა

ცხრილის სტრიქონებისა და სვეტების შესაქმნელად <tr> </tr> და <col> </col> ტეგები გამოიყენება.

<tr> ტეგი (ინგლ. *Table Row* - ცხრილის სტრიქონი) ცხრილის სტრიქონს ქმნის. სტრიქონს შეიძლება align (ჰორიზონტალურად სწორება) და valign (ვერტიკალურად სწორება) ატრიბუტები ჰქონდეს.

<col> (ინგლ. Column - სვეტი) ტეგი ცხრილის სვეტის შესაქმნელად გამოიყენება და შეიძლება width, bgcolor, align, valign და id ატრიბუტები ჰქონდეს.

- ატრიბუტი width - სვეტში უჯრის სიგანეს განსაზღვრავს პიქსელებში;

```
<col width=რიცხვი>
```

- ატრიბუტი bgcolor - სვეტში უჯრის ფონს განსაზღვრავს:

```
<col bgcolor=ფერი>
```

- ატრიბუტი align - უჯრის შიგთავსის ჰორიზონტალურად სწორების რეჟიმს განსაზღვრავს. მან შეიძლება left (მარცხნივ), center (ცენტრში) და right (მარჯვნივ) მნიშვნელობები მიიღოს;
- ატრიბუტი valign - უჯრის შიგთავსის ვერტიკალურად სწორების რეჟიმს განსაზღვრავს. მან შეიძლება top (ზემოთ), middle (შუაში) და bottom (ქვემოთ) მნიშვნელობები მიიღოს.

ცხრილის თითოეული სტრიქონი იწყება ტეგით <tr> და მთავრდება </tr>-ით.

```
<table>
<tr>  <!... I უჯრის სტრიქონის აღწერა></tr>
<tr>  <!... II უჯრის სტრიქონის აღწერა></tr>
</table>
```

ყველაფერი, რაც ჩვენს ცხრილში უნდა გამოჩნდეს, იწერება <td>-სა და </td>-ს შორის.

ცხრილის უჯრედების შექმნა

ცხრილის უჯრის შესაქმნელად <td> </td> (ინგლ. *Table Data* - ცხრილური მონაცემი) ტეგი გამოიყენება.

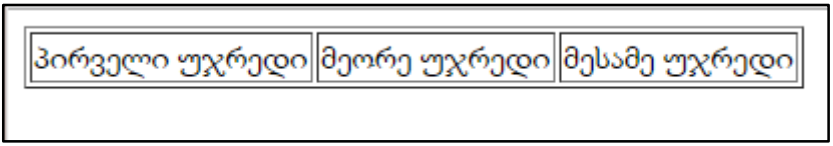
მაგალითი: ერთუჯრიანი ცხრილი

```
<!DOCTYPE html>
<Html>
<head><title>Table</title></head>
<body>
<table border="1">
<tr>
<td>ერთუჯრიანი ცხრილი</td>
</tr>
</table>
```

```
</body>
</html>
```

მაგალითი: სამუჯრიანი ერთსტრიქონიანი ცხრილი

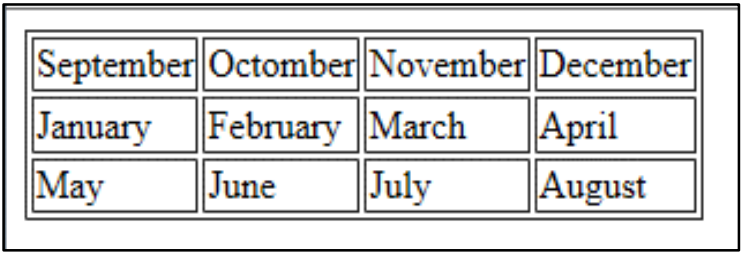
```
<!DOCTYPE html>
<html>
<head><title>Table</title></head>
<body>
<table border="1">
<tr>
<td>პირველი უჯრედი</td>
<td>მეორე უჯრედი</td>
<td>მესამე  უჯრედი</td>
</tr>
</table>
</body>
</html>
```



მაგალითი: სამი სტრიქონისა და ოთხი სვეტისაგან შემდგარი ცხრილი:

```
<!DOCTYPE html>
<html>
<head><title>Table</title></head>
<body>
<table border="1">
<tr>
<td>September</td>
```

```
<td>Octomber</td>
<td>November</td>
<td>December</td>
</tr>
<tr>
<td>January</td>
<td>February</td>
<td>March</td>
<td>April</td>
</tr>
<tr>
<td>May</td>
<td>June</td>
<td>July</td>
<td>August</td>
</tr>
</table>
</body>
</html>
```



September	Octomber	November	December
January	February	March	April
May	June	July	August

ცხრილის სათაურის უჯრედი და წარწერა

ცხრილის სათაურის შესაქმნელად <th>, </th> (ინგლ. *Table Head* - ცხრილის სათაური) ტეგი გამოიყენება. ცხრილის სათაურს მთელი ცხრილის სიგანე აქვს. მოცემულ უჯრედში ტექსტს bold და align=center ატრიბუტები აქვს.

ცხრილის წარწერის შესაქმნელად <caption> (ინგლ. *caption* - წარწერა) წყვილი ტეგი გამოიყენება. <caption> ტეგი <table> და </table> ტეგებს შორის და სტრიქონისა და უჯრის აღწერის გარეთ უნდა იყოს განთავსებული. align ატრიბუტი განსაზღვრავს, სად უნდა იყოს წარწერა.

მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<title> Table caption </title
</head>
<body>
<table border="1px">
  <caption>Monthly savings</caption>
  <tr>
    <th>Month</th>
    <th>Savings</th>
  </tr>
  <tr>
    <td>January</td>
    <td>$100</td>
  </tr>
  <tr>
    <td>February</td>
```

```
    <td>$50</td>
  </tr>
</table>
</body>
</html>
```

მიღებული შედეგი:

Monthly savings

Month	Savings
January	$100
February	$50

ცხრილის ელემენტების ძირითადი ატრიბუტები

ცხრილის ელემენტების ზოგიერთ ატრიბუტს უკვე გავეცანით ცხრილებთან დაკავშირებული ტეგების შესწავლის დროს. ახლა განვიხილოთ დანარჩენი მნიშვნელოვანი ატრიბუტები.

<tr> ტეგს შეიძლება რამდენიმე ატრიბუტი ჰქონდეს:

Align = left ცხრილის სტრიქონისათვის ასწორებს ტექსტს მარცხენა მხრიდან; Align=”center” (ტექსტს ათავსებს ცენტრში), Align=”right” (ტექსტს ასწორებს მარჯვენა მხრიდან).

Valign - მიუთითებს ტექსტის ვერტიკალურ მდებარეობაზე.

valign=”top” (ტექსტს უახლოებს უჯრის ზედა ნაწილს)

valign=”middle” (ტექსტი უჯრის შუაში, ცენტრშია)

valign=”bottom” (ტექსტი უჯრის ქვედა კიდეშია)

bgcolor - მოცემული სტრიქონისათვის ცხრილის ფერს განსაზღვრავს.

<td>-ს ატრიბუტებია:

- colspan-ის მნიშვნელობა განსაზღვრავს ცხრილის რამდენი უჯრედი უნდა გაერთიანდეს ჰორიზონტალურად მონიშნულ უჯრედთან ერთად.
  მაგალითად, colspan = 3 ნიშნავს, რომ ჰორიზონტალურად სამი უჯრედი ერთ უჯრედად გაერთიანდა, გაიზარდა.
- rowspan, რომელიც მიუთითებს ცხრილის რამდენი უჯრედი უნდა გაერთიანდეს ვერტიკალურად მონიშნულ უჯრედთან ერთად. rowspan=2 ნიშნავს, რომ უჯრედი იკავებს ორ სტრიქონს;
- align უჯრედში ტექსტის განლაგებას ცვლის. დასაშვები მნიშვნელობებია: align = “left”; align=”right”; align=”center”;
- valign ტექსტის ვერტიკალურ მდებარეობას განსაზღვრავს. valign=”top” (ტექსტი მიახლოებულია უჯრედის ზედა კიდესთან); valign=”middle” (ტექსტის მოთავსებულია ცენტრში); valign=”bottom” (ტექსტი უჯრის ქვედა კიდეშია);
- width განსაზღვრავს უჯრის სიგანეს პიქსელებში (მაგ., width = 200);
- height განსაზღვრავს უჯრის სიმაღლეს პიქსელებში;
- bgcolor მითითებული უჯრისთვის ცხრილის ფერს მიუთითებს;

- ატრიბუტი rules - უჯრედებს შორის ხაზების გამოტანის წესს განსაზღვრავს, მაგალითად,

```
<table rules=all>
```

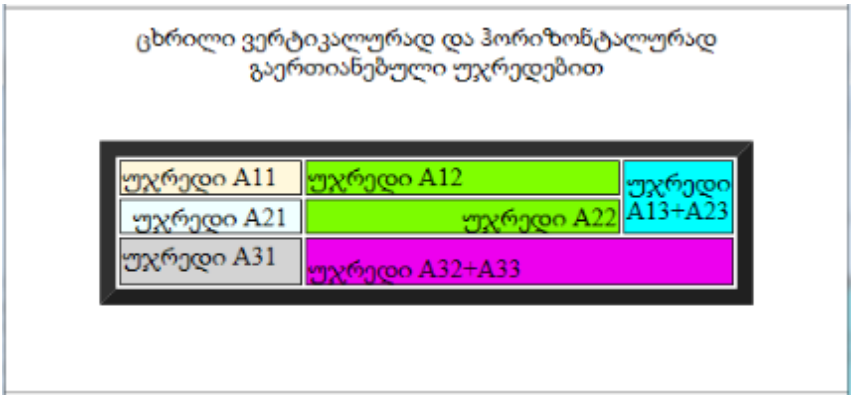
მისი დასაშვები მნიშვნელობებია:

- none - ხაზი არ არის (ეს მნიშვნელობა ჩუმათობის პრინციპით მიენიჭება);
- rows - ხაზები მხოლოდ სტრიქონებს შორის;
- cols - ხაზები მხოლოდ სვეტებს შორის;
- all - ხაზები სტრიქონებსა და სვეტებს შორის;

მაგალითი: ქვემოთ მოცემულ პროგრამაში rowspan და

colspan ატრიბუტების გამოყენებით ცხრილის უჯრედების გაერთიანებაა განხილული.

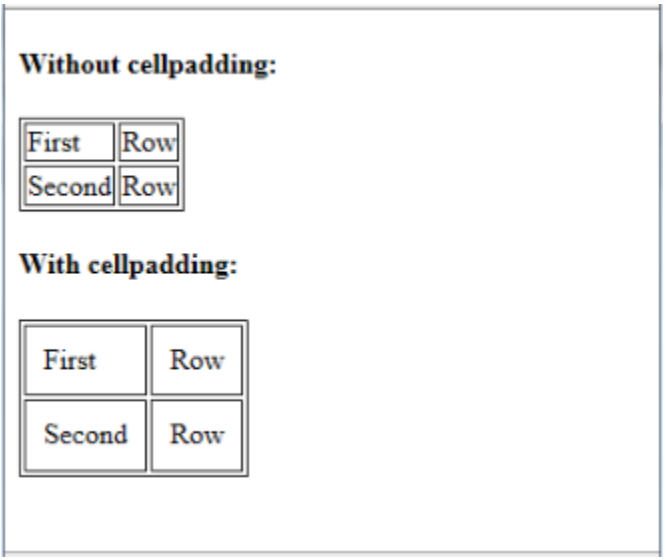
```
<!DOCTYPE  html>
<html>
<head>
<title>ცხრილები</title>
</head>
<body>
<table border=10 width=80% align=center>
<caption>ცხრილი ვერტიკალურად და ჰორიზონტალურად გაერთიანებული უჯრედებით <br><br><br>
</caption>
<tr><td width=30% bgcolor=cornsilk>უჯრედი A11 </td>
<td width=200 bgcolor=chartreuse>უჯრედი A12 </td>
<td rowspan=2 bgcolor=aqua>უჯრედი  A13+A23 </td></tr>
<tr><td align=center bgcolor=azure> უჯრედი A21 </td>
<td align=right bgcolor=lawngreen> უჯრედი A22 </td></tr>
<tr><td height=30 valign=top bgcolor=lightgrey> უჯრედი  A31 </td>
<td colspan=2 valign=bottom bgcolor=mediumspringgreen> უჯრედი A32+A33 </td></tr>
</table>
</body>
</html>
```



მაგალითი: cellpadding ატრიბუტის გამოყენება

```
<!DOCTYPE html>
<html>
<body>
<h4>Without cellpadding:</h4>
<table border="1">
<tr>
  <td>First</td>
  <td>Row</td>
</tr>
<tr>
  <td>Second</td>
  <td>Row</td>
</tr>
</table>
<h4>With cellpadding:</h4>
<table border="1"
cellpadding="10">
<tr>
```

```
  <td>First</td>
  <td>Row</td>
</tr>
<tr>
  <td>Second</td>
  <td>Row</td>
</tr>
</table>
</body>
</html>
```



მაგალითი: cellspacing ატრიბუტის გამოყენება

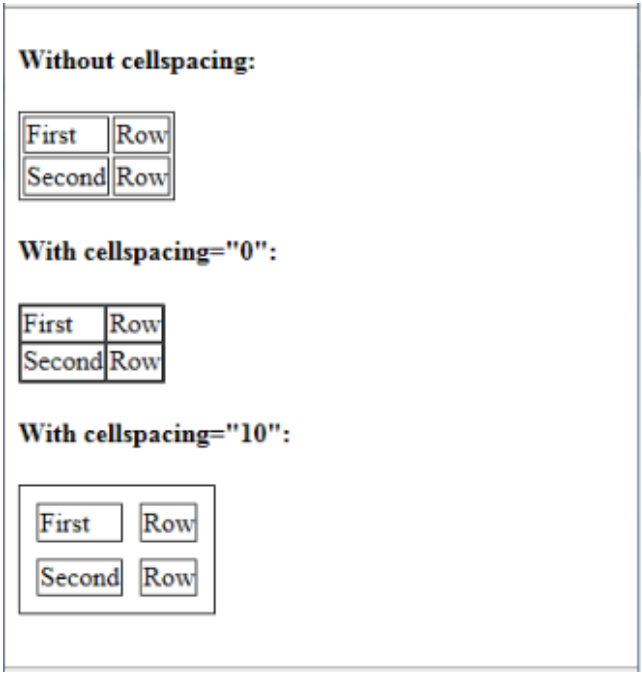
```
<!DOCTYPE html>
<html>
<body>
<h4>Without cellspacing:</h4>
```

```
<table border="1">
<tr>
  <td>First</td>
  <td>Row</td>
</tr>
<tr>
  <td>Second</td>
  <td>Row</td>
</tr>
</table>

<h4>With cellspacing="0":</h4>
<table border="1" cellspacing="0">
<tr>
  <td>First</td>
  <td>Row</td>
</tr>
<tr>
  <td>Second</td>
  <td>Row</td>
</tr>
</table>

<h4>With cellspacing="10":</h4>
<table border="1" cellspacing="10">
<tr>
  <td>First</td>
  <td>Row</td>
</tr>
<tr>
  <td>Second</td>
```

```
  <td>Row</td>
</tr>
</table>
</body>
</html>
```

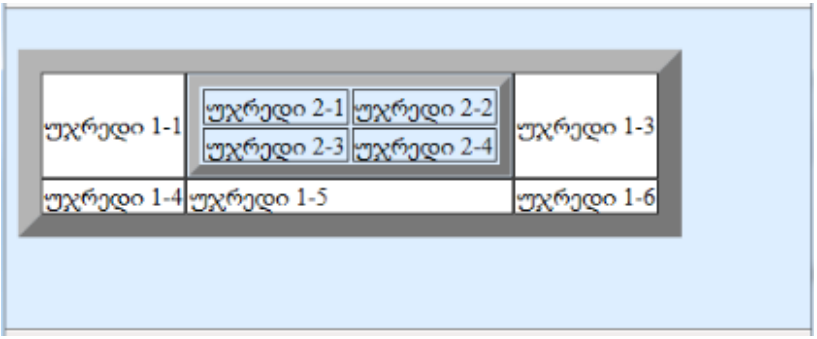


ქვემოთ მოცემულია ცხრილის მაგალითი, რომელიც სხვა ცხრილის უჯრედშია მოთავსებული:

```
<!DOCTYPE html>
<html>
<head>
<title>ცხრილი ცხრილში</title>
</head>
<body bgcolor=#ddeeff>
```

```
<table border=15 cellspacing=0>
<tr>
<td bgcolor="#ffffff">უჯრედი 1-1</td>
<td>
<table border=7>
<tr>
<td>უჯრედი 2-1</td>
<td>უჯრედი 2-2</td>
 </tr>
<tr>
<td>უჯრედი 2-3 </td>
<td>უჯრედი 2-4</td>
</tr>
</table>
</td>
<td bgcolor="#ffffff">უჯრედი 1-3</td>
 </tr>
<tr>
<td bgcolor="#ffffff">უჯრედი 1-4</td>
<td bgcolor="#ffffff">უჯრედი 1-5</td>
<td bgcolor="#ffffff">უჯრედი 1-6</td>
 </tr>
</table>
</body>
</html>
```

ბრაუზერის ფანჯარაში მიღებულ შედეგს ექნება სახე:



ფორმები

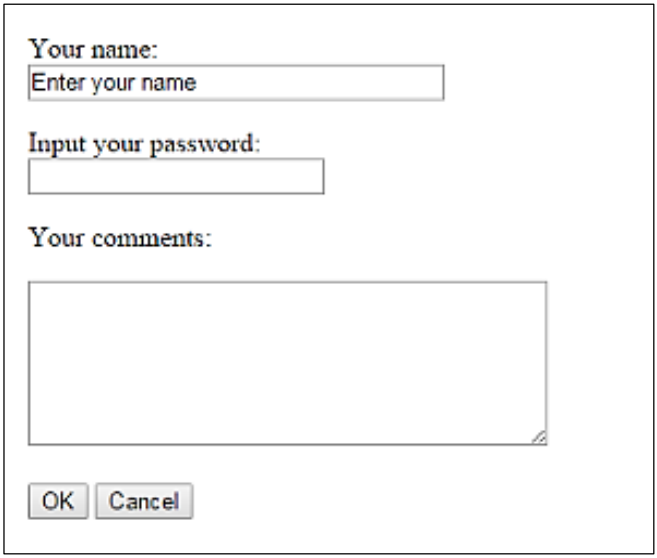
ფორმები მსოფლიო აბლაბუდაში ინტერაქტიული ურთიერთქმედებისთვის ერთ-ერთ ყველაზე პოპულარული საშუალებაა. ფორმების საშუალებით მომხმარებლისგან ინფორმაციის მიღებაა შესაძლებელი. HTML-ის დახმარებით შეგიძლიათ მარტივი ფორმები შექმნათ, რომლებიც“დიახ” და “არა” ტიპის პასუხებს ითვალისწინებს. აგრეთვე, შეგიძლიათ შექმნათ რთული ფორმები შეკვეთებისთვის ან იმისთვის, რომ თქვენი მკითხველებისაგან რაიმე კომენტარი ან კეთილი სურვილები მიიღოთ.

ფორმები რამდენიმე ველისგან შედგება, სადაც მომხმარებელს შეუძლია გარკვეული ინფორმაცია შეიტანოს ან რომელიმე პარამეტრი აირჩიოს. მას შემდეგ, რაც მომხმარებელი გაგზავნის ინფორმაციას, ის სერვერზე მოთავსებული პროგრამის (სკრიპტის) მიერ დამუშავდება. სკრიპტი მოკლე პროგრამაა, რომელიც თითოეული ფორმის დასამუშავებლად სპეციალურადაა შექმნილი.

HTML-ის დახმარებით მხოლოდ ფორმის ვიზუალურ მხარეს განვსაზღვრავთ. იმისათვის, რომ ფორმის ღილაკებმა რეალურად იმუშაოს, სხვა ტექნოლოგია, რაიმე პროგრამა ან სკრიპტია საჭირო, რომელიც ჩვენს ფორმას უნდა მივაბათ.

ფორმის ტეგებთან მუშაობა

როგორც ზემოთ უკვე აღვნიშნეთ, იმისათვის, რომ ფორმამ რეალურად იმუშაოს და შევსებული ფორმის მონაცემები დამუშავდეს, HTML-ის ცოდნა საკმარისი არ არის, აქ უკვე სხვა ტექნოლოგიაა საჭირო. HTML-ს არ შეუძლია მონაცემების დამუშავება, ამისათვის სერვერული დაპროგრამების სპეციალური ენები არსებობს, მაგალითად, PHP, ამიტომაც ფორმა ყოველთვის იმ სერვერულ დამმუშავებელთან უნდა იყოს დაკავშირებული, რომელიც ფორმის მონაცემებს დაამუშავებს. განვიხილოთ მუშაობის სქემა. მაგალითად, როდესაც ქვემოთ მოცემული ტიპის ფორმის შევსება ხდება და მომხმარებელი OK ღილაკს აჭერს ხელს, მონაცემები გადაეგზავნება მონაცემების დამმუშავებელს. ამისათვის, გამოვიყენოთ ფაილი formdata.php.



ეს ფაილი ფორმიდან მიღებულ მონაცემებს საიტის მფლობელის სურვილის მიხედვით დაამუშავებს. შეიძლება მას სურდეს

ფორმაში არსებული ინფორმაცია ელექტრონული წერილის სახით მიუვიდეს ან მოხდეს მისი ჩაწერა ბაზაში. ამისათვის, formdata.php-ში შესაბამისი კოდი უნდა ჩაიწეროს, რომელიც ან ერთი, ან მეორე გზით უზრუნველყოფს ინფორმაციის მიწოდებას.

ფორმის კარკასის შესაქმნელად საჭირო კოდს ქვემოთ მოცემული სახე აქვს:

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>forms1</title>
</head>
<body>
<form action =”formdata.php” method=”POST” name =”form1”>
</form>
</body>
</html>
```

ფორმები ინფორმაციის შეტანისათვისაა განკუთვნილი. ყველა ფორმა <form> (ინგლ. form - ფორმა) ტეგით იწყება და </form> ტეგით მთავრდება. ფორმას შეიძლება შემდეგი ატრიბუტები ჰქონდეს: name, action, method, target.

- ატრიბუტი name ფორმის სახელს განსაზღვრავს. თუ დოკუმენტში რამდენიმე ფორმა არ არის გამოყენებული, მაშინ მისი მითითება აუცილებელი არ არის;
- ატრიბუტი action უზრუნველყოფს ფორმის მონაცემების შესაბამისი ფაილისთვის დასამუშავებლად გადაგზავნას.
- ატრიბუტი method ფორმის პარამეტრების გადაცემის მეთოდს განსაზღვრავს. იგი get ან post მნიშვნელობებს ღებულობს.
- ატრიბუტი target განსაზღვრავს ფანჯარას, რომელშიც

გაგზავნილი ფორმის დამუშავების შედეგები დაბრუნდება. მასში ფანჯრის სახელი ცხადად უნდა ჩაიწეროს ან ერთ-ერთი შემდეგი პარამეტრი უნდა მიეთითოს:

- blank - დოკუმენტი ახალ ფანჯარაში გაიხსნება;
- self - დოკუმენტი იგივე ფანჯარაში გაიხსნება, რომელ-შიც მიმდინარე დოკუმენტია გახსნილი;
- parent - დოკუმენტი იმ ფრეიმში გაიხსნება, რომელიც მიმდინარე დოკუმენტის შემცველი ფრეიმის მიმართ მშობლიურ ფრეიმს წარმოადგენს (თუ მშობლიური ფრეიმი არ არსებობს, მაშინ დოკუმენტი მიმდინარე ფრეიმში გაიხსნება);
- top - დოკუმენტი მიმდინარე ფანჯარაში გაიხსნება და ყველა ფანჯარას დაიჭერს.

ზემოთ მოცემული კოდის მუშაობის შედეგი ბრაუზერში არ გამოჩნდება, რადგანაც ჩვენ მხოლოდ ფორმის კარკასი შევქმენით. იმისათვის, რომ ფორმა გამოჩნდეს, საჭიროა ფორმის ელემენტების შექმნა.

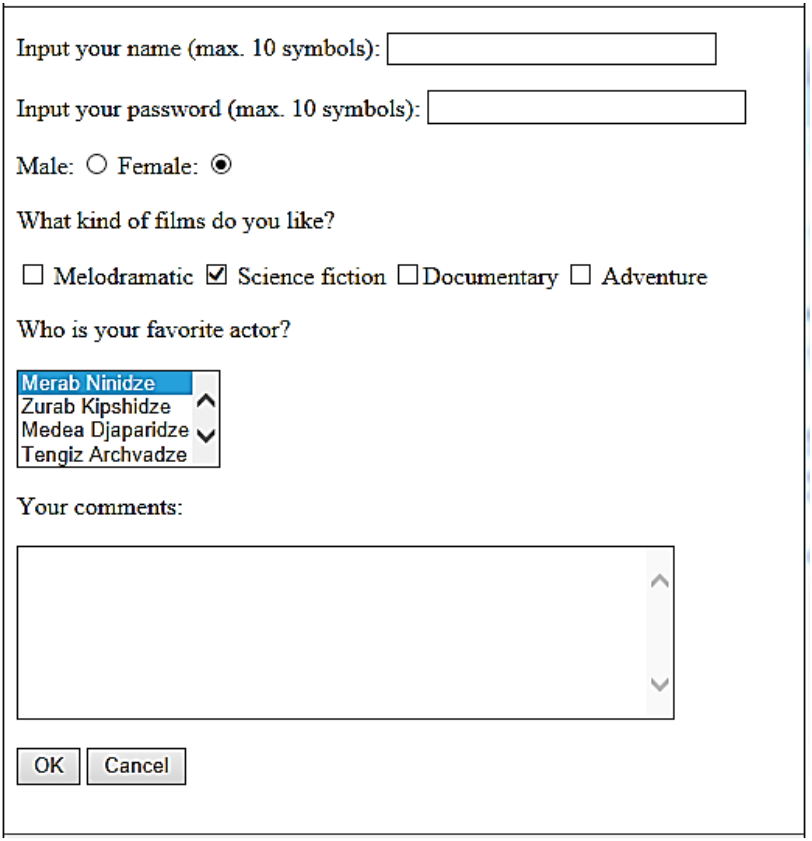
ჩვენი კოდი შემდეგ სახეს მიიღებს:

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>forms1</title>
</head>
<body>
<form action ="formdata.php" method="POST" name ="form1">
<p>
<label>Input your name (max. 10 symbols): <input type="text"
name="name" size="30" maxlength="10">
</label>
```

```
    </p>
<p> <label>Input your password (max. 10 symbols): <input
type="password" name="password" size="30" maxlength="10"> </label>
    </p>
    <p> <label> Male: <input type="radio" name="mf" value="Male">
</label>
    <label> Female: <input type="radio" name="mf" value="Female"
checked>    </label> </p>
    <p> What k ind of films do you like?</p>
    <p> <label> <input type="checkbox" name="melodrama">
Melodramatic </label>
    <label> <input type="checkbox" name="science fiction" checked>
Science fiction </label>
    <label> <input type="checkbox"
name="documentary">Documentary </label>
    <label> <input type="checkbox" name="adventure"> Adventure
</label> </p>
    <p> Who is your favorite actor? </p>
    <select  name: "actor" size ="4">
    <option value ="Ninidze" selected> Merab Ninidze
    <option value ="Kipshidze"> Zurab Kipshidze
    <option value ="Medea"> Medea Djaparidze
    <option value ="Archvadze"> Tengiz Archvadze
    <option value ="Lia"> Lia Eliava
    <option value ="Koberidze"> Otar Koberidze
    </select>
    <p>Your comments:</p>
    <textarea rows="7" cols="50" name="comments"> </textarea> <br>
    <br>
    <input type="submit" value="OK">
```

```
<input type="reset" value="Cancel">
</form>
</body>
</html>
```

ბრაუზერის ფანჯარაში დავინახავთ:



მართვის ელემენტების ფორმაზე განთავსება

მონაცემთა შესატანი ველის, რომელსაც ზემოთ გავეცანით და მართვის სხვა ელემენტების შესაქმნელად <input> (ინგლ. input - შეტანა) ტეგი გამოიყენება. ამ ტეგს name, size, checked, maxlength, src, type ატრიბუტები აქვს.

• ატრიბუტი name - მონაცემთა შესატანი ველის სახელს განსაზღვრავს. მოცემული სახელი განიხილება, როგორც ველის უნიკალური იდენტიფიკატორი, რომლის მიხედვითაც შემდგომში შეიძლება მომხმარებლის მიერ ამ ველში შეტანილი მონაცემები მივიღოთ;

• ატრიბუტი size - მონაცემთა შესატანი ველის ზომას განსაზღვრავს. ეს ატრიბუტი განსაზღვრავს ველის სიგრძეს და არა მასში შესატანი სიმბოლოების რაოდენობას. თუ მონაცემთა შესატან ველში შეტანილი სიმბოლოების რაოდენობა ველის სიგრძეს აღემატება, მაშინ იქ გადაფურცვლის ზოლი გაჩნდება;

• ატრიბუტი checked - ალმებით მოსანიშნად და გადამრთველებისათვის გამოიყენება. იგი უჩვენებს, რომ ვებგვერდის გახსნის დროს ამ ატრიბუტის მქონე ველი მონიშნული იქნება;

• ატრიბუტი maxlength - მომხმარებლის მიერ მონაცემთა შესატან ველში შესატანი სიმბოლოების დასაშვები რაოდენობა. ამ რაოდენობაზე მეტი სიმბოლოს შეტანის მცდელობის დროს ბრაუზერი ხმოვან სიგნალს გამოსცემს და ზედმეტი სიმბოლოს შეტანის საშუალებას არ მოგვცემს. თუ მოცემული ატრიბუტი არ იქნება მითითებული, მაშინ მისი მნიშვნელობა ზოგადად უსასრულობის ტოლია;

• ატრიბუტი src - გამოსახულების URL-მისამართს მიუთითებს (image ატრიბუტთან ერთად გამოიყენება);

• ატრიბუტი type - მართვის ელემენტის ტიპს განსაზღვრავს. თუ იგი არ იქნება მითითებული, მაშინ გამოდის ერთსტრიქონიანი

მონაცემთა შესატანი ველი. ყველა დანარჩენი ტიპი აუცილებლად უნდა მიეთითოს; type ატრიბუტმა შეიძლება შემდეგი მნიშვნელობები მიიღოს:

- Checkbox, რომელიც მონაცემთა შესატან ველს ალმით მოსანიშნ ველად გარდაქმნის;
- Hidden, რომელიც მართვის ელემენტს მალავს - იგი ბრაუზერის მიერ არ აისახება და მომხმარებელს ჩუმათობის პრინციპით მინიჭებული მნიშვნელობის შეცვლის უფლებას არ აძლევს. მართვის ასეთი ელემენტი პროგრამისათვის უცვლელი ინფორმაციის გადასაცემად გამოიყენება, მაგალითად, მომხმარებლის იდენტიფიკატორი ან პაროლი;
- image საშუალებას იძლევა მართვის ელემენტის სახელს გამოსახულება დაუკავშირდეს.
- password მონაცემთა შესატან ველს გარეგნულად არ ცვლის, მაგრამ ბრაუზერი მომხმარებლის მიერ შეტანილ მნიშვნელობებს ეკრანზე არ ასახავს;
- radio მონაცემთა შესატან ველს ისეთ გადამრთველად გადააქცევს, რომელიც საშუალებას იძლევა name ატრიბუტის ერთი და იგივე მნიშვნელობის და value ატრიბუტით მოცემულ გადამრთველის სხვადასხვა მნიშვნელობას შორის მხოლოდ ერთი მნიშვნელობა აირჩეს.
- text ერთსტრიქონიან მონაცემთა შესატან ველს აღწერს. მაქსიმალური სიგრძის მნიშვნელობისა და მონაცემთა შესატანი ველის განსაზღვრისათვის maxlength და size ატრიბუტები გამოიყენება.
- reset მონაცემთა შესატან ველს ღილაკად გარდაქმნის, რომელზეც მაუსის დაწკაპუნებით ფორმის მართვის ყველა ელემენტი მათთვის ჩუმათობის პრინციპით მინიჭებულ მნიშვნელობას მიიღებს;

- submit მონაცემთა შესატან ველს ღილაკად გარდაქმნის, რომელზეც მაუსის დაწკაპუნებით ხდება ინფორმაციის გაგზავნა სერვერზე.

- *ატრიბუტი value* - საშუალებას იძლევა მართვის ელემენტს მიენიჭოს მნიშვნელობა ჩუმათობის პრინციპით ან მნიშვნელობა, რომელიც name ატრიბუტს შესაბამისი გადამრთველის დაყენების საშუალებით მიენიჭება (გადამრთველის ტიპისათვის მოცემული ატრიბუტი აუცილებელია).

ფორმაში მრავალსტრიქონიანი ტექსტური ველების შესაქმნელად, რომელიც მომხმარებელს დიდი მოცულობის ინ-ფორმაციის შეტანის საშუალებას აძლევს, <textarea> ტეგი გამო-იყენება. ამ ტეგს name, rows და cols ატრიბუტები აქვს.

- ატრიბუტი name - მონაცემთა შეტანის ველის სახელს გან-საზღვრავს;
- ატრიბუტი rows - მონაცემთა შეტანის ველის სიმაღლეს გან-საზღვრავს სიმბოლოებში;
- ატრიბუტი cols - მონაცემთა შეტანის ველის სიგანეს გან-საზღვრავს სიმბოლოებში;

იმისათვის, რომ მონაცემთა შეტანის ველში ფორმის გახსნის დროს ჩუმათობის პრინციპით რაიმე ტექსტი გამოვიდეს, საჭიროა იგი <textarea> და </textarea> ტეგებს შორის ჩავსვათ.

დავუბრუნდეთ ჩვენს ფორმას და მიღებული შედეგი განვიხილოთ:

როგორც ვიცით, <input> ტეგის მეშვეობით ფორმის სხვა-დასხვა სახის ელემენტის შექმნა შეიძლება. ელემენტის ტიპი განისაზღვრება ამ ტეგის ატრიბუტით, მაგალითად, <input type="text"> ერთსტრიქონიანი ტექსტის შესატან ელემენტს ქმნის. ჩვენს შემთხვევაში ამ ელემენტის სახელია name.

<input type="text" name="name"> - input ტეგის ატრიბუტის სახელია name და მნიშვნელობა - name.

გარდა type და name აუცილებელი ატრიბუტებისა, გვაქვს სხვა ატრიბუტებიც: value - ტექსტის შეტანის ველისათვის (და არა მხოლოდ მისთვის), size - ტექსტის შესატანი ველის სიგრძე (იზომება ნაბეჭდ სიმბოლოებში, მისი მნიშვნელობა ავტომატურად 20-ის ტოლია), maxlength - იმ სიმბოლოების მაქსიმალურ რიცხვს განსაზღვრავს, რომელიც მომხმარებლის მიერ იქნება შეტანილი ფორმის ველში. შეტანის ველის ავტოაქტივაციისთვის გამოიყენება label ტეგი.

<input type=”password”> ქმნის ველს პაროლის შესატანად.

გადამრთველები (radio button) - ფორმის ამ ელემენტის თავისებურება იმაში მდგომარეობს, რომ ის მომხმარებელს არჩევის საშუალებას აძლევს - შემოთავაზებული ვარიანტებიდან ვირჩევთ მხოლოდ ერთს. ჩვენი მაგალითის შემთხვევაში, ბრაუზერის ეკრანზე შედეგს

Male Female

უზრუნველყოფს კოდი:

```
male <input type="radio" name = "mf" value = "male">
female <input type = radio” name = "mf" value = "female">
```

ამრიგად, “გადამრთველები” (radio button) ფორმაში თქვენთვის უკვე ნაცნობი <input> ტეგის მეშვეობით შემოდის. მიაქციეთ ყურადღება, რომ „გადამრთველებისათვის” სახელი ერთი და იგივე იყოს, რადგან ისინი ფაქტიურად ერთი ელემენტია.

ელემენტი checkbox (ე. წ. “ალამი”) ასევე არჩევანის გაკეთების საშუალებას იძლევა. გადამრთველებისაგან (radio button) განსხვავებით, აქ მომხმარებელს როგორც ერთი, ასევე რამდენიმე ვარიანტის ერთდროულად მონიშვნა შეუძლია. ამ შემთხვევაში, input ტეგის type ატრიბუტი „checkbox” მნიშვნელობას იღებს.

რომელიმე „ალმის“ ან „გადამრთველის“ ავტომატურად მოსანიშნად, კოდში checked ატრიბუტი უნდა შეიტანოთ. რა თქმა

უნდა, მომხმარებელს შეუძლია შეცვალოს ეს მონიშვნა და შეარჩიოს ის, რომელიც თვითონ სურს.

შემდეგი ელემენტი მოცემულ კოდში არის select. მისი ატრიბუტი size მიუთითებს სიაში არსებული ჩამონათვალიდან ერთდროულად რამდენი ჩანს ბრაუზერის ეკრანზე. ჩვენს შემთხვევაში, ამ ატრიბუტის მნიშვნელობაა ოთხი და ეკრანზე ოთხი მსახიობის სახელი და გვარი ჩანს, თუმცა მათი რაოდენობა შესაბამის ჩამოშლად ველში მეტია. იმ შემთხვევაში, თუ ზომას (size) არ მივუთითებთ და რომელიმე ერთი მსახიობისთვის შევიტანთ selected ატრიბუტს, მაშინ ეკრანზე ეს ელემენტი გამოჩნდება როგორც მონიშნული.

<textarea> მრავალსტრიქონიანი ტექსტის შესატან ელემენტს ქმნის. მისი rows და cols ატრიბუტები ველის სიმაღლეს (rows - ველში სტრიქონების რაოდენობა) და სიგანეს (cols – ნაბეჭდი სიმბოლოების რაოდენობა სტრიქონში) განსაზღვრავს.

მოცემულ ფორმაში ასევე სერვერზე ინფორმაციის გაგზავნისა (submit) და გაგზავნის შეწყვეტის (reset) ღილაკები გვაქვს:

```
<input type="submit" value="OK">
<input type="reset" value="Cancel">
```

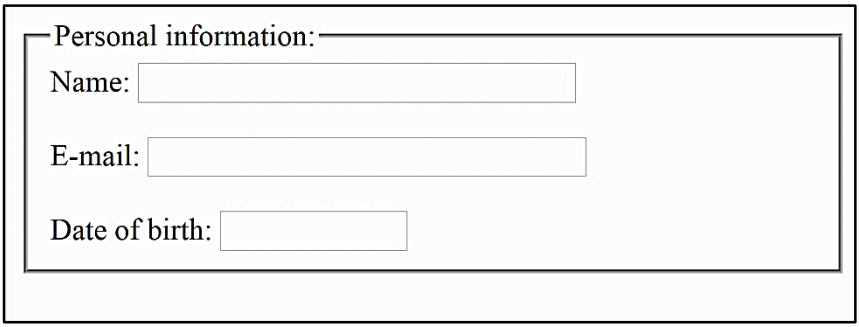
ღილაკებზე სტანდარტული წარწერების (submit, reset) შეცვლა value ტეგის საშუალებითაა შესაძლებელი.

ამ ფორმის შევსების და OK ღილაკზე დაჭერის შემთხვევაში დავინახავთ, რომ ეს ფორმა არ მუშაობს, რადგან არ არსებობს ფაილი formdata.php, რომელიც ფორმაში შეტანილ მონაცემებს დაამუშავებდა. როგორც ზემოთ აღვნიშნეთ, შესაბამისი პროგრამის (კოდის) დაწერა ვებტექნოლოგიების სხვადასხვა ენაზეა (JavaScript, PHP და სხვა) შესაძლებელი.

ფორმების შედგენის მაგალითები

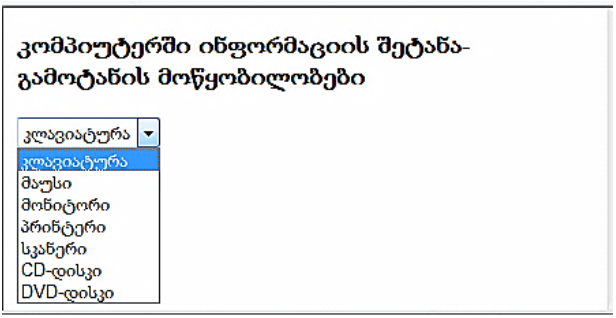
მაგალითი: ჩარჩო ფორმის ელემენტების გარეთ

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>forms4</title>
</head>
<body>
<form action="">
<fieldset>
<legend>Personal information:</legend>
Name: <input type="text" size="30"><br><br>
E-mail: <input type="text" size="30"><br><br>
Date of birth: <input type="text" size="10">
</fieldset>
</form>
</body>
</html>
```



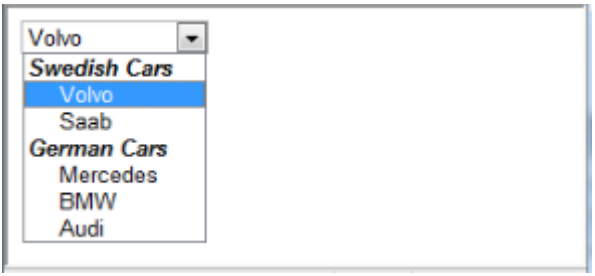
ქვემოთ განხილულ მაგალითში ფორმაში სიების შესაქმნელად <SELECT> წყვილი ტეგი გამოიყენება, რომელიც საშუალებას იძლევა მრავალი შესაძლო მნიშვნელობიდან ერთი ან რამდენიმე მნიშვნელობა ამოირჩეს. ჩუმათობის პრინციპით, სიის ველში მისი პირველი ელემენტი აისახება. სიის ელემენტი <option> ტეგის საშუალებით იქმნება.

```
<!DOCTYPE html>
<html>
<head>
<title>ფორმები</title>
</head>
<body>
<h3 align=left>კომპიუტერში ინფორმაციის შეტანა-გამოტანის მოწყობილობები</h3>
<form>
<select name=group>
<option> კლავიატურა
<option> მაუსი
<option> მონიტორი
<option> პრინტერი
<option> სკანერი
<option> CD-დისკი
<option> DVD-დისკი
</select>
</form>
</body>
</html>
```



<optgroup> ტეგის საშუალებით სიებში მონაცემთა რაიმე ნიშნით დაჯგუფებაა შესაძლებელი.

```
<!DOCTYPE html>
<html>
<body>
<select>
  <optgroup label="Swedish Cars">
    <option value="volvo">Volvo</option>
    <option value="saab">Saab</option>
  </optgroup>
  <optgroup label="German Cars">
    <option value="mercedes">Mercedes</option>
    <option value="BMW">BMW</option>
    <option value="audi">Audi</option>
  </optgroup>
</select>
 </body>
</html>
```



ღილაკის შექმნა

მიუხედავად იმისა, რომ ფორმებში ღილაკების შესაქმნელად <input> ტეგი გამოიყენება, HTML 5-ში ამ მიზნით კიდევ <button> ტეგის გამოყენებაა შესაძლებელი. <button> აქვს შემდეგი ატრიბუტები: type, name, value, autofocus, disabled, form, formaction, formenctype, formmethod და formnovalidate.

- ატრიბუტი type ღილაკის ტიპს განსაზღვრავს. მას შემდეგი მნიშვნელობების მიღება შეუძლია: button - ჩვეულებრივი, submit - ფორმის გაგზავნისა და reset - გაგზავნის შეწყვეტის ღილაკები;
- ატრიბუტი name ღილაკის სახელს განსაზღვრავს;
- ატრიბუტი value ღილაკის წარწერას განსაზღვრავს;
- ატრიბუტი autofocus მიუთითებს, რომ ღილაკი გვერდის ჩატვირთვისთანავე გახდეს აქტიური;
- ატრიბუტი disabled მიუთითებს, რომ ღილაკი უნდა იყოს უმოქმედო (გამორთული);
- form ატრიბუტის მნიშვნელობა იმ ფორმის სახელია, რომელსაც აღნიშნული ღილაკი ეკუთვნის;
- formaction ატრიბუტის მნიშვნელობა ის URL მისამართია, რომელზეც ფორმის მონაცემები უნდა გადაიგზავნოს. გამოიყენება მაშინ, როდესაც type=submit;
- ატრიბუტი formenctype მიუთითებს, თუ როგორ უნდა მოხდეს ფორმის მონაცემების კოდირება მისი გადაგ-

ზავნის წინ (გამოიყენება მხოლოდ type=submit შემთხვევაში). მისი შესაძლო მნიშვნელობებია: application, x-www-form-urlencoded, multipart/ form-data text და plain;

- ატრიბუტი formmethod ფორმის მონაცემების გაგზავნის მეთოდს მიუთითებს. გამოიყენება მხოლოდ, როცა type=submit. მისი შესაძლო მნიშვნელობებია: get და post;
- ატრიბუტი formnovalidate მიუთითებს იმაზე, რომ ფორმის მონაცემები გაგზავნის წინ არ უნდა შემოწმდეს. გამოიყენება, მხოლოდ, როცა type=submit.

„სახატავი ტილოს“ შექმნა

სახატავი არის ე. წ. „სახატავი ტილოს” შექმნა HTML 5-ში შესაძლებელია და ამისთვის <canvas> ტეგი გამოიყენება. იგი JavaScript-ის ან კლიენტის სხვა სკრიპტების დახმარებით ვებგვერდებზე სხვადასხვა სახის ფიგურის დახატვის საშუალებას იძლევა.

კანვასი არის მართკუთხა არე html გვერდზე. ჩუმათობის პრინციპით, მას არა აქვს ჩარჩო და კონტენტი. <canvas> ელემენტის შექმნის დროს id ატრიბუტი აუცილებლად უნდა მიეთითოს. იგი ელემენტის სახელს განსაზღვრავს, რომლითაც შემდგომში სკრიპტიდან მასზე მიმართვა მოხდება. აგრეთვე, აუცილებელია სახატავი არის ზომების მითითება, რისთვისაც width (სიგანე პიქსელებში) და height (სიმაღლე პიქსელებში) ატრიბუტები გამოიყენება:

```
<canvas id="myCanvas" width="200" height="100"></canvas>
```

ჩარჩოს შესაქმნელად, საჭიროა style ატრიბუტის გამოყენება:

```
<!DOCTYPE html>
<html>
<body>
<canvas id="myCanvas" width="200" height="100"
style="border:1px solid #000000;">
Your browser does not support the HTML5 canvas tag.
</canvas>
</body>
</html>
```

მულტიმედია

ბოლო პერიოდამდე აუდიო- და ვიდეორგოლების ვებგვერდებზე განთავსება მხოლოდ საკმაოდ დიდი HTML კოდის ჩაწერით ან დამატებითი პროგრამით იყო შესაძლებელი. HTML 5-ის გამოჩენის შემდეგ ეს პროცესი გამარტივდა და ამისთვის ჩვენთვის ნაცნობი <img> ტეგი გამოიყენება.

ფაილების ფორმატი და კოდირების ფორმატები

მულტიმედიური ფაილების მრავალი ფორმატი არსებობს. მულტიმედიური ფაილების კოდირება თითქმის ყოველთვის სხვადასხვა ფორმატით ხდება. პრაქტიკულად, მულტიმედიური ფაილების კოდირების ყველა ფორმატი მათ შეკუმშვას უჭერს მხარს. ამის გამო, ასეთი ტიპის ფაილების ზომები შედარებით მცირდება, რაც მათი ქსელით გადაცემის სიჩქარეზე დადებითად მოქმედებს. ქვემოთ ჩამოთვლილია და მოკლედ აღწერილი ვებდიზაინში გამოყენებული და ბრაუზერების მიერ მხარდაჭერილი მულტიმედიური ფაილების ფორმატები.

- *WAV* - მულტიმედიურ ფორმატებს შორის ყველაზე ძველი, გასული საუკუნის 90-იანი წლების დასაწყისში Microsoft-ის მიერ აუდიომონაცემების შესანახად იყო დამუშავებული და ამჟამადაც გამოიყენება. ასეთი ფორმატის ფაილებს wav გაფართოება აქვს;
- *OGG* - შედარებით ახალი ფორმატი. იგი Xiph.org არაკომერციული ორგანიზაციის მიერ აუდიო- და ვიდეოინფორმაციის შესანახად შეიქმნა. ამ ტიპის ფაილებს ogg (უნივერსალური ფორმატი), oga (აუდიოფაილების) და ogv (ვიდეოფაილების) გაფართოება აქვს.
- *MP4* - ესეც შედარებით ახალი ფორმატია. MP4 ფორმატი Motion Picture Expert Group (მოძრავი გამოსახულების საკითხების ექსპერტთა ჯგუფი) ორგანიზაციის მიერ 1998 წელს აუდიო- და ვიდეომონაცემების შენახვის მიზნით დამუშავდა. ამ ფორმატის ფაილებს mp4 გაფართოება აქვს;
- *QuickTime* ფორმატი Apple ფირმის მიერ 1989 წელს აუდიო- და ვიდეომონაცემების შესანახად იყო დამუშავებული. ასეთი ფორმატის ფაილებს mov გაფართოება აქვს;
- *PCM* (Pulse-Coded Modulation - იმპულსურ-კოდური მოდულაცია) - კოდირების ყველაზე მარტივი და ყველაზე ძველი ფორმატი. ის შემჭიდროებასაც კი არ უჭერს მხარს. იგი აუდიომონაცემების კოდირებისთვის გამოიყენება.
- *Vorbis* - კოდირების თანამედროვე ფორმატი. დამუშავებულია 2002 წელს Xiph.org-ის მიერ. გამოიყენება აუდიომონაცემების კოდირებისთვის.
- *AAC* (Advanced Audio Coding) კოდირების არც ისე ახალი ფორმატია. ის დამუშავდა 1997 წელს Motion Picture Expert Group-ის მიერ. გამოიყენება აუდიომონაცემების კოდირებისთვის.

- *Theora* - კოდირების ერთ-ერთი ყველაზე „ახალგაზრდა“ ფორმატი. ეს ფორმატი 2004 წელს გამოჩნდა და Xiph.org-მა შეიმუშავა. გამოიყენება ვიდეომონაცემების კოდირებისთვის.
- *H.264* - ძალიან „ახალგაზრდა“ ფორმატია. ის 2003 წელს Motion Picture Expert Group და Video Coding Experts Group (ვიდეოკოდირების ექსპერტთა ჯგუფი) ორგანიზაციების მიერ იყო წარმოდგენილი. განკუთვნილია ვიდეომონაცემების კოდირებისთვის.

თითქმის ყველა ზემოთ ჩამოთვლილი ფორმატი ღიაა, გამონაკლისი მხოლოდ QuickTime ფორმატის ფაილებია, რომლებიც Apple ფირმას მიეკუთვნება და H.264 კოდირების ფორმატი, რომელიც ასზე მეტი პატენტითაა დაცული.

ქვემოთ მოცემულ ცხრილში ბრაუზერებისა და კოდირების ფორმატების ურთიერთმხარდაჭერაა მოყვანილი.

აუდიო	ვიდეო	Firefox	Opera	Safari	Chrome
აუდიოინფორმაციის შემცველი ფაილები					
WAV-PCM		*	*		*
OGG-Vorbis		*	*		*
MOV-AAC				*	
აუდიო- და ვიდეოინფორმაციის შემცველი ფაილები					
OGG-Vorbis	OGG-Theora	*	*		*
MOV-AAC	MOV-H.264			*	
MP4-AAC	MP4-H.264			*	

როგორც ცხრილიდან ჩანს, სხვადასხვა ბრაუზერი სხვადასხვა ფორმატს უჭერს მხარს, ამან კი ვებგვერდების დამპროექტებლებს შეიძლება გარკვეული პრობლემები შეუქმნას.

MIME ტიპები

ქსელის საშუალებით სხვადასხვა სახის მონაცემი გადაიცემა: ვებგვერდები, გრაფიკული გამოსახულება, აუდიო- და ვიდეო-ფაილები, არქივი, შესრულებადი ფაილები და სხვა. ეს მონაცემები სხვადასხვა პროგრამისთვისაა გათვალისწინებული. ამასთან, მიმღები პროგრამა შეიძლება სხვადასხვანაირად მოიქცეს. ასე, მაგალითად, ვებგვერდის ან გრაფიკული გამოსახულების მიღების შემთხვევაში მიმღები პროგრამა მას ეკრანზე გამოიტანს, ხოლო თუ ეს ფაილი არქივი ან შესრულებადი ფაილია, მაშინ მას გახსნის ან დისკზე შეინახავს.

ქსელის საშუალებით გადაცემულ ყველა მონაცემს განსაკუთრებული აღნიშვნა მიენიჭება, რომელიც ცალსახად მის ბუნებაზე მიუთითებს. ამ აღნიშვნას MIME (Multipurpose Internet Mail Extensions - ინტერნეტის ფოსტის მრავალმიზნობრივი გაფართოება) ტიპი ეწოდება. პროგრამების მონაცემებს ამ ტიპს მისი გამგზავნი ანიჭებს, მაგალითად, ვებსერვერი. ხოლო მიმღები პროგრამა MIME ტიპის მიხედვით განსაზღვრავს, მოცემული ბრაუზერი მხარს უჭერს თუ არა მიღებულ მონაცემებს და თუ უჭერს, მაშინ რა მოქმედება უნდა განახორციელოს ამ მონაცემებთან დაკავშირებით.

ვებგვერდის MIME ტიპია text/html, GIF ფორმატის გრაფიკული გამოსახულების - image/gif, შესრულებადი ფაილის - application/x-msdownload, ხოლო ZIP არქივის - application/x-zip-compressed. თავისი MIME ტიპები აქვს მულტიმედიურ ფაილებსაც.

ქვემოთ, ცხრილში მოცემულია მულტიმედიური ფაილის ფორმატის MIME ტიპები, რომლებსაც თანამედროვე ბრაუზერები უჭერს მხარს. როგორც ცხრილიდან ჩანს, ერთი ფორმატის ფაილებს შეიძლება რამდენიმე MIME ტიპი ჰქონდეს.

ფაილის ფორმატი	MIME ტიპი
WAV	audio/wave audio/wav audio/x-wav audio/x-pn-wav
OGG	audio/ogg (აუდიოფაილებისათვის) video/ogg (ვიდეოფაილებისათვის) application/ogg
MP4	video/mp4
QuickTime	video/quicktime

აუდიორგოლის ჩასმა

ვებგვერდზე აუდიორგოლის ჩასასმელად HTML 5 ენა <audio> ტეგს იყენებს. ფაილის ინტერნეტმისამართს, რომელშიც მოცემული აუდიორგოლი ინახება, src ატრიბუტის საშუალებით მივუთითებთ:

```
<audio src="sound.wav"></audio>
```

ბრაუზერმა შეიძლება აუდიოფაილი მაშინვე ჩატვირთოს და გაუშვას ან მხოლოდ ჩატვირთოს და არაფერი არ გააკეთოს. მან ასევე შეიძლება გამოიტანოს მართვის ელემენტები, რომლის საშუალებითაც მომხმარებელს შეუძლია აუდიოფაილი გაუშვას, შეაჩეროს, წინ და უკან გადაახვიოს, არეგულიროს ხმა. ყოველივე ეს <audio> ტეგის ატრიბუტების საშუალებით რეგულირდება.

ბრაუზერი აუდიორგოლის ავტომატურად გაშვებას ვერ უზრუნველყოფს, ამისათვის, <audio> ტეგში autoplay ატრიბუტი უნდა მივუთითოთ. მას არ გააჩნია მნიშვნელობა - მთავარია იგი ეწეროს ტეგში და აუდიორგოლი ჩატვირთვისთანავე შესრულებაზე გაეშვება:

```
<p>ახლა გაიგონებთ ხმას!</p>
```

```
<audio src="sound.ogg" autoplay></audio>
```

თუ <audio> ტეგში მნიშვნელობის გარეშე controls ატრიბუტს მივუთითებთ, მაშინ ბრაუზერი აუდიორგოლის გაშვების მართვის ელემენტებს გამოიტანს. მასში შედის გაშვებისა და გაჩერების ღილაკები, დროის მართვის პანელი და ხმის რეგულატორი:

```
<p>ხმის გასაგებად დააჭირეთ გამშვებ ღილაკს.</p>
<audio src="sound.ogg" controls></audio>
```

თუ <audio> ტეგში მნიშვნელობის გარეშე autobuffer ატრიბუტს მივუთითებთ, მაშინ ბრაუზერი ვებგვერდის ჩატვირთვის შემდეგ მაშინვე აუდიორგოლსაც ჩატვირთავს, რათა შეყოვნება არ მოხდეს. ამასთან, უნდა გვახსოვდეს, რომ ამ ატრიბუტის გამოყენება მხოლოდ იქ შეიძლება, სადაც ატრიბუტი autoplay არ არის გამოყენებული.

ქვემოთ აუდიორგოლის გამოტანის მაგალითია მოცემული:

```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="content-type" content="text/html; charset=utf-8">
<title>audio ტეგი</title>
</head>
<body>
<h1>audio ტეგი</h1>
<p>ტეგი audio ვებგვერდზე აუდიორგოლის ჩასასმელად გამოიყენება.</p>
<h6>მაგალითი:</h6>
<pre>&lt;audio src=&quot;kalimba.wav&quot; controls&gt;&lt;/audio&gt; </pre>
<h6>შედეგი:</h6>
```

```
<audio src="kalimba.wav" controls></audio>
</body>
</html>
```

შექმნილი ვებგვერდი და აუდიორგოლი ერთსა და იმავე საქაღალდეში უნდა შევინახოთ.

ვიდეორგოლის ჩასმა

ვებგვერდზე ვიდეორგოლის ჩასასმელად <video> ტეგი გამოიყენება. ვიდეოფაილის ინტერნეტმისამართი ამ ტეგის src ატრიბუტით მიეთითება:

```
<video src="film.ogg"></video>
```

ბრაუზერმა შეიძლება ვიდეოფაილი მაშინვე ჩატვირთოს და გაუშვას ან მხოლოდ ჩატვირთოს და არაფერი არ გააკეთოს. მან ასევე შეიძლება გამოიტანოს მართვის ელემენტები, რომლის საშუალებითაც მომხმარებელს შეუძლია ვიდეოფაილი გაუშვას, შეაჩეროს, წინ და უკან გადაახვიოს, არეგულიროს ხმა. ყოველივე ეს <video> ტეგის ატრიბუტების საშუალებით რეგულირდება.

<video> ტეგს autoplay, controls და autobuffer ატრიბუტები აქვს. მაგალითად,

```
<video src="film.ogg" autoplay controls></video>
```

თუ ვიდეორგოლის ჩვენება ჯერ არ არის გაშვებული, მაშინ ჩვენების პანელში მისი პირველი კადრი ან საერთოდ არაფერი არ იქნება გამოტანილი (ეს დამოკიდებულია ბრაუზერზე). შეგვიძლია მივუთითოთ ის გრაფიკული გამოსახულება, რომელიც პირველად „ფარდის“ სახით იქნება გამოტანილი. ამისათვის, <video> ტეგის poster ატრიბუტი გამოიყენება. მისი მნიშვნელობა საჭირო გრაფიკული ფაილის ინტერნეტმისამართზე მიუთითებს. მაგალითად,

```
<video src="film.ogg" controls poster="filmposter.jpg"> </video>
```

შენიშვნა. თუ საჭირო ფორმატის ვიდეორგოლს ვერ მოძებნით, მაშინ შეიძლება შექმნათ ის სხვა ფორმატში შენახული ვიდეორგოლის კოდირების შეცვლით. ამისათვის, SUPER © უტილიტა გამოიყენება, რომელიც შემდეგ ინტერნეტ-მისამართზე შეიძლება იპოვოთ:

http://www.erightsoft.com/ SUPER.html.

ქვემოთ მოცემულია ვებგვერდზე ვიდეორგოლის ჩასმის მაგალითი:

```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="content-type" content="text/html; charset=utf-8">
<title>video ტეგი</title>
</head>
<body>
<h1>video ტეგი</h1>
```

```
    <p>video ტეგი ვებგვერდზე ვიდეორგოლის ჩასასმელად გამოიყენება.</p>
    <h6>მაგალითი:</h6>
    <pre>&lt;video src=&quot; wildlife.wmv&quot; controls&gt;&lt;/video&gt; </pre>
    <h6>შედეგი:</h6>
    <video src="wildlife.wmv" controls></video>
    </body>
    </html>
```

Youtube-დან ვიდეოს ჩასასმელად კოდს თვითონ youtube გვაძლევს.

მაგალითი:

```
    <!DOCTYPE html>
    <head>
```

```
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8"
/>
    <title>Html5 video</title>
    </head>
    <body>
    <p>Video from youtube</p>
    <embed width="420" height="315"
    src="http://www.youtube.com/embed/HuKRX_Dwdzc">
    </body>
    </html>
```

CSS

CSS (Cascading Style Sheets) - სტილების კასკადური ცხრილები - ამა თუ იმ HTML ტეგების სტილური აღწერების ნაკრებია, რომელიც შეიძლება როგორც ცალკეული ტეგის - ელემენტის, ასევე მთელი საიტის ყველა იდენტური ელემენტის მიმართ იყოს გამოყენებული. CSS თავისი არსით HTML-ის დამატებაა და მის შესაძლებლობებს მნიშვნელოვნად აფართოებს.

CSS-ის შესწავლის აუცილებლობის უკეთესად გააზრების მიზნით რამდენიმე მაგალითს მოვიყვანთ:

HTML-ს ინსტრუმენტების საკმაოდ შეზღუდული ნაკრები გააჩნია, რომლის საშუალებითაც ვებდიზაინერის ზოგიერთი ჩანაფიქრის განხორციელება შეუძლებელია. მაგალითად, დამწყები ვებდიზაინერისთვის „მტკივნეული“ საკითხები ავიღოთ: როგორ მოვაცილოთ ბმულებს ხაზგასმა ან როგორ გავაკეთოთ ისე, რომ ბმულზე კურსორის გადატარებისას მისი ფერი შეიცვალოს? მხოლოდ HTML-ის გამოყენებით ამის გაკეთება შეუძლებელია. ასეთი პრობლემები ძალიან ბევრია და CSS-ს სწორედ მსგავსი საკითხების გადაჭრა შეუძლია.

დავუშვათ, შევქმენით საიტი, რომელიც 100 გვერდს მოიცავს. რაღაც მიზეზების გამო მისი დიზაინის შეცვლა დაგვჭირდა. საიტის 100 გვერდის გადასაკეთებლად ძალიან დიდი დრო დაგვეხარჯება. ახლა წარმოვიდგინოთ, რომ საიტის მთელი დიზაინის აღწერა შეიძლება ცალკე ფაილში განვათავსოთ: დავუშვათ, მთელი დოკუმენტისთვის <h1> სათაური წითელი ფერის იყოს, <p> აბზაცები დახრილად დაიწეროს, <a> ბმულებზე ხაზგასმა არ მოხდეს, ყველა გვერდის ფონი ცისფერი იყოს და ა.შ. მას შემდეგ, რაც უკვე CSS ფაილი გვაქვს, რომელშიც მთელი საიტის დიზაინია აღწერილი, საჭიროა ჩვენი საიტის ასივე გვერდი ამ ერთ ფაილს დავუკავშიროთ და მისგან საიტის დიზაინის შესახებ საჭირო

ინფორმაცია მივიღოთ. ამის შემდეგ, როდესაც დაგვჭირდება ჩვენ საიტში ყველა სათაურის წითელი ფერი მწვანე ფერით შევცვალოთ, ასივე გვერდის გახსნა, მასში <h1> ტეგების მოძებნა და წითელი ფერის მწვანით შეცვლა არ დაგვჭირდება. ჩვენ მხოლოდ CSS ფაილი უნდა გავხსნათ და მასში <h1> ელემენტის წითელი ფერი მწვანე ფერით შევცვალოთ. საიტის ყველა გვერდისთვის ყველა სათაურის ფერი მწვანე გახდება.

იმის გამო, რომ CSS ერთი და იგივე ელემენტის განმეორებადი სტილური აღწერების ერთ ფაილში გატანის საშუალებას იძლევა, ეს HTML დოკუმენტების მნიშვნელოვან „განტვირთვას“ იწვევს, რაც მოცულობის, დროის და ტრაფიკის ეკონომიაა. HTML კოდი მცირე ზომის ხდება და კითხვისა და რედაქტირებისათვის უფრო მოხერხებულია.

დღეს, CSS ვებგვერდის დიზაინის აწყობის ყველაზე გავრცელებული, მარტივი და მოხერხებელი ტექნოლოგიაა. CSS ტექნოლოგიის მხარდაჭერა ყველა თანამედროვე ბრაუზერს აქვს, ამიტომ ვებსივრცეში თითქმის არ არსებობს ვებგვერდი, რომელიც აღნიშნულ ტექნოლოგიას არ იყენებს.

CSS კოდის HTML დოკუმენტთან დაკავშირება

CSS ტექნოლოგია HTML-ის ნებისმიერ ტეგთან მუშაობის საშუალებას იძლევა. HTML-ის გამოყენებით ვებგვერდის სტრუქტურის აგება ხდება, CSS კი მოცემული სტრუქტურის დახვეწასა და დიზაინის თვალსაზრისით მის სრულყოფას უზრუნველყოფს.

CSS კოდის HTML დოკუმენტში ჩართვა სამი ხერხითაა შესაძლებელი. ქვემოთ სამივე მეთოდი განვიხილოთ.

მეთოდი 1. style ატრიბუტის საშუალებით CSS-ის კოდის ჩასმა.

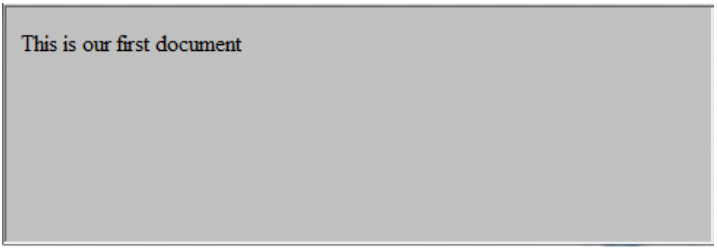
HTML დოკუმენტის მიმართ CSS-ის გამოყენება HTML ობიექტის style ატრიბუტის საშუალებით შეიძლება.

მაგალითი: HTML დოკუმენტისთვის CSS-ის საშუალებით ფონის მიცემა.

შევქმნათ practcss1_1.html დოკუმენტი და დავწეროთ შემდეგი კოდი:

```
<!DOCTYPE html>
<html>
<head>
<title>practcss1_1.html </title>
</head>
<body style="background-color:silver;">
<p>This is our first document</p>
</body>
</html>
```

შევინახოთ შექმნილი დოკუმენტი და გავხსნათ რომელიმე ბრაუზერის საშუალებით. შედეგი ასეთი იქნება:



ზემომოყვანილ მაგალითში

```
<body style="background-color:silver;">
```

ბრძანება დოკუმენტისთვის ვერცხლისფერი ფონის მიცემას უზრუნველყოფს.

CSS-ის სტილების ჩაწერა შესაბამისი HTML ობიექტისათვის რომელიმე თვისების ჩაწერას და ამ თვისებისთვის რაიმე მნიშვნელობის მინიჭებას მოიცავს.

CSS სტილების ჩაწერის სინტაქსი ასეთია:

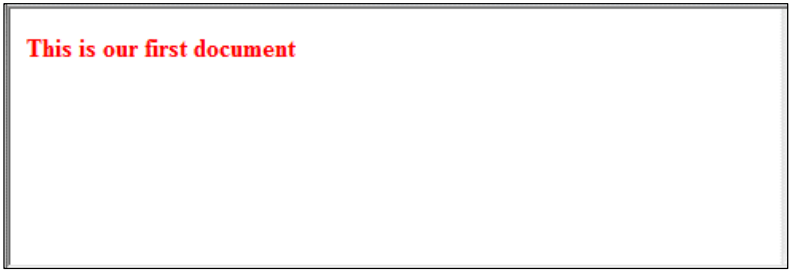
```
html ობიექტის თვისება: მნიშვნელობა;
```

მოცემულ შემთხვევაში, background-color CSS-ის საშუალებით ჩაწერილი HTML ობიექტის თვისებაა, რომლის მნიშვნელობაა silver.

მაგალითი: CSS-ის საშუალებით აბზაცში ტექსტის დაფორმატება (ტექსტი იყოს წითელი ფერის, გამუქებული). ამისათვის შევქმნათ practcss1_2.html დოკუმენტი და დავწეროთ შემდეგი კოდი:

```
<!DOCTYPE html>
<html>
<head>
<title>practcss1_2.html</title>
</head>
<body>
<p style="color:red; font-weight:bold;">This is our first document</p>
</body>
</html>
```

მიღებული შედეგი:



ჩვენს პროგრამაში

```
<p style="color:red; font-weight:bold;">
```

ბრძანება აბზაცში ტექსტისთვის წითელი ფერის მიცემას და მის გამუქებას უზრუნველყოფს.

მოცემულ შემთხვევაში, color და font-weight CSS-ის საშუალებით ჩაწერილი HTML ობიექტის თვისებებია, რომელთა მნიშვნელობები, შესაბამისად, red და bold-ია.

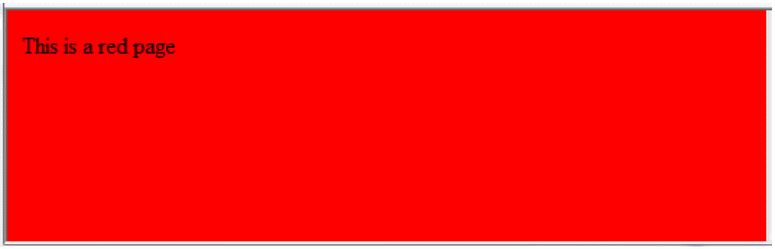
მეთოდი 2. HTML დოკუმენტში CSS კოდის <style> ტეგის გამოყენებით ჩასმა.

მაგალითი: დოკუმენტის ფონის ფერის მიცემა <style> ტეგის გამოყენებით.

შევქმნათ practcss1_3.html დოკუმენტი და დავწეროთ შემდეგი კოდი:

```
<!DOCTYPE html>
<html>
<head>
<title>practcss1_3</title>
<style>
body
    {
    background-color:#ff0000;
    }
</style>
</head>
<body>
<p>This is a red page</p>
</body>
</html>
```

მიღებული შედეგი იქნება:



HTML დოკუმენტში CSS კოდის ჩასმის დროს <style> კონტეინერი <head> განყოფილებაში თავსდება. <style> ტეგში მოთავსებული სელექტორები ფორმალურად შეიძლება ასე ჩავწეროთ:

```
სელექტორი
 {
    თვისება: მნიშვნელობა;
}
```

ფიგურულ ფრჩხილებში იწერება სელექტორის ტანი, რომელიც კონკრეტული სელექტორის CSS სტილებს განსაზღვრავს.

ზემოთ განხილულ მაგალითში სელექტორს body წარმოადგენს:

```
<style>
Body
    {
    background-color:#ff0000;
    }
</style>
```

ზოგადად, სელექტორი მიუთითებს, თუ რომელი HTML ტეგის (ელემენტის) მიმართ გამოიყენება ამ სელექტორის ტანში გაწერილი თვისებები.

ჩვენს შემთხვევაში body სელექტორს აქვს თვისება background-color, რომელიც დოკუმენტის ტანისთვის ფონის ფერის შერჩევას უზრუნველყოფს. მოცემულ შემთხვევაში მისი მნიშვნელობაა #ff0000.

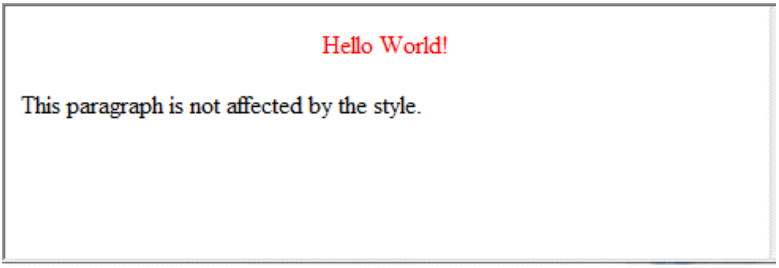
HTML ელემენტისთვის სტილის დანიშვნა id და class იდენტიფიკატორების საშუალებითაც შეიძლება მოხდეს.

მაგალითი: id იდენტიფიკატორის გამოყენება.

შევქმნათ practcss1_4.html დოკუმენტი:

```
<!DOCTYPE html>
<html>
<head>
<title>practcss1_4</title>
<style>
#style1
 {
      text-align:center;
      color:red;
 }
</style>
</head>
<body>
<p id="style1">Hello World!</p>
<p>This paragraph is not affected by the style.</p>
</body>
</html>
```

მივიღებთ შედეგს:

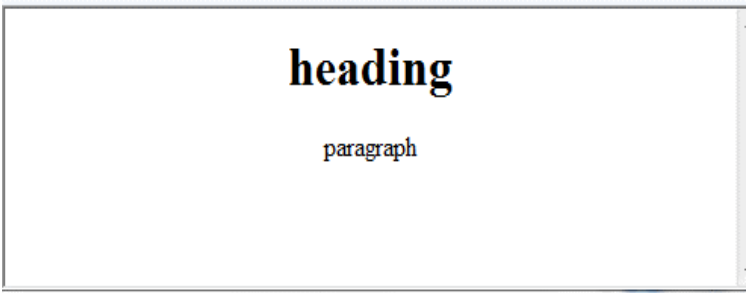


მაგალითი: class სელექტორის გამოყენება.

შევქმნათ practcss1_5.html დოკუმენტი და დავწეროთ შემდეგი კოდი:

```
<!DOCTYPE html>
<html>
<head>
<title>practcss1_5</title>
<style>
.center
{
text-align:center;
}
</style>
</head>
<body>
<h1 class="center">heading</h1>
<p class ="center">paragraph</p>
</body>
</html>
```

ქვემოთ ნაჩვენებია მიღებული შედეგი:



ამ მაგალითში ყველა HTML ელემენტი, რომლისთვისაც class="center", ცენტრში მოთავსდება, რისი განსაზღვრაც <style> ... </style> ტეგებში მოთავსებული ბრძანებით ხდება:

```
.center {
text-align:center;
}
```

ცენტრში მოთავსდება სათაური და აბზაცი, რომელიც ჩვენს მაგალითში პირობითად თითო სიტყვისგან შედგება:

```
<h1 class="center">heading</h1>
<p class ="center">paragraph</p>
```

ასევე შესაძლებელია, რომ კლასის მოქმედება მხოლოდ განსაზღვრულ HTML ელემენტზე გავრცელდეს, თუ <style> ... </style> ტეგებს შორის center კლასი მხოლოდ p ელემენტისთვის იქნება განსაზღვრული:

```
p.center {
text-align:center;
}
```

ამ შემთხვევაში ის მხოლოდ p ელემენტისთვის იმოქმედებს:

```
<h1 class="center">This heading will not be affected</h1>
<p class="center">This paragraph will be center-aligned.</p>
```

აბზაცის შესაბამისი ტექსტი ცენტრში განთავსდება, ხოლო <h1> ელემენტისთვის კი class=”center” ბრძანება არ იმუშავებს, რადგანაც შესაბამისი სტილი მხოლოდ აბზაცის p ტეგისთვისაა დანიშნული.

მეთოდი 3. სტილების გარე ცხრილის გამოყენება.

ეს მეთოდი ე. წ. სტილების გარე ცხრილზე მიმართვას გულისხმობს. სტილების გარე ცხრილი .css გაფართოების მქონე ტექსტური ფაილია.

დავუშვათ, გვაქვს სტილების ცხრილი, რომელსაც style.css დავარქვით და ის style საქაღალდეშია მოთავსებული. ამოცანა იმაში მდგომარეობს, რომ HTML დოკუმენტიდან (default.html) სტილების ცხრილზე (style.css) მიმართვა შეიქმნას. ეს შეიძლება HTML კოდის ერთი სტრიქონის საშუალებით გაკეთდეს:

```
<link rel="stylesheet" type= "text/css" href="style/style.css"/>
```

ეს სტრიქონი HTML დოკუმენტში <head> და </head> ტეგებს შორის უნდა განთავსდეს. იგი ბრაუზერს მიუთითებს, რომ მან HTML ფაილის ეკრანზე ასახვის წესი CSS ფაილიდან უნდა აიღოს. ყველაზე მნიშვნელოვანი ის არის, რომ სტილების ერთ ცხრილს შეიძლება რამდენიმე HTML დოკუმენტმა მიმართოს. სხვა სიტყვებით რომ ვთქვათ, ერთი CSS ფაილი მრავალი HTML დოკუმენტის ბრაუზერის ეკრანზე ასახვის სამართავად შეიძლება იქნეს გამოყენებული. სტილების ცხრილში შეტანილი ცვლილება ყველა იმ HTML დოკუმენტში აისახება, რომლებიც სტილების ამ ცხრილს მიმართავს.

მაგალითი: სტილების გარე ცხრილის შექმნა და მისი HTML დოკუმენტთან დაკავშირება.

შევქმნათ practcss1_6.html და style.css დოკუმენტები და ისინი ერთ საქაღალდეში მოვათავსოთ.

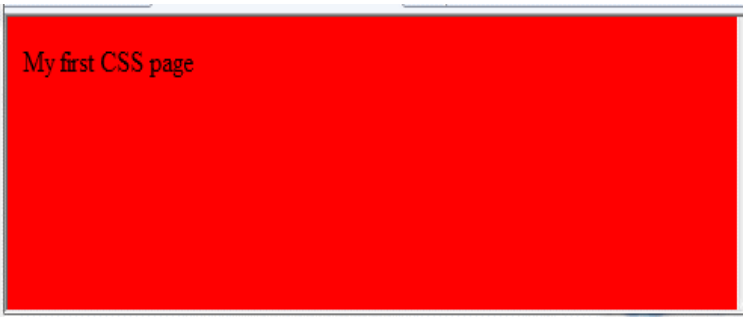
practcss1_6.html დოკუმენტს აქვს შემდეგი სახე:

```
<!DOCTYPE html>
<html>
<head>
<title>
სტილების გარე ცხრილთან დაკავშირება
</title>
<link rel="stylesheet" type="text/css" href="style.css"/>
</head>
<body>
My first CSS page
</body>
</html>
```

Style.css-ში ვწერთ კოდს:

```
body  {
background-color: #ff0000;
}
```

მიღებული შედეგი მოცემულია ქვემოთ:



Practcss1_6.html ფაილის გახსნის შედეგად დავინახავთ, რომ გვერდის ფონის ფერი იქნება წითელი. ეს ნიშნავს, რომ შევქმენით

სტილების გარე ფაილი და იგი HTML დოკუმენტთან დავაკავშირეთ. ფონის ფერი დოკუმენტისთვის მითითებულია სტილების ფაილში, რომელიც ჩვენს შემთხვევაში მხოლოდ ერთ სტრიქონს შეიცავს:

```
body    { background-color: #ff0000;}
```

დავუშვათ, გვაქვს შემთხვევა, როდესაც HTML დოკუმენტი-დან სტილების გარე ცხრილზე ხდება მიმართვა და იმავე დოკუმენტშივე გვაქვს <style>…</style> ტეგების გამოყენებით CSS კოდი ჩასმული, თან სტილების ორივე ცხრილის შემთხვევაში სტილი ერთი და იმავე სელექტორისთვისაა დანიშნული. საინტერესოა, რა შეიძლება მოხდეს ამ დროს? რომელი სტილი აღმოჩნდება უფრო პრიორიტეტული?

მაგალითი: პრიორიტეტები რამდენიმე კასკადური ცხრილის შემთხვევაში.

შევქმნათ დოკუმენტები: practcss1_7.html და style.css. მოვათავსოთ ისინი ერთ საქაღალდეში.

practcss1_7.html დოკუმენტში გვაქვს შემდეგი სახის კოდი:

```
<!DOCTYPE html>
<html>
<head>
<title>practcss1_7.html</title>
      <link rel="stylesheet" type="text/css" href="style.css"/>
<style>
h3 { text-align:right;
font-size:20pt;
}
</style>
</head>
```

```
<body>
<h3>
Tea Todua
</h3>
</body>
</html>
```

style.css-ში ვწერთ კოდს:

```
h3 {
color:red;
text-align:left;
font-size:8pt;
}
```

მიღებული შედეგი:

სტილების გარე ცხრილში h3 სელექტორს აქვს სამი თვისება:

```
h3 {
color:red;
text-align:left;
font-size:8pt;
}
```

იგივე h3 სელექტორისთვის სტილების შიგა ცხრილში შემდეგი თვისებები გვაქვს:

```
h3 {
text-align:right;
font-size:20pt;
}
```

ასეთ შემთხვევაში მოხდება ფერის თვისების მემკვიდრეობით გადაცემა სტილების გარე ცხრილიდან შიგა ცხრილისთვის, ტექსტის დაშორება კიდიდან და შრიფტის ზომა სტილების შიგა ცხრილიდან აიღება. ამის გამო, h3 სელექტორის თვისებებს ექნება სახე:

```
color:red;
text-align:right;
font-size:20pt;
```

ყველაზე მაღალი პრიორიტეტი <style> ატრიბუტით განსაზღვრულ სტილებს აქვს, რაც ნიშნავს, რომ ის „გადაფარავს“ <head>... </head> ტეგის შიგნით, გარე სტილების ფაილში განსაზღვრულ ან ბრაუზერის მიერ ჩუმათობის პრინციპით განსაზღვრულ სტილებს.

თუ სტილების გარე ცხრილთან მიმართვა <head>... </head> ტეგის შიგნით მოთავსებული სტილების ცხრილის შემდეგაა განთავსებული, მაშინ გარე სტილების ცხრილი შიგა ცხრილების სტილს „გადაფარავს“.

ფონური გამოსახულება CSS-ში

CSS-ში background თვისება გამოიყენება ფონური გამოსახულების სხვადასხვა სახის ეფექტის მისაღებად. ფონური ეფექტების მისანიჭებლად გამოიყენება შემდეგი CSS თვისებები:

- background-color

- background-image
- background-repeat
- background-position
- background-attachment

background-color თვისება ელემენტის ფონის ფერს აღწერს.

მაგალითი:

background-color თვისების გამოყენება დოკუმენტის ფონის ფერისა და სათაურის ფონის ფერის მისათითებლად.

```
<!DOCTYPE html>
<html>
<head>
<title>practcss1_8.html</title>
<style>
Body {
background-color:#b0c4de;
}
h1 {
color:red; background-color:yellow;
}
</style>
</head>
<body>
<h1 align="center">What is CSS!</h1>
<p>Cascading Style Sheets (CSS) is a style sheet language used for describing the look and formatting of a document written in a markup language. While most often used to change the style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any kind of XML document, including plain XML, SVG and XUL. Along with HTML and JavaScript, CSS is a
```

cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

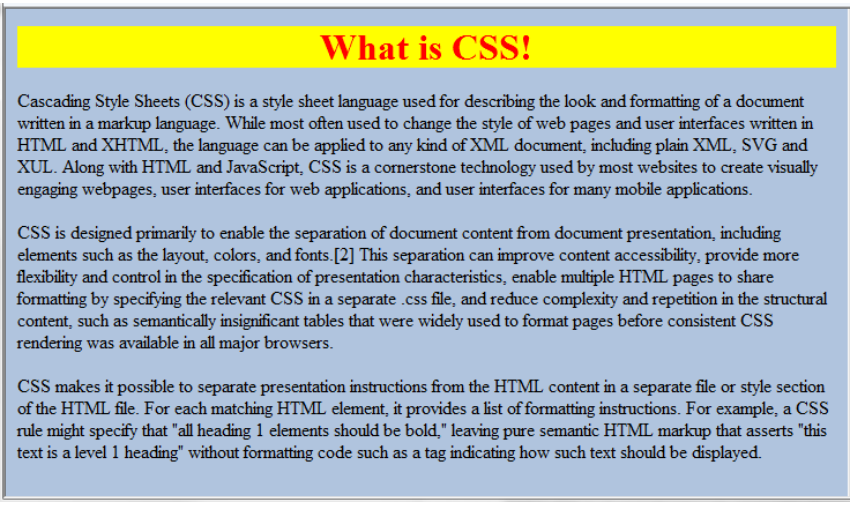
CSS is designed primarily to enable the separation of document content from document presentation, including elements such as the layout, colors, and fonts.[2] This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content, such as semantically insignificant tables that were widely used to format pages before consistent CSS rendering was available in all major browsers.

CSS makes it possible to separate presentation instructions from the HTML content in a separate file or style section of the HTML file. For each matching HTML element, it provides a list of formatting instructions. For example, a CSS rule might specify that "all heading 1 elements should be bold," leaving pure semantic HTML markup that asserts "this text is a level 1 heading" without formatting code such as a <bold> tag indicating how such text should be displayed.

</body>

</html>

მიღებული შედეგი მოცემულია ქვემოთ:



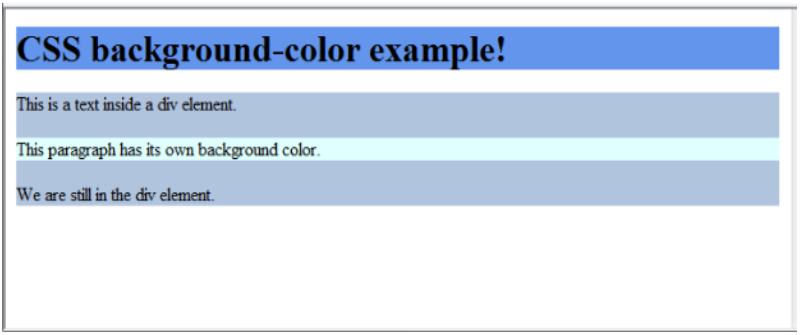
<body>-ის შიგნით თავსდება HTML დოკუმენტის მთელი შემცველობა. ამიტომ, მთელი გვერდის ფონის შესაცვლელად საჭირო background-color თვისება ელემენტ <body>-ის მიმართ უნდა გამოვიყენოთ. ამ თვისების გამოყენება სხვა ელემენტების, მათ შორის სათაურისა და ტექსტის მიმართაცაა შესაძლებელი. ყურადღება მიაქციეთ იმას, რომ h1 ტეგისთვის ორი თვისების - სათაურისა და სათაურის ფონის ფერის მითითება მოხდა.

მაგალითი: h1, p და div ელემენტების მიმართ background-color თვისების გამოყენება.

```
<!DOCTYPE html>
<html>
<head>
<title>practcss1_9.html</title>
<style>
h1{
background-color:#6495ed;
```

```
}
P {
background-color:#e0ffff;
}
div{
background-color:#b0c4de;
}
</style>
</head>
<body>
<h1>CSS background-color example!</h1>
<div>This is a text inside a div element.
<p>This paragraph has its own background color.</p>
We are still in the div element. </div>
</body>
</html>
```

მივიღებთ შედეგს:



ამ მაგალითში h1, p და div ელემენტებისთვის ფონის ფერი მიეთითა. div ბლოკური ელემენტის შიგნით მოთავსებული p ელემენტისთვის ტექსტის ფერი განისაზღვრა მისთვის დანიშნული სტილის შესაბამისად, ხოლო იმ ტექსტებისთვის, რომლებიც

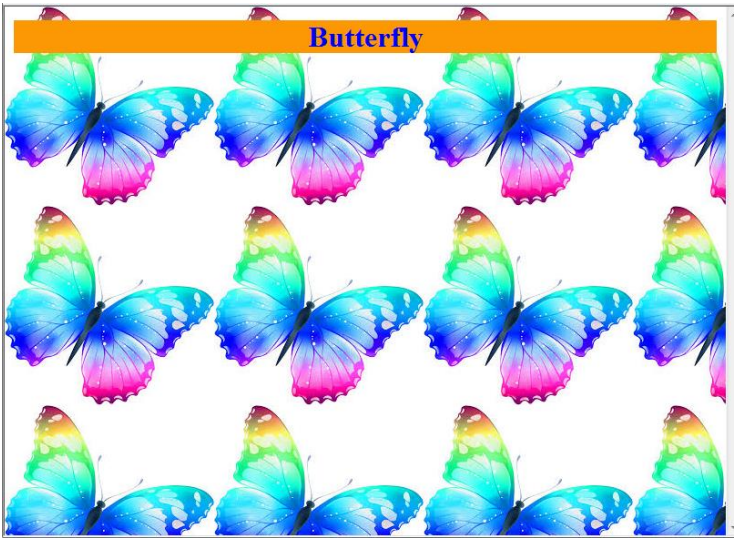
div ელემენტს მიეკუთვნება, განისაზღვრა იმ სტილის შესაბამისად, რომელიც div-ს დაენიშნა.

background-image თვისება ფონური გამოსახულების ჩასასმელად გამოიყენება.

მაგალითი: ფონური გამოსახულების ჩასმა.

```
<!DOCTYPE html>
<html>
<head>
<title>practcss1_10.html</title>
<style>
body {
background-image: url("butterfly.jpg");
}
    h1 {
    color: blue;
    background-color: #fc9804;
     }
      </style>
      </head>
<body>
<h1 align="center" >
Butterfly
</h1>
</body>
</html>
```

მიღებული შედეგი:



სურათის ფონად ჩასასმელად <body> ტეგში გამოიყენება background-image თვისება და სურათის ადგილმდებარეობა მიეთითება:

```
url("butterfly.jpg").
```

ეს ნიშნავს, რომ ფაილი იმავე საქაღალდეშია, რომელშიც არის ჩვენი HTML დოკუმენტი. რა თქმა უნდა, შესაძლებელია სხვა საქაღალდეში ან ინტერნეტში არსებული სურათებისადმი მიმართვაც, ასეთ შემთხვევაში მისი სრული მისამართი უნდა მიეთითოს: `url("../images/butterfly.jpg")`

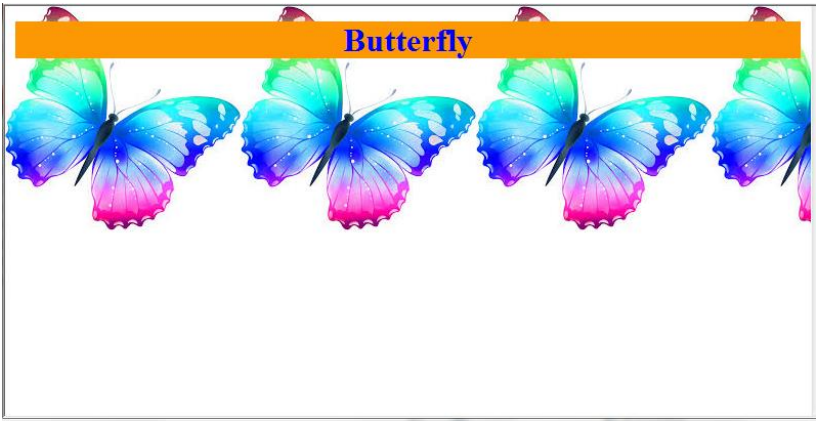
ან

```
url("http://www.teabesotemo.ge/butterfly.jpg")
```

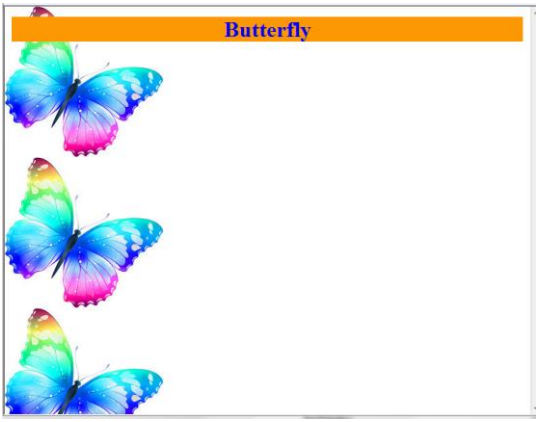
url("images/butterfly.jpg") მისამართის მითითება ნიშნავს, რომ images საქაღალდე იმავე საქაღალდეშია მოთავსებული, რომელშიც HTML დოკუმენტი.

background-repeat თვისება ფონური გამოსახულების გამეორებას მართავს. ამ თვისებას ოთხი მნიშვნელობის მიღება შეუძლია:

ა) background-repeat:repeat-x (ფონური გამოსახულება მეორდება ჰორიზონტალზე)



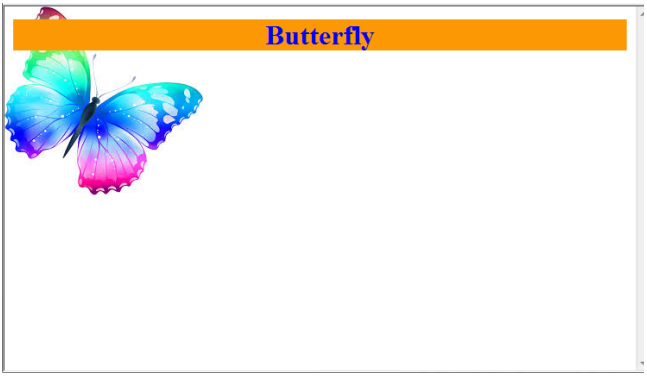
ბ) background-repeat:repeat-y (ფონური გამოსახულება მეორდება ვერტიკალზე).



გ) background-repeat:repeat (ფონური გამოსახულება მეორდება ჰორიზონტალზეც და ვერტიკალზეც).

ეს შემთხვევა უკვე განვიხილეთ ზემოთ განხილულ მაგალითში, როდესაც ფონური გამოსახულება ვერტიკალზეც და ჰორიზონტალზეც repeat თვისების მითითების გარეშეც გამეორდა, რადგანაც სისტემაში ჩუმათობის პრინციპის თანახმად მიღებულია, რომ გამოსახულება გამეორდეს და მთელი ეკრანი დაფაროს ჰორიზონტალურადაც და ვერტიკალურადაც.

დ) background-repeat:no-repeat (ფონური გამოსახულება არ მეორდება).



background-attachment თვისება განსაზღვრავს, ფონური სურათი დაფიქსირებულია (ბლოკირებულია) თუ გვერდის შემცველობასთან ერთად მოძრაობს.

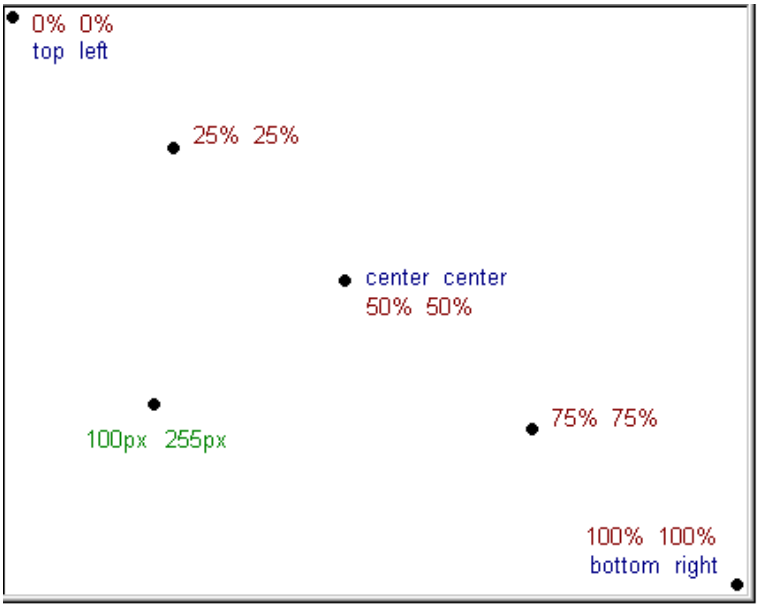
ცხრილში background-attachment-ის ორი მნიშვნელობაა ნაჩვენები.

მნიშვნელობა	აღწერა
Background-attachment: scroll	გამოსახულება მოძრაობს გვერდთან ერთად
Background-attachment: fixed	გამოსახულება ბლოკირებულია

თუ თქვენ style.css ფაილს ქვემოთ მოცემული სახე ექნება, მაშინ გამოსახულება დაფიქსირებული იქნება:

```
body {
background-color: #ffcc66;
background-image: url("butterfly.gif");
background-repeat: no-repeat;
background-attachment: fixed;
}
h1
 {
color: #990000; background-color: #FC9804;
 }
```

background-position თვისება საშუალებას გვაძლევს ფონური სურათი ეკრანის ნებისმიერ ადგილას განვათავსოთ. ჩუმათობის პრინციპით, ფონური გამოსახულება ეკრანის ზედა მარცხენა კუთხეში განთავსდება. ფონური გამოსახულების სასურველი ადგილმდებარეობის მისათითებლად რამდენიმე ხერხი არსებობს. ყველა ხერხი კოორდინატების მითითებას ეფუძნება. მაგალითად, ”100px 200px” ფონურ გამოსახულებას ბრაუზერის ფანჯრის მარცხენა კიდიდან 100 და ზემოდან 200 პიქსელის დაშორებით განათავსებს. კოორდინატების მითითება ასევე პროცენტებშია შესაძლებელი ან გამოიყენება სიტყვები: top, bottom, center, left და right.

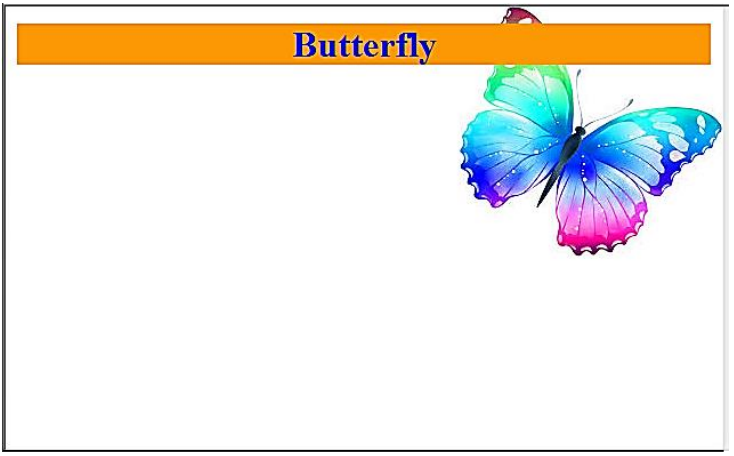


მაგალითი: ეკრანზე გამოსახულების მდებარეობის მითითება background-position თვისების საშუალებით.

```
<!DOCTYPE html>
<html>
<head>
<title>practcss1_11.html</title>
<style>
body { background-image: url("butterfly.jpg");
background-repeat: no-repeat;
background-attachment:fixed;
background-position: right top;
}
h1 {
color: blue;
background-color: #fc9804;
```

```
    }
</style>
</head>
<body>
<h1 align="center" >Butterfly</h1>
</body>
</html>
```

მიღებული შედეგი :



როგორც სურათიდან ჩანს, გამოსახულება მარჯვენა ზედა კუთხეშია მოთავსებული, რასაც ბრძანება background-position: right top უზრუნველყოფს. გამოსახულება არ მეორდება, რადგან მოცემული სურათისთვის no-repeat თვისება გვაქვს მინიჭებული.

მეტი მოხერხებულობისათვის უფრო მოკლე ჩანაწერის გაკეთებაა შესაძლებელი:

```
body {
background: url("butterfly.jpg")no-repeat fixed right top;
}
```

თვისებების შემოკლებული ჩანაწერის გამოყენებისას იგულისხმება, რომ ისინი ქვემოთ მოცემული თანამიმდევრობითაა ჩაწერილი:

- background-color
- background-image
- background-repeat
- background-attachment
- background-position.

Background-სთვის ზემოთ აღწერილ შემოკლებულ ჩანაწერში background-color თვისებაა გამოტოვებული, ამას მნიშვნელობა არა აქვს, მთავარია მათი ჩაწერის დადგენილი თანამიმდევრობა იყოს დაცული.

ტექსტის დაფორმატება CSS-ში

ტექსტის ფერი

color თვისება ტექსტის ფერის მისათითებლად გამოიყენება. ფერი CSS-ში უმეტესწილად განისაზღვრება:

- HEX (თექვსმეტობითი კოდი) მნიშვნელობით (მაგ., „#ff0000”);
- RGB მნიშვნელობით (მაგ., “rgb (255,0,0)”);
- ფერის დასახელებით (მაგ., “red”).

როგორც ზემოთ აღვნიშნეთ, RGB აღნიშნავს ფერთა სამ პალიტრას R – red (წითელი), G – green (მწვანე), B – blue (ლურჯი). მოცემული სამი ფერის შერევით (კომბინაციით) მიიღება ფერები კომპიუტერის მონიტორზე. rgb(0,0,255) ჩანაწერში თითოეული პარამეტრი ფერის ინტენსიურობას (გაჯერებულობას) აღნიშნავს და შეიძლება იყოს მთელი რიცხვი 0-255 დიაპაზონში, ასევე შეიძლება მიიღოს პროცენტული მნიშვნელობა (0%-100%).

rgb(0,0,255) ლურჯ ფერს აღნიშნავს, რადგანაც მასში ლურჯი ფერის შესაბამისი პარამეტრის მნიშვნელობა 255-ია და დანარჩენი ორი პარამეტრის მნიშვნელობა ნულის ტოლია. იგივე ფერს აღნიშნავს ჩანაწერი rgb(0%,0%,100%).

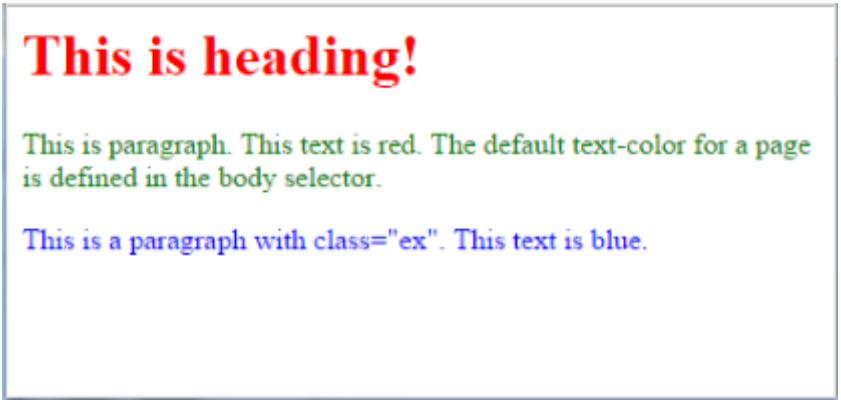
გვერდისთვის ძირითადი ფერი body სელექტორში განისაზღვრება.

მაგალითი: CSS-ში ტექსტისთვის ფერის მინიჭება.

```
<!DOCTYPE html>
<html>
<head>
<title>practcss1_12.html</title>
<style>
body
{
    color:green;
}
h1{
    color:red;
}
.ex
{
    color:blue;
}
</style>
</head>
<body>
<h1>This is heading!</h1>
<p> This is paragraph. This text is red.
The default text-color for a page is defined in the body selector.</p>
```

```
    <p class="ex"> This is a paragraph with class="ex". This text is blue.</p>
    </body>
    </html>
```

ამ კოდის შესრულების შედეგად მივიღებთ:.



გვერდისთვის ძირითადი ფერის მნიშვნელობა body სელექტორში body {color:green;} ბრძანებით განისაზღვრა.

სათაურისთვის ფერი განისაზღვრა ბრძანებით:

```
    h1 {
    color: red;
    }
```

<style> ... </style> ტეგებს შორის ex კლასი განისაზღვრა:

```
    .ex {
    color:blue;
    }
```

შესაბამისი ფერი ამ კონკრეტული p ელემენტისთვის მიენიჭა შემდეგი ხერხით:

```
<p class="ex"> This is a paragraph with class="ex". This text is blue.</p>
```

ტექსტის განლაგება გვერდის კიდეების მიმართ

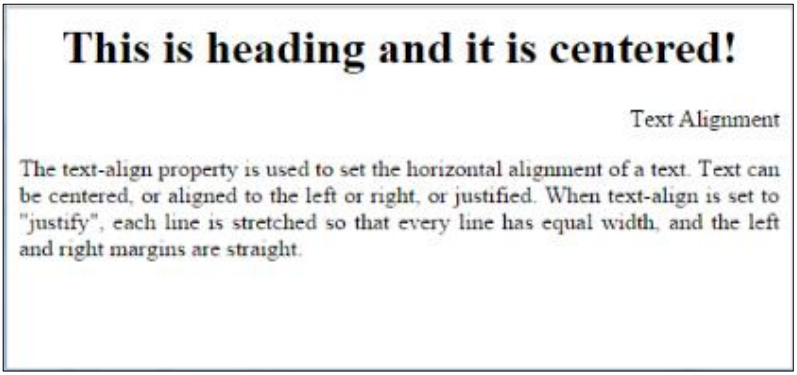
text-align თვისება ტექსტის გვერდის კიდეების გასასწორებლად გამოიყენება. ტექსტი შეიძლება მოთავსებული იყოს ცენტრში, გასწორებული იყოს მარჯვენა ან მარცხენა კიდის მიმართ, ან გასწორებული იყოს ერთდროულად ორივე – მარცხენა და მარჯვენა კიდის მიმართ.

მაგალითი: ტექსტის განლაგება გვერდის კიდეების მიმართ.

```
<!DOCTYPE html>
<html>
<head>
<title>practcss1_13.html</title>
<style>
h1 {
text-align:center;
}
p.tt {
text-align:right;
}
p.main {
text-align:justify;
}
</style>
</head>
<body>
<h1>This is heading and it is centered!</h1>
<p class="tt"> Text Alignment</p>
```

```
    <p class="main"> The text-align property is used to set the
horizontal alignment of a text. Text can be centered, or aligned to the
left or right, or justified. When text-align is set to "justify", each line is
stretched so that every line has equal width, and the left and right
margins are straight.</p>
    </body>
     </html>
```

მიღებული შედეგი:



როგორც ხედავთ, სათაური მოთავსებულია ცენტრში, ეს text-align:center ბრძანების საშუალებით მიიღწევა. p ელემენტისთვის, რომელიც მოცემულ შემთხვევაში ორჯერაა სხვადასხვა მიზნით გამოყენებული, განსაზღვრულია ორი სხვადასხვა კლასი, რომელსაც ჩვენი სურვილისამებრ დავარქვით სახელები: “tt” და “main”. tt კლასის p ელემენტისთვის მოქმედებს ბრძანება text-align:right, ხოლო main კლასის p ელემენტისთვის – text-align:justify, რომელთაგან პირველი ბრძანება უზრუნველყოფს ტექსტის გასწორებას მარჯვენა კიდის მიმართ, ხოლო მეორე – ტექსტის გასწორებას ორივე მხრიდან.

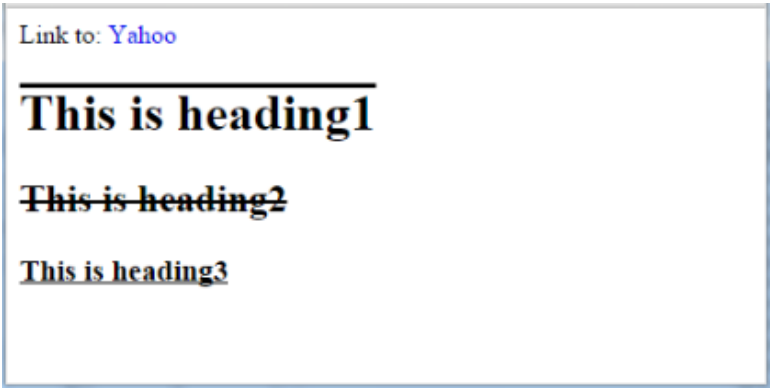
ტექსტის დეკორატიული ელემენტი

Text-decoration თვისება უმეტესწილად ბმულებისთვის ხაზგასმის მოსაცილებლად, ასევე ტექსტისთვის გარკვეული დეკორატიული ელემენტების მისანიჭებლად გამოიყენება, ეს შეიძლება ტექსტის ზემოთ, ტექსტის ქვეშ ან ტექსტის შუა ნაწილში გასმული ხაზი იყოს.

მაგალითი: text-decoration თვისების გამოყენება.

```
<!DOCTYPE html>
<html>
<head>
<title>practcss1_14.html</title>
<style>
a {text-decoration:none;}
h1 {text-decoration:overline;}
h2 {text-decoration: line-through;}
h3 {text-decoration:underline;}
</style>
</head>
<body>
<p>
Link to:
<a href="http://www.yahoo.com">Yahoo </a>
</p>
<h1> This is heading1</h1>
<h2> This is heading2</h2>
<h3> This is heading3</h3>
</body>
</html>
```

მიღებული შედეგი:



ბმულისთვის ხაზგასმის მოცილებას უზრუნველყოფს ბრძანება text-decoration: none; ტექსტის ზემოთ ხაზის გავლებას – text-decoration: overline, ტექსტის შუაში ხაზის გავლებას – text-decoration: line-through, ტექსტისთვის ქვემოთ ხაზის გასმას – text-decoration: underline.

სიმბოლოების რეგისტრის მართვა

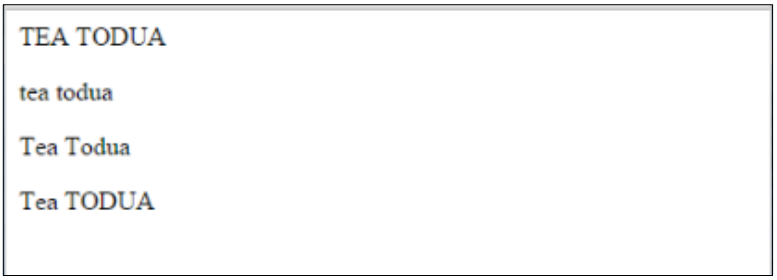
text-transform თვისება სიმბოლოების რეგისტრებს მართავს. შეიძლება ავირჩიოთ capitalize, uppercase ან lowercase. მაგალითად, სიტყვა "headline" შეიძლება ვაჩვენოთ, როგორც "HEADLINE" ან "Headline".

მაგალითი: text-transform თვისების გამოყენება.

```
<!DOCTYPE html>
<html>
<head>
<title>practcss1_15.html</title>
<style>
p.uppercase {text-transform:uppercase;}
```

```
p.lowercase {text-transform:lowercase;}
p.capitalize {text-transform:capitalize;}
p.none {text-transform:none;}
</style>
</head>
<body>
<p class="uppercase">tea todua</p>
<p class="lowercase">TEA TODUA</p>
<p class="capitalize">tea todua</p>
<p class="none"> Tea TODUA</p>
</body>
</html>
```

შედეგად მივიღებთ:



text-transform თვისებამ შეიძლება შემდეგი 4 მნიშვნელობიდან ერთ-ერთი მიიღოს:

1. capitalize საშუალებას იძლევა მთავრული ასოებით დაიწეროს სიტყვის დასაწყისი. ზემოთ მოყვანილ მაგალითში “tea todua” ეკრანზე გამოისახა როგორც “Tea Todua“.
2. uppercase უზრუნველყოფს ყველა სიმბოლოს ზედა რეგისტრში კონვერტირებას. ჩვენს მაგალითში “tea todua” ბრაუზერის ეკრანზე აისახა როგორც “TEA TODUA".

3. lowercase უზრუნველყოფს ყველა სიმბოლოს ქვედა რეგისტრში კონვერტირებას. კოდში დაწერილი “TEA TODUA" ბრაუზერის ეკრანზე გამოჩნდა როგორც “tea todua”.
4. none მნიშვნელობა გულისხმობს, რომ ტრანსფორმაცია არ ხდება. ბრაუზერში ტექსტი ისევე აისახა, როგორც ის HTML კოდში იყო მოცემული.

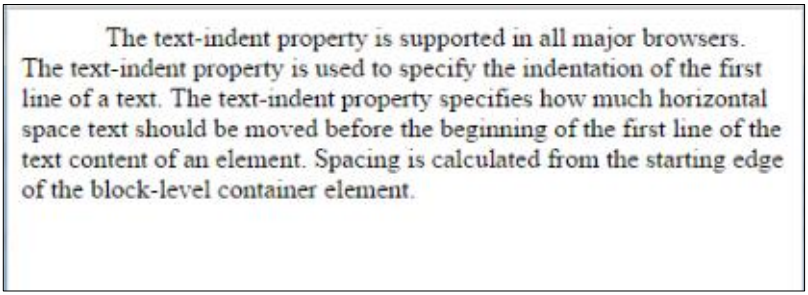
აბზაცის ზომა

text-indent თვისება იმისათვის გამოიყენება, რომ პირველი სტრიქონისათვის სააბზაცო შეწევა განსაზღვროს.

მაგალითი: text-indent თვისების გამოყენება.

```
<!DOCTYPE html>
<html>
<head>
<title>practcss1_16.html</title>
<style>
p {text-indent:50px;}
</style>
</head>
<body>
<p>The text-indent property is supported in all major browsers.The text-indent property is used to specify the  indentation of the first line of a text. The text-indent property specifies how much horizontal space text should  be moved before the beginning of the first line of the text content of an element. Spacing is calculated from the starting edge of the block-level container element.</p>
</body>
</html>
```

მივიღებთ შედეგს:



განხილულ მაგალითში პირველი სტრიქონისათვის 50 პიქსელით სააბზაცო შეწევა p {text-indent:50px;} ბრძანებამ უზრუნველყო.

text-indent თვისების მნიშვნელობა, გარდა პიქსელებისა, შეიძლება პროცენტებსა და სანტიმეტრებში განისაზღვროს, მაგალითად:

```
text-indent:50%;
text-indent:2cm;
```

სიმბოლოებსა და სიტყვებს შორის ინტერვალი

letter-spacing თვისება სიმბოლოებს შორის ინტერვალის დაყენებას უზრუნველყოფს.

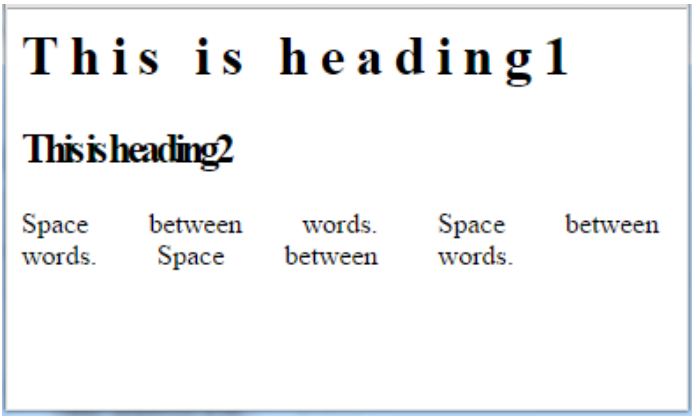
word-spacing თვისება სიტყვებს შორის ინტერვალის გაზრდის საშუალებას იძლევა.

მაგალითი: სიმბოლოებსა და სიტყვებს შორის ინტერვალის დაყენება. დავწეროთ შემდეგი კოდი:

```
<html>
<head>
<title>practcss1_17.html</title>
<style>
```

```
h1 {letter-spacing:6px;}
h2 {letter-spacing:-3px;}
p {word-spacing:30px};
</style>
</head>
<body>
<h1> This is heading1</h1>
<h2> This is heading2</h2>
<p> Space between words. Space between words. Space between words.</p>
</body>
</html>
```

კოდის შესრულების შედეგად მივიღებთ:



სიმბოლოებს შორის სასურველი ინტერვალის დაყენების საშუალებას letter:spacing თვისება იძლევა, ხოლო სიტყვებს შორის ინტერვალის გაზრდა word:spacing თვისების საშუალებითაა შესაძლებელი.

სიმბოლოებსა და სიტყვებს შორის ინტერვალის მითითება პიქსელებში ხდება.

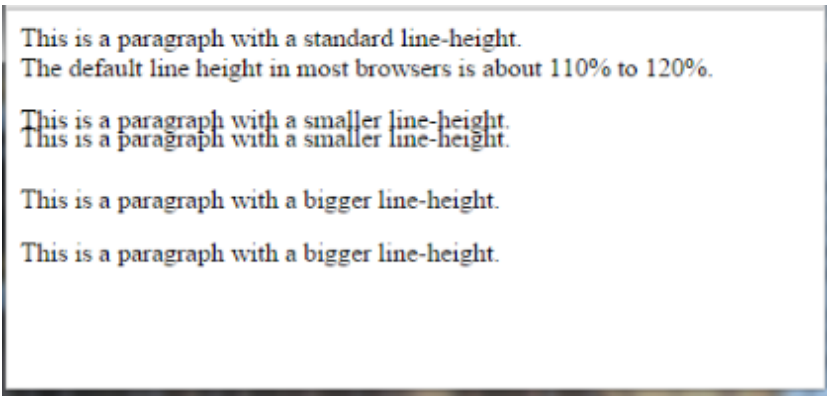
სტრიქონებს შორის მანძილი

line:height თვისება აბზაცში სტრიქონებს შორის მანძილის დაყენების საშუალებას იძლევა.

მაგალითი: სტრიქონებს შორის ინტერვალის დაყენება.

```
<!DOCTYPE html>
<html>
<head>
<title>practcss3_7.html</title>
<style>
p.small {line-height:70%;}
p.big {line-height:200%;}
</style>
</head>
<body>
<p>
This is a paragraph with a standard line-height.<br>
The default line height in most browsers is about 110% to 120%.<br>
</p>
<p class="small">
This is a paragraph with a smaller line-height.<br>
This is a paragraph with a smaller line-height.<br>
</p>
<p class="big">
This is a paragraph with a bigger line-height.<br>
This is a paragraph with a bigger line-height.<br>
</p>
</body>
</html>
```

პროგრამის შესრულების შედეგად მივიღებთ:



სტრიქონებს შორის ინტერვალის დაყენებას line:height ბრძანება უზრუნველყოფს. განხილულ მაგალითში, პირველ აბზაცში სტრიქონებს შორის მანძილი სტანდარტულია, ამიტომაც მისთვის სტილის დანიშვნა საჭირო არ არის, ბრაუზერების უმეტესობისთვის ეს სტანდარტი 110%-დან 120%-მდე მერყეობს. მეორე აბზაცში სტრიქონებს შორის მანძილად მითითებულია 70%, რაც line:height: 70% ბრძანებითაა უზრუნველყოფილი. p ელემენტისთვის კი small კლასი გვაქვს განსაზღვრული, რომლის line:height თვისების მნიშვნელობაც მოცემული აბზაცის სტრიქონებს შორის დაშორებას განსაზღვრავს. მესამე აბზაცში სტრიქონებს შორის მანძილად 200%-ია მითითებული. ამ შემთხვევაში, კონკრეტული აბზაცისთვის big კლასია განსაზღვრული, რომლის line:height თვისების მნიშვნელობაც მოცემული აბზაცის სტრიქონებს შორის დაშორებას განსაზღვრავს.

სტრიქონებს შორის დაშორება შეიძლება როგორც პროცენტებში, ასევე პიქსელებში მიეთითოს, შესაძლებელია ეს მნიშვნელობა გამოხატული იყოს რიცხვებშიც:

line-height:1;

```
line-height:4;
line:height: 25px;
```

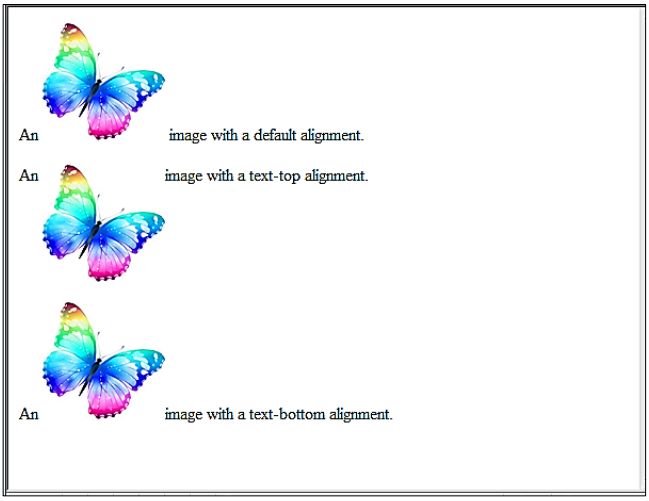
ელემენტის მიმართ ტექსტის მდებარეობის განსაზღვრა

vertical-align თვისება სხვა ელემენტის მიმართ ტექსტის ვერტიკალურ მდებარეობას განსაზღვრავს.

მაგალითი: ტექსტის ვერტიკალური მდებარეობის (საბაზისო ხაზზე, ზემოთ, ქვემოთ) მითითება გამოსახულების მიმართ.

```
<!DOCYPE html>
<head>
<title>practcss1_19.html</title>
<style>
img.top {vertical-align:text-top;}
img.buttom {vertical-align:text-buttom;}
</style>
</head>
<body>
<p> An <img src="butterfly.jpg" width="150" height="50" /> image with a default alignment. </p>
<p> An <img class="top" src="butterfly.jpg" width="150" height="50" />image with a text-top alignment.</p>
<p> An <img class="bottom" src="butterfly.jpg" width="150" height="50" />image with a text-bottom alignment.</p>
</p>
</body>
</html>
```

პროგრამის შესრულების შედეგი:



vertical-align თვისების მნიშვნელობები: top და bottom განსაზღვრავს გამოსახულების მიმართ ტექსტის განლაგებას, შესაბამისად ზემოთ და ქვემოთ. <style> … </style> ტეგებს შორის img ელემენტისთვის top და bottom კლასებია განსაზღვრული, რომლებისთვისაც vertical-align თვისების text-top და text-bottom მნიშვნელობებია მითითებული. vertical-align თვისების super და sub მნიშვნელობები ტექსტის შესაბამისად ზედა და ქვედა არეში სხვა ტექსტის მოთავსებას უზრუნველყოფს:

vertical-align:super This is text$^{\text{this is super}}$

vertical-align:sub This is text$_{\text{this is sub}}$

შრიფტები

შრიფტის დაყენება html დოკუმენტის გაფორმებისას ერთ-ერთი მნიშვნელოვანია. CSS-ს შრიფტების დაყენების და, შესაბამისად, ტექსტის დაფორმატების მძლავრი შესაძლებლობები გააჩნია.

შრიფტების ოჯახი

font-family თვისება მოცემული ელემენტის ეკრანზე ასახვის მიზნით გამოყენებულ შრიფტების პრიორიტეტულ სიაზე მიუთითებს. თუ სიაში მოცემული პირველი შრიფტი არ არის დაყენებული იმ კომპიუტერში, რომლიდანაც მოცემულ საიტზე ხდება წვდომა, მაშინ სიის მომდევნო შრიფტზე მოხდება გადასვლა.

შრიფტების კატეგორიზაციისათვის სახელების ორი ტიპი გამოიყენება: font-family და generic family.

Font-family (ხშირად ეწოდება უბრალოდ ”შრიფტი”). ეს არის, მაგალითად, "Arial", "Times New Roman“ ან "Tahoma".

Generic family შეიძლება აღიწეროს, როგორც ფონტების ჯგუფი, რომელსაც დამახასიათებელი საერთო შტრიხები აქვს. მაგალითად, sans-serif, serif, monospace.

ვებსაიტისთვის შრიფტების მითითების დროს ბუნებრივია პირველად ის შრიფტი უნდა მიუთითოთ, რომელსაც უპირატესობას ანიჭებთ, ხოლო შემდეგ – ალტერნატიული შრიფტები. სიის ბოლოს რეკომენდებულია generic family ტიპის მითითება. ასეთ შემთხვევაში, თუ ჩამოთვლილი შრიფტებიდან კომპიუტერში არც ერთი არ არის, ვებგვერდზე იმავე ოჯახის რომელიმე შრიფტი აისახება.

შრიფტების სია შეიძლება ასე გამოიყურებოდეს:

```
h1 {
font-family: arial, verdana, sans-serif;
}
h2 {
font-family: "Times New Roman", serif;
}
```

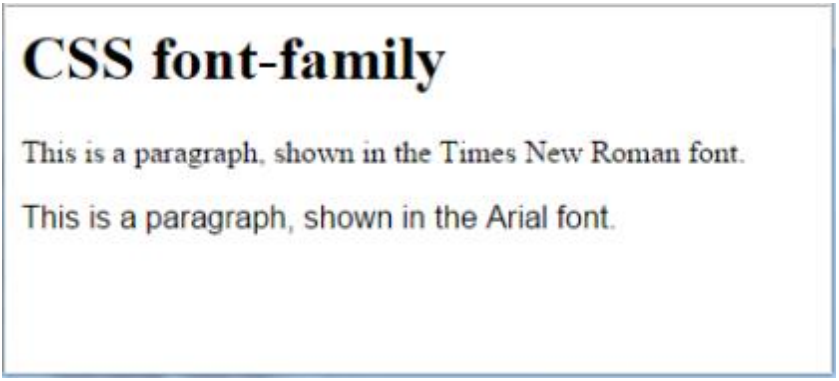
<h1> სათაურები აისახება შრიფტით "Arial», თუ ის მომხმარებლის კომპიუტერზე არ არის დაყენებული, მაშინ გამოიყენება "Verdana". თუ არც ერთი შრიფტი არ არის, მაშინ

სათაურების ეკრანზე ასახვის მიზნით sans-serif ოჯახის შრიფტი გამოიყენება. ვინაიდან "Times New Roman" შრიფტი ჰარებს (ინტერვალებს) შეიცავს, ამიტომაც ის ბრჭყალებშია ჩასმული.

მაგალითი:

```
<!DOCTYPE  html>
<html>
<head>
<style>
p.serif {
   font-family: "Times New Roman", Times, serif;
}
p.sansserif {
   font-family: Arial, Helvetica, sans-serif;
}
</style>
</head>
<body>
<h1>CSS font-family</h1>
<p class="serif">
This is a paragraph, shown in the Times New Roman font.
</p>
<p class="sansserif">
This is a paragraph, shown in the Arial font.
</p>
</body>
  </html>
```

მოცემული კოდის შესრულების შედეგი:



შრიფტის სტილი

შრიფტის სტილს (Font-Style) შეუძლია ქვემოთ მოცემული სამი მნიშვნელობიდან ერთ-ერთი მიიღოს: normal (ჩვეულებრივი ტექსტი), italic (დახრილი ტექსტი), oblique (ესეც დახრილი, ძალიან ჰგავს italic-ს, მაგრამ გამოიყენება იშვიათად);

ქვემოთ მოყვანილი მაგალითის თანახმად, ყველა <h2> სათაური დახრილი იქნება:

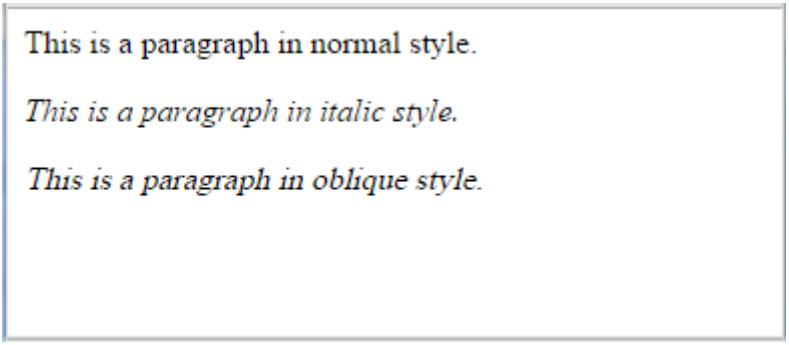
```
h2 {font-family: "Times New Roman", serif; font-style: italic;}
```

მაგალითი: Font-style თვისება

```
<!DOCTYPE  html>
<html>
<head>
<style>
p.normal {
   font-style: normal;
}
```

```
p.italic  {
   font-style: italic;
}
p.oblique {
   font-style: oblique;
}
</style>
</head>
<body>
<p class="normal">This is a paragraph in normal style.</p>
<p class="italic">This is a paragraph in italic style.</p>
<p class="oblique">This is a paragraph in oblique style.</p>
</body>
</html>
```

ამ კოდის მუშაობის შედეგი:



font-variant თვისება normal და small-caps ვარიანტებიდან ერთ-ერთის ამოსარჩევად გამოიყენება. small-caps შრიფტი დაბალი რეგისტრის სიმბოლოების ნაცვლად პატარა მთავრულ ასოებს (upper case) იყენებს.

მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<style>
p.normal {
    font-variant: normal;
}
p.small {
font-variant: small-caps;
}
</style>
</head>
<body>
<p class="normal">Georgian Technical University.</p>
<p class="small"> Georgian Technical University.</p>
</body>
</html>
```

მიღებული შედეგი:

Georgian Technical University.

GEORGIAN TECHNICAL UNIVERSITY.

შრიფტის სისქე

font-weight თვისება აღწერს, თუ რამდენად სქელი უნდა იყოს ფონტი. ფონტი შეიძლება იყოს normal ან bold. ზოგიერთ ბრაუზერს ფონტის სისქის აღსაწერად რიცხვითი მნიშვნელობის მხარდაჭერაც კი აქვს.

```
p {
font-family: arial, verdana, sans-serif; font-weight: bold;
}
```

შრიფტის ზომა

შრიფტის ზომა font-size თვისების საშუალებით განისაზღვრება. შრიფტის ზომის აღსაწერად სხვადასხვა ერთეული გამოიყენება, მაგალითად:

```
p {font-size: 30px;}
p {font-size: 12pt;}
p {font-size: 120%;}
p {font-size: 1em;}
```

ბაზური ზომები:
16px;
12pt;
100%;
1em;

CSS-ში font-ის შემოკლებული ჩაწერაც არის შესაძლებელი, მაგალითად:

```
p {
 font-style: italic;
 font-weight: bold;
 font-size: 30px;
```

```
 font-family: arial, sans-serif;
 }
```

ეს ჩანაწერი შეიძლება შემოკლებული ფორმით შემდეგნაირად ჩაიწეროს:

```
p {
 font: italic bold 30px arial, sans-serif;
}
```

ფერები CSS3-ში

CSS-ში ფერების მითითება მათი სახელების საშუალებით, თექვსმეტობითი კოდით და ფერთა RGB კოდითაა შესაძლებელი. გარდა ამისა, CSS3-ში ასევე ფერთა შემდეგი კოდების გამოყენებაა შესაძლებელი:

- ფერთა RGBA კოდი;
- ფერთა HSL კოდი;
- ფერთა HSLA კოდი.

ფერთა RGBA კოდი

ფერთა RGBA კოდი RGB კოდის გაფართოებას ალფა-პარამეტრით წარმოადგენს. ეს პარამეტრი ობიექტის გამჭვირვალობას განსაზღვრავს. ფერთა RGBA კოდის შეტანა შემდეგი სახით ხდება: rgba (წითელი, მწვანე, ლურჯი, ალფა). ალფა პარამეტრის მნიშვნელობა 0.0-დან (სრულად გამჭვირვალე) 1.0-მდე (სრულად გაუმჭვირვალე) იცვლება.

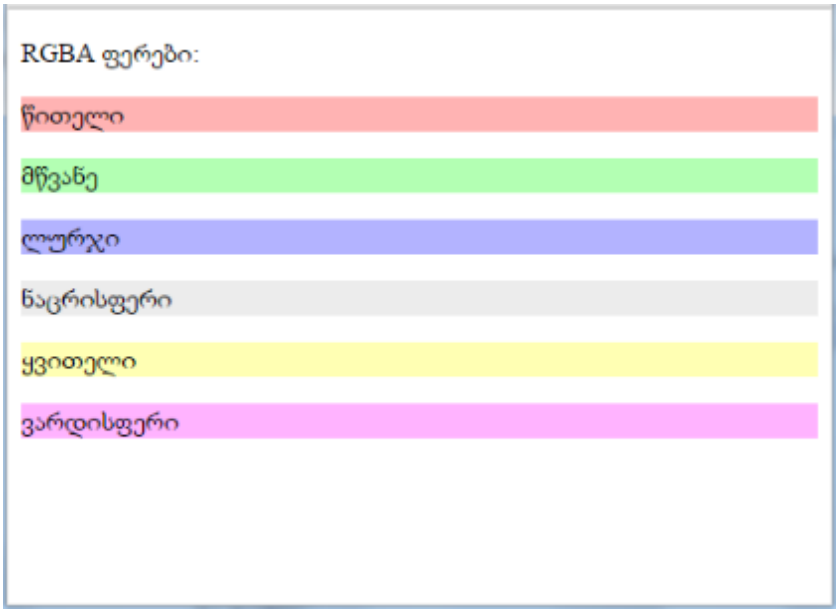
ქვემოთ მოყვანილ მაგალითში სხვადასხვა ფერი RGBA კოდშია განსაზღვრული:

```
<!DOCTYPE html>
<html>
```

```
<head>
<style>
#p1 {
background-color:rgba(255,0,0,0.3);
}
#p2 {
background-color:rgba(0,255,0,0.3);
}
#p3 {
background-color:rgba(0,0,255,0.3);
}
#p4 {
background-color:rgba(192,192,192,0.3);
}
#p5 {
background-color:rgba(255,255,0,0.3);
}
#p6 {
background-color:rgba(255,0,255,0.3);
}
</style>
</head>
<body>
<p>RGBA ფერები:</p>
<p id="p1">წითელი</p>
<p id="p2">მწვანე</p>
<p id="p3">ლურჯი</p>
<p id="p4">ნაცრისფერი</p>
<p id="p5">ყვითელი</p>
<p id="p6">ვარდისფერი</p>
```

```
</body>
</html>
```

მიღებულ შედეგს აქვს ასეთი სახე:



ფერთა HSL კოდი

ფერთა HSL კოდი შედგება ფერის ტონის, ინტენსიურობის და სიკაშკაშისგან. ფერთა HSL კოდის შეტანა შემდეგი სახით ხდება: hsl(ტონი, ინტენსიურობა, სიკაშკაშე).

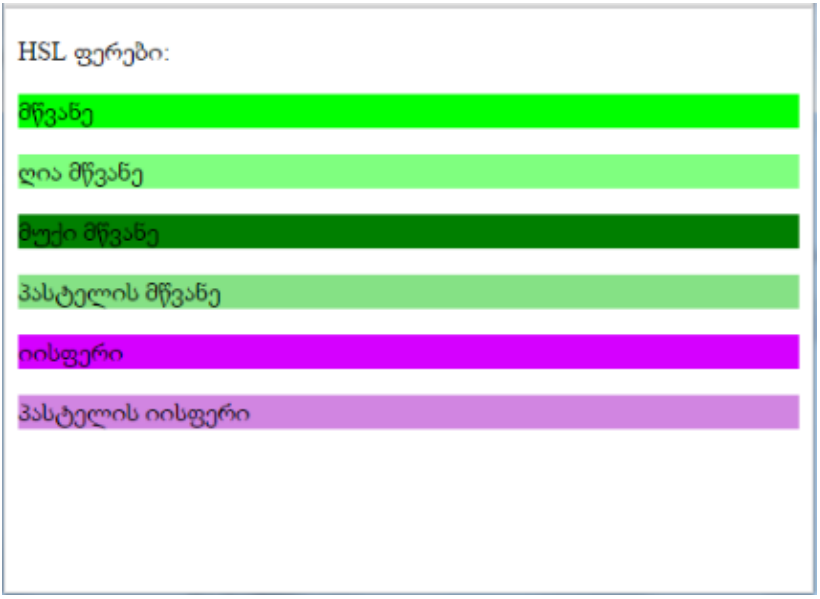
1. ფერის ტონი შეესაბამება ფერთა წრეზე მისი მდებარეობის გრადუსულ მნიშვნელობას (0-დან 360-მდე):
    - 0 (ან 360) არის წითელი
    - 120 - მწვანე
    - 240 - ლურჯი

2. ფერის ინტენსიურობა პროცენტული მნიშვნელობაა: 100% სრული ინტენსიურობის მქონე ფერს აღნიშნავს.
3. სიკაშკაშეც პროცენტული მნიშვნელობაა: 0% მუქია (შავი) და 100% ღია ფერია (თეთრი).

შემდეგ მაგალითში სხვადასხვა ფერი HSL კოდშია განსაზღვრული:

```
<!DOCTYPE html>
<html>
<head>
<style>
#p1 {background-color:hsl(120,100%,50%);}
#p2 {background-color:hsl(120,100%,75%);}
#p3 {background-color:hsl(120,100%,25%);}
#p4 {background-color:hsl(120,60%,70%);}
#p5 {background-color:hsl(290,100%,50%);}
#p6 {background-color:hsl(290,60%,70%);}
</style>
</head>
<body>
<p>HSL ფერები:</p>
<p id="p1">მწვანე</p>
<p id="p2">ღია მწვანე</p>
<p id="p3">მუქი მწვანე</p>
<p id="p4">პასტელის მწვანე</p>
<p id="p5">იისფერი</p>
<p id="p6">პასტელის იისფერი</p>
</body>
</html>
```

მიღებული შედეგი:



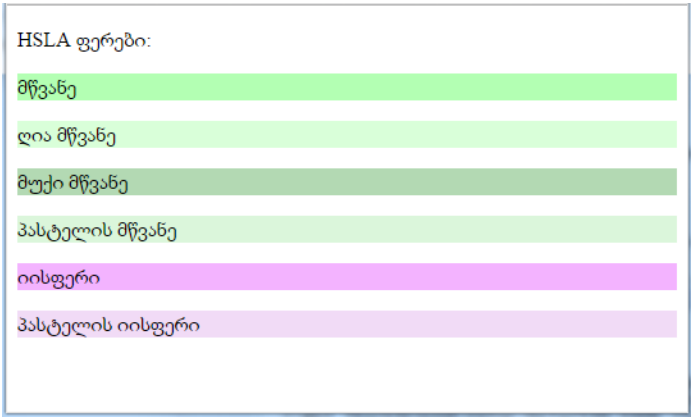
ფერთა HSLA კოდი

ფერთა HSLA კოდი HSL კოდის გაფართოებას წარმოადგენს ალფა პარამეტრით. ეს პარამეტრი ფერის გამჭვირვალობას განსაზღვრავს. ფერთა HSLA კოდის შეტანა შემდეგი სახით ხდება: hsla (ტონი, ინტენსიურობა, სიკაშკაშე, ალფა). ალფა პარამეტრის მნიშვნელობა 0.0-დან (სრულად გამჭვირვალე) 1.0-მდე (სრულად გაუმჭვირი) იცვლება.

შემდეგ მაგალითში სხვადასხვა ფერი HSLA კოდშია განსაზღვრული:

```
<!DOCTYPE html>
<html>
<head>
```

```
<style>
#p1 {background-color:hsla(120,100%,50%,0.3);}
#p2 {background-color:hsla(120,100%,75%,0.3);}
#p3 {background-color:hsla(120,100%,25%,0.3);}
#p4 {background-color:hsla(120,60%,70%,0.3);}
#p5 {background-color:hsla(290,100%,50%,0.3);}
#p6 {background-color:hsla(290,60%,70%,0.3);}
</style>
</head>
<body>
<p>HSLA ფერები:</p>
<p id="p1">მწვანე</p>
<p id="p2">ღია მწვანე</p>
<p id="p3">მუქი მწვანე</p>
<p id="p4">პასტელის მწვანე</p>
<p id="p5">იისფერი</p>
<p id="p6">პასტელის იისფერი</p>
</body>
</html>
```

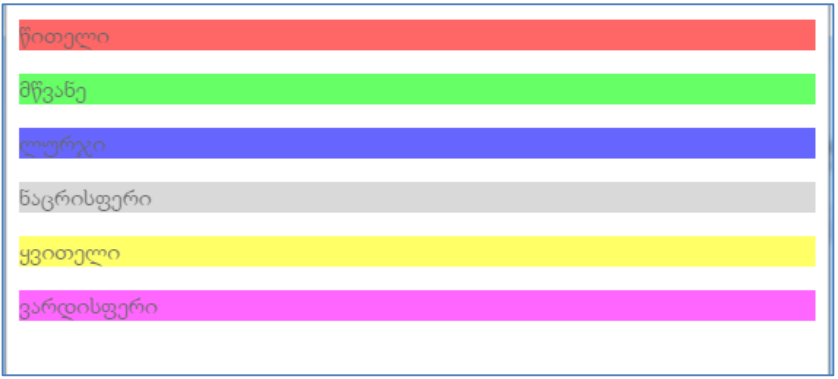


გამჭვირვალობის კოეფიციენტი

CSS3-ში გამჭვირვალობის კოეფიციენტი ფერთა მოცემული RGB კოდისათვის გამჭვირვალობის სხვადასხვა მნიშვნელობას იძლევა. გამჭვირვალობის კოეფიციენტის მნიშვნელობა მოთავსებული უნდა იყოს 0.0-სა (სრულად გამჭვირვალე) და 1.0-ს (სრულად გაუმჭვირი) შორის.

ქვემოთ მოყვანილ მაგალითში მოცემულია ფერთა RGB კოდი გამჭვირვალობის კოეფიციენტთან ერთად.

```
<!DOCTYPE html>
<html>
<head>
<style>
#p1 {background-color:rgb(255,0,0);opacity:0.6;}
#p2 {background-color:rgb(0,255,0);opacity:0.6;}
#p3 {background-color:rgb(0,0,255);opacity:0.6;}
#p4 {background-color:rgb(192,192,192);opacity:0.6;}
#p5 {background-color:rgb(255,255,0);opacity:0.6;}
#p6 {background-color:rgb(255,0,255);opacity:0.6;}
</style>
</head>
<body>
<p id="p1">წითელი</p>
<p id="p2">მწვანე</p>
<p id="p3">ლურჯი</p>
<p id="p4">ნაცრისფერი</p>
<p id="p5">ყვითელი</p>
<p id="p6">ვარდისფერი</p>
</body>
</html>
```



ჩარჩოები (Borders)

ჩარჩოებს ფართო გამოყენება აქვს, ისინი შეიძლება გამოყენებულ იქნეს, როგორც დეკორატიული ელემენტი ან ორი ობიექტის განსაცალკევების მიზნით.

ჩარჩოს ტიპები (border-style)

ჩარჩოს ტიპის მისათითებლად border-style თვისება გამოიყენება. ქვემოთ რვა სხვადასხვა ტიპის ჩარჩოა ნაჩვენები. ამ მაგალითებში "gold" ფერი და "thick" სისქეა მითითებული, თუმცა, რა თქმა უნდა, სხვა ფერისა და სისქის ჩარჩოების შექმნაცაა შესაძლებელი. თუ არ გნებავთ ჩარჩოს გამოჩენა, მაშინ border-style თვისებაში none ან hidden მნიშვნელობა უნდა მიუთითოთ.

border-style თვისება განსაზღვრავს, ჩარჩოს რომელი ტიპი უნდა მიეთითოს. მასში შეიძლება შემდეგი მნიშვნელობები იყოს მითითებული:

- dotted - განსაზღვრავს წერტილოვან ჩარჩოს;
- dashed - განსაზღვრავს წყვეტილ ჩარჩოს;
- solid - განსაზღვრავს უწყვეტ ჩარჩოს;

- double - განსაზღვრავს ორმაგ ჩარჩოს;
- groove - განსაზღვრავს 3D ჩაღრმავებული ეფექტის მქონე ჩარჩოს. ეფექტი ჩარჩოს ფერზეა დამოკიდებული;
- ridge - განსაზღვრავს 3D ამობურცული ეფექტის მქონე ჩარჩოს. ეფექტი ჩარჩოს ფერზეა დამოკიდებული;
- inset - განსაზღვრავს 3D ჩარჩოში ჩაღრმავების ეფექტს. ეფექტი ჩარჩოს ფერზეა დამოკიდებული;
- outset - განსაზღვრავს 3D ჩარჩოში ამოზნექილობის ეფექტს. ეფექტი ჩარჩოს ფერზეა დამოკიდებული;
- none - ჩარჩო არ არის;
- hidden - ჩარჩო დამალულია.

border-style თვისებას შეიძლება ოთხი არგუმენტი (ზედა ჩარჩო, მარჯვენა ჩარჩო, ქვედა ჩარჩო და მარცხენა ჩარჩო) ჰქონდეს ანუ შესაძლებელია ჩარჩოს სხვადასხვა ნაწილისთვის სხვადასხვა სტილი მიეთითოს.

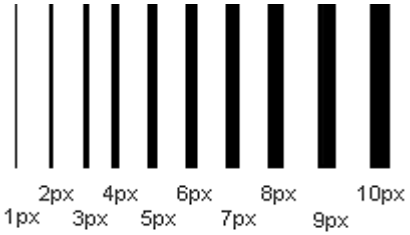
მაგალითი:

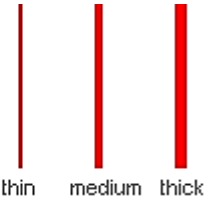
```
<!DOCTYPE html>
<html>
<head>
<style>
p {
border-color:gold;
border-width:thick;
}
p.dotted
 {
border-style: dotted;
}
p.dashed {border-style: dashed;}
```

```
p.solid {border-style: solid;}
p.double {border-style: double;}
p.groove {border-style: groove;}
p.ridge {border-style: ridge;}
p.inset {border-style: inset;}
p.outset {border-style: outset;}
p.none {border-style: none;}
p.hidden {border-style: hidden;}
p.mix {border-style: dotted dashed solid double;}
</style>
</head>
<body>
<h2>border-style თვისება</h2>
<p>მოცემული თვისება განსაზღვრავს, როგორ აისახება
ჩარჩოს გვერდები:</p>
<p class="dotted"> dotted - წერტილოვანი ჩარჩო.</p>
<p class="dashed">dashed - წყვეტილი ჩარჩო.</p>
<p class="solid">solid - უწყვეტი ჩარჩო.</p>
<p class="double">double - ორმაგი ჩარჩო.</p>
<p class="groove">groove - ჩაღრმავებული ჩარჩო.</p>
<p class="ridge"> ridge - ამობურცული ჩარჩო.</p>
<p class="inset"> inset - ჩარჩოში ჩაღრმავებული.</p>
<p class="outset"> outset - ჩარჩოში ამობურცული.</p>
<p class="none"> none - ჩარჩო არ არის.</p>
<p class="hidden"> hidden - დამალული ჩარჩო.</p>
<p class="mix"> mix -  ჩარჩო შერეული საზღვრებით.</p>
</body>
</html>
```

ჩარჩოს სისქე (border-width)

ჩარჩოს სისქე border-width თვისებით განისაზღვრება, რომელსაც შეიძლება ხაზის სისქის რიცხვითი მნიშვნელობა მივანიჭოთ (px, pt, cm - სმ, mm - მმ და სხვა) ან შეიძლება ჰქონდეს ერთ-ერთი თვისება: thin (წვრილი), medium (საშუალო) და thick (სქელი).





border-width თვისება ოთხივე მხარის ჩარჩოს სისქეს განსაზღვრავს.

border-width თვისებას შეიძლება ჰქონდეს ოთხი არგუმენტი (ზედა, მარჯვენა, ქვედა და მარცხენა ჩარჩო) ანუ შესაძლებელია მიეთითოს სხვადასხვა სისქე ჩარჩოს ზედა, ქვედა, მარჯვენა ან მარცხენა ნაწილისთვის.

ჩარჩოს ფერი (border-color)

ჩარჩოს ფერს border-color თვისება განსაზღვრავს.

CSS3-ის border-radius თვისება

CSS3-ში შეიძლება ნებისმიერი ელემენტისათვის „მომრგვალებული კუთხეების“ შესაქმნელად border-radius თვისება გამოიყენოთ. ქვემოთ მისი გამოყენების სამი მაგალითია განხილული:

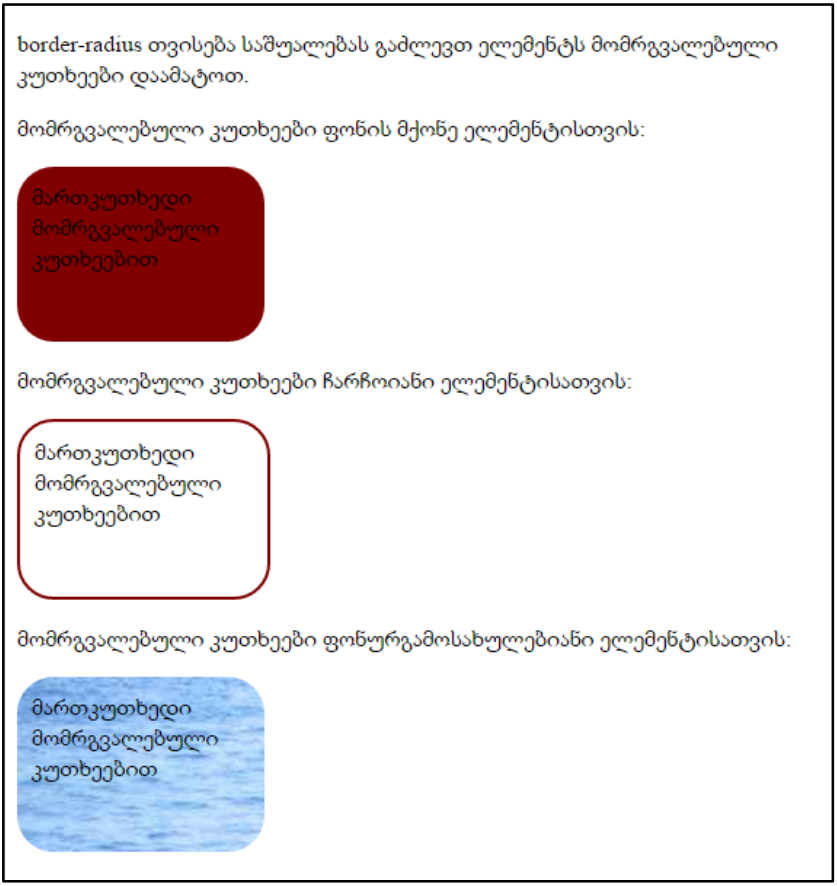
1. მომრგვალებული კუთხეები ფონის მქონე ელემენტისათვის;
2. მომრგვალებული კუთხეები ჩარჩოიანი ელემენტისათვის;
3. მომრგვალებული კუთხეები ფონური გამოსახულების მქონე ელემენტისათვის.

ასეთ კოდს შემდეგი სახე ექნება:

```
<!DOCTYPE html>
<html>
<head>
<style>
#rcorners1
{
   border-radius: 25px;
   background: #800000;
   padding: 10px;
   width: 150px;
   height: 100px;
}
#rcorners2
{
   border-radius: 25px;
   border: 2px solid #800000;
   padding: 10px;
   width: 150px;
   height: 100px;
}
#rcorners3
{
   border-radius: 25px;
   background: url(oceano.jpg);
   background-position: left top;
   background-repeat: repeat;
   padding: 10px;
   width: 150px;
   height: 100px;
}
```

```
    </style>
    </head>
    <body>
    <p>border-radius თვისება საშუალებას გაძლევთ ელემენტს
მომრგვალებული კუთხეები დაამატოთ.</p>
    <p>
    მომრგვალებული კუთხეები ფონის მქონე ელემენტისთვის:
    </p>
    <p id="rcorners1">
    მართკუთხედი მომრგვალებული კუთხეებით
    </p>
    <p>
    მომრგვალებული კუთხეები ჩარჩოიანი ელემენტისათვის:
    </p>
    <p id="rcorners2">
    მართკუთხედი მომრგვალებული კუთხეებით
    </p>
    <p>
    მომრგვალებული კუთხეები ფონურგამოსახულებიანი
ელემენტისათვის:
    </p>
    <p id="rcorners3">
    მართკუთხედი მომრგვალებული კუთხეებით
    </p>
    </body>
    </html>
```

მიღებულ შედეგს შემდეგი სახე ექნება:



უნდა აღინიშნოს, რომ border-radius თვისება, შემდეგი ოთხი თვისების - border-top-left-radius, border-top-right-radius, border-bottom-right-radius და border-bottom-left-radius-ის შემოკლებული ჩანაწერია.

CSS3-ის border-radius თვისება ფიგურის ყოველ კუთხეს განსაზღვრავს. თუ border-radius თვისებაში მხოლოდ ერთ მნიშვნელობას მივუთითებთ, მაშინ ეს რადიუსი ფიგურის ოთხივე კუთხისთვის გამოიყენება. მიუხედავად ამისა, მომხმარებელს თუ

სურვილი აქვს, შეუძლია თითოეული კუთხისთვის ცალკ-ცალკე მიუთითოს რადიუსი შემდეგი წესის გამოყენებით:

1. თუ მითითებულია ოთხივე მნიშვნელობა: პირველი მნიშვნელობა მიეკუთვნება ზედა მარცხენა კუთხეს, მეორე - ზედა მარჯვენა კუთხეს, მესამე - ქვედა მარჯვენა კუთხეს და მეოთხე - ქვედა მარცხენა კუთხეს.
2. თუ მითითებულია სამი მნიშვნელობა: პირველი მნიშვნელობა მიეკუთვნება ზედა მარცხენა კუთხეს, მეორე - ზედა მარჯვენა კუთხეს და ქვედა მარცხენა კუთხეს, მესამე - ქვედა მარჯვენა კუთხეს.
3. თუ მითითებულია ორი მნიშვნელობა: პირველი მნიშვნელობა მიეკუთვნება ზედა მარცხენა კუთხეს და ქვედა მარჯვენა კუთხეს, ხოლო მეორე - ზედა მარჯვენა კუთხეს და ქვედა მარცხენა კუთხეს.
4. ერთი მნიშვნელობა მიუთითებს, რომ ოთხივე კუთხე თანაბრად იქნება მომრგვალებული.

შემდეგ მაგალითში სამი სხვადასხვა შემთხვევა გვაქვს განხილული:

1. ოთხი მნიშვნელობა - border-radius: 15px 50px 30px 5px;
2. სამი მნიშვნელობა - border-radius: 15px 50px 30px;
3. ორი მნიშვნელობა - border-radius: 15px 50px.

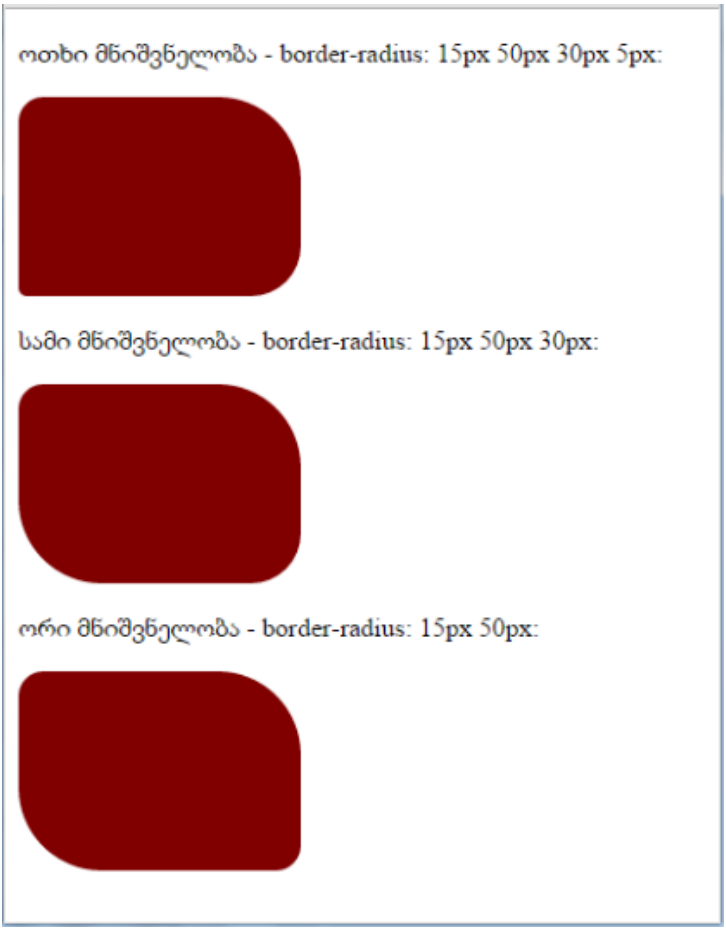
მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<style>
#rcorners4
{
   border-radius: 15px 50px 30px 5px;
   background: #800000;
```

```
    padding: 10px;
    width: 150px;
    height: 100px;
}
#rcorners5
 {
    border-radius: 15px 50px 30px;
    background: #800000;
    padding: 10px;
    width: 150px;
    height: 100px;
}
#rcorners6
 {
    border-radius: 15px 50px;
    background: #800000;
    padding: 10px;
    width: 150px;
    height: 100px;
}
</style>
</head>
<body>
<p>
ოთხი მნიშვნელობა - border-radius: 15px 50px 30px 5px:
</p>
<p id="rcorners4"></p>
<p>
სამი მნიშვნელობა - border-radius: 15px 50px 30px:
</p>
```

```
<p id="rcorners5"></p>
<p>ორი მნიშვნელობა - border-radius: 15px 50px:</p>
<p id="rcorners6"></p>
</body>
</html>
```

შედეგად მივიღებთ:

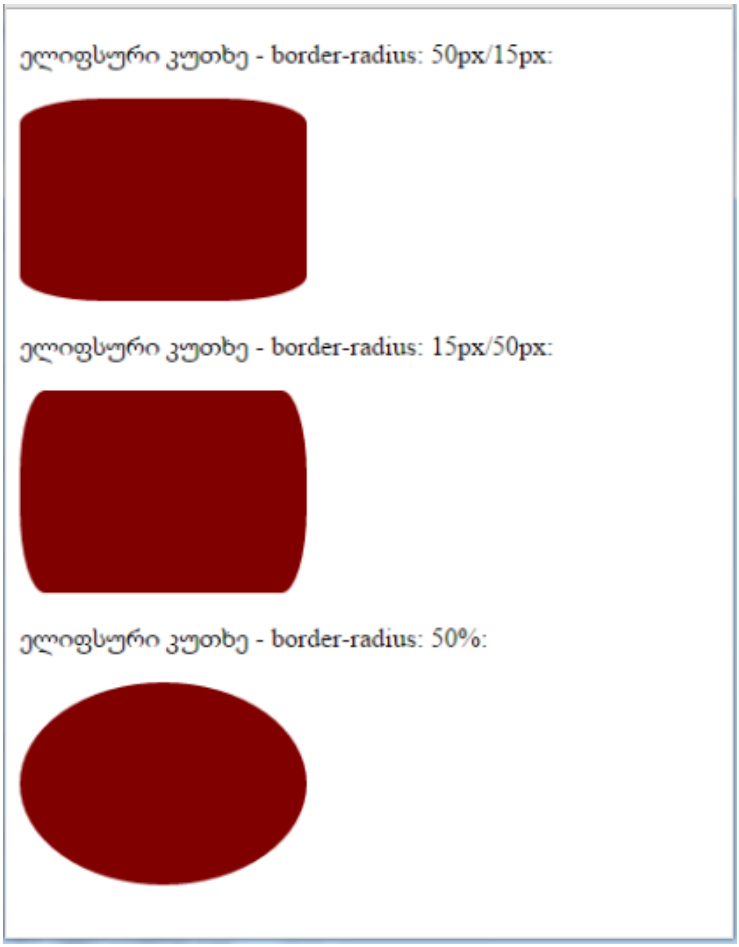


ასევე შეგიძლიათ შექმნათ ელიფსური კუთხეები:

```
<!DOCTYPE html>
<html>
<head>
<style>
#rcorners7 {
  border-radius: 50px/15px;
  background: #800000;
  padding: 10px;
  width: 150px;
  height: 100px; }
#rcorners8 {
  border-radius: 15px/50px;
  background: #800000;
  padding: 10px;
  width: 150px;
  height: 100px; }
#rcorners9 {
  border-radius: 50%;
  background: #800000;
  padding: 10px;
  width: 150px;
  height: 100px; }
</style>
</head>
<body>
<p>ელიფსური კუთხე - border-radius: 50px/15px:</p>
<p id="rcorners7"></p>
<p>ელიფსური კუთხე - border-radius: 15px/50px:</p>
```

```
<p id="rcorners8"></p>
<p>ელიფსური კუთხე - border-radius: 50%:</p>
<p id="rcorners9"></p>
</body>
</html>
```

მიღებულ შედეგს შემდეგი სახე ექნება:



მსგავსად სხვა CSS თვისებებისა, შემოკლებული ჩანაწერი შესაძლებელია border თვისებისთვისაც გვქონდეს. ვთქვათ, გვაქვს ჩანაწერი

```
p
{
border-width: 1px;
border-style: solid;
 border-color: blue;
}
```

იგი შემოკლებით ასე შეიძლება ჩაიწეროს:

```
p
{
 border: 1px solid blue;
 }
```

ქვემოთ მოყვანილია ჩარჩოების გამოყენების კიდევ ერთი პრაქტიკული მაგალითი, რისთვისაც შევქმნათ practcss1_20.html და style.css დოკუმენტები:

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=utf-8">
<title> practcss1_20.html</title>
<link href="style.css" rel="stylesheet" type="text/css">
</head>
<body>
<h2> CSS თვისებები</h2>
  <table class="table1">
  <tr>
```

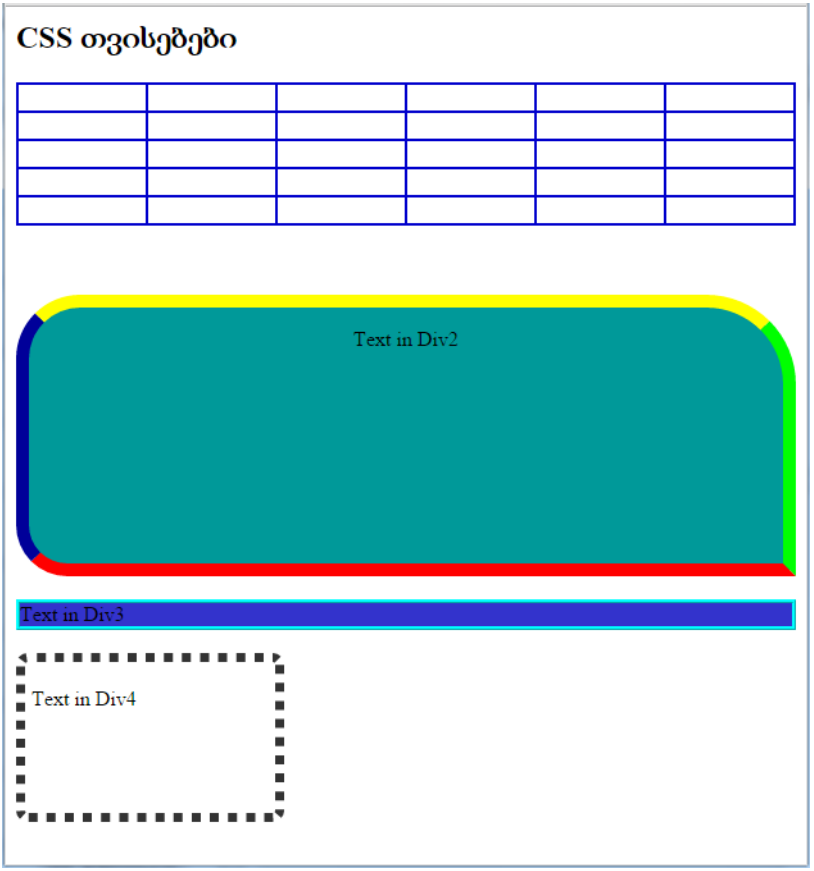
```
    <td> </td>
    <td> </td>
    <td> </td>
    <td> </td>
    <td> </td>
    <td> </td>
  </tr>
  <tr>
    <td> </td>
    <td> </td>
    <td> </td>
    <td> </td>
    <td> </td>
    <td> </td>
  </tr>
  <tr>
    <td> </td>
    <td> </td>
    <td> </td>
    <td> </td>
    <td> </td>
    <td> </td>
  </tr>
  <tr>
    <td> </td>
    <td> </td>
    <td> </td>
    <td> </td>
    <td> </td>
    <td> </td>
```

```
  </tr>
  <tr>
    <td> </td>
    <td> </td>
    <td> </td>
    <td> </td>
    <td> </td>
    <td> </td>
  </tr>
</table>
     <br/>
    <br/>
     <div class="div1"> </div>
    <br/>
    <div class="div2">
     <p align="center"> Text in Div2 </p>
    </div>
    <br/>
    <div class="div3">
     Text in Div3
    </div>
    <br/>
    <div class="div4">
     Text in Div4
    </div>
    <br/>
    </body>
</html>
```

style.css დოკუმენტს აქვს სახე:

```
.div1       {       width:300px;
                    height:200px;
                    background-color:#999;
                    border-bottom-style:solid;
                    border-bottom-color:#F00;
                    border-bottom-width:3px;}
.div2       {       border-bottom:solid #F00 10px;
                    border-left:solid #009 10px;
                    border-right:solid #0F0 10px;
                    border-top:solid #FF0 10px;
                    background-color:#099;
                    border-bottom-left-radius:40px;
                    border-top-left-radius:50px;
                    border-top-right-radius:70px;
                    height:200px;}
.div3       {       background-color:#33C;
                    border-style:ridge;
                    border-color:#0FF;
                    border-width:3px;}
.div4       {       border:dotted #333 7px;
                    border-radius:9px;
                    width:120px;
                    padding:20px 70px 80px 5px;}
.table1     {       width:100%;
                    border-spacing:0px;
                    border-collapse:collapse;}
.table1 td  { border:2px solid #00C;}
```

მიღებულ შედეგს შემდეგი სახე ექნება:



თუ ბრაუზერის ეკრანზე მიღებულ შედეგს შევხედავთ, დავინახავთ ტექსტს - „CSS თვისებები”. ეს ტექსტი მოცემული დოკუმენტისთვის სათაურია და <h2> ტეგის საშუალებითაა შექმნილი.

შემდეგი ელემენტი 6 სვეტისა და 5 სტრიქონისაგან შემდგარი ცხრილია, რომლისთვისაც CSS კოდში შემდეგი თვისებები გვაქვს გაწერილი:

```
.table1      {      width:100%;
                    border-spacing:0px;
                    border-collapse:collapse;}
.table1 td   { border:2px solid #00C;}
```

ეს ნიშნავს, რომ ცხრილის სიგანე ბრაუზერის მთელ ეკრანს (width:100%;) მოიცავს, ჩარჩოებს შორის დაშორება არის 0 პიქსელი (border-spacing:0px;). ჩარჩოს სისქე 2 პიქსელია (border:2px), ფერი კი - ლურჯი; ვინაიდან, ჩვენს მაგალითში border:collapse:collapse თვისება გვაქვს გამოყენებული, იმის მიუხედავად, რომ ცხრილის ჩარჩოს სისქე 2 პიქსელია და ჩარჩოებს შორის დაშორება (border-spacing) 0 პიქსელი, ცხრილის შიგა ჩარჩოს სისქე 4 პიქსელი (ანუ არ შეერთდება ორი ჩარჩოს სისქე) ვერ გახდება. იმ შემთხვევაში, თუ ჩვენ border:collapse თვისებას არ მივუთითებთ და ჩარჩოებს შორის დაშორება (border-spacing) ნულის ტოლი იქნება, ცხრილის შიგა ჩარჩოების სისქე, მათი ერთმანეთთან შერწყმის გამო (ვინაიდან მათ შორის მანძილი ნულის ტოლია), გაორმაგდება. იმისათვის, რომ მსგავსი სიტუაცია თავიდან ავიცილოთ, ვწერთ: border-collapse: collapse.

მთელი დოკუმენტი div ელემენტის საშუალებით ბლოკებადაა დაყოფილი.

პირველ ბლოკს, პირობითად დავარქვით სახელი div1. მისი CSS თვისებები შემდეგნაირად გამოიყურება:

```
.div1        {      width:300px;
                    height:200px;
                    background-color:#999;
                    border-bottom-style:solid;
                    border-bottom-color:#F00;
                    border-bottom-width:3px;}
```

გასაგებია, რომ ამ ბლოკის სიგანე ბრაუზერის ეკრანზე 300 პიქსელს, ხოლო სისქე - 200 პიქსელს მოიცავს. მოცემული ბლოკისთვის განსაზღვრულია ფონის ფერი და ქვედა ჩარჩოს ხაზის სტილი, ფერი და სისქე. ხაზის სტილი არის მთლიანი (არაწყვეტილი), ფერი - ლურჯი და სისქე კი 3 პიქსელის ტოლია.

შემდეგი ელემენტი ბრაუზერის ეკრანზე არის div2 ბლოკი. მისი CSS თვისებები შემდეგნაირად გამოიყურება:

```
.div2          { border-bottom:solid #F00 10px;
                 border-left:solid #009 10px;
                 border-right:solid #0F0 10px;
                 border-top:solid #FF0 10px;
                 background-color:#099;
                 border-bottom-left-radius:40px;
                 border-top-left-radius:50px;
                 border-top-right-radius:70px;
                 height:200px;}
```

ელემენტის ქვედა, ზედა, მარჯვენა და მარცხენა ჩარჩოები 10 პიქსელის სისქისაა, ხაზის სტილი არის solid, ხოლო ჩარჩოს ფერები, შესაბამისად, წითელი, ყვითელი, მწვანე და ლურჯია. მითითებულია ასევე ფონის ფერი და ჩარჩოს მარცხენა ქვედა, მარცხენა ზედა და მარჯვენა ზედა მხრისთვის კუთხეების მომრგვალების ზომები (border-bottom-left-radius:40px; border-top-left-radius:50px; border-top-right-radius:70px;). რაც მეტია პიქსელების მნიშვნელობა, მით მეტადაა მომრგვალებული კუთხე.

div3 ბლოკისთვის გაწერილია შემდეგი CSS თვისებები:

```
.div3          { background-color:#33C;
                 border-style:ridge;
                 border-color:#0FF;
```

```
                    border-width:3px;}
```

მითითებულია ფონის ფერი, ჩარჩოს სტილი - ამობურცული ეფექტით, ჩარჩოს ფერი და ჩარჩო სისქე - 3 პიქსელი.

div4 ბლოკისთვის კი შემდეგი სტილები გვაქვს:

```
.div4    {    border:dotted #333 7px;
              border-radius:9px;
              width:120px;
              padding:20px 70px 80px 5px;}
```

ჩარჩო წყვეტილია, კუთხეები 9 პიქსელითაა მომრგვალებული, ჩარჩოს სიგანე 120 პიქსელია და ტექსტის დაშორება ჩარჩოს ზედა, მარჯვენა, ქვედა და მარცხენა კიდეებიდან, შესაბამისად, 20, 70, 80 და 5 პიქსელი (padding:20px 70px 80px 5px;).

ფერთა გადასვლა (გრადიენტი) CSS3-ში

CSS3-ში ფერთა გადასვლა ანუ გრადიენტი ორ ან მეტ ფერს შორის რბილად გადასვლას უზრუნველყოფს. ადრე ასეთი ეფექტის განსახორციელებლად გამოსახულების გამოყენება იყო საჭირო. ამჟამად, CSS3-ის გრადიენტების გამოყენებით მათი ჩატვირთვის დრო მცირეა და გამოსახულებაც მისი გადიდების შემთხვევაში უფრო ხარისხიანია, ვინაიდან გრადიენტის გენერირება ბრაუზერში ხდება.

CSS3-ში ორი ტიპის გრადიენტი განიხილება:

- ხაზოვანი გრადიენტი (მიმართულია ქვემოთ| ზემოთ| მარცხნივ| მარჯვნივ| დიაგონალურად);
- რადიალური გრადიენტი (განისაზღვრება ცენტრის მიმართ).

ხაზოვანი გრადიენტი

ხაზოვანი გრადიენტის შესაქმნელად მინიმუმ ორი ფერი უნდა განისაზღვროს. ეს ორი ფერი არის ის ფერები, რომელთა შორისაც გადასვლა უნდა მოხდეს. ასევე გრადიენტის ეფექტთან ერთად უნდა განისაზღვროს გრადიენტის საწყისი წერტილი, მიმართულება (ან კუთხე).

ხაზოვანი გრადიენტის მაგალითი:

მისი სინტაქსია:

```
background: linear-gradient(მიმართულება, ფერი1, ფერი2, ...);
```

ქვემოთ მოყვანილია ხაზოვანი გრადიენტის მაგალითი, სადაც ფერები ზემოდან ქვემოთ იცვლება.

```
<!DOCTYPE html>
<html>
<head>
<style>
#grad1 {   height: 200px;
  background: -o-linear-gradient(red, green); /* Opera 11.1-დან 12.0-მდე */
  background: -moz-linear-gradient(red, green); /* Firefox 3.6-დან 15-მდე */
```

```
        background: linear-gradient(red, green); /* სტანდარტული
სინტაქსი */      }
    </style>
    </head>
    <body>
    <h3>ხაზოვანი გრადიენტი -ზემოდან ქვემოთ</h3>
    <p>მოცემული ხაზოვანი გრადიენტი იწყება ზემოდან. იგი
იწყება წითელი ფერით და თანდათან გადადის მწვანეში:</p>
    <div id="grad1"></div>
    </body>
    </html>
```

ანალოგიურად, თუ ზემოთ მოყვანილ პროგრამაში სტილს შემდეგნაირად ჩავწერთ:

```
    #grad1 {    height: 200px;
        background: -o-linear-gradient(right, red, green); /* Opera 11.1-
დან 12.0-მდე */
        background: -moz-linear-gradient(right, red, green); /* Firefox
3.6-დან 15-მდე */
```

```
        background: linear-gradient(to right, red , green); /*
სტანდარტული სინტაქსი */    }
```

მაშინ მივიღებთ ხაზოვან გრადიენტს მარცხნიდან მარჯვნივ, ხოლო თუ შემდეგ ჩანაწერს გავაკეთებთ

```
    #grad1 {    height: 200px;
        background: -o-linear-gradient(bottom right, red, green); /*
Opera 11.1-დან 12.0-მდე */
        background: -moz-linear-gradient(bottom right, red, green); /*
Firefox 3.6-დან 15-მდე */
        background: linear-gradient(to bottom right, red , green); /*
სტანდარტული სინტაქსი */    }
```

მაშინ მივიღებთ დიაგონალურ ხაზოვან გრადიენტს ზედა მარცხენა კუთხიდან ქვედა მარჯვენა კუთხისაკენ.

გრადიენტის მიმართულების კონტროლი კუთხეების გამოყენებით

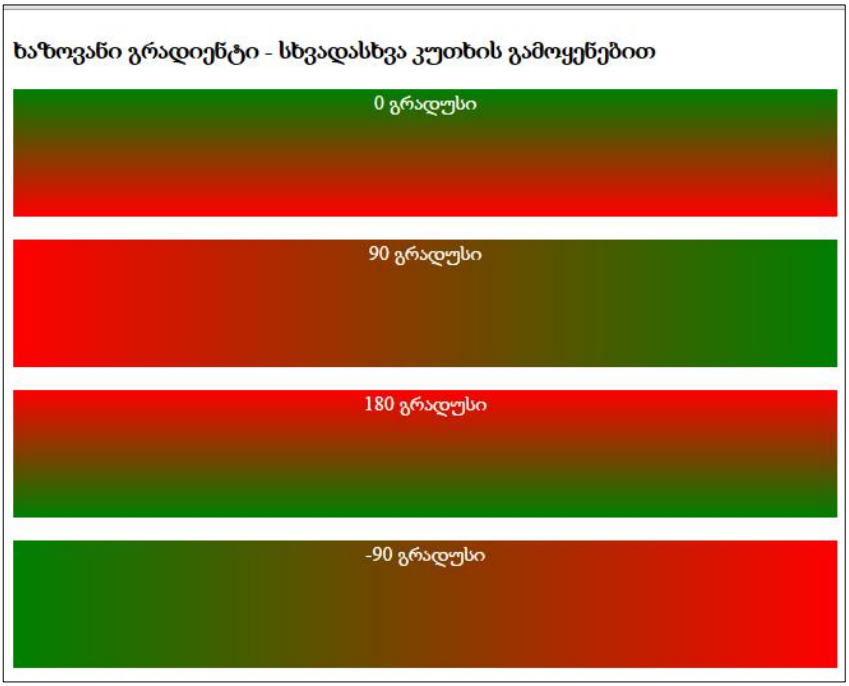
თუ გსურთ გრადიენტის მიმართულების კონტროლი, მაშინ წინასწარ მოცემული მიმართულების ნაცვლად (ქვემოთ, ზემოთ, მარცხნივ, მარჯვნივ და ა. შ.) უნდა განსაზღვროთ კუთხე. მისი სინტაქსი იქნება:

```
    background: linear-gradient (კუთხე, ფერი1, ფერი2);
```

ამ შემთხვევაში კუთხე უნდა ჩავწეროთ, როგორც კუთხე ჰორიზონტალურ ხაზსა და გრადიენტს შორის საათის ისრის საპირისპირო მიმართულებით. სხვა სიტყვებით რომ ვთქვათ, 0^0 ქმნის ქვემოდან ზემოთ გრადიენტს, 90^0 მარცხნიდან მარჯვნივ და ა. შ. მომდევნო მაგალითში ხაზოვან გრადიენტებში კუთხეების გამოყენებაა ნაჩვენები:

```
<!DOCTYPE html>
<html>
<head>
<style>
#grad1 {
    height: 100px;
    background: -o-linear-gradient(0deg, red, green); /* Opera 11.1-
დან 12.0-მდე */
    background: -moz-linear-gradient(0deg, red, green); /* Firefox
3.6-დან 15-მდე */
    background: linear-gradient(0deg, red, green); /*
სტანდარტული სინტაქსი */    }
#grad2 {
 height: 100px;
 background: -o-linear-gradient(90deg, red, green); /* Opera 11.1-
დან 12.0-მდე */
    background: -moz-linear-gradient(90deg, red, green); /* Firefox
3.6-დან 15-მდე */
    background: linear-gradient(90deg, red, green); /*
სტანდარტული სინტაქსი */    }
#grad3 {
    height: 100px;
    background: -o-linear-gradient(180deg, red, green); /* Opera
11.1-დან 12.0-მდე */
    background: -moz-linear-gradient(180deg, red, green); /* Firefox
3.6-დან 15-მდე */
    background: linear-gradient(180deg, red, green); /*
სტანდარტული სინტაქსი */    }
#grad4 {
height: 100px;
```

```
        background: -o-linear-gradient(-90deg, red, green); /* Opera
11.1-დან 12.0-მდე */
        background: -moz-linear-gradient(-90deg, red, green); /* Firefox
3.6-დან 15-მდე */
        background: linear-gradient(-90deg, red, green); /*
სტანდარტული სინტაქსი */    }
    </style>
    </head>
    <body>
    <h3>ხაზოვანი გრადიენტი - სხვადასხვა კუთხის გამოყენებით
    </h3>
    <div id="grad1" style="color:white;text-align:center;">
    0 გრადუსი
    </div>
    <br>
    <div id="grad2" style="color:white;text-align:center;">
    90 გრადუსი
    </div><br>
    <div id="grad3" style="color:white;text-align:center;">
    180 გრადუსი
    </div>
    <br>
    <div id="grad4" style="color:white;text-align:center;">
    -90 გრადუსი
    </div>
    </body>
    </html>
```



ქვემოთ მოყვანილია რამდენიმე ფერის (ცისარტყელას ფერების) გამოყენებით ხაზოვანი გრადიენტის მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<style>
#grad1 {    height: 55px;
    background: -o-linear-gradient(left, red, orange, yellow, green, blue, indigo, violet); /* Opera 11.1-დან 12.0-მდე */
    background: -moz-linear-gradient(left, red, orange, yellow, green, blue, indigo, violet); /* Firefox 3.6-დან 15-მდე */
    background: linear-gradient(to right, red, orange, yellow, green, blue, indigo, violet); /* სტანდარტული სინტაქსი */       }
```

```
    </style>
    </head>
    <body>
    <div id="grad1" style="text-align:center;margin:auto;
color:#888888;font-size:40px;font-weight:bold">
    გრადიენტის ფონი
    </div>
    </body>
    </html>
```

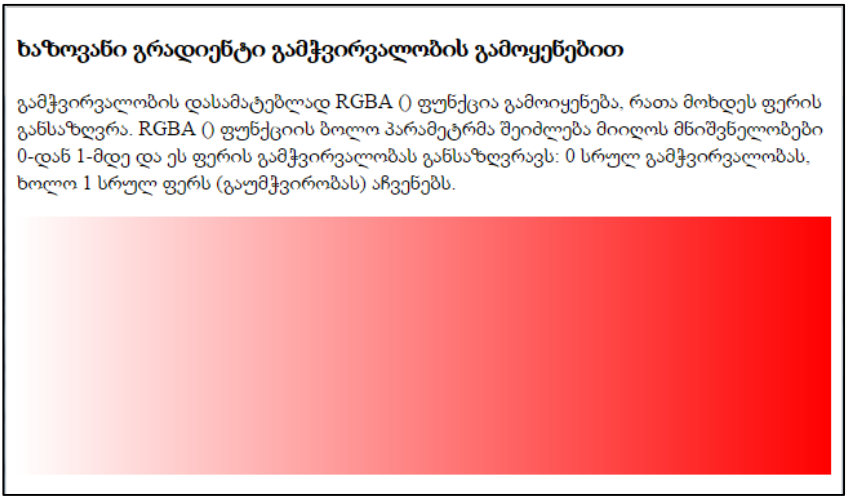
გამჭვირვალობის გამოყენება გრადიენტებში

გრადიენტში ასევე შეიძლება გამოყენებულ იქნეს გამჭვირვალობაც. გამჭვირვალობის დასამატებლად RGBA () ფუნქცია გამოიყენება. RGBA() ფუნქციის ბოლო პარამეტრმა შეიძლება მიიღოს მნიშვნელობები 0-დან 1-მდე და ეს ფერის გამჭვირვალობას განსაზღვრავს: 0 - სრულად გამჭვირვალე, ხოლო 1 - სრულად გაუმჭვირ ფერს გულისხმობს. შემდეგ მაგალითში ხაზოვანი გრადიენტია მოცემული, რომელიც მარცხენა მხრიდან იწყება და რომელშიც სრულად გამჭვირვალე ფერი თანდათან გაუმჭვირ წითელ ფერში გადადის:

```
    <!DOCTYPE html>
    <html>
    <head>
    <title>
    gradients
    </title>
```

```
<style>
#grad1 {
  height: 200px;
  background: -o-linear-gradient(right, rgba(255,0,0,0),
rgba(255,0,0,1)); /* Opera 11.1-დან 12.0-მდე */
  background: -moz-linear-gradient(right, rgba(255,0,0,0),
rgba(255,0,0,1)); /* Firefox 3.6-დან 15-მდე */
  background: linear-gradient(to right, rgba(255,0,0,0),
rgba(255,0,0,1)); /* სტანდარტული სინტაქსი */
}
</style>
</head>
<body>
<h3>
ხაზოვანი გრადიენტი გამჭვირვალობის გამოყენებით
</h3>
<p>
გამჭვირვალობის დასამატებლად RGBA () ფუნქცია
გამოიყენება, რათა მოხდეს ფერის განსაზღვრა.
RGBA () ფუნქციის ბოლო პარამეტრმა შეიძლება მიიღოს
მნიშვნელობები 0-დან 1-მდე და ეს ფერის გამჭვირვალობას
განსაზღვრავს: 0 სრულ გამჭვირვალობას, ხოლო 1 სრულ ფერს
(გაუმჭვირობას) აჩვენებს.
</p>
<div id="grad1"></div>
</body>
</html>
```

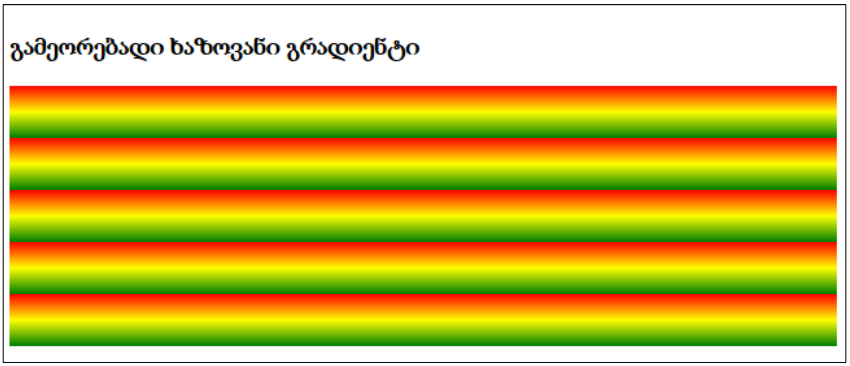
მიღებული შედეგი:



ხაზოვანი გრადიენტების გამეორებისათვის repeating-linear-gradient() ფუნქცია გამოიყენება. მაგალითად:

```
<!DOCTYPE html>
<html>
<head>
<style>
#grad1 {    height: 200px;
    background: -o-repeating-linear-gradient(red, yellow 10%, green 20%); /* Opera 11.1-დან 12.0-მდე */
    background: -moz-repeating-linear-gradient(red, yellow 10%, green 20%); /* Firefox 3.6-დან 15-მდე */
    background: repeating-linear-gradient(red, yellow 10%, green 20%); /* სტანდარტული სინტაქსი */    }
</style>
</head>
<body>
```

```
<h3>გამეორებადი ხაზოვანი გრადიენტი</h3>
<div id="grad1"></div>
</body>
</html>
```



რადიალური გრადიენტი ფერების თანაბარი განაწილებით

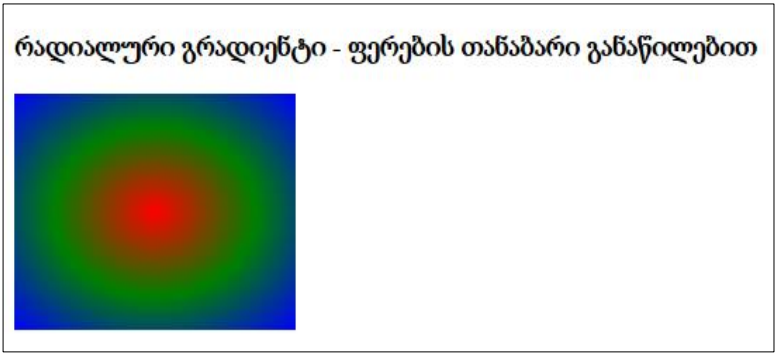
რადიალური გრადიენტის დროს პირველ რიგში მისი ცენტრი განისაზღვრება. აგრეთვე, მისი შექმნისთვის მინიმუმ ორი ფერია საჭირო.
მისი სინტაქსია:

background: radial-gradient(*ფორმა, სასტარტო ფერი, ..., ბოლო ფერი*);

თუ ფორმა არ არის მითითებული, მაშინ გრადიენტს ელიფსის ფორმა ექნება. თუ გვინდა, რომ მას წრის ფორმა ჰქონდეს, მაშინ circle-ს ვუთითებთ.

```
<!DOCTYPE html>
<html>
<head>
<style>
```

```
    #grad1 {
      height: 150px;
      width: 200px;
      background: -o-radial-gradient(red, green, blue); /* Opera 11.1-
დან 12.0-მდე */
      background: -moz-radial-gradient(red, green, blue); /* Firefox
3.6-დან 15-მდე */
      background: radial-gradient(red, green, blue); /* სტანდარტული
სინტაქსი */
    }
    </style>
    </head>
    <body>
    <h3>
    რადიალური გრადიენტი - ფერების თანაბარი განაწილებით
    </h3>
    <div id="grad1"></div>
    </body>
    </html>
```



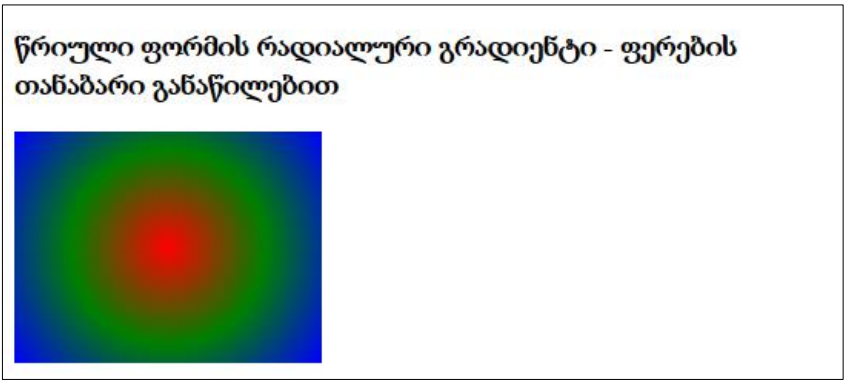
რადიალური გრადიენტი ფერების არათანაბარი განაწილებით

მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<style>
#grad1 {
  height: 150px;
  width: 200px;
  background: -o-radial-gradient(red 5%, green 15%, blue 60%);
/* Opera 11.1-დან 12.0-მდე */
  background: -moz-radial-gradient(red 5%, green 15%, blue 60%);
/* Firefox 3.6-დან 15-მდე */
  background: radial-gradient(red 5%, green 15%, blue 60%);
/* სტანდარტული სინტაქსი */ }
</style>
</head>
<body>
<h3>
რადიალური გრადიენტი - ფერების არათანაბარი
განაწილებით
</h3>
<div id="grad1"></div>
</body>
</html>
```

ქვემოთ მოცემულია წრიული ფორმის რადიალური გრადიენტის მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<style>
#grad1 {    height: 150px;
    width: 200px;
    background: -o-radial-gradient(circle, red, green, blue); /* Opera
11.1-დან 12.0-მდე */
    background: -moz-radial-gradient(circle, red, green, blue);
/* Firefox 3.6-დან 15-მდე */
    background: radial-gradient(circle, red, green, blue);
/* სტანდარტული სინტაქსი */ }
</style>
</head>
<body>
<h3>წრიული ფორმის რადიალური გრადიენტი - ფერების
თანაბარი განაწილებით</h3>
<div id="grad1"></div>
</body>
</html>
```



ჩრდილის ეფექტები CSS3-ში

CSS3-ში შესაძლებელია ჩრდილის ეფექტების დამატება ტექსტისა და ზოგიერთი ელემენტისთვის:

- text-shadow - ჩრდილის ეფექტი ტექსტისთვის;
- box-shadow - ჩრდილის ეფექტი სხვადასხვა ელემენტისთვის.

ჩრდილის ეფექტები ტექსტისათვის

CSS3-ში ტექსტისათვის ჩრდილის ეფექტები text-shadow თვისებით ხორციელდება.

თუ ამ თვისების არგუმენტებად მხოლოდ ჰორიზონტალური და ვერტიკალური ჩრდილის ზომის პარამეტრებს მივუთითებთ, მაშინ ჩრდილი შავი ფერის იქნება. ჩრდილის ფერის შესაცვლელად იმავე თვისებაში მესამე პარამეტრად ფერი უნდა დავამატოთ.

ჩრდილის ეფექტის ყველაზე მარტივი გამოყენების მაგალითში ჰორიზონტალური და ვერტიკალური ჩრდილისთვის მივუთითოთ 2px, ხოლო ფერი - წითელი. კოდს შემდეგი სახე ექნება:

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
   text-shadow: 2px 2px red;
}
</style>
</head>
<body>
<h1>ჩრდილის ეფექტები ტექსტისათვის!</h1>
</body>
</html>
```

ჩრდილის ეფექტებში შეიძლება დამატებული იყოს ბუნდოვანების ეფექტი. შესაბამისი პარამეტრი ფერის წინ დაემატება და პიქსელებში გამოისახება. მაგალითად:

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
   text-shadow: 2px 2px 5px red;
}
</style>
</head>
```

```
<body>
<h1>ჩრდილის ეფექტები ტექსტისათვის!</h1>
</body>
</html>
```

შემდეგ მაგალითში თეთრი ტექსტი შავი ჩრდილითაა ნაჩვენები:

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
   color: white;
   text-shadow: 2px 2px 4px #000000;
}
</style>
</head>
<body>
<h1>ჩრდილის ეფექტები ტექსტისათვის!</h1>
</body>
</html>
```

ქვემოთ მოცემულ მაგალითში ნაჩვენებია ტექსტის წითელი ჩრდილი ნეონის ნათების ეფექტით:

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
   text-shadow: 0 0 3px #FF0000;
}
</style>
</head>
<body>
<h1>ჩრდილის ეფექტები ტექსტისათვის!</h1>
</body>
</html>
```

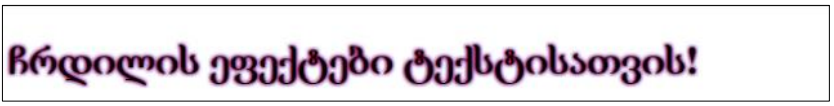
ჩრდილის რამდენიმე ეფექტის ერთდროული გამოყენება

ტექსტს შეიძლება ერთდროულად ჩრდილის რამდენიმე ეფექტი დავუმატოთ, ამისათვის საჭიროა text-shadow თვისებას არგუმენტებად ერთმანეთისაგან მძიმით გამოყოფილი ჩრდილების სია დავურთოთ.

ქვემოთ მოყვანილია მაგალითი, რომელშიც ტექსტის ორი - ლურჯი და წითელი ჩრდილია ნეონის ნათების ეფექტით:

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {   text-shadow: 0 0 3px #FF0000, 0 0 5px #0000FF;}
```

```
</style>
</head>
<body>
<h1>ჩრდილის ეფექტები ტექსტისათვის!</h1>
</body>
</html>
```



ქვემოთ მოყვანილია მაგალითი თეთრი ტექსტითა და შავი, ლურჯი და მუქი ლურჯი ჩრდილით:

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 { color: white;
 text-shadow: 1px 1px 2px black, 0 0 25px blue, 0 0 5px darkblue;}
</style>
</head>
<body>
<h1>ჩრდილის ეფექტები ტექსტისათვის!</h1>
</body>
</html>
```

ჩრდილის ეფექტი სხვადასხვა ელემენტისთვის

ელემენტისთვის ჩრდილის შესაქმნელად CSS3-ში box-shadow თვისება გამოიყენება. მისი გამოყენების დროს, როგორც ტექსტის შემთხვევაში, ჰორიზონტალური და ვერტიკალური ჩრდილის ზომები, აგრეთვე ბუნდოვანების ეფექტი და ჩრდილის ფერი უნდა მივუთითოთ. თუ ჩრდილის ფერი მითითებული არ არის, მაშინ ჩუმათობის პრინციპით ჩრდილი შავი ფერის იქნება. ქვემოთ განხილულ მაგალითში მოცემულია ყვითელი ელემენტი შავი ჩრდილით:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
   width: 300px;
   height: 100px;
   padding: 15px;
   background-color: yellow;
   box-shadow: 10px 10px;
}
</style>
</head>
<body>
<div>ეს არის div ელემენტი ჩრდილით</div>
</body>
</html>
```

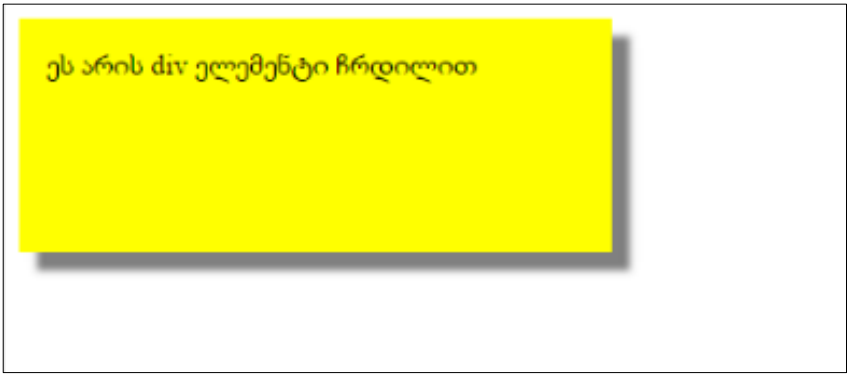
თუ გვსურს ყვითელი ელემენტის ჩრდილის ფერი იყოს, მაგალითად, ნაცრისფერი, მაშინ ჩანაწერი მიიღებს სახეს:

```
    div {
box-shadow: 10px 10px grey;
    }
```

თუ ფერს ბუნდოვანების ეფექტსაც დავუმატებთ, მაშინ კოდს შემდეგი სახე ექნება:

```
    <!DOCTYPE html>
    <html>
    <head>
    <style>
    div {
        width: 300px;
        height: 100px;
        padding: 15px;
        background-color: yellow;
        box-shadow: 10px 10px 5px grey;
    }
    </style>
```

```
</head>
<body>
<div>ეს არის div ელემენტი ჩრდილით</div>
</body>
</html>
```



გადასვლები CSS3-ში

თვისებების მნიშვნელობების ცვლილება (ერთი მნიშვნელობიდან მეორეზე გადასვლა) დროის გარკვეულ მონაკვეთში ნელა უნდა განხორციელდეს.

როგორ გამოვიყენოთ გადასვლები CSS3-ში?

იმისათვის, რომ გადასვლის ეფექტი შევქმნათ, საჭიროა CSS-ის თვისებას დავუმატოთ ეფექტი და მისი მოქმედების დრო.

თუ ეფექტის მოქმედების დრო მითითებული არ არის, მაშინ არავითარი გადასვლა არ მოხდება, ვინაიდან ჩუმათობის პრინციპით ეს დრო 0-ის ტოლია.

შემდეგ მაგალითში მოცემულია ელემენტი ზომებით 100px * 100px. ელემენტს ასევე მითითებული აქვს გადასვლის ეფექტი, რომლის მიხედვითაც 2 წამის განმავლობაში უნდა გაიზარდოს მისი სიგანე.

მაგალითი:

```
div {
   width: 100px;
   height: 100px;
   background: red;
   -webkit-transition: width 2s; /* Safari-სათვის */
   transition: width 2s; /*სტანდარტული სინტაქსი */
}
```

გადასვლის ეფექტის განხორციელება მაშინ დაიწყება, როდე-საც CSS თვისების (width) ცვლილება იქნება მითითებული. ეხლა width თვისების ახალი მნიშვნელობა მივუთითოთ და შემდეგ მაუსის მაჩვენებელი ელემენტთან მივიყვანოთ. მაგალითად:

```
div:hover {
   width: 300px;
}
```

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
   width: 100px;
   height: 100px;
   background: red;
   -webkit-transition: width 2s; /* Safari-სათვის 3.1-დან 6.0-მდე */
   transition: width 2s;  /*სტანდარტული სინტაქსი */
}
div:hover {
   width: 300px; }
```

```
    </style>
    </head>
    <body>
    <div></div>
    <p>გადასვლის ეფექტის სანახავად მაუსის მაჩვენებელი მიიტანეთ ელემენტთან.</p>
    </body>
    </html>
```

მიაქციეთ ყურადღება, რომ როდესაც მაუსის მაჩვენებელს ელემენტს მოვაცილებთ, იგი თანდათან საწყის მდგომარეობას დაუბრუნდება.

ობიექტის რამდენიმე თვისების ერთდროული ცვლილება

შემდეგ მაგალითში გადასვლის ეფექტი დავამატოთ სიგანისა და სიმაღლის ერთდროულად შეცვლის მიზნით. დავუშვათ, გვინდა სიგანის ცვლილება 2 წამში, ხოლო სიმაღლის - 4 წამში. ელემენტში განსახორციელებელ ცვლილებას შემდეგი სახე ექნება:

```
div {
  -webkit-transition: width 2s, height 4s; /* Safari-სათვის */
  transition: width 2s, height 4s;  /*სტანდარტული სინტაქსი */
}
```

 ელემენტის ახალი მნიშვნელობა იქნება:

```
div:hover {
  width: 300px;
  height: 300px; }
```

თუ წინა მაგალითში ამ ცვლილებებს შევიტანთ, მაშინ ელემენტთან მაუსის მიტანის შემთხვევაში მისი სიგანისა და სიმაღლის ცვლილება ერთდროულად მოხდება.

გადასვლის სიჩქარე

გადასვლის ეფექტის სიჩქარე შემდეგი თვისებით რეგულირდება: transition-timing-function .

transition-timing-function თვისებას შემდეგი მნიშვნელობების მიღება შეუძლია:

- ease - გადასვლის ეს ეფექტი განსაზღვრავს გადასვლას ნელი სტარტიდან უფრო სწრაფ და ბოლოს ისევ ნელ მოქმედებაზე;
- linear - გადასვლის ეს ეფექტი განსაზღვრავს ერთი და იმავე სიჩქარით გადასვლას თავიდან ბოლომდე;
- ease-in - გადასვლის ეს ეფექტი განსაზღვრავს გადასვლას ნელი სტარტით;
- ease-out - გადასვლის ეს ეფექტი განსაზღვრავს გადასვლას ნელი დასასრულით;
- ease-in-out - გადასვლის ეს ეფექტი განსაზღვრავს გადასვლას ნელი სტარტით და დასასრულით.

მაგალითად, გადასვლის შესაბამისი ეფექტები შეიძლება ასე ჩაიწეროს:

```
#div1 {transition-timing-function: linear;}
#div2 {transition-timing-function: ease;}
#div3 {transition-timing-function: ease-in;}
#div4 {transition-timing-function: ease-out;}
#div5 {transition-timing-function: ease-in-out;}
```

გადასვლის ეფექტი დაყოვნებით

გადასვლის ეფექტი დაყოვნებით (წამებში) transition-delay თვისებით ხორციელდება.

შემდეგ მაგალითში გადასვლის ეფექტის დაწყების წინ 1-წამიანი დაყოვნება ხდება

```
div {   -webkit-transition-delay: 1s; /* Safari-სათვის */
    transition-delay: 1s;  }
```

გადასვლის ეფექტი და გარდაქმნა

ქვემოთ მოცემულ მაგალითში გადასვლის ეფექტს დამატებული აქვს გარდაქმნაც, რის შედეგადაც კვადრატის ზომების გადიდება მის ბრუნვასთან ერთად ხორციელდება.

```
<!DOCTYPE html>
<html>
<head>
<style>
div {   width: 100px;
    height: 100px;
    background: red;
    -webkit-transition: width 2s, height 2s, -webkit-transform 2s;
/* Safari-სთვის */
    transition: width 2s, height 2s, transform 2s; /*სტანდარტული
სინტაქსი */   }
div:hover {   width: 300px;
                height: 300px;
    -webkit-transform: rotate(180deg); /* Safari-სათვის */
    transform: rotate(180deg);  /*სტანდარტული სინტაქსი */ }
</style>
```

```
    </head>
    <body>
    <div></div>
    <p>გადასვლის ეფექტის სანახავად მაუსის მაჩვენებელი მიიტანეთ ელემენტთან. </p>
    </body>
    </html>
```

div ტეგების საშუალებით საიტის სტრუქტურის განაწილება

იმისათვის, რომ საიტის სტრუქტურა შევქმნათ, წინასწარ უნდა გვქონდეს გააზრებული როგორი საიტის აგება გვსურს. მაგალითისათვის ავიღოთ საიტი ruseller.com.

ამ საიტის სტრუქტურას თუ დავაკვირდებით, შევამჩნევთ, რომ აქ არის ე.წ. header - საიტის სათაური, შემდეგ მოდის ზედა

მენიუ, მარცხენა მენიუ, მარჯვნივ - ავტორიზაციის ბლოკი, რეკლამები და ა.შ., შუა ნაწილი, სადაც ძირითადი ინფორმაციაა გამოტანილი და ბოლოს ე.წ. footer - ქვედა კოლონტიტული.

div ტეგების გამოყენებით ავაგოთ ასეთი სტრუქტურის საიტი: საიტის ზედა ნაწილში გვექნება საიტის სათაური (header), სათაურის ქვემოთ გვექნება ჰორიზონტალური მენიუ, დავტოვებთ ადგილს მარცხენა ნაწილის მენიუსთვის, შუა ნაწილისთვის, მარჯვენა ნაწილისთვის და, რა თქმა უნდა, გვექნება ე.წ. ქვედა კოლონტიტული (footer).

შევქმნათ index.html და style.css ფაილები. საიტის სტრუქტურას თუ დავაკვირდებით, ჯერ გვაქვს header, შემდეგ მენიუ, შემდეგ მარცხენა ნაწილი, შემდეგ შუა, შემდეგ მარჯვენა და ქვემოთ გვაქვს footer.

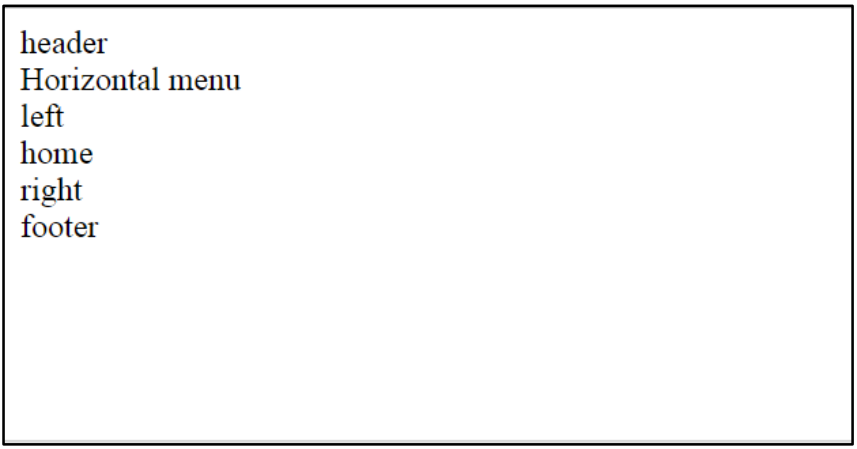
თუ საიტის სტრუქტურას ვერტიკალურად ჩამოვყვებით, მარცხენა, შუა და მარჯვენა ნაწილები ფაქტიურად გაერთიანებულია, ამიტომაც ამ ნაწილებს ერთი div ტეგის შიგნით ვსვამთ.

საიტის სტრუქტურის განაწილებას div ტეგების საშუალებით შემდეგი სახე ექნება:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>
Index
</title>
<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
<div class="header"> header</div>
<div class="horizontalMenu"> Horizontal menu </div>
```

```
<div class="body">
    <div class="left"> left</div>
    <div class="home"> home</div>
    <div class="right"> right </div>
  </div>
  <div class="footer"> footer</div>
</body>
</html>
```

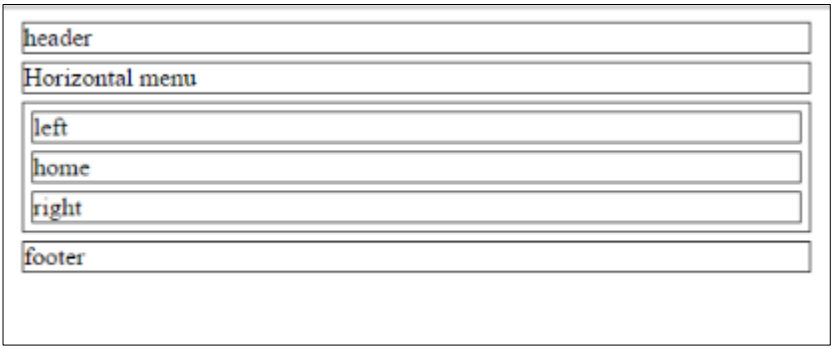
მიღებულ შედეგს აქვს სახე:



მივიღეთ ტექსტები, რომლებიც ერთმანეთის ქვემოთ იმ მიმდევრობით არის განლაგებული, რა მიმდევრობითაც ისინი div ტეგშია გაწერილი. იმის შემოწმება, თუ როგორ განლაგდება div ბლოკები ერთმანეთის ქვემოთ, საკმაოდ მარტივია. გავწეროთ ზოგადი სტილი div ბლოკისთვის:

```
div {
   border: 1px solid #333; margin:5px;
}
```

მიღებული შედეგი:



მიაქციეთ ყურადღება, რომ left, home და right ერთი div ტეგის შიგნით არის მოთავსებული.

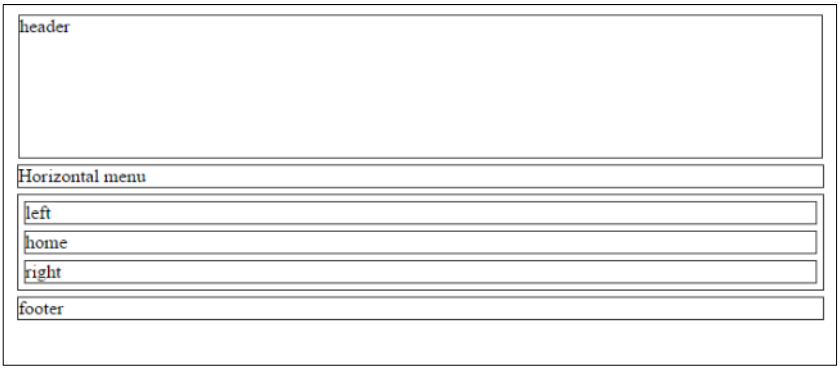
ფაქტობრივად, სტრუქტურა განაწილებულია, Header, Footer და Horizontal menu მენიუები ისე განლაგდება, როგორც იყო ჩაფიქრებული, ხოლო home, left და right ერთმანეთის გვერდი-გვერდ მოთავსდება.

როდესაც ეკრანის მასშტაბს ვზრდით, div ტეგი ზომაში მცირდება, სტატიკური ვებგვერდისთვის უმჯობესია, რომ div ტეგს ფიქსირებული ზომა ჰქონდეს ანუ ეკრანის გადიდება-დაპატარავებაზე არ იყოს დამოკიდებული. გავწეროთ სტილები header-ისთვის.

ჩვენი css ფაილი შემდეგ სახეს მიიღებს:

```
div {
border: 1px solid #333; margin:5px;
}
.header {
width:1080px; height:120px; margin:auto;
}
```

ბრაუზერის ფანჯარაში მიიღება:



ეკრანის მასშტაბის გაზრდისას header, ეკრანზე არსებული სხვა ელემენტებისგან განსხვავებით, თავის სიგანეს შეინარჩუნებს.

გავწეროთ სტილი body-სთვის: body{ background:#CCC;}. ჩავსვით რუხი ფერის ფონი.

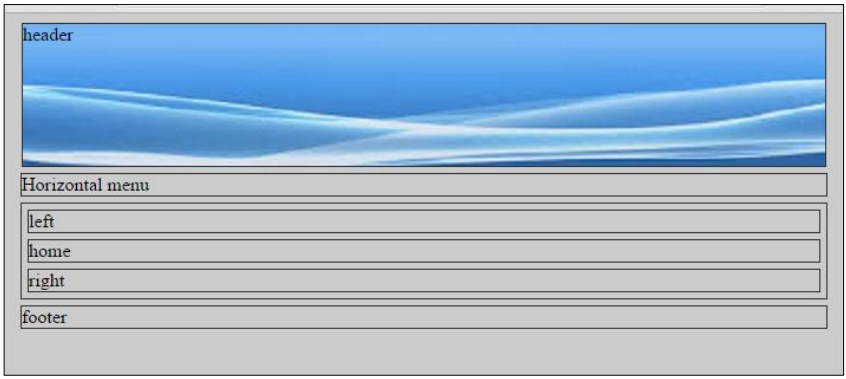
ახლა header-ისთვის ფონად სურათი ჩავსვათ. მოვძებნოთ საძიებელ სისტემაში header background სიტყვებით ზუსტად იგივე ზომის სურათი, რა ზომისაც ჩვენი header-ია. ჩვენს შემთხვევაში სურათის ზომებია: სიგანე - 1080, სიმაღლე - 120.

Style.css ფაილს ექნება სახე:

```
body{
 background:#CCC;
}
div {
border: 1px solid #333; margin:5px;
}
.header {
 width:1080px;
 height:120px;
```

```
 margin:auto;
background-image:url(header.png);
}
```

ბრაუზერის ფანჯარაში მიიღება შედეგი:



header-ის ბლოკი დავასრულეთ. შეიძლება იგი ისე გავასწოროთ, რომ ეს ბლოკი ეკრანის კიდეს ეხებოდეს. ამის გამოსწორებაც ადვილად შეიძლება. ამისათვის style.css ფაილში body-სთვის უნდა ჩავწეროთ პარამეტრი margin-top=0;

შემდეგ ეტაპზე ჰორიზონტალური მენიუსთვის სტილები დავწეროთ:

```
.horizontalMenu {
                        width:1080px;
                         min-height:40px;
                         margin:auto;
                         margin-top:5px;
                    }
```

გადავიდეთ index.html ფაილში და დავიწყოთ ჰორიზონტალური მენიუს შექმნა:

```
<div class="horizontalMenu">
```

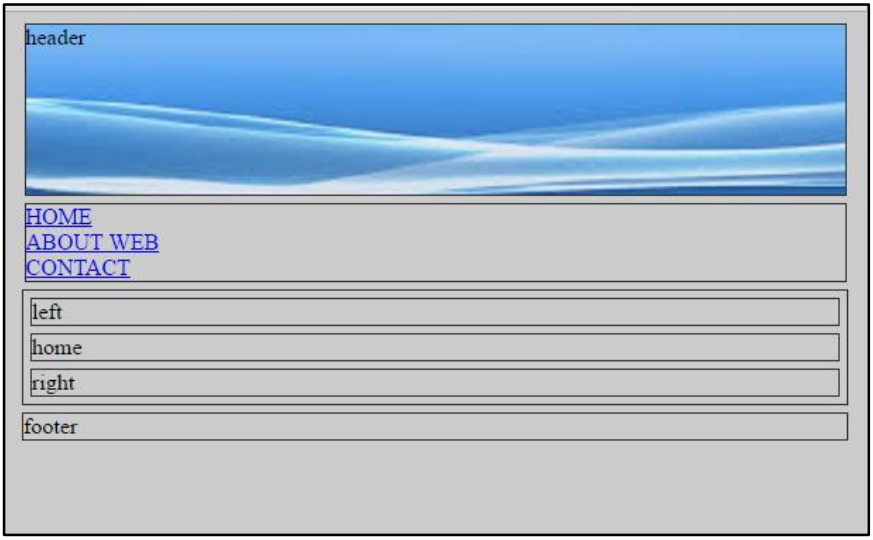
```
        <ul>
        <li> <a href="#">HOME </a></li>
        <li> <a href="#">ABOUT WEB</a></li>
        <li> <a href="#">CONTACT </a></li>
        </ul>
    </div>
```

ბრაუზერის ფანჯარაში მიიღება შედეგი:

რა თქმა უნდა, მენიუს ტექსტის ვერტიკალური განლაგება არ გვაწყობს. იმისათვის, რომ ტექსტი ჰორიზონტალურად განლაგდეს, საჭიროა სიისთვის შესაბამისი სტილების დანიშვნა მოხდეს.

```
.horizontalMenu ul {
    list-style:none;
    margin:0;
    padding:0;
}
```

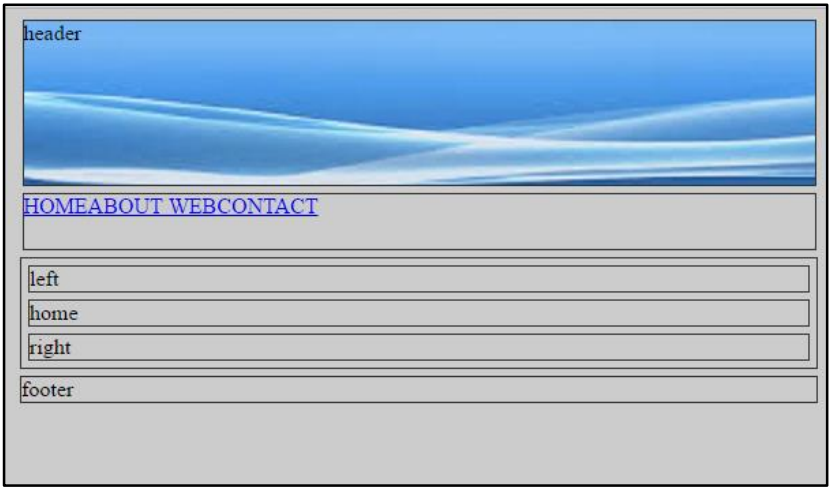
შედეგად ბრაუზერის ფანჯარაში მიიღება:



ამის შემდეგ, მენიუს ტექსტი უნდა განვალაგოთ ჰორიზონტალურად. ამისათვის სტილების ფაილში ჩავწეროთ:

```
.horizontalMenu ul li {float:left;}
```

მენიუს ტექსტი ჰორიზონტალურად განლაგდება:



გავასწოროთ ტექსტებს შორის დაშორებები. ამისათვის სტილების ფაილში ჩავამატოთ:

```
.horizontalMenu ul li a {padding:5px;}
```

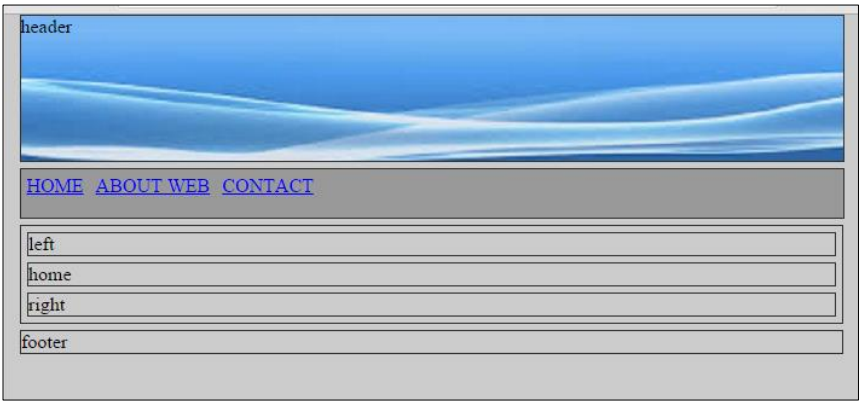
იმისათვის, რომ მენიუს ტექსტი div ტეგის კიდეს არ ეხებოდეს, ul-ისათვის დანიშნულ სტილებში ჩავამატოთ margin-top:5px. გარდა ამისა, ჰორიზონტალურ მენიუს მუქი რუხი ფერის ფონი გავუკეთოთ.

ამ დროისათვის ჩვენს css ფაილს შემდეგი სახე ექნება:

```
body{
 background:#CCC; margin-top:0;
}
div {
border: 1px solid #333; margin:5px;
}
.header                { width:1080px;
                         height:120px;
                         margin:auto;
                         background-image:url(header.png);  }
.horizontalMenu      {
                           width:1080px;
                           min-height:40px;
                           margin:auto;
                           margin-top:5px;
                           background:#999;
                          }
.horizontalMenu ul     {
                             list-style:none;
                             margin:0;
                             padding:0;
                              margin-top:5px;
```

```
                                }
.horizontalMenu ul li    {
                               float:left;
                                }
.horizontalMenu ul li a {
                                 padding:5px;
                               }
```

ბრაუზერის ფანჯარაში მიღებულ შედეგს აქვს სახე:



გადავიდეთ შემდეგ ელემენტზე, ვებგვერდის ძირითად კონტენტზე, აქ div ბლოკის შიგნით სამი div ბლოკი გვაქვს. გავიხსენოთ, რა გვაქვს index.html ფაილში ამ დროისათვის:

```
<body>
<div class="header"> </div>
<div class="horizontalMenu">
<ul>
<li> <a href="#">HOME </a></li>
<li> <a href="#">ABOUT WEB</a></li>
<li> <a href="#">CONTACT </a></li>
```

```
</ul>
 </div>
<div class="body">
    <div class="left"> left</div>
  <div class="home"> home</div>
  <div class="right"> right </div>
 </div>
 <div class="footer"> footer</div>
</body>
```

იმ div ბლოკს, რომელშიც სამი div ბლოკია მოთავსებული, body კლასი აქვს მინიჭებული, ამიტომ სტილების ფაილში ვწერთ:

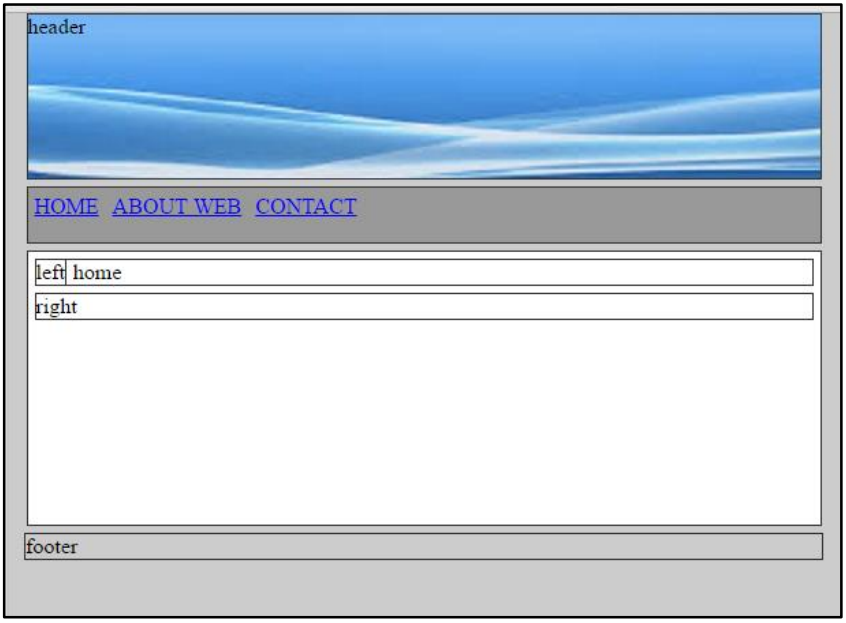
```
.body   {
            width:1080px;
            min-height:400px;
            background-color:#FFF;
            margin:auto;
            margin-top:5px;
      }
```

min-height ნიშნავს, რომ მთლიანი ბლოკის სიმაღლე 400px-ია, მაგრამ თუ მასში იმდენი ინფორმაცია იქნება, რამდენსაც ეს სიმაღლე არ ჰყოფნის, მაშინ ავტომატურად ბლოკის სიმაღლეც გაიზრდება.

ამის შემდეგ left, home და right ბლოკები გვერდიგვერდ უნდა მოვათავსოთ. დავიწყოთ left-ით. თუ style.css ფაილში დავწერთ:

```
.left {
float:left;
}
```

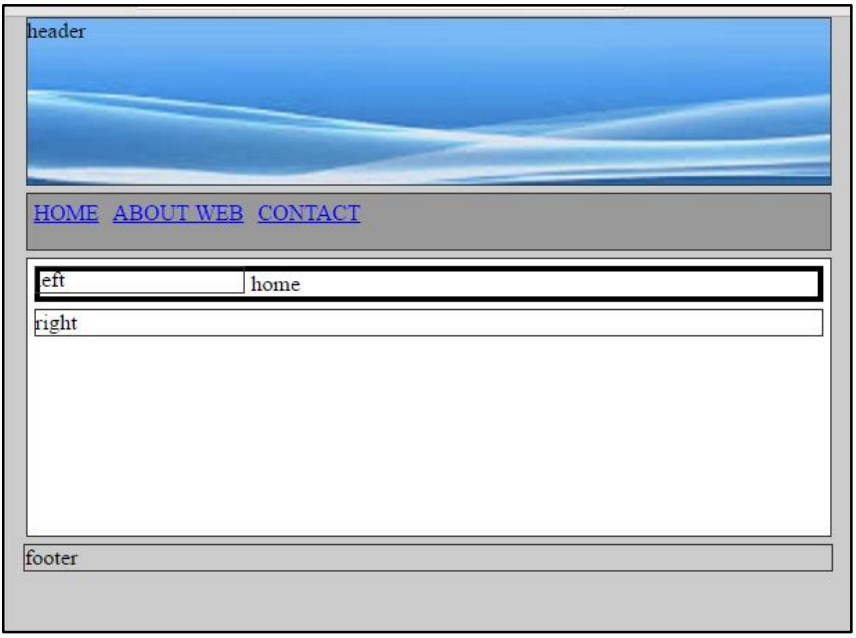
შედეგად ბრაუზერის ფანჯარაში მივიღებთ:



ეს ნიშნავს, რომ home ბლოკს აქ left ბლოკი გადაეფარა. ეს რომ თვალნათლივ დავინახოთ, სტილები home ბლოკისთვისაც ჩავწეროთ:

```
.left  {
    float:left;
      width:150px;
}
.home {
border:4px solid #000;
}
```

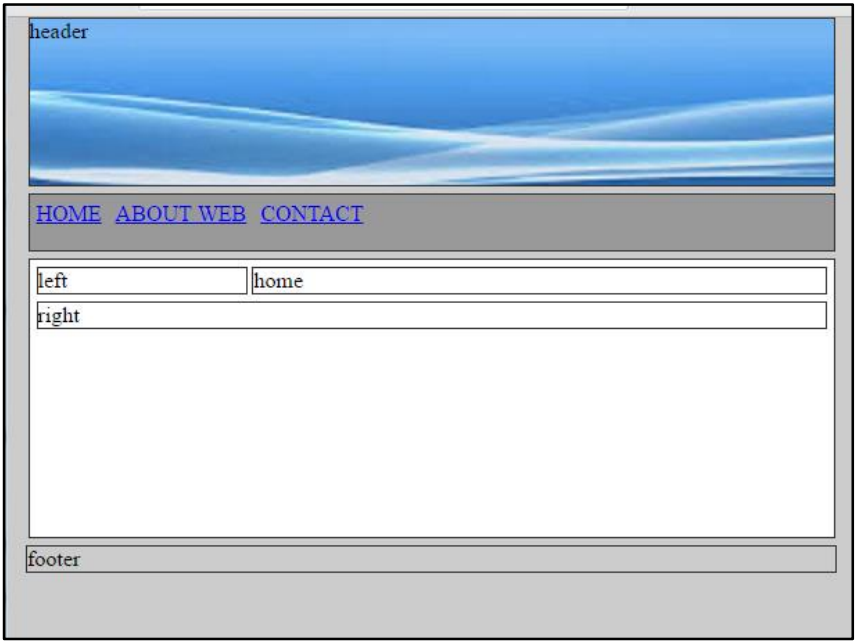
ბრაუზერის ფანჯარაში მიღებულ შედეგს აქვს შემდეგი სახე:



left ბლოკმა home ბლოკი გადაფარა, ასევე კარგად გამოჩნდა მთლიანად home ბლოკის საზღვრებიც. იმისათვის, რომ გადაფარვა არ მოხდეს, home ბლოკი მარცხნიდან 160 პიქსელით უნდა დავაცილოთ კიდეს (რადგან left ბლოკის სიგანე 150 პიქსელია).

```
.left  {
      float:left;
      width:150px;
        }
.home  {
      margin-left:160px;
}
```

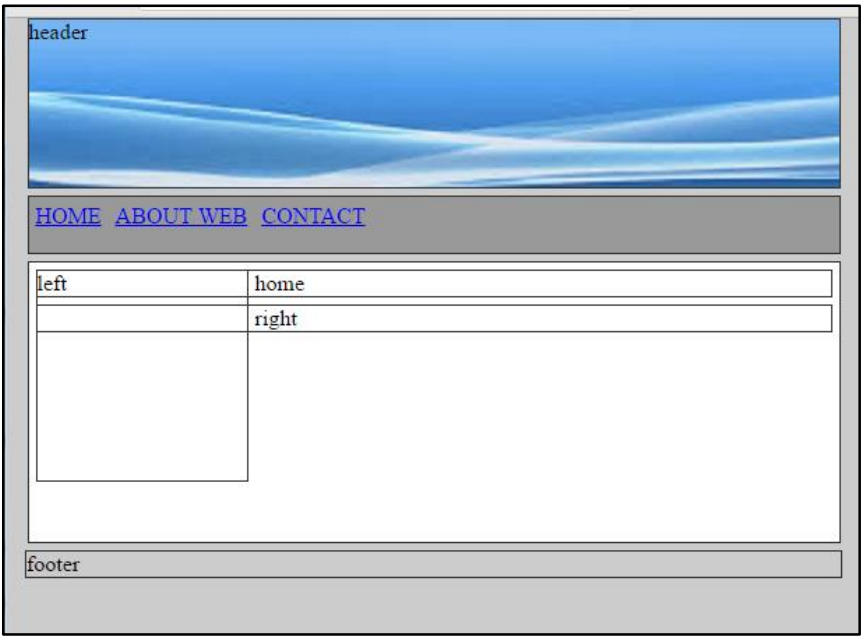
შედეგი ბრაუზერის ფანჯარაში:



თუ left ბლოკს გარკვეულ სიმაღლეს მივანიჭებთ და სტილს შემდეგნაირად დავწიშნავთ:

```
.left {
    float:left;
    width:150px;
    min-height:200px;
}
```

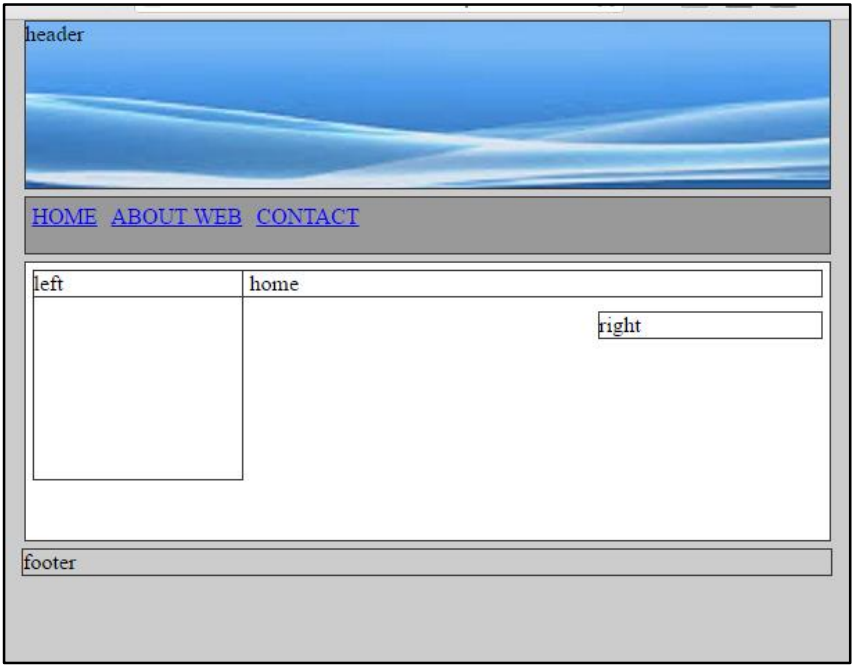
ბრაუზერის ფანჯარაში მიიღება შედეგი:



ამ შემთხვევაში, left ბლოკი right ბლოკის ნაწილს გადაფარავს. გავწიოთ right ბლოკი მარჯვნივ:

```
.right {
float:right; width:160px;
}
```

ბრაუზერის ფანჯარაში მიიღება შედეგი:



გასაგებია, რომ right ბლოკი უფრო ზემოთ უნდა მოთავსდეს. ამისათვის html ფაილში გადავიდეთ. იქ ბლოკების მიმდევრობას შემდეგი სახე აქვს: left, home, right. html ფაილში ბლოკების მიმდევრობა შევცვალოთ შემდეგნაირად:

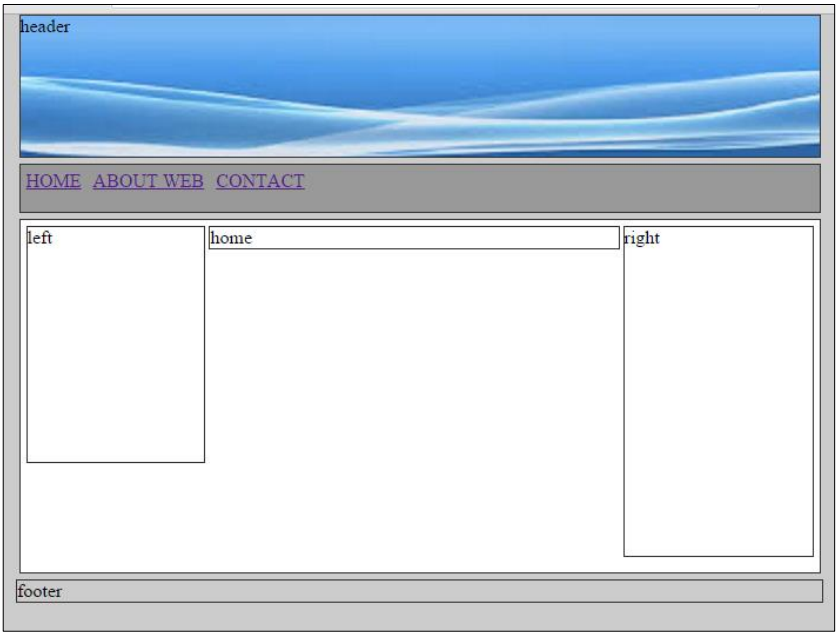
```
<div class="left"> left</div>
<div class="right"> right </div>
<div class="home"> home</div>
```

ერთი პატარა დეტალიც, იმისათვის რომ right ბლოკმა home ბლოკი არ გადაფაროს ჩავწეროთ:

```
margin:right:170px;
```

ხოლო Right ბლოკისათვის მივუთითოთ min-height:300px;

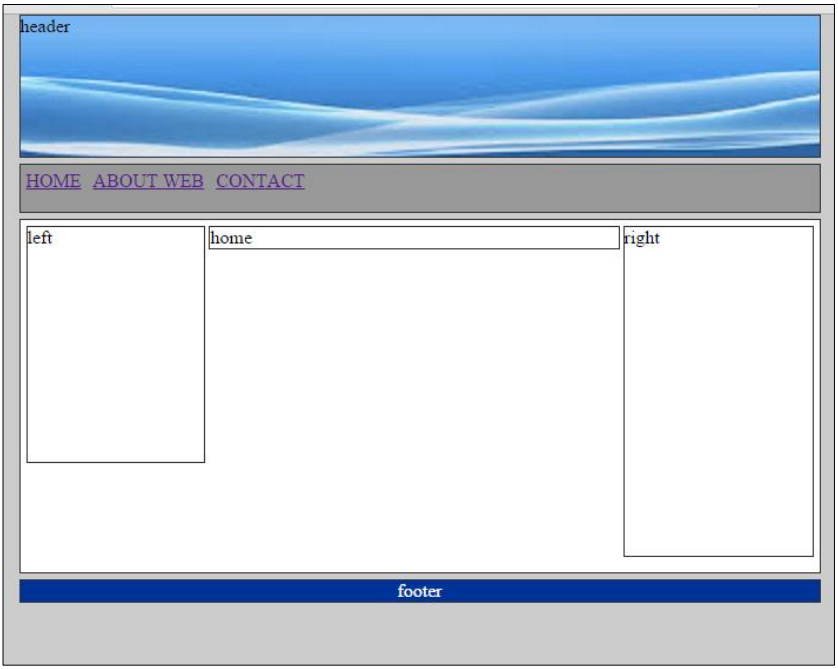
ბრაუზერის ფანჯარაში მივიღებთ:



დავგრჩა footer ბლოკი.

```
.footer
    {
    width:1080px;
    background-color:#039;
    margin:auto;
    margin-top:5px;
    text-align:center;
    }
```

ბრაუზერის ფანჯარაში მიიღება:



ფაქტობრივად, მოხდა საიტის სტრუქტურის გენერირება. იმის მიხედვით, რა მოცულობის ტექსტებს ჩავსვამთ, გაიზრდება ამ ბლოკების მოცულობაც.

საბოლოოდ, ჩვენს ფაილებს შემდეგი სახე ექნება:

Index.html

```
<DOCTYPE  html>
<html>
<head>
<meta charset="utf-8">
<title>index</title>
<link rel="stylesheet" type="text/css" href="style.css">
</head>
```

```
<body>
<div class="header"> </div>
<div class="horizontalMenu">
<ul>
<li> <a href="#">HOME </a></li>
<li> <a href="#">ABOUT WEB</a></li>
<li> <a href="#">CONTACT </a></li>
</ul>
 </div>
<div class="body">
    <div class="left"> left</div>
     <div class="right"> right </div>
     <div class="home"> home</div>
  </div>
  <div class="footer"> footer</div>
</body>
</html>
```

Style.css

```
@charset "utf-8";
/* CSS Document */
body{ background:#CCC; margin-top:0;}
div {border: 1px solid #333; margin:5px;}
.header {
    width:1080px;
     height:120px;
      margin:auto;
      background-image:url(header.png);}
.horizontalMenu {
                width:1080px;
```

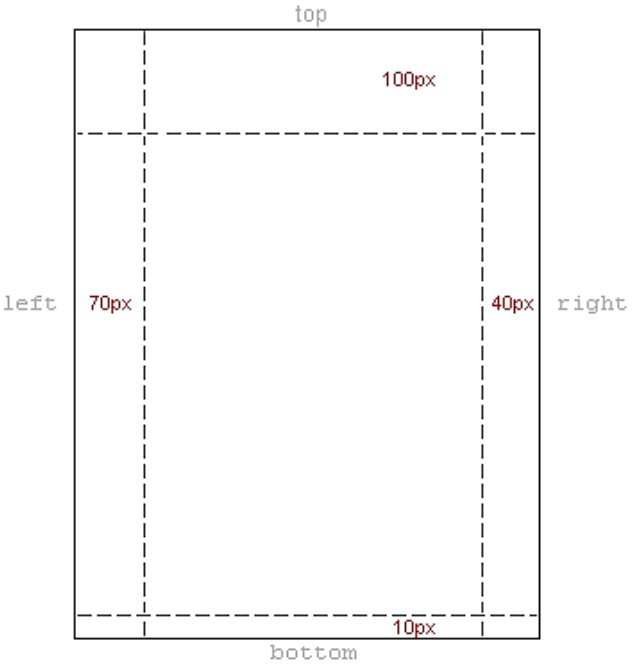
```
                    min-height:30px;
                    margin:auto;
                    margin-top:5px;
                    background:#999;
                }
.horizontalMenu ul {
    list-style:none;
    margin:0;
    padding:0;
    margin-top:5px;
}
.horizontalMenu ul li {float:left;}
.horizontalMenu ul li a {padding:5px;}
    .body {width:1080px;
            min-height:400px;
            background-color:#FFF;
            margin:auto;
            margin-top:5px;}
    .left {
            float:left;
            width:150px;
            min-height:200px;
        }
.home {
          margin-left:160px; margin-right:170px;
       }
.right  {
          float:right; width:160px; min-height:300px;
       }
.footer {
```

```
width:1080px;
 background-color:#039;
 margin:auto;
 margin-top:5px;
 text-align:center;     }
```

ელემენტის საზღვრის ველების (margin) მომართვა

ელემენტს აქვს ოთხი მხარე: right, left, top და bottom. საზღვრის ველები (margin) განსაზღვრავს თითოეული მხარიდან მეზობელ ელემენტამდე ან დოკუმენტის კიდეებამდე დაშორებას.

სურათზე ეს ველებია ნაჩვენები:



Css კოდი ამ მაგალითისთვის ასე გამოიყურება:

```
body {
margin-top: 100px; margin-right: 40px; margin-bottom: 10px;
margin-left: 70px;
 }
```

ან იგივე შემოკლებით შეიძლება ასეც ჩაიწეროს:

```
body {
margin: 100px 40px 10px 70px;
}
```

ამ დროს ჩაწერას ზედა ველიდან ვიწყებთ და საათის ისრის მიმართულებით ვაგრძელებთ.

ამგვარად, შეგიძლიათ ელემენტის საზღვრის ველები თითქმის ნებისმიერი ელემენტისთვის დააყენოთ. მაგალითად, შესაძლებელია საზღვრის ველების განსაზღვრა ყველა პარაგრაფისთვის:

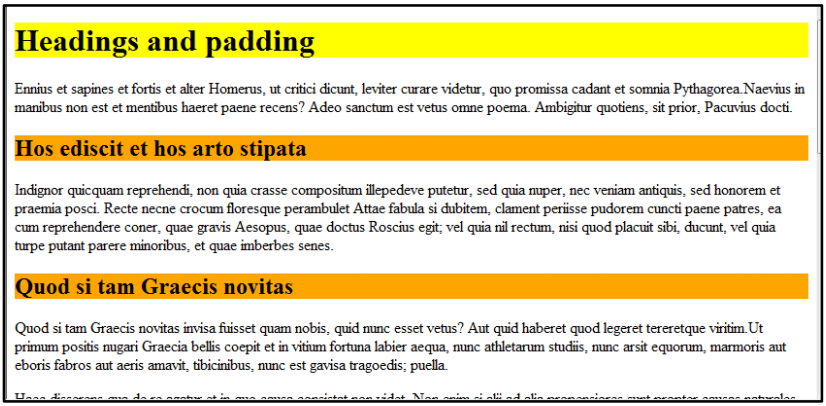
```
Body  {
margin: 100px 40px 10px 70px;
}
 p {
margin: 5px 50px 5px 50px;
}
```

ჩარჩოსა და ელემენტს შორის დაშორების განსაზღვრა

ჩარჩოსა და ელემენტის შემცველობას შორის შიგა დაშორებას padding ელემენტი განსაზღვრავს.

padding-ის გამოყენება მარტივ მაგალითზე შეიძლება ვაჩვენოთ, სადაც ყველა სათაურს ფერადი ფონი აქვს.

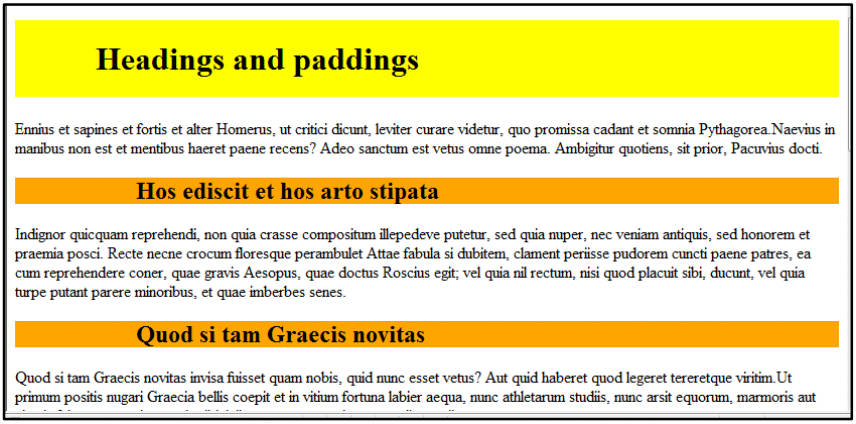
```
h1 {
background: yellow;
}
h2 {
  background: orange;
}
```



მართალია, padding-ის საშუალებით სათაურისათვის საიტის კიდეებიდან დაცილებას განვსაზღვრავთ, მაგრამ ამავე კოდით საზღვრის ველების სიდიდე სათაურის მთელი ტექსტისთვის ავტომატურად განისაზღვრება:

```
h1  {
    background: yellow; padding: 20px 20px 20px 80px; }
h2 {
    background: orange; padding-left:120px;
     }
```

ამ კოდის ჩაწერის შემდეგ ვებგვერდს შემდეგი სახე ექნება:

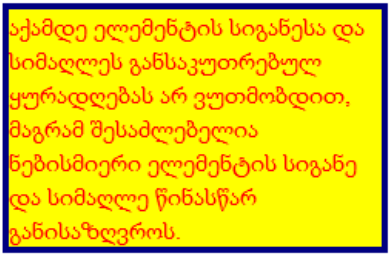


ელემენტების სიგანე და სიმაღლე

width თვისება ელემენტის სიგანეს განსაზღვრავს, მაგალითად:

```
div.box {
    width: 250px;
    border: 4px solid navy;
    background: yellow;
    color: red; }
```

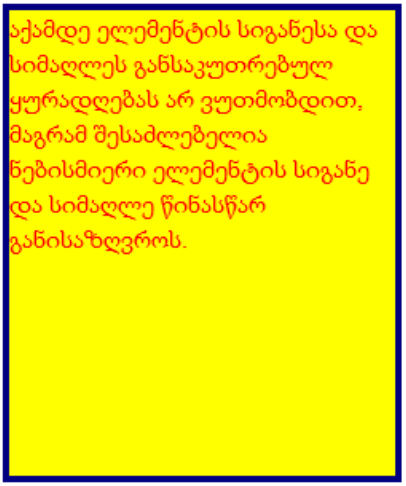
შედეგად ტექსტური ბლოკი 250 პიქსელი სიგანის, 4 პიქსელი მუქი ლურჯი ჩარჩოთი, ყვითელი ფონითა და წითელი შრიფტით გვექნება:



როდესაც ელემენტის (მოცემულ შემთხვევაში ტექსტური ბლოკი) სიმაღლე მითითებული არ არის, მაშინ ის მასში მოცემული ტექსტით განისაზღვრება. ზოგადად, ელემენტის სიმაღლე შეიძლება height თვისებით განვსაზღვროთ.

```
div.box {
    height: 300px;
    width: 250px;
    border: 4px solid navy;
    background: yellow;
    color: red;         }
```

შედეგად ასეთი სახის ტექსტური ბლოკი მიიღება:



CSS-ში სურათის გამჭვირვალობა

CSS-ში გამჭვირვალე სურათების შექმნა მარტივია.

შევქმნათ გამჭვირვალე გამოსახულება. ამისათვის CSS3-ის opacity თვისება გამოვიყენოთ.

საწყისი გამოსახულება:

გარდაქმნის შემდეგ ის ასეთი გახდება:

შესაბამის კოდს შემდეგი სახე ექნება:

```
<!DOCTYPE html>
<html>
<head>
<style>
img {
   opacity: 0.4;
}
</style>
```

```
</head>
<body>
<h3>სურათის გამჭვირვალობა</h3>
<img src="roze.jpg" alt="ვარდი">
</body>
</html>
```

opacity თვისებას შეუძლია მიიღოს მნიშვნელობები 0.0-დან 1.0-მდე. რაც მცირეა მისი მნიშვნელობა, მით უფრო გამჭვირვალეა გამოსახულება.

ეხლა განვიხილოთ მაგალითი, როდესაც გამჭვირვალობის ეფექტს გამოსახულებაზე მაუსის მაჩვენებლის მიახლოების დროს გამოვიყენებთ:

კოდს შემდეგი სახე ექნება:

```
<!DOCTYPE html>
<html>
<head>
<style>
img {
   opacity: 0.4;
}
img:hover {
   opacity: 1.0;}
</style>
</head>
<body>
<h1>სურათის გამჭვირვალობა</h1>
<img src="roze.jpg" width="150" height="113" alt="ვარდი">
</body>
</html>
```

ამ კოდის პირველი ბლოკი წინა მაგალითის მსგავსია, მაგრამ ჩვენ მეორე ბლოკიც დავამატეთ, რომლის მიხედვითაც შემდეგი მოქმედება ხორციელდება: თუ მომხმარებელი მაუსს გამოსახულებასთან მიიყვანს, გამოსახულება გაუმჭვირი გახდება ანუ opacity:1;. გამოსახულებიდან მაუსის მაჩვენებლის გადაადგილების შემთხვევაში გამოსახულება ისევ გამჭვირვალე ხდება.

CSS3-ის იმავე თვისების გამოყენებით შესაძლებელია გამოსახულების გამჭვირვალე ნაწილზე ტექსტი განვათავსოთ. ასეთ კოდს შემდეგი სახე ექნება:

```
<!DOCTYPE html>
<html>
<head>
<style>
div.background {
    background: url(roze.jpg) repeat;
    border: 2px solid black;
}
div.transbox {
    margin: 75px;
    background-color: #ffffff;
    border: 1px solid black;
    opacity: 0.6;
    }
div.transbox p {
    margin: 5%;
    font-weight: bold;
    color: #000000;
}
</style>
```

```
    </head>
    <body>
    <div class="background">
      <div class="transbox">
        <p>ეს ტექსტი გამოსახულების გამჭვირვალე ნაწილშია
განთავსებული
    .</p>
      </div>
    </div>
    </body>
    </html>
```

პირველ რიგში ელემენტით (class="background") შეიქმნა ელემენტი გამოსახულებიანი ფონით და ჩარჩოთი. შემდეგ ჩვენ პირველი -ის შიგნით კიდევ ერთ -ს (class="transbox") ვქმნით. -ს კი ფონად და ჩარჩოდ გამჭვირვალე გამოსახულება აქვს. გამჭვირვალე არის შიგნით კი <p> ელემენტის საშუალებით განთავსებულია ტექსტი.

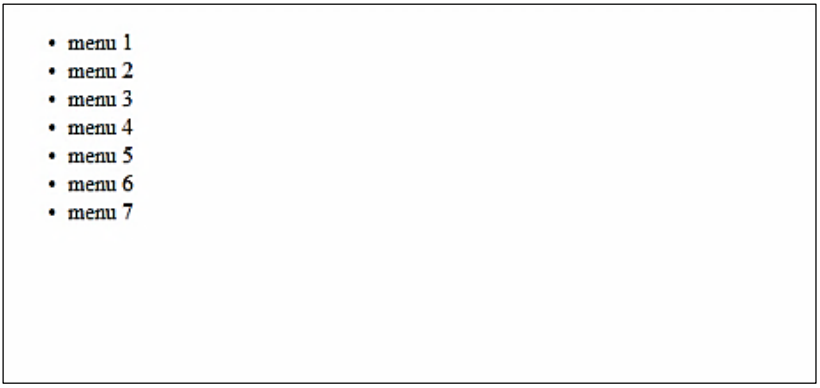
ორდონიანი მენიუს აწყობა

ჩვენი ამოცანა მრავალდონიანი მენიუს შექმნაა. ჯერ შევქმნათ ფოლდერი menu და შემდეგ მასში index.html ფაილი. სანამ მრავალდონიან მენიუს ავაწყობდეთ, ერთდონიანი მენიუ შევქმნათ. ამისათვის css ფაილიც დაგვჭირდება. შევქმნათ style.css დოკუმენტი.

ასეთი ტიპის მენიუ მარკირებული სიის გამოყენებით იწყობა. ჩვენს html ფაილს შემდეგი სახე ექნება:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>menu</title>
<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
<ul>
    <li>menu 1</li>
    <li>menu 2</li>
    <li>menu 3</li>
    <li>menu 4</li>
    <li>menu 5</li>
    <li>menu 6</li>
    <li>menu 7</li>
</ul>
</body>
</html>
```

ბრაუზერის ფანჯარაში მივიღებთ:



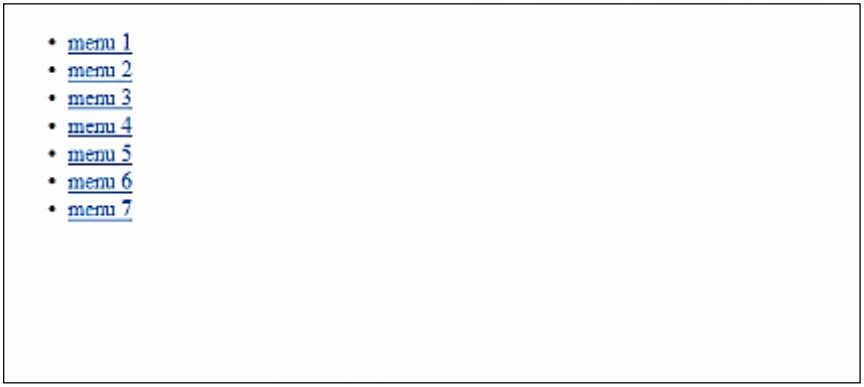
რა თქმა უნდა, როცა მენიუზე გვაქვს საუბარი, ბმულების შექმნა აუცილებელია. ბმულებიც ჩავსვათ.

html ფაილს ექნება სახე:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>menu</title>
<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
<ul>
<li><a href="#">menu 1 </a> </li>
<li><a href="#">menu 2 </a></li>
<li><a href="#">menu 3 </a></li>
<li><a href="#">menu 4 </a></li>
<li><a href="#">menu 5 </a></li>
<li><a href="#">menu 6 </a></li>
```

```
<li><a href="#">menu 7 </a></li>
</ul>
</body>
</html>
```

ბრაუზერის ეკრანზე მიიღება შედეგი:



ვთქვათ, გვინდა menu3, menu5 და menu7-ზე ჩამოსაშლელი მენიუ შევქმნათ.

იქ, სადაც ჩამოსაშლელი მენიუ გვინდა იყოს, <li> ტეგებს შორის ტეგის შემდეგ ვდგებით და შიგნით <ul> სიას ვსვამთ. ჩვენი html ფაილი მიიღებს სახეს:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>menu</title>
<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
<ul>
```

```
<li><a href="#">menu 1 </a> </li>
<li><a href="#">menu 2 </a></li>
<li><a href="#">menu 3 </a>
        <ul>
                <li><a href="#">menu 3.1 </a> </li>
                <li><a href="#">menu 3.2 </a> </li>
                <li><a href="#">menu 3.3 </a> </li>
        </ul>
</li>
<li><a href="#">menu 4 </a></li>
<li><a href="#">menu 5 </a>
      <ul>
                <li><a href="#">menu 5.1 </a> </li>
                <li><a href="#">menu 5.2 </a> </li>
                <li><a href="#">menu 5.3 </a> </li>
                <li><a href="#">menu 5.4 </a> </li>
        </ul>
</li>
<li><a href="#">menu 6 </a>
        <ul>
                <li><a href="#">menu 6.1 </a> </li>
                <li><a href="#">menu 6.2 </a> </li>
        </ul>
</li>
<li><a href="#">menu 7 </a></li>
</ul>
</body>
</html>
```

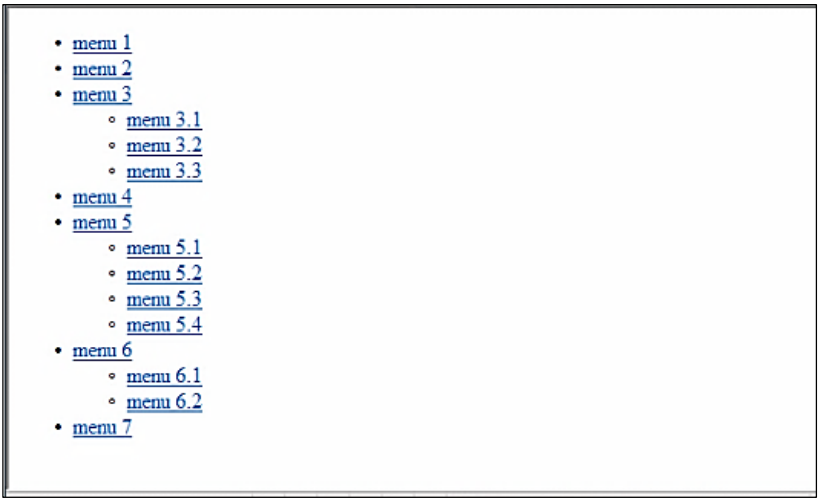
ძირითად <ul> ტეგს მივანიჭოთ კლასი: <ul class=”menu”>

გადავიდეთ სტილების ფაილში. მუშაობა დავიწყოთ menu კლასთან. პირველი, რასაც ვწერთ, არის:

```
.menu {
  list-style:none
}
```

ეს ბრძანება გარეთა <ul> ტეგს სტილებს დააშორებს.

ბრაუზერის ფანჯარაში მივიღებთ:



ამის შემდეგ css ფაილში დავამატოთ Margin:0 და Padding:0. საბოლოოდ css ფაილში მივიღებთ:

```
.menu {
list-style:none;
margin:0;
padding:0;
}
```

ამ შემთხვევაში ამ პარამეტრების დამატებას რაიმე განსაკუთრებული მნიშვნელობა არა აქვს, მაგრამ როდესაც ეს მენიუ

საიტის სტრუქტურაში ზის, მაშინ შესაძლოა მნიშვნელოვანი გახდეს.

ამ დროისათვის ჩვენს index.html ფაილს ექნება სახე:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>
Menu
</title>
<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
<ul class="menu">
<li><a href="#">menu 1 </a> </li>
<li><a href="#">menu 2 </a></li>
<li><a href="#">menu 3 </a>
        <ul>
        <li><a href="#">menu 3.1 </a> </li>
        <li><a href="#">menu 3.2 </a> </li>
        <li><a href="#">menu 3.3 </a> </li>
        </ul>
</li>
<li><a href="#">menu 4 </a></li>
<li><a href="#">menu 5 </a>
<ul>
        <li><a href="#">menu 5.1 </a> </li>
        <li><a href="#">menu 5.2 </a> </li>
        <li><a href="#">menu 5.3 </a> </li>
```

```
            <li><a href="#">menu 5.4 </a> </li>
                    </ul>
        </li>
        <li><a href="#">menu 6 </a>
                    <ul>
                    <li><a href="#">menu 6.1 </a> </li>
                    <li><a href="#">menu 6.2 </a> </li>
                    </ul>
        </li>
        <li><a href="#">menu 7 </a></li>
        </ul>
        </body>
        </html>
```

Style.css ფაილს შემდეგი სახე აქვს:

```
@charset "utf-8";
/* CSS Document */
.menu {
          list-style:none;
          margin:0;
          padding:0;
          }
```

გადავიდეთ სტილების ფაილში და ჩავწეროთ:

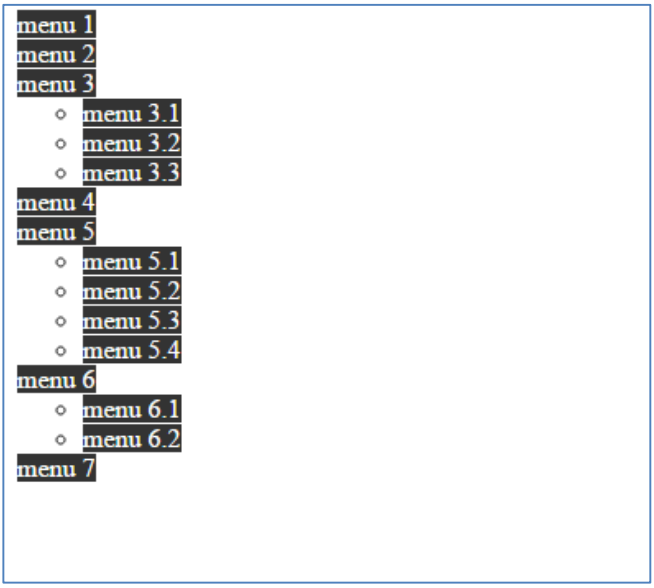
```
        . menu li a
                    {
                    text-decoration: none;
                    }
```

შედეგად ბრაუზერის ფანჯარაში მიიღება:

გავაგრძელოთ დაწერა სტილების ფაილში და ფონისა და ტექსტის ფერი დავამატოთ.

```
.menu  {
        list-style:none;
        margin:0;
        padding:0;
        }
.menu li a     {
                text-decoration:none;
                background-color:#333;
                color:#FFF;
               }
```

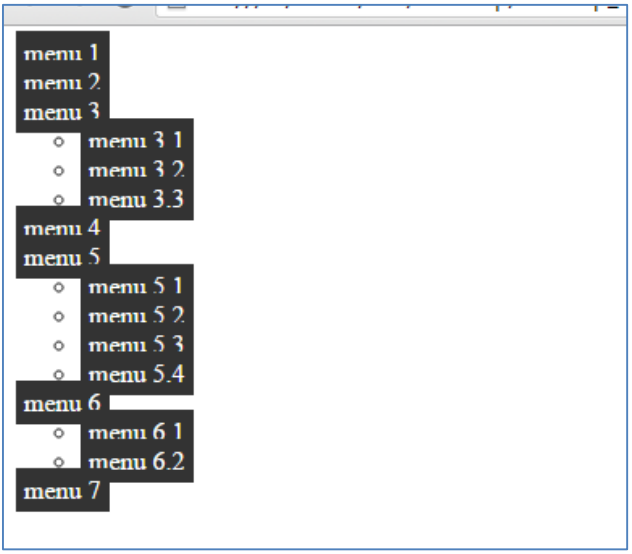
ბრაუზერის ფანჯარაში მივიღებთ შედეგს:



მოვაცილოთ ტექსტებს ხაზგასმები და დავაშოროთ შიგა საზღვრებიდან:

```
.menu  {
        list-style:none;
        margin:0;
        padding:0;
         }
.menu li a {text-decoration:none;
            background-color:#333;
            color:#FFF;
            padding:5px;
            }
```

ბრაუზერის ფანჯარაში მივიღებთ:



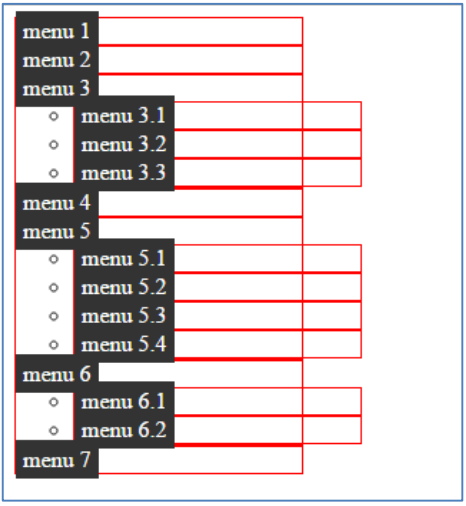
როგორც ხედავთ, ზოგმა მენიუმ ერთმანეთი გადაფარა. ამიტომ <li> ტეგს მივცეთ სიგანე:

```
.menu li  {
  width:200px;
   }
```

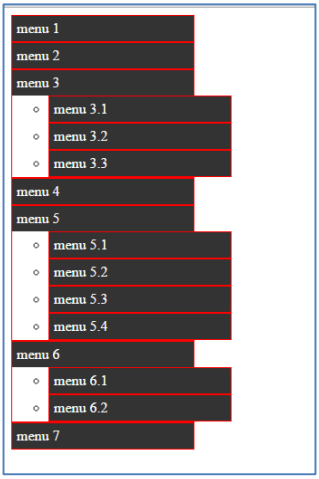
ბრაუზერის ფანჯარაში არავითარი ცვლილება არ გამოჩნდება. თუ border პარამეტრს დავამატებთ, მაშინ ამ ცვლილებებს თვალნათლივ დავინახავთ.

```
.menu li   {
     width:200px;
      border:1px solid #F00;
     }
```

მიღებულ შედეგს ექნება სახე:



ახლა მიზნად დავისახოთ, რომ ბმული არა მხოლოდ იქ იყოს, სადაც menu1, menu2 და სხვა მენიუს ელემენტები გვაქვს, არამედ მთლიანად იმ არეში, რომელიც წითელი ჩარჩოთია შემოსაზღვრული. ამისათვის, <a> ტეგისთვის დავამატოთ: display:block.



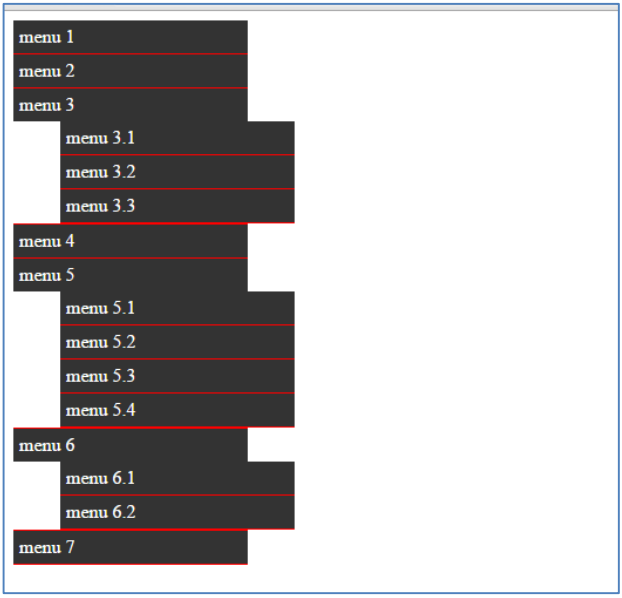
ახლა ბმული უკვე ყველგანაა, რადგანაც a ბლოკური ელემენტი გახდა. მიღებული შედეგით დავინახავთ, რომ ჩარჩო ზოგან სქელია, ზოგან შედარებით თხელი. ამისათვის სტილების ფაილში ჩავწეროთ: border:bottom:1px.

შედეგად მივიღებთ:

შემდეგი ნაბიჯით მარკირება ჩამოშლად მენიუებსაც მოვაშოროთ. გარე <ul> ტეგს მარკირება მოშორებული აქვს, რადგან სტილების ფაილში .menu {list-style:none;} გვაქვს ჩაწერილი. ახლა მარკირება შიგა <ul> ტეგსაც მოვაშოროთ. ამისათვის სტილების ფაილში ვწერთ:

```
.menu ul {
                list-style:none;
                 }
```

ბრაუზერის ფანჯარაში მივიღებთ:

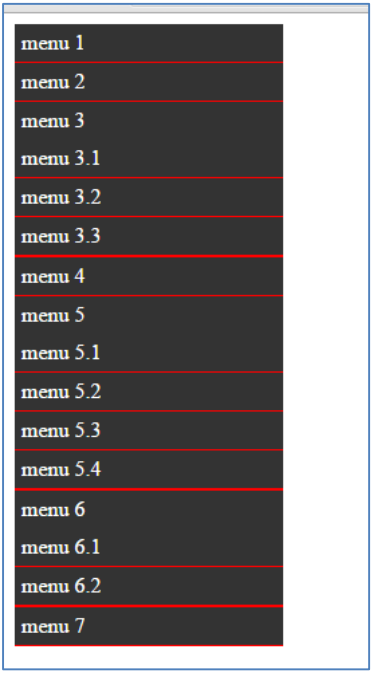


მივიღოთ შემდეგი აღნიშვნები: menu1, menu2, menu3, menu4, menu5, menu6, menu7-ს - გარე <ul> ტეგის ელემენტები, ხოლო menu3.1, menu3.2, menu 3.3, menu 5.1, menu5.2, menu5.3, menu5.4, menu6.1, menu 6.2-ს - შიგა <ul> ტეგის ელემენტები ვუწოდოთ. შიგა <ul> (შეგვიძლია მას მეორე დონის <ul> ტეგიც დავარქვათ) უნდა შეიწიოს მარცხნივ და საერთოდ გაქრეს. ამ ელემენტების მარცხნივ გადასაწევად margin და padding ელემენტების განულება დაგვჭირდება.

სტილების ფაილში ვწერთ:

```
.menu ul {
        list-style:none;
        margin:0;
        padding:0;}
```

ბრაუზერის ფანჯარაში მივიღებთ:



ამ დროისათვის html და css ფაილებს შემდეგი სახე ექნება:

Index.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>menu</title>
<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
<ul class="menu">
```

```
<li><a href="#">menu 1 </a> </li>
<li><a href="#">menu 2 </a></li>
<li><a href="#">menu 3 </a>
  <ul>
        <li><a href="#">menu 3.1 </a> </li>
        <li><a href="#">menu 3.2 </a> </li>
        <li><a href="#">menu 3.3 </a> </li>
  </ul>
</li>
<li><a href="#">menu 4 </a></li>
<li><a href="#">menu 5 </a>
<ul>
        <li><a href="#">menu 5.1 </a> </li>
        <li><a href="#">menu 5.2 </a> </li>
        <li><a href="#">menu 5.3 </a> </li>
        <li><a href="#">menu 5.4 </a> </li>
  </ul>
</li>
<li><a href="#">menu 6 </a>
  <ul>
        <li><a href="#">menu 6.1 </a> </li>
        <li><a href="#">menu 6.2 </a> </li>
  </ul>
</li>
<li><a href="#">menu 7 </a></li>
</ul>
</body>
</html>
```

Style.css

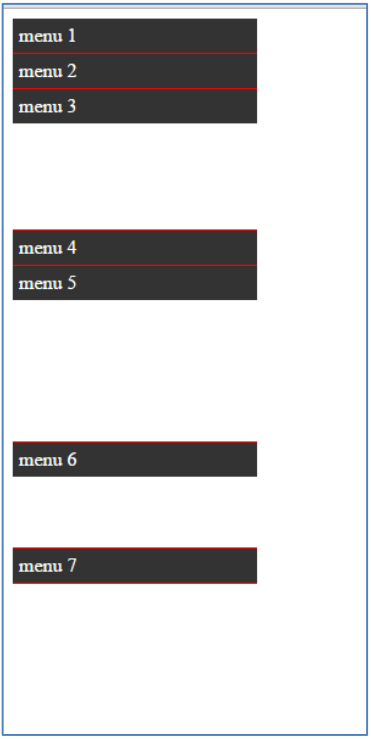
```
@charset "utf-8";
/* CSS Document */
.menu   {list-style:none;
margin:0;
padding:0; }

.menu li {width:200px;
                                        border-bottom:1px solid #F00;
          }
.menu li a {text-decoration:none;
                                       background-color:#333;
                                       color:#FFF;
                                      padding: 5px;
                                      display:block;
                                      }
.menu ul {list-style:none;
              margin:0;
              padding:0;}
```

ამის შემდგომ ჩვენს ამოცანად menu3.1, menu3.2, menu3.3, menu5.1, menu5.2, menu5.3, menu5.4, menu6.1, menu6.2 ბმულების საერთოდ გაქრობა უნდა დავსახოთ. ამისათვის სტილების ფაილში დავამატოთ visibility:hidden სტილი.

```
menu ul {list-style:none;
              margin:0;
              padding:0;
              visibility:hidden;
            }
```

ბრაუზერის ფანჯარაში მივიღებთ:

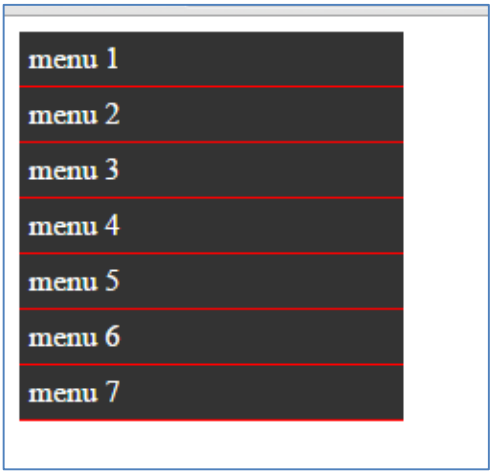


ამით მეორე დონის მენიუები გაქრა. ეს არის ის, რისი მიღებაც ამ დროისათვის გვინდოდა. ამ შემთხვევაში მენიუ კი გაქრა, მაგრამ ადგილები ცარიელი დარჩა ანუ მენიუ არ „აიკეცა“. ასეთ შემთხვევაში, სტილების ფაილში უნდა მივუთითოთ position: absolute.

ანუ შიგა <ul> ტეგებისთვის სტილების ფაილში გვექნება:

```
.menu ul {list-style:none;
          margin:0;
          padding:0;
          visibility:hidden;
          position: absolute;}
```

მიღებულ შედეგს აქვს სახე:

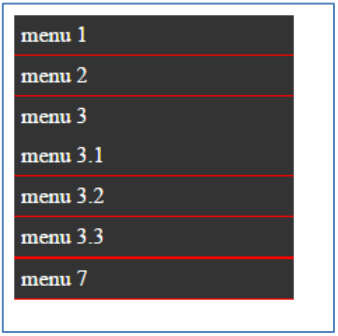


ამით შეიძლება ითქვას, რომ მენიუს შექმნის პირველი ეტაპი დამთავრდა.

შემდეგ ისეთი კოდი უნდა დაიწეროს, რომ menu3-თან მაუსის მიტანისას ჩამოსაშლელი მენიუ გამოჩნდეს. ამისათვის, სტილების ფაილში ჩავამატოთ:

```
.menu li:hover ul {visibility:visible;}
```

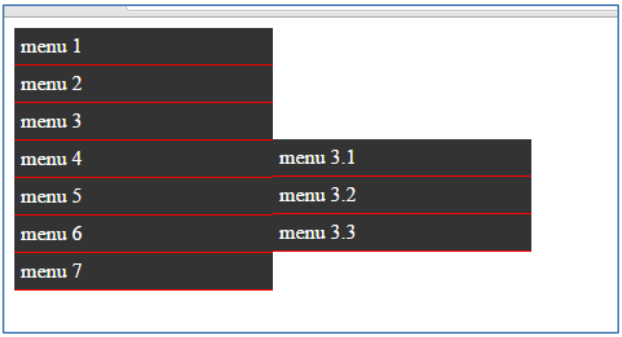
მიღებულ შედეგს ბრაუზერის ფანჯარაში აქვს შემდეგი სახე:



menu3-ზე მაუსის გადატარებისას menu3.1, menu3.2 და menu3.3 ქვემენიუები ჩამოიშლება. იგივე მოხდება menu5-ზე ან menu6-ზე მაუსის გადატარების შემთხვევაშიც. ცუდი ისაა, რომ მენიუების ჩამოშლა ხდება შიგნით და არა გარეთ ისე, რომ არ გადაეფაროს სხვა ელემენტებს. ჩვენ გვინდა, რომ ეს ჩამოშლადი მენიუ გავიდეს გვერდზე, მარჯვნივ. ამისათვის სტილების ფაილში ვწერთ:

```
.menu li:hover ul {visibility:visible;
                   margin-left:200px;
                   }
```

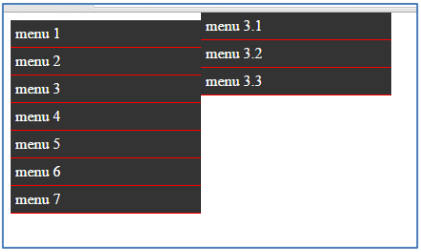
მიღებული შედეგი:



მეორე დონის მენიუ გაიწია მარჯვნივ. ამჯერად გასასწორებელია ის, რომ menu3.1 დგას menu4-ის გასწვრივ. ეს უნდა გასწორდეს და ზემოთ აიწიოს. ამისათვის სტილების ფაილში ვწერთ:

```
.menu li:hover ul {visibility:visible;
                   margin-left:200px;
                   top:0;}
```

მიღებული შედეგი:

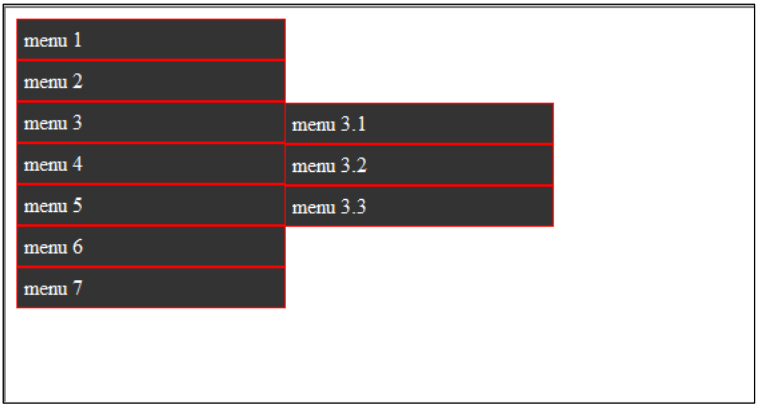


როგორც ხედავთ, მეორე დონის მენიუ კი აიწია ზემოთ, მაგრამ გასწორდა არა menu3-ის მიმართ, არამედ მიუახლოვდა სულ ზედა კიდეს. ამის გამოსასწორებლად სტილების ფაილში ვწერთ:

```
.menu li {width:200px;
          border-bottom:1px solid #F00;
          position:relative;
          }
```

ანუ აქ <li> ტეგის მიმართ (menu3 არის <li> ტეგში მოთავსებული) მოხდება გასწორება.

შედეგად ბრაუზერის ფანჯარაში მივიღებთ:



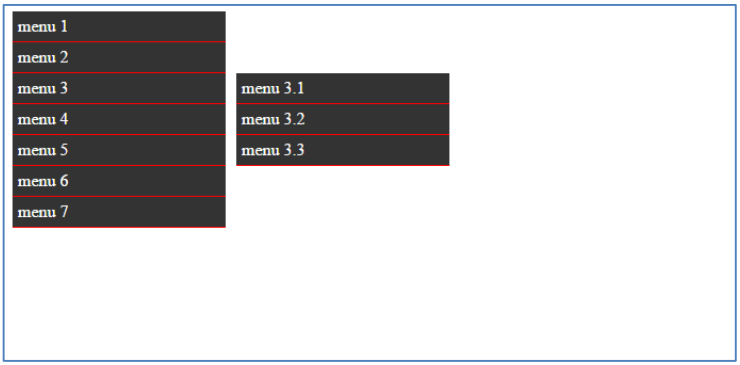
ახლა გავაკეთოთ ისე, რომ პირველი დონის მენიუზე მაუსის გადატარებისას, პირველი დონის მენიუსა და მეორე დონის (ჩამოშლად) მენიუს შორის წყვეტა იყოს ანუ ადგილი დარჩეს.

ამ დროისათვის ჩვენს სტილების ფაილს შემდეგი სახე აქვს:

```
@charset "utf-8";
/* CSS Document */
.menu {list-style:none;
        margin:0;
        padding:0; }
.menu li {width:200px;
          border-bottom:1px solid #F00;
          position:relative;
         }
.menu li a {text-decoration:none;
            background-color:#333;
            color:#FFF;
            padding: 5px;
            display:block;
           }
.menu ul {list-style:none;
          margin:0;
          padding:0;
          visibility:hidden;
          position: absolute;}
.menu li:hover ul {visibility:visible;
                   margin-left:200px;
                   top:0;}
```

თუ გვსურს, რომ პირველი დონის მენიუსა და ჩამოსაშლელ მენიუს შორის წყვეტა გავაკეთოთ, ამისათვის margin-left:200px-ის

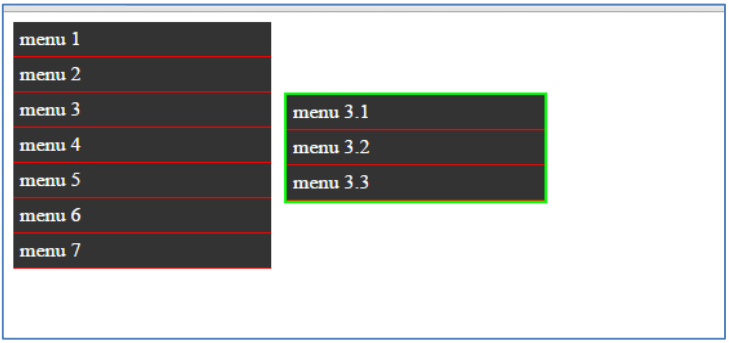
ნაცვლად დავწეროთ margin-left:210px, მენიუ გაიწევა და წყვეტა მოხდება, მაგრამ წყვეტა იქნება ისეთი, რომ პირველი დონის მენიუზე მაუსის გადატარებისას უკვე მეორე დონის მენიუზე გადასვლას ვეღარ მოვახერხებთ.



ჩამოსაშლელი მენიუსთვის გავაკეთოთ ჩარჩო:

```
.menu li:hover ul {visibility:visible;
                   margin-left:210px;
                   top:0;
                   border:2px solid #0F0;}
```

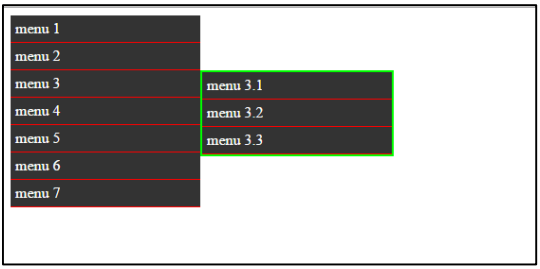
ასეთ შემთხვევაში მივიღებთ:



ჩამოსაშლელი მენიუ ძირითად მენიუს ფაქტობრივად მოწყვეტილია. სწორედ ამის გამო, რომ ეს წყვეტა არ გვქონდეს, 210px-ის ჩაწერა გამართლებული არ იქნება, ისევ 200px ჩავწეროთ ანუ გვექნება:

```
.menu li:hover ul {visibility:visible;
                   margin-left:200px;
                   top:0;
                   border:2px solid #0F0;}
```

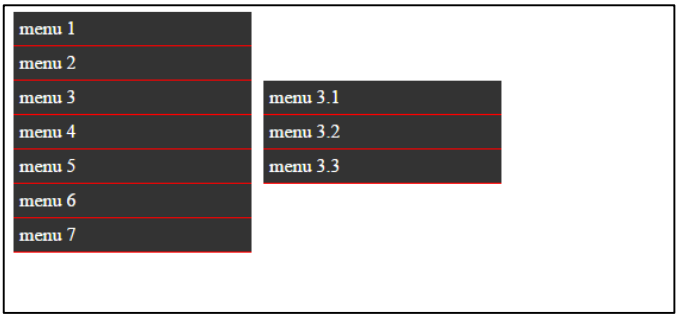
მიღებული შედეგი:



ახლა უკვე ჩამოსაშლელ მენიუზე უპრობლემოდ გადასვლა შესაძლებელია. თუმცა, თავიდან მიზნად დავისახეთ, რომ პირველ და მეორე დონის მენიუებს შორის ადგილი ყოფილიყო, ამისათვის სტილების ფაილში ვწერთ:

```
.menu li:hover ul {visibility:visible;
                   margin-left:200px;
                   padding-left:10px;
                   top:0;
                   border:2px solid #0F0;}
```

ანუ აქ padding-left:10px; დავამატეთ. ამის შემდეგ შეგვიძლია border წავშალოთ და შედეგად, ვიზუალურად პირველი და მეორე დონის მენიუები ერთმანეთისგან დაცილებული იქნება.

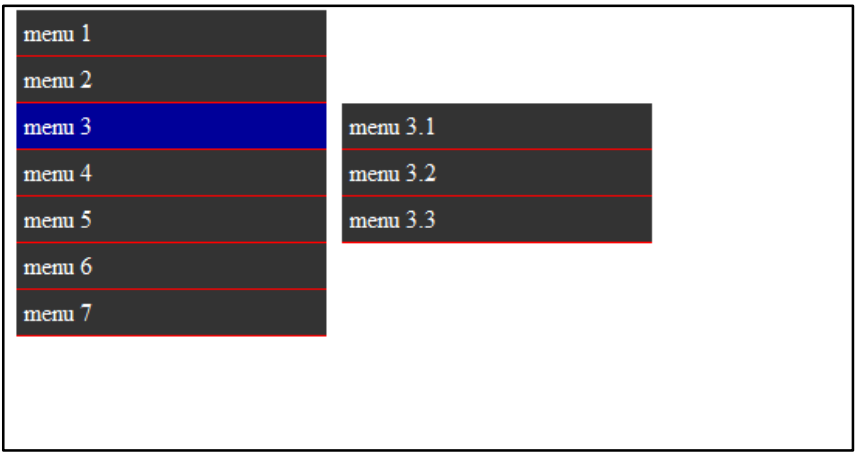


ამით ჩვენი ჩამოსაშლელი მენიუ უკვე აგებულია.

ბოლოს, შეგვიძლია მენიუზე მაუსის გადატარებისას პატარა ანიმაცია გავაკეთოთ, თუ სტილების ფაილში ჩავწერთ

```
.menu li a:hover
{
background-color:#009;
}
```

მენიუზე მაუსის გადატარებისას ფონის ფერი შეიცვლება.



საბოლოოდ, HTML და CSS ფაილებს ასეთი სახე ექნება:

Index. html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>menu</title>
<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
<ul class="menu">
<li><a href="#">menu 1 </a> </li>
<li><a href="#">menu 2 </a></li>
<li><a href="#">menu 3 </a>
<ul>
<li><a href="#">menu 3.1 </a> </li>
<li><a href="#">menu 3.2 </a> </li>
<li><a href="#">menu 3.3 </a> </li>
</ul>
</li>
<li><a href="#">menu 4 </a></li>
<li><a href="#">menu 5 </a>
<ul>
<li><a href="#">menu 5.1 </a> </li>
<li><a href="#">menu 5.2 </a> </li>
<li><a href="#">menu 5.3 </a> </li>
<li><a href="#">menu 5.4 </a> </li>
</ul>
</li>
```

```
<li><a href="#">menu 6 </a>
<ul>
<li><a href="#">menu 6.1 </a> </li>
<li><a href="#">menu 6.2 </a> </li>
</ul>
</li>
<li><a href="#">menu 7 </a></li>
</ul>
</body>
</html>
```

Style.css

```
@charset "utf-8";
/* CSS Document */
.menu          {
                 list-style:none;
                  margin:0;
                  padding:0; }
.menu li {
               width:200px;
              border-bottom:1px solid #F00;
              position:relative;
            }
.menu li a {
                text-decoration:none;
                background-color:#333;
                color:#FFF;
                padding: 5px;
                display:block;
}
```

```
.menu ul {
            list-style:none;
            margin:0;
            padding:0;
            visibility:hidden;
            position: absolute;}
.menu li:hover ul {
              visibility:visible;
              margin-left:200px;
              padding-left:10px;
              top:0;
               }
.menu li a:hover {
               background-color:#009;
                  }
```

ელემენტების სასურველ ადგილას განთავსება

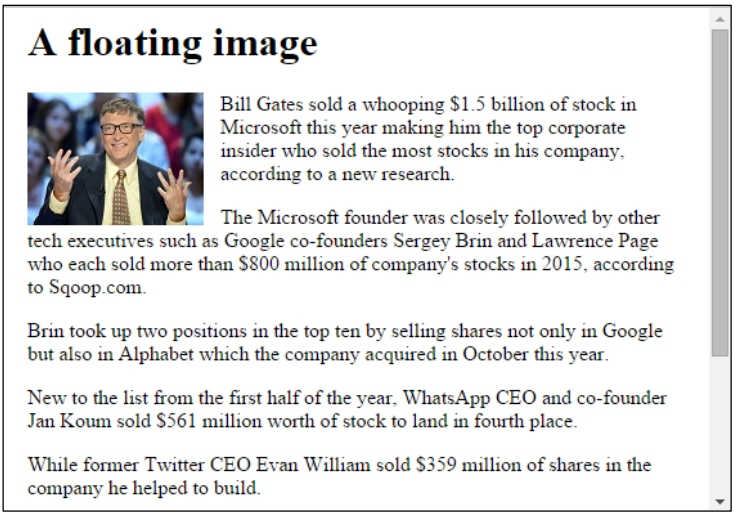
ელემენტების სასურველ ადგილას განსათავსებლად float თვისება გამოიყენება. დავუშვათ, გვინდა ვებგვერდზე სურათი მარცხნივ და ტექსტი მარჯვნივ განვათავსოთ. ასეთ შემთხვევაში, HTML კოდში გვექნება:

```
<div id="picture">
    <img src="bill.jpg" alt="Bill Gates">
    </div>
<p> text ...</p>
```

CSS კოდს კი შემდეგი სახე ექნება:

```
#picture {
```

```
float:left;
width: 150px;
}
```



float თვისება შეიძლება გამოვიყენოთ ასევე დოკუმენტის სვეტების სახით გამოსატანად.

HTML კოდი:

```
<div id="column1">
<p>ტექსტი...</p>
</div>
<div id="column2">
<p>ტექსტი...</p>
</div>
<div id="column3">
<p>ტექსტი...</p>
</div>
```

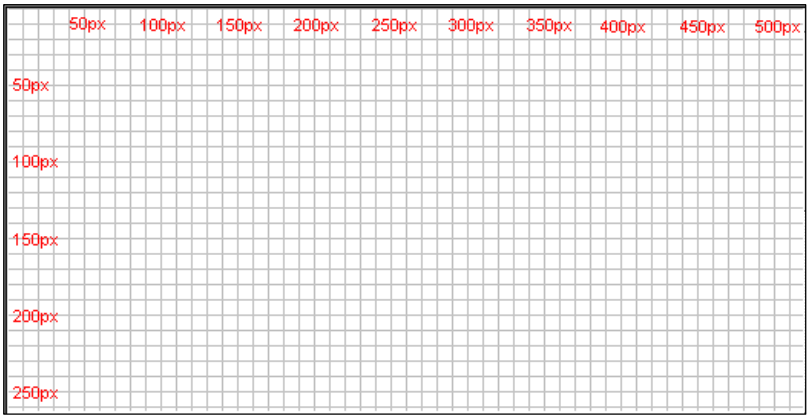
CSS კოდი:

```
#column1 {
    float:left;
    width: 33%;
    }
#column2 {
    float:left;
    width: 33%;
    }
#column3 {
    float:left;
    width: 33%;
    }
```

ელემენტების პოზიციონირება

CSS პოზიციონირების საშუალებით შესაძლებელია ელემენტის ზუსტად საჭირო ადგილას განთავსება.

ბრაუზერის ფანჯარა წარმოვადგინოთ როგორც კოორდინატთა სისტემა:

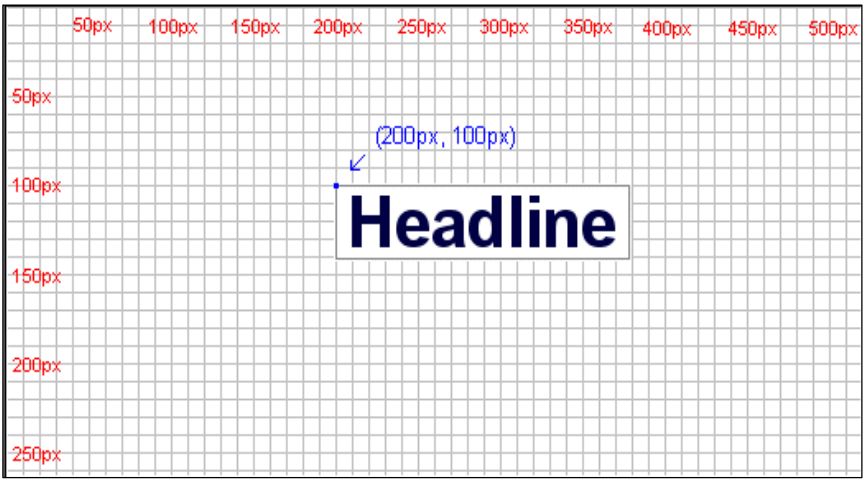


CSS პოზიციონირების პრინციპი იმაში მდგომარეობს, რომ კოორდინატთა სისტემაში ელემენტი შეგიძლიათ მოათავსოთ იქ, სადაც გნებავთ.

დავუშვათ, გვინდა სათაურის პოზიციონირება. თუ გვინდა, რომ ის დოკუმენტის ზედა საზღვრიდან 100 პიქსელზე და მარცხენა საზღვრიდან 200 პიქსელზე მოვათავსოთ, მაშინ შემდეგი სახის CSS კოდი უნდა დავწეროთ:

```
h1 {
    position:absolute;
    top: 100px;
    left: 200px;
    }
```

შედეგად მიიღება:



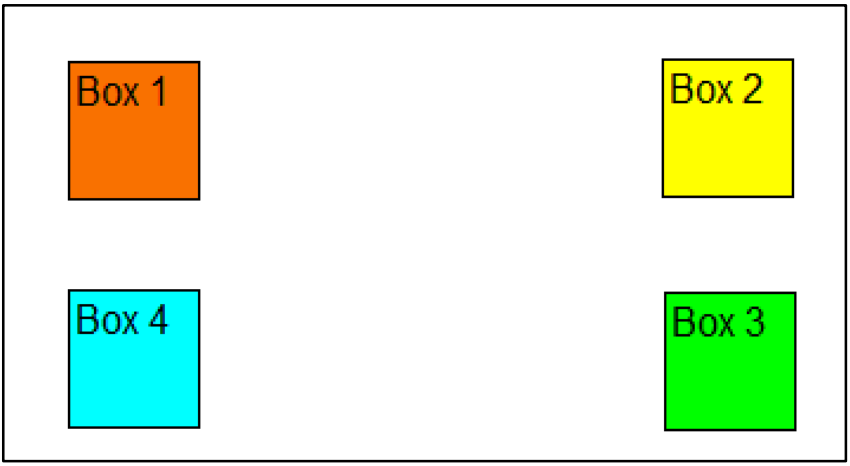
მაგალითისათვის დოკუმენტის ოთხივე კუთხეში აბსოლუტური პოზიციონირების გზით ოთხი ბლოკი მოვათავსოთ. ამისათვის შემდეგი სახის CSS კოდი უნდა დავწეროთ:

```
#box1 {
    position:absolute;
    top: 50px;
    left: 50px;
    }
#box2 {
    position:absolute;
    top: 50px;
    right: 50px;
    }
#box3 {
    position:absolute;
    bottom: 50px;
    right: 50px;
    }
#box4 {
    position:absolute;
    bottom: 50px;
    left: 50px;
    }
```

ხოლო HTML კოდს ექნება სახე:

```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>ელემენტების პოზიციონირება</title>
</head>
<body>
```

```
<link rel="stylesheet" type="text/css" href="style.css">
<div id="box1">
    <img src="box1.png" >
    </div>
<div id="box2">
    <img src="box2.png" >
    </div>
<div id="box3">
    <img src="box3.png" >
    </div>
<div id="box4">
    <img src="box4.png" >
    </div>
</body>
</html>
```



ფარდობითი პოზიციონირება

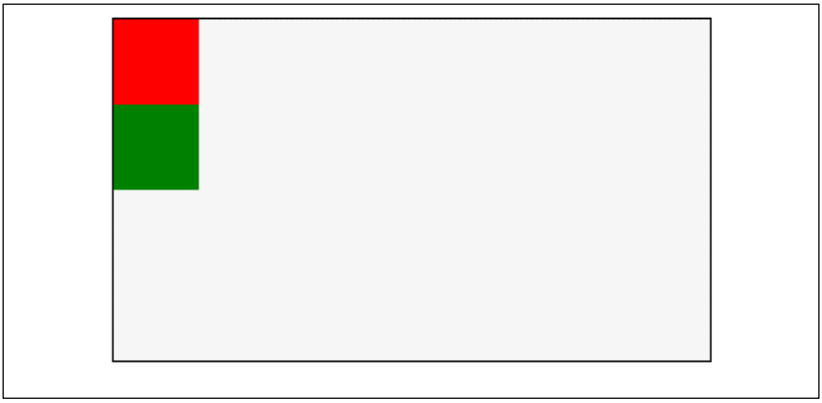
ელემენტის ფარდობითი პოზიციონირებისთვის position თვისებას relative მნიშვნელობა მივანიჭოთ. ფარდობითი პოზიციონირების დროს ელემენტის მდებარეობის შეცვლა მისი არსებული პოზიციის მიმართ (მის პოზიციასთან ფარდობით) ხდება. ეს ნიშნავს, რომ თქვენ ამ ელემენტის წანაცვლებას მარცხნიდან მარჯვნივ ან მარჯვნიდან მარცხნივ, ზემოდან ქვემოთ ან ქვემოდან ზემოთ ახდენთ.

ფარდობითი პოზიციონირების დემონსტრირებისთვის შემდეგი კოდი ჩავწეროთ:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Position</title>
<style>
    .home  {
            border: solid 1px black;
            width: 350px;
            height: 200px;
            margin: auto;
            background-color: #f6f6f6;
    }
    .position1  {
            width: 50px;
            height: 50px;
            background-color: #ff0000;
    }
    .position2  {
            width: 50px;
            height: 50px;
```

```
            background-color: #008000;
        }
    </style>
    </head>
    <body>
        <div class='home'>
            <div class='position1'></div>
            <div class='position2'></div>
        </div>
    </body>
    </html>
```

მიღებული შედეგი იქნება:

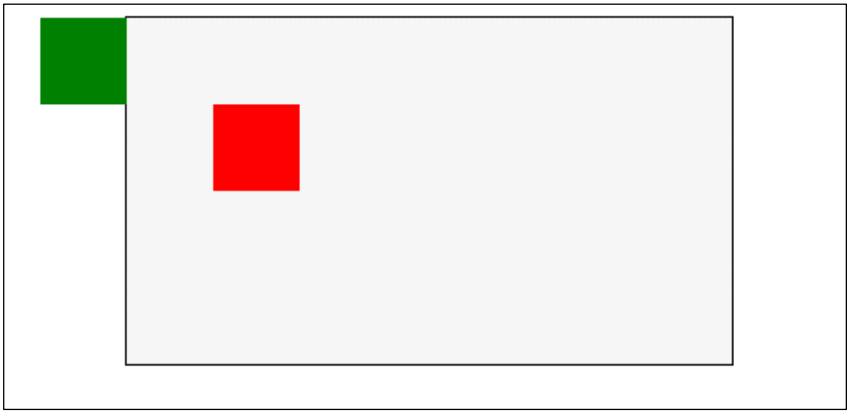


შემდეგ კოდში მოვახდინოთ ჩასწორება:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Position</title>
<style>
    .home {
```

```
            border: solid 1px black;
            width: 350px;
            height: 200px;
            margin: auto;
            background-color: #f6f6f6;      }
    .position1  {
            width: 50px;
            height: 50px;
            background-color: #ff0000;
            position: relative;
            top:50px;
            left:50px;          }
    .position2  {
            width: 50px;
            height: 50px;
            background-color: #008000;
            position: relative;
            right:50px;
            bottom:50px;    }
</style>
</head>
<body>
    <div class='home'>
            <div class='position1'></div>
            <div class='position2'></div>
    </div>
</body>
</html>
```

ჩასწორების შედეგად მივიღებთ:



კოდში შეტანილი ჩასწორებები შემდეგ ცვლილებებს გამოიწვევს:

- top:50px; გადაადგილებს ობიექტს საწყისი პოზიციიდან ზემოდან ქვემოთ 50 პიქსელით.
- bottom:50px; გადაადგილებს ობიექტს საწყისი პოზიციიდან ქვემოდან ზემოთ 50 პიქსელით.
- left:50px; გადაადგილებს ობიექტს საწყისი პოზიციიდან მარცხნიდან მარჯვნივ 50 პიქსელით.
- right:50px; გადაადგილებს ობიექტს საწყისი პოზიციიდან მარჯვნიდან მარცნივ 50 პიქსელით.

განვიხილოთ კიდევ ერთი მაგალითი:

```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>ფარდობითი პოზიციონირება</title>
```

```
<style>
    .div1{
            width:500px;
            height:350px;
            background-color:#CCC;
            margin:auto;}
    .div2{
            width:250px;
            height:150px;
            background-color:#F00;
            top:20px;;
            right:-30px;
            position:relative;}
    .div3{
            width:300px;
            height:250px;
            background-color:#00C;
            top:5px;
            right:30px;
            position:absolute;}
    .img1{
            border:2px solid #F00;
            position:relative;
            left:200px;}
</style>
</head>
<body>
<div class="div3">
```

Conventionally, a computer consists of at least one processing element, typically a central processing unit (CPU), and some form of

memory. The processing element carries out arithmetic and logic operations, and a sequencing and control unit can change the order of operations in response to stored information. Peripheral devices allow information to be retrieved from an external source, and the result of operations saved and retrieved.

A personal computer is a general-purpose computer whose size, capabilities and original sale price make it useful for individuals, and is intended to be operated directly by an end-user with no intervening computer operator. This contrasts with the batch processing or time-sharing models that allowed larger, more expensive minicomputer and mainframe systems to be used by many people, usually at the same time. A related term is "PC" that was initially an acronym for "personal computer", but later became used primarily to refer to the ubiquitous Wintel platform.

A computer is a general-purpose device that can be programmed to carry out a set of arithmetic or logical operations automatically. Since a sequence of operations can be readily changed, the computer can solve more than one kind of problem.

<p>

Software applications for most personal computers include, but are not limited to, word processing, spreadsheets, databases, web browsers and e-mail clients, digital media playback, games and myriad personal productivity and special-purpose software applications. Modern personal computers often have connections to the Internet, allowing access to the World Wide Web and a wide range of other resources. Personal computers may be connected to a local area network (LAN), either by a

cable or a wireless connection. A personal computer may be a desktop computer or a laptop, netbook, tablet or a handheld PC.

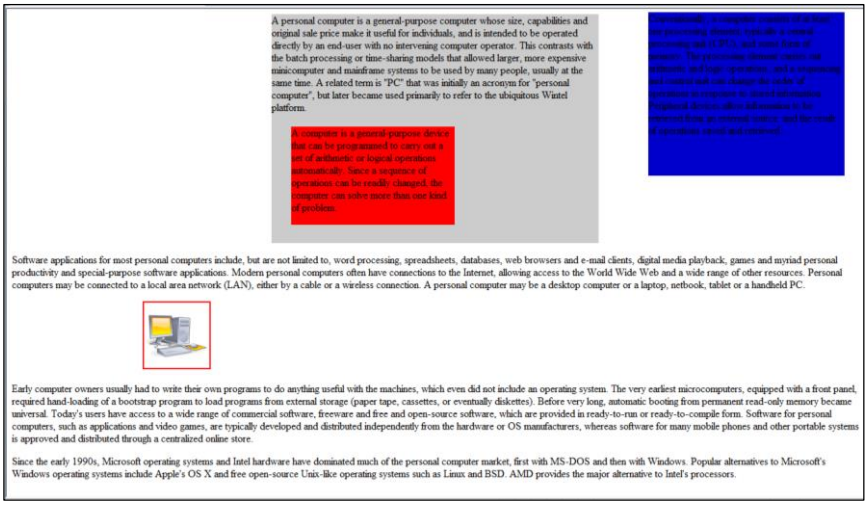
<img class="img1" src="Computer.png" width="100" height="100">

Early computer owners usually had to write their own programs to do anything useful with the machines, which even did not include an operating system. The very earliest microcomputers, equipped with a front panel, required hand-loading of a bootstrap program to load programs from external storage (paper tape, cassettes, or eventually diskettes). Before very long, automatic booting from permanent read-only memory became universal. Today's users have access to a wide range of commercial software, freeware and free and open-source software, which are provided in ready-to-run or ready-to-compile form. Software for personal computers, such as applications and video games, are typically developed and distributed independently from the hardware or OS manufacturers, whereas software for many mobile phones and other portable systems is approved and distributed through a centralized online store.

Since the early 1990s, Microsoft operating systems and Intel hardware have dominated much of the personal computer market, first with MS-DOS and then with Windows. Popular alternatives to Microsoft's Windows operating systems include Apple's OS X and free open-source Unix-like operating systems such as Linux and BSD. AMD provides the major alternative to Intel's processors.

</body>

</html>



ფარდობითი პოზიციონირებისას ახალი პოზიცია ორიგინალი გამოსახულების ზედა მარცხენა კუთხიდან აითვლება. ჩვენს შემთხვევაში, div1 ბლოკში div2 ბლოკია ჩაშენებული.

div2 ბლოკისათვის ჩანაწერი

```
top:20px;;
right:-30px;
position:relative;
```

div1 ბლოკის ზედა მარცხენა კუთხიდან 30 პიქსელით მარჯვნივ და 20 პიქსელით ქვემოთ წანაცვლებას უზრუნველყოფს. right:-30px; ჩანაწერი იგივეა, რაც left:30px.

ჩანაწერი

```
.img1{ border:2px solid #F00;
   position:relative;
   left:200px;}
```

აღნიშნავს, რომ გამოსახულება მისი პირვანდელი მდებარეობიდან 200 პიქსელით მარცხნიდან მარჯვნივაა წანაცვლებული.

ფენების შექმნა Z-ინდექსის საშუალებით

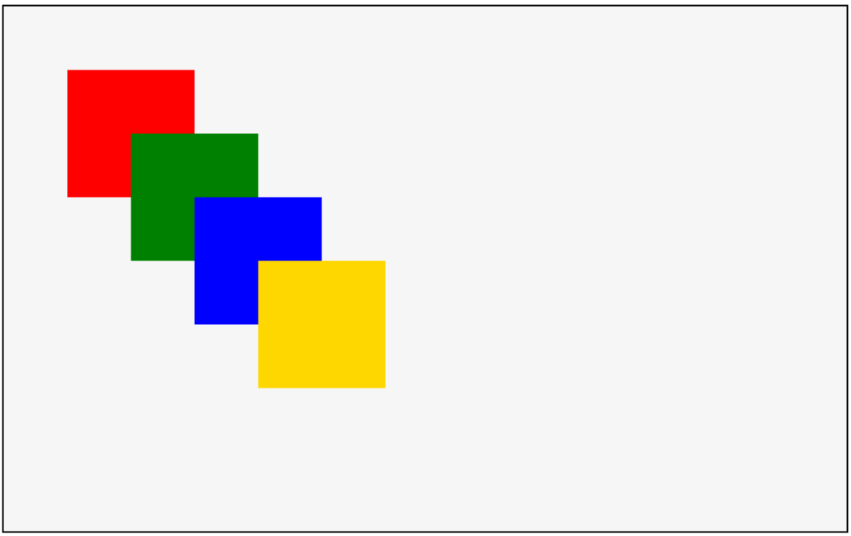
განვიხილოთ შრეების (layers) შექმნის საკითხი. ეს ნიშნავს, რომ ელემენტები ისე უნდა დავალაგოთ, რომ ერთმანეთი გადაფარონ. ამისათვის, თითოეულ ელემენტს უნდა მივანიჭოთ ნომერი (z-index). ელემენტი უფრო მეტი ნომრით ფარავს უფრო დაბალი ნომრის მქონე ელემენტს.

ჩავწეროთ შემდეგი კოდი:

```
<!DOCTYPE html>
<html>
<head>
    <title>Z-Index</title>
<style>
    .home {
            border: solid 1px black;
            width: 450px;
            height: 250px;
            margin: auto;
            padding: 40px;
            background-color: #f6f6f6;      }
    .zindex1 {
            width: 80px;
            height: 80px;
            background-color: #ff0000;
            position: relative;         }
    .zindex2 {
            width: 80px;
            height: 80px;
            background-color: #008000;
```

```
            position: relative;
            top:-40px;
            left: 40px;         }
    .zindex3 {
            width: 80px;
            height: 80px;
            background-color: #0000ff;
            position: relative;
            top:-80px;
            left:80px;          }
    .zindex4 {
            width: 80px;
            height: 80px;
            background-color: #ffd700;
            position: relative;
            top:-120px;
            left:120px;         }
</style>
</head>
<body>
    <div class='home'>
            <div class='zindex1'></div>
            <div class='zindex2'></div>
            <div class='zindex3'></div>
            <div class='zindex4'></div>
    </div>
</body>
</html>
```

შედეგად მიიღება:



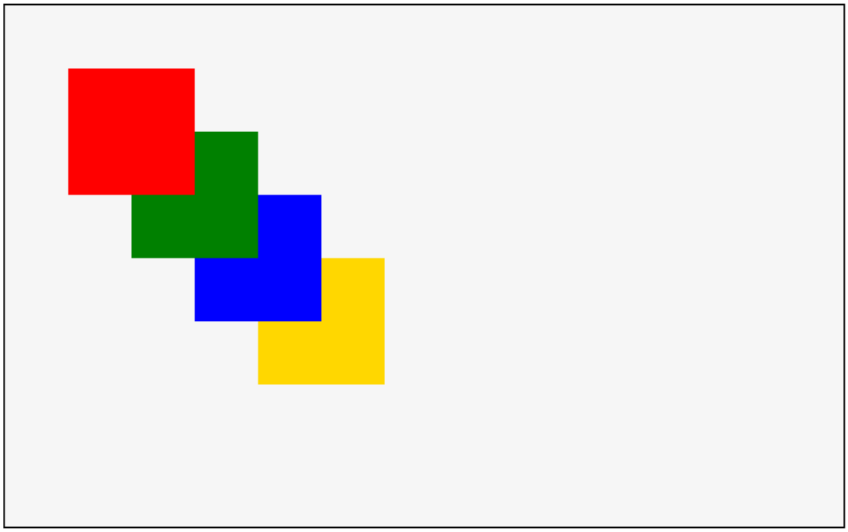
შემოვიტანოთ z-index თვისებები და კოდი შემდეგი სახით ჩავასწოროთ:

```
<!DOCTYPE html>
<html>
<head>
    <title>Z-Index</title>
<style>
    .home{
            border: solid 1px black;
            width: 450px;
            height: 250px;
            margin: auto;
            padding: 40px;
            background-color: #f6f6f6;
    }
```

```
.zindex1{
        width: 80px;
        height: 80px;
        background-color: #ff0000;
        position: relative;
        z-index: 4;
}
.zindex2{
        width: 80px;
        height: 80px;
        background-color: #008000;
        position: relative;
        top:-40px;
        left: 40px;
        z-index: 3;
}
.zindex3{
        width: 80px;
        height: 80px;
        background-color: #0000ff;
        position: relative;
        top:-80px;
        left:80px;
        z-index: 2;
}
.zindex4{
        width: 80px;
        height: 80px;
        background-color: #ffd700;
        position: relative;
```

```
                top:-120px;
                left:120px;
                z-index: 1;
    }
</style>
</head>
<body>
    <div class='home'>
                <div class='zindex1'></div>
                <div class='zindex2'></div>
                <div class='zindex3'></div>
                <div class='zindex4'></div>
    </div>
</body>
</html>
```

ჩასწორების შემდეგ ფენები შემდეგნაირად დალაგდება:



CSS3 ანიმაცია

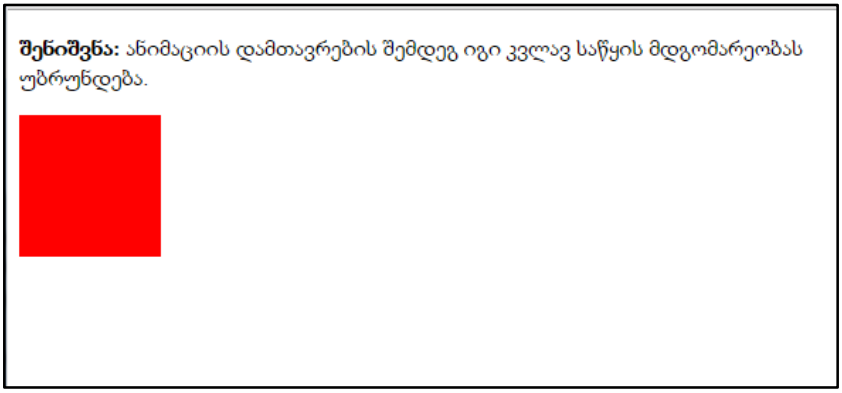
CSS3-ში ანიმაციის შესაქმნელად @keyframes თვისება გამოიყენება. ეს თვისება კონტეინერს წარმოადგენს, რომელშიც დოკუმენტის გაფორმებისათვის საჭირო სხვადასხვა თვისება უნდა მოთავსდეს.

მას შემდეგ, რაც ანიმაცია შეიქმნება, აუცილებელია შეიქმნას ელემენტი, რომლის ანიმაციაც გვინდა. ამ ელემენტისთვის უნდა მივუთითოთ ანიმაციის სახელი (animation-name) და დრო (animation-duration), რომლის განმავლობაში ეს ანიმაცია შესრულდება.

ქვემოთ მოცემული კოდი საშუალებას იძლევა ბლოკის ფონის ფერი მასში მითითებული მნიშვნელობების მიხედვით შეიცვალოს:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
    width: 100px;
    height: 100px;
    background-color: red;
    -webkit-animation-name: example; /* Chrome, Safari, Opera */
    -webkit-animation-duration: 4s; /* Chrome, Safari, Opera */
    animation-name: example;
    animation-duration: 4s;
}
/* Chrome, Safari, Opera */
@-webkit-keyframes example {
    0%  {background-color: red;
```

```
    }
      25%  {background-color: yellow;}
      50%  {background-color: blue;}
      100% {background-color: green;}
    }
    /* სტანდარტული სინტაქსი */
    @keyframes example
    {
      0%   {background-color: red;}
      25%  {background-color: yellow;}
      50%  {background-color: blue;}
      100% {background-color: green;}
    }
    </style>
    </head>
    <body>
    <p>
    <b>შენიშვნა:
    </b>
    ანიმაციის დამთავრების შემდეგ იგი კვლავ საწყის
მდგომარეობას უბრუნდება.
    </p>
    <div></div>
    </body>
    </html>
```



ასევე, შესაძლებელია ელემენტისთვის შეიცვალოს როგორც ფონის ფერი, ასევე მისი მდებარეობა:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
    width: 100px;
    height: 100px;
    background-color: red;
    position: relative;
    -webkit-animation-name: example; /* Chrome, Safari, Opera */
    -webkit-animation-duration: 4s; /* Chrome, Safari, Opera */
    animation-name: example;
    animation-duration: 4s;
}
/* Chrome, Safari, Opera */
@-webkit-keyframes example {
    0%   {background-color:red; left:0px; top:0px;}
    25%  {background-color:yellow; left:200px; top:0px;}
```

```
    50%  {background-color:blue; left:200px; top:200px;}
    75%  {background-color:green; left:0px; top:200px;}
    100% {background-color:red; left:0px; top:0px;}
}
/* სტანდარტული სინტაქსი */
@keyframes example {
    0%   {background-color:red; left:0px; top:0px;}
    25%  {background-color:yellow; left:200px; top:0px;}
    50%  {background-color:blue; left:200px; top:200px;}
    75%  {background-color:green; left:0px; top:200px;}
    100% {background-color:red; left:0px; top:0px;}
}
</style>
</head>
<body>
<div></div>
</body>
</html>
```

დაყოვნება ანიმაციაში

animation-delay თვისება ანიმაციის მითითებული დროით დაყოვნებას იწვევს. თუ, მაგალითად, გვსურს ანიმაციის დასაწყისი 2 წამით შევაყოვნოთ, მაშინ მას ასეთი სახე ექნება:

```
div {
    width: 100px;
    height: 100px;
    position: relative;
    background-color: red;
    animation-name: example;
```

```
    animation-duration: 4s;
    animation-delay: 2s;
}
```

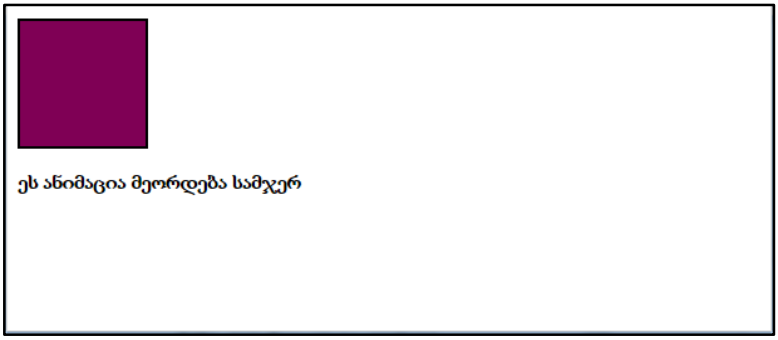
animation-iteration-count თვისება განსაზღვრავს, თუ რამდენჯერ უნდა შესრულდეს ანიმაცია. შემდეგ მაგალითში ანიმაცია იმუშავებს 3-ჯერ და შემდეგ გაჩერდება. თუ ამ თვისებაში რიცხვის მაგივრად მითითებული იქნება „infinite“, მაშინ ანიმაცია უსასრულოდ გამეორდება.

```
@keyframes anim {
from {margin-left: 3px;}
to {margin-left: 500px;}
}
#wrap1 {
Animation-name: anim;
animation-duration: 4s;
animation-iteration-count: 3;
}
```

განვიხილოთ შემდეგი კოდი:

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-
8">
<title>ანიმაცია</title>
<style>
@keyframes anim
{
from {margin-left: 3px;}
```

```
to {margin-left: 500px;}
}
@-moz-keyframes anim{
from {margin-left: 3px;}
to {margin-left: 500px;}
}
@-webkit-keyframes anim{
from {margin-left: 3px;}
to {margin-left: 500px;}
}
#wrap1{
border:2px #000 solid;
background-color:#7F0055;
height:100px;
width:100px;
animation-name: anim;
animation-duration: 4s;
animation-iteration-count: 3;
-webkit-animation-name: anim; /* Chrome, Safari, Opera */
-webkit-animation-duration: 4s; /* Chrome, Safari, Opera */
-webkit-animation-iteration-count: 3; /* Chrome, Safari, Opera */
}
</style>
</head>
<body>
<div id="wrap1"></div>
<p><b>ეს ანიმაცია მეორდება სამჯერ</b></p>
</body>
</html>
```



სურათზე გამოსახული მართკუთხედი ბრაუზერის ეკრანს 4 წმ-ის განმავლობაში სამჯერ გაივლის.

ანიმაციის შესრულების მიმდინარეობა, არა მხოლოდ from და to სიტყვების მეშვეობით, არამედ %-ითაც შეიძლება განისაზღვროს.

%-ის მეშვეობით შეგიძლიათ ანიმაციის მიმდინარეობა უფრო ზუსტად გააკონტროლოთ, მაგალითად, შეგიძლიათ მიუთითოთ, რომ ანიმაციის დასაწყისში (0%) განსაზღვრული ელემენტი უნდა იყოს თეთრი, შუაში (50%) უნდა შეიფეროს ნარინჯისფრად და ბოლოს (100%) გახდეს შავი.

ქვემოთ შესაბამისი კოდია ნაჩვენები:

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>ანიმაცია - 2</title>
<style>
@keyframes anim {
0% {margin-left:3px;margin-top:3px;background-color:# f09900;}
30% {margin-left:3px;margin-top:250px;background-color:#ffc608;}
```

```
60% {margin-left:500px;margin-top:250px;background-color:black;}
100% {margin-left:3px;margin-top:3px;background-color:# f09900;}
}
@-moz-keyframes anim {
0% {margin-left:3px;margin-top:3px;background-color:# f09900;}
30%{margin-left:3px;margin-top:250px;background-color:# ffc608;}
60% {margin-left:500px;margin-top:250px;background-color:black;}
100% {margin-left:3px;margin-top:3px;background-color:# f09900;}
}
@-webkit-keyframes anim {
0% {margin-left:3px;margin-top:3px;background-color:# f09900;}
30%{margin-left:3px;margin-top:250px;background-color:# ffc608;}
60% {margin-left:500px;margin-top:250px;background-color:black;}
100% {margin-left:3px;margin-top:3px;background-color:# f09900;}
}
#wrap1 {
border:2px #000 solid;
background-color:# f09900;
height:100px;
width:100px;
animation:anim 6s 3;
-webkit-animation:anim 6s 3;
}
</style>
</head>
<body>
<div id="wrap1"></div>
</body>
</html>
```

ანიმაციის შესრულების მიმართულება

ანიმაციის შესრულების მიმართულებას animation-direction თვისება განსაზღვრავს.

მაგალითი:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
    width: 100px;
    height: 100px;
    background-color: red;
    position: relative;
    -webkit-animation-name: example; /* Chrome, Safari, Opera */
    -webkit-animation-duration: 4s; /* Chrome, Safari, Opera */
    -webkit-animation-iteration-count: 3; /* Chrome, Safari, Opera */
    -webkit-animation-direction: reverse; /* Chrome, Safari, Opera */
    animation-name: example;
    animation-duration: 4s;
    animation-iteration-count: 3;
    animation-direction: reverse;
}
/* Chrome, Safari, Opera */
@-webkit-keyframes example {
    0%   {background-color:red; left:0px; top:0px;}
    25%  {background-color:yellow; left:200px; top:0px;}
    50%  {background-color:blue; left:200px; top:200px;}
    75%  {background-color:green; left:0px; top:200px;}
```

```
    100% {background-color:red; left:0px; top:0px;}
}
/* სტანდარტული სინტაქსი */
@keyframes example {
    0%   {background-color:red; left:0px; top:0px;}
    25%  {background-color:yellow; left:200px; top:0px;}
    50%  {background-color:blue; left:200px; top:200px;}
    75%  {background-color:green; left:0px; top:200px;}
    100% {background-color:red; left:0px; top:0px;}
}
</style>
</head>
<body>
<div></div>
</body>
</html>
```

შემდეგ მაგალითში გამოყენებულია თვისება alternate, რომლის მიხედვითაც ანიმაცია ჯერ ერთი მიმართულებით მოძრაობს, შემდეგ - უკუმიმართულებით, შემდეგ ისევ საწყისი მიმართულებით და ა. შ.

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
    width: 100px;
    height: 100px;
    background-color: red;
    position: relative;
```

```
        -webkit-animation-name: example; /* Chrome, Safari, Opera */
        -webkit-animation-duration: 4s; /* Chrome, Safari, Opera */
        -webkit-animation-iteration-count: 3; /* Chrome, Safari, Opera */
        -webkit-animation-direction: alternate;    /* Chrome, Safari,
Opera */
        animation-name: example;
        animation-duration: 4s;
        animation-iteration-count: 4;
        animation-direction: alternate;    }
    /* Chrome, Safari, Opera */
    @-webkit-keyframes example {
        0%   {background-color:red; left:0px; top:0px;}
        25%  {background-color:yellow; left:200px; top:0px;}
        50%  {background-color:blue; left:200px; top:200px;}
        75%  {background-color:green; left:0px; top:200px;}
        100% {background-color:red; left:0px; top:0px;}
    }
    /* სტანდარტული სინტაქსი */
    @keyframes example {
        0%   {background-color:red; left:0px; top:0px;}
        25%  {background-color:yellow; left:200px; top:0px;}
        50%  {background-color:blue; left:200px; top:200px;}
        75%  {background-color:green; left:0px; top:200px;}
        100% {background-color:red; left:0px; top:0px;} }
    </style>
    </head>
    <body>
    <div></div>
    </body>
    </html>
```

ანიმაციის შესრულების სიჩქარე

animation-timing-function თვისება ანიმაციის შესრულების სიჩქარეს განსაზღვრავს.

animation-timing-function თვისებას შემდეგი მნიშვნელობების მიღება შეუძლია:

- ease - ანიმაციის ეს ეფექტი განსაზღვრავს მის შესრულებას ნელი სტარტიდან უფრო სწრაფ, საბოლოოდ ისევ ნელ მოქმედებაზე (ეს მნიშვნელობა აიღება ჩუმათობის პრინციპით);
- linear - ანიმაციის ეს ეფექტი განსაზღვრავს მის შესრულებას ერთი და იგივე სიჩქარით თავიდან ბოლომდე;
- ease-in - ანიმაციის ეს ეფექტი განსაზღვრავს მის შესრულებას ნელი სტარტით;
- ease-out - ანიმაციის ეს ეფექტი განსაზღვრავს მის შესრულებას ნელი დასასრულით;
- ease-in-out - ანიმაციის ეს ეფექტი განსაზღვრავს მის შესრულებას ნელი სტარტით და დასასრულით;

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
    width: 100px;
    height: 50px;
    background-color: red;
    font-weight: bold;
    position: relative;
    -webkit-animation: mymove 5s infinite;   /* Chrome, Safari,
Opera */
```

```
    animation: mymove 5s infinite;
}
/* Chrome, Safari, Opera */
#div1 {-webkit-animation-timing-function: linear;}
#div2 {-webkit-animation-timing-function: ease;}
#div3 {-webkit-animation-timing-function: ease-in;}
#div4 {-webkit-animation-timing-function: ease-out;}
#div5 {-webkit-animation-timing-function: ease-in-out;}
/* სტანდარტული სინტაქსი */
#div1 {animation-timing-function: linear;}
#div2 {animation-timing-function: ease;}
#div3 {animation-timing-function: ease-in;}
#div4 {animation-timing-function: ease-out;}
#div5 {animation-timing-function: ease-in-out;}
/* Chrome, Safari, Opera */
@-webkit-keyframes mymove {
    from {left: 0px;}
    to {left: 300px;}
}
/* სტანდარტული სინტაქსი */
@keyframes mymove {
    from {left: 0px;}
    to {left: 300px;}
}
</style>
</head>
<body>
<div id="div1">linear</div>
<div id="div2">ease</div>
<div id="div3">ease-in</div>
```

```
<div id="div4">ease-out</div>
<div id="div5">ease-in-out</div>
</body>
</html>
```

ანიმაციის შესრულების შემოკლებული ჩანაწერი

დავუშვათ, ანიმაციის შესრულების დროს შემდეგი 6 თვისების გათვალისწინება გვსურს:

```
div {
   animation-name: example;
   animation-duration: 5s;
   animation-timing-function: linear;
   animation-delay: 2s;
   animation-iteration-count: infinite;
   animation-direction: alternate;
}
```

ანიმაციის იგივე ეფექტის შესრულება შემდეგი შემოკლებული ჩანაწერითაც მიიღწევა:

```
div {
   animation: example 5s linear 2s infinite alternate;
}
```

CSS ტრანსფორმაციები

transform თვისება ყველა თანამედროვე ბრაუზერში მუშაობს, მაგრამ ზოგიერთი ბრაუზერისთვის სპეციალური პრეფიქსების დამატებაა საჭირო. CSS-ში 2D და 3D ტრანსფორმაციების განხორციელებაა შესაძლებელი.

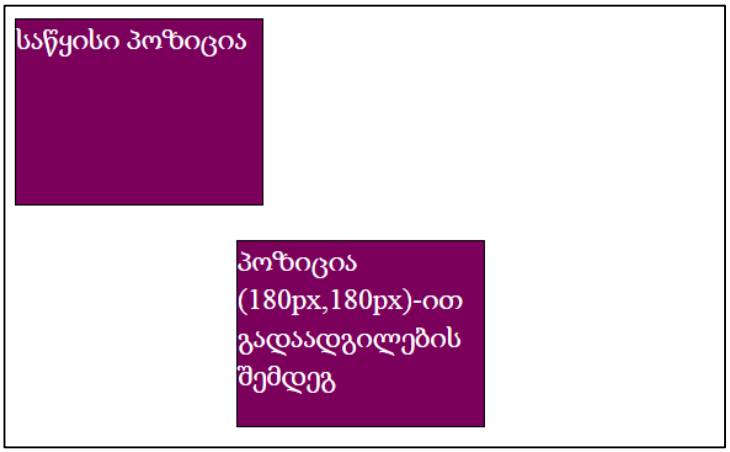
2D ტრანსფორმაციები

translate(x,y) ფუნქციის მეშვეობით შეგიძლიათ ელემენტი პიქსელების მითითებული რაოდენობით ჰორიზონტალსა და ვერტიკალზე გადაანაცვლოთ.

```
<!DOCTYPE html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>CSS გარდაქმნები</title>
<style>
#el1,#el2 {
position:absolute;
top:10px;
left:10px;
background-color:#7A005C;
color:white;
width:200px;
height:150px;
font-size:1.5em;
border:1px #000 solid;
}
#el2  {
transform: translate(180px,180px);
```

```
    -webkit-transform: translate(180px,180px); /* Chrome-ისა და
Safari-ისათვის */
    -ms-transform: translate(180px,180px); /* IE-ისათვის */
    }
    </style>
    </head>
    <body>
    <div id='el1'>
    საწყისი პოზიცია
    </div>
    <div id='el2'>პოზიცია (180px,180px)-ით გადაადგილების
შემდეგ</div>
    </body>
    </html>
```

შედეგი:

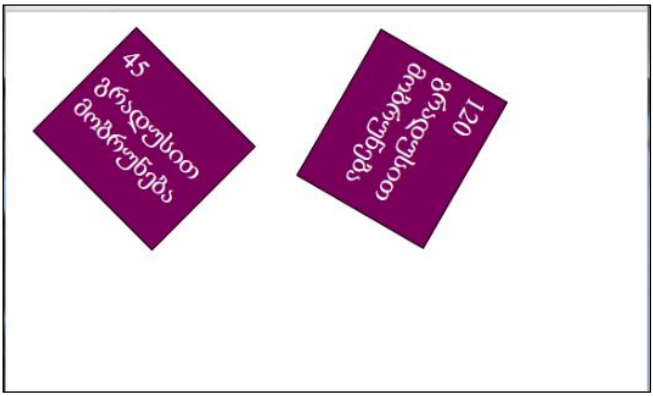


rotate ფუნქციის დახმარებით შეგიძლიათ მოაბრუნოთ ელემენტი გრადუსების მითითებული რაოდენობით საათის ისრის მოძრაობის მიმართულებით.

```
<!DOCTYPE html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>CSS გარდაქმნები</title>
<style>
#el3,#el4 {
margin:40px;
background-color:#7A005C;
width:130px;
height:110px;
font-size:1.5em;
border:1px #000 solid;
color:white;
padding:10px;
float:left;
}
#el3 {
transform: rotate(45deg);
-webkit-transform: rotate(45deg); /* Chrome-ისა და Safari-ისათვის */
-ms-transform: rotate(45deg); /* IE-ისათვის */
}
#el4 {
transform: rotate(120deg);
-webkit-transform: rotate(120deg); /* Chrome-ისა და Safari-ისათვის */
-ms-transform: rotate(120deg); /* IE-ისათვის */}
</style>
</head>
```

```
<body>
<div id='el3'>45 გრადუსით მობრუნება</div>
<div id='el4'>120 გრადუსით მობრუნება </div>
</body>
</html>
```

შედეგი:



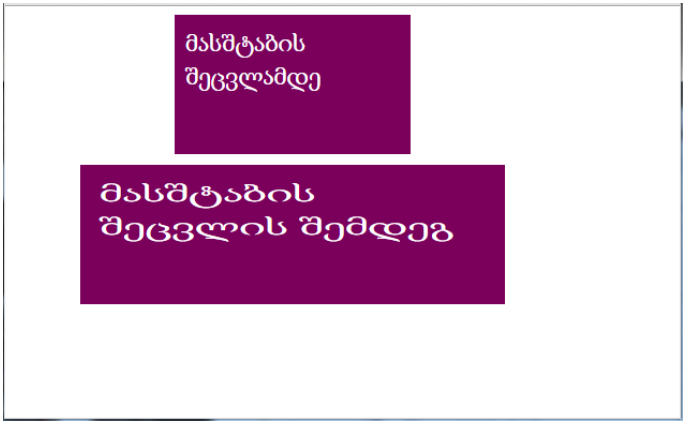
scale (x, y) მეთოდის საშუალებით ელემენტი შეიძლება გაიწელოს სიგანესა და სიმაღლეში.

მაგალითი:

```
<!DOCTYPE html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>CSS გარდაქმნები</title>
<style>
#el5,#el6 {
margin-left:150px;
margin-bottom:10px;
background-color:#7A005C;
```

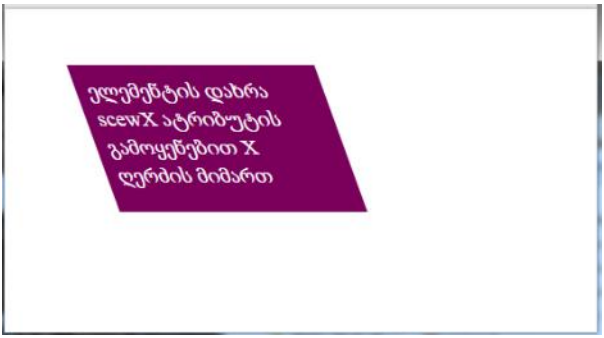
```
    width:200px;
    height:110px;
    font-size:1.5em;
    color:white;
    padding:10px;
    }
    #el6 {
    transform:scale(1.8,1);
    -webkit-transform:scale(1.8,1); /* Chrome-ისა და Safari-ისათვის
*/
    -ms-transform:scale(1.8,1);     } /* IE-ისათვის */
    </style>
    </head>
    <body>
    <div id='el5'>მასშტაბის შეცვლამდე </div>
    <div id='el6'>მასშტაბის შეცვლის შემდეგ </div>
    </body>
    </html>
```

შედეგი ბრაუზერის ფანჯარაში:



skewX(x) მეთოდის მეშვეობით შეგიძლიათ დახაროთ ელემენტი გრადუსების მითითებული რაოდენობით X ღერძის მიმართ.

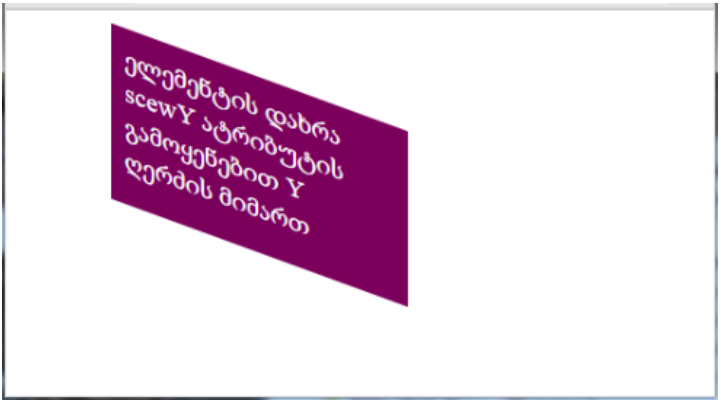
```
<!DOCTYPE html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>CSS გარდაქმნები</title>
<style>
#el8 {
margin-left:70px;
margin-top:50px;
background-color:#7A005C;
width:200px;
height:110px;
font-size:1.2em;
color:white;
padding:10px;
transform:skewX(20deg);
-webkit-transform:skewX(20deg); /* Chrome-ისა და Safari-ისათვის */
-ms-transform:skewX(20deg); } /* IE-ისათვის */
</style>
</head>
<body>
<div id='el8'>ელემენტის დახრა scewX ატრიბუტის გამოყენებით X ღერძის მიმართ </div>
</body>
</html>
```



skewY(x) მეთოდის მეშვეობით შეგიძლიათ დახაროთ ელემენტი გრადუსების მითითებული რაოდენობით Y ღერძის მიმართ.

```
<!DOCTYPE html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>CSS გარდაქმნები</title>
<style>
#el9 {
margin-left:70px;
margin-top:50px;
background-color:#7A005C;
width:200px;
height:110px;
font-size:1.2em;
color:white;
padding:10px;
transform:skewY(20deg);
-webkit-transform:skewY(20deg); /* Chrome-ისა და Safari-ისათვის */
```

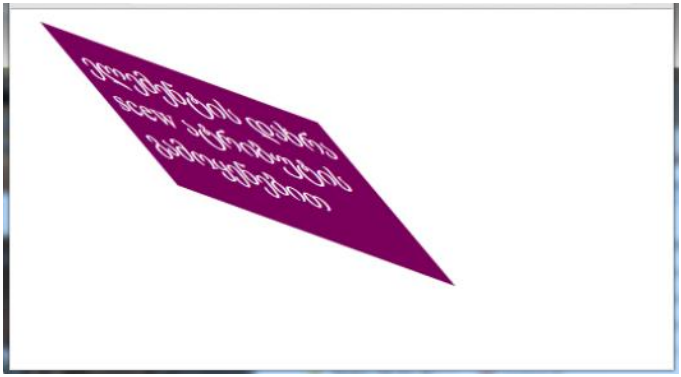
```
-ms-transform:skewY(20deg);     } /* IE-ისათვის */
</style>
</head>
<body>
<div id='el9'>ელემენტის დახრა scewY ატრიბუტის გამოყენებით Y ღერძის მიმართ </div>
</body>
</html>
```



skew(x,y) მეთოდის მეშვეობით შეგიძლიათ დახაროთ ელემენტი გრადუსების მითითებული რაოდენობით X და Y ღერძის მიმართ ერთდროულად.

```
<!DOCTYPE html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>CSS გარდაქმნები</title>
<style>
#el10 {
margin-left:70px;
```

```
    margin-top:50px;
    background-color:#7A005C;
    width:200px;
    height:110px;
    font-size:1.5em;
    color:white;
    padding:10px;
    transform:skew(40deg,20deg);
    -webkit-transform:skew(40deg,20deg); /* Chrome-ისა და Safari-ისათვის */
    -ms-transform:skew(40deg,20deg);    } /* IE-ისათვის */
    </style>
    </head>
    <body>
    <div id='el10'>ელემენტის დახრა scew ატრიბუტის გამოყენებით </div>
    </body>
    </html>
```

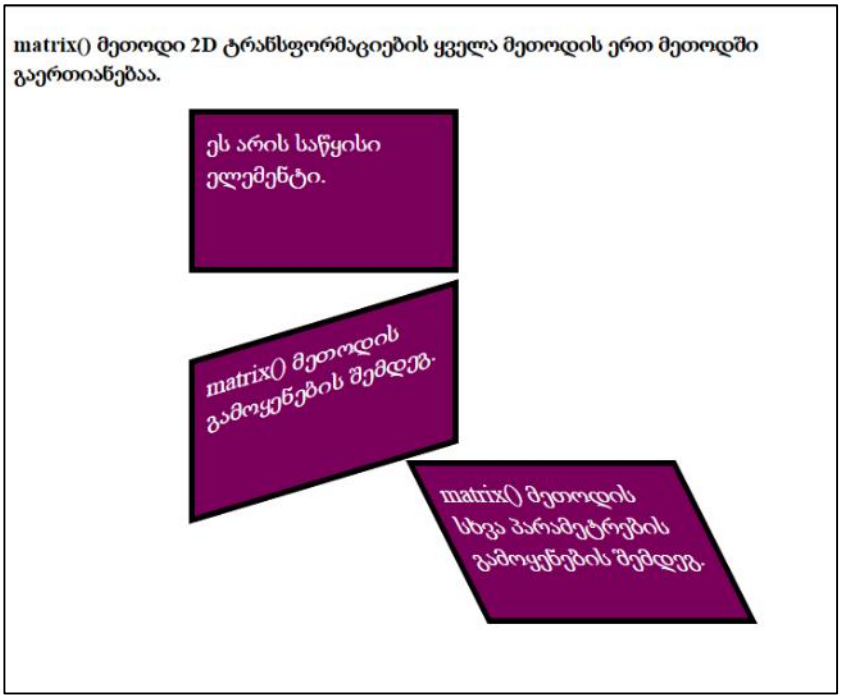


matrix() მეთოდი 2D ტრანსფორმაციების ყველა მეთოდის კომბინაციაა. მას ექვსი პარამეტრი აქვს. პირველი პარამეტრი არის

ჰორიზონტალურად გაზრდის კოეფიციენტი, მეორე - X ღერძის მიმართ დახრის კოეფიციენტი, მესამე - Y ღერძის მიმართ დახრის კოეფიციენტი, მეოთხე – ვერტიკალურად გაზრდის კოეფიციენტი, მეხუთე - პიქსელების მითითებული რაოდენობით გადაადგილება X ღერძის მიმართ, ხოლო მეექვსე - გადაადგილება Y ღერძის მიმართ.

```
<!DOCTYPE html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>CSS გარდაქმნები</title>
<style>
div
 {
margin-left:150px;
margin-bottom:10px;
background-color:#7A005C;
border: 5px solid black;
width:200px;
height:110px;
font-size:1.3em;
color:white;
padding:10px;
}
div#myDiv1
 {
   -webkit-transform: matrix(1, -0.3, 0, 1, 0, 30); /* Chrome-ისა და Safari-ისათვის */
   -ms-transform: matrix(1, -0.3, 0, 1, 0, 30); /* IE-ისათვის */
```

```
    }
    div#myDiv2 {
       -webkit-transform: matrix(1, 0, 0.5, 1, 220, 0); /* Chrome-ისა და
Safari-ისათვის */
         -ms-transform: matrix(1, 0, 0.5, 1, 220, 0); /* IE-ისათვის */
     }
    </style>
    </head>
    <body>
    <h3>
    matrix() მეთოდი 2D ტრანსფორმაციების ყველა მეთოდის ერთ
მეთოდში გაერთიანებაა.
     </h3>
    <div>
    ეს არის საწყისი ელემენტი.
    </div>
    <div id="myDiv1">
    matrix() მეთოდის გამოყენების შემდეგ.
    </div>
    <div id="myDiv2">
    matrix() მეთოდის სხვა პარამეტრების გამოყენების შემდეგ.
    </div>
    </body>
    </html>
```



3D ტრანსფორმაციები

rotateX () მეთოდის საშუალებით ხდება ელემენტის მობრუნება მისი X ღერძის გარშემო მოცემული კუთხით.

მაგალითი:

```
<!DOCTYPE html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>CSS transform</title>
<style>
div{
```

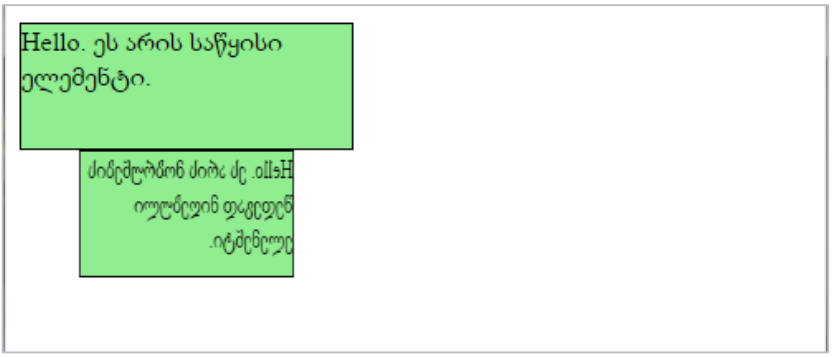
```
    width:100px;
    height:175px;
    background-color:lightgreen;
    border:1px solid black;}
    div#div2{
    transform: rotateX(120deg);
    -webkit-transform: rotateX(120deg); /* Chrome-ისა და Safari-ისათვის */ }
    </style>
    </head>
    <body>
    <div>Hello. ეს არის საწყისი ელემენტი.</div>
    <div id="div2">Hello. ეს არის მობრუნების შედეგად მიღებული ელემენტი.
    </div>
    </body>
    </html>
```

ელემენტის 120^0 მობრუნების შედეგად მიიღება:

rotateY () მეთოდის საშუალებით ხდება ელემენტის მობრუნება მისი Y ღერძის გარშემო მოცემული კუთხით.

```
<!DOCTYPE html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>CSS transform</title>
<style>
div {
width:200px;
height:75px;
background-color:lightgreen;
border:1px solid black; }
div#div2
{
transform:rotateY(130deg);
-webkit-transform:rotateY(130deg); /* Chrome-ისა და Safari-ისათვის */
}
</style>
</head>
<body>
<div>Hello. ეს არის საწყისი ელემენტი.</div>
<div id="div2">Hello. ეს არის მობრუნების შედეგად მიღებული ელემენტი.
</div>
</body>
</html>
```

Y ღერძის მიმართ 130⁰ ელემენტის მობრუნების შედეგი ბრაუზერის ფანჯარაში ასე აისახება:

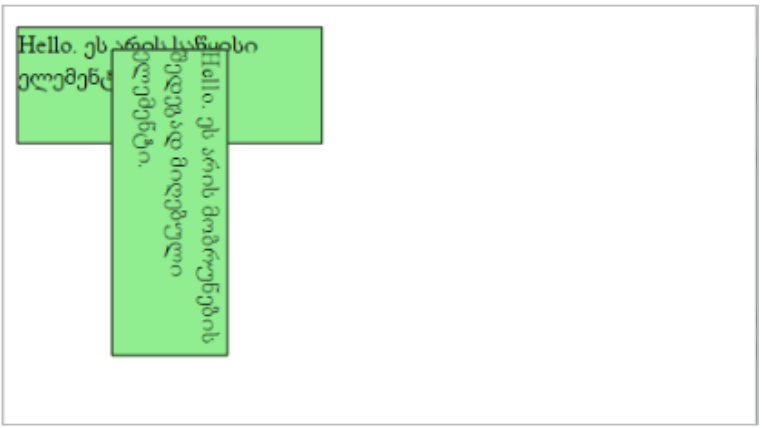


rotateZ () მეთოდის საშუალებით ხდება ელემენტის მობრუნება მისი Z ღერძის გარშემო მოცემული კუთხით.

```
<!DOCTYPE html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>CSS transform</title>
<style>
div  {width:200px;
      height:75px;
      background-color:lightgreen;
      border:1px solid black; }
div#div2  {
transform:rotateZ(90deg);
-webkit-transform:rotateZ(90deg); /* Chrome-ისა და Safari-ისათვის */}
</style>
</head>
```

```
<body>
<div>Hello. ეს არის საწყისი ელემენტი.</div>
<div id="div2">Hello. ეს არის მობრუნების შედეგად მიღებული ელემენტი.
</div>
</body>
</html>
```

Z ღერძის გარშემო 90^0 მობრუნებით მივიღებთ:



CSS3 Media Queries (მედიამოთხოვნები)

CSS2-ში შემოტანილ იქნა მედიამოთხოვნები.

@media წესებმა, რომელიც CSS2-ში იყო შემოღებული, სხვადასხვა მედიატიპის სტილისთვის სხვადასხვა წესის განსაზღვრის საშუალება მოგვცა.

მაგალითად, შეიძლება სტილების ერთი ნაკრები გვქონოდა კომპიუტერის ეკრანისთვის, პლანშეტისთვის, მობილურისთვის და ა.შ.

გავეცნოთ მედიამოთხოვნებს CSS3-ში.

მედიამოთხოვნები CSS3-ში CSS2-ის მედიატიპების იდეას ეფუძნება: ნაცვლად იმისა, რომ ეძებოს მოწყობილობის ტიპი, ისინი მოწყობილობის შესაძლებლობებს ითვალისწინებს.

მედიამოთხოვნები სხვადასხვა მახასიათებლის შესამოწმებლად შეიძლება იქნეს გამოყენებული, როგორიცაა:

- დათვალიერების ფანჯრის სიგანე და სიმაღლე;
- მოწყობილობის სიგანე და სიმაღლე;
- ორიენტაცია (გამოიყენება ძირითადად პლანშეტებისა და მობილურებისათვის);
- გარჩევადობა.

მედიამოთხოვნების გამოყენება ძალზე პოპულარული მეთოდია პლანშეტებში, iPhone-სა და Androids-ში სტილების ცხრილების მისაწოდებლად.

მედიამოთხოვნები შედგება მედიატიპისაგან და შეიძლება ერთ ან რამდენიმე გამოსახულებას შეიცავდეს, რომელიც true ან false მნიშვნელობას ღებულობს.

```
@media not|only მედიატიპი and (expressions) {
  CSS-კოდი;
}
```

მოთხოვნების შედეგი ჭეშმარიტია, თუ მედიატიპი დოკუმენტის მოწყობილობაზე ასახვის ტიპს შეესაბამება და მედიამოთხოვნების ყველა გამოსახულების მნიშვნელობაა true. როდესაც მედიამოთხოვნების მნიშვნელობა არის true, მისი შესაბამისი სტილები ან სტილების წესები ჩვეულებრივად გამოიყენება.

თუ მხოლოდ კონკრეტულ მოწყობილობას იყენებთ, მაშინ მედიატიპის მითითება აუცილებელია და თუ არ იყენებთ, მაშინ all მნიშვნელობა იგულისხმება.

თქვენ ასევე შეიძლება მედიასაშუალებებისთვის სხვადასხვა სტილის ცხრილი გქონდეთ:

```
<link rel="stylesheet" media="მედიატიპი and|not|only (expressions)" href="print.css">
```

CSS3 მედიატიპები

მნიშვნელობა	აღწერა
all	გამოიყენება მედიატიპის ნებისმიერი მოწყობილობისათვის
print	გამოიყენება პრინტერისთვის
screen	გამოიყენება კომპიუტერების, პლანშეტების, სმარტფონებისა და სხვათა შემთხვევაში
speech	გამოიყენება ელექტრონული წიგნების ხმამაღლა წამკითხველის დროს

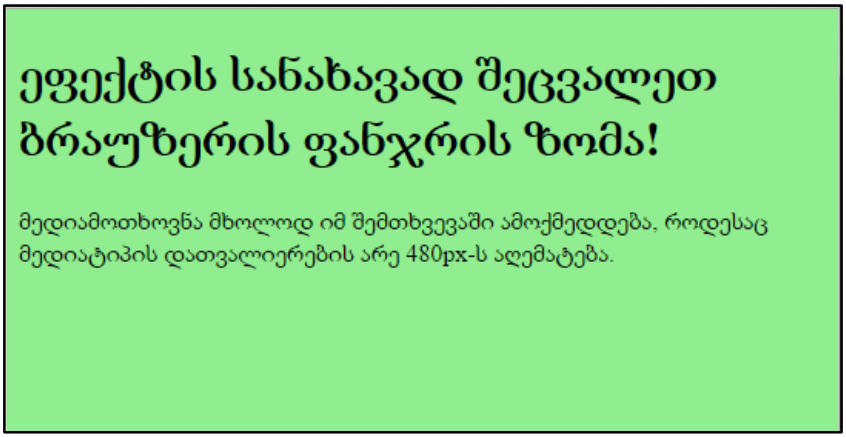
მედიამოთხოვნების ზოგიერთი მაგალითი

მედიამოთხოვნების გამოყენების ერთ-ერთი საშუალება არის ის, რომ თქვენი სტილების ცხრილში CSS-ის ალტერნატიული სტილების განყოფილება გქონდეთ.

შემდეგ მაგალითში ფონის ფერი ღია მწვანით იცვლება, მაშინ როდესაც დათვალიერების არე 480 პიქსელს აღემატება (თუ დათვალიერების არე 480 პიქსელზე ნაკლებია, მაშინ ფონი ვარდისფერია):

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background-color: pink;
```

```
    }
    @media screen and (min-width: 480px) {
      body {
        background-color: lightgreen;
      }
    }
    </style>
    </head>
    <body>
    <h1>ეფექტის სანახავად შეცვალეთ ბრაუზერის ფანჯრის
ზომა!</h1>
    <p>მედიამოთხოვნა მხოლოდ იმ შემთხვევაში ამოქმედდება,
როდესაც მედიატიპის დათვალიერების არე 480px-ს აღემატება.
</p>
    </body>
    </html>
```



შემდეგ მაგალითში არსებული მენიუ, დათვალიერების არის სიგანის 420 პიქსელზე მეტად გაზრდის შემთხვევაში, ტექსტის

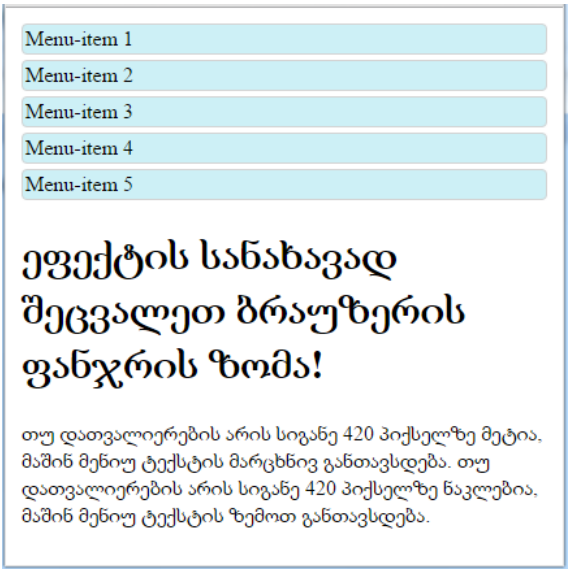
მარცხნივ გადაინაცვლებს, ხოლო შემცირების შემთხვევაში ტექსტის ზემოთ გადაადგილდება:

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-
scale=1.0">
<style>
.wrapper {overflow:auto;}
#main {margin-left: 4px;}
#leftsidebar {float: none;width: auto;}
#menulist {margin:0;padding:0;}
.menuitem  {   background:#CDF0F6;
               border:1px solid #d4d4d4;
               border-radius:4px;
               list-style-type:none;
               margin:4px;
               padding:2px;  }
@media screen and (min-width: 420px) {
   #leftsidebar {width:200px;float:left;}
   #main {margin-left:216px;}
}
</style>
</head>
<body>
<div class="wrapper">
  <div id="leftsidebar">
    <ul id="menulist">
      <li class="menuitem">Menu-item 1</li>
```

```
            <li class="menuitem">Menu-item 2</li>
            <li class="menuitem">Menu-item 3</li>
            <li class="menuitem">Menu-item 4</li>
            <li class="menuitem">Menu-item 5</li>
          </ul>
        </div>
        <div id="main">
          <h1>ეფექტის სანახავად შეცვალეთ ბრაუზერის ფანჯრის
ზომა!</h1>
          <p>თუ დათვალიერების არის სიგანე 420 პიქსელზე მეტია,
მაშინ მენიუ ტექსტის მარცხნივ განთავსდება. თუ დათვალიერების
არის სიგანე 420 პიქსელზე ნაკლებია, მაშინ მენიუ ტექსტის ზემოთ
განთავსდება.</p>
        </div>
      </div>
      </body>
      </html>
```

როდესაც დათვალიერების არის სიგანე 420 პიქსელზე მეტია:

ხოლო როდესაც დათვალიერების არის სიგანე 420 პიქსელზე ნაკლებია, მაშინ შედეგი ასეთი იქნება:



განვიხილოთ მედიამოთხოვნის რამდენიმე მაგალითი:

პირველ რიგში ისეთი სია განვიხილოთ, სადაც ელემენტებად ელექტრონული ფოსტის მისამართებია გამოყენებული:

```
<!DOCTYPE html>
<html>
<head>
<style>
ul {
   list-style-type: none;
}
ul li a {
   color: green;
   text-decoration: none;
```

```
        padding: 3px;
        display: block;
    }
    </style>
    </head>
    <body>
    <ul>
      <li><a data-
email="tea_todua@yahoo.com" href="mailto:tea_todua@yahoo.com">Te
a Todua</a></li>
      <li><a data-
email="besotabatadze84@gmail.com" href="mailto:besotabatadze84@gm
ail.com">Beso Tabatadze</a></li>
      <li><a data-
email="temuri.sturua@gmail.com"href="mailto:temuri.sturua@gmail.co
m">Teimuraz Sturua</a></li>
    </ul>
    </body>
    </html>
```

ყურადღება მიაქციეთ data-email ატრიბუტს. HTML 5-ში ატრიბუტი პრეფიქსით data- შეგვიძლია ინფორმაციის შესანახად გამოვიყენოთ.

განვიხილოთ შემთხვევა, როცა ფანჯრის სიგანე 420 პიქსელსა და 599 პიქსელს შორის იქნება მოთავსებული, მაშინ ელექტრონული ფოსტის ნიშანი ყველა ბმულის წინ გამოიყენება. ჩვენ კოდს ამ შემთხვევაში შემდეგი მედიამოთხოვნა დაემატება:

```
@media screen and (max-width: 599px) and (min-width: 420px) {
    ul li a {
        padding-left: 30px;
```

```
        background: url(email-icon.jpg) left center no-repeat;
    }
}
```

დავამატოთ ახალი ეფექტი, როდესაც დათვალიერების არის სიგანე 600 პიქსელსა და 900 პიქსელს შორისაა მოთავსებული, მაშინ ბმულების წინ გამოჩნდეს ტექსტი "Email: ".

```
@media screen and (max-width: 900px) and (min-width: 600px) {
    ul li a:before {
        content: "Email: ";
        font-style: italic;
        color: #607080;
    }
}
```

მომდევნო მაგალითში დამატებულია ეფექტი, როდესაც დათვალიერების არის სიგანე 900 პიქსელზე მეტია, მაშინ ბმულების შემდეგ გამოჩნდება მათი ელექტრონული ფოსტის ზუსტი მისამართიც:

```
@media screen and (min-width: 901px) {
    ul li a:after {
        content: " (" attr(data-email) ")";
        font-size: 12px;
        font-style: italic;
        color: #607080;
    }
}
```

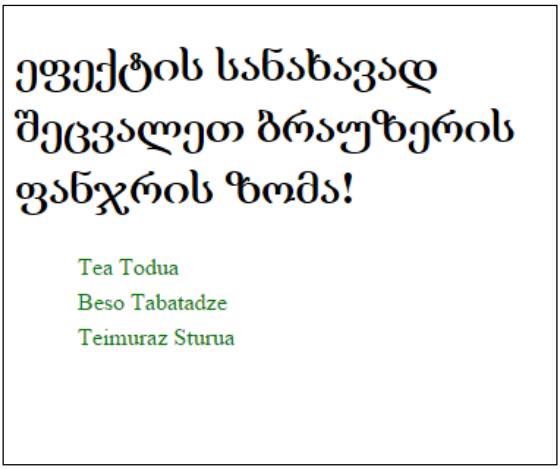
როგორც ზემოთ ვნახეთ, ახლაც თუ დათვალიერების არის სიგანე 1051 პიქსელზე მეტია, მაშინ ყველა ბმულის წინ ელექ-ტრონული ფოსტის ნიშანი დავუმატოთ. ამისათვის დამატებითი

ბლოკის ჩაწერა არ იქნება საჭირო, უბრალოდ არსებულ ბლოკში მძიმის შემდეგ კიდევ ერთ მედიამოთხოვნას (min-width: 1051px) დავუმატებთ. საბოლოოდ, მიღებულ კოდს შემდეგი სახე ექნება:

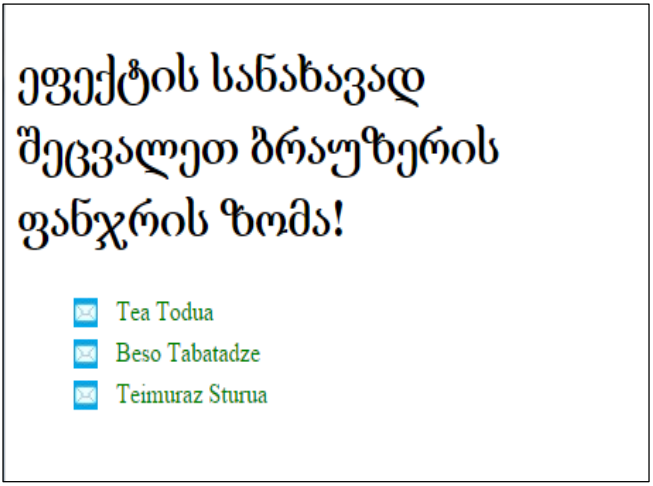
```
<!DOCTYPE html>
<html>
<head>
<style>
ul {
  list-style-type: none;
}
ul li a {
  color: green;
  text-decoration: none;
  padding: 3px;
  display: block;
}
@media screen and (max-width: 599px) and (min-width: 420px),
(min-width: 1051px) {
  ul li a {
    padding-left: 30px;
    background: url(email-icon.jpg) left center no-repeat;
  }
}
@media screen and (max-width: 900px) and (min-width: 600px) {
  ul li a:before {
    content: "Email: ";
    font-style: italic;
    color: #607080;
  }
```

```
    }
    @media screen and (min-width: 901px) {
      ul li a:after {
        content: " (" attr(data-email) ")";
        font-size: 12px;
        font-style: italic;
        color: #607080;
      }
    }
    </style>
    </head>
    <body>
      <h1>ეფექტის სანახავად შეცვალეთ ბრაუზერის ფანჯრის
ზომა!</h1>
    <ul>
      <li><a data-
email="tea_todua@yahoo.com" href="mailto:tea_todua@yahoo.com">Te
a Todua</a></li>
      <li><a data-
email="besotabatadze84@gmail.com" href="mailto:besotabatadze84@gm
ail.com">Beso Tabatadze</a></li>
      <li><a data-
email="temuri.sturua@gmail.com"href="mailto:temuri.sturua@gmail.co
m">Teimuraz Sturua</a></li>
    </ul>
    </body>
    </html>
```

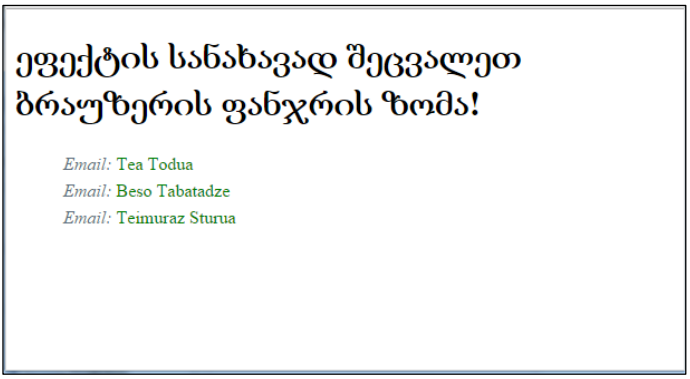
როდესაც დათვალიერების არის სიგანე 420 პიქსელზე ნაკლებია, მაშინ შედეგი ასეთი იქნება:



ამ კოდის შესრულების შედეგად, როდესაც დათვალიერების არის სიგანე 420 პიქსელსა და 599 პიქსელს შორისაა მოთავსებული, მაშინ შედეგი ასეთი იქნება:

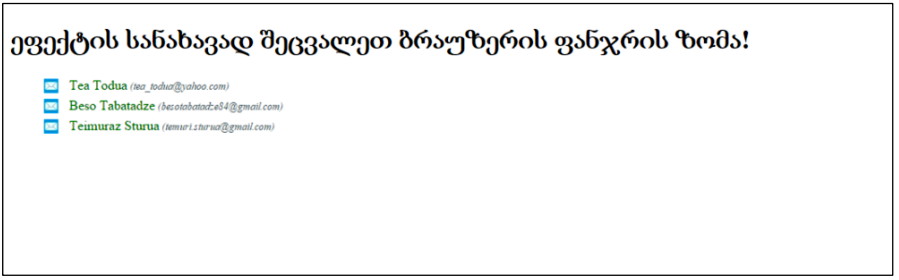


როდესაც დათვალიერების არის სიგანე 600 პიქსელსა და 900 პიქსელს შორისაა მოთავსებული, მაშინ შედეგი ასეთი იქნება:



როდესაც დათვალიერების არის სიგანე 901 პიქსელსა და 1051 პიქსელს შორისაა მოთავსებული, მაშინ შედეგი ასეთი იქნება:

როდესაც დათვალიერების არის სიგანე 1051 პიქსელზე მეტია, მაშინ შედეგი ასეთი იქნება:



დანართი 1. HTML5-ის ახალი ტეგები

ტეგის დასახელება	ტეგის აღწერა	თანამედროვე ბრაუზერების მხრიდან მხარდაჭერა
<article>	გამოიყენება ცალკეული ინფორმაციების განთავსების შემთხვევაში	სრული
<aside>	გამოიყენება გვერდის ცალკეული ნაწილებისთვის	სრული
<audio>	აუდიოფაილის განთავსება	სრული
<canvas>	სკრიპტების დახმარებით ნებისმიერი ობიექტის დახატვა	სრული
<datalist>	ინფორმაციის შეტანის ელემენტებისათვის მონაცემთა სიას განსაზღვრავს	ნაწილობრივი
<details>	დამატებითი ინფორმაციის ველს განსაზღვრავს, რომელიც შეიძლება მომხმარებელმა დამალოს ან გამოაჩინოს	ნაწილობრივი
<embed>	გამოიყენება სხვადასხვა ტიპის ფაილების ჩასატვირთად	სრული
<figcaption>	გამოსახულებისათვის წარწერას განსაზღვრავს	სრული
<figure>	გამოსახულებისათვის წარწერის შესაქმნელად გამოიყენება	სრული
<footer>	დოკუმენტისათვის განსაზღვრავს ბოლოსართს	სრული
<header>	დოკუმენტისათვის განსაზღვრავს თავსართის ბლოკს	სრული
<hggroup>	სათაურების დაჯგუფების საშუალებას იძლევა	სრული

<keygen>	სერვერთან უსაფრთხო კავშირისთვის ღია და დახურული გასაღების გენერირება	ნაწილობრივი
<mark>	შესაძლებელია ტექსტის მნიშვნელობის ხაზგასმა	ნაწილობრივი
<meter>	საზომ ზოლს განსაზღვრავს	ნაწილობრივი
<menu>	კონტექსტური მენიუს და ინსტრუმენტთა პანელის განსაზღვრის საშუალებას იძლევა	ნაწილობრივი
<nav>	საიტის სანავიგაციო ბლოკს განსაზღვრავს	სრული
<output>	მონაცემების გამოსატანად გამოიყენება	ნაწილობრივი
<progress>	რომელიმე პროცესის შესრულების მდგომარეობას ასახავს	ნაწილობრივი
<section>	ლოგიკურად დაკავშირებული შინაარსის დაჯგუფება	სრული
<source>	<audio> და <video> ელემენტების გასაშვებად რამდენიმე წყაროს მითითების საშუალებას იძლევა	სრული
<summary>	<details> ელემენტისათვის ხილულ სათაურს განსაზღვრავს	ნაწილობრივი
<time>	თარიღსა და დროს განსაზღვრავს	არც ერთი ბრაუზერი მხარს არ უჭერს
<track>	<audio> და <video> ელემენტები-სთვის სუბტიტრების დამატების საშუალებას იძლევა	არც ერთი ბრაუზერი მხარს არ უჭერს
<video>	ვიდეოფაილების განთავსება	სრული
<wbr>	ტექსტში სათანადო ადგილს აღნიშნავს გადატანისათვის	ნაწილობრივი

დანართი 2. HTML-ის ძველი ტეგები

ქვემოთ ცხრილში მხოლოდ ის ძველი ტეგებია მოცემული, რომლებიც უკვე HTML4-ში ან მის წინა ვერსიებში იყო და ახლაც აქტუალურია

ტეგის დასახელება	ტეგის აღწერა
<!--....-->	განსაზღვრავს ერთსტრიქონიან კომენტარს
<!	განსაზღვრავს მრავალსტრიქონიან კომენტარს
<!DOCTYPE>	ინსტრუქცია განსაზღვრავს დოკუმენტის ტიპს, გამოიყენება HTML-ის ვერსიის მისათითებლად
<a>	განსაზღვრავს ჰიპერბმულს
<abbr>	განსაზღვრავს აკრონიმს ანუ აბრევიატურას
<address>	განსაზღვრავს დოკუმენტის ავტორის საკონტაქტო ინფორმაციას
<area />*	განსაზღვრავს მიმართვას გამოსახულებაზე
<b>	განსაზღვრავს მუქ ტექსტს
<base />*	განსაზღვრავს ბმულის საბაზო მისამართს, რომელიც ავტომატურად ფარდობით მისამართებს დაემატება
<bdo>	განსაზღვრავს ტექსტის მიმართულებას
<blockquote>	განსაზღვრავს დიდი მოცულობის ციტატას
<body>	განსაზღვრავს დოკუმენტის ტანს
 *	ახალ სტრიქონზე გადასვლა

<button>	განსაზღვრავს ღილაკს
<caption>	ცხრილის სათაურის განსაზღვრა
<cite>	ტექსტს ციტატის სახით აფორმებს
<code>	ტექსტს კომპიუტერული კოდის სახით აფორმებს
<dd>	სია განსაზღვრებაში ტერმინის მნიშვნელობას განსაზღვრავს
<del>	განსაზღვრავს გადახაზულ ტექსტს
	განსაზღვრავს დოკუმენტში ცალკე ბლოკს
<dl>	გამოიყენება განსაზღვრებათა სიის შესაქმნელად
<dt>	განსაზღვრებათა სიაში ტერმინს განსაზღვრავს
	განსაზღვრავს აქცენტირებულ ტექსტს
<fieldset>	ფორმის ელემენტების გარშემო საზღვრებს განსაზღვრავს
<form>	მომხმარებლის მონაცემების შესატან ფორმას განსაზღვრავს
<h1> - <h6>	სათაურის დონეებს განსაზღვრავს
<head>	დოკუმენტის შესახებ საცნობარო ინფორმაციას განსაზღვრავს
<hr />*	განსაზღვრავს ჰორიზონტალურ ხაზებს
<html>	HTML დოკუმენტის ძირითად ტეგებს განსაზღვრავს
<i>	განსაზღვრავს დახრილ ტექსტს
<iframe>	განსაზღვრავს სტრიქონულ ფრეიმს

<img />*	განსაზღვრავს გამოსახულებას
<input />*	განსაზღვრავს ფორმაში მონაცემების შესატან ველს
<ins>	განსაზღვრავს ჩამატებულ ტექსტს
<label>	Input ელემენტისათვის განსაზღვრავს ჭდეს
<legend>	fieldset, figure და details ელემენტებისათვის განსაზღვრავს სათაურს
	განსაზღვრავს სიის ელემენტებს
<link />*	მიმდინარე დოკუმენტსა და გარე ფაილს შორის კავშირს განსაზღვრავს
<map>	განსაზღვრავს გამოსახულების რუკას
<meta />*	HTML დოკუმენტის მეტა მონაცემებს განსაზღვრავს
<noscript>	ალტერნატიულ შინაარსს განსაზღვრავს იმ მომხმარებლებისათვის, რომელთა ბრაუზერები კლიენტის სკრიპტებს მხარს არ უჭერს
<object>	განსაზღვრავს ჩაშენებულ ობიექტს
	განსაზღვრავს დანომრილ სიებს
<optgroup>	ამოსარჩევ სიაში მსგავსი ელემენტების ჯგუფს განსაზღვრავს
<option>	ამოსარჩევ სიაში განსაზღვრავს ვარიანტს
<p>	განსაზღვრავს აბზაცს
<param />*	ჩაშენებული ობიექტისათვის განსაზღვრავს პარამეტრს
<pre>	წინასწარ დაფორმატებულ ტექსტს განსაზღვრავს

<script>	კლიენტის სკრიპტს განსაზღვრავს
<select>	განსაზღვრავს ჩამოშლად სიას
<small>	განსაზღვრავს ტექსტს პატარა შრიფტით
	გამოიყენება ტექსტური დონის ინფორმაციის გასაფორმებლად
	განსაზღვრავს მნიშვნელოვან ტექსტს
<style>	განსაზღვრავს ინფორმაციას დოკუმენტის სტილის შესახებ
<sub>	განსაზღვრავს ელემენტის ქვედა ინდექსს
<sup>	განსაზღვრავს ელემენტის ზედა ინდექსს
<table>	განსაზღვრავს ცხრილს
<tbody>	აჯგუფებს ცხრილის ტანის შიგთავსს
<td>	ცხრილში განსაზღვრავს უჯრედებს
<textarea>	მრავალსტრიქონიანი ტექსტის შესატან ველს განსაზღვრავს
<tfoot>	აჯგუფებს ცხრილის ქვედა შიგთავსს
<th>	ცხრილში განსაზღვრავს სათაურს
<thead>	აჯგუფებს ცხრილში სათაურის უჯრედებს
<title>	დოკუმენტის სათაურს განსაზღვრავს
<tr>	ცხრილში განსაზღვრავს სტრიქონს
	განსაზღვრავს მარკირებულ სიებს

*<ტეგი />- აღნიშნული ტეგები არაწყვილი ტეგებია.

რედაქტორი ლ. მამალაძე

გადაეცა წარმოებას 10.05.2016. ხელმოწერილია დასაბეჭდად 02.06.2016. ქაღალდის ზომა 60X84 1/16. პირობითი ნაბეჭდი თაბახი 20,5.

საგამომცემლო სახლი „ტექნიკური უნივერსიტეტი“, თბილისი, კოსტავას 77

Verba volant, scripta manent

ი.მ. „გოჩა დალაქიშვილი“,
თბილისი, ვარკეთილი 3, კორპ. 333, ბინა 38