სტილები

რამდენ ელემენტსაც არ უნდა შეიცავდეს, html დოკუმენტი იქნება უსიცოცხლო სტილების გამოყენების გარეშე. სტილები, ანუ უფრო ზუსტად სტილების კასკადური ცხრილები (Cascading Style Sheets) ან უბრალოდ CSS განსაზღვრავს დოკუმენტის გარეგნულ იერსახეს.მოკლედ განვიხილოთ სტილების გამოყენება HTML5-ში.

CSS სტილი წარმოადგენს წესს და უთითებს ბრაუზერს, თუ როგორ უნდა გამოისახოს ესა თუ ის ელემენტი, როგორ მოხდეს მისი ფორმატირება. ფორმატირება გულისხმობს ელემენტის ფერს, ფონს, შრიფტს, ზომებს და ა. შ.

სტილის განსაზღვრა შედგება ორი ნაწილისგან: სელექტორი, რომელიც მიუთითებს ელემენტზე და სტილის გამოცხადების ბლოკი - ბრძანებების ნაკრები, რომლებიც ადგენენ ფორმატირების წესებს. მაგალითად:

| * div{ * background-color:red; * width: 100px; * height: 60px; * } |
| --- |

მოცემულ მაგალითში სელექტორს წარმოადგენს div. ეს სელექტორი მიუთითებს, რომ მოცემული სტილი გამოიყენება ყველა div ელემენტისთვის. სელექტორის შემდეგ ფიგურულ ფრჩხილებში მოცემულია სტილების გამოცხადების ბლოკი. ფიგურულ ფრჩხილებში მოქცეულია ბრძანებები, რომლებიც განსაზღვრავენ, თუ როგორ უნდა მოხდეს ელემენტების ფორმატირება.

თითოეული ბრძანება შედგება თვისებისგან და მნიშვნელობისგან. მაგალითად ბრძანებაში

| * background-color:red; |
| --- |

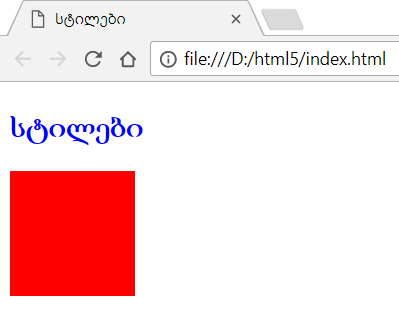
background-color წარმოადგენს თვისებას, ხოლო red - მნიშვნელობას. თვისება განსაზღვრავს კონკრეტულ სტილს. თვისებები CSS-ში მრავლადაა. მაგალითად, background-color განსაზღვრავს ფონის ფერს. ორწერტილის შემდეგ მოდის ამ თვისების მნიშვნელობა. ზემოთ მოცემული ბრძანება განსაზღვრავს, რომ ფონის ფერი არის წითელი. ყოველი ასეთი ბრძანების ბოლოს იწერება წერტილმძიმე.

ასეთი სტილების ერთობლიობას უწოდებენ სტილების ცხრილს ანუ CSS (Cascading Style Sheets). არსებობს სტილების გამოცხადების სხვადასხვა ხერხი.

ატრიბუტი style

სტილების გამოცხადების ერთ-ერთი ხერხი მდგომარეობს იმაში, რომ ხდება სტილის ჩაშენება უშუალოდ ელემენტში style ატრიბუტის სახით:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>სტილები</title> * </head> * <body> * <h2 style="color:blue;">სტილები</h2> * <div style="width: 100px; height: 100px; background-color: red;"></div> * </body> * </html> |
| --- |



აქ მოცემულია ორი ელემენტი: სათაური h2 და div. სათურს განსაზღვრული აქვს ტექსტის ფერი თვისების color მეშვეობით, ხოლო div ელემენტს თვისებები სიგანე (width), simaRle(height) და ფონის ფერი (background-color).

მეორე მეთოდი მდგომარეობს html დოცუმენტში style ელემენტის გამოყენება. იგი უთითებს ბრაუზერს, რომ მის შიგნით არსებული მონაცემები წარმოადგენს CSS კოდს, და არა html-ს:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>სტილები</title> * <style> * h2{ * color:blue; * } * div{ * width: 100px; * height: 100px; * background-color: red; * } * </style> * </head> * <body> * <h2>სტილები</h2> * <div></div> * </body> * </html> |
| --- |

ამ შემთხვევაშიც შედეგი იქნება ზუსტად ისეთივე, როგორც წინა შემთხვევაში.

ელემენტი style ძირითადად განისაზღვრება head ელემენტის შიგნით, თუმცა შეიძლება იყოს სხვა ადგილზეც. იგი შეიცავს სტილების ერთობლიობას. თითოეული სტილი შედგება სელექტორისგან და ფიგურულ ფრჩხილებში მოქცეული თვისებებისაგან თავისი მნიშვნელობებით.

პირველისგან განსხვავებით, მეორე ხერხი კოდს ხდის უფრო სუფთას სტილების ცალკე ელემენტად გამოტანის გამო. ასევე არსებობს მესამე ხერხიც, რომელიც მდგომარეობს სტილების ცალკე ფაილად გატანაში.

შევქმნათ იგივე საქაღალდეში, სადაც ჩვენი html ფაილია, ფაილი styles.css და შიგნით ჩავწეროთ იგივე სტილები, რაც გვქონდა style ელემენტში.:

| * h2{ * color:blue; * } * div{ * width: 100px; * height: 100px; * background-color: red; * } |
| --- |

თვითონ html ფაილის ტექსტი შევცვალოთ შემდეგნაირად:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>სტილები</title> * <link rel="stylesheet" type="text/css" href="styles.css"/> * </style> * </head> * <body> * <h2>სტილები</h2> * <div></div> * </body> * </html> |
| --- |

html დოკუმენტში ახლა უკვე არა გვაქვს ელემენტი style, სამაგიეროდ დაემატა ელემენტი link, რომელიც ახდენს ჩვენს მიერ შექმნილი სტილების ფაილის styles.css ჩართვას: <link rel="stylesheet" type="text/css" href="styles.css"/>.

ამგვარად, ცალკე ფაილში სტილების განსაზღვრის შედეგად html კოდი უფრო სუფთაა, გვერდის სტრუქტურა გამოყოფილია მისი სტილიზაციისაგან. ასეთი მიდგომისას უფრო იოლია სტილების მოდიფიცირება, ვიდრე იმ შემთხვევაში, თუ სტილები იქნებოდა ელემენტ style-ში ან თითოეული ელემენტის ატრიბუტ style-ში. შესაბამისად ასეთი მიდგომა უფრო მიღებული და უპირატესია HTML5-ში.

ცალკე ფაილებში სტილების გატანა ამცირებს დატვირთვას ვებსერვერზე კეშირების მექანიზმის გამოყენებით. რადგან ვებბრაუზერს შეუძლია მოახდინოს CSS ფაილის კეშირება და შემდგომში მიმართავს კეშირებულ ფაილს სერვერის ნაცვლად.

შესაძლებელია მოხდეს ამ სამი მეთოდის შერწყმა, როდესაც ერთი ელემენტის ზოგიერთი თვისება განსაზღვრულია ცალკე css ფაილში, ზოგი style ელემენტში, ხოლო ზოგი თვითონ ამ ელემენტის style ატრიბუტში. მაგალითად:

| * <!DOCTYPE html> * <html> * <head> * <link rel="stylesheet" type="text/css" href="styles.css"/> * <style> * div{ * width:200px; * } * </style> * </head> * <body> * <div style="width:120px;"></div> * </body> * </html> |
| --- |

| * div{ * width:50px; * height:50px; * background-color:red; * } |
| --- |

მოცემულ მაგალითში div ელემენტისთვის ერთი და იგივე თვისება width განსაზღვრულია 3 ადგილას სხვადასხვა მნიშვნელობებით. რომელ მნიშვნელობას მიიღებს ელემენტი? მოქმედებს პრიორიტეტების შემდეგი სისტემა:

* თუ ელემენტს განსაზღვრული აქვს ჩაშენებული სტილები (inline-სტილი) style ატრიბუტის სახით, ისინი წარმოადგენენ უმაღლეს პრიორიტეტს. ჩვენს შემთხვევაში ელემენტის სიგანე იქნება 12 პიქსელი.
* შემდეგი პრიორიტეტში მოდის სტილები, განსაზღვრული ელემენტში style.
* ცალკე ფაილებში განსაზღვრულ სტილებს აქვთ ყველაზე დაბალი პრიორიტეტი.

html ატრიბუტები და CSS

html-ის ბევრ ელემენტს აქვს შესაძლებლობა სტილები განესაზღვროს ატრიბუტების მეშვეობით. მაგალითად, ბევრ ელემენტს აქვს ატრიბუტები width და height სიგანისა და სიმაღლის მისათითებლად. თუმცა ყოველთვის უმჯობესია თავი ავარიდოთ სტილების ამგვარად განსაზღვრას და გამოვიყენოთ CSS სტილები. უმჯობესია html-ში მოცემული იყოს ვებგვერდის სტრუქტურა, ხოლო სტილიზაცია მოხდეს CSS სტილების მეშვეობით.

CSS კოდის ვალიდაცია

CSS კოდის დაწერისას შეიძლება წამოიჭრას კითხვები, სწორად არის ტუ არა განსაზღვრული სტილები, კორექტულია თუ არა ისინი. ასეთ შემთხვევაში შეიძლება ვისარგებლოთ CSS ვალიდატორით, რომელიც ხელმისაწვდომია მისამართზე: <http://jigsaw.w3.org/css-validator/> .

სელექტორები

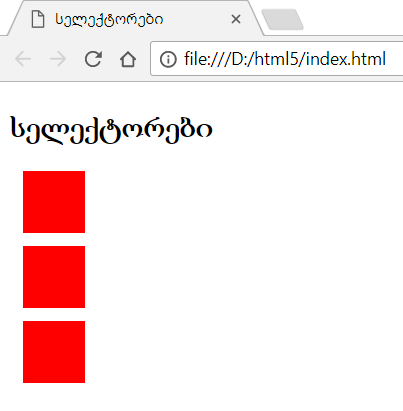
სტილის განსაზღვრა იწყება სელექტორით:

| * div{ * width:50px; /\* სიგანე \*/ * height:50px; /\* სიმაღლე \*/ * background-color:red; /\* ფონის ფერი \*/ * margin: 10px; /\* დაშორება სხვა ელემენტებისგან \*/ * } |
| --- |

მოცემულ მაგალითში სელექტორს წარმოადგენს div. რიგ სელექტორებს აქვთ იმ ელემენტის სახელი, რომლის ფორმატირებასაც ისინი ემსახურებიან. მაგალითად, div, p, h2 და ა. შ. ასეთი სელექტორებით განსაზღვრული სტილები ვრცელდება ყველა შესაბამის ელემენტზე. შესაბამისად, ზემოთ განსაზღვრული სტილი გავრცელდება გვერდის ყველა div ელემენტზე:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>სელექტორები</title> * <style> * div{ * width:50px; * height:50px; * background-color:red; * margin: 10px; * } * </style> * </head> * <body> * <h2>სელექტორები</h2> * <div></div> * <div></div> * <div></div> * </body> * </html> |
| --- |

აქ სამი div ელემენტია და სამივე მიიღებს განსაზღვრულ სტილს:



კლასები

ზოგჯერ ერთი და იგივე ელემენტებისთვის საჭიროა სხვადასხვანაირი სტილის მიცემა. ამისთვის გამოიყენება კლასები. კლასის სელექტორის განსაზღვრისთვის მისი სახელის წინ დაისმის წერტილი:

| * .redBlock{ * background-color:red; * } |
| --- |

კლასის დასახელება შეიძლება იყოს ნებისმიერი. ჩვენს მაგალითში კლასის სახელია redBlock. კლასის დასახელებაში დასაშვებია ასოების, ციფრების, ტირეს და ქვედა ტირეს გამოყენება, პირველი სიმბოლო აუცილებლად უნდა იყოს ასო.

მნიშვნელობა აქვს ასოების რეგისტრს: "article" და "ARTICLE" იქნება სხვადასხვა კლასი.

კლასის განსაზღვრის შემდეგ მისი გამოყენება შეიძლება ელემენტთან class ატრიბუტის სახით. მაგალითად:

| * <div class="redBlock"></div> |
| --- |

განვსაზღვროთ და გამოვიყენოთ რამდენიმე კლასი:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>კლასები</title> * <style> * div{ * width: 50px; * height: 50px; * margin: 10px; * } * .redBlock{ * background-color: red; * } * .blueBlock{ * background-color: blue; * } * </style> * </head> * <body> * <h2>კლასები</h2> * <div class="redBlock"></div> * <div class="blueBlock"></div> * <div class="redBlock"></div> * </body> * </html> |
| --- |



იდენტიფიკატორები

ვებგვერდის უნიკალური ელემენტების იდენტიფიცირებისთვის გამოიყენება იდენტიფიკატორები, რომლებიც გამოისახება ატრიბუტით id. მაგალითად, გვერდზე შეიძლება იყოს სათაურის ბლოკი:

| * <div id="header"></div> |
| --- |

იდენტიფიკატორებისთვის სტილის განსაზღვრა კლასებისთვის სტილის განსაზღვრის იდენტურია, მხოლოდ სელექტორის წინ წერტილის ნაცვლად დაისმის სიმბოლო #:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>იდენტიფიკატორები</title> * <style> * div{ * margin: 10px; * border: 1px solid #222; * } * #header{ * height: 80px; * background-color: #ccc; * } * #content{ * height: 180px; * background-color: #eee; * } * #footer{ * height: 80px; * background-color: #ccc; * } * </style> * </head> * <body> * <div id="header">საიტის სათაური</div> * <div id="content">ძირითადი ნაწილი</div> * <div id="footer">ქვედა ნაწილი</div> * </body> * </html> |
| --- |



უნდა აღინიშნოს, რომ იდენტიფიკატორები ძირითადად გამოიყენება არა სტილიზაციისთვის, არამედ ვებგვერდის სტრუქტურის ასაგებად. სტილიზაციისთვის ძირითადად გამოიყენება კლასები.

უნივერესალური სელექტორი

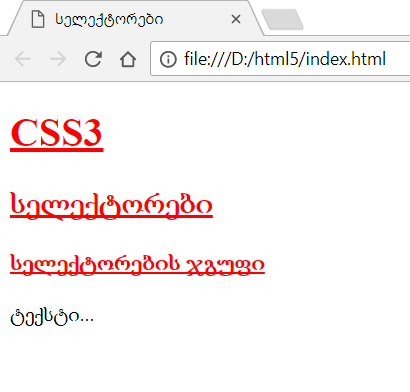
CSS-ში არის ე. წ. უნივერსალური სელექტორი ვარსკვლავის (8) სახით. ის ანიჭებს სტილს ვებგვერდის ყველა ელემენტს:

| * \*{ * background-color: red; * } |
| --- |

სელექტორთა ჯგუფების სტილიზაცია

ზოგჯერ სტილები განისაზღვრება სელექტორთა მთელი ჯგუფისთვის. მაგალითად, თუ გვინდა ყველა სათაურისთვის ხაზგასმის გამოყენება, შესაბამისი სელექტორები უნდა ჩამოვთვალოთ მძიმით გამოყოფილი:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>სელექტორები</title> * <style> * h1, h2, h3, h4{ * text-decoration: underline; * color: red; * } * </style> * </head> * <body> * <h1>CSS3<h1> * <h2>სელექტორები</h2> * <h3>სელექტორების ჯგუფი</h3> * <p>ტექსტი...</p> * </body> * </html> |
| --- |



ჯგუფში შეიძლება გაერთიანებული იყოს როგორც ტეგების, ასევე კლასების და იდენტიფიკატორების სელექტორები:

| * h1, #header, .redBlock{ * color: red; * } |
| --- |

"შთამომავლების" სელექტორები

ვებგვერდს გააჩნია რთული ორგანიზაცია, რომლის დროსაც ერთი ელემენტები შეიცავენ სხვა ელემენტებს. ჩასმულ ელემენტებს (რომლებიც სხვა ელემენტის შიგნით არიან ჩასმული) მეორენაირად "შთამომავლებს" უწოდებენ, ხოლო კონტეინერს, რომელიც შეიცავს ამ ელემენტებს - მშობელს.

ვთქვათ, ელემენტი body-ს აქვს შემდეგი სახე:

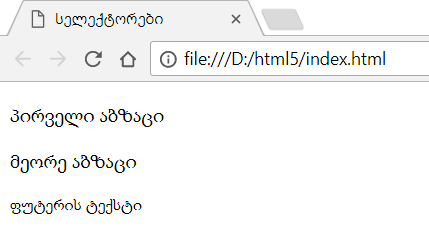
| * <body> * <h2>სათაური</h2> * <div> * <p>ტექსტი</p> * </div> * </body> |
| --- |

body ელემენტის შიგნით გვაქვს სამი ჩასმული ელემენტი: h2, div და p. სამივე მათგანი წარმოადგენს body ელემენტის შთამომავალს.

div ელემენტის შიგნით ჩასმულია ერთი ელემენტი p, ამიტომ div-ს მხოლოდ ერთი შთამომავალი გააჩნია.

სპეციალური სელექტორების მეშვეობით შეგვიძლია მოვახდინოთ სტილიზაცია მკაცრად განსაზღვრული ელემენტების შთამომავლებისთვის. მაგალითად, შეიძლება გვქონდეს p ელემენტი როგორც ძირითადი ნაწილის შიგნით, ასევე ფუტერში, მაგრამ გვინდოდეს მათთვის სხვადასხვა შრიფტის მინიჭება.

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>სელექტორები</title> * <style> * #main p{ * font-size: 16px; * } * #footer p{ * font-size: 13px; * } * </style> * </head> * <body> * <div id="main"> * <p>პირველი აბზაცი</p> * <p>მეორე აბზაცი</p> * </div> * <div id="footer"> * <p>ფუტერის ტექსტი</p> * </div> * </body> * </html> |
| --- |



ჩასმული ელემენტისთვის სტილის მისაყენებლად გამოიყენება სელექტორი, რომელიც შეიცავს ჯერ მშობელ ელემენტს და შემდეგ ჩასმულს:

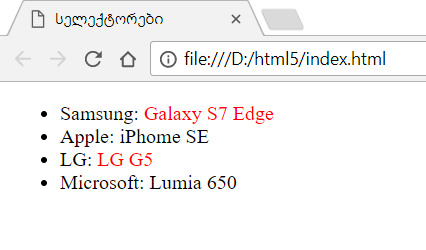
| * #main p{ * font-size: 16px; * } |
| --- |

ეს სტილი გავრცელდება მხოლოდ იმ p ელემენტებზე, რომლებიც იმყოფებიან main იდენტიფიკატორის მქონე ელემენტის შიგნით.

ვნახოთ მეორე მაგალითი:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>სელექტორები</title> * <style> * li .redLink{ * color: red; * } * </style> * </head> * <body> * <ul> * <li>Samsung: <a class="redLink">Galaxy S7 Edge</a></li> * <li>Apple: <a>iPhome SE</a></li> * <li>LG: <a class="redLink">LG G5</a></li> * <li>Microsoft: <a>Lumia 650</a></li> * </ul> * </body> * </html> |
| --- |

აქ სტილი გამოიყენება ელემენტებისათვის კლასით "redLink", რომლებიც <li>ელემენტის შიგნითაა:



მიაქციეთ ყურადღება, რომ li ელემენტსა და .redLink კლასს შორის ადგილია გამოტოვებული. ადგილის გამოტოვებას დიდი მნიშვნელობა აქვს. იგი მიუთითებს, რომ სტილი ეხება მხოლოდ იმ ელემენტებს, რომელთა კლასია "redLink" და ამასთან იმყოფებიან li ელემენტის შიგნით.

თუ მოვაშორებთ ადგილის გამოტოვებას, სელექტორის შინაარსი შეიცვლება:

| * li.redLink{ * color: red; * } |
| --- |

ახლა უკვე სტილი ეხება იმ liელემენტებს, რომელთაც აქვთ კლასი "redLink". იგი ეხება ქვემოთ მოცემულ ელემენტს:

| * <li class="redLink">Microsoft: <a>Lumia 650</a></li> |
| --- |

მაგრამ არა ამ ელემენტს:

| * <li>LG: <a class="redLink">LG G5</a></li> |
| --- |

"შვილი" ელემენტების სელექტორები

შვილი ელემენტების სელექტორები შთამომავლების სელექტორებისგან განსხვავდება იმით, რომ ეხება მხოლოდ პირველი დონის ჩასმულ ელემენტებს. მაგალითად:

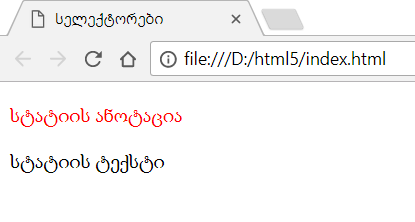
| * <body> * <h2>სათაური</h2> * <div> * <p>ტექსტი</p> * </div> * </body> |
| --- |

მოცემულ მაგალითში body ელემენტისთვის ჩასმულ ელემენტებს წარმოადგენენ h2, div და p, მაგრამ მათგან შვილი ელემენტი მხოლოდ ორია - h2 და div, რადგან იმყოფებიან პირველ დონეზე.

შვილი ელემენტებისადმი მიმართვის სელექტორში გამოიყენება კუთხოვანი ფრჩხილი:

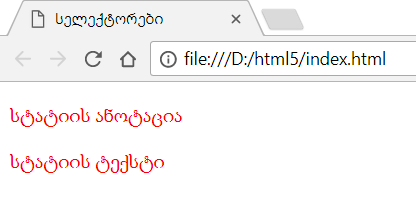
| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>სელექტორები</title> * <style> * .article > p{ * color: red; * } * </style> * </head> * <body> * <div class="article"> * <p>სტატიის ანოტაცია</p> * <div class="content"> * <p>სტატიის ტექსტი</p> * </div> * </div> * </body> * </html> |
| --- |

ბლოკში კლასით "article" არის ორი პარაგრაფი p, მაგრამ სელექტორი .article>p ეხება მხოლოდ იმ პარაგრაფს, რომელიც უშუალოდ .article ბლოკის შიგნითაა:



ჩვენ რომ გამოგვეყენებინა სელექტორი ადგილის გამოტოვებით კუთხოვანი ფრჩხილის ნაცვლად, სტილი გავრცელდებოდა ყველა p ელემენტზე, რადგან მივიღებდით სელექტორს შთამომავლებისთვის:

| * .article p{ * color: red; * } |
| --- |



სელექტორები ერთი დონის ელემენტებისთვის

ერთი დონის ელემენტებს მეორენაირად სიბლინგებს (siblings) ან დებს უწოდებენ.მაგალითდ:

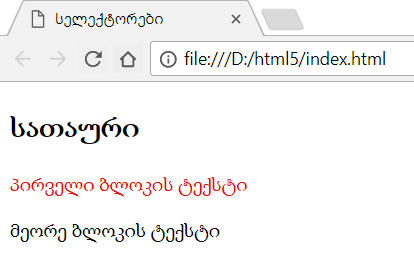
| * <body> * <h2>Заголовок</h2> * <div> * <p>Текст первого блока</p> * </div> * <div> * <p>Текст второго блока</p> * </div> * </body> |
| --- |

მოცემულ მაგალითში ელემენტი h2 და ორივე ელემენტი div წარმოადგენს ერთი დონის ელემენტებს. ხოლო პარაგრაფის ელემენტები p არ წარმოადგენენ მათთვის იგივე დონის ელემენტებს, რადგან ისინი div ელემენტების შიგნით იმყოფებიან.

ერთ დონეზე მყოფი პირველი მომდევნო ელემენტის სტილიზაციისთვის სელექტორში გამოიყენება + ნიშანი:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>სელექტორები</title> * <style> * h2+div { color: red; } * </style> * </head> * <body> * <h2>სათაური</h2> * <div> * <p>პირველი ბლოკის ტექსტი</p> * </div> * <div> * <p>მეორე ბლოკის ტექსტი</p> * </div> * </body> * </html> |
| --- |

სელექტორი h2+div საშუალებას იძლევა მოხდეს სტილიზაცია იმ div ბლოკის, რომელიც უშუალოდ h2 ელემენტის შემდეგ მოდის:



ამ სელექტორით div ბლოკის სტილიზაცია ხდება მხოლოდ იმ შემთხვევაში, თუ ის უშუალოდ h2 ელემენტის შემდეგაა. თუ შუაში სხვა ელემენტიც იქნება, სელექტორი div ბლოკს არ შეეხება.

თუ ჩვენ გვინდა მოვახდინოთ h2 ელემენტთან ერთ დონეზე მყოფი ყველა div ელემენტის სტილიზაცია, მაშინ +-ს ნაცვლად უნდა გამოვიყენოთ სიმბოლო ~ :

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>სელექტორები</title> * <style> * h2~div { color: red; } * </style> * </head> * <body> * <h2>სათაური</h2> * <p>ანოტაცია</p> * <div> * <p>პირველი ბლოკის ტექსტი</p> * </div> * <div> * <p>მეორე ბლოკის ტექსტი</p> * </div> * </body> * </html> |
| --- |

