ფერი CSS-ში

CSS-ში ფართო გამოყენება აქვს ფერებს. ფერი ენიჭება ტექსტს, ფონს, ჩარჩოებს. მაგალითად, div ელემენტის ფონს მივანიჭოთ წითელი ფერი:

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

CSS-ში არის რამდენიმე თვისება, რომელსაც უნდა მიენიჭოს ფერი. ტექსტის ფერის მისანიჭებლად გამოიყენება თვისება color, ფონისთვის - border-color, ჩარჩოს ფერისთვის - border-color.

ფერის განსაზღვრისთვის რამდენიმე ხერხი არსებობს:

* თექვსმეტობითი რიცხვი - თექვსმეტობით სისტემაში ხდება წითელი, მწვანე და ლურჯი ფერის მნიშვნელობების კოდირება. მაგალითად, #1C4463. პირველი ორი სიმბოლო 1C აღნიშნავს წითელს, მომდევნო ორი - 44 მწვანეს, ბოლო ორი - 63 ლურჯს. საბოლოო ფერი მიიღება ამ ფერების შეზავებით. თუ ყველა წყვილი განმეორებადი სიმბოლოებისგან შედგება, შეიძლება მათი შემცირება თითომდე, მაგალითად #5522AA-ს ნაცვლად შეიძლება ჩავწეროთ #52A, #eeeeee-ს ნაცვლად - #eee. ამასთან, რეგისტრს მნიშვნელობა არა აქვს.
* RGB მნიშვნელობა - RGB წარმოადგენს წითელი, მწვანე და ლურჯი ფერების რიცხვით მნიშვნელობათა თანმიმდევრობას (Red - წითლი, Green - მწვანე, Blue - ლურჯი). რიცხვითი მნიშვნელობა შეიძლება იყოს გამოსახული პროცენტებში (0-დან 100 %-მდე) ან რიცხვებით (0-დან 255-მდე). მაგალითად:

| * background-color: rgb(100%,100%,100%); |
| --- |

აქ თითოეულ ფერს აქვს მნიშვნელობა 100%, შედეგად მივიღებთ თეთრ ფერს. თუ სამივე ფერი იქნებოდა 0%, მივიღებდით შავს.

უფრო ხშირად გამოიყენება არა პროცენტული, არამედ რიცხვითი მნიშვნელობები. მაგალითად:

| * background-color: rgb(28,68,99); |
| --- |

* RGBA მნიშვნელობა - ეს იგივეა, რაც RGB, რომელსაც დამატებული აქვს გამჭვირვალობის კოეფიციენტი (Alpha). ის ღებულობს მნიშვნელობას 0-დან (სრულიად გამჭვირვალე) 1-მდე (გაუმჭვირვალე). მაგალითად:

| * background-color: rgb(28, 68, 99, 0.6); |
| --- |

* HSL მნიშვნელობა - HSL წარმოადგენს აბრევიატურას: Hue - ტონი, Saturation - გაჯერებულობა, Lightness - განათებულობა. Hue არის კუთხე ფერების წრეში 0-დან 360 გრადუსამდე. Saturation მიუთითებს, რამდენად გაჯერებულია ფერი. იგი გამოისახება პროცენტებით 0-დან 100 %-მდე. Lightness განათებულობა ასევე პროცენტებში გამოისახება და ღებულობს მნიშვნელობას 0-დან (განათებულობის არარსებობა, ანუ შავი) 100%-მდე (მთლიანად განათებული, ანუ თეთრი). სუფთა ფერის მისაღებად გამოიყენება 50 %. მაგალითად:

| * background-color: hsl(206, 56%, 25%); |
| --- |

ეს ფერი წარმოადგენს #1C4463 და rgb(28, 68, 99) ფერების ეკვივალენტს.

* HSLA მნიშვნელობა - RGBA-ს ანალოგიურად მიიღება HSL-ისგან გამჭვირვალობის (Alpha)დამატებით. მაგალითად:

| * background-color: hsl(206, 56%, 25%, 0.6); |
| --- |

* ტექსტური მნიშვნელობა - ზოგიერთი ფერისთვის არსებობს ტექსტური მნიშვნელობა, მაგალითად red - წითელი, green - მწვანე, blue - ლურჯი, yellow - ყვითელი და ა. შ. მაგალითად:

| * background-color: darkblue; |
| --- |

darkblue - მუქი ლურჯი, წარმოადგენს #00008B-ს ეკვივალენტს.

გამჭვირვალობა

ფერების მნიშვნელობებში გათვალისწინებულია გამჭვირვალობა (RGBA, HSLA). თუმცა CSS-ში არის ცალკე თვისებაც, რომელიც განსაზღვრავს გამჭვირვალობას. ეს თვისებაა opacity. ის ღებულობს მნიშვნელობას 0-დან (სრულად გამჭვირვალე) 1-მდე (გაუმჭვირვალე). მაგალითად:

| * div{ * width: 100px; * height: 100px; * background-color: red; * opacity: 0.4; * } |
| --- |

შრიფტების სტილიზაცია

შრიფტების ოჯახები

თვისება font-family განსაზღვრავს შრიფტის ოჯახს. მაგალითად:

| * body{ * font-family: Arial; * } |
| --- |

მოცემულ შემთხვევაში გამოიყენება შრიფტი Arial.

ეს თვისება იმუშავებს მხოლოდ იმ შემთხვევაში, თუ მომხმარებელს კომპიუტერზე გააჩნია ასეთი შრიფტი. ამიტომ უპირატესად გამოიყენება ისეთი შრიფტები, რომლებიც ფართოდაა გავრცელებული, როგორიცაა Arial, Verdana და ა. შ. ასევე ხშირად ეთითება რამდენიმე შრიფტი. მაგალითად:

| * body{ * font-family: Arial, Verdana, Helvetica; * } |
| --- |

ამ შემთხვევაში ძირითადი შრიფტია Arial. თუ მომხმარებელს ეს შრიფტი არ აღმოაჩნდა, გამოიყენება მეორე - Verdana და ა. შ.

თუ შრიფტის დასახელება რამდენიმე სიტყვისგან შედგება, უნდა ჩავსვათ ბრჭყალებში:

| * body{ * font-family: "Times New Roman"; * } |
| --- |

კონკრეტული შრიფტების გარდა შეიძლება მიეთითოს საერთო უნივერსალური შრიფტები, რაც გამოისახება მნიშვნელობით sans-serif და serif:

| * body{ * font-family: Arial, Verdana, sans-serif; * } |
| --- |

შრიფტის სისქე

თვისება font-weight განსაზღვრავს შრიფტის სისქეს. ის იღებს 9 მნიშვნელობას: 100,200,300,...,900. 100 ძალიან წვრილიდან 900 ძალიან სქელამდე.

რეალობაში ძირითადად გამოიყენება ორი მნიშვნელობა: normal (ნორმალური, ჩვეულებრივი) და bold (ნახევრადსქელი):

| * font-weight: normal; * font-weight: bold; |
| --- |

შრიფტის დახრა

თვისება font-style საშუალებას გვაძლევს გამოვიყენოთ დახრილი შრიფტი. ამისთვის გამოიყენება მნიშვნელობა italic:

| * font-style: italic; |
| --- |

თუ გვინდა დახრის გაუქმება,გამოიყენება მნიშვნელობა normal:

| * font-style: normal; |
| --- |

შრიფტის ფერი

თვისება color განსაზღვრავს შრიფტის ფერს:

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

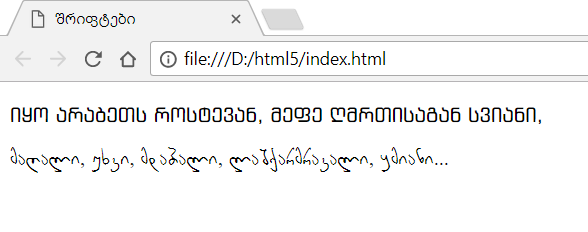
გარე შრიფტები

სტანდარტული შრიფტების გამოყენება ყოველთვის არ არის მისაღები. ხშირად საჭიროა რაღაც განსხვავებული შრიფტის გამოყენება, რომელიც არაა ფართოდ გავრცელებული და მომხმარებლის კომპიუტერზე შეიძლება არ აღმოჩნდეს. ასეთ შემთხვევაში გამოიყენება თვისება font-face:

| * @font-face { * font-family: 'unicode\_3d'; * src: url(3d\_unicode.ttf); * } |
| --- |

src ატრიბუტის მნიშვნელობა არის შრიფტის მისამართი. მაგალითად:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>სტილების კასკადურობა</title> * <style> * @font-face { * font-family: 'unicode\_3d'; * src: url(3d\_unicode.ttf); * } * @font-face { * font-family: 'bpg\_extramt'; * src: url(bpg\_extrasquare\_mtavruli\_2009.ttf); * } * .mtavr { * font-family: bpg\_extramt; * } * .daxr { * font-family: unicode\_3d; * } * </style> * </head> * <body> * <p class="mtavr">იყო არაბეთს როსტევან, მეფე ღმრთისაგან სვიანი,</p> * <p class="daxr">მაღალი, უხვი, მდაბალი, ლაშქარმრავალი, ყმიანი...</p> * </body> * </html> |
| --- |



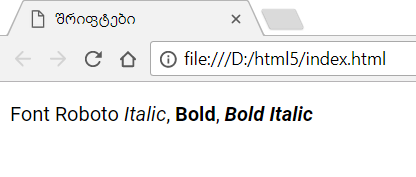
არსებობს შრიფტის სხვადასხვა ფორმატი: TrueType( გაფართოება ttf), Open Type (გაფართოება otf), Embedded Open Type (გაფართოება eot), Web Open Font Format (woff/woff2), Scalable Vector Graphic (svg). სხვადასხვა ბრაუზერს შეიძლება ჰქონდეს სხვადასხვა ტიპის შრიფტის მხარდაჭერა. ამ პრობლემის გადასაჭრელად, ხშირად შრიფტის შემქმნელები ერთი და იგივე შრიფტს ქმნიან სხვადასხვა ფორმატში და შესაძლებელია ერთდროულად რამდენიმე ფორმატის მითითება:

| * @font-face { * font-family:'FontAwesome'; * src: url('https://maxcdn.bootstrapcdn.com/font-awesome/4.6.1/fonts/fontawesome-webfont.eot'); * src: url('https://maxcdn.bootstrapcdn.com/font-awesome/4.6.1/fonts/fontawesome-webfont.eot?#iefix') format('embedded-opentype'), * url('https://maxcdn.bootstrapcdn.com/font-awesome/4.6.1/fonts/fontawesome-webfont.woff2') format('woff2'), * url('https://maxcdn.bootstrapcdn.com/font-awesome/4.6.1/fonts/fontawesome-webfont.woff') format('woff'), * url('https://maxcdn.bootstrapcdn.com/font-awesome/4.6.1/fonts/fontawesome-webfont.ttf') format('truetype'), * url('https://maxcdn.bootstrapcdn.com/font-awesome/4.6.1/fonts/fontawesome-webfont.svg') format('svg'); * } |
| --- |

შრიფტის სხვადასხვა ვერსია

ხანდახან შრიფტი შედგება არა ერთი, არამედ რამდენიმე ფაილისგან, მაგალითად შრიფტს ცალკე ფაილებში შეიძლება ჰქონდეს bold, italic, bold-italic და ა. შ. ვერსიები. ასეთ შემთხვევაში იქმნება რამდენიმე font-face და ფაილის მისამართთან ერთად ეთითება შრიფტის ტიპი. მაგალითად:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>შრიფტები</title> * <style> * @font-face { * font-family: 'Roboto'; * src: url(fonts/Roboto-Regular.ttf); * font-weight: normal; * font-style: normal; * } * @font-face { * font-family: 'Roboto'; * src: url(fonts/Roboto-Italic.ttf); * font-weight: normal; * font-style: italic; * } * @font-face { * font-family: 'Roboto'; * src: url(fonts/Roboto-Bold.ttf); * font-weight: bold; * font-style: normal; * } * @font-face { * font-family: 'Roboto'; * src: url(fonts/Roboto-BoldItalic.ttf); * font-weight: bold; * font-style: italic; * } * p{ * font-family: Roboto; * } * </style> * </head> * <body> * <p>Font Roboto <i>Italic</i>, <b>Bold</b>, <b><i>Bold Italic</i></b></p> * </body> * </html> |
| --- |



შრიფტის ზომა

შრიფტისთვის ზომის მისაცემად გამოიყენება თვისება font-size:

| * div{ * font-size: 18px; * } |
| --- |

მოცემულ მაგალითში შრიფტის ზიმა (სიმაღლე) არის 18 პიქსელი. პიქსელი წარმოადგენს ყველაზე ხშირად გამოყენებულ ზომის ერთეულს. ის არინიშნება "px"-ით, რომელიც იწერება რიცხვითი მნიშვნელობის შემდეგ.

თუ ტექსტის ზომას არ მივუთითებთ, ჩვეულებრივი ტექსტისთვის შრიფტის ზომა იქნება 16 პიქსელი. სხვადასხვა ბაზურ ელემენტს სხვადასხვა სტანდარტული შრიფტის ზომა გააჩნია. მაგალითად, თუ პარაგრაფისთვის (p) იგი 16 პიქსელის ტოლია, h1 სათაურისთვის ის 32 პიქსელია, h2-სთვის 24 პიქსელი და ა. შ.

შრიფტის ზომისთვის ასევე გამოიყენება სხვადასხვა ზომის ერთეულები.

შრიფტის ტექსტური ზომები

CSS-ში არის შემდეგი შრიფტის ზომები:

* medium: საბაზისო ზომა (16 პიქსელი)
* small: 13 პიქსელი
* x-small: 10 პიქსელი
* xx-small: 9 პიქსელი
* large: 18 პიქსელი
* x-large: 24 პიქსელი
* xx-large: 32 პიქსელი

მაგალითად:

| font-size: x-large; |
| --- |

შრიფტის ზომა პროცენტებით

პროცენტები საშუალებას გვაძლევს შრიფტის ზომა მივუთითოთ პროცენტებით საბაზისო ან მემკვიდრეობით მიღებული შრიფტის ზომისგან. მაგალითად:

| font-size: 150%; |
| --- |

ამ შემთხვევაში შრიფტის ზომა იქნება საბაზისო ზომის (16px) 150% ანუ 24 px.

შრიფტის ზომის პროცენტებით მითითებისას მემკვიდრეობითობამ შეიძლება შეცვალოს შრიფტის საბოლოო ზომა. მაგალითად:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>შრიფტები</title> * <style> * div {font-size: 10px;} * p {font-size: 150%;} * </style> * </head> * <body> * <div> * <p>შრიფტის ზომა მემკვიდრეობითობის გათვალისწინებით.</p> * </div> * </body> * </html> |
| --- |

მოცემულ მაგალითში პარაგრაფი p ელემენტისგან div მემკვიდრეობით იღებს შრიფტის ზომას 10 პიქსელს. შესაბამისად, შრიფტის საბაზისო ზომა მისთვის იქნება არა 16, არამედ 10 პიქსელი, საიდანაც დაითვლება p სელექტორში მითითებული 150 % და მივიღებთ საბოლოო შრიფტის ზომას 15 პიქსელს.

ზომის ერთეული em

ზომის ერთეული em პროცენტის მსგავსია. 1em არის 100%, 0.5em - 50% და ა. შ.

ტექსტის ფორმატირება

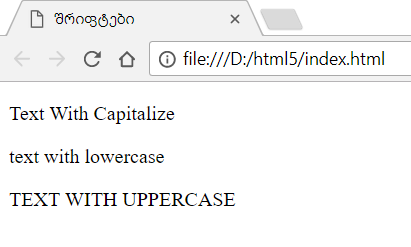
text-transform

თვისება text-transform ცვლის შრიფტის რეგისტრს. შეიძლება მიიღოს შემდეგი მნიშვნელობები:

* capitalize: სიტყვის პირველი ასო გადაჰყავს მაღალ რეგისტრში
* uppercase: მთლი სიტყვა გადაჰყავს მაღალ რეგისტრში
* lowercase: მთელი სიტყვა გადაჰყავს დაბალ რეგისტრში
* none: სიმბოლოების რეგისტრი არ იცვლება

მაგალითად:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>შრიფტები</title> * <style> * p.lowercase {text-transform: lowercase;} * p.uppercase {text-transform: uppercase;} * p.capitalize { text-transform: capitalize;} * </style> * </head> * <body> * <div> * <p class="capitalize">Text with capitalize</p> * <p class="lowercase">Text with lowercase</p> * <p class="uppercase">Text with uppercase</p> * </div> * </body> * </html> |
| --- |



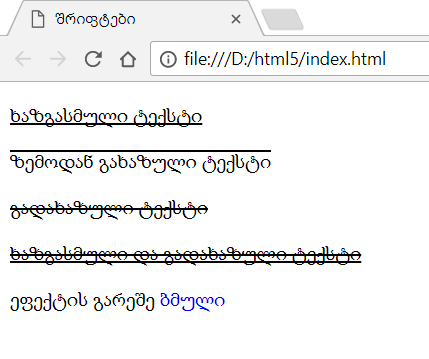
თვისება text-decoration

თვისება text-decoration უმატებს ტექსტს დამატებით ეფექტს. მან შეიძლება მიიღოს შემდეგი მნიშვნელობები:

* underline: ტექსტის ხაზგასმა (გახაზვა ქვემოდან)
* overline: ტექსტის გახაზვა ზემოდან
* line-through: ტექსტის გადახაზვა
* none: ტექსტზე არ გამოიყენება დეკორატიული ეფექტები

მაგალითად:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>შრიფტები</title> * <style> * p.under { * text-decoration: underline; * } * p.over { * text-decoration: overline; * } * p.line { * text-decoration: line-through; * } * p.mixed { * text-decoration: underline line-through; * } * a.none { * text-decoration: none; * } * </style> * </head> * <body> * <div> * <p class="under">ხაზგასმული ტექსტი</p> * <p class="over">ზემოდან გახაზული ტექსტი</p> * <p class="line">გადახაზული ტექსტი</p> * <p class="mixed">ხაზგასმული და გადახაზული ტექსტი</p> * <p>ეფექტის გარეშე <a href="index.php" class="none">ბმული<a></p> * </div> * </body> * </html> |
| --- |

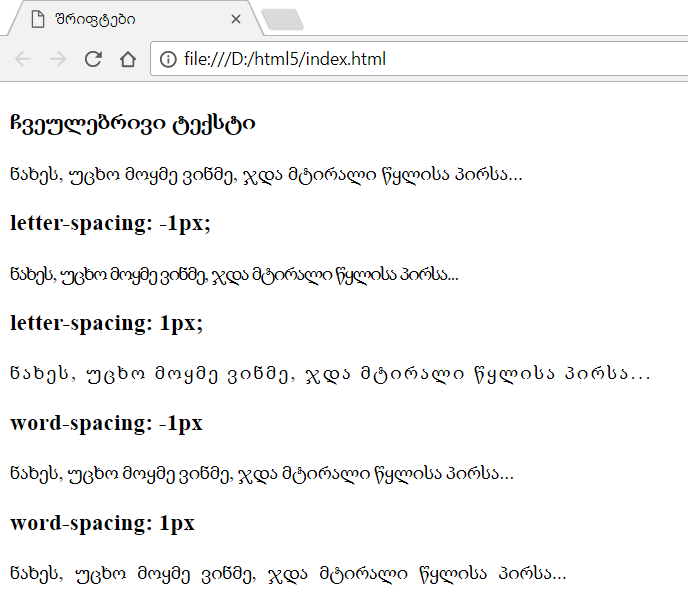


საჭიროების შემთხვევაში შესაძლებელია მნიშვნელობების კომბინირება. ბოლოსწინა სტრიქონში ერთდროულად გამოყენებულია ხაზგასმა და გადახაზვა.

დაშორება სიმბოლოებს შორის

CSS-ის ორი თვისება შესაძლებლობას გვაძლევს დავარეგულიროთ ასოებსა და სიტყვებს შორის დაშორებები. ასოებს შორის ინტერვალის დასარეგულირებლად გამოიყენება თვისება letter-spacing, ხოლო სიტყვებს შორის ინტერვალისთვის - word-spacing. მაგალითად:

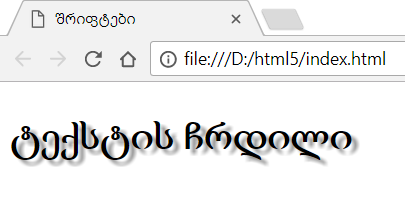
| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>შრიფტები</title> * <style> * p.smallLetterSpace { * letter-spacing: -1px; * } * p.bigLetterSpace { * letter-spacing: 2px; * } * p.smallWordSpace{ * word-spacing: -1px; * } * p.bigWordSpace{ * word-spacing: 5px; * } * </style> * </head> * <body> * <div> * <h3>ჩვეულებრივი ტექსტი</h3> * <p>ნახეს, უცხო მოყმე ვინმე, ჯდა მტირალი წყლისა პირსა...</p> * <h3>letter-spacing: -1px;</h3> * <p class="smallLetterSpace">ნახეს, უცხო მოყმე ვინმე, ჯდა მტირალი წყლისა პირსა...</p> * <h3>letter-spacing: 1px;</h3> * <p class="bigLetterSpace">ნახეს, უცხო მოყმე ვინმე, ჯდა მტირალი წყლისა პირსა...</p> * <h3>word-spacing: -1px</h3> * <p class="smallWordSpace">ნახეს, უცხო მოყმე ვინმე, ჯდა მტირალი წყლისა პირსა...</p> * <h3>word-spacing: 1px</h3> * <p class="bigWordSpace">ნახეს, უცხო მოყმე ვინმე, ჯდა მტირალი წყლისა პირსა...</p> * </div> * </body> * </html> |
| --- |



text-shadow

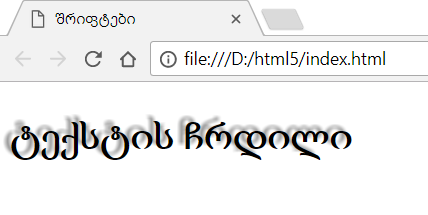
თვისება text-shadow -ის მეშვეობით შესაძლებელია ტექსტისთვის ჩრდილების მიცემა. მნიშვნელობა შედგება 4 პარამეტრისაგან: ჰორიზონტალური წანაცვლება, ვერტიკალური წანაცვლება, გაბნევის ხარისხი და ჩრდილის ფერი.მაგალითად:

| * h1{ * text-shadow: 5px 4px 3px #999; * } |
| --- |



მოცემულ მაგალითში ჩრდილის წანაცვლება ხდება მარჯვნივ 5 პიქსელით, ქვევით 4 პიქსელით, გაბნევადობის ხარისხია 3 პიქსელი და ჩრდილის ფერი #999. თუ ჩრდილის წანაცვლება გვინდა მარცხნივ ან ზევით, სესაბამისი მნიშვნელობები უნდა იყოს უარყოფითი. მაგალითად:

| * h1{ * text-shadow: -5px -4px 3px #999; * } |
| --- |



აბზაცების სტილიზაცია

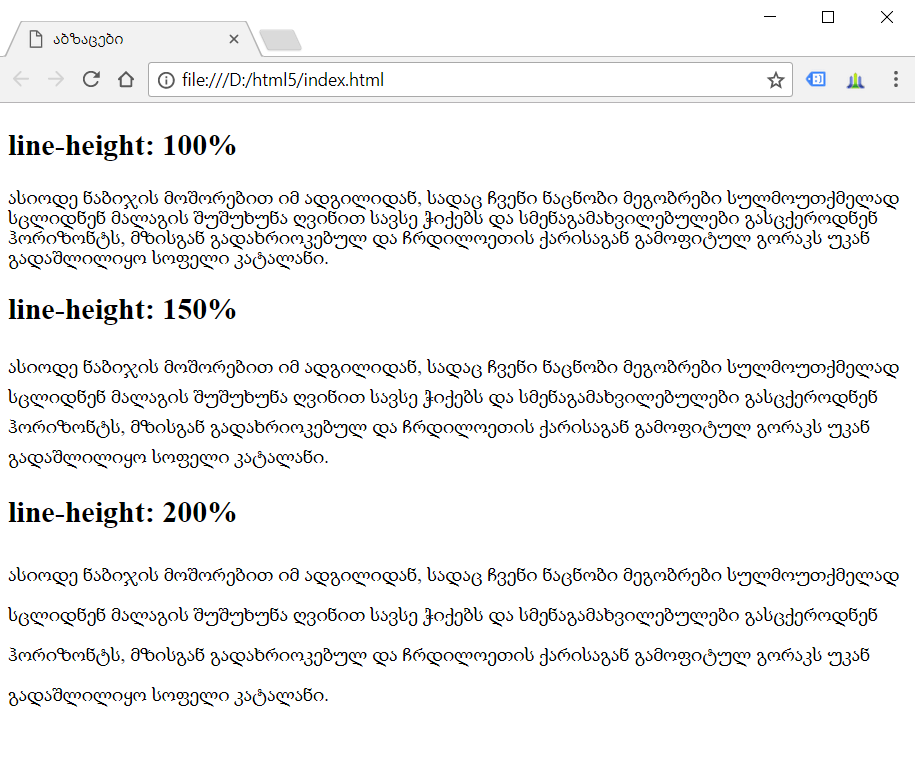
CSS-ის თვისებების ცალკე ჯგუფი ემსახურება ტექსტის დიდი ფრაგმენტების სტილიზაციას. მათი მეშვეობით შესაძლებელია სტრიქონებს შორის დაშორების შეცვლა, ტექსტის გასწორება კიდეებზე და ა. შ.

line-height

თვისების line-height დანიშნულებაა სტრიქონებს შორის დაშორების რეგულირება. მისი მნიშვნელობა გამოისახება პიქსელებით, პროცენტებით ან em ერთეულით. ძირითადად გამოიყენება პროცენტები და em. მაგალითად:

| * p{ * line-height: 150%; * } |
| --- |

თუ ეს თვისება არაა მინიჭებული, ამ შემთხვევაში გამოიყენება line-height:120%.



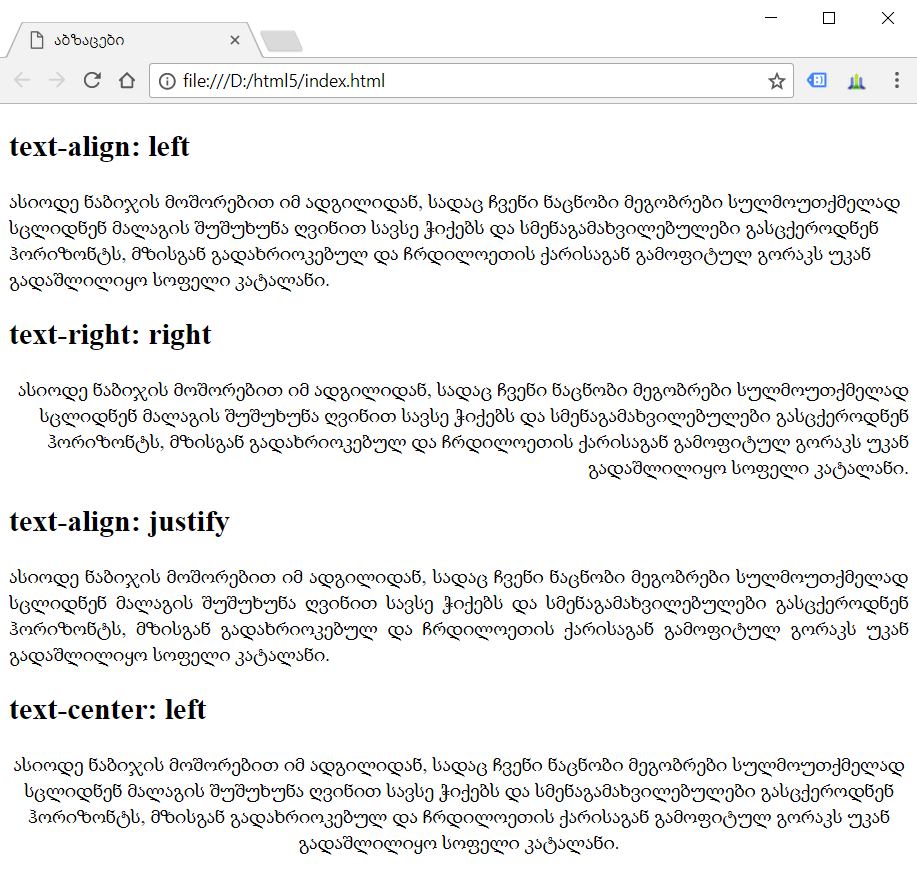
text-align

თვისება text-align ემსახურება ტექსტის გასწორებას გვერდის კიდეების მიმართ. ის ღებულობს შემდეგ მნიშვნელობებს:

* left: ტექსტის გასწორება მარცხნივ
* right: ტექსტის გასწორება მარჯვნივ
* justify: გასწორება ორივე კიდის მიმართ სიტყვებს შორის დაშორებების ხარჯზე
* center: გასწორება ცენტრში

მაგალითად:

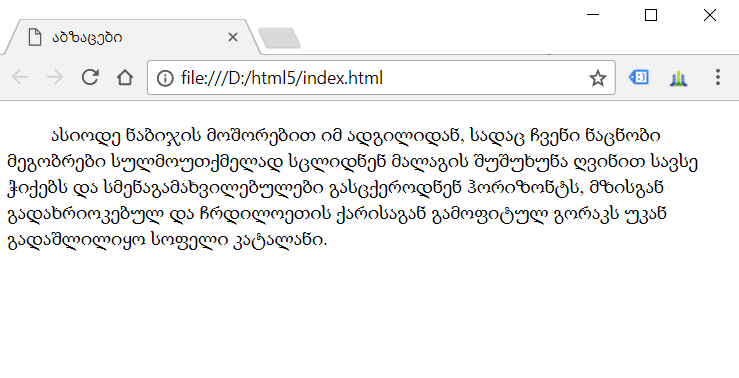
| * p{ * text-align: left; * } |
| --- |



text-indent

თვისება text-indent არეგულირებს აბზაცის პირველი სტრიქონის დაშორებას კიდიდან. მნიშვნელობისთვის გამოიყენება სტანდარტული ზომის ერთეულები: px, em:

| * p{ * text-indent: 35px; * } |
| --- |



სიების სტილიზაცია

list-style-type

* CSS-ში არის სპეციალური თვისებები სიების სტილიზაციისთვის. ერთ-ერთ ასეთ თვისებას წარმოადგენს list-style-type. მან შეიძლება მიიროს შემდეგი მნიშვნელობები დანომრილი სიებისთვის:
* decimal: ათწილადები, ათვლა იწყება 1-დან
* decimal-leading-zero: 0-ით დაწყებული ათწილადები: 01, 02, 03, ...,98,99
* lower-roman: დაბალი რეგისტრის რომაული რიცხვები: i, ii, iii, iv, v
* upper-roman: მაღალი რეგისტრის რომაული რიცხვები: I, II, III, IV, V…
* lower-alpha: დაბალი რეგისტრის ლათინური ასოები: a, b, c..., z
* upper-alpha: მაღალი რეგისტრის ლათინური ასოები: A, B, C, … Z
* georgian: ქართული ანბანის ასოები: ა,ბ,გ,...ჰ

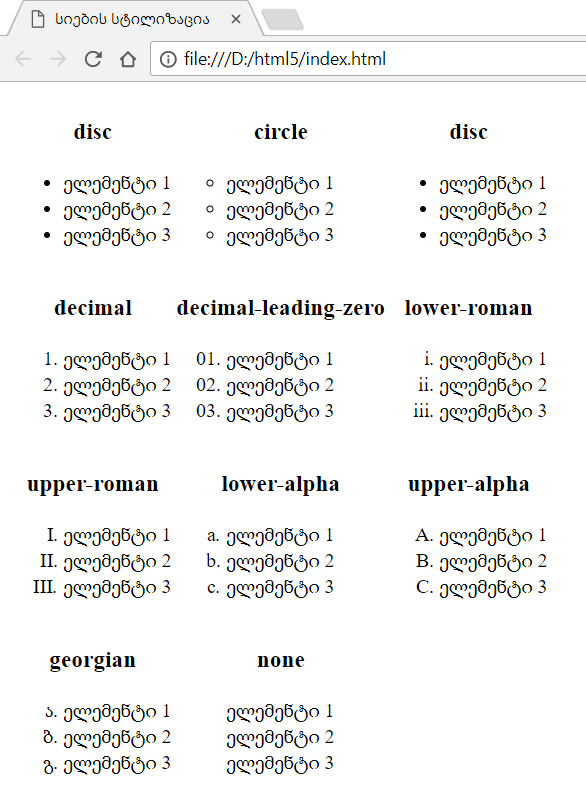
დაუნომრავი სიებისთვის:

* disc: შავი წრე
* circle: წარიელი წრეწირი
* square: შავი კვადრატი

როგორც დანომრილ, ასევე უნომრო სიებში მარკერის გამოსართავად გამოიყენება list-style-type:none.

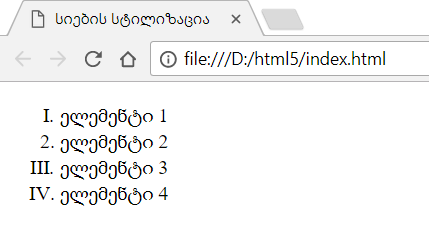
მაგალითად:

| * ul{ * list-style-type: square; * } * ol{ * list-style-type: georgian; * } |
| --- |



თვისება list-style-type შეიძლება გამოვიყენოთ როგორც მთლიანად სიის, ასევე სიის ცალკეული ელემენტის მიმართ:

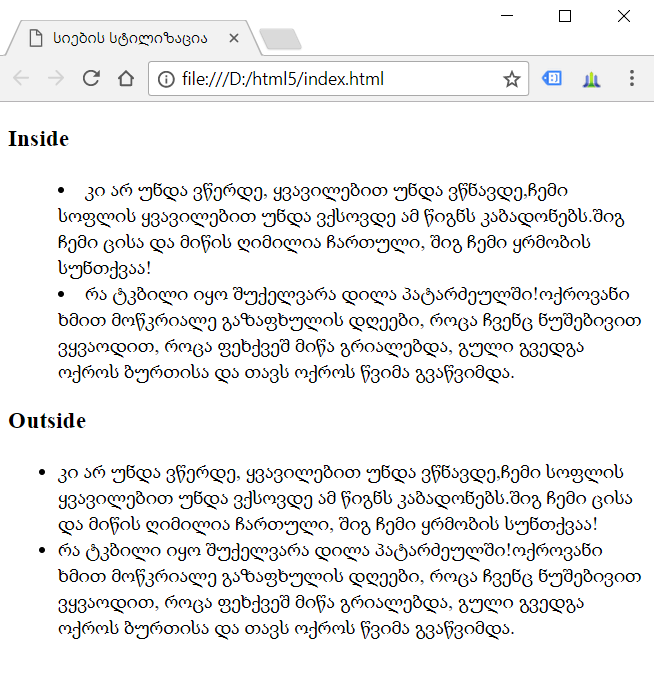
| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>სიების სტილიზაცია</title> * <style> * .decimal{ * list-style-type: decimal; * } * ol{ * list-style-type: upper-roman; * } * </style> * </head> * <body> * <ol> * <li>ელემენტი 1</li> * <li class="decimal">ელემენტი 2</li> * <li>ელემენტი 3</li> * <li>ელემენტი 4</li> * </ol> * </body> * </html> |
| --- |



list-style-position

list-style-position თვისების მეშვეობით შეგვიძლია ვცვალოთ სიის მარკერის პოზიცია. ღებულობს ორ მნიშვნელობას:

* outside: მარკერი გამოწეულია
* inside მარკერი შეწეულია

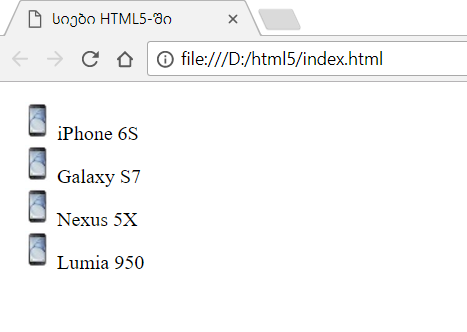


list-style-image

list-style-image თვისება საშუალებას გვაძლევს სიის მარკერად გამოვიყენოთ სურათი:

| * <ul style="list-style-image:url(phone.jpeg);"> * <li>iPhone 6S</li> * <li>Galaxy S7</li> * <li>Nexus 5X</li> * <li>Lumia 950</li> * </ul> |
| --- |

მნიშვნელობაში გადაეცემა იმ სურათის მისამართი, რომელიც უნდა გამოვიყენოთ მარკერად.



ცხრილების სტილიზაცია

CSS-ში განსაზღვრულია თვისებები, რომელთა მეშვეობით შესაძლებელია ცხრილების სტილიზაცია:

* border-collapse: მიმდებარე უჯრებს შორის ჩარჩოს სტილიზაცია
* border-spacing: მიმდებარე უჯრებს შორის არის სტილიზაცია
* caption-side: განსაზღვრავს სათაურის მდებარეობას
* empty-cells: სარიელი უჯრების დახატვის რეჟიმი
* table-layout: განსაზღვრავს ცხრილის ზომებს

ჩარჩოები

ცხრილის ჩარჩოებისთვის ადრე გამოიყენებოდა ატრიბუტი border, მაგალითად:

| * <table border="2px" > |
| --- |

მაგრამ დღეისათვის ტენდენციაა ყველანაირი სტილიზაცია მოხდეს CSS-ის სტილების მეშვეობით. ამისთვის CSS-ში არის თვისება border:

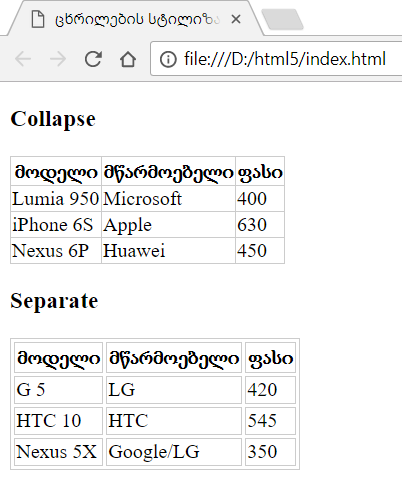
| * table { * border: 1px solid #ccc; /\* მთელი ცხრილის ჩარჩო \*/ * } * tr { * border: 1px solid #ccc; /\* ჩარჩო სტრიქონებს შორის \*/ * } * td, th { * border: 1px solid #ccc; /\* ჩარჩო სვეტებს შორის \*/ * } |
| --- |

მიმდებარე უჯრებს შორის ჩარჩოების სტილიზაციისთვის იყენებენ თვისებას border-collapse, რომელსაც აქვს შემდეგი მნიშვნელობები:

* collapse: მიმდებარე უჯრებს საერთო ჩარჩო აქვთ
* separate: მიმდებარე უჯრებს თავ-თავისი ჩარჩო აქვთ, რომელთა შორის სიცარიელეა

თუ მიმდებარე უჯრებს ცალ-ცალკე ჩარჩო აქვთ, border-spacing თვისების მეშვეობით შეიძლება ვცვალოთ ჩარჩოებს შორის დაშორება:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>ცხრილების სტილიზაცია</title> * <style> * table { * border: 1px solid #ccc; * border-spacing: 3px; * } * td, th{ * border: solid 1px #ccc; * } * .collapsed{ * border-collapse: collapse; * } * .separated{ * border-collapse: separate; * } * </style> * </head> * <body> * <h3>Collapse</h3> * <table class="collapsed"> * <tr><th>მოდელი</th><th>მწარმოებელი</th><th>ფასი</th></tr> * <tr><td>Lumia 950</td><td>Microsoft</td><td>400</td></tr> * <tr><td>iPhone 6S</td><td>Apple</td><td>630</td></tr> * <tr><td>Nexus 6P</td><td>Huawei</td><td>450</td></tr> * </table> * <h3>Separate</h3> * <table class="separated"> * <tr><th>მოდელი</th><th>მწარმოებელი</th><th>ფასი</th></tr> * <tr><td>G 5</td><td>LG</td><td>420</td></tr> * <tr><td>HTC 10</td><td>HTC</td><td>545</td></tr> * <tr><td>Nexus 5X</td><td>Google/LG</td><td>350</td></tr> * </table> * </body> * </html> |
| --- |

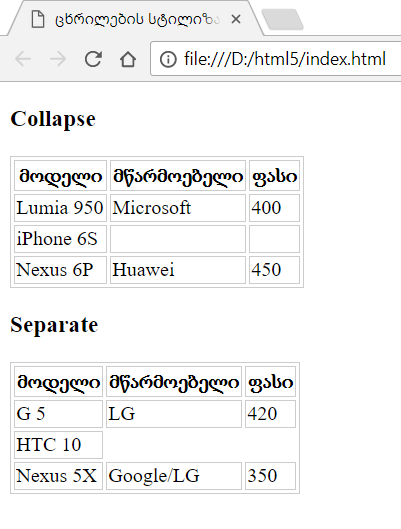


ცარიელი უჯრები

თვისება empty-cells განსაზღვრავს, გამოცნდეს ცარიელი უჯრები თუ არა. მისი მნიშვნელობებია:

* show: ცარიელი უჯრის გამოჩენა (მნიშვნელობა მიუთითებლობისას)
* hide: ცარიელი უჯრის დამალვა

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>ცხრილების სტილიზაცია</title> * <style> * table { * border: 1px solid #ccc; * border-spacing: 3px; * } * td, th{ * border: solid 1px #ccc; * } * .hidden-empty-cells{ * empty-cells: hide; * } * </style> * </head> * <body> * <h3>Collapse</h3> * <table> * <tr><th>მოდელი</th><th>მწარმოებელი</th><th>ფასი</th></tr> * <tr><td>Lumia 950</td><td>Microsoft</td><td>400</td></tr> * <tr><td>iPhone 6S</td><td></td><td></td></tr> * <tr><td>Nexus 6P</td><td>Huawei</td><td>450</td></tr> * </table> * <h3>Separate</h3> * <table class="hidden-empty-cells"> * <tr><th>მოდელი</th><th>მწარმოებელი</th><th>ფასი</th></tr> * <tr><td>G 5</td><td>LG</td><td>420</td></tr> * <tr><td>HTC 10</td><td></td><td></td></tr> * <tr><td>Nexus 5X</td><td>Google/LG</td><td>350</td></tr> * </table> * </body> * </html> |
| --- |



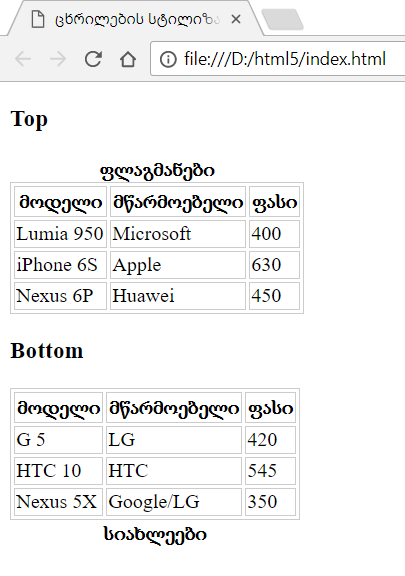
სათაურის პოზიცია

caption-side თვისების მეშვეობით შესაძლებელია ცხრილის სათაურის პოზიციის შეცვლა. ღებულობს შემდეგ მნიშვნელობებს:

top: ცხრილის ზემოთ (მნიშვნელობა მიუთითებლობისას)

bottom: ცხრილის ქვემოთ

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>ცხრილების სტილიზაცია</title> * <style> * table { * border: 1px solid #ccc; * border-spacing: 3px; * } * caption { * font-weight: bold; * } * td, th{ * border: solid 1px #ccc; * } * .captionBottom{ * caption-side: bottom; * } * </style> * </head> * <body> * <h3>Top</h3> * <table> * <caption>ფლაგმანები</caption> * <tr><th>მოდელი</th><th>მწარმოებელი</th><th>ფასი</th></tr> * <tr><td>Lumia 950</td><td>Microsoft</td><td>400</td></tr> * <tr><td>iPhone 6S</td><td>Apple</td><td>630</td></tr> * <tr><td>Nexus 6P</td><td>Huawei</td><td>450</td></tr> * </table> * <h3>Bottom</h3> * <table class="captionBottom"> * <caption>სიახლეები</caption> * <tr><th>მოდელი</th><th>მწარმოებელი</th><th>ფასი</th></tr> * <tr><td>G 5</td><td>LG</td><td>420</td></tr> * <tr><td>HTC 10</td><td>HTC</td><td>545</td></tr> * <tr><td>Nexus 5X</td><td>Google/LG</td><td>350</td></tr> * </table> * </body> * </html> |
| --- |



ცხრილის ზომები

table-layout თვისების მეშვეობით შესაძლებელია ცხრილის ზომების მართვა. მიუთითებლად მისი მნიშვნელობა არის auto, რომლის დროსაც სვეტის სიგანე ამ სვეტის ყველაზე გრზელი უჯრის სიგანის ტოლია. ცალკეული სვეტების სიგანეთა ჯამი შეადგენს ცხრილის სიგანეს. თუ table-layout თვისებას მივანიჭებთ მნიშვნელობას fixed, ცხრილს შეიძლება მივანიჭოთ ფიქსირებული სიგანე:

| * table { * border: 1px solid #ccc; * border-spacing: 3px; * table-layout: fixed; * width:350px; * } |
| --- |

უჯრების ტექსტის ვერტიკალური გასწორება

როგორც წესი, უჯრის შიგთავსი ვერტიკალურად შუაში თავსდება, თუმცა vertical-align თვისების მეშვეობით შეგვიძლია პოზიციის ცვლილება. მისი მნიშვნელობებია:

* top: გასწორება ზემოთ
* baseline: ტექსტის პირველი სტრიქონის გასწორება ზემოთ
* middle: გასწორება შუაში
* bottom: გასწორება ქვემოთ

თვისება vertical-align გამოიყენება მხოლოდ <th> და <td> ელემენტებისთვის:

| * td, th{ * border: solid 1px #ccc; * vertical-align: bottom; * height: 30px; |
| --- |