თვისება float

როგორც წესი, ვებგვერდის ყველა ბლოკი და ელემენტი ბრაუზერში გამოჩნდება იმ თანმიმდევრობით, რა თანმიმდევრობითაც არის ისინი html კოდში. მაგრამ თვისება float საშუალებას იძლევა მოვახდინოთ ობიექტების მიმაგრება კიდეებზე. იგი ღებულობს შემდეგ მნიშვნელობებს:

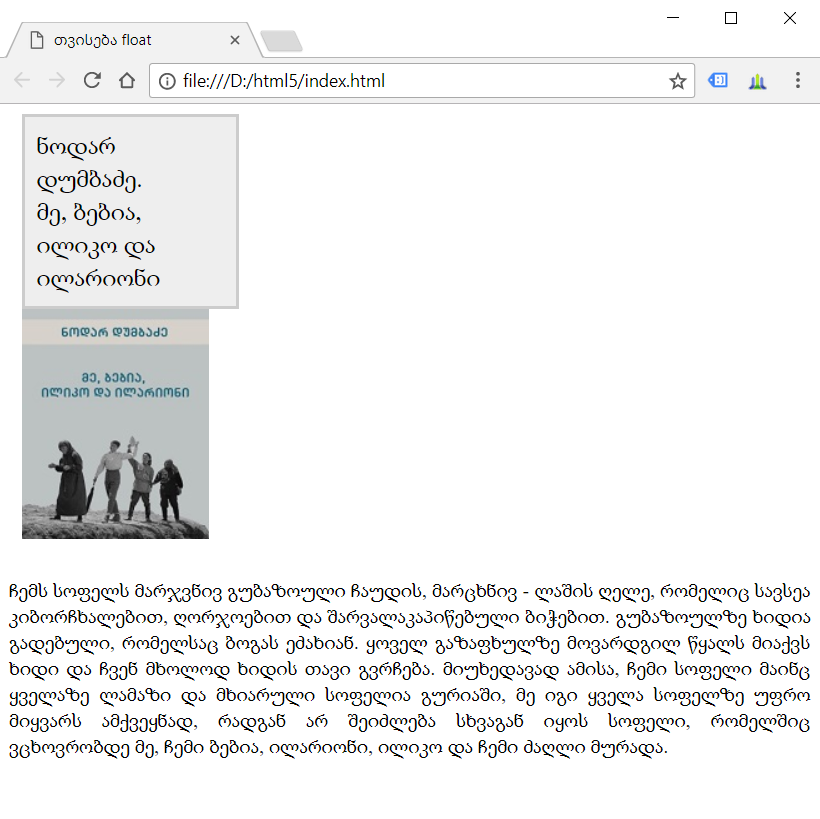
* left: ელემენტი გადაადგილდება მარცხნივ, ხოლო ყველა მისი მომდევნო ელემენტი გარსშემოედინება მას მრჯვნიდან
* right: ელემენტი გადაადგილდება მარჯვნივ
* none: აუქმებს გარსშემოდინებას და ობიექტი ბრუნდება ჩვეულებრივ პოზიციაში

float თვისების გამოყენებისას გარდა img ელემენტისა სასურველია გამოვიყენოთ თვისება width.

ვნახოთ float თვისების გამოყენება კონკრეტულ მაგალითზე. განვალაგოთ ერთი და იგივე ელემენტები float თვისების გამოყენების გარეშე და მისი გამოყენებით.

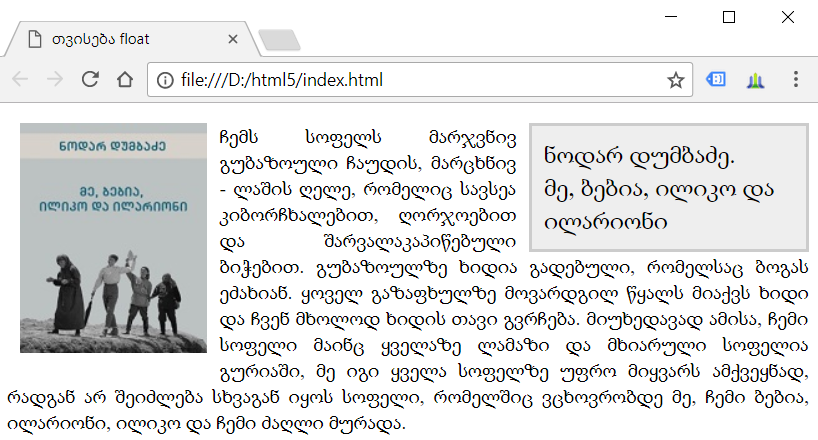
float თვისების გამოყენების გარეშე:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>თვისება float</title> * <style> * .image { * margin:10px; * margin-top:0px; * } * .sidebar{ * border: 2px solid #ccc; * background-color: #eee; * width: 150px; * padding: 10px; * margin-left:10px; * font-size: 20px; * } * p{ * text-align: justify; * } * </style> * </head> * <body> * <div> * <div class="sidebar">ნოდარ დუმბაძე.<br />მე, ბებია, ილიკო და ილარიონი</div> * <img src="cover.jpg" class="image" alt="მე, ბებია, ილიკო და ილარიონი" /> * <p>ჩემს სოფელს მარჯვნივ გუბაზოული ჩაუდის, მარცხნივ - ლაშის * ღელე, რომელიც სავსეა კიბორჩხალებით, ღორჯოებით და * შარვალაკაპიწებული ბიჭებით. გუბაზოულზე ხიდია გადებული, * რომელსაც ბოგას ეძახიან. ყოველ გაზაფხულზე მოვარდგილ * წყალს მიაქვს ხიდი და ჩვენ მხოლოდ ხიდის თავი გვრჩება. * მიუხედავად ამისა, ჩემი სოფელი მაინც ყველაზე ლამაზი და * მხიარული სოფელია გურიაში, მე იგი ყველა სოფელზე უფრო * მიყვარს ამქვეყნად, რადგან არ შეიძლება სხვაგან იყოს * სოფელი, რომელშიც ვცხოვრობდე მე, ჩემი ბებია, ილარიონი, * ილიკო და ჩემი ძაღლი მურადა.</p> * </div> * </body> * </html> |
| --- |



float თვისების გამოყენებით:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>თვისება float</title> * <style> * .image { * float:left; * margin:10px; * margin-top:0px; * } * .sidebar{ * border: 2px solid #ccc; * background-color: #eee; * width: 200px; * padding: 10px; * margin-left:10px; * font-size: 20px; * float: right; * } * p{ * text-align: justify; * } * </style> * </head> * <body> * <div> * <div class="sidebar">ნოდარ დუმბაძე.<br />მე, ბებია, ილიკო და ილარიონი</div> * <img src="cover.jpg" class="image" alt="მე, ბებია, ილიკო და ილარიონი" /> * <p>ჩემს სოფელს მარჯვნივ გუბაზოული ჩაუდის, მარცხნივ - ლაშის * ღელე, რომელიც სავსეა კიბორჩხალებით, ღორჯოებით და * შარვალაკაპიწებული ბიჭებით. გუბაზოულზე ხიდია გადებული, * რომელსაც ბოგას ეძახიან. ყოველ გაზაფხულზე მოვარდგილ * წყალს მიაქვს ხიდი და ჩვენ მხოლოდ ხიდის თავი გვრჩება. * მიუხედავად ამისა, ჩემი სოფელი მაინც ყველაზე ლამაზი და * მხიარული სოფელია გურიაში, მე იგი ყველა სოფელზე უფრო * მიყვარს ამქვეყნად, რადგან არ შეიძლება სხვაგან იყოს * სოფელი, რომელშიც ვცხოვრობდე მე, ჩემი ბებია, ილარიონი, * ილიკო და ჩემი ძაღლი მურადა.</p> * </div> * </body> * </html> |
| --- |



ელემენტებს, რომლებზეც გამოყენებულია თვისება float, მცურავ ელემენტებსაც (floating elements) ეძახიან.

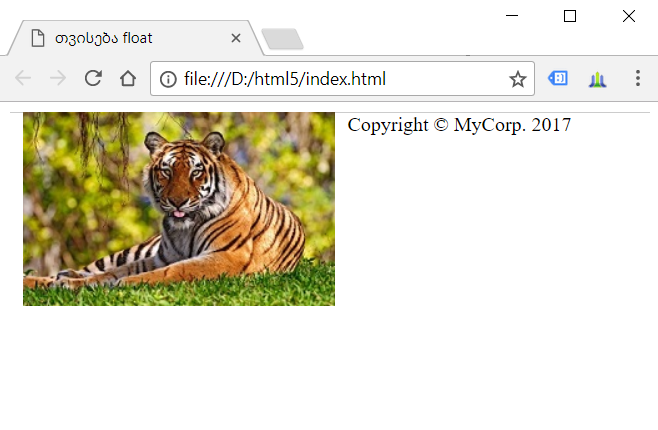
გარსშემოდინების აკრძალვა. თვისება clear

ხანდახან საჭიროა გარსშემოდინების აკრძალვა. ეს საჭირო ხდება იმ შემთხვევაში, როცა გვინდა მომდევნო ბლოკის გადატანა ახალ სტრიქონზე და არა მცურავი ელემენტის გარსშემოდინება.მაგალითად, ფუტერი უნდა იყოს მკაცრად გვერდის ბოლოში მთელ სიგანეზე და არ უნდა შემოედინებოდეს სხვა ელემენტებს. ამისთვის გამოიყენება თვისება clear. ის ღებულობს შემდეგ მნიშვნელობებს:

* left: ელემენტი გარსშემოედინება მცურავ ელემენტს მარჯვნიდან, ხოლო მარცხნიდან - არა
* right: ელემენტი გარსშემოედინება მცურავ ელემენტს მარცხნიდან, ხოლო მარჯვნიდან - არა
* both: ელემენტი გარსშემოედინება მცურავ ელემენტს, მაგრამ გადაადგილდება მის მიმართ ქვემოთ
* none: ელემენტი იქცევა სტანდარტულად, ანუ გარსშემოედინება მცურავ ელემენტებს

მაგალითად, ვთქვათ გვაქვს ვებგვერდი ფუტერით:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>თვისება float</title> * <style> * .image { * float:left; * margin:10px; * margin-top:0px; * } * .footer{ * border-top: 1px solid #ccc; * } * </style> * </head> * <body> * <img src="tiger.jpg" class="image" alt="ვეფხვი" /> * <div class="footer">Copyright © MyCorp. 2017</div> * </body> * </html> |
| --- |



თუ შევცვლით ფუტერის სტილს, იგი გადაინაცვლებს ქვევით:

| * .footer{ * border-top: 1px solid #ccc; * clear:both; * } |
| --- |

ელემენტების გადახვევა

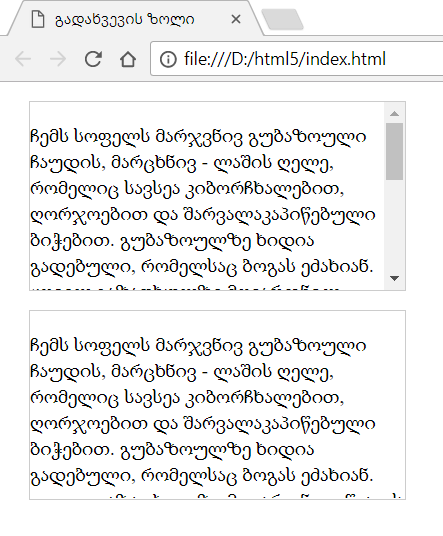
ვებგვერდის შექმნის დროს ხშირად არის სიტუაცია, როცა ელემენტის შიგთავსი უფრო დიდია, ვიდრე თვითონ ელემენტის ზომები. ასეთ დროს ბრაუზერი მაინც აჩენს შიგთავსს მთლიანად, მიუხედავად იმისა, რომ ის გამოდის ელემენტის საზღვრებს გარეთ.

თვისება overflow საშუალებას გვაძლევს დავამატოთ ასეთ დროს გადახვევა (scroll) ელემენტზე. ის ღებულობს შემდეგ მნიშვნელობებს:

* auto: თუ ელემენტის შიგთავსი (კონტენტი) გამოდის ელემენტის საზღვრებს გარეთ, ჩნდება გადახვევის ზოლი (ჰორიზონტალური, ვერტიკალური). სხვა დროს გადახვევის ზოლები არ ჩანს
* hidden: ჩანს შიგთავსის მხოლოდ ის ნაწილი, რაც ეტევა ელემენტის ზომებს შიგნით. გადახვევის ზოლები არ ჩნდება
* scroll: გადახვევის ზოლები ჩანს ყოველთვის, მაშინაც, როცა სიგთავსი ეტევა ელემენტის შიგნით
* visible: მნიშვნელობა მიუთითებლობისას. გადახვევის ზოლები არ ჩნდება, შიგთავსი ჩანს იმის მიუხედავად, ეტევა ელემენტის საზღვრებში თუ არა

მაგალითად:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>გადახვევის ზოლი</title> * <style> * .article1{ * width: 300px; * height: 150px; * margin:15px; * border: 1px solid #ccc; * overflow: auto; * } * .article2{ * width: 300px; * height: 150px; * margin:15px; * border: 1px solid #ccc; * overflow: hidden; * } * </style> * </head> * <body> * <div class="article1"> * <p>ჩემს სოფელს მარჯვნივ გუბაზოული ჩაუდის, მარცხნივ - ლაშის ღელე, რომელიც სავსეა კიბორჩხალებით, ღორჯოებით და * შარვალაკაპიწებული ბიჭებით. გუბაზოულზე ხიდია გადებული, რომელსაც ბოგას ეძახიან. ყოველ გაზაფხულზე მოვარდნილ * წყალს მიაქვს ხიდი და ჩვენ მხოლოდ ხიდის თავი გვრჩება. მიუხედავად ამისა, ჩემი სოფელი მაინც ყველაზე ლამაზი და * მხიარული სოფელია გურიაში, მე იგი ყველა სოფელზე უფრო მიყვარს ამქვეყნად, რადგან არ შეიძლება სხვაგან იყოს * სოფელი, რომელშიც ვცხოვრობდე მე, ჩემი ბებია, ილარიონი, ილიკო და ჩემი ძაღლი მურადა.</p> * </div> * <div class="article2"> * <p>ჩემს სოფელს მარჯვნივ გუბაზოული ჩაუდის, მარცხნივ - ლაშის ღელე, რომელიც სავსეა კიბორჩხალებით, ღორჯოებით და * შარვალაკაპიწებული ბიჭებით. გუბაზოულზე ხიდია გადებული, რომელსაც ბოგას ეძახიან. ყოველ გაზაფხულზე მოვარდნილ * წყალს მიაქვს ხიდი და ჩვენ მხოლოდ ხიდის თავი გვრჩება. მიუხედავად ამისა, ჩემი სოფელი მაინც ყველაზე ლამაზი და * მხიარული სოფელია გურიაში, მე იგი ყველა სოფელზე უფრო მიყვარს ამქვეყნად, რადგან არ შეიძლება სხვაგან იყოს * სოფელი, რომელშიც ვცხოვრობდე მე, ჩემი ბებია, ილარიონი, ილიკო და ჩემი ძაღლი მურადა.</p> * </div> * </body> * </html> |
| --- |



თვისება overflow მართავს ელემენტის გადახვევას როგორც ჰორიზონტალური, ასევე ვერტიკალური მიმართულებით. დამატებითი თვისებებით overflow-x და overflow-y შეიძლება ელემენტს ცალ-ცალკე განესაზღვროს გადახვევა ჰორიზონტალური და ვერტიკალური მიმართულებისთვის. ისინი ღებულობენ იმავე მნიშვნელობებს, რასაც overflow. მაგალითად:

| * overflow-x: auto; * overflow-y: hidden; |
| --- |

წრფივი გრადიენტი

გრადიენტი წარმოადგენს ერთი ფერის თანდათანობით გადასვლას მეორეში. CSS3-ში არსებობს ჩაშენებული გრადიენტები, რომლებიც შიძლება გამოვიყენოთ ელემენტის ფონად.

გრადიენტისთვის CSS-ში არაა ცალკე თვისება, იგი გამოისახება მნიშვნელობით, რომელიც უნდა მივანიჭოთ თვისებას background-image.

წრფივი გრადიენტი იწყება ელემენტის ერთი ბოლოდან და ვრცელდება მოპირდაპირე ბოლომდე, ამასთან ფერი თანდათანობით გადადის ერთიდან მეორეში.

წრფივი გრადიენტის შესაქმნელად უნდა მივუთითოთ დაწყების ადგილი და რამდენიმე ფერი:

| * background-image: linear-gradient(left,blue,red); |
| --- |

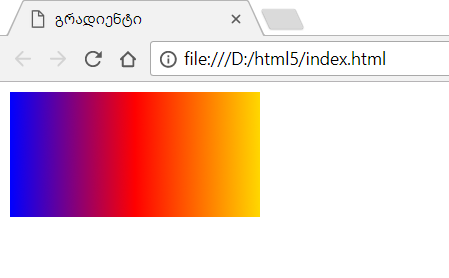
მოცემულ მაგალითში გრადიენტი იწყება ელემენტის მარცხენა ბოლოდან (left) ლურჯი ფერით (blue) და ვრცელდება მარჯვენა ბოლომდე, ამასთან ფერი თანდათანობით გადადის წითლში (red).

გრადიენტის გამოყენებას გააჩნია ერთი ნაკლი: ბრაუზერების მრავალფეროვნების გამო აუცილებელია მივუთითოთ შესაბამისი პრეფიქსი:

* -webkit- : ბრაუზერებისთვის Google Chrome, Safari, Microsoft Edge, Opera მე-15 ვერსიის ზევით
* -moz- : ბრაუზერისთვის Mozilla Firefox
* -o- : ბრაუზერისთვის Opera მე-15 ვერსიამდე

ბრაუზერების გათვალისწინებით გრადიენტის გამოყენება შემდეგნაირად გამოიყურება:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>გრადიენტი</title> * <style> * div { * width: 200px; * height: 100px; * background-image: linear-gradient(left,blue,red,gold); * background-image: -o-linear-gradient(left,blue,red,gold); * background-image: -moz-linear-gradient(left,blue,red,gold); * background-image: -webkit-linear-gradient(left,blue,red,gold); * } * </style> * </head> * <body> * <div></div> * </body> * </html> |
| --- |



გრადიენტის დაწყების ადგილის მისათითებლად გამოიყენება მნიშვნელობები:

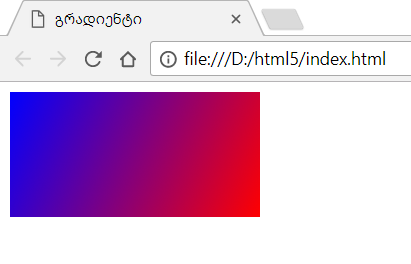
* left: მარცხნიდან მარჯვნივ
* right: მარჯვნიდან მარცხნივ
* top: ზემოდან ქვემოთ
* bottom: ქვემოდან ზემოთ

მაგალითად, გრადიენტი ქვემოდან ზემოთ:

| * background-image: linear-gradient(bottom,blue,red); |
| --- |

ასევე შეიძლება დიაგონალური გრადიენტის მიღება ორი მნიშვნელობის მითითებით:

| * background-image: linear-gradient(top left,blue,red); |
| --- |



მიმართულების განსაზღვრისთვის კონკრეტული მნიშვნელობების გარდა (top left) შეიძლება მიეთითოს კუთხის გრადუსული მნიშვნელობა 0-დან 360 გრადუსამდე. კუთხის რიცხვითი მნიშვნელობის მერე იწერება deg:

| * background-image: linear-gradient(30deg,blue,red); |
| --- |

მაგალითად, 0 გრადუსი ნიშნავს, რომ გრადიენტი იწყება მარცხენა ბოლოდან და მიემართება ჰორიზონტალურად მარჯვენა ბოლომდე, ხოლო 45 გრადუსის შემთხვევაში, გრადიენტი იწყება მარცხენა ქვედა კუთხეში და 45 გრადუსიანი კუთხით მიემართება მეორე ბოლოსკენ.

გრადიენტის დაწყების ადგილის მერე ეთითება ფერები. ფერი შეიძლება იყოს რამდენიმე. გრადიენტში თითოეული ფერი თანაბრად ნაწილლდება. თუმცა შესაძლებელია მიეთითოს კონკრეტული ფერის ადგილი. ამისთვის ფერის მერე ეთითება მეორე მნიშვნელობა, რომელიც განსაზღვრავს ამ ფერის ადგილს:

| * background-image: linear-gradient(left, #ccc, red 20%, red 80%, #ccc); |
| --- |

განმეორებადი გრადიენტი

repeating-linear-gradient-ის მეშვეობით შესაძლებელია განმეორებადი გრადიენტის შექმნა. მაგალითად:

| * background-image: repeating-linear-gradient(left, #ccc 20px, red 30px, rgba(0, 0, 126, .5) 40px); * background-image: -moz-repeating-linear-gradient(left, #ccc 20px, red 30px, rgba(0, 0, 126, .5) 40px); * background-image: -webkit-repeating-linear-gradient(left, #ccc 20px, red 30px, rgba(0, 0, 126, .5) 40px); |
| --- |

მოცემულ მაგალითში გრადიენტი იწყება მარცხნიდან ნაცრისფერი 20 პიქსელიანი ზოლით, შემდეგ 30 პიქსელამდე მიდის გადასვლა წითელში, შემდეგ 40 პიქსელამდე გადადის ღია ცისფერში. ამის შემდეგ ბრაუზერი იმეორებს ყველაფერს თავიდან ელემენტის ბოლომდე.

რადიალური გრადიენტი

წრფივი გრადიენტისგან განსხვავებით, რადიალური გრადიენტი ვრცელდება ცენტრიდან კიდეებისკენ წრიულად. რადიალური გრადიენტის შესაქმნელად საკმარისია მივუთითოთ ფერი, რომელიც იქნება ცენტრში და მეორე ფერი, რომელშიც თანაბრად გადავა. ეს ფერები გადაეცემა ფუნქციას radial-gradient. მაგალითად:

| * <!DOCTYPE html> * <html> * <head> * <meta charset="utf-8"> * <title>გრადიენტი</title> * <style> * div { * width: 200px; * height: 200px; * border-radius: 100px; * background-color: #eee; * background-image: radial-gradient(white, black); * background-image: -moz-radial-gradient(white, black); * background-image: -webkit-radial-gradient(white, black); * } * p{ * margin: 0; * padding-top: 60px; * text-align: center; * color: #eee; * } * </style> * </head> * <body> * <div><p>რადიალური გრადიენტი</p></div> * </body> * </html> |
| --- |



ისევე, როგორც წრფივი გრადიენტის შემთხვევაში, აქაც საჭიროა პრეფიქსების გამოყენება სხვადასხვა ბრაუზერების მხარდაჭერისათვის.

რადიალურ გრადიენტს შეიძლება ჰქონდეს ორი ფორმა: წრიული და ელიფსური. ელიფსური ფორმა წარმოადგენს გრადიენტის გავრცელებას ელიფსური სახით და ამისთვის მიეთითება მნიშვნელობა ellipse:

| * background-image: radial-gradient(ellipse, white, black); |
| --- |

ვინაიდან ეს მნიშვნელობა არის მნიშვნელობა მიუთითებლობისას, შესაძლებელია ის გამოვტოვოთ.

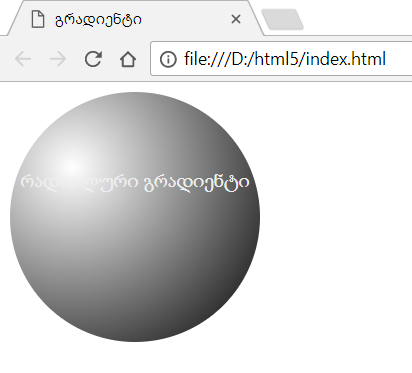
წრიული გრადიენტისთვის გამოიყენება circle:

| * background-image: radial-gradient(circle, white, black); |
| --- |

როგორც წესი, გრადიენტი იწყება ელემენტის ცენტრიდან, მაგრამ შესაბამისი პარამეტრებით შევცვალოთ დაწყების ადგილი, ამისათვის ეთითება პარამეტრი background-position:

| * background-image: radial-gradient(25% 30%, circle, white, black); |
| --- |

25% 30% ნიშნავს, რომ გრადიენტის ცენტრი მდებარეობს ელემენტის მარცხენა საზღვრიდან 25%-ის და ზედა საზღვრიდან 30%-ის დაშორებით.



სპეციალური დამატებითი პარამეტრების მეშვეობით შეიძლება შევცვალოთ გრადიენტის ზომა:

* closest-side: გრადიენტი ვრცელდება ცენტრიდან ელემენტის უახლოეს მხარემდე, ანუ გრადიენტი ელემენტის შიგნით რჩება
* closest-corner: გრადიენტის სიგქანე ითვლება ცენტრიდან უაზლოეს კუთხემდე. შესაბამისად, გრადიენტი შეიძლება გავიდეს ელემენტის საზღვრებს გარეთ
* farthest-side: გრადიენტი ვრცელდება ცენტრიდან ყველაზე დაშორებულ მხარემდე
* farthest-corner: გრადიენტის სიგანე ითვლება ცენტრიდან ყველაზე დაშორებულ კუთხემდე

| * background-image: radial-gradient(25% 30%, circle farthest-corner, white, black); |
| --- |