4.1 4.1 Teoria

Step 1

Gdybyśmy zobaczyli sieć w 1993 roku, uznalibyśmy, że jest to ponura sprawa według dzisiejszych standardów – każde tło było szare, a cały tekst czarny. Potem pojawił się `Netscape Navigator`, a wraz z nim garść atrybutów `HTML`, które pozwalały na podstawową (ale mile widzianą) kontrolę nad kolorami czcionek i tłami. Przez lata robiliśmy to. Ale na szczęście mamy teraz właściwości arkusza stylów, które odłożyły na bok te nieopisane atrybuty prezentacji. Istnieją dwa główne sposoby określania kolorów w arkuszach stylów — za pomocą predefiniowanej nazwy koloru, tak jak to robiliśmy do tej pory:

```
color: red;
color: olive;
color: blue;
```

Lub, częściej, z wartością liczbową opisującą konkretny kolor `RGB` (model kolorów na monitorach komputerowych).

```
color: #FF0000;
color: #808000;
color: #00F;
```

Za chwilę przejdziemy do wszystkich tajników kolorów RGB, ale najpierw krótka i słodka sekcja dotycząca standardowych nazw kolorów.

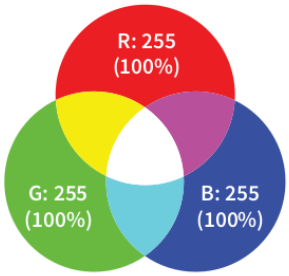
Najbardziej intuicyjnym sposobem określenia koloru **jest nazwanie go po nazwie**. Niestety, nie możemy wymyślić byle jakiej nazwy koloru i oczekiwać, że zadziała. Musi to być jedno ze słów kluczowych kolorów predefiniowanych w zaleceniu `CSS`. `CSS1` i `CSS2` przyjęły 16 standardowych nazw kolorów wprowadzonych pierwotnie w `HTML 4.01`. `CSS2.1` dodaje dodatkowych 17 kolorów. `CSS3` dodaje obsługę rozszerzonego zestawu 140 (raczej fantazyjnych) nazw kolorów. Teraz możemy określić nazwy takie jak `burlywood`, `peachpuff`, `oldlace`, i mój ulubiony, `papayawhip`! Rozszerzone kolory są pokazane poniżej, ale jeśli chcemy uzyskać dokładniejszy widok, warto wejść na stronę https://learningwebdesign.com/colornames.html (https://learningwebdesign.com/colornames.html). `CSS3` dodał również słowo kluczowe `transparent`, które może być używane z dowolną właściwością, która ma wartość koloru. Nazwy kolorów są łatwe w użyciu — wystarczy umieścić jedną na miejscu jako wartość dla dowolnej właściwości związanej z kolorem:

```
color: silver;
background-color: gray;
border-bottom-color: teal;
```

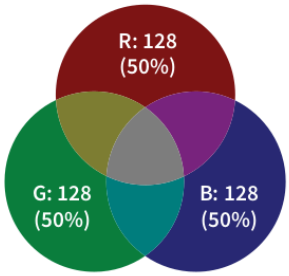
Nazwy są łatwe, ale jak widać, są ograniczone. Zdecydowanie najczęstszym sposobem określenia koloru jest jego wartość `RGB`. Daje również miliony kolorów do wyboru. Dla tych, którzy nie są zaznajomieni z tym, jak komputery radzą sobie z kolorami, zacznijmy od podstaw, zanim wskoczymy do składni `CSS`.

Komputery tworzą kolory widoczne na monitorze, łącząc trzy kolory światła: **czerwony, zielony** i **niebieski**. Jest to znane jako model kolorów `RGB`. Możemy podać receptury (rodzaju) kolorów, informując komputer, ile z każdego koloru ma się zmieszać. Ilość światła w każdym „kanale" koloru jest zwykle opisana **w skali od 0 (brak) do 255 (pełny rozbłysk)**, chociaż może być również podany w procentach. Im bliżej te trzy wartości zbliżają się do 255 (100%), tym bliżej wynikowy kolor zbliża się do bieli. **Każdy kolor, który widzimy na monitorze, można opisać ciągiem trzech liczb**: wartości czerwonej, wartości zielonej i wartości niebieskiej. Jest to jeden ze sposobów, w jaki edytory obrazów, takie jak `Adobe Photoshop`, śledzą kolory każdego piksela obrazu. Dzięki systemowi kolorów RGB kolor `pleasant lavender` można opisać jako `R:200`, `G:178`, `B:230`. Łącznie 255 kolorów w każdym kanale może zdefiniować około **16,7 miliona kombinacji kolorów**. Ta przestrzeń kolorów obejmująca miliony kolorów jest znana jako **Truecolor**. Istnieją różne sposoby kodowania tych kolorów (czyli konwertowania ich na bajty dla komputerów), a w Internecie używa się kodowania o nazwie `sRGB`.

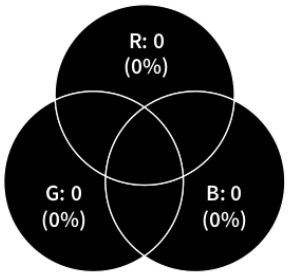
The RGB Color Model



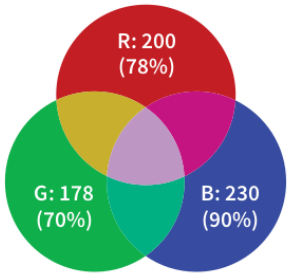
RGB: 255, 255, 255
white



RGB: 128, 128, 128
gray



RGB: 0, 0, 0
black



RGB: 200, 178, 230
pleasant lavender

Istnieje wiele sposobów na wybranie koloru i znalezienie jego wartości kolorów `RGB`. Jedną z szybkich i łatwych opcji jest przejście do google.com i wyszukanie „`color picker`" i voilà — w pełni funkcjonalnego selektora kolorów. `RGB` jest najczęstszym elementem projektowania stron internetowych, więc skupiamy naszą uwagę na tym. `HSL` (Hue Saturation Lightness or Luminosity) to kolejna opcja określania koloru w arkuszach stylów. `CMYK` (Cyan Magenta Yellow black) jest używany głównie do mediów drukowanych, więc nie będziemy go używać, chyba że do przetłumaczenia kolorów wydruku na ich odpowiedniki ekranowe.

`CSS` umożliwia określenie wartości kolorów `RGB` w wielu formatach. Wracając do `pleasant lavender`, moglibyśmy dodać ją do arkusza stylów, wymieniając każdą wartość w skali od 0 do 255:

```
color: rgb(200, 178, 230);
```

Możemy również podać je jako wartości procentowe, chociaż jest to mniej powszechne:

```
color: rgb(78%, 70%, 90%);
```

Możemy też podać sześciocyfrową wersję szesnastkową, którą widzieliśmy w selektorach kolorów. Te sześć cyfr reprezentuje te same trzy wartości `RGB`, z wyjątkiem tego, że zostały one przekonwertowane na odpowiedniki szesnastkowe (lub w skrócie szesnastkowe). Zauważmy, że szesnastkowe wartości `RGB` są poprzedzone symbolem `#` i nie wymagają notacji `rgb()` pokazanej w poprzednich przykładach. Mogą być pisane wielkimi lub małymi literami, ale zaleca się, aby być konsekwentnym:

```
color: #C8B2E6;
```

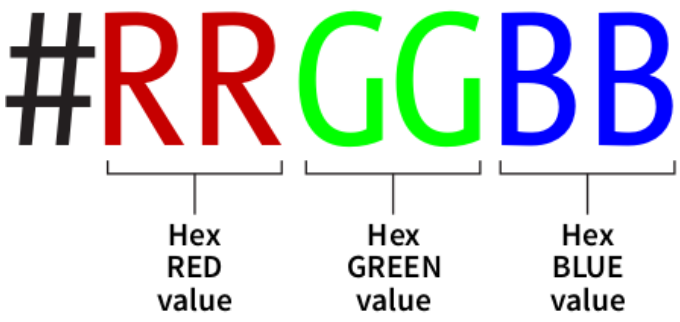
Jest jeszcze ostatni skrócony sposób określania szesnastkowych wartości kolorów. Jeśli wartość składa się z trzech par podwójnych cyfr lub liter, na przykład

```
color: #FFCC00; or color: #993366;
```

możemy skrócić każdą parę do jednej cyfry lub litery. Łatwiej jest pisać i czytać, a także nieznacznie zmniejsza rozmiar pliku. Te przykłady są równoważne z wymienionymi powyżej:

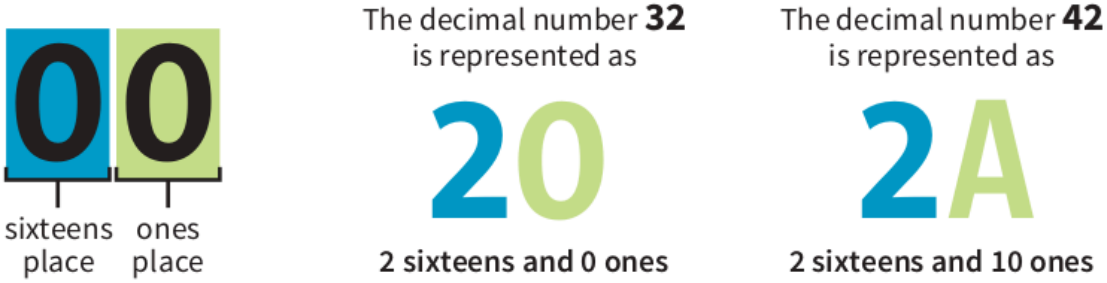
```
color: #FC0; or color: #936;
```

Czas wyjaśnić, co się dzieje z tym sześciocyfrowym ciągiem znaków. To, na co patrzymy, to w rzeczywistości seria trzech dwucyfrowych liczb, po jednej dla czerwonego, zielonego i niebieskiego. Ale zamiast dziesiętnego (podstawa 10, system, do którego jesteśmy przyzwyczajeni), te wartości są zapisywane w systemie szesnastkowym lub podstawie 16.



System szesnastkowy wykorzystuje 16 cyfr: `0–9` i `A–F` (do przedstawienia wielkości 10–15). System szesnastkowy jest szeroko stosowany w informatyce, ponieważ zmniejsza przestrzeń potrzebną do przechowywania niektórych informacji. Na przykład wartości `RGB` są zmniejszane z trzech do dwóch cyfr po przekonwertowaniu na szesnastkowe. Teraz, gdy większość oprogramowania graficznego i oprogramowania do tworzenia stron internetowych zapewnia łatwy dostęp do szesnastkowych wartości kolorów, nie ma zbyt wiele potrzeby samodzielnego tłumaczenia wartości `RGB` na szesnastkowe, jak to robiliśmy w dawnych czasach . Jeśli zajdzie taka potrzeba, dostępnych jest wiele konwerterów dziesiętnych na szesnastkowe online.

Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

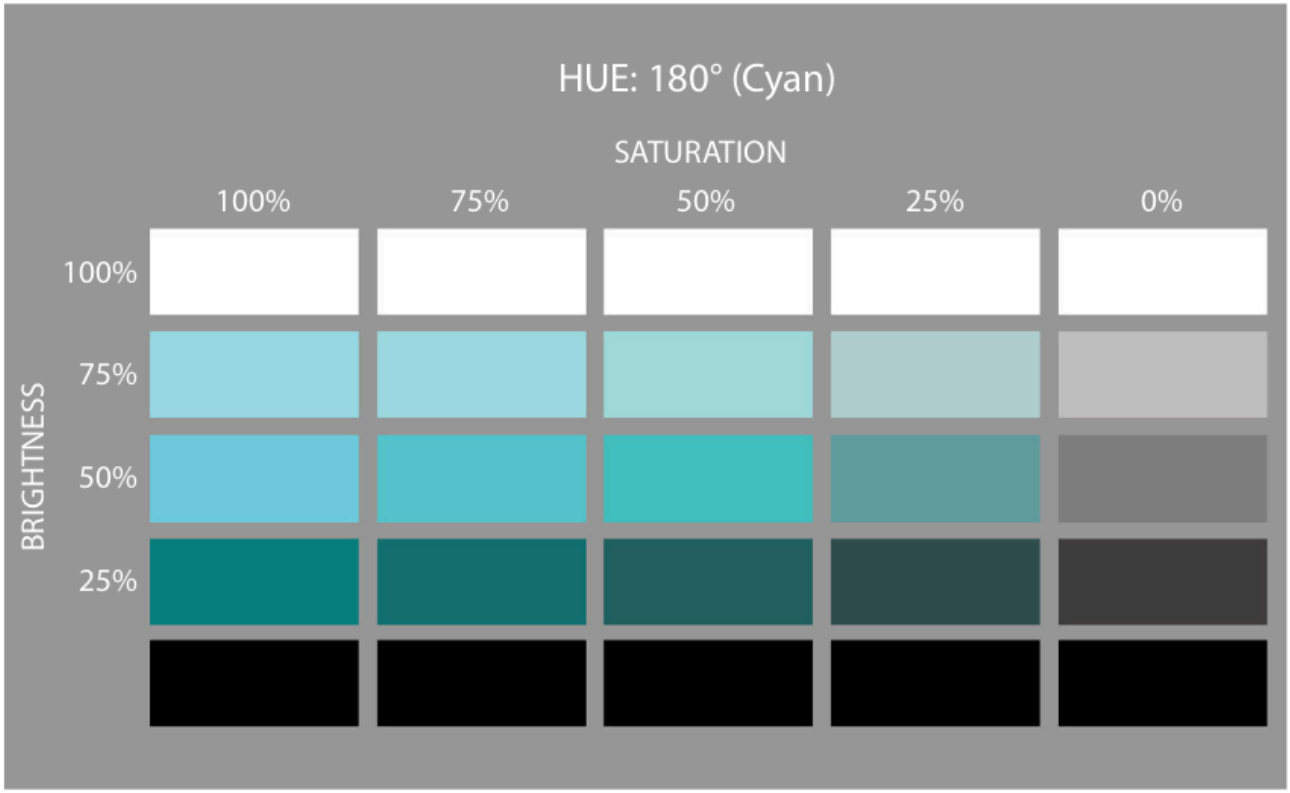


Kolor `RGBa` pozwala określić kolor i uczynić go tak przezroczystym lub nieprzezroczystym, jak chcesz. Litera „ `a` ” w „ `RGBa` ” oznacza alfa, czyli dodatkowy kanał, który kontroluje poziom przezroczystości w skali od 0 (w pełni przezroczysty) do 1 (w pełni nieprzezroczysty). Oto jak to wygląda napisane zgodnie z regułą stylu:

```
color: rgba(0, 0, 0, .5);
```

Pierwsze trzy wartości w nawiasach to zwykłe stare wartości `RGB` , w tym przypadku tworzące kolor czarny. Czwarta wartość, `0.5` , to poziom przezroczystości. Więc ten kolor jest czarny z 50% przejrzystością. Pozwala to na lekkie prześwitywanie innych kolorów lub wzorów tła.

`CSS3` wprowadził możliwość określania kolorów według ich wartości `HSL` : **Barwa**, **Nasycenie** i **Jasność**. W tym systemie kolory są rozłożone wokół koła w kolejności tęczy, z czerwienią u góry (na godzinie 12). Wartości **barwy** są następnie mierzone w stopniach wokół okręgu: czerwony przy `0°/360°` , zielony przy `120°` i niebieski przy `240°` , z innymi kolorami pomiędzy. **Nasycenie** to wartość procentowa od `0%` (szary) do `100%` (kolor przy pełnym nadmuchu). **Jasność** to również wartość procentowa od `0%` (najciemniejszy) do `100%` (najjaśniejszy). Rysunek poniżej pokazuje jeden odcień, `cyan` (umieszczony pod kątem 180° na kole) z powiązanymi poziomami nasycenia i jasności. Możemy zobaczyć, dlaczego niektórzy ludzie uważają ten system za bardziej intuicyjny w użyciu, ponieważ po zablokowaniu odcienia łatwo jest go wzmocnić, przyciemnić lub rozjaśnić, zwiększając lub zmniejszając wartości procentowe. Wartości `RGB` nie są w ogóle intuicyjne, chociaż niektórzy doświadczeni projektanci wyczuwają je.



W `CSS` wartości `HSL` są podawane jako wartość odcienia i dwa procenty. Nigdy nie są konwertowane na wartości szesnastkowe, jak to ma miejsce w przypadku `RGB` .

```
color: hsl(265, 51%, 80%);
```

Podobnie jak w przypadku `RGB` , możemy dodać kanał alfa, aby ustawić przezroczystość kolorów `HSL` , czego wynikiem będzie model kolorów `HSL` . W przypadku `RGBa` czwartą wartością jest stopień przezroczystości w skali od 0 (w pełni przezroczysty) do 1 (w pełni nieprzezroczysty). W tym przykładzie określono kolor, który jest nieprzezroczysty w 65%:

```
color: hsla(70, 60%, 58%, .65);
```

Step 2

Teraz, gdy wiemy, jak zapisywać wartości kolorów, przejdźmy do właściwości związanych z kolorami. Możemy określić kolory pierwszego planu i tła dla dowolnego elementu `HTML` .

Pierwszy plan elementu (`foreground`) składa się z jego tekstu i obramowania (jeśli jest określony). Kolor `foreground` określamy za pomocą właściwości `color` . Oto szczegóły dotyczące właściwości `color` jeszcze raz.

```
color
Wartości: wartość koloru (nazwa lub numeryczna)
Domyślny: zależy od przeglądarki i preferencji użytkownika
Dotyczy: wszystkich elementów
Dziedziczy: tak
```

W poniższym przykładzie pierwszy plan elementu cytatu blokowego jest ustawiony na zielony z nazwą koloru. Jak widać, zastosowanie właściwości color do elementu blockquote oznacza, że kolor jest dziedziczony przez elementy `p` i `em` , które zawiera poniższy rysunek. Gruba przerywana ramka wokół całego cytatu blokowego jest również zielona; gdybyśmy jednak zastosowali właściwość `border-color` do tego samego elementu, ten kolor zastąpiłby ustawienie zielonego pierwszego planu.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        blockquote {
            border: 4px dashed;
            color: green;
        }
    </style>
</head>
<body>
<blockquote>
    To be or not to be
</blockquote>
</body>
</html>
```

Właściwość `background-color` służy do określania koloru tła do dowolnego elementu.

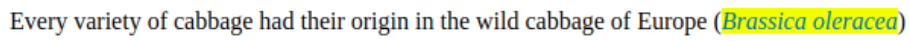
```
background-color
Wartości: wartość koloru (nazwa lub numeryczna) | przezroczysty
Domyślnie: przezroczysty
Dotyczy: wszystkich elementów
Dziedziczy: nie
```

Właściwość `background-color` wypełnia płótno za elementem, który zawiera obszar zawartości, oraz wszelkie dopełnienie (dodatkową przestrzeń) dodaną wokół zawartości, rozciągające się za obramowaniem aż do jej zewnętrznej krawędzi. Zobaczmy, co się stanie, gdy użyjemy właściwości `background-color`, aby tło tego samego przykładowego cytatu blokowego było jasnozielone.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        blockquote {
            border: 4px dashed;
            color: green;
            background-color: #c6de89;
        }
    </style>
</head>
<body>
<blockquote>
    To be or not to be
</blockquote>
</body>
</html>
```

Zgodnie z oczekiwaniami kolor tła wypełnia obszar za tekstem, aż do granicy. Przyjrzyjmy się uważnie przerwom w obramowaniu, a zobaczymy, że kolor tła przechodzi na zewnętrzną krawędź. Ale na tym kończy się tło; jeśli zastosujemy margines wokół tego elementu, tło nie wejdzie w margines. Ponownie przyjrzymy się wszystkim tym komponentom elementu, gdy będziemy mówić o modelu pudełkowym `CSS`. Domyślnie, jeśli ramka elementu ma luki, tło będzie widoczne. Warto zauważyć, że kolory tła nie dziedziczą, ale ponieważ domyślne ustawienie tła dla wszystkich elementów jest przezroczyste , kolor tła rodzica jest widoczny przez elementy potomne. Na przykład, możemy zmienić kolor tła całej strony, stosując właściwość `background-color` do elementu `body`, a kolor będzie widoczny we wszystkich elementach na stronie. Oprócz ustawienia koloru całej strony, możemy zmienić kolor tła dowolnego elementu, zarówno na poziomie bloku (jak cytat blokowy pokazany w poprzednim przykładzie), jak i w wierszu. W kolejnym przykładzie użyto właściwości `color` i `background-color`, aby wyróżnić słowo oznaczone jako termin „słowniczek”.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        .glossary {
            color: #0378a9; /* blue */
            background-color: yellow;
        }
    </style>
</head>
<body>
<p>Every variety of cabbage had their origin in the wild cabbage of
    Europe (<dfn class="glossary"><i>Brassica oleracea</i></dfn>)</p>
</body>
</html>
```

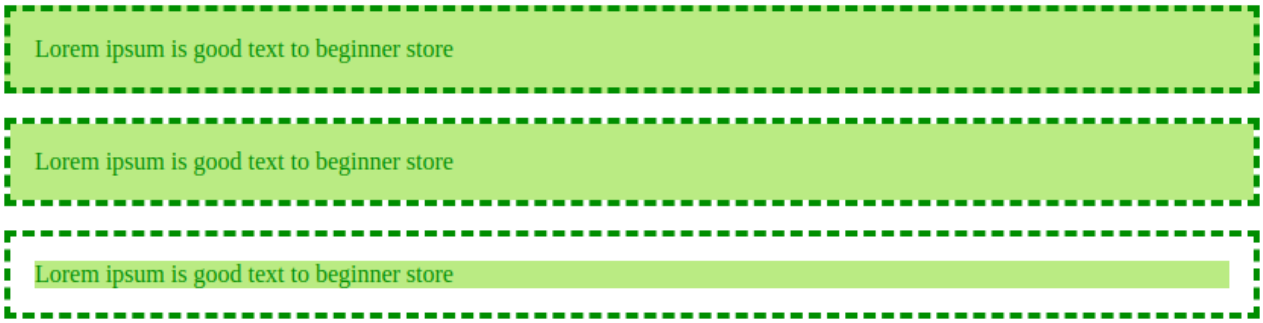


Tradycyjnie obszar malowania tła (obszar, na którym nakładane są kolory wypełnienia) elementu rozciąga się aż do zewnętrznej krawędzi obramowania, jak widzieliśmy poprzednio . `CSS3` wprowadził właściwość `background-clip` , aby dać projektantom większą kontrolę nad tym, gdzie zaczyna się i kończy obszar malowania.

```
background-clip
Wartości: border-box | padding-box | zawartość-box
Domyślnie: border-box
Dotyczy: wszystkich elementów
Dziedziczy: nie
```

Domyślna wartość `border-box` rysuje obszar malowania do zewnętrznej krawędzi obramowania, jak widzieliśmy. Przykład poniżej pokazuje, że `padding-box` rozpoczyna obszar malowania na zewnętrznej krawędzi obszaru `padding` dla elementu (i na wewnętrznej krawędzi obramowania). Wreszcie `content-box` umożliwia tłu wypełnienie tylko obszaru zawartości elementu.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        blockquote {
            padding: 1em;
            border: 4px dashed;
            color: green;
            background-color: #C6DE89;
        }
    </style>
</head>
<body>
    <blockquote style="background-clip: border-box">
        Lorem ipsum is good text to beginner store
    </blockquote>
    <blockquote style="background-clip: padding-box">
        Lorem ipsum is good text to beginner store
    </blockquote>
    <blockquote style="background-clip: content-box">
        Lorem ipsum is good text to beginner store
    </blockquote>
</body>
</html>
```

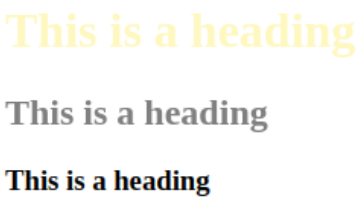


Wcześniej mówiliśmy o formacie kolorów `RGBa`, który dodaje pewien poziom przezroczystości, gdy jest stosowany do koloru lub tła. Jest jednak inny sposób na sprawienie, by element był nieco przezroczysty — właściwość `opacity` `CSS3`.

```
opacity
Wartości: liczba (0 do 1)
Domyślnie: 1
Dotyczy: wszystkich elementów
Dziedziczy: nie
```

Wartość `opacity` to liczba od 0 (całkowicie przezroczysta) do 1 (całkowicie nieprzezroczysta). Wartość 0,5 daje elementowi krycie 50%. Ustawienie `opacity` dotyczy całego elementu — zarówno pierwszego planu, jak i tła (jeśli zostało ustawione). Jeśli chcemy wpłynąć tylko na jedno lub drugie, należy użyć zamiast tego wartości koloru `RGBa`. W poniższym przykładzie nagłówkowi nadano kolor `gold`, a tłu `white`. Gdy właściwość `opacity` jest ustawiona, niebieskie tło strony jest widoczne zarówno przez tekst, jak i przez pole elementu.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
      h1 {
          color:gold;
          background:white;
      }
    </style>
</head>
<body>
    <h1 style="opacity: .25">This is a heading</h1>
    <h2 style="opacity: .5">This is a heading</h2>
    <h3 style="opacity: 1">This is a heading</h3>
</body>
</html>
```



Step 3

Łącze (link) często ma jeden kolor po kliknięciu i inny kolor, gdy wracamy do tej strony? Dzieje się tak, ponieważ za kulisami przeglądarka śledzi, które linki zostały kliknięte (lub „odwiedzone”, aby użyć żargonu). Przeglądarka śledzi również inne stany, takie jak to, czy kursor użytkownika znajduje się nad elementem (stan najechania), czy element jest pierwszym tego typu, czy jest pierwszym czy ostatnim dzieckiem swojego rodzica i czy element formularza został zaznaczony lub wyłączony, żeby wymienić tylko kilka. W `CSS` możemy zastosować style do elementów w tych stanach, używając specjalnego rodzaju selektora zwanego **selektorem pseudoklasowym**. To dziwna nazwa, ale możemy myśleć o niej tak, jakby elementy w określonym stanie należały do tej samej klasy. Jednak nazwy klasy nie ma w znacznikach — jest to coś, co przeglądarka po prostu śledzi. Więc to jest trochę jak klasa… to

pseudoklasa. **Selektory pseudoklas są oznaczone znakiem dwukropka** (`:`). Zwykle następują bezpośrednio po nazwie elementu — na przykład `li:first-child` . Istnieje sporo pseudoklas w `CSS3` , a `W3C` trochę zwariowało w module selektora `CSS Level 4` , krążąc wokół nowych pseudoklas, z których większość nie obsługuje przeglądarek w chwili pisania tego tekstu.

Najbardziej podstawowe selektory pseudoklas ukierunkowują linki (elementy) na podstawie tego, czy zostały kliknięte. Pseudoklasy linków są rodzajem pseudoklas dynamicznych, ponieważ są stosowane w wyniku interakcji użytkownika ze stroną, a nie w wyniku działania znaczników.

```
:link Stosuje styl do nieklikniętych (nieodwiedzonych) linków
:visited Stosuje styl do linków, które już zostały kliknięte
```

Domyślnie przeglądarki zazwyczaj wyświetlają tekst linków w kolorze niebieskim, a linki, które zostały kliknięte, w kolorze fioletowym, ale można to zmienić za pomocą kilku reguł stylu. Istnieją ograniczenia dotyczące właściwości, które można zastosować do linków `:visited` . W tych przykładach zmieniono kolor nieklikniętych linków na bordowy, a odwiedzonych na szary. Często odwiedzane linki mają bardziej stonowany kolor niż nieklikniętе linki:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        a:link {
            color: maroon;
        }
        a:visited {
            color: gray;
        }
    </style>
</head>
<body>
    <a href="https://google.pl">Google.pl</a>
    <a href="https://duckduckgo.pl">DuckDuckGo.pl</a>
</body>
</html>
```

Google.pl DuckDuckGo.pl

Przeglądarka wizualnie podkreśla element formularza po jego wybraniu. Kiedy element jest podświetlony i gotowy do wprowadzenia, mówi się, że ma `„focus”` . Selektor `:focus` pozwala zastosować niestandardowe style do elementów, gdy są one w stanie skupienia.

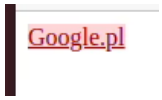
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        input:focus {
            background-color: yellow;
        }
    </style>
</head>
<body>
    <form>
        <label>Name: <input type="text"></label>
        <input type="reset">
        <input type="submit">
    </form>
</body>
</html>
```

Name: daddasdada Resetuj Prześlij

Ciekawym jest selektor `:hover` . Celuje w elementy, gdy wskaźnik myszy użytkownika znajduje się bezpośrednio nad nimi. Możemy użyć stanu `hover` z dowolnym elementem, chociaż jest on najczęściej używany z linkami, aby dać użytkownikowi wizualną informację zwrotną, że akcja jest możliwa. Stany `hover` są również używane do uruchamiania wyskakujących menu do nawigacji lub do

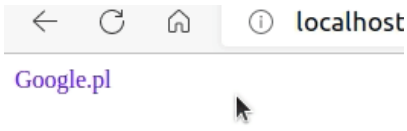
wyświetlania dodatkowych informacji o obiekcie na stronie. Ta reguła nadaje linkom jasnoróżowy kolor tła, gdy kursor myszy nad nimi się znajdzie:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        a:hover {
            color: maroon;
            background-color: #ffd9d9;
        }
    </style>
</head>
<body>
    <a href="https://google.pl">Google.pl</a>
</body>
</html>
```



W poprzednim rozdziale widzieliśmy właściwość `text-decoration` wykorzystywaną do wyłączania podkreśleń pod łączami. Możemy użyć selektora `:hover`, aby podkreślenia pojawiały się tylko „po najechaniu”.

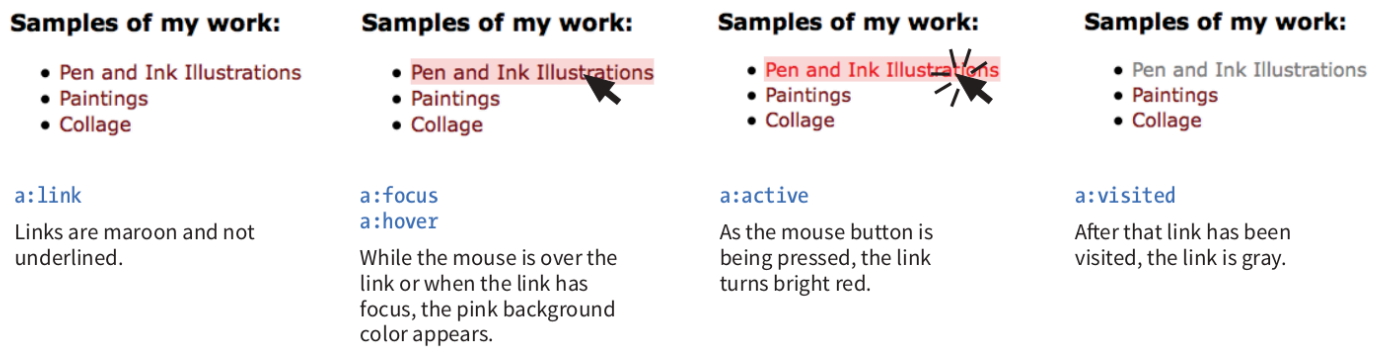
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        a {
            text-decoration: none;
        }
        a:hover {
            color: maroon;
            background-color: #ffd9d9;
            text-decoration: underline;
        }
    </style>
</head>
<body>
    <a href="https://google.pl">Google.pl</a>
</body>
</html>
```



Należy zauważyć, że na urządzeniach z ekranem dotykowym, takich jak smartfony i tablety, nie ma prawdziwego stanu najechania kursorem, dlatego efekty najechania należy stosować ostrożnie i stosować alternatywne rozwiązania.

Selektor `:active` stosuje style do elementu, który jest w trakcie aktywowania. W przypadku łącza jest to styl, który jest stosowany podczas klikania lub dotykania go koniuszkiem palca na ekranie dotykowym. Ten styl może być pokazywany tylko przez chwilę, ale może dać subtelną wskazówkę, że coś się wydarzyło.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        a {
            text-decoration: none;
        }
        a:hover {
            color: maroon;
            background-color: #ffd9d9;
            text-decoration: underline;
        }
        a:active {
            color: red;
            background-color: #ffd9d9;
        }
    </style>
</head>
<body>
    <a href="https://google.pl">Google.pl</a>
</body>
</html>
```



`W3C` tworzy wiele interesujących sposobów wybierania treści do stylizacji w oparciu o stany, które przeglądarka śledzi w locie. `CSS3` wprowadził całą masę pseudoklas, z których większość jest obecnie obsługiwana przez przeglądarki. Doskonałym źródłem informacji na temat tych selektorów CSS Level 3 i 4, w tym informacji o obsłudze przeglądarek, jest strona https://css4-selectors.com/ (https://css4-selectors.com/) autorstwa Nelly Brekardin.

Pseudoklasy strukturalne umożliwiają wybór na podstawie tego, gdzie element znajduje się w strukturze dokumentu (drzewo dokumentu):

```
:root
:empty
:first-child
:last-child
:only-child
:first-of-type
:last-of-type
:only-of-type
:nth-child()
:nth-last-child()
:nth-of-type()
```

Pseudoklasy wejściowe te selektory mają zastosowanie do stanów typowych dla danych wejściowych formularza:

```
:enabled
:disabled
:checked
```

Pseudoklasy lokalizacji (oprócz `:link` i `:visited`):

```
:target (fragment identifier)
```

Pseudoklasa językowa:

```
:lang()
```

Logiczna pseudoklasa:

```
:not()
```

Step 4

Pseudoklasy nie są jedynym rodzajem **pseudoselektorów**. Istnieją również **cztery pseudoelementy**, które działają tak, jakby wstawiały fikcyjne elementy do struktury dokumentu w celu stylizacji. W `CSS3` pseudoelementy są **oznaczone symbolem podwójnego dwukropka** (`::`), aby odróżnić je od pseudoklas. Jednak wszystkie przeglądarki obsługują składnię z jednym dwukropkiem (:) zdefiniowaną w `CSS2` , więc wielu programistów trzyma się tego, aby zapewnić wsteczną kompatybilność ze starszymi przeglądarkami.

Poniższe pseudoselectory służą do zaznaczania pierwszego wiersza lub pierwszej litery tekstu w elemencie wyświetlanym w przeglądarce.

Pseudoselektor `::first-line` stosuje regułę stylu do pierwszego wiersza określonego elementu. Jedyne właściwości, które możemy zastosować, to:

```
color
text-decoration
font properties
vertical-align
background properties
text-transform
word-spacing
line-height
letter-spacing
```

```
<!--pseudoselector_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Pseudoselector examples</title>
    <style>
      p::first-line {
        color: red;
      }
    </style>
</head>
<body>
  <p>
    This is my first line.<br>
    This is my second line.
  </p>
</body>
</html>
```

Pseudoselector `::firstletter` definiuje regułę stylu do pierwszej litery określonego elementu. Właściwości, które możemy zastosować, są ograniczone do następujących:

```
color
vertical-align (if float is none )
font properties
padding properties
background properties
margin properties
letter-spacing
border properties
word-spacing line-height
text-decoration float
text-transform
```

```
<!--pseudoselector_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Pseudoselector examples</title>
    <style>
      p::first-line {
        color: red;
      }
      p::first-letter{
        font-size: 300%;
        color: orange;
      }
    </style>
</head>
<body>
  <p>
    This is my first line.<br>
    This is my second line.
  </p>
</body>
</html>
```

This is my first line.
This is my second line.

Widzieliśmy, jak przeglądarki automatycznie dodają punktory i numery do list, nawet jeśli nie znajdują się one w źródle `HTML`. To jest przykład **treści generowanych**, treści, które przeglądarki wstawiają w locie. Można nakazać przeglądarkom, aby generowały treść przed lub po dowolnym elemencie, używając pseudoelementów `::before` i `::after`. Wygenerowane treści można wykorzystać do dodawania ikon przed pozycjami listy, wyświetlania adresów URL obok łączy podczas drukowania dokumentów internetowych, dodawania odpowiednich dla języka cudzysłowów wokół cytatu i wiele więcej. Oto prosty przykład, który wstawia obraz za pomocą funkcji `url()` przed akapitem i „Thank you". na końcu akapitu.

```
<!--before_and_after_pseudoselectors_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Before and after pseudoselectors_example</title>
    <style>
      p.warning::before {
        content: url(exclamation.png);
        margin-right: 6px;
      }
      p.warning::after {
        content: " Thank You ";
        color: red;
      }
    </style>
</head>
<body>
    <p class="warning">We are required to warn you.</p>
</body>
</html>
```

We are required to warn you. Thank You

W tym przykładzie warto zwrócić uwagę na kilka rzeczy:

- Selektor pseudoelementu przechodzi bezpośrednio po elemencie docelowym bez spacji.
- Reguła pseudoelementu zarówno wstawia treść, jak i określa sposób jej stylizacji w jednym bloku deklaracji.
- Właściwość `content`, która zapewnia zawartość, którą chcemy wstawić, jest wymagana. Bez niego selektor nic nie zrobi.
- Aby spacje między wygenerowaną treścią a treścią z dokumentu źródłowego były konieczne, należy umieścić spacje znaków wewnątrz cudzysłowów wartości lub zastosować margines.

Korzystając z wygenerowanej zawartości, pamiętajmy, że to, co wstawiamy, nie staje się częścią DOM dokumentu. Istnieje tylko na ekranie przeglądarki i nie jest dostępna dla urządzeń pomocniczych, takich jak czytniki ekranu. Najlepiej jest używać wygenerowanych treści do dekoracji i innych „dodatków", które nie są krytyczne dla naszego znaczenia i wiadomości.

Step 5

Selektory atrybutów celują w elementy w oparciu o nazwy lub wartości atrybutów, co zapewnia dużą elastyczność przy wybieraniu elementów bez konieczności dodawania dużej ilości znaczników klas lub identyfikatorów. Lista selektorów atrybutów `CSS3` znajduje się tutaj:

`element[attribute]` - Prosty selektor atrybutu kieruje elementy z określonym atrybutem, niezależnie od jego wartości. Poniższy przykład wybiera dowolny obraz, który ma atrybut `title`.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Attribute selectors example</title>
    <style>
      img[title] {
        border: 3px solid;
      }
    </style>
</head>
<body>
  <img src="exclamation.png" title="exclamation" alt="exclamation.png">
  <img src="exclamation.png" alt="exclamation.png">
</body>
</html>
```

`element[attribute="exact value"]` - Selektor dokładnej wartości atrybutu wybiera elementy o określonej wartości atrybutu. Ten selektor dopasowuje obrazy o dokładnie takiej wartości tytułu `exclamation`.

```
<!--attribute_selectors_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Attribute selectors example</title>
    <style>
      img[title="exclamation"] {
        border: 3px solid;
      }
    </style>
</head>
<body>
  <img src="exclamation.png" title="exclamation" alt="exclamation.png">
  <img src="exclamation.png" title="exclamation1" alt="exclamation.png">
</body>
</html>
```

`element[attribute~="value"]` Selektor częściowej wartości atrybutu (oznaczony tyldą, `~`) umożliwia określenie jednej części wartości atrybutu. Poniższy przykład szuka słowa `grade` w tytule, więc zostaną wybrane obrazy z wartościami `first grade` i `second grade`.

```
<!--attribute_selectors_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Attribute selectors example</title>
    <style>
      img[title~="grade"] {
        border: 3px solid;
      }
    </style>
</head>
<body>
  <img src="exclamation.png" title="first grade" alt="exclamation.png">
  <img src="exclamation.png" title="first" alt="exclamation.png">
</body>
</html>
```

`element[attribute|="value"]` - Selektor wartości atrybutów rozdzielonych myślnikami (oznaczony kreską, `|`) odnosi się do wartości rozdzielonych myślnikami. Ten selektor dopasowuje dowolny link, który wskazuje na dokument napisany w odmianie języka angielskiego (`en`), niezależnie od tego, czy wartość atrybutu to `en-us` (amerykański angielski), `en-in` (indyjski angielski), `en-au-tas` (australijski). angielski) i tak dalej.

```
<!--attribute_selectors_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Attribute selectors example</title>
    <style>
      a[hreflang|="en"] {
        color: red;
      }
    </style>
</head>
<body>
  <a hreflang="pl">English</a><br/>
  <a hreflang="en-us">English (US)</a><br/>
  <a hreflang="en-gb">English (GB)</a><br/>
</body>
</html>
```

English
English (US)
English (GB)

`element[attribute^="first part of the value"]` - Początkowy selektor wartości atrybutu podłańcucha (oznaczony karatem, `^`) pasuje do elementów, których określone wartości atrybutu zaczynają się w ciągu znaków w selektorze. Ten przykład stosuje styl tylko do obrazów znajdujących się w katalogu `/images/icons`.

```
<!--attribute_selectors_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Attribute selectors example</title>
    <style>
      img[src^="images/icons"] {
        border: 3px solid;
      }
    </style>
</head>
<body>
  <img src="images/icons/css.png" alt="css.png">
  <img src="exclamation.png" alt="exclamation.png">
</body>
</html>
```

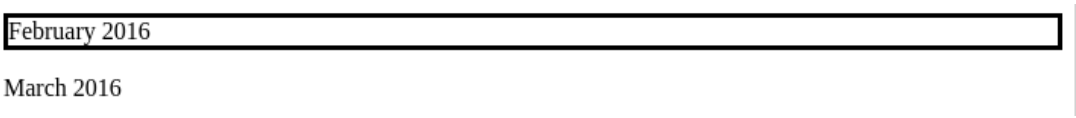
`element[attribute$="last part of value"]` - Końcowy selektor wartości atrybutu podciągu (oznaczony znakiem dolara, `$`) pasuje do elementów, których określone wartości atrybutu kończą się ciągiem znaków w selektorze. W tym przykładzie styl można zastosować tylko do elementów `a`, które łączą się z plikami `PDF`.

```
<!--attribute_selectors_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Attribute selectors example</title>
    <style>
      a[href$=".pdf"] {
        color: red;
      }
    </style>
</head>
<body>
    <hr/>
    <a href="http://www.google.com">Google</a><br/>
    <a href="http://www.google.com.pdf">Google PDF</a><br/>
</body>
</html>
```

Google
Google PDF

`element[attribute*="any part of the value"]` - Selektor wartości atrybutu dowolnego podłańcucha (oznaczony gwiazdką, `*`) szuka podanego ciągu tekstowego w dowolnej części określonej wartości atrybutu. Ta reguła wybiera dowolny obraz, który zawiera słowo „February” gdzieś w jego tytule.

```
<!--attribute_selectors_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Attribute selectors example</title>
    <style>
      p[title*="February"]{
          border: 3px solid;
      }
    </style>
</head>
<body>
    <p title="February 2016">February 2016</p>
    <p title="March 2016">March 2016</p>
</body>
</html>
```



Step 6

Widzieliśmy, jak dodawać obrazy do treści dokumentu za pomocą elementu `img`, ale większość ozdobnych obrazów jest dodawanych do stron i elementów jako tła za pomocą `CSS`. W końcu dekoracje, takie jak kafelkowanie deseni w tle, są nieodłączną częścią prezentacji, a nie strukturą. Przyjrzymy się zbiorowi właściwości używanych do umieszczania i przesuwania obrazów tła, zaczynając od podstawowej właściwości `background-image`.

Właściwość `background-image` dodaje obraz tła do dowolnego elementu. Jego głównym zadaniem jest podanie lokalizacji pliku obrazu.

```
background-image

Wartości: (adres URL obrazu) | none
Domyślnie: none
Dotyczy: Wszystkich elementów
Dziedziczy: Nie
```

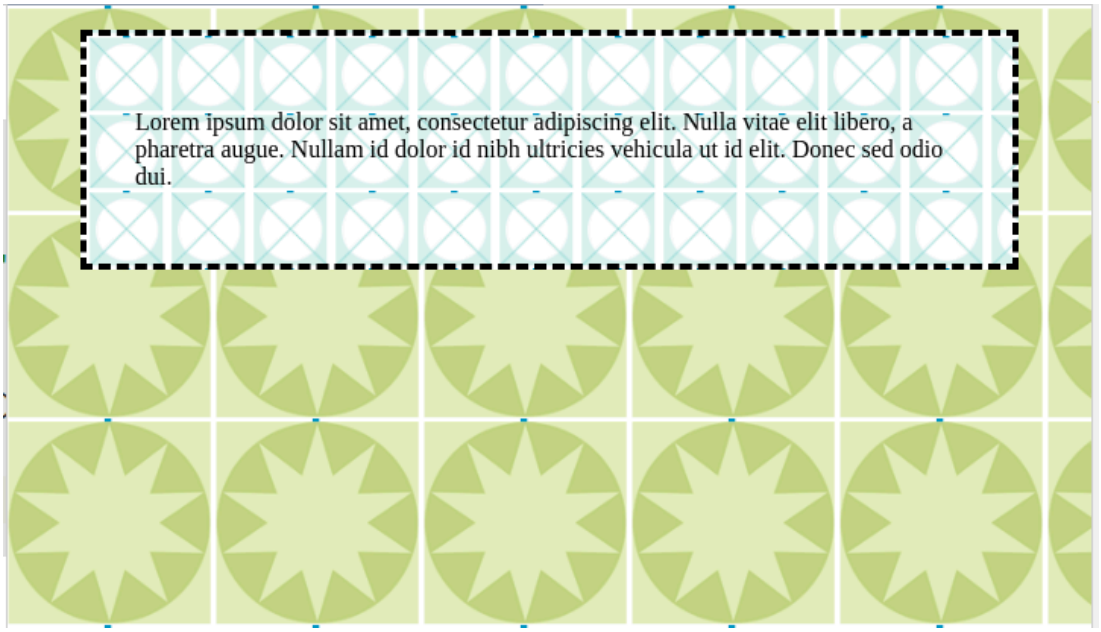
Wartość `background-image` jest rodzajem posiadacza adresu `URL`, który zawiera lokalizację obrazu. Adres `URL` odnosi się do miejsca, w którym w danej chwili znajduje się reguła `CSS`. Jeśli reguła znajduje się w osadzonym arkuszu stylów (element stylu w dokumencie `HTML`), wówczas nazwa ścieżki w adresie `URL` powinna odnosić się do lokalizacji pliku `HTML`. Jeśli reguła `CSS` znajduje się w zewnętrznym arkuszu stylów, ścieżka do obrazu powinna być względna do lokalizacji pliku `.css`. Alternatywnie podanie względnych adresów `URL` katalogu głównego dla obrazów zapewnia, że obraz tła można znaleźć niezależnie od lokalizacji reguł stylu.

Katalog główny jest oznaczony ukośnikiem na początku adresu `URL`. Na przykład:

```
background-image: url(/images/background.jpg);
```

Wadą, podobnie jak w przypadku wszystkich względnych adresów `URL` katalogu głównego, jest to, że nie będzie można go przetestować lokalnie (z własnego komputera), chyba że skonfigurowano go jako serwer. Poniżej znajduje się przykład, który pokazuje obrazy tła zastosowane za całą stroną (treścią) oraz pojedynczy element cytatu blokowego z dopełnieniem i obramowaniem.

```
<!--background_image_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Background image example</title>
    <style>
      body {
        background-image: url("images/background/star.png");
      }
      blockquote {
        background-image: url("images/background/dot.png");
        padding: 2em;
        border: 4px dashed;
      }
    </style>
</head>
<body>
    <blockquote>
        <p>
        Lorem ipsum dolor sit amet, consectetur adipiscing elit.
        Nulla vitae elit libero, a pharetra augue.
        Nullam id dolor id nibh ultricies vehicula ut id elit.
        Donec sed odio dui.
      </p> </blockquote>
</body>
</html>
```



Tutaj możemy zobaczyć domyślne zachowanie obrazu tła. Obraz zaczyna się w lewym górnym rogu i jest kafelkowany w poziomie i pionie, aż cały element zostanie wypełniony. Podobnie jak kolory tła, kafelkowe obrazy tła wypełniają obszar za obszarem zawartości, wypełniają dodatkową przestrzeń wokół zawartości i rozciągają się do zewnętrznej krawędzi obramowania (jeśli istnieje). Możemy zmienić obszar malowania tła za pomocą właściwości `background-clip`.

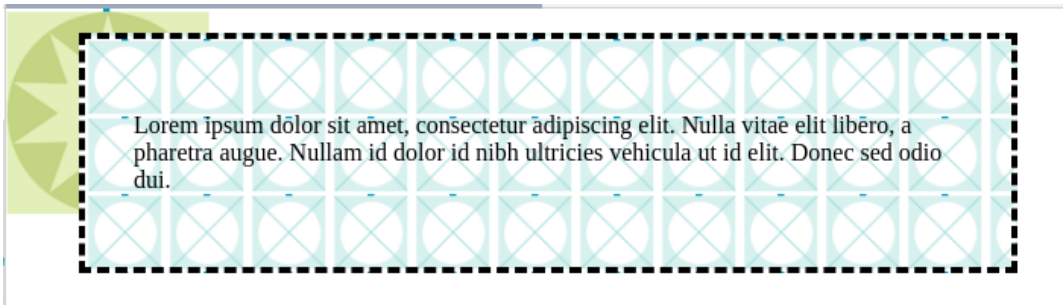
Jeśli dostarczymy do elementu zarówno `background-color`, jak i `background-image`, obraz zostanie umieszczony na górze koloru. W rzeczywistości zaleca się zapewnienie koloru zapasowego o podobnym odcieniu na wypadek, gdyby nie udało się pobrać obrazu.

Jak widzieliśmy przed chwilą, obrazy są rozmieszczane obok siebie w lewo i w prawo, w górę i w dół, gdy są pozostawione do ich własnych urządzeń. Możemy zmienić to zachowanie za pomocą właściwości `background-repeat`.

```
background-repeat

Wartości: repeat | no-repeat | repeat-x | repeat-y | space | round
Domyślnie: repeat
Dotyczy: wszystkich elementów
Dziedziczy: nie
```

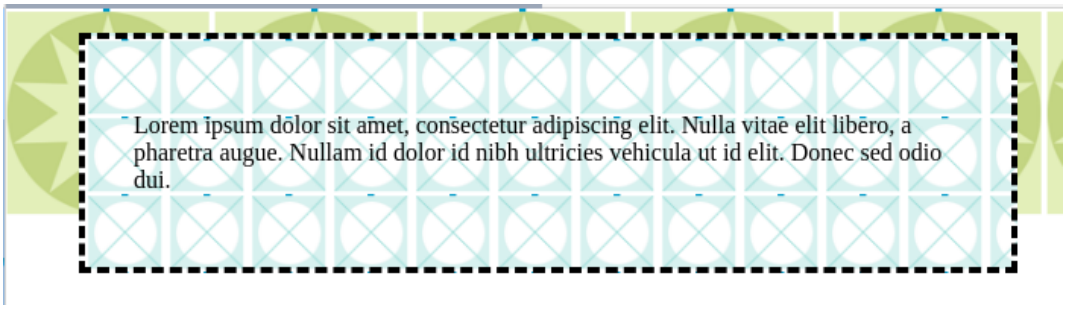
Jeśli chcemy, aby obraz tła pojawił się tylko raz, należy użyć wartości `no-repeat`:

```
<!--background_image_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Background image example</title>
    <style>
      body {
        background-image: url("images/background/star.png");
        background-repeat: no-repeat;
      }
      blockquote {
        background-image: url("images/background/dot.png");
        padding: 2em;
        border: 4px dashed;
      }
    </style>
</head>
<body>
    <blockquote>
        <p>
        Lorem ipsum dolor sit amet, consectetur adipiscing elit.
        Nulla vitae elit libero, a pharetra augue.
        Nullam id dolor id nibh ultricies vehicula ut id elit.
        Donec sed odio dui.
      </p> </blockquote>
</body>
</html>
```



Możemy również ograniczyć obraz do kafelkowania tylko poziomo (`repeat-x`) lub pionowo (`repeat-y`), jak pokazano w poniższych przykładach:

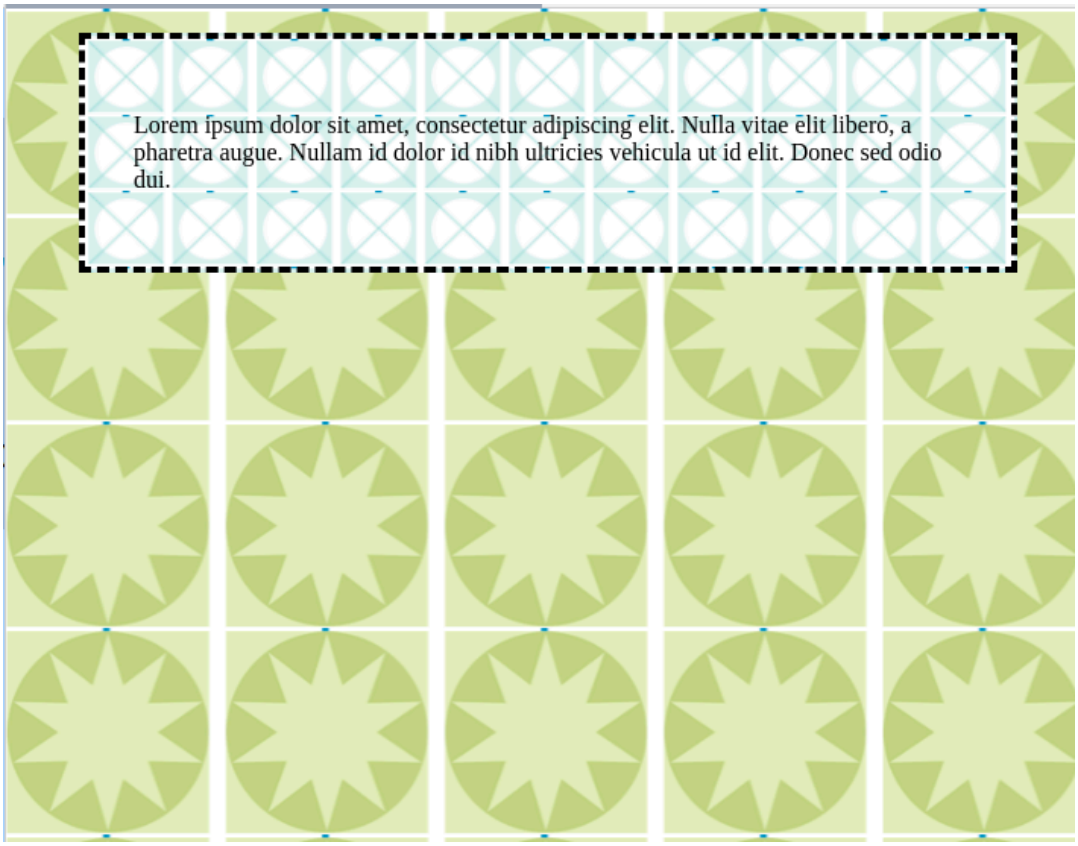
```
<!--background_image_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Background image example</title>
    <style>
      body {
        background-image: url("images/background/star.png");
        background-repeat: repeat-x;
      }
      blockquote {
        background-image: url("images/background/dot.png");
        padding: 2em;
        border: 4px dashed;
      }
    </style>
</head>
<body>
    <blockquote>
        <p>
        Lorem ipsum dolor sit amet, consectetur adipiscing elit.
        Nulla vitae elit libero, a pharetra augue.
        Nullam id dolor id nibh ultricies vehicula ut id elit.
        Donec sed odio dui.
      </p> </blockquote>
</body>
</html>
```




```
<!--background_image_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Background image example</title>
    <style>
      body {
        background-image: url("images/background/star.png");
        background-repeat: repeat-y;
      }
      blockquote {
        background-image: url("images/background/dot.png");
        padding: 2em;
        border: 4px dashed;
      }
    </style>
</head>
<body>
    <blockquote>
        <p>
        Lorem ipsum dolor sit amet, consectetur adipiscing elit.
        Nulla vitae elit libero, a pharetra augue.
        Nullam id dolor id nibh ultricies vehicula ut id elit.
        Donec sed odio dui.
      </p> </blockquote>
</body>
</html>
```

Zauważmy, że we wszystkich przykładach kafelkowanie zaczyna się w lewym górnym rogu elementu (lub w oknie przeglądarki, gdy obraz jest zastosowany do elementu `body`). Pozostałe wartości, `space` i `round`, próbują wypełnić dostępny obszar malowania tła parzystą liczbę razy. Gdy właściwość `background-repeat` jest ustawiona na `space`, przeglądarka oblicza, ile obrazów tła może zmieścić się na całej szerokości i wysokości obszaru tła, a następnie dodaje równe odstępy między każdym obrazem. Rezultatem są parzyste rzędy i kolumny oraz brak przyciętych obrazów. Słowo kluczowe `round` powoduje, że przeglądarka ściska obraz tła w poziomie i pionie (niekoniecznie proporcjonalnie), aby zmieścić się w obszarze tła parzystą liczbę razy.

```
<!--background_image_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Background image example</title>
    <style>
      body {
        background-image: url("images/background/star.png");
        background-repeat: space repeat;
      }
      blockquote {
        background-image: url("images/background/dot.png");
        background-repeat: round;
        padding: 2em;
        border: 4px dashed;
      }
    </style>
</head>
<body>
    <blockquote>
        <p>
        Lorem ipsum dolor sit amet, consectetur adipiscing elit.
        Nulla vitae elit libero, a pharetra augue.
        Nullam id dolor id nibh ultricies vehicula ut id elit.
        Donec sed odio dui.
      </p> </blockquote>
</body>
</html>
```



Właściwość `background-position` określa położenie obrazu początkowego w tle. Możemy myśleć o obrazie początkowym jako pierwszym obrazie umieszczonym w tle, z którego rozciągają się obrazy kafelkowe. Oto własność i jej różne wartości.

```
background-position

Wartości: length measurement | percentage | left | center | right | top | bottom
Domyślnie:0% 0% (same as left top)
Dotyczy: wszystkich elementów
Dziedziczy: nie
```

Aby ustawić obraz początkowy, należy podać wartości poziome i pionowe, które opisują, gdzie go umieścić. Można to zrobić na wiele sposobów.

Pozycjonowanie według słów kluczowych - Wartości słów kluczowych (`left` , `right` , `top` , `bottom` i `center`) pozycjonują obraz początkowy względem zewnętrznych krawędzi wypełnienia elementu. Na przykład `left` ustawia obraz aż do lewej krawędzi obszaru tła. Domyślna pozycja początkowa odpowiada lewej górze. Słowa kluczowe są zwykle używane w parach, jak w poniższych przykładach:

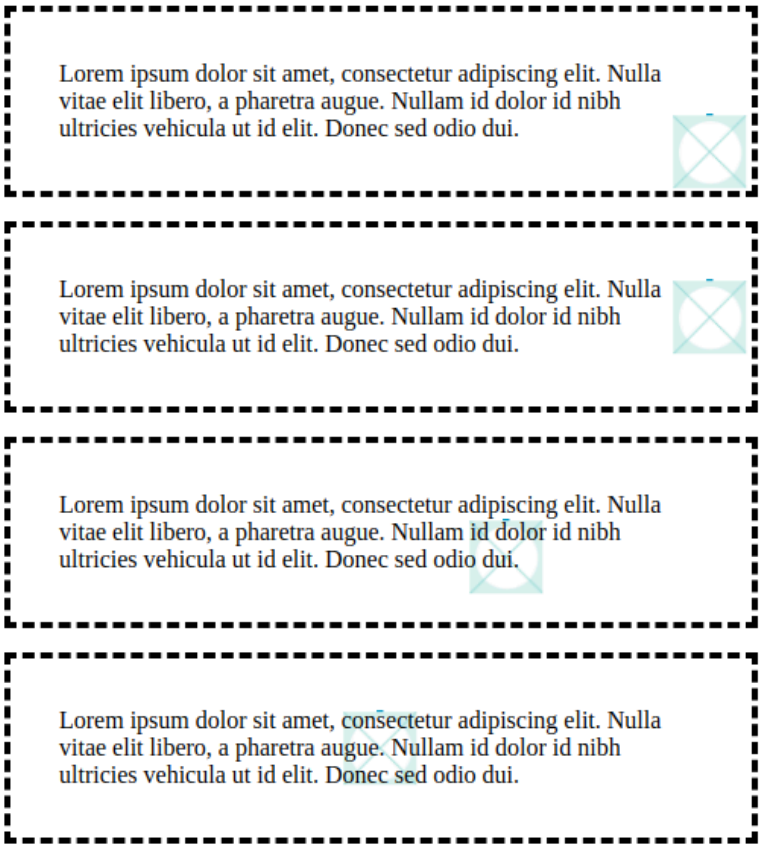
```
background-position: left bottom;
background-position: right center;
```

Słowa kluczowe mogą pojawiać się w dowolnej kolejności. W przypadku podania tylko jednego słowa kluczowego zakłada się, że brakujące słowo kluczowe to `center` . Zatem `background-position: right` ma taki sam efekt jak `background-position: right center` .

Pomiary długości - Określanie pozycji za pomocą pomiarów długości, takich jak **piksele** lub `em` , wskazuje wielkość przesunięcia od lewego górnego rogu elementu do lewego górnego rogu obrazu początkowego tła. Kiedy podajemy wartości długości, pomiar poziomy zawsze jest pierwszy. Określanie wartości ujemnych jest dozwolone i powoduje, że obraz jest zawieszony poza widocznym obszarem tła. Kiedy podajemy wartości długości lub wartości procentowe, pomiar poziomy zawsze jest pierwszy.

Procenty - Wartości procentowe są podawane w parach poziomych/pionowych, przy czym 0% odpowiada lewemu górnemu rogowi, a 100% odpowiada prawemu dolnemu rogowi. Podobnie jak w przypadku wartości długości, pomiar poziomy zawsze jest pierwszy. Należy zauważyć, że wartość procentowa dotyczy zarówno obszaru płótna, jak i samego obrazu. Wartość pozioma `25%` umieszcza punkt `25%` od lewej krawędzi obrazu w punkcie, który znajduje się `25%` od lewej krawędzi obszaru pozycjonowania tła. Wartość pionowa `100%` umieszcza dolną krawędź obrazu na dolnej krawędzi obszaru pozycjonowania.

```
<!--background_position_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Background position example</title>
  <style>
    blockquote {
      padding: 2em;
      border: 4px dashed;
    }
    blockquote[title="left bottom"] {
      background-image: url("images/background/dot.png");
      background-position: right bottom;
      background-repeat: no-repeat;
    }
    blockquote[title="right"] {
      background-image: url("images/background/dot.png");
      background-position: right;
      background-repeat: no-repeat;
    }
    blockquote[title="pixels"] {
      background-image: url("images/background/dot.png");
      background-position: 300px 50px;
      background-repeat: no-repeat;
    }
    blockquote[title="percent"] {
      background-image: url("images/background/dot.png");
      background-position: 50% 50%;
      background-repeat: no-repeat;
    }

  </style>
</head>
<body>
  <blockquote title="left bottom">
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </blockquote>
  <blockquote title="right">
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </blockquote>
  <blockquote title="pixels">
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </blockquote>
  <blockquote title="percent">
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </blockquote>
</body>
</html>
```



Możliwe jest pozycjonowanie obrazu początkowego i umieszczanie go stamtąd, w obu kierunkach lub po prostu w poziomie lub w pionie. Gdy obraz jest sąsiadujący, pozycja początkowego obrazu może nie być oczywista, ale można użyć `background-position`, aby wzór sąsiadujący zaczynał się w punkcie innym niż lewa krawędź obrazu. Może to służyć do utrzymania wyśrodkowanego i symetrycznego wzoru tła.

Step 7

Gdy obraz źródłowy został umieszczony w rogu elementu, zostaje umieszczony wewnątrz ramki (tylko powtarzające się obrazy rozciągają się pod ramką do jego zewnętrznej krawędzi). Jest to pozycja domyślna, ale można ją zmienić za pomocą właściwości `background-origin`.

```
background-origin

Wartości: border-box | padding-box | content-box
Domyślnie: padding-box
Dotyczy: Wszystkich elementów
Dziedziczy: nie
```

Ta właściwość definiuje granice obszaru pozycjonowania tła w ten sam sposób, w jaki `background-clip` definiuje obszar malowania tła. Możemy ustawić granice ramki `border-box` (więc obraz źródłowy jest umieszczany pod zewnętrzną krawędzią ramki, `padding-box` (zewnętrzna krawędź dopełnienia, tuż wewnątrz ramki) lub `content-box` (rzeczywisty obszar zawartości).

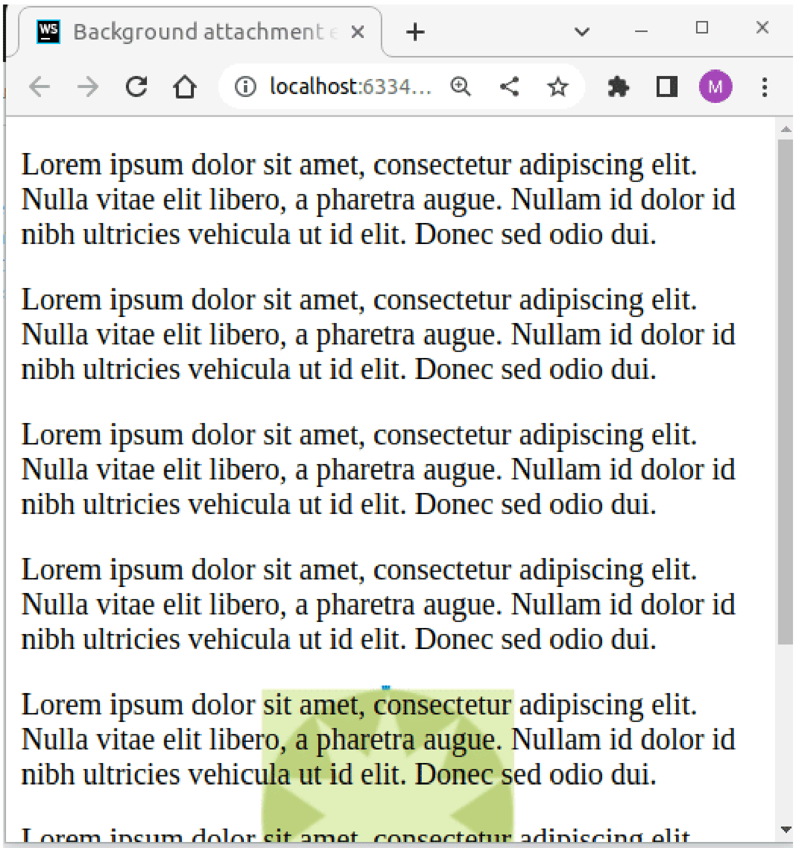
```
<!--background_origin_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Background origin example</title>
    <style>
      blockquote{
        background-image: url("images/background/dot.png");
        background-repeat: no-repeat;
        padding: 2em;
        border: 4px dashed;
      }
      blockquote[title="border-box"] {
        background-origin: border-box;
      }
      blockquote[title="padding-box"] {
        background-origin: padding-box;
      }
      blockquote[title="content-box"] {
        background-origin: border-box;
      }
    </style>

</head>
<body>
  <blockquote title="border-box">
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </blockquote>
  <blockquote title="padding-box">
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </blockquote>
  <blockquote title="content-box">
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </blockquote>
</body>
</html>
```

Właściwości `background-attachment`, uwolnia tło z treści i pozostawia je w jednej pozycji, podczas gdy reszta treści będzie się przewijać.

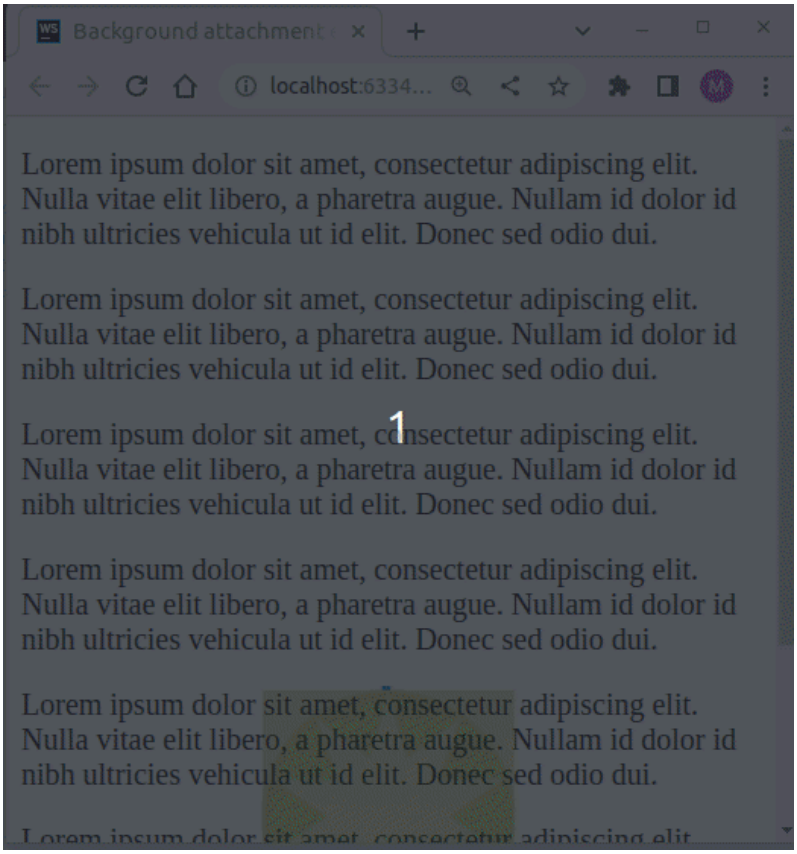
```
background-attachment

Wartości: scroll | fixed | local
Domyślnie: scroll
Dotyczy: wszystkich elementów
Dziedziczy: nie
```

Dzięki właściwości `background-attachment` mamy wybór, czy obraz tła przewija się wraz z treścią, czy pozostaje w stałej pozycji. Gdy obraz jest ustawiony na fixed, pozostaje w tej samej pozycji względem przeglądarki (w przeciwieństwie do elementu, który wypełnia). W poniższym przykładzie w tle całego dokumentu (element `body`) umieszczany jest duży obraz, który nie jest kafelkowy. Domyślnie podczas przewijania dokumentu przewija się również obraz, przesuwając się w górę i poza stronę. Jeśli jednak ustawimy `background-attachment` na `fixed`, pozostanie ona tam, gdzie była początkowo umieszczona, a tekst będzie przewijał się nad nią.

```
<!--background_attachment_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Background attachment example</title>
  <style>
    body {
      background-image: url("images/background/star.png");
      background-repeat: no-repeat;
      background-position: center 300px;
      background-attachment: scroll;
    }
  </style>
</head>
<body>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
</body>
</html>
```


Background attachment e
localhost:6334...
M
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue. Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue. Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue. Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue. Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue. Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.

```html
<!--background_attachment_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Background attachment example</title>
  <style>
    body {
      background-image: url("images/background/star.png");
      background-repeat: no-repeat;
      background-position: center 300px;
      background-attachment: fixed;
    }
  </style>
</head>
<body>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
</body>
</html>
```



Wartość `local`, która została dodana w `CSS3`, jest przydatna, gdy element posiada własny mechanizm przewijania. Zamiast przewijać za pomocą scrollera okienka ekranu, `local` sprawia, że obraz tła jest przyklejany do zawartości przewijanego elementu.

OK, mamy jeszcze tylko jedną właściwość obrazu tła do omówienia, zanim zapakujemy to wszystko za pomocą skróconej właściwości `background`. Do tej pory obrazy tła, które widzieliśmy, są wyświetlane w rzeczywistym rozmiarze samego obrazu. Możemy zmienić rozmiar obrazu za pomocą właściwości `background-size`.

```
background-size

Wartości: length | percentage | auto | cover | contain
Domyślnie: auto
Dotyczy: Wszystkich elementów
Dziedziczy: nie
```

Rozmiar obrazu tła można określić na kilka sposobów. Być może najprostszym jest określenie wymiarów w jednostkach długości, takich jak piksele lub `em`. Jak zwykle, gdy podane są dwie wartości, pierwsza jest używana jako pomiar poziomy. Jeśli podamy tylko jedną wartość, zostanie ona użyta jako pomiar poziomy, a wartość pionowa zostanie ustawiona na `auto`.

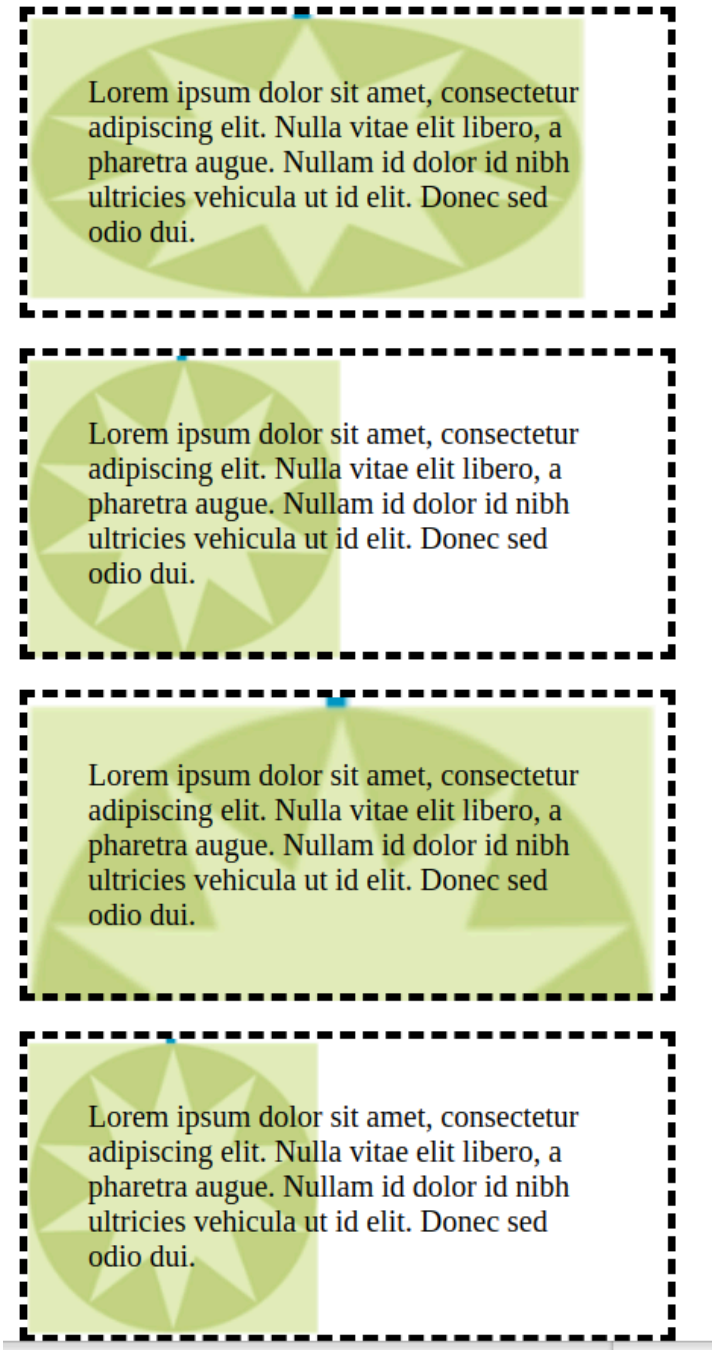
Wartości procentowe są **obliczane na podstawie obszaru pozycjonowania tła**, który domyślnie biegnie do wewnętrznej krawędzi obramowania, ale mógł zostać zmieniony w przypadku pochodzenia tła — o czym należy pamiętać. Tak więc pozioma wartość `50%` nie powoduje, że obraz jest o połowę mniejszy; raczej dopasowuje go do `50%` szerokości obszaru pozycjonowania. Ponownie, pierwsza jest wartość pozioma.

Słowo kluczowe `auto` zmienia rozmiar obrazu w kierunku niezbędnym do zachowania jego proporcji. Obrazy bitmapowe, takie jak `GIF`, `JPEG` i `PNG`, mają wewnętrzne proporcje, więc zawsze pozostają proporcjonalne, gdy jedna wartość rozmiaru jest ustawiona na `auto`. Niektóre obrazy, takie jak gradienty `SVG` i `CSS`, nie mają wewnętrznych proporcji. W takim przypadku `auto` ustawia szerokość lub wysokość na `100%` szerokości lub wysokości obszaru pozycjonowania tła.

Słowo kluczowe `cover` i `contain` zostały dodane w `CSS3`. Gdy ustawimy rozmiar tła na `cover`, przeglądarka zmieni rozmiar obrazu tła wystarczająco duży, aby dotrzeć do wszystkich boków obszaru pozycjonowania tła. Będzie tylko jeden obraz, ponieważ wypełnia on cały element i jest prawdopodobne, że fragmenty obrazu wyjdą poza obszar pozycjonowania, jeśli proporcje obrazu i obszar pozycjonowania nie będą się zgadzać.

W przeciwieństwie do tego, `contain` zawiera rozmiary obrazu na tyle duże, aby wypełnić szerokość lub wysokość obszaru pozycjonowania (w zależności od proporcji obrazu). Cały obraz będzie widoczny i „zawarty” w obszarze tła. Jeśli pozostało miejsce, obraz tła jest powtarzany, chyba że opcja `background-repeat` jest ustawiona na `no-repeat`.

```
<!--background_size_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Background size example</title>
  <style>
    blockquote {
      background-image: url("images/background/star.png");
      background-repeat: no-repeat;
      padding: 2em;
      border: 4px dashed;
    }
    blockquote[title="pixels"]{
      background-size: 300px 150px;
    }
    blockquote[title="mix"]{
      background-size: 50% 10em;
    }
    blockquote[title="cover"]{
      background-size: cover;
    }
    blockquote[title="contain"]{
      background-size: contain;
    }
  </style>
</head>
<body>
  <blockquote title="pixels">
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </blockquote>
  <blockquote title="mix">
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </blockquote>
  <blockquote title="cover">
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </blockquote>
  <blockquote title="contain">
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </blockquote>
</body>
</html>
```



Możemy użyć poręcznej właściwości `background`, aby określić wszystkie style tła w jednej deklaracji.

```
background

Wartości: background-color background-image background-repeat background-attachment background-position
background-clip background-origin background-size
Domyślne wartości: takie same jak we wyżej wymienionych właściwościach
Dotyczy: Wszystkich elementów
Dziedziczy: nie
```

Właściwość `background` jest listą wartości, które byłyby dostarczane dla poszczególnych wymienionych wcześniej właściwości tła. Na przykład ta jedna zasada tła:

```
body { background: white url(star.png) no-repeat right top fixed; }
```

zastępuje tę zasadę pięcioma oddzielnymi deklaracjami:

```
body {
  background-color: white;
  background-image: url(star.png);
  background-repeat: no-repeat;
  background-position: right top;
  background-attachment: fixed;
}
```

Wszystkie wartości `background` są opcjonalne i mogą pojawiać się w dowolnej kolejności. Jedynym ograniczeniem jest to, że gdy podajemy współrzędne dla właściwości `background-position`, najpierw musi pojawić się wartość pozioma, a zaraz po niej wartość pionowa. Podobnie jak w przypadku każdej skróconej właściwości, należy pamiętać, że jeśli jakakolwiek wartość zostanie pominięta, zostanie zresetowana do wartości domyślnej.

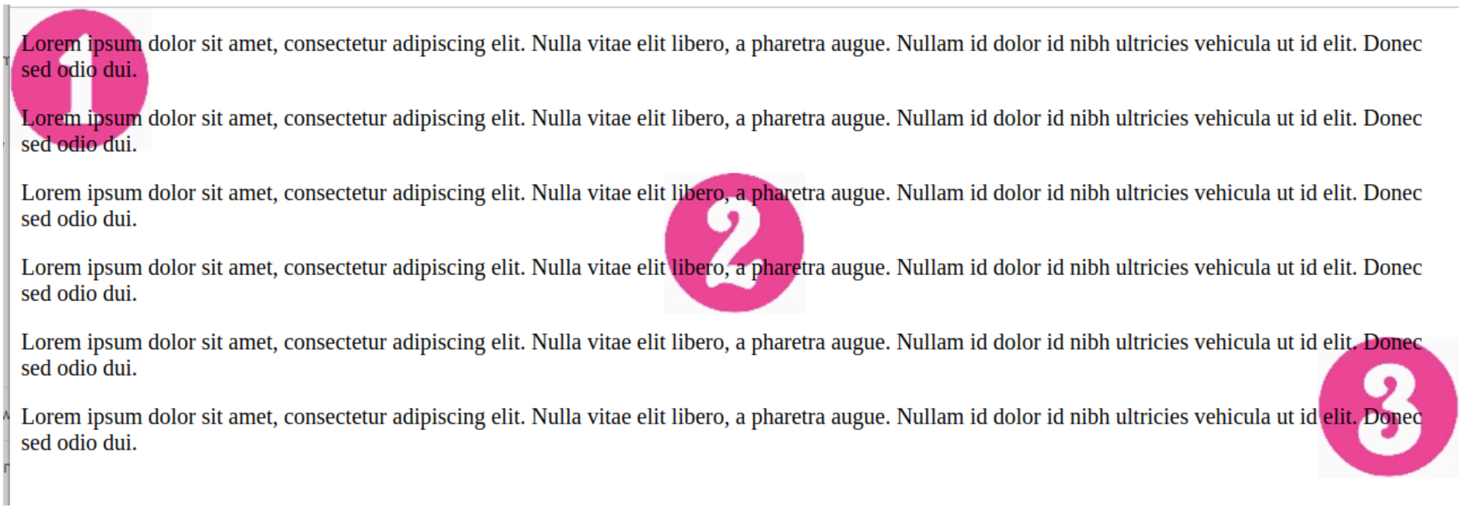
`CSS3` wprowadził możliwość zastosowania wielu obrazów tła do jednego elementu. Aby zastosować wiele wartości dla obrazu tła, należy umieścić je na liście oddzielonej przecinkami. Dodatkowe wartości właściwości związanych z tłem również są umieszczane na listach oddzielonych przecinkami; pierwsza wymieniona wartość dotyczy pierwszego obrazu, druga wartość drugiego i tak dalej.

Chociaż deklaracje `CSS` zwykle działają na zasadzie „ostatni wygrywa", w przypadku wielu obrazów tła ten, który jest wymieniony jako ostatni, jest umieszczany na dole, a każdy obraz znajdujący się wcześniej na liście jest nakładany na wierzch. Możesz myśleć o nich jak o warstwach Photoshopa, ponieważ są układane w stos w kolejności, w jakiej pojawiają się na liście. Innymi słowy, obraz zdefiniowany przez pierwszą wartość znajdzie się z przodu, a inne ustawią się za nim, w kolejności, w jakiej są wymienione.

```
body {
  background-image: url(image1.png), url(image2.png), url(image3.png);
  background-position: left top, center center, right bottom;
  background-repeat: no-repeat, no-repeat, no-repeat;
…
}
```

Alternatywnie możemy skorzystać z właściwości `background`, aby uprościć regułę. Teraz właściwość `background` ma trzy serie wartości oddzielone przecinkami:

```
<!--multiple_backgrounds_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Multiple backgrounds example</title>
  <style>
    body {
      background:
        url(images/background/back1.jpeg) left top no-repeat,
        url(images/background/back2.jpeg) center center no-repeat,
        url(images/background/back3.jpeg) right bottom no-repeat;
    }
  </style>
</head>
<body>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
  <p>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla vitae elit libero, a pharetra augue.
Nullam id dolor id nibh ultricies vehicula ut id elit. Donec sed odio dui.
  </p>
</body>
</html>
```



Step 8

Gradient to przejście od jednego koloru do drugiego, czasami przez wiele kolorów. W przeszłości jedynym sposobem na umieszczenie gradientu na stronie internetowej było utworzenie go w programie do edycji obrazów i dodanie wynikowego obrazu za pomocą `CSS`. Teraz możemy określić gradienty kolorów za pomocą samej notacji `CSS`, pozostawiając zadanie renderowania mieszanek kolorów przeglądarce. Chociaż są one określone w kodzie, gradienty są obrazami. Po prostu są generowane w locie. Obraz gradientowy nie ma własnego rozmiaru ani proporcji; rozmiar pasuje do elementu, do którego zostanie zastosowany. Gradienty można zastosować w dowolnym miejscu, w którym można zastosować obraz: `background-image`, `border-image` i `list-style-image`.

Istnieją dwa rodzaje gradientów:

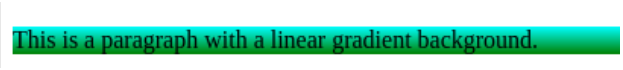
- **Gradienty liniowe** zmieniają kolory wzdłuż linii, od jednej krawędzi elementu do drugiego.
- **Gradienty promieniowe** zaczynają się w punkcie i rozchodzą na zewnątrz w kształcie kołowym lub eliptycznym.

Notacja `linear-gradient()` zapewnia kąt linii gradientu i jeden lub więcej punktów wzdłuż tej linii, w których znajduje się czysty kolor (kolor zatrzymuje się). Możemy użyć nazw kolorów lub dowolnych liczbowych wartości kolorów omówionych wcześniej, w tym przezroczystości (transparency). Kąt linii gradientu określany jest w stopniach (`ndeg`) lub za pomocą słów kluczowych. Przy stopniach 0 stopni wskazuje w górę, a dodatnie kąty obracają się zgodnie z ruchem wskazówek zegara, tak że 90 stopni wskazuje w prawo. Dlatego, jeśli chcemy przejść od `aqua` na górnej krawędzi do zielonej na dolnej krawędzi, ustawmy obrót na 180 stopni:

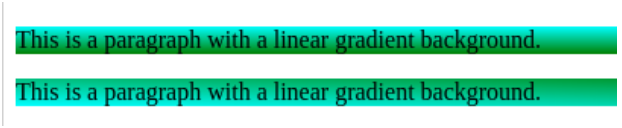
```
<!--linear_gradient_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Linear gradient example</title>
</head>
<body>
  <p style="background-image: linear-gradient(180deg,aqua,green)">
    This is a paragraph with a linear gradient background.
  </p>
</body>
</html>
```

Słowa kluczowe opisują kierunek w odstępach co 90° (`to top`, `to right`, `to bottom`, `to left`). Nasz gradient 180 stopni można również określić za pomocą słowa kluczowego `to bottom`.

```
<!--linear_gradient_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Linear gradient example</title>
</head>
<body>
  <p style="background-image: linear-gradient(to bottom,aqua,green)">
    This is a paragraph with a linear gradient background.
  </p>
</body>
</html>
```

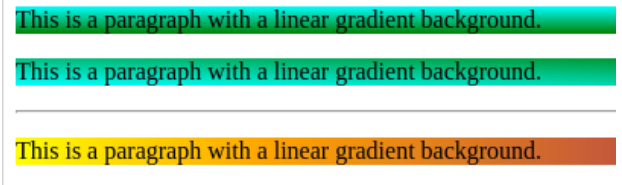


Możemy również użyć składni `to`, aby wskazać granice. Poniższy gradient będzie rysowany od lewego dolnego rogu do prawego górnego rogu. Wynikowy kąt gradientu rysowanego między rogami jest określany przez współczynnik kształtu pudełka.



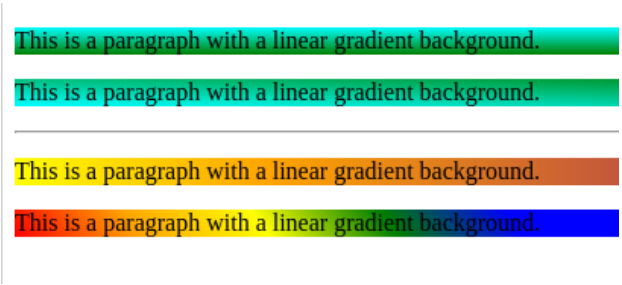
W poniższym przykładzie gradient przebiega teraz od lewej do prawej (`90` stopni) i zawiera trzeci kolor, `orange`, który pojawia się na `25%` długości linii gradientu. Widać, że położenie ogranicznika koloru jest wskazane po wartości koloru. Możemy użyć wartości procentowych lub dowolnego pomiaru długości. Pierwszy i ostatni znacznik koloru nie wymagają pozycji, ponieważ są one domyślnie ustawione odpowiednio na `0%` i `100%`.

```
<!--linear_gradient_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Linear gradient example</title>
</head>
<body>
  <p style="background-image: linear-gradient(to bottom,aqua,green)">
    This is a paragraph with a linear gradient background.
  </p>
  <p style="background-image: linear-gradient(to top right,aqua,green)">
      This is a paragraph with a linear gradient background.
  </p>
  <hr/>
  <p style="background-image: linear-gradient(90deg, yellow, orange 25%, purple)">
      This is a paragraph with a linear gradient background.
  </p>
</body>
</html>
```



Z pewnością nie ograniczamy się do kątów prostych. Określmy dowolny stopień, aby gradient liniowy był skierowany w tym kierunku. Możemy również określić dowolną liczbę kolorów. Jeśli nie określono pozycji, kolory są rozmieszczone równomiernie na całej długości linii gradientu. Jeśli umieścimy ostatni kolor przed końcem linii gradientu (np. `blue` na `50%` w tym przykładzie), ostatni kolor będzie kontynuowany do końca linii gradientu.

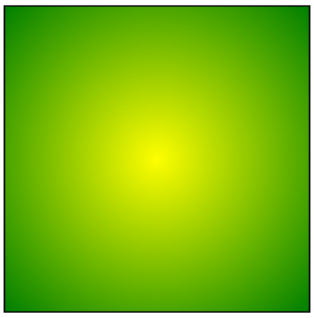
```
<!--linear_gradient_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Linear gradient example</title>
</head>
<body>
  <p style="background-image: linear-gradient(to bottom,aqua,green)">
    This is a paragraph with a linear gradient background.
  </p>
  <p style="background-image: linear-gradient(to top right,aqua,green)">
      This is a paragraph with a linear gradient background.
  </p>
  <hr/>
  <p style="background-image: linear-gradient(90deg, yellow, orange 25%, purple)">
      This is a paragraph with a linear gradient background.
  </p>
  <p style="background-image: linear-gradient(54deg, red, orange, yellow, green, blue 50%)">
      This is a paragraph with a linear gradient background.
</body>
</html>
```



Gradienty mają zarówno zalety, jak i wady, jeśli chodzi o wydajność. Plusem jest to, że nie wymagają dodatkowego połączenia z serwerem i wymagają mniej bajtów do pobrania niż obrazy. Z drugiej strony, całe to renderowanie w locie wymaga czasu i mocy obliczeniowej, które mogą negatywnie wpłynąć na wydajność. Gradienty promieniowe są najgorszymi winowajcami. Mogą być szczególnie problematyczne na urządzeniach mobilnych, gdzie moc obliczeniowa może być ograniczona. Rozważyć wówczas należy udostępnienie na urządzeniach mobilnych osobnego arkusza stylów bez gradientów.

Gradienty promieniowe, jak sama nazwa wskazuje, rozchodzą się promieniście od punktu w kole wzdłuż promienia gradientu (jak linia gradientu, ale zawsze wskazuje na zewnątrz od środka). Gradient promieniowy wymaga co najmniej dwóch punktów zatrzymania koloru, jak pokazano w tym przykładzie:

```
<!--radial_gradients_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Radial gradients example</title>
    <style>
      div {
        width: 200px;
        height: 200px;
        margin: 10px;
        border: 1px solid black;
      }
    </style>
</head>
<body>
  <div style="background-image: radial-gradient(yellow, green)">
  </div>
</body>
</html>
```

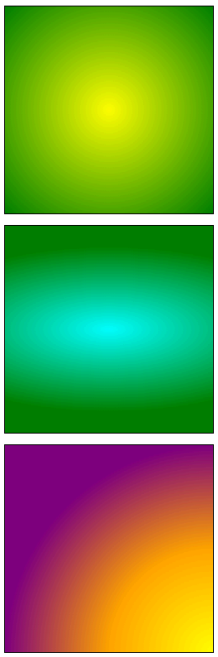


Domyślnie gradient wypełnia dostępny obszar tła, a jego środek znajduje się w środku elementu. Wynikiem jest elipsa, jeśli element zawierający jest prostokątem i okręgiem, jeśli element jest kwadratowy.

To już wygląda dość fajnie, ale nie musimy zadowalać się wartością domyślną. Notacja `radial-gradient()` umożliwia określenie kształtu, rozmiaru i położenia środka gradientu:

- **Kształt** (ang shape) - W większości przypadków kształt gradientu promieniowego będzie wynikał z kształtu elementu lub określonego rozmiaru, który do niego zastosujemy, ale możemy również określić kształt, używając słów kluczowych `circle` lub `ellipse`. Kiedy tworzymy gradient jako `circle` (bez sprzecznych specyfikacji rozmiaru), pozostaje on okrągły, nawet jeśli znajduje się w elemencie prostokątnym.
- **Rozmiar** (ang size) - Rozmiar gradientu promieniowego można określić w jednostkach długości lub procentach, które dotyczą promienia gradientu, lub za pomocą słów kluczowych. Jeśli podamy tylko jedną długość, zostanie ona użyta zarówno dla szerokości, jak i wysokości, w wyniku czego powstanie okrąg. Jeśli podamy dwie długości, pierwsza z nich to pomiar poziomy, a drugi to pomiar pionowy. W przypadku kształtu `ellipse` można również podać wartości procentowe lub mieszać wartości procentowe z wartościami długości. Istnieją również cztery słowa kluczowe — `closest-side`, `closest-corner`, `farthest-side` i `farthest-corner` — które określają długość promienia gradientu względem punktów.

```
<!--radial_gradients_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Radial gradients example</title>
    <style>
      div {
        width: 200px;
        height: 200px;
        margin: 10px;
        border: 1px solid black;
      }
    </style>
</head>
<body>
  <div style="background-image: radial-gradient(yellow, green)">
  </div>
  <div style="background-image: radial-gradient(200px 80px, aqua, green)">
  </div>
</body>
</html>
```



- **Położenie** - Domyślnie środek gradientu znajduje się na środku środka, ale można to zmienić, używając składni pozycjonowania, którą omówiliśmy dla właściwości `background-position`. Składnia jest taka sama, ale należy ją poprzedzić słowem kluczowym `at`. Zwróćmy uwagę, że w powyższym przykładzie dodano dodatkowy punkt koloru pomarańczowego przy znaku `50%`.

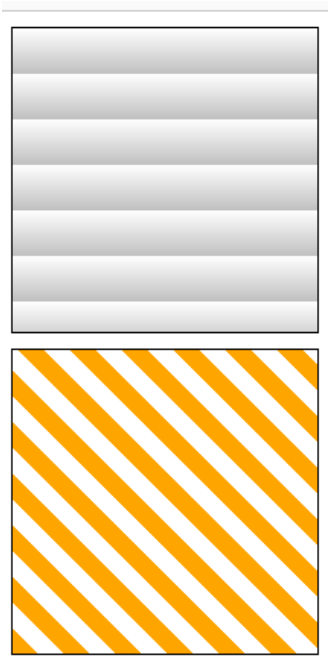
Step 9

Jeśli chcemy, aby wzór gradientu się powtarzał, należy użyć notacji `repeating-linear-gradient()` lub `repeating-radial-gradient()`. Składnia jest taka sama, jak w przypadku pojedynczych gradientów, ale dodanie `repeating-` powoduje, że wzór powtarza się, a kolor zatrzymuje się w nieskończoność w obu kierunkach. Jest to powszechnie używane do tworzenia ciekawych pasiastych wzorów. W tym prostym przykładzie gradient od białego do srebrnego (jasnoszary) powtarza się co `30` pikseli:

```
<!--repeating_gradient_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Repeating gradient example</title>
  <style>
    div {
      width: 200px;
      height: 200px;
      margin: 10px;
      border: 1px solid black;
    }
  </style>
</head>
<body>
<div style="background-image: repeating-linear-gradient(to bottom, white, silver 30px)"></div>
</body>
</html>
```

Kolejny przykład tworzy ukośny wzór pomarańczowo-białych pasków. Krawędzie są ostre, ponieważ biały pasek zaczyna się dokładnie w miejscu, w którym kończy się pomarańczowy (przy `12px`):

```
<!--repeating_gradient_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Repeating gradient example</title>
  <style>
    div {
      width: 200px;
      height: 200px;
      margin: 10px;
      border: 1px solid black;
    }
  </style>
</head>
<body>
<div style="background-image: repeating-linear-gradient(to bottom, white, silver 30px)"></div>
<div style="background-image: repeating-linear-gradient(45deg, orange, orange 12px, white 12px, white
24px)"></div>
</body>
</html>
```



Wszystkie główne przeglądarki zaczęły dodawać obsługę standardowej składni gradientu w latach 2012–2013, więc były niezawodne przez wiele lat. Jeśli jednak chcemy obsługiwać starsze przeglądarki, możemy to zrobić, korzystając z zastrzeżonej składni gradientu każdej przeglądarki z prefiksem dostawcy (ang. **vendor prefix**). Twórcy przeglądarek zwykle zaczynają majstrować przy autorskich rozwiązaniach dla najnowocześniejszych technologii internetowych, zanim specyfikacje zostaną w pełni ustalone. Przez wiele lat utrzymywali eksperymenty oddzielnie od ostatecznej implementacji, dodając **prefiks dostawcy** (lub **prefiks przeglądarki**) do nazwy właściwości lub funkcji. Prefiks wskazuje, że wdrożenie jest zastrzeżone i wciąż trwają prace. Na przykład, gdy `Safari` implementowało kształty zawijania tekstu, używało własnej wersji standardowej właściwości `shape-outside` z przedrostkiem `-webkit-`:

```
-webkit-shape-outside: url(cube.png);
```

Prefiksy dostawców umożliwiły programistom rozpoczęcie korzystania z nowych, fajnych funkcji `CSS` w przeglądarkach, które je obsługiwały, co było dodatkowym atutem za posunięcie naprzód projektowania stron internetowych i specyfikacji. Z drugiej strony cały system okazał się skomplikowany i często nadużywany. Ostatecznie twórcy przeglądarek zgodzili się odłożyć system prefiksów na spoczynek i nie udostępniać już żadnych własności. Obecnie przeglądarki ukrywają eksperymentalne funkcje za „flagami” (opcjami, które można włączyć lub wyłączyć) lub w osobnych wersjach podglądu technologii, do których programiści mają dostęp wyłącznie w

celach testowych. Gdy funkcja wydaje się stabilna, jest upubliczniana w oficjalnej wersji przeglądarki. Istnieje jednak kilka właściwości i funkcji `CSS`, które stały się modne w erze prefiksów, które nadal wymagają prefiksów, aby działały w starszych przeglądarkach, jeśli zdecydujemy się je obsługiwać. Poniższa tabela przedstawia używane prefiksy dostawców dla danych przeglądarek:

Prefix	Organizacja	Przeglądarki
`-ms-`	Microsoft	Internet Explorer (RIP)
`-moz-`	Mozilla Foundation	Firefox, Camino, SeaMonkey
`-o-`	Opera Software	Opera, Opera Mini, Opera Mobile
`-webkit-`	Open source	Safari, Chrome, Android, Silk, WebOS i wiele innych

Jeśli napiszemy swój `CSS` w standardowej składni, możemy uruchomić go przez postprocesor, taki jak https://autoprefixer.github.io/ (https://autoprefixer.github.io/). `Autoprefixer` analizuje style, a następnie automatycznie dodaje przedrostki tylko dla właściwości i notacji, które ich potrzebują.

Pomijając prefiksy dostawcy, samo opisywanie gradientów może być trudne. Chociaż nie jest niemożliwe napisanie kodu ręcznie, warto użyć narzędzia gradientowego online. Jedną z opcji jest `Ultimate CSS Gradient Generator` firmy `Colorzilla` (https://.colorzilla.com/gradient-editor/ (https://.colorzilla.com/gradient-editor/)). Po prostu wprowadźmy tyle kolorów, ile chcemy, przesuńmy suwaki, aż uzyskamy pożądany wygląd, a następnie skopiujmy kod. Generator gradientów `CSS` firmy `Virtuosoft` to kolejna dobra opcja, która obejmuje również obsługę powtarzających się gradientów (https://www.virtuosoft.eu/tools/css-gradient-generator/ (https://www.virtuosoft.eu/tools/css-gradient-generator/)).

Step 10

Istnieją trzy sposoby łączenia arkuszy stylów z dokumentem `HTML`: wbudowany w atrybut `style`, osadzony w elemencie `style` oraz jako zewnętrzny dokument `.css` połączony z dokumentem lub zaimportowany do niego. W tej sekcji w końcu dochodzimy do tej trzeciej opcji.

Zewnętrzne arkusze stylów są zdecydowanie najpotężniejszym sposobem korzystania z `CSS`, ponieważ możemy wprowadzać zmiany stylów w całej witrynie, po prostu edytując pojedynczy dokument arkusza stylów. Jest to zaleta posiadania wszystkich informacji o stylu w jednym miejscu, a nie pomieszania ze źródłem dokumentu.

Co więcej, ponieważ jeden dokument stylu jest pobierany i buforowany przez przeglądarkę dla całej witryny, w każdym dokumencie jest mniej kodu do pobrania, co skutkuje lepszą wydajnością.

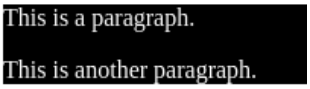
Najpierw trochę o samym dokumencie arkusza stylów. Zewnętrzny arkusz stylów to dokument w postaci zwykłego tekstu z co najmniej jedną regułą arkusza stylów. Może nie zawierać żadnych tagów `HTML` (i tak nie ma powodu, aby je uwzględniać). Może zawierać komentarze, ale muszą używać składni komentarzy `CSS`, którą już widzieliśmy:

```
/* This is the end of the section */
```

Arkusz stylów powinien być nazwany przyrostkiem `.css` (istnieją pewne wyjątki od tej reguły). Może również zacząć się od reguły `@charset at`, aby zadeklarować kodowanie znaków, chociaż naprawdę musimy to zrobić tylko wtedy, gdy używasz kodowania innego niż `UTF-8`. Jeśli używamy `@charset`, musi to być pierwszy element w arkuszu stylów, bez poprzedzających go znaków, w tym komentarzy i reguł stylów. Istnieją dwa sposoby zastosowania zewnętrznego arkusza stylów: element `link` i reguła `@import`.

Element `link` definiuje relację między bieżącym dokumentem a zasobem zewnętrznym. Zdecydowanie najbardziej popularnym zastosowaniem jest linkowanie do arkuszy stylów. Element `link` znajduje się w nagłówku dokumentu, jak pokazano tutaj:

```
<!--link_element_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Link element example</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <div>
    <p>This is a paragraph.</p>
    <p>This is another paragraph.</p>
  </div>
</body>
</html>
```



Musimy dołączyć dwa atrybuty w elemencie `link`:

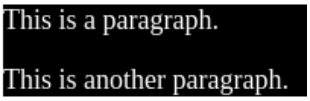
- `rel="stylesheet"` - Definiuje relację połączonego dokumentu z bieżącym dokumentem. Wartością atrybutu `rel` jest zawsze arkusz stylów, gdy łączymy się z arkuszem stylów.
- `href="url"` - Podaje lokalizację pliku `.css`.

Możemy dołączyć wiele elementów linków do różnych arkuszy stylów i wszystkie będą miały zastosowanie. Jeśli wystąpią konflikty, którykolwiek z nich jest wymieniony jako ostatni, zastąpi poprzednie ustawienia ze względu na kolejność reguł i kaskadę.

Inną metodą dołączania zewnętrznego arkusza stylów do dokumentu jest zaimportowanie go za pomocą reguły `@import`. `@import` to inny typ reguły, którą można dodać do arkusza stylów, w zewnętrznym dokumencie arkusza stylów `.css` lub bezpośrednio w elemencie stylu, jak pokazano w poniższym przykładzie:

```
<!--import_style_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Import style example</title>
  <style>
    @import url("styles.css");
    p {
      font-size: 18px;
    }
  </style>
</head>
<body>
  <div>
    <p>This is a paragraph.</p>
    <p>This is another paragraph.</p>
  </div>
</body>
</html>
```

W tym przykładzie wyświetlany jest względny adres `URL`, ale może to być również adres bezwzględny (zaczynający się od `http://`). Reguła `@import` musi znajdować się na początku arkusza stylów przed jakimikolwiek selektorami. Możemy zaimportować więcej niż jeden arkusz stylów i wszystkie zostaną zastosowane, ale reguły z ostatniego arkusza stylów na liście mają pierwszeństwo przed wcześniejszymi.



Możemy również ograniczyć import arkusza stylów do określonych typów mediów (takich jak ekran, wydruk lub projektor, żeby wymienić tylko kilka) lub środowisk wyświetlania (orientacja, rozmiar ekranu itp.) za pomocą **media queries**. **Media queries** to metoda stosowania stylów oparta na medium używanym do wyświetlania dokumentu. Pojawiają się po regule `@import` na liście rozdzielanej

przecinkami. Na przykład, jeśli utworzyliśmy arkusz stylów, który powinien być importowany i używany tylko podczas drukowania dokumentu, użyjmy tej reguły:

```
@import url(print_styles.css) print;
```

Ponieważ można kompilować informacje z wielu zewnętrznych arkuszy stylów, **modułowe arkusze stylów** stały się popularną techniką zarządzania stylami. Wielu programistów przechowuje style, których często używa ponownie — takie jak zabiegi typograficzne, reguły układu lub style związane z formularzami — w osobnych arkuszach stylów, a następnie łączy je w sposób mieszany i dopasowywany za pomocą reguł `@import`. Ponownie, reguły `@import` muszą poprzedzać reguły używające selektorów.

```
<!--modular_styles_example.html-->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Modular styles example</title>
  <style>
    @import url("section.css");
    @import url("article.css");
    body {
      background-color: orange;
    }
  </style>
</head>
<body>
    <section>
        <h1>Section title</h1>
        <article>
        <h2>Article title</h2>
        <p>Article content</p>
        </article>
    </section>
</body>
</html>
```

Jest to dobra technika, o której należy pamiętać podczas zdobywania doświadczenia w tworzeniu witryn. Przekonamy się, że istnieją pewne rozwiązania, które działają dobrze dla nas i fajnie jest nie musieć wymyślać koła na nowo dla każdej nowej witryny. Modułowe arkusze stylów to dobre narzędzie do oszczędzania czasu i organizacji; mogą jednak stanowić problem z wydajnością i buforowaniem. W przypadku korzystania z tej metody zaleca się skompilowanie wszystkich stylów w jeden dokument przed przesłaniem ich do przeglądarki. Nie musisz tego robić ręcznie; istnieją narzędzia, które zrobią to za nas. Preprocesory `LESS` i `Sass CSS` to tylko dwa narzędzia oferujące funkcjonalność kompilacji.