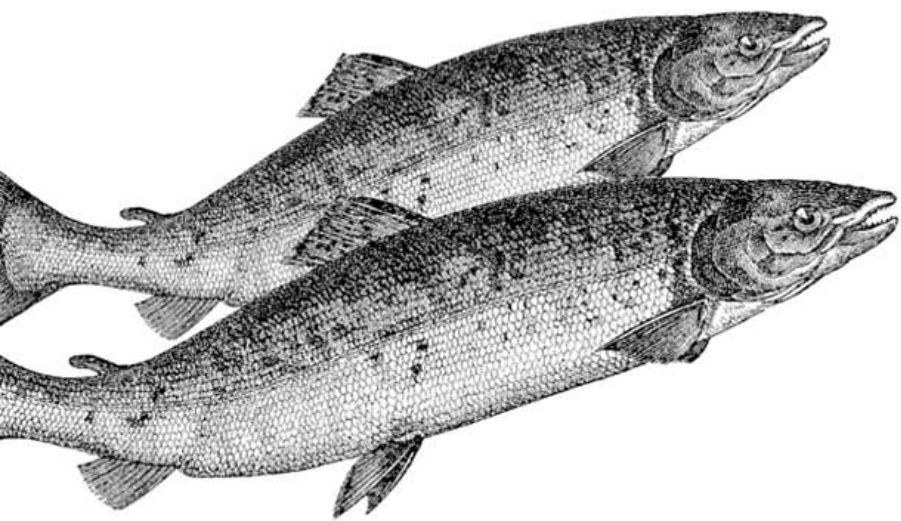
CSS3 zawsze pod ręką!

Wydanie IV

CSS

Leksykon kieszonkowy



O'REILLY®

Eric A. Meyer

HELION

Tytuł oryginału: CSS Pocket Reference. 4th Edition

Tłumaczenie: Jakub Hubisz

ISBN: 978-83-246-3757-7

© 2012 Helion S.A.

Authorized Polish translation of the English edition of "CSS Pocket Reference, 4th Edition" 9781449399030 © 2011 O'Reilly Media, Inc.

This translation is published and sold by permission of O'Reilly Media, Inc., which owns or controls all rights to publish and sell the same.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from the Publisher.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniej-szej publikacji w jakiejkolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficz-ną fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz Wydawnictwo HELION dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz Wydawnictwo HELION nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: helion@helion.pl
WWW: http://helion.pl (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
http://helion.pl/user/opinie/csslk4
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

• Kup książkę
• Poleć książkę
• Oceń książkę

• Księgarnia internetowa
• Lubię to! » Nasza społeczność

Spis treści

Kup książkę Poleć książkę

Kup książkę Poleć książkę

Rozdział 3.
Selektory

Selektory

Selektor uniwersalny

Wzór:

*

Opis:

Ten selektor odpowiada nazwie dowolnego elementu w języku dokumentu. Jeżeli reguła nie ma wyraźnego selektora, domniemywa się, iż należy użyć selektora uniwersalnego.

Przykłady:

```
* {color: red;}
div * p {color:  blue;}
```

Selektor typu

Wzór:

```
element1
```

Opis:

Ten selektor odpowiada nazwie określonego elementu w języku dokumentu. Będzie stosowany w każdym przypadku wystąpienia danego elementu (w CSS1 ten selektor nazywany był selektorem elementu).

Przykłady:

```
body {background: #FFF;}
p {font-size: 1em;}
```

Selektor elementu potomnego

Wzór:

```
element1 element2
```


Kup książkę Poleć książkę

Opis:

Selektor pozwala autorowi na wskazanie elementu na podstawie jego statusu jako elementu potomnego innego elementu. Wskazywany element może być dzieckiem, wnukiem, prawnukiem itp. elementu przodka (w CSS1 selektor ten nazywany był selektorem kontekstowym).

Przykłady:

```
body h1 {font-size: 200%;}
table tr td div ul li {color: purple;}
```

Selektor elementu dziecka

Wzór:

```
element1 > element2
```

Opis:

Selektor wskazuje element na podstawie jego statusu jako dziecka innego elementu. Jest bardziej restrykcyjny niż element elementu potomnego, ponieważ wskaże tylko na element dziecko.

Przykłady:

```
div > p {color:  cyan;}
ul > li {font-weight:  bold;}
```

Selektor elementu bezpośredniego rodzeństwa

Wzór:

```
element1 + element2
```

Opis:

Selektor pozwala autorowi na wskazanie elementu, który jest kolejnym elementem przylegającym innego elementu. Tekst znajdujący się pomiędzy elementami jest ignorowany, pod uwagę brane są tylko elementy i ich pozycje w drzewie dokumentu.

Przykłady:

```
table + p {margin-top: 2.5em;}
h1 + * {margin-top: 0;}
```

Selektor elementu rodzeństwa

Wzór:

```
element1 ~ element2
```


Kup książkę Poleć książkę

Opis:

Selektor pozwala autorowi na wskazanie elementu, który jest podległym tego samego rodzica i następuje po nim w drzewie dokumentu. Tekst lub inne elementy znajdujące się pomiędzy elementami są ignorowane, pod uwagę brane są tylko elementy i ich pozycje w drzewie dokumentu.

Przykłady:

```
h1 ~ h2 {margin-top: 2.5em;}
div#navlinks ~ div {margin-top: 0;}
```

Selektor klasy

Wzór:

```
element1.classname
element1.classname1.classname2
```

Opis:

W językach, które na to zezwalają, takich jak HTML, XHTML, SVG oraz MathML, selektor klasy używający notacji kropkowej może być wykorzystany do wskazania elementów posiadających klasę zwierającą wskazaną wartość (lub wartości). Nazwa klasy musi następować bezpośrednio po kropce. Kilka wartości klas można połączyć w jeden ciąg, jednak Internet Explorer starszy niż wersja 7 ma problemy ze wsparciem dla takiego zapisu. Jeżeli kropka nie jest poprzedzona nazwą elementu, selektor wskaże wszystkie elementy zawierające daną wartość klasy (lub klas).

Przykłady:

```
p.pilne {color: red;}
a.zewnetrzne {font-style: italic;}
.przyklad {background: olive;}
.uwaga.wazne {background: yellow;}
```

Uwaga:

Wersje Internet Explorera starsze niż 7 nie obsługują łączenia klas, pozwalają jednak na zapisanie więcej niż jednej nazwy klasy w atrybucie klasy.

Selektor identyfikatora ID

Wzór:

```
element1#idname
```


Kup książkę Poleć książkę

Opis:

W językach, które na to zezwalają, takich jak HTML lub XHTML, selektor identyfikatora ID korzystający z notacji kratkowej może być użyty do wskazania elementów o ID zawierających określoną wartość lub wartości. Nazwa wartości ID musi następować bezpośrednio po znaku kratki (#). Jeżeli przed tym znakiem nie zostanie zmieszczona nazwa elementu, selektor wskaże wszystkie elementy zawierające daną wartość ID.

Przykłady:

```
h1#tytul-strony {font-size: 250%;}
body#glowna {background: silver;}
#przyklad {background: lime;}
```

Prosty selektor atrybutu

Wzór:

```
element1[attr]
```

Opis:

Pozwala na wskazanie dowolnego elementu na podstawie wartości atrybutu, bez względu na wartość atrybutu.

Przykłady:

```
a[rel] {border-bottom: 3px double gray;}
p[class] {border: 1px dotted silver;}
```

Selektor ścisłej wartości atrybutu

Wzór:

```
element1[attr="value"]
```

Opis:

Pozwala na wskazanie dowolnego elementu na podstawie ścisłej wartości atrybutu.

Przykłady:

```
a[rel="Start"] {font-weight: bold;}
p[class="urgent"] {color: red;}
```

Selektor częściowej wartości atrybutu

Wzór:

```
element1[attr~="value"]
```


Kup książkę Poleć książkę

Opis:

Pozwala na wskazanie dowolnego elementu na podstawie rozdzielonej spacjami części wartości atrybutu. Należy zauważyć, iż `[class~="value"]` nie jest tożsame z `.value` (patrz wyżej).

Przykłady:

```
a[rel~="friend"] {text-transform: uppercase;}
p[class~="warning"] {background: yellow;}
```

Selektor części początkowej wartości atrybutu

Wzór:

```
element1[attr^="ciąg"]
```

Opis:

Pozwala na wskazanie dowolnego elementu na podstawie wystąpienia ciągu na początku wartości atrybutu.

Przykłady:

```
a[href^="/blog"] {text-transform: uppercase;}
p[class^="test-"] {background: yellow;}
```

Selektor części końcowej wartości atrybutu

Wzór:

```
element1[attr$="ciąg"]
```

Opis:

Pozwala na wskazanie dowolnego elementu na podstawie wystąpienia ciągu na końcu wartości atrybutu.

Przykłady:

```
a[href$=".pdf"] {font-style: italic;}
```

Selektor części obowiązkowej wartości atrybutu

Wzór:

```
element1[attr*="ciąg"]
```

Opis:

Pozwala na wskazanie dowolnego elementu na podstawie wystąpienia ciągu w dowolnym miejscu wartości atrybutu.

Kup książkę Poleć książkę

Przykłady:

```
a[href*="helion.pl"] {font-weight: bold;}
div [class*="port"] {border: 1px solid red;}
```

Selektor atrybutu języka

Wzór:

```
element1[lang|="lc"]
```

Opis:

Pozwala na wskazanie dowolnego elementu mającego atrybut `lang`, którego wartość znajduje się na rozdzielonej znakami pionowej kreski liście wartości rozpoczynającej się od wartości wskazanej przez selektor.

Przykłady:

```
html[lang|="tr"] {color:  red;}
```

Pseudoklasy strukturalne

Ściśle rzecz biorąc, wszystkie pseudoklasy (np. wszystkie selektory) są strukturalne — są w pewnej mierze zależne od struktury dokumentu. Poniższe pseudoklasy zostały wyróżnione, ponieważ określają wzorce występujące w strukturze dokumentu, np. wybranie co drugiego akapitu lub ostatnich elementów dzieci jakiegoś elementu.

:empty

Zastosowanie:

Wszystkie elementy

Opis:

Dopasowuje elementy, które nie posiadają węzłów podrzędnych, czyli nie mają elementów podrzędnych ani zawartości. Zawartość może stanowić tekst, białe znaki, symbole encji lub węzły CDDATA. Oznacza to, że element `<p> </p>` *nie* jest pusty; nie będzie pusty także w przypadku, gdy spacja zostanie zastąpiona znakiem nowej linii. Ta pseudoklasa nie odnosi się do elementów pustych, takich jak `br`, `img`, `input` itp.

Przykłady:

```
p:empty {padding: 1em; background: red;}
li:empty {display: none;}
```


Kup książkę Poleć książkę

:first-child

Zastosowanie:

Wszystkie elementy

Opis:

Dopasowuje elementy będące pierwszym dzieckiem innego elementu. Oznacza to, że zapis `div:first-child` spowoduje dopasowanie wszystkich elementów `div` będących pierwszymi dziećmi innych elementów, *nie* pierwszych dzieci dowolnego elementu `div`.

Przykłady:

```
td:first-child {border-left: 1px solid;}
p:first-child {text-indent: 0; margin-top: 2em;}
```

:first-of-type

Zastosowanie:

Wszystkie elementy

Opis:

Dopasowuje elementy będące pierwszym dzieckiem danego typu. Czyli zapis `div:first-of-type` spowoduje dopasowanie elementów `div` będących pierwszymi elementami dziećmi typu `div` dla innego elementu.

Przykłady:

```
td:first-of-type {border-left: 1px dotted;}
h2:first-of-type {color: fuchsia;}
```

:lang

Zastosowanie:

Dowolny element z przypisaną informacją o języku

Opis:

Selektor wskazuje element na podstawie informacji o języku (chodzi o język ludzki, a nie język programowania). Informacja tego rodzaju musi być zawarta w dokumencie lub w inny sposób z nim połączona i nie może być zdefiniowana za pomocą CSS. Obsługa selektora `:lang` jest taka sama jak selektora atrybutu `|=`. Przykładowo w dokumencie HTML język elementu jest określony atrybutem `lang`. Jeżeli element nie posiada tego atrybutu, język elementu określany jest

Kup książkę Poleć książkę

za pomocą ostatniego atrybutu `lang` najbliższego elementu przodka, a jeśli go brak — przez pole nagłówka HTTP `Content-Language` (lub odpowiedni znacznik meta `http-equiv`) dokumentu.

Przykłady:

```
html:lang(en) {background: silver;}
*:lang(fr) {quotes: '&#171; ' ' &#187;';}
```

:last-child

Zastosowanie:

Wszystkie elementy

Opis:

Dopasowuje elementy będące ostatnim dzieckiem innego elementu. Oznacza to, że zapis `div:last-child` spowoduje dopasowanie wszystkich elementów `div` będących ostatnimi dziećmi innych elementów, *nie* ostatnich dzieci dowolnego elementu `div`.

Przykłady:

```
td:last-child {border-right: 1px solid;}
p:last-child {margin-bottom: 2em;}
```

:last-of-type

Zastosowanie:

Wszystkie elementy

Opis:

Dopasowuje elementy będące ostatnim dzieckiem danego typu. Czyli zapis `div:last-of-type` spowoduje dopasowanie elementów `div` będących ostatnimi elementami dziećmi typu `div` dla innego elementu.

Przykłady:

```
td:last-of-type {border-right: 1px dotted;}
h2:last-of-type {color: fuchsia;}
```

:nth-child(an+b)

Zastosowanie:

Wszystkie elementy

Kup książkę Poleć książkę

Opis:

Dopasowuje każdy *n*-ty element dziecko według wzorca zdefiniowanego przez formułę `an+b`, gdzie `a` i `b` to liczby całkowite, a `n` reprezentuje nieskończoną serię liczb całkowitych liczonych w przód od pierwszego elementu dziecka. Czyli aby wybrać co czwarty element dziecko dla elementu `body`, rozpoczynając od pierwszego dziecka, należy zapisać: `body > *:nth-child(4n+1)`. To spowoduje wybieranie elementów będących pierwszym, piątym, dziewiątym, trzynastym itd. dzieckiem elementu `body`. Jeżeli powinien zostać wybrany element czwarty, ósmy, dwunasty itd., należy zmienić powyższy zapis na następujący: `body > *:nth-child(4n)`. Możliwe jest także wykorzystanie jako `b` liczb ujemnych — zapis `body > *:nth-child(4n-1)` spowoduje wybranie elementów trzeciego, siódmego, jedenastego, piętnastego itd.

W miejscu formuły `an+b` możliwe jest wykorzystanie dwóch słów kluczowych: `even` (parzyste) i `odd` (nieparzyste). Są one kolejno synonimami zapisów `2n` i `2n+1`.

Przykłady:

```
*:nth-child(4n+1) {font-weight: bold;}
tbody tr:nth-child(odd) {background-color: #EEF;}
```

:nth-last-child(an+b)

Zastosowanie:

Wszystkie elementy

Opis:

Dopasowuje każdy *n*-ty element dziecko według wzorca zdefiniowanego przez formułę `an+b`, gdzie `a` i `b` to liczby całkowite, a `n` reprezentuje nieskończoną serię liczb całkowitych *liczonych wstecz od ostatniego elementu dziecka*. Czyli aby wybrać co czwarty od końca element dziecko elementu `body`, należy zapisać: `body > *:nth-last-child(4n+1)`. Można powiedzieć, że jest to pseudoklasa lustrzana do `:nth-child`.

W miejscu formuły `an+b` możliwe jest wykorzystanie dwóch słów kluczowych: `even` (parzyste) i `odd` (nieparzyste). Są one kolejno synonimami zapisów `2n` i `2n+1`.

Przykłady:

```
*:nth-last-child(4n+1) {font-weight: bold;}
tbody tr:nth-last-child(odd) {background-color: #EEF;}
```


Kup książkę Poleć książkę

:nth-last-of-type(an+b)

Zastosowanie:

Wszystkie elementy

Opis:

Dopasowuje każdy *n*-ty element typu zgodnego z elementem nazwanym według wzorca zdefiniowanego przez formułę `an+b`, gdzie `a` i `b` to liczby całkowite, a `n` reprezentuje nieskończoną serię liczb całkowitych *liczonych wstecz od ostatniego takiego elementu*. Czyli aby wybrać co trzeci od końca element akapitu (`p`) będący dzieckiem elementu `body`, zaczynając od ostatniego takiego akapitu, należy zapisać: `body > p:nth-last-of-type(3n+1)`. Kolejne akapity będą wybierane, nawet jeżeli pomiędzy nimi znajdą się inne elementy, np. listy lub tabele.

W miejscu formuły `an+b` możliwe jest wykorzystanie dwóch słów kluczowych: `even` (parzyste) i `odd` (nieparzyste). Są one kolejno synonimami zapisów `2n` i `2n+1`.

Przykłady:

```
td:nth-last-of-type(even) {background-color: #FCC;}
img:nth-last-of-type(3n) {float: left; border: 2px solid;}
```

:nth-of-type(an+b)

Zastosowanie:

Wszystkie elementy

Opis:

Dopasowuje każdy *n*-ty element typu zgodnego z elementem nazwanym według wzorca zdefiniowanego przez formułę `an+b`, gdzie `a` i `b` to liczby całkowite, a `n` reprezentuje nieskończoną serię liczb całkowitych *liczonych w przód od pierwszego takiego elementu*. Czyli aby wybrać co trzeci element akapitu (`p`) będący dzieckiem elementu `body`, zaczynając od pierwszego takiego akapitu, należy zapisać: `body > p:nth- of-type(3n+1)`. Kolejne akapity będą wybierane, nawet jeżeli pomiędzy nimi znajdą się inne elementy, np. listy lub tabele.

W miejscu formuły `an+b` możliwe jest wykorzystanie dwóch słów kluczowych: `even` (parzyste) i `odd` (nieparzyste). Są one kolejno synonimami zapisów `2n` i `2n+1`.

Przykłady:

```
td:nth-of-type(even) {background-color: #FCC;}
img:nth-of-type(3n) {float: right; margin-left: 1em;}
```


Kup książkę Poleć książkę

:only-child

Zastosowanie:

Wszystkie elementy

Opis:

Dopasowuje element będący jedynym elementem dzieckiem dla swojego elementu nadrzędnego. Powszechnym zastosowaniem jest usunięcie obramowania obrazka wewnątrz linku, przy założeniu, że obrazek jest jedynym elementem wewnątrz elementu linku. Należy zauważyć, że element zostanie wybrany, nawet jeżeli posiada własne elementy dzieci; istotne jest jedynie to, że jest jedynym dzieckiem swojego elementu nadrzędnego.

Przykłady:

```
a img:only-child {border: 0;}
table div:only-child {margin: 5px;}
```

:only-of-type

Zastosowanie:

Wszystkie elementy

Opis:

Dopasowuje element będący jedynym elementem dzieckiem danego typu dla swojego elementu nadrzędnego. Należy zauważyć, że element zostanie wybrany, nawet jeżeli posiada własne elementy dzieci; istotne jest jedynie to, że jest jedynym dzieckiem danego typu dla swojego elementu nadrzędnego.

Przykłady:

```
p em:only-of-type {font-weight: bold;}
section article:only-of-type {margin: 2em 0 3em;}
```

:root

Zastosowanie:

Wszystkie elementy

Opis:

Dopasowuje główny element, czyli korzeń dokumentu, którym w przypadku HTML-a i XHTML-a jest zawsze element `html`. W dokumencie XML korzeń dokumentu może mieć dowolną nazwę i w związku z tym selektor może być potrzebny.

Kup książkę Poleć książkę

Przykłady:

```
:root {font: medium serif;}
:root > * {margin: 1.5em 0;}
```

Pseudoklasy negujące

Istnieje tylko jedna negująca pseudoklasa, jest ona jednak tak specyficzna, że zasługuje na swoją własną sekcję.

:not(e)

Zastosowanie:

Wszystkie elementy

Opis:

Dopasowuje dowolny element, który *nie* jest opisany przez selektor w nawiasach *e*. Pozwala to np. na odrzucenie wszystkich elementów niebędących akapitami poprzez zapis `*:not(p)`. Negacja może być wykorzystywana razem z selektorami elementów potomnych. Przykładem takiego wykorzystania może być zapis `table *:not(td)`, pozwalający na wybranie wszystkich elementów wewnątrz tabeli niebędących jej komórkami. Innym przykładem może być wybranie wszystkich elementów, których atrybut ID ma wartość inną niż „szukaj", poprzez zapis `[id]:not([id="search"])`.

Jedynym wyjątkiem, który nie może być wykorzystany jako *e*, jest pseudoklasa negacji. Niedopuszczalny jest zapis `:not(:not(div))`. Nie jest to jednak problemem, ponieważ synonimem tego zapisu będzie po prostu `div`.

Ponieważ `:not()` jest pseudoklasą, może być łączona z innymi pseudoklasami oraz z instancjami siebie samej. Na przykład aby wybrać dowolny element posiadający fokus, niebędący elementem typu `a`, należy zastosować zapis `*:focus:not(a)`. Aby wybrać dowolny element niebędący ani akapitem, ani elementem `div`, wystarczy zapisać `*:not(p):not(div)`.

W połowie roku 2011 wewnątrz wyrażeń `:not()` nie można było stosować selektorów elementu potomnego oraz selektorów grupujących. Ta restrykcja prawdopodobnie zostanie zniesiona w przyszłych wersjach CSS.

Kup książkę Poleć książkę

Przykłady:

```
ul *:not(li) {text-indent: 2em;}
fieldset *:not([type="checkbox"]):not([type="radio"])
{margin: 0 1em;}
```

Pseudoklasy interakcji

Pseudoklasy wymienione poniżej odnoszą się do interakcji użytkownika z dokumentem: przypisywania stylów dla różnych stanów łącza, oznaczania elementu będącego celem identyfikatora fragmentu lub przypisania stylów w zależności od tego, czy element formularza jest włączony, czy wyłączony.

:active

Zastosowanie:

Element aktywny

Opis:

Ten rodzaj pseudoklasy ma zastosowanie do elementu w czasie, gdy jest on aktywny. Najpopularniejszym przykładem jest kliknięcie łącza w dokumencie HTML. Podczas przytrzymywania wciśniętego przycisku myszy łącze jest aktywne. Teoretycznie możliwe są inne sposoby aktywowania elementów oraz inne elementy, które mogą być aktywowane, jednak CSS ich nie definiuje.

Przykłady:

```
a:active {color:  red;}
*:active {background:  blue;}
```

:checked

Zastosowanie:

Wszystkie elementy

Opis:

Dopasowuje dowolny element na interfejsie, który został zaznaczony, np. zaznaczone pole typu checkbox lub wybrany przycisk radio.

Przykłady:

```
input:checked {outline: 3px solid rgba(127,127,127,0.5);}
input[type="checkbox"]:checked {box-shadow: red 0 0 5px;}
```


Kup książkę Poleć książkę

:disabled

Zastosowanie:

Wszystkie elementy

Opis:

Dopasowuje dowolny element na interfejsie, który nie przyjmuje interakcji użytkownika z powodu wybranych atrybutów języka lub innych niezwiązanych z prezentacją, np. `<input type="text" disabled>` w HTML5. Należy zauważyć, że `:disabled` nie ma zastosowania, jeżeli element został usunięty z widocznego obszaru poprzez wykorzystanie właściwości `position` lub `display`.

Przykłady:

```
input:disabled {opacity: 0.5;}
```

:enabled

Zastosowanie:

Wszystkie elementy

Opis:

Dopasowuje dowolny element na interfejsie, który może przyjmować interakcję użytkownika oraz może być oznaczony jako włączony lub wyłączony w kodzie dokumentu. Dotyczy dowolnych elementów formularza do wprowadzania danych, ale nie dotyczy linków.

Przykłady:

```
input:enabled {background: #FCC;}
```

:focus

Zastosowanie:

Element, który posiada fokus

Opis:

Selektor ma zastosowanie do elementu w czasie, gdy ten posiada fokus. Przykładem w HTML-u jest wejściowe pole tekstowe, w którym znajduje się kursor wpisywania tekstu, tak iż w chwili rozpoczęcia pisania tekst znajdzie się w polu. Inne elementy, takie jak łącza, także mogą mieć fokus, CSS nie definiuje jednak, które to elementy.

Kup książkę Poleć książkę

Przykłady:

```
a:focus {outline: 1px dotted red;}
input:focus {background: yellow;}
```

Uwagi:

W przeglądarce Internet Explorer `:focus` dotyczy wyłącznie linków — nie dotyczy kontrolek formularza.

:hover

Zastosowanie:

Element, na który najechano wskaźnikiem myszy

Opis:

Selektor ma zastosowanie do elementu, który został *wskazany* przez użytkownika, jednak nie został jeszcze aktywowany. Najpopularniejszym przykładem jest przesunięcie wskaźnika myszy w granice łącza w dokumencie HTML. Teoretycznie selektor może być stosowany do innych elementów, jednak CSS ich nie definiuje.

Przykłady:

```
a[href]:hover {text-decoration: underline;}
p:hover {background: yellow;}
```

Uwagi:

W przeglądarce Internet Explorer w wersjach starszych niż 7 `:hover` dotyczy wyłącznie linków.

:link

Zastosowanie:

Łącze do innego zasobu, które nie zostało jeszcze odwiedzone

Opis:

Selektor stosowany do łącza do URI, które nie zostało jeszcze odwiedzone. Oznacza to, iż URI, do którego prowadzi łącze, nie znajduje się w historii przeglądarki. Stan ten wyklucza się wzajemnie ze stanem `:visited`.

Przykłady:

```
a:link {color: blue;}
*:link {text-decoration: underline;}
```


Kup książkę Poleć książkę

:target

Zastosowanie:

Wszystkie elementy

Opis:

Dopasowuje element, który odpowiada identyfikatorowi fragmentu z adresu URI wykorzystanego do otwarcia strony. Czyli *http://www.w3.org/TR/css3-selectors/#target-pseudo* zostałoby dopasowane przez `:target` i przypisałoby zadeklarowane style do wszystkich elementów o id równym `target-pseudo`. Jeżeli element ten byłby akapitem, zostałby także dopasowany przez `p:target`.

Przykłady:

```
:target {background: #EEO;}
```

:visited

Zastosowanie:

Łącze do innego zasobu, które zostało już odwiedzone

Opis:

Selektor stosowany do łącza do URI, które zostało już odwiedzone. Adres URI, do którego prowadzi łącze, znajduje się w historii przeglądarki. Stan ten wyklucza się wzajemnie ze stanem `:link`.

Przykłady:

```
a:visited {color: purple;}
*:visited {color: gray;}
```

Pseudoelementy

W CSS1 i CSS2 pseudoelementy były poprzedzone pojedynczym dwukropkiem, tak jak pseudoklasy. W CSS3 pseudoelementy poprzedzone są podwójnym dwukropkiem, aby odróżnić je od pseudoklas. Ze względów historycznych przeglądarki obsługują oba zapisy, jednak zalecana jest składnia z dwoma dwukropkami.

::after

Tworzy:

Pseudoelement zawierający wygenerowaną zawartość umieszczoną za zawartością elementu

Kup książkę Poleć książkę

Opis:

Pozwala autorowi na wstawianie wygenerowanej zawartości na końcu istniejącej zawartości elementu. Domyślnie będzie to pseudoelement wierszowy, ale można to zmienić za pomocą właściwości `display`.

Przykłady:

```
a.external:after {content:  "  "  url(/icons/globe.gif);)
p:after {content:  "   |   ";}
```

::before

Tworzy:

Pseudoelement zawierający wygenerowaną zawartość umieszczoną przed zawartością elementu

Opis:

Pozwala autorowi na wstawianie wygenerowanej zawartości na początku istniejącej zawartości elementu. Domyślnie będzie to pseudoelement wierszowy, ale można to zmienić za pomocą właściwości `display`.

Przykłady:

```
a[href]:before {content: "[LINK] ";)
p:before {content: attr(class);}
a[rel~="met"]:before {content: " *";}
```

::first-letter

Tworzy:

Pseudoelement zawierający pierwszą literę elementu

Opis:

Selektor używany do nadania stylu pierwszej literze elementu. Rozpoczynające znaki przestankowe uzyskają styl razem z pierwszą literą. Niektóre języki używają kombinacji liter, które powinny być traktowane jako pojedynczy znak. Przed CSS2.1 selektor `:first-letter` mógł być dołączany tylko do elementów blokowych. CSS2.1 umożliwił użycie go dla elementów blokowych, pozycji na liście, komórki tabeli, nagłówka tabeli oraz elementu wierszowego. Do pierwszej litery można zastosować tylko ograniczony zestaw właściwości.

Przykłady:

```
h1:first-letter {font-size: 166%;}
p:first-letter {text-decoration: underline;}
```


Kup książkę Poleć książkę

::first-line

Tworzy:

Pseudoelement zawierający pierwszy sformatowany wiersz elementu

Opis:

Selektor używany do nadania stylu pierwszemu wierszowi tekstu elementu. Bez znaczenia jest, ile słów zawiera dany wiersz. Selektor może być dołączony tylko do elementów blokowych. Do pierwszego wiersza można zastosować jedynie ograniczony zestaw właściwości.

Przykłady:

```
p.lead:first-line {font-weight: bold;}
```

Zapytania o media

Przy wykorzystaniu zapytań o media autor może zdefiniować środowisko, w jakim arkusz stylów będzie wykorzystany przez przeglądarkę. W przeszłości było to wykonywane poprzez wykorzystanie atrybutu `media` w elemencie `link` lub poprzez deskryptor mediów w deklaracji `@import`. Zapytania o media pozwalają na znacznie więcej: pozwalają autorom wybrać odpowiedni arkusz stylów w zależności od cech danego typu mediów.

Podstawowe założenia

Położenie zapytań o media będzie wyglądało znajomo dla każdego, kto kiedykolwiek zetknął się z ustalaniem typów mediów. Poniżej zaprezentowane są dwa sposoby wykorzystania arkusza stylów podczas renderowania dokumentu na kolorowej drukarce:

```
<link href="print-color.css" type="text/css" media="print and
↪(color)" rel="stylesheet">
@import url(print-color.css) print and (color);
```

Zapytanie o media może być wykorzystane w każdym miejscu, w którym może zostać ustalony typ mediów. Oznacza to, że możliwe jest umieszczenie więcej niż jednego zapytania na liście oddzielonej przecinkami:

```
<link href="print-color.css" type="text/css" media="print and
↪(color), projection and (color)" rel="stylesheet">
@import url(print-color.css) print and (color), projection and
↪(color);
```


Kup książkę Poleć książkę

W sytuacji gdy którekolwiek z zapytań zwróci „prawdę”, zostanie wykorzystany przypisany plik arkusza stylów. Czyli w poprzednim przykładzie plik `print-color.css` zostanie wykorzystany, jeżeli dokument renderowany będzie na kolorowej drukarce lub kolorowym projektorze. Jeżeli dokument będzie drukowany na czarno-białej drukarce, oba zapytania zwrócą „fałsz” i arkusz stylów nie zostanie wykorzystany w dokumencie. To samo dotyczy ekranów, czarno-białych projektorów, mediów dźwiękowych itd.

Każde zapytanie o media składa się z typu mediów oraz jednej lub większej liczby cech. Każda z cech otoczona jest nawiasami okrągłymi, a jeżeli jest ich więcej niż jedna, są rozdzielane słowem kluczowym `and`. W zapytaniach o media wykorzystywane są dwa logiczne słowa kluczowe:

`and`
: Łączy ze sobą dwie lub więcej cech mediów w taki sposób, że wszystkie muszą zaistnieć, aby całe zapytanie było zrealizowane. Na przykład definicja `(color) and (orientation: landscape) and (min-device-width: 800px)` oznacza, że aby została zrealizowana, medium musi obsługiwać kolory, mieć orientację poziomą, a szerokość musi wynosić minimum 800 pikseli.

`not`
: Neguje całe zapytanie: jeżeli wszystkie warunki są spełnione, arkusz stylów *nie* zostanie wykorzystany. Zapis `(color) and (orientation: landscape) and (min-device-width: 800px)` oznacza, że jeżeli wszystkie warunki są spełnione, zapytanie przyjmuje wartość „fałsz”. Czyli jeżeli medium obsługuje kolory, ma orientację poziomą, a szerokość wynosi minimum 800 pikseli, arkusz stylów jest pomijany. Słowo kluczowe `not` może być wykorzystane jedynie na początku zapytania. *Niepoprawna* będzie definicja `(color) and not (mid-device-width: 800px)`. W takich przypadkach zapytanie zostanie zignorowane. Przeglądarki zbyt stare, aby obsłużyć zapytania o media, zawsze pominą deklaracje rozpoczynające się od `not`.

Wewnątrz zapytania nie ma możliwości wykorzystania słowa kluczowego `or`. W zapytaniach funkcję tę pełnią przecinki — zapis `screen, print` oznacza: „Wykorzystaj arkusz, jeżeli medium to ekran lub drukarka”. Czyli zamiast zapisu `screen and (max-color: 2) or (monochrome)`, który jest niepoprawny i, co za tym idzie, ignorowany, należy użyć `screen and (max-color: 2), screen and (monochrome)`.

Kup książkę Poleć książkę

Jest jeszcze jedno słowo kluczowe: `only`, które zostało stworzone w celu zapewnienia niekompatybilności wstecz.

`only`
: Wykorzystywane, aby ukryć arkusz stylów przed przeglądarkami zbyt starymi, by zrozumiały zapytania o media. Na przykład aby arkusz stylów został przypisany do wszystkich mediów, ale tylko dla przeglądarek, które rozumieją zapytania o media, należałoby napisać: `@import url(new.css) only all`. W przeglądarkach, które rozumieją zapytania o media, słowo kluczowe `only` jest ignorowane. Słowo kluczowe `only` może być wykorzystane tylko na początku zapytania.

Wartości zapytań o media

Dla zapytań o media zostały wprowadzone dwa nowe typy mediów, które (w połowie roku 2011) nie są wykorzystywane w żadnym innym kontekście.

`<proporcja>`
: Wartość proporcji podawana jest jako dwie liczby całkowite oddzielone ukośnikiem (/) z opcjonalną spacją. Pierwsza wartość odnosi się do szerokości, druga do wysokości. Czyli aby zapisać stosunek szerokości do wysokości wynoszący 16:9, należy napisać `16/9` lub `16 / 9`.

`<rozdzielczość>`
: Rozdzielczość jest dodatnią liczbą całkowitą, po której następuje znak jednostki: `dpi` lub `dpcm`. Między liczbą a jednostką nie jest dozwolona spacja.

Cechy mediów

Żadna z poniższych wartości nie może być ujemna.

`width`, `min-width`, `max-width`
wartości: *`<szerokość>`*
: Odnosi się do szerokości obszaru roboczego. Dla przeglądarki internetowej będzie to obszar strony wraz z ewentualnymi paskami przewijania. W mediach stronicowanych jest to szerokość ramki strony. Czyli `min-width: 850px` jest spełnione, jeżeli obszar roboczy ma przynajmniej 850 pikseli szerokości.

Kup książkę Poleć książkę

`device-width`, `min-device-width`, `max-device-width`
wartości: `<szerokość>`

Odnosi się do całkowitej szerokości renederowania urządzenia wyjściowego. W mediach ekranowych jest to szerokość ekranu. W mediach stronicowanych jest to całkowita szerokość strony. Czyli `max-device-width: 1200px` będzie spełnione, jeżeli całkowita powierzchnia wyjściowa ma nie więcej niż 1200 pikseli szerokości.

`height`, `min-height`, `max-height`
wartości: `<wysokość>`

Odnosi się do wysokości obszaru roboczego. Dla przeglądarki internetowej będzie to obszar strony wraz z ewentualnymi paskami przewijania. W mediach stronicowanych jest to wysokość ramki strony. Czyli `min-height: 567px` jest spełnione, jeżeli obszar roboczy ma przynajmniej 567 pikseli wysokości.

`device-height`, `min-device-height`, `max-device-height`
wartości: `<wysokość>`

Odnosi się do całkowitej wysokości renederowania urządzenia wyjściowego. W mediach ekranowych jest to wysokość ekranu. W mediach stronicowanych jest to całkowita wysokość strony. Czyli `max-device-height: 400px` będzie spełnione, jeżeli całkowita powierzchnia wyjściowa ma nie więcej niż 400 pikseli wysokości.

`aspect-ratio`, `min-aspect-ratio`, `max-aspect-ratio`
wartości: `<proporcja>`

Odnosi się do proporcji wynikającej ze stosunku szerokości medium do jego wysokości. Czyli `min-aspect-ratio: 2/1` odnosi się do tych mediów, dla których stosunek szerokości do wysokości wynosi 2:1.

`device-aspect-ratio`, `min-device-aspect-ratio`, `max-device-aspect-ratio`

wartości: `<proporcja>`

Odnosi się do proporcji wynikającej ze stosunku całkowitej szerokości medium do jego całkowitej wysokości. Czyli `device-aspect-ratio: 16/9` odnosi się do tych mediów, dla których stosunek tych wartości wynosi 16:9.

`color`, `min-color`, `max-color`
wartości: `<liczba-całkowita>`

Odnosi się do zdolności wyświetlania kolorów przez medium wyjściowe, z opcjonalną liczbą reprezentującą liczbę bitów wykorzystywaną do zapisu składowej koloru. Czyli `color` odnosi

Kup książkę Poleć książkę

się do wszystkich mediów, które potrafią wyświetlać kolory, natomiast `min-color: 4` oznacza, że dla każdej ze składowych koloru muszą być wykorzystane przynajmniej cztery bity. Dla każdego urządzenia nieobsługującego kolorów zostanie zwrócone `0`.

`color-index`, `min-color-index`, `max-color-index`
wartości: *<liczba-całkowita>*

Odnosi się do całkowitej liczby kolorów dostępnej w tabeli sprawdzania kolorów dla urządzenia wyjściowego. Jeżeli urządzenie nie korzysta z tabeli sprawdzania kolorów, zwrócone zostanie `0`. Czyli `min-colorindex: 256` odnosi się do wszystkich mediów, dla których dostępne jest minimum 256 kolorów.

`monochrome`, `min-monochrome`, `max-monochrome`
wartości: *<liczba-całkowita>*

Odnosi się do urządzeń monochromatycznych, z opcjonalną liczbą bitów na piksel w buforze ramki. Każde urządzenie, które nie jest monochromatyczne, zwróci `0`. Czyli `monochrome` odnosi się do każdego monochromatycznego urządzenia, natomiast `min-monochrome:2` odnosi się do urządzeń monochromatycznych wykorzystujących przynajmniej dwa bity na piksel w buforze ramki.

`resolution`, `min-resolution`, `max-resolution`
wartości: *<rozdzielczość>*

Odnosi się do rozdzielczości urządzenia wyjściowego w ujęciu gęstości pikseli, mierzonej liczbą pikseli na cal (`dpi`) lub punktów na centymetr (`dpcm`). Jeżeli urządzenie wyjściowe ma piksele, które nie są kwadratowe, wykorzystywana jest najmniej gęsta oś. Na przykład jeżeli urządzenie ma `120dpcm` na jednej osi i `100dpcm` na drugiej, to wykorzystana zostanie wartość `100`. Dodatkowo cecha `resolution` bez wartości nigdy nie będzie dopasowana (`min-resolution` i `max-resolution` mogą zostać dopasowane).

`orientation`
wartości: `portrait | landscape`

Odnosi się do całkowitego obszaru wyświetlania urządzenia wyjściowego. Wartość `portrait` jest zwracana, jeżeli wysokość jest równa lub większa niż szerokość, w przeciwnym razie zwracane jest `landscape`.

`scan`
wartości: `progressive | interlace`

Odnosi się do procesu skanowania dla urządzeń o typie medium `tv`.

Kup książkę Poleć książkę

`grid`
wartości: `0 | 1`

Odnosi się do obecności (lub jej braku) wyświetlania opartego na siatce w urządzeniach takich, jak terminal `tty`. Urządzenie oparte na siatce zwróci `1`, w przeciwnym razie zwrócone zostanie `0`.

Kup książkę Poleć książkę

Skorowidz

Kup książkę Poleć książkę

C

D

E

F

H

I

J

Kup książkę Poleć książkę

Kup książkę Poleć książkę

Q

R

S

T

Kup książkę Poleć książkę

U

V

W

Z

Kup książkę Poleć książkę

PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ HELION

1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW
w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!
http://program-partnerski.helion.pl

GRUPA WYDAWNICZA
Helion SA

WWW

O'REILLY®

CSS. Leksykon kieszonkowy. Wydanie IV

Kaskadowe arkusze stylów liczą sobie już dobre parenaście lat – ich pierwsza wersja została opublikowana w roku 1996. Dzięki wprowadzeniu tego pomysłowego rozwiązania udało się w końcu uporządkować kod stron internetowych. Co prawda zajęło to sporo czasu, ale obecnie większość witryn poprawnie oddziela warstwę prezentacji od zawartości strony. Pozwala to na większą elastyczność, dopasowanie stron WWW do urządzeń, na których są oglądane, oraz łatwiejszy dostęp do informacji dla osób z dysfunkcjami. Obecnie do przeglądarek odważnie wchodzi trzecia wersja CSS.

Ten niezwykle użyteczny leksykon kieszonkowy poświęcony CSS zawiera wszystkie informacje na temat dostępnych właściwości oraz atrybutów kaskadowych arkuszy stylów. Jego najnowsze wydanie uwzględnia wiele poprawek dotyczących zauważonych niezgodności. Zostało też rozszerzone o nowości, które pojawiły się wraz z CSS3. Dzięki tej książce możesz mieć zawsze pod ręką kompletny zbiór informacji na temat formatowania tekstu, pozycjonowania elementów, tworzenia układów, wykorzystania selektorów i kontenerów. Niezależnie od miejsca i czasu błyskawicznie sprawdzisz, jak ustawić kolor elementu, dziedziczyć właściwości oraz sterować wysokością wierszy. Jest to idealna pozycja dla każdego szanującego swój czas webmastera!

- Dodawanie arkuszy stylów do dokumentów HTML/XHTML
- Struktura reguł
- Komentowanie arkusza CSS
- Dziedziczenie właściwości
- Klasyfikacja elementów
- Pozycjonowanie elementów
- Dostępne właściwości i atrybuty
- Selektory
- Pseudoklasy
- Wykaz właściwości

CSS3 – prosty, efektowny i wydajny!

sięgnij po WIĘCEJ

helion.pl
katalogi
internetowe

Nr katalogowy: 7655

Księgarnia internetowa:
http://helion.pl

Zamówienia telefoniczne:
0 801 339900
0 601 339900

Helion
Sprawdź najnowsze promocje:
http://helion.pl/promocje
Książki najchętniej czytane:
http://helion.pl/bestsellery
Zamów informacje o nowościach:
http://helion.pl/nowosci

Helion SA
ul. Kościuszki 1c, 44-100 Gliwice
tel.: 32 230 98 63
e-mail: helion@helion.pl
http://helion.pl

ISBN 978-83-246-3757-7

Cena 39,00 zł

Informatyka w najlepszym wydaniu