# Wprowadzenie

CSS (ang. Cascading Style Sheets) jest językiem służącym do opisu sposobu prezentowania informacji na stronach WWW, który został opracowany przez organizację [W3C](http://www.w3c.org) . Często skrót ten tłumaczony jest na język polski jako "Kaskadowe arkusze stylów".  
Głównym celem wprowadzenia CSS była chęć rozdzielenia zawartości merytorycznej strony HTML od sposobu jej prezentacji.  
Dzięki prezentacji niezależnej od treści dokumentu można wielu dokumentom łatwo nadać ten sam spójny styl. Można też jeden dokument przedstawić na różne sposoby — np. inaczej na ekranie komputera, inaczej do druku, a jeszcze inaczej na małych ekranach urządzeń przenośnych. [idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Definicja reguły stylu

Atrybuty formatowania w języku CSS definiuje się za pomocą tzw. reguł stylów. Każda reguła odnosi się do konkretnego elementu w dokumencie.  
Definicja reguły stylu nie jest skomplikowaną konstrukcją. Jej składania została przedstawiona poniżej.

selektor\_X { właściwość\_A : wartość\_A; właściwość\_B : wartość\_B; ... }

selektor

definiuje do jakiego elementu (np. nagłówek, paragraf, tabela) stosowana będzie dana reguła

właściwość

określa jaka cecha elementu będzie zmieniana przez daną regułę (np. kolor, krój pisma, marginesy)

wartość

określa wartość jaka zostanie przypisana do wybranej cechy wskazanego elementu

Przykład:  
Żeby zmienić tło całego dokumentu HTML należy ustalić odpowiednią wartość atrybutu bgcolor znacznika <body>.

<body bgcolor="blue">

To samo za pomocą stylu CSS można osiągnąć pisząc:

body {background-color: blue;}

[idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Jednostki miar stosowane w CSS

Kaskadowe arkusze stylów pozwalają nam na określenie wielkości wielu elementów, dopuszczając przy tym różne jednostki miar względnych i bezwzględnych.

* **Jednostki względne**

px — piksele

Jednostka ta opiera się na pojedynczych punktach widocznych na ekranie monitora — pikselach.

em — proporcje wysokości do czcionki danego elementu

Zasada działania tej jednostki polega na określaniu zależności pomiędzy poszczególnymi wielkościami. Jeżeli zdefiniujemy domyślną czcionkę o wielkości 12 pt, to będzie ona równa 1em.

ex — proporcje do wysokości litery x

Stosowanie jednostki ex wiąże się z rodzajem użytej czcionki.

% — procenty

Procenty służą do określania wielkości względem wartości domyślnej.

* **Jednostki bezwzględne**

in — cale

Cale wywodzą się z amerykańskiego systemu miarowego i głównie tam są wykorzystywane.

pt — punkty

Punkty wywodzą się z typografii, gdzie są standardową jednostką miary. W praktyce 72 punkty odpowiadają jednemu calowi, a ten z kolei równa się 2,54 cm.

cm — centymetry

Centymetr jest miarą stosowaną w większości krajów na świecie (1 cm = 0,39 in).

mm — milimetry

Milimetry są jednostką mniejszą niż centymetr. Występują w systemie metrycznym, na jeden centymetr składa się 10 mm.

pc — pica

Podobnie jak punkt, jest jednostką typograficzną. 1 pica odpowiada 12 punktom.

[idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Nazewnictwo kolorów używane w CSS

* nazwa koloru (podobnie jak w HTML)
* DIV {color:red;}
* określenie składowych koloru w modelu RGB w sposób bezwzględny
* DIV {color:rgb(255,0,0);}
* określenie składowych koloru w modelu RGB w sposób względny
* DIV {color:rgb(100%,0%,0%);}
* określenie składowych koloru w modelu RGB w postaci szesnastkowej
* DIV {color:#FF0000;}
* nazwa systemowa koloru
* DIV {color:ActiveCaption;}

[idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Dziedziczenie w CSS

Z drzewem dokumentu związana jest własność dziedziczności stylów. Polega ona na tym, że elementy leżące niżej w hierarchii (potomkowie), jeśli nie zaznaczymy inaczej, dziedziczą (przejmują) atrybuty formatowania, które zostały nadane ich przodkom. [idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Kaskadowość w CSS

Kaskadowość odpowiada za określenie hierarchii stosowanych stylów w dokumencie. Wiadomo, że style można wstawiać do dokumentu na kilka sposobów (bezpośrednio w kodzie strony jako atrybut dowolnego znacznika, w nagłówku <head></head>, globalnie dla danego dokumentu oraz przez dołączenie zewnętrznego arkusza). Mieszanie zastosowanych stylów jest jak oczywiście możliwe i dość często spotykane, dlatego konieczne jest określenie ważności poszczególnych metod.  
Zasada kaskadowości przyjęta przez twórców wygląda następująco: \_LIST(,( najpierw ładowane i uwzględniane są zewnętrzne arkusze, następnie style wpisane do nagłówka <head></head>, a na samym końcu style wpisane bezpośrednio do znacznika )) Takie rozwiązanie umożliwia pełną kontrolę nad dokumentem, a w przypadku sprzeczności zdefiniowanych stylów użyty zostanie ten, który jest najbliżej formatowanego dokumentu. [idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Kolejność aplikowania stylów

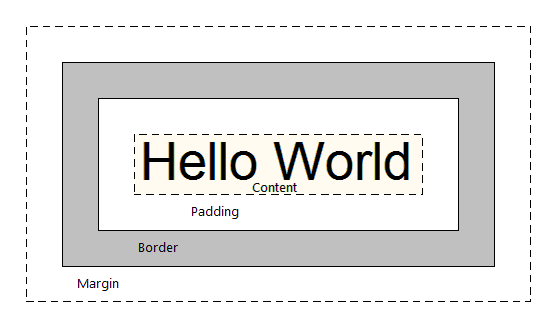
* reguła !important - ważne mają pierwszeństwo
* precyzja selektora - im bardziej precyzyjny selektor tym jest on ważniejszy
* kolejność dołączania arkuszy - później dołączone arkusze są ważniejsze
* kolejność reguł - reguły późniejsze nadpisują wcześniejsze

[idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Model blokowy

Kaskadowe arkusze stylów do formatowania wszystkich elementów wykorzystują model blokowy. W praktyce polega to na tym, że wszystkie elementy umieszczane są w dodatkowej przestrzeni o kształcie prostokątów. W ramach każdego bloku można kontrolować następujące właściwości:

* Marginesy (ang. margins) — odstępy dzielące poszczególne elementy — bloki.
* Obramowanie (ang. border) — linie rozdzielające marginesy i dopełnienie.
* Dopełnienia (ang. padding) — odstępy pomiędzy elementem a otaczającą go ramką - nazywane również marginesami wewnętrznymi.
* Zawartość (ang. content) — właściwa zawartość.

  
[idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Połączenie CSS i HTML - styl lokalny

Style wpisane odnoszą się tylko do danego obiektu objętego tymi znacznikami.

1﻿<html>

2 <head>

3 </head>

4 <body>

5 <h1>Nagłówek</h1>

6 <p style="color:green; font-style:italic">

7 to jest paragraf pisany pochyłą czcionką w kolorze zielonym.

8 </p>

9 <p style="color:red; font-weight:bold">

10 to jest paragraf pisany pogrubioną czcionką w kolorze czerwonym.

11 </p>

12 </body>

13</html>

[[pokaż zawartość pliku]](http://www.cs.put.poznan.pl/jkobusinski/css/cssIntegration01.txt) [[pobierz plik]](http://www.cs.put.poznan.pl/jkobusinski/css/cssIntegration01.html) [idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Połączenie CSS i HTML - styl wewnętrzny

Stylami wewnętrznymi nazywamy arkusz stylów, który jest osadzony w sekcji <HEAD> dokumentu HTML. Dzięki deklaracji type="text/css" definiujemy osadzony arkusz stylów. Dalej nie wymagane, jednak dość pomocnicze znaczniki <!-- -->. Dzięki nim arkusz stylów jest usuwany przed przeglądarkami, które nie potrafią z nich korzystać.

1﻿<html>

2 <head>

3 <style rel="stylesheet" type="text/css" >

4<!--

5 p {

6 color:green;

7 font-style:italic;

8 }

9-->

10 </style>

11 </head>

12 <body>

13 <h1>Nagłówek</h1>

14 <p>

15 to jest paragraf pisany pochyłą czcionką w kolorze zielonym.

16 </p>

17 <p>

18 to też jest paragraf pisany pochyłą czcionką w kolorze zielonym.

19 </p>

20 </body>

21</html>

[[pokaż zawartość pliku]](http://www.cs.put.poznan.pl/jkobusinski/css/cssIntegration02.txt) [[pobierz plik]](http://www.cs.put.poznan.pl/jkobusinski/css/cssIntegration02.html) [idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Połączenie CSS i HTML - styl zewnętrzny

Style zewnętrzne umieszczone są poza dokumentem HTML w odrębnym pliku \*.css. Plik ten zawiera po prostu definicję stylów. W pliku HTML wystarczy wskazać, skąd mają być odczytywane style za pomocą znacznika <link>.  
**Plik HTML:**

1﻿<html>

2 <head>

3 <link rel="stylesheet" type="text/css" href="cssIntegration03b.css">

4 </head>

5 <body>

6 <h1>Nagłówek</h1>

7 <p>

8 to jest paragraf pisany pochyłą czcionką w kolorze zielonym.

9 </p>

10 <p>

11 to też jest paragraf pisany pochyłą czcionką w kolorze zielonym.

12 </p>

13 </body>

14</html>

[[pokaż zawartość pliku]](http://www.cs.put.poznan.pl/jkobusinski/css/cssIntegration03a.txt) [[pobierz plik]](http://www.cs.put.poznan.pl/jkobusinski/css/cssIntegration03a.html)  
**Zewnętrzny plik CSS:**

1 p {

2 color:green;

3 font-style:italic;

4 }

[[pokaż zawartość pliku]](http://www.cs.put.poznan.pl/jkobusinski/css/cssIntegration03b.txt) [[pobierz plik]](http://www.cs.put.poznan.pl/jkobusinski/css/cssIntegration03b.css) [idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Selektory proste

W przypadku selektorów prostych selektorem jest znacznik języka HTML.  
**Przykład:**

TABLE {color:blue;}

P {margin-left:20px;}

[idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Selektor uniwersalny

Zdarza się, że zachodzi potrzeba zdefiniowania jakiegoś typu formatowania w ten sposób, aby zadziałał na cały dokument. Do tego celu służy selektor uniwersalny (ogólny) "\*".  
**Przykład:**

\* {background-color:Yellow;}

[idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Selektory potomka

Selektor tego typu pozwala nadać atrybuty elementom, które leżą niżej w hierarchii drzewa dokumentu (zawierają się w innych zewnętrznych znacznikach). Dzięki temu możemy zmienić typ formatowania tylko dla określonych elementów, które są podrzędne w stosunku do innych (przodków).  
Potomek nie musi leżeć bezpośrednio wewnątrz znacznika przodka. Może być zawarty jeszcze w innych znacznikach, które z kolei zawierają się w rodzicu. Nie jest wtedy konieczne podawanie w deklaracji wszystkich rodziców, a jedynie przodka i potomka.  
**Przykład:**

P SPAN {font-style:italic;}

TABLE A {color:Red;}

[idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Selektory dziecka

Selektor tego typu pozwala nadać atrybuty elementom, które leżą o jeden rząd niżej w hierarchii drzewa dokumentu (zawierają się w innym zewnętrznym znaczniku). W odróżnieniu do poprzedniego przypadku, tutaj znacznik będący dzieckiem, musi znajdować się bezpośrednio wewnątrz znacznika rodzica.  
**Przykład:**

P > SPAN {font-style:italic;}

H > P {color:Red;}

[idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Selektory rodzeństwa

Selektor ten umożliwia nadanie określonych atrybutów jednemu z sąsiadujących ze sobą braci - temu, który w deklaracji został podany jako drugi. Zwykły tekst znajdujący się pomiędzy braćmi nie ma wpływu na działanie selektora, tzn. jest ignorowany.  
**Przykład:**

I + B {color:blue;}

[idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Selektory atrybutu

Polecenie to może zostać wykorzystane do wskazywania elementów, które posiadają określony atrybut. Pod uwagę może (ale nie musi) być też brana wartość tego atrybutu.  
**Przykład:**  
Każdy akapit, któremu został nadany atrybut title powinien mieć kolor niebieski

P[title] {color:blue;}

Każdy akapit, któremu został nadany atrybut title o wartości "test", powinien mieć kolor niebieski

P[title="test"] {color:blue;}

Każdy akapit, któremu został nadany atrybut title w którego wartości występuje słowo "test", powinien mieć kolor niebieski

P[title=~"test"] {color:blue;}

Każdy akapit, któremu został nadany atrybut title w którego wartości występuje słowo "test", powinien mieć kolor niebieski [idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Identyfikatory

Element może mieć nadany unikalny identyfikator. Tylko jeden element w dokumencie może mieć dany identyfikator. Selektor zapisuje się jako hash (#) i nazwę identyfikatora połączone z innym selektorem:  
**Przykład:**  
Akapit o identyfikatorze "test" powinien mieć kolor niebieski.

P#test {color:blue;}

[idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Klasy

Elementy mogą mieć jedną lub więcej klas. Klas używa się do określenia szczegółowych cech elementów lub określenia przynależności wielu różnych elementów do danej klasy (grupy, rodziny). Selektor klasy jest nazwą klasy połączoną z innym selektorem za pomocą kropki.  
**Przykład:**  
Wszystkie akapity klasy "test" powinny mieć kolor niebieski.

P.test {color:blue;}

Klas używa się do nadania stylów elementom, które są „rozsypane” po dokumencie tak, że nie sposób im nadać styl w inny sposób. [idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Pseudoklasy

Poza klasą nadaną przez autora, element może mieć pseudoklasę nadaną przez przeglądarkę. Selektor pseudoklasy jest nazwą połączoną z innym selektorem za pomocą dwukropka (nie może być spacji wokół dwukropka).  
Dostępne pseudoklasy:

link

Odnośniki, które nie zostały jeszcze odwiedzone.

visited

Odnośniki, które zostały odwiedzone.

hover

Pseudoklasa nadawana elementom, gdy znajduje się nad nimi wskaźnik myszy.

focus

Elementy, które mają fokus, np. element formularza, który użytkownik właśnie wypełnia.Bug w IE

active

Elementy, które są aktywne, np. odnośniki podczas klikania na nie myszą.

first-child

Element, który jest pierwszym dzieckiem swojego rodzica — np. pierwszy element listy.

**Przykład:**  
Wszystkie odnośniki, które zostały odwiedzone powinny mieć kolor zielony.

A:visited {color:green;}

[idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Pseudoelementy

Pseudoelementy są to fragmenty elementów dokumentu. Składnia tego selektora jest identyczna jak pseudoklas.

first-line

Pierwsza linia tekstu zawieranego przez element.

first-letter

Pierwsza litera tekstu zawieranego przez element.

**Przykład:**  
Wszystkie pierwsze litery paragrafów powinny mieć rozmiar dwa razy większy niż tekst.

P:first-letter {font-size:200%;}

[idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Grupowanie selektorów

Jedna reguła może mieć wiele selektorów. Kolejne selektory oddziela się przecinkiem.  
**Przykład:**  
Nagłówki pierwszego, drugiego i trzeciego poziomu powinny mieć kolor zielony.

H1, H2, H3 {color:green;}

[idź do góry](http://www.cs.put.poznan.pl/jkobusinski/css.html#topPage)

# Zadania

Przygotuj dokument HTML, w którym elementy wyznaczone przez poniższe reguły będą prezentowane w kolorze czerwonym.

1. 1p[class][dir="ltr"][title~="akapit"][lang|="pl"]
2. 1element.klasa1.klasa2#id:hover:first-letter
3. 1test.user #main > :last-child a:visited:hover

Zdefiniuj styl, który wskazuje na elementy zawierające tekst "wybierz mnie!":

1. 1<foo>wybierz mnie!</foo>
2. 2<bar>nie mnie ale <foo>wybierz mnie!</foo></bar>
3. 1<bar>nie mnie <baz><foo>mnie nie bierz!</foo></baz></bar>
4. 2<bar>nie mnie ale <foo>wybierz mnie!</foo></bar>
5. 1 <foo class="bar">nie mnie</foo>
6. 2 <foo class="quz">nie mnie<foo class="bar">wybierz mnie!</foo></foo>
7. 3 <foo class="quz">nie mnie!</foo>