



## PODSTAWOWE POJĘCIA

## OSADZANIE STYLÓW W DOKUMENCIE

## Styl wewnętrzny

## Styl lokalny

```
<p style="color:green;font-size:15pt;
">Treść dokumentu</p>
```

## Styl zagnieźdżony

```
<html>
<head>
<style type="text/css">
P
{
color:green;
font-size:15pt;
}
</style>
</head>
<body>
<p>Tekst akapitu</p>
</body>
</html>
```

## Styl zewnętrzny

## Zewnętrzny arkusz stylów

Umieszczany jest w nagłówku <head> dokumentu. Styl zewnętrzny obejmuje działaniem całą zawartość strony i może być wykorzystany również w podstronach.

```
<html>
<head>
<link href="arkusz.css" rel="stylesheet"
type="text/css" />
</head>
<body>
<p>Tekst akapitu</p>
</body>
</html>
```

W przypadku języka XHTML polecenie odpowiedziałne za podłączenie zewnętrznego arkusza stylów ma postać:

```
<?xml-stylesheet type="text/css"
href="arkusz.css" ?>
Dla zachowania zgodności dokumentów XHTML zaleca się
umieszczanie podwójnej deklaracji wywołującej arkusz stylów.
Oczywiście oba wpisy muszą znaleźć się w nagłówku kodu
witryny.
```

```
<link rel="stylesheet" type="text/css"
href="arkusz.css" />
<?xml-stylesheet type="text/css"
href="arkusz.css" ?>
```

## Styl importowany

Jest pobierany z oddzielnego pliku znajdującego się pod wskazanym adresem. Podobnie jak styl zewnętrzny, działaniem obejmuje całą treść strony, na której został użyty.

```
<html>
<head>
<style type="text/css">
@import url("arkusz.css");
P
{
color:green;
}
</style>
</head>
<body>
<p>Tekst akapitu</p>
</body>
</html>
```

## Budowa zewnętrznego arkusza stylów

Zewnętrzny arkusz jest zwykłym plikiem tekstowym, a jego zawartość może wyglądać tak jak na poniższym przykładzie. Ważne jest tylko to, aby plik miał rozszerzenie .css, np. arkusz.css.

```
(* To jest przykład zewnętrznego arkusza stylów *)
P
{
color:green;
font-size:15pt;
}
```

## SELEKTORY

## GRUPOWANIE SELEKTORÓW

Wyobraź sobie, że w Twoim dokumencie tekst ma mieć kolor czarny, a wszystkie użyte nagłówki powinny być czerwone. Odpowiedni arkusz stylów może mieć postać:

```
P
{
color:black;
}
H1
{
color:red;
}
H2
{
color:red;
}
H3
{
color:red;
}
Powyższy kod możemy skrócić, grupując selektory. Uproszczona
wersja może mieć postać:
```

```
P
{
color:black;
}
H1, H2, H3
{
color:red;
}
```

## SELEKTOR PROSTY

Selektorem prostym jest znacznik języka HTML przypisany do elementu, który chcemy w danym momencie formatować. Styl zdefiniowany dla przykładowego selektora P będzie się odnosił wyłącznie do akapitów zamkniętych w znaczniku <p></p>.

```
P
{
color:red;
font-size:14pt;
}
```

## SELEKTOR UNIWERSALNY

Selektor uniwersalny pozwala na ustalenie formatowania dla wszystkich elementów strony WWW. Obecnie dostępne są dwa selektory uniwersalne: BODY oraz \*.

```
*
{
color:red;
}
BODY
{
color:red;
}
```

## SELEKTOR POTOMKA

Selektor potomka pozwala na ustawienie formatowania dla elementu występującego we wnętrzu innego znacznika. Przykład wymusza nadanie formatowania fragmentowi tekstu zamkniętego w <span></span> i znajdującego się w znaczniku <h1></h1>.

## H1 SPAN

```
{
font-size:10pt;
}
```

## SELEKTOR DZIECKA

Selektor dziecka jest wynikiem zależności panujących pomiędzy poszczególnymi znacznikami języka HTML. Przykładowy styl zadziała, gdy wewnątrz bloku tekstu zostanie umieszczony znacznik <span></span>.

```
P > SPAN
{
font-style:italic;
text-decoration:underline;
}
```

## SELEKTOR RODZIEŃSTWA

Na podstawie tego rodzaju selektora możemy przypisać styl elementowi bezpośrednio sąsiadującemu z innymi. By warunek został spełniony, oba selektory muszą mieć wspólnego rodzica.

```
P + SPAN
{
font-style:italic;
text-decoration:underline;
}
```

## SELEKTOR ATRYBUTU

Selektor atrybutu odnosi się do wybranego znacznika HTML mającego ściśle określony atrybut. Przykładowy styl zadziała w przypadku nagłówka stopnia pierwszego o określonym atrybucie title.

```
H1[title]
{
color:red;
}
```

## SELEKTOR ATRYBUTU Z MOŻLIWOŚCIĄ DOPASOWANIA CIĄGU ZNAKÓW

Selektor atrybutu z możliwością dopasowania ciągu znaków daje możliwość odwołania się do wybranego elementu strony na podstawie fragmentu ciągu znaków występujących jako atrybut. Dostępne są trzy sposoby odwołania się do ciągu znaków. Pierwszy z selektorów wykorzystujących ciągi znaków pozwala na podpięcie stylu tekstu, od którego zaczyna się dany ciąg znaków. Zwróć uwagę na znak \$ w zapisie selektora. Poniższy przykład dotyczy wszystkich znaczników H1, w których występuje atrybut title zaczynający się od znaków pie.

```
H1[title^="pie"]
{
color:red;
}
```

Drugi z selektorów wykorzystujących ciągi znaków pozwala na podpięcie stylu tekstu, którym kończy się dany ciąg znaków. Zwróć uwagę na znak \$ w zapisie selektora. Poniższy przykład dotyczy wszystkich znaczników H1, w których występuje atrybut title kończący się na ony.

```
H1[title$="ony"]
{
text-decoration:underline;
}
```

Trzeci z selektorów wykorzystujących ciągi znaków pozwala na podpięcie stylu tekstu, który występuje w dowolnym miejscu ciągu znaków. Zwróć uwagę na znak \* w zapisie selektora. Poniższy przykład dotyczy wszystkich znaczników H1, w których występuje atrybut title zawierający ciąg znaków nagłówek.

```
H1[title="*nagłówek"]
{
font-style:italic;
}
```

## SELEKTOR OGÓLNEGO RODZIEŃSTWA

Selektor ogólnego rodzeństwa składa się z dwóch prostych selektorów rozdzielonych znakiem ~ (tyldy) i pasuje do tych elementów selektora drugiego (w poniższym przykładzie jest to P), które są poprzedzone elementami pierwszego selektora (w poniższym przykładzie jest to H1). Oba elementy muszą mieć wspólnego rodzica (w naszym przypadku jest to DIV) i dodatkowo drugi element nie musi być bezpośrednio poprzedzony pierwszym elementem.

```
<div>
<h1></h1>
<p></p>
<h1></h1>
<p></p>
<h1></h1>
<p></p>
</div>
```

Przykład selektora ogólnego rodzeństwa wygląda następująco:

```
H1 ~ P
{
color:red;
font-weight:bolder;
}
```

## GRUPOWANIE SELEKTORÓW

Selektory można dowolnie grupować, jeżeli pewne wartości mają obowiązywać dla każdego z nich. Dzięki zbiorczemu zapisowi możemy znacznie uprościć strukturę arkusza i pracę z kodem witryny. Poszczególne selektory wchodzące w skład grupy rozdzielamy przecinkami. Poniżej przedstawiono przykład zgrupowanych selektorów H1, H2 oraz P.

```
H1, H2, P
{
color:green;
}
```

## IDENTYFIKATORY

Identyfikator pozwala na przypisanie formatowania do wybranego znacznika mającego unikalny atrybut id. Dzięki temu możemy różnicować sposób prezentacji elementów na stronie.

```
#pochyly
{
font-style:italic;
font-size:20pt;
}
```

Aby skorzystać z przykładowego stylu, wybrany znacznik musi mieć dodatkowy atrybut id.

```
<p id="pochyly">Treść akapitu</p>
```

Możliwy jest również inny zapis, który wymusza przypisanie identyfikatora wyłącznie do określonego znacznika.

```
H1#pochyly
{
font-style:italic;
font-size:20pt;
}
```

## KLASY

Klasa umożliwiła zróżnicowanie formatowania wybranych elementów na stronie w zależności od atrybutu class umieszczonego w znaczniku HTML. W odróżnieniu od identyfikatora klasa może być wykorzystywana wielokrotnie.

```
.moja
{
color:green;
font-size:14pt;
}
```

Aby skorzystać z przykładowej klasy, znacznik musi mieć zdefiniowany atrybut class z nazwą klasy.

```
<p class="moja">Treść akapitu</p>
```

Klasa może zostać również powiązana z konkretnym znacznikiem HTML.

```
P.moja
{
color:green;
font-size:14pt;
}
```

## PSEUDOKLASY

Specyfikacja CSS przewiduje kilka ściśle określonych klas pozwalających na formatowanie niektórych elementów dokumentu. Do najpopularniejszych pseudoklas z całą pewnością zaliczają się pseudoklasy odsyłaczy.

## Pseudoklasa :link

Dotyczy formatowania odnośnika w stanie nienaruszonym.

```
A:link
{
color:navy;
}
```

## Pseudoklasa :visited

Dotyczy formatowania odnośnika, który został już odwiedzony. Pozwala na wyróżnienie go spośród innych odnośników znajdujących się na stronie.

```
A:visited
{
color:green;
}
```

## Pseudoklasa :hover

Pozwala na ustalenie formatowania elementu, nad którym w danej chwili znajduje się kursor myszy.

```
A:hover
{
color:red;
text-decoration:underline;
}
```

## Pseudoklasa :active

Odpowiada za ustalenie wyglądu aktywnego elementu na stronie.

```
A:active
{
color:silver;
}
```

## Pseudoklasa :focus

Pozwala na wyróżnienie aktualnie wyświetlanego elementu, np. odsyłacza.

```
A:focus
{
color:braun;
}
```

## Pseudoklasa :lang

Jest wykorzystywana do określania formatowania dokumentów wielojęzycznych.

```
P:lang(pl)
{
font-weight:bold;
color:black;
}
```

Aby skorzystać z przykładowej pseudoklasy :lang zdefiniowanej powyżej, kod HTML musi wyglądać następująco:

```
<p lang="pl">Tekst w języku polskim.</p>
```

## Pseudoklasa :target

Stosując pseudoklasę :target, możemy wymusić formatowanie elementów będących kotwicami na stronie. Formatowanie będzie miało miejsce tylko wtedy, gdy odnośnik prowadzący do kotwicy zostanie wybrany, a adres będzie miał postać: <http://adres/pl/plik.html#kotwica>. Odpowiedni arkusz stylów może mieć następującą postać:

```
*:target
{
text-decoration:none;
color:green;
}
```

## Pseudoklasa :enabled i :disabled

Stosując pseudoklasy :enabled oraz :disabled, możemy kontrolować wygląd elementów interfejsu użytkownika (formularzy), w zależności od tego, czy są one dostępne, czy też nie.

```
input[type="text"]:enabled
{
background:#C0FFC0;
}
input[type="text"]:disabled
{
background:#FFC0C0;
}
```

## Pseudoklasa :checked

Dzięki pseudoklasie :checked możemy kontrolować wygląd aktywnych pól formularza typu radio lub checkbox.

```
input:checked
{
border:1px solid red;
color:green;
width:25px;
height:25px;
background-color:red;
}
```

## Pseudoklasa :root

Pseudoklasa :root należy do grupy pseudoklas strukturalnych, za pomocą których możemy odwołać się do struktury dokumentu XHTML/HTML; ta struktura jest poza zasięgiem klasycznych selektorów. Omawiana pseudoklasa odnosi się do korzenia struktury, którym w przypadku dokumentów XHTML/HTML jest element HTML, i zawsze działa w całym dokumencie.

```
:root
{
background-color:#C0FFC0;
color:navy;
}
```

## Pseudoklasa :nth-child()

Dzięki pseudoklasie :nth-child() możemy odwołać się do elementu, który ma przed sobą określoną w nawiasie liczbę rodzeństwa w strukturze dokumentu. Wartość umieszczona w nawiasie może być numerem, słowem kluczowym lub formułą.

```
p:nth-child(3)
{
color:red;
}
UL LI:nth-child(even)
{
color:red;
font-style:italic;
}
OL LI:nth-child(odd)
{
color:green;
font-weight:bolder;
}
LI:nth-child(2n+5)
{
color:red;
font-style:italic;
font-weight:bolder;
}
```

*Dalsza część książki dostępna w wersji  
pełnej.*

