Tworzenie Stron Internetowych

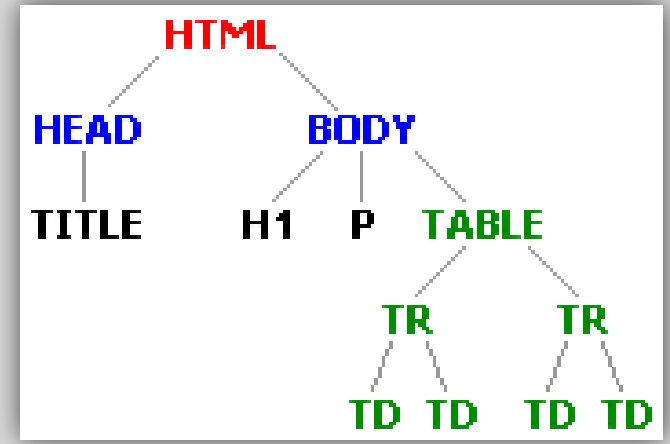
odcinek 7

CSS – dziedziczenie

Drzewo dokumentu

Drzewo dokumentu to hierarchia elementów umieszczonych w dokumencie źródłowym HTML. Każdy element w takim drzewie ma dokładnie jednego rodzica, z wyjątkiem elementu podstawowego (korzenia drzewa, root). Wzajemne relacje elementów w drzewie są następujące:

- **Dziecko (child)** – element położony o jeden szczebel niżej względem danego elementu
- **Potomek (descendant)** – element położony o jeden lub więcej szczebli niżej
- **Rodzic (parent)** – element położony o jeden szczebel wyżej
- **Przodek (ancestor)** – element położony o jeden lub więcej szczebli wyżej
- **Brat (sibling)** – element mający tego samego rodzica co inny element; jeśli znajduje się obok niego, to jest to **brat przylegający** (**adjacent sibling**).



Elementy leżące niżej w hierarchii drzewa dokumentu, zawierają się wewnątrz znaczników elementów nadrzędnych, np. znaczniki **<body>** oraz **</body>** muszą być umiejscowione pomiędzy znacznikami **<html>** i **</html>**, które są nadrzędne dla wszystkich innych (root).

CSS – dziedziczenie

Dziedziczenie stylów

Z drzewem dokumentu związana jest własność dziedziczności stylów – elementy leżące niżej w hierarchii (**potomkowie**), jeśli nie zaznaczymy inaczej, **dziedziczą styl nadany ich przodkom**. Jednak nie wszystkie cechy dziedziczone są automatycznie.

Uwaga: *należy sprawdzać działanie w praktyce tej własności ponieważ w niektórych przeglądarkach internetowych zdarza się błędna interpretacja dziedziczenia stylów.*

Dla przykładu po umieszczeniu w CSS reguły

```
p {
   color: red;
   border:1px solid black;
   }
```

a w HTML:

```
<p> treść akapitu z fragmentem <em> pogrubionym </em></p>
```

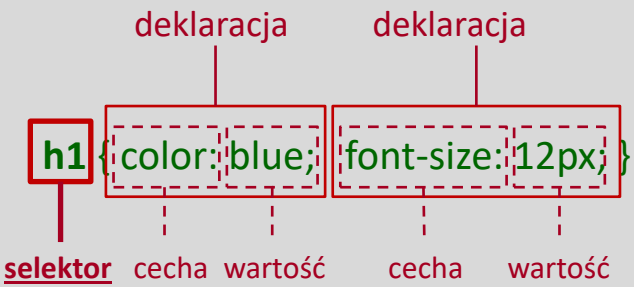
otrzymamy akapit, w którym fragment (potomek) dziedziczy kolor po akapicie (przodek). Natomiast obramowanie nie jest dziedziczone.

CSS – selektory

Język CSS składa się z reguł (przypomnienie)

Każda reguła składa się ze zbioru **cech**, którym nadaje **wartości** (*np. kolor czcionki ma być czerwony, wielkość pisma ma być duża*) oraz określenia (tzw. **selektora**) wskazującego, do których elementów w dokumencie HTML ma ta reguła być zastosowana (*np. zastosuj to tylko do pierwszego akapitu*).
Każda para cecha-wartość to tzw. deklaracja. Przy jednym selektorze można umieścić wiele deklaracji oddzielając je średnikiem.

selektor {cecha: wartość; inna-cecha: jakieś_wartości;}



Selektory umożliwiają precyzyjne wskazywanie elementów w kodzie źródłowym dokument HTML na podstawie ich nazwy, atrybutu, wartości atrybutu lub pozycji w drzewie dokumentu.

Selektory dzielą się na pięć kategorii:

- selektory proste
- kombinatory
- atrybutowe
- pseudoklas
- pseudoelementów

CSS – selektory proste

Selektor elementu (prosty)

Selektor elementu to podstawowy rodzaj selektora. Pozwala wskazać elementy HTML po ich nazwie i następnie nadać im odpowiedni styl.

```
selektor { cecha: wartość }
```

np.:

```
p { text-align: justify }
```

pozwala uzyskać wszystkie akapity wyjustowane.

Selektor uniwersalny

Selektor uniwersalny pozwala ustalić dany styl dla wszystkich elementów w dokumencie.

```
* { cecha: wartość }
```

Niestety w starszych przeglądarkach jest problem z poprawną interpretacją tego polecenia. Dlatego bezpieczniej jest używać selektora dla elementu body.

```
body { cecha: wartość }
```

np.:

```
body { text-align: center }
```

CSS – selektory proste

Grupowanie selektorów

Grupowanie selektorów pozwala nadać ten sam styl **kilku różnym elementom jednocześnie** (bez względu na ich położenie w hierarchii drzewa dokumentu).

```
selektor1, selektor2, selektor3... { cecha: wartość }
```

np.:

```
ul, ol {
        text-decoration: underline;
        font-family:  Arial, Verdana, Sans-serif
        }
```

dodaje podkreślenie tekstu we wszystkich listach ul i ol i określa krój (rodzinę) czcionki .

CSS – selektory proste

Identyfikator: id

Atrybut **id** nadaje unikalny identyfikator elementowi. Przeznaczenie identyfikatora jest różne, np: nadawanie stylów CSS. Identyfikator jest unikalny, znaczy to, że w danym dokumencie HTML może występować tylko jeden element o danym identyfikatorze. Identyfikator musi zaczynać się od litery, po niej mogą się znajdować: cyfry lub inne litery, myślniki, podkreślniki, dwukropki oraz kropki.

```
<h2 id="rozdzial-2">Rozdział drugi</h2>
```

Klasa: class

Atrybut **class** pozwala dowolną liczbę elementów w dokumencie HTML "zgrupować" razem poprzez nadanie im takiej samej klasy. Elementy mogą mieć nawet kilka klas jednocześnie (kolejność nie ma znaczenia), a poszczególne klasy oddziela się spacją. Nazwa klasy nie może zawierać spacji. Klasy przede wszystkim przydatne są nadawania stylów za pomocą CSS. Dobra nazwa klasy opisuje rolę, jaką pełni element w dokumencie (czym jest, a nie jak wygląda). Wymagania odnośnie nazwy są takie same jak dla identyfikatora.

```
<p class="info">terść akapitu</p>
```

Identyfikator a klasa

Identyfikator stosuje się dla elementów, które występują tylko raz w dokumencie i nie ma sensu, żeby było ich więcej. Klasy stosuje się do elementów, które mogą się powtarzać.

CSS – selektory proste

Selektor identyfikatora

Selektor identyfikatora pozwala nadać styl elementowi, który ma konkretny, unikalny identyfikator (**id**), czyli występuje tylko raz dokumencie. Wykorzystuje się go do nadawania stylu np.: obszarom w układzie strony internetowej (nagłówek, menu nawigacyjne, stopka, ...).

```
selektor#identyfikator { cecha: wartość }
#identyfikator { cecha: wartość }
```

krótszy zapis

(*selektor* to dowolny selektor elementu)

Przykład:
nadanie identyfikatora w HTML:

```
<div id="stopka">zawartość stopki</div>
```

nadanie stylu w CSS:

```
#stopka { color: red }
```

Reguła

```
*#info { color: red }
```

nada styl wszystkim elementom z ***id="info"*** (umieszczonych w kilku różnych dokumentach) niezależnie od ich typu (wykorzystujemy tu selektor uniwersalny). W składni można opuścić *, czyli wracamy do zwykłej składni.

CSS – selektory proste

Selektor klasy

Selektor ten pozwala nadać jednolity styl grupie elementów, którym nadano tę samą klasę (**class**). Selektor klasy jest przydatny w przypadku, gdy na wielu stronach serwisu znajdują się elementy, które powinny mieć taki sam styl, a dodatkowo nie można dla nich posłużyć się selektorem typu, ponieważ na stronie znajdują się inne elementy tego samego typu, co wybrany element, ale nie chcemy, aby i one otrzymały ten sam styl.

```
selektor.klasa { cecha: wartość }
```
dłuższy zapis / tylko wybrany(e) element(y) z danej klasy

```
.klasa { cecha: wartość }
```
krótszy zapis / wszystkie elementy z danej klasy

(*selektor* to dowolny selektor elementu)

Przykład:
nadanie klasy w HTML:

```
<h1 class="uwaga">tytuł</h1>
<p class="uwaga">treść akapitu</p>
```

nadanie stylu w CSS:

```
p.uwaga, h1.uwaga { color: red }
```

lub krócej

```
.uwaga { color: red }
```

CSS – selektory proste

Selektor klasy – lista klas

Do pojedynczego elementu można przypisać kilka różnych klas. Taki element otrzyma styl przypisany do wszystkich tych klas.

```
<element class="klasa1 klasa2 klasa3...">...</element>
```

(*element* to dowolny element HTML, kolejność wpisywania *klasa1 klasa2 klasa3...* nie ma znaczenia)

Natomiast reguła

```
*.info { color: red }
```

nada podany styl wszystkim elementom z *class="info"* niezależnie od ich typu (wykorzystujemy tu selektor uniwersalny). W składni można opuścić ***, czyli wracamy do zwykłej składni.

CSS – selektory proste

Selektor klasy – podzbiory klas

Istnieje możliwość określenia tzw. podzbioru klas, czyli podania, jakie klasy musi mieć jednocześnie przypisane element, aby wybrana reguła CSS miała do niego zastosowanie. Na przykład deklaracja:

```
selektor.klasa1.klasa3 { cecha: wartość }
```

nadaje styl elementowi

```
<selektor class="klasa1 klasa2 klasa3">...</selektor>
```

ale nie elementowi

```
<selektor class="klasa1 klasa2">...</selektor>
```

(*selektor* to dowolny element HTML, kolejność wpisywania *klasa1 klasa2 klasa3...* nie ma znaczenia)

CSS – selektory-kombinatory

Selektor potomka

Selektor potomka pozwala nadać styl **elementom**, które leżą **niżej w hierarchii** drzewa dokumentu. Dzięki temu możemy odnieść się tylko do elementów, które są podrzędne w stosunku do innych (przodków). Potomek nie musi leżeć bezpośrednio wewnątrz znacznika przodka i nie jest wtedy konieczne podawanie w deklaracji wszystkich pośrednich rodziców, a jedynie przodka i potomka.

```
przodek potomek { cecha: wartość }
```

(wyrazy *przodek* oraz *potomek* są selektorami elementu, oddziela je spacja)
np.:

```
div h1 { font-size:10vw }
```

pozwala uzyskać rozmiar czcionki 10vw (viewport width*) tylko dla elementów <h1> położonych w bloku .

* 1vw to 1% szerokości okna przeglądarki. Taka jednostka pozwala na skalowanie rozmiarów zależnie od rozmiaru okna przeglądarki.

CSS – selektory-kombinatory

Selektor dziecka

Selektor dziecka pozwala nadać styl **elementom**, które leżą o **jeden rząd niżej** w hierarchii drzewa dokumentu. W odróżnieniu od selektora potomka, tutaj element będący dzieckiem, musi znajdować się bezpośrednio wewnątrz elementu rodzica.

```
rodzic > dziecko { cecha: wartość }
```

(wyrazy *rodzic* i *dziecko* są selektorami elementu)
np.:

```
p > strong { font-weight: bold }
```

pozwala uzyskać pogrubienie czcionki tylko dla elementów będących dziećmi elementu <p>.

CSS – selektory-kombinatory

Selektor braci przylegających

Selektor braci przylegających umożliwia nadanie określonego stylu **jednemu z sąsiadujących bezpośrednio ze sobą braci** – temu, który w deklaracji został podany jako drugi.

```
brat1 + brat2 { cecha: wartość }
```

(wyrazy *brat1* i *brat2* są selektorami elementu)
np.:

```
div + p { background-color: yellow }
```

zadziała tylko, jeśli wewnątrz tego samego elementu (rodzica), będą znajdowały się bezpośrednio obok siebie elementy oraz <p>, wtedy to ten drugi uzyska podany styl.

CSS – selektor-kombinatory

Ogólny selektor braci

Selektor ten umożliwia nadanie określonego stylu **wszystkim braciom danego elementu**. Deklaracja działa na wszystkie pojawienia się elementu, który w deklaracji został podany jako drugi. Inne elementy znajdujące się pomiędzy braćmi nie mają wpływu na działanie selektora.

```
brat1 ~ brat2 { cecha: wartość }
```

(wyrazy *brat1* i *brat2* są selektorami elementu)
np.:

```
div ~ p { font-style: italic }
```

zadziała tylko, jeśli wewnątrz tego samego elementu (rodzica), będą znajdowały się elementy oraz <p>, wtedy to ten drugi uzyska podany styl. Deklaracja zadziała na wszystkie <p>, niezależnie od innych elementów znajdujących się również w tym samym rodzicu.

CSS – selektory atrybutowe

Prosty selektor atrybutu

Selektor ten pozwala nadać styl elementom, którym w kodzie HTML został nadany **określony atrybut** (jego wartość nie ma znaczenia).

```
selektor[atrybut] { cecha: wartość }
```

(*selektorem* jest dowolny element HTML)

Dozwolone jest podane kilku atrybutów. Wtedy element musi posiadać wszystkie z nich, aby otrzymał zadany styl.

Podając, np.:

```
p[title] { font-variant: small-caps }
```

otrzymamy kapitaliki tylko w akapitach, którym został nadany atrybut **title**.

CSS – selektory atrybutowe

Selektor atrybutu o określonej wartości

Selektor ten pozwala nadać styl elementom, którym w kodzie HTML został nadany **określony atrybut z określoną wartością**.

selektor[atrybut="*wartość atrybutu*"] { cecha: wartość }

(*selektorem* jest dowolny element HTML)

Dozwolone jest podane kilku atrybutów. Element otrzyma formatowanie tylko wtedy, gdy posiada wszystkie z nich i każdy musi mieć przypisaną podaną wartość.

Podając, np.:

p[title="akapit"] { color: yellow }

w kolorze żółtym otrzymamy tylko te akapity, którym został nadany atrybut **title** o wartości **"akapit”**.

CSS – selektory atrybutowe

Selektor atrybutu zawierającego określony wyraz

Selektor ten pozwala nadać styl elementom, którym w kodzie HTML został nadany **określony atrybut z pewną wartością, w której występuje określony wyraz** (oprócz niego mogą występować również inne wyrazy). Wyraz ten nie może zawierać spacji, ale od innych wyrazów musi być oddzielony spacjami.

```
selektor[atrybut~="wyraz"] { cecha: wartość }
```

(*selektorem* jest dowolny element HTML)

Dozwolone jest podane kilku atrybutów lub/i wyrazów. Element otrzyma formatowanie tylko jeśli posiada wszystkie z nich i każdy zawiera wyszczególniony wyraz.

Podając, np.:

```
p[title~="akapit"] { color: yellow }
```

otrzymamy te akapity w kolorze żółtym, którym został nadany atrybut **title** o wartości, w której występuje wyraz *"***akapit***"*, np. *"***akapit 1***"* lub *"***akapit 2***"* lub *"***ogolny akapit***"*. Nie zadziała dla np. *"***akapit-1***"*.

CSS – selektory atrybutowe

Selektor atrybutu zawierającego myślniki

Selektor ten pozwala nadać styl elementom, którym w kodzie HTML został nadany **określony atrybut o wartości rozpoczynającej się od podanego wyrazu**, a wyraz ten musi być oddzielony myślnikiem lub być sam. Selektor stworzono do obsługi języków (atrybut *lang*), których skróty zawierają często łączniki (np. "en-us", "en-au"), ale może być wykorzystany inaczej (do innych atrybutów).

```
selektor[atrybut|="początek"] { cecha: wartość }
```

(*selektorem* jest dowolny element HTML, *"początek"* to wyraz, od którego rozpoczyna się wartość atrybutu)

Podając, np.:

```
p[title|="akapit"] { color: yellow }
```

otrzymamy akapity w kolorze żółtym, jeżeli został nadany im atrybut **title** o wartości rozpoczynającej się od wyrazu *"***akapit***"* (np.: *"akapit-pierwszy"*).

CSS – selektory atrybutowe

Selektor atrybutu o wartości rozpoczynającej się od…

Selektor ten pozwala nadać styl elementom, którym w kodzie HTML został nadany **określony atrybut o wartości rozpoczynającej się od podanego wyrazu lub ciągu znaków**.

```
selektor[atrybut^="początek"] { cecha: wartość }
```

(*selektorem* jest dowolny element HTML, *"początek"* to wyraz lub ciąg znaków, od którego rozpoczyna się wartość atrybutu)

Podając, np.:

```
p[title^="akapit"] { color: red}
```

otrzymamy akapity w kolorze czerwonym, jeżeli został nadany im atrybut **title** o wartości rozpoczynającej się od wyrazu *"***akapit***"* (np.: *"akapit drugi"* lub *"akapit10"* lub *"akapit-kolejny", …*).

CSS – selektory atrybutowe

Selektor atrybutu o wartości kończącej się na…

Selektor ten pozwala nadać styl elementom, którym w kodzie HTML został nadany **określony atrybut o wartości kończącej się na podanym wyrazie lub ciągu znaków**.

```
selektor[atrybut$="koniec"] { cecha: wartość }
```

(*selektorem* jest dowolny element HTML, *"koniec"* to wyraz lub ciąg znaków, na którym kończy się wartość atrybutu)

Podając, np.:

```
p[title$="akapit"] { color: red}
```

otrzymamy akapity w kolorze czerwonym, jeżeli został nadany im atrybut **title** o wartości kończącej się wyrazem *"***akapit***"* (np.: *"nowy akapit" lub "dalszy-akapit", "11akapit", …*).

CSS – selektory atrybutowe

Selektor atrybutu zawierającego określony tekst

Selektor ten pozwala nadać styl elementom, którym w kodzie HTML został nadany **określony atrybut o wartości zawierającej podany tekst**. Tekst ten może zawierać spacje i może być dowolnym ciągiem znaków.

```
selektor[atrybut*="tekst"] { cecha: wartość }
```

(*selektorem* jest dowolny element HTML, *"tekst"* to dowolny fragment tekstu występujący w wartości atrybutu)

Podając, np.:

```
p[title*="akapit 1"] { color: red}
```

otrzymamy akapity w kolorze czerwonym, jeżeli został nadany im atrybut **title** o wartości zawierającej *"***akapit 1***"* (np.: *"nowy akapit 1 wersja 2"*).

CSS – selektory atrybutowe

Łączenie selektorów atrybutów

Różne typy selektorów atrybutów można łączyć, podając je w regule CSS kolejno po sobie (bez żadnych odstępów). W takim przypadku, aby wybrany element otrzymał określony styl, musi posiadać wszystkie z wyszczególnionych atrybutów z ewentualnymi przypisanymi określonymi wartościami.
Poniższy styl (kolor żółty tekstu)

```
p[class][dir="ltr"][title~="akapit"][lang|="pl-pl"] { color: yellow }
```

otrzyma akapit posiadający np.: następujące atrybuty:

class="…", dir="ltr", title="To jest akapit", lang="pl".

Natomiast akapit o atrybutach:

class="…", dir="ltr", lang="pl"

nie otrzyma tego stylu.

CSS – selektory pseudoelementów

Pseudoelementy

Pseudoelementy CSS pozwalają odnieść się do tych miejsc w strukturze dokumentu HTML, które trudno lub wręcz nie można wskazać wykorzystując elementy HTML, np. pierwsza linia akapitu. Pseudoelementy pozwalają również automatycznie generować pewną zawartość w określonych miejscach dokumentu, która normalnie nie znajduje się w kodzie, np. tekst poprzedzający akapit.

Ogólna składnia reguły CSS dla pseudoelementów:

```
selektor::pseudoelement {cecha: wartość;}
```

CSS3

```
selektor:pseudoelement {cecha: wartość;}
```

CSS1, CSS2

Pierwsza linia

Selektor pierwszej linii pozwala na nadanie określonych cech wszystkim pierwszym linijkom, znajdującym się wewnątrz znacznika, na który wskazuje selektor.

```
selektor::first-line { cecha: wartość; }
```

(*selektorem* jest element blokowy)

Przykład

```
p::first-line { font-weight: 900; }
```

nadaje czcionce we wszystkich pierwszych liniach wchodzących w skład akapitów grubość 900 (maksymalna możliwa).

CSS – selektory pseudoelementów

Pierwsza litera

Selektor pierwszej litery pozwala na nadanie określonych cech pierwszej literze treści, znajdującej się wewnątrz elementu, na który wskazuje selektor.

```
selektor::first-letter { cecha: wartość; }
```

(*selektorem* jest element blokowy)

Przykład

```
p::first-letter { font-size: xx-large; }
```

nadaje wszystkim pierwszym literom wchodzącym w skład akapitów większy rozmiar.

Pseudoelementy można łączyć, np. zastosować do akapitów jednocześnie ::first-letter i ::first-line:

```
p::first-letter { color: red; font-size: large; }
p::first-line { color: blue; font-variant: small-caps; }
```

Możliwe jest też łączenie np. selektora klasy i pseudoelementu:

```
p.intro::first-letter { color: #ff0000; font-size: 200%; }
```

Powyższa reguła odnosi się tylko do pierwszej litery akapitu(-ów) z class="intro".

CSS – selektory pseudoelementów

Poprzedzanie

Reguła poniższa pozwala automatycznie umieścić podaną treść lub obraz przed elementami, na które wskazuje selektor.

selektor::before { content: "*treść*" }

selektor::before { content: *url(ścieżka dostępu do obrazka*) }

(*selektorem* jest element blokowy; ścieżkę dostępu należy konstruować względem dokumentu HTML a nie dokumentu CSS – to ważne, jeśli te dwa dokumenty nie znajdują się w jednym katalogu.)

Przykład

p::before { content: "*Twierdzenie*"; color: red; }

wszystkie akapity będą rozpoczynały się od czerwonego wyrazu "*Twierdzenie*".

CSS – selektory pseudoelementów

Następowanie

Reguła poniższa pozwala automatycznie umieścić podaną treść lub obraz po elementach, na które wskazuje selektor.

selektor::after { content: "*treść*" }

selektor::after { content: *url(ścieżka dostępu do obrazka*) }

(*selektorem* jest element blokowy; ścieżkę dostępu należy konstruować względem dokumentu HTML a nie dokumentu CSS – to ważne, jeśli te dwa dokumenty nie znajdują się w jednym katalogu.))

Przykład

p::after { content: "*koniec dowodu*"; color: red; }

wszystkie akapity będą kończyły się czerwonym wyrazem "*koniec dowodu*".

CSS – selektory pseudoelementów

Markery list

Reguła poniższa pozwala nadać odpowiedni styl markerom dla list uporządkowanych i nieuporządkowanych.

```
::marker{ cecha: wartość}
```

(*selektor* nie jest obecny)

Przykład

```
::marker{
  color: red;
  font-size: 20px;
  }
```

Wszystkie wypunktowania będą miały czerwone markery, a listy uporządkowane dodatkowo czcionkę 20px.

CSS – selektory pseudoelementów

Zaznaczenia

Reguła poniższa pozwala zmienić styl zaznaczonemu przez użytkownika fragmentowi jakiegoś elementu. Możliwa jest zmiana następujących cech: color, background, cursor i outline.

```
::selection { cecha: wartość}
```

(*selektor* nie jest obecny)

Przykład

```
::selection {
  color: blue;
  background: grey;
  }
```

Zaznaczone przez użytkownika fragmenty będą miały szare tło i niebieską czcionkę.

CSS – selektory pseudoklas

Pseudoklasy

Pseudoklasy pozwalają wybierać elementy inaczej niż po ich nazwie, atrybutach czy zawartości. Mogą być dynamiczne w tym sensie, że element nabywa lub traci pseudoklasę wskutek interakcji z użytkownikiem (np. zmiana wyglądu elementu przy wskazaniu go myszką). Pseudoklasa związana jest ze stanem elementu.

Ogólna składnia reguły CSS dla pseudoklas:

```
selektor:pseudoklasa {cecha: wartość;}
```

Wszystkie pseudoklasy można podzielić na grupy:

- Pseudoklasy linków: **:link**, **:visited**
- Pseudoklasy dynamiczne: **:active**, **:hover**, **:focus**
- Pseudoklasa etykiety: **:target**
- Pseudoklasa języka: **:lang()**
- Pseudoklasy interfejsu użytkownika: **:enabled**, **:disabled**, **:checked**
- Pseudoklasy strukturalne: **:root**, **:nth-child()**, **:nth-last-child()**, **:nth-of-type()**, **:nth-last-of-type()**, **:first-child**, **:last-child**, **:only-child**, **:first-of-type**, **:last-of-type**, **:only-of-type**, **:empty**
- Pseudoklasa negacji: **:not()**

Możliwe jest też łączenie np. selektora klasy i pseudoklasy. Przykład:

```
a.menu:hover {color: #ff0000;}
```

CSS – selektory pseudoklas

Odnośnik podstawowy

Reguła pozwala nadać styl wszystkim podstawowym odnośnikom na stronie, czyli takim, które nie zostały jeszcze odwiedzone przez użytkownika.

```
a:link { cecha: wartość; }
```

Odnośnik odwiedzony

Reguła pozwala nadać styl wszystkim odnośnikom na stronie, które zostały już odwiedzone przez użytkownika.

```
a:visited { cecha: wartość; }
```

Zogniskowanie

Reguła nadaje styl elementom, które są w danym momencie zogniskowane (np. pole formularza, w którym znajduje się właśnie kursor).

```
selektor:focus { cecha: wartość; }
```

(*selektor* to dowolny element HTML)

Przykład dla pól formularza:

reguła CSS: `input:focus{background-color:yellow;}`

kod HTML: `<form>Imię: <input type="text" name="imie" /></form>`

CSS – selektory pseudoklas

Aktywacja

Polecenie pozwala nadać styl elementom, które zostały aktywowane przez użytkownika (np. kiedy użytkownik wciśnie i przytrzyma przycisk myszki na odnośniku).

```
selektor:active { cecha: wartość; }
```

(*selektor* to dowolny selektor typu)

Reguła dla odnośnika:

```
a:active { cecha: wartość; }
```

Wskazanie

Polecenie pozwala nadać styl elementom, nad którymi znajduje się wskaźnik myszki, ale nie nastąpiła aktywacja.

```
selektor:hover { cecha: wartość; }
```

(*selektor* to dowolny selektor typu)

Reguła dla odnośnika

```
a:hover { cecha: wartość; }
```

CSS – selektory pseudoklas

Kolejność

Kolejność deklarowania pseudoklas przypisanych do odnośników ma znaczenie. Kolejność inna niż podana poniżej może być przyczyną braku efektu podczas aktywacji czy wskazania myszką. Oczywiście nie wszystkie cztery pseudoklasy muszą być użyte.

```
a:link { cecha: wartość; }
a:visited { cecha: wartość; }
a:hover { cecha: wartość; }
a:active { cecha: wartość;}
```

Pusty element

Reguła pozwala nadać styl elementowi pustemu, np.
, .

```
selektor:empty { cecha: wartość; }
```

(*selektor* to dowolny selektor typu)

Przykład:

```
p:empty { background-color: red; width: 100%; height: 1em; }
```

Formatowanie dla pustych elementów : tło czerwone, szerokość 100%, wysokość 1em.

CSS – selektory pseudoklas

Zastosowanie pseudoklasy strukturalnej w tabeli

Pseudoklasę **:nth-child()** można wykorzystać do uzyskania innego wyglądu wierszy parzystych i nieparzystych. Dla przykładu inne tło dla wierszy parzystych (even):

```
tr:nth-child(even) {background-color: #f2f2f2}
```

Dla wierszy nieparzystych należy w nawiasie podać: *odd*

Dla wskazania co trzeciego wiersza począwszy od trzeciego: *3n+0*

Dodatkowo wiersz nagłówkowy możne mieć odmienny styl nadany z wykorzystaniem elementu komórek nagłówkowych <th>.

```
th { background-color: #000099; color: white; }
```