



Zaawansowane Techniki WWW (HTML, CSS i JavaScript)

Dr inż. Marcin Zieliński

Środa 15:30 - 17:00 sala: A-1-04

WYKŁAD 4

Wykład dla kierunku: **Informatyka Stosowana II rok**

Rok akademicki: **2015/2016 - semestr zimowy**

Przypomnienie

Podstawowe znaczniki HTML

Semantyczny HTML 5

Budowa serwisu internetowego - model pudełkowy

HTML5 - Ogólne zasady

1. Wszystko co było przed wprowadzeniem HTML5 działa w niezaburzony sposób oraz jest obsługiwane.
2. Standaryzacja dotychczasowych technik oraz dostosowanie ich do istniejących przeglądarek.
3. Praktyczność - oznaczająca, że wszelkie zmiany w standardzie języka powinny być uzasadnione praktycznością ich wprowadzania.



Ogólna specyfikacja języka:



www.w3.org/TR/html5

Najnowsze zmiany języka:



<https://html.spec.whatwg.org/multipage/>

DTD dla HTML5

<!DOCTYPE ... >

HTML 4.01
Strict

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

HTML 4.01
Strict

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">

HTML 4.01
Transitional

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

XHTML 1.0
Strict

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

XHTML 1.1

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

HTML 5

<!DOCTYPE html>



Struktura dokumentu w HTML5

<code><!DOCTYPE html></code>	deklaracja typu dokumentu
<code><html></code>	początek dokumentu hipertekstowego
<code><head></code>	sekcja nagłówkowa
<code><title></code> Tytuł strony <code></title></code>	tytuł strony
<code></head></code>	
<code><body></code> <code><p>Treść strony</p></code> <code></body></code>	sekcja zawartości strony
<code></html></code>	koniec dokumentu hipertekstowego

MODEL PUDEŁKOWY

Kodowanie znaków i język strony

Kodowanie jest standardem według którego komputery konwertują tekst na sekwencję bajtów przy zapisie i odczycie danych z pliku. Jest wiele rodzajów kodowania jednak w ostatnich latach ze względu na szybkość działania i obsługę dużej liczby znaków najczęściej wykorzystuje się standard UTF-8.

W HTML5 kodowanie znaków jest bardzo ułatwione i sprowadza się do jednej dyrektywy meta w części head naszej strony internetowej:

Kodowanie:

```
<head>
  <meta charset="utf-8">
  <title> Gzegżółka </title>
</head>
```

Kodowanie znaków i język strony

Kodowanie jest standardem według którego komputery konwertują tekst na sekwencję bajtów przy zapisie i odczycie danych z pliku. Jest wiele rodzajów kodowania jednak w ostatnich latach ze względu na szybkość działania i obsługę dużej liczby znaków najczęściej wykorzystuje się standard UTF-8.

W HTML5 kodowanie znaków jest bardzo ułatwione i sprowadza się do jednej dyrektywy meta w części head naszej strony internetowej:

Kodowanie:

```
<head>
  <meta charset="utf-8">
  <title> Gzegżółka </title>
</head>
```

Dodatkowo dobrą praktyką i oznaką profesjonalizmu jest wskazanie języka strony, co pozwala np. wyszukiwarką na odfiltrowywanie treści pasujących tylko do języka wyszukiwanego. Atrybut wskazujący na język można dodać do dowolnego znacznika jednak najczęściej umieszcza się go w znaczniku html na początku strony:

```
<html lang="pl">
```

Dodawanie arkusza stylu i skryptów

W ostatecznym kształcie strona internetowa korzysta z arkusza stylu (CSS) który nadaje jej odpowiedni wygląd oraz ze skryptu, który obsługuje różne zdarzenia na stronie. Informacje o stylach i skryptach są najczęściej umieszczone w osobnych plikach. Aby dołączyć je do naszej strony musimy zapisać odpowiednie dyrektywy w sekcji head:

Style CSS:


```
<head>  
  <link href="styl.css" rel="stylesheet">  
</head>
```


Dodawanie arkusza stylu i skryptów

W ostatecznym kształcie strona internetowa korzysta z arkusza stylu (CSS) który nadaje jej odpowiedni wygląd oraz ze skryptu, który obsługuje różne zdarzenia na stronie. Informacje o stylach i skryptach są najczęściej umieszczone w osobnych plikach. Aby dołączyć je do naszej strony musimy zapisać odpowiednie dyrektywy w sekcji head:

Style CSS:

```
<head>  
  <link href="styl.css" rel="stylesheet">  
</head>
```



W porównaniu z
HTML4 i XHTML
nie ma atrybutu
`type="text/css"`

Dodawanie arkusza stylu i skryptów

W ostatecznym kształcie strona internetowa korzysta z arkusza stylu (CSS) który nadaje jej odpowiedni wygląd oraz ze skryptu, który obsługuje różne zdarzenia na stronie. Informacje o stylach i skryptach są najczęściej umieszczone w osobnych plikach. Aby dołączyć je do naszej strony musimy zapisać odpowiednie dyrektywy w sekcji head:

Style CSS:

```
<head>  
  <link href="styl.css" rel="stylesheet">  
</head>
```

W porównaniu z
HTML4 i XHTML
nie ma atrybutu
`type="text/css"`

Uwaga:
Ten znacznik musi
być zapisany
dokładnie w ten
sposób, a nie
`<script/>`

Skrypty:

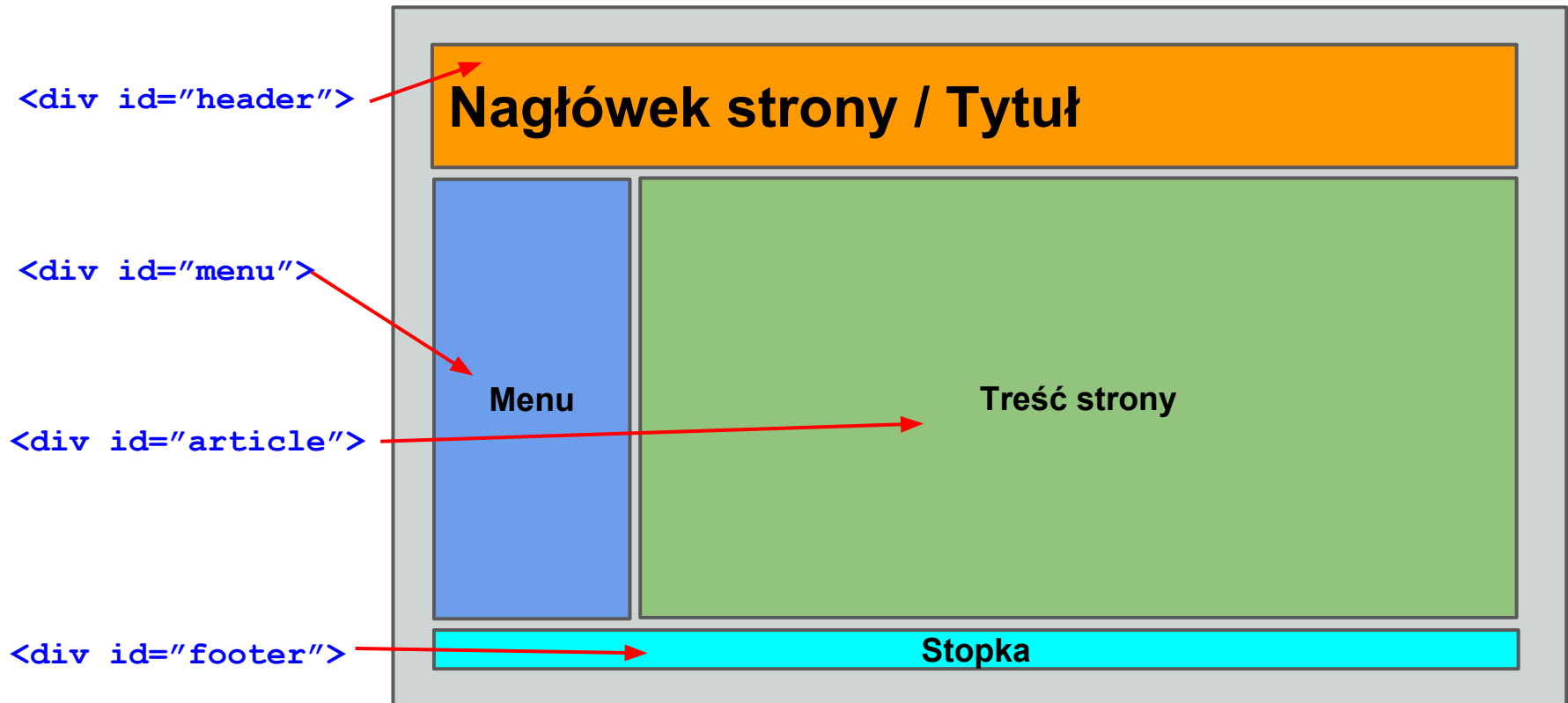
```
<head>  
  <script src="skrypt.js"></script>  
</head>
```

Ostateczna struktura

```
<!DOCTYPE html>
<html lang="pl">
  <head>
    <meta charset="utf-8">
    <link href="styl.css" rel="stylesheet">
    <title>Tytuł strony</title>
    <script scr="skrypt.js"></script>
  </head>
  <body>
    <p>Treść strony</p>
  </body>
</html>
```

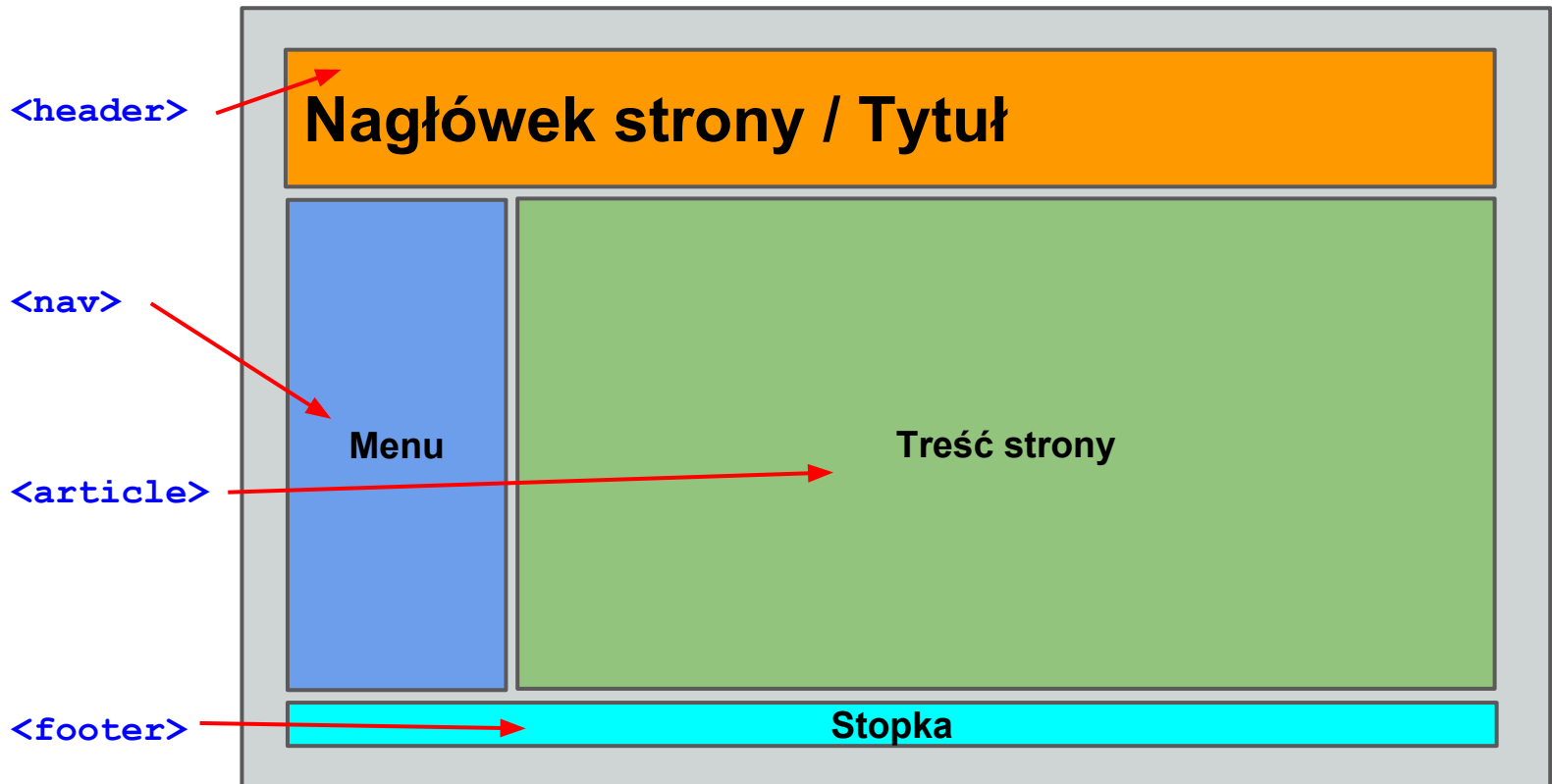
HTML5 i semantyka

Standard HTML5 wprowadza kilkanaście nowych znaczników oraz nowe i lepsze metody organizacji treści na stronie internetowej. Do tej pory strony były zbudowane najczęściej w oparciu o elementy podziału (elementy blokowe) `<div>`, które umożliwiały w miarę logiczne rozdzielanie treści na stronie.



HTML5 i semantyka

Dzięki HTML5 mamy nowe znaczniki które organizują treści na stronie w sposób logiczny oraz dzięki temu są bardziej zoptymalizowane pod kątem wyszukiwarek. Dodatkowo pozwalają na przyjazną organizację strony i czynią stronę bardziej dostępną np. dla czytników ekranów ułatwiających przeglądanie stron osobą niedowidzącym.



HTML5 i semantyka

Dzięki HTML5 mamy nowe znaczniki które organizują treści na stronie w sposób logiczny oraz dzięki temu są bardziej zoptymalizowane pod kątem wyszukiwarek. Dodatkowo pozwalają na przyjazną organizację strony i czynią stronę bardziej dostępną np. dla czytników ekranów ułatwiających przeglądanie stron osobą niedowidzącym.

`<header>`

Nagłówek strony / Tytuł

`<nav>`

Menu

`<article>`

Treść strony

`<footer>`

Stopka

Uwaga:
Znaczniki
same z
siebie
nie
robią
nic!

CSS (Kaskadowe Arkusze Stylu)

CSS (Cascading Style Sheets) [Kaskadowe Arkusze Stylu] - pozwalają na przeniesienie a jednocześnie oddzielenie warstwy formatowania (prezentacji) od treści/danych. Dzięki temu kod strony jest jasny prosty i przejrzysty, a treść strony jest niezależna od jej wyglądu. Same arkusze CSS nie wprowadzają nowych elementów w języku HTML natomiast pozwalają na jego optymalizację i lepsze go wykorzystanie.

Standard CSS został opracowany przez organizację W3C w 1996 roku, jednak jego początki sięgają roku 1994 kiedy został zaproponowany przez **Håkon Wium Lie**.



Håkon Wium Lie

CSS (Kaskadowe Arkusze Stylu)

CSS (Cascading Style Sheets) [Kaskadowe Arkusze Stylu] - pozwalają na przeniesienie a jednocześnie oddzielenie warstwy formatowania (prezentacji) od treści/danych. Dzięki temu kod strony jest jasny prosty i przejrzysty, a treść strony jest niezależna od jej wyglądu. Same arkusze CSS nie wprowadzają nowych elementów w języku HTML natomiast pozwalają na jego optymalizację i lepsze go wykorzystanie.

Standard CSS został opracowany przez organizację W3C w 1996 roku, jednak jego początki sięgają roku 1994 kiedy został zaproponowany przez **Håkon Wium Lie**.



HTML & CSS

Strona projektu:

<http://www.w3.org/Style/CSS/>

CSS 2.1 wersja rekomendowana od 7 lipca 2011



Håkon Wium Lie

CSS (Kaskadowe Arkusze Stylu)

CSS (Cascading Style Sheets) [Kaskadowe Arkusze Stylu] - pozwalają na przeniesienie a jednocześnie oddzielenie warstwy formatowania (prezentacji) od treści/danych. Dzięki temu kod strony jest jasny prosty i przejrzysty, a treść strony jest niezależna od jej wyglądu. Same arkusze CSS nie wprowadzają nowych elementów w języku HTML natomiast pozwalają na jego optymalizację i lepsze go wykorzystanie.

Standard CSS został opracowany przez organizację W3C w 1996 roku, jednak jego początki sięgają roku 1994 kiedy został zaproponowany przez **Håkon Wium Lie**.



HTML & CSS

Strona projektu:

<http://www.w3.org/Style/CSS/>

CSS 2.1 wersja rekomendowana od 7 lipca 2011

CSS 3.0 w trakcie standaryzacji



Håkon Wium Lie

CSS (Kaskadowe Arkusze Stylu)

Kaskadowe arkusze stylu dzielimy na trzy grupy ze względu na miejsce wystąpienia:

1. **Wbudowane (inline-style)** - są zapisane w miejscu ich działania, tzn. w znaczniku, któremu mają nadawać specyficzne cechy.
 2. **Osadzone (embedded-style)** - są zapisane za pomocą znacznika `<style>` w sekcji head dokumentu hipertekstowego.
 3. **Dołączone (linked-style)** - są zapisane w osobnych plikach o rozszerzeniu .css.
-

CSS (Kaskadowe Arkusze Stylu)

Kaskadowe arkusze stylu dzielimy na trzy grupy ze względu na miejsce wystąpienia:

1. **Wbudowane (inline-style)** - są zapisane w miejscu ich działania, tzn. w znaczniku, któremu mają nadawać specyficzne cechy.
2. **Osadzone (embedded-style)** - są zapisane za pomocą znacznika `<style>` w sekcji head dokumentu hipertekstowego.
3. **Dołączone (linked-style)** - są zapisane w osobnych plikach o rozszerzeniu .css.

Która z metod zapisywania CSS ma największy priorytet ?

CSS (Kaskadowe Arkusze Stylu)

Kaskadowe arkusze stylu dzielimy na trzy grupy ze względu na miejsce wystąpienia:

1. **Wbudowane (inline-style)** - są zapisane w miejscu ich działania, tzn. w znaczniku, któremu mają nadawać specyficzne cechy.
2. **Osadzone (embedded-style)** - są zapisane za pomocą znacznika `<style>` w sekcji head dokumentu hipertekstowego.
3. **Dołączone (linked-style)** - są zapisane w osobnych plikach o rozszerzeniu .css.

Która z metod zapisywania CSS ma największy priorytet ?

KASKADOWOŚĆ

CSS (Kaskadowe Akusze Stylu)

KASKADOWOŚĆ (DZIEDZICZENIE)

CSS (Kaskadowe Akusze Stylu)

KASKADOWOŚĆ
(DZIEDZICZENIE)

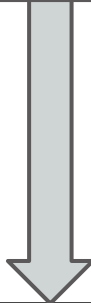


**Kolejność oddziaływania cech
zapisanych w regułach CSS na
elementy strony.**

CSS (Kaskadowe Akusze Stylu)

KASKADOWOŚĆ (DZIEDZICZENIE)

Linked-Style



**Kolejność oddziaływania cech
zapisanych w regułach CSS na
elementy strony.**

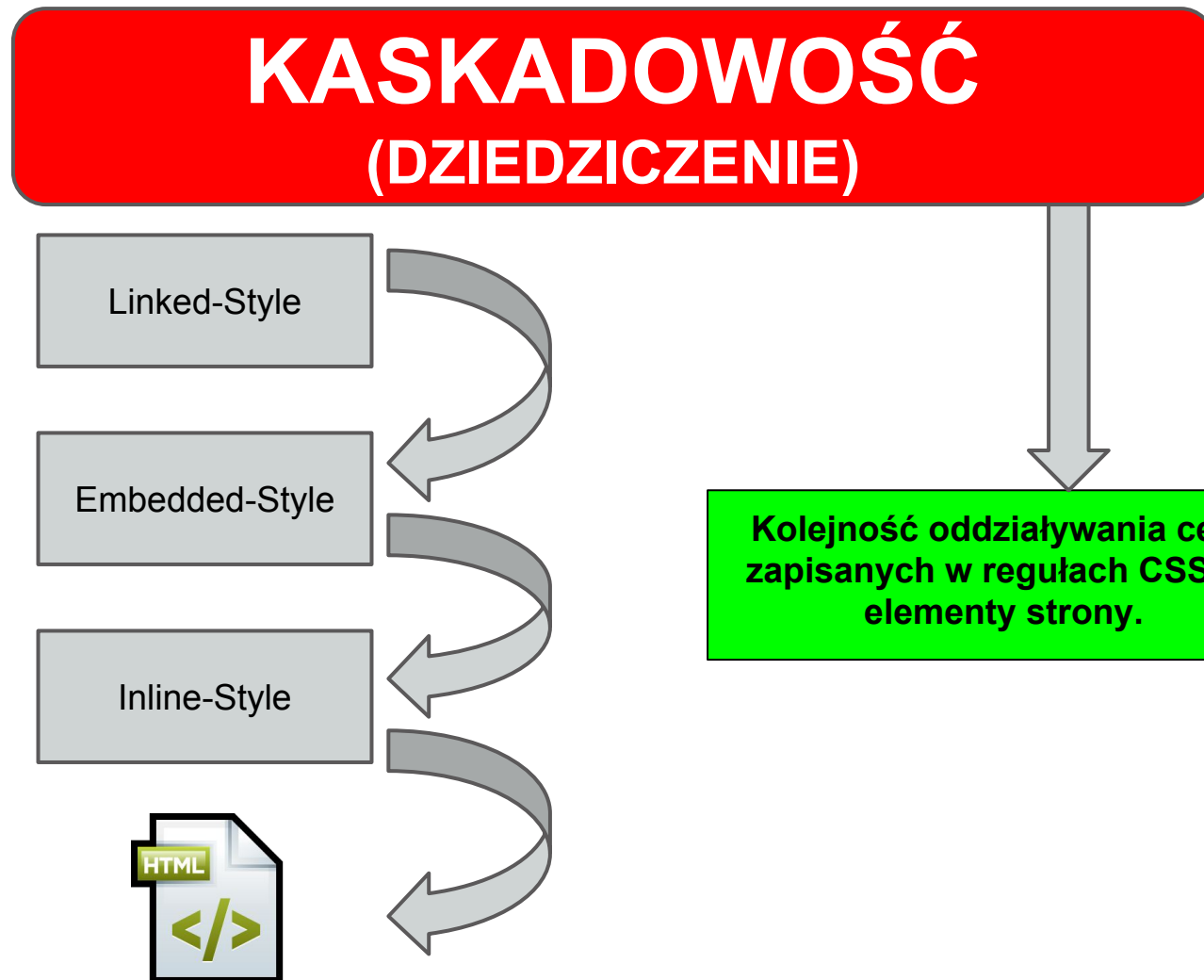
CSS (Kaskadowe Arkusze Stylu)



CSS (Kaskadowe Akusze Styłu)



CSS (Kaskadowe Arkusze Stylu)



CSS (Kaskadowe Arkusze Stylu)

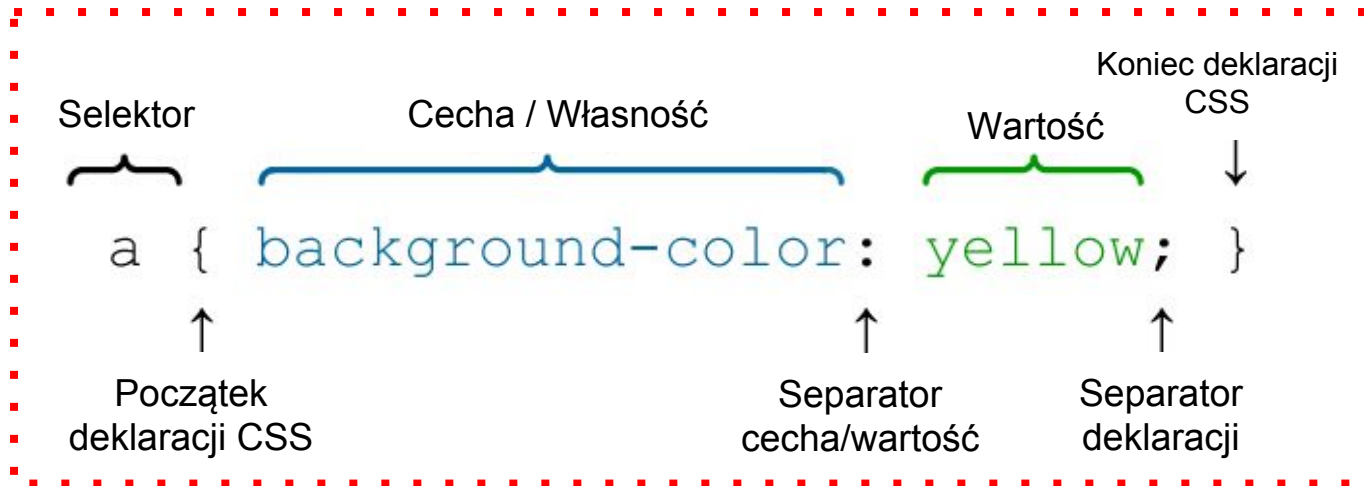
Kaskadowy arkusz stylu zawiera definicję zestawu specyficznych właściwości, jakie mają zostać nadane przez przeglądarkę wybranemu znacznikowi HTML, a następnie wyświetlone użytkownikowi. Reguły w CSS polegają na określeniu dla jakiego znacznika ma zostać nadana specyficzna właściwość oraz jaka ta właściwość ma być. W języku CSS znacznik jest określany mianem "selektora" natomiast właściwość mianem "cechy". Ogólny zapis formatu CSS wygląda następująco:

```
selektor1, selektor2, ... { cecha1: wartość;  
                           cecha2: wartość;  
                           cecha3: wartość;  
                           . . . ;  
                           }
```

W ogólności liczba selektorów oraz cech jest dowolna.

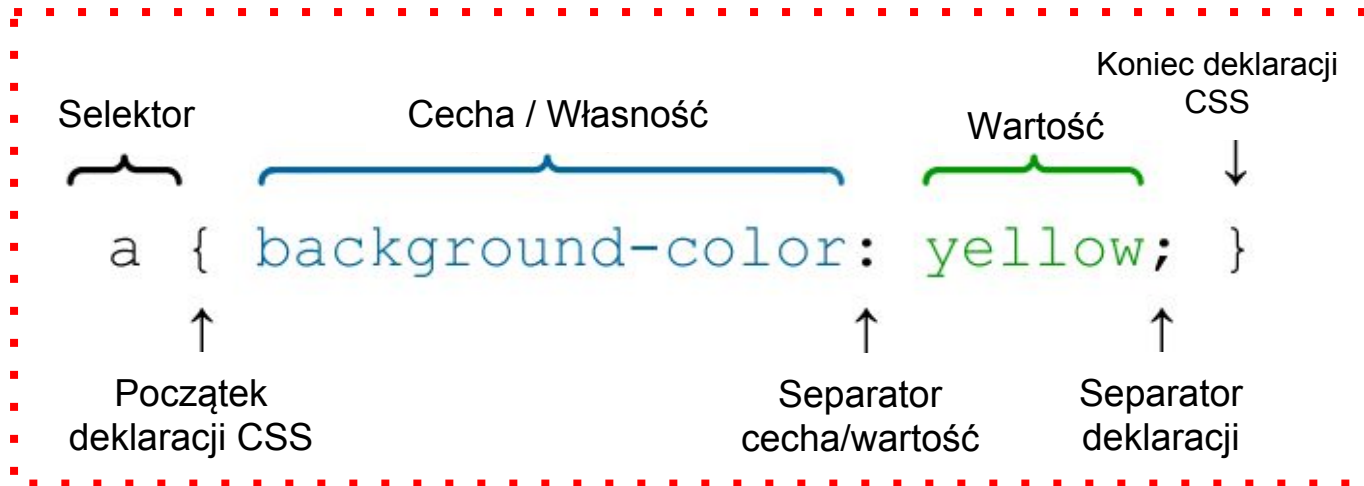
CSS (Kaskadowe Arkusze Stylu)

Przykład:



CSS (Kaskadowe Arkusze Stylu)

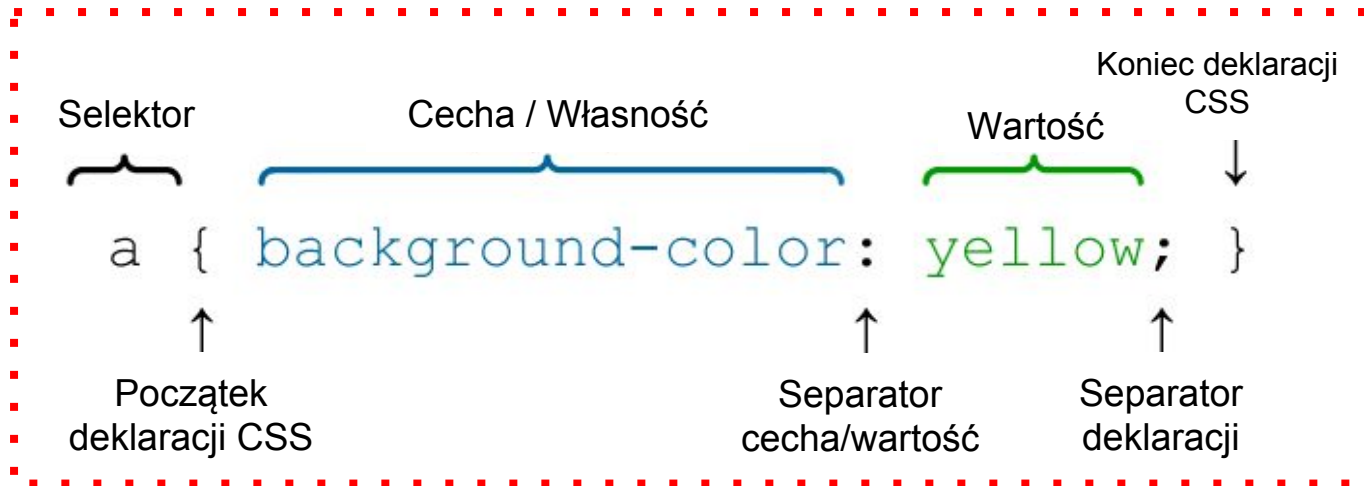
Przykład:



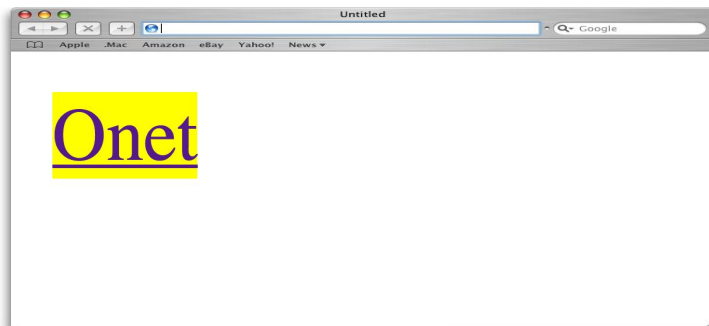
Przykładowa deklaracja powoduje ustawienie koloru tła dla wszystkich hiperłączy na kolor żółty.

CSS (Kaskadowe Arkusze Stylu)

Przykład:



Przykładowa deklaracja powoduje ustawienie koloru tła dla wszystkich hiperłączy na kolor żółty.



```
<a href="http://www.onet.pl">Onet</a>
```

CSS (Kaskadowe Arkusze Stylu)

Linked-Style

```
<body>  
  <a href="http://www.onet.pl">  
    Onet  
  </a>  
</body>
```

Plik
.html

```
a { background-color:  
      yellow;  
}
```

Plik
.css

CSS (Kaskadowe Arkusze Stylu)

Linked-Style

```
<body>
  <a href="http://www.onet.pl">
    Onet
  </a>
</body>
```

Plik
.html

```
a { background-color:
      yellow;
}
```

Plik
.css

Embedded-Style

```
<head>
<style>
  a { background-color: yellow;
    }
</style>
<body>
<a href="http://www.onet.pl">Onet</a>
</body>
```

Plik
.html

CSS (Kaskadowe Arkusze Stylu)

Linked-Style

```
<body>
  <a href="http://www.onet.pl">
    Onet
  </a>
</body>
```

Plik
.html

```
a { background-color:
      yellow;
}
```

Plik
.css

Embedded-Style

```
<head>
<style>
  a { background-color: yellow;
    }
</style>
<body>
<a href="http://www.onet.pl">Onet</a>
</body>
```

Plik
.html

Inline-Style

```
<a style="background-color:yellow;"
  href="http://www.onet.pl">
  Onet
</a>
```

Plik
.html

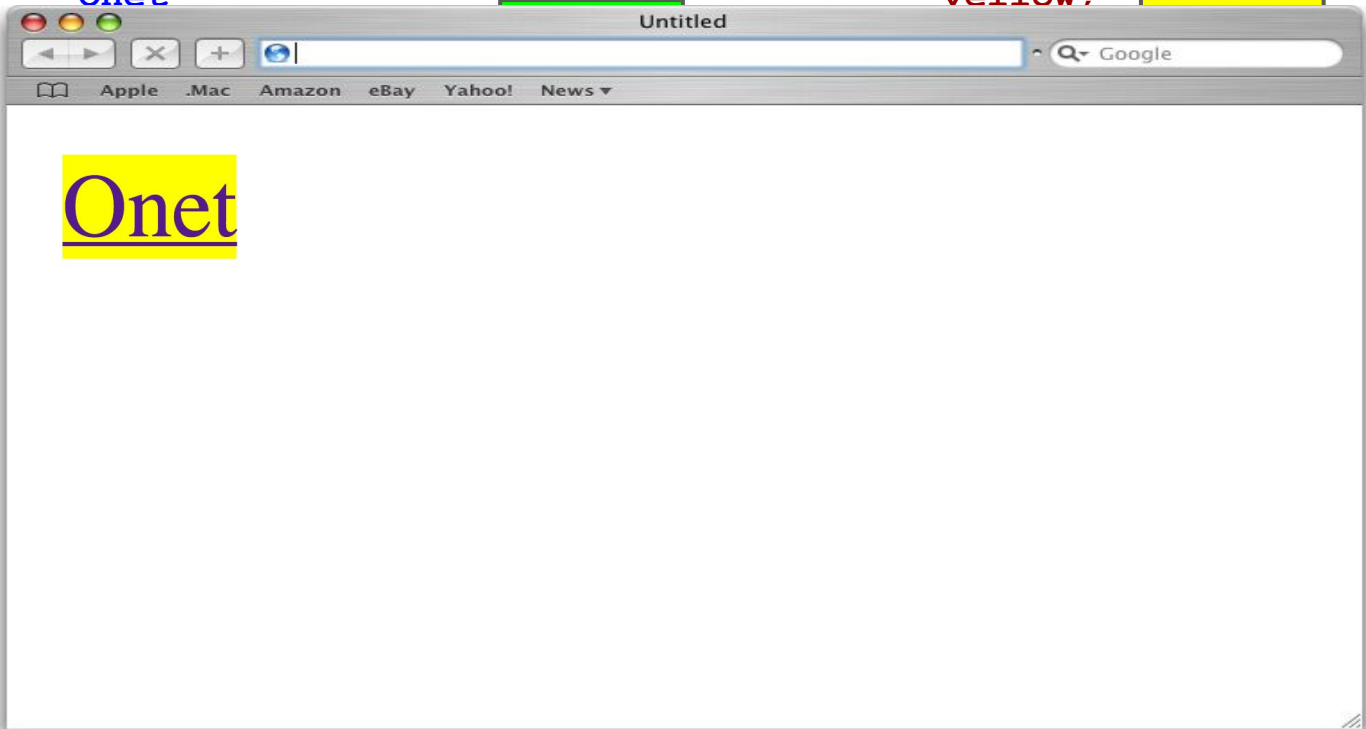
CSS (Kaskadowe Arkusze Stylu)

Linked-Style

```
<body>  
<a href="http://www.onet.pl">  
  Onet
```

```
a { background-color:  
      yellow;
```

Embedded-Style



Inline-Style

```
  href="http://www.onet.pl">  
    Onet  
</a>
```

.html

CSS (Kaskadowe Arkusze Stylu)

Linked-Style

```
<body>  
  <a href="http://www.onet.pl">  
    Onet  
  </a>  
</body>
```

Plik
.html

```
a { background-color:  
      green;  
}
```

Plik
.css

CSS (Kaskadowe Arkusze Stylu)

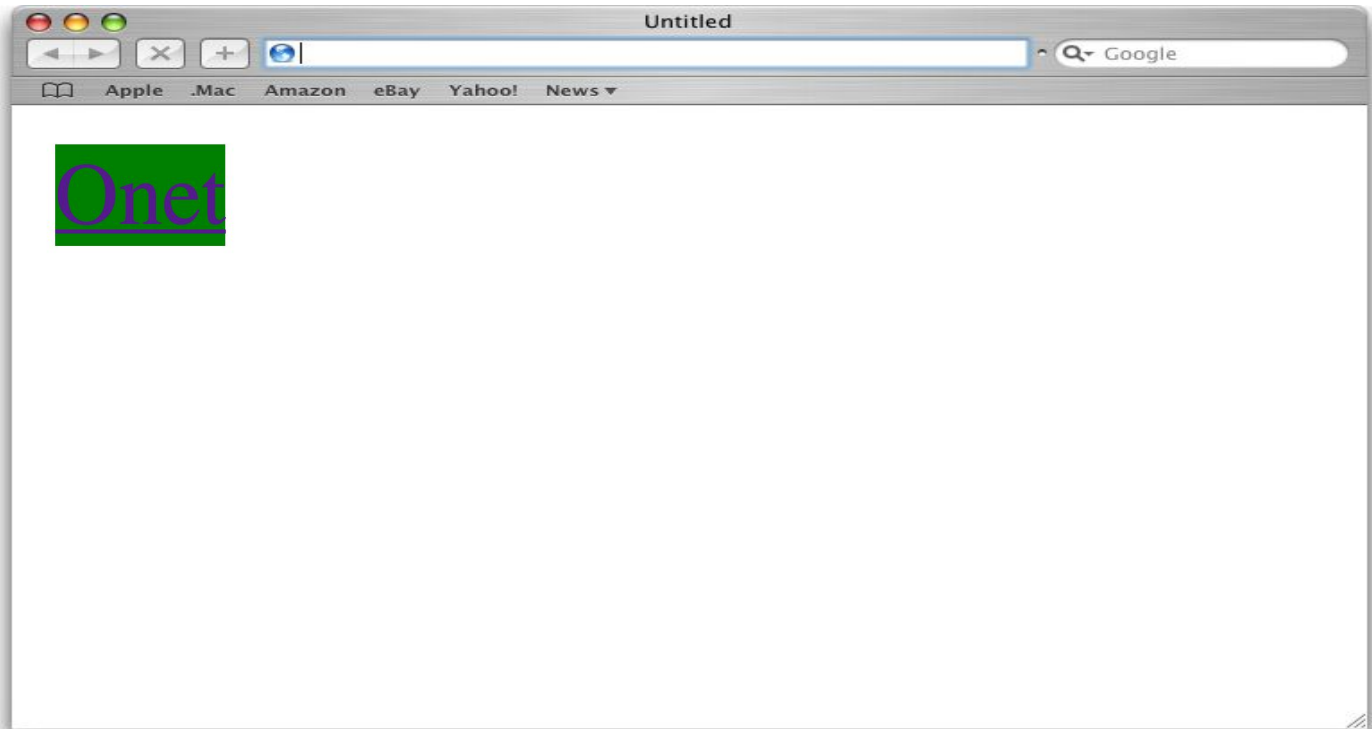
Linked-Style

```
<body>  
  <a href="http://www.onet.pl">  
    Onet  
  </a>  
</body>
```

Plik
.html

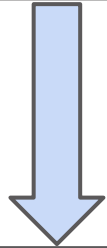
```
a { background-color:  
      green;  
}
```

Plik
.css



CSS (Kaskadowe Arkusze Stylu)

Linked-Style



Embedded-Style

```
<body>
  <a href="http://www.onet.pl">
    Onet
  </a>
</body>
```

Plik
.html

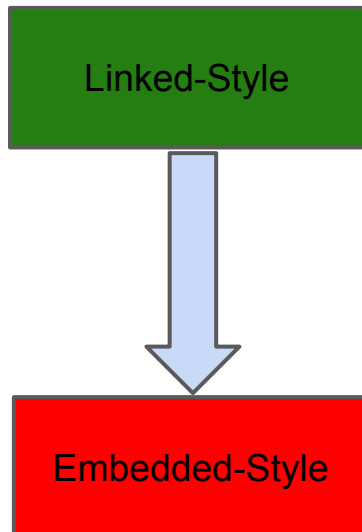
```
a { background-color:
      green;
}
```

Plik
.css

```
<head>
<style>
  a { background-color: red;
    }
</style>
<body>
<a href="http://www.onet.pl">Onet</a>
</body>
```

Plik
.html

CSS (Kaskadowe Arkusze Stylu)

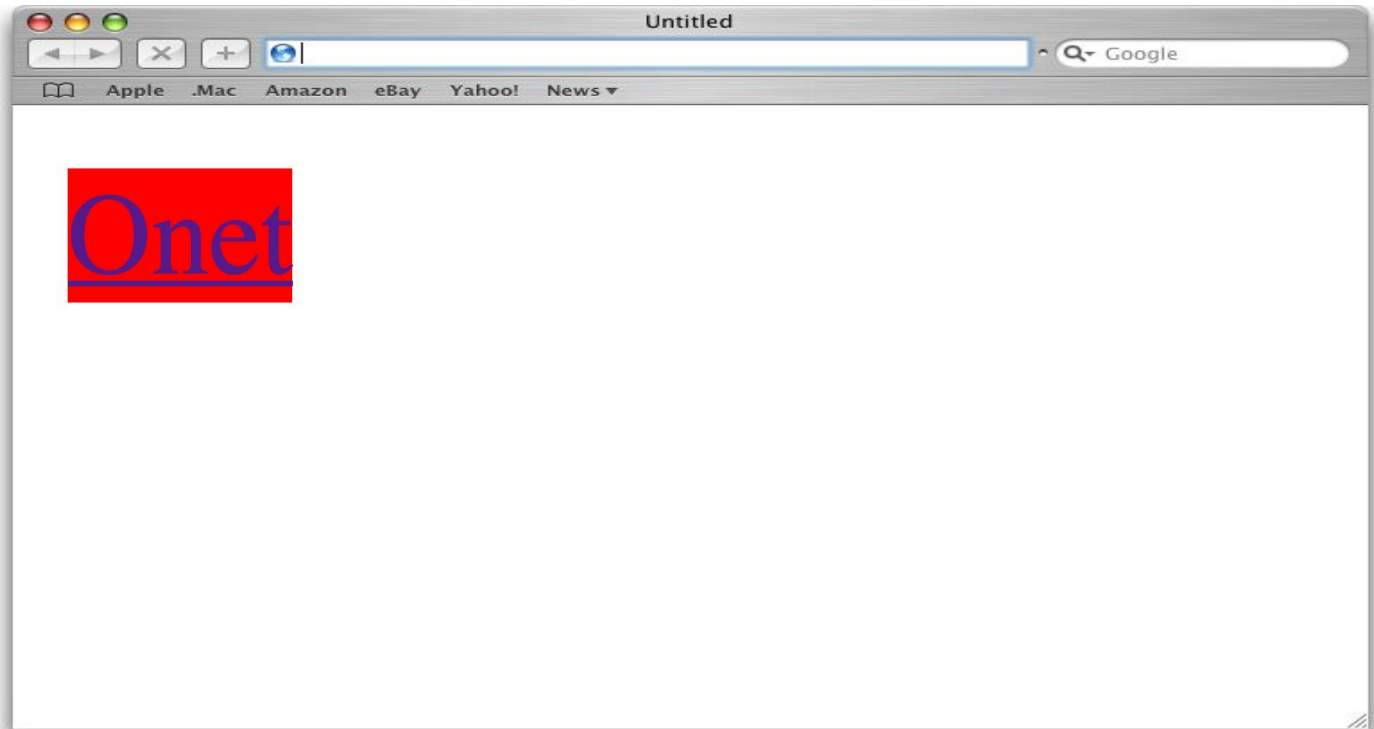


```
<body>
  <a href="http://www.onet.pl">
    Onet
  </a>
</body>
```

Plik
.html

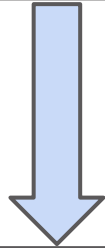
```
a { background-color:
      green;
}
```

Plik
.css

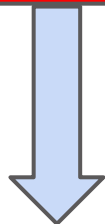


CSS (Kaskadowe Arkusze Stylu)

Linked-Style



Embedded-Style



Inline-Style

```
<body>
  <a href="http://www.onet.pl">
    Onet
  </a>
</body>
```

Plik
.html

```
a { background-color:
      green;
}
```

Plik
.css

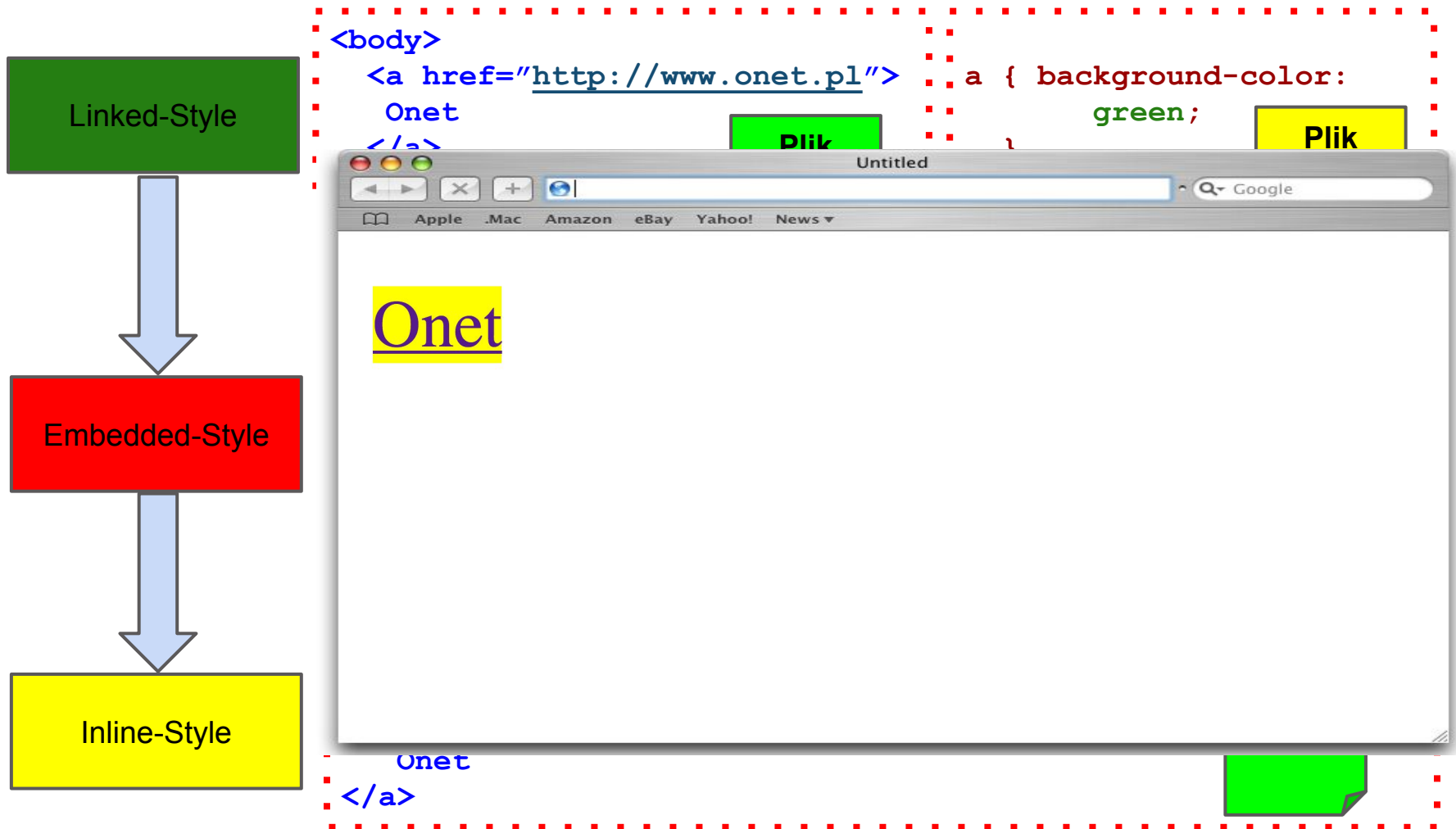
```
<head>
<style>
  a { background-color: red;
    }
</style>
<body>
<a href="http://www.onet.pl">Onet</a>
</body>
```

Plik
.html

```
<a style="background-color:yellow;"
  href="http://www.onet.pl">
  Onet
</a>
```

Plik
.html

CSS (Kaskadowe Arkusze Stylu)



CSS (Kaskadowe Arkusze Stylu)

Dobre praktyki w pisaniu CSS:

1. Stosujemy podobnie jak w html odstępy i wcięcia co pozwala na utrzymanie przejrzystości kodu.
 2. Starajmy się zapisywać style CSS w dołączonych plikach tak aby oddzielić warstwę formatowania od danych.
 3. Dobrą praktyką jest nie nazywanie dołączonych plików CSS np. “styl.css”.
-

Klasy i identyfikatory

Jaka jest różnica pomiędzy “klasą” a “identyfikatorem” ?

Klasy i identyfikatory

Jaka jest różnica pomiędzy “klasą” a “identyfikatorem” ?

Na jednej stronie (jednym pliku) może wystąpić **WIELE** elementów należących do danej klasy (np. znaczniki p/div/span):

```
<p class="czerwony">  
    Tekst rozdziału bardzo podobny znacznik do paragrafu p!  
</p>  
<h1 class="czerwony">  
    Czerwony tytuł nagłówka  
</h1>
```

Na jednej stronie (jednym pliku) może wystąpić **TYLKO 1 RAZ** element oznaczony wybranym identyfikatorem:

```
<h2 id="zielony">  
    Ten tytuł jest zielony  
</h2>
```

Klasy i identyfikatory

W ogólności selektorem nie musi być sam znacznik! HTML umożliwia wprowadzenie oznaczeń dla znaczników za pomocą tzw. “klas” oraz “identyfikatorów”, które pozwalają na odróżnienie od siebie elementów strony:

Klasa:

```
<div class="content">  
  <p> Tekst rozdziału bardzo podobny znacznik do paragrafu p! </p>  
</div>  
<div class="foot">  
  <p> prawa zastrzeżone </p>  
</div>
```

Identyfikator:

```
<div id="main">  
  Tekst rozdziału bardzo podobny znacznik do paragrafu p!  
</div>
```

Jaka jest różnica pomiędzy “klasą” a “identyfikatorem” ?

Klasy i identyfikatory

Jak klasy i identyfikatory pomagają w definicji stylu strony ?

Możemy za pomocą klasy lub identyfikatora konkretnie wybrać element(y) dla którego mają być nadane specyficzne własności i cechy.

Selektor klasy:

```
selektor.nazwaklasz { cecha1: wartość;  
                      cecha2: wartość;  
                      ...;  
}
```

↑
Separator klasy

Selektor identyfikatora:

```
selektor#nazwaklasz { cecha1: wartość;  
                      cecha2: wartość;  
                      ...;  
}
```

↑
Separator
identyfikatora

Klasy i identyfikatory

W ogólności reguły CSS nie muszą być związane z konkretnym znacznikiem (selektorem znacznikowym), ale wystarczy aby odnosiły się do konkretnej klasy lub identyfikatora:

Selektor klasy:

```
.nazwaklasa { cecha1: wartość;  
              cecha2: wartość;  
              ...;  
            }
```

↑
Separator klasy

Selektor identyfikatora:

```
#nazwaklasa { cecha1: wartość;  
              cecha2: wartość;  
              ...;  
            }
```

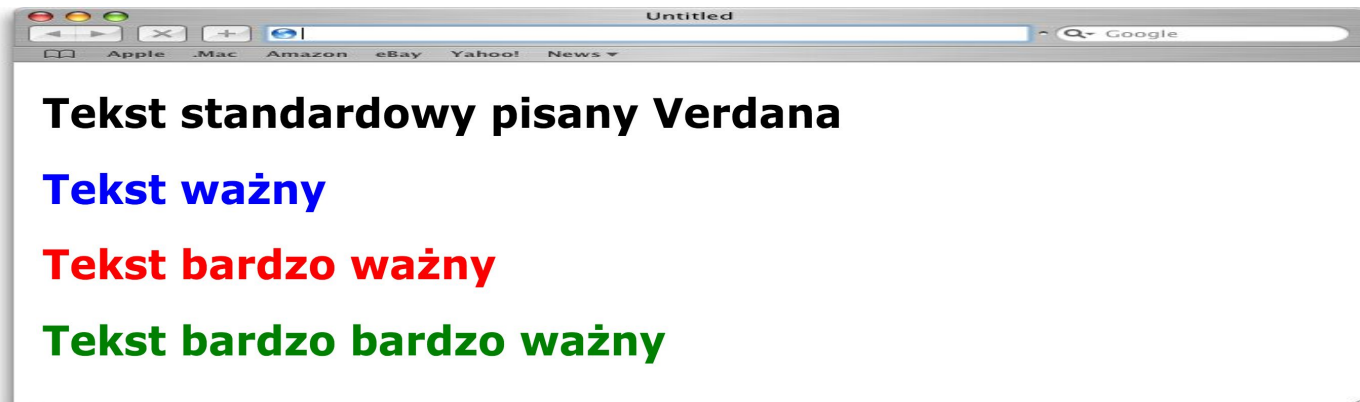
↑
Separator
identyfikatora

Klasy i identyfikatory

Selektor klasy:

```
h1 { font-family: Verdana; }  
h1.wazny { color: blue; }  
h1.bwazny { color: red; }  
h1.bbwarzny {color: green;}
```

```
<h1> Tekst standardowy pisany Verdana </h1>  
<h1 class="wazny"> Tekst wazny </h1>  
<h1 class="bwazny"> Tekst bardzo wazny </h1>  
<h1 class="bbwarzny"> Tekst bardzo bardzo wazny </h1>
```

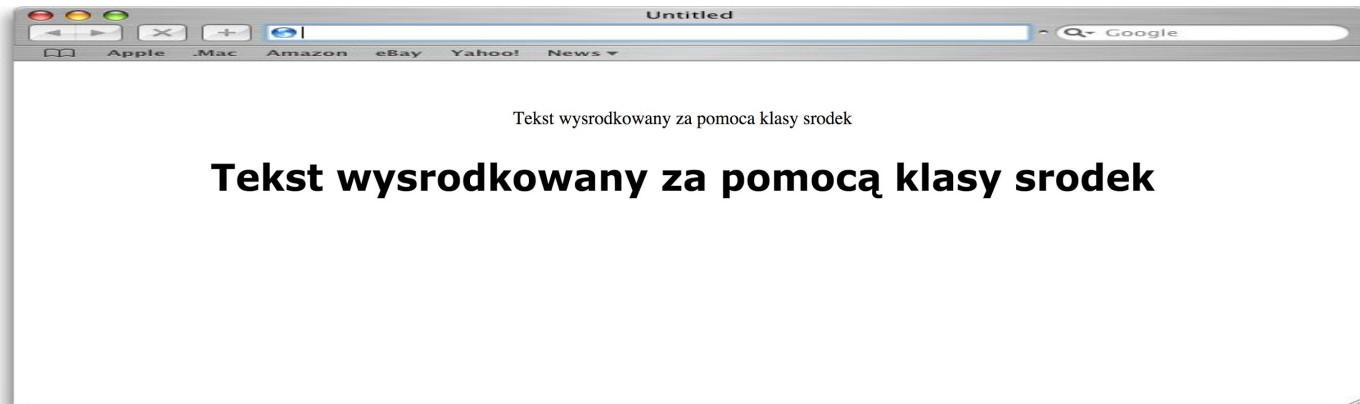


Klasy i identyfikatory

Selektor klasy:

```
.srodek {  
    text-align: center;  
}
```

```
<p class="srodek"> Tekst wysrodkowany za pomoca klasy srodek </p>  
<h1 class="srodek"> Tekst wysrodkowany za pomoca klasy srodek </h1>
```



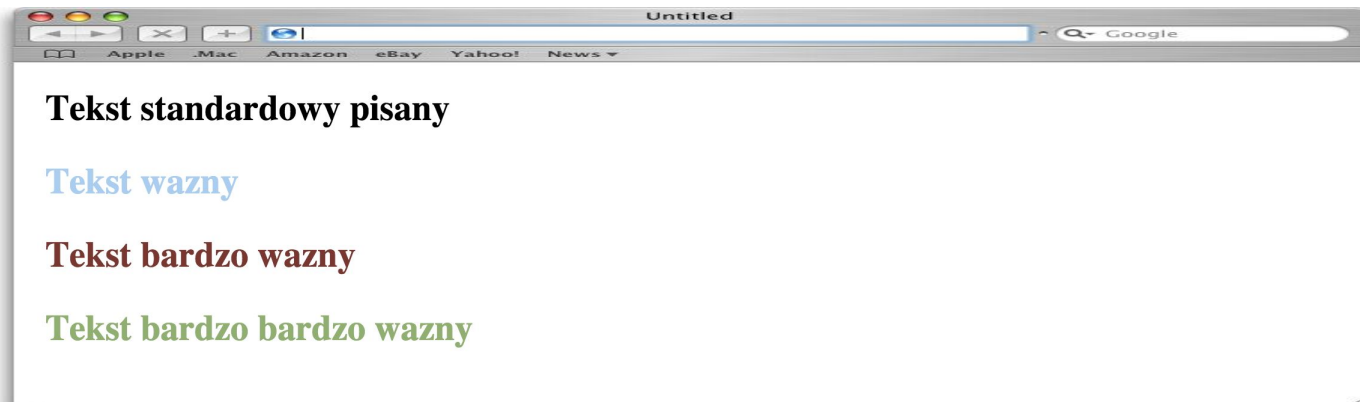
Klasy i identyfikatory

Selektor identyfikatora:

```
h2 { font-family: Times; }  
h2#wazny {color: #ABCDEF; }  
h2#bwazny {color: #793732; }  
h2#bbwazny {color: #91AF72; }
```

Kolory mogą być definiowane z całej palety 24 bitowej za pomocą identyfikatorów HEX.

```
<h2> Tekst standardowy pisany </h2>  
<h2 id="wazny"> Tekst wazny </h2>  
<h2 id="bwazny"> Tekst bardzo wazny </h2>  
<h2 id="bbwazny"> Tekst bardzo bardzo wazny </h2>
```



Najczęściej używane cechy CSS

	Własności
Barwy	color background-color
Przestrzeń i odstępy	margin padding margin-left, margin-right, margin-top, margin-bottom padding-left, padding-right, padding-top, padding-bottom
Obramowanie	border-width border-style border-color border (ustawia grubość, styl i barwę za jednym zamachem)
Wyrównanie tekstu	text-align text-indent word-spacing letter-spacing line-height white-space

Najczęściej używane cechy CSS

	Własności
Fonty	font-family font-size font-weight font-style font-variant text-decoration @font-face
Wielkość	width height
Układ	position left, right float, clear
Grafika	background-image background-repeat background-position

Selektor uniwersalny

Selektor uniwersalny odnosi się do wszystkich elementów zapisanych w dokumencie hipertekstowym i pozwala na nadanie odpowiednich cech i własności wszystkim znacznikom. Selektor uniwersalny jest zapisywany za pomocą gwiazdki:

```
* { cecha1: wartość;  
    cecha2: wartość;  
    ...;  
}
```

Selektor uniwersalny

Selektor uniwersalny odnosi się do wszystkich elementów zapisanych w dokumencie hipertekstowym i pozwala na nadanie odpowiednich cech i własności wszystkim znacznikom. Selektor uniwersalny jest zapisywany za pomocą gwiazdki:

```
* { cecha1: wartość;  
    cecha2: wartość;  
    ...;  
}
```

W praktyce najczęściej stosuje się je do wyzerowania marginesów i odstępów we wszystkich elementach strony np.:

```
* { margin:0 ;  
    padding: 0;  
}
```

Selektor potomka

Selektor potomka stosuje regułę CSS tylko wtedy kiedy dany selektor znajdzie się wewnątrz innego zdefiniowanego selektora na dowolnym poziomie zagnieżdżenia (w dowolnym pokoleniu). Selektor potomka jest oznaczony w regule CSS pisząc bez żadnego separatora nazwy dwóch selektorów:

```
selektor-rodzica selektor-potomka ... { cecha1: wartość;  
                                         cecha2: wartość;  
                                         ...;  
                                         }
```

Przykład: `div span { color:red; }`

```
<div>  
  <span> Text </span>  
</div>
```

```
<div>  
  <p>  
    <span> Text </span>  
  </p>  
</div>
```

W obu przypadkach
reguła CSS będzie
zastosowana.

Selektor “dziecka”

Selektor dziecka stosuje regułę CSS tylko wtedy kiedy dany selektor znajdzie się bezpośrednio wewnątrz innego zdefiniowanego selektora. Selektor dziecka jest oznaczony w regule CSS pisząc pomiędzy nazwami dwóch selektorów znak “>” (większości):

```
selektor-rodzica > selektor-dziecka ... { cecha1: wartość;  
                                           cecha2: wartość;  
                                           ...;  
                                           }
```

Przykład: `div > span { color:red; }`

W tym przypadku
reguła CSS będzie
zastosowana.

```
<div>  
  <span> Text </span>  
</div>
```

W tym przypadku
reguła CSS **NIE**
BĘDZIE zastosowana.

```
<div>  
  <p>  
    <span> Text </span>  
  </p>  
</div>
```

Selektor “brata”

Selektor brata stosuje regułę CSS tylko wtedy kiedy dany selektor znajdzie się bezpośrednio **ZA** innym zdefiniowanym selektorem. Selektor brata jest oznaczony w regule CSS pisząc pomiędzy nazwami dwóch selektorów znak “+” (plus):

```
selektor + selektor-brata ... { cecha1: wartość;  
                                cecha2: wartość;  
                                ...;  
                                }
```

Przykład
1:

```
div + span { color:blue; }
```

```
<div>  
  <h1> Tytuł </h1>  
</div>  
<span> Text </span>
```


Selektor “brata”

Selektor brata stosuje regułę CSS tylko wtedy kiedy dany selektor znajdzie się bezpośrednio **ZA** innym zdefiniowanym selektorem. Selektor brata jest oznaczony w regule CSS pisząc pomiędzy nazwami dwóch selektorów znak “+” (plus):

```
selektor + selektor-brata ... { cecha1: wartość;  
                                cecha2: wartość;  
                                ...;  
                                }
```

Przykład
2:

```
div + p + span { color:blue; }
```

```
<div>  
  <h1> Tytuł </h1>  
</div>  
<p> Inny tekst </p>  
<span> Text </span>
```

Selektor “brataci”

Selektor braci stosuje regułę CSS do wszystkich selektorów znajdujących się bezpośrednio **ZA** danym zdefiniowanym selektorem. Selektor braci jest oznaczony w regule CSS pisząc pomiędzy nazwami dwóch selektorów znak “~” (tyldy):

```
selektor ~ selektor-brata ... { cecha1: wartość;  
                                cecha2: wartość;  
                                ...;  
                                }
```

Przykład: `div ~ span { color:blue; }`

```
<div> </div>  
    <span> czerwony </span>  
    <span> czerwony</span>  
    <span>czerwony</span>  
    <div> <span> czarny </span></div>  
    <span> czerwony </span>  
    <div> <p> <span> czarny </span></p> </div>
```

Selektor atrybutu

Reguły CSS można stosować również w zależności jakie atrybuty przyjmuje dany znacznik w kodzie dokumentu hipertekstowego:

```
selektor[nazwa-atrybutu] { cecha: wartość; }
```

Przykład:

```
<style>
    div[title] {color:red;}
</style>
...

<div title="www"> czerwony </div>
<hr>
<p title="www"> czarny </p>
```

Selektor atrybutu z wartością

Regułę można skonstruować tak aby wybierała elementy, których atrybuty mają konkretną wartość:

```
selektor[nazwa-atrybutu="wartość"] { cecha: wartość; }
```

Przykład 1:

```
<style>
  div[title="www"] {color:red;}
</style>
...

<div title="www"> czerwony </div>
<hr>
<p title="www"> czarny </p>
```

Selektor atrybutu z wartością

Regułę można skonstruować tak aby wybierała elementy, których atrybuty mają konkretną wartość:

```
selektor[nazwa-atrybutu="wartość"] { cecha: wartość; }
```

Przykład 2:

```
<style>  
  [title="www"] {color:red;}  
</style>  
...  
<div title="www"> czerwony </div>  
<hr>  
<p title="www"> czarny </p>
```

Reguła ma zastosowanie do wszystkich elementów zawierających atrybut "title" którego wartość wynosi "www".

Selektor atrybutu z wartością

Inne specyficzne selektory dla atrybutów będą pojawiać się jeszcze w trakcie trwania kursu.

Stylizowanie hiperłączy

Nadanie odpowiedniego stylu hiperłączy (linkowi / odsyłaczowi) stanowi odrębny temat. Hiperłącza charakteryzują się tym że w zależności od wykonanych czynności przez użytkownika w przeglądarce przyjmują różne stany:

- **link** – podstawowy odsyłacz (brak działań użytkownika),
- **hover** – odsyłacz, nad którym zatrzymano kursor myszy,
- **focus** – odsyłacz z tzw. "fokusem", czyli miejscem zaznaczonym przy poruszaniu się po dokumencie przy użyciu klawisza tabulacji. Naciśnięcie klawisza Enter – spowoduje taką samą reakcję przeglądarki, jakby dany odsyłacz został kliknięty,
- **active** – aktywny odsyłacz (kliknięty).
- **visited** – odsyłacz, który prowadzi do wcześniej już odwiedzonego dokumentu HTML, tj. takiego, który znajduje się w historii przeglądarki.

Jak ustawić regułę w zależności od stanu hiperłączy ?

Stylizowanie hiperłączy

Nadanie odpowiedniego stylu hiperłączy (linkowi / odsyłaczowi) stanowi odrębny temat. Hiperłącza charakteryzują się tym że w zależności od wykonanych czynności przez użytkownika w przeglądarce przyjmują różne stany:

- **link** – podstawowy odsyłacz (brak działań użytkownika),
- **hover** – odsyłacz, nad którym zatrzymano kursor myszy,
- **focus** – odsyłacz z tzw. "fokusem", czyli miejscem zaznaczonym przy poruszaniu się po dokumencie przy użyciu klawisza tabulacji. Naciśnięcie klawisza Enter – spowoduje taką samą reakcję przeglądarki, jakby dany odsyłacz został kliknięty,
- **active** – aktywny odsyłacz (kliknięty).
- **visited** – odsyłacz, który prowadzi do wcześniej już odwiedzonego dokumentu HTML, tj. takiego, który znajduje się w historii przeglądarki.

Jak ustawić regułę w zależności od stanu hiperłączy ?



Używamy pseudo-klas odpowiadającym konkretnym stanom.

Stylizowanie hiperłączy

Pseudo-klasy w regułach CSS są zaznaczane za pomocą dwukropka pomiędzy nazwą selektora, a nazwą pseudoklasy:

```
selektor:nazwa-pseudoklasy { cecha: wartość; }
```

```
<style>
```

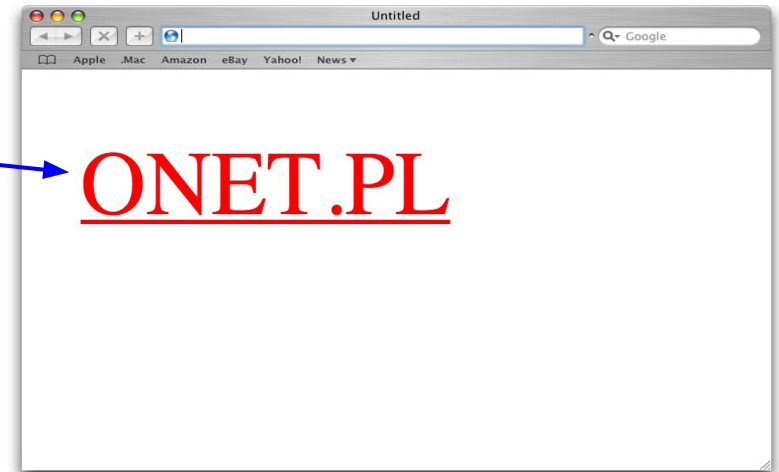
```
  a:link {color: red;}
```

```
  a:visited {color: #9AB676;}
```

```
  a:hover {color: yellow;}
```

```
</style>
```

```
<a href="http://www.onet.pl">ONET.PL</a>
```



Stylizowanie hiperłączy

Pseudo-klasy w regułach CSS są zaznaczane za pomocą dwukropka pomiędzy nazwą selektora, a nazwą pseudoklasy:

```
selektor:nazwa-pseudoklasy { cecha: wartość; }
```

```
<style>
```

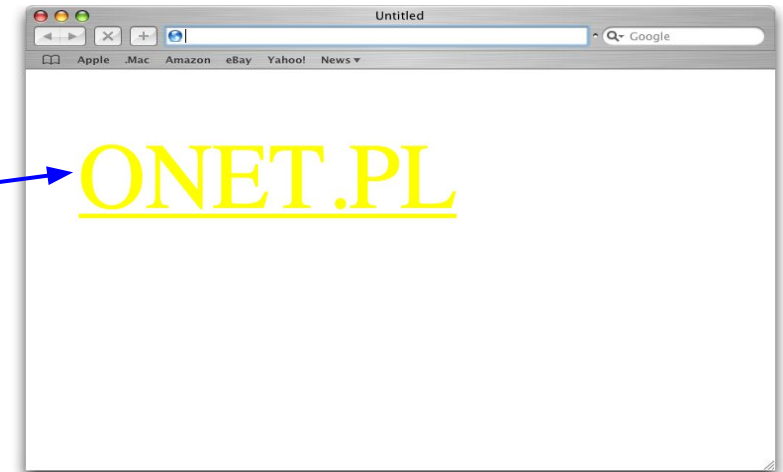
```
  a:link {color: red;}
```

```
  a:visited {color: #9AB676;}
```

```
  a:hover {color: yellow;}
```

```
</style>
```

```
<a href="http://www.onet.pl">ONET.PL</a>
```



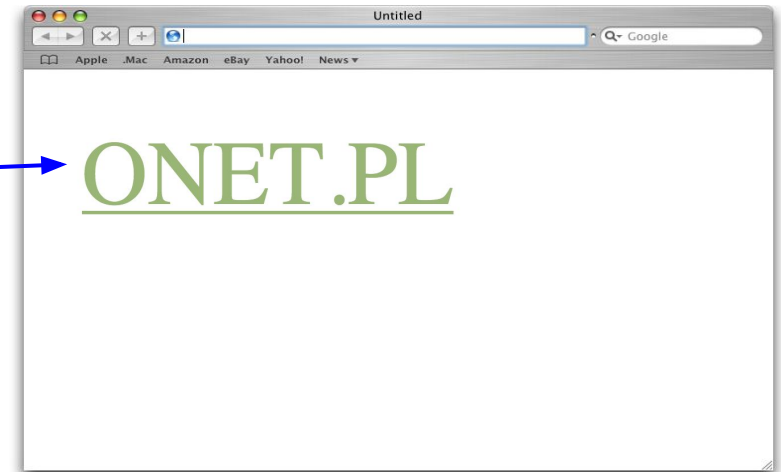
Stylizowanie hiperłączy

Pseudo-klasy w regułach CSS są zaznaczane za pomocą dwukropka pomiędzy nazwą selektora, a nazwą pseudoklasy:

```
selektor:nazwa-pseudoklasy { cecha: wartość; }
```

```
<style>
  a:link {color: red;}
  a:visited {color: #9AB676;}
  a:hover {color: yellow;}
</style>

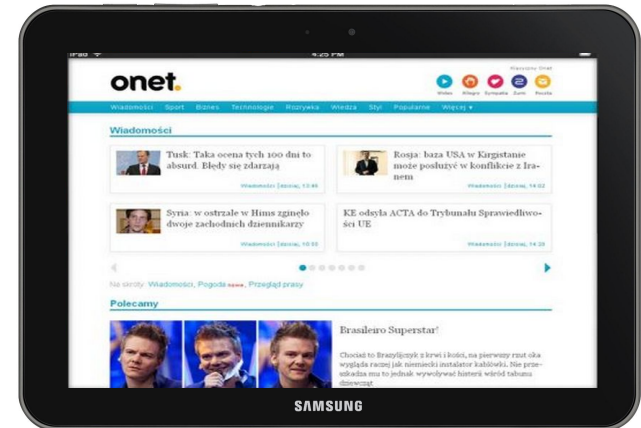
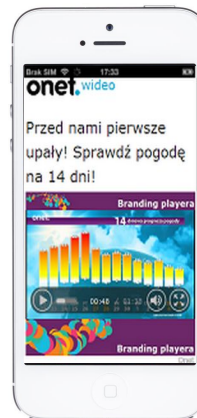
<a href="http://www.onet.pl">ONET.PL</a>
```



Uwaga: Definicje pseudo-klas można łączyć również z klasami i identyfikatorami: np.:
a:link#uj { color: red;} lub **a:link.uj { color: red;}**

Strony dzisiaj ...

Obecnie internet to nie tylko sieć połączonych ze sobą komputerów, ale także telefonów komórkowych, smartfonów, tabletów, lodówek, pralek etc.



Strony dzisiaj ...

Obecnie internet to nie tylko sieć połączonych ze sobą komputerów, ale także telefonów komórkowych, smartfonów, tabletów, lodówek, pralek etc.



Czy będzie poprawnie wyświetlana ?

Czy będzie reagować na działania użytkownika w taki sam sposób ?

Czy będzie przekazywać wszystkie treści ?

Strony dzisiaj ...

Ale dla jakiej rozdzielczości napisać stronę ? Różne modele smartfonów mają różne rozmiary ekranu!



Strony dzisiaj ...

Problemem nie są tylko smartfony ale również telewizory ...

1920 px



1920 px

Strony dzisiaj ...

Obecnie internet to nie tylko sieć połączonych ze sobą komputerów, ale także telefonów komórkowych, smartfonów, tabletów, lodówek, pralek etc.



**3 x TAK
=
RWD**

Czy będzie poprawnie wyświetlana ?

Czy będzie reagować na działania użytkownika w taki sam sposób ?

Czy będzie przekazywać wszystkie treści ?

RWD (Responsive Web Design)

RWD Responsive Web Design [*Reaktywne Projektowanie Stron*] - jest to nowoczesne podejście do projektowania stron internetowych, w którym programista zapewnia, że niezależnie od tego na jakie urządzenie strona zostanie podana wyświetli się ona prawidłowo. W tym kontekście programista zapewnia dostosowanie się wyglądu strony, rozmiaru oraz układu strony do rozmiaru okna przeglądarki w którym będzie wyświetlana.

Jak to osiągnąć ?

RWD (Responsive Web Design)

RWD Responsive Web Design [*Reaktywne Projektowanie Stron*] - jest to nowoczesne podejście do projektowania stron internetowych, w którym programista zapewnia, że niezależnie od tego na jakie urządzenie strona zostanie podana wyświetli się ona prawidłowo. W tym kontekście programista zapewnia dostosowanie się wyglądu strony, rozmiaru oraz układu strony do rozmiaru okna przeglądarki w którym będzie wyświetlana.

Jak to osiągnąć ?



CSS3 &
media queries

CSS3 i Mediaqueries



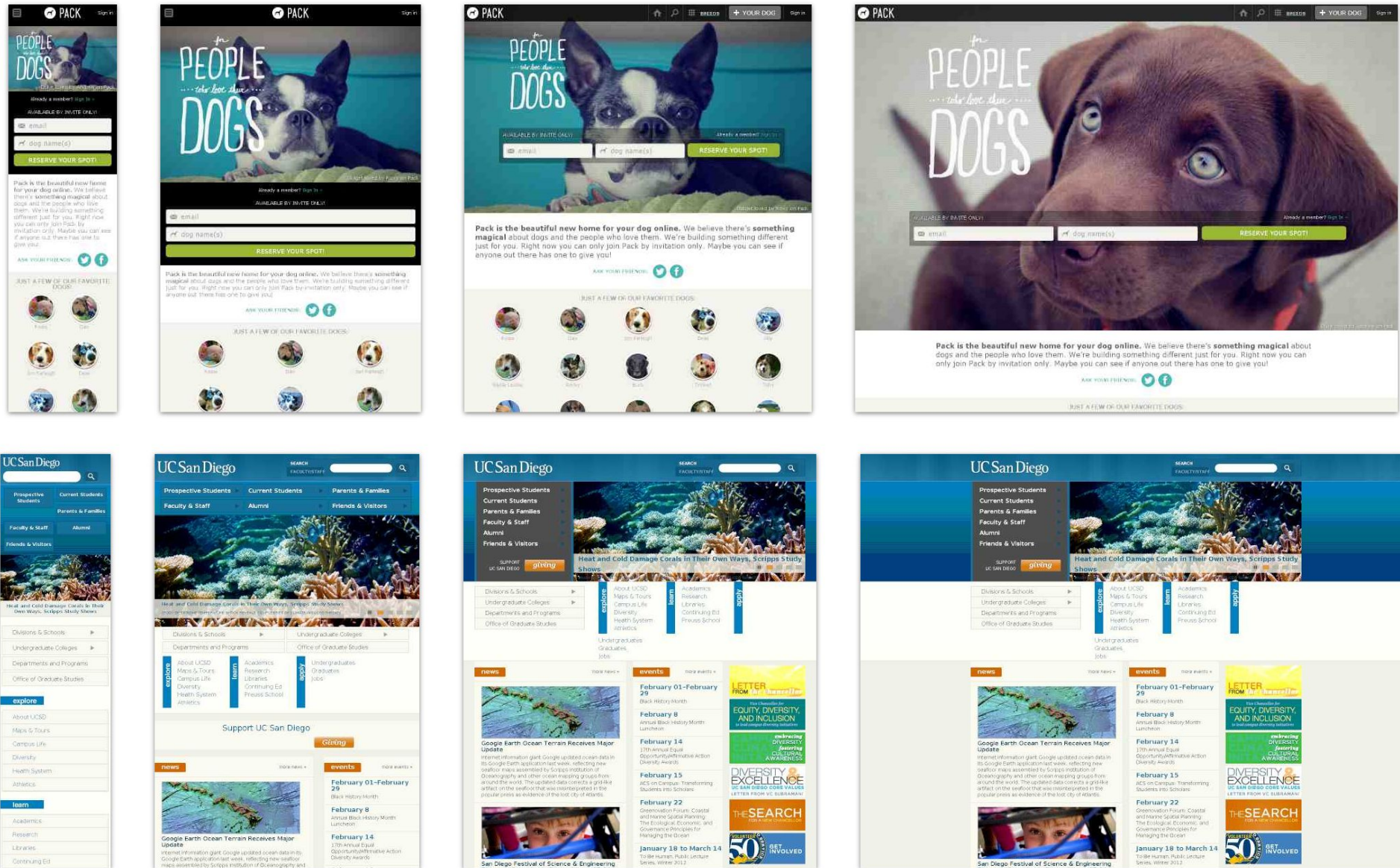
Oficjalna rekomendacja W3C:

<http://www.w3.org/TR/css3-mediaqueries>

CSS3 jest standardem wciąż ulegającym ciągłej ewolucji, jednak metodyka tworzenia tej technologii oparta jest o system modułowy, w którym kolejne elementy są udostępniane użytkownikom po ich opracowaniu dlatego nawet kiedy brak jest całego standardu CSS3 są dostępne jego moduły. Jednym z elementów który został zaprojektowany z myślą o urządzeniach mobilnych i wyświetlaniu na nich stron jest moduł “**media-queries**”, dla którego rekomendacja została zatwierdzona przez W3C w lipcu 2012 roku.

Pierwsza propozycja dla media-queries pojawiła się już w CSS 1.0 (1994 rok), na długo przed powstaniem pierwszych nowoczesnych smartfonów.

CSS3 i Mediaqueries



CSS3 i Mediaqueries



Więcej przykładów projektów na których zastosowano podejście RWD można obejrzeć na stronie:
<http://mediaqueri.es/>



Jak stosować Mediaqueries?

ZASADA:

Jak stosować Mediaqueries?



ZASADA:

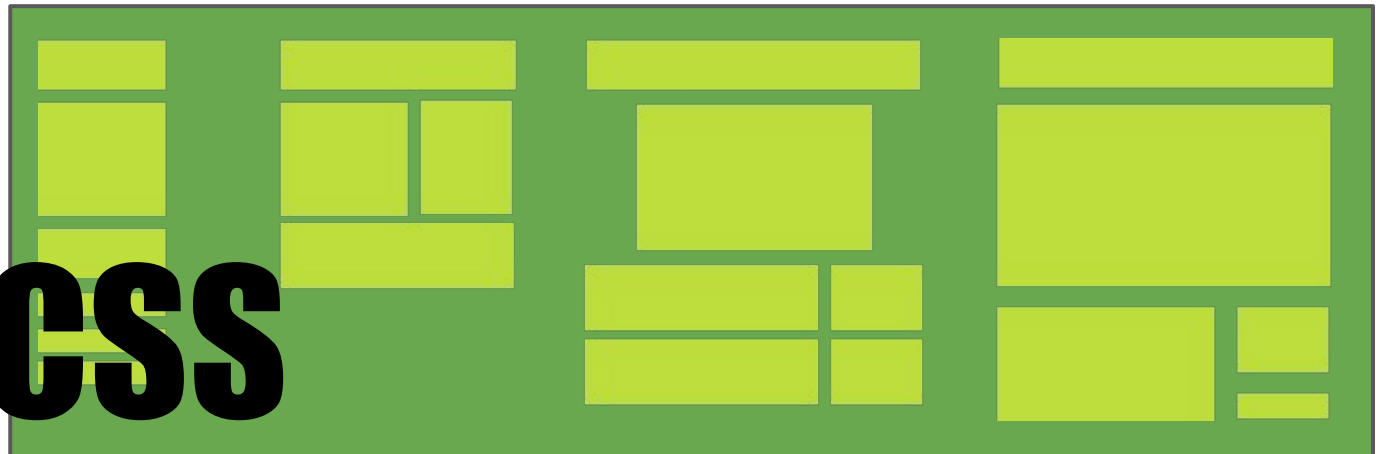
Jak stosować Mediaqueries?



ZASADA:

W ogólności zasada jaka powinna przyświecać tworzeniu stron zgodnie z metodologią RWD jest tworzenie jednego pliku HTML, a dla formatowania jego wyglądu wiele “layoutów” w CSS które będą zawierać odpowiednie reguły wyświetlania strony w zależności od rozdzielczości ekranu.

n x CSS



Czy reguły Mediaqueries zadziałają wszędzie ?

W ogólności zasada Mediaqueries są obecnie wspierane przez większość przeglądarek. W przypadku obecnych smartfonów nie ma z nimi najmniejszego problemu.

mobile

safari iOS, Opera Mini, Opera Mobile,
Android Browser, BlackBerry Browser, Chrome, FF

FF 3.5+

IE 9+

Opera 9.5+

Chrome 4+

Safari 4+

Czy reguły Mediaqueries zadziałają wszędzie ?

W ogólności zasada Mediaqueries są obecnie wspierane przez większość przeglądarek. W przypadku obecnych smartfonów nie ma z nimi najmniejszego problemu.

W przypadku “starszych” przeglądarek stacjonarnych jest trochę gorzej. W przypadku przeglądarek IE<9, FF<3.5 czy Opery <9.5 - niestety reguły te nie są interpretowane. Dlatego trzeba skorzystać z alternatywnych rozwiązań:

Czy reguły Mediaqueries zadziałają wszędzie ?

W ogólności zasada Mediaqueries są obecnie wspierane przez większość przeglądarek. W przypadku obecnych smartfonów nie ma z nimi najmniejszego problemu.

W przypadku “starszych” przeglądarek stacjonarnych jest trochę gorzej. W przypadku przeglądarek IE<9, FF<3.5 czy Opery <9.5 - niestety reguły te nie są interpretowane. Dlatego trzeba skorzystać z alternatywnych rozwiązań:

```
<!--[if lt IE 9]>  
<script src="html5shiv.js"></script>  
<![endif]-->
```

```
<!--[if lt IE 9]>  
<script src="css3-mediaqueries.js"></script>  
<![endif]-->
```

Biblioteki JS takie jak: *html5shiv.js* czy *css3-mediaqueries.js*, które pozwalają za pomocą JavaScriptu podmieniać wartości CSS poszczególnych elementów strony.

CSS3 i Mediaqueries

Aby zdefiniować reguły CSS które będą adaptować stronę w zależności od tego na jakim urządzeniu jest ona w danym momencie wyświetlana musimy skorzystać z “selektora”:

@media

CSS3 i Mediaqueries

Aby zdefiniować reguły CSS które będą adaptować stronę w zależności od tego na jakim urządzeniu jest ona w danym momencie wyświetlana musimy skorzystać z “selektora”:

@media

Selektor ten odnosi się bezpośrednio do określenia typu urządzenia dla którego ma zostać zastosowana reguła CSS. Możliwe typy to np.:

all	Dla wszystkich urządzeń.
aural	Dla syntezy mowy i dźwięku.
braille	Dla urządzeń którymi posługują się niewidomi w celu przeglądania stron internetowych.
embossed	Dla drukarek breilla.
handheld	Dla urządzeń ręcznych bezprzewodowych .
print	Dla drukarek / materiałów drukowanych.
projection	Dla projektorów ściennych.
screen	Dla komputerów z ekranem.
tty	Dla terminali z ograniczonymi możliwościami wyświetlania.
tv	Dla telewizorów.

CSS3 i Mediaqueries

Określenie reguły odbywa się w następujący sposób:

```
@media media-type {  
    selector {  
        cecha: wartość;  
    }  
}
```

gdzie “*media-type*” oznacza określenie typu urządzenia z poprzedniej tabeli.

```
<style>  
    @media screen {  
        div {  
            background-color:red;  
        }  
    }  
</style>
```

CSS3 i Mediaqueries

Przykład dla dwóch urządzeń:

```
<style>
  div {
    background-color:blue;
  }
  @media tv, handheld {
    div {
      background-color:red;
    }
  }
</style>
```

W tym wypadku kiedy strona zostanie wyświetlona na telewizorze lub smartfonie kolor tła elementów div powinien przybrać kolor czerwony, we wszystkich pozostałych przypadkach kolor niebieski.

CSS3 i Mediaqueries

W zależności od tego na jakim urządzeniu wyświetlamy stronę wczytujemy dedykowany plik z regułami CSS:

```
<head>  
  <link media="tv,handheld" href="media.css">  
</head>
```

lub możemy użyć dyrektywy @import:

```
<style>  
  @import url('styl-media3.css') screen and (color);  
</style>
```


CSS3 i Mediaqueries

Dla lepszego sterowania regułami media możemy posługiwać się atrybutami:

Oznaczenie	Przeznaczenie
width	określenie wartości szerokości okna przeglądarki internetowej
height	określenie wartości wysokości okna przeglądarki internetowej
device-width	określenie wartości rozdzielczości ekranu urządzenia (szerokość)
device-height	określenie wartości rozdzielczości ekranu urządzenia (wysokość)
color	określenie liczby bitów na kolor lub określenie czy urządzenie posiada kolorowy ekran
color-index	określenie wartości głębi kolorów, które obsługuje dane urządzenie
aspect-ratio	określenie wartości proporcji szerokości do wysokości okna przeglądarki internetowej

CSS3 i Mediaqueries

Dla lepszego sterowania regułami media możemy posługiwać się atrybutami:

<u>device-aspect-ratio</u>	określenie wartości proporcji szerokości do wysokości rozdzielczości ekranu urządzenia
<u>grid</u>	określenie urządzenia z ograniczonymi możliwościami wyświetlania
<u>monochrome</u>	określenie liczby bitów na piksel w urządzeniach monochromatycznych, jednokolorowych
<u>orientation</u>	określenie orientacji pionowej lub poziomej urządzenia
<u>resolution</u>	określenie wartości gęstości pikseli dla danego urządzenia
<u>scan</u>	określenie czy urządzenie posiada skanowanie obrazu progresywne czy międzyliniowe

CSS3 i Mediaqueries

Oraz operatory logiczne:

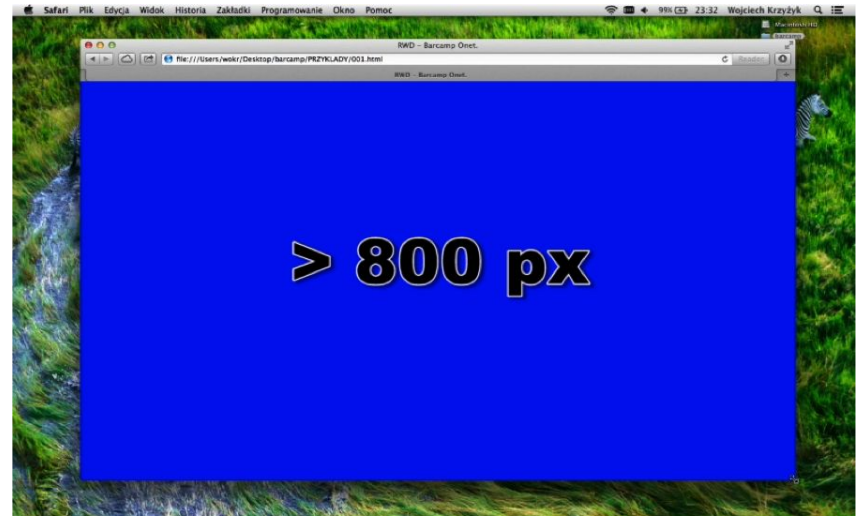
Oznaczenie	Przeznaczenie
and	operator " i ", służy do tworzenia bardziej precyzyjnych warunków w regule @media
przecinek	operator " lub ", służy do tworzenia bardziej precyzyjnych warunków w regule @media
not	operator " negacji ", służy do tworzenia bardziej precyzyjnych warunków w regule @media
only	operator przeznaczony dla starszych przeglądarek internetowych

CSS3 i Mediaqueries

Dodatkowo do poza określaniem konkretnej wartości atrybutów można zadać zakresy dolny i górny:

Reguły ograniczające
z góry i z dołu:

{ min-
max- }



```
@media screen and ( min-width:  
max-width
```

CSS3 i Mediaqueries

Przykład zastosowania:

```
<style>
  @media all and (min-width:480px) and (max-width:800px) {
    div {
      background-color:green;
    }
  }
</style>
```

Definicja określona powyżej mówi przeglądarce że tło elementu <div> ma zostać ustawione na kolor zielony na wszystkich typach urządzeń tylko wtedy kiedy szerokość okna przeglądarki będzie zawierać się w przedziale od 480px od 800px.

CSS3 i Mediaqueries

Przykład zastosowania:

```
<style>
  @media all and (min-color-index:256) {
    div {
      background-color:green;
    }
  }
</style>
```

W przypadku tej reguły kolor tła elementu <div> zostanie zmieniony na zielony tylko wtedy kiedy wartość głębi kolorów możliwych do wyświetlenia na danym urządzeniu nie jest mniejsza od 256.

CSS3 i Mediaqueries

Zastosowanie operatora “only” nie ma on wpływu na definicję reguły, ale zapewnia kompatybilność ze starszymi przeglądarkami, które nie obsługują MediaQueries. Operator był dostępny w wersji 2.1 CSS - ograniczał reguły css dla danego typu "media". Ponieważ starsze przeglądarki nie rozpoznają reguł - nie zinterpretują prawidłowo cssów z media.

```
<style>
  @media only screen (min-width:480px) and (max-width:800px) {
    div {
      background-color:red;
    }
  }
</style>
```

KONIEC WYKŁADU 4
