Witryny i Aplikacje Internetowe

CSS - wstęp

Z czasem pomieszanie warstwy strukturalnego dokumentu z warstwą prezentacyjną zaczęło **stwarzać problemy**. Rozwiązanie tych problemów stało się możliwe po oddzieleniu **warstwy strukturalnej i treści** od **warstwy prezentacyjnej**.

Za warstwę strukturalną, semantyczną i treści odpowiada język HTML. Za warstwę prezentacyjną – język CSS.

CSS - wstęp

Cechy języka CSS:

- oddziela strukturę informacyjną od prezentacyjnej
- większe możliwości formatowania tekstu
- informacje o wyglądzie jako osobny tekstowym
- formatowanie wielu dokumentów jednym arkuszem
- stosowanie układów w zależności od typu urządzenia

CSS - wstęp

Język CSS oferuje nowe **możliwości formatowania**, które były **niedostępne w HTML**. Wśród nich należy wymienić zaawansowane formatowanie tekstu, tła, definiowanie obramowania, dodatkowe właściwości definiowania list, zmianę wyglądu odsyłaczy, stosowanie filtrów.

CSS - wstęp

CSS definiuje **jedynie sposób formatowania elementów** dokumentu HTML. Nie tworzą ich. Elementy muszą być **uprzednio zdefiniowane za pomocą znaczników** w pliku HTML.

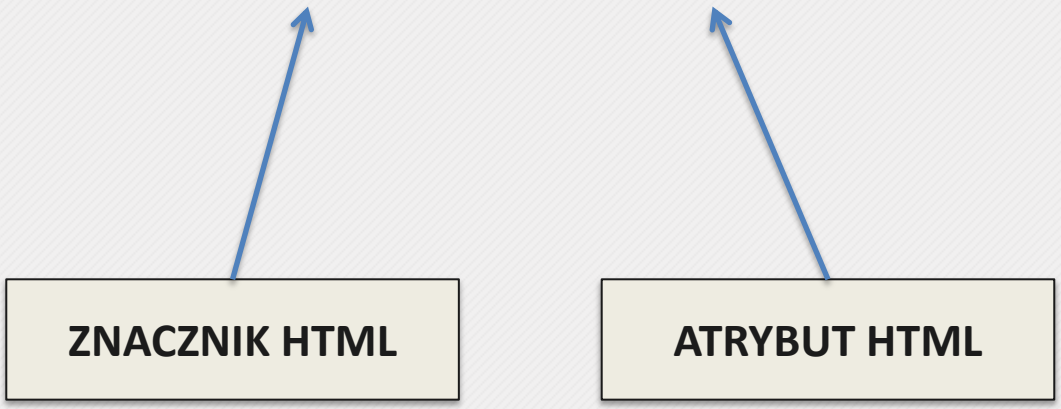
Wstawianie stylów:

- styl lokalny
- wewnętrzny arkusz stylów
- zewnętrzny arkusz stylów

CSS

W podstawowej składni CSS jest kilka stałych elementów.

selektor { właściwości: wartość; }



CSS - wstęp

Przykład:

body { background-color: black; }

Odpowiednik w HTML:

```
<body bgcolor=„black”>
    <!-- zawartość -->
</body>
```

CSS - wstęp

Przykład:

body { background-color: black; color: red; }

Odpowiednik w HTML:

```
<body bgcolor=„black”>
    <font color=„red”>Tekst</font>
    <!-- dalsza zawartość -->
</body>
```

Można zdefiniować kilka atrybutów w selektorze !!!

CSS - styl lokalny

Korzystając ze stylu lokalnego, można zdefiniować formatowanie pojedynczego elementu strony. Taki styl jest definiowany w tej samej linii (stąd nazwa, inline) w atrybucie *style*. Umieszczamy kod bezpośrednio w dokumencie HTML.

Przykład:

```
<body style=„background-color: black”>
    <!-- zawartość strony -->
</body>
```

CSS - styl lokalny

Znacznik <*span*> **służy do grupowania kilku elementów liniowych** strony (np. słów, obrazków) w celu nadania im **określonego stylu**.

Zwykle wykorzystywany jest wtedy, gdy trzeba inaczej sformatować kilka znaków w obszarze o określonym sposobie formatowania.

CSS - styl lokalny

Przykład:
<h3>Nagłówek <span style=„color: red”> H3 </span</h3>

Efekt:
Nagłówek H3

CSS - styl lokalny

Znacznik <*div*> **służy do grupowania kilku elementów liniowych i/lub blokowych** strony (np. słów, obrazków, akapitów) **w bloki** w celu nadania im **określonego stylu**.

Zwykle wykorzystywany jest wtedy, gdy trzeba inaczej sformatować kilka znaków w obszarze o określonym sposobie formatowania.

CSS - styl lokalny

Przykład:

```
<div style=„background-color:yellow”>
<h3>Nagłówek <span style=„color: red”> H3 </span></h3>
</div>
```

Efekt:

Naglowek H3

CSS - styl lokalny

CSS - styl lokalny

```
<div style="background-color:yellow;">
    <h3>Naglowek <span style="color: red;"> H3 </span></h3>
</div>

<div style="background-color:pink;">
    <h3>Naglowek <span style="color: red;"> H3 </span></h3>
</div>

<div style="background-color:yellow;">
    <h3>Naglowek <span style="color: red;"> H3 </span></h3>
</div>
```

CSS - wewnętrzny

Wewnętrzny arkusz stylów, jest umieszczany w dokumencie HTML dzięki **zastosowaniu znacznika** *style.* Występuje on w części nagłówkowej dokumentu HTML.

Metodę tę najlepiej stosować, gdy elementy formatowane **pojawiają się na stronie wielokrotnie** i wszystkie powinny mieć takie same atrybuty formatowania.

CSS - wewnętrzny

Przyjmijmy, że chcemy aby wszystkie nagłówki H3 miały kolor tła domyślnie ustawiony na żółty.

```
<head>
    <style type="text/css">
        div { background-color: yellow; }
    </style>
</head>

<body>
    <div>
        <h3>Naglowek <span style="color: red;"> H3 </span></h3>
    </div>
    <div style="background-color:pink;">
        <h3>Naglowek <span style="color: red;"> H3 </span></h3>
    </div>
    <div>
        <h3>Naglowek <span style="color: red;"> H3 </span></h3>
    </div>

</body>
```

CSS - zewnętrzny

Największą zaletą stosowania CSS jest możliwość wstawiania zewnętrznych arkuszy stylów. Polega ona na **utworzeniu pliku tekstowego** z rozszerzeniem .css, który będzie zawierał definicję **wszystkich stylów** używanych w projektowanej witrynie.

W dokumencie HTML powinien znaleźć się **odnośnik** do tzw. zewnętrznego arkusza, czyli do pliku css.

CSS - zewnętrzny

Odnośnik taki powinien znajdować się w części nagłówkowej i mieć odpowiednią postać.

<link rel=„stylesheet” type=„text/css” href=„arkusz.css”/>

Wartością atrybutu *href* powinna być ścieżka dostępu do pliku.

Warto utworzyć osobny katalog i trzymać się konwencji np. css/style.css.

CSS

Do **zewnętrznego** lub **wewnętrznego** arkusza można zaimportować **zewnętrzny arkusz stylów**. Plik z zaimportowanym arkuszem stylów może znajdować się w dowolnym miejscu.

Polecenie importowania arkusza ma postać:

@import url (adres.http/plik.css);

CSS

Przy użyciu polecenia import w języku **XHTML** zalecane jest stosowanie zapisu elementów nie związanych z znacznikami przy użyciu pola CDATA.

```
/* <! [CDATA[  */
@import url (....)
/* ]]> */
```

Polecenie importu powinno występować na początku arkusza !!!

CSS

Zdarza się, że w **dokumencie** umieszczone są odwołania **do kilku arkuszy** zewnętrznych, wewnętrznych lub lokalnych. Wtedy może się pojawić **konflikt** dotyczący formatowania tego samego elementu w różnych arkuszach.

```
<body style=„color:red;”>
    Tekst  <span style=„color:blue;”>Tekst2</span>
</body>
```

CSS

Zawsze pierwszeństwo mają style, które umieszczone są bliżej formatowanego elementu. Kaskadowość arkuszy stylów polega na ścisłym określeniu priorytetów stylów i przestrzeganiu zasad formatowania zgodnie z priorytetami.

Kaskadowe arkusze CSS

1. **Styl lokalny**

Atrybut style pozwala na dołączenie reguły stylu do konkretnego elementu strony. Atrybut style jest umieszczany wewnątrz kodu treści dokumentu.

Znacznikiem może być prawie każdy znacznik HTML, ale istnieją wyjątki.

Kaskadowe arkusze CSS

Wyjątki:

<base>, <basefont>,<head>,<html>
<meta>,<parm>,<script>,<style>,<title>

Styl taki powinien być stosowany tylko wtedy, gdy konieczne jest zastosowanie specyficznego stylu do pojedynczego wystąpienia danego elementu.

Kaskadowe arkusze CSS

2. Rozciąganie stylu

Rozciąganie stylu polega na objęciu stylem pewnej sekcji dokumentu HTML. Służy do tego para znaczników ** ** .

Element *span* jest szczególnie przydatny, gdy konieczne jest odmienne sformatowanie kilku znaków w obszarze, w którym styl narzucają arkusze stylów.

Kaskadowe arkusze CSS

3. Wydzielone bloki

Wydzielone bloki to kolejna metoda pozwalająca na nadanie blokom dokumentu innego stylu. Służy do tego para znaczników** .**

Metoda ta jest bardzo podobna do *span*, lecz obejmuje zwykle większe fragmenty dokumentu

Kaskadowe arkusze CSS

4. Wewnętrzny arkusz stylów

Deklaracja osadzonego arkusza stylów w **XHTML:**

```
<head>
    <style type="text/css">
    /* <![CDATA[ */
        selektor1 {właściwość: wartość}
        selektor2 {właściwość: wartość}
        selektor3 {właściwość: wartość}
        ...
    /* ]]> */
    </style>
</head>
```

Kaskadowe arkusze CSS

4. Wewnętrzny arkusz stylów

Deklaracja osadzonego arkusza stylów w **HTML:**

```
<head>
    <style type="text/css">
    <!--
        selektor1 {właściwość: wartość}
        selektor2 {właściwość: wartość}
        selektor3 {właściwość: wartość}
        ...
    -->
    </style>
</head>
```

Kaskadowe arkusze CSS

5. Zewnętrzny arkusz stylów

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

Kaskadowe arkusze CSS

6. **Import stylów do zewnętrznego arkusza stylów**

```
<style>
/* <![CDATA[ */
    @import url(adres_zewnętrznego_arkusza_stylów.css);
/* ]]> */
</style>
```

Kaskadowe arkusze CSS

7. **Atrybuty definiowane w HTML**

```
<font color="yellow">Kolor czcionki</font>
```

Kaskadowe arkusze CSS

Styl z numerem 1 ma większy priorytet od tego z numerem 7.

Zasady kaskadowości można **zmieniać**!

Służy do tego polecenie *!important*. Umieszcza się je w **deklaracji stylu po wartości**, której dotyczy.

Kaskadowe arkusze CSS

W języku HTML często występuje **zagnieżdżanie** jednego elementu wewnątrz drugiego. Jeśli dla elementu nadrzędnego w arkuszach stylów zostały zdefiniowane właściwości, to w większości przypadków elementowi **podrzędnemu** zostaną przypisane te same właściwości, nawet **jeśli nie zostały wprost zdefiniowane**. Mechanizm ten nazywamy **dziedziczeniem**.

Kaskadowe arkusze CSS

Arkusz stylów jest zwykłym **plikiem tekstowym**.

W języku CSS posługujemy się **regułami stylów**.
Każda reguła składa się z **selektora** i **deklaracji**.

```
<style type="text/css">
    h1 { color: black; }
</style>
```

selektor { właściwości: wartość; }

Kaskadowe arkusze CSS

W definicji stylu można m.in. **grupować selektory**.

```
<style type="text/css">
     h1,h2 { color: black; }
</style>
```

Kaskadowe arkusze CSS

warto zapamiętać

- **styl lokalny, wewnętrzny, zewnętrzny arkusz**
- **kaskadowość arkuszy stylów**
- **znaczniki span i div**
- **dziedziczenie**
- **składnia języka CSS**
- **grupowanie selektorów**

Kaskadowe arkusze CSS

warto przećwiczyć

Napiszemy nasz pierwszy arkusz CSS.
Przetestujemy działanie atrybutu !important
Sprawdzimy czy dziedziczenie działa.

1. Styl lokalny
2. Rozciąganie (span)
3. Bloki (div)
4. Wewnętrzny CSS
5. Zewnętrzny CSS, atrybuty HTML

Kaskadowe arkusze CSS

W kodzie CSS można również umieszczać **komentarze.**
Komentarze zaczynają się od znaków /* a kończą */

```
/* przykład komentarza */
body { background-color: blue } /* tutaj komentarz */

/*
    tutaj jest dłuższy komentarz
    tutaj jest dłuższy komentarz
*/
```

Kaskadowe arkusze CSS

Każda właściwość używana do definiowania selektora zawiera zbiór dopuszczalnych wartości.

Liczby – całkowite i rzeczywiste (oddzielone kropką)
Procenty
Wartości względne – em, ex
Wartości bezwzględne – in, cm, mm, pc, px
URL – np. url(img.gif)
Kolory – predefiniowane, #RRGGBB, #RGB, rgb(x,y,z), rgba

Kaskadowe arkusze CSS

W języku HTML przyjmuje się, że wartości liczbowe bez podanych jednostek mają jednostkę px. W CSS brak jednostki traktowany jest **jako błąd**.

Gdy podana wartość wynosi 0 wówczas jednostka nie ma znaczenia.

Kaskadowe arkusze CSS

Selektorem może być **dowolny element języka HTML**, dla którego chcemy zdefiniować parametry formatowania.

Rodzaje selektorów:

- elementów
- atrybutów
- specjalne
- pseudoklas
- pseudoelementów

Kaskadowe arkusze CSS

Selektory **elementów**:

- typu	h1 { color: …. }
- uniwersalny	* { color: …. }
- potomka	h i { color: …. }
- dziecka	rodzic>dziecko { color: …. }
- braci	brat1+brat2 { …. }

Kaskadowe arkusze CSS

Selektory **atrybutów**:

- prosty	p[align] { color: …. }
- określonej o wartości	p[align=„left”] { color: …. }
- zawierający	p[align~=„wyraz”] { color: … }

selektor [atrybut=„wartość”] { właściwość: wartość; }

Kaskadowe arkusze CSS

Selektory **specjalne**:

- klasy — p.klasa { color: …. }
  .klasa {color : … }
- identyfikatora — p#id {color : …. }
  #id {color: …. }

Odwołanie do klasy w dokumencie HTML ma postać:

* …. * * …. *

Kaskadowe arkusze CSS

Selektory **pseudoklasy**:

- a:link (nie odwiedzony url)
- a:visited (link odwiedzony)
- a:hover (link gotowy do kliknięcia, kursor nad)
- a:active (link odwiedzany, strona wczytana)
- :focus (element formularza)

Kaskadowe arkusze CSS

Selektory **pseudoelementów**:

- :first-line
- :first-letter
- :first-child
- :before
- :after

Precyzja selektorów

Jest kluczowa do **zrozumienia interakcji** pomiędzy regułami CSS. Jest to liczbowa reprezentacja **dokładności selektora**. Określa się na podstawie trzech czynników.

Każdy selektor wnosi precycję 0,0,0,1
Każda klasa, pseudoklasa, selektor atrybutu 0,0,1,0
Każdy identyfikator ma precycję 0,1,0,0

Precyzja selektorów

Przykład:

div ul ul li	precyzja	0,0,0,4
div.right ul li	precyzja	0,0,1,3
div.right a:hover	precyzja	0,0,2,2
h1#title em	precyzja	0,1,0,2

Obie mają 0,0,0,2 – która będzie wyświetlona?

```
ul li { color: red; }
html li {color: black}
```

Precyzja selektorów

Obie mają 0,0,0,2 – która będzie wyświetlona?
ul li { color: red; }
html li {color: black}

Mimo, iż element ul jest składniowo „bliższy” li to zostanie zastosowana ta, która wpisana została ostatnia.

Tak więc, nie bierze się pod uwagę struktury dokumentu.

Precyzja selektorów

Pierwsze zero w zapisie precyzji selektorów dotyczy stylów **śródliniowych** (ang. Inline).

Przykład:

div { color: black; }	prezycja 0,0,0,1
<div style=„color: yellow”>	prezycja 1,0,0,0

Precyzja selektorów

Reguła ważności !important **przewyższa siłą** reguły precyzji. Jeśli deklaracja oznaczona jest jako !important jest wówczas **ważniejsza od wszystkich deklaracji** bez takiego oznaczenia.

Przykład:

```
div#home a#home { color: black; }
div a { color: yellow !important; }
```

Precyzja selektorów

Przykład 2:
div#home a#home { color: black !important; }
div a { color: yellow !important; }

Ponieważ **obie deklaracje** kolorów są oznaczone jako !important, konflikt został rozwiązany według **typowych zasad kaskadowości**.

Kaskadowe arkusze CSS

W CSS **formatowanie elementów** strony jest realizowane przez **ustawianie właściwości** tych elementów.

Czcionki są **najistotniejszym elementem strony**, który podlega formatowaniu.

Rodzaj czcionki : font-family: rodzaj1,rodzaj2,…. ;
(serif, sans-serif, monospace, cursive, fantasy)

Kaskadowe arkusze CSS

Rozmiar czcionki: font-size: rozmiar;

Według słów **kluczowych** (wzg. rozm. podstawowego) :

- xx-small
- x-small
- small
- medium
- large
- x-large
- xx-large

Kaskadowe arkusze CSS

Rozmiar czcionki: font-size: rozmiar;

Za pomocą wartości **względnych** (względem el. nadrz.):

- smaller
- larger

Jednostki miary:

- px,pt,in,cm,mm

Kaskadowe arkusze CSS

Rozmiar czcionki: font-size: rozmiar;

Istnieje też możliwość podania wielkość w procentach (względem rozmiaru podstawowego).

Kaskadowe arkusze CSS

Styl czcionki: font-style : styl;

Dostępne style:

- normal
- italic
- oblique

Kaskadowe arkusze CSS

Wariant czcionki: font-variant : wariant;

Dostępne warianty:

- normal
- small-caps (kapitaliki)

Kaskadowe arkusze CSS

Grubość czcionki: font-weight: wartość;

Dostępne warianty:

- normal
- bold
- lighter
- bolder

Kaskadowe arkusze CSS

Odstęp między wierszami: line-height: wartość;

Dostępne warianty:

- liczba (wielokrotność aktualnego odstępu)
- wysokość (px,mm,cm)
- procent

Kaskadowe arkusze CSS

warto przećwiczyć

Ćw. 1

Wykorzystując arkusze stylów, zdefiniuj style określające właściwości czcionki (rodzaj, rozmiar, styl) dla znaczników h2,h3, p

Podpowiedź:
font-style (normal, italic, oblique),
font-variant (normal, small-caps), font-weight(px,bold,bolder,…)
font-size, line-height,
font-family (serif, sans-serif, monospace,cursive)

Kaskadowe arkusze CSS

Język CSS pozwala na **dowolne formatowanie tekstu** poprzez dodanie do niego stylu. Umożliwia to nie tylko szybką zmianę wyglądu tekstu na stronie, ale także **swobodne manipulowanie tekstem** i umieszczaną grafiką.

Atrybut **wcięcia tekstu** *text-indent* umożliwia definicję wcięcie pierwszego wiersza akapitu.

Kaskadowe arkusze CSS

Przykład:

```
<p>
    Akapit wieloliniowy.<br>
    Akapit wieloliniowy
</p>
<p style="text-indent:20px">
    Akapit wieloliniowy.<br>
    Akapit wieloliniowy
</p>
```

Akapit wieloliniowy.
Akapit wieloliniowy

Akapit wieloliniowy.
Akapit wieloliniowy

Kaskadowe arkusze CSS

Wyrównanie tekstu przy użyciu atrybutu *text-align* jest odpowiednikiem *align* w języku HTML.

Wartości atrybutu - left|right|center|justify

Kaskadowe arkusze CSS

Przykład:


```
<h1 style="text-align:left">
    Wyrównanie tekstu
</h1>
<h1 style="text-align:right">
    Wyrównanie tekstu
</h1>
```

Kaskadowe arkusze CSS

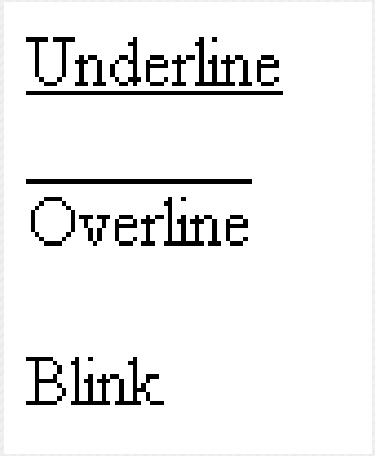
Do dodawania **efektów** takich jak: podkreślenie, przekreślenie, umieszczanie linii nad tekstem, używany jest atrybut *text-decoration*:

Wartości atrybutu - none|underline|overline|line-through|blink

Kaskadowe arkusze CSS

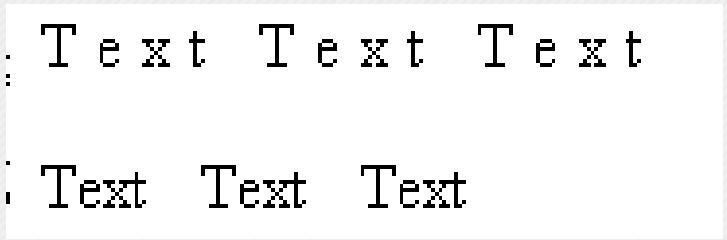
Przykład:

```
<p style="text-decoration:underline">
    Underline
</p>
<p style="text-decoration:overline">
    Overline
</p>
<p style="text-decoration:blink">
    Blink
</p>
```



Kaskadowe arkusze CSS

Odstęp **pomiędzy literami** uzyskuje się dodając atrybut *letter-spacing*.
Odstęp pomiędzy **wyrazami** można zmienić ustawiając atrybut *word-spacing*.



Kaskadowe arkusze CSS

Transformacja tekstu możliwa jest przy użyciu atrybutu text-transform. Atrybut kontroluje wielkość liter w tekście i dokonuje ich transformacji.

Proszę wypróbować atrybuty: capitalize, uppercase, lowercase

Kaskadowe arkusze CSS

Sposób wyświetlania białych znaków można kontrolować przy użyciu atrybutu *white-space*.

Dostępne wartości:

normal, pre,nowrap, pre-wrap,pre-line

Kaskadowe arkusze CSS

warto przećwiczyć

Ćw. 1

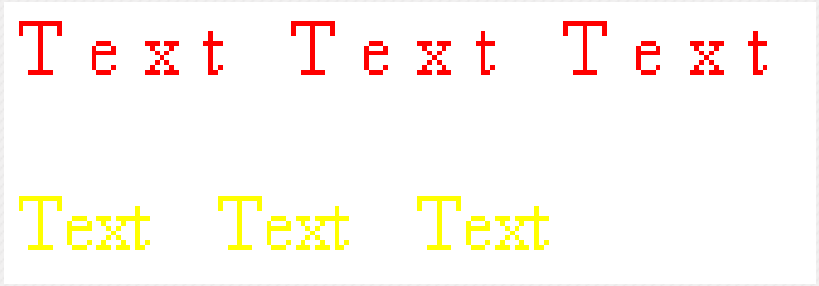
Wykorzystując poznane techniki formatowania tekstu stwórz pasek z ważna informacja.

Podpowiedź:
text-indent, text-align, text-decoration,letter-spacing, word-spacing
text-transform (uppercase, capitalize, lowercase),
white-space (pre,wrap)

Tło i kolor w CSS

Korzystając z atrybutu color można opisać **pierwszoplanowy kolor** wybranego elementu.

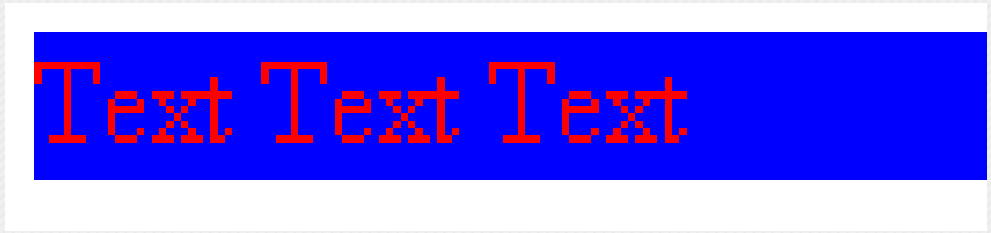
```
<p style="color: red;letter-spacing:5px">
    Text Text Text
</p>
<p style="color:yellow;word-spacing:10px">
    Text Text Text
</p>
```



Tło i kolor w CSS

Kolor tła elementu jest definiowany za pomocą atrybutu background-color

```
<p style="background-color:blue;color: red;">
    Text Text Text
</p>
```



Tło i kolor w CSS

Do umieszczenia **elementu graficznego jako tło** służy atrybut backgroud-image.

Bardzo często wykorzystywany jest łącznie z atrybutem background-repeat. Definiuje on **powtarzanie obrazka**.

Repeat – w obu kierunkach
Repeat-x – w kierunku poziomym
Repeat-y – w kierunku pionowym
No-repeat – brak powtarzania

Tło i kolor w CSS

Przykład:

```
<style type="text/css">
    body { background-image: url(link do obrazka);
</style>
```

Tło i kolor w CSS

Jeżeli treść strony jest przewijana za pomocą suwaka, to wstawiona jako tło strony grafika przesuwa się razem z tekstem. Aby grafika w trakcie takich działań była **nieruchoma należy** użyć atrybutu background-attachment.

Dostępne wartości: scroll, fixed, local

Tło i kolor w CSS

Grafika wstawiona na stronę zostaje umieszczona w **lewym górnym rogu ekranu**. Do zmiany standardowych ustawień i pozycjonowania grafiki w dowolnym miejscu służy atrybut background-position.

Center, left,right,top,bottom, dlugosc px (od lewego marginesu)
Left top, left bottom, right top, …

Tło i kolor w CSS

warto przećwiczyć

Ćw. 1

Utwórz baner reklamowy wykorzystując grafikę oraz elementy p,div

Podpowiedź:
Color, background-color, background-image,
background-repeat, background-attachment,background-position

Pozycjonowanie

Pozwala zdefiniować **położenie elementów na stronie**. Można je rozmieszczać nie tylko **względem brzegów** strony, ale również **względem jej poszczególnych elementów**. Można umieścić elementy tak, by wybrany element **przykrywał inny**.

Po pozycjonowania elementów służy atrybut *position*.

Pozycjonowanie

Sposoby pozycjonowania elementów:

- *static*
- *relative*
- *absolute*
- *fixed*

Pozycjonowanie

Pozycjonowanie *static* jest domyślnym ustawieniem elementu.

Przykład:
h1 {position: relative; left: 50%; }
<h1 style=„position:static”>text </h1>

Pozycjonowanie

Pozycjonowanie *relative (względne)* pozwala przesunąć element **względem położenia pierwotnego** (czyli takiego, w którym nie używamy pozycjonowania).

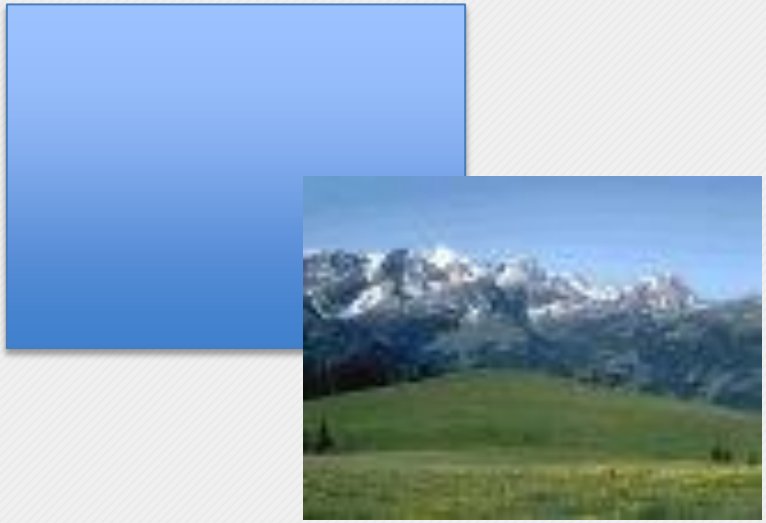
Jako parametry należy **podać krawędzie** *left*, *right*, *top*, *bottom*. Wartość tych parametrów oznacza konkretną odległość.

Można **łączyć parametry oraz stosować wartości ujemne**. Pamiętajmy: left i top mają **priorytet**.

Pozycjonowanie

Przykład:

```
img { position: relative; left: 50px; top: 45px; }
```



Pozycjonowanie

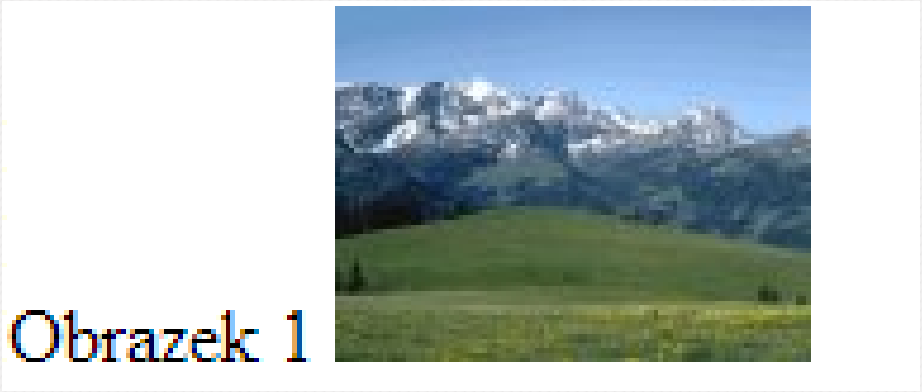
Pozycjonowanie *absolute (absolutne)* pozwala przesunąć element **względem bloku obejmującego**. Blokiem takim najczęściej jest okno przeglądarki lub ramka.

Jeżeli element, który pozycjonujemy znajduje się **wewnątrz innego elementu**, który wcześniej został poddany **pozycjonowaniu** (nie *static*!), to położenie jest obliczane względem pozycji tego **elementu nadrzędnego**, a nie okna przeglądarki.

Pozycjonowanie

Przykład:

```
<div class="image">
    <span>Obrazek 1</span>
    <img src="http://www.kurshtml.edu.pl/pliki/obrazek.jpg"/>
</div>
```



Pozycjonowanie

Przykład:

```
.image { position: relative; height:75px;width:100px; }
div.image span {
    color: white;
    z-index:1;
    position:absolute;
    left:0px;
    top:50px;
    height:1em;
    width:100%
}
```

Pozycjonowanie

Przykład:

Pozycjonowanie

Przykład:

```
<div class="elem">
    <span class="label start">IMG START</span>
    <span class="label stop">IMG END</span>
    <img src="http://learnlayout.com/images/ilta.png"/>
</div>
```

Pozycjonowanie

Przykład:

IMG START  IMG END

Pozycjonowanie

Przykład:

```
div.elem { position: relative;
    max-width:400px;
    border: 2px solid red;
    padding:20px 20px;
    text-align: center;
}
.label { position: absolute; line-height: 1em;background-color: red;color: white; }
.start { left:0px;top:0px; }
.stop { bottom:0px;right:0px;}
```

Pozycjonowanie

Przykład:

Pozycjonowanie

Pozycjonowanie *fixed (ustalone)* działa bardzo podobnie jak absolutne, z tą różnicą że pozycja elementu obliczana jest **zawsze** względem **okna przeglądarki**. Dodatkowo element taki jest nieruchomy podczas przewijania strony, przez co jest cały czas widoczny na ekranie, dokładnie w tym samym miejscu.

Model blokowy CSS

Wszystkie elementy projektowane w języku HTML można traktować jako bloki odpowiednio sformatowane i rozmieszczone na stronie.

Każdy blok posiada:

- margines zewnętrzny (margin)
- obramowanie (border)
- odstęp pomiędzy obramowaniem i zawartością (padding)
- zawartość elementu (content)

Model blokowy CSS

Obramowanie to ramka narysowana wokół elementu. Może być wykorzystywane do dekoracji elementu lub do oddzielenia go od innych elementów.

Szerokość obramowania: border-width

1 – dla wszystkich
2 – dla poziomych i pionowych krawędzi
3 – górna, pionowe, dolna
4 – każda osobno

Model blokowy CSS

Szerokość krawędzi może być definiowana w dowolnych jednostkach lub za pomocą wyrażeń:

- thin
- medium
- thick

Możliwe jest również definiowane szerokości pojedynczych krawędzi.
border-(top|left|right|bottom)-width

Model blokowy CSS

Do definiowania **stylu obramowania** służy atrybut border-style.

Dostępne style:

- none
- hidden
- dotted
- dashed
- solid
- double
- groove
- ridge
- inset
- outset

Model blokowy CSS

Kolor obramowania definiowany jest za pomocą border-color.

Przykład:

```
border-top-color: red;
border: red;
```

Model blokowy CSS

Margines zewnętrzny określa przestrzeń wokół definiowanego elementu. Przestrzeń ta oddziela element od innych elementów strony.

Do każdego elementu można **zdefiniować odległość** od góry, dołu, z lewej oraz prawej strony.

Przykład:
margin-left,margin-right, margin-bottom, margin-top

Model blokowy CSS

Margines wewnętrzny to przestrzeń zawarta pomiędzy zawartością elementu a obramowaniem. Definicja jest bardzo podobna do poprzednich.

Służy do tego atrybut *margin-padding*.

Model blokowy CSS

Obramowanie wokół elementu może być tworzone poprzez zdefiniowane obrysu.

Obrys różni się od obramowania tym, że :

- nie zajmuje miejsca
- jest tworzony na wierzchu elementu
- dla wszystkich krawędzi jednocześnie

div { outline-color: red; outline-width: 10px; outline-style: dashed }

Model blokowy CSS

W modelu blokowym można ustalać dokładne **rozmiary różnych elementów** np. akapit, obrazek, blok div.

width
height
min-width
max-width

Model blokowy CSS

Jeśli zawartość elementu nie mieści się w rozmiarach podanych za pomocą atrybutów width i height, możliwe jest :

- ukrycie niemieszczącej się zawartości
- powiększenie rozmiarów elementu
- wyświetlenie suwaków

Mechanizm ten nazywa się przepelnienie.

Model blokowy CSS

Aby ustawić przepełnienie należy skorzystać z selektora overflow. Ma on dostępne trzy opcje:

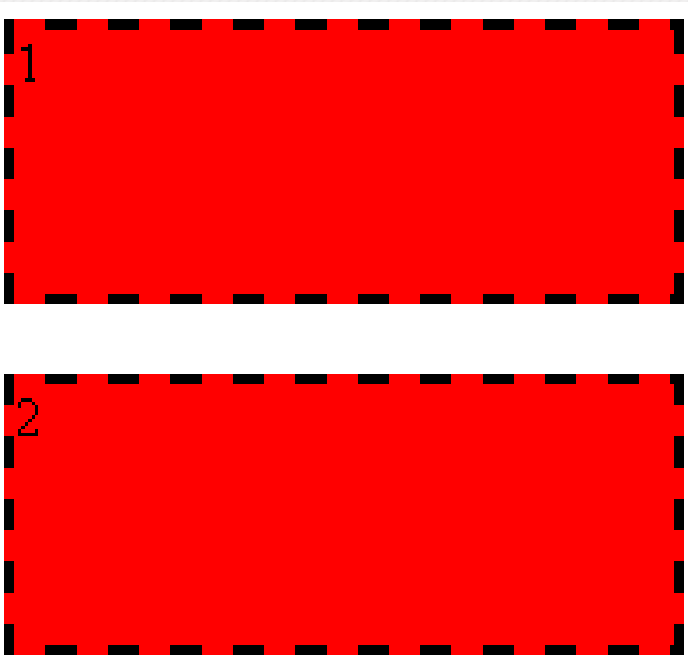
- visible – pokazywana jest cala zawartośc elementu
- hidden – niemieszcząca się zawartość jest ukryta
- scroll – wyswietlaja się suwaki
- auto jeżeli jest to konieczne, suwaki zostają wyświetlone

Model blokowy CSS

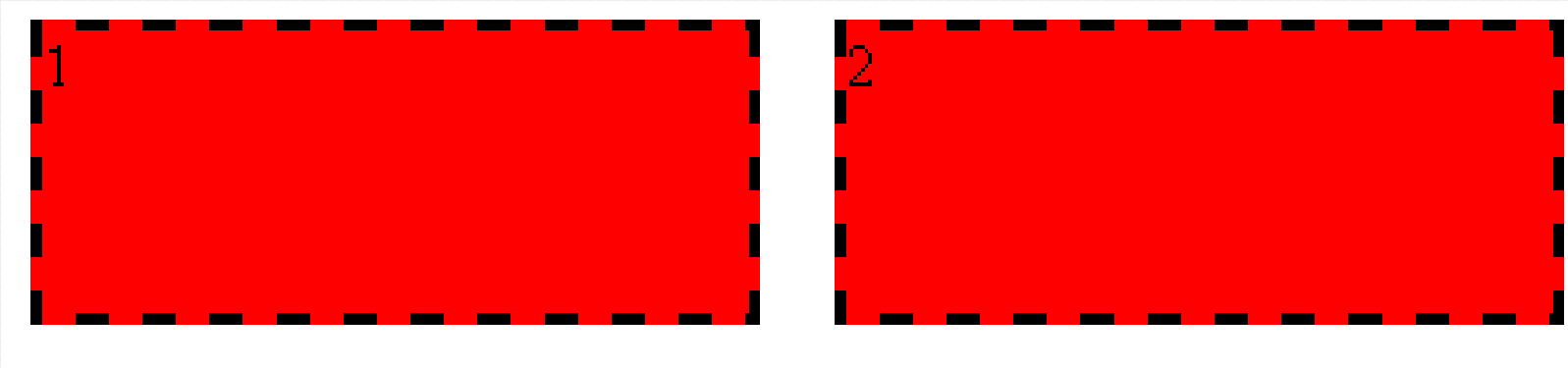
warto przećwiczyć

Pływanie elementów

Pływanie elementów



Pływanie elementów



Pływanie elementów

Aby wyłączyć „pływanie” elementów należy skorzystać z atrybutu clear.

left, right, both, none

Wyświetlanie elementów

Zdefiniowane w kodzie elementy są wyświetlane na stronie w sposób domyślny. Jeżeli chcemy zrezygnować z takiego pokazywania elementów, możemy zdefiniować atrybut display.

Parametry display:

- block
- inline
- list-item
- inline-block
- none
- run-in

Układy

Rodzaje układów:

- 2 kolumnowe
- 3 kolumnowe

Względne oraz bezwzględne.

Układy

Prosty układ 2 kolumnowy.

```
<div class="column one">...</div>
<div class="column two">...</div>
<div class="footer">...</div>
```

```
.column {float: right; width: 50%;}
.column {float: right; width: 30%; margin: 0 10%;}
.footer {clear: both;}
```

Układy

Prosty układ 3 kolumnowy.

```
<div class="column one">...</div>
<div class="column two">...</div>
<div class="column three">...</div>
<div class="footer">...</div>
```

```
.column {width: 20%; margin: 0 5%; float: left;}
.two {width: 30%;}
.footer {clear: both;}
```

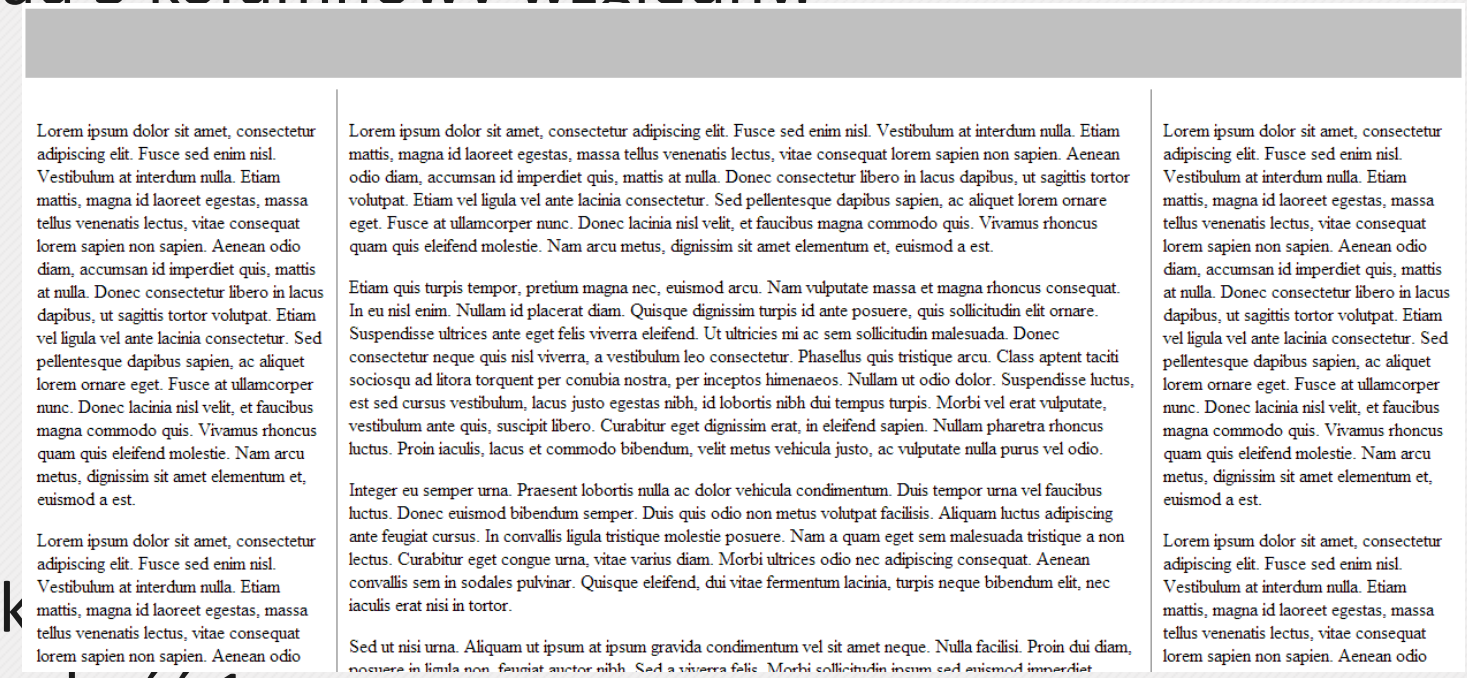
Układy

Ulepszanie układu 3 kolumnowego.

```
.column {width: 20%; margin: 0 2%; padding: 0 2%; float: left;}
.two {width: 30%; border: 1px solid grey; border-width: 0 1px;}
```

Układy

Układ 3 kolumnowy względny.



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce sed enim nisl. Vestibulum at interdum nulla. Etiam mattis, magna id laoreet egestas, massa tellus venenatis lectus, vitae consequat lorem sapien non sapien. Aenean odio diam, accumsan id imperdiet quis, mattis at nulla. Donec consectetur libero in lacus dapibus, ut sagittis tortor volutpat. Etiam vel ligula vel ante lacinia consectetur. Sed pellentesque dapibus sapien, ac aliquet lorem ornare eget. Fusce at ullamcorper nunc. Donec lacinia nisl velit, et faucibus magna commodo quis. Vivamus rhoncus quam quis eleifend molestie. Nam arcu metus, dignissim sit amet elementum et, euismod a est.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce sed enim nisl. Vestibulum at interdum nulla. Etiam mattis, magna id laoreet egestas, massa tellus venenatis lectus, vitae consequat lorem sapien non sapien. Aenean odio

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce sed enim nisl. Vestibulum at interdum nulla. Etiam mattis, magna id laoreet egestas, massa tellus venenatis lectus, vitae consequat lorem sapien non sapien. Aenean odio diam, accumsan id imperdiet quis, mattis at nulla. Donec consectetur libero in lacus dapibus, ut sagittis tortor volutpat. Etiam vel ligula vel ante lacinia consectetur. Sed pellentesque dapibus sapien, ac aliquet lorem ornare eget. Fusce at ullamcorper nunc. Donec lacinia nisl velit, et faucibus magna commodo quis. Vivamus rhoncus quam quis eleifend molestie. Nam arcu metus, dignissim sit amet elementum et, euismod a est.

Etiam quis turpis tempor, pretium magna nec, euismod arcu. Nam vulputate massa et magna rhoncus consequat. In eu nisl enim. Nullam id placerat diam. Quisque dignissim turpis id ante posuere, quis sollicitudin elit ornare. Suspendisse ultrices ante eget felis viverra eleifend. Ut ultricies mi ac sem sollicitudin malesuada. Donec consectetur neque quis nisl viverra, a vestibulum leo consectetur. Phasellus quis tristique arcu. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Nullam ut odio dolor. Suspendisse luctus, est sed cursus vestibulum, lacus justo egestas nibh, id lobortis nibh dui tempus turpis. Morbi vel erat vulputate, vestibulum ante quis, suscipit libero. Curabitur eget dignissim erat, in eleifend sapien. Nullam pharetra rhoncus luctus. Proin iaculis, lacus et commodo bibendum, velit metus vehicula justo, ac vulputate nulla purus vel odio.

Integer eu semper urna. Praesent lobortis nulla ac dolor vehicula condimentum. Duis tempor urna vel faucibus luctus. Donec euismod bibendum semper. Duis quis odio non metus volutpat facilisis. Aliquam luctus adipiscing ante feugiat cursus. In convallis ligula tristique molestie posuere. Nam a quam eget sem malesuada tristique a non lectus. Curabitur eget congue urna, vitae varius diam. Morbi ultrices odio nec adipiscing consequat. Aenean convallis sem in sodales pulvinar. Quisque eleifend, dui vitae fermentum lacinia, turpis neque bibendum elit, nec iaculis erat nisi in tortor.

Sed ut nisi urna. Aliquam ut ipsum at ipsum gravida condimentum vel sit amet neque. Nulla facilisi. Proin dui diam,

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce sed enim nisl. Vestibulum at interdum nulla. Etiam mattis, magna id laoreet egestas, massa tellus venenatis lectus, vitae consequat lorem sapien non sapien. Aenean odio diam, accumsan id imperdiet quis, mattis at nulla. Donec consectetur libero in lacus dapibus, ut sagittis tortor volutpat. Etiam vel ligula vel ante lacinia consectetur. Sed pellentesque dapibus sapien, ac aliquet lorem ornare eget. Fusce at ullamcorper nunc. Donec lacinia nisl velit, et faucibus magna commodo quis. Vivamus rhoncus quam quis eleifend molestie. Nam arcu metus, dignissim sit amet elementum et, euismod a est.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce sed enim nisl. Vestibulum at interdum nulla. Etiam mattis, magna id laoreet egestas, massa tellus venenatis lectus, vitae consequat lorem sapien non sapien. Aenean odio

Wyk

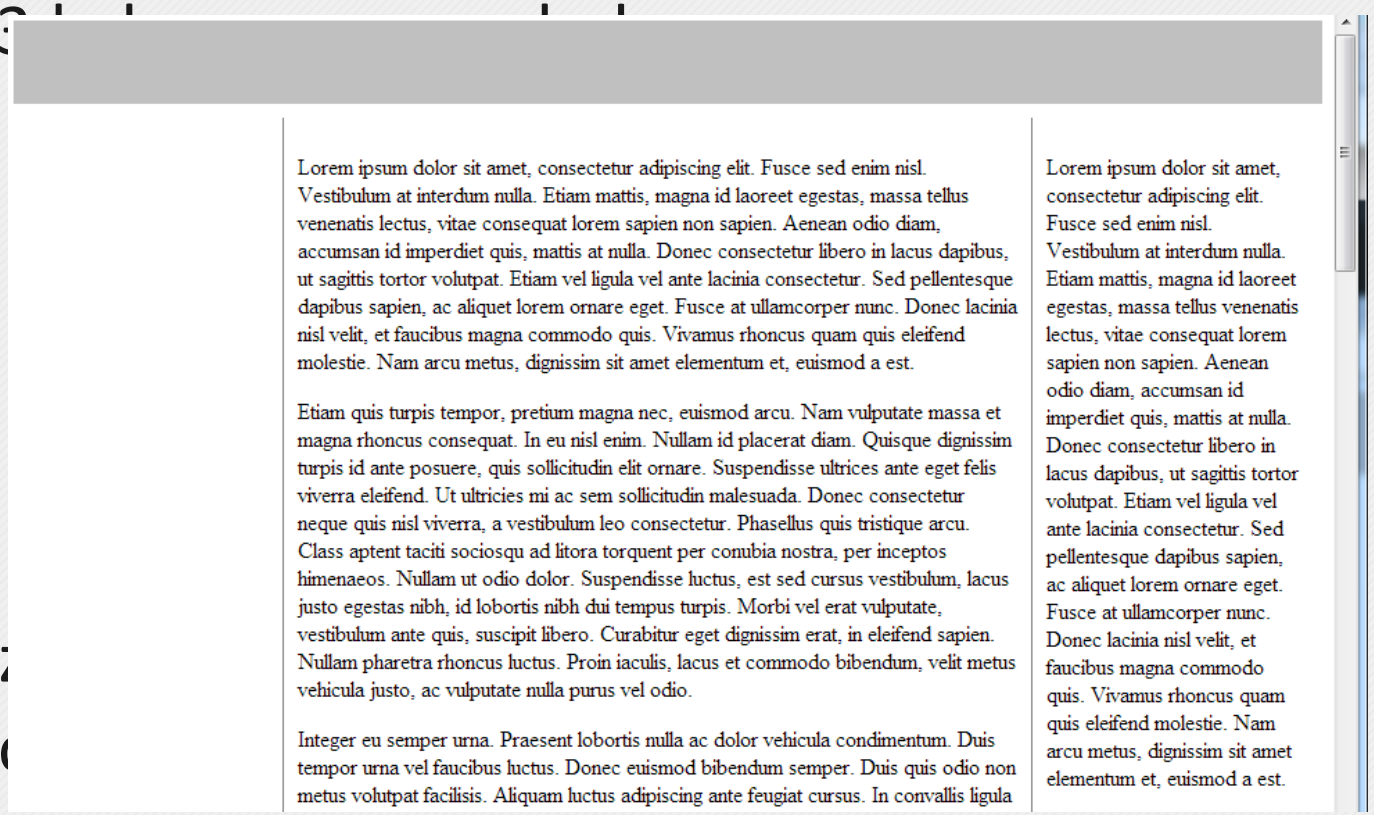
Szerokość 1

Układy

Układ 3

Wykorz

Szeroko



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce sed enim nisl. Vestibulum at interdum nulla. Etiam mattis, magna id laoreet egestas, massa tellus venenatis lectus, vitae consequat lorem sapien non sapien. Aenean odio diam, accumsan id imperdiet quis, mattis at nulla. Donec consectetur libero in lacus dapibus, ut sagittis tortor volutpat. Etiam vel ligula vel ante lacinia consectetur. Sed pellentesque dapibus sapien, ac aliquet lorem ornare eget. Fusce at ullamcorper nunc. Donec lacinia nisl velit, et faucibus magna commodo quis. Vivamus rhoncus quam quis eleifend molestie. Nam arcu metus, dignissim sit amet elementum et, euismod a est.

Etiam quis turpis tempor, pretium magna nec, euismod arcu. Nam vulputate massa et magna rhoncus consequat. In eu nisl enim. Nullam id placerat diam. Quisque dignissim turpis id ante posuere, quis sollicitudin elit ornare. Suspendisse ultrices ante eget felis viverra eleifend. Ut ultricies mi ac sem sollicitudin malesuada. Donec consectetur neque quis nisl viverra, a vestibulum leo consectetur. Phasellus quis tristique arcu. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Nullam ut odio dolor. Suspendisse luctus, est sed cursus vestibulum, lacus justo egestas nibh, id lobortis nibh dui tempus turpis. Morbi vel erat vulputate, vestibulum ante quis, suscipit libero. Curabitur eget dignissim erat, in eleifend sapien. Nullam pharetra rhoncus luctus. Proin iaculis, lacus et commodo bibendum, velit metus vehicula justo, ac vulputate nulla purus vel odio.

Integer eu semper urna. Praesent lobortis nulla ac dolor vehicula condimentum. Duis tempor urna vel faucibus luctus. Donec euismod bibendum semper. Duis quis odio non metus volutpat facilisis. Aliquam luctus adipiscing ante feugiat cursus. In convallis ligula

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce sed enim nisl. Vestibulum at interdum nulla. Etiam mattis, magna id laoreet egestas, massa tellus venenatis lectus, vitae consequat lorem sapien non sapien. Aenean odio diam, accumsan id imperdiet quis, mattis at nulla. Donec consectetur libero in lacus dapibus, ut sagittis tortor volutpat. Etiam vel ligula vel ante lacinia consectetur. Sed pellentesque dapibus sapien, ac aliquet lorem ornare eget. Fusce at ullamcorper nunc. Donec lacinia nisl velit, et faucibus magna commodo quis. Vivamus rhoncus quam quis eleifend molestie. Nam arcu metus, dignissim sit amet elementum et, euismod a est.

Witryna Internetowa

warto zapamiętać

- **definicja witryna, portal, wortal, blog**
- **pierwsza witryna Internetowa**

Pytanie:
www.nova.edu.pl – portal czy wortal?

Projekt witryny

warto przećwiczyć

PAINT – aplikacja firmy Microsoft do podstawowej obróbki grafiki rastrowej.

Ćwiczenie:
Wykorzystując program paint wykonaj projekt (szkic) strony internetowej.