ЛАБОРАТОРНА РОБОТА №3.
Ознайомлення з мовою розмічування HTML

ДОДАТКОВЕ ЗАВДАННЯ

Додати інформацію про стилі (оформлення) до web-сторінок, створених у першій частині лабораторної роботи. Таблиця стилів повинна міститися у зовнішньому файлі та проходити перевірку валідатором (онлайн: http://jigsaw.w3.org/css-validator/). Також дуже бажано перевірити таблицю стилів через http://csslint.net/ .

КАСКАДНІ ТАБЛИЦІ СТИЛІВ (CSS)

ЗАГАЛЬНА ІДЕОЛОГІЯ

Головна перевага таблиць стилів у тому, що вони дозволяють розділити зміст документа і його фізичне представлення. Основа стилів – розмічування, наприклад, таке, яке існує в мові розмічування HTML. Без елементів, до яких стилі прив'язані, вони нічого не означають. Призначення таблиць стилів – визначити оформлення того чи іншого елемента чи набору елементів.

Таблиця стилів є просто набором правил, які складаються з двох частин: **селектора** і його **визначення**. Селектором може бути будь-який тег HTML, клас або ідентифікатор (можливі і набагато складніші випадки). Визначення складається також з двох частин: **властивості** та її **значення**, розділених двокрапкою. В описі стилю можуть використовуватись декілька правил, розділених крапкою з комою.

Загальний синтаксис: `селектор {властивість: значення}`

Приклад: `h1 {color:green; background-color:grey}`

Таблиця стилів може містити **коментарі**. ***Синтаксис:*** `/* це коментар */`

При оголошенні стилю можна також користуватися деякими скороченнями і групами правил. Зокрема, для зменшення розмірів таблиці стилів можна групувати різні селектори, якщо для них задаються однакові правила: `h1, h2, h3 {font-weight: bold}`

Синтаксис правил CSS не чутливий до регістру, проте чутливий до правильності назв селекторів, властивостей і значень. Будь-яка граматична помилка призводить до того, що відповідне правило взагалі ігнорується браузером.

Існують 3 основні способи додавання інформації про стилі в документ HTML:

1) використання зовнішньої таблиці стилів, яка або **імпортується**, або **зв'язується** з документом;
2) **вкладення** інформації про стилі, які діють в даному документі, в частину **<head>**;
3) **вбудовування** інформації про стиль безпосередньо в тому місці, де вона має бути застосована (за допомогою атрибута **style**, визначеного майже для будь-якого тега).

Зв'язування таблиці стилів з документом.

Зовнішня таблиця стилів – це звичайний текстовий файл, який містить визначення стилів для тегів HTML або класів. HTML-файл, який буде використовувати цю таблицю, посилається на неї за допомогою тега **<link>** в частині **<head>**.

*Приклад**:***

```
<link rel="stylesheet" href="mystyle.css" media="screen" type="text/css" />
```

Посилання на таблицю стилів можуть бути як локальними, так і глобальними. Значення атрибута **type="text/css"** означає відповідний тип документа.

Вкладення та імпортування таблиць стилів

Вкласти таблицю стилів в документ HTML - означає вписати правила стилів безпосередньо в сам документ. У цьому випадку правила таблиці стилів розміщуються в стильовому блоці **<style>** частини **<head>:**

```
<head>
<title>Назва </title>
<style type="text/css">
<!--
```

```
    Зміст правил CSS
  -->
  </style>
  </head>
```

Зверніть увагу на коментування текста правил в блоці **<style>.** Це необхідно для того, щоб браузери, які не розуміють цього тега, не вивели визначення стилів на екран або якось інакше небажано не відреагували на нього. Всередині блока **<style>** діють правила синтаксису CSS.

Можна також розмістити правила стилів в окремому файлі, а потім **імпортувати** цей файл (в такому випадку визначення пріоритетів правил дещо змінюється порівняно з попереднім випадком):

```
<head>
  <title>Назва </title>
  <style type=”text/css”>
  <!--
    @import: url(mystyle.css);
  -->
  </style>
</head>
```

Вбудовані стилі

Крім застосування стилів до всього документа, можна додавати інформацію про стиль до конкретного елемента, наприклад:

```
<h1 style=”font-size:48px; color:green; text-align:left”>Заголовок</h1>
```

ІДЕНТИФІКАТОРИ, КЛАСИ ТА ПСЕВДОКЛАСИ

Додавання інформації про стилі до конкретного елемента за допомогою атрибута **style** – це найпростіший, але не найкращий спосіб, оскільки він зводить майже нанівець розділення структури і представлення документа. Якщо необхідно надати деякий стиль окремому елементу або групі елементів, існують і інші способи.

Майже кожний елемент має атрибути **id** і **class**. За допомогою атрибута **id** відповідному елементу можна присвоїти унікальне ім’я, а потім його використовувати (зокрема, як селектор в таблиці стилів):

Частина таблиці стилів: `#par11 {color:green}`

Елемент HTML: `<p id=”#par11”>Зелений текст</p>`

Атрибут **class** визначає назву класу, до якого належить даний елемент (цей атрибут не визначений для **<html>, <head>, <body>,
**). Декілька елементів можуть бути членами одного класу. Ім’я класу вказується в селекторі правила після імені тега і відокремлюється крапкою:

Частина таблиці стилів: `h1.green {color:green}`

Елемент HTML: `<h1 class=”green”>Зелений заголовок</h1>`

Елементами одного класу можуть бути навіть різнотипні елементи. В такому випадку правило таблиці стилів не утримує імені тега: `.green {color:green}`

Для елемента посилання **<a>** визначені також псевдокласи:

```
a:link {color:red} /* звичайне невідвідане посилання */
a:visited {color:blue} /* відвідане посилання */
a:active {color:green} /* активне посилання */
a:hover{color:yellow} /* посилання, над яким знаходиться вказівник миші */
```

У стандарті існують також псевдокласи для інших елементів, наприклад: `p:first-letter`, але не всі браузери можуть з ними працювати (це стосується і псевдокласів `a:active` і `a:hover`).

Селектори нащадків

В HTML деякі теги можуть утримувати в собі інші. У подібних випадках **вкладений** елемент **успадковує** правила форматування батьківського елемента, **якщо для нього не визначені власні**. Якщо ж

треба для вкладеного елемента визначити властивості особливим чином саме у тому випадку, коли він є вкладеним, використовують **селектори нащадків**. Селектор нащадка складається з декількох (як мінімум, двох) простих, розділених пробілами і перелічених саме у тому порядку, у якому вони повинні бути розміщені в тексті для того, щоб правило спрацювало:

Частина таблиці стилів: `h1 em {color:green}`

Елемент HTML: `<h1><em>Дуже важливе</em> правило</h1>`

ПРИНЦИП ПРІОРИТЕТНОСТІ

У досить грубому наближенні основна ідея цього принципу виражається так: чим ближче до елемента розташовано правило, тим більший пріоритет воно має.

Рівень браузера:
Стилі браузера за замовчанням < стилі, задані користувачем < стилі, задані автором документа.

Рівень способу додавання правил:
Зв'язані таблиці < імпортовані таблиці < включені правила < вбудовані правила.

Всередині однієї таблиці можливі конфлікти розв'язуться за наступними правилами.

Рівень задання правила:
Значення за замовчанням < успадковане значення < явно задане значення.

Специфічність селектора:
Простий < нащадок < клас < клас нащадка < ідентифікатор.

Пріоритет правила можна збільшити (приблизно на 1 рівень) за допомогою значення **important**:

```
h1 {color: red !important;}
```

Крім цього, правила сортуються за порядком їх згадування в таблиці: задане **пізніше** має **вищий** пріоритет.

Простий висновок з усього згаданого вище: найвищий пріоритет мають вбудовані стилі, і тому їх слід використовувати у тих випадках, коли треба «перекрити» для конкретного елемента правила загальної таблиці стилів. Але треба також зважати на те, що використання вбудованих стилів небажане без явної потреби, оскільки воно порушує правило відокремлення структури від зовнішнього представлення.

ОДИНИЦІ ВИМІРЮВАННЯ ТА ІНШІ СИНТАКСИЧНІ ПРАВИЛА ЗАДАННЯ ЗНАЧЕНЬ

Одиниці довжини можуть бути абсолютні і відносні.

Відносні (в переважній більшості випадків краще використовувати саме їх):
`em` (висота шрифта елемента); `ex` (висота маленької літери x); `px` (піксел); `%` (відсоток).

Абсолютні: `in` (дюйм = 2.54 см); `cm` (см); `mm` (мм); `pt` (пункт = 1/72 дюйма); `pc` (піка = 12 пунктів).

Кольори можуть задаватися:

- стандартними назвами (див., наприклад, http://www.w3schools.com/tags/ref_colornames.asp);
- у шістнадцятковому форматі `#rrggbb` (тут `r, g, b` – шістнадцяткові цифри (0-f));
- у десятковому форматі `(red, green, blue)` (тут `red, green, blue` – десяткові числа від 0 до 255).

Для задання **посилань** використовується синтаксис url (…), *наприклад*:

```
url (http://www.smth.com/logo.gif)
```

Як і в інших випадках, посилання можуть бути глобальними і локальними. Зверніть увагу, що для зв'язаних і імпортованих таблиць стилів **локальні** посилання, вказані в них, відраховуються **від адреси файлу з таблицею стилів**, а не від адреси документа, для якого вони призначені!

ВИЗНАЧЕННЯ ВЛАСТИВОСТЕЙ ЕЛЕМЕНТІВ

Властивості шрифтів

font-family – визначає сімейство шрифтів. Можна задавати список шрифтів, у такому разі останній елемент списку – назва сімейства. З урахуванням проблем з різними кириличними кодуваннями, цією властивістю слід користуватися з обережністю.

Сімейств шрифтів усього 5

- serif (приклад: Times New Roman);
- sans-serif (приклади: Arial, Verdana);
- cursive (приклад: Comic Sans MS);
- fantasy (приклад: Monotype Corsiva);
- monospace (приклад: Courier New).

Зверніть увагу, що якщо назва шрифту складається з 2 або більше слів, то її треба писати в лапках: `font-family: “Times New Roman”, serif;`

font-size – задає абсолютний (фізичний) або відносний розмір поточного шрифту. Абсолютні значення задаються як: xx-small, x-small, small, medium, lagre, x-large, xx-large (і в цьому випадку залежать від масштабу шрифтів браузера) або абсолютним значенням довжини, наприклад, 12 pt. Відносні задаються як: larger, smaller, або у відносних значеннях довжини, наприклад 120%, 1.2 em (і у цьому випадку залежать від розміру шрифту батьківського елемента);

font-style – визначає стиль шрифту з обраного сімейства: нормальний (normal), курсивний (*italic*), або похилий (oblique), який генерується з нормального невеликим нахиленням символів.

font-variant – можливі значення: normal і SMALL-CAPS.

font-weight – визначає жирність шрифта із заданого сімейства. Може задаватися числами `100, 200, ..., 900` або ключовими словами: `extra-light, light, demi-light, medium (normal), demi-bold, bold, extra-bold.` Гарантовано підтримуються лише `normal (400), bold (700)`. Може визначатися відносно: `bolder, lighter`.

Можна задати всі властивості шрифта **одночасно**. *Синтаксис*:
`font: font-style font-variant font-weight font-size/line-height font-family`

Приклад: `p {font: italic small-caps 600 18pt/24pt “Arial, sans-serif”}`

Якщо якесь значення не задане, використовується відповідне значення за замовчанням. Усі властивості шрифтів успадковуються вкладеними елементами і можуть застосовуватися до усіх елементів HTML.

Колір і фон

color – визначає колір тексту елемента, наприклад: `p {color: green};`

background-color – задає колір фону. Значення за замовчанням: `transparent` (прозорий);

background-image – визначає фонове зображення. Задається як абсолютне або відносне посилання на графічний файл. Краще не задавати без фонового кольору. Значення за замовчанням: none (немає зображення).

Приклад:
`body {background-color: steelblue; background-image:url(images/back.gif)}`

background-repeat – визначає спосіб повторюваності фонового зображення. Можливі значення: repeat (повторювати в обох напрямках, за замовчанням), repeat-x (тільки в горизонтальному напрямку), repeat-y(тільки у вертикальному напрямку), no-repeat (не повторювати).

background-attachment – визначає, чи буде фон фіксованим. Можливі значення: fixed (фіксований), scroll (прокручування, за замовчанням).

background-position – визначає початкове положення фонового зображення (координати лівого верхнього кута) відносно лівого верхнього кута блока зображення. Визначено тільки для елементів рівня блоку! Може задаватися у відсотках, абсолютних одиницях довжини або ключовими словами `left (відповідає 0%), right (100%), center (50%), top (0%), bottom (100%)`. Значення за замовчанням: `0% 0%` (або `top left`). Якщо задане лише одне з ключових слів (`left, right`, або `top, bottom`), то друге вважається заданим як `center`.

Можна задати всі властивості фона **одночасно**. *Приклад:*
`body {background: steelblue url(images/back.gif) center;}`

Форматування тексту

letter-spacing – визначає відстань між літерами. Задається в одиницях довжини. Це значення **додається** до встановленого за замовчанням! *Приклад:* `p {letter-spacing: 0.5 em};`

text-transform – визначає регістр літер текста. Можливі значення: `none` (за замовчанням), `capitalize`, `uppercase`, `lowercase`;

text-decoration – Можливі значення: `underline` (риска знизу), `overline` (риска зверху), `line-through` (перекреслений), `blink` (блимає), `none` (значення за замовчанням);

text-align – задає вирівнювання тексту елемента рівня відносно цього блока. Можливі значення: `left` (по лівому краю, значення за замовчанням для більшості елементів), `center` (по центру), `right` (по правому краю), `justify`(по ширині);

text-indent – задає абзацний відступ в елементах рівня блока в одиницях довжини (якщо у відсотках – від ширини блока). Можливі від'ємні значення. *Приклад:* `p{text-indent: 5%};`

vertical-align – задає вирівнювання елемента по вертикалі відносно батьківського елемента. Можливі значення: `baseline` (по базовій лінії, значення за замовчанням), `middle` (по середині), `top` (по верхньому краю найвищого елемента), `bottom` (по нижньому краю найнижчого елемента), `text-top` (по верхньому краю шрифту), `text-bottom` (по нижньому краю шрифту), `sub` (нижній індекс), `super` (верхній індекс). Може задаватися також у одиницях довжини або відсотках від висоти рядка (`line-height`) самого елемента – тоді означає зміщення базової лінії елемента відносно базової лінії батьківського елемента.

line-height – задає величину відстані між базовими лініями рядків (інтерлін'яж) – у відсотках від висоти шрифту, одиницях довжини або числом (тоді 1 відповідає висоті шрифту). Значення за замовчанням: `normal (100%).` *Приклад:* `p{line-height: 200%}`

Форматування блоків

Будь-який елемент рівня блока має **поле** (margin, за замовчанням прозоре), **рамку** (за замовчанням вона не відображається), **відступ** (padding, за замовчанням кольору фона самого елемента) і, власне, простір для тексту або іншого вмісту блока.

Властивості форматування **полів** встановлюють ширину відповідних полів: **margin-top** (верхнього), **margin-right** (правого), **margin-bottom** (нижнього), **margin-left** (лівого). Значення задаються в одиницях довжини або відсотках від ширини батьківського елемента (за замовчанням - 0). Властивість **margin** дозволяє задати значення всіх чотирьох, причому можна задавати не усі значення – тоді решта визначаються за наступним **правилом**: порядок перелічення значень - **верхнє, праве, нижнє, ліве**; незадане значення береться з протилежного боку.

Приклад:

```
margin: 1px 2px 3px 4px   /*top=1px, right=2px, bottom=3px, left=4px */
margin: 1px 2px 3px       /*top=1px, right=2px, bottom=3px, left=2px */
margin: 1px 2px           /*top=1px, right=2px, bottom=1px, left=2px */
margin: 1px               /*top=1px, right=1px, bottom=1px, left=1px */
```

Властивості форматування відступів абсолютно аналогічні до описаних вище для полів, треба лише слово margin поміняти на слово **padding**.

Для форматування **рамок** існує більше можливостей:

border-style – визначає вигляд рамки. Можливі значення: `none` (рамки немає, значення за замовчанням), `dotted` (з крапок), `dashed` (пунктирна), `solid` (суцільна), `double` (подвійна), `groove` (вдавлена), `ridge` (опукла), `inset` (елемент вдавлений), `outset` (елемент опуклий). Властивість **border-style** визначає рамку з усіх боків елемента (аналогічно полям і відступам можна задавати різні значення для 4 сторін – правило те саме). Властивості **border-top-style, border-right-style, border-bottom-style, border-left-style** можуть задавати вигляд окремо для відповідних частин рамки;

border-width – визначає ширину рамки в одницях довжини або ключовими словами `thin, medium` (значення за замовчанням), `thick`. Усі можливості аналогічні властивості **border-style,** в тому числі і існування окремих **border-top-width, border-right-width, border-bottom-width, border-left-width.**

border-color – визначає колір рамки. Усі можливості аналогічні до властивостей **border-style** і **border-width**.

Значення властивостей для рамок можна групувати як по змісту властивості (колір, ширина, стиль), так і по розташуванню рамки – це дозволяють зробити властивості **`border-top, border-right, border-bottom, border-left.`** *Приклад:* `p {border-top: double thin red}`

Властивості для рамок з усіх боків одночасно можна задати за допомогою властивості **`border.`**

`width` – задає ширину області вивода змісту елемента (внутрішню, без урахування полів, рамок і відступів) у одиницях довжини або відсотках від ширини батьківського елемента;

`height` – задає висоту області вивода змісту елемента в одиницях довжини. Краще задавати значення цієї властивості лише для зображень або їх аналогів;

`float` – вказує, з якого боку об'єкт може обтікатися текстом. Можливі значення: `left, right, none` (за замовчанням);

`clear` – вказує, який з боків блока не може бути сумісним з попереднім блоком (відключає обтікання по цьому боці). Можливі значення: `left, right, both, none` (за замовчанням).

Візуальне форматування

`position` – визначає спосіб позиціонування елемента на сторінці. Можливі значення: static (статичний, значення за замовчанням), absolute (абсолютний, дозволяє розмістити елемент на сторінці довільним чином), relative (відносний, зміщує елемент відносно його розташування за замовчанням).

Для зміщення абсолютно і відносно позиціонованих елементів використовуються властивості **`top`** і **`left`**, які визначають в пікселах зміщення вниз і праворуч лівого верхнього кута блока відображення елемента.

`z-index` – задає порядок перекривання абсолютно чи відносно позиціонованих елементів. Можливі значення: невід'ємні числа (елемент з меншим значенням перекривається елементом з більшим значенням) або auto (за замовчанням);

`clip` – дозволяє «обрізати» видиме зображення абсолютно позиціонованого елемента. Можливі значення: auto (за замовчанням) або **`rect ([top] [left] [bottom] [right])`**, де top, left, bottom і right визначають межі видимого зображення елемента відносно блока його відображення.
Приклад: `img {clip: rect(10 70 60 10)};`

`overflow` – визначає спосіб відображення елемента у випадку, коли розміри його зміста не відповідають розмірам блока відображення, встановленим значеннями властивостей **`width`** і **`height`**. Можливі значення: `visible` (виводить повністю), `hidden` (обрізає зайве), `scroll` (додає смуги прокручування), `auto` (за замовчанням);

`visibility` – керує відображенням елемента. Можливі значення: `visible` (відображує, значення за замовчанням), `hidden` (не відображує, але залишає вільним місце під елемент), `inherit` (успадковане від батьківського елемента);

`display` – дозволяє змінювати тип елемента з точки зору виводу на екран. Можливі значення: `block` (елемент рівня блока), `inline` (елемент рівня текста), `list` (елемент списка), `none` (не відображує елемент взагалі, місце під нього не резервується).

Відображення списків (впорядкованих і невпорядкованих)

`list-style-type` – визначає тип маркера списка. Можливі значення: `none; disk, circle, square; decimal` (арабські цифри), `lower-roman, upper-roman` (римські цифри), `lower-alpha, upper-alpha` (латинські літери);

`list-style-image` – задає графічне зображення, яке використовується як маркер списка. Можливі значення: none (за замовчанням) або посилання на файл `url (…)`;

`list-style-position` – визначає положення маркера. Можливі значення: `inside, outside` (за замовчанням);

Властивість **`list-style`** дозволяє згрупувати значення властивостей, перелічених вище.
Приклад: `ul {list-style: circle url(list.gif) inside};`