Лабораторна робота № 3

Тема. Розробка web-сторінок зі стильовим оформленням

Мета: отримання практичних навичок оформлення гіпертекстових документів з використанням каскадних таблиць стилів.

Короткі теоретичні відомості

CSS – це абревіатура від слів Cascading Style Sheets, що означає каскадні таблиці стилів. Стилем називають набір параметрів форматування, який застосовується до елементів документа, щоб змінити їх зовнішній вигляд. За допомогою стилів можна задавати колір, розмір тексту, фон, відступи та інші параметри для будь-якого тега. Загалом CSS дозволяє використовувати набагато більше можливостей для форматування, ніж звичайний HTML, чим розширює можливості дизайну і верстки веб-сторінок.

Розрізняють такі типи стилів.

1. Стиль браузера, тобто оформлення, яке за замовчуванням застосовується до елементів веб-сторінки браузером. Це оформлення можна побачити в разі «чистого» HTML, коли до документа не додано ніяких стилів.

2. Стиль автора, тобто стиль, який додає до документа його розробник.

3. Стиль користувача, тобто стиль, який може включити користувач сайту через налаштування браузера. Такий стиль має більш високий пріоритет і перевизначає вихідне оформлення документа.

Пріоритетність цих видів стилів така: найвищий пріоритет має стиль користувача, потім стиль автора, і останнім іде стиль браузера.

Синтаксисис описання стилю дуже простий:

селектор {властивість: значення;}

Якщо властивостей декілька, вони розділяються пробілами.

За способом вбудування таблиця стилів може бути:

1) зовнішньою, або пов’язаною (linked).

Це текстовий файл з розширенням .css, в якому знаходиться набір CSS-стилів елементів, без HTML-розмітки. Зовнішня таблиця стилів підключається до веб-сторінки за допомогою тега <*link*>, розташованого в розділі <*head*>.

Такі стилі працюють для всіх сторінок сайту.

До кожної веб-сторінки можна приєднати кілька таблиць стилів, додаючи послідовно кілька тегів <*link*>, вказавши в атрибуті тега *media* призначення цієї таблиці стилів. Атрибут *rel="stylesheet"* вказує тип посилання – на таблицю стилів. Наприклад:

<head>

<link rel = "stylesheet" href = "css / style.css">

<link rel = "stylesheet" href = "css / assets.css" media = "all">

</ head>

2) внутрішньою, або впровадженою (embedded).

Внутрішні стилі вбудовуються в розділ <*head*> HTML-документа і визначаються всередині тега <*style*>. Внутрішні стилі мають пріоритет над зовнішніми, але поступаються вбудованим стилям (заданим через атрибут*style*). Наприклад:

<head>

<style>

h1,

h2 {

color: red;

font-family: "Times New Roman", Georgia, Serif;

line-height: 1.3em;

}

</ style>

</ head>

3) Вбудованою (inline).

Вбудовані стилі описують в тілі HTML-документа, безпосередньо всередині тега елемента, за допомогою атрибута *style*. Наприклад:

<p style = "font-weight: bold; color: red;"> Виділений текст. </ p>

Ці стилі діють лише на той елемент, для якого задані.

Далі розглянемо види селекторів, що використовуються для визначення

стилів. Селекторами можуть бути елементи, їх класи та ідентифікатори, а також псевдокласи і псевдоелементи.

1. Універсальний селектор – символ *.

Відповідає будь-якому HTML-елементу. Наприклад:

** {margin: 0;}*

Такий стиль обнулить зовнішні відступи для всіх елементів сайту.

2. Селектор елемента.

Селектори елементів дозволяють форматувати всі елементи цього типу на всіх сторінках сайту. Наприклад, можна задати загальний стиль форматування всіх заголовків *h1*:

h1 {font-family: Times New Roman, cursive;}

3. Селектор класа.

Селектори класа дозволяють задавати стилі для одного і більше елементів з однаковим ім'ям класу, при цьому розміщені вони можуть бути в різних місцях сторінки або на різних сторінках сайту. Наприклад, щоб створити заголовок з класом *top_head*, необхідно додати атрибут *class* зі значенням *top_head* у відкриваючий тег *<h1>* і задати стиль для зазначеного класу. Стилі, створені за допомогою класу, можна застосовувати до інших елементів, необов'язково цього типу.

CSS:

<h1 class = "top_head"> Загаловок найвищого рівня </ h1>

HTML:

.top_head {

text-transform: uppercase;

color: red;

}

Якщо елемент має декілька атрибутів класу, їх значення перераховують через пробіл:

<h1 class = "top_head post-title"> Загаловок найвищого рівня </ h1>

4. Селектор ідентифікатора.

Селектор ідентифікатора дозволяє форматувати лише один конкретний елемент. Значення *id* має бути унікальним, на одній сторінці може зустрічатися тільки один раз і має складатися щонайменше з одного символу. Значення не

має містити пробілів, але інших обмежень для *id* немає, зокрема, ідентифікатори можуть складатися тільки з цифр, починатися з цифри, починатися з підкреслення, складатися тільки з розділових знаків тощо. Наприклад:

HTML:

* </ div>*

CSS:

#left_sidebar {

width: 2200px;

float: left;

}

5. Селектор нащадка.

Селектори нащадків застосовують стилі до елементів, які розташовані всередині елемента-контейнера. Наприклад, *ul li {text-transform: uppercase;}* – вибере всі елементи *li*, які є нащадками всіх елементів *ul*.

Якщо потрібно відформатувати нащадка лише певного елемента, для цього елемента потрібно задати стильовий клас. Наприклад:

p.first a {color: red;}

Цей стиль буде використано до всіх посилань-нащадків абзацу з класом *first.*

p .first a {color: red;}

Якщо після імені тега додати пробіл, то будуть стилізовані посилання,розташовані всередині будь-якого тега класу *first,* який є нащадком елемента

<p>.

.first a {color: red;}

Цей стиль буде використано до будь-якого посилання, розташованого усередині іншого елемента, позначеного класом *first.*

6. Дочірній селектор.

Дочірній елемент є прямим нащадком елемента, що його містить. У одного елемента може бути кілька дочірніх елементів, а батьківський елемент у кожного елемента може бути лише один.

Дочірній селектор дозволяє застосувати стилі, тільки якщо дочірній елемент розташований відразу за батьківським і між ними немає інших елементів, тобто дочірній елемент не вкладений в інший тег.

Наприклад:

p> strong

Цей селектор вибере всі елементи *strong*, які є дочірніми по відношенню до елемента *p*.

7. Сестринський селектор

Сестринські відносини виникають між елементами, що мають загального батька. Селектори сестринських елементів дозволяють вибрати елементи з групи елементів одного рівня. Наприклад:

h1 + p

Такий селектор вибере всі перші абзаци, що йдуть безпосередньо за будь-яким тегом <*h1*>, не зачіпаючи інші абзаци.

h1 ~ p

Такий селектор вибере всі абзаци, які є сестринськими по відношенню до будь-якого заголовку *h1* і йдуть відразу після нього.

8. Селектор атрибута.

Селектори атрибутів вибирають елементи на основі імені атрибута або значення атрибута:

1) *[Атрибут]* – вибирає всі елементи, що містять вказаний атрибут, наприклад: *[alt]* – всі елементи, для яких зацей атрибут *alt*;

2) *селектор [атрибут]* – вибирає елементи заданого типу, що містять вказаний атрибут, наприклад: *img [alt]* – тільки малюнки, для яких зацей атрибут *alt*;

3) *селектор [атрибут = "значення"]* – вибирає елементи заданого типу, що містять вказаний атрибут з вказаним конкретним значенням, наприклад:

img [title = "flower"] – вибирає всі малюнки, назва яких містить слово *flower*;

4) *селектор [атрибут ~ = "значення"]* – вибирає елементи, які частково містять задане значення, наприклад, якщо для елемента задано кілька класів через пробіл, то *p [class ~ = "feature"]* вибере абзаци, ім'я класу яких містить *feature*;

5) *селектор [атрибут | = "значення"]* – вибирає лише ті елементи, список значень атрибута яких починається із вказаного після знака «=» слова, наприклад: *p [class | = "feature"]* – вибере абзаци, ім'я класу яких *feature* або починається на *feature*;

6) *селектор [атрибут ^ = "значення"]* – вибирає ті елементи, значення атрибута яких починається із вказаного після знака «=» значення, наприклад:

a [href ^ = "http://"] – вибере всі посилання, що починаються на *http://*;

7) селектор [атрибут $ = "значення"] – вибирає елементи, значення атрибута яких закінчується вказаним значенням, наприклад *img [src $ = ".png"]* – вибере всі малюнки у форматі png;

8) *селектор [атрибут * = "значення"]* – вибирає елементи, значення атрибута яких містить в будь-якому місці вказане слово, наприклад: *a [href * = "book"]* – вибере всі посилання, назва яких містить *book*.

9. Селектор псевдокласу

Псевдокласи – це класи, що фактично не прикріплені до HTML-тегів. Вони дозволяють застосувати CSS-правила до елементів у разі настання події або при виконанні певного правила.

Псевдокласи характеризують елементи з такими властивостями:

– *: link* – не відвідуване посилання;

– *: visited* – відвідане посилання;

– *: hover* – будь-який елемент, над яким проводять курсором миші;

– *: focus* – інтерактивний елемент, до якого перейшли за допомогою клавіатури або активували за допомогою миші;

– *: active* – елемент, який був активізований користувачем;

– *: valid* – поля форми, вміст яких пройшов перевірку в браузері на

відповідність зазначеного типу даних;

– *: invalid* – поля форми, вміст яких не відповідає вказаним типам даних;

– *: enabled* – всі активні поля форм;

– *: disabled* – заблоковані поля форм, тобто такі, що знаходяться в неактивному стані;

– *: in-range* – поля форми, значення яких знаходяться в заданому діапазоні;

– *: out-of-range* – поля форми, значення яких виходять за встановлений діапазон;

– *: lang ()* – елементи з текстом зазначеною мовою;

– *: not (селектор)* – елементи, які не містять вказаний селектор – клас, ідентифікатор, назву або тип поля форми, наприклад *: not ([type = "submit"])*;

– *: target* – елемент із символом *#*, на який посилаються в документі;

– *: checked* – виділені (вибрані користувачем) елементи форми.

10. Селектор структурних псевдокласів.

Структурні псевдокласи відбирають дочірні елементи відповідно до параметрів, зазначених в круглих дужках:

– *: nth-child (odd)* – непарні дочірні елементи;

– *: nth-child (even)* – парні дочірні елементи;

– *: nth-child (3n)* – кожен третій елемент серед дочірніх;

– *: nth-child (3n + 2)* – вибирає кожен третій елемент, починаючи з другого дочірнього елемента *(+2)*;

– *: nth-child (n + 2)* – вибирає всі елементи, починаючи з другого;

– *: nth-child (3)* – вибирає третій дочірній елемент;

– *: nth-last-child ()* – в списку дочірніх елементів вибирає елемент із зазначеним місцем розташування, аналогічно до *: nth-child ()*, але, починаючи з останнього, у зворотньому напрямку;

– *: first-child* – дозволяє форматувати тільки найперший дочірній елемент тега;

– *: last-child* – дозволяє форматувати останній дочірній елемент тега;

– *: only-child* – вибирає елемент, який є єдиним дочірнім елементом;

– *: empty* – вибирає елементи, у яких немає дочірніх елементів;

– *: root* – вибирає елемент, який є кореневим в документі (елемент *html*).

Для більш точного відбору елементів для форматування можна використовувати *комбінації селекторів*, наприклад:

a [href] [title]

Такий селектор вибере всі посилання, для яких задані атрибути *href* і *title*;

*img [alt * = "css"]: nth-of-type (even)*

Такий селектор вибере всі парні малюнки, альтернативний текст яких містить слово *css*.

Один і той же стиль можна одночасно *застосувати до кількох елементів*. Для цього необхідно в лівій частині оголошення перелічити через кому потрібні селектори, наприклад:

h1,

h2,

p,

span {

color: red;

background: green;

}

Розглянемо ще два фундаментальні поняття в CSS – спадкування і каскадування.

1. Спадкування полягає в тому, що елементи успадковують властивості від свого «батька» (елемента, що їх містить).

Це механізм, за допомогою якого певні властивості передаються від предка до його нащадкам. Специфікацією CSS передбачено спадкування властивостей, що відносяться до текстового вмісту сторінки, таких як *color, font, letter-spacing, line-height, list-style, text-align, text-indent, text-transform, visibility, white-space* і *word- spacing*. У багатьох випадках це зручно, оскільки не потрібно задавати подібні властивості для кожного елемента веб-сторінки.

Властивості, що відносяться до форматування блоків, не успадковуються. Це *background*, *border*, *display*, *float*, *clear*, *height*, *width*, *margin*, *min- max-height*, *min- max-width*, *outline*, *overflow*, *padding*, *position*, *text-decoration*, *vertical-align* та *z-index*.

За допомогою ключового слова *inherit* можна примусити елемент успадковувати будь-яке значення властивості батьківського елемента. Це працює навіть для тих властивостей, які не успадковуються за замовчуванням.

Стилі можуть успадковуватися від батьківського елемента (успадковані властивості або за допомогою значення *inherit*).

Стилі, розташовані в таблиці стилів нижче, скасовують стилі, розташовані в таблиці вище.

До одного елемента можна застосовувати стилі з різних джерел. Перевірити, які стилі застосовуються, можна в режимі розробника браузера. При визначенні стилю можна використовувати будь-яку комбінацію селекторів – селектор елемента, псевдокласу елемента, класу або ідентифікатора елемента. Наприклад:

HTML:

```
<div id = "wrap" class = "box clear"> </ div>
```

CSS:

```
div {border: 1px solid #eee;}
#wrap {width: 500px;}
.box {float: left;}
.clear {clear: both;}
```

2. Каскадування проявляється в тому, як різні види таблиць стилів застосовуються до документу і як конфліктуючі правила скасовують одне одного.

Каскадування – це механізм, який керує кінцевим результатом в ситуації, коли до одного елемента застосовуються різні CSS-правила. Існує три критерії, які визначають порядок застосування властивостей – правило *!Important*, специфічність і порядок, в якому підключені таблиці стилів.

1. Правило *!Important.* Вагу правила можна задати за допомогою ключового слова *!Important*, яке додається відразу після значення властивості, наприклад:

span {font-weight: bold !Important;}

Таке оголошення буде мати пріоритет над усіма іншими правилами. Це правило дозволяє скасувати значення властивості і встановити нове для елемента з групи елементів в разі, коли немає прямого доступу до файлу зі стилями.

Специфічність. Для кожного правила браузер обчислює специфічність селектора, і якщо у елемента є конфліктуючі оголошення властивостей, до уваги береться правило, яке має найбільшу специфічність. Значення специфічності складається з чотирьох частин: 0, 0, 0, 0. Специфічність селектора визначається так:

– для *id* додається 0, 1, 0, 0;

– для *class* додається 0, 0, 1, 0;

– для кожного елемента і псевдоелемента додається 0, 0, 0, 1;

– для вбудованого стилю, доданого безпосередньо до елемента додається 1, 0, 0, 0;

– універсальний селектор не має специфічності.

Наприклад:

h1 {color: lightblue;} /* специфічність 0, 0, 0, 1 */

em {color: silver;} /* специфічність 0, 0, 0, 1 */

h1 em {color: gold;} /* специфічність: 0, 0, 0, 1 + 0, 0, 0, 1 = 0, 0, 0, 2 */

div # main p.about {color: blue;} /* специфічність: 0, 0, 0, 1 + 0, 1, 0, 0 + 0, 0, 0, 1 + 0, 0, 1, 0 = 0, 1, 1, 2 */

.sidebar {color: grey;} /* специфічність 0, 0, 1, 0 */

#sidebar {color: orange;} /* специфічність 0, 1, 0, 0 */

li # sidebar {color: aqua;} /* специфічність: 0, 0, 0, 1 + 0, 1, 0, 0=0, 1, 0, 1 */

В результаті до елемента буде застосовано те правило, специфічність якого більша. Наприклад, якщо на елемент діють дві специфічності зі

значеннями 0, 0, 0, 2 і 0, 1, 0, 1, то виграє друге правило.

Якщо до однієї веб-сторінки підключено декілька зовнішніх таблиць стилів і в різних таблицях будуть зустрічатися різні значення властивостей одного елемента, то в результаті до елемента буде застосовано правило, що знаходиться в таблиці стилів, яка йде в списку нижче.

У табл. 4 наведено властивості CSS, які використовуються найчастіше.Таблиця 4 – Властивості елементів CSS

Назва	Значення	Опис
background	*[background-color* || *background-image* || *background-repeat* || *background-attachment* || *background-position]* | *inherit*	Фон елемента
background-color	*<color>* | *transparent* | *inherit*	Колір фона
background-image	*<uri>* | *none* | *inherit*	Фонове зображення
background-position	*[[<percentage>* | *<length>]{1,2}* | *[[top* | *center* | *bottom]* || *[left* | *center* | *right]]]* | *inherit*	Положення фонового зображення
background-repeat	*repeat* | *repeat-x* | *repeat-y* | *no-repeat* | *inherit*	Повторення фонового зображення
border	*[border-width* || *border-style* || *<color>]* | *inherit*	Рамки елемента
border-collapse	*collapse* | *separate* | *inherit*	Об’єднання/розділення суміжних рамок таблиці
border-color	*<color>{1,4}* | *transparent* | *inherit*	Колір рамки
border-style	*<border-style>{1,4}* | *inherit*	Стиль лінії рамки

border-top border-right border-bottom border-left	*[border-top-width || border-style || <color>] | inherit*	Керквання стилем заданної рамки
border-width	*<border-width>{1,4} | inherit*	Товщина лінії рамки
bottom	*<length> | <percentage> | auto | inherit*	Низ елемента

Назва	Значення	Опис
clear	*none | left | right | both | inherit*	Заборона заповнення вільного простору поруч з елементом
clip	*<shape> | auto | inherit*	Обрізання вмісту елемента
color	*<color> | inherit*	Колір вмісту
cursor	*[[<uri> ,]* [auto | crosshair | default | pointer | move | e-resize | ne-resize | nw-resize | n-resize | se-resize | sw-resize | s-resize | w-resize| text | wait | help]] | inherit*	Форма курсора
display	*inline | block | list-item | run-in | compact | marker | table | inline-table | table-row-group | table-header-group | table-footer-group | table-row | table-column-group | table-column | table-cell | table-caption | none | inherit*	Спосіб відображення елемента
empty-cells	*show | hide | inherit*	Відображення пустих клітинок таблиці
float	*left | right | none | inherit*	Вільне розміщення елемента

font	*[[font-style || font-variant || font-weight]? font-size [/ line-height]? font-family] | caption | icon | menu | message-box | small-caption | status-bar | inherit*	Керування шрифтом

Назва	Значення	Опис
font-family	*[[<family-name> | <generic-family>],]* [<family-name> | <generic-family>] | inherit*	Гарнітура
font-size	*<absolute-size> | <relative-size> | <length> | <percentage> | inherit*	Кегль
font-style	*normal | italic | oblique | inherit*	Стиль шрифта
font-variant	*normal | small-caps | inherit*	Варіанти відображення шрифта
font-weight	*normal | bold | bolder | lighter | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900 | inherit*	Товщина шрифта
height	*<length> | <percentage> | auto | inherit*	Ширина елемента
left	*<length> | <percentage> | auto | inherit*	Положення лівої межі елемента
line-height	*normal | <number> | <length> | <percentage> | inherit*	Висота рядка
list-style	*[list-style-type || list-style-position || list-style-image] | inherit*	Стиль списка
margin	*<margin-width>{1,4} | inherit*	Зовнішній відступ

margin-top margin-right margin-bottom margin-left	*<margin-width> | inherit*	Зовнішній відступ із заданого боку
padding	*<padding-width>{1,4} | inherit*	Внутрішній відступ
z-index	*auto | <integer> | inherit*	Порядок переходу по натисканню клавиші Tab

Назва	Значення	Опис
padding-top padding-right padding-bottom padding-left	*<padding-width> | inherit*	Внутрішній відступ із заданого боку
position	*static | relative | absolute | fixed | inherit*	Позиціонування елемента
right	*<length> | <percentage> | auto | inherit*	Положення правої межі
text-align	*left | right | center | justify | <string> | inherit*	Вирівнювання текстового блока
text-decoration	*none | [underline || overline || line-through || blink] | inherit*	Текстові ефекти
text-indent	*<length> | <percentage> | inherit*	Абзацний відступ
text-transform	*capitalize | uppercase | lowercase | none | inherit*	Написання текста
top	*<length> | <percentage> | auto | inherit*	Положення верхньої межі елемента
vertical-align	*baseline | sub | super | top | text-top | middle | bottom | text-bottom | <percentage> | <length> | inherit*	Вертикальне вирівнювання в межах блоку

visibility	*visible* | *hidden* | *collapse* | *inherit*	Керування видимістю елемента
white-space	*normal* | *pre* | *nowrap* | *inherit*	Керування пробілами між словами
width	<*length*> | <*percentage*> | *auto* | *inherit*	Ширина елемента

Порядок виконання роботи

1. Розробити макети оформлення для створених у попередніх лабораторних роботах HTML-сторінок та реалізувати їх у будь-якому графічному редакторі.

2. Виконати оформлення відповідних HTML-сторінок відповідно до розроблених макетів з використанням зовнішніх таблиць стилів.

3. Отримати у викладача та реалізувати індивідуальне завдання щодо стильового оформлення HTML-документа.

Зміст звіту

1. Назва та мета роботи.

2. Опис виконання роботи та текст розробленого CSS-файла.

3. Письмові відповіді на контрольні запитання.

Контрольні питання

1. Види каскадних таблиць стилів. Їх переваги та недоліки.
2. Синтаксис описання стилю. Групи селекторів.
3. Властивості шрифту та тексту.
4. Властивості кольору, фону та обрамлення.
5. Стильове оформлення таблиць.
6. Стильове оформлення зображень.
7. Блочна верстка.
8. Позиціонування.
9. Робота з шарами.