МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Київський національний університет будівництва та архітектури

В.М. Хроленко, В.Г. Голенков

WEB-ПРОГРАМУВАННЯ

Конспект лекцій
для здобувачів першого (бакалаврського) рівня вищої освіти
за спеціальністю 122 “Комп’ютерні науки”

Київ 2024

УДК 681.3
Х 94

Рецензент О.В. Горда, канд. техн. наук, доцент

Відповідальна за випуск Т.А. Гончаренко, канд. техн. наук, доцент

Затверджено на навчально-методичній раді КНУБА, протокол № 11 від 20 травня 2024 року.

В авторській редакції.

Хроленко В.М.

Х 94 WEB-програмування: конспект лекцій / В.М. Хроленко, В.Г. Голенков. – Київ: КНУБА, 2024. – 68 с.

Містить теоретичні відомості з WEB-програмування, що допоможуть вирішувати такі завдання: вивчити теоретичні принципи та практичні засоби (сервіси) WEB-програмування, програмування у хмарних середовищах, зробити вибір інфраструктури, технології платформної незалежності програмного забезпечення тощо.

Призначено для здобувачів першого (бакалаврського) рівня вищої освіти спеціальності 122 “Комп’ютерні науки”.

КНУБА, 2024

Зміст

Загальні положення

Метою дисципліни „WEB-програмування” є навчити здобувачів створювати сучасні WEB-інтерфейси, працюючи з кодом вручну, на основі графічних макетів, підготовлених дизайнером. Студент зможе самостійно створювати WEB-сторінки початкового і середнього рівня складності.

WEB-програмування є дисципліною циклу професійної та практичної підготовки. Роль і значення дисципліни полягає в тому, що студенти знайомляться з сучасним станом розвитку WEB-програмування як програмного забезпечення. Ця дисципліна дає навички та тренінг, необхідний студентам у їх майбутній роботі.

Мета викладання навчальної дисципліни „WEB-програмування” полягає в тому, щоб навчити здобувачів використовувати основні прийоми створення розподілених застосувань з урахуванням сучасних концепцій і тенденцій розвитку хмарних технологій. При успішному закінченні курсу здобувач отримує вичерпні знання щодо ідей і механізмів реалізації інформаційних систем з використанням WEB-програмування.

В ході виконання практичних завдань кожен студент матиме можливість самостійно випробувати серверні і клієнтські частини WEB-програмування і перевірити їх функціонування в різних оточеннях та платформах.

В даній дисципліні вирішуються такі основні завдання: вивчаються теоретичні принципи та практичні засоби (сервіси) WEB-програмування, програмування у хмарних середовищах, вибору інфраструктури, технології платформної незалежності програмного забезпечення тощо.

Для успішного опанування WEB-програмування необхідно мати базові знання із дисциплін „Алгоритмізація та програмування”, „Об’єктно-орієнтоване програмування”, „Організація баз даних”, „Операційні системи”.

При вивченні дисципліни здобувачі повинні опрацьовувати лекційний матеріал, самостійно вивчати додаткову літературу, здійснювати підготовку до практичних занять. Поточний та підсумковий контроль здійснюються за допомогою усних опитувань на лекціях, проведення модульних письмових контрольних робіт, виконання індивідуальних лабораторних завдань за допомогою відповідного програмного забезпечення на комп’ютерах, проведення підсумкового комплексного контролю.

У курсі розглядаються такі теми:

1. Введення в WEB-технології.
2. Знайомство з HTML.
3. Знайомство з CSS.
4. Розмітка.
5. Графічний редактор для WEB-розробника.
6. Сітки.
7. Позиціонування.
8. Декоративні елементи.
9. Стилізація елементів змісту.
10. Публікація проекту.

Додатково: Уроки по JavaScript. Програмування JavaScript (JS) надає WEB-сторінкам можливість реагувати на дії користувача і перетворювати статичні сторінки в динамічний вид, так, щоб сторінки «оживали» на очах.

Результати навчання: вміння і навички вирішення типових задач клієнтської WEB-розробки, використовуючи мови HTML і CSS, HTML5, JavaScript (JS).

Для успішного опанування курсу необхідні базові навички роботи з комп'ютером і мережею Інтернет**.**

На комп'ютері повинен бути встановлений текстовий редактор з підсвічуванням синтаксису мов HTML і CSS, а також вільно розповсюджуваний графічний редактор GIMP (GNU Image Manipulation Program), який можна безкоштовно завантажити за адресою: http://www.gimp.org/

Лекція 1. Введення в HTML

Курс присвячений базовим технологіям WEB-програмування – HTML, CSS, HTML5.

Різновиди мов WEB-програмування

На сьогоднішній день майже кожна компанія, кожна організація, магазин має свій власний сайт.

Сайт виконує функцію реклами, а також спрощує роботу з клієнтами.

Вважається, що WEB-програмування – це створення сайтів і програм, які потребують мережі. Важливою ланкою між WEB-розробником і сайтами є мова WEB-програмування.

Мови WEB-програмування

Мова WEB-програмування – це сукупність операторів, за допомогою яких створюються коди WEB-програм, або їх ще називають скриптами, сценаріями.

Мова програмування передає зрозумілі комп'ютеру інструкції (коди) для виконання певних операцій.

Зазвичай коди, написані на WEB-мовах, читаються браузерами. Серед найпоширеніших мов WEB-програмування можна відзначити: HTML, CSS, PHP, JavaScript, Perl, jQuery.

Мова розмітки HTML

HTML (HyperText Markup Language – «мова розмітки гіпертексту») – найвідоміша для WEB-розробників мова програмування, хоча за своєю функціональністю вона швидше за все відноситься до мов розмітки.

HTML був створений в 1991-1992 винахідником Тімом Бернерс-Лі, британцем за походженням.

Мова застосовується для розподілу об'єктів і тексту на WEB-сторінках.

Мова оснащена тегами, які і є, по суті, інструкціями комп'ютера.

Мова програмування PHP

PHP (HypertextPreprocessor- «процесор гіпертексту») – є CI-подібною скриптовою мовою.

Найперша версія PHP була розроблена ще в 1994, але до 1998 року з'явилася основна версія PHP – 5.4.

Мова PHP широко використовується програмістами для написання сценаріїв, які виконуються на серверах при кожному оновленні сторінки сайту.

PHP дійсно схожа на мову CI, і багато вона запозичила з мови JAVA і технології JSP.

Сьогодні PHP використовується багатьма програмістами, тому ядром величезної кількості сайтів є php-код.

Ряд WEB-серверів передбачають вбудовані інтерпретатори спеціальних мов для динамічного WEB-програмування.

Прикладами є ASP для WEB-сервера Internet Information Server (IIS) і PHP (наприклад, для WEB-сервера Apache).

ASP (або, відповідно, PHP) сторінка являє собою звичайний HTML файл, який крім тексту і тегів HTML містить ще й інші спорудження відповідного мови (ASP або PHP)

Орієнтована мова програмування JavaScript

JavaScript – мова програмування, створена для «оживлення і додання динамічності» WEB-сайтів.

Розвиток мови почався в 1996 році. Програми, написані мовою JavaScript, називаються скриптами, які виконуються спільно з HTML – документами.

За допомогою JavaScript програмісти створюють деякі функції, як наприклад відкриття нового віконця з висновком в ньому повідомлення про помилку після деякого дії користувача.

Мова JavaScript здатна виконувати свої скрипти через задані інтервали часу. Загалом, JavaScript це і самостійна мова, але також її можна назвати допоміжною для інших, так як за допомогою неї легко зробити сайт більш функціональним і цікавим для користувача.

jQuery

jQuery – це бібліотека багаторазово використовуваних об'єктів і функції JavaScript, створена Джоном Резігом і представлена в 2006 році. Зазвичай jQuery є окремим JavaScript-файлом. jQuery можна назвати фреймворком (framework), тобто набором операції та інструкції для вирішення однотипних завдань. Бібліотека дозволяє вам працювати і управляти різними об'єктами на WEB-сторінках.

Динамічна мова програмування Perl

Perl (Practical Extraction and Report Language – «практична мова витягів і звітів») – мова програмування, чиєю найважливішою перевагою є розширені можливості роботи з текстом. Була створена лінгвістом Ларрі Уоллом в 1989 році. До функцій Perl також відносяться WEB-розробка,

системне адміністрування, розробка графічного інтерфейсу, ігр.Perl відомий також своїм величезним набором модулів. Зазвичай програмісти звертаються до мови Perl тоді, коли сценарій занадто складний для написання на інших мовах WEB-програмування.

Cascading Style Sheets

- CSS (Cascading Style Sheets – «каскадні таблиці стилів») – мова програмування, яка швидше за також відноситься до мов розмітки і форматування.
- CSS стала розроблятися в 1994 році Хокон Віум Лі і Бертом Босом.
- Основним завданням було створення мови, щоб форматувати HTML-об'єкти і текст: працювати зі шрифтами, кольорами, стилями.
- Загалом, CSS працює із зовнішнім виглядом сайтів. Мова CSS використовується з метою «прикрасити» WEB-сторінки.

MySQL

При створенні сайтів програмісти стикаються з проблемою зберігання величезної кількості інформації. Тут на допомогу можуть прийти бази даних, які дозволяють зберігати в собі необмежений обсяг даних. Для створення, ведення та використання баз даних існують СУБД (Система Управління Базами Даних). Однією з найбільш відомих і популярних серед WEB-розробників вважається реляційна СУБД MySQL. MySQL підтримує величезну кількість таблиць, вона часто застосовується разом з PHP.

Мови WEB-програмування - яку вибрати?

Нижче наведемо короткий огляд популярних фреймворків і мов WEB-програмування в рамках теми створення сайту, призначений для новачків сайтобудування.

Необхідно відразу сказати, що не існує однієї мови програмування, яка б переважала всі решту.

Перевага будь-якої мови програмування може проявлятися тільки в контексті якоїсь задачі. Багато задач можуть бути ефективно вирішені за допомогою будь-якої сучасної популярної мови програмування.

Часто вибір мови та фреймворку визначається тим, якими знаннями володіють програмісти, готові реалізувати даний проект.

Необхідно також розуміти відмінність між мовою програмування і фреймворком.

Мова програмування – це просто деякий базовий синтаксис (можливо зі стандартними бібліотеками), за допомогою якого можна створювати додатки.

Фреймворк же надає програмісту різні бібліотеки, що значно спрощують створення програм і сайтів.

Деякі мови і фреймворки являють собою нерозривне ціле (наприклад, ASP.NET і JSP).

Інші мови можуть використовуватися без фреймворку (PHP і Perl).

Незалежно від того, яка мова програмування буде обрана, в основі будь-якого сайту лежить мова гіпертекстової розмітки – HTML.

HTML повинні знати всі WEB-розробники. Не завадять хоча б базові знання HTML і тим сайтобудівникам, які нічого самі не пишуть, а використовують готові рішення (стандартні або замовні).

Власне, одного HTML вже досить для того, щоб робити сайти.

Але це будуть статичні сайти, без зворотного зв'язку з користувачами. Крім того, поновлення таких статичних сайтів трудомісткі. Для додання сайту динаміки використовуються мови WEB-програмування.

У першу чергу, мови WEB-програмування можна класифікувати на клієнтські і серверні.

Як випливає з назви, клієнтські мови використовуються для написання програм, які виконуються на стороні клієнта (WEB-браузер), а серверні – для програм, які виконуються на сервері.

Серед клієнтських мов WEB-програмування треба виділити JavaScript, що, також як і HTML, лежить в основі багатьох WEB-технологій (наприклад, в основі популярної останнім часом технології AJAX) і вміння програмувати на ньому ставиться до базових знань WEB-розробника.

Інші популярні клієнтські мови, а точніше фреймворки – це Adobe Flash (мова ActionScript) і SilverLight (будь-які .NET мови).

Adobe Flash застосовується WEB-майстрами дуже давно. Основне застосування цієї технології – інтерактивні сайти і сервіси, онлайнові ігри, мультимедійний контент і реклама.

SilverLight – це технологія, що розроблена компанією Microsoft і позиціонується як заміна Adobe Flash.

Не дивлячись на те, що за допомогою Adobe Flash або SilverLight можна побудувати повністю весь сайт, так робити не слід (за рідкісним винятком).

Справа в тому, що пошукові системи поки не вміють індексувати ні Adobe Flash, ні SilverLight.

Серверні мови WEB-програмування можуть бути умовно розділені по операційній системі, на якій вони працюють: Windows і * nix.

Це поділ до деякої міри умовно, тому що практично всі популярні мови і фреймворки перенесені на обидві ОС. Проте, вони рідко використовуються на нерідних ОС.

Якщо говорити про ОС Windows, то тут безроздільно панує технологія ASP.NET, розроблена компанією Microsoft.

За допомогою ASP.NET можна створювати сайти будь-якого рівня складності, від найпростіших, що складаються з декількох сторінок, до дуже складних, оброблюючих мільйони запитів на день (сайти Microsoft, написані на ASP.NET, є одними з найбільш відвідуваних в Інтернет).

Технологія ASP.NET приваблива для тих, хто непогано розбирається в ОС Windows, але незнайомий з Unix-подібними системами.

Основний недолік – менша, в порівнянні з * nix, кількість дешевих хостингів або необхідність покупки серверної ліцензії, у випадку з виділеним хостингом. Однак, у порівнянні з вартістю розробки складних сайтів, а також вартістю трафіку, різниця витрат на Windows і * nix хостинг може бути дуже мала.

Найпопулярнішою мовою WEB-програмування є, безумовно, PHP. Її основними перевагами є простий синтаксис, висока швидкодія, підтримка більшістю хостингів. Дуже вагомою перевагою є те, що на PHP написано багато популярних движків (наприклад, найпопулярніший движок для stand alone блогів – WordPress).

Інша популярна мова WEB-програмування на платформі Unix – мова Perl. Вона має складний заплутаний синтаксис і ніколи не призначалася для WEB-програмування. Не рекомендується її використовувати для створення сайтів.

JSP (Java Server Pages) – це частина технології J2EE, призначена для створення сайтів за допомогою мови Java.

JSP має дуже багато спільного з ASP.NET і вибір між цими двома технологіями найчастіше грунтується на суб'єктивних перевагах, а не на будь-яких перевагах чи недоліках цих платформ.

Останнім часом високу популярність придбала мова Ruby і, зокрема, фреймворк Ruby on Rails. З його допомогою можна дуже швидко створити

сайт з необхідною функціональністю. Одним із суттєвих недоліків Ruby є низька швидкодія.

Введення в HTML

Створення HTML документа

HTML – це мова гіпертекстової розмітки, яка складається з коду і тексту, що містить інструкції для браузера (оглядача WEB-сторінок) як відображати дані на екрані монітора.

Щоб почати кодувати, вам необхідно використовувати текстовий редактор.

Найпростіший текстовий редактор – це добре знайомий блокнот або WordPad, які є на кожному комп'ютері.

Відкриваєте, вносите свої записи, а потім зберігаєте файл з розширенням .HTML або .htm.

Слід, звичайно, визначитися з Директорією, вкажіть довільне ім'я файлу з розширенням HTML (наприклад, kobru.HTML зверніть увагу на точку).

Елементи

Будь-який HTML-документ являє собою набір елементів, що описують складові документа, такі як заголовки, списки, абзаци тексту, таблиці та ін.

Імена більшості елементів являють собою загальновживані слова англійської мови, зрозумілі скорочення і позначення.

Найчастіше елемент розмітки складається з трьох частин: початкового та кінцевого компонентів, між якими розміщуються текст або інші елементи документа. Ці компоненти являють собою спеціальні керуючі конструкції для розмітки вмісту HTML-документа і називаються тегами.

Код HTML

Код складається з тегів (від англ. Tag), або як ще їх називають – дескрипторів, які поділяються на парні і одиночні. Перші мають наступний синтаксис:

<**tag**>.....</**tag**>

У нашому випадку під tag розуміється будь-який парний тег, спочатку тег відкривається і наприкінці він закривається.

Самі теги укладені в кутових дужках <>. . Парні теги завжди повинні закриватися слешем (/).

Як правило всередині парних тегів міститься текст:

<**tag**>текст</**tag**>

Крім тексту теги можу включати в себе інші теги:

<**tag1**><**tag2**>текст</**tag2**></**tag1**>

Зверніть свою увагу на те, як закриваються вкладені теги! Теги повинні обов'язково закриватися у порядку їх відкриття. Така побудова нагадує матрьошку.

Варто зауважити, що написання тегів можливо як в нижньому, так і в верхньому регістрі, але, як правило, для зручності читання вони записуються в нижньому регістрі (так як ми показали вам вище), хоча запис у вигляді <TAG> ... </ TAG> помилки викликати не буде.

Теги можуть включати в себе атрибути, які беруть в лапки, наприклад, у вигляді вказівки кольору (англ. Color) і розміру ширини (width) та висоти (height) (про це ви дізнаєтеся пізніше в наступних уроках).

Приклад:

<**tag** height=«200»
width=«100»>текст</**tag**>

Тепер варто сказати пару слів про поодинокі теги. Їх синтаксис простий, вони не мають закривати тега, наприклад тег зображення img (від англ. Image):

<**img** width=«100»
height=«200»>

Структура HTML-документа

Структура HTML-файлу складається з *ключових елементів і об'єктів*. Один з головних елементів сторінки – відкривається тег <HTML> на самому початку сторінки і закривається тег </ HTML> в самому кінці сторінки:

<**HTML**>
</**HTML**>

Сказати по правді, ці теги необов'язкові в останній версії HTML і сучасних браузерах, але включення їх в документ вважається хорошим тоном. Між цими тегами прописується парний тег заголовка документу **<head> </ head>**, де містяться основні відомості про HTML-сторінці (назва, метадані, елементи сценарію і т.д.):

<**HTML**>
<**head**>

</**head**>
</**HTML**>

Серед такої інформації пишемо назву сторінки, яка позначається за допомогою парного тега **<title> </ title>:**

<**HTML**>
<**head**>
<**title**>Назва сторінки</**title**>
</**head**>
</**HTML**>

Також у відомостях про HTML-сторінку можуть бути присутніми метадані, в яких може міститися різного роду інформація: автор, кодування, опис, ключові слова для пошукових систем і т.п. (3 рядок нижче):

<**HTML**>
<**head**>
<**meta** http-equiv=«Content-Type» content=«text/HTML;
charset=windows-1251»/>
<**title**> Назва сторінки </**title**>
</**head**>
</**HTML**>

Крім цих даних в HTML-документ часто включають інформацію про його типи, яку необхідно розташувати на самому початку, наприклад так:

<!**DOCTYPE** HTML PUBLIC «-//W3C//DTD XHTML 1.0
Transitional//EN» «http://www.w3.org/TR/xHTML1/DTD/xHTML1-transitional.dtd«>
<**HTML**>
<**head**>
<**meta** http-equiv=«Content-Type» content=«text/HTML;
charset=windows-1251» />
<**title**> Назва сторінки </**title**>
</**head**>
</**HTML**>

Продовжуючи наше опис, важливо відзначити ту область HTML-документа, де відбувається основне інформаційне наповнення. Така область називається тілом документа і полягає між тегами **<body> і </ body>**:

<**HTML**>
<**head**>
<**title**> Назва сторінки </**title**>
</**head**>
<**body**>
Тут знаходиться тіло документа
</**body**>
</**HTML**>

Уважно подивіться на алгоритм дій. Наш парний тег **<body>** записується після закінчення тега </ head>, але всередині тегів <HTML> </ HTML>. Це дуже важливий елемент HTML-сторінки і ми почнемо вивчати його починаючи з наступного розділу.

Елемент META містить метаописувачі і деякі властивості документа, призначені для браузерів і пошукових систем.

Оголошення DOCTYPE

Згідно зі специфікацією HTML 4.01, директива DOCTYPE призначена для оголошення типу документа – так званого DTD (Document Type Definition, визначення типу документа). DTD є формальною конструкцією, в якій виражена вся специфіка HTML як одного з додатків мови SGML. Визначення типу документа являє собою перелік різних конструкцій SGML, які визначають, наприклад, які елементи в якому порядку можуть зустрічатися всередині кожного з елементів; повний список допустимих елементів із зазначенням на обов'язковість для кожного з них початкового і кінцевого тегів; повний список атрибутів для кожного елемента і значеннями за замовчуванням і інші правила синтаксису.

Оголошення DOCTYPE виглядає наступним чином:

<!DOCTYPE HTML PUBLIC «-//W3C//DTD HTML 4.01//EN» «http://www.w3.org/TR/HTML4/strict.dtd«>

Найбільш важливою частиною DOCTYPE є два рядки, взяті в лапки. Перший рядок «- // W3C // DTD HTML 4.01 // EN» стверджує, що даний документ є документом DTD, використовує англійську мову тексту для опису об'єкта, описує HTML версії 4.01 і опублікований організацією W3C. Другий рядок «http://www.w3.org/TR/HTML4/strict.dtd» вказує на розміщення самого документа DTD, використовуваного для цього DOCTYPE.

Абзац в HTML

Створення абзацу в HTML.

Теги абзаців є одними з найпоширеніших тегів в HTML-розмітки.

Для того, щоб на сторінці прописати абзац, необхідно в документі HTML використовувати спеціальний код у вигляді тега <p> (від англ. Paragraph). Цей тег абзацу є парним. Між «відкривається» і «закривається» тегами абзаців записується текст:

<**HTML**>
<**head**>
<**title**>Назва сторінки</**title**>
</**head**>
<**body**>
<**p**>Пишемо перший абзац</**p**>
</**body**>
</**HTML**>

Раніше було відзначено, що область, звана тілом документа, необхідна для інформаційного наповнення.

Як і що прописувати в тіло документа WEB-розробник вирішує самостійно.

З цих причин в подальшому ми не будемо описувати весь код в нашому візуальному блокноті, а обмежимося лише окремим кодом по темі уроку.

Наприклад, код, розташований нижче, повністю відповідає 07 рядку перерахованого вище коду:

<**p**>Пишемо перший абзац</**p**>

Вирівнювання абзаців в HTML

За замовчуванням абзаци вирівнюються по лівому краю, але це легко можна змінити за допомогою атрибута align, який приймає як параметр чотири варіанти, записаних в лапки:

- left (вирівнює абзац по лівому краю);
- center (вирівнює абзац по центру);
- right (вирівнює абзац по правому краю);
- justify (вирівнює абзац по ширині);

<**p** align=«left»> Цей абзац вирівняний по лівому краю </**p**>
<**p** align=«center»> Цей абзац центрований </**p**>
<**p** align=«right»> Тут абзац вирівняний з правого краю </**p**>

<**p** align=«justify»> Це останній абзац, він вирівняний по ширині</**p**>

Запустити!

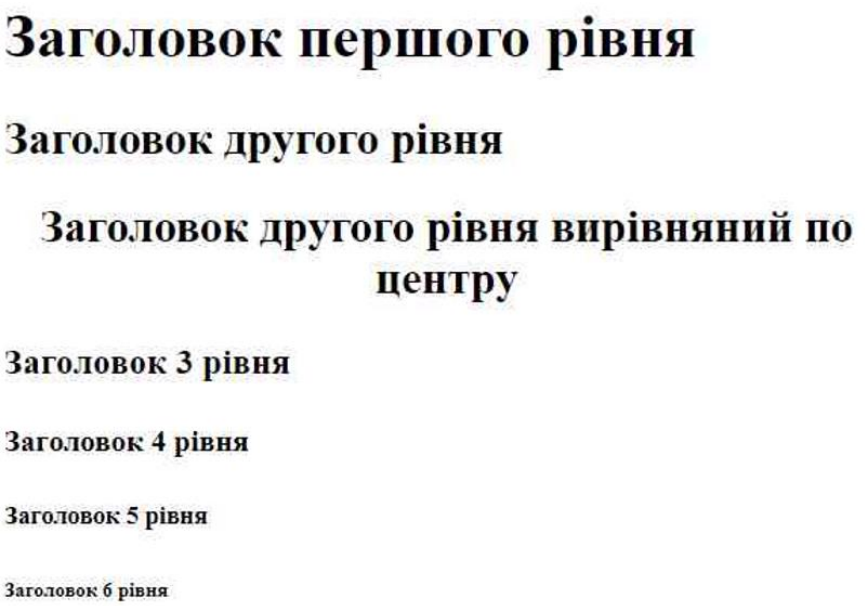
Заголовки в HTML

Тепер, коли ви знаєте як прописувати абзаци на сторінці, ви можете з їх допомогою складати цілі теми, розділи або глави, які потрібно виразно озаглавити.

У структурі HTML-документа все заголовки прописуються з використанням парного тега <h>.

Всього існує шість різних рівнів HTML заголовків від <h1> до <h6>, які виділяються напівжирним шрифтом:

<**h1**>Перший HTML заголовок</**h1**>
<**h2**>Другий HTML заголовок</**h2**>
<**h3**>Третій HTML заголовок</**h3**>
<**h4** align=«center»>Четвертий HTML заголовок</**h4**>
<**h5**>П’ятий HTML заголовок</**h5**>
<**h6**>Шостий HTML заголовок</**h6**>
Запустити!



Зверніть свою увагу на заголовок <h4>, до якого ми застосували атрибут горизонтального вирівнювання align з метою повторити пройдений матеріал для кращого засвоєння.

В даному заголовку нами присвоєно значення **center** атрибуту **align**, в результаті чого наш заголовок був успішно центрирован.

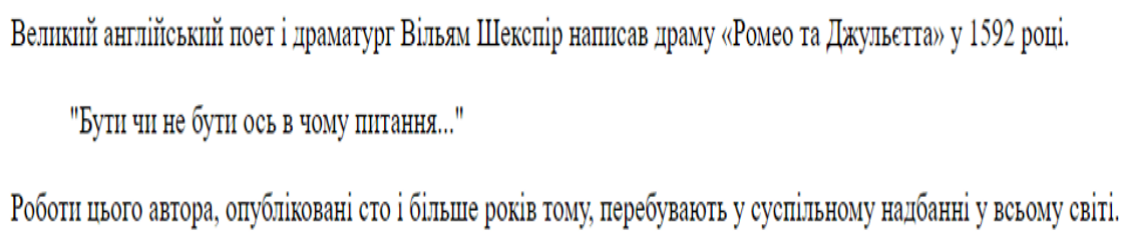
Цитати і коментарі в HTML

Тег цитати на WEB-сторінці.

Зазвичай, щоб виділити абзац на тлі решти сторінки його цитують за допомогою спеціального парного тега blockquote:

<**p**> Великий англійський поет та драматург Вільям Шекспір написав драму «Ромео та Джульєтта» в 1592 році</**p**>
<**blockquote**>«Бути чи не бути, ось у чому питання...»</**blockquote**>
<**p**> Роботи цього автора, опубліковані сто і більше років тому, перебувають у громадському надбанні у всьому світі.</**p**>\

Запустити!



Коментарі в HTML

Коментарі в HTML-документі дуже і дуже важливі теги!

Вони пояснюють написаний вами або іншим розробником код на сторінці.

Якщо ви після HTML-розмітки захочете вивчити якісь мови програмування, то обов'язково зіткнетеся з цими, ще раз повторимо, вкрай важливими тегами.

В HTML-розмітці теги коментарів зустрічаються не так часто, проте ми їх будемо використовувати в подальшому для пояснення будь-яких записів в нашому візуальному редакторі.

У різних мовах програмування вони позначаються по різному.

В HTML-розмітці додавання коментаря відбувається наступним чином:

<!-- Коментар в HTML -->

Перед коментарем вводимо тег <! - після чого пишемо текст коментаря, потім закриваємо тег коментаря ->.

Суть коментарів в тому, що вони не відображаються на WEB-сторінці, а використовуються тільки для пояснення того чи іншого фрагмента в коді документа.

Запустити!

```
<!-- Нижче пишемо текст абзаца -->
<p>Тут не буде ніяких коментарів!</p>
```

Тут не буде ніяких коментарів!

Теги пробілів і перенесення в HTML

Тег пробілу в HTML.

В основному, пробіли потрібні для того, щоб відокремлювати слова один від одного, хоча це необов'язково можуть бути слова. В HTML-коді ми без особливих проблем пишемо слова з пробілами і інтерпретатор легко їх розпізнає, але тільки один пробіл між слів. Іншими словами, скільки б ви не надрукували прогалин між словами, на екрані буде тільки один-єдиний пробіл і не більше того:

Запустити!

```
<!-- абзац нижче з одним пробілом між словами -->
<p>пробіл в HTML коді</p>
<!-- абзац нижче з кількома пробілами між словами -->
<p>пробіл в HTML коді</p>
<!-- Дані абзаци на сторінці виглядають однаково -->
```

пробіл в HTML-коді
пробіл в HTML-коді

Щоб надрукувати в HTML-документі відразу кілька пробілів між словами, які будуть візуально представлені на сторінці, використовуйте спеціальний знак пробілу – & nbsp;, що складається з шести символів. Подивимося на приклад:

Запустити!

```
<!-- абзац нижче з одним пробілом між словами -->
<p>пробіл в HTML коде</p>
<!-- абзац нижче з кількома пробілами між словами -->
<p>пробіл   в   
HTML   коде</p>
```

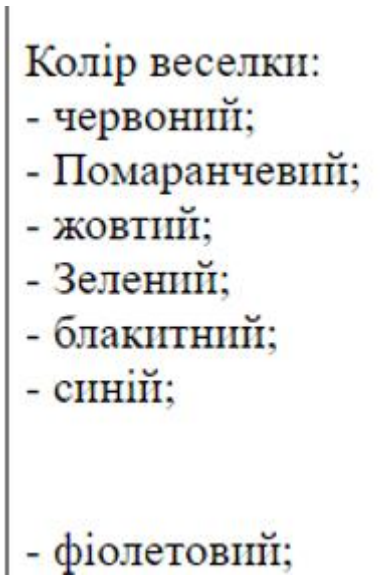
пробіл в HTML-коді
пробіл в HTML-коді

Перехід на новий рядок

Як правило, браузери переводять текст на новий рядок автоматично при досягненні правого краю вікна, але ви це можете зробити навмисне за допомогою спеціального тега
. Цей тег розриву рядків є поодиноким і не має закриває тега!

Запустити!

```
<p>
Колір веселки:<br>
- червоний; <br>
- Помаранчевий; <br>
- жовтий; <br>
- Зелений; <br>
- блакитний; <br>
- синій; <br><br><br>
- фіолетовий;
</p>
```



Зверніть вашу увагу на 8-й рядок вищезгаданого прикладу, де в самому кінці ми надрукували відразу три тега перенесення на новий рядок, тим самим ми додали порожні рядки і збільшили відстань між сусідніми словами «- синій;» і «- фіолетовий;».

Нумерований список HTML

Введення нумерованого списку.

Як ви можете здогадатися, нумеровані списки так називаються, бо в них йде послідовна нумерація елементів, де важливий порядок перерахування: 1 2 3 4 5 і так далі. Алгоритм побудови нумерованого списку складається з парного тега <ol>, усередині якого записують парні теги <li>:

Запустити!

```
<ol>
   <li>Один</li>
   <li>Два</li>
   <li>Три</li>
   <li>Чотири</li>
   <li>П’ять</li>
</ol>
```

1. Один
2. Два
3. Три
4. Чотири
5. П’ять

Щоб додати елемент в готовий список, вставте його в код і укладіть між тегами <li> і </ li>.

Стилі нумерації

Нумерований список необов'язково повинен бути оформлений арабськими цифрами. Додатково нумерація може бути представлена у вигляді латинських букв як в нижньому, так і в верхньому регістрі, або римськими цифрами. Для того, щоб змінити стандартну нумерацію списку необхідно прописати атрибут type з потрібним значенням (A a I і 1) між лапок в тезі <ol>:

Запустити!

```
<h4>Перший варіант:</h4>
    <ol type=«A»>
      <li>Один</li>
      <li>Два</li>
      <li>Три</li>
      <li>Чотири</li>
      <li>П’ять</li>
    </ol>
<h4>Другий варіант:</h4>
    <ol type=«a»>
      <li>Один</li>
      <li>Два</li>
      <li>Три</li>
      <li>Чотири</li>
      <li>П’ять</li>
    </ol>
<h4>Третій варіант:</h4>
    <ol type=«I»>
      <li>Один</li>
      <li>Два</li>
      <li>Три</li>
      <li>Чотири</li>
      <li>П’ять</li>
    </ol>
<h4>Четвертий варіант:</h4>
    <ol type=«i»>
```

```
            <li>Один</li>
            <li>Два</li>
            <li>Три</li>
            <li>Чотири</li>
            <li>П’ять</li>
        </ol>
    <h4>П’ятий варіант:</h4>
        <ol type=«1»>
            <li>Один</li>
            <li>Два</li>
            <li>Три</li>
            <li>Чотири</li>
            <li>П’ять</li>
        </ol>
```

Перший варіант:

A. Один
B. Два
C. Три
D. Чотири
E. П’ять

Другий варіант:

a. Один
b. Два
c. Три
d. Чотири
e. П’ять

Третій варіант:

I. Один
II. Два
III. Три
IV. Чотири
V. П’ять

Четвертий варіант:

i. Один
ii. Два
iii. Три

iv. Чотири
v. П’ять

П’ятий варіант:

1. Один
2. Два
3. Три
4. Чотири
5. П’ять

За замовчуванням, нумерація починається з першої позиції, щоб це змінити потрібно в тег <ol> додати атрибут start, де між лапок ви пишіть початок відліку:

Запустити!

```
<ol start=«3» type=«1»>
   <li>Один</li>
   <li>Два</li>
   <li>Три</li>
   <li>Чотири</li>
   <li>П’ять</li>
</ol>
```

3. Один
4. Два
5. Три
6. Чотири
7. П’ять

Маркований список

Додавання маркованого списку.

На відміну від нумерованих списків, в маркованих не важлива послідовність перерахування елементів списку. Їх також називають неупорядкованими списками. Синтаксис побудови такого списку схожий на нумерований, тільки складається з парного тега <ul>, усередині якого записують ті ж парні теги <li>:

Запустити!

```
<ul>
   <li>Один</li>
   <li>Два</li>
```

```
    <li>Три</li>
    <li>Чотири</li>
    <li>П’ять</li>
 </ul>
```

- Один
- Два
- Три
- Чотири
- П’ять

Стилі нумерації

За замовчуванням, марковані списки візуально представлені на сторінці у вигляді чорних суцільних кіл, званими маркерами. Звідси і назва списку.

Крім того, існують і інші стилі маркування (circle і square), які легко змінити за допомогою атрибута type:

Запустити!

```
<h4>Перший варіант:</h4>
<ul type=«circle»>
        <li>Один</li>
        <li>Два</li>
        <li>Три</li>
        <li>Чотири</li>
        <li>П’ять</li>
    </ul>
<h4>Другий варіант:</h4>
    <ul type=«square»>
        <li>Один</li>
        <li>Два</li>
        <li>Три</li>
        <li>Чотири</li>
        <li>П’ять</li>
    </ul>
<h4>Третій варіант:</h4>
    <ul type=«disc»>
        <li>Один</li>
```

```
        <li>Два</li>
        <li>Три</li>
        <li>Чотири</li>
        <li>П’ять</li>
    </ul>
```

Перший варіант:

- Один
- Два
- Три
- Чотири
- П’ять

Другий варіант:

- Один
- Два
- Три
- Чотири
- П’ять

Третій варіант:

- Один
- Два
- Три
- Чотири
- П’ять

Вкладені списки

Суть вкладених списків у тому, що один список знаходиться всередині іншого списку, причому типи списків можуть бути різними.

Таким чином утворюється великий багаторівневий список.

Щоб це продемонструвати, спершу напишемо маркований список, всередині якого вставимо нумерований список з перерахованими елементами, за винятком останнього:

Запустити!

```
 <ul type=«circle»>
    <li>Уроки HTML
        <ol>
          <li>Введення в HTML</li>
```

```
            <li>Структура HTML-документа</li>
            <li>Абзац в HTML</li>
            <li>Заголовки в HTML</li>
        </ol>
    </li>
    <li>Уроки CSS
        <ol type=«a»>
            <li>Відступи</li>
            <li>Поля</li>
            <li>Рамки</li>
        </ol>
    </li>
    <li>Уроки PHP
        <ol type=«i»>
            <li>Типыи даних</li>
            <li>Змінні</li>
            <li>Константи</li>
        </ol>
    </li>
    <li>Уроки JS
        <ul>
            <li>Введення в JavaScript</li>
            <li>Конструкція і побудова JavaScript</li>
            <li>Змінні і типи даних</li>
            <li>Оператори та їх класификація</li>
        </ul>
    </li>
</ul>
```

- Уроки HTML

1. Введення в HTML
2. Структура HTML-документа
3. Абзац в HTML
4. Заголовки в HTML

- Уроки CSS

1. Відступи
2. Поля

3. Рамки

- Уроки PHP

1. Типи даних
2. Змінні
3. Константи

- Уроки JS

1. Введення в JavaScript
2. Конструкція і побудова JavaScript
3. Оператори та їх класифікація

<!DOCTYPE HTML PUBLIC «-//W3C//DTD XHTML 1.0 Transitional//EN» «http://www.w3.org/TR/xHTML1/DTD/xHTML1-transitional.dtd«>

```
<HTML>
 <head>
  <meta http-equiv=«Content-Type» content=«text/HTML; charset=windows-1251» />
  <title>Name page</title>
 </head>
 <body>
  <p>First abzach</p>
 </body>
</HTML>
```

Контрольні запитання:

1. Які мови програмування ви знаєте?
2. Що таке JavaScript?
3. Як вибрати мову програмування?
4. Яка процедура створення HTML-документа?
5. Які списки ви знаєте?

Лекція 2. Основи CSS

Що таке CSS?

CSS – це формальна мова (зі своїм синтаксисом) опису зовнішнього вигляду документа, написаного з використанням мови розмітки. CSS розшифровується як Cascading Style Sheets – каскадні таблиці стилів.

Цей термін був запропонований норвезьким інженером Хоконом Віумом Лі у 1994 році, який на даний момент є технічним директором Opera Software. На даний момент найактуальніша версія CSS є CSS 3.

Навіщо це потрібно?

Стилі є зручним, практичним і ефективним інструментом при верстці WEB-сторінок і оформленні тексту, посилань, зображень та інших елементів. Незважаючи на явні плюси застосування стилів, розглянемо всі переваги CSS, в тому числі і непомітні на перший погляд.

Розширені в порівнянні з HTML способи оформлення елементів

На відміну від HTML, стилі мають набагато більше можливостей по оформленню елементів WEB-сторінок. Простими засобами можна змінити колір фону елемента, додати рамку, встановити шрифт, визначити розміри, положення і багато іншого.

Розмежування коду і оформлення

Ідея про те, щоб код HTML був вільний від елементів оформлення типу установки кольору, розміру шрифту та інших параметрів, стара як світ. В ідеалі, WEB-сторінка повинна містити тільки теги логічного форматування, а вигляд елементів задається через стилі. При подібному поділі робота над дизайном і версткою сайту може вестися паралельно.

Різне оформлення для різних пристроїв

За допомогою стилів можна визначити вид WEB-сторінки для різних пристроїв виводу: монітора, принтера, смартфона, планшета і ін., Наприклад, на екрані монітора відображати сторінку в одному оформленні, а при її друку – в іншому. Ця можливість також дозволяє приховувати або показувати деякі елементи документа при відображенні на різних пристроях.

Прискорення завантаження сайту

При зберіганні стилів в окремому файлі, він кешується і при повторному зверненні до нього витягується з кешу браузера. За рахунок

кешування і того, що стилі зберігаються в окремому файлі, зменшується код WEB-сторінок і знижується час завантаження документів.

Кешем називається спеціальне місце на локальному комп'ютері користувача, куди браузер зберігає файли при першому зверненні до сайту. При наступному зверненні до сайту ці файли вже не викачуються по мережі, а беруться з локального диска. Такий підхід дозволяє істотно підвищити швидкість завантаження WEB-сторінок.

Єдине стильове оформлення безлічі документів

Сайт – це не просто набір пов'язаних між собою документів, а й однакове розташування основних блоків, і їх вид. Застосування однакового оформлення заголовків, основного тексту та інших елементів створює спадкоємність між сторінками і полегшує користувачам роботу з сайтом і його сприйняття в цілому. Розробникам ж використання стилів істотно спрощує проектування дизайну.

Централізоване зберігання

Стилі, як правило, зберігаються в одному або декількох спеціальних файлах, посилання на які вказується в усіх документах сайту. Завдяки цьому зручно правити стиль в одному місці, при цьому оформлення елементів автоматично змінюється на всіх сторінках, які пов'язані із зазначеним файлом. Замість того, щоб модифікувати десятки HTML-файлів, досить відредагувати один файл зі стилем і оформлення потрібних документів відразу ж зміниться.

Способи підключення стилів

Розділяють 3 основних способи підключення та роботи зі стилями.

Вбудований

Цей спосіб ми вже бачили раніше у прикладах, коли вивчали HTML. Він найзручніший на дрібних проектах, але «найбрудніший» з усіх трьох. Для його використання потрібно додати атрибут style відповідного тегу, до якого ми хочемо застосувати стиль.

Коли ми пишемо вбудовані стилі, ми пишемо CSS-код в HTML-файл, безпосередньо всередині тега елемента за допомогою атрибута style:

HTML

<p style=«font-weight: bold; color: red;»> Зверніть увагу на цей текст.</p>

Такі стилі діють тільки на той елемент, для якого вони задані.

Вкладений (внутрішній, глобальний)

Вкладений стиль визначається за допомогою тега-елемента <style> (він закритий, тобто вимагає другий тег) всередині секції <head> HTML сторінки.

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

Приклад:

```
<style>
h1,
h2 {
color: red;
font-family: «Times New Roman», Georgia, Serif;
line-height: 1.3em;
}
</style>
</head>
<body>
...
</body>
```

Зовнішній

В даному випадком створюється окремий файл з розширенням .css в якому прописуються всі стилі, а на HTML сторінці в head прописується посилання в тезі <link> на файл.

Файл створюється в редакторі коду, так само як і HTML-сторінка.

Всередині файлу можуть містяться тільки стилі, без HTML-розмітки.

Зовнішня таблиця стилів підключається до WEB-сторінки за допомогою тега <link>, розташованого усередині розділу <head> </ head>.

Такі стилі працюють для всіх сторінок сайту.

До кожної WEB-сторінці можна приєднати кілька таблиць стилів, додаючи послідовно кілька тегів <link>, вказавши в атрибуті тега media призначення даної таблиці стилів.

rel = «stylesheet» вказує тип посилання (посилання на таблицю стилів).

```
<head>
<link rel=«stylesheet» href=«css/style.css»>
<link rel=«stylesheet» href=«css/assets.css» media=«all»>
</head>
```

Атрибут type = «text / css» не є обов'язковим за стандартом HTML5, тому його можна не вказувати. Якщо атрибут відсутній, за замовчуванням використовується значення type = «text / css».

Приклад:

```
<!DOCTYPE HTML>
<HTML>
 <head>
  <meta charset=«utf-8»>
  <title>Стили</title>
  <link rel=«stylesheet» href=«http://HTMLbook.ua/mysite.css»>
  <link rel=«stylesheet» href=«http://www.HTMLbook.ua/main.css»>
 </head>
 <body>
   <h1>Заголовок</h1>
   <p>Текст</p>
 </body>
</HTML>

H1 {
  color: #000080;
  font-size: 200%;
  font-family: Arial, Verdana, sans-serif;
  text-align: center;
}
P {
  padding-left: 20px;
}
```

Так само, не можна не згадати про стилі для різних платформ.

Широкий розвиток різних платформ і пристроїв змушує розробників робити під них спеціальні версії сайтів, що досить трудомістко і

проблематично. Разом з тим, часи і потреби змінюються, і створення сайту для різних пристроїв є неминучою і необхідною ланкою його розвитку.

З урахуванням цього, в CSS введено поняття типу носія, коли стиль застосовується тільки для певного пристрою.

Приклад:

<style>

@import «/**style**/**main**.css» screen; /* Стиль для вывода результата на монитор */

@import «/**style**/smart.css» print, handheld; /* Стиль для печати и смартфона */ **</style>**

Правило @import

Правило @import дозволяє завантажувати зовнішні таблиці стилів.

Щоб директива @import працювала, вона повинна розташовуватися в таблиці стилів (зовнішньої чи внутрішньої) перед усіма іншими правилами:

```
<style>
@import url(mobile.css);
p {
font-size: 0.9em;
color: grey;
}
</style>
```

HTML

Правило @import також використовується для підключення WEB-шрифтів:

```
@import
url(https://fonts.googleapis.com/css?family=Open+Sans&subset=latin,cyrillic);
```

CSS

Контрольні запитання:

1. Які є способи оформлення елементів?
2. Які способи підключення стилів ви знаєте?
3. Що таке правило @import?

Лекція 3. Основи CSS в1

Синтаксис CSS

Каскадні таблиці стилів описують правила форматування елементів за допомогою властивостей і допустимих значень цих властивостей.

Для кожного елемента можна використовувати обмежений набір властивостей, інші властивості не будуть чинити на нього ніякого впливу.

Оголошення стилю складається з двох частин: селектора і оголошення.

Оголошення складається з двох частин: ім'я властивості (наприклад, color) і значення властивості (grey).

Селектор повідомляє браузеру, який саме елемент форматувати, а в блоці оголошення (код в фігурних дужках) перераховуються команди - властивості і їх значення.

В HTML імена елементів нечутливі до регістру, тому «h1» працює так само, як і «H1».

Основним поняттям виступає селектор – це ім'я стилю, для якого додаються параметри форматування.

Як селекторі виступають теги, класи і ідентифікатори.

Загальний спосіб запису має наступний вигляд (спрощений).

Спочатку пишеться ім'я селектора, наприклад, TABLE, це означає, що все стильові параметри будуть застосовуватися до тегу <table>, потім йдуть фігурні дужки, в яких записується стильова властивість, а його значення вказується після двокрапки.

Стильові властивості поділяються між собою крапкою з комою, в кінці цей символ можна опустити.

CSS не чутливий до регістру, перенесення рядків, прогалин і символів табуляції, тому форма запису залежить від бажання розробника.

Селектори представляють структуру WEB-сторінки. З їх допомогою створюються правила для форматування елементів WEB-сторінки.

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

Універсальний селектор

Відповідає будь-якому HTML-елементу. Наприклад, * {margin: 0;} обнулить зовнішні відступи для всіх елементів сайту.

Також селектор може використовуватися в комбінації з псевдоклас або псевдоелементи:

*: after {CSS-стилі}, *: checked {CSS-стилі}.

Селектор елемента

Селектори елементів дозволяють форматувати всі елементи даного типу на всіх сторінках сайту.

Наприклад, h1 {font-family: Lobster, cursive;} задасть загальний стиль форматування всіх заголовків h1.

Селектор класу

Селектор класу використовується для вибору одного або декількох HTML-елементів з однаковим значенням атрибута CSS класу **class**.

Приклад:

HTML код:

<P class = «content»>

<Font>

<Font class = ««> Це наш перший пункт. </ Font>

</ Font>

</ P>

<P class = «content»>

<Font>

<Font> Це наш другий абзац. </ Font>

</ Font>

</ P>

<P class = «content»>

<Font>

<Font> Це наш третій абзац. </ Font>

</ Font>

</ P>

CSS

.content {margin: 20px 0; line-height: 24px; font-size: 15px}

Селектори класу дозволяють задавати стилі для одного і більше елементів з однаковим ім'ям класу, розміщених в різних місцях сторінки або на різних сторінках сайту.

Наприклад, для створення заголовка з класом headline необхідно додати атрибут class зі значенням headline що відкриває тег <h1> і задати стиль для зазначеного класу.

Стилі, створені за допомогою класу, можна застосовувати до інших елементів, не обов'язково даного типу.

```
<h1 class=«headline»> Інструкція користування персональним комп'ютером </h1>
```

HTML

```
.headline {
text-transform: uppercase;
color: lightblue;
}
```

CSS

Якщо елемент має декілька атрибутів класу, їх значення об'єднуються з пробілами.

```
<h1 class=«headline post-title»> Інструкція користування персональним комп'ютером </h1>
```

Селектори ідентифікаторів

Селектори ідентифікаторів дозволяють вам змінювати стиль елемента HTML, який має атрибут id, незважаючи на його позицію в дереві документа, але тільки один раз на сторінці.

Значення id має бути унікальним, на одній сторінці може зустрічатися тільки один раз і повинна містити щонайменше один символ. Значення не повинно містити пробілів.

Немає ніяких інших обмежень на те, яку форму може приймати id, зокрема, ідентифікатори можуть складатися тільки з цифр, починатися з цифри, починатися з підкреслення, складатися тільки з розділових знаків і т. д. Унікальний ідентифікатор елемента може використовуватися для різних цілей, зокрема, як спосіб посилання на конкретні частини документа з використанням ідентифікаторів фрагментів, як спосіб націлювання на елемент при створенні сценаріїв і як спосіб стилізації конкретного елемента з CSS.

```
<div id=«sidebar»></div>
```

HTML

```
#sidebar {
width: 300px;
float: left;
}
```

Приклад:

Код HTML:

```
<Div id = «intro»>
    <P> This paragraph is in the intro section. </ P>
</ Div>
    <P> This paragraph is not in the intro section. </ P>
```

Код CSS:

```
#intro {
color: white;
background-color: gray; }
```

Коментарі

Коментар в CSS виглядає наступним чином:

/ * Коментар * /

Виділення HTML тексту

Існує кілька способів виділення тексту у HTML-розмітці.

Основними є: напівжирне виділення тексту; виділення тексту курсивом та виділення підкресленням. Всі ці методи ми розглянемо докладніше.

Виділення HTML тексту напівжирним

Виділення тексту напівжирним шрифтом у HTML-розмітці можливе завдяки використанню парного тега :

Запустити!

```
<p>Сайт <b>KobRU</b> створений спеціально для WEB-розробників початківців!</p>
```

Сайт **KobRU** створений спеціально для WEB-розробників початківців!

У нашому випадку ми вставили перед словом Kobru тег, що відкривається , а в самому кінці слова поставили тег, що закривається , в результаті чого отримали виділення напівжирним шрифтом.

Варто зазначити, що синонімом тега є тег :

Запустити!

<**p**>Сайт <**strong**>KobRU</**strong**> создан специально для начинающих WEB-разработчиков!</**p**>

Сайт **KobRU** створений спеціально для WEB-розробників початківців!

Щоб виділити текст курсивом у структурі HTML-документа, вам потрібно укласти вибраний текст у парний тег <i>, як показано нижче:

Запустити!

<**p**>
Сайт <**strong**>KobRU</**strong**> створений спеціально для
<**i**>WEB-розробників</**i**> початківців!
</**p**>

Сайт **KobRU** створений спеціально для *WEB-розробників* початківців!

Щоб виділити текст курсивом у структурі HTML-документа, вам потрібно вкласти вибраний текст у парний тег <i>, як показано нижче:

Запустити!

<**p**>
Сайт <**strong**>KobRU</**strong**> створений спеціально для
<**i**>WEB-разработчиков</**i**> <**u**>початківців</**u**>!
</**p**>

Сайт **KobRU** створений спеціально для *WEB-розробників* початківців!

Інші способи виділення тексту

Запустити!

<**p**><**strike**>інші
сайти</**strike**></**p**>

~~Інші сайти~~

Запустити!

<**p**>Сайт
KobRU<**sup**>2</**sup**
</**p**>

Сайт KobRU^2

Запустити!

<**p**>Сайт
KobRU<**sub**>2</**sub**
</**p**>

Сайт $KobRU_2$

Гарнітура та розмір шрифту

Гарнітура шрифту

Поміняти гарнітуру шрифту в HTML-розмітці можна за допомогою парного тега <font> і атрибута face, значення якого записують назву шрифту:

```
<p><font face=«Arial, Times New Roman»>Робота з HTML тегами</font></p>
```

У нашому прикладі атрибут face містить через кому відразу кілька шрифтів.

Це свого роду страховка на той випадок, якщо бажаний шрифт не відобразиться.

Справа в тому, що за замовчуванням, на вашому комп'ютері є кілька стандартних шрифтів, наприклад Arial, Times New Roman, Verdana і т.п.

Ви також можете вказати інші екзотичні шрифти, але потрібно враховувати той факт, що такі шрифти можуть і не бути в арсеналі користувачів вашого сайту, тому завжди вказуйте альтернативу у вигляді кількох стандартних шрифтів!

Існує інший варіант відображення нестандартних шрифтів на станиці для користувачів, на комп'ютері яких немає таких шрифтів.

Він повинен зберігатись на сервері у форматі .eot. Подивимося приклад:

```
@font-face {font-family: «Scriptorama»; src: url(scriptorama.eot)}
```

Цей приклад використовує каскадні таблиці стилів, які ми обговоримо в розділі CSS.

Зміна розміру шрифту

На сторінці розмір тексту можна змінити за допомогою атрибута size, який набуває значення від 1 до 7 - це розміри шрифтів від 8пт до 36 пт:

Запустити!

```
<p><font face=«Arial, Times New Roman» size=«4»>Робота з HTML тегами</font></p>
```

Зміна розміру всього тексту задається на початку HTML-документа, відразу після тега body прописуєте тег <basefont size=««>, де у значенні атрибута size вказуєте потрібний шрифт від 1 до 7.

Варто зазначити, що з цією метою найкраще використовувати каскадні таблиці стилів (CSS). Це більш раціональне та сучасне рішення!

Крім абсолютних розмірів шрифтів можна задати і їхню відносну величину символами + або -:

Запустити!

```
<p><font face=«Arial, Times New Roman» size=«4»>Робота з HTML тегами</font></p>
```

Зміна розміру всього тексту задається на початку HTML-документа, відразу після тега body прописуєте тег <basefont size=««>, де в значенні атрибута size вказуєте потрібний шрифт від 1 до 7.

Варто зазначити, що з цією метою найкраще використовувати каскадні таблиці стилів (CSS). Це більш раціональне та сучасне рішення!

Крім абсолютних розмірів шрифтів можна задати і їхню відносну величину символами + або -:

```
<!-- Збільшуємо текст на два розміри →
<p><font face=«Arial, Times New Roman» size=«+2»>Робота з HTML тегами</font></p>
<!-- Зменшуємо текст на два розміри -->
<p><font face=«Arial, Times New Roman» size=«-2»>Робота з HTML тегами</font></p>
```

Зміна кольору тексту в HTML

Змінити колір тексту в HTML-документі досить просто. Вам потрібно укласти потрібний фрагмент тексту в парний тег <font> прописати у тезі, що відкривається, атрибут color і поставити в потрібний колір:

Запустити!

```
<p>виділення <font color=«red»>тексту</font> в HTML</p>
```

Виділення тексту у HTML

Самі кольори в HTML-розмітці виражаються шістнадцятковими значеннями, які починаються із символу ґрат (#) і складаються із шести знаків.У прикладі вище ми використали англійський словесний еквівалент червоного кольору (red).

У списку наведеному нижче представлені 16 основних кольорів:

Колір: Код:

black	#000000
white	#ffffff
blue	#0000ff
red	#ff0000
yellow	#ffff00
green	#008000
purple	#800080
gray	#808080
silver	#c0c0c0
navy	#000080
fuchsia	#ff00ff
lime	#00ff00
maroon	#800000
olive	#808000
teal	#008080

Колір тексту на сторінці має бути розбірливим! Підбирайте зручний для сприйняття на око колір тексту! Щоб змінити колір всього тексту на сторінці HTML необхідно прописати у тезі <body> атрибут text с потрібним значенням:

<**body**
text=«#cccccc»>

Кордони та фон сторінки

Відступи від краю сторінки

Відступ від краю зверху, знизу, праворуч і ліворуч у HTML-документі задається за допомогою спеціальних атрибутів, що керують кордонами, які в різних браузерах прописуються по-різному:

Для Internet Explorer:	Для Netscape Navigator:
leftmargin відступ зліва	marginwidth;
rightmargin відступ справа	
topmargin відступ зверху	marginheight;
bottommargin відступ знизу	

<body leftmarging=«100» rightmarging=«100» topmargin=«0» marginwidth=«100»>

Для того, щоб сторінка HTML виглядала коректно в будь-якому WEB-браузері, вказуйте і ті, й інші атрибути.

Варто також відзначити, що раціональне рішення - застосування каскадних таблиць стилів (CSS).

Фон сторінки сайту у HTML

У HTML фон сторінки задається за допомогою атрибуту bgcolor тега <body>:

```
<!-- класичний білий фон сторінки -->
    <body bgcolor=«#ffffff»>
```

Вставлення зображення на сайт

HTML-код зображення

На WEB-сторінці зображення може бути у вигляді картинки, фотографії, логотипу, піктограми, піктограми та іншого візуального об'єкта.

Справа в тому, що в HTML-документі зображення представлене у вигляді вбудованого елемента. Тобто код зображення міститься в тілі документа разом із текстом.

Щоб вставити зображення на сайт, пропишіть в потрібному місці тіла документа одинарний тег <img>.

У HTML-розмітці він не є парним і не вимагає тега, що закривається.

Як атрибут впишіть src=««, у значенні якого вкажіть шлях до файлу зображення.

Наприклад так:

```
<img src=«photo.jpg»>
```

У HTML-розмітці серед множини розширень використовують три типи файлів: .jpg; .gif та .png.

Перед тим, як вставити зображення на сторінку, його можна обробити у графічному редакторі.

Один з найпопулярніших, безперечно, є Adobe Photoshop, який містить багато функцій з обробки зображень, у тому числі і для зменшення розміру і збереження файлу у форматі WEB.

Намагайтеся оптимізувати зображення так, щоб воно не перевищувало оптимального розміру для HTML-документа в 60Кб.

Великі зображення довго завантажуються і користувачі вашого ресурсу можуть у такому разі не дочекатися завантаження і перейти до конкурентів. Крім вищесказаного, є користувачі, які навмисне відключають зображення на WEB-сторінці або для прискореного завантаження, або у них мобільний

інтернет з високою оплатою за Мб або є ще якісь причини так робити. Тому хорошим тоном вважається в тегу зображення прописувати атрибут alt=««, в якому вказуєте альтернативний текст з описом зображення.

Існує також атрибут title=««, у значенні якого вказуєте назву картинки:

<**img** src=«photo.jpg» alt=«альтернативний текст» title=«назва картинки»>

Розмір зображення

Якщо зображення на WEB-сторінці виглядає великим або навпаки занадто маленьким, можна за допомогою коду HTML маніпулювати розмірами.

Для того, щоб змінити ширину, потрібно вписати атрибут width=««, для висоти - атрибут height=««, у значенні яких вказуєте розмір у px (пікселях) або у відсотковому співвідношенні:

<**img** src=«photo.jpg» width=«100» height=«80»>

Ви також можете використовувати графічний редактор, який дасть повний контроль над зображенням та дозволить оптимізувати його для WEB-сторінки.

І не забувайте про авторське право!

Вирівнювання зображень

У HTML-документі зображення вирівнюються по горизонталі, по вертикалі та по центру.

Горизонтальне вирівнювання зображення

Для того, щоб керувати зображенням по горизонталі, використовуйте атрибут align=««.

У його значенні вкажіть left, якщо хочете вирівняти зображення з лівого краю сторінки, або значення right - з правого краю.

За промовчанням зображення завжди вирівнюється по лівому краю сторінки:

<**img** src=«photo.jpg» align=«right»>

Вертикальне вирівнювання зображення

За допомогою того ж атрибуту align=«« можна керувати розташуванням зображення та по вертикалі.

Для цього у його значенні потрібно прописати

• top для вирівнювання по верхній межі,

• middle – по середині та bottom – по нижній межі:

<**img** src=«images/photo.jpg» align=«top»>

Примітка: запис images/photo.jpg означає, що зображення знаходиться не в одній директорії, а в папці images.

Вирівнювання зображення по центру сторінки

Крім того, зображення можна центрувати.

Для цього вставте тег зображення в парний тег :

<**center**>
<**img** src=«photo.jpg» align=«top»>
</**center**>

Контрольні запитання:

1. Що таке універсальний селектор?
2. Які гарнітури ви знаєте?
3. Як змінити колір тексту?
4. Як встановити кордони і фон сторінки?
5. Як вставити на сайт зображення?

Лекція 4. Основи CSS. Властивості

Детальніше про властивості

Перед тим, як поглибитися в самі властивості CSS розглянемо таблиці вимірювань, які використовуються.

Колір:	Код:
black	#000000
white	#ffffff
blue	#0000ff
red	#ff0000
yellow	#ffff00
green	#008000
purple	#800080
gray	#808080
silver	#c0c0c0

navy #000080
fuchsia #ff00ff
lime #00ff00
maroon #800000
olive #808000
teal #008080

Для задання кольорів використовуються числа в шістнадцятковому коді. Так само браузери підтримують деякі кольори по їх назві.

Можна визначити колір, використовуючи значення червоної, зеленої та синьої компоненти в десятковому численні. Значення кожного з трьох кольорів може набувати значень від 0 до 255.

Також можна задавати колір в процентному відношенні. Спочатку вказується ключове слово rgb, а потім в дужках, через кому вказуються компоненти кольору, наприклад rgb (255, 0, 0) або rgb (100%, 20%, 20%).

Властивості

Встановлює сімейство шрифту, яке буде використовуватися для оформлення тексту вмісту. Список шрифтів може включати одне або кілька назв, розділених комою. Якщо в імені шрифту містяться прогалини, наприклад, Trebuchet MS, воно повинно полягати в одинарні або подвійні лапки.

Коли браузер зустрічає перший шрифт у списку, він перевіряє його наявність на комп'ютері користувача. Якщо такого шрифту немає, береться наступне ім'я зі списку і також аналізується на присутність. Тому кілька шрифтів збільшує ймовірність, що хоча б один з них буде виявлений на клієнтському комп'ютері. Закінчують список зазвичай ключовим словом, яке описує тип шрифту - serif, sans-serif, cursive, fantasy або monospace. Таким чином, послідовність шрифтів краще починати з екзотичних типів і закінчувати узагальненим ім'ям, яке задає вид накреслення (рис.1).

• serif - шрифти із зарубками (антиквенні), типу Times;

• sans-serif - рубані шрифти (шрифти без зарубок або гротески), типовий представник - Arial;

• cursive - курсивні шрифти;

• fantasy - декоративні шрифти;

• monospace - моноширинні шрифти, ширина кожного символу в такому сімействі однакова (шрифт Courier).

Приклад:

The HTML:

```
<p class=«serif»>
  This is a paragraph shown in serif font.
</p>
<p class=«sansserif»>
  This is a paragraph shown in sans-serif font.
</p>
<p class=«monospace»>
  This is a paragraph shown in monospace font.
</p>
<p class=«cursive»>
  This is a paragraph shown in cursive font.
</p>
<p class=«fantasy»>
  This is a paragraph shown in fantasy font.
</p>
```

The CSS:

```
p.serif {
  font-family: «Times New Roman», Times, serif;
}
p.sansserif {
  font-family: Helvetica, Arial, sans-serif;
}
p.monospace {
  font-family: «Courier New», Courier, monospace;
}
p.cursive {
  font-family: Florence, cursive;
}
p.fantasy {
  font-family: Blippo, fantasy;
}
```

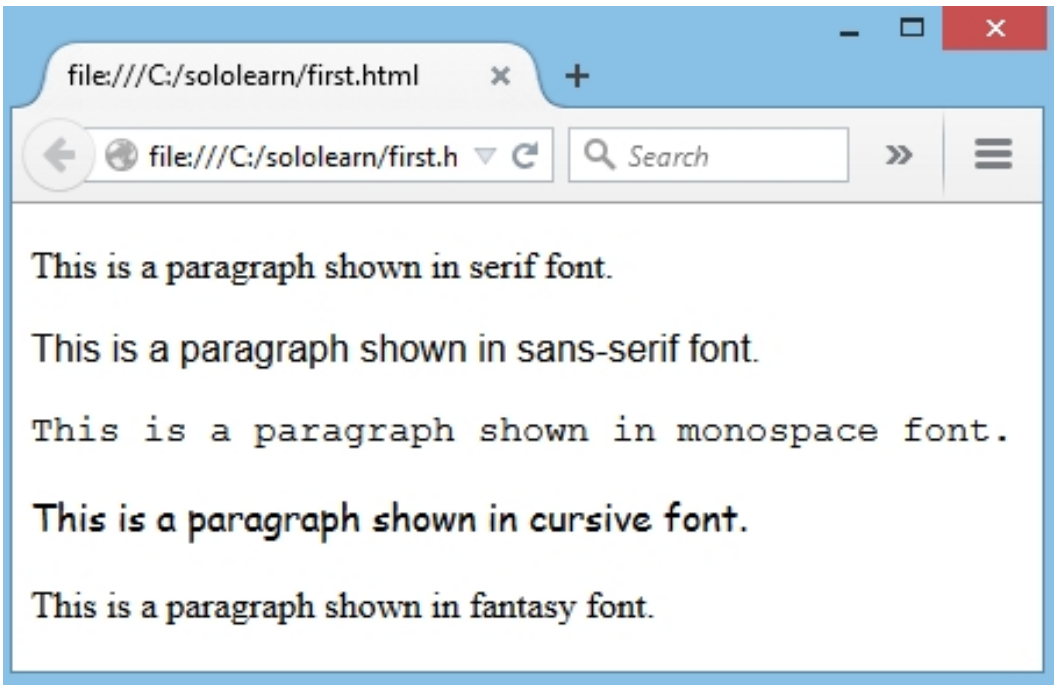


Рисунок 1. Властивість font-family

Розмір шрифту: font-size

Визначає розмір шрифту елемента. Розмір може бути встановлений декількома способами. Набір констант (xx-small, x-small, small, medium, large, x-large, xx-large) задає розмір, який називається абсолютним. По правді кажучи, вони не зовсім абсолютні, оскільки залежать від налаштувань браузера і операційної системи.

Інший набір констант (larger, smaller) встановлює відносні розміри шрифту. Оскільки розмір успадкований від батьківського елемента, ці відносні розміри застосовуються до батьківського елементу, щоб визначити розмір шрифту поточного елементу.

В кінцевому підсумку, розмір шрифту сильно залежить від значення властивості font-size у батька елемента.

Приклад:

The HTML:

<P class = «small»>
  Paragraph text set to be small
</ P>
<P class = «medium»>
  Paragraph text set to be medium
</ P>
<P class = «large»>

Paragraph text set to be large

</ P>

<P class = «xlarge»>

Paragraph text set to be very large

</ P>

The CSS:

```
p.small {
  font-size: small; }
p.medium {
  font-size: medium; }
p.large {
  font-size: large; }
p.xlarge {
  font-size: x-large; }
```

Стиль шрифту: font-style

Визначає зображення шрифту - звичайне, курсивне або похиле. Коли для тексту встановлено курсивне або похиле накреслення, браузер звертається до системи для пошуку відповідного шрифту. Якщо заданий шрифт не знайдений, браузер використовує спеціальний алгоритм для імітації потрібного виду тексту. Результат і якість при цьому можуть вийти незадовільними, особливо при друці документа.

• Normal - Звичайне зображення тексту.

• Italic - Курсивне накреслення.

• Oblique - Похиле накреслення. Курсив і похилий шрифт при всій їх схожості не одне і те ж. Курсив це спеціальний шрифт імітує рукописний, похилий же утворюється шляхом нахилу звичайних знаків вправо (рис.2).

Приклад:

The HTML:

<P class = «normal»> This paragraph is normal. </ P>

<P class = «italic»> This paragraph is italic. </ P>

<P class = «oblique»> This paragraph is oblique. </ P>

The CSS:

```
p.normal {
font-style: normal;
}
p.italic {
```

```
font-style: italic;
}
p.oblique {
font-style: oblique;
}
```

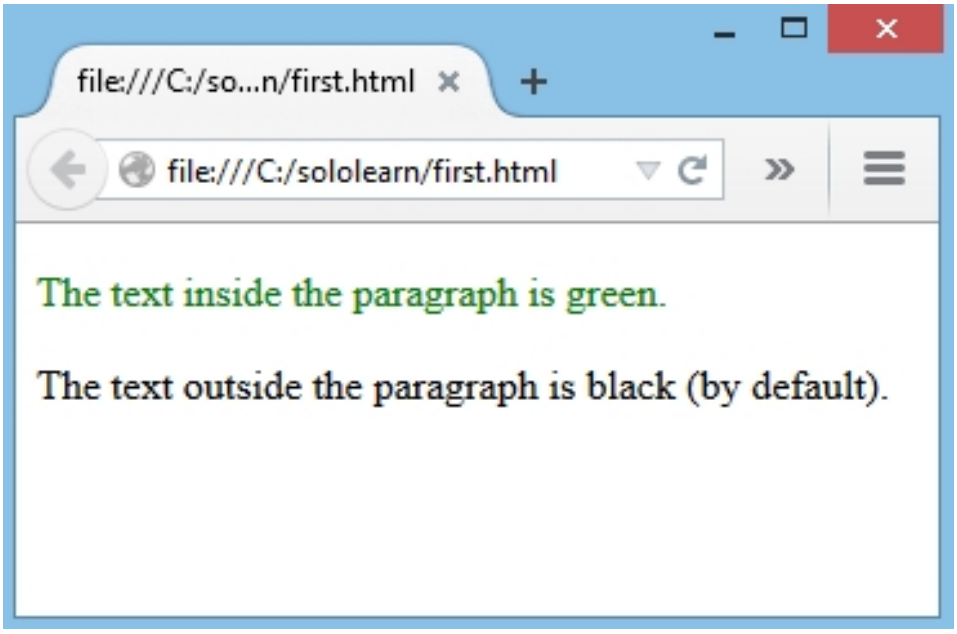


Рисунок 2. Властивість font-style

Вирівнювання тексту: text-align

Далі визначаємо горизонтальне вирівнювання тексту в межах елементу. Center - вирівнювання тексту по центру. Текст розміщується у центрі горизонталі вікна браузера або контейнера, де розташований текстовий блок. Рядки тексту немов нанизуються на невидиму вісь, яка проходить по центру WEB-сторінки. Подібний спосіб вирівнювання активно використовується в заголовках і різних підписах, він надає офіційний вигляд оформленню тексту. У всіх інших випадках вирівнювання по центру застосовується рідко з тієї причини, що читати великий обсяг такого тексту незручно.

Justify - вирівнювання по ширині, що означає одночасне вирівнювання по лівому і правому краю. Щоб зробити цю дію, браузер в цьому випадку додає пропуски між словами.

Left - вирівнювання тексту по лівому краю. В цьому випадку рядки тексту вирівнюється по лівому краю, а правий край розташовується «драбинкою». Такий спосіб вирівнювання є найбільш популярним на

сайтах, оскільки дозволяє користувачеві легко відшукувати поглядом новий рядок і комфортно читати великий текст.

Right - вирівнювання тексту по правому краю. Цей спосіб вирівнювання виступає в ролі антагоніста щодо попереднього типу. А саме, рядки тексту рівняються по правому краю, а лівий залишається «рваним». Через те, що лівий край не вирівняний, а саме з нього починається читання нових рядків, такий текст читати важче, ніж, якби він був вирівняний по лівому краю. Тому вирівнювання по правому краю застосовується зазвичай для коротких заголовків обсягом не більше трьох рядків (рис.3). Ми не розглядаємо специфічні сайти, де текст доводиться читати справа наліво, там можливо подібний спосіб вирівнювання і стане в нагоді.

Auto - не змінює положення елемента.

Inherit - успадковує значення батька.

Приклад:

The HTML:

This paragraph is aligned to left.

This paragraph is aligned to right.

This paragraph is aligned to center.

The CSS:

```
p.left {
  text-align: left;
}
p.right {
  text-align: right;
}
p.center {
  text-align: center;
}
```

Рисунок 3. Властивість text-align

Тіні тексту: text-shadow

Додає тінь до тексту, а також встановлює її параметри: колір тіні, зміщення щодо напису і радіус розмиття.

None - скасовує додавання тіні.

Колір - колір тіні в будь-якому доступному CSS форматі. За замовчуванням колір тіні збігається з кольором тексту. Необов'язковий параметр.

Зрушення по x - зсув тіні по горизонталі щодо тексту. Позитивне значення цього параметра задає зсув тіні вправо, негативне - вліво. Обов'язковий параметр.

Зрушення по y - зсув тіні по вертикалі щодо тексту. Також допустимо використовувати від'ємне значення, яке піднімає тінь вище тексту. Обов'язковий параметр.

Радіус - задає радіус розмиття тіні. Чим більше це значення, тим сильніше тінь згладжується, стає ширше і світліше. Якщо цей параметр не заданий, за замовчуванням встановлюється рівним 0 (рис.4). Врахуйте, що алгоритм згладжування в браузерах зазвичай різний, тому вид тіні може дещо відрізнятися в залежності від заданих параметрів згладжування.

Приклад:

The HTML:

<H1> Text-shadow example </ h1>

The CSS:

```
h1 {
  color: blue;
  font-size: 30pt;
  text-shadow: 5px 2px 4px grey;
}
```

5px - X координата
2px - Y координата
4px - радіус розмиття
Grey - колір тіні

Рисунок 4. Властивість text-shadow

Колір фону: background-color

Визначає колір фону елемента. Хоча ця властивість не буде наслідувати властивості свого батька, через те, що початкове значення встановлюється прозорим, колір фону дочірніх елементів збігається з кольором фону батьківського елемента (рис.5).

Приклад 6.7

The HTML:

```
<P> The background color of this page is set to LightSkyBlue. </ P>
```

The CSS:

```
body {
    background-color: # 87CEFA; }
```

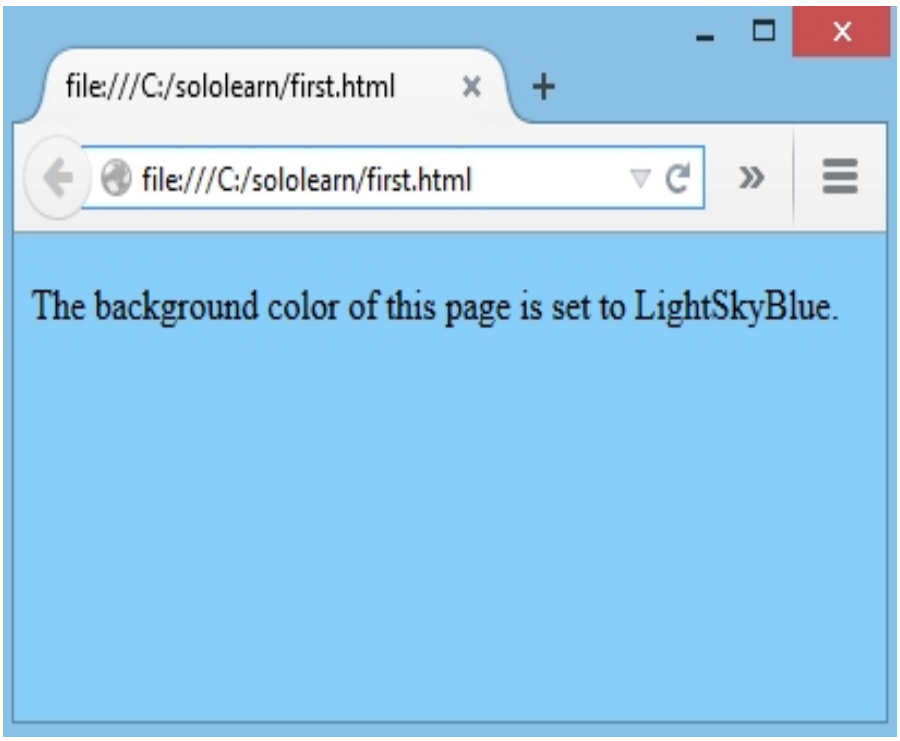


Рисунок 5. Властивість background-color

Фонове зображення: background-image

Встановлює фонове зображення для елемента (рис.6). Якщо одночасно для елемента задано колір фону, він буде показаний, поки фонова картинка не завантажиться повністю. Те ж відбудеться, якщо зображення не доступні або їх показ в браузері відключений. У разі наявності в малюнку прозорих областей, через них буде проглядатися фоновий колір. В CSS3 допустимо вказувати кілька фонових зображень, перераховуючи їх параметри через кому.

Приклад:

```
body {
  background-image: url ( «css_logo.png»);
  background-color: # e9e9e9;
}
```

Рисунок 6. Властивість background-image

Так само у background є безліч інших варіацій, які дають гнучкі налаштування фону.

Всі властивості перераховувати тут і пояснювати немає сенсу: їх дуже багато і за раз не запам'ятаєш. Є ресурси, ми вже розглядали раніше, на яких можна знайти ті властивості, що вам треба і детальну інформацію про них.

Контрольні запитання:

1. Які властивості CSS ви знаєте?
2. Як додати тінь?
3. Як змінити колір фону?
4. Як встановити фонове зображення?

Лекція 5. Основи CSS. Блокова модель

Всі HTML елементи можуть бути представлені як блоки. Блокова модель мови CSS являє собою дизайн і макет сайту. Вона складається з зовнішніх відступів, рамок, внутрішніх полів, і актуального контенту.

Властивості працюють в такому ж порядку: верх, право, низ, і ліво.

У HTML-документі кожному елементу на сторінці відповідає прямокутна область (бокс або блок). Движок рендеринга в браузері визначає розміри і положення боксів на сторінці, а також їх властивості на зразок кольору, фонової картинки для того, щоб відобразити їх на екрані (рис.7).

У мові CSS є спеціальна боксова модель (також блокова модель або блокова модель, англ. Box model), яка описує, з чого складається бокс і які властивості впливають на його розміри.

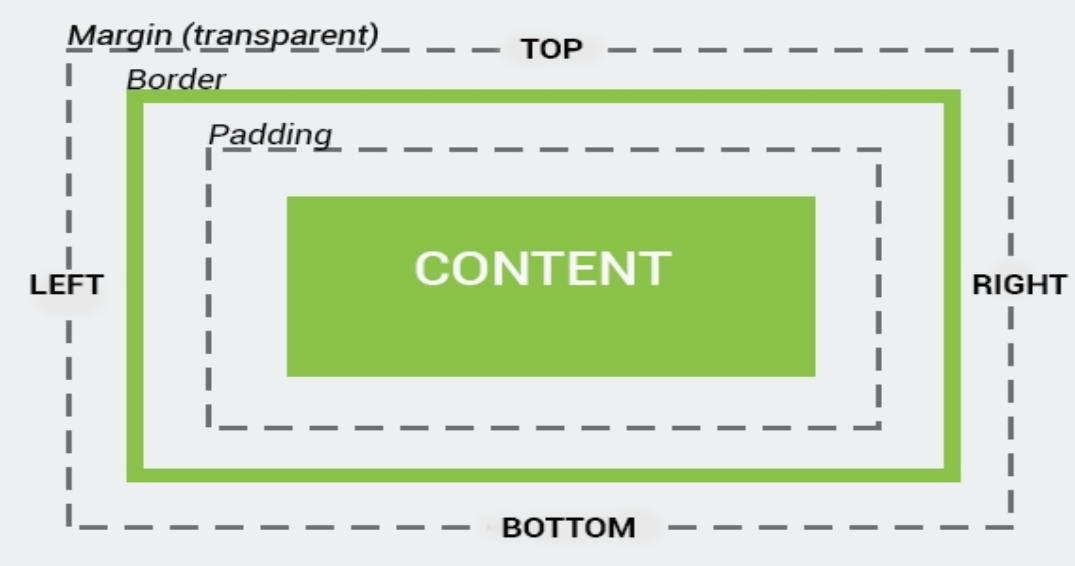


Рисунок 7. Блокова модель

У ній у кожного бокса є 4 області: margin (зовнішні відступи), border (рамка), padding (внутрішні поля), і content (контент або вміст).

Ширина блоку

Ширина блоку - це комплексна величина і складається з декількох значень властивостей:

• width - ширина контенту, тобто вмісту блоку;

• padding-left і padding-right - поле зліва і справа від контенту;

• border-left і border-right - товщина кордону ліворуч і праворуч;

• margin-left і margin-right - відступ зліва і справа.

Як уже згадувалося, якісь властивості можуть бути відсутніми і в цьому випадку на ширину не впливають.

Властивість display

Кожен елемент на WEB сторінці є прямокутним блоком. Властивість display визначає поведінку цих блоків. Елемент блоку - це елемент, який займає повністю доступну ширину, з переносами рядків до і після.

Display:

……..

1Inline

Елемент відображається як вбудований. Використання блокових тегів, таких як і <p>, автоматично створює перенос і показує вміст цих тегів з нового рядка (рис.8). Значення inline скасовує цю особливість, тому вміст блокових елементів починається з того місця, де закінчився попередній елемент.

Приклад:

The HTML:

<Span> First paragraph. </ Span>

<Span> Second paragraph. </ Span>

<Span> Third paragraph. </ Span>

<Span> Fourth paragraph. </ Span>

<Span> Fifth paragraph. </ Span>

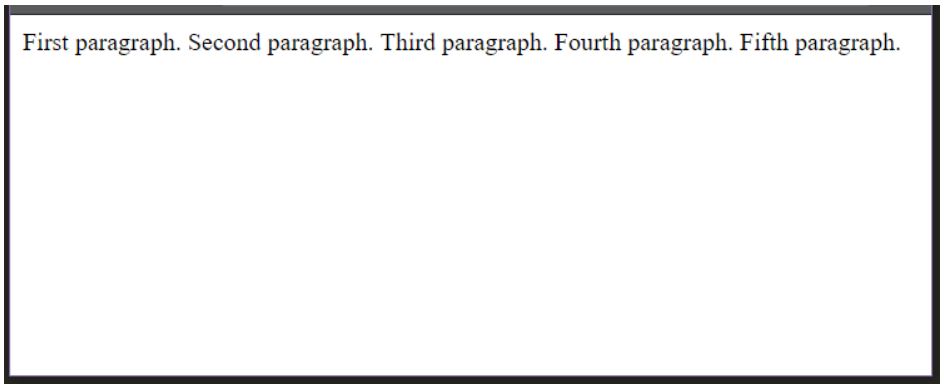


Рисунок 8. Властивість display: inline

Block

Елемент показується як блоковий. Застосування цього значення для вбудованих елементів, наприклад, тега , змушує його вести подібно блокам - відбувається перенесення рядків на початку і в кінці вмісту (рис.9).

Приклад:

The HTML:

<Span> First paragraph. </ Span>

<Span> Second paragraph. </ Span>
<Span> Third paragraph. </ Span>
<Span> Fourth paragraph. </ Span>
<Span> Fifth paragraph. </ Span>
The CSS:

```
span {
   display: block;
}
```

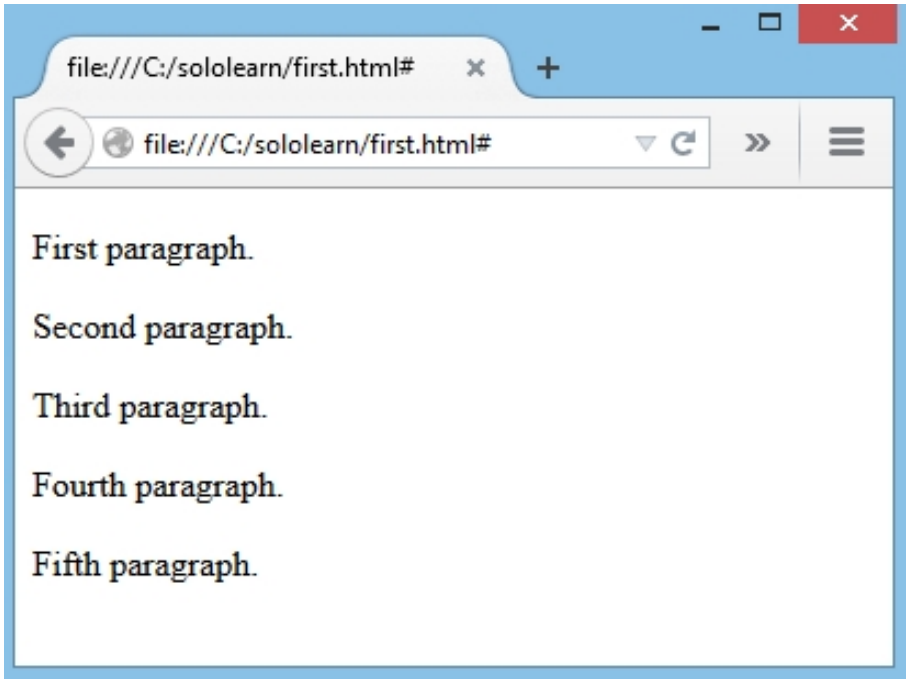


Рисунок 9. Властивість display: block

Inline-block

Це значення генерує блоковий елемент, який обтікається іншими елементами WEB-сторінки подібно вбудованому елементу. Фактично такий елемент по своїй дії схожий на вбудовані елементи (на кшталт тега <img>, рис.10). При цьому його внутрішня частина форматується як блоковий елемент, а сам елемент - як вбудований.

Приклад:
The HTML:
<Span> First paragraph. </ Span>
<Span> Second paragraph. </ Span>
<Span> Third paragraph. </ Span>
<Span> Fourth paragraph. </ Span>
<Span> Fifth paragraph. </ Span>

The CSS:

```
span {
display: inline-block
}
```

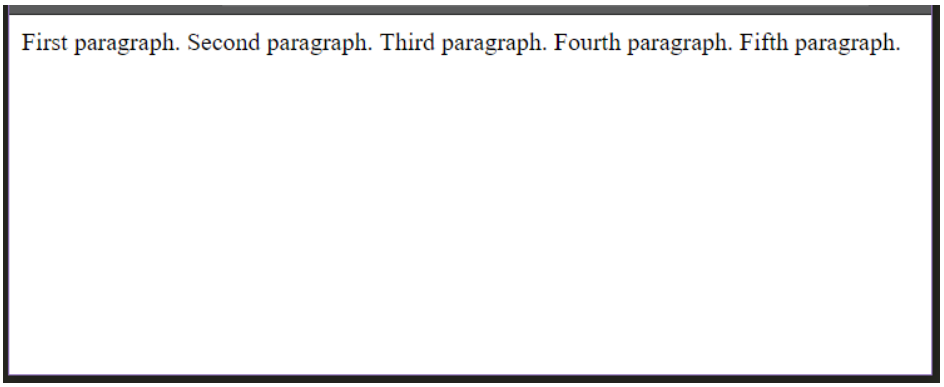


Рисунок 10. Властивість display: inline-block

Обтікання float

З властивістю CSS float, елемент може бути вирівнюються по лівому або правому краях, дозволяючи всім іншим елементам обтікати його навколо.

Float часто використовується з зображеннями, але воно також корисне при роботі з розміткою.

Значеннями властивості float є left, right, і none.

Значення left і right властивості float вирівнюють елементи у відповідних напрямках (по лівому і правому краях). none (за замовчуванням) гарантує, що елемент не буде обтікати.

Нижче, на рис.11, представлений приклад зображення, яке вирівняне по правому краю за допомогою властивості float.

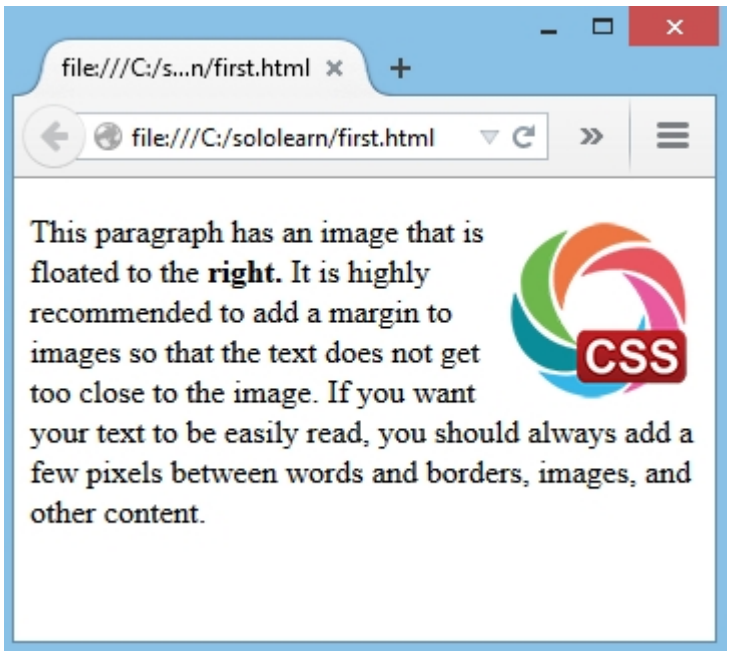


Рисунок 11. Властивість float

Приклад:

The HTML:

<P>

This paragraph has an image that is floated to the right. </ Strong>

It is highly recommended to add a margin to images so that the text does not get too close to the image. If you want your text to be easily read, you should always add a few pixels between words and borders, images, and other content.

</ P>

The CSS:

```
img {
  float: right;
}
```

Властивість clear встановлює, з якого боку елемента заборонено його обтікання іншими елементами. Якщо задано обтікання елемента за допомогою властивості float, то clear скасовує його дію для зазначених сторін.

Властивість Position

Це властивість встановлює спосіб позиціонування елемента щодо вікна браузера або інших об'єктів на WEB-сторінці.

position: absolute | fixed | relative | static | inherit

absolute

Вказує, що елемент абсолютно позиціонується, при цьому інші елементи відображаються на WEB-сторінці, немов абсолютно позиціонованого елемента взагалі немає. Положення елемента задається властивостями left, top, right і bottom, також на положення впливає значення властивості position батьківського елемента. Так, якщо у батька значення position встановлено як static або батька немає, то відлік координат ведеться від краю вікна браузера. Якщо у батька значення position задано як fixed, relative або absolute, то відлік координат ведеться від краю батьківського елементу.

fixed

За своєю дією це значення близьке до absolute, але на відміну від нього прив'язується до зазначеної властивостями left, top, right і bottom точці на екрані і не змінює свого положення при прокручуванні WEB-сторінки. Браузер Firefox взагалі не відображає смуги прокрутки, якщо положення елемента задано фіксованим, і воно не поміщається цілком у вікно браузера. У браузері Opera хоча і показуються смуги прокрутки, але вони ніяк не впливають на позицію елемента.

relative

Положення елемента встановлюється щодо його вихідного місця. Додавання властивостей left, top, right і bottom змінює позицію елемента і зрушує його в ту чи іншу сторону від первісного розташування (рис.12).

static

Елементи відображаються як зазвичай. Використання властивостей left, top, right і bottom не призводить до якихось результатів.

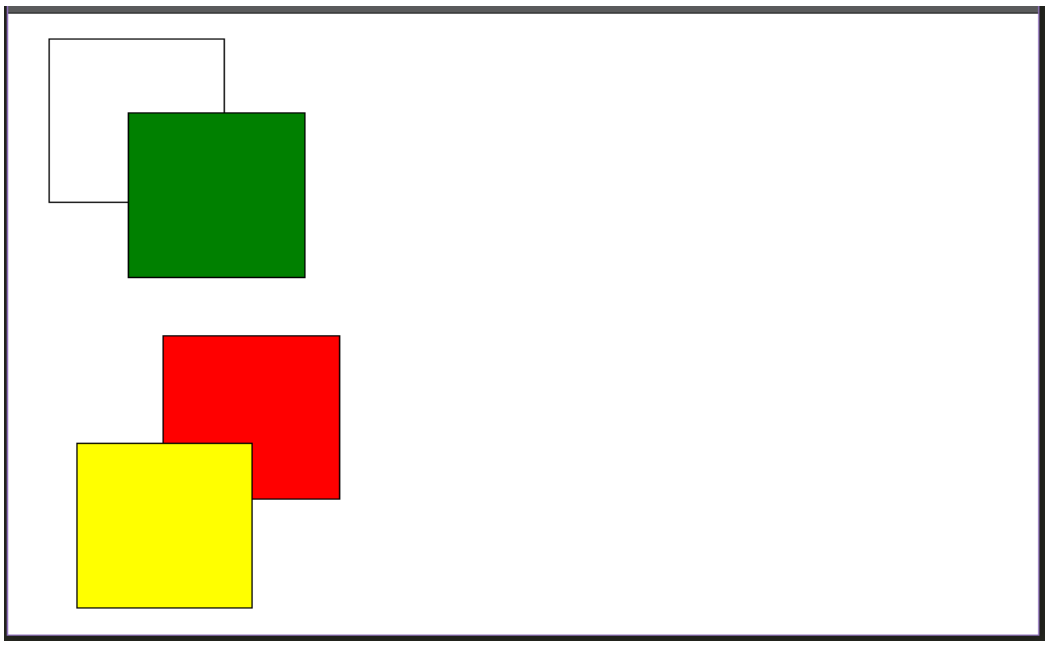


Рисунок 12. Властивість position

Приклад:

The HTML:

```
<Div>
      <Div id = «Green»> </ div>
      <Div id = «Red»> </ div>
      <Div id = «Yellow»> </ div>
  </ Div>
```

The CSS:

```
div {
        border: 1px solid black;
        width: 100px;
        height: 100px;
        margin: 15px;
      }
      #Green {
        top: 30px;
        left: 30px;
background-color: Green;
        position: relative;
      }
      #Red {
        top: 50px;
```

```
    left: 50px;
    background-color: Red;
    position: relative;
}
#Yellow {
    background-color: Yellow;
    position: relative;
}
```

Будь-які позиційовані елементи на WEB-сторінці можуть накладатися один на одного в певному порядку, імітуючи тим самим третій вимір, перпендикулярний екрану. Кожен елемент може перебувати як нижче, так і вище інших об'єктів WEB-сторінки, їх розміщенням по z-осі і управляє z-index. Це властивість працює тільки для елементів, у яких значення position задано як absolute, fixed або relative.

Як значення використовуються цілі числа (позитивні, негативні і нуль). Чим більше значення, тим вище знаходиться елемент в порівнянні з тими елементами, у яких воно менше. При рівному значенні z-index, на передньому плані знаходиться той елемент, який в коді HTML описаний нижче. Хоча специфікація і дозволяє використовувати негативні значення z-index, але такі елементи не відображаються в браузері Firefox до версії 2.0 включно.

Стилізація посилань

Посилання можуть бути стилізовані за допомогою будь-яких CSS властивостей (наприклад color, font-family, background, і т.д.).

Також, посилання можуть бути стилізовані по-іншому, в залежності від того, в якому вони стані. В наявності є такі псевдо-селектори:

a: link - визначає стиль для нормальної не відвіданих посилань.

a: visited - визначає стиль для відвіданих посилань.

a: active - посилання стає активною при натисканні на неї.

a: hover - активізується, коли курсор миші знаходиться в межах елемента.

За замовчуванням, посилання підкреслюються браузером.

Одним з найбільш загальних використань CSS до посилань є видалення нижнього підкреслення. Властивість text-decoration використовується для видалення нижнього підкреслення.

Властивість border

Універсальна властивість border дозволяє одночасно встановити товщину, стиль і колір кордону навколо елемента. Значення можуть йти в будь-якому порядку, розділяючись пробілом, браузер сам визначить, яке з них відповідає потрібній властивості. Для установки кордону тільки на певних сторонах елементу, скористайтеся властивостями border-top, border-bottom, border-left, border-right.

Градієнти

Градієнти CSS3 дозволяють вам встановити колір фону елемента на градієнт. Доступно два типи градієнтів: Linear (лінійний) і Radial (радіальний).

Контрольні запитання:

1. Що таке блокова модель?
2. З чого складається ширина блоку?
3. Які є блокові елементи?
4. Як стилізувати посилання?

Лекція 6. Основи CSS. Нові особливості

CSS3: Нові особливості

Властивість border-radius

За допомогою CSS3, ви можете дати будь-якого елементу element «закруглені кути» використовуючи властивість border-radius (рис.13). Специфічні значення можуть бути застосовані для властивості border-radius в наступному порядку: верх-ліво, верх-право, низ-право, низ-ліво.

Приклад:

```
  div {
border-radius: 20px;
background-color: green;
color: white;
padding: 50px;}
```

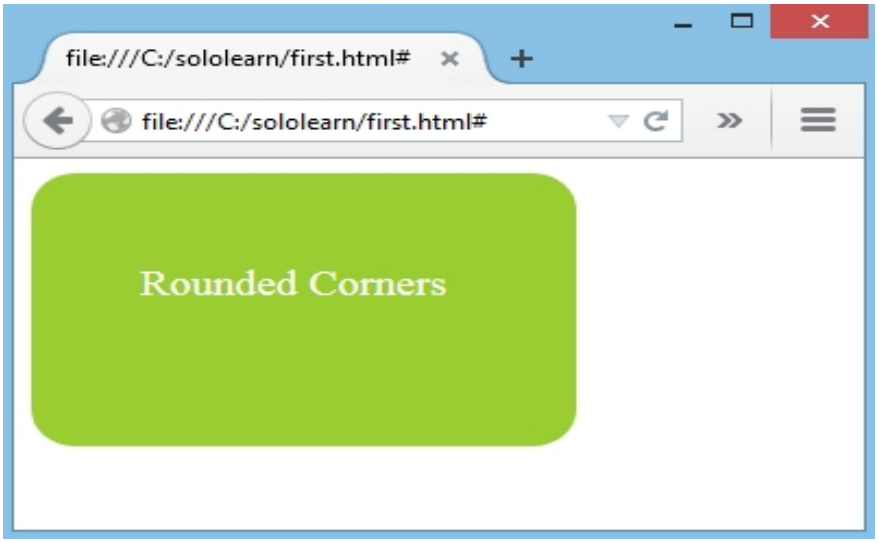


Рисунок 13. Властивість border-radius

Створення окружності

Прямокутник може бути перетворений в коло за допомогою CSS.

Щоб створити коло, радіус рамки повинен бути рівний половині висоти і ширини.

Прямокутник у наведеному зразку має ширину і висоту рівні 200px. Встановивши радіус рамки рівним 100px, кути будуть округлені для формування кола:

Приклад:

```
div {
color: green;
width: 200px;
height: 200px;
border-radius: 100px;}
```

Властивість box-shadow

CSS3 властивість box-shadow застосовує до елементів тінь.

Компоненти властивості box-shadow декодуються браузерами в наступній манері (рис.14):

- перша довжина для горизонтального зміщення викличе тінь праворуч від блоку (значення обов'язкове)

- друга довжина для вертикального зсуву викличе тінь під блоком (значення обов'язкове)

- колір тіні (значення опціональне).

Приклад:

HTML:
<Div> </ div>
CSS:
div {
width: 300px;
height: 100px;
background-color: # 9ACD32;
box-shadow: 10px 10px # 888888;
}

Рисунок 14. Властивість box-shadow

Працюємо з псевдокласом

Псевдокласи CSS дозволяють нам стилізувати елементи, або частини елементів, які існують в дереві документа без використання JavaScript або інших скриптів. Псевдокласи починаються з «:» (двокрапки).

Найбільш часто використовуваними псевдоклас є: first-child і: last-child.

Псевдоклас: first-child знаходить будь-який елемент, який є першим у свого батька.

Псевдокаласс: hover визначає стиль елемента при наведенні на нього курсора миші, але при цьому елемент ще не активований, іншими словами кнопка миші не була натиснута (раніше ми застосовували його до стилізації посилання).

Працюємо з псевдоелементами

Псевдоелементи використовуються для вибору специфічних частин елементу.

Є п'ять псевдоелементів в CSS, кожен починається з подвійного двокрапки (: :):

:: first-line - перший рядок тексту селектора

:: first-letter - перша буква тексту в селекторі

:: selection - вибирає частину тексту, яка вказана користувачем

:: before - вставляє деякий вміст до елемента

:: after - вставляє деякий вміст після елемента

Переходи CSS3

Переходи CSS3 дозволяють нам змінювати одне значення властивості на інше на протязі заданої тривалості.

transition-property - вказує властивість, до якого буде застосований перехід

transition-duration - визначає тривалість, протягом якої відбудеться перехід

transition-timing-function - визначає, як буде змінюватися темп переходу під час його тривалості

transition-delay - визначає затримку (в секундах) для ефекту переходу

Метод scale ()

Метод scale () збільшує або зменшує розмір елемента, відповідно до заданих параметрів ширини і висоти. 1 відповідає оригінальному розміру, 2 відповідає подвоєному оригінальному розміру тощо.

Контрольні запитання:

1. Що таке нові особливості CSS?
2. Як створити окружність?
3. Що таке псевдокласс?
4. Що таке псевдоелементи?
5. Що таке метод Метод scale ()?

Список літератури

Базова

1. Бутенко В. М., Павленко Є. П. Інженерія програмного забезпечення. WEB програмування: навч. посібник. – Харків: УкрДУЗТ, 2019. – 127 с.
2. Баран. С.В. Основи web-програмування: навчальний посібник. – Кривий Ріг: Державний університет економіки і технологій, 2023. – 316 с.
3. Двірничук К.В., Вацек Д.О. Веб-програмування та веб-дизайн: навч. посіб. – Чернівці: Чернівец. нац. ун-т ім. Ю. Федьковича, 2022. – 472 с.
4. Грицюк Ю.І. Аналіз вимог до програмного забезпечення. – Львів: Вид-во НУ “Львівська політехніка”, 2018. – 456 с.
5. Мартін Р. Чистий код. – Харків: Фабула, 2019. – 416 с.

Інформаційні ресурси

1. Основи WEB-програмування. Мова HTML та CSS. | Інші методичні матеріали. Інформатика [Електронний ресурс]. – Режим доступу: http://vseosvita.ua
2. Офіційний сайт системи MOODLE [Електронний ресурс]. – Режим доступу: http://www.moodle.org
3. Moodle Statistics // Moodle. [Електронний ресурс]. – Режим доступу: http://moodle.org/stats

Навчальне видання

ХРОЛЕНКО Володимир Миколайович
ГОЛЕНКОВ Володимир Геннадійович

WEB-ПРОГРАМУВАННЯ

Конспект лекцій