ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
“ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ”
МІНІСТЕРСТВА ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ
Кафедра математичного моделювання

Лісняк А.О., Чопоров С.В.

Навчально-методичний посібник до лабораторних занять з курсу «Технології та протоколи Інтернет»

для студентів, що навчаються за освітньо-професійним напрямом «Програмна інженерія»

Затверджено
вченою радою ЗНУ
Протокол № __
від «___» _____20__

Запоріжжя 2015

УДК

Методичні вказівки до виконання лабораторних робіт з дисципліни «Технології та протоколи Інтернет» для студентів спеціальності 6.040301 – «Програмна інженерія». Укладачі: Лісняк А.О., Чопоров С.В. – Запоріжжя: ЗНУ, 2015. – __с.

Методичні вказівки містять загальні теоретичний матеріал, а також завдання для виконання лабораторних робіт з курсу «Технології та протоколи Інтернет».

Методичні вказівки призначені для викладачів, студентів, а також усіх, хто цікавиться питаннями зазначеної тематики.

Рецензент

Відповідальний за випуск

Зміст

Лабораторна робота №1. Реєстрація хостингу

Мета: реєстрація налаштування та тестування працездатності основних Web-сервісів безкоштовного хостингу.

Теоретичні відомості

Хостинг (англ. hosting) - послуга з надання обчислювальних потужностей для фізичного розміщення інформації на сервері, що постійно перебуває в мережі (зазвичай Інтернет). Хостингом також називається послуга по розміщенню обладнання клієнта на території провайдера із забезпеченням підключення його до каналів зв'язку з високою пропускною здатністю.

Зазвичай під поняттям послуги хостингу мають на увазі як мінімум послугу розміщення файлів сайту на сервері, на якому запущене ПЗ, необхідне для обробки запитів до цих файлів (веб-сервер). Як правило, в послугу хостингу вже входить надання місця для поштової кореспонденції, баз даних, DNS, файлового сховища і т. п., а також підтримка функціонування відповідних сервісів.

Хостинг баз даних, розміщення файлів, хостинг електронної пошти, послуги DNS можуть надаватися окремо як самостійна послуга, або входити в поняття послуги.

Одним з важливих критеріїв вибору хостингу є використовувана операційна система, оскільки від цього залежить програмне забезпечення, яке буде підтримувати функціональність тих або інших сервісів. Важливим аспектом опису хостингу є наявність тих чи інших служб і можливостей:

- підтримка CGI: Perl, PHP, Python, ASP, Ruby, JSP;
- підтримка . htaccess/. htpasswd (для Apache);
- підтримка баз даних.

Хостинг як послугу порівнюють і описують по наступним обмеженням:

- розмір дискового простору;
- кількість місячного трафіку;
- кількість сайтів, які можна розмістити в рамках одного облікового запису;
- кількість FTP користувачів;
- кількість E-Mail ящиків і обсяг місця, призначеного для пошти;
- кількість баз даних і кількість місця під бази даних;
- кількість одночасних процесів на користувача;
- кількість ОЗУ, і максимальний час виконання що виділяється кожному процесу користувача;

Деякі платні хостингові компанії надають безкоштовний тест на певний період, після закінчення якого користувач повинен визначитися чи підходить для нього вибрана хостингова компанія, і чи має сенс оплачувати великі періоди. Такі

тести надаються тільки власникам доменів другого рівня, щоб уникнути спекуляцій з тестовими акаунтами.

Крім платних хостерів існують також і безкоштовні хостинг компанії, що підтримують більшість описаних технологій та сервісів.

Завдання

1. Зареєструватися на доступному безкоштовному хостингу.
2. Налагодити можливість доступу до панелі керування хостингом та доступу до власних файлів з використанням протоколу FTP.
3. Створити html-документ (index.html), що містить інформацію наступного змісту:
   - ПІБ;
   - номер академічної групи;
   - назва дисципліни;
4. Переглянути документ index.html як локальний файл у будь-якому браузері.
5. Створити директорію lab1 у кореневій директорії хостингу та скопіювати туди файл index.html.
6. Перевірити можливість доступу до документа index.html на хостингу.
7. Результати виконання лабораторної роботи розташувати у власному репозиторії будь якої відкритої системи контролю версій[1] з тегом lab1.
8. Предметами захисту роботи є вихідний текст у репозиторії системи контролю версій та загальнодоступний ресурс, що відображає результат виконання лабораторної роботи.

Контрольні питання

1. Що таке хостинг?
2. Які основні сервіси хостингу?
3. Яким чином можливо отримати доступ до файлів на віддаленому сервері?
4. Які функцій виконує системи контролю версій?
5. Які основні команди роботи із системою контролю версій github/bitbucket?

[1] https://github.com або https://bitbucket.org

Лабораторна робота №2. Основи HTML

Мета: опанувати основні можливості форматування та оформлення текстових матеріалів з використання HTML.

Теоретичні відомості

HTML (https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0 HyperText Markup Language) – стандартна мова розмітки веб-сторінок в Інтернеті. Більшість веб-сторінок створюються за допомогою мови HTML (або XHTML). Документ HTML опрацьовується браузером та відтворюється на екрані у звичному для людини вигляді.

HTML разом із *каскадними таблицями стилів (CSS)* та *вбудованими скриптами* – три основні технології побудови веб-сторінок.

В HTML версії 4 завдяки використанню CSS можливо створювати сторінки з добре зрозумілою для користувачів візуальної структурою, але мало «зрозумілою» для пошукових систем або браузерів. Типова структура документа в HTML4 представлена на рис. 2.1.

Рисунок 2.1 – Структура типового документа HTML4

Для вирішення цієї проблеми в HTML5 були введені *семантичні теги,* за допомогою яких сторінки сайтів стають більш «зрозумілими» для пошукових систем і браузерів. Так на HTML5 структура документа прийме інший вигляд (рис. 2.2).

Рисунок 2.2 – Структура типового документа HTML5

Теги HTML4

В таблиці 2.1 представлені теги і їх короткий опис мови розмітки 4-ї версії[2]. У колонці статус вказано які зміни відбулися з даним елементом в HTML5.

Таблиця 2.1 – Основные теги HTML4

Назва тегу	**Опис**	**Статус**
<!--....-->	Визначає коментар	-
<!DOCTYPE>	Визначає тип документа	змінений
<a>	Визначає посилання	змінений
<abbr>	Визначає абревіатуру	-
<address>	Визначає контактну інформацію автора документа	змінений
<area />	Визначає область-посилання на зображення	змінений
<b>	Визначає жирний текст	-

[2] Більш детальний опис синтаксису, атрибутів і призначення кожного з тегів – http://htmlbook.ru/

<base />	Визначає адресу або спосіб відкриття всіх посилань сторінці за замовчуванням	-
<bdo>	Визначає напрямок тексту	-
<blockquote>	Визначає довгу цитату.	-
<body>	Визначає тіло документа	змінений
 	Визначає перенесення на новий рядок	-
<button>	Визначає кнопку	змінений
<caption>	Визначає заголовок таблиці	змінений
<cite>	Оформляє текст як цитату	-
<code>	Оформляє текст як комп'ютерний код	-
<dd>	Визначає значення терміна в списку визначень	-
<del>	Визначає закреслений текст	-
	Визначає секцію в документі	-
<dl>	Визначає список визначень	-
<dt>	Визначає терміни в списку визначень	-
	Визначає акцентований текст	-
<fieldset>	Визначає межу навколо елементів форми.	змінений
<form>	Визначає форму для введення даних користувача	змінений
<h1> - <h6>	Определяет заголовки.	-
<head>	Определяет справочную информацию о документе.	-
<hr />	Определяет горизонтальную линию.	змінений
<html>	Определяет корневой тэг HTML5 документа.	змінений
<i>	Определяет курсивный текст.	-
<iframe>	Определяет строковый фрейм.	змінений
<img />	Определяет картинку.	-
<input />	Определяет поля ввода данных в форме.	змінений

<ins>	Определяет вложенный текст.	-
<label>	Определяет метку для элемента input.	змінений
<legend>	Определяет заголовок для элементов fieldset, figure, и details.	змінений
	Определяет список.	-
<link />	Определяет взаимосвязь между текущим документом и внешним файлом.	-
<map>	Определяет карту изображений.	змінений
<meta />	Определяет метаданные HTML5 документа.	змінений
<noscript>	Определяет альтернативное содержимое для пользователей браузер которых не поддерживает клиентские скрипты.	-
<object>	Определяет встраиваемый объект.	змінений
	Определяет упорядоченный список.	змінений
<optgroup>	Определяет группу похожих элементов в списке выбора.	-
<option>	Определяет вариант в списке выбора.	-
<p>	Определяет абзац.	-
<param />	Определяет параметр для встраиваемого объекта.	змінений
<pre>	Определяет отформатированный текст.	-
<script>	Определяет клиентский скрипт.	змінений
<select>	Определяет выпадающий список.	змінений
<small>	Определяет маленький текст.	-
	Определяет секцию в документе.	-
	Определяет важный текст.	змінений
<style>	Определяет информацию о стиле документа.	змінений
<sub>	Определяет нижний индекс элемента.	-
<sup>	Определяет верхний индекс элемента.	-

<table>	Определяет таблицу.	змінений
<tbody>	Группирует содержимое тела таблицы.	змінений
<td>	Определяет ячейку в таблице.	змінений
<textarea>	Определяет многострочное поле для ввода текста.	змінений
<tfoot>	Группирует нижнее содержимое таблицы.	змінений
<th>	Определяет заголовочную ячейку в таблице.	змінений
<thead>	Группирует головное содержимое таблицы.	змінений
<title>	Определяет заголовок документа.	-
<tr>	Определяет строку в таблице.	змінений
	Определяет неупорядоченный список.	-

Теги HTML5

В таблице 2.2 представлены только тэги, которые были добавлены в HTML5[3]. В колонке «Поддержка»[4] отображено состоянии поддержки данного тэга в современных браузерах. Возможные значения:

- Полная – тэг поддерживается всеми современными браузерами;
- Частичная – тэг частично поддерживается современными браузерами;
- Отсутствует – тэг не поддерживается ни одним современным браузером.

Таблица 2.2 – Основные теги HTML5.

Название тэга	**Описание**	**Поддержка**
<article>	Определяет независимое содержимое страницы.	Полная
<aside>	Определяет косвенно связанные с содержимым элементы.	Полная
<audio>	Встраивает аудио файл.	Полная

[3] Более детальное описание синтаксиса, атрибутов и назначения каждого из тегов – http://htmlbook.ru/

[4] Информация в таблице актуальна на момент ее составления и может изменится через некоторый промежуток времени.

<canvas>	Позволяет рисовать произвольные объекты с помощью скриптов.	Полная
<datalist>	Определяет список с данными для элемента ввода.	Частичная
<details>	Определяет поле с дополнительной информацией, которое пользователь может скрывать или отображать.	Частичная
<embed>	Позволяет встроить внешнее содержимое.	Полная
<figcaption>	Определяет подпись для изображения.	Полная
<figure>	Используется для создания подписей для изображений.	Полная
<footer>	Определяет футер для документа.	Полная
<header>	Определяет заголовочный блок для документа.	Полная
<hggroup>	Позволяет сгруппировать заголовки.	Полная
<keygen>	Генерирует открытый и закрытый ключ для безопасной связи с сервером.	Частичная
<mark>	Подсвечивает (выделяет) текст.	Частичная
<meter>	Определяет полосу измерения.	Частичная
<menu>	Позволяет определять контекстные меню и панели инструментов.	Частичная
<main>	Призначений для основного вмісту документа, який має бути унікальним і не включати типові блоки (шапки сайту, кінцевик, навігації, бічні панелі, форми пошуку і т. п).	Частичная
<nav>	Определяет навигационный блок сайта.	Полная
<output>	Используются для вывода данных.	Частичная
<progress>	Отображает состояние выполнения какого-либо процесса.	Частичная
<rp>	Указывает, что должно отображаться в браузерах не поддерживающих агаты.	Полная

<rt>	Позволяет задать агаты (используются для уточнения чтения иероглифов в китайском и японском языке).	Полная
<ruby>	Позволяет задать иероглиф, прочтение которого необходимо уточнить.	Полная
<section>	Позволяет сгруппировать логически связанное содержимое.	Полная
<source>	Позволяет задать несколько источников воспроизведения для элементов <audio> и <video>.	Полная
<summary>	Определяет видимый заголовок для элемента <details>.	Частичная
<time>	Определяет дату и время.	Отсутствует
<track>	Позволяет добавить субтитры к элементам <audio> и <video>.	Отсутствует
<video>	Позволяет добавлять на веб-страницы видео файлы.	Полная
<wbr>	Отмечает подходящее место в тексте для возможного переноса.	Частичная

Удаленные тэги

В таблице 2.3 располагаются тэги, которые присутствовали в HTML4, но в HTML5 были удалены или считаются устаревшими.

Таблица 2.2 – Теги, которые считаются устаревшими в HTML5.

Название тэга	**Описание**
<acronym>	Определяет акроним.
<big>	Определяет большой текст.
	Выравнивает текст по центру.
<dir>	Позволяет задать список каталогов.

<font>	Позволяет задать тип, размер и цвет шрифта.
<frame />	Определяет фрейм (окно) в наборе фреймов.
<frameset>	Определяет набор фреймов.
<noframes>	Определяет альтернативное содержимое для пользователей браузер которых не поддерживает фреймы.
<strike>	Определяет зачеркнутый текст.

Правильна структура документа HTML5:

```
<!DOCTYPE HTML>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <title>Ваш сайт</title>
</head>
<body>
   <header>
    <nav>
     <ul>
       <li>Пункт меню 1</li>
       <li>Пункт меню 2</li>
       <li>Пункт меню 3</li>
     </ul>
    </nav>
   </header>
   <section>
   <h1>Статті на сайті</h1>
      <article>
        <header>
          <h2>Заголовок статьи</h2>
          <p>Опубликовано
             <time datetime="2014-01-30T16:31:24+03:00">30.01.2014</time>
          </p>
        </header>
        <p>Давно выяснено, что при оценке дизайна и композиции читаемый текст
мешает сосредоточиться.
  </p>
      </article>
      <article>
         <header>
          <h2>Заголовок статьи</h2>
            <p>Опубликовано
        <time datetime="2014-01-30T16:31:24+03:00">30.01.2014</time>
      </p>
         </header>
         <p>Многие программы электронной вёрстки и редакторы HTML используют Lorem
Ipsum в качестве текста по умолчанию.
    </p>
       </article>
   </section>
   <aside>
      <h2>Об авторе</h2>
        <p>Нет никого, кто любил бы боль саму по себе, кто искал бы её и кто хотел
бы иметь её просто потому, что это боль..</p>
```

```
   </aside>
<footer>
    <p>Copyright 2014 Ваш сайт</p>
</footer>
</body>
</html>
```

Завдання

1. Використовуючи текстовий редактор[5], створити набір документів:
   - index.html
   - rozklad.html
   - news.html
   - photo.html

2. Кожна з сторінок повинна мати однакову структура з використанням відповідних семантичних елементів HTML5.

3. Навігація кожної з сторінок повинна мати посилання на всі інші документи перелічені у пункті 1.

4. Макет сторінки index.html повинен відповідати макету наведеному на рис. 2.3 та використовувати як можна більше тегів html для оформлення тексту (цитати, посилання, примітки, переліки і таке інше).

5. Сторінка rozklad.html повинна містити таблицю з розкладом занять поточного семестру та повний маркований перелік викладачів, що викладають дисципліни у Вашій академічній групі. Макет сторінки наведено на рис. 2.3.

6. Сторінка photo.html повинна містити мініатюри 8-10 зображень (усі однакового розміру) з підписами (назва та розмір оригіналу). Кожна з фото повинна бути посиланням на оригінал зображення, який відкривається у окремому вікні браузера. Макет сторінки наведено на рис. 2.4.

7. Сторінка news.html повинна містити анонси (зображення, дата публікації, короткий опис, посилання на повну версію) 5-8 статей-новин (з популярних порталів новин). Посилання повинне відкривати повну версію матеріалу на сторонньому ресурсі. Макет сторінки наведено на рис. 2.4

8. Пов’язані графічні матеріали розташувати поруч з html-документами у окремій директорії – images.

9. Розташувати створені сторінки та зображення на хостингу у окремій директорії (lab2).

10. Виконати перевірку[6] (валідацію) сторінок на відповідність стандартам HTML5.

11. Результати виконання лабораторної роботи розташувати у власному репозиторії будь якої відкритої системи контролю версій з тегом lab2.

---

[5] Редактер без можливостейатоматичної генерації HTML, наприклад (notepad, notepad++, Sublime Text та подібні)
[6] Наприклад, ресурс http://www.validome.org/

12. Предметами захисту роботи є вихідний текст у репозиторії системи контролю версій та загальнодоступний ресурс, що відображає результат виконання лабораторної роботи.

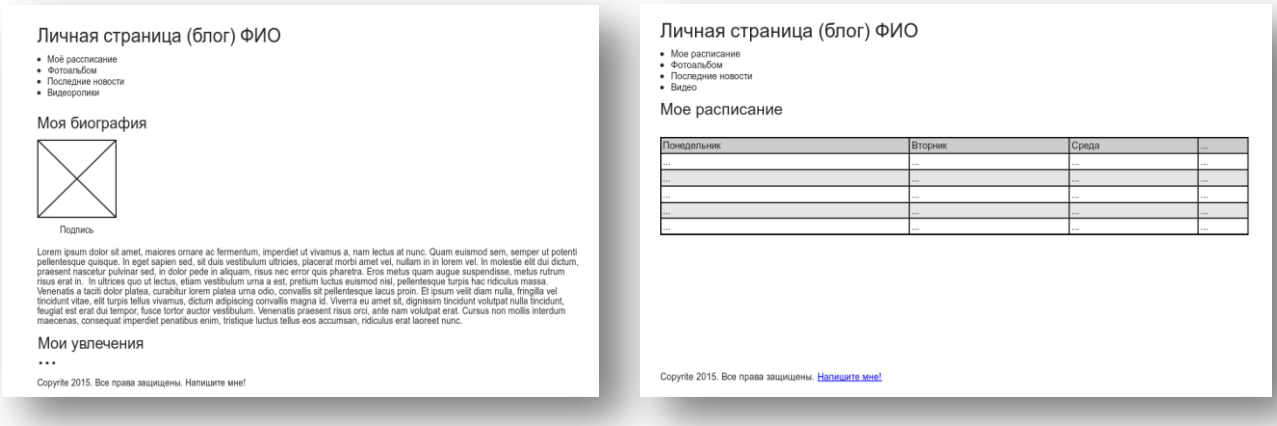


Рисунок 2.3 – Оформлення тексту та таблиць.

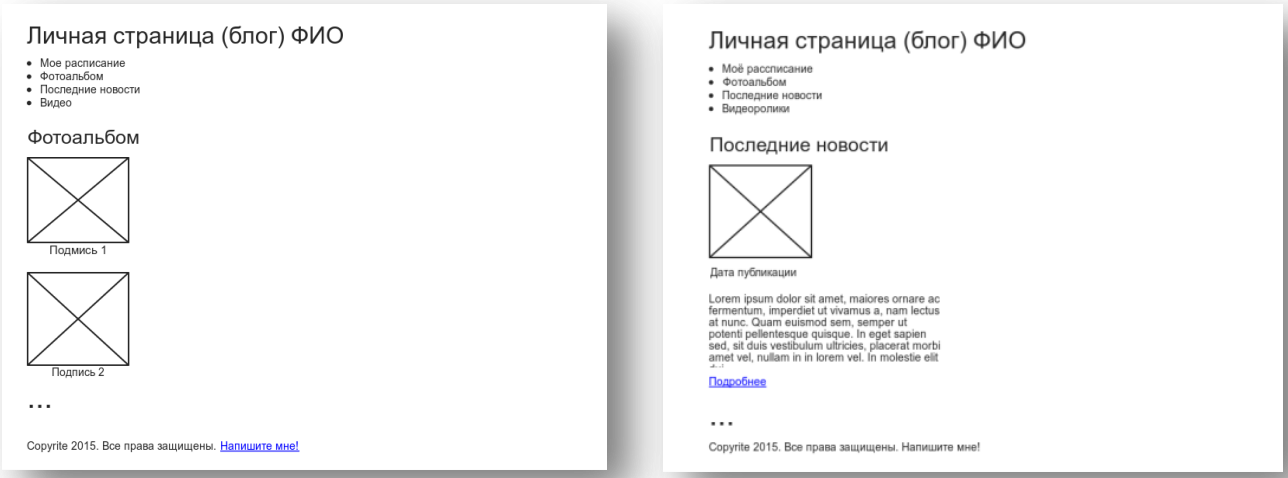


Рисунок 2.4 – Оформлення зображень та новин.

Контрольні питання

1. Наведіть приклади тегів, що є застарілими у HTML5.
2. Які правила використання тегу <main>?
3. Перелічіть основні теги для визначення семантичної структури документа HTML5.
4. Як браузер визначає версію HTML документа?
5. Які особливості документів XHTML?

Лабораторна робота №3. HTML шаблонізатор

Мета: опанувати основні можливості форматування та оформлення текстових матеріалів з використання щаблонізаторів jade та pubjs.

Теоретичні відомості

npm (аббр. node package manager) — это стандартный менеджер пакетов, автоматически устанавливающийся вместе с Node.js. Он используется для скачивания пакетов из облачного сервера npm, либо для загрузки пакетов на эти сервера.

- Файл package.json содержит в себе информацию о вашем приложении: название, версия, зависимости и тому подобное. Любая директория, в которой есть этот файл, интерпретируется как Node.js-пакет, даже если вы не собираетесь публиковать его.
  Способ использования файла package.json зависит от того, собираетесь ли вы скачивать пакет или публиковать его.
- Чтобы скачать пакет вручную, вам необязательно использовать для этого package.json. Вы можете выполнить в терминале команду npm с названием нужного пакета в качестве аргумента команды, и пакет будет автоматически скачан в текущую директорию. Например:
  $ npm install canvas-chart

Yarn - это новый менеджер пакетов, совместно созданный Facebook, Google, Exponent и Tilde. Как можно прочитать в официальной документации, его целью является решение целого ряда проблем, с которыми столкнулись разработчики при использовании npm, а именно:

установка пакетов не была достаточно быстрой и последовательной;

существовали проблемы с безопасностью, так как npm позволяет пакетам запускать код при установке.

Главное отличие **npm** и **Bower** — подход к установке зависимостей пакетов.

npm устанавливает зависимости для каждого пакета отдельно, в итоге получается большое дерево пакетов (node_modules/grunt/node_modules/glob/node_modules/…), где может быть несколько версий одного и того же пакета. В клиентском Яваскрипте это недопустимо: нельзя подключить на страницу две версии jQuery или любой другой библиотеки.

В Бовере каждый пакет устанавливается один раз (jQuery всегда будет в папке bower_components/jquery, сколько бы пакетов от него не зависело) и в случае конфликта зависимостей, Бовер просто не станет устанавливать пакет, не совместимый с уже установленными.

Установка Бовера

- Для работы с Бовером вам потребуются Node и Git. Установка:
  npm install -g bower
  Работа с пакетами
  Попробуем что-нибудь установить, например, jQuery:
  bower install --save jquery # Или bower i -S jquery

Завдання

1. Сворити власний пакет npm.
2. Створити файлову структуру та сгенерувати файл збирача проектів (gulp[7]).
3. Перетворити html-текст сторінок з попередньої лабораторної роботи на код шаблонізатору[8].
4. Створити загальний щаблон сторінок
5. Створити необхідні та корисні заготовки (mixin)
6. «Зібрати» проект
7. Результати виконання лабораторної роботи розташувати у власному репозиторії будь якої відкритої системи контролю версій[9] з тегом lab3.
8. Предметами захисту роботи є вихідний текст у репозиторії системи контролю версій та загальнодоступний ресурс, що відображає результат виконання лабораторної роботи.

Контрольні питання

[7] https://steelydylan.github.io/gulp-generator/#
[8] http://html2jade.org/
[9] https://github.com або https://bitbucket.org

Лабораторна робота №4. Основи CSS.

Мета: опанувати основні можливості стилізації веб-сторінок за допомогою CSS.

Теоретичні відомості

CSS (Cascading Style Sheets — Каскадні таблиці стилів) – спеціальна мова, що відповідає за зовнішній вигляд HTML-сторінок.

CSS має різні рівні та профілі. Наступний рівень CSS створюється на основі попередніх, додаючи нову функціональність або розширюючи вже наявні функції. Рівні позначаються як CSS1, CSS2 та CSS3.

Профіль CSS – сукупність правил CSS одного або більше рівнів, створені для окремих типів пристроїв або інтерфейсів. Наприклад, існують профілі CSS для принтерів, мобільних пристроїв тощо.

Синтаксис мови досить простий: він складається з селекторів та властивостей.

Стилі складаються зі списку правил. Кожне правило має один або більше селекторів та блок визначення. Блок визначення складається із оточеного фігурними дужками списку властивостей.

Найпростіші селектори – це селектори по іменах тегів. З їх допомогою можна задати стилі для всіх абзаців на сторінці, для всіх посилань, заголовків першого рівня і так далі. Таги селектори містять ім'я тега без символів < та >. Наприклад:

```
/* Визначаємо червоний колір та розмір шрифту для всіх тегів
<p> на сторінці */
p {
     color: red;
     font-size:
}
```

До більш складних селекторів можна віднести селектори класів, ідентифікаторів, псевдокласів, контекстні селектори, селектори псевдоелементів та ідентифікаторів.

Селектори класів

Клас дозволяє об'єднувати різні елементи в смислові групи і застосовувати до них однакове оформлення. Наприклад, можна створити клас «елементи з помилкою» і задати йому червоний колір тексту. Потім можна додавати цей клас до будь-якого HTML-тегу: абзацу, заголовку, елементу списку і так далі.
Клас тега можна задати за допомогою атрибуту class, який містить ім'я класу (або імена класів через пробіл). Наприклад:

```
<p class = "help">…</ p>
<p class = "help error">…</p>
```

Селектор з використанням класу задається так: .імя_класса. Наприклад:

```
.help {...}
.error {...}
```

Контекстні селектори

Селектор може складатися з декількох частин, розділених пропуском, наприклад:

```
p strong {...}
ul .hit {...}
.footer .menu a {...}
```

Такі селектори називають **контекстними** або **вкладеними**. Їх використовують для того, щоб застосувати стилі до елементу, тільки якщо він вкладений в потрібний елемент.

Наприклад, селектор *.menu a* спрацює для посилання a тільки в тому випадку, якщо вона розташована всередині елемента з класом *.menu*.

Читати їх найпростіше справа наліво:

```
/* Вибрати все теги strong всередині тегів p */
p strong {...}

/* Вибрати всі елементи з класом .hit всередині тегів ul */
ul .hit {...}

/* Вибрати всі посилання всередині елементів з класом .menu,
   які лежать всередині елементів з класом .footer */
.footer .menu a {...}
```

Таким чином, можна задавати елементам різні стилі в залежності від їх контексту. Якщо посилання розташована всередині меню, зробити її більше, а якщо всередині основного тексту, то задати їй потрібний колір.

Сусідні селектори

Контекстні селектори використовуються для вкладених один в одного елементів, а **сусідні** – для розташованих поруч.Наприклад, теги <li> в списку є сусідніми по відношенню один до одного і вкладеними в тег <ul>.

Сусідні селектори записуються за допомогою знаку +. Наприклад:

```
селектор1 + селектор2
```

Стилі застосуються до елементу, відповідному до селектор2, тільки якщо відразу перед ним розташований елемент, який підходить до селектор1. Наприклад є два елементи списку:

```
<li class = "hit"> </ li>
<li class = "miss"> </ li>
```

Селектор *.hit + .miss* застосує стилі до елементу з класом *miss*, так як перед ним є елемент з класом *hit*.

Селектор *.hit + li* теж застосує стилі до елементу з класом *miss*, а селектор *.miss + .hit* не спрацює.

Дочірні селектори

Нащадком називаються будь-які елементи, розташовані всередині батьківського елементу. А **дочірніми елементами** називаються найближчі нащадки. Наприклад:

```
<ul>
  <li> <span> ... </ span> </li>
  <li> <span> ... </ span> </li>
</ul>
```

Елементи <ul> та <li> є дочірніми елементами і нащадками, а – нащадки, але не дочірні елементи.

Контекстні селектори впливають на всіх нащадків, що не завжди зручно. Іноді необхідно задати стилі тільки для дочірніх елементів. Особливо це корисно при роботі з багаторівневими списками.

Для цього існує дочірній селектор, в якому використовується символ >. Наприклад:

```
ul > li { … }
ul > li> span { … }
```

Псевдокласи

Псевдокласи – це доповнення до звичайних селекторам, які роблять їх ще точнішими та потужнішими. Псевдоклас додається до селектора с допомогою символу «:» без пропусків. Наприклад:

```
a:visited {...}
li:last-child {...}
.alert:hover {...}
```

Переліку основних псевдокласів[10]:

– **:first-child** – дозволяє вибрати перший дочірній елемент;

– **:last-child** – дозволяє вибрати останній дочірній елемент;

– **:link** – дозволяє вибрати ще не відвідані посилання;

– **:visited** – дозволяє вибрати відвідані посилання;

– **:active** – дозволяє вибрати активні посилання (кнопка миші затиснута);

– **:hover** – дозволяє вибрати елемент, коли на нього наведений курсор миші;

– **:nth-child** – використовується для додавання стилю до елементів на основі нумерації в дереві елементів;

– **:focus** – дозволяє вибрати елемент, який в даний момент у фокусі;

Псевдоелементи

Псевдоелементи – це особливий вид селекторів, які дозволяють працювати не над самим елементом, а над його окремою частиною.

Перелік основних псевдоелеметов:

---

[10] Повний перелік та приклади використання псевдокласів http://htmlbook.ru/css

– **:first-letter** – стиль першої літери текстового блоку;
– **:first-line** – стиль першого рядка текстового блоку;
– **:after** – додає вміст після елемента;
– **:before** – додає вміст до елемента;
– **::selection** – стиль виділеного користувачем тексту.

Так само як і у випадку з псевдокласів, псевдоелементи використовуються згідно наступного синтаксису:

```
p:first-letter {
     color: #ff0000
}
```

Псевдоелементи **after** та **before** призначені для "врізки" в сторінку сайту контенту, який спочатку відсутній в HTML документі. Вставляється зміст перед (:before) або після (:after) будь-якого елементу за допомогою властивості content, яке власне і визначає вміст для вставки. Наприклад:

```
p:after {
     content: "Кінець абзацу!";
}
```

Після кожного параграфа буде додаватися напис: "Кінець абзацу!".

Селектори атрибутів

Селектори атрибутів – це селектори, які дозволяють вибирати елементи з будь-яких атрибутів.

Найчастіше такі селектори використовуються при роботі з формами, так як поля форм мають атрибут type з різними значеннями.

Селектори атрибутів записуються з використанням квадратних дужок:

```
елемент [атрибут].
```

Наприклад:

```
input [checked] {...}
input [type = "text"] {...}
```

Перший селектор вибере поля форми, у яких є атрибут *checked*, другий селектор вибере поля форми, у яких атрибут *type* має значення *text*.

Селектор по id

Існує ще один HTML-атрибут, для якого існує спеціальний селектор. Цей атрибут id (ідентифікатор), а селектор записується за допомогою символу #, наприклад

```
#some-id {…}
```

На значення id поширюються ті ж обмеження, що і на ім'я класу. Крім того id повинен бути унікальним на сторінці.

Використання селекторів такого типу при оформленні вважається поганою практикою.

Спадкування

Спадкування – механізм, за допомогою якого значення властивостей елемента-батька передаються його елементіам-нащадкам.

Стилі, присвоєні деякому елементу, успадковуються всіма нащадками (вкладеними елементами), якщо вони не перевизначені явно. Наприклад, розмір шрифту і його колір досить застосувати до тегу body, щоб всі елементи всередині мали ті ж властивості.

Спадкування дозволяє скоротити розмір таблиці стилів, але якщо стилів багато, то відстежити який батьківський елемент встановив деякий властивість, стає складніше.

До спадкоємних властивостей відносяться в першу чергу властивості, що визначають параметри відображення тексту:

```
font-size, font-family, font-style, font-weight, color, text-
align, text-transform, text-indent, line-height, letter-
spacing, word-spacing, white-space, direction.
```

Також до спадкоємною властивостей відносяться list-style, cursor, visibility, border-collapse і деякі інші. Але вони використовуються значно рідше.

Всі інші властивості не успадковуються. Як правило, це параметри позиціонування, розмірів, відступів, фону, рамок:

```
background, border, padding, margin, width, height, position
```

Повний перелік властивостей обох типів наведено у стандарті CSS.

Каскадність

Каскадність – набір правил, які визначають, які саме стильові властивості елементів веб-сторінок будуть застосовані до конкретних елементів. Або іншими словами до одного і того ж елементу може застосовуватися кілька CSS-правил (наборів CSS-властивостей). Серед цих властивостей можуть бути і конфліктуючі між собою. Тому існують інструкції, які визначають, яким буде фінальний набір властивостей елемента.

Стилі для елемента можуть бути визначені в декількох місцях:

- внутрішні стилі – вказані у атрибуті style;
- глобальні стилі – стилі визначені тегами <style> у документі;
- пов’язані стилі – стилі, що підключаються як зовнішні файли;
- стилі визначені браузером.

Браузер знаходить всі CSS-правила, що зачіпають даний елемент, а потім комбінує їх і отримує підсумковий список властивостей для цього елемента. Комбінування властивостей проводиться за чіткими правилами, які спираються на пріоритетність та специфічність, порядок вихідного коду.

Порядок пріоритетності правил невступний:

1. правила визначені атрибутом style;
2. правила визначені селектором по id;
3. правила визначені селектором класу, псевдокласу та атрибута;
4. правила визначені селектором тега та псевдоелементи.

Але у разі контестних та дочерніх селекторів, псвдокласів або псевдоелементів пріорітетність визначається з урахуванням специфічності селекторів.

Всі 4 правила приорітетності зводяться в одну систему a-b-c-d (де a - найвищий пріоритет) і утворюють специфічність.

Принцип побудови такої системи специфічності приведено у наступній таблиці:

Селектор	Специфічність a-b-c-d	Правило №
*	0-0-0-0	-
li	0-0-0-1	4
li:first-line	0-0-0-2	4
ul li	0-0-0-2	4

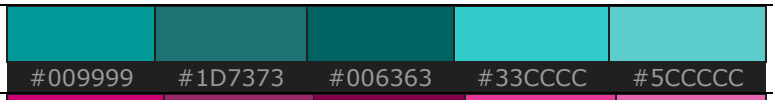
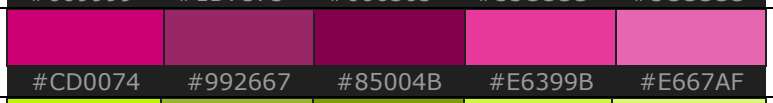
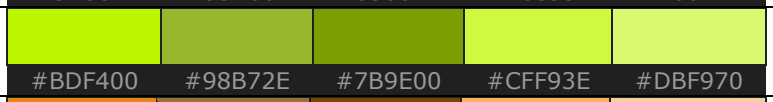
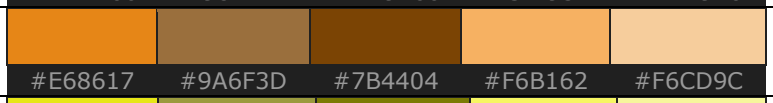
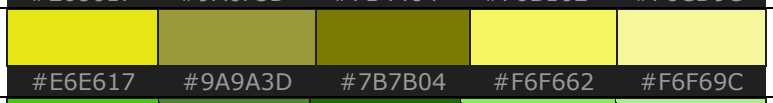
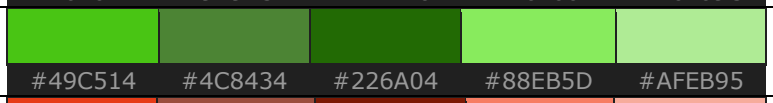
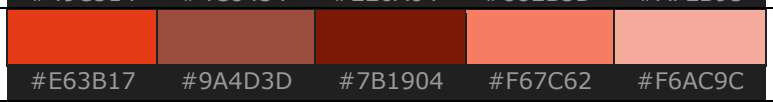
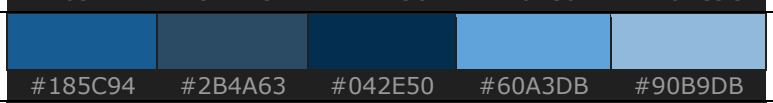
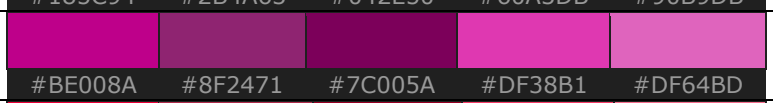
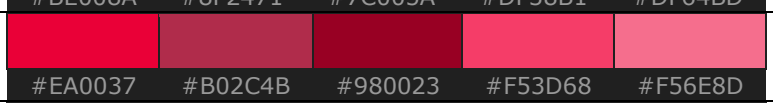
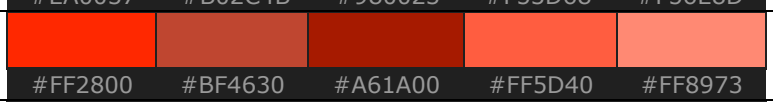
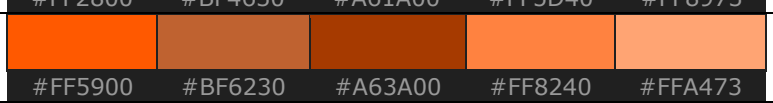
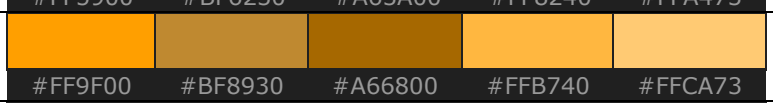
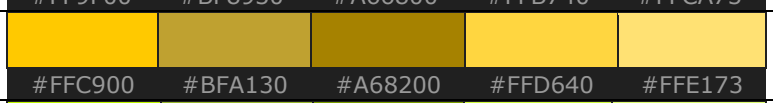
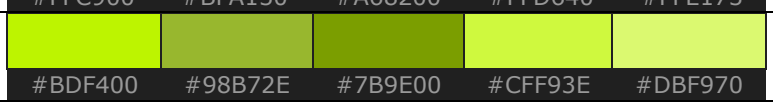
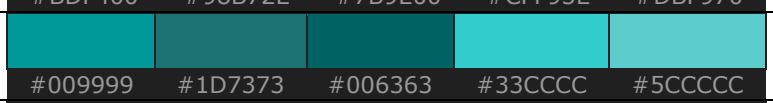
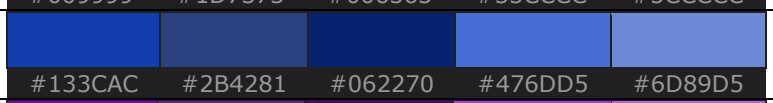
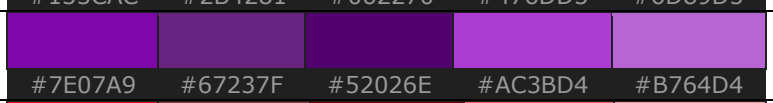
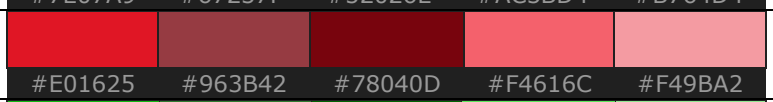
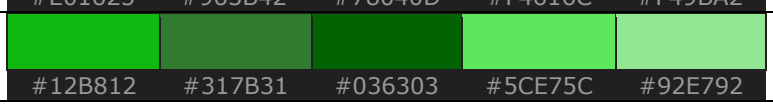
ul ol+li	0-0-0-3	4
form + *[type=text]	0-0-1-1	3, 4
table tr td.second	0-0-1-3	3, 4
h2.block.title.	0-0-2-1	3, 4
#xyz	0-1-0-0	2
style=" "	1-0-0-0	1

Завдання

1. Завдання лабораторної роботи №3 виконується на базі результату лабораторної роботи №2. Файли *.html не повинні містити жодного форматування за допомогою глобальної таблиці стилів та атрибутів тегів.
2. Створити файл style.css та розмістити у директорії css поруч з файлами *.html з лабораторної роботою №2.
3. Поруч з файлом style.css створити (завантажити у мережі) файл reset.css[11] або normalize.css[12] для скиданням стандартних стилів браузера.
4. Підключити файли reset.css та style.css у заголовку усіх сторінок лабораторної роботи №2.
5. Використовуючи CSS параметри форматування тексту налаштування формат тексту, заголовків, посилань і т.д згідно варіанту палітри та макету сторінки для обраної схеми кольорів[13].
6. Використовуючи параметри CSS3 для роботи з градієнтами, встановити фоновий колір сторінок сайту у вигляді градієнта використовуючи кольори палітри.
7. Зафіксувати ширину контенту сайту та вирівняти його по центру. Встановити відступи та поля (додати для цього необхідні теги у всіх сторінках).
8. Розмістити файли на хостингу у директорії lab3.
9. Результати виконання лабораторної роботи розташувати у власному репозиторії будь якої відкритої системи контролю версій з тегом lab4.
10. Предметами захисту роботи є вихідний текст у репозиторії системи контролю версій та загальнодоступний ресурс, що відображає результат виконання лабораторної роботи.

Варіанти палітри кольорів

[11] http://habrahabr.ru/post/45296/
[12] https://necolas.github.io/normalize.css/
[13] За бажанням виконавця можливо змінювати параметри контрастності та насичення кольорів палітри

№	Палітра	Посилання на тему
1		https://colorscheme.ru/#3i11Tw0w0w0w0
2		https://colorscheme.ru/#5a11Tw0w0w0w0
3		https://colorscheme.ru/#2c11Tw0w0w0w0
4		https://colorscheme.ru/#0D11WsOsOFfFf
5		https://colorscheme.ru/#1T11TsOsOFfFf
6		https://colorscheme.ru/#1T11TsOsOFfFf
7		https://colorscheme.ru/#0811TsOsOFfFf
8		https://colorscheme.ru/#3y11TsOsOFfFf
9		https://colorscheme.ru/#5311Tw0w0w0w0
10		https://colorscheme.ru/#5t11Tw0w0w0w0
11		https://colorscheme.ru/#0711Tw0w0w0w0
12		https://colorscheme.ru/#0k11Tw0w0w0w0
13		https://colorscheme.ru/#0Q11Tw0w0w0w0
14		https://colorscheme.ru/#1i11Tw0w0w0w0
15		https://colorscheme.ru/#2c11Tw0w0w0w0
16		https://colorscheme.ru/#3i11Tw0w0w0w0
17		https://colorscheme.ru/#3N11Tw0w0w0w0
18		https://colorscheme.ru/#4M11Tw0w0w0w0
19		https://colorscheme.ru/#5A11TsOsOFfFf
20		https://colorscheme.ru/#2P11TsOsOFfFf

Контрольні питання

1. Яке основне призначення каскадної таблиці стилів?
2. Для чого необхідні контекстні селектори?
3. Для чого призначені псевдоелементи?
4. Як працює механізм спадкування CSS?
5. Як визначити специфічність селектора?

Для нотаток:

Лабораторна робота №5. Використання CSS для створення меню.

Мета: опанувати основні властивості CSS для виводу навігаційних списків.

Теоретичні відомості

В HTML формирование элементов на странице происходит сверху вниз согласно схеме документа. Слой, размещенный в самом верху кода, отобразится раньше слоя, который расположен в коде ниже. Такая логика позволяет легко прогнозировать результат вывода элементов и управлять им. Порядок вывода объектов на странице и называется «**потоком**». При этом существует несколько возможностей «вырвать» элемент из потока и придать ему почти мифические свойства. Раз он не существует в потоке, то в коде его можно описать где угодно, а также выводить в заданное место окна.

Свойство position

Это свойство устанавливает способ позиционирования элемента относительно окна браузера или других объектов на веб-странице и используется в следующем формате:

```
position: absolute | fixed | relative | static | inherit
```

absolute – указывает, что элемент абсолютно позиционирован, при этом другие элементы отображаются на веб-странице словно абсолютно позиционированного элемента и нет. Положение элемента задается свойствами **left**, **top**, **right** и **bottom.**

На положение влияет значение свойства position родительского элемента. Так, если у родителя значение positionустановлено как static или родителя нет, то отсчет координат ведется от края окна браузера. Если у родителя значение position задано как fixed, relative или absolute, то отсчет координат ведется от края родительского элемента.

Fixed – по своему действию это значение близко к absolute, но в отличие от него привязывается к указанной свойствами left, top, right и bottom точке на экране и не меняет своего положения при прокрутке веб-страницы.

Relative – устанавливае положение элемента относительно его исходного места. Добавление свойств left, top, right и bottom изменяет позицию элемента и сдвигает его в ту или иную сторону от первоначального расположения.

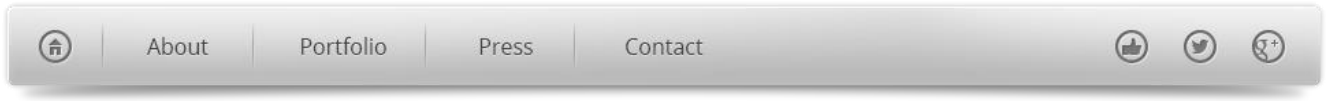
Static – элементы отображаются как обычно. Использование свойств left, top, right и bottom не приводит к каким-либо результатам.

Inherit – Наследует значение родителя.

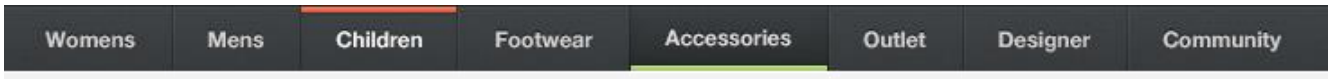
Завдання

Завдання лабораторної роботи №4 виконується на базі результату лабораторної роботи №3.

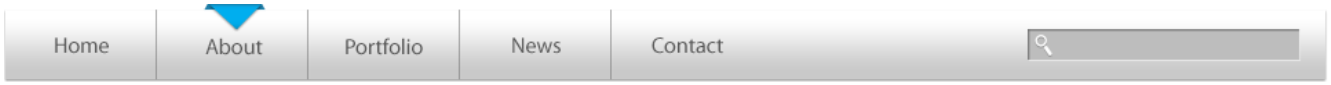
1. Для системи навігації сайту визначити необхідні правила CSS, обравши у якості макету шаблон меню зображений на рис 4.1 а, а у якості активного пункту меню використати приклади з рис 4.1 б або в.[14].
2. Кольори меню необхідно адаптувати то палітри з варіанту попередньої роботи.
3. Меню обовязково повинно містити посилання на головну сторінку та сторінки соціальних мереж студента.
4. У нижньому кутку всіх сторінок розмістити посилання «До гори», що повертає сторінку до початку сторінки.
5. Розмістити файли на хостингу у директорії lab4.
6. Результати виконання лабораторної роботи розташувати у власному репозиторії будь якої відкритої системи контролю версій з тегом lab5.
7. Предметами захисту роботи є вихідний текст у репозиторії системи контролю версій та загальнодоступний ресурс, що відображає результат виконання лабораторної роботи.



а)



б)



в)

Рисунок 4.1 – Варіанти оформлення меню.

Контрольні питання

[14] Макети меню у форматі PSD: https://drive.google.com/drive/folders/0B3z9wLpjOlQnT3l2QUItQjQtMjQ?usp=sharing

Лабораторна робота №6. CSS препроцессори.

Мета: опанувати популярні CSS препроцессори.

Теоретичні відомості

Завдання

1. Налагодити завдання збирача проектів gulp для обробки файлів *.sass, *.less, *.stylus
2. Налагодити відстеження змін та запуск CSS препроцессорів для *.sass, *.less, *.stylus файлів відповідно.
3. Рівномірно розподілити та адаптувати css з попередньої роботи у файли *.sass, *.less, *.stylus.
4. Обов’язково використати вкладеність, Домішки, Спадкування Імпорт, Операції тощо.
5. Результати виконання лабораторної роботи розташувати у власному репозиторії будь якої відкритої системи контролю версій[15] з тегом lab1.
6. Предметами захисту роботи є вихідний текст у репозиторії системи контролю версій та загальнодоступний ресурс, що відображає результат виконання лабораторної роботи.

[15] https://github.com або https://bitbucket.org

Лабораторна робота №7. Медіа запити.

Мета: опанувати основні властивості CSS для виводу на різні типи пристроїв.

Теоретичні відомості

Як правило, сторінки сайтів переглядають на різних пристроях (монітори настільних ПК, ноутбуки, планшети, телефони, тощо). Таким чином, роздільна здатність екранів цих пристроїв змінюється у широкому діапазоні. Для коректного відображення сторінок сайту на різних пристроях використовують механізми адаптивного верстання.

Адаптивна верстка – підхід, що передбачає зміну дизайну в залежності від поведінки користувача, розміру екрана, платформи і орієнтації девайса. Іншими словами, сторінка повинна автоматично підлаштовуватися під роздільну здатність, змінювати розмір картинок, шрифтів тощо (рис 4.1).

Рисунок 4.1 – Приклад використання адаптивного верстання.

Основний механізм, який дозволяє «адаптуватися» до різноманітних пристроїв є так звані медіа-запити в CSS.

Медіа запити дозволяють використовувати особливі css-стилі для конкретних пристроїв виведення. До появи CSS3 розробники могли підключати спеціальні стилі для різних пристроїв за допомогою атрибута media, наприклад:

```
<link rel = "stylesheet" href = "print.css" media = "print">
```

Перелік доступних значень атрибуту media приведено у таблиці 4.1.

Таблиця 4.1 – Типі пристроїв та їх опис

Тип	Описание
all	Всі типи. Це значення використовується за замовченням.
braille	Пристрої, засновані на системі Брайля, які призначені для читання сліпими людьми.
embossed	Принтери, що використовують для друку систему Брайля.
handheld	Смартфони і аналогічні їм апарати.
print	Принтери і інші друкарські пристрої.
projection	Проектори.
screen	Екран монітора.
speech	Мовні синтезатори, а також програми для відтворення тексту вголос. Сюди, наприклад, можна віднести мовні браузери.
tty	Пристрої з фіксованим розміром символів (телетайпи, термінали, пристрої з обмеженнями дисплея).
tv	Телевізори.

Медіа запити являють собою логічні вирази, які можуть бути істинними або помилковими. У разі істинності виразу, застосуються відповідні правила CSS.

Окрім типів носіїв до CSS3 включена підтримка різних технічних параметрів пристроїв, на основі яких потрібно завантажувати ті чи інші стилі.

Перелік параметрів, які можна використовувати наведено у таблиці 4.2.

Таблиця 4.1 – Типі пристроїв та їх опис

Параметр	Описание
width	Перевіряє ширину області перегляду. Значення задаються в одиницях довжини, px, em і т.д., наприклад, (width: 800px). Зазвичай для перевірки використовуються мінімальні і максимальні значення ширини: **min-width** – ширина області перегляду більше значення, зазначеного в запиті; **max-width** – ширина області перегляду менше значення, зазначеного в запиті.
height	Перевіряє висоту області перегляду. Значення задаються в одиницях довжини, px, em і т.д., наприклад, (height: 500px). Зазвичай для перевірки використовуються мінімальні і

	максимальні значення висоти: **min-height** – висота області перегляду більше значення, зазначеного в запиті; **max-height** – висота області перегляду менше значення, зазначеного в запиті.
aspect-ratio	Перевіряє співвідношення ширини до висоти області перегляду. Широкоекранний дисплей зі співвідношенням сторін 16:9 може бути позначений як (aspect-ratio: 16/9). **min-aspect-ratio** – перевіряє мінімальне співвідношення. **max-aspect-ratio** – максимальне співвідношення ширини до висоти області перегляду.
orientation	Перевіряє орієнтацію області перегляду. Приймає два значення: (**orientation: portrait**) і (**orientation: landscape**).
resolution	Перевіряє роздільну здатність екрана (кількість пікселів). Значення також можуть перевіряти кількість точок на дюйм (dpi) або кількість точок на сантиметр (dpcm), наприклад, (resolution: 300dpi). **min-resolution** – перевіряє мінімальну роздільну здатність екрана, **max-resolution** – максимальну.
color	Перевіряє кількість біт на кожен з кольорних компонентів пристрою виведення. Наприклад, (min-color: 4) означає, що екран конкретного пристрою повинен мати 4-бітну глибину кольору. **min-color** – перевіряє мінімальну кількість біт, **max-color** – максимальну кількість біт.
color-index	Перевіряє кількість записів в таблиці підстановки кольорів. Як правило вказується позитивне число, наприклад (color-index: 256). **min-color-index** – перевіряє мінімальну кількість записів, **max-color-index** – максимальна кількість записів.
monochrome	Перевіряє кількість бітів на піксель монохромного пристрою. Значення задається цілим позитивним числом, наприклад, (min-monochrome: 8). **min-monochrome** – перевіряє мінімальну кількість бітів, **max-monochrome** – максимальна кількість бітів.
-webkit-device-pixel-ratio	Задає кількість фізичних пікселів пристрої на кожен css-піксель.

При складанні медіа запитів потрібно орієнтуватися на так звані переломні точки дизайну, тобто такі значення ширини області перегляду, в яких дизайн сайту істотно змінюється, наприклад, з'являється горизонтальна смуга прокрутки.

Щоб визначити ці точки, потрібно відкрити сайт в браузері і поступово зменшувати область перегляду.

Щоб адаптувати дизайн сайту під різні пристрої, необхідно задати різні стилі для різних роздільних здатносте екранів, використовуючи такі контрольні точки (не обов'язково все):

```
/* Smartphones (вертикальная и горизонтальная ориентация) -------- */
@media only screen and (min-width : 320px) and (max-width : 480px) {
/* стили */
}

/* Smartphones (горизонтальная) ----------- */
@media only screen and (min-width: 321px) {
/* стили */
}

/* Smartphones (вертикальная) ----------- */
@media only screen and (max-width: 320px) {
/* стили */
}

/* iPads (вертикальная и горизонтальная) ----------- */
@media only screen and (min-width: 768px) and (max-width: 1024px) {
/* стили */
}

/* iPads (горизонтальная) ----------- */
@media only screen and (min-width: 768px) and (max-width: 1024px) and
(orientation: landscape) {
/* стили */
}

/* iPads (вертикальная) ----------- */
@media only screen and (min-width: 768px) and (max-width: 1024px) and
(orientation: portrait) {
/* стили */
}

/* iPad 3**********/
@media only screen and (min-width: 768px) and (max-width: 1024px) and
(orientation: landscape) and (-webkit-min-device-pixel-ratio: 2) {
/* стили */
}
@media only screen and (min-width: 768px) and (max-width: 1024px) and
(orientation: portrait) and (-webkit-min-device-pixel-ratio: 2) {
/* стили */
}

/* Настольные компьютеры и ноутбуки ----------- */
@media only screen  and (min-width: 1224px) {
/* стили */
}

/* Большие экраны ----------- */
```

```
@media only screen  and (min-width: 1824px) {
/* стили */
}

/* iPhone 4 ----------- */
@media only screen and (min-width: 320px) and (max-width: 480px) and
(orientation: landscape) and (-webkit-min-device-pixel-ratio: 2) {
/* стили */
}
@media only screen and (min-width: 320px) and (max-width: 480px) and
(orientation: portrait) and (-webkit-min-device-pixel-ratio: 2) {
/* стили */
}

/* iPhone 5 ----------- */
@media only screen and (min-width: 320px) and (max-height: 568px) and
(orientation: landscape) and (-webkit-device-pixel-ratio: 2){
/* стили */
}
@media only screen and (min-width: 320px) and (max-height: 568px) and
(orientation: portrait) and (-webkit-device-pixel-ratio: 2){
/* стили */
}

/* iPhone 6 ----------- */
@media only screen and (min-width: 375px) and (max-height: 667px) and
(orientation: landscape) and (-webkit-device-pixel-ratio: 2){
/* стили */
}
@media only screen and (min-width: 375px) and (max-height: 667px) and
(orientation: portrait) and (-webkit-device-pixel-ratio: 2){
/* стили */
}

/* iPhone 6+ ----------- */
@media only screen and (min-width: 414px) and (max-height: 736px) and
(orientation: landscape) and (-webkit-device-pixel-ratio: 2){
/* стили */
}
@media only screen and (min-width: 414px) and (max-height: 736px) and
(orientation: portrait) and (-webkit-device-pixel-ratio: 2){
/* стили */
}

/* Samsung Galaxy S3 ----------- */
@media only screen and (min-width: 320px) and (max-height: 640px) and
(orientation: landscape) and (-webkit-device-pixel-ratio: 2){
/* стили */
}
@media only screen and (min-width: 320px) and (max-height: 640px) and
(orientation: portrait) and (-webkit-device-pixel-ratio: 2){
/* стили */
}

/* Samsung Galaxy S4 ----------- */
@media only screen and (min-width: 320px) and (max-height: 640px) and
(orientation: landscape) and (-webkit-device-pixel-ratio: 3){
/* стили */
```

```
}
@media only screen and (min-width: 320px) and (max-height: 640px) and
(orientation: portrait) and (-webkit-device-pixel-ratio: 3){
/* стили */
}

/* Samsung Galaxy S5 ----------- */
@media only screen and (min-width: 360px) and (max-height: 640px) and
(orientation: landscape) and (-webkit-device-pixel-ratio: 3){
/* стили */
}
@media only screen and (min-width: 360px) and (max-height: 640px) and
(orientation: portrait) and (-webkit-device-pixel-ratio: 3){
/* стили */
}
```

Завдання

1. Для попередньої лабораторної роботи реалізувати окремий файл стилів responsive.css та підключити його до загальних (файл styles.css) стилів використовуючи @import.
2. У responsive.css реалізувати медіа запити та відповідні стилі згідно наступних вимог:
    a. при ширині вікна браузера, що менше ніж фіксована ширина виводу контенту, перетворити фіксовану ширину виводу контенту на відносну 90% (обов'язково центрувати);
    b. головне меню, при зменшенню розміру вікна браузера, перетворити з горизонтального на вертикальне, елементи меню відносної ширини та центровані (шапка сайту відповідно змінює висоту);
    c. у версіях браузера Internet Explorer 7.x та нижчих відобразити відповідне (передбачити заздалегідь приховане) повідомлення у шапці сайту.
3. Створити та підключити за допомогою тегу <link> файл print.css для вивода друкованої версії сторінок сайту згідно наступних вимог:
    a. Виводити при друці назву сайту, заголовок та вміст сторінки, приховавши решту шапки та підвал сайту;
    b. Видалити з виводу всі фонові зображення та кольри
    c. Відобразити повний URL взамін гіпертекстових посилань;
    d. Для тексту передбачити вивід з використанням шрифтів с засіками (Times New Roman, Georgia, Palatino);
    e. Встановити поля при друці рівні 50px;

4. Розмістити файли сайту на хостингу у директорії lab7 та протестувати результати роботи.
5. Результати виконання лабораторної роботи розташувати у власному репозиторії будь якої відкритої системи контролю версій[16] з тегом lab1.
6. Предметами захисту роботи є вихідний текст у репозиторії системи контролю версій та загальнодоступний ресурс, що відображає результат виконання лабораторної роботи.

[16] https://github.com або https://bitbucket.org

Лабораторна робота №8. Основи JavaScript

Мета: опанувати основи мови програмування JavaScript та застосувати їх для тестування браузерів на сумісність із специфікацією HTML5.

Теоретичні відомості

Сучасний **JavaScript** – це скриптова мова програмування загального призначення, можливості якої залежать від середовища виконання.

У браузері JavaScript має можливість вміє робити створювати нові HTML-теги, видаляти існуючі, змінювати стилі елементів, ховати, показувати елементи тощо. JavaScript реагує на дії користувачів, обробляє кліки миші, переміщення курсору, натискання на клаватуру та інші події. Також він виконує асинхронну взаємодію (Ajax) з сервером та багато іншого.

Більшість можливостей JavaScript в браузері обмежена поточним вікном і сторінкою, а саме (рис 6.1):

− JavaScript не може читати / записувати довільні файли на жорсткий диск, копіювати їх або викликати програми, виключаючи роботу у спеціально виділеній директорії – «пісочниця»;

− JavaScript, що працює в одній вкладці, не може спілкуватися з іншими вкладками і вікнами, за винятком випадку, коли він сам відкрив це вікно або декілька вкладок з одного джерела (однаковий домен, порт, протокол).

− JavaScript може легко посилати запити на сервер, з якого прийшла сторінка а взаємодія з інший доменами допускається не у всіх випадках, існують обмеження безпеки.

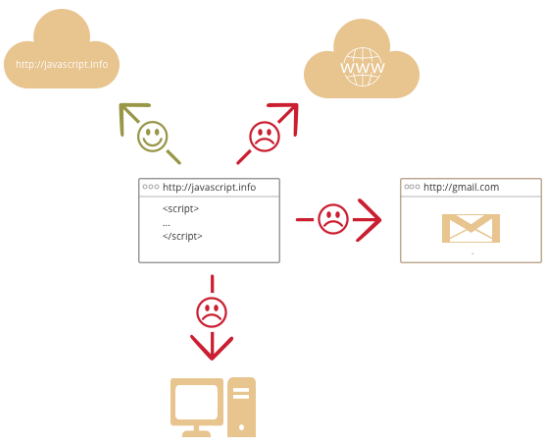


Рисунок 6.1 – Можливості JavaScript

Основними елементами з якими взаємодіє JavaScript є DOM та BOM

Об'єктна модель документа (англ. Document Object Model, DOM) — специфікація прикладного програмного інтерфейсу для роботи зі структурованими документами (як правило, документів XML). З точки зору об'єктно-орієнтованого програмування, DOM визначає класи, методи та атрибути цих методів для аналізу структури документів та роботи із представленям документів у вигляді дерева. Все це призначено для того, аби надати можливість комп'ютерній програмі доступу та динамічної модифікації структури, змісту та оформлення документа.

BOM (Browser object model) – забезпечує доступ до вікна браузера і дозволяє маніпулювати ним і його елементами.

Завдання

1. Використовуючи мову програмування JavaScript (без використання jQuery) та відповідні правила CSS розробити слайдер зображень з можливостями конфігурування:
   - кількості елементів слайдеру;
   - швидкість анімації зображень;
   - напрям анімації;
   - ширина контейнеру.
2. Розташувати код JS у окремому файлі з назвою sdudent_firstname-slider.js (sdudent_firstname – прізвище студента англійською мовою).
3. Передбачити можливість визначення параметрів за допомогою JS об’єкту та параметрів контейнеру (зовнішній блочний елемент) у якому буде розташований слайдер.
4. Параметри JS об’єкту повинні мати вищій приорітет ніж параметри визначені атрибутами data-*.
5. Для визначення CSS властивостей слайдеру передбачити окремий файл css з назвою sdudent_firstname-slider.css
6. Реалізація слайдеру спирається на наступні принципи:
   - стрічка зображень в розмітці повинна бути представлена як список <ul> тегів <img>, який потрібно розташувати всередині контейнеру фіксованого розміру, так щоб в один момент було видно тільки необхідну частини списку (рис 6.1);

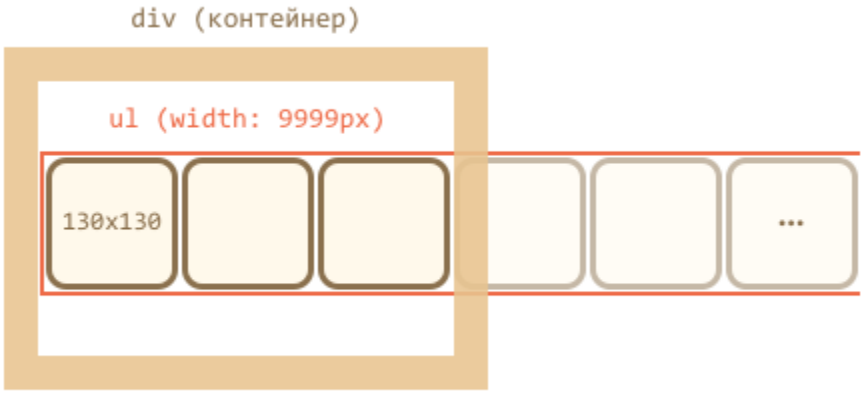


Рисунок 6.1 – Принцип відображення галереї.

- щоб список був довгий і елементи не переходили вниз, потрібно встановити width: 9999px, а елементам <li>, відповідно, float: left, або display: inline-block;
- для тегів <img> потрібно поставити display: block;
- для «прокрутки» необхідно переміщувати <ul> за допомогою CSS-властивості transform: translateX () або margin-left:

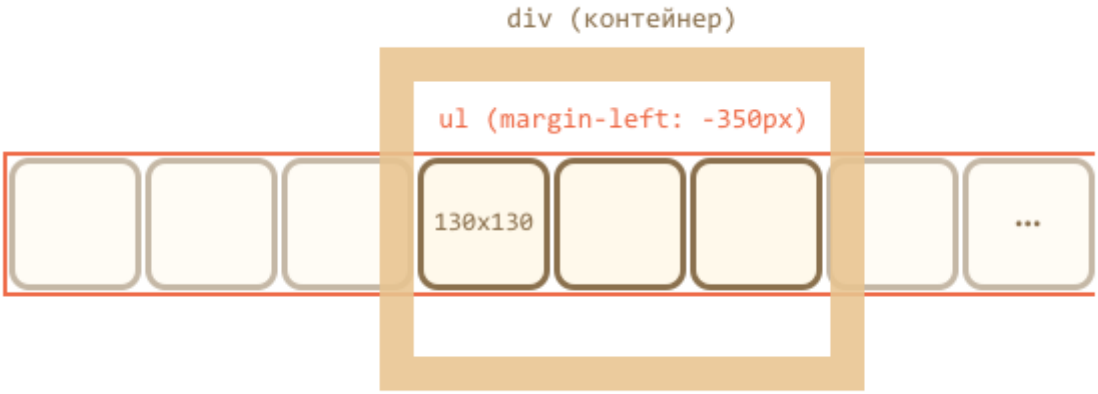


Рисунок 6.2 – Принцип прокрутки елементів

7. Для тестування роботи слайдеру реалізувати окрему сторінку сайту розробленого при виконанні попередніх лабораторних робіт.

8. Результати виконання лабораторної роботи розташувати у власному репозиторії будь якої відкритої системи контролю версій з тегом lab6.

9. Предметами захисту роботи є вихідний текст у репозиторії системи контролю версій та загальнодоступний ресурс, що відображає результат виконання лабораторної роботи.

Лабораторна робота №9. Основи jQuery

Завдання

Реалізувати виконання попередньої лабораторної роботи, але з використанням бібліотеки jQuery.

Лабораторна робота №10. Плагіни та віджети jQuery

Завдання

1. На сторінці галереї реалізувати слайд-шоу використовуючи будь який доступний у мережі плагін jQuery.
2. Існуючи на сторінці фото розділити на дві категорії та відобразити розділи у вигляді вкладок (jQueryUI).
3. У вкладках реалізувати відображення фото з використання плагінів LightBox та ColorBox.

Індивідуальне завдання

Завдання

1. Обрати довільний шаблон сайту у форматі psd, jpeg або іншому.

2. Затвердити шаблон з викладачем

3. Ознайомитися з технологією БЕМ (наприклад https://ru.bem.info/, http://habrahabr.ru/post/203440/)

4. Використовуючи принципи БЕМ зверстати головну та другорядну сторінку сайту.

5. Захистити індивідуальну роботу

Література

Основна:

1. Пейтон К. PHP-5 и MySQL 5 – Бином: Русская Редакция, 2007.- 368 с.
2. Хольцнер С. PHP в примерах (включая версию 6)– Бином: Русская Редакция, 2007.- 352 с.
3. Колисниченко Д.Н. НиТ - Самоучитель PHP 5 .- М.: Русская Редакция, 2007,- 632с.
4. Шлосснейгл Джордж. Профессиональное программирование на PHP- Вильямс: Русская Редакция, 2006,- 624с.
5. Ларри Ульман. Основы программирования на PHP - М.:ДМК Пресс, 2001,- 288с.
6. А. Мазуркевич, Д. Еловой. PHP - настольная книга програмиста. - М.:Новое знание, 2004,- 480с.
7. Флэнаган Д. Javascript. Подробное руководство .- М.: Русская Редакция, 2008,- 984с
8. Аллен Вайк. JavaScript Спровочник - DiaSoft: Русская Редакция, 2007,- 1424с.

Додаткова:

1. 1 Энди Бадд, Камерон Молл, Саймон Коллизон. Мастерская CSS: профессиональное применение Web-стандартов - М.:Вильямс, 2007,- 272с.
2. Дэйв Ши, Молли Е. Хольцшлаг. Философия CSS-дизайна - М.:НТ Пресс, 2003,- 312с.
3. Дейв Паскарслло, Эрик Джеймс Даррен. Ajax в действии. : Пер. с англ. - М.: Вильямс, 2006. - 640 с. :
4. http://www.mysql.ru/
5. http://javascript.ru/
6. http://www.php.su