МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЦЕНТРАЛЬНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ

**до самостійної роботи
з дисципліни «Web-програмування»
для студентів денної та заочної форм навчання за спеціальностями
122/F3 «Комп'ютерні науки», 123/F7 «Комп'ютерна інженерія»,
125/F5 «Кібербезпека та захист інформації», 172/G5 «Електроніка,
електронні комунікації, приладобудування та радіотехніка»**

ЗАТВЕРДЖЕНО
На засідання кафедри кібербезпеки та
програмного забезпечення,
протокол від 25.08.2025 року № 1

КРОПИВНИЦЬКИЙ
2025

Методичні рекомендації до самостійної роботи з дисципліни «Web-програмування» для студентів денної та заочної форм навчання за спеціальностями 122/F3 «Комп’ютерні науки», 123/F7 «Комп’ютерна інженерія», 125/F5 «Кібербезпека та захист інформації», 172/G5 «Електроніка, електронні комунікації, приладобудування та радіотехніка» / уклад. В.В. Босько, Л.В. Константинова — Кропивницький: ЦНТУ, 2025. — 44 с.

Укладач: Босько В. В., Константинова Л. В.

Рецензенти: Смірнов О. А., д-р техн. наук, професор;
Дрєєв О. М., канд. техн. наук.

Теми самостійної роботи

з навчальної дисципліни «Web-програмування» для студентів денної та заочної форм навчання за спеціальностями 122/F3 «Комп'ютерні науки», 123/F7 «Комп'ютерна інженерія», 125/F5 «Кібербезпека та захист інформації», 172/G5 «Електроніка, електронні комунікації, приладобудування та радіотехніка»

№з.с.	Назва теми
1	Тема 1. Веб-служба. Загальні положення. Веб-сервер. Операційні системи для серверів.
2	Тема 2. Взаємодія браузера та веб-серверу.
3	Тема 3. HTML та CSS як головні інструменти розробки фронт-енд частини вебсайту
4	Тема 4. CSS стилі в HTML. CSS-препроцесори. Медіа запити.
5	Тема 5. Можливості й обмеження JavaScript. Особливості програмування мовою JavaScript в DOM.
6	Тема 6. Адаптивний дизайн Bootstrap
7	Тема 7. Основні аспекти PHP у веб-програмуванні.
8	Тема 8. Популярні фреймворки. Переваги та недоліки використання фреймворків.

Методичні рекомендації призначені для студентів для студентів денної та заочної форм навчання за спеціальностями 122/F3 «Комп'ютерні науки», 123/F7 «Комп'ютерна інженерія», 125/F5 «Кібербезпека та захист інформації», 172/G5 «Електроніка, електронні комунікації, приладобудування та радіотехніка». Для виконання завдань до самостійної роботи представлено теоретичний матеріал за темами, що наведено в таблиці та контрольні запитання для самоперевірки. Дані рекомендації доповнюють методичні рекомендації до лабораторних робіт і допоможуть у вивченні матеріалу з дисципліни «Web-програмування». В кінці наведено список рекомендованих джерел інформації.

Самостійна робота №1

Тема: Веб-служба. Загальні положення. Веб-сервер. Операційні системи для серверів.

Теоретичні відомості.

Веб-служба (web service)- це програмна система, що надає послуги через Інтернет за допомогою стандартних протоколів (HTTP, SOAP, REST) для обміну даними та функціями між іншими програмами чи системами, незалежно від їх платформи чи мови програмування, дозволяючи автоматизувати процеси та інтегрувати різні додатки.

Комп'ютери являються найбільш важливими апаратними компонентами системи. Одні комп'ютери виступають в ролі *серверів* – це означає, що вони надають послуги іншим комп'ютерам. Комп'ютери, які користуються їх послугами, називаються клієнтами. На рисунку 1.1 - зображено клієнт-серверну архітектуру. Клієнт виконує два завдання – представлення й запити. Представлення також називається – інтерфейсом; він просто дає клієнту можливість взаємодіяти з комп'ютером і перетворює дані, отримані від сервера, в формат зрозумілий користувачу. Клієнт також виконує запити на сервер, і сервер відповідає на запити, відправлені клієнтом. Це є простий опис – на практиці він набагато складніший. Для того щоб зрозуміти як працює система клієнт-сервер, використаємо для прикладу Інтернет. Коли ми використовуємо Інтернет, ми використовуємо багаторівневу архітектуру. Але зупинимося на дворівневій архітектурі (клієнт – один сервер).

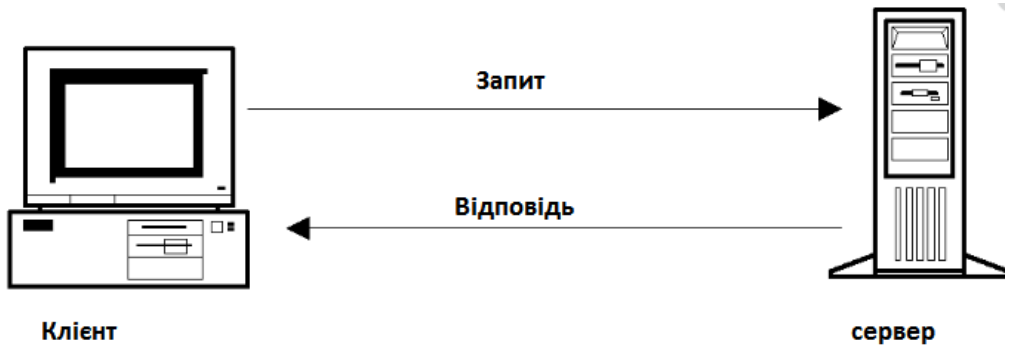

Рисунок 1.1 – Клієнт-серверна архітектура Інтернету

В браузері є місце, де можна ввести адресу, й метод віддати команду щоб перейти на цю адресу. В дійсності, коли ми натискаємо клавішу Enter, браузер відправляє нас на Web-сервер Google запит домашньої сторінки Google.

Домашня сторінка – це файл, написаний на мові HTML. Web-сервер відповідає на запит нашого браузера, відправляючи через Інтернет дані на наш комп’ютер (клієнт). На нашому комп’ютері ці дані повинні утворити домашню сторінку пошукової системи. Web-браузер приймає дані (в вигляді HTML-файлу) і конвертує код мови HTML, як ви це вже бачите на екрані комп’ютера. Ще одна частина функції представлення: відображення інформації в придатній формі. Відповідно клієнт (комп’ютер із запущеною програмою браузера) дає нам зручну можливість створити запит на сервер, а потім відправити цей запит через Інтернет на Web-сервер Google. Сервер після цього відправляє запитані дані через Інтернет на Ваш комп’ютер, де Web-браузер перетворює їх на Web-сторінку. Цей процес графічно зображено на рисунку 1.2.

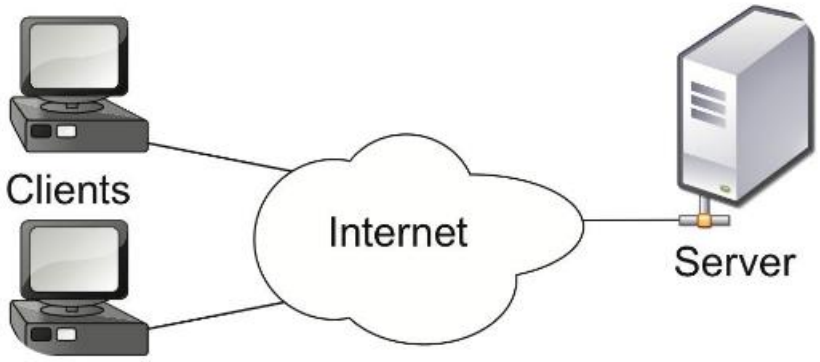

Рисунок 1.2 – Приклад клієнт-серверної архітектури Інтернету

HTTP (Hyper Text Transfer Protocol) - протокол передачі даних, що використовується в комп'ютерних мережах. Якщо без зайвих ускладнень, то це проста текстова мова, яка дозволяє двом комп'ютерам спілкуватися один з одним. До прикладу, ноутбук чи смартфон (далі просто клієнт) звертається до сервера використовуючи HTTP і чекає відповідь. Сервер обробляє запит і повертає клієнту тією ж мовою відповідь. Запит - це свого роду текстове повідомлення, яке створюється клієнтом. Тільки формується він відповідно до спеціальних правил формату, який відомий як HTTP.

Широко використовуються наступні HTTP-методи:

- GET — отримати ресурс із серверу;
- POST — створити ресурс на сервері;
- PUT — оновити ресурс на сервері;

- DELETE — видалити ресурс із серверу.

Веб-сервер

Веб-серверами можуть бути комп'ютери або спеціальні програми, які виконують роль сервера. Коли користувач намагається отримати HTML-документ через рядок вводу адреси, то браузер посилає запит через протокол передачі даних HTTP. Коли запит досягає потрібного веб-сервера (залізо), сервер HTTP (програмне забезпечення) передає запитуваний документ назад, також через HTTP.

Веб-сервер може бути статичним або ж динамічним. Статичний сервер просто надсилає потрібні файли в браузер. Динамічний також вміє надсилати файли в браузер, але на ньому встановлене додаткове програмне забезпечення, яке перед відправкою в браузер змінює вихідні файли. По суті, на льоту генерується відповідь — виконуються обчислення, беруться дані з бази тощо.

Apache - найбільш популярний веб-сервер у світі. Проте чимало високонавантажених вебсайтів використовують Nginx або комбінує їх.

Сучасний сайт являє собою не просто набір HTML-документів, але і включає в себе безліч технологій, бази даних та багато іншого.

Для вивчення серверних технологій не зручно та й не ефективно використовувати справжній доступний в мережі Інтернет сервер, тому варто встановити необхідний комплект програм на локальний комп'ютер і розробляти все на ньому.

Найбільш популярною зв'язкою таких програм є веб-сервер Apache, мова програмування PHP і система керування базами даних MySQL.

Редактори коду

Є чимало різних варіантів і кожен підбере той редактор, який буде найзручнішим для нього. Але слід уточнити, що умовно редактори можна поділити на текстові редактори й IDE (англ. Integrated Development Environment).

В той час як перші є легкими, інші (наприклад, Eclipse, NetBeans або AptanaStudio) - важкі, вимогливі до ресурсів, проте мають значно більше корисних функцій.

Вимоги до сучасного редактора коду

– Підсвічування синтаксису мови програмування, на якій пишуть код.

– Підказки коду, авто-завершення.

– Можливість відлагоджувати код (англ. debugging).

– Розширюваність за допомогою плагінів.

– Можливість працювати з Git (або іншою системою керування версіями файлів).

– Підтримка препроцесорів, якщо ви їх використовуєте (Less/Sass тощо).

– Наявність вбудованого FTP-клієнта, SSH тощо.

Завдання:

1. Встановіть зручний браузер на комп’ютер.
2. Встановіть веб-сервер.
3. Обрати та встановити редактор коду.

Самостійна робота №2
Тема: Взаємодія браузера та веб-серверу

Теоретичні відомості

Взаємодія браузера і веб-сервера — це процес обміну запитами та відповідями за **клієнт-серверною моделлю**, де браузер (клієнт) запитує дані (HTML, зображення), а сервер їх надсилає через **HTTP/HTTPS**, використовуючи DNS для пошуку адреси, щоб відобразити веб-сторінку користувачеві.

Якщо є потреба скористатися, наприклад, пошуковою системою google.com. При наявному підключенні до Інтернету запускається Web-браузер. Тому просто необхідно ввести в потрібному місці адресу й натиснути кнопку Enter. Це буде приклад, як клієнт виконує задачу представлення. В браузері є місце, де можна ввести адресу, й метод віддати команду щоб перейти на цю адресу. В дійсності, коли ми натискаємо клавішу Enter, браузер відправляє нас на Web-сервер Google запит домашньої сторінки Google. Домашня сторінка – це файл, написаний на мові HTML.

Web-сервер відповідає на запит нашого браузера, відправляючи через Інтернет дані на наш комп'ютер (клієнт). На нашому комп'ютері ці дані повинні утворити домашню сторінку пошукової системи. Web-браузер приймає дані (в вигляді HTML-файлу) і конвертує код мови HTML, як ви це вже бачите на екрані комп'ютера. Ще одна частина функції представлення: відображення інформації в придатній формі. Відповідно клієнт (комп'ютер із запущеною програмою браузера) дає нам зручну можливість створити запит на сервер, а потім відправити цей запит через Інтернет на Web-сервер Google. Сервер після цього відправляє запитані дані через Інтернет на комп'ютер, де Web-браузер перетворює їх на Web-сторінку.

Кроки взаємодії:

- Введення адреси (URL): Користувач вводить URL або клікає на посилання в браузері.
- Пошук IP-адреси (DNS): Браузер запитує DNS-сервер, щоб знайти IP-адресу сервера, де зберігається сайт.

- HTTP-запит: Браузер надсилає HTTP-запит (GET/POST) на цю IP-адресу, запитуючи конкретну сторінку чи дані.
- Обробка на сервері: Вебсервер отримує запит, знаходить потрібний файл (HTML, CSS, JS, зображення) або запускає серверний код для генерації динамічного вмісту (наприклад, з бази даних).
- HTTP-відповідь: Сервер відправляє відповідь, яка містить статус (200 OK, 404 Not Found) та дані (тіло відповіді).
- Рендеринг у браузері: Браузер отримує дані, інтерпретує їх і відображає веб-сторінку.

Завдання:

1. Введіть посилання в браузері на сайт дистанційної освіти.
2. Отримайте результат.
3. Проаналізуйте, які кроки взаємодії було виконано, який запит HTTP-запит було здійснено.

Самостійна робота №3

Тема: HTML та CSS як головні інструменти розробки фронт-енд частини вебсайту

Теоретичні відомості

Розробка сайту з нуля розбивається на такі етапи: дизайн сайту (шаблон); фронтенд розробка; бекенд розробка; робота з базами даних; адміністрування сайту.

Одним із ключових моментів в розвитку всесвітньої павутини є веб-розробка – процес створення вебсайту або вебдодатку. Термін включає розробку додатків електронної комерції, веб-дизайн, програмування для web на стороні клієнта й серверу, а також конфігурування веб-серверу. Основними етапами веб-розробки є:

- проектування сайту або вебдодатку;
- створення макетів сторінок;
- наповнення;
- обслуговування працюючого сайту або його програмної основи;
- подальше просування сайту в мережі та підняття його рейтингу.

Однією з головних існуючих технологій для створення сайтів є використання HTML. Більшість веб-сторінок створюються за допомогою мови HTML (або XHTML). Документ HTML оброблюється браузером та відтворюється на екрані у звичному для людини вигляді. HTML дозволяє відзначити, де в документі повинен бути заголовок або абзац за допомогою тега HTML, а потім надає Web-браузеру інтерпретувати ці теги. Наприклад, один Web-браузер може розпізнавати тег початку абзацу й представляти документ у потрібному вигляді, а інший не має такої можливості.

HTML - теги можуть бути умовно розділені на дві категорії:

- Теги, що визначають, як саме буде відображатися Web-браузером тіло документа в цілому.
- Теги, що описують загальні властивості документа, такі як заголовок чи хто є автор документа.

HTML-документи можуть бути створені за допомогою будь-якого текстового редактора або спеціалізованих HTML-редакторів і конвертерів.

Вибір редактора, який буде використовуватися для створення HTML-документів, залежить виключно від поняття зручності й особистих вподобань кожного розробника. Основна перевага HTML полягає в тому, що ваш документ може бути переглянутий на Web-браузерах різних типів і на різних платформах.

Каскадні таблиці стилів (англ. Cascading Style Sheets або скорочено CSS) - спеціальна мова, що використовується для опису сторінок, написаних мовами розмітки даних.

Найчастіше CSS використовують для візуальної презентації сторінок, написаних HTML та XHTML, але формат CSS може застосовуватися до інших видів XML-документів.

Таблицю стилів CSS можна вмонтувати прямо в HTML -сторінку - це внутрішня таблиця стилів. Або ж її можна створити в окремому файлі, і вже потім приєднати посилання на нього до потрібної HTML-сторінки - це зовнішня таблиця стилів. Зовнішню таблицю необхідно підключити до основного HTML-документу за допомогою спеціальних тегів.

Таким чином, для початку створення вебдодатків в першу чергу варто вивчити HTML і CSS (актуальні версії CSS 3, HTML 5). Ця зв'язка дозволяє змінювати розміщення, оформлення елементів на HTML-сторінці. Але перед тим, як поринути в мови розмітки/програмування, необхідно познайомитись з основними поняттями у веб-розробці, а також з корисними інструментами, які допоможуть у роботі.

Якщо створити текстовий документ з наведеним на рисунку 3.1 вмістом, то можна побачити, як створювати прості сторінки.

Результат виконання можна побачити якщо запустити збережений файл в браузері (рисунок 3.2).

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>First Page</title>
</head>
<body>
<h1>My First Heading</h1>
<p>My First paragraph</p>
</body>
</html>
```

Рисунок 3.1 – Застосування тегів для побудови сторінки

Рисунок 3.2 – Результат виконання

Завдання:

1 . Створити текстовий файл Index.html.

2. Введіть теги для побудови першої сторінки за рисунком 3.1.

3. Отриманий результат і порівняйте з результатом з рисунку 3.2.

Самостійна робота №4
Тема: CSS стилі в HTML. CSS-препроцесори. Медіа запити

Теоретичні відомості.

Якщо узагальнити, то CSS-правила складаються із **селектора** та **блоку оголошень**. Схематично це виглядає так:

```
певний-селектор {
властивість-1: значення-1;
властивість-2: значення-2;
...
}
```

Селектор вказує на HTML-елемент(и), які ми намагаємось стилізувати. Тоді як в блоці оголошень ми зазначаємо CSS-властивості цих елементів і задаємо їм певні значення.

Блок оголошень слід "огорнути" фігурними дужками. Всередині цих дужок можна вказати одне або декілька оголошень, розділених між собою крапкою з комою.

Застосуємо відповідні стилі й подивимося не результат виконання. Для цього в файлі style.css використаємо відповідні стилі:

```
h1 {
      font-family: Georgia, serif;
         text-transform:uppercase;
       }
p {
      color:#f46155;
         font-family: "TimesNewRoman", Times, serif;
         text-align: justify;
         font-size: 20px;
      }
```

Результат виконання наведено на рисунку 4.1.

Рисунок 4.1 – Результат виконання підключених файлів стилів

Таким чином, для того, щоб застосувати стилі до HTML-документу, ми можемо обрати один з чотирьох способів, або ж комбінувати їх:

– застосувати зовнішні стилі за допомогою елемента link;

– додати CSS-блок за допомогою елемента style;

– вказати стиль конкретному HTML-елементу за допомогою HTML-атрибуту style (inline-стилі);

– використати @import (правило @import дозволяє імпортувати (завантажити) вміст CSS-файлу в поточну стильову таблицю).

Найчастіше використовується метод підключення за допомогою елемента *link*, який повинен розташовуватися всередині елемента *head*.

Селектори класів

Класи CSS - це чудовий інструмент, який розширює можливості створення стилів в рази. Для кращого розуміння ми будемо розглядати все на прикладах.

Отже, трохи вище ми застосували стиль для всіх тегів <h1> та <p> на веб-сторінці – абзац має шрифт TimesNewRoman і червоний колір , а для заголовка <h1> встановлено, що він буде оформлений великими літерами (text-transform:uppercase) і шрифт (font-family: Georgia).

У випадку, якщо знадобиться змінити колір одного з тегів <h1> на зелений, наприклад, то на допомогу приходять селектори класів. Все що необхідно зробити, це створити стиль, де селектор – придумане будь-яке ім'я класу. Наприклад, назвемо клас .greentext (назва класу починається крапкою перед назвою) і запишемо правило:

```
.greentext {
color: green;
}
```

Але це ще не все. Тепер, щоб змінити колір для одного з тегів <h1> на сторінці, потрібно відредагувати HTML-документ, застосувавши клас greentext до необхідного нам тегу. Записується це так:

<h1 class = "greentext"></ h1>

Створений клас можна застосовувати до будь-яких елементів веб-сторінки. Ви можете надавати стиль не тільки цілим заголовкам й абзацам, а й окремим фрагментам сторінки, наприклад, словам (використовуючи тег привласнюючи йому клас).

Запам'ятайте кілька правил написання класів:

- CSS перед назвою селектора класу обов'язково ставиться крапка (але при присвоєнні класу в HTML-документі ця крапка не потрібна);
- в назві класів можна використовувати тільки букви латинського алфавіту, дефіс, підкреслення, цифри;
- назва класу завжди повинна починатися з літери (правильні варіанти назв: .intro, .img-border, .nav_menu_01; неправильні: .2color,.-link, ._divider);
- назви класів CSS чутливі до регістру, тому класи на зразок .review і .Review будуть сприйматися як два окремих.

Селектори ID

Ідентифікатор визначає унікальну назву елемента. Записується він майже так само, як і клас, тільки в CSS замість крапки ставиться символ решітки #:

```
#footer {
width: 100%;
}
```

У HTML-документі ідентифікатор присвоюється за допомогою атрибута id:

Існує кілька відмінностей між ідентифікатором і класом:

-ID - це унікальна назва елемента на веб-сторінці, яка повинна зустрічатися на ній тільки один раз (наприклад, шапка сайту і підвал: id =

"header" і id = "footer"), в той час як клас може назначатися до кількох елементів з метою відрізняти їх від інших;

- ідентифікатори зручні для JavaScript-розробників, оскільки дозволяють отримати швидкий доступ до елементу DOM з скриптів (багато в чому саме тому необхідно, щоб ID зустрічався на сторінці лише один раз);

- кожне правило CSS має свій пріоритет (від пріоритету залежить, яке з правил отримає більш високий пріоритет при виконанні). Ідентифікатор має більшу вагу, ніж клас, тому, якщо елементу присвоєно і ID, і клас, перевага віддається ID. приклад:

```
<pid = "text" class = "content"> текст </ p>
#text {
color: yellow;
}
.content {
color: blue;
}
```

У ID вищий пріоритет, тому колір тексту буде жовтим (yellow).

За допомогою ідентифікаторів можна ставити якірні посилання на певні елементи веб-сторінки. Досить привласнити цьому елементу id:

<h3 id = "description"> Опис </ h3>

І потім дати на нього посилання виду: http://site.com/category/page/#description.

Групові селектори

Щоб все записати набагато простіше й коротше, ніж описувати кожен селектор окремо, треба перерахувати імена через кому та створити таким чином груповий селектор:

```
p, h1, h2, h3 {
 font-weight: bold;
 }
```

Звичайно, в перерахуванні можуть брати участь не тільки селектори тегів, а й класи, та ідентифікатори. А якщо з якоїсь причини вам необхідно створити стиль абсолютно для всіх елементів веб-сторінки, можна скористатися

універсальним селектором CSS, для позначення якого використовується символ зірочки *.

CSS Flexbox як інструмент побудови контейнерів сайту

CSS Flexbox (Flexible Box Layout Module) - модуль макета гнучкого контейнера - являє собою спосіб компонування елементів, в основі лежить ідея осі.

Flexbox складається з гнучкого контейнера (flex container) і гнучких елементів (flex items). Гнучкі елементи можуть вибудовуватися в рядок або стовпчик, а вільний простір розподіляється між ними різними способами.

Створення флекс-контейнера

Переведення контейнера в стан «флекс» є досить простим. Все, що необхідно - це змінити властивість дисплею на flex чи inline-flex як показано: display: flex; чи: display: inline-flex;

Після зміни властивості display на одну з вказаних вище, елемент стає флекс-контейнером, а вміст - флекс-контентом. Зміна властивості дисплею на «flex» перетворює контейнер на блоковий елемент, тоді як зміна на «inline-flex» перетворює контейнер на інлайн-елемент.

CSS-препроцесори (наприклад, Less, Sass і Stylus) розширюють можливості стандартного CSS. Дозволяють використовувати змінні, вкладені правила, mixins, вбудований імпорт тощо. Що в свою чергу допомагає тримати CSS-стилі добре організованими, робить їх більш зрозумілими й компактними. CSS-препроцесори ще називають динамічними мовами стилів.

CSS препроцесор – це програма, яка компілює написаний код (з використанням спеціального синтаксису) в чистий CSS код. При цьому, препроцесор надає розробнику нові можливості та функції.

Завдання:

1 . Створити селектори різних типів і групові у файлі stile.css.

2. Продемонструйте с**творення флекс-контейнера**.

3. Продемонструйте роботу будь-якого CSS-препроцесору.

Контрольні запитання:

1. Що представляє собою процес веб-розробки?

2. Що представляє собою HTML?

3. Що представляє собою CSS?

4. Як використовують сервери механізм Cookies?

5. Що називають фронтендом?

6. Що представляє собою бекенд?

7. Що представляє собою фреймворк?

8. Які вимоги до сучасного редактора коду?

9. Що представляють собою HTML-теги?

10. Що представляє собою елемент HTML?

11. Що означають елементи: <html>, <body>, <p>?

12.Що означає DOCTYPE у HTML-документі?

13. Що представляє собою заголовок HTML-документа?

14. Яку частину називають тіло HTML-документа?

15. Що визначають <header>, <nav>, <section>, і <footer> елементи?

16. Яка основна мета використання семантичних тегів?

17. Що представляють собою HTML-форми?

18. Які методи передачі даних форми використовуються?

19. За допомогою якого елемента виконується підключення стилів до файлу?

20. З чого складаються CSS-правила?

21. Як визначаються селектори класів?

22. Що представляють собою селектори ID?

23. Що представляє собою CSS Flexbox?

24. Які функції виконують CSS-препроцесори?

Самостійна робота №5

Тема: Можливості й обмеження JavaScript. Особливості програмування мовою JavaScript в DOM

Теоретичні відомості.

JavaScript (JS) - об’єктно-орієнтована мова програмування сценаріїв. Оновлення окремих блоків веб-сторінки, інтерактивні карти, анімування графіки, прокручування відео в медіапрогравачі та не тільки можна реалізувати за допомогою цієї мови.

JavaScript має надзвичайно багато застосувань. Можна розпочати з малого: створити "каруселі", галереї зображень, динамічні макети сторінок, відповіді на натиски кнопок, тощо. Із досвідом, стає можливим створювати ігри, 2D та 3D графіку, складні застосунки з використанням баз даних та багато іншого!

JavaScript є мовою, що інтерпретується (англ. interpreted programming language). Це означає, що їй потрібен інтерпретатор для роботи. Веб-браузер - один з таких інтерпретаторів.

З появою Node.js ви можете розробляти на JavaScript не лише клієнтську частину, але й серверну. В таких умовах розробник(и) спроможні писати проект лише на одній мові, що має свої плюси.

Використовуючи такі інструменти, як Electron, React Native та інші, JavaScript дозволяє створювати програми для настільних комп'ютерів, мобільні додатки, веб-додатки тощо.

JavaScript, можна використовувати он-лайн редактори, які дозволяють одразу в браузері писати фрагменти HTML/CSS/JavaScript коду:

- jsfiddle.net;
- jsbin.com;
- codepen.io.

Зручно, бо не потрібно нічого встановлювати й результат доступний звідусіль.

Властивості JS

- Перша властивість - виконання на клієнтському комп'ютері. Якщо ви знаєте, як взагалі працює інтернет, то вам повинно бути відомо, що веб-сторінки, які відображає браузер, створюються (або просто зберігаються) на іншому комп'ютері, так званому сервері. Браузер надсилає запит на сервер і

отримує у відповідь HTML-код сторінки. В цьому випадку браузер називається клієнтом. Головне - розуміти, що після того, як сторінка віддана браузеру, сервер вже не може змінити її вміст.

У випадку ж з JavaScript програми, а точніше - скрипти - виконуються прямо в браузері. Це дає таким скриптам можливість отримувати доступ до завантаженої сторінки, і змінювати її.

- Друга властивість - вбудованість. Для того щоб виконувати скрипти, написані на мові JavaScript, не потрібно ніяких додаткових програм - все необхідне для роботи скрипта вже є в браузері.

В різних браузерах JavaScript поводиться трохи по-різному. Це не стосується самої мови - одні й ті ж конструкції будуть виконуватися однаково. Вся справа в засобах, які браузер надає скрипту - так, наприклад, добре скрипти працюють в браузері MozillaFireFox, але видають помилки в інших браузерах, наприклад, в AppleSafari. Що правда, ці браузери вже не є такими популярним в наш час і весь код написаний на JS будемо розглядати в найбільш популярному браузері GoogleChrome.

Незважаючи на те, що мова JS найчастіше використовується для невеликих вставок в сторінки, це мова програмування досить потужна. Вона практично не поступається вже згаданій мові Java. Так, наприклад, на цій мові можна створювати (звичайно, не без участі браузера) програми, які мало чим відрізняються від звичних, так званих десктопних.

Щоб не бути голослівними, наведемо приклади:

– Google.com. Ця фірма використовує мову JavaScript в безлічі своїх продуктів. Так, наприклад, відома пошта Gmail практично повністю побудована на JS. А такі проекти, як GoogleMaps або GoogleDocs взагалі неможливі без застосування JavaScript.

– ExtJS.com. Тут можна подивитися, на що здатний JS - найкращим прикладом буде WebDesktop, цілком написаний на цій мові.

Крім наведених сайтів, можна було б назвати ще тисячі, які в тій чи іншій мірі використовують JS.

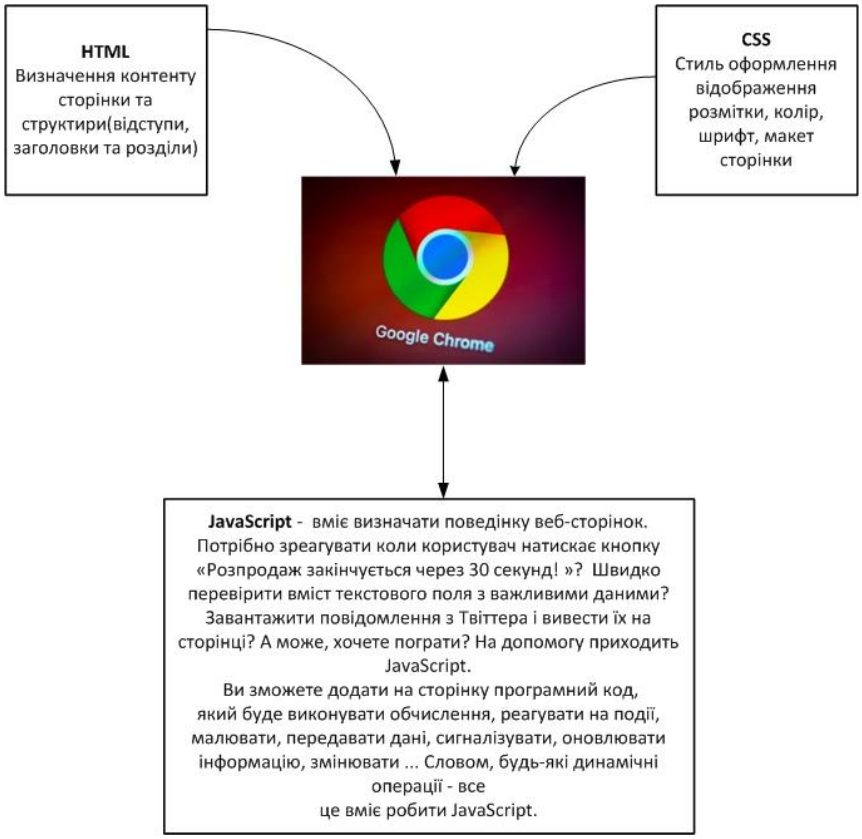

Рисунок 5.1 - Місце JavaScript у програмуванні веб сторінок

JavaScript займає особливе місце в світі програмування (рисунок 5.1). Типова класична програма з'являється наступним чином. Пишеться код, компілюється, проводиться компоновку й встановлюється на комп'ютер.

Написання та виконання коду

Сторінка створюється, зазвичай: з контентом HTML і стильовим оформленням CSS. На сторінку додається код JavaScript. За аналогією з HTML і CSS всі компоненти можна розмістити в одному файлі або ж виділити код JavaScript в окремий файл, який включається в сторінку. Браузер виконує код, як тільки зустріячає його на сторінці.

Включення коду на HTML сторінку

Виконується за допомогою елемента <script>…. </script>

Візьмемо одну веб-сторінку і визначимо для неї динамічну поведінку в елементі <script>.

```
<!DOCTYPE html>
<htmllang="en">
<head>
    <metacharset="UTF-8">
```

```
        <title>Title</title>
        <script>
        alert('HelloWorld!')
        </script>
    </head>
    <body>
    </body>
    </html>
```

В розділ <head> сторінки було додано елемент <script> та в ньому записано фрагмент коду на мові програмування JS.

Після збереження і завантаження сторінки буде виконано код, а саме викликається модальне вікно з повідомленням 'HelloWorld!'.

Код JavaScript складається з команд. Кожна команда описує невелику частину операції, що виконується, а весь набір команд визначає поведінку сторінки. Команда виконує невелику частину роботи, наприклад, визначає змінні в яких буде зберігатися інформація.

Іншим варіантом підключення JavaScript є створення окремого файла скрипта. Для цього перейдіть до тестового сайту та створіть нову теку з іменем "scripts" (без лапок). Тоді, всередині нової теки для скриптів, що з'явилась, створіть новий файл з назвою main.js. Збережіть його в теці scripts.

Далі, у файлі index.html введіть наступний елемент в новому рядку просто перед закриваючим тегом </body> :

<script src="scripts/main.js"></script>

Загалом, це те ж саме, що і <link> елемент для CSS - він додає JavaScript до сторінки, тож це може впливати на HTML (також на CSS, та будь що інше на сторінці).

Тепер додайте цей код до файлу main.js:

Var myHeading = document.querySelector('h1');

myHeading.textContent = 'Helloworld!';

Наприкінці, переконайтесь, що файли HTML та JavaScript збережені та приєднані до index.html в браузері.

Результатом виконання коду, наведеного вище буде заміна тексту заголовка на "Helloworld!", використовуючи JavaScript. Це було виконано за допомогою функції querySelector() (en-US) щоб отримати посилання на заголовок та зберегти його у змінній myHeading.

Після цього, привласнюється значення змінної myHeading властивості textContent (en-US) (яка містить контекст заголовка), а саме "Helloworld!".

!!!Причиною, чому <script> елемент вставляють ближче до низу HTML файлу, є те, що HTML завантажується браузером в порядку того, як він прописаний в файлі. Якщо JavaScript завантажено першим, а очікується, що він має впливати на HTML під ним, він може не працювати, якщо JavaScript завантажується раніше, ніж HTML, над яким він має здійснювати якісь маніпуляції. Тож, писати підключення JavaScript наприкінці коду HTML це найкраща стратегія.

Змінні та значення

Змінні - контейнери, у яких можна зберігати значення. Можливо розпочати з оголошення змінної за допомогою ключового слова var, після якого вказавши ім'я, яким потрібно назвати змінну.

Змінні потрібні для всіх цікавих у програмуванні речей. Якби змінні не могли змінюватись, то ви не могли б робити ніяких динамічних речей, як то персоналізовані вітальні повідомлення чи зміна відображуваних в галереї зображень.

В JavaScript використовуються змінні, які призначенні для зберігання значень.

Крім чисел, рядків і логічних значень змінні можуть зберігати й інші дані, але незалежно від виду даних всі змінні створюються за одними правилами.

Цифри 0..9, букви латинського алфавіту, нижнє підкреслення та знак долара. Не можна розпочинати з цифри.

Вибір імен змінних

У змінної є ім'я і значення.

- Ім'я змінної має починатися з літери, підкреслення або знака долара.

- Потім можуть слідувати літери, цифри, підкреслення і знаки долара - в будь-якій кількості.
- Як правило, змінну визначають зі слів із великих літер. Наприклад: FootSize (верблюжа нотація).
- Також в JS може бути довільна довжина змінних.

JavaScript це мова чутлива до регістру - myVariable, це не те саме, що myvariable чи MYvariable. Якщо ви маєте проблеми з кодом, можливо, варто перевірити регістр.

У JavaScript є три види оголошення змінних:

– var

– let

– const

Найчастіше використовують var.

Слово **let**- оголошує змінну з блоковою областю видимості. Це дозволяє оголосити локальну змінну з обмеженою поточним блоком { } коду областю видимості. На відміну від **var**, яке оголошує змінну глобально або локально в усій функції незалежно від області блоку.

let var1 [= value1] [, var2 [= value2]] [..., varN [= valueN]];

Приклад:

```
var x=5;
if(x>3){
    lety='недвійка'; //зміна x буде видимою лише у блоці if {...}
    console.log(y);
    }
alert(y); //ReferenceError:  y isnot defined - змінної y не існує
```

Приклад різниці між var і let:

```
function LET(a){
    if(a<0){
        let a=0;
        console.log(a); //0
    }
```

```
        console.log(a); //-9
    }
    function VAR(a){
        if(a<0){
            var a=0;
            console.log(a);  //0
        }
    VAR(-9);
```

LET(-9);

Подвійне оголошення змінної у одному ж блоці призведе до синтаксичної помилки коду:

let x=5;

alert(x);

let x=12; //SyntaxError: Identifier'x'hasalreadybeendeclared

Оголошення змінної такої ж самої назви у різник блоках не призводить до помилки:

```
if(10>1){
let x=5; //оголошення змінної x в межах блоку if {...}
alert(x);
}
let x=12;
alert(x);
```

Слово **const** - оголошує константу, доступну лише для читання.

constnames = values [, name2 = value2 [, ... [, nameN = valueN]]];

Параметри:

name - назва константи;

values - значення константи, може містити будь який тип даних.

Оголошення const створює посилання на значення, яке змінити не можна. Константа - це та сама змінна, тільки її присвоєння не можна перезаписати після оголошення (це може призвести до виникнення помилки або браузер просто проігнорує зміну значення й у константи значення не зміниться). Також константу не можна повторно оголошувати.

!!!Константи з'явилися у ECMAScript 6 відповідно старі браузери їх не підтримують!!!

Константи зручно використовувати, тоді коли необхідно, щоб значення змінної не змінювали в подальшому в коді.

Зверніть увагу, що в константі в JavaScript не можливо змінити присвоєне посилання, але можливо змінити значення, на яке посилається константа. Прикладом цього може бути присвоєння об'єкта константі й зміна значення об'єкта:

```
const a={test:'js'};
a.test='JavaScript';
alert(a.test);
```

Змінні JavaScript задаються в наступному форматі й можуть бути оголошені в будь-якому місці коду скрипта:

var cnt;

Можливе завдання початкового значення змінної при її оголошенні:

var name = "значення";

Можна звертатись до змінної, просто вказавши її ім'я:

myVariable;

Після надання змінній значення, ви можете пізніше змінити його:…

var myVariable = 'Bob';

myVariable = 'Steve'; …

JavaScript слабко типізована мова. А саме, якщо це число, то до нього відносяться і цілі числа, і дробові, з плаваючою точкою і т.д.

Основні три типи: Boolean() (використовуються у випадку, коли змінна приймає 2 значення false або true), Number (число), String (текст). Та три допоміжні типи: Undefined (невизначений тип), null (абсолютно пуста змінна) і Object (об’єкти – окрема розмова).

Також змінні можуть змінювати типи.

Тип змінної задається, як правило, форматом її значення. У процесі виконання програми скрипта та сама змінна може мати різний тип залежно від значення, що в ній зберігається. Описана нами змінна name очевидно буде мати

рядковий тип. У деяких випадках можемо створювати типізовані змінні для породження екземплярів того або іншого класу об'єктів:

today=newDate();

Тут породжений екземпляр об'єкта Date, що дозволяє здійснювати роботу з датою.

Оператор виведення типу змінних alert(typeof змінна).

Приклад створення змінних:

var win=3;

Так команда з ключовим словом var створює змінну з іменем win та привласнює числове значення 3.

var name = 'Тарас';

Змінній з іменем name привласнюється послідовність символів‘Тарас’, яка називається рядком.

var siElig=false;

Змінній siElig привласнюється значення false (логічна змінна).

var losers;

Змінна оголошена без знаку дорівнюється і власне значення, якщо змінну планується якось використовувати далі в програмі.

Змінні можуть містити значення різних типів.

Область видимості змінних

У JavaScript змінні за областю видимості поділяються на: глобальні й локальні.

Глобальна змінна - це змінна, яка доступна для всього коду веб-сторінки.

Локальна змінна - це змінна, яка доступна в певній функції або блоку коду. Якщо об'являти змінну за допомогою var, то змінна локальна в функції, якщо ж за допомогою let, const, то змінна локальна у блоці { ... }.

Глобальна змінна існує доки завантажена веб-сторінка, якщо перезавантажити сторінку, то змінна обновиться. А локальна змінна існує доки виконується функція або є блок.

Синтаксис JS

- Кожна команда завершується символом ";":

x = x + 1;

- Однорядковий коментар починається з двох косих рис (//). Коментарі містять інформацію про код для вас і інших розробників. У програмі вони не виконуються.

Оператори виведення даних в JavaScript

Будь-яка мова програмування немислима без операторів виводу. JavaScript не є виключенням. Виведення даних на екран може відбуватися різними способами. При цьому оператори виводу оптимізовані для найбільш зручного застосування. Найбільш простим є застосування оператора alert(). Аргументом оператора може бути будь-який рядковий вираз. Якщо аргумент має нерядковий тип, то він переводиться в рядковий. Результатом виконання оператора alert є виведення на екран діалогового вікна, вмістом якого є отримане значення. При цьому діалогове вікно буде очікувати натискання користувачем кнопки ОК. Тільки після виконання цієї дії виконання програми й відображення сторінки буде продовжено. Виведення за допомогою вікна оператора alert досить зручно використовувати для контролю значень змінних на тому або іншому етапі виконання програми, тобто при налагодженні.

Також можна застосовувати виведення на консоль, як це наведено в наступному прикладі:

console.log('Hello World!');

Завдання:

1. Створіть сторінку html і продемонструйте підключення скрипту на ній з виведенням повідомлення про автора сторінки (Наприклад: Шевченко О.О. вітає Вас!)

2. За допомогою JS виведіть на консоль повідомлення про автора програми.

3. Виконайте прості арифметичні дії за допомогою JS і винесіть в окремий файл.

Контрольні запитання:

1. Що представляє собою JavaScript?

2. Які властивості JS?

3. Які є способи розміщення коду JavaScript на HTML-Сторінці?

4. За допомогою якого елемента виконується включення коду JS на HTML сторінку?

5. Як можна оголошувати змінні в JS?

6. Що оголошує слово let?

7. Чим let відрізняється від var?

8. Доки існує глобальна змінна?

9. Доки існує локальна змінна?

10. Який синтаксис в JavaScript?

11. Які оператори виведення даних в JavaScript?

12. Які оператори введення даних в JavaScript ?

13. Для яких операцій існує об'єкт Math у JS?

Самостійна робота №6
Тема: Адаптивний дизайн Bootstrap

Теоретичні відомості.

Bootstrap - це потужний фронтенд-фреймворк для створення адаптивних, мобільних сайтів завдяки своїй системі сіток (grid system) та підходу **Mobile First**, що дозволяє легко створювати гнучкі макети за допомогою CSS-класів, які автоматично підлаштовуються під різні розміри екранів (телефони, планшети, десктопи). Його адаптивність ґрунтується на медіа-запитах (media queries) та готових класах для контейнерів, колонок і компонентів, забезпечуючи єдиний вигляд та функціональність на всіх пристроях.

Для методу CDN – підключення фреймворку використання нам буде необхідно постійне підключення до мережі Інтернет.

Тому, якщо ми хочемо використовувати фреймворк без постійного підключення нам необхідно скачати одну із версій, запропоновану на сайті перейшовши на відповідну вкладку Download за адресою:

https://getbootstrap.com/docs/4.6/getting-started/download/

Файлова структура

Як тільки скомпільована версія Bootstrap 4 буде завантажена, розархівуйте ZIP-файл, і ви побачите наступну структуру файлів / каталогів:

Рисунок 6.1 – Структура каталогів Bootsrap

Як видно, скомпільовані CSS і JS (bootstrap.*), А також скомпільовані й мінімізовані CSS і JS (bootstrap.min.*).

Розглянемо, для чого призначені файли Bootstrap:

- bootstrap.css - головний файл Bootstrap, що визначає оформлення css елементів бібліотеки;
- bootstrap.min.css - містить ті самі стилі, що і попередній файл, але стиснутий для швидкого завантаження;
- bootstrap - theme.css - файл перевизначає стандартні стилі оформлення елементів Bootstrap, можна використовувати як часткову заміну bootstrap.css;
- bootstrap - theme.min.css - містить ті самі стильові описи, що і попередній файл, але стиснутий для швидкого завантаження;
- bootstrap-theme.css.map - дозволяє працювати з файлами так, начебто вони не були стиснуті;
- bootstrap.css.map - дозволяє працювати з файлом bootstrap.min.css, так наче він не був стиснутий;
- bootstrap.js - файл, що відповідає за динамічні можливості бібліотеки;
- bootstrap.min.js - стиснутий файл bootstrap.js;
- glyphicons - halflings - regular.eot - у даному файлі знаходяться векторні іконки бібліотеки для браузера Internet Explorer;
- glyphicons - halflings - regular.svg - у даному файлі знаходяться всі шрифти й іконки, що в попередньому, але у форматі векторної графіки svg;
- glyphicons - halflings - regular.ttf - стандартний файл шрифтів;
- glyphicons - halflings - regular.woff - стиснутий файл шрифтів.

Для початку роботи слід підключити bootstrap.css і bootstrap.js. Також для повноцінної роботи бібліотеки слід підключити JQuery.

Завантажений файл дистрибутива поміщаємо в теку з нашим проектом та перейменовуємо її в Bootstrap.

Після розархівування відповідного файлу дистрибутива фреймворку ми отримаємо дві теки CSS та JS.

Для режиму розробки підключаємо не стиснуту версію, для цього перед тегом </head> прописуємо даний код:

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

Далі необхідно підключити файли роботи з JS, для цього підключивши відповідний файл перед закриваючим тегом </body>:

<script src="bootstrap/js/bootstrap.js"></script>

Щоб перевірити чи підключився фреймворк запустимо наш проект і зайдемо в консоль браузера (перед цим закоментувавши файли підключення CDN). Буде видно, що для нашого тега h1 підключаються відповідні правила з файлу bootstrap.css.

Адаптивний дизайн (англ. Responsive Web Design) об'єднує в собі три методики - гнучкий макет на основі сіток, гнучкі зображення та медіазапити. Гнучкість макета базується на використанні відносних одиниць вимірювання замість фіксованих піксельних значень. Проблема гнучких зображень вирішується за допомогою правила img (width: 100%; max-width: 100%;) для всіх картинок на сайті. Це правило гарантує, що зображення ніколи не будуть ширше, ніж їхні контейнери і ніколи не перевищать своїх справжніх розмірів на великих екранах. Медіазапити змінюють стилі на підставі характеристик пристрою, пов'язаних з відображенням контенту, включаючи тип, ширину, висоту, орієнтацію й характеристики екрану. За допомогою медіазапитів створюється адаптивний дизайн, в якому до кожного розміру екрану застосовуються відповідні стилі.

Адаптивний мета-тег

Bootstrap розроблявся спочатку як мобільний (mobile first), тобто його налаштування попередньо оптимізовані під мобільні пристрої, а потім за допомогою медіа-запитів підганяється масштаб компонентів як необхідно на інших пристроях. Необхідно помістити цей код між тегами <head>:

<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

Після вставки даного коду сторінка, що відкрита в браузері, буде підлаштовуватися до розміру екрану й блоки будуть вже адаптивні. Це головний мета-тег, який відповідає за адаптивність сторінки.

Основні аспекти адаптивності у Bootstrap:

- Mobile First підхід: Спочатку стилі пишуться для маленьких екранів (десктопні стилі додаються пізніше через медіа-запити), що забезпечує оптимальну швидкість завантаження та роботу на мобільних пристроях.

- Система сіток (Grid System): Використовує класи типу col-md-6, col-lg-4, що визначають, як елементи будуть розташовуватися на різних розмірах екранів. Має 5 рівнів адаптивності (breakpoints): xs (за замовчуванням), sm, md, lg, xl, що дозволяють точно контролювати розбиття на колонки (наприклад, 12 колонок у рядку).

- Класи контейнерів: container (фіксована ширина) та container-fluid (повна ширина) задають основні межі для вмісту, які адаптуються залежно від розміру екрана.

- Медіа-запити (Media Queries): Вбудовані у фреймворк для автоматичного застосування стилів при певній ширині екрану, забезпечуючи гнучкість макета.

- Утиліти та компоненти: Готові елементи (навігаційні панелі, картки, форми) також є адаптивними та легко кастомізуються, що прискорює розробку.

Адаптивні зображення: додати до тегу img стильовий клас img-responsive. Для кожного зображення додавати цей клас не дуже зручно, тому можна вчинити по-іншому – створити свій style.css, в якому вже для всіх картинок вказати такі правила, щоб вони були адаптивні. Правила: Клас img-responsive додає картинкам наступні властивості:

```
img{
max-width: 100%;
height: auto;
display: block;
}
```

Відповідно, можна прописати їх не для класу, а просто для всіх зображень.

Фреймворк пропонує простий приклад зробити **адаптивною таблицю**: Просто помістіть всю таблицю div з класом table-responsive. Тобто це буде адаптивний контейнер для таблиці. Якщо вона стиснеться до своїх максимальних розмірів, далі горизонтальний скрол буде відображатися у всієї сторінки, а тільки у блоку з таблицею.

Простий приклад зробити **адаптивне відео**:

Зазвичай відео вставляється на сторінку в тег iframe, де вказується адреса до нього. Щоб адаптувати відео до перегляду на будь-яких екранах, оберніть його в div, якому вкажіть 2 класи. Перший – embed-responsive. Другий буде уточнюючим: embed-responsive-16by9 або embed-responsive-4by3.

Відповідно, вам потрібно визначити дозвіл відео, і потім вибрати, який клас підходить. В більшості випадків буде 16 на 9.

Завдання:

1. За допомогою Bootstrap створіть на сторінці адаптивне зображення.
2. За допомогою Bootstrap створіть на сторінці адаптивну таблицю.
3. За допомогою Bootstrap створіть на сторінці адаптивне відео.

Самостійна робота №7
Тема: Основні аспекти PHP у веб-програмуванні

Теоретичні відомості.

PHP(Hypertext Preprocessor) - це популярна, серверна мова програмування з відкритим кодом, створена для розробки динамічних веб-сторінок та веб-застосунків, яка виконується на сервері, генерує HTML-код та відправляє його в браузер, інтегрується з базами даних (наприклад, MySQL) і підтримується більшістю хостинг-провайдерів, що робить її ідеальним інструментом для бекенд-розробки, особливо з використанням фреймворків типу Laravel.

Основні аспекти PHP у веб-програмуванні:

- **Серверна мова**: Код PHP виконується на сервері, а не в браузері користувача, що забезпечує швидкість та безпеку, оскільки користувач не бачить вихідний PHP-код.
- **Динамічний контент**: Дозволяє створювати контент, що змінюється залежно від дій користувача, даних у базах даних або часу.
- **Інтеграція з HTML**: Легко вбудовується в HTML-код, дозволяючи створювати інтерактивні сторінки.
- **Робота з базами даних**: Ідеально підходить для взаємодії з базами даних (MySQL, PostgreSQL), що є основою для більшості веб-застосунків.
- **CMS та фреймворки**: На PHP працюють популярні системи керування контентом (CMS) як WordPress, Joomla, Drupal, а також потужні фреймворки (Laravel, Symfony), які прискорюють розробку.

Сфери застосування:

- Розробка сайтів (від простих до великих порталів).
- Інтернет-магазини.
- Соціальні мережі та блоги.
- API для мобільних додатків.
- Скрипти для автоматизації завдань (через командний рядок).

Ця мова програмування має ряд значних переваг, які роблять її основним інструментом для розробки серверної частини додатків та сайтів.

1. PHP має відкритий код і безкоштовна.

2. PHP підтримується за замовчуванням майже будь-яким сучасним хостинг-провайдером, що є запорукою доступної вартості серверів для додатку або сайту, написаного цією мовою.

3. З використанням PHP створено велику кількість систем управління контентом (CMS) для сайтів, таких як WordPress, Joomla, Drupal — системи загального призначення, OpenCart, Magento — системи для управління Інтернет-магазинами. Це лише найбільш розповсюджені та безкоштовні CMS, їх реальна кількість — безкоштовних та платних — дуже велика.

4. Створено значну кількість високоефективних інструментів та фреймворків, що засновуються на PHP. Вони дозволяють розробникам створювати додатки швидше, при цьому підтримуючи код програми в чистоті. До таких інструментів належать Symfony, Laravel, Yii2, CodeIgniter 4 — це лише найбільш розповсюджені.

5. Висока швидкість роботи. Останні версії PHP вражають швидкістю відпрацювання скриптів у порівнянні з іншими мовами програмування.

6. Багатоплатформність. Працює на Windows, Linux, macOS.

7. Простота вивчення: Відносно простий синтаксис, схожий на C/Java, робить його доступним для новачків.

PHP, як мова програмування має ряд недоліків. Справа в тому, що мова має дуже низький поріг входження. Це обумовлено тим, що спочатку він створювався як інструмент для розробки домашніх сторінок — такий, який кожен зміг би освоїти. На перший погляд, в цьому немає нічого поганого. Але насправді ця особливість дозволяє входити в PHP розробку тим, хто не має достатнього досвіду в програмуванні. А створення програмного забезпечення недосвідченим розробником може призвести до проблем з безпекою та технічною підтримкою проекту в подальшому.

Таким чином, PHP — це чудовий інструмент для розробки серверного програмного забезпечення, який може забезпечити високу швидкодію та безпеку, а також простоту подальшої технічної підтримки, але лише в тому випадку, коли цей інструмент знаходиться в руках справжнього майстра.

Завдання:

Відповісти на запитання:

1. Чим PHP відрізняється від інших мов?
2. Які основні аспекти PHP?
3. Які сфери застосування PHP?
4. Які основні переваги застосування PHP?
5. Які є недоліки PHP?

Самостійна робота №8

Тема: Популярні фреймворки. Переваги та недоліки використання фреймворків

Теоретичні відомості

Фреймворки

Фреймворк (англ. framework) - це набір всіляких бібліотек (інструментів) для швидкої розробки повсякденних (рутинних) завдань. Головна мета фреймворку - надати програмісту зручне середовище для проекту з великим і добре розширюваним функціоналом.

Переваги фреймворків:

- фреймворки максимально полегшують роботу розробників, зменшуючи час розробки;

- завдяки фреймворкам, код виходить структурованим, зрозумілим і доступним для повторного використання;

- фрейморки використовують популярні шаблони проектування (наприклад, Singleton, MVC).

- надають безпеку.

Програмні фреймворки полегшують життя розробникам веб-сайтів та програм, дозволяючи їм контролювати процес розробки програмного забезпечення за допомогою єдиної платформи.

Якщо в проекті ви використовуєте фреймворк, то більша частина коду й структура проекту будуть базуватись на цьому каркасі. Ви отримаєте, як інструмент, добре спроектовану систему, оминаючи чимало підводних каменів, про які ви навіть можете не підозрювати на початку вивчення веб-розробки.

Створення свого власного міні-фреймворка - це задача, з якою стикається кожен фронтенд-розробник. Зазвичай він складається з тих правил і функцій, які повторювалися у всіх недавніх проектах. Після того, як вони будуть зібрані в одну бібліотеку, їх буде легше разом підключити до нового проекту й користуватися готовими рішеннями. У бібліотеку може входити сітка з колонками, в яких знаходиться будь-який контент, стандартні правила для спрайтів, зовнішніх відступів, заголовків і т.д.

Ось десять популярних Бекенд і Фронтенд фреймворків:

Фреймворк	Категорія	Мова програмування	Додатки
React	Фронтенд	Javascript	Facebook Yahoo Khan Academy
Angular	Фронтенд	Typescript	Gmail Forbes PayPal
Vuejs	Фронтенд	Javascript	Chargebee Yousign Infermedica
jQuery	Фронтенд	Javascript	Upwork LinkedIn Udemy
Emberjs	Фронтенд	Javascript	TED Netflix Square
Django	Бекенд	Python	National Geographic Mozilla Pinterest
Laravel	Бекенд	PHP	Deltanet Travel Neighborhood Lender World Walking
Ruby on Rails	Бекенд	Ruby	Twitter Zendesk Github
Cake PHP	Бекенд	PHP	Coconala Goodfirms Croogo
Express JS	Бекенд	Node	Uber Groupon GoDaddy

Якщо в розробці проекту бере участь кілька професійних фронтенд-розробників, то подібні фреймворки потрібно стандартизувати. Якщо так, то перевага віддається вже стандартизованим фреймворками. Одним із найбільш популярних зараз є - фреймворк Bootstrap.

Переваги Bootstrap:

– Швидкість роботи - створення макетів з Bootstrap займає менше часу завдяки великому набору готових до використання елементів.

– Гнучкість - додавання нових елементів не порушує загальну структуру завдяки сітці, що динамічно змінюється.

– Легка змінюваність - правка стилів досягається завдяки додаванню нових CSS правил, які скасовують існуючі. При цьому, вам не потрібно використовувати атрибути типу !Important.

– Велика кількість шаблонів (буде розглянуто далі).

– Величезне співтовариство прихильників / розробників.

– Широкий спектр застосування - Bootstrap використовується для створення тем майже для будь-якої CMS (Magento, Joomla, WordPress або будь-якої іншої), включаючи односторінкові Лендінги.

– Зрозуміла та доступна офіційна документація.

– Bootstrap особливо популярний серед тих, хто займається створенням так званих «Лендингів» (посадочних / цільових сторінок).

Недоліки та виклики при роботі з React

При роботі з React можуть виникати наступні недоліки та виклики:

- Велика кількість концепцій: навчання та зрозуміння всіх основних концепцій React, таких як компоненти, стан, властивості, контексти тощо, може виявитися складним для початківців у веб-розробці;

- Комплексність стану: управління станом додатку у React може бути складним завданням, особливо для великих проєктів, що може викликати заплутаність та складність у розробці та підтримці коду;

- Потреба відслідковувати зміни: при роботі з компонентною архітектурою React доводиться активно відслідковувати зміни у структурі даних та їх вплив на рендеринг відображення, що може вимагати додаткового зусилля та уваги розробника;

- Великий екосистема та швидкі зміни: екосистема React постійно розвивається, і нові бібліотеки та інструменти появляються з великою швидкістю. Це може призвести до нестабільності та несумісності між версіями, а також до необхідності постійного оновлення навичок;

- Продуктивність: при створенні великих та складних додатків у React може виникати проблема з продуктивністю, особливо при рендерингу великої кількості даних або компонентів.

Вибір відповідного комплекту технологій для розробки веб-сайтів і

додатків може виявитися непростим завданням, але залежить від того, яка технологія поєднується з ним і яким варіантом використання програми.

Завдання:

1. Встановити фреймворк.

2. Відповісти на запитання:

1. Що представляє собою фреймворк?
2. Які є відомі фреймворки?
3. Які відомі переваги застосування фреймворків?
4. Які недоліки у фреймворків відомі?

Контрольні запитання:

1. Що представляє собою Bootsrap?

2. Які переваги Bootstrap?

3. Назвіть головні компоненти Bootstrap.

4. Яку роль відіграє сітка для макета?

5. В яких форматах може бути представлено відображення сповіщень у фреймворку Bootstrap?

6. Які види дизайнів для навігації містить Bootstrap?

7. Скільки класів має система Grid Bootstrap 4?

8. Які типи контейнерів використовується у Bootstrap?

9. Який клас слід використовувати для створення гумового (гнучкого) макета?

10. Який клас слід використовувати для створення відступів?

11. Для чого використовуються класи .order?

12. Які класи застосовують для блоків, які виглядають і розташовуються однаково на всіх пристроях будь-якого розміру?

13. Для яких цілей використовують клас *.no-gutters* для класу .row?

14. Який клас застосовують для створення колонок, що змінюють свою ширину за шириною вмісту?

15. Які є три класи для вирівнювання блоків по горизонталі, які вказуються для всього ряда .row?

16. Які два класи використовуються для вставки на сторінку відео?

17. Для чого застосовується компонент Alert?

18. Що представляють собою модальні вікна побудовані за допомогою HTML, CSS та JavaScript?

19. Що забезпечують картки Bootstrap (Cards)?

20. Який клас застосовують для створення адаптивного зображення, яке підлаштовується під розмір батьківського блоку**?**

21. За допомогою якого класу створюються таблиці в Bootstrap?

22. Який базовий елемент в Bootstrap?

23. За допомогою чого будується панель навігації в Bootstrap?

24. Які два режими відображення має компонент навігації в Bootstrap?

25. З чого складається блок посилань меню компоненту «NavBar»?

Рекомендовані джерела інформації

1. Web-програмування. Частина 1 (frontend) : навч. посіб. / В. В. Босько, Л. В. Константинова, К. М. Марченко, О. С. Улічев ; М-во освіти і науки України, Центральноукраїн. нац. техн. ун-т. - Кропивницький : ЦНТУ, 2022. - 208 с.

2. Мельник Р. Програмування веб-застосувань (фронт-енд та бек-енд) Видавництво: Львівська політехніка, 2018, 248с.

3. Resources for developers, by developers. MDN Web Docs [Електронний ресурс] – Режим доступу: https://developer.mozilla.org/en-US/docs/Web (дата звернення: 18.01.2022)

4. Build fast, responsive sites with Bootstrap [Електронний ресурс] – Режим доступу: https://getbootstrap.com/docs/5.1/getting-started/introduction (дата звернення: 18.01.2022)

5. Пасічник О.Г., Пасічник О.В., Стеценко І.В. Основи веб-дизайну Видавництво: Вид група BHV, 2009, 336с. [Електронний ресурс] – Режим доступу: http://school1k24.at.ua/10CLASS_WEB/OsnovyWebDis.pdf (дата звернення: 20.01.2022)

6. Методичні рекомендації до виконання лабораторних робіт з дисципліни «Web-програмування» для студентів денної та заочної форми навчання спеціальностей 123 «Комп'ютерна інженерія», 125 «Кібербезпека» та 122 «Комп'ютерні науки» / [уклад. : Є. В. Мелешко, В. В. Босько, Л. В. Константинова] ; М-во освіти і науки України, Центральноукраїн. нац. техн. ун-т. - Кропивницький : ЦНТУ, 2023. - 87 с.

7. Український веб-довідник [Електронний ресурс] – Режим доступу: https://css.in.ua/html/events (дата звернення: 20.01.2023).

8. Трегубенко І.Б. Сучасні технології програмування в Т-66 мережах [Електронний ресурс]/ І.Б.Трегубенко, Г.Т.Олійник, О.М. Панаско, 2-ге вид..- Черкаси: ЧДТУ, 2010.-175с.

9. Бородкіна І.Л., Бородкін Р. О. Web-технології та Web-дизайн : застосування мови HTML для створення електронних ресурсів. Видавництво: Ліра До, 2020, 212с.

10. HTML Living Standard [Електронний ресурс] – Режим доступу: https://html.spec.whatwg.org/multipage//(дата звернення: 20.01.2023).

11. Манако В., Манако Д., Данилова О., Войченко О.П. Основи будування сайтів Видавництво: Шкільний світ, 2006, 120с.

12. Stylus. Expressive, dynamic, robust CSS [Електронний ресурс] – Режим доступу:https://stylus-lang.com/ (дата звернення: 21.01.2022)

13. Sass. CSS with superpowers [Електронний ресурс] – Режим доступу:https://sass-lang.com/ (дата звернення: 21.01.2022)

14. Методичні вказівки до лабораторних робіт з дисципліни "Web-програмування" для студентів напряму підготовки 6.050101 “Комп’ютерні науки” всіх форм навчання / уклад. Т. В. Федорончак. – Запоріжжя : ЗНТУ, 2017. – 59 с.

15. Web-програмування. Лабораторний практикум [Електронний ресурс] : навч. посіб. для студ. спеціальності 125 «Кібербезпека» та 113 «Прикладна математика» / А. Ю. Шелестов, Н. М. Куссуль; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 1047 Кбайт). – Київ : КПІ ім. Ігоря Сікорського, 2021. – 61 с.

16. Client-side form validation [Електронний ресурс] – Режим доступу: https://developer.mozilla.org/en-US/docs/Learn/Forms/Form_validation (дата звернення: 20.01.2023).

17. Robin Nixon. Learning PHP, MySQL & JavaScript. 6th Ed. 2021, 825p.

18. Ричард Вагнер, Аллен Вайк. JavaScript. Київ: ДиаСофт, 2001 р., 464с.

19. Dave Crane, Eric Pascarello, Darren James. Ajax in Action. Manning; 1st edition (November 3, 2005) 680 pages.

20. jQuery Validation Plugin [Електронний ресурс] – Режим доступу: https://jqueryvalidation.org/documentation/ (дата звернення: 20.01.2022)