Міністерство освіти і науки України
Харківський національний університет імені В. Н. Каразіна

О. Г. Сузікова

ВЕБ ТЕХНОЛОГІЇ В ІНФОРМАЦІЙНОМУ ОБМІНІ

Методичні рекомендації
до практичних занять для здобувачів вищої освіти першого (бакалаврського) рівня з дисципліни «Обробка, зберігання та передача даних в сучасних інформаційних технологіях» за спеціальністю «Прикладна математика»

У 3 частинах

Частина 3: JavaScript

Електронний ресурс

Харків – 2025

УДК: 004.439 (072)
С 89

Рецензенти:

Ю. В. Ромашов – доктор технічних наук, доцент, професор кафедри комп'ютерно-інтегрованих технологій, автоматизації та робототехніки Харківського національного університету радіоелектроніки;

С. М. Севидов – кандидат фіз.-мат. наук, доцент кафедри математичного моделювання та аналізу даних ННІ комп'ютерних наук та штучного інтелекту Харківського національного університету імені В. Н. Каразіна.

Затверджено до розміщення в мережі Інтернет рішенням Науково-методичної ради Харківського національного університету імені В. Н. Каразіна (протокол № 11 від 25 червня 2025 року)

Сузікова О. Г.

С 89 Веб технології в інформаційному обміні : методичні рекомендації до практичних занять для здобувачів вищої освіти першого (бакалаврського) рівня з дисципліни «Обробка, зберігання та передача даних в сучасних інформаційних технологіях» за спеціальністю «Прикладна математика». У 3 ч. Частина 3: JavaScript [Електронний ресурс] / О. Г. Сузікова. – Харків : ХНУ імені В. Н. Каразіна, 2025. – (PDF 27 с.)

Методичні рекомендації містять початкові відомості про мову JavaScript. До кожного розділу наведені питання для самоконтролю та практичні завдання.

Методичні рекомендації призначені для здобувачів вищої освіти першого (бакалаврського) рівня, які вивчають дисципліну «Обробка, зберігання та передача даних в сучасних інформаційних технологіях».

УДК: 004.439 (072)

ЗМІСТ

Вступ

JavaScript (JS) є однією з ключових мов програмування у сфері сучасних інформаційних технологій. Вона забезпечує динамічність та інтерактивність веб-додатків, дозволяючи створювати анімації, реагувати на дії користувачів, оновлювати контент без перезавантаження сторінки та багато іншого. На відміну від HTML та CSS, які визначають структуру та стиль веб-сторінки, JavaScript додає їм функціональність, що робить його незамінним інструментом у веб-розробці.

Окрім веб-додатків, мова JavaScript знайшла широке застосування в бекенд-розробці (через середовище Node.js), мобільній розробці (з використанням фреймворків на кшталт ReactNative), робототехніці, штучному інтелекті та навіть у розробці ігор. Її універсальність і гнучкість дозволяють використовувати єдину мову як для клієнтської, так і для серверної частини застосунку, що значно спрощує процес розробки та підтримки програмних продуктів.

Знання JavaScript є важливим не лише для розробників, а й для аналітиків, тестувальників, UI/UX-дизайнерів та інших спеціалістів, які працюють у сфері інформаційних технологій. Навіть базове розуміння JS допомагає краще взаємодіяти з розробниками, розбиратися в технічних аспектах проєктів та знаходити ефективні рішення у сфері автоматизації та оптимізації робочих процесів. Це одна з найбільш затребуваних мов програмування, яка продовжує відігравати ключову роль у розвитку цифрових технологій.

Методичні рекомендації до практичних занять містять початкові відомості про мову JavaScript. До кожного розділу наведені питання для самоконтролю та практичні завдання. Рекомендуємо обов’язково виконувати ці завдання, щоб швидше освоїтися у практичному застосуванні JavaScript.

1. Контекст

JavaScript – це досить проста об'єктно-орієнтована мова, призначена для створення невеличких клієнтських і серверних додатків для Internet. Програми, написані мовою JavaScript, включаються до складу HTML-документів та поширюються разом із ними. Програми перегляду (браузери) розпізнають вбудовані в текст документа програми-вставки (script-коди) і виконують їх.

Наприклад, на JavaScript можна написати програми, які перевіряють введені користувачем дані або виконують якісь дії під час відкриття чи закриття документа. Програми можуть реагувати на дії користувача – натискання кнопок «миші», введення даних у форму або переміщення «миші» на сторінці. Більш того, JavaScript-програми можуть керувати браузером та атрибутами документа.

Мова JavaScript схожа за синтаксисом з мовою Java, за винятком об’єктної моделі, але не має статичних типів даних і не має строгої типізації. На відміну від Java, синтаксис JavaScript не спирається на поняття класів.

Натомість основою синтаксичних конструкцій у JavaScript є невеликий набір стандартних типів даних: числові типи, булевський тип, рядки; функції, які можуть бути як самостійними, так і методами об’єктів; об’єктна модель з великим набором вбудованих об’єктів зі своїми властивостями і методами, а також правилами, за якими в програмі користувача задаються нові об’єкти.

Для створення програм на JavaScript не потрібні додаткові засоби чи середовища. Потрібний лише браузер, що підтримує мову JavaScript, і текстовий редактор, в якому можна створювати HTML-документи.

Оскільки програма JavaScript вбудовується безпосередньо в текст HTML-документа, ви можете одразу побачити результати своєї роботи під час перегляду документа браузером і за необхідності внести зміни.

За допомогою скриптів можна динамічно керувати відображенням та вмістом HTML-документів. Можна записувати довільний HTML-код, що відображається на екрані, в процесі синтаксичного аналізу завантаженого браузером документа. За допомогою об'єкта Document можна генерувати документи «з нуля» залежно від попередніх дій користувача або будь-яких інших факторів.

JavaScript дозволяє контролювати роботу браузера. Наприклад, об'єкт Window підтримує методи, що дозволяють виводити на екран діалогові вікна, створювати, відкривати і закривати нові вікна браузера, задавати режими прокручування і розміри вікон і т.д.

JavaScript дозволяє взаємодіяти із вмістом документів. Об'єктна модель документа та об'єкти, що там містяться, дозволяють програмам читати частини HTML-документа і іноді взаємодіяти з ними. Взаємодія відбувається не зі змістовним текстом документа, а з його структурними елементами. Наприклад, можна отримати список гіпертекстових посилань, наявних у цьому документі. На даний момент широкі можливості взаємодії з вмістом документів забезпечує об'єкт Form та об'єкти, які він може містити: Button, Checkbox, Hidden, Password, Radio, Reset, Select, Submit, Text та Textarea.

JavaScript забезпечує взаємодію з користувачем. Важливою особливістю цієї мови є реалізована в ній можливість визначати обробники подій – довільні фрагменти коду, які виконуються при конкретних подіях, зазвичай – це дії користувача. JavaScript дозволяє використовувати як обробники подій будь-які нові попередньо задані функції.

Наприклад, можна написати програму, яка виведе у рядку стану спеціальне повідомлення, якщо користувач встановить покажчик «миші» на гіпертекстове посилання, або виведе на екран діалогове вікно з запитом на підтвердження виконання певної дії, або здійснить перевірку введеного користувачем значення, в разі помилки введення надасть відповідну діагностику та запропонує ввести правильне значення.

JavaScript дозволяє виконувати довільні математичні обчислення. Крім того, ця мова має розвинені засоби роботи зі значеннями дати та часу.

Як працює JavaScript? Вихідний код програм вбудовується у HTML-документ або завантажується з незалежних файлів. Програма викладається на сервер у вигляді вихідного коду в текстовій формі та надалі інтерпретується (без попередньої компіляції) браузером після завантаження її з сервера.

JavaScript – це об’єктна мова. В ній можна програмувати і без використання об’єктного програмування, а можна використовувати визначені вбудовані класи. Є теоретична можливість розширення цих класів, але вона ніколи не використовується.

На відміну від переважної більшості мов об’єктного програмування, в мові JavaScript структура об’єктів не задається однозначно описом класу, а є динамічною, тобто може змінюватися на етапі виконання програми. Об’єкти можуть динамічно отримувати нові поля та методи, або можуть динамічно змінюватися будь-які параметри існуючих полів і методів.

У мові JavaScript використовується вільна типізація: типи даних змінних не описуються, а при присвоєнні тип змінної визначається за результатом присвоєння.

Крім того, у мові JavaScript реалізується динамічний зв’язок коду з об’єктами: посилання на об’єкти перевіряються під час виконання програми.

2. Мова JS

2.1.Базовий синтаксис мови JS

Способи оголошення змінних.

Оголошення змінної робиться оператором var:
var a;

Змінні бувають трьох основних типів:

- ✓ number – будь-яке число (12; -123; 12.3; 1.23e2)
- ✓ boolean – приймає лише два значення: правда і брехня ("2 + 2 = 4" – true; "2 * 5 = -9" – false)
- ✓ string – текстовий рядок ("Будь-яка фраза")

JavaScript сам визначить тип змінної при присвоєнні їй значення.

Присвоїти змінній значення можна будь-яким з наступних способів:

```
<script>
var a = 6363;
var s;
s = "Це текстовий рядок";
YourName = prompt( "Ваше ім'я", "Вводити тут");
</script>
```

Тобто для оголошення змінної не завжди потрібно користуватися оператором var.

Іменем змінної може бути не довільна комбінація символів. Ім’я змінної має складатися з латинських букв, цифр, краще не використовувати спеціальні символи (крім символу підкреслювання _),

не можна використовувати пробіли, ім’я не повинно починатися з цифри і не повинно збігатися з зарезервованим словом JScript.

Регістр, яким написаний текст, в JavaScript має значення.

Запуск JavaScript.

Що необхідно зробити, щоб запускати скрипти, написані на мові JavaScript? Для цього потрібний браузер, здатний працювати з JavaScript. Нічого додаткового встановлювати не треба.

Створимо документ HTML з довільним ім’ям та розширенням html, наприклад, такий:

<!DOCTYPE html> <html> <head> <title>Моє перше знайомство з JS </title> </head> <body> <script> document.write('Hello, World!'); </script> </body> </html>	При відображенні сторінки в браузері на екрані з’явиться напис Hello, World!

Тут ми скористалися методом write вбудованого об’єкту document, який створюється браузером з HTML-файлу.

Для того, щоб створити інтерактивне випадаюче вікно з певним текстом, можна використати функцію alert():

<!DOCTYPE html> <html> <head> <title>Моє перше знайомство з JS </title> </head> <body> <script> alert('Hello, World!'); </script> </body> </html>	При відображенні сторінки в браузері з’явиться випадаюче вікно з написом Hello, World! і кнопка підтвердження.

Для того, щоб користувач міг ввести інформацію, можна використати функцію prompt():

<!DOCTYPE html> <html> <head> <title>Моє перше знайомство з JS </title> </head> <body> <script> name = prompt("Будь ласка, введіть Ваше ім'я"); document.write("Привіт," + name + "!") </script> </body> </html>	При відображенні сторінки в браузері з’явиться випадаюче вікно з питанням щодо імені, а коли користувач введе ім’я, на сторінці з’явиться привітання.

Тут при створенні рядка-привітання використовується символ «+» для конкатенації рядків.

Отже, для вбудовування скриптів в HTML-документ використовується контейнер SCRIPT. Але не всі браузери здатні розпізнавати і виконувати скрипти, тому іноді саме тіло скрипта поміщується в контейнер коментаря.

<!DOCTYPE html> <html> <head> <title>Моє перше знайомство з JS </title> </head> <body> <script> <!-- name = prompt("Будь ласка, введіть Ваше ім'я"); document.write("Привіт," + name + "!") --> </script> </body> </html>	При відображенні сторінки в браузері з’явиться випадаюче вікно з питанням щодо імені, а коли користувач введе ім’я, на сторінці з’явиться привітання.

2.2.Питання для самоконтролю

Які типи даних існують в JS?

Що означає нестрога типізація?

Як оголосити змінну?

Як оформлюються скрипти?

Як запустити скрипти у браузері?

2.3.Практичне завдання

Написати скрипт, що проводить опитування користувача про ім’я, вік, хобі тощо, а потім формує і виводить текст про користувача з використанням отриманих відповідей. Використовуйте методи prompt(), alert() і змінні.

3. Математичні оператори

3.1.Математичні оператори в JS

Математичні оператори

- ✓ додавання +
- ✓ віднімання -
- ✓ множення *
- ✓ ділення /
- ✓ остача від ділення %

Наведемо приклад:

```
<script>
a = prompt("Введіть сторону квадрата")
document.write("Площа Вашого квадрата дорівнює ", a*a)
</script>
```

Об’єкти, які більш детально будуть розглянуті далі, дозволяють отримати додатковий набір операторів.

Почнемо з об’єкта Math.

Цей об'єкт надає додатковий набір математичних операторів та констант. Ось найпоширеніші:

- ✓ LN2; LN10: LOG2E; LOG10E – різні логарифми
- ✓ E; PI – число e і число π відповідно
- ✓ SQRT2; SQRT1_2 – корінь квадратний з двох і 0,5 відповідно

Ось приклад:

```
<script>
a = prompt("Введіть сторону квадрата")
document.write("Діагональ Вашого квадрата дорівнює ", Math.SQRT2*a)
</script>
```

Можна використовувати наступні функції:

- ✓ sin (); cos (); tan (); asin (); acos (); atan () – синус, косинус, тангенс, арксинус, арккосинус, арктангенс
- ✓ abs () – модуль числа
- ✓ sqrt () – квадратний корінь з числа
- ✓ pow () – степінь числа (в дужках через кому подаються основа і показник степеня)
- ✓ exp () – експонента числа
- ✓ log () – логарифм числа
- ✓ random () – псевдовипадкове число
- ✓ round () – округлення числа
- ✓ ceil () – округлення вверх
- ✓ floor () – округлення вниз (ціла частина від ділення двох цілих чисел)

- ✓ max () – максимальне значення з тих чисел, які через кому подані в дужках
- ✓ min () – мінімальне значення з тих чисел, які через кому подані в дужках

3.2.Питання для самоконтролю

Які математичні оператори існують в JS?

Як записуються в скриптах математичні операції?

Як використовується об’єкт Math?

Які операції, константи та функції цього об’єкту ви знаєте?

3.3.Практичне завдання

1. Написати скрипт, що обчислює площу круга з радіусом, який вводить користувач.
2. Написати скрипт, що обчислює об’єм куба зі сферичною порожниною всередині нього.
3. Написати скрипт – генератор випадкових чисел від 1 до 100 з використанням вбудованої функції Math.random(), яка видає псевдовипадкове дробове число в діапазоні від 0 до 1.

4. Умовні оператори

4.1.Умовні оператори в JS

Тип Boolean

Як і в інших мовах, які ви вивчали, в JS існує тип змінної Boolean. Цей тип використовують умовні оператори: якщо деякий вираз або змінна приймає значення true, то виконується один набір операторів, а якщо цей вираз або змінна мають значення false, то виконується інший набір операторів або не виконується нічого.

Для отримання булевського результату виразів використовуються оператори порівняння:

- ✓ == Дорівнює (5 == 5)
- ✓ < Менше (-1 <1)
- ✓ > Більше (1> -2)
- ✓ <> Не дорівнює (1 <> 2)
- ✓ >= Більше або дорівнює (10 > = 5)
- ✓ <= Менше або дорівнює (5 <= 10)

і логічні оператори:

- ✓ ! = Логічне НЕ (1! = 2)
- ✓ && логічне ТА ((2 > 1) && (3 > 1))
- ✓ || логічне АБО ((2 > 1) || (0 > 1))
- ✓

Якщо потрібно перевірити кілька умов, то кожну умову слід взяти в дужки.

Для реалізації умовних сценаріїв використовується умовний оператор if. Наведемо простий приклад:

```
<script>
a = prompt("Введіть сторону квадрата")
if(a<5) alert('Площа Вашого квадрата менше '+25);
```

```
else alert('Площа Вашого квадрата не менше '+25);
</script>
```

Для реалізації розгалужень з багатьма гілками зручно використовувати оператор switch.

4.2.Питання для самоконтролю

Які оператори порівняння існують в JS?

Які логічні операції JS визнаєте?

Який тип результату дії оператора порівняння або застосування логічного оператора?

4.3.Практичне завдання

1. Розробити скрипт, який говорив би користувачеві «Ласкаво просимо» тільки в тому випадку, якщо він введе 1, інакше буде говорити «Вхід заборонений». Використовуйте умовний оператор і оператори порівняння.
2. Розробити скрипт, який перевірятиме, що користувач ввів непарне число, що не кратне 3 і не більше 100, і підтверджує виконання цих умов. Використовуйте умовні оператори, оператори порівняння, логічні оператори.

5. Об’єкти та робота з ними

5.1. Об’єкти в JS

Ідея об’єктів JavaScript дуже проста. Всі операції, які можна виконувати в програмі на мові JavaScript, описують дії над об’єктами, якими є елементи робочої області браузера і контейнери мови HTML. Саме в цьому полягає об’єктна орієнтованість мови JavaScript. Ніяких класів об’єктів, а тим більше успадкування в JavaScript немає. Є тільки об’єкти з набором властивостей і набір функцій над об’єктами, які називаються методами.

Крім методів існують і інші функції, більше схожі на функції з традиційних мов програмування, які дозволяють працювати зі стандартними математичними типами або керувати процесом виконання програми. Перш ніж звернутися до об’єктів, що становлять об’єктну модель браузера, визначимо поняття, які ми будемо використовувати.

Об’єкт – сукупність властивостей, методів, подій і колекцій, що надається браузером в рамках об’єктної моделі. Всі об’єкти створюються самим браузером, і доступ до них здійснюється через екземпляр.

Властивість – змінна в рамках об’єкта, яка може використовуватися для отримання якихось значень або установки нових. Багато властивостей можуть бути доступні тільки для читання.

Метод – процедура або функція, яку надає об’єкт для виконання будь-яких дій або управління властивостями об’єкта.

У більшості скриптів на мові JavaScript ми будемо користуватися функціями (або методами). Зазвичай функції є лише способом поєднати кілька команд.

Наведемо маленький приклад скрипту, що друкує якийсь текст.

```
<script>
function myFunction () {
document.write ( "Ласкаво просимо на мою сторінку! <br>");
document.write ( "Це JavaScript! <br>");
}
myFunction()
</script>
```

Всі команди скрипта, що знаходяться всередині фігурних дужок, тобто всередині {}, належать функції myFunction(). Це означає, що дві команди document.write() тепер з'єднані і можуть бути виконані за викликом функції myFunction().

Опис функції можна винести в секцію <head>:

```
<!DOCTYPE html>
<head>
<script language="JavaScript">
function myFunction () {
document.write ( "Ласкаво просимо на мою сторінку! <br>");
document.write ( "Це JavaScript! <br>");
}
</script>
</head>
<body>
<script>
myFunction()
</script>
</body>
```

В такому разі функція myFunction задається до того, як буде завантажена сторінка.

Більш того, можна винести весь скрипт в заголовок документа і таким чином розділити структуру сторінки та обробку подій на ній. Події ми розглянемо далі, а зараз наведемо приклад.

```
<!DOCTYPE html>
<html>
<head>
<script language = "JavaScript">
function myFunction ()
{
document.write ( "Ласкаво просимо на мою сторінку! <br>");
document.write ( "Це JavaScript! <br>");
}
</script>
</head>
<body>
<form>
<input type = "button" value = "Виклик функції" onClick = "myFunction
()">
</form>
</body>
</html>
```

Тут в тілі документа немає скриптів. Ми використовуємо об’єкти зі структури документа: form, button тощо. Можна перевірити, що в цьому прикладі при натисканні на кнопку здійснюється виклик функції myFunction().

Скрипт, що винесений у заголовок HTML-документа, часто називають сценарієм. Помістивши сценарій на JavaScript у розділі <head> документа, ми робимо так, що весь сценарій буде завантажений до того, як потрібно буде його виконати. Загальний вигляд HTML-документа зі

сценарієм у заголовку такий:

<!DOCTYPE html> <html> <head> <script language="JavaScript"> Код сценарію </script> </head> <body> Код HTML </body> </html>	Приклад вбудовування скрипта в HTML-документ.

Таким чином, ми розділяємо структуру сторінки та інтерактивні дії і обробку подій на цій сторінці.

5.2.Питання для самоконтролю

Що таке об’єкт JS?

Що таке властивість об’єкта?

Що таке метод об’єкта?

Як об’єкти, властивості та функції описуються в скриптах?

5.3.Практичне завдання

Написати скрипт, який після натискання відповідної кнопки (одної з трьох – з написами «червоний», «жовтий», «зелений») описує дії пішохода при відповідному сигналі світлофора.

6.Події

6.1.1. Події та їх обробники

У JavaScript є події – аналог програмних переривань. Ці події також орієнтовані на роботу в WorldWideWeb, наприклад, завантаження сторінки в робочу область Navigator або вибір гіпертекстового посилання.

Використовуючи події, автор гіпертекстової сторінки і програми, що її відображає, може організувати перегляд динамічних об'єктів, наприклад, рядка, що біжить, або управління багатовіконним інтерфейсом.

Події та обробники подій є дуже важливою частиною JavaScript. Події, головним чином, ініціюються тими чи іншими діями користувача: користувач водить курсором, натискає на кнопки, вводить якусь інформацію в поля вводу тощо. Якщо він виконує якусь дію, виникає подія. Наприклад, при натисканні на кнопку виникає подія onClick. Якщо покажчик миші перетинає посилання гіпертексту – виникає подія MouseOver.

Існує кілька різних типів подібних подій. Ми можемо змусити нашу програму реагувати на деякі з них. Все це може бути реалізовано за допомогою спеціальних програм обробки подій, приклади яких наведено далі.

Так, нехай ми хочемо, щоб в результаті натискання кнопки миші випадало вікно. Це означає, що створення вікна повинно бути реакцією на подію – на «Click». Програма-оброблювач подій, яку ми повинні використовувати в даному випадку, називається onClick. Вона описує, що потрібно робити, якщо станеться ця подія.

Наведений нижче код представляє простий приклад програми обробки події onClick:

```
<form>
```

```
<input type="button" value="Натисні" onClick = "alert('YES')">
</form>
```

Тут виводиться кнопка, при натисканні на яку виводиться повідомлення «YES». Зауважимо, що в даному випадку ми не використовуємо тег <script>. Ми створюємо форму з кнопкою; тег <input> містить onClick = "alert ('YES')". Цей атрибут визначає, що відбувається, коли натискають на кнопку. Тобто якщо має місце подія onClick, виконується виклик alert ('YES').

Аргумент функції alert() – це рядок. В нашому випадку це 'YES' – текст, що з'явиться у випадаючому вікні.

Отже, коли користувач натискає на кнопку, наш скрипт створює вікно, що містить текст 'YES'.

У цьому прикладі ми написали onClick = "alert ('YES')" – тобто ми використовували і подвійні, і одинарні лапки. Це пов'язано з тим, що запис onClick = "alert (" YES ")" не дозволяє однозначно визначити, до якої з частин конструкції відноситься функція обробки подій onClick. У таких випадках використовуються два типи лапок, але не має значення, в якому порядку.

Для того, щоб можна було звернутися до якогось окремого об'єкта, при його описі використовується атрибут ID або name

```
<p ID=passage>
```

Тепер до параграфу можна звернутися, використовуючи passage.

Наведемо приклад, який використовує розглянуті можливості.

Нехай ми хочемо, щоб користувач ввів деякий текст, а потім (при натисканні кнопки) цей текст з'явився на екрані. Це можна зробити так.

```
<!DOCTYPE html>
 <html>
<body>
<form>
<input type="button" value="Введіть текст" onClick="text=prompt()">
<br><br>
<input type="button" value="Надрукувати введений текст"
onclick='document.getElementById("passage").innerHTML=text'>
</form>
<br>
<p id="passage"></p>
</body>
</html>
```

Тут знов не використовується тег <script>, ми використовуємо лише об’єкти зі структури документа.

Детальніше, створюються дві кнопки – одна за написом «Введіть текст», а другі – з написом «Надрукувати введений текст», а також параграф, який спочатку порожній.

При натисканні на першу кнопку з’являється випадаюче вікно з рядком вводу, в який можна ввести текст. Точніше, викликається функція alert(), а результат, який вона повертає, присвоюється змінній text. При натисканні на другу кнопку змінюється властивість об’єкта (параграфа) з іменем passage, а саме, його вміст: в параграфі з’являється текст, що міститься у змінній text.

Ще один приклад:

```
<!DOCTYPE html>
 <html>
<body>
<form>
```

```
<label for="text"> Введіть текст:</label>
      <input type="text" id="text">
<br><br>
<input type="button" value="Надрукувати введений текст"
onclick='document.getElementById("passage").innerHTML=document.getEle
mentById("text").value'>
</form><br>
<p id="passage"></p>
</body>
</html>
```

Тут текст вводиться у вікно, парад яким з'являється напис «Введіть текст». Сам текст є полем елемента (рядка вводу), у якого id дорівнює text. При натисканні на кнопку з написом «Надрукувати введений текст» цей вміст, як і в попередньому прикладі, передається елементу з іменем passage.

6.2.Питання для самоконтролю

Що таке подія в JS?
Що таке обробник події в JS?
Що таке сценарій в JS?
Як в коді описуються сценарії?

6.3.Практичне завдання

Створити таку форму відправки листів.

1. Складові форми – поля для введення: адресат, тема, текст.
2. Для відправки використовується кнопка «відправити».
3. При натисканні на кнопку має виводитись напис «Лист відправлено».

4. Зберегти документ і перевірити, як він працює.
5. Додати кнопку, яка б очищувала текстові поля.
6. Використовуючи стилі html, оформіть красиво створену форму.

Список літератури

1. Інтернет технології. Методичні вказівки до лабораторних робіт для студентів напрямку підготовки 6.050802 «Електронні пристрої та системи». – К.: НТУУ “КПІ”, 2014. – 90 с.
2. Методичні вказівки і завдання до лабораторних робіт з курсу інформатики. Частина 2. Основи web-розробки: Навч.-метод. посіб. / Автори-укладачі: О.В. Присяжнюк, О.В. Рєзіна – Кропивницький: ЦДПУ імені В. Винниченка, 2021. – 42 с.
3. Пасічник О.Г., Пасічник О.В., Стеценко І.В. Основи веб-дизайну дизайну : навч. посіб. – К.: Вид. група BHV, 2009. – 336 с.
4. Розробка ігрових web додатків: електронний навчальний посібник для підготовки студентів на першому (бакалаврському) рівні вищої освіти, галузі знань 12 «Інформаційні технології», спеціальності 121 «Інженерія програмного забезпечення» / Укладач: М.В. Жарікова. – Херсон: видавництво ФОП Вишемирський В.С., 2018. – 218 с.

Інтернет-ресурси

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide

https://w3schoolsua.github.io/js/index.html#gsc.tab=0

Електронне навчальне видання комбінованого використання
Можна використовувати в локальному та мережному режимі

Сузікова Олена Геннадіївна

ВЕБ ТЕХНОЛОГІЇ В ІНФОРМАЦІЙНОМУ ОБМІНІ

Методичні рекомендації
до практичних занять для здобувачів вищої освіти першого (бакалаврського) рівня з дисципліни «Обробка, зберігання та передача даних в сучасних інформаційних технологіях» за спеціальністю «Прикладна математика»

У 3 частинах

Частина 3: JavaScript

В авторській редакції

Підписано до розміщення 25.06.2025. Гарнітура Times New Roman.
Ум. друк. арк. 1,59. Обсяг 0,894. Зам. № 248/25.

Харківський національний університет імені В. Н. Каразіна,
61022, м. Харків, майдан Свободи, 4.
Свідоцтво суб'єкта видавничої справи ДК № 3367 від 13.01.2009.
Видавництво ХНУ імені В. Н. Каразіна