Урок **.** **Кнопки, події, функції**

**Мета:**

* ***формування предметних компетентностей***: забезпечити засвоєння знань про об’єктну модель документа; формувати вміння описувати об’єктну модель документа;
* ***формування ключових компетентностей:***
	+ *спілкування рідною мовою*: уміння обговорювати результати дослідження і робити висновки;
	+ *спілкування іноземними мовами*: уміння оперувати базовою міжнародною ІТ-термінологією;
	+ *уміння вчитися*: навички самостійно обирати завдання навчальної діяльності;
	+ *ініціативність і підприємливість*: уміння узагальнювати інформацію іприймати рішення.

**Очікувані результати:** учні вміють аналізувати і описувати об’єктну модель документа.

**Обладнання:** комп’ютери з доступом в мережу Інтернет, інтерактивна дошка.

**Тип уроку**: комбінований.

**Хід уроку**

**І. Організаційний етап**

* привітання
* перевірка присутніх
* перевірка готовності учнів до уроку

Створення атмосфери психологічного комфорту.

**ІІ. Актуалізація опорних знань і життєвого досвіду**

***Прийом «Моє уявлення про поняття»***

Учитель пропонує учням розповісти, що вони уявляють під словами «Об’єктна модель документа».

* Ви можете пояснити, що це таке?
* Для чого воно потрібно?
* Де застосовується?
* Які асоціації викликає?

**ІІІ. Повідомлення теми. Організація спільного з учнями формулювання мети й завдань уроку**

Учитель пропонує учням уважно прочитати тему уроку та визначити, що від них очікується, яких знань їм слід набути, що навчитися робити.

**Завдання уроку:** зрозуміти суть поняття «об’єктна модель документа»; навчитися описувати об’єктну модель документа.

Важливою ознакою інтерактивних HTML-сторінок є можливість реакції на дії користувача. Наприклад, натиск на кнопці повинен викликати появу діалогового вікна, або виконання перевірки правильності введених користувачем даних.

В **JavaScript** інтерактивність реалізована за допомогою перехвату та обробки подій, викликаних в результаті дій користувача. Для цього в теги деяких елементів введені параметри обробки подій. Ім'я параметру обробки події починається з префіксу **on**, за яким йде назва події.

Наприклад, події клік кнопкою миші **Click**, відповідає параметр обробки події з назвою **onClick**.

Назви та характеристики деяких подій наведені в таблиці.



**IV. Мотивація навчальної діяльності**

***Мотиваційні запитання***

* Чим для мене цікава тема уроку?
* Як мені налаштуватися на роботу на уроці?
* Яку особисту мету я планую на цьому уроці?

**V. Вивчення нового матеріалу**

**Пояснення вчителя з елементами демонстрування презентації**

*(використовуються можливості локальної мережі кабінету або проектор)*

**VI. Закріплення вивченого матеріалу**

**VIІ. Практична робота «Розв’язування задач»**

**Задача**. Необхідно, щоб при наведені курсору на комірку таблиці із написом "Привіт" з’являлось вікно повідомлення з фразою "Hello".

Варіант 1:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"

 "http://www.w3.org/TR/html4/strict.dtd">

<HTML>

<HEAD>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8">

<TITLE>JavaScript</TITLE>

</HEAD>

<BODY>

<table border="1">

<td onClick="alert('Hello')"> Привіт </td>

</table>

</BODY>

</HTML>

Варіант 2:

В варіанті 1, код JavaScript був записаний безпосередньо в тезі, а в варіанті 2 наслідком кліку став виклик функції. Варіант 2 слід використовувати, якщо код обробки події великий за обсягом.

<script>

function Go() {

alert("Hellllllllo")

}

</script>

<td onClick="Go()"> Привіiiiiiiiт </td>

</table>

## Кнопки

Для відправки введених даних на формі використовуються кнопки.

Для створення кнопки використовується або елемент **button**:

<button name="send">Відправити</button>

або елемент **input**:

<input type="submit" name="send" value="Відправити" />

З точки зору функціональності в **html** ці елементи не зовсім рівноцінні, але в даному випадку вони нас цікавлять з точки зору взаємодії з кодом **javascript**.

При натисканні на будь-який з цих двох варіантів кнопки відбувається відправка форми за адресою, яка вказана у форми в атрибуті **action**, або за адресою веб-сторінки, якщо атрибут **action** не вказано. Однак в коді **javascript** ми можемо перехопити відправку, обробляючи подію**click**

<!DOCTYPE html>

При натисканні на кнопку відбувається подія **click**, і для його обробки до кнопки прикріплюємо обробник **sendForm**. У цьому обробнику перевіряємо введений в текстове поле текст. Якщо його довжина більше 5 символів, то виводимо повідомлення про неприпустиму довжину і перериваємо звичайний хід події за допомогою виклику **e.preventDefault ()**. В результаті форма не відправляється.

Якщо ж довжина тексту менше шести символів, то також виводиться повідомлення, і потім форма відправляється.



<!DOCTYPE html>

<html>

<head>

 <meta charset="utf-8" />

</head>

<body>

<form name="search">

 <input type="text" name="key"></input>

 <input type="submit" name="send" value="Відправити" />

</form>

<script>

function sendForm(e){

 // отримуємо значення поля key

 var keyBox = document.search.key;

 var val = keyBox.value;

 if(val.length>5){

 alert("Неприпустима довжина рядка");

 e.preventDefault();

 }

 else

 alert("Відправка дозволена");

}

var sendButton = document.search.send;

sendButton.addEventListener("click", sendForm);

</script>

</body>

</html>

Також ми можемо при відправці змінити адресу, на який відправляються дані:

function sendForm(e){

 // отримуємо значення поля key

 var keyBox = document.search.key;

 var val = keyBox.value;

 if(val.length>5){

 alert("Неприпустима довжина рядка");

 document.search.action="PostForm";

 }

 else

 alert("Відправка дозволена");

}

В даному випадку, якщо довжина тексту більше п'яти символів, то текст відправляється, тільки тепер він відправляється за адресою **PostForm**, оскільки задана властивість **action**:

document.search.action="PostForm";

Для очищення форми призначені наступні рівноцінні по функціональності кнопки:

<button type="reset">Очистити</button>

<input type="reset" value="Очистити" />

При натисканні на кнопки відбудеться очищення форм. Але також функціональність по очищенню полів форми можна реалізувати за допомогою методу **reset()**:

function sendForm(e){

 // отримуємо значення поля key

 var keyBox = document.search.key;

 var val = keyBox.value;

 if(val.length>5){

 alert("Неприпустима довжина рядка");

document.search.reset();

 e.preventDefault(); }

 else

 alert("Відправка дозволена");

}

Крім спеціальних кнопок відправки і очищення на формі також може використовуватися звичайна кнопка:

<input type="button" name="send" value="Отправить" />

При натисканні на подібну кнопку відправки даних не відбувається, хоча також генерується подія **click**:

<!DOCTYPE html>

<html>

<head>

 <meta charset="utf-8" />

</head>

<body>

<form name="search">

 <input type="text" name="key" placeholder="Введіть ключ"></input>

 <input type="button" name="print" value="Друк" />

</form>

<div id="printBlock"></div>

<script>

function printForm(e){

 // отримуємо значення поля key

 var keyBox = document.search.key;

 var val = keyBox.value;

 // отримуємо елемент printBlock

 var printBlock = document.getElementById("printBlock");

 // створюємо новий параграф

 var pElement = document.createElement("p");

 // вставляємо у него текст

 pElement.textContent = val;

 // додаємо параграф в printBlock

 printBlock.appendChild(pElement);

}

var printButton = document.search.print;

printButton.addEventListener("click", printForm);

</script>

</body>

</html>

При натисканні на кнопку отримуємо введений в текстове поле текст, створюємо новий елемент параграфа для цього тексту і додаємо параграф в елемент printBlock.



<html>

<head>

<meta charset="utf-8" />

</head>

<body>

<form name="search">

<input type="text" name="key"></input>

<input type="submit" name="send" value="Відправити" />

</form>

<script>

function sendForm(e){

// отримуємо значення поля key

var keyBox = document.search.key;

var val = keyBox.value;

if(val.length>5){

alert("Неприпустима довжина рядка");

e.preventDefault();

}

else

alert("Відправка дозволена");

}

var sendButton = document.search.send;

sendButton.addEventListener("click", sendForm);

</script>

</body>

</html>

## Елементи форми

Форма може містити **поля для введення текстової інформації**, списки для вибору заздалегідь визначених відповідей, прапорці, перемикачі, кнопки та інші елементи керування. Розглянемо докладніше деякі з них.

### Текстові поля

Для введення текстових даних у спеціальні поля використовують тег <INPUT>, в якого атрибут TYPE має значення t e x t :

<INPUT TYPE="text">

Інші атрибути тегу <INPUT>:

• NAME — ім'я змінної, в якій зберігається введене значення;

• VALUE — початкове значення;

• SIZE — довжина текстового поля;

• MAXLENGTH — максимальна кількість символів для введення.

Цей самий тег, в якого атрибут TYPE має значення password використовують у разі потреби ввести пароль. При цьому надруковані символи будуть приховані за зірочками чи крапками.

<INPUT TYPE="password">

Для текстів великого обсягу зі смугами прокручування використовують парний тег <TEXTAREA> з такими атрибутами:

• ROWS — висота текстового прямокутника в символах;

• COLS — ширина текстового прямокутника в символах;

• WRAP — визначення способу переходу на інший рядок (off — за допомогою клавіші Enter, virtual — автоматично);

• NAME — ім'я для доступу до текстової області з програми-сценарію.

### Перемикачі

Щоб дізнатися про думку відвідувача з того чи іншого приводу, не змушуючи його вводити інформацію, використовують перемикачі з варіантами відповідей. Відвідувач переглядає їх і вибирає потрібний. Перемикачі дають змогу вибрати лише один із запропонованих варіантів.

Для створення перемикачів використовують тег <INPUT>, в якого атрибут TYPE (тип) має значення radio. Групі перемикачів, що стосуються одного питання, обов'язково присвоюють однакове ім'я (NAME). Атрибут VALUE позначає відповідний перемикачу варіант відповіді для розробника форми, тоді як напис після тегу <INPUT> — для користувача. Для вибору одного з перемикачів за умовчанням використовують атрибут CHECKED.

<Н4>Вибери мову:</Н4>

<INPUT TYPE="radio" NAME="lang" VALUE="українська" CHECKED>Українська <BR><INPUT TYPE="radio" NAME="lang" VALUE="російська">Російська <BR>

<INPUT TYPE="radio" NAME="lang" VALUE="англійська">Англійська <BR>

### Прапорці

Прапорці дають змогу вибирати кілька варіантів із запропонованих.

Для них атрибут TYPE тегу <INPUT> має значення checkbox.

Вибери екзамен:

<INPUT TYPE="checkbox" NAME="exam" VALUE="українська"> Українська <BR>

<INPUT TYPE="checkbox" NAME="exam" VALUE="фізика"> Фізика <BR>

<INPUT TYPE="checkbox" NAME="exam" VALUE="математика">

Математика <BR>

### Списки

Списки, які можна розміщувати у формі так само, як перемикачі та прапорці, також позбавляють відвідувачів веб-сторінки віднеобхідності вводити інформацію вручну, даючи змогу вибрати відповідь із запропонованих варіантів. Список розміщують міжтегами <SELECT> та </SELECT>, а його елементи визначають за допомогою тегу <OPTION>. **Наприклад:**

Район:

<SELECT>

<OPTION NAME="frank">Франківський

<OPTION NAME="syh">Сихівський

<OPTION NAME="gal">Галицький

</SELECT>

Якщо замість <SELECT> записати <SELECT MULTIPLE>, користувач отримає можливість вибрати зі списку кілька варіантів. Атрибут SIZE=N тегу <SELECT> обмежує КІЛЬКІСТЬ показаних елементів списку числом N, після чого буде використано прокручування.

### Кнопки

Заповнивши форму текстом та вибравши потрібні елементи керування, користувач повинен мати можливість підтвердити свої дії або скасувати помилково введені дані. Для підтвердження правильності введення використовують кнопку, яку створює тег <INPUT> з атрибутом TYPE="submit". На цій кнопці буде напис, заданий атрибутом VALUE. Аналогічно створюють кнопку скасування дії: атрибут TYPE у цьому випадку повинен мати значення reset.

Звичайна кнопка, з якою можна зв'язати будь-яку дію, має атрибут TYPE="button".

<INPUT TYPE="submit" VALUE="Biдicлaти">

<INPUT TYPE="reset" VALUE="Оновити форму">

<INPUT TYPE="button" VALUE="Обчислити">

Для оригінального оформлення кнопки в неї можна вставити малюнок.

Для цього використовують такий синтаксис:

<INPUT TYPE="image" SRC="..."

WIDTH=... HEIGHT=... ALT="..." VALUE=...>

### Надсилання форми

Отже, розробник веб-сторінки має змогу отримати відповіді користувача на поставлені запитання, не обмежуючись діалоговими вікнами методів Confirm та Prompt. Форми можна проектувати відповідно до своїх потреб, додавати до них зображення та інші елементи. Після заповнення форми користувач надсилає дані на подальше опрацювання.

Таблиці каскадних стилів і динамічні веб-сайти Включені в документ HTML елементи на зразок полів введення даних, перемикачів, прапорців та кнопок вміщують у тег форми <FORM>. . .</FORM>. Цей тег повинен мати певні атрибути, наприклад:

<FORM METHOD="post" ACTION="/bin/serv"

ENCTYPE="text/plain">

Атрибут METHOD може мати значення post або get, які визначають різні методи передавання інформації з форми на URL-адресу сценарію-обробника, що зазначена в атрибуті ACTION. У свого провайдера потрібно уточнити, який із методів слід використовувати.

Необхідно також знати місце розташування доступних сценаріїв та їхні імена.

Дані форми можна пересилати на певну адресу електронної пошти, якщо задати атрибут ACTION=mailto:aflpeca\_nomTM.

Атрибут ENCTYPE визначає, в який спосіб дані форми потрібно кодувати перед надсиланням на сервер. За умовчанням використовується кодування, під час якого символи замінюються комбінацією символу ≪% ≫ та шістнадцяткового коду символу тексту в ASCII- таблиці. Щоб дані форми не перетворювати на шістнадцяткові числа, слід задавати значення text/plain .\_\_

**VIІI. Підбиття підсумків уроку. Рефлексія**

***Запитання для рефлексії***

* Що на уроці в мене виходило найкраще?
* Наскільки активно я працював на уроці?
* Чи маю я радість від того, що чомусь навчився?

**VIІІ. Домашнє завдання**

Дослідити структуру DOM улюблених сайтів

**ІX. Оцінювання роботи учнів**