

ПРАКТИЧНА РОБОТА №10
З навчальної дисципліни «Веб-технології»
на тему: «Взаємодія з користувачем. Обробка подій і форми»

Навчальний час: 2 год

Мета:

1. Ознайомитись з подіями в JavaScript та навчитись їх застосовувати..
2. Ознайомитись із інструментами для створення HTML-форм.
3. Навчитись створювати форми.
4. Навчитись компоувати форми за допомогою HTML-таблиць та списків

Хід роботи:

1. Ознайомитись із короткими теоретичними відомостями;
2. Виконати тренувальні завдання із порядку виконання роботи для здобуття навичок роботи із JS.
3. Виконати практичні завдання, роблячи скрін-шоти виконання скриптів у браузері та поміщаючи їх у звіт.
4. Оформити звіт, помістивши туди фрагменти коду для кожного сценарію та результат виконання (скрін-шот)

Короткі теоретичні відомості

1. Загальні поняття

Форма — це інструмент, за допомогою якого HTML-документ може відправити інформацію в заздалегідь певну точку зовнішнього миру. Форми використовуються для опитування відвідувачів, покупки чого-небудь, відправлення електронної пошти.

Принцип роботи форм наступний: відвідувач, що зайшов до вас на сторінку заповнює форму, а після натискання певної кнопки форма бере дані із заповнених полів і відправляє їх у призначене місце.

1.1. Атрибути форми

Форми розміщуються між тегамі <FORM></FORM>. HTML-документ може містити в собі кілька форм, але вони не повинні перебувати одна усередині іншої. Тег <FORM> може містити наступні атрибути (див. табл.1):

Таблиця 1 – Основні атрибути тегу <FORM>

Атрибут	Призначення атрибуту
ACTION	Обов'язковий атрибут. Визначає, де перебуває оброблювач форми.
METHOD	Визначає, яким чином дані з форми будуть передані оброблювачеві. Припустимі значення: METHOD=POST й METHOD=GET. Якщо значення атрибута не встановлено, за замовчуванням передбачається METHOD=GET.
ENCTYPE	Визначає, яким чином дані з форми будуть закодовані для передачі оброблювачеві.

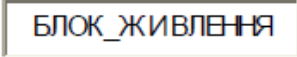
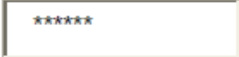
Більше інформації про форми а також приклади коду із результатами виконання можна знайти тут:

- <http://htmlbook.ru/html/form>
- <https://css.in.ua/html/tag/form>

1.2. Елементи форми

Для внесення інформації користувачем у форму використовується елемент `<INPUT>` Це і є поля, у які користувач уводить інформацію. Кожен елемент `<INPUT>` включає атрибут `NAME=ім'я`, що визначає ім'я даного поля (ідентифікатор поля). У таблиці 2 представлені основні типи, застосовуваних елементів `<INPUT>`:

Таблиця 2 – Основні типи елементів тегу `<INPUT>`:

Елемент	Призначення елементу
TYPE=text	<p>Визначає вікно для введення рядка тексту. Може містити додаткові атрибути <code>SIZE=число</code> (ширина вікна введення в символах) і <code>MAXLENGTH=число</code> (максимально припустима довжина вводиться строки, що, у символах):</p> <pre><INPUT TYPE=text SIZE=20 NAME=User VALUE="БЛОК_ЖИВЛЕННЯ"></pre>  <p>Визначає вікно шириною 20 символів для введення тексту. За замовчуванням у вікні перебуває текст БЛОК_ЖИВЛЕННЯ, що користувач може змінити.</p>
TYPE=password	<p>Визначає вікно для введення пароля. Абсолютно аналогічний типу text, тільки замість символів тексту, що вводиться, показує на екрані зірочки (*):</p> <pre><INPUT TYPE=password NAME=PW SIZE=20 MAXLENGTH=10></pre>  <p>Визначає вікно шириною 20 символів для введення пароля. Максимально припустима довжина пароля - 10 символів.</p>

TYPE=radio	<p>Визначає радіокнопку. Може містити додатковий атрибут CHECKED (показує, що кнопка відзначена). У групі радіокнопок з однаковими іменами може бути тільки одна позначена радіокнопка:</p> <pre><INPUT TYPE=radio NAME=Question VALUE="Yes" CHECKED> Так <INPUT TYPE=radio NAME=Question VALUE="No"> Немає <INPUT TYPE=radio NAME=Question VALUE="Possible"> Можливо</pre> <p> <input checked="" type="radio"/> Так <input type="radio"/> Немає <input type="radio"/> Можливо </p> <p>Визначає групу із трьох радіокнопок, підписаних Yes, No й Possible. Спочатку позначена перша із кнопок. Якщо користувач не відзначить іншу кнопку, оброблювачеві буде передана змінна Question зі значенням Yes. Якщо користувач відзначить іншу кнопку, оброблювачеві буде передана змінна Question зі значенням No або Possible.</p>
TYPE=checkbox	<p>Визначає квадрат, у якому можна зробити позначку. Може містити додатковий атрибут CHECKED (показує, що квадрат позначений). На відміну від радіокнопок, у групі квадратів з однаковими іменами може бути кілька позначених квадратів:</p> <pre><INPUT TYPE=checkbox NAME=Comp VALUE="CPU"> Процесори <INPUT TYPE=checkbox NAME=Comp VALUE="Video" CHECKED> Відеоадаптери <INPUT TYPE=checkbox NAME=Comp VALUE="Scan"> Сканери <INPUT TYPE=checkbox NAME=Comp VALUE="Modem" CHECKED> Модеми</pre> <p> <input type="checkbox"/> Процесори <input checked="" type="checkbox"/> Відеоадаптери <input type="checkbox"/> Сканери <input checked="" type="checkbox"/> Модеми </p> <p>Визначає групу із чотирьох квадратів. Спочатку позначені другий і четвертий квадрати. Якщо користувач не зробить змін, оброблювачеві будуть передані дві змінні: Comp=Video й Comp=Modem.</p>

TYPE=hidden	<p>Визначає схований елемент даних, що не видний користувачеві при заповненні форми й передається оброблювачеві без змін. Такий елемент іноді корисно мати у формі, що час від часу піддається переробці, щоб оброблювач міг знати, з якою версією форми він має справу.</p> <p><INPUT TYPE=hidden NAME=version VALUE="1.1"></p> <p>Визначає сховану змінну version, що передається оброблювачеві зі значенням 1.1.</p>
TYPE=submit	<p>Визначає кнопку, при натисканні на яку запускається процес передачі даних з форми оброблювачеві:</p> <p><INPUT TYPE=submit VALUE="Відправити"></p>
	<div>Відправити</div>
TYPE=reset	<p>Визначає кнопку, при натисканні на яку очищуються поля форми. Оскільки при використанні цієї кнопки дані оброблювачеві не передаються, кнопка типу reset може й не мати атрибута name:</p> <p><INPUT TYPE=reset VALUE=" Скидання "></p> <div>Скидання</div>

1.2.1 Елемент форми <TEXTAREA>:

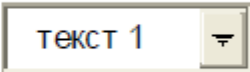
Форми можуть містити поля для введення великого тексту <TEXTAREA>, наприклад:

<TEXTAREA NAME=address ROWS=5 COLS=50> Наберіть тут нове повідомлення </TEXTAREA>

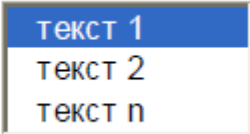
Атрибут NAME визначає ім'я, під яким уміст вікна буде передано оброблювачеві. Атрибут ROWS установлює висоту вікна в рядках. Атрибут COLS установлює ширину вікна в символах. Текст, розміщений між тегамі <TEXTAREA></TEXTAREA>, являє собою вміст вікна за замовчуванням. Користувач може його відредагувати або просто стерти.

1.2.2 Елемент форми <SELECT>

Форми можуть містити меню вибору, що починається відкриваючим тегом <SELECT> (містить обов'язковий атрибут NAME, який визначає ім'я меню) і завершується закриваючим </SELECT>. Між ними перебувають теги <OPTION>, що визначають елемент меню. Обов'язковий атрибут VALUE установлює значення, що буде передано оброблювачеві, якщо обрано цей елемент меню. Тег <OPTION> може включати атрибут selected, який показує, що даний елемент обраний/відзначений за замовчуванням.

<pre><SELECT NAME="ім'я"> <OPTION VALUE="option_1" selected>текст 1 <OPTION VALUE="option_2">текст 2 <OPTION VALUE="option_n">текст n </SELECT></pre>	
---	---

Тег <SELECT> може також містити атрибут MULTIPLE, присутність якого показує, що з меню можна вибрати кілька елементів. Більшість Оглядачів показують меню <SELECT MULTIPLE> у вигляді вікна, у якому перебувають елементи меню. Висоту вікна в рядках можна задати атрибутом SIZE=число.

<pre><SELECT MULTIPLE SIZE=3 NAME="ім'я"> <OPTION VALUE="option_1" selected>текст 1 <OPTION VALUE="option_2">текст 2 <OPTION VALUE="option_n">текст n </SELECT></pre>	
---	---

2. Обробка подій у JavaScript

Оброблювачі подій являють собою невеликі підпрограми, що пов'язують дії користувачів зі сценаріями, які необхідно виконати у відповідь на ці дії. До таких дій відносяться: клацання мишею, натискання клавіш клавіатури, вибір або зміна елементів форми, а також завантаження і вивантаження Web-сторінки. Оброблювачі подій JavaScript можна помістити у дескриптори HTML таким же чином, як інші атрибути. Наприклад, оброблювач події onClick можна увести як атрибут дескриптора кнопки <button> та ототожнити з будь-якою функцією:

```
<button onClick="Hello();">
```

Даний дескриптор не тільки відображає на Web-сторінці кнопку. Після клацання по цій кнопці виконується функція Hello().

Нижче (табл. 3) наведено найпоширеніші оброблювачі подій.

Таблиця 3 – Опис найпоширеніших подій

Подія	Опис
onmouseover	Подія сигналізує про те, що покажчик миші розташований на деякому елементі
onmouseout	Подія відбувається тоді, коли покажчик миші переміщується з елемента
onclick	Оброблювач події зазвичай приєднується до кнопок форми, наприклад таким, як submit і reset,
ondblclick	Подвійне клацання мишею можна зафіксувати за допомогою події. Оброблювач цієї події можна ввести в ті ж елементи, що і й оброблювач події onclick, включаючи дескриптори
onfocus	Оброблювач події виконує програму JavaScript, коли поле отримує фокус.
onblur	Оброблювач події виконує програму JavaScript, коли відбувається втрата фокусу.
onload	Подія load відбувається, коли браузер завершує завантаження вікна або фреймів. Оброблювач події виконує програму JavaScript, коли Web-сторінка або фрейми повністю завантажаться.
onchange	Оброблювач події фіксує момент зміни елемента форми. Подію change можна використовувати в дескрипторах <select>, <input> та <textarea>.

Приклад використання оброблювачів подій onclick та ondblclick.

Нижче показано код (лістинг 1), який створює кнопку у вікні браузера. При одинарному клацанні по цій кнопці колір фону змінюється на червоний, а при подвійному клацанні по ній колір фону змінюється на синій.

Лістинг 1 – Використання оброблювачів подій onclick та ondblclick

```
<html>
<head>
<script language=javascript>
function it_1() {
document.bgColor="red";
}
function it_2() {
document.bgColor="blue";
}
</script>
<body>
<form>
<input type="button" value="Зміна кольору"
onclick="it_1()" ondblclick="it_2()">
</form>
</body>
</html>
```

Приклад використання оброблювачів подій onfocus та onblur.

Нижче наведено код (лістинг 2), результатом якого є створення у вікні браузера двох текстових полів (рис.1)

Лістинг 2 – Використання оброблювачів подій onfocus та onblur

```
<html>
<head>
</head>
<body>
<script language="javascript">
</script>
<form name='form1'>
<input type="text" name="text1" value="Немає фокусу"
onfocus="document.form1.text1.value='фокус є'"
onblur="document.form1.text1.value='Загубив фокус'">
<input type="text" name="text2" value="Немає фокусу"
onfocus="document.form1.text2.value='фокус є'"
onblur="document.form1.text2.value='Загубив фокус'">
</form>
</body>
</html>
```

При клацанні в перше текстове поле воно отримує фокус, що призводить до виконання оброблювача події onfocus. В результаті чого текст першого поля змінюється. При клацанні в друге текстове поле виконуються два оброблювача

подій: onfocus та onblur. В результаті чого текст першого і другого поля змінюється:

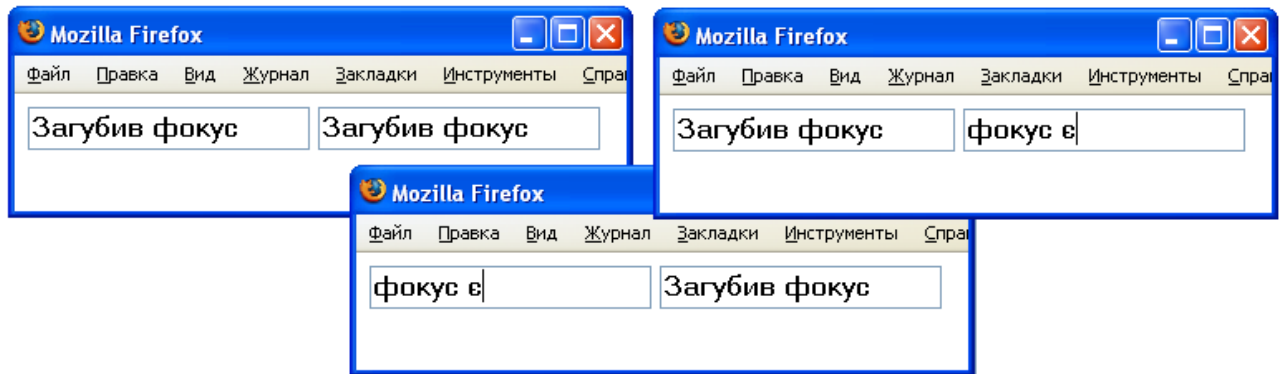


Рис. 1 – Використання оброблювачів подій onfocus та onblur

Приклад використання оброблювача подій onchange.

Оброблювач події onchange, що застосовується з текстовими полями та елементами <textarea>, фіксує зміну тексту в полі (лістинг 3). За допомогою цієї події можна визначити, чи оновлювався текст. Можна також перевірити, чи заповнені відповідні поля перед відправкою форми серверу.

Лістинг 3 – Використання оброблювача подій onchange

```
<html>
<head>
</head>
<body>
<script language="javascript">
function chan_1() {
if (document.form1.text1.value < 0 ||
document.form1.text1.value > 10) {
window.alert("Це значення не допустиме! Введіть ціле число в межах від 0 до
10");
document.form1.text1.value=0;}
}
</script>
<form name='form1'>
<input type="text" name="text1" value="0"
onchange="chan_1()">
</form>
</body>
</html>
```

В результаті виконання коду у вікні браузера відобразиться текстове поле із значенням «0»(рис.2).

При зміні текстового поля відбувається виклик функції. Дана функція перевіряє, чи не менше значення текстового поля, ніж 0, і чи не більше, ніж 10. Якщо це так, то на екран виводиться попереджувальне діалогове вікно з повідомленням:

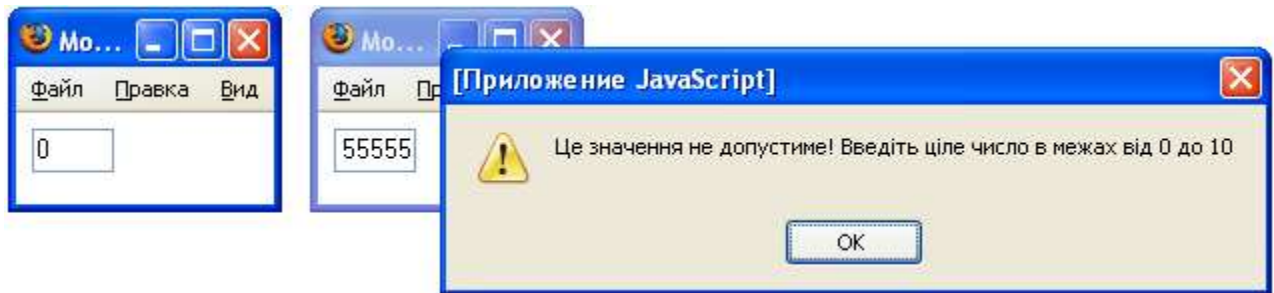


Рис. 2 – Використання оброблювача подій onchange

Порядок виконання роботи:

Тренувальні завдання

1. Створіть HTML – сторінку, що містить форму для аутентифікації користувача, як показано на рис. 1. Від користувача вимагається ввести своє ім'я, пароль і натиснути кнопку «ПЕРЕДАТИ». Якщо дані введені неправильно, то можна натиснути кнопку «ОЧИСТИТИ». У зв'язку із тим, що розглядається клієнтський сценарій, створіть «заглушку» обробника аутентифікації користувача у вигляді HTML – сторінку із текстом «Документ, який нібито обробляє інформацію про користувача». «Заглушка» спрацює при натисканні на кнопку «ПЕРЕДАТИ».

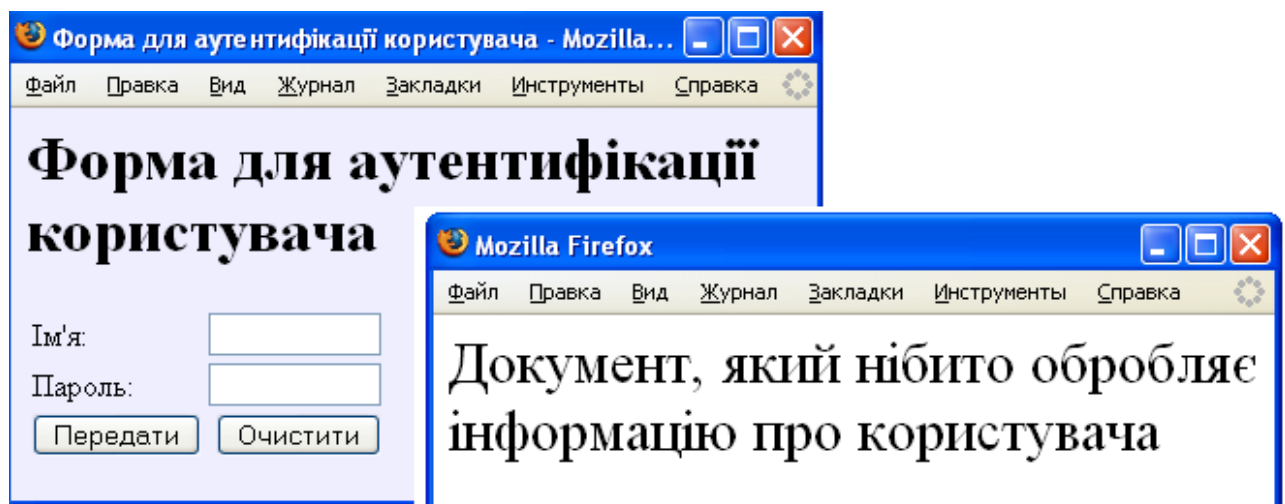


Рис. 1 – Форма для аутентифікації користувача

Запустіть PHP Storm і створіть у ньому файл **Test2JS.html** і в контейнері <BODY> введіть текст:

<H1>Форма для аутентифікації користувача</H1>

Під цим текстом введіть дескриптор <FORM> і вкажіть URL-адресу, по якій слід передати введену інформацію (тобто URL-адресу сценарію, що оброблює форму). Для цього скористайтесь атрибутом ACTION дескриптора <FORM>..Далі вкажіть метод пересилки введеної інформації сценарію. Для цього скористайтесь атрибутом METHOD дескриптора <FORM>. Вкажіть ім'я даної форми: form1. Для цього скористайтесь атрибутом NAME дескриптора <FORM>. Для нашого випадку, коли дані форми, нібито обробляються файлом zaglyshka.html. За-пис вищезгаданого виглядає так:

<FORM METHOD=GET ACTION="zaglyshka.html" NAME=form1>

Для того, щоб форма мала охайний вигляд, елементи форми необхідно вставити у таблицю, яка б мала 3 рядка і 2 стовпчика.

В першій комірці (комірки рахують з права наліво і зверху вниз) записуємо текст «Ім'я». У другій комірці за допомогою елементу TYPE="text" створюємо вікно для вводу тексту. Наприклад:

<INPUT TYPE="TEXT" NAME="NM" SIZE=11 >

Дане текстове поле повинно вміщує 1 рядок із 11 символів.

В першій комірці другого рядка таблиці записуємо текст «Пароль». У другій комірці за допомогою елементу TYPE="password" створюємо вікно для вводу паролю з 11 символів. Наприклад:

<INPUT TYPE="PASSWORD" NAME="PASS" SIZE=11>

У комірці третього рядка таблиці вставте кнопки типу «SUBMIT» та «RESET». Наприклад:

< INPUT TYPE="SUBMIT" VALUE="Передати">

<INPUT TYPE="RESET" VALUE="Очистити">

Остаточний код сторінки повинен мати наступний вигляд (лістинг 1):

Лістинг 1 – Код сторінки із формою аутентифікації користувача

<HTML>

<HEAD>

<TITLE>Форма для аутентифікації користувача</TITLE>

<META HTTP-EQUIV = "Content-Type" CONTENT = "text/html; CHARSET = windows-1251">

</HEAD>

<BODY BGCOLOR="#EEEEFF">

<H1>Форма для аутентифікації користувача</H1>

<FORM METHOD=GET ACTION="zaglushka.html">

<TABLE BORDER=0>

```
<TR>
<TD ALIGN="LEFT">Ім'я:</TD>
<TD ALIGN="RIGHT" ><INPUT TYPE="TEXT" NAME="NM" SIZE=11></TD>
</TR>
<TR>
<TD ALIGN="LEFT">Пароль:</TD>
<TD ALIGN="RIGHT"><INPUT TYPE="PASSWORD" NAME="PASS" SIZE=11>
</TD>
</TR>
<TR ALIGN="CENTER">
<TD ALIGN="LEFT"><INPUT TYPE="SUBMIT" VALUE="Передати"></TD>
<TD ALIGN="RIGHT"><INPUT TYPE="RESET" VALUE="Очистити"></TD>
</TR>
</TABLE>
</FORM>
</BODY>
</HTML>
```

Далі створіть сторінку «заглушку» обробника аутентифікації користувача у вигляді HTML – сторінки із текстом «Документ, який нібито обробляє інформацію про користувача». Збережіть створений документ у папці даного проекту під назвою **zaglushka.html**. Перевірте, чи спрацьовує «заглушка» при натискання на кнопку «ПЕРЕДАТИ» форми аутентифікації користувача (рис. 1).

2.Доповніть файл **Test2JS.html**, створений у попередньому завданні формами, наведеними на рис.2.

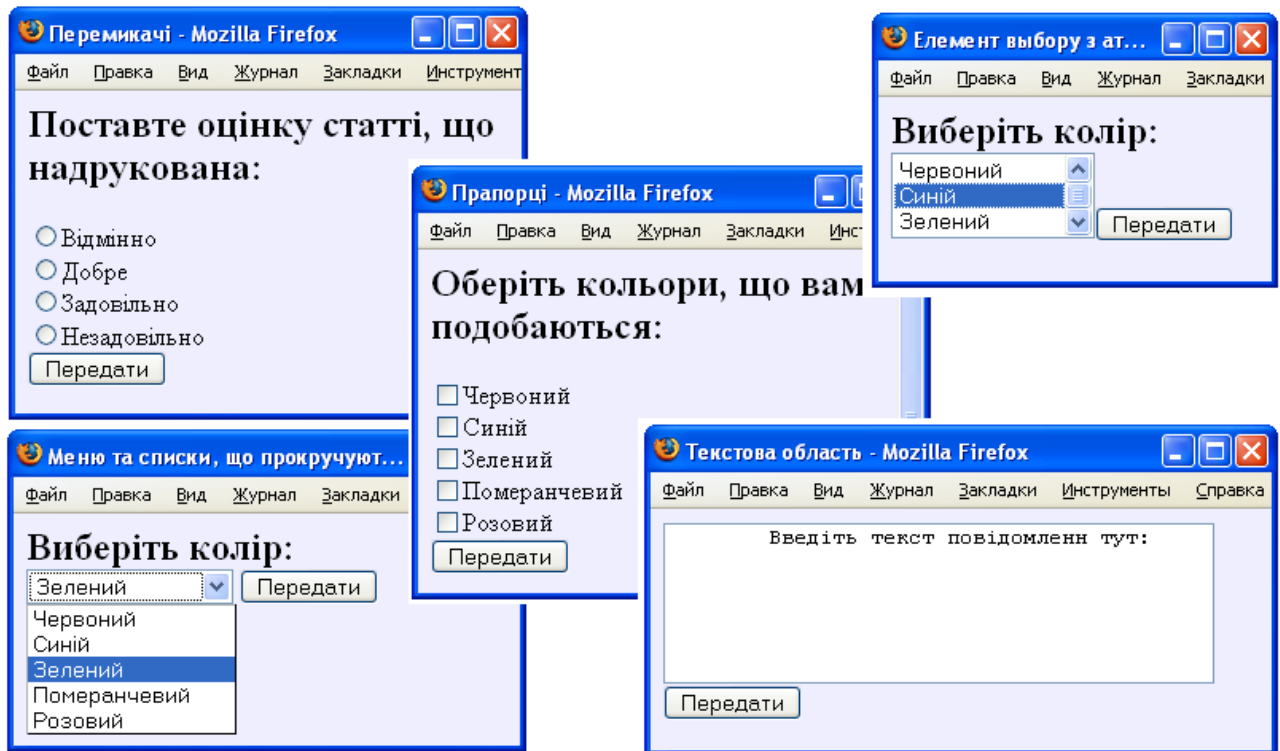


Рис. 2 – Додаткові форми для файлу аутентифікації користувача

Відкрийте файл **Test2JS.html**, Використовуючи інформацію про елементи форм, яка викладена в таблицях 1 та 2 із теоретичних відомостей внесіть відповідні зміни. Як приклад, наведено код для форми «Виберіть колір» із обраним синім кольором.

Лістинг 2 – Код форми «Виберіть колір»

```
<FORM METHOD="GET" ACTION="zaglushka.html">
```

```
<SELECT NAME="Color" SIZE=3 MULTIPLE>
```

```
<OPTION >Червоний
```

```
<OPTION SELECTED >Синій
```

```
<OPTION>Зелений
```

```
<OPTION SELECTED>Померанчевий
```

```
<OPTION>Розовий
```

```
</SELECT>
```

```
<INPUT TYPE="SUBMIT" VALUE="Передати">
```

Рекомендується спробувати декілька типів введення у форму.

3. Під час лекції 1 (Ознайомлення з елементами HTML) ми ознайомились із властивістю тегу посилання для відправлення користувачем повідомлення із сайту на вашу електронну пошту. Використовуючи JS це можна зробити, скориставшись формами на сторінці. Крім того можна ще організувати й перевірку полів форми на випадок, коли користувач не заповнив усі поля.

Створіть форму, яка вирішує описані завдання (рис.3).

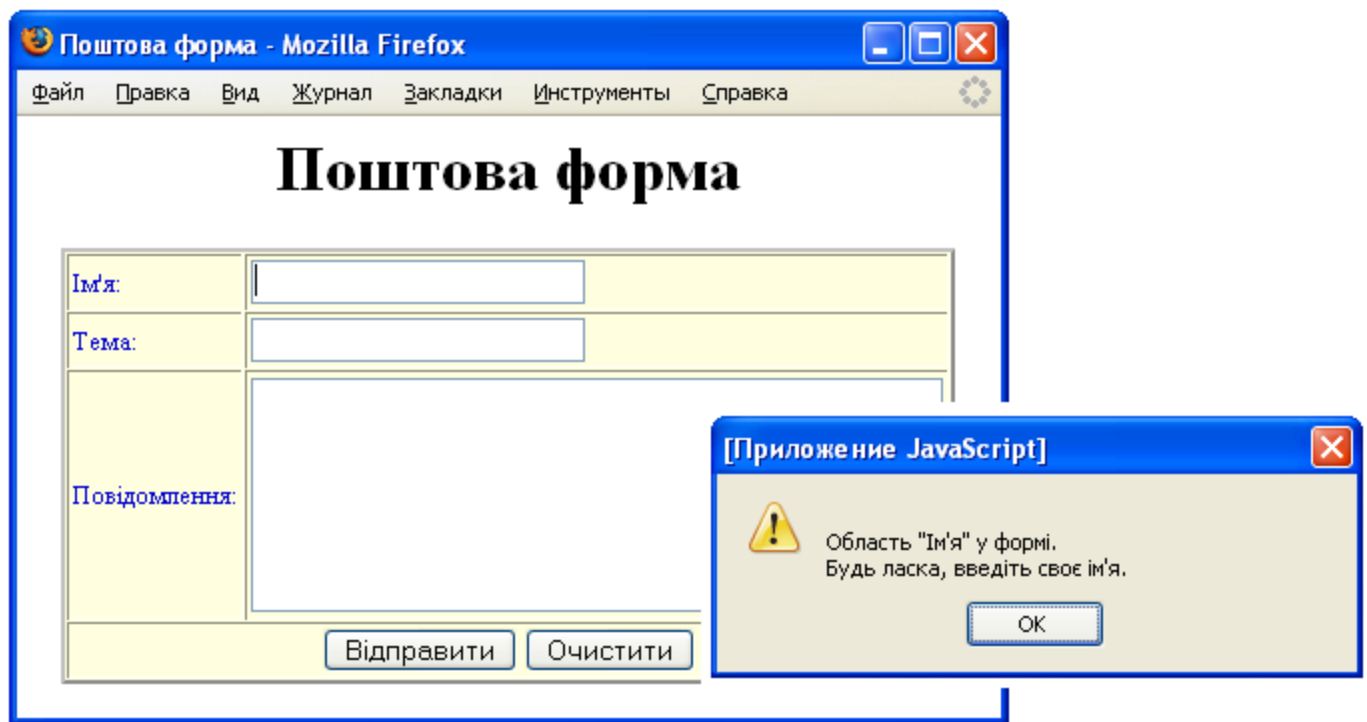


Рис. 3 – Поштова форма, що реагує на пусті поля

Для виконання завдання наберіть код, що реалізує форму (лістинг 3).

Лістинг 3 – Код поштової форми.

```
<HTML>
```

```
<HEAD>
```

```
<TITLE>Поштова форма</TITLE>
```

```
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; CHARSET=windows-1251">
```

```
</HEAD>
```

```
<BODY>
```

```
<H1 align=center>Поштова форма</H1>
```

```
<FORM NAME="mailer" METHOD="post" ACTION="" ENCTYPE= "text/plain"
onSubmit="(document.mailer.action += mailtoandSubject)">

<table border=2 align=center cellspacing=1 cellpadding=2 BgColor=#FFFFE0>

<tr><td><FONT size="2" color="#0000CD">Ім'я:</font></td>

<td><INPUT    TYPE="text"    NAME="Name"    size="24"    onChange=
"msg(this.form)"></td></tr>

<tr><td><FONT size="2" color="#0000CD">Тема:</font></td>

<td><INPUT    TYPE="text"    NAME="Subject"    size="24"    onChange=
"msg(this.form)"></td></tr>

<tr><td><FONT size="2" color="#0000CD">Повідомлення:</font></td>

<td><TEXTAREA  NAME="Message"  COLS=40  ROWS=6  onChange=
"msg(this.form)"></TEXTAREA></td></tr>

<tr><td colspan=2 align=center><INPUT TYPE = "submit" VALUE = "Відн-
правити" ONCLICK="return checkIt()">

<INPUT TYPE=reset VALUE=" Очистити " ></td></tr></table>

</FORM>

</BODY>

</HTML>
```

Кожному полю форми призначено конкретне ім'я, яке задано атрибутом NAME (тобто: Name, Subject, Message).

При натисканні на кнопку "Відправити" виконується виклик функції checkIt(), якщо поле не заповнено, виводиться вікно з повідомленням і кнопкою Ok, після натискання на яку курсор переходить на незаповнене поле.

Опис функції checkIt() знаходиться у JavaScript-кодi, наведеному у лістингу 3.1. Код вмонтовується у сторінку із формою в заголовку документа між тегами <HEAD>...</HEAD>. Програма виконує перевірку форми на незаповнені поля.

Лістинг 3.1 – Код перевірки заповнення полів поштової форми.

```
<SCRIPT LANGUAGE="JavaScript">

function checkIt() { // функція перевірки полів форми
```

//-----

```
if (document.forms.mailer.Name.value != "") { // функція перевірки поля Name  
} else {  
    alert("\nОбласть \"Ім'я\" у формі. \n\nБудь ласка, введіть своє ім'я."); // ви-  
        вводить повідомлення, якщо поле Name не заповнено  
    document.forms.mailer.Name.focus(); // вертає курсор на поле Name  
    return false;  
}
```

//-----

```
if (document.forms.mailer.Subject.value != "") { // функція перевірки поля Subject  
} else {  
    alert("\nОбласть \"Тема\" в формі. \n\nБудь ласка, введіть, введіть те-му."); //  
        виводить повідомлення, якщо поле Subject не заповнено  
    document.forms.mailer.Subject.focus(); // вертає курсор на поле Subject  
    return false;  
}
```

//-----

```
if (document.forms.mailer.Message.value != "") { // функція перевірки поля  
    Message  
    return true; // ВСЕ ВІДМІННО  
} else {  
    alert("\nОбласть \"Повідомлення\" в формі. \n\nБудь ласка, напишіть по-  
        відомлення."); // виводить повідомлення, якщо поле Message не заповнено  
    document.forms.mailer.Message.focus(); // вертає курсор на поле Message  
    return false;  
}
```

//-----


```
}  
  
function msg() { // функція відправки  
  
    document.mailer.action = "mailto:astral990@meta.ua"  
  
    mailtoandSubject = (('Subject=' + document.mailer.Subject.value) + '&Body=' +  
    document.mailer.Message.value);  
  
}  
  
</SCRIPT>
```

Для зручності, функції перевірки розділені коментарем: `//-----`.

Остання функція повинна містити значення **return true**. Це значення повідомляє про успішне закінчення перевірки.

Змінна `astral990@meta.ua` вказує на пошту, куди будуть надходити повідомлення (замість неї можете вказати власну поштову адресу). Наведена форма використовує поштову програму користувача, яка визначена за замовченням, і, якщо вона не налаштована, нічого не відправиться.

На рис.3.1 виведено повідомлення, яке видається після відправки даних форми у випадку, коли поштова програма Outlook Express не використовується за замовченням.

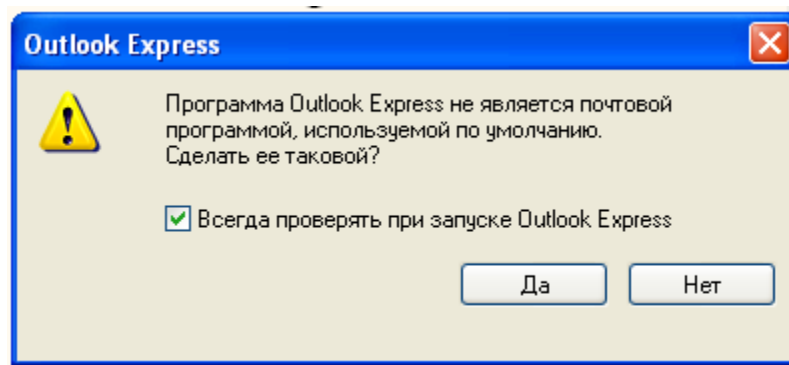


Рис. 3.1 – Повідомлення від поштової програми Outlook Express.

Якщо поштова програма налаштована, з'явиться наступне вікно (рис. 3.2).

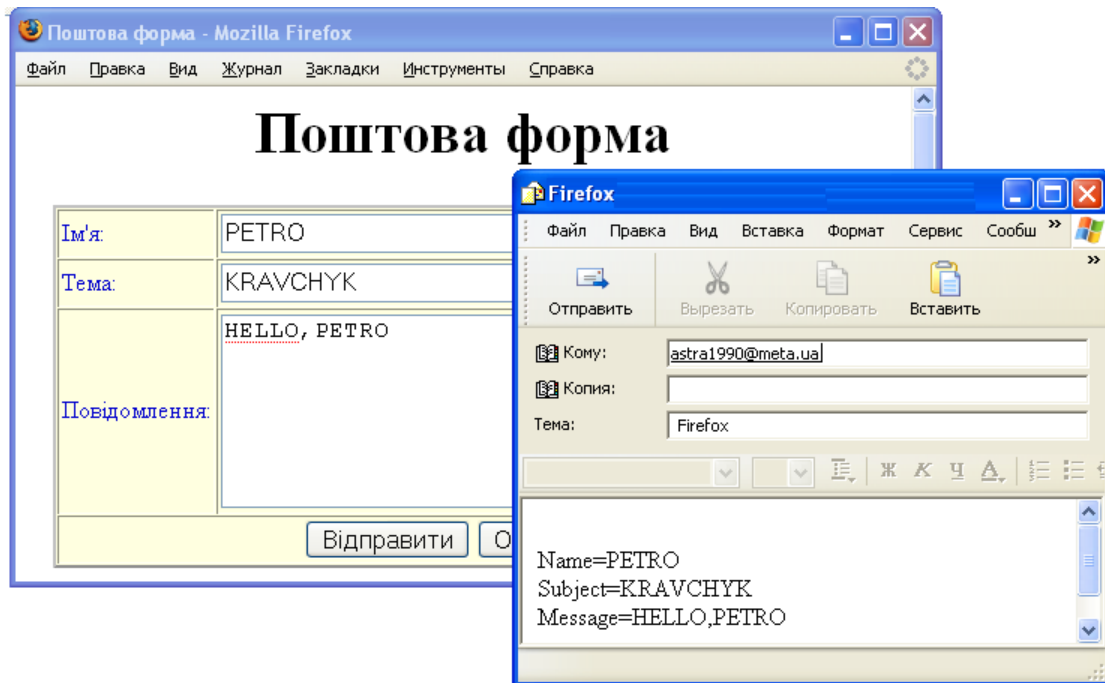


Рис. 3.7 – Вікно поштової програми Outlook Express.

4. Створіть веб-сторінку, що демонструє як працюють найбільш поширені обробників подій. Кожний із прикладів внесено у комірку таблиці 3×3, як показано на рис. 4. Після того як подія настала, виводиться відповідне повідомлення наступного змісту: «Сценарій запущено в момент ...», наприклад торкання клавіші або клацання по клавіші і т.п.



Рис. 4 – веб-сторінка, що демонструє як працюють обробників подій.

Для виконання завдання наберіть код таблиці розмірами 3×3 (пригадаємо із Практичної роботи 3-4 – створення таблиці в html). Відкрийте створений документ і у кожену комірку запишіть код для створення конкретного елемента

форми із сценарієм, що виводить відповідне повідомлення. Як приклад, далі наведено фрагмент коду останньої комірки таблиці.

```
<td><FONT size="3" color="#0000CD"><div align="center">
```

```
Введіть символ<br><br>
```

```
<form name="Formpress2">
```

```
<input name="InpText" value="" size="4" onkeyup="Fpress2()" type="text">
```

```
</form></font></div></td>
```

Сценарій JavaScript, що реагує на подію «відпускання клавіші» (вводу символу) показано нижче:

```
<script language="JavaScript">
```

```
function Fpress2()
```

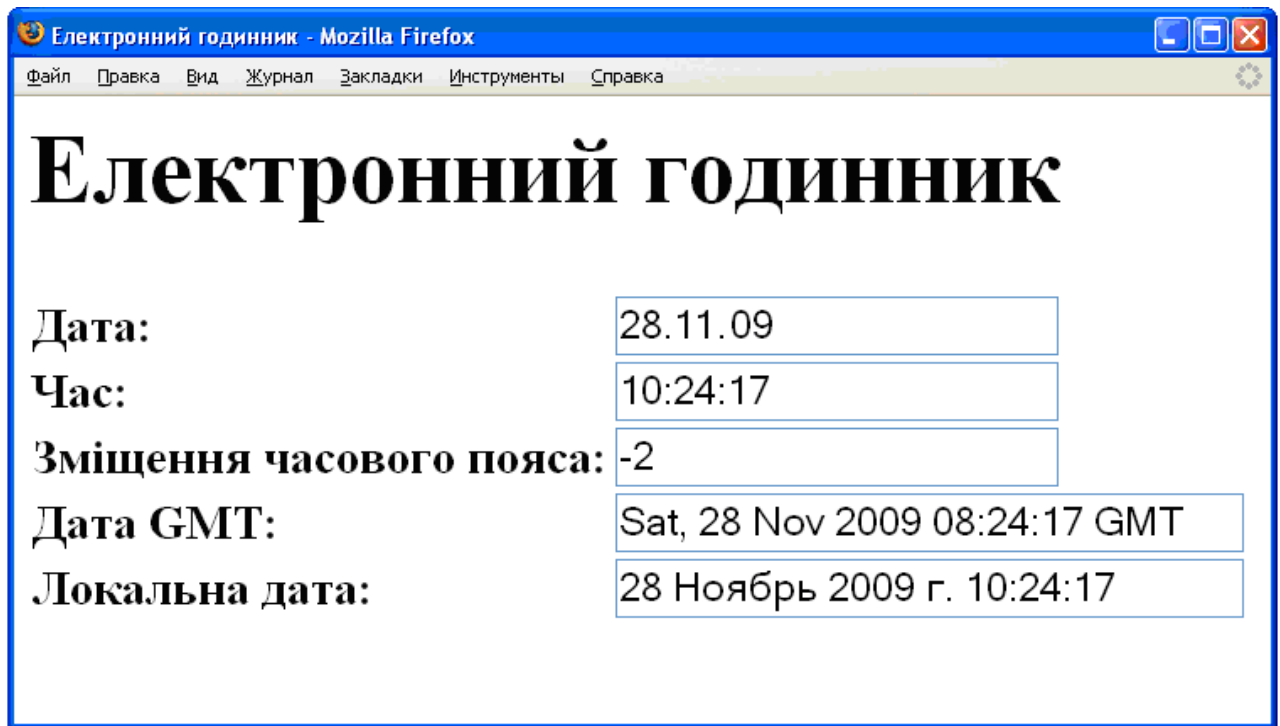
```
{
```

```
alert("Сценарій запущено в момент відпускання клавіші");
```

```
}
```

```
</script>
```

5. За допомогою сценарію JavaScript та форми створіть електронний годинник, що буде показувати не тільки час, але і дату в різних форматах з урахуванням зміщення часового поясу (рис. 5).



Дата:	<input type="text" value="28.11.09"/>
Час:	<input type="text" value="10:24:17"/>
Зміщення часового пояса:	<input type="text" value="-2"/>
Дата GMT:	<input type="text" value="Sat, 28 Nov 2009 08:24:17 GMT"/>
Локальна дата:	<input type="text" value="28 Ноябрь 2009 г. 10:24:17"/>

Рис. 5 – Форма «електронний годинник»

Завдання виконуються у наступній послідовності:

Наберіть код HTML-сторінки із назвою «Електронний годинник» (заголовок першого рівня).

Під назвою створіть форму (<FORM NAME="Clock">), що містить п'ять текстових полів:

```
<INPUT TYPE="text" NAME="dat" SIZE="20" >
```

```
<INPUT TYPE="text" NAME="time" SIZE="20" >
```

```
<INPUT TYPE="text" NAME="timeZone" SIZE="20" >
```

```
<INPUT TYPE="text" NAME="gmt" SIZE="30" >
```

```
<INPUT TYPE="text" NAME="loc" SIZE="30" >
```

Назва полів показана на рис.5. Для вирівнювання елементів форми скористайтесь таблицею із п'яти рядків та двох стовпчиків.

Наступним кроком є додавання сценарію JavaScript.

У заголовку створеного документа запишіть опис функції updateClock.(лістинг 5)

Лістинг 5 – Код функції updateClock.

```
<SCRIPT LANGUAGE="JavaScript">
<!--
nTimer = 0;
var szCurrentTime = "";
var szCurrentDate = "";
var szTimeZone = "";
var szGMT = "";
var szLocal = "";
function updateClock()
{
var dtDate = new Date();
var nHours = dtDate.getHours();
var nMinutes = dtDate.getMinutes();
var nSeconds = dtDate.getSeconds();
szCurrentTime = nHours + ":" + nMinutes +
":" + nSeconds;
szCurrentDate = dtDate.getDate() + "." +
dtDate.getMonth() + "." + dtDate.getYear();
szTimeZone = dtDate.getTimezoneOffset() / 60;
szGMT = dtDate.toGMTString();
szLocal = dtDate.toLocaleString();
Clock.time.value = szCurrentTime;
Clock.dat.value = szCurrentDate;
Clock.timeZone.value = szTimeZone;
Clock.gmt.value = szGMT;
Clock.loc.value = szLocal;
nTimer = setTimeout("updateClock()", 1000);
}
// -->
</SCRIPT>
```

Далі, під описом форми введіть звертання до функції *updateClock* у вигляді:

```
<SCRIPT LANGUAGE="JavaScript">
<!--
updateClock();
// -->
</SCRIPT>
```

Годинник запускається відразу після завантаження документа HTML у вікно браузера. При цьому керування одержує функція *updateClock*. Ця функція визначена в заголовку, а її виклик перебуває наприкінці області тіла документа HTML.

Одержавши керування, функція *updateClock* створює об'єкт *Data* для поточної дати, установленної в комп'ютері:

```
var dtDate = new Date();
```

Потім функція визначає три компоненти часу (години, хвилини й секунди), викликаючи для цього методи `getHours`, `getMinutes` й `getSeconds`, визначені в класі `Date`:

```
var nHours = dtDate.getHours();
```

```
var nMinutes = dtDate.getMinutes();
```

```
var nSeconds = dtDate.getSeconds();
```

Отримані в такий спосіб значення комбінуються в текстовому рядку `szCurrentTime`:

```
szCurrentTime = nHours + ":" + nMinutes +  
":" + nSeconds;
```

Надалі час буде записано функцією `updateClock` у поле `time` форми `Clock`, як це показано нижче:

```
Clock.time.value = szCurrentTime;
```

Рядок поточної дати виходить у результаті виклику методів `getDate`, `getMonth` й `getFullYear`:

```
szCurrentDate = dtDate.getDate() + "." +  
dtDate.getMonth() + "." + dtDate.getFullYear();
```

Цей рядок записується в поле `dat` форми `Clock`:

```
Clock.dat.value = szCurrentDate;
```

Для обчислення зсуву годинного пояса ми скористалися методом `getTimezoneOffset`, що повертає значення у хвилинах:

```
szTimeZone = dtDate.getTimezoneOffset() / 60;
```

Для того щоб обчислити зсув у годинах, ми розділили результат, отриманий від методу `getTimezoneOffset`, на 60.

Обчислений зсув годинного пояса записується в поле `timeZone` форми `Clock`:

```
Clock.timeZone.value = szTimeZone;
```

Для одержання часу за Гринвічем ми скористалися методом `toGMTString`:

```
szGMT = dtDate.toGMTString();
```

Отримане значення буде записано в поле `gmt` форми `Clock`:

```
Clock.gmt.value = szGMT;
```

І, нарешті, локальну дату й час ми визначаємо за допомогою методу `toLocaleString`:

```
szLocal = dtDate.toLocaleString();
```

Результат записується в поле `loc` форми `Clock`:

```
Clock.loc.value = szLocal;
```

Періодичне виконання функції `updateClock` досягається за допомогою методу `setTimeout`:

```
nTimer = setTimeout("updateClock()", 1000);
```

Тут ми вказали, що період відновлення показань наших годин повинен дорівнювати одній секунді.

Самостійна робота студента

Завдання самостійної роботи оформлюються кожне в окремому файлі та поміщаються у папку на Гугл-диску. Також код кожного завдання разом із скрін-шотом виконання додається до звіту Практичної роботи №10. Отже, папка Практична робота 10 повинна містити 6 файлів проектів, а також звіт із кодом до кожного із 6-ти завдань та скрін-шотами їх виконання у браузері.

Завдання 1.

Створити сценарій «Кольори фону». Скрипт містить список, при виборі елемента якого кольори фону сторінки міняються на обраний користувачем. Кольори в списку (і порядок їхнього проходження) повинні відповідати Вашому варіанту (див. таблицю 1). Номер варіанту відповідає порядковому номеру студента у списку групи.

Таблиця 1 – Вихідні дані для виконання самостійного завдання №1

Номер варіанта	Колір 1	Колір 2	Колір 3
1	Червоний	Білий	Синій
2	Синій	Білий	Жовтий
3	Зелений	Білий	Синій
4	Коричневий	Білий	Сірий
5	Червоний	Жовтий	Сірий
6	Синій	Жовтий	Червоний
7	Зелений	Жовтий	Червоний
8	Коричневий	Жовтий	Червоний
9	Червоний	Голубий	Зелений
10	Синій	Голубий	Зелений
11	Сірий	Голубий	Зелений
12	Коричневий	Голубий	Зелений
13	Червоний	Сірий	Коричневий
14	Жовтий	Червоний	Синій
15	Зелений	Сірий	Синій
16	Жовтий	Червоний	Зелений
17	Голубий	Зелений	Синій
18	Голубий	Зелений	Жовтий
19	Зелений	Білий	Синій
20	Коричневий	Білий	Сірий
21	Червоний	Жовтий	Сірий
22	Синій	Зелений	Червоний
23	Зелений	Чорний	Червоний
24	Коричневий	Голубий	Червоний
25	Червоний	Голубий	Зелений
26	Червоний	Білий	Зелений
27	Синій	Білий	Зелений
28	Чорний	Білий	Зелений
29	Коричневий	Білий	Чорний
30	Зелений	Жовтий	Чорний

Завдання 2.

Створіть сценарій «Тестування». Скрипт, призначений для перевірки знання таблиці істинності логічного елемента. Тип логічної функції, форму введення інформації й вид події, що запускає перевірку, вибрати з таблиці 2 відповідно до варіанта. Для позначення подій прийняті наступні скорочення: ЩЛК - щиклик лівою кнопкою миші (onClick); ПЩК - подвійний щиклик лівою кнопкою миші (onDblClick); НМК - наведення курсору миші на кнопку (onMouseOver). Для інформації щодо таблиць істинності скористайтесь онлайн-довідником.

Таблиця 2 – Вихідні дані для виконання самостійного завдання №2

Номер варіанта	Логічна операція	Форма	Подія
1	I	Кнопка	ЩЛК
2	I-НЕ	Перемикач	ДЩК
3	АБО-НЕ	Вмикач	НKK
4	ВИКЛЮЧАЮЧЕ АБО	Список	ЩЛК
5	ВИКЛЮЧАЮЧЕ АБО НЕ	Кнопка	ДЩК
6	I	Перемикач	НKK
7	I-НЕ	Вмикач	ЩЛК
8	АБО-НЕ	Список	ДЩК
9	ВИКЛЮЧАЮЧЕ АБО	Кнопка	НKK
10	ВИКЛЮЧАЮЧЕ АБО НЕ	Перемикач	ЩЛК
11	I	Вмикач	ДЩК
12	I-НЕ	Список	НKK
13	АБО-НЕ	Кнопка	ЩЛК
14	ВИКЛЮЧАЮЧЕ АБО	Список	ДЩК
15	ВИКЛЮЧАЮЧЕ АБО НЕ	Вмикач	ЩЛК
16	I-НЕ	Список	ДЩК
17	АБО-НЕ	Кнопка	ЩЛК
18	ВИКЛЮЧАЮЧЕ АБО	Перемикач	ДЩК
19	ВИКЛЮЧАЮЧЕ АБО НЕ	Вмикач	НKK
20	I	Список	ЩЛК
21	I-НЕ	Кнопка	ДЩК
22	АБО-НЕ	Перемикач	НKK
23	ВИКЛЮЧАЮЧЕ АБО	Вмикач	ЩЛК
24	ВИКЛЮЧАЮЧЕ АБО НЕ	Список	ДЩК
25	I-НЕ	Кнопка	НKK
26	АБО-НЕ	Перемикач	ЩЛК
27	ВИКЛЮЧАЮЧЕ АБО	Вмикач	ДЩК
28	ВИКЛЮЧАЮЧЕ АБО НЕ	Список	НKK
29	I	Кнопка	ЩЛК
30	I-НЕ	Список	ЩЛК

Таблиця 2.1 –Зразки оформлення форм для виконання самостійного завдання №2

Назва форми	Вигляд форми																						
<p>«Кнопка»</p> <p>Перевірка знання таблиці істинності за допомогою командних кнопок</p>	<p>Оберіть правильну відповідь</p> <div><p>Логічний елемент АБО (операція диз'юнкції, логічне додавання)</p><table><tr><th>Логічні вирази</th><th colspan="2">Можливі відповіді</th></tr><tr><td>$0 \vee 0 =$</td><td><input type="radio"/> 1</td><td><input type="radio"/> 0</td></tr><tr><td>$0 \vee 1 =$</td><td><input type="radio"/> 1</td><td><input type="radio"/> 0</td></tr><tr><td>$1 \vee 0 =$</td><td><input type="radio"/> 1</td><td><input type="radio"/> 0</td></tr><tr><td>$1 \vee 1 =$</td><td><input type="radio"/> 1</td><td><input type="radio"/> 0</td></tr></table></div>	Логічні вирази	Можливі відповіді		$0 \vee 0 =$	<input type="radio"/> 1	<input type="radio"/> 0	$0 \vee 1 =$	<input type="radio"/> 1	<input type="radio"/> 0	$1 \vee 0 =$	<input type="radio"/> 1	<input type="radio"/> 0	$1 \vee 1 =$	<input type="radio"/> 1	<input type="radio"/> 0							
Логічні вирази	Можливі відповіді																						
$0 \vee 0 =$	<input type="radio"/> 1	<input type="radio"/> 0																					
$0 \vee 1 =$	<input type="radio"/> 1	<input type="radio"/> 0																					
$1 \vee 0 =$	<input type="radio"/> 1	<input type="radio"/> 0																					
$1 \vee 1 =$	<input type="radio"/> 1	<input type="radio"/> 0																					
<p>«Перемикач»</p> <p>Перевірка знання таблиці істинності за допомогою перемикачів (радіокнопок).</p>	<p>Виберіть правильну відповідь</p> <div><p>Логічний елемент АБО</p><table><tr><th rowspan="2">Логічні вирази</th><th colspan="2">Можливі відповіді</th><th rowspan="2">Для перевірки прийнятого рішення зробіть клік по кнопці</th></tr><tr><th>1</th><th>0</th></tr><tr><td>$0 \vee 0 =$</td><td><input type="radio"/></td><td><input type="radio"/></td><td><input type="button" value="Перевірка"/></td></tr><tr><td>$0 \vee 1 =$</td><td><input type="radio"/></td><td><input type="radio"/></td><td><input type="button" value="Перевірка"/></td></tr><tr><td>$1 \vee 0 =$</td><td><input type="radio"/></td><td><input type="radio"/></td><td><input type="button" value="Перевірка"/></td></tr><tr><td>$1 \vee 1 =$</td><td><input type="radio"/></td><td><input type="radio"/></td><td><input type="button" value="Перевірка"/></td></tr></table></div>	Логічні вирази	Можливі відповіді		Для перевірки прийнятого рішення зробіть клік по кнопці	1	0	$0 \vee 0 =$	<input type="radio"/>	<input type="radio"/>	<input type="button" value="Перевірка"/>	$0 \vee 1 =$	<input type="radio"/>	<input type="radio"/>	<input type="button" value="Перевірка"/>	$1 \vee 0 =$	<input type="radio"/>	<input type="radio"/>	<input type="button" value="Перевірка"/>	$1 \vee 1 =$	<input type="radio"/>	<input type="radio"/>	<input type="button" value="Перевірка"/>
Логічні вирази	Можливі відповіді		Для перевірки прийнятого рішення зробіть клік по кнопці																				
	1	0																					
$0 \vee 0 =$	<input type="radio"/>	<input type="radio"/>	<input type="button" value="Перевірка"/>																				
$0 \vee 1 =$	<input type="radio"/>	<input type="radio"/>	<input type="button" value="Перевірка"/>																				
$1 \vee 0 =$	<input type="radio"/>	<input type="radio"/>	<input type="button" value="Перевірка"/>																				
$1 \vee 1 =$	<input type="radio"/>	<input type="radio"/>	<input type="button" value="Перевірка"/>																				
<p>«Вмикач»</p> <p>Перевірка знання таблиці істинності за допомогою допомогою вмикачів (прапорців)</p>	<p>Виберіть правильну відповідь</p> <div><p>Логічний елемент АБО (операція диз'юнкції, логічне додавання)</p><table><tr><th>Логічний вираз</th><th>Для вводу 1 встановіть прапорець</th></tr><tr><td>$0 \vee 0 =$</td><td><input type="checkbox"/></td></tr><tr><td>$0 \vee 1 =$</td><td><input type="checkbox"/></td></tr><tr><td>$1 \vee 0 =$</td><td><input type="checkbox"/></td></tr><tr><td>$1 \vee 1 =$</td><td><input type="checkbox"/></td></tr></table><p><input type="button" value="Перевірити рішення"/></p></div>	Логічний вираз	Для вводу 1 встановіть прапорець	$0 \vee 0 =$	<input type="checkbox"/>	$0 \vee 1 =$	<input type="checkbox"/>	$1 \vee 0 =$	<input type="checkbox"/>	$1 \vee 1 =$	<input type="checkbox"/>												
Логічний вираз	Для вводу 1 встановіть прапорець																						
$0 \vee 0 =$	<input type="checkbox"/>																						
$0 \vee 1 =$	<input type="checkbox"/>																						
$1 \vee 0 =$	<input type="checkbox"/>																						
$1 \vee 1 =$	<input type="checkbox"/>																						
<p>«Список»</p> <p>Перевірка знання таблиці істинності за допомогою списку із якого необхідно вибрати правдиву відповідь</p>	<p>Який логічний елемент описано за допомогою цієї таблиці істинності?</p> <table><tr><th>x_2</th><th>x_1</th><th>y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table> <p>Виберіть відповідь із запропонованого списку</p> <div><p>Елемент І (кон'юнкція) Елемент АБО (диз'юнкція) Елемент ВИКЛЮЧАЮЧЕ АБО (нерівнозначність) Елемент АБО-НЕ (стрілка Пірса) Елемент І-НЕ (штрих Шеффера) Елемент ВИКЛЮЧАЮЧЕ АБО-НЕ (рівнозначність)</p></div>	x_2	x_1	y	0	0	0	0	1	1	1	0	1	1	1	1							
x_2	x_1	y																					
0	0	0																					
0	1	1																					
1	0	1																					
1	1	1																					

Завдання 3.

Розробити інтерактивну форму нескладного тестування учнів початкових класів відповідно до свого варіанту (див. табл. 3). Учень має вибрати одну вірну відповідь з трьох запропонованих.

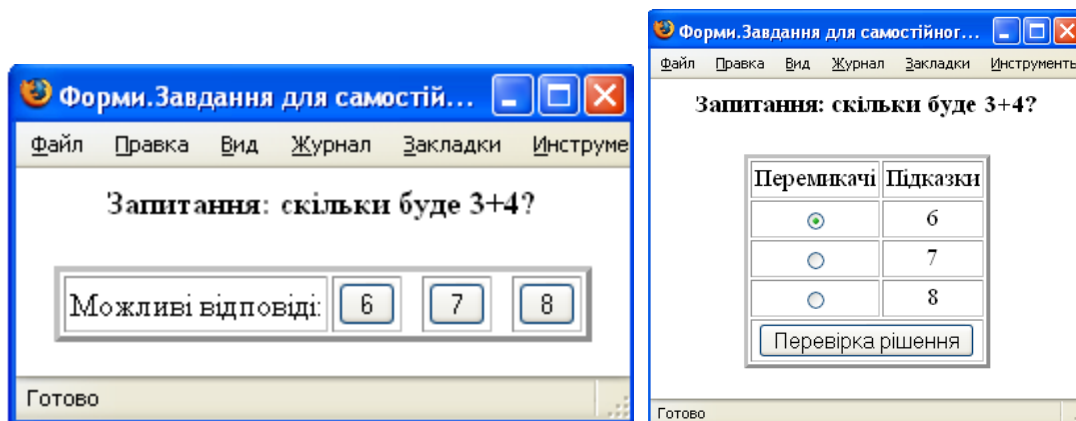


Рис. 3 – Зразки форм для виконання самостійного завдання №3

Таблиця 3 – Вихідні дані для виконання самостійного завдання №3

Номер варіанта	Зміст запитання	Вигляд форми
1	Скільки буде 3×4 ?	Список
2	Скільки буде $12 - 3$?	Кнопка
3	Скільки буде $5 + 2$?	Перемикач
4	Скільки буде $10 \div 5$?	Список
5	Хто сказав мяу?	Кнопка
6	Хто сказав гав?	Перемикач
7	Хто сказав му-му?	Список
8	Хто сказав цвірінь-цвірінь?	Кнопка
9	Хто літає?	Перемикач
10	Що літає?	Список
11	Хто бігає?	Кнопка
12	Хто скаче?	Перемикач
13	Скільки хвилин у годині?	Список
14	Скільки секунд у хвилині?	Кнопка
15	Скільки днів у місяці?	Перемикач
16	Скільки тижнів у році?	Список
17	Якого кольору небо?	Кнопка
18	Якого кольору сонце?	Перемикач
19	Якого кольору трава?	Список
20	Якого кольору море?	Кнопка

21	Що можна пити?	Перемикач
22	Що можна їсти?	Список
23	Що можна ковтати?	Кнопка
24	Що можна жувати?	Перемикач
25	Скільки буде 4×4 ?	Список
26	Скільки буде $16 - 9$?	Кнопка
27	Скільки буде $5 + 6$?	Перемикач
28	Скільки буде $15 \div 3$?	Список
29	Що є металом?	Кнопка
30	Що є деревиною?	Перемикач

Завдання 4

Модифікувавши код із тренувальних завдань, створіть форму для реєстрації нового користувача Web-сайту (уводяться реєстраційні дані й персональні дані).

Завдання 5

Модифікувавши код із тренувальних завдань, розробіть програму обчислення суми двох чисел, у якій вихідні числа вводяться в окремих полях форми, а результат виводиться в діалоговому вікні й/або в окремому полі введення цієї ж форми.

Завдання 6

Модифікувавши код із тренувальних завдань, розробіть форму для відправлення адреси e-mail (*підказка*: у властивості action необхідно вказати "mailto" й адресу електронної пошти одержувача).