

Лабораторна робота №9

«Робота з формами в JS»

Мета роботи: навчитися управляти формою через сценарій.

Форма використовується для введення користувачем через вікно браузера даних і передачі їх на веб-сервер. Форма складається з контейнера <FORM> ... </ FORM> і вкладених в нього тегів (елементів) <INPUT>, <SELECT> і <TEXTAREA>.

Для зменшення навантаження на мережу і веб-сервер можна перевіряти введені користувачем дані на браузері за допомогою сценарію на JavaScript. Якщо в даних виявиться помилка, то користувачеві надається можливість її виправити. Введені правильно дані відправляються на веб-сервер. Використання сценарію для управління формою демонструється в прикладі 3.1.

Приклад 1.1

```
<HTML>
<HEAD> <TITLE> Перша сторінка </ title> </ head>
<H2> Назвіться, будь ласка </ h2>
<FORM name = F1 METHOD = "POST" ACTION = "">
  Ім'я ....
  <INPUT TYPE = "text" NAME = "name"> <BR>
  Вік <INPUT TYPE = "text" NAME = "age">
  <P> <INPUT TYPE = "submit" VALUE = "ВВЕДЕННЯ" onclick =
"Proverka ()">
</ FORM>
<SCRIPT>
function Proverka ()
{Im = document.F1.name.value
vozs = document.forms [0] .elements [1] .value st = ""
if (im == "") st = "ім'я \ n"
if (vozs == "") st + = "вік"
if (st == "") document.F1.action = "Prim3_1.php"
else // скасування передачі на веб-сервер
{Str = "Введіть: \ n" + st
  alert (str)
  return = false
}
}
</ Script>
</ Html>
<HTML>
<HEAD> <TITLE> Друга сторінка </ title> </ head>
<BODY>
<H3> П Р І В Е Т С Т В А </ h3>
<? Php
$ Imja = $ _ POST [ "name"]; // прийом параметрів з форми
$ Age = $ _ POST [ "age"]; $ X = "Привіт!
$ Imja. ";
if ($ age> 50) echo "$ x Ви включені в старшу групу.";
elseif ($ age> 30) echo "$ x Ви включених групу середнього
віку.";
else echo "$ x Ваше ставлення до молодіжної групи.";
?>
<P>
<A href="Prim3_1.html"> Повернення </a>
```

```
</ Body>
</ Html>
```

Приклад 1.1 складається з двох сторінок. Перша сторінка служить для введення користувачем даних і їх перевірки за допомогою скрипта, написаного на JavaScript. Якщо дані введені правильно, то вони відправляються на веб-сервер. На веб-сервері отримані дані обробляються скриптом, написаним на мові PHP, формується і пересилається на браузер користувача нова сторінка.

В скрипті, написаному на JavaScript, щоб отримати доступ до даних, що знаходяться в формі, використовуються імена та індекси елементів форми. Для завдання адреси (URL) сторінки, що містить PHP-скрипт, використовується властивість action об'єкта Form.

Отримання даних з спливаючого списку

Іноді можна повністю вирішувати задачу ведення діалогу з користувачем засобами JavaScript, не звертаючись до веб-сервера. У прикладі 1.2 користувач вводить код кольору в моделі RGB і вибирає зі списку назву кольору. Після натискання кнопки Enter на екран виводяться розфарбовані в обрані кольори код і назва кольору.

Приклад 1.2

```
<Html>
<HEAD> <TITLE> Скрипт SELECT </ title> </ head>
<Body>
<!-- Приклад вибору і виводана екранелемента списку Select -->
<SCRIPT>
function select_ ()
{A = document.all.Kod.value // код кольору
b = document.all.Gor.selectedIndex
// номервибранного елемента // списокselect
c = document.all.Gor.options [b] .text // текст
елемента списка
d = document.all.Gor.options [b] .value // передане з форми
// значення <option value = red>
e = "<FONT size = 7 color =" + a + ">" + a + "</ font>" //
трансляція і
document.all.alfa.innerHTML = e // висновок на екран HTML-
рядки
e = "<FONT size = 7 color =" + d + ">" + c + "</ font>"
document.all.beta.innerHTML = e
}
</ Script>
<H2> Підбір відтінків кольору </ h2>
У першому полі потрібно ввести шістнадцятковий код кольору.
<BR>
Наприклад, червоний колір має код FF0000.
<BR> Зі списку в другому полі вибирається для порівняння
<BR> один з основних кольорів (червоний, зелений, синій)
<P> Введіть код кольору
<Input TYPE = "text" name = "Kod"> <P> Виберіть цвет
<SELECT NAME = "Gor">
<Option value = "red"> Червоний </ option>
<Option value = "yellow"> Жовтий </ option>
<Option value = "maroon"> каштановий </ option>
<Option value = green "> Зелений </ option>
<Option value = "blue"> Синий
```

```

</ Select>
<P> <BUTTON onclick = "select _ ()"> Виконати </ button>
<P> <B ID = "alfa"> </ b>
<br> <B ID = "beta"> </ b>
</ Body>
</ Html>

```

У прикладі 1.2 немає необхідності використовувати контейнер `<FORM> ... </ form>`, так як на веб-сервер нічого не передається. Для виведення на екран коду і назви кольору використовується властивість `innerHTML`. Строго кажучи, `innerHTML` варто було б називати не властивістю, а методом. Розглянемо застосування цієї властивості на прикладі виведення на екран введеного користувачем коду кольору. Нехай користувач ввів код кольору `a = FF0000`, тобто червоний.

```

В результаті виконання оператора
e = "<FONT size = 7 color =" + a + ">" + a + "</font>"
сформується рядок
e = "<FONT size = 7 color = FF0000> FF0000 </font>"
В результаті виконання оператора
document.all.alfa.innerHTML = e
в елемент <B ID = "alfa"> </ b> вставиться значення змінної e:
<B ID = "alfa"> <FONT size = 7 color = FF0000> FF0000 </ font>
</b>

```

і браузер виконає перетворений злементов (оператор мови HTML), тобто виведе на екран **FF0000**

Таким чином, за допомогою властивості `innerHTML` можна на стороні браузера програмним шляхом вносити зміни в HTML-документ (сторінку сайту).

Завдання 1.1. Створіть сайт з двох сторінок. Перша сторінка має заголовок **Замовлення меблів**. На ній розташовані два поля зі списками (теги `<SELECT>`), поле (`<INPUT>`) і кнопка (`<SUBMIT>`). З першого поля зі списком користувач вибирає виріб (шафа, стіл, сервант і т.д.). З другого поля зі списком користувач вибирає матеріал (дуб, горіх, бук). У третє поле потрібно ввести кількість замовлених виробів. Після введення даних необхідно перевірити, чи всі дані введені. Якщо виявлена помилка, то потрібно вивести повідомлення і запропонувати її виправити. Правильно введені дані потрібно відправити на веб-сервер. Друга сторінка містить написаний на `php` скрипт, за допомогою якого формується наступне повідомлення:

```

Ваше замовлення прийнято
Замовлено виріб - назва замовленого виробу
Матеріал - замовлений матеріал
Кількість - замовлене кількість

```

[Вказівки до завдання 1.1](#)

Перевірка даних відразу після введення

Якщо в формі потрібно заповнити багато полів, то користувачеві зручно отримувати повідомлення про помилки відразу після закінчення введення даних в чергове поле, тобто після натискання клавіші `Tab` або клавіші зі стрілкою. Для негайної перевірки введених даних служить подія `onchange`:

```

<INPUT TYPE = "text" SIZE = 6 onchange = "arg (this)">

```

Функція, що викликається подією onchange, має приблизно таку структуру:

```
function arg (fld)
{X = fld.value // введене значення
  if (x...) // умови перевірки
  {Alert ( "Повідомлення про помилку");
    fld.focus ();
    fld.select ()
  }
}
```

Методи focus () і select () служать для повернення курсору мишки в поле введення і виділення помилкових даних. Ці методи без використання спеціальних прийомів правильно працюють тільки в браузері Mozilla.

Завдання 1.2. Додати сторінку (рис. 1) для обчислення тригонометричних функцій. Введені користувачем дані повинні перевірятися негайно після введення і після натискання кнопки Обчислити.

ТРИГОНОМЕТРИЧЕСКИЕ ФУНКЦИИ

Угол должен быть больше нуля и меньше 90 градусов

Угол в градусах

Функция

синус

косинус

тангенс

$\sin(30^\circ) = 0.5$

Рис.1

Вказівки. Не забудьте перевести градуси в радіани. Назва тригонометричної функції можна передавати як параметр тега:

```
<Option value = "sin". . . >
```

Сформуйте текстовий рядок виду

"Math." + Імя_ф + "(" + знач_аргумента + ")" // імя_ф - sin, cos або tan

Потім скористайтеся функцією eval (рядок), яка виконує вираз, що зберігається в рядку.