

Лабораторна робота № 6

Тема: Застосування масивів для створення меню та форм засобами JavaScript.

Мета: Навчити працювати з визначеними об'єктами: Date, Array, Math.

Теоретичні відомості:

В JavaScript дозволено користуватися деякими заздалегідь заданими об'єктами. Прикладами таких об'єктів можуть служити Date, Array або Math. Об'єкт Date дозволяє працювати як з часом, так і з датою. Наприклад, можна легко визначити, скільки днів ще залишається до наступного Різдва. Або можете внести в HTML-документ запис поточного часу.

Розглянемо приклад, який висвічує на екран поточний час. Спершу ми повинні створити новий об'єкт Date. Для цього використовуємо оператором new:

```
today = new Date ()
```

Тут створюється новий об'єкт Date, з ім'ям today. Якщо при створенні цього нового об'єкта Date не вказані будь-які певні дата і час, то будуть надані поточні дата і час. Тобто, після виконання команди today = new Date () новостворений об'єкт today буде вказувати саме ті дату і час, коли дана команда була виконана.

Об'єкт Date надає нам деякі методи, які тепер можуть застосовуватися до нашого об'єкту today. Наприклад, є методи - getHours (), setHours (), getMinutes (), setMinutes (), getMonth (), setMonth () ітакдалее. Зверніть будь ласка увагу, що об'єкт Date лише містить певну запис про дату і час. Він не уподібнюється годинах, автоматично відстежує час кожен секунду, або мілісекунди. Щоб зафіксувати якесь інші дату і час, ми можемо скористатися видозміненим конструктором (це буде метод Date (), який при створенні нового об'єкта Date викликається через оператор new):

```
today = new Date (1997, 0, 1, 17, 35, 23)
```

При цьому буде створено об'єкт Date, в якому буде зафіксовано перше січня 1997 року 17:35 і 23 секунд. Таким чином, вибираються дата і час за таким шаблоном:

```
Date (year, month, day, hours, minutes, seconds)
```

Для позначення січня необхідно використовувати число 0, а не 1. Число 1 означатиме лютий, ну і так далі.

Масиви грають в програмуванні дуже важливу роль. Масив може бути корисний там, де є багато взаємопов'язаних змінних. При цьому до кожної з них можна отримати доступ, скориставшись загальною назвою і якимсь номером. Припустимо, є масив в іменем names. У цьому випадку ми можемо отримати доступ до першої змінної з ім'ям name, написавши names [0]. Друга змінна має name [1] і так далі. Починаючи з версії 1.1 мови JavaScript (Netscape Навігатор 3.0), можна використовувати об'єкт Array. Можна створити новий масив, записавши myArray = new Array (). Після цього можна почати заносити в масив значення:

```
myArray [0] = 17;  
myArray [1] = "Stefan";  
myArray [2] = "Koch";
```

Масиви JavaScript мають велику гнучкість. Наприклад, немає потреби турбуватися про розмір масиву - він встановлюється динамічно. Якщо написати myArray [99] = "xyz", розмір масиву буде встановлено 100 елементів.

Не має значення, заносяться чи в масив числа, рядки, або інші об'єкти. Якщо необхідно в скрипті виконувати математичні розрахунки, то деякі корисні методи для цього можна знайти в об'єкті Math. Наприклад, є метод random (). Напишемо функцію, що дозволяє генерувати випадкові числа. Тепер, щоб працювати на всіх без винятку платформах, нам не потрібно нічого, крім методу random ().

Якщо викликати функцію Math.random (), то отримаєте випадкове число, яке лежить в діапазоні між 0 і 1. Один з можливих результатів виклику document.write (Math.random ()) (при кожній нове завантаженні даної сторінки тут буде з'являтися інше число) :

```
0.9070647660301312
```

Практична частина

Завдання 1.

Створіть документ, що друкує поточну дату і час.

```
<Script language = "JavaScript">
<!-- hide
now = new Date ();
document.write ( "Time:" + now.getHours () + ":" + now.getMinutes () + "<br>");
document.write ( "Date:" + (now.getMonth () + 1) + "/" + now.getDate () + "/" +
(now.getFullYear ());
// -->
</ Script>
```

Результат буде виглядати приблизно так:

Time: 20:40

Date: 10/24/2009

Тут використовуються такі методи, як `getHours ()`, щоб вивести на екран час і дату, зазначені в об'єкті `Date` з ім'ям `now`. Пам'ятайте також, що потрібно збільшувати на одиницю значення, що отримується від методу `getMonth ()`.

В даному скрипті не виконується перевірки на той випадок, якщо кількість хвилин виявиться менше, ніж 10. Це означає, що виводиться запис часу приблизно в наступному вигляді: 14: 3, що насправді мало б означати 14:03.

Завдання 2.

1 Створіть документ, що створює на екрані зображення працюючого годинника.

```
<html>
<head>
<script Language="JavaScript">
<!-- hide
var timeStr, dateStr;
function clock() {
    now= new Date();
// час
    hours= now.getHours();
    minutes= now.getMinutes();
    seconds= now.getSeconds();
    timeStr= "" + hours;
    timeStr+= ((minutes < 10) ? ":0" : ":") + minutes;
    timeStr+= ((seconds < 10) ? ":0" : ":") + seconds;
    document.clock.time.value = timeStr;
// дата
    date= now.getDate();
    month= now.getMonth()+1;
    year= now.getFullYear();
    dateStr= "" + month;
    dateStr+= ((date < 10) ? "/0" : "/") + date;
    dateStr+= "/" + year;
    document.clock.date.value = dateStr;
    Timer= setTimeout("clock()",1000);}
// -->
</script>
</head>
<body onLoad="clock()">
<form name="clock">
Час:
```

```
<input type="text" name="time" size="8" value=""><br>
Дата:
<input type="text" name="date" size="8" value="">
</form>
</body>
</html>
```

2 Перегляньте результат.

Приклад:

Час:

Дата:

Тут для щосекундної корекції часу та дати використовується метод `setTimeout ()`. Фактично це зводиться до того, що кожен секунду створюється новий об'єкт `Date`, в який заноситься поточний час.

Можна бачити, що функції `clock ()` викликаються програмою обробки події `onLoad`, вміщений в тег `<body>`. У розділі `body` HTML-сторінки є два елементи форми для введення тексту. Функція `clock ()` записує в обидва ці елементи в коректному форматі поточні час і дату. Для цієї мети використовуються два рядки `timeStr` і `dateStr`. Існує проблема з індикацією, коли кількість хвилин менше 10 - в даному скрипті ця проблема вирішується за допомогою наступного рядка: `timeStr += ((minutes <10)? ": 0": ":") + minutes;`

Як бачимо, кількість хвилин заноситься в рядок `timeStr`. Якщо у нас менше 10 хвилин, то ми ще повинні приписати спереду 0. Цей рядок в скрипті може здатися трохи дивною, і її можна було б переписати в більш знайомому Вам вигляді:

```
if (minutes <10) timeStr += ": 0" + minutes
else timeStr += ":" + minutes;
```

Завдання 3.

1. Створіть документ, який використовує масив:

```
<Scriptlanguage = "JavaScript">
<! - hide
myArray = new Array ();
myArray [0] = "first element";
myArray [1] = "second element";
myArray [2] = "third element";
for (var i = 0; i <3; i ++) {
    document.write (myArray [i] + "<br>");}
// ->
</ Script>
```

Даний скрипт друкує наступний текст:

first element
second element
third element

Насамперед створюється тут новий масив з ім'ям `myArray`. Потім заносяться в нього три різних значення. Після цього ми запускаємо цикл, який тричі виконує команду `document.write (myArray [i] + "
");`. В змінній `i` ведеться відлік циклів від 0 до 2. Зауважимо, що в циклі ми користуємося конструкцією `myArray [i]`. І оскільки `i` змінює значення від 0 до 2, то в результаті ми отримуємо три різних виклику `document.write ()`. Іншими словами, ми могли б розписати цей цикл як:

```
document.write (myArray [0] + "<br>");
```

```
document.write (myArray [1] + "<br>");  
document.write (myArray [2] + "<br>");
```

Питання для самоконтролю:

- 1 Прикладами яких об'єктів служать об'єкти Date, Array, Math?
- 2 Для чого призначений об'єкт Date?
- 3 Що називають масивом?
- 4 Для чого призначені масиви?
- 5 Яким чином здійснюється доступ до масиву?
- 6 Яким чином проводиться установка розміру масиву?
- 7 Для чого призначений об'єкт Math?
- 8 Перерахуйте методи об'єкта Date.
- 9 Яким чином проводиться створення об'єктів Date, Array, Math?
- 10 Який об'єкт дозволяє працювати з часом?