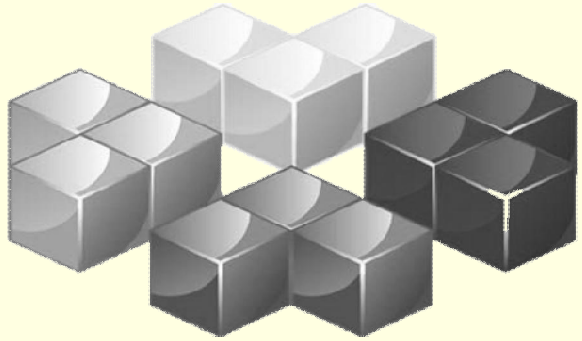
Мова програмування JavaScript



Створення вікон

Динамічне створення документів

Зображення на web-сторінці

Створення вікон

Відкриття нових вікон у браузері - грандіозна можливість мови JavaScript. Ви можете або завантажувати в нове вікно нові документи (наприклад, ті ж документи HTML), або (динамічно) *створювати* нові матеріали. Подивимося спочатку, як можна відкрити нове вікно, потім як завантажити в це вікно HTML-сторінку й, нарешті, як його закрити.

Наведений далі скрипт відкриває нове вікно браузера й завантажує в нього вказану web-сторінку:

```
<html>
<head>
<script language="JavaScript">
function openWin() {
  myWin= open("page4.htm");
}
</script>
</head>
<body>
<form>
<input type="button" value="Відкрити нове вікно" onClick="openWin()">
</form>
</body>
</html>
```

У представленому прикладі в нове вікно за допомогою методу open() записується сторінка *page4.htm*.

Список властивостей вікна

directories	yes|no	Дозволяє вказувати, чи відображається панель кнопок для вибору каталогів.
height	кількість пікселів	Задає висоту вікна в пікселях. Мінімальне значення -100 .
location	yes|no	Дозволяє вказувати, чи відображається рядок для введення адреси.
menubar	yes|no	Дозволяє вказувати, чи відображається панель меню.
resizable	yes|no	Дозволяє вказувати, чи може вікно міняти свій розмір.
scrollbars	yes|no	Задає відображення горизонтальної й вертикальної смуг прокручування.
status	yes|no	Дозволяє вказувати, чи відображається рядок стану.
toolbar	yes|no	Дозволяє вказувати, чи відображається панель інструментів.
width	кількість пікселів	Задає ширину вікна в пікселях. Мінімальне значення -100 .
fullscreen	yes|no	Указує, чи показується нове вікно на повний екран або як звичайне вікно. За замовчуванням показується звичайне вікно.
channelmode	yes|no	Дозволяє вказати, чи відображається меню каналів.
top	число	Задає вертикальну координату лівого верхнього кута.
left	число	Задає горизонтальну координату лівого верхнього кута.

Ім'я вікна

Відкриваючи вікна, ми використовуємо три аргументи:

```
myWin= open ("page4.htm", "displayWindow",
"width=400,height=300,status=no,toolbar=no,menubar=no")
```

Другий аргумент - це ім'я вікна. Раніше ми бачили, як воно використовувалося в параметрі target. Так, якщо ви знаєте ім'я вікна, то можете завантажити туди нову сторінку за допомогою запису:

```
<a href="page4.html" target="displayWindow">
```

При цьому вам необхідно вказати ім'я відповідного вікна (якщо ж такого вікна не існує, то із цим ім'ям буде створене нове).

Зверніть увагу, що *myWin* - це зовсім не ім'я вікна. Але тільки за допомогою цієї змінної ви можете одержати доступ до вікна. І оскільки це звичайна змінна, то область її дії - лише той скрипт, у якому вона визначена. А тим часом, ім'я вікна (у цьому випадку це *displayWindow*) - унікальний ідентифікатор, яким можна користуватися з *кожного* з вікон браузера.

Закриття вікон

Ви можете закривати вікна за допомогою мови JavaScript. Щоб зробити це, вам знадобиться метод close(). Розглянемо на наступному прикладі відкриття нового вікна та завантажимо туди чергову сторінку:

```
<html>
<script language="JavaScript">
function closeIt() {
  close();
}
</script>
<center>
<form>
<input type=button value="Закрити вікно" onClick="closeIt()">
</form>
</center>
</html>
```

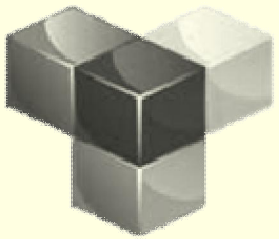
Якщо тепер у новому вікні ви натиснете кнопку, то воно буде закрито.

Open() і close() - це методи об'єкта window. Необхідно пам'ятати, що варто писати не просто open() і close(), а window.open() і window.close(). Однак у нашому випадку об'єкт window можна опустити - вам немає необхідності писати префікс window, якщо ви хочете всього лише викликати один з методів цього об'єкта (і таке можливо тільки для цього об'єкта).

Динамічне створення документів

Що ж таке динамічне створення документів? Тобто ви можете дозволити вашому скрипту мовою JavaScript самому створювати нові HTML-сторінки. Більше того, ви можете в такий же спосіб створювати й інші документи Web, такі як VRML-сцени й т.д. Для зручності ви можете розміщати ці документи в окремому вікні або фреймі.

Розглянемо приклад: для початку створимо простий HTML-документ, що покажемо в новому вікні. Запишемо наступний скрипт.



```
<html>
<head>
<script language="JavaScript">
function openWin4() {
  myWin= open("", "displayWindow",
    "width=500,height=400,status=yes,toolbar=yes,menubar=yes")
  // відкрити об'єкт document для наступного друку
  myWin.document.open()
  // генерувати новий документ
  myWin.document.write("<html><head><title>Динамічне створення HTML-документа")
  myWin.document.write("</title><META HTTP-EQUIV='Content-Type' CONTENT='text/html;")
  myWin.document.write("charset=windows-1251'></head><body background='bg.gif'>")
  myWin.document.write("<center><font size=+3>")
  myWin.document.write("Цей HTML-документ створено за допомогою ");
  myWin.document.write("JavaScript!");
  myWin.document.write("</font></center>")
  myWin.document.write("</body></html>")
  // закрити документ - (але не вікно!)
  myWin.document.close();
}
</script>
</head>
<body>
<form>
<input type=button value="Динамічне створення HTML-документа" onClick="openWin4()">
</form>
</body>
</html>
```

Текст скрипту

Пояснення

Розглянемо функцію **openWin4().** Очевидно, ми спочатку відкриваємо нове вікно браузера. Оскільки перший аргумент функції open() - порожній рядок (""), а це значить, що ми не бажаємо в цьому випадку вказувати конкретну адресу URL. Браузер повинен не тільки обробити наявний документ - JavaScript зобов'язаний створити додатково новий документ.

У скрипті визначаємо змінну *myWin*. І з її допомогою можемо одержувати доступ до нового вікна. Зверніть будь ласка увагу, що в цьому випадку ми не можемо скористатися для цієї мети ім'ям вікна (*displayWindow*).

Після того, як ми відкрили вікно, наступає черга відкрити для запису об'єкт document. Робиться це за допомогою команди:

```
// відкрити об'єкт document для наступного друку
myWin.document.open()
```

У вище наведеному скрипті ми звертаємося до open() - методу об'єкта *document*. Однак це зовсім не те ж саме, що метод open() об'єкта *window*! Ця команда не відкриває нового вікна - вона лише готовить document до майбутнього друку. Крім того, ми повинні поставити перед *document.open()* приставку *myWin*, щоб одержати можливість писати в новому вікні.
У наступних рядках скрипту за допомогою виклику *document.write()* формується текст нового документа:

```
// генерувати новий документ
myWin.document.write("<html><head><title>Dinamic HTML")
myWin.document.write("</title></head><body background='bg.gif'>")
myWin.document.write("<center><font size=+3>")
myWin.document.write("This HTML-document has been created ");
myWin.document.write("with the help of JavaScript!");
myWin.document.write("</font></center>")
myWin.document.write("</body></html>")
```

Як видно, тут записані в документ звичайні теги мови HTML. При цьому ви можете використати абсолютно будь-які теги HTML.

По завершенні цього ми зобов'язані знову закрити документ. Це робиться за допомогою наступної команди:

```
// закрити документ - (але не вікно!)
myWin.document.close()
```

Як уже згадувалося раніше, ви можете не тільки динамічно створювати документи, але й за своїм вибором розміщувати їх у тому чи іншому фреймі. Наприклад, якщо ви одержали два фрейми з іменами *frame1* і *frame2*, а тепер в *frame2* хочете згенерувати новий документ, то для цього в *frame1* вам досить буде написати наступне:

```
parent.frame2.document.open();
parent.frame2.document.write("Тут буде розташовуватися Ваш HTML-код");
parent.frame2.document.close();
```

Зображення на web-сторінці

Розглянемо тепер об'єкт Image. За допомогою об'єкта Image ви можете вносити зміни в графічні об'єкти, що знаходяться на web-сторінці.

Завантаження нових зображень

Для зміни зображень на web-сторінці нам знадобиться атрибут *src*. Як і у випадку тегу <img>, атрибут *src* містить адресу обраного зображення. Тепер - у мові JavaScript - ви маєте можливість призначати нову адресу зображенню, уже завантаженому на web-сторінку. І в результаті, зображення буде завантажено із цієї нової адреси, замінивши на web-сторінці старе. Розглянемо наприклад запис:

```
<img src="img1.gif" name="myImage" width=100 height=100>
```

Тут завантажується зображення img1.gif і одержує ім'я myImage. У наступному рядку колишнє зображення img1.gif заміняється вже на нове - img2.gif:

```
document.myImage.src= "img2.src";
```

При цьому нове зображення завжди одержує той же розмір, що був у старого. І ви вже не можете змінити розмір поля, у якому це зображення розміщається.

Упереджуюче завантаження зображення

Один з недоліків такого підходу може полягати у тому, що після запису в *src* нової адреси починається процес завантаження відповідного зображення. І оскільки цього не було зроблено заздалегідь, то ще пройде якийсь час, перш ніж нове зображення буде передане через Інтернет і вставлене на своє місце. У деяких ситуаціях це припустимо, однак часто подібні затримки неприйнятні. Що ж можна з цим зробити? Звичайно, рішенням проблеми було б упереджуюче завантаження зображення. Для цього необхідно створити новий об'єкт Image. Розглянемо наступні рядки:

```
hiddenImg= new Image();
hiddenImg.src= "img3.gif";
```

У першому рядку створюється новий об'єкт Image. У другому рядку вказується адреса зображення, що надалі буде представлено за допомогою об'єкта hiddenImg. Як ми вже бачили, запис нової адреси в атрибуті src змушує браузер завантажувати зображення із зазначеної адреси. Тому, коли виконується другий рядок нашого прикладу, починає завантажуватися зображення img2.gif. Але як мається на увазі самою назвою hiddenImg ("схована картинка"), після того, як браузер закінчить завантаження, зображення на екрані не з'явиться.

Зображення лише буде збережено в пам'яті комп'ютера (або точніше в кеші) для наступного використання. Щоб викликати зображення на екран, ми можемо скористатися рядком:

```
document.myImage.src= hiddenImg.src;
```

Але тепер зображення вже негайно показується на екрані. Таким чином, зараз ми керували упереджуючим завантаженням зображення.

Зміна зображень у зв'язку з подіями, що ініціюються користувачем

Ви можете створити гарні ефекти, використовуючи зміну зображень як реакцію на певні події. Наприклад, ви можете змінювати зображення у той момент, коли курсор миші попадає на певну частину сторінки. Перевірте, як працює наступний приклад, просто помістивши курсор миші на картинку.

Вихідний код цього прикладу виглядає у такий спосіб:

```
<a href="#"
onMouseOver="document.myImage2.src='img2.gif'"
onMouseOut="document.myImage2.src='img1.gif'">
<img src="img1.gif" name="myImage2" width=160 height=50 border=0></a>
```

Завдання

Питання для самоконтролю:

1. Назвіть основні властивості вікна?
2. Назвіть основні принципи динамічного створення документів.
3. Для чого використовується упереджуюче завантаження зображення?

Завдання на лабораторну роботу

1. Вивчити теоретичний матеріал.
2. Створити Web-сторінку з використанням вивченого матеріалу.
3. Захистити роботу викладачеві.