Лабораторна робота №8
«Обробка подій в JS»

Мета роботи - навчитися використовувати наявні в моделі документа події для внесення змін в сторінку.

Найбільш часто в сценаріях використовується розглянуте вище подія onclick. Для того щоб звернути увагу користувача на певний елемент HTML-документа, можна змінювати властивості цього елемент при попаданні на нього курсора мишки, а при знятті курсора відновлювати колишні значення властивостей. Наприклад, можна змінювати колір або розмір елемента. Попадання курсора мишки на елемент фіксується подією onMouseOver. Парне для нього подія onMouseOut відбувається при знятті курсора мишки з елемента.

Інша пара подій onMouseDown і OnMouseUp відбувається при натисканні і відпусканні лівої кнопки мишки. Цю пару подій зручно застосовувати для зміни властивостей елементів або заміни елементів на час утримання кнопки мишки, утримуючи.

Реакція на подію в окремому елементі

Так як в об'єктної моделі об'єкти можуть бути вкладені одна в одну, то подія, що відбувається в дочірньому об'єкті, одночасно відбувається і в батьківському об'єкті. JavaScript надає різні способи локалізації впливу події на ієрархію об'єктів. Найпростішою спосіб локалізації (приклад 2.1) полягає в розміщенні сценарію в тезі, на який має впливати подія.

Приклад 2.1.

```
<HTML>
<BODY>
<P align = right ID = 'alfa'
onMouseOver = "document.all.alfa.align = 'center'
"OnMouseOut =" this.align = 'left' ">
Події onMouseOver і onMouseOut </ p>
</ Body>
</ Html>
```

Сторінка в прикладі 2.1 складається з одного рядка, вкладеного в контейнер <P> ... </ P>. В об'єктній моделі сторінки подія, що відбувається з об'єктом P, відбувається також і з батьківським об'єктом BODY. Щоб локалізувати реакцію на подію тільки межами рядки, тобто об'єкта P, сценарій реакції на події поміщений в тег <P>.

В результаті виконання сценарію змінюється положення тексту на рядку. Спочатку рядок притиснута до правого краю вікна. При попаданні на нього курсору вона вирівнюється по центру, а після зняття курсора притискається до лівого краю вікна. Для звернення до об'єкту використовується колекція all, яка правильно сприймається браузерами Internet Explorer 6.0 і Mozilla Firefox 2.0. Ключове слово this означає посилання на поточний об'єкт.

Якщо при настанні події потрібно зробити багато дій, то зручно сценарій написати у вигляді функції і помістити її окремо від елемента в спеціально призначений для сценаріїв контейнер <SCRIPT> ... </SCRIPT>. У прикладі 2.2 кожне з подій onMouseOver і onMouseOut викликає два дії: вирівнювання і зміна кольору тексту в рядку.

Приклад 2.2

```
<HTML>
<P align = right ID = 'alfa' onMouseOver = "M_Over ()"
onMouseOut = "M_Out ()">
```

```
Подія onMouseOver </ p>
<SCRIPT>
function M_Over ()
{Document.all.alfa.align = 'center'
 document.all.alfa.style.color = 'FF00FF'
}
function M_Out ()
{Document.all.alfa.align = 'left'
 document.all.alfa.style.color = '0000FF'
}
</ Script>
</ Html>
```

Завдання 2.1. Напишіть HTML-документ, що відображається у вікні браузера у вигляді наступних чотирьох рядків:

- П'ять подій з мишкою
- Клацніть на мене мишкою
- На цьому тексті натисніть, потримайте і відпустіть ліву кнопку мишки
- Повільно проведіть курсором мишки по цій написи

Перший рядок - заголовок сторінки. Другий рядок змінюється при натисканні мишкою наступним чином:

- шрифт збільшується до 48pt;
- колір шрифту змінюється на білий;
- колір фону змінюється на блакитний.

Повторне клацання мишкою повертає другий рядок до первісного вигляду.

Фон третього рядка змінюється, коли курсор мишки знаходиться на ній і натискається або відпускається ліва кнопка мишки. При натисканні фон стає зеленим, а при відпускунні - жовтим.

При попаданні курсора мишки на четверту рядок її фон стає червоним, а при знятті - блакитним.

Фіксація події в батьківському елементі

Якщо реакцію на будь-яку подію вимагають кілька елементів, розташованих на сторінці, то можна викликати функцію для обробки цієї події тільки в батьківському елементі. У функції визначається, на якому елементі відбулася подія, і виконуються відповідні дії. Зручність такого підходу полягає в тому, що весь алгоритм перетворень знаходиться в одному місці, а недолік - в складності самої функції. Розглянемо сценарій (приклад 2.3), в якому для зміни властивостей будь-якого з трьох об'єктів, що знаходяться у вікні браузера служить одна функція.

Приклад 2.3

```
<HTML> <HEAD> <TITLE> Реакція на подію </ TITLE>
<Meta http-equiv = "Content-Type" content = "text / html; charset = windows-1251">
<STYLE> H1 {color: FF00FF}
# K1 {position: absolute; left: 50; top: 200; width: 300; height: 100; background-color: blue}
 # K2 {position: relative; left: 50; top: 25; width: 200; height: 50; background-color: yellow}
```

```
</ STYLE>
</ HEAD>
<BODY ID = "B" onclick = "rodEl (event)">
 <H1 ID = "HH"> КОЛІР </ H1>
 <DIV ID = "k1">
  <DIV ID = "k2"> </ div>
 </ Div>
<SCRIPT>
/ * Функція запускається при натисканні мишкою по будь-якій точці документа * /
flag = 0;
function rodEl (evt)
{Var e = evt || window.event; // е-це об'єкт event
  var elem = e.target || e.srcElement; // elem - елемент (об'єкт),
                          // на якому відбулася подія
  id1 = elem.id
  z1 = document.getElementById (id1)
 switch (id1)
 {Case "k1": // Зміна кольору зовнішнього прямокутника
    z = z1.style.backgroundColor
    if (z! = "red") z = "red"
    else z = "green"
    z1.style.backgroundColor = z
    break
  case "k2": // Зміна кольору внутрішнього прямокутника
    z = z1.style.backgroundColor
    if (z! = 'rgb (0, 255, 255)') {z = 'rgb (0, 255, 255)'}
    else {z = 'rgb (0, 255, 0)'}
    z1.style.backgroundColor = z
    break
  case "B": // Зміна кольору фону документа
    z = z1.style.backgroundColor
    if (z! = 'rgb (190, 190, 190)') {z = 'rgb (190, 190, 190)'}
    else {z = 'rgb (0, 190, 190)'}
    z1.style.backgroundColor = z
    break
  case "HH": // Зміна кольору слова "Колір"
    if (flag == 0)
    {Document.getElementById (id1) .style.color = 'rgb (170,0,170)';
      flag = 1;
    }
    else
    {Document.getElementById (id1) .style.color = 'rgb (0,255,170)';
      flag = 0;
```

```
    }
  }
}
</ SCRIPT>
</ BODY>
</ HTML>
```

У прикладі 2.3 батьківським по відношенню до елементу H1 і двом елементам DIV є елемент BODY. Тому в тезі <BODY> викликається функція rodEl (), що служить для обробки події onclick.

У момент настання події вся інформація про нього запам'ятовується в об'єкті event. Цей об'єкт по різному описується в стандарті W3C і в браузері Internet Explorer. Нові версії Internet Explorer підтримують і стандарт W3C.

У стандарті W3C об'єкт event передається в функцію як параметр, а для звернення до об'єкта, на якому відбулася подія, служить властивість event.target.

В Internet Explorer об'єкт window.event - глобальний і тому передавати його в функцію у вигляді параметра не потрібно. Для звернення до об'єкту, на якому відбулася подія, в Internet Explorer служить властивість window.event.srcElement

У прикладі 2.3 перші два рядки тіла функції rodEl () служать для кросплатформного (в будь-якому браузері) звернення до об'єкту, на якому відбулася подія. У в наступних рядках функції rodEl () спочатку визначається Id елемента, за яким користувач клацнув мишкою, а потім за допомогою оператора Switch робиться перехід до зміни властивостей зазначеного елемента.

Завдання 2.2. Додати сторінку із зображенням і підписом під ним. При натисканні на підписи, вона повинна змінювати свій колір. Клацання по зображенню повинно викликати заміну зображення і підписи. Функція для обробки події повинна викликатися з батьківського по відношенню до зображення і підпису об'єкта.

Запобігання спливання події. властивість cancelBubble

У прикладі 2.3 розглядалася проста сторінка з невеликою кількістю елементів, але навіть в такому простому випадку функція реакції на подію вийшла складною. Простіше для кожного елемента написати свою функцію обробки події, а поширення події вгору по дереву ієрархічної структури від "дітей" до "батькам" (в цьому випадку говорять про Спливання події) блокувати з допомогу спеціально для цього призначеної властивості cancelBubble об'єкта event. Змінимо приклад 2.3 так, щоб для реакції на клацання по кожному з чотирьох елементів сторінки служила своя функція:

Приклад 2.4

```
<HTML> <HEAD> <TITLE> Реакція на подію </ TITLE>
<Meta http-equiv = "Content-Type" content = "text / html; charset = windows-1251">
<STYLE> H1 {color: # ff00ff}
# K1 {position: absolute; left: 50; top: 200;
    width: 300; height: 100; background-color: blue}
# K2 {position: relative; left: 50; top: 25;
     width: 200; height: 50; background-color: yellow}
</ STYLE> </ HEAD>
<BODY ID = "B" bgcolor = "AAAAAA" onclick = "rodEl ()" style = "height: 600px">
<H1 ID = "HH" onclick = "H_l ()"> КОЛІР </ H1>
```

```
<DIV ID = "k1" onclick = "D_l ()">
<DIV ID = "k2" style = "background-color: yellow" onclick = "D_2 (this)"> </
div>
</ Div>
</ BODY>
<SCRIPT>
/ * Функція запускається при натисканні мишкою по
будь-якій точці документа * /
function rodEl ()
{// Зміна кольору фону документа
var z = document.all.B.bgColor
if (z! = "# 777777") {z = "# 777777"}
else {z = "AAAAAA"}
   document.all.B.bgColor = z
}
function D_l ()
 {// Зміна кольору зовнішнього прямокутника

  var z = document.all.k1.style.backgroundColor
if (z! = "red") z = "red"
else z = "green"
document.all.k1.style.backgroundColor = z
}
/ * Function D_2 ()
{// Зміна кольору внутрішнього прямокутника
   var z = document.all.k2.style.backgroundColor
 if (z == "# ff00ff" || z == 'rgb (255, 0, 255)') {z = "# 00ffff"}
else {z = "ff00ff"}
document.all.k2.style.backgroundColor = z
} * /
function D_2 (thi)
{// Зміна кольору внутрішнього прямокутника
   var z = thi.style.backgroundColor
 if (z == "# ff00ff" || z == 'rgb (255, 0, 255)') {z = "# 00ffff"}
else {z = "ff00ff"}
thi.style.backgroundColor = z
}
function H_l ()
{// Зміна кольору слова "Колір"
   var z = document.all.HH.style.color
   // alert ( "1. Z =" + document.all.HH.style.color)
   if (z == "# aa00aa" || z == 'rgb (170, 0, 170)')
{Z = "# 00ffff"
// alert ( "2. Z =" + z)
```

```
}
else {z = "# aa00aa"}
document.all.HH.style.color = z
}
</ SCRIPT> </ HTML>
```

Скопіюйте в свій каталог і перегляньте приклад 2.4 в браузері Internet Explorer. Клацніть по слову КОЛІР. Зміниться не тільки колір слова, але і колір фону документа, так як після клацання спочатку виконається функція H_1 (), а потім подія спливе до батьківського елементу BODY і виконається функція rodE1 (). При натисканні по внутрішньому прямокутнику будуть змінюватися кольори обох прямокутників і фону документа. Клацання по зовнішньому прямокутнику змінить колір цього прямокутника і колір фону документа.

Завдання 2.3. Щоб запобігти спливання події в прикладі 2.4, вставте в початок всіх функцій, крім першої, оператор

```
window.event.cancelBubble = true
```