Лабораторная работа №4
«Создание движущихся объектов»

Цель работы:

1. Изучение возможностей языка JavaScript по созданию на Web-странице движущихся объектов.

2. Получение практических навыков по созданию движущихся объектов с разными свойствами.

Оборудование и ПО: ПК, операционная система Windows7, офисный пакет Microsoft.

Порядок выполнения работы:

1. Ознакомление с методическими рекомендациями по созданию на Web-странице движущихся объектов с разными свойствами. *Просмотр обучающих роликов к лабораторной работе № 4.*

2. Выполнение практической части задания в соответствии с рекомендациями разделов методического пособия.

3. Ответы на контрольные вопросы.

4. Оформление отчета.

I. Теоретическая часть и рекомендации

Движение элемента на Web-странице осуществляется путем изменения значений свойств, задающих его координаты.

Изменение координат элемента может быть реализовано:

– путем программно организованного циклического процесса;

– по событиям мышки;

– путем "привязки" элемента к курсору мышки, т.е. путем задания координат элемента равными координатам курсора мышки;

– по событиям клавиатуры.

Рассмотрим первые три пункта, а четвертый будет рассмотрен в следующих статьях.

1.1 Изменение координат элемента с помощью циклического процесса

Для организации циклического процесса могут быть использованы следующие методы объекта window:

- setTimeout("функция или выражение", интервал [,список аргументов функции, разделенных запятыми]): вычисляет значение выражения или вызывает функцию по истечению заданного интервала (в миллисекундах), если до этого не был вызван метод clearTimeout(), может передавать в функцию заданные в списке аргументы, возвращает указатель на объект таймера, который может быть использован в методе clearTimeout() для остановки и уничтожения таймера;

- clearTimeout(таймер): останавливает таймер, установленный методом setTimeout();

- setInterval("функция или выражение", интервал [,список аргументов функции, разделенных запятыми]): вычисляет значение выражения или вызывает функцию каждый раз по истечению данного интервала (в миллисекундах), если до этого не был вызван метод clearTimeout(), может передавать в функцию заданные в списке аргументы, возвращает указатель на объект таймера, который может быть использован в методе clearTimeout() для остановки и уничтожения таймера;

- clearInterval(таймер): останавливает таймер, установленный методом setInterval().

Использование метода setTimeout() для организации движения элемента показано в примере № 1, где реализовано движение элемента Web-страницы - буквы Z, заданной с помощью тэга <P>, по траектории в виде синусоиды ():

Пример 1

```
<HTML>
<HEAD>
<TITLE>Изменение координат элемента с помощью циклического процесса</TITLE>
<SCRIPT>
x=0; y=100; dx=0; dy=0;
function f(x) { return 60*Math.sin(x*Math.PI/180);}
function moveTxt()
```

```
{
if ((z.style.pixelLeft<document.body.clientWidth-30) && (z.style.pixelTop>30))
{
z.style.left=x+dx;
z.style.top=y+dy-f(x+dx);
dx+=20;
dy+=7;
setTimeout("moveTxt()",125);
}
}
</SCRIPT>
</HEAD>
<BODY>
<P ID="z" STYLE="color:blue;font:8mm; position:absolute;left:0;top:100"
onClick="moveTxt()">Z</P>
</BODY>
</HTML>
```

В **примере 1** с помощью функции f(x) реализовано движение объекта справа налево и сверху вниз по синусоиде y=60sin(x). Поскольку траектория указана в виде тригометрической функции, для ее задания был использован объект Math, свойства и методы которого описаны в разделе 3. При этом было выполнено преобразование аргумента функции sin(x) из градусов в радианы. Направление и скорость движения объекта зависит от величины и знака приращений его координат dx и dy. На скорость движения также влияет промежуток времени, через который эти приращения происходят (2-й параметр метода setTimeout()).

Выбор амплитуды и скорости движения в **примере 1** осуществлен из соображений наглядности.

По щелчку мышки по букве Z вызывается функция moveTxt(), которая осуществляет движение этого элемента из точки окна браузера с координатами: left=0; top=100 путем задания приращений его координатам left и top:

```
z.style.left=x+dx;
z.style.top=y+dy-f(x+dx);
```

до тех пор, пока он не приблизится к границам окна браузера (граница не достигается, чтобы буква Z оставалась в области полной видимости).

Значения левой и нижней границы окна браузера определяются с помощью свойств тэга <BODY> clientWidth и clientHeight. Эти свойства могут быть использованы для определения размеров любых элементов Web-страницы, которые имеют ширину и высоту.

При использовании альтернативных методов setInterval() и clearInterval() описание функции moveTxt() необходимо несколько изменить:

```
function moveTxt()
{
if ((z.style.pixelLeft<document.body.clientWidth-30) && (z.style.pixelTop>30))
{
z.style.left=x+dx;
z.style.top=y+dy-f(x+dx);
dx+=20;
dy+=7;
}
else clearInterval(timer);
}
```

Кроме того, вызов функции moveTxt() в этом случае следует выполнять так:

```
onClick="timer=setInterval('moveTxt()',125);"
```

1.2. Изменение координат элемента по событиям мышки

Движения объектов Web-страницы по экрану кроме рассмотренного выше способа изменения его координат с помощью программно организованного циклического процесса, может осуществить сам пользователь Web-сайта с помощью мышки.

В примере № 2 описана функция runRef(), которая каждый раз при достижении указателем мышки ссылки Ссылка или при движении указателя мышки по ссылке выполняет приращения координат этого элемента, реализуя эффект "убегающей" от пользователя ссылки. Все его попытки щелкнуть по ссылке, чтобы загрузить новую страницу (js_1.htm), оказываются безрезультатными.

С помощью переменных dx и dy, которые меняют свой знак на противоположный при достижении соответственно вертикальных или горизонтальных границ окна браузера, реализовано отражение ссылки при движении по экрану от его границ.

Вызов функции runRef() как по событию Mousemove, так и по событию Mouseover обеспечивает более надежную защиту ссылки от щелчка мышки.

Пример 2

```
<HTML>
<HEAD>
<TITLE>Организация движения элементов Web-страницы с помощью
мышки </TITLE>
<SCRIPT>
dx=10; dy=10;
function runRef()
{
if ((parseInt(m.style.left)<25) ||
(parseInt(m.style.left)>document.body.clientWidth-25)) dx=-dx;
if ((m.style.posTop<25) ||
(m.style.posTop>document.body.clientHeight-25)) dy=-dy;
m.style.pixelLeft+=dx;
m.style.pixelTop+=dy;
}
</SCRIPT>
</HEAD>
<BODY">
<A NAME="m" HREF="js_1.htm"
STYLE="position:absolute;left:100;top:100;color:red"
onMousemove="runRef()"
onMouseover="runRef()"> Ссылка </A>
</BODY>
</HTML>
```

1.3. Перетаскивание элементов

Перетаскивание элементов Web-страницы с помощью мышки реализуется путем присваивания текущим координатам перемещаемого элемента текущих координат мышки. Доступ к координатам мышки можно осуществить, используя такие свойства объекта event:

- clientX - возвращает горизонтальную координату курсора мышки относительно клиенткой части окна (без учета рамок, заголовка, строки меню, панели инстументов и строки состояния);
- clientY - возвращает вертикальную координату курсора мышки относительно клиенткой части окна (без учета рамок, заголовка, строки меню, панели инстументов и строки состояния);
- offsetX - возвращает горизонтальную координату курсора мышки относительно элемента страницы, вызвавшего это событие;
- offsetY - возвращает вертикальную координату курсора мышки относительно элемента страницы, вызвавшего это событие;
- screenX - возвращает горизонтальную координату курсора мышки относительно экрана;
- screenY - возвращает вертикальную координату курсора мышки относительно экрана;

– x - возвращает горизонтальную координату курсора мышки относительно родительского элемента;

– y - возвращает вертикальную координату курсора мышки относительно родительского элемента.

В **примере № 3** создается Web-страница, содержащая три элемента - рисунок и два слова "ТЕКСТ" одинакового размера, но разных цветов.

Пример 3

```
<HTML>
<HEAD>
<TITLE>Перетаскивание элементов Web-страницы</TITLE>
<SCRIPT>
var n=0; l=0;
function drag()
{
with (document.all(n).style)
{
position="absolute";
/* window.status= "n="+n+" z-index="+l+" left="+pixelLeft+" top="+pixelTop+
" x="+event.x+" y="+event.y; */
left=event.x;
top=event.y;
zIndex=l;
}
}
</SCRIPT>
</HEAD>
<BODY onClick="n=event.srcElement.sourceIndex;"
onContextmenu="if(l==2) l=0; else l++;return false"
onMousemove="drag()">
<IMG SRC="fish.gif" >
<P STYLE="color:#0FF;font:8mm">TEXT1
<P STYLE="color:#F0F;font:8mm">TEXT2
</BODY>
</HTML>
```

При щелчке мышки по любому из этих объектов (по событию Click) для отмеченного объекта устанавливается режим перетаскивания: в переменную n заносится порядковый номер этого элемента Web-страницы, например, для рисунка: n=5, для первого текста: n=6, для второго: n=7 (при щелчке по свободному пространству окна браузера n принимает значение 4 - номер тэга BODY).

Режим перетаскивания отмеченного объекта реализуется путем вызова при движении мышки (по событию Mousemove) функции drag(), которая выполняет следующие действия:

устанавливает для объекта с номером n абсолютное позиционирование;

присваивает горизонтальной и вертикальной координатам этого объекта соответственно горизонтальную и вертикальную координату указателя мышкки, что позволяет ему перемещаться за движущейся мышкой;

устанавливает для объекта текущее значение свойства z-index, заданное в переменной l.

При щелчке правой кнопкой мышки (по событию Contextmenu) циклически меняется значение переменной l (0,1,2,0,...), задающей текущее значение свойства z-index, что позволяет любому объекту при задании соответствующего значения или перекрывать другие объкты Web-страницы, или располагаться под ними.

Для отмены режима перетаскивания отмеченного объекта необходимо повторно щелчкнуть мышкой. При этом в переменную n заносится номер уже нового объекта - того, который в момент щелчка находился под перетаскиваемым объектом. Это связано с тем, что при движении координаты указателя мышки на пиксел опережают координаты самого объекта. В этом случае для нового объекта устанавливается режим перетаскивания (если только этим объектом не является тэг BODY).

Добавим, что закомментированные строки функции drag() были использованы при отладке программы для вывода в поле статуса окна браузера значений таких контрольных данных: номер, z-index и координаты перетаскиваемого объекта, а также координаты мышки.

2. Контрольные вопросы:

1. Каким образом используются возможности JavaScript для создания движущихся объектов на Web-странице?
2. Перечислите возможные варианты изменения координат элементов.
3. Какие функции используются при реализации варианта с изменением координат элемента с помощью циклического процесса?
4. Какие функции используются при реализации варианта с изменением координат элемента по событиям мышки?
5. Какие функции используются при реализации варианта с перетаскиванием элементов?

3. Содержание отчета:

1. Наименование и цель лабораторной работы.
2. Результаты выполненных действий практической части в соответствии с заданием.
3. Ответы на контрольные вопросы.