Лабораторная работа 12

РАЗРАБОТКА ПРОГРАММЫ ДЛЯ ПОСТРОЕНИЯ ДВИЖУЩЕГОСЯ ИЗОБРАЖЕНИЯ

Цель работы: научиться программировать движение изображения по экрану.

ЗАДАНИЕ

1. Возьмите рисунок с координатами базовых точек, нарисованный при выполнении лабораторной работы 7. Пронумеруйте базовые точки. Около каждой базовой i-той точки напишите имена переменных Xi и Yi.

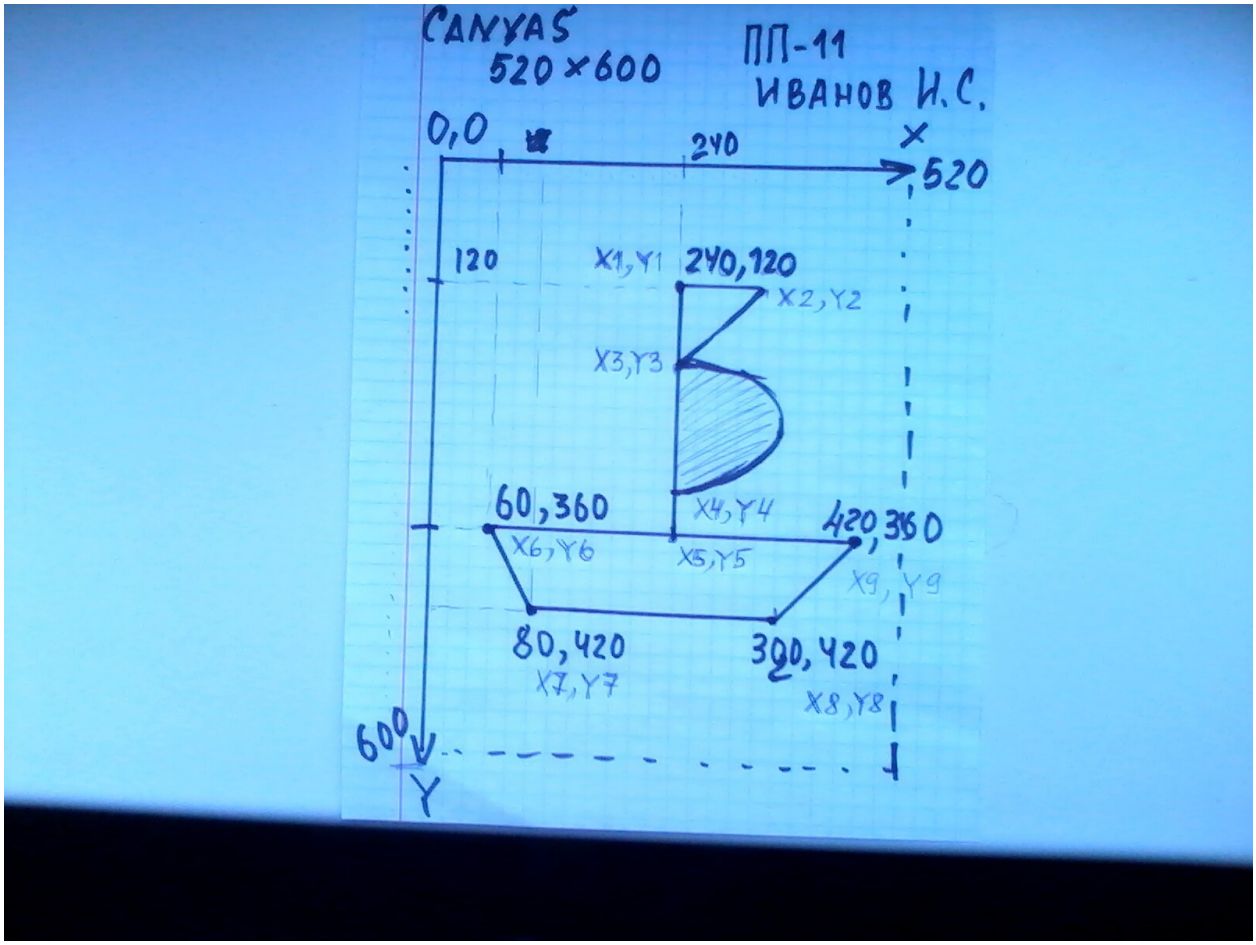
2. Добавьте в начало вашего скрипта объявление всех перемененных Xi и Yi.

3. В функции, которая рисует объект, замените координаты базовых точек этими переменными.

Результаты работы:

1) фото рисунка с именами переменных для значений координат базовых точек
2) HTML файл

ПРИМЕР Фото



ТЕХНОЛОГИЯ СОЗДАНИЯ ДВИЖУЩЕГОСЯ ИЗОБРАЖЕНИЯ

На самом деле, на экране компьютера ничего не движется. Программа создает иллюзию движения. Вспомните, как «делают» мультфильмы (анимационные фильмы). Мультипликация – это многократное повторение изображения одного и того же объекта в различных местах экрана.

Если нам надо создать иллюзию перемещения нарисованного объекта, например, вправо по экрану, наша программа должна многократно выполнять следующую последовательность действий:

1. Перерасчет координат всех базовых точек объекта.
2. Стирание предыдущего изображения
3. Построение изображения объекта с новыми значениями координат базовых точек.
   Как из неподвижной картинки сделать картинку с движущимся объектом, покажем на примере программки kolobok (программу можно скачать из вашего курса).

ЧАСТЬ 1 Лабораторная работа 11 «Рисование фона и колобка с помощью функций» - это было начало нашего проекта. Для того, чтобы иметь возможность многократно рисовать и перерисовывать фон и объект (колобка) для анимации, вы уже добавили в программу функции.

ЧАСТЬ 2 Задание координат базовых точек движущегося объекта с помощью переменных

Для того, чтобы иметь возможность программно изменять значения координат базовых точек нашего колобка, мы должны в командах рисования колобка заменить константы на переменные. В нашем примере в изображении колобка 3 опорных точки – это центры окружностей (ГОЛОВА И ГЛАЗКИ): x1, y1, x2,y2 , x3, y3.

Все эти переменные x1, y1, x2,y2 , x3, y3 в программе должны быть объявлены в начале скрипта с помощью служебного слова **var.**

Затем переменным должны быть присвоены начальные значения – это координаты, по которым строится наш колобок. Мы берем эти значения координат из функции function Kolobok().

В функции, function Kolobok(), которая рисует наш объект, заменяем координаты базовых точек этими переменными.

Изменения скрипта примера показаны красным цветом.

```
<!DOCTYPE HTML>
<html>
<body>
<canvas id="myCanvas" width="600" height="400" style="border:3px solid #c3c3c3;">
Your browser does not support the canvas element.
</canvas>
<script>
var c=document.getElementById("myCanvas");
var cxt=c.getContext("2d");

var x1,y1,x2,y2,x3,y3; /* Объявление переменных
// Задание начальных значений координат базовых точек
x1=50;
y1=200;
x2=40;
y2=200;
x3=60;
y3=200;

function fon()
{
// Эта функция не меняется, потому что фон не двигается
// Рисование фона - начало
cxt.clearRect(0, 0, 600, 400);
cxt.fillStyle="blue";
cxt.fillRect(0,0,600,200);
cxt.fillStyle="green";
cxt.fillRect(0,200,600,400);
// Рисование фона - конец
}

function Kolobok()
{
//колобок - начало
cxt.strokeStyle="red";
cxt.fillStyle="yellow";
cxt.beginPath();
// меняем числовые значения на соответствующие  переменные
cxt.arc(x1,y1,35,0,Math.PI*2,true);
cxt.closePath();
cxt.stroke();
```

```
cxt.fill();
// глазки
cxt.strokeStyle="black";
cxt.fillStyle="black";
cxt.beginPath();
cxt.arc(x2,y2,5,0,Math.PI*2,true);
cxt.closePath();
cxt.stroke();
cxt.fill();
cxt.strokeStyle="black";
cxt.fillStyle="black";
cxt.beginPath();
cxt.arc(x3,y3,5,0,Math.PI*2,true);
cxt.closePath();
cxt.stroke();
cxt.fill();
// Колобок -конец
}

fon();
Kolobok();

</script>
</body>
</html>
```

ЧАСТЬ 3

ЗАДАНИЕ

Доработайте графическую программу , разработанную в лабораторной работе №8, таким образом, чтобы она выполняла следующие функции:

1. **При запуске программы на экране появляются :**

1) фон;
2) объект в начальном положении;
3) 4 кнопки с обозначениями направлений движения влево, вправо, вниз и вверх.

2. **При нажатии на каждую из командных кнопок ваш объект должен перемещаться на маленькое расстояние в соответствующем направлении.**
3. **ВАЖНО: при движении в любом направлении объект не должен выходить за пределы Canvas и не должен исчезать с экрана.**

Результат работы: HTML файл

ТЕХНОЛОГИЯ СОЗДАНИЯ ДВИЖУЩЕГОСЯ ИЗОБРАЖЕНИЯ

На самом деле, на экране компьютера ничего не движется. Программа создает иллюзию движения. Вспомните, как «делают» мультфильмы (анимационные фильмы). Мультипликация – это многократное повторение изображения одного и того же объекта в различных местах экрана.

Если нам надо создать иллюзию перемещения нарисованного объекта, например, вправо по экрану, наша программа должна многократно выполнять следующую последовательность действий:

1. Перерасчет координат всех базовых точек объекта.
2. Стирание предыдущего изображения
3. Построение изображения объекта с новыми значениями координат базовых точек.
   Как из неподвижной картинки сделать картинку с движущимся объектом, покажем на примере программки kolobok (программу можно скачать из вашего курса).

Шаг 3 Добавление командной кнопки для управления движением

Часто в игровых программах объект на экране начинает движение ответ на нажатие пользователем на командную кнопку на экране. Для этого нам надо сделать следующее:
1) Добавить командную кнопку.
2) Запрограммировать функцию, которая будет вызываться при нажатии пользователем на эту командную кнопку.

Пример добавления командной кнопки для перемещения колобка вправо.

```
<button type = "button" onclick="Vpravo();" name = "button1"  style = "width: 80px; height:50px;">
  <b> >> </b>
</button><br><br>
```

При нажатии на кнопку button1 в программе происходит событие onclick, которое запускает на выполнение функцию Vpravo():

```
function Vpravo()
{
// вызов функции рисования фона
  Fon();
// Пересчет координат базовых точек
  x1=x1+10;
  x2=x2+10;
  x3=x3+10;
// При движении вправо крайняя правая точка колобка не должна выходить за правый край  Canvas
// Координата крайней правой точки колобка по оси X =X1+35 (центр окружности плюс радиус)
If (X1+35<myCanvas.width)
   {
  // вызов функции рисования колобка
     Kolobok();
  }
}
```

Финальная версия программы

```
<!DOCTYPE HTML>
<html>
<body>
<canvas id="myCanvas" width="600" height="400" style="border:3px solid #c3c3c3;">
Your browser does not support the canvas element.
</canvas>
<script>
var c=document.getElementById("myCanvas");
var cxt=c.getContext("2d");
var x1,y1,xl,yl,xr,yr;

function Fon()
{
cxt.clearRect(0, 0, 600, 400);
cxt.fillStyle="blue";
cxt.fillRect(0,0,600,200);
cxt.fillStyle="green";
cxt.fillRect(0,200,600,400);
}

function Kolobok()
{
//колобок
cxt.strokeStyle="red";
cxt.fillStyle="yellow";
cxt.beginPath();
cxt.arc(x1,y1,35,0,Math.PI*2,true);
```

```
cxt.closePath();
cxt.stroke();
cxt.fill();
// глазки
cxt.strokeStyle="black";
cxt.fillStyle="black";
cxt.beginPath();
cxt.arc(x2,y2,5,0,Math.PI*2,true);
cxt.closePath();
cxt.stroke();
cxt.fill();
cxt.strokeStyle="black";
cxt.fillStyle="black";
cxt.beginPath();
cxt.arc(x3,y3,5,0,Math.PI*2,true);
cxt.closePath();
cxt.stroke();
cxt.fill();
}

function Vpravo()
{
Fon();
x1=x1+10;
x2=x2+10;
x3=x3+10;
Kolobok();

}
//Начало игры
//стартовые координаты  колобка
x1=50;
y1=200;
y2=200;
y3=200;
x2=40;
x3=60;
Fon();
Kolobok();

</script>
// кнопка >> вправо
<br> <br>
<button type = "button" onclick="Vpravo();" name = "button1"  style = "width: 80px;
height:50px;">
  <b> >> </b>
</button><br><br>
```

```
</body>
</html>
```