## Canvas, основы работы с помощью JavaScript

Что такое canvas и зачем он нужен?

**Canvаs (холст) — элемент** **HTML5 для создания растрового двухмерного изображения. Обычно используется совместно с javascript**.

Ширину и высоту **canvas** можно изменять.

Используется в основном для отрисовки графиков и игровый элементов в браузерных играх, вставки видео, создания полноценного плеера.

**Canvas** так же используется в **WebGL** для аппаратного ускорения 3D графики. В результате можно создавать даже 3D игры, работающие в окне браузера.

Создание нашего canvas

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7*  *8*  *9*  *10*  *11*  *12*  *13*  *14*  *15*  *16* | *<!doctype html>*  *<html>*  *<head>*  *<title>Canvas</title>*  *<meta charset='utf-8' />*  *</head>*  *<body>*  *<canvas id='test' height='320' width='480'>Текст показывается, если элемент не поддерживается</canvas>*  *<script type="text/javascript">*  *var canvas  = document.getElementById("test");*  *var ctx     = canvas.getContext('2d');*  */\* Рисует контур прямоугольника на всю ширину и высоту canvas \*/*  *ctx.strokeRect(0, 0, canvas.width, canvas.height);*  *</script>*  *</body>*  *</html>* |

Создается **canvas** путем обычной вставки тега**<canvas>** в **html-код**. Далее через обычный **javascript** мы получаем элемент и содержимое **canvas** (строка 10, 11) и рисуем обведенный прямоугольник, который растягивается на всю ширину и высоту **canvas**.

Примитивы в canvas

В **canvas** можно рисовать такие геометрические элементы, как:

1. Прямоугольники
2. Линии
3. Окружности, дуги
4. Различные кривые, эллипс

Прямоугольники

Самая простая фигура для **canvas** — прямоугольник. Чтобы его нарисовать нам нужна всего одна строчка кода.

|  |  |
| --- | --- |
| *1* | *strokeRect(x, y, width, height); // пустой прямоугольник с обводкой* |

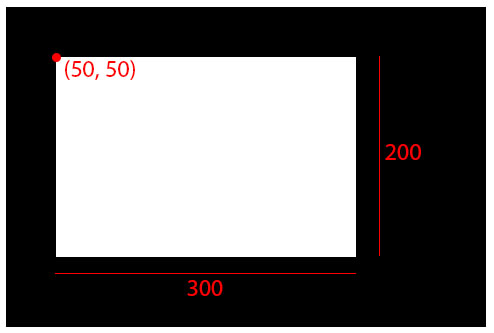
Есть еще 2 варианта для рисования прямоугольников:

|  |  |
| --- | --- |
| *1*  *2* | *fillRect(x, y, width, height)   // Закрашенный прямоугольник*  *clearRect(x, y, width, height)  // Очищам область на холсте  в виде прямоугольника заданного размера* |

Ниже приведен пример использования этих 2-х способов:

|  |  |
| --- | --- |
| *1*  *2* | *ctx.fillRect(0, 0, canvas.width, canvas.height);*  *ctx.clearRect(50, 50, 300, 200); // вырезаем прямоугольник* |

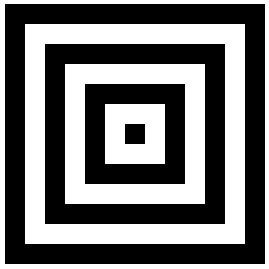
В результате мы нарисовали большой черный прямоугольник и вырезали область из точки (50, 50) размером 300 пикселей по ширине и 200 пикселей по высоте. Ниже представлено, как это будет выглядеть.



Более интересный пример:

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7*  *8* | *ctx.strokeRect(5, 5, 138, 138);*  *ctx.fillRect(10, 10, 128, 128);*  *for (i = 0; i <= 1; i += 2) {*  *for (j = 0; j <= 1; j += 2) {*  *ctx.clearRect(10 + i \* 64, 10 + j \* 64, 64, 64);*  *ctx.clearRect(10 + (i + 1) \* 64, 10 + (j + 1) \* 64, 64, 64);*  *}*  *}* |

Ниже пример, как это выглядит:



Линии, окружности, дуги

Рисование фигур из линий происходит немного сложней. Здесь используется 4 метода:

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7*  *8*  *9*  *10* | */\* дословно: начать путь. Используется, чтобы*  *начать рисовать фигуры \*/*  *beginPath()*  */\* дословно: закрыть путь. Используется, чтобы*  *завершить рисование. Необязательный метод \*/*  *closePath()*  */\* метод обводит фигуру линиями \*/*  *stroke()*  */\* Заливает фигуру сплошным цветом \*/*  *fill()* |

Так же нам понадобятся следующие методы для рисования линий:

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5* | *moveTo(x, y) // перемещает "курсор" в указанное место*  *lineTo(x, y) // ведёт линию из текущей позиции в новое*  *arc(x, y, radius, startAngle, endAngle, anticlockwise) /\* рисование окружности.*  *startAngle, endAngle - начальный и конечный угол.*  *anticlockwise - направление \*/* |

Ниже приведен пример использования данных методов:

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7*  *8*  *9*  *10*  *11*  *12*  *13*  *14*  *15* | *ctx.beginPath(); // начинаем рисовать*  *ctx.arc(160, 160, 30, 0, 360, false);*  *ctx.fill(); // заливаем окружность*    *ctx.moveTo(160, 0);*  *ctx.lineTo(200, 120);*  *ctx.lineTo(320, 160);*  *ctx.lineTo(200, 200);*  *ctx.lineTo(160, 320);*  *ctx.lineTo(120, 200);*  *ctx.lineTo(0, 160);*  *javascript_13_2_14_3ctx.lineTo(120, 120);*  *ctx.lineTo(160, 0);*  *ctx.stroke(); // обводим фигуры*  *ctx.closePath(); // закончили рисовать* |

Вот такая замечательная звезда у нас получилась:

**Кривая Безье**

Нарисовать любую фигуру так же можно с помощью кривых Безье. Для этого используется 2 метода **quadraticCurveTo**, **bezierCurveTo**. Для кривых Безье должна быть задана начальная точка, от которой будет прорисовываться фигура.

Посмотрим код:

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5* | *javascript_13_2_14_4ctx.lineWidth = 4;  // ширина линии*  *ctx.beginPath();*  *ctx.moveTo(50, 150);*  *ctx.bezierCurveTo(0, 40, 160, 80, 240, 40); // линия по 3-м точкам*  *ctx.stroke();* |

**Цвет линий и заливки в canvas**

Для задания цвета есть 2 свойства: **fillStyle** и **strokeStyle**. Задать цвет можно несколькими вариантами:

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4* | *ctx.strokeStyle = "red";*  *ctx.strokeStyle = "#FF0000";*  *ctx.strokeStyle = "rgb(255,0,0)";*  *ctx.strokeStyle = "rgba(255,0,0,1)"* |

Попробуем что-нибудь раскрасить, используя разные способы задания цвета:

|  |  |
| --- | --- |
| *1*  *2*  *3*  *4*  *5*  *6*  *7*  *8*  *9*  *10*  *11*  *12*  *13*  *14* | *ctx.strokeStyle = 'red'; // меняем цвет рамки*  *ctx.strokeRect(15, 15, 100, 100);*  *ctx.fillStyle = 'yellow'; // меняем цвет прямоугольника*  *ctx.fillRect(20, 20, 90, 90);*    *ctx.strokeStyle = '#00FF00'; // меняем цвет рамки*  *ctx.strokeRect(125, 15, 100, 100);*  *ctx.fillStyle = '#5500FF'; // меняем цвет прямоугольника*  *ctx.fillRect(130, 20, 90, 90);*    *ctx.strokeStyle = 'rgb(0,0,255)'; // меняем цвет рамки*  *ctx.strokeRect(235, 15, 100, 100);*  *ctx.fillStyle = 'rgb(255,0,0)'; // меняем цвет прямоугольника*  *ctx.fillRect(240, 20, 90, 90);* |

