

# Курс: Javascript. Событийно-ориентированное программирование

## Тема 7. Canvas

### *Теоретические сведения*

Что такое Canvas?

В браузере Internet Explorer на заре времён была встроенная технология элементов ActiveX, и, в частности, поддерживался DirectX; можно было динамически строить фигуры и применять к ним геометрические преобразования.

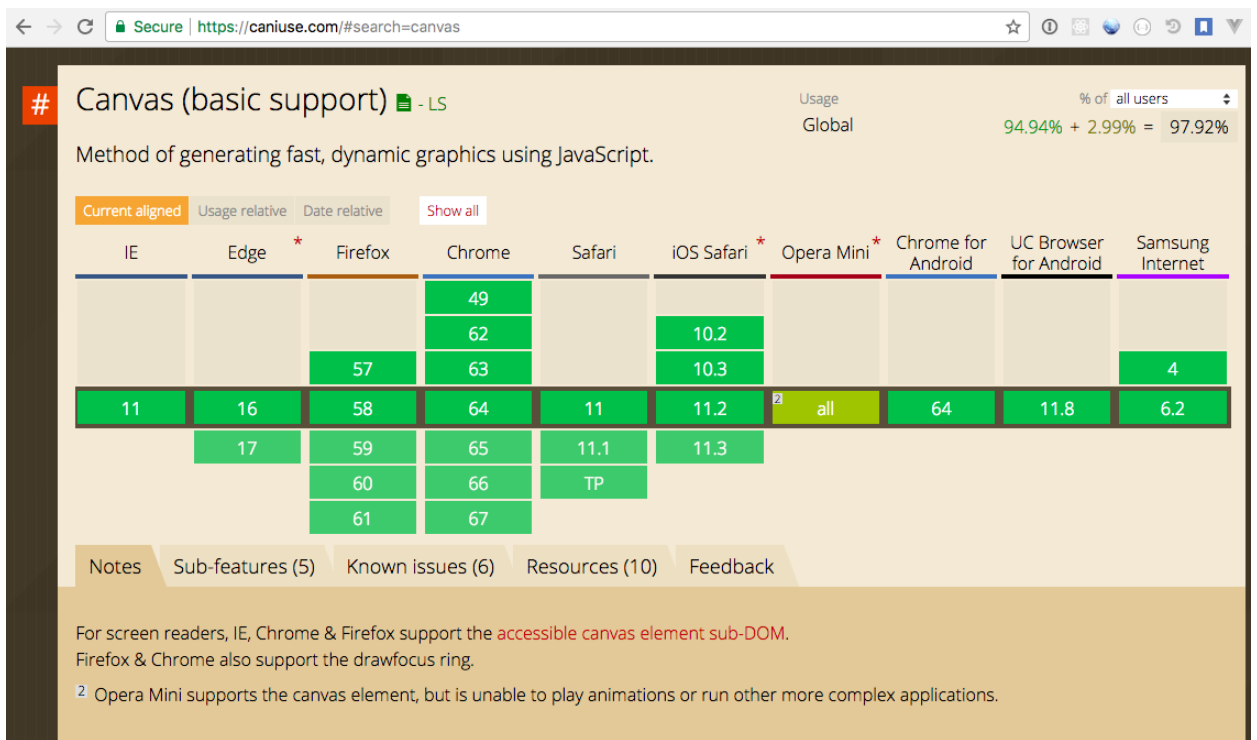
Сегодня Canvas стандартизовал этот подход. Его можно рассматривать как один из аспектов HTML5, вместе с SVG, Audio/Video и ... WebGL

Зачем нужен Canvas?

Перечислим основные кейсы применения:

- визуализация данных (графики-диаграммы);
- картография;
- игры;
- вообще геометрия.

Ядро Canvas определяет минимальный прикладной интерфейс, но оно необъектноориентированное (сам холст – это объект, но создание изображений происходит в функциональном стиле); существует много библиотек и фреймворков, например, Fabric.js, которые создают вокруг исходного объекта обёртки.



Что в основе?

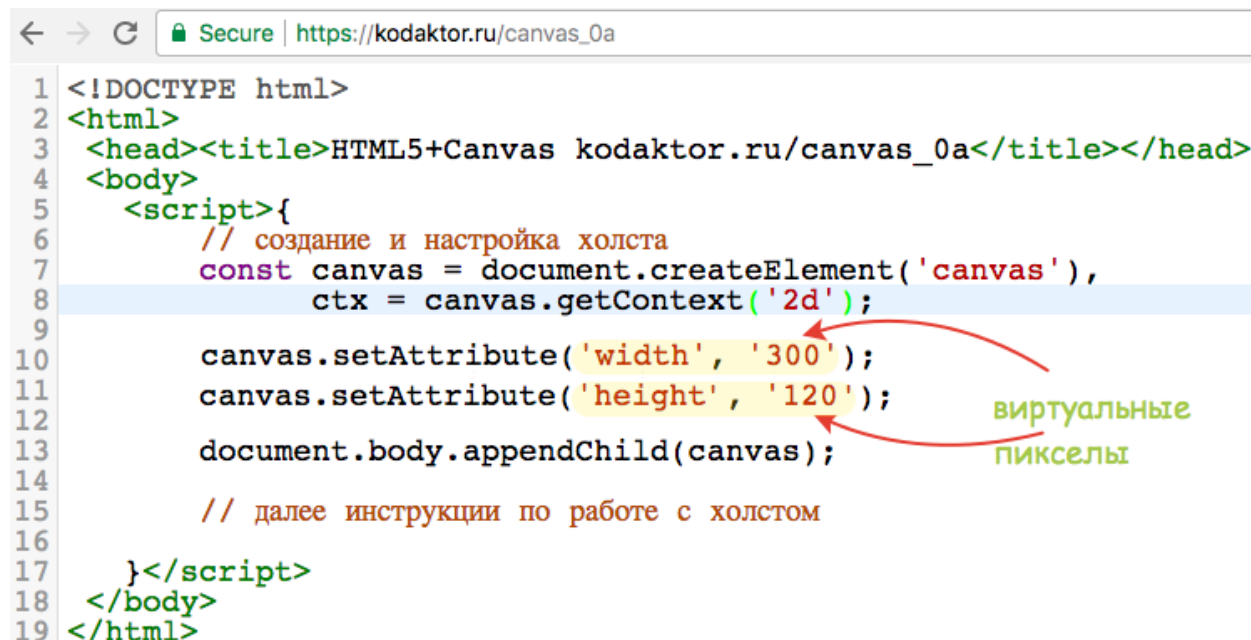
Система координат, связанная с прямоугольным полотном, вписанным в тело веб-страницы. Это прямоугольная система координат с началом координат в левом верхнем углу холста; ось абсцисс направлена вправо, а ось ординат – вниз (так по умолчанию, систему можно преобразовать).



Внедряем теги canvas внутрь тела веб-страницы ([kodaktor.ru/canvas\\_0a](http://kodaktor.ru/canvas_0a))  
 Вызываем метод `getContext` с параметром `2d`

Метод `getContext` возвращает объект `CanvasRenderingContext2D`, который, собственно, и инкапсулирует функциональность рисования.

Так выглядит минимальная отдельная страница с автоматизированной вставкой холста:



```
1 <!DOCTYPE html>
2 <html>
3 <head><title>HTML5+Canvas kodaktor.ru/canvas_0a</title></head>
4 <body>
5   <script>{
6     // создание и настройка холста
7     const canvas = document.createElement('canvas'),
8     ctx = canvas.getContext('2d');
9
10    canvas.setAttribute('width', '300');
11    canvas.setAttribute('height', '120');
12    document.body.appendChild(canvas);
13
14    // далее инструкции по работе с холстом
15
16  }</script>
17 </body>
18 </html>
```

(Размеры холста фундаментальны, их изменение вызывает сброс.)

Создадим отдельный сценарий в виде функции, возвращающей объект с канвасом и контекстом:



```
function makeCanvas(x, y) {
  const canvas = document.createElement('canvas'),
    ctx = canvas.getContext('2d');
  canvas.setAttribute('width', x);
  canvas.setAttribute('height', y);
  return { canvas, ctx };
}
```

Нужно подключить скрипт и вставить созданный объект в тело документа:

```
<body>
  <script src="/j/canvas_0b"></script>
  <script>{
    const { canvas, ctx } = makeCanvas(300, 120);
    document.body.appendChild( canvas );
  }</script>

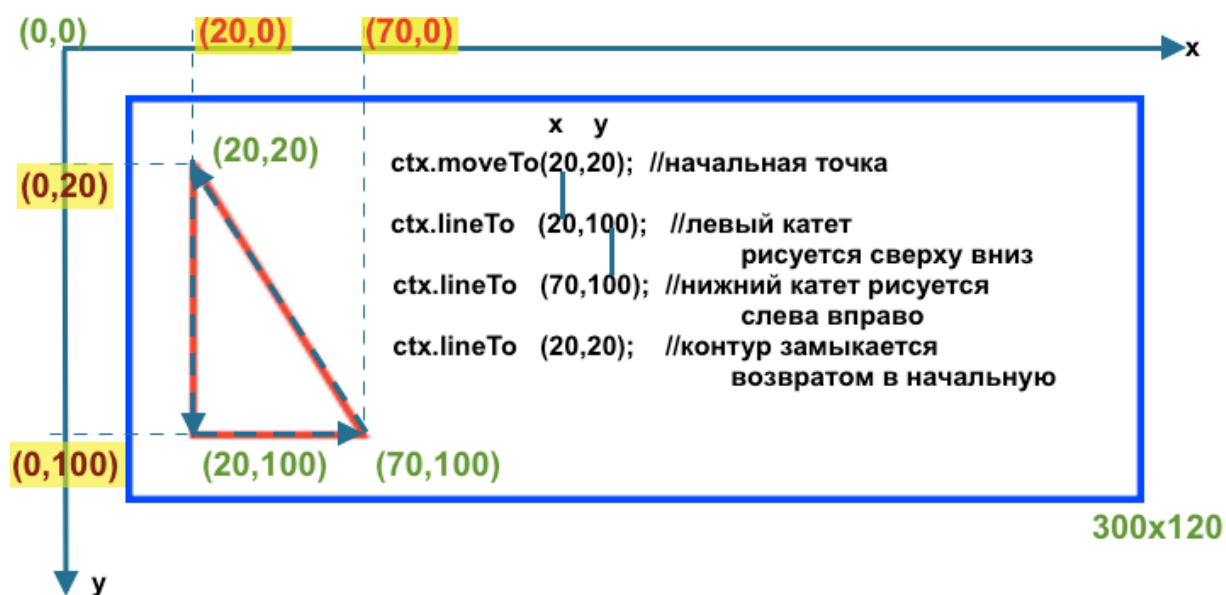
</body>
```

Каковая идеология рисования?

Подход "Чертёжника" или "Черепашки": рисование перемещением от одной точки к другой

Начальная точка задаётся как moveTo(x,y)

Далее.lineTo или например bezierCurveTo



Рамка:

[kodaktor.ru/canvas\\_1](http://kodaktor.ru/canvas_1)

Треугольник:

[kodaktor.ru/canvas\\_2](http://kodaktor.ru/canvas_2)

Что такое контур?

Фрагмент контура – это последовательность двух и более точек, соединённых линиями

Создаётся:

- метод `moveTo()` – начальная точка
- метод `lineTo()` – соединение с новой точкой

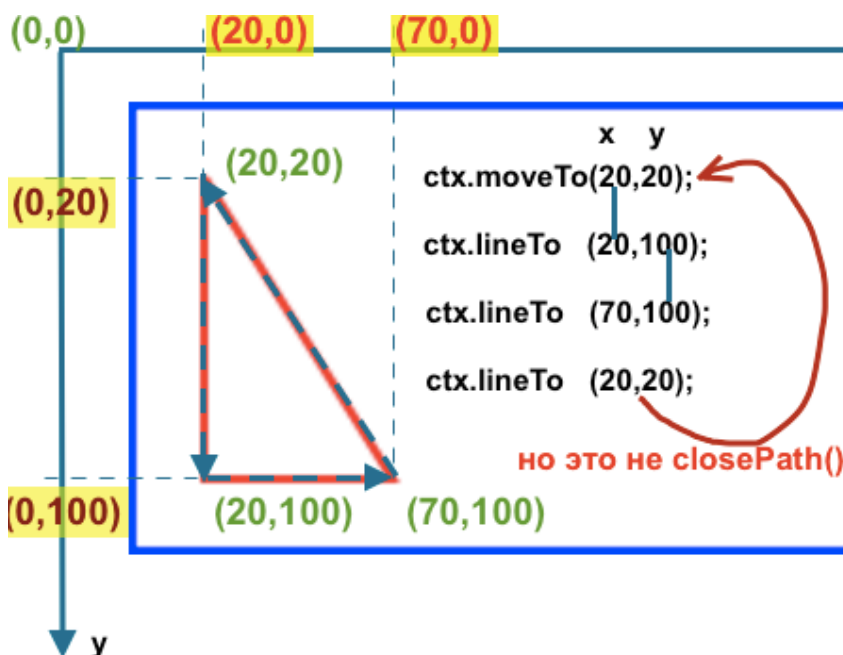
Контур (path) – это последовательность фрагментов (открытый контур – если последняя точка не совпадает с первой). Создаётся: `beginPath()`

[Закрывается: `closePath()`]

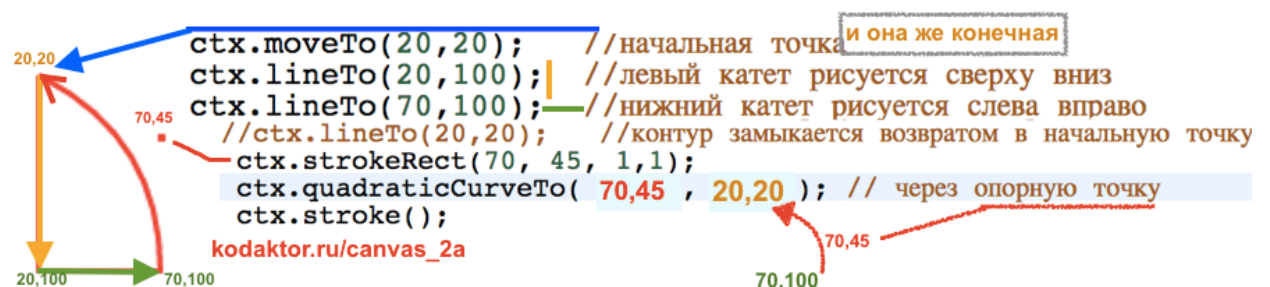
Эти команды как бы объявляют линии,  
а фактически отрисовывают их – вот эти:

- `stroke()`
- `fill()`

`fill` заливает область, как если бы она была замкнута (контур закрыт)



Заменяем прямую линию от (70,100) к (20,20) на кривую. Рисование квадратичной кривой идёт от предыдущей полученной точки к конечной точке "через" промежуточную.



### Упражнение

Чтобы нарисовать рамку по контуру холста, мы можем получить информацию о его ширине и высоте и воспользоваться методом `rect`

```
ctx.lineWidth = 2;
```

```
ctx.strokeStyle = 'blue';
```

```
ctx.rect(1, 1, canvas.width-2, canvas.height-2);    ctx.stroke();
```

Метод `rect` принимает 4 параметра: координаты левого верхнего угла (в текущей системе координат) и ширину с высотой – всё это в пикселах.

Что, если эти величины заданы массивом?

Например, `const rect1 = [1, 1, canvas.width-2, canvas.height-2];`

### Ответ

```
ctx.strokeRect(...rect1);
```

или

```
  apply:
```

```
ctx.rect.apply(ctx, rect1);
```

Холст работает с состояниями: это своего рода конечный автомат; текущее состояние определяется набором настроек типа толщины линии или типа заливки

Текущее состояние действует на текущий контур (`path`), т.е. если начать и не закрывать контур, изменения в состоянии будут на него воздействовать – об этом нужно помнить

Чтобы закрыть один контур и начать новый "текущий", нужно использовать `beginPath()`

Пример: [kodaktor.ru/canvas\\_3](http://kodaktor.ru/canvas_3)

Обратите внимание на то, что если контур не начинать, изменение цвета контура влияет на всё изображение...

### Реюзабильность

**[kodaktor.ru/canvas\\_6](http://kodaktor.ru/canvas_6)**

Холст позволяет использовать паттерны: превращать изображения, элементы `video` и сами холсты (включая виртуальные) в объекты повторного использования `CanvasPattern`.

Например, можно создать холст, залить его градиентом и нарисовать узор, а потом создать на его основе `createPattern` и использовать для других холстов.

### Аффинные преобразования, интерактивность и объектный стиль

**[kodaktor.ru/affine\\_c1](http://kodaktor.ru/affine_c1)**

Холст позволяет применять аффинные преобразования отдельно (сдвиг, масштабирование, вращение) и в матричной форме. При этом приходится держать в голове порядок применения преобразований; полезным бывает сохранение (`save`) текущих настроек и восстановление их (`restore`)

$$x' = a_1x + b_1y + c_1$$
$$y' = a_2x + b_2y + c_2$$

```
ctx.setTransform(a1, a2, b1, b2, c1, c2);
```

Если нужно работать с событиями, обращение происходит через обычный DOM (нет событийной специфики для реагирования на мышь, перетаскивание и т.д.)

Всё это лучше делать в фреймворке, таком как fabric

Пример: [kodaktor.ru/canvas\\_fabric1](https://kodaktor.ru/canvas_fabric1)

