Графическая информация в WEB

SVG
https://developer.mozilla.org/en-US/docs/Web/SVG
Canvas
https://developer.mozilla.org/ru/docs/Web/API/Canvas_API
Web GL
https://developer.mozilla.org/ru/docs/Web/API/WebGL_API
A-frame
https://aframe.io/

Ф.В.Филиппов, доцент кафедры ИУС СПбГУТ им. проф. М.А.Бонч-Бруевича 9000096@mail.ru

SVG (Scalable Vector Graphics)

Технология масштабируемой векторной графики позволяет объединить в одном формате текст, графику, анимацию и интерактивные компоненты и базируется на трех типах графических изображений: векторных формах, рисунках и тексте.

Раздел 1
ГРАФИЧЕСКИЕ ОБЪЕКТЫ SVG

1.1. Вводные замечания
1.2. Базовые объекты SVG
1.3. Группировка объектов
1.4. Трансформации объектов

1.1. Вводные замечания

Характеристика формата SVG

ТЕКСТОВЫЙ ФОРМАТ — файлы SVG можно читать и редактировать , меньше по размеру, чем сравнимые по качеству изображения в форматах JPEG или GIF, а также хорошо поддаются сжатию.

МАСШТАБИРУЕМОСТЬ — можно увеличить любую часть изображения SVG без потери качества.

МОЖНО ПРИМЕНЯТЬ ФИЛЬТРЫ — специальные модификаторы для создания эффектов, подобных применяемым при обработке растровых изображений (размытие, выдавливание, сложные системы трансформации и др.)

ИНДЕКСИРУЕТСЯ ПОИСКОВЫМИ МАШИНАМИ, не нужно создавать дополнительные метафайлы для поисковых роботов.

АНИМАЦИЯ - обеспечивается событийная модель, отслеживаются события (*загрузка страницы, изменение ее параметров, события мыши, клавиатуры и др.*)

ОТКРЫТЫЙ СТАНДАРТ - отличие от некоторых других форматов, SVG не является чьей-либо собственностью.

ИНТЕГРИРУЕТСЯ С HTML И XHTML документами – элементы SVG совместимы с HTML и XHTML (объединены моделью DOM).

СОВМЕСТИМ С CSS - отображением (форматированием и декорированием) SVG элементов можно управлять с помощью таблицы стилей CSS.

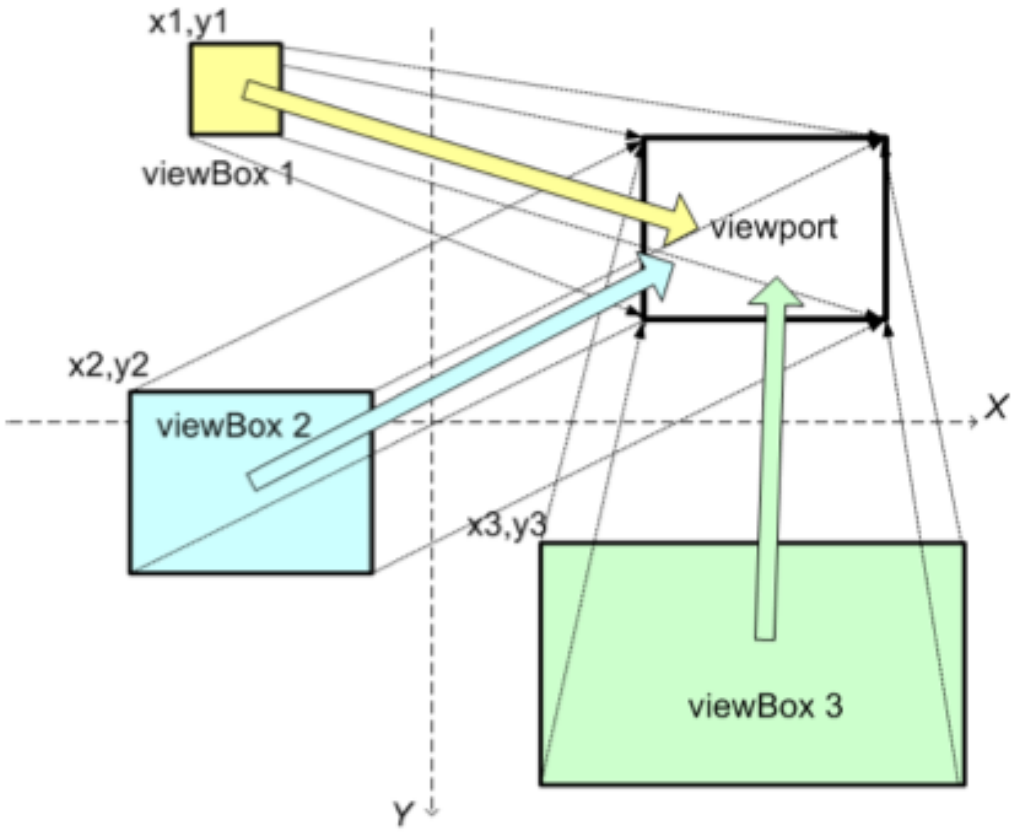
Пример SVG графики

https://svg-art.ru/?p=1083

Координаты объектов SVG графики

Размер окна просмотра viewport

```
<svg width="800" height="600">
        <!-- SVG контент ->
</svg>
```



1.2. Базовые объекты SVG

Все графические фигуры определяются некоторой комбинацией прямых и кривых линий …

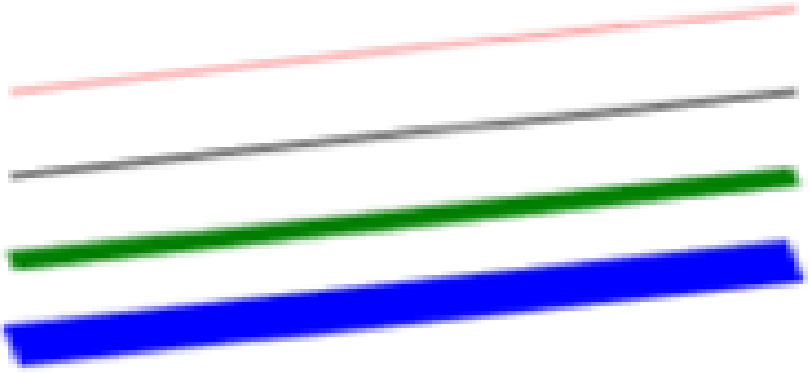
Базовые объекты SVG графики формируются с помощью тегов

line	– для отрезков прямой линии;
rect	– для прямоугольников;
circle	– для кругов и окружностей;
ellipse	– для эллипсов;
polyline	– для ломаных прямых линий;
polygon	– для многоугольников;
path	– для произвольных траекторий и фигур;
text	– для вставки текста;
image	– для вставки изображений.

Использование каждого тега предполагает указание координат (*x*, *y*) места расположения соответствующего объекта и некоторых его свойств.

Линия *<line>*

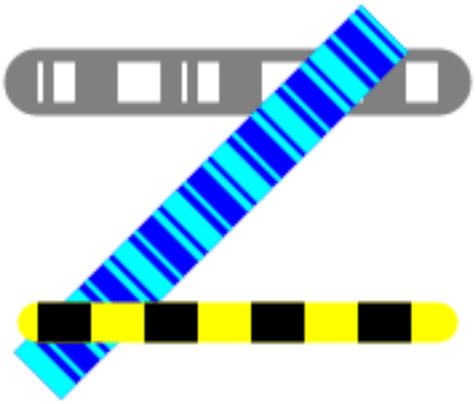
```
<svg xmlns="http://www.w3.org/2000/svg">
<line x1="5" y1="30" x2="100" y2="10" stroke-width=".5" stroke="red"/>
<line x1="5" y1="50" x2="100" y2="30" stroke-width="1" stroke="black"/>
<line x1="5" y1="70" x2="100" y2="50" stroke-width="4.5" stroke="green"/>
 <line x1="5" y1="90" x2="100" y2="70" stroke-width="10" stroke="blue"/>
</svg>
```



Влияние атрибутов *stroke-width* и *stroke*

Линия *<line>*

```
<svg xmlns="http://www.w3.org/2000/svg">
<line x1="30" y1="50" x2="180" y2="50" stroke-width="25" stroke="grey" stroke-linecap="round"/>
<line x1="30" y1="50" x2="180" y2="50" stroke-width="15" stroke="white" stroke-dasharray="2,4,8,16,16,8"/>
<line x1="30" y1="160" x2="160" y2="30" stroke-width="25" stroke="blue"/>
<line x1="30" y1="160" x2="160" y2="30" stroke-width="25" stroke="aqua" stroke-dasharray="8,3,2"/>
<line x1="30" y1="140" x2="180" y2="140" stroke-width="15" stroke="yellow" stroke-linecap="round"/>
<line x1="30" y1="140" x2="180" y2="140" stroke-width="15" stroke="black" stroke-dasharray="20,20"/>
</svg>
```



Влияние атрибутов *stroke-linecap* и *stroke=dasharray*

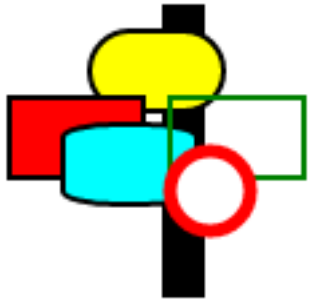
Линия *<line>*

```
<svg xmlns="http://www.w3.org/2000/svg">
<line x1="80" y1="80" x2="80" y2="80" stroke-width="100" stroke="blue" stroke-linecap="round"/>
<line x1="10" y1="40" x2="150" y2="40" stroke-width="15" stroke="yellow" stroke-opacity="0.4"/>
<line x1="10" y1="70" x2="150" y2="70" stroke-width="15" stroke="yellow" stroke-opacity="0.6"/>
<line x1="10" y1="100" x2="150" y2="100" stroke-width="15" stroke="yellow" stroke-opacity="0.8"/>
<line x1="10" y1="130" x2="150" y2="130" stroke-width="15" stroke="yellow" stroke-opacity="1.0"/>
</svg>
```

Влияние атрибутов *stroke-opacity*

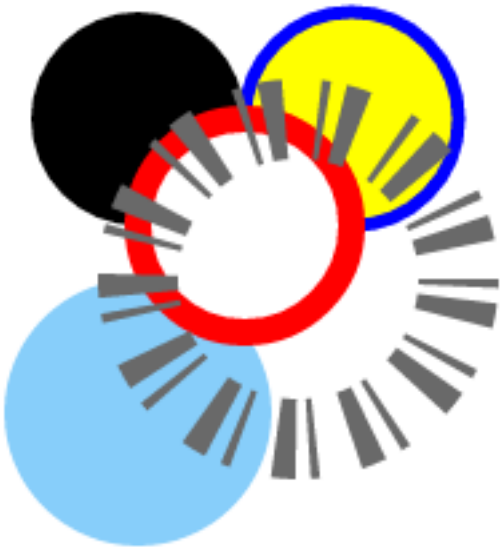
Прямоугольник *<rect>*

```
<svg xmlns="http://www.w3.org/2000/svg">
<rect x="62" y="25" height="110" width="16"/>
<rect x="35" y="35" height="30" width="50" fill="yellow" stroke="black" stroke-width="2" rx="15" />
<rect   x="5" y="60" height="30" width="50" fill="red" stroke="black" stroke-width="2"/>
<rect x="25" y="70" height="30" width="50" fill="cyan" stroke="black" stroke-width="2" rx="20" ry="5" />
<rect x="65" y="60" height="30" width="50" fill="none" stroke="green" stroke-width="2"/>
<rect x="65" y="80" height="30" width="30" fill="white" stroke="red" stroke-width="5" rx="15" />
</svg>
```



Окружность и круг *<circle>*

```
<svg xmlns="http://www.w3.org/2000/svg">
<circle cx="80" cy="50" r="40"/>
<circle cx="160" cy="50" r="40" fill="yellow" stroke="blue" stroke-width="5"/>
<circle cx="80" cy="160" r="50" fill="lightskyblue"/>
<circle cx="120" cy="90" r="40" fill="white" stroke="red" stroke-width="10" />
<circle cx="140" cy="110" r="60" fill="none” stroke="dimgray" stroke-width="30" stroke-dasharray="3,5,8,13"/>
</svg>
```



Android из *<line>*, *<rect>* и *<circle>*

```
<svg xmlns="http://www.w3.org/2000/svg">
<line x1="145" y1="75" x2="185" y2="145" stroke-width="15" stroke="yellowgreen" stroke-linecap="round"/>
<line x1="375" y1="75" x2="330" y2="145" stroke-width="15" stroke="yellowgreen" stroke-linecap="round"/>
<line x1="260" y1="200" x2="260" y2="250" stroke-width="15" stroke="white" stroke-linecap="round"/>
<line x1="200" y1="270" x2="325" y2="270" stroke-width="15" stroke="white" stroke-linecap="round"/>
<circle cx="260" cy="270" r="150" fill="yellowgreen"/>
<circle cx="185" cy="200" r="15" fill="white"/>
<circle cx="330" cy="200" r="15" fill="white"/>
<rect x="110" y="270" height="260" width="300" fill="yellowgreen" rx="25"/>
<rect x="110" y="270" height="10" width="300" fill="white"/>
<rect x="30" y="280" height="180" width="70" fill="yellowgreen" rx="35"/>
<rect x="418" y="280" height="180" width="70" fill="yellowgreen" rx="35"/>
<rect x="150" y="500" height="180" width="70" fill="yellowgreen" rx="35"/>
<rect x="295" y="500" height="180" width="70" fill="yellowgreen" rx="35"/>
</svg>
```

Эллипс *<ellipse>*

```
<svg xmlns="http://www.w3.org/2000/svg">
<ellipse cx="80" cy="110" rx="75" ry="105" fill="blue"/>
<ellipse cx="80" cy="110" rx="60" ry="40" fill="black" stroke="red" stroke-width="25"/>
<ellipse cx="80" cy="110" rx="35" ry="20" fill="green" stroke="yellow" stroke-width="25"/>
<ellipse cx="80" cy="50" rx="40" ry="30" fill="red" stroke="black" stroke-width="25"/>
<ellipse cx="80" cy="50" rx="30" ry="20" fill="orange" stroke="red" stroke-width="10" stroke-dasharray="5,5"/>
<ellipse cx="80" cy="170" rx="40" ry="30" fill="yellow" stroke="orange" stroke-width="25" stroke-dasharray="2,2"/>
<ellipse cx="80" cy="170" rx="30" ry="20" fill="red" stroke="black" stroke-width="10"/>
</svg>
```

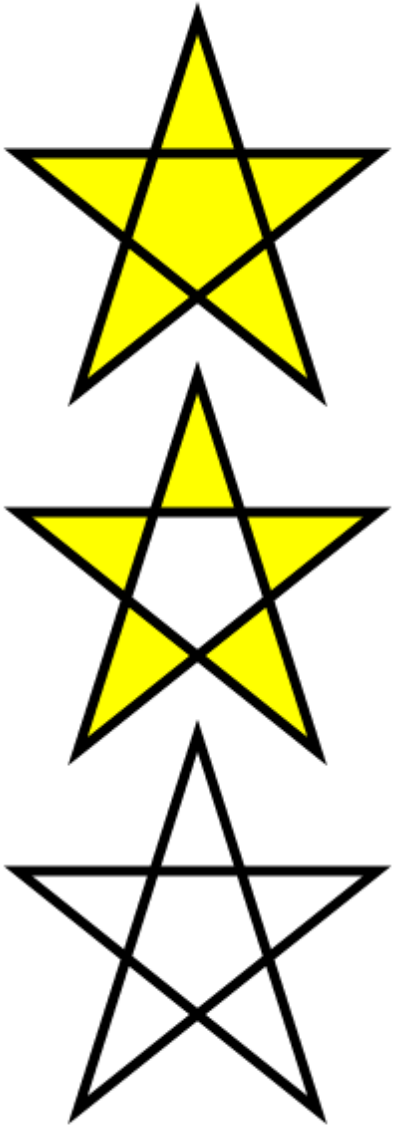
Ломаная линия *<polyline>*

```
<svg xmlns="http://www.w3.org/2000/svg">
<polyline points="80,70 30,70 70,20 70,110"/>
<polyline points="80,70 30,70 70,20 70,110" stroke="red" stroke-width="7" transform="translate(90,0)"/>
<polyline points="80,70 30,70 70,20 70,110" stroke="red" stroke-width="7" fill="none"
transform="translate(180,0)"/>
</svg>
```

transform="translate(90,0) – сдвиг на 90 пикселов вправо

Многоугольник *<polygon>*

```
<svg xmlns="http://www.w3.org/2000/svg">
<polygon points="100,10 40,198 190,78 10,78 160,198”
fill="yellow" stroke="black" stroke-width="5" fill-rule="nonzero"/>

<polygon points="100,10 40,198 190,78 10,78 160,198”
 fill="yellow" stroke="black" stroke-width="5" fill-rule="evenodd"
transform="translate(0,180)"/>

<polygon points="100,10 40,198 190,78 10,78 160,198"
fill="none" stroke="black" stroke-width="5" transform="translate(0,360)"/>
</svg>
```



transform="translate(0,180) – сдвиг на 180 пикселов вниз

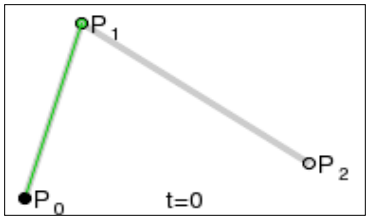
Кривые Безье

Квадратичные (Quadratic) **кривые**

Квадратичная кривая Пьера Безье опирается на три опорные точки P0, P1 и P2 которые определяют форму кривой.

Пример:

M10 10 Q 160,150 200,10



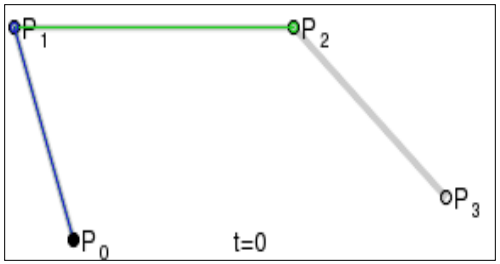
P0 — M10,10. Начало рисования.
P1 — 160,150. Контрольная точка, которая изменяет траекторию кривой.
P2 — 200,10. Финальная точка.
Q — определяет квадратичность кривой.

Кубические (Cubic) **кривые**

Для описания кубической кривой требуется четыре точки: P0, P1, P2 и P3, определяющие форму кривой.

Пример:

M10 10 C 160,150 200,10 300,150



P0 — M10 10 . Начало рисования.
P1 и P2 — 160,150 и 200,10. Направляющие, которые отклоняют линию от прямой.
P3 — 300,150 . Финальная точка.
C — определяет тип (кубичность) кривой.

Теория кривых Пьера Безье

Траектория <path>

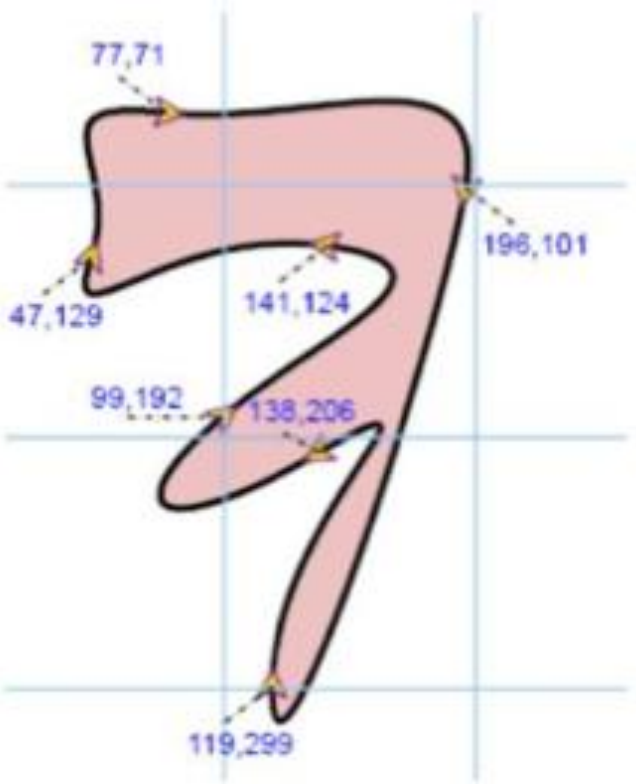
Функция	Описание
M-m	Переместиться в точку
L-l	Провести линию до точки
H-h	Провести горизонтальную линию до точки
V-v	Провести вертикальную линию до точки
C-c	Нарисовать кубическую кривую Безье
S-s	Нарисовать гладкую кривую
Q-q	Нарисовать квадратичную кривую Безье
T-t	Нарисовать гладкую квадратичную кривую Безье
A-a	Нарисовать эллиптическую дугу до точки
Z-z	Замкнуть траекторию в точке

Тег <path> позволяет создавать произвольные траектории с использованием набора специальных функций, приведенных в таблице.

С помощью тега <path> можно создать любую геометрическую фигуру из всех предыдущих примеров, и значительно более сложную.

Траектория *<path>*

```
<svg xmlns="http://www.w3.org/2000/svg">
<path stroke="black" stroke-width="3" fill="#eec1c2"
d="M 99 192
C 137 160 204 133 141 124
C 78 115 34 167 47 129
C 60 91 20 65 77 71
C 134 77 206 43 196 101
C 186 159 118 368 119 299
C 120 230 201 169 138 206
C 75 243 53 231 99 192" />
</svg>
```



Вставка текста *<text>*

```
<svg xmlns="http://www.w3.org/2000/svg">
<text x="10" y="30" font-size="20" fill="blue"> Текст в SVG! </text>
</svg>
```

Текст в SVG!

Атрибут	Описание
font-family	Определяет шрифт, например *Arial*
font-size	Задает размер шрифта
kerning	Задает промежуток между буквами
letter-spacing	То же, что и *kerning*
word-spacing	Задает промежуток между словами
text-decoration	Может принимать значения *underline*, *overline* или *line-through*
stroke	Задает цвет обводки букв
stroke-width	Задает толщину обводки букв
fill	Задает цвет букв

```
<svg xmlns="http://www.w3.org/2000/svg"
        xmlns:xlink="http://www.w3.org/1999/xlink">
<defs> <path id="path1" d="M75,20 a1,1 0 0,0 100,20"/> </defs>
<text x="10" y="100" fill="green">
<textPath xlink:href="#path1">Текст по пути path1 !</textPath> </text>
</svg>
```

Вставка изображений *<image>*

```
<svg xmlns="http://www.w3.org/2000/svg"
xmlns:xlink= "http://www.w3.org/1999/xlink">
<image xlink:href="photo.png" height="200" width="100”   x="10"   y="10"/>
<image xlink:href="photo.png" height="100" width="100" x="100”   y="10"/>
<image xlink:href="photo.png" height=  "30" width=  "30" x="150" y="150"/>
</svg>
```

1.3. Группировка объектов

SVG предоставляет возможности для структурирования документа посредством специальных элементов, которые позволяют определять и группировать объекты, а также ссылаться на них в дальнейшем …

Группировка <g>

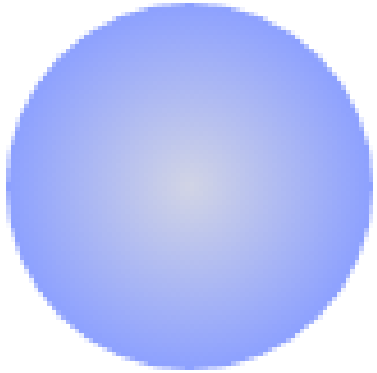
Тег <g> играет роль контейнера, в котором объединяет в группу все свое содержимое. Этой группе присваивается идентификатор, по которому производится обращение к ней в дальнейшем. Любые свойства, которыми наделяется элемент группы <g>, будут также применены ко всем его потомкам. Это позволяет задавать различные стили и преобразования, а также добавлять интерактивность и анимацию сразу целой группе объектов.

```
<svg xmlns="http://www.w3.org/2000/svg">

<g id="gr1" stroke="red" fill="white" stroke-width="5">
<circle cx="25" cy="25" r="15"/>
<circle cx="40" cy="25" r="15"/>
<circle cx="55" cy="25" r="15"/>
<circle cx="70" cy="25" r="15" stroke="green" fill="yellow" stroke-width="3"/>
</g>

</svg>
```

Определение *<defs>*

Тег *<defs>* может использоваться для хранения содержимого, которое не будет отображаться при определении. Содержимое в нем хранится в скрытом виде и ждет, когда оно будет использовано и отображено другими элементами *SVG*. Храниться в *<defs>* может что угодно, начиная с группы элементов и заканчивая маской, градиентом или фильтром.

```
<svg xmlns="http://www.w3.org/2000/svg"
        xmlns:xlink="http://www.w3.org/1999/xlink">

<defs>
<radialGradient id="gradient">
<stop offset="0%" stop-color="#d1d4e5" />
<stop offset="100%" stop-color="#8ca0ff" />
</radialGradient>
</defs>

<circle cx="50" cy="50" r="40" fill="url(#gradient)"/>

</svg>
```



Повторное использование *<use>*

Тег *<use>* использует в качестве атрибутов координаты *x*, *y* для размещения объекта и ссылку *xlink:href* на объект. В качестве ссылки выступает идентификатор объекта или группы.

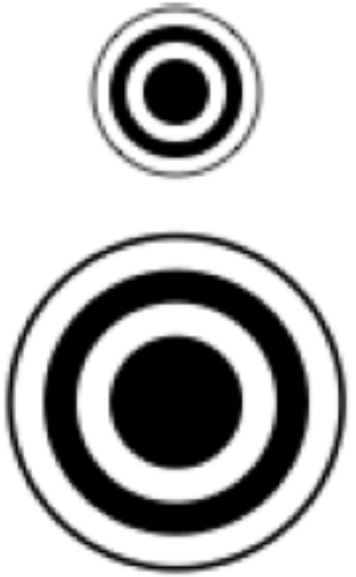
```
<svg xmlns="http://www.w3.org/2000/svg"
        xmlns:xlink="http://www.w3.org/1999/xlink">

<defs>
<g id="gr1">
<circle cx="25" cy="25" r="15"/>
<line x1="25" y1="40" x2="25" y2="100" stroke-width="1" stroke="gray"/>
</g>
</defs>

<use x="10" y="10" xlink:href="#gr1" fill="palegreen"/>
<use x="50" y="25" xlink:href="#gr1" fill="salmon" transform="scale(0.8)"/>
<use x="40" y="10" xlink:href="#gr1" fill="khaki" transform="scale(1.4)"/>

</svg>
```

Группировка *<symbol>*

Тег *<symbol>* по своим возможностям похож на тег *<g>* – он также предоставляет возможность группировать элементы, но имеет два отличия:

1) тег *<symbol>* не отображается сам по себе (как *<defs>*);
2) тег *<symbol>* может иметь собственные атрибуты *viewBox* и *preserveAspectRatio*. Это позволяет ему уместиться в области просмотра *(viewport)* так, как необходимо, а не как это определено по умолчанию.

```
<svg xmlns="http://www.w3.org/2000/svg"
      xmlns:xlink="http://www.w3.org/1999/xlink" width="256" height="256" viewBox="0 0 48 48" >

<symbol id="target" viewBox="0 0 200 200" preserveAspectRatio="xMidYMid meet">
 <circle cx="120" cy="120" r="50" fill="white" stroke="black" stroke-width="2"/>
<circle cx="120" cy="120" r="40" fill="black"/>
<circle cx="120" cy="120" r="30" fill="white"/>
<circle cx="120" cy="120" r="20" fill="black"/>
</symbol>

<use x="9" y="0" width="15" height="15" xlink:href="#target" />
<use x="0" y="5" width="30" height="30" xlink:href="#target" />
</svg>
```



1.4. Трансформации объектов

Трансформации служат для изменения формы или места расположения *SVG*-объектов …

Атрибут *transform*

Все преобразования осуществляются при помощи атрибута *transform*, который позволяет реализовать следующие трансформации:

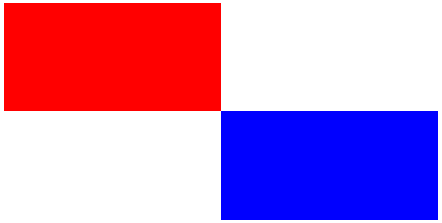
translate	– перемещение объекта
rotate	– поворот на заданный угол
scale	– масштабирование
skewX/skewY	– скос объекта вдоль оси *x* и/или *y*
matrix(a, b, c, d, e, f)	– множественная трансформация

К объектам SVG можно последовательно применять любое число раз- личных преобразований, перечисленных выше.

Перемещение объекта *translate*

Функция *translate* представляет собой перенос начала системы отсчета в новую точку с заданными координатами.

```
<svg xmlns="http://www.w3.org/2000/svg">
<rect x="25" y="25" width="50" height="25" fill="red"/>
<rect x="25" y="25" width="50" height="25" fill="blue" transform="translate(50,25)"/>
</svg>
```

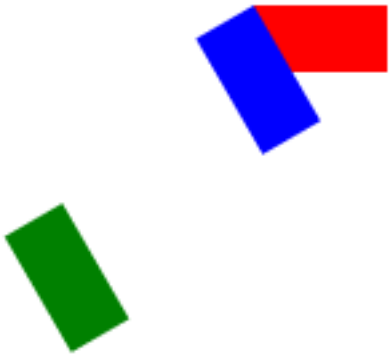


Поворот объекта *rotate*

Функция *rotate* поворачивает систему координат относительно точки отсчета на задаваемый угол. Если в функции *rotate* задан только угол поворота, то поворот объекта осуществляется относительно точки начала координат (0,0).

Для поворота фигуры вокруг точки принадлежащей самой фигуре в функции *rotate* следует указать координаты этой точки.

```
<svg xmlns="http://www.w3.org/2000/svg">
<rect x="100" y="25" width="50" height="25" fill="red"/>
<rect x="100" y="25" width="50" height="25" fill="green" transform="rotate(60)"/>
<rect x="100" y="25" width="50" height="25" fill="blue" transform="rotate(60, 100, 25)"/>
</svg>
```



Масштабирование *scale*

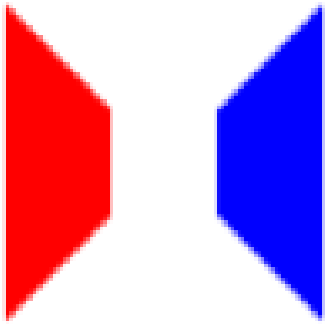
Функция *scale* позволяет масштабировать объект с указанием кратности изменения размеров вдоль горизонтальной и вертикальной осей. Если в функции указывается один параметр, то он принимается для обеих осей.

```
<svg xmlns="http://www.w3.org/2000/svg">
<rect x="10" y="25" width="50" height="25" fill="red"/>
<rect x="20" y="25" width="50" height="25" fill="green" transform="scale(2)"/>
<rect x="10" y="25" width="50" height="25" fill="blue" transform="scale(2,3)"/>
</svg>
```

Масштабирование *scale(-1,1)*

```
<svg xmlns="http://www.w3.org/2000/svg">
<path d="M20,20 l20,20 l0,20 l-20,20 Z" fill="red"/>
<path d="M20,20 l20,20 l0,20 l-20,20 Z" fill="blue” transform="translate(100, 0) scale(-1, 1)" />
</svg>
```

Исходная трапеция красного цвета. Прежде чем получить ее зеркальное отображение, т. е. выполнить функцию *scale(–1,1)*, она была окрашена в синий цвет и был выполнен ее сдвиг на 100 пикселов вправо.



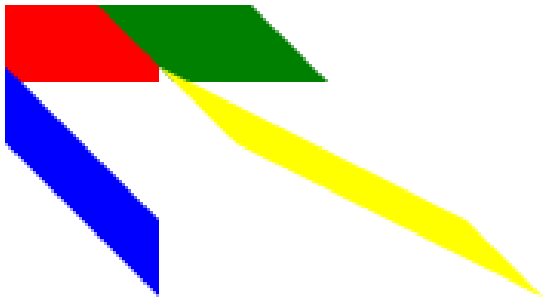
Скос *skewX* и *skewY*

Функции *skewX* и *skewY* предназначены для искажения формы фигуры в направлении горизонтальной и вертикальной осей.

Функция *skewX(n)* вычисляет новые значения ординат по формуле *Y*tg(n)*, а значения абсцисс оставляет без изменения (n – значение угла в градусах).

Это создает эффект покосившегося объекта.

```
<svg xmlns="http://www.w3.org/2000/svg">
<rect x="20" y="30" width="50" height="25" fill="red"/>
<rect x="20" y="30" width="50" height="25" fill="green" transform="skewX(45)"/>
<rect x="20" y="30" width="50" height="25" fill="blue" transform="skewY(45)"/>
<rect x="20" y="30" width="50" height="25" fill="yellow" transform="skewX(45) skewY(45)"/>
</svg>
```

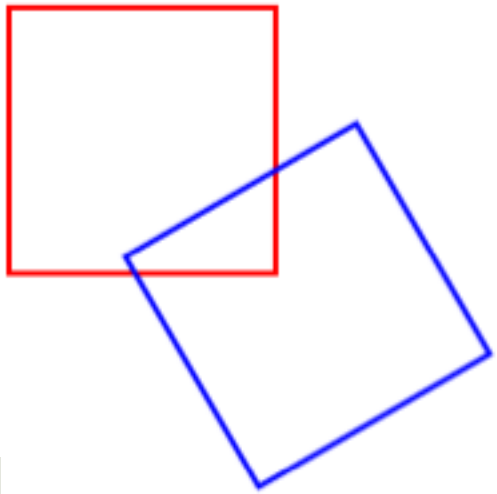


Множественная трансформация *matrix(a, b, c, d, e, f)*

a c e
b d f
0 0 1

Преобразование	Матричная запись
translate (x, y)	matrix(1,0,0,1,x,y)
rotate (a)	matrix(cos(a),sin(a),-sin(a),cos(a),0,0)
scale (kx, ky)	matrix(kx,0,0,ky,0,0)
skewX (a)	matrix(1,0,tg(a),1,0,0)
skewY (a)	matrix(1,tg(a),0,1,0,0)
Тождественное преобразование	matrix(1,0,0,1,0,0)

```
<svg xmlns="http://www.w3.org/2000/svg">
<polygon points="0,0 0,100 100,100 100,0” fill="none" stroke="red" stroke-width="2"/>
<polygon points="0,0 0,100 100,100 100,0” fill="none" stroke="blue" stroke-width="2"
transform="matrix(.866 -.5 .5 .866 40 100)"/> </svg>
```



поворот *rotate(30)* - matrix(cos(30),sin(30),–sin(30),cos(30),0,0)
перемещение *translate(40,100)* - matrix(1,0,0,1,40,100)

matrix(0.866,–0.5,0.5,0.866,40,100)

Раздел 2
СТАТИЧЕСКИЕ ВИЗУАЛЬНЫЕ ЭФФЕКТЫ

2.1. Градиент
2.2. Вырезание и маскирование
2.3. Фильтры

2.1. Градиент

Градиентом называют плавный переход от одного цвета к другому, причем самих цветов и переходов между ними может быть несколько…

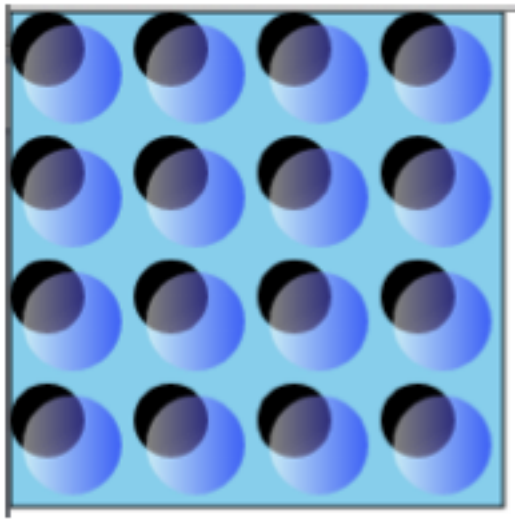
Линейный градиент *<linearGradient>*

```
<svg xmlns="http://www.w3.org/2000/svg"
        xmlns:xlink="http://www.w3.org/1999/xlink">

<linearGradient id="Grad1">
<stop offset="30%" stop-color="green"/>
<stop offset="70%" stop-color="yellow"/>
</linearGradient>

<linearGradient id="Grad2">
<stop offset="50%" stop-color="green"/>
<stop offset="50%" stop-color="yellow"/>
</linearGradient>

<linearGradient id="Grad3">
<stop offset="0%" stop-color="green"/>
<stop offset="100%" stop-color="yellow"/>
</linearGradient>

<rect x="20" y="20" width="50" height="50" fill="url(#Grad1)" />
<rect x="20" y="90" width="50" height="50" fill="url(#Grad2)" />
<rect x="20" y="160" width="50" height="50" fill="url(#Grad3)" />
</svg>
```

Радиальный градиент *<radialGradient>*

```
<svg xmlns="http://www.w3.org/2000/svg"
        xmlns:xlink="http://www.w3.org/1999/xlink">

<radialGradient id="G1">
<stop offset="0%" stop-color="yellow"/>
<stop offset="100%" stop-color="blue"/>
</radialGradient>
<rect height="150" width="150" x="0" y="0" fill="url(#G1)"/>

<radialGradient id="G2" cx="50%" cy="50%" r="20%">
<stop offset="0%" stop-color="yellow" stop-opacity=".4"/>
<stop offset="100%" stop-color="blue"/>
</radialGradient>
<rect height="150" width="150" x="0" y="160" fill="url(#G2)"/>

<radialGradient id="G3" cx="25%" cy="50%" r="100%">
<stop offset="0%" stop-color="yellow"/>
<stop offset="100%" stop-color="blue"/>
</radialGradient>
<rect height="150" width="150" x="0" y="320" fill="url(#G3)"/>

</svg>
```

Узор *<pattern>*



```
<svg xmlns="http://www.w3.org/2000/svg"
      xmlns:xlink="http://www.w3.org/1999/xlink">

<linearGradient id="Gr1">
<stop offset="5%" stop-color="white"/>
<stop offset="95%" stop-color="blue"/>
</linearGradient>

<pattern id="Pattern" x="0" y="0" width=".25" height=".25">
<rect x="0" y="0" width="50" height="50" fill="skyblue"/>
<circle cx="15" cy="15" r="15" fill="black"/>
<circle cx="25" cy="25" r="20" fill="url(#Gr1)" fill-opacity="0.5"/>
</pattern>

<rect fill="url(#Pattern)" stroke="black" x="0" y="0" width="200" height="200"/>
</svg>
```

2.2. Вырезание и маскирование

Иногда требуется выделить только часть изображения, скрывая остальные детали. Либо продемонстрировать объект, показывая его через полупрозрачный занавес…

Вырезание *<clipPath>*

```
<svg xmlns="http://www.w3.org/2000/svg"
        xmlns:xlink="http://www.w3.org/1999/xlink">

<clipPath id="c1">
<circle cx="275" cy="110" r="80"> </circle>
</clipPath>

<image x="10" y="10" width="550" height="400” clip-path="url(#c1)" xlink:href="bird.png"/>

<clipPath id="c2">
<circle cx="220" cy="110" r="80"> </circle>
<circle cx="320" cy="110" r="80"> </circle>
</clipPath>

<g transform="translate(10,200)">
<image x="10" y="10" width="550" height="400” clip-path="url(#c2)" xlink:href="bird.png"/>
</g>
</svg>
```

Маскирование *<mask>*

```
<svg xmlns="http://www.w3.org/2000/svg"
      xmlns:xlink="http://www.w3.org/1999/xlink">

<radialGradient id="Grad">
<stop offset="80%" stop-color="white" stop-opacity="1"/>
<stop offset="100%" stop-color="white" stop-opacity="0"/>
</radialGradient>

<mask id="Circ">
<circle cx="275" cy="110" r="80" fill="url(#Grad)"/>
</mask>

<image x="10" y="10" width="550" height="400" mask="url(#Circ)" xlink:href="bird.png"/>
</svg>
```

2.3. Фильтры

SVG-фильтры – очень мощный и вместе с тем достаточно сложный инструмент для получения различных визуальных эффектов…

Тег (тип фильтра)	Назначение
feBlend	Смешивание изображений
feColorMatrix	Коррекция цвета и яркости изображения
feComponentTransfer	Коррекция контраста и цветового баланса
feComposite	Арифметические и логические операции над слоями
feConvolveMatrix	Сверточные операции
feDiffuseLighting	Освещение изображения с использованием альфа-канала
feDisplacementMap	Смещение пикселов в изображении
feFlood	Тонирование изображений
feGaussainBlur	Операция размытия изображения
feImage	Позиционирование изображения
feMerge	Совмещение нескольких фильтров
feMorphology	Художественная обработка изображений
feOffset	Смещение исходного изображения
feSpecularLighting	Добавление источника света к изображению
feTile	Формирование текстуры из изображения
feTurbulence	Синтез текстур с использованием шума Перлина
Фильтры – источники света	
fePointLight	Точечное освещение изображения
feDistantLight	Дистанционное освещение изображения
feSpotLight	Спот-освещение изображения

Фильтр *<feGaussianBlur>*

```
<svg xmlns="http://www.w3.org/2000/svg"
        xmlns:xlink="http://www.w3.org/1999/xlink">

<circle cx="60" cy="60" r="50" fill="blue" />

<filter id="fGB1">
<feGaussianBlur in="SourceGraphic" stdDeviation="3" />
</filter>
<circle cx="170" cy="60" r="50" fill="blue" filter="url(#fGB1)" />

<filter id="fGB2">
<feGaussianBlur in="SourceGraphic" stdDeviation="9" />
</filter>
<circle cx="280" cy="60" r="50" fill="blue" filter="url(#fGB2)" />
</svg>
```

Раздел 3
ДИНАМИЧЕСКИЕ ВИЗУАЛЬНЫЕ ЭФФЕКТЫ

3.1. Анимация атрибутов
3.2. Анимация движения
3.3. Анимация трансформации
3.4. Анимация с использованием *viewBox*

3.1. Анимация атрибутов

Тег <*animate*> – позволяет анимировать атрибуты объектов в течение определенного периода времени…

Атрибуты для анимации атрибутов

Атрибут	Описание
begin	Задание времени старта анимации
dur	Продолжительность анимации
end	Задание времени окончания
restart	Возможность повторного воспроизведения
repeatCount	Число повторений анимации
repeatDur	Задание времени для возможности повторного воспроизведения
fill	Определение поведения анимируемого элемента по окончании анимации
min	Минимальный период анимации
max	Максимальный период анимации
from	Начальное значение анимируемого свойства (цвет, координата, размер)
to	Конечное значение анимируемого свойства
by	Указание относительного сдвига анимации
values	Промежуточные значения анимируемого свойства
keyTimes	Указание ключевых кадров
calcMode	Задание темпа анимации
keySplines	Задание темпа анимации каждого ключевого кадра
additive	Задание способа отсчета значений атрибутов from и to
accumulate	Задание накопительной анимации

Атрибут *r*

```
<svg xmlns="http://www.w3.org/2000/svg">
<circle cx="140" cy="140" fill="red" r="">
<animate attributeType="XML"
                attributeName="r"
                from="25"
                to="100"
                dur="5s"
                repeatCount="indefinite" />
</circle>
</svg>
```

ЗАПУСК

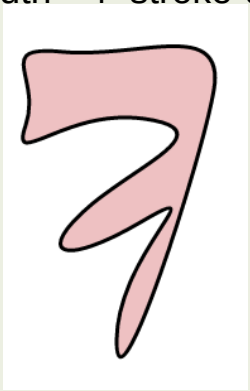
Атрибуты *r* и *fill*

```
<svg xmlns="http://www.w3.org/2000/svg">
<circle cx="140" cy="140" fill="red" r="">
<animate attributeType="XML"
                attributeName="r"
                from="25"
                to="100"
                dur="5s"
                repeatCount="indefinite" />
<animate attributeType="XML"
                attributeName="fill"
                from="red"
                to="yellow"
                dur="5s"
                repeatCount="indefinite" />
</circle>
</svg>
```

ЗАПУСК

Атрибут *stroke-dasharray*

```
<svg xmlns="http://www.w3.org/2000/svg"
        xmlns:xlink="http://www.w3.org/1999/xlink">
<path class="path" fill="#FFFFFF" stroke="#000000" stroke-width="4" stroke-dasharray="0,1000"
        d="M 99 192
        C 137 160 204 133 141 124
        C 78 115 34 167 47 129
        C 60 91 20 65 77 71
        C 134 77 206 43 196 101
        C 186 159 118 368 119 299
        C 120 230 201 169 138 206
        C 75 243 53 231 99 192" >
  <animate attributeName="stroke-dasharray"
        from="0,1000”
        to="1000,0"
        begin="0”
        dur="10s”
        repeatCount="indefinite"
        restart="whenNotActive” />
</path>
</svg>
```



Запуск

3.2. Анимация движения

Тег <*animateMotion*> позволяет анимировать движение объекта вдоль заданной траектории. Траектория движения объекта может быть задана двумя способами: с помощью атрибута *path* или с использованием тега внешнего пути <*mpath*>…

Движение круга по траектории *path*

```
<svg xmlns="http://www.w3.org/2000/svg"
        xmlns:xlink="http://www.w3.org/1999/xlink">

<circle id="circ1" r="20" cx="100" cy="100" fill="red" />
<animateMotion
        xlink:href="#circ1"
        dur="3s"
        begin="click"
        fill="freeze"
        path="M0,0 c3.2-3.4,18.4-0.6,23.4-0.6
        c5.7,0.1,10.8,0.9,16.3,2.3
        c13.5,3.5,26.1,9.6,38.5,16.2 c12.3,6.5,21.3,16.8,31.9,25.4
        c10.8,8.7,21,18.3,31.7,26.9 c9.3,7.4,20.9,11.5,31.4,16.7
        c13.7,6.8,26.8,9.7,41.8,9 c21.4-1,40.8-3.7,61.3-10.4    c10.9-
3.5,18.9-11.3,28.5-17.8c5.4-3.7,10.4-6.7,14.8-11.5       c1.9-2.1,3.7-
5.5,6.5-6.5"/>
</svg>
```

Запуск

Тег *<mpath>* для задания внешнего пути

```
<svg xmlns="http://www.w3.org/2000/svg"
        xmlns:xlink="http://www.w3.org/1999/xlink">
<path id="motionPath" fill="none" stroke="gainsboro" stroke-width="5"
      d="M 99 192
      C 137 160 204 133 141 124
      C 78 115 34 167 47 129
      C 60 91 20 65 77 71
      C 134 77 206 43 196 101
      C 186 159 118 368 119 299
      C 120 230 201 169 138 206
      C 75 243 53 231 99 192" />

<circle id="circle" r="10" cx="0" cy="0" fill="red" />
<animateMotion
      xlink:href="#circle"
      dur="5s"
      begin="0s"
      fill="freeze"
      repeatCount="indefinite">
<mpath xlink:href="#motionPath" />
</animateMotion>
</svg>
```

Запуск

3.3. Анимация трансформаций

Тег <*animateTransform*> анимирует атрибуты трансформации целевого объекта, тем самым позволяя управлять сдвигом, масштабом, вращением и наклоном…

Анимация вращения

```
<svg xmlns="http://www.w3.org/2000/svg"
        xmlns:xlink="http://www.w3.org/1999/xlink">
<rect id="greenRect" width="50" height="50" x="50" y="50" fill="green" />
<animateTransform
        xlink:href="#greenRect"
        attributeName="transform"
        attributeType="XML"
        type="rotate"
        from="0 75 75"
        to="360 75 75"
        dur="2s"
        begin="0s"
        repeatCount="indefinite"
        fill="freeze”/>
</svg>
```

Запуск

Анимация размера путем изменения масштаба

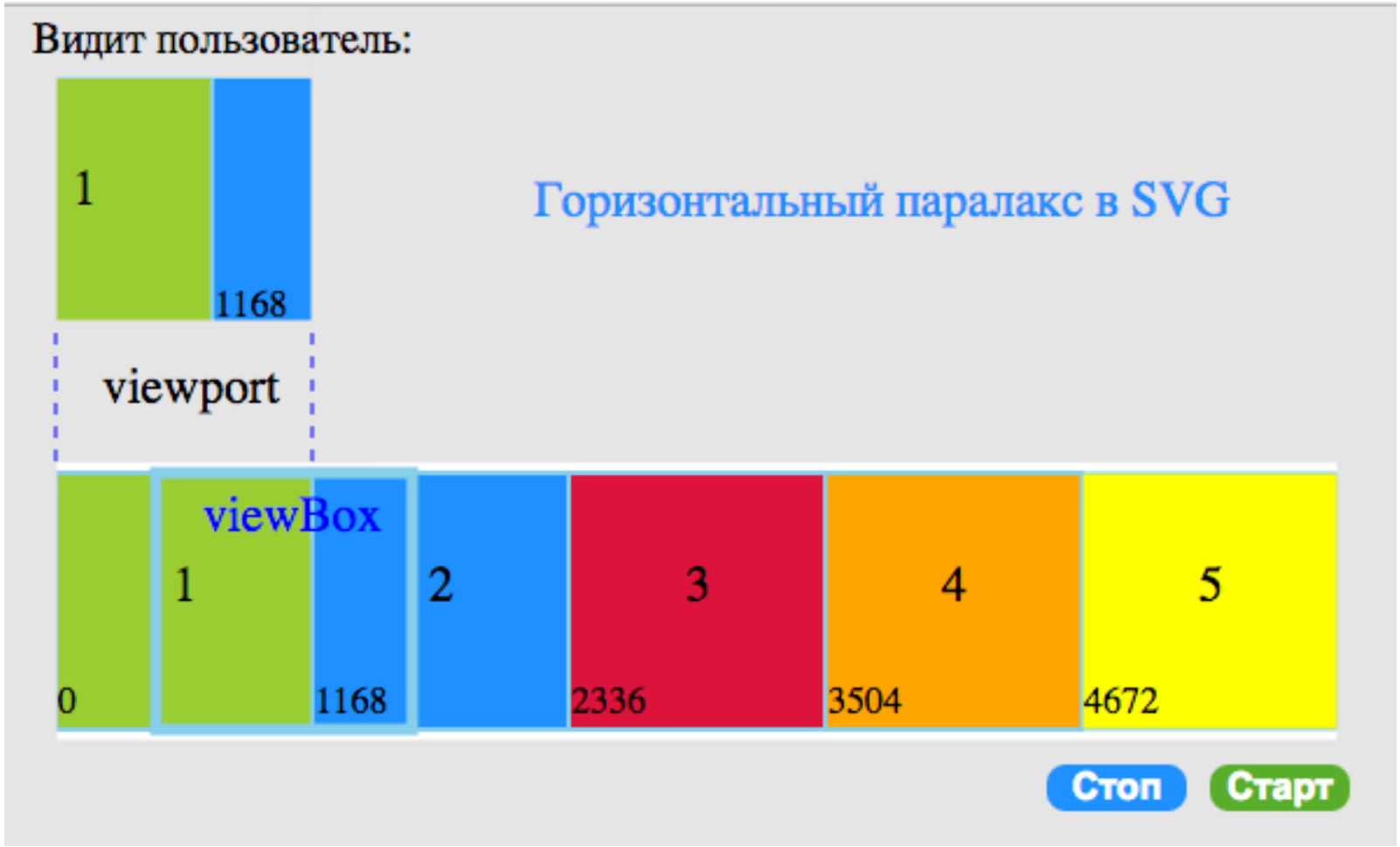
```
<svg xmlns="http://www.w3.org/2000/svg">
    <rect width="100" height="100">
    <animateTransform
        attributeName="transform"
        type="scale"
        from="2"
        to="12"
        repeatCount="10"
        dur="12s"
        fill="freeze"/>
    </rect>
</svg>
```

Запуск

3.4. Анимация с использованием *viewBox*

Если заставить видовое окно *viewBox* перемещаться, наподобие перемещения видеокамеры, изображение в окне просмотра также будет перемещаться. На базе этого свойства можно создать анимацию …

Горизонтальный параллакс



Раздел 4
ИНТЕРАКТИВНОСТЬ SVG

4.1. Гиперссылки
4.2. События
4.3. Взаимодействие с HTML
4.4. Сценарии JavaScript
4.5. Библиотеки JavaScript

4.1. Гиперссылки

Одна из наиболее распространенных форм интерактивности – это переходы по гиперссылкам…

Использование тега <a>

```
<svg xmlns="http://www.w3.org/2000/svg"
        xmlns:xlink="http://www.w3.org/1999/xlink">

      <a xlink:href="https://www.sut.ru" target="_blank">
            <rect height="30" width="100" y="10" x="10" rx="15"
fill="darkorange"/>            <text fill="white" x="18" y="31">БОНЧ
здесь!</text>
      </a>
</svg>
```

4.2. События

Спецификация консорциума *W3C* разбивает атрибуты, поддерживающие события по трем типам: атрибуты событий для объектов, атрибуты событий *SVG*-документа и атрибуты событий анимации…

Обработка событий мыши

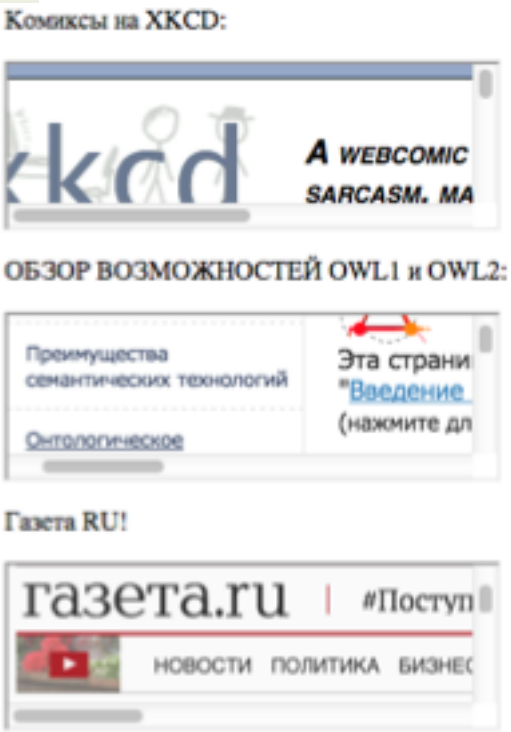
```
<svg xmlns="http://www.w3.org/2000/svg">
<circle cx="50" cy="50" r="40" fill="red" stroke="black" stroke-width="2" >
<set attributeName="fill"
        to="green"
        begin="mousemove"
        end="mouseout"/>
</circle>

<circle cx="50" cy="150" r="40" fill="yellow" stroke="black" stroke-width="2">
<animate attributeName="r"
        from="40"
        to="100"
        begin="click"
        dur="3s"/>
</circle>

<circle cx="50" cy="250" r="40" fill="green" stroke="black" stroke-width="2">
<animate attributeName="cy"
        from="250"
        to="50"
        begin="focusin"
        dur="3s"
        end="focusout"/>
</circle> </svg>
```

4.3. Взаимодействие с HTML

Как любой *SVG*-объект может быть легко встроен в любое место на страницу *HTML*-документа, так и наоборот, *HTML*-код может быть использован внутри *SVG*-документа…

Пример вставки *HTML* в *SVG*

```
<svg xmlns = "http://www.w3.org/2000/svg"
        xmlns:xlink="http://www.w3.org/1999/xlink" width="800" height="800">

<foreignObject x="80" y="30" width="800" height="800">

        <p>Комиксы на XKCD:</p>
        <iframe src="http://xkcd.com/559/" width="300" height="100"></iframe>

        <p>ОБЗОР ВОЗМОЖНОСТЕЙ OWL1 и OWL2:</p>
        <iframe src="http://trinidata.ru/tech_owl.htm" width="300"
                                height="100"></iframe>

        <p>Газета RU!</p>
        <iframe src="http://gazeta.ru" width="300" height="100"></iframe>

</foreignObject>

</svg>
```



4.4. Сценарии JavaScript

Добавление скриптов внутрь *SVG* можно реализовывать как внутри документа, так и с помощью внешних ссылок…

Движущаяся мишень

```
<html>
<h1>Попади в мишень </h1>
 <svg height="150" width="100%">
        <circle id="aim" r="50" cy="75" fill="blue">
        <animate attributeName="cx"
                values="0%;100%;30%;65%;5%;95%;0%;75%;20%;90%;0%"
                dur="10s"
                repeatDur="indefinite"
                calcMode="paced" />
        </circle>
</svg>
<!--Формирование таблицы для записи результатов-->
        <ul id="log"> Результат:</ul>
<!--Вызов внешнего JavaScript сценария-->
        <script src="Hit.js"></script>
</html>
```

Запуск

4.5. Библиотеки JavaScript

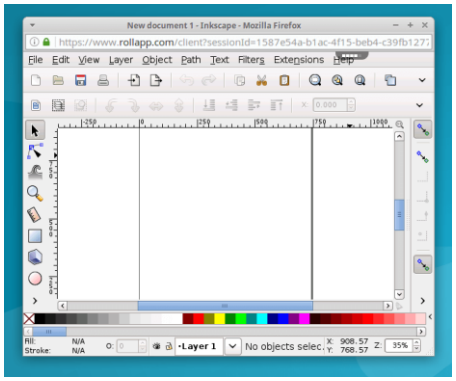
Для программистов и веб-дизайнеров, существует *CDNJS – Content Delivery Network JavaScript* – сеть доставки контента, включающего *JavaScript*-библиотеки различного назначения. ..

Раздел 5
РЕДАКТОРЫ SVG

5.1. Редактор Inkscape
5.2. Редактор Method Draw
5.3. Редактор Boxy SVG
5.4. Редактор Archer Editor

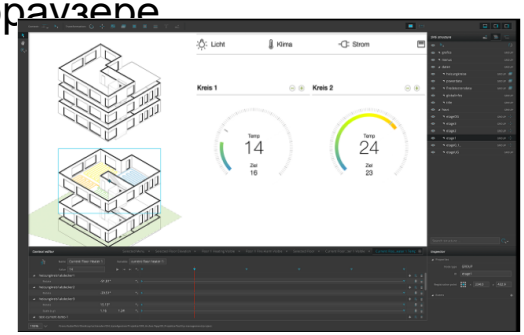
SVG редакторы

- профессиональный векторный графический редактор

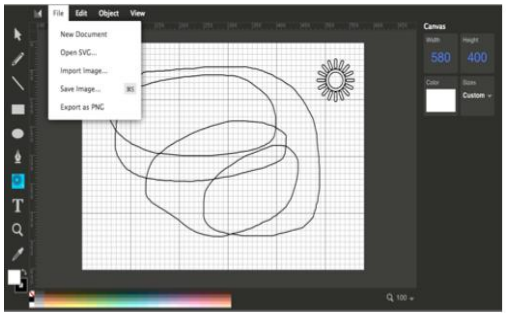


RollApp – Inscape в браузере

BoxySVG - с инспектором кода



Archer Editor – для больших проектов



Method Draw

Экспорт из - Adobe Illustrator, CorlDraw, Scribus, Microsoft Visio …