

## **Лабораторная работа № 14**

### **МАСШТАБИРУЕМАЯ ВЕКТОРНАЯ ГРАФИКА**

**Цель работы:** изучить способы вставки SVG-изображения на web-страницу, принципы создания svg-фигур и svg-контуров; познакомиться с правилами применения трансформации, градиентной заливки и анимации к svg-фигурам.

#### **Теоретические сведения для выполнения работы**

##### **Использование SVG**

Масштабируемая векторная графика (Scalable Vector Graphics, SVG) представляет собой вид графики, который создается с помощью математического описания геометрических примитивов (линий, кругов, эллипсов, прямоугольников, кривых), которые образуют изображение. Изображения SVG описываются тестовыми файлами с применением языка XML и предназначены для описание двухмерной векторной или смешанной векторно-растровой графики.

К преимуществам SVG-изображений относится:

1. Отсутствие потери качества при масштабировании.
2. Могут создаваться и редактироваться в любом текстовом редакторе.
3. Совместимость со стандартами консорциума W3C: DOM и XSL.
4. Размеры их файлов являются небольшими по сравнению с любым другим типом файлов изображений.
5. Можно добавлять несколько гиперссылок.
6. Поддержка скриптов и анимации в SVG позволяют создавать динамичную и интерактивную графику.

Преимущественно .svg используют в дизайне иконок, логотипов и элементов пользовательского интерфейса для веб-сайтов, а также можно создавать графики и диаграммы, простую инфографику, масштабируемые дорожные карты, легкие игры вроде sudoku или кроссвордов.

Существуют следующие способы использования svg в веб-бразерах:

1. Вставка SVG-файла в HTML-документ с помощью тегов `img`, `embed`, `object` и `iframe`.

```

<embed src="example.svg" type="image/svg+xml">
<object data="example.svg" type="image/svg+xml"></object>
<iframe src="example.svg" width="200" height="300"
style="border: none"></iframe>
```

#### 14.1 Способы вставки файла с расширением `svg`

2. Вставка кода в HTML-документ в элементе `<svg>...</svg>`:

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
<!-- SVG-код -->
</svg>
```

Рис. 14.2 Вставка векторной графики с помощью тега `<svg>`

3. подключение в PHP-документ с помощью функции `include`: `<? include("example.svg"); ?>`.

4. Использование SVG-файла в качестве фонового изображения в CSS: **`background: url(example.svg)`**.

Контейнер SVG имеет бесконечные размеры. *Viewport* и *viewBox* — это две прямоугольные области просмотра, которые ограничены конечными значениями высоты и ширины, указанными в атрибутах **`width`** и **`height`**. Область видимости *viewport* принимает значения атрибутов высоты и ширины, а *viewBox* дает возможность отобразить без искажений или трансформировать конкретный фрагмент SVG. Например, `<svg width="400" height="400" version="1.1" viewBox="0 0 800 800" xmlns="//www.w3.org/2000/svg">` определяет пользовательскую область просмотра *viewport* равной 400×400 px. Первые два значения атрибута *ViewBox* **`min-x`** и **`min-y`** определяют начало системы координат пользовательской области просмотра, последующие два значения ее ширину и высоту и одновременно масштабирование изображения. В примере пользовательская область занимает прямоугольный фрагмент размером 800×800 px, то есть область видимости *viewport* и дополнительно по 400px справа и снизу. Таким образом, масштаб видимости SVG-изображения уменьшается.

Так как *viewport* является предком для *viewBox*, то начало системы координат *viewBox*, по умолчанию, также как и системы координат *viewport* находится в левом верхнем углу (0,0) и положительное направление оси «X» – будет вправо, а оси «Y» – вниз.

## Основные элементы SVG

К основным элементам, которые могут быть созданы являются прямая линия, ломанная линия, многоугольник, прямоугольник, круг, эллипс, сложная траектория. Соответствующие им теги представлены в таблице 14.1

Таблица 14.1

Основные геометрические элементы svg

Элементы SVG	Атрибуты
line (прямая линия)	x1 — координата начальной точки линии по оси X; y1 — координата начальной точки линии по оси Y; x2 — координата конечной точки линии по оси X; y2 — координата конечной точки линии по оси Y
polyline (ломанная линия)	points — координаты ломанной линии парами x,y через пробел
polygon (многоугольник)	points — координаты ломанной линии парами x,y через пробел
rect (прямоугольник)	x — координата левой верхней точки прямоугольника по оси X; y — Координата левой верхней точки прямоугольника по оси Y; width — ширина прямоугольника; height — высота прямоугольника; rx — радиус закругления углов прямоугольника по оси X; ry — радиус закругления углов прямоугольника по оси Y;
circle (круг)	cx — координата центра круга по оси X; cy — координата центра круга по оси Y; r — радиус круга;
ellipse (эллипс)	cx — координата центра эллипса по оси X; cy — координата центра эллипса по оси Y; rx — радиус эллипса по оси X; ry — радиус эллипса по оси Y;

Создание сложной траектории осуществляются тегом *<path>*, который позволяет создавать произвольные фигуры. Форма фигу-

ры задается атрибутом *d*, значение которого — это набор специальных команд. Эти команды могут быть и в верхнем, и в нижнем регистре. Верхний регистр указывает на то, что применяется абсолютное позиционирование, а нижний — относительное. Список команд и их значений представлены в таблице 14.2, а пример на рис. 14.3.

Таблица 14.2

Значения элемента <path>

Команды тега <path>	Значение команды
M, m — начальная точка	mx, my — координаты точки
L, l — отрезок прямой	lx, ly — координаты от текущей точки линии к указанной
H, h — горизонтальная линия	hx — координата, до которой создается линия по оси X
V, v — вертикальная линия	vy — координата до которой создается линия по оси Y
A, a — дуга эллипса	rx,ry — радиусы дуги эллипса; x-axis-rotation — угол поворота дуги относительно оси X; large-arc-flag — если (=1), то строится большая часть дуги, если (=0) — меньшая; sweep-flag — если (=1), то дуга строится по часовой стрелке, если (=0) — против часовой стрелке; x,y — координаты конечной точки дуги
C, c — кубическая кривая Безье	x1,y1 — координаты первой контрольной точки; x2,y2 — координаты второй контрольной точки; x,y — координаты конечной точки кривой.
S, s — гладкая кубическая кривая Безье	x2,y2 — координаты второй контрольной точки; x,y — координаты конечной точки кривой. Первая контрольная точка является зеркальным отражением второй контрольной точки
Q, q — квадратичная кривая Безье	x1,y1 — координаты контрольной точки; x,y — координаты конечной точки кривой.
T, t — гладкая квадратичная кривая Безье	x,y — координаты конечной точки кривой. Контрольная точка этой команды является зеркальным отражением контрольной точки предыдущей команды.
Z, z — замыкание траектории	не имеет значений

Сложные SVG фигуры можно нарисовать в векторных редакторах Adobe Illustrator, CorelDRAW, Inkscape (рекомендуемый сво-

бодный редактор SVG-графики) и сохранить в формате svg. Далее полученный документ открывается в Блокноте, FrontPage или любом другом редакторе, в окне которого будет представлен автоматически корректно созданный код. Данный код можно скопировать и вставить в HTML-документ.

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1"
width="600" height="100">
  <path d="M10,15 h50 v60 L110,55 A25,35 -30 0,1 150,30
    M160,50 C160,110 260,110 260,50 S360,-10 360,50
    M370,50 Q420,100 470,50 T570,50 z"
    stroke="#b4241b" stroke-width="3" fill="none" />
</svg>
```

Рис. 14.3 Пример создания сложной траектории

К общим атрибутам, которые могут быть во всех элементах, относятся:

1. *stroke* — цвет линии;
2. *stroke-width* — толщина линии;
3. *stroke-linecap* — стиль концов линии. Возможные значения атрибута: *round* – по форме круга; *square* – по форме квадрата;
4. *stroke-dasharray* — чередование штрихов и пробелов в пунктирной линии;
5. *fill* — цвет заливки (*none* – без заливки);
6. *fill-opacity* — прозрачность заливки (от 0 до 1);
7. *fill-rule* — правило заливки. Возможные значения атрибута: *nonzero* — сплошная заливка; *evenodd* – внутренняя часть фигуры не заливается.
8. *style* — стиль элемента;
9. *class* — класс элемента.

Преобразования задаются в атрибуте *transform*. Можно указать несколько преобразований через пробел. Виды трансформации:

- *rotate*(*rotate-angle* [*sx sy*]) – поворот;
- *scale*(*sx* [*sy*]) – масштабирование;
- *translate*(*tx* [*ty*]) – перенос;
- *skewX*(*skew-angle*) – наклон по оси X;
- *skewY*(*skew-angle*) – наклон по оси Y.

Для хранения повторно используемого содержимого используется тег `<defs>`. Содержимое этого тега является скрытым и будет использовано только при обращении к нему по *id*. В теге можно хранить, например, градиентную заливку (`linearGradient`, `radialGradient`) и применить ее к отдельным фигурам. Также можно хранить любые элементы SVG: `pattern`, `marker`, `path`, `gradient`, основные элементы SVG.

```
<defs>
  <linearGradient id = "MyGradient">
    <stop offset = "30%" stop-color = "red"/>
    <stop offset = "70%" stop-color = "yellow"/>
  </linearGradient>
</defs>
<rect x = "0" y = "0" width = "150" height="150" fill =
"url(#MyGradient)"/>
```

Рис. 14.4 Использование тега `<defs>`

Для объединения нескольких фигур в группу для последующих действий над ней, как над одним целым используется парный тег `<g>`. Группе так же может быть присвоен уникальный *id* для повторного использования. Несколько групп могут быть объединены в одну.

```
<g id="gr1-gr2" transform="translate(50,70)">
  <g id="gr1">
    <rect x="30" y="50" width="120" height="50" style="fill-
opacity: 0.7; fill: red;" />
    <rect x="30" y="140" width="120" height="50"
style="fill:yellow; stroke-width:3; stroke: blue;"/>
  </g>
  <g id="gr2">
    <rect x="30" y="230" width="120" height="50" style="fill:none;
stroke-width:3; stroke: blue;"/>
    <rect x="30" y="320" width="120" height="50" style="fill:none;
stroke-width:8; stroke: red; stroke-opacity:0.4;"/>
  </g></g>
```

Рис. 14.5 Пример использования тега `<g>`

Для создания копий SVG-фигур и их размещения на странице, а также добавления различных преобразований используется тег `<use>`, указывается `id` контура и прописываются его координаты, например, `<use xlink:href="#myCircle" x="10" fill="blue"/>`.

## Анимация SVG

SMIL (the Synchronized Multimedia Integration Language) – язык разметки на основе XML, с помощью которого осуществляется анимация. Каждой отдельной геометрической svg-фигуре можно присвоить анимации SMIL. Для этого используется непарный тег `<animate>`, который анимирует отдельные свойства, Свойства прописываются с указанием анимированного свойства в атрибуте `attributeName`. В примере на рис. 14. анимируется свойство `cx`, расположение по оси X изменяется от 100 до 300px за 5 секунд.

```
<circle cy="70" r="50" fill="red">
<animate attributeName="cx" from="100" to="300" dur="5s"/>
</circle>
```

Рис. 14.6 Пример использования анимации svg-фигуры

Можно задавать сразу несколько анимаций, и они будут выполняться одновременно. В теге `<animate>` можно ссылаться на анимируемый объект через его *id* с помощью атрибута `xlink:href`:

Для создания анимации трансформаций предназначен тег `<animateTransform>`, вид трансформации указывается в атрибуте *type*: `<animateTransform xlink:href="#mygroup" attributeName="transform" attributeType="XML" type="rotate" from="0, 60 50" to="45,60,50" dur="5s" additive="sum" fill="freeze"/>`.

Для обработки событий запуска анимации можно воспользоваться с атрибутами *begin* и *end*, например `begin="mouseover"` для начала анимации при наведении на элемент, `end="mouseout"` для завершения анимации при отводе курсора мыши.

## Работа с текстом

Текст в элементе SVG определяется с помощью тега `<text>`. К специфическим атрибутам, используемыми для работы с текстом относятся:

1.  $x$  и  $y$  – координаты расположения текста на экране: `<text x="0" y="20">Text</text>`

2.  $dx$  и  $dy$  – размещение текстовых областей относительно текущей позиции;

3. *text-anchor* – выравнивание текстовой строки относительно точки ( $x$ ,  $y$ ). Может принимать значения **start**, **middle**, **end**;

4. *rotate* – поворот текста на заданный угол;

5. *textLength* – устанавливает ширину строки;

6. *lengthAdjust* – сжатие и растягивание текста, используется вместе с атрибутом *textLength*. Может принимать значения **spacing** и **spacingAndGlyphs**.

Тег `<tspan>` в SVG аналогичен тегу `<span>`. Используется при необходимости применить стиль к определенной содержимого. Для ссылки на существующий текст можно воспользоваться тегом `<tref>`.

С помощью тега `<textPath>` осуществляется отображение текста вдоль направляющей линии. Пример использования `<textPath>` представлен на рис. 14.7

```
<defs>
<path id="idname" fill="..." stroke="..." d="..." />
</defs>
<use xlink:href="#idname" />
<text x="..." y="..." font-family="Arial">
<textPath xlink:href="#idname">
Write your text here.
</textPath>
</text>
```

Рис. 14.7 Пример использования тега `<textPath>`

Текст в SVG может быть стилизован с помощью свойств CSS, которые могут быть установлены как атрибуты.

## Задания к лабораторной работе № 14

**Задание 1** Создайте новый документ, в котором разместите текст вдоль произвольной кривой. Данный текст разместите по



центру, выделите произвольным цветом. Размер шрифта должен составлять 36px.

**Задание 2** Создайте в документе задания 1 элементы представленные на рис. 14.8

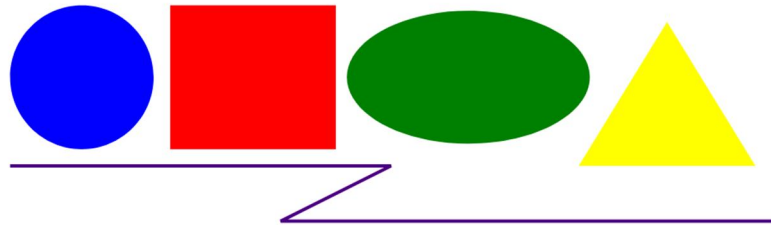


Рис. 14.8 Элементы для задания 2

**Задание 3** Сделайте элемент, представленный на рис. 14.9 используя графический редактор для работы с векторной графикой. Используя только тег `<path>` создайте несколько дополнительных фигур в виде украшений. К дополнительным фигурам можно применить различные анимации.

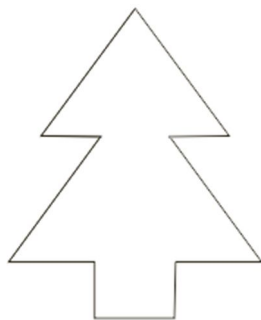


Рис. 14.9 Пример фигуры для задания 3

**Задание 4** Откройте `svgicon.html` файл из папки `labs`. Используйте любой `svg`-код иконки из файла и поместите в `<defs>`. Создайте 5 копий иконок и разместите их на странице. Примените к элементам различные трансформации.

**Задание 5** Сделав предварительно копию документа с элементами из задания 2 анимировать для них следующие свойства:

**Задание 5.1** Изменение для треугольника желтой заливки на линейную градиентную заливку от зеленого к оранжевому.

**Задание 5.2** Для эллипса сделать изменение заливки цветом при щелчке мыши на нем.

**Задание 5.3** Для квадрата сделать анимацию появления контура вокруг него.

**Задание 5.4** Для треугольника при наведении мыши изменение цвет контура.

**Задание 6** В копии задания 1 лабораторной работы № 2 внизу страницы создать четыре svg-фигуры в виде кругов радиуса 45px. Каждый из них должен быть содержать гиперссылку на задания из лаб. раб. № 2. Для копии документа измените ранее созданные CSS-стили, устаревшие атрибуты на SCSS.

### Контрольные вопросы

1. Дайте понятие SVG. Как расшифровывается аббревиатура?
2. Какие преимущества SVG перед остальными форматами?
3. Как использовать SVG в HTML?
4. Каким образом создать прямую линию и ломанную линию?
5. Каким образом создать прямоугольник и многоугольник?
6. Каким образом создать круг и эллипс?
7. Для чего предназначен тег `<path>`? Что означают значения в теге `<path>`?
8. Какие атрибуты относятся к общим?
9. Как создать заливку svg-фигуры?
10. Как изменить цвет и размер ширины контура svg-фигуры?
11. Каким образом трансформировать svg-фигуру?
12. Для чего используется тег `<use>`?
13. Каким образом использовать графические редакторы для создания svg?
14. Каким образом создать текст в svg?
15. Для чего используется тег `<defs>`?
16. Каким образом создать градиентную заливку?
17. Каким образом создать анимацию?
18. Какие атрибуты могут быть использованы при создании анимации?
19. Для чего используется `viewBox`?
20. Для чего используется тег `<g>`?
21. Создайте логотип компании Apple и браузера Google Chrome используя только тег `<path>`.