

Table of Contents

- Overview
- Shapes
- Text
- Stroke
- Filters
- Patterns
- Gradients
- Interactivity
- Linking



Overview

What is SVG?

- SVG, Scalable Vector Graphics is an XML based language to define vector based graphics.
- SVG is intended to display images over the web.
- Being vector images, SVG image never loses quality no matter how they are zoomed out or resized.
- SVG images supports interactivity and animation.
- SVG is a W3C standard.
- Others image formats like raster images can also be clubbed with SVG images.
- SVG integrates well with XSLT and DOM of HTML.

Advantages

- Use any text editor to create and edit SVG images.
- Being XML based, SVG images are searchable, indexable and can be scripted and compressed.
- SVG images are highly scalable as they never loses quality no matter how they are zoomed out or resized
- Good printing quality at any resolution
- SVG is an Open Standard

Disadvantages

- Being text format size is larger than compared to binary formatted raster images.
- Size can be big even for small image.

Example

Following XML snippet can be used to draw a circle in web browser.

```
<svg width="100" height="100">
  <circle cx="50" cy="50" r="40" stroke="red" stroke-width="2" fill="green"/>
</svg>
```

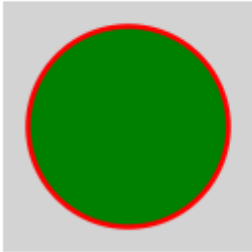
Embed the SVG XML directly in an HTML page.

01.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>SVG Image</title>
</head>
<body>
  <h1>Sample SVG Image</h1>
  <svg width="100" height="100" style="background: lightgray">
    <circle cx="50" cy="50" r="40" stroke="red" stroke-width="2" fill="green" />
  </svg>
</body>
</html>
```

Output

Sample SVG Image



How SVG integrates with HTML

- `<svg>` element indicates the start of SVG image.
- `<svg>` element's width and height attributes define the height and width of the SVG image.
- In above example, we've used a `<circle>` element to draw a circle.
- `cx` and `cy` attribute represents center of the circle. Default value is (0,0). `r` attribute represents radius of circle.
- Other attributes `stroke` and `stroke-width` controls the outlining of the circle.
- `fill` attribute defines the fill color of the circle.
- Closing `</svg>` tag indicates the end of SVG image.

Shapes

SVG provides number of shapes which can be used to draw images. Following are the common shapes.

No.	Shape Type & Description
1	rect Used to draw a rectangle.
2	circle Used to draw a circle.
3	ellipse Used to draw an ellipse.
4	line Used to draw a line.
5	polygon Used to draw a closed shape consisting of connected straight lines.
6	polyline Used to draw a open shape consisting of connected straight lines.
7	path Used to draw any path.

rect

Example

01rect.html

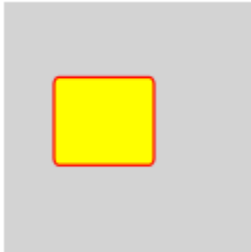
```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Rectangle</title>
</head>
<body>
  <!--
    x:      x-axis co-ordinate of top left of the rectangle. Default is 0
    y:      y-axis co-ordinate of top left of the rectangle. Default is 0
    width:  width of the rectangle
    height: height of the rectangle
    rx:     used to round the corner of the rounded rectangle
    ry:     used to round the corner of the rounded rectangle
    style:  used to specify inline styles
    class:  used to specify external style name to the element
  -->
  <svg width="100" height="100" style="background: lightgray">
    <rect x="20"
```

```

        y="30"
        width="40"
        height="35"
        rx="2"
        ry="3"
        style="stroke:red;fill:yellow;"
        class="myRect">
    </rect>
</svg>
</body>
</html>

```

Output



circle

Example

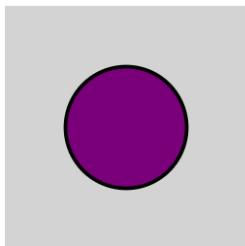
02circle.html

```

<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <title>Circle</title>
</head>
<body>
    <!--
        cx: x-axis co-ordinate of the center of the circle. Default is 0
        cy: y-axis co-ordinate of the center of the circle. Default is 0
        r:  radius of the circle
    -->
    <svg width="200" height="200" style="background: lightgray">
        <circle cx="100" cy="100" r="50" stroke="black"
            stroke-width="3" fill="rgb(121,0,121)"></circle>
    </svg>
</body>
</html>

```

Output



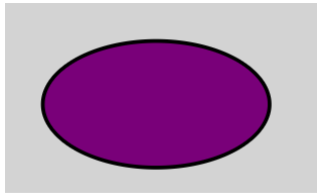
ellipse

Example

03ellipse.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Ellipse</title>
</head>
<body>
  <!--
    cx: x-axis co-ordinate of the center of the ellipse. Default is 0
    cy: y-axis co-ordinate of the center of the ellipse. Default is 0
    rx: x-axis radius of the ellipse
    ry: y-axis radius of the ellipse
  -->
  <svg width="250" height="150" style="background: lightgray">
    <ellipse cx="120" cy="80" rx="90" ry="50"
      stroke="black" stroke-width="3" fill="rgb(121,0,121)"></ellipse>
  </svg>
</body>
</html>
```

Output



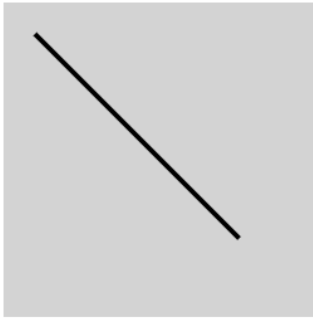
line

Example

04line.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Line</title>
</head>
<body>
  <!--
    x1: x-axis co-ordinate of the start point. Default is 0
    y1: y-axis co-ordinate of the start point. Default is 0
    x2: x-axis co-ordinate of the end point. Default is 0
    y2: y-axis co-ordinate of the end point. Default is 0
  -->
  <svg width="200" height="200" style="background: lightgray">
    <line x1="20" y1="20" x2="150" y2="150"
      stroke="black" stroke-width="3" fill="rgb(121,0,121)"></line>
  </svg>
</body>
</html>
```

Output



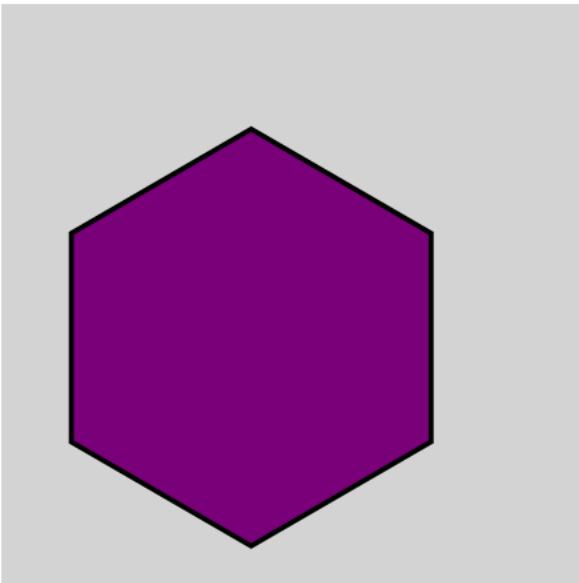
polygon

Example

05polygon.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Polygon</title>
</head>
<body>
  <!--
    points: List of points to make up a polygon
  -->
  <svg width="350" height="350" style="background: lightgray">
    <polygon points="150,75 258,137.5 258,262.5 150,325 42,262.6 42,137.5"
      stroke="black" stroke-width="3" fill="rgb(121,0,121)"></polygon>
  </svg>
</body>
</html>
```

Output



polyline

Example

06polyline.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Polyline</title>
</head>
<body>
  <!--
    points: List of points to make up a polyline
  -->
  <svg width="250" height="120" style="background: lightgray">
    <!-- Example of a polyline with the default fill -->
    <polyline points="0,100 50,25 50,75 100,0" />

    <!-- Example of the same polyline shape with stroke and no fill -->
    <polyline points="100,100 150,25 150,75 200,0"
      fill="none" stroke="black" />
  </svg>
</body>
</html>
```

Output



path

Example

07path.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Path</title>
</head>
<body>
  <!--
    d: path data, usually a set of commands like moveto, lineto etc.

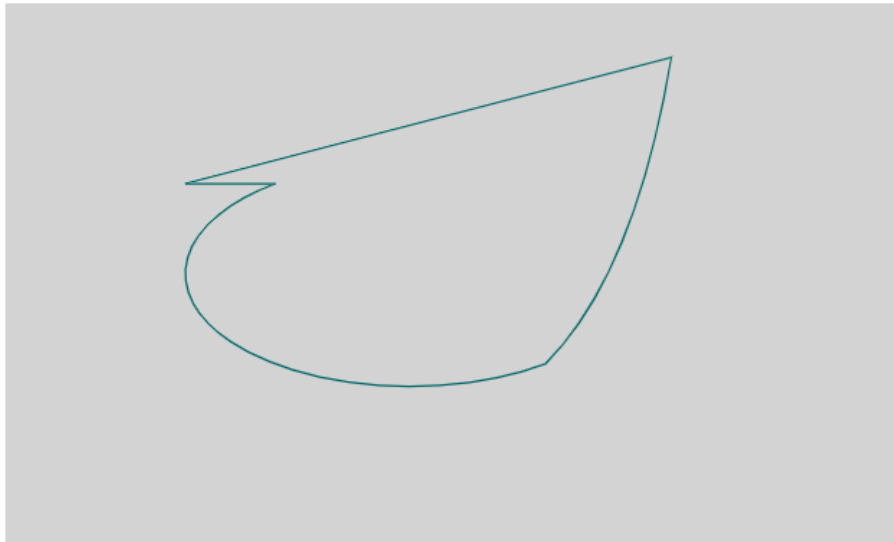
    Command & Description
    -----
    M - moveto - move from one point to another point
    L - lineto - create a line
  -->
</body>
</html>
```

```

H - horizontal lineto - create a horizontal line
V - vertical lineto - create a vertical line
C - curveto - create a curve
S - smooth curveto - create a smooth curve
Q - quadratic Bezier curve - create a quadratic Bezier curve
T - smooth quadratic Bezier curveto - create a smooth quadratic Bezier curve
A - elliptical Arc - create a elliptical arc
Z - closepath - close the path
-->
<svg width="500" height="300" style="background: lightgray">
  <path d="M100,100
    L150,100
    a50,25 0 0,0 150,100
    q50,-50 70,-170
    Z"
    style="stroke: #006666; fill: none;" />
</svg>
</body>
</html>

```

Output



Example

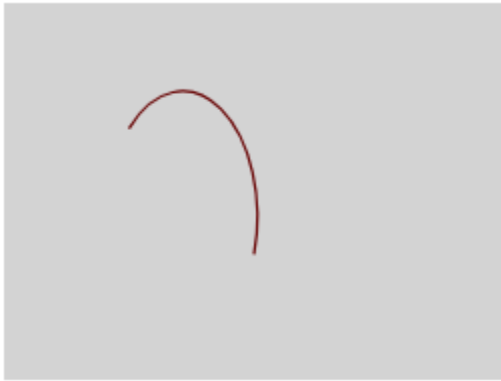
08arc.html

```

<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Path(Arcs)</title>
</head>
<body>
  <svg width="200" height="150" style="background: lightgray">
    <path d="M50,50
      A30,50 0 0,1 100,100"
      style="stroke:#660000; fill:none;" />
  </svg>
</body>
</html>

```

Output

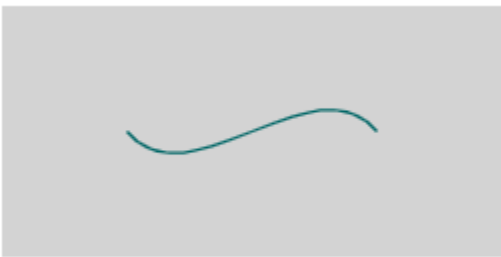


Example

09curve.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Path(Bezier Curves)</title>
</head>
<body>
  <svg width="200" height="100" style="background: lightgray">
    <path d="M50,50 C75,80 125,20 150,50"
          style="stroke: #006666; fill:none;" />
  </svg>
</body>
</html>
```

Output



Text

<text> element is used to draw text.

Declaration

Following is the syntax declaration of **<text>** element. We've shown main attributes only.

```
<text
  x="x-coordinates"
  y="y-coordinates"
  dx="list of lengths"
  dy="list of lengths"
  rotate="list of numbers"
  textlength="length"
  lengthAdjust="spacing" >
</text>
```

Attributes

No.	Attribute & Description
1	x – x axis coordinates of glyphs.
2	y – y axis coordinates of glyphs.
3	dx – shift along with x-axis.
4	dy – shift along with y-axis.
5	rotate – rotation applied to all glyphs.
6	textlength – rendering length of the text.
7	lengthAdjust – type of adjustment with the rendered length of the text.

Example

01.html

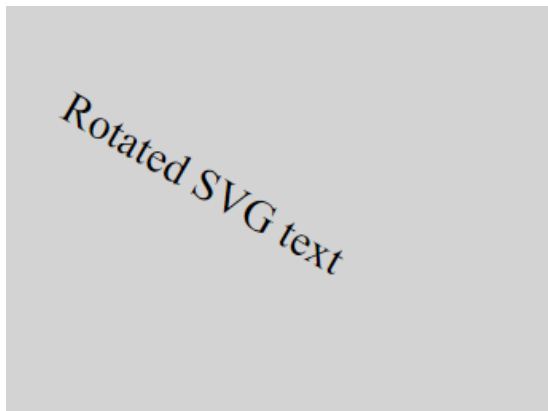
```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
  <title>Text</title>
</head>
<body>
  <svg width="200" height="50" style="background: lightgray">
    <text x="30" y="12">Text: </text>
    <text x="30" y="30" fill="rgb(121,0,121)"> www.google.com</text>
  </svg>
</body>
</html>
```

Text with rotate

02Text with rotate.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Text with rotate</title>
</head>
<body>
  <svg width="200" height="150" style="background: lightgray">
    <text x="20" y="40"
      transform="rotate(30 20,40)"
      style="stroke:none; fill:#000000;">Rotated SVG text</text>
  </svg>
</body>
</html>
```

Output

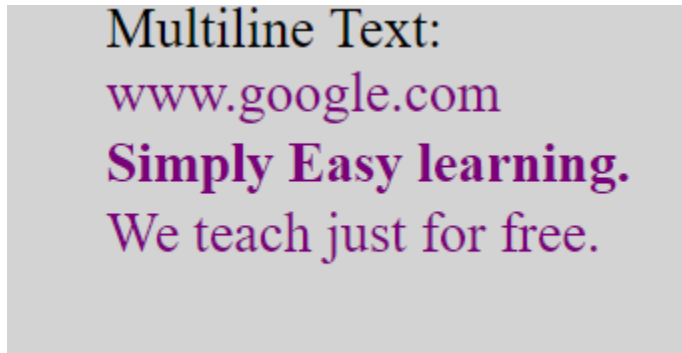


Multiline Text

03Multiline Text.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Multiline Text</title>
</head>
<body>
  <svg width="200" height="100" style="background: lightgray">
    <text x="30" y="12">Multiline Text: </text>
    <text x="30" y="30" fill="rgb(121,0,121)">
      www.google.com
    <tspan x="30" y="50" font-weight="bold">
      Simply Easy learning.
    </tspan>
    <tspan x="30" y="70">We teach just for free.</tspan>
  </text>
</svg>
</body>
</html>
```

Output



Hyper link Text

04Hyper link Text.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Hyper link Text</title>
</head>
<body>
  <svg width="250" height="100" style="background: lightgray">
    <text x="30" y="15">Text as Link: </text>
    <a xlink:href="http://www.google.com/" target="_blank">
      <text font-family="Verdana" font-size="20" x="30" y="30"
        fill="rgb(121,0,121)">www.google.com</text>
    </a>
  </svg>
</body>
</html>
```

Output



Stroke

SVG supports multiple stroke properties.

Following are the main stroke properties used.

No.	Stroke Type & Description
1	stroke – defines color of text, line or outline of any element.
2	stroke-width – defines thickness of text, line or outline of any element.
3	stroke-linecap – defines different types of ending of a line or outline of any path.
4	stroke-dasharray – used to create dashed lines.

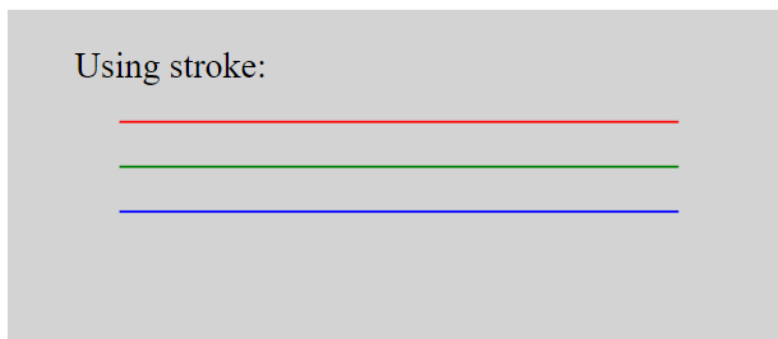
Example

01.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>SVG Stroke</title>
</head>
<body>
  <h1>Sample SVG Stroke</h1>
  <svg width="350" height="150" style="background: lightgray">
    <text x="30" y="30">Using stroke: </text>
    <path stroke="red" d="M 50 50 L 300 50" />
    <path stroke="green" d="M 50 70 L 300 70" />
    <path stroke="blue" d="M 50 90 L 300 90" />
  </svg>
</body>
</html>
```

Output

Sample SVG Stroke



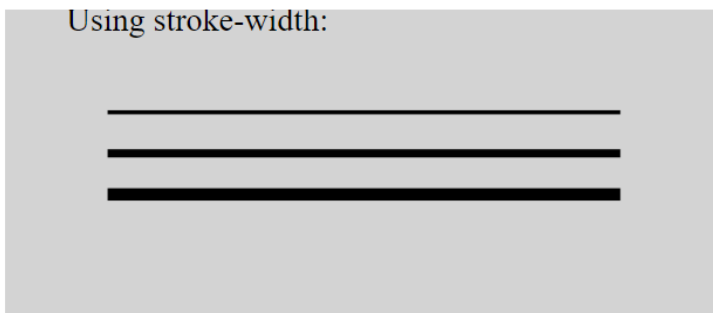
Stroke width

02Stroke width.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>SVG Stroke Width</title>
</head>
<body>
  <h1>Sample SVG Stroke</h1>
  <svg width="350" height="150" style="background: lightgray">
    <text x="30" y="10">Using stroke-width: </text>
    <path stroke-width="2" stroke="black" d="M 50 50 L 300 50" />
    <path stroke-width="4" stroke="black" d="M 50 70 L 300 70" />
    <path stroke-width="6" stroke="black" d="M 50 90 L 300 90" />
  </svg>
</body>
</html>
```

Output

Sample SVG Stroke



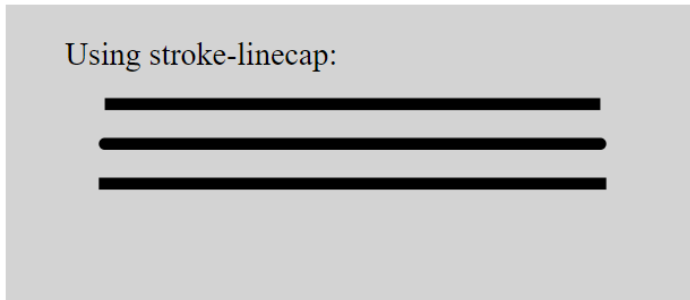
stroke-linecap

03stroke-linecap.html

```
<html>
  <title>SVG Stroke</title>
  <body>
    <h1>Sample SVG Stroke</h1>
    <svg width="800" height="800">
      <g>
        <text x="30" y="30">Using stroke-linecap: </text>
        <path stroke-linecap="butt" stroke-width="6"
              stroke="black" d="M 50 50 L 300 50" />
        <path stroke-linecap="round" stroke-width="6"
              stroke="black" d="M 50 70 L 300 70" />
        <path stroke-linecap="square" stroke-width="6"
              stroke="black" d="M 50 90 L 300 90" />
      </g>
    </svg>
  </body>
</html>
```

Output

Sample SVG Stroke



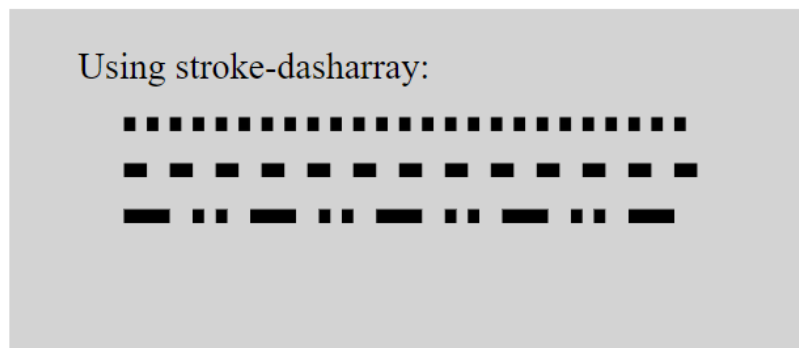
stroke-dasharray

04stroke-dasharray.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>SVG stroke-dasharray</title>
</head>
<body>
  <h1>Sample SVG Stroke</h1>
  <svg width="350" height="150" style="background: lightgray">
    <text x="30" y="30">Using stroke-dasharray: </text>
    <path stroke-dasharray="5,5" stroke-width="6"
          stroke="black" d="M 50 50 L 300 50" />
    <path stroke-dasharray="10,10" stroke-width="6"
          stroke="black" d="M 50 70 L 300 70" />
    <path stroke-dasharray="20,10,5,5,5,10" stroke-width="6"
          stroke="black" d="M 50 90 L 300 90" />
  </svg>
</body>
</html>
```

Output

Sample SVG Stroke



Filters

SVG uses <filter> element to define filters. <filter> element uses an id attribute to uniquely identify it. Filters are defined within <def> elements and then are referenced by graphics elements by their ids.

SVG provides a rich set of filters. Following is the list of the commonly used filters.

- feBlend
- feColorMatrix
- feComponentTransfer
- feComposite
- feConvolveMatrix
- feDiffuseLighting
- feDisplacementMap
- feFlood
- feGaussianBlur
- feImage
- feMerge
- feMorphology
- feOffset - filter for drop shadows
- feSpecularLighting
- feTile
- feTurbulence
- feDistantLight
- fePointLight
- feSpotLight

Declaration

Following is the syntax declaration of **<filter>** element. We've shown main attributes only.

```
<filter
  filterUnits="units to define filter effect region"
  primitiveUnits="units to define primitive filter subregion"

  x="x-axis co-ordinate"
  y="y-axis co-ordinate"

  width="length"
  height="length"

  filterRes="numbers for filter region"
  xlink:href="reference to another filter" >
</filter>
```

Attributes

No.	Name & Description
1	filterUnits – units to define filter effect region. It specifies the coordinate system for the various length values within the filter and for the attributes defining the filter subregion. If filterUnits="userSpaceOnUse", values represent values in the current user coordinate system in place at the time when the 'filter' element is used. If filterUnits="objectBoundingBox", values represent values in fractions or percentages of the bounding box on the referencing element in place at the time when the 'filter' element is used. Default is userSpaceOnUse.
2	primitiveUnits – units to define filter effect region. It specifies the coordinate system for the various length values within the filter and for the attributes defining the filter subregion. If filterUnits="userSpaceOnUse", values represent values in the current user coordinate system in place at the time when the 'filter' element is used. If filterUnits="objectBoundingBox", values represent values in fractions or percentages of the bounding box on the referencing element in place at the time when the 'filter' element is used. Default is userSpaceOnUse.
3	x – x-axis co-ordinate of the filter bounding box. Default is 0.
4	y – y-axis co-ordinate of the filter bounding box. Default is 0.
5	width – width of the filter bounding box. Default is 0.
6	height – height of the filter bounding box. Default is 0.
7	filterRes – numbers representing filter regions.
8	xlink:href – used to refer to another filter.

Example

01.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>SVG Filter</title>
</head>
<body>
  <h1>Sample SVG Filter</h1>
  <svg width="200" height="150" style="background: lightgray">
    <defs>
      <filter id="filter1" x="0" y="0">
        <feGaussianBlur in="SourceGraphic" stdDeviation="8" />
      </filter>

      <filter id="filter2" x="0" y="0" width="200%" height="200%">
        <feOffset result="offOut" in="SourceAlpha" dx="20" dy="20" />
        <feGaussianBlur result="blurOut" in="offOut" stdDeviation="10" />
      </filter>
    </defs>
  </svg>
</body>
</html>
```

```

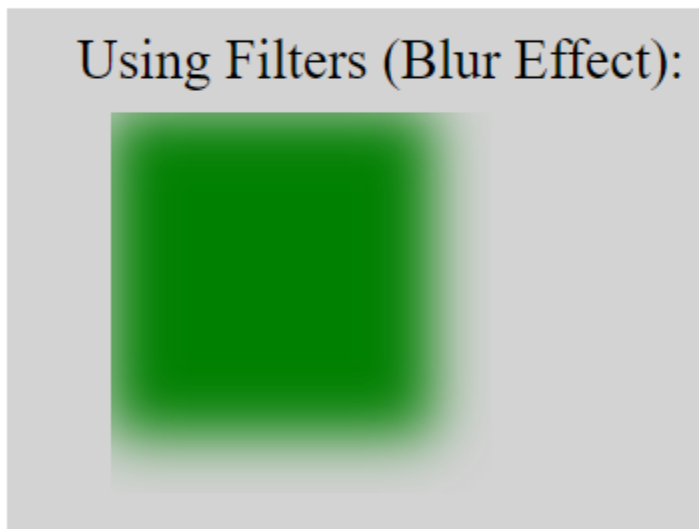
        <feBlend in="SourceGraphic" in2="blurOut" mode="normal" />
    </filter>
</defs>
<text x="20" y="20">Using Filters (Blur Effect): </text>
<rect x="30" y="30" width="90" height="90" stroke="green"
      stroke-width="3" fill="green" filter="url(#filter1)" />
</svg>
</body>
</html>

```

- Two <filter> elements defined as filter1 and filter2.
- feGaussianBlur filter effect defines the blur effect with the amount of blur using stdDeviation.
- in="SourceGraphic" defines that the effect is applicable for the entire element.
- feOffset filter effect is used to create shadow effect. in="SourceAlpha" defines that the effect is applicable for the alpha part of RGBA graphics.
- <rect> elements linked the filters using filter attribute.

Output

Sample SVG Filter



Filter with Shadow effect

02Filter with Shadow effect.html

```

<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>SVG Filter</title>
</head>
<body>
  <h1>Sample SVG Filter</h1>
  <svg width="250" height="150" style="background: lightgray">

```

```

<defs>
  <filter id="filter1" x="0" y="0">
    <feGaussianBlur in="SourceGraphic" stdDeviation="8" />
  </filter>

  <filter id="filter2" x="0" y="0" width="200%" height="200%">
    <feOffset result="offOut" in="SourceAlpha" dx="20" dy="20" />
    <feGaussianBlur result="blurOut" in="offOut" stdDeviation="10" />
    <feBlend in="SourceGraphic" in2="blurOut" mode="normal" />
  </filter>
</defs>

<text x="20" y="20">Using Filters (Shadow Effect): </text>
<rect x="30" y="30" width="90" height="90" stroke="green"
      stroke-width="3" fill="green" filter="url(#filter2)" />
</svg>
</body>
</html>

```

Output

Sample SVG Filter



Patterns

SVG uses <pattern> element to define patterns. Patterns are defined using <pattern> element and are used to fill graphics elements in tiled fashion.

Declaration

Following is the syntax declaration of <pattern> element. We've shown main attributes only.

```
<pattern
  patternUnits="units to define x,y, width and height attributes."
  patternContentUnits="units to define co-ordinate system of contents of
pattern"
  patternTransform = "definition of an additional transformation from the
pattern coordinate system onto the target coordinate system"

  x="x-axis co-ordinate"
  y="y-axis co-ordinate"

  width="length"
  height="length"

  preserveAspectRatio="to preserve width/height ratio of original content"
  xlink:href="reference to another pattern" >
</pattern>
```

Attributes

No.	Name & Description
1	patternUnits – units to define patterns effect region. It specifies the coordinate system for the various length values within the pattern and for the attributes defining the pattern subregion. If patternUnits="userSpaceOnUse", values represent values in the current user coordinate system in place at the time when the 'pattern' element is used. If patternUnits="objectBoundingBox", values represent values in fractions or percentages of the bounding box on the referencing element in place at the time when the 'pattern' element is used. Default is userSpaceOnUse.
2	patternContentUnits – units to define pattern content region. It specifies the coordinate system for the various length values within the pattern and for the attributes defining the pattern subregion. If patternContentUnits="userSpaceOnUse", values represent values in the current user coordinate system in place at the time when the 'pattern' element is used. If patternContentUnits="objectBoundingBox", values represent values in fractions or percentages of the bounding box on the referencing element in place at the time when the 'pattern' element is used. Default is userSpaceOnUse.
3	x – x-axis co-ordinate of the pattern bounding box. Defeault is 0.
4	y – y-axis co-ordinate of the pattern bounding box. Default is 0.

5	width – width of the pattern bounding box. Default is 0.
6	height – height of the pattern bounding box. Default is 0.
7	preserveAspectRatio - to preserve width/height ratio of original content.
8	xlink:href – used to refer to another pattern.

Example

01.html

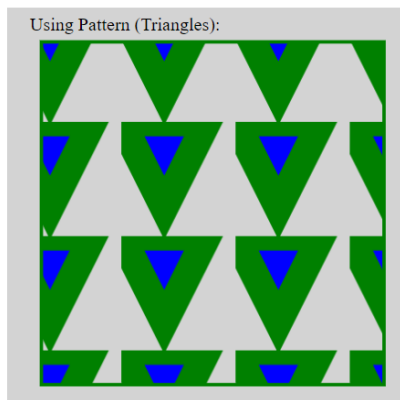
```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>SVG Pattern</title>
</head>
<body>
  <h1>Sample SVG Pattern</h1>
  <svg width="350" height="350" style="background: lightgray">
    <defs>
      <pattern id="pattern1" patternUnits="userSpaceOnUse" x="0" y="0"
        width="100" height="100" viewBox="0 0 4 4">
        <path d="M 0 0 L 3 0 L 1.5 3 z" fill="blue" stroke="green" />
      </pattern>
    </defs>

    <text x="20" y="20">Using Pattern (Triangles): </text>
    <rect x="30" y="30" width="300" height="300" stroke="green"
      stroke-width="3" fill="url(#pattern1)" />
  </svg>
</body>
</html>>
```

- One <pattern> element defined as pattern1.
- In pattern, a viewBox is defined and a path which is to be used as pattern is defined.
- in rect element, in fill attribute, url of the pattern is specified to fill the rectangle with pattern created earlier.

Output

Sample SVG Pattern



Gradients

Gradient refers to smooth transition of one color to another color within a shape. SVG provides two types of gradients.

- **Linear Gradients** – Represents linear transition of one color to another from one direction to another.
- **Radial Gradients** – Represents circular transition of one color to another from one direction to another.

Linear Gradients

Declaration

Following is the syntax declaration of **<linearGradient>** element. We've shown main attributes only.

```
<linearGradient
  gradientUnits = "units to define co-ordinate system of contents of gradient"
  gradientTransform = "definition of an additional transformation from the
gradient coordinate system onto the target coordinate system"

  x1="x-axis co-ordinate"
  y1="y-axis co-ordinate"
  x2="x-axis co-ordinate"
  y2="y-axis co-ordinate"

  spreadMethod="indicates method of spreading the gradient within graphics
element"
  xlink:href="reference to another gradient" >
</linearGradient>
<linearGradient
  gradientUnits = "units to define co-ordinate system of contents of gradient"
  gradientTransform = "definition of an additional transformation from the
gradient coordinate system onto the target coordinate system"

  x1="x-axis co-ordinate"
  y1="y-axis co-ordinate"
  x2="x-axis co-ordinate"
  y2="y-axis co-ordinate"

  spreadMethod="indicates method of spreading the gradient within graphics
element"
  xlink:href="reference to another gradient" >
</linearGradient>
```

Attributes

No.	Name & Description
1	gradientUnits – units to define the coordinate system for the various length values within the gradient. If gradientUnits="userSpaceOnUse", values represent values in the current user coordinate system in place at the time when the gradient element is used. If patternContentUnits="objectBoundingBox", values represent values in fractions or percentages of the bounding box on the referencing element in place at the time when the gradient element is used. Default is userSpaceOnUse.
2	x1 – x-axis co-ordinate of the gradient vector. Default is 0.
3	y1 – y-axis co-ordinate of the gradient vector. Default is 0.
4	x2 – x-axis co-ordinate of the gradient vector. Default is 0.
5	y2 – y-axis co-ordinate of the gradient vector. Default is 0.
6	spreadMethod – indicates method of spreading the gradient within graphics element. Default is 'pad'.
7	xlink:href – used to refer to another gradient.

Example

01Linear Gradients.html

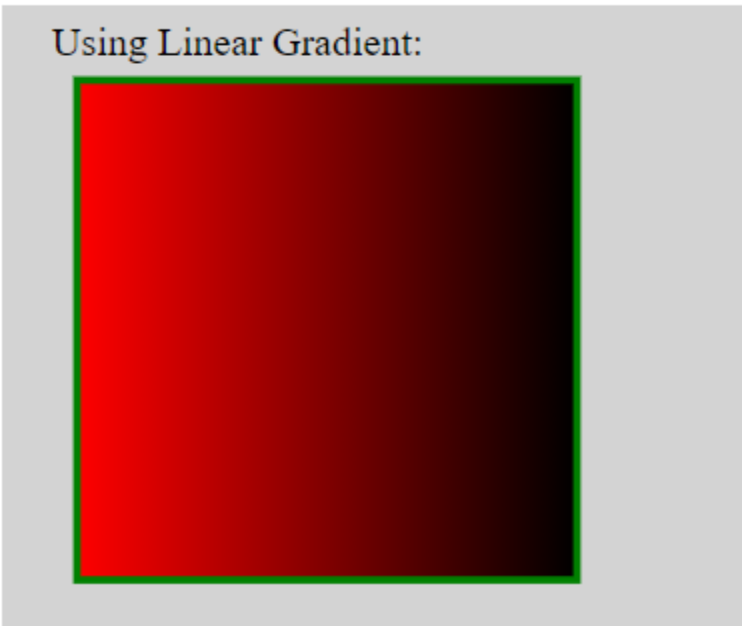
```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>SVG Linear Gradient</title>
</head>
<body>
  <h1>Sample SVG Linear Gradient</h1>
  <svg width="300" height="250" style="background: lightgray">
    <defs>
      <linearGradient id="sampleGradient">
        <stop offset="0%" stop-color="#FF0000" />
        <stop offset="100%" stop-color="#00FFFF" />
      </linearGradient>
    </defs>

    <text x="20" y="20">Using Linear Gradient: </text>
    <rect x="30" y="30" width="200" height="200" stroke="green"
          stroke-width="3" fill="url(#sampleGradient)" />
  </svg>
</body>
</html>
```

- One <linearGradient> element defined as sampleGradient.
- In linearGradient, two offsets are defined with two colors.
- in rect element, in fill attribute, url of the gradient is specified to fill the rectangle with gradient created earlier.

Output

Sample SVG Linear Gradient



Radial Gradients

Declaration

Following is the syntax declaration of **<radialGradient>** element. We've shown main attributes only.

```
<radialGradient
  gradientUnits = "units to define co-ordinate system of contents of gradient"
  gradientTransform = "definition of an additional transformation from the
gradient coordinate system onto the target coordinate system"

  cx="x-axis co-ordinate of center of circle."
  cy="y-axis co-ordinate of center of circle."

  r="radius of circle"

  fx="focal point for the radial gradient"
  fy="focal point for the radial gradient"

  spreadMethod="indicates method of spreading the gradient within graphics
element"
  xlink:href="reference to another gradient" >
</radialGradient>
```

Attributes

No.	Name & Description
1	gradientUnits – units to define the coordinate system for the various length values within the gradient. If gradientUnits="userSpaceOnUse", values represent values in the current user coordinate system in place at the time when the gradient element is used. If patternContentUnits="objectBoundingBox", values represent values in fractions or percentages of the bounding box on the referencing element in place at the time when the gradient element is used. Default is userSpaceOnUse.
2	cx – x-axis co-ordinate of the center of largest circle of gradient vector. Default is 0.
3	cy – y-axis co-ordinate of the center of largest circle of gradient vector. Default is 0.
4	r – radius of the center of largest circle of gradient vector. Default is 0.
5	fx – focal point of radial gradient. Default is 0.
6	fy – focal point of radial gradient. Default is 0.
7	spreadMethod – indicates method of spreading the gradient within graphics element. Default is 'pad'.
8	xlink:href – used to refer to another gradient.

Example

02Radial Gradients.html

```

<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>SVG Radial Gradient</title>
</head>
<body>
  <h1>Sample SVG Radial Gradient</h1>
  <svg width="300" height="250" style="background: lightgray">
    <defs>
      <radialGradient id="sampleGradient">
        <stop offset="0%" stop-color="#FF0000" />
        <stop offset="100%" stop-color="#00FFFF" />
      </radialGradient>
    </defs>

    <text x="20" y="20">Using Radial Gradient: </text>
    <rect x="30" y="30" width="200" height="200" stroke="green"
      stroke-width="3" fill="url(#sampleGradient)" />
  </svg>
</body>
</html>

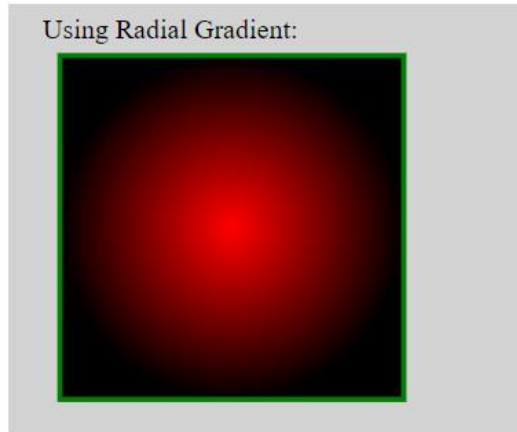
```

- One <radialGradient> element defined as sampleGradient.
- In radialGradient, two offsets are defined with two colors.

- in rect element, in fill attribute, url of the gradient is specified to fill the rectangle with gradient created earlier.

Output

Sample SVG Radial Gradient



Interactivity

SVG images can be made responsive to user actions. SVG supports pointer events, keyboard events and document events. Consider the following example.

Example

01.htm

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>SVG Interactivity</title>
  <script>
    function showColor() {
      alert("Color of the Rectangle is: " +
        document.getElementById("rect1").getAttributeNS(null, "fill"));
    }
    function showArea(event) {
      var w = parseFloat(event.target.getAttributeNS(null, "width"));
      var h = parseFloat(event.target.getAttributeNS(null, "height"));
      alert("Area of the rectangle is: " + w + " x " + h);
    }
    function showRootChildrenCount() {
      alert("Total Children: " +
        document.documentElement.childNodes.length);
    }
  </script>
</head>
<body>
  <h1>Sample Interactivity</h1>
  <svg width="300" height="300" style="background: lightgray">
    <text x="20" y="20"
      onclick="showColor()">Click me to show rectangle color.</text>

    <rect id="rect1" x="30" y="30" width="200" height="200"
      stroke="green" stroke-width="3" fill="red"
      onclick="showArea(event)" />

    <text x="30" y="250" onclick="showRootChildrenCount()">
      Click me to print child node count.
    </text>
  </svg>
</body>
</html>
```

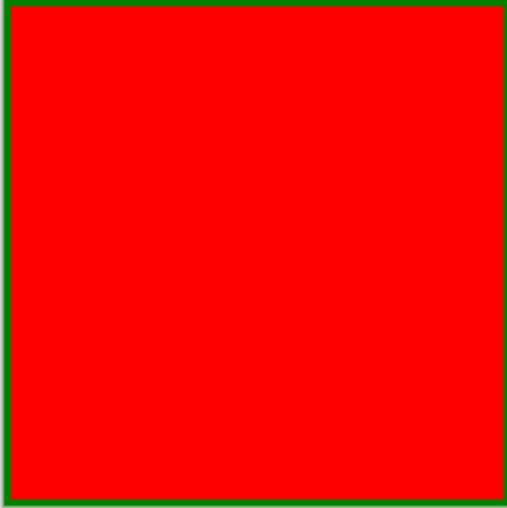
Explanation

- SVG supports JavaScript/ECMAScript functions. Script block is to be in CDATA block consider character data support in XML.
- SVG elements support mouse events, keyboard events. We've used onClick event to call a javascript functions.
- In javascript functions, document represents SVG document and can be used to get the SVG elements.
- In javascript functions, event represents current event and can be used to get the target element on which event got raised.

Output

Sample Interactivity

Click me to show rectangle color.



Click me to print child node count.

Linking

<a> element is used to create hyperlink. "xlink:href" attribute is used to pass the IRI (Internationalized Resource Identifiers) which is complementary to URI (Uniform Resource Identifiers).

Declaration

Following is the syntax declaration of <a> element. We've shown main attributes only.

```
<a
  xlink:show = "new" | "replace"
  xlink:actuate = "onRequest"
  xlink:href = "<IRI>"
  target= "_replace" | "_self" | "_parent" | "_top" | "_blank" | "<XML-Name>"
/>
```

Attributes

No.	Name & Description
1	xlink:show – for documentation purpose for XLink aware processors. Default is new.
2	xlink:actuate – for documentation purpose for XLink aware processors.
3	xlink:href – location of the referenced object.
4	target – used when targets for the ending resource are possible.

Example

01.htm

```
<html>
  <title>SVG Linking</title>
  <body>
    <h1>Sample Link</h1>
    <svg width="800" height="800">
      <g>
        <a xlink:href="http://www.google.com">
          <text x="0" y="15" fill="black" >
            Click me to load Google.com</text>
          </a>
        </g>

        <g>
          <text x="0" y="65" fill="black" >
            Click in the rectangle to load Google.com </text>
          <a xlink:href="http://www.google.com">
            <rect x="100" y="80" width="300" height="100"
              style="fill:rgb(121,0,121);stroke-width:3;stroke:rgb(0,0,0)"/>
          </a>
        </g>
      </svg>
    </body>
  </html>
```

Output

Sample Link

Click me to load Google.com

Click in the rectangle to load Google.com

