Anatomy of a Scalable Vector Graphic (SVG)

Basic Setup

```
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 420
250" fill="none" >
</svg>
```

You can also set the fill color

Closing Tag

XMLNS = XML Namespace. This one relates to SVGs.

viewBox = Sets the viewable area of the SVG. Four values: min-x, min-y, width, height.

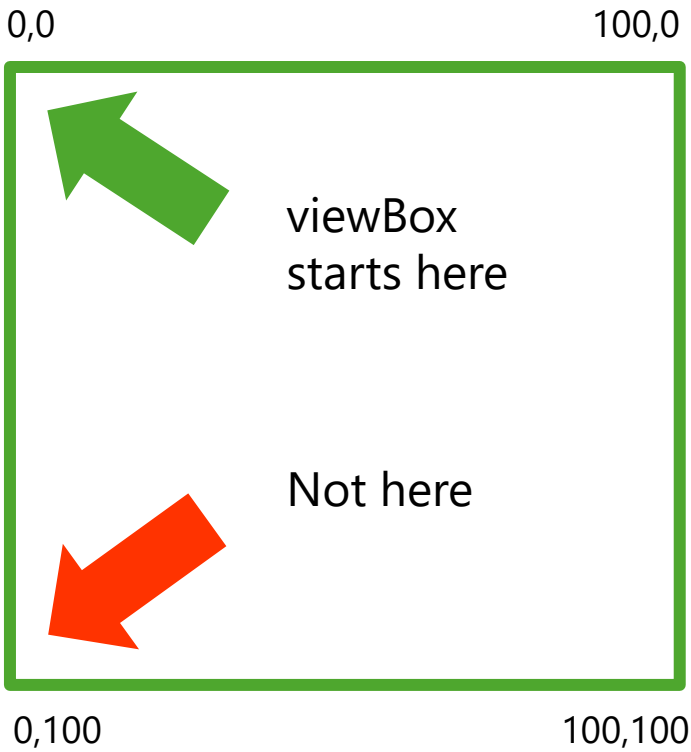
Note: You can also set the width and the height of entire SVG

Important: If you are making a SVG for Power BI, You will likely have to append "**data:image/svg+xml;utf8,**" to the beginning (unless using HTML Custom Viz).

What is the viewBox?

The viewBox is the viewable area for your SVG. The starting points (Min-X, Min-Y) begin in the top left corner, instead of bottom left corner.

This can be counterintuitive for Data Analysts, as we are used to Cartesian plane.



Anatomy of a Scalable Vector Graphic (SVG)

Rectangles

```
<svg ....>
    <rect x="50" y="20“ width = “300“ height = “100“
          style=“fill:black;” />
</svg>
```

Defined with rect tag

You can also include inline styling

Main attributes are x, y, width, and height

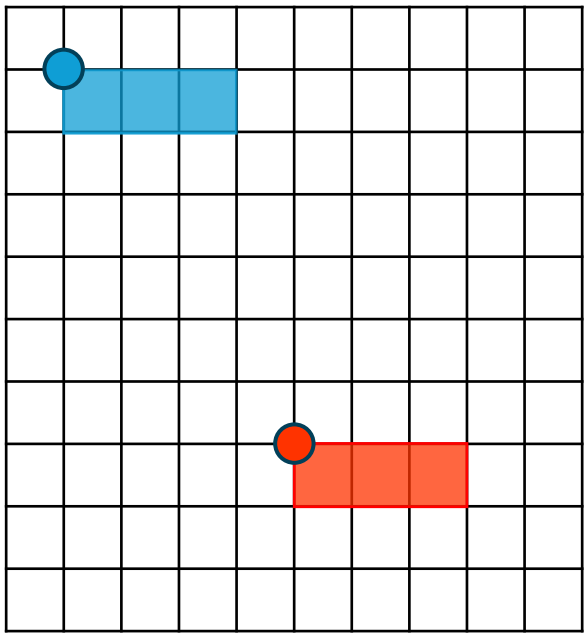
You may also define styling attributes like stroke and fill individually (e.g. stroke="black“)

Placing the Shape

Imagine each square in the grid to the right is 10 x 10 pixels.

Setting x=10 and y=10 would put the top left corner of the rectangle at the blue circle.

Setting x=50 and y=70 would put the top left corner of the rectangle at the red circle.



Anatomy of a Scalable Vector Graphic (SVG)

Circles

```
<svg ....>
<circle cx="50" cy="20“ r=“40”  style=“fill:black;” />
</svg>
```

Defined with circle tag

X coordinate for center

Y coordinate for center

Radius of circle

Inline style

You may also define styling attributes like stroke and fill individually (e.g. stroke="black“)

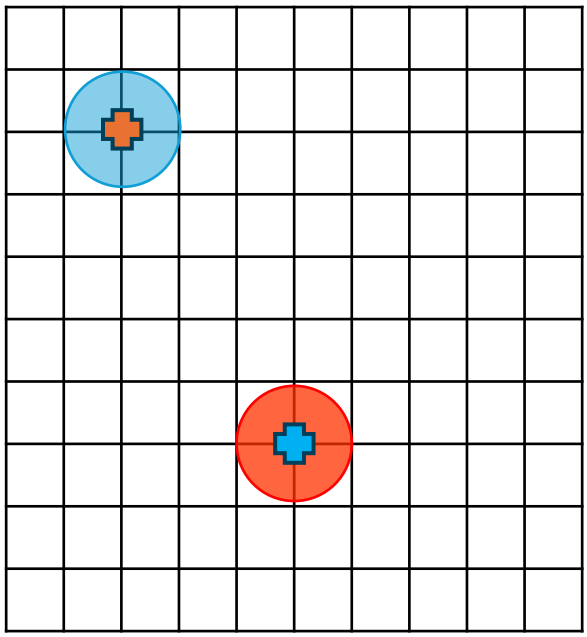
Placing the Shape

Imagine each square in the grid to the right is 10 x 10 pixels.

Setting cx=10 and cy=10 would put the center of the blue circle at the orange cross.

Setting cx=50 and cy=70 would put the center of the orange circle at the blue cross.

Both circles have a r=10. This means that 10 pixels will extend in all directions from the center to form the outer line of the circle.



Anatomy of a Scalable Vector Graphic (SVG)

Ellipses

```
<svg ....>
<ellipse cx="50" cy="20“ rx=“100” ry =“30” />
</svg>
```

- Defined with ellipse tag
- X coordinate for center
- Y coordinate for center
- Radius of width of ellipse
- Radius of height of ellipse

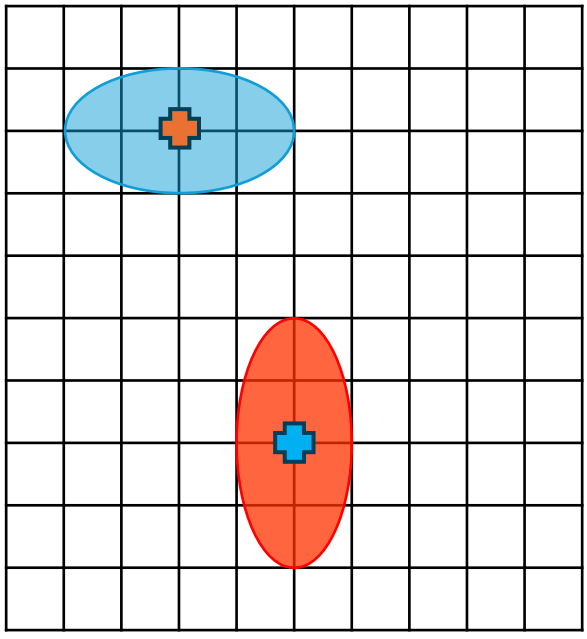
You may also define styling attributes like stroke and fill individually (e.g. stroke="black“) or together (e.g. style=“fill : black;”).

Placing the Shape

Imagine each square in the grid to the right is 10 x 10 pixels.

Setting cx=30 and cy=20 would put the center of the blue ellipse at the orange cross. Setting rx=20 and ry=10 would give the ellipse this shape.

Setting cx=50 and cy=70 would put the center of the orange ellipse at the blue cross. Setting rx = 10 and ry=20 would give the ellipse this shape.



Anatomy of a Scalable Vector Graphic (SVG)

Polylines

```
<svg ....>
    <polyline points="100,100 150,25 150,75 200,0"
              stroke="black"/>
</svg>
```

Defined with polyline tag

Color for the line

Points attribute is an array of x,y pairs

You may also define styling attributes like stroke and fill individually (e.g. stroke="black") or together (e.g. style="fill : black;").

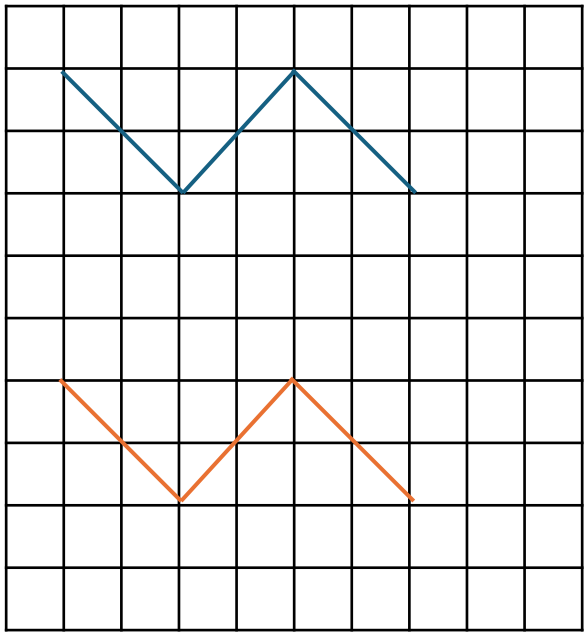
Placing the Shape

Imagine each square in the grid to the right is 10 x 10 pixels.

Polylines, at their simplest, are a set of instructions stating where to start and in order each point to connect to.

The **blue line** would be defined as:
points = "10,10 30,30 50,10 70,30"

The **orange line** would be defined as:
points = "10,60 30,80 50,60 70,80"



Anatomy of a Scalable Vector Graphic (SVG)

Polygons

```
<svg ....>
    <polygon points="0,100 50,25 50,75 100,0"
             stroke="black"/>
</svg>
```

Defined with polygon tag

Color for the polygon

Points attribute is an array of x,y pairs

You may also define styling attributes like stroke and fill individually (e.g. stroke="black") or together (e.g. style="fill : black;").

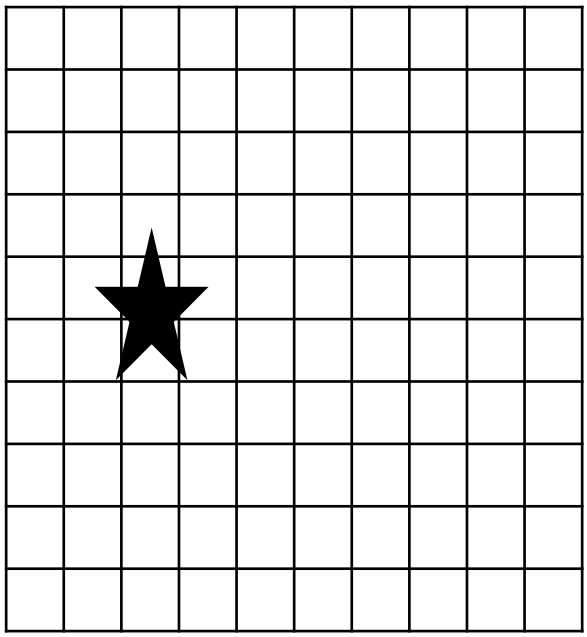
Placing the Shape

Imagine each square in the grid to the right is 10 x 10 pixels.

Polygons, at their simplest, are a set of instructions stating where to start and in order each point to connect to.

The main difference between polylines and polygons is that polygon points all connect.

A polygon with points="35,37.5 37.9,46.1 46.9,46.1 39.7,51.5 42.3,60.1 35,55 27.7,60.1 30.3,51.5 23.1,46.1 32.1,46.1" would roughly look like this.



Anatomy of a Scalable Vector Graphic (SVG)

Intro to Paths

```
<svg ....>
    <path d="M 10 10 L 100 10 Z"
          stroke="black"/>
</svg>
```

Defined with path tag

Color for the path

Basic attributes of M, L, and Z define the path.

M = makeline, or where to start from
L = lineto, or the next point in the path
Z = Signifies the end of the path

Absolute vs Relative

Capitalization matters when defining path attributes. M, L, Z means you are making absolute references, whereas m,l,z mean relative references.

For Excel champs, think of the difference between =B2 and =B2 when writing a formula.

<path d="M 10,10 L 20,20 30, 30 Z"/>

is the same as

<path d="m 10,10 l 10,10 10,10 z"/>

On this SVG, I set the stroke to black on one line and red on the other. The red line had an opacity of 0.5. The paths were set to the two examples to the left.

The red line was in the same path as the black line but defined with relative references.

Anatomy of a Scalable Vector Graphic (SVG)

Horizontal and Vertical Path Lines

```
<svg ....>
  <path d="M 10 10 L H 20Z" stroke="black"/>
</svg>
```

Defined with path tag

Using H for a horizontal line

Color for the path

H, h: A horizontal line
V, v: A vertical line
Note: Upper case is absolute, lower case is relative

Making Path Lines

Drawing path lines is simple and for Power BI users this would be handy for constant/reference lines.

Using horizonal as an example:

<path d= "M 10,10 H 20 Z"/>

Means

"Start at 10,10 and draw a line to 20,10" (absolute reference)

<path d="m 10,10 h 20 z"/>

Means

"Start at 10,10 and draw a line to 30, 10" (relative reference)

EXAMPLE

```
<svg xmlns="http://www.w3.org/2000/svg"
viewBox="0 0 100 100" >

<path d="M 10,10 H20 Z" stroke="black"/>
<path d="m 10,20 h20 z" stroke="red"/>

</svg>
```

Anatomy of a Scalable Vector Graphic (SVG)

Elliptical Arcs

```
<svg ....>
        <path d="M # # A rx ,ry x-axis-rotation large-arc, sweep x,y"
         stroke="black"/>
</svg>
```

WHERE

rx = x-radius
ry = y-radius
x-axis-rotation = ° of rotation

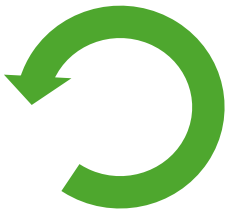
large-arc = if arc is >= 180, then 1 else 0
sweep = if positive direction then 1 else 0
x,y = end point

What is the Difference Between Sweep and Large-Arc?

Sweep and large-arc can be thought of as Boolean flags for direction and degrees of the ellipse.

Sweep of 1 means that the ellipse is drawn clockwise, while zero means that it is drawn counter-clockwise.

Large-arc of 1 means that the ellipse is 180 degrees or more, where 0 means it is less than 180 degrees.



sweep = 0

sweep = 1

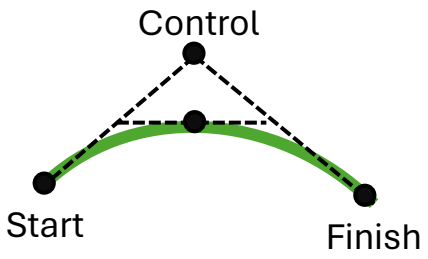
large-arc = 0

large-arc = 1

Anatomy of a Scalable Vector Graphic (SVG)

Quadratic Bézier Curves

Definition
Q x1 y1, x y where x1 y1 = control x y = start



What is a Quadratic Bézier Curve?

This type of Bézier curve is more straightforward than the Cubic Bézier Curve.

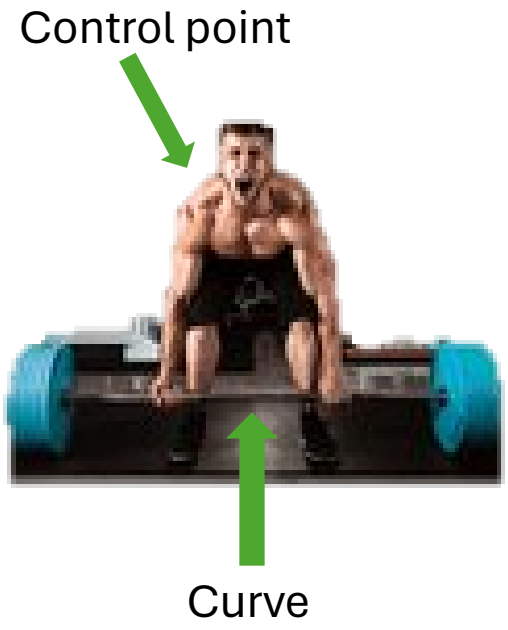
You could explain it this way:

“Given a START and CONTROL point, draw a curved line to the FINISH.”

The math behind it essentially finds the end by calculating where it is from the control point.

Think of the control point as pulling the line towards it. This “creates” the curve.

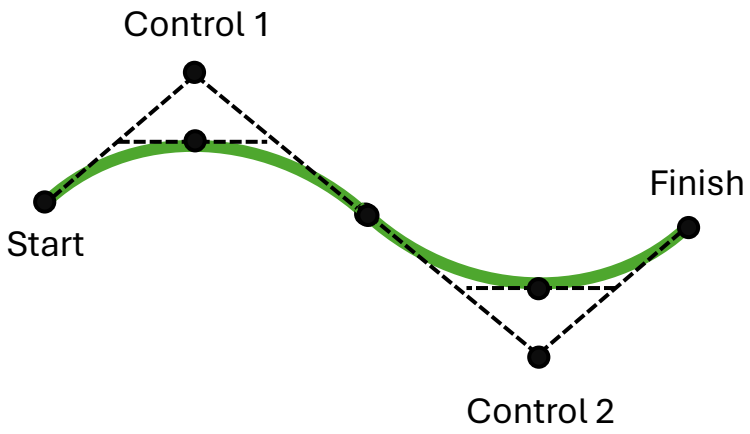
To the right is a bad analogy. Gym bro is the control point, deadlifting the ellipse and making the curve. No way the lunk alarm does not go off.



Anatomy of a Scalable Vector Graphic (SVG)

Cubic Bézier Curves

Definition
C x1 y1, x2 y2, x y where x1 y1 = control 1 x2 y2 = control 2 x y = start

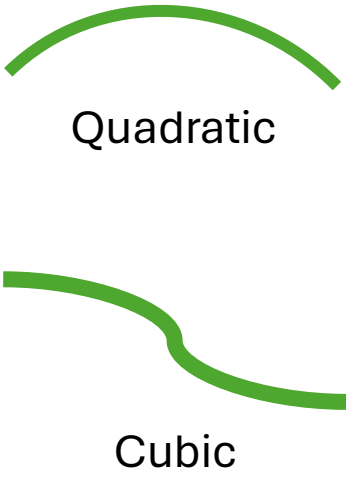


How is a Cubic Bézier Curve Different from a Quadratic?

The difference between a quadratic curve and a cubic curve is that a quadratic curve only has one bend.

Cubic curves can have multiple and in different directions.

Because cubic curves can have multiple bends, you must define multiple control points.



Anatomy of a Scalable Vector Graphic (SVG)

Grouping and Referencing

Groups

Definition

```
<g id="shape" ...>
        <rect ... />
        <circle ... />
</g>
```

What Are Groups?

Groups are simply a way to "group" one or more shapes, lines, paths, etc.

<defs> and <use>

Definition

```
<defs>
    <g id="shape"... />
</defs>

<use
xlink:href="#shape" ...
/>
```

What Are These Tags?

Defs and **use** allow you to "define" something you want to "use" more than once.

In the example to the left, by using **defs** with a **group**, I can define a group of shapes that I want to use later.

With grouping and referencing, you can make your code shorter, easier-to-read, and easier-to-maintain long-term.

You can also apply CSS rules to groups!

Anatomy of a Scalable Vector Graphic (SVG)

Transformations: Translate

What is the Translate Transformation?

Translate is a transform attribute that visually moves an element.

What is the syntax for Translate?

transform="translate(x,y)"

Example of Using Translate With <defs> and <use>

```svg
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 200" fill="none">
  <defs>
    <g id="rectangle">
      <rect x="10" y="10" width="25" height="25" rx="5" fill="#4EA72E"/>
    </g>
  </defs>
  <use href="#rectangle" transform="translate(60,60)"/>
  <use href="#rectangle" transform="translate(80,80)"/>
  <use href="#rectangle" transform="translate(100,100)"/>
  <use href="#rectangle" transform="translate(120,80)"/>
  <use href="#rectangle" transform="translate(140,60)"/>
  <use href="#rectangle" transform="translate(160,40)"/>
  <use href="#rectangle" transform="translate(180,20)"/>
</svg>
```

Anatomy of a Scalable Vector Graphic (SVG)

Transformations: Scale

What is the Scale Transformation?

The **scale** transformation resizes an x-value and y-value based on one or more multipliers.

What is the syntax for Scale?

transform="scale(x)"
or
transform="scale(x, y)"

Example of Using Scale With <defs> and <use>

```svg
<svg xmlns="http://www.w3.org/2000/svg" width="400" height="200" viewBox="0 0 400 200" fill="none
  <defs>
    <rect id="square" width="20" height="20" fill="#F97316"/>
  </defs>

  <!-One scale multiplied against x and y-!>
  <use href="#square" transform="translate(60,60) scale(1)"/>
  <use href="#square" transform="translate(100,100) scale(1.25)"/>
  <use href="#square" transform="translate(150,150) scale(1.5)"/>

  <!-Individual scales multiplied against repsective x and y-!>
  <use href="#square" transform="translate(200,100) scale(2, 1)"/>
  <use href="#square" transform="translate(250,50) scale(1, 2)"/>
  <use href="#square" transform="translate(300,20) scale(2, 2)"/>

</svg>
```

Anatomy of a Scalable Vector Graphic (SVG)

Transformations: Rotate

What is the Rotate Transformation?

The **rotate** transformation allows you to rotate something a certain number of degrees around a point.

What is the syntax for Rotate?

transform="rotate(degrees)"
or
transform="rotate(degrees,x, y)"

Example of Using Rotate With <defs> and <use>

```
<svg xmlns="http://www.w3.org/2000/svg" width="400" height="200" viewBox="0 0 400 200" fill="none">
  <defs>
    <rect id="square" width="50" height="50" fill="#F97316"/>
  </defs>

  <!-Using translate and rotate together to show rotation around a defined point-!>
  <use href="#square" transform="translate(50,50) rotate(45,75,75)"/>
  <use href="#square" transform="translate(75,75) rotate(70,100,100)"/>
  <use href="#square" transform="translate(25,100) rotate(70,50,125)"/>

</svg>
```

Anatomy of a Scalable Vector Graphic (SVG)

Transformations: skewX and skewY

What are the Skew Transformations?

skewX and **skewY** transformations stretch the figure either along the X or Y axis based on the respective attribute.

What is the syntax for Skew?

transform="skewX(degrees)"
or
transform="skewY(degrees)"

Example of Using Skew With <defs> and <use>

```svg
<svg xmlns="http://www.w3.org/2000/svg" width="400" height="200" viewBox="0 0 400 200" fill="none">
  <defs>
    <rect id="square" width="50" height="50" fill="#F97316"/>
  </defs>

  <!-Using translate and skews to show effects along different axes-!>
  <use href="#square" transform="translate(50,50)"/>
  <use href="#square" transform="translate(125,50) skewX(30)"/>
  <use href="#square" transform="translate(225,50) skewY(30)"/>

</svg>
```

Anatomy of a Scalable Vector Graphic (SVG)

Common Text Attributes for SVG

What are the Common Attributes?

Attribute	Definition
font-family	Style of font (e.g. Arial)
font-size	Size of font (pt, em, ex, %)
font-weight	Usually bold or normal
font-style	Usually italic or normal
text-decoration	none, underline, overline, line-through
word-spacing	+ to increase space between words, - to decrease
letter-spacing	+ to increase space between letters, - to decrease

How Do I Align My Text?

Alignment is handled through **text-anchor**.

text-anchor:start -> aligns to the beginning of the text

text-anchor:middle -> aligns to the middle of the text

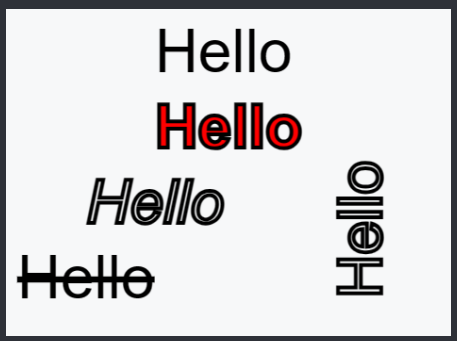
text-anchor:end -> aligns to the end of the text

Anatomy of a Scalable Vector Graphic (SVG)

Common Text Attributes for SVG

Examples of Commonly Used Attributes

```svg
<svg xmlns="http://www.w3.org/2000/svg" width="400" height="400" viewBox="0 0 400 400" fill="none">
  <defs>
    <text id="hello" style="font-family:Arial; font-size:12pt;">Hello</text>
  </defs>


  <!- visual examples to the right -!>


  <!-No major styling-!>
  <use href="#hello" fill="black" transform="translate(40,40)"/>

  <!-Red fill, black outline, bold, aligned to start-!>
  <use href="#hello"  fill="red" stroke="black" stroke-width="1px" transform="translate(40,60)" style="font-weight:bold; text-anchor:start;"/>

  <!-No fill, black outline, italicized, aligned to middle-!>
  <use href="#hello" fill="none" stroke="black" stroke-width="1px" transform="translate(40,80)" style="font-style:italic; text-anchor:middle;"/>

  <!-Black fill, no outline, strike-through, aligned to end-!>
  <use href="#hello"  fill="black" transform="translate(40,100)" style="text-decoration:line-through; text-anchor:end;"/>

    <!-No fill, black outline, rotated 270 degrees-!>
  <use href="#hello" fill="none" stroke="black" stroke-width="1px" transform="translate(100,100) rotate(270) "/>

</svg>
```



Anatomy of a Scalable Vector Graphic (SVG)

Animations

What is the <animate/> tag in SVG?

<animate/> allows you to add custom animations to visuals.

What are the basic attributes for <animate/>?

Attribute	Description
attributeName	What is being animated (e.g. height)
attributeType	Usually either XML or CSS, default XML if omitted
starting/ending	Starting and ending values
beginning/duration	When it begins and for how long
fill	What will happen once duration is done

Example of <animate/>

```
<svg xmlns="http://www.w3.org/2000/svg" width="200" height="100">
  <rect x="10" y="10" width="150" height="50" fill="#F97316">
    <animate
      attributeName="width"
      attributeType="XML"
      from="0" to="150"
      begin="0s" dur="1s"
      fill="freeze"
      />
  </rect>
</svg>
```

What does this do?

Animates a rectangle and makes it grow to full size (think animated bar charts)

Anatomy of a Scalable Vector Graphic (SVG)

Gradients

How do you use gradient colors with SVG?

<linearGradient> defines color stops along a linear axis
and
<radialGradient> defines stops that radiant outward from a point

Could you show me an example of using a gradient?

<linearGradient> example

```
<defs>
    <linearGradient id = "lg">
        <stop offset = "0%" style= "stop-
            color :#ff0000;"/>
        <stop offset = "50%" style= "stop-
            color :#ffffff;"/>
        <stop offset = "100%" style= "stop-
            color:#00ff00;"/>
     </linearGradient>
</defs>
<rect … style="fill:url(#lg);"/>
```

WHERE

offset = point when the color should equal the stop-color

stop-color = the color at the respective offset