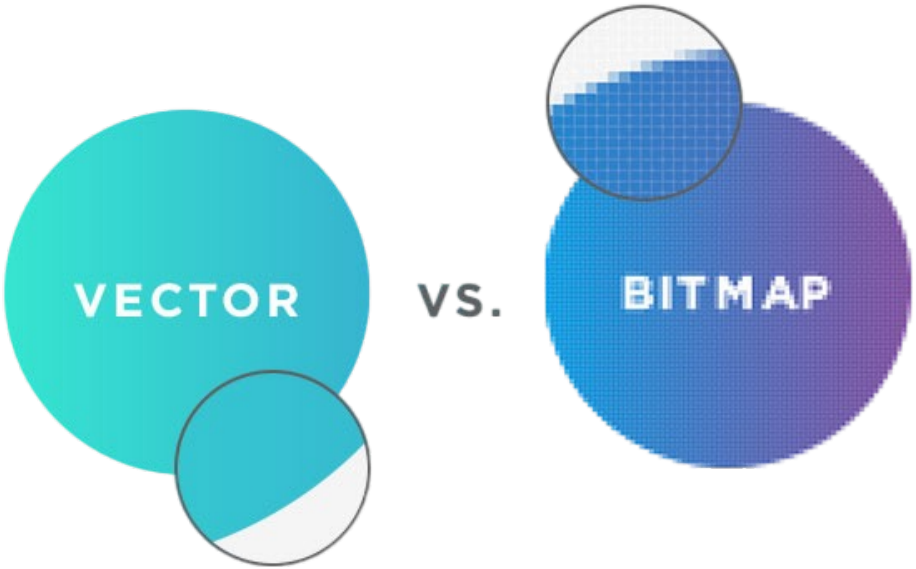
Deep Dive into

SVG

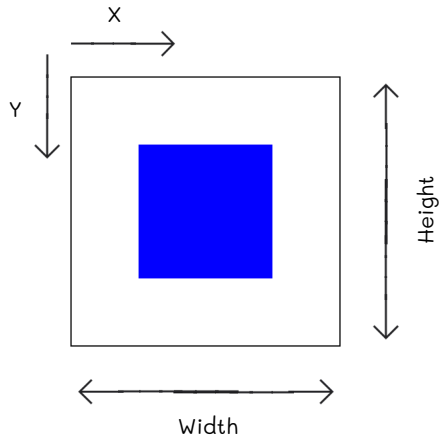
Mohsen Madani — Feb 2024

Why?



Rectangle

```svg
<svg
  width="200"
  height="200"
  style="margin: 100px;border: 1px solid;"
  viewBox="0 0 100 100"
>
  <rect width="50" height="50" y="25" x="25" fill="blue" />
</svg>
```



Path (Line)

MoveTo (M or m): Moves the "pen" to a new location without drawing a line.

M 10 10: Moves the pen to the coordinates (10, 10)

m 20 20: Moves the pen 20 units right and 20 units down from its current position

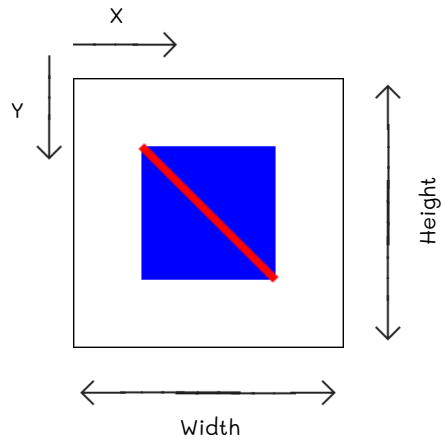
LineTo (L or l): Draws a line from the current position to a new position.

L 20 20: Draws a line from the current position to the coordinates (20, 20)

l 30 30: Draws a line 30 units right and 30 units down from the current position

Path (Line)

```svg
<svg
  width="200"
  height="200"
  style="margin: 100px;border: 1px solid;"
  viewBox="0 0 100 100"
>
  <rect width="50" height="50" y="25" x="25" fill="blue" />
  <path stroke-width="3" d="M 25 25 L 75 75" stroke="red" />
</svg>
```



Path (Line)

Horizontal LineTo (H or h): Draws a horizontal line from the current position to a new x-coordinate.

H 50: Draws a horizontal line from the current position to the x-coordinate 50

h 10: Draws a horizontal line 10 units to the right from the current position

Vertical LineTo (V or v): Draws a vertical line from the current position to a new y-coordinate.

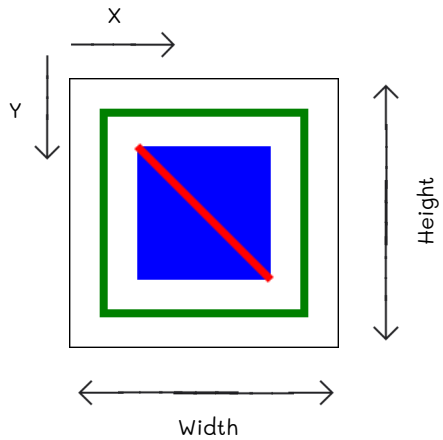
V 60: Draws a vertical line from the current position to the y-coordinate 60

v 20: Draws a vertical line 20 units down from the current position

ClosePath (Z or z): Closes the current subpath by drawing a straight line from the current position back to the starting point of the subpath.

Path (Line)

```svg
<svg
  width="200"
  height="200"
  style="margin: 100px;border: 1px solid;"
  viewBox="0 0 100 100"
>
  <rect width="50" height="50" y="25" x="25" fill="blue" />
  <path stroke-width="3" d="M 25 25 L 75 75" stroke="red" />
  <path
    stroke-width="3"
    stroke="green"
    d="M 12.5 12.5 H 87.5 V 87.5 L 12.5 87.5 Z"
    fill="transparent"
  />
</svg>
```

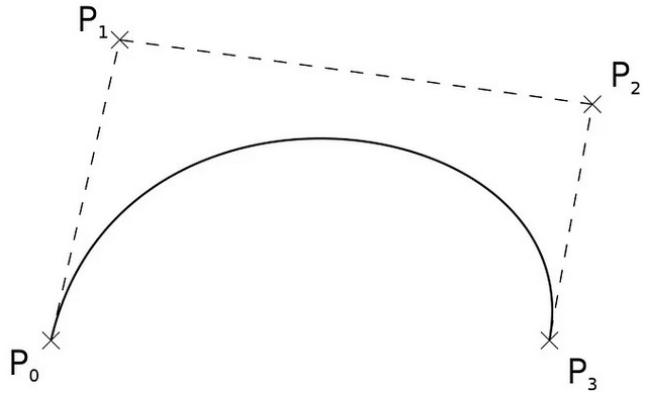


Path (Curves)

- Bézier Curves
  - Cubic
  - Quadratic
- Arcs

Bézier Curves

Bézier curves are parametric curves (with the parameter t varying from 0 to 1) that are defined by a set of **control points**.



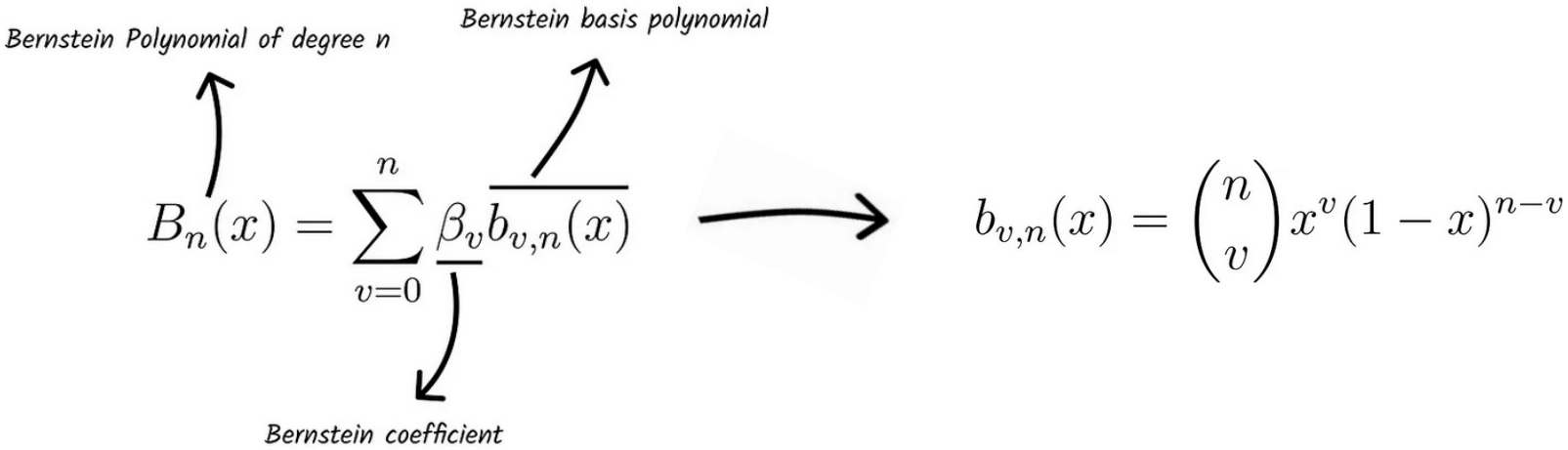
A bezier curve with 4 control points (cubic curve)

Bézier Curves

The curve goes from the "**Start**" point to the "**End**" point while the "**Control**" points define its curvature.

Bernstein Polynomials

A Bernstein polynomial of degree *n* is defined as a sum of **Bernstein basis polynomials**, each multiplied by a **Bernstein coefficient**.



Bernstein Coefficient

It's a Bernstein polynomial where the Bernstein coefficients are the **control points**. So, when building a bézier curve, we're essentially approximating a **real function**!

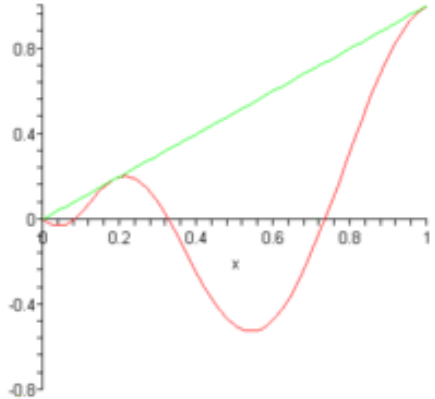
Let ***f*** be a continuous function on the interval [0, 1]. Consider the Bernstein polynomial

$$B_n(f)(x) = \sum_{\nu=0}^{n} f\left(\frac{\nu}{n}\right) b_{\nu,n}(x).$$

It can be shown that

$$\lim_{n\to\infty} B_n(f) = f$$

uniformly on the interval [0, 1]



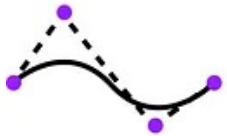
Bézier Curves

Linear

$$P_0 \cdot (1-t) + P_1 \cdot t$$

Quadratic

$$(1-t)^2 P_0 + 2(1-t)tP_1 + t^2 P_2$$



Cubic

$$(1-t)^3 P_0 + 3(1-t)^2 tP_1 + 3(1-t)^2 P_2 + t^3 P_3$$

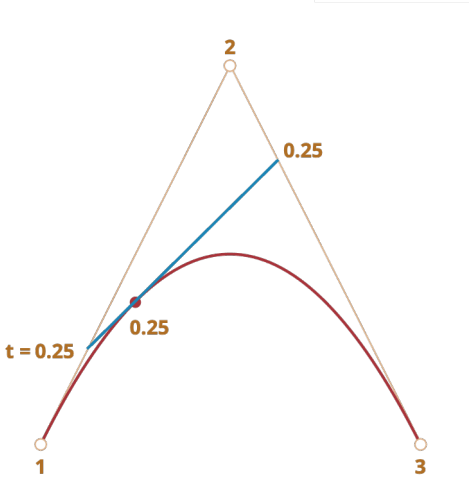
Recursive Bézier Curves

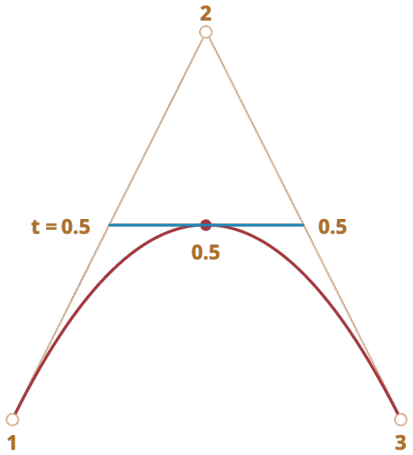
$$
\begin{aligned}
& (1-t)^2 P_0 + 2(1-t)tP_1 + t^2 P_2 \\
= \; & (1-t) \cdot \underbrace{[(1-t)P_0 + tP_1]}_{\text{Line that connects P0 and P1}} + t \cdot \underbrace{[(1-t)P_1 + tP_2]}_{\text{Line that connects P1 and P2}}
\end{aligned}
$$

Line that connects lines P0_P1 and P1_P2

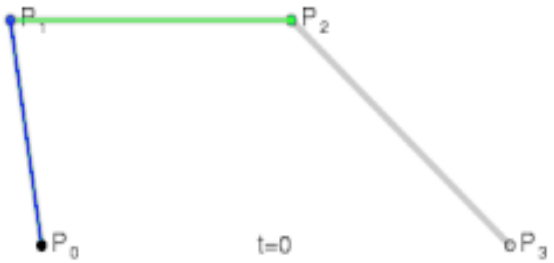
$$B_{P_0,P_1,\ldots,P_n}(t) = (1-t)B_{P_0,\ldots,P_{n-1}}(t) + tB_{P_1,\ldots,P_n}(t)$$

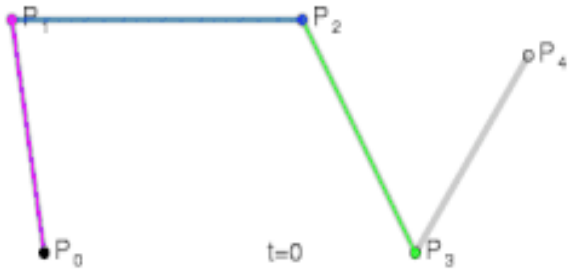
De Casteljau's algorithm



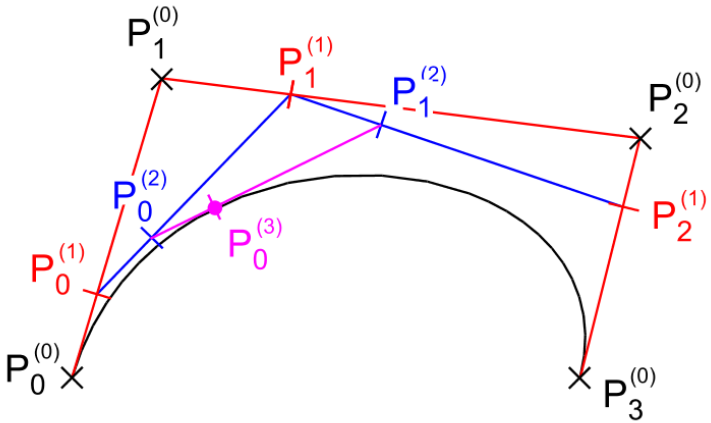


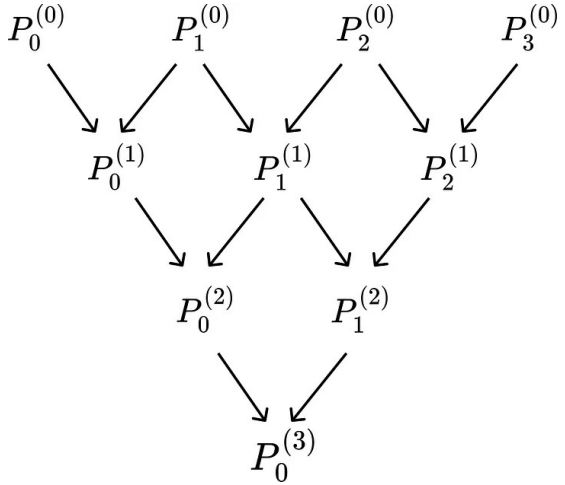
De Casteljau's algorithm





De Casteljau's algorithm



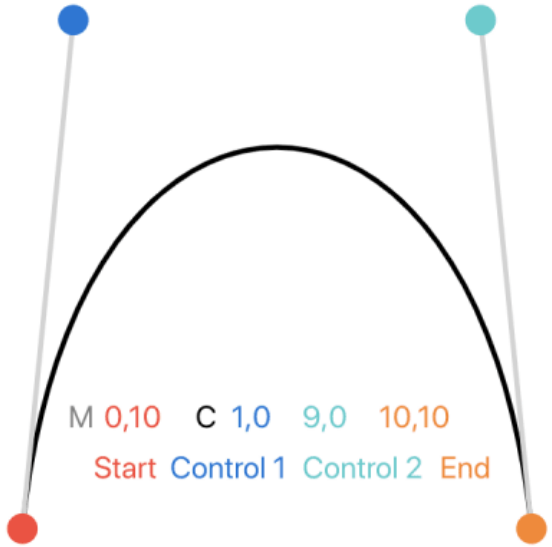


$$P_i^{(j)} = (1-t)P_i^{j-1} + tP_{i+1}^{j-1}$$

Path (Curves)

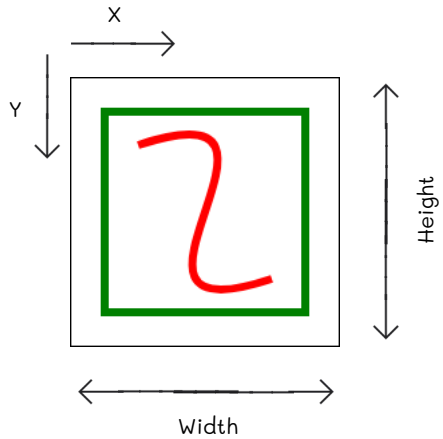
- Bézier Curves
  - Cubic
  - Quadratic
- Arcs

Cubic Bézier Curves

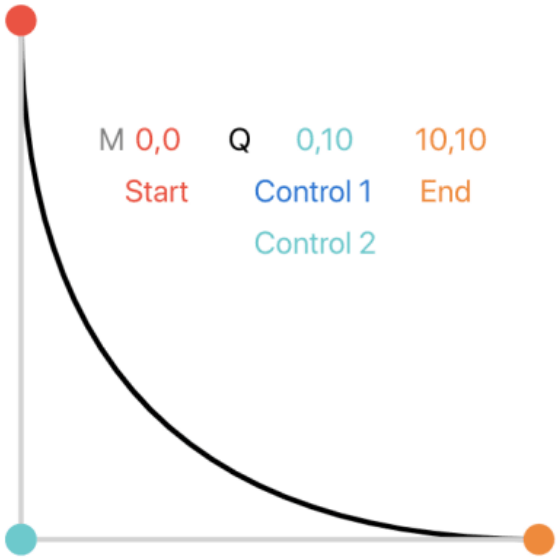


Path (Cubic Bézier Curves)

```svg
<svg
  width="200"
  height="200"
  style="margin: 100px;border: 1px solid;"
  viewBox="0 0 100 100"
>
  <path
    stroke-width="3"
    stroke="red"
    d="M 25 25 C 100 0, 0 100, 75 75"
    fill="transparent"
  />
  <path
    stroke-width="3"
    stroke="green"
    d="M 12.5 12.5 H 87.5 V 87.5 L 12.5 87.5 Z"
    fill="transparent"
  />
</svg>
```

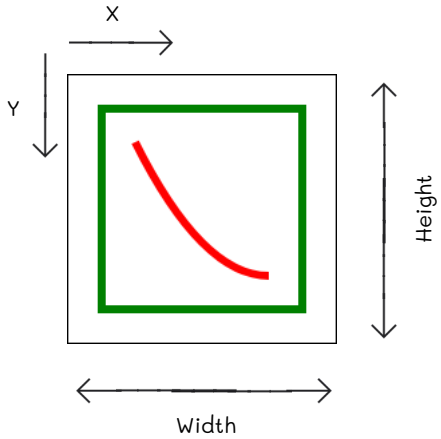


Quadratic Bézier Curves



Path (Quadratic Bézier Curves)

```svg
<svg
  width="200"
  height="200"
  style="margin: 100px;border: 1px solid;"
  viewBox="0 0 100 100"
>
  <path
    stroke-width="3"
    stroke="red"
    fill="transparent"
    d="M 25 25 Q 50 75, 75 75"
  />
  <path
    stroke-width="3"
    stroke="green"
    d="M 12.5 12.5 H 87.5 V 87.5 L 12.5 87.5 Z"
    fill="transparent"
  />
</svg>
```



Path (Curves)

- Bézier Curves
  - Cubic ✓
  - Quadratic ✓
- Arcs

Path (Arcs)

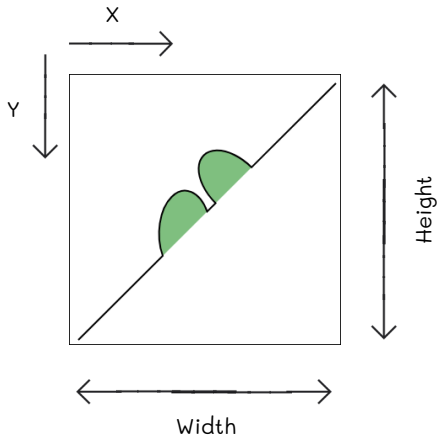
The SVG path command for drawing arcs is **A** or **a**. This command allows you to draw a section of an ellipse. The syntax for the command is as follows:

```
A  rx  ry  x-axis-rotation  large-arc-flag  sweep-flag  x  y
a  rx  ry  x-axis-rotation  large-arc-flag  sweep-flag  dx  dy
```

- **rx** and **ry**: These are the x and y radii of the ellipse. They define the size of the ellipse.
- **x-axis-rotation**: This parameter specifies the rotation angle of the ellipse's x-axis.
- **large-arc-flag**: This flag determines which of the four possible arcs to draw between the current position and the end point (x, y). If the value is 0, the smaller arc is chosen. If it's 1, the larger arc is chosen.
- **sweep-flag**: This flag determines the direction in which the arc is drawn. If the value is 0, the arc is drawn in a negative-angle direction. If it's 1, the arc is drawn in a positive-angle direction.
- **x** and **y** or **dx** and **dy**: These are the x and y coordinates where the arc should end. If the command is capitalized (**A**), the coordinates are absolute. If it's lowercase (**a**), the coordinates are relative to the current position.

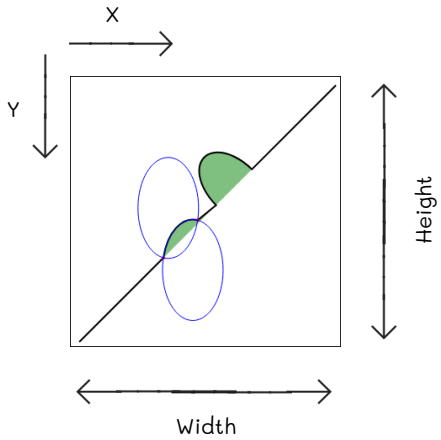
Path (Arcs)

```svg
<svg
  width="320"
  height="320"
  style="margin: 100px;border: 1px solid;"
>
  <path
    d="M 10 315 L 110 215 A 30 50 0 0 1 162.55 162.45 L 172.55 152.45
       A 30 50 -45 0 1 215.1 109.9 L 315 10"
    stroke="black"
    fill="green"
    stroke-width="2"
    fill-opacity="0.5"
  />
</svg>
```

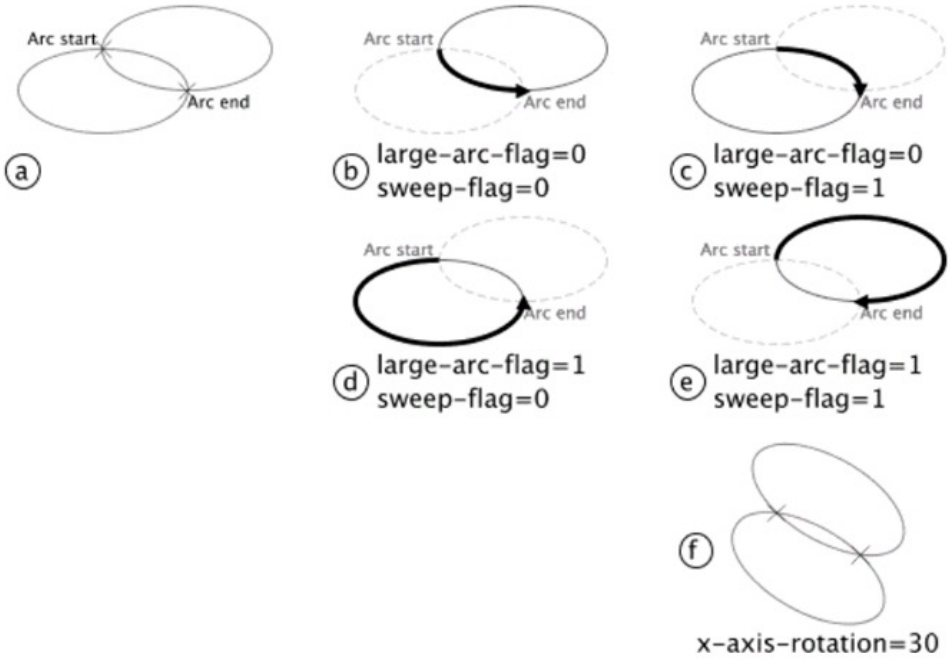


Path (Arcs)

```svg
<svg
  width="320"
  height="320"
  style="margin: 100px;border: 1px solid;"
>
  <path
    d="M 10 315 L 110 215 A 36 60 0 0 1 150.71 170.29 L 172.55 152.45
       A 30 50 -45 0 1 215.1 109.9 L 315 10"
    stroke="black"
    fill="green"
    stroke-width="2"
    fill-opacity="0.5"
  />
</svg>
```

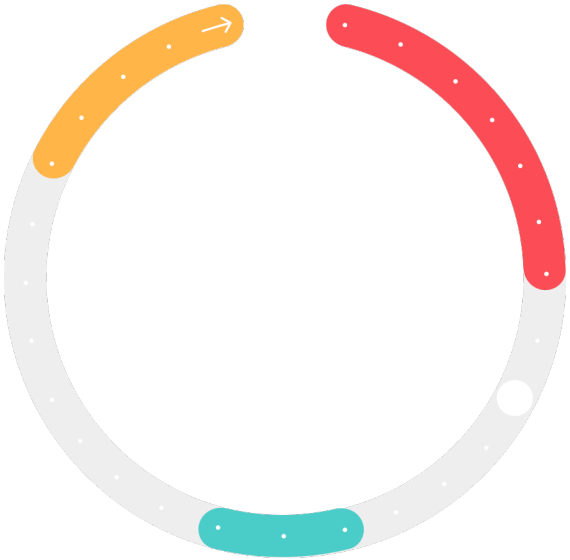


Path (Arcs)


Arc start
Arc end
a
Arc start
Arc end
b
large-arc-flag=0
sweep-flag=0
Arc start
Arc end
c
large-arc-flag=0
sweep-flag=1
Arc start
Arc end
d
large-arc-flag=1
sweep-flag=0
Arc start
Arc end
e
large-arc-flag=1
sweep-flag=1
f
x-axis-rotation=30

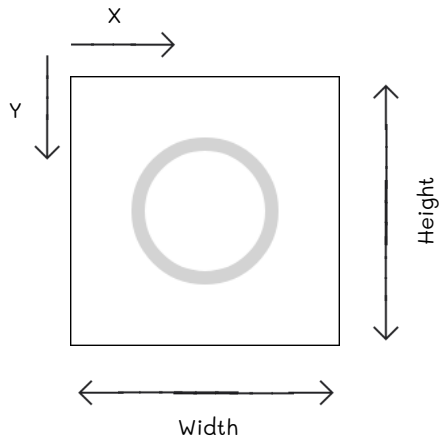

THE
END
IS
NEAR

Ready to recreate the **HerLife** cycle?

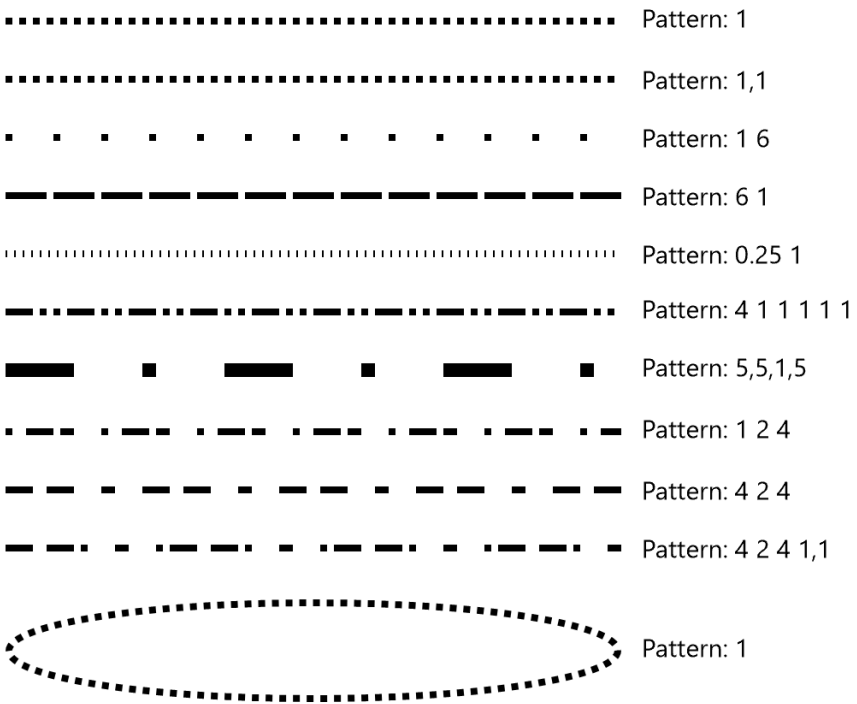


Circle

```svg
<svg
  width="200"
  height="200"
  style="margin: 100px;border: 1px solid;"
  viewBox="0 0 100 100"
>
  <circle
    r="25"
    cx="50"
    cy="50"
    stroke="lightgray"
    fill="transparent"
    stroke-width="5"
  />
</svg>
```

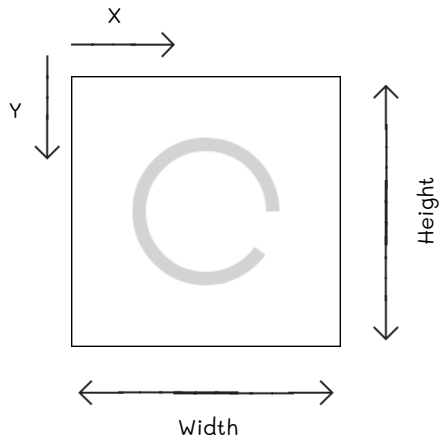


Stroke Dash Array



Stroke Dash Array

```svg
<svg
  width="200"
  height="200"
  style="margin: 100px;border: 1px solid;"
  viewBox="0 0 100 100"
>
  <circle
    r="25"
    cx="50"
    cy="50"
    stroke="lightgray"
    fill="transparent"
    stroke-width="5"
    stroke-dasharray="0 15.7 141.3"
  />
</svg>
```



Stroke Dash Array

```svg
<svg
  width="200"
  height="200"
  style="margin: 100px;border: 1px solid;"
  viewBox="0 0 100 100"
>
  <circle
    r="25"
    cx="50"
    cy="50"
    stroke="lightgray"
    fill="transparent"
    stroke-width="5"
    stroke-dasharray="0 15.7 141.3"
  />
</svg>
```

r = 25

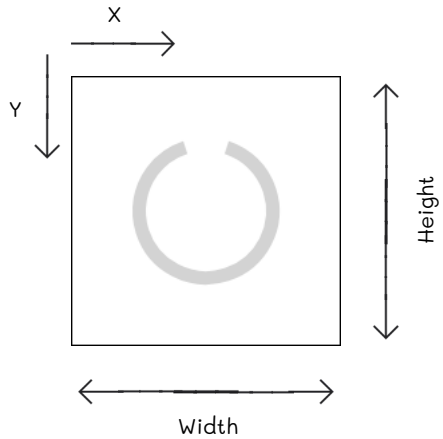
Circumference = 2πr =
2*3.14*25 = 157

Circumference * 10% = 15.7

Circumference * 90% = **141.3**

Rotate

```svg
<svg
  width="200"
  height="200"
  style="margin: 100px;border: 1px solid;"
  viewBox="0 0 100 100"
>
  <circle
    r="25"
    cx="50"
    cy="50"
    stroke="lightgray"
    fill="transparent"
    stroke-width="5"
    stroke-dasharray="0 15.7 141.3"
    transform="rotate(-108, 50, 50)"
  />
</svg>
```



Rotate

```svg
<svg
  width="200"
  height="200"
  style="margin: 100px;border: 1px solid;"
  viewBox="0 0 100 100"
>
  <circle
    r="25"
    cx="50"
    cy="50"
    stroke="lightgray"
    fill="transparent"
    stroke-width="5"
    stroke-dasharray="0 15.7 141.3"
    transform="rotate(-108, 50, 50)"
  />
</svg>
```

Circle = 360 degree

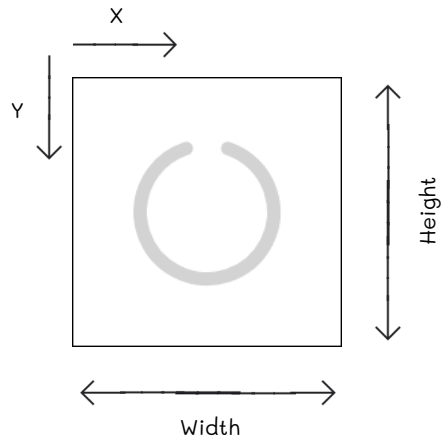
5% Circle = 18

90 + 18 = 108

Stroke Line Cap

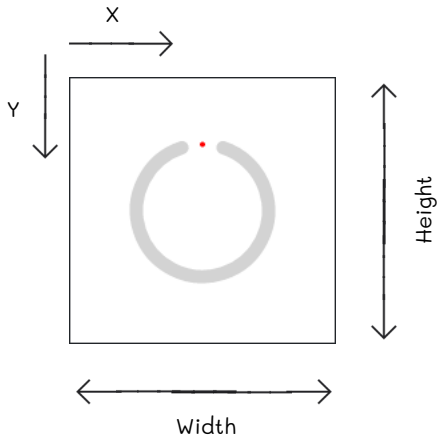
Stroke Line Cap

```svg
<svg
  width="200"
  height="200"
  style="margin: 100px;border: 1px solid;"
  viewBox="0 0 100 100"
>
  <circle
    r="25"
    cx="50"
    cy="50"
    stroke="lightgray"
    fill="transparent"
    stroke-width="5"
    stroke-dasharray="0 15.7 125.6"
    transform="rotate(-108, 50, 50)"
    stroke-linecap="round"
  />
</svg>
```



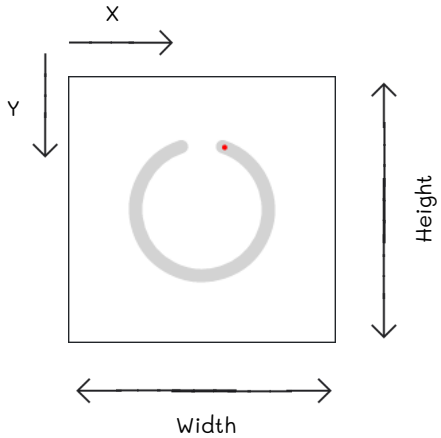
Adding Dots

```svg
<svg
  width="200"
  height="200"
  style="margin: 100px;border: 1px solid;"
  viewBox="0 0 100 100"
>
  <circle
    r="25"
    cx="50"
    cy="50"
    stroke="lightgray"
    fill="transparent"
    stroke-width="5"
    stroke-dasharray="0 15.7 141.3"
    transform="rotate(-108, 50, 50)"
    stroke-linecap="round"
  />
  <circle
    r="1"
    cx="50"
    cy="25"
    fill="red"
  />
</svg>
```

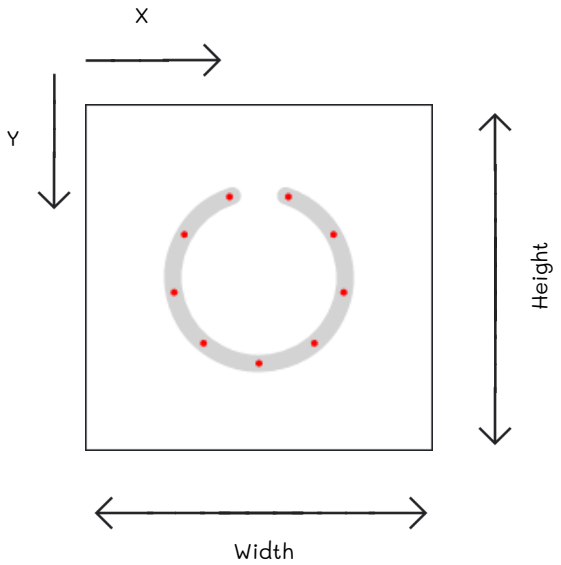


Rotating Dots

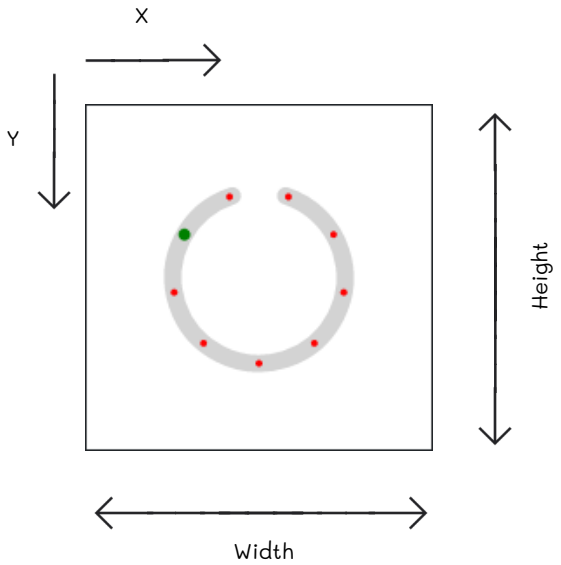
```svg
<svg
  width="200"
  height="200"
  style="margin: 100px;border: 1px solid;"
  viewBox="0 0 100 100"
>
  <circle
    r="25"
    cx="50"
    cy="50"
    stroke="lightgray"
    fill="transparent"
    stroke-width="5"
    stroke-dasharray="0 15.7 141.3"
    transform="rotate(-108, 50, 50)"
    stroke-linecap="round"
  />
  <circle
    r="1"
    cx="50"
    cy="25"
    fill="red"
    transform="rotate(20, 50, 50)"
  />
</svg>
```



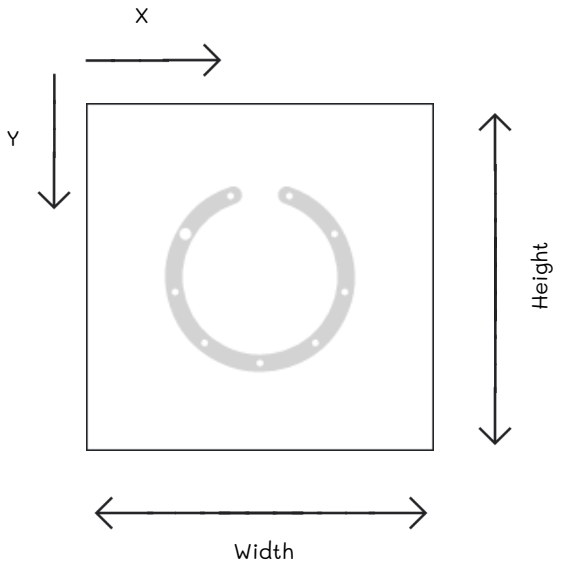
Add More Dots



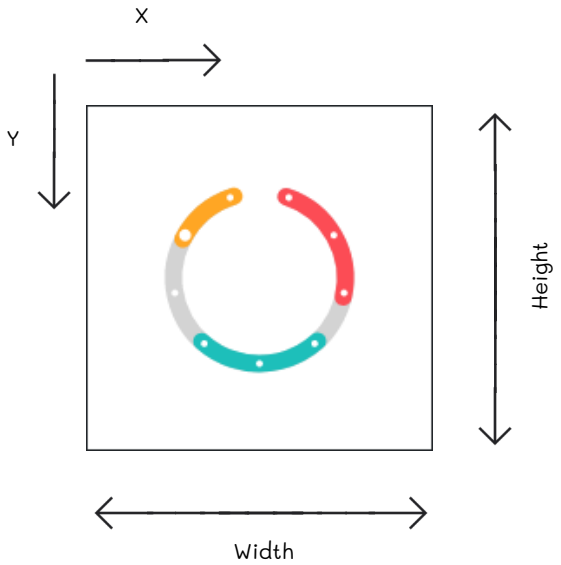
Add Current Dot



Colors



Add Ranges



Mission Complete

References

MDN web docs (link)

Javascript Info (link)

DevTo (link)

Wikimedia Commons (link)

Wikipedia (link)

Q&A

Ask me question if you have any

The END

Mohsen Madani