SVG Basics

Introduction to SVG

SVG (Scalable Vector Graphics) is an XML-based vector image format for the web. It allows for high-quality graphics that scale without losing resolution.

Why Use SVG?

- Resolution independent
- Lightweight and text-based
- Easily styled with CSS
- Interactive and animatable
- Supported by all modern browsers

What is SVG?

SVG is a markup language for describing two-dimensional graphics. It is based on XML and can be embedded directly in HTML.

Basic SVG Syntax

SVG elements are defined using XML tags. Example:

```
<svg width="100" height="100">
    <circle cx="50" cy="50" r="40" fill="blue" />
</svg>
```

Example: Drawing a Circle

```
<svg width="200" height="200">
    <circle cx="100" cy="100" r="50" fill="red" />
</svg>
```

This creates a red circle with a radius of 50.

Shapes in SVG

SVG supports basic shapes:

- `<rect>` for rectangles
- `<circle>` for circles
- `<ellipse>` for ellipses
- `<line>` for lines
- `<polygon>` for polygons
- `<path>` for custom shapes

Example: Rectangle

```
<svg width="200" height="100">
    <rect width="200" height="100" fill="green" />
</svg>
```

This creates a green rectangle.

Styling SVG with CSS

SVG elements can be styled using CSS:

```
<svg width="100" height="100">
    <circle cx="50" cy="50" r="40" class="my-circle" />
</svg>
<style>
    .my-circle {
        fill: orange;
        stroke: black;
        stroke-width: 2;
    }
</style>
```

Adding Text to SVG

SVG supports text elements:

```
<svg width="200" height="50">
    <text x="10" y="30" font-size="20" fill="black">Hello SVG</text>
</svg>
```

Transformations in SVG

SVG supports transformations like rotate, scale, and translate:

```
<svg width="200" height="200">
    <rect width="100" height="50" fill="blue" transform="rotate(45 50 50)" />
</svg>
```

Animations in SVG

SVG supports animations using `<animate>` and `<animateTransform>` :

```
<svg width="100" height="100">
    <circle cx="50" cy="50" r="40" fill="purple">
        <animate attributeName="r" from="10" to="40" dur="2s" repeatCount="indef
    </circle>
</svg>
```

SVG vs Canvas

SVG and Canvas are both used for graphics, but they differ in approach:

- SVG is declarative, Canvas is imperative.
- SVG is resolution independent, Canvas is pixel-based.
- SVG is better for static or interactive vector graphics, Canvas is better for dynamic, pixel-based graphics.

Example: SVG vs Canvas

SVG:

```html
<svg width="100" height="100">
    <circle cx="50" cy="50" r="40" fill="blue" />
</svg>
```

Canvas:

```html
<canvas width="100" height="100"></canvas>
<script>
    const canvas = document.querySelector('canvas');
    const ctx = canvas.getContext('2d');
    ctx.fillStyle = 'blue';
    ctx.beginPath();
    ctx.arc(50, 50, 40, 0, Math.PI * 2);
    ctx.fill();
</script>
```

Real-World Use Cases

- Icons and logos
- Charts and graphs
- Animations
- Interactive maps