Canvas and SVG(Scalable Vector Graphics)

류관희
충북대학교

HTML5 Canvas? (1/2)

- The *HTML5 Canvas* is an *Immediate Mode* bit-mapped area of the screen that can be manipulated with JavaScript and CSS.
- The HTML5 <canvas> element is used to draw graphics, on the fly, via scripting (usually JavaScript).
- The <canvas> element is only a container for graphics. You must use a script to actually draw the graphics.
- Canvas has several methods for drawing paths, boxes, circles, characters, and adding images.

HTML5 Canvas? (2/2)

- Features
  - 2D Context
    - Save, Restore : push or pop graphics state stack
    - Transformation
    - Global alpha
    - Shapes : Fill and Stroke Path
    - Canvas State : Fill and Stroke Style, Matrix, Clipping region…
    - Image
    - Text
- Hardware Accelerated Canvas Element

Flash vs Canvas

- Finished Essential guide To Flash Games in March 2010.
- Steve Jobs refused Flash on the iOS just days later.
- Easy Tools: a web browser, text editor and JavaScript was all that was required.
- The HTML5 Canvas allowed for a bitmapped graphics, much Flash's bitmapped canvas.
- Our specific Type Of Game Development translates well (tile sheets, bitmaps)

Retained mode vs Immediate

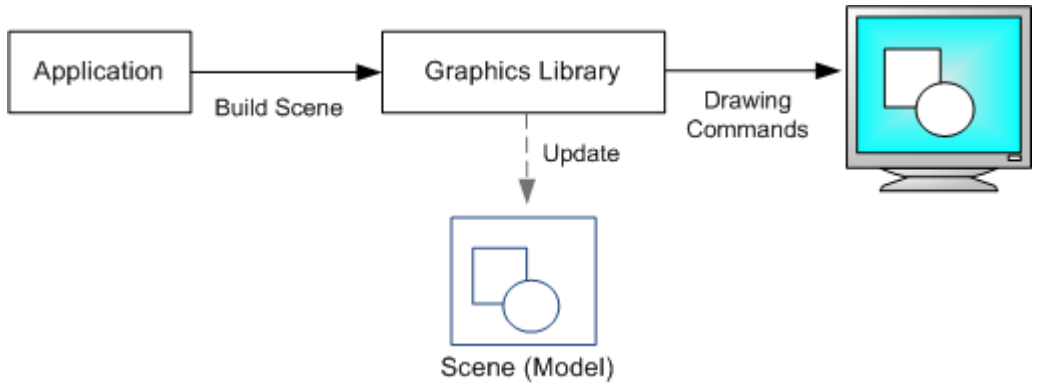


Fig1. Retained Mode

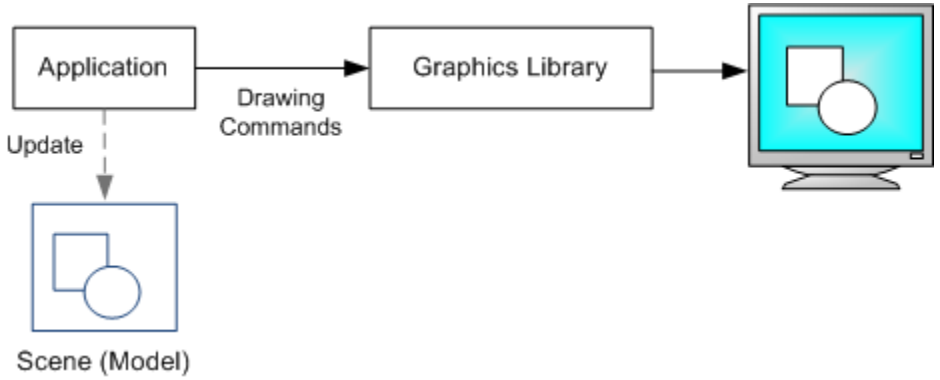
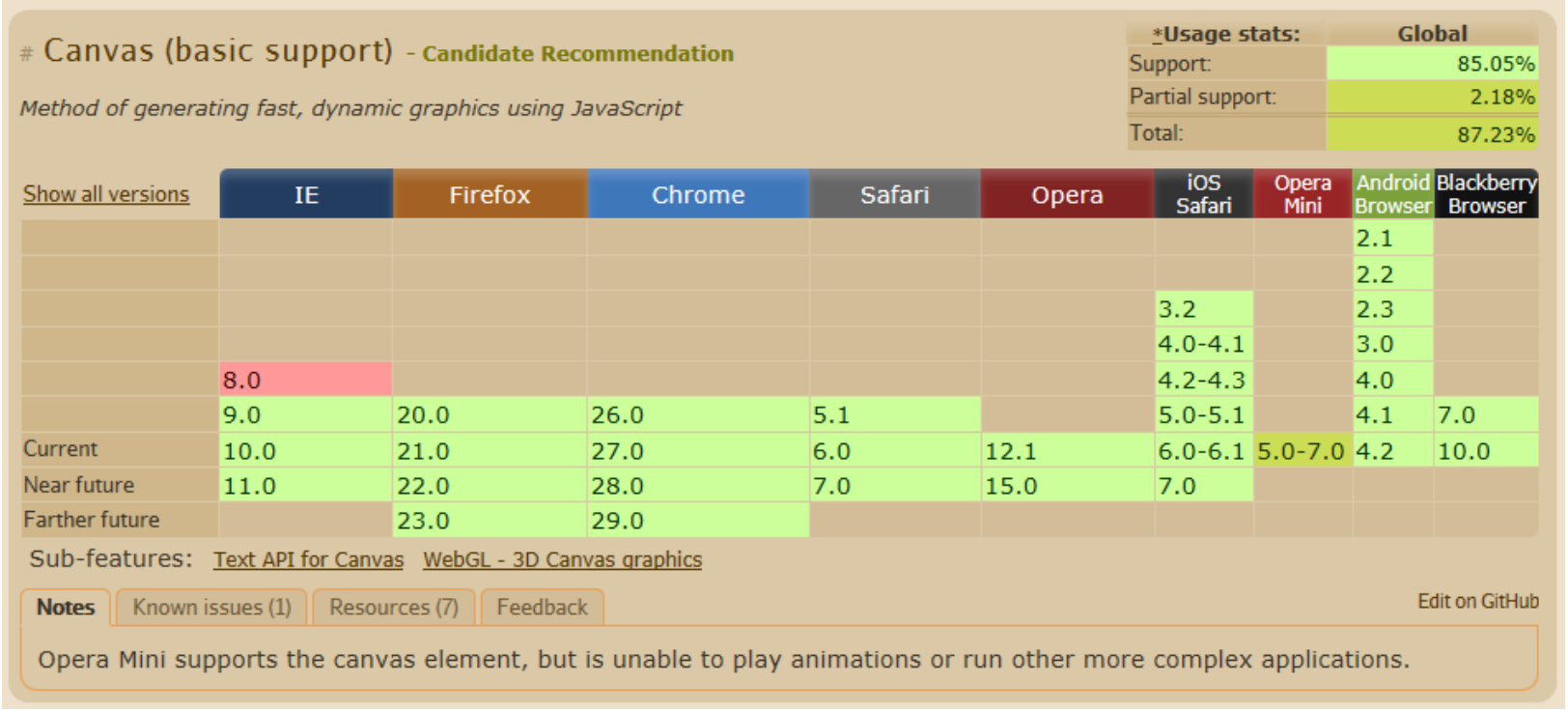


Fig2. Immediate Mode

HTML5 Canvas Support



Canvas (basic support) - Candidate Recommendation

Method of generating fast, dynamic graphics using JavaScript

*Usage stats:	Global
Support:	85.05%
Partial support:	2.18%
Total:	87.23%

Show all versions	IE	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Blackberry Browser
								2.1	
								2.2	
						3.2		2.3	
						4.0-4.1		3.0	
	8.0					4.2-4.3		4.0	
	9.0	20.0	26.0	5.1		5.0-5.1		4.1	7.0
Current	10.0	21.0	27.0	6.0	12.1	6.0-6.1	5.0-7.0	4.2	10.0
Near future	11.0	22.0	28.0	7.0	15.0	7.0			
Farther future		23.0	29.0						

Sub-features: Text API for Canvas WebGL - 3D Canvas graphics

Notes | Known issues (1) | Resources (7) | Feedback | Edit on GitHub

Opera Mini supports the canvas element, but is unable to play animations or run other more complex applications.

HTML5 Canvas properties & methods

- width
- height
- id

```
<canvas id="MyFirstCanvas" width="600" heght="200">
This text is displayed if your browser does not support HTML5 Canvas.
</canvas>
```

Colors, Styles, and Shadows

Property	Description
fillStyle	Sets or returns the color, gradient, or pattern used to fill the drawing
strokeStyle	Sets or returns the color, gradient, or pattern used for strokes
shadowColor	Sets or returns the color to use for shadows
shadowBlur	Sets or returns the blur level for shadows
shadowOffsetX	Sets or returns the horizontal distance of the shadow from the shape
shadowOffsetY	Sets or returns the vertical distance of the shadow from the shape

Method	Description
createLinearGradient()	Creates a linear gradient (to use on canvas content)
createPattern()	Repeats a specified element in the specified direction
createRadialGradient()	Creates a radial/circular gradient (to use on canvas content)
addColorStop()	Specifies the colors and stop positions in a gradient object

Rectangles

Method	Description
rect()	Creates a rectangle
fillRect()	Draws a "filled" rectangle
strokeRect()	Draws a rectangle (no fill)
clearRect()	Clears the specified pixels within a given rectangle

Line Styles

Property	Description
lineCap	Sets or returns the style of the end caps for a line
lineJoin	Sets or returns the type of corner created, when two lines meet
lineWidth	Sets or returns the current line width
miterLimit	Sets or returns the maximum miter length

Paths

Method	Description
fill()	Fills the current drawing (path)
stroke()	Actually draws the path you have defined
beginPath()	Begins a path, or resets the current path
moveTo()	Moves the path to the specified point in the canvas, without creating a line
closePath()	Creates a path from the current point back to the starting point
lineTo()	Adds a new point and creates a line from that point to the last specified point in the canvas
clip()	Clips a region of any shape and size from the original canvas
quadraticCurveTo()	Creates a quadratic Bézier curve
bezierCurveTo()	Creates a cubic Bézier curve
arc()	Creates an arc/curve (used to create circles, or parts of circles)
arcTo()	Creates an arc/curve between two tangents
isPointInPath()	Returns true if the specified point is in the current path, otherwise false

Transformations

Method	Description
scale()	Scales the current drawing bigger or smaller
rotate()	Rotates the current drawing
translate()	Remaps the (0,0) position on the canvas
transform()	Replaces the current transformation matrix for the drawing
setTransform()	Resets the current transform to the identity matrix. Then runs transform()

Text

Property	Description
font	Sets or returns the current font properties for text content
textAlign	Sets or returns the current alignment for text content
textBaseline	Sets or returns the current text baseline used when drawing text

Method	Description
fillText()	Draws "filled" text on the canvas
strokeText()	Draws text on the canvas (no fill)
measureText()	Returns an object that contains the width of the specified text

Images

Method	Description
drawImage()	Draws an image, canvas, or video onto the canvas

Property	Description
width	Returns the width of an ImageData object
height	Returns the height of an ImageData object
data	Returns an object that contains image data of a specified ImageData object

Method	Description
createImageData()	Creates a new, blank ImageData object
getImageData()	Returns an ImageData object that copies the pixel data for the specified rectangle on a canvas
putImageData()	Puts the image data (from a specified ImageData object) back onto the canvas

Compositing & Others

Property	Description
globalAlpha	Sets or returns the current alpha or transparency value of the drawing
globalCompositeOperation	Sets or returns how a new image are drawn onto an existing image

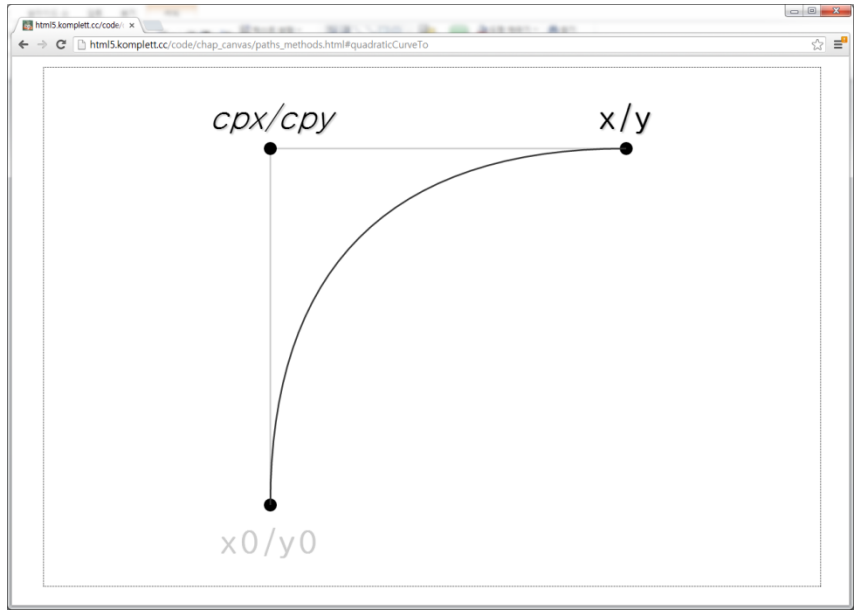
Method	Description
save()	Saves the state of the current context
restore()	Returns previously saved path state and attributes
createEvent()	
getContext()	
toDataURL()	

Example : Two overlapped rectanges

```
<script>
    var canvas = document.querySelector("canvas");
    var context = canvas.getContext('2d');
    context.fillStyle = 'red';
    context.fillRect(0,0,800,600);
    context.fillStyle = 'rgba(255,255,0,0.5)';
    context.fillRect(400,200,800,600);
  </script>
```

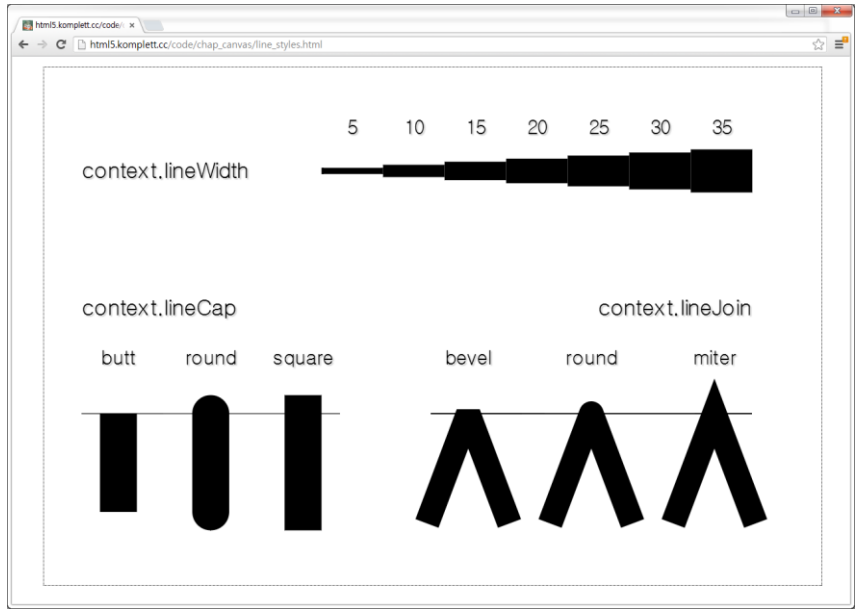
Example : Curve

```
 <script>
    var canvas = document.querySelector("canvas");
    var context = canvas.getContext('2d');
.....................
    // geometry
    context.save();
    context.beginPath();
    context.moveTo(20,130);
    context.quadraticCurveTo(20,20,130,20);
    context.strokeStyle = colors.path;
    context.stroke();
    context.restore();
.....................
</script>
```



Example : Stroke

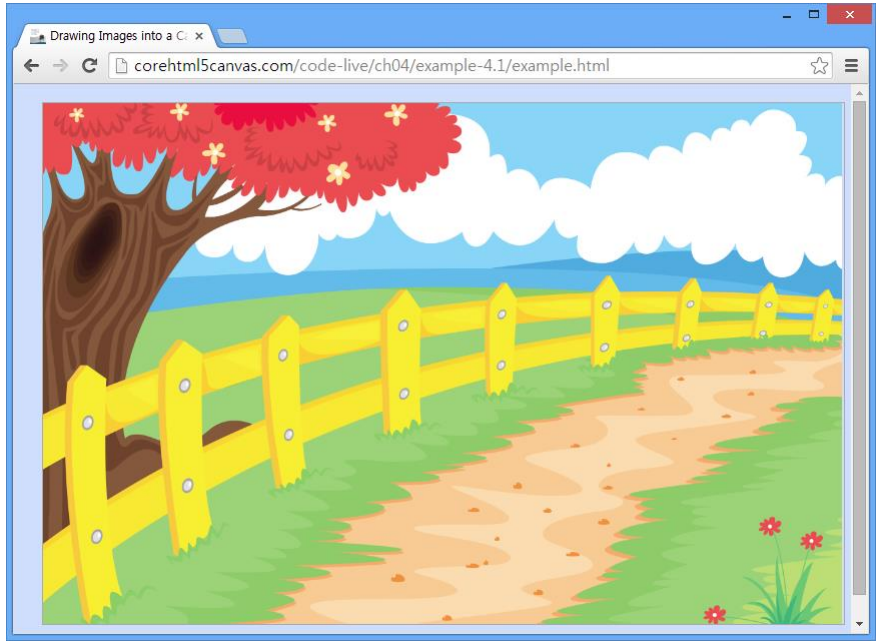
```
<script>
    var canvas = document.querySelector("canvas");
    var context = canvas.getContext('2d');
.....
    // thick line
    context.lineWidth = 20.0;
    // join style
    context.beginPath();
    context.lineJoin = 'bevel';
    context.lineJoin = 'round';
    context.lineJoin = 'miter';
    context.moveTo(20,130);
    context.lineTo(50,50);
    context.lineTo(80,130);
    context.stroke();
.....
    // cap style
    context.beginPath();
    context.lineCap = 'butt';
    context.lineCap = 'round';
    context.lineCap = 'square';
    context.moveTo(50,130);
    context.lineTo(50,50);
    context.stroke();
.....
</script>
```



Example : Image

```
<script>
var canvas = document.getElementById('canvas'),
    context = canvas.getContext('2d'),
    image = new Image();

image.src = '../../shared/images/countrypath.jpg';
image.onload = function(e) {
   context.drawImage(image, 0, 0);
};
</script>
```



Canvas Demo

- http://www.upstreamdown.com/fishtank/

SVG (Scalable Vector Graphics)

- **SVG**
- **SVG Characteristic**
- **SVG History**

- **Tutorial**
  - ◯ **Rectangle <rect>, Circle <circle>, Ellipse <ellipse>, Line <line>**
  - ◯ **Polygon <polygon>, Polyline <polyline>**
  - ◯ **Path <path>**
  - ◯ **Gradient, Text**
  - ◯ **Stroking**
  - ◯ **Etc**
    - **Exporting**
    - **애니메이션 효과 예제.. 원형차트 그리기**
    - **SVG를 사용한 그래픽 라이브러리 소개**

SVG (Scalable Vector Graphics)

:: XML형식의 그래픽을 정의하고 웹을 위한 벡터 기반의 그래픽을 정의하기 위해 사용됨

Bitmap_VS_SVG.svg (SVG 파일, 실제 크기 512 × 327 픽셀, 파일 크기: 16 KB)

- 확대하거나 크기를 변화시켜도 그래픽이 손상되지 않음.
- 다양한 해상도와 픽셀밀도에 대응이 가능함.
- 애니메이션과 미디어쿼리지원 등을 포함하고 있음.

SVG의 특징

- **XML기반 ::**
  CSS(Cascading Style Sheets), XSL(XML Style Language)를 지원
  (스타일시트를 통한 커스터마이징이 가능/ XML기반의 다른 문서와 통합 가능)

- **멀티 플랫폼(Multi - Platform) ::**
  시스템, 운영체제에 장애를 받지 않음. (PC, Mobile, android, iOS 등에서 모두 지원됨)

- **상호작용(Interactive Graphics) ::**
  키보드, 마우스 입력에 대한 응답이 가능하고 Interactive 그래프 등을 비교적 쉽게 구현가능

- **데이터 연동(Data-Driven Graphics) ::** 데이터베이스연동이 가능
  (복잡한 데이터의 가시화가 필요한 지리정보, 실시간 정보 가시화 등에 효과적)

- **개인화(Personalized Graphics)::**

SVG History

- 1998 ::

  ○Adobe, Sun, Netscape에서 함께 벡터 그래픽을 위한 웹 문서 표준으로 PGML(Precision Graphics Markup Language) 제안

  (이전에 Microsystem, MacroMedia에서 VML을 제안해으나 그에 대응하여 PGML을 제안)

  ○W3C(World Wide Web Consortium) ::

  PGML과 VML 기반의 새로운 벡터 그래픽 포맷 제정을 결정함.

  - VML(vector markup language),
  - PGML(Precision Graphics Markup Language)

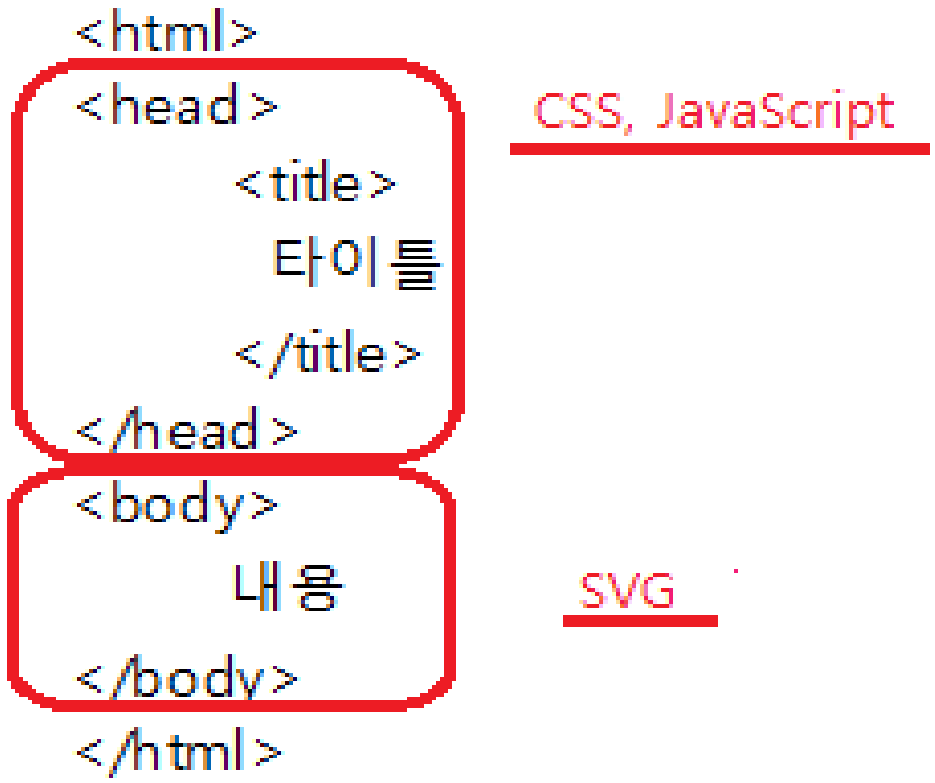
SVG History

- 1998년 08월 ::
  - ◯SVG Working Group
    Adobe, Macromedia, IBM, Corel, Apple, HP,
    Microsoft, AutoDesk, Netscape, CSIRO, Kodak
- 1999년 02월 :: 초안 발표
- 2001년 09월 SVG 1.0 권고
- 2003년 01월 :: SVG 1.1 권고 (W3C 권고안으로 발표)
  - ◯Mobile SVG Profile인 SVG Tiny와 SVG Basic
- 2008년 12월 :: SVGT 1.2 권고

SVG Tutorial

- **Tutorial** (http://www.w3schools.com/svg/)
  - ○Basic Shape ::
    Rectangle <rect>, Circle <circle>,
    Ellipse <ellipse>, Line <line>
  - ○Polygon <polygon>, Polyline <polyline>
  - ○Path <path>
  - ○Text
  - ○Stroking
  - ○Etc
    - 애니메이션 효과 예제.. 원형차트 그리기
    - SVG를 사용한 그래픽 라이브러리 소개

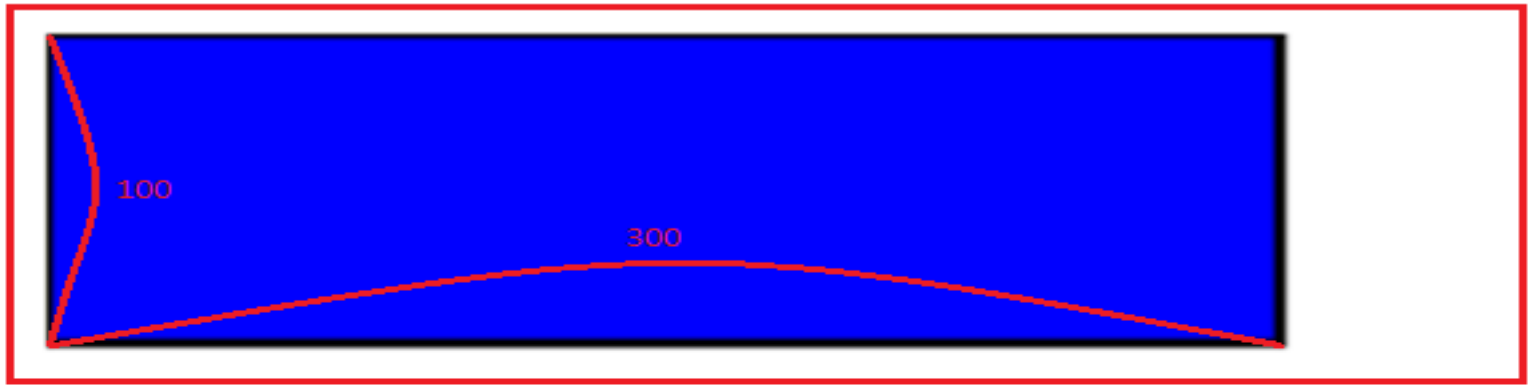
HTML Form



- <Head>
  - ○JavaScript
    - <script> 내용 </script>
  - ○CSS
    - <style>내용</style>
- <Body>
  - ○SVG
    - <svg>내용</svg>
    - canvas
    - 정적인 html

Rectangle

● <rect>



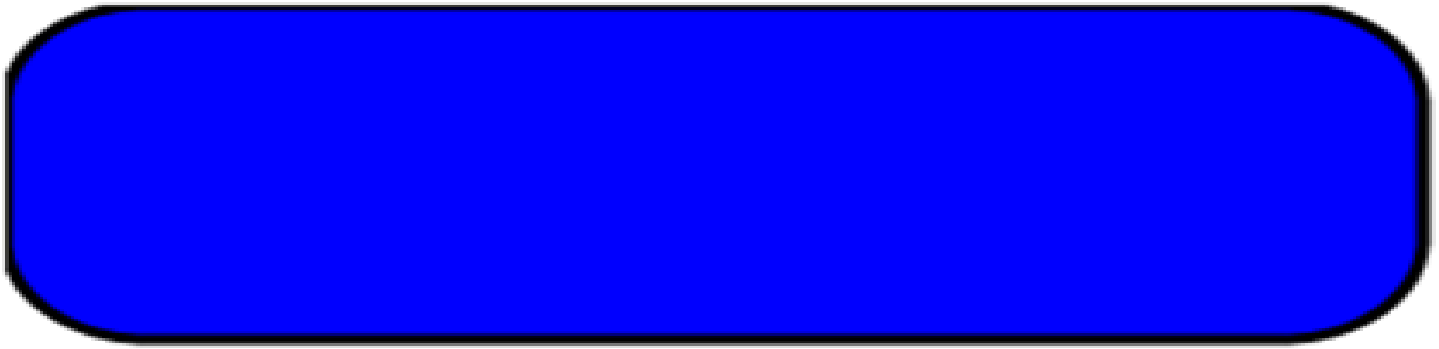
- Source ::

```
<svg width="400"  height="110">
  <rect width="300"  height="100"
        style="fill:rgb(0,0,255);stroke-width:3;  stroke:rgb(0,0,0)">
</svg>
```

SVG태그 시작 / SVG가 Display되는 영역 / 사각형 크기 / 도형을 채우는 색 / 경계선 굵기 / 경계선 색 / SVG태그 끝

Rectangle

- <rect>
  => rx와 ry 추가

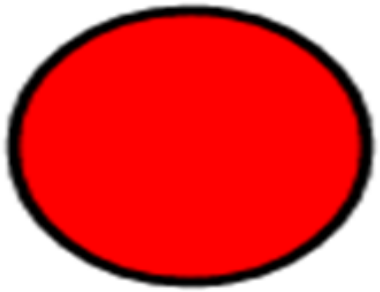


얼마나 둥글게 만들 것인가
rx = x축 방향
ry = y축 방향

```
<svg width="400"  height="110">
   <rect width="300"  height="100"  rx ="30"  ry="30"
                  style="fill:rgb(0,0,255);stroke-width:3;   stroke:rgb(0,0,0)">
</svg>
```

Circle

- <circle>



cx, cy = 중심점
r = 반지름

- Source ::

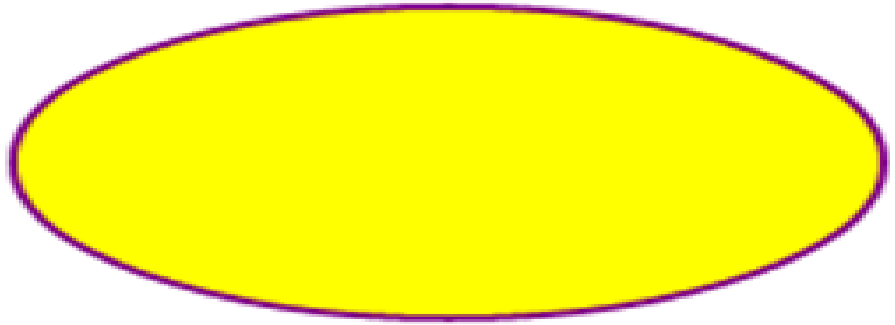
```
<svg height="100"  width="100">
  <circle cx="50"  cy="50"  r="40"  stroke="black"  stroke-width="3"  fill="red" />
</svg>
```

원 중심점 (cx="50" cy="50")

반지름 (r="40")

Ellipse

- <ellipse>



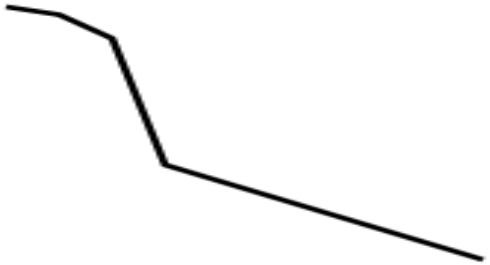
rx = x축 반지름
ry = y축 반지름

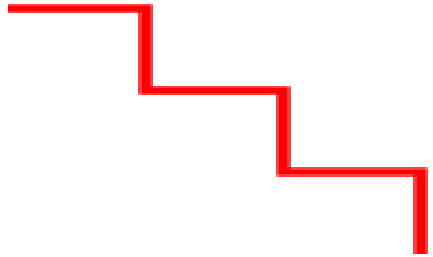
- Source ::

```
<svg height="140"  width="500">
 <ellipse cx="200"  cy="80"  rx="100"  ry="50"
                    style="fill:  yellow;  stroke:  purple;  stroke-width:2"  />
</svg>
```

Line & Polyline

< line >





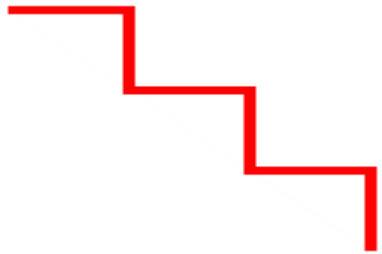
< polyline >

- Source ::

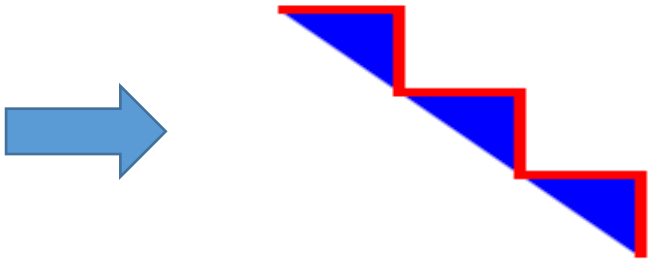
```
<svg height="210" width="500">
<line x1="0" y1="0" x2="200" y2="200" style="stroke:rgb(255,0,0);stroke-width:2" />
</svg>

<polyline points="20,20 40,25 60,40 80,120 120,140 200,180"
                                        style="fill:none;stroke:black;stroke-width:3" />
<polyline points="0,40 40,40 40,80 80,80 80,120 120,120 120,160"
                                        style="fill:white;stroke:red;stroke-width:4" />
```

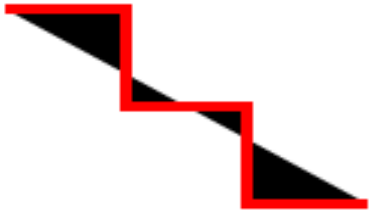
Polyline



fill : white or none



fill : blue



시작점, 끝점 기준

- Source ::

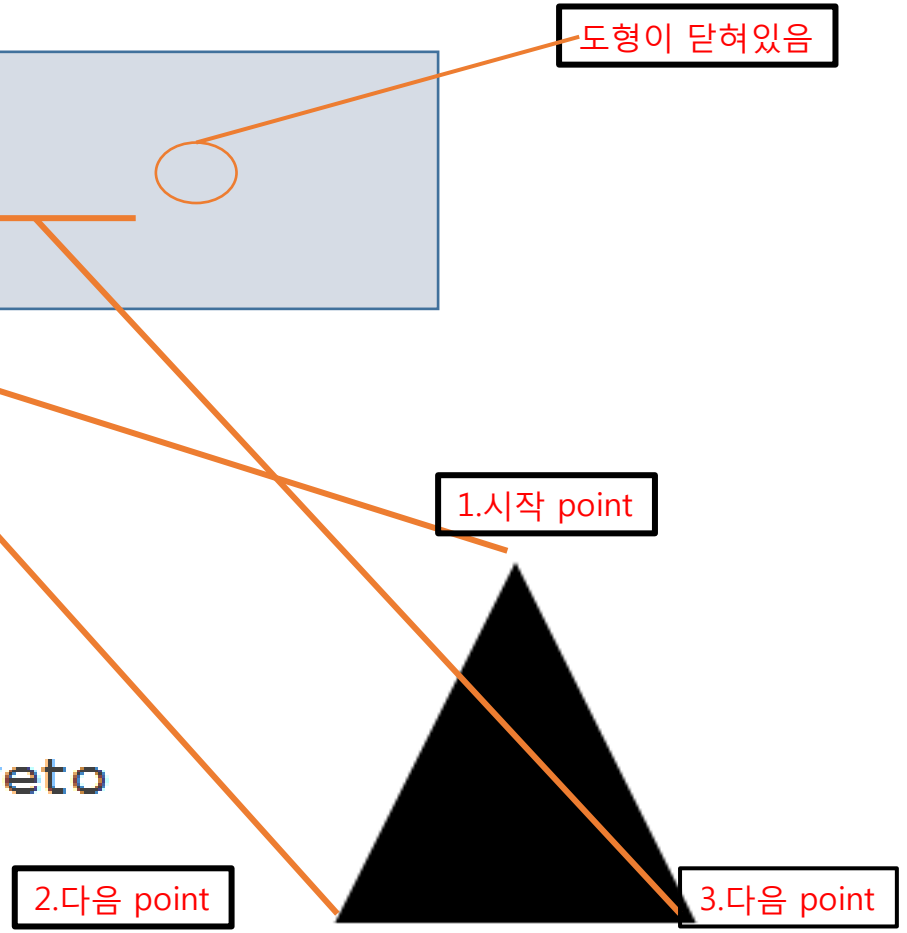
```
<polyline points=      "0,40 40,40 40,80 80,80
                        80,120 120,120 120,160"
                       style="fill:white; stroke:red; stroke-width:4" />
                              => fill: blue;
```

Path

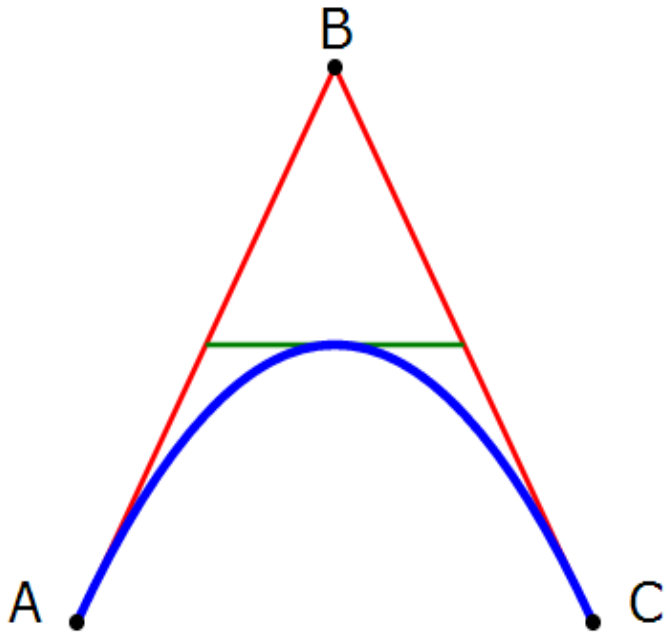
- Path property

```
<svg height="210" width="400">
        <path d="M150 0 L75 200 L225 200 Z"/>
</svg>
```

- M = moveto
- L = lineto
- H = horizontal lineto
- V = vertical lineto
- C = curveto
- S = smooth curveto
- Q = quadratic Bézier curve
- T = smooth quadratic Bézier curveto
- A = elliptical Arc
- Z = closepath`



Path




```
<svg height="400" width="450">
 <path id="lineAB" d="M 100 350 L 150 -300"
                                                stroke="red" stroke-width="3" fill="none" />
 <path id="lineBC" d="M 250 50 L 150 300"
                                                stroke="red" stroke-width="3" fill="none" />
 <path d="M 175 200 L 150 0"
                                                stroke="green" stroke-width="3" fill="none" />
 <path d="M 100 350 q 150 -300 300 0"
                                                stroke="blue" stroke-width="5" fill="none" />

 <g stroke="black" stroke-width="3" fill="black">
  <circle id="pointA" cx="100" cy="350" r="3" />
  <circle id="pointB" cx="250" cy="50" r="3" />
  <circle id="pointC" cx="400" cy="350" r="3" />
 </g>

 <g font-size="30" font="sans-serif" fill="black"
                                                stroke="none"  text-anchor="middle">
  <text x="100" y="350" dx="-30">A</text>
  <text x="250" y="50" dy="-10">B</text>
  <text x="400" y="350" dx="30">C</text>
 </g>
</svg>
```

Path

대문자 : 절대 좌표
소문자 : 상대 좌표

Path command ::

M = moveto
L = lineto
H = horizontal lineto
V = vertical lineto
Z = closepath

- M(x,y) :: 시작점
- L (x,y) :: 기준점에서 좌표까지 직선
- H :: 다음포인트까지 수평선
- V :: 다음포인트까지 수직선
- Z :: path를 닫는다. (마지막 point에서 M point로 직선)

Path Curve

Curve command ::

Q = quadratic Bézier curve
T = smooth quadratic Bézier curveto
C = Cubic bezier curveto
S = smooth curveto
A = elliptical Arc

=> 2차 Bezier curve

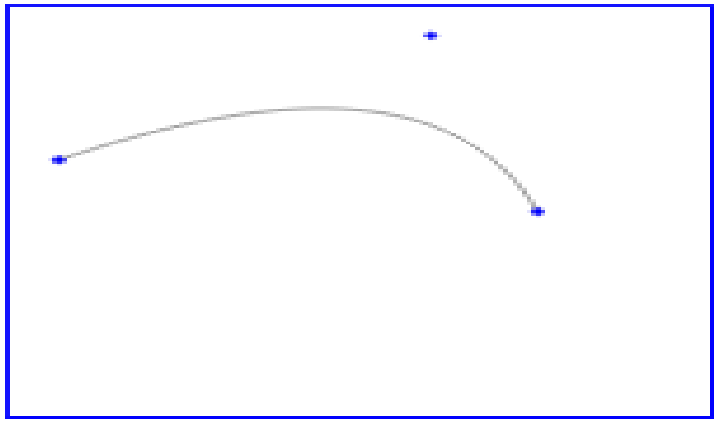
- **Q(x1,y1 x,y) :: 기준점에서 x,y까지 곡선을 그려준다.(x1,y1은 control point)**
- **T(x,y) :: 베지어커브에 이어서 커브를 그릴 때 사용함.**

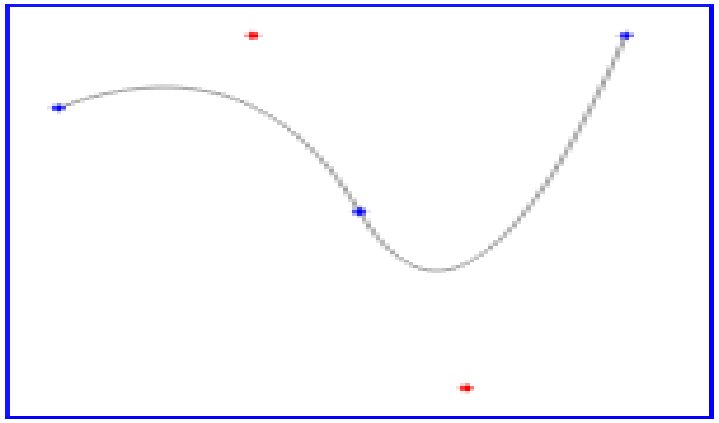
=> 3차 Bezier curve

- **C(x2,y2 x1,y1 x,y) :: Q에서 조절점이 하나 추가**
- **S(x1,y1 x,y) ::조절 점은 앞선 커브에 대칭되는 위치에 놓이게 된다.**

- **A(rx,ry x-axis-rotation large-arch-flag, sweepflag x,y) ::**

Bezier curve(2차)



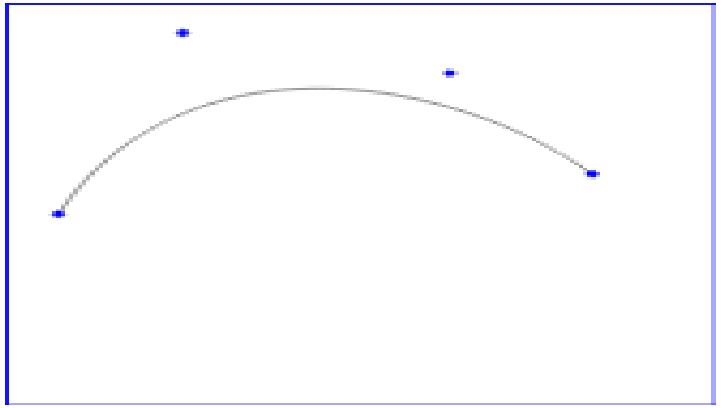


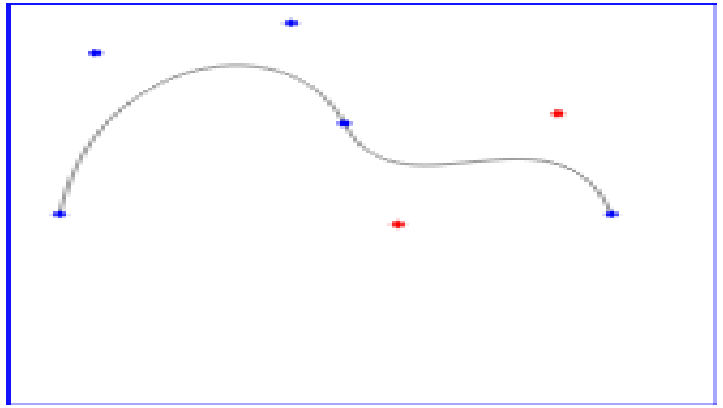
- **Source ::**

```
<path d="M 30 150 Q240 30 300 200"
                                        fill="none" stroke="black"/>
<path d="M 30 100 Q140 30 200 200 T350 30"
                                        fill="none" stroke="black"/>
```

Q(x1,y1 x,y) :: 기준점에서 x,y까지 곡선을 그려준다.
T(x,y) :: 연속해서 커브를 그릴 때 사용함.

Bezier curve(3차)



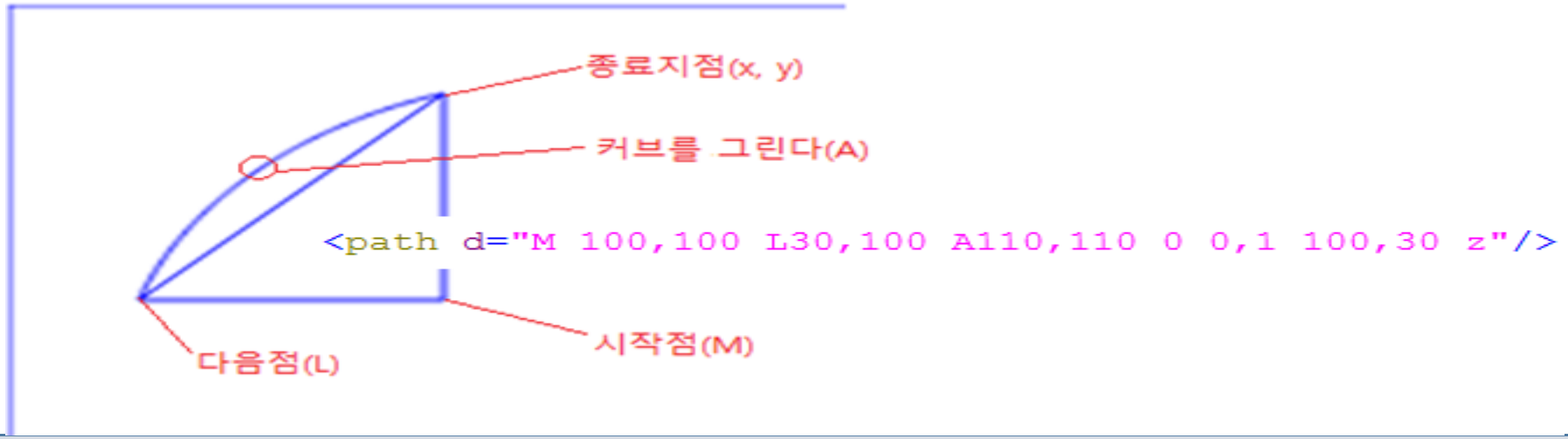


- **Source ::**

```
<path d="M 30 150 Q240 30 300 200"
                                        fill="none" stroke="black"/>
<path d="M 30 100 Q140 30 200 200 T350 30"
                                        fill="none" stroke="black"/>
```

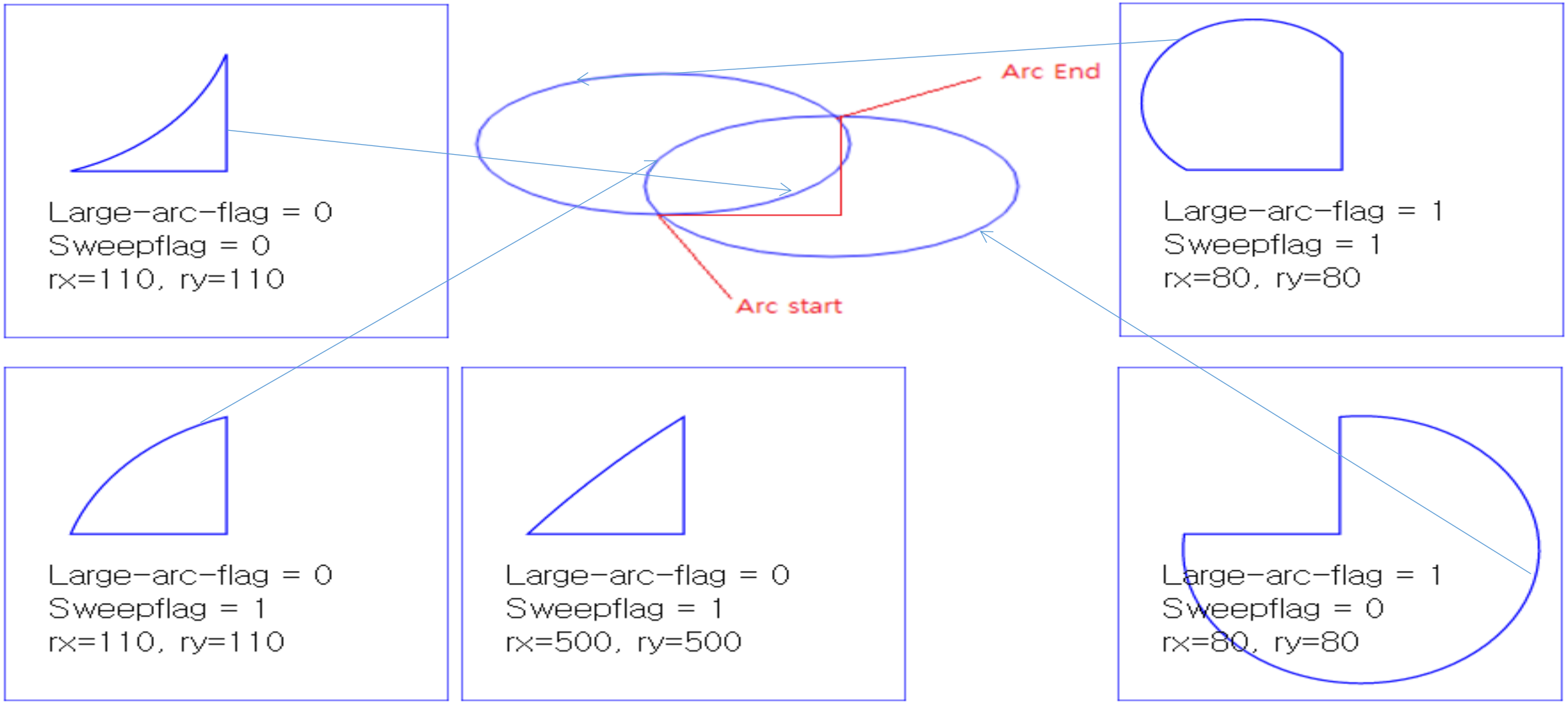
Q(x1,y1 x,y) :: 기준점에서 x,y까지 곡선을 그려준다.
T(x,y) :: 연속해서 커브를 그릴 때 사용함.

Elliptical Arc



- A(rx,ry x-axis-rotation large-arch-flag, sweepflag x,y) ::
  - ○ rx, ry :: x, y방향 타원의 반지름
  - ○ x-axis-rotation : x축 주변으로 얼마나 구부러지는지 (rx와 ry의 값이 다를 경우에만 효과있음)
  - ○ Large-arch-flag : 커브의 크기(0 | 1) (1이면 크게)
  - ○ Sweepflag : 커브의 방향 (0 | 1) (1이면 바깥?)
  - ○ x, y : 종료지점

Elliptical Arc parameter



Gradient

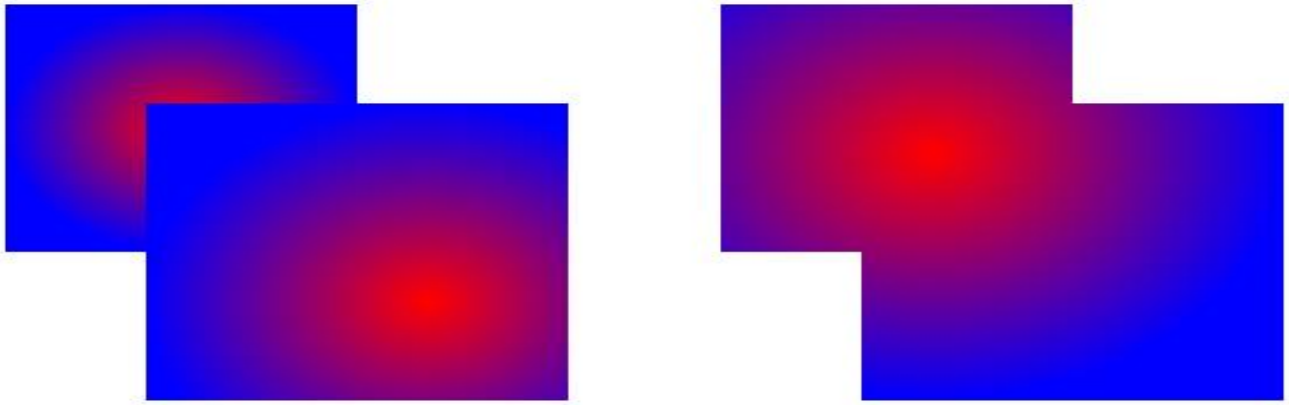
- Linear

Source ::

```
<defs>                <ObjectBoundingBox>      <userSpaceOnUse>
   <linearGradient id ="Exam1" x1="0%" y1="30%" x2="30%"
         y2="100%"gradientUnits=“ ????????????? ">
   <stop offset="0%" stop-color="red"/>
   <stop offset="100%" stop-color="blue"/>
   </linearGradient>
</defs>
```

Gradient

- **Radial**



Source ::

```
<defs>          <ObjectBoundingBox>      <userSpaceOnUse>
  <radialGradient id ="Exam1" x1="0%" y1="0%" x2="100%"
        y2="100%"gradientUnits="??????????????????">
  <stop offset="0%" stop-color="red"/>
  <stop offset="100%" stop-color="blue"/>
  </radialGradient>
</defs>
```

Text

- Source :: Basic Text form

```
<text x="0" y="15">
                        I love SVG!   => 시작 위치, Text message
</text>
<text x="0" y="15" fill="red">
                        I love SVG! => 시작 위치, fill, Text message
</text>
```

- Source :: Transform

```
<text x="0" y="15" fill="red“  transform="rotate(30 20, 40)">I love SVG</text>
=> transform="rotate(angle) translate(tx, ty)“
=> rotate( angle x, y )  // text를 20, 40 이동시켜서 기울기 30의 텍스트 출력
```

Text(tspan, link)

Tspen을 사용하면 Text내용의 일부에
font-size 20pt. 2-2rd line.

Rotate tspan. (rotate(angle))

여기를 클릭.

속 성	설 명
x="<coordinate>"	텍스트의 x 절대 좌표 값
y="<coordinate>"	텍스트의 y 절대 좌표 값
dx="<length>"	텍스트의 x 상대 좌표 값
dy="<length>"	텍스트의 y 상대 좌표 값
rotate="<angle>"	회전 각도를 지정한다

- **tspan ::**

```
<text x="10" y="20" style="fill:red;">Several lines:
        <tspan x="10" y="20">First line.</tspan>
        <tspan x="10" y="70">Second line.</tspan>
</text>
```

- **Source :: hyperlink**

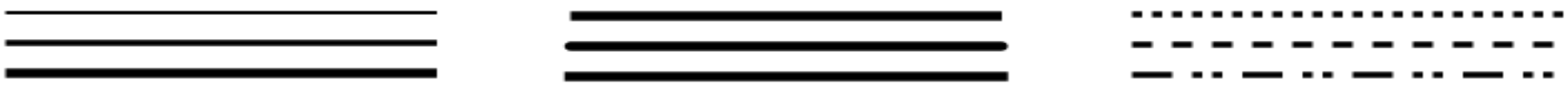
```
<a xlink:href="http://www.w3schools.com/svg/" target="_blank">
<text x="0" y="15" fill="red">I love SVG!</text>
</a>
<!-- <a> 지정된 text에 link를 걸어준다. -->
```

Text_Path

- 웹상의 SVG를 수정

1. Circle
2. Radial Gradient
3. Path
4. Text_Path

Stroke



width　　linecap　　dasharray

●그래픽 오브젝트의 윤곽을 지정하는 것

○Stroke-width=“<length>”
○Stroke-linecap= butt | round | square
○stroke-dasharray= “선,공백,선,공백….”

SVG Animation

- 동적인 웹환경에 적합한 형태
- 시간에 따라 변화

  SVG Animation의 5가지 요소

  ◯animate : 스칼라 값을 가진 속성을 시간에 따라 다른 값이 할당 되도록 한다.

  ◯set : 비수치 값을 가진 속성을 애니메이션 하는데 사용된다.

  ◯animateMotion : 요소를 이동 패스에 따라 움직이게 한다.

  ◯animateColor : 시간에 따라 컬러 값을 변경한다.

  ◯animateTransform : SVG 변환 속성을 시간에 따라 변경한다.

Exporting

- D3로 작성된 Image를 파일로 저장하기

  ○Bitmap Image로 저장
  - Print_scrin 으로 화면을 캡쳐한다.

  ○PDF 문서로 저장
  - PDF로 인쇄하는 기능이 있는 프로그램 사용
  - 맥은 별도의 프로그램 필요 없음(운영체제지원)

  ○SVG 문서로 저장
  - SVG코드를 Text_Editor에복사 =>.svg로 저장

References - Canvas

- Web sites
  - http://diveintohtml5.org/canvas.html
  - https://developer.mozilla.org/en/Canvas_tutorial
  - http://www.whatwg.org/specs/web-apps/current-work/multipage/the-canvas-element.html
  - http://www.html5canvastutorials.com
  - http://www.w3schools.com
  - http://kineticjs.com/

- Books
  - HTML5: Guidelines for Web Developers by Klaus Förster and Bernd Öggl
  - HTML5 Canvas by Steve Fulton, Jeff Fulton

참고자료

- **책 (충북대 도서관)**
  - **모바일 컨텐츠 제작을 위한 SVG 프로그래밍**
- 웹
  - **SVG Tool**
    - http://svg-edit.googlecode.com/svn/trunk/editor/svg-editor.html
    - http://sqler.pe.kr/bIISLec/bHTML5/401826
  - javaScript 그래픽 라이브러리 소개
    - http://blog.jidolstar.com/872
  - Pie chart
    - http://jbkflex.wordpress.com/2011/07/28/creating-a-svg-pie-chart-html5/
    - http://bl.ocks.org/mbostock/3887235