Université de Montréal

SVG

Guy Lapalme

DIRO

Scalable Vector Graphics principes

- Éléments XML pour graphisme 2D
- Interprétés par tous les browsers *modernes*, y compris les mobiles, norme W3C (vs Flash)
- S'adapte à l'échelle de présentation
  - exemple du Tigre SVG
- Approche déclarative au graphisme
  - différent de `canvas`
- Modifiable par CSS
- Souvent généré par programme ou logiciel
  - Inkscape, Illustrator, Batik, D3

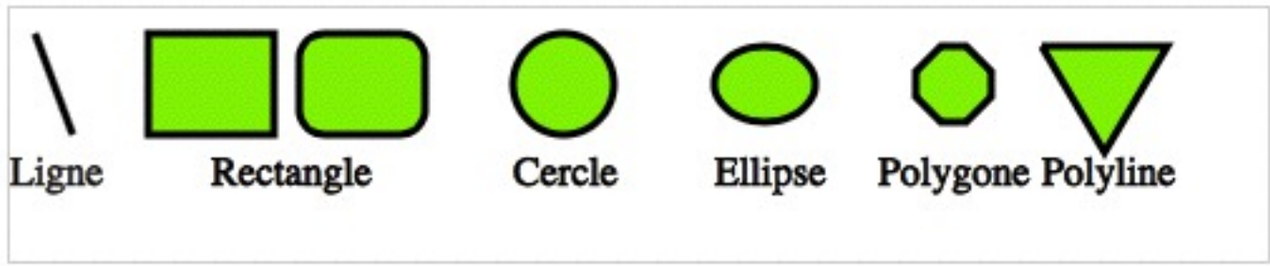
Zone de visualisation

- Espace (canevas) de dessin (infini)
- Rendu dans une zone définie par un élément XML
  - espace de nom XML spécial
  - spécifie la largeur et hauteur du rectangle de rendu
- Espace autonome ou dans une page HTML

```
<svg xmlns="http://www.w3.org/2000/svg"
     xmlns:xlink="http://www.w3.org/1999/xlink"
          width="..." height="...">
     contenu du graphique
</svg>
```

Premier exemple

Formes de base




```
<line x1="10" y1="10" x2="25" y2="50"/>
<rect x="55" y="10" width="50" height="40"/>
<rect x="115" y="10" width="50" height="40" rx="10" ry="10"/>
<circle cx="220" cy="30" r="20"/>
<ellipse cx="300" cy="30" rx="20" ry="15"/>
<polygon points="370,15 380,15 390,25 390,35 380,45
                 370,45 360,35 360,25"/>
<polyline points="410,15 460,15 435,57 410,15"/>
```

Tracé (*path*) à partir d'un point

M : déplacement, L : ligne *x,y*
H : horizontal, V: vertical *d*
Z : retour au début
A : arc *rx,ry rot-x flag-grand-arc,flag-bal x,y*
Q : Bézier quadratique *cont-x,cont-y x,y*
C : Bézier cubique *ct-x1,ct-y2 ct-x2,ct-y2 x,y*
lettre minuscule: coordonnées relatives

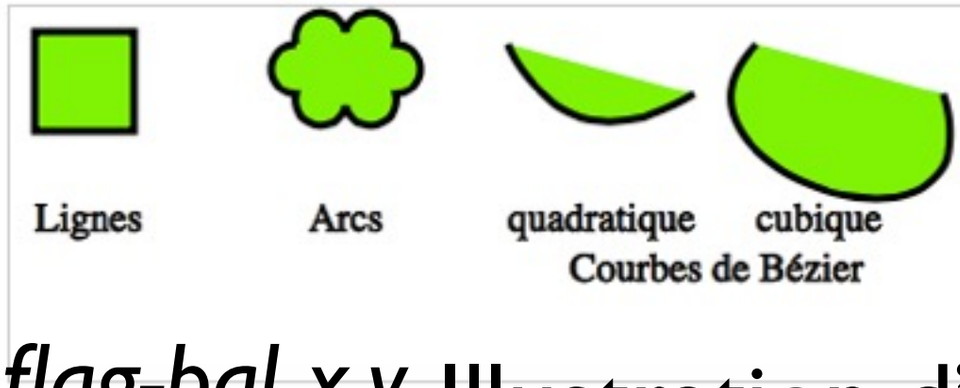


Illustration d'arc

Démo de paths

```
<path d="M10,10 H 50 V 50 H 10 Z"/>

<path d="M115,15 A1,1 0 0,1 135,10 A1,1 0 0,1 155,15
         A1,1 0 0,1 155,35 A1,1 0 0,1 135,40 A1,1 0 0,1 115,35
         A1,1 0 0,1 115,15"/>

<path d="M200,15 q25,50 75,20"/>

<path d="M300,15 c-50,50 100,100 75,20"/>
```

DIRO


```
<defs>
   <path id="myTextPath" d="M275,20 a1,1 0 0,0 100,0"/>
</defs>

<text x="20" y="25" style="font-size: 20">Texte</text>
<text x="100" y="80"
    style="fill:lightgreen; font-size: 75px">Texte</text>
<text x="25">
  <textPath xlink:href="#myTextPath">Texte sur un chemin</textPath>
</text>
<text x="400" y="25">Texte
   <tspan x="400" dy="20">sur plusieurs</tspan>
   <tspan x="400" dy="20">lignes</tspan>
</text>
```

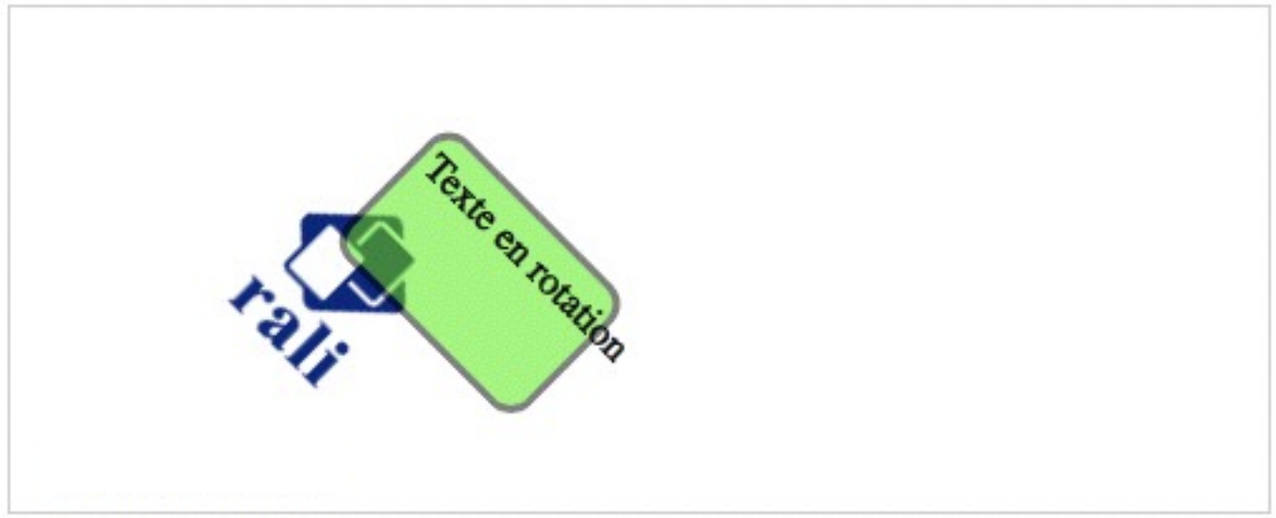
Images, superposition et opacité


```
<image x="120" y="17" width="54" height="71" xlink:href="logoRALI.jpg"/>
<rect x="55" y="25" width="50" height="40"/>
<rect x="115" y="25" width="75" height="65" rx="10" ry="10" opacity=".5"/>
<circle cx="280" cy="65" r="30"/>
<ellipse cx="300" cy="65" rx="30" ry="20" style="fill: blue;" opacity=".7"/>
<circle cx="370" cy="65" r="30" style="fill: red"/>
<circle cx="410" cy="65" r="30" style="fill: green" opacity=".5"/>
<circle cx="390" cy="85" r="30" style="fill: blue;" opacity=".5"/>
<text x="85" y="137">Texte sous la forme</text>
<rect x="55" y="135" width="200" height="40"/>
<text x="85" y="150">Texte sur une forme</text>
```

Transformations
attribut `transform`

- suite d'énoncés
  - `translate(`*`dx`*`,`*`dy`*`)`
  - `rotate(`*`angle`*`)` ou `rotate(`*`angle x`*`,`*`y`*`)`
  - `scale(`*`facteur`*`)`
- s'applique à cet élément et ses dépendants
  - `g, svg`
  - `line, path, text, ...`
- possibilité de cascades

Transformations




```
<g transform="rotate(45 100,100)">
    <image x="100" y="50"  width="54" height="71"
           xlink:href="logoRALI.jpg"/>
    <rect x="115" y="10" width="100" height="65"
          rx="10" ry="10" opacity=".5"/>
    <text x="120" y="30">Texte en rotation</text>
</g>
```

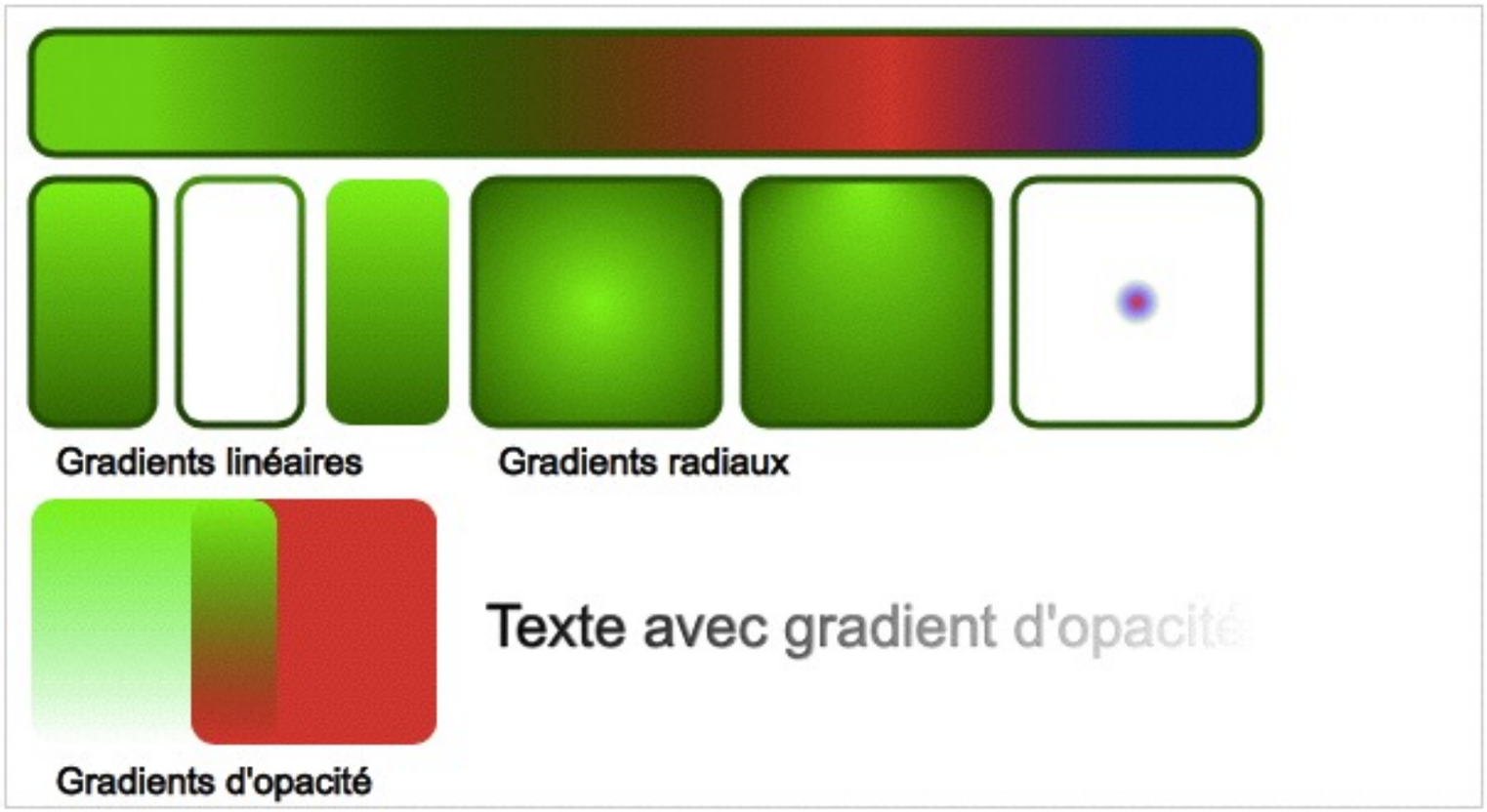
Autres transformations

Viewport vs View Box

- *viewport* : partie visible d'une image (potentiellement infinie)
  - spécifié par `width` et `height` de `svg`
  - unités : `px` par défaut, mais aussi: `em`, `cm`, `mm`, `in`
- *view box* : coordonnées internes du *viewport*
  - spécifié par l'attribut `viewBox` de `svg`
  - 4 nombres : *x y largeur hauteur*

<u>Illustration</u>

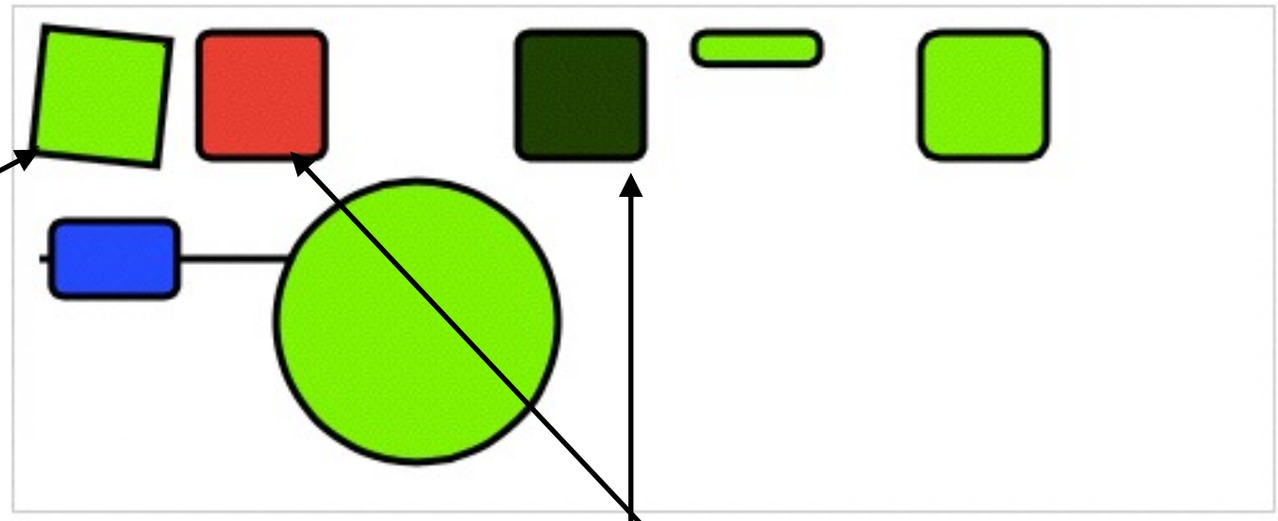
Gradients




```
<linearGradient id="myLinearGradient1"
      x1="0%" y1="0%" x2="100%" y2="0%" spreadMethod="pad">
   <stop offset="10%" stop-color="#00cc00" stop-opacity="1"/>
   <stop offset="30%" stop-color="#006600" stop-opacity="1"/>
   <stop offset="70%" stop-color="#cc0000" stop-opacity="1"/>
   <stop offset="90%" stop-color="#000099" stop-opacity="1"/>
</linearGradient>

<rect x="10" y="10" width="500" height="50" rx="10" ry="10"
    style="fill:url(#myLinearGradient1); stroke: #005000; stroke-width: 3;"/>
```

Animations



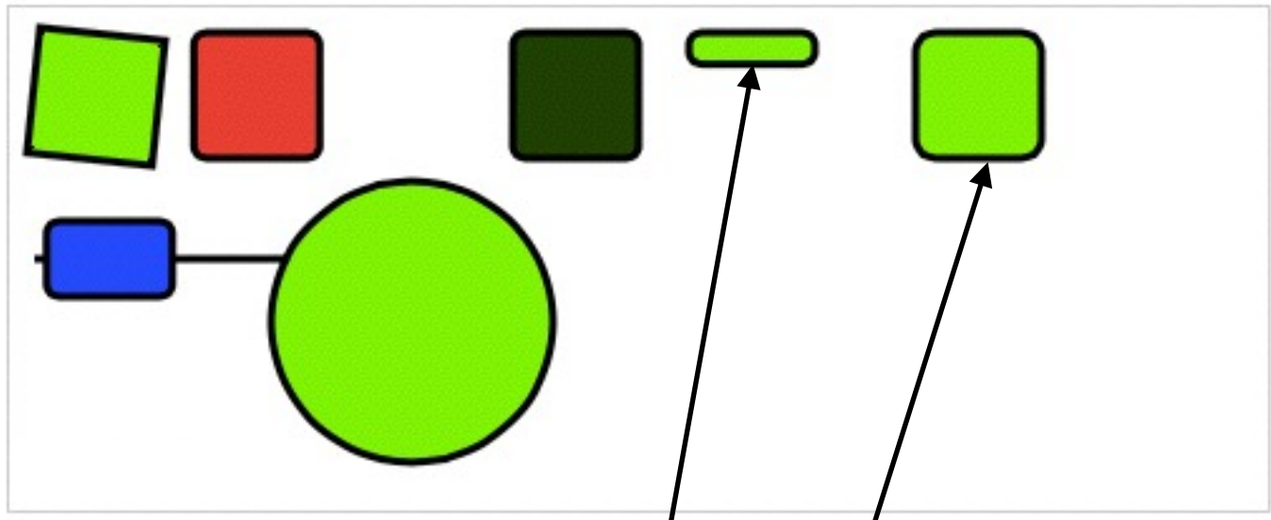
```
<rect x="10" y="10" height="50" width="50">
   <animateTransform attributeName="transform"
       begin="0s" dur="20s" type="rotate"
       from="0 30 30" to="360 30 30" repeatCount="indefinite"/>
</rect>

<rect x="10" y="10" rx="5" ry="5" height="50" width="50" style="fill:red">
   <animate id="et" attributeName="x" begin="0s"
            dur="5s" from="70" to="120" repeatCount="indefinite"/>
</rect>

<rect x="200" y="10" rx="5" ry="5" height="50" width="50">
    <animateColor attributeName="fill" begin="0s" dur="5s" from="#003300"
                  to="#00ff00" repeatCount="indefinite"/>
</rect>
```

DIRO

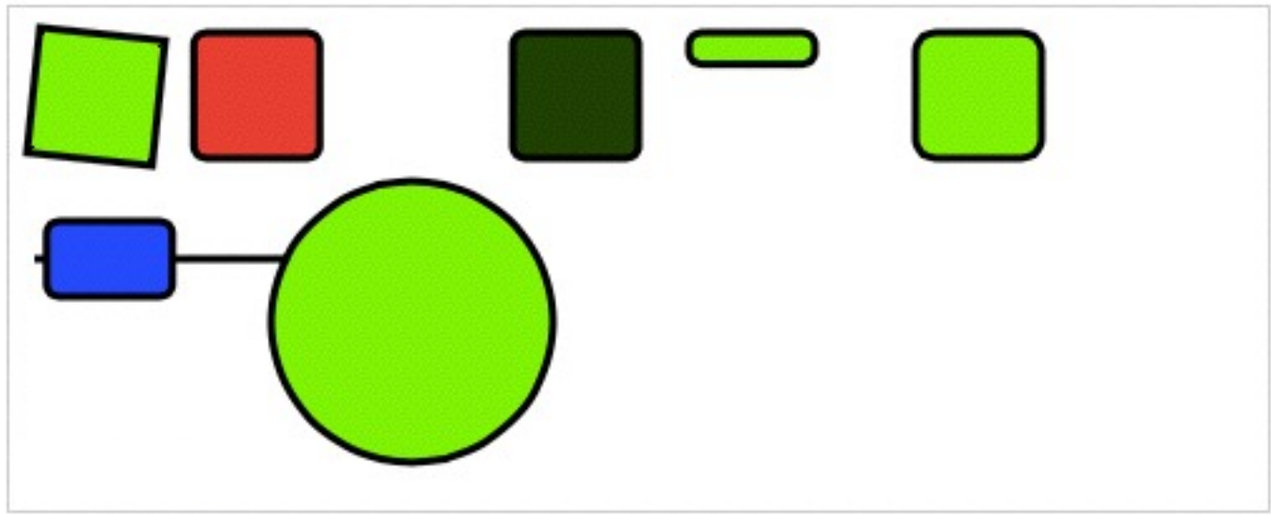
Animations



```
<rect x="270" y="10" rx="5" ry="5" height="50" width="50" >
   <animate attributeName="height" begin="0s" dur="5s"
            from="10" to="50" repeatCount="indefinite"/>
</rect>

<rect x="360" y="10" rx="5" ry="5" height="50" width="50" >
   <animate attributeName="rx" begin="0s" dur="5s"
            from="5" to="50" repeatCount="indefinite"/>
   <animate attributeName="ry" begin="0s" dur="5s"
            from="5" to="50" repeatCount="indefinite"/>
</rect>
```

Animations



```
<path id="chemin"
      d="M10,100 l100,0 a1,1 0 0,1 100,50 a1,1 0 0,1 -100,-50 " />

<rect x="-25" y="-15" rx="5" ry="5" height="30" width="50" style="fill:blue">
   <animateMotion  begin="0s" dur="5s" rotate="auto" repeatCount="indefinite">
     <mpath xlink:href="#chemin"/>
   </animateMotion>
</rect>
```