les écoles supérieures

cpnv Centre professionnel du Nord vaudois

AWA1

ANIMATION WEB

SVG

Contenu

1. Introduction SVG

- **Qu'est-ce que le SVG?**

- SVG est l’abreviation de : Scalable Vector Graphics
- SVG est utilisé pour créer des images vectorielles pour le web
- SVG est écrit en syntaxe XML
- SVG ne perd pas de qualité au redimmensionnement
- Chaque élément et chaque attribut SVG peut être animé
- SVG est une recommendation W3C
- SVG s’intègre dans le DOM

1. Introduction SVG (suite)

1. Introduction SVG (suite)

Possibilité de fallback :

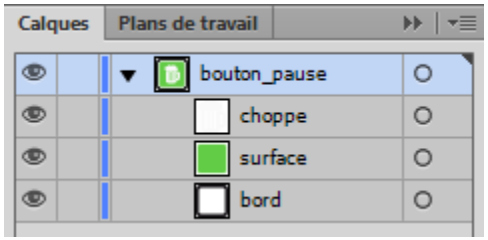
```
<img src="image.svg" onerror="this.onerror=null; this.src='image.png'">
```

ou (en intégrant jQuery + Modernizr.js) :

```
if (!Modernizr.svg) { $(".logo img").attr("src", "images/logo.png"); }
```

2. Création de fichier SVG

1. Créez une image avec Adobe Illustrator (ou Incscape).



2. Recadrez le plan de travail sur les bords de la forme (optimisation).

3. Fichier/Scripts/EnregistrerDocAuFormatSVG (enregistre le fichier bouton_pause.svg)

4. Pour enlever les attributs superflus et optimiser le code:
https://petercollingridge.appspot.com/svg-editor

2. Création de fichier SVG (suite)

Résultat obtenu : 'bouton_pause.svg' fichier optimisé

```
<svg id="bouton_pause" xmlns="http://www.w3.org/2000/svg" x="0" y="0"
 width="229" height="233" viewBox="0 0 229 233">

    <path id="bord" d="M218.3 203.5c0 10-8.45 18.11-18.88 18.11H30.07c-10.43 … "
    style="fill:none;stroke-width:22;stroke:#1A171B"/>

    <path id="surface" d="M209.04 195.92c0 9.12-7.71 16.52-17.21 16.52H37.42 …"
    style="fill:#63CC46;stroke-width:9;stroke:#FFF"/>

    <path id="choppe" d="M160.38 156.85h-12.63v10.03c0 7.17-6.38 12.96-13.55 …"
     fill="#F9F9F9"/>

</svg>
```

3. Inclure SVG

Une image : `<img src="bouton_pause.svg" width="229px" height="233px">`

Une image de fond :

CSS

```
.logo {
display: block;
width: 229px;
height: 233px;
background: url(bouton_pause.svg);
background-size: 229px 233px;
}
```

HTML

```
<a href="/" class="logo"></a>
```

3. Inclure SVG (suite)

Inline : Intégré dans la page html

```
<!DOCTYPE html>
<html>
<body>

<h1>My first SVG</h1>

<svg id="bouton_pause" width="229" height="233"…>
  <path id="bord" d="M218.3 203.5c0 …"/>
  <path id="surface" d="M209.04 1 …"/>
  <path id="choppe" d="M160.38 …"/>
</svg>

</body>
</html>
```

3. Inclure SVG (suite)

Technique inclusion	Anim CSS	Interact CSS
`<img src="mySVG.svg" alt=".." />`	Si dans <svg>	Non
`.el {background: url(mySVG.svg);}`	Si dans <svg>	Non
`<object type="image/svg+xml" data="mySVG.svg"></object>`	Si dans <svg>	Si dans <svg>
`<embed type="image/svg+xml" src="mySVG.svg" />`	Si dans <svg>	Si dans <svg>
`<iframe src="mySVG.svg"><!--fallback--></iframe>`	Si dans <svg>	Si dans <svg>
`<svg><!--SVG content--></svg>`	**Oui**	**Oui**

4. Structure interne SVG

viewPort et viewBox :

```
<svg width="115" height="190" viewBox="0 0 115 190">
  <path="…"/>
</svg>
```

- Pour éviter que les éléments soient coupés, la viewBox doit avoir la même taille que le viewPort, la section visible du svg (width/height)

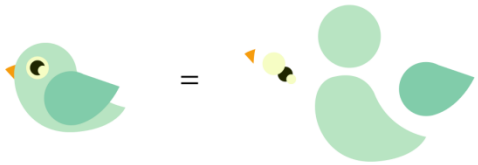
- Si on devait redimensionner le viewPort (pour du responsive) on peut faire en sorte que la viewBox s'adapte tout en gardant les proportions x/y) avec `preserveAspectRatio`

```
<svg width="350" height="150" viewBox="0 0 300 300"
 preserveAspectRatio="xMinYMax meet">
  <path="…"/>
</svg>
```

4. Structure interne SVG (suite)

Groupe : <g> (Reprend les groupes d'Adobe Illustrator)

```
<svg width="300px" height="400px" viewBox="0 0 300 400">
   <g id="bird">
      <g id="body">
          <path id="trunk" d="M56,48.42,78.11…"/>
          <path id="wing" d="M75.09,105.78,75…"/>
      </g>
      <g id="head">
         <path id="beak" d="M50.43,68.52c0,…"/>
         <path class="eye-ball" d="M60.53,71…"/>
         <path id="pupil" d="M67.82,79.22,64…"/>
       </g>
   </g>
</svg>
```



4. Structure interne SVG(suite)

Utiliser : <use> Permet de réutiliser des éléments

Permet de réutiliser (donc afficher) le groupe 'bird' défini précédemment :

```
<use x="100" y="100" xlink:href="#bird" />
```

Symbole : <symbol>

Identique a <g> mais n'affiche pas l'élément.

On créé un symbole pour l'utiliser plus tard avec <use>.

4. Structure interne SVG (suite)

Définir : <def> Permet de définir un élément a utiliser ultérieurement

```
<svg>
    <defs>
        <linearGradient id="gradient">
            <stop offset="0%" style="stop-color: deepPink"></stop>
            <stop offset="100%" style="stop-color: #009966"></stop>
        </linearGradient>
    </defs>

    <rect stroke="#eee" stroke-width="5" fill="url(#gradient)"></rect>
</svg>
```

5. Formes SVG

Rectangle

```
<svg width="400" height="400">

<rect x="10" y="40" width="300" height="50"
 style="fill:rgb(100,200,200);stroke-width:5;stroke:rgb(0,0,0)" />

<rect x="50" y="10" rx="20" ry="20" width="200" height="150"
  style="fill:green;stroke:black;stroke-width:5;opacity:0.5" />

</svg>
```

5. Formes SVG (suite)

Cercle et Ellipse

```
<svg height="400" width="400">

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

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

</svg>
```

5. Formes SVG (suite)

Ligne et Polyline

```
<svg height="100" width="100">

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

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

</svg>
```

```
points="x1,y1 x2,y2 x3,y3 …"
```

5. Formes SVG (suite)

Polygone

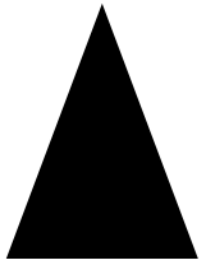
```
<svg height="200" width="200">

<polygon points="200,10 250,190 160,210"
style="fill:lime;stroke:purple;stroke-width:1" />

</svg>
```

`points="x1,y1 x2,y2 x3,y3 …"`

5. Formes SVG (suite)

Tracé : Path

```
<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

6. Traits SVG

Traitillé : stroke-dasharray

```
<svg height="80" width="300">
  <g fill="none" stroke="black" stroke-width="4">
    <path stroke-dasharray="5,5" d="M5 20 l215 0" />
    <path stroke-dasharray="10,10" d="M5 40 l215 0" />
    <path stroke-dasharray="20,10,5,5,5,10" d="M5 60 l215 0" />
  </g>
</svg>
```

6. Traits SVG (suite)

Extrémités des traits : stroke-linecap

```
<svg height="80" width="300">
  <g fill="none" stroke="black" stroke-width="6">
    <path stroke-linecap="butt" d="M5 20 l215 0" />
    <path stroke-linecap="round" d="M5 40 l215 0" />
    <path stroke-linecap="square" d="M5 60 l215 0" />
  </g>
</svg>
```

6. Traits SVG (suite)

Liaison des traits : stroke-linejoin

```
<svg height="80" width="300">
  <g fill="none" stroke="black" stroke-width="6">
     <path stroke-linejoin="miter" d="M5 20 l215 0 …" />
     <path stroke-linejoin="round" d="M5 20 l215 0 …" />
     <path stroke-linejoin="bevel" d="M5 20 l215 0 …" />
  </g>
</svg>
```

7. Mise en forme CSS

Exemple de style et transition de SVG inline avec CSS

.CSS

```
<style>
   circle { fill: red; transition: .3s ease-out; }
   circle:hover { fill: #009966; }
</style>
```

.htm

```
<svg height="300" width="300">
   <circle cx="100" cy="100" r="75" style="fill:red;" />
</svg>
```

A voir : http://tutsplus.github.io/Styling-Iconic/styling/index.html

7. Mise en forme CSS responsive

```
<div class="svg-container">
   <svg viewBox="0 0 500 500" preserveAspectRatio="xMinYMin meet"
    class="svg-content">
      <circle fill="#F7941E" stroke="#231F20" stroke-width="10"
       cx="250" cy="250" r="200" opacity="0.6" />
   </svg>
</div>
```

```
.svg-container {
   display: inline-block;
   position: relative;
   width: 100%;
   padding-bottom: 100%;
   vertical-align: middle;
   overflow: hidden;
}
```

```
.svg-content {
   display: inline-block;
   position: absolute;
   top: 0;
   left: 0;
}
```

8. Texte SVG

Texte simple

```
<svg width="620" height="100">
   <text x="30" y="90" fill="#ED6E46" font-size="100" font-family="'Leckerli'">
       Watermelon
  </text>
</svg>
```

Le texte s'écrit dans une boite de taille 30x90

Autres attributs te text : textLength, lengthAjust, kerning, letterSpacing

8. Texte SVG (suite)

Texte sur plusieurs lignes

```
<svg width="775" height="500">
   <text x="15" y="90" fill="#ED6E46" font-size="60" font-family="'Leckerli One'">
       Watermelons
      <tspan dy="-30" fill="#bbc42a" font-size="80">are</tspan>
      <tspan dy="50">delicious</tspan>
   </text>
</svg>
```

8. Texte SVG (suite)

Texte sur un tracé:

```
<svg width="620" height="200">
  <defs>
    <path id="testPath" d="M3.858,58.607 …" />
  </defs>
  <text x="2" y="40%" fill="#765373" font-size="30" font-family="'Lato'">
    <textPath xlink:href="#testPath">
      There are over 8,000 grape varieties  worldwide.
    </textPath>
  </text>
</svg>
```

9. Remplissage SVG

Définition et utilisation de dégradé

```
<svg width="405" height="105">
  <defs>
    <linearGradient id="Gradient1" x1="0" y1="0" x2="100%" y2="0">
      <stop offset="0%" stop-color="#BBC42A" />
      <stop offset="100%" stop-color="#ED6E46" />
    </linearGradient>
  </defs>
  <rect x="2" y="2" width="400" height="100" fill="url(#Gradient1)"
   stroke="#333333" stroke-width="4px" />
</svg>
```

Dégradé radial : radialGradient

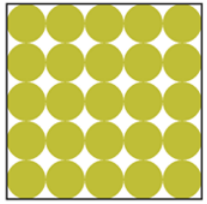
```
<svg width="850px" height="300px">
   <defs>
      <radialGradient id="Gradient2" cy="60%" fx="95%" fy="70%" r="2">
         <stop offset="0%" stop-color="#ED6E46" />
         <stop offset="10%" stop-color="#b4c63b" />
         …
         <stop offset="90%" stop-color="#503969" />
         <stop offset="100%" stop-color="#ab6294" />
      </radialGradient>
   </defs>
   <text x="20%" y="75%" fill="url(#Gradient2)"
    font-family="Signika" font-size="200">Cherry</text>
 </svg>
```

fx,fy : position du centre du dégradé
cx,cy,r : position extérieure du dégradé

9. Remplissage SVG (suite)

Motif : pattern

```
<svg width="220" height="220">
   <defs>
      <pattern id="basicPattern" x="10" y="10" width="40" height="40"
       patternUnits="userSpaceOnUse">
         <circle cx="20" cy="20" r="20" fill= "#BBC42A" />
      </pattern>
   </defs>
   <rect x="10" y="10" width="200" height="200" stroke="#333333" stroke-
   width="2px" fill="url(#basicPattern)" />
</svg>
```



10. Transformations SVG

Translation : transform="translate(<tx> <ty>)"

```
<circle cx="0" cy="0" r="100" transform="translate(100 300)" />
```

Échelle : transform="scale(<s>)" ou transform="scale(<sx> <sy>)"

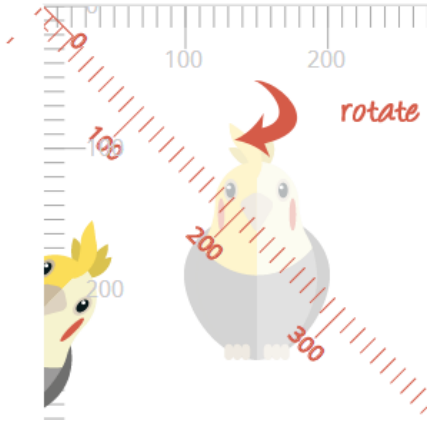
```
<rect width="150" height="100" transform="scale(2 0.5)" x="0" y="0" />
```

Inclinaison transform="skewX(<x>)" ou transform="skewY(<y>)"

10. Transformations SVG (suite)

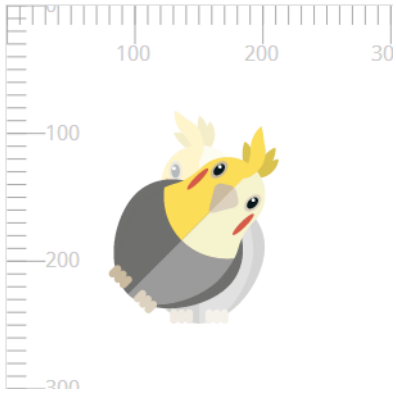
Rotation simple : transform="rotate(<rotation angle>)

```
<g id="parrot" transform="rotate(45)">
```



Rotation centrée : transform=rotate(<rotation angle> <cx> <cy>)

```
<g id="parrot" transform="rotate(45 150 170)">
```



Equivalent :
transform="translate(150 170) rotate(45) translate(-150 -170)">

11. Filtres SVG

Liste des filtres SVG

- <feBlend>
- <feColorMatrix
- <feComponentTransfer>
- <feComposite>
- <feConvolveMatrix>
- <feDiffuseLighting>
- <feDisplacementMap>
- <feFlood>
- <feGaussianBlur>
- <feImage>
- <feMerge>
- <feMorphology>
- <feOffset>
- <feSpecularLighting>
- <feTile>
- <feTurbulence>
- <feDistantLight>
- <fePointLight>
- <feSpotLight>

Compliqué et le résultat n'est pas toujours bon. Nous allons voir 2 exemples utiles :

11. Filtres SVG (suite)

Flou gaussien : feGaussianBlur

```
<defs>
   <filter id="blurF" y="-5" height="40"/>
        <feGaussianBlur in="SourceGraphic" stdDeviation="3"/>
   </filter>
</defs>
<ellipse cx="55" cy="60" rx="25" ry="15" style="fill:#0000ff; "/>
<ellipse cx="155" cy="60" rx="25" ry="15" style="fill:#0000ff; filter:url(#blurF);"/>
```

11. Filtres SVG (suite)

Ombre portée : feMerge de feOffset et feGaussianBlur

```
<defs>
    <filter id="blurF" y="-10" height="40" x="-10" width="150">
        <feOffset in="SourceAlpha" dx="3" dy="3" result="offset2" />
        <feGaussianBlur in="offset2" stdDeviation="3" result="blur2"/>
        <feMerge>
            <feMergeNode in="blur2" />
            <feMergeNode in="SourceGraphic" />
        </feMerge>
    </filter>
</defs>
<ellipse cx="55" cy="60" rx="25" ry="15" style="fill:#0000ff; filter: url(#blurF);" />
```

12. Animation SVG

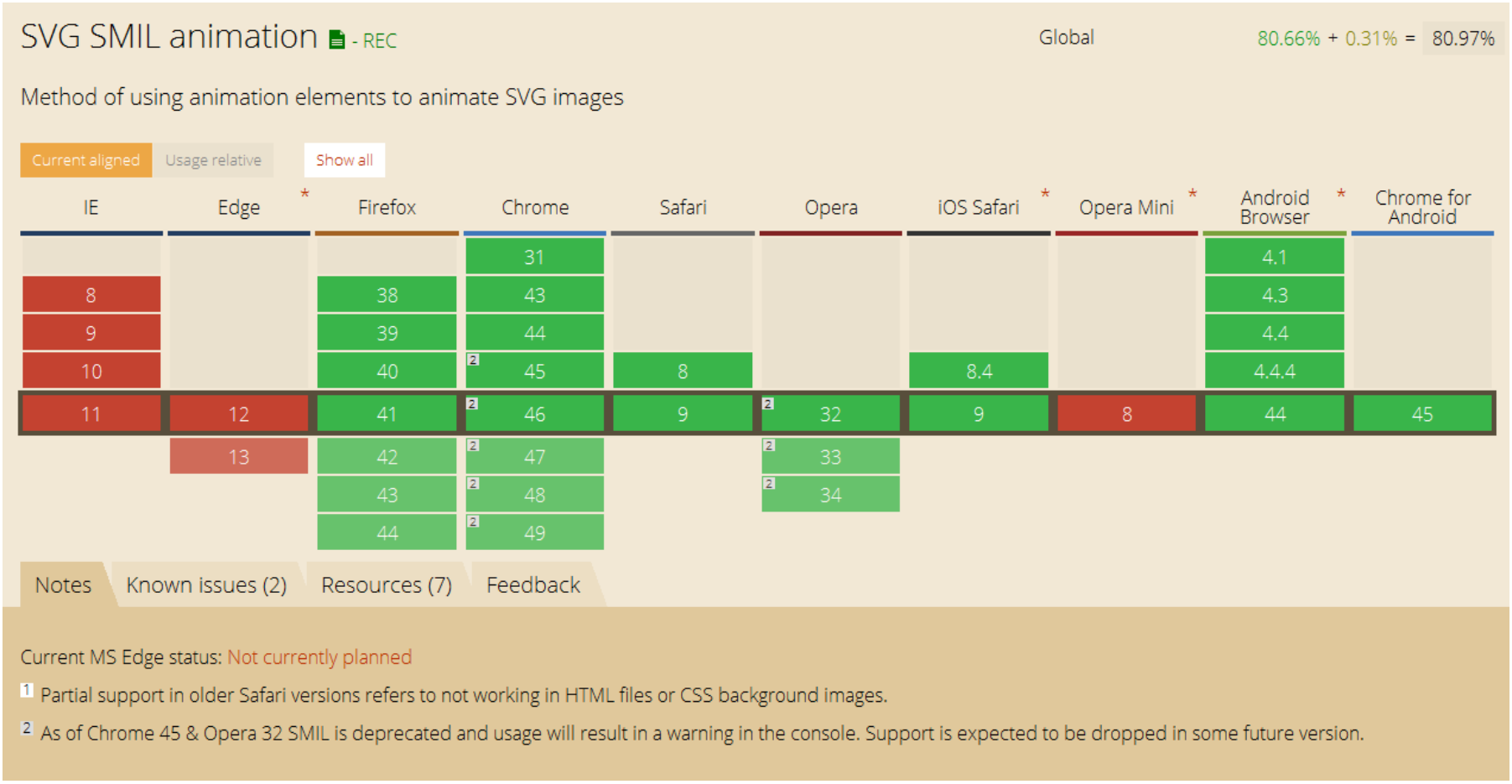
- Pour des transitions et animations de base, (interactivité sur click et hover) utilisez les transitions et animations CSS.
- Pour des animations plus complexes, on utilisera de préférence des librairies javascript : Snap.svg, velocity (léger) ou GSAP.
- Attention : Le langage SMIL, intégré au format svg, est en voie de displarition. Il n'est pas compatible IE et dépréciation (Chrome).

12. Animation avec SMIL

Synchronized Multimedia Integration Language

`<animate>`	Pour animer les propriétés des SVG.
`<animateTransform>`	Pour animer les transformations SVG
`<animateMotion>`	Pour animer un SVG sur un path SVG.
`<set>`	Pour modifier des valeurs non numériques (ex visibility)

12. Animation avec SMIL



12. Animation avec SMIL

id de l'anim ->

Element svg que l'on anime ->

Attribut qu'on anime ->

Pos départ a pos arrivée ->

Durée ->

Reste a la fin (sinon au début) ->

Nb répétitions ->

Evénement déclencheur ->

```
<animate
id="move"
link:href="#myCircle"
attributeName="cx"
attributeType="XML"
from="50" to="450"
dur="5s"
fill="freeze"
repeatCount="indefinite"
begin="click"
 />
```

12. Animation avec SMIL

Synchronisation et redémarrage:

```
<circle id="orange-circle" r="30" cx="50" cy="50" fill="orange" />
<rect id="blue-rectangle" width="50" height="50" x="25" y="200" fill="#0099cc">
</rect>

<animate xlink:href="#orange-circle" attributeName="cx" from="50" to="450" dur="5s"
 fill="freeze" begin="click" restart="whenNotActive" id="circ-anim"/>

<animate xlink:href="#orange-circle" attributeName="fill" from="orange" to="#0099aa"
dur="2s" fill="freeze" begin="circ-anim.begin + 5s" id="circ-color-anim"/>

<animate xlink:href="#blue-rectangle" attributeName="x" from="50" to="425" dur="5s"
fill="freeze" begin="circ-anim.begin + 1s" id="rect-anim"/>
```