Lezione 2:

SVG JAVASCRIPT SMIL

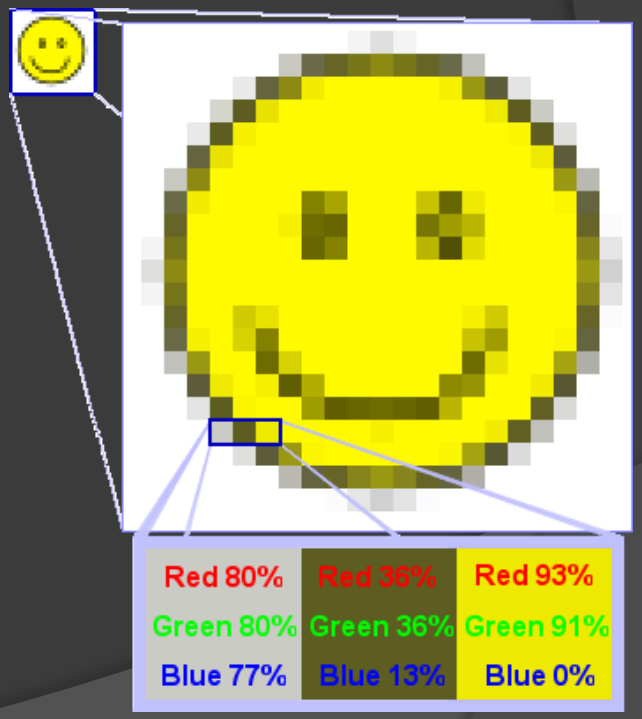
Presentazioni:

- Renato Mainetti (Grafica Programmazione Reti)
- Mail per info: renato.mainetti@gmail.com
  (grande fantasia)
- Sito Web da cui scaricare le slide :
  http://tamberlo.altervista.org

Grafica Raster:

Raster (trama, reticolo, griglia) indica la griglia ortogonale di punti che costituisce un'immagine raster. Nella grafica raster l'immagine viene vista come una scacchiera e ad ogni elemento della scacchiera, chiamato pixel, viene associato uno specifico colore.

- Matrice di pixel (w*h)
- Composta da canali 1,3,4
- Canale trasparenza Alpha
- Vari Formati: bmp, jpg, gif,png



Grafica Vettoriale:

Nella grafica vettoriale un'immagine è descritta mediante un insieme di primitive geometriche che definiscono punti, linee, curve e poligoni ai quali possono essere attribuiti colori e anche sfumature.

È radicalmente diversa dalla grafica raster in quanto nella grafica raster le immagini vengono descritte come una griglia di pixel opportunamente colorati.

Chi vince tra i due ?

http://designwashere.com/design-battle-vector-vs-raster/

Grafica Vettoriale:

VANTAGGI:

- possibilità di esprimere i dati in una forma direttamente comprensibile ad un essere umano (es. lo standard SVG);
- possibilità di esprimere i dati in un formato che occupi (molto) meno spazio rispetto all'equivalente raster;
- possibilità di ingrandire l'immagine arbitrariamente, senza che si verifichi una perdita di risoluzione dell'immagine stessa.

SVANTAGGI:

- Un'immagine vettoriale molto complessa può essere molto *corposa computazionalmente* e richiedere l'impiego di un computer molto potente per essere elaborata.

Grafica Vettoriale:

RICORDATE IL DISCORSO

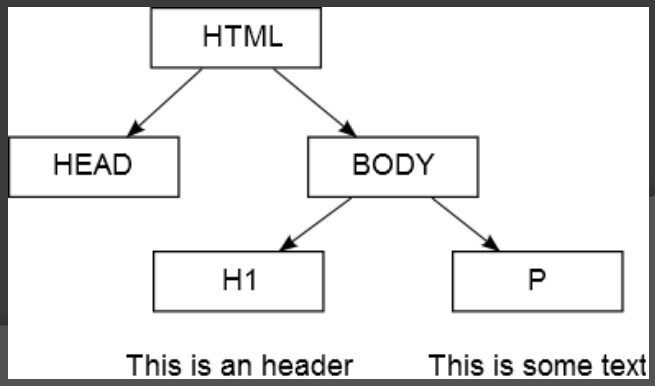
COMPRESSIONE
Vs.
COMPLESSITA’

GLOSSARIO:

XML: (sigla di **eXtensible Markup Language**) è un metalinguaggio di markup, ovvero un linguaggio marcatore che definisce un meccanismo sintattico che consente di estendere o controllare il significato di altri linguaggi marcatori.

Metalinguaggio: Nella logica e nella teoria dei linguaggi formali per **metalinguaggio** si intende un linguaggio formalmente definito che ha come scopo la definizione di altri linguaggi artificiali che diciamo *linguaggi obiettivo (xhtml, svg, etc.)*

DOM: Il **Document Object Model** è lo standard ufficiale del W3C per la rappresentazione di documenti strutturati in maniera da essere neutrali sia per la lingua che per la piattaforma. Permette a progammi e script di accedere dinamicamente e modificare i contenuti, struttura e stile del documento.



GLOSSARIO:

SMIL: SMIL (Synchronized Multimedia Integration Language) è un linguaggio basato sull' XML, è utilizzato per realizzare presentazioni multimediali interattive.
E' in grado di combinare audio, video, ipertesti, immagini nel tempo e nello spazio, permettendo di effettuare delle transizioni.

JavaScript: è un linguaggio di scripting orientato agli oggetti comunemente usato nei siti web. JavaScript è stato standardizzato per la prima volta tra il 1997 e il 1999 dalla ECMA con il nome **ECMAScript**.

What is SVG?

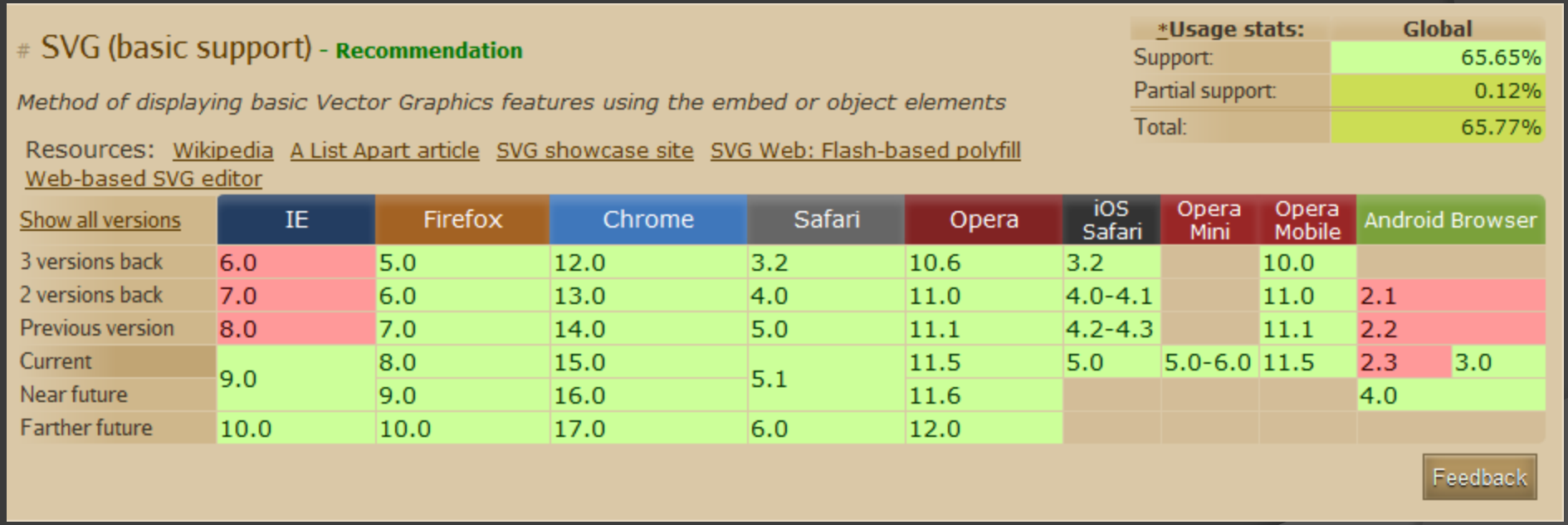
Scalable Vector Graphics (SVG) is like HTML for graphics. It is a markup language for describing all aspects of an image or Web application, from the geometry of shapes, to the styling of text and shapes, to animation, to multimedia presentations including video and audio. It is fully interactive, and includes a scriptable DOM as well as declarative animation (via the SMIL specification). It supports a wide range of visual features such as gradients, opacity, filters, clipping, and masking.

The use of SVG allows fully scalable, smooth, reusable graphics, from simple graphics to enhance HTML pages, to fully interactive chart and data visualization, to games, to standalone high-quality static images. SVG is natively supported by most modern browsers (with plugins to allow its use on all browsers), and is widely available on mobile devices and set-top boxes. All major vector graphics drawing tools import and export SVG, and they can also be generated through client-side or server-side scripting languages.

SVG:

Ma è supportato dai Browser ?

http://caniuse.com/#cats=SVG



SVG (basic support) - **Recommendation**

Method of displaying basic Vector Graphics features using the embed or object elements

Resources: Wikipedia, A List Apart article, SVG showcase site, SVG Web: Flash-based polyfill, Web-based SVG editor

*Usage stats:	Global
Support:	65.65%
Partial support:	0.12%
Total:	65.77%

Show all versions	IE	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Opera Mobile	Android Browser	
3 versions back	6.0	5.0	12.0	3.2	10.6	3.2		10.0		
2 versions back	7.0	6.0	13.0	4.0	11.0	4.0-4.1		11.0	2.1	
Previous version	8.0	7.0	14.0	5.0	11.1	4.2-4.3		11.1	2.2	
Current	9.0	8.0	15.0	5.1	11.5	5.0	5.0-6.0	11.5	2.3	3.0
Near future		9.0	16.0		11.6				4.0	
Farther future	10.0	10.0	17.0	6.0	12.0					

Feedback

SVG: quanto è fico…

1. W3C Standards
SVG is an open standard and developers may use it without restrictions.

2. Image scaling
SVGs can be scaled to any size without incurring pixelation or loss of detail.

3. Smaller file size
Simple images such as logos and charts will generally have a smaller file size than their bitmapped JPG, PNG or GIF equivalent.

4. SVG tools are already available
SVG XML code can be created, verified, manipulated and compressed using a variety of existing tools from Notepad to Inkscape, OpenOffice.org Draw, and Microsoft Visio.

SVG: quanto è fico…

5. Server-side generation
SVG XML can be created and manipulated on the server using PHP, .NET, Python or any other language/framework.

6. Client-side generation
SVG XML can be created and manipulated on the client using JavaScript to create dynamic effects and animation. Event handlers, such as click or mouseover, can be applied to any SVG element.

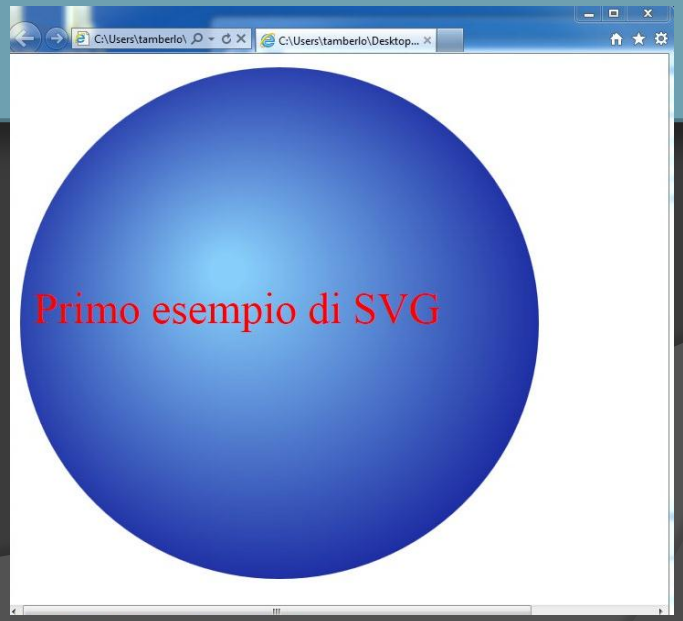
7. Compatibility
Although the facilities offered by SVG rendering engines may differ, the format is backward and forward compatible. SVG engines will render what they can and ignore the rest.

8. Accessibility
SVGs are accessible; text and drawing elements are machine-readable so screen readers can other devices can parse the images.

9. Search Engine Optimization
SVGs offer improved SEO(search engine optimization) because Google, Yahoo, Bing, and other search engines can index an image's content.

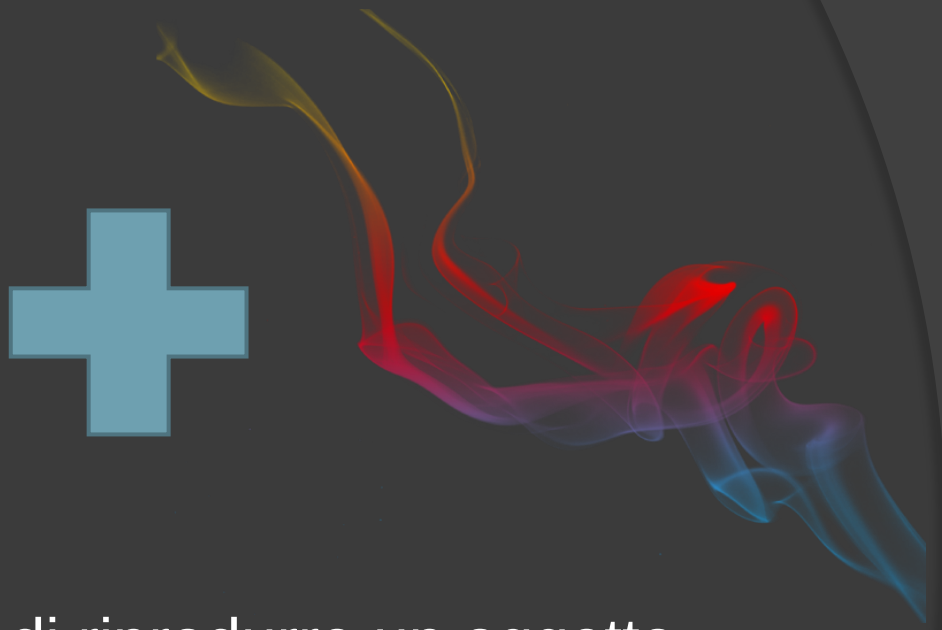
Grafica Vettoriale:

```
<?xml version="1.0" encoding="iso-8859-1" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//Dtd SVG 1.1//EN" "http://www.w3.org/Graphics/SVG/1.1/Dtd/svg11.dtd">

<svg width="300" height="200" version="1.1" xmlns="http://www.w3.org/2000/svg">

        <defs>
                <radialGradient id="rg" cx="100" cy="100" fx="80" fy="80" gradientUnits="userSpaceOnUse">
                <stop offset="5%" stop-color="lightskyblue" />
                <stop offset="100%" stop-color="darkblue" />
                </radialGradient>
        </defs>

        <circle id="circle_1" cx="100" cy="100" r="95" fill="url(#rg)"/>

        <text x="10" y="100" style="fill:red;font-family:times;font-size:18">Primo esempio di SVG</text>

</svg>
```



Disegnamo scrivendo del testo…

SVG:



Il ***mock-up*** è l'attività di riprodurre un oggetto o modello in scala ridotta o maggiorata

Oggi l'attività del *mock-up artist* si rivolge anche al *web design*, con la creazione di modelli di pagine web eseguiti mediante software grafico (ad esempio Photoshop) e successivamente tramite HTML e CSS.

SVG:

Synergy with Other XML Technologies:

One of the strategic strengths of SVG is that it is XML based. SVG can be handled just like any other type of XML, transformed by XSLT, or combined with other XML vocabularies, such as XHTML, XSL-FO (Extensible Stylesheet Language Formatting Objects), SMIL (the Synchronized Multimedia Integration Language), and can be modified by CSS.

SVG: (struttura)

```
<?xml version="1.0" encoding="iso-8859-1" standalone="no"?>
```
// Dichiara che il file in questione è xml conforme a standard 1.0

```
<!DOCTYPE svg PUBLIC "-//W3C//Dtd SVG 1.1//EN“
"http://www.w3.org/Graphics/SVG/1.1/Dtd/svg11.dtd">
```
//informiamo il programma che si occuperà di visualizzare l’immagine che il file SVG è stato scritto seguendo le regole della versione 1.1 del linguaggio.

```
<svg width="300" height="200" version="1.1"
xmlns="http://www.w3.org/2000/svg">
```
//il tag <svg> è il tag che contiene tutti gli elementi grafici della nosta immagine.
L'attributo **xmlns** serve ad indicare il namespace a cui fanno riferimento i tag del file SVG. In generale la dichiarazione del namespace, in un documento XML, indica a quale linguaggio appartengono i tag utilizzati all'interno del file.

```
    <text x="10" y="100" style="fill:red;font-family:times;font-size:18">
            Hello World :)
    </text>
```
// la dichiarazione dell'unico elemento grafico che contiene la nostra immagine

```
 </svg>
```

SVG:

Direttamente incluso in una pagina HTML:

```
<object width="300" height="200" data="hello.svg" type="image/xml+svg">

   Scarica il plugin SVG per visualizzare l'immagine

</object>
```

SVG & CSS:

SVG è un linguaggio basato su XML e come tale può avvalersi del meccanismo dei CSS (Cascading Style Sheets) per la definizione dello stile, proprio come avviene in HTML e XHTML.

In SVG possiamo definire lo stile di un elemento, seguendo la sintassi CSS, all'interno dell'elemento <style>.

Vediamo quindi l'uso dei CSS, considerando il seguente esempio:

```
<svg viewbox="0 0 300 300">
   <style type="text/css">
         <![CDATA[
               rect {stroke:black;fill:red;stroke-width:5}
         ]]>
   </style>
   <rect x="10" y="10" width="200" height="100"/>
</svg>
```

SVG & CSS:

Vale per tutti gli elementi rect contenuti nella nostra immagine SVG.
Nel caso di immagini complesse (quindi con molti elementi grafici) ci evita di dover definire lo stile di ogni singolo elemento.

Qualora avessimo la necessità di definire uno stile diverso per un particolare elemento, ad esempio disegnare tutti i rettangoli rossi tranne uno blu, in CSS dovremmo scrivere:

```
<svg viewbox="0 0 300 300">
   <style type="text/css">
   <![CDATA[
      rect {stroke:black;fill:red;stroke-width:5}
      rect.coloreblu {stroke:black;fill:blue;stroke-width:5}
   ]]>
   </style>    <rect x="10" y="10" width="50" height="100"/>
   <rect class="coloreblue" x="110" y="10" width="50" height="100"/>
</svg>
```

SVG: Grouping

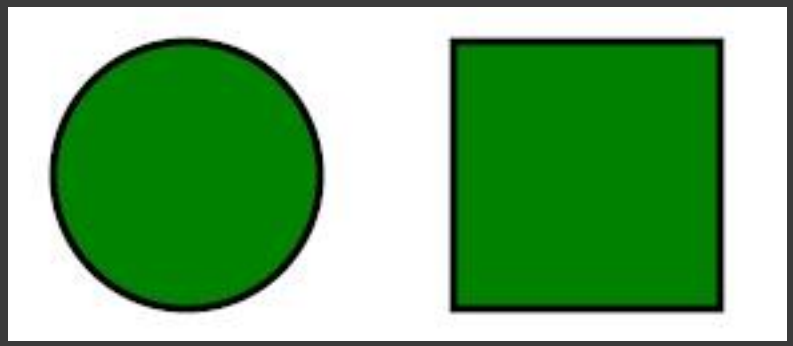
(raggruppare più elementi) -> TAG <g> </g>

Creato un gruppo, ogni attributo che andremo ad inserire all'interno dell'elemento <g>, verrà condiviso da tutti gli elementi grafici appartenenti al gruppo.

Ad esempio:

```
<g style="fill:green;stroke:black;stroke-width:3">
   <circle cx="100" cy="100" r="50"/>
   <rect x="200" y="50" width="100"  height="100"/>
</g>
```

Produrrà:



P.S. E' Possibile creare anche sotto gruppi !

SVG: Trasformazioni

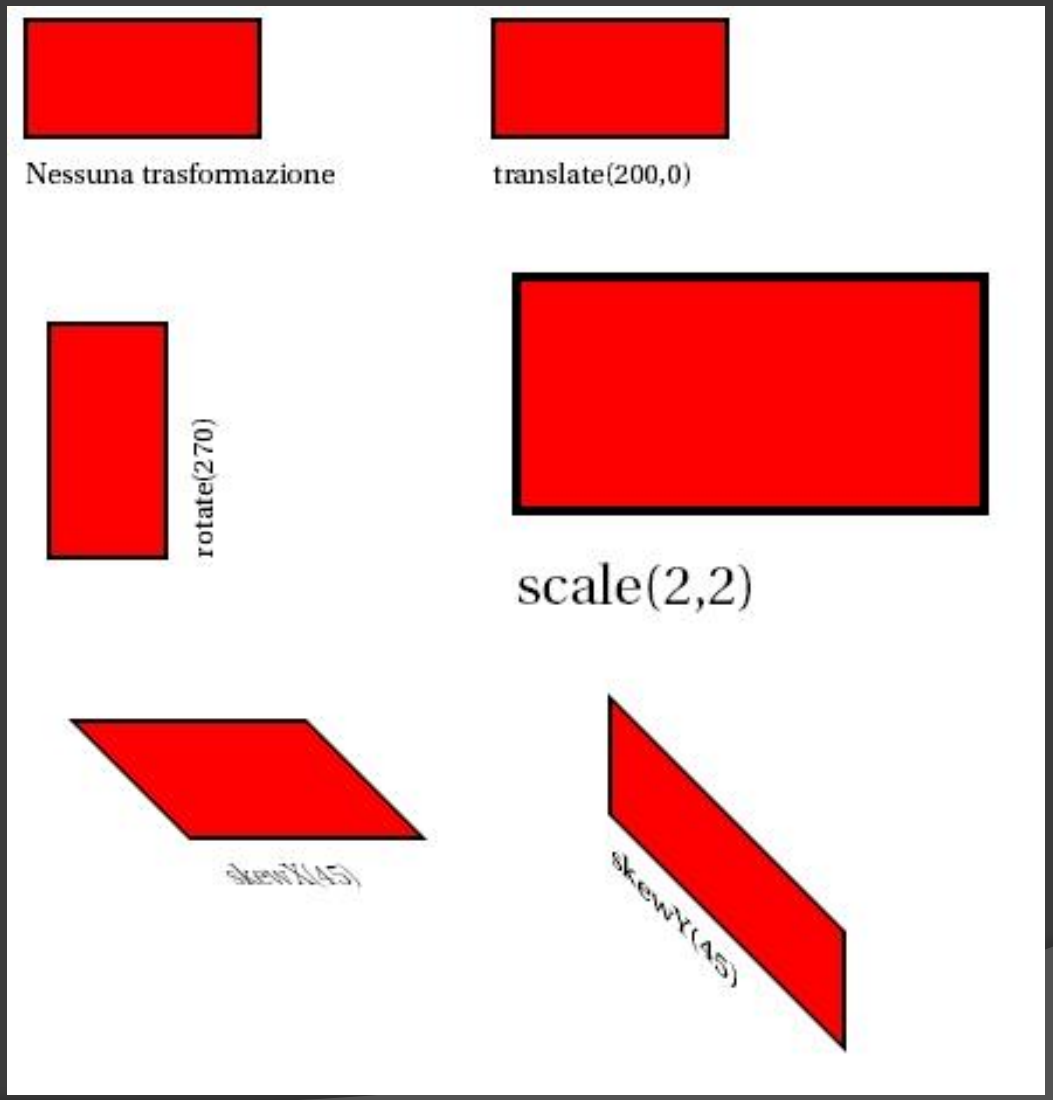
In SVG è possibile applicare alcune trasformazioni agli elementi grafici utilizzando l'attributo **transform**.

Le trasformazioni possibili sono:

- **translate(tx,ty): sposta su x e y di tx,ty**
- **scale(sx,sy): scala su x e y con valore sx,sy**
- **rotate(ra,cx,cy): ra -> angolo rotazione cx,cy opzionali centro rotazione**
- **skewX(sa): inclina asse x di angolo pari a sa**
- **skeyY(sa): inclina asse y di angolo pari a sa**
- **matrix(a,b,c,d,e,f)**: applica all'elemento grafico la matrice di trasformazione (3x3) indicata da [[a c e] [b d f] [0 0 1]].

http://mathforum.org/mathimages/index.php/Transformation_Matrix

SVG: risultati delle Trasformazioni



SVG: link <a>

Per inserire dei link esterni in SVG bisogna utilizzare l'elemento <a> la cui funzione è molto simile all'analogo tag esistente in HTML.
Vediamo subito un esempio di link esterno:

```
<svg width="200" height="200“
xmlns:xlink="http://www.w3.org/1999/xlink">
<a xlink:href="http://www.prova.it">
   <rect x="10" y="10" width="180" height="100" style="fill:yellow;stroke:black;stroke-width:2"/>
   <text x="55" y="60" style="font-width:bold;font-family:times;font-size:24;fill:black">HTML.it</text>
</a>
</svg>
```

SVG: assegnamo un id

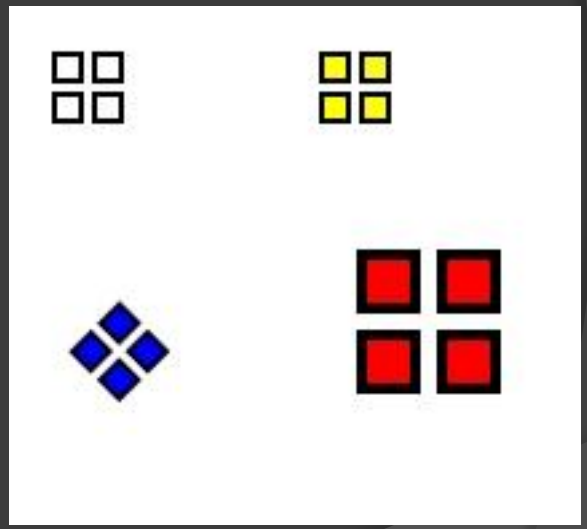
A ciascun elemento grafico contenuto all'interno di un'immagine SVG, possiamo associare un nome per identificarlo in maniera agevole, anche attraverso Javascript. Per fare ciò dobbiamo utilizzare l'attibuto **id**.

```
<rect id="retRosso" x="10" y="10" width="100" heigth="100" style="fill:red;stroke:black"/>

<rect id="retGiallo" x="10" y="10" width="100" heigth="100" style="fill:yellow;stroke:black"/>
```

SVG: <Symbol> & <Use>

```
<svg width="500" height="500"
xmlns:xlink="http://www.w3.org/1999/xlink">
<defs>
  <symbol id="template1">
    <rect x="5" y="5" width="10" height="10"/>
    <rect x="20" y="5" width="10" height="10"/>
    <rect x="20" y="20" width="10" height="10"/>
    <rect x="5" y="20" width="10" height="10"/>
  </symbol>
</defs>
<use x="50" y="50" xlink:href="#template1"
style="fill:none;stroke:black;stroke-width:2"/>
<use x="150" y="50" xlink:href="#template1"
style="fill:yellow;stroke:black;stroke-width:2"/>
<use x="50" y="150" xlink:href="#template1"
transform="translate(150,0) rotate(45)"
style="fill:blue;stroke:black;stroke-width:2"/>
<use x="80" y="60" xlink:href="#template1" transform="scale(2)"
style="fill:red;stroke:black;stroke-width:2"/>
</svg>
```

Symbol: insieme di primitive
Use: specifica uso del symbol
In posizione x,y usando il link interno.



SVG: Filtri <filter>

Un **filtro grafico** è un algoritmo matematico, tipico del campo dell'image processing, che riceve come input l'immagine (o un singolo elemento grafico) e produce come output l'immagine elaborata.
SVG mette a disposizione una serie di primitive grafiche che realizzano i comandi base di elaborazione dell'immagine come ad esempio effetti di luce, ombreggiature, composizioni di colori, fusioni di elementi, etc.
Un filtro grafico è realizzato utilizzando una combinazione di una o più di queste primitive.
L'elemento SVG per descrivere il filtro è **<filter>** ed al suo interno vanno definite tutte le primitive che compongono il filtro. Consideriamo un esempio di filtro grafico che produce l'ombra di una scritta.

SVG: Filtri <filter>

```
<svg width="300" height="300">
<defs>
   <filter id="ombra">
      <feGaussianBlur in="SourceAlpha" stdDeviation="3"
result="passo1"/>
      <feOffset in="passo1" dx="5" dy="5" result="passo2"/>
      <feMerge>
         <feMergeNode in="passo2"/>
         <feMergeNode in="SourceGraphic"/>
      </feMerge>
   </filter>
</defs>
<text x="50" y="100" style="font-family:times;font-size:60;fill:red"
filter="url(#ombra)">Ombra</text>
</svg>
```

SVG: JAVASCRIPT

SVG mette a disposizione l'interfaccia DOM (Document Object Model), per accedere e manipolare i nodi della struttura.
I principali metodi sono:

getElementById(nome_id): restituisce l'elemento grafico di cui abbiamo specificato l'identificatore;
setAttribute(nome_attributo,valore_attributo): consente di modificare il valore di un determinato attributo di un nodo;
getAttribute(nome_attributo): permette di leggere il valore di un attributo di un elemento;
createElement(nome_elemento): consente di creare un nuovo elemento grafico;
appendChild(nome_elemento): consente di inserire un nuovo elemento come figlio del nodo a cui questa funzione è applicata.
Per cercare di capire meglio come modificare un'immagine SVG attraverso l'uso di questi metodi consideriamo il seguente esempio:

SVG: JAVASCRIPT

```
<svg id="elementoRadice" width="300" height="300"
onload="aggiungiRect()">
   <script type="text/ecmascript"><![CDATA[
      function aggiungiRect(){
         var svgdoc=document.getElementById("elementoRadice");
         var newrect=document.createElement("rect");
         newrect.setAttribute("x",10);
         newrect.setAttribute("y",150);
         newrect.setAttribute("width",250);
         newrect.setAttribute("height",100);
         newrect.setAttribute("style","fill:blue;stroke:black;stroke-width:2;");
         svgdoc.appendChild(newrect);
      }
   ]]></script>
   <rect x="10" y="10" width="250" height="100"
style="stroke:black;fill:red;stroke-width:2"/>
</svg>
```

SVG: Eventi

Gli eventi principali gestiti da SVG sono:

onclick: evento scatenato dal click del mouse su di un elemento grafico;

onmousemove: evento associato al movimento del mouse;

onmouseover: evento scatenato dal passaggio del mouse su di un oggetto grafico;

onmouseout: evento scatenato quando il puntatore del mouse abbandona l'area di un elemento grafico;

onmousedown: evento associato alla pressione del tasto del mouse su di un elemento grafico;

onload: evento scatenato quando il documento SVG viene caricato dal visualizzatore.

SVG: Eventi

```
<svg width="500" height="300" viewBox="0 0 500 300">
   <script type="text/ecmascript"><![CDATA[
        function cambiaColore(evt){
            var elem=evt.target;
            var vecchioColore=elem.getAttribute("fill");
            if (vecchioColore=="blue") nuovoColore="yellow"
            else nuovoColore="blue";
            elem.setAttribute("fill",nuovoColore);
         }
         function cambiaRaggio(evt,dimensioneRaggio){
            var elem=evt.target;
            elem.setAttribute("r",dimensioneRaggio);
         }

   ]]></script>

   <rect id="rettangolo1" x="10" y="10" width="150" height="100" fill="blue"
stroke="black" stroke-width="2" onclick="cambiaColore(evt,'yellow')" />

   <circle id="cerchio" cx="350" cy="150" r="50" fill="red" stroke="black" stroke-
width="2" onmouseover="cambiaRaggio(evt,'100')"
onmouseout="cambiaRaggio(evt,'50')"/>

</svg>
```

SVG: SMIL

Descriviamo in modo schematico gli elementi e le estensioni principali di SMIL supportati da SVG.

animate: elemento utilizzato per animare un singolo attributo o una proprietà ;

set: elemento che assegna un determinato valore ad un attributo;

animateMotion: elemento che permette di animare un oggetto grafico lungo un percorso definito da un path;

animateColor: elemento che consente di modificare il colore di un elemento.

animatetransform: estensione che definisce un elemento per modificare una trasformazione SVG nel corso del tempo;

mpath: estensione che definisce un elemento con il quale descrivere il path utilizzato da animateMotion;

SVG: SMIL

Descriviamo in modo schematico gli elementi e le estensioni principali di SMIL supportati da SVG.

animate: elemento utilizzato per animare un singolo attributo o una proprietà ;

set: elemento che assegna un determinato valore ad un attributo;

animateMotion: elemento che permette di animare un oggetto grafico lungo un percorso definito da un path;

animateColor: elemento che consente di modificare il colore di un elemento.

animatetransform: estensione che definisce un elemento per modificare una trasformazione SVG nel corso del tempo;

mpath: estensione che definisce un elemento con il quale descrivere il path utilizzato da animateMotion;

SVG:

http://mauveweb.co.uk/bubbles/

view-source:http://www.bogotobogo.com/svg_source/svgclock.svg

Qual è il software giusto?

SOFTWARE RASTER:

- Gimp
- Adobe Photoshop
- MS-Paint
- …

SOFTWARE VETTORIALI:

- InkScape
- Adobe Illustrator
- Flash
- …

Domande? Chiarimenti? Proposte?