Tutorial Outline

- Part 1: Why another graphics format
- Part 2: Why in XML
- Part 3: SVG use of other W3C specs
- Part 4: SVG progress
- Part 5: Demonstrations
- Part 6: Features of SVG
- Questions and break
- Part 7: More features of SVG
- Part 8: SVG and other XML namespaces
- Part 9: Demonstrations and Questions

Scalable Vector Graphics (SVG) Tu

Chris Lilley, W3C
chris@w3.org
www.w3.org/people/chris

9th International World Wide Web Conference
2000

Why another graphics format?

Raster formats

- JPEG: lossy, good for photos; no transparency
- PNG: lossless, good for screenshots; transparency, accurate color
- GIF: widely used for animation

Vector formats

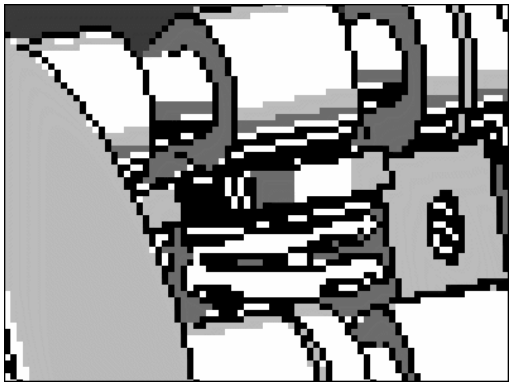
- WebCGM: great for technical graphics, good HTML integration

What is SVG

SVG is a vector graphics format, written in XML and stylable with CSS, and usable as an XML namespace in compound documents.

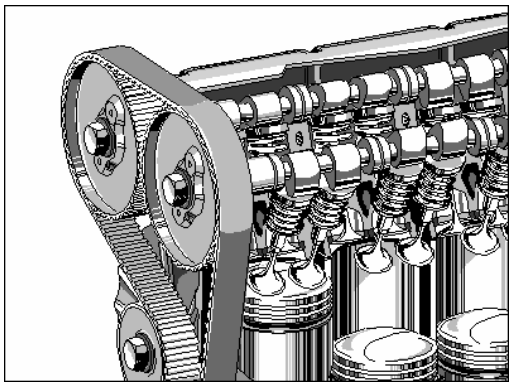
Rasters - WYS *really Is* WYG

- Fixed display density (1:1 image pixels:screen pixels)
- Zooming in does not increase detail

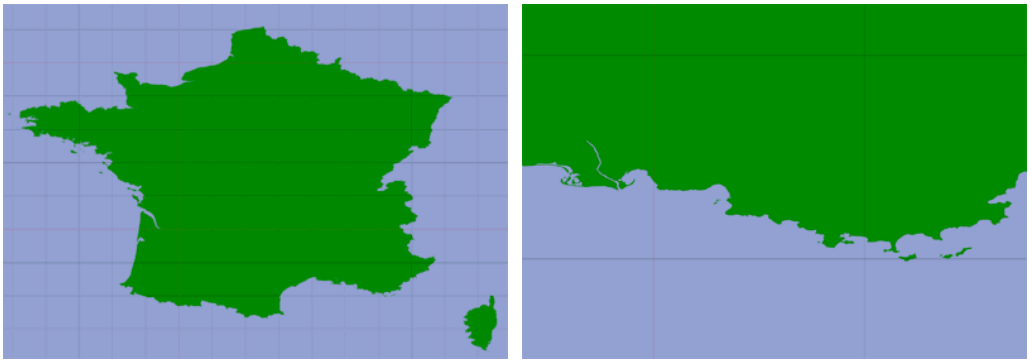


Raster graphics

- The normal Web graphics that we use today



Vectors - zoom to see more



Wide range of device resolutions

- Resolutions and sizes becoming more varied:
  - mobile Web
  - Internet Appliances
  - Web on TV
  - Hi-res LCD

Graphics Scalability

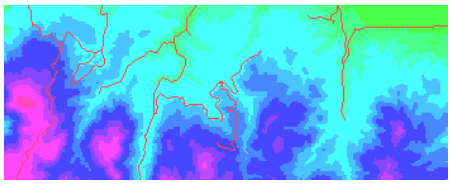
- SVG graphics can be
  - displayed on multiple device resolutions
  - displayed at different sizes on the same page
  - re-used at different sizes
  - cropped, and re-used at different zoom factors

Scalability

"To increase or decrease uniformly"

- Graphics scalability
  - not limited to fixed pixel dimensions
- Web scalability
  - not limited to small numbers of files, servers, users, areas of application

Typical uses



Web Scalability

- Inclusion of entire SVG graphic inside another
- Reference to parts of other SVG graphics - symbols, markers, gradients, fonts, patterns
- No centralized registry of symbols or fill patterns

XML - what is it

- Extensible Markup Language
- Not a markup language in itself
- Toolkit to define markup languages
- One general parser for all uses of XML
- Used for more than just documents
- "ASCII of the 90's" - Unicode
- Add your own tags and attributes ...
  - ... no machine-readable way to convey what they mean

Text in Graphics

- Around half of all Web graphics contain text
- Raster graphics can represent any script in Unicode - and new scripts
- Vector outlines can represent the same shapes - any script in Unicode
- Pictures of text not searchable, only apparent to sighted humans
- Storing Unicode characters is better, but requires fonts and line layout capability

...which looks like this

Typical XML Sample

```
<?xml version="1.0"?>
<article type="scientific">
  <headline>
    <site>INRIA</site> makes Web faster</headline>
  <intro><para>
    <location name="Sophia" country="fr"/>
  At the French
     Research Centre... </para>
  <para>But, as researchers quickly
point out,
     making things fast is hard.</para>
</intro></article>
```

...which looks like this

A graphical example

```
<?xml version="1.0"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 12 August 1999//EN"
"http://www.w3.org/Graphics/SVG/svg-19990812.dtd">
<svg width="4in" height="3in">
  <desc>A small group of simple shapes</desc>
  <style type="text/css"><![CDATA[
    g { fill: #6F9; stroke-width: 10px; font-size: 80 }
    g > rect {stroke: green; fill: #600 }
   ]]></style>
  <g transform="rotate(45)">
    <circle cx="370" r="170"/>
    <rect x="120" y="-80" width="500" height="120"/>
    <text x="150" y="10"> Hello World</text>
  </g>
</svg>
```

The Web with XML

- Free choice of document type
- Well formed documents mandatory
- DTD-less processing possible (and common)
- Presentation-less XML common
  - RDF, CDF, ICE, etc ...
- *no default semantics or presentation*

Uses of XML today

- Electronic commerce
- News content syndication
- Stock exchange summaries
- Distributed remote authoring
- Java bean serialisation
- Genome sequence analysis data
- Metadata for catalogs
- ... anything not requiring presentation

What (else) does XML buy us?

- Document Object Model (that works)
- Generic, powerfull link processing
- Generation on the fly
- Unicode text - searchable, stylable, accessible
- XML-specific search engines
- Namespaces allow vertical market integration

What does XML buy us?

- Well-formedness as a minimum requirement
- Unambiguous parsing, even with extensions
- Validation (DTDs now, later XSchema)
- Internal DTD subset - add entities, new attributes
- High level of implementation interoperability
- Ready availability of standard parsers

Related W3C Recommendations

- Document Object Model, level 1 - programatic manipulation
- Cascading Stylesheets, level 2 - stylesheets
- Resource Description Framework (RDF) - metadata assertion
  - Model and Syntax
  - RDF Schema definition
- XSL-T - XSL Transformation Language
- MathML 1.01 - XML namespace for mathematics
- SMIL 1.0 - XML namespace for timing and streaming integration

XML - just the first step

- For "Web pages", also need:
  - Hyperlinking (XLink, XPointer)
  - Presentation (CSS, XSL, xml-stylesheet PI)
  - Sharing and integration of tagsets (XML namespaces, Xschema)
  - Dynamic control (SMIL, DOM, CSS, CSS OM, BECSS)

SVG use of XML Namespaces

- All elements in SVG namespace
- Can insert SVG graphic as an image in other XML documents
- SVG graphic can contain other namespaces, eg 'metadata' element
- All SVG linking attributes in XLink namespace

XML Namespaces

- May want to use someone else's tagset
- May want to *combine* tagsets from multiple sources
- Possible name collisions
- XML namespaces associates a (unique) URL with each tagset
- Combination of URL+element name is unique, avoids collisions
- Similar to qualified import in programming languages

Linking and re-use

- Structuring as a graph, on top of the parse tree
- *Re-use* of paths, symbols, markers, gradient fills...
- Graphical elements spead over multiple files
- Symbol and marker libraries
- Inclusion of child SVG graphics

SVG and XLink/XPointer

- IDREF only does local references
- Use of XLink and XPointer allows uniform local and Web-wide reference
- Site management tools
  - need to identify links
  - don't know all possible namespaces
- XLink: A single link identification for XML

Cascading Style Sheets

- XML describes content and structure
- CSS describes presentation
- CSS makes pages:
  - faster to download
  - more accessible
  - more maintainable
  - more democratic: both users and authors can influence presentation

Styling Graphics

- Styling for site management
- Styling for reusability
- Styling for dynamism
- Styling for document transformation
- Need to style graphics as well as text

Example of CSS

```
foo [bar="toto"] {
  margin-left: 12%;
  margin-right: 7em;
  color: #000609;
  background: #fff;
  font-size: 40pt;
  font-family: tahoma, trebuchet, "gill sans",
               arial, helvetica, sans-serif ;
}
```


CSS basics

- Cascades the readers, authors and browsers style sheets together
- Formatting object tree nearly identical to document tree
- Selectors determine the elements to be styled
- Formatting properties are given values
- `slide > item { color: green }`

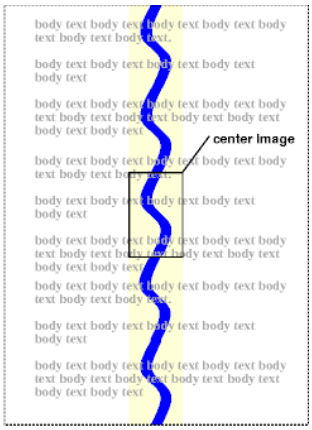
CSS2 Media-specific style sheets

```
  BODY {
    color: black;
    background: white;
  }

@media tv {
  BODY {
    color: white;
    background: black;
  }
}
```


Background images

- Background color/image, on any element
- Control of positioning, and tiling in x and y



XSL and transformation styling

- Needed for on-demand generated graphics
- Can help produce accessible graphics for other media (speech, Braille)
- XSL-T can transform source XML into SVG
  - XSL-T can switch namespace (to SVG)
  - Need to use only *external* stylesheets
  - Tools exist to compile XSL-T into servlet
- Also procedural (DOM) server-side transformation in Perl, Java, Javascript, etc

CSS supports

- progressive rendering
- dynamic modification
- downloadable fonts
- well defined colors (sRGB ICC profile, but full gamut)
- visual and aural rendering
- cascading reader/author balance

CSS is specifically designed for the Web

Implementation status

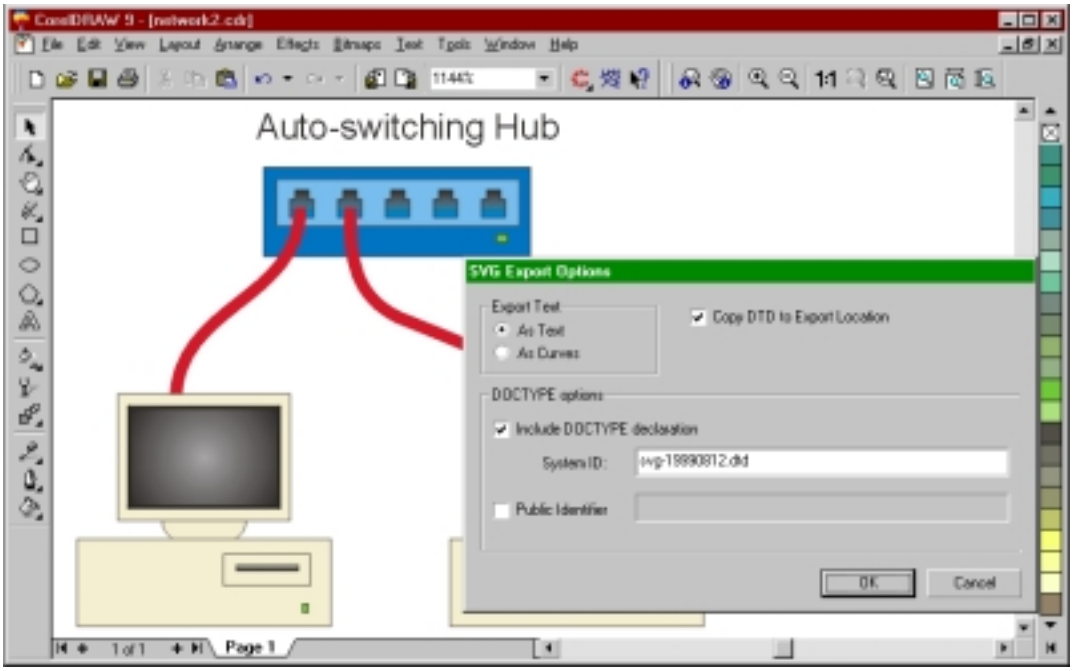
Publically announced implementations

- Viewers from Adobe (NS/IE plugins), Blackdirt, CSIRO, IBM, INRIA, Mozilla
- Exporters from Adobe Illustrator, CorelDraw!, Gill, Mayura Draw, Sketch, Sodipodi, JASC
- Converters from Blackdirt (from WMF), IBM (from AFP and from CGM), CSIRO (to JPEG)

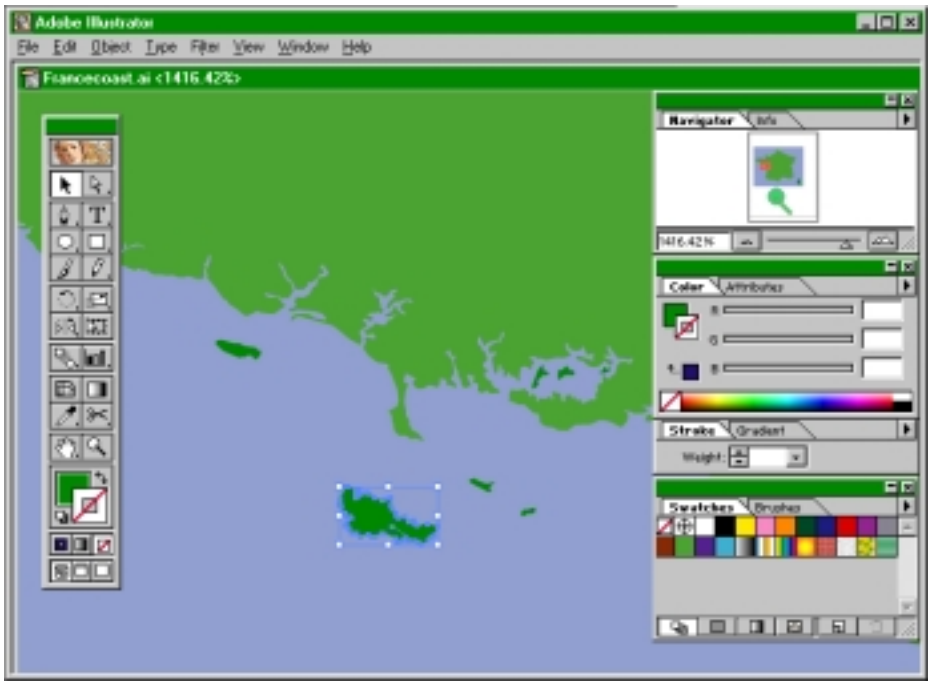
SVG Progress

- Requirements Document, Oct 1998
- First public working Draft, February 1999
- Eight public WDs (Feb 1999 to Mar 2000)
- Public release of SVG Test Suite
- Nearing completion at W3C
- Public page http://www.w3.org/Graphics/SVG

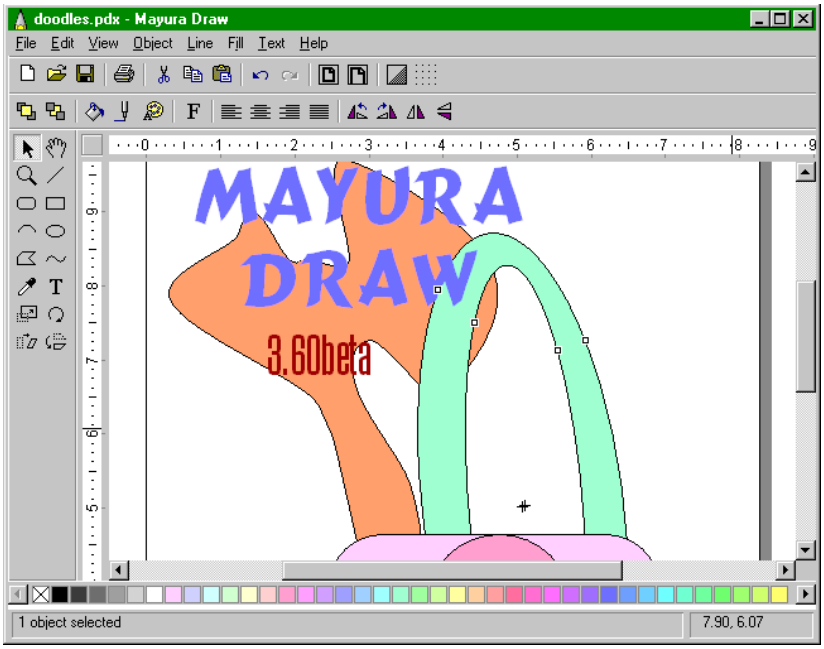
Exporting SVG in Corel Draw!



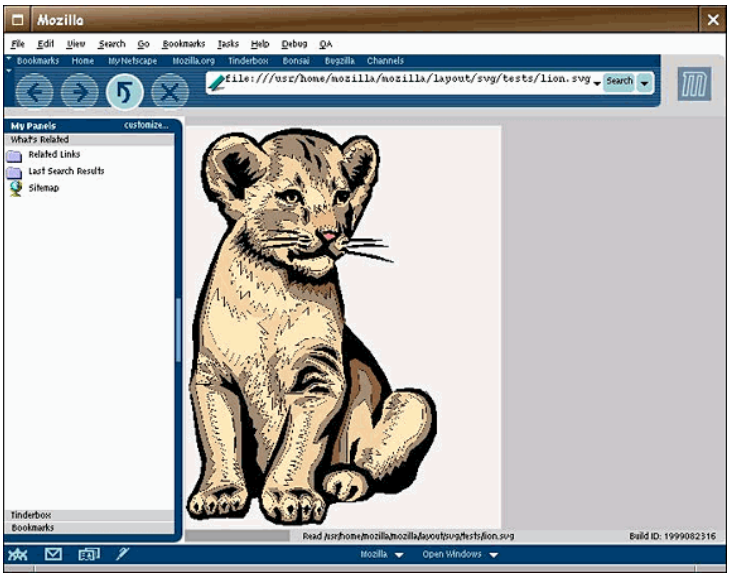
Editing SVG in Illustrator



Exporting SVG, Mayura Draw

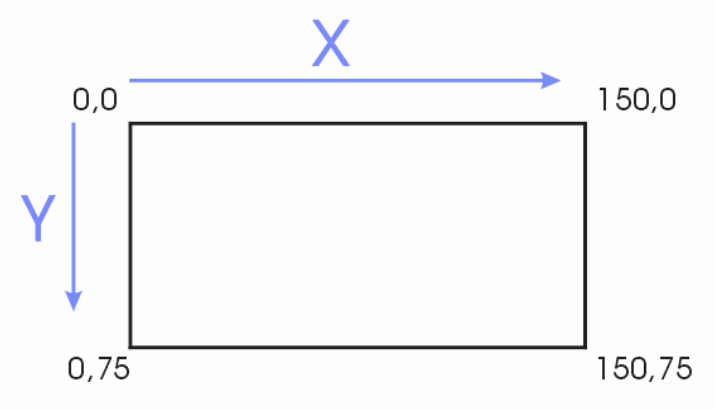


Displaying SVG, Mozilla



The SVG Canvas

- Points defined by *(x,y)* co-ordinate pair
- Default world coordinate system is Y-down (origin at top left), one unit is one pixel
- 'width' and 'height' attributes size the canvas



Features of SVG

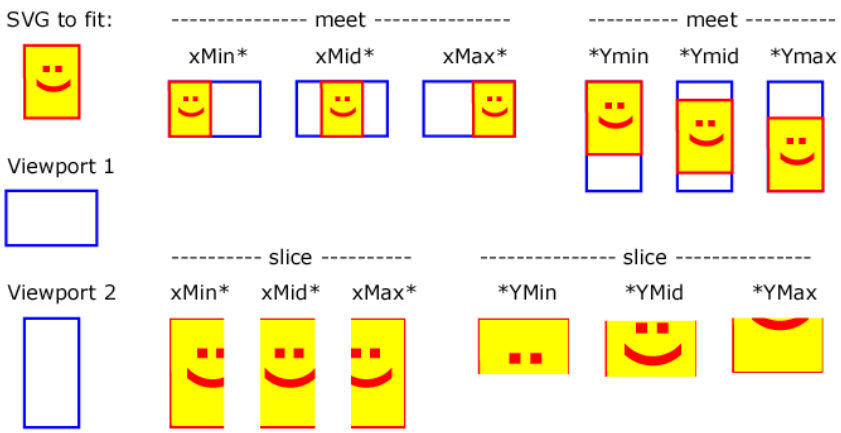
- The SVG canvas
- SVG objects
- Geometry and transformation
- Fills and strokes
- Templates/symbol libraries
- Inclusion of images
- Clipping

SVG objects

- Normal vector graphics stuff
  - Beziér curves, polylines etc
- *Graphical object* as fundamental primitive - not point or line
- Similar to most textual markup
  - Para as primitive, not verbs, nouns, syllables

The SVG Viewport

- A viewport is an area in world coordinates
- 'viewBox' attribute defines area to be viewed
- 'preserveAspectRatio' attribute fits viewport to canvas



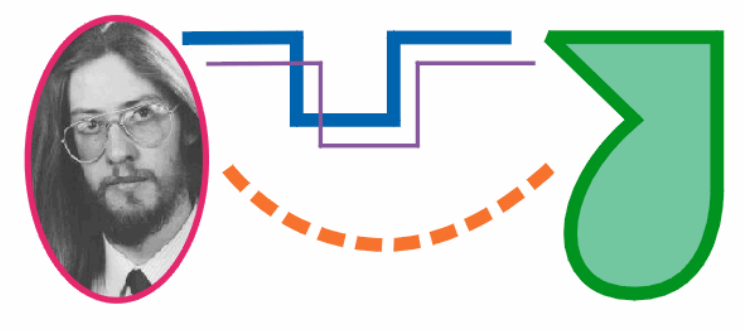
Paths in SVG

- 'path' element describes a single path
- 'g' element fror grouping paths

```
<?xml version="1.0"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG August 1999//EN"
"http://www.w3.org/Graphics/SVG/SVG-19990812.dtd">
<svg width="4in" height="3in">
    <path d="M 100,100 L 180,100 L 140,180 z"/>
</svg>
```

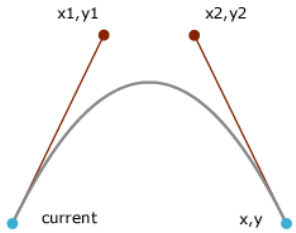

Paths

- Simple or compound paths, closed or open
- Can be filled, stroked, marked, used for clipping

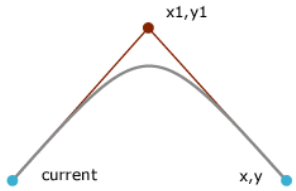


Curves (explained)

- Cubic Bezièr has two independent control points
- Type 1 fonts use cubic Bezièrs



- Quadratic Bezièr has one shared control point; typically more segments are needed
- TrueType fonts use quadratic Bezièrs



Path commands (lines)

- Uppercase is absolute, lowercase is relative
- Path segment draws from current point in path

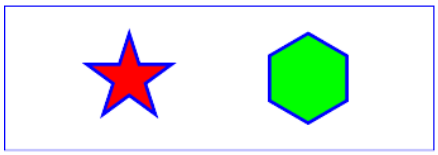
Command	Name	Arguments
M, m	moveto	x,y
L, l	lineto	(x,y)+
H, h	horizontal lineto	x+
V, v	vertical lineto	y+
Z, z	closepath	

SVG basic shapes

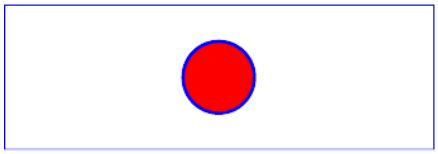
Particularly useful for hand coders

rect

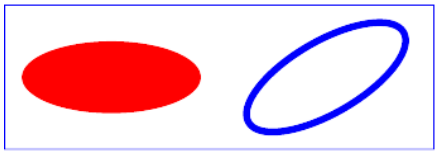
polygon



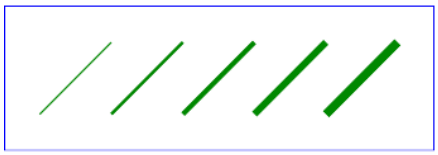
circle



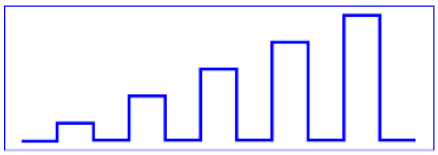
ellipse



line



polyline



Path commands (curves)

Command	Name	Arguments
C, c	cubic curveto	(x1,y1 x2,y2 x,y)+
S, s	short cubic curveto	(x2,y2 x,y)+
Q, q	quadratic curveto	(x1,y1 x,y)+
T, t	short quadratic curveto	(x,y)+
A, a	elliptical arc	(rx,ry rot arc-flag sweep-flag x,y)+

Transformations

- Transformations alter the coordinate system
  - 2x3 homogeneous transformation matrix for computers
  - Translate, rotate, scale, skew for humans
- With Matrices
  - coordinates expressed as *homogeneous coordinates* `x,y,1`
  - Matrices are 3x3
  - Constant terms are omitted

SVG basic shapes (continued)

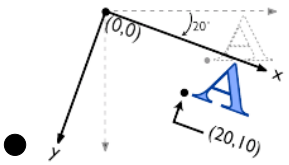
- One element represents one graphic object,as with paths
- Syntactic sugar for the paths of common geometric shapes

```
<?xml version="1.0"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG August 1999//EN"
"http://www.w3.org/Graphics/SVG/SVG-19990812.dtd">
<svg width="4in" height="3in">
    <rect x="20" y="30" width="50" height="90"/>
    <ellipse cx="90" cy="80" rx="45" ry="65/>
    <polygon points="20,20 50,100 200,80 70,300"/>
    <path d="M 20,20 L 50,100 200,80 70,300 z"/>
</svg>
```

Transformations - Rotation

Rotation about the origin, in degrees

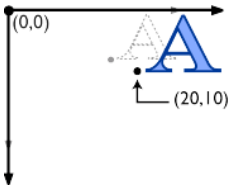
- As a command: `transform="rotate(45)"`
- As a matrix: $\begin{bmatrix} \cos(a) & -\sin(a) & 0 \\ \sin(a) & \cos(a) & 0 \\ 0 & 0 & 1 \end{bmatrix}$



Transformations - Scaling

Scaling from the origin, `x` and `y` can be specified separately

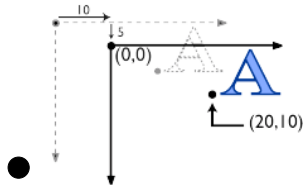
- As a command: `transform="scale(3 2)"`
- As a matrix: $\begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix}$



Transformations - Translate

Skewing about the origin, **x** and **y** skew are specified separately

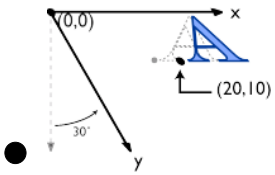
- As a command:
  transform="translate(50 80)"
- As a matrix: $\begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix}$



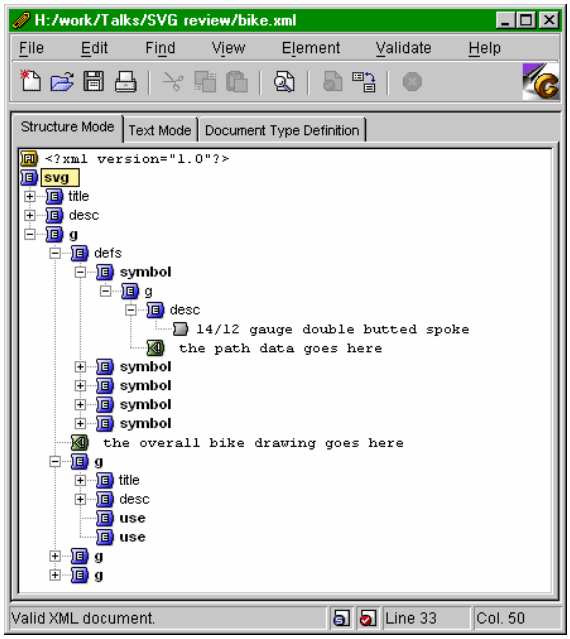
Transformations - Skew

Skewing about the origin, **x** and **y** skew are specified separately

- As a command:
  transform="skewX(30)"
- As a matrix: $\begin{bmatrix} 1 & \tan(a) & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$



Bike example (continued)



Bike example

- A drawing made of a hierarchy of parts
- Each part has a textual description
  - content not attribute
  - can be marked up in a different namespace
- Symbols are defined
- Symbols re-used, placed with transformations, restyled
- Symbols can be anywhere on the Web

Raster Image Example

- Same image composited on four different backgrounds, with transformations

SVG and Raster Images

- SVG can include JPEG and PNG images
- Size in pixels, user space or real-world units
- Resampling preserves scalability
- Images can be clipped, have transparent overlays

SVG and real Text

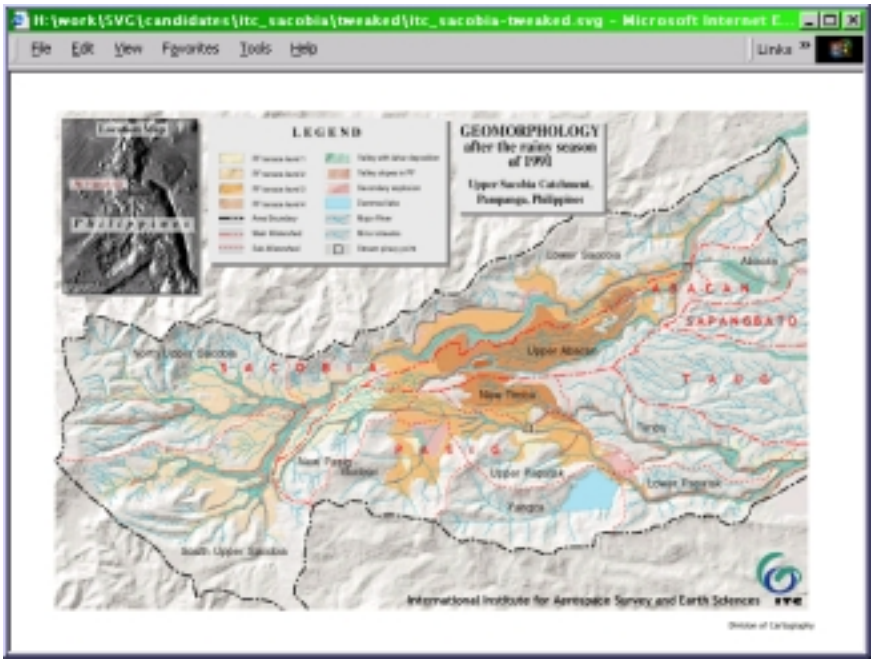
- SVG `title`, `desc` and `text` are:
  - Internationalized - any Unicode character
  - Searchable, selectable and indexable
  - Restylable
  - Accessible for the print-impaired
  - Easy to maintain
  - Dynamically modifiable

Text in Graphics

existing solutions inadequate

- Text inside GIFs - a big problem
- HTML `alt` & `longdesc`: lots of problems
- Sliced images, CSS text, tables - messy

Example of text in SVG



Text display in SVG

- Text can be displayed on a path
- Text can have gradient fills, be rotated, etc
- There is *no text flow* in SVG
- Horizontal (ltr, rtl) and vertical text supported

SVG Fonts

- Reliance on client installed fonts can compromise design
- Converting to outlines saves design, compromises everything else
- WebFont™ download works - but no universally supported format
- SVG can already describe cubic and quadratic Beziérs
- SVG fonts describe glyphs, with kerning
- Preserves exact glyph shape, preserves text as text

Fonts in SVG

- CSS allows list of font families

```
font-family: hattenschweiller, 'MS Gothic', tahoma, trebuchet,
        "gill sans", arial, helvetica, sans-serif ;
```

- WebFonts™ for intelligent matching, font download

Multilingual graphics

- `'switch'` allows testing and conditional display
- `'system-language'` tests against users preferred language
- So, graphics containing multiple language captioning are possible
- Great benefit for localisation and internationalisation

Fonts, SVG fonts, and curves

- Fonts must be on client, or downloaded
  - Platform, format, layout assumptions
- SVG fonts can be in same file
  - No platform or format assumptions, but still needs layout
- Curves can represent nearly anything
  - No platform, format or layout assumptions, no text either

Raster Effects

- Client-side rasterisation of vector image
- SVG adds intermediate, continuous tone rasterization phase
- Image processing nodes act on intermediate image
- Results of nodes can be composited and merged

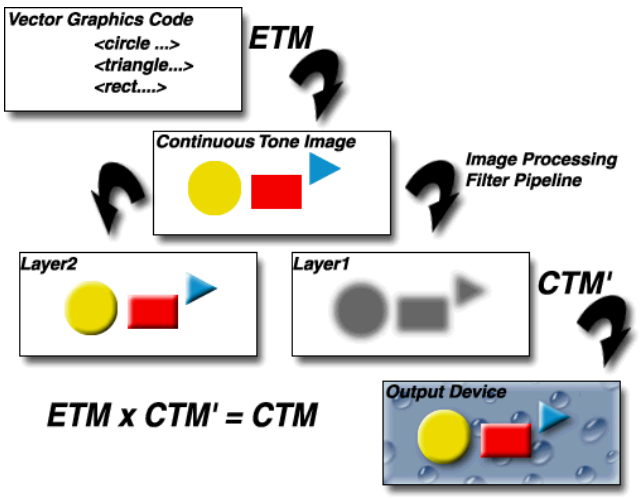
An example of 'switch'

```
<?xml version="1.0"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 12 August 1999//EN"
 "http://www.w3.org/Graphics/SVG/svg-19990812.dtd">
<svg width="300pt" height="140pt">
  <switch>
    <text x="20" y="80" system-language="fr">Bonjour</text>
    <text x="20" y="80" system-language="nl">Goedemorgen</text
    <text x="20" y="80">Hello
     <tspan xml:lang="fr">tout le monde!</tspan></text>
  </switch>
</svg>
```

Raster Effects example

```
<filter id="MyFilter" filterUnits="objectBoundingBox"
   x="-10%" y="-10%" width="110%" height="120%">
   <feGaussianBlur in="SourceAlpha" stdDeviation="4" nodeId="blur"/>
   <feOffset in="blur" dx="4" dy="4" nodeId="offsetBlurredAlpha"/>
   <feSpecularLighting in="blur" surfaceScale="5"
     specularConstant="1" specularExponent="10"
     lightColor="white" nodeId="specularOut">
     <fePointLight x="-5000" y="-10000" z="20000"/>
   </feSpecularLighting>
   <feComposite in="specularOut" in2="SourceAlpha"
                operator="in" nodeId="specularOut"/>
   <feComposite in="SourceGraphic" in2="specularOut"
     operator="arithmetic" k1="0" k2="1" k3="1" k4="0" nodeId="litPaint"/>
   <feMerge>
     <feMergeNode in="offsetBlurredAlpha"/>
     <feMergeNode in="litPaint"/>
   </feMerge>
 </filter>
```


Raster Effects (continued)



CSS and multi-namespace XML

- Need for stylable graphics
- Enhance document and graphic re-use
- Style compound documents
  - text and graphics at the same time
- Leverage existing knowledge
- Build on CSS-OM
- Enclosing document rendered by CSS; SVG rendered by SVG processor

Raster effects sample (rendered)

MathML

- Written in XML
- Both semantics and presentation of equations
- a-b

```
<apply>
  <minus/>
  <ci>a</ci>
  <ci>b</ci>
</apply>
```


SVG and other XML namespaces

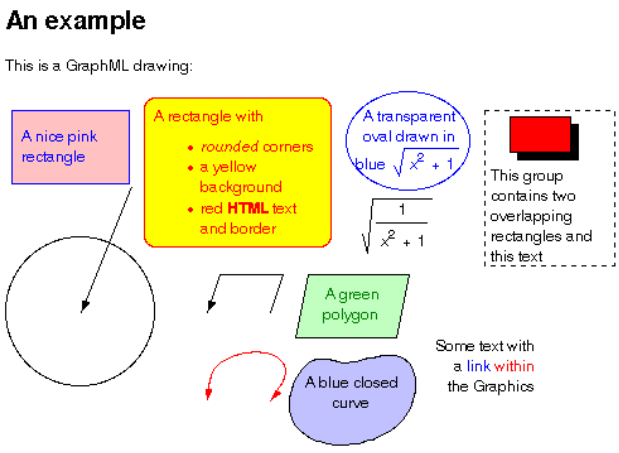
The real power of SVG can be seen when it is used in combination with other XML namespaces. We have already seen the use of the XML Linking (XLink) namespace. Now we will look at others which make good partners for SVG.

RDF - metadata in XML

- Written in XML
- Developed from PICS content rating language
- Describes metadata, in terms of graph theory
- Nodes (data) and arcs (metadata, relationships)
- "Author of this document is John Smith"
- Increasingly used for content syndication

MathML and SVG

- Experiments in Amaya, mixing:
  - XML vector graphics (SVG precusror)
  - A MathML implementation
  - An XML-ised HTML (XHTML precursor)



How does DOM fit in?

- XML DOM (DOM 1) for querying and modifying the content
- CSS OM (DOM 2) for querying and modifying the style sheets
- Event model (DOM 2)
- SVG DOM provides utility functions, built on top of XML and CSS DOM

RDF and SVG

```
<?xml version="1.0" standalone="yes"?>
<svg width="4in" height="3in"
    xmlns = 'http://www.w3.org/Graphics/SVG/SVG-19990812.dtd'>
    <desc>Show floor layout, XML World</desc>
    <metadata>
      <rdf:RDF
               xmlns:rdf = "http://www.w3.org/...-rdf-syntax-ns"
               xmlns:rdfs = "http://www.w3.org/TR/...-schema"
               xmlns:dc = "http://purl.org/dc/elements/1.0/"
               xmlns:svgmetadata = "http://www.w3.org/..." >
        <rdf:Description about=""
                 dc:title="XML 99 Layout"
                 dc:description="Show floor layout for XML 99 conference"
                 dc:publisher="Graphic Communications Association"
                 dc:date="1999-12-105"
                 dc:format="image/svg"
                 dc:language="en" >
          <dc:creator>
            <rdf:Bag>
                <rdf:li>Joe Bloggs</rdf:li>
                <rdf:li>Döina La Bruyère</rdf:li>
            </rdf:Bag>
          </dc:creator>
          <svgmetadata:General MeetsAccessibilityGuidelines="true"/>
        </rdf:Description>
      </rdf:RDF>
    </metadata>
    <g><!-- graphic goes here --></g>
</svg>
```

SMIL

- Written in XML
- Specifies synchronisation of streaming and non-streaming media
  - images, audio, video, text
- Specifies layout (CSS subset) and **timing**
- Widely used in net players
- New version (SMIL Boston) in development

Animation and SVG

- Dynamic CSS binding: `:hover`, `cursor`, `:active`
- DOM animation (Javascript, JAVA, etc) for dynamic manipulation, client-side generation
- Declarative animation of attributes and properties (in conjunction with SYMM)

SMIL and SVG

The W3C SYMM WG (which develops SMIL) and the SVG WG have worked together on a core model for *timing* and for *animation* for XML. SVG is the first language to use this. SMIL Animation just exited last call.

Declarative animation allows repeated *editing* in different authoring tools

SMIL in use

Summary: SVG

- A better way to do even raster images
- Maximises investment in XML, CSS, DOM,...
- Restylable and encourages graphics reuse
- Well internationalised
- Meets the challenge of Web device diversification
- Enhances Web accessibility
- Gives Web designers many new design capabilities
- Easily localised or translated
- .... SVG is rather cool, actually

SVG animation example

```
<rect id="foo" width="40" height="25">
<animate href="#foo"
       attributeType="xml" attribute="height"
       from="20" to="32" by="0.5" dur="12s" />
<animate href="#foo"
       attributeType="css" attribute="opacity"
       from="1" to="0" dur="5s"
       repeatCount="indefinite" />
```

For further information

www.w3.org/Graphics/SVG

Questions and Demonstrations