Document generated on 07/01/2025 12:04 a.m.

Documentation et bibliothèques

DOCUMENTATION BIBLIOTHÈQUES

SVG contre *Flash* : théorie et pratique
SVG versus *Flash*: Theory and Practice
SVG frente *Flash*: teoría y práctica

Myriam Beauchemin

Volume 52, Number 1, January–March 2006

URI: https://id.erudit.org/iderudit/1030025ar
DOI: https://doi.org/10.7202/1030025ar

See table of contents

Publisher(s)

Association pour l'avancement des sciences et des techniques de la documentation (ASTED)

ISSN

0315-2340 (print)
2291-8949 (digital)

Explore this journal

Cite this article

Beauchemin, M. (2006). SVG contre *Flash* : théorie et pratique. *Documentation et bibliothèques*, *52*(1), 29–36. https://doi.org/10.7202/1030025ar

Article abstract

Given that open software and standards are increasingly popular, we would like to compare SVG with *Flash*. This comparison is based on the distinction between free standards and proprietary standards; the history, the theory and the practical aspects of each of these tools is also presented. This comparison ends with a summary of the results of a trial to convert an object in SVG to *Flash*. if *Flash* offers certain functions that are unavailable in SVG, the latter is not to be considered as an inferior product and must be given serious consideration.

Tous droits réservés © Association pour l'avancement des sciences et des techniques de la documentation (ASTED), 2006

This document is protected by copyright law. Use of the services of Érudit (including reproduction) is subject to its terms and conditions, which can be viewed online.

https://apropos.erudit.org/en/users/policy-on-use/

érudit

This article is disseminated and preserved by Érudit.

Érudit is a non-profit inter-university consortium of the Université de Montréal, Université Laval, and the Université du Québec à Montréal. Its mission is to promote and disseminate research.

https://www.erudit.org/en/

DOCUMENTATION BIBLIOTHÈQUES

SVG contre *Flash* : théorie et pratique

MYRIAM BEAUCHEMIN
Bibliothécaire, Affaires indiennes et du Nord Canada
myriam@myriam.ca

Résumé

RÉSUMÉ | ABSTRACTS | RESUMEN

Alors que logiciels et normes libres sont en plein essor, nous proposons une comparaison sommaire entre SVG et Flash. *Celle-ci s'appuie principalement sur la distinction entre normes libres et normes propriétaires, mais y sont aussi exposés histoire, éléments de théorie et aspects pratiques de chacun de ces outils. Cette comparaison est complétée par le compte rendu de la réalisation d'un exercice de reproduction en SVG d'un objet initialement conçu en* Flash. *Si* Flash *propose certaines fonctionnalités non offertes par SVG, ce dernier n'est pas pour autant démuni face à son concurrent et mérite qu'on s'y arrête.*

SVG versus *Flash*: Theory and Practice

Given that open software and standards are increasingly popular, we would like to compare SVG with Flash. *This comparison is based on the distinction between free standards and proprietary standards; the history, the theory and the practical aspects of each of these tools is also presented. This comparison ends with a summary of the results of a trial to convert an object in SVG to* Flash. *If* Flash *offers certain functions that are unavailable in SVG, the latter is not to be considered as an inferior product and must be given serious consideration.*

SVG frente *Flash*: teoría y práctica

En la época en que los programas y normas libres están en pleno auge, se propone una somera comparación entre SVG y Flash. *Esta trata sobre todo de la distinción entre normas libres y normas de propiedad privada pero también se expone la historia, los elementos teóricos y los aspectos prácticos de cada una de estas herramientas. Se concluye esta comparación con la reseña de la realización de un ejercicio de reproducción en SVG de un objeto inicialmente concebido en* Flash. *Si bien* Flash *propone ciertas funciones que SVC no ofrece, éste último no está, sin embargo, en desventaja frente a su competidor y merece que se le preste atención.*

Introduction

Logiciels et normes libres sont en plein essor et suscitent un intérêt toujours grandissant. Dans ce contexte, nous proposons une comparaison sommaire entre SVG (*Scalable Vector Graphics*) et *Flash*. Cette comparaison prend racine sur le fait que les philosophies sous-jacentes de ces deux outils de développement pour supports interactifs ne peuvent être plus éloignées l'une de l'autre : norme ouverte pour l'un, norme propriétaire pour l'autre, SVG étant une recommandation du World Wide Web Consortium (W3C) et *Flash*, la propriété de Macromedia. Aussi, bien que le langage SVG soit souvent défini comme un équivalent gratuit de *Flash*, il est légitime de croire que son développement se fasse à un rythme plus lent et donc qu'il ait moins à offrir que son compétiteur. Une analyse plus poussée des deux technologies pourrait cependant nous mener à une conclusion différente.

Nous distinguerons d'abord les normes ouvertes des normes propriétaires, tout en exposant les risques des unes et les avantages des autres, le tout dans un contexte informationnel. Nous présenterons ensuite séparément SVG et *Flash*, et nous en ferons ressortir leurs principales différences. Ce volet théorique sera suivi du compte rendu d'un exercice de reproduction en SVG d'un objet conçu en *Flash*. Enfin, nous suggérerons des applications possibles de SVG dans le vaste domaine des sciences de l'information.

L'énorme différence

Une norme informatique, qu'elle soit ouverte ou propriétaire, est un ensemble de spécifications décrivant les caractéristiques d'un matériel, d'un logiciel, d'un format de fichier, d'un langage ou encore d'un protocole de communication. *« Le respect d'une norme* [...] *est un acte essentiel afin de pouvoir communiquer, agir, échanger entre des individus, des machines. C'est ce qu'on appelle l'interopérabilité. »* (Dubost, cité par Dumais, 2003).

Une norme ouverte n'est la propriété d'aucune instance commerciale ou privée. Elle est mise à la disposition de tous avec une licence garantissant que ses droits d'utilisation sont libres. *« Dans le domaine informatique, les normes ouvertes sont produites de façon collégiale par un organisme regroupant tous*

les acteurs du domaine [...] *et qui prennent part au consensus de l'élaboration* [...]. » (Dumais, 2003). En contrepartie, une norme propriétaire appartient exclusivement à une entité commerciale ou privée, laquelle peut en permettre ou non l'utilisation, mais sans offrir de licence qui en garantit la libre exploitation.

Risques des normes propriétaires

Une norme propriétaire n'est donc pas sans présenter quelques risques pour ses utilisateurs : l'entreprise détentrice pourrait disparaître, cesser de supporter son produit, décréter une interdiction d'utilisation, imposer le paiement d'une nouvelle licence, l'achat d'une nouvelle version du produit, voire l'acquisition d'un produit supplémentaire. L'information et les données étant captives d'une norme (ou d'un produit), il devient hasardeux de la délaisser au profit d'une autre, ce qui diminue d'autant la marge de manœuvre et risque d'engendrer des déboursés additionnels.

Avantages des normes ouvertes

L'utilisation d'une norme ouverte n'empêche pas, pour des raisons économiques, stratégiques ou autres, d'opter ensuite pour un autre produit ni même de développer soi-même son propre produit, *« mais au moins, les informations seront stockées dans un format leur permettant d'évoluer librement de toutes contraintes »* (Dubost, cité par Dumais, 2003), sans risque de les voir devenir inaccessibles.

Julien Tayon (2002), dans son plaidoyer pour les standards ouverts, énumère une série d'arguments économiques et éthiques qui militent en faveur de ces derniers. Nous présentons ici ceux qui sauront assurément être entendus par les différents acteurs du domaine des sciences de l'information :

> « **Diminuer le prix des licences.** *Certains acteurs tirent de leurs normes fermées une rente de situation qui leur permet d'imposer un prix excessif pour leurs produits parce que les utilisateurs sont dans un marché captif. En ouvrant le marché à la concurrence, les prix des outils* [...] *devraient baisser.* »
>
> « **Pérenniser les investissements informatiques.** *Si un éditeur (de logiciels libres ou non) vous fournit une solution basée sur des standards ouverts, il pourra toujours disparaître, vos données* [...] *seront toujours récupérables. Cela favorise l'indépendance face au fournisseur* [...]. »
>
> « **Pérenniser les normes.** *Nous constatons que des normes ouvertes* [...] *ont des durées de vie supérieures aux normes fermées, qui ne durent parfois que le temps d'un produit* [...]. »
>
> « **Diminuer le rapport signal/bruit.** *L'incompatibilité des standards, notamment au niveau des données, entraîne automatiquement une perte d'information à chaque transformation de format.* [...] *De plus, l'incompatibilité et la transformation de formats est coûteuse en temps de travail* [...] *sans réelle valeur ajoutée.* »
>
> « **Faciliter les communications entre individus.** *Au même titre que le système métrique international a facilité le commerce, cette démarche généreuse profitera à* [l'humanité]. *Les* [pays] *en voie de développement pourront directement profiter et participer à notre travail, sans consentir de gros investissements en dehors des infrastructures d'égal à égal.* »
>
> « **Développer les connaissances.** *Les normes ouvertes, c'est aussi la possibilité pour les individus de se former tout au long de la vie sans être dépendant d'une entreprise ou d'un éditeur.* »

Dans les entrailles de SVG

SVG dans le temps

Les images matricielles classiquement utilisées dans les pages Web (GIF, JPEG, PNG), bien qu'elles soient tout à fait appropriées pour les images photographiques, présentent de nombreux inconvénients (Duce, 2001) :

- ▷ fichiers de taille considérable ;
- ▷ résolution prédéfinie ;
- ▷ format binaire rendant difficile l'ajout de métadonnées ;
- ▷ hyperliens multiples devant être réalisés à l'aide d'images cliquables ;
- ▷ animation nulle ou minimale ;
- ▷ génération dynamique impossible ;
- ▷ texte non indexable.

Afin de contrer ces limitations, SVG a été élaboré par le W3C, un organisme *« créé pour stopper l'émergence de langages propriétaires sur le Web et rassembler tous les acteurs autour d'une table afin de définir des standards »* (Dumais, 2003). Le processus de développement a été enclenché en mars 1998, à la suite de la soumission conjointe du Rutherford Appleton Laboratory et de l'Institut national de recherche en informatique et automatique (INRIA) d'une proposition d'utilisation de XML pour la représentation de diagrammes schématiques en deux dimensions (*Web Schematics*). Cette première proposition fut rapidement suivie par deux autres : *Precision Graphics Markup Language* (PGML) par Adobe Systems et Sun Microsystems, et *Vector Graphics Markup Language* (VML) par Macromedia et Microsoft. Une quatrième

proposition, DrawML, fut reçue plus tard cette même année (Duce, 2001; Wikipedia, 2005b).

Le W3C mit alors rapidement en place un groupe de travail pour le développement de SVG, comprenant des représentants de nombreuses compagnies: Adobe, Agfa, Apple, Canon, Corel, Ericsson, HP, IBM, Kodak, Macromedia, Microsoft, Nokia, Sharp et Sun Microsystems (W3C, 2004a). Une première collecte de spécifications, lesquelles furent consignées le 29 octobre 1998 sous la forme d'un document de travail (W3C, 1998), permit à SVG de prendre peu à peu forme (Duce, 2001).

La version 1.0 de SVG est devenue une recommandation du W3C le 4 septembre 2001, la version 1.1 (la version actuelle) a obtenu ce statut le 14 janvier 2003, tandis que les versions 1.2 et 2.0 sont toujours en cours d'élaboration (W3C, 2001; W3C, 2003; W3C, 2002).

SVG en théorie

SVG, l'acronyme de *Scalable Vector Graphics*, est utilisé pour la définition de graphiques vectoriels en deux dimensions pouvant être visionnés sur le Web. Une application SVG peut comporter trois types d'objets: des formes géométriques de base, des images matricielles incorporées et du texte. Les formes de base sont la ligne, le cercle, l'ellipse, le rectangle, le polygone, la polyligne, en plus du chemin (défini à l'aide d'une succession de lignes, d'arcs elliptiques, de splines et de courbes de Bézier), qui peut être ouvert ou fermé. Ces quelques primitives graphiques permettent la création d'images on ne peut plus complexes, mais ne sont pas adaptées à la représentation de photographies pour lesquelles les formats matriciels demeurent plus appropriés, bien que celles-ci puissent être incluses dans une image SVG (dans une application d'album photos, par exemple).

Les différents éléments peuvent être groupés, stylés, transformés ou combinés afin de former de nouveaux objets. En plus du cadre de découpage, SVG offre une variété de styles de remplissage, d'effets de filtre et supporte les couches alpha pour la transparence. Tout élément et toute propriété peuvent être animés. De plus, les transformations peuvent être imbriquées, c'est-à-dire qu'une transformation géométrique peut être appliquée à une autre transformation. En outre, du fait de leur nature vectorielle, les images SVG sont extensibles et peuvent être réduites ou agrandies sans perte de qualité graphique.

Les applications SVG peuvent être rendues dynamiques et interactives par la gestion des événements (clic de souris, déplacement du curseur, etc.) et le scriptage, le plus souvent à l'aide des langages JavaScript ou ECMAScript. Dans un même ordre d'idées, tout élément et tout groupe d'éléments peuvent être définis comme liens hypertextuels ou simplement réagir à certains événements. Enfin, sans entrer dans les détails techniques, précisons que SVG est conforme à plusieurs autres spécifications du W3C, incluant DOM, HTML, CSS, XML, XSL, XHTML et SMIL, ce qui permet d'accéder par scriptage à des éléments de différents espaces de nommage dans une même page Web (W3C, 2004a), contribuant de ce fait au dynamisme des applications.

SVG en pratique

Création et édition

La création et l'édition d'un fichier SVG ne nécessitent qu'un banal éditeur de texte, par exemple Bloc-note de Windows ou vi. Toutefois, il peut s'avérer plus commode d'éditer des images dans un environnement graphique, entre autres pour la définition de courbes de Bézier ou la création de formes complexes. Deux types d'éditeurs graphiques peuvent être utilisés à cette fin: les éditeurs natifs et ceux pouvant exporter du SVG. *«A native SVG editor is an editor that has the main purpose to produce SVG. It is optimized for SVG functionality and should implement most SVG elements.»* (Hauser et Wenz, 2004). En ce sens, puisqu'ils ne sont pas conçus à cette fin, les éditeurs exportateurs devraient être réservés à la génération du code SVG représentant une forme complexe, code à incorporer ensuite dans le fichier SVG en cours d'édition (manuelle), non pas à la création d'images entières.

Le W3C met à la disposition des internautes une liste officielle d'implémentations SVG, incluant visionneurs et éditeurs (W3C, 2004d). Par contre, nos recherches nous ont fait réaliser que cette liste mériterait d'être mise à jour plus régulièrement, puisque des éditeurs mentionnés sur les sites SVG.org et SVGfr.org n'y figurent pas, soit Sketsa de Kiyut et Inkspace, un logiciel libre dérivé de Sodipodi. Parmi les éditeurs natifs, mentionnons aussi Amaya du W3C, EvolGraphix XStudio, ainsi que Jasc WebDraw, bien que ce dernier ne soit plus offert. La liste des éditeurs exportateurs de SVG est beaucoup plus garnie et comprend entre autres des logiciels de dessin parmi les plus populaires: Adobe Illustrator et CorelDraw.

Inclusion des polices de caractères

Les polices de caractères non standards explicitement utilisées dans des documents SVG (de même que HTML, PDF, etc.) peuvent soulever un problème si le poste client n'a pas ces mêmes polices. De fait, les polices manquantes seront remplacées à l'affichage par des polices présentes sur le poste client, le plus souvent Arial ou Times New Roman. Le problème est

mineur dans le cas d'un document strictement textuel, mais peut complètement gâcher le design d'une application SVG si le positionnement des caractères ou des mots avait été calculé au pixel près par le concepteur.

Afin de contrer ce désagrément, il a été prévu des balises permettant d'inclure à même un fichier SVG la définition d'une police de caractères, toujours sous format XML. Bien qu'une telle définition soit sous forme textuelle, il est pratiquement impossible de la rédiger sans l'aide d'un convertisseur de polices, à cause des chemins complexes utilisés pour dessiner chacune des lettres. Voyons, par exemple, le code nécessaire au dessin d'un simple O dans la police Forte :

```
<glyph unicode="O" glyph-name="O" horiz-
adv-x="1300" d="M662 1087Q593 1018 562 974
T494 846Q467 786 455 719Q440 660 440 567Q4
40 436 465 349T539 217T657 172Q713 172 771 213
T877 320T954 469T983 627Q983 764 889 913Q8
54 973 854 995Q854 1036 899 1077Q950 1126 1016
1164T1120 1202Q1142 1202 1166 1192T1204 1165
Q1236 1132 1257 1044T1278 856Q1278 620 1182 4
21T920 105T561 -12Q450 -12 341 29T164 139Q49
254 49 440Q49 548 91 669T206 904T376 1114T5
75 1264Q668 1314 775 1344T963 1374Q1077 1374
1077 1339Q1077 1325 1053 1296Q1009 1255 977 12
38T870 1200Q790 1176 754 1157T662 1087Z" />
```

De tels convertisseurs sont disponibles gratuitement, dont l'application ttf2svg, une composante du Batik SVG Toolkit développé dans le cadre du projet XML de Apache (Apache XML Project). Celle-ci prend en entrée le fichier d'une police de caractères de type TrueType et en génère la définition SVG équivalente qui peut alors être insérée dans le fichier SVG nécessitant cette police. Cette application exige par contre quelques connaissances informatiques supplémentaires, puisqu'elle nécessite une installation préalable de l'environnement d'exécution Java sur le poste de travail, et qu'elle s'exécute depuis la ligne de commande avec une série d'options.

Compression des fichiers

Le langage SVG est du pur XML, aussi les applications SVG se présentent-elles sous la forme de fichiers texte (avec l'extension.svg). Ceux-ci peuvent cependant être compressés à l'aide de l'utilitaire gzip, lui aussi exécutable depuis la ligne de commande avec une série d'options. Puisque les fichiers SVG contiennent beaucoup de données redondantes, cette compression permet d'en réduire considérablement la taille (Wikipedia 2005b). Ils deviennent de ce fait des fichiers binaires (avec l'extension.svgz). Ils seront décompressés par le logiciel client avant l'affichage, étape qui est tout à fait transparente pour l'usager.

Visionnement des images et des applications

Dans un passé encore récent, si l'on fait abstraction d'Amaya, l'éditeur-navigateur du W3C, aucun navigateur Web n'offrait de support natif pour le visionnement des documents SVG, mais des développements rapides ont lieu de ce côté. Opera 8, lancé le 19 avril 2005, serait le premier navigateur a en avoir offert un support complet (Opera Software). Mozilla Firefox, quant à lui, espère l'offrir dès la version 1.1 (Mozilla). Autrement, l'installation d'un plugiciel de visionnement est nécessaire, le plus répandu étant Adobe SVG Viewer, téléchargeable gratuitement.

Flash à vol d'oiseau

Flash dans le temps

La genèse de ce qui allait devenir *Flash* débute en 1993, quand la compagnie FutureWave est créée. Celle-ci développe alors SmartSketch, un logiciel de dessin initialement conçu pour les ordinateurs à stylet, qui a ensuite évolué pour Windows et Macintosh. À l'été 1995, après plusieurs suggestions en ce sens, il a été décidé d'en faire un logiciel d'animation. À la même époque, le Web gagnait peu à peu en popularité; FutureWave y voyait déjà un marché intéressant pour un logiciel d'animation en deux dimensions.

On voulut d'abord nommer le futur logiciel d'animation SmartSketch Animation, mais le peu de reconnaissance commerciale du produit original fit plutôt opter ses concepteurs pour un nom inédit, soit CelAnimator. Craignant de voir le nouveau produit étiqueté comme un logiciel d'animation de dessins animés, on décida de plutôt le nommer FutureSplash Animator.

En octobre 1995, Adobe Systems eut de sérieuses visées sur FutureWave, mais recula. En décembre de la même année, Fractal Design eut des visées similaires, mais recula aussi. Les deux compagnies, qui étaient surtout intéressées par le logiciel de dessin SmartSketch, n'avaient vraisemblablement pas été impressionnées par la lenteur de la démonstration de FutureSplash Animator! Celui-ci fut officiellement lancé l'été suivant. En novembre 1996, FutureWave fut de nouveau approchée, par Macromedia cette fois. La transaction de vente était complétée le mois suivant; Macromedia *Flash* 1.0 était né. (*Flashmagazine*, 2002). La bête a, depuis, beaucoup évolué et en est rendue à la version 7.2. L'histoire se répétant souvent, Adobe est revenue à la charge afin d'acquérir Macromedia, sans reculer cette fois.

Flash en théorie

Flash pourrait à la base être défini et présenté de la même manière que SVG : graphiques vectoriels, Web, dynamisme, interactivité, scriptage (via ActionScript,

un dérivé d'ECMAScript), gestion des événements, inclusion d'objets. Nous présenterons plus loin ce qui le distingue foncièrement de SVG.

Flash en pratique

Création et édition

Les applications *Flash* sont principalement créées et éditées à l'aide des logiciels *Flash* MX 2004 et *Flash* MX Professional 2004 de Macromedia. Il existe cependant quelques autres logiciels, conséquence du fait que, en octobre 1998,

> « [...] *Macromedia disclosed the Flash Version 3 Specification to the world on its Website. It did this in response to many new and often semi-open formats competing with SWF, such as XARA's Flare and Sharp's Extended Vector Animation formats. Several developers quickly created a C libray for producing SWF.* [...] *Macromedia also hired Middlesoft to create a freely-available developers' kit for the SWF file format versions 3 to 5.* [...] *Macromedia has made the Flash Files specifications for versions 6 and later available only as a PDF under a non-disclosure agreement.* » (Wikipedia, 2005a).

Parmi les autres éditeurs offerts pour l'édition *Flash*, mentionnons CoffeCup Firestarter, KoolMoves, Glanda et SWF Quicker de Sothink, différents produits de SWiSH, ainsi que Linx de Wildform. Ajoutons que quelques logiciels peuvent aussi exporter en *Flash*, dont ToonBoom Studio. Notons que l'édition d'un projet d'application *Flash* génère différents fichiers (.fla,.as,.swd,.flv, etc.), et qu'une fois complétée et publiée (extension.swf), l'application est non éditable (Wikipedia, 2005a).

L'inclusion des polices de caractères s'effectue depuis l'éditeur, sans l'aide d'un outil externe. Notons enfin que les applications *Flash* peuvent être compressées à l'aide de zlib, une librairie développée par les concepteurs de gzip, lui-même utilisé pour la compression de fichiers SVG.

Visionnement des images et des applications

Aucun navigateur Web n'offre de support natif pour les applications *Flash*. Aussi, l'installation d'un plugiciel de visionnement est-elle obligatoire. Celui-ci est gratuitement offert par Macromedia et « [...] *est souvent livré avec les dernières versions des navigateurs actuels* » (Wikipédia, 2005a).

Principales différences entre SVG et *Flash*

Une comparaison détaillée des spécifications des versions 1.1 de SVG et 7 de *Flash* est en cours d'élaboration par George Held et ses collègues (2004). Une analyse sommaire de ce document, combinée aux données présentées plus haut, nous permet de faire ressortir les principales différences entre ces deux outils.

Il appert tout d'abord que SVG et *Flash* sont issus d'univers bénéficiant de ressources quasi antinomiques, collaboratives et bénévoles pour le premier, économiques pour le second. C'est ce qui explique probablement le fait que SVG soit une norme libre et *Flash*, basé sur une norme propriétaire et disponible uniquement sous entente de confidentialité. On remarque ensuite que les fichiers supportant les applications conçues en SVG et celles en *Flash* sont de nature bien différente, soit textuelle contre binaire. En outre, les fichiers SVG, tout comme les fichiers HTML, peuvent être lus et modifiés à l'aide d'un simple éditeur de texte, alors que les fichiers SWF ne peuvent tout simplement pas être édités; seuls les fichiers ayant servi à la création de l'application SWF peuvent l'être et ce, avec le support d'un éditeur relativement complexe.

Les fichiers strictement textuels présentent l'avantage de pouvoir être indexés par les robots Web. Macromedia offre une interface de programmation permettant d'extraire les chaînes textuelles et les hyperliens des fichiers SWF pour fin d'indexation (Wikipédia, 2005a; Held *et al.*, 2004), mais cette opération est technologiquement plus coûteuse que ne peut l'être une indexation directe. Le code source des fichiers SVG peut être visionné depuis le navigateur, même s'il était initialement compressé, ce qui facilite d'autant le partage d'informations et de connaissances en matière de développement d'applications. De plus, puisqu'ils utilisent une définition de type de document (DTD) du W3C, les documents SVG peuvent être soumis à l'interface de validation du W3C. Il s'agit là d'une caractéristique non négligeable, puisqu'un document en apparence fonctionnel pourrait camoufler une structure inappropriée.

Flash offre un meilleur support des blocs de texte (ou paragraphes) que SVG. En fait, les blocs de texte sont inexistants en SVG; ceux-ci doivent être représentés une ligne à la fois, la détermination des sauts de ligne et de l'espacement entre les lignes incombant alors au concepteur. Un support des blocs de texte est cependant planifié pour une prochaine version de SVG (W3C, 2002).

Contrairement à SVG, *Flash* offre très peu de primitives géométriques, si ce n'est la ligne et la spline. Les autres formes (polyligne, rectangle, cercle, ellipse, polygone) doivent être réalisées à l'aide d'une succession de lignes et de courbes. *Flash* supporte

cependant des objets inconnus de SVG, c'est-à-dire le son et la vidéo.

Alors que SVG met une grande variété d'unités de mesure à la disposition des concepteurs (pixel, point, centimètre, millimètre, pouce, pourcentage), *Flash* se contente d'offrir le pixel. Même chose pour ce qui est des mesures angulaires : degré, radian et grade pour SVG, degré seul pour *Flash*.

Soulignons, enfin, le grand écart entre les tailles respectives des plugiciels : Adobe SVG Viewer fait 2,27 mo, tandis que Macromedia *Flash* Player en fait seulement 480 ko. Il est vrai que cette différence n'est valide que très ponctuellement, mais le temps nécessaire au téléchargement du plugiciel de visionnement d'applications SVG pourrait rebuter un internaute ne bénéficiant pas d'une connexion Internet haute vitesse.

Reproduction d'un objet *Flash* en SVG

Le but de cet exercice était de vérifier par la pratique si SVG se comparait vraiment à *Flash*. Les résultats obtenus n'ont aucune valeur scientifique, mais offrent malgré tout un éclairage intéressant. Les applications SWF et SVG, de même que des saisies d'écran, peuvent être consultées sur le site Web personnel de l'auteure.

Recherche et Sélection de l'objet *Flash* à reproduire

Nous désirions reproduire une application *Flash* offrant une diversité de formes géométriques, plusieurs éléments de texte, une variété d'animations et un minimum de dynamisme et d'interactivité. Nous tenions par contre à éviter les formes trop complexes qui auraient exigé un temps de reproduction inutilement long pour peu de valeur ajoutée.

Nous avons recherché cet objet *Flash* à l'aide de Google, en précisant le type de fichier : « filetype :swf ». Nous devions accompagner ce critère d'au moins un terme ; sans trop nous casser la tête, nous avons inscrit « test », puis nous avons procédé à la recherche. Contre toute attente, le premier fichier proposé par Google respectait nos conditions. Bien que situé sur le site d'une quelconque association américaine, l'objet *Flash* offrait un lien vers un site externe qui semblait lié à ses origines. Une petite visite sur ce second site nous a permis d'y lire que cet objet *Flash* peut être librement utilisé, afin d'ajouter une petite touche d'humour sur notre site Web. Nous en avons conclu que nous ne risquions pas de poursuites judiciaires si nous le récupérions et que nous le traduisions, d'abord en français, mais surtout en SVG.

Méthodologie de reproduction

Le fichier SWF a d'abord été copié localement, puis l'application a été exécutée afin d'en découvrir toutes les subtilités et d'évaluer la durée de chacune des animations. Cette application se présentant plus ou moins sous la forme d'un diaporama de 11 diapositives (appelées ci-après écrans), nous les avons capturées et imprimées. Pour fin de traitement ultérieur en SVG, les écrans ont été numérotés de 1 à 11.

À l'aide d'un éditeur de SVG, d'impressions papier et d'une règle à mesurer, nous avons recréé le deuxième écran et ébauché un gabarit pour les huit écrans suivants, tous construits sur un même modèle. Nous sommes rapidement passée en mode d'édition textuelle, afin de produire du code SVG propre et bien indenté. En effet, les éditeurs SVG en font quelquefois à leur tête relativement à présentation du code, ce qui n'est pas sans nous déplaire... Les éléments textuels et graphiques des écrans 3 à 10 reproduits, nous avons ébauché le onzième et dernier écran, en n'y positionnant d'abord que les éléments statiques.

Nous avons ensuite ajouté le code nécessaire à la gestion des événements permettant un changement de la couleur des boutons lorsque survolés par le curseur, de même qu'au calcul du résultat final en fonction des actions de l'usager. Les animations inter et intra écrans, soit la représentation visuelle de l'écoulement du temps et les changements d'écran déclenchés par un bouton ou l'écoulement du temps, ont ensuite été implantées. Puis, nous avons ajouté l'affichage dynamique du résultat obtenu par l'usager sur le dernier écran. La page de garde étant accessoire, elle n'a été reproduite qu'à la toute fin, d'abord par une détermination des propriétés du texte statique, auquel ont été appliquées une à une rotations, translations et autres transformations.

Il était alors temps d'inclure dans le fichier SVG la définition de la police de caractères utilisée par l'application. À ce propos, nous avons pris soin de n'inclure que la définition des caractères utilisés afin de ne pas inutilement surcharger le fichier. Nous compressions notre fichier SVG, qui passait alors de 61 KB à 12 KB (le fichier SWF fait 28 KB), et nous y étions.

Outils utilisés

Nous avons mené à bien cet exercice sous environnement Microsoft Windows 2000 SP4, à l'aide des outils informatiques suivants :

- Microsoft Internet Explorer 6.0 SP1 ;
- Macromedia *Flash* Viewer 7.0.19.0 ;
- Jasc WebDraw 1.02 ;
- Lemmy 4.5.100 de Software Online (un émulateur de l'éditeur vi pour Windows) ;
- Adobe SVG Viewer 3.2 ;

- Microsoft WEFT 5.3.2 (afin de nous assurer qu'il nous était permis d'inclure la définition de la police choisie dans notre fichier SVG);
- environnement d'exécution Java 1.4.1_07;
- ttf2svg 1.6;
- gzip 1.2.4.

Difficultés rencontrées

Il nous a fallu un certain moment avant de réaliser que SVG exige de connaître l'encodage utilisé par l'application quand des caractères « extraterrestres » (lire: signes diacritiques) sont utilisés. Nous avons donc ajouté au début de notre fichier le bout de code idoine, mais Jasc WebDraw s'obstinait à le faire disparaître. La chose aurait pu être tolérable si l'éditeur n'avait pas fait subir aux lettres accentuées un sort pire encore: « Débuter » devenait « D ?ter » !

Dans la même veine, l'application de conversion de police de caractères ttf2svg ne convertit par défaut que les caractères ASCII, et ce malgré l'utilisation d'une option exigeant aussi le traitement des caractères autres. Ce faisant, toute lettre munie d'un signe diacritique est omise. Nous avons tout de même pu obtenir la définition du *e* accent aigu minuscule en la demandant explicitement via une autre option. La francophilie exige décidément beaucoup de persévérance et d'ingéniosité!

Selon la documentation, il devrait être possible de forcer un changement du type de curseur en fonction de certains événements dans une application SVG. Nous désirions utiliser cette fonction pour que les boutons puissent signaler à l'usager leur réactivité. Il nous a cependant été impossible de réaliser cet exploit sans avoir recours à des liens hypertextuels bidon, c'est-à-dire ne pointant vers rien.

En dernier lieu, il nous a été ardu de déterminer la séquence des rapides animations de la page de garde. Une application *Flash* ne pouvant être ralentie, nous avons dû la stopper, puis la faire reculer (merci au plugiciel!), probablement à raison d'un dixième de seconde à la fois, afin de capturer les différents positionnements des caractères et d'arriver à évaluer lesdites animations.

Analyse des résultats

Malgré quelques problèmes, cet exercice de reproduction en SVG d'un objet *Flash* nous semble concluant. Nous avons pu traduire intégralement l'application initiale, tout en y apportant une petite touche personnelle. Nous sommes néanmoins consciente du fait que notre expérience antérieure en informatique, notre connaissance du HTML, du XML, des feuilles de style, de JavaScript, de Java, ainsi que de la programmation objet, constituait un atout non négligeable face à des outils informatiques pas toujours conviviaux. Si de tels outils exigent souvent de s'aventurer au-delà de l'interface graphique offerte par le système d'exploitation, ils permettent aussi d'avoir un meilleur contrôle des opérations et de mieux comprendre les rouages de l'environnement de travail et de l'application produite.

Conclusion

Au regard des informations et des résultats présentés précédemment, il appert que si *Flash* propose certaines fonctionnalités non offertes par SVG (blocs de texte, son, vidéo), SVG n'est pas pour autant démuni face à son concurrent: qu'on pense au fait qu'il est une norme libre et que son édition ne nécessite qu'un éditeur de texte. Aussi, à moins d'avoir un impératif besoin d'inclure du son ou de la vidéo dans une application vectorielle, dynamique et interactive destinée au Web, l'effort consenti à l'apprentissage de SVG ne peut être que profitable, tant pour les concepteurs que pour les utilisateurs éventuels. Pourraient alors être réalisés des modules interactifs destinés à la formation des usagers en bibliothèque, des cartes géographiques interactives, des sites Web dynamiques indexables par les robots Web, des présentations de type Microsoft PowerPoint, et *tutti quanti*, le tout dans un format lisible par tout un chacun, développé et supporté par un organisme reconnu, et surtout voué à un brillant avenir. ◉

Sources consultées

Adobe. *Scalable Vector Graphics.* <*http://www.adobe.com/svg/*> (consulté sur Internet le 1er août 2005).

Adobe to acquire Macromedia FAQ. <*http://www.adobe.com/aboutadobe/invrelations/adobeandmacromedia_faq.html*> (consulté sur Internet le 1er août 2005).

Apache XML Project. *Batik SVG Toolkit.* <*http://xml.apache.org/batik/*> (consulté sur Internet le 1er août 2005).

Arciniegas, Fabio. 2004a. SVG and Typography: Animation. *XML.com.* <*http://www.xml.com/pub/a/2004/06/30/svgtype.html*>.

Arciniegas, Fabio. 2004b. SVG and Typography: Characters. *XML.com.* <*http://www.xml.com/pub/a/2004/05/12/svg.html*>.

Duce, David. 2001. Scalable Vector Graphics (SVG): Vector Graphics for the Web. *Ariadne* nº 28. <*http://www.ariadne.ac.uk/issue28/graphics/*>.

Dumais, Michel. 2003. Les normes libres, l'enjeu du moment. *Le Devoir*, 21 juillet 2003. <*http://www.ledevoir.com/2003/07/21/32207.html*>.

ECMA. *ECMAScript Language Specification.* <*http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf*>.

Flashmagazine. 2002. *The Flash History: how it all began.* <*http://www.Flashmagazine.com/413.htm*>.

Flash SWF Editors. <*http://www.bestFlashanimationsite.com/resources/swf-editors/*> (consulté sur Internet le 3 août 2005).

Hauser, Tobias and Christian Wenz. 2004. SVG Editors: a Survey about the Market of native SVG Editors. *SVG Open* 2004.

<http://www.svgopen.org/2004/papers/EditorsDesignersPerspective/>.

Held, George *et al.* 2004. *Comparing.SWF (ShockWave Flash) and. SVG (Scalable Vector Graphics) file format specifications. <http://www.carto.net/papers/svg/comparison_Flash_svg/>* (consulté sur Internet le 1er août 2005).

Mozilla. *SVG Project. <http://www.mozilla.org/projects/svg/>* (consulté sur Internet le 1er août 2005).

Opera Software. *Opera browser. <http://www.opera.com/products/desktop/>* (consulté sur Internet le 1er août 2005).

OQLF. *Le grand dictionnaire terminologique. <http://w3.grand dictionnaire.com/>* (consulté sur Internet le 1er août 2005).

Schonefeld, Peter. 2003. *SVG is Real Flash! <http://digitalcraft.com.au/svg/blurbs/01/blurb001.aspx>* (consulté sur Internet le 8 février 2005).

SVG.org. *<http://svg.org/>* (consulté sur Internet le 1er août 2005).

SVGfr.org. *<http://www.svgfr.org/>* (consulté sur Internet le 1er août 2005).

Tayon, Julien. 2002. *Vers un manifeste pour les standards ouverts. <http://www.libroscope.org/Vers-un-manifeste-pour-les>*.

W3C. 1998. *Scalable Vector Graphics (SVG) Requirements. <http://www.w3.org/TR/WD-SVGReq>*.

_____. 1999. *Namespaces in XML. <http://www.w3.org/TR/REC-xml-names/>*.

_____. 2000. *Document Object Model (DOM) Level 2 Core Specification. <http://www.w3.org/TR/DOM-Level-2-Core/>*.

_____. 2001. *Scalable Vector Graphics (SVG) 1.0 Specification. <http://www.w3.org/TR/SVG10/>*.

_____. 2002. *SVG 1.1/1.2/2.0 Requirements. <http://www.w3.org/TR/SVG2Reqs/>*.

_____. 2003. *Scalable Vector Graphics (SVG) 1.1 Specification. <http://www.w3.org/TR/SVG11/>*.

_____. 2004a. *About SVG: 2d Graphics in XML. <http://www.w3.org/Graphics/SVG/About>* (consulté sur Internet le 1er août 2005).

_____. 2004b. *Extensible Markup Language (XML) 1.0 (Third Edition). <http://www.w3.org/TR/REC-xml/>*.

_____. 2004c. *Scalable Vector Graphics (SVG). <http://www.w3.org/Graphics/SVG/>* (consulté sur Internet le 1er août 2005).

_____. 2004d. *SVG Implementations. <http://www.w3.org/Graphics/SVG/SVG-Implementations>* (consulté sur Internet le 1er août 2005).

W3Schools. *Flash Tutorial. <http://www.w3schools.com/Flash/>* (consulté sur Internet le 1er août 2005).

_____. *SVG Tutorial.* <http://www.w3schools.com/svg/> (consulté sur Internet le 1er août 2005).

Wikibooks. *SVG (XML)SVG. <http://en.wikibooks.org/wiki/SVG_(XML)SVG>* (consulté sur Internet le 1er août 2005).

Wikipedia. 2005a. *Macromedia Flash. <http://en.wikipedia.org/wiki/Macromedia_Flash>* (consulté sur Internet le 1er août 2005).

_____. 2005b. *Scalable Vector Graphics. <http://en.wikipedia.org/wiki/Scalable_Vector_Graphics>* (consulté sur Internet le 1er août 2005).

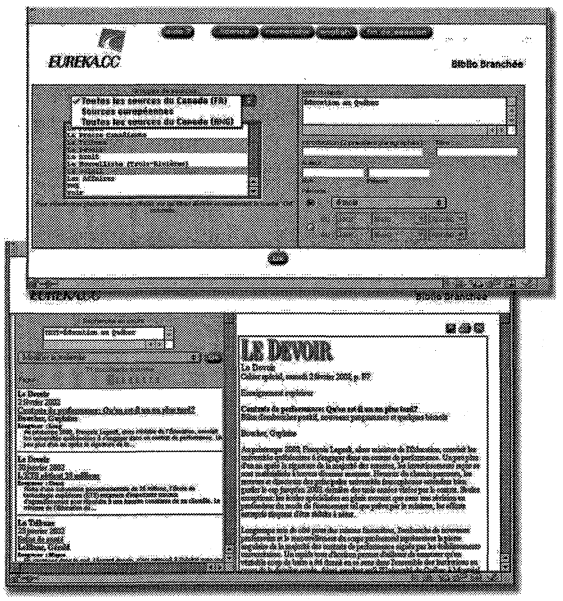
_____. 2005a. *Macromedia Flash. <http://fr.wikipedia.org/wiki/Macromedia_Flash>* (consulté sur Internet le 1er août 2005).

_____. 2005b. *Scalable Vector Graphics. <http://fr.wikipedia.org/wiki/Scalable_Vector_Graphics>* (consulté sur Internet le 1er août 2005).

Biblio Branchée d'Eureka.cc

La solution à vos besoins d'information de presse sur mesure pour les bibliothèques publiques et de l'éducation.

En plus d'accéder aux nouvelles du jour, **Biblio Branchée d'Eureka.cc** vous permet d'effectuer des recherches dans des millions d'articles contenus dans une banque d'archives depuis 1985.
Ce service vous propose un accès Internet illimité à une impressionnante collection de titres de presse provenant de sources d'information locales, nationales et internationales mises à jour quotidiennement. Avec l'ajout régulier de nouvelles sources en provenance du Canada et d'ailleurs, **Biblio Branchée d'Eureka.cc** vous propose plus que jamais un contenu étoffé et de qualité en langue française et anglaise.



www.biblio.eureka.cc

EUREKA.CC
une solution de CEDROM SNI