Scalable Vector Graphics:

The Little-Known Treasure of Document Viewing
1 Introduction

Most everyone has experienced that dreaded feeling of downloading an image from the web to use in a document or powerpoint—say an icon or logo—only to find it horribly distorted in its new location. The pixilation occurring is a byproduct of using bitmap images, such as JPEGs, TIFFs, or PNGs, which are images created with a matrix of dots. To maintain the fidelity of these bitmap images, the size of the image must remain the same.

Luckily, though, there is an alternative to bitmap images—SVG or scalable vector graphics. Unlike bitmaps, vector graphics allow for the scaling and display of lines (text shapes, drawings, etc.) without suffering any loss of clarity at varying resolutions or zoom levels. The benefits don’t end there, either. Along with displaying crisper and higher resolution text and drawings, SVGs are supported in all modern browsers and are often much smaller files, which reduces processing and memory requirements on the server.

In this eBook, we will take a deep dive into all things SVG—describing how SVG became such a popular graphic format for web professionals, expanding on the benefits listed above, and lastly, taking a long look at solutions to consider when viewing and converting documents to scalable vector graphics.

QUOTE:
Doug Schepers, one of the core members of the W3C SVG Working Group, summarizes the appeal of SVG succinctly:

“The fundamental idea of SVG is beautiful: take the best from popular vector programs like Illustrator, and the structure, dynamic adaptability, and hyperlinking of web formats like HTML and CSS, and then add in animation and raster effects like filters to make it fun, funky, and functional.”
SVG was first developed by a special World Wide Web Consortium (W3C) Working Group in 1998. The SVG Working Group sought to establish a standard for graphical interactivity on the web and mobile platforms. Since then, SVG has been under constant development. Because the Working Group developed SVG with all of the same general approaches and framework of HTML (markup, DOM, scripting, styling), SVG is interoperable and coexists with many technologies associated with HTML (and specifically, its most recent version, HTML5), including CSS, JavaScript, and AJAX.

To fully understand how SVG fits in with XML and HTML, it is important to grasp the difference between HTML and XML, which were designed with different goals. XML was designed to focus on what data is and carrying information, while HTML was designed to focus on how data looks and is displayed. SVG, then, is what describes vector-based graphics within the XML format. These graphic elements can be embedded directly into HTML pages. So while all the different technologies have different functions, it is easy to see how XML, HTML, and SVG are technologies that coexist seamlessly with each other.

Because the basic elements of SVG were familiar technologies found in HTML, the learning curve for the early adopters (designers, programmers, web developers, etc.) was minimal. These web professionals could simply leverage their existing HTML knowledge when working with SVG, while avoiding the hassle of learning an entirely new scripting language. The easy-to-learn nature of the technology and the advantages it has over images or canvas-based renderings (rendered pixel by pixel) resulted in a large number of developers adopting SVG as their graphic output technology of choice very early on.
History of SVG continued

Today, the SVG standard has improved to the point that it is now available in all modern web browsers, and by extension, billions of mobile devices. Examples of SVG can be seen everywhere on the current web landscape. Flow charts, business graphics, and mapping are just a few examples of scalable vector graphics being employed on the web. At first glance, it might be hard to determine whether a web page is utilizing scalable vector graphics, but when zooming in on a web page within the browser or downloading an image to view in another client, one will quickly be able to see whether the image pixilated or not. Basically, SVG is agnostic to the size of the screen, the zoom level, or the resolution of the device.

As browser implementations continue to improve, there is less and less need to fuss with cumbersome proprietary SVG plug-ins which require downloads, installations and updating. Also, the SVG Working Group continues to bring SVG and CSS closer and add more features (filters, patterns, and gradients) into the SVG standard, making it more and more appealing for use in HTML documents. All of these factors are leading to more SVG-enhanced web applications, ultimately resulting in more creativity, more complex interactivity (for things like animation), and more real-time data (with its ability to be indexed within search engines).
3 Benefits of SVG

To fully appreciate the benefits of SVG, one first has to understand how scalable vector graphics differ from traditional bitmap images:

**Bitmap Images:** Bitmap images (or raster graphics) are pixel-based, meaning a dot is placed at coordinates for each pixel within a bitmap. Examples of bitmap images include JPEG, TIFF, and PNG.

**Vector Graphics:** Vector-based graphics, on the other hand, compose a picture out of shapes, which are each described by a simple formula and filled with a texture (a mix of colors, gradients, lines, and patterns). Examples of vector-based documents include PDF, DOC, XLS, and AFP.

Since any line or arc can be described as a simple set of parameters that include length, thickness, color, curvature and position, any vector image can be described by a small number of XML instructions. SVG documents can describe any length line or arc without varying file size (for example, specifying a line is 20 units long or 30 units doesn't increase the file size), whereas bitmap formats—depending on the format sophistication—can increase in size with line length or complexity because more pixels may need to be encoded.

**SVG vs. Bitmap**
### Benefits of SVG continued

This leads us to the first of 10 major benefits of SVG:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Image Scaling:</strong></td>
<td>Vector graphics allow for the display of lines (text, shapes, line drawings) to be scaled to any size without incurring any pixilation or loss of detail. When a vector graphic is zoomed in or zoomed out, the resolution does not change and there is no loss of clarity.</td>
</tr>
<tr>
<td><strong>Smaller File Sizes:</strong></td>
<td>Simple vector images such as logos and charts will generally have a smaller file size than if they were a bitmapped image like a JPG, PNG, or GIF. On a large scale, this results in saved memory space and increased performance on machines and servers.</td>
</tr>
<tr>
<td><strong>W3C Standard:</strong></td>
<td>SVG was created by the W3C as an open standard. Anyone can view the source code that underlies the graphic. It is nonproprietary and vendor-neutral, meaning developers may use it without restrictions across platforms and in conjunction with other XML languages. Because of this, SVG is widely used and SVG XML code can be created, verified, manipulated, and compressed using a variety of existing tools (Notepad, Inkscape, etc.).</td>
</tr>
<tr>
<td><strong>Familiar Technology:</strong></td>
<td>As mentioned above, SVG utilizes technologies already familiar to web programmers such as DOM, JavaScript, CSS, and AJAX. Designers, programmers, and web professionals can simply leverage their existing knowledge to utilize SVG.</td>
</tr>
<tr>
<td><strong>Server-Side Generation:</strong></td>
<td>SVG XML can be created and manipulated on the server using PHP (hypertext preprocessor that is an open source general-purpose scripting language that can be directly embedded into HTML), .NET (a software framework that primarily runs on Microsoft Windows), Python (a powerful object-oriented programming language), or any other language/framework.</td>
</tr>
<tr>
<td><strong>Client-Side Generation:</strong></td>
<td>SVG XML can be created and manipulated on the client using JavaScript to create dynamic, interactive effects and animation. Impact on the web-server is light.</td>
</tr>
<tr>
<td><strong>Compatibility:</strong></td>
<td>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.</td>
</tr>
<tr>
<td><strong>Accessibility:</strong></td>
<td>Because the SVG source code is written in XML, both text and drawing elements are machine-readable to screen readers and other devices that can parse the source to create images.</td>
</tr>
<tr>
<td><strong>Web Applications:</strong></td>
<td>SVGs are easily incorporated into HTML5 and web-based applications. All major web browsers, including Chrome, Safari, and Mozilla, currently support it.</td>
</tr>
<tr>
<td><strong>Search Engine Optimization:</strong></td>
<td>SVGs offer improved SEO (search engine optimization) because search engines such as Google, Yahoo, and Bing can index an image’s content. Whether it is a standalone file or embedded directly into HTML, these search engines will index SVG content. Along with being indexed, scalable vector graphics also usually make web pages load faster, which is one of Google’s metrics that determines page rankings. Because of these benefits, SVG documents have a distinct advantage over pixel-based imagery.</td>
</tr>
</tbody>
</table>
Viewing & Converting Documents to SVG

The key to enjoying the benefits of SVG is to utilize powerful and sophisticated document viewing and conversion solutions. When paired together, conversion software can leverage the vector elements in various document types (PDF, AFP, DOC/XLS, and DWG—to name a few) and convert these documents to scalable vector graphics, which can then be viewed and zoomed within a pure HTML5 document viewer.

For those not aware of the benefits of a powerful web-based document viewer, an HTML5 viewer provides users with a sophisticated solution for viewing, manipulating, annotating, redacting, and modifying documents all without having to download or install any additional applications.

Once these vector documents have been converted to SVG, they can be viewed easily as crisp representations of the original documents within a browser-based document viewer. At this point, all of the benefits of SVG can be unlocked. Because all major browsers support vector object scaling, such documents can be viewed and zoomed (in either direction) within the HTML5 document viewer without any loss of clarity. The browser does the literal drawing of the geometric objects (text, shapes, lines) based on formulas (draw line from point “a” to point “b,” for example) rather than simply displaying the dots found in bitmap/raster images.

For vector documents like DOC and PDF files, this results in sharper and more readable content within the browser for text, line drawings and charts. Likewise, since these actions are occurring within the browser, when users zoom, rotate, and scroll through documents they are able to do so quickly without having to rely on any processing from a server (assuming the server software is properly designed).
This is also a major benefit for CAD (Computer-Aided Design) files, such as DWG and DXF. These files contain 2D or 3D design data and vector-based graphics to depict the objects of traditional drafting, often for the purposes of engineering and mechanical design. When CAD drawings are saved as raster images for the purposes of viewing (because many devices don't support viewing of the CAD format), the fidelity of the images is often compromised when zoomed. However, when converted to SVG and viewed within a pure HTML5 document viewer, these files can be accessed in high fidelity from any device or browser—providing incredible flexibility for users looking to view CAD files in the field and on the go.

In all of these SVG viewing scenarios, however, it is important not to discount what is happening on the server. Even though the server is more or less out of the picture after the conversion happens because the processing is occurring on the client/browser, the server is responsible for controlling the conversion process and the generation of the SVG. A lot of hard work must go into the technology on the server-side to ensure everything works correctly and efficiently within the browser. The converted and formatted SVG documents, as well as the HTML5, JavaScript and CSS web page content, must be generated correctly to permit an accurate, well-performing document display. A PDF that is rendered imperfectly on your server into a browser-compatible SVG will look bad, no matter how good the browser’s SVG technology might be. It is important to be cautious of bad SVG generators, of which there are many.

Of course, what image type you start with matters. Embedded images such as JPEGs or TIFFs within an SVG document can’t improve because they are still bitmaps. However, CAD images (almost always vector-based), text (PDF, Word, AFP) and most charts can be drawn as vector documents.
The Difference Between Converting Vector Documents & Rasterizing Bitmap Images

The action described before—using a powerful document conversion tool to leverage the vector elements of documents such as PDF or DOC and convert them to SVG—should not be confused with other SVG conversion tools, often found online. Many of them do not actually convert the document. Instead, they will often just encode the image as base64, which is just another form of a bitmap image.

Other SVG conversion tools will claim to convert raster/bitmap graphics by tracing or “vectorizing” the image. The issue becomes much more difficult because the image is a set of individual pixel values, not a set of vectors. There is much room for error with this process. Images converted in this manner often become too large because there is too much conversion fidelity with the color and shape or they become too unlike the original. In the latter case, a number of shading effects from the conversion become embedded in the new SVG image, often resulting in cloudy or jagged edges. Because of these bitmap to SVG conversion pitfalls, the resulting image almost universally requires manual correction after conversion, making it an imperfect process. Those looking for such a conversion tool should be naturally skeptical of any conversion tools promising otherwise.

It is important, then, to find a vendor that has developed its own SVG conversion technology. A proper SVG converter will take the original document’s (PDF, AFP, MS Office, etc.) vector drawing instructions and translate them to the vector drawing instructions in SVG. If there is a raster or bitmap image within the original document, it can be copied from the original into the destination.
Closing

Scalable vector documents are the future of graphic imaging on the web. They can be created and edited with any text editor, they can be searched and indexed by all major search engines, they are scalable, they can be printed and zoomed with high quality at any resolution, and they are pure XML and an open standard—meaning it will only continue to improve as people collaborate on it more and more.

Today, businesses are already rapidly moving from applications that need to be installed and supported to HTML5 solutions to lighten the load on IT as well as reduce headaches from end users. Adding SVG capabilities is a natural extension of that movement and one that many businesses will continue to make.

For users who interact with scalable vector graphics on a large scale, it is important to find the right document viewing and conversion solutions to fully unlock the benefits of SVG documents within a daily workflow. As mentioned above, not all of these solutions are created equally. Find a vendor who has developed their own SVG technology and has put thought into their server-side design so all of the above benefits of SVG can be fully realized.