Styling Map Elements for Geometric Structures

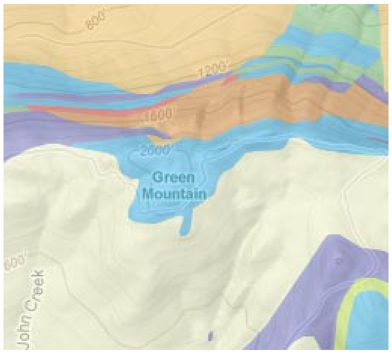
MicroImages' KML and SVG geometric tilesets are tiled structures that are created by TNTmips to present map data in a web browser in styled geometric form for any size area over a range of Google Maps zoom levels. (See the *Tileset* Technical Guides entitled *SVG Geometric Structure* and *Geometric KML Structure* for descriptions and illustrations of these TNTmips capabilities.) The Export Geometric Tileset process, which creates these tilesets, renders geometric elements from a source vector object into KML or SVG tiles using the display style settings stored with that source object. The tiles for each zoom level are independently rendered directly from the source vector object at the map scale corresponding to that Google Maps zoom level. Source vector lines are automatically simplified (thinned) to reduce line complexity for successively lower zoom levels (for more details see the *Tileset* TechGuide entitled *Export Geometric Structures*).

The SVG and KML formats used in geometric tilesets differ in their capabilities to store and present complex map styles. This Technical Guide presents guidelines for setting up styling for the various types of geometric elements in the vector objects that you intend to convert to SVG and KML geometric tilesets. See the Tutorial entitled *Creating and Using Styles* for general instructions on setting up styles for vector elements.

A geometric tileset can be used on a website to present styled map data over a very large range of map scales. Careful design is required to set the display styles in the TNT vector object you plan to convert into a geometric tileset so that its graphic elements are appropriately styled over a large range of map scales. The *Tileset* TechGuide entitled *Scaling Map Elements for Geometric Structures* suggests general strategies for scaling graphic element styles to produce appropriately-scaled map elements in geometric tilesets.

Styling Polygons

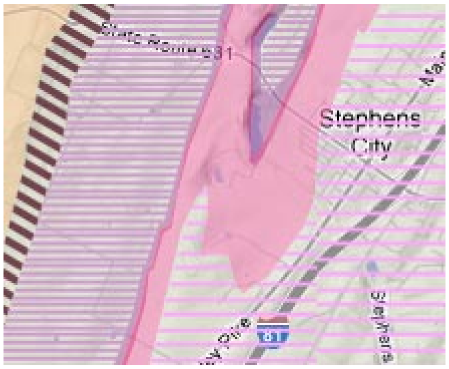
In TNTmips you can style polygon fills using bitmap patterns, hatch (geometric) patterns, or solid colors with variable transparency. Bitmap patterns cannot be rendered to geometric tilesets and so should not be used in your source vector objects. Solid color fills work best in both KML and SVG geometric tilesets, as they can be very compactly described in the tile files. Using partially transparent fills allows other underlying layers and the Google Maps base map to be visible through your tileset overlay. Transparency values of 35% to 50% work well for light-toned to moderate-toned overlay colors (see illustration below), retaining adequate color saturation while still allowing underlying layers to be visible.



Geologic map polygons in a KML geometric tileset overlay shown over the Google Maps terrain basemap. All fill colors for these polygons in the source vector object were set with a transparency of 35%. A global transparency value can also be set in the Assemble Geomashup process for all colors in a tileset layer.

The Assemble Geomashup process lets you select geometric tilesets to be used as overlays in Google Maps. It also lets you set a global transparency value for all colors in the layer. This setting overrides any transparency values set for individual colors in the KML or SVG tile files. Thus you do not need to set transparency values individually for each fill color in the source vector object prior to making the tileset; you can instead render fully opaque polygon fills to the geometric tileset yet display them in the Google Maps mashup with partial transparency. You can also choose to add a slider to the layer controls for the mashup that lets the user dynamically alter the transparency of the layer at any time.

Hatch fill patterns can also be rendered to geometric tilesets, but are not generally practical for KML tilesets because they introduce too much graphical complexity into the KML tiles. That complexity results in larger KML tile files that take more time to parse and render in the browser. Because SVG tile files describe graphic data more compactly and are directly interpreted by browsers, hatch fill patterns can be used in SVG geometric tilesets without noticeably affecting viewing performance.



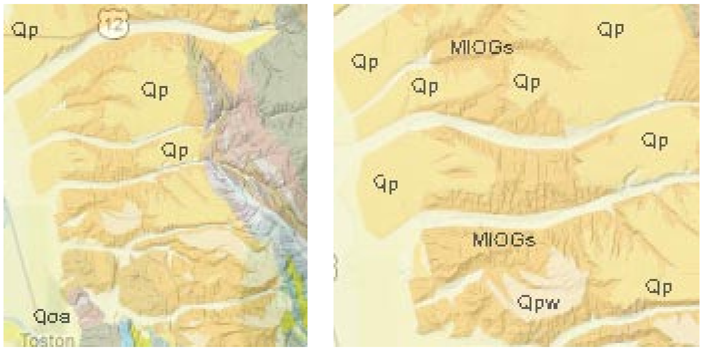
Geologic map polygons in an SVG geometric tileset shown over the Google Maps terrain basemap. Three of the polygon classes show here were styled using hatch patterns in the source vector object.

Vector polygon elements can also be styled in TNTmips with a border, which can be either a solid line style or a line pattern. If you wish to use line patterns to style the borders of polygons that are contiguous (sharing common line boundaries), it is best to style the polygons without borders and instead apply the desired "border" style to the corresponding line elements in the source vector. Otherwise the border patterns of adjacent polygons may conflict with or overwrite each other.

Polygons in a vector object in TNTmips can also be set to show dynamic text labels from attached attributes in the polygon database. Although the KML format does not support text labels, the SVG format does. When you make an SVG geometric tileset these dynamic labels are rendered into the SVG tiles independently for each zoom level. It is usually best to set the sizes of these labels in screen pixels independent of map scale so they retain a constant screen size at all zoom levels. It is also best to set the position option for these labels in the Vector Layer Controls window to *Fit Inside or None*. With this setting, labels are only shown if they fit inside the parent polygon. Thus at low zoom levels only the larger polygons have labels, but as you zoom in to higher levels more and more labels appear as the screen size of the smaller polygons increases (see illustration on the reverse side of this page). You can

(over)

also set an explicit range of map scales within which the dynamic labels are shown in the vector object. This scale range is retained when the vector object is converted to a tileset and so can be used to exclude labels entirely at the lowest zoom levels.

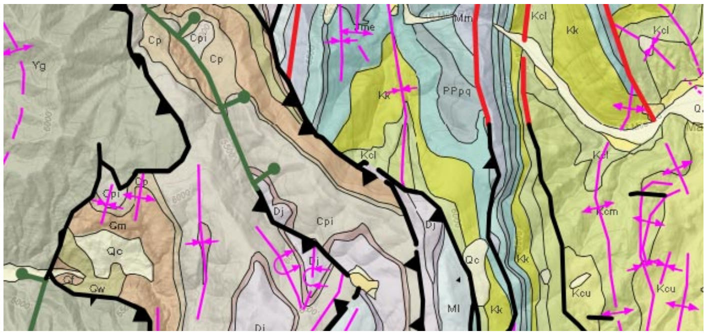


Google Maps views of SVG geometric tileset with geologic map polygons created from a TNT vector object with dynamic polygon labels with positioning setting *Fit Inside or None*. Left view is at zoom level 10, right view at zoom level 11. Since polygons and their labels are rendered to each zoom level independently, more labels appear at higher zoom levels as the screen size of polygons increases relative to the fixed-size labels.

Styling Lines

Line elements in TNT vector objects can be styled as solid lines of varied colors and widths, or using line patterns, such as dashed or dotted lines. You can also create and use complex static line patterns combining multiple solid, dashed, or dotted elements, or use CartoScripts to generate complex custom symbols that are rendered dynamically for each line based on line attributes (see the Tutorial booklet entitled *Using CartoScripts*).

Complex line patterns and custom CartoScript line styles are reproduced with excellent fidelity in SVG geometric tilesets, as the SVG file format directly supports dashed line styles, and individual components of more complex symbols are rendered as graphic elements into the individual SVG tile files at each zoom level. Examples of such styles in SVG tilesets are shown in the illustration below.



Google Maps geomashup of several SVG geometric tileset layers showing elements of a geologic map. SVG layers with line elements representing fold lines (pink) and faults (black, red, and green) have complex line symbols created dynamically by CartoScripts from the line attributes during rendering. These complex symbols are reproduced well in the SVG tilesets.

On the other hand, the KML file format provides more limited options for styling lines, recording only the color and width of solid lines. When you are styling vector lines for conversion to a KML geometric tileset, it is best to use solid line styles with varying widths and colors. Although the KML format does not explicitly support dashed lines, the Export Geometric Tileset process automatically converts dashed lines to multigeometry elements when creating a KML geometric tileset. An individual solid line element is stored for each dash, but all "dash" elements for a particular vector line are grouped together to share the same style and a single set of attributes. Complex CartoScript line symbols are also automatically tranferred to KML with each part of the symbol stored as an individual graphic element, though these symbols are reproduced in KML less faithfully than in SVG. However, dashed lines and more complex symbols add significantly to the size and complexity of the KML tile files, which increases the time required in the browser to interpret and render the elements. Complex line styles should thus be used sparingly in vector objects that are used as sources for KML geometric tilesets.

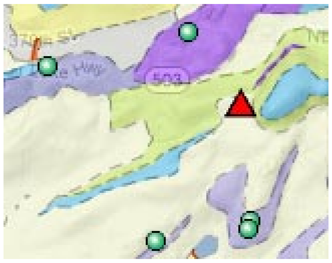
Styling Points

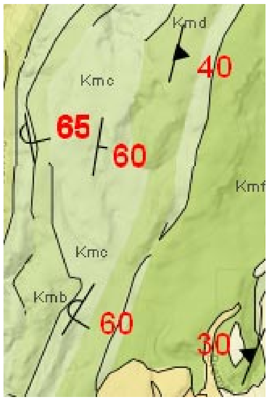
Point elements in TNT vector objects can be styled with:

- geometric symbols ranging from simple predefined shapes to complex symbols you design,
- bitmap point symbols, and
- custom graphic symbols that are rendered dynamically from attributes during display using CartoScripts

Geometric point symbols and CartosScript symbols are rendered directly into the tile files of SVG geometric tilesets as graphic elements with appropriate styling. The SVG format supports the concept of geometric point symbols, so SVG geometric tilesets can faithfully reproduce point symbols of varied complexity, including complex custom CartoScript renderings. Bitmap point symbols are automatically embedded as bitmaps within the individual SVG tile files.

When you create a KML geometric tileset, all static symbols (geometric and bitmap) are rendered to very small PNG image files that are automatically linked to the output KML files. Each unique symbol is represented by only a single PNG image file, which is linked to all occurrences of that symbol in the tileset, avoiding redundant symbol files. These PNG image files are able to reproduce typical point symbols quite well. If the points in the source vector object have custom symbols rendered dynamically by a CartoScript, each part of the resulting symbol is rendered as an individual graphic element into the KML tile files. This procedure can capture simple to moderately complex symbols created by CartoScripts. However, the KML file format does not explicitly support the concept of a geometric point symbol or predefined geometric shapes (circles, rectangles, etc.), so the KML equivalent of a TNT CartoScript symbol that uses geometric shapes may be only approximated. SVG tilesets support geometric shapes and do a better job of reproducing complex, CartoScript-rendered symbols.





Above, KML geometric tilesets with PNG point symbols rendered from simple geometric symbols. Right, SVG geometric tileset with complex symbols rendered from TNT CartoScript.