PTC®

Document - CS230452

Document Information

Type	Article
Date Created	23-Mar-2016
Date Last Updated	02-Mar-2016
Rating	4 Votes

Document Details

Title	Enhancements to the SVG image syntax generated by Creo Illustrate 3.1 F000 and Arbortext IsoDraw 7.3 M060 and later
Description	• The major releases of Creo Illustrate 3.1 F000 and IsoDraw 7.3 M060 include a number of important SVG changes • Additional changes have been implemented in Creo Illustate 3.1 M010 and IsoDraw / IsoView / IsoConvert 7.3. M060 • Enhancements were made to the SVG syntax generated by Creo Illustrate and IsoDraw • The changes had one primary objective: allowing end users the create SVG applications faster and easier. • This article provides details about the syntax of the SVG files as exported by these products **Customer context:** • Where is the enhanced SVG Syntax that's exported by Creo Illustrate 3.1 F000 and IsoDraw 7.3 M050 documented?
Applies To	• Creo Illustrate 3.1 F000 • Arbortext IsoDraw CADprocess 7.3 M050
Cause	
Resolution	• Reported to R&D as SPR 4818358 against documentation

Missing Documentation:

Table of Contents:

Creo Illustrate 3.1 F000 SVG Changes

Version Information

While we will try to preserve the current SVG syntax in future releases of Illustrate, it is always possible that some syntax will change. In order to offer customers backwards capabilities a version string was added to the SVG syntax. The version information can be found as an XML comment under the **doctype** node. See the comment starting with "**<!– Generator:**" in the SVG snippet below:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN" "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">
```

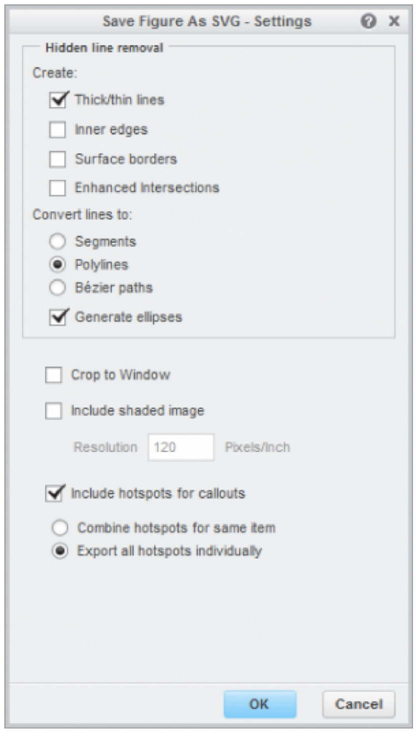
```
<!-- Generator: PTC Creo Illustrate, Creo 3.1, version: 10.3.15.1 -->
<svg width="100%" height="100%" viewBox="0 0 100 50" ...>
```

Callouts

A number of enhancements were made to the callout syntax in this release of Creo Illustrate. The two main changes are:

1. The grouping of callout geometry
2. A new option for creating callout overlays (i.e., **hotspot**)

In previous versions, it was difficult to add event handlers (ex: mouse-click) on call-outs but the lack of a parent group was causing this problem. Keep in mind that events can be triggered from any parts of the geometry (leader line, text, balloon, etc...). This makes it challenging to manage the events without an event handler on each items of the geometry.


Save Figure As SVG - Settings

Hidden line removal

Create:
- [✓] Thick/thin lines
- [ ] Inner edges
- [ ] Surface borders
- [ ] Enhanced Intersections

Convert lines to:
- ○ Segments
- ● Polylines
- ○ Bézier paths
- [✓] Generate ellipses

- [ ] Crop to Window
- [ ] Include shaded image

Resolution 120 Pixels/Inch

- [✓] Include hotspots for callouts
  - ○ Combine hotspots for same item
  - ● Export all hotspots individually

The structure was improved in X-26, the callout geometry is now contained within a group. For example:

```
<g id="ID_1055" class="callout callout_8">
   <line .../>
   <path .../>
   <text ...>8</text>
</g>
```

With the new structure, event handlers can be added at the <g> level.

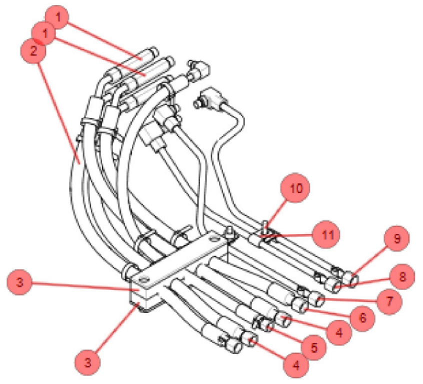
```
<g id="ID_1055" class="callout callout_8" onclick"alert('8');">
   <line .../>
   <path .../>
   <text ...>8</text>
</g>
```

Note: The SVG filter does not add the event handlers, it must be done programmatically by the user. This is frequently done by web developers. Javascript (or another language) can add event handlers upon loading of the SVG file. Because the **onclick** is assigned to a container element (**<g>**), the click will be handled regardless of the mouse event's origin (leader line, text, balloon).

The two samples above make use of the **class** attribute, for further detail refer to the 'class' attribute section.

Another important enhancement in this release is the option for generating callout overlays (or hotspots). Depending on the SVG usecase, choosing to generate callout overlays can provide a significant convenience factor.

Callout overlays are an invisible but clickable region on top of the callout geometry. The image below illustrates what callout overlays may look like:



They are typically invisible but were made semi-transparent for the example. The main purpose of the callout overlay is two fold:

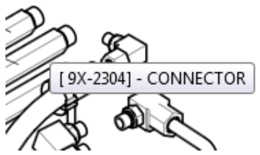
1. Better handle multiple occurrences of the callout item, ex: six occurences of callout **1**
2. Simplifying the java script

Choosing to use callout overlays or not really depends on what the user is trying to do. Overlays make the file a bit larger (5-10%), therefore if file size is critical, a user may want to re-consider this option. If file size is not too important, then using the option can probably simplify the scripts and offer more possibilities to the end user.

Enhanced in Creo Illustrate 3.1 M010. Introducing the new hotspot grouping option.

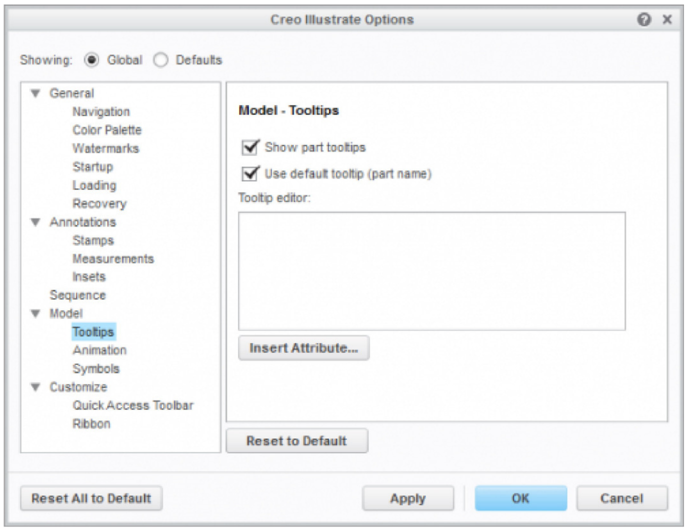
Tooltips

The SVG export filter is now hooked up with the application's tooltip options. In Creo Illustrate, a user can specify if a tooltip should be displayed when hovering over a part. By default, the tooltip text will match the sBOM part name but a user can override the default and customize the tooltip to display something else. For example: a part attribute value:



In SVG 1.1, tooltips can be displayed by using the **<title>** element. **Note**: Browsers are not required to display tooltips but the modern browsers seem to do so (Firefox, IE10, etc...)

If the Creo Illustrate option **Show part tooltips** is checked, the SVG filter will honor the option and generate tooltips. For example, the syntax for a default tooltip could be **<title>CRANK_SHAFT.PRT</title>**. If the user decides to create a custom tooltip and say, retrieve the **PTC_WM_MODIFIED_ON** attribute value, the custom tooltip syntax might look something like **<title>Last modified on 28-Apr-09 04:11:52 PM</title>**



Model - Tooltips

✓ Show part tooltips

✓ Use default tooltip (part name)

Tooltip editor:

If the user doesn't want tooltips in the SVG file, he/she may choose between the two following options:

a. Turn Off **Show part tooltips** before exporting the illustration.
b. Programmatically remove the **<title>** elements when loading the illustration

In the event where an end user wants a different appearance then the browser tooltip, a user may:

i. Programmatically extract the **<title>** character data
ii. Write a script routine to display the extracted content using SVG markup
iii. Then, finally remove the **<title>** elements from the SVG file

Part names

Prior to Creo Illustrate 3.1, the sBOM part names could be found as a **class** attribute value. While the syntax is valid, that approach limits the usefulness of the **class** attribute. For more details on how to use the **class** attribute see the 'class' attribute section.

In Creo Illustrate 3.1, the sBOM part names are found in the **<desc>** elements. The **<desc>** element is a child node of the sBOM part, e.g.:

```
<g id="ID_1250" class="part part_1">
   <title>[9X-2304]</title>
   <desc>[ 9X-2304] - CONNECTOR</desc>     <!-- <<< contains part name -->
   <!-- part geometry -->
</g
```

Knowing the part names may be meaningful to an SVG application. e.g. if creating a parts catalog application. A user can retrieve the part names using scripting or XSLT.

The 'class' attribute

Class names have many use cases, the two main ones being:

a. Styling
b. General purpose information

An element can have more than one class name assigned to it. If a set of class names are assigned to an element, they must be separated by a white space, e.g.:

```
class="building house bungalow"
```

Class names can be very useful to script writers. A new DOM API called **getElementsByClassName()** which is currently in Working Draft state is widely supported by browsers. This method allows to retrieve all elements with a given class name with a single call.

The Creo Illustrate SVG filter will assign class names to specific elements found in the illustration.

hotspot class

States that the current element is a callout overlay (hotspot) as in the following SVG snippet:

```
<path id="ID_1483" class="hotspot hotspot_1" .../>
<path id="ID_1484" class="hotspot hotspot_2" .../>
<path id="ID_1485" class="hotspot hotspot_3" .../>
```

If a user wanted to add an event handler to the above hotspots, he/she would likely write something along these lines:

```
var hotspots = document.getElementsByClassName("hotspot");
   for (var i=0; i < hotspots.length; i++ ) {
      hotspots.item(i).addEventListener("click", function(e) {
          myClick(e, e.target.id); }, false);
   }
```

Enhanced in Creo Illustrate 3.1 M010. The overlay syntax was modified slightly for improved flexibility.

callout class

States that the current element is a callout. The SVG filter currently assigns two class names to callout elements. The first is a general **callout** class name. The second is concatenation of **class** and the current item number, for example: **class_2** or **class_ii**.

If a script writer wants to retrieve all callouts, **getElementsByClassName("callout");** will do the trick. If a user wants to retrieve all occurrences of callout **1**, **getElementsByClassName("callout_1");** will return that list.

part class

States that the current element is associated to a callout. It is important to note that with the current implementation, not all sBOM parts are assigned a **part** class, only the parts which were called out.

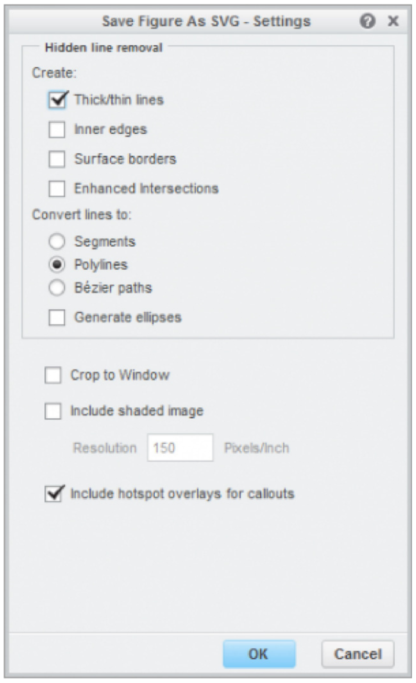
Somewhat similar to callouts, the part class is a concatenation of **part** and current item number. For example: a part which has a callout **1** pointing to it will be assigned the **part_1** class name. The entire **class** attribute value could be **part part_1**.

Enhanced in Creo Illustrate 3.1 M010. A minor change was done to the **class** attribute value.

Maintenance Release Changes

Note: The following information is forward looking and subject to change without notification.

Creo Illustrate 3.1 M010 brings the following syntax changes



Hidden line removal

Create:

- [✓] Thick/thin lines
- [ ] Inner edges
- [ ] Surface borders
- [ ] Enhanced Intersections

Convert lines to:

- ○ Segments
- ● Polylines
- ○ Bézier paths

- [ ] Generate ellipses
- [ ] Crop to Window
- [ ] Include shaded image

Resolution 150 Pixels/Inch

- [✓] Include hotspot overlays for callouts

Hotspot grouping options

As it was introduced in Creo Illustrate 3.1 F000, all overlays for a given ITM will be grouped into one overlay. With Creo Illustrate 3.1 M010 the option **Combine hotspots for same item**, has been introduced to control this behavior. If a user chooses to export overlays individually, then overlays for the same item are not grouped together. This allows the script writer to target a specific overlay out of multiple occurrences. This output could be useful for a service procedure for example.

Overlay syntax

The overlay syntax is slightly different compared to F000. The **id** attribute value was appended to the **class** attribute value.

M010 syntax: **<path id="ID_1483" class="hotspot hotspot_1" .../>**
F000 syntax: **<path id="hotspot_1" class="hotspot" .../>**

The new syntax provides more flexibility for the end-user. It allows the application to export hotspots individually which was previously difficult because of ID duplication.

Part syntax

The **class** attribute value was modified from **class='part_1'** to **class='part part_1'**. This change allows the script writer to retrieve all parts in a single DOM call.

Relationship with IsoDraw and IsoConvert 7.3 M060

Most of the syntax described above also applies to **IsoDraw** / **IsoConvert**. Users should take note that the concept of callouts only exists in IsoDraw files, it does not exist in CGM. For that reason, syntax containing **part** and **callout** attribute values will only be generated when converting IDR files to SVG. When converting CGM to SVG, the **hotspot** class attribute value will be used instead.

A new **Hotspot grouping** SVG export option is implemented for IsoDraw 7.3 M060. This option controls whether two of more region hotspots with the same name should be grouped as a single hotspots overlay. Exporting those region hotspots individually (instead of grouping them) can allow a script writer to highlight a specific hotspots instead of highlight all hotspots with the same name.
Note: For **IsoConvert** this option can be set in **IsoConvert.prf** by adding the line **Export Hotspots Individually: 0**

Related

- Hotspot from 7.3 M060 and above does not display in the same way in SVG format files compare to SVG files from M050 ?
- Hotspots included in an exported SVG file are not available (Creo Illusrate)

SPR Information

Related Documents

Type	Number	Status	Description