SVG viewer for Thingworx

This is a widget allowing visualization of svg files. It allows viewing the files, and also change element attributes based on external data. It also allows zooming and panning.

How to use

Here is how to use the widget in a simple usecase:

The svg file

You have your SVG file, where the elements that you want to update/interact with are clearly identifiable (via `id` or a custom attribute, like the `tagName` below) The SVG file can look something like this:

```
<svg>
  <g tagName="myId1">
    <ellipse cx="2.12898292" cy="1.64099586" rx="1.59673719" ry="1.64099586"/>
    <path d="M2.12898292,3.55549103 L2.12898292,8.4784786" stroke-linecap="round"/>
  </g>
  <g tagName="myId2" transform="translate(108.000000, 8.000000)" stroke="#C6D8E1"
stroke-width="2" stroke-linecap="round">
    <path d="M1.5,5 L14.5,5" id="Line1"/>
    <path d="M11.5,1 L24.5,1" id="Line2"/>
    <path d="M17.5,5 L21,5" id="Line3"/>
  </g>
  <g tagName="myId3">
    <ellipse cx="2.12898292" cy="1.64099586" rx="1.59673719" ry="1.64099586"/>
    <path d="M2.12898292,3.55549103 L2.12898292,8.4784786" stroke-linecap="round"/>
  </g>
</svg>
```

You link that svg file into the widget. It's important to bind the link to the svg file rather writing the file path directly in the widget properties (bug at the moment). The svg file can be uploaded to a file repository, to a media entity, or accessible using an url.

The data infotable

You create an infotable that contains "overrides" (more on that later). This infotable would look something like this:

ElementName	AdditionalMetadata	override-fill	override-stroke	override-stroke-width
myId1	This idtest	red	yellow	3.5
myId2	Test foo	#431234	green	2

The only required column is `ElementName`. This column maps the override rows to the elements in the svg. For example, the infotable above will change the `fill`, `stroke` and `stroke-width` of all the elements in the group `myId1` to the values (red, yellow and 3.5 respectively).

The elements in the infotable also dictate what elements are clickable. The infotable above will make the groups `myId1`, `myId2` selectable. The group `myId3` will not be clickable, as it's not included in the infotable.

It's also important to note that the `override-fill`, `override-stroke` are just examples. You can override any attribute of the svg if you prefix its name with `override-`. However, there are a couple of special `overrides` that you can include: * `override-tooltip`: Specify a tooltip for an element. This is visible when hovering. * `override-text-stroke-width`: If you override the `stroke-width` of a group, you may find that the `text` elements are hard to read. This is a special override for the `stroke-width` that applies only to `text` elements.

Bindings and properties

- Bind the url to the svg file to the `SVGFileUrl` property. Do not select a MediaEntity using the entity picker. A binding must be created.
- Bind that infotable into the `Data` property of the widget. This includes the overrides, as well as other information that can be used when binding to other widgets. Syncronized selection exists, so if this infotable is also bound to a list you can select from the list and see the selected element in the svg.
- Select the column in the infotable that contains the nanmes of the elements (in this example, it's `ElementName`).
- In the `SVGIdField` property, enter the tag name in the svg that maps with the `DataIdField`.
- Configure the `SelectedStyle` (style of the selected elements).
- Depending on your svg file, you may want to uncheck `ApplyToChildren`. This specifies whether to apply the overrides to the element or to its children.
- Decide if you want `ZoomPanEnabled` or not. If it's enabled, than you can configure the initial zoom level or the position. If not, you can scale the SVG as you want.
- There is also another property, `DexpiDataSource`. If you know what it is, keep it enabled. If not, it must be disabled. HINT: if your svg file was generated from the DEXPI Graphics Builder, is must be enabled.
- `SelectedElementID` also points to the current selected element.

The following events are also available: * `ElementClicked`, `ElementDoubleClicked`, `ElementMiddleClicked`. They are triggered after a named element has been clicked.

The following service is available: * `PanOntoSelected`: Pans onto the selected element, as to bring it into the center of the screen. `ZoomPanEnabled` must be set to true for this to work.

Building and publishing

The following commands allow you to build and compile your widget:

- `npm run build`: builds the extension. Creates a new extension zip file under the `zip` folder.
- `npm run watch`: watches the source files, and whenever they change, do a build
- `npm run upload`: creates a build, and uploads the extension zip to the thingworx server

configured in `package.json`.