⚠ The review is overdue, so the information might be old. Please contact the responsible.

To login form

WORK INSTRUCTIONS, HOW-TO'S

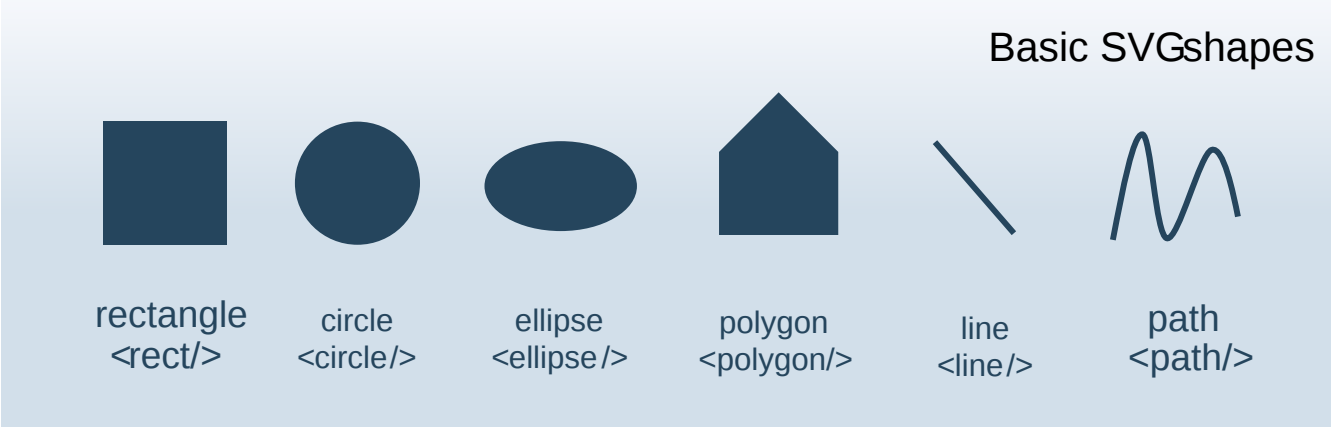
How to create SVG animation

This page helps to understand SVGs and explains how to create them.

Understanding SVG

General information

Instead of delivering a graphic as a PNG in different resolutions, SVG brings optimal graphics for every monitor size at a low price. SVG is plain text like HTML, it can be generated with programs such as Inkscape and Adobe Illustrator, Microsoft Viso or Powerpoint, formatted with CSS and dynamically changed with Javascript. **Like** HTML, SVG tags are written in angle brackets, they must be closed and can contain attributes such as id, class, width, fill etc. **Unlike** HTML, SVG values needs correct spelling and does not forgive mistakes. Scalable Vector graphics can contain tags for rectangles, circles, lines, polygons and elements for paths, texts, animations and raster graphics (bitmap or pixel images). Usually you do not have to write the code with all elements yourself, the progam you use to create a graphic will generate the SVG code when it is exported into a SVG file. Knowing the basic shapes will help you to understand the structure of any SVG code.



SVG Elements

Beside the basic shapes there are more elements that can used in a SVG code. The graphic elements are arranged or superimposed on the drawing area in the order in which they are defined. What lies further down in the source code lies above the previous SVG tags and hides its predecessors if it is not transparent.

Element	Description
<svg>	The <svg> element is a container that defines a new coordinate system and viewport. It is used as the outermost element of SVG documents, but it can also be used to embed an SVG fragment inside an SVG or HTML document.

Element	Description
<metadata>	The <metadata> element adds metadata to SVG content. Metadata is structured information about data.
<style>	The SVG <style> element allows style sheets to be embedded directly within SVG content.
<script>	The SVG <script> element allows to add scripts to an SVG document.
<desc>	The <desc> element provides an accessible, long-text description of any SVG container element or graphic element.
<defs>	The <defs> element is used to embed definitions that can be reused inside an SVG image. For instance, you can group SVG shapes together and reuse them as a single shape.
<use>	The <use> element takes nodes from within the SVG document, and duplicates them somewhere else.
<g>	The <g> element is a container used to group other SVG elements.
<a>	The <a> SVG element creates a hyperlink to other web pages, files, locations in the same page, email addresses, or any other URL. It is very similar to HTML's <a> element.
<text>	SVG renders text in the same way as shapes. The SVG text tag contains information about the position of the text, transformations and properties that are specified as CSS style or SVG attribute.
<tspan>	The <tspan> element is used for multi-line texts.
<image>	The <image> element is used to integrate or embedded Images (.png, .gif, .jpg) in a SVG.
<symbol>	The <symbol> element is used to define graphical template objects which can be instantiated by a element.
<animate>	The <animate> element provides a way to animate an attribute of an element over time.
<clipPath>	The <clipPath> SVG element defines a clipping path, to be used by the clip-path property. A clipping path restricts the region to which paint can be applied. Conceptually, parts of the drawing that lie outside of the region bounded by the clipping path are not drawn.
<linearGradient>	The <linearGradient> element lets authors define linear gradients that can be applied to fill or stroke of graphical elements.
<radialGradient>	The <radialGradient> element lets authors define radial gradients that can be applied to fill or stroke of graphical elements.
<stop>	The SVG <stop> element defines a color and its position to use on a gradient. This element is always a child of a <linearGradient> or <radialGradient> element.
<set>	The SVG <set> element provides a simple means of just setting the value of an attribute for a specified duration.

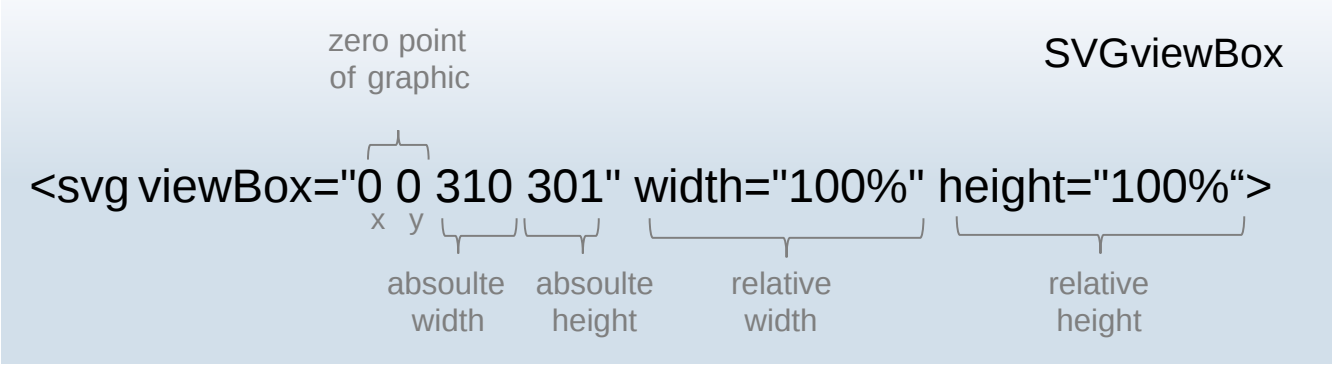
Edit code

For editing the SVG Code use atom and open the SVG Toggle Preview to see how changes in the code affect the visual representaion of the graphic.
Install Package in atom (Package – search SVG-preview from josa42).

SVG viewBox

The SVG-viewBox attribute determines the visible section and the aspect ratio of a graphic. Just don't overlook the fact that viewBox is written with a capital "B„ in the code!

The viewBox is structured as follows:



(Basically this is what the first line should look in your code. Just the absolute width and height are variable sizes.)

- Offset the origin of the graphic by negative values, the object moves to the right / down out of the viewBox.
- Positive values for x / y shift the object to the left / up.
- Increasing width / height makes the object smaller, decreasing width / height zooms in, on the object within the viewBox.

If the relative width and height are set to 100%, all modern browsers adapt the graphic to the space available in the comprehensive container which makes the SVG responive. It It even is scalable bigger or smaller when embed later.

Animation

There are two different types of animation in CSS: transition and keyframe animation. Transitions are easier to use compared to keyframe animation, but there are fewer animation options. Keyframe animations, on the other hand, have a greater scope for animation, but are much more extensive in the source code.

SVG CSS style attributes

You can use following attributes to style a SVG and create an animation.

Attribute	Description	Property	supported by
color	The color attribute can use to change text or backround color of elements.	You can use color names, rgb values, hex values, rgba values, etc.	Elements using fill, stroke, stop-color
opacity	The opacity sets the opacity of an element. Opacity is the degree to which content behind an element is hidden, and is the opposite of transparency.	0 (not visible) to 1 (full visible)	Container, graphic & text elements
display	The display attribute sets whether an element is treated as a block or inline element.	none (not displayed), block (displayed)	Container, graphic & text elements
fill and stroke	Using fill sets the color inside the object and stroke sets the color of the line drawn around the object.	You can use color names, rgb values, hex values, rgba values, etc.	Graphic elements & text content elements
stroke-dashoffset	A presentation attribute defining an offset on the rendering of the associated dash array.	numerical values	circle, ellipse, path, line, polygon, polyline, rect, text and tspan
stroke-dasharry	A presentation attribute defining the pattern of dashes and gaps used to paint the outline of the shape.	numerical values	circle, ellipse, path, line, polygon, polyline, rect, text and tspan

Attribute	Description	Property	supported by
cursor	The cursor attribute sets the type of mouse cursor, if any, to show when the mouse pointer is over an element.	default, auto, pointer etc. see more here	Container elements & graphic elements

Create a CSS hover animation with transition

A CSS hover animation occurs when the mouse hovers over an element, and the element responds with motion or a transition. It's used to highlight key items on a graphic.

The transition attribute allows you to change property values smoothly, over a given duration. A transition is always assigned to the initial state.
States of the element can be animated are hover:, target:, focus:, valid:, checked:, active:, disabled:, invalid:.

Further animation effects can be achieved with the following commands:

property values	
linear	Specifies a transition effect with the same speed from start to end.
ease	Default value. Specifies a transition effect with a slow start, then fast, then end slowly
ease-in	Specifies a transition effect with a slow start
ease-out	Specifies a transition effect with a slow end
ease-in-out	Specifies a transition effect with a slow start and end

Example SVG hover animation

Code before editing:

```
<svg width="186" height="186" xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" overflow="hidden"><defs><clipPath id="clip0">
<rect x="-322" y="146" width="186" height="186"/></clipPath></defs><g clip-path="url(#clip0)"
transform="translate(322 -146)"><path d="M-322 239.5C-322 188.414-280.586 147-229.5 147-178.414 147-
137 188.414-137 239.5-137 290.586-178.414 332-229.5 332-280.586 332-322 290.586-322 239.5Z"
fill="#ED7D31" fill-rule="evenodd"/><path d="M-253.5 246.5C-253.5 240.977-249.023 236.5-243.5 236.5-
237.977 236.5-233.5 240.977-233.5 246.5-233.5 252.023-237.977 256.5-243.5 256.5-249.023 256.5-253.5
252.023-253.5 246.5Z" stroke="#C55A11" stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11"
fill-rule="evenodd"/>
<path d="M-240.5 219C-240.5 212.649-235.575 207.5-229.5 207.5-223.425 207.5-218.5 212.649-218.5 219-
218.5 225.351-223.425 230.5-229.5 230.5-235.575 230.5-240.5 225.351-240.5 219Z" stroke="#C55A11"
stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/><path d="M-223.5
247C-223.5 241.753-219.471 237.5-214.5 237.5-209.529 237.5-205.5 241.753-205.5 247-205.5 252.247-
209.529 256.5-214.5 256.5-219.471 256.5-223.5 252.247-223.5 247Z" stroke="#C55A11" stroke-
width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
<path d="M-279.5 265.5C-279.5 260.53-275.247 256.5-270 256.5-264.753 256.5-260.5 260.53-260.5 265.5-
260.5 270.471-264.753 274.5-270 274.5-275.247 274.5-279.5 270.471-279.5 265.5Z" stroke="#C55A11"
stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/><path d="M-244.5
274.5C-244.5 266.768-238.456 260.5-231 260.5-223.544 260.5-217.5 266.768-217.5 274.5-217.5 282.232-
223.544 288.5-231 288.5-238.456 288.5-244.5 282.232-244.5 274.5Z" stroke="#C55A11" stroke-
width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
<path d="M-273.5 230.5C-273.5 226.634-270.366 223.5-266.5 223.5-262.634 223.5-259.5 226.634-259.5
230.5-259.5 234.366-262.634 237.5-266.5 237.5-270.366 237.5-273.5 234.366-273.5 230.5Z"
stroke="#C55A11" stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
<path d="M-267 230-244.568 246.357" stroke="#C55A11" stroke-width="2.4" stroke-miterlimit="8"
fill="#C55A11" fill-rule="evenodd"/><path d="M0 0 14.6992 28.4557" stroke="#C55A11" stroke-
width="2.4" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd" transform="matrix(-1 0 0 1
-230.301 218)"/>
<path d="M0 0 26.1623 19.1372" stroke="#C55A11" stroke-width="2.4" stroke-miterlimit="8"
fill="#C55A11" fill-rule="evenodd" transform="matrix(1 0 0 -1 -271 265.137)"/><path d="M-231.926
274.697-243 249" stroke="#C55A11" stroke-width="2.4" stroke-miterlimit="8" fill="#C55A11" fill-
rule="evenodd"/><path d="M-244 246-214.7 246.835" stroke="#C55A11" stroke-width="2.4" stroke-
miterlimit="8" fill="#C55A11" fill-rule="evenodd"/><text fill="#FFFFFF" font-
family="Ubuntu,Ubuntu_MSFontService,sans-serif" font-weight="700" font-size="43"
transform="translate(-292.741 257)">iPortal</text></g></svg>
```

The graphic was created in Powerpoint and saved as a SVG file, the code usally looks confusing, it is helpful to organize it by incorporating paragraphs between the diffrent elements.

First step

Add viewBox attribute and style element to the code.
Some programs generate the viewBox, be sure that relative width and height set to 100%.

```
<svg viewBox="0 0 186 186" width="100%" height="100%“>
  <style>

  </style>
...
</svg>
```

Second step

Adding id or class to elements make it easier to pick the exact element. Writing CSS style rules for hover animation - here with opacity and transition. Also some elements were group with the g element, this is also helpful if there are more than one hover events in one graphic!

Code after editing:

```
<svg width="100%" height="100%" viewBox="0 0 187 186" xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" overflow="hidden">
    <style>
      #innovator{
      transition: opacity 0.2s ease
      }
      #demo:hover #innovator{
      opacity:1
      }
    </style>
    <defs>
      <clipPath id="clip0"><rect x="189" y="228" width="187" height="186"/></clipPath>
    </defs>
    <g clip-path="url(#clip0)" transform="translate(-189 -228)">
      <g id="demo">
      <path id="circle" d="M190 320.5C190 269.414 231.414 228 282.5 228 333.586 228 375 269.414 375
320.5 375 371.586 333.586 413 282.5 413 231.414 413 190 371.586 190 320.5Z" fill="#ED7D31" fill-
rule="evenodd"/>
  <g id="innovator" opacity="0">
      <path d="M257.5 327.5C257.5 321.977 261.977 317.5 267.5 317.5 273.023 317.5 277.5 321.977
277.5 327.5 277.5 333.023 273.023 337.5 267.5 337.5 261.977 337.5 257.5 333.023 257.5 327.5Z"
stroke="#C55A11" stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
      <path d="M270.5 300.5C270.5 294.425 275.649 289.5 282 289.5 288.351 289.5 293.5 294.425 293.5
300.5 293.5 306.575 288.351 311.5 282 311.5 275.649 311.5 270.5 306.575 270.5 300.5Z"
stroke="#C55A11" stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
      <path d="M288.5 328.5C288.5 323.53 292.53 319.5 297.5 319.5 302.471 319.5 306.5 323.53 306.5
328.5 306.5 333.471 302.471 337.5 297.5 337.5 292.53 337.5 288.5 333.471 288.5 328.5Z"
stroke="#C55A11" stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
      <path d="M232.5 347C232.5 341.753 236.529 337.5 241.5 337.5 246.471 337.5 250.5 341.753 250.5
347 250.5 352.247 246.471 356.5 241.5 356.5 236.529 356.5 232.5 352.247 232.5 347Z" stroke="#C55A11"
stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
      <path d="M266.5 355.5C266.5 347.768 272.768 341.5 280.5 341.5 288.232 341.5 294.5 347.768
294.5 355.5 294.5 363.232 288.232 369.5 280.5 369.5 272.768 369.5 266.5 363.232 266.5 355.5Z"
stroke="#C55A11" stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
      <path d="M237.5 312C237.5 307.858 240.858 304.5 245 304.5 249.142 304.5 252.5 307.858 252.5
312 252.5 316.142 249.142 319.5 245 319.5 240.858 319.5 237.5 316.142 237.5 312Z" stroke="#C55A11"
stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
      <path d="M244 311 266.432 327.357" stroke="#C55A11" stroke-width="2.4" stroke-miterlimit="8"
fill="#C55A11" fill-rule="evenodd"/>
      <path d="M0 0 14.6992 28.4557" stroke="#C55A11" stroke-width="2.4" stroke-miterlimit="8"
fill="#C55A11" fill-rule="evenodd" transform="matrix(-1 -8.74228e-08 -8.74228e-08 1 281.699 299)"/>
      <path d="M0 0 26.1623 19.1372" stroke="#C55A11" stroke-width="2.4" stroke-miterlimit="8"
fill="#C55A11" fill-rule="evenodd" transform="matrix(1 0 0 -1 241 346.137)"/>
      <path d="M279.074 355.697 268 330" stroke="#C55A11" stroke-width="2.4" stroke-miterlimit="8"
fill="#C55A11" fill-rule="evenodd"/>
      <path d="M267 327 296.3 327.835" stroke="#C55A11" stroke-width="2.4" stroke-miterlimit="8"
fill="#C55A11" fill-rule="evenodd"/>
    </g>
  <a xlink:href="https://iportal-innovators.avato.net/" text-decoration="none" target="_blank">
      <text id="txt" font-family="Ubuntu,Ubuntu_MSFontService,sans-serif" font-weight="700" font-
size="43" transform="translate(218 339)" fill="#fff">iPortal</text>
  </a>
    </g>
```

```
    </g>
  </svg>
```

Create a CSS animation with @keyframes

First of all an suitable animation depends on the graphic, there are many ways to animate a SVG with CSS @keyframes. A keyframe animation is made up of other keyframes. Each keyframe shows an animation booth. For example you can transform sizes of elements (zoom in/out effect) but there are many more options like skewing or roatating objects.

The @keyframes Rule
When you specify CSS styles inside the @keyframes rule, the animation will gradually change from the current style to the new style at a certain time. To get an animation to work, you must bind the animation to an element.

Syntax
@keyframes "animation-name" {
[from | to | percantage]
[from | to | percantage] }

See more Information of keyframes animation here

Example animation with keyframes

Code before editing:

```
<svg width="186" height="186" xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" overflow="hidden"><defs><clipPath id="clip0">
<rect x="-322" y="146" width="186" height="186"/></clipPath></defs><g clip-path="url(#clip0)"
transform="translate(322 -146)"><path d="M-322 239.5C-322 188.414-280.586 147-229.5 147-178.414 147-
137 188.414-137 239.5-137 290.586-178.414 332-229.5 332-280.586 332-322 290.586-322 239.5Z"
fill="#ED7D31" fill-rule="evenodd"/><path d="M-253.5 246.5C-253.5 240.977-249.023 236.5-243.5 236.5-
237.977 236.5-233.5 240.977-233.5 246.5-233.5 252.023-237.977 256.5-243.5 256.5-249.023 256.5-253.5
252.023-253.5 246.5Z" stroke="#C55A11" stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11"
fill-rule="evenodd"/>
<path d="M-240.5 219C-240.5 212.649-235.575 207.5-229.5 207.5-223.425 207.5-218.5 212.649-218.5 219-
218.5 225.351-223.425 230.5-229.5 230.5-235.575 230.5-240.5 225.351-240.5 219Z" stroke="#C55A11"
stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/><path d="M-223.5
247C-223.5 241.753-219.471 237.5-214.5 237.5-209.529 237.5-205.5 241.753-205.5 247-205.5 252.247-
209.529 256.5-214.5 256.5-219.471 256.5-223.5 252.247-223.5 247Z" stroke="#C55A11" stroke-
width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
<path d="M-279.5 265.5C-279.5 260.53-275.247 256.5-270 256.5-264.753 256.5-260.5 260.53-260.5 265.5-
260.5 270.471-264.753 274.5-270 274.5-275.247 274.5-279.5 270.471-279.5 265.5Z" stroke="#C55A11"
stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/><path d="M-244.5
274.5C-244.5 266.768-238.456 260.5-231 260.5-223.544 260.5-217.5 266.768-217.5 274.5-217.5 282.232-
223.544 288.5-231 288.5-238.456 288.5-244.5 282.232-244.5 274.5Z" stroke="#C55A11" stroke-
width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
<path d="M-273.5 230.5C-273.5 226.634-270.366 223.5-266.5 223.5-262.634 223.5-259.5 226.634-259.5
230.5-259.5 234.366-262.634 237.5-266.5 237.5-270.366 237.5-273.5 234.366-273.5 230.5Z"
stroke="#C55A11" stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
<path d="M-267 230-244.568 246.357" stroke="#C55A11" stroke-width="2.4" stroke-miterlimit="8"
fill="#C55A11" fill-rule="evenodd"/><path d="M0 0 14.6992 28.4557" stroke="#C55A11" stroke-
width="2.4" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd" transform="matrix(-1 0 0 1
-230.301 218)"/>
<path d="M0 0 26.1623 19.1372" stroke="#C55A11" stroke-width="2.4" stroke-miterlimit="8"
fill="#C55A11" fill-rule="evenodd" transform="matrix(1 0 0 -1 -271 265.137)"/><path d="M-231.926
274.697-243 249" stroke="#C55A11" stroke-width="2.4" stroke-miterlimit="8" fill="#C55A11" fill-
rule="evenodd"/><path d="M-244 246-214.7 246.835" stroke="#C55A11" stroke-width="2.4" stroke-
miterlimit="8" fill="#C55A11" fill-rule="evenodd"/><text fill="#FFFFFF" font-
family="Ubuntu,Ubuntu_MSFontService,sans-serif" font-weight="700" font-size="43"
transform="translate(-292.741 257)">iPortal</text></g></svg>
```

The graphic was created in Powerpoint and saved as a SVG file, the code usally looks confusing, it is helpful to organize it by incorporating paragraphs between the diffrent elements.

First step

Add viewBox attribute and style element to the code.
Some programs generate the viewBox, be sure that relative width and height set to 100%.

```
<svg viewBox="0 0 186 186" width="100%" height="100%“>
<style>

</style>
...
</svg>
```

Second step

Adding id or class to elements make it easier to pick the exact element. Set the values for transform, name the animation and add it to the element. In the code also the elements were group with the g element.

code after editing:

```
<svg width="100%" height="100%" viewBox="0 0 187 186" xmlns="http://www.w3.org/2000/svg"
xmlns:xlink="http://www.w3.org/1999/xlink" overflow="hidden">
  <style>
    #innovator{
    animation: beat 3.5s forwards 0.3s infinite;
        }
    @-webkit-keyframes beat {
        0% { transform: scale(1);}
        50% { transform: scale(0.95);}
        }
  </style>
  <defs>
    <clipPath id="clip0"><rect x="189" y="228" width="187" height="186"/></clipPath>
  </defs>
  <g clip-path="url(#clip0)" transform="translate(-189 -228)">
    <g id="demo">
    <path id="circle" d="M190 320.5C190 269.414 231.414 228 282.5 228 333.586 228 375 269.414 375
320.5 375 371.586 333.586 413 282.5 413 231.414 413 190 371.586 190 320.5Z" fill="#ED7D31" fill-
rule="evenodd"/>
<g id="innovator" opacity="1">
    <path d="M257.5 327.5C257.5 321.977 261.977 317.5 267.5 317.5 273.023 317.5 277.5 321.977 277.5
327.5 277.5 333.023 273.023 337.5 267.5 337.5 261.977 337.5 257.5 333.023 257.5 327.5Z"
stroke="#C55A11" stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
    <path d="M270.5 300.5C270.5 294.425 275.649 289.5 282 289.5 288.351 289.5 293.5 294.425 293.5
300.5 293.5 306.575 288.351 311.5 282 311.5 275.649 311.5 270.5 306.575 270.5 300.5Z"
stroke="#C55A11" stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
    <path d="M288.5 328.5C288.5 323.53 292.53 319.5 297.5 319.5 302.471 319.5 306.5 323.53 306.5
328.5 306.5 333.471 302.471 337.5 297.5 337.5 292.53 337.5 288.5 333.471 288.5 328.5Z"
stroke="#C55A11" stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
    <path d="M232.5 347C232.5 341.753 236.529 337.5 241.5 337.5 246.471 337.5 250.5 341.753 250.5
347 250.5 352.247 246.471 356.5 241.5 356.5 236.529 356.5 232.5 352.247 232.5 347Z" stroke="#C55A11"
stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
    <path d="M266.5 355.5C266.5 347.768 272.768 341.5 280.5 341.5 288.232 341.5 294.5 347.768 294.5
355.5 294.5 363.232 288.232 369.5 280.5 369.5 272.768 369.5 266.5 363.232 266.5 355.5Z"
stroke="#C55A11" stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
    <path d="M237.5 312C237.5 307.858 240.858 304.5 245 304.5 249.142 304.5 252.5 307.858 252.5 312
252.5 316.142 249.142 319.5 245 319.5 240.858 319.5 237.5 316.142 237.5 312Z" stroke="#C55A11"
stroke-width="0.666667" stroke-miterlimit="8" fill="#C55A11" fill-rule="evenodd"/>
    <path d="M244 311 266.432 327.357" stroke="#C55A11" stroke-width="2.4" stroke-miterlimit="8"
fill="#C55A11" fill-rule="evenodd"/>
    <path d="M0 0 14.6992 28.4557" stroke="#C55A11" stroke-width="2.4" stroke-miterlimit="8"
fill="#C55A11" fill-rule="evenodd" transform="matrix(-1 -8.74228e-08 -8.74228e-08 1 281.699 299)"/>
    <path d="M0 0 26.1623 19.1372" stroke="#C55A11" stroke-width="2.4" stroke-miterlimit="8"
fill="#C55A11" fill-rule="evenodd" transform="matrix(1 0 0 -1 241 346.137)"/>
    <path d="M279.074 355.697 268 330" stroke="#C55A11" stroke-width="2.4" stroke-miterlimit="8"
fill="#C55A11" fill-rule="evenodd"/>
    <path d="M267 327 296.3 327.835" stroke="#C55A11" stroke-width="2.4" stroke-miterlimit="8"
fill="#C55A11" fill-rule="evenodd"/>
</g>
  <a xlink:href="https://iportal-innovators.avato.net/" text-decoration="none" target="_blank">
    <text id="txt" font-family="Ubuntu,Ubuntu_MSFontService,sans-serif" font-weight="700" font-
size="43" transform="translate(218 339)" fill="#fff">iPortal</text>
</a>
  </g>
```

```
</g>
</svg>
```

SVG Font

This is not necessary if the SVG is later embed with an img tag in the html file. Only relevante if the object tag is used for embed the SVG file!

There are two options to show the right font in a SVG image.

- Create a path
- Embed font-face into the SVG code

Create a path

The progam you use to create a graphic often have a function to convert the text in path. Using this method makes the SVG code longer and every letter is a seperate path, which makes it difficult to animate and change text later in the code.

Embed font-face into SVG code

The Advantage of using this method is that it is working on already animate SVG, so you do not have to write the animation again. Therefore it is easy in use. It works with base64 code. Generate the font you want at squirrle generator with following options.

- Click **Expert** after upload your font at the generator section of squirrle
- Be sure that **font formats** are woff and woff2
- At **CSS** section choose "Base64 Encode"
- **Download** the Kit and open the file named "stylesheet"
- **Copy** the code and **paste** into your SVG file
- In additon **add** following CSS rules into your SVG:

```
<style>
  text{
  font-family:'MYFONT';
  }
    @media screen and (-webkit-min-device-pixel-ratio:0) {
        @font-face {
         font-family: 'MYFONT';
         src: url('MYFONT.svg#MYFONT') format('svg');
        }
        ...
        </style>
```

For Innovators we use the font-family "Ubuntu" the whole CSS Code for the SVG file (included base64 code) is:

```
 <style>
  text{
  font-family:'ubunturegular';
  }

    @media screen and (-webkit-min-device-pixel-ratio:0) {
        @font-face {
         font-family: 'ubunturegular';
         src: url('ubunturegular.svg#ubunturegular') format('svg');
        }
    }

    @font-face {
      font-family: 'ubunturegular';
      src: url(data:application/font-woff2;charset=utf-
8;base64,d09GMgABAAAAAG+4ABIAAAABt/wAAG9QAAEAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAP0ZGVE0cGjAbg/lqHIVoBmAAg2oIRg
 format('woff2'),
           url(data:application/font-woff;charset=utf-
8;base64,d09GRgABAAAAAI9EABIAAAABt/wAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABGRlRNAAABlAAAABwAAAAce2/eBEdERUYAAA
 format('woff');
      font-weight: normal;
      font-style: normal;

  }
  ...
  </style>
```

Link elements/text

- Adding a link to a shape or text with an element
- Like in html you write the element around the element/s you want to link

Use the a element as following example:

```
<a href="https://iportal-innovators.avato.net" target="_parent" text-decoration="none">
<text>Innovators<text> <a>
```

Values for target:

- _self (default value): same tab or page
- _blank: new tab or window
- _parent: parent window

Embed in html

Embed your SVG with the <object> element.

```
<object data="" type="image/svg+xml" width="">
</object>
```

⚠️ Tips

- After you edit the code, prove the graphic before publishing in innovators!
  Control your work by open it up in your Browser window. If its full shown and the animaiton works, everything is fine and ready for release!
- It is not possible to link the object element in HTML, the link need to be in the SVG Code to work.
- Browser support: SVG is supported by the most newest common browsers like Google Chrome, Firefox, Microsoft Edge, Opera, Safari etc.
- **Note hover Animation:** It is useful to group your elements with the <g> element in case you want to animate a hover event that is triggered not by the element itself rather by a second extra element.

Styleguide

For an consitance use of SVG graphics follow the Innovators Illustration Guide!

Appendix: Popup Content

Comments are only visibile to logged in users.