Generate SVG images from Oracle Database

Zoran Tica, 18.10.2023

Agenda

- Brief History and Introduction to SVG
- SVG structure and organization
- Basic building blocks… or how to draw an image
- *“The path of the righteous man…” (Ezekiel 25:17)*
- Write “Hello world”… just like in 80-ties
- Transformers at work
- Link it up with URLs
- Grouping, Re-use
- SVG and CSS hand in hand
- Gradients, Patterns, Filters, Masks
- Integration with Javascript… if needed

Generate SVG images from Oracle Database with live demo

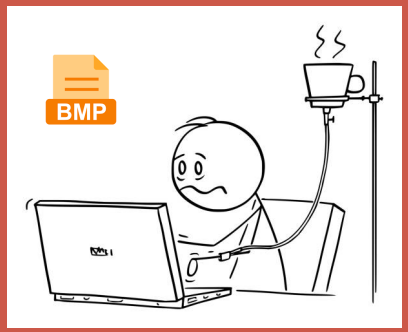
Zoran Tica…

- Slovenia
- Mid 80-ties - commodore 64
- Mid 90-ties - Business Analysis, Database Modeling, Software Development
- From 2000+ Oracle Technologies
- Principal Database Design & APEX Consultant at https://right-thing.solutions
- Speaker at various regional conferences
- OCP Advanced PL/SQL, Forms and Java SE developer
- Oracle ACE Associate
- https://www.linkedin.com/in/zoran-tica
- https://github.com/zorantica

@zoran_tica

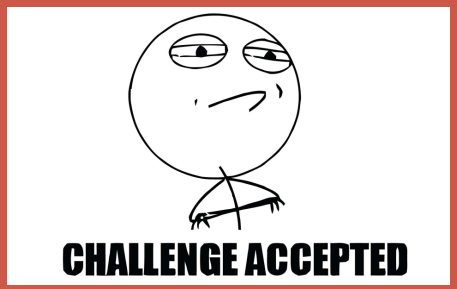
zoran.tica@right-thing.solutions

Before the beginning

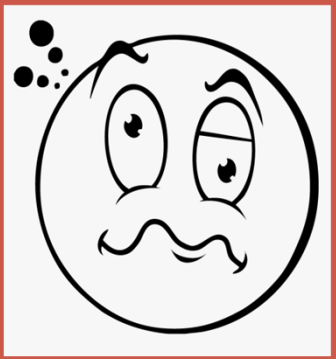


2018

2019



Before the beginning



2019-2023

```
CREATE OR REPLACE PACKAGE ZT_SVG AS
/*****************************************************************************
    Author:     Zoran Tica
                The Right Thing Solutions
                https://right-thing.solutions/

    PURPOSE:    A package for Scalable Vector Graphics (SVG) images generation

    REVISIONS:
    Ver        Date        Author           Description
    ---------  ----------  ---------------  ------------------------------------
    1.0        27/01/2023  Zoran Tica       First version of package.

    ---------------------------------------------------------------------------
    Copyright (C) 2023 - Zoran Tica

    Permission is hereby granted, free of charge, to any person obtaining a copy
    of this software and associated documentation files (the "Software"), to deal
    in the Software without restriction, including without limitation the rights
    to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
    copies of the Software, and to permit persons to whom the Software is
    furnished to do so, subject to the following conditions:

```

Introduction to SVG

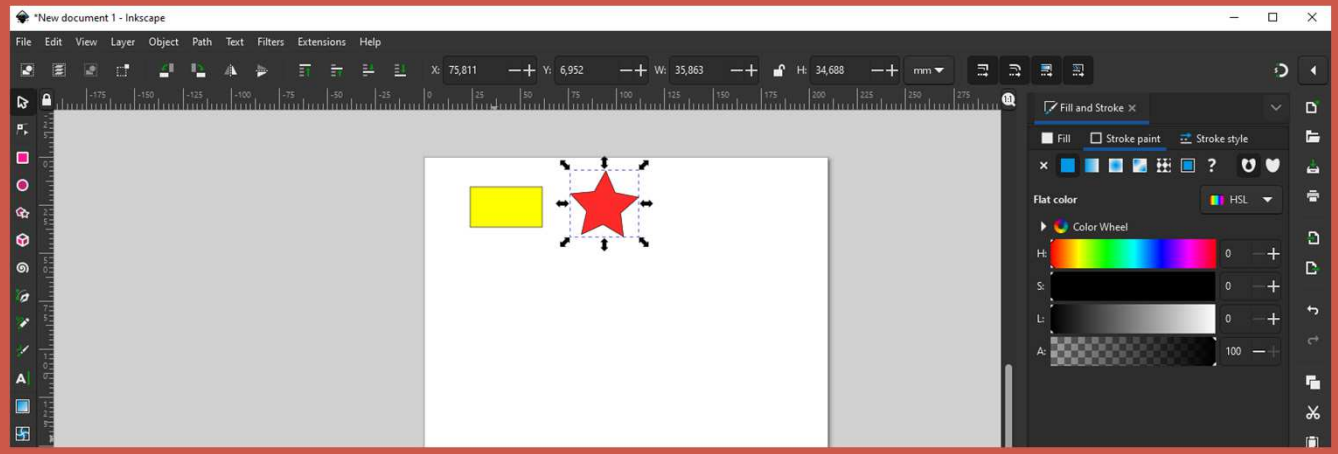
- Scalable Vector Graphics (SVG)
- Open standard developed by the World Wide Web Consortium (W3C) since 1999
- XML Markup language
- Selected among 6 competing vector graphics submissions
- Targeted for HTML and internet era

- Integration with HTML, CSS and Javascript

- Various browsers support and compatibility

Introduction to SVG

- A really popular image format
- Icons
- A LOT OF free images on internet
- Supported by various graphic editing software
- Inkscape
  https://inkscape.org/



Structure and Organization

- XML Markup language similar to HTML

- Main <svg> tag with attributes
- Image Height and Width
- Pixels and percentages
- Scalar values and lists
- ViewBox
- Other attributes like preserveAspectRatio, X, Y, Version…

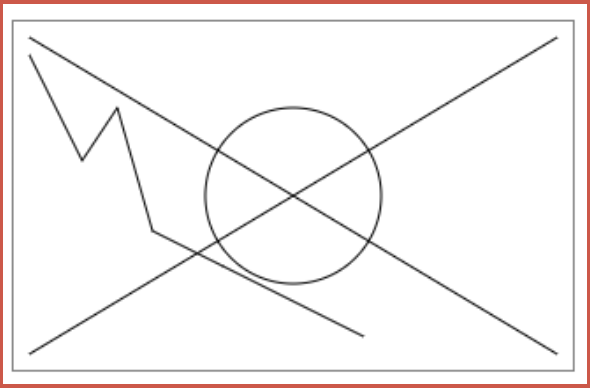
```
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 200 100" width="50%" height="200">

</svg>
```

- <defs> section, used for definitions like CSS classes
- Drawing elements section

Basic Drawing Elements

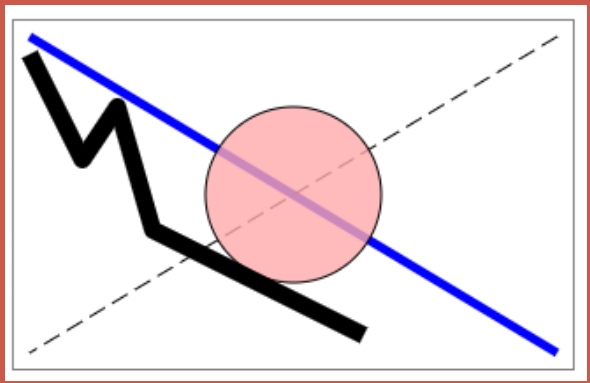
- Called “Shapes”
  - Line
  - Rectangle
  - Circle
  - Ellipse
  - Polyline
  - Polygon
- Attributes contain coordinates, colors…

```
<svg xmlns="http://www.w3.org/2000/svg" width="320" height="200" stroke="black" fill="none">
    <rect width="100%" height="100%"/>
    <line x1="10" y1="10" x2="310" y2="190"/>
    <line x1="310" y1="10" x2="10" y2="190"/>
    <circle cx="50%" cy="50%" r="50"/>
    <polyline points="10,20 40,80 60,50 80,120 120,140 200,180"/>
</svg>
```

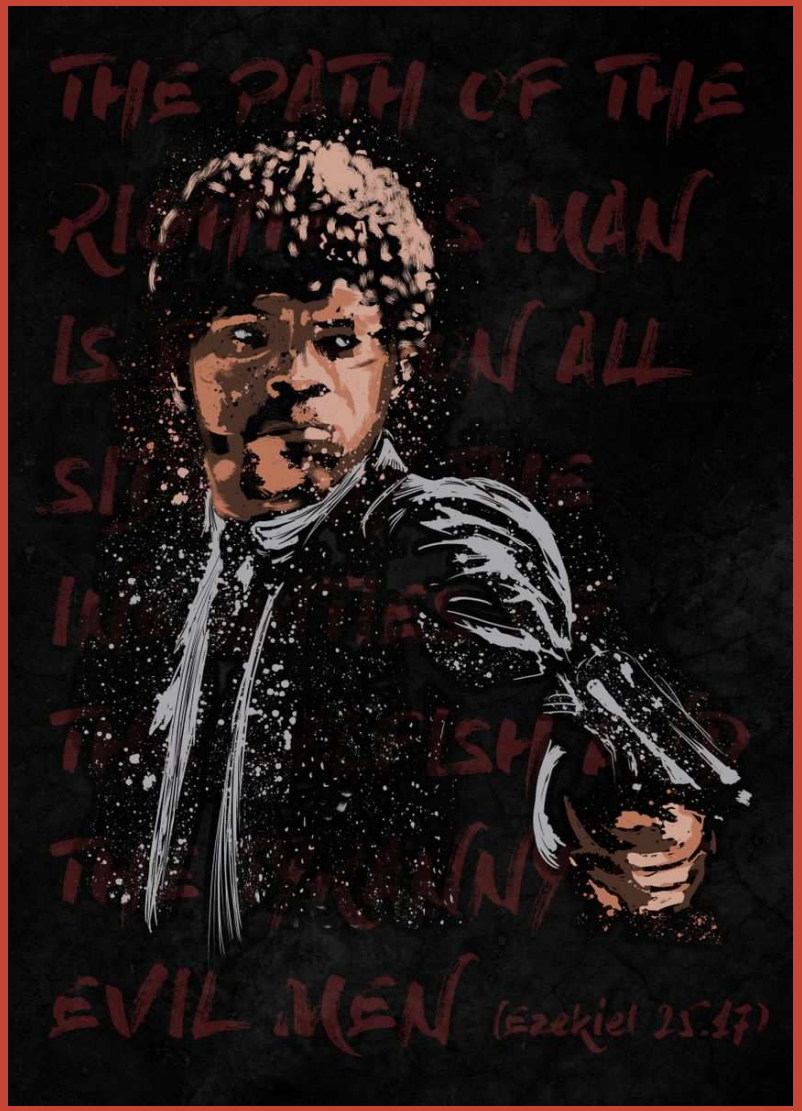


Drawing Elements –Stroke and Fill

- Color (named and RGB) and Transparency
- Stroke Width (thickness)
- Stroke Line Cap (line ending shape)
- Stroke Line Join (corner of 2 lines shape)
- Stroke Dash Array (dashed lines)
- Basic Fill



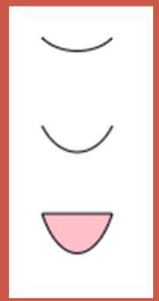
```
<svg xmlns="http://www.w3.org/2000/svg" width="320" height="200" stroke="black" fill="none">
    <rect width="100%" height="100%"/>
    <line x1="10" y1="10" x2="310" y2="190" stroke="blue" stroke-width="5"/>
    <line x1="310" y1="10" x2="10" y2="190" stroke-dasharray="10,5"/>
    <circle cx="50%" cy="50%" r="50" fill="#FFAAAA" fill-opacity="0.8"/>
    <polyline points="10,20 40,80 60,50 80,120 120,140 200,180" stroke-width="10" stroke-linejoin="round"/>
</svg>
```


THE PATH OF THE
MAN
ALL
EVIL MEN (Ezekiel 25:17)

Complex Shape - Path

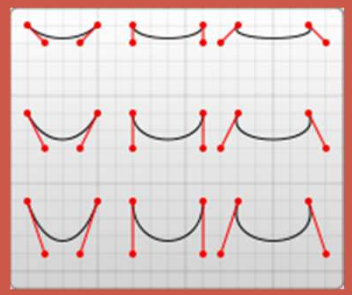
- Constructed from segments of type
  - Line
  - Bézier Curve
  - Arc
- Can draw practically anything
- Really complex to use
- Requires a list of commands, coordinates and segment parameters

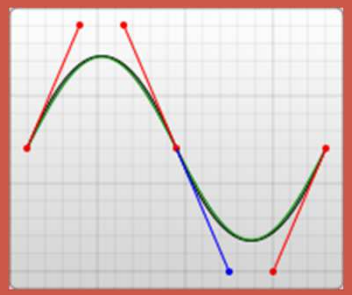
```
<svg width="190" height="160" xmlns="http://www.w3.org/2000/svg" stroke="black" fill="transparent">
    <path d="M 10 10 C 20 20, 40 20, 50 10" />
    <path d="M 10 60 C 20 80, 40 80, 50 60" />
    <path d="M 10 110 C 20 140, 40 140, 50 110 Z" fill="pink" />
</svg>
```

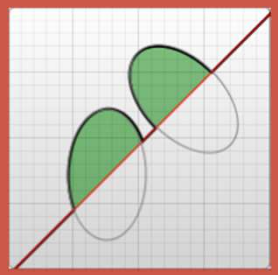


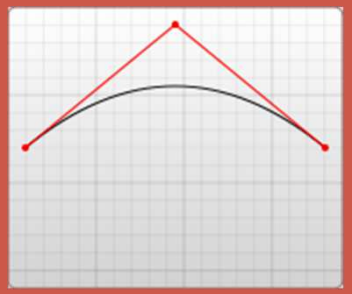
Complex Shape – Path Commands

- M - starting point of path (M x y)
- L – straight line segment (L x y)
- H – horizontal straight line (H x)
- V - vertical straight line (V x)
- Z – close the path at the end
- C – cubic Bézier Curve (C x1 y1, x2 y2, x y)
- S – continuous cubic Bézier Curve (S x2 y2, x y)
- Q – quadratic Bézier Curve (Q x1 y1, x y)
- T – continuous quadratic Bézier Curve (T x y)
- A – arc (rx ry x-axis-rotation large-arc-flag sweep-flag x y)
- X and Y vs DX and DY









Write a Text

- Basic <text> tag
- Subtext (<tspan> tag)
- Absolute text position with “x” and “y” attributes
- Relative subtext position with “dx” and “dy” attributes
- Fonts (style, size, spacing…)

Several lines: *First line.*

Second line.

```
<svg xmlns="http://www.w3.org/2000/svg" width="320" height="200" stroke="none" fill="red">
    <text x="10" y="50">Several lines:
    <tspan dx="0" dy="10" font-style="italic" font-family="Comic Sans MS" font-size="24">First line.</tspan>
        <tspan x="10" y="100" font-size="18" font-weight="bold">Second line.</tspan>
    </text>
</svg>
```

Transformations

- “transform” attribute
- Translation – translate(x,y)
- Rotation – rotate (angle centerx,centery)
- Skewing – skewX(angle) and skewY(angle)
- Scaling – scale(scalex, scaley)
- Possible to combine transformations

- Matrix

```
<svg viewBox="-5 -5 50 50" xmlns="http://www.w3.org/2000/svg">
    <rect x="-3" y="-3" width="6" height="6" />
    <rect x="-3" y="-3" width="6" height="6" fill="red" transform="skewX(-30) translate(10)" />
</svg>
```

URLs

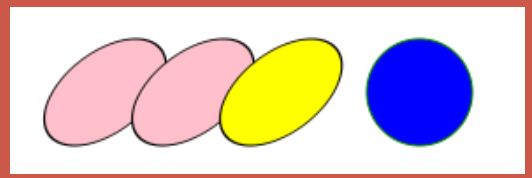
- Ordinary “a” tag
- Elements with URL are positioned within “a” tag

```
<svg viewBox="-5 -5 50 50" xmlns="http://www.w3.org/2000/svg">
    <a href="https://right-thing.solutions/">
        <rect x="-3" y="-3" width="6" height="6" />
        <rect x="-3" y="-3" width="6" height="6" fill="red" transform="skewX(-30) translate(10)" />
    </a>
</svg>
```

Grouping

- <g> … </g> tag
- Elements are located within tag
- Inherit attributes from group (stroke, fill, transformations…)

```svg
<svg width="100%" xmlns="http://www.w3.org/2000/svg" fill="blue" stroke="green">
    <g fill="pink" stroke="black" transform="skewX(-30)">
        <circle cx="150" cy="50" r="30"/>
        <circle cx="200" cy="50" r="30"/>
        <circle cx="250" cy="50" r="30" fill="yellow"/>
    </g>
    <circle cx="300" cy="50" r="30"/>
</svg>
```

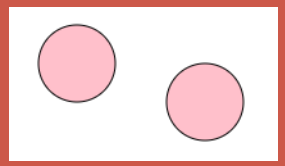


Re-use Elements and Objects

- “image” tag – inserts JPG, PNG or SVG image into current SVG image
- “use” tag – duplicates and displays a part of SVG image
- “object” tag – inserts object such as PDF document into image

```
<svg width="100%" xmlns="http://www.w3.org/2000/svg">
    <image
        href="https://right-thing.solutions/ords/r/app/170/files/static/v197/TRT_logo.svg"
        width="400"
    />
</svg>
```

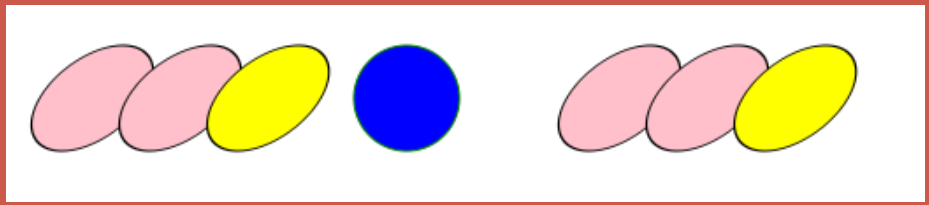

```
<svg width="100%" xmlns="http://www.w3.org/2000/svg">
    <circle cx="50" cy="50" r="30" fill="pink" stroke="black" id="myCircle" />
    <use href="#myCircle" x="100" y="30"/>
</svg>
```



Re-use Elements and Objects

- Re-use complete group by ID

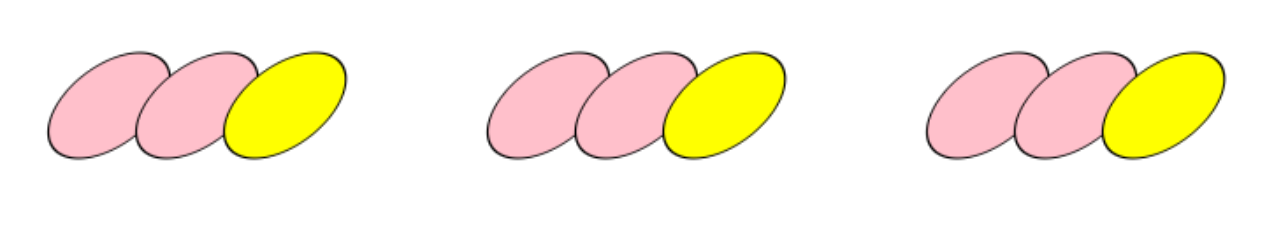
```svg
<svg width="100%" xmlns="http://www.w3.org/2000/svg" fill="blue" stroke="green">
    <g fill="pink" stroke="black" transform="skewX(-30)" id="myGroup">
        <circle cx="150" cy="50" r="30"/>
        <circle cx="200" cy="50" r="30"/>
        <circle cx="250" cy="50" r="30" fill="yellow"/>
    </g>
    <circle cx="300" cy="50" r="30"/>
    <use href="#myGroup" x="300" y="0">
</svg>
```



Re-use Elements and Objects

- Re-use elements or groups from <defs> section

```svg
<svg width="100%" xmlns="http://www.w3.org/2000/svg">
    <defs>
        <g fill="pink" stroke="black" transform="skewX(-30)" id="myGroup">
            <circle cx="150" cy="50" r="30"/>
            <circle cx="200" cy="50" r="30"/>
            <circle cx="250" cy="50" r="30" fill="yellow"/>
        </g>
    </defs>
    <use href="#myGroup" x="0" y="0"/>
    <use href="#myGroup" x="250" y="0"/>
    <use href="#myGroup" x="500" y="0"/>
</svg>
```



Style and CSS classes

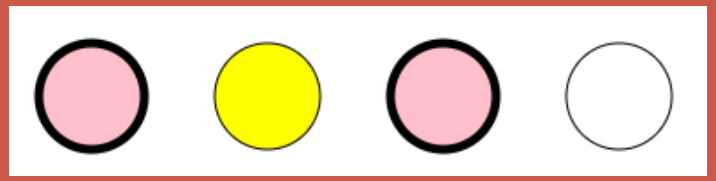
- SVG-specific styles are used
- “style” element attribute, similar to HTML

```
<svg width="100%" xmlns="http://www.w3.org/2000/svg">
    <circle
        cx="100"
        cy="50"
        r="30"
        style="fill:pink; stroke:black; stroke-width:5px;" />
    <text
        x="200"
        y="50"
        style="fill:red; font-size:24px; font-style:italic">Colored text</text>
</svg>
```

Style and CSS classes

- CSS classes are defined in <defs> section with “style” tag
- Used in elements with “class” attribute

```svg
<svg width="100%" xmlns="http://www.w3.org/2000/svg">
    <defs>
        <style>
            .pinkCircle { fill:pink; stroke:black; stroke-width:5px; }
            .yellowCircle { fill:yellow; stroke:black; }
        </style>
    </defs>
    <circle cx="100" cy="50" r="30" class="pinkCircle"/>
    <circle cx="200" cy="50" r="30" class="yellowCircle"/>
    <circle cx="300" cy="50" r="30" class="pinkCircle"/>
    <circle cx="400" cy="50" r="30" fill="none" stroke="black"/>
</svg>
```

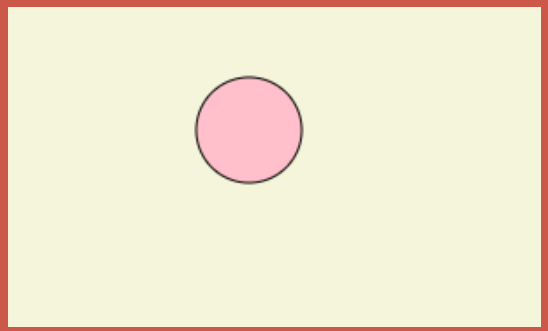


```svg
<svg
  width="600px"
  height="600px"
  viewBox="-300 -300 600 600"
  xmlns="http://www.w3.org/2000/svg"
  xmlns:xlink="http://www.w3.org/1999/xlink">
  <link rel="stylesheet" href="style8.css" type="text/css" />
```

- Can be also linked with external css file (“link” tag)

Style and CSS classes

- CSS classes outside of SVG

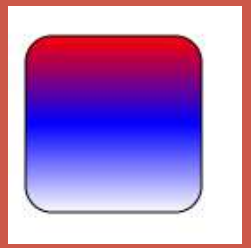


```html
<!DOCTYPE html>
<html>
    <head>
        <title>This is SVG example</title>
        <style>
            /*** SVG demonstration ***/

            /* page */
            svg {
                background-color: beige;
            }
            circle {
                fill: pink;
                stroke: black;
            }</style>
    </head>
    <body>
        <svg width="600px" height="600px" viewBox="-300 -300 600 600"
            xmlns="http://www.w3.org/2000/svg">
            <circle x="100" y="10" r="30"/>
        </svg>
    </body>
</html>
```

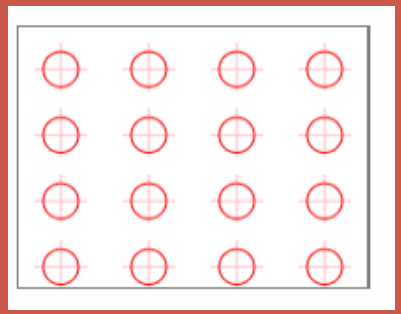
Gradients, Patterns

- Gradients – defined in <defs> section; fill color change; linear or circle
- Patterns – fill element by repeating another SVG elements

```svg
<svg width="100%" xmlns="http://www.w3.org/2000/svg">
    <defs>
    <linearGradient id="myGrad" x1="0" y1="0" x2="0" y2="100%">
        <stop offset="0%" style="stop-color:rgb(255,0,0); stop-opacity:1" />
        <stop offset="50%" style="stop-color:rgb(0,0,255); stop-opacity:1" />
        <stop offset="100%" style="stop-color:#FFFFFF; stop-opacity:1" />
    </linearGradient>
    </defs>
    <rect x="10" y="10" rx="15" ry="15" width="100" height="100" fill="url(#myGrad)" stroke="black"/>
</svg>
```



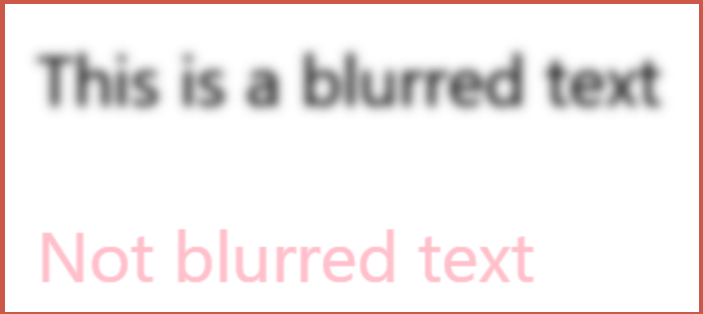
```svg
<svg width="100%" xmlns="http://www.w3.org/2000/svg">
    <defs>
        <pattern id="myPattern" x="0" y="0" width="25%" height="25%">
            <line x1="25" y1="10" x2="25" y2="40" stroke="pink"/>
            <line x1="10" y1="25" x2="40" y2="25" stroke="pink"/>
            <circle cx="25" cy="25" r="10" stroke="red" fill="none"/>
        </pattern>
    </defs>
    <rect fill="url(#myPattern)" stroke="black" width="200" height="150"/>
</svg>
```



Filters

- Filters – used for graphic effects like blur, lightning…
- Effects combining

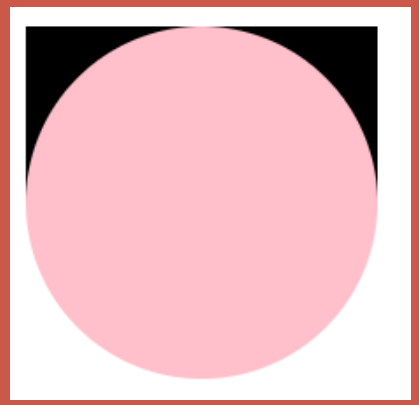
```svg
<svg width="100%" height="300" xmlns="http://www.w3.org/2000/svg" fill="pink">
    <defs>
        <filter id="myFilter" filterUnits="userSpaceOnUse" x="0" y="0" width="500" height="500">
            <feGaussianBlur in="SourceAlpha" stdDeviation="3" result="blur"/>
        </filter>
    </defs>
    <text filter="url(#myFilter)" x="10" y="50" font-size="40">This is a blurred text</text>
    <text x="10" y="150" font-size="40">Not blurred text</text>
</svg>
```



Clipping

- Clipping – removing element parts defined by other element

```svg
<svg width="100%" height="300">
    <rect x="0" y="0" width="200" height="100"/>
    <circle cx="100" cy="100" r="100" fill="pink"/>
</svg>
```



```svg
<svg width="100%" height="300">
    <defs>
        <clipPath id="myClip">
            <rect x="0" y="0" width="200" height="100"/>
        </clipPath>
    </defs>
    <circle cx="100" cy="100" r="100" fill="pink" clip-path="url(#myClip)"/>
</svg>
```

Masking

- Masking – complex clipping, can be combined with transparency

```svg
<svg viewBox="-10 -10 640 480">
    <rect x="0" y="0" width="100" height="100" fill="white" stroke="black"/>
    <path d="M10,35 A20,20,0,0,1,50,35 A20,20,0,0,1,90,35 Q90,65,50,95 Q10,65,10,35 Z" fill="black"/>
</svg>
```

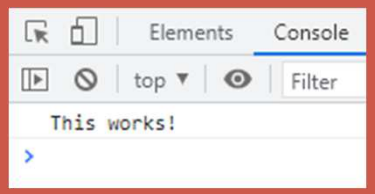
```svg
<svg viewBox="-10 -10 640 480">
    <mask id="myMask">
        <rect x="0" y="0" width="100" height="100" fill="white"/>
        <path d="M10,35 A20,20,0,0,1,50,35 A20,20,0,0,1,90,35 Q90,65,50,95 Q10,65,10,35 Z" fill="black"/>
    </mask>
    <polygon points="-10,110 110,110 110,-10" fill="orange"/>
    <circle cx="50" cy="50" r="50" fill="pink" mask="url(#myMask)"/>
</svg>
```

SVG and Javascript

- onClick and similar events

```svg
<svg xmlns="http://www.w3.org/2000/svg" width="700" height="600">
    <defs>
        <style type="text/css">
            <![CDATA[
                .oneCircle {
                    fill:pink;
                }
                .oneCircle:hover {
                    fill-opacity:0.5; cursor:pointer;
                }
            ]]></style>
    </defs>
    <circle cx="100" cy="100" r="50" class="oneCircle" onclick="myFunction()"/>
</svg>
```

```javascript
function myFunction() {
    console.log("This works!");
}
```



- Use jQuery to reference APEX dynamic actions

PL/SQL API – live demo

- ZT_SVG database package

```
--image handling
PROCEDURE p_new_image (
    p_image_reference varchar2 default zt_svg.gcDefaultIndex,
    p_viewbox_X number default null,
    p_viewbox_Y number default null,
    p_viewbox_width number default null,
    p_viewbox_height number default null,
    p_image_width varchar2 default null,  --number or percentage
    p_image_height varchar2 default null  --number or percentage
);
```

```
PROCEDURE p_draw_text (
    p_image_reference varchar2 default zt_svg.gcDefaultIndex,
    p_supertext_ref pls_integer default null,
    p_id varchar2 default null,
    p_x number default null,
    p_y number default null,
    p_dx number default null,
    p_dy number default null,
    p_text varchar2,
    p_font_ref varchar2 default null,
    p_font r_font default grDefaultFont,
    p_fill zt_svg.r_fill default zt_svg.grDefaultFill,
    p_stroke_ref varchar2 default null,
    p_stroke r_stroke default zt_svg.grDefaultStroke,
    p_style varchar2 default null,
    p_class_name varchar2 default null,
    p_url r_url default null,
    p_transform r_transform default null
);
```

```
FUNCTION f_draw_circle (
    p_image_reference varchar2 default zt_svg.gcDefaultIndex,
    p_id varchar2 default null,
    p_center_x number,
    p_center_y number,
    p_radius number,
    p_fill zt_svg.r_fill default zt_svg.grDefaultFill,
    p_stroke_ref varchar2 default null,
    p_stroke r_stroke default zt_svg.grDefaultStroke,
    p_style varchar2 default null,
    p_class_name varchar2 default null,
    p_url r_url default null,
    p_transform r_transform default null,
    p_custom_attributes varchar2 default null
) RETURN pls_integer;
```

```
--finish image and return HTML
FUNCTION f_finish_image (
    p_image_reference varchar2 default zt_svg.gcDefaultIndex
) RETURN clob;
```

LIVE DEMO

Conclusion

Conclusion

- What is SVG
- Brief history
- Organization and structure
- Basic and complex drawing elements
- Integration with CSS, HTML and Javascript

- PL/SQL API for generating images from Oracle Database
- Live demo

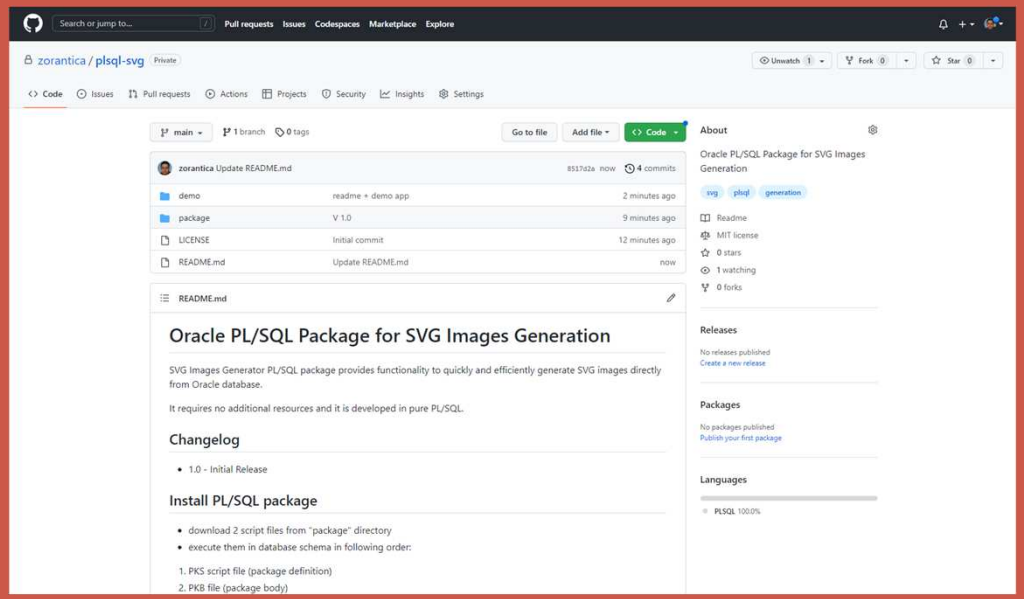
Conclusion

Useful scenarios:

- Different clients connect to same Oracle database
- Only PL/SQL developers in team
- Easy integration with APEX
- Customer doesn’t allow third party libraries on client side
- Restrictions on external services usage
- Generate SVG images during data processing

Easy to install and use on every project with Oracle database. Just install one PL/SQL package and You’re good to go.

A Gift for Community



https://github.com/zorantica/plsql-svg

Questions?


THE END
Thank you for listening!

References

- https://developer.mozilla.org/en-US/docs/Web/SVG
- https://dev.w3.org/SVG/tools/svgweb/samples/svg-files/
- https://docs.aspose.com/svg/net/drawing-basics/svg-transformations
- https://www.w3.org/Graphics/SVG/IG/resources/svgprimer.html

- https://inkscape.org/

- https://en.wikipedia.org/wiki/SVG