# Computer Programming – CS 6011
# Lecture 13: JavaScript Drawing

Fall 2023

## Topics

- Drawing in JavaScript

# Drawing with JavaScript

- There are 2 approaches
  - Canvas
  - **S**calable **V**ector **G**raphics (SVG)

# Canvas vs. SVG

- Canvas
  - Represented in DOM as an <img> element (Single HTML element).
  - Does NOT preserve clarity when zooming or scaling.
  - Erase the canvas and redraw to animate
  - Need a `<canvas width="1000" height="500" />` tag in your HTML.
- SVG (Scalable Vector Graphics)
  - Becomes part of the DOM tree (SVG elements are similar to HTML elements).
  - Preserve clarity when zooming or scaling.
  - Add elements to the SVG using the DOM API and manipulate their attributes (to move them, etc). The browser will automatically re-render the SVG (and its children).

## Canvas Animation Pseudocode

```
function handleMouseMoveCB() {
        // Store location of the "mouse" object.
}
function draw() {
        // clear background rectangle
        // loop over enemy objects
                // draw object
                // update object (position, etc)
        // draw "mouse" object (Where is "mouse" object updated?)
        // request another animation frame
}
// Request initial animation frame (using draw function).
```

## Canvas Example – In Class

- Add a <canvas> tag in your HTML...
- let canvas = document.getElementsByTagName( 'canvas' )[ 0 ];
- let ctx = canvas.getContext( '2d' ); // Get an object with 2d drawing methods.
- let winWidth  = window.innerWidth;
- let winHeight = window.innerHeight;
- canvas.width = winWidth;
- canvas.height = winHeight;
- ctx.clearRect( 0, 0, winWidth, winHeight ); // Erase whatever is there.
- ctx.fillRect( 10, 10, 1000, 1000 ); // Draw a 1000x1000 rectangle at 10, 10.
- let myImg = new Image();
- myImg.src = "msd_logo.jpg";
- **myImg.onload = function() { ctx.drawImage( myImg, 20, 20 ); }** // Must wait until the image is loaded to draw it...

## Drawing with SVG

- SVGs are more powerful, but also more cumbersome
- SVG is its own XML format.
  - [XML stands for: eXtensible Markup Language]
- Instead of assigning an attribute directly like we do with normal DOM elements
  - myNode.width = 100; // Normal DOM element
  - SVG uses: `setAttribute()` and `getAttribute()`
- In some places we will have to specify the XML Namespace to indicate that we are dealing with SVG elements.

# SVG Example

```
<svg id="svg" width="1000" height="500">
    <circle cx="50" cy="50" r="40" stroke="green" stroke-width="4" fill="yellow" />
    <rect x="200" y="20" width="400" height="100" style="fill:red; stroke-width:10; stroke:white" />
    <rect x="10" y="150" rx="20" ry="20" width="150" height="150"
         style="fill:red;stroke:black;stroke-width:5;opacity:0.5" />  //alternative fill:rgb(0,0,255)
    <polygon points="100,10 40,198 190,78 10,78 160,198"
           style="fill:lime;stroke:purple;stroke-width:5;fill-rule:evenodd;"
           transform="translate(200,200)" />
  </svg>
```

## SVG JavaScript

```
let svgNS = "http://www.w3.org/2000/svg";
let myImg = document.createElementNS( svgNS, "image" ); // Make an SVG image node.
myImg.setAttributeNS( null, "href", "msd_logo.jpg" );
myImg.setAttributeNS( null, "transform", "translate( 650, 100 )" );
let svg = document.getElementById( "svg" );
svg.appendChild( myImg );
let circle = document.createElementNS( svgNS, "circle" );
circle.setAttribute( "r", "20" );
circle.setAttribute( "cx", "800" );
circle.setAttribute( "cy", "50" );
svg.appendChild( circle );
```

## Animating

- Use the *window.requestAnimationFrame()* method to pass a callback that will be executed when the browser decides it is time to redraw the screen (usually 60 times / second).
- For a canvas based drawing, the callback will contain lots of drawing code.
- For the SVG based drawing, the callback might only have updates to element attributes.
- To track the mouse, we add a listener for mousemove events to the document and then remember those positions ourself.
- The canvas has offsetLeft and offsetTop attributes that help us convert to canvas coordinates (if the canvas is not in the upper left corner of the window).
- SVG has a method getClientBoundingRect() that returns information about the SVG's position in the screen and its size.

# Thoughts on today's assignment…

- Bees Game
    - How to store a bee?
        - As an object!
    - How to store all the bees?
        - As an array (of objects).
    - What does a bee object look like?
        - position, image*
    - How to create one?
        - let bee = {};
        - bee.img = new Image();…
        - bee.pos = {};
        - bee.pos.x = 0;
        - bee.pos.y = 0;

## Wednesday Assignments

- Code Review – Synthesizer Final
- Time to check your submissions and your grades
- HW 6 – Not The Bees!