# Unit 3 - Multimedia Animation

## HTMLS - CANVAS:

HTML5Canvasisapowerfulfeaturethatallowsdeveloperstodynamicallycreate and manipulate graphics and animations using JavaScript.

## The Rendering Context

The rendering context in Canvas is essentially the interface through which JavaScript code can interact with the Canvas element to create and manipulate graphicalcontent.Thecontextprovidesasetofdrawingfunctionsthatcanbeused to create lines, shapes, text, images, and animations on the Canvas element.

There are two types of rendering contexts in Canvas: 2D and WebGL. The 2D context is the default context and provides a powerful and flexible API for creating 2Dgraphicsandanimations.TheWebGLcontext,ontheotherhand,allowsforthe creation of 3D graphics using the WebGL API.

## Browser Support

In terms of browser support, HTML5 Canvas is supported in all modern browsers, including Chrome, Firefox, Safari, and Edge. However, the level of support for different features may vary between browsers. To ensure compatibility across different browsers, it is important to test your code thoroughly and use feature detection and fallbacks as necessary. Additionally, some older browsers may not supportCanvasatall,soitisimportanttoprovidealternativecontentorfunctionality for users on those platforms.

## HTML 5 - Canvas Examples

Here are a few examples of what can be done with HTML5 Canvas:

**1. Drawingshapesandlines:**TheHTML5Canvaselementcanbeusedtodraw various shapes and lines on a web page. Here's an example of a red rectangle:

```
<canvas id="myCanvas" width="200"height="200"></canvas>

<script>
  varcanvas=document.getElementById("myCanvas"); var
  context = canvas.getContext("2d");
  context.fillStyle = "red";
  context.fillRect(10,10,50,50);
```

```
</script>
```

**2. Creatinganimations:**TheHTML5Canvaselementcanalsobeusedtocreate animations on a web page. Here's an example of a bouncing ball animation:

```
<canvas id="myCanvas" width="200"height="200"></canvas>

<script>
  varcanvas=document.getElementById("myCanvas"); var
  context = canvas.getContext("2d");

  var x = canvas.width / 2;
  vary=canvas.height/2; var
  dx = 2;
  var dy = -2;
  var ballRadius = 10;

  function drawBall() {
    context.beginPath();
    context.arc(x,y,ballRadius,0,Math.PI*2);
    context.fillStyle = "#0095DD";
    context.fill();
    context.closePath();
  }

  function draw() {
    context.clearRect(0,0,canvas.width,canvas.height);
    drawBall();

    if(x+dx>canvas.width-ballRadius||x+dx<ballRadius){ dx = -dx;
    }
    if(y+dy>canvas.height-ballRadius||y+dy<ballRadius){ dy = -dy;
    }

    x+=dx;
    y += dy;
  }

  setInterval(draw,10);
</script>
```

**3. Image manipulation:** The HTML5 Canvas element can also be used to manipulateimagesonawebpage.Here'sanexampleofapplyingagrayscalefilter to an image:

```
<canvas id="myCanvas" width="200"height="200"></canvas>

<script>
  varcanvas=document.getElementById("myCanvas"); var
  context = canvas.getContext("2d");

  var image = new Image();
  image.src = "myImage.png";
  image.onload = function() {
  context.drawImage(image, 0, 0);

    varimageData=context.getImageData(0,0,canvas.width,canvas.height); var
    data = imageData.data;

    for (var i = 0; i < data.length; i += 4) {
      vargray=(data[i]+data[i+1]+data[i+2])/3; data[i] =
      gray;
      data[i+1]=gray;
      data[i + 2] = gray;
    }

    context.putImageData(imageData, 0, 0);
  };
</script>
```

ThesearejustafewexamplesofwhatcanbedonewithHTML5Canvas.The possibilities are endless!

## Canvas Drawing Techniques

CanvasisapowerfulHTMLelementthatallowsyoutodrawgraphicsandcreate animations using JavaScript. Here are some basic canvas drawing techniques:

**1. DrawingRectangles:**Todrawarectangleonthecanvas,usethe`fillRect()`or `strokeRect()` methods. For example:

```
```
// create a canvas element
varcanvas=document.getElementById('myCanvas'); var
ctx = canvas.getContext('2d');
```
```

```
// draw a filled rectangle
ctx.fillRect(10, 10, 100, 50);

// draw an outlined rectangle
ctx.strokeRect(10,10,100,50);
```

**2. DrawingPaths:**Youcandrawcomplexshapesbydefiningapathonthecanvas. Tocreateapath,usethe`beginPath()`method,andthenuse`moveTo()`,`lineTo()`, and `arc()` methods to draw the path. For example:

```
// create a canvas element
varcanvas=document.getElementById('myCanvas'); var
ctx = canvas.getContext('2d');

// create a path
ctx.beginPath();
ctx.moveTo(10,10);
ctx.lineTo(50,50);
ctx.lineTo(100,10);
ctx.closePath();

// fill the path
ctx.fillStyle='red';
ctx.fill();

// outline the path
ctx.strokeStyle='black';
ctx.stroke();
```

**3. DrawingLines:**Todrawastraightlineonthecanvas,usethe`moveTo()`and `lineTo()`methods.Forexample:

```
// create a canvas element
varcanvas=document.getElementById('myCanvas'); var
ctx = canvas.getContext('2d');

// draw a line
ctx.beginPath();
ctx.moveTo(10,10);
```

```
ctx.lineTo(100, 100);
ctx.strokeStyle='black';
ctx.stroke();
```

**4. DrawingBezierCurves:**Beziercurvesareusedtocreatesmoothcurvesonthe canvas. To draw a Bezier curve, use the `bezierCurveTo()` method. For example:

```
// create a canvas element
varcanvas=document.getElementById('myCanvas'); var
ctx = canvas.getContext('2d');

//drawaBeziercurve
ctx.beginPath();
ctx.moveTo(10, 10);
ctx.bezierCurveTo(50,50,100,50,100,10); ctx.strokeStyle
= 'black';
ctx.stroke();
```

**5. DrawingQuadraticCurves:**QuadraticcurvesaresimplerthanBeziercurvesand can be used to create smooth curves on the canvas. To draw a quadratic curve, use the `quadraticCurveTo()` method. For example:

```
// create a canvas element
varcanvas=document.getElementById('myCanvas'); var
ctx = canvas.getContext('2d');

//drawaquadraticcurve
ctx.beginPath();
ctx.moveTo(10, 10);
ctx.quadraticCurveTo(50,50,100,10); ctx.strokeStyle
= 'black';
ctx.stroke();
```

**6. UsingImages:**Youcanuseimagesonthecanvasbycreatingan`Image`object and then using the `drawImage()` method. For example:

```
// create a canvas element
var canvas = document.getElementById('myCanvas');
```

```
var ctx = canvas.getContext('2d');

//createanImageobject var
img = new Image();
img.src='myImage.png';

// draw the image
img.onload = function() {
ctx.drawImage(img, 0, 0);
};
```

**7. CreatingGradients:**Gradientscanbeusedtocreateinterestingeffectsonthe canvas. To create a gradient, use the `createLinearGradient()` or `createRadialGradient()`method,andthenusethe`addColorStop()`methodto define the colors of the gradient. For example:

```
// create a canvas element
varcanvas=document.getElementById('myCanvas'); var
ctx = canvas.getContext('2d');

// create a linear gradient
vargradient=ctx.createLinearGradient(0,0,200,0);
gradient.addColorStop(0, 'red');
gradient.addColorStop(0.5, 'green');
gradient.addColorStop(1, 'blue');

//usethegradientasthefillstyle ctx.fillStyle =
gradient;

//drawarectanglewiththegradientfill
ctx.fillRect(10, 10, 200, 100);
```

This creates a linear gradient that goes from red to green to blue horizontally, and then uses it as the fill style for a rectangle on the canvas. You can also create radial gradientsusing`createRadialGradient()`,whichcreatesagradientthatradiatesfrom a center point.

**HTML5-CanvasTransformations HTML5**

**- Styles and Colors:**

In HTML5, styles and colors are used to change the appearance of elements on a webpage.YoucanuseCSStodefinestylesforyourHTMLelements,includingfont sizes, font colors, background colors, and more.

**Canvas-TextandFonts:**

In HTML5 Canvas, you can use the `fillText()` or `strokeText()` methods to draw text onthecanvas.Youcanalsospecifythefontsize,fontfamily,andfontstyleusingthe `font` property.

```
 // Set font properties
ctx.font='bold24pxArial';

// Draw text on canvas
ctx.fillText('HelloWorld!',50,50);
```

**Canvas - Pattern and Shadow:**

HTML5Canvasalsosupportspatternsandshadows.Youcancreateapatternusing an image or another canvas element, and then use that pattern to fill a shape or draw a stroke. You can also add shadows to shapes using the `shadowBlur`, `shadowColor`,and`shadowOffset`properties.

**Canvas - Save and Restore States:**

WhenworkingwiththeHTML5Canvas ,youcansavethecurrentstateofthecanvas usingthe`save()`method.Thisincludesthingslikethecurrenttransformationmatrix, fill color, stroke color, and more. You can later restore this state using the `restore()` method.

**Canvas- Translation:**

Translation is the process of moving the canvas to a new location. In HTML5 Canvas,youcanusethe`translate()`methodtomovethecanvasalongthex-axis and y-axis.

**Canvas - Rotation:**

Rotation is the process of rotating the canvas around a specific point. In HTML5 Canvas,youcanusethe`rotate()`methodtorotatethecanvasaroundtheorigin,or youcanusethe`translate()`methodtomovethecanvastoanewlocationandthen rotate it.

**Canvas - Scaling:**

Scalingistheprocessofchangingthesizeofthecanvas.InHTML5Canvas,you can use the `scale()` method to scale the canvas along the x-axis and y-axis.

**Canvas - Transforms:**

Transforms allow you to combine multiple transformation operations into a single operation.InHTML5Canvas,youcanusethe`transform()`methodtoapplyamatrix transformation to the canvas.

**HTML5 Canvas Composition:**

Canvas composition allows you to combine multiple shapes or images into a single image.InHTML5Canvas,youcanusethe`globalCompositeOperation`propertyto specify how the shapes or images should be combined.

**Canvas - Animations:**

HTML5Canvasalsosupportsanimations.Youcanusethe `requestAnimationFrame()` method to request that the browser call a function to updatethecanvasonthenextanimationframe.Thisallowsyoutocreatesmooth, responsive animations on your webpage.