Web Multimedia Assessment 1 60 points

*Remember that you should **not** be changing the content of the provided .html or .css files to accomplish these effects. They should be achieved via jQuery code.

Specs: Magnify Zoomer jQuery Plugin

Write a jQuery plugin (**jquery.magnifyZoom_yourUserName.js**) to create a new jQuery method (named ***magnifyZoom***) that creates the magnification effect on part of an image on the page as demonstrated in the Lesson 1 inClass project that we did together in class.

Use the standard jQuery plugin design pattern we have explored together in class as a template for this project.

It should create a custom object named **MagnifyZoom** with the following items:

- ✓ Create a constructor function for this **MagnifyZoom** object that accepts as parameters any passed-in options from the method call (in a formal parameter named ***options***) and the DOM element object that was selected in the jQuery object that called this plugin method (near the end of the .html file).

  The constructor function should:
  1. store the passed-in element in a property named **$imageContainer**

  2. call a method of the new object named **_init()** passing it the passed-in ***options***

- ✓ Assign an object literal containing the default options values to a property of our **$.MagnifyZoom** object named ***defaults***. The object literal we assign should have the following property: value pairs:

  width = 300
  height = 300
  cornerRounding = '50%'

- ✓ Use the new object's *prototype* object property to create the following methods for our new **$.MagnifyZoom** object:

  - **_init**

    Parameters: ***options***

Purpose: Create an Image object named **imageObject** whose source file will be the file being referenced in the **img.small** element, create properties for our new object and call this object's **_getLocation()** methods.

Create the following properties:

Property name	*Property Value*
options	should be an object formed by deep-merging our object's defaults with any passed-in options
nativeWidth	Numeric property that should contain the ***actual width*** dimension of the image being used (not the width of the image being displayed on the web page).
nativeHeight	Numeric property that should contain the ***actual height*** dimension of the image being used (not the height of the image being displayed on the web page).
$glass	jQuery object containing HTML element being used for our magnifying glass
$smallImage	jQuery object containing HTML element being used for our smaller image that is being displayed on the page

➢ _getLocation

Parameters: none

Purpose: Will handle any mouse movement events on our ***div.magnify*** that contains the image on the page. The logic flow should look like this:

- *Remember* our new object (***this***) in a variable named ***self*** because our event handler is about to change the context of this.

- Set up a mouse movement event handler using jQuery on the div.magnify tag. The event handler function should do the following tasks:

    - Wrap the object the mouse move occurred on in a jQuery object storing it in a variable named **$target**

- Get the coordinates (x,y) of the element the mouse move occurred on *relative to the edges of the web page*. Use a jQuery method on **$target** to accomplish this storing the resulting JavaScript object in a variable named **magnifyOffset**.

  Tip: It would be a good idea to console.log the left and top properties of the above magnifyOffset object out to make sure they look like values you would expect. Just remember to remove the console.log()'s from your submitted project.

- Get the *actual* mouse cursor's x and y position values by subtracting the left and top offset values determined above from the mouse cursor's x and y position values on the page (*hint*: use the automatically passed-in event object to get these). Store the resulting actual x and y position values in properties of your new object named **mouseX** and **mouseY**.

  Note that actual mouse cursor position here just means the mouse's x and y position relative to the edges of div.magnify rather than the edges of the page.

  Again, you should console these values out to make sure they look reasonable…

- Call your new object's **_zoom()** method passing it **$target** which will magnify the part of the image that is under the glass.

➢ _zoom

Parameters: $target jQuery object containing the object the mouse move occurred on

Purpose: Magnify the part of the image that is currently under the magnifying glass.

Incorporate the following tasks:

- Using jQuery methods, fade in the magnifying glass (**$glass** property) if the mouse cursor is inside div.magnify and fade it out when the mouse leaves div.magnify. This sounds like an if-else.

- If the magnifying glass is visible (use a jQuery method on your **$glass** property to determine the truth of this condition), perform the following tasks:

  - Use jQuery methods to determine and store the width and height of your magnifying glass. Store the results in variables **glassWidth** and **glassHeight**.

- Determine the x and y position values for the *background image* of your magnifying glass storing them in variables **rx** and **ry** respectively.

  Tip: the formulas to determine these values were covered in the Lesson 1 inClass project we did together.

- Determine the *left* and *top* positioning values you will use for your magnifying glass storing them in variables **posX** and **posY** respectively.

  Tip: these would be determined by subtracting half the width/height of the magnifying glass from the mouse's x/y positions (again, see the Lesson 1 inClass project for example of how we did this if needed).

- Use jQuery's css() method to apply the new values you have to the magnifying glass and its background.

  Set the following properties of your magnifying glass:

Property name	Value
width	**width** option
height	**height** option
border-radius	**cornerRounding** option
top	**posY** value
left	**posX** value
Background-position	Use **rx** and **ry** values to build a string appropriate for setting your element's background image's position

NOTE: ***Your application must be functional to receive above 70% credit***.

Deliverable: Zip up your entire project into a folder named **MagnifyZoom_yourUserName** and submit to the provided dropbox in E360.