Video and Audio

Overview

In this lab, you'll implement a web page that displays a video and plays an audio clip.

Roadmap

There are 3 exercises in this lab, of which the last exercise is "if time permits". Here is a brief summary of the tasks you will perform in each exercise; more detailed instructions follow later:

1. Displaying a video
2. Controlling video playback
3. Playing an audio clip (if time permits)

Familiarization

Open Internet Explorer and browse to `VideoPlayer.html` in the *Start* folder. The web page has a video player, but you have to hover over it before it plays. When you hover over the video player, the following things happen:

- A short audio blip is played.
- The video starts playing.
- The shape of the video player mutates.
- A reflection of the video fades into view beneath the video player.

Now move the mouse off the video player. The following things happen:

- The video stops playing.
- The shape of the video player returns to its original rectangular state.
- The reflection of the video fades out of view beneath the video player.

Exercise 1: Displaying a Video

In File Explorer, go to the *Start* folder and open `VideoPlayer.html` in the text editor. Where indicated by the TODO 1 comment, add a `<video>` element to play the video in the `videos` sub-folder.

- Make sure the `<video>` element supports all the available video formats.
- Configure the `<video>` element to display the provided poster image during loading, and for the `<video>` element to automatically preload.
- Also configure the `<video>` element to show the controls, to start automatically, and to loop indefinitely.

Open the *Student* web page in Internet Explorer. Verify the video starts playing immediately. When you hover over the video player, it should automatically change shape (due to a `:hover` CSS rule in the style sheet). Likewise, when you move the mouse away from the video player, it should become rectangular again.

Exercise 2: Controlling Video Playback

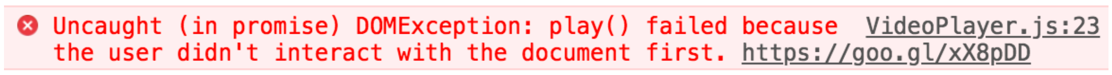
Modify the `<video>` element in `VideoPlayer.html` so that the video doesn't start automatically.

Now open `VideoPlayer.js` in the text editor and take a look at the existing code. Don't worry too much about the `translate()` and `scale()` functions – we'll be taking a good look at canvas transformations in the next chapter.

Add code so that the web page only plays the video while the mouse is hovering over the video player, as indicated by the TODO comments:

- TODO 2a:
  Get the `<video>` element and store it in `myVideo`.
- TODO 2b:
  Handle the `mouseenter` event for the video. When this occurs, play the video and fade-in the reflection canvas (use the jQuery `fadeIn()` function).

  Important: Chrome no longer allows videos to be played automatically before the user interacts with the document. And hovering isn't regarded as interacting. So after refreshing the page, click the page (somewhere) first. Read more here: https://developers.google.com/web/updates/2017/09/autoplay-policy-changes

  ```
  Uncaught (in promise) DOMException: play() failed because  VideoPlayer.js:23
  the user didn't interact with the document first. https://goo.gl/xX8pDD
  ```

  In the solution:

  ```
  var promise = myVideo.play();
  promise.catch(function(error) { alert(error.message); });
  ```

- TODO 2c:
  Handle the `mouseleave` event for the video. When this occurs, pause the video and fade-out the reflection canvas (use the `fadeOut()` function).
- TODO 2d:
  Handle the `play` event for the video. When this occurs, set up a JavaScript interval (e.g. every 20ms) to copy the current video frame into the reflection canvas.
  Handle the `pause` event for the video. When this occurs, clear the interval set up in the `play` event.

  Hint 1: To display the current video frame in the canvas, call the `context.drawImage()` method. Pass in 5 parameters, i.e. the video object followed by the left, top, width, and height values for the canvas.

  Hint 2: To apply the opaque-to-transparent gradient to the canvas, call the `context.drawRect()` method. Pass in 4 parameters, i.e. the left, top, width, and height values for the canvas.

Refresh the *Start* web page in Internet Explorer. Verify the video only plays while the mouse hovers over the video player. Also verify the reflection canvas fades in and out accordingly.

Exercise 3 (If Time Permits): Playing an Audio Clip

In this exercise, you'll refactor the web page so that it plays a short audio clip just before it starts playing the video…

In `VideoPlayer.html`, where indicated by the TODO 3 comment, add an `<audio>` element to play the audio clip in the `audios` sub-folder.

- Make sure the `<audio>` element supports all the available audio formats.
- Configure the `<audio>` element to automatically preload.
- Do NOT configure the `<audio>` element to show the controls, to start automatically, or to loop indefinitely.

In `VideoPlayer.js`, refactor the code as follows:

- When the mouse hovers over the video player, DO NOT play the video or fade-in the reflection canvas. Instead, just play the audio clip.
- When the audio clip has ended, now is the time to play the video and fade-in the reflection canvas.

Refresh the *Student* web page in Internet Explorer. Verify the audio clip plays when the mouse enters the video player. Ensure the video only starts playing after the audio has finished (you might like to experiment with the audio `playbackRate` property to affect the audio playback speed, to see what effect it has.