Read file content as binary

INTRODUCTION

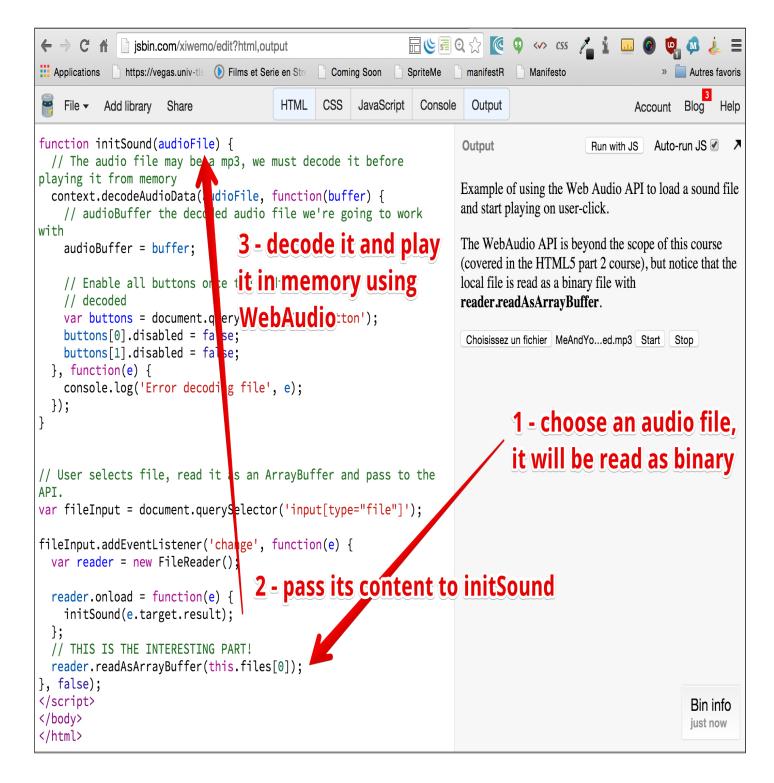
This method is rarely used except for loading "raw" binary data. For images you would like to see in your HTML page using the or for drawing in a canvas, or for audio and video files that you would like to play using the <audio> or<video> elements, you would preferably use thereadAsDataURL method presented in the next page of the course.

readAsArrayBuffer is often used for reading audio samples that should be loaded in memory and played using the WebAudio API, for loading textures that you will use width WebGL for 3D animations, etc.

EXAMPLE 1: READ A LOCAL AUDIO FILE AND PLAY IT WITH THE WEBAUDIO API

The WebAudio API is useful for reading audio sound sample from memory (no streaming), and has been designed for music application and games. This example shows how a local audio file can be read and played directly in the browser, without the need for a server!

Example on JS Bin (does not work on IE, as it does not support the WebAudio API). We could not embed it here on the edX platform as it prevents code that uses Ajax to run in its pages.



Source code extract:

```
// User selects file, read it as an ArrayBuffer and pass to the API.
var fileInput =document.querySelector('input[type="file"]');
fileInput.addEventListener('change', function(e) {
   var reader = new FileReader();
```

```
reader.onload = function(e) {
    initSound(e.target.result);
};

10. // THIS IS THE INTERESTING PART!
    reader.readAsArrayBuffer(this.files[0]);
}, false);
```

Explanations:

- Line 1: we get a pointer to the file selector, the variablefileInput.
- Line 4: we define a change listener. In this example, we use an anonymous function directly included in the listener definition (the listener is the function (e) {...}).
- Line 11: when a user chooses a file, the listener will be executed. Line 11 will start the reading of the file content, as a binary file (this is what readAsArrayBuffer means: read as binary!). Once the file will be entirely read, the onload callback will be asynchronously called by the browser.
- Line 7 is the onload callback, executed when the file content is loaded in memory. We pass the file content to the initSound function (see JS Bin example for complete source code) that uses WebAudio to decode it (it may be a compressed file, a mp3 for example, and WebAudio works only with uncompressed audio formats in memory), and to play it.