Event-Driven Programming

Green Screen Web Page

Upload and Display an Image 7 min
Try It! Upload and Display an Image 30 min
Convert Image to Grayscale 8 min
Try It! Convert an Image to Grayscale 1h
Moving to CodePen 9 min
Try It! Green Screen Online 1h 30m
Quiz: Interactive Web Pages 7 questions
End of Module Survey 10 min

Which of the following best describes how the upload function works?

- It accesses the image file selected by the user and draws it on the canvas.
- It accesses the image file selected by the user and stores it on the webpage.
- It accesses the image file selected by the user, creates a SimpleImage from the image, and draws it on the canvas.

Correct

Continue

Upload and Display an Image

Have a question? Discuss this lecture in the week forums.

Interactive Transcript

Search Transcript

English

0:03
In this lesson, we'll go over a few concepts that will allow you to upload image files and display them in an HTML canvas. We'll need some new programming concepts and JavaScript tools as we move towards creating an interactive web page for the Green Screen algorithm.

0:20
You'll be able to create more web pages using these new concepts and tools as we move from a very simple page towards the more complex and creative Green Screen and MiniProject pages.

0:33
We'll start with a prototype, a web page that's a model of what we want, but is simpler to understand. We need a new type of input element to upload our image files but we'll start with a simple idea in this first prototype so we can concentrate on the idea rather than the details of the new HTML element.

0:54
A simpler input type would be text input which will display a text box like the one you see here that allows the user to type text.

1:03
We'll test and understand the concepts with a prototype, reviewing the JavaScript for processing this, and any other HTML element.

1:13
Then we'll change the input type to file, and we'll introduce concepts needed to upload an image file and display it in an HTML canvas. Let's look at this prototype to understand the web page and the JavaScript. The HTML text input element allows the user to enter any text. As you can see in this simple page, the user has entered lion.jpg and is about to click the button.

1:42
Here we'll use a button to process an event. It's possible to process text when the user presses the Enter key, but we'll use a button since they're familiar.

1:53
As usual, when the user clicks the button to generate an event, we'll write JavaScript to handle that event. Here's that using the onclick event handler.

2:06
Here's the JavaScript to process what the user entered. We use familiar JavaScript to access the value of the text input element and display that result in an alert box. So when the user clicks Upload, the alert box pops up.

2:21
Remember, this is an example of text input. The input element doesn't care if we enter a file name. We could have entered any text. But if we change the input type to file we will be guaranteed that the value we get represents an actual file. Though, as we'll see soon, we will need to do some work to ensure that that file is an image.

2:44
Let's look at some new concepts.

2:47
We'll use the last Pen as a model and extend that prototype to a web page that allows the user to choose an image file, upload it, and display it in the web page.

2:56
We'll replace the two inputs of type text and button with a single input of type file.

3:02
We'll create an image from the file chosen by the user and upload it. This will be a SimpleImage from the Duke Learn to Program library that you've worked with before.

3:12
You'll create the web page by extending the functionality we saw in the prototype. Starting with something familiar helps with understanding the new concepts. In this case, the file input and a library of JavaScript code.

3:26
Starting with something familiar also helps with debugging since we start with a working web page and working JavaScript. Let's look as how HTML and JavaScript combine to allow you to upload and display an image.

3:40
Here's the HTML for specifying an input element that allows the user to choose a file and upload it to a website.

3:48
You'll use file, rather than button or color, as the type of the input. You'll make sure the user only uploads one file, not several, by using the multiple attribute and making it false, signifying that multiple files cannot be chosen.

4:05
You'll only allow the user to select image files, not text files, and not audio files, for example.

4:11
You do this by specifying a value for the accept attribute, the value is image/* as shown here. * meaning all image types, not just JPEGs for example.

4:25
As usual, you'll provide an input to make it possible to find this element within your JavaScript code.

4:31
You'll use onchange for the event handler. This event is triggered when the user chooses a file by clicking the button displayed on the input element. Note, you do not need to add the button element, it's included with the file input type.

4:48
Here is what the user sees when she clicks on Choose Files. A file selector appears. This one only allows image files. Yours may look different since different browsers handle this restriction differently. And once the image lion.jpg is selected, the file input is changed to contain the file. This triggers the onchange event handler. So let's see what the upload function should look like in order to achieve our goal of displaying the image in our canvas element.

5:22
The upload function gets the canvas element and the file input element, each by its ID specified in the HTML.

5:32
Then you'll create a simple image variable from the HTML file input itself. Wait! SimpleImage is not a standard JavaScript. It's a library of JavaScript code created for this course and hosted at DukeLearnToProgram.com. The code is automatically included when you use our custom JavaScript environment. But for web pages you create, you need to specify the source of this library. Let's see where to do that in CodePen.

6:04
In your editing view for the CodePen page, the source for JavaScript goes in the HTML panel, not the JavaScript panel, enclosed in open and closed script tags, with the attribute src for source. You can see the last part of the URL is simpleimage.js because that is the library we want the page to be able to use.

6:29
Outside of CodePen, you would also use the script tag. Here is the code that includes the simple image.jslibrary from our Duke Learn to Program site.

6:39
Now that we know how to tell a web page where to find our JavaScript code, let's finish putting an image in the canvas.

6:48
We have just created a new simple image from the file input HTML element. The last step is to use the drawTo method, a method included in the SimpleImage library. We will call image.drawTo and use the canvas element as the parameter to indicate the SimpleImage should be drawn on a specific canvas of our choosing.

7:11
As always, you should try not to memorize all of these methods. You can consult the documentation at any time, like this documentation from DukeLearnToProgram.com.

7:23
Now let's see the upload function in action.

7:27
As you can see, when lion.jpg is selected as file input, the file selector is changed and the onchange event handler is triggered. For this event, that is the upload function which displays the selected image on a web page by drawing it to an HTML canvas. Now you can create web pages that allow users to upload images too.

Downloads

Lecture Video mp4
Subtitles (English) WebVTT
Transcript (English) txt
Lecture Slides pdf

Would you like to help us translate the transcript and subtitles into additional languages?