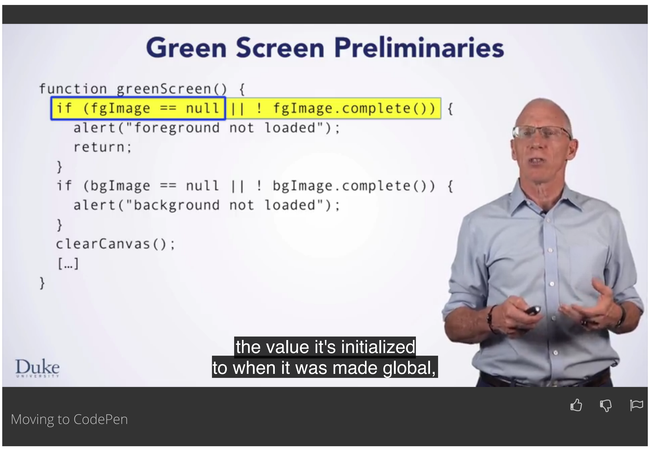

Moving to CodePen

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

Interactive Transcript

Search Transcript English

0:03
Let's look at how to use concepts, tools, and programming to create an interactive Green Screen page. We'll use HTML elements like canvas and inputs for files, and buttons to create an interactive green screen page like this one. You can see two images have been uploaded. I'm about to press the create composite button. That will combine the two uploaded images into one using green screen code you've seen in the Duke Learn to Program environment.

0:32
There are several steps you'll need to take to use the code you developed in the Duke Learn to Program environment and make it work in a web page. You'll need to use HTML input elements to allow users to select files. Those input elements need event handlers to connect the events to JavaScript code. You'll need to create functions to load the images and global variables so that all functions can use these images. To make the green screen appear you'll use the drawTo method instead of print. Finally, we'll show you how to protect your code with checks to make sure that it works without error.

1:14
Let's look at the components of this interactive web page. We'll look at the HTML elements first, and then we'll connect these to JavaScript. You can see here that there are two standard canvas elements. Each one has a border that's part of the CSS for the canvas. There's one canvas where the foreground image will be displayed, and another canvas for the background image. You'll also see four input elements. Two are file inputs for the images, and two are buttons that change what you see in the canvases. The file inputs are labeled with their purpose on the web page. Upload a foreground image and a background image. Two buttons are used, one to create the green screen composite and one to clear the canvases.

2:03
Let's look in detail at HTML file elements.

2:09
We have two file inputs, one to upload the foreground image and one for the background image.

2:16
Here's the HTML for these labeled images. We'll review the HTML elements and the concept that you've seen before. We create a file input to allow the user to upload and display an image file. Here's the foreground file input and there's also a file input button for the background image.

2:38
These file input buttons have unique IDs. This will allow them to be identified as needed in the JavaScript code that interacts with the web page. We see fgfile for the foreground input and bgfile for the background input. We connect these inputs to corresponding JavaScript to load and display the image in a canvas. We use the onchange event handler and call a JavaScript function. Here we're calling, loadForegroundImage. And here, the event handler calls a JavaScript function named, loadBackgroungImage. Let's look at the JavaScript code that's called when the user uploads an image.

3:21
Uploading and displaying an image use concepts we've seen before. Here's the JavaScript function to load the foreground image. The background image function is very similar, so we'll just look at this one. We'll see that there's a global variable used for each image. The foreground image is referenced by the global variable fgImage. Here you see that fgImage is initialized to null, initializing global variables is a good idea. Using null will allow us to determine if an image has been loaded. When the user clicks, they create composite button for example.

3:56
Null is a special value used in JavaScript and other languages to represent nothing. You'll see this idea again later and with other programming languages. Here we see that the code that accesses fgImage. Notice it doesn't use the word var, because the variable fgImage is global.

4:17
The variable is assigned a new simple image element, created from the file inputElement.

4:24
Here we see fg element used again to draw to the canvas element that will be displayed on the web page.

4:33
We will also use many local variables, like var imgFile, it references the file input for the foreground image. We use that HTML element because it passes a parameter when we create the simple image. Using local variables with the word var is a really good idea. Here, the local variable represents the canvas, where the foreground image will be displayed. When a value is accessed in multiple functions, making it global is required, otherwise, make variables local. To avoid losing track when you have hundreds of variables, which might happen in the mini project.

5:12
Let's look at the JavaScript code to create the green screen composite.

5:19
We're going to look at the steps to create the green screen composite.

5:24
We'll need some basic ideas from image processing that you're seeing in the Duke Learn to Program environment. We've adapted these for use in a web page, when the user uploads an image, and then processes the image. For example we need to check that the user has uploaded two images, before clicking the create composite button. We accomplish this by checking to be sure that the foreground image is ready to go. If the global variable fgImage is null,

5:54
the value it's initialized to when it was made global, then the user hasn't clicked the upload foreground image button we just saw. Because that JavaScript creates a non-null version for fgImage. It's also possible that the user has uploaded the image, but the image hasn't been completely constructed, large images take time to create. The method .complete allows us to determine if the image is completely loaded. Here it says not complete, that's the bang or exclamation point. And if it's not ready because either it's null or it hasn't finished loading, we'll need to let the user know. We do this by displaying an alert, indicating the code won't work. We check both the foreground image and we check the background image before creating the green screen composite.

6:49
Now let's look more closely at the code to create the composite.

6:53
Here's the JavaScript code adapted from the Duke Learn To Program or DLTP environment to create a green screen composite. The first step is to create new image with the appropriate width and height. It will be the composite of the foreground and background image.

7:09
The new image is initially all black, so we need to set each pixel by looping all for all the pixels of the foreground image and copying either a foreground pixel or a background pixel depending on how green the pixel we're looping over is. Here, we see that if the pixel's green value is greater than the sum threshold, we use the background pixel in the new image we created. And if it's not greater than the threshold, we use the other pixel.

7:39
We'll also need JavaScript to display the new image in a canvas. Notice that the threshold variable, green threshold must be global, because it's not defined here with the word VAR.

7:52
Let's look at the final steps, we'll need to display or draw the new composite image to create a canvas element with the image in it. We'll use the .draw2 method, before making sure that the image appears. We know it will because we've checked to make sure that the image is loaded. And we'll clear the canvas elements before drawing so that we show only one image, the green screen composite. Let's take a look.

8:22
Your working green screen page allows the user to interact by uploading images. One for the foreground and one for the background.

8:31
We have to wait for those images to load and be displayed.

8:36
Then we'll click the create composite and watch the green screen be created. We can see that the button stays clicked. It takes some time to create this composite. Then we see it, riding dinosaurs, we can also clear them. There are no dinosaurs, where do they go? We know that combining two people, and dinosaurs can be lots of fun. You will also get to have lots of fun when you use the green screen program to create your own green screen composites. Have fun with programming.

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?