Event-Driven Programming

Green Screen Web Page

Which of these must you do in order to draw shapes or write text on a canvas?

- Access the context of the canvas
- Change the background of the canvas to white
- Use CSS to put a border around the canvas

Skip Submit

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

Interactive Transcript

Search Transcript English

0:03
Hi, let's see how to create a webpage and write code that will move us closer to the green screen page. Rather than using elements, we'll use a new HTML element named <canvas>. The canvas HTML element is used to store images in the Duke Learn to Program, or DLTP environment that you've seen before. A canvas has a different functionality than a div. As you'll see, it's possible to draw shapes, lines, and more. We'll only explore basic functionality of this HTML element. You'll see how it's accessed via JavaScript and you'll be able to do more if you explore additional resources. Previously you have seen the div element used to specify a section of a web page. So, the background color or other attributes can be changed programmatically by referring to the element ID. In this case, d1. On this page, the Make Lime button changes the background color to lime. And the Make Ybutton changes the background color to yellow. Let's try using canvas element instead of div.

1:12
You'll see that we have to make a few changes to the CSS, and to the HTML panels. When we use canvas, the hello text disappeared.

1:27
It turns out that special methods are necessary to write text in a canvas.

1:33
But as you can see here with the canvas, the background color changes work the same as they did with HTML div elements.

1:43
We'll see more closely how to program with a canvas. As we explore this functionality in more detail. Let's talk about how a canvas is used. It's used specifically to draw graphics. Usually, it's used with JavaScript. You should think of it like a container for graphics where the container needs to be filled in which you do by writing a script. There are several methods that you use to create these graphic tasks such as drawing paths, boxes, circles, text, and adding images. We'll show a simple interactive page using a canvas element. We'll create a webpage with two buttons, using HTML, CSS, and JavaScript. Each button will change a different feature of the HTML canvas element. One will make the background lime. And the other will draw yellow squares and create text in the canvas all with JavaScript. Let's program. Let's look at an example of using CodePen to combine HTML, CSS, and JavaScript to make an interactive webpage using a canvas. As we saw in the recent lesson, I'm going to have a canvas that's going to make both color background changes and draw using the graphics element. I'm starting with a canvas. We see that we've got the canvas element here. It's id is d1. And we have two buttons, one that says make lime and calls a dolime( ) JavaScript function. One that says make yellow and calls a doyellow( ) javascript function. The functionality that we have that we start with, which I've already coded, is make lime. We can come over here and just review a couple of concepts. I use the standard document.getElementByid to grab the element whose id is d1 as a reminder, that's this canvas. And then I make the background color lime, that's why the function is called dolime. In the make yellow function, which is the button that calls doyellow, I get the element by it's ID and I make the background white. So, right now making yellow actually turns it white. But I'm going to make it white in preparation for making some yellow rectangles, let's also just review very quickly that there's some CSS for the canvas that styles it have a specific width, a specific height and a border. That's why when we start with the page blank, we see that the canvas is here

4:15
with nothing in it, a little narrow border that's one pixel wide as we can see here specified in this CSS and these two buttons. So, let's look at the doyellow JavaScript function and see how to make yellow rectangles appear by using the canvas. The first step is going to be that rather than drawing in the canvas, we're going to have to access the context of the canvas. That's a new concept for canvases, that the graphics are actually displayed in the canvas's context. The way we do that is create another variable, I'll call it ctx for context and I'll get the context from the canvas which is stored in variable d1. And the name of my method is getContext. I'm actually going to take off my bracelet because it's kind of jangly. So excuse me, while I remove it and put it right there. It's kind of hitting the keyboard and so annoying to me, might be annoying to you. So I'm going to call git context and I have to specify that I'm getting the 2d context, in other examples we might see. Something beyond two dimensional, but all the graphics I've ever done and that will ask you to do are the 2D graphics. Now that I have the context, I have to fill it with yellow squares. The first thing I'm going to do is do something similar to what I did by setting the background color, I'm going to set the fill style to yellow.

5:44
That means anything I draw will be in yellow.

5:48
Then I'm going to fill a rectangle. That's the name of the method there is fillRect. And I have to specify where it's going to be. I'm going to use this to see what happens. I'm going to actually test it. First I'm going to save it.

6:03
And I'm going to review the code that I wrote and then test it.

6:08
i got the graphic context, then I set the fill style of that context to yellow, and then I created a fill in rectangle and I specified the coordinates of the upper left portion of that rectangle which are 10,10 and the width and height of the rectangle which are 40, 40. Let's see how that works.

6:30
There's, I made it lime. Making yellow doesn't do anything. So right now, I have to look because there's a bug in my code. Luckily, CodePen helps me find that bug. You can see that this is in yellow whereas this is in blue. I didn't spell the name of the variable right. Context is ctx, so now I'll call Make Yellow and I see up here a yellow rectangle exactly as I expect it. Just to see I understand the parameters, let's make this rectangle bigger, 60 by 60. I'm hitting Save, Make Yellow. That rectangle is definitely bigger just because I can, I'm going to make that rectangle a 100 by a 100. I want to make sure I understand that that makes the rectangle bigger. Sure enough, it's bigger. I can actually change where it's drawn by changing that 10 to a 50. Let's see what happens, the rectangle is lower on the page because these are the X and Y coordinates. X goes along the width and Y goes along the height with the upper left corner, zero, zero. As you learned in our Duke Learn to Program. When we were talking about images. I'm going to set that back to 10, 10, 40, 40. And then I'm going to draw another rectangle. This rectangle is going to be near to the other one, almost adjacent to it. I'm going to make it at 60, 10, and also make it 40, 40. I'm going to not forget my semicolon. Let's see what happens when I make that yellow. I now have two rectangles, notice if I make the background lime, that draws behind the rectangles because it's the background color making yellow, changes it to white because as we saw here the background color was changed to white. Remember that when you write code the method or function that you call execute from top to bottom. Make it white get the context, make the context yellow draw two rectangles. I've got one more thing to do with the graphics and that is I want to create some text. I'm going to make the text in a different color so, I'm going to set the fillStyle to black.

8:34
And then I'm going to draw some text. And I draw text by saying, fillText. I specify the help, the text I want to draw, which in this case is hello. And I specify where to draw it, which is 10, 80. Let's see how that works.

8:50
I can't see hello anywhere. Once again I've spelled a variable wrong. Apparently I do that often by mistyping. There's Hello, very very small down there below the yellow rectangles. I can make the font different by setting the font style of the text just as I set the fill style. So, I'll set the font to a bigger font 30 pixels. And because I'm fond of Arial, I'll change the font to Arial. Notice, I now have Hello in a bigger font, it's 30 pixels. Just to make sure, let's make it 50 pixels, and see if that makes it bigger. That's definitely bigger. Unfortunately, it overwrites the rectangles, so I'm going to set it back to 30. I'm going to save it, I'm going to test all my buttons to make lime, make yellow, make lime. I've got two yellow rectangles done by the context fillRect methods. Notice I've set the style to yellow, if I had set the style to purple for example and spelled it correctly. When I call make yellow, it makes purple rectangles. That might be confusing, so I'll leave it back at yellow.

10:06
You can explore more of the graphics context using the resources that we provide. Have fun writing code with canvases, I do.

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?