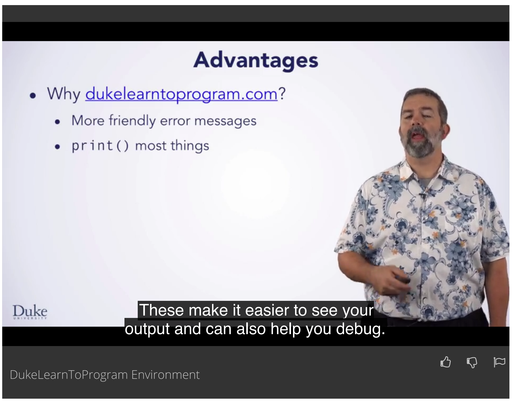

DukeLearnToProgram Environment

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

Interactive Transcript

Search Transcript

English

0:03
Now that you're learning a bit about JavaScript, you might wonder where you would write it, or how you would run it.

0:09
In CodePen, you can write JavaScript in the right frame, which we have highlighted here. If you use some other tool to write a Web page, you can generally write JavaScript code in your Web page easily, though the specifics will depend on the tool.

0:23
However, for getting you started, we have a slightly more novice friendly environment, which we hope will help you write and debug your code more easily. If you want to write JavaScript for a Web page somewhere else, you can always use this environment to develop, your code then copy it into your Web page later.

0:41
Here, you can see a screenshot of the dukelearntoprogram.com web environment for developing JavaScript code. This is a page for the green screen algorithm, where the box to write code is on the left, is preloaded with the algorithm that we have developed together in a previous video. If you click on this button at the bottom, it will run this code. Here, the code is not finished so not much would happen. But if you were to finish this code, as we will do soon, the output would show up on the right here. In fact, if we had finished this code before we hit the run code, you'd see Drew and Robert appear with their dinosaurs.

1:19
Okay, so why use dukelearntoprogram.com rather than just writing your code in CodePen? For one, we have set up the Duke Learn To Program environment to give you more friendly error messages. This will make it easier to fix syntax errors in your code or to find and fix problems. We have also set it up so that you can print a variety of things, such as SimpleImages and SimplePixels. These make it easier to see your output and can also help you debug.

1:47
Speaking of SimpleImage and SimplePixel, these are not standard JavaScript libraries. Instead, they are libraries that we have set up for you so that you can solve interesting problems earlier in your learning. These libraries make it easier to manipulate images without a bunch of nitty-gritty JavaScript knowledge. Of course, if you want to use them for your Web pages that you make elsewhere, you could import the libraries and use them in any Web page you build.

2:14
For this course, later we'll look at how to use them in CodePen.

2:19
These pages are also set up to be preloaded with images and pseudocode for the problems that you can work on so that you have a good point to get started from. Lastly, most programmers use environments like this to make the tasks of programming easier, whether it is CodePen, BlueJ, which we'll introduce when you learn Java in the next course, or dukelearntoprogram.com. All right, let's see it in action. Welcome to the Duke Learn to Program environment. This is what we've created for you to help you learn to do your JavaScript programming. We've done that because we believe that it provides a better environment. So let's start writing some code and playing with it and just sort of seeing what happens.

3:02
The first thing I'm going to do is simply create a variable,

3:07
name it x, and give it an initial value. Then I'm going to create another variable. And I'm going to give that a value, As well, that's dependent upon x. And now I want to see the result of that computation, see if I've actually done something that is going to be what I expect. And so I can print out the value of y. And when I run it, when I press the Run Code button down here, I see the results over here in the See It window. So the result of 3 x 3 is going to be 9 which is just like what we'd expect.

3:46
Let's do something a little more complicated. Let's write our own function. So that starts with the function keyword and then we can give it whatever name we want. I'm going to write a very small function that simply squares whatever value is passed in. So I'm going to say the square of x is what happens when you multiply x by itself and you get that result. You'll notice that when I put in that initial brace that it put one in for me, kind of filling that out. That's again an attempt to help you by filling in that stuff. And you'll also notice here that the fact that I used a variable named x doesn't have anything to do with this variable named x up here. This is just a generic name that I've chosen to refer to any value that you pass in when you call square. So now let's try actually using that. And we'll go ahead and make a new assignment to y. And we'll say, we want y to be the result of calling square of 5, say.

4:54
So now when I run that and I want to see what the result of that is. So I'm going to print out y again and I run that code, you'll see that the first print is still called. So those lines are executed and I get the result of that and that's 9. Then I define my function, and then I call that function with the value of 5. That multiplies 5 by 5 and assigns that value to y. And I print that value out and I get 25. So both of the print statements occurred, and I get to see their results. I can also print out the result of calling a function directly if I want. I don't have to assign it to a value. If I just want to see that, I can say print the square of 4. And again, you'll see that it filled in those parentheses for me. And I want to put a semicolon at the end of the line to mark that I'm done with that line. And I say run code. And again all of my print statements are executed in the order that we see in my code. So first I get 9 as before. Then I get 25 as before. And now I get 16, which is the result of calling it directly.

6:08
If I wanted to, I could choose not to execute any of these prints. And typically, a way to do that, if I'm not quite ready to get rid of the code, is I can go ahead and I can put a comment in front of it. I can put those two slashes. And again, you'll notice that the environment color codes that, just like you saw in the CodePen environment, this is a common feature of most development environments, and so it's made that a green color to indicate that that's a comment. Typically you would use comments like the one on our first line here that's used to kind of describe your code or explain what's going on. But you can also use it to just comment out a piece of code to say, I'm not ready to have this execute during this run of the program. And so you'll see, now I just get the two print statements, the 9 and the 16. Okay, let's try one more thing. Let's go ahead and make a variable that represents an image.

7:07
So if I create a new SimpleImage, then I can go ahead and print out the result of that. But what value should I use to initialize my SimpleImage with? Well, in our environment we've provided you with a number of standard images that are already uploaded and ready to go. And different problems may have different sets of images that are associated with them. So in this particular case, we can see that, we have the very first image here is chapel.png. So if I just put that string in there, chapel.png, then that will refer to this image down here, that's already been loaded.

7:50
I can go ahead then and say print my image. And, when I run that code, you'll see I still get the first two, the 9 and the 16, those prints still being executed. And then I also get an image printed out over there, which is the image that I loaded in. And I can go ahead and change that if I want. Over here is on the other end, is an image of Professor Roger. So I'm going to go ahead and put Roger in there instead of chapel. And when I run that code, I get a different image.

8:24
If you want to, you can also drag your own images over. So here I have some images. And I can go ahead and drag the image down into that space and you can see that it loaded brownhorse.jpg, and it also includes the size of that image so that I can do calculations on that or check to see if I got things correctly. So I'm going to go ahead and change this to brownhorse.jpg. And in addition to printing out the image, I'm going to print the width of that image.

9:08
And when I run that code, you can see that I got 1280 as the width, which was noted down here. And you can see that I got a very, very big image that is taking up a lot of space over in the See It area. But I can see that image if I want.

9:25
I do want to take a moment to assure you that anything that you upload to your browser to display as images to work with are going to stay on your local machine. So all of this work that's happening through this environment is happening on your local machine and your local browser. Nothing's being sent to Coursera. Nothing's being sent to Duke. Nothing's being sent out of your computer because you're doing that. In order to make sure you don't lose your code, when you're using your web browser, if you try to go to a different page and you haven't yet saved your code, you'll get an analog box that says, hey, are you sure you want to do that? And you can go ahead and say, I'd like to stay on this page and then save the code to your computer to make sure that you have a version of it that you're comfortable with. And now I can go ahead and leave the page because I have a saved version of it.

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?