Event-Driven Programming

Introduction 3 min

Buttons with Divs 6 min

Try It! Buttons with Divs 30 min

Changing Pages Interactively 7 min

Try It! Change Pages Interactively 30 min

Using HTML5 Canvas 10 min

Try It! Canvas 30 min

Inputs and Events 8 min

Try It! Inputs and Events 30 min

Practice Quiz: Event-Driven Programming 3 questions

Green Screen Web Page

A PDF copy of all of this lesson's Try It exercises can be downloaded from the **Resources** tab.

Create Canvas Elements

1. Fork your previous practice pen, so that you start with two div elements and several buttons.

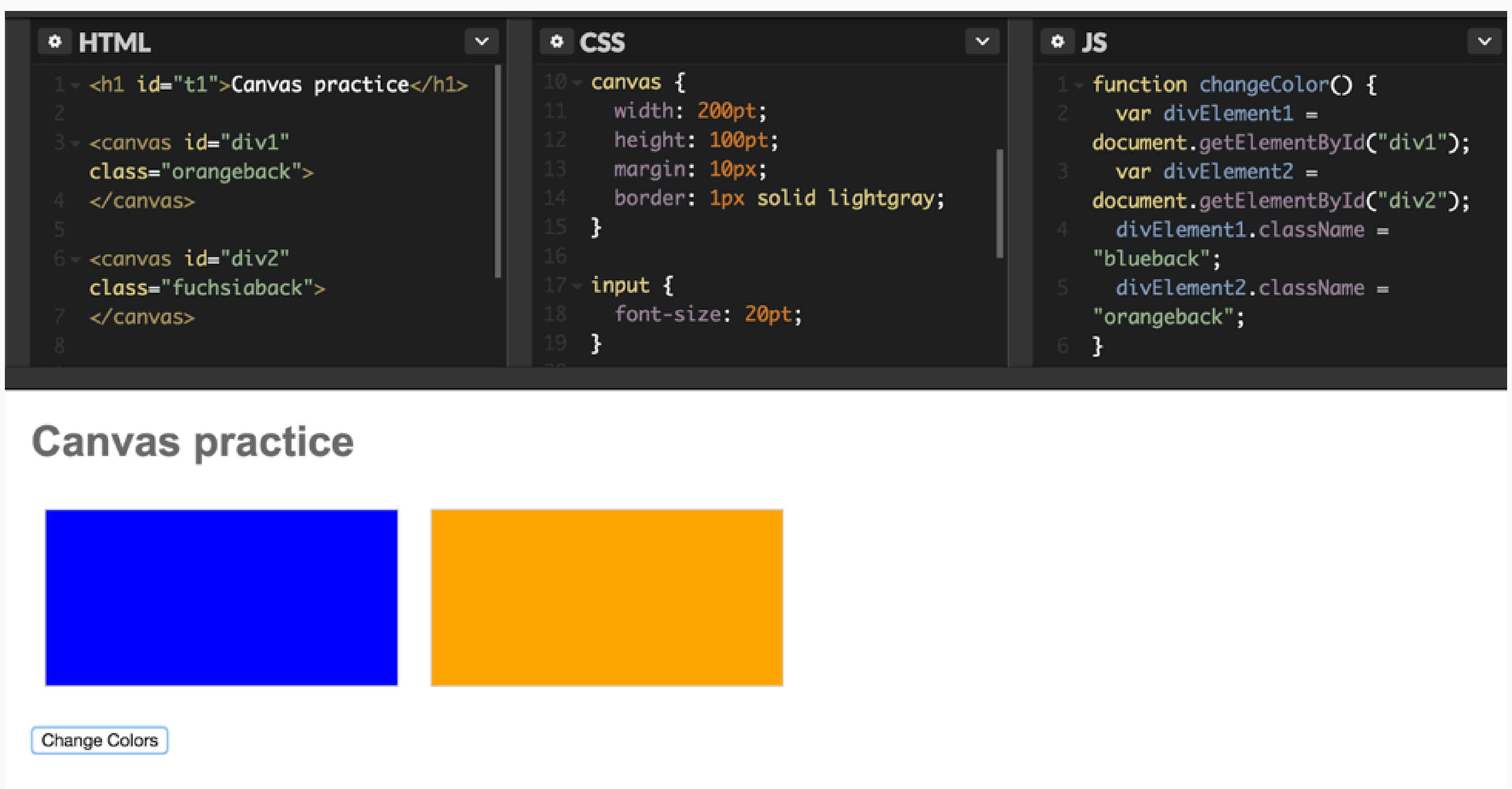
2. In the HTML pane, change each div element to a canvas element. While canvas elements have different functionality than divs, they can both appear as rectangles when a background color is set. Note that the text disappears, since text must be created differently on a canvas.

3. In the HTML and JS panes, remove all but the "Change Color" button and any related functions. You can also remove text between the canvas tags, since it will not appear.

4. Specify a width and height for canvas elements in the CSS panel.

5. Check that your "Change Color" button still works.

Need help? See the following example; review the **Using HTML5 Canvas** video; and ask for help in the forums.



```html
<h1 id="t1">Canvas practice</h1>

<canvas id="div1"
class="orangeback">
</canvas>

<canvas id="div2"
class="fuchsiaback">
</canvas>
```

```css
canvas {
  width: 200pt;
  height: 100pt;
  margin: 10px;
  border: 1px solid lightgray;
}

input {
  font-size: 20pt;
}
```

```javascript
function changeColor() {
  var divElement1 =
document.getElementById("div1");
  var divElement2 =
document.getElementById("div2");
  divElement1.className =
"blueback";
  divElement2.className =
"orangeback";
}
```

Change Canvas Colors with Buttons

1. In the HTML panel, create two buttons, one for each of two colors you choose (e.g., red and blue). Do not worry about the event handler yet.

2. In the JS panel, write two *doColor()* functions in the JS panel (e.g., *doRed()* and *doBlue()*). For each one:

- **Get the left canvas element**.
- **Then use the *style.backgroundColor* property to change the background color**. Previously you changed the color by changing the CSS class; this is an alternative way to change a color property.

3. Have your two new buttons each the appropriate *doColor()* functions.

Add Shapes and Text to Canvas

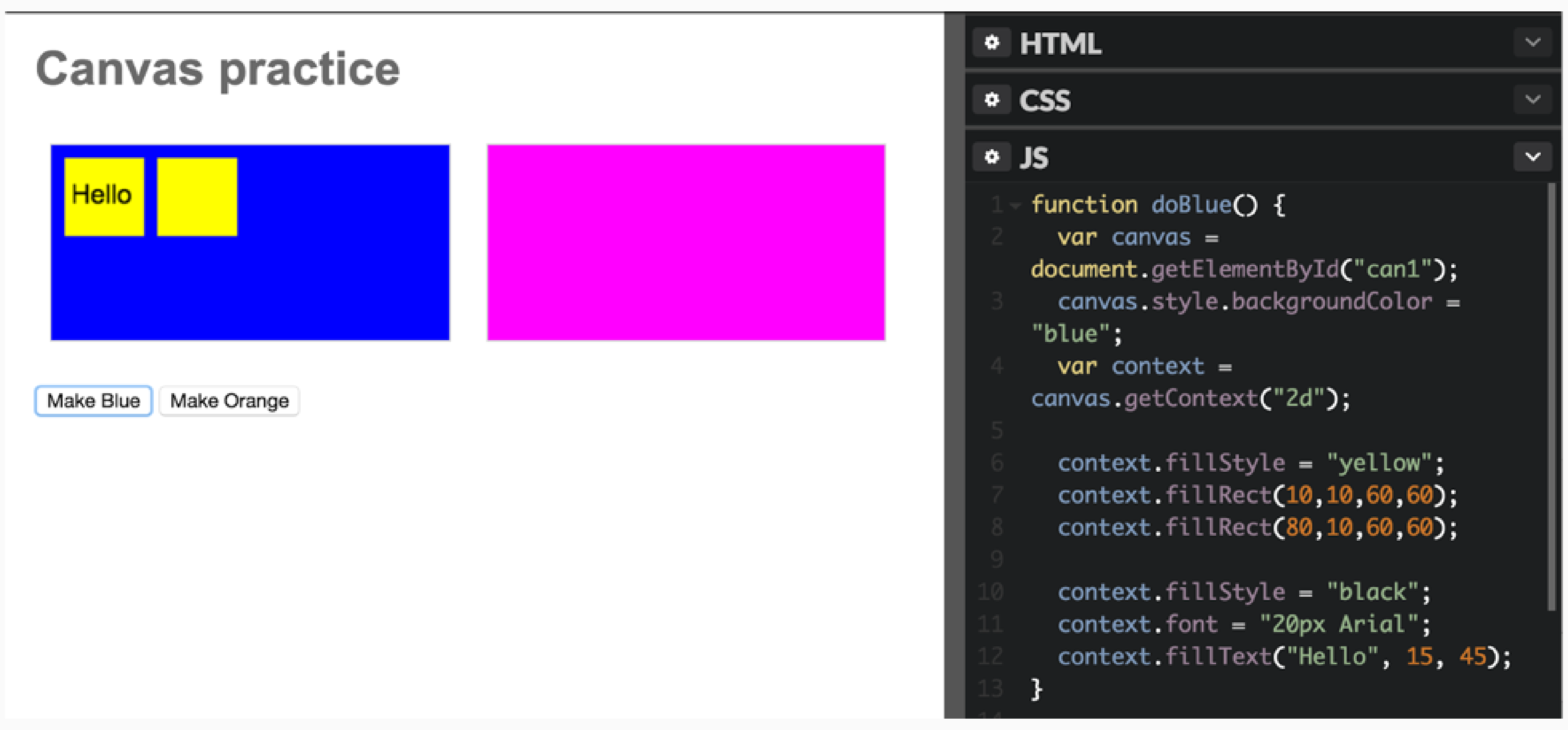
In order to draw shapes or write text on a canvas, you need to get its context. In this exercise, choose one of your **doColor()** functions and follow these steps to add functionality.

1. Use the *canvas.getContext("2d")* method to get the context of the canvas.

2. Draw a rectangle. Remember to use the context.fillStyle and context.fillRect methods. You can determine the color and size of your rectangles.

3. Try adding text to your canvas. You will need to use the context.fillStyle, context.font, and context.fillText methods.

Need help? Review the **Using HTML5 Canvas** video, see the below example, and go to the forums.



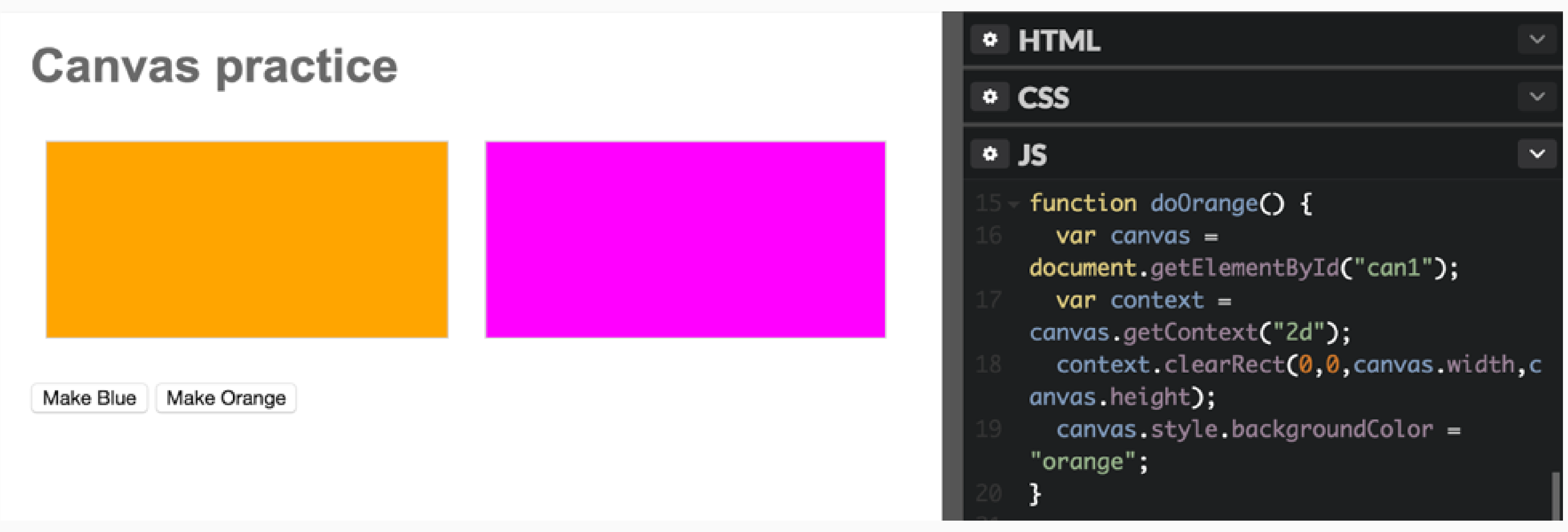
```javascript
function doBlue() {
  var canvas =
document.getElementById("can1");
  canvas.style.backgroundColor =
"blue";
  var context =
canvas.getContext("2d");

  context.fillStyle = "yellow";
  context.fillRect(10,10,60,60);
  context.fillRect(80,10,60,60);

  context.fillStyle = "black";
  context.font = "20px Arial";
  context.fillText("Hello", 15, 45);
}
```

Extra Challenge!

1. See what happens when you use the second button to change the background color—the background color changes, but the rectangles and text remain in the canvas.

2. Add functionality to your second **doColor()** function so that the canvas context is cleared. Get the context, and use the **context.clearRect()** method to do this. See below example.

3. If you would like to experiment with other canvas drawing capabilities, here is a resource: http://www.w3schools.com/html/html5_canvas.asp.



```javascript
function doOrange() {
  var canvas =
document.getElementById("can1");
  var context =
canvas.getContext("2d");
  context.clearRect(0,0,canvas.width,c
anvas.height);
  canvas.style.backgroundColor =
"orange";
}
```

Mark as completed