Name: Sourav Shailesh Toshniwal
Class: TY CSE-8 AIEC-1
Batch: A
Roll no: 2213047
Serial no: 6

Assignment 10

Title: Write a JQuery code to give animation to div on click of button.

Theory:
To implement an animation in jQuery when a button is clicked, you'll need to understand some basic jQuery concepts. Here's a theoretical overview of what you'll need to know:

1. Including jQuery: First, make sure you include the jQuery library in your HTML file. You can do this by adding a `<script>` tag that links to the jQuery library hosted on a content delivery network (CDN) or by downloading and hosting it locally.

2. Selecting Elements: jQuery allows you to select HTML elements using CSS-style selectors. You'll need to select the button element and the div element that you want to animate.

3. Binding Events: You'll use the `.on()` method or other event binding methods to attach a click event handler to the button. This handler function will be executed when the button is clicked.

4. Animating Elements: You can use jQuery's animation functions, such as `.animate()`, to create animations on the selected div element. You'll specify the properties to animate (e.g., `width`, `height`, `opacity`, etc.), the duration of the animation, and any easing effects.

5. Event Handling: Within your event handler function, you'll write code to trigger the animation on the div element when the button is clicked. This can involve modifying CSS properties or using jQuery's animation functions.

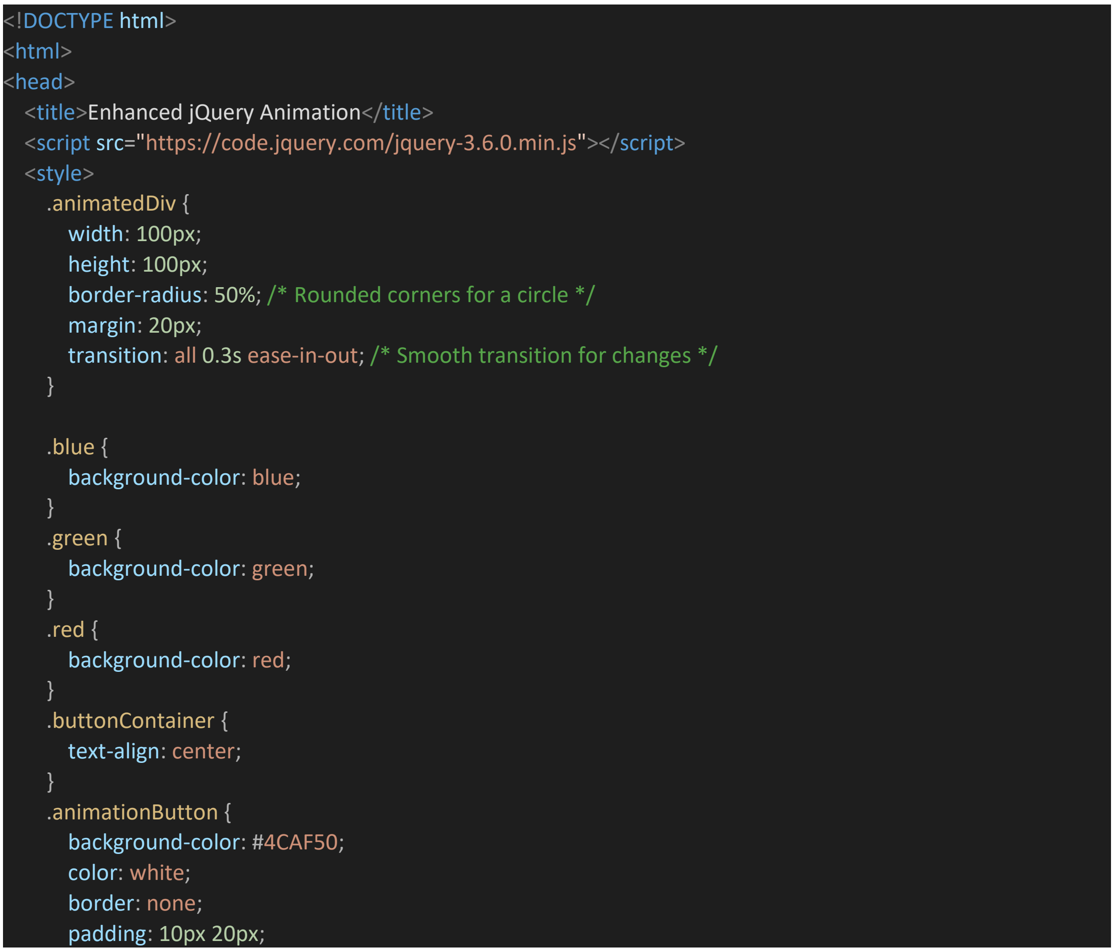
6. Callback Functions: You can also use callback functions to execute code after the animation is complete. For example, you might want to reset the properties of the animated div to their original values or perform other actions.

7. Document Ready: It's a good practice to enclose your jQuery code inside a `$(document).ready()` function to ensure that the DOM is fully loaded before your code runs.

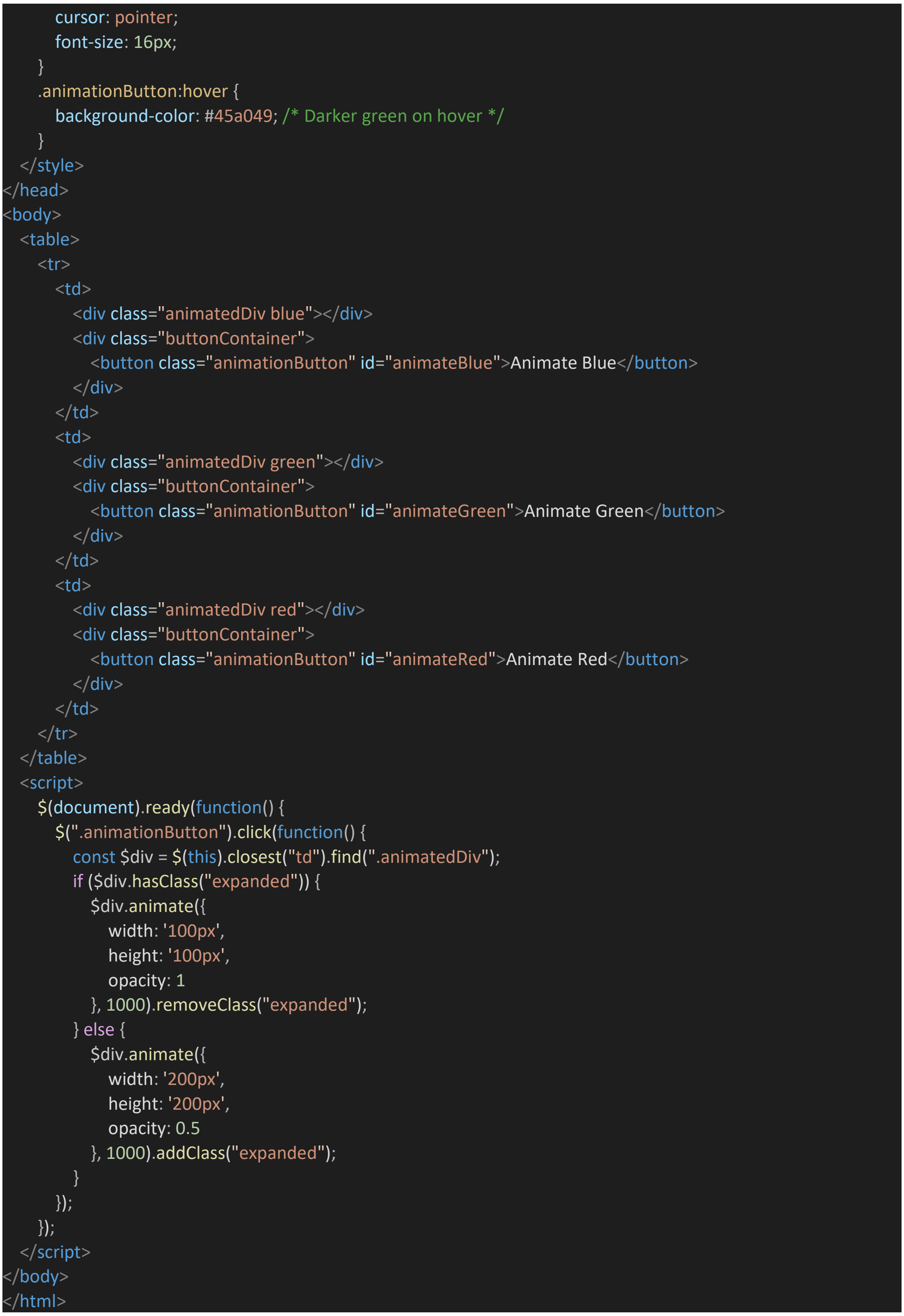
8. Optional: CSS Styles: You can use CSS styles to define initial properties of the div, which will be modified during the animation.

Remember that this is just a theoretical overview of what you'll need to understand. When implementing the code, you'll need to combine these concepts and write the actual jQuery code to achieve the desired animation effect. If you need help with a specific aspect or have any questions, feel free to ask for further guidance.

Code:

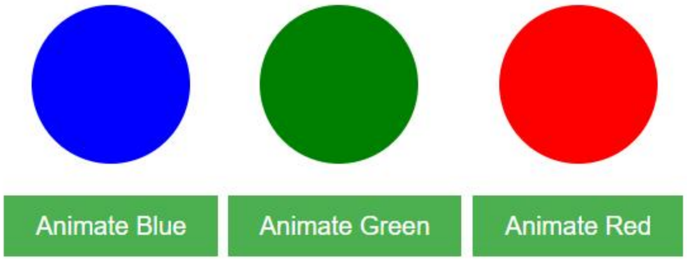


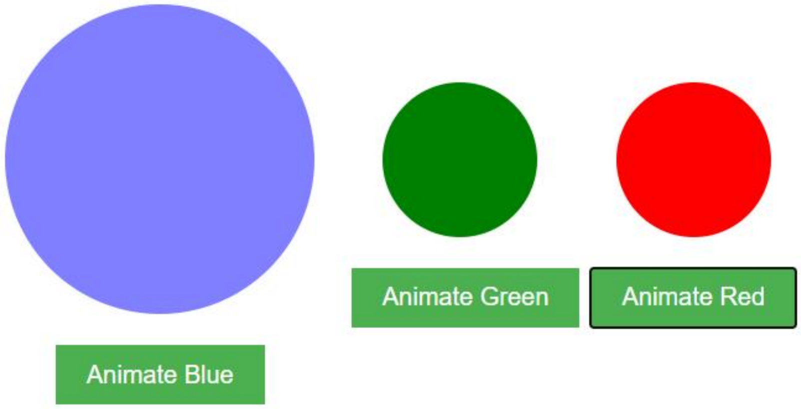
```
<!DOCTYPE html>
<html>
<head>
  <title>Enhanced jQuery Animation</title>
  <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
  <style>
    .animatedDiv {
      width: 100px;
      height: 100px;
      border-radius: 50%; /* Rounded corners for a circle */
      margin: 20px;
      transition: all 0.3s ease-in-out; /* Smooth transition for changes */
    }

    .blue {
      background-color: blue;
    }
    .green {
      background-color: green;
    }
    .red {
      background-color: red;
    }
    .buttonContainer {
      text-align: center;
    }
    .animationButton {
      background-color: #4CAF50;
      color: white;
      border: none;
      padding: 10px 20px;
```



```
      cursor: pointer;
      font-size: 16px;
    }
    .animationButton:hover {
      background-color: #45a049; /* Darker green on hover */
    }
  </style>
</head>
<body>
  <table>
    <tr>
      <td>
        <div class="animatedDiv blue"></div>
        <div class="buttonContainer">
          <button class="animationButton" id="animateBlue">Animate Blue</button>
        </div>
      </td>
      <td>
        <div class="animatedDiv green"></div>
        <div class="buttonContainer">
          <button class="animationButton" id="animateGreen">Animate Green</button>
        </div>
      </td>
      <td>
        <div class="animatedDiv red"></div>
        <div class="buttonContainer">
          <button class="animationButton" id="animateRed">Animate Red</button>
        </div>
      </td>
    </tr>
  </table>
  <script>
    $(document).ready(function() {
      $(".animationButton").click(function() {
        const $div = $(this).closest("td").find(".animatedDiv");
        if ($div.hasClass("expanded")) {
          $div.animate({
            width: '100px',
            height: '100px',
            opacity: 1
          }, 1000).removeClass("expanded");
        } else {
          $div.animate({
            width: '200px',
            height: '200px',
            opacity: 0.5
          }, 1000).addClass("expanded");
        }
      });
    });
  </script>
</body>
</html>
```

Output:





Conclusion: Conclusively, by applying the principles of jQuery, we can create interactive and engaging web experiences. In this experiment, we have leveraged jQuery's event handling and animation capabilities to trigger dynamic effects when a button is clicked. Understanding how to select elements, bind events, and implement animations is fundamental to enhancing user interactivity and visual appeal in web development. This experiment demonstrates the potential of jQuery for creating compelling user interfaces through efficient and concise JavaScript code.