JQuery Assignment

Q1 What is jQuery?

Ans: jQuery is a popular, fast, and concise JavaScript library that simplifies many tasks and interactions with HTML documents, event handling, animations, and AJAX interactions for web development. It aims to ease the complexities of JavaScript by providing an easy-to-use API that abstracts and simplifies common tasks.

Q2. How to Apply CSS Using JQuery, How to Add Class and Remove Class in Jquery, JQuery Animation?

Ans: **Apply CSS using jQuery:**

You can use jQuery to apply CSS styles to HTML elements. The `css()` function in jQuery allows you to set or get CSS properties of selected elements.

Expl…

```
// Set CSS properties
$('#elementID').css({
    'property1': 'value1',
    'property2': 'value2'
});


// Get CSS properties
```

```
var propValue = $('#elementID').css('property');
```

Add and Remove Classes in jQuery:

jQuery makes it easy to add or remove classes from HTML elements using `addClass()` and `removeClass()` functions.

```
// Add a class
$('#elementID').addClass('className');


// Remove a class
$('#elementID').removeClass('className');
```

jQuery Animation:

jQuery simplifies animation on HTML elements. You can use functions like `animate()`, `fadeIn()`, `fadeOut()`, `slideUp()`, `slideDown()`, etc., to create animations.

```
// Animating an element
$('#elementID').animate({
    'property1': 'value1',
    'property2': 'value2'
}, duration);


// Fade In
$('#elementID').fadeIn(duration);


// Fade Out
$('#elementID').fadeOut(duration);
```

```
// Slide Up
$('#elementID').slideUp(duration);


// Slide Down
$('#elementID').slideDown(duration);
```

Replace `'elementID'`, `'className'`, and other placeholders with the appropriate element IDs, classes, or properties you want to manipulate.

Remember to include the jQuery library in your HTML file before using these jQuery functions:

```
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js
"></script>
```

Q3. How to create a slider with animation?

Ans: Creating a slider with animation typically involves HTML, CSS, and JavaScript/jQuery. Here's a simple example using jQuery for a basic image slider:

HTML Structure:

```
<div class="slider">
  <div class="slides">
    <img src="image1.jpg" alt="Image 1">
    <img src="image2.jpg" alt="Image 2">
    <img src="image3.jpg" alt="Image 3">
    <!-- Add more images as needed -->
  </div>
</div>
```

CSS (Basic Styling):

```css
.slider {
   width: 100%;
   overflow: hidden;
   /* Set height and other styles as needed */
}

.slides {
   display: flex; /* Make the images flexibly displayed */
   transition: transform 0.5s ease; /* Animation transition */
}

.slides img {
   width: 100%; /* Ensure images take full width of the slider */
}
```

JavaScript (jQuery for Animation):

```javascript
$(document).ready(function() {
   const slider = $('.slides');
   const slides = $('.slides img');
   const slideWidth = slides.width();
   let currentSlide = 0;

   function slide() {
       slider.animate({
           'margin-left': -slideWidth * currentSlide
       }, 500);
   }

   // Automatically move to the next slide every few seconds
   setInterval(function() {
       currentSlide = (currentSlide + 1) % slides.length;
       slide();
   }, 3000); // Change slide every 3 seconds (adjust as needed)
});
```

This code assumes you have a container (`.slider`) with a fixed width that clips the images and a set of images (`img` tags) inside the slider. The JavaScript code calculates the width of each slide and uses jQuery's `animate()` function to shift the slides left by the slide width to create the sliding effect.

Q4 Event bubbling tickling example

Ans:
```
<div id="outer" style="padding: 20px; border: 1px solid black;">
    <div id="middle" style="padding: 20px; border: 1px solid red;">
        <div id="inner" style="padding: 20px; border: 1px solid blue;">
            Click me!
        </div>
    </div>
</div>
```

jQuery JavaScript:

```
$(document).ready(function() {
  $("#outer").on("click", function(event) {
    console.log("Outer div clicked");
    console.log("Event target: " + event.target.id);
  });

  $("#middle").on("click", function(event) {
    console.log("Middle div clicked");
    console.log("Event target: " + event.target.id);
  });

  $("#inner").on("click", function(event) {
    console.log("Inner div clicked");
    console.log("Event target: " + event.target.id);
  });
});
```

In this example:

- There are three nested `div` elements: `outer`, `middle`, and `inner`.
- Each `div` has its own click event handler attached using jQuery's `on()` method.
- When you click on the innermost `div` (`#inner`), the event bubbles up through its parent elements (`#middle` and `#outer`).

If you click on the innermost `div`, you'll see in the console that all three event handlers are triggered due to event bubbling. The innermost element's event triggers first, followed by its parent elements in the order of nesting.

This example demonstrates how events bubble up through nested elements in jQuery, allowing you to handle the event at various levels of the DOM hierarchy.