Exercise 1: JavaScript Debugging

Problem Statement:

You've been given a simple JavaScript code snippet that's intended to toggle the visibility of an element when a button is clicked. However, it's not working as expected.

Code:  
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Toggle Element</title>

</head>

<body>

    <button onclick="toggleElement()">Toggle Element</button>

    <div id="target" style="display: none;">This is the target element.</div>

    <script>

        function toggleElement() {

            var element = document.getElementById("target");

            element.style.display = (element.style.display === "none") ? "block" : "none";

        }

    </script>

</body>

</html>

Tasks:

Identify the issue in the provided JavaScript code.

Debug and fix the code so that clicking the button toggles the visibility of the element.

The JavaScript code you provided seems correct. There is a small issue, with how the display property is being toggled. Currently the code only checks if the display value is "none". Sets it to "block" in that case. However it doesn't account for values like "inline" or "flex".

To fix this issue you can modify the code to toggle the display property between "none" and an empty string. This will cause it to revert back to the elements default display style. Here's the updated code.

Corrected Code :

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Toggle Element</title>

</head>

<body>

<button onclick="toggleElement()">Toggle Element</button>

<div id="target" style="display: none;">This is the target element.</div>

<script>

function toggleElement() {

var element = document.getElementById("target");

element.style.display = (element.style.display === "none") ? "" : "none";

}

</script>

</body>

</html>

Exercise 2: CSS Troubleshooting

Problem Statement:

You've been given an HTML and CSS code snippet that's supposed to create a centered, responsive container. However, it's not displaying as expected.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Centered Container</title>

<style>

.container {

margin: auto;

width: 50%;

background-color: #f0f0f0;

padding: 20px;

}

</style>

</head>

<body>

<div class="container">

<h1>Centered Container</h1>

<p>This container should be centered on the page.</p>

</div>

</body>

</html>

Tasks:

Identify the issue in the provided CSS code.

Debug and fix the code so that the container is centered on the page.

I made some modifications to the code:

I applied CSS styles to the <body> element in order to center its content using flexbox. To achieve this I set the properties display; justify content; center and align items; center. Additionally I added the height; 100vh property to ensure that the body covers the height of the viewport.

To simplify things I removed the margin; auto; from the.container class since it is not needed when using flexbox for centering.

With these adjustments, in place you can expect the container to be centered both horizontally and vertically on your page resulting in a aligned and responsive layout.

Corrected code :

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Centered Container</title>

<style>

body {

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

margin: 0;

}

.container {

width: 50%;

background-color: #f0f0f0;

padding: 20px;

text-align: center;

}

</style>

</head>

<body>

<div class="container">

<h1>Centered Container</h1>

<p>This container should be centered on the page.</p>

</div>

</body>

</html>