

## DAY31:

**ASSIGNMENT2:::** Use regular expressions in JavaScript to validate the email address entered in the form. It should check for the correct format of the email and display a message to the user if the format is incorrect.

## ANSWER::

Can use a regular expression to validate email addresses in JavaScript. Here's a simple example of how you can do it:

## Javascript code::

```
function validateEmail(email) {  
    // Regular expression for validating email addresses  
    const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;  
    return emailRegex.test(email);  
}
```

```
// Example usage  
const email = "example@example.com";  
if (validateEmail(email)) {  
    console.log("Email is valid.");  
} else {  
    console.log("Please enter a valid email address.");  
}
```

In this code:

`^[^\s@]+@[^\s@]+\.[^\s@]+$`: This regular expression is used to validate the email address format. Here's what it does:

`^`: Start of the string.

`[^\s@]+`: Matches one or more characters that are not whitespace or '@'.

`@`: Matches the '@' character.

`[^\s@]+`: Matches one or more characters that are not whitespace or '@'.

`\.`: Matches the '.' character. (Note that we escape it with a backslash because '.' has a special meaning in regular expressions.)

`[^\s@]+`: Matches one or more characters that are not whitespace or '@'.

`$`: End of the string.

`emailRegex.test(email)`: This line tests whether the provided email matches the regular expression. If it does, it returns true, indicating that the email is valid; otherwise, it returns false.

You can integrate this function into your form validation logic to display a message to the user if the email format is incorrect.

## HTML CODE::

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Email Validation</title>
  <style>
    .error {
      color: red;
    }
  </style>
</head>
<body>
  <form id="emailForm">
    <label for="email">Email:</label>
```

```
<input type="email" id="email" name="email">

<span id="emailError" class="error"></span>

<br>

<button type="submit">Submit</button>

</form>


<script>
function validateEmail(email) {
    const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
    return emailRegex.test(email);
}

document.getElementById("emailForm").addEventListener("submit", function(event) {
    event.preventDefault(); // Prevent form submission

    const emailInput = document.getElementById("email");
    const emailError = document.getElementById("emailError");
    const email = emailInput.value.trim();

    if (validateEmail(email)) {
        // Email is valid, do whatever you need to do
        console.log("Email is valid.");
    } else {
        // Email is invalid, display error message
        emailError.textContent = "Please enter a valid email address.";
        emailInput.focus(); // Set focus back to email input field
    }
});
</script>
</body>
</html>
```

## In this HTML code:

- We have a simple form with an email input field and a submit button.
- We've added an empty `<span>` element with the id "emailError" to display the error message if the email format is incorrect.
- We've added a `<style>` block to define the appearance of the error message.
- At the end of the HTML code, we have a `<script>` block where we define the `validateEmail` function and attach a submit event listener to the form.
- Inside the event listener function, we prevent the default form submission behavior using `event.preventDefault()`.
- We then retrieve the value of the email input field, trim any leading or trailing whitespace, and validate it using the `validateEmail` function.
- If the email is valid, you can perform any necessary actions. Otherwise, we set the error message in the `<span>` element and set focus back to the email input field.