Aim: Live Character Counter for Textarea.

Theory: The live character counter is a user experience (UX) feature based on the principle of **Event-Driven Programming**. Its goal is to provide immediate, real-time feedback to the user.

Code:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Live Character Counter</title>

    <style>

        body {

            font-family: Arial, sans-serif;

            max-width: 800px;

            margin: 50px auto;

            padding: 20px;

            background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

            min-height: 100vh;

        }

        .container {

            background: linear-gradient(135deg, #ffffff 0%, #f8f9ff 100%);

            padding: 30px;

            border-radius: 15px;

            box-shadow: 0 8px 32px rgba(0, 0, 0, 0.1);

            backdrop-filter: blur(10px);

            border: 1px solid rgba(255, 255, 255, 0.2);

        }

        h1 {

            text-align: center;

            background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

            -webkit-background-clip: text;

            -webkit-text-fill-color: transparent;

            background-clip: text;

            margin-bottom: 30px;

            font-size: 2.5em;

            font-weight: bold;

        }

        #textInput {

            width: 100%;

            height: 200px;

            padding: 15px;

            border: 2px solid transparent;

            background: linear-gradient(white, white) padding-box,

                       linear-gradient(135deg, #667eea, #764ba2) border-box;

            border-radius: 12px;

            font-size: 16px;

            line-height: 1.5;

            resize: vertical;

            box-sizing: border-box;

            font-family: inherit;

            transition: all 0.3s ease;

        }

        #textInput:focus {

            border: 2px solid transparent;

            background: linear-gradient(white, white) padding-box,

                       linear-gradient(135deg, #667eea, #764ba2) border-box;

            outline: none;

            box-shadow: 0 0 20px rgba(102, 126, 234, 0.3);

            transform: translateY(-2px);

        }

        .counter-section {

            margin-top: 20px;

            padding: 20px;

            background: linear-gradient(135deg, rgba(102, 126, 234, 0.1) 0%, rgba(118, 75, 162, 0.1) 100%);

            border-radius: 12px;

            border: 1px solid rgba(102, 126, 234, 0.2);

            backdrop-filter: blur(5px);

        }

        .counter-display {

            display: flex;

            justify-content: space-between;

            align-items: center;

            flex-wrap: wrap;

            gap: 15px;

        }

        .counter-item {

            display: flex;

            flex-direction: column;

            align-items: center;

            min-width: 120px;

        }

        .counter-label {

            font-size: 14px;

            color: #666;

            margin-bottom: 5px;

        }

        .counter-value {

            font-size: 24px;

            font-weight: bold;

            background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

            -webkit-background-clip: text;

            -webkit-text-fill-color: transparent;

            background-clip: text;

        }

        .progress-bar {

            width: 100%;

            height: 10px;

            background: linear-gradient(90deg, rgba(102, 126, 234, 0.2) 0%, rgba(118, 75, 162, 0.2) 100%);

            border-radius: 6px;

            margin-top: 15px;

            overflow: hidden;

            border: 1px solid rgba(102, 126, 234, 0.1);

        }

        .progress-fill {

            height: 100%;

            background: linear-gradient(90deg, #667eea 0%, #764ba2 100%);

            transition: width 0.3s ease;

            border-radius: 6px;

            box-shadow: 0 2px 8px rgba(102, 126, 234, 0.3);

        }

        .stats {

            display: grid;

            grid-template-columns: repeat(auto-fit, minmax(150px, 1fr));

            gap: 15px;

            margin-top: 20px;

        }

        .stat-card {

            background: linear-gradient(135deg, rgba(255, 255, 255, 0.9) 0%, rgba(248, 249, 255, 0.9) 100%);

            padding: 20px;

            border-radius: 10px;

            text-align: center;

            border: 1px solid rgba(102, 126, 234, 0.2);

            backdrop-filter: blur(5px);

            transition: transform 0.3s ease, box-shadow 0.3s ease;

        }

        .stat-card:hover {

            transform: translateY(-5px);

            box-shadow: 0 10px 25px rgba(102, 126, 234, 0.2);

        }

        .stat-number {

            font-size: 24px;

            font-weight: bold;

            background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

            -webkit-background-clip: text;

            -webkit-text-fill-color: transparent;

            background-clip: text;

        }

        .stat-label {

            font-size: 12px;

            color: #666;

            text-transform: uppercase;

            margin-top: 5px;

        }

        @media (max-width: 600px) {

            .counter-display {

                flex-direction: column;

            }

            .stats {

                grid-template-columns: 1fr;

            }

        }

    </style>

</head>

<body>

    <div class="container">

        <h1>Live Character Counter</h1>

        <textarea

            id="textInput"

            placeholder="Start typing here to see the live character count..."

            maxlength="1000"

        ></textarea>

        <div class="counter-section">

            <div class="counter-display">

                <div class="counter-item">

                    <div class="counter-label">Characters</div>

                    <div class="counter-value" id="charCount">0</div>

                </div>

                <div class="counter-item">

                    <div class="counter-label">Remaining</div>

                    <div class="counter-value" id="remaining">1000</div>

                </div>

                <div class="counter-item">

                    <div class="counter-label">Words</div>

                    <div class="counter-value" id="wordCount">0</div>

                </div>

            </div>

            <div class="progress-bar">

                <div class="progress-fill" id="progressBar"></div>

            </div>

        </div>

        <div class="stats">

            <div class="stat-card">

                <div class="stat-number" id="paragraphs">0</div>

                <div class="stat-label">Paragraphs</div>

            </div>

            <div class="stat-card">

                <div class="stat-number" id="sentences">0</div>

                <div class="stat-label">Sentences</div>

            </div>

            <div class="stat-card">

                <div class="stat-number" id="avgWordLength">0</div>

                <div class="stat-label">Avg Word Length</div>

            </div>

            <div class="stat-card">

                <div class="stat-number" id="readingTime">0</div>

                <div class="stat-label">Reading Time (min)</div>

            </div>

        </div>

    </div>

    <script>

        // Get DOM elements

        const textInput = document.getElementById('textInput');

        const charCount = document.getElementById('charCount');

        const remaining = document.getElementById('remaining');

        const wordCount = document.getElementById('wordCount');

        const progressBar = document.getElementById('progressBar');

        const paragraphs = document.getElementById('paragraphs');

        const sentences = document.getElementById('sentences');

        const avgWordLength = document.getElementById('avgWordLength');

        const readingTime = document.getElementById('readingTime');

        const maxLength = 1000;

        // Function to update all counters

        function updateCounters() {

            const text = textInput.value;

            const charLength = text.length;

            // Update character count

            charCount.textContent = charLength;

            // Update remaining characters

            const remainingChars = maxLength - charLength;

            remaining.textContent = remainingChars;

            // Update progress bar

            const progressPercentage = (charLength / maxLength) \* 100;

            progressBar.style.width = progressPercentage + '%';

            // Change color based on remaining characters

            if (remainingChars < 50) {

                progressBar.style.background = 'linear-gradient(90deg, #ff6b6b 0%, #ee5a52 100%)';

                progressBar.style.boxShadow = '0 2px 8px rgba(255, 107, 107, 0.3)';

                remaining.style.background = 'linear-gradient(135deg, #ff6b6b 0%, #ee5a52 100%)';

                remaining.style.webkitBackgroundClip = 'text';

                remaining.style.webkitTextFillColor = 'transparent';

                remaining.style.backgroundClip = 'text';

            } else if (remainingChars < 200) {

                progressBar.style.background = 'linear-gradient(90deg, #ffa726 0%, #ff9800 100%)';

                progressBar.style.boxShadow = '0 2px 8px rgba(255, 167, 38, 0.3)';

                remaining.style.background = 'linear-gradient(135deg, #ffa726 0%, #ff9800 100%)';

                remaining.style.webkitBackgroundClip = 'text';

                remaining.style.webkitTextFillColor = 'transparent';

                remaining.style.backgroundClip = 'text';

            } else {

                progressBar.style.background = 'linear-gradient(90deg, #667eea 0%, #764ba2 100%)';

                progressBar.style.boxShadow = '0 2px 8px rgba(102, 126, 234, 0.3)';

                remaining.style.background = 'linear-gradient(135deg, #667eea 0%, #764ba2 100%)';

                remaining.style.webkitBackgroundClip = 'text';

                remaining.style.webkitTextFillColor = 'transparent';

                remaining.style.backgroundClip = 'text';

            }

            // Count words (split by whitespace and filter empty strings)

            const words = text.trim().split(/\s+/).filter(word => word.length > 0);

            const wordCountValue = text.trim() === '' ? 0 : words.length;

            wordCount.textContent = wordCountValue;

            // Count paragraphs (split by double newlines or single newlines)

            const paragraphCount = text.trim() === '' ? 0 :

                text.split(/\n\s\*\n/).filter(p => p.trim().length > 0).length;

            paragraphs.textContent = paragraphCount;

            // Count sentences (simple sentence detection)

            const sentenceCount = text.trim() === '' ? 0 :

                text.split(/[.!?]+/).filter(s => s.trim().length > 0).length;

            sentences.textContent = sentenceCount;

            // Calculate average word length

            if (wordCountValue > 0) {

                const totalWordLength = words.reduce((sum, word) => sum + word.length, 0);

                const avgLength = Math.round(totalWordLength / wordCountValue \* 10) / 10;

                avgWordLength.textContent = avgLength;

            } else {

                avgWordLength.textContent = '0';

            }

            // Calculate estimated reading time (average 200 words per minute)

            const readingTimeMinutes = Math.ceil(wordCountValue / 200);

            readingTime.textContent = readingTimeMinutes;

        }

        // Add event listeners for real-time updates

        textInput.addEventListener('input', updateCounters);

        textInput.addEventListener('paste', function() {

            // Small delay to ensure pasted content is processed

            setTimeout(updateCounters, 10);

        });

        // Initialize counters on page load

        updateCounters();

        // Optional: Add keyboard shortcuts info

        textInput.addEventListener('keydown', function(e) {

            // Ctrl+A to select all

            if (e.ctrlKey && e.key === 'a') {

                setTimeout(updateCounters, 10);

            }

        });

    </script>

</body>

</html>

Output:

