<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Simple Calculator</title>

<style>

body {

display: flex;

justify-content: center;

align-items: center;

min-height: 100vh;

background-color: #2c3e50;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

margin: 0;

padding: 20px;

}

.calculator {

background-color: #34495e;

border-radius: 15px;

padding: 20px;

box-shadow: 0 10px 20px rgba(0, 0, 0, 0.3);

max-width: 320px;

width: 100%;

}

.display {

background-color: #233140;

color: #ecf0f1;

padding: 20px;

border-radius: 10px;

text-align: right;

font-size: 2.5em;

margin-bottom: 20px;

word-wrap: break-word;

word-break: break-all;

min-height: 50px;

overflow: hidden;

}

.buttons-grid {

display: grid;

grid-template-columns: repeat(4, 1fr);

gap: 12px;

}

.btn {

background-color: #466786;

color: #ecf0f1;

border: none;

padding: 20px;

font-size: 1.5em;

cursor: pointer;

border-radius: 10px;

transition: background-color 0.2s, transform 0.1s;

touch-action: manipulation; /\* For better mobile responsiveness \*/

}

.btn:hover {

background-color: #5a7b9a;

}

.btn:active {

transform: scale(0.95);

}

.btn.operator {

background-color: #f39c12;

}

.btn.operator:hover {

background-color: #e67e22;

}

.btn.equals {

grid-column: span 2;

background-color: #2ecc71;

}

.btn.equals:hover {

background-color: #27ae60;

}

.btn.clear {

background-color: #e74c3c;

}

.btn.clear:hover {

background-color: #c0392b;

}

@media (max-width: 400px) {

.display {

font-size: 2em;

}

.btn {

font-size: 1.2em;

}

}

</style>

</head>

<body>

<div class="calculator">

<div class="display" id="display">0</div>

<div class="buttons-grid">

<button class="btn clear" data-action="clear">AC</button>

<button class="btn operator" data-action="negate">+/-</button>

<button class="btn operator" data-action="percent">%</button>

<button class="btn operator" data-action="divide">÷</button>

<button class="btn" data-value="7">7</button>

<button class="btn" data-value="8">8</button>

<button class="btn" data-value="9">9</button>

<button class="btn operator" data-action="multiply">×</button>

<button class="btn" data-value="4">4</button>

<button class="btn" data-value="5">5</button>

<button class="btn" data-value="6">6</button>

<button class="btn operator" data-action="subtract">-</button>

<button class="btn" data-value="1">1</button>

<button class="btn" data-value="2">2</button>

<button class="btn" data-value="3">3</button>

<button class="btn operator" data-action="add">+</button>

<button class="btn" data-value="0">0</button>

<button class="btn" data-value=".">.</button>

<button class="btn equals" data-action="equals">=</button>

</div>

</div>

<script>

document.addEventListener('DOMContentLoaded', () => {

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

const buttons = document.querySelectorAll('.btn');

let currentInput = '0';

let firstOperand = null;

let operator = null;

let waitingForSecondOperand = false;

// Function to reset all state variables

function resetCalculator() {

currentInput = '0';

firstOperand = null;

operator = null;

waitingForSecondOperand = false;

}

// Function to update the display

function updateDisplay() {

display.textContent = currentInput;

}

// Handle number and decimal input

function handleNumber(value) {

if (waitingForSecondOperand === true) {

currentInput = value;

waitingForSecondOperand = false;

} else {

currentInput = currentInput === '0' ? value : currentInput + value;

}

updateDisplay();

}

// Handle operator input

function handleOperator(nextOperator) {

const inputValue = parseFloat(currentInput);

if (operator && waitingForSecondOperand) {

operator = nextOperator;

return;

}

if (firstOperand === null) {

firstOperand = inputValue;

} else if (operator) {

const result = performCalculation[operator](firstOperand, inputValue);

currentInput = String(result);

firstOperand = result;

}

waitingForSecondOperand = true;

operator = nextOperator;

updateDisplay();

}

// Perform the calculation

const performCalculation = {

'divide': (first, second) => first / second,

'multiply': (first, second) => first \* second,

'add': (first, second) => first + second,

'subtract': (first, second) => first - second

};

// Handle special actions (clear, equals, percent, negate)

function handleAction(action) {

switch (action) {

case 'clear':

resetCalculator();

break;

case 'equals':

if (operator && !waitingForSecondOperand) {

const inputValue = parseFloat(currentInput);

currentInput = String(performCalculation[operator](firstOperand, inputValue));

firstOperand = null;

operator = null;

waitingForSecondOperand = false;

}

break;

case 'percent':

currentInput = String(parseFloat(currentInput) / 100);

break;

case 'negate':

currentInput = String(parseFloat(currentInput) \* -1);

break;

}

updateDisplay();

}

// Event listener for all calculator buttons

buttons.forEach(button => {

button.addEventListener('click', (event) => {

const { value, action } = event.target.dataset;

if (value) {

handleNumber(value);

} else if (action) {

if (action === 'equals' || action === 'clear' || action === 'percent' || action === 'negate') {

handleAction(action);

} else {

handleOperator(action);

}

}

});

});

// Initial display update

updateDisplay();

});

</script>

</body>

</html>