UNIVERZÁLNÍ DEBUG CONSOLE - COMPLETE SETUP GUIDE

Přidej na jakoukoli stránku v BaseCamp projektu

KROK 1: Vytvoř `debugConsole.js`

Vytvoř nový soubor: `/src/utils/debugConsole.js`

javascript

```
// debugConsole.js - Universal Debug Console for BaseCamp
// Usage: import './debugConsole.js' at the top of any JS file

let debugLogs = [];
let isInitialized = false;

// ========================================
// CREATE DEBUG CONSOLE UI
// ========================================
export function createDebugConsole() {
    if (isInitialized) return;
    isInitialized = true;

    const debugHTML = `
        <div id="debug-console" style="
            position: fixed;
            bottom: 20px;
            right: 20px;
            width: 450px;
            max-height: 600px;
            background: #1e293b;
            border: 2px solid #0052FF;
            border-radius: 12px;
            box-shadow: 0 20px 40px rgba(0,82,255,0.3);
            z-index: 999999;
            font-family: 'Courier New', monospace;
            font-size: 11px;
```

```
            display: none;
            flex-direction: column;
        ">
            <div id="debug-header" style="
                background: linear-gradient(135deg, #0052FF 0%,
#0041CC 100%);
                color: white;
                padding: 12px 16px;
                cursor: move;
                user-select: none;
                display: flex;
                justify-content: space-between;
                align-items: center;
                border-radius: 10px 10px 0 0;
            ">
                <strong style="font-size: 13px;">🐛 BaseCamp Debug
Console</strong>
                <div style="display: flex; gap: 8px;">
                    <button onclick="window.exportDebugLogs()"
style="
                        background: #10b981;
                        border: none;
                        color: white;
                        padding: 4px 10px;
                        border-radius: 4px;
                        cursor: pointer;
                        font-size: 10px;
                        font-weight: bold;
                    ">Export</button>
                    <button onclick="window.clearDebugConsole()"
style="
                        background: #ef4444;
                        border: none;
                        color: white;
                        padding: 4px 10px;
                        border-radius: 4px;
                        cursor: pointer;
                        font-size: 10px;
                        font-weight: bold;
                    ">Clear</button>
                </div>
```

```
            </div>
            <div id="debug-tabs" style="
                display: flex;
                background: #334155;
                border-bottom: 1px solid #475569;
            ">
                <button class="debug-tab active" data-tab="all"
style="
                    flex: 1;
                    padding: 10px;
                    background: none;
                    border: none;
                    color: white;
                    cursor: pointer;
                    font-weight: 600;
                    border-bottom: 2px solid #0052FF;
                ">All</button>
                <button class="debug-tab" data-tab="info" style="
                    flex: 1;
                    padding: 10px;
                    background: none;
                    border: none;
                    color: #94a3b8;
                    cursor: pointer;
                ">Info</button>
                <button class="debug-tab" data-tab="success" style="
                    flex: 1;
                    padding: 10px;
                    background: none;
                    border: none;
                    color: #94a3b8;
                    cursor: pointer;
                ">Success</button>
                <button class="debug-tab" data-tab="warn" style="
                    flex: 1;
                    padding: 10px;
                    background: none;
                    border: none;
                    color: #94a3b8;
                    cursor: pointer;
                ">Warn</button>
```

```
                <button class="debug-tab" data-tab="error" style="
                    flex: 1;
                    padding: 10px;
                    background: none;
                    border: none;
                    color: #94a3b8;
                    cursor: pointer;
                ">Error</button>
            </div>
            <div id="debug-logs" style="
                padding: 12px;
                max-height: 450px;
                overflow-y: auto;
                background: #0f172a;
                color: #e2e8f0;
                border-radius: 0 0 10px 10px;
            "></div>
        </div>
        <button id="debug-toggle" style="
            position: fixed;
            bottom: 20px;
            right: 20px;
            width: 60px;
            height: 60px;
            background: linear-gradient(135deg, #0052FF 0%, #0041CC
100%);
            border: none;
            border-radius: 50%;
            color: white;
            font-size: 24px;
            cursor: pointer;
            z-index: 1000000;
            box-shadow: 0 8px 20px rgba(0,82,255,0.4);
            transition: transform 0.2s ease;
        ">🐛</button>
    `;

    document.body.insertAdjacentHTML('beforeend', debugHTML);

    initDebugListeners();
    makeDraggable();
```

```
}

// ========================================
// INITIALIZE EVENT LISTENERS
// ========================================
function initDebugListeners() {
    const toggleBtn = document.getElementById('debug-toggle');
    const console = document.getElementById('debug-console');

    toggleBtn.onclick = () => {
        const isVisible = console.style.display === 'flex';
        console.style.display = isVisible ? 'none' : 'flex';
        toggleBtn.style.display = isVisible ? 'block' : 'none';
    };

    toggleBtn.onmouseenter = () => {
        toggleBtn.style.transform = 'scale(1.1)';
    };
    toggleBtn.onmouseleave = () => {
        toggleBtn.style.transform = 'scale(1)';
    };

    // Tab switching
    document.querySelectorAll('.debug-tab').forEach(tab => {
        tab.onclick = (e) => {
            document.querySelectorAll('.debug-tab').forEach(t => {
                t.classList.remove('active');
                t.style.color = '#94a3b8';
                t.style.borderBottom = 'none';
            });
            e.target.classList.add('active');
            e.target.style.color = 'white';
            e.target.style.borderBottom = '2px solid #0052FF';

            filterLogs(e.target.dataset.tab);
        };
    });
}

// ========================================
// MAKE CONSOLE DRAGGABLE
```

```
// ========================================
function makeDraggable() {
    const console = document.getElementById('debug-console');
    const header = document.getElementById('debug-header');
    let isDragging = false;
    let startX, startY, startRight, startBottom;

    header.onmousedown = (e) => {
        if (e.target.tagName === 'BUTTON') return;
        isDragging = true;
        startX = e.clientX;
        startY = e.clientY;

        const rect = console.getBoundingClientRect();
        startRight = window.innerWidth - rect.right;
        startBottom = window.innerHeight - rect.bottom;

        document.onmousemove = (e) => {
            if (!isDragging) return;
            const dx = startX - e.clientX;
            const dy = startY - e.clientY;
            console.style.right = (startRight + dx) + 'px';
            console.style.bottom = (startBottom + dy) + 'px';
        };

        document.onmouseup = () => {
            isDragging = false;
            document.onmousemove = null;
            document.onmouseup = null;
        };
    };
}

// ========================================
// DEBUG LOG FUNCTION
// ========================================
export function debugLog(message, type = 'info') {
    const timestamp = new Date().toLocaleTimeString('cs-CZ', {
        hour: '2-digit',
        minute: '2-digit',
        second: '2-digit',
```

```
            fractionalSecondDigits: 3
        });

        const colors = {
            info: '#3b82f6',
            success: '#10b981',
            warn: '#f59e0b',
            error: '#ef4444'
        };

        const icons = {
            info: 'ℹ️',
            success: '✅',
            warn: '⚠️',
            error: '❌'
        };

        const logEntry = {
            time: timestamp,
            message: message,
            type: type,
            page: window.location.pathname
        };

        debugLogs.push(logEntry);
        console.log(`[${timestamp}] [${type.toUpperCase()}]
${message}`);

        const logsContainer = document.getElementById('debug-logs');
        if (logsContainer) {
            const logEl = document.createElement('div');
            logEl.className = `debug-log-item debug-${type}`;
            logEl.dataset.type = type;
            logEl.style.cssText = `
                margin-bottom: 10px;
                padding: 10px;
                border-left: 4px solid ${colors[type]};
                background: rgba(255,255,255,0.05);
                border-radius: 6px;
                word-wrap: break-word;
                line-height: 1.5;
```

```
        `;
        logEl.innerHTML = `
            <div style="display: flex; justify-content:
space-between; margin-bottom: 4px;">
                <strong style="color: #94a3b8; font-size:
10px;">[${timestamp}]</strong>
                <span style="color: ${colors[type]}; font-weight:
bold; font-size: 10px;">${icons[type]} ${type.toUpperCase()}</span>
            </div>
            <div style="color: #e2e8f0; font-size:
12px;">${escapeHtml(message)}</div>
        `;
        logsContainer.insertBefore(logEl, logsContainer.firstChild);

        // Auto-scroll to top
        logsContainer.scrollTop = 0;
    }
}

// ========================================
// FILTER LOGS BY TYPE
// ========================================
function filterLogs(type) {
    const allLogs = document.querySelectorAll('.debug-log-item');

    allLogs.forEach(log => {
        if (type === 'all') {
            log.style.display = 'block';
        } else {
            log.style.display = log.dataset.type === type ? 'block'
: 'none';
        }
    });
}

// ========================================
// CLEAR CONSOLE
// ========================================
window.clearDebugConsole = function() {
    const logsContainer = document.getElementById('debug-logs');
    if (logsContainer) {
```

```
        logsContainer.innerHTML = '';
        debugLogs = [];
    }
    debugLog('Console cleared', 'info');
};

// ========================================
// EXPORT LOGS TO FILE
// ========================================
window.exportDebugLogs = function() {
    const logText = debugLogs.map(log =>
        `[${log.time}] [${log.type.toUpperCase()}] ${log.message}`
    ).join('\n');

    const blob = new Blob([logText], { type: 'text/plain' });
    const url = URL.createObjectURL(blob);
    const a = document.createElement('a');
    a.href = url;
    a.download = `basecamp-debug-${Date.now()}.txt`;
    a.click();
    URL.revokeObjectURL(url);

    debugLog('Logs exported successfully', 'success');
};

// ========================================
// ESCAPE HTML FOR SECURITY
// ========================================
function escapeHtml(text) {
    const div = document.createElement('div');
    div.textContent = text;
    return div.innerHTML;
}

// ========================================
// AUTO-INITIALIZE ON LOAD
// ========================================
if (typeof window !== 'undefined') {
    window.addEventListener('load', () => {
        createDebugConsole();
        debugLog('Debug console initialized', 'success');
```

```
        debugLog(`Page: ${window.location.pathname}`, 'info');
    });
}

// Export for use in other modules
export { debugLogs };
```

KROK 2: Použití v jakémkoli JS souboru

javascript

```
// theory.js, labMenu.js, faucet.js, atd.
import { debugLog } from '../utils/debugConsole.js';

async function initWallet() {
    try {
        debugLog('Starting wallet initialization...', 'info');

        await sdk.actions.ready();
        debugLog('SDK ready!', 'success');

        const wallet = accounts[0];
        debugLog(`Wallet connected: ${wallet}`, 'success');

    } catch (error) {
        debugLog(`Wallet init failed: ${error.message}`, 'error');
    }
}
```

looks like this: