```
(function() {

    /* ============================================================
     * HELIKON - MINDTOUCH SEARCH ADD-ON v1.7.2 — 2026-09-30
     * --- firnCore customization ---
     * Suche via /@api/deki/site/query?parser=bestguess
     * (gleiche Engine wie die MindTouch-Hauptsuche)
     * ============================================================ */

    /* --- firnCore customization --- v1.7.1: Guard gegen
Mehrfach-Rendering */
    if (window.helikonSearchAddonLoaded) return;
    window.helikonSearchAddonLoaded = true;

    var PF = '"Press Start 2P","Courier New",Courier,monospace';
    var SITE = 'https://docs.trialanderror.tech';
    var AT = String.fromCharCode(64);

    /* ============================================================
     * KONFIGURATION — hier anpassen (Doku siehe Header)
     * ============================================================ */
    var MAX_PAGE_RESULTS = 1;
    var MAX_FILE_RESULTS = 1;
    var MAX_TAGS = 5;
    var FETCH_LIMIT = 25;
    var INCLUDE_USER_PAGES = false;
    var PREFIX_FALLBACK = true;
    var TAG_MIN_WORD = 3;
    var SNIPPET_LEN = 200;
    var SNIPPET_LEAD = 60;
    var DEBUG = true;
    /* --- firnCore customization --- v1.7.2: _internal_-Filter */
    var HIDE_PREFIX = '_internal_';

    /* === CSS === */
    var css = ''
    + '.mtSearch { margin:0; border:2px solid #0077aa; border-radius:8px;
background:#f0f8ff; overflow:hidden; font-family:' + PF + '; }'
    + '.mtSearch .hdr {
background:linear-gradient(135deg,#005580,#0077aa); color:#fff;
padding:10px 14px; font-size:12px; line-height:1.6; display:flex;
justify-content:space-between; align-items:center; }'
    + '.mtSearch .hdr .badge { background:rgba(255,255,255,0.2);
padding:2px 8px; border-radius:10px; font-size:11px; }'
    + '.mtSearch .sec { padding:6px 14px; background:#e0ecf4;
font-size:11px; font-weight:bold; color:#005580; border-top:1px solid
#cce0ee; }'
    + '.mtSearch .item { padding:10px 14px; border-top:1px solid #e0ecf4;
font-size:11px; line-height:1.6; transition:background 0.1s; }'
```

```
    + '.mtSearch .item:hover { background:#e8f4fd; }'
    + '.mtSearch .item a { color:#0077aa; text-decoration:none;
font-size:13px; font-weight:bold; display:block; }'
    + '.mtSearch .item a:hover { text-decoration:underline;
color:#005580; }'
    + '.mtSearch .item .path { color:#999; font-size:10px; display:block;
margin-top:2px; word-break:break-all; }'
    + '.mtSearch .item .snippet { color:#444; font-size:10px;
display:block; margin-top:5px; line-height:1.6; border-left:2px solid
#b0d4e8; padding-left:8px; font-family:"Courier New",Courier,monospace; }'
    + '.mtSearch .item .snippet .hit { background:#fff3b0; color:#333;
border-radius:2px; padding:0 2px; }'
    + '.mtSearch .item .meta { color:#777; font-size:10px;
margin-top:4px; }'
    + '.mtSearch .item .via { color:#cc8800; font-size:10px;
font-style:italic; }'
    + '.mtSearch .file-item { padding:8px 14px; border-top:1px solid
#e0ecf4; font-size:11px; display:flex; align-items:center; gap:10px;
transition:background 0.1s; }'
    + '.mtSearch .file-item:hover { background:#e8f4fd; }'
    + '.mtSearch .file-item a { color:#0077aa; text-decoration:none;
font-weight:bold; }'
    + '.mtSearch .file-item a:hover { text-decoration:underline; }'
    + '.mtSearch .file-thumb { width:60px; height:60px; object-fit:cover;
border:1px solid #cce0ee; border-radius:4px; cursor:pointer;
background:#fff; }'
    + '.mtSearch .file-info { flex:1; }'
    + '.mtSearch .file-info .fname { font-size:12px; font-weight:bold;
display:block; }'
    + '.mtSearch .file-info .fmeta { font-size:10px; color:#999; }'
    + '.mtSearch .tags { padding:8px 14px 12px; border-top:1px solid
#cce0ee; }'
    + '.mtSearch .tags .lbl { font-size:10px; color:#999;
margin-bottom:4px; }'
    + '.mtSearch .tag { display:inline-block; background:#0077aa;
color:#fff; padding:3px 10px; border-radius:12px; font-size:10px;
margin:3px 4px 3px 0; text-decoration:none; transition:background 0.15s;
}'
    + '.mtSearch .tag:hover { background:#005580; }'
    + '.mtSearch .empty { padding:14px; color:#999; font-size:12px;
text-align:center; line-height:1.6; }'
    + '.mtSearch .ld { padding:14px; color:#0077aa; font-size:11px;
text-align:center; }'
    + AT + 'keyframes mtP { 0%,100% { opacity:1; } 50% { opacity:0.4; } }'
    + '.mtSearch .ld { animation:mtP 1.2s ease-in-out infinite; }'
    + AT + 'keyframes mtFadeOut { from { opacity:1; max-height:60px; } to
{ opacity:0; max-height:0; padding:0; margin:0; border-width:0; } }'
    + '.mtSearch.fadeout { animation:mtFadeOut 0.6s ease-out forwards;
overflow:hidden; }'
    + '.mt-autolink { color:#0077aa; text-decoration:none;
```

```
border-bottom:1px dotted #0077aa; font-family:' + PF + ';
font-size:inherit; }'
    + '.mt-autolink:hover { color:#005580; border-bottom-style:solid; }';

    var sEl = document.createElement('style');
    sEl.type = 'text/css';
    sEl.appendChild(document.createTextNode(css));
    document.getElementsByTagName('head')[0].appendChild(sEl);

    /* === HTML TAG HELPERS (wie Buddy Chat) === */
    var T = String.fromCharCode(60);
    var TC = String.fromCharCode(60) + '/';
    var G = String.fromCharCode(62);

    /* === DEBUG LOG === */
    function dbg(msg) {
        if (DEBUG && window.console) console.log('[MT Search v1.7.2] ' +
msg);
    }

    /* === HTML ESCAPE === */
    function esc(s) {
        if (!s && s !== 0) return '';
        var out = '' + s;
        out = out.replace(/&/g, '&amp;');
        out = out.replace(new RegExp(T, 'g'), '&lt;');
        out = out.replace(new RegExp(G, 'g'), '&gt;');
        return out;
    }
    function escA(s) {
        if (!s && s !== 0) return '';
        return esc('' + s).replace(/"/g, '&quot;');
    }

    /* === DATE FORMAT === */
    function fmtDate(d) {
        if (!d) return '';
        try {
            var dt = new Date(d);
            if (isNaN(dt.getTime())) return '';
            var dd = dt.getDate(); var mm = dt.getMonth() + 1; var yy =
dt.getFullYear();
            return (dd < 10 ? '0' : '') + dd + '.' + (mm < 10 ? '0' : '')
+ mm + '.' + yy;
        } catch(e) { return ''; }
    }

    /* === GREETING CHECK === */
    var GREETINGS = new
Array('hi','hallo','hey','moin','servus','grüezi','grüzi','salü','yo','guten
```

```
tag','guten morgen','guten abend','hoi');
    function isGreeting(msg) {
        var m = msg.toLowerCase().replace(/[?.,:;!]+/g,
'').replace(/^\s+|\s+$/g, '');
        var i;
        for (i = 0; i < GREETINGS.length; i++) { if (m === GREETINGS[i])
return true; }
        return false;
    }

    /* === STOPWORDS === */
    var STOP_WORDS = new Array(
        /* DE: Fragen / Aufforderungen */

'kennst','kennt','weisst','weißt','was','ist','sind','war','waren','wer','wie','

'kannst','koennte','könnte','kann','soll','sollte','will','willst','moechte','mö

'sagen','sag','zeig','zeige','zeigen','finde','finden','suche','suchen','such','
        /* DE: Artikel / Pronomen / Praepositionen / Fuellwoerter */

'ein','eine','einen','einem','einer','eines','der','die','das','den','dem','des'

'und','oder','aber','ich','mich','mir','du','dir','es','er','sie','wir','uns','i

'von','vom','zu','zum','zur','im','in','am','an','auf','aus','bei','mit','nach',

'bitte','mal','denn','auch','noch','schon','ja','nein','alle','allen','alles','e

'info','infos','information','informationen','irgendwas','irgendwelche',
        'danke','merci','ok','okay','thx','super','prima','genau',
        /* EN */

'do','does','you','know','what','is','are','was','were','the','a','an','can','co

'find','search','for','tell','about','where','when','who','how','which','of','on

'give','any','some','all','please','have','has','there','anything','thanks'
    );
    function isStopWord(w) {
        var s;
        for (s = 0; s < STOP_WORDS.length; s++) { if (w ===
STOP_WORDS[s]) return true; }
        return false;
    }

    /* === CLEAN QUERY (Lucene-Sonderzeichen neutralisiert) === */
    function cleanQuery(msg) {
        var q = ('' + msg)
            .replace(/[?.,:;!'"()\[\]\u007B\u007D+\-&|^~*\\\/]+/g, ' ')
```

```
            .replace(/\s+/g, ' ')
            .replace(/^\s+|\s+$/g, '');
        if (!q) return '';
        var words = q.split(' ');
        var kept = new Array();
        var i;
        for (i = 0; i < words.length; i++) {
            var w = words[i].toLowerCase();
            if (w.length > 1 && !isStopWord(w)) kept.push(words[i]);
        }
        return kept.join(' ');
    }

    /* === QUERY WORDS (Highlight + Tag-Match) === */
    function queryWords(q) {
        var parts = q.toLowerCase().split(/\s+/);
        var seen = new Object();
        var out = new Array();
        var i;
        for (i = 0; i < parts.length; i++) {
            var w = parts[i];
            if (w.length >= TAG_MIN_WORD && !seen[w]) { seen[w] = true;
out.push(w); }
        }
        return out;
    }

    /* === FILE SIZE FORMAT === */
    function fmtSize(bytes) {
        if (!bytes) return '';
        var n = parseInt(bytes, 10);
        if (isNaN(n)) return '';
        if (n < 1024) return n + ' B';
        if (n < 1048576) return Math.round(n / 1024) + ' KB';
        return (n / 1048576).toFixed(1) + ' MB';
    }

    function isImage(type) {
        if (!type) return false;
        return type.indexOf('image/') === 0;
    }

    /* === JUNK FILTERS (Namespace-Filter wie Hauptsuche) === */
    /* --- firnCore customization --- v1.7.2: _internal_-Check
(case-insensitive) */
    function isHidden(s) {
        if (!HIDE_PREFIX) return false;
        return ('' + (s ||
'')).toLowerCase().indexOf(HIDE_PREFIX.toLowerCase()) !== -1;
    }
```

```
    function isJunkPath(path) {
        var p = '' + (path || '');
        if (p.indexOf('Template:') === 0) return true;
        if (p.indexOf('Special:') === 0) return true;
        if (!INCLUDE_USER_PAGES && p.indexOf('User:') === 0) return true;
        if (p.indexOf('[object') !== -1) return true;
        if (isHidden(p)) return true;
        return false;
    }
    function isJunkFile(fn) {
        fn = ('' + (fn || '')).toLowerCase();
        if (isHidden(fn)) return true;
        if (fn.indexOf('mindtouch.import') !== -1) return true;
        if (fn.indexOf('mindtouch#') !== -1) return true;
        if (fn === 'element.js') return true;
        if (fn.length < 20 && fn.indexOf('.js') === fn.length - 3) return
true;
        if (fn.length < 20 && fn.indexOf('.css') === fn.length - 4)
return true;
        return false;
    }

    /* === SNIPPET: Ausschnitt rund um ersten Treffer === */
    function makeSnippet(content, words) {
        if (!content) return '';
        var plain = ('' + content).replace(/[\s ]+/g, '
').replace(/^\s+|\s+$/g, '');
        if (!plain) return '';
        var lower = plain.toLowerCase();
        var idx = -1;
        var i;
        for (i = 0; i < words.length; i++) {
            var p = lower.indexOf(words[i]);
            if (p !== -1 && (idx === -1 || p < idx)) idx = p;
        }
        var start = 0;
        if (idx > SNIPPET_LEAD) {
            start = idx - SNIPPET_LEAD;
            var sp = plain.indexOf(' ', start);
            if (sp !== -1 && sp < idx) start = sp + 1;
        }
        var end = start + SNIPPET_LEN;
        if (end >= plain.length) {
            end = plain.length;
        } else {
            var sp2 = plain.lastIndexOf(' ', end);
            if (sp2 > start + Math.floor(SNIPPET_LEN / 2)) end = sp2;
        }
        return (start > 0 ? '… ' : '') + plain.substring(start, end) +
(end < plain.length ? ' …' : '');
```

```
    }

    /* === SNIPPET HIGHLIGHT (alle Vorkommen, sicher escaped) === */
    var HIT_OPEN = T + 'span class="hit"' + G;
    var HIT_CLOSE = TC + 'span' + G;
    function highlightSnippet(text, words) {
        if (!text) return '';
        if (!words || words.length === 0) return esc(text);
        var lower = text.toLowerCase();
        var mark = new Array();
        var i, w, k;
        for (i = 0; i < text.length; i++) mark.push(false);
        for (w = 0; w < words.length; w++) {
            var word = words[w];
            var pos = lower.indexOf(word);
            while (pos !== -1) {
                for (k = pos; k < pos + word.length; k++) mark[k] = true;
                pos = lower.indexOf(word, pos + word.length);
            }
        }
        var out = '';
        var run = '';
        var runHit = false;
        for (i = 0; i < text.length; i++) {
            if (i > 0 && mark[i] !== runHit) {
                out += runHit ? HIT_OPEN + esc(run) + HIT_CLOSE :
esc(run);
                run = '';
            }
            runHit = mark[i];
            run += text.charAt(i);
        }
        if (run.length > 0) out += runHit ? HIT_OPEN + esc(run) +
HIT_CLOSE : esc(run);
        return out;
    }

    /* === API HELPER === */
    function mtJson(path, cb) {
        var url = '/' + AT + 'api/deki' + path;
        url += (url.indexOf('?') === -1 ? '?' : '&') +
'dream.out.format=json';
        dbg('GET ' + url);
        var x = new XMLHttpRequest();
        x.open('GET', url, true);
        x.onload = function() {
            if (x.status >= 200 && x.status < 300) {
                try { cb(null, JSON.parse(x.responseText)); }
                catch(e) { cb('parse'); }
            } else { cb('http ' + x.status); }
```

```
        };
        x.onerror = function() { cb('net'); };
        x.send();
    }

    function fix(u) {
        if (!u) return '';
        return ('' + u).replace('http://docs.trialanderror.tech', SITE)
                .replace(/^http:\/\/172\.\d+\.\d+\.\d+/, SITE);
    }

    function asArray(v) {
        var out = new Array();
        if (!v) return out;
        if (typeof v.length === 'number' && typeof v !== 'string') return
v;
        out.push(v);
        return out;
    }

    /* === SEARCH: /site/query (bestguess) === */
    function runQuery(q, words, cb) {
        var path = '/site/query?q=' + encodeURIComponent(q)
                 + '&parser=bestguess&limit=' + FETCH_LIMIT;
        mtJson(path, function(err, data) {
            var res = new Object();
            res.pages = new Array();
            res.files = new Array();
            res.tags = new Array();
            if (err || !data) { dbg('Fehler: ' + err); cb(res); return; }

            dbg('count=' + (data[AT + 'count'] || 0) + ' parsedQuery=' +
(data.parsedQuery || ''));

            var arr = asArray(data.result);
            var seenPage = new Object();
            var seenFile = new Object();
            var tagSeen = new Object();
            var i, t, w;

            for (i = 0; i < arr.length; i++) {
                var r = arr[i] || new Object();
                var type = r.type || '';
                var pi = r.page || new Object();
                var ppath = '' + (pi.path || '');

                if (isJunkPath(ppath)) continue;

                if (type === 'file') {
                    var fname = r.title || '';
```

```
                    if (!fname || isJunkFile(fname)) { if
(isHidden(fname)) dbg('versteckt: ' + fname); continue; }
                    var fid = r.id || fname;
                    if (seenFile[fid]) continue;
                    seenFile[fid] = true;
                    var fe = new Object();
                    fe.id = fid;
                    fe.filename = fname;
                    fe.type = r.mime || '';
                    fe.size = r.size || '';
                    fe.href = fix(r.uri || '');
                    fe.parentTitle = pi.title || '';
                    fe.parentUri = ppath ? SITE + '/' + ppath : '';
                    res.files.push(fe);
                } else if (type === 'page' || type === 'comment') {
                    if (seenPage[ppath]) continue;
                    seenPage[ppath] = true;
                    var pe = new Object();
                    pe.id = r.id || '';
                    pe.title = (type === 'page' ? r.title : pi.title) ||
ppath;
                    pe.path = ppath;
                    pe.uri = type === 'page' ? fix(r.uri || '') : SITE +
'/' + ppath;
                    pe.date = r['date.modified'] || '';
                    pe.source = type;
                    pe.snippet = makeSnippet(r.content || '', words);
                    res.pages.push(pe);

                    /* Passende Tags aus dem tag-Feld ableiten */
                    var tl = ('' + (r.tag || '')).split('\n');
                    for (t = 0; t < tl.length; t++) {
                        var tg = tl[t].replace(/^\s+|\s+$/g, '');
                        var tgl = tg.toLowerCase();
                        if (!tg || tagSeen[tgl]) continue;
                        for (w = 0; w < words.length; w++) {
                            if (tgl.indexOf(words[w]) !== -1) {
                                tagSeen[tgl] = true;
                                res.tags.push(tg);
                                break;
                            }
                        }
                    }
                }
            }
            dbg('nach Filter: pages=' + res.pages.length + ' files=' +
res.files.length + ' tags=' + res.tags.length);
            cb(res);
        });
    }
```

```
    /* === RENDER === */
    function openTag(el, cls) { return T + el + ' class="' + cls + '"' +
G; }
    function closeTag(el) { return TC + el + G; }
    function link(cls, href, inner) {
        return T + 'a' + (cls ? ' class="' + cls + '"' : '') + ' href="'
+ escA(href) + '" target="_blank"' + G + inner + TC + 'a' + G;
    }

    function render(query, words, pages, files, tags) {
        var total = pages.length + files.length;
        var h = '';
        var i, fi, t;
        h += openTag('div', 'mtSearch');
        h += openTag('div', 'hdr');
        h += T + 'span' + G + '\uD83D\uDD0D "' + esc(query) + '"' +
closeTag('span');
        h += openTag('span', 'badge') + total + ' Treffer' +
closeTag('span');
        h += closeTag('div');

        if (pages.length > 0) {
            h += openTag('div', 'sec') + '\uD83D\uDCC4 Seiten (' +
pages.length + ')' + closeTag('div');
            for (i = 0; i < pages.length; i++) {
                var pg = pages[i];
                var url = pg.uri || (SITE + '/' + pg.path);
                h += openTag('div', 'item');
                h += link('', url, esc(pg.title || pg.path));
                if (pg.path) h += openTag('span', 'path') + '/' +
esc(pg.path) + closeTag('span');
                if (pg.snippet) h += openTag('span', 'snippet') +
highlightSnippet(pg.snippet, words) + closeTag('span');
                var meta = new Array();
                var dt = fmtDate(pg.date);
                if (dt) meta.push(dt);
                if (pg.source === 'comment') meta.push(openTag('span',
'via') + 'via Kommentar' + closeTag('span'));
                if (meta.length > 0) h += openTag('span', 'meta') +
meta.join(' • ') + closeTag('span');
                h += closeTag('div');
            }
        }

        if (files.length > 0) {
            h += openTag('div', 'sec') + '\uD83D\uDCCE Dateien (' +
files.length + ')' + closeTag('div');
            for (fi = 0; fi < files.length; fi++) {
                var f = files[fi];
```

```
                h += openTag('div', 'file-item');
                if (isImage(f.type) && f.href) {
                    var img = T + 'img class="file-thumb" src="' +
escA(f.href + '?size=thumb') + '" alt="' + escA(f.filename) + '" /' + G;
                    h += link('', f.parentUri || f.href, img);
                }
                h += openTag('div', 'file-info');
                h += link('fname', f.href || '#', esc(f.filename));
                var fmeta = new Array();
                if (f.parentTitle) fmeta.push('auf: ' +
esc(f.parentTitle));
                var sz = fmtSize(f.size);
                if (sz) fmeta.push(sz);
                if (fmeta.length > 0) h += openTag('span', 'fmeta') +
fmeta.join(' • ') + closeTag('span');
                h += closeTag('div');
                h += closeTag('div');
            }
        }

        if (tags.length > 0) {
            h += openTag('div', 'tags');
            h += openTag('div', 'lbl') + 'Passende Tags:' +
closeTag('div');
            for (t = 0; t < tags.length; t++) {
                h += link('tag', SITE + '/Special:Tags?tag=' +
encodeURIComponent(tags[t]), esc(tags[t]));
            }
            h += closeTag('div');
        }

        h += closeTag('div');
        return h;
    }

    /* === AUTOLINKER (v1.6.2 / v1.6.6, Logik unveraendert) === */
    var WIKI_SECTIONS = new
Array('Bereiche','Special','FILM','GAMES','INTERN','LABOR','MINAKTIV','MUSEUM','
    function looksLikeWikiPath(raw) {
        var i;
        for (i = 0; i < WIKI_SECTIONS.length; i++) { if
(raw.indexOf(WIKI_SECTIONS[i]) === 0) return true; }
        return false;
    }
    function extractWikiPath(s) {
        if (!s || s.length === 0) return null;
        var first = s.charAt(0);
        if (!((first >= 'A' && first <= 'Z') || (first >= 'a' && first <=
'z') || first === '_')) return null;
        var i = 0;
```

```
        while (i < s.length) {
            var c = s.charAt(i);
            if ((c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z') || (c >=
'0' && c <= '9') ||
                c === '_' || c === ':' || c === '?' || c === '=' || c ===
'&' || c === '%' ||
                c === '-' || c === '/' || c === String.fromCharCode(35)
|| c === '.') { i++; }
            else { break; }
        }
        if (i < 2) return null;
        var path = s.substring(0, i);
        while (path.length > 0 && path.charAt(path.length - 1) === '.') {
            path = path.substring(0, path.length - 1);
        }
        if (path.length < 2) return null;
        return path;
    }
    function makePart(type, val) { var p = new Object(); p.type = type;
p.val = val; return p; }
    function autolinkNode(el) {
        var walker = document.createTreeWalker(el, 4, null, false);
        var textNodes = new Array();
        var node = walker.nextNode();
        var ti, pi;
        while (node) {
            var parent = node.parentNode; var skip = false;
            while (parent && parent !== el) {
                if (parent.tagName && parent.tagName.toLowerCase() ===
'a') { skip = true; break; }
                parent = parent.parentNode;
            }
            if (!skip) textNodes.push(node);
            node = walker.nextNode();
        }
        for (ti = 0; ti < textNodes.length; ti++) {
            var tn = textNodes[ti];
            var text = tn.nodeValue;
            if (text.indexOf('/') === -1) continue;
            var parts = new Array();
            var pos = 0; var fragStart = 0; var found = false;
            while (pos < text.length) {
                if (text.charAt(pos) === '/') {
                    var pathMatch = extractWikiPath(text.substring(pos +
1));
                    if (pathMatch !== null &&
looksLikeWikiPath(pathMatch)) {
                        parts.push(makePart('text',
text.substring(fragStart, pos)));
                        parts.push(makePart('link', '/' + pathMatch));
```

```
                        pos = pos + 1 + pathMatch.length;
                        fragStart = pos; found = true; continue;
                    }
                }
                pos++;
            }
            if (!found) continue;
            parts.push(makePart('text', text.substring(fragStart)));
            var frag = document.createDocumentFragment();
            for (pi = 0; pi < parts.length; pi++) {
                var part = parts[pi];
                if (part.type === 'text') { if (part.val.length > 0)
frag.appendChild(document.createTextNode(part.val)); }
                else {
                    var a = document.createElement('a');
                    a.href = SITE + part.val; a.target = '_blank';
a.className = 'mt-autolink';
                    a.appendChild(document.createTextNode(part.val));
frag.appendChild(a);
                }
            }
            tn.parentNode.replaceChild(frag, tn);
        }
    }

    /* === OBSERVER === */
    function init() {
        var chatLog = document.getElementById('chatLog');
        if (!chatLog) { setTimeout(init, 300); return; }
        var lastMsg = '';
        var obs = new MutationObserver(function(muts) {
            var mi, ni, li;
            for (mi = 0; mi < muts.length; mi++) {
                var added = muts[mi].addedNodes;
                for (ni = 0; ni < added.length; ni++) {
                    var n = added[ni];
                    if (n.nodeType !== 1) continue;
                    var links = n.getElementsByTagName('a');
                    for (li = 0; li < links.length; li++) {
links[li].setAttribute('target', '_blank'); }
                    var cn = n.className || '';
                    if (cn.indexOf('chatMsg') !== -1 &&
cn.indexOf('user') === -1) {
                        (function(nd) { setTimeout(function() {
autolinkNode(nd); }, 80); })(n);
                    }
                    if (cn.indexOf('chatMsg') !== -1 &&
cn.indexOf('user') !== -1) {
                        var msg = (n.textContent ||
'').replace(/^\s+|\s+$/g, '');
```

```
                        if (msg && msg !== lastMsg && !isGreeting(msg)) {
                            lastMsg = msg;
                            if (typeof window.helikonSearchEnabled !==
'undefined' && !window.helikonSearchEnabled) { return; }
                            doSearch(msg, chatLog);
                        }
                    }
                }
            }
        });
        var obsCfg = new Object();
        obsCfg.childList = true;
        obs.observe(chatLog, obsCfg);
        if (window.console) console.log('[MT Search Add-on v1.7.2] Aktiv
— /site/query bestguess, MAX_PAGE_RESULTS=' + MAX_PAGE_RESULTS + '
MAX_FILE_RESULTS=' + MAX_FILE_RESULTS + ' DEBUG=' + DEBUG);
    }

    /* === SEARCH FLOW === */
    function doSearch(rawQuery, chatLog) {
        var query = cleanQuery(rawQuery);
        if (!query || query.length < 2) { dbg('keine Suche (nur
Fuellwoerter): "' + rawQuery + '"'); return; }
        var words = queryWords(query);
        dbg('Prompt: "' + rawQuery + '" -> Query: "' + query + '"');

        var ph = document.createElement('div');
        ph.className = 'mtSearch';
        ph.innerHTML = openTag('div', 'ld') + '\uD83D\uDD0D Durchsuche
MindTouch...' + closeTag('div');
        var typing = null;
        var ch = chatLog.children;
        var i;
        for (i = ch.length - 1; i >= 0; i--) {
            if (ch[i].className && ch[i].className.indexOf('typing') !==
-1) { typing = ch[i]; break; }
        }
        if (typing) chatLog.insertBefore(ph, typing);
        else chatLog.appendChild(ph);
        chatLog.scrollTop = chatLog.scrollHeight;

        function finish(res) {
            var pages = res.pages;
            var files = res.files;
            var tags = res.tags;
            if (pages.length > MAX_PAGE_RESULTS) pages = pages.slice(0,
MAX_PAGE_RESULTS);
            if (files.length > MAX_FILE_RESULTS) files = files.slice(0,
MAX_FILE_RESULTS);
            if (tags.length > MAX_TAGS) tags = tags.slice(0, MAX_TAGS);
```

```
            if (pages.length === 0 && files.length === 0 && tags.length
=== 0) {
                ph.className = 'mtSearch fadeout';
                setTimeout(function() { if (ph.parentNode)
ph.parentNode.removeChild(ph); }, 650);
                return;
            }
            ph.outerHTML = render(query, words, pages, files, tags);
            chatLog.scrollTop = chatLog.scrollHeight;
        }

        runQuery(query, words, function(res) {
            var empty = res.pages.length === 0 && res.files.length === 0;
            if (empty && PREFIX_FALLBACK && words.length > 0) {
                var pq = words.join('* ') + '*';
                dbg('0 Treffer -> Prefix-Fallback: "' + pq + '"');
                runQuery(pq, words, finish);
            } else {
                finish(res);
            }
        });
    }

    /* === START === */
    if (document.readyState === 'complete' || document.readyState ===
'interactive') {
        setTimeout(init, 500);
    } else {
        document.addEventListener('DOMContentLoaded', function() {
setTimeout(init, 500); });
    }

})();
```