```
#serverBuddy {
    position: fixed;
    display: none;
    right: 16px;
    top: 50%;
    transform: translateY(-50%);
    z-index: 2147483000;
    cursor: grab;
    -webkit-user-select: none;
    -moz-user-select: none;
    user-select: none;
    opacity: 0.3;
    transition: opacity 0.5s ease;
    touch-action: none;
}
#serverBuddy.active {
    opacity: 1;
}
#serverBuddy:hover {
    opacity: 1;
}
#serverBuddy.dragging {
    opacity: 1;
    cursor: grabbing;
    transition: none;
}
#serverBuddy img {
    width: 96px;
    height: 96px;
    image-rendering: pixelated;
    image-rendering: crisp-edges;
    display: block;
    border: none;
    outline: none;
    pointer-events: none;
    -webkit-user-drag: none;
    user-drag: none;
}
#serverBuddy.dizzy img {
    animation: buddyShake 0.15s ease-in-out 6;
}
@keyframes buddyShake {
    0%   { transform: translate(0, 0) rotate(0deg); }
    25%  { transform: translate(-4px, 2px) rotate(-8deg); }
    50%  { transform: translate(4px, -2px) rotate(8deg); }
    75%  { transform: translate(-2px, 1px) rotate(-4deg); }
    100% { transform: translate(0, 0) rotate(0deg); }
}
```

```
/* --- firnCore customization: Idle-Pers\u00F6nlichkeit v2.2 --- */
#serverBuddy.breathing #buddyImg {
    animation: buddyBreathe 4.5s ease-in-out infinite;
}
@keyframes buddyBreathe {
    0%, 100% { transform: scale(1); }
    50%      { transform: scale(1.018); }
}
#serverBuddy.yawning #buddyImg {
    animation: buddyYawn 0.9s ease-in-out;
}
@keyframes buddyYawn {
    0%   { transform: scale(1) rotate(0deg); }
    30%  { transform: scale(1.04) rotate(-1deg); }
    60%  { transform: scale(1.05) rotate(1deg); }
    100% { transform: scale(1) rotate(0deg); }
}
.buddyEye {
    position: absolute;
    width: 10px;
    height: 8px;
    pointer-events: none;
    opacity: 0;
    transition: opacity 0.4s ease;
}
#serverBuddy.eyesActive .buddyEye {
    opacity: 1;
}
.buddyEyeLeft {
    left: 35.2%;
    top: 36.7%;
    transform: translate(-50%, -50%);
}
.buddyEyeRight {
    left: 60%;
    top: 36.7%;
    transform: translate(-50%, -50%);
}
.buddyPupil {
    position: absolute;
    left: 50%;
    top: 50%;
    width: 3px;
    height: 3px;
    border-radius: 50%;
    background: #0a0c14;
    transform: translate(-50%, -50%);
    transition: margin 0.15s ease-out;
}
.buddyLid {
```

```
    position: absolute;
    left: 0;
    top: 0;
    width: 100%;
    height: 100%;
    background: #141623;
    border-radius: 2px;
    transform: scaleY(0);
    transform-origin: center;
}
.buddyEye.blinking .buddyLid {
    animation: buddyBlink 0.18s ease-in-out;
}
@keyframes buddyBlink {
    0%   { transform: scaleY(0); }
    50%  { transform: scaleY(1); }
    100% { transform: scaleY(0); }
}
/* --- end firnCore customization --- */

#buddySpeech {
    position: fixed;
    right: 120px;
    top: 50%;
    transform: translateY(-50%) scale(0);
    background: #fff;
    color: #333;
    padding: 12px 18px;
    border-radius: 12px;
    border: 2px solid #ccc;
    font-family: 'Press Start 2P', 'Courier New', Courier, monospace;
    font-size: 16px;
    line-height: 1.8;
    -webkit-font-smoothing: none;
    -moz-osx-font-smoothing: unset;
    font-smooth: never;
    text-rendering: optimizeSpeed;
    -webkit-backface-visibility: hidden;
    backface-visibility: hidden;
    white-space: normal;
    max-width: 320px;
    word-wrap: break-word;
    z-index: 2147483001;
    opacity: 0;
    transition: opacity 0.3s ease, transform 0.3s ease;
    pointer-events: none;
    box-shadow: 0 2px 8px rgba(0,0,0,0.1);
}
#buddySpeech.visible {
    opacity: 1;
```

```
    transform: translateY(-50%) scale(1);
}
#buddySpeech::after {
    content: '';
    position: absolute;
    right: -8px;
    top: 50%;
    transform: translateY(-50%);
    width: 0;
    height: 0;
    border-top: 8px solid transparent;
    border-bottom: 8px solid transparent;
    border-left: 8px solid #ccc;
}
#buddySpeech.bubble-right::after {
    right: auto;
    left: -8px;
    border-left: none;
    border-right: 8px solid #ccc;
}
```

```
(function() {
    /* ============================================================
     * Server Buddy v2.3 — 2026-10-06
     * FIX: Page Topic via helikon-proxy.php?topic= (Properties-API
     *      ist per NGINX gesperrt), console.warn bei Abbruch.
     * v2.2: Idle-Pers\u00F6nlichkeit \u2014 Augen folgen der Maus
(Overlay,
     *      kein neues GIF n\u00F6tig), Blinzeln, Atmen, variable
     *      Ambient-Idle-Spr\u00FCche, G\u00E4hn-\u00DCbergang vor Sleep.
     * ============================================================ */

    var fontLink = document.createElement('link');
    fontLink.rel = 'stylesheet';
    fontLink.href = 'https://fonts.googleapis.com/
css2?family=Press+Start+2P';
    document.head.appendChild(fontLink);

    /* === CONFIG === */
    var SLEEP_SEC   = 30;
    var HAPPY_SEC   = 6;
    var ERROR_SEC   = 8;
    var LOAD_SEC    = 5;
```

```
    var FADE_SEC    = 8;
    var DIZZY_SEC   = 5;
    var CHAT_URL    = '/Willkommen/Helikon';
    var TOPIC_DELAY = 6;
    var DRAG_THRESHOLD = 6;
    var SHAKE_THRESHOLD = 5;
    var SHAKE_WINDOW    = 800;
    var SHAKE_MIN_DIST  = 15;
    var STORAGE_KEY = 'serverBuddyPos';
    var MSG_DEFAULT = 6000;
    var MSG_LONG    = 10000;
    var HOVER_COOLDOWN = 6000;

    /* --- firnCore customization: Idle-Pers\u00F6nlichkeit config v2.2
--- */
    var BLINK_MIN_MS      = 3000;
    var BLINK_MAX_MS      = 6500;
    var EYE_MOVE_RADIUS   = 3;
    var EYE_TRACK_DIST    = 220;
    var IDLE_EVENT_MIN_START = 45000;
    var IDLE_EVENT_MAX_START = 90000;
    var IDLE_EVENT_MIN_LATE  = 15000;
    var IDLE_EVENT_MAX_LATE  = 30000;
    var IDLE_RAMP_MS      = 5 * 60 * 1000;
    var YAWN_LEAD_SEC     = 3;
    /* --- end config --- */

    /* === GIFS === */
    var GIFS = new Array(5);
    GIFS[0] = '/@api/deki/files/410/=buddy_idle.gif';
    GIFS[1] = '/@api/deki/files/412/=buddy_sleeping.gif';
    GIFS[2] = '/@api/deki/files/409/=buddy_happy.gif';
    GIFS[3] = '/@api/deki/files/408/=buddy_error.gif';
    GIFS[4] = '/@api/deki/files/411/=buddy_loading.gif';

    var S_IDLE = 0, S_SLEEP = 1, S_HAPPY = 2, S_ERROR = 3, S_LOAD = 4,
S_DIZZY = 5;

    /* === SPEECH === */
    var SPEECH = new Array(6);
    SPEECH[0] = new Array('Bin ready!', 'Schieb mich rum!', 'Chatte mit
mir!', 'Doppelklick = Chat!');
    SPEECH[1] = new Array('zzz...', 'ZZzz...', '*schnarch*',
'oiiioiii...');
    SPEECH[2] = new Array('So so!', 'Super!', 'Nice!', 'Toll!', 'Weiter
so!');
    SPEECH[3] = new Array('ALARM!', 'Oh nein!', 'Fehler!', 'Hilfe!');
    SPEECH[4] = new Array('Moment...', 'Bin dran...', 'Gleich...', 'Bitte
warten...');
    SPEECH[5] = new Array('Aua mein Kopf!', 'Mir ist schwindlig!', 'H\
```

```
u00F6r auf zu sch\u00FCtteln!', 'Alles dreht sich!', 'Ich seh Sterne...',
'Weniger sch\u00FCtteln bitte!');

    var HELLOS = new Array('Salu\u00E9 du!', '*kitzel*', '01001000
01101001', '*blinzel*', 'Lass uns reden, Doppelklick!', 'Execute Order
66', 'Moin!');

    /* --- firnCore customization: Ambient-Idle-Spr\u00FCche v2.2 --- */
    var IDLE_AMBIENT = new Array('*summt leise*', 'Hmm...', '...', 'Alles
ruhig hier.', '*wippt leicht*', '...Server-Gedanken...', 'Der Takt l\
u00E4uft.', 'Alles im gr\u00FCnen Bereich.');
    var IDLE_NIGHT   = new Array('Ruhig heute Nacht...', 'zzz bald...',
'Nachtschicht l\u00E4uft.', 'Die Sterne stehen gut f\u00FCrs Debugging.',
'Wer ist um die Zeit noch wach...');
    var IDLE_MORNING = new Array('Guten Morgen!', 'Schon wach?', 'Ich f\
u00FChle mich frisch gebootet.');
    /* --- end ambient speech --- */

    /* === STATE === */
    var curState   = -1;
    var lastAct    = Date.now();
    var sTimer     = null;
    var fadeTimer   = null;
    var leaving    = false;
    var pageError  = false;
    var preloading = false;
    var msgTimer   = null;
    var yawnTriggered = false;

    var clickCount  = 0;
    var clickTimer  = null;
    var pressTimer  = null;

    /* === DRAG STATE === */
    var isDragging  = false;
    var wasDragged  = false;
    var dragStartX  = 0;
    var dragStartY  = 0;
    var dragOffsetX = 0;
    var dragOffsetY = 0;
    var positioned  = false;

    /* === SHAKE DETECTION === */
    var shakeHistory = new Array();
    var lastDragY    = 0;
    var lastDragDir  = 0;

    /* === DOM === */
    var bWrap   = document.getElementById('serverBuddy');
    var bImg    = document.getElementById('buddyImg');
```

```
    var bSpeech = document.getElementById('buddySpeech');
    if (!bWrap || !bImg || !bSpeech) return;

    /* firnCore customization: Idle-Eye-Overlay refs (optional, degrade
gracefully) */
    var eyeLeft     = document.getElementById('buddyEyeLeft');
    var eyeRight    = document.getElementById('buddyEyeRight');
    var pupilLeft   = document.getElementById('buddyPupilLeft');
    var pupilRight  = document.getElementById('buddyPupilRight');

    bWrap.style.display = 'block';

    bImg.setAttribute('alt', '');
    bImg.setAttribute('title', '');
    bImg.setAttribute('draggable', 'false');
    bImg.addEventListener('dragstart', function(e) { e.preventDefault();
});
    bImg.addEventListener('contextmenu', function(e) {
e.preventDefault(); });

    /* FIX: Fallback bei Ladefehler - Error-GIF zeigen */
    bImg.addEventListener('error', function() {
        if (bImg.src.indexOf('buddy_error') === -1) {
            bImg.src = GIFS[S_ERROR];
        }
    });

    /* === HELPERS === */
    function pick(arr) {
        return arr[Math.floor(Math.random() * arr.length)];
    }

    function isBubbleVisible() {
        return bSpeech.className.indexOf('visible') !== -1;
    }

    function hideBubble() {
        if (msgTimer) clearTimeout(msgTimer);
        bSpeech.className = '';
    }

    function clamp(val, min, max) {
        if (val < min) return min;
        if (val > max) return max;
        return val;
    }

    /* === POSITION MANAGEMENT === */
    function applyPosition(x, y) {
        bWrap.style.right = 'auto';
```

```
        bWrap.style.transform = 'none';
        bWrap.style.left = x + 'px';
        bWrap.style.top = y + 'px';
        positioned = true;
        updateBubblePos(x);
    }

    function savePosition(x, y) {
        try {
            localStorage.setItem(STORAGE_KEY, JSON.stringify({ x: x, y: y
}));
        } catch (e) {}
    }

    function loadPosition() {
        try {
            var data = localStorage.getItem(STORAGE_KEY);
            if (!data) return null;
            var pos = JSON.parse(data);
            if (typeof pos.x === 'number' && typeof pos.y === 'number') {
                var maxX = window.innerWidth - 96;
                var maxY = window.innerHeight - 96;
                return { x: clamp(pos.x, 0, maxX), y: clamp(pos.y, 0,
maxY) };
            }
        } catch (e) {}
        return null;
    }

    function updateBubblePos(buddyX) {
        var midX = window.innerWidth / 2;
        if (buddyX < midX) {
            bSpeech.style.right = 'auto';
            bSpeech.style.left = (buddyX + 104) + 'px';
            if (bSpeech.className.indexOf('bubble-right') === -1) {
                bSpeech.className =
bSpeech.className.replace('bubble-right', '').replace(/\s+/g, ' ');
                bSpeech.className += ' bubble-right';
            }
        } else {
            bSpeech.style.left = 'auto';
            bSpeech.style.right = (window.innerWidth - buddyX + 8) + 'px';
            bSpeech.className = bSpeech.className.replace('bubble-right',
'').replace(/\s+/g, ' ');
        }
        bSpeech.style.top = bWrap.style.top;
        bSpeech.style.transform = 'translateY(-50%) scale(' +
(bSpeech.className.indexOf('visible') !== -1 ? '1' : '0') + ')';
    }
```

```
    /* === RESTORE SAVED POSITION === */
    var savedPos = loadPosition();
    if (savedPos) {
        applyPosition(savedPos.x, savedPos.y);
    }

    /* === WAKE / FADE === */
    function wakeUp() {
        bWrap.className = bWrap.className.replace(/\bactive\b/g,
'').replace(/\s+/g, ' ');
        bWrap.className += ' active';
        if (fadeTimer) clearTimeout(fadeTimer);
        fadeTimer = setTimeout(function() {
            if (curState !== S_ERROR && curState !== S_LOAD && curState
!== S_DIZZY && !preloading && !isDragging) {
                bWrap.className = bWrap.className.replace(/\bactive\b/g,
'').replace(/\s+/g, ' ');
            }
        }, FADE_SEC * 1000);
    }

    function showMsg(txt, duration) {
        bSpeech.textContent = txt;
        var cls = 'visible';
        if (positioned) {
            var bx = parseInt(bWrap.style.left, 10) || 0;
            if (bx < window.innerWidth / 2) cls += ' bubble-right';
        }
        bSpeech.className = cls;
        if (positioned) {
            updateBubblePos(parseInt(bWrap.style.left, 10) || 0);
        }
        wakeUp();
        if (msgTimer) clearTimeout(msgTimer);
        msgTimer = setTimeout(function() {
            bSpeech.className = '';
        }, duration || MSG_DEFAULT);
    }

    /* --- firnCore customization: Idle-overlay class toggling + pupil
recenter v2.2 --- */
    function setIdleOverlays(on) {
        bWrap.className = bWrap.className.replace(/\bbreathing\b/g,
'').replace(/\beyesActive\b/g, '').replace(/\s+/g, ' ');
        if (on) {
            bWrap.className += ' breathing eyesActive';
        } else {
            if (pupilLeft)  { pupilLeft.style.marginLeft = '0px';
pupilLeft.style.marginTop = '0px'; }
            if (pupilRight) { pupilRight.style.marginLeft = '0px';
```

```
pupilRight.style.marginTop = '0px'; }
        }
    }
    /* --- end --- */

    function setS(s, msg) {
        if (curState === s && s !== S_HAPPY && s !== S_DIZZY) return;
        curState = s;

        /* dizzy uses error gif + CSS shake */
        if (s === S_DIZZY) {
            bImg.src = GIFS[S_ERROR];
            bWrap.className = bWrap.className.replace(/\bdizzy\b/g, '');
            void bWrap.offsetWidth;
            bWrap.className += ' dizzy';
        } else {
            bImg.src = GIFS[s];
            bWrap.className = bWrap.className.replace(/\bdizzy\b/g,
'').replace(/\s+/g, ' ');
        }

        /* firnCore customization: nur im Idle-State Augen/Atmen aktiv */
        setIdleOverlays(s === S_IDLE);

        showMsg(msg || pick(SPEECH[s] || SPEECH[0]));
        if (sTimer) clearTimeout(sTimer);

        if (s === S_HAPPY) {
            sTimer = setTimeout(function() { setS(S_IDLE); }, HAPPY_SEC *
1000);
        }
        if (s === S_ERROR && !pageError) {
            sTimer = setTimeout(function() { setS(S_IDLE); }, ERROR_SEC *
1000);
        }
        if (s === S_LOAD) {
            sTimer = setTimeout(function() { setS(S_HAPPY, 'Fertig!'); },
LOAD_SEC * 1000);
        }
        if (s === S_DIZZY) {
            sTimer = setTimeout(function() {
                bWrap.className = bWrap.className.replace(/\bdizzy\b/g,
'').replace(/\s+/g, ' ');
                setS(S_IDLE, 'Puh... besser.');
            }, DIZZY_SEC * 1000);
        }
    }

    /* === PAGE TOPIC === */
    /* --- firnCore customization: Topic via Proxy v2.3 ---
```

```
     * Die Properties-API (/@api/deki/pages/{id}/properties) ist per
     * NGINX gesperrt (403). helikon-proxy.php liest die Property
     * serverseitig und liefert nur {"topic":"..."} zurueck.
     * Seiten ohne buddy.pagetopic liefern topic = '' (kein Fehler). */
    function topicWarn(msg) {
        if (window.console && console.warn) console.warn('[ServerBuddy]
pagetopic: ' + msg);
    }

    function fetchPageTopic() {
        var pid = 0;
        if (typeof Deki !== 'undefined' && Deki.PageId) {
            pid = Deki.PageId;
        }
        if (!pid) {
            topicWarn('keine Deki.PageId, kein Abruf');
            return;
        }
        var url = '/helikon-proxy.php?topic=' + encodeURIComponent(pid);
        var xhr = new XMLHttpRequest();
        xhr.open('GET', url, true);
        xhr.onload = function() {
            if (xhr.status !== 200) {
                topicWarn('Proxy antwortet mit HTTP ' + xhr.status + ' ('
+ url + ')');
                return;
            }
            try {
                var data = JSON.parse(xhr.responseText);
                var topic = '';
                if (data && typeof data.topic === 'string') {
                    topic = data.topic.replace(/^\s+|\s+$/g, '');
                }
                if (topic) {
                    showMsg(topic, MSG_LONG);
                }
            } catch (e) {
                topicWarn('Antwort ist kein JSON (' + url + ')');
            }
        };
        xhr.onerror = function() {
            topicWarn('Netzwerkfehler (' + url + ')');
        };
        xhr.send();
    }
    /* --- end firnCore customization --- */

    /* === NAVIGATION === */
    function goToChat() {
        curState = -1;
```

```
        setS(S_HAPPY, 'Komm mit!');
        setTimeout(function() {
            window.location.href = CHAT_URL;
        }, 500);
    }

    function onSingleClick() {
        if (isBubbleVisible()) {
            hideBubble();
        } else {
            curState = -1;
            setS(S_HAPPY, pick(HELLOS));
        }
    }

    /* === IDLE === */
    function resetIdle() {
        lastAct = Date.now();
        yawnTriggered = false;
        if (curState === S_SLEEP && !leaving && !preloading) setS(S_IDLE);
    }

    function checkIdle() {
        if (leaving || pageError || preloading) return;
        var sec = Math.floor((Date.now() - lastAct) / 1000);

        /* firnCore customization: G\u00E4hn-\u00DCbergang kurz vor Sleep
v2.2 */
        if (curState === S_IDLE && !yawnTriggered && sec >= (SLEEP_SEC -
YAWN_LEAD_SEC) && sec < SLEEP_SEC) {
            yawnTriggered = true;
            bWrap.className += ' yawning';
            showMsg('*g\u00E4hn*', 2000);
            setTimeout(function() {
                bWrap.className = bWrap.className.replace(/\byawning\b/g,
'').replace(/\s+/g, ' ');
            }, 900);
        }

        if (sec >= SLEEP_SEC && curState === S_IDLE) setS(S_SLEEP);
    }

    /* === ERROR CHECK === */
    function checkPageError() {
        var errEl = document.querySelector('.errormsg.systemmsg');
        var hasErr = (errEl && errEl.offsetParent !== null);
        if (hasErr && !pageError) {
            pageError = true;
            if (sTimer) clearTimeout(sTimer);
            curState = -1;
```

```
        setS(S_ERROR, 'Systemfehler!');
    } else if (!hasErr && pageError) {
        pageError = false;
        setS(S_IDLE, 'Wieder OK!');
    }
}

/* === PRELOADER === */
function isPreloaderVisible() {
    var pl = document.getElementById('preloader');
    if (!pl) return false;
    return pl.className.indexOf('show') !== -1;
}

function checkPreloader() {
    var plShow = isPreloaderVisible();
    if (plShow && !preloading) {
        preloading = true;
        wakeUp();
        curState = -1;
        setS(S_SLEEP, 'zzz...');
    } else if (!plShow && preloading) {
        preloading = false;
        curState = -1;
        setS(S_LOAD, 'Danke f\u00FCrs Warten...');
        setTimeout(function() {
            leaving = false;
            setS(S_IDLE, 'Bereit!');
            setTimeout(fetchPageTopic, TOPIC_DELAY * 1000);
        }, 2000);
    }
}

/* === DRAG & DROP === */
function getEventXY(e) {
    if (e.touches && e.touches.length > 0) {
        return { x: e.touches[0].clientX, y: e.touches[0].clientY };
    }
    return { x: e.clientX, y: e.clientY };
}

function onDragStart(e) {
    if (e.button && e.button !== 0) return;
    var pt = getEventXY(e);
    var rect = bWrap.getBoundingClientRect();
    dragStartX  = pt.x;
    dragStartY  = pt.y;
    dragOffsetX = pt.x - rect.left;
    dragOffsetY = pt.y - rect.top;
    isDragging  = false;
```

```
        wasDragged  = false;

        /* reset shake tracking */
        shakeHistory = new Array();
        lastDragY   = pt.y;
        lastDragDir = 0;
    }

    function onDragMove(e) {
        if (dragOffsetX === 0 && dragOffsetY === 0) return;
        var pt = getEventXY(e);
        var dx = Math.abs(pt.x - dragStartX);
        var dy = Math.abs(pt.y - dragStartY);

        if (!isDragging && (dx > DRAG_THRESHOLD || dy > DRAG_THRESHOLD)) {
            isDragging = true;
            wasDragged = true;
            bWrap.className = bWrap.className.replace(/\bdragging\b/g,
'') + ' dragging active';
            hideBubble();
        }

        if (!isDragging) return;
        e.preventDefault();

        var maxX = window.innerWidth - 96;
        var maxY = window.innerHeight - 96;
        var nx = clamp(pt.x - dragOffsetX, 0, maxX);
        var ny = clamp(pt.y - dragOffsetY, 0, maxY);

        applyPosition(nx, ny);

        /* === SHAKE DETECTION (Y-axis direction changes) === */
        var deltaY = pt.y - lastDragY;
        if (Math.abs(deltaY) > SHAKE_MIN_DIST) {
            var dir = deltaY > 0 ? 1 : -1;
            if (lastDragDir !== 0 && dir !== lastDragDir) {
                shakeHistory.push(Date.now());
            }
            lastDragDir = dir;
            lastDragY = pt.y;
        }

        /* clean old entries */
        var now = Date.now();
        var cleaned = new Array();
        var si;
        for (si = 0; si < shakeHistory.length; si++) {
            if (now - shakeHistory[si] < SHAKE_WINDOW) {
                cleaned.push(shakeHistory[si]);
```

```
            }
        }
        shakeHistory = cleaned;

        if (shakeHistory.length >= SHAKE_THRESHOLD) {
            shakeHistory = new Array();
            if (curState !== S_DIZZY) {
                curState = -1;
                setS(S_DIZZY);
            }
        }
    }

    function onDragEnd(e) {
        if (isDragging) {
            var rect = bWrap.getBoundingClientRect();
            savePosition(rect.left, rect.top);
        }
        bWrap.className = bWrap.className.replace(/\bdragging\b/g,
'').replace(/\s+/g, ' ');
        isDragging  = false;
        dragOffsetX = 0;
        dragOffsetY = 0;
        shakeHistory = new Array();
        lastDragDir  = 0;
    }

    /* mouse drag */
    bWrap.addEventListener('mousedown', function(e) {
        onDragStart(e);
        /* long press for chat */
        pressTimer = setTimeout(function() {
            if (!wasDragged) goToChat();
        }, 800);
    });
    document.addEventListener('mousemove', function(e) { onDragMove(e);
});
    document.addEventListener('mouseup', function(e) {
        if (pressTimer) clearTimeout(pressTimer);
        onDragEnd(e);
    });

    /* touch drag */
    bWrap.addEventListener('touchstart', function(e) {
        onDragStart(e);
        pressTimer = setTimeout(function() {
            if (!wasDragged) {
                e.preventDefault();
                goToChat();
            }
```

```
        }, 800);
    }, { passive: false });
    document.addEventListener('touchmove', function(e) {
        if (isDragging) onDragMove(e);
    }, { passive: false });
    document.addEventListener('touchend', function(e) {
        if (pressTimer) clearTimeout(pressTimer);
        onDragEnd(e);
    });
    bWrap.addEventListener('touchmove', function() {
        if (pressTimer) clearTimeout(pressTimer);
    });

    /* === IDLE TRACKERS === */
    document.addEventListener('mousemove', resetIdle);
    document.addEventListener('scroll', resetIdle);
    document.addEventListener('keydown', resetIdle);
    document.addEventListener('touchstart', resetIdle);

    bWrap.addEventListener('mouseenter', function() { wakeUp(); });
    bWrap.addEventListener('mouseleave', function() {
        if (pressTimer) clearTimeout(pressTimer);
    });

    /* === LINK CLICKS === */
    document.addEventListener('click', function(e) {
        resetIdle();
        var t = e.target;
        var link = null;
        if (t.tagName === 'A') {
            link = t;
        } else if (t.parentNode && t.parentNode.tagName === 'A') {
            link = t.parentNode;
        }
        if (link) {
            var href = link.getAttribute('href') || '';
            if (href && href.charAt(0) !== '#' &&
href.indexOf('javascript:') !== 0) {
                leaving = true;
                curState = -1;
                setS(S_SLEEP, 'Bis gleich...');
            } else {
                setS(S_HAPPY);
            }
        }
    });

    /* FIX: Nicht GIF wechseln beim Seitenwechsel (Request wird
abgebrochen = kaputtes Bild).
     * Stattdessen: Buddy ausblenden. Auf neuer Seite startet er frisch.
```

```
*/
    window.addEventListener('beforeunload', function() {
        leaving = true;
        bWrap.style.opacity = '0';
        bSpeech.className = '';
    });

    /* === CLICK: single = dismiss/greet, double = chat === */
    bWrap.addEventListener('click', function(e) {
        e.stopPropagation();
        e.preventDefault();
        if (wasDragged) {
            wasDragged = false;
            return;
        }
        clickCount++;
        if (clickCount === 1) {
            clickTimer = setTimeout(function() {
                clickCount = 0;
                onSingleClick();
            }, 350);
        } else if (clickCount === 2) {
            clearTimeout(clickTimer);
            clickCount = 0;
            goToChat();
        }
    });

    /* === GLOBAL ERRORS === */
    window.addEventListener('error', function() { setS(S_ERROR); });

    /* === VISIBILITY === */
    document.addEventListener('visibilitychange', function() {
        if (document.hidden) {
            setS(S_SLEEP, '*g\u00E4hn*');
        } else {
            leaving = false;
            resetIdle();
            setS(S_IDLE, 'Wieder da!');
        }
    });

    /* === WINDOW RESIZE: keep buddy in bounds === */
    window.addEventListener('resize', function() {
        if (!positioned) return;
        var cx = parseInt(bWrap.style.left, 10) || 0;
        var cy = parseInt(bWrap.style.top, 10) || 0;
        var maxX = window.innerWidth - 96;
        var maxY = window.innerHeight - 96;
        var nx = clamp(cx, 0, maxX);
```

```
        var ny = clamp(cy, 0, maxY);
        applyPosition(nx, ny);
        savePosition(nx, ny);
    });

    /* --- firnCore customization: Augen folgen der Maus v2.2 --- */
    function updateEyeTracking(clientX, clientY) {
        if (!pupilLeft || !pupilRight) return;
        if (curState !== S_IDLE) return;
        var rect = bWrap.getBoundingClientRect();
        var cx = rect.left + rect.width * 0.47;
        var cy = rect.top + rect.height * 0.37;
        var dx = clientX - cx;
        var dy = clientY - cy;
        var dist = Math.sqrt(dx * dx + dy * dy);
        if (dist < 1) return;
        var strength = dist > EYE_TRACK_DIST ? 1 : dist / EYE_TRACK_DIST;
        var ox = (dx / dist) * EYE_MOVE_RADIUS * strength;
        var oy = (dy / dist) * EYE_MOVE_RADIUS * strength;
        pupilLeft.style.marginLeft  = ox + 'px';
        pupilLeft.style.marginTop   = oy + 'px';
        pupilRight.style.marginLeft = ox + 'px';
        pupilRight.style.marginTop  = oy + 'px';
    }
    document.addEventListener('mousemove', function(e) {
updateEyeTracking(e.clientX, e.clientY); });
    document.addEventListener('touchmove', function(e) {
        if (e.touches && e.touches.length > 0) {
            updateEyeTracking(e.touches[0].clientX, e.touches[0].clientY);
        }
    }, { passive: true });

    /* --- firnCore customization: Blinzeln v2.2 --- */
    function scheduleBlink() {
        var delay = BLINK_MIN_MS + Math.random() * (BLINK_MAX_MS -
BLINK_MIN_MS);
        setTimeout(function() {
            if (curState === S_IDLE && eyeLeft && eyeRight) {
                eyeLeft.className  = 'buddyEye buddyEyeLeft blinking';
                eyeRight.className = 'buddyEye buddyEyeRight blinking';
                setTimeout(function() {
                    eyeLeft.className  = 'buddyEye buddyEyeLeft';
                    eyeRight.className = 'buddyEye buddyEyeRight';
                }, 200);
            }
            scheduleBlink();
        }, delay);
    }
    if (eyeLeft && eyeRight) scheduleBlink();
```

```
    /* --- firnCore customization: Ambient-Idle-Events, Frequenz steigt
mit Verweildauer v2.2 --- */
    var pageLoadTime = Date.now();

    function getIdlePhrase() {
        var h = new Date().getHours();
        if (h >= 23 || h < 6) return pick(IDLE_NIGHT);
        if (h >= 6 && h < 10) return pick(IDLE_MORNING);
        return pick(IDLE_AMBIENT);
    }

    function scheduleIdleEvent() {
        var dwell = Date.now() - pageLoadTime;
        var minD, maxD;
        if (dwell > IDLE_RAMP_MS) {
            minD = IDLE_EVENT_MIN_LATE;
            maxD = IDLE_EVENT_MAX_LATE;
        } else {
            minD = IDLE_EVENT_MIN_START;
            maxD = IDLE_EVENT_MAX_START;
        }
        var delay = minD + Math.random() * (maxD - minD);
        setTimeout(function() {
            if (curState === S_IDLE && !isDragging && !preloading &&
!leaving) {
                showMsg(getIdlePhrase(), MSG_DEFAULT);
            }
            scheduleIdleEvent();
        }, delay);
    }
    scheduleIdleEvent();
    /* --- end Idle-Pers\u00F6nlichkeit v2.2 --- */

    /* === PRELOAD GIFS === */
    var i;
    for (i = 0; i < 5; i++) {
        var preImg = new Image();
        preImg.src = GIFS[i];
    }

    /* === INIT === */
    /* FIX: Warten bis Loading-GIF geladen ist, bevor Buddy angezeigt
wird */
    if (isPreloaderVisible()) {
        preloading = true;
        wakeUp();
        setS(S_SLEEP, 'zzz...');
    } else {
        bImg.src = GIFS[S_LOAD];
        bImg.onload = function() {
```

```
            bImg.onload = null;
            setS(S_LOAD, 'Danke f\u00FCrs Warten. Bin scho alt...');
            setTimeout(function() {
                leaving = false;
                setS(S_IDLE, 'Bereit!');
                setTimeout(fetchPageTopic, TOPIC_DELAY * 1000);
            }, 2000);
        };
        /* Fallback falls onload nicht feuert */
        setTimeout(function() {
            if (bImg.onload) {
                bImg.onload = null;
                setS(S_LOAD, 'Danke f\u00FCrs Warten. Bin scho alt...');
                setTimeout(function() {
                    leaving = false;
                    setS(S_IDLE, 'Bereit!');
                    setTimeout(fetchPageTopic, TOPIC_DELAY * 1000);
                }, 2000);
            }
        }, 3000);
    }

    setInterval(checkIdle, 1000);
    setInterval(checkPageError, 1000);
    setInterval(checkPreloader, 300);

    /* === HEADER HOVER FACTS === */
    var headerFacts = new Object();
    var hf = headerFacts;

    hf['Charles Babbage'] = new Array(
        'Der Urvater aller Computer! Hat die Analytical Engine nie fertig
gebaut. Kennen wir das nicht alle?',
        'Babbage hasste Strassenmusikanten. H\u00E4tte er
Slack-Notifications erlebt...',
        'Fun fact: Die britische Regierung gab ihm 17000 Pfund. Das w\
u00E4re heute \u00FCber 3 Millionen!'
    );
    hf['Ada Lovelace'] = new Array(
        'Die erste Programmiererin der Welt! Und das ohne Stack
Overflow.',
        'Ada schrieb den ersten Algorithmus bevor es Computer gab. Talk
about early adopter!',
        'Tochter von Lord Byron. Poesie und Code liegen n\u00E4her als
man denkt.'
    );
    hf['Konrad Zuse'] = new Array(
        'Z3: Der erste funktionierende Computer der Welt. Made in
Germany!',
        'Zuse hat nebenbei auch die erste h\u00F6here Programmiersprache
```

```
erfunden: Plankalk\u00FCl.',
        'Er baute den Z1 im Wohnzimmer seiner Eltern. Der erste
Nerd-Keller war ein Wohnzimmer.'
    );
    hf['Howard Aiken'] = new Array(
        'Harvard Mark I: 16 Meter lang, 5 Tonnen schwer. Und weniger
Rechenpower als deine Smartwatch.',
        'Aiken sagte: Die Welt braucht vielleicht 5 Computer. Ups.',
        'Der Mark I klang wie eine N\u00E4hfabrik. Noise-Cancelling gabs
noch nicht.'
    );
    hf['Alan Turing'] = new Array(
        'Hat den Enigma-Code geknackt und den Krieg verk\u00FCrzt. Nicht
schlecht f\u00FCr einen Mathematiker.',
        'Der Turing-Test: Kann eine Maschine denken? Frag mich nochmal in
10 Jahren.',
        'Turing liebte Langstreckenl\u00E4ufe. Debug-Methode: Rennen bis
die L\u00F6sung kommt.'
    );
    hf['John von Neumann'] = new Array(
        'Die Von-Neumann-Architektur: Jeder Computer den du je benutzt
hast basiert darauf.',
        'Er konnte ganze B\u00FCcher auswendig zitieren. Menschliches RAM
auf Maximum.',
        'Von Neumann rechnete schneller im Kopf als fr\u00FChere
Computer. Kein Witz.'
    );
    hf['Steve Jobs bis 1976'] = new Array(
        'Apple wurde in einer Garage gegr\u00FCndet. Jedes Startup
braucht eine Origin-Story.',
        'Jobs war barfuss bei wichtigen Meetings. Der erste Casual
Friday.',
        'Er ging nach Indien um sich selbst zu finden. Kam zur\u00FCck
und erfand Apple.'
    );
    hf['IBM PC 5150'] = new Array(
        'Der IBM PC: 4.77 MHz, 16 KB RAM. Dein Browser-Tab braucht heute
mehr.',
        '1981 kostete er 1565 Dollar. Inflation-bereinigt ca. 5000
Dollar. F\u00FCr 16 KB RAM!',
        'IBM dachte PCs w\u00E4ren ein Nischenprodukt. Klassischer
Management-Move.'
    );
    hf['Apple II'] = new Array(
        'Der Apple II: Erster Computer mit Farbgrafik f\u00FCr zuhause.
Revolution!',
        'Wozniak hat den Apple II quasi alleine designed. Jobs hat ihn
verkauft.',
        'VisiCalc auf dem Apple II war die erste Killer-App. Excel-Urahn!'
    );
```

```
    hf['Steve Jobs bis 1985'] = new Array(
        'Von Apple gefeuert. Von der eigenen Firma. Das muss man erstmal
schaffen.',
        'Jobs Reality Distortion Field: Er konnte dich \u00FCberzeugen
dass schwarz weiss ist.',
        'Nach Apple gr\u00FCndete er NeXT. Spoiler: Er kommt zur\u00FCck.'
    );
    hf['Diskette'] = new Array(
        'Speichern-Icon bis heute! Kinder fragen: Was ist dieses
3D-gedruckte Save-Symbol?',
        '1.44 MB. Da passt nicht mal ein Handyfoto drauf.',
        'Das befriedigende Klick-Ger\u00E4usch beim Einschieben. ASMR der
90er.'
    );
    hf['Game Boy'] = new Array(
        'Tetris auf dem Game Boy: Produktivit\u00E4tskiller seit 1989.',
        'Der Game Boy \u00FCberlebte einen Bombenangriff im Golfkrieg.
Nokia-Level H\u00E4rte.',
        '118 Millionen St\u00FCck verkauft. Und alle spielten Tetris.'
    );
    hf['NeXTstation'] = new Array(
        'Tim Berners-Lee erfand das Web auf einem NeXT. Danke, Steve!',
        'Der NeXT-W\u00FCrfel kostete 6500 Dollar. Design hat seinen
Preis.',
        'NeXTSTEP wurde zu macOS. Die beste Rache ist Erfolg.'
    );
    hf['CD-ROM'] = new Array(
        '700 MB! Das war mal unvorstellbar viel Speicher.',
        'Bitte nicht die gl\u00E4nzende Seite anfassen. BITTE!',
        'AOL-CDs als Untersetzer. Recycling der 90er.'
    );
    hf['Netscape Navigator'] = new Array(
        'Netscape: Der Browser der das Internet popul\u00E4r machte.
RIP.',
        'Der Browserkrieg hat begonnen! Netscape vs. Microsoft, Runde 1.',
        'Aus Netscape wurde Mozilla Firefox. Der Phoenix aus der Asche!'
    );
    hf['Internet Explorer'] = new Array(
        'Internet Explorer: Nur genutzt um einen anderen Browser
runterzuladen.',
        'IE6 hat eine ganze Generation von Webentwicklern traumatisiert.',
        'Ruhe in Frieden, IE. Du warst... ein Browser. Irgendwie.'
    );
    hf['Bill Gates'] = new Array(
        '640 KB sollten f\u00FCr jeden reichen. Legend\u00E4res
Fehlzitat!',
        'Von Harvard abgebrochen. Eltern hassen diesen Trick.',
        'Ctrl+Alt+Delete war ein Fehler, sagt er selbst. Wenigstens
ehrlich.'
    );
```

```
    hf['DVD und Blu-ray'] = new Array(
        'DVD: Rewind war gestern! Aber Regionalcodes waren die neue
Nerverei.',
        'Blu-ray vs HD-DVD: Der letzte grosse Formatkrieg.',
        'Wer hat noch eine DVD-Sammlung im Regal? Sei ehrlich.'
    );
    hf['Geschichte von Linux'] = new Array(
        'Linus Torvalds: "Ich mache ein freies Betriebssystem. Wird wohl
nichts Grosses."',
        'Linux l\u00E4uft auf Supercomputern, Handys, K\u00FChlschr\
u00E4nken und dem Mars-Rover!',
        'BTW, I use Arch. Der Veganer-Spruch der IT.'
    );
    hf['List of Linux distributions'] = new Array(
        'Es gibt \u00FCber 600 Linux-Distros. F\u00FCr jedes Problem eine
eigene Distribution.',
        'Ubuntu, Debian, Arch, Fedora, Mint... Die Qual der Wahl.',
        'Jeder Linux-User hat schon 3x die Distro gewechselt. Mindestens.'
    );
    hf['Handheld'] = new Array(
        'Von Handhelds zu Smartphones. Die Evolution der Ablenkung.',
        'Palm Pilot! Wer erinnert sich an Graffiti-Schrifterkennung?',
        'Handhelds: Als Taschencomputer noch eine eigene Kategorie waren.'
    );
    hf['Personal Digital Assistant'] = new Array(
        'PDA: Personal Digital Assistant. Heute sagt man einfach Handy.',
        'Mit dem Stylus auf dem PDA tippen. Das war Premium-UX!',
        'Newton MessagePad: Apples erster Flop. Oder seiner Zeit voraus?'
    );
    hf['Google'] = new Array(
        'Googles erster Server war in Lego-Steinen eingebaut. Kein Witz!',
        'Googles erstes B\u00FCro: Eine Garage. Schon wieder eine
Garage.',
        '"Don\'t be evil" hiess es mal. Heute heisst es... naja.'
    );
    hf['Notebook'] = new Array(
        'Vom 10-Kilo-Laptop zum Ultrabook. Unsere Rucks\u00E4cke danken
es.',
        'Erster Laptop: Osborne 1, 1981. 11 Kilo. Armtraining inklusive.',
        'Laptop im Bett: Modernes Lagerfeuer f\u00FCr digitale Nomaden.'
    );
    hf['USB-Stick'] = new Array(
        'USB: Braucht immer 3 Versuche zum Einstecken. Immer.',
        'USB-C: Endlich egal welche Seite! Nur 20 Jahre hat es gedauert.',
        'Wer hat nicht mindestens 5 USB-Sticks in der Schublade liegen?'
    );
    hf['Youtube'] = new Array(
        'Erstes Youtube-Video: "Me at the zoo". 19 Sekunden die alles
\u00E4nderten.',
        'Youtube frisst mehr Bandbreite als TV. Die Zukunft ist hier.',
```

```
        'Katzenvideos: Der wahre Grund warum das Internet erfunden wurde.'
    );
    hf['Tim Berners-Lee'] = new Array(
        'Hat das World Wide Web erfunden. Und nie Milliarden damit
verdient. Respekt!',
        'Erster Webserver lief auf einem NeXT-W\u00FCrfel. Mit einem
Zettel: "Nicht ausschalten!"',
        'Ohne ihn w\u00E4rst du jetzt nicht hier. Danke, Tim!'
    );
    hf['Containervirtualisierung'] = new Array(
        'Docker: Funktioniert auf meinem Computer! Endlich auch auf
deinem.',
        'Container sind wie Schiffscontainer: Alles drin, \u00FCberall
einsetzbar.',
        'kubectl get pods: Der Befehl der DevOps-Engineers nicht schlafen
l\u00E4sst.'
    );
    hf['Smartphone OS'] = new Array(
        'Android vs iOS: Der ewige Glaubenskrieg. Schlimmer als Tabs vs
Spaces.',
        'Es gab mal Windows Phone. Moment der Stille bitte.',
        'Dein Smartphone hat mehr Power als die Apollo-11-Rakete. Und du
spielst Candy Crush.'
    );
    hf['SaaS'] = new Array(
        'Software as a Service: Fr\u00FCher kaufte man Software. Heute
mietet man sie. F\u00FCr immer.',
        'SaaS: Wo jede App ein monatliches Abo ist. Dein Konto weint
leise.',
        'Die Cloud ist nur der Computer von jemand anderem. Romantisch,
oder?'
    );
    hf['Cloud Computing'] = new Array(
        'Die Cloud: Klingt fluffy, ist ein riesiges Rechenzentrum
irgendwo in Irland.',
        'Cloud Computing: Weil "anderer-Leute-Computer" nicht so cool
klingt.',
        'Fr\u00FCher hatte man Server im Keller. Jetzt hat Amazon sie.
Fortschritt?'
    );

    var hoverCooldown = false;
    var areas = document.querySelectorAll('area[title]');
    var ai;
    for (ai = 0; ai < areas.length; ai++) {
        (function(area) {
            area.addEventListener('mouseenter', function() {
                if (hoverCooldown) return;
                var title = area.getAttribute('title') || '';
                var facts = headerFacts[title];
```

```
                if (!facts) return;
                hoverCooldown = true;
                curState = -1;
                setS(S_HAPPY);
                showMsg(pick(facts), MSG_LONG);
                setTimeout(function() { hoverCooldown = false; },
HOVER_COOLDOWN);
            });
        })(areas[ai]);
    }

    /* === PUBLIC API === */
    window.serverBuddy = new Object();
    window.serverBuddy.setState = function(name, msg) {
        var map = new Object();
        map['idle'] = 0; map['sleeping'] = 1; map['happy'] = 2;
        map['error'] = 3; map['loading'] = 4; map['dizzy'] = 5;
        if (map[name] !== undefined) setS(map[name], msg);
    };
    window.serverBuddy.say = showMsg;
    window.serverBuddy.moveTo = function(x, y) {
        applyPosition(x, y);
        savePosition(x, y);
    };
    window.serverBuddy.resetPosition = function() {
        try { localStorage.removeItem(STORAGE_KEY); } catch(e) {}
        positioned = false;
        bWrap.style.left = '';
        bWrap.style.top = '';
        bWrap.style.right = '16px';
        bWrap.style.transform = 'translateY(-50%)';
        bSpeech.style.left = '';
        bSpeech.style.right = '120px';
        bSpeech.style.top = '50%';
    };
})();
```