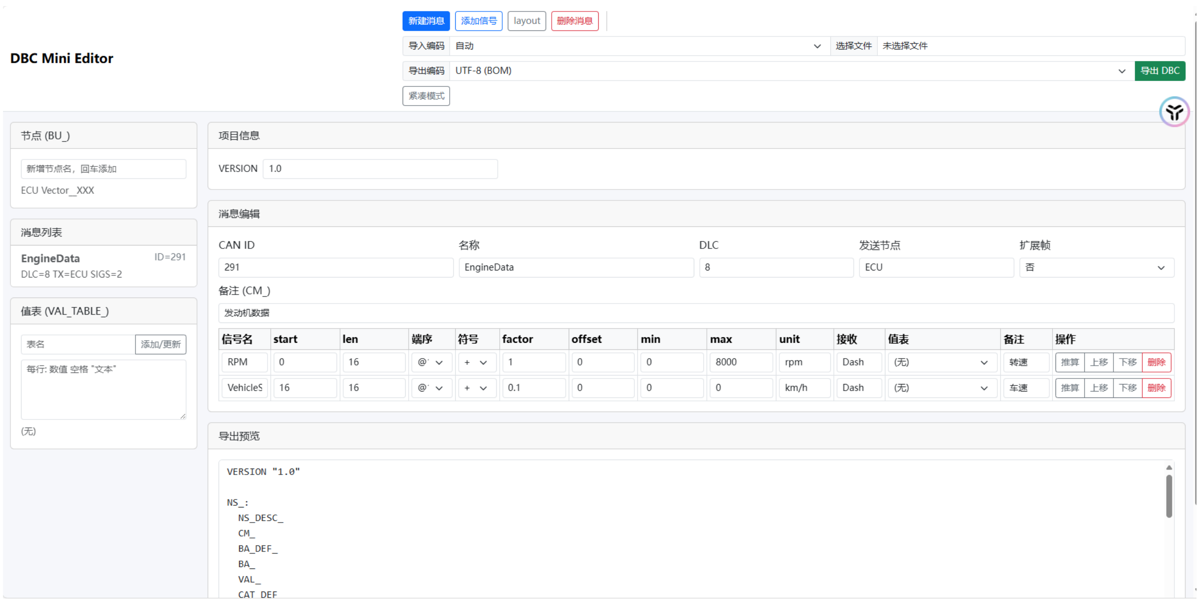
使用说明

1. 如何使用

很简单自己摸索吧！



DBC Mini Editor

导入编码 自动　选择文件 未选择文件

导出编码 UTF-8 (BOM)

节点 (BU_)

新增节点名，回车添加

ECU Vector__XXX

消息列表

EngineData ID=291
DLC=8 TX=ECU SIGS=2

值表 (VAL_TABLE_)

表名 添加/更新

每行: 数值 空格 "文本"

(无)

项目信息

VERSION 1.0

消息编辑

CAN ID	名称	DLC	发送节点	扩展帧
291	EngineData	8	ECU	否

备注 (CM_)

发动机数据

信号名	start	len	端序	符号	factor	offset	min	max	unit	接收	值表	备注	操作
RPM	0	16	@'	+	1	0	0	8000	rpm	Dash	(无)	转速	推算 上移 下移 删除
VehicleS	16	16	@'	+	0.1	0	0	0	km/h	Dash	(无)	车速	推算 上移 下移 删除

导出预览

```
VERSION "1.0"

NS_:
  NS_DESC_
  CM_
  BA_DEF_
  BA_
  VAL_
  CAT_DEF
```

Image 1

2. 注意事项

1. 默认使用的框架是`BootStrap`+`Jquery`，如果对页面不了解请不要删除。
2. `app\_source.js`是源码，`app.js`是混淆后代码。
3. 启动页面`index.html`。

3. 源码

```
(function () {
    'use strict';

    /* ===== 错误浮层，防止白屏 ===== */
    (function () {
        var el = document.getElementById('errOverlay');
        function show(msg) { el.textContent = 'Error: ' + msg; el.style.display = 'block'; }
        window.addEventListener('error', function (e) { show(e.message || String(e.error || e)); });
        window.addEventListener('unhandledrejection', function (e) { show((e.reason && e.reason.message) || String(e.reason)); });
    })();

    /* ===== 工具函数 ===== */
    function escQuote(s) { return String(s || '').replace(/\\/g, '\\\\').replace(/"/g, '\\"'); }
    function toCRLF(s) { return s.replace(/\r?\n/g, '\r\n'); }
    function withBOM(u8) { var out = new Uint8Array(u8.length + 3); out[0] = 0xEF; out[1] = 0xBB; out[2] = 0xBF; out.set(u8, 3); return out; }
    function stripBOM(s) { return s.replace(/^\uFEFF/, ''); }
    /* ===== 数据模型 ===== */
    function Signal() {
        this.name = ''; this.start = 0; this.len = 1; this.le = true;
this.signed = false;
        this.factor = 1; this.offset = 0; this.min = 0; this.max = 0; this.unit
= '';
        this.receivers = ['Vector__XXX']; this.comment = ''; this.valTable =
'';
    }
    Signal.prototype.toDBC = function () {
        var endian = this.le ? '1' : '0';
        var sign = this.signed ? '-' : '+';
        var recv = (this.receivers && this.receivers.length ?
this.receivers.join(',') : 'Vector__XXX');
        return ' SG_ ' + this.name + ' : ' + this.start + '|' + this.len + '@'
+ endian + sign +
            ' (' + this.factor + ',' + this.offset + ') [' + this.min + '|' +
this.max + '] "' + escQuote(this.unit) + '" ' + recv;
    };

    function Message() { this.canId = 0; this.name = 'Msg'; this.dlc = 8;
this.tx = 'ECU'; this.ext = false; this.comment = ''; this.signals = []; }
    Message.prototype.toDBC = function () {
        var head = 'BO_ ' + this.canId + ' ' + this.name + ': ' + this.dlc + '
' + this.tx;
        var sigs = this.signals.map(function (s) { return s.toDBC();
}).join('\n');
        return sigs ? (head + '\n' + sigs) : head;
    };

    function Project() { this.version = '1.0'; this.nodes = ['ECU',
'Vector__XXX']; this.messages = []; this.valueTables = {}; }
    Project.prototype.addNode = function (n) { if (n && this.nodes.indexOf(n) <
0) this.nodes.push(n); };
    Project.prototype.addMessage = function (m) { this.messages.push(m);
this.addNode(m.tx); };

    // 导出顺序
    Project.prototype.toDBC = function () {
        var NS = ["NS_DESC_", "CM_", "BA_DEF_", "BA_", "VAL_", "CAT_DEF_",
"CAT_", "FILTER", "BA_DEF_DEF_", "EV_DATA_", "ENVVAR_DATA_", "SGTYPE_",
"SGTYPE_VAL_", "BA_DEF_SGTYPE_", "BA_SGTYPE_", "SIG_TYPE_REF_", "VAL_TABLE_",
"SIG_GROUP_", "SIG_VALTYPE_", "SIGTYPE_VALTYPE_", "BO_TX_BU_", "BA_DEF_REL_",
"BA_REL_", "BA_DEF_DEF_REL_", "BU_SG_REL_", "BU_EV_REL_", "BU_BO_REL_",
"SG_MUL_VAL_"];
        var lines = [], i, j, k, t, m, s, items, parts;

        // 头
        lines.push('VERSION "' + escQuote(this.version) + '"');
        lines.push('');
        lines.push('NS_:'); for (i = 0; i < NS.length; i++) lines.push('   ' +
NS[i]);
        lines.push(''); lines.push('BS_:'); lines.push('');

        // BU_ 节
        lines.push('BU_: ' + this.nodes.join(' '));
        lines.push('');

        // VAL_TABLE_
        for (var name in this.valueTables) {
            if (!this.valueTables.hasOwnProperty(name)) continue;
            items = this.valueTables[name] || []; parts = [];
            for (k = 0; k < items.length; k++) parts.push(items[k][0] + ' "' +
escQuote(items[k][1]) + '"');
            lines.push('VAL_TABLE_ ' + name + ' ' + parts.join(' ') + ' ;');
        }
        if (Object.keys(this.valueTables).length) lines.push('');

        // 消息与信号
        for (j = 0; j < this.messages.length; j++) {
            lines.push(this.messages[j].toDBC());
            lines.push('');
        }

        // 备注
        for (j = 0; j < this.messages.length; j++) {
            m = this.messages[j];
            if (m.comment) lines.push('CM_ BO_ ' + m.canId + ' "' +
escQuote(m.comment) + '";');
            for (k = 0; k < m.signals.length; k++) {
                s = m.signals[k];
                if (s.comment) lines.push('CM_ SG_ ' + m.canId + ' ' + s.name +
' "' + escQuote(s.comment) + '";');
            }
        }

        // VAL_
        for (j = 0; j < this.messages.length; j++) {
            m = this.messages[j];
            for (k = 0; k < m.signals.length; k++) {
                s = m.signals[k];
                if (s.valTable && this.valueTables[s.valTable]) {
                    items = this.valueTables[s.valTable]; parts = [];
                    for (t = 0; t < items.length; t++) parts.push(items[t][0] +
' "' + escQuote(items[t][1]) + '"');
                    lines.push('VAL_ ' + m.canId + ' ' + s.name + ' ' +
parts.join(' ') + ' ;');
                }
            }
        }

        lines.push('');
        return lines.join('\n');
    };

    /* ===== 计算 ===== */
    function computePhysRange(len, signed, factor, offset) {
        if (!(len >= 1 && len <= 64)) throw new Error('length 需在 1..64');
        var rawMin, rawMax;
        if (signed) { rawMin = -Math.pow(2, len - 1); rawMax = Math.pow(2, len
- 1) - 1; }
```

```
        else { rawMin = 0; rawMax = Math.pow(2, len) - 1; }
        return [rawMin * factor + offset, rawMax * factor + offset];
    }

    /* ===== 解析 ===== */
    // 1) 把文本切成“语句”（引号外遇到 ; 才结束；没有 ; 的行式语句则用换行结束）
    function splitDBCIntoStatements(text) {
        const stmts = [];
        let cur = '';
        let inQuote = false;

        for (let i = 0; i < text.length; i++) {
            const ch = text[i];
            const prev = text[i - 1];

            // 进入/退出引号（支持 \" 转义）
            if (ch === '"' && prev !== '\\') {
                inQuote = !inQuote;
                cur += ch;
                continue;
            }

            // 引号外遇到 ; ：结束一条“语句”
            if (ch === ';' && !inQuote) {
                cur += ch;
                if (cur.trim()) stmts.push(cur.trim());
                cur = '';
                continue;
            }

            // 对于没有 ; 的行式规则（如 BO_/SG_/BU_/VERSION），引号外的换行也结束
            if ((ch === '\n' || ch === '\r') && !inQuote) {
                if (cur.trim()) {
                    stmts.push(cur.trim());
                    cur = '';
                }
                continue;
            }

            cur += ch;
        }

        if (cur.trim()) stmts.push(cur.trim());
        return stmts;
    }

    // 2) 解析入口：用“语句”而不是“行”，并让 CM_ 的正则可跨行
    function parseDBCIntoProject(text, projOut) {
        projOut.messages.length = 0;
        projOut.nodes = ['ECU', 'Vector__XXX'];
        projOut.valueTables = {};

        // 改为按“语句”遍历
        var lines = splitDBCIntoStatements(text);

        // 原有正则
        var reBO = /^\s*BO_\s+(\d+)\s+([A-Za-z_][\w]*)\s*:\s*(\d+)\s+([A-Za-z_]
[\w]*)/;
        var reSG = /^\s*SG_\s+([A-Za-z_][\w]*)\s*:\s*(\d+)\|(\d+)\@([01])
([+-])\s*\(([-+]?[\d.]+),\s*([-+]?[\d.]+)\)\s*\[([-+]?[\d.]+)\|([-+]?
[\d.]+)\]\s*"([^"]*)"\s+(.+)\s*$/;
        var reBU = /^\s*BU_:\s*(.*)$/;
        var reVER = /^\s*VERSION\s+"(.*)"\s*$/;

        // 允许注释字符串跨多行（非贪婪）
        var reCM_BO = /^\s*CM_\s+BO_\s+(\d+)\s+"([\s\S]*?)"\s*;\s*$/;
        var reCM_SG = /^\s*CM_\s+SG_\s+(\d+)\s+([A-Za-z_][\w]*)\s+"
([\s\S]*?)"\s*;\s*$/;

        var reVT = /^\s*VAL_TABLE_\s+([A-Za-z_][\w]*)\s+(.*?)\s*;\s*$/;
        var reVAL = /^\s*VAL_\s+(\d+)\s+([A-Za-z_][\w]*)\s+(.*?)\s*;\s*$/;

        var curMsg = null, i, m, j, k, mm, items, reItem, id, msg, sigName;

        for (i = 0; i < lines.length; i++) {
            var ln = lines[i];

            if ((m = reVER.exec(ln))) { projOut.version = m[1]; continue; }
            if ((m = reBU.exec(ln))) { var ns = (m[1] ||
'').trim().split(/\s+/); if (ns.length && ns[0]) projOut.nodes = ns; continue;
}

            if ((m = reBO.exec(ln))) {
                curMsg = new Message();
                curMsg.canId = parseInt(m[1], 10);
                curMsg.name = m[2];
                curMsg.dlc = parseInt(m[3], 10);
                curMsg.tx = m[4];
                projOut.addMessage(curMsg);
                continue;
            }

            if ((m = reSG.exec(ln))) {
                if (!curMsg) continue;
                var s = new Signal();
                s.name = m[1];
                s.start = parseInt(m[2], 10);
                s.len = parseInt(m[3], 10);
                s.le = (m[4] === '1');
                s.signed = (m[5] === '-');
                s.factor = parseFloat(m[6]);
                s.offset = parseFloat(m[7]);
                s.min = parseFloat(m[8]);
                s.max = parseFloat(m[9]);
                s.unit = m[10];
                s.receivers = m[11]
                    .split(',')
                    .map(function (x) { return x.trim(); })
                    .filter(function (x) { return !!x; });
                curMsg.signals.push(s);
                continue;
            }
            if ((m = reCM_BO.exec(ln))) {
                id = parseInt(m[1], 10);
                msg = null;
                for (j = 0; j < projOut.messages.length; j++) {
                    if (projOut.messages[j].canId === id) { msg =
projOut.messages[j]; break; }
                }
                if (msg) msg.comment = m[2];
                continue;
            }

            if ((m = reCM_SG.exec(ln))) {
                id = parseInt(m[1], 10);
                msg = null;
                for (j = 0; j < projOut.messages.length; j++) {
                    if (projOut.messages[j].canId === id) { msg =
projOut.messages[j]; break; }
                }
                if (msg) {
                    for (k = 0; k < msg.signals.length; k++) {
                        if (msg.signals[k].name === m[2]) {
msg.signals[k].comment = m[3]; break; }
                    }
                }
                continue;
            }

            if ((m = reVT.exec(ln))) {
                items = [];
                reItem = /(-?\d+)\s+"([^"]*)"/g;
                while ((mm = reItem.exec(m[2])) !== null) {
                    items.push([parseInt(mm[1], 10), mm[2]]);
                }
                projOut.valueTables[m[1]] = items;
                continue;
            }

            if ((m = reVAL.exec(ln))) {
                id = parseInt(m[1], 10);
                sigName = m[2];
                items = [];
                reItem = /(-?\d+)\s+"([^"]*)"/g;
                while ((mm = reItem.exec(m[3])) !== null) {
                    items.push([parseInt(mm[1], 10), mm[2]]);
                }
                msg = null;
                for (j = 0; j < projOut.messages.length; j++) {
                    if (projOut.messages[j].canId === id) { msg =
projOut.messages[j]; break; }
                }
                if (msg) {
                    var vtName = '__VAL__' + id + '__' + sigName;
                    projOut.valueTables[vtName] = items;
                    for (k = 0; k < msg.signals.length; k++) {
```

```
                    if (msg.signals[k].name === sigName) {
msg.signals[k].valTable = vtName; break; }
                }
            }
            continue;
        }
    }
}


    /* ===== 值表辅助 ===== */
    function getValTableNames() {
        var names = [], k; for (k in proj.valueTables) { if
(proj.valueTables.hasOwnProperty(k)) names.push(k); }
        names.sort(); return names;
    }
    function buildVtSelect(currentName) {
        var sel = document.createElement('select');
        sel.className = 'form-select form-select-sm sig-vt';
        var op0 = document.createElement('option'); op0.value = '';
op0.textContent = '(无)'; sel.appendChild(op0);
        getValTableNames().forEach(function (n) {
            var o = document.createElement('option'); o.value = n;
o.textContent = n; sel.appendChild(o);
        });
        sel.value = currentName || '';
        return sel;
    }
    function refreshAllSignalVtSelects() {
        var names = getValTableNames();
        var nodes = document.querySelectorAll('.sig-vt');
        for (var i = 0; i < nodes.length; i++) {
            var sel = nodes[i], cur = sel.value;
            sel.innerHTML = '';
            var op0 = document.createElement('option'); op0.value = '';
op0.textContent = '(无)'; sel.appendChild(op0);
            for (var j = 0; j < names.length; j++) {
                var n = names[j], o = document.createElement('option'); o.value
= n; o.textContent = n; sel.appendChild(o);
            }
            sel.value = cur || '';
        }
    }

    /* ===== 状态 / UI 绑定 ===== */
    var els = {
        msgList: document.getElementById('msgList'),
        nodeList: document.getElementById('nodeList'),
        inpNode: document.getElementById('inpNode'),
        msgEditor: document.getElementById('msgEditor'),
        preview: document.getElementById('preview'),
        btnNewMsg: document.getElementById('btnNewMsg'),
        btnNewSig: document.getElementById('btnNewSig'),
        btnDelMsg: document.getElementById('btnDelMsg'),
        btnLayout: document.getElementById('btnLayout'),
        fileImport: document.getElementById('fileImport'),
        selImportEnc: document.getElementById('selImportEnc'),
        btnExport: document.getElementById('btnExport'),
        selExportEnc: document.getElementById('selExportEnc'),
        btnToggleDensity: document.getElementById('btnToggleDensity'),
        // msg fields
        mCanId: document.getElementById('mCanId'),
        mName: document.getElementById('mName'),
        mDlc: document.getElementById('mDlc'),
        mTx: document.getElementById('mTx'),
        mExt: document.getElementById('mExt'),
        mComment: document.getElementById('mComment'),
        // grid
        sigGrid:
document.getElementById('sigGrid').getElementsByTagName('tbody')[0],
        // value tables
        vtName: document.getElementById('vtName'),
        vtItems: document.getElementById('vtItems'),
        vtAdd: document.getElementById('vtAdd'),
        vtList: document.getElementById('vtList'),
        inpVersion: document.getElementById('inpVersion')
    };

    var proj = new Project();
    var selectedMsgIndex = -1;

    function renderNodes() { els.nodeList.textContent = proj.nodes.join(' '); }

    function renderMsgList() {
        els.msgList.innerHTML = '';
        for (var i = 0; i < proj.messages.length; i++) {
            (function (idx) {
                var m = proj.messages[idx];
                var a = document.createElement('button');
                a.type = 'button';
                a.className = 'list-group-item list-group-item-action text-
truncate' + (idx === selectedMsgIndex ? ' active' : '');
                a.title = m.name + '  ID=' + m.canId;
                a.innerHTML = '<div class="d-flex w-100 justify-content-
between"><strong>' + m.name +
                    '</strong><small>ID=' + m.canId + '</small></div><small
class="text-muted">DLC=' + m.dlc + ' TX=' + m.tx + ' SIGS=' + m.signals.length
+ '</small>';
                a.addEventListener('click', function () { selectedMsgIndex =
idx; openMsg(); renderMsgList(); });
                els.msgList.appendChild(a);
            })(i);
        }
    }

    function openMsg() {
        var m = proj.messages[selectedMsgIndex];
        if (!m) { els.msgEditor.style.display = 'none'; return; }
        els.msgEditor.style.display = '';
        els.mCanId.value = String(m.canId);
        els.mName.value = m.name;
        els.mDlc.value = m.dlc;
        els.mTx.value = m.tx;
        els.mExt.value = m.ext ? '1' : '0';
        els.mComment.value = m.comment || '';
        renderSigGrid(m);
        refreshPreview();
    }

    function renderSigGrid(m) {
        els.sigGrid.innerHTML = '';
        for (var i = 0; i < m.signals.length; i++) {
            (function (i) {
                var s = m.signals[i];
                var tr = document.createElement('tr');
                tr.innerHTML = [
                    '<td><input class="form-control form-control-sm" data-
k="name" value="' + s.name + '"></td>',
                    '<td><input class="form-control form-control-sm" data-
k="start" type="number" value="' + s.start + '"></td>',
                    '<td><input class="form-control form-control-sm" data-
k="len" type="number" value="' + s.len + '"></td>',
                    '<td><select class="form-select form-select-sm" data-
k="le"><option value="1"' + (s.le ? ' selected' : '') + '>@1</option><option
value="0"' + (!s.le ? ' selected' : '') + '>@0</option></select></td>',
                    '<td><select class="form-select form-select-sm" data-
k="signed"><option value="0"' + (!s.signed ? ' selected' : '') + '>+</option>
<option value="1"' + (s.signed ? ' selected' : '') + '>-</option></select>
</td>',
                    '<td><input class="form-control form-control-sm" data-
k="factor" type="number" step="any" value="' + s.factor + '"></td>',
                    '<td><input class="form-control form-control-sm" data-
k="offset" type="number" step="any" value="' + s.offset + '"></td>',
                    '<td><input class="form-control form-control-sm" data-
k="min" type="number" step="any" value="' + s.min + '"></td>',
                    '<td><input class="form-control form-control-sm" data-
k="max" type="number" step="any" value="' + s.max + '"></td>',
                    '<td><input class="form-control form-control-sm" data-
k="unit" value="' + s.unit + '"></td>',
                    '<td><input class="form-control form-control-sm" data-
k="receivers" value="' + (s.receivers || []).join(',') + '"></td>',
                    '<td class="vt-col"></td>',
                    '<td><input class="form-control form-control-sm" data-
k="comment" value="' + (s.comment || '') + '"></td>',
                    '<td class="text-nowrap"><div class="btn-group btn-group-
sm"><button class="btn btn-outline-secondary" data-op="auto">推算</button>
<button class="btn btn-outline-secondary" data-op="up">上移</button><button
class="btn btn-outline-secondary" data-op="down">下移</button><button class="btn
btn-outline-danger" data-op="del">删除</button></div></td>'
                ].join('');

                // 值表 select
                var vtTd = tr.children[11];
                var sel = buildVtSelect(s.valTable || '');
                sel.addEventListener('change', function () { s.valTable =
sel.value || ''; refreshPreview(); });
                vtTd.appendChild(sel);
```

```
                    tr.addEventListener('input', function (e) {
                        var k = e.target.getAttribute('data-k'); if (!k) return;
                        var v = e.target.value;
                        if (k === 'start' || k === 'len') v = parseInt(v || '0',
10);
                        if (k === 'le' || k === 'signed') v = (v === '1');
                        if (k === 'factor' || k === 'offset' || k === 'min' || k
=== 'max') v = parseFloat(v || '0');
                        if (k === 'receivers') v = (v.trim() ?
v.split(',').map(function (x) { return x.trim(); }).filter(function (x) {
return !!x; }) : []);
                        s[k] = v; refreshPreview();
                    });

                    tr.addEventListener('click', function (e) {
                        var op = e.target.getAttribute('data-op'); if (!op) return;
                        if (op === 'del') { m.signals.splice(i, 1);
renderSigGrid(m); refreshPreview(); }
                        else if (op === 'up' && i > 0) { var t = m.signals[i - 1];
m.signals[i - 1] = m.signals[i]; m.signals[i] = t; renderSigGrid(m);
refreshPreview(); }
                        else if (op === 'down' && i < m.signals.length - 1) { var
t2 = m.signals[i + 1]; m.signals[i + 1] = m.signals[i]; m.signals[i] = t2;
renderSigGrid(m); refreshPreview(); }
                        else if (op === 'auto') { var rr = computePhysRange(s.len,
s.signed, s.factor, s.offset); s.min = rr[0]; s.max = rr[1]; renderSigGrid(m);
refreshPreview(); }
                    });

                    els.sigGrid.appendChild(tr);
                })(i);
            }
        }

    function refreshPreview() { els.preview.value = proj.toDBC(); }

    /* ===== 值表 左侧列表 ===== */
    function renderVtList() {
        var box = els.vtList; box.innerHTML = '';
        var names = getValTableNames();
        if (!names.length) { box.textContent = '(无)'; return; }
        names.forEach(function (name) {
            var row = document.createElement('div');
            row.className = 'vt-item';
            row.innerHTML = '<span class="badge text-bg-
light">VAL_TABLE_</span>' +
                '<code class="text-truncate" title="' + name + '">' + name +
'</code>' +
                '<span class="text-muted small ms-2">(' +
(proj.valueTables[name] || []).length + ' 项)</span>';
            box.appendChild(row);
        });
    }
    els.vtAdd.addEventListener('click', function () {
        var name = (els.vtName.value || '').trim(); if (!name) return;
        var items = [], lines = (els.vtItems.value || '').split(/\r?\n/), i,
mm;
        var re = /^\s*(-?\d+)\s+"(.*)"\s*$/;
        for (i = 0; i < lines.length; i++) { mm = re.exec(lines[i]); if (mm) {
items.push([parseInt(mm[1], 10), mm[2]]); } }
        proj.valueTables[name] = items; renderVtList(); refreshPreview();
refreshAllSignalVtSelects();
    });

    /* ===== 输入与按钮事件 ===== */
    els.inpNode.addEventListener('keydown', function (e) {
        if (e.key === 'Enter') {
            var n = (els.inpNode.value || '').trim();
            if (n) { proj.addNode(n); els.inpNode.value = ''; renderNodes();
refreshPreview(); }
        }
    });

    els.btnNewMsg.addEventListener('click', function () {
        var m = new Message();
        m.name = 'NewMsg';
        m.canId = proj.messages.length ? (proj.messages[proj.messages.length -
1].canId + 1) : 256;
        proj.addMessage(m);
        selectedMsgIndex = proj.messages.length - 1;
        renderMsgList(); openMsg();
    });

    els.btnDelMsg.addEventListener('click', function () {
        if (selectedMsgIndex < 0) return;
        proj.messages.splice(selectedMsgIndex, 1);
        selectedMsgIndex = Math.min(selectedMsgIndex, proj.messages.length -
1);
        renderMsgList(); openMsg(); refreshPreview();
    });

    els.btnNewSig.addEventListener('click', function () {
        var m = proj.messages[selectedMsgIndex]; if (!m) return;
        var s = new Signal(); s.name = 'Sig' + (m.signals.length + 1);
        m.signals.push(s); renderSigGrid(m); refreshPreview();
    });

    // layout 按钮
    if (els.btnLayout) {
        els.btnLayout.addEventListener('click', function () {
            var m = proj.messages[selectedMsgIndex];
            if (!m) return;
            openLayoutModal(m);
        });
    }

    ['mCanId', 'mName', 'mDlc', 'mTx', 'mExt', 'mComment',
'inpVersion'].forEach(function (id) {
        var el = els[id];
        el.addEventListener('input', function () {
            if (id === 'inpVersion') { proj.version = els.inpVersion.value;
refreshPreview(); return; }
            var m = proj.messages[selectedMsgIndex]; if (!m) return;
            if (id === 'mCanId') {
                var v = els.mCanId.value.trim();
                m.canId = (v.indexOf('0x') === 0 || v.indexOf('0X') === 0) ?
parseInt(v, 16) : parseInt(v || '0', 10);
            } else if (id === 'mDlc') { m.dlc = parseInt(els.mDlc.value || '8',
10); }
            else if (id === 'mExt') { m.ext = (els.mExt.value === '1'); }
            else if (id === 'mName') { m.name = els.mName.value; }
            else if (id === 'mTx') { m.tx = els.mTx.value; proj.addNode(m.tx);
renderNodes(); }
            else if (id === 'mComment') { m.comment = els.mComment.value; }
            renderMsgList(); refreshPreview();
        });
    });

    /* ===== 导入/导出 ===== */
    function download(filename, content, enc) {
        var data = toCRLF(content);
        var u8 = new TextEncoder().encode(data);
        var out = (enc === 'utf8bom') ? withBOM(u8) : u8;
        var blob = new Blob([out], { type: 'text/plain;charset=utf-8' });
        var a = document.createElement('a'); a.href =
URL.createObjectURL(blob); a.download = filename;
        document.body.appendChild(a); a.click(); setTimeout(function () {
URL.revokeObjectURL(a.href); a.remove(); }, 0);
    }
    els.btnExport.addEventListener('click', function () {
download('project.dbc', proj.toDBC(), els.selExportEnc.value); });
    document.addEventListener('keydown', function (e) {
        var key = (e.key || '').toLowerCase();
        if ((e.ctrlKey || e.metaKey) && key === 's') { e.preventDefault();
els.btnExport.click(); }
    });

    els.fileImport.addEventListener('change', function () {
        var f = els.fileImport.files && els.fileImport.files[0];
        if (!f) return;
        var mode = els.selImportEnc.value || 'auto';
        var readAsText = function (file) { return file.text ? file.text() : new
Promise(function (res, rej) { var r = new FileReader(); r.onload = function ()
{ res(r.result) }; r.onerror = rej; r.readAsText(file); }); };
        (async function () {
            var text = '';
            try {
                if (mode === 'utf8') { text = await readAsText(f); }
                else if (mode === 'gbk') {
                    if (window.TextDecoder) {
                        try { var buf = await f.arrayBuffer(); var dec = new
TextDecoder('gb18030'); text = dec.decode(buf); }
                        catch (e) { text = await readAsText(f); }
                    } else { text = await readAsText(f); }
                } else {
                    if (window.TextDecoder) {
                        var buf = await f.arrayBuffer(); var u8 = new
Uint8Array(buf);
                        if (u8[0] === 0xEF && u8[1] === 0xBB && u8[2] === 0xBF)
{ text = new TextDecoder('utf-8').decode(u8.subarray(3)); }
                        else {
                            try { text = new TextDecoder('utf-8', { fatal: true
}).decode(u8); }
                            catch (_) { try { text = new
TextDecoder('gb18030').decode(u8); } catch (_e) { text = new
TextDecoder().decode(u8); } }
                        }
                    } else { text = await readAsText(f); }
                }
            } catch (e) { alert('读取失败: ' + (e.message || e)); return; }
            text = stripBOM(text); parseDBCIntoProject(text, proj);
            selectedMsgIndex = proj.messages.length ? 0 : -1;
            renderNodes(); renderMsgList(); openMsg();
            renderVtList();              // 导入后刷新“值表”侧栏
            refreshAllSignalVtSelects(); // 同步中间下拉
            refreshPreview();
            els.fileImport.value = '';
        })();
    });

    /* ===== 自测（原断言 + 额外两条） ===== */
    (function selfTests() {
        function assert(cond, msg) { if (!cond) throw new Error('Test failed: '
+ msg); }
        var r = computePhysRange(8, false, 1, 0); assert(r[0] === 0 && r[1] ===
255, 'u8 raw range');
        r = computePhysRange(16, true, 0.1, -10); assert(r[0] === -10 + (-32768
* 0.1) && r[1] === -10 + (32767 * 0.1), 's16 scaled');

        var p = new Project(); p.version = '测试V1';
        var m = new Message(); m.canId = 256; m.name = 'Msg'; m.dlc = 8; m.tx =
'ECU'; m.comment = '消息备注';
        var s = new Signal(); s.name = 'Speed'; s.start = 0; s.len = 16; s.le =
true; s.signed = false; s.factor = 0.1; s.offset = 0; s.min = 0; s.max = 250;
s.unit = 'km/h'; s.receivers = ['NodeA']; s.comment = '速度';
        m.signals.push(s); p.addMessage(m);
        p.valueTables['OnOff'] = [[0, 'Off'], [1, 'On']]; s.valTable = 'OnOff';
        var out = p.toDBC();
        assert(/BO_\s+256\s+Msg: 8 ECU/.test(out), 'BO_ line');
        assert(/SG_\s+Speed\s*:\s*0\|16@1\+/.test(out), 'SG_ line');
        assert(/CM_\s+BO_\s+256\s+"消息备注";/.test(out), 'CM_ BO_');
        assert(/CM_\s+SG_\s+256\s+Speed\s+"速度";/.test(out), 'CM_ SG_');
        assert(/VAL_TABLE_\s+OnOff\s+0\s+"Off"\s+1\s+"On"\s+;/.test(out),
'VAL_TABLE_');
        assert(/VAL_\s+256\s+Speed\s+0\s+"Off"\s+1\s+"On"\s+;/.test(out),
'VAL_');

        var p2 = new Project(); parseDBCIntoProject(out, p2);
        assert(p2.messages.length === 1 && p2.messages[0].signals.length === 1,
'import BO/SG');
        assert(p2.messages[0].comment === '消息备注' &&
p2.messages[0].signals[0].comment === '速度', 'import CM_');
```

```
        var keys = []; for (var kk in p2.valueTables) { if (p2.valueTables.hasOwnProperty(kk)) keys.push(kk); }
        assert(keys.length >= 1, 'import value tables');

        r = computePhysRange(8, false, 0.5, 1); assert(r[0] === 1 && r[1] === 1 + 255 * 0.5, 'u8 half scale with offset');
        r = computePhysRange(32, true, 1, 0); assert(isFinite(r[0]) && isFinite(r[1]), 's32 finite');

        if (window.console && console.log) console.log('%cSelf-tests passed', 'color:green;font-weight:bold');
    })();

    /* ====== Layout 可视化 ====== */
    function colorForIndex(i) {
        var h = (i * 57) % 360, s = 65, l = 62; s /= 100; l /= 100;
        var c = (1 - Math.abs(2 * l - 1)) * s, x = c * (1 - Math.abs(((h / 60) % 2) - 1)), m = l - c / 2, r = 0, g = 0, b = 0;
        if (h < 60) { r = c; g = x; } else if (h < 120) { r = x; g = c; }
        else if (h < 180) { g = c; b = x; } else if (h < 240) { g = x; b = c; }
        else if (h < 300) { r = x; b = c; } else { r = c; b = x; }
        r = Math.round((r + m) * 255); g = Math.round((g + m) * 255); b = Math.round((b + m) * 255);
        return 'rgb(' + r + ',' + g + ',' + b + ')';
    }
    function expandBitsLE(start, len) {
        var out = [], idx = start;
        for (var k = 0; k < len; k++, idx++) {
            var byte = Math.floor(idx / 8), bit = idx % 8;
            out.push({ byte: byte, bit: bit });
        }
        return out;
    }
    function expandBitsBE(start, len) {
        var out = [];
        var byte = Math.floor(start / 8), bit = start % 8;
        for (var k = 0; k < len; k++) {
            out.push({ byte: byte, bit: bit });
            bit--; if (bit < 0) { bit = 7; byte++; }
        }
        return out;
    }
    function groupByRow(bitList) {
        var map = {}, i;
        for (i = 0; i < bitList.length; i++) {
            var b = bitList[i], col = 7 - b.bit;
            if (!map[b.byte]) map[b.byte] = [];
            map[b.byte].push(col);
        }
        var rows = [], byteStr;
        Object.keys(map).sort(function (a, b) { return a - b; }).forEach(function (byteStr) {
            var byte = parseInt(byteStr, 10);
            var cols = map[byte].sort(function (a, b) { return a - b; });
            var segs = [], curStart = cols[0], prev = cols[0];
            for (i = 1; i < cols.length; i++) {
                if (cols[i] === prev + 1) { prev = cols[i]; }
                else { segs.push({ from: curStart, to: prev }); curStart = prev = cols[i]; }
            }
            segs.push({ from: curStart, to: prev });
            rows.push({ byte: byte, segs: segs });
        });
        return rows;
    }
    function drawLayoutForMessage(msg) {
        var dlc = Math.max(1, Math.min(64, msg.dlc || 8));
        var cellW = 90, cellH = 60, headW = 80, headH = 40, cols = 8, rows = dlc;
        var cvs = document.getElementById('layoutCanvas'); var ctx = cvs.getContext('2d');
        cvs.width = headW + cols * cellW; cvs.height = headH + rows * cellH;
        ctx.clearRect(0, 0, cvs.width, cvs.height);
        ctx.font = '12px ui-sans-serif, system-ui, -apple-system, "Segoe UI", "PingFang SC", Arial';
        ctx.textBaseline = 'middle'; ctx.textAlign = 'center';

        // 列头
        ctx.fillStyle = '#e9ecef'; ctx.fillRect(headW, 0, cols * cellW, headH);
        ctx.strokeStyle = '#adb5bd'; ctx.strokeRect(headW, 0, cols * cellW, headH);
        ctx.fillStyle = '#495057';
        for (var c = 0; c < cols; c++) {
            var x = headW + c * cellW + cellW / 2; ctx.fillText(String(7 - c), x, headH / 2);
            ctx.beginPath(); ctx.moveTo(headW + c * cellW, 0); ctx.lineTo(headW + c * cellW, headH + rows * cellH); ctx.strokeStyle = '#dee2e6'; ctx.stroke();
        }
        // 行头
        ctx.fillStyle = '#e9ecef'; ctx.fillRect(0, headH, headW, rows * cellH);
        ctx.strokeStyle = '#adb5bd'; ctx.strokeRect(0, headH, headW, rows * cellH);
        ctx.fillStyle = '#495057';
        for (var r = 0; r < rows; r++) {
            var y = headH + r * cellH + cellH / 2; ctx.fillText(String(r), headW / 2, y);
            ctx.beginPath(); ctx.moveTo(0, headH + r * cellH); ctx.lineTo(headW + cols * cellW, headH + r * cellH); ctx.strokeStyle = '#dee2e6'; ctx.stroke();
        }

        // 信号
        for (var i = 0; i < msg.signals.length; i++) {
            var s = msg.signals[i];
            var bits = s.le ? expandBitsLE(s.start | 0, s.len | 0) : expandBitsBE(s.start | 0, s.len | 0);
            var grouped = groupByRow(bits);
            var color = colorForIndex(i);

            grouped.forEach(function (rowSeg) {
                rowSeg.segs.forEach(function (seg) {
                    var x = headW + seg.from * cellW, y = headH + rowSeg.byte * cellH, w = (seg.to - seg.from + 1) * cellW, h = cellH;
                    ctx.fillStyle = color; ctx.globalAlpha = 0.75;
ctx.fillRect(x, y, w, h); ctx.globalAlpha = 1;
                    ctx.strokeStyle = '#666'; ctx.strokeRect(x + 0.5, y + 0.5,
w - 1, h - 1);
                    ctx.fillStyle = '#111'; ctx.fillText(s.name || ('Sig' + (i
+ 1)), x + w / 2, y + h / 2);
                });
            });

            // msb / lsb 标注（首/末位）
            var first = bits[0], last = bits[bits.length - 1];
            var msbCol = 7 - first.bit, lsbCol = 7 - last.bit;
            var msbX = headW + msbCol * cellW, msbY = headH + first.byte *
cellH;
            var lsbX = headW + lsbCol * cellW, lsbY = headH + last.byte *
cellH;
            ctx.fillStyle = '#d6336c';
            // msb
            ctx.beginPath(); ctx.moveTo(msbX + 6, msbY + 6); ctx.lineTo(msbX +
18, msbY + 6); ctx.lineTo(msbX + 12, msbY + 16); ctx.closePath(); ctx.fill();
            ctx.fillText('msb', msbX + 24, msbY + 14);
            // lsb
            ctx.beginPath(); ctx.moveTo(lsbX + 6, lsbY + cellH - 6);
ctx.lineTo(lsbX + 18, lsbY + cellH - 6); ctx.lineTo(lsbX + 12, lsbY + cellH -
16); ctx.closePath(); ctx.fill();
            ctx.fillText('lsb', lsbX + 24, lsbY + cellH - 14);
        }
    }
    function openLayoutModal(msg) {
        if (!msg) return;
        var meta = document.getElementById('layoutMeta');
        meta.textContent = '消息: ' + msg.name + '   CAN ID: ' + msg.canId + '
DLC: ' + msg.dlc + '   信号数: ' + msg.signals.length;
        var cvs = document.getElementById('layoutCanvas'); cvs.width = 1200;
cvs.height = 700;
        drawLayoutForMessage(msg);
        var modal = new
bootstrap.Modal(document.getElementById('layoutModal')); modal.show();
    }

    /* ===== 初始化示例 & 紧凑模式 ===== */
    (function initDemo() {
        var m = new Message(); m.canId = 291; m.name = 'EngineData'; m.dlc = 8;
m.tx = 'ECU'; m.comment = '发动机数据';
        var s1 = new Signal(); s1.name = 'RPM'; s1.start = 0; s1.len = 16;
s1.factor = 1; s1.max = 8000; s1.unit = 'rpm'; s1.receivers = ['Dash'];
s1.comment = '转速';
        var s2 = new Signal(); s2.name = 'VehicleSpeed'; s2.start = 16; s2.len
= 16; s2.factor = 0.1; s2.unit = 'km/h'; s2.receivers = ['Dash']; s2.comment =
'车速';
        m.signals.push(s1, s2); proj.addMessage(m);
        document.getElementById('inpVersion').value = proj.version;

        renderNodes(); renderMsgList(); selectedMsgIndex = 0; openMsg();
refreshPreview();
```

```
        try {
            var saved = localStorage.getItem('dbc-density') || 'normal';
            if (saved === 'compact') document.body.classList.add('compact');
            els.btnToggleDensity.textContent =
document.body.classList.contains('compact') ? '标准模式' : '紧凑模式';
            els.btnToggleDensity.addEventListener('click', function () {
                document.body.classList.toggle('compact');
                var compact = document.body.classList.contains('compact');
                els.btnToggleDensity.textContent = compact ? '标准模式' : '紧凑模
式';
                try { localStorage.setItem('dbc-density', compact ? 'compact' :
'normal'); } catch (_) { }
            });
        } catch (_) { }
    })();

})();
```

Fence 1