

CSV Viewer 2.1

PROMPT:

The CSV Table Tool Master Prompt

Task: Create a dynamic CSV-to-HTML table converter that scans a webpage for .csv links and replaces them with an interactive data grid.

Mandatory Features & Logic:

Sticky & Resizable Headers: Headers must remain fixed at the top during vertical scrolls. Include a 6px invisible resizer handle on the right edge of each th for manual column width adjustment.

Column Management Popup: Instead of a simple dropdown, provide a "Columns ⚙️" button that opens a floating modal. Inside, provide checkboxes for every column in the dataset to toggle visibility (Show/Hide).

Horizontal Scroll Integrity: The table must be wrapped in a container that allows horizontal scrolling even when columns are hidden via the manager. Use CSS classes to hide columns so the DOM structure remains intact.

Selection Modes: Include a toggle button "Select: ROWS/COLUMNS".

Rows Mode: Standard text selection.

Columns Mode: Highlight the entire vertical column on hover and allow easy vertical text selection while preventing accidental row highlighting.

Right-Justified Header Controls: The "Manage Columns", "Row Count" (10-100), and "Font Size" (A-/A+) controls must be grouped and right-aligned.

Search with Clear Tool: A search bar on the left with a functional "X" button that appears only when text is entered to clear the input instantly.

Miniaturized Jump-to-Page: > * Located in the bottom pagination bar.

Dimensions: Exact 50px width and 18px height.

Styling: Use !important on height, min-height, and line-height to bypass external CMS/Bootstrap global styles.

Pagination: Full navigation including First («), Previous (<), Page Numbers (max 5 visible), Next (>), and Last (»).

Visual Polish: Use a default 9pt font, alternating row hover effects, and a clean, grey-scale professional border aesthetic.

Dependencies: Use PapaParse for client-side CSV parsing.

```
<script src="/custom-assets/papaparse.min.js"></script>

<style>
  .csv-table-wrapper { margin: 20px 0; font-family: -apple-system, BlinkMacSystemFont,
"Segoe UI", Roboto, sans-serif; position: relative; }

  .table-container { overflow: auto; border: 1px solid #eee; max-height: 500px; width: 100%;
}

  .custom-table { width: 100%; border-collapse: collapse; text-align: left; background:
white; table-layout: fixed; }
  .custom-table tr:hover { background-color: #f9f9f9 !important; }

  /* Default Truncated Style */
  .custom-table td { padding: 2px 5px; border: 1px solid #eee; color: #444; white-space:
nowrap; overflow: hidden; text-overflow: ellipsis; }
```

```

/* Text Wrap Mode Class */
.wrap-mode td { white-space: normal !important; overflow: visible !important; word-break:
break-word; vertical-align: top; }

.custom-table th {
padding: 2px 5px; border: 1px solid #ddd; color: #333; font-weight: 600;
white-space: nowrap; position: sticky; top: 0; background: #f1f1f1; z-index: 20; user-
select: none;
}

.col-hidden { display: none !important; }

/* Column Manager Modal */
.col-modal {
display: none; position: absolute; top: 30px; right: 80px; background: white;
border: 1px solid #ccc; box-shadow: 0 4px 12px rgba(0,0,0,0.15); z-index: 100;
padding: 10px; border-radius: 4px; min-width: 150px; max-height: 300px; overflow-y:
auto;
}
.col-modal label { display: flex; align-items: center; font-size: 11px; padding: 3px 0;
cursor: pointer; color: #444; }
.col-modal input { margin-right: 8px; }

.resizer { width: 6px; position: absolute; right: 0; top: 0; bottom: 0; cursor: col-
resize; z-index: 21; }
.resizer:hover, .resizer.active { background: #007bff; opacity: 0.5; }

.search-container { position: relative; display: flex; align-items: center; }
.table-search, .row-select, .mode-btn, .wrap-btn, .font-ctrl-group, .manage-cols-btn {
height: 22px !important; box-sizing: border-box; font-size: 11px; border: 1px solid
#ddd; border-radius: 3px; display: flex; align-items: center;
}

.table-search { width: 120px; background: white; padding: 0 18px 0 2px; line-height: 1
!important; min-height: 0 !important; }
.row-select { width: 45px; cursor: pointer; background: white; padding: 0; min-height: 0
!important; }
.manage-cols-btn { padding: 0 8px; background: #fff; cursor: pointer; }

.jump-input {

```

```

        width: 50px !important; height: 18px !important; min-height: 0 !important; line-
height: 1 !important;

        font-size: 10px; border: 1px solid #ddd; border-radius: 2px; text-align: center;
margin: 0 2px; padding: 0 !important;
    }
    .jump-go {
        height: 18px !important; min-height: 0 !important; font-size: 9px; padding: 0 6px;
background: #f0f0f0; border: 1px solid #ccc;

        border-radius: 2px; cursor: pointer; line-height: 1 !important; display: flex; align-
items: center;
    }

    .mode-btn, .wrap-btn { padding: 0 8px; background: #f8f9fa; cursor: pointer; transition:
all 0.2s; white-space: nowrap; min-height: 0 !important; }
    .mode-btn.active, .wrap-btn.active { background: #007bff; color: white; border-color:
#0056b3; }

    .clear-btn { position: absolute; right: 5px; cursor: pointer; color: #bbb; font-weight:
bold; font-size: 14px; display: none; line-height: 22px; user-select: none; z-index: 5; }

    .disable-selection td { user-select: none; }
    .disable-selection td.active-col { user-select: text !important; background-color: rgba(0,
123, 255, 0.05); }

    .font-ctrl-group { background: #f8f9fa; overflow: hidden; padding: 0; min-height: 0
!important; }
    .font-btn { padding: 0 6px; border: none; cursor: pointer; font-size: 10px; background:
transparent; height: 100%; min-height: 0 !important; }
    .font-btn:first-child { border-right: 1px solid #ddd; }
</style>

<script>
function renderCsvTables() {
    const contentArea = document.querySelector('.page-content');
    if (!contentArea) return;
    const links = contentArea.querySelectorAll('a');

    links.forEach(link => {
        if (link.href.toLowerCase().endsWith('.csv') ||
link.innerText.toLowerCase().includes('.csv')) {

```

```

const wrapper = document.createElement('div');
wrapper.className = 'csv-table-wrapper';
link.parentNode.insertBefore(wrapper, link.nextSibling);

Papa.parse(link.href, {
  download: true, header: true, skipEmptyLines: true,
  complete: function(results) {
    let data = results.data;
    const allColumns = Object.keys(data[0]);
    let currentPage = 1, rowsPerPage = 20;
    let visibleCols = new Set(allColumns);
    let sortCol = null, sortAsc = true, currentFontSize = 9, isColMode =
false, isWrapMode = false;

    wrapper.innerHTML = `
      <div style="display: flex; justify-content: space-between; align-
items: center; margin-bottom: 6px; gap: 8px;">
        <div style="display: flex; gap: 4px; align-items: center;">
          <div class="search-container">
            <input type="text" class="table-search"
placeholder="Search">
            <span class="clear-btn">&times;</span>
          </div>
          <button class="mode-btn">Select: ROWS</button>
          <button class="wrap-btn">Wrap: OFF</button>
        </div>
        <div style="display: flex; gap: 4px; align-items: center;
position: relative;">
          <button class="manage-cols-btn">Columns ✖</button>
          <div class="col-modal">
            <div style="font-weight:bold; font-size:10px; margin-
bottom:5px; border-bottom:1px solid #eee;">Show/Hide</div>
            ${allColumns.map(col => `<label><input type="checkbox"
checked data-col="${col}"> ${col}</label>`).join('')}
          </div>
          <select class="row-select">
            ${[10,20,30,40,50,60,70,80,90,100].map(v => `<option
value="${v}" ${v==20?'selected':''}>${v}</option>`).join('')}
          </select>
          <div class="font-ctrl-group">

```

```

        <button class="font-btn" data-dir="down">A-
</button><button class="font-btn" data-dir="up">A+</button>
    </div>
</div>
</div>
<div class="table-container">
    <table class="custom-table" style="font-size:
${currentFontSize}pt;">
        <thead><tr class="header-row"></tr></thead>
        <tbody class="table-body"></tbody>
    </table>
</div>
<div style="margin-top: 8px; display: flex; justify-content: space-
between; align-items: center; flex-wrap: wrap; gap: 10px;">
    <div class="pagination-info" style="font-size: 11px; color:
#777;"></div>
    <div style="display: flex; align-items: center; gap: 10px;">
        <div style="display: flex; align-items: center;">
            <span style="font-size: 9px; color: #888;">Jump:</span>
            <input type="number" class="jump-input" min="1">
            <button class="jump-go">Go</button>
        </div>
        <div class="pagination-nav" style="display: flex; gap: 2px;
align-items: center;"></div>
    </div>
</div>
`;

const tableEl = wrapper.querySelector('.custom-table'), bodyEl =
wrapper.querySelector('.table-body'), headRow = wrapper.querySelector('.header-row');
const searchInput = wrapper.querySelector('.table-search'), clearBtn =
wrapper.querySelector('.clear-btn');
const rowSelect = wrapper.querySelector('.row-select'), modeBtn =
wrapper.querySelector('.mode-btn'), wrapBtn = wrapper.querySelector('.wrap-btn');
const colModal = wrapper.querySelector('.col-modal'), manageBtn =
wrapper.querySelector('.manage-cols-btn');

manageBtn.onclick = (e) => { e.stopPropagation(); colModal.style.display =
colModal.style.display === 'block' ? 'none' : 'block'; };
document.addEventListener('click', () => colModal.style.display = 'none');
```

```

colModal.querySelectorAll('input').forEach(cb => {
  cb.onChange = () => {
    if (cb.checked) visibleCols.add(cb.dataset.col);
    else visibleCols.delete(cb.dataset.col);
    renderTable();
  };
});

wrapBtn.onclick = () => {
  isWrapMode = !isWrapMode;
  wrapBtn.classList.toggle('active', isWrapMode);
  wrapBtn.innerText = isWrapMode ? "Wrap: ON" : "Wrap: OFF";
  tableEl.classList.toggle('wrap-mode', isWrapMode);
};

function renderHeaders() {
  headRow.innerHTML = '';
  allColumns.forEach((col) => {
    const th = document.createElement('th');
    if (!visibleCols.has(col)) th.classList.add('col-hidden');
    th.innerHTML = `<span>${col} <small
style="color:#bbb">${sortCol===col?(sortAsc?'↑':'↓'):'↕'}</small></span>`;
    th.style.cursor = 'pointer';
    th.onclick = (e) => {
      if (e.target.classList.contains('resizer')) return;
      if (sortCol === col) sortAsc = !sortAsc; else { sortCol = col;
sortAsc = true; }
      renderTable();
    };
    const resizer = document.createElement('div');
    resizer.className = 'resizer';
    resizer.onpointerdown = e => {
      e.preventDefault(); e.stopPropagation();
      resizer.setPointerCapture(e.pointerId);
      const startX = e.pageX, startW = th.offsetWidth;
      const onMove = me => { th.style.width = th.style.minWidth =
Math.max(startW + (me.pageX - startX), 30) + 'px'; };
      const onUp = () => {
resizer.releasePointerCapture(e.pointerId); document.removeEventListener('pointermove',

```

```

onMove); document.removeEventListener('pointerup', onUp); });

        document.addEventListener('pointermove', onMove);
document.addEventListener('pointerup', onUp);

        };
        th.appendChild(resizer); headRow.appendChild(th);
    });
}

function renderTable() {
    const term = searchInput.value.toLowerCase();
    let filtered = data.filter(row => Object.values(row).some(v =>
String(v ?? '').toLowerCase().includes(term)));
    if (sortCol) {
        filtered.sort((a, b) => {
            const vA = String(a[sortCol] ?? '').trim(), vB =
String(b[sortCol] ?? '').trim();
            return vA.localeCompare(vB, undefined, { numeric: true,
sensitivity: 'base' }) * (sortAsc ? 1 : -1);
        });
    }
    renderHeaders();
    const totalPages = Math.ceil(filtered.length / rowsPerPage) || 1;
    if (currentPage > totalPages) currentPage = totalPages;
    const start = (currentPage - 1) * rowsPerPage;

    bodyEl.innerHTML = filtered.slice(start, start + rowsPerPage).map(row
=>
        `<tr>${allColumns.map(c => `<td class="${!visibleCols.has(c) ?
'col-hidden' : ''}">${row[c] ?? ''}</td>`).join('')}</tr>`
    ).join('');

    clearBtn.style.display = searchInput.value ? 'block' : 'none';
    wrapper.querySelector('.pagination-info').innerText = `Showing
${filtered.length ? start+1 : 0}-${Math.min(start+rowsPerPage, filtered.length)} of
${filtered.length}`;

    renderNav(totalPages);
}

function renderNav(totalPages) {
    const navEl = wrapper.querySelector('.pagination-nav');

```

```

        navEl.innerHTML = `">Page ${currentPage} of ${totalPages}</span>`;

        const navItems = [{ t: '«', p: 1, d: currentPage === 1 }, { t: '<', p:
currentPage - 1, d: currentPage === 1 }];

        let startPage = Math.max(1, currentPage - 2);
        let endPage = Math.min(totalPages, startPage + 4);
        if (endPage - startPage < 4) startPage = Math.max(1, endPage - 4);
        for (let i = startPage; i <= endPage; i++) navItems.push({ t: i, p: i,
a: i === currentPage });

        navItems.push({ t: '>', p: currentPage + 1, d: currentPage ===
totalPages }, { t: '»', p: totalPages, d: currentPage === totalPages });

        navItems.forEach(item => {
            const btn = document.createElement('button');
            btn.innerText = item.t; btn.disabled = !!item.d;
            btn.style.cssText = `padding: 0 6px; border: 1px solid #ddd;
border-radius: 2px; font-size: 10px; min-width: 22px; height: 20px; cursor: ${item.d ?
'default' : 'pointer'}; background: ${item.a ? '#007bff' : (item.d ? '#f9f9f9' : '#fff')};
color: ${item.a ? '#fff' : (item.d ? '#ccc' : '#333')}; display: inline-flex; align-items:
center; justify-content: center;`;
            btn.onclick = () => { if (!item.d) { currentPage = item.p;
renderTable(); } };

            navEl.appendChild(btn);
        });
    }

    modeBtn.onclick = () => {
        isColMode = !isColMode;
        modeBtn.classList.toggle('active', isColMode);
        modeBtn.innerText = isColMode ? "Select: COLUMNS" : "Select: ROWS";
        tableEl.classList.toggle('disable-selection', isColMode);
    };

    bodyEl.addEventListener('mouseover', (e) => {
        if (!isColMode) return;
        const cell = e.target.closest('td');
        if (!cell) return;
        const idx = cell.cellIndex;
        bodyEl.querySelectorAll('td').forEach(td =>
td.classList.remove('active-col'));
        bodyEl.querySelectorAll(`tr td:nth-child(${idx + 1}):not(.col-

```

```

hidden)`).forEach(td => td.classList.add('active-col'));
        });

        rowSelect.onchange = () => { rowsPerPage = parseInt(rowSelect.value);
currentPage = 1; renderTable(); };
        searchInput.oninput = () => { currentPage = 1; renderTable(); };
        clearBtn.onclick = () => { searchInput.value = ''; currentPage = 1;
renderTable(); };
        wrapper.querySelector('.jump-go').onclick = () => {
            const v = parseInt(wrapper.querySelector('.jump-input').value);
            if (v >= 1) { currentPage = v; renderTable(); }
        };
        wrapper.querySelectorAll('.font-btn').forEach(b => b.onclick = () => {
            currentFontSize = Math.min(16, Math.max(6, currentFontSize +
(b.dataset.dir==='up'?1:-1)));
            tableEl.style.fontSize = currentFontSize + 'pt';
        });
        renderTable(); link.style.display = 'none';
    }
    });
}
});
}
window.addEventListener('DOMContentLoaded', renderCsvTables);
</script>

```

Revision #1

Created 2026-01-16 18:36:13 UTC by Admin

Updated 2026-01-16 18:37:36 UTC by Admin