SOLAR HEALTH INDEX — Embeddable Widget

Copy everything between the <div id="solar-health-widget">
tags (plus the <style> and <script> blocks) and paste into
any webpage. The widget is fully self-contained.

HOW TO EMBED IN YOUR OWN PAGE

---

1. Copy the <style> block and paste inside your page's <head>.

```
<!-- ══════════════════════════════════════════════════════════
     PASTE THIS <style> BLOCK INTO YOUR PAGE'S <head>
     ══════════════════════════════════════════════════════════
-->
  <style>
    /* ── Solar Health Index Widget Styles ── */
    #solar-health-widget {
      --shi-bg:        #f0f4f8;
      --shi-surface:   #ffffff;
      --shi-border:    rgba(15,23,60,0.09);
      --shi-shadow:    rgba(15,23,60,0.07);
      --shi-shadow2:   rgba(15,23,60,0.13);
      --shi-text:      #0f172a;
      --shi-text2:     #334155;
      --shi-dim:       #64748b;
      --shi-dimmer:    #94a3b8;
      --shi-green:     #059669;
      --shi-green-lt: #d1fae5;
      --shi-green-bd: #6ee7b7;
      --shi-gg:        rgba(5,150,105,.18);
      --shi-sg:        #16a34a;
      --shi-sg-lt:     #bbf7d0;
      --shi-sg-bd:     #4ade80;
      --shi-sgg:       rgba(22,163,74,.22);
      --shi-yellow:    #b45309;
      --shi-yellow-lt:#fef3c7;
      --shi-yellow-bd:#fcd34d;
      --shi-yg:        rgba(180,83,9,.15);
      --shi-red:       #dc2626;
      --shi-red-lt:    #fee2e2;
      --shi-red-bd:    #fca5a5;
      --shi-rg:        rgba(220,38,38,.15);
      --shi-grey:      #94a3b8;
      --shi-grey-lt:   #f1f5f9;
      --shi-fd:        'Orbitron', monospace;
      --shi-fm:        'Space Mono', monospace;
      --shi-fb:        'Inter', sans-serif;

      font-family: var(--shi-fb);
      background: var(--shi-surface);
```

```
    border: 1px solid var(--shi-border);
    border-radius: 18px;
    box-shadow: 0 4px 24px var(--shi-shadow2);
    padding: 28px 28px 22px;
    max-width: 420px;
    width: 100%;
    box-sizing: border-box;
    display: flex;
    flex-direction: column;
    align-items: center;
    position: relative;
    overflow: hidden;
  }

  /* Subtle dot grid background */
  #solar-health-widget::before {
    content: '';
    position: absolute;
    inset: 0;
    background-image: radial-gradient(circle, rgba(100,116,139,.12) 1px, transparent 1px);
    background-size: 22px 22px;
    border-radius: 18px;
    pointer-events: none;
  }

  /* Sun icon */
  .shi-sun { position: relative; width: 44px; height: 44px; flex-shrink: 0; margin-bottom: 6px; }
  .shi-sun .shi-core {
    position: absolute; inset: 28%;
    background: radial-gradient(circle, #fef08a 0%, #f59e0b 55%, #d97706 100%);
    border-radius: 50%;
    box-shadow: 0 0 12px 4px rgba(245,158,11,.5);
  }
  .shi-sun .shi-ring {
    position: absolute; inset: 0; border-radius: 50%;
    border: 1.5px solid rgba(245,158,11,.4);
    animation: shi-spin 20s linear infinite;
  }
  .shi-sun .shi-ring::before {
    content: ''; position: absolute; inset: -9px;
    border-radius: 50%; border: 1px dashed rgba(245,158,11,.2);
  }
  .shi-sun .shi-rays {
    position: absolute; inset: -4px;
    background: repeating-conic-gradient(rgba(245,158,11,.15) 0deg, transparent 3deg,
transparent 42deg, rgba(245,158,11,.15) 45deg);
    border-radius: 50%; animation: shi-spin 40s linear infinite;
  }
  @keyframes shi-spin { to { transform: rotate(360deg); } }

  /* Header text */
  .shi-heading {
    font-family: var(--shi-fd);
    font-size: clamp(.7rem, 3vw, .85rem);
    font-weight: 800;
```

```
  color: var(--shi-text);
  letter-spacing: .18em;
  text-transform: uppercase;
  text-align: center;
  margin-bottom: 4px;
}
.shi-subhead {
  font-family: var(--shi-fm);
  font-size: .52rem;
  color: var(--shi-dim);
  letter-spacing: .1em;
  text-align: center;
  margin-bottom: 18px;
  text-transform: uppercase;
}

/* Gauge canvas + overlay */
.shi-gauge-wrap {
  position: relative;
  width: 300px;
  height: 178px;
  margin: 0 auto 6px;
}
#shi-gauge-canvas { width: 300px; height: 178px; display: block; }
.shi-score-overlay {
  position: absolute;
  bottom: 8px; left: 50%; transform: translateX(-50%);
  text-align: center; pointer-events: none; min-width: 140px;
}
.shi-score-num {
  font-family: var(--shi-fd);
  font-size: 2.1rem; font-weight: 800;
  color: var(--shi-text); line-height: 1;
  letter-spacing: -.02em; transition: color .5s;
}
.shi-score-lbl {
  font-family: var(--shi-fd);
  font-size: .58rem; font-weight: 600;
  letter-spacing: .14em; text-transform: uppercase;
  margin-top: 4px; color: var(--shi-dim); transition: color .5s;
}

/* Zone labels */
.shi-zone-row {
  display: flex; justify-content: space-between;
  width: 300px; margin: 4px auto 18px;
}
.shi-zone {
  font-size: .46rem; letter-spacing: .07em;
  color: var(--shi-dimmer); text-align: center;
  flex: 1; font-family: var(--shi-fm); text-transform: uppercase;
}
.shi-gz-red    { color: var(--shi-red); }
.shi-gz-yellow { color: var(--shi-yellow); }
.shi-gz-green  { color: var(--shi-green); }
```

```
.shi-gz-sgreen { color: var(--shi-sg); }

/* Source breakdown mini bars */
.shi-breakdown {
  width: 100%; margin-bottom: 16px;
}
.shi-bk-title {
  font-family: var(--shi-fd); font-size: .5rem; letter-spacing: .14em;
  color: var(--shi-dim); text-transform: uppercase;
  margin-bottom: 10px; text-align: center;
}
.shi-bk-item {
  display: flex; align-items: center; gap: 8px;
  margin-bottom: 7px;
}
.shi-bk-label {
  font-size: .56rem; color: var(--shi-dim); letter-spacing: .03em;
  flex: 0 0 130px; font-family: var(--shi-fb); line-height: 1.2;
}
.shi-bk-bar-wrap {
  flex: 1; height: 5px; background: var(--shi-grey-lt);
  border-radius: 3px; overflow: hidden;
}
.shi-bk-bar {
  height: 100%; width: 0%; border-radius: 3px;
  background: var(--shi-grey); transition: width .9s cubic-bezier(.4,0,.2,1), background .5s;
}
.shi-bk-val {
  font-family: var(--shi-fd); font-size: .56rem; font-weight: 600;
  color: var(--shi-dim); min-width: 22px; text-align: right; transition: color .5s;
}
.shi-bk-item.shi-bk-sgreen .shi-bk-bar { background: var(--shi-sg); }
.shi-bk-item.shi-bk-green  .shi-bk-bar { background: var(--shi-green); }
.shi-bk-item.shi-bk-yellow .shi-bk-bar { background: var(--shi-yellow); }
.shi-bk-item.shi-bk-red    .shi-bk-bar { background: var(--shi-red); }
.shi-bk-item.shi-bk-sgreen .shi-bk-val { color: var(--shi-sg); }
.shi-bk-item.shi-bk-green  .shi-bk-val { color: var(--shi-green); }
.shi-bk-item.shi-bk-yellow .shi-bk-val { color: var(--shi-yellow); }
.shi-bk-item.shi-bk-red    .shi-bk-val { color: var(--shi-red); }

/* Status pill */
.shi-pill {
  display: inline-flex; align-items: center; gap: 7px;
  padding: 5px 14px; border-radius: 20px;
  border: 1.5px solid var(--shi-border);
  font-family: var(--shi-fd); font-size: .6rem; font-weight: 600;
  letter-spacing: .1em; text-transform: uppercase;
  background: var(--shi-grey-lt); color: var(--shi-dim);
  transition: all .4s; margin-bottom: 16px;
}
.shi-dot {
  width: 7px; height: 7px; border-radius: 50%;
  background: var(--shi-grey); transition: background .4s, box-shadow .4s;
}
#solar-health-widget.shi-sgreen .shi-pill { background: var(--shi-sg-lt); border-color: var(--
```

```
shi-sg-bd); color: var(--shi-sg); }
  #solar-health-widget.shi-green  .shi-pill { background: var(--shi-green-lt); border-color: var(--shi-green-bd); color: var(--shi-green); }
  #solar-health-widget.shi-yellow .shi-pill { background: var(--shi-yellow-lt); border-color: var(--shi-yellow-bd); color: var(--shi-yellow); }
  #solar-health-widget.shi-red    .shi-pill { background: var(--shi-red-lt); border-color: var(--shi-red-bd); color: var(--shi-red); }
  #solar-health-widget.shi-sgreen .shi-dot { background: var(--shi-sg); box-shadow: 0 0 0 3px var(--shi-sg-lt); animation: shi-pdot .9s ease-in-out infinite alternate; }
  #solar-health-widget.shi-green  .shi-dot { background: var(--shi-green); box-shadow: 0 0 0 3px var(--shi-green-lt); }
  #solar-health-widget.shi-yellow .shi-dot { background: var(--shi-yellow); box-shadow: 0 0 0 3px var(--shi-yellow-lt); }
  #solar-health-widget.shi-red    .shi-dot { background: var(--shi-red); box-shadow: 0 0 0 3px var(--shi-red-lt); animation: shi-pdot .5s ease-in-out infinite alternate; }
  @keyframes shi-pdot { from { transform: scale(1); opacity: .7; } to { transform: scale(1.4); opacity: 1; } }

  /* Schumann toggle */
  .shi-sr-section {
    width: 100%;
    background: var(--shi-grey-lt);
    border: 1px solid var(--shi-border);
    border-radius: 8px;
    padding: 10px 12px 8px;
    margin-bottom: 14px;
  }
  .shi-sr-lbl {
    font-size: .49rem; color: var(--shi-dim);
    letter-spacing: .1em; text-transform: uppercase;
    font-family: var(--shi-fm); margin-bottom: 6px; text-align: center;
  }
  .shi-sr-btn-row {
    display: grid; grid-template-columns: repeat(4, 1fr); gap: 5px;
  }
  .shi-sr-btn {
    font-family: var(--shi-fd); font-size: .5rem; padding: 6px 3px;
    border-radius: 5px; border: 1.5px solid var(--shi-border);
    background: var(--shi-surface); color: var(--shi-dim);
    cursor: pointer; letter-spacing: .05em; transition: all .2s; font-weight: 600;
  }
  .shi-sr-btn:hover { border-color: var(--shi-text2); color: var(--shi-text); }
  .shi-sr-btn[data-s="sgreen"].shi-sr-active { background: var(--shi-sg); border-color: var(--shi-sg); color: #fff; box-shadow: 0 2px 8px var(--shi-sgg); }
  .shi-sr-btn[data-s="green"].shi-sr-active  { background: var(--shi-green); border-color: var(--shi-green); color: #fff; box-shadow: 0 2px 8px var(--shi-gg); }
  .shi-sr-btn[data-s="yellow"].shi-sr-active { background: var(--shi-yellow); border-color: var(--shi-yellow); color: #fff; box-shadow: 0 2px 8px var(--shi-yg); }
  .shi-sr-btn[data-s="red"].shi-sr-active    { background: var(--shi-red); border-color: var(--shi-red); color: #fff; box-shadow: 0 2px 8px var(--shi-rg); }

  /* Meta row */
  .shi-meta {
    display: flex; justify-content: space-between; align-items: center;
    width: 100%; margin-bottom: 12px; flex-wrap: wrap; gap: 6px;
```

```
    }
    .shi-update {
      font-size: .5rem; color: var(--shi-dim);
      letter-spacing: .05em; font-family: var(--shi-fm);
    }
    .shi-refresh-btn {
      font-family: var(--shi-fm); font-size: .52rem;
      background: transparent; border: 1px solid var(--shi-border);
      color: var(--shi-dim); padding: 4px 10px; cursor: pointer;
      border-radius: 4px; letter-spacing: .07em; transition: all .2s;
    }
    .shi-refresh-btn:hover {
      background: var(--shi-green); border-color: var(--shi-green);
      color: #fff; box-shadow: 0 2px 8px var(--shi-gg);
    }

    /* Book link */
    .shi-book-link {
      font-family: var(--shi-fb); font-size: .62rem;
      color: var(--shi-dim); text-align: center;
      line-height: 1.5; border-top: 1px solid var(--shi-border);
      padding-top: 12px; width: 100%;
    }
    .shi-book-link a {
      color: #2563eb; text-decoration: none;
      font-style: italic; transition: color .2s;
    }
    .shi-book-link a:hover { color: #1d4ed8; text-decoration: underline; }
    .shi-book-icon { font-size: .8rem; margin-right: 4px; }

    /* Legend */
    .shi-legend {
      display: flex; gap: 12px; flex-wrap: wrap;
      justify-content: center; width: 100%;
      margin-bottom: 14px;
    }
    .shi-li { display: flex; align-items: center; gap: 5px; font-size: .5rem; color: var(--shi-dim);
font-family: var(--shi-fm); }
    .shi-ld { width: 8px; height: 8px; border-radius: 50%; }
    .shi-ld.g  { background: var(--shi-green); }
    .shi-ld.sg { background: var(--shi-sg); }
    .shi-ld.y  { background: var(--shi-yellow); }
    .shi-ld.r  { background: var(--shi-red); }

    @media (max-width: 440px) {
      #solar-health-widget { padding: 20px 14px 16px; }
      .shi-gauge-wrap, #shi-gauge-canvas { width: 260px; height: 154px; }
      .shi-zone-row { width: 260px; }
    }
  </style>
  <!-- ═══════ END OF <style> BLOCK ═══════ -->
```

2. Copy the … block and
   paste wherever you want the widget to appear in your <body>.

```
<!--
═══════════════════════════════════════════════════════════════
    PASTE THIS <div> BLOCK WHEREVER YOU WANT THE WIDGET TO APPEAR
═══════════════════════════════════════════════════════════════ -->
<div id="solar-health-widget">

  <!-- Sun icon -->
  <div class="shi-sun">
    <div class="shi-rays"></div>
    <div class="shi-ring"></div>
    <div class="shi-core"></div>
  </div>

  <!-- Title -->
  <div class="shi-heading">Solar Health Index</div>
  <div class="shi-subhead">Live Space Weather · Auto-Refresh Every 5 Min</div>

  <!-- Status pill -->
  <div class="shi-pill"><span class="shi-dot"></span><span id="shi-pill-
txt">CALCULATING…</span></div>

  <!-- Speedometer gauge -->
  <div class="shi-gauge-wrap">
    <canvas id="shi-gauge-canvas"></canvas>
    <div class="shi-score-overlay">
      <div class="shi-score-num" id="shi-num">—</div>
      <div class="shi-score-lbl" id="shi-lbl">LOADING DATA…</div>
    </div>
  </div>

  <!-- Zone labels -->
  <div class="shi-zone-row">
    <div class="shi-zone shi-gz-red">UNFAV.</div>
    <div class="shi-zone shi-gz-yellow">CAUTION</div>
    <div class="shi-zone shi-gz-green">FAVORABLE</div>
    <div class="shi-zone shi-gz-sgreen">STRONG ★</div>
  </div>

  <!-- Source breakdown -->
  <div class="shi-breakdown">
    <div class="shi-bk-title">Source Breakdown</div>
    <div class="shi-bk-item" id="shi-bk-eflux">
      <span class="shi-bk-label">Electron Flux (GOES 7-Day)</span>
      <div class="shi-bk-bar-wrap"><div class="shi-bk-bar" id="shi-bb-eflux"></div></div>
      <span class="shi-bk-val" id="shi-bv-eflux">—</span>
    </div>
    <div class="shi-bk-item" id="shi-bk-kp">
      <span class="shi-bk-label">K-Index (24-hr Σkp)</span>
      <div class="shi-bk-bar-wrap"><div class="shi-bk-bar" id="shi-bb-kp"></div></div>
```

```
      <span class="shi-bk-val" id="shi-bv-kp">—</span>
    </div>
    <div class="shi-bk-item" id="shi-bk-sw">
      <span class="shi-bk-label">Solar Wind Speed</span>
      <div class="shi-bk-bar-wrap"><div class="shi-bk-bar" id="shi-bb-sw"></div></div>
      <span class="shi-bk-val" id="shi-bv-sw">—</span>
    </div>
    <div class="shi-bk-item" id="shi-bk-cr">
      <span class="shi-bk-label">Cosmic Rays (Mawson)</span>
      <div class="shi-bk-bar-wrap"><div class="shi-bk-bar" id="shi-bb-cr"></div></div>
      <span class="shi-bk-val" id="shi-bv-cr">—</span>
    </div>
    <div class="shi-bk-item" id="shi-bk-sr">
      <span class="shi-bk-label">Schumann Resonance</span>
      <div class="shi-bk-bar-wrap"><div class="shi-bk-bar" id="shi-bb-sr"></div></div>
      <span class="shi-bk-val" id="shi-bv-sr">—</span>
    </div>
  </div>

  <!-- Schumann manual rating -->
  <div class="shi-sr-section">
    <div class="shi-sr-lbl">Rate Schumann Resonance from live charts:</div>
    <div class="shi-sr-btn-row">
      <button class="shi-sr-btn" data-s="sgreen" onclick="shiSetSR('sgreen')">★
QUIET</button>
      <button class="shi-sr-btn" data-s="green"
onclick="shiSetSR('green')">NORMAL</button>
      <button class="shi-sr-btn" data-s="yellow"
onclick="shiSetSR('yellow')">ELEVATED</button>
      <button class="shi-sr-btn" data-s="red"    onclick="shiSetSR('red')">HIGH</button>
    </div>
  </div>

  <!-- Legend -->
  <div class="shi-legend">
    <div class="shi-li"><span class="shi-ld sg"></span>Strong Favorable</div>
    <div class="shi-li"><span class="shi-ld g"></span>Favorable</div>
    <div class="shi-li"><span class="shi-ld y"></span>Caution</div>
    <div class="shi-li"><span class="shi-ld r"></span>Unfavorable</div>
  </div>

  <!-- Meta / refresh -->
  <div class="shi-meta">
    <span class="shi-update" id="shi-update">AWAITING DATA…</span>
    <button class="shi-refresh-btn" id="shi-btn-refresh">↻ REFRESH</button>
  </div>

  <!-- Book link -->
  <div class="shi-book-link">
    <a href="https://scott-rauvers.com/solar_health.html" target="_blank" rel="noopener">
      <img src="http://ez3dbiz.com/images/solar_flares_revised_book_completed.png"
alt="Solar Flares Book Cover" style="max-width:100%;height:auto;display:block;margin:0
auto 8px;border-radius:6px;"/>
    </a>
    <span class="shi-book-icon">□</span>
```

```
    <a href="https://scott-rauvers.com/solar_health.html" target="_blank" rel="noopener">
      Long Term Health Effects of Solar Flares, Sunspots, CME's and the Schumann
Resonance
    </a>
    <div style="margin-top:8px;">
      <a href="https://www.scott-rauvers.com/solar_monitor.html" target="_blank"
rel="noopener">Insert the solar monitor onto your own webpage.</a>
    </div>
  </div>

</div>
<!-- ═══════ END OF WIDGET <div> BLOCK ═══════ -->
```

3. Copy the <script> block and paste just before </body>.

```
<!--
═══════════════════════════════════════════════════════════════════════════
═══════
    PASTE THIS <script> BLOCK JUST BEFORE YOUR CLOSING </body> TAG

═══════════════════════════════════════════════════════════════════════════
═══════ -->
<script>
(function () {
  'use strict';

  /* ── Scores & helpers
──────────────────────────────────────────────────────────── */
  const shiScores = { eflux: null, kp: null, sw: null, cr: null, sr: null };
  const shiStateScore = { sgreen: 100, green: 67, yellow: 33, red: 0 };
  const shiStateLabel = { sgreen: 'STRONG FAVORABLE', green: 'FAVORABLE', yellow:
'CAUTION', red: 'UNFAVORABLE', grey: 'NO DATA' };

  function shiWidget() { return document.getElementById('solar-health-widget'); }

  /* ── Fetch with 4-layer proxy fallback ──────────────────────────────── */
  async function shiFetchText(url) {
    const proxies = [
      { label: 'direct',    go: () => fetch(url, { signal: AbortSignal.timeout(8000) }).then(r => { if
(!r.ok) throw new Error(r.status); return r.text(); }) },
      { label: 'aorigins',  go: () =>
fetch(`https://api.allorigins.win/raw?url=${encodeURIComponent(url)}`, { signal:
AbortSignal.timeout(10000) }).then(r => { if (!r.ok) throw new Error(r.status); return r.text(); })
},
      { label: 'aorigins2', go: () =>
fetch(`https://api.allorigins.win/get?url=${encodeURIComponent(url)}`, { signal:
AbortSignal.timeout(10000) }).then(r => r.json()).then(j => { if (!j?.contents) throw new
Error('empty'); return j.contents; }) },
      { label: 'corsproxy', go: () =>
fetch(`https://corsproxy.io/?url=${encodeURIComponent(url)}`, { signal:
AbortSignal.timeout(10000) }).then(r => { if (!r.ok) throw new Error(r.status); return r.text(); })
},
    ];
```

```
    for (const { label, go } of proxies) {
      try {
        const text = await go();
        if (!text || text.length < 30) throw new Error('too short');
        if (/^\{"error"/.test(text.trim())) throw new Error('proxy error');
        return { text, via: label };
      } catch (e) { /* try next */ }
    }
    throw new Error(`All fetches failed: ${url}`);
  }
  async function shiFetchJSON(url) { const { text, via } = await shiFetchText(url); return {
data: JSON.parse(text), via }; }

  /* ── Gauge drawing
──────────────────────────────────────────────────────────────── */
  const shiGC = document.getElementById('shi-gauge-canvas');
  const shiGX = shiGC.getContext('2d');
  let shiGaugeScore = 0, shiGaugeTarget = 0, shiGaugeAnim = null;

  function shiInitGauge() {
    const dpr = devicePixelRatio || 1;
    const W = shiGC.offsetWidth || 300, H = shiGC.offsetHeight || 178;
    shiGC.width = W * dpr; shiGC.height = H * dpr;
    shiGX.scale(dpr, dpr);
  }
  shiInitGauge();

  const GCX = 150, GCY = 148, GR = 118, GRI = 74, GGAP = 0.03;
  const GSECTORS = [
    { a1: Math.PI + GGAP,       a2: Math.PI * 5 / 4 - GGAP, fill: 'rgba(220,38,38,.18)',    stroke:
'#dc2626', shadow: 'rgba(220,38,38,.3)' },
    { a1: Math.PI * 5 / 4 + GGAP, a2: Math.PI * 3 / 2 - GGAP, fill: 'rgba(217,119,6,.15)', stroke:
'#d97706', shadow: 'rgba(217,119,6,.25)' },
    { a1: Math.PI * 3 / 2 + GGAP, a2: Math.PI * 7 / 4 - GGAP, fill: 'rgba(5,150,105,.15)', stroke:
'#059669', shadow: 'rgba(5,150,105,.25)' },
    { a1: Math.PI * 7 / 4 + GGAP, a2: Math.PI * 2 - GGAP,     fill: 'rgba(22,163,74,.18)', stroke:
'#16a34a', shadow: 'rgba(22,163,74,.3)' },
  ];

  function shiDrawGauge(score) {
    const W = 300, H = 178;
    shiGX.clearRect(0, 0, W, H);
    // Background arc
    shiGX.beginPath(); shiGX.arc(GCX, GCY, GR + 5, Math.PI, Math.PI * 2, false);
shiGX.arc(GCX, GCY, GRI - 5, Math.PI * 2, Math.PI, true); shiGX.closePath();
    shiGX.fillStyle = '#f1f5f9'; shiGX.fill(); shiGX.strokeStyle = '#e2e8f0'; shiGX.lineWidth = 1;
shiGX.stroke();
    // Sectors fill
    GSECTORS.forEach(s => { shiGX.beginPath(); shiGX.arc(GCX, GCY, GR, s.a1, s.a2,
false); shiGX.arc(GCX, GCY, GRI, s.a2, s.a1, true); shiGX.closePath(); shiGX.fillStyle = s.fill;
shiGX.fill(); });
    // Sector borders
    GSECTORS.forEach(s => {
      shiGX.beginPath(); shiGX.arc(GCX, GCY, GR, s.a1, s.a2, false); shiGX.strokeStyle =
s.stroke; shiGX.lineWidth = 3; shiGX.shadowColor = s.shadow; shiGX.shadowBlur = 6;
```

```
shiGX.stroke(); shiGX.shadowBlur = 0;
      shiGX.beginPath(); shiGX.arc(GCX, GCY, GRI, s.a1, s.a2, false); shiGX.strokeStyle =
s.stroke + '88'; shiGX.lineWidth = 1.5; shiGX.stroke();
    });
    // Tick marks & labels
    [0, 25, 50, 75, 100].forEach(pct => {
      const ang = Math.PI + (pct / 100) * Math.PI, isMaj = pct % 50 === 0;
      const ro = GR + (isMaj ? 14 : 9), ri2 = GR + 2;
      shiGX.beginPath(); shiGX.moveTo(GCX + ro * Math.cos(ang), GCY + ro *
Math.sin(ang)); shiGX.lineTo(GCX + ri2 * Math.cos(ang), GCY + ri2 * Math.sin(ang));
      shiGX.strokeStyle = isMaj ? '#334155' : '#94a3b8'; shiGX.lineWidth = isMaj ? 2 : 1;
shiGX.stroke();
      if (isMaj) { shiGX.font = "600 9px 'Orbitron',monospace"; shiGX.fillStyle = '#64748b';
shiGX.textAlign = 'center'; shiGX.textBaseline = 'middle'; shiGX.fillText(pct, GCX + (ro + 11)
* Math.cos(ang), GCY + (ro + 11) * Math.sin(ang)); }
    });
    // Needle
    let nCol, nShadow;
    if (score < 25) { nCol = '#dc2626'; nShadow = 'rgba(220,38,38,.35)'; }
    else if (score < 50) { nCol = '#d97706'; nShadow = 'rgba(217,119,6,.3)'; }
    else if (score < 75) { nCol = '#059669'; nShadow = 'rgba(5,150,105,.3)'; }
    else { nCol = '#16a34a'; nShadow = 'rgba(22,163,74,.35)'; }
    const ang = Math.PI + (score / 100) * Math.PI, nLen = GR - 10;
    const nx = GCX + nLen * Math.cos(ang), ny = GCY + nLen * Math.sin(ang);
    const perp = ang + Math.PI / 2, nbh = 5;
    shiGX.beginPath(); shiGX.moveTo(nx, ny); shiGX.lineTo(GCX + nbh * Math.cos(perp),
GCY + nbh * Math.sin(perp)); shiGX.lineTo(GCX - nbh * Math.cos(perp), GCY - nbh *
Math.sin(perp)); shiGX.closePath();
    shiGX.shadowColor = nShadow; shiGX.shadowBlur = 10; shiGX.fillStyle = nCol;
shiGX.fill(); shiGX.shadowBlur = 0;
    shiGX.beginPath(); shiGX.moveTo(GCX, GCY); shiGX.lineTo(nx, ny); shiGX.strokeStyle =
nCol + 'aa'; shiGX.lineWidth = 1.5; shiGX.stroke();
    shiGX.beginPath(); shiGX.moveTo(GCX, GCY); shiGX.lineTo(GCX - 15 * Math.cos(ang),
GCY - 15 * Math.sin(ang)); shiGX.strokeStyle = '#cbd5e1'; shiGX.lineWidth = 3;
shiGX.lineCap = 'round'; shiGX.stroke();
    // Center cap
    shiGX.beginPath(); shiGX.arc(GCX, GCY, 10, 0, Math.PI * 2);
    const cap = shiGX.createRadialGradient(GCX - 2, GCY - 2, 1, GCX, GCY, 10);
cap.addColorStop(0, '#ffffff'); cap.addColorStop(1, nCol);
    shiGX.fillStyle = cap; shiGX.shadowColor = nShadow; shiGX.shadowBlur = 8; shiGX.fill();
shiGX.shadowBlur = 0;
    shiGX.beginPath(); shiGX.arc(GCX, GCY, 11, 0, Math.PI * 2); shiGX.strokeStyle =
'#e2e8f0'; shiGX.lineWidth = 1.5; shiGX.stroke();
  }

  function shiAnimateGaugeTo(target) {
    shiGaugeTarget = Math.max(0, Math.min(100, target));
    if (shiGaugeAnim) cancelAnimationFrame(shiGaugeAnim);
    function step() {
      const d = shiGaugeTarget - shiGaugeScore;
      if (Math.abs(d) < 0.3) { shiGaugeScore = shiGaugeTarget;
shiDrawGauge(shiGaugeScore); return; }
      shiGaugeScore += d * 0.08; shiDrawGauge(shiGaugeScore); shiGaugeAnim =
requestAnimationFrame(step);
    }
```

```
    step();
  }

  /* ── Update gauge from scores ──────────────────────────────────────── */
  function shiUpdateGauge() {
    let wSum = 0, wTotal = 0;
    for (const k of Object.keys(shiScores)) {
      if (shiScores[k] != null) { wSum += shiScores[k]; wTotal++; }
    }
    if (!wTotal) return;
    const avg = wSum / wTotal;
    shiAnimateGaugeTo(avg);
    document.getElementById('shi-num').textContent = avg.toFixed(0);
    let lbl, col, stateClass;
    if (avg < 25) { lbl = 'UNFAVORABLE'; col = '#dc2626'; stateClass = 'shi-red'; }
    else if (avg < 50) { lbl = 'CAUTION'; col = '#d97706'; stateClass = 'shi-yellow'; }
    else if (avg < 75) { lbl = 'FAVORABLE'; col = '#059669'; stateClass = 'shi-green'; }
    else { lbl = 'STRONG FAVORABLE'; col = '#16a34a'; stateClass = 'shi-sgreen'; }
    document.getElementById('shi-lbl').textContent = lbl;
    document.getElementById('shi-lbl').style.color = col;
    document.getElementById('shi-num').style.color = col;
    document.getElementById('shi-pill-txt').textContent = lbl;
    const w = shiWidget();
    w.classList.remove('shi-sgreen', 'shi-green', 'shi-yellow', 'shi-red');
    w.classList.add(stateClass);
  }

  /* ── Update breakdown bars ─────────────────────────────────────────
*/
  function shiUpdateBreakdown(id, state, pct) {
    const item = document.getElementById(`shi-bk-${id}`); if (!item) return;
    item.className = `shi-bk-item shi-bk-${state}`;
    document.getElementById(`shi-bb-${id}`).style.width = Math.min(100, Math.max(0, pct ||
0)) + '%';
    document.getElementById(`shi-bv-${id}`).textContent = shiStateScore[state] ?? '—';
  }

  function shiSetScore(id, state, pct) {
    if (state !== 'grey') {
      shiScores[id] = shiStateScore[state] ?? null;
      shiUpdateBreakdown(id, state, pct);
      shiUpdateGauge();
    }
  }

  /* ── Schumann Resonance manual selector ──────────────────────────── */
  const SR_MAP = {
    sgreen: { pct: 90 }, green: { pct: 60 }, yellow: { pct: 35 }, red: { pct: 15 }
  };
  function shiSetSR(state) {
    if (!SR_MAP[state]) return;
    document.querySelectorAll('.shi-sr-btn').forEach(b => b.classList.toggle('shi-sr-active',
b.dataset.s === state));
    shiSetScore('sr', state, SR_MAP[state].pct);
  }
```

```
  window.shiSetSR = shiSetSR; // expose to onclick

  /*
═══════════════════════════════════════════════════════════════════════
═══════
    DATA FETCHES
═══════════════════════════════════════════════════════════════════════
═══════ */

  /* 1 · GOES Electron Flux ──────────────────────────────────────────── */
  async function shiFetchEFlux() {
    const urls = [
      'https://services.swpc.noaa.gov/json/goes/primary/integral-electrons-7-day.json',
      'https://services.swpc.noaa.gov/json/goes/secondary/integral-electrons-7-day.json'
    ];
    for (const url of urls) {
      try {
        const { data } = await shiFetchJSON(url);
        const valid = data.filter(d => d.flux != null && d.flux > 0 && d.energy === '>=2 MeV');
        if (!valid.length) continue;
        const latestFlux = valid[valid.length - 1].flux;
        const dayMap = {};
        valid.forEach(d => { const k = d.time_tag.slice(0, 10); (dayMap[k] = dayMap[k] ||
[]).push(d.flux); });
        const dayKeys = Object.keys(dayMap).sort();
        const dayStats = dayKeys.map(k => {
          const vals = dayMap[k], mean = vals.reduce((a, b) => a + b, 0) / vals.length;
          const variance = vals.reduce((s, v) => s + (v - mean) ** 2, 0) / vals.length;
          return { cv: mean > 0 ? Math.sqrt(variance) / mean : 0 };
        });
        const priorStats = dayStats.slice(0, -1), todayStat = dayStats[dayStats.length - 1];
        const priorCVs = priorStats.map(d => d.cv).sort((a, b) => a - b);
        const medianCV = priorCVs.length ? priorCVs[Math.floor(priorCVs.length / 2)] : 0;
        let stableStreak = 0;
        for (let i = priorStats.length - 1; i >= 0; i--) { if (priorStats[i].cv <= medianCV * 1.5)
stableStreak++; else break; }
        const stableMedianCV = stableStreak >= 3 ? (() => { const cvs = priorStats.slice(-
stableStreak).map(d => d.cv).sort((a, b) => a - b); return cvs[Math.floor(cvs.length / 2)]; })() :
medianCV;
        const suddenChange = stableStreak >= 3 && todayStat.cv > stableMedianCV * 2.0;
        const isAlertLevel = latestFlux >= 1000;
        const pct = Math.max(2, Math.min(98, ((Math.log10(Math.max(latestFlux, 1)) - 1) / 5) *
100));
        let state;
        if (suddenChange || isAlertLevel) state = 'red';
        else if (stableStreak >= 3) state = 'sgreen';
        else if (stableStreak >= 1) state = 'green';
        else state = 'yellow';
        shiSetScore('eflux', state, pct);
        return;
      } catch (e) { /* try next */ }
    }
    shiSetScore('eflux', 'grey', 0);
  }
```

```
  /* 2 · Planetary K-Index ──────────────────────────────────────────────── */
  async function shiFetchKp() {
    try {
      const { data } = await shiFetchJSON('https://services.swpc.noaa.gov/products/noaa-
planetary-k-index.json');
      const sorted = data.filter(d => d.Kp != null && !isNaN(d.Kp)).sort((a, b) => new
Date(a.time_tag) - new Date(b.time_tag));
      const last24 = sorted.slice(-8);
      const sum24 = last24.reduce((a, d) => a + parseFloat(d.Kp), 0), peak24 =
Math.max(...last24.map(d => parseFloat(d.Kp)));
      const pct = Math.max(3, Math.min(94, (sum24 / 72) * 100));
      const state = sum24 > 24 ? 'red' : sum24 > 16 ? 'yellow' : sum24 > 8 ? 'green' : 'sgreen';
      shiSetScore('kp', state, pct);
    } catch (e) { shiSetScore('kp', 'grey', 0); }
  }

  /* 3 · Solar Wind Speed ──────────────────────────────────────────────── */
  async function shiFetchSW() {
    try {
      const { text } = await shiFetchText('https://services.swpc.noaa.gov/text/ace-swepam.txt');
      const rows = [];
      text.split('\n').forEach(l => {
        const p = l.trim().split(/\s+/);
        if (!p.length || !/^\d{4}$/.test(p[0])) return;
        if (+p[0] < 2000 || +p[0] > 2100 || +p[6] !== 0) return;
        let speed = null;
        for (let i = 7; i < p.length; i++) { const v = parseFloat(p[i]); if (v >= 250 && v <= 2000 &&
p[i].includes('.')) { const prev = i > 0 ? parseFloat(p[i - 1]) : NaN; if (!isNaN(prev) && prev >
0.1 && prev < 300) { speed = v; break; } if (speed === null) speed = v; } }
        if (speed && speed > 0) rows.push(speed);
      });
      if (!rows.length) throw new Error('no rows');
      const latestSpeed = rows[rows.length - 1];
      const pct = Math.max(2, ((latestSpeed - 250) / 450) * 100);
      let state;
      if (latestSpeed >= 600) state = 'red';
      else if (latestSpeed >= 410) state = 'yellow';
      else if (latestSpeed < 350) state = 'sgreen';
      else state = 'green';
      shiSetScore('sw', state, pct);
    } catch (e) { shiSetScore('sw', 'grey', 0); }
  }

  /* 4 · Cosmic Rays ──────────────────────────────────────────────────────
*/
  async function shiFetchCR() {
    const now = new Date();
    const pad = n => String(n).padStart(2, '0');
    const yr = now.getUTCFullYear(), mo = pad(now.getUTCMonth() + 1), dy =
pad(now.getUTCDate());
    const start = new Date(now); start.setDate(start.getDate() - 7);
    const syr = start.getUTCFullYear(), smo = pad(start.getUTCMonth() + 1), sdy =
pad(start.getUTCDate());
    const ouluBase =
```

```
`https://cosmicrays.oulu.fi/webform/onlinequery.cgi?station=OULU&startday=${sdy}&startmonth=${smo}&startyear=${syr}&starttime=00:00&endday=${dy}&endmonth=${mo}&endyear=${yr}&endtime=23:59&resolution=1440`;
    const strategies = [
      { url:
'https://www.nmdb.eu/nest/draw_graph.php?formchk=1&stations[]=OULU&output=ascii&tabchoice=revori&dtype=corr_for_efficiency&date_choice=last&last_days=7&last_label=days_label&tresolution=1440&yunits=0', parse: parseNMDB },
      { url: ouluBase + '&outputmode=json', parse: parseOuluJ },
      { url: ouluBase + '&outputmode=csv', parse: parseOuluC },
    ];
    function parseNMDB(txt) {
      const lines = txt.split('\n').filter(l => /^\d{4}-\d{2}-\d{2}/.test(l.trim()));
      const counts = lines.map(l => { const p = l.trim().split(/[;,\s]+/); return parseFloat(p[1] || p[2]); }).filter(v => !isNaN(v) && v > 0);
      return counts;
    }
    function parseOuluJ(txt) {
      const d = JSON.parse(txt);
      if (!d?.data) return [];
      return d.data.filter(r => r[1] > 0).map(r => r[1]);
    }
    function parseOuluC(txt) {
      return txt.split('\n').filter(l => /^\d{4}/.test(l.trim())).map(l => { const p = l.split(/[;,\s]+/); return parseFloat(p[1] || p[2]); }).filter(v => !isNaN(v) && v > 0);
    }
    for (const { url, parse } of strategies) {
      try {
        const { text } = await shiFetchText(url);
        const counts = parse(text);
        if (!counts || counts.length < 3) continue;
        const latest = counts[counts.length - 1];
        const mean5 = counts.slice(-5).reduce((a, b) => a + b, 0) / Math.min(5, counts.length);
        const pctDiff = ((latest - mean5) / mean5) * 100;
        // Streaks
        let rising = 0, steady = 0;
        const STEADY_PCT = 1.5;
        for (let i = counts.length - 1; i >= 1; i--) { if (counts[i] > counts[i - 1]) rising++; else break; }
        const mean = counts.reduce((a, b) => a + b, 0) / counts.length;
        for (let i = counts.length - 1; i >= 0; i--) { if (Math.abs(counts[i] - mean) / mean * 100 <= STEADY_PCT) steady++; else break; }
        let state, pct;
        if (rising >= 3) { state = 'sgreen'; pct = 82; }
        else if (steady >= 5) { state = 'green'; pct = 70; }
        else if (rising >= 1) { state = 'yellow'; pct = 40; }
        else if (pctDiff < -3) { state = 'red'; pct = 20; }
        else { state = 'green'; pct = 48; }
        shiSetScore('cr', state, pct);
        return;
      } catch (e) { /* try next */ }
    }
    shiSetScore('cr', 'grey', 0);
  }
```

```
  /* ── Countdown / refresh ──────────────────────────────────────────────────
*/
  let shiTimer = null, shiSecs = 300;
  function shiStartTimer() {
    clearInterval(shiTimer); shiSecs = 300;
    shiTimer = setInterval(() => {
      shiSecs--;
      if (shiSecs <= 0) { clearInterval(shiTimer); shiRefresh(); }
    }, 1000);
  }

  async function shiRefresh() {
    document.getElementById('shi-update').textContent = `Updated: ${new
Date().toUTCString().split(' ').slice(1, 5).join(' ')} UTC`;
    shiStartTimer();
    await Promise.allSettled([shiFetchEFlux(), shiFetchKp(), shiFetchSW(), shiFetchCR()]);
  }

  /* ── Init
──────────────────────────────────────────────────────────────────────────── */
  shiDrawGauge(0);
  shiSetSR('green'); // default Schumann to NORMAL
  shiRefresh();

  document.getElementById('shi-btn-refresh').addEventListener('click', () => {
clearInterval(shiTimer); shiRefresh(); });
  window.addEventListener('resize', () => { shiInitGauge(); shiDrawGauge(shiGaugeScore);
});

})();
</script>
<!-- ═══════ END OF <script> BLOCK ═══════ -->
```

4. Done — no frameworks, no dependencies, no API keys needed.
   The widget fetches live space weather data automatically.

[Return to Scott-Rauvers.com]