 Countdown-Widget (ioBroker VIS + MagicMirror)

Dieses Widget zeigt automatisch die zwei nächsten anstehenden Countdowns an (z. B. Weihnachten, Silvester, Urlaub, Touren usw.).

Abgelaufene Countdowns werden automatisch ausgeblendet.

⚠ WICHTIGE HINWEISE (bitte lesen!)

JEDER Countdown muss im versteckten Subscribe-Block stehen

KEINE Einzeiler mit { … } im JavaScript

Kein Array / kein Objekt-Literal

Code-Formatierung NICHT verändern

Sonst erscheint:

SyntaxError: Unexpected token 'null'

 1. Subscribe-Block (Pflicht!)

Hier meldet VIS alle benötigten Datenpunkte an.

```
{countdown.0.countdowns.Weihnachten.fullJson}
{countdown.0.countdowns.Silvester.fullJson}
{countdown.0.countdowns.SommerurlaubItalien.fullJson}
```

 Für jeden neuen Countdown MUSS hier eine Zeile ergänzt werden!

 2. Anzeige-HTML (2 Slots)

```
<div id="cd_text_xmas" style="font-size:28px;font-weight:700;line-
height:1.15;">
  Lade...
</div>
```

```
<div style="height:18px;"></div>
```

```
<div id="cd_text_ny" style="font-size:28px;font-weight:700;line-
height:1.15;">
  Lade...
</div>
```

Die IDs bleiben absichtlich so – sie sind MagicMirror-erprobt.

 3. JavaScript – Finale Version

⚠ NICHT kürzen, NICHT umformatieren

<script> (function () {

```
/* ============================================================
   KONFIGURATION – HIER ERWEITERN
   ============================================================ */
```

```
const OID_1 = "countdown.0.countdowns.Weihnachten.fullJson";
const OID_2 = "countdown.0.countdowns.Silvester.fullJson";
const OID_3 = "countdown.0.countdowns.SommerurlaubItalien.fullJson";
```

```
const EMPTY_TEXT = "…";
```

```
/* ============================================================
   HILFSFUNKTIONEN
   ============================================================ */
```

```
function safeNum(x, fallback)
{
  const n = Number(x);
  if (Number.isFinite(n)) return n;
  return fallback;
}
```

```
function parseFullJson(val)
{
  if (val === null || val === undefined || val === "") return null;
```

```
  try
  {
    let t = val;
```

```
    if (typeof t === "string")
    {
      t = t.trim();
      if (t.startsWith('"') && t.endsWith('"')) t = JSON.parse(t);
      return JSON.parse(t);
    }
```

```
    return t;
  }
  catch (e)
  {
    return null;
  }
}
```

```
function subline(s)
{
  return "<br><span style='font-size:18px;font-weight:400;opacity:.85'>(noch
```

```
" + s + ")</span>";
}
```

```
function dayWord(n)
{
  if (n === 1) return "Tag";
  return "Tagen";
}
```

```
function hourWord(n)
{
  if (n === 1) return "Stunde";
  return "Stunden";
}
```

```
/* ============================================================
   RESTZEIT
   ============================================================ */
```

```
function remainingSeconds(val)
{
  const d = parseFullJson(val);
  if (!d) return 999999999;
```

```
  if (d.total && d.total.seconds !== undefined)
  {
    return safeNum(d.total.seconds, 999999999);
  }
```

```
  const days  = safeNum((d.total && d.total.days  !== undefined) ?
d.total.days  : d.days, 0);
  const hours = safeNum((d.total && d.total.hours !== undefined) ?
d.total.hours : d.hours, 0);
  const mins  = safeNum((d.total && d.total.minutes !== undefined) ?
d.total.minutes : d.minutes, 0);
  const secs  = safeNum((d.total && d.total.seconds !== undefined) ?
d.total.seconds : d.seconds, 0);
```

```
  return (days * 86400) + (hours * 3600) + (mins * 60) + secs;
}
```

```
function isExpired(val)
{
  return remainingSeconds(val) <= 0;
}
```

```
/* ============================================================
   RENDER-FUNKTIONEN
   ============================================================ */
```

```
function renderWeihnachten(val)
{
  const data = parseFullJson(val);
  const now  = new Date();

  if (now.getMonth() === 11 && now.getDate() >= 24 && now.getDate() <= 26)
  {
    return "□ Frohe Weihnachten □<br><span style='font-
size:26px;'>□□□</span>";
  }

  if (!data) return EMPTY_TEXT;

  const days  = safeNum(data.total.days, 0);
  const hours = safeNum(data.total.hours, 0);

  return "□ Weihnachten in " + days + " " + dayWord(days) +
         " und " + hours + " " + hourWord(hours);
}

function renderSilvester(val)
{
  const data = parseFullJson(val);
  const now  = new Date();

  if (now.getMonth() === 11 && now.getDate() === 31)
  {
    return "□ Silvester heute";
  }

  if (!data) return EMPTY_TEXT;

  const days  = safeNum(data.total.days, 0);
  const hours = safeNum(data.total.hours, 0);

  return "□ Silvester in " + days + " " + dayWord(days) +
         " und " + hours + " " + hourWord(hours);
}

function renderGeneric(val, title, emoji)
{
  const data = parseFullJson(val);
  if (!data) return EMPTY_TEXT;

  const days  = safeNum(data.total.days, 0);
  const hours = safeNum(data.total.hours, 0);

  return emoji + " " + title + " in " + days + " " + dayWord(days) +
         " und " + hours + " " + hourWord(hours);
```

```
}
```

```
/* ============================================================
   EVENTS (KONFIGURIERBAR)
   ============================================================ */
```

```
const E1_OID = OID_1;
const E2_OID = OID_2;
const E3_OID = OID_3;
```

```
function renderEvent(i, val)
{
  if (i === 1) return renderWeihnachten(val);
  if (i === 2) return renderSilvester(val);
  if (i === 3) return renderGeneric(val, "Sommerurlaub Italien", "☀□□□");
  return EMPTY_TEXT;
}
```

```
function pickTwo()
{
  let b1 = -1;
  let b2 = -1;
  let r1 = 999999999;
  let r2 = 999999999;
```

```
  let r;
```

```
  r = remainingSeconds(vis.states.attr(E1_OID + ".val"));
  if (!isExpired(vis.states.attr(E1_OID + ".val")) && r < r1)
  {
    r2 = r1; b2 = b1;
    r1 = r;  b1 = 1;
  }
```

```
  r = remainingSeconds(vis.states.attr(E2_OID + ".val"));
  if (!isExpired(vis.states.attr(E2_OID + ".val")) && r < r1)
  {
    r2 = r1; b2 = b1;
    r1 = r;  b1 = 2;
  }
```

```
  r = remainingSeconds(vis.states.attr(E3_OID + ".val"));
  if (!isExpired(vis.states.attr(E3_OID + ".val")) && r < r2)
  {
    b2 = 3;
  }
```

```
  return String(b1) + "|" + String(b2);
}
```

```
function start()
{
  const el1 = document.getElementById("cd_text_xmas");
  const el2 = document.getElementById("cd_text_ny");
```

```
  function update()
  {
    const p = pickTwo().split("|");
```

```
    const v1 = vis.states.attr((p[0] === "1" ? E1_OID : p[0] === "2" ?
E2_OID : E3_OID) + ".val");
    const v2 = vis.states.attr((p[1] === "1" ? E1_OID : p[1] === "2" ?
E2_OID : E3_OID) + ".val");
```

```
    el1.innerHTML = renderEvent(Number(p[0]), v1);
    el2.innerHTML = renderEvent(Number(p[1]), v2);
  }
```

```
  vis.states.bind(E1_OID + ".val", update);
  vis.states.bind(E2_OID + ".val", update);
  vis.states.bind(E3_OID + ".val", update);
```

```
  update();
}
```

```
if (document.readyState === "complete")
{
  start();
}
else
{
  window.addEventListener("load", start);
}
```

})(); </script>

 Neuen Countdown hinzufügen (Checkliste) Beispiel: countdown.0.countdowns.Herbsturlaub.fullJson
 Schritt 1 – Subscribe-Block {countdown.0.countdowns.Herbsturlaub.fullJson}

 Schritt 2 – OID definieren const OID_4 = „countdown.0.countdowns.Herbsturlaub.fullJson“;

 Schritt 3 – Renderfunktion ergänzen if (i === 4) return renderGeneric(val, „Herbsturlaub“, „□□“);

 Schritt 4 – remainingSeconds / pickTwo erweitern

(analog zu Event 3)