```
#!/usr/bin/env bash
set -euo pipefail

# ─── Config
─────────────────────────────────────────────────────────────────────
SCRIPT_DIR="$(cd "$(dirname "${BASH_SOURCE[0]}")" && pwd)"
HOSTS_FILE="$SCRIPT_DIR/hosts.csv"
SITES_FILE="$SCRIPT_DIR/sites.csv"
OUTPUT_FILE="/var/www/html/dns/index.html"

# ─── Trim leading/trailing whitespace (safe with apostrophes, unlike xargs)
───
trim() { sed 's/^[[:space:]]*//;s/[[:space:]]*$//' <<< "$1"; }

# ─── Colour palette (cycles if more sites than colours)
──────────────────────
palette=("#3b82f6" "#10b981" "#f59e0b" "#8b5cf6" "#ef4444" "#ec4899"
"#06b6d4" "#84cc16")

# ─── Load sites.csv
──────────────────────────────────────────────────────────────
declare -A site_map        # prefix  -> site name
declare -A site_color_map  # site name -> colour hex
declare -a site_order=()   # insertion order for summary cards
color_idx=0
header=true

while IFS=, read -r prefix site; do
    $header && { header=false; continue; }
    prefix="$(trim "$prefix")"
    site="$(trim "$site")"
    site_map["$prefix"]="$site"
    if [[ -z "${site_color_map[$site]+_}" ]]; then
        site_color_map["$site"]="${palette[$((color_idx %
${#palette[@]}))]}"
        site_order+=("$site")
        (( color_idx++ )) || true
    fi
done < "$SITES_FILE"

# ─── DNS lookup
──────────────────────────────────────────────────────────────────
resolve() {
    # getent uses the system resolver (same as applications)
    getent hosts "$1" 2>/dev/null | awk '{print $1; exit}' || true
}

get_site() {
    local ip="$1"
    local oct3 oct2 oct1
    oct3="$(echo "$ip" | cut -d. -f1-3)"
```

```
    oct2="$(echo "$ip" | cut -d. -f1-2)"
    oct1="$(echo "$ip" | cut -d. -f1)"
    # Match longest prefix first (most specific wins)
    if   [[ -n "${site_map[$oct3]+_}" ]]; then echo "${site_map[$oct3]}"
    elif [[ -n "${site_map[$oct2]+_}" ]]; then echo "${site_map[$oct2]}"
    elif [[ -n "${site_map[$oct1]+_}" ]]; then echo "${site_map[$oct1]}"
    else echo "Unknown Location"
    fi
}

# ─── Ping test (1 packet, 2s timeout) ──────────────────────────
ping_host() {
    local result time
    result=$(ping -c 1 -w 2 "$1" 2>/dev/null) || true
    time=$(echo "$result" | grep -oP 'time=\K[0-9.]+')
    if [[ -n "$time" ]]; then
        printf "ok|%.1f" "$time"
    else
        echo "fail|"
    fi
}

# ─── Process hosts.csv ─────────────────────────────────────────────────
declare -a rows=()
declare -A site_counts
total=0
error_count=0
header=true

while IFS=, read -r hostname name; do
    $header && { header=false; continue; }
    hostname="$(trim "$hostname")"
    name="$(trim "$name")"

    ip="$(resolve "$hostname")"

    if [[ -z "$ip" ]]; then
        site="DNS Lookup Failed"
        color="#6b7280"
        ip_display="unresolved"
        ping_status="fail"
        ping_avg=""
        (( error_count++ )) || true
    else
        site="$(get_site "$ip")"
        color="${site_color_map[$site]:-#6b7280}"
        ip_display="$ip"
        site_counts["$site"]=$(( ${site_counts[$site]:-0} + 1 ))
        ping_result="$(ping_host "$hostname")"
```

```
        ping_status="${ping_result%%|*}"
        ping_avg="${ping_result##*|}"
    fi

rows+=("$hostname|$name|$ip_display|$site|$color|$ping_status|$ping_avg")
    (( total++ )) || true
done < "$HOSTS_FILE"

last_updated="$(date '+%Y-%m-%d %H:%M:%S')"

# ─── Build summary cards ─────────────────────────────────────────
site_cards=""
for site in "${site_order[@]}"; do
    count="${site_counts[$site]:-0}"
    [[ "$count" -eq 0 ]] && continue
    col="${site_color_map[$site]}"
    site_cards+="
    <div class=\"card\" style=\"border-color:${col}33\">
      <div class=\"card-label\">Hosts at site</div>
      <div class=\"card-value\" style=\"color:${col}\">${count}</div>
      <div class=\"card-sub\" title=\"${site}\">${site}</div>
    </div>"
done

if [[ "$error_count" -gt 0 ]]; then
    site_cards+="
    <div class=\"card\" style=\"border-color:#ef444433\">
      <div class=\"card-label\">DNS Failures</div>
      <div class=\"card-value\" style=\"color:#ef4444\">${error_count}</div>
      <div class=\"card-sub\">could not resolve</div>
    </div>"
fi

# ─── Build table rows ────────────────────────────────────────────
table_rows=""
for row in "${rows[@]}"; do
    IFS='|' read -r hostname name ip site color ping_status ping_avg <<<
"$row"
    if [[ "$ip" == "unresolved" ]]; then
        ip_html="<span class=\"ip failed\">unresolved</span>"
    else
        ip_html="<span class=\"ip\">${ip}</span>"
    fi
    if [[ "$ping_status" == "ok" ]]; then
        ping_html="<span class=\"badge ping-ok\">${ping_avg} ms</span>"
    else
        ping_html="<span class=\"badge ping-fail\">offline</span>"
    fi
    table_rows+="
```

```
        <tr>
          <td><span class=\"hostname\">${hostname}</span></td>
          <td><span class=\"friendly\">${name}</span></td>
          <td>${ip_html}</td>
          <td><span class=\"badge\" style=\"background:${color}22;
color:${color}; border:1px solid ${color}55\">${site}</span></td>
          <td>${ping_html}</td>
        </tr>"
done

# ─── Write HTML
──────────────────────────────────────────────────────────────
mkdir -p "$(dirname "$OUTPUT_FILE")"

cat > "$OUTPUT_FILE" <<HTML
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>DNS Location Lookup</title>
<style>
  *, *::before, *::after { box-sizing: border-box; margin: 0; padding: 0; }
  body {
    font-family: system-ui, -apple-system, 'Segoe UI', sans-serif;
    background: #0f172a; color: #e2e8f0;
    min-height: 100vh; padding: 2rem 1rem;
  }
  .container { max-width: 960px; margin: 0 auto; }
  header {
    display: flex; align-items: center; justify-content: space-between;
    flex-wrap: wrap; gap: 1rem; margin-bottom: 2rem;
  }
  .header-left h1 { font-size: 1.6rem; font-weight: 700; color: #f1f5f9; }
  .header-left p  { font-size: 0.8rem; color: #64748b; margin-top: 0.2rem; }
  .summary { display: flex; flex-wrap: wrap; gap: 0.75rem; margin-bottom:
2rem; }
  .card {
    background: #1e293b; border: 1px solid #334155;
    border-radius: 0.75rem; padding: 0.9rem 1.25rem; flex: 1 1 140px;
  }
  .card-label { font-size: 0.7rem; text-transform: uppercase; letter-
spacing: 0.08em; color: #64748b; margin-bottom: 0.3rem; }
  .card-value { font-size: 1.8rem; font-weight: 700; line-height: 1; }
  .card-sub   { font-size: 0.75rem; color: #94a3b8; margin-top: 0.25rem;
white-space: nowrap; overflow: hidden; text-overflow: ellipsis; }
  .table-wrap { background: #1e293b; border: 1px solid #334155; border-
radius: 0.75rem; overflow: hidden; }
  table { width: 100%; border-collapse: collapse; }
  thead { background: #0f172a; }
  th {
```

```
    text-align: left; padding: 0.75rem 1rem; font-size: 0.7rem;
    text-transform: uppercase; letter-spacing: 0.07em; color: #64748b; font-
weight: 600;
  }
  td { padding: 0.85rem 1rem; border-top: 1px solid #1e293b; font-size:
0.9rem; vertical-align: middle; }
  tr:nth-child(even) td { background: #172033; }
  tr:hover td { background: #1d2d45; }
  .hostname { font-weight: 600; color: #f1f5f9; font-family: monospace;
font-size: 0.95rem; }
  .friendly { color: #94a3b8; }
  .ip       { font-family: monospace; color: #7dd3fc; }
  .ip.failed { color: #f87171; font-style: italic; }
  .badge { display: inline-block; padding: 0.25rem 0.7rem; border-radius:
9999px; font-size: 0.75rem; font-weight: 600; white-space: nowrap; }
  .ping-ok   { background:#10b98122; color:#10b981; border:1px solid
#10b98155; }
  .ping-fail { background:#ef444422; color:#ef4444; border:1px solid
#ef444455; }
  .hint {
    margin-top: 1.5rem; background: #1e293b; border: 1px solid #334155;
    border-radius: 0.75rem; padding: 1rem 1.25rem; font-size: 0.8rem;
    color: #64748b; display: flex; gap: 2rem; flex-wrap: wrap;
  }
  .hint strong { color: #94a3b8; display: block; margin-bottom: 0.25rem; }
  .hint code {
    background: #0f172a; border: 1px solid #334155;
    padding: 0.1rem 0.4rem; border-radius: 0.25rem; font-family: monospace;
color: #7dd3fc;
  }
  @media (max-width: 600px) { th:nth-child(3), td:nth-child(3) { display:
none; } }
</style>
</head>
<body>
<div class="container">
  <header>
    <div class="header-left">
      <h1>Location Lookup</h1>
      <p>Last updated: ${last_updated}</p>
    </div>
  </header>
  <div class="summary">
    <div class="card">
      <div class="card-label">Total Hosts</div>
      <div class="card-value" style="color:#f1f5f9">${total}</div>
      <div class="card-sub">across all sites</div>
    </div>
    ${site_cards}
  </div>
  <div class="table-wrap">
```

```
    <table>
      <thead>
        <tr>
          <th>Hostname</th>
          <th>Friendly Name</th>
          <th>IP Address</th>
          <th>Location</th>
          <th>Ping</th>
        </tr>
      </thead>
      <tbody>
        ${table_rows}
      </tbody>
    </table>
  </div>
  <div class="hint">
    <div>
      <strong>Add / remove hosts</strong>
      Edit <code>hosts.csv</code> &mdash; columns: <code>hostname</code>,
<code>name</code>
    </div>
    <div>
      <strong>Add / change sites</strong>
      Edit <code>sites.csv</code> &mdash; columns: <code>prefix</code>
(first two or three octets), <code>site</code>
    </div>
    <div>
      <strong>Auto-refresh</strong>
      Page regenerated by cron &mdash; last run shown above
    </div>
  </div>
</div>
</body>
</html>
HTML

echo "Written to ${OUTPUT_FILE} at ${last_updated}"
```

From:
https://wiki.jdlewis.net/ - **John D. Lewis**

Permanent link:
**https://wiki.jdlewis.net/doku.php?id=temp&rev=1772226413**

Last update: **2026/02/27 21:06**