SIDEBAR Operating Systems

# Windows 11 Performance Counters Cheat Sheet

## Sidebar

Operating Systems · Windows 11 Performance Counters Cheat Sheet

The counter path syntax, the counters worth watching for CPU, disk, network, processes and GPU, and the tools that read and log them (Performance Monitor, Resource Monitor, `Get-Counter`, `typeperf`). Every counter path below returned values on the PC this was written on (Windows 11 Pro, build 26300), as a normal user, and `Get-Counter`, `Export-Counter` and `typeperf -o` were run. The thresholds are common rules of thumb, not Microsoft limits, and are marked as such. Memory counters are on the Memory Troubleshooting sheet.

### COUNTER PATHS

| | |
|---|---|
| `\Object(instance)\Counter` | The syntax: `\Processor(_Total)\% Processor Time`. `(*)` means every instance |
| `\\PCNAME\Object\Counter` | Add a computer name to read another PC (needs remote access rights) |
| `_Total` | The all-instances sum for objects that have one (Processor, Process, PhysicalDisk) |
| `Get-Counter -ListSet *` | Every counter set (234 here); `-ListSet Memory \| % Counter` lists one set's counters |
| `(Get-Counter -ListSet Process).PathsWithInstances` | Every concrete path, with instances |
| `Localised names` | On a non-English Windows the object and counter names are translated |

### READING THEM

| | |
|---|---|
| `perfmon` | Performance Monitor: add counters (green +), choose graph, histogram or report view |
| `resmon` | Resource Monitor: CPU, Memory, Disk, Network tabs with per-process detail |
| `perfmon /rel` | Reliability Monitor (a different tool: stability history) |
| `Task Manager > Performance` | The summary graphs (see the Task Manager sheet) |
| `Get-Counter` | One sample now; `-Continuous` until Ctrl+C; `-SampleInterval 5 -MaxSamples 12` for one minute |
| `typeperf` | Command Prompt equivalent: `typeperf "\Memory\Available MBytes" -sc 5` |

## CPU

| | |
|---|---|
| `\Processor(_Total)\% Processor Time` | Busy share of all logical processors (the classic figure) |
| `\Processor Information(_Total)\% Processor Utility` | Busy share scaled by current frequency; Task Manager's CPU graph matches this more closely (my understanding) |
| `\System\Processor Queue Length` | Threads waiting for a processor. Sustained values above about 2 per logical processor mean CPU contention (rule of thumb) |
| `\System\Context Switches/sec` | Thread switches. Only meaningful compared with a baseline |
| `\Process(name)\% Processor Time` | One process; can exceed 100% on many cores (it sums the cores) |
| `\Processor(_Total)\% Interrupt Time` | Time spent on hardware interrupts; persistently high points at a driver or device |
| `\Processor(_Total)\% DPC Time` | Deferred procedure calls: also drivers. Same reading |

## DISK

| | |
|---|---|
| `\PhysicalDisk(_Total)\% Disk Time` | How busy; can pass 100% on fast or parallel devices, so not a measure of saturation |
| `\PhysicalDisk(*)\Avg. Disk sec/Read` | Average read latency in seconds. Sustained above roughly 0.020 (20 ms) feels slow on a spinning disk (rule of thumb); SSDs should be far lower |
| `\PhysicalDisk(*)\Avg. Disk sec/Write` | Write latency, same reading |
| `\PhysicalDisk(*)\Current Disk Queue Length` | Requests waiting right now; sustained growth means the disk cannot keep up |
| `\PhysicalDisk(*)\Disk Bytes/sec` | Throughput, read + write |
| `\LogicalDisk(C:)\% Free Space` | Space left on the volume |
| `(*) instances` | `0 C:` style names; map them to drives with `Get-Disk`, `Get-Volume` |

On this PC the disk counters are worth reading per disk: two USB drives log thousands of device errors (see the Event Viewer sheet), and their latencies are the first thing to check.

## NETWORK

| | |
|---|---|
| `\Network Interface(*)\Bytes Total/sec` | Throughput per adapter (three instances active here) |
| `\Network Interface(*)\Output Queue Length` | Packets waiting to send; sustained above 2 suggests a bottleneck (rule of thumb) |
| `\Network Interface(*)\Packets Received Errors` | Receive errors; non-zero and rising is a cable, driver or hardware lead |
| `\Network Interface(*)\Current Bandwidth` | The link speed in bits per second |
| `Instance names` | Adapter names have special characters replaced; list with `(Get-Counter -ListSet 'Network Interface').PathsWithInstances` |

## PROCESSES, HANDLES AND SYSTEM

| | |
|---|---|
| `\Process(_Total)\Handle Count` | System-wide open handles (192,314 here) |
| `\Process(_Total)\Thread Count` | System-wide threads |
| `\System\Processes` | Number of processes |
| `\System\System Up Time` | Seconds since boot (counts a fast restart's clock differently) |
| `\Process(name)\Working Set - Private` | One program's private working set ( `firefox` returned one instance here although it has many processes: the instance names are `firefox` , `firefox#1` , …) |
| `\Process(name)\Private Bytes` | One program's commit |
| `\Paging File(_Total)\% Usage` | Page file use (see Memory Troubleshooting) |

## GPU

| | |
|---|---|
| `\GPU Engine(*)\Utilization Percentage` | Per-engine, per-process use (510 instances here). Sum by process or engine to match Task Manager |
| `\GPU Process Memory(*)\Dedicated Usage` | Dedicated video memory per process (counter exists on current Windows; not checked here) |
| `\GPU Adapter Memory(*)\Dedicated Usage` | Per adapter (not checked here) |
| `Task Manager > Performance > GPU` | The easier view for most checks |

### LOGGING TO A FILE

```
Get-Counter '\Memory\Available MBytes' -SampleInterval 5 -MaxSamples 720 | Export-Counter -Path C:\log\mem.csv -FileFormat csv
```

One hour at 5-second spacing to CSV (a small version was run here)

```
typeperf "\Memory\Available MBytes" "\Processor(_Total)\% Processor Time" -si 5 -o C:\log\p.csv -y
```

Several counters to CSV; `-y` overwrites without asking (run here with one counter)

```
-f BIN / .blg
```

`Export-Counter -FileFormat blg` writes a binary log Performance Monitor can open

```
logman create counter
```

A background data collector set; creating one usually needs admin and the Performance Log Users rights (not tested here)

```
relog
```

Convert or trim a `.blg` log

For a leak, log every 30–60 seconds for many hours, and note what you were doing (the Memory Troubleshooting sheet explains why short samples mislead).

### A BASELINE BEATS A THRESHOLD

Rules of thumb like *20 ms* or *queue of 2* are starting points. A number means something only compared with the same PC when it was working: log a quiet hour now, and keep the file.

### RATES NEED TWO SAMPLES

Counters such as % Processor Time, Bytes/sec and Pages/sec are computed between samples; the first sample of a rate counter is zero or meaningless. `Get-Counter` takes a pair of samples for you.