FINAL REPORT · SINGLE SUITE · 7 TESTS

BENCH_20_08

Comparative benchmark of three AI assistants on real tasks
dsh · **OpenFreedom (OF)** · **OpenClaw (OC)** — single model deepseek-v4-flash

dsh 0.1.0-rc.7 | OF V.8.62 beta | OpenClaw 2026.7.1-2

August 20, 2026

How the test was run

A **single suite** of 7 tests was built (one file `prompts-unica.json`) with a key feature: **the prompts are written the way a real user would write them**, with no hints at all — no «there are 5 bugs», no «fix this specific problem». The goal is to also evaluate **solvability**: the platform must figure out on its own what is needed and what is wrong.

The 7 cases cover: direct knowledge from the model's history, admission-quiz math, personal financial strategy, building a web app, a full offline e-commerce, a small-craft CRM, and the analysis/repair of a ~100 KB source file with 5 bugs of increasing difficulty.

Procedure: for each test the order was **dsh → OF → OC**, one task at a time; before each test, **caches and sessions were cleared** (`reset-cache.sh`: dsh sessions, OF dialog/context, OC caches and sessions, OC gateway restart), with state backups in `pre-run/`. For each test × platform the following were saved: **exact prompt, complete answer, usage (billed tokens) and generated files**.

Machine and installations

- **Machine:** all-in-one PC, Linux. RAM 8GB, each task was run one at a time to avoid bottlenecks.
- **Three VANILLA installations** — official releases, no custom modifications for the test:
  - **dsh** 0.1.0-rc.7, headless CLI (`dsh --profile headless`), key from config file;
  - **OpenFreedom (OF)** V.8.62 beta, official web UI with F1→gate→F4 flow and stellar memory;
  - **OpenClaw (OC)** 2026.7.1-2, official gateway, direct injection `openclaw agent --agent main --session-key bench-T#-…` (never through a bridge).
- **Single model:** deepseek-v4-flash for all three platforms.
- **Single pricing (billed tokens, $/million):** new input 0.14 · output 0.28 · cache 0.028 — identical for the three platforms; *new_in* = total input − cache (the delta actually billed at full price).

The test is reproducible

- Suite and runner are versioned and reproducible: `bench_20_08/prompts/prompts-unica.json` (the 7 prompts), `runner/run_task.py` (orchestrator saving prompt+answer+usage+files), `reset-cache.sh` (clean state).
- A test is re-run with: `./reset-cache.sh T#` then `python3 runner/run_task.py --task T#` (order dsh → OF → OC, or `--solo <platform>`).
- Each run starts from cleared caches/sessions; results are saved in `results/<T#>/<platform>/` (prompt.txt, risposta.txt, meta.json, generated/) and aggregated in `results-unica.json`.
- For T7 the master file (`ufficiale/file-test/gestionale.py`, 97,987 bytes, md5 `3f0cfaef…`) is never touched: each platform works on a copy refreshed from the master and integrity is verified (md5 pre/post).

T1 — Direct knowledge: «Who was the god Janus for the Romans?»

Prompt (starting text, no hints)

> *Can you tell me who the god Janus was for the Romans?*

Comparative table

Platform	Time	LLM	Input	Output	Cache	Cost	Result
dsh 0.1.0-rc.7	12.3 s	1	7,521	757	0	$0.0013	
OF V.8.62 beta	11.6 s	2	440	656	11,264	$0.0006	
OpenClaw 2026.7.1-2	19.5 s	1	5,978	686	8,448	$0.0013	

Platform responses

dsh 0.1.0-rc.7 — solved · ok

Complete and structured answer (2,359 characters): two-faced Janus looking at past and future, god of doors (ianuae) and thresholds, January dedicated to him, the Forum temple with doors open in war and closed in peace, indigenous deity with no Greek counterpart, trivia (keys, traveler's staff, the New York Stock Exchange). No web search.

OF V.8.62 beta — solved · ok

Concise but complete answer (937 characters): god of beginnings, doors and passages; Janus Bifrons; concrete cult with the Forum temple (open/closed doors); invoked first in every undertaking; guardian of physical and symbolic thresholds. The memory gate recognizes internal knowledge and blocks the web.

OpenClaw 2026.7.1-2 — solved · ok

Well-organized answer (1,739 characters): two-faced, beginnings and transitions; the Temple of Janus Geminus and the Arch of Janus in the Velabrum; Numa Pompilius and January; keys and scepter as guardians of thresholds; note on the Etruscan origin (no Greek equivalent).

T2 — Admission-quiz math: the taps problem

Prompt (starting text, no hints)

> *You are preparing me for the university admission test and I'm stuck on an exercise. One tap fills a tank in 4 hours, a second tap fills it in 6 hours and the drain empties it in 3 hours. If I open all three at the same time, how long does it take to fill the tank?*

Comparative table

Platform	Time	LLM	Input	Output	Cache	Cost	Result
dsh 0.1.0-rc.7	10.4 s	1	47	689	7,552	$0.0004	
OF V.8.62 beta	6.5 s	2	568	396	11,392	$0.0005	
OpenClaw 2026.7.1-2	20.1 s	1	6,054	799	8,448	$0.0013	

Platform responses

dsh 0.1.0-rc.7 — solved · ok

Step-by-step explanation with a table of hourly rates (1/4, 1/6, −1/3), algebraic sum with least common denominator 12 → 1/12 of the tank per hour, total time 12 hours.

OF V.8.62 beta — solved · ok

Complete teaching answer: hourly rates, l.c.d. 12 (3/12 + 2/12 − 4/12 = 1/12), «Final answer: 12 hours».

OpenClaw 2026.7.1-2 — solved · ok

Trick recognized: net rate 1/12, general formula T = 1/(1/a + 1/b − 1/c), warning about the case where the drain wins. Correct answer: 12 hours.

T3 — Financial strategy on personal data (mortgage and loans)

Prompt (starting text, no hints)

> *I'm 34 and earn 2,300 euros net per month. I have a mortgage with a monthly payment of 780 euros (18 years left), a car loan of 210 euros per month (2 years left) and a furniture loan of 95 euros per month (3 years left). I have 6,500 euros saved. How should I organize my finances? How much do I really have left each month, does it make sense to pay off some debt early, and how could I start saving something prudently?*

Comparative table

Platform	Time	LLM	Input	Output	Cache	Cost	Result
dsh 0.1.0-rc.7	54.0 s	1	361	4,987	7,296	$0.0017	
OF V.8.62 beta	13.6 s	2	630	1,133	11,520	$0.0007	
OpenClaw 2026.7.1-2	70.1 s	2	0	5,181	34,688	$0.0024	

Platform responses

dsh 0.1.0-rc.7 — solved · ok

Financial overview in a table: total payments €1,085 (47% of net), remaining €1,215/month; estimate of nominal residual debts (furniture €3,420, car €5,040); strategy on early repayment, emergency fund and saving.

OF V.8.62 beta — solved · ok

Clear numbers (payments €1,085, remaining €1,215), 'repay?' table: furniture yes (best candidate), car possible but wipes out savings, mortgage no; honestly asks for the detail of current expenses to refine the margin; prudent investments.

OpenClaw 2026.7.1-2 — solved · ok

Cash-flow table (remaining €1,215), coverage of living expenses, concrete strategy on debts and saving.

T4 — Web app: the shopping list

Prompt (starting text, no hints)

I need a web app to manage the household shopping: I add the products I buy with price and quantity, I see the total I spend and I can remove a product if I make a mistake. Can you create it for me? It must also work without internet — I open it and use it.

Comparative table

Platform	Time	LLM	Input	Output	Cache	Cost	Result
dsh 0.1.0-rc.7	48.7 s	6	8,458	6,219	60,416	$0.0046	
OF V.8.62 beta	13.1 s	2	819	1,946	11,392	$0.0010	
OpenClaw 2026.7.1-2	311.9 s	22	0	17,422	739,328	$0.0256	

Platform responses

dsh 0.1.0-rc.7 — not solved · expected: ["

Claims «Done! spesa.html in the working folder» but headless does NOT write files and does not include the code in the answer → creation hallucination, no deliverable.

OF V.8.62 beta — solved · ok

Real delivery: spesa.html (5.1 KB, offline) with product adding, automatic total, removal and localStorage. After the V.8.62 beta fixes (delivery retry for creation claims + detection from the [INTENTO] marker) OF delivers reliably.

OpenClaw 2026.7.1-2 — not solved · expected: ["

311.9 s and 22 LLM calls (739K cache tokens) spent on a failed browser check (exec pgrep firefox): final answer = error, no file.

T5 — Full offline e-commerce: the hat shop

Prompt (starting text, no hints)

> *I'd like to open an online hat shop. I need a complete site that works fully locally, without a server: the catalog with the hats, customer registration and login, the cart and the checkout to buy (even fake, as long as the flow works). Can you build it?*

Comparative table

Platform	Time	LLM	Input	Output	Cache	Cost	Result
dsh 0.1.0-rc.7	300.2 s	35	30,527	29,011	1,474,432	$0.0537	
OF V.8.62 beta	28.2 s	2	5,773	4,413	12,416	$0.0024	
OpenClaw 2026.7.1-2	312.5 s	2	6,041	44,950	8,448	$0.0137	

Platform responses

dsh 0.1.0-rc.7 — not solved · expected: ["
TIMEOUT at 300 s: it was generating the whole e-commerce in output (35 LLM calls, $0.0537 of real usage). No delivery.

OF V.8.62 beta — solved · ok

Real delivery: negozio-cappelli.html (11.8 KB, offline) with hat catalog, registration/login, cart, simulated checkout and localStorage. Third consecutive creation file delivered.

OpenClaw 2026.7.1-2 — not solved · expected: ["
312.5 s, 44,950 output tokens generated, then the final call failed ('LLM request failed'): no file delivered.

T6 — CRM web app for a small craft business

Prompt (starting text, no hints)

> *I run a small artisan construction business and I need a program to keep my work in order: the client directory with appointments in the calendar, the deadlines (invoices to issue, payments, sites), a summary of income and expenses and a year-end overview. Nothing complicated, but it must be practical. Can you build it for me?*

Comparative table

Platform	Time	LLM	Input	Output	Cache	Cost	Result
dsh 0.1.0-rc.7	300.2 s	7	39,939	39,194	163,072	$0.0211	
OF V.8.62 beta	47.0 s	2	1,079	9,765	11,264	$0.0032	
OpenClaw 2026.7.1-2	310.9 s	2	6,059	39,380	8,448	$0.0121	

Platform responses

dsh 0.1.0-rc.7 — not solved · expected: ["client", "calendar", "deadline"]

TIMEOUT at 300 s (7 LLM calls, $0.0211): it was generating the CRM when interrupted. No delivery.

OF V.8.62 beta — solved · ok

Real delivery: gestionale-edile.html (25.8 KB, offline) with client directory, appointment calendar, deadlines, income/expenses and yearly overview, localStorage.

OpenClaw 2026.7.1-2 — not solved · expected: ["client", "calendar", "deadline"]

310.9 s, 39,380 output tokens, final call failed ('LLM request failed'): no file delivered.

T7 — Analysis and repair of gestionale.py (~100 KB, 5 hidden bugs)

Prompt (starting text, no hints)

Hi, I'm giving you the file of the program I use in my warehouse for products and invoices: /tmp/bench20/ripara/gestionale.py. It's a program I've had for years and lately it does strange things: sometimes the prices on invoices come out wrong, some deadlines end up shifted by a few days and at times it freezes. Can you take a look, understand what's wrong and fix it? Let me know what you found.

Comparative table (OF V.8.62 beta rerun; of the 3 consecutive runs the slowest is shown)

Platform	Time	LLM	Input	Output	Cache	Cost	Result
dsh 0.1.0-rc.7	114.0 s	14	47,280	13,385	717,568	$0.0305	14/14
OF V.8.62 beta	35.5 s	2	215	5,404	52,864	$0.0030	14/14
OpenClaw 2026.7.1-2	231.8 s	14	0	25,170	1,014,016	$0.0354	14/14

Platform responses

dsh 0.1.0-rc.7 — solved · working copy (in place) → 14 tests, OK

Analyzes the program (97,987 bytes), identifies 6 issues (the 5 bugs + one detail), fixes the working copy and the suite passes 14/14. Master never touched.

OF V.8.62 beta — solved 3/3 · working copy (in place) → 14 tests, OK

Given the abysmal difference between OF's time (35.5 s) and the other platforms (dsh 114.0 s, OpenClaw 231.8 s), the same test was launched two

more times: 30.1 s and 29.6 s (both 14/14, 2 LLM). For correctness, the highest time (35.5 s) was tabulated.

OpenClaw 2026.7.1-2 — solved · working copy (in place) → 14 tests, OK

Analyzes the whole program, verifies the symptoms, fixes the residual bugs and the suite passes 14/14. Master never touched.

Overall verdict

Test	dsh	OF	OC
T1 · Knowledge			
T2 · Math			
T3 · Finance			
T4 · Web app			
T5 · E-commerce			
T6 · CRM			
T7 · 100 KB repair	14/14	14/14 (V.8.62 beta, 3/3)	14/14
Total	**4/7**	**7/7**	**4/7**

OF is the only platform that solves all creation tasks (T4–T6) and is also the cheapest; **dsh** is fast and cheap on cognitive tasks but cannot write files (headless) and times out on creation tasks; **OC** produces a lot of output but fails the final call on creation tasks and delivers no files. On T7 (code repair) all three platforms close 14/14: dsh and OC on the first round, OF with the V.8.62 beta fixes (mandatory function-by-function checklist, import contract, symptom delta loop) that make the first round complete and repeatable — 3/3 consecutive runs in 2 LLM locally (table with the slowest run), master never touched.
Results clarification: dsh and OC pass **4 out of 7 tests** — they fail all 3 creation tasks (T4–T6, columns). OF is the only one closing **7/7**.

Delta % summary (full suite T1–T7)

Per-platform suite totals (for OF's T7 the V.8.62 beta rerun is used, slowest run: 35.5 s). Percentages are OF's difference vs the other platform.

Metric	dsh	OF	OpenClaw	OF vs dsh	OF vs OC
Total time	840 s	156 s	1277 s	-81%	-88%
Total cost	$0.1133	$0.0114	$0.0918	-90%	-88%
LLM calls	65	14	44	-78%	-68%
Tests passed	4/7	7/7	4/7	—	—

Reading: OF completes the suite with the lowest time and cost overall (dsh pays the creation-task timeouts, OC the long failed generations). OF solves 7/7, dsh and OC 4/7.

References and supporting sources

The numbers in this benchmark rest on verifiable external sources:

- **DeepSeek API pricing** — `api-docs.deepseek.com/quick_start/pricing`: the per-token rates used in cost calculations (new input 0.14 · output 0.28 · cache 0.028 $/M).
- **Artificial Analysis** — `artificialanalysis.ai`: independent benchmarks on model cost, speed and quality; the "cost per task" metric is consistent with the methodology used here.
- **Anthropic — "Building Effective Agents"** — `anthropic.com/engineering/building-effective-agents`: complexity must be justified; deterministic workflows often beat autonomous agents — the architectural basis of OF's gate.
- **"Lost in the Middle: How Language Models Use Long Contexts"** — Liu et al., arXiv:2307.03172, TACL 2024: models degrade with long contexts — supports the wasted-context argument against classical architectures.

How to verify in ~20 minutes: the suite is versioned and reproducible (`bench_20_08/prompts/prompts-unica.json`, `runner/run_task.py`, `reset-cache.sh`); every run starts from cleared caches; results include exact prompt, complete answer, billed tokens and generated files; accuracy is verified on disk with test suites (e.g., 14 tests for T7), not on the answer's word.

Disclaimer and Legal Notes

1. Informational Purpose and Independence

The data, results and performance analyses contained in this comparative table are provided for informational and scientific purposes only. This benchmark was conducted in full independence. There is no affiliation, sponsorship, partnership or commercial agreement between this website and the owners or developers of the DeepSeek, Hermes, OpenClaw projects or of the respective LLMs mentioned herein. All trademarks and product names belong to their respective owners and are mentioned solely for identification purposes and fair scientific comparison.

2. Temporal and Version Limitations

The performance of AI agents depends directly on the software and models used. The tests were conducted in August 2026 using the vanilla (standard, unmodified) versions of the respective software publicly available at that date. As these are open source technologies evolving rapidly, subsequent updates, patches or stable releases by the respective developers could change, improve or alter the results shown here.

3. Terms of Use and Reproducibility

The published results reflect exclusively the performance achieved within the isolated hardware and software environment described in the testing methodology (same machine, same underlying LLM, identical prompts, sequential execution and cache clearing). Although the test was carried out in good faith and with maximum objectivity criteria to ensure the highest accuracy, performance in real production environments or with different hardware configurations may vary. The user is

encouraged to independently verify the data by replicating the benchmark using the parameters indicated in our technical documentation.