Open-Source Plugin Brings Infinite Dungeon Crawls to Engineers Between Coding Sessions

claude-dungeon uses LLM-generated worlds, natural language commands, and ASCII art to fight AI fatigue—one five-minute break at a time

Press Release

SAN FRANCISCO — August 15, 2026 — Today, Punt Labs released claude-dungeon, a free, open-source Claude Code plugin that drops a roguelike text adventure directly into engineers' terminal sessions. Unlike the vocabulary-limited text adventures of the 1980s, claude-dungeon uses large language models to understand natural language, generate procedurally unique worlds, and render ASCII art on the fly—giving developers a five-minute dungeon crawl that research shows can reduce cognitive fatigue and restore focus (see FAQ 8).

Problem

Engineers using AI coding assistants spend hours in deep terminal sessions, experiencing what Steve Yegge recently called "AI vampire syndrome" [1]—massive daily fatigue from the constant cognitive load of reviewing, verifying, and directing AI-generated code. UC Berkeley researchers confirmed that AI-assisted work creates "workload creep, cognitive fatigue, burnout, and weakened decision-making" [2]. A Haystack Analytics study found that 83% of software developers suffer from workplace burnout [3], and the rise of AI-assisted coding is making it worse, not better [4].

The available coping mechanisms—switching to a browser tab, scrolling social media, or stepping away entirely—break flow state and make returning to focused work harder. Attention Restoration Theory [5] suggests that directed attention fatigue recovers not through passive rest but through engagement with activities that provide involuntary fascination—absorption without cognitive effort. Engineers need a break that recharges directed attention without the context-switching overhead of leaving the terminal (see FAQ 19).

Solution

claude-dungeon embeds a roguelike text adventure into the terminal where engineers already work. Type `/d` to take your next turn—no parser vocabulary to memorize, no syntax to learn. The LLM generates dungeons, encounters, items, and narrative on the fly, so no two sessions are alike. Players choose their adventure: a haunted library, a derelict space station, a classic Tolkien-esque cavern, or whatever they describe. The game state persists across sessions automatically; when you're ready for a fresh quest, `/dungeon:new` starts a new world.

Attention Restoration Theory predicts that activities providing "soft fascination"—gentle, absorbing engagement—restore the directed attention depleted by sustained analytical work [5]. A roguelike dungeon crawl fits this model: it is absorbing enough to displace the mental residue of code review, but low-stakes enough to release rather than consume cognitive resources. Research on gaming as recovery from work stress confirms that game play pursued with harmonious passion facilitates cognitive resource accumulation [6]. claude-dungeon is designed for exactly that cadence—type `/d`, take a turn, go back to coding. Short enough to refresh, long enough to engage, and always waiting where you left off (see FAQ 9).

Customer Quote

> *"Before claude-dungeon, my breaks meant doomscrolling Reddit or staring at the wall. I'd come back to my session feeling worse. Now I type /d, take a turn, maybe solve a puzzle or find a magic sword, and go right back to coding. Sometimes I take one turn. Sometimes I take five. Either way I come back with fresh eyes. It's absurd that killing fictional orcs is better for my productivity than anything else I've tried, but here we are."*
>
> — **Marcus Chen**, Senior Platform Engineer at Anthropic

Getting Started

Getting started takes one command. Engineers run the install script:

```
curl -sL https://raw.githubusercontent.com/punt-labs/claude-dungeon/main/install.sh |
                                                bash
```

Then type `/d` in any Claude Code session, and within thirty seconds they are standing at the entrance of a procedurally generated dungeon. Every subsequent `/d` picks up exactly where they left off. There is no account to create, no API key to configure, and no subscription to manage—the game runs entirely within the existing Claude Code session.

Spokesperson Quote

> "We built claude-dungeon because the irony was hard to ignore—the tool that makes engineers most productive is also the one that drains them most. The fix isn't another productivity feature. It's permission to play. By putting the game where engineers already work, in the terminal, with the same LLM they're already talking to, we removed every barrier between 'I need a break' and actually taking one."
>
> — **Jim Freeman**, Creator of claude-dungeon, Punt Labs

Call to Action

claude-dungeon is available now at github.com/punt-labs/claude-dungeon. The plugin is free, open source under the MIT license, and installs in under ten seconds. Star the repository to follow development and join the community of engineers who believe that play and productivity are not opposites.

Contact: hello@punt-labs.com

Frequently Asked Questions

External FAQs

Q1: What is claude-dungeon and who is it for?

claude-dungeon is a free, open-source Claude Code plugin that turns coding breaks into rogue-like text adventures. It is built for engineers who spend long sessions in Claude Code—especially those who grew up on UNIX terminals and remember Zork, Wizardry, and Rogue—and want a way to recharge without leaving the terminal. Type `/d`, take a turn in a procedurally generated dungeon, and go back to coding.

Q2: How is this different from just playing a game in another terminal tab?

Context switching is the enemy. Opening another terminal to play nethack, launching a browser game, or picking up your phone costs cognitive overhead—the same overhead that makes "taking a break" feel like it makes things worse. claude-dungeon lives inside the same Claude Code session where you are already working. Type `/d`, take one turn, continue coding. No window switching, no lost context.

The game is also LLM-powered, so it understands natural language instead of requiring memorized parser commands. You type what you want to do ("search the bookshelf for hidden passages"), not what the parser expects ("EXAMINE BOOKSHELF").

Q3: How do I get started?

One command to install, one command to play:

1. Run the install script:
   ```
   curl -sL https://raw.githubusercontent.com/punt-labs/claude-dungeon/main/install.sh | bash
   ```
2. Type `/d` in any Claude Code session.
3. You are in a dungeon. What do you do?

Every subsequent `/d` picks up where you left off. Type `/dungeon:new` when you want a fresh world.

Q4: How much does it cost?

Nothing. claude-dungeon is free and open source under the MIT license. It uses the Claude API through your existing Claude Code session—no additional API keys, subscriptions, or accounts required.

Q5: Is the content safe for work?

Yes. claude-dungeon enforces content guardrails that keep all generated content appropriate for professional settings and all ages. The LLM is instructed to generate fantasy adventure content—monsters, puzzles, treasure, traps—without graphic violence, horror, sexual content, or other mature themes. Content filtering runs on every generated response before it reaches the player.

Q6: What happens to my game data?

Game state is stored locally on your machine in the plugin's data directory. No game data is sent to external servers beyond the Claude API calls needed to generate the game (which are part of your existing Claude Code session). Save files are plain text and can be inspected or deleted at any time.

Internal FAQs

Value & Market

Q7: What is the total addressable market?

Bottoms-up estimate. Claude Code had 115,000 developers as of July 2025 [7]. The broader Claude platform has seen rapid enterprise adoption, with Business of Apps reporting significant usage growth through 2025–2026 [8].

Projection (August 2026): Assuming Claude Code user growth between 2x (low) and 5x (high) over the 13 months from July 2025 to August 2026, the user base ranges from 230,000 to 575,000. Midpoint estimate: ~350,000 Claude Code users. These growth rates are assumptions—Anthropic does not publish Claude Code user counts regularly.

At a 5–15% install rate (typical range for popular free plugins in developer tool ecosystems), that yields 17,500 to 86,000 installs. The broader market includes all AI coding assistant users (GitHub Copilot, Cursor, Windsurf), but the initial product targets Claude Code exclusively.

This is a free, open-source product. TAM is measured in users and GitHub stars, not revenue. The adjacent opportunity is establishing Punt Labs as a recognized name in the Claude Code ecosystem.

Q8: What evidence do we have that customers want this?

Hypothesis stage—no primary customer data yet. The evidence is indirect but convergent:

Evidence that the problem exists:

- Steve Yegge's "AI Vampire" essay (February 2026) resonated widely, suggesting AI fatigue is broadly felt [1].
- UC Berkeley researchers confirmed AI causes "workload creep, cognitive fatigue, burnout, and weakened decision-making" [2].
- Haystack Analytics found 83% of software developers suffer from workplace burnout [3]. A 2025 study using the validated Maslach Burnout Inventory found 22% of engineering professionals at critical burnout levels [9].
- Harvard Business Review documented that AI "intensifies" work rather than reducing it [4].

Evidence that the solution approach works:

- AI Dungeon proved demand for LLM-powered text adventures—100,000 players in its first week, 1.5 million within six months [10, 11].
- Attention Restoration Theory predicts that "soft fascination" activities—absorbing but low-effort—restore the directed attention depleted by sustained analytical work [5]. A dungeon crawl fits this model better than passive scrolling.
- Research on gaming as work recovery found that video game play facilitates cognitive resource accumulation when pursued with harmonious (non-compulsive) passion [6].
- The Bash Screensavers project proved developer appetite for terminal entertainment [12].
- The roguelike market is growing at 8.7% CAGR [13].

What is not yet validated: Whether engineers will actually play a game *during work hours* inside their coding tool. The theoretical case is strong—attention restoration through absorbing non-work activity—but this specific context (game embedded in a coding assistant) has no precedent. This is the core value risk.

Q9: Who are the competitors and why will we win?

AI Dungeon [10]: The closest precedent. LLM-powered text adventure that reached 100,000 players in its first week and 1.5 million within six months [11]. However: standalone consumer app, not embedded in a development tool. Retired from Steam in March 2024. Targeted general gamers, not developers taking work breaks.

Classic roguelikes (nethack, Angband, DCSS): Available in the terminal but require a separate window, have steep learning curves, and demand long sessions incompatible with five-minute breaks.

Browser games, Reddit, social media: The current "break" behavior for most developers. High context-switching cost. These provide distraction but not the "soft fascination" that Attention Restoration Theory identifies as restorative [5]—passive scrolling does not engage the mind deeply enough to displace the cognitive residue of the previous task. Any break is better than no break, but an absorbing activity is better than a passive one.

Bash Screensavers: Terminal entertainment that proved the appetite, but passive rather than interactive [12].

Why we win: We are not competing for "game time"—we are competing for "break time." No existing product is an interactive, LLM-powered game embedded in an AI coding assistant. The `/d` command has lower activation energy than any alternative.

If a competitor saw this press release: AI Dungeon could build a Claude Code plugin, but they exited their platform and are not in developer tools. Anthropic could build this themselves, but terminal games are orthogonal to their mission. The moat is community and brand, not technology—first to build a beloved plugin with a loyal player community wins.

Q10: What is the customer acquisition strategy?

Organic distribution through free, open-source channels:

1. **GitHub discovery**: README with compelling ASCII art screenshots and a one-line install command.
2. **Hacker News launch**: The "AI fatigue → dungeon crawling" angle is catnip for HN.
3. **Claude Code plugin marketplace**: Listed alongside productivity plugins, standing out as the fun one.
4. **Word of mouth**: Engineers showing colleagues a dungeon in their terminal. "Wait, you can do that?"

No paid acquisition. The product is free; the only cost of a new user is zero.

Q11: What is the next step to validate this vision?

Ship an MVP to 20 Claude Code users and measure three things over two weeks:

1. **Repeat usage**: Do they play more than once? Target: 40%+ return within one week.
2. **Session length**: Do they play for 5–15 minutes, or abandon after one turn? Target: median session of 3+ turns.
3. **Self-reported effect**: Do they feel refreshed? Quick post-session survey.

If repeat usage exceeds 40% and session length matches the microbreak window, the concept is validated. If not, we learn whether the problem is the game (not fun enough) or the context (people won't play at work).

Technical

Q12: What are the major technical risks?

1. **Content safety**: The LLM generates narrative content. Risk that adversarial prompts or edge-case game states produce inappropriate content. Mitigation: system prompt guardrails, output content filtering, community reporting mechanism. This is the highest-priority technical risk.
2. **Game coherence**: LLM-generated narrative may be inconsistent across sessions—forgetting items, contradicting earlier events, breaking continuity. Mitigation: structured game state (inventory, map, event history) stored locally and fed to the LLM as context on each turn.
3. **Latency**: Each `/d` turn requires an LLM API call. If response time exceeds 5–10 seconds, the experience feels sluggish for a "quick break." Mitigation: compact prompts, minimal context windows, Claude's fast output mode.
4. **Token consumption**: The game uses the player's Claude Code session tokens. Heavy play could consume meaningful token budget. Mitigation: keep prompts compact, cache world descriptions locally, make token usage transparent to the player.

Q13: What dependencies exist?

- **Claude Code plugin API**: Depends on the plugin system remaining stable. Currently in active development by Anthropic.
- **Claude API**: Core dependency for all content generation. Risk is pricing changes or rate limits.
- **LanceDB**: Local embedded vector store for game state persistence and semantic search of world history. Lightweight, no server dependency. Already present in the project.

No dependencies on other teams. Single-developer project with no external coordination required.

Q14: What is the estimated development timeline?

- **Phase 1** (2 weeks): Core game loop—`/d` command, basic dungeon generation, turn-based interaction, save/load state with LanceDB.
- **Phase 2** (2 weeks): Content safety guardrails, ASCII art rendering, adventure theme selection (`/dungeon:new`).
- **Phase 3** (1 week): Polish, documentation, install script, GitHub release.

MVP in five weeks. If timeline compresses, cut ASCII art and theme selection from Phase 2—the core game loop works without them.

Q15: If this succeeds, what breaks first?

Nothing at the infrastructure level—the plugin is stateless, runs locally, and requires no servers. At scale, the pressures are:

1. **Token consumption**: Each `/d` turn consumes tokens from the player's Claude Code session. Assuming 500 tokens per turn and 10 turns per day, that is roughly 5,000 tokens per active user per day. At 50,000 active users, aggregate consumption is meaningful—but it is distributed across individual accounts, not centralized. The risk is player perception: if users feel the game "wastes" their token budget, they will stop playing. Mitigation: transparent token usage reporting per session.
2. **LanceDB storage**: Game state grows with play. A long-running game might accumulate several megabytes of event history. At 50,000 users this is local disk usage, not a shared

resource—but semantic search over large histories could slow down if not pruned. Mitigation: cap event history at a rolling window.

3. **Maintainer capacity**: At 1,000+ GitHub stars, issue volume and feature requests will exceed solo maintainer bandwidth. Mitigation: clear contribution guidelines, aggressive scope discipline via the Won't Do list, and community co-maintainers recruited from active contributors.
4. **Model evolution**: As Claude models are updated, game prompts may need tuning to maintain narrative quality and content safety. Mitigation: version-pinned prompts with regression testing.

Q16: What does the cost structure look like at steady state?

claude-dungeon has zero infrastructure cost—no servers, no databases, no hosted APIs. The real cost is maintainer time:

- **Pre-launch**: ~5 weeks of development (the entire build).
- **At 500 stars**: 2–4 hours per week covering issue triage, prompt maintenance as Claude models evolve, and community engagement.
- **At 1,000+ users**: 4–8 hours per week. Maintainer capacity becomes the constraint.

The plugin is designed to be self-sustaining: the codebase is plugin skills, a LanceDB store, and prompt files—no backend, no API keys beyond the user's existing Claude Code session. If Punt Labs disappears tomorrow, the plugin continues working. This zero-infrastructure architecture makes it trivially forkable. The opportunity cost is hours not spent on other Punt Labs projects—acceptable while the plugin establishes ecosystem presence, re-evaluate if other priorities demand the same hours.

Business

Q17: What is the revenue model?

None. claude-dungeon is free, open source under the MIT license. Strategic value to Punt Labs:

- Establish reputation in the Claude Code plugin ecosystem.
- Generate GitHub stars and community goodwill.
- Create a platform for future Punt Labs plugins that may have commercial models.
- Demonstrate that Claude Code plugins can be experiences, not just productivity tools.

Q18: What are the key metrics for success?

Metric	Target (3 months)	Kill threshold
GitHub stars	1,000	<200
Unique installs	5,000	<500
Weekly repeat usage	40%+	<15%
Median session length	5–15 min	<1 min
Content safety incidents	0	Any unresolved

If stars are below 200 and installs below 500 after three months, the concept does not resonate and the project should be archived or pivoted.

Q19: Why now? What has changed?

Three things converged:

1. **AI fatigue is newly named.** Steve Yegge's "AI Vampire" essay (February 2026) put a label on what developers were feeling [1]. UC Berkeley research validated it [2]. The problem is legible for the first time.
2. **Claude Code's plugin ecosystem is maturing.** The platform now supports the kind of interactive, persistent, slash-command-driven plugin that claude-dungeon requires.
3. **LLMs are good enough for games.** Claude's narrative generation is qualitatively better than what was available when AI Dungeon launched with GPT-2 in 2019 [10]. Natural language understanding, coherent multi-turn narrative, and content safety capabilities have all crossed the threshold needed for a genuinely enjoyable game experience.

Risk Assessment

Risk	Rating	Assessment
Value	Medium	The problem (AI fatigue) is real and well-documented. The untested assumption is whether engineers will play a game *during work hours* inside their coding tool. AI Dungeon's 1.5M users proves demand for LLM text adventures, but that was a standalone consumer product, not an embedded work break. See FAQ 8.
Usability	Medium	Activation friction is near zero (`/d`). But activation is not the risk—retention is. Whether the LLM generates narrative that is coherent, fun, and game-like rather than chatbot-like is entirely unvalidated. AI Dungeon struggled with narrative coherence at scale. Latency above 5 seconds would undermine the "quick break" experience. See FAQ 12.
Feasibility	Medium	Core components exist: Claude API, LanceDB, plugin system. No novel technology for the game loop. However, content safety in interactive LLM fiction is not a solved problem—AI Dungeon's content moderation struggles led to provider conflicts and eventual Steam retirement [14]. A solo developer implementing robust guardrails for a tool used in professional settings carries real risk. See FAQ 12.
Viability	Low	Free, open-source product with no revenue expectations. The only cost is development time. Viability risk is limited to opportunity cost—is this the best use of Punt Labs' time for building ecosystem reputation? See FAQ 17.

Feature Appendix

Defines the scope boundary for the product. Every feature is explicitly categorized to prevent scope creep and force prioritization decisions upfront.

Must Do

Features that are essential for launch. Without these, the product does not solve the core problem.

- **F1.** **Core game loop** — the `/d` command triggers a turn-based dungeon crawl—describe an action in natural language, receive a narrated outcome
- **F2.** **Natural language input** — players type what they want to do in plain English; no parser vocabulary or command syntax to memorize
- **F3.** **Procedural world generation** — the LLM generates dungeons, encounters, items, and narrative on each turn; no two sessions are alike
- **F4.** **Persistent game state** — game saves automatically to local storage (LanceDB) and resumes on the next `/d`; `/dungeon:new` starts a fresh world
- **F5.** **Content safety guardrails** — system prompts and output filtering enforce SFW content appropriate for professional settings and all ages
- **F6.** **Curl install script** — one-command installation from GitHub; no manual configuration or dependency management

Should Do

Features that meaningfully improve the product but are not launch-blocking. Candidates for fast follow-up.

- **F7.** **ASCII art rendering** — dungeon maps, monsters, items, and scenes rendered in ASCII art within the terminal
- **F8.** **Adventure theme selection** — players describe the kind of world they want when starting a new game—haunted library, space station, classic fantasy
- **F9.** **Character progression** — levels, inventory, stats, and abilities that persist across sessions and create a sense of growth
- **F10.** **World memory via LanceDB** — semantic search of past events and locations to maintain narrative coherence across long-running games

Won't Do

Features explicitly excluded. Naming what you won't build is as important as naming what you will—it prevents scope creep and clarifies the product's identity.

- **F11.** **Multiplayer** — adds networking complexity and social dynamics without solving the core problem of solo break-time recovery; the game is a personal retreat, not a social platform
- **F12.** **Graphical rendering** — the terminal is the medium; a GUI defeats the purpose and breaks the "stay in your coding environment" value proposition
- **F13.** **Leaderboards or competitive features** — this is a personal break, not a competition; adding rankings would create pressure that undermines the relaxation goal
- **F14.** **Mobile or web client** — the product is a Claude Code plugin, not a standalone game; scope expansion to other platforms would dilute focus and compete with established games
- **F15.** **Monetization infrastructure** — free and open source is the strategic choice; adding payment systems contradicts the community-first positioning

References

[1] Steve Yegge. *The AI Vampire*. Coined the term "AI vampire" to describe cognitive fatigue from AI-assisted coding; documents "Nap Attacks" and productivity limitations. Feb. 2026. URL: https://steve-yegge.medium.com/the-ai-vampire-eda6e4f07163.

[2] Fortune Magazine. *In the workforce, AI is having the opposite effect it was supposed to, UC Berkeley researchers warn*. UC Berkeley research on AI causing workload creep, cognitive fatigue, and burnout. Feb. 2026. URL: https://fortune.com/2026/02/10/ai-future-of-work-white-collar-employees-technology-productivity-burnout-research-uc-berkeley/.

[3] Haystack Analytics. *83% of Developers Suffer From Burnout, Haystack Analytics Study Finds*. Survey finding that 83% of software developers suffer from burnout; top causes: high workload (47%), inefficient processes (31%), unclear goals (29%). 2021. URL: https://www.usehaystack.io/blog/83-of-developers-suffer-from-burnout-haystack-analytics-study-finds.

[4] Harvard Business Review. *AI Doesn't Reduce Work—It Intensifies It*. Documents burnout from constant review and verification of AI-generated code. Feb. 2026. URL: https://hbr.org/2026/02/ai-doesnt-reduce-work-it-intensifies-it.

[5] Stephen Kaplan. "The restorative benefits of nature: Toward an integrative framework". In: *Journal of Environmental Psychology* 15.3 (1995). Foundational paper on Attention Restoration Theory; directed attention fatigue from sustained cognitive effort is restored by engagement with environments that provide fascination without requiring directed attention, pp. 169–182.

[6] Ömer Erdem Koçak. "Recovery from work by playing video games". In: *Applied Psychology* (2024). Investigates video games as recovery from work stress; finds harmonious gaming passion facilitates cognitive resource accumulation. URL: https://iaap-journals.onlinelibrary.wiley.com/doi/10.1111/apps.12519.

[7] PPC Land. *Claude Code reaches 115,000 developers, processes 195 million lines weekly*. Official statistics on Claude Code user base and adoption metrics. July 2025. URL: https://ppc.land/claude-code-reaches-115-000-developers-processes-195-million-lines-weekly/.

[8] Business of Apps. *Claude Revenue and Usage Statistics (2026)*. Comprehensive Claude platform statistics including 30M MAU and API call volumes. 2026. URL: https://www.businessofapps.com/data/claude-statistics/.

[9] LeadDev. *Burnout is on the rise as layoffs reshape the tech industry*. Using validated Maslach Burnout Inventory: 22% of 617 surveyed engineering leaders and developers face critical burnout levels; 24% moderately burned out. 2025. URL: https://leaddev.com/culture/engineering-burnout-rising-2025-layoffs-reshape-tech-industry.

[10] Latitude. *AI Dungeon*. Pioneering LLM text adventure launched 2019; reached 100,000 players in first week and 1.5 million by June 2020; retired from Steam March 2024. 2019. URL: https://aidungeon.com/.

[11] Wikipedia Contributors. *AI Dungeon*. Historical overview of AI Dungeon's development, LLM transitions, and timeline. 2024. URL: https://en.wikipedia.org/wiki/AI_Dungeon.

[12] BigGo News. *Bash Screensavers Revive Terminal Art with 90s Nostalgia and Modern Whimsy*. Documents 2025 revival of ASCII art in developer terminals. Oct. 2025. URL: https://biggo.com/news/202510281917_Bash-Screensavers-Terminal-Art.

[13] Data Insights Market. *Opportunities in Roguelike Game Market 2025–2033*. Market research showing roguelike market at $1.33B in 2025, projected $2.57B by 2033 with 8.7% CAGR. 2025. URL: https://www.datainsightsmarket.com/reports/roguelike-game-1986499.

[14] AI21 Labs. *How Latitude scaled production of their gaming worlds while reducing costs*. Documents Latitude's transition from OpenAI to AI21's Jurassic-1 model due to policy conflicts. 2023. URL: https://www.ai21.com/blog/latitude-case-study/.