Vibe Coding & The Diminishing Role of Copyright in AI-Generated Software

Related Professionals

Jason Mueller

Related Services

Copyrights

Intellectual Property

CLIENT ALERT | 2.11.2026

It is axiomatic that "ideas are not protectable" under the constitutional protections for intellectual property (IP). Rather, protection inures only upon unique creative expression of an idea (copyright); using some manifestation of the idea as a source-indicator (trademark); or reducing the idea to practice (patent). While this is not a comprehensive statement of what jurisdictions require to protect IP, it is a hasty distinction of three primary sources of IP legal rights. Other categories of IP include industrial or patentable designs, trade secrets and variations that exist across international jurisdictions. This article focuses on the idea/expression dichotomy that is primarily a concept developed in analyzing copyright protection—which has been the primary source of legal protection for software code.

The US Supreme Court has long recognized that ideas are not protectable under copyright law. In *Harper & Row Publishers, Inc. v. Nation Enterprises*, 471 U.S. 539, 556 (1985), the Court affirmed the foundational principle—rooted in 17 U.S.C. § 102(b)—that "[n]o author may copyright his ideas or the facts he narrates," limiting protection to the specific expression of those ideas. This idea-expression dichotomy, tracing its modern articulation to *Baker v. Selden*, 101 U.S. 99 (1879), has profound implications in the context of vibe coding, an emerging development practice in which programmers or product teams provide high-level, intuitive, natural-language prompts to advanced AI coding agents such as Claude Code.

Last week the *Wall Street Journal* reported that "Vibe coding, the process of turning a text prompt into actual software, has taken the AI world by storm." And in late January 2026, Boris Cherny—head of Claude Code at Anthropic and its creator—publicly stated that he personally hasn't written any code by hand in over two months. He claimed 100% of his code (including shipping 20+ pull requests per day) is now generated by Claude Code powered by models like Opus 4.5/4.6.

These prompts typically describe desired functionality, user experience, "vibe," or high-level architecture rather than supplying detailed, line-by-line specifications or original expressive code. The AI then generates

the bulk of the implementation, logic, structure, and even ancillary elements such as tests and documentation. In traditional software development, human authorship is evident through the selection, arrangement, and creative refinement of code.

Vibe coding disrupts this model. Human contribution in a vibe approach consists primarily of conveying unprotectable ideas and high-level directives, not protectable expression. Thus the resulting code may fail to satisfy the originality and human authorship requirements for copyright protection.

The U.S. Copyright Office has consistently held, in its post-2023 guidance and reports on AI-generated works, that outputs lacking sufficient human creative control are not eligible for registration. Judicial precedent reinforces this position. In Thaler v. Perlmutter, the D.C. Circuit (affirming the district court) confirmed that copyright demands human authorship; purely AI-generated material stands outside the scope of protection. When prompts remain at the "vibe" level—vague descriptors such as "create a modern, intuitive dashboard with clean animations and responsive design"—the AI likely performs the unpredictable, detailed expressive work. The final codebase therefore risks being treated as unprotectable or only weakly protectable, limited perhaps to any subsequent human edits, curation of modules, or overarching selection and arrangement.

But does it matter? Vibe coding can have significant consequences for the product and code lifecycle. Software development cycles have always been relatively short compared to other creative industries, with meaningful competitive differentiation often lasting weeks or months before iteration becomes necessary. Vibe coding compresses these timelines dramatically, enabling near-real-time prototyping, deployment, and refinement. In such an accelerated environment, the economic value of any static copyright in the code's expression diminishes sharply. The protected asset is no longer a fixed textual artifact but the living product: ongoing feature velocity, proprietary integration patterns, user data flywheels, and real-world performance insights that are difficult to replicate through prompting alone. Companies engaged in heavy vibe coding thus derive less strategic advantage from asserting copyright claims over the generated code and more from maintaining trade secret protection over proprietary prompting strategies, internal fine-tuning data, architectural decisions informed by production telemetry, and non-public system-level optimizations. (For a primer on trade secrets, see here).

From a broader economic policy perspective, IP regimes—particularly copyright in software—exist primarily to address the public-goods problem inherent in information goods. Creation entails high fixed costs (talent, time, and expertise), while marginal costs of copying and distribution approach zero, creating a risk of underinvestment absent temporary exclusivity. Vibe coding fundamentally alters this calculus by collapsing the fixed costs of code production. But it is also more likely to disqualify the output from copyright protection.

Balance your approach to fit your needs and protect what you can. What once required teams of skilled engineers laboring over weeks or months can now be accomplished through sophisticated prompting and iterative human oversight in a fraction of the time. In economic terms, this reduction in production costs weakens the marginal incentive effect of strong expressive copyright. Policy analysis suggests that when technological change lowers the cost of creation so substantially, optimal legal frameworks should tilt toward greater diffusion and follow-on innovation rather than extended monopolies over expressive output.

Pausing to capture and submit a snapshot of a version of code – as often done in a copyright protection strategy - risks impeding the very progress and speed the business seeks to promote. For innovators seeking to protect their IP on vibe-coded projects, several practical considerations follow:

- First, robust documentation of the development process becomes essential: detailed records of prompts, iteration histories, human review notes, significant edits, and creative judgments can help establish the requisite human authorship for copyright registration or enforcement, or lay a foundation for showing a proprietary (and otherwise non-discoverable) market advantage.
- Second, hybrid workflows—combining high-level prompting with deliberate human-sourced iterative improvements, novel algorithmic contributions, or creative architectural layering—can offer a stronger path to protectability than pure vibe-driven generation.
- Third, continue to flag elevated infringement risks arising from the training data of the underlying models (although a detailed discussion of this third point is outside the scope of this blog post).

In sum, vibe coding does not eliminate the relevance of IP, but it reorients it. The traditional emphasis on copyright in the expressive text of software yields to a greater reliance on trade secrets, method patents (where novel technical solutions are claimed and protectable), contractual protections, data advantages, and the competitive discipline of relentless execution velocity. As this practice matures, courts and policymakers will need to grapple with whether the idea-expression dichotomy, forged in an analog era, continues to serve its constitutional ends in an age when machines can translate ideas into functional code at unprecedented speed and scale.