

Google v. Oracle: What We Learned from Oral Argument

October 22, 2020 – *The TMCA*

by Connor J. Hansen and Stefan Szpajda



Related People

- Connor J. Hansen

Related Capabilities

- Trademark, Copyright + Advertising
- Intellectual Property Litigation

On October 7, 2020, the Supreme Court heard arguments in *Google v. Oracle*, a decade-long battle challenging Oracle's claim to own copyrights in certain aspects of its Java software platform that Google implemented in Android phones without taking a license.

We've been tracking this case since last December, with deep dives on each of the questions presented to the Court: (1) whether copyright protection extends to software interfaces, and (2) whether, as the jury found, Google's use of Oracle's software interfaces in the context of creating a new computer program constitutes fair use. Over approximately ninety minutes on the record, the Justices peppered counsel with more than 50 questions on these topics that, on balance, signaled skepticism of Google's positions on each of the issues before the Court.

Scope of Copyright Protection

Before considering the first question the Court will address, recall what this case is **not** about. The Court will not decide whether computer programs can be protected under the Copyright Act—that’s answered by the plain text of the Copyright Act itself, which confirms that they can. Instead, the question is narrower: is there a subset of computer code that is so functional that it cannot be protected by the Copyright Act? Google says yes, and puts Oracle’s declaring code in that category. But despite widespread industry support for Google’s position, the Court seems poised to reject this argument.

Method of Operation

Under 17 U.S.C. § 102(a), copyright protection extends to original works of authorship, and § 101 expressly lists computer programs among the works of authorship that can be protected. But, per § 102(b), copyright does not extend to “any idea, procedure, process, system, method of operation, concept, principle or discovery” embodied in the work. As we detailed in March, Google asks the Court to distinguish declaring code (i.e., the computer code that “declares” what the computer program will do, which Google alternately refers to as “software interfaces”) from implementing code (i.e., the computer code that actually performs the task stated in the declaring code), arguing that although implementing code may be protectable, declaring code is not because it is a method of operation. More specifically, Google takes the view that declaring code is distinguishable from implementing code because it is a software interface—a set of commands that instructs a computer what to do, not how to do it.

Google’s argument that declaring code does not warrant copyright protection is premised on the proposition that it is purely functional and not expressive. According to Google, because declaring code allows one program to efficiently communicate with another, it constitutes an essential building block of software development that allows different computer programs to seamlessly communicate with each other. In its briefing and at oral argument Google analogized the dispute to *Baker v. Selden*, a Supreme Court decision from 1880 holding that a copyright in a book about an improved system of book-keeping did not entitle the author to prevent others from using the ledger system described in the book. A problem with Google’s reliance on *Baker*—beyond the disorienting leap from 19th century bookkeeping to smartphones—is that Google didn’t copy just the idea of using declaring code or the functions performed by the declaring code, it copied the expressive content of the declaring code. That is not true of the ledger system described in *Baker*. As Oracle put it, “this case would be like *Baker* if we were trying to block others from using their own [declaring code] to organize their own prewritten programs. But [Oracle] wrote its own specific layout and filled in the blanks 30,000 times over. We seek to protect only that fully realized expression.”

The Justices struggled with Google’s reasoning. Justice Roberts noted that Oracle was attempting to protect its “particular expression,” and not the functionality of the declaring code. Justice Thomas wondered why § 101’s identification of computer programs as protectable did not end the inquiry, while Justice Alito worried that Google’s argument that declaring code is a method of operation would put all computer code at risk of losing

copyright protection. Justice Gorsuch deemed Google’s decision to “mov[e] past” its § 102(b) argument a “wise move given the fact that 101 says computer programs, including statements or instructions, in order to bring about a certain result, may be copyrighted.” Justice Kavanaugh wondered whether Google was still making the method of operation argument while pressing Google on why its reasoning would not swallow the Copyright Act’s protection for computer programs. Justice Breyer found it “pretty tough” to understand how declaring code differed from any other computer program. Justice Kagan noted that she was “a little bit surprised or confused” by the arguments made by Google, which appeared to abandon § 102(b) in favor of the merger doctrine (discussed below). And Justice Sotomayor worried that Google could not draw a “clean line” to define a method of operation in the context of source code.

Despite all that, Google had at least one bright moment when Justice Sotomayor implied in a question posed to Oracle that it is well-settled that declaring code is not copyrightable. She further summarized the industry’s understanding of that precedent as: “they can copy only what’s necessary to run on the application, but they have to change everything else.” She then asked: “Why should we change that understanding?” Oracle disputed Justice Sotomayor’s interpretation of this precedent and argued that not a single court has drawn such a distinction between declaring and implementing code. It does seem a stretch to state that the decisions referenced by Justice Sotomayor held declaring code is not copyrightable, and in its briefing, even Google did not argue so. In fact, Google acknowledged in its [petition](#) that “the courts of appeals are deeply divided on the appropriate standard for determining the circumstances under which a software interface is copyrightable.”

Aside from Justice Sotomayor’s remarks, the Court seemed skeptical that software interfaces—as distinct from other types of computer code—are unprotectable methods of operation.

Merger Doctrine

Even if declaring code is not a method of operation, Google argues it should be excluded from protection by the merger doctrine. Under the merger doctrine, if an idea can only be expressed in one (or very few) ways, then the expression of the idea “merges” with the idea and cannot be the subject of copyright protection. The merger doctrine is intended to prevent one entity from claiming ownership over an idea merely because it was the first to discover the only way to express it. A classic example illustrating the merger doctrine is the rules to a game. A rule can be expressed in writing, and thus can be protected by copyright. But if that rule is so basic that granting copyright protection would preclude the use of the underlying idea itself in other games, then the merger doctrine denies copyright protection because to hold otherwise would be to grant a *de facto* copyright to the idea itself.

Google’s argument that the merger doctrine should apply to declaring code is premised on the fact that there is only one way to write Oracle’s declaring code, and thus granting Oracle a copyright to it would preclude others from using the functionality it describes. But Deputy Solicitor General Malcolm Stewart (arguing on behalf of the United States in

support of Oracle) zeroed in on a weakness in Google's position. Acknowledging that there is only one way to write the declaring code, he argued that Google's premise is circular because Google defines the underlying function—i.e., the idea that the declaring code's expression would need to monopolize for the merger doctrine to apply—as invoking implementing code in response to the calls made by the declaring code. This is a problem for Google because under § 302(a), copyright protection arises from the time of the work's creation, and when Oracle's predecessor-in-interest wrote the declaring code there were no ideas independent of the code itself that the declaring code could have monopolized. And, per Deputy Solicitor General Stewart, the dispute arose not because Oracle's declaring code is the only way to accomplish the functions it expressed, but because Oracle's particular way of expressing it became popular among developers. Deputy Solicitor General Stewart then offered the Court a bright-line rule: the merger doctrine should apply to computer code if a particular line of code is, without regard to the "acquired expertise of other actors, the only way to make the computer perform a particular function." By this reasoning, the merger doctrine does not apply where, at the time the code was written, there were multiple options available to whomever wrote it.

Here, again, the Court struggled with Google's position. Judge Roberts noted that there are "a lot of ways" to write the functions ultimately expressed by the declaring code when Oracle wrote it. Justice Sotomayor pressed Google to explain how it could acknowledge that the merger doctrine does not apply to implementing code, while insisting that it applies to declaring code. Justice Kagan observed that the problems Oracle solved with its declaring code could have been solved in other ways, but Oracle "happened to come up with a particularly elegant one." Justice Gorsuch echoed his colleague, characterizing Google's argument as a "wish to share the facilities of a more successful rival because they've come up with a particularly elegant or efficient or successful or highly adopted solution." Justice Kavanaugh criticized the circularity of Google's argument, noting that Google appeared to "define the relevant idea" in terms of what it copied, adding—no doubt to the consternation of any engineers who might have been listening—that "[y]ou're not allowed to copy a song just because it's the only way to express that song."

In the end, the Court seemed receptive to the argument that because at the time Oracle created the declaring code it could have written it in many other ways to achieve the same functionality, the merger doctrine should not be invoked to deny Oracle copyright protection to its declaring code.

Fair Use

On the second issue before the Court, i.e., whether, as the jury found, Google's use of Oracle's software interfaces in the context of creating a new computer program constitutes fair use, the Justices revealed less through their questioning.

The Four-Factor Test

The Justices spent comparatively little time probing the individual fair use factors under § 107. Of the four factors, the Court seemed most interested in whether Google's use of

Oracle's declaring code in Android (a smartphone, as opposed to a desktop, operating system) was transformative. Justice Thomas wondered what a transformative use might look like in the context of source code, which by its nature is intended to operate the same way in each environment in which it is used. Picking up on this theme, Justice Sotomayor seemed dissatisfied by Oracle's argument that adopting Java for the smartphone environment was not transformative. Justice Kagan took it one step further, wondering whether transformative use "isn't the right question here, although it is in other contexts." Taken together, these questions hint at the possibility that the Court will comment on how lower courts should consider the purpose and character of the use of computer code in future fair use analyses.

Standard of Review

Substance aside, procedural questions loom large over how the fair use issue came to the Court. Before the hearing, the Court ordered supplemental briefing on whether the Federal Circuit applied the proper standard of review when it reversed the jury's determination finding fair use. In sum, Google argues that the Federal Circuit should have applied the more deferential "reasonable juror" standard to the jury's finding of fair use, rather than the *de novo* standard of review. Oracle responds that the Federal Circuit correctly reviewed the findings *de novo*, but that it is immaterial which standard of review applies because the Federal Circuit also found that "no reasonable jury" could have found in Google's favor on fair use. Thus, Oracle contends it would have prevailed even if the Federal Circuit had formally applied the more deferential standard of review.

Although the conventional wisdom appears to be that the Federal Circuit did something unusual when it overturned the jury's finding of fair use, each side took pains to establish the other's desired outcome as unprecedented. Google reiterated that before the Federal Circuit's decision in this case, "no prior appellate court ever overturned a fair use verdict." Oracle, however, countered that "[n]o court has found fair use or upheld a fair use verdict where a copyist copied so much valuable expression into a competing commercial sequel to mean the same thing and serve the same purpose as the original." Indeed, Oracle challenged the premise and the weight that should be afforded to the uniqueness of the Federal Circuit's decision overturning the jury's finding of fair use, noting that Google could only identify 5 cases that went to a jury on the issue of fair use in the last 30 years. In contrast, in the same timeframe, over 100 fair use cases were decided as a matter of law on summary judgment. Oracle also stressed that there is nothing about fair use that prevents an appellate court from overturning a district court's determination and referenced cases in which appellate courts did just that.

At the hearing, however, the Justices directed less attention to procedural questions than anticipated, and the Court offered few clues as to which way it might rule on this aspect of this case. The most overt statement came from Justice Alito when—echoing Oracle—he asked what he should do if he finds that factors 1 and 4 of the fair use analysis "weigh very heavily" against Google, and that as a result a jury "couldn't reasonably find" in Google's favor.

Industry Impact

If you've managed to wade through the ins-and-outs of this dispute, you may still be wondering why it is so hotly contested. The reason, in part, is its potential to profoundly affect how engineers do their jobs throughout the software industry. What the rather doctrinal questions before the Court tend to obscure, and what many amicus briefs filed by computer scientists, scholars, and representatives of the software industry passionately argue, is that a ruling in Oracle's favor may force tens of thousands of engineers to abandon longstanding practices that include reusing declaring code without a license. Per the amici, doing so will make the process of coding computer programs less efficient, will stand in the way of innovation, and will hurt consumers. So while Google stands alone as the petitioner, many engineers see the company as a proxy for their own concerns. Oracle, of course, has its own policy advocates, though they tend to focus at a higher level of abstraction on the incentives the Copyright Act is intended to promote and protect.

The Court is well aware that its decision will reach beyond the narrow questions before it, and each Justice devoted attention to the policy concerns expressed by the parties and many amici. Several Justices, however, appeared to downplay the stakes. Justices Kavanaugh and Alito appeared to dismiss concerns that the "sky will fall" if the Court rules in Oracle's favor. Sticking with the same metaphor, Justice Sotomayor in turn dismissed concerns that the sky will fall if the Court rules in Google's favor.

If there is one broad takeaway from the case, it is that there is a significant disconnect between how lawyers and judges focused on the text and ostensible purpose of the Copyright Act view this dispute, and how the engineers who will be most directly affected by the Court's decision view it. When you strip away all the legal doctrines and technical jargon, *Google v. Oracle* is just another reminder that law and policy do not always align in a way that those affected by it find intuitive.

Conclusion

The questions above are now for the Court to answer. We will provide another—perhaps final—update, unless the Court remands on fair use, once the opinion issues in 2021.