Copyright Case of the Century Decided: Supreme Court Rules in Google's Favor in $9 Billion Software Dispute

May 3, 2021 Jason Bloom, Lee Johnston

PRACTICES Media Entertainment and Sports, Intellectual Property, Copyright, Intellectual Property Litigation, Media and Entertainment Litigation

At long last, and after more than a decade of litigation, the United States Supreme Court has ruled in the multi-billion dollar copyright dispute between Google LLC and Oracle America, Inc. In a matchup that could be described as Goliath vs. Goliath, the technology giants waged war for years over Google's unlicensed use of Oracle's Application Programming Interfaces ("API") in early versions of its Android smartphone platform. In a case involving two jury trials, two Federal Circuit appeals, and two petitions for certiorari to the Supreme Court, both sides had wins and losses along the way, but Google ultimately prevailed before the Supreme Court and was absolved of liability.

The case involved two key issues: (i) whether Oracle's APIs were subject to copyright protection; and (ii) whether Google's use of the APIs was a fair use. The Court dodged the copyrightability question, which had the potential to send shockwaves throughout the industry, and instead focused its analysis on the fact-intensive fair use defense. By doing so, the Court potentially limited the direct impact of its decision to the dispute at hand and ones like it. But the Court's broad application of the fair use analysis is likely to lead to greater reliance on the defense in the software industry and beyond. Unfortunately, what the Court's opinion failed to do is add much-needed clarity and certainty to what has long been a convoluted, inconsistently-applied, and murky area of copyright law.

How Did This Start?

More than 15 years ago, Google set out to develop a software platform for smartphones. Sun Microsystems' (Oracle's predecessor) Java SE program offered a desirable solution. Ubiquitously used in computers, programmers know Java well. If Google could integrate Java into its Android platform, then programmers could readily develop new, compatible smartphone programs, making the Android smartphone even more desirable.

Unable to agree on the terms of a license with Sun, Google set out to develop its own Android platform. The end-product included millions of lines of novel code, but also copied approximately 11,500 lines of Oracle's API tool from Java. Could Google have developed its platform without the copying? Sure. But without Oracle's code, programmers could not use the API tool they know so well.

Oracle's API: Explained

Oracle's API gives programmers access to thousands of prewritten computing tasks (each, called a "method") with the use of simple commands. This is tremendously useful. So long as programmers understand the API's commands, they can integrate the API's methods into their programs. Without the API, programmers would have to write their own code from scratch to accomplish the same computing functions.

The API functions in three parts. First, programmers enter a short-form command (called a "method call") associated with the desired method. Second, the API's declaring code facilitates retrieval of that method. The declaring code organizes *thousands* of methods into classes and those classes into packages. Once a method call is entered, the declaring code reads it, provides the name and location of the method within its organizational structure, and calls up the method. Finally, the API's implementing code tells the computer how to execute the method.

Although Google wrote its own implementing code, it copied 37 packages of Oracle's declaring code, allowing programmers to rely on the familiar Java method calls to access Google's computing tasks. Without Google's copying, programmers would have had to learn an entirely new system to program for the Android platform.

The $9 Billion-Dollar Question(s)

Oracle sued Google, claiming that Google's copying infringed both its copyrights and patents. Its copyright claims raised two key issues: first, whether Oracle's declaring code is copyrightable, and second, whether Google's copying of the code was fair use. Google secured its first win in 2012 from the District Court of Northern District of California, which found that the declaring code was not copyrightable because it was merely a "system or method of operation"—which copyright law excludes from protection. The Federal Circuit reversed, and directed the District Court to consider the fair use defense. In 2016, a jury found Google's use was fair use but, again, the Federal Circuit reversed, and Google's petition for writ of certiorari to the Supreme Court followed.

A Victory for Fair Use

Although fair use is an affirmative defense to copyright infringement, the Supreme Court declined to determine whether the declaring code is actually copyrightable. Instead, the 6-2 majority (with Justice Barrett not participating) only examined the notoriously flexible, fact-specific test for fair use. The test considers: "(1) the purpose and character of the use, including whether such use is of a commercial nature or is for nonprofit educational purposes; (2) the nature of the copyrighted work; (3) the amount and substantiality of the portion used in relation to the copyrighted work as a whole; and (4) the effect of the use upon the potential market for or value of the copyrighted work."

On each of these factors, the Court found for Google.

Factor 1: The Nature of the Work

Here, the Court contrasted implementing code—which necessarily involves creative design and writing—with declaring code—which is inextricably bound with uncopyrightable ideas (such as task division and organization). Thus, the Court found that declaring code's value is not derived from creativity, but rather from the number of programmers who learn to use it. Because copyright law seeks to protect creative expression, the Court concluded that declaring code is farther from the core of what copyright law protects and, thus, that this factor weighs in Google's favor.

Factor 2: The Purpose and Character of the Use

Traditionally, for this factor to weigh in favor of fair use, the copier's use must "add something new" or "transform" the use of the copyrighted material. Although the Court concluded that Google's use of the declaring code served the same function as Oracle's (i.e., to enable programmers to integrate methods from the API into their own programs), the Court nevertheless found this factor weighs in Google's favor. Specifically, the Court concluded that Google's use was "transformative"

because it: (1) sought to create new products and (2) enabled programmers to access a new collection of computing tasks in a different computing environment (smartphones as opposed to computers).

The dissent, by Justice Thomas and joined by Justice Alito, opined that the majority's novel application of the "transformative" analysis eviscerates copyright. Indeed, courts routinely reject a fair use defense where the copier merely creates a new product without altering the original with "new expression, meaning, or message." Just days before the Supreme Court's decision, the Second Circuit rejected Andy Warhol's fair use defense against infringement of a photograph of pop-star Prince that Warhol used to create artistic prints. The Second Circuit found that Warhol's work could not "stand[] apart from the 'raw material' used to create it" and, thus, could not be transformative. Here, the Supreme Court's transformative analysis is a far cry from that standard. Indeed, Google's copying would have been hard-pressed to survive that test.

Factor 3: The Amount and Substantiality of the Portion Used

Although Google copied only a small quantitative amount of the API (approximately 11,500 lines of 2.8 million), Oracle contended the taking was qualitatively substantial. The Court disagreed and found that Google's taking was not intended to usurp creativity, but rather to promote it. Thus, the copying was not "substantial," because it served a valid, transformative purpose.

The dissent criticized the majority's analysis for concluding that 11,500 lines of declaring code was a quantitatively insubstantial part of the entire Java platform, including the implementing code. The dissent argued that the proper frame of analysis was to compare the volume of declaring code copied to the volume of declaring code in the Java platform. Under such an apples-to-apples analysis, the dissent contended the copying was quantitatively substantial.

Factor 4: Effect on the Market

Finally, although Oracle claimed it was due more than nine billion dollars in damages as a result of Google's copying, the Court was unpersuaded that the market effects were in Oracle's favor. The Court concluded that Android was not a market substitute for Java's software and, in fact, that Oracle benefitted from Google's expansion of Java into the smartphone market. And while the dramatic value Google derived from the API was undeniable, the Court declined to attribute that value to Oracle's copyright. Instead, the Court found that the value was a product of the time programmers have invested to learn the API—which copyright law does not protect. Taken together, the Court concluded all factors weighed in favor of fair use.

What's Next? Implications of the Google-Oracle Decision on Software Development

The Court's decision undoubtedly was met with sighs of relief from software engineers tasked with the job of developing interoperable, scalable software solutions. To many in the software industry, the outcome of this decade-long dispute validated what had been considered a "best practice" by developers – the re-implementation of API declarations. But, does the Court's ruling mean there is now an "open season" on APIs? Far from it.

The Court's decision to side-step the threshold question of copyrightability of API declarations and instead focus on the highly fact-specific analysis of the fair use factors underscores the narrowness of its decision. The Court made it clear that it did not view its decision as one that overruled its prior decisions or dramatically altered the copyright fair use legal landscape. Indeed, the Court used existing precedent to characterize API declarations as being remote from the core of what copyright

seeks to protect (i.e., creative expression), and thus, more amenable to fair use than the code used to implement the API declarations.

The limited nature of the Court's decision and the specific facts surrounding Google's conduct provide important lessons for software development going forward. First, Google took only what it needed from Oracle's API declarations to ensure interoperability. Equally important, Google utilized a well-vetted "clean room" protocol to write and test the API's code used to implement the API declarations. Software developers using APIs should take heed of Google's prudent approach.

In Sum

While the long-term impact of the decision on technology companies and software developers remains to be seen, it is anticipated that fair use will be lodged as a defense more frequently in software cases going forward. Nevertheless, parties should exercise extreme caution before using another's copyrighted work and should rarely rely on fair use as a get out of jail free card in their decision-making process. Fair use is an extremely fact-intensive inquiry that is inconsistently applied throughout the federal courts. The fact that the Federal Circuit and two justices of the Supreme Court disagreed with the applicability of fair use to this case illustrates this point. And, although Google ultimately prevailed, it was only after spending millions of dollars in attorneys' fees and more than a decade in time-consuming litigation, a burden most companies could not easily bear. And, because the Supreme Court's opinion did little to add clarity or certainty to the applicability of fair use outside of the Google/Oracle case, relying on fair use in other contexts could be risky business.