ILLINOIS STATE BAR ASSOCIATION

SEPTEMBER 2021 VOL 61 NO. 1

Intellectual Property

The newsletter of the Illinois State Bar Association's Section on Intellectual Property Law

Google v. Oracle: U.S. Supreme Court Whiffs on a Chance to Declare Code's Status

BY PHILLIP R. VAN NESS

I must confess that this has been one of the most difficult cases to review. While appellate decisions that leave some issues unresolved for later are not unusual, this one, in my opinion, is particularly notable in a negative sense, both for the shamefully extended process inflicted upon the litigants to get to a conclusion and for what seems an unnecessarily convoluted course of reasoning that invites further litigation for no compelling reason.

In its April 5, 2021 decision in *Google LLC v. Oracle America Inc.* [18-956], the Supreme Court overwhelmingly [6-2] sided with Google in its long-running copyright dispute with Oracle. The decision generally pleased tech companies but upset the movie and recording industries as well as publishers and authors, not to mention the Trump administration, which sided with Oracle in this battle of heavyweights, although Google is obviously the larger of the two combatants. The monetary stakes were impressive: Oracle had claimed damages in the billions.

In a nutshell, the high court gave its blessing to Google's admitted outright copying of over 11,000 lines of code developed by Oracle [actually its predecessor, Sun Microsystems] for its Java platform for computers and tablets. The asserted reason for using all that code was to enable Google to use the familiar computer programming shortcuts and protocols employed by Java, in order to develop Google's Android operating system for smart phones. The Court concluded that such extensive copying nevertheless fell within the doctrine of fair use.

Practitioners familiar with the fair use doctrine might find themselves torn by this decision. The dissenters [Clarence Thomas and Justice Samuel Alito] seemed less torn than outraged, but it is possible to argue both sides without blushing; even Justice Breyer, author of the majority decision, noted the "thoughtful dissent" by Justice Thomas. Clearly, where one stands affects one's view.

The crux of the justices' differences lay in the difference between what programmers call "Implementing Code" vs. "Declaring Code." To understand the decision, the reader must understand the difference between those species of computer code.

To create the Android program, which was released in 2007, Google wrote millions of lines of computer code. The vast majority of that code by volume was "Implementing Code" and virtually all of it was brand-new, not derived from Oracle's Java program. However, to make all this new code work, Google also used about 11,500 lines of the "Declaring Code" embedded in Oracle's Java platform. This was apparently a big chunk of the "Declaring Code" component of the Java program, though obviously a tiny fraction of the overall Android program.

In support of his conclusion that such naked copying was nonetheless fair use, Justice Breyer wrote that Google "took only what was needed" and that "Google's copying was transformative," meaning a use that "adds something new and important." To reach that conclusion, however, he had to articulate how copying "Declaring Code" cannot be lumped together with copying "Implementing Code." In doing so, he employed a number of analogies. In all candor, I found some of his analogies ridiculous, even if his result was ultimately sensible. My quarrel was with his analogies, not with his logic; there were manifestly better analogies he could have used, in my opinion; in fact, some of his analogies were appropriate and helpful, but Breyer's attempts to augment those with additional analogies created needless confusion and provided an opening for Justice Thomas to pounce.

With little thanks to Breyer, the difference between the two species of computer code, as I understand it, is that Declaring Code provides the background structure and shorthand instructions for navigating through a program, while

Implementing Code articulates the tasks to be performed. At one point, Breyer used a robot as part of an analogy to explain how computer programs work, in which Declaring Code [playing the part of the robot] uses a set of shorthand predefined commands to find a recipe [playing the role of Implementing Code] and delivers it to the cook [playing the role of the computer] to prepare the dish. That was helpful if a bit odd, but for some reason he felt it necessary to try other sets of analogies, including one in which he stated that "the declaring code's shortcut function is similar to a gas pedal in a car that tells the car to move faster or the QWERTY keyboard on a typewriter that calls up a certain letter when you press a particular key." As strained as the robot analogy is, the QWERTY and gas pedal analogies are off-the-charts worse, and unfortunately masks the true functions of Declaring Code, which [as Breyer elsewhere does note] are two: to bundle commonly-used predefined tasks into packages of instructions, and to provide a programming structure in which these packages of instructions are stored and retrieved; the latter function is better explained by Breyer by analogy to the Dewey Decimal System by which books are categorized. I wish only that Breyer had stuck with the Dewey Decimal System and maybe the robot; anyone familiar with the QWERTY keyboard [which is all of us], knows that pressing a key to get a designated letter or number bears no more resemblance to a set of computer instructions than a box of letters from a Scrabble game bears to an encyclopedia.

In any event, Breyer and the majority sided with the district court judge who had conducted a six-week jury trial, and rebuked the federal circuit, which had reversed the trial judge.

Notably, the original trial had addressed the issues in this case on both patent and copyright grounds. While the jury rejected Oracle's patent claims outright, it deadlocked on the copyright claims, specifically on the fair use defense. To break that impasse, the trial judge held that "Declaring Code" could not be copyrighted, since it amounted to merely a "system or method of operation" within the meaning of 17 U.S.C. §102(b). For the casual reader, here is what that section says:

"(b) In no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work."

In rejecting that part of the trial judge's view, the federal circuit held that declaring code was copyright-eligible, noting that Google could have created its own declaring code scheme and structure if it had wanted to. Yet it also rejected Oracle's request that it decide Google's fair use defense as well, asserting that this "is not a case in which the record contains sufficient factual findings upon which we could base a de novo assessment of Google's affirmative defense of fair use." [750 F. 3d, at 1372-1373]. So, it remanded the case back to the district court for another trial on that issue. Though Google appealed that decision to the Supreme Court, its petition was denied, so back to the district court it went. Another trial, another jury, another blow to Oracle: After three days, the jury decided that indeed Google had demonstrated that its copying of Oracle's Declaring Code was a fair use. Once again, Oracle appealed to the federal circuit.

When the case re-appeared in front of the federal circuit, it reversed the district court once again. But this time, it addressed the fair use question head-on, holding that, even assuming all facts in Oracle's favor, the question of fair use is a question of law, not fact, and then held as a matter of law that Google's copying of all that declaring code was fair use.

As an aside, it is difficult for your author to reconcile how the federal circuit could remand an issue for a trial, allegedly to rectify the lack of "sufficient factual findings," then decide the same issue as a question of law.

Then again, it is difficult for your author to understand why the Supreme Court accepted the case on the second bounce but not the first, or why the majority did not simply answer the core question: is Declaring Code copyrightable? Instead, it merely stated that: "the Court assumes for argument's sake that the copied lines can be copyrighted, and focuses on whether Google's use of those lines was a fair use." [Op. cit at 5]. It then labored for 19 pages through the familiar four factors set forth in the fair use statute at 17 U.S.C. §107, namely:

A. The Nature of the Copyrighted Work;

B. The Purpose and Character of the Use;

C. The Amount and Substantiality of the Portion Used; and

D. Market Effects.

In each case, Breyer and his majority struggled mightily to bend those factors negatively to Oracle and in favor of fair use. This required resort to tortured logic, as where the majority declared the "amount and substantiality" test in favor of fair use despite acknowledging that those copied lines of Declaring Code "amount to virtually all the declaring code needed to call up hundreds of different tasks" [*Op. cit* at 32]. Similarly, it declared that the "market effect" on Oracle was negligible, since "Sun's many efforts to move into the mobile phone market had proved unsuccessful" [*Op. cit* at 35]. This prompted Justice Thomas to remind the majority that the record showed that Google had tried repeatedly to negotiate a license deal with Oracle for its code, but that:

"when the companies could not agree on terms, Google simply copied verbatim 11,500 lines of code from the library. As a result, it erased 97.5% of the value of Oracle's partnership with Amazon, made tens of billions of dollars, and established its position as the owner of the largest mobile operating system in the world." [*Op. cit* at 44].

Apparently, the "amount and substantiality" as well as the "market effect" of Google's copying were more obvious to Google than it was to the Court.

In any event, the majority's strained reasoning, not to mention two rounds of trials and appeals, could have been avoided by simply acknowledging, as had the trial judge, the plain import of 17

U.S.C. §102(b), and holding that, in this case, the Declaring Code was no more than a method of operation and thus ineligible for copyright protection. This is, in effect, what Breyer did in the guise of addressing the "amount and substantiality" test of fair use, when he stated that "Google copied those lines not because of their creativity, their beauty, or even (in a sense) because of their purpose. It copied them because programmers had already learned to work with the Sun Java API's system..." [*Op. cit* at 33]. To this author, this is simply another way of saying that Java's Declaring Code was merely a "system" or "method of operation"; it was thus unnecessary to decide the non-issue of whether Google's copying was "transformative."

To be clear, your author believes the result was proper, though the means to that end were needlessly foggy. While many of Thomas' criticisms of Breyer's decision have obvious merit, Thomas himself turns a blind eye to 17 U.S.C. §102(b) when he posits that "The Copyright Act expressly protects computer code. It recognizes that a "computer program" is protected by copyright" [*Op. cit* at 47] but then only cites to 17 U. S. C. §§109(b), 117, and 506(a). While Thomas excoriates the majority for failing to address the core question, his glossing over the issue seems no better.

In the final analysis, this is the kind of decision that gives courts [and lawyers] a bad name. Pages of dicey legal arguments were constructed that could and should have been avoided. Litigants, juries and lower courts were dragged needlessly through a process that consumed untold hours and dollars and was at times procedurally absurd. Future litigation is assured, and that is a shame.■

Phillip Van Ness is of counsel to the Urbana law firm of Webber & Thies, P.C., <PVanness@WebberThies.com>