IRWIN IP
INTELLECTUAL PROPERTY LITIGATION

IP CASE OF THE WEEK

# Fair Use Ping-Pong

*Google LLC v. Oracle America, Inc.*, No. 18-596, 2021 WL 124090 (Apr. 5, 2021)

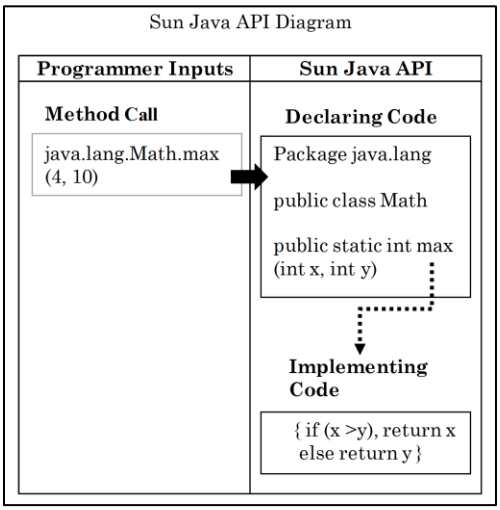
By: Ifti Zaim and Peter Danos | July 14, 2021

In the fiercely contested case of *Google v. Oracle*, the Supreme Court held that Google's copying of 11,500 lines of code from Sun Microsystems' Java Application Programming Interface ("API") to allow software developers to leverage their accrued Java coding skills on Android constituted fair use as a matter of law. Case No. 18-956 (Apr. 5, 2021). The decision underscores the importance of considering the nature of the copyrighted work, its real-world applications, and the constitutionally mandated purpose of the copyright laws (*i.e.* not to reward the labor of authors, but to promote the progress of science and the useful arts).

The decision came on the heels of the Federal Circuit (CAFC)'s 2018 reversal as a matter of law of the District Court's finding and jury's verdict of fair use, where the CAFC unemphatically stated: "there is nothing fair about taking a copyrighted work verbatim and using it for the same purpose and function as the original in a competing platform." The CAFC found only the second factor (the nature of the work) to favor fair use but disregarded it as typically "relatively unimportant." Yet the Supreme Court held that all four factors favored fair use. So, how did two panels arrive with such conviction at opposite conclusions? Context.

The Java API streamlines development by "allow[ing] programmers to use … prewritten code to build certain functions into their own programs, rather than write their own code to perform those functions from scratch." This prewritten code, known as "implementing code," lets a programmer simply name the function they want performed and fill in a few blanks with parameters the function will need. When the program runs the system will automatically pull in the appropriate unit of implementing code (known as a "method"), which contains the specific computer instructions needed to perform the function on that user's operating system and hardware.

In Java, methods are each contained within a "class." Similar classes are grouped into and made available by the API in "packages." This nested organizational structure is known as Java's Structure, Sequence, and Organization ("SSO"). Java's ***"declaring code"*** defines the names of these methods, how those methods are organized into classes and packages, the input parameters each method receives, and the outputs it returns. A programmer usually "calls" a method to action by spelling out its full path, which generally looks like: ***[package].[class].[method](parameters)***. Programmers learn this SSO and the associated commands with experience coding in Java. The diagram appended to the Court's decision illustrates these units of code and how they work:



As proved important to the Court's fair use analysis, Google did not copy Java's implementing code; only the declaring code for the 37 Java packages most important for the mobile device environment. Starting with the **second fair use factor** (the nature of the work), the Court found the copied material to be part of the Java API's "user interface," which, "if copyrightable at all, [is] further than are most computer programs (such as the implementing code) from the core of copyright," and that "its use is inherently bound together with uncopyrightable ideas (general task division and organization) and new creative expression (Android's implementing code)." Further much of the code's value derived from ***effort third-party software developers invested to learn Java's SSO.*** The Court not only gave weight to this second fair use factor but used it to color its analysis of every other factor.

On the **first factor,** the purpose and character of the use, the Court found Google's use—a "reimplementation," *i.e*, a repurposing of the words and syntaxes of the Java language—transformative because it expanded the use and usefulness of Android smartphones" (versus desktop and laptop computers). The Court again reinforced that Google sought to leverage programmers' acquired skills in the Java language, stating: "[t]o the extent Google used parts of the Sun Java API to create a new platform that could be readily used by programmers, its use was consistent with that creative 'progress' that is the basic constitutional objective of copyright itself."

The Court also found the **third fair use factor,** the amount and substantiality of the work used, to favor fair use. Notwithstanding the 11,500 lines of code copied the Court noted the millions of lines of implementing code that Google did not copy, instead correlating the declaring code with its own implementing code tailored for the mobile environment. And again, the Court emphasized that the specific portion copied was important not because of its "beauty" or even "purpose," but because it, again, enabled programmers to employ their accrued skills.

On the **fourth fair use factor,** market effects, the Court weighed the amount and type of harm against the public benefits of the copying. The Court held that Google's monetary gains were not cognizable market losses under the Copyright Act because that value flowed from the efforts of third-party programmers who had learned to use Java, and "correspondingly [had] less to do with Sun's investment in creating the Sun Java API." Finally, the Court held that enforcing Oracle's copyright "would risk harm to the public" because it would turn the declaring code into "a lock limiting the future creativity of new programs," running counter to "copyright's basic creativity objectives." Thus, the Court found the fourth factor also favored fair use.

Two jury trials, two appellate reversals, and a trip to the Supreme Court later, thus ends of one of the most closely watched rallies in at least recent fair use, copyright, and intellectual property law history. It closed with a whisper rather than a bang as the Court maintained the industry *status quo*. But, in some mirror world, our mirror selves are weathering the aftershocks of a copyright monopoly over the dictionary (not just the definitions, but the words and grammar too) of one of the most ubiquitous programming languages on Earth. The difference is context.