Orrin G. Hatch
Foundation

HATCH CENTER REPORT

GOOGLE v. ORACLE *and the* FUTURE OF COPYRIGHT

Policy Report

Christopher Bates

April 2021

It was billed as the "copyright case of the decade"[1]—the case that would decide the scope of software copyright and whether certain types of computer code, essential to the operation of the Android smartphone platform, were covered by copyright law. At stake were billions of dollars, the future of the smartphone industry, and the ability of developers to require licenses for the use of computer code that has become so widespread that it is arguably essential for future software development. Even the name of the case, *Google LLC v. Oracle America, Inc.*,[2] indicated it was no run-of-the-mill dispute. In the end, the Supreme Court sidestepped what many viewed as the pivotal question in the case—whether such types of essential computer code can be copyrighted in the first place—and instead ruled that, even if such types of code are copyrightable, Google's use of the code to build out its Android platform was not an infringement of copyright because it was a "fair use."[3] Under the fair use doctrine, an individual or entity is permitted to make limited use of copyrighted material without seeking permission from the copyright holder if the individual or entity is able to satisfy a four-factor balancing test.

Because the Court sidestepped the question of copyrightability, the scope of copyright protection in software following the decision remains unclear. The practical consequences of the Court's decision, however, could run deep. Because the Court found Google's copying of thousands of lines of Oracle's code for commercial purposes to constitute fair use, it would appear that software programmers who wish to use lines of code that are essential to platform operability (and program interoperability) have relatively free rein to do so—provided they copy no more than is necessary to create their new program. The decision also has the potential to impact the fair use landscape more broadly. Although the Court took care to specify the limited nature of its decision—emphasizing that, unlike many other types of copyrightable works, computer programs are "primarily functional" and that the Court was not "overturn[ing] or modify[ing] our earlier cases involving fair use"[4]—certain passages in the Court's opinion, if applied to other contexts, could make fair use findings more likely. The ultimate impact of the decision beyond the software industry will, of course, be fought out in the lower courts.

This paper provides an overview of the case, the Supreme Court's decision, and the potential impact of the decision going forward.

Background

In 2005, Google acquired a startup called Android, Inc. with the intent of developing a software platform for smartphone devices.[5] At the time, many software developers wrote programs using a programming language called Java, which had been created by Sun Microsystems (a company later acquired by Oracle).[6] These programmers used a platform called Java SE to create new programs primarily for use on desktop and laptop computers. By using Java SE, programmers could write programs that were able to run on any desktop or laptop regardless of underlying hardware, thus rendering the programs "interoperable" across different devices and with other Java programs.[7]

Google wanted to use Java SE to create a new software platform—Android—for smartphone devices. But it did not want to require that all programs written for Android be interoperable with other Java programs and across different devices. Google entered into negotiations with Sun for a license to use Java SE to build out the Android platform, but negotiations broke down over Sun's insistence that programs created for Android be interoperable.[8]

Google then proceeded to build out the Android platform. In the process, Google programmers copied approximately 11,500 lines of code from Java SE.[9] These lines were part of what's called an Application Programming Interface, or API, which is a tool that allows programmers to use prewritten code to carry out certain functions rather than having to "write their own code to perform those functions from scratch."[10] As the Supreme Court explained in its decision, "[i]t would be difficult, perhaps impossible, for a programmer to create complex software programs without drawing on prewritten task-implementing programs to execute discrete tasks."[11]

As relevant here, an API contains two types of code. First is "declaring code."[12] This type of code is used to identify certain prewritten tasks for the computer to complete and to direct the computer to complete those tasks.[13] One such task—discussed in both the Supreme Court's opinion and in the Federal Circuit's decision below[14]—might be determining which of two integers is larger. The second type of code is "implementing code."[15] This type of code instructs the computer how to carry out the identified task.[16] To continue the above example, the declaring code would instruct the computer to determine which of two integers is larger, while the implementing code would instruct the computer how to do so. The implementing code for a given task may be hundreds of lines long.[17]

In addition to instructing the computer to carry out prewritten tasks, declaring code serves two important functions. First, it provides a series of shortcuts for programmers.[18] Rather than having to memorize millions of lines of implementing code for all sorts of different tasks, programmers need to learn only the relevant declaring code to call up the desired task. Second, it provides a scheme for organizing different sets of tasks.[19] To use the example of the Sun Java API, each individual task is known as a "method," similar methods are grouped together in "classes," and similar classes are grouped together in "packages."[20] The declaring code links together a word or phrase that identifies the relevant package, with another word or phrase that identifies the relevant class, and a final word or phrase that identifies the relevant method.[21] The full expression (package + class + method), known as a "method call," directs the computer to carry out the desired task by using the prewritten implementing code that instructs the computer how to carry out the task.[22] The method call for determining the larger of two integers in the Sun Java API is "java.lang.Math.max(X, Y)," with X and Y being the two integers.[23] "Java.lang" identifies the package, "Math" identifies the class, and "max" identifies the method. If a programmer writes out this method call, the Sun Java API will direct the computer to identify the larger of X and Y by executing the prewritten implementing code.[24]

In building out the Android platform, Google copied the declaring code for 37 packages from the Sun Java API.[25] As noted above, this constituted approximately 11,500 lines of code, out of a total of 2.86 million lines of code in the full Sun Java API.[26] For the other 131 API packages in the Android platform,[27] Google wrote its own declaring code.[28] Google did not copy any implementing code from the Sun Java API. Rather, Google wrote its own implementing code.[29] Altogether, Google wrote millions of lines of new code for the Android platform.

According to Google, the declaring code it copied from the Sun Java API was necessary for its programmers to be able to "use the 'task calling' system that they had already learned" through their prior experience with Java.[30] Three of the packages were "fundamental to being able to use the Java language at all."[31] Without the packages, Google's programmers would have "need[ed] to learn an entirely new system to call up the same tasks."[32] Google launched its Android platform in 2007 to great success. By 2015, sales of Android totaled more than $40 billion in revenue.[33]

Oracle acquired Sun in 2010. Later that year, Oracle filed suit against Google alleging both copyright and patent infringement based on Google's use of the Sun Java API.[34] Following a trial, a jury rejected Oracle's patent claims but found that Google had infringed Oracle's copyrights.[35] The jury deadlocked on whether Google's actions constituted fair use. The trial judge then ruled that the declaring code Google had copied was not copyrightable and entered judgment for Google.[36] On appeal, the Federal Circuit reversed, holding that the declaring code was copyrightable, and remanded for a trial on whether Google's copying was a fair use.[37] Following a second trial, a jury determined it was.[38] On appeal, the Federal Circuit again reversed, writing that "[t]here 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."[39]

The Supreme Court granted Google's petition for certiorari as to both copyrightability and fair use. The matter was thus teed up for the Court to decide whether Google could copy—without a license—lines of code from the Sun Java API that were necessary for Google's programmers to be able to use the "task-calling" system they already knew, rather than pay a license or (alternatively) create an entirely new system that programmers would need to learn from scratch. Given the popularity of the Android platform and the widespread use of the Sun Java API among programmers, the outcome of the case promised to have a significant impact on the future of the smartphone market and the ability of programmers to use lines of code that have become so prevalent as to become practically essential for future software development without paying a license to the creator of the code.

"

At stake were billions of dollars, the future of the smartphone industry, and the ability of developers to require licenses for the use of computer code that has become so widespread that it is arguably essential for future software development.

The Decision

The Supreme Court issued its decision on April 5, 2021. In an opinion by Justice Breyer,[40] the Court reversed the Federal Circuit on the ground that Google's copying of the 37 Sun Java API packages was a fair use and therefore not an infringement of copyright. The Court expressly declined to opine on whether the API was copyrightable in the first instance. Rather, the Court assumed "purely for argument's sake" that the API *was* copyrightable and then proceeded to the question of whether Google's copying of the API was a fair use.[41]

Under the Copyright Act, "the fair use of a copyrighted work . . . for purposes such as criticism, comment, news reporting, teaching . . . , scholarship, or research, is not an infringement of copyright."[42] The Act sets forth four factors for courts to consider in determining "whether the use made of a work in any particular case is a fair use":

> (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.[43]

This list of factors is not exhaustive, nor is the list of purposes (criticism, comment, news reporting, etc.) set forth in the provision.[44] Rather, the fair use statute "set[s] forth general principles, the application of which requires judicial balancing, depending upon relevant circumstances."[45]

Before delving into its analysis of the four fair use factors, the Court offered some comments about the origin and purposes of the fair use doctrine, particularly as it applies to computer programs. Describing fair use as a "flexible" concept that must be applied "in light of the sometimes conflicting aims of copyright law," the Court emphasized that computer programs differ from many other types of copyrightable works in that they "almost always serve functional purposes."[46] Because copyright law typically provides "stronger" protection where copyrighted material "serves an artistic rather than a utilitarian function," these and other differences "have led at least some judges to complain that 'applying copyright law to computer programs is like assembling a jigsaw puzzle whose pieces do not quite fit.'"[47] Accordingly, "fair use can play an important role in determining the lawful scope of a computer program copyright," including by "distinguish[ing] between expressive and functional features of computer code" and avoiding the creation of "unrelated or illegitimate harms in other markets or to the development of other products."[48] The majority thus offered a flexible, consequentialist account of fair use that ties application of the doctrine to the purposes of the copyrighted work and the predicted impact of enforcing the copyright on the development and availability of future works.

With that background, the Court then turned to the four fair use factors, concluding that each factor weighed in favor of fair use.[49] Interestingly, the Court began with the *second* factor, the "nature of the copyrighted work"[50]—a decision that Justice Thomas criticized in dissent as seeking to create an unfounded "distinction between declaring code and implementing code that renders the former less worthy of protection than the latter."[51] According to the majority, the declaring code that Google copied is "further than are most computer programs (such as the implementing code) from the core of copyright," because declaring code is "inherently bound together with uncopyrightable ideas" (such as task division and organization) and "new creative expression" (i.e., the implementing code, which tells the computer *how* to carry out the tasks).[52] In addition, unlike other computer programs, the "value" of the copied code derives, not so much from the code itself, but from the "time and effort" programmers invest in learning the Sun Java API system and the way in which the code "encourage[s] programmers to learn and to use that system" so that they will be able to use *other* Java-based implementing programs.[53] In the Court's view, these features, which differentiate the API packages Google copied from the "mine run of other computer programs," meant that the nature of the copyrighted work pointed in favor of fair use.[54]

The Court then turned to the first fair use factor, "the purpose and character of the use"[55]—the factor that typically leads off fair use analysis. Here, the Court asked whether Google's use of the declaring code was "transformative," that is, whether it added "something new and important."[56] The Court acknowledged that Google copied the code "for the same reason that Sun created [the copied] portions"—to "enable programmers to call up implementing programs that would accomplish particular tasks."[57] But that did not end the inquiry, the Court said, because Google copied the code "to create new products."[58] In particular, Google used the copied code to "expand the use and usefulness of Android-based smartphones" and to "create a new platform that could be readily used by programmers."[59] The Court additionally emphasized that Google copied Oracle's API packages "only insofar as needed" to allow its programmers to call up tasks useful in smartphone programs without having to learn a new programming language.[60] That Google copied Oracle's code for a commercial reason did not alter the analysis. Even if a commercial motive may, in some cases, weigh against a finding of fair use, it was not "dispositive" of the first factor, "particularly in light of the inherently transformative role that the reimplementation [of Oracle's code] played in the new Android system."[61] Thus, the purpose and character of the use pointed in favor of fair use.

As to the third fair use factor, "the amount and substantiality of the portion used,"[62] the Court reasoned that the approximately 11,500 lines of code Google copied—a number that, viewed in isolation, seemed quite large—needed to be considered in light of "the several million lines of code that Google did *not* copy."[63] This was because the lines of code Google copied, out of the millions of lines it could have copied, were the ones it needed to accomplish its "valid, and transformative, purpose."[64] The Court thus linked the third fair use factor back to the first factor: because the copying was done for a transformative purpose, the *amount* of copying also weighed in favor of fair use. The majority rejected the view that Google could have accomplished its purpose by copying substantially fewer lines of code—namely, the 170 lines of code "necessary to write in the Java language"—because Google's purpose was not simply to make Java usable on the Android platform, but to enable its programmers to "make use of their *knowledge and experience* using the Sun Java API" when writing programs for Android.[65] Accordingly, the third factor weighed in favor of fair use.

The Court devoted the most extended discussion to the fourth fair use factor, "the effect of the use upon the potential market" for the copyrighted work.[66] "[W]here computer programs are at issue," the Court said, this factor "can prove more complex than at first it may seem."[67] A reviewing court, of course, should "consider the amount of money that the copyright owner might lose" because of the unauthorized use, but it must also take into account both "the source of the loss" and "the public benefits the copying will likely produce."[68] As to the likely amount of loss from Google's copying, the Court noted that evidence at trial indicated that, "regardless of Android's smartphone technology, Sun was poorly positioned to succeed in the mobile phone market."[69] Java SE's "primary market" was laptop and desktop computers, and Sun's prior efforts to enter the mobile phone market had been "unsuccessful."[70]

"

Given its view that all four factors pointed toward fair use, the majority had little trouble concluding that Google's copying was a fair use.

As to the source of the loss, the Court reasoned that requiring Google to compensate Oracle for the use of the copied code would reward Oracle for the efforts of third parties—computer programmers who had learned to use the Sun Java API—rather than Oracle's own original "investment" in creating the API.[71] According to the Court, the record demonstrated that Google wanted to use the Sun Java API, not necessarily because it was better than other API alternatives as a qualitative matter, but rather because programmers were "just used to it."[72] Furthermore, the ultimate success of the Android platform—and the significant revenue Android generated—had less to do with Sun's investment in creating the Sun Java API in the first instance, and far more to do with the efforts of Google's programmers in using the API to build out a successful platform.

Finally, as to the question of public benefits, the Court reasoned that allowing Oracle to enforce its copyright in the copied code (assuming, again, that the code was copyrightable) "would risk harm to the public."[73] According to the Court, "allowing enforcement here would make of the Sun Java API's declaring code a lock limiting the future creativity of new programs," and "Oracle alone would hold the key."[74] Such a result would run the risk of hindering the creation of "improvements, new applications, and new uses developed by users who have learned to work with [the Sun Java] interface."[75] These concerns, combined with the "uncertain" nature of Sun's ability to enter the smartphone marketplace and the "sources" of Oracle's lost revenue (value attributable to programmers, not Oracle's own investment in the creation of the Sun Java API) meant that the fourth factor likewise pointed toward fair use.[76]

Given its view that all four factors pointed toward fair use, the majority had little trouble concluding that Google's copying was a fair use. Before closing, the Court emphasized once more that "[t]he fact that computer programs are primarily functional makes it difficult to apply traditional copyright concepts."[77] The Court then explained that in ruling that Google's copying was a fair use, it was not "chang[ing] the nature of those concepts" or "overturn[ing] or modify[ing] our earlier cases involving fair use."[78] Rather, it was merely applying those concepts to a "different kind of copyrighted work."[79]

The Dissent

Justice Thomas dissented, joined by Justice Alito. Unlike the majority, Justice Thomas addressed both copyrightability and fair use, concluding that the copied declaring code was copyrightable and that Google's copying was not a fair use. Justice Thomas chastised the majority for "skipping over the copyrightability question," arguing that in doing so, the majority "disregard[ed] half the relevant statutory text"—the portion that defines what types of works receive protection—and "distort[ed] its fair-use analysis."[80]

Starting with the question of whether the copied API packages were copyrightable, Justice Thomas concluded they were because, under the Copyright Act, they are "original works of authorship fixed in a[] tangible medium of expression."[81] They are "original" because Sun could have created the copied packages "any number of ways," and they are "fixed" in the form of "words, numbers, or other verbal or numerical symbols."[82] Justice Thomas rejected Google's argument that the packages (and associated declaring code) were uncopyrightable "methods of operation," because the packages constituted the "specific expression" Oracle had chosen to instruct computers using the Sun Java API to identify and carry out certain functions.[83] As Justice Thomas explained, although "Oracle cannot copyright the idea of using declaring code" to instruct computers to carry out prewritten tasks, "it can copyright the *specific expression* of that idea found in its library."[84]

Turning to fair use, Justice Thomas characterized the majority's application of the doctrine as "far from ordinary."[85] In his view, in addition to analyzing the four fair use factors out of order, the majority had created a "categorical distinction[]" between declaring code and implementing code that the Copyright Act does not permit, because both types of code are copyrightable and neither serves any function without the other.[86] "The result of this distorting analysis," Justice Thomas said, "is an opinion that makes it difficult to imagine any circumstance in which declaring code will remain protected by copyright."[87]

Turning to the first fair use factor the majority considered—the "nature of the copyrighted work"[88]—Justice Thomas acknowledged that this factor will often favor fair use when computer code is involved because computer code is "predominantly functional."[89] But he then argued that the reasons the Court gave for why Oracle's declaring code is "far 'from the core of copyright'" apply equally to implementing code, "which the majority concedes is copyrightable."[90] Like declaring code, implementing code is "inherently bound up with" uncopyrightable ideas—the "division of computing tasks."[91] And like declaring code, the "value" of implementing code is "directly proportional to how much programmers value the associated declaring code," because declaring code is how programmers "access" implementing code in the first place.[92] Moreover, virtually *every* type of copyrightable work is "inherently bound up with" uncopyrightable ideas (e.g., books are inherently bound up with the ideas of having a plot, using chapters, and including dialogue), and the value of many other works is similarly impacted by "how much time third parties invest in learning" the work (e.g., the value of a script depends on the investment of time by actors in learning the script).[93] So the second fair use factor, even if it weighed in favor of fair use, could not do the work the majority sought of rendering declaring code "generally unworthy of protection."[94]

Justice Thomas then turned to the fourth fair use factor, "the effect of the use upon the potential market" for the copyrighted work.[95] Noting that this factor is typically considered the "single most important element" of the fair use analysis, Justice Thomas argued that Google's copying of the Sun Java API "ruined Oracle's potential market in at least two ways."[96] First, it "eliminated the reason [device] manufacturers were willing to pay to install the Java platform."[97] Because Google's Android platform—which Google offered for free (because Google had a different business model than Oracle that depended on advertising rather than licensing revenue)—"included much of Oracle's code," device manufacturers no longer had a reason to pay to embed the Java platform.[98] Second, Google's copying interfered with Oracle's efforts to license Java to other developers, because they could simply use Google's free Android platform. Unsurprisingly, after Google incorporated the Sun Java API into its Android platform and then released Android for free, the value of Oracle's contracts with mobile device manufacturers dropped by nearly 98 percent.[99]

Justice Thomas also pushed back on the majority's claim that allowing Oracle to enforce its copyrights against Google would "harm the public" by giving Oracle the ability to "'limit the future creativity' of programs on Android."[100] To begin, less than 8 percent of active Android devices still run the versions of Android at issue in the suit.[101] Next, Oracle never had a "lock[]" over future software development in the first place.[102] Apple and Microsoft both created mobile operating systems without using Oracle's declaring code. Oracle also always made its declaring code "freely available to programmers," provided they agreed to make their programs interoperable.[103] Finally, even if Oracle were permitted to enforce its copyrights in the Sun Java API against Google, that wouldn't necessarily give Oracle "control" over Android.[104] Rather, the result might be a damages award, or an award of perpetual royalties. For good measure, Justice Thomas also pointed out that in the years since Google had copied Oracle's declaring code and released Android, it was *Google* that had sought to "monopoliz[e]" the smartphone market and it is *Google* that now controls "the most widely used mobile operating system in the world."[105] "If the majority is worried about monopolization," Justice Thomas wryly noted, "it ought to consider whether Google is the greater threat."[106] Accordingly, in Justice Thomas's view, the effect of Google's copying on the market for Oracle's work weighed heavily against a finding of fair use.

As to the "purpose and character of the use,"[107] which Justice Thomas described as the "second-most important" fair use factor, Justice Thomas first noted that Google's copying of Oracle's code was "overwhelming[ly] commercial" in nature.[108] In 2015 alone, Google earned $18 billion from Android.[109] Even if the fact that a use is commercial "does 'not necessarily' weigh against fair use," Justice Thomas said, "we have never found fair use for copying that reaches into the tens of billions of dollars and wrecks the copyright holder's market."[110] Nor was Google's use of the copied API packages, in Justice Thomas's view, transformative. Google did not "use Oracle's code to teach or reverse engineer a system to ensure compatibility."[111] Rather, it "used the declaring code for the same exact purpose Oracle did."[112]

Justice Thomas was particularly unpersuaded by the majority's argument that Google's use was transformative because it "help[ed] others 'create new products.'"[113] "That new definition," he said, "eviscerates copyright."[114] A movie studio that converts a book into a film both creates a new product (the film) and enables others to create other new products (merchandise, DVDs, highlight reels, etc.). Indeed, "[n]early every computer program," if copied, "can be used to create new products."[115] The majority's fundamental error, Justice Thomas said, was conflating transformative use with derivative use: "To be transformative, a work must do something fundamentally different from the original. A work that simply serves the same purpose in a new context—which the majority concedes is true here—is derivative, not transformative."[116] Accordingly, the purpose and character of the use weighed against fair use.

Justice Thomas last addressed "the amount and substantiality of the portion used,"[117] arguing that this factor, too, weighed against fair use. This was because Google "copied the heart or focal points of Oracle's work."[118] It was the declaring code, after all, that "attracted programmers to the Java platform and why Google was so interested in that code."[119] That Google may have taken "no more" of the code "than necessary to create new products" did not alter the analysis for Justice Thomas, because in his view, Google's use was not transformative to begin with.[120]

"

Unlike the majority, Justice Thomas addressed both copyrightability and fair use, concluding that the copied declaring code was copyrightable and that Google's copying was not a fair use.

Taken together, then, three of the fair use factors "weigh[ed] decidedly" against fair use for Justice Thomas.[121] And the "sole factor possibly favoring Google"—the nature of the copyrighted work—"cannot by itself support a determination of fair use because holding otherwise would improperly override Congress' determination that declaring code is copyrightable."[122] Thus, in Justice Thomas's view, Google's copying of the Sun Java API was not a fair use.

The Impact

The most obvious (and immediate) impact of the Court's decision is that Google will not have to pay damages or seek a license from Oracle for its use of the copied API packages. This is no minor consequence, as Oracle had sought roughly $9 *billion* in damages from Google.[123] The Court's decision also likely means that other companies that want to use declaring code from the Sun Java API to create new programs can do so without a license, provided they copy "only what [is] needed" to enable their programmers to make use of their preexisting knowledge of and familiarity with the API.[124] Enforcing copyrights in declaring code more generally may also prove challenging, given the Court's statements that declaring code is "further than are most computer programs . . . from the core of copyright" and that "unlike many other programs," declaring code's value derives "in significant part" from the time and effort third parties—programmers—invest in learning the code.[125] These statements would seem to apply generally to all declaring code, not just the particular code at issue in this case. Thus, as Justice Thomas argued in dissent, it may be "difficult to imagine any circumstance" under the Court's opinion in which declaring code would receive protection.[126]

As to the decision's impact on computer code (and software) more broadly, different lines in the Court's opinion point in different directions. On the one hand, the opinion repeatedly emphasizes declaring code's "functional" nature and the fact that works with "utilitarian" (as opposed to artistic) uses tend to receive less copyright protection.[127] These characteristics are true of most computer code. So, too, is the ability of programmers to "repurpose[]" code for new uses and new programs.[128] Indeed, the entire reason for copying code in the first place is to build on previous programmers' work rather than having to reinvent the wheel when writing new programs. And, of course, requiring developers to obtain a license to use code written by someone else could give the original author of the code a "lock limiting . . . new applications[] and new uses" of the code.[129] All of these characteristics, which the Court relied upon in concluding that Google's copying was a fair use, would seem to apply broadly to all (or most) types of computer code.

On the other hand, however, the majority also took care throughout its opinion to emphasize the ways in which declaring code *differs* from other types of code. Indeed, the differences between declaring code and implementing code played a key role in the Court's analysis of why the second fair use factor—the nature of the copyrighted work—weighed in favor of fair use. Whereas implementing code is "innovative" and requires balancing competing considerations like processing speed, battery life, and memory size, declaring code is "user-centered" and focuses on attracting programmers who like the code's functionality and will want to use it for future work.[130] Thus, "[u]nlike many other programs" (including implementing code), declaring code's "value" as part of an API lies in the way it "encourage[s] programmers to learn and to use" the API to create future works.[131] The majority also emphasized that declaring code is "inherently bound together with" the "uncopyrightable ideas" (such as task division and organization) it embodies.[132] Indeed, separating declaring code from the organizing and task-identifying functions it plays is likely impossible. Computer programs that have value separate and apart from their use in future programming or that are more closely linked to creative or expressive activity as a standalone matter would seem to fall outside this reasoning.

A close reading of the Court's explanation for why Google's use of the copied declaring code was "transformative" also suggests limits on the Court's holding as applied to other types of computer programs. Although the Court placed great emphasis on the fact that Google used the copied portions of the Sun Java API to "create new products," that was not the end of the analysis.[133] The "new product" Google created was not just any old product (or program). It was a "new *platform* that could be readily used by programmers" to "expand the use and usefulness of Android-based smartphones."[134] At the outset of its opinion, the Court described a "software platform" as a sort of "factory floor where computer programmers . . . might come, use a set of tools found there, and create new applications for use in, say, smartphones."[135] Thus, the "new product" Google used the copied code to create was a program (a virtual "factory floor") that would enable programmers to create *additional* programs to "expand the use and usefulness" of Android devices.[136] Indeed, not only would the new program enable programmers to create further new programs, but that was the entire point of the new program in the first place. It seems fair to think that many, if not most, "new" computer programs would fall outside this line of reasoning. Even if, say, a software installation program or a search program could be repurposed or incorporated into a "new" computer program, the *purpose* of the installation or search program is not to enable the creation of new programs. Nor would the installation or search program be akin to a "factory floor" enabling programmers to let their creativity run free. Rather, the two programs serve a much more limited purpose. Thus, there would be a strong argument that copying an existing program (or portion of a program) to create the installation or search program would not count as a "transformative" use under the majority's reasoning.

A third and final distinction between declaring code and other types of code (or programs) that was clearly relevant to the Court's holding concerns the manner in which declaring code can act as a necessary (or highly useful) input for future software development. As the Court explained, by copying Oracle's API packages, Google enabled its programmers to use their knowledge of the Sun Java API's task-calling system to build out the Android platform without having to "learn an entirely new system" for calling up tasks.[137] This sort of repurposing (or "reimplementation") of existing code "is necessary if programmers are to be able to use their acquired skills," and can be "necessary for different programs to speak to each other."[138] Thus, key to the Court's analysis was its understanding that, without the ability to use Oracle's declaring code, Google's programmers would have been unable to employ a significant portion of the knowledge and skills they had acquired through their prior years of programming experience. Under this reasoning, it's not enough merely to say that copying code someone else wrote will prevent inconvenience or avoid having to redo that person's work. Rather, the copying must be "necessary" in order to apply one's own experience or skills in the first place.[139] Once again, it seems fair to think that this sort of scenario is unlikely to apply to many, if not most, computer programs. For all these reasons, the impact of the Court's decision on copyright protection for other types of computer programs not at issue in the case may ultimately be somewhat limited, although there is plenty in the Court's opinion for enterprising lawyers to mine.

Turning, finally, to the decision's impact beyond the software industry—that is, to its impact on copyright protection more generally—here again, there are passages in the Court's opinion that can give heart (or heartburn) to both creators and copiers. The first point to highlight is the Court's repeated emphasis on the differences between computer programs and other types of copyrighted material. Toward the beginning of its fair use analysis, the Court stressed that computer programs "differ from books, films, and many other 'literary works' in that such programs almost always serve functional purposes."[140] Similarly, near the end of its opinion, the Court underscored that the "functional" nature of computer programs "makes it difficult to apply traditional copyright concepts in that technological world" and that the Court had done its best to apply existing fair use principles "to this different type of copyrighted work."[141] The Court also stated explicitly that its decision did not "change[] the nature of" traditional copyright concepts or "overturn or modify our earlier cases involving fair use."[142] These and other statements may properly be read to indicate that the Court understood that it was dealing with a relatively unique—and specialized—form of copyrighted work (computer code) and that the Court did not intend its decision to break new ground beyond that particular type of work.

There are, however, certain elements of the Court's analysis that, if applied broadly, could make fair use findings more likely. First is the Court's reasoning that Google's use of the copied code was "transformative" because Google used it "to create new products."[143] As Justice Thomas rightly noted in his dissent, "[t]hat new definition eviscerates copyright."[144] Applied broadly, this reasoning could mean that *any* use of copyrighted material to create a "new product"—such as a film or television show based on a book—could constitute fair use. Given the destabilizing effect such a broad reading could have on copyright protection for other types of works, coupled with the Court's statement that it was not "overturn[ing] or modify[ing]" its prior decisions on fair use,[145] the Court's

"

Turning, finally, to the decision's impact beyond the software industry—that is, to its impact on copyright protection more generally—here again, there are passages in the Court's opinion that can give heart (or heartburn) to both creators and copiers.

reasoning on this point is thus probably best understood as—in Justice Thomas's words—a "good-for-declaring-code-only precedent."[146]

The Court's analysis of the fourth fair use factor, the effect of the copying on the potential market for the copyrighted work, was also somewhat puzzling. The Court placed significant emphasis on the facts that—prior to Google's copying—Sun's efforts to enter the mobile phone market had been "unsuccessful" and Sun faced "business challenges in developing a mobile phone product."[147] But as Justice Thomas pointed out in his dissent, Sun had existing contracts with a range of mobile manufacturers and developers, and the value of those contracts fell precipitously after Android was released.[148] The majority also appeared to disregard the potential revenue Sun (later Oracle) might have made through *licensing* the Sun Java API for smartphone devices.[149] Even if, despite its best efforts, Sun (or Oracle) ultimately had not been able to enter the smartphone market directly, it still might have made significant money through licensing its API for smartphone uses. Yet the Court seemed unconcerned with this significant loss of potential income—a lack of concern made all the more startling by the fact that, within eight years of launch, Android had already generated over $40 *billion* in revenue.[150] Such a strict approach to proof of market harm, which appears to disregard both seemingly obvious and historically undisputed losses, could prove problematic if applied to other settings.

The overriding emphasis that the majority appeared to place on the value of the Sun Java API that owes to the time and effort of third parties (i.e., programmers) could also prove meddlesome if applied outside the software context. The Court acknowledged that "Google's copying [of the Sun Java API] helped Google make a vast amount of money from its Android platform."[151] But then it turned right around and downplayed the value of Sun's original "investment in creating the Sun Java API."[152] More relevant, in the Court's view, was the investment of "third parties[] (say, programmers[])" in learning to use and operate the API.[153] But of course the reason programmers chose to learn the Sun Java API in the first place was because of the API's popularity and "intuitive[]" design.[154] The use of the program was tied to its quality. The Court's apparent view that the value of a product owing to its popularity shouldn't factor into its value for purposes of market effects analysis (or at least shouldn't factor heavily) is, one might think, a bit strange. Nor is computer code the only type of copyrightable work whose value is affected by the actions of third parties. As Justice Thomas observed in his dissent, the value of a theater script depends on "actors and singers . . . invest[ing] time learning and rehearsing it."[155] The value of a book or television show, in turn, may be significantly impacted by the efforts of agents or advertisers in promoting the work to the public. Attempting to disaggregate the value of a creator's original "investment" in a copyrighted work from the subsequent popularity the work attains through the efforts of third parties seems a fraught exercise at best.

Finally, the Court's concern that allowing Oracle to enforce its copyrights in the Sun Java API could give Oracle a "lock limiting . . . future creativity"[156] would seem to apply *whenever* a copyright holder seeks to enforce its rights. Copyright enables a creator[157] to determine how and when a work is used. If a creator doesn't want his or her work incorporated into a new creative work, or doesn't want to allow derivative uses, copyright grants the creator that choice. Copyright also grants the creator (with certain exceptions) the ability to set the price and terms of permissible use. In this sense, copyright enforcement *always* gives the creator the ability to limit "future creativity" that depends on, or derives from, the original work. The Court's concern that enforcing Oracle's copyrights could limit creativity is thus a concern that could arise in many cases. Accordingly, the Court's concern must be understood in its proper context, in combination with all of the other factors the Court felt pointed in favor of fair use. Certainly it cannot be a standalone justification for finding fair use.

Conclusion

The meaning and import of the Supreme Court's decision in *Google v. Oracle* undoubtedly will be fought out in the lower courts for years to come. The decision's most likely impact—other than that Google won't have to pay damages to Oracle or seek a license for use of the Sun Java API—will be weakened protection for declaring code. What the decision portends for other types of computer code (or programs), or for copyright law more generally, remains to be seen. Certainly, there are ample statements in the Court's opinion that suggest courts should be cautious in applying it beyond the software context. Creative lawyers, however, will no doubt seek to apply the decision in other settings in ways that benefit their clients. The fight over fair use will continue, with *Google v. Oracle* merely the most recent signpost.

About the Author

Christopher Bates is a Legal Fellow at the Orrin G. Hatch Foundation. He also serves as a Visiting Scholar at Southern Virginia University and as Senior Vice President for Legal and Business Affairs at the National Music Publishers' Association (NMPA). Senator Hatch and NMPA filed amicus briefs in Google v. Oracle *in support of Oracle. Mr. Bates was not affiliated with the Hatch Foundation or NMPA at the time of filing and did not participate in the drafting or review of either brief. Any views expressed herein are his own.*

Endnotes

1 Chris Kemp, *United States:* Google v. Oracle*: The Copyright Case of the Decade*, Mondaq (May 20, 2020), https://www.mondaq.com/unitedstates/copyright/938300/google-v-oracle-the-copyright-case-of-the-decade.

2 No 18-956 (U.S. Apr. 5, 2021).

3 *Id.*, slip op. at 15.

4 *Id.* at 35.

5 *Id.* at 1-2.

6 *Id.* at 2.

7 *Id.* at 2-3.

8 *Id.* at 3.

9 *Id.*

10 *Id.* at 3-4 (quoting *Oracle Am., Inc. v. Google, Inc.*, 750 F.3d 1339, 1349 (Fed. Cir. 2014)) (internal quotation mark omitted).

11 *Id.* at 4.

12 *Id.* at 5.

13 *Id.*

14 *See id.* at 7-8, 38; *Oracle*, 750 F.3d at 1349-50.

15 *Google*, No. 18-956, slip op. at 4.

16 *Id.*

17 *Id.*

18 *See id.* at 5-6.

19 *See id.* at 6-8.

20 *Id.* at 4.

21 *See id.* at 7.

22 *Id.*

23 *Id.*

24 For the Android API, the implementing code for this method call is "{ if (x >y), return x else return y }." *Id.* at 38.

25 *Id.* at 8.

26 *See id.* at 28.

27 *See Oracle Am., Inc. v. Google, Inc.*, 750 F.3d 1339, 1350 (Fed. Cir. 2014) (stating that the Android platform "grew to include 168 API packages").

28 *Google*, No. 18-956, slip op. at 8.

29 *Id.*

30 *Id.*

31 *Id.* (quoting *Oracle Am., Inc. v. Google, Inc.*, 872 F. Supp. 2d 974, 982 (N.D. Cal. 2012)) (internal quotation mark omitted).

32 *Id.*

33 *Id.* at 9.

34 *Id.*

35 *Id.*

36 *Id.* at 9-10.

37 *Id.* at 10.

38 *Id.* at 10-11.

39 *Id.* at 11 (quoting *Oracle Am., Inc. v. Google LLC*, 886 F.3d 1179, 1210 (Fed. Cir. 2018)) (internal quotation marks omitted) (alteration in original).

40 Justice Breyer was joined by Chief Justice Roberts and Justices Sotomayor, Kagan, Gorsuch, and Kavanaugh. Justice Thomas wrote a dissenting opinion, *see infra* at 6-7, which Justice Alito joined. Justice Barrett, who was appointed to the Court after the case was argued, did not participate in the decision.

41 *Google*, No. 18-956, slip op. at 15.

42 17 U.S.C. § 107.

43 *Id.*

44 *See Google*, No. 18-956, slip op. at 14.

45 *Id.*

46 *Id.* at 15-16.

47 *Id.* (quoting *Lotus Dev. Corp. v. Borland Int'l, Inc.*, 49 F.3d 807, 820 (1st Cir. 1995) (Boudin, J., concurring)).

48 *Id.* at 16-17.

49 The Court also addressed what standard of review applies to a jury finding of fair use and whether reviewing such a finding *de novo* contravenes the Seventh Amendment's jury trial right and prohibition on reexamining facts found by a jury. *See id.* at 18-21. The Court ruled that because the "ultimate question" of whether a use was "fair" is a legal question, the proper standard of review is *de novo*, and applying such a standard does not violate the Seventh Amendment. *Id.* at 20-21.

50 17 U.S.C. § 107(2).

51 *Google*, No. 18-956, dissenting op. at 9.

52 *Id.*, majority op. at 24.

53 *Id.*

54 *Id.* at 23-24.

55 17 U.S.C. § 107(1).

56 *Google*, No. 18-956, slip op. at 24.

57 *Id.* at 25.

58 *Id.*

59 *Id.*

60 *Id.* at 26.

61 *Id.* at 27.

62 17 U.S.C. § 107(3).

63 *Google*, No. 18-956, slip op. at 29 (emphasis added).

64 *Id.*

65 *Id.* at 29-30 (quoting *Oracle Am., Inc. v. Google LLC*, 886 F.3d 1179, 1206 (Fed. Cir. 2018)) (internal quotation marks omitted) (emphasis added).

66 17 U.S.C. § 107(4).

67 *Google*, No. 18-956, slip op. at 30.

68 *Id.* at 30-31.

69 *Id.* at 31; *see also id.* at 33 (stating that the evidence Oracle submitted could not "overcome evidence indicating that, at a minimum, it would have been difficult for Sun to enter the smartphone market, even had Google not used portions of the Sun Java API").

70 *Id.* at 31.

71 *Id.* at 34.

72 *Id.*

73 *Id.*

74 *Id.*

75 *Id.*

76 *Id.* at 35.

77 *Id.*

78 *Id.*

79 *Id.*

80 *Id.*, dissenting op. at 1-2.

81 *Id.* at 5 (quoting 17 U.S.C. § 102(a)) (internal quotation mark omitted).

82 *Id.* (quoting 17 U.S.C. § 101) (internal quotation mark omitted).

83 *Id.* at 7.

84 *Id.* (emphasis added). Justice Thomas also rejected Google's argument that Oracle's declaring code was uncopyrightable under the "merger doctrine" because it was the "only way" to express the idea of using code to carry out prewritten tasks using the Sun Java API. *Id.* Even if there was "only one way for Google to *copy* the lines of declaring code," Justice Thomas said, "there were innumerable ways for Oracle to *write* them." *Id.* (emphasis added).

85 *Id.*

86 *Id.*; *see also id.* at 6 (arguing that "[t]he functionality of both declaring code and implementing code will . . . typically rise and fall together").

87 *Id.* at 7-8.

88 17 U.S.C. § 107(2).

89 *Google*, No. 18-956, dissenting op. at 9.

90 *Id.* at 9-10 (quoting majority op. at 24).

91 *Id.* at 10 (quoting majority op. at 22) (internal quotation marks omitted).

92 *Id.* at 11.

93 *Id.* at 10-11.

94 *Id.* at 11.

95 17 U.S.C. § 107(4).

96 *Google*, No. 18-956, dissenting op. at 11 (quoting *Harper & Row, Publ'rs, Inc. v. Nation Enters.*, 471 U. S. 539, 566 (1985)).

97 *Id.*

98 *Id.* at 12.

99 *See id.* (noting that after Google released Android, the value of Oracle's contracts with Amazon and Samsung dropped by 97.5 percent).

100 *Id.* at 13 (quoting majority op. at 34) (alteration omitted).

101 *Id.*

102 *Id.*

103 *Id.* at 14.

104 *Id.*

105 *Id.*

106 *Id.*

107 17 U.S.C. § 107(1).

108 *Google*, No. 18-956, dissenting op. at 15.

109 *Id.*

110 *Id.* at 15-16 (quoting majority op. at 27).

111 *Id.* at 16.

112 *Id.*

113 *Id.* (quoting majority op. at 25).

114 *Id.* at 17.

115 *Id.* "Because the majority's reasoning would undermine copyright protection for so many products long understood to be protected," Justice Thomas said, "I understand the majority's holding as a good-for-declaring-code-only precedent." *Id.* at 17 n.11.

116 *Id.* at 17.

117 17 U.S.C. § 107(3).

118 *Google*, No. 18-956, dissenting op. at 18.

119 *Id.*

120 *Id.* Justice Thomas also chided the majority for comparing the 11,500 lines Google copied against the 2.86 million lines of code in the full Java Sun API. "[T]he proper denominator," he said, "is *declaring code*, not all code." *Id.* (emphasis in original). The declaring code "is what attracted programmers" and "made Android a market substitute for potentially licensed derivatives of Oracle's Java platform." *Id.* (internal quotation marks omitted).

121 *Id.*

122 *Id.* at 18-19.

123 Richard Wolf, *Supreme Court Wrestles with Google-Oracle Copyright Battle that Could Upend Tech Industry*, USA Today (Oct. 7, 2020), https://www.usatoday.com/story/news/politics/2020/10/07/google-v-oracle-supreme-court-wrestles-over-9-billion-copyright-suit/5908012002.

124 *Google*, No. 18-956, slip op. at 35.

125 *Id.* at 24.

126 *Id.*, dissenting op. at 7-8.

127 *Id.*, majority op. at 15-16.

128 *Id.* at 26 (quoting Brief for R Street Institute et al. as Amici Curiae Supporting Petitioner).

129 *Id.* at 34.

130 *Id.* at 23.

131 *Id.* at 24.

132 *Id.*

133 *Id.* at 25.

134 *Id.* (emphasis added).

135 *Id.* at 2.

136 *Id.* at 25.

137 *Id.* at 8.

138 *Id.* at 26.

139 *Id.*

140 *Id.* at 16.

141 *Id.* at 35.

142 *Id.*

143 *Id.* at 25.

144 *Id.*, dissenting op. at 17.

145 *Id.*, majority op. at 35.

146 *Id.*, dissenting op. at 17 n.11.

147 *Id.*, majority op. at 31-32.

148 *Id.*, dissenting op. at 12.

149 *See id.*, majority op. at 33; *see also id.*, dissenting op. at 13 (noting that in assessing the market effects of copying for purposes of fair use, "[w]e look at not only the potential market 'that creators of original works in general develop' but also those potential markets the copyright holder might 'license others to develop'" (quoting *Campbell v. Acuff-Rose Music, Inc.*, 510 U.S. 569, 592 (1994))).

150 *Id.*, majority op. at 9.

151 *Id.* at 33.

152 *Id.* at 34.

153 *Id.*

154 *Id.* at 23.

155 *Id.*, dissenting op. at 11.

156 *Id.*, majority op. at 34.

157 For purposes of this paragraph, the "creator" is assumed to be the copyright holder. When the copyright holder is someone other than the creator (e.g., the heir of a deceased creator or an employer who contracted a work for hire), the copyright holder—not the creator—determines the permissible future creative uses of the work.

HATCH CENTER
CIVILITY & SOLUTIONS

(385) 355-4380 | info@orrinhatchfoundation.org

SLC, UT Office | 411 E. South Temple, Salt Lake City, Utah 84111

Washington, DC Office | 655 15th St. NW, Suite 800, Washington, DC 20005

@senatororrinhatch @orrinhatch @orrinhatchcenter

www.orrinhatchfoundation.org