No. 18-956

IN THE

# Supreme Court of the United States

GOOGLE LLC,

*Petitioner,*

v.

ORACLE AMERICA, INC,

*Respondent.*

On Writ of Certiorari
to the United States Court of Appeals
for the Federal Circuit

## BRIEF OF AMICUS CURIAE ENGINE ADVOCACY IN SUPPORT OF PETITIONER

Jef Pearlman
*Counsel of Record*
INTELLECTUAL PROPERTY & TECHNOLOGY LAW CLINIC
UNIVERSITY OF SOUTHERN CALIFORNIA GOULD SCHOOL OF LAW
699 Exposition Blvd.
Los Angeles, CA 90089-0071
(213) 740-7088
jef@law.usc.edu

*Counsel for Amicus Curiae*

## TABLE OF CONTENTS

## TABLE OF AUTHORITIES

**Page(s)**

**Cases**

**Statutes**



**Legislative Authorities**

**Other Authorities**

## INTEREST OF AMICUS CURIAE

Amicus Engine Advocacy ("Engine") is a non-profit technology policy, research, and advocacy organization that bridges the gap between policymakers and startups, working with government and a community of high-technology, growth-oriented startups across the nation to support the development of technology entrepreneurship.[1] Engine conducts research, orpganizes events, and spearheads campaigns to educate elected officials, the entrepreneur community, and the general public on issues vital to fostering technological innovation.

Engine seeks to bring to the Court's attention the unique perspective of high-technology startups on the impact of this case. In particular, Engine submits this brief to highlight the damage to startups, small businesses, entrepreneurs, and innovators that would result from extending copyright protection to API declarations. Such an extension would be a stark departure from longstanding legal principles and industry practice. Engine urges this Court to reverse the Federal Circuit's decision, which would hinder American innovation and harm the economy as a whole.

---

[1] Petitioner's blanket consent is on file with the Clerk and Respondent consented to the filing of this brief. No counsel for a party authored this brief in whole or in part, and no party or counsel for a party made a monetary contribution intended to fund its preparation or submission. No person other than the amici or their counsel made a monetary contribution to the preparation or submission of this brief.

## SUMMARY OF ARGUMENT

On its face, this case presents questions about copyright law. But under the surface, this lawsuit represents an end-run around the carefully constructed requirements and limitations of patent law. Respondent seeks to use copyright to obtain a monopoly on a functional system—its API declarations. But protection of ideas, systems, and methods is and has always been the domain of patents.

More than a century ago, in *Baker v. Selden*, this Court recognized that copyright protected only the expression of an idea. It could not grant exclusivity over the idea itself; that was the exclusive domain of patents. When Congress codified *Baker* in 17 U.S.C. § 102(b), it excluded any "process, system, [or] method of operation" from copyright protection. And since *Baker*, courts have been careful to avoid allowing copyright to encroach on the domain of patents.

The API declarations at issue here are just such a "process, system, [or] method of operation." Interoperability between Java developers and the Android operating system is impossible without the use of those declarations; writing different declarations will not work. The creative choices made in naming the declarations are no more expressive than the choices of labels on a filing cabinet of forms. In other words, to the extent there is any expression in those choices, that expression has merged with the functionality and is not protectable under copyright law.

If the decision below is upheld, it will create exactly the type of "surprise and fraud upon the public" that this Court warned of in *Baker*. Patent law

has a host of policy-related requirements that must be met before a system or process may be protected. For example:

- A patent must complete a rigorous examination process.
- It must claim patent-eligible subject matter.
- It must claim a new and non-obvious invention.
- It must prove to the reader that the applicant possessed the invention.
- It must teach the reader how to use the invention.

Even when granted, a patent monopoly is limited:

- Exclusivity lasts 20 years.
- Exclusivity is limited to the precise invention defined for the public by the patent's claims.

Copyright law contains *none* of these requirements or restrictions. Copyright attaches as soon as words are put to paper or saved on a computer—no examination is necessary. No one checks to determine if copyrighted matter is novel or non-obvious because it does not have to be. Copyright lasts for 95 years or 70 years beyond the lifetime of the author. And the author does not need to inform anyone of the scope of the rights they are claiming.

The decision below permits Oracle to use copyright to obtain patent-like protection over the functionality of their API declarations. This gives it broad, long-lasting rights of unspecified scope, with no requirement that their creation is new and no examination by anyone.

Allowing this end-run around patent law will have far-reaching consequences. The software industry, which has relied on free copying of declarations since

it began, will find itself in a morass of copyright-based protection of inventions. Startups in particular will find it difficult to withstand the licensing or litigation costs associated with this retroactive source of liability. And those that wish to interoperate with existing software will find it particularly difficult to secure funding, survive, and innovate.

The Court should therefore reverse the decision below and hold that API declarations are uncopyrightable under 17 U.S.C. § 102(b).

## ARGUMENT

The API declarations[2] at issue in this case are, at their core, functional systems. If a monopoly over those systems is to be granted, it must be granted through patents. This case represents an attempt by Respondent to use copyright law to evade patent law.

If it succeeds, Respondent will have avoided the critical requirements and limitations of patent law to obtain a long-lasting, unexamined, and automatic monopoly over a method. This is precisely what this

---

[2] Throughout this brief, we use the terms "API declarations" or "declarations" to refer to the portions of the API at issue here, including any structure, sequence, or organization (SSO) embodied by those declarations, but not the code implementing the API.

We do not use the term "declaring code," despite its use by the court below. "Declaring code" has never been used this way in the software industry or by computer scientists. *See*, *e.g.*, *Ngram Viewer*, Google Books, https://perma.cc/5CTS-SXHQ (showing common use of "API declaration" but not a *single* use of "declaring code" in books through 2008). Declarations do not execute and are therefore not "code."

Court explicitly avoided in *Baker v. Selden* in 1879 and Congress subsequently forbade, but that the Federal Circuit's decision on copyrightability makes real. *See Oracle Am., Inc. v. Google Inc.*, 750 F.3d 1339 (Fed. Cir. 2014) [hereinafter *Copyrightability Opinion*].

## I. The API Declarations at Issue Are a Fundamentally Functional System.

The API declarations at issue in this case are the type of fundamentally functional system that has historically been excluded from copyright protection and instead lies within the province of patents. As others have aptly explained in this proceeding, API declarations "embod[y]" a "process, system, [or] method of operation."[3] *See* Pet. Br. 17–18. *See also* Computer Scientists Br. Cert. Supp. Pet. 5 (2019). A monopoly on a set of API declarations is not, at its core, the exclusive right to a creative work. It is the exclusive right to control a particular way to operate a computer.

It is undisputed that source code may generally be copyrighted, even though it can be compiled and

---

[3] Throughout this brief, we use these terms interchangeably, as all apply:

- Each individual declaration also defines a method of operation through which one piece of software can operate another. (It is, in fact, called a "method" in programming.)
- Each declaration also defines the process through which that method can be used by a programmer.
- Taken together, a set of declarations specifies a functional system.

The precise classification does not matter; none are copyrightable under 17 U.S.C. § 102(b).

executed to perform a function. But it is also undisputed that another party may write their own code that performs the same function, because a copyright does not confer an exclusive right to the function performed by the source code.

API declarations are different. If a third party writes an app in the Java language, and Google wants to write software—Android—that can run that app, it *must* use API declarations that are essentially the same as Oracle's. Allowing Oracle to copyright those declarations, and the SSO embodied in them, *gives Oracle an exclusive right to that functionality*—running Java code. All the millions of independently written lines of source code for Android's Java implementation cannot function without those declarations to connect Java developers to the Android system.

This difference between source code and API declarations is critical: while copyright can be used to protect code, it cannot be used to protect the functionality itself. API declarations, if they are to be protected, fall within the domain of a patented system, uncopyrightable under 17 U.S.C. § 102(b).

Google has correctly analogized the names of Java packages and methods to labels on a filing cabinet, its drawers, and the folders it contains. Pet. Br. 4–5. It may be that the labels embody some creativity. But the labels are also a system or process for locating methods in their respective folders. And if they are given copyright protection, they give control over the process for locating folders, not just the names of the folders.

Since *Baker*, such systems and processes have explicitly been the domain of patents. If a party wants to offer their own folders to clerks who have experience with a particular filing system, they are free to do so unless the system itself is patented. But that party *must* use the same labels if they want clerks to be able to access their folders. If the labels are different, clerks will not know how to use their system and all their previous experience is wasted.[4] Granting copyright protection to the labels thereby grants a monopoly not over the expression, but the system and how it functions.

## II. Congress and the Courts Have Carefully Avoided Awarding Copyright Protection to Functionality and Invading the Domain of Patents.

Where patents protect inventions, copyrights protect expressive works. *See, e.g.*, Dennis S. Karjala, *The Relative Roles of Patent and Copyright in the Protection of Computer Programs*, 17 J. Marshall J. Computer & Info. L. 41, 45 (1998) (noting that functionality is the main differentiator between traditional patent and copyright subject matter, whereas "[p]atents protect creative, functional invention; copyright protects creative, nonfunctional authorship.").

---

[4] This case has involved some dispute over the level of interoperability between Android and Java. That dispute is irrelevant. Even if the new system only provides some of the cabinets and folders that were in the original and adds new ones, the old ones must use the existing labels for clerks to use them. The same is true of Java methods.

Congress, the courts, and federal agencies consistently make clear the respective, non-overlapping roles of copyrights and patents. In particular, they have sought to prevent awarding exclusivity over functionality through copyright. Oracle's attempt to extend copyright protection to API declarations crosses the law's carefully drawn lines.[5]

### A. For 140 Years, Copyright Protection Has Been Unavailable for the Ideas or Systems Described in Works.

Since at least the late 19th century, this Court has recognized the harms that flow from misapplying copyright to protect function. In *Baker v. Selden*, the Court addressed an attempt to use copyright in a book describing a system of bookkeeping to control the use of that system.

The Court recognized the danger, and held that copyright could not be used in this fashion:

> The description of the art in a book, though entitled to the benefit of copyright, lays no foundation for an exclusive claim to the art itself. The object of the one is explanation; the object of the other is use. The former may be secured by copyright. The latter can only be secured, if it can be secured at all, by letters-patent.

---

[5] In addition to the statutory and case law discussed herein, there are other doctrines that ensure copyright protection is limited to non-functional, expressive content. For example, though inapplicable here, the useful article doctrine prohibits protection of the useful aspects of pictorial, graphic, or sculptural works. *See* 17 U.S.C. § 101.

*Baker v. Selden*, 101 U.S. 99, 105 (1879).

Critically, the Court also held that the forms in the book embodied the bookkeeping system itself, and were therefore also unprotectable under copyright:

> In describing the art, the illustrations and *diagrams employed happen to correspond more closely than usual with the actual work performed by the operator who uses the art*. Those illustrations and diagrams consist of ruled lines and headings of accounts; and it is similar ruled lines and headings of accounts which, in the application of the art, the bookkeeper makes with his pen, or the stationer with his press; . . .

*Id.* at 104–05 (emphasis added).

*Baker* held that copyright protection did not apply to forms that embodied an accounting method because copyright should not extend to utilitarian works whose expression exists solely to assist in performing its function. Thus, while Selden could copyright the explanation of his bookkeeping system, he could not copyright the bookkeeping forms even though they contained original expression because they embodied a process that others had to copy "for the purpose of practical application." *Id.* at 103.

The Court also signaled how the legal system has viewed and continued to view copyright and patent as playing distinct roles in protecting different intellectual creations. *See id.* *Baker* showed how copyright law could provide one layer of protection for some aspects of original works (Selden's explanation of his bookkeeping system), while patent law could apply to other aspects (the bookkeeping system

embodied in Selden's book). Granting copyright over functionality to authors of useful arts "is the province of letters patent," so allowing copyright over functionality when "no examination of its novelty [under patent law] has ever been officially made" would be "a surprise and a fraud upon the public." *Id.* at 102.

Because the Federal Circuit's *Copyrightability Opinion* ignores these fundamental principles, threatens to undermine decades of well-reasoned case law, and upends industry expectations surrounding software innovation, it should not be allowed to stand.

### B. Section 102(b) Confirms that Protection of Functionality Is the Domain of Patents, Not Copyright.

Section 102(b), which codifies the distinction articulated in *Baker*, states that copyright protection for an original work of authorship does not extend to "any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is . . . embodied in such work." 17 U.S.C. § 102(b). The legislative history of § 102(b) shows that the statute was enacted to confirm *Baker* and its progeny to ensure that computer program copyrights would not be construed to embrace functionality:

> Some concern has been expressed lest copyright in computer programs should extend protection to the methodology or processes adopted by the programmer, rather than merely to the 'writing' expressing his ideas. Section 102(b) is intended, among other things, to make clear that the expression adopted by the programmer is the

> copyrightable element in a computer program, and that the actual processes or methods embodied in the program are not within the scope of the copyright law . . . . Section 102(b) in no way enlarges or contracts the scope of copyright.

H.R. Rep. No. 94-1476, at 57 (1976).

While testifying at the 1967 Senate hearings leading up to the passage of § 102(b), Professor Arthur Miller raised concerns that software copyrights would "likely confer patent like protection under the guise of copyright" and asked Congress to make clear that copyright would not cover "the art, process or scheme that is fixed in [a] program." Pamela Samuelson, *Why Copyright Law Excludes Systems and Processes from the Scope of Its Protection*, 85 Tex. L. Rev. 1921, 1950 (2007) (citations omitted). When asked to craft specific language for this purpose, Professor Miller provided the text that would ultimately become § 102(b). *Id.*

Section 102(b)'s prohibition against copyrighting processes, systems, and methods of operation is further enforced through the merger doctrine. This doctrine prevents an author from indirectly asserting copyright over a system or method by claiming the exclusive right to the only expression for that system or method. *See* Pet. Br. 21. In software,

> [w]hen the "idea" and its "expression" are [] inseparable, copying the "expression" will not be barred, since protecting the "expression" in such circumstances would confer a monopoly of the "idea" upon the copyright owner free of the conditions and limitations imposed by the patent law.

*Atari Games Corp. v. Nintendo of America, Inc.,* 975 F.2d 832, 837, 839. *See also, e.g.*, Uses of Copyrighted Works, *Final Report* 20 (1979) ("when specific instructions, even though previously copyrighted, are the only and essential means of accomplishing a given task, their later use by another will not amount to an infringement."). Thus, one cannot control the use of the functionality of API declarations merely because they include expressive names.

### C. Courts and Agencies Since *Baker* and § 102(b) Have Consistently Cabined Copyright.

These principles continue to be endorsed by federal agencies and the courts. A 1991 study conducted by the U.S. Copyright Office and the U.S. Patent and Trademark Office concluded that copyright "protection of the functionality of [] software itself" would be contrary to § 102(b). U.S. Copyright Office & U.S. Patent & Trademark Office, *Patent-Copyright Laws Overlap Study* 11, 87–88 (1991), https://perma.cc/5ZU9-HCFX. The categories described in § 102(b) cannot qualify for copyright protection, and are instead "assigned to patents where a much more rigorous test must be undergone and the barriers to entry, in terms of time, cost, and complexity, are higher." *Id.* at 88.

Courts have applied these principles in the software context. For example, the Federal Circuit has itself noted that whereas copyright protects program expression, authors should look to patents to protect methods of operation or processes. *See Atari*, 975 F.2d at 842 ("An author cannot acquire patent-like protection by putting an idea, process, or method of operation in an unintelligible format and asserting

copyright infringement against those who try to understand that idea, process, or method of operation.") (citing *Feist Publications, Inc. v. Rural Tel. Serv. Co.*, 499 U.S. 340, 349-50 (1991); 17 U.S.C. § 102(b)).

In *Lotus v. Borland*, the First Circuit affirmed these principles, holding that a "menu command hierarchy is an uncopyrightable 'method of operation.'" *Lotus Dev. Corp. v. Borland Int'l, Inc.*, 49 F.3d 807, 815 (1st Cir. 1995), *aff'd by an equally divided Court*, 516 U.S. 233 (1996) (per curiam). The facts and reasoning in *Lotus* are quite close. Borland had replicated the names and structure of the menus in Lotus's product, which allowed both users and software "macros" to interact with spreadsheet commands without relearning a new set of commands. *Id.* at 809–810. Borland did not copy Lotus's underlying code. *Id.* at 810.

In other words, just like with the Java API declarations, Borland had to provide the exact same command names and structure in order for existing users and software to interact with their new, original software. But as the court recognized, "[t]he 'expressive' choices of what to name the command terms and how to arrange them do not magically change the uncopyrightable menu command hierarchy into copyrightable subject matter." *Id.* at 816. The *Lotus* court also recognized the difference between the *uncopyrightable names and structure* of the commands and the *copyrightable code* that implements them:

> The Lotus menu command hierarchy is [] different from the underlying computer code, because while code is necessary for the program to work, its precise formulation is

> not. . . . [T]o allow users to operate its programs in substantially the same way, however, Borland had to copy the Lotus menu command hierarchy.

*Id.* at 816.

Judge Boudin additionally noted how granting copyright protection to certain aspects could have "some of the consequences of patent protection" because it would "limit[] other people's ability to perform a task in the most efficient manner." *Id.* at 819 (Boudin, J., concurring). And he recognized the tension of using copyright rather than patent protection, explaining that "[i]t is no accident that patent protection has preconditions that copyright protection does not—notably, the requirements of novelty and non-obviousness—and that patents are granted for a shorter period than copyrights." *Id.*

Other courts have maintained the same distinction between copyrightable code and uncopyrightable names and structures. For example, in *MiTek Holdings*, the court held a menu and submenu command tree structure to be an uncopyrightable process under § 102(b). *MiTek Holdings, Inc. v. Arce Engineering Co.*, 89 F.3d 1548, 1556 (11th Cir. 1996). To find otherwise "would be affording copyright protection to a process that is the province of patent law." *Id.* at 1556 n.19.

Relatedly, courts consistently limit the scope of copyright claims over the functionality of software. In *Altai*, the Second Circuit adopted a "successive filtering method for separating protectable expression from non-protectable material." *Computer Assocs. Int'l, Inc. v. Altai, Inc.*, 982 F.2d 693, 707 (2d Cir. 1992)

(quotation omitted). The court excluded from protection components of the software that were not protectible, including those "dictated by external factors," including "compatibility requirements of other programs with which a program is designed to operate in conjunction." *Id.* at 710 (quoting 3 Melville B. Nimmer & David Nimmer, *Nimmer on Copyright* § 13.03[F][3] at 13–66–71 (1991)). The court recognized that allowing copying of these elements "is neither unfair nor unfortunate. It is the means by which copyright advances the progress of science and art." *Id.* at 721 (quoting *Feist*, 499 U.S. 340).[6]

*Altai* quickly became a standard, ultimately becoming "the dominant test." 4 Nimmer on Copyright § 13.03[A][1][d] (2019). At least the Fourth, Fifth, Ninth, Tenth, and Eleventh circuits use the *Altai* approach to determining the scope of protection of software under copyright law. *Id.* § 13.03[F][1][c] n.283.08. Some explicitly invoke the contrast to patent law. For instance, the Ninth

---

[6] The *Altai* decision is also noteworthy for its explicit rejection of the decision in *Whelanh Associates, Inc. v. Jaslow Dental Laboratory, Inc.*, 797 F.2d 1222 (3d Cir. 1986). In *Whelan*, the court had held that that the SSO of computer programs was broadly copyrightable. *Id.* at 1248. This effectively allowed the plaintiff to assert a copyright over the functionality of that software. The *Whelan* decision has been widely criticized for adopting patent-like standards for copyright infringement that could have a chilling effect on software development and stifle innovation. *See, e.g.*, David Nimmer et al., *A Structured Approach to Analyzing the Substantial Similarity of Computer Software in Copyright Infringement Cases*, 20 Ariz. St. L.J. 625, 630 (1988); Bruce Abramson, *Promoting Innovation in the Software Industry: A First Principles Approach to Intellectual Property Reform*, 8 B.U.J. Sci. & Tech. L. 75, 131 (2002).

Circuit in *Apple Computer, Inc. v. Microsoft Corp.*, 35 F.3d 1435 (9th Cir. 1994) emphasized that Apple could not "get patentlike protection for the idea of a graphical user interface" by claiming copyright in "an essentially functional process," *id.* at 1443–44.

In sum, the enactment and interpretations of § 102(b) reinforce the distinction between patent and copyright. Protecting API declarations under copyright would invade the province of patent law in a manner contrary to both the intent of § 102(b) and longstanding court decisions.

### D. The Federal Circuit Erroneously Allowed Copyright to Protect Functionality by Confusing "Software Programs" and "Declarations."

The Federal Circuit's *Copyrightability Opinion* deviates from the case law in part because it turned the question of API declaration copyrightability into the question of whether *software generally* should be protected by copyright or patents. But that is simply not the issue. Instead, the question before the Federal Circuit and now before this Court is whether the asserted API declarations should be protected by copyright (or patent). It was erroneous to conflate "software programs" with "interfaces" and "declarations."

Just because something is on a computer does not make it protectable "software." In particular, API interfaces are not "computer programs," which are defined as "a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result." 17 U.S.C. § 101. Instead, a software interface "specifies the set of

commands used to operate a computer program or system." Computer Scientists Br. Cert. Supp. Pet. 5 (2019). "Declarations" are purely functional and consists of the name, inputs, and outputs of each command in an interface. *Id.* at 6. Without actual software to implement them, they do nothing.

The Federal Circuit missed the point when it "decline[d] any invitation to declare that protection of software programs should be the domain of patent law, and only patent law." *Oracle Am., Inc.*, 750 F.3d at 1381. This case requires no such statement. It requires only that the Court recognize that the functional aspects of API declarations, like all systems and methods, are outside the domain of copyright law. Copyright over the appropriate expressive elements of computer programs—e.g., source code—can remain intact.

* * *

Because code written by third parties and code by Google cannot interoperate unless the Java SE declarations are copied, those declarations are exactly the kind of functional process that copyright excludes. *Baker* and its progeny make clear that if Oracle wanted to protect the functionality of the Java declarations, it should have sought patent and not copyright protection. Because the *Copyrightability Opinion* ignores well-established functional limits on copyrightability where protection of such functionality is properly the domain of patents, it should be reversed.

## III. The Decision Below Permits Parties to Evade the Carefully Constructed Requirements for and Limitations on Patenting.

The quid pro quo of the patent system dictates that an inventor who wants to obtain exclusive rights to a useful process—something functional such as a method, a system, or a way to perform a task—must satisfy certain requirements. For example, he or she must prove that the invention is new, must describe it in specific patent claims to delineate the scope of exclusivity, and must allow the invention to enter the public domain within (approximately) 20 years.

There are practical differences between the protection the law affords to patents versus copyrights because the nature of the protection and how one obtains it are different. "In general, copyrights are easier to secure and last substantially longer than patents, although the scope of protection afforded copyrights is narrower and less absolute than that given to patents." Robert P. Merges et al., *Intellectual Property in the New Technological Age: 2019* 37 (2019). By contrast, the requirements for securing a patent are intentionally more onerous. Patents afford their owners broad rights, preventing others from performing certain useful processes and going so far as to bar others who reverse engineer or independently invent those processes from performing them. *See, e.g.*, *id.* at 167.

The API declarations in question—indeed, API declarations in general—are functional.[7] Open Source Initiative et al. Br. Cert. Supp. Pet. 7–12 (2014); Hewlett Packard et al. Br. Cert. Supp. Pet. 15–17 (2014). As explained above, the API declarations at issue in this case only exist for one use: allowing two pieces of software to communicate. *See also* Mozilla Corp. et al. Br. Supp. Pet. 8 n.2 (2019).

Because they are functional processes, any exclusivity afforded API declarations should fall under the rubric of patent, not copyright. And because patent rights are nearly absolute, the law requires that "a much more rigorous test must be undergone." *Overlap Study*, *supra*, at 88. But Oracle did not put its Java API declarations through those tests, and it avoided the ex ante time, cost, and complexity of the patent system by ex post asserting copyright protection.

Engine takes no position on whether there are any patentable inventions as issue in this case.[8] But there is reason to doubt that the methods embodied in API

---

[7] They are, in fact, called "functions"; a "method" is simply a function attached to an "object." *See* Marcin Moskala, *Kotlin Programmer Dictionary: Function vs Method vs Procedure* (Oct. 25, 2017), https://perma.cc/368P-FB7U ("all methods are functions").

[8] According to the district court, "[b]oth Oracle and Sun have applied for and received patents that claim aspects of the Java API. *See, e.g.,* U.S. Patents 6,598,093 and 7,006,855. (These were not asserted at trial.)" *Oracle Am., Inc. v. Google Inc.*, 872 F. Supp. 2d 974, 996 (N.D. Cal. 2012), *rev'd and remanded*, 750 F.3d 1339 (Fed. Cir. 2014).

declarations at issue here would meet all of the Patent Act's requirements. If those methods are unpatentable, then they are properly in the public domain. Copyright law should not be used as a gap-filler to protect unpatentable technology.

### A. Patent Applicants Must Satisfy Numerous Requirements to Prove New Inventions or Technologies Warrant Patent Protection.

There are multiple requirements for patentability that limit both what is eligible for protection and the scope of exclusivity granted. Many of these requirements are particularly applicable to software. And the *Copyrightability Opinion* allows parties like Oracle to evade all of them. Through that decision, Oracle has obtained a much longer window of exclusivity using copyright without ever having to establish that the API declarations are, e.g., novel, non-obvious, or adequately described. The following are some, but not all, of the restrictions that can be evaded if one can copyright systems and methods:

Examination. First and foremost, patents must go through a process in which an examiner at the U.S. Patent & Trademark Office assesses whether a patent application satisfies the statutory requirements for patentability. This rigorous process is necessary because patents grant strong rights, allowing the rightsholder to exclude others from using some technologies. Such expansive rights are only granted when the public gets something in return: a new, fully disclosed invention. *J.E.M. Ag Supply, Inc. v. Pioneer Hi-Bred Int'l, Inc.*, 534 U.S. 124, 142 (2001) ("The disclosure required by the Patent Act is the *quid pro quo* of the right to exclude.") (quotation omitted).

Examination also costs money and takes time. The average patent application currently takes 23.9 months. *Data Visualization Center*, USPTO, https://www.uspto.gov/dashboards/patents/main.dashxml (last accessed Jan. 11, 2020). Prosecuting an application for a software patent can cost upwards of $16,000 dollars. Gene Quinn, *The Cost of Obtaining a Patent in the US*, IPWatchDog (Apr. 4, 2015), https://perma.cc/4TLC-ZAWL. This forces potential applicants to decide, ex ante, when their time resources are well-spent on a patent application.

Copyright protection, by contrast, is effectively automatic.[9] If API declarations were copyrightable, one would obtain a copyright merely by writing the API declaration down or entering it into a computer.

---

[9] One need register a copyright to sue and to obtain certain remedies, but rights vest automatically. *See* 17 U.S.C. § 102(a) (copyright protection arises once the work is fixed in any tangible medium of expression); § 408 (describing registration).

This is particularly troubling given the facts of this case because, as noted by other amici, the entire software industry has been treating APIs as uncopyrightable since the inception of that industry. *See* Computer Scientists Br. Cert. Supp. Pet. 2, 4–6 (2014). Because copyright protection is automatic, finding Oracle's declarations copyrightable in 2020 could retroactively apply to *all* APIs.

The law, for good reason, does not automatically allow an inventor to exclude others from performing a useful process in absence of patent examination. It should not be enough to call oneself an "author," instead of "inventor" to capture the same functional exclusivity through the automatic process of copyright.

Novelty & Non-Obviousness. Patented inventions must be novel and cannot be obvious. 35 U.S.C. §§ 102, 103. In short, if the subject matter of a patent claim has been disclosed to the public, or would have been obvious, before a patent applicant files or makes their own public disclosure, that claim is invalid and cannot be infringed. *Id.* These requirements stem from the fact that "Congress may not authorize the issuance of patents whose effects are to remove existent knowledge from the public domain, or to restrict free access to materials already available. Innovation, advancement, and things which add to the sum of useful knowledge are inherent requisites in [the] patent system." *Graham v. John Deere Co.*, 383 U.S. 1, 6 (1966).

There is no such requirement in copyright. A party is welcome to copyright something that is not new as long as it was not copied. One can likewise copyright a work that contributes nothing substantive

beyond what is already known. If Oracle's API declarations are copyrighted, this does not technically block independent creation of the same API. But it does effectively block any follow-on uses of the API declarations because their core purpose is interoperability through reuse, which as a practical matter requires copying. *See* Pet. Br. 27. Thus, even if Oracle's methods are not new or unique, copyright protection could allow it to effectively obtain a monopoly on its functionality.[10]

Patent Eligibility. In the same way that copyright law applying to software has developed careful limitations to avoid encroaching on the domain of patents, patent law has developed rules to avoid granting protection to abstract ideas. The law excludes such abstract ideas because they are "the basic tools of scientific and technological work." *Alice Corp. Pty. Ltd. v. CLS Bank Int'l*, 573 U.S. 208, 216 (2014) (quotations omitted). Allowing a monopoly over an abstract idea would tend to impede, not promote, innovation. *Id.* (quoting U.S. Const., Art. I, § 8, cl. 8). This key eligibility question is considered under the rubric of subject matter, governed by 35 U.S.C. § 101—not copyright law.

This Court has addressed software and computer program-related patents multiple times, consistently interpreting patent eligibility in a way that ensures

---

[10] Indeed, much of the Java API replicates pre-existing APIs from other programming languages, suggesting that even if APIs were copyrightable, Oracle might not be entitled to one here. *See, e.g.,* Trial Testimony of Joshua Bloch, JA155 ("And instead of designing our own API from scratch, we decided we would use the regular expression API from this language called Perl 5.")

that patents not be allowed to preempt abstract ideas. Specifically, *Gottschalk v. Benson*, 409 U.S. 63 (1972) found that the method for transforming binary coded decimals to pure binary form was too abstract to be patented. *Id*. at 68. And this Court's decision in *Parker v. Flook,* 437 U.S. 584 (1978) held that a software process for updating alarm limits in a catalytic converter was not patentable. *Id.* at 594.

This court revisited patentable subject matter in 2014 in *Alice Corp. v. CLS Bank International. Alice* clarified the scope of software patents. The Court applied a two-part test for when a patent claimed an ineligible abstract idea. *Alice Corp.*, 573 U.S. at 217–218. It ultimately held that "mere recitation of a generic computer cannot transform a patent-ineligible abstract idea into a patent-eligible invention." *Id.* at 223.

Since *Alice*, numerous courts have considered the patent eligibility of software and computer program-related claims. The law allows patents on such inventions that offer a technological solution to a technological problem, including solutions involving software. *See, e.g.*, *Enfish, LLC v. Microsoft Corp.*, 822 F. 3d 1327 (Fed. Cir. 2016). But the law bars protection over patent claims that recite nothing inventive or transformative beyond an abstract idea. *See, e.g.*, *Intellectual Ventures I LLC v. Capital One Financial Corp.*, 850 F.3d 1332 (Fed. Cir. 2017). This same framework should be the one applied to evaluate the eligibility of API declarations.

Overall, the answer to whether or a system or method is protectable must be given by patent law, not copyright law. The critical subject matter restrictions

that apply to software are completely evaded by allowing copyright to protect functionality.

Limitations on Functional Claiming. Patent law permits claims that recite a function without including the "structure, material, or acts in support thereof." 35 U.S.C. § 112(f). These "functional claims" are much like an API specification—they tell the reader *what* is going to be done, but not *how* to do it. In order to ensure that the public knows the scope of the monopoly, patent law limits such claims to the "the corresponding structure, material, or acts described in the specification"—in other words, to the specific implementation(s) proposed by the patentee in the same patent. *Id.*; 35 U.S.C. § 112(f).

No such limitation is found in copyright law. Indeed, copyright law does not require one to recite the scope of the protection because copyrights have no claims and protection is automatic. Put another way, a copyright owner does not have to tell the public what the scope of the copyright exclusivity is *at all*, so there are no requirements on how detailed such a description must be. Once more, patent law has specifically excluded the types of broad, easy protection that arise when copyright is permitted to cover functionality.

### B. Limitations on the Nature and Scope of Patent and Copyright Protection Diverge in Important Ways.

In addition to the substantive requirements for patentability, there are differences in the protection afforded under copyright and patent law that have practical significance when considering the copyrightability of API declarations.

Shorter Terms. Patents have a 20-year term from the date of application. 35 U.S.C. § 154(a)(2). Copyright lasts for the lifetime of the author plus 70 years or for 95 years from publication, depending on the circumstances. 17 U.S.C. § 302.

Congress has determined that inventions should be free for the public to use far sooner than creative works. *See Lotus Dev. Corp.*, 49 F.3d at 819 (noting that although the need to incentivize the creation of new inventions would typically suggest granting a longer monopoly through patent protection, the "high value on public access" requires "that patents are granted for a shorter period than copyrights."). Here, Oracle would gain at least an additional 50 years of protection—and potentially more than a century—by side-stepping patents.

Clarity of Scope. A patent grants a monopoly only over what is claimed. *Dawson Chem. Co. v. Rohm & Haas Co.*, 448 U.S. 176, 221 (1980). This provides the public a useful if imperfect guide to ensure that they know what activity is infringing and should be licensed and what activity they may freely undertake.

Copyright contains no such requirement. While it is well-understood that only portions of a work may be subject to copyright protection, there is no requirement or mechanism for authors to make clear what rights they have or even what rights they are claiming. Pamela Samuelson, *Strategies for Discerning the Boundaries of Copyright and Patent Protections*, 92 Notre Dame L.R. 1493, 1498 (2017). If copyright grants exclusivity over API declarations, alone or in some combination, no user will know what portions, if any, are free to use until they are sued.

## IV. The Consequences of Allowing This End-Run Are Particularly Bad for Startups.

If the Federal Circuit's rulings are upheld, the entire software industry will face new, pervasive, and potentially retroactive forms of liability and the accompanying legal costs. While larger companies may have the resources and budgets to handle such a burden, these new costs would likely be fatal for many startups. This, in turn, will discourage investment in future startups, thereby creating a less-competitive and less-innovative industry.

### A. Startups Drive Innovation and Job Creation.

Startups play a vital role in our economy as innovators and job creators. Startups are more willing and able than their well-established counterparts to take the risks necessary to bring new ideas to the market. Sam Hogg, *Why Small Companies Have the Innovation Advantage*, Entrepreneur (Nov. 15, 2011), https://perma.cc/RFN8-LF7Y. In doing so, they act as a primary source of job creation and drive "economic dynamism" by injecting competition into markets and accelerating innovation. Jason Wiens & Chris Jackson, *The Importance of Young Firms for Economic Growth*, Ewing Marion Kauffman Foundation (Sept. 13, 2015), https://perma.cc/GC4G-RVTQ.

The economic importance of high-tech startups is not lost on lawmakers. For example, Senator Wyden has recognized, "[m]ost innovation in the digital economy comes from the startups and small firms." Senator Ron Wyden, Press Release, *Wyden Issues Warning About SESTA* (Nov. 8, 2017), https://perma.cc/XXD6-QMX2; *see also* Representative

Chrissy Houlahan & Representative Cathy McMorris Rodgers, *What Makes America Great is What Makes American Startups Thrive*, Roll Call (Aug. 21, 2019), https://perma.cc/26GX-L93T ("small businesses and startups are the true drivers of our economy.").

### B. Copyright-Protected APIs Threaten Interoperability.

API declarations are not just functional; in enabling interoperability they enable a function that is essential to high-tech startups. Exclusivity over the use of API declarations has patent-like consequences, which exacerbates the problem of permitting copyright protection to bar their use. *See, e.g.*, *Lotus Dev. Corp.*, 49 F.3d at 819 (Boudin, J., concurring) ("[i]t is no accident that patent protection has preconditions that copyright protection does not . . . and that patents are granted for a shorter period than copyrights").

API declarations make development tools more accessible and compatible, which means "competition and innovation in the software industry have thrived," and a vast "array of interoperable software products and services" are available to consumers. Pamela Samuelson & Clark D. Asay, *Saving Software's Fair Use Future*, 31 Harv. J.L. & Tech. 535, 562 (2018). Indeed, personal computers exist today because of the long-held assumption that APIs were uncopyrightable, which enabled developers to reimplement and build off of existing operating systems to create new compatible systems. *See* Computer Scientists Br. Cert. Supp. Pet. 6–10 (2014).

Startups particularly depend on access to API declarations. Software Innovators, Startups, and Investors Br. Cert. Supp. Pet. 5–9 (2019). "Any

startup must confront this interoperative world, and find ways for its products to connect to the existing universe of products, platforms, content, and services." *Id.* at 5.

Extending copyright to API declarations would improperly threaten interoperability, limiting a startup's "ability to perform" very important "task[s] in the most efficient manner." *Lotus Dev. Corp.*, 49 F.3d at 819 (Boudin, J., concurring). API rightsholders would be given "veto power over any developer who wants to create a compatible program." Corynne McSherry, *Dangerous Decision in Oracle v. Google: Federal Circuit Reverses Sensible Lower Court Ruling on APIs*, Electronic Frontier Foundation (May 9, 2014), https://perma.cc/9N5T-8VCC. Companies would be forced to seek permission to compete from their competitors.

While large companies might have the bargaining power to reach such an agreement, startups would likely be unable to operate. As a result, extending copyright to APIs would "create an environment where only larger, more established businesses can operate, and further reduc[e] the ability of start-ups and small businesses to compete." *Quantifying Risks to Interoperability in the Software Industry*, Developers Alliance & NDP Analytics 12 (Dec. 2017), https://perma.cc/Q8J3-CSVK.

### C. Holding APIs Copyrightable Would Create Unsustainable Legal Costs for Startups.

Holding API declarations copyrightable would upend decades of settled expectations within the software industry by creating new, possibly

retroactive sources of liability that would force companies to assume serious financial costs that many startups are ill-equipped to handle. Indeed, developers have long reused existing API declarations with the shared understanding that those API declarations were not subject to claims of copyright protection.

At least part of the problem is that none of those API declarations are disclosed in patent claims. If, by contrast, the Java SE declarations were disclosed in patent claims, the developer community would at least be put on notice of potential infringement liability associated with using them. Giving API declarations, properly "the province of letters patent," copyright protection now would result in the exact type of "surprise and [] fraud upon the public" that this Court has warned of. *Baker*, 101 U.S. at 102. And the practical consequences of that surprise could be incredibly costly for startups.

Copying elements from APIs has become a standard practice in the software industry. *See* Computer Scientists Br. Cert. Supp. Pet. 4–6 (2014). For reimplementation, as Google did, declarations must be written identically. *Oracle Am., Inc.,* 872 F. Supp. 2d at 978 ("thus, the 'declaration' . . . must be identical to carry out the given function."). Likewise, since programmers are often intimately familiar with one API, the developer of a competing API would choose to write theirs with identical declarations and organization for the sake of accessibility, in much the same way that the designer of a new car would use the same order of gears on the car's gearshift—Park, Reverse, Neutral, Drive, and Low (commonly referred

to as "PRNDL"). Open Source Initiative et al. Br. Cert. Supp. Pet. 10 (2014).

Because of these functional concerns, extending copyright protection to API declarations would create "a legal minefield" for software developers. Charles Duan, *Oracle Copied Amazon's API—Was That Copyright Infringement?*, ArsTechnica (Jan. 3, 2020), https://perma.cc/YH6J-QPBG. As noted above, unlike patent, copyright provides blanket protection "without requiring authors to specify the expressive elements of their works to which copyright extends." Samuelson, *Strategies*, *supra*, at 1498. Therefore, if the *Copyrightability Opinion* is affirmed, startups will have spent decades unknowingly developing their way into expansive and expensive copyright litigation risk by using API declarations in the standard way.

The risk of this type of litigation is real. Indeed, as other amici have explained, parties have already begun attempting to obtain protection over functionality through copyright. *See* Electronic Frontier Foundation Br. Cert. Supp. Pet. 7–10 (2019). For instance, parties in recent litigation have sought a copyright-based monopoly over the commands used to configure network switches and the use of particular programming languages, explicitly relying on the decisions below. *Id.*

Under the Federal Circuit's rule, "it could be open season on [] software companies, big and small, that rely on unlicensed APIs," as every company could suddenly be exposed to litigation. Jeff John Roberts, *Google and Oracle's $9.3 Billion Fair Use Fight Starts Today, Here's a Guide*, Fortune (May 9, 2016), https://fortune.com/2016/05/09/google-oracle-fair-use/. As a result, "large and small software tech companies

[would] have to divert more and more resources away from development, and toward litigation." McSherry, *supra*.

The problem extends beyond creating new liability for past use of API declarations. If the lines between copyright and patent are allowed to blur, future cases may allow companies to overstep even further to gain copyright protection over more useful processes. This will create increasing uncertainties and liability that could stifle startup innovation.

Large, well-established companies likely have the resources to thoroughly assess their legal liabilities or withstand litigation over poorly delineated rights. Startups are a different story. Startups have limited budgets and depend on investment. Many lack the resources necessary to adequately assess their litigation risks, let alone defend a copyright infringement suit from start to finish.

Extending copyright protection to APIs, or other functional aspects of computer programs, would therefore impose likely-unsustainable costs on numerous software startups, forcing many out of business and in turn discouraging both entrepreneurs and investors from future ventures. *See, e.g.*, Senator Ron Wyden, *Floor Remarks: CDA 230 and SESTA*, Medium (Mar. 21, 2018), https://perma.cc/V5WZ-VCAE ("Fewer [investors] will be willing to risk their deep pockets if their early-round investments are swallowed up by legal fees instead of paying for coders."). Both startups and investors need certainty before they take the technological and business risks inherent in innovation.

### D. Discouraging Startups Would Stunt Innovation and Competition in the Industry.

As discussed above, increased litigation and licensing costs would negatively affect the entire software industry. *Supra* Part IV.B-C. However, not all companies would be affected equally. Instead, these costs would fall disproportionately on small players by creating a higher barrier to entry for newcomers while entrenching and consolidating the incumbents that can afford to adapt.

By discouraging startups, the industry would lose the many benefits they provide. As a result, there would be fewer jobs and less innovation, potentially costing $77 billion in lost economic productivity over the next several years. Developers Alliance, *supra*, at 2. For this reason, extending copyright to APIs would cause irreparable harm to startups, and in turn, the software industry that fuels the American economy.

## CONCLUSION

For the above reasons, amicus respectfully requests that the court hold that API declarations are uncopyrightable under 17 U.S.C. § 102(b).

Respectfully submitted,

Jef Pearlman
*Counsel of Record*
INTELLECTUAL PROPERTY & TECHNOLOGY LAW CLINIC
UNIVERSITY OF SOUTHERN CALIFORNIA GOULD SCHOOL OF LAW
699 Exposition Blvd.
Los Angeles, CA 90089-0071
(213) 740-7088
jef@law.usc.edu
*Counsel for Amicus Curiae*

January 13, 2020