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 *AMICI CURIAE* ALLIANCE OF U.S. STARTUPS AND INVENTORS FOR JOBS ("USIJ") IN SUPPORT OF RESPONDENT

DAVID BOUNDY
CAMBRIDGE TECHNOLOGY LAW LLC
P.O. Box 590638,
Newton, MA 02459

STEVEN C. CARLSON
ROBINS KAPLAN LLP
2440 West El Camino Real, Suite 100
Mountain View, CA 94040
(650) 784-4012

ROBERT P. TAYLOR
Counsel of Record
RPT LEGAL STRATEGIES PC
2443 Fillmore Street, Suite 332
San Francisco, CA 94115
(415) 447-3975
robert.taylor@rptstrategies.com

Counsel for Amici Curiae

294667

TABLE OF CONTENTS

Page

TABLE OF CITED AUTHORITIES

Page(s)

CASES

STATUTES AND OTHER AUTHORITIES

STATEMENT OF INTEREST

USIJ is a coalition of 21 startup companies and their affiliated executives, inventors and investors that depend on stable and reliable intellectual property protection as an essential foundation for their businesses. A list of USIJ members is attached as Appendix A.[1] USIJ was formed in 2012 to address concerns that legislation, policies and practices adopted by the U.S. Congress, the Federal Judiciary and certain Federal agencies were and are placing individual inventors and research-intensive startups at an unsustainable disadvantage relative to their larger incumbent rivals, both domestic and foreign, and others that would misappropriate their inventions and the creative expression in the software that often is part of any invention in the modern world. A disproportionately large number of breakthrough inventions are attributable to these individual inventors and small companies.

USIJ's fundamental mission is to assist and educate Members of Congress, the Federal Judiciary and leaders in the Executive branch regarding the critical role that patents and copyrights play in our nation's economic system and the particular importance of startups and small companies to our

---

[1] No counsel for a party authored this brief in whole or in part, and no such counsel or party made a monetary contribution intended to fund the preparation or submission of this brief. No person other than the amicus curiae made a monetary contribution to its preparation or submission. Both parties have consented to the filing of this brief.

country's dominance of strategically critical technologies for more than a century.

SUMMARY OF ARGUMENT

USIJ's principal concern in this case is broader than simply the impact that this Court's decision might have on the parties or on the use of the Java language and Java platform. Adoption of Google's arguments would risk unintended collateral damage to entrepreneurs, startups and other highly creative companies that rely on copyright protection to justify their investment of time and resources to build new businesses and bring new software products and services to market.

Computer source code is treated as a literary work under the Copyright Act, because it is the ultimate "expression" of the "ideas" that underlie the functional operation of a given software program. Although filtering the unprotectable abstract ideas from protected expression as to the nonliteral aspects of a software program can sometimes require a more extensive look at peripheral facts, the specific words arbitrarily selected by an author-programmer as his or her "expression" of the creative work are unquestionably protected as copyright-eligible subject matter. 17 U.S.C. §§ 101, 102(a). The incontrovertible fact controlling this case is that Google copied **verbatim** 11,000 lines of the very words authored by the programmers at Sun Microsystems (now Oracle) as "declaring code" for use with the Java platform.

The development of a new software product that will have substantial commercial value is a lengthy, complex and expensive undertaking.

Without reliable protection of the creative expression in a finished program, the willingness of software developers to invest time and resources into such an undertaking will be severely impaired.

A stable environment is essential to entrepreneurs and investors who would create software products that work by licensing other programmers to use their copyrighted works. This point is particularly critical today, where the U.S. and many other countries – China in particular – are racing to become the global leaders in numerous fields of technology that depend on computer software to advance the state of the art, including artificial intelligence, robotics, 5G communications and the use of quantum computing that promises to increase exponentially the speed and utility of computers.[2] Software is at the core of technological advances in many if not most industries today, and will be a key to maintaining our country's technology leadership in the 21st century. The law should encourage all of our citizens who have the intelligence, training and capital resources to continue solving the daunting problems of the modern world. Without adequate copyright protection, a crucial incentive for entrepreneurs and their investors to develop new

---

[2] This Court's jurisprudence on patent eligibility under 35 U.S.C. § 101, which many of the lower courts have interpreted to render numerous aspects of computer software ineligible for patent protection, has narrowed significantly the extent to which software can qualify as patentable subject matter under Title 35. In that light, it is particularly important for this Court to bear in mind the incentive structures that Congress intended and the role that copyright protection plays in that regard.

software systems and products that require interaction with programmers will be lost.

ARGUMENT

I. The Purpose of Copyright Protection Is to Encourage the Commitment of Time and Resources by Investors and Developers.

We urge this Court to bear in mind that the congressional impetus in the first instance for providing copyright protection to computer programs is rooted in the incentive mechanisms established by the U.S. Constitution.[3] In part, these constitutional incentives exist to limit free-riding on creative labor.

Economists have long acknowledged the deleterious effects of free riders on innovation by others. *See, e.g.*, FTC, *To Promote Innovation: The Proper Balance of Competition and Patent Law and Policy* 6 n.35 (2003) ("[I]f inventions can easily be duplicated or exploited by free riders, [the] resulting inability of inventors to capitalize on their inventions would lead to an environment where too few inventions are made.") (citing Roger E. Schechter & John R. Thomas, *Intellectual Property: The Law of Copyrights, Patents and Trademark*, § 13.4.1 at 288 (2003));. Charles F. Rule, *Technology Licensing and the Second American Revolution: Storming the Ramparts of Antitrust and Misuse, Before the John Marshall Law School* 5 (1985) ("Unless the 'free rider' problem is somehow addressed, those who might

---

[3] Art. I, §8, cl. 8 allows Congress to "promote the progress of Science and the Useful Arts" through the grant of exclusive rights. See, fn.5, *infra*, discussing the legislative history of Sections 101 and 102(a).

otherwise undertake risky and expensive R&D will not do so. Fewer technologies will be developed and consumers will face higher prices and fewer choices."); J. Thomas Rosch, Commissioner, FTC, Address at the Bates White Antitrust Conference (June 1, 2009) (detailing the development of free rider theories by leading economists and their acceptance by this Court).

Software development is particularly susceptible to free riding, as noted in *Apple Computer, Inc. v. Franklin Computer Corp.*, 714 F.2d. 1240, 1254 (3d Cir. 1983), wherein the Third Circuit made the obvious observation that "[t]he cost of developing computer programs is far greater than the cost of their duplication." The appellate court also acknowledged the plaintiff's "substantial evidence of the considerable time and money it had invested in the development of the computer programs in suit." *Id.*

Similarly, in *Apple Computer, Inc. v. Microsoft Corp.*, 799 F. Supp. 1006 (N.D. Cal. 1993), *aff'd in relevant part*, 35 F.3d 1435 (9th Cir. 1994), the trial judge noted the need to balance the dangers of free riders with the freedom of others to use the ideas inherent in a work. *Id.* at 1022. The judge observed, however, that using the ideas in a work does not extend to "identical copying, for this is the province of the ultimate free rider who makes a zero investment in creativity." *Id.* So too here: this case is not about the "ideas" in Java, it is about Google's "identical copying" of source code, a literary work.

The development of a complex software program involves extraordinary attention to detail and

a major investment of time and monetary resources. As an initial step, the developer must identify a specific need that will support the cost of developing and maintaining the product. Once the need and potential market have been identified, the developer must carry out a high level design of the program with sufficient detail to inform coding personnel specifically how each of the thousands or even millions of steps of the program will execute. Thereafter, the code itself must be written as source code and then compiled into machine code that the computer will recognize, after which will follow a period of debugging the code before it can be sold. Finally, the developer must market the product to potential users, educate users and monitor the user experience.

Moreover, many software programs rely on "libraries" of subprograms, another name for the "methods" that are at issue in this case. These subprograms are discrete units of prepackaged instructions that can be called upon to perform specific operations needed as the software is adapted to particular uses or required to address different operating variables. The deployment of libraries of commonly needed subprograms makes it much more efficient and versatile for a developer to adapt its programs to different customers, different industries, different operating conditions and different types of computers.

In addition, many software programs allow the customer to initiate specific subprograms to solve specific problems – *e.g.*, software used by engineers to design a building might include a subprogram to calculate the strength of a steel beam or a quantity of

concrete. Another common feature of many software programs includes the ability of the customer's own programmers to write their own subprograms to run on top of the developer's programs. A program for controlling autonomous cars, for example, might permit the maker of the car to write its own subprograms to control braking, taking into account such variables as weight, speed, weather and road surface conditions, etc.

Typically, subprograms are initiated by a programmer through an application programming interface or "API," *i.e.*, a set of precise words and phrases that allow programmers to connect subprograms with larger programs.

Therein lies USIJ's specific concern. Unless the source code for the APIs in a program is given the legal protection provided by the Copyright Act, it becomes far too easy for free riders to copy the APIs and then sell their own version of code that mimics operation of the original, thereby avoiding all of the costs of planning, developing and marketing the program from scratch as the original developer was required to do. The free rider is also able to wait until it becomes clear that a market has emerged and thereby avoid the risks of non-acceptance by customers, technological obsolescence and other risks assumed by the original developer. The fact that a free rider can sell its own knock-off more cheaply without having to recover all the development and marketing costs in its sales price, and the fact that the customer's users and developers are accustomed to using the library of APIs created in the original program allow the free rider to enjoy a substantial

and highly unfair advantage over the innovator who created the product.

It is extremely important for copyright law to protect these investments of time and resources. *See, e.g., Apple v. Franklin*, 714 F.2d at 1254 (noting "the jeopardy to Apple's investment and competitive position caused by Franklin's wholesale copying of many of its key operating programs"). As seen by those who would develop software programs that employ subprograms initiated through the use of APIs, the loss of copyright protection for APIs, as urged by Google and its amici, would be devastating. Without reliable copyright protection for the source code that makes up a compendium of proprietary APIs created by or for the creative entity whose efforts have given rise to the program in the first place, one of two undesirable consequences would likely follow – either the incentives that Congress thought it was creating through copyright protection of source code would be lost or severely impaired, or developers would be forced to abandon the efficiencies and elegance of using preconstructed blocks of code as part of their programs. Neither outcome should be forced on one of the most important industries in this country.

II. Google is a Free Rider As to the Java Declaring Code

The parties refer to the Java APIs at issue here as "declaring code" and to the prewritten instruction sets in the Java libraries as "implementing code." Google acknowledges that it copied significant portions of the "declaring code" for use by its own programmers and developers who would like to take advantage of the Java libraries that these developers

that already know how to use. Google Br. 9. Google says that it did not copy the Java implementing code, despite the fact that Google's sole purpose in copying the Java declaring code was to facilitate its developers doing so in ways that contravene the policies of the Java community.

Google refers to this unauthorized use of Java SE library as "reimplementing" the subprograms, as if giving the activity a different name changes the essential fact that the Google, at a tiny fraction of the cost that went into the original development of the Java SE library, wants to use something to which it is not entitled. Irrespective of what Google calls its activity, the Java declaring code is still a literary work that was authored by programmers at Sun Microsystems, Oracle's predecessor, and that falls squarely within the protective provisions of the Copyright Act – §§ 101, 102(a).

These sections were added to the Copyright Act in 1976 and 1980 to protect computer programs from copyists in same way that other categories of literary works are protected. They were enacted by Congress in recognition that computer software had become an important industry in its own right, and that the incentive structures envisioned by the U.S. Constitution for "promot[ing] the Progress of Science and Useful Arts" should be extended to the writing of source code for computer programs.[4] Java "declaring

[4] A thorough analysis of the many factors, pro and con, that were considered by Congress in enacting those amendments is set forth in a 1993 article written by Harvard law professor, Arthur Miller, entitled "Copyright Protection for Computer Programs, Databases, and Computer Generated Works: Is

code" is entitled under Sections 101 and 102(a) to the same level of copyright protection that is accorded any other literary work.

Nor does the declaring code at issue here define any aspect of Java's function, *i.e.*, the "ideas," embodied in Java SE. Rather, the declaring code is distinct from "function" in at least three ways. First, it is purely declarative – instructional text that informs human programmers how to combine separate programmers' contributions into an integrated whole. There are no verbs or other functional notations that establish instructions to be executed by the computer. Second, the parties to this case agree that Sun Microsystems (now Oracle) had available a nearly-infinite number of potential choices of words when it authored the declaring code, none of which were required by the various functional operations carried out by the implementing code. Whimsicality in the selection of words in a creative work is the antithesis of functionality. Third, the

---

Anything New Since CONTU?", 106 Harv. L. Rev. 977. Professor Miller was a member of the National Commission on New Technological Uses of Copyrighted Works ("CONTU"), which was created by Congress in 1974 to provide advice on the extent to which copyright should be applied to computer programs and the mechanisms for doing so. Professor Miller explains that Congress chose to designate computer programs as literary works rather than to try and develop an entirely new body of law that would take years to sort out in a rapidly evolving industry. He notes that "[c]ourts should continue to resolve questions of the copyrightability and scope of protection of computer programs by using the flexible principles that apply to literary works and not resort to arbitrary exclusion of particular software elements." *Id.* at 1036.

parties agree that Google could have written its own code that would be functionally-equivalent to Java SE, and indeed that companies other than Google have done so.[5]

III. The Literal Elements of a Program, *i.e.,* the Source Code, Are Copyrightable Subject Matter.

A significant portion of Google's brief is devoted to the contention that the structure, sequence and organization ("SSO") of the Java Standard Library are not subject to copyright protection. *See* Google Br. at 33 n.7 (citing *Computer Assocs. Int'l, Inc. v. Altai, Inc.*, 982 F.2d 693 (2d Cir. 1992)). An important distinction needs to be drawn between claims related to the nonliteral aspects of the Java program and the literal source code. Irrespective of the copyrightability of the SSO – *i.e.*, the nonliteral aspects of the Java SE libraries – the copying of source code is unquestionably a violation of Oracle's copyrights. Unlike the situation here, *Altai* did not involve the copying of source code and is not relevant to whether Google's decision to help itself to Java's declaring code was proper. Indeed, the court in *Altai* addressed a situation in which the defendant, after a discovering the actual copying of the plaintiff's source code by its employee, resorted to creating a clean room version of the same program to carry out the same functions as the originally copied code. This rewritten program also was charged with infringement based

---

[5] As the Federal Circuit acknowledged, Microsoft and Apple have products that provide similar operational functionality but do not use the Java code at all. *See* Oracle Br. at 31.

on the copyright doctrine of "substantial similarity" as to the nonliteral elements. The Second Circuit held that the rewritten program did not infringe based upon that court's own novel, highly structured and fact-dependent analytical methodology that examined separately each aspect of similarity. The appellate court acknowledged, however, the protection given to the source code: "It is now well settled that the literal elements of computer programs, *i.e.*, their source and object codes, are the subject of copyright protection." *Id.* at 702.

IV. Section 102(b) Is Not Applicable to Java's Declaring Code.

Google relies on 17 U.S.C. § 102(b) to argue that the declaring code it copied is not subject to copyright protection, because the code is a "method of operation" or in any event is "functional." Google Br. at 19. Section 102(b) is not applicable here. That provision provides only that an "idea, procedure, process, system, method of operation, concept, principle, or discovery" is ineligible for copyright, essentially codifying the distinction between "ideas" and the "expression" of ideas that has existed in copyright law since the first Copyright Act in 1790. *See, e.g.*, *Golan v. Holder*, 565 U.S. 302, 329 (2012) ("The idea/expression dichotomy is codified at 17 U.S.C. § 102(b)."); *Eldred v. Ashcroft*, 537 U.S. 186, 219 (2003) (same); *Feist Publ'ns, Inc. v. Rural Tel. Serv. Co., Inc.*, 499 U.S. 340, 356 (1991) (noting the "purpose" of Section 102(b) "is to restate … that the basic dichotomy between expression and idea remains unchanged"); *see also Holmes v. Hurst*, 174 U.S. 82, 86 (1899) (discussing common law origins of copyright

law, stating "[t]he right thus secured by the copyright act is not … the right to ideas alone, since in the absence of means of communicating them they are of value to no one but the author").

The declaring code at issue here does not meet the requirements of any aspect of §102(b). Google did not copy simply an underlying idea; Google copied the source code itself, which is the "expression" of the idea that underlies Java. If word-for-word copying of 11,000 lines from a literary work is not copyright infringement, it would be difficult to understand why we have a copyright law at all. And if source code is not protected, it would difficult to say what part of a program could be covered.

Google argues, again citing § 102(b), that its use of Java's declaring code is "functional," because that code is essential if programmers want to use the familiar Java SE Library to avoid writing their own source code. Google Br. 19. This argument is faulty. Even if Google's application developers were permitted to make whatever use they chose of the Java SE Library, that still would not strip the declaring code of the copyright protection that came into existence at the time the work was first authored and that will last until the copyright expires. 17 U.S.C. § 302. Google points to nothing in the particular words selected by the authors of the "declaring code" or in the organizational structure of the Library that was dictated by functional considerations. The authors at Sun selected arbitrarily the terminology of the code and the layout of the Java SE Library from a nearly unlimited number of possibilities, a process quite similar to the

manner in which the author of a play thinks up character descriptions and names, quirky behavior, small defining scenes in a story, and other expressions of a creative imagination. The declaring code falls squarely within the definition of a literary work that is protected against copying.

As a variation on the foregoing, Google argues that circumstances today are not the same as when the declaring code was first created and that widespread acceptance of the Java platform somehow gives others the right to make use of the code. Google Br. at 27. At the time of its creation, the Java declaring code was not constrained in any way by functional requirements; the authors arbitrarily selected the words used as source code for particular commands. That some of the Java code subsequently became well known and widely used by programmers today should not alter the protected status of Oracle's copyrights. To hold otherwise would endanger virtually all authors of software source code that is developed for the purpose of building a business based on the software.

At the time Sun set up the terms and conditions on which developers were permitted to use the Java platform and write software using the Java language, Sun did not know whether or to what extent its efforts would be successful. This is the uncertainty facing every company in the technology ecosystem that relies on intellectual property protection as the reasons for investing another dollar or another hour into what may turn out to be futile.

Worth noting is that Java's declaring code is not essential for a programmer to write applications

for an operating system such as Android, even today. Both Apple and Microsoft created their own mobile platform technology that does not use Java. Google wants for its programmers and developers to enjoy the use of the Java libraries without accepting the obligations that accompany such use. In other words, Google wants to benefit from the licensing arrangement that governs the use of the Java language and platform without the obligations that accompany such use. USIJ acknowledges the need to balance the incentives for creating new software with the ability of creative programmers to employ the most efficient methodology to achieving the purposes for which a program is to be put. But this need for balance applies far more to the non-literal aspects of a program than to the literal source code.

V. The Implicit Functionality of All Computer Code Is Not a Basis for Applying §102(b)

Google argues that the Java declaring code should be denied protection under § 102(b), because the code is intended to achieve functional objectives. Google Br. 19. This contention overstates the proper scope of Section 102(b) and its role in preserving the so-called "idea/expression" dichotomy. This Court recognized in *Mazer v. Stein*, 347 U.S. 201 (1954), that copyright protection is not lost merely because the protected work also has utility not related to the protected features. Stated differently, a functional object may also embody protectable "expression," and a proper legal analysis must separate function from expression, conceptually, to allow protection for the expressive component. Although it may be accurate to say that source code, by its very nature, is

"functional" in the sense that each line of code sets into motion one or more discrete instructions that carry out functional steps, any such "functionality" must be viewed in context – Congress chose to treat source code as a literary work. Without such context, the exclusions listed in § 102(b) would swallow all other considerations and there would be no protection for any object that also had a function, let alone for software.

The § 102(b) exclusions should come into play only when the number of alternatives available to the public is so small as to invoke the "merger doctrine. That is clearly not the case here, since Google acknowledges that both Apple and Microsoft have written their own software that does not use the Java source code.

The only cases that Google cites wherein the copying of source code was held to be lawful involved situations in which a small portion of code was required to initiate the operation of a separate peripheral hardware device. Thus, in *Sega Enters., Ltd.* v. *Accolade, Inc.*, 977 F.2d 1510 (9th Cir. 1993), the Ninth Circuit held that it was fair use for Accolade to use a portion of the object code in a Sega video game that was necessary to activate the game console. The Sega game console contained a lock-out feature that allowed the console to operate only upon seeing a bit-by-bit copy of the lockout code. To have held otherwise would have allowed Sega to limit how its hardware devices were used by their owners, clearly a competitive activity not protected by copyright. Critical to the decision is that Accolade copied only that portion of code essential to activate the Sega

console so that its games could compete with Sega's. *Id.* at 1526. *See also Lexmark Int'l* v. *Static Control Components, Inc.*, 387 F.3d 522 (6th Cir. 2004) (similar outcome as to small piece of lockout code necessary to allow a competitor's ink cartridges to operate with the plaintiff's printers).

The instant case, by contrast, involves the wholesale copying of a large portion of the Java APIs to facilitate the use of code that Google has no legitimate right to use and that is not required to achieve the commercial objectives that Google seeks.

Nor does the widespread adoption of the Java language by software developers alter its status vis-à-vis §§ 101, 102(a) and 102(b). If the source code authored by Sun did not fall within the exclusionary ambit of § 102(b) when it was written, nothing in copyright jurisprudence suggests that the exclusionary provision should suddenly spring to life after adoption by a specified number of users or the passage of a specified period of time. To hold otherwise would endanger virtually all authors of software source code that is developed at considerable expense for the purpose of building a business based on the software.

CONCLUSION

Congress has spoken with respect to the protection that is to be accorded computer programs. At the time Sun created the Java platform and wrote the source code for using it, Sun was writing on a blank slate – arbitrary in every way. Neither the APIs nor the structure, sequence and operation of the libraries of subprograms was dictated in any way by the

operational outcome they were intended to achieve. They are the essence of arbitrary creativity no different at bottom from the myriad of characters that comprise a Harry Potter novel. At the time, Sun set up the terms and conditions on which developers were permitted to use the Java platform and write software using the Java language, Sun did not know whether or to what extent its efforts would be successful. This is the uncertainty facing every company in the technology ecosystem that relies on intellectual property protection as the reasons for investing another dollar or another hour into what may turn out to be futile. It would be enormously unfair at this juncture to penalize Oracle for the success that Java has enjoyed.

Respectfully submitted,

DAVID BOUNDY
CAMBRIDGE TECHNOLOGY LAW LLC
P.O. Box 590638,
Newton, MA 02459

STEVEN C. CARLSON
ROBINS KAPLAN LLP
2440 West El Camino Real, Suite 100
Mountain View, CA 94040
(650) 784-4012

ROBERT P. TAYLOR
Counsel of Record
RPT LEGAL STRATEGIES PC
2443 Fillmore Street, Suite 332
San Francisco, CA 94115
(415) 447-3975
robert.taylor@rptstrategies.com

Counsel for Amici Curiae

APPENDIX

APPENDIX — MEMBERS OF USIJ

Member Companies

- EnterVault
- The Foundry
- MedicalCue
- Materna Medical
- Puracath Medical
- Precision Biopsy
- Prescient Surgical
- BioCardia
- Siesta Medical
- Autonomic Technologies
- Tallwood Venture Capital
- ExploraMed
- Fogarty Institute for Innovation
- Moximed
- Rearden Studios
- Zipline Medical
- Soraa
- Aegea Medical
- Array Photonics
- EarLens Corporation
- Louder Partners, LLC