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* PROFESSOR AND FORMER CONTU MEMBER ARTHUR R. MILLER IN SUPPORT OF RESPONDENT

THEODORE STEVENSON, III
Counsel of Record
CHELSEA A. PRIEST
PATRICK PIJLS
MCKOOL SMITH, P.C.
300 Crescent Court, Suite 1500
Dallas, TX 75201
(214) 978-4974
tstevenson@mckoolsmith.com

TABLE OF CONTENTS

Page

TABLE OF AUTHORITIES

Page(s)

Cases

Statutes and Regulations

Other Authorities

INTEREST OF *AMICUS CURIAE*[1]

Amicus Professor Arthur R. Miller currently is a University Professor at the New York University School of Law. He has devoted a substantial part of his professional career to the study of federal practice and procedure and is a co-author of Charles Alan Wright, Arthur R. Miller, Mary Kay Kane & Richard L. Marcus, *Federal Practice and Procedure*. Professor Miller has taught copyright for more than forty years at a number of law schools.

Most relevant here, Professor Miller participated in the debates, hearings, and negotiations that resulted in the Computer Software Copyright Act of 1980. President Gerald Ford also appointed Professor Miller to the National Commission on New Technological Uses of Copyrighted Works ("CONTU"), where he served on the Software Subcommittee. That Subcommittee investigated the relationship between software and copyright and drafted the relevant portion of the CONTU's Final Report, which both parties cite in their briefing. See National Commission on New Technological Uses of Copyrighted Works, *Final Report* (1979) [hereinafter "Report"].

In 1993, Professor Miller wrote an article defending and reiterating CONTU's recommendations and addressing the specific issue before this Court. See Arthur R. Miller, *Copyright Protection for Computer*

---

[1] The parties have consented to the filing of this brief and received timely notice of the intention to file. No counsel for any party authored this brief in whole or in part. No counsel or party made a monetary contribution intended to fund the preparation or submission of this brief.

Programs, Databases, and Computer-Generated Works: Is Anything New Since CONTU?, 106 Harv. L. Rev. 977 (1993).

Given his longstanding research on and participation in debates regarding copyrightability of software, Professor Miller submits this brief to offer the Court his professional academic perspective on the issues presented in this case.[2] He writes to expound on that defense of CONTU as it applies to the questions before the Court and reiterate the view that "program interfaces are treated no differently from other program features for copyright purposes." Miller, *supra*, at 1032.

SUMMARY OF ARGUMENT

Petitioner would have this Court declare that use of a copyrighted work is "necessary" whenever it is popular and an infringer wants to take advantage of that popularity. That is not the law. CONTU recommended and Congress agreed that software should be treated exactly as every other literary work. Neither functionality nor popularity exempts a creative work from copyright protection or otherwise excuses an infringer's actions. The decision below should be affirmed.

---

[2] *Amicus*'s affiliations and past positions are included for context only. *Amicus* submits this brief solely in his personal capacity and does not speak on behalf of any other person or institution. Professor Miller's complete biography is available at https://its.law.nyu.edu/facultyprofiles/index.cfm?fuseaction=profile.biography&personid=20130.

I. Nothing Has Changed Since CONTU Addressed Many of the Issues Before This Court

In 1974, Congress created CONTU with the mandate to counsel Congress on, among other things, "the impact of the computer on copyrighted works." Report, *supra*, at 5. CONTU was designed to incorporate views from a variety of perspectives; thus, it included representatives of authors and copyright owners, copyright users, and members of the public. *Id.* at 4. In addition, CONTU held hearings and seminars and sought advice from numerous stakeholders and experts, including, for example, the Massachusetts Institute of Technology and *The New York Times*. *Id.* at 6–7, App'x F, G.

CONTU issued its final report in 1978—a report that was the product of "three years of data collection, hearings, analysis, and deliberation." *Id.* at 1. CONTU's recommendation was "to make it explicit that computer programs, to the extent that they embody an author's original creation, are proper subject matter of copyright." *Ibid*. Two years later, Congress heeded CONTU's recommendation to "put the new wine into the old bottles," Miller, *supra*, at 979, and passed the Computer Software Copyright Act of 1980, Pub. L. No. 96-517, 94 Stat. 3015.[3] Given the rigor of CONTU's work, the report merits great weight in interpreting the Copyright Act. As peti-

[3] The statutory changes were minimal. CONTU recommended (1) defining "computer program" and (2) clarifying that although placing software into a computer creates a copy, whose unauthorized use would otherwise constitute piracy, a lawful possessor of the software is entitled to do so to use the software. Report, *supra*, at 12–13.

tioner concedes, the report is the "authoritative guide to congressional intent." Pet. Br. 24 (quoting *Sega Enters. Ltd.* v. *Accolade, Inc.*, 977 F.2d 1510, 1520 n.5 (9th Cir. 1992), *as amended* (Jan. 6, 1993)); see Ralph Oman, *Computer Software As Copyrightable Subject Matter:* Oracle v. Google*, Legislative Intent, and the Scope of Rights in Digital Works*, 31 Harv. J.L. & Tech. 639, 642 (2018) ("Congress adopted CONTU's recommendations wholesale, making its report particularly useful."). The "underlying premise" of CONTU's recommendation—and therefore the statutory text—is that computer programs "are entitled to copyright protection under the same principles that govern other literary works." Miller, *supra*, at 1008.

CONTU's description of the relationship between copyright and software remains dispositive of the questions before this Court today—in other words, nothing has changed since CONTU. See *id.* at 977. The need for protecting software, CONTU stated, has "grown proportionally with two related concurrent trends": the proliferation of uses to which software can be put, which follows from the proliferation of computers themselves. See Report, *supra*, at 10. Even as early as the 1970s, CONTU recognized the importance of copyright protection for software: The "underlying principle of copyright," CONTU proclaimed, is that the lesser the cost of duplicating a work, the greater the need for legal protection to maintain the "necessary incentive" to spur its creation. See *ibid.*; see also Miller, *supra*, at 1020 (noting that "subordinating the interests of those who create software in favor of those who copy software for their own commercial ends * * * inevitably undercut[s] the

incentives that Congress intended the Copyright Act to provide authors of programs").

When computers first appeared, a particular software program was of little use without a particular computer. The two went hand-in-hand. When hardware and software development evolved into separate spheres, however, software could be developed for use in any number of different computers. With the cost of replicating the software no longer tied to the cost of building the computer, then, software became relatively inexpensive to reproduce. CONTU saw the writing on the wall. See Report, *supra*, at 11 (observing that "[i]f present computer industry trends continue, it is all but certain that programs written by nonmachine manufacturers will gain an increasing share of the market"). The schism of computer and software design meant that the latter—"the product of great intellectual effort"—demanded protection. *Ibid.*

CONTU was prescient. Take this case as an example. Oracle, the world's second largest software company, developed software libraries that Google used to develop the world's most-used mobile operating system. Because Google could simply copy and paste Oracle's declaring code on which petitioner's operating system and applications rely, the cost to Google of replicating Oracle's work was minimal. Absent the continued protection by the Copyright Act that CONTU envisioned, "the product of [Oracle's] great intellectual effort" would be exposed to free copying.

CONTU also foresaw the exact type of dispute that arises in this case, distinguishing between "copyrightable computer programs and uncopyrightable

processes or methods of operation." *Id.* at 18. Because that distinction "does not always seem to 'shimmer with clarity,'" CONTU thought it "important that the distinction between programs and processes be made clear." *Id.* So CONTU described how it envisioned the "'idea-expression identity' exception" would operate in the software context: "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." *Id.* at 20. But "[w]hen other language *is* available, programmers are free to read copyrighted programs and use the ideas embodied in them in preparing their own works." *Ibid.* (emphasis original).[4] In so saying, CONTU found that "[t]he availability of alternative noninfringing language is the rule rather than the exception." *Id.* at 20 n.106.

Google suggests that CONTU's language represents a reversal of the traditional approach to merger, which focuses on the availability of alternatives to the creator of a computer program. Pet. Br. 30. Not so. As discussed above, CONTU's overarching recommendation was to apply traditional copyright principles to computer programs. Moreover, the specific text Google quotes was in the context of a section focused on "the fruits of intellectual labor" of the *original author*. Report, *supra*, at 20 (quoting *United*

[4] Contrary to petitioner's argument, CONTU did not intend this language to shift the merger analysis from the time of creation to the time of infringement. See Resp. Br. 29–30. The Copyright Act provides protection from the time of conception, and as will be described in more detail below, a copyrighted work does not lose its protection merely because it is popular.

States v. *Steffens*, 100 U.S. 82, 94 (1879)). "This exception would mean that a 'program' consisting of a very few obvious steps could not be a subject of copyright." Report, *supra*, at 20. Where there are unlimited choices available, merger does not apply. See *ibid.* (stating that merger applies when there are "a limited number of ways to express a given idea").

Pursuant to CONTU's work, Congress afforded software clear-cut protection in the Computer Software Copyright Act of 1980. CONTU's lesson is simple. "All that copyright protection for programs * * * means is that users may not take the works of others to operate their machines." *Id.* at 21. "In each instance," however, "one is always free to make the machine do the same thing it would if it had the copyrighted work placed in it, but only by one's own creative effort rather than by piracy." *Ibid.* This is the clear statement of Congress' intent in the field.

As explained below, *see infra* Parts II–III, petitioner's reasons for ejecting interfaces from the ambit of the Copyright Act or otherwise treating them differently are misplaced. But to the extent the policy arguments are given any merit, any changes in the scope of the Copyright Act should originate in Congress, which retains the power to revise its own statute. Here, not only has Congress done so—amending the Copyright Act dozens of times since its first enactment[5]— but in establishing CONTU in the first

[5] The first copyright statute enacted pursuant to the Intellectual Property Clause was the Copyright Act of 1790, 1 Stat. 124 (1790). Since then, Congress has passed four principal revisions. See Copyright Act of 1976, Pub. L. No. 94-553, 90 Stat. 2541 (1976); Copyright Act of 1909, Pub. L. No. 60-349, 35 Stat. 1075 (1909); Copyright Act of 1870, 16 Stat. 198 (1870); Copy-

place Congress signaled both its cognizance of the Act's importance and Congress's custody over the Act's future.

II. Congress Has Accorded Copyright Protection to Functional Works Since 1790

Petitioner and its *amici* focus much attention on the purported "functionality" of the copied code. Yet nothing in the Copyright Act renders interfaces (or any other computer software or program) per se uncopyrightable by virtue of their functionality. Indeed, Congress has consistently granted copyright protection to works with functional attributes. See Jane C. Ginsburg*, Four Reasons and a Paradox: The Manifest Superiority of Copyright Over Sui Generis Protection of Computer Software*, 94 Colum. L. Rev. 2559, 2567 (1994).

The Copyright Act has protected functional works since its first iteration in 1790, which granted copyright protection to maps and charts. See Act of May 31, 1790, ch. XV, 1 Stat. 124; see also *Burrow-Giles Lithographic Co.* v. *Sarony*, 111 U.S. 53, 57 (1884) ("This statute not only makes maps and charts subjects of copyright, but mentions them before books in the order of designation.").[6] Courts similarly have

---

right Act of 1831, 4 Stat. 436 (1831). The revisions to Title 17 of the United States Code since 1976 are too numerous to catalog, but they number in the dozens. See U.S. Copyright Office, Circular 92, *Copyright Law of the United States and Related Laws* v-xii (2016), https://www.copyright.gov/title17/title17.pdf. Just recently, Congress passed the Orrin G. Hatch–Bob Goodlatte Music Modernization Act, Pub. L. No. 115-264, 132 Stat. 3676 (2018).

[6] Google's *amici* analogize to maps in another way, by arguing that a color-coded map with a key was found not copyright-

granted protection to functional literary works, such as tables of contents and indices, even though the underlying material (state statutes) was not copyrightable. *See Howell* v. *Miller*, 91 F. 129, 138 (6th Cir. 1898).

That protection of functional expression continues to this day. *See Rockford Map Publ'rs* v. *Directory Serv. Co. of Colo.*, 768 F.2d 145 (7th Cir. 1985) (copyright infringement of plat maps). For example, courts have found copyright law applicable to protect architectural drawings, even though "[t]he intrinsic function of an architectural plan is to convey the information necessary to enable the reader to construct a building." *Eales* v. *Envtl. Lifestyles, Inc.*, 958 F.2d 876, 879–80 (9th Cir. 1992). Likewise, the Act protects useful works including "fact compilations, dictionaries, code books, encyclopedias, advertising, and 'how to' instruction manuals." Miller, *supra*, at 986.

Like maps, charts, and architectural works, software is copyrightable despite its functionality. The Copyright Act defines a computer program in terms that make clear that such programs are functional: "a set of statements or instructions to be used direct-

---

able in *Perris* v. *Hexamer*, 99 U.S. 674 (1878). But that case applied an early form of *scenes a faire*, because maps are expected to have keys; "[s]carcely any map is published on which certain arbitrary signs, explained by a key printed at some convenient place for reference, are not used to designate objects of special interest." *Id.* at 676. Yet "it has never been supposed that a simple copyright of the map gave the publisher an exclusive right to the use upon other maps of the particular signs and key which he saw fit to adopt." *Ibid.* Both the district court and the court of appeals (correctly) rejected Google's *scenes a faire* defense, Pet. App. 155a-157a, and Google does not make any *scenes a faire* argument in its briefing to this Court.

ly or indirectly in a computer *in order to bring about a certain result*." 17 U.S.C. § 101 (emphasis added). CONTU recommended—and Congress accepted—that software be copyrightable just like any other literary work. Treating software any differently or according it weaker copyright protection "explicitly contradicts CONTU's finding, enacted into law by Congress, that computer programs are to be treated as literary works and accorded corresponding copyright protection." Miller, *supra*, at 1022. And "interfaces are treated no differently from other program features for copyright purposes." *Id.* at 1032.

To be sure, the protection can be limited, but only when the idea and the expression merge, such that it is impossible to perform the function without also copying the idea. That was the issue in *Baker* v. *Selden*, 101 U.S. 99 (1879), where this Court explained that the public was free to use the accounting method Selden described, but "in using the art, the ruled lines and headings of accounts must necessarily be used as incident to it." *Id.* at 104. Thus, the forms at issue were not copyrightable. *Id.* at 107; see also Miller, *supra*, at 986 (noting that *Baker* distinguished between works "having their final end in application and use" and those "whose essence consists only in their statement," but "found the expression in *both* categories eligible for protection").

As respondent explains, merger analysis must be conducted *ex ante*, at the time the work is created, not *ex post*, at the time of infringement. *See* Resp. Br. 29 ("[E]very circuit to consider the issue has concluded that what matters are the options available to the author creating the original work."). Without careful adherence to the temporal distinction between an *ex*

ante and *ex post* merger analysis, there is risk that the law might erroneously deny copyright protection to works that have become so highly popular that the gamut of alternative expressions has, *ex post*, become limited.

CONTU debated extensively the merger doctrine and whether it could properly resolve the utility/non-utility problem. CONTU members engaged in endless discussions about the copyrightability of programs embedded in chips and programs used for entirely utilitarian purposes, such as the air-fuel mixture in automobiles. It concluded that copyright law, including merger, struck the proper balance for protecting works in the software context as copyright law does in other fields. See Report, *supra*, at 20.

In other words, a developer is free to write code that causes a computer to perform a novel function. The developer cannot use copyright to prevent anyone else from performing that function. But the developer can use copyright to prevent anyone from using the *same* expression—i.e., code—to perform that function. See *Apple Comput., Inc.* v. *Franklin Comput. Corp.*, 714 F.2d 1240, 1253 (3d Cir. 1983) ("If other programs can be written or created which perform the same function as [a developer's] operating system program, then that program is an expression of the idea and hence copyrightable."). The only exception is if the specific code is the only way to perform the function. See *id.* ("[T]his inquiry is no different than that made to determine whether the expression and idea have merged, which has been stated to occur where there are no or few other ways of expressing a particular idea.").

But as CONTU heard at public hearings, "[t]he availability of alternative noninfringing language is the rule rather than the exception." Report, *supra*, at 20 n.106. The following colloquy between Professor (then-Commissioner) Miller and the Vice President of the Association of Computing Machinery makes this clear:

> Commissioner Miller: How many different ways are there to produce a program * * * *?
>
> Dan McCracken: An infinite number in principle, and in practice dozens, hundreds.
>
> Commissioner Miller: So it is comparable to the theoretically infinite number of ways of writing *Hamlet*?
>
> McCracken: I believe so. It is not really true that there is a very restrictive way to write a program [which might make it] not copyrightable. I don't believe that at all.

Id. (quoting Transcript, CONTU Meeting No. 10, at 44–45). Professor Miller reiterated that view years later. See Miller, *supra*, at 1042 n.292 (describing "assurances that a given computer program ordinarily can be written in innumerable ways"); see also Anthony L. Clapes, *Software, Copyright, and Competition: The "Look and Feel" of the Law* 115–16 (1989) ("[T]he range of constructs and manipulations from which the program author selects are constrained more by the author's imagination than by the relatively weak constraints imposed by the nature of the desired result.").

As another author said, "there are usually many ways to implement any function." Steven R. Englund, *Idea, Process, or Protected Expression?: De-*

termining the Scope of Copyright Protection of the Structure of Computer Programs, 88 Mich. L. Rev. 866, 900 (1990). "Claims of necessity should thus be viewed with some skepticism." *Id.* Only when the code is "necessary to the purpose or function of a computer program" is the code unprotectable. Miller, *supra*, at 996 (citing *Whelan Assocs., Inc.* v. *Jaslow Dental Lab., Inc.*, 797 F.2d 1222, 1236 (3d Cir. 1986)).

This is as true for single instructions as it is for entire computer programs. Petitioner is wrong to claim that the Copyright Act "does not extend copyright protection to an isolated computer instruction." See Pet. Br. 23 (emphasis omitted). Just like the merger doctrine does not destroy the copyrightability of a sentence within a larger book, it does not preclude the copyrightability of a single line of computer code, so long as the same function could have been accomplished a different way.[7] *See Rockford Map Publ'rs*, 768 F.2d at 148–49 ("Dickens did not need to complete *Bleak House* before receiving a copyright; every chapter—indeed every sentence—could be protected standing alone.").

The question before the Court is whether copyright law properly resolves, consistent with sound policy objectives, the current dispute, if one assumes that Google copied the declaring code not because it was the only way to accomplish the desired function, but because the declaring code had become popular

[7] Putting aside, for the moment, that "[w]ords and short phrases such as names, titles, and slogans" are not copyrightable. 37 C.F.R. § 202.1(a). Google argues in this Court only that the calls developers use—not the declaring code at issue here—fall within that exception. Pet. Br. 9; Resp. Br. 23.

to the point that it is now standard language for developers, and thus the efficient solution. Existing copyright is situated to resolve this type of dispute: "a court should not declare a computer program element uncopyrightable simply because it can be conclusorily classified as 'efficient' or 'externally determined.'" Miller, *supra*, at 1009. Even if copying is the more "efficient" choice in a particular case, "the mere fact that the expression is efficient should not, without more, bar protection for original authorship in the programming context any more than it does with prose work." *Id.* at 1004. The court must "protect a program's expressive elements." *Id.* at 1009.

In short, functionality alone does not remove software—even interfaces—from Congress's command that software is copyrightable. In a case in which the accused infringer admits that the copyright owner's code is original, and admits that it could have achieved the same function without copying, there would not be merger.

III. If a Work Becomes a Standard Form of Expression, Copyright Law Does Not Deny Protection Via Copyrightability or a Fair Use Defense

Petitioner's justification for its copying is that it was "necessary" because it wanted developers to be able to use their preexisting knowledge of Java's API calls. Yet the "narrow[]" grounds on which petitioner urges the Court to decide the copyrightability issue, Pet. Br. 19, would upend the foundations of copyright law. That a particular work is "popular" does not mean it becomes uncopyrightable.

There is nothing new to Google's position—while "dressed in anti-monopolistic garb, at bottom it is an argument for standardization." Miller, *supra*, at 1019.[8] At its root is the notion that "popular systems are to lose their copyright protection merely because their popularity denominates them 'standards.'" *Id.* at 1020. This notion "would not be taken seriously if the copyrighted works were Steinbeck's *Grapes of Wrath*, Hemingway's *The Sun Also Rises*, or Miller's *Death of a Salesman*." *Ibid.*

This issue was discussed at length by CONTU, which understood that there are rare instances in which extrinsic public policy dictated denial of a copyright. At the time, some asserted that should be the fate of the Kennedy assassination films taken by Mr. Zapruder. But CONTU also understood that mere popularity—like mere utility—did not make something uncopyrightable. Works of genius and works of folly often become extraordinarily "popular." That does not mean, however, they become uncopyrightable. When CONTU said that normal copyright principles should apply to computer programs, implicit in that recommendation is the range of reasons that a particular work might get less protection or no protection at all: principles concerning fair use, the First Amendment, antitrust law, and generic works. Equally implicit, though, is the notion that if a computer program becomes popular, so be it, but that

---

[8] While Google's *amici* quote Professor Miller's 1967 statement expressing concerns as to copyright protection of underlying processes, Congress subsequently enacted Section 102(b) at his recommendation; Professor Miller noted that his concerns had "moderated" by 1993. See Miller, *supra*, at 981 n.10.

does not void the copyright *ex post*, and it does not prevent basic copyrightability.

Congress had good reason not to enact a popularity exception to copyright. As an initial matter, such an exception would lure the courts into a hopeless exercise in line-drawing: Just how popular must a work become before the creator is penalized with loss of protection? The exercise is even more impossible when, as in this case, substitute means of expression abound. *See* Pet. App. 47a ("Google also conceded that it could have written the APIs differently to achieve the same function.").

Nor does calling the copied material an "interface" aid in the line-drawing exercise. Though that term "may seem precise * * * it really has no specific meaning in programming. Certainly, it has no meaning that has any relevance to copyright principles." Miller, *supra*, at 1034 (footnote omitted). "Any limitation on the protection of 'interfaces' thus would be a limitation on the protection of much of the valuable expression in programs, and would invite plagiarists to label as an 'interface' whatever they have chosen to copy without permission." *Ibid.*

More importantly, a popularity exception would eviscerate the goal of the Copyright Act, which is to promote advancements. "The purpose of copyright is to create incentives for creative effort." *Sony Corp. of Am.* v. *Universal City Studios, Inc.*, 464 U.S. 417, 450 (1984). But advance too far and create widely desired work, petitioner warns, and risk losing copyright protection altogether; anyone will be able to copy the previously protected material by claiming that doing so was "necessary." That logic is head-scratching. "[P]romoting the unauthorized copying of interfaces

penalizes the creative effort of the original designer, something that runs directly counter to the core purposes of copyright law because it may freeze or substantially impede human innovation and technological growth." Miller, *supra*, at 1034.

No doubt, promoting works through copyright imposes an obligation on would-be users of those works to compensate the creator. Indeed, if the sole concern were public welfare in the abstract, *see* Pet. Br. 23, society would grant "free translation of literary works to make them comprehensible to non-English-speaking citizens," "free reproduction rights for books used in schools and other public institutions," and the like. See *id.* at 1029. But that is not the legislative scheme that Congress enacted.

Measuring the tradeoff between protecting incentives to create works and ensuring the public's ability to use them is "an exceptionally difficult, if not an impossible, task." *Id.* at 1021. The Court, fortunately, need not wrestle with such policy questions, however, because Congress already has. The Copyright Act contains numerous limitations on copyrights, where Congress has expressly tipped the scales in favor of more accessible use. See 17 U.S.C. §§ 107–22. The Act affords some level of copy privilege to:

- anyone for purposes of "criticism, comment, news reporting, teaching * * *, scholarship, or research";
- libraries and archives;
- educators;
- religious organizations;

- governments and not-for-profits related to agriculture and horticulture;
- retail outlets;
- places of public accommodation;
- disabled persons;
- not-for-profits benefiting veterans or fraternal organizations; and
- blind persons.

Id. §§ 107, 108, 110.

These carve-outs are in addition to other means of calibrating the "general good," like prescribing temporal limits on copyright and proscribing the copyright of ideas. See Report, *supra*, at 23; *see also* 17 U.S.C. § 102(b); *id.* §§ 301–05.

Moreover, following CONTU's recommendation, Congress adopted a carve-out specific to software. The Computer Software Copyright Act of 1980 permits a licensed user to download software onto a computer and make a new copy or adaptation in certain situations (not applicable here). 17 U.S.C. § 117. Importantly, this "circumscribed" exception "carries with it the negative implication that the statute's permission does not extend to commercial competitors." Miller, *supra*, at 1023.

What is the upshot of all this legislation? The statutory scheme, laden with exceptions for various classes of uses and users, evinces Congress's intent not to except works merely because they eventually become popular. *Cf.* Antonin Scalia & Bryan A. Garner, *Reading Law: The Interpretation of Legal Texts* 107 (1st ed. 2012) ("The expression of one thing im-

plies the exclusion of others."). Accordingly, the Court should not tamper with the legislation today. *Cf.* Report, *supra*, at 15 ("On no occasion in American history has copyright protection been withdrawn from a class of works for which it has been available."). "[C]ontinuing to treat interfaces no differently than any other aspect of computer programs seems eminently sound." Miller, *supra*, at 1034.

CONCLUSION

The judgment of the court of appeals should be affirmed.

Respectfully submitted.

THEODORE STEVENSON, III
Counsel of Record
CHELSEA A. PRIEST
PATRICK PIJLS
MCKOOL SMITH, P.C.
300 Crescent Court, Suite 1500
Dallas, TX 75201
(214) 978-4974
tstevenson@mckoolsmith.com

FEBRUARY 2020