Brigham Young University Prelaw Review

Volume 36 Article 5

4-2022

Forty-five Years Later: Redefining and Reclassifying Computer Code Under the Copyright Act of 1976

Trevor Oldham
Brigham Young University

Topher Hill
Brigham Young University, tophertap@gmail.com

Forty-five Years Later: Redefining and Reclassifying Computer Code Under the Copyright Act of 1976

Trevor Oldham and Topher Hill[1]

The Copyright Act of 1976 (Copyright Act) was written with the goal of promoting innovation and creativity among the general public.[2] Based on the technologies and information available at the time, the Copyright Act offered unprecedented protection to both current and future technologies. However, as time passed and technology advanced, courts found it difficult to establish precedents when dealing with certain provisions in the Act, particularly when it comes to computer code.[3] This difficulty made apparent that the Copyright Act needed to be amended. As time passed, new technologies emerged. As this occurred, the courts interpreted and reinterpreted the Copyright Act, often with conflicting results.[4]

The unrivaled explosion in popularity and influence of computer code on our lives in such a short time frame made it difficult for courts to fully cope with the legal nuances that need to be in place in order to ensure the proper protections are granted to the correct code. Currently, the courts do not have enough experience

1 Trevor Oldham is a senior studying Electrical Engineering. He plans to attend BYU Law starting in the fall of 2022. Christopher Hill is a senior majoring in Communications Studies. He plans to attend law school in the fall of 2023.

2 *See* U.S. Const. Art. I, §8, cl. 8.; *see also* Google v. Oracle, 141 S. Ct. 1183, 1195 (2021).

3 *See* Oracle Am., Inc. v. Google Inc., 750 F.3d 1339 (9th Cir. 2014) (showing judicial disagreement regarding the copyrightability of computer code).

4 *Id*.

with computer code to understand it fully, which limits their ability to make consistent and informed decisions. This leads to turmoil among major companies and small firms alike; all would like to have the code that they wrote be proprietary, but courts have in some cases inadequately protected and in other cases over enforced the limits of copyright on computer code.

Part of the confusion surrounding the copyrightability of computer code involves how it is included in the Copyright Act itself. Computer code is currently 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."[5] It is also classified as a "literary work."[6] The Copyright Act defines literary works as "works, other than audiovisual works, expressed in words, numbers, or other verbal or numerical symbols or indicia, regardless of the nature of the material objects, such as books, periodicals, manuscripts, phonorecords, film, tapes, disks, or cards, in which they are embodied."[7] There are clear discrepancies between what a literary work is and what computer programs are in this definition.

Given the limitations *Google v. Oracle* and other cases have exposed in the Copyright Act of 1976, computer code should be redefined and, as a whole, computer code should be removed from the literary works copyright classification and be granted a new classification. This redefinition and reclassification would clarify the complexities inherent in resolving disputes regarding fair use of different types of code, specifically Application Programming Interfaces (APIs) and declaring code.

In this article, we will argue that most code is copyrightable, and will explain the history of copyright protections for computer code. We will then introduce a new and expanded definition of computer program and propose a new classification for computer code within the Copyright Act. In the final section, we will discuss copyright protection and fair use under the new category and definition.

5 17 U.S.C. § 101.

6 *See id.*

7 *Id*.

I. Background

While current copyright law is governed by the Copyright Act of 1976, copyright protection in the United States started with the Constitution. Article I, Section 8, Clause 8 says, "The Congress shall have power . . . to promote the progress of science and useful arts, by securing for limited times to authors and inventors the exclusive right to their respective writings and discoveries."[8] This clause became the basis for both copyright and patent laws, the first of which—the Copyright Act of 1790—was passed just three years after the ratification of the Constitution.[9]

In 1976, Congress rescinded all previous copyright laws and passed a new copyright law. This law offered creators greater protection of their original works by creating different classifications, each of which can be protected by copyright. Congress intentionally included computer code in the Copyright Act of 1976 under the literary works category, though it was so new that they were unsure how to best protect it. They appointed a committee to research and provide recommendations to Congress on how to protect computer code under the new law. The current definition of computer code in the Copyright Act came from the recommendations of this committee.

After a few years, it became apparent that the Copyright Act of 1976 did not offer sufficient protection for certain types of works. Architectural works in particular were not protected under copyright law until Congress granted them a separate category within the Copyright Act in 1990. These works won copyright protection with the argument that "architecture performs a significant societal purpose, domestically and internationally."[10] They were afforded explicit, well-defined protection. While computer code is currently included in the Copyright Act, the definition and protections offered

8 U.S. Const. art. I, § 8, cl. 8.

9 *See* "Copyright Act of 1790," United States Copyright Office, https://copyright.gov/about/1790-copyright-act.html (last visited 25 Feb. 2022).

10 Copyright Amendments Act of 1990, Pub. L. No. 100-568, Tit. II, 102 Stat. 2853, 5 (1990).

to it are very ambiguous and subject to interpretation. This paper proposes ways to change that.

One of the major principles of computer code is abstraction. Computer code is written in a layered approach. For instance, to write a code that can add two numbers together, you need two different pieces. The first layer is the code that adds the number together. This is called a function or method. But, to save time while coding, programmers use "interfaces," which are shortcuts that allow them to run the "add two numbers" function, without having to rewrite the function every time. These interfaces have recently become the subject of a massive dispute between Google LLC and Oracle America, Inc., which will be discussed later.

The debate over computer code as intellectual property began in 1970 when computer makers began selling computer software separately from the hardware that ran it. Up to that point, computer code had been written for specific machines, but computer manufacturers quickly began to realize that they were spending much more money on the software development than they were spending on the hardware. They also realized that there was an enormous potential market for software. In 2020, the computer software industry was worth almost $390 billion.[11]

However, with the rise of software for sale came the rise of software piracy. This piracy took two forms: piracy by individuals and piracy by developers.[12] The former happens when a user copies the software for him- or herself, a friend, or a coworker, which was often done innocently. This is akin to "burning" someone a copy of a CD or DVD. The latter occurs when a software developer uses code developed by another developer and calls it his or her own. The changes to the Copyright Act that this paper proposes mostly affect

11 *See Business Software and Services Market Size, Share and Trends Analysis Report by Software, by Service, by Deployment, by End-use, by Enterprise Size, by Region and Segment Forecasts, 2021-2028,* Grand View Research (Apr. 2021), https://www.grandviewresearch.com/industry-analysis/business-software-services-market.

12 Sylvia Ann Matthews, *Copyrightability of Software: Piracy on the Waters of Protection,* 37 S.C.L. Rev. 679, 679-80 (1986).

piracy by developers. The next few paragraphs contain overviews of cases that shaped software copyright law.

In 1986, the Supreme Court heard *Whelan v. Jaslow*.[13] Jaslow, a dentist, envisioned a piece of software that would help dentists better manage their dental practices. After failing to create the software himself, he outsourced the development to a company called Strohl where Whelan worked. Since Whelan was the developer, when the software was finished, Strohl held the copyright. When Whelan changed jobs, she took the copyright with her. Jaslow later created another dental practice management software, which was more widely accessible and was marketed as a successor to the first product. Whelan sued Jaslow for copyright infringement.[14] The court found that the structure, sequence, and organization (SSO) were similar enough to constitute copyright infringement. This decision shows that even if component parts are not copyrightable, the organization of a computer program can be.

Several years later, a tech developer named Computer Associates (CA) created a scheduler that used a "translator."[15] This translator made it possible to run the same program on different operating systems, which made it much more accessible.[16] Altai, a rival company, wanted to develop a similar product. They recruited an employee from CA with knowledge of the program and the translator, though they claimed they did not know about his experience with the translator. The recruit copied 30% of the original translator code exactly as it appeared in CA's product. CA sued, and a court found in their favor, awarding them a large amount of money.[17] This case introduced the Abstraction-Filtration-Comparison (AFC) test, which

13 *See generally* Whelan Associates v. Jaslow Dental Laboratory, Inc., 797 F.2d 1222, 1248 (3rd Cir. 1986).

14 *See* Whelan Associates v. Jaslow Dental Laboratory, Inc., 797 F.2d 1222, 1225-1227 (3rd Cir. 1986).

15 Computer Associates International, Inc. v. Altai, Inc., 61 F.3d 695, 721(2nd Cir. 1995).

16 *Id.*

17 *See Computer Associates International*, 61 F.3d 695, 698-701.

courts use to determine whether two computer programs are substantially similar enough to constitute copyright infringement.[18]

In 2021, the Supreme Court decided *Google LLC v. Oracle America, Inc*, which examined the fair use of copyrighted code.[19] When Google acquired and finished the development of the Android operating system. When the program was complete, it used 11,500 lines of an API developed by Java SE. This API allowed programmers to work in a language they already knew, rather than learning a new programming language if they wanted to develop Android apps. Oracle claimed this was copyright infringement; Google claimed fair use,[20] the legal doctrine that permits the use of a copyrighted work as long as it meets certain conditions.[21] Unlike the first two courts that heard the case, the Supreme Court did not address the issue of copyrightability of APIs, but ruled in Google's favor, arguing that Google's use of the code met the standards for fair use.

For most people in the coding community, this ruling came as a relief. A narrow definition of fair use of computer code would have made development prohibitively expensive for all but the richest of developers, but a broad understanding of fair use promotes creativity and innovation. However, because code is complex, case law is not sufficient to address all of its multifaceted issues. The new definition and reclassification provide a statutory framework which will simplify judicial decisions about the fair use of computer code.

II. Proposed Definition

When the Copyright Act's current definition of computer program was written in the 1970s, software development was still a new field, and lawmakers were certainly not experts on the topic. The definition introduced in the Copyright Act of 1976 is, "a set of statements or instructions to be used directly or indirectly in a computer in

18 *See Computer Associates International*, 61 F.3d 695, 721.

19 Google LLC v. Oracle America, Inc., 141 S.Ct.1183, 1186 (2021).

20 *Id.*

21 *See* James S. Heller & Sarah K. Wiant, Copyright Handbook 9-14 (1884).

order to bring about a certain result."[22] This all-encompassing definition has functioned for the last 45 years, but it should be expanded to reflect a greater understanding of the nuance of computer code. We propose the following definition:

> A 'computer program' is a set of original statements or instructions, literal and nonliteral, written with particular sequence, structure, and order to be used directly or indirectly by a computer to bring about a desired result.

This definition is informed by several landmark decisions about the copyrightability of software[23] and by several key principles of copyright law (the "idea-expression dichotomy" and "scenes a faire") influenced this definition. These considerations will provide increased protection for both creators and those making fair use claims.

The proposed definition includes the sequence, structure, and order of the proposed computer code. In *Whelan v. Jaslow*, a third circuit court decided that the sequence, structure, and organization (SSO) of a computer code are important in determining copyright infringement.[24] While some elements of code must occur in a certain order, much of a computer program is flexible in terms of the order in which a programmer chooses to organize it. In *Whelan v. Jaslow*, the court found there was copyright infringement because the SSO of the two codes was substantially similar.[25]

The sequence, structure, and organization of a program are expressions of the programmer and should be copyrightable. For this reason, they are included in the proposed definition. In particular, this allows for the explicit protection of things like software application programming interfaces, or API's. APIs are written to access

22 17 U.S.C. § 101.

23 *See* Whelan Associates v. Jaslow Dental Laboratory, Inc., 797 F.2d 1222 (3rd Cir. 1986); *see also* Google v. Oracle, 141 S.Ct. 1183 (2021); *see also* Computer Associates International, Inc. v. Altai, Inc., 61 F.3d 5 (2nd Cir. 1995).

24 *See generally* Whelan Associates v. Jaslow Dental Laboratory, Inc., 797 F.2d 1222, 1248 (3rd Cir. 1986).

25 *Id.*

different methods or functions within the program. The general idea of accessing prewritten functions within a program should not and in fact cannot be copyrighted, however the structure, sequencing, organization, and naming used in the API are most certainly a unique expression of this idea and should thus be afforded copyright protections. This would have benefited the courts in cases such as *Google v Oracle*, as a key point in determining the outcome of the case was whether or not APIs are copyrightable.

Structure, sequence, and organization are only three examples of nonliteral aspects of code. More examples of nonliteral aspects of code include "screen displays, menu structures[,] and user interfaces."[26] *Computer Associates v. Altai* was a landmark case because it afforded copyright protection for non-literal aspects of the computer code beyond SSO. These non-literal elements of code deserve copyright protection because they often drive much of the base innovation and invention that is done when developing code. For example, user interfaces and menu structures are both vital to most end-user experiences, and a lot of development time goes to ensuring that both of these meet the best criteria. Were these not protected by copyright, there would be little incentive to create new ones, and the market as a whole would suffer.

Another important facet of the definition is the word "original," which is inspired by the copyright doctrine of the idea-expression dichotomy. Under copyright law, an expression of an idea can be copyrighted, but the idea itself cannot be. For example, a program that listens to and identifies songs is copyrightable, but a programmer cannot obtain a copyright for the idea of a program that listens to and identifies a song. If another programmer decides to write a similar program, so long as the code is original and the SSO is substantially different, there is no copyright infringement. In fact, for any work to qualify for copyright protection, it must be an original work of authorship. For this reason, the word "original" appears in the proposed definition.

26 *The Software Protection Paradigm*, Smith Gambrell Russell, https://sgrlaw.com/briefings/474/ (last visited 25 Feb 2022).

While not new in this definition, the phrase "set of statements or instructions" is also important because a single line of code that fulfills a single task is not copyrightable. There are only so many expressions of the idea behind that line of code. This is an application of the merger doctrine. This doctrine states that where the idea behind something and the expression of that idea are one and the same, the idea and the expression are not copyrightable. In cases where the size, scale, and scope of computer code are limited and it is demonstrably difficult to implement in unique ways, then the code should not be granted copyright protection. As the code becomes more complex and accomplishes more tasks, the argument for its copyrightability gains proportional traction. This must be tackled on a case-by-case basis. However, cases in which copyright is not granted should be rare, as the majority of copyright seekers have implemented a sufficiently complex code base that will be granted copyright protection.

III. Effects of New Definition on Copyrightability of Software

The proposed new definition will go a long way to help resolve some of the confusion surrounding copyright and computer code. Currently, computer code is 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."[27] To those familiar with coding, this implies that close to any program one writes is copyrightable. However, *Google v. Oracle* made it all the way to the Supreme Court and even they refused to make a commitment as to the copyrightability of the code.[28] In the specific case of *Google v. Oracle*, the interfaces are of course copyrightable. Even though they themselves are not operating methods, but instead calling the methods, they are still "statements that bring about the result" of calling the methods.[29] If the courts at every level of *Google v. Oracle* had been working from

27 17 U.S.C. § 101.

28 *See* Google v. Oracle, 141 S.Ct. 1183, 1186 (2021).

29 17 U.S.C. § 101.

the new definition, it is likely that there would have been consensus about the copyrightability of the API, rather than confusion.

There are certain dangers associated with arguing that some code is copyrightable, and some code is not. If any firm line is drawn between code that is and is not copyrightable, there is the potential to inhibit further innovations and inventions. The code that is deemed uncopyrightable will suffer as incentive to create innovations on it would drop to zero. Conversely, if all code were to be copyrightable, then it would become prohibitively expensive to develop technology, as expensive licensing fees for generic code would impede progress. This would go against the very nature of the Copyright Act, which seeks to foster technological advancement by ensuring owners of intellectual property the rights to whatever they create.[30] However, this is not to say that a line should not be drawn. As discussed above, there is code that is inherently uncopyrightable. The proposed definition makes it clear which code falls into both the copyrightable and uncopyrightable categories.

IV. Proposed Reclassification

To clarify the law surrounding copyrightability of software and the accepted use of copyrighted code by someone other than the copyright holder, a new classification should be added to the Copyright Act of 1976 granting computer code its own classification. This classification would include the expanded definition of "computer program" and provide guidance on matters of fair use.

Creating such a category under the Copyright Act is not unprecedented. In 1990, architecture was added as a new legal category to the Copyright Act. One of the main considerations in favor of adding this protection was that "Architecture plays a central role in our daily lives, not only as a form of shelter or as an investment, but also as a work of art. It is an art form that performs a very public, social purpose."[31] This, along with the multitude of amendments made to the Act since its inception, shows a pattern of modifying the

30 *See* U.S. Const. Art. I, §8, cl. 8.

31 Copyright Amendments Act of 1990, Pub. L.

Copyright Act when the current provisions are either insufficient or incorrect, which is the current situation with computer code. Placing computer code into its own legal category would provide assurance of copyright protections for computer code as well as provide clarification regarding copyright infringement and fair use.

V. Copyright Infringement and Fair Use

There are also some aspects of computer code that are unavoidable. For instance, it is common practice to abstract sections of the code into smaller blocks called "functions" or "methods" that perform individual tasks. This is done to improve code readability and to allow the same section of code to be referenced throughout the program without the need to rewrite the same block of code multiple times. It is also common practice to abstract meaningful numbers and characters that a program uses to fulfill its task into "variables." Variables are used in almost every single computer program, and they act as one of the main pillars that allow any program to execute its task successfully and efficiently. Within copyright law, there is a doctrine called "scenes a faire," which protects similarities that occur naturally because of a situation. In *Cain v. Universal Pictures*, a novelist sued a movie production company for copyright infringement over a scene in a movie.[32] In both the movie and the book, characters find refuge from a storm in a church. The court found that other than hiding in a church, there were few similarities between the two scenes, and therefore, there was no copyright infringement, only scenes a faire. Similarly, variables, functions, and other elements of code that occur frequently across programs should be considered scenes a faire. A developer may not allege copyright infringement for using universal elements.

The fair use doctrine also impacts developers and programmers. In fact, fair use was the point at issue in the Supreme Court's decision in *Google v. Oracle*. According to the Copyright Act, "Fair use is a legal doctrine that promotes freedom of expression

32 *See generally* Cain v. Universal Pictures Co., 47 F. Supp. 1013 (S.D. Cal. 1942)

by permitting the unlicensed use of copyright-protected works in certain circumstances."[33] These circumstances are typically limited to "criticism, comment, news reporting, teaching, scholarship, and research."[34] However, the nature of computer code once again denies the sufficiency of such limitations. While computer code can be used in teaching, scholarship, and research, the vast majority of it is used to bring about a result in a computer or to accomplish a series of tasks.

To copy any section of code that has been granted copyright protection would inherently infringe on the copyright. Under the new definition of computer code it is acknowledged that there exist unique and original ways the code could have been written. It seems apparent, then, that fair use of copyrighted code should almost never be granted. Such a strict limitation on the fair use of code, however, would limit the collaborative nature of programming as a whole, causing a decrease in innovation. While it would increase incentive to write and seek a copyright for original code, it does not guarantee that the quality of the new code is any better than what was previously written.

Therefore, when evaluating fair use of software, courts should continue to use the four criteria used to evaluate other cases of fair use: the "purpose and character of the use," the "nature of the copyrighted work," the "amount and substantiality of the portion used in relation to the copyrighted work," and the "effect of the use upon the potential market for or value of the copyrighted work."[35] In making this determination, courts should weigh the "amount and substantiality of the portion used in relation to the copyrighted work" the most heavily, followed by the "purpose and character of the use," and then the other two.

VI. Conclusion

Computer code and computer programs contribute significantly to American culture and society, influencing everything from how we

33 17 U.S.C. § 107.

34 *Id*.

35 *Id*.

work, to how we entertain ourselves, to how we sleep and wake up. Because of computer code's prevalence, it is in the best interest of the United States to foster innovation and problem-solving, a task which has been left up to copyright. Congress passed the Copyright Act when computer programming was in its infancy, and since 1976, the field of computer code has grown and expanded to encompass much more than legislators could even imagine at the time.

The definition of computer code is broad and not particularly useful for the courts trying to make decisions about the copyrightability of computer code. The proposed definition would clarify what is and is not copyrightable and better inform the courts about what constitutes copyright infringement. Code currently falls into the literary works category of copyright protection, but that classification is inaccurate and has led to confusion and ambiguity. To reduce this ambiguity and provide programmers and judges with clearer direction, computer code should receive its own classification. Google's victory earlier this year was celebrated by programmers all over the country, but much of the arguing could have been avoided with clear guidance, which would be best accomplished by redefining and reclassifying computer code under the Copyright Act of 1976.