IN THE UNITED STATES DISTRICT COURT

FOR THE NORTHERN DISTRICT OF CALIFORNIA

ORACLE AMERICA, INC.,

Plaintiff,

v.

GOOGLE INC.,

Defendant.
_______________________________________/

No. C 10-03561 WHA

ORDER PARTIALLY GRANTING AND PARTIALLY DENYING DEFENDANT'S MOTION FOR SUMMARY JUDGMENT ON COPYRIGHT CLAIM

INTRODUCTION

In this patent and copyright infringement action involving features of Java and Android, defendant moves for summary judgment on the copyright infringement claim. With one exception described below, the motion is **DENIED**.

STATEMENT

Oracle America Inc. accuses Google Inc. of infringing some of Oracle's Java-related copyrights in portions of Google's Android software platform. Specifically, Oracle accuses twelve code files and 37 specifications for application programming interface packages. The Java technology and the basics of object-oriented programming were explained in the claim construction order (Dkt. No. 137). An overview of application programming interfaces and their role in Java and Android is provided here.

1. APPLICATION PROGRAMMING INTERFACES (APIs).

Conceptually, an API is what allows software programs to communicate with one another. It is a set of definitions governing how the services of a particular program can be called upon, including what types of input the program must be given and what kind of output will be returned. APIs make it possible for programs (and programmers) to use the services of a given program without knowing *how* the service is performed. APIs also insulate programs from one another, making it possible to change the way a given program performs a service without disrupting other programs that use the service.

APIs typically are composed of "methods," also known as "functions," which are software programs that perform particular services. For example, a programmer might write a software program method *A*, which calculates the area of a room when given the shape and dimensions of the room. A second programmer then could write a program method called *B*, which calculates the square footage of an entire house when given the shape and dimensions of each room. Rather than reinventing a new way to calculate area, the second programmer could simply write an instruction in *B*, "for each room, ask program *A* to calculate the area; then add all of the return values," using, of course, real programming language. As long as the second programmer knows what *A* is named, what type of "arguments" *A* must be given as inputs, and what return *A* outputs, the second programmer can write a program that will call on the services of *A*. The second programmer does not need to know how *A* actually works, or is "implemented." There may in fact be multiple ways to implement *A* — for example, different ways to divide an oddly shaped room into geometric components — and the first programmer may refine his implementation of program *A* without disrupting program *B*.

A method must be defined before it can be used. A method can be "declared" (*i.e.*, defined) in a programming language such as Java by stating its name and describing its argument(s) and return(s) according to syntax conventions. Once a method has been declared, it can documented and implemented. *Documentation* is not code; it is a reference item that provides programmers with information about the method, its requirements, and its use. An *implementation* is code that actually tells the computer how to carry out the method. Often, as in the example above, multiple implementations are possible for a given method.

In object-oriented programming, methods are grouped into "classes." A class file typically contains several methods and related data. Classes, in turn, are grouped into "packages" known as API packages. Whereas a class generally corresponds to a single file, a package is more like a folder or directory providing an organizational structure for the class files. A given API package could contain many sub-packages, each with its own classes and sub-classes, which in turn contain their own methods. These elements generally are named and grouped in ways that help human programmers find, understand, and use them. A well developed set of API packages, sometimes called a "class library," is a powerful tool for software developers; as such, it can help attract developers to a particular platform.

The specification for a class library — much like the specification for an automobile — is an item of detailed documentation that explains the organization and function of all packages, classes, methods, and data fields in the library. The class library specification for a given software platform, sometimes called the "API Specification" is an important reference item for programmers. In order to make effective use of the APIs, a programmer must be able to find the portion of the specification describing the particular package, class, and method needed for a given programming task.

2. JAVA AND ANDROID.

As explained in previous orders, Java and Android are both complex software platforms with many components. For example, the Java platform includes the Java programming language, Java class libraries, the Java virtual machine, and other elements. The Java programming language has been made freely available for use by anyone without charge. Both sides agree on this. Other aspects of the Java platform, however, such as the virtual machine and class libraries, allegedly are protected by patents and copyrights.

The Android platform uses the Java programming language; thus, software developers already familiar with the Java language do not have to learn a new language in order to write programs for Android. In contrast to Java, the Android platform uses the Dalvik virtual machine instead of the Java virtual machine, provides Android class libraries, and has other non-Java components. The Java platform has been used primarily on desktop computers, but it also has been used on cell phones and other mobile computing devices. Android, on the other hand, was designed specifically for mobile devices. Java and Android compete in the market for mobile computing software.

According to Oracle, Android is an unauthorized and incompatible Java implementation. The Java platform and the Android platform each includes class libraries with more than one hundred API packages. Android allegedly supports some, but not all, of the APIs defined for the Java platform. Thus, some programs written for the Java platform will not run properly on the Android platform, even though both use the Java language. Similarly, the Android platform allegedly includes additional APIs that are not part of the Java platform. Thus, some programs written for the Android platform will not run properly on the Java platform, even though they are written in the Java language. This so-called fragmentation undermines the "write once, run anywhere" concept underlying the Java system and supposedly damages Oracle by decreasing Java's appeal to software developers.

3. TERMINOLOGY.

The term API is slippery. It has been used by the parties and in the industry as shorthand to refer to many related concepts, ranging from individual methods to code implementations to entire class libraries and specifications. In this order, the term API will be used only to refer to the abstract concept of an application programming interface. API *documentation* (*e.g.*, the specification for a class library or for an API package within the library) and API *implementations* (*e.g.*, the source code relating to a particular method within a class file) will be referenced as such. Having clarified this linguistic point, this order proceeds to consider the specific items accused of copyright infringement in this action: twelve files of code, and 37 API package specifications.[1]

[1] At the hearing, counsel for Oracle suggested that Google's code *implementations* of the 37 API package specifications are unauthorized derivative works. This theory was disclosed by Oracle during discovery (Dkt. No. 263-3 at 11), but it was dismissed summarily in Google's summary judgment brief (Br. 9). Because the briefing does not address this theory, it will not be addressed herein.

ANALYSIS

Summary judgment is proper when "there is no genuine dispute as to any material fact and the movant is entitled to judgment as a matter of law." FRCP 56(a). Where the party moving for summary judgment would bear the burden of proof at trial, that party bears the initial burden of producing evidence that would entitle it to a directed verdict if uncontroverted at trial. *See C.A.R. Transp. Brokerage Co. v. Darden Rests., Inc.*, 213 F.3d 474, 480 (9th Cir. 2000). Where the party moving for summary judgment would not bear the burden of proof at trial, that party bears the initial burden of either producing evidence that negates an essential element of the non-moving party's claims, or showing that the non-moving party does not have enough evidence of an essential element to carry its ultimate burden of persuasion at trial. If the moving party satisfies its initial burden of production, then the non-moving party must produce admissible evidence to show there exists a genuine issue of material fact. *See Nissan Fire & Marine Ins. Co. v. Fritz Cos.*, 210 F.3d 1099, 1102–03 (9th Cir. 2000).

Copyright protection subsists in "original works of authorship fixed in any tangible medium of expression." 17 U.S.C. 102. In order to succeed on a copyright infringement claim, a plaintiff must show that it owns the copyright and that the defendant copied protected elements of the work. Only expressive elements that are "original," *i.e.*, independently created, are protected. Copying can be proven by showing that the alleged infringer had access to the copyrighted work and that the protected portions of the works are substantially similar. *Jada Toys, Inc. v. Mattel, Inc.*, 518 F.3d 628, 636–37 (9th Cir. 2008). Google advances a number of arguments why Oracle supposedly cannot prove all or part of its copyright infringement claim. Google is entitled to summary judgment on only one issue.

1. THE CODE FILES.

Regarding the twelve code files at issue, Google argues that its alleged copying was *de minimis* (Br. 22–24). In the copyright infringement context, "a taking is considered *de minimis* only if it is so meager and fragmentary that the average audience would not recognize the appropriation." *Fisher v. Dees*, 794 F.2d 432, 434 n.2 (9th Cir. 1986). The extent of the copying "is measured by considering the qualitative and quantitative significance of the copied portion in relation to the plaintiff's work as a whole." *Newton v. Diamond*, 388 F.3d 1189, 1195 (9th Cir. 2004).

Here, the parties dispute what constitutes the plaintiff's work as a whole. Google argues that its alleged copying should be compared to the entire Java platform, which Oracle registered as a single work (Br. 22–23; Kwun Exh. B). Oracle, on the other hand, argues that each of the twelve code files at issue is a separate work for purposes of this analysis (Opp. 23–24). Google has not shown that the Java platform is the proper basis for comparison. Google cites two provisions of the copyright regulations, but neither one supports Google's position (Reply Br. 12–13).

First, Google misapplies 37 C.F.R. 202.3(b)(4)(i)(A). That provision states: "For the purpose of registration on a single application and upon payment of a single registration fee, the following shall be considered a single work: (A) In the case of published works: all copyrightable elements that are otherwise recognizable as self-contained works, that are included in a single unit of publication, and in which the copyright claimant is the same." The plain meaning of this provision is that when a single published unit contains multiple elements "that are otherwise recognizable as self-contained works," the unit is considered a single work *for the limited purpose of registration*, while its elements may be recognized as separate works for other purposes. Courts considering Section 202.3(b)(4)(i)(A) generally agree with this interpretation. *See, e.g.*, *Tattoo Art, Inc. v. TAT Int'l., LLC*, --- F. Supp. 2d. ---, No. 2:10cv323, 2011 WL 2585376, at *15–16 (E.D. Va. June 29, 2011) (interpreting Section 202.3(b)(4)(i)(A) to codify the principle that "the copyrights in multiple works may be registered on a single form, and thus considered one work *for the purposes of registration* while still qualifying as separate 'works' for purposes of awarding statutory damages"). Google relies on Section 202.3(b)(4)(i)(A) to show that the code files comprising the Java platform should be treated collectively as a single work *for purposes of an infringement analysis*. This interpretation is contrary to the plain language of the regulation and is not supported by any cited authority.

Second, Google cites to 37 C.F.R. 202.3(b)(3), which concerns continuation sheets. Continuation sheets are used "only in submissions for which a paper application is used and where additional space is needed by the applicant to provide all relevant information." 37 C.F.R 202.3(b)(3). The regulation requires use of a separate continuation sheet "to list contents titles, i.e., titles of independent works in which copyright is being claimed and which appear within a larger work." *Ibid.* It does not, however, state that a failure to list individual titles precludes an applicant from later asserting those titles as separate works in infringement litigation. Nor does it address works registered by means other than a paper application. Google does not provide enough factual context to show that Section 202.3(b)(3) applies to the works at issue, and Google does not explain how it might bear upon the dispute at hand, even if it does apply.

Google cites no other authority. This order finds that, at least on the present record, Google has not shown that the Java platform as a whole is the work to which Google's alleged copying should be compared. Because all of Google's *de minimis* arguments compare the accused material in the code files to the entire Java platform as a whole, this order need not consider the *de minimis* question further.

2. THE API PACKAGE SPECIFICATIONS.

Regarding the 37 API package specifications at issue, which are reference items and not code, Google argues that the only similarities between the accused works and the asserted works are elements that are not subject to copyright protection. Google, however, does not specify which elements it views as similar. Google instead presents an array of theories why various *categories* of specification elements do not merit copyright protection. With one exception, this broad categorical approach fails. Google's other arguments regarding the API package specifications — that the disputed works are not virtually identical or substantially similar, and that Google's alleged copying was fair use — also fail to earn summary judgment for Google.

A. Names.

"Words and short phrases such as names, titles, and slogans" are "not subject to copyright." 37 C.F.R. 202.1(a); *Planesi v. Peters*, No. 04-16936, slip op. at *1 (9th Cir. Aug. 15, 2005). Google argues that "the names of the Java language API files, packages, classes, and methods are not protectable as a matter of law" (Br. 17). This order agrees. Because names and other short phrases are not subject to copyright, the names of the various items appearing in the disputed API package specifications are not protected. *See Sega Enters. Ltd. v. Accolade, Inc.*, 977 F.2d 1510, 1524 n.7 (9th Cir. 1992) ("Sega's security code is of such de minimis length that it is probably unprotected under the words and short phrases doctrine.").

Oracle argues that it is entitled to a "presumption that the names in the Java API specifications are original" (Opp. 14). Not so. The decision Oracle cites for this proposition shows only that a certificate of registration may entitle its holder to a presumption of copyright validity as to the registered work. *Swirsky v. Carey*, 376 F.3d 841, 851 (9th Cir. 2004) (citing 17 U.S.C. 410(c)). Oracle cites no authority requiring a presumption of *originality* as to *specific elements* of a registered work.

Oracle also argues that its selection and arrangement of component names within the specifications is entitled to copyright protection (Opp. 15). This argument is non-responsive. Copyright protection for the selection and arrangement of elements within a work is a separate question from whether the elements themselves are protected by copyright. In finding that the names of the various items appearing in the disputed API package specifications are not protected by copyright, this order does not foreclose the possibility that the selection or arrangement of those names is subject to copyright protection. *See Lamps Plus, Inc. v. Seattle Lighting Fixture Co.*, 345 F.3d 1140, 1147 (9th Cir. 2003) ("[A] combination of *unprotectable elements* is eligible for copyright protection only if those elements are numerous enough and their selection and arrangement original enough that their combination constitutes an original work of authorship.") (emphasis added).

Having found that the names of the various items appearing in the disputed API package specifications are not protected by copyright on account of the words and short phrases doctrine, this order need not consider Google's alternative theory that the names are unprotected because they are the result of customary programming practices.

B. *Scenes a Faire* and the Merger Doctrine.

"Under the *scenes a faire* doctrine, when certain commonplace expressions are indispensable and naturally associated with the treatment of a given idea, those expressions are treated like ideas and therefore not protected by copyright." *Swirsky v. Carey*, 376 F.3d at 850. "Under the merger doctrine, courts will not protect a copyrighted work from infringement if the idea underlying the copyrighted work can be expressed in only one way, lest there be a monopoly on the underlying idea." *Satava v. Lowry*, 323 F.3d 805, 812 n.5 (9th Cir. 2003).

Google argues that "[t]he API declarations are unprotectable *scenes a faire* or unprotectable under the merger doctrine" (Br. 14). Google, however, does not specify what it means by "API declarations." Google applies this argument to *all* of "[t]he allegedly copied elements of the Java language API packages," providing only a few examples: "the names of packages and methods and definitions" (*id.* at 14–16). To the extent Google directs this argument to names, it is moot in light of the above ruling. To the extent Google directs this argument to other elements of the API package specifications, it is not adequately supported.

Google's lack of specificity is fatal. If Google believes, for example, that a particular method declaration is a *scene a faire* or is the only possible way to express a given function, then Google should provide evidence and argument supporting its views as to that method declaration. Instead, Google argues — relying mostly on non-binding authority[2] — that entire *categories* of elements in API specifications do not merit copyright protection. This approach ignores the possibility that some method declarations (for example) may be subject to the merger doctrine or may be *scenes a faire*, whereas other method declarations may be creative contributions subject to copyright protection. Google has not justified the sweeping ruling it requests. Google has not even identified which categories of specification elements it deems unprotectable under these doctrines. This order declines to hold that API package specifications, or any particular category of elements they contain, are unprotectable under the *scenes a faire* or merger doctrines.

C. Methods of Operation.

"In no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, *method of operation*, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work."

[2] The only binding authority Google cites is the *Sega* decision. The cited discussion addresses computer program code, not documentation. Google has not justified applying the *Sega* rationale to documentation such as the API package specifications at issue here.

17 U.S.C. 102(b) (emphasis added). Google argues that "APIs for a programming language" are unprotected methods of operation (Br. 13). Google, however, does not use the term API consistently in the relevant portions of its briefs, so it is unclear precisely what Google is attempting to characterize as a method of operation. Google states that *all* "elements common to Oracle's Java language APIs and the Android APIs are unprotectable methods of operation," but Google does not specify which elements it views as common (*id.* at 12). Context suggests two possible interpretations for Google's use of the term APIs. Both of Google's apparent arguments are unavailing.

First, Google appears to direct its methods-of-operation argument to APIs themselves as the term is used in this order — that is, to the abstract concept of an interface between programs. In its reply brief, Google distinguishes APIs both from their *implementation* in libraries of code ("the APIs are not the libraries themselves") and from their *documentation* in reference materials ("The APIs do not 'tell' how to use the libraries, they are the *means by which one uses* the libraries; the *documentation* for the APIs 'tells' how to use the libraries.") (Reply Br. 2–3). Google's argument that APIs are unprotectable methods of operation attacks a straw man. *It is not the APIs but rather the specifications for 37 API packages that are accused.* Even if Google can show that *APIs* are methods of operation not subject to copyright protection, that would not defeat Oracle's infringement claim concerning the accused *specifications*.

Google may be trying to head off a possible argument by Oracle that the APIs described in the specifications are nonliteral elements of the specifications subject to copyright protection. It is unclear whether Oracle is advancing such an argument. Oracle's opposition brief seems to use the term API to refer to API packages *and* API package *specifications*. If this interpretation is correct, then the parties' arguments concerning whether "APIs" are methods of operation simply swipe past each other, with each party using the term in a different way. Because the issue is not properly teed up for summary judgment, this order does not decide whether APIs are methods of operation.

Second, Google also states that "API *specifications* are methods of operation" (Br. 14). This conclusion does not follow from Google's argument that APIs — meaning conceptual interfaces between programs — are methods of operation. No other supporting argument is provided. *API specifications are written documentation.* Even if Google could show that APIs are methods of operation, that would not mean that a written work that describes or embodies APIs is automatically exempt from copyright protection. This order finds that the API package specifications at issue are not "methods of operation" under 17 U.S.C. 102(b).

D. Degree of Similarity.

The copying element of copyright infringement generally can be proven by showing that the alleged infringer had access to the copyrighted work and that the protected portions of the works are substantially similar. *Jada Toys*, 518 F.3d at 636–37. "When the range of protectable and unauthorized expression is narrow," however, "the appropriate standard for illicit copying is virtual identity" rather than substantial similarity. *Apple Computer, Inc. v. Microsoft Corp.*, 35 F.3d 1435, 1439 (9th Cir. 1994).

Google argues that "[g]iven the substantial unprotected elements in the documentation (such as the API method declarations), the 'virtual identity' standard applies here" (Br. 24). This order agrees with Google that the *names* of the various items appearing in the disputed API package specifications are not protected by copyright. Google, however, has not shown that any other elements of the specifications are exempt from copyright protection. Because Google has not proven that a substantial portion of the specifications is unprotected, Google's justification for applying the virtual identity standard fails. This order therefore need not consider Google's arguments that the disputed Java and Android API package specifications are not virtually identical. In particular, Google analyzes the selection and arrangement of elements within the specifications under only the virtual identity standard (Br. 24–25).

As a fallback position, Google argues that even under the substantial similarity standard, the disputed Java and Android API package specifications are not sufficiently similar to show copying. Google analogizes the specifications to dictionary definitions whose similarities are driven by external constraints, and Google cites an expert opinion that the Java and Android platforms are not substantially similar (Br. 24; Astrachan Exh. 1 at 77). Predictably, Oracle presents an opposing expert opinion that the API package specifications at issue *are* substantially similar (Mitchell Exh. 1 at 45). This conflicting expert testimony highlights a factual issue that precludes summary judgment; a reasonable trier of fact might agree with either expert's analysis of the degree of similarity between the asserted and accused specifications.

Google argues that Oracle's expert testimony is not sufficient to defeat summary judgment. Google criticizes the expert for offering a "summary 'conclusion'" based on a "single illustrative example," which Google interprets differently (Reply Br. 11). In his report, however, the expert provides multiple examples and explains that he conducted a detailed comparison of each of the API package specification pairs at issue (Mitchell Exh. 1 at 60–63). His opinion that the Android specifications are substantially similar to their Java counterparts is not a mere "[c]onclusory statement[] without factual support." *See Surrell v. Cal. Water Serv. Co.*, 518 F.3d 1097, 1103 (9th Cir. 2008). If Google disputes the basis for the opinion by Oracle's expert or his analysis of the specifications, then Google should raise its critiques during cross-examination at trial. Google has not earned summary judgment of no copying under either of the possible standards for comparison — virtual identity or substantial similarity.

E. Fair Use.

The following factors are considered in determining whether the use made of a work is a fair use: (1) the purpose and character of the use, including whether such use is of a commercial nature or is for nonprofit educational purposes; (2) the nature of the copyrighted work; (3) the amount and substantiality of the portion used in relation to the copyrighted work as a whole; and (4) the effect of the use upon the potential market for or value of the copyrighted work.

17 U.S.C. 107. Google argues that its alleged use of elements from the Java API package specifications in its Android API specifications was fair (Br. 19–22). Evaluation of the fair use factors, however, depends upon disputed questions of material fact. As such, no finding of fair use can be made on the summary judgment record.

For example, with respect to factor four, Google argues that "Android has contributed positively to the market for the copyrighted works by increasing the number of Java language developers" (Br. 21). Google cites positive reactions by Sun executives at the time when Android was first released in 2007. These statements do not prove anything about Android's actual impact

United States District Court
For the Northern District of California

on the Java market since that time. Moreover, Oracle presents sworn testimony that Android fragmented the Java platform and locked Java out of the smartphone market (Swoopes Exh. 6 at 111–12). Oracle and Google both employ complex business models for their respective products. The question of damages is one of the most complicated and hotly contested issues in this action. On the present record, a reasonable fact finder could disagree with Google's rosy depiction of Android's impact on the Java market.

Because fact issues preclude a summary judgment finding of fair use, this order does not reach the parties' arguments on all of the fair use factors.

* * *

This order finds that the names of the various items appearing in the disputed API package specifications are not protected by copyright. This order makes no finding as to whether any other elements of the API package specifications (or their selection or arrangement) are protected or infringed.

3. INDIRECT INFRINGEMENT.

Google argues that Oracle's indirect copyright infringement theories fail because Oracle cannot establish any underlying direct copyright infringement (Br. 25). Because Google is not entitled to summary judgment on direct infringement, Google also is not entitled to summary judgment on indirect infringement.

CONCLUSION

For the foregoing reasons, defendant's motion for summary judgment on the copyright infringement claim is **GRANTED IN PART AND DENIED IN PART**. This order finds that the names of the various items appearing in the disputed API package specifications are not protected by copyright. To that extent, the motion is **GRANTED**. All of defendant's other summary judgment theories regarding the copyright claim are **DENIED**. Plaintiff's evidentiary objections to the Bornstein declaration and the Astrachan declaration are **MOOT**.

IT IS SO ORDERED.

Dated: September 15, 2011.

WILLIAM ALSUP
UNITED STATES DISTRICT JUDGE