MORRISON & FOERSTER LLP
MICHAEL A. JACOBS (Bar No. 111664)
mjacobs@mofo.com
KENNETH A. KUWAYTI (Bar No. 145384)
kkuwayti@mofo.com
MARC DAVID PETERS (Bar No. 211725)
mdpeters@mofo.com
DANIEL P. MUINO (Bar No. 209624)
dmuino@mofo.com
755 Page Mill Road, Palo Alto, CA 94304-1018
Telephone: (650) 813-5600 / Facsimile: (650) 494-0792

BOIES, SCHILLER & FLEXNER LLP
DAVID BOIES (Admitted *Pro Hac Vice*)
dboies@bsfllp.com
333 Main Street, Armonk, NY 10504
Telephone: (914) 749-8200 / Facsimile: (914) 749-8300
STEVEN C. HOLTZMAN (Bar No. 144177)
sholtzman@bsfllp.com
1999 Harrison St., Suite 900, Oakland, CA 94612
Telephone: (510) 874-1000 / Facsimile: (510) 874-1460

ORACLE CORPORATION
DORIAN DALEY (Bar No. 129049)
dorian.daley@oracle.com
DEBORAH K. MILLER (Bar No. 95527)
deborah.miller@oracle.com
MATTHEW M. SARBORARIA (Bar No. 211600)
matthew.sarboraria@oracle.com
500 Oracle Parkway, Redwood City, CA 94065
Telephone: (650) 506-5200 / Facsimile: (650) 506-7114

Attorneys for Plaintiff
ORACLE AMERICA, INC.

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO DIVISION

ORACLE AMERICA, INC. Plaintiff, v. GOOGLE INC. Defendant.	Case No. CV 10-03561 WHA **ORACLE'S MAY 24, 2012 COPYRIGHT REPLY BRIEF** Dept.: Courtroom 8, 19th Floor Judge: Honorable William H. Alsup

Android is not interoperable with Java. Java applications cannot run on Android. Android applications cannot run on Java. This was intentional. Google did not want an interoperable platform and offers no proof it set out to create one. Google took only what it wanted, hijacking the Java developers by using their familiarity with the Java APIs to get them to program for Android. Google's "compatibility" claim is not supported by the law or the facts.

I. INTERFACES AND EXCEPTIONS

The parties agree that Android Froyo includes 158 of the 171 interfaces from the 37 API packages in J2SE 5.0. ECF No. 1192 at 1. This illustrates not just the extent of Google's copying, but also that Google *selected* the interfaces it wanted to copy. Google's failure to implement *all* of the interfaces puts the lie to Google's claim that "the public interfaces in the 37 API packages are functionally required for compatibility with the APIs in those packages." *Id.* at 3. Google's copying served its own business goals and was not compelled by any compatibility requirement, functional or otherwise. In fact, Google deliberately chose not to be compatible.

Google's example of the Comparable interface demonstrates this as well. The Java programming language does not require a Comparable interface (TX 1062), and it did not exist in Java before version 1.2 (ECF No. 1192-3 at line 79), so Google did not have to copy it. In J2SE, 50 classes in a variety of packages implement the Comparable interface. Android copies this same structure, but omits a few of the 50 classes. For example, the classes CompositeName and CompoundName from the javax.naming API package each implement Comparable, but Android does not include that package. *See* TX 610.2 at docs\api\java\lang\Comparable.html.

The parties count different numbers of exceptions, but they agree Google copied over 1250 throws clauses. *See* ECF No. 1192 at 1. Oracle's exception count shows Google again copied almost all, but not all, of the throws clauses exactly, further confirming that compatibility requirements did not drive its extensive copying. Google gives the example of the FileNotFoundException, which is not required by the Java programming language (TX 1062), but was included in Android anyway. Google failed to include other exceptions defined in J2SE 5.0, such as RefreshFailedException, which is part of Java's API package javax.security.auth but not

Android's version of the package. *Compare* TX 610.2 at docs\api\javax\security\auth\package-summary.html *with* TX 767 at javax\security\auth\package-summary.html.

II. ANDROID AND JAVA ARE NOT INTEROPERABLE

Android is not interoperable with Java. Surely it is a huge red flag that in response to the Court's questions about interoperability, Google resorts to claiming Android is "compatible with the skills and expectations of Java language programmers." ECF No. 1192 at 7.

Google states there is no quantitative data in the record showing the extent to which applications written for one platform will run on the other. *Id.* at 5. Actually, there is. The testimony of both parties' experts is 0%. *See* ECF No. 1191 at 4-5. Google could not identify a single application at trial or in its brief that can run on both platforms. But the record contains plenty of evidence of simple applications that will not run on both platforms because Google left out many Java API packages, apparently in favor of its own "better" APIs. *See id.* at 3-7.

Google can point only to the short piece of code Dr. Astrachan wrote at trial, and his testimony that "For those 37 packages, the code that I write on one platform will run on the other platform." ECF No. 1192 at 5. As discussed in Oracle's opening brief, even this is not true, as Dr. Astrachan himself admitted. *See* ECF No. 1191 at 3-6. Moreover, Google presented no evidence at trial that its selective API copying will allow the reuse of a significant number of such code fragments. It cites only to Dan Bornstein's vague statement that "there's a lot source code out there that wasn't—you know, wasn't written by—well, that was written by lots of people that already existed that could potentially work just fine on Android." ECF No. 1192 at 6. Google now claims it selective copying "reduced the effort required to 'port' an application" from Java to Android in a way that is "similar" to what is necessary to port an application from one Java platform to another. *Id.* But the record contains no evidence at all of the effort required to port code from one Java platform to another, or how that compares with porting code from Java to Android. Google's quotation from Dr. Reinhold has nothing to do with this point. *See id.*

Google fails to identify any evidence in the record that its copying motive was interoperability. Google cites to only two things. The first is Dan Bornstein's testimony that the goal was *not* to create interoperability but only to "provide something that was familiar to

developers." ECF 1192 at 7 (quoting RT 1783:19-21 (Bornstein)). Mr. Bornstein confirms immediately before and after this sentence that it was not a goal to implement all of the packages in any Java platform. *See* RT 1783:15-22 (Bornstein). Google also cites the Noser statement of work, which it quotes as stating Google "was 'interested in *compatibility* with J2SE 1.5…..'" ECF No. 1192 at 7 (emphasis in original). But the document requires Noser to provide only a subset of packages, and makes clear Google never intended to fully implement even those, stating Google will deliver "a 'detailed minimum list of methods and classes to be implemented.'" TX 2765 at 21-22. Google wanted to capture Java developers, not achieve interoperability.

III. GOOGLE DISTORTS THE NINTH CIRCUIT'S HOLDING IN *SEGA*

Sega is a fair use case. The Ninth Circuit summarized its holding as follows:

> We conclude that where disassembly is the only way to gain access to the ideas and functional elements embodied in a copyrighted computer program and where there is a legitimate reason for seeking such access, disassembly is a fair use of the copyrighted work, as a matter of law. Our conclusion does not, of course, insulate Accolade from a claim of copyright infringement with respect to its finished products.

Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1527-28 (9th Cir. 1993).

Google seizes on the court's statement that "Accolade copied Sega's software solely in order to discover the functional requirements for compatibility with the Genesis console—aspects of Sega's programs that are not protected by copyright." *Id.* at 1522. But that statement does not help it here. *First*, Google did not copy the Java APIs to "discover" anything. The APIs were readily available for viewing online and Google copied them into Android, so the holding of *Sega* does not even apply. *Second*, Google has not shown the Java APIs are merely "functional requirements." The Ninth Circuit did not conduct a detailed analysis of this issue in *Sega* because the question of infringement in the final product was reserved by Sega and left for remand. *See id.* at 1528. But the decision shows that in determining whether an element of a computer program is a mere functional requirement the court will look to the level of creative expression involved. That is what the court used to distinguish the S-E-G-A 20 byte initialization code from the "*original program*" in *Atari v. Nintendo*, 975 F.2d 832, 840 (Fed. Cir. 1992). *Sega*, 977 F.2d at 1524 n.7 (emphasis in original). Google witnesses admitted there was significant creative

expression here. *See* RT 750:2-752:14 (Bloch); TX 1090 at 128:8-18 (Astrachan Dep.). *Third*, uses the term "compatibility" very differently from the Ninth Circuit in *Sega*. The issue in *Sega* was that defendant's games "would not operate" on Sony's console without copying, not that being "compatible with the skills and expectations of Java language programmers" permitted copying. *Compare Sega*, 977 F.2d at 1515-16 *with* ECF No. 1192 at 7.

Google also overlooks the fact that *Sega* was about fair use. It incorrectly claims that under *Sega "it does not matter"* whether or not it copied to make Android compatible. ECF No. 1192 at 8. *Sega* requires a defendant to have "a legitimate reason" for even intermediate copying. *See, e.g., Sega*, 977 F.2d at 1527-28. Google's interoperability argument does not pass the red face test. Copying to "embrace and extend" is the opposite of what the court endorsed.

IV. *SONY V. CONNECTIX*

The Court asked the parties to address what Connectix ultimately duplicated to allow desktops to run PlayStation games. The Ninth Circuit and district court opinions do not discuss or analyze this issue. Both courts emphasize that Sony did not accuse the final product of infringement. *See, e.g., Sony Computer Entm't Inc. v. Connectix Corp.*, 48 F. Supp. 2d 1212, 1217 (N.D. Cal. 1999) (*"Sony I"*) ("Sony's copyright infringement claim is based on a theory of *intermediate* infringement."); *Sony Computer Entm't, Inc. v. Connectix Corp.*, 203 F.3d 596, 604 (9th Cir. 2000) ("*Sony II*") ("nor does Sony contend that Connectix's final product contains infringing material"). Unfortunately, because *Sony* is twelve years old, Oracle could not obtain copies of the complete record within the Court's briefing timetable.

The materials Oracle has been able to retrieve show the content Connectix duplicated is far simpler than the Java APIs. Connectix alleged it "began with an empty table consisting of entry points into the BIOS." Connectix's Opening Appellate Brief, 1999 WL 33623860, at *13 (9th Cir. May 27, 1999). The precise nature of this empty table or its entry points is unclear, but most likely it was a list of memory addresses of functions that game programs could invoke as needed. Connectix alleged that about a third to a half of the functions were "standard 'C' language functions that would be familiar to any experienced programmer." *Id.* Connectix alleged that it implemented 137 of 242 of the functions from Sony's BIOS. *See id.* at *18.

Oracle does not see any reference in the record it retrieved to an application binary interface ("ABI"). An ABI describes low-level system conventions, such as how one routine passes arguments to, and receives a return value from, another. Unlike an API, an ABI does not specify which routines must exist or how they are intended to be used.

As *Sony* shows, not all interfaces are the same. Nothing in the *Sony* record suggests that there was significant creativity in the entry point table Connectix copied, in stark contrast to the trial record in this case. Neither opinion addressed whether Connectix copied an original structure, sequence, or organization in its BIOS program or the table. The district court noted that "Sony BIOS consists of a combination of C source code and R3000 micro-processor assembly language." *Sony I*, 48 F. Supp. 2d at 1215. The C language does not recognize the concepts of a class or interface hierarchy, and nothing in Connectix's table approaches the complex interrelationships in the Java APIs. The only reference to sequence at all is Connectix's claim that its entry point table needed to "contain the same entry points, and be in the same order and format" as Sony's table. Connectix Opening Appellate Br., 1999 WL 33623860, at *13.

Whether Connectix copied names from Sony's BIOS, and for what purpose, is also unclear, but it appears to have been very limited, unlike here. Sony alleged that Connectix's Virtual Game Station ("VGS") "includes several function names that are identical to function names used in the PlayStation operating system." Pl.'s Mot. TRO, 1999 WL 33743495 at 2 (N.D. Cal. Feb. 3, 1999). Sony argued this point not to claim that Connectix's VGS (the final product) was infringing, but instead as evidence to prove that Connectix had, at some intermediate point, disassembled (and therefore copied) Sony's BIOS when creating the VGS. *See id.* But it is not clear that the names were used to call the functions—they could have been used for debugging purposes instead—and Connectix's allegation regarding the table of entry points suggests they were not. Connectix's Opening Appellate Brief, 1999 WL 33623860, at *13.

In any event, none of this had anything to do with the holding in *Sony*. *Sony*, like *Sega*, was a fair use case about intermediate copying for reverse engineering. *See Sony II*, 203 F.3d at 608. For the same reasons that *Sega* does not bear on the copyrightability of the SSO of the Java APIs or absolve Google for its infringement, neither does *Sony*.

Dated: May 24, 2012

MORRISON & FOERSTER LLP

By: */s/ Michael A. Jacobs*
Michael A. Jacobs

Attorneys for Plaintiff
ORACLE AMERICA, INC.