Java API 複製にフェアユース適用を認めた米国連邦最高裁判決

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

会員　小林　和人※，会員　齋藤　歩記※※

要　約

Java は Sun が開発したプログラミング言語である。今日まで Sun の Java プラットフォーム（Java SE）を通じてプログラマに提供され，Java でプログラミングされたソフトウェアが多くの製品に実装されてきた。Google は Java API（Application Programming Interface）の一部を複製して自社の Android プラットフォームで使用した。その後，Oracle は Sun を買収して Java の著作権者となり，Google の複製行為は著作権侵害であるとして提訴した。地裁は API の著作物性を否定したが，控訴審は著作物性を肯定してフェアユース要件について再審理するよう差戻し判決した。差戻し後の地裁では陪審はフェアユースの適用を認めたが，控訴審はこれを破棄したところ，Google は最高裁判所に裁量上告した。最高裁判所は，訴えのあった著作物性については判断せず，著作物性があると仮定した上でフェアユース要件のみ検討した。その結果，Google の行為は著作権法に違反しないと判決した。本稿は，Java API に関する Google v. Oracle 米国最高裁判決の全貌を解説し，考察を加えるものである。

目次

1．はじめに

Java はかつてエンジニアリングワークステーション（EWS）市場を席捲していた Sun Microsystems（以下，「Sun」という。）が開発したプログラミング言語である。様々なハードウェア・OS への移植性が高いことを特徴とする。今日まで Sun の Java プラットフォーム（Java SE（Standard Edition）等）を通じてプログラマに提供され，Java でプログラミングされたソフトウェアは多くの製品に実装されてきた。その後，Oracle America, Inc.（以下，「Oracle」という。）が Sun を買収し Java の著作権者になった。Google LLC（以下，「Google」という。）はインターネット関連のサービスや製品で大きな市場支配力をもっている米国の企業である。その Google が Java API の一部（宣言コード）を複製して自社の Android（スマートフォン用 OS）プラットフォームで使用した。

API（Application Programming Interface）とは，特定の機能を動作するよう予め書かれたソフトウェアの集まりであって，プログラマがプログラムを設計する際に呼び出して（call）使用できるようにしたものである。API の内部の呼び出し名の定義を宣言コード（Declaring Code）と呼び，呼び出し名と紐づけされたソフトウェアを実装コード（Implementation

※　次世代パテントプラットフォーム研究会，東京工業大学
※※ 次世代パテントプラットフォーム研究会

Code）と呼ぶ。個々の実装コードはコンピュータのハードウェアと協働して，実際に所定のタスクを実行する。したがって，プログラマは巨大な模型のようなプログラムを設計する際に，模型に使用する部品を全て素材から作成する（Java 言語を使って書く）必要はなく，すでにプラモデルのようにできあがったパーツ（API）群をうまく利用することでプログラムの設計を効率的に進めることができる（もちろん，自分でゼロから作成しなければならない部品（ソフトウェア）もある）。

2010 年，Oracle は Sun を買収して Java の著作権者となり，同年に Google の複製行為は著作権侵害であるとしてカリフォルニア北部地区連邦地方裁判所（以下，「地裁」という。）に提訴した。本件はその後，控訴，差戻しを繰り返し，2018 年に，連邦巡回区控訴裁判所（以下，「CAFC」という。）の判決に対して，Google は裁量上告の申立を最高裁判所（以下，必要がない限り「裁判所」と略記する。）に提出した。その申立は，Google が複製した部分（宣言コード）に著作物性はあるか（米国著作権法第 102 条（b），以下，説明がない限り，条文番号は米国著作権法），②著作物性があるとした場合に，Google の複製行為について「フェアユース」（公正な利用，第 107 条）の抗弁が成立し，著作権侵害から免責されるかどうかの判断を求めるものであった。

裁判所は議論を進めるにあたって，本件訴訟の対象である API は著作権の保護対象であると仮定した上で，Google の API の複製はフェアユースを構成していることから，著作権侵害ではないと判示した(1)。

2．経緯

2．1　Java と Android

2003 年，Android 社はスマートフォンのソフトウェアのビジネスへの参入を目指して設立された。2005 年，同じくスマートフォンなどのソフトウェア・プラットフォームを開発しようとしていた Google は，Android 社を買収した。ソフトウェア・プラットフォームとは，プログラマがプログラムやアプリケーションを開発するために必要な開発環境を提供するソフトウェアツール群である。

Google は，ソフトウェア開発者が無料で使用できる，自由かつオープンな Android プラットフォームを構想していた。すなわち，Google はより多くの開発者が Android プラットフォームを使用することで，より多くの Android ベースのアプリケーションが開発され，Android ベースのスマートフォンも消費者にとって魅力的になることで，その購買も促進されると考えた。この構想の実現には，多くの熟練プログラマを引き付けることが必要だった。当時，全世界で 600 万人と推定されるソフトウェア開発者がプログラミング言語の Java を習得してソフトウェアを設計していた。Java は，Sun が開発し，仕様を公開したプログラミング言語である。Java は Java 仮想マシンを OS 上で動作させることで，コンピュータのハードウェアや OS には依存せずに，Java プログラムが安定動作することが大きな特徴である。多くのプログラマが，Sun の提供していた Java SE プラットフォームを使用し，主にデスクトップやラップトップのコンピュータ向けのプログラムを開発していた。

そこで，Google は Android 社を買収すると，Android プラットフォームに対する Java プラットフォーム全体のライセンス提供について Sun と交渉を開始した。Sun はライセンスポリシーとして，Java SE プラットフォームとの相互接続性（Interoperability）・互換性（Compatibility）が確保されることを条件としていた。つまり Sun Java プラットフォームの API が Android プラットフォームで動作し，Android プラットフォームで新たに追加された API も Sun Java プラットフォームで正常に動作することである。ところが，Google は Android プラットフォームで改変された API が Sun の Java SE プラットフォームで動作するとは約束したくなかった。実際，Sun は Java コミュニティでの厳格な審査プロセスで新規 API の登録を承認していたが，Google はプログラマからの新規 API の自由な追加変更を認めるようなオープンなビジネスモデルを目論んでおり，Sun の互換性ポリシーに従っていると Google のビジネスモデル構想は弱体化されると考えた。このような意見の対立から両者の交渉は決裂した。2007 年，Google は独自の Android プラットフォームを構築することを決定し，独自仕様の Java API と独自の仮想 Java マシンを開発して，同社が中心となって設立したコンソーシアム（Open Handset Alliance（OHA））を通じて公開した。2008 年からは複数のメーカーが Android ベースのスマートフォンの販売を開始した。

２．２　API の複製について

Google は Java API の一部（約 11,500 行の宣言コードに相当）を複製して自社の Android プラットフォームで使用した。API はプログラマの呼び出し名に相当する宣言コードと実際に特定の機能を実行する実装コードから構成されている。宣言コードについては，Google はその 37 個のパッケージに Sun Java API の宣言コードを複製し，その他の新しいパッケージでは独自に宣言コードを設計した。一方で，実装コードについては Sun Java API から複製せず，Google が全て独自に設計した。Google が，このように Sun Java API の一部を Android プラットフォームに複製したことで，Java プログラマは習得している「タスク（メソッド）呼び出し」の使い方の知見を Android プラットフォームでのプログラミングで活用することができた。もし，Google がこのような複製をせずに Sun Java API とは別の API システムを開発した場合，プログラマは同じようなタスクを実行するために新しい API システムの再習得を余儀なくされたことは疑いない。

Android ベースの機器は，2007 年の発売から 5 年のうちに米国市場で大きなシェアを獲得し，2015 年には Android の売上高は 420 億ドル以上となっていた。対する Oracle は 2010 年に Sun を買収し，ただちに Google の Java API 複製が著作権侵害であるとして地裁に訴訟を提起した。

２．３　訴訟の経緯

本件訴訟は複雑で長い歴史がある。Oracle は当初，Google による Sun Java API の使用は著作権法と特許法の両方に違反していると地裁に訴えた。地裁は訴訟手続を 3 つに整理した。1 つ目は著作権の問題，2 つ目は特許権の問題，3 つ目は損害額の計算（必要に応じて）である。地裁は，著作権法で API が保護されるべきかを裁判官が決定し，Google による API の複製が著作権を侵害しているか，侵害している場合にフェアユースの抗弁が成立するか，は陪審員が決定するものと判断した。

6 週間の審理の後，陪審員は Oracle の特許権侵害の主張を却下した。また，陪審員は，著作権侵害を一部認定したが，フェアユースの抗弁については判断することができなかった。地裁の裁判官は Java API の構造（Structure），シーケンス（Sequence），組織（Organization）（以下，「SSO」という。）に創作性・独創性があることを認めながらも，コマンド構造・システム・動作方法（事前に付与された所定機能を実行するコマンドの階層構造）は第 102 条（b）の「システム又は動作方法」に該当するとして著作物性を否定し，著作権侵害ではないと判示した[(2)]。

控訴審（CAFC）は，API の宣言コードと SSO の両方が著作権で保護されると判断した。CAFC の指摘では Google は実装コードを作成したのであるから宣言コードも独自に作成するのに困難な理由はない。フェアユース抗弁については十分な証拠がないことを理由に改めて判断するよう地裁に差し戻した（CAFC2014 年判決）。Google は CAFC の判断について裁量上告を申し立てしたが，最高裁判所はこれを受理しなかった[(3)]。

差戻し後の地裁の陪審は，フェアユースの適用を認めたが，Oracle は控訴した。控訴審（CAFC）は「著作権で保護された著作物を一字一句そのままに抜き出して，競合プラットフォームで原著作物と同一の目的と機能のために使用するのは，公平なことではない」として原判決を再び破棄し，損害賠償の審理のために，本件を再度地裁に差し戻した（CAFC2018 年判決）。Google はその後，最高裁判所に裁量上告の申立を提出した。Google は，著作物性とフェアユースの両方に関する CAFC の決定を見直すよう裁判所に求めた。最高裁判所はこの申立を受理した。

３．裁判所の検討

３．１　米国著作権法

本件では現行著作権法の次の 4 つが重要である。第一には，著作権を取得するためには「著作者が作成した著作物である」こと，その著作物が「創作的」であること，その著作物が「有形的表現媒体」に「固定されている」ことが必要である（第 102 条（a））。

第二には，著作権で保護される著作物の種類を列挙している。これには，「言語」，「音楽」，「演劇」，「映画」，「建築」，その他の著作物が含まれる（第 102 条（a））。1980 年，連邦議会は著作権法の適用範囲を拡大し，コンピュータプログラムを対象に含めた。「コンピュータプログラム」は，「一定の結果を得るためにコンピュータで直接または間接に使用される，文または命令の集合」と定義されている（第 101 条）。

第三に，著作権で保護される可能性のある著作物に

対する制限を定めている。例えば，著作権による保護は，「いかなる場合にも，アイデア，手順，プロセス，システム，操作方法，概念，原理または発見等」には拡張されない（第 102 条（b））。そのため，判例では，しばしば，アイデアを保護する特許とは異なって，著作権は「表現」を保護するがその背後のアイデアは保護しない，と手短かに述べている。

第四には，著作権を有する著作物に関しても，その保護の範囲に制限を課している。例えば，著作権法は，著作者の独占的権利を，実演や展示（第 110 条）または録音物の実演（第 114 条）に限定している。また，本件に直接関連することとして，著作権者は，他人による著作物の「フェアユース」を妨げることはできない（第 107 条）。

3．2　フェアユース

米国著作権法第 107 条はフェアユースの要件を規定している（表 1）[4]。

表 1　米国著作権法第 107 条（訳出典：著作権情報センター外国著作権法（アメリカ編））

第 107 条　排他的権利の制限：フェア・ユース 第 106 条および第 106A 条の規定にかかわらず，批評，解説，ニュース報道，教授（教室における使用のために複数のコピーを作成する行為を含む），研究または調査等を目的とする著作権のある著作物のフェア・ユース（コピーまたはレコードへの複製その他第 106 条に定める手段による使用を含む）は，著作権の侵害とならない。著作物の使用がフェア・ユースとなるか否かを判断する場合に考慮すべき要素は，以下のものを含む。 (1) 使用の目的および性質（使用が商業性を有するかまたは非営利的教育目的かを含む）。 (2) 著作権のある著作物の性質。 (3) 著作権のある著作物全体との関連における使用された部分の量および実質性，および (4) 著作権のある著作物の潜在的市場または価値に対する使用の影響。 上記のすべての要素を考慮してフェア・ユースが認定された場合，著作物が未発行であるという事実自体は，かかる認定を妨げない。

「フェアユース」の法理は，「合理的な衡平法」であって，「著作権法が育成しようとする創作性を阻害する場合には，裁判所が著作権法の厳格な適用を避けることを認める」ものである。本条項を適用するにあたり，裁判所は，本条項に掲げる要素は網羅的なものではなく，また，その他の例を除外するものではなく，さらにいくつかの要素は事案に応じてその重要性も異なってくることを認識しており，状況に応じた判断が必要であると理解している。

3．3　裁量上告と争点

Google の裁量上告は 2 つの問題を提起している。第一の問題は，Java の API に著作物性があるか（第 102 条（a）），あるいは著作物性のない「プロセス」，「システム」[5]，「操作方法」（第 102 条（b））に該当するかである。Google は，API の宣言コードと組織（Organization）は保護対象から明らかに除外されると主張している。第二の問題は，Google による API の使用が「フェアユース」かどうかである。Google は，フェアユースに該当すると主張している。

提起されたいずれかの問題について Google を支持する判決が下されれば，Oracle の著作権の主張は棄却されることになる。一方で，急速に変化する技術，経済，ビジネス関連の諸状況を考慮すると，裁判所は当事者の紛争の解決に必要な範囲を越えて回答すべきではないと考える。したがって，本件では，著作物性についての判断は保留する。その上で，本件の議論に限定して Sun Java API 全体が著作権の対象であると仮定し，Google の API の一部使用が「フェアユース」であるかどうかを，検討する。

3．4　コンピュータプログラムとフェアユース

「フェアユース」条項（第 107 条）の 4 要件は，1841 年の Folsom v. Marsh 事件（ワシントン大統領の著作集からのワシントンの手紙の複製が争われた）での Story 判事の意見をその源とし，のちに著作権法で定められた。現代の裁判所もこの法理を使用していることからも，この概念は柔軟性があって，裁判所は著作権法の相反する目的を考慮してこの法理を適用しなければならず，その適用は状況に応じて変わりうることが明らかである。したがって，著作権で保護されている素材が，事実ではなくフィクションの場合，ニュース報道ではなく映画の場合，または実用的な機能ではなく芸術的な機能を果たす場合，著作権の保護はより強くなる可能性がある。最高裁判所（Feist 事件）は，著作権で保護される素材が著作権で保護されない素材と結びついているような状況では，著作権保護は「薄い」と判示している。

一般的に，コンピュータプログラムは，機能的な特徴を備えている点で，本，映画，その他の多くの「文学作品」とは異なる。こうした違いから，連邦議会は

コンピュータプログラムの著作権保護を認めるべきかどうかについて，長い時間をかけて真剣に検討してきた。1974 年，連邦議会はこの件を調査するために，著作物の新技術利用に関する国家委員会（CONTU）を設立した。CONTU は，数年にわたる調査の後，「コンピュータプログラムの著作権保護が可能であることが望ましい」と結論づけた。同時に CONTU は，コンピュータプログラムには固有の特徴があることも認め，「プログラムのユーザーや一般の人々に不当な負担をかけない」ように配慮し，著作権は「創作意欲を高めるために必要である以上の経済力を誰かに与えてはならない」と報告書に記している。CONTU は，裁判所が著作権の既存の法理（フェアユース等）をケースバイケースで適用することにより，著作権者がイノベーションを阻害するような著作権を行使することを抑制できると考えた。これを踏まえて，1976 年に連邦議会はコンピュータプログラムの保護を著作権法に盛り込んだ。

結論として，コンピュータプログラムの著作権の法的な範囲を決定する上でフェアユースの法理は重要な役割を果たすと考える。フェアユースは，個々の技術を区別して判断するのに有用であって，コンピュータコードに表現的な特徴と機能的な特徴が混在していても，それらを区別することができる。また，フェアユースは，著作物を創作するインセンティブの必要性に焦点を当てつつ，保護を強化することで，どの程度，他の市場や他の製品の開発に影響がないか，あるいは不当な弊害を生じさせるかを検討することができる。一言で言えば，フェアユースは，著作権の独占力を合法的な範囲内に収めるために，状況に応じたチェック機能を提供するという基本的な目的を果たすことができる。

なお，Thomas 判事は，反対意見を以下のとおり述べている。プログラムの使用方法や設計方法には重大な違いがあったとしても，それを「著作物の性質」の考慮によってコンピュータコードを区別（API の宣言コードと実装コード）しようとするのは無理がある。コンピュータコードを複製する理由は，他の種類の著作物の場合とは大きく異なるとしても，新しいプログラムにコードを再利用することが，有効な「目的と性格」を持つこともない。

また，裁判の開始後に両当事者からの補充書面の提出により，陪審審判を受ける権利を保証している合衆国憲法修正 7 条等の観点から，CAFC が陪審の判断を覆してフェアユースを否定した手続きの適否が論点となった。裁判所は，フェアユースは法律と事実の混合問題であり，本件の最終的な問題は，主に法律問題であると説明した上で，陪審評決への異議を決議する際に，CAFC が支配法を判断したことは再審理条項に違反しないと判断した。

４．フェアユース要件の検討

４．１　著作物の性質

Google の Sun Java API の 37 個のパッケージの宣言コードと SSO の使用が「フェアユース」に該当するのか，フェアユース第 2 要素（著作物の性質）から検討を始める。

Sun Java API は「ユーザーインターフェース」である。これはプログラマという API の利用者（プログラムのユーザーではないので注意）が，一連のメニューコマンドを介して，タスクを実行するコンピュータプログラム（実装コード）を操作，制御する手段を提供する。この API には 3 つの特徴がある。第一に，API には各タスクを実行するための手順を実際にコンピュータに指示する「実装コード」が含まれており，Google は，実装コードを独自に作成している。第二に，Sun Java API は，各タスクの呼び出しに「メソッド呼び出し」と呼ばれる特定のコマンドを関連付けている。Oracle は，プログラマが作成するソフトウェアの中で「メソッド呼び出し」を使用することに対しては著作権侵害を主張していない。

第三に，Sun Java API には，宣言コードが含まれている。宣言コードは，API 内の特定のタスクに名称を定義し，「メソッド」を「パッケージ」と「クラス」という階層化した構造に編成する（クラス，パッケージについての詳細は後述する）。Oracle は，宣言コードの使用は著作権侵害であると主張している。

宣言コードは，コンピュータタスクの分割という一般的な「システム」と密接に結びついている。「システム」は第 102 条（b）で著作権の保護対象ではないことが示されている。また，宣言コードは，タスクを「パッケージ」と「クラス」に編成するという「アイデア」と表裏一体の関係にあるが，「アイデア」も著作権法の保護対象ではない。一方で，宣言コードには，他のコンピュータプログラムと同様に，機能的な性質を持っており，その名前が直感的に覚えやすいよ

うに付けられている側面もある。これに対して実装コードは，その作成のためには，コンピュータのタスク処理スピード，タスクの大きさ，バッテリー管理などを考慮してバランスをとる必要があり，機能的な性質と創作性において，宣言コードとは大きく異なる。

これらの状況を総合すると，宣言コードの使用はその他の多くのプログラムとは異なり，著作権の保護を受けないアイデア（一般的なタスクの分割と編成）と新しい創作的な表現（Google が作成した Android の実装コード）とをインターフェースとして結びつけることをその本質としている。また，複製された宣言コードの価値は，プログラマが，API 習得に費やした時間と価値に由来するところが大きい。裁判所の見解は，先に述べた理由により，宣言コードは，著作権があるとしても，多くのコンピュータプログラム（実装コードなど）に比べると，著作権の中核からは離れている。以上の検討から，本件で第 2 要素はフェアユースの方向を向いている。

４．２　使用の目的と性格

フェアユース第 1 要素の「使用の目的と性格」について，裁判所はこれまで，「新たな目的や異なる性格を持った新規なものが追加され」，「新たな表現，意味，メッセージによって著作物が改変されるか」どうかを検討してきた。この新規で重要なものを追加するという複製行為は「変容的利用（transformative use）」という用語で説明されている。そして使用が「変容的」であるかどうかを判断するには，その複製のより具体的な「目的」と「性格」を検討する必要がある。

本件では，Google が Sun Java API を使用する「複製の目的」は，新しい製品，すなわち Android ベースのスマートフォンを開発し，その用途と有用性を拡大することである。そして，それらの新しい製品は，スマートフォン環境用に非常に創作的で革新的な（開発）ツールをプログラマに提供する。Google のそのような（開発）ツールでの Sun Java API の使用は，著作権法の基本目的である創作的な「進歩」に合致するものである。

Google による Sun Java API の「複製の性格」は，Android に関連するタスクや特定のプログラミング要求に限定されたものであった。Google は，スマートフォン用のプログラムに Google が作成した実装コードを組み込むために，Sun がデスクトップやラップトップコンピュータ向けに作成した API を必要な範囲でのみ複製した。つまりは，Google はデスクトップやラップトップコンピュータとは全く異なるコンピュータ環境（Android ベースのスマートフォン）で動作する新しい実装コードを設計しており（一般に，「再実装」と呼ぶ。），API の一部を複製した理由は Sun Java API を習得したプログラマが，その基本的なプログラミングスキルを新しいシステム（Android）で発揮できるようにするためであった。裁判の証拠は，コンピュータプログラムの効率的開発にはインターフェースの再実装が有効であること，プログラマが習得したプログラミングスキルを活用するために，業界では API の再利用が一般的であることを示している。

第 1 要素（使用の目的と性格）の検討において，しばしば取り上げられるその他の考慮事項として「営利性」と「不誠実性」がある。第 107 条の文言は，免責される複製行為の典型例として，教育や研究等の様々な非営利的用途を含んでいるが，営利性があるとフェアユースではないとは規定されていない。第 107 条には，通常営利目的で行われる「ニュース報道」のような例も含まれている。したがって，CAFC2018 年判決は営利性をフェアユースの認定に不利だと判断したが，裁判所は，Google の使用が営利的な試みであったとしても再実装が本質的に変容をもたらす役割を果たした点で，第 1 要素を否定するものではないと判断する。「不誠実性」については，Campbell 事件判決で，不誠実性がフェアユース分析の有効な考慮事項であるのかについて懐疑的な見方が示されている。本件では，不誠実性の一般的な問題について述べる場ではないことから，不誠実性は結論に際しての決定的な事項ではないと指摘するのみに留める。

４．３　使用された部分の量および実質性

フェアユース第 3 要素（使用の量および実質性）に関して，Google が複製したのは Sun Java API の宣言コードの 37 個のパッケージ，コード数は約 11,500 行に達する。行数だけとらえると，複製した量は膨大である。一方，Sun Java API のソフトウェア全体では，総数 286 万行に対して複製されたのは全体の 0.4％に過ぎない。ここで問題となるのは，複製した 11,500 行のコードを単独で見るべきか，それともかなり大き

な全体のほんの一部として見るべきかということである。最高裁判所（Harper&Row 事件等）は，これまで複製された抜粋部分が原著作物の創作的表現の「核心」から構成されている場合，たとえ複製したのが少量であっても「フェアユースの範囲」からは外れることがあると述べてきた。

Google がした複製の特徴を把握するには，複製した量を評価するよりも Google が複製しなかった数百万行の実体を検討することが有効である。そもそも，Sun Java API は複製されなかったタスク実装コードと不可分な関係にあって，API の目的は実装コードを呼び出すことである。また，Google が複製した理由は API に創作性や美しさがあるからではなく，プログラマがすでに Sun Java API を習得して使いこなしていたからである。Sun Java API から複製した API がなければ，Android スマートフォンシステムを構築するためにプログラマを集めることは極めて困難であったことが予想される。Campbell 事件でも，複製の量が有効かつ変容的な目的に結びつけられている場合には，「実質性」という要素は，フェアユースに有利に働くことが示されている。

Google の基本的な目的は，単に Android システム上で Java プログラミング言語を使えるようにすることではなく，プログラマが Android プラットフォームを搭載したスマートフォン向けに新しいプログラムを作成する際に，Sun Java API の利用で培った知識と経験を活用できるようにすることであった。したがって，Google の行為には「複製の量が有効かつ変容的な目的に結びつけられている」と認められ，「実質性」の要素はフェアユースに有利に働く。

４．４　市場への影響

第 4 要素は，「著作物の市場や価値」の観点での複製の「影響」に焦点を当てる。著作権者の逸失利益を考慮する必要があるが，損失の額だけでなくその原因も考慮する必要がある。また，複製がもたらす公共の利益と著作権者の逸失利益を損失の原因も含めて考慮して判断することが肝要である。

逸失利益については，Android が Java SE の実際または潜在的な市場に損害を与えていないと陪審員は判断した可能性がある。また，Google の API 複製の有無かかわらず，Sun（現在の Oracle）はスマートフォン市場に首尾よく参入することはできなかったと陪審員が判断した可能性もある。その第一の理由として，Android のスマートフォン技術には関係なく，Sun が携帯電話市場で成功するには劣勢な立場にあったことを証拠が示している。また，Java SE の主要市場はラップトップとデスクトップのパソコンであったという十分な証拠も陪審員が認識していた。また，携帯電話市場に進出しようとした Sun の多くの努力が失敗に終わったという事実もあった。その第二の理由として，Google の Android プラットフォームを使用しているデバイスは，Sun がその技術をライセンスしているデバイスとは種類が違う（例えばスマートフォンの登場前の携帯電話や Kindle など）という事実を陪審員は度々聞いている。つまりは Google の主張は，Android が Java ソフトウェアの市場代替品ではない，「モバイル OS スタック」である Android プラットフォームは，「単なるアプリケーション・プログラミング・フレームワーク」である Java SE とは全く異なるタイプの製品である，というものである。Sun は，これに反論し，Android 市場に参入しようとした証拠を示した。しかし，少なくとも，Google が Sun Java API の一部利用をしていなかったとしても，Sun がスマートフォン市場に参入することは困難であったとの証拠を覆すことはできないことを，陪審員のフェアユースの判断は示している。

一方，Google は API を複製して，Android プラットフォームで膨大な収益を稼ぎ出しており，Oracle は Sun Java API の著作権を権利行使することで，（ライセンス収入または損害賠償金として）これらの収益の配分を得ることができた可能性はある。しかし，何故また如何にして Oracle がこの収益を得る権利があるのかを考えることも重要である。一般に，API や表計算プログラムのような新しいインターフェースは初めて市場に登場するとき，その画面の美しさなどの表現力や優れた機能性で，新しいユーザーを引き付けることができる。しかし，時間が経ち，利用者であるプログラマがそのインターフェースを使って仕事をする方法を習得してしまうと，（表現力や機能性といった価値は低下し），彼らが「使い慣れた」という別の価値が生まれることもある。すなわち，Android の収益性の源泉は，第三者（プログラマ）による Sun Java プログラムへの習得の投資に負うところが大きく，Sun が Java API を作成するために行った開発投資に負うところは少ない。ところが，著作権法が，創

作物の操作方法を習得するための第三者（プログラマ）の投資を保護しようとしていると考える理由は見当たらない（Campbell 事件）。

最後に，プログラマが Sun Java API の習得に投資していることを考慮すると，本件において Oracle の著作権の行使を認めることは公衆の利益に害を及ぼす危険性がある。

すなわち，本件で権利行使を認めると，Sun Java API の宣言コードが将来の新しいプログラムの創作性を制限する錠（lock）となり，それを開ける鍵（key）を握っている Oracle（あるいはコンピュータ・インターフェースの著作権を持つ他の企業）に高い利益がもたらされる可能性がある。しかし，その場合の利益は，創造的な改良や新たなアプリケーション，そしてそのインターフェースを使いこなした利用者たるプログラマの新たな用途から得られる。その意味においては，Oracle の錠は著作権の基本的な創作性の目的を促進せず，むしろ妨げることになる。結論として，ユーザーインターフェースの再実装は，創作的な新しいコンピュータコードをより容易に市場に投入することを可能にするものである。総合的に判断すると第 4 要素（市場への影響）もまたフェアユースを支持している。

４．５　判決（法廷意見）

コンピュータプログラムが本来機能的であるという事実から，その技術の世界に伝統的な著作権の概念をそのまま適用することには一定の困難が伴う。しかし，裁判所は，フェアユースの法理の適用が，議会や裁判所の長年の協力的な努力の賜物であることを熟慮し，本件ではフェアユースの概念の本質を変更しないこととする。

Google は，利用者たるプログラマが蓄積した Java プログラミングのスキルを，新規で「変容的」なプログラムで活用するのに必要なコードだけを取り出して，ユーザーインターフェースを再実装したものである。裁判所は，Google による Sun Java API の複製は，法律問題として，その素材のフェアユースであるという結論に達した。CAFC の相反する判決を破棄し，本見解に沿った更なる手続のために本件を差し戻す。以上のように判示する。

なお，Barrett 判事は，本件の検討・決定には関与していない。Thomas 判事，Alito 判事は宣言コードと実装コードを区分する根拠がないこと等を理由として反対意見を提出している。

５．考察

５．１　変容的利用（フェアユース第１要素）

Google が Sun Java API を使用することで，新しい製品，すなわち Android ベースのスマートフォンを開発し，その用途と有用性を拡大することを目指していることがフェアユースの「変容的利用」の要件で肯定的に評価されている。口頭審理にあっても非専門家の陪審員はスマートフォンや Android が革新的な技術であること，またデスクトップパソコンやノートパソコンと Android は全く異なるアーキテクチャであることを専門家から繰り返し説明されている。そもそも Java はハードウェアや OS に依存することのない動作を前提としており，Android で動作させること自体は変容的利用ではなく当然に想定されるものである。CAFC2018 年判決が指摘するように，スマートフォンの限定的な作業環境に適合するよう API を実装すること自体は新規なものの追加や用途拡大にはあたらないと考えられる。

また，ハードウェアキーボードがなくタッチパネルの操作でサードパーティのアプリケーションを連携動作させるのは Android の機能であり（iPhone も同様の機能を備えている），Java API が直接的に実現しているものではない。「変容的利用」の要件をコンピュータプログラムにあてはめて評価するのは容易ではないとは理解するが，口頭審理では劇場公開した映画の観賞とインターネットでのライブ視聴の違いや QWERTY キーボードと携帯電話での文字入力機能の比較など，日常的なアナロジーや平易な機能比較の議論に留まっており[(6)]，そもそも Android アーキテクチャ上で Java API を実装することでどんな新規なものが追加され，またどのように改変されて使用されたことで Android の革新的と呼ばれる性能を実現されたのか，例えば Kindle や BlackBerry とは何が違うのかを含めての説明が，確認した範囲では見当たらず，Android での Java API の使用自体が変容的利用であることの技術的説明のエビデンスが欠如した印象をもつ。また，パロディが新しい創作として社会的価値を提供するとした Campbell 事件，ゲーム機器のリバースエンジニアリングの過程での中間複製をエミュレータという新しいプラットフォームを生み出すとし

たSony v. Connectix事件と比べると，パーソナルコンピュータとスマートフォンではAPIの使用が変容的利用と認めるほどの差異を見出すのは容易ではない。映画産業会はアミカス・ブリーフで，スマートフォンが新しいというだけならばNapsterのデジタル音楽配信市場も同じことがいえたはずだと指摘する(7)。

Thomas判事は，新しいプログラムにコードを再利用することが，有効な「目的と性格」を持つこともないとの反対意見を述べており，もしJava APIの宣言コードをAndroidプラットフォームで使用することが有効な「目的と性格」と判断するのであれば，技術的にどのように使用されてどのようにAndroidの革新性に寄与したのか具体的に示すのが必須ではないか。特許の侵害性や無効性の判断であれば，技術専門的な議論を尽くすことが当然であるが，著作物についてはAPIが技術専門的な内容であるにも関わらず，そこを迂回した議論には素朴な違和感がある。

5．2　フェアユース法理の適用

本件の裁判所の判断は，フェアユース要件が充足されていることを理由として著作権侵害にはならないと判決した。従前からフェアユース法理の適用は予見が困難であるとの認識があったが，改めてそれを再認識する結果となった。そもそものGoogleの行為を振り返るとSunが中核となっていたJavaの開発のコミュニティでのJava APIをその互換性の確保には従わず，いわば外部仕様のつまみ食いによって似て非なるAndroid　Javaの設計と開発を行ったものである。当初のOracleとGoogleのライセンス交渉では，Java APIがSun Javaプラットフォームと Androidプラットフォームの両方で互換性をもち安定した動作を確保することがOracleからの条件として示され，協議が進められていた。GoogleがJavaの開発のコミュニティに協調することなく，ユーザーの自由な改版を認める独自のオープン開発プラットフォームを作成すること自体は，産業界での覇権争い・ゲームチェンジとして認めることができ，著作権問題とは別としてそれだけをもってGoogleの不実等を非難するものではない。かつてはSunもMicrosoft等のOSの上にJavaプラットフォームを形成した立場にある。ところが，ライセンス交渉が決裂して，裁判でGoogleが勝訴した結果，フェアユースの法理によってGoogleは一切のロイヤリティ負担を免責されることとなっている。当事者間で相当のロイヤリティ支払いをもって解決し，また決裂したのであれば裁判所がその対価を判断するべきであるが，フェアユースの適用によって実質的にロイヤリティは無償と判断された帰結にはいささかの違和感が伴う。

ハドソン研究所のアミカス・ブリーフでの指摘を敷衍すれば，Java APIは業界標準ということができ，標準規格での必須特許に相当するものである。すなわち必須特許であれば，ライセンス取得を義務付ける一方でそのロイヤリティはFRANDという合理的かつ非差別的な条件が課せられる(8)。Java APIという業界標準の著作物についてFRANDのようなライセンス義務とロイヤリティ支払いの枠組みで評価して議論する余地はなかったのだろうか（当事者がそのような主張と反論を選択した帰結ではあるが）。

裁判所では，フェアユースが有効と判断した検討の中で，プログラマのJava習得のスキルの活用の公益性や産業発達への寄与を強調し，不利益を被るプログラマ擁護の抗弁のようにも聞こえる。しかし，GoogleがFRAND相当のロイヤリティを支払ってプログラマのJava APIの使用を認めるように手当をすれば足りるものであって，無償でなければ実現できないものでもない。

5．3　今後の影響

裁判所の判断がどの程度適正であったかには関わらず，本件は最高裁判所の判断として今後の事件にも影響を持つことになる。この事件に限っていえばGoogleはその主張が認められ，Androidビジネスはさらに弾みがつくものと思われる。一方で，この判決は既存のプラットフォームが構築したAPIは次の後進企業がフェアユースとして独自利用することを肯定的に評価する指針となった。つまりは，将来Googleも Oracleと同じ立場になる可能性が残ったともいえるのではないだろうか。

6．おわりに

Java APIに関するGoogle v. Oracle米国最高裁判決の全貌を解説し，考察を加えた。本稿をまとめるにあたって，下級審から長年にわたり裁判を詳細に分析されてきた椙山敬士，石新智規両弁護士から自著等論文の提供とご教示をいただいた。また，大和田昭彦弁理士ほかの次世代パテントプラットフォーム研究会の

メンバーに助言をいただいた。ここに深く感謝申し上げる。

（注記）

(1) GOOGLE LLC v. ORACLE AMERICA, INC 141 S.Ct. 1183（2021）

(2) 石新智規，ORACLE AMERICA, INC v. GOOGLE INC　米連邦控訴審裁判所（CAFC）2014 年 5 月 9 日判決～アプリケーションプログラミングインターフェースの著作物性が肯定された事例～No.138　SLN　Softic Law News（2014）

(3) 椙山敬士，ORACLE AMERICA, INC., v. GOOGLE LLC 米連邦巡回区控訴裁判所（CAFC）2018 年 3 月 27 日判決―フェアユースの適用を否定―No.159　SLN　Softic Law News（2018）

(4) 著作権情報センター　外国著作権法（アメリカ編）山田隆司 訳（2018）https://www.cric.or.jp/db/world/america/america_c1a.html

(5) 前掲 4）では「方式」と訳しているが，本稿では「システム」と訳した

(6) 石新智規，Google LLC v. Oracle America, INC. 米連邦最高裁口頭弁論*―米国ソフトウェア著作権の行方―No.165 SLN　Softic Law News（2020）

(7) 玉井克哉，裁判所における「熟議」―グーグル対オラクル著作権侵害事件におけるアミカス・ブリーフを素材に―4 Vol.42 2020 Summer Nextcom（2020）

(8) 前掲注 7）

付録（API に関する技術的説明）

（1） API とは

CAFC は，API を「プログラマが利用できるツールであって，特定の関数を実行するためにゼロからコードを作成（from scratch）するのではなく，それらの関数を動作させるよう予め作成されたコードを，プログラマがプログラムに組み込むことができるようにするしくみである」と説明している（CAFC2014 年判決）。プログラマは API を通じて，予め作成された複雑なタスクの実行コードのライブラリーを活用できる。（なお，API は Java 固有のものではなくて多くのプログラミング言語や Web アプリケーションが提供している。）API の機能は，単純なものから非常に複雑なものまで，例えば，2 つの数値のうちどちらが大きいか，1000 個の数値を昇順に並べ替え等，さまざまである。

また，API はタスクの集まりを特定の方法で編成しており，プログラマは API を通じてプログラムに必要なタスクを効率的に選択できるようになっている。Sun の API（Sun Java API）では，個々のタスクを「メソッド」，類似するメソッドの集まりを「クラス」，類似するクラスの集まりを「パッケージ」と呼んでいる。この Java のメソッド・クラス・パッケージの組織構造が，ソフトウェア著作物についての判断基準の SSO（構造，順序，構成：structure, sequence, and organization）（1986 年の Whelan v. Jaslow 事件で導入）に対応する。1 章で示した模型（プラモデル）の例えで説明すれば，宣言コードはプラモデルの組み立て前の個々のパーツの名称であって，部品の種別に応じて，部品を番号や記号で構造的な一覧表にしたものを Sun Java の SSO に対応すると説明することができる。

各タスクには，実装コード（implementing code）が存在し，プログラマがその実行を依頼すると，実装コードがコンピュータで実際にタスクを実行する。本件では，実装コードは Google が独自に作成しており，著作権侵害の争点にはなっていない。プログラマが，このような予め作成されたタスク実行プログラムを利用せずに複雑なソフトウェアを作成するのは，極めて困難であろう。

プログラマが実装コード・プログラムを選択，コンピュータにタスク実行を指示するには，プログラムの中で，特定のタスクに対応するコマンド（メソッド呼び出し）を取り込んで（宣言して）呼び出すよう書けばよい。これによって，実装コードに書かれたプログラムがタスクの実行される。Java に精通しているプログラマは，無数のタスクを呼び出すためのメソッド呼び出しを熟知している。

またプログラマが入力するメソッド呼び出しと特定のタスクを実行する実装コードの対応づけは API の中の「宣言コード」と呼ばれるコードが紐づけを行っている。この宣言コードは，各タスクの名称と API の組織システム全体の中でのそのタスクの配置（すなわち，あるクラスの中でのメソッドの配置，あるパッケージの中でのクラスの配置）を提供する。

すなわち，宣言コードは，Sun Java API では，プログラマに一連のショートカットを提供し，Java の開発者が多様なタスクを効率的に呼び出すためのグルーピングの思想が反映されていると言える。

（2） API の具体例とその動作

次に，地裁が API について説明した例（図 1，出典：最高裁判決文 AppendixB）を参考に，プログラ

マは API を使ってどのようにプログラムを書き，API はどのような動作をするか，を具体的に説明する。

プログラマは，プログラムの中で 2 つの整数のどちらが大きいかを判断したいと考えている。これを Java 言語で行うには，最初に java.lang と書く。これは Java で利用される最も基本的なクラスを集めたパッケージの名前を指している。続いて，プログラマは Math と書く。この単語は「クラス」を参照する。次に max と書く。この単語は「メソッド」である。次に，「( )」を作成し，括弧の間に，比較したい 2 つの整数，例えば 4 と 10 を入れる。式全体，つまりメソッド呼び出しは，「java.lang.Math.max（4，10)」となる。この式の API は，4，10 の大きい方の数を決定するタスク実装プログラムを呼び出す（図 1 左)。

API 側ではメソッド呼び出しは，所定のパッケージ（Package java.lang）の，所定のクラス（public class Math）の，メソッド（public static int max（int x,int y)）に対応する。この 3 行の全体が「宣言コード」である。このメソッドは，API 内部の実装コード（Implementing Code）に関連付けられている（破線矢印)。

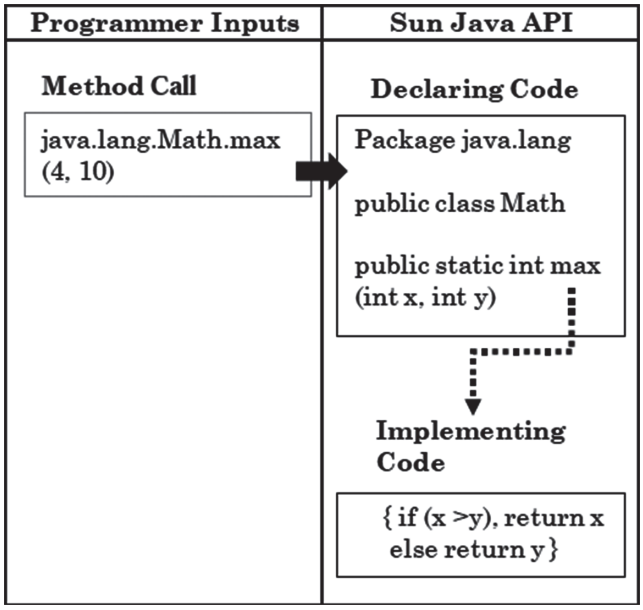

図 1　API の構造

実装コードは Google が予め Android の API 向けに作成しており，入力された x，y を比較して大きい方を戻り値として返す（if（x ＞ y)，return x else returny)（図 1 右)。

（原稿受領 2021.7.12）