

まえがき

この規格は、工業標準化法に基づいて、日本工業標準調査会の審議を経て、通商産業大臣が制定した日本工業規格である。

JIS X 3060 には、次に示す附属書がある。

附属書 1（参考） ECMAScript 言語の概説

附属書 2（参考） **ISO/IEC 16262** : 1998 Information technology — ECMAScript language specification

ECMAScript 言語

Information technology— ECMAScript language specification

序文 この規格は、1998 年に発行された **ISO/IEC 16262**, Information technology—ECMAScript language specification について、技術的内容を変更することなく日本工業規格として採用するために作成されたものであり、**1.～3.**については原国際規格の同項目を全文翻訳し、**4.**以降については、それぞれ原国際規格の同項目の内容を引用するものとした。ただし、原国際規格の **4.**は、“規定の一部ではない。”と原国際規格で明記されているが、解説として有用なので特に全文翻訳し、**附属書 1 (参考)**として添付した。

1. 適用範囲 この規格は、ECMAScript 言語を規定する。

2. 適合性 ECMAScript に適合する実装は、この規格で規定する、型、値、オブジェクト、特性、関数及びプログラム構文のすべてを提供しなければならない。

この規格に適合する実装は、Unicode™ 標準の第 2.0 版に適合する文字符号、又は **JIS X 0221** が実装水準 3 の 2 オクテット BMP 形式として規定する文字符号を、解釈できなければならない。実装が、採用する文字集合として、**JIS X 0221** の部分集合を特に指定しなかった場合は、BMP 部分集合である部分集合用図形文字の組 300 が指定されているものとする。

ECMAScript に適合する実装は、この規格で規定する以外の付加的な型、値、オブジェクト、特性及び関数を提供してもよい。特に、ECMAScript に適合する実装は、この規格で規定しない特性及びその値を、この規格で規定するオブジェクトに対して提供してもよい。

ECMAScript に適合する実装は、この規格で規定しないプログラム構文を提供してもよい。特に、ECMAScript に適合する実装は、この規格の **7.4.3** に挙げる“将来使用する予約語”を使用するプログラム構文を提供してもよい。

3. 引用規格 次に掲げる規格は、この規格に引用されることによって、この規格の一部を構成する。これらの引用規格のうちで、発行年を付記してあるものは、記載の年の版だけがこの規格の規定を構成するものであって、その後の改正版・追補には適用しない。発行年を付記していない引用規格は、追補を含むその最新版を適用する。

— **JIS X 0201** 7ビット及び8ビットの情報交換用符号化文字集合

備考 **ISO/IEC 646** : 1991, Information technology—ISO 7-bit coded character set for information interchange に、**JIS X 0201** が整合している。

— **JIS X 0221** 国際符号化文字集合 (UCS) — 第 1 部 体系及び基本多言語面

備考 ISO/IEC 10646-1 : 1993, Information Technology—Universal Multiple—Octet Coded Character Set (UCS) —Part 1 : Architecture and Basic Multilingual Plane が, **JIS X 0221** と一致している。

—**JIS X 3010** プログラム言語 C

備考 ISO/IEC 9899 : 1990, Programming languages—C が, **JIS X 3010** と一致している。

—**ANSI/IEEE Std 754-1985**, IEEE Standard for Binary Floating—Point Arithmetic. Institute of Electrical and Electronics Engineers, New York (1985).

—**Unicode Inc.** (1996), The Unicode™ Standard, Version 2.0. ISBN : 0-201-48345-9, Addison—Wesley Publishing Co., Menlo Park, California.

4. **概要** ISO/IEC 16262 : 1998 の 4.Overview による。
5. **記法** ISO/IEC 16262 : 1998 の 5.Notational Conventions による。
6. **ソーステキスト** ISO/IEC 16262 : 1998 の 6.Source Text による。
7. **字句** ISO/IEC 16262 : 1998 の 7.Lexical Conventions による。
8. **型** ISO/IEC 16262 : 1998 の 8.Types による。
9. **型変換** ISO/IEC 16262 : 1998 の 9.Type Conversion による。
10. **実行文脈** ISO/IEC 16262 : 1998 の 10.Execution Contexts による。
11. **式** ISO/IEC 16262 : 1998 の 11.Expressions による。
12. **文** ISO/IEC 16262 : 1998 の 12.Statements による。
13. **関数定義** ISO/IEC 16262 : 1998 の 13.Function Definition による。
14. **プログラム** ISO/IEC 16262 : 1998 の 14.Program による。
15. **標準固有オブジェクト** ISO/IEC 16262 : 1998 の 15.Native ECMAScript objects による。
16. **誤り** ISO/IEC 16262 : 1998 の 16.Errors による。

附属書 1 (参考) ECMAScript 言語の概説

この附属書は、原国際規格 ISO/IEC 16262 : 1998, Information technology – ECMAScript language specification の“規定の一部ではない。”と明記された“4.Overview”を、ECMAScript の概要理解のために、参考として完全翻訳したものであって、規定の一部ではない。

1. 概観 ECMAScript は、ホスト環境の中で、処理を行い、処理オブジェクトを操作するためのオブジェクト指向言語である。この規格で規定する ECMAScript は、それ自体で処理が完結することを意図していない。実際、この規格には、外部データの入力及び処理結果の出力についての規定が存在しない。その代わりに、ECMAScript プログラムの処理環境が、この規格が規定するオブジェクト及び種々の機能に加えて、その環境に固有のホストオブジェクトを提供することになっている。ホストオブジェクトは、この規格の規定対象外ではあるが、ECMAScript プログラムから扱える幾つかの特性と呼出し可能な幾つかの関数とを提供することになる。

スクリプト言語とは、既存のシステムがもつ機能进行操作し、調整し、自動化するためのプログラム言語を指す。このようなシステムでは、多くの機能がユーザインタフェースとして用意されていて、それらの機能をプログラム制御するための機構としてスクリプト言語は動作する。この意味で、既存のシステムは、オブジェクト及び機能を与えるホスト環境となつて、スクリプト言語の能力を補完する。スクリプト言語は、専門のプログラマが使うだけでなく、専門ではないプログラマも使うことを想定しているので、格式張らない形が幾つも取り入れてある。

ECMAScript は、もともとウェブ用のスクリプト言語として設計されたもので、ウェブページのブラウザに能動性を与える機能、及びウェブを用いたクライアントサーバ構成でのサーバ処理を行う機能をもつ。ECMAScript は、様々なホスト環境に対してのスクリプト能力をもっているので、この規格では、核となるスクリプト言語を特定のホスト環境に依存しない形で規定する。

ECMAScript の幾つかの機能は、他のプログラム言語の機能に類似する。特に、Java™ 言語及び Self 言語に類似する。これらの言語は、次に示す文献に詳しい。

- Gosling, James, Bill Joy and Guy Steele. The Java Language Specification. Addison Wesley Publishing Co., 1996.
- Ungar, David, and Smith, Randall B. Self: The Power of Simplicity. OOPSLA'87 Conference Proceedings, pp.227-241, Orlando, FL, October, 1987.

2. ウェブのスクリプト ウェブブラウザは、クライアント側での ECMAScript のホスト環境となり、ウィンドウ、メニュー、ポップアップ、対話ボックス、テキスト領域、アンカ、フレーム、履歴、クッキー、入出力などを提供する。このホスト環境は、さらに、注目対象の移動、ページ及び画像のロード又はアンロード、誤り及び異常終了、選択、フォーム送信、マウス動作などの事象に、スクリプトプログラムを連動させる手段も提供する。スクリプトプログラムを HTML ページの中を書くことによって、ユーザインタフェース要素に加え、固定したテキスト及び画像だけでなく、処理を加えたテキスト及び画像も含むページを表示できる。こうしたスクリプトプログラムは、ユーザからの反応に呼応して動作するので、いわゆる主プログラムを必要としない。

ウェブサーバは、サーバ側での ECMAScript のクライアント側とは別のホスト環境となり、要求、クラ

イアント、ファイルなどを表すオブジェクトを提供し、データを共有したりロックしたりするための機能を提供する。クライアント側のスクリプトもサーバ側のスクリプトも用いることで、クライアントとサーバとにまたがった分散処理を行いながら、ウェブに基づく応用システムでの調整されたユーザインタフェースが提供できるようになる。

ECMAScript 環境を提供するウェブブラウザ及びサーバは、それぞれに固有のホスト環境を提供することで、ECMAScript 実行環境を補完する。

3. 言語の概説 ECMAScript は、オブジェクトに基づく。すなわち、基本言語機能及びホスト機能をオブジェクトが提供するので、ECMAScript プログラムは、通信し合うオブジェクトの集まりの形をとる。ECMAScript オブジェクトは、名前をもった特性の集まりをいう。各特性は、その使用方法を決定する 0 個以上の属性をもつ。例えば、特性に対する `ReadOnly`（読み専用）属性が `true`（真）に設定されている場合、実行中の ECMAScript プログラムがその特性の値を変更しようとしても変更が生じない。特性は、ほかのオブジェクト、基本値又はメソッドを入れておく容器として機能する。基本値とは、幾つかの組込み型の要素をいう。基本値を要素とする組込み型に、`Undefined`, `Null`, `Boolean`, `Number` 及び `String` がある。組込み型には、もう一つ、`Object` があり、オブジェクトは、すべて、その要素となる。メソッドとは、特性に入っている関数をいう。

ECMAScript には、一群の組込みオブジェクトが用意してある。これらは、ECMAScript の実体を定義付ける性格をもつ。組込みオブジェクトには、`Global` オブジェクト、`Object` オブジェクト、`Function` オブジェクト、`Array` オブジェクト、`String` オブジェクト、`Boolean` オブジェクト、`Number` オブジェクト、`Math` オブジェクト及び `Date` オブジェクトがある。

ECMAScript には、演算子が組み込んである。演算子は、厳密にいうと、関数でもメソッドでもない。ECMAScript 演算子には、単項演算子、乗法演算子、加法演算子、ビット単位のシフト演算子、関係演算子、等価演算子、ビット単位演算子、論理演算子、代入演算子及びコンマ演算子がある。

ECMAScript 構文は、意図的に Java 構文に似せてある。しかしながら、規則をゆるめ、スクリプト言語として簡易に使える工夫が施してある。例えば、変数の型を宣言する必要はないし、特性を型付ける必要もない。ユーザ定義関数にも、宣言の位置より後ろでないと呼び出してはならないという縛りが無い。

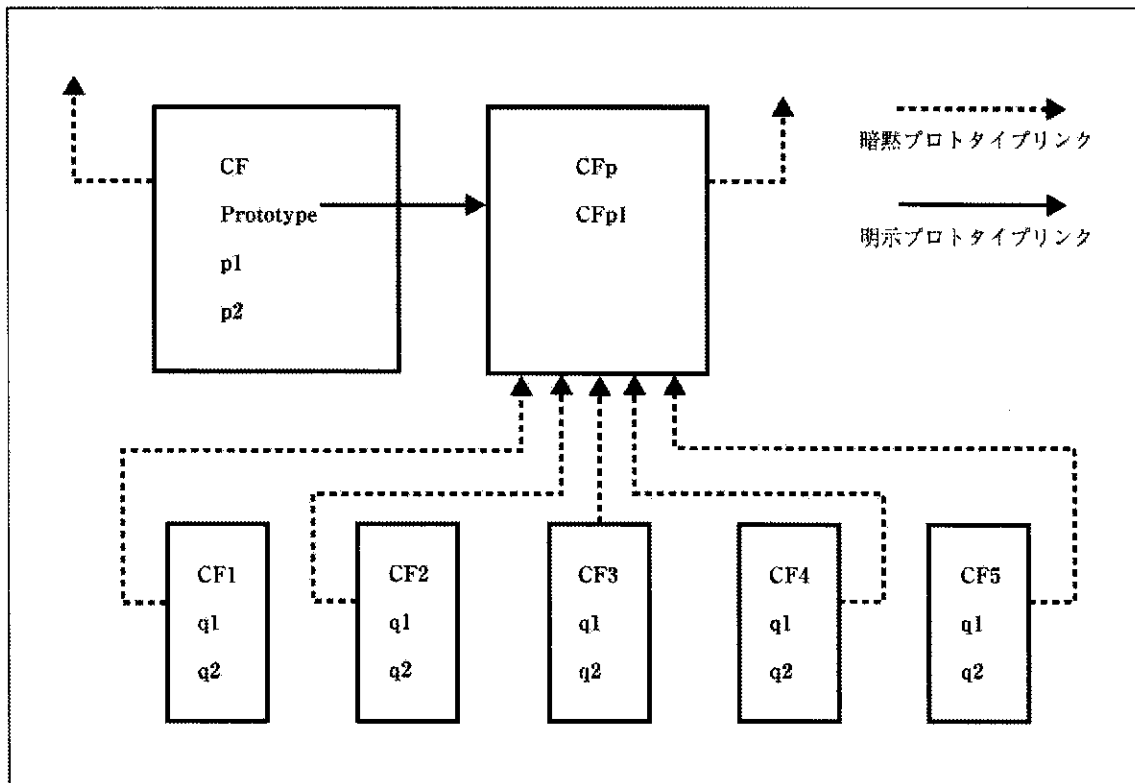
3.1 オブジェクト ECMAScript には、C++、Smalltalk、Java などとは違って、本来のクラスがない。代わりに、オブジェクトを生成するためのコンストラクタ（`constructor`、構築子ともいう。）を置く。コンストラクタは、関数であり、それを実行することで、オブジェクトへの記憶域の割付けとオブジェクトの初期設定とが生じる。オブジェクトの初期設定とは、オブジェクトの特性の全部又は一部に初期値を与えることをいう。コンストラクタを含めて、すべての関数はオブジェクトだが、すべてのオブジェクトがコンストラクタというわけではない。すべてのコンストラクタは、特性 `Prototype` をもつ。この特性は、プロトタイプに基づく継承及び共有特性を実装するために用いる。オブジェクトは、コンストラクタを用いた `new` 式によって生成する。例えば、`new String("A String")` は、新しい文字列オブジェクトを生成する。`new` 式を使わずにコンストラクタを呼び出した場合の結果は、コンストラクタに依存する。例えば、`String("A String")` は、基本値の文字列を生成し、オブジェクトを生成しない。

ECMAScript では、プロトタイプに基づく継承が行われる。すべてのコンストラクタは付随するプロトタイプをもつ。コンストラクタによって生成されたオブジェクトは、すべて、そのコンストラクタに付随するプロトタイプ（これをオブジェクトのプロトタイプという。）への暗黙参照をもつ。さらに、そのプロトタイプが、`null` ではない暗黙参照をもち、それ自体のプロトタイプを指していることがある。この関係

の繰返しを、プロトタイプ連鎖という。オブジェクトに対する特性の参照は、プロトタイプ連鎖の中で最初に現れる、その名前の特性をもつオブジェクトのその特性への参照となる。すなわち、まず、直接に指定されているオブジェクトについて、その特性が調べられる。そのオブジェクトがその名前の特性をもっている場合は、その特性が参照される。オブジェクトがその名前の特性をもたない場合には、次にそのオブジェクトのプロトタイプが調べられる。以下同様に調べられていく。

クラスに基づくオブジェクト指向言語では、一般に、状態はインスタンスが保持し、メソッドはクラスが保持しているので、構造と振舞いだけが継承の対象となる。ECMAScript では、状態もメソッドもオブジェクトが保持していて、構造、振舞い及び状態のすべてが継承の対象となる。

プロトタイプどうしは、プロトタイプがもつ特性について、それと同じ特性を自らもっているのではない限り、それらの特性及びその値を共有することになる。附属書 1 図 1 にこの状況を例示する。



附属書 1 図 1 ECMAScript の継承関係

CF は、コンストラクタとする。これは、オブジェクトでもある。CF1, CF2, CF3, CF4 及び CF5 の五つのオブジェクトを new 式で生成したとする。これらオブジェクトの各々は、q1 及び q2 という名前の付いた特性をもつ。破線は、プロトタイプの暗黙参照を示す。例えば、CF3 のプロトタイプは、CFp となる。コンストラクタ CF は、それ自体、p1 及び p2 という名前の付いた二つの特性をもつ。これらの特性は、CFp, CF1, CF2, CF3, CF4 及び CF5 には見えない。CFp の中の CFp1 という名前の付いた特性は、CF1, CF2, CF3, CF4 及び CF5 によって共有される。同様に、CFp の暗黙プロトタイプ連鎖の中に存在する、名前が q1, q2 及び CFp1 ではない任意の特性も共有される。CFp と CF との間にはプロトタイプの暗黙参照リンクが存在しないことに注意すること。

クラスに基づくオブジェクト言語と違って、オブジェクトには、動的に特性を追加することができる。それには、その特性に値を代入するだけでよい。すなわち、コンストラクタがオブジェクトを生成する際

に、オブジェクトのすべての特性に名前を付けたり、値を代入したりしておく必要はない。**附属書 1 図 1** では、CFp の特性に新しい値を割り当てることで、CF1、CF2、CF3、CF4 及び CF5 に対する新しい共有特性を追加できる。

4. 定義 この**附属書**で用いる主な用語の定義の大略は、次のとおりとする。

4.1 型 (type) データ値の集合。

4.2 基本値 (primitive value) Undefined 型、Null 型、Boolean 型、Number 型又は String 型の要素。基本値は、処理系の最下層で直接にデータ値を表現する。

4.3 オブジェクト (object) Object 型の要素。個々のオブジェクトは、特性の集まりからなる。特性は、基本値、オブジェクト又は関数を保持する。オブジェクトの特性の中で関数を保持する特性を、そのオブジェクトのメソッドという。

4.4 コンストラクタ、構築子 (constructor) オブジェクトを生成し初期設定する関数。コンストラクタは、付随するプロトタイプオブジェクトをもつ。プロトタイプは、継承及び特性共有を実装するのに用いる。

4.5 プロトタイプ (prototype) ECMAScript における、構造、状態及び振舞いの継承を実装する際に用いるオブジェクト。オブジェクトは、そのオブジェクトを生成したコンストラクタの付随プロトタイプへの暗黙参照をもつ。

この暗黙参照は、特性の参照の解決に用いる。コンストラクタ c の付随プロトタイプは、プログラムのうえでは、c.prototype と書いて参照する。オブジェクトのプロトタイプに特性を追加すると、継承の仕組みによって、同じプロトタイプをもつオブジェクトが共有することになる。

4.6 固有オブジェクト (native object) ECMAScript の処理系がホスト環境によらず提供するオブジェクト。標準の固有オブジェクトは、この規格で規定する。固有オブジェクトには、組込みオブジェクトもあれば、ECMAScript の実行中にコンストラクタを使って生成されるオブジェクトもある。

4.7 組込みオブジェクト (built-in object) ECMAScript の処理系がホスト環境によらず提供し、ECMAScript プログラムの実行開始時点に存在しているオブジェクト。標準の組込みオブジェクトは、この規格で規定する。ECMAScript の処理系は、その他の組込みオブジェクトを定義してもよい。すべての組込みオブジェクトは、固有オブジェクトとなる。

4.8 ホストオブジェクト (host object) ECMAScript の実行環境を補完する形でホスト環境が提供するオブジェクト。固有オブジェクトでないオブジェクトは、ホストオブジェクトとなる。

4.9 不定値 (undefined value) 値を代入していない変数がもつ基本値。

4.10 Undefined 型 (Undefined type) 基本値である不定値だけを要素とする集合。

4.11 ナル値 (null value) 空参照を表す基本値。プログラムのうえでは、null と書く。

4.12 Null 型 (Null type) 基本値 null だけを要素とする集合。

4.13 論理値 (boolean value) 基本値 true 及び false。true は論理での真に対応し、false は偽に対応する。

4.14 Boolean 型 (Boolean type) 基本値 true 及び false だけを要素とする集合。

4.15 論理オブジェクト (boolean object) 組込みオブジェクト Boolean のインスタンスである Object 型の要素。言い換えると、new 式でコンストラクタ Boolean を指定し、その引数に論理値を与えて生成したオブジェクト。生成されたオブジェクトは、その暗黙の（名前をもたない）特性として、その論理値をもつ。論理オブジェクトは、その論理値に強制的に型変換されるので、論理値が必要となる位置に書くこともできる。

この強制的な型変換は、ECMAScript のもつ便宜性の一つである。その目的は、どんな経歴のプログラマにも対応することにある。手続き型言語又は命令型言語に慣れたプログラマにとっては、論理値、文字列値、数値などを使うのが自然であるし、オブジェクト指向言語に慣れたプログラマにとっては、論理オブジェクト、文字列オブジェクト、数値オブジェクトなどを使うのが直観に合う。

4.16 文字列値 (string value) String 型の要素。個々の文字列値は、Unicode 文字⁽¹⁾の有限列 (長さ 0 以上) からなる。

注(1) この規格でいう Unicode 文字とは、Unicode 標準が定義する文字のことであって、符号化文字データ要素 (coded-character-data-element) ではないことに注意。したがって、文字合成列 (JIS X 0221を参照。) が単一の文字として取り扱われることはない。

なお、原国際規格の制定時、ISO/IEC 10646 及び Unicode 標準において、BMP 以外の面への文字の定義が行われていなかったため、この規格においては、16 ビットの符号なし整数値として表現される Unicode 符号値と、一つ以上の Unicode 符号値からなる列によって符号化される Unicode 文字との、明確な区別がなされていない。この規格における String 型の要素は、実際には、Unicode 文字ではなく、Unicode 符号値と解釈するほうがより正確である。

4.17 String 型 (String type) Unicode 文字の有限列 (長さ 0 以上) のすべてからなる集合。

4.18 文字列オブジェクト (string object) 組込みオブジェクト String のインスタンスである Object 型の要素。言い換えると、new 式でコンストラクタ String を指定し、その引数に文字列値を与えて生成したオブジェクト。生成されたオブジェクトは、その暗黙の (名前をもたない) 特性として、その文字列値をもつ。文字列オブジェクトは、その文字列値に強制的に型変換されるので、文字列値が必要となる位置に書くこともできる。

4.19 数値 (number value) Number 型の要素。1 個の数値を直接に表現する。

4.20 Number 型 (Number type) 数値の集合。ECMAScript での数値は、IEEE754 の 64 ビット形式倍精度浮動小数点表現を用いて表す。特殊値として、正の無限大、負の無限大及び“非数” (“Not-a-Number”, NaN) も含む。

4.21 数値オブジェクト (number object) 組込みオブジェクト Number のインスタンスである Object 型の要素。言い換えると、new 式でコンストラクタ Number を指定し、その引数に数値を与えて生成したオブジェクト。生成されたオブジェクトは、その暗黙の (名前をもたない) 特性として、その数値をもつ。数値オブジェクトは、その数値に強制的に型変換されるので、数値が必要となる位置に書くこともできる。数値オブジェクトは、特性をプロトタイプ Number に加えることで、その特性を共有特性としてもつことができる。

4.22 Infinity (Infinity) 数値の一つ。正の無限大を表す。

4.23 NaN (Not-a-Number) 数値の一つ。IEEE754 での“非数”を表す。

原案作成委員会 構成表

	氏名			所属
(主査)	黒	川	利 明	株式会社 CSK EC 開発本部 EC リサーチ
	石	渡	克 己	日本ネットスケープコミュニケーション株式会社マーケティング統括部
	内	山	光 一	株式会社東芝デジタルメディア機器社コンピュータ&ネットワーク開発センター開発第3部
	笥		捷 彦	早稲田大学理工学部情報学科
	木	戸	彰 夫	日本アイ・ビー・エム株式会社ソフトウェア開発研究所
	後	藤	志津雄	株式会社日立製作所ソフトウェア事業部言語図形設計部
	豊	田	祐 司	マイクロソフト株式会社研究開発本部デベロッパー製品開発統括部
	平	山	亮	金沢工業大学情報工学科
	福	留	治 隆	株式会社アスキー株式会社メディア技術開発室
	村	田	晴 久	日本ユニシス株式会社生産技術部技術1室
(工業技術院)	田	川	淳	通商産業省工業技術院標準部標準業務課情報電気標準化推進室
(事務局)	橋	本	孔 佐	財団法人日本規格協会情報技術標準化研究センター