

---

---

**Information technology — Programming  
languages, their environments and  
system software interfaces —  
ECMAScript language specification**

*Technologies de l'information — Langages de programmation, leurs  
environnements et interfaces de logiciel système — Spécification du  
langage ECMAScript*

Withdrawn

Withdrawn



**COPYRIGHT PROTECTED DOCUMENT**

© ISO/IEC 2011

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office  
Case postale 56 • CH-1211 Geneva 20  
Tel. + 41 22 749 01 11  
Fax + 41 22 749 09 47  
E-mail [copyright@iso.org](mailto:copyright@iso.org)  
Web [www.iso.org](http://www.iso.org)

Published in Switzerland

## Contents

Page

Foreword .....	viii
Introduction.....	ix
1 Scope .....	1
2 Conformance.....	1
3 Normative references .....	1
4 Overview .....	1
4.1 Web Scripting.....	2
4.2 Language Overview .....	2
4.2.1 Objects.....	3
4.2.2 The Strict Variant of ECMAScript .....	4
4.3 Terms and definitions.....	4
5 Notational Conventions .....	8
5.1 Syntactic and Lexical Grammars.....	8
5.1.1 Context-Free Grammars .....	8
5.1.2 The Lexical and RegExp Grammars .....	8
5.1.3 The Numeric String Grammar .....	8
5.1.4 The Syntactic Grammar .....	8
5.1.5 The JSON Grammar .....	9
5.1.6 Grammar Notation .....	9
5.2 Algorithm Conventions .....	12
6 Source Text .....	13
7 Lexical Conventions .....	14
7.1 Unicode Format-Control Characters .....	14
7.2 White Space.....	15
7.3 Line Terminators .....	15
7.4 Comments .....	16
7.5 Tokens .....	17
7.6 Identifier Names and Identifiers.....	17
7.6.1 Reserved Words.....	18
7.7 Punctuators .....	19
7.8 Literals .....	20
7.8.1 Null Literals .....	20
7.8.2 Boolean Literals.....	20
7.8.3 Numeric Literals.....	20
7.8.4 String Literals.....	22
7.8.5 Regular Expression Literals.....	25
7.9 Automatic Semicolon Insertion .....	26
7.9.1 Rules of Automatic Semicolon Insertion .....	26
7.9.2 Examples of Automatic Semicolon Insertion.....	27
8 Types .....	28
8.1 The Undefined Type.....	28
8.2 The Null Type .....	28
8.3 The Boolean Type .....	29
8.4 The String Type.....	29
8.5 The Number Type.....	29
8.6 The Object Type .....	30
8.6.1 Property Attributes .....	30
8.6.2 Object Internal Properties and Methods .....	31

8.7	The Reference Specification Type .....	35
8.7.1	GetValue (V) .....	35
8.7.2	PutValue (V, W) .....	36
8.8	The List Specification Type .....	36
8.9	The Completion Specification Type .....	36
8.10	The Property Descriptor and Property Identifier Specification Types .....	37
8.10.1	IsAccessorDescriptor ( Desc ) .....	37
8.10.2	IsDataDescriptor ( Desc ) .....	37
8.10.3	IsGenericDescriptor ( Desc ) .....	37
8.10.4	FromPropertyDescriptor ( Desc ) .....	38
8.10.5	ToPropertyDescriptor ( Obj ) .....	38
8.11	The Lexical Environment and Environment Record Specification Types .....	39
8.12	Algorithms for Object Internal Methods .....	39
8.12.1	[[GetOwnProperty]] ( P ) .....	39
8.12.2	[[GetProperty]] ( P ) .....	39
8.12.3	[[Get]] ( P ) .....	39
8.12.4	[[CanPut]] ( P ) .....	39
8.12.5	[[Put]] ( P, V, Throw ) .....	40
8.12.6	[[HasProperty]] ( P ) .....	40
8.12.7	[[Delete]] ( P, Throw ) .....	41
8.12.8	[[DefaultValue]] ( hint ) .....	41
8.12.9	[[DefineOwnProperty]] ( P, Desc, Throw ) .....	41
9	Type Conversion and Testing .....	43
9.1	ToPrimitive .....	43
9.2	ToBoolean .....	43
9.3	ToNumber .....	43
9.3.1	ToNumber Applied to the String Type .....	44
9.4	ToInteger .....	46
9.5	ToInt32: (Signed 32 Bit Integer) .....	47
9.6	ToUint32: (Unsigned 32 Bit Integer) .....	47
9.7	ToUint16: (Unsigned 16 Bit Integer) .....	47
9.8	ToString .....	48
9.8.1	ToString Applied to the Number Type .....	48
9.9	ToObject .....	49
9.10	CheckObjectCoercible .....	49
9.11	IsCallable .....	49
9.12	The SameValue Algorithm .....	50
10	Executable Code and Execution Contexts .....	50
10.1	Types of Executable Code .....	50
10.1.1	Strict Mode Code .....	51
10.2	Lexical Environments .....	51
10.2.1	Environment Records .....	51
10.2.2	Lexical Environment Operations .....	56
10.2.3	The Global Environment .....	56
10.3	Execution Contexts .....	56
10.3.1	Identifier Resolution .....	57
10.4	Establishing an Execution Context .....	57
10.4.1	Entering Global Code .....	58
10.4.2	Entering Eval Code .....	58
10.4.3	Entering Function Code .....	58
10.5	Declaration Binding Instantiation .....	59
10.6	Arguments Object .....	60
11	Expressions .....	63
11.1	Primary Expressions .....	63
11.1.1	The <code>this</code> Keyword .....	63
11.1.2	Identifier Reference .....	63
11.1.3	Literal Reference .....	63
11.1.4	Array Initialiser .....	63

11.1.5	Object Initialiser .....	65
11.1.6	The Grouping Operator .....	67
11.2	Left-Hand-Side Expressions .....	67
11.2.1	Property Accessors .....	67
11.2.2	The new Operator .....	68
11.2.3	Function Calls .....	68
11.2.4	Argument Lists.....	69
11.2.5	Function Expressions.....	69
11.3	Postfix Expressions.....	69
11.3.1	Postfix Increment Operator.....	70
11.3.2	Postfix Decrement Operator.....	70
11.4	Unary Operators.....	70
11.4.1	The delete Operator.....	70
11.4.2	The void Operator .....	71
11.4.3	The typeof Operator.....	71
11.4.4	Prefix Increment Operator.....	71
11.4.5	Prefix Decrement Operator .....	72
11.4.6	Unary + Operator .....	72
11.4.7	Unary - Operator .....	72
11.4.8	Bitwise NOT Operator ( ~ ) .....	72
11.4.9	Logical NOT Operator ( ! ) .....	73
11.5	Multiplicative Operators .....	73
11.5.1	Applying the * Operator .....	73
11.5.2	Applying the / Operator .....	74
11.5.3	Applying the % Operator .....	74
11.6	Additive Operators.....	75
11.6.1	The Addition operator ( + ) .....	75
11.6.2	The Subtraction Operator ( - ) .....	75
11.6.3	Applying the Additive Operators to Numbers.....	75
11.7	Bitwise Shift Operators .....	76
11.7.1	The Left Shift Operator ( << ) .....	76
11.7.2	The Signed Right Shift Operator ( >> ) .....	76
11.7.3	The Unsigned Right Shift Operator ( >>> ) .....	77
11.8	Relational Operators.....	77
11.8.1	The Less-than Operator ( < ) .....	77
11.8.2	The Greater-than Operator ( > ) .....	78
11.8.3	The Less-than-or-equal Operator ( <= ) .....	78
11.8.4	The Greater-than-or-equal Operator ( >= ) .....	78
11.8.5	The Abstract Relational Comparison Algorithm.....	78
11.8.6	The instanceof operator .....	79
11.8.7	The in operator.....	79
11.9	Equality Operators .....	80
11.9.1	The Equals Operator ( == ) .....	80
11.9.2	The Does-not-equals Operator ( != ) .....	80
11.9.3	The Abstract Equality Comparison Algorithm.....	80
11.9.4	The Strict Equals Operator ( === ) .....	81
11.9.5	The Strict Does-not-equal Operator ( !== ) .....	81
11.9.6	The Strict Equality Comparison Algorithm .....	82
11.10	Binary Bitwise Operators .....	82
11.11	Binary Logical Operators .....	83
11.12	Conditional Operator ( ? : ) .....	84
11.13	Assignment Operators .....	84
11.13.1	Simple Assignment ( = ) .....	85
11.13.2	Compound Assignment ( op= ) .....	85
11.14	Comma Operator ( , ) .....	85
12	Statements.....	86
12.1	Block.....	86

12.2	Variable Statement.....	87
12.2.1	Strict Mode Restrictions.....	88
12.3	Empty Statement.....	88
12.4	Expression Statement.....	89
12.5	The if Statement.....	89
12.6	Iteration Statements.....	89
12.6.1	The do-while Statement.....	90
12.6.2	The while Statement.....	90
12.6.3	The for Statement.....	90
12.6.4	The for-in Statement.....	91
12.7	The continue Statement.....	92
12.8	The break Statement.....	93
12.9	The return Statement.....	93
12.10	The with Statement.....	93
12.10.1	Strict Mode Restrictions.....	94
12.11	The switch Statement.....	94
12.12	Labelled Statements.....	96
12.13	The throw Statement.....	96
12.14	The try Statement.....	96
12.14.1	Strict Mode Restrictions.....	97
12.15	The debugger statement.....	97
13	Function Definition.....	98
13.1	Strict Mode Restrictions.....	99
13.2	Creating Function Objects.....	99
13.2.1	[[Call]].....	100
13.2.2	[[Construct]].....	100
13.2.3	The [[ThrowTypeError]] Function Object.....	100
14	Program.....	101
14.1	Directive Prologues and the Use Strict Directive.....	101
15	Standard Built-in ECMAScript Objects.....	102
15.1	The Global Object.....	103
15.1.1	Value Properties of the Global Object.....	103
15.1.2	Function Properties of the Global Object.....	104
15.1.3	URI Handling Function Properties.....	105
15.1.4	Constructor Properties of the Global Object.....	110
15.1.5	Other Properties of the Global Object.....	111
15.2	Object Objects.....	111
15.2.1	The Object Constructor Called as a Function.....	111
15.2.2	The Object Constructor.....	112
15.2.3	Properties of the Object Constructor.....	112
15.2.4	Properties of the Object Prototype Object.....	115
15.2.5	Properties of Object Instances.....	117
15.3	Function Objects.....	117
15.3.1	The Function Constructor Called as a Function.....	117
15.3.2	The Function Constructor.....	117
15.3.3	Properties of the Function Constructor.....	118
15.3.4	Properties of the Function Prototype Object.....	118
15.3.5	Properties of Function Instances.....	121
15.4	Array Objects.....	122
15.4.1	The Array Constructor Called as a Function.....	122
15.4.2	The Array Constructor.....	123
15.4.3	Properties of the Array Constructor.....	123
15.4.4	Properties of the Array Prototype Object.....	124
15.4.5	Properties of Array Instances.....	140
15.5	String Objects.....	141
15.5.1	The String Constructor Called as a Function.....	141
15.5.2	The String Constructor.....	142

15.5.3	Properties of the String Constructor .....	142
15.5.4	Properties of the String Prototype Object.....	142
15.5.5	Properties of String Instances .....	152
15.6	Boolean Objects.....	152
15.6.1	The Boolean Constructor Called as a Function.....	152
15.6.2	The Boolean Constructor .....	152
15.6.3	Properties of the Boolean Constructor .....	153
15.6.4	Properties of the Boolean Prototype Object.....	153
15.6.5	Properties of Boolean Instances .....	154
15.7	Number Objects .....	154
15.7.1	The Number Constructor Called as a Function .....	154
15.7.2	The Number Constructor.....	154
15.7.3	Properties of the Number Constructor.....	154
15.7.4	Properties of the Number Prototype Object.....	155
15.7.5	Properties of Number Instances .....	159
15.8	The Math Object .....	159
15.8.1	Value Properties of the Math Object.....	159
15.8.2	Function Properties of the Math Object .....	161
15.9	Date Objects .....	165
15.9.1	Overview of Date Objects and Definitions of Abstract Operators.....	165
15.9.2	The Date Constructor Called as a Function .....	171
15.9.3	The Date Constructor .....	171
15.9.4	Properties of the Date Constructor.....	172
15.9.5	Properties of the Date Prototype Object .....	173
15.9.6	Properties of Date Instances.....	181
15.10	RegExp (Regular Expression) Objects.....	181
15.10.1	Patterns .....	181
15.10.2	Pattern Semantics.....	183
15.10.3	The RegExp Constructor Called as a Function .....	195
15.10.4	The RegExp Constructor.....	195
15.10.5	Properties of the RegExp Constructor .....	196
15.10.6	Properties of the RegExp Prototype Object.....	196
15.10.7	Properties of RegExp Instances .....	198
15.11	Error Objects.....	198
15.11.1	The Error Constructor Called as a Function.....	199
15.11.2	The Error Constructor.....	199
15.11.3	Properties of the Error Constructor.....	199
15.11.4	Properties of the Error Prototype Object .....	199
15.11.5	Properties of Error Instances.....	200
15.11.6	Native Error Types Used in This Standard.....	200
15.11.7	<i>NativeError</i> Object Structure.....	201
15.12	The JSON Object.....	203
15.12.1	The JSON Grammar.....	203
15.12.2	parse ( text [ , reviver ] ).....	204
15.12.3	stringify ( value [ , replacer [ , space ] ] ).....	206
16	Errors.....	209
Annex A	(informative) Grammar Summary .....	211
Annex B	(informative) Compatibility .....	230
Annex C	(informative) The Strict Mode of ECMAScript.....	234
Annex D	(informative) Corrections and Clarifications in the 3 <sup>rd</sup> Edition with Possible 2 <sup>nd</sup> Edition Compatibility Impact.....	236
Annex E	(informative) Additions and Changes in the 3 <sup>rd</sup> Edition that Introduce Incompatibilities with the 2 <sup>nd</sup> Edition .....	237
Bibliography	.....	240

## Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 16262 was prepared by Ecma International (as ECMA-262) and was adopted, under a special “fast-track procedure”, by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, in parallel with its approval by national bodies of ISO and IEC.

This third edition cancels and replaces the second edition (ISO/IEC 16262:2002), which has been technically revised.

Withhold



## Introduction

This International Standard is based on several originating technologies, the most well-known being JavaScript (Netscape) and JScript (Microsoft). The language was invented by Brendan Eich at Netscape and first appeared in that company's Navigator 2.0 browser. It has appeared in all subsequent browsers from Netscape and in all browsers from Microsoft starting with Internet Explorer 3.0.

The development of this International Standard started in November 1996. The first edition of this International Standard was adopted by the Ecma General Assembly of June 1997.

That International Standard was submitted to ISO/IEC JTC 1 for adoption under the fast-track procedure, and approved as ISO/IEC 16262, first edition, in April 1998.

The second edition of this International Standard introduced powerful regular expressions, better string handling, new control statements, try/catch exception handling, tighter definition of errors, formatting for numeric output and minor changes in anticipation of forthcoming internationalization facilities and future language growth. The second edition of the ECMAScript standard was published as ISO/IEC 16262 in June 2002.

Since publication of the second edition of ISO/IEC 16262:2002, ECMAScript has achieved massive adoption in conjunction with the World Wide Web where it has become the programming language that is supported by essentially all web browsers. Significant work was done to develop a third edition of ECMAScript. Although that work was not completed and not published as a new edition of ECMAScript, it informs continuing evolution of the language. The present third edition of ISO/IEC 16262 (published as ECMA-262 5th edition) codifies de facto interpretations of the language specification that have become common among browser implementations and adds support for new features that have emerged since the publication of the third edition. Such features include accessor properties, reflective creation and inspection of objects, program control of property attributes, additional array manipulation functions, support for the JSON object encoding format, and a strict mode that provides enhanced error checking and program security.

ECMAScript is a vibrant language and the evolution of the language is not complete. Significant technical enhancement will continue with future editions of this International Standard.

# Information technology — Programming languages, their environments and system software interfaces — ECMAScript language specification

## 1 Scope

This International Standard defines the ECMAScript scripting language.

## 2 Conformance

A conforming implementation of ECMAScript must provide and support all the types, values, objects, properties, functions, and program syntax and semantics described in this International Standard.

A conforming implementation of this International Standard shall interpret characters in conformance with the Unicode Standard, Version 3.0 or later, and ISO/IEC 10646 with either UCS-2 or UTF-16 as the adopted encoding form, implementation level 3. If the adopted ISO/IEC 10646 subset is not otherwise specified, it is presumed to be the BMP subset, collection 300. If the adopted encoding form is not otherwise specified, it is presumed to be the UTF-16 encoding form.

A conforming implementation of ECMAScript is permitted to provide additional types, values, objects, properties, and functions beyond those described in this International Standard. In particular, a conforming implementation of ECMAScript is permitted to provide properties not described in this International Standard, and values for those properties, for objects that are described in this International Standard.

A conforming implementation of ECMAScript is permitted to support program and regular expression syntax not described in this International Standard. In particular, a conforming implementation of ECMAScript is permitted to support program syntax that makes use of the "future reserved words" listed in 7.6.1.2 of this International Standard.

## 3 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 10646, *Universal Coded Character Set (UCS)*