
Information technology — Programming languages — Ruby

Technologies de l'information — Langages de programmation — Ruby



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2012

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
Web www.iso.org

Published in Switzerland

Contents	Page	
1	Scope	1
2	Normative references	1
3	Conformance	1
4	Terms and definitions	2
5	Notational conventions	4
5.1	General description	4
5.2	Syntax	4
5.2.1	General description	4
5.2.2	Productions	5
5.2.3	Syntactic term sequences	6
5.2.4	Syntactic terms	7
5.2.5	Conceptual names	10
5.3	Semantics	10
5.4	Attributes of execution contexts	11
6	Fundamental concepts	12
6.1	Objects	12
6.2	Variables	12
6.2.1	General description	12
6.2.2	Instance variables	13
6.3	Methods	13
6.4	Blocks	14
6.5	Classes, singleton classes, and modules	14
6.5.1	General description	14
6.5.2	Classes	14
6.5.3	Singleton classes	15
6.5.4	Inheritance	16
6.5.5	Modules	17
6.6	Boolean values	18
7	Execution contexts	18
7.1	General description	18
7.2	The initial state	19
8	Lexical structure	19
8.1	General description	19
8.2	Program text	20
8.3	Line terminators	20
8.4	Whitespace	21
8.5	Comments	22
8.6	End-of-program markers	23
8.7	Tokens	23
8.7.1	General description	23
8.7.2	Keywords	23
8.7.3	Identifiers	24
8.7.4	Punctuators	25
8.7.5	Operators	25

8.7.6	Literals	26
8.7.6.1	General description	26
8.7.6.2	Numeric literals	26
8.7.6.3	String literals	29
8.7.6.3.1	General description	29
8.7.6.3.2	Single quoted strings	29
8.7.6.3.3	Double quoted strings	30
8.7.6.3.4	Quoted non-expanded literal strings	33
8.7.6.3.5	Quoted expanded literal strings	35
8.7.6.3.6	Here documents	36
8.7.6.3.7	External command execution	38
8.7.6.4	Array literals	39
8.7.6.5	Regular expression literals	42
8.7.6.6	Symbol literals	43
9	Scope of variables	44
9.1	General description	44
9.2	Scope of local variables	44
9.3	Scope of global variables	45
10	Program structure	45
10.1	Program	45
10.2	Compound statement	46
11	Expressions	47
11.1	General description	47
11.2	Logical expressions	47
11.2.1	General description	47
11.2.2	Logical NOT expressions	48
11.2.3	Logical AND expressions	49
11.2.4	Logical OR expressions	49
11.3	Method invocation expressions	50
11.3.1	General description	50
11.3.2	Method arguments	55
11.3.3	Blocks	58
11.3.4	The <code>super</code> expression	61
11.3.5	The <code>yield</code> expression	64
11.4	Operator expressions	65
11.4.1	General description	65
11.4.2	Assignments	66
11.4.2.1	General description	66
11.4.2.2	Single assignments	66
11.4.2.2.1	General description	66
11.4.2.2.2	Single variable assignments	67
11.4.2.2.3	Scoped constant assignments	69
11.4.2.2.4	Single indexing assignments	69
11.4.2.2.5	Single method assignments	70
11.4.2.3	Abbreviated assignments	71
11.4.2.3.1	General description	71
11.4.2.3.2	Abbreviated variable assignments	71
11.4.2.3.3	Abbreviated indexing assignments	72
11.4.2.3.4	Abbreviated method assignments	73
11.4.2.4	Multiple assignments	74

11.4.2.5	Assignments with rescue modifiers	78
11.4.3	Unary operator expressions	78
11.4.3.1	General description	78
11.4.3.2	The defined? expression	79
11.4.4	Binary operator expressions	81
11.5	Primary expressions	84
11.5.1	General description	84
11.5.2	Control structures	85
11.5.2.1	General description	85
11.5.2.2	Conditional expressions	85
11.5.2.2.1	General description	85
11.5.2.2.2	The if expression	85
11.5.2.2.3	The unless expression	87
11.5.2.2.4	The case expression	87
11.5.2.2.5	Conditional operator expression	88
11.5.2.3	Iteration expressions	89
11.5.2.3.1	General description	89
11.5.2.3.2	The while expression	90
11.5.2.3.3	The until expression	91
11.5.2.3.4	The for expression	91
11.5.2.4	Jump expressions	92
11.5.2.4.1	General description	92
11.5.2.4.2	The return expression	93
11.5.2.4.3	The break expression	94
11.5.2.4.4	The next expression	95
11.5.2.4.5	The redo expression	96
11.5.2.4.6	The retry expression	97
11.5.2.5	The begin expression	97
11.5.3	Grouping expression	99
11.5.4	Variable references	99
11.5.4.1	General description	99
11.5.4.2	Constants	100
11.5.4.3	Scoped constants	101
11.5.4.4	Global variables	101
11.5.4.5	Class variables	102
11.5.4.6	Instance variables	102
11.5.4.7	Local variables or method invocations	102
11.5.4.7.1	General description	102
11.5.4.7.2	Determination of the type of local variable identifiers	103
11.5.4.7.3	Local variables	103
11.5.4.7.4	Method invocations	104
11.5.4.8	Pseudo variables	104
11.5.4.8.1	General description	104
11.5.4.8.2	The nil expression	104
11.5.4.8.3	The true expression and the false expression	104
11.5.4.8.4	The self expression	105
11.5.5	Object constructors	105
11.5.5.1	Array constructor	105
11.5.5.2	Hash constructor	105
11.5.5.3	Range expression	106
12	Statements	107
12.1	General description	107

12.2	Expression statement	107
12.3	The <code>if</code> modifier statement	108
12.4	The <code>unless</code> modifier statement	108
12.5	The <code>while</code> modifier statement	108
12.6	The <code>until</code> modifier statement	109
12.7	The <code>rescue</code> modifier statement	109
13	Classes and modules	110
13.1	Modules	110
13.1.1	General description	110
13.1.2	Module definition	111
13.1.3	Module inclusion	112
13.2	Classes	112
13.2.1	General description	112
13.2.2	Class definition	112
13.2.3	Inheritance	114
13.2.4	Instance creation	114
13.3	Methods	115
13.3.1	Method definition	115
13.3.2	Method parameters	116
13.3.3	Method invocation	118
13.3.4	Method lookup	120
13.3.5	Method visibility	121
13.3.5.1	General description	121
13.3.5.2	Public methods	121
13.3.5.3	Private methods	121
13.3.5.4	Protected methods	121
13.3.5.5	Visibility change	122
13.3.6	The <code>alias</code> statement	122
13.3.7	The <code>undef</code> statement	123
13.4	Singleton classes	124
13.4.1	General description	124
13.4.2	Singleton class definition	125
13.4.3	Singleton method definition	126
14	Exceptions	127
14.1	General description	127
14.2	Cause of exceptions	127
14.3	Exception handling	127
15	Built-in classes and modules	128
15.1	General description	128
15.2	Built-in classes	131
15.2.1	Object	131
15.2.1.1	General description	131
15.2.1.2	Direct superclass	131
15.2.1.3	Included modules	131
15.2.1.4	Constants	131
15.2.1.5	Instance methods	132
15.2.1.5.1	Object#initialize	132
15.2.2	Module	132
15.2.2.1	General description	132
15.2.2.2	Direct superclass	132

15.2.2.3	Singleton methods	132
15.2.2.3.1	Module.constants	132
15.2.2.3.2	Module.nesting	133
15.2.2.4	Instance methods	133
15.2.2.4.1	Module#<=>	133
15.2.2.4.2	Module#<	134
15.2.2.4.3	Module#<=	134
15.2.2.4.4	Module#>	134
15.2.2.4.5	Module#>=	135
15.2.2.4.6	Module#==	135
15.2.2.4.7	Module#===	135
15.2.2.4.8	Module#alias_method	135
15.2.2.4.9	Module#ancestors	136
15.2.2.4.10	Module#append_features	136
15.2.2.4.11	Module#attr	136
15.2.2.4.12	Module#attr_accessor	137
15.2.2.4.13	Module#attr_reader	137
15.2.2.4.14	Module#attr_writer	138
15.2.2.4.15	Module#class_eval	138
15.2.2.4.16	Module#class_variable_defined?	139
15.2.2.4.17	Module#class_variable_get	140
15.2.2.4.18	Module#class_variable_set	140
15.2.2.4.19	Module#class_variables	140
15.2.2.4.20	Module#const_defined?	141
15.2.2.4.21	Module#const_get	142
15.2.2.4.22	Module#const_missing	142
15.2.2.4.23	Module#const_set	143
15.2.2.4.24	Module#constants	143
15.2.2.4.25	Module#extend_object	144
15.2.2.4.26	Module#extended	144
15.2.2.4.27	Module#include	144
15.2.2.4.28	Module#include?	145
15.2.2.4.29	Module#included	145
15.2.2.4.30	Module#included_modules	145
15.2.2.4.31	Module#initialize	145
15.2.2.4.32	Module#initialize_copy	146
15.2.2.4.33	Module#instance_methods	147
15.2.2.4.34	Module#method_defined?	147
15.2.2.4.35	Module#module_eval	148
15.2.2.4.36	Module#private	148
15.2.2.4.37	Module#protected	148
15.2.2.4.38	Module#public	148
15.2.2.4.39	Module#remove_class_variable	149
15.2.2.4.40	Module#remove_const	150
15.2.2.4.41	Module#remove_method	150
15.2.2.4.42	Module#undef_method	151
15.2.3	Class	151
15.2.3.1	General description	151
15.2.3.2	Direct superclass	151
15.2.3.3	Instance methods	151
15.2.3.3.1	Class#initialize	151
15.2.3.3.2	Class#initialize_copy	152

15.2.3.3.3	Class#new	152
15.2.3.3.4	Class#superclass	153
15.2.4	NilClass	153
15.2.4.1	General description	153
15.2.4.2	Direct superclass	153
15.2.4.3	Instance methods	153
15.2.4.3.1	NilClass#&	153
15.2.4.3.2	NilClass#	154
15.2.4.3.3	NilClass#^	154
15.2.4.3.4	NilClass#nil?	154
15.2.4.3.5	NilClass#to_s	154
15.2.5	TrueClass	154
15.2.5.1	General description	154
15.2.5.2	Direct superclass	155
15.2.5.3	Instance methods	155
15.2.5.3.1	TrueClass#&	155
15.2.5.3.2	TrueClass#	155
15.2.5.3.3	TrueClass#^	155
15.2.5.3.4	TrueClass#to_s	155
15.2.6	FalseClass	156
15.2.6.1	General description	156
15.2.6.2	Direct superclass	156
15.2.6.3	Instance methods	156
15.2.6.3.1	FalseClass#&	156
15.2.6.3.2	FalseClass#	156
15.2.6.3.3	FalseClass#^	156
15.2.6.3.4	FalseClass#to_s	157
15.2.7	Numeric	157
15.2.7.1	General description	157
15.2.7.2	Direct superclass	157
15.2.7.3	Included modules	157
15.2.7.4	Instance methods	157
15.2.7.4.1	Numeric#+@	157
15.2.7.4.2	Numeric#-@	158
15.2.7.4.3	Numeric#abs	158
15.2.7.4.4	Numeric#coerce	158
15.2.8	Integer	159
15.2.8.1	General description	159
15.2.8.2	Direct superclass	160
15.2.8.3	Instance methods	160
15.2.8.3.1	Integer#<=>	160
15.2.8.3.2	Integer#==	160
15.2.8.3.3	Integer#+	161
15.2.8.3.4	Integer#-	161
15.2.8.3.5	Integer#*	162
15.2.8.3.6	Integer#/.	162
15.2.8.3.7	Integer#%	163
15.2.8.3.8	Integer#~	164
15.2.8.3.9	Integer#&	164
15.2.8.3.10	Integer#	164
15.2.8.3.11	Integer#^	165
15.2.8.3.12	Integer#<<	165

15.2.8.3.13	Integer#>>	165
15.2.8.3.14	Integer#ceil	165
15.2.8.3.15	Integer#downto	166
15.2.8.3.16	Integer#eql?	166
15.2.8.3.17	Integer#floor	166
15.2.8.3.18	Integer#hash	167
15.2.8.3.19	Integer#next	167
15.2.8.3.20	Integer#round	167
15.2.8.3.21	Integer#succ	167
15.2.8.3.22	Integer#times	167
15.2.8.3.23	Integer#to_f	168
15.2.8.3.24	Integer#to_i	168
15.2.8.3.25	Integer#to_s	168
15.2.8.3.26	Integer#truncate	169
15.2.8.3.27	Integer#upto	169
15.2.9	Float	169
15.2.9.1	General description	169
15.2.9.2	Direct superclass	170
15.2.9.3	Instance methods	170
15.2.9.3.1	Float#<=>	170
15.2.9.3.2	Float#==	170
15.2.9.3.3	Float#+	171
15.2.9.3.4	Float#-	172
15.2.9.3.5	Float#*	172
15.2.9.3.6	Float#/	173
15.2.9.3.7	Float#%	173
15.2.9.3.8	Float#ceil	174
15.2.9.3.9	Float#finite?	174
15.2.9.3.10	Float#floor	175
15.2.9.3.11	Float#infinite?	175
15.2.9.3.12	Float#round	175
15.2.9.3.13	Float#to_f	175
15.2.9.3.14	Float#to_i	175
15.2.9.3.15	Float#truncate	176
15.2.10	String	176
15.2.10.1	General description	176
15.2.10.2	Direct superclass	176
15.2.10.3	Included modules	176
15.2.10.4	Upper-case and lower-case characters	177
15.2.10.5	Instance methods	177
15.2.10.5.1	String#<=>	177
15.2.10.5.2	String#==	178
15.2.10.5.3	String#=~	178
15.2.10.5.4	String#+	179
15.2.10.5.5	String#*	179
15.2.10.5.6	String#[]	179
15.2.10.5.7	String#capitalize	181
15.2.10.5.8	String#capitalize!	181
15.2.10.5.9	String#chomp	181
15.2.10.5.10	String#chomp!	182
15.2.10.5.11	String#chop	182
15.2.10.5.12	String#chop!	182

15.2.10.5.13	String#downcase	183
15.2.10.5.14	String#downcase!	183
15.2.10.5.15	String#each_line	183
15.2.10.5.16	String#empty?	184
15.2.10.5.17	String#eql?	184
15.2.10.5.18	String#gsub	184
15.2.10.5.19	String#gsub!	186
15.2.10.5.20	String#hash	186
15.2.10.5.21	String#include?	186
15.2.10.5.22	String#index	187
15.2.10.5.23	String#initialize	187
15.2.10.5.24	String#initialize_copy	188
15.2.10.5.25	String#intern	188
15.2.10.5.26	String#length	188
15.2.10.5.27	String#match	188
15.2.10.5.28	String#replace	189
15.2.10.5.29	String#reverse	189
15.2.10.5.30	String#reverse!	189
15.2.10.5.31	String#rindex	189
15.2.10.5.32	String#scan	190
15.2.10.5.33	String#size	191
15.2.10.5.34	String#slice	191
15.2.10.5.35	String#split	191
15.2.10.5.36	String#sub	193
15.2.10.5.37	String#sub!	193
15.2.10.5.38	String#to_f	194
15.2.10.5.39	String#to_i	194
15.2.10.5.40	String#to_s	195
15.2.10.5.41	String#to_sym	195
15.2.10.5.42	String#upcase	195
15.2.10.5.43	String#upcase!	196
15.2.11	Symbol	196
15.2.11.1	General description	196
15.2.11.2	Direct superclass	196
15.2.11.3	Instance methods	196
15.2.11.3.1	Symbol#===	196
15.2.11.3.2	Symbol#id2name	197
15.2.11.3.3	Symbol#to_s	197
15.2.11.3.4	Symbol#to_sym	197
15.2.12	Array	197
15.2.12.1	General description	197
15.2.12.2	Direct superclass	198
15.2.12.3	Included modules	198
15.2.12.4	Singleton methods	198
15.2.12.4.1	Array.[]	198
15.2.12.5	Instance methods	198
15.2.12.5.1	Array#+	198
15.2.12.5.2	Array#*	199
15.2.12.5.3	Array#<<	199
15.2.12.5.4	Array#[]	199
15.2.12.5.5	Array#[]=	200
15.2.12.5.6	Array#clear	201

15.2.12.5.7	Array#collect!	201
15.2.12.5.8	Array#concat	201
15.2.12.5.9	Array#delete_at	202
15.2.12.5.10	Array#each	202
15.2.12.5.11	Array#each_index	202
15.2.12.5.12	Array#empty?	203
15.2.12.5.13	Array#first	203
15.2.12.5.14	Array#index	204
15.2.12.5.15	Array#initialize	204
15.2.12.5.16	Array#initialize_copy	205
15.2.12.5.17	Array#join	205
15.2.12.5.18	Array#last	206
15.2.12.5.19	Array#length	206
15.2.12.5.20	Array#map!	207
15.2.12.5.21	Array#pop	207
15.2.12.5.22	Array#push	207
15.2.12.5.23	Array#replace	207
15.2.12.5.24	Array#reverse	207
15.2.12.5.25	Array#reverse!	208
15.2.12.5.26	Array#rindex	208
15.2.12.5.27	Array#shift	208
15.2.12.5.28	Array#size	209
15.2.12.5.29	Array#slice	209
15.2.12.5.30	Array#unshift	209
15.2.13	Hash	209
15.2.13.1	General description	209
15.2.13.2	Direct superclass	210
15.2.13.3	Included modules	210
15.2.13.4	Instance methods	210
15.2.13.4.1	Hash#==	210
15.2.13.4.2	Hash#[]	211
15.2.13.4.3	Hash#[]=	211
15.2.13.4.4	Hash#clear	212
15.2.13.4.5	Hash#default	212
15.2.13.4.6	Hash#default=	212
15.2.13.4.7	Hash#default_proc	213
15.2.13.4.8	Hash#delete	213
15.2.13.4.9	Hash#each	213
15.2.13.4.10	Hash#each_key	214
15.2.13.4.11	Hash#each_value	214
15.2.13.4.12	Hash#empty?	214
15.2.13.4.13	Hash#has_key?	215
15.2.13.4.14	Hash#has_value?	215
15.2.13.4.15	Hash#include?	215
15.2.13.4.16	Hash#initialize	215
15.2.13.4.17	Hash#initialize_copy	216
15.2.13.4.18	Hash#key?	216
15.2.13.4.19	Hash#keys	216
15.2.13.4.20	Hash#length	217
15.2.13.4.21	Hash#member?	217
15.2.13.4.22	Hash#merge	217
15.2.13.4.23	Hash#replace	218

15.2.13.4.24	Hash#shift	218
15.2.13.4.25	Hash#size	218
15.2.13.4.26	Hash#store	219
15.2.13.4.27	Hash#value?	219
15.2.13.4.28	Hash#values	219
15.2.14	Range	219
15.2.14.1	General description	219
15.2.14.2	Direct superclass	219
15.2.14.3	Included modules	220
15.2.14.4	Instance methods	220
15.2.14.4.1	Range#==	220
15.2.14.4.2	Range#===	220
15.2.14.4.3	Range#begin	221
15.2.14.4.4	Range#each	221
15.2.14.4.5	Range#end	222
15.2.14.4.6	Range#exclude_end?	222
15.2.14.4.7	Range#first	222
15.2.14.4.8	Range#include?	222
15.2.14.4.9	Range#initialize	222
15.2.14.4.10	Range#last	223
15.2.14.4.11	Range#member?	223
15.2.15	Regexp	223
15.2.15.1	General description	223
15.2.15.2	Direct superclass	224
15.2.15.3	Constants	224
15.2.15.4	Patterns	224
15.2.15.5	Matching process	228
15.2.15.6	Singleton methods	229
15.2.15.6.1	Regexp.compile	229
15.2.15.6.2	Regexp.escape	229
15.2.15.6.3	Regexp.last_match	230
15.2.15.6.4	Regexp.quote	231
15.2.15.7	Instance methods	231
15.2.15.7.1	Regexp#==	231
15.2.15.7.2	Regexp#===	231
15.2.15.7.3	Regexp#=~	232
15.2.15.7.4	Regexp#casefold?	232
15.2.15.7.5	Regexp#initialize	233
15.2.15.7.6	Regexp#initialize_copy	233
15.2.15.7.7	Regexp#match	234
15.2.15.7.8	Regexp#source	234
15.2.16	MatchData	234
15.2.16.1	General description	234
15.2.16.2	Direct superclass	235
15.2.16.3	Instance methods	235
15.2.16.3.1	MatchData#[]	235
15.2.16.3.2	MatchData#begin	235
15.2.16.3.3	MatchData#captures	235
15.2.16.3.4	MatchData#end	236
15.2.16.3.5	MatchData#initialize_copy	236
15.2.16.3.6	MatchData#length	237
15.2.16.3.7	MatchData#offset	237

15.2.16.3.8	MatchData#post_match	237
15.2.16.3.9	MatchData#pre_match	237
15.2.16.3.10	MatchData#size	238
15.2.16.3.11	MatchData#string	238
15.2.16.3.12	MatchData#to_a	238
15.2.16.3.13	MatchData#to_s	238
15.2.17	Proc	239
15.2.17.1	General description	239
15.2.17.2	Direct superclass	239
15.2.17.3	Singleton methods	239
15.2.17.3.1	Proc.new	239
15.2.17.4	Instance methods	239
15.2.17.4.1	Proc#[]	239
15.2.17.4.2	Proc#arity	239
15.2.17.4.3	Proc#call	240
15.2.17.4.4	Proc#clone	241
15.2.17.4.5	Proc#dup	241
15.2.18	Struct	242
15.2.18.1	General description	242
15.2.18.2	Direct superclass	242
15.2.18.3	Singleton methods	242
15.2.18.3.1	Struct.new	242
15.2.18.4	Instance methods	244
15.2.18.4.1	Struct#==	244
15.2.18.4.2	Struct#[]	244
15.2.18.4.3	Struct#[]=	245
15.2.18.4.4	Struct#each	246
15.2.18.4.5	Struct#each_pair	246
15.2.18.4.6	Struct#initialize	246
15.2.18.4.7	Struct#initialize_copy	247
15.2.18.4.8	Struct#members	247
15.2.18.4.9	Struct#select	247
15.2.19	Time	248
15.2.19.1	General description	248
15.2.19.2	Direct superclass	248
15.2.19.3	Time computation	248
15.2.19.3.1	Day	248
15.2.19.3.2	Year	249
15.2.19.3.3	Month	249
15.2.19.3.4	Days of month	250
15.2.19.3.5	Hours, Minutes, and Seconds	250
15.2.19.4	Time zone and Local time	251
15.2.19.5	Daylight saving time	251
15.2.19.6	Singleton methods	251
15.2.19.6.1	Time.at	251
15.2.19.6.2	Time.gm	252
15.2.19.6.3	Time.local	254
15.2.19.6.4	Time.mktime	254
15.2.19.6.5	Time.now	254
15.2.19.6.6	Time.utc	254
15.2.19.7	Instance methods	255
15.2.19.7.1	Time#<=>	255

15.2.19.7.2	Time#+	255
15.2.19.7.3	Time#-	256
15.2.19.7.4	Time#asctime	256
15.2.19.7.5	Time#ctime	257
15.2.19.7.6	Time#day	257
15.2.19.7.7	Time#dst?	257
15.2.19.7.8	Time#getgm	258
15.2.19.7.9	Time#getlocal	258
15.2.19.7.10	Time#getutc	258
15.2.19.7.11	Time#gmt?	258
15.2.19.7.12	Time#gmt_offset	258
15.2.19.7.13	Time#gmtime	259
15.2.19.7.14	Time#gmtoff	259
15.2.19.7.15	Time#hour	259
15.2.19.7.16	Time#initialize	259
15.2.19.7.17	Time#initialize_copy	260
15.2.19.7.18	Time#localtime	260
15.2.19.7.19	Time#mday	260
15.2.19.7.20	Time#min	261
15.2.19.7.21	Time#mon	261
15.2.19.7.22	Time#month	261
15.2.19.7.23	Time#sec	261
15.2.19.7.24	Time#to_f	262
15.2.19.7.25	Time#to_i	262
15.2.19.7.26	Time#usec	262
15.2.19.7.27	Time#utc	262
15.2.19.7.28	Time#utc?	263
15.2.19.7.29	Time#utc_offset	263
15.2.19.7.30	Time#wday	263
15.2.19.7.31	Time#yday	263
15.2.19.7.32	Time#year	264
15.2.19.7.33	Time#zone	264
15.2.20	IO	264
15.2.20.1	General description	264
15.2.20.2	Direct superclass	265
15.2.20.3	Included modules	265
15.2.20.4	Singleton methods	265
15.2.20.4.1	IO.open	265
15.2.20.5	Instance methods	266
15.2.20.5.1	IO#close	266
15.2.20.5.2	IO#closed?	266
15.2.20.5.3	IO#each	267
15.2.20.5.4	IO#each_byte	267
15.2.20.5.5	IO#each_line	268
15.2.20.5.6	IO#eof?	268
15.2.20.5.7	IO#flush	268
15.2.20.5.8	IO#getc	268
15.2.20.5.9	IO#gets	269
15.2.20.5.10	IO#initialize_copy	269
15.2.20.5.11	IO#print	269
15.2.20.5.12	IO#putc	270
15.2.20.5.13	IO#puts	270

15.2.20.5.14	IO#read	271
15.2.20.5.15	IO#readchar	271
15.2.20.5.16	IO#readline	272
15.2.20.5.17	IO#readlines	272
15.2.20.5.18	IO#sync	273
15.2.20.5.19	IO#sync=	273
15.2.20.5.20	IO#write	273
15.2.21	File	274
15.2.21.1	General description	274
15.2.21.2	Direct superclass	274
15.2.21.3	Singleton methods	274
15.2.21.3.1	File.exist?	274
15.2.21.4	Instance methods	274
15.2.21.4.1	File#initialize	274
15.2.21.4.2	File#path	275
15.2.22	Exception	275
15.2.22.1	General description	275
15.2.22.2	Direct superclass	275
15.2.22.3	Singleton methods	275
15.2.22.3.1	Exception.exception	275
15.2.22.4	Instance methods	276
15.2.22.4.1	Exception#exception	276
15.2.22.4.2	Exception#initialize	276
15.2.22.4.3	Exception#message	276
15.2.22.4.4	Exception#to_s	277
15.2.23	StandardError	277
15.2.23.1	General description	277
15.2.23.2	Direct superclass	277
15.2.24	ArgumentError	277
15.2.24.1	General description	277
15.2.24.2	Direct superclass	277
15.2.25	LocalJumpError	277
15.2.25.1	Direct superclass	278
15.2.25.2	Instance methods	278
15.2.25.2.1	LocalJumpError#exit_value	278
15.2.25.2.2	LocalJumpError#reason	278
15.2.26	RangeError	278
15.2.26.1	General description	278
15.2.26.2	Direct superclass	278
15.2.27	RegexpError	278
15.2.27.1	General description	278
15.2.27.2	Direct superclass	278
15.2.28	RuntimeError	278
15.2.28.1	General description	278
15.2.28.2	Direct superclass	279
15.2.29	TypeError	279
15.2.29.1	General description	279
15.2.29.2	Direct superclass	279
15.2.30	ZeroDivisionError	279
15.2.30.1	General description	279
15.2.30.2	Direct superclass	279
15.2.31	NameError	279

15.2.31.1	Direct superclass	279
15.2.31.2	Instance methods	279
15.2.31.2.1	NameError#initialize	279
15.2.31.2.2	NameError#name	280
15.2.32	NoMethodError	280
15.2.32.1	Direct superclass	280
15.2.32.2	Instance methods	280
15.2.32.2.1	NoMethodError#args	280
15.2.32.2.2	NoMethodError#initialize	280
15.2.33	IndexError	281
15.2.33.1	General description	281
15.2.33.2	Direct superclass	281
15.2.34	IOError	281
15.2.34.1	General description	281
15.2.34.2	Direct superclass	281
15.2.35	EOFError	281
15.2.35.1	General description	281
15.2.35.2	Direct superclass	281
15.2.36	SystemCallError	281
15.2.36.1	General description	281
15.2.36.2	Direct superclass	281
15.2.37	ScriptError	281
15.2.37.1	General description	281
15.2.37.2	Direct superclass	282
15.2.38	SyntaxError	282
15.2.38.1	General description	282
15.2.38.2	Direct superclass	282
15.2.39	LoadError	282
15.2.39.1	General description	282
15.2.39.2	Direct superclass	282
15.3	Built-in modules	282
15.3.1	Kernel	282
15.3.1.1	General description	282
15.3.1.2	Singleton methods	282
15.3.1.2.1	Kernel.`	282
15.3.1.2.2	Kernel.block_given?	283
15.3.1.2.3	Kernel.eval	283
15.3.1.2.4	Kernel.global_variables	283
15.3.1.2.5	Kernel.iterator?	284
15.3.1.2.6	Kernel.lambda	284
15.3.1.2.7	Kernel.local_variables	285
15.3.1.2.8	Kernel.loop	285
15.3.1.2.9	Kernel.p	285
15.3.1.2.10	Kernel.print	286
15.3.1.2.11	Kernel.puts	286
15.3.1.2.12	Kernel.raise	286
15.3.1.2.13	Kernel.require	287
15.3.1.3	Instance methods	288
15.3.1.3.1	Kernel#==	288
15.3.1.3.2	Kernel#===	288
15.3.1.3.3	Kernel#`	289
15.3.1.3.4	Kernel#_id_	289

15.3.1.3.5	Kernel#_send_	289
15.3.1.3.6	Kernel#block_given?	289
15.3.1.3.7	Kernel#class	289
15.3.1.3.8	Kernel#clone	290
15.3.1.3.9	Kernel#dup	290
15.3.1.3.10	Kernel#eql?	291
15.3.1.3.11	Kernel#equal?	291
15.3.1.3.12	Kernel#eval	291
15.3.1.3.13	Kernel#extend	292
15.3.1.3.14	Kernel#global_variables	292
15.3.1.3.15	Kernel#hash	292
15.3.1.3.16	Kernel#initialize_copy	293
15.3.1.3.17	Kernel#inspect	293
15.3.1.3.18	Kernel#instance_eval	293
15.3.1.3.19	Kernel#instance_of?	294
15.3.1.3.20	Kernel#instance_variable_defined?	294
15.3.1.3.21	Kernel#instance_variable_get	294
15.3.1.3.22	Kernel#instance_variable_set	295
15.3.1.3.23	Kernel#instance_variables	295
15.3.1.3.24	Kernel#is_a?	295
15.3.1.3.25	Kernel#iterator?	296
15.3.1.3.26	Kernel#kind_of?	296
15.3.1.3.27	Kernel#lambda	296
15.3.1.3.28	Kernel#local_variables	296
15.3.1.3.29	Kernel#loop	297
15.3.1.3.30	Kernel#method_missing	297
15.3.1.3.31	Kernel#methods	297
15.3.1.3.32	Kernel#nil?	297
15.3.1.3.33	Kernel#object_id	298
15.3.1.3.34	Kernel#p	298
15.3.1.3.35	Kernel#print	298
15.3.1.3.36	Kernel#private_methods	298
15.3.1.3.37	Kernel#protected_methods	299
15.3.1.3.38	Kernel#public_methods	299
15.3.1.3.39	Kernel#puts	300
15.3.1.3.40	Kernel#raise	300
15.3.1.3.41	Kernel#remove_instance_variable	300
15.3.1.3.42	Kernel#require	301
15.3.1.3.43	Kernel#respond_to?	301
15.3.1.3.44	Kernel#send	301
15.3.1.3.45	Kernel#singleton_methods	302
15.3.1.3.46	Kernel#to_s	302
15.3.2	Enumerable	302
15.3.2.1	General description	302
15.3.2.2	Instance methods	303
15.3.2.2.1	Enumerable#all?	303
15.3.2.2.2	Enumerable#any?	303
15.3.2.2.3	Enumerable#collect	303
15.3.2.2.4	Enumerable#detect	304
15.3.2.2.5	Enumerable#each_with_index	304
15.3.2.2.6	Enumerable#entries	305
15.3.2.2.7	Enumerable#find	305

15.3.2.2.8	Enumerable#find_all	305
15.3.2.2.9	Enumerable#grep	305
15.3.2.2.10	Enumerable#include?	306
15.3.2.2.11	Enumerable#inject	306
15.3.2.2.12	Enumerable#map	307
15.3.2.2.13	Enumerable#max	307
15.3.2.2.14	Enumerable#member?	308
15.3.2.2.15	Enumerable#min	308
15.3.2.2.16	Enumerable#partition	309
15.3.2.2.17	Enumerable#reject	309
15.3.2.2.18	Enumerable#select	310
15.3.2.2.19	Enumerable#sort	310
15.3.2.2.20	Enumerable#to_a	311
15.3.3	Comparable	311
15.3.3.1	General description	311
15.3.3.2	Instance methods	311
15.3.3.2.1	Comparable#<	311
15.3.3.2.2	Comparable#<=	311
15.3.3.2.3	Comparable#>	312
15.3.3.2.4	Comparable#>=	312
15.3.3.2.5	Comparable#==	312
15.3.3.2.6	Comparable#between?	313

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 30170 was prepared by the Japanese Industrial Standards Committee (as JIS X3017) 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 the national bodies of ISO and IEC.

Introduction

This International Standard is based upon a submission from the Japanese National Standards Body called JIS X3017 “Programming Language Ruby”, registered and published in 2011.

Ruby is an object-oriented scripting language designed to be developer-friendly, productive and intuitive. There is a continuing growth of interest in Ruby around the world, especially among web application developers, while its use spans from web applications to private tools.

As the Ruby language grows and spreads, there is no globally agreed upon documented Ruby specification. In order to avoid confusion as a result of diversification of usage and incompatibility among implementations, the Japan Industry Standard is proposed as an international standard.

There are multiple Ruby implementations available. Many of them are distributed as open source software. The implementation called “Matz Ruby Implementation (MRI)” has been treated as a reference implementation insofar as virtually all implementers check compatibility of their implementations by comparing them to MRI. Therefore, this specification of Ruby is codified as a strict subset of MRI.

This International Standard specifies only core language features and core libraries which are stable enough and common between MRI versions and compatible between existing implementations. There are two versions of MRI currently distributed and maintained: MRI 1.8, which has been available since 2003 and MRI 1.9, which was released in 2010. Currently, MRI 1.8 is more widely used than MRI 1.9. Use of MRI 1.9 will likely spread in the next several years. To avoid future divergence, features which are planned or prospected to be changed are excluded from this version of the specification, or it is clearly stated that the behavior of the features are not specified. For example, this specification does not specify the handling of character type in detail because it is planned to be changed in MRI 1.9 for full support of ISO/IEC 10646. The full support of ISO/IEC 10646 is going to be standardized in a future version of this standard. The library defined in this specification is limited to that which is commonly used or necessary to write simple programs.

This International Standard introduces special notations and a concept called “Execution context” in order to specify flexible syntax and dynamic semantics of the Ruby language as simple as possible.

Information technology — Programming languages — Ruby

1 Scope

This International Standard specifies the syntax and semantics of the computer programming language Ruby, and the requirements for conforming Ruby processors, strictly conforming Ruby programs, and conforming Ruby programs.

This International Standard does not specify,

- the limit of size or complexity of a program text which a conforming processor evaluates,
- the minimal requirements of a data processing system that is capable of supporting a conforming processor,
- the method for activating the execution of programs on a data processing system, and
- the method for reporting syntactic and runtime errors.

NOTE Execution of a Ruby program is to evaluate the *program* (see 10) by a Ruby processor.

2 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 646:1991, *Information technology – ISO 7-bit coded character set for information interchange*.
- IEC 60559:1989, *Binary floating-point arithmetic for microprocessor systems*.
- ISO/IEC 2382-1:1993, *Information technology – Vocabulary – Part 1: Fundamental terms*.