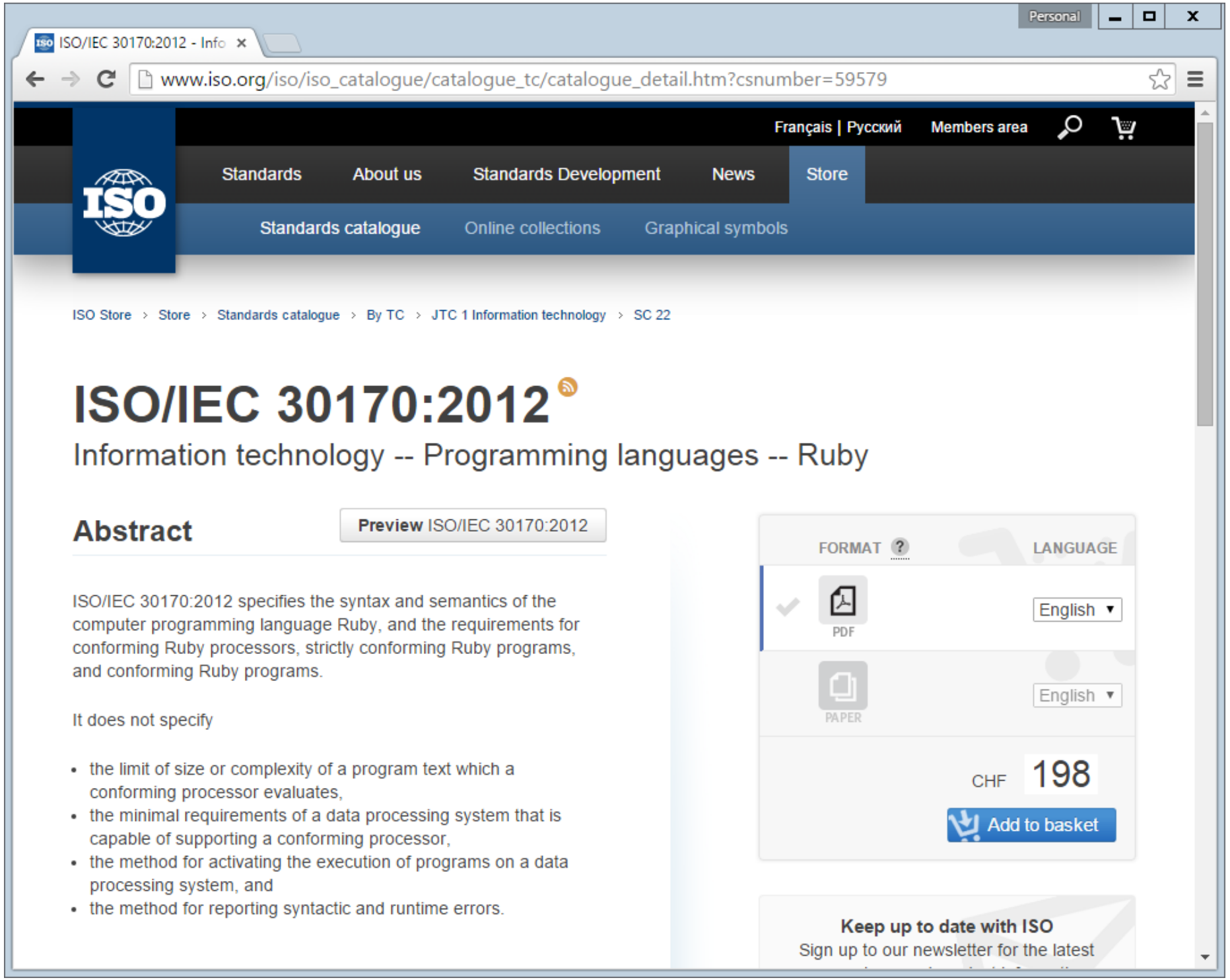
Ruby ISO Standard

David Grayson
2015-07-22


Personal
ISO/IEC 30170:2012 - Info
www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=59579
Français | Русский
Members area
Standards
About us
Standards Development
News
Store
Standards catalogue
Online collections
Graphical symbols
ISO Store › Store › Standards catalogue › By TC › JTC 1 Information technology › SC 22
ISO/IEC 30170:2012
Information technology -- Programming languages -- Ruby
Abstract
Preview ISO/IEC 30170:2012
ISO/IEC 30170:2012 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.
It 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.
FORMAT
LANGUAGE
PDF
English
PAPER
English
CHF 198
Add to basket
Keep up to date with ISO
Sign up to our newsletter for the latest

Basic info

- Based on Ruby 1.8.7
- The final draft is from August 25, 2010
- 331 pages
- Does not cover the whole standard library
- You can download it for free if you look in the right place: https://www.ipa.go.jp/files/000011432.pdf

Who?

- Ruby Standardization Working Group
- Chairman: Ikuo Nakata
- Shugo Maeda

Why?

- Supposedly for government procurement in Japan
- Another way to know what your code means
- Tell Ruby users what they can count on
- Tell Ruby implementers what they can get do

Overview

1) Scope
2) Normative references
3) Conformance
4) Terms and definitions
5) Notational conventions
6) Fundamental concepts
7) Execution contexts
8) Lexical structure
9) Scope of variables
10) Program structure
11) Expressions
12) Statements
13) Classes and modules
14) Exceptions
15) **Built-in classes & modules**

Built-in classes & modules

- Object
- Module
- Class
- NilClass
- TrueClass
- FalseClass
- Numeric
- Integer
- Float
- String
- Symbol
- Array
- Hash
- Range
- Regexp
- MatchData
- Proc
- Struct
- Time
- IO
- File
- Exception…
- Kernel
- Enumerable
- Comparable

Return value of #puts

- See section 15.2.20.5.13
- Return value is *implementation defined*

```
puts "hello" or puts "hi"
```

Return value of Array#each

- See section 15.2.12.5.10
- If a block is given, returns the receiver

```
a = [1, 2, 3].each { |n| puts n }
```

- If a block is not given, the behavior is *unspecified*

Terms

- *Unspecified:* Possibly differing between implementations, and not necessarily defined for any particular implementation.
- *Implementation defined:* Possibly differing between implementations, but defined for every implementation.

Enumerable#sort

```
a = [x, y, z].sort
```

- See section 15.3.2.2.19
- Overly specific: says to create an empty array.
- Definition depends on what exactly “suppose” means.
- Doesn't actually require that any two elements be compared.
- Disallows comparing the same objects twice!

Better specification of sort

- Define what it means for results from the comparison operation to be well-behaved.
- If the results are not well-behaved, behavior of #sort is unspecified.
- #sort must do enough comparisons so there is no ambiguity in the ordering of objects.
- #sort must return a sorted array.

Ruby ISO vs. RubySpec

- RubySpec is executable.
  - Based on a finite number of test cases
  - Cannot test *implentation defined* and *unspecified*
- RubySpec only really tests the interpreter, doesn't define valid Ruby programs.

RubySpec sample

```
it "dumps a Fixnum" do
  [ [Marshal,  0,        "\004\bi\000"],
    [Marshal,  5,        "\004\bi\n"],
    [Marshal,  8,        "\004\bi\r"],
    [Marshal,  122,      "\004\bi\177"],
    [Marshal,  123,      "\004\bi\001{"],
    [Marshal,  1234,     "\004\bi\002\322\004"],
    [Marshal, -8,        "\004\bi\363"],
    [Marshal, -123,      "\004\bi\200"],
    [Marshal, -124,      "\004\bi\377\204"],
    [Marshal, -1234,     "\004\bi\376.\373"],
    [Marshal, -4516727, "\004\bi\375\211\024\273"],
    [Marshal,  2**8,     "\004\bi\002\000\001"],
    [Marshal,  2**16,    "\004\bi\003\000\000\001"],
    [Marshal,  2**24,    "\004\bi\004\000\000\000\001"],
    [Marshal, -2**8,     "\004\bi\377\000"],
    [Marshal, -2**16,    "\004\bi\376\000\000"],
    [Marshal, -2**24,    "\004\bi\375\000\000\000"],
  ].should be_computed_by(:dump)
end
```

https://github.com/rubyspec/rubyspec/blob/archive/core/marshal/dump_spec.rb

Comparison to C++ standard

- C++ designer Bjarne Stroustrup stuck with the ISO standards efforts for 20+ years.
- Periodic new versions: C++11, C++14, C++17...
- 1354 pages
- Covers the whole standard library
- Concept of undefined behavior
- Standard is regularly mentioned on StackOverflow

References

https://www.ipa.go.jp/files/000011432.pdf

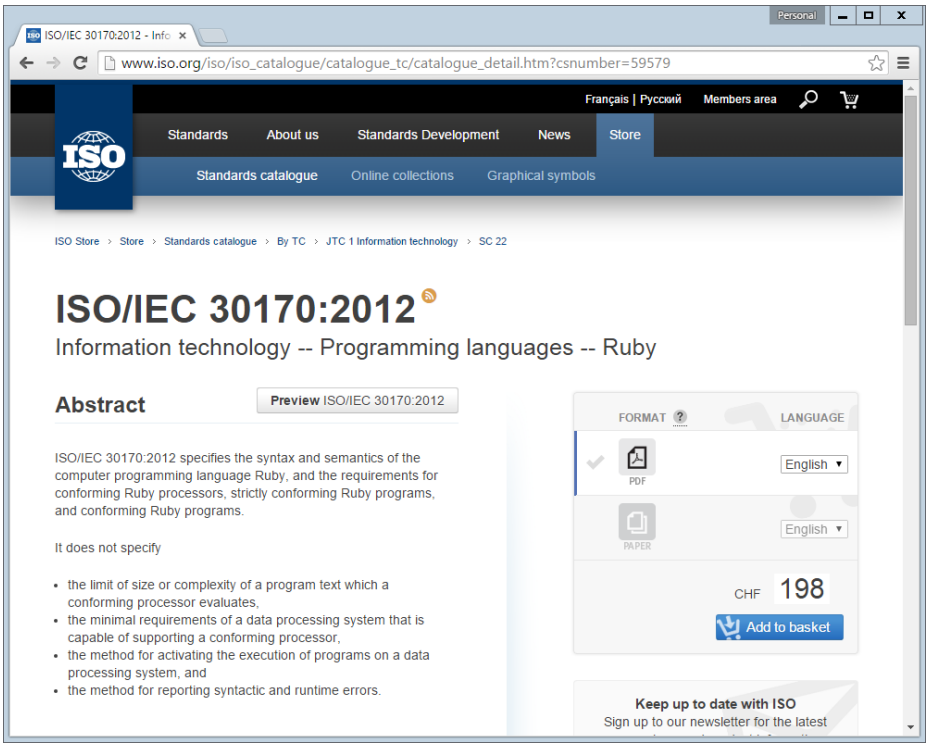
https://www.ipa.go.jp/osc/english/ruby/

http://www.rubyinside.com/ruby-iso-spec-draft-2900.html

http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-core/26969

Ruby ISO Standard

David Grayson
2015-07-22


ISO/IEC 30170:2012 - Info
www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=59579
Français | Русский
Members area
Standards
About us
Standards Development
News
Store
Standards catalogue
Online collections
Graphical symbols
ISO Store › Store › Standards catalogue › By TC › JTC 1 Information technology › SC 22
ISO/IEC 30170:2012
Information technology -- Programming languages -- Ruby
Abstract
Preview ISO/IEC 30170:2012
ISO/IEC 30170:2012 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.
It 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.
FORMAT
LANGUAGE
PDF
English
PAPER
English
CHF 198
Add to basket
Keep up to date with ISO
Sign up to our newsletter for the latest

Basic info

- Based on Ruby 1.8.7
- The final draft is from August 25, 2010
- 331 pages
- Does not cover the whole standard library
- You can download it for free if you look in the right place: https://www.ipa.go.jp/files/000011432.pdf

Who?

- Ruby Standardization Working Group
- Chairman: Ikuo Nakata
- Shugo Maeda

Why?

- Supposedly for government procurement in Japan
- Another way to know what your code means
- Tell Ruby users what they can count on
- Tell Ruby implementers what they can get do

Overview

1) Scope
2) Normative references
3) Conformance
4) Terms and definitions
5) Notational conventions
6) Fundamental concepts
7) Execution contexts
8) Lexical structure
9) Scope of variables
10) Program structure
11) Expressions
12) Statements
13) Classes and modules
14) Exceptions
15) **Built-in classes & modules**

Built-in classes & modules

- Object
- Module
- Class
- NilClass
- TrueClass
- FalseClass
- Numeric
- Integer
- Float
- String
- Symbol
- Array
- Hash
- Range
- Regexp
- MatchData
- Proc
- Struct
- Time
- IO
- File
- Exception…
- Kernel
- Enumerable
- Comparable

Return value of #puts

- See section 15.2.20.5.13
- Return value is *implementation defined*

```
puts "hello" or puts "hi"
```

Return value of Array#each

- See section 15.2.12.5.10
- If a block is given, returns the receiver

```
a = [1, 2, 3].each { |n| puts n }
```

- If a block is not given, the behavior is *unspecified*

Terms

- *Unspecified:* Possibly differing between implementations, and not necessarily defined for any particular implementation.
- *Implementation defined:* Possibly differing between implementations, but defined for every implementation.

Enumerable#sort

```
a = [x, y, z].sort
```

- See section 15.3.2.2.19
- Overly specific: says to create an empty array.
- Definition depends on what exactly “suppose” means.
- Doesn't actually require that any two elements be compared.
- Disallows comparing the same objects twice!

Better specification of sort

- Define what it means for results from the comparison operation to be well-behaved.
- If the results are not well-behaved, behavior of #sort is unspecified.
- #sort must do enough comparisons so there is no ambiguity in the ordering of objects.
- #sort must return a sorted array.

Ruby ISO vs. RubySpec

- RubySpec is executable.
  - Based on a finite number of test cases
  - Cannot test *implentation defined* and *unspecified*
- RubySpec only really tests the interpreter, doesn't define valid Ruby programs.

RubySpec sample

```
it "dumps a Fixnum" do
  [ [Marshal,  0,       "\004\bi\000"],
    [Marshal,  5,       "\004\bi\n"],
    [Marshal,  8,       "\004\bi\r"],
    [Marshal,  122,     "\004\bi\177"],
    [Marshal,  123,     "\004\bi\001{"],
    [Marshal,  1234,    "\004\bi\002\322\004"],
    [Marshal, -8,       "\004\bi\363"],
    [Marshal, -123,     "\004\bi\200"],
    [Marshal, -124,     "\004\bi\377\204"],
    [Marshal, -1234,    "\004\bi\376.\373"],
    [Marshal, -4516727, "\004\bi\375\211\024\273"],
    [Marshal,  2**8,    "\004\bi\002\000\001"],
    [Marshal,  2**16,   "\004\bi\003\000\000\001"],
    [Marshal,  2**24,   "\004\bi\004\000\000\000\001"],
    [Marshal, -2**8,    "\004\bi\377\000"],
    [Marshal, -2**16,   "\004\bi\376\000\000"],
    [Marshal, -2**24,   "\004\bi\375\000\000\000"],
  ].should be_computed_by(:dump)
end
```

https://github.com/rubyspec/rubyspec/blob/archive/core/marshal/dump_spec.rb

Comparison to C++ standard

- C++ designer Bjarne Stroustrup stuck with the ISO standards efforts for 20+ years.
- Periodic new versions: C++11, C++14, C++17...
- 1354 pages
- Covers the whole standard library
- Concept of undefined behavior
- Standard is regularly mentioned on StackOverflow

References

https://www.ipa.go.jp/files/000011432.pdf

https://www.ipa.go.jp/osc/english/ruby/

http://www.rubyinside.com/ruby-iso-spec-draft-2900.html

http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-core/26969