

# R6RS: Scheme

Version 9.0.0.10

December 15, 2025

The The Revised<sup>6</sup> Report on the Algorithmic Language Scheme defines a dialect of Scheme. We use *R<sup>6</sup>RS* to refer to both the standard and the language defined by the standard.

*R<sup>6</sup>RS* defines both *libraries* and *top-level programs*. Both correspond to Racket *modules* (see §6 “Modules”). That is, although *R<sup>6</sup>RS* defines top-level programs as entry points, you can just as easily treat a library as an entry point when using Racket. The only difference is that an *R<sup>6</sup>RS* top-level program cannot export any bindings to other modules.

See §23 “Dialects of Racket and Scheme” for general information about different dialects of Scheme within Racket.

# Contents

<b>1</b>	<b>Using R<sup>6</sup>RS with DrRacket</b>	<b>4</b>
<b>2</b>	<b>Running Top-Level Programs</b>	<b>5</b>
<b>3</b>	<b>Installing Libraries</b>	<b>6</b>
<b>4</b>	<b>R<sup>6</sup>RS Module Language</b>	<b>8</b>
4.1	Using R <sup>6</sup> RS . . . . .	8
4.2	The Implementation of R <sup>6</sup> RS . . . . .	8
<b>5</b>	<b>Libraries and Collections</b>	<b>9</b>
<b>6</b>	<b>Language Interoperability</b>	<b>10</b>
<b>7</b>	<b>R<sup>6</sup>RS Conformance</b>	<b>11</b>
<b>8</b>	<b>R<sup>6</sup>RS Libraries</b>	<b>13</b>
8.1	( <code>rnrs base (6)</code> ): Base . . . . .	13
8.2	( <code>rnrs unicode (6)</code> ): Unicode . . . . .	13
8.3	( <code>rnrs bytevectors (6)</code> ): Bytevectors . . . . .	13
8.4	( <code>rnrs lists (6)</code> ): List utilities . . . . .	13
8.5	( <code>rnrs sorting (6)</code> ): Sorting . . . . .	13
8.6	( <code>rnrs control (6)</code> ): Control Structures . . . . .	13
8.7	( <code>rnrs records syntactic (6)</code> ): Records: Syntactic . . . . .	14
8.8	( <code>rnrs records procedural (6)</code> ): Records: Procedural . . . . .	14
8.9	( <code>rnrs records inspection (6)</code> ): Records: Inspection . . . . .	14
8.10	( <code>rnrs exceptions (6)</code> ): Exceptions . . . . .	14

8.11 <code>(rnrs conditions (6))</code> : Conditions . . . . .	14
8.12 <code>(rnrs io ports (6))</code> : I/O: Ports . . . . .	14
8.13 <code>(rnrs io simple (6))</code> : I/O: Simple . . . . .	15
8.14 <code>(rnrs files (6))</code> : File System . . . . .	15
8.15 <code>(rnrs programs (6))</code> : Command-line Access and Exit Values . . . . .	15
8.16 <code>(rnrs arithmetic fixnums (6))</code> : Arithmetic: Fixnums . . . . .	15
8.17 <code>(rnrs arithmetic flonums (6))</code> : Arithmetic: Flonums . . . . .	15
8.18 <code>(rnrs arithmetic bitwise (6))</code> : Arithmetic: Bitwise . . . . .	15
8.19 <code>(rnrs syntax-case (6))</code> : Syntax-Case . . . . .	16
8.20 <code>(rnrs hashtables (6))</code> : Hashtables . . . . .	16
8.21 <code>(rnrs enums (6))</code> : Enumerations . . . . .	16
8.22 <code>(rnrs eval (6))</code> : Eval . . . . .	16
8.23 <code>(rnrs mutable-pairs (6))</code> : Mutable Pairs . . . . .	16
8.24 <code>(rnrs mutable-strings (6))</code> : Mutable Strings . . . . .	16
8.25 <code>(rnrs r5rs (6))</code> : R5RS Compatibility . . . . .	17
<b>Index</b>	<b>18</b>
<b>Index</b>	<b>18</b>

# 1 Using R<sup>6</sup>RS with DrRacket

To run an R<sup>6</sup>RS program with DrRacket choose Use language declared in source from the language dialog box and add the following line to the top of your program. `#!r6rs`.

Here is a small example R<sup>6</sup>RS program that will work in DrRacket.

```
#!r6rs
(import (rnrs lists (6))
        (rnrs base (6))
        (rnrs io simple (6)))
(display (find even? '(3 1 4 1 5 9)))
```

## 2 Running Top-Level Programs

To run a top-level program, either:

- Use the `plt-r6rs` executable, supplying the file that contains the program on the command line:

```
plt-r6rs <program-file>
```

Additional command-line arguments are propagated as command-line arguments to the program (accessed via `command-line`).

To compile the file to bytecode (to speed future runs of the program), use `plt-r6rs` with the `--compile` flag:

```
plt-r6rs --compile <program-file>
```

The bytecode file is written in a "compiled" sub-directory next to `<program-file>`.

For example, if "hi.sps" contains

```
(import (rnrs))
(display "hello\n")
```

then

```
plt-r6rs hi.sps
```

prints "hello."

- Prefix the program with `#!r6rs`, which counts as a comment from the R<sup>6</sup>RS perspective, but is a synonym for `#lang r6rs` from the Racket perspective. Such files can be run like any other Racket module, such as using `racket`:

```
racket <program-file>
```

or using DrRacket. The file can also be compiled to bytecode using `raco make`:

```
raco make <program-file>
```

For example, if "hi.sps" contains

```
#!r6rs
(import (rnrs))
(display "hello\n")
```

then

```
racket hi.sps
```

prints "hello." Similarly, opening "hi.sps" in DrRacket and clicking Run prints "hello" within the DrRacket interactions window.

### 3 Installing Libraries

To reference an R<sup>6</sup>RS library from a top-level program or another library, it must be installed as a collection-based library in Racket.

One way to produce an R<sup>6</sup>RS installed library is to create in a collection a file that starts with `#!r6rs` and that contains a `library` form. For example, the following file might be created in a "hello.sls" file within a "examples" collection directory:

```
#!r6rs
(library (examples hello)
  (export greet)
  (import (rnrs))

  (define (greet)
    (display "hello\n")))
```

Alternately, the `plt-r6rs` executable with the `--install` flag accepts a sequence of `library` declarations and installs them into separate files in a collection directory, based on the declared name of each library:

```
plt-r6rs --install <libraries-file>
```

By default, libraries are installed into the user-specific collection directory (see [find-user-collects-dir](#)). The `--all-users` flag causes the libraries to be installed into the main installation, instead (see [find-collects-dir](#)):

```
plt-r6rs --install --all-users <libraries-file>
```

You may as well specify an arbitrary collections directory by using the `--collections` flag:

```
plt-r6rs --install --collections <directory> <libraries-file>
```

See §5 “Libraries and Collections” for information on how R<sup>6</sup>RS library names are turned into collection-based module paths, which determines where the files are written. Libraries installed by `plt-r6rs --install` are automatically compiled to bytecode form.

One final option is to supply a `++path` flag to `plt-r6rs`. A path added with `++path` extends the set of directories that are searched to find a collection (i.e., it sets [current-collection-paths](#)). If `<dir>` contains "duck" and "cow" sub-directories with "duck/feather.sls" and "cow/bell.sls", and if each file is an R<sup>6</sup>RS library prefixed with `#!r6rs`, then `plt-r6rs ++path <dir>` directs the R<sup>6</sup>RS library references `(duck feather)` and `(cow bell)` to the files. Note that this technique does not support accessing "duck.sls" directly within `<dir>`, since the library reference `(duck)` is treated like `(duck main)` for finding the library, as explained in §5 “Libraries and Collections”. Multiple paths

can be provided with multiple uses of `++path`; the paths are searched in order, and before the installation's collections.

## 4 R<sup>6</sup>RS Module Language

```
#lang r6rs      package: r6rs-lib
```

The `r6rs` language is usually used in the form `#!r6rs`, which is equivalent to `#lang r6rs` and is also valid R<sup>6</sup>RS syntax.

### 4.1 Using R<sup>6</sup>RS

See §1 “Using R<sup>6</sup>RS with DrRacket”, §2 “Running Top-Level Programs”, and §3 “Installing Libraries” for more information on writing and running R<sup>6</sup>RS programs with Racket.

### 4.2 The Implementation of R<sup>6</sup>RS

The R<sup>6</sup>RS language is itself implemented as a module within Racket. The details of that implementation, as provided in this section, are not normally relevant to programmers using R<sup>6</sup>RS; see the links in §4.1 “Using R<sup>6</sup>RS”, instead. The details may be relevant to programmers who are developing new tools or deriving variants of R<sup>6</sup>RS within Racket.

As a Racket module, the `r6rs` module language provides only a `#%module-begin` binding, which is used to process the entire body of a Racket module (see `module`). The `#%module-begin` binding from `r6rs` allows the body of a module to use the syntax of either a R<sup>6</sup>RS library or a R<sup>6</sup>RS top-level program.

```
(#%module-begin
  (library library-name
    (export export-spec ...)
    (import import-spec ...)
    library-body ...))
(#%module-begin
  (import import-spec ...)
  program-body ...)
```

An `r6rs` module that contains a single `library` form defines an R<sup>6</sup>RS library, while a module body that starts with an `import` form defines an R<sup>6</sup>RS top-level program.

The `library`, `export`, and `import` identifiers are not exported by the `r6rs` library; they are recognized through equivalence to unbound identifiers.

## 5 Libraries and Collections

An R<sup>6</sup>RS library name is sequence of symbols, optionally followed by a version as a sequence of exact, non-negative integers. Roughly, such a name is converted to a Racket module pathname (see §6.3 “Module Paths”) by concatenating the symbols with a `/` separator, and then appending the version integers each with a preceding `-`. As a special case, when an R<sup>6</sup>RS path contains a single symbol (optionally followed by a version), a `main` symbol is effectively inserted after the initial symbol. See below for further encoding considerations.

When an R<sup>6</sup>RS library or top-level program refers to another library, it can supply version constraints rather than naming a specific version. Version constraints are always resolved at compile time by searching the set of installed files.

In addition, when an R<sup>6</sup>RS library path is converted, a file extension is selected at compile time based on installed files. The search order for file extensions is `".mzscheme.ss"`, `".mzscheme.sls"`, `".ss"`, `".sls"`, and `".rkt"`. When resolving version constraints, these extensions are all tried when looking for matches.

To ensure that all R<sup>6</sup>RS library names can be converted to a unique and distinct library module path, the following conversions are applied to each symbol before concatenating them:

- The symbol is encoded using UTF-8, and the resulting bytes are treated as Latin-1 encoded characters. ASCII letters, digits, `+`, `-`, and `_` are left as-is; other characters are replaced by `%` followed by two lowercase hexadecimal digits. Note that UTF-8 encodes ASCII letters, digits, etc. as themselves, so typical library names correspond to readable module paths.
- If the R<sup>6</sup>RS library reference has two symbol elements and the second one is `main` followed by any number of underscores, then an extra underscore is added to that symbol. This conversion avoids a collision between an explicit `main` and the implicit `main` when a library path has a single symbol element.

Examples (assuming a typical Racket installation):

<code>(rnrs io simple (6))</code>	means	<code>(lib "rnrs/io/simple-6.rkt")</code>
<code>(rnrs)</code>	means	<code>(lib "rnrs/main-6.rkt")</code>
<code>(rnrs main)</code>	means	<code>(lib "rnrs/main_.rkt")</code>
<code>(rnrs (6))</code>	means	<code>(lib "rnrs/main-6.rkt")</code>
<code>(racket base)</code>	means	<code>(lib "racket/base.rkt")</code>
<code>(achtung!)</code>	means	<code>(lib "achtung%21/main.rkt")</code>
<code>(funco new-λ)</code>	means	<code>(lib "funco/new-%ce%bb.rkt")</code>

## 6 Language Interoperability

Using the conversion rules in §5 “Libraries and Collections”, and R<sup>6</sup>RS library can refer to modules that are implemented in other dialects supported by Racket, and other Racket modules can refer to libraries that are implemented in R<sup>6</sup>RS.

Beware that a *pair* in R<sup>6</sup>RS corresponds to a *mutable pair* in `racket/base`. Otherwise, R<sup>6</sup>RS libraries and `racket/base` share the same datatype for numbers, characters, strings, bytevectors (a.k.a. byte strings), vectors, and so on. Hash tables are different. Input and output ports from `racket/base` can be used directly as binary ports with R<sup>6</sup>RS libraries, and all R<sup>6</sup>RS ports can be used as ports in `racket/base` programs, but only textual ports created via R<sup>6</sup>RS libraries can be used by other R<sup>6</sup>RS operations that expect textual ports.

## 7 R<sup>6</sup>RS Conformance

Racket's R<sup>6</sup>RS support does not conform with the standard in several known ways:

- When guard catches an exception that no clause matches, the exception is re-`raiseed` without restoring the continuation to the one that raised the exception.

This difference can be made visible using `dynamic-wind`. According to R<sup>6</sup>RS, the following program should print "in" and "out" twice, but each prints once using Racket:

```
(guard (exn [(equal? exn 5) 'five])
  (guard (exn [(equal? exn 6) 'six])
    (dynamic-wind
      (lambda () (display "in") (newline))
      (lambda () (raise 5))
      (lambda () (display "out") (newline))))))
```

Along similar lines, continuation capture and invocation within an exception handler is restricted. Unless the exception is raised through `raise-continuable`, a handler can escape only through a continuation that is a tail of the current continuation, and a continuation captured within the handler cannot be invoked after control escapes from the raise.

The initial exception handler does not return for non-`&serious` conditions, but `raise` and `raise-continuable` both install an uncaught-exception handler (via `parameterize` and `uncaught-exception-handler`) to one that returns for non-`&serious` conditions.

- Inexact numbers are printed without a precision indicator, and precision indicators are ignored on input (e.g., `0.5|7` is read the same as `0.5`).
- Word boundaries for `string-downcase`, `string-upcase`, and `string-titlecase` are not determined as specified by Unicode Standard Annex #29.
- A custom textual port must represent positions using integers, and the positions must correspond to bytes in a UTF-8 encoding of the port's data. For custom ports (byte or character) that support both input and output, beware that buffered input can create a mismatch between the position implemented by the custom procedures and the port's current position; the result from a custom position procedure is automatically adjusted to account for buffering, and setting the port's position flushes all buffered bytes, but writing after a read does *not* automatically reset the port's position to counteract the effects of buffering.
- The bindings in a namespace produced by `null-environment` or `scheme-report-environment` correspond to R<sup>5</sup>RS bindings instead of R<sup>6</sup>RS bindings. In particular, `=>`, `else`, `_`, and `...` are not bound.

- Bindings for `#%datum`, `#%app`, `#%top`, and `#%top-interaction` are imported into every library and program, and at every phase level for which the library or program has imports.

Changed in version 6.0.1.4: When an identifier bound by `letrec` or `letrec*` is referenced before it is initialized, an exception is raised, instead of producing `#<undefined>`.

## 8 R<sup>6</sup>RS Libraries

### 8.1 (rnrs base (6)): Base

(require rnrs/base-6) package: r6rs-lib

Original specification: Base

### 8.2 (rnrs unicode (6)): Unicode

(require rnrs/unicode-6) package: r6rs-lib

Original specification: Unicode

### 8.3 (rnrs bytevectors (6)): Bytevectors

(require rnrs/bytevectors-6) package: r6rs-lib

Original specification: Bytevectors

### 8.4 (rnrs lists (6)): List utilities

(require rnrs/lists-6) package: r6rs-lib

Original specification: List utilities

### 8.5 (rnrs sorting (6)): Sorting

(require rnrs/sorting-6) package: r6rs-lib

Original specification: Sorting

### 8.6 (rnrs control (6)): Control Structures

(require rnrs/control-6) package: r6rs-lib

Original specification: Control Structures

## 8.7 (rnrs records syntactic (6)): Records: Syntactic

(require rnrs/records/syntactic-6) package: r6rs-lib

Original specification: Records: Syntactic

## 8.8 (rnrs records procedural (6)): Records: Procedural

(require rnrs/records/procedural-6) package: r6rs-lib

Original specification: Records: Procedural

## 8.9 (rnrs records inspection (6)): Records: Inspection

(require rnrs/records/inspection-6) package: r6rs-lib

Original specification: Records: Inspection

## 8.10 (rnrs exceptions (6)): Exceptions

(require rnrs/exceptions-6) package: r6rs-lib

Original specification: Exceptions

See also §7 “R<sup>6</sup>RS Conformance”.

## 8.11 (rnrs conditions (6)): Conditions

(require rnrs/conditions-6) package: r6rs-lib

Original specification: Conditions

## 8.12 (rnrs io ports (6)): I/O: Ports

(require rnrs/io/ports-6) package: r6rs-lib

Original specification: I/O: Ports

### 8.13 (rnrs io simple (6)): I/O: Simple

(require rnrs/io/simple-6) package: r6rs-lib

Original specification: I/O: Simple

### 8.14 (rnrs files (6)): File System

(require rnrs/files-6) package: r6rs-lib

Original specification: File System

### 8.15 (rnrs programs (6)): Command-line Access and Exit Values

(require rnrs/programs-6) package: r6rs-lib

Original specification: Command-line Access and Exit Values

### 8.16 (rnrs arithmetic fixnums (6)): Arithmetic: Fixnums

(require rnrs/arithmetic/fixnums-6) package: r6rs-lib

Original specification: Arithmetic: Fixnums

### 8.17 (rnrs arithmetic flonums (6)): Arithmetic: Flonums

(require rnrs/arithmetic/flonums-6) package: r6rs-lib

Original specification: Arithmetic: Flonums

### 8.18 (rnrs arithmetic bitwise (6)): Arithmetic: Bitwise

(require rnrs/arithmetic/bitwise-6) package: r6rs-lib

Original specification: Arithmetic: Bitwise

## 8.19 (rnrs syntax-case (6)): Syntax-Case

(require rnrs/syntax-case-6) package: r6rs-lib

Original specification: Syntax-Case

## 8.20 (rnrs hashtables (6)): Hashtables

(require rnrs/hashtables-6) package: r6rs-lib

Original specification: Hashtables

A hashtable is a dictionary in the sense of `racket/dict`, and hash table operations interact with threads in the same way for hash tables created with `make-hash` (e.g., `hashtable-ref` and `hashtable-set!` are thread-safe).

## 8.21 (rnrs enums (6)): Enumerations

(require rnrs/enums-6) package: r6rs-lib

Original specification: Enumerations

## 8.22 (rnrs eval (6)): Eval

(require rnrs/eval-6) package: r6rs-lib

Original specification: Eval

## 8.23 (rnrs mutable-pairs (6)): Mutable Pairs

(require rnrs/mutable-pairs-6) package: r6rs-lib

Original specification: Mutable Pairs

## 8.24 (rnrs mutable-strings (6)): Mutable Strings

(require rnrs/mutable-strings-6) package: r6rs-lib

Original specification: Mutable Strings

## 8.25 (rnrs r5rs (6)): R5RS Compatibility

(require rnrs/r5rs-6) package: r6rs-lib

Original specification: R5RS Compatibility

See also §7 “R<sup>6</sup>RS Conformance”.

## Index

%#module-begin, 8  
&**assertion**, 14  
&**condition**, 14  
&**error**, 14  
&**i/o**, 14  
&**i/o-decoding**, 14  
&**i/o-encoding**, 14  
&**i/o-file-already-exists**, 14  
&**i/o-file-does-not-exist**, 14  
&**i/o-file-is-read-only**, 14  
&**i/o-file-protection**, 14  
&**i/o-filename**, 14  
&**i/o-invalid-position**, 14  
&**i/o-port**, 14  
&**i/o-read**, 14  
&**i/o-write**, 14  
&**implementation-restriction**, 14  
&**irritants**, 14  
&**lexical**, 14  
&**message**, 14  
&**no-infinities**, 15  
&**no-nans**, 15  
&**non-continuable**, 14  
&**serious**, 14  
&**syntax**, 14  
&**undefined**, 14  
&**violation**, 14  
&**warning**, 14  
&**who**, 14  
(**rnrs arithmetic bitwise** (6)): Arithmetic: Bitwise, 15  
(**rnrs arithmetic fixnums** (6)): Arithmetic: Fixnums, 15  
(**rnrs arithmetic flonums** (6)): Arithmetic: Flonums, 15  
(**rnrs base** (6)): Base, 13  
(**rnrs bytevectors** (6)): Bytevectors, 13  
(**rnrs conditions** (6)): Conditions, 14  
(**rnrs control** (6)): Control Structures, 13  
(**rnrs exceptions** (6)): Exceptions, 14  
(**rnrs files** (6)): File System, 15  
(**rnrs hashtables** (6)): Hashtables, 16  
(**rnrs io ports** (6)): I/O: Ports, 14  
(**rnrs io simple** (6)): I/O: Simple, 15  
(**rnrs lists** (6)): List utilities, 13  
(**rnrs mutable-pairs** (6)): Mutable Pairs, 16  
(**rnrs mutable-strings** (6)): Mutable Strings, 16  
(**rnrs programs** (6)): Command-line Access and Exit Values, 15  
(**rnrs r5rs** (6)): R5RS Compatibility, 17  
(**rnrs records inspection** (6)): Records: Inspection, 14  
(**rnrs records procedural** (6)): Records: Procedural, 14  
(**rnrs records syntactic** (6)): Records: Syntactic, 14  
(**rnrs sorting** (6)): Sorting, 13  
(**rnrs syntax-case** (6)): Syntax-Case, 16  
(**rnrs unicode** (6)): Unicode, 13  
\*, 13  
+, 13  
++path, 6  
-, 13  
..., 13  
..., 16  
/, 13  
<, 13  
<=, 13  
=, 13  
=>, 13  
=>, 14  
>, 13  
>=, 13  
\_, 13  
\_, 16  
abs, 13

`acos`, 13  
`and`, 13  
`angle`, 13  
`append`, 13  
`apply`, 13  
`asin`, 13  
`assert`, 13  
`assertion-violation`, 13  
`assertion-violation?`, 14  
`assoc`, 13  
`assp`, 13  
`assq`, 13  
`assv`, 13  
`atan`, 13  
`begin`, 13  
`binary-port?`, 14  
`bitwise-and`, 15  
`bitwise-arithmetic-shift`, 15  
`bitwise-arithmetic-shift-left`, 15  
`bitwise-arithmetic-shift-right`, 15  
`bitwise-bit-count`, 15  
`bitwise-bit-field`, 15  
`bitwise-bit-set?`, 15  
`bitwise-copy-bit`, 15  
`bitwise-copy-bit-field`, 15  
`bitwise-first-bit-set`, 15  
`bitwise-if`, 15  
`bitwise-ior`, 15  
`bitwise-length`, 15  
`bitwise-not`, 15  
`bitwise-reverse-bit-field`, 15  
`bitwise-rotate-bit-field`, 15  
`bitwise-xor`, 15  
`boolean=?`, 13  
`boolean?`, 13  
`bound-identifier=?`, 16  
`buffer-mode`, 14  
`buffer-mode?`, 14  
`bytevector->sint-list`, 13  
`bytevector->string`, 14  
`bytevector->u8-list`, 13  
`bytevector->uint-list`, 13  
`bytevector-copy`, 13  
`bytevector-copy!`, 13  
`bytevector-fill!`, 13  
`bytevector-ieee-double-native-ref`, 13  
`bytevector-ieee-double-native-set!`, 13  
`bytevector-ieee-double-ref`, 13  
`bytevector-ieee-single-native-ref`, 13  
`bytevector-ieee-single-native-set!`, 13  
`bytevector-ieee-single-ref`, 13  
`bytevector-length`, 13  
`bytevector-s16-native-ref`, 13  
`bytevector-s16-native-set!`, 13  
`bytevector-s16-ref`, 13  
`bytevector-s16-set!`, 13  
`bytevector-s32-native-ref`, 13  
`bytevector-s32-native-set!`, 13  
`bytevector-s32-ref`, 13  
`bytevector-s32-set!`, 13  
`bytevector-s64-native-ref`, 13  
`bytevector-s64-native-set!`, 13  
`bytevector-s64-ref`, 13  
`bytevector-s64-set!`, 13  
`bytevector-s8-ref`, 13  
`bytevector-s8-set!`, 13  
`bytevector-sint-ref`, 13  
`bytevector-sint-set!`, 13  
`bytevector-u16-native-ref`, 13  
`bytevector-u16-native-set!`, 13  
`bytevector-u16-ref`, 13  
`bytevector-u16-set!`, 13  
`bytevector-u32-native-ref`, 13  
`bytevector-u32-native-set!`, 13  
`bytevector-u32-ref`, 13  
`bytevector-u32-set!`, 13  
`bytevector-u64-native-ref`, 13  
`bytevector-u64-native-set!`, 13  
`bytevector-u64-ref`, 13  
`bytevector-u64-set!`, 13

bytevector-u8-ref, 13	char<?, 13
bytevector-u8-set!, 13	char=?, 13
bytevector-uint-ref, 13	char>=?, 13
bytevector-uint-set!, 13	char>?, 13
bytevector=? , 13	char?, 13
bytevector?, 13	close-input-port, 15
caa , 13	close-output-port, 15
cadr , 13	close-port, 14
call-with-bytevector-output-port, 14	command-line, 15
call-with-current-continuation, 13	complex?, 13
call-with-input-file, 15	cond, 13
call-with-output-file, 15	condition, 14
call-with-port, 14	condition-accessor, 14
call-with-string-output-port, 14	condition-irritants, 14
call-with-values, 13	condition-message, 14
call/cc, 13	condition-predicate, 14
car, 13	condition-who, 14
case, 13	condition?, 14
case-lambda, 13	cons, 13
cdddar, 13	cons*, 13
cddddr, 13	cos, 13
cdr, 13	current-error-port, 14
ceiling, 13	current-input-port, 14
char->integer, 13	current-output-port, 14
char-alphabetic?, 13	datum->syntax, 16
char-ci<=? , 13	define, 13
char-ci<?, 13	define-condition-type, 14
char-ci=?, 13	define-enumeration, 16
char-ci>=? , 13	define-record-type, 14
char-ci>?, 13	define-syntax, 13
char-downcase, 13	delay, 17
char-foldcase, 13	delete-file, 15
char-general-category, 13	denominator, 13
char-lower-case?, 13	display, 15
char-numeric?, 13	div, 13
char-title-case?, 13	div-and-mod, 13
char-titlecase, 13	div0, 13
char-upcase, 13	div0-and-mod0, 13
char-upper-case?, 13	do, 13
char-whitespace?, 13	dynamic-wind, 13
char<=? , 13	else, 13
	else, 14

endianness, 13	fixnum?, 15
enum-set->list, 16	fl*, 15
enum-set-complement, 16	fl+, 15
enum-set-constructor, 16	fl-, 15
enum-set-difference, 16	fl/, 15
enum-set-indexer, 16	fl<?, 15
enum-set-intersection, 16	fl<?, 15
enum-set-member?, 16	fl=?, 15
enum-set-projection, 16	fl>?, 15
enum-set-subset?, 16	fl>?, 15
enum-set-union, 16	flabs, 15
enum-set-universe, 16	flacos, 15
enum-set=?, 16	flasin, 15
environment, 16	flatan, 15
eof-object, 14	flceiling, 15
eof-object?, 14	flcos, 15
eol-style, 14	fldenominator, 15
eq?, 13	fldiv, 15
equal-hash, 16	fldiv-and-mod, 15
equal?, 13	fldiv0, 15
eqv?, 13	fldiv0-and-mod0, 15
error, 13	fleven?, 15
error-handling-mode, 14	flexp, 15
error?, 14	flexpt, 15
eval, 16	flfinite?, 15
even?, 13	flfloor, 15
exact, 13	flinfinite?, 15
exact->inexact, 17	flinteger?, 15
exact-integer-sqrt, 13	fllog, 15
exact?, 13	flmax, 15
exists, 13	flmin, 15
exit, 15	flmod, 15
exp, 13	flmod0, 15
expt, 13	flnan?, 15
fields, 14	flnegative?, 15
file-exists?, 15	flnumerator, 15
file-options, 14	flodd?, 15
filter, 13	flonum?, 15
find, 13	floor, 13
finite?, 13	flpositive?, 15
fixnum->flonum, 15	flround, 15
fixnum-width, 15	flsin, 15

<code>f(sqrt</code>	, 15	<code>fxmod</code> , 15
<code>f(tan</code>	, 15	<code>fxmod0</code> , 15
<code>ftruncate</code>	, 15	<code>fxnegative?</code> , 15
<code>flush-output-port</code>	, 14	<code>fxnot</code> , 15
<code>flzero?</code>	, 15	<code>fxodd?</code> , 15
<code>fold-left</code>	, 13	<code>fxpositive?</code> , 15
<code>fold-right</code>	, 13	<code>fxreverse-bit-field</code> , 15
<code>for-all</code>	, 13	<code>fxrotate-bit-field</code> , 15
<code>for-each</code>	, 13	<code>fxxor</code> , 15
<code>force</code>	, 17	<code>fxzero?</code> , 15
<code>free-identifier=?</code>	, 16	<code>gcd</code> , 13
<code>fx*</code>	, 15	<code>generate-temporaries</code> , 16
<code>fx*/carry</code>	, 15	<code>get-bytevector-all</code> , 14
<code>fx+</code>	, 15	<code>get-bytevector-n</code> , 14
<code>fx+/carry</code>	, 15	<code>get-bytevector-n!</code> , 14
<code>fx-</code>	, 15	<code>get-bytevector-some</code> , 14
<code>fx-/carry</code>	, 15	<code>get-char</code> , 14
<code>fx&lt;=?</code>	, 15	<code>get-datum</code> , 14
<code>fx&lt;?</code>	, 15	<code>get-line</code> , 14
<code>fx=?</code>	, 15	<code>get-string-all</code> , 14
<code>fx&gt;=?</code>	, 15	<code>get-string-n</code> , 14
<code>fx&gt;?</code>	, 15	<code>get-string-n!</code> , 14
<code>fxand</code>	, 15	<code>get-u8</code> , 14
<code>fxarithmetic-shift</code>	, 15	<code>greatest-fixnum</code> , 15
<code>fxarithmetic-shift-left</code>	, 15	<code>guard</code> , 14
<code>fxarithmetic-shift-right</code>	, 15	<code>hashtable-clear!</code> , 16
<code>fxbit-count</code>	, 15	<code>hashtable-contains?</code> , 16
<code>fxbit-field</code>	, 15	<code>hashtable-copy</code> , 16
<code>fxbit-set?</code>	, 15	<code>hashtable-delete!</code> , 16
<code>fxcopy-bit</code>	, 15	<code>hashtable-entries</code> , 16
<code>fxcopy-bit-field</code>	, 15	<code>hashtable-equivalence-function</code> , 16
<code>fxdiv</code>	, 15	<code>hashtable-hash-function</code> , 16
<code>fxdiv-and-mod</code>	, 15	<code>hashtable-keys</code> , 16
<code>fxdiv0</code>	, 15	<code>hashtable-mutable?</code> , 16
<code>fxdiv0-and-mod0</code>	, 15	<code>hashtable-ref</code> , 16
<code>fxeven?</code>	, 15	<code>hashtable-set!</code> , 16
<code>fxfirst-bit-set</code>	, 15	<code>hashtable-size</code> , 16
<code>fxif</code>	, 15	<code>hashtable-update!</code> , 16
<code>fxior</code>	, 15	<code>hashtable?</code> , 16
<code>fxlength</code>	, 15	<code>i/o-decoding-error?</code> , 14
<code>fxmax</code>	, 15	<code>i/o-encoding-error-char</code> , 14
<code>fxmin</code>	, 15	<code>i/o-encoding-error?</code> , 14

*i/o-error-filename*, 14  
*i/o-error-port*, 14  
*i/o-error-position*, 14  
*i/o-error?*, 14  
*i/o-file-already-exists-error?*, 14  
*i/o-file-does-not-exist-error?*, 14  
*i/o-file-is-read-only-error?*, 14  
*i/o-file-protection-error?*, 14  
*i/o-filename-error?*, 14  
*i/o-invalid-position-error?*, 14  
*i/o-port-error?*, 14  
*i/o-read-error?*, 14  
*i/o-write-error?*, 14  
*identifier-syntax*, 13  
*identifier?*, 16  
*if*, 13  
*imag-part*, 13  
*immutable*, 14  
*implementation-restriction-violation?*, 14  
*inexact*, 13  
*inexact->exact*, 17  
*inexact?*, 13  
*infinite?*, 13  
*input-port?*, 14  
 Installing Libraries, 6  
*integer->char*, 13  
*integer-valued?*, 13  
*integer?*, 13  
*irritants-condition?*, 14  
*lambda*, 13  
 Language Interoperability, 10  
*latin-1-codec*, 14  
*lcm*, 13  
*least-fixnum*, 15  
*length*, 13  
*let*, 13  
*let\**, 13  
*let\*-values*, 13  
*let-syntax*, 13  
*let-values*, 13  
*letrec*, 13  
*letrec\**, 13  
*letrec-syntax*, 13  
*lexical-violation?*, 14  
 Libraries and Collections, 9  
*list*, 13  
*list->string*, 13  
*list->vector*, 13  
*list-ref*, 13  
*list-sort*, 13  
*list-tail*, 13  
*list?*, 13  
*log*, 13  
*lookahead-char*, 14  
*lookahead-u8*, 14  
*magnitude*, 13  
*make-assertion-violation*, 14  
*make-bytevector*, 13  
*make-custom-binary-input-port*, 14  
*make-custom-binary-input/output-port*, 14  
*make-custom-binary-output-port*, 14  
*make-custom-textual-input-port*, 14  
*make-custom-textual-input/output-port*, 14  
*make-custom-textual-output-port*, 14  
*make-enumeration*, 16  
*make-eq-hashtable*, 16  
*make-eqv-hashtable*, 16  
*make-error*, 14  
*make-hashtable*, 16  
*make-i/o-decoding-error*, 14  
*make-i/o-encoding-error*, 14  
*make-i/o-error*, 14  
*make-i/o-file-already-exists-error*, 14  
*make-i/o-file-does-not-exist-error*, 14  
*make-i/o-file-is-read-only-error*, 14  
*make-i/o-file-protection-error*, 14  
*make-i/o-filename-error*, 14  
*make-i/o-invalid-position-error*, 14

**make-i/o-port-error**, 14  
**make-i/o-read-error**, 14  
**make-i/o-write-error**, 14  
**make-implementation-restriction-violation**, 14  
**make-irritants-condition**, 14  
**make-lexical-violation**, 14  
**make-message-condition**, 14  
**make-no-infinities-violation**, 15  
**make-no-nans-violation**, 15  
**make-non-continuable-violation**, 14  
**make-polar**, 13  
**make-record-constructor-descriptor**, 14  
**make-record-type-descriptor**, 14  
**make-rectangular**, 13  
**make-serious-condition**, 14  
**make-string**, 13  
**make-syntax-violation**, 14  
**make-transcoder**, 14  
**make-undefined-violation**, 14  
**make-variable-transformer**, 16  
**make-vector**, 13  
**make-violation**, 14  
**make-warning**, 14  
**make-who-condition**, 14  
**map**, 13  
**max**, 13  
**member**, 13  
**memp**, 13  
**memq**, 13  
**memv**, 13  
**message-condition?**, 14  
**min**, 13  
**mod**, 13  
**mod0**, 13  
**modulo**, 17  
**mutable**, 14  
**nan?**, 13  
**native-endianness**, 13  
**native-eol-style**, 14  
**native-transcoder**, 14  
**negative?**, 13  
**newline**, 15  
**no-infinities-violation?**, 15  
**no-nans-violation?**, 15  
**non-continuable-violation?**, 14  
**nongenerative**, 14  
**not**, 13  
**null-environment**, 17  
**null?**, 13  
**number->string**, 13  
**number?**, 13  
**numerator**, 13  
**odd?**, 13  
**opaque**, 14  
**open-bytevector-input-port**, 14  
**open-bytevector-output-port**, 14  
**open-file-input-port**, 14  
**open-file-input/output-port**, 14  
**open-file-output-port**, 14  
**open-input-file**, 15  
**open-output-file**, 15  
**open-string-input-port**, 14  
**open-string-output-port**, 14  
**or**, 13  
**output-port-buffer-mode**, 14  
**output-port?**, 14  
**pair?**, 13  
**parent**, 14  
**parent-rtd**, 14  
**partition**, 13  
**peek-char**, 15  
**port-eof?**, 14  
**port-has-port-position?**, 14  
**port-has-set-port-position!?**, 14  
**port-position**, 14  
**port-transcoder**, 14  
**port?**, 14  
**positive?**, 13  
**procedure?**, 13  
**protocol**, 14  
**put-bytevector**, 14  
**put-char**, 14

<code>put-datum</code> , 14	<code>remp</code> , 13
<code>put-string</code> , 14	<code>remq</code> , 13
<code>put-u8</code> , 14	<code>remv</code> , 13
<code>quasiquote</code> , 13	<code>reverse</code> , 13
<code>quasisyntax</code> , 16	<code>rnrs/arithmetic/bitwise-6</code> , 15
<code>quote</code> , 13	<code>rnrs/arithmetic/fixnums-6</code> , 15
<code>quotient</code> , 17	<code>rnrs/arithmetic/flonums-6</code> , 15
<code>r6rs</code> , 8	<code>rnrs/base-6</code> , 13
<code>R<sup>6</sup>RS Conformance</code> , 11	<code>rnrs/bytectors-6</code> , 13
<code>R<sup>6</sup>RS Libraries</code> , 13	<code>rnrs/conditions-6</code> , 14
<code>R<sup>6</sup>RS Module Language</code> , 8	<code>rnrs/control-6</code> , 13
<code>R6RS: Scheme</code> , 1	<code>rnrs/enums-6</code> , 16
<code>raise</code> , 14	<code>rnrs/eval-6</code> , 16
<code>raise-continuable</code> , 14	<code>rnrs/exceptions-6</code> , 14
<code>rational-valued?</code> , 13	<code>rnrs/files-6</code> , 15
<code>rational?</code> , 13	<code>rnrs/hashtables-6</code> , 16
<code>rationalize</code> , 13	<code>rnrs/io/ports-6</code> , 14
<code>read</code> , 15	<code>rnrs/io/simple-6</code> , 15
<code>read-char</code> , 15	<code>rnrs/lists-6</code> , 13
<code>real-&gt;flonum</code> , 15	<code>rnrs/mutable-pairs-6</code> , 16
<code>real-part</code> , 13	<code>rnrs/mutable-strings-6</code> , 16
<code>real-valued?</code> , 13	<code>rnrs/programs-6</code> , 15
<code>real?</code> , 13	<code>rnrs/r5rs-6</code> , 17
<code>record-accessor</code> , 14	<code>rnrs/records/inspection-6</code> , 14
<code>record-constructor</code> , 14	<code>rnrs/records/procedural-6</code> , 14
<code>record-constructor-descriptor</code> , 14	<code>rnrs/records/syntactic-6</code> , 14
<code>record-field-mutable?</code> , 14	<code>rnrs/sorting-6</code> , 13
<code>record-mutator</code> , 14	<code>rnrs/syntax-case-6</code> , 16
<code>record-predicate</code> , 14	<code>rnrs/unicode-6</code> , 13
<code>record-rtd</code> , 14	<code>round</code> , 13
<code>record-type-descriptor</code> , 14	Running Top-Level Programs, 5
<code>record-type-descriptor?</code> , 14	<code>scheme-report-environment</code> , 17
<code>record-type-field-names</code> , 14	<code>sealed</code> , 14
<code>record-type-generative?</code> , 14	<code>serious-condition?</code> , 14
<code>record-type-name</code> , 14	<code>set!</code> , 13
<code>record-type-opaque?</code> , 14	<code>set-car!</code> , 16
<code>record-type-parent</code> , 14	<code>set-cdr!</code> , 16
<code>record-type-sealed?</code> , 14	<code>set-port-position!</code> , 14
<code>record-type-uid</code> , 14	<code>simple-conditions</code> , 14
<code>record?</code> , 14	<code>sin</code> , 13
<code>remainder</code> , 17	<code>sint-list-&gt;bytector</code> , 13
<code>remove</code> , 13	<code>sqrt</code> , 13

**standard-error-port**, 14  
**standard-input-port**, 14  
**standard-output-port**, 14  
**string**, 13  
**string->bytvector**, 14  
**string->list**, 13  
**string->number**, 13  
**string->symbol**, 13  
**string->utf16**, 13  
**string->utf32**, 13  
**string->utf8**, 13  
**string-append**, 13  
**string-ci-hash**, 16  
**string-ci=?**, 13  
**string-ci<?**, 13  
**string-ci=?**, 13  
**string-ci>=?**, 13  
**string-ci>?**, 13  
**string-copy**, 13  
**string-downcase**, 13  
**string-fill!**, 16  
**string-foldcase**, 13  
**string-for-each**, 13  
**string-hash**, 16  
**string-length**, 13  
**string-normalize-nfc**, 13  
**string-normalize-nfd**, 13  
**string-normalize-nfkc**, 13  
**string-normalize-nfkd**, 13  
**string-ref**, 13  
**string-set!**, 16  
**string-titlecase**, 13  
**string-upcase**, 13  
**string<=?**, 13  
**string<?**, 13  
**string=?**, 13  
**string>=?**, 13  
**string>?**, 13  
**string?**, 13  
**substring**, 13  
**symbol->string**, 13  
**symbol-hash**, 16  
**symbol=?**, 13  
**symbol?**, 13  
**syntax**, 16  
**syntax->datum**, 16  
**syntax-case**, 16  
**syntax-rules**, 13  
**syntax-violation**, 16  
**syntax-violation-form**, 14  
**syntax-violation-subform**, 14  
**syntax-violation?**, 14  
**tan**, 13  
**textual-port?**, 14  
The Implementation of R<sup>6</sup>RS, 8  
**transcoded-port**, 14  
**transcoder-codec**, 14  
**transcoder-eol-style**, 14  
**transcoder-error-handling-mode**, 14  
**truncate**, 13  
**u8-list->bytvector**, 13  
**uint-list->bytvector**, 13  
**undefined-violation?**, 14  
unless, 13  
unquote, 13  
unquote-splicing, 13  
unsyntax, 16  
unsyntax-splicing, 16  
Using R<sup>6</sup>RS, 8  
Using R<sup>6</sup>RS with DrRacket, 4  
**utf-16-codec**, 14  
**utf-8-codec**, 14  
**utf16->string**, 13  
**utf32->string**, 13  
**utf8->string**, 13  
values, 13  
vector, 13  
**vector->list**, 13  
**vector-fill!**, 13  
**vector-for-each**, 13  
**vector-length**, 13  
**vector-map**, 13  
**vector-ref**, 13  
**vector-set!**, 13

```
vector-sort, 13
vector-sort!, 13
vector?, 13
violation?, 14
warning?, 14
when, 13
who-condition?, 14
with-exception-handler, 14
with-input-from-file, 15
with-output-to-file, 15
with-syntax, 16
write, 15
write-char, 15
zero?, 13
```