S1 Manual All-Styles Document

Version 9.0.0.10

December 15, 2025

Contents

1 Code Styles

- `RktSym` (identifier without `for-label` binding): `unbound` or `example`

- `RktValLink` (link to variable form): `cons`

- `RktValDef` (definition of variable, normally combined with `RktValLink`): `list` in

  | `(list)` → `any/c`

  | `list` : `any/c`

- `RktStxLink` (link to syntactic form): `lambda`

- `RktStxDef` (definition of syntactic form, normally combined with `RktStxLink`): `lambda` in

  | `(lambda ...)`

- `RktSymDef` (definition without binding, normally a mistake, combined with `RktSym`): `unbound-identifier` in

  | `(unbound-identifier)`

- `RktVar` (local variable or meta-variable): *variable* or *example*

- `RktRes` (REPL result): `'(1 2 3)` or `example`

- `RktOut` (as written to the current output port): `example`

- `RktErr` (errors): *example* or the error message in

  ```
  > (+ 1 'a)
  +: contract violation
    expected: number?
    given: 'a
  ```

- `RktCmt` (comments): `example` or

  ```
  ; comment
  ```

- `RktVal` (values): `'(1 2 3)` or `example`

- `highlighted` (highlight via background): `(not-this example nor-this)`

- `RktIn` on a `RktInBG`: `example`

- `RktPn` (parentheses, etc.): `([{}])` or `example`

- RktRdr (reader shorthands): non-parentheses in `(#`() ,@())`

- RktMeta (the unquote comma): `,1` or `example` or "#reader" below.

  `#reader module`   package: `base`

- RktMod (module name; normally RktModLink instead): `example`

- RktModLink (a linked module reference): `racket/base`

- RktOpt (option-argument brackets): brackets in

  ```
  (f [x]) → any
    x : any/c = 1
  ```

- RktKw (not normally used): `example`

The `RktBlk` style class is used for a table of multiple lines (more than 1) of Racket code:

```
(define x (+ 1 2))
(+ x 3)
```

2 Definition Blocks

```
(require racket/base)        package: base
```

The module-declaration box above is in a `defmodule` table. The package-specification part is in an `RpackageSpec` wrapper.

The definitions below are marked so that they are not link targets. If they were link targets, the table-of-contents panel on the left would have entries for them.

```
(cons really-long-name-for-the-first-argument
      really-long-name-for-the-second-argument) → pair?
  really-long-name-for-the-first-argument : any/c
  really-long-name-for-the-second-argument : (or/c any/c
                                                   any/c)
```

This definition box starts with a `SVInsetFlow` wrapper, which is a `scribble/base` style class for the `'vertical-inset` style name on a block; it should give the block suitable vertical space before and after.

The next layer is a `boxed` plus `RBoxed` table. The `boxed` style class is from `scribble/base` and the `'boxed` style name on a table. The `RBoxed` style class is from the `scribble/manual` layer. Both `boxed` and `RBoxed` are used for all definition boxes by `scribble/manual` forms.

The initial content of the table includes a `SubFlow` (a `scribble/base` style class for non-indented flow) to combine blocks for the background label with the first line of the table. The background label "procedure" has an `RBackgroundLabel` outer wrapper, which makes the label float right. (The wrapper also has the `SIEHidden` style class, which built-in for all Scribble HTML output and makes the label hidden on Internet Explorer 6 and earlier.) The background label has an `RBackgroundLabelInner` inner wrapper, which makes the label suitably faint. The content part of the first line is wrapped in `RForeground`, which ensures that it is in front of the background label.

In a procedure definition box:

- When the initial "prototype" call in the definition box spans multiple lines, the table that contains the call has the `prototype` style class in addition to `RForeground`.

- When the contract or default value for an argument spans multiple lines, then the contract, the "=" for a value (if any), and value (if any) are wrapped in an table with the `argcontract` style class.

Finally, the definition box and all of the associated explanation text are wrapped in `SIntrapara` blocks and grouped into a single `<p>`.

```
(lambda ...)

example = good
        |  bad
```

When a syntactic-from specification has a grammar, the grammar is in a table with the spec-grammar style class.

Since no explanation flow is attached to the above defform use, there's no SIntrapara block around the table (just a <p>).

```
(cons a d) → pair?
  a : any/c
  d : any/c
(lambda ...)
```

Putting definitions together with deftogether converts the RBoxed and boxed tables that would be generated for the individual definitions into tables with the together style class. The tables are then combined as rows in a new table with the RBoxed and boxed style classes.

A defsubform, specsubform, etc., such as

```
(lambda ...)
```

is indented though a wrapper with a leftindent style class.

3 `bitmap-dc%`

`bitmap-dc%` : `class?`
  superclass: `object%`
  extends: `dc<%>`

In multi-page mode, this class definition gets its own page, and there's an "inherited methods" table in the margin. The table has style class `inherited`, and the words "inherited methods:" and "from" have style class `inheritedlbl`.

(`send` *a-bitmap-dc* `set-bitmap` *bm*) → `any`
  bm : `any/c`

A method example; nothing new here, but note how the defined identifier is not at the start of the box.

4 Miscellaneous

In `filebox` rendering,

```
"example.rkt"
```

This is a file box

a `Rfilebox` wrapper surrounds the file name in a `Rfiletitle` outer wrapper and an `Rfile-name` inner wrapper, plus the file content in an `Rfilecontent` wrapper.

> The `inset-flow` form generates a `nested-flow` with style class `insetpara`.

Changed in version 1.0: History paragraphs have the `SHistory` style class.

5 Bibliography

The bibliography table for the citation [Example] as the `RBibliography` style class.

Bibliography

[Example] "Example bibliography entry."