Unix Domain Sockets

Version 9.4.0.3

November 2, 2026

Ryan Culpepper <ryanc@racket-lang.org>

```
(require racket/unix-socket)
```
package: unix-socket-lib

This library provides support for *unix domain sockets* (specifically, sockets with family `AF_UNIX` and type `SOCK_STREAM`).

```
unix-socket-available? : boolean?
```

A boolean value that indicates whether unix domain sockets are available and supported on the current platform. The supported platforms are Linux, Mac OS X, and variants of BSD. This library does not currently support other Unix variants, and Windows does not have unix domain sockets.

```
(unix-socket-connect socket-path) → input-port? output-port?
  socket-path : unix-socket-path?
```

Connects to the unix domain socket associated with *socket-path* and returns an input port and output port for communicating with the socket. The connection is closed when both ports are closed.

```
(unix-socket-path? v) → boolean?
  v : any/c
```

Returns `#t` if *v* is a valid unix domain socket path for the current system. There are two kinds of socket paths: filesystem paths and abstract socket names.

- If *v* is a path (`path-string?`), the length of *v*'s corresponding absolute path must be less than or equal to the platform-specific length (108 bytes on Linux, 104 bytes on BSD and Mac OS X). Example: `"/tmp/mysocket"`.

- If *v* is a bytestring (bytes?), it represents an abstract socket name, supported only on Linux. The first byte of *v* must be NUL, and its length must be less than or equal to 108 bytes. Such a value refers to a socket in the Linux abstract socket namespace. Example: #"\0mysocket".

  Note that NUL bytes are allowed in abstract socket names. For example, #"\0abc" and #"\0abc\0" are both valid—and different—abstract socket names.

Otherwise, returns #f.

```
(unix-socket-listen socket-path [backlog]) → unix-socket-listener?
  socket-path : unix-socket-path?
  backlog : exact-nonnegative-integer? = 4
```

Listen for connections on a unix domain socket bound to *socket-path*, returning a listener that can be used to accept incoming connections.

If *socket-path* refers to a filesystem path, binding the socket creates a file that must be deleted separately from closing the listener.

```
(unix-socket-listener? v) → boolean?
  v : any/c
```

Returns #t if *v* is a unix socket listener created with unix-socket-listen; #f otherwise.

A unix socket listener acts as a synchronizable event. It is ready when a client connection is ready to be accepted (see unix-socket-accept), and its synchronization result is the listener itself.

```
(unix-socket-close-listener listener) → void?
  listener : unix-socket-listener?
```

Closes a unix socket listener. The socket file must be deleted separately (e.g. using delete-file.)

```
(unix-socket-accept listener) → input-port? output-port?
  listener : unix-socket-listener?
```

Accepts a client connection for *listener*. If no client connection is waiting to be accepted, the call to unix-socket-accept will block.

```
(unix-socket-accept-evt listener) → evt?
  listener : unix-socket-listener?
```

Returns a synchronizable event that is ready for synchronization when unix-socket-accept on *listener* would not block. The synchronization result is a list containing two

items: an input port and an output port. The ports are managed by the custodian that is the current custodian at the time that `unix-socket-accept-evt` is called.

Added in version 1.2 of package `unix-socket-lib`.

1 TCP Unit

```
(require racket/unix-socket-tcp-unit)
```
package: unix-socket-lib

This module provides an implementation of the `tcp^` signature that can be used to make the Web Server listen on a unix domain socket.

```
(make-unix-socket-tcp@ socket-path) → unit?
  socket-path : unix-socket-path?
```

Returns a unit suitable for use with the Web Server's `serve` function.

For example, the following code would instruct the web server to listen for connections via a unix socket at `"/var/server.sock"`:

```
(serve
 #:tcp@ (make-unix-socket-tcp@ "/var/server.sock")
 #:dispatch a-dispatcher)
```

Added in version 1.3 of package `unix-socket-lib`.