Erlangen API Documentation

Max Rottenkolber

Monday, 20 November 2017

Table of Contents

1 erlangen (Package)

Distributed, asychronous message passing system for Clozure Common Lisp.

1.1 *agent-debug* (Variable)

Initial Value:

`NIL`

Description:

If `*agent-debug*` is *true* when calling `spawn`, *conditions* of *type* `serious-condition` will not be automatically handled for the spawned *agent*. The debugger will be entered so that the call stack can be inspected. Invoking the `exit` *restart* will resume normal operation except that the exit reason will be the *agent* instead of the fatal *condition*.

When an *agent* is started it *binds* this variable to its value in the environment where it was spawned, effectively inheriting the *binding*.

Affected By:

`spawn`

1.2 *default-mailbox-size* (Variable)

Initial Value:

`64`

Description:

`*default-mailbox-size*` is the default value of the `:mailbox-size` parameter to `spawn`.

When an *agent* is started it *binds* this variable to its value in the environment where it was spawned, effectively inheriting the *binding*.

Affected By:

`spawn`

1.3 agent (Function)

Syntax:

— Function: **agent** <*no arguments*>

Description:

`agent` returns the *calling agent*.

1.4 agent (Type)

Syntax:

agent::= *structure* | *keyword* | *string*

Description:

An *agent* can either be an *agent structure*, a *keyword* denoting a registered *agent* or a *string* denoting a *remote agent*.

A *remote agent* is denoted by a *string* of the form "*host/node/agent*" where *host* is the host name, *node* is the *node name* and *agent* is the *agent identifier* of the *remote agent*.

An *agent identifier* is either a hexadecimal digit string denoting an *anonymous agent* or a colon followed by a *symbol name* denoting a *registered agent*. In the latter case, the *symbol name* may not contain the slash (/) character.

Notes:

Only *agent structures* are of *type* `agent`.

1.5 call (Type)

Syntax:

call::= (*function argument**)

Arguments and Values:

function—a *symbol* denoting a *function*.

argument—a *serializable object*.

Description:

A *call* denotes a portable function call to be invoked on a given *node*. A *call* is a *list* whose first element is a *symbol* denoting a *function* and whose remaining elements are arguments to be applied to the denoted *function*.

1.6 exit (Function)

Syntax:

— Function: **exit** `&optional` *reason agent*

Arguments and Values:

reason—an *object*. The default is `:kill`.

agent—an *agent*. The default is the *calling agent*.

Description:

`exit` kills *agent* with *reason* as the *exit reason*. Subsequent attempts to send messages to *agent* will fail. If *agent* is the *calling agent* it exits immediately, otherwise `exit` delivers an *exit message* to *agent*.

Exceptional Situations:

If *agent* is a *keyword* that is not registered as a name an *error* of *type* `simple-error` is signaled.

1.7 link (Function)

Syntax:

— Function: **link** *agent* `&optional` *mode*

Arguments and Values:

agent—an *agent*.

mode—either `:link` or `:monitor`. The default is `:link`.

Description:

`link` *links* the *calling agent* to *agent*. After two *agents* are *linked* they behave as follows:

When the *calling agent* exits, an *exit message* with the *exit reason* of the *calling agent* is delivered to the *linked agent*.

When the linked *agent* exits, and *mode* is `:link`, an *exit message* with the *exit reason* of the *linked agent* is delivered to the *calling agent*.

When the *linked agent* exits, and *mode* is `:monitor`, an *exit notification* is delivered to the *calling agent*.

An *exit notification* is of the form

`(` *agent status* . *values* `)`

status::= `:ok` | `:exit`

The *status* `:ok` indicates that the *agent* exited normally, and *values* will be a list of its *return values*.

The *status* `:exit` indicates that the *agent* was either killed by `exit` or aborted because of an unhandled *condition* of *type* `serious-condition`, and *values* will be the *exit reason* supplied to `exit`, or the *condition object*.

Exceptional Situations:

If *agent* is the *calling agent* an *error* of *type* `simple-error` is signaled.

If *agent* is a *keyword* that is not registered as a name an *error* of *type* `simple-error` is signaled.

1.8 node (Function)

Syntax:

— Function: **node** `&key` *host name*

Arguments and Values:

host—a *host* as accepted by *resolve-address* (http://ccl.clozure.com/docs/ccl.html#f_resolve-address). The default is the local host name as reported by `machine-instance`.

name—a *string*. The default is a unique name.

Description:

`node` spawns the node protocol server to listen on a random free port of *host*. It then registers its *name* and listening port with the port mapper.

Once the node is registered, it is capable of communicating with remote nodes.

Examples:

```
;; Start talking to remote nodes:
(spawn 'node)
```

Side Effects:

`node` changes the host name of the local node to *host*, which is subsequently used as a default argument to `spawn`.

Exceptional Situations:

If *name* can not be registered (e.g., because it has already been registered by another node, or because the port mapper is unreachable) an *error* of *type* `error` is signaled, and the node protocol server is killed.

1.9 receive (Function)

Syntax:

— Function: **receive** `&key` *timeout*

Arguments and Values:

timeout—a non-negative *real* denoting a time interval in seconds.

Description:

`receive` returns the next message for the *calling agent*. If the message is an *exit message* the *calling agent* exits immediately. If the *mailbox* of the *calling agent* is empty, `receive` will block until a message arrives.

If *timeout* is supplied `receive` will block for at most *timeout* seconds.

Exceptional Situations:

If *timeout* is supplied and the specified time interval exceeded an *error* of *type* `timeout` is signaled.

1.10 register (Function)

Syntax:

— Function: **register** *name* `&optional` *agent*

Arguments and Values:

name—a *keyword*.

agent—an *agent*. Default is the *calling agent*.

Description:

`register` associates *name* with *agent*.

Exceptional Situations:

If *name* is already associated with an *agent* an *error* of *type* `simple-error` is signaled.

1.11 registered (Function)

Syntax:

— Function: **registered** <*no arguments*>

Description:

`registered` returns a *list* of names associated with *agents*.

1.12 select (Macro)

Syntax:

— Macro: **select** `&rest` *clauses*

clauses::= *normal-clause** [*receive-clause*]

normal-clause::= (*poll-form vars body-form**)

receive-clause::= (`:receive` *vars body-form**)

Arguments and Values:

poll-form, *body-form—forms*.

vars—a *list* of *symbols*.

Description:

`select` repeatedly calls the *poll-forms* of each *normal-clause* (in order) until a *poll-form* returns a non-nil value as its first result and *vars* is non-nil. It then evaluates each *body-form* of the respective *normal-clause* with the return values of its *poll-forms* bound to *vars* and returns their result.

If a *receive-clause* is supplied and its *vars* are non-nil, `select` will evaluate each *body-form* of the clause with the received message bound to the first *symbol* in *vars* and return their result. If no *receive-clause* is supplied, `select` will silently discard incoming messages.

1.13 send (Function)

Syntax:

— Function: **send** *message agent*

Arguments and Values:

message—an *object*.

agent—an *agent*.

Description:

`send` transmits *message* to *agent*. There is no guarantee as to whether *message* could be successfully delivered.

Exceptional Situations:

If *agent* is a *keyword* that is not registered as a name an *error* of *type* `simple-error` is signaled.

1.14 spawn (Function)

Syntax:

— Function: **spawn** *function* `&key` *attach mailbox-size node host*

Arguments and Values:

function—a *function designator* or a *call*.

attach—either `:link`, `:monitor`, or `nil`. The default is `nil`.

mailbox-size—a positive *unsigned integer*. The default is `*default-mailbox-size*`.

node—a *node name* or `nil`. The default is `nil`.

host—a *host* as accepted by *resolve-address* (http://ccl.clozure.com/docs/ccl.html#f_resolve-address). The default is the host name of the local node.

Description:

`spawn` starts and returns a new *agent* with a mailbox capacity of *mailbox-size*. If *attach* is `:link` or `:monitor` the *calling agent* will be linked to the new *agent* as if by `link` but before the *agent* is started. Once the *agent* is started it will execute *function*.

If *node* is *non-nil* the *agent* is started on *node* of *host* instead of the local node.

Affected By:

`node`

Exceptional Situations:

If `spawn` fails to start the *agent* an an *error* of *type* `error` is signaled.

1.15 timeout (Condition Type)

Class Precedence List:

`timeout`, `error`, `serious-condition`, `condition`, `standard-object`, `t`

Description:

Describes an error condition that can occur when using functions with a timeout. It denotes a that the operation was unable to successfully complete within a given duration.

1.16 unlink (Function)

Syntax:

— Function: **unlink** *agent*

Arguments and Values:

agent—an *agent*.

Description:

`unlink` removes any *link* between *agent* and the *calling agent*.

Exceptional Situations:

If *agent* is the *calling agent* an *error* of *type* `simple-error` is signaled.

If *agent* is a *keyword* that is not registered as a name an *error* of *type* `simple-error` is signaled.

1.17 unregister (Function)

Syntax:

— Function: **unregister** *name*

Arguments and Values:

name—a *keyword*.

Description:

`unregister` removes the registered *name*, associated with an *agent*.

Exceptional Situations:

If the *name* is not associated with an *agent* an *error* of *type* `simple-error` is signaled.

2 erlangen.management (Package)

Management extensions for Erlangen including functions for agent tree introspection, and retrieval of statistics for agents and remote connections.

2.1 agent-stats (Function)

Syntax:

— Function: **agent-stats** *agent*

→ *messages-received*, *messages-dropped*, *birthtime*, *deathtime*

Arguments and Values:

agent—an *agent*.

messages-received—a non-negative *integer* denoting the number of messages received by *agent*.

messages-dropped—a non-negative *integer* denoting the number of messages dropped by *agent* because its mailbox was full.

birthtime—a *universal time* denoting the time when *agent* was started.

deathtime—a *universal time* denoting the time when *agent* exited, or `nil` if *agent* has not exited.

Description:

`agent-stats` returns various current statistics for *agent*.

2.2 agent-tree (Function)

Syntax:

— Function: **agent-tree** *agent*

→ *agent-tree*

Arguments and Values:

agent—an *agent*.

agent-tree—an *instance* of *class* `agent-tree`.

Description:

`agent-tree` returns the current *agent-tree* whose root is *agent*.

2.3 agent-tree (Class)

Syntax:

— Class: **agent-tree** `&key` *root linked monitored*

Class Precedence List:

`agent-tree`, `standard-object`, `t`

Description:

Instances of class `agent-tree` denote views of the *agent* graph at a specific point in time. Their `print-object` method prints an elaborate description of that view when `*print-readably*` is `nil`.

2.4 connection-stats (Function)

Syntax:

— Function: **connection-stats** `&optional` *host node*

→ *stats-for-connections*

→ *errors*, *established*

Arguments and Values:

host—a *string* denoting a *host name* or `nil`. The default is `nil`.

node—a *string* denoting a *node name* or `nil`. The default is `nil`.

stats-for-connections—a *list* with one element for each matching connection. Each element is a *list* of four elements containing the host name, node name, *errors*, and *established* of the respective connection.

errors—a non-negative *integer* denoting the number of errors on the matching connection.

established—a *universal time* denoting the time when the matching connection was initially established, or `nil` denoting that the connection has not been established yet.

Description:

`connection-stats` returns essential statistics for connections to remote nodes. If called without arguments it returns *stats-for-connections*

to all remote nodes to which connections where established. If *host* is supplied *stats-for-connections* includes only connections to *host*. If *node* is supplied `connection-stats` returns *errors* and *established* for the connection to *node*.

2.5 flush-messages (Function)

Syntax:

— Function: **flush-messages** `&key` *print-p stream*

Arguments and Values:

print-p—a *generalized boolean*. The default is *true*.

stream—an *output stream*. The default is `*standard-output*`.

Description:

`flush-messages` dequeues messages from the mailbox of the *calling agent* until there are no more pending messages. If *print-p* is *true* each dequeued message is printed to *stream*.

2.6 linked (Generic Function)

Syntax:

— Generic Function: **linked** *agent-tree*

→ *list*

Arguments and Values:

agent-tree—an *object* of *type* `agent-tree`.

list—a *list* of *agents*.

Description:

`linked` returns a *list* of *agents* that are linked to but not monitored by the root *agent* of *agent-tree*.

2.7 monitored (Generic Function)

Syntax:

— Generic Function: **monitored** *agent-tree*

→ *subtrees*

Arguments and Values:

agent-tree—an *object* of *type* `agent-tree`.

subtrees—a *list* of *objects* of *type* `agent-tree`.

Description:

`monitored` returns a *list* of the *subtrees* whose root *agents* are monitored by the root *agent* of *agent-tree*.

2.8 process-agent (Function)

Syntax:

— Function: **process-agent** *process* `&key` *timeout*

→ *agent*

Arguments and Values:

process—a *process*.

timeout—a non-negative *real* denoting a time interval in seconds. The default is 1.

agent—an *agent* or `nil`.

Description:

`process-agent` interrupts *process* to retrieve its associated *agent*. It returns the respective *agent* or `nil`. A return value of `nil` indicates that *process* could not be interrupted within the duration specified by *timeout*.

2.9 root (Generic Function)

Syntax:

— Generic Function: **root** *agent-tree*

→ *agent*

Arguments and Values:

agent-tree—an *object* of *type* `agent-tree`.

agent—an *agent*.

Description:

`root` returns the *agent* that is the root of *agent-tree*.