Class
Process::Status < Object

Process::Status encapsulates the information on the status of a running or terminated system process. The built-in variable $? is either nil or a Process::Status object.

```
fork { exit 99 }   # =>   84972
Process.wait       # =>   84972
$?.class           # =>   Process::Status
$?.to_i            # =>   25344
$? >> 8            # =>   99
$?.stopped?        # =>   false
$?.exited?         # =>   true
$?.exitstatus      # =>   99
```

POSIX systems record information on processes using a 16-bit integer. The lower bits record the process status (stopped, exited, signaled), and the upper bits possibly contain additional information (for example, the program's return code in the case of exited processes). Before Ruby 1.8, these bits were exposed directly to the Ruby program. Ruby now encapsulates these in a Process::Status object. To maximize compatibility, however, these objects retain a bit-oriented interface. In the descriptions that follow, when we talk about the integer value of *stat*, we're referring to this 16-bit value.

Instance methods

==

stat == *other* → true or false

Returns true if the integer value of *stat* equals *other*.

&

stat & *num* → *fixnum*

Logical AND of the bits in *stat* with *num*.

```
fork { exit 0x37 }
Process.wait
sprintf('%04x', $?.to_i)       # =>   "3700"
sprintf('%04x', $? & 0x1e00)   # =>   "1600"
```

>>

stat >> *num* → *fixnum*

Shifts the bits in *stat* right *num* places.

```
fork { exit 99 }   # =>   84978
Process.wait       # =>   84978
$?.to_i            # =>   25344
$? >> 8            # =>   99
```

coredump?

stat.coredump → true or false

Returns true if *stat* generated a coredump when it terminated. Not available on all platforms.

exited?

stat.exited? → true or false

Returns true if *stat* exited normally (for example using an exit call or finishing the program).

exitstatus

stat.exitstatus → *fixnum* or nil

Returns the least significant 8 bits of the return code of *stat*. Available only if exited? is true.

```
fork { }           # =>   84981
Process.wait       # =>   84981
$?.exited?         # =>   true
$?.exitstatus      # =>   0

fork { exit 99 }   # =>   84982
Process.wait       # =>   84982
$?.exited?         # =>   true
$?.exitstatus      # =>   99
```

pid

stat.pid → *fixnum*

Returns the ID of the process associated with this status object.

```
fork { exit }   # =>   84985
Process.wait    # =>   84985
$?.pid          # =>   84985
```

signaled?

stat.signaled? → true or false

Returns true if *stat* terminated because of an uncaught signal.

```
pid = fork { sleep 100 }
Process.kill(9, pid)   # =>   1
Process.wait           # =>   84988
$?.signaled?           # =>   true
```

stopped?

stat.stopped? → true or false

Returns true if this process is stopped. This is returned only if the corresponding wait call had the WUNTRACED flag set.

success?

stat.success? → nil, or true or false

Returns true if *stat* refers to a process that exited successfully, returns false if it exited with a failure, and returns nil if *stat* does not refer to a process that has exited.

stopsig

stat.stopsig → *fixnum* or nil

Returns the number of the signal that caused *stat* to stop (or nil if self is not stopped).

termsig

stat.termsig → *fixnum* or nil

Returns the number of the signal that caused *stat* to terminate (or nil if self was not terminated by an uncaught signal).

to_i

stat.to_i → *fixnum*

Returns the bits in *stat* as a Fixnum. Poking around in these bits is platform dependent.

```
fork { exit 0xab }         # =>   84991
Process.wait               # =>   84991
sprintf('%04x', $?.to_i)   # =>   "ab00"
```

to_s

stat.to_s → *string*

Equivalent to *stat*.to_i.to_s.