NAME

"IO::Async::Timer::Absolute" – event callback at a fixed future time

SYNOPSIS

```
use IO::Async::Timer::Absolute;

use POSIX qw( mktime );

use IO::Async::Loop;
my $loop = IO::Async::Loop->new;

my @time = gmtime;

my $timer = IO::Async::Timer::Absolute->new(
   time => mktime( 0, 0, 0, $time[3]+1, $time[4], $time[5] ),

   on_expire => sub {
      print "It's midnight\n";
      $loop->stop;
   },
);

$loop->add( $timer );

$loop->run;
```

DESCRIPTION

This subclass of IO::Async::Timer implements one-shot events at a fixed time in the future. The object waits for a given timestamp, and invokes its callback at that point in the future.

For a `Timer` object that waits for a delay relative to the time it is started, see instead IO::Async::Timer::Countdown.

EVENTS

The following events are invoked, either using subclass methods or CODE references in parameters:

on_expire

Invoked when the timer expires.

PARAMETERS

The following named parameters may be passed to `new` or `configure`:

on_expire => CODE

CODE reference for the `on_expire` event.

time => NUM

The epoch time at which the timer will expire.

Once constructed, the timer object will need to be added to the `Loop` before it will work.

Unlike other timers, it does not make sense to `start` this object, because its expiry time is absolute, and not relative to the time it is started.

AUTHOR

Paul Evans <leonerd@leonerd.org.uk>