NAME

"IO::Async::Loop::Select" – use IO::Async with "select(2)"

SYNOPSIS

Normally an instance of this class would not be directly constructed by a program. It may however, be useful for runinng IO::Async with an existing program already using a `select` call.

```
use IO::Async::Loop::Select;

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

$loop->add( ... );

while(1) {
   my ( $rvec, $wvec, $evec ) = ('') x 3;
   my $timeout;

   $loop->pre_select( \$rvec, \$wvec, \$evec, \$timeout );
   ...
   my $ret = select( $rvec, $wvec, $evec, $timeout );
   ...
   $loop->post_select( $rvec, $evec, $wvec );
}
```

DESCRIPTION

This subclass of IO::Async::Loop uses the `select(2)` syscall to perform read-ready and write-ready tests.

To integrate with an existing `select`–based event loop, a pair of methods `pre_select` and `post_select` can be called immediately before and after a `select` call. The relevant bits in the read-ready, write-ready and exceptional-state bitvectors are set by the `pre_select` method, and tested by the `post_select` method to pick which event callbacks to invoke.

CONSTRUCTOR

new

```
$loop = IO::Async::Loop::Select->new
```

This function returns a new instance of a `IO::Async::Loop::Select` object. It takes no special arguments.

METHODS

pre_select

```
$loop->pre_select( \$readvec, \$writevec, \$exceptvec, \$timeout )
```

This method prepares the bitvectors for a `select` call, setting the bits that the Loop is interested in. It will also adjust the `$timeout` value if appropriate, reducing it if the next event timeout the Loop requires is sooner than the current value.

\$readvec
\$writevec
\$exceptvec
: Scalar references to the reading, writing and exception bitvectors

\$timeout
: Scalar reference to the timeout value

post_select

```
$loop->post_select( $readvec, $writevec, $exceptvec )
```

This method checks the returned bitvectors from a `select` call, and calls any of the callbacks that are appropriate.

`$readvec`
`$writevec`
`$exceptvec`

Scalars containing the read-ready, write-ready and exception bitvectors

loop_once

```
$count = $loop->loop_once( $timeout )
```

This method calls the `pre_select` method to prepare the bitvectors for a `select` syscall, performs it, then calls `post_select` to process the result. It returns the total number of callbacks invoked by the `post_select` method, or `undef` if the underlying `select(2)` syscall returned an error.

SEE ALSO

- IO::Select – OO interface to select system call

AUTHOR

Paul Evans <leonerd@leonerd.org.uk>