NAME

TAP::Parser::IteratorFactory - Internal TAP::Parser Iterator

VERSION

Version 3.17

SYNOPSIS

```
use TAP::Parser::IteratorFactory;
my $factory = TAP::Parser::IteratorFactory->new;
my $iter = $factory->make_iterator(\*TEST);
my $iter = $factory->make_iterator(\@array);
my $iter = $factory->make_iterator(\%hash);

my $line = $iter->next;
```

DESCRIPTION

This is a factory class for simple iterator wrappers for arrays, filehandles, and hashes. Unless you're subclassing, you probably won't need to use this module directly.

METHODS

Class Methods

new

Creates a new factory class. *Note:* You currently don't need to instantiate a factory in order to use it.

make_iterator

Create an iterator. The type of iterator created depends on the arguments to the constructor:

```
my $iter = TAP::Parser::Iterator->make_iterator( $filehandle );
```

Creates a *stream* iterator (see *make_stream_iterator*).

```
my $iter = TAP::Parser::Iterator->make_iterator( $array_reference );
```

Creates an *array* iterator (see *make_array_iterator*).

```
my $iter = TAP::Parser::Iterator->make_iterator( $hash_reference );
```

Creates a *process* iterator (see *make_process_iterator*).

make_stream_iterator

Make a new stream iterator and return it. Passes through any arguments given. Defaults to a *TAP::Parser::Iterator::Stream*.

make_array_iterator

Make a new array iterator and return it. Passes through any arguments given. Defaults to a *TAP::Parser::Iterator::Array*.

make_process_iterator

Make a new process iterator and return it. Passes through any arguments given. Defaults to a *TAP::Parser::Iterator::Process*.

SUBCLASSING

Please see *"SUBCLASSING" in TAP::Parser* for a subclassing overview.

There are a few things to bear in mind when creating your own `ResultFactory`:

1 The factory itself is never instantiated (this *may* change in the future). This means that `_initialize` is never called.

Example

```
package MyIteratorFactory;

use strict;
use vars '@ISA';

use MyStreamIterator;
use TAP::Parser::IteratorFactory;

@ISA = qw( TAP::Parser::IteratorFactory );

# override stream iterator
sub make_stream_iterator {
  my $proto = shift;
  MyStreamIterator->new(@_);
}

1;
```

ATTRIBUTION

Originally ripped off from *Test::Harness*.

SEE ALSO

TAP::Object, *TAP::Parser*, *TAP::Parser::Iterator*, *TAP::Parser::Iterator::Array*, *TAP::Parser::Iterator::Stream*, *TAP::Parser::Iterator::Process*,