NAME

TAP::Parser::SourceHandler::File - Stream TAP from a text file.

VERSION

Version 3.36

SYNOPSIS

```
use TAP::Parser::Source;
use TAP::Parser::SourceHandler::File;

my $source = TAP::Parser::Source->new->raw( \'file.tap' );
$source->assemble_meta;

my $class = 'TAP::Parser::SourceHandler::File';
my $vote = $class->can_handle( $source );
my $iter = $class->make_iterator( $source );
```

DESCRIPTION

This is a *raw TAP stored in a file* TAP::Parser::SourceHandler - it has 2 jobs:

1. Figure out if the *raw* source it's given is a file containing raw TAP output. See TAP::Parser::IteratorFactory for more details.

2. Takes raw TAP from the text file given, and converts into an iterator.

Unless you're writing a plugin or subclassing TAP::Parser, you probably won't need to use this module directly.

METHODS

Class Methods

can_handle

```
my $vote = $class->can_handle( $source );
```

Only votes if `$source` looks like a regular file. Casts the following votes:

```
0.9 if it's a .tap file
0.9 if it has an extension matching any given in user config.
```

make_iterator

```
my $iterator = $class->make_iterator( $source );
```

Returns a new TAP::Parser::Iterator::Stream for the source. `croak`s on error.

iterator_class

The class of iterator to use, override if you're sub-classing. Defaults to TAP::Parser::Iterator::Stream.

CONFIGURATION

```
{
extensions => [ @case_insensitive_exts_to_match ]
}
```

SUBCLASSING

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

SEE ALSO

TAP::Object, TAP::Parser, TAP::Parser::SourceHandler, TAP::Parser::SourceHandler::Executable, TAP::Parser::SourceHandler::Perl, TAP::Parser::SourceHandler::Handle, TAP::Parser::SourceHandler::RawTAP