



**Full credit is given to the above companies including the Operating System (OS) that this PDF file was generated!**

## ***Rocky Enterprise Linux 9.2 Manual Pages on command 'TAP::Parser::SourceHandler::Perl.3perl'***

***\$ man TAP::Parser::SourceHandler::Perl.3perl***

TAP::Parser::SourceHandler::Perl(PerlProgrammers ReferencTAP::Parser::SourceHandler::Perl(3perl)

### NAME

TAP::Parser::SourceHandler::Perl - Stream TAP from a Perl executable

### VERSION

Version 3.43

### SYNOPSIS

```
use TAP::Parser::Source;  
  
use TAP::Parser::SourceHandler::Perl;  
  
my $source = TAP::Parser::Source->new->raw( 'script.pl' );  
  
$source->assemble_meta;  
  
my $class = 'TAP::Parser::SourceHandler::Perl';  
  
my $vote = $class->can_handle( $source );  
  
my $iter = $class->make_iterator( $source );
```

### DESCRIPTION

This is a Perl TAP::Parser::SourceHandler - it has 2 jobs:

1. Figure out if the TAP::Parser::Source it's given is actually a Perl script ("can\_handle").
2. Creates an iterator for Perl sources ("make\_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 file. Casts the following votes:

0.9 if it has a shebang ala "#!...perl"

0.75 if it has any shebang

0.8 if it's a .t file

0.9 if it's a .pl file

0.75 if it's in a 't' directory

0.25 by default (backwards compat)

"make\_iterator"

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

Constructs & returns a new TAP::Parser::Iterator::Process for the source. Assumes

"\$source->raw" contains a reference to the perl script. "croak"s if the file could not be found.

The command to run is built as follows:

```
$perl @switches $perl_script @test_args
```

The perl command to use is determined by "get\_perl". The command generated is guaranteed to preserve:

PERL5LIB

PERL5OPT

Taint Mode, if set in the script's shebang

Note: the command generated will not respect any shebang line defined in your Perl script.

This is only a problem if you have compiled a custom version of Perl or if you want to use a specific version of Perl for one test and a different version for another, for example:

```
#!/path/to/a/custom_perl --some --args
```

```
#!/usr/local/perl-5.6/bin/perl -w
```

Currently you need to write a plugin to get around this.

"get\_taint"

Decode any taint switches from a Perl shebang line.

```
# $taint will be 't'
```

```
my $taint = TAP::Parser::SourceHandler::Perl->get_taint( '#!/usr/bin/perl -t' );
```

```
# $untaint will be undefined
```

```
my $untaint = TAP::Parser::SourceHandler::Perl->get_taint( '#!/usr/bin/perl' );
```

"get\_perl"

Gets the version of Perl currently running the test suite.

## SUBCLASSING

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

### Example

```
package MyPerlSourceHandler;

use strict;

use TAP::Parser::SourceHandler::Perl;

use base 'TAP::Parser::SourceHandler::Perl';

# use the version of perl from the shebang line in the test file

sub get_perl {

    my $self = shift;

    if (my $shebang = $self->shebang( $self->{file} )) {

        $shebang =~ /^#!(.*\bperl.*?)(?:\s|(?:$))/;

        return $1 if $1;

    }

    return $self->SUPER::get_perl(@_);

}
```

## SEE ALSO

TAP::Object, TAP::Parser, TAP::Parser::IteratorFactory, TAP::Parser::SourceHandler,  
TAP::Parser::SourceHandler::Executable, TAP::Parser::SourceHandler::File,  
TAP::Parser::SourceHandler::Handle, TAP::Parser::SourceHandler::RawTAP

perl v5.34.0

2023-11-23 TAP::Parser::SourceHandler::Perl(3perl)