Rocky Enterprise Linux 9.2 Manual Pages on command 'XML::SAX.3pm'

C:\>man XML::SAX.3pm

XML::SAX(3pm) User Contributed Perl Documentation XML::SAX(3pm)

NAME

XML::SAX - Simple API for XML

SYNOPSIS

```
use XML::SAX;
# get a list of known parsers
my $parsers = XML::SAX->parsers();
# add/update a parser
XML::SAX->add_parser(q(XML::SAX::PurePerl));
# remove parser
XML::SAX->remove_parser(q(XML::SAX::Foodelberry));
# save parsers
XML::SAX->save_parsers();
```

DESCRIPTION

XML::SAX is a SAX parser access API for Perl. It includes classes and APIs required for implementing SAX drivers, along with a factory class for returning any SAX parser installed on the user's system.

USING A SAX2 PARSER

The factory class is XML::SAX::ParserFactory. Please see the documentation of that module for how to instantiate a SAX parser: XML::SAX::ParserFactory. However if you don't want to load up another manual page, here's a short synopsis:

```
use XML::SAX::ParserFactory;
```

```
use XML::SAX::XYZHandler;
my $handler = XML::SAX::XYZHandler->new();
my $p = XML::SAX::ParserFactory->parser(Handler => $handler);
$p->parse_uri("foo.xml");
# or $p->parse_string("<foo/>") or $p->parse_file($fh);
```

This will automatically load a SAX2 parser (defaulting to XML::SAX::PurePerl if no others are found) and return it to you.

In order to learn how to use SAX to parse XML, you will need to read XML::SAX::Intro and for reference, XML::SAX::Specification.

WRITING A SAX2 PARSER

The first thing to remember in writing a SAX2 parser is to subclass XML::SAX::Base. This will make your life infinitely easier, by providing a number of methods automagically for you. See XML::SAX::Base for more details.

When writing a SAX2 parser that is compatible with XML::SAX, you need to inform XML::SAX of the presence of that driver when you install it. In order to do that, XML::SAX contains methods for saving the fact that the parser exists on your system to a "INI" file, which is then loaded to determine which parsers are installed.

The best way to do this is to follow these rules:

? Add XML::SAX as a prerequisite in Makefile.PL:

```
WriteMakefile(
    ...
    PREREQ_PM => { 'XML::SAX' => 0 },
    ...
);
```

Alternatively you may wish to check for it in other ways that will cause more than just a warning.

? Add the following code snippet to your Makefile.PL:

```
sub MY::install {
  package MY;
  my $script = shift->SUPER::install(@_);
  if (ExtUtils::MakeMaker::prompt(
    "Do you want to modify ParserDetails.ini?", 'Y')
    =~ /^y/i) {
```

```
    $script =~ s/install :: (.*)$/install :: $1 install_sax_driver/m;
    $script .= <<"INSTALL";
install_sax_driver :
\t\@\$(PERL) -MXML::SAX -e "XML::SAX->add_parser(q(\$(NAME)))->save_parsers()"
INSTALL
  }
  return $script;
}
```

Note that you should check the output of this - \$(NAME) will use the name of your distribution, which may not be exactly what you want. For example XML::LibXML has a driver called XML::LibXML::SAX::Generator, which is used in place of \$(NAME) in the above.

? Add an XML::SAX test:

A test file should be added to your t/ directory containing something like the following:

```
use Test;
BEGIN { plan tests => 3 }
use XML::SAX;
use XML::SAX::PurePerl::DebugHandler;
XML::SAX->add_parser(q(XML::SAX::MyDriver));
local $XML::SAX::ParserPackage = 'XML::SAX::MyDriver';
eval {
  my $handler = XML::SAX::PurePerl::DebugHandler->new();
  ok($handler);
  my $parser = XML::SAX::ParserFactory->parser(Handler => $handler);
  ok($parser);
  ok($parser->isa('XML::SAX::MyDriver');
  $parser->parse_string("<tag/>");
  ok($handler->{seen}{start_element});
};
```

EXPORTS

By default, XML::SAX exports nothing into the caller's namespace. However you can request the symbols "Namespaces" and "Validation" which are the URIs for those

features, allowing an easier way to request those features via ParserFactory:

```
use XML::SAX qw(Namespaces Validation);
my $factory = XML::SAX::ParserFactory->new();
$factory->require_feature(Namespaces);
$factory->require_feature(Validation);
my $parser = $factory->parser();
```

AUTHOR

Current maintainer: Grant McLean, grantm@cpan.org

Originally written by:

Matt Sergeant, matt@sergeant.org

Kip Hampton, khampton@totalcinema.com

Robin Berjon, robin@knowscape.com

LICENSE

SEE ALSO

XML::SAX::Base for writing SAX Filters and Parsers

XML::SAX::PurePerl for an XML parser written in 100% pure perl.

XML::SAX::Exception for details on exception handling