NAME

XML::LibXML::Error – Structured Errors

SYNOPSIS

```
eval { ... };
        if (ref($@)) {
          # handle a structured error (XML::LibXML::Error object)
        } elsif ($@) {
          # error, but not an XML::LibXML::Error object
        } else {
          # no error
        }

$XML::LibXML::Error::WARNINGS=1;
$message = $@->as_string();
print $@->dump();
$error_domain = $@->domain();
$error_code = $@->code();
$error_message = $@->message();
$error_level = $@->level();
$filename = $@->file();
$line = $@->line();
$nodename = $@->nodename();
$error_str1 = $@->str1();
$error_str2 = $@->str2();
$error_str3 = $@->str3();
$error_num1 = $@->num1();
$error_num2 = $@->num2();
$string = $@->context();
$offset = $@->column();
$previous_error = $@->_prev();
```

DESCRIPTION

The XML::LibXML::Error class is a tiny frontend to *libxml2*'s structured error support. If XML::LibXML is compiled with structured error support, all errors reported by libxml2 are transformed to XML::LibXML::Error objects. These objects automatically serialize to the corresponding error messages when printed or used in a string operation, but as objects, can also be used to get a detailed and structured information about the error that occurred.

Unlike most other XML::LibXML objects, XML::LibXML::Error doesn't wrap an underlying *libxml2* structure directly, but rather transforms it to a blessed Perl hash reference containing the individual fields of the structured error information as hash key-value pairs. Individual items (fields) of a structured error can either be obtained directly as $@->{field}, or using autoloaded methods such as $@->**field()** (where field is the field name). XML::LibXML::Error objects have the following fields: domain, code, level, file, line, nodename, message, str1, str2, str3, num1, num2, and _prev (some of them may be undefined).

`$XML::LibXML::Error::WARNINGS`

```
$XML::LibXML::Error::WARNINGS=1;
```

Traditionally, XML::LibXML was suppressing parser warnings by setting libxml2's global variable xmlGetWarningsDefaultValue to 0. Since 1.70 we do not change libxml2's global variables anymore; for backward compatibility, XML::LibXML suppresses warnings. This variable can be set to 1 to enable reporting of these warnings via Perl `warn` and to 2 to report hem via `die`.

as_string

```
$message = $@->as_string();
```

This function serializes an XML::LibXML::Error object to a string containing the full error message

close to the message produced by *libxml2* default error handlers and tools like xmllint. This method is also used to overload "" operator on XML::LibXML::Error, so it is automatically called whenever XML::LibXML::Error object is treated as a string (e.g. in print $@).

dump

```
print $@->dump();
```

This function serializes an XML::LibXML::Error to a string displaying all fields of the error structure individually on separate lines of the form 'name' => 'value'.

domain

```
$error_domain = $@->domain();
```

Returns string containing information about what part of the library raised the error. Can be one of: "parser", "tree", "namespace", "validity", "HTML parser", "memory", "output", "I/O", "ftp", "http", "XInclude", "XPath", "xpointer", "regexp", "Schemas datatype", "Schemas parser", "Schemas validity", "Relax-NG parser", "Relax-NG validity", "Catalog", "C14N", "XSLT", "validity".

code

```
$error_code = $@->code();
```

Returns the actual libxml2 error code. The XML::LibXML::ErrNo module defines constants for individual error codes. Currently libxml2 uses over 480 different error codes.

message

```
$error_message = $@->message();
```

Returns a human-readable informative error message.

level

```
$error_level = $@->level();
```

Returns an integer value describing how consequent is the error. XML::LibXML::Error defines the following constants:

- XML_ERR_NONE = 0
- XML_ERR_WARNING = 1 : A simple warning.
- XML_ERR_ERROR = 2 : A recoverable error.
- XML_ERR_FATAL = 3 : A fatal error.

file

```
$filename = $@->file();
```

Returns the filename of the file being processed while the error occurred.

line

```
$line = $@->line();
```

The line number, if available.

nodename

```
$nodename = $@->nodename();
```

Name of the node where error occurred, if available. When this field is non-empty, libxml2 actually returned a physical pointer to the specified node. Due to memory management issues, it is very difficult to implement a way to expose the pointer to the Perl level as a XML::LibXML::Node. For this reason, XML::LibXML::Error currently only exposes the name the node.

str1

```
$error_str1 = $@->str1();
```

Error specific. Extra string information.

str2

```
$error_str2 = $@->str2();
```

Error specific. Extra string information.

str3

```
$error_str3 = $@->str3();
```

Error specific. Extra string information.

num1

```
$error_num1 = $@->num1();
```

Error specific. Extra numeric information.

num2

```
$error_num2 = $@->num2();
```

In recent libxml2 versions, this value contains a column number of the error or 0 if N/A.

context

```
$string = $@->context();
```

For parsing errors, this field contains about 80 characters of the XML near the place where the error occurred. The field `$@->column()` contains the corresponding offset. Where N/A, the field is undefined.

column

```
$offset = $@->column();
```

See `$@->column()` above.

_prev

```
$previous_error = $@->_prev();
```

This field can possibly hold a reference to another XML::LibXML::Error object representing an error which occurred just before this error.

AUTHORS

Matt Sergeant, Christian Glahn, Petr Pajas

VERSION

2.0134

COPYRIGHT

LICENSE