# Package: xmlparsedata (via r-universe)

June 12, 2024

**Title** Parse Data of 'R' Code as an 'XML' Tree

**Version** 1.0.5.9000

**Description** Convert the output of 'utils::getParseData()' to an 'XML' tree, that one can search via 'XPath', and easier to manipulate in general.

**License** MIT + file LICENSE

**URL** https://github.com/r-lib/xmlparsedata#readme, https://r-lib.github.io/xmlparsedata/

**BugReports** https://github.com/r-lib/xmlparsedata/issues

**Depends** R (>= 3.0.0)

**Suggests** covr, testthat (>= 3.0.0), xml2

**Config/Needs/website** tidyverse/tidytemplate

**Config/testthat/edition** 3

**Encoding** UTF-8

**LazyData** true

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.2.3

**Repository** https://r-lib.r-universe.dev

**RemoteUrl** https://github.com/r-lib/xmlparsedata

**RemoteRef** HEAD

**RemoteSha** 1598c72053ec84ea25b765d8f8daaddf700b6874

## Contents

## expr_as_xml *Get an XML representation of an expression*

### Description

Get an XML representation of an expression

### Usage

```
expr_as_xml(expr)
```

### Arguments

| | |
|---|---|
| expr | An expression. |

## xmlparsedata *Parse Data of R Code as an 'XML' Tree*

### Description

Convert the output of 'utils::getParseData()' to an 'XML' tree, that is searchable and easier to manipulate in general.

## xml_parse_data *Convert R parse data to XML*

### Description

In recent R versions the parser can attach source code location information to the parsed expressions. This information is often useful for static analysis, e.g. code linting. It can be accessed via the `utils::getParseData()` function.

### Usage

```
xml_parse_data(x, includeText = NA, pretty = FALSE)
```

### Arguments

| | |
|---|---|
| x | an expression returned from `parse`, or a function or other object with source reference information |
| includeText | logical; whether to include the text of parsed items in the result |
| pretty | Whether to pretty-indent the XML output. It has a small overhead which probably only matters for very large source files. |

## Details

`xml_parse_data()` converts this information to an XML tree. The R parser's token names are preserved in the XML as much as possible, but some of them are not valid XML tag names, so they are renamed, see the xml_parse_token_map vector for the mapping.

The top XML tag is `<exprlist>`, which is a list of expressions, each expression is an `<expr>` tag. Each tag has attributes that define the location: `line1`, `col1`, `line2`, `col2`. These are from the `getParseData()` data frame column names. Next, there are two attributes, `start` and `end`, which can be used as an ordering of expressions in the document. Note that while the values are correlated with (and in some cases may match exactly) positions in the document, this cannot be relied upon.

See an example below. See also the README at `https://github.com/r-lib/xmlparsedata#readme` for examples on how to search the XML tree with the `xml2` package and XPath expressions.

Note that `xml_parse_data()` silently drops all control characters (0x01-0x1f) from the input, except horizontal tab (0x09) and newline (0x0a), because they are invalid in XML 1.0.

## Value

An XML string representing the parse data. See details below.

## See Also

xml_parse_token_map for the token names. `https://github.com/r-lib/xmlparsedata#readme` for more information and use cases.

## Examples

```
code <- "function(a = 1, b = 2) {\n  a + b\n}\n"
expr <- parse(text = code, keep.source = TRUE)

# The base R way:
getParseData(expr)

cat(xml_parse_data(expr, pretty = TRUE))
```

---

# xml_parse_token_map *Map token names of the R parser to token names in xml_parse_data()*

---

## Description

Some of the R token names are not valid XML tag names, so `xml_parse_data()` needs to replace them to create a valid XML file.

## Usage

```
xml_parse_token_map
```

## Format

An object of class `character` of length 20.

## See Also

`xml_parse_data()`

# Index