

Generated Documentation



Contents

| | |
|---|----|
| Package SAFOX Procedural Elements | 2 |
| safox_g.cls.php | 2 |
| Define SAFOXDOC | 3 |
| Define SAFOXVERSION | 3 |
| Define SAFOX_G | 3 |
| Package SAFOX Classes | 4 |
| Class xmlDoc | 4 |
| Constructor xmlDoc | 4 |
| Method addNode | 4 |
| Method addNodeAfter | 5 |
| Method addNodeBefore | 5 |
| Method cleanUp | 5 |
| Method destroy | 5 |
| Method getChildNodeByTagName | 6 |
| Method getId | 6 |
| Method getNextNode | 6 |
| Method getNodeById | 6 |
| Method setEncoding | 7 |
| Method setStandAlone | 7 |
| Method setUida | 7 |
| Method setVersion | 7 |
| Method writeToFile | 8 |
| Method writeXML | 8 |
| Class xmlNode | 8 |
| Constructor xmlNode | 8 |
| Method addNode | 9 |
| Method addNodeAfter | 9 |
| Method addNodeBefore | 9 |
| Method cleanUp | 10 |
| Method delete | 10 |
| Method destroy | 10 |
| Method getAttribute | 10 |
| Method getCData | 11 |
| Method getChildNodeByTagName | 11 |
| Method getChildNodeByType | 11 |
| Method getId | 11 |
| Method getNextNode | 11 |
| Method getParent | 12 |
| Method getType | 12 |
| Method hasChildren | 12 |
| Method hasParent | 12 |
| Method remove | 12 |
| Method setAttribute | 12 |

| | |
|--|----|
| Method setCData | 13 |
| Method setId | 13 |
| Method setParent | 13 |
| Method setType | 13 |
| Method writeXML | 13 |
| safox_p.cls.php | 15 |
| Define SAFOX_P | 16 |
| Define XML_MAX_LINE_SIZE | 16 |
| Define XPE_ON_PARSE_ATT | 16 |
| Define XPE_ON_PARSE_CDT | 16 |
| Define XPE_ON_PARSE_CMT | 16 |
| Define XPE_ON_PARSE_DEF | 16 |
| Define XPE_ON_PARSE_DTD | 16 |
| Define XPE_ON_PARSE_NOD | 16 |
| Define XPE_ON_PARSE_TAG | 16 |
| Define XPE_ON_PARSE_XSP | 16 |
| Function onParseCDATA | 16 |
| Class xmlParser | 17 |
| Constructor xmlParser | 18 |
| Method destroy | 18 |
| Method getError | 18 |
| Method getErrors | 18 |
| Method getXmlDoc | 18 |
| Method loadFile | 18 |
| Method parse | 18 |
| Method registerEvent | 19 |
| Method setString | 20 |
| Method setUida | 20 |
| Appendices | 21 |
| Appendix A - Class Trees | 22 |
| SAFOX | 22 |

Package SAFOX Procedural Elements

safox_g.cls.php

The SAFOX package is a collection of light-weight API for object-oriented PHP to handle XML files.

Providing three base classes, XMLDoc, XMLNode, XMLParser, the SAFOX package allows an easy and clean way to handle XML files. The SAFOX package is currently developed in PHP4.X and copyright by Christian Hansel, cpi service, Leipzig. The code is provided under the GNU Public License and may freely be used, modified, and distributed with original copyright notices to be maintained.

If no copy of the license is provided check <http://www.gnu.org/licenses/gpl.txt>

Other Contributors: Cristiano Degiorgis [cri@webprofessor.it] SAFOX_G : simple API for XML - Generation The SAFOX Package consists of two files:

- safox_g.cls.php -- Provides the XMLDOC & XMLNODE Classes
- safox_p.cls.php -- Provides a simple, light-weight, but strong Parser for XML Documents that returns a XMLDOC Object

full api documentation and examples available at <http://www.cpi-service.com/safox/>

Changes History :

version 0.42

- BUGFIX 20051130-2 fixes a problem where comments containing tags caused the the parser to break
version 0.41 of SAFOX package includes changes
- BUGFIX 20051130 Code Cleanup provided by Cristiano Degiorgis [cri@webprofessor.it]
version 0.4 of SAFOX package includes changes
- BUGFIX 20051128 that fixes problems with removing nodes from the document includes complete rework of destroy method
- remove, and delete methods added as aliases to destroy in XMLNode class
- method cleanUP added to XMLDoc class
- method cleanUP added to XMLNode class - see destription
version 0.3 of SAFOX package includes changes
- BUGFIX 20051123 corrects a problem that was caused when a repeated call to setld of child Nodes was made
- addNodeAfter Method added to XMLDoc Class
- addNodeAfter Method added to XMLNode Class
- addNodeBefore Method added to XMLDoc Class

- `addNodeBefore` Method added to `XMLNode` Class
version 0.2 of SAFOX package includes changes
- `getChildNodeByTagName()` Method added to `XMLDOC`
- `writeToFile` Method added to `XMLDOC`
- BUGFIX 20051117 in `xmlParser` - corrects a mishandling of `<![CDATA[..]]>` enclosed tag content that
caused the parser to break when dealing with tags within the CDATA

- **Package** SAFOX
- **Sub-Package** SAFOX_G
- **Copyright** (c) 2003-2005 CVH, cpi-service
- **File** `safox_g.cls.php`
- **Date** 2005-11-28
- **Version** 0.41
- **Link** <http://www.cpi-service.com>
- **License** GNU
- **Author** CVH, cpi-service ; cvh@cpi-service.com

SAFOXDOC = -1 *[line 62]*
SAFOXVERSION = 0.42 *[line 64]*
SAFOX_G = true *[line 60]*

Package SAFOX Classes

Class xmlDoc *[line 78]*

class xmlDoc

Changes History : See File Information for changes *

- **Package** SAFOX
- **Sub-Package** SAFOX_G
- **License** GNU
- **Version** 0.42
- **Author** CVH, cpi-service; cvh@cpi-service.com

Constructor *void* function xmlDoc::xmlDoc() *[line 122]*

Constructor

xmlnode function xmlDoc::addNode([\$nodeType = "DATA"], [\$sid = false]) *[line 193]*

Function Parameters:

- *\$nodeType* **\$nodeType** typ or TagName of of node to be created e.g. "product" for <product>
- *\$sid* **\$sid** sets a value for the uida (unique ID attribute)

addNode Adds a Node to the Main part of the xmlDoc;

xmlnode function xmlDoc::addNodeAfter(&\$nodeBefore, [\$nodeType = "DATA"], [\$sid = false]) *[line 218]*

Function Parameters:

- ***\$nodeBefore* &*\$nodeBefore***
 - xmlreference to the Node
- ***\$nodeType* *\$nodeType*** or TagName of of node to be created e.g. "product" for <product>
- ***\$sid* *\$sid*** sets a value for the uida (unique ID attribute)

Adds a new Node to the Document directly *after* a specified node, whereas addNode() method adds a node to the end of the Document

xmlnode function xmlDoc::addNodeBefore(&\$nodeAfter, [\$nodeType = "DATA"], [\$sid = false]) *[line 270]*

Function Parameters:

- ***\$nodeBefore* &*\$nodeAfter***
 - xmlreference to the Node
- ***\$nodeType* *\$nodeType*** or TagName of of node to be created e.g. "product" for <product>
- ***\$sid* *\$sid*** sets a value for the uida (unique ID attribute)

Adds a new Node to the Document directly *before* a specified node, whereas addNode() method adds a node to the end of the Document

void function xmlDoc::cleanUp() *[line 161]*

Cleans up internal data arrays from removed nodes and rebuilds the ID Referencing Array

This method should be called directly after removing nodes from the document, need not be called if `node->destroy(clean)` is called with optional parameter `clean = true` (default) When removing more than one node it is faster to call `node->destroy(clean)` with parameter `clean` set to false and calling `xmlDoc->cleanUp()` after making all removals

void function xmlDoc::destroy([*\$clean* = false]) *[line 131]*

Function Parameters:

- *clean* **\$clean**
 - boolean if set true all nodes will be destroyed recursively

Destructor- empties internal data

mixed function xmlDoc::getChildNodeByTagName(\$tagName) [*line 423*]

Function Parameters:

- *string* **\$tagName** -- node's tagname to return

getChildNodeByTagName - direct reference to xmlNode Object if only one tag found, array of xmlNode Object references if more found, or boolean false -- suggested by Cristiano

SAFOXDOC function xmlDoc::getId() [*line 377*]

return Id (Constant SAFOXDOC) This function returns the mixed returns the UIDA (Unique ID Attribute) Value of current node, for the XMLDOC returns SAFOXDOC (constant)

mixed function xmlDoc::getNextNode([\$step = 1]) [*line 396*]

Function Parameters:

- *\$step* **\$step** -- Int Step & Direction : -1 previous;0-current; 1-next; 2-next after next etc.

Returns next Reference to Child Node or False;

mixed function xmlDoc::getNodeById(\$id) [*line 383*]

Function Parameters:

- *\$id* **\$id**
 - mixed registered ID value (UIDA)

Return a xmlNodeObject or False

`void function xmlDoc::setEncoding([$encoding = "utf-8"]) [line 322]`

Function Parameters:

- **`$encoding $encoding`**
 - string by default "utf-8"

Set the XML-Document Encoding Attribute

`void function xmlDoc::setStandAlone([$standalone = true]) [line 338]`

Function Parameters:

- **`$standalone $standalone`**
 - boolean sets the standalone attribute of xml document

Sets the XML Document StandAlone Attribute

`void function xmlDoc::setUida([$uida = "id"]) [line 148]`

Function Parameters:

- **`$uida $uida`**
 - the name of the unique ID attribute

Set the "Unique ID Attribute"-name (usually ID)

`void function xmlDoc::setVersion([$version = "1.0"]) [line 330]`

Function Parameters:

- **`$version $version`**
 - string by default "1.0"

Sets the XML-Document Version-Attribute

boolean function xmlDoc::writeToFile(\$filepath) [*line 484*]

Function Parameters:

- **\$filepath \$filepath**
 - specified filepath

Writing an XML Document to File

string function xmlDoc::writeXML() [*line 444*]

returns a string of the XML Document

Class xmlDoc

[*line 509*]

class xmlDoc

Changes History : See File Information for changes

- **Package** SAFOX
- **Sub-Package** SAFOX_G
- **License** GNU
- **Version** 0.42
- **Author** CVH, cpi-service; cvh@cpi-service.com

Constructor *void* function xmlDoc::xmlDoc([\$type = "DATA"], &\$doc) [*line 571*]

Function Parameters:

- **\$type \$type**
- **\$doc &\$doc**

Constructor

reference function xmlNode::addNode(\$type, [\$sid = "false"]) [line 723]

Function Parameters:

- *\$type* **\$type** -- Type of tag/Node

addNode - Adds a child node to the current Object (Node) and returns a reference to childNode

xmlNode function xmlNode::addNodeAfter(&\$nodeBefore, [\$nodeType = "DATA"], [\$sid = false]) [line 744]

Function Parameters:

- *\$nodeBefore* **&\$nodeBefore**
 - xmlreference to the Node
- *\$nodeType* **\$nodeType** or TagName of of node to be created e.g. "product" for <product>
- *\$sid* **\$id** sets a value for the uida (unique ID attribute)

Adds a new Node to the Node directly *after* a specified ChildNode, whereas addNode() method adds a node to the end of the Node

xmlNode function xmlNode::addNodeBefore(&\$nodeAfter, [\$nodeType = "DATA"], [\$sid = false]) [line 797]

Function Parameters:

- *\$nodeBefore* **&\$nodeAfter**
 - xmlreference to the Node
- *\$nodeType* **\$nodeType** or TagName of of node to be created e.g. "product" for <product>
- *\$sid* **\$id** sets a value for the uida (unique ID attribute)

Adds a new Node to the Node directly *before* a specified ChildNode, whereas addNode() method adds a node to the end of the Node

void function xmlNode::cleanUp() [line 873]

Cleans up internal data arrays from removed nodes and rebuilds the ID Referencing Array

This method should not be called directly. Instead call the CleanUp method of the parenting XMLDOC Object after removing nodes from the document. This function need not be called if `node->destroy(clean)` is called with optional parameter `clean = true` (default). When removing more than one node it is faster to call `node->destroy(clean)` with parameter `clean` set to false and calling `xmlDoc->cleanUp()` after making all removals.

void function xmlNode::delete([\$clean = true]) [line 894]

Function Parameters:

- ***\$clean* *\$clean***
 - boolean if set true deletes child-nodes recursively and resets the internal arrays if this

Alias to destroy - Deletes a node and all childnodes, if the optional parameter clean is set true (default), a call to the cleanUp Method of the XMLDOC Object is made automatically.

To enhance performance you may want to set `clean` to false if you are removing more than one node and call `xmlDoc->cleanUp()` after these operations manually. Otherwise the `cleanUp` operation is called automatically everytime a call to `destroy` is made.

void function xmlNode::destroy([\$clean = true]) [line 918]

Function Parameters:

- ***\$clean* *\$clean***
 - boolean if set true deletes child-nodes recursively and resets the internal arrays if this

Deletes a node and all childnodes, if the optional parameter clean is set true (default), a call to the cleanUp Method of the XMLDOC Object is made automatically.

To enhance performance you may want to set `clean` to false if you are removing more than one node and call `xmlDoc->cleanUp()` after these operations manually. Otherwise the `cleanUp` operation is called automatically everytime a call to `destroy` is made.

mixed function xmlNode::getAttribute(\$attribute) [line 620]

Function Parameters:

- *\$attribute* **\$attribute**
 - string name of attribute

getAttribute returns the value of an attribute

unknown function xmlNode::getCdata() [*line 635*]

setCdata - returns the CData content of a node

direct function xmlNode::getChildNodeByTagName(\$tagName) [*line 687*]

Function Parameters:

- *string* **\$tagName** -- node's tagname to return

getChildNodeByTagName - returns a reference to specific ChildNode of current Node
-- suggested by Cristiano

direct function xmlNode::getChildNodeByType(\$type, \$tagName) [*line 678*]

Function Parameters:

- *string* **\$tagName** -- node's tagname to return

alias to getChildNodeByTagName - returns a reference to specific ChildNode of current Node -- suggested by Cristiano

mixed function xmlNode::getId() [*line 642*]

getId -- return the Unique ID attribute value

reference function xmlNode::getNextNode([\$step = 1]) [*line 650*]

Function Parameters:

- *optional* **\$step** -- defines step and direction (+1 = next, -1 = previous etc.)

getNextNode - returns a reference to the Next ChildNode of current Node

reference function `xmlNode::getParent()` [*line 713*]

getParent - Returns a reference to the Parent NodeObject or null

string function `xmlNode::getType()` [*line 588*]

getType returns the type/TagName of a xml node

boolean function `xmlNode::hasChildren()` [*line 859*]

hasChildren - Returns true if childNodes exist or false

boolean function `xmlNode::hasParent()` [*line 706*]

hasParent - Returns true if a reference to a parent node exists

void function `xmlNode::remove([$clean = true])` [*line 906*]

Function Parameters:

- ***\$clean* *\$clean***
 - boolean if set true deletes child-nodes recursively and resets the internal arrays if this

Alias to destroy - Deletes a node and all childnodes, if the optional parameter **clean** is set true (default), a call to the **cleanUp** Method of the XMLDOC Object is made automatically.

To enhance performance you may want to set **clean** to false if you are removing more than one node and call `xmlDoc->cleanUp()` after these operations manually. Otherwise the **cleanUp** operation is called automatically everytime a call to **destroy** is made

void function `xmlNode::setAttribute($attribute, $value)` [*line 612*]

Function Parameters:

- ***\$attribute* *\$attribute***
 - name of an attribute
- ***\$value* *\$value***
 - string or int - Value to be set

setAttribute sets an Attribute

void function xmlNode::setCDATA(\$string) [line 628]

Function Parameters:

- ***\$string* \$string**
 - string value of content to be set

setCDATA - sets the CDATA content of a node

void function xmlNode::setId(\$id) [line 596]

Function Parameters:

- ***\$id* \$id**
 - mixed string or int that uniquely identifies the node

setId sets the Unique ID Attribute defined by xmlDoc->setUida or "ID"

void function xmlNode::setParent(&\$node) [line 850]

Function Parameters:

- ***\$node* &\$node**
 - reference to parentNode

setParent - Sets the parentNode (usually used internally or if moved)

void function xmlNode::setType(\$type) [line 581]

Function Parameters:

- ***\$type* \$type** string of tagname/type of node e.g. "product" for <product>

setType Sets the type of a xml node/tag

string function xmlNode::writeXML() [line 937]

returns a string containing the wellformed xml of current Node/tag

safox_p.cls.php

The SAFOX package is a collection of light-weight API for object-oriented PHP to handle XML files.

Providing three base classes, XMLDoc, XMLNode, XMLParser, the SAFOX package allows an easy and clean way to handle XML files. The SAFOX package is currently developed in PHP4.X and copyright by Christian Hansel, cpi service, Leipzig. The code is provided under the GNU Public License and may freely be used, modified, and distributed with original copyright notices to be maintained.

If no copy of the license is provided check <http://www.gnu.org/licenses/gpl.txt>

Other Contributors: Cristiano Degiorgis [cri@webprofessor.it]

SAFOX_P : simple API for XML - PARSING The SAFOX Package consists of two files:

- safox_g.cls.php -- Provides the XMLDOC & XMLNODE Classes
- safox_p.cls.php -- Provides a simple, light-weight, but strong Parser for XML Documents that returns a XMLDOC Object

Events fired during Parsing are: XPE_ON_PARSE_DEF - Event On Parsing <?xml .. ?>

XPE_ON_PARSE_DTD - On Parsing DTD

XPE_ON_PARSE_TAG - On Parsing TagName/Type of Node

XPE_ON_PARSE_ATT - On Parsing Attribute String

XPE_ON_PARSE_CDT - On Parsing CDATA/content of node

XPE_ON_PARSE_NOD - On Node Parsing Completed Object

XPE_ON_PARSE_CMT - On Parsing a Comment

XPE_ON_PARSE_XSP - On Parsing XML Specific Tags <? ?> * see the documented onParseCDATA function that is by default registered as eventhandler for XPE_ON_PARSE_CDT for further information and an explanatory example full api documentation and examples available at <http://www.cpi-service.com/safox/>

Changes History :

version 0.42

- BUGFIX 20051130-2 fixes a problem where comments containing tags caused the the parser to break
version 0.41 of SAFOX package includes changes
- BUGFIX 20051130 Code Cleanup provided by Cristiano Degiorgis [cri@webprofessor.it]
Changes History : version 0.4 of SAFOX package includes changes
- BUGFIX 20051128 that fixes problems with removing nodes from the document
- remove, and delete methods added as aliases to destroy in XMLNode class
- method cleanUP added to XMLDoc class
- method cleanUP added to XMLNode class - see destription
Changes History : version 0.3 of SAFOX package includes changes
- BUGFIX 20051123 corrects a problem that was caused when a repeated call to setld of child Nodes was made
- addNodeAfter Method added to XMLDoc Class

- addNodeAfter Method added to XMLNode Class
- addNodeBefore Method added to XMLDoc Class
- addNodeBefore Method added to XMLNode Class
- version 0.2 of SAFOX package includes changes
- getChildNodeByTagName() Method added to XMLDOC
- writeToFile Method added to XMLDOC
- BUGFIX 20051117 in xmlParser - corrects a mishandling of <![CDATA[..]]> enclosed tag content that caused the parser to break when dealing with tags within the CDATA

- **Package** SAFOX
- **Sub-Package** SAFOX_P
- **Copyright** (c) 2003-2005 CVH, cpi-service
- **File** safox_p.cls.php
- **Date** 2005-11-28
- **Version** 0.41
- **Link** <http://www.cpi-service.com>
- **License** GNU
- **Author** CVH, cpi-service ; cvh@cpi-service.com

SAFOX_P = true *[line 83]*

Event identifiers & constants

Constants defining types of events fired during parsing

XML_MAX_LINE_SIZE = 4096 *[line 88]*

XPE_ON_PARSE_ATT = 3 *[line 92]*

XPE_ON_PARSE_CDT = 4 *[line 93]*

XPE_ON_PARSE_CMT = 6 *[line 95]*

XPE_ON_PARSE_DEF = 0 *[line 89]*

XPE_ON_PARSE_DTD = 1 *[line 90]*

XPE_ON_PARSE_NOD = 5 *[line 94]*

XPE_ON_PARSE_TAG = 2 *[line 91]*

XPE_ON_PARSE_XSP = 7 *[line 96]*

void function onParseCDATA(&\$cdata, [\$node = null], [\$parser = null]) *[line 131]*

Function Parameters:

- ***\$cdata* & *\$cdata***
 - referenced variable containing the content /CDATA that was detected during parsing
- ***\$node* *\$node***
 - referenced node object to which the content will be added during parsing
- ***\$parser* *\$parser***
 - reference to parser object

OnParseCDATA - is registered in Class XMLPARSER as a PreProcessing EventHandler for CDATA-Strings called when the XPE_ON_PARSE_CDT event is fired

Listed here to demonstrate how events should be handled

```
function onParseCDATA(&$cdata,$node = null,$parser = null) {
```

```
    if ($cdata) {
        $search = array ( "\![CDATA\[\"'\"([\r\n\s]*)\"'\",
            "\"&(quot|#34);'\"&(amp|#38);'\"&(lt|#60);'\"&(gt|#62);'\"&(apos|#39);'\" );
        $replace = array( "\"\"\", \"'\", \"&\", \"<\", \">\", \"\" );
        $cdata = preg_replace ($search, $replace, $cdata);
```

```
    }} Events fired during Parsing are: XPE_ON_PARSE_DEF - Event On Parsing <?xml ..
```

```
?>
```

XPE_ON_PARSE_DTD - On Parsing DTD

XPE_ON_PARSE_TAG - On Parsing TagName/Type of Node

XPE_ON_PARSE_ATT - On Parsing Attribute String

XPE_ON_PARSE_CDT - On Parsing CDATA/content of node

XPE_ON_PARSE_NOD - On Node Parsing Completed Object

XPE_ON_PARSE_CMT - On Parsing a Comment

XPE_ON_PARSE_XSP - On Parsing XML Specific Tags <? ?> *

```
require_once dirname(__FILE__)."/". "safox_g.cls.php" [line 85]
```

Class xmlParser

[line 172]

class xmlParser

Changes History : See File Information for changes

- **Package** SAFOX
- **Sub-Package** SAFOX_P
- **License** GNU
- **Version** 0.42
- **Author** CVH, cpi-service; cvh@cpi-service.com

Constructor *void* function xmlParser::xmlParser() [*line 217*]

Constructor

void function xmlParser::destroy() [*line 226*]

Destructor

string function xmlParser::getError() [*line 585*]

returns the last occurred error or false when all errors have been returned

-- function xmlParser::getErrors() [*line 577*]

Returns Array of Errors occurred or false

(*xmlDoc*) function xmlParser::getXmlDoc() [*line 248*]

return the XMLDOC Object if successfully parsed otherwise false

boolean function xmlParser::loadFile(\$filename) [*line 269*]

Function Parameters:

- ***\$filename* \$filename**
 - string filename to be set

Load a String to be parsed from a xml Document File

boolean function xmlParser::parse() [*line 291*]

Parse the String and create an xmlDoc Object

- **Access public**

void function xmlParser::registerEvent(\$event, \$functionname) [line 649]

Function Parameters:

- **\$event \$event**
- **\$functionname \$functionname**

registerEvent - registers a functionname to be executed when event is fired

currently supported events: XPE_ON_PARSE_DEF (0); // Event On Parsing <? xml ..
 String XPE_ON_PARSE_DTD (1); // On Parsing DTD XPE_ON_PARSE_TAG (2); // On
 Parsing TagName XPE_ON_PARSE_ATT (3); // On Parsing Attribute String
 XPE_ON_PARSE_CDT (4); // On Parsing CDATA XPE_ON_PARSE_NOD (5); // On Node
 Parsing Completed Object XPE_ON_PARSE_CMT (6); // On Parsing a Comment
 XPE_ON_PARSE_XSP (7); // On Parsing XML Specific Tags <?;

User-Defined Eventhandlers will be called with two parameters :

- **&\$data** - relevant information for event (e.g. the unparsed DOCTYPE string in the XPE_ON_PARSE_DTD event)
- **&\$node** - reference to Node Object

EventHandlers should be considered for "PreProcessing Data" if Events are handled by XMLPARSER themselves FullProcessing - The Following Events are currently not handled by the XMLPARSER

- XPE_ON_PARSE_DTD (an event is fired providing the DOCTYPE string that otherwise is ignored by the Parser and the reference to the xmlDoc-object and the parser object-ref)
- XPE_ON_PARSE_CMT (an event is fired providing the comment-tag string that otherwise is ignored by the Parser and the reference to the xmlDoc (or Node)-object and the parser object-ref)
- XPE_ON_PARSE_XSP (an event is fired providing the special XML-tag string that otherwise is ignored by the Parser and the reference to the xmlDoc-object and the parser object-ref)

Post-Processing Events :

- XPE_PN_PARSE_NOD (an event is fired providing an empty string (no unparsed stringdata left) and the reference to the node-object and the parser object-ref)

All Other Events are to be used for PreProcessing Purposes and will be provided the relevant string-data, a null var , and the parser reference

- **Access public**

void function xmlParser::setString(\$string) [line 236]

Function Parameters:

- *\$string* **\$string**

Set String to be parsed

function xmlParser::setUida(\$uida, \$node-) [line 261]

Function Parameters:

- *\$uida* **\$node-**
 - set the "unique ID Attribute"-name to which the parser should listen to call >setId()

setUida - sets the "unique ID Attribute"-name

that is used for GetNodeById, SetId etc of XMLNode/Doc Objects a uida usually is "id" but sometimes in xml may be something like "recordno", "uid" ...

Appendices

Appendix A - Class Trees

Package SAFOX

xmlDoc

- [xmlDoc](#)

xmlNode

- [xmlNode](#)

xmlParser

- [xmlParser](#)

Index

C

| | |
|--|----|
| constructor xmlParser::xmlParser() | 18 |
| <i>Constructor</i> | |
| constructor xmlNode::xmlNode() | 8 |
| <i>Constructor</i> | |
| constructor xmlDoc::xmlDoc() | 4 |
| <i>Constructor</i> | |

O

| | |
|--|----|
| onParseCDATA() | 16 |
| <i>OnParseCDATA - is registered in Class XMLPARSER as a PreProcessing EventHandler for CDATA-Strings called when the XPE_ON_PARSE_CDT event is fired</i> | |

S

| | |
|---|----|
| safox_p.cls.php | 15 |
| <i>The SAFOX package is a collection of light-weight API for object-oriented PHP to handle XML files.</i> | |
| SAFOX_P | 16 |
| <i>Event identifiers & constants</i> | |
| SAFOX_G | 3 |
| SAFOXVERSION | 3 |
| SAFOXDOC | 3 |
| safox_g.cls.php | 2 |
| <i>The SAFOX package is a collection of light-weight API for object-oriented PHP to handle XML files.</i> | |

X

| | |
|---|----|
| xmlNode::writeXML() | 13 |
| <i>returns a string containing the wellformed xml of current Node/tag</i> | |
| xmlNode::setType() | 13 |
| <i>setType Sets the type of a xml node/tag</i> | |
| XML_MAX_LINE_SIZE | 16 |
| XPE_ON_PARSE_ATT | 16 |
| XPE_ON_PARSE_CMT | 16 |
| XPE_ON_PARSE_CDT | 16 |
| xmlNode::setParent() | 13 |
| <i>setParent - Sets the ParentNode (usually used internally or if moved)</i> | |
| xmlNode::setId() | 13 |
| <i>setId sets the Unique ID Attribute defined by xmlDoc->setUida or "ID"</i> | |

| | |
|--|----|
| xmlNode::hasParent() | 12 |
| <i>hasParent - Returns true if a reference to a parent node exists</i> | |
| xmlNode::hasChildren() | 12 |
| <i>hasChildren - Returns true if childNodes exist or false</i> | |
| xmlNode::remove() | 12 |
| <i>Alias to destroy - Deletes a node and all childnodes, if the optional parameter clean is set true (default), a call to the cleanUp Method of the XMLDOC Object is made automatically.</i> | |
| xmlNode::setAttribute() | 12 |
| <i>setAttribute sets an Attribute</i> | |
| xmlNode::setCdata() | 13 |
| <i>setCdata - sets the CData content of a node</i> | |
| XPE ON PARSE DEF | 16 |
| XPE ON PARSE DTD | 16 |
| xmlParser::loadFile() | 18 |
| <i>Load a String to be parsed from a xml Document File</i> | |
| xmlParser::getXmldoc() | 18 |
| <i>return the XMLDOC Object if succesfully parsed otherwise false</i> | |
| xmlParser::parse() | 18 |
| <i>Parse the String and create an xmldoc Object</i> | |
| xmlParser::registerEvent() | 19 |
| <i>registerEvent - registers a functionname to be exucted when event is fired</i> | |
| xmlParser::setUida() | 20 |
| <i>setUida - sets the "unique ID Attribute"-name</i> | |
| xmlParser::setString() | 20 |
| <i>Set String to be parsed</i> | |
| xmlParser::getErrors() | 18 |
| <i>Returns Array of Errors occured or false</i> | |
| xmlParser::getError() | 18 |
| <i>returns the last occured error or false when all errors have been returned</i> | |
| XPE ON PARSE TAG | 16 |
| XPE ON PARSE NOD | 16 |
| XPE ON PARSE XSP | 16 |
| xmlParser | 17 |
| <i>class xmlParser</i> | |
| xmlParser::destroy() | 18 |
| <i>Destructor</i> | |
| xmlNode::getType() | 12 |
| <i>getType returns the type/TagName of a xml node</i> | |
| xmlNode::getParent() | 12 |
| <i>getParent - Returns a reference to the Parent NodeObject or null</i> | |
| xmlDoc::getNodeById() | 6 |
| <i>Return a xmlNodeObject or False</i> | |
| xmlDoc::getNextNode() | 6 |
| <i>Returns next Reference to Child Node or False;</i> | |
| xmlDoc::setEncoding() | 7 |
| <i>Set the XML-Document Encoding Attribute</i> | |
| xmlDoc::setStandAlone() | 7 |
| <i>Sets the XML Document StandAlone Attribute</i> | |
| xmlDoc::setVersion() | 7 |
| <i>Sets the XML-Document Version-Attribute</i> | |
| xmlDoc::setUida() | 7 |
| <i>Set the "Unique ID Attribute"-name (usually ID)</i> | |

| | |
|--|----|
| xmlDoc::getId() | 6 |
| <i>return Id (Constant SAFOXDOC) This function returns the mixed returns the UIDA (Unique ID Attribute) Value of current node, for the XMLDOC returns SAFOXDOC (constant)</i> | |
| xmlDoc::getChildNodeByTagName() | 6 |
| <i>getChildNodeByTagName - direct reference to xmlNode Object if only one tag found, array of xmlNode Object references if more found, or boolean false</i> | |
| <i>-- suggested by Cristiano</i> | |
| xmlDoc::addNodeAfter() | 5 |
| <i>Adds a new Node to the Document directly *after* a specified node, whereas addNode() method</i> | |
| <i>adds a node to the end of the Document</i> | |
| xmlDoc::addNode() | 4 |
| <i>addNode Adds a Node to the Main part of the xmlDoc;</i> | |
| xmlDoc::addNodeBefore() | 5 |
| <i>Adds a new Node to the Document directly *before* a specified node, whereas addNode() method</i> | |
| <i>adds a node to the end of the Document</i> | |
| xmlDoc::cleanUp() | 5 |
| <i>Cleans up internal data arrays from removed nodes and rebuilds the ID Referencing Array</i> | |
| xmlDoc::destroy() | 5 |
| <i>Destructor- empties internal data</i> | |
| xmlDoc::writeToFile() | 8 |
| <i>Writing an XML Document to File</i> | |
| xmlDoc::writeXML() | 8 |
| <i>returns a string of the XML Document</i> | |
| xmlNode::getCdata() | 11 |
| <i>setCdata - returns the CData content of a node</i> | |
| xmlNode::getAttribute() | 10 |
| <i>getAttribute returns the value of an attribute</i> | |
| xmlNode::getChildNodeByTagName() | 11 |
| <i>getChildNodeByTagName - returns a reference to specific ChildNode of current Node</i> | |
| <i>-- suggested by Cristiano</i> | |
| xmlNode::getChildNodeByType() | 11 |
| <i>alias to getChildNodeByTagName - returns a reference to specific ChildNode of current Node</i> | |
| <i>-- suggested by Cristiano</i> | |
| xmlNode::getNextNode() | 11 |
| <i>getNextNode - returns a reference to the Next ChildNode of current Node</i> | |
| xmlNode::getId() | 11 |
| <i>getId -- return the Unique ID attribute value</i> | |
| xmlNode::destroy() | 10 |
| <i>Deletes a node and all childnodes, if the optional parameter clean is set true (default), a call to the cleanUp Method of the XMLDOC Object is made automatically.</i> | |
| xmlNode::delete() | 10 |
| <i>Alias to destroy - Deletes a node and all childnodes, if the optional parameter clean is set true (default), a call to the cleanUp Method of the XMLDOC Object is made automatically.</i> | |
| xmlNode::addNode() | 9 |
| <i>addNode - Adds a child node to the current Object (Node) and returns a reference to childNode</i> | |
| xmlNode | 8 |
| <i>class xmlNode</i> | |
| xmlNode::addNodeAfter() | 9 |

| | | |
|--|--|----|
| | <i>Adds a new Node to the Node directly *after* a specified ChildNode, whereas addNode() method</i> | |
| | <i>adds a node to the end of the Node</i> | |
| xmlNode::addNodeBefore() | | 9 |
| | <i>Adds a new Node to the Node directly *before* a specified ChildNode, whereas addNode() method</i> | |
| | <i>adds a node to the end of the Node</i> | |
| xmlNode::cleanUp() | | 10 |
| | <i>Cleans up internal data arrays from removed nodes and rebuilds the ID Referencing Array</i> | |
| xmlDoc | | 4 |
| | <i>class xmlDoc</i> | |