

**NAME**

XML::XPathEngine::NodeSet – a list of XML document nodes

**DESCRIPTION**

An XML::XPathEngine::NodeSet object contains an ordered list of nodes. The nodes each take the same format as described in XML::XPathEngine::XMLParser.

**SYNOPSIS**

```
my $results = $xp->find('//someelement');
if (!$results->isa('XML::XPathEngine::NodeSet')) {
    print "Found $results\n";
    exit;
}
foreach my $context ($results->get_nodelist) {
    my $newresults = $xp->find('./other/element', $context);
    ...
}
```

**API***new()*

You will almost never have to create a new NodeSet object, as it is all done for you by XPath.

*get\_nodelist()*

Returns a list of nodes. See XML::XPathEngine::XMLParser for the format of the nodes.

*string\_value()*

Returns the string-value of the first node in the list. See the XPath specification for what “string-value” means.

*string\_values()*

Returns a list of the string-values of all the nodes in the list.

*to\_literal()*

Returns the concatenation of all the string-values of all the nodes in the list.

**get\_node(\$pos)**

Returns the node at \$pos. The node position in XPath is based at 1, not 0.

*size()*

Returns the number of nodes in the NodeSet.

*pop()*

Equivalent to perl’s pop function.

**push(@nodes)**

Equivalent to perl’s push function.

**append(\$nodeset)**

Given a nodeset, appends the list of nodes in \$nodeset to the end of the current list.

*shift()*

Equivalent to perl’s shift function.

**unshift(@nodes)**

Equivalent to perl’s unshift function.

**prepend(\$nodeset)**

Given a nodeset, prepends the list of nodes in \$nodeset to the front of the current list.