\s	a whitespace character (space, tab, newline)
\S	non-whitespace character
\d	a digit (0-9)
\D	a non-digit
\w	a word character (a-z, A-Z, 0-9, _)
\W	a non-word character
[aeiou]	matches a single character in the given set
[^aeiou]	matches a single character outside the given set
(foo|bar|baz)	matches any of the alternatives specified

Modifiers

Several modifiers are available that can make your work with regexps much easier, like case sensitivity, searching in multiple lines etc.

Modifier	Description
i	Makes the match case insensitive
m	Specifies that if the string has newline or carriage return characters, the ^ and $ operators will now match against a newline boundary, instead of a string boundary
o	Evaluates the expression only once
s	Allows use of . to match a newline character
x	Allows you to use white space in the expression for clarity
g	Globally finds all matches
cg	Allows a search to continue even after a global match fails

PHP's Regexp PERL Compatible Functions

PHP offers following functions for searching strings using Perl-compatible regular expressions:

Function	Description
preg_match()	The preg_match() function searches string for pattern, returning true if pattern exists, and false otherwise.
preg_match_all()	The preg_match_all() function matches all occurrences of pattern in string.
preg_replace()	The preg_replace() function operates just like ereg_replace(), except that regular expressions can be used in the pattern and replacement input parameters.

preg_split()	The preg_split() function operates exactly like split(), except that regular expressions are accepted as input parameters for pattern.
preg_grep()	The preg_grep() function searches all elements of input_array, returning all elements matching the regexp pattern.
preg_quote()	Quote regular expression characters

XML

XML is a markup language that looks a lot like HTML. An XML document is plain text and contains tags delimited by < and >.There are two big differences between XML and HTML:

- XML doesn't define a specific set of tags you must use.
- XML is extremely picky about document structure.

XML gives you a lot more freedom than HTML. HTML has a certain set of tags: the <a> tags surround a link, the <p> startsa paragraph and so on. An XML document, however, can use any tags you want. Put <rating></rating> tags around a movie rating, >height></height> tags around someone's height. Thus XML gives you option to device your own tags.

XML is very strict when it comes to document structure. HTML lets you play fast and loose with some opening and closing tags. BUt this is not the case with XML.

HTML list that's not valid XML

```
<ul>
<li>Braised Sea Cucumber
<li>Baked Giblets with Salt
<li>Abalone with Marrow and Duck Feet
</ul>
```

This is not a valid XML document because there are no closing tags to match up with the three opening <li> tags. Every opened tag in an XML document must be closed.

HTML list that is valid XML

```
<ul>
<li>Braised Sea Cucumber</li>
<li>Baked Giblets with Salt</li>
<li>Abalone with Marrow and Duck Feet</li>
```

```
</ul>
```

Parsing an XML Document

PHP 5's new **SimpleXML** module makes parsing an XML document, well, simple. It turns an XML document into an object that provides structured access to the XML.

To create a SimpleXML object from an XML document stored in a string, pass the string to **simplexml_load_string( )**. It returns a SimpleXML object.

Example

Try out following example:

```
<?php

$channel =<<<_XML_
<channel>
<title>What's For Dinner<title>
<link>http://menu.example.com/<link>
<description>Choose what to eat tonight.</description>
</channel>
_XML_;

$xml = simplexml_load_string($channel);
print "The $xml->title channel is available at $xml->link. ";
print "The description is \"$xml->description\"";
?>
```

It will produce following result:

```
The What's For Dinner channel is available at http://menu.example.com/. The description is "Choose what to eat tonight."
```

NOTE: You can use function **simplexml_load_file(filename)** if you have XML content in a file.

For a complete detail of XML parsing function check **PHP Function Reference**.

Generating an XML Document

SimpleXML is good for parsing existing XML documents, but you can't use it to create a new one from scratch.

The easiest way to generate an XML document is to build a PHP array whose structure mirrors that of the XML document and then to iterate through the array, printing each element with appropriate formatting.

Example

Try out following example:

```
<?php

$channel = array('title' => "What's For Dinner",
                 'link' => 'http://menu.example.com/',
                 'description' => 'Choose what to eat tonight.');
print "<channel>\n";
foreach ($channel as $element => $content) {
  print " <$element>";
  print htmlentities($content);
  print "</$element>\n";
}
print "</channel>";
?>
```

It will produce following result:

```
<channel>
<title>What's For Dinner</title>
<link>http://menu.example.com/</link>
<description>Choose what to eat tonight.</description>
</channel></html>
```

DOM

A DOM (Document Object Model) defines a standard way for accessing and manipulating documents.

The XML DOM defines a standard way for accessing and manipulating XML documents.