# XML Expat 解析器

* [Previous Page](http://www.w3school.com.cn/php/php_db_odbc.asp)
* [Next Page](http://www.w3school.com.cn/php/php_xml_dom.asp)

**内建的 Expat 解析器使在 PHP 中处理 XML 文档成为可能。**

## 什么是 XML？

XML 用于描述数据，其焦点是数据是什么。XML 文件描述了数据的结构。

在 XML 中，没有预定义的标签。您必须定义自己的标签。

如果希望学习更多有关 XML 的内容，请访问我们的 [XML 教程](http://www.w3school.com.cn/xml/index.asp)。

## 什么是 Expat？

如需读取和更新 - 创建创建并处理 - 一个 XML 文档，您需要 XML 解析器。

有两种基本的 XML 解析器类型：

* 基于树的解析器：这种解析器把 XML 文档转换为树型结构。它分析整篇文档，并提供了 API 来访问树种的元素，例如文档对象模型 (DOM)。
* 基于事件的解析器：将 XML 文档视为一系列的事件。当某个具体的事件发生时，解析器会调用函数来处理。

Expat 解析器是基于事件的解析器。

基于事件的解析器集中在 XML 文档的内容，而不是它们的结果。正因如此，基于事件的解析器能够比基于树的解析器更快地访问数据。

请看下面的 XML 片段：

<from>John</from>

基于事件的解析器把上面的 XML 报告为一连串的三个事件：

* 开始元素：from
* 开始 CDATA 部分, 值：John
* 关闭元素： from

上面的 XML 范例包含了形式良好的 XML。不过这个例子是无效的 XML，因为没有与它关联的文档类型声明 (DTD)，也没有内嵌的 DTD。

不过，在使用 Expat 解析器时，这没有区别。Expat 是不检查有效性的解析器，忽略任何 DTD。

作为一款基于事件、非验证的 XML 解析器，Expat 快速且轻巧，十分适合 PHP 的 web 应用程序。

注释：XML 文档必须形式良好，否则 Expat 会生成错误。

## 安装

XML Expat 解析器是 PHP 核心的组成部分。无需安装就可以使用这些函数。

## XML 文件

将在我们的例子中使用下面的 XML 文件：

<?xml version="1.0" encoding="ISO-8859-1"?>

<note>

<to>George</to>

<from>John</from>

<heading>Reminder</heading>

<body>Don't forget the meeting!</body>

</note>

## 初始化 XML 解析器

我们要在 PHP 中初始化 XML 解析器，为不同的 XML 事件定义处理器，然后解析这个 XML 文件。

### 例子

<?php

//Initialize the XML parser

$parser=xml\_parser\_create();

//Function to use at the start of an element

function start($parser,$element\_name,$element\_attrs)

{

switch($element\_name)

{

case "NOTE":

echo "-- Note --<br />";

break;

case "TO":

echo "To: ";

break;

case "FROM":

echo "From: ";

break;

case "HEADING":

echo "Heading: ";

break;

case "BODY":

echo "Message: ";

}

}

//Function to use at the end of an element

function stop($parser,$element\_name)

{

echo "<br />";

}

//Function to use when finding character data

function char($parser,$data)

{

echo $data;

}

//Specify element handler

xml\_set\_element\_handler($parser,"start","stop");

//Specify data handler

xml\_set\_character\_data\_handler($parser,"char");

//Open XML file

$fp=fopen("test.xml","r");

//Read data

while ($data=fread($fp,4096))

{

xml\_parse($parser,$data,feof($fp)) or

die (sprintf("XML Error: %s at line %d",

xml\_error\_string(xml\_get\_error\_code($parser)),

xml\_get\_current\_line\_number($parser)));

}

//Free the XML parser

xml\_parser\_free($parser);

?>

以上代码的输出：

-- Note --

To: George

From: John

Heading: Reminder

Message: Don't forget the meeting!

### 工作原理解释：

* 通过 xml\_parser\_create() 函数初始化 XML 解析器
* 创建配合不同事件处理程序的的函数
* 添加 xml\_set\_element\_handler() 函数来定义，当解析器遇到开始和结束标签时执行哪个函数
* 添加 xml\_set\_character\_data\_handler() 函数来定义，当解析器遇到字符数据时执行哪个函数
* 通过 xml\_parse() 函数来解析文件 "test.xml"
* 万一有错误的话，添加 xml\_error\_string() 函数把 XML 错误转换为文本说明
* 调用 xml\_parser\_free() 函数来释放分配给 xml\_parser\_create() 函数的内存

## 更多 PHP Expat 解析器的信息

如需更多有关 PHP Expat 函数的信息，请访问我们的 PHP XML Parser 参考手册。

# XML DOM

* [Previous Page](http://www.w3school.com.cn/php/php_xml_parser_expat.asp)
* [Next Page](http://www.w3school.com.cn/php/php_xml_simplexml.asp)

**内建的 DOM 解析器使在 PHP 中处理 XML 文档成为可能。**

## 什么是 DOM？

W3C DOM 提供了针对 HTML 和 XML 文档的标准对象集，以及用于访问和操作这些文档的标准接口。

W3C DOM 被分为不同的部分 (Core, XML 和 HTML) 和不同的级别 (DOM Level 1/2/3)：

* Core DOM - 为任何结构化文档定义标准的对象集
* XML DOM - 为 XML 文档定义标准的对象集
* HTML DOM - 为 HTML 文档定义标准的对象集

如果您希望学习更多有关 XML DOM 的知识，请访问我们的 [XML DOM 教程](http://www.w3school.com.cn/xmldom/index.asp)。

## XML 解析

如需读取和更新 - 创建创建并处理 - 一个 XML 文档，您需要 XML 解析器。

有两种基本的 XML 解析器类型：

* 基于树的解析器：这种解析器把 XML 文档转换为树型结构。它分析整篇文档，并提供了 API 来访问树种的元素，例如文档对象模型 (DOM)。
* 基于事件的解析器：将 XML 文档视为一系列的事件。当某个具体的事件发生时，解析器会调用函数来处理。

DOM 解析器是基于树的解析器。

请看下面的 XML 文档片段：

<?xml version="1.0" encoding="ISO-8859-1"?>

<from>John</from>

XML DOM 把 XML 视为一个树形结构：

* Level 1: XML 文档
* Level 2: 根元素: <from>
* Level 3: 文本元素: "John"

## 安装

DOM XML 解析器函数是 PHP 核心的组成部分。无需安装就可以使用这些函数。

## XML 文件

将在我们的例子中使用下面的 XML 文件：

<?xml version="1.0" encoding="ISO-8859-1"?>

<note>

<to>George</to>

<from>John</from>

<heading>Reminder</heading>

<body>Don't forget the meeting!</body>

</note>

## 加载和输出 XML

我们需要初始化 XML 解析器，加载 XML，并把它输出：

### 例子

<?php

$xmlDoc = new DOMDocument();

$xmlDoc->load("note.xml");

print $xmlDoc->saveXML();

?>

以上代码的输出：

George John Reminder Don't forget the meeting!

假如您在浏览器窗口中查看源代码，会看到下面这些 HTML：

<?xml version="1.0" encoding="ISO-8859-1"?>

<note>

<to>George</to>

<from>John</from>

<heading>Reminder</heading>

<body>Don't forget the meeting!</body>

</note>

上面的例子创建了一个 DOMDocument-Object，并把 "note.xml" 中的 XML 载入这个文档对象中。

saveXML() 函数把内部 XML 文档放入一个字符串，这样我们就可以输出它。

## 循环 XML

我们要初始化 XML 解析器，加载 XML，并循环 <note> 元素的所有元素：

### 例子

<?php

$xmlDoc = new DOMDocument();

$xmlDoc->load("note.xml");

$x = $xmlDoc->documentElement;

foreach ($x->childNodes AS $item)

{

print $item->nodeName . " = " . $item->nodeValue . "<br />";

}

?>

以上代码的输出：

#text =

to = George

#text =

from = John

#text =

heading = Reminder

#text =

body = Don't forget the meeting!

#text =

在上面的例子中，您看到了每个元素之间存在空的文本节点。

当 XML 生成时，它通常会在节点之间包含空白。XML DOM 解析器把它们当作普通的元素，如果您不注意它们，有时会产生问题。

如果您希望学习更多有关 XML DOM 的知识，请访问我们的 [XML DOM 教程](http://www.w3school.com.cn/xmldom/index.asp)。

# SimpleXML

* [Previous Page](http://www.w3school.com.cn/php/php_xml_dom.asp)
* [Next Page](http://www.w3school.com.cn/php/php_ajax_intro.asp)

**SimpleXML 处理最普通的 XML 任务，其余的任务则交由其它扩展。**

## 什么是 SimpleXML？

SimpleXML 是 PHP 5 中的新特性。在了解 XML 文档 layout 的情况下，它是一种取得元素属性和文本的便利途径。

与 DOM 或 Expat 解析器相比，SimpleXML 仅仅用几行代码就可以从元素中读取文本数据。

SimpleXML 可把 XML 文档转换为对象，比如：

* 元素 - 被转换为 SimpleXMLElement 对象的单一属性。当同一级别上存在多个元素时，它们会被置于数组中。
* 属性 - 通过使用关联数组进行访问，其中的下标对应属性名称。
* 元素数据 - 来自元素的文本数据被转换为字符串。如果一个元素拥有多个文本节点，则按照它们被找到的顺序进行排列。

当执行类似下列的基础任务时，SimpleXML 使用起来非常快捷：

* 读取 XML 文件
* 从 XML 字符串中提取数据
* 编辑文本节点或属性

不过，在处理高级 XML 时，比如命名空间，最好使用 Expat 解析器或 XML DOM。

## 安装

从 PHP 5.0 开始，SimpleXML 函数是 PHP 核心的组成部分。无需安装就可以使用这些函数。

## 使用 SimpleXML

下面是 XML 文件：

<?xml version="1.0" encoding="ISO-8859-1"?>

<note>

<to>George</to>

<from>John</from>

<heading>Reminder</heading>

<body>Don't forget the meeting!</body>

</note>

我们打算从上面的 XML 文件输出元素的名称和数据。

这是需要做的事情：

1. 加载 XML 文件
2. 取得第一个元素的名称
3. 使用 children() 函数创建在每个子节点上触发的循环
4. 输出每个子节点的元素名称和数据

### 例子

<?php

$xml = simplexml\_load\_file("test.xml");

echo $xml->getName() . "<br />";

foreach($xml->children() as $child)

{

echo $child->getName() . ": " . $child . "<br />";

}

?>

以上代码的输出：

note

to: George

from: John

heading: Reminder

body: Don't forget the meeting!

## 更多有关 PHP SimpleXML 的信息

如需更多有关 PHP SimpleXML 的信息，请访问我们的 [PHP SimpleXML 参考手册](http://www.w3school.com.cn/php/php_ref_simplexml.asp)。

窗体顶端