万能的数据传输格式XML

# 一、学习目标

1、知道xml在编程领域的作用

2、掌握如何编写规范的xml文档

3、掌握如何用php语言解析xml

4、掌握如何获取xml格式的接口内容

5、掌握常用的xpath语法

6、知道如何查阅PHP手册中关于XML的部分

# 二、XML概述

1、发展史（了解）

GML 通用标记语言

1969年，IBM公司的一些砖家为了解决不同系统中文档格式的问题而开发的一种文档描述语言。

SGML 标准通用标记语言

1986年，ANSI组织对GML进行整理并升级为SGML。但是SGML是一种非常严谨的文档描述语言，其结构非常复杂（标准手册就有500多页），难以理解和学习，进而影响其推广与应用。

1993年 HTML 超文本标记语言（W3C）

HTML继承了SGML的许多重要的特点，比如结构化、实现独立和可描述性，但是同时它也存在很多缺陷：比如它只能使用固定的有限的标记，而且它只侧重于对内容的显示。

1998年 XML

随着Web上数据的增多，这些HTML存在的缺点就变的不可被忽略。为了解决这些问题， W3C组织又基于SGML语言开发一款新的标记语言，并称之为XML。是xml1.0.**到目前为止，xml的版本还是1.0**

2、XML定义

XML（eXtensible Markup Language）：可扩展标记语言

用于标记电子文件使其具有结构性的标记语言，可以用来标记数据、定义数据类型，是一种允许用户对自己的标记语言进行定义的源语言。

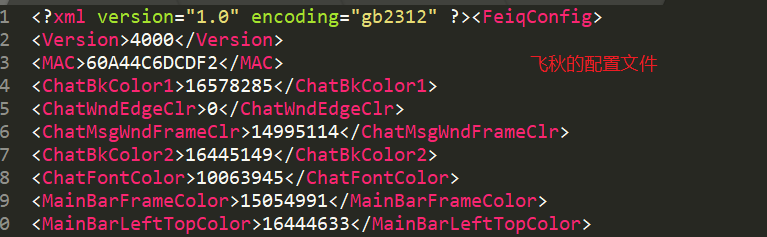
xml也是一种标记（标签）语言。也是使用<table></table>这种标签

xml的标签可扩展。也就是说xml的标签可以自定义。比如用<name></name>来表示姓名。

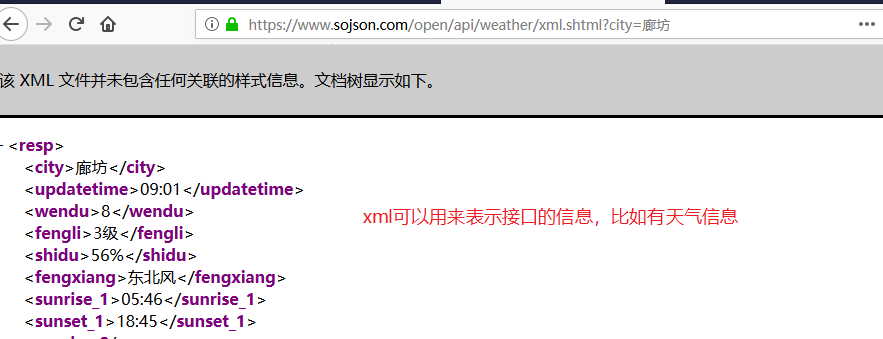
xml侧重于描述数据，html侧重于展示数据。Html用浏览器打开会有不同的样式，xml只是显示成树桩结构。

3、XML的应用场景举例

1、飞秋的配置文件



2、xml可以用于提供接口信息，比如天气接口



# 三、XML快速入门

1、使用XML描述一本书的信息

一本书有什么特点？

书名，作者，出版社，价格，类型

<book>

<name>斗破苍穹</name>

<author>天蚕土豆</author>

<chubanshe>起点</chubanshe>

<price>免费</price>

<type>玄幻</type>

</book>

2、使用XML写一个数据库配置文件

<db>

<type>mysql</type>

<host>localhost</host>

<user>….</user>

….

</db>

3、使用XML存储班里所有的人（基本信息）

人有很多，每个人都有名字，性别，年龄，身体，体重，三围

<students>

<stu>

<name>张三疯</name>

<sex>男</sex>

<age>98</age>

<weight>65kg</weight>

<height>175cm</height>

<sanwei>50,45,60</sanwei>

</stu>

<stu>

<name>刘志远</name>

<sex>女</sex>

<age>65</age>

<weight>65kg</weight>

<height>175cm</height>

<sanwei>100,200,300</sanwei>

</stu>

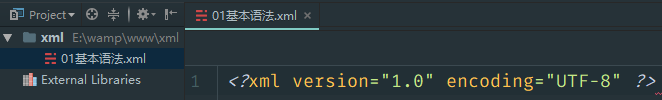
</students>

# 四、XML基本语法

1、文档声明

一个完整的xml文档必须有一个文档声明。

<?xml version="1.0" encoding="UTF-8"?>

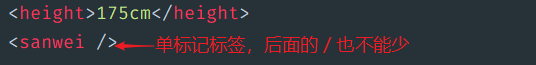


2、关于元素（标签、节点）

**元素也就是标签，在dom操作中，也叫做节点**。

①、一个XML文档必须有且仅有一个根标签

②、一个标签必须闭合，即使是单标记标签也必须闭合



③、标签不允许有交叉嵌套



除此以外，元素的命名也必须遵循一定的规则

① 元素(标签)的名称可以包含字母、数字、中横线、下划线和英文句点；

② 只能以字母或下划线开头；

③ 严格区分大小写；

④ 名称字符之间不能有空格或制表符；

⑤ 名称字符之间不能使用冒号，冒号有特殊含义。

一句话，标签命名和PHP中的变量命名一致，肯定没问题。

3、关于属性

① 属性名称的命名规范与元素的命名规范相同

② 属性值一定要用引号(单引号或双引号)引起来

③ 一个元素可以有多个属性

④ 元素中的属性是不允许重复的

⑤ 标签属性所代表的信息也可以被改成用子元素的形式来描述

4、关于注释

和html的注释一样

<!--内容-->

5、特殊符号及CDATA

在xml文档描述信息中，不要使用特殊符号，如果使用特殊符号的话，可以使用实体符号代替，如小于号<可以用实体符号&lt;表示。



如果说一个内容区有大段的文字，文字中有很多特殊符号，一个一个的替换肯定是比较麻烦，解决的办法是使用CDATA。

<![CDATA[内容]]



6、其他指令(了解)

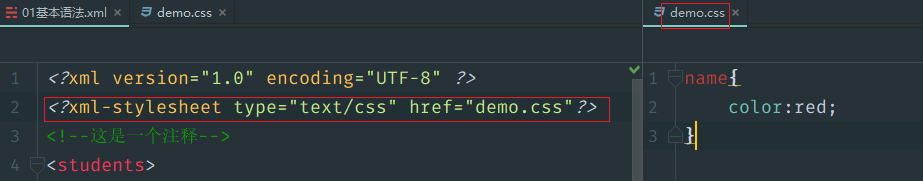
基本语法：

必须以“<?”作为开头，以“?>”作为结尾。

在实际应用中，其处理指令主要有：

xml-stylesheet指令

<?xml-stylesheet type='text/css' href='demo.css' ?>



# 五、总结XML与HTML的区别

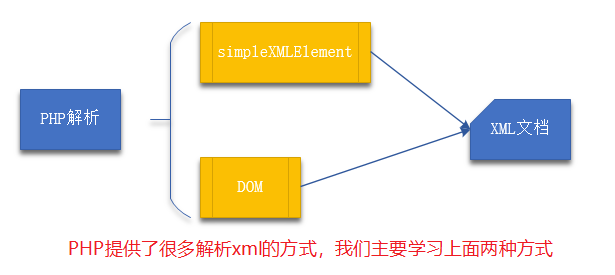
1、语法上，XML标签可以自定义，HTML不可以自定义；XML语法比较严格，HTML语法不严格

2、意义上，XML侧重于描述数据，HTML侧重于展示数据。

# 六、使用simpleXML解析XML数据

1. 何为解析

解析就是增删改查。XML文档中有标签，标签还有属性，所以这里的解析就包括对标签的增删改查和对属性的增删改查。

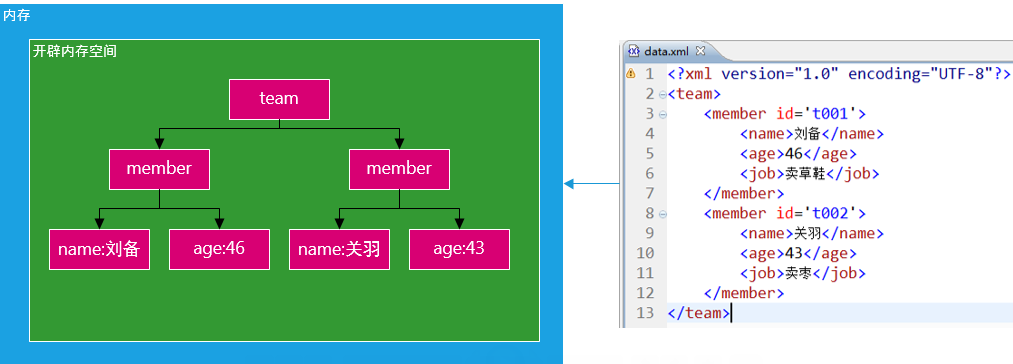


2、SimpleXML解析XML工作原理

① 开辟内存空间

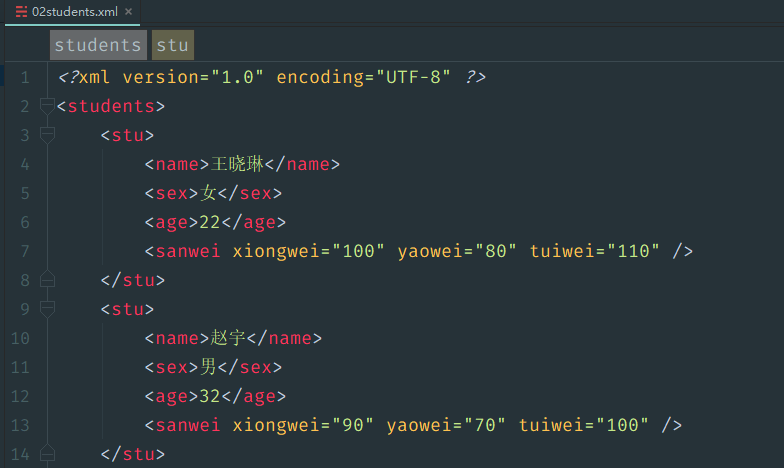
② 载入XML文件到内存形成DOM树结构

③ 生成SimpleXMLElement对象

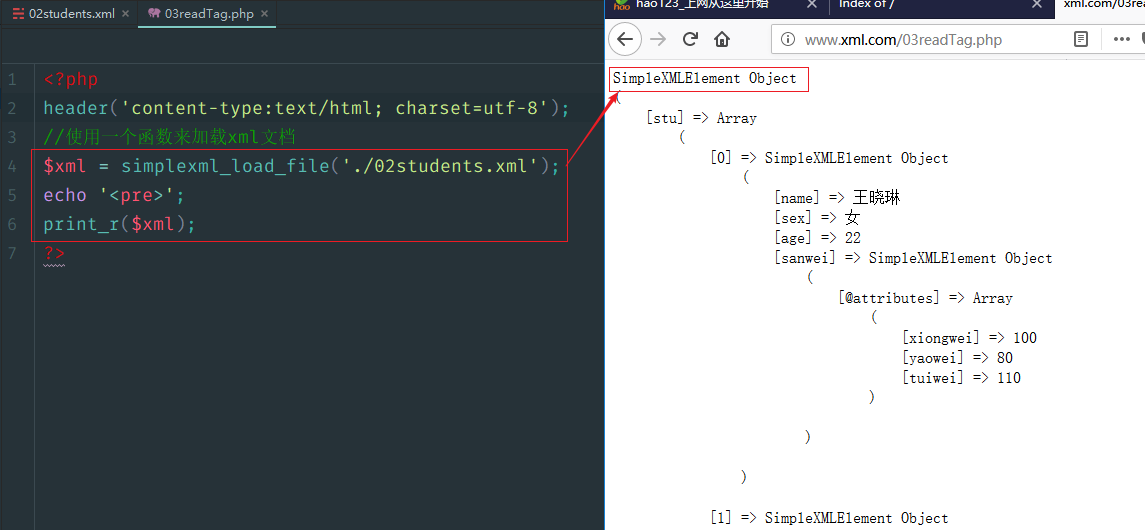


3、对元素节点进行增删改查

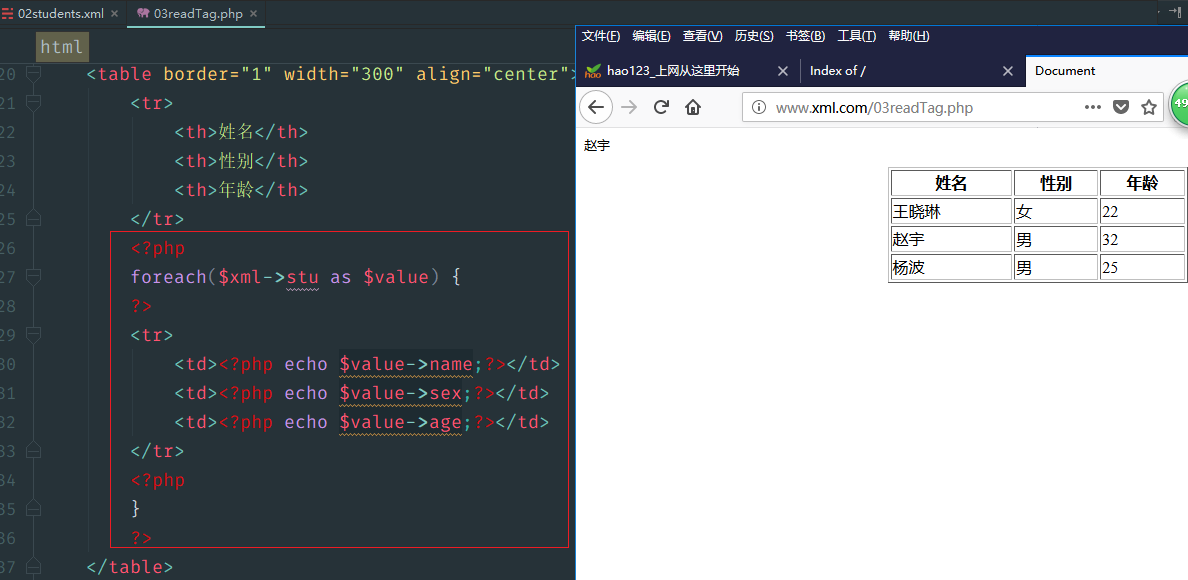
1、创建一个用于测试的xml文档



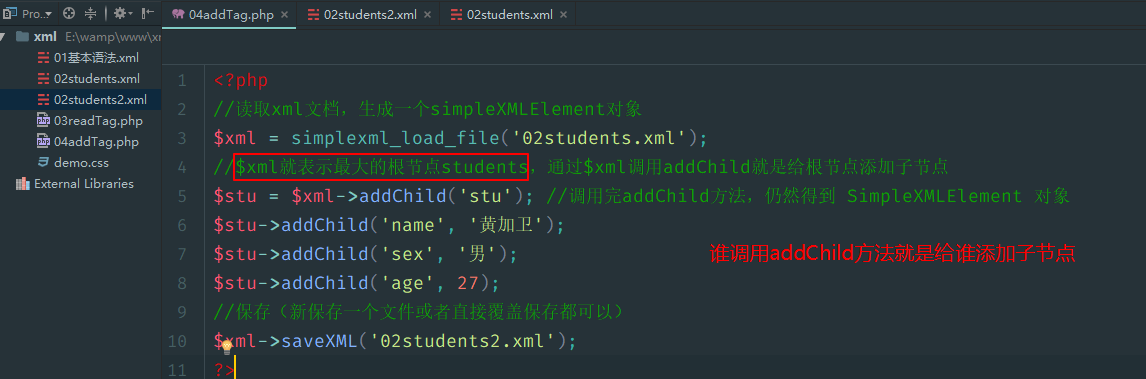
2、查询（读取）标签节点



以表格的形式输出：



3、添加标签节点



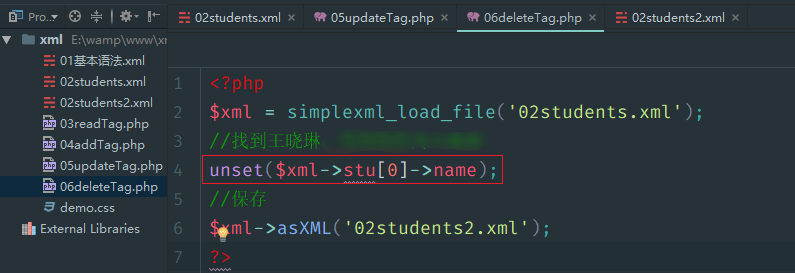
4、修改标签节点

先找到这个标签，然后给其重新赋值，最后保存即可。



5、删除标签节点

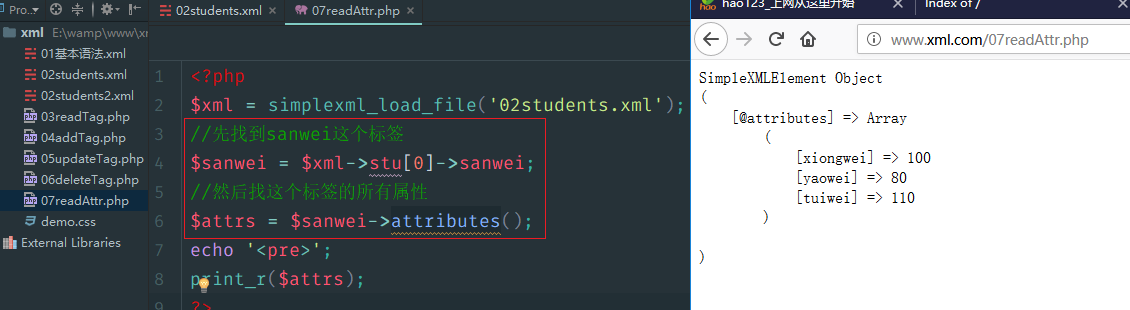
找到要删除的元素，然后用unset处理即可，最后保存。



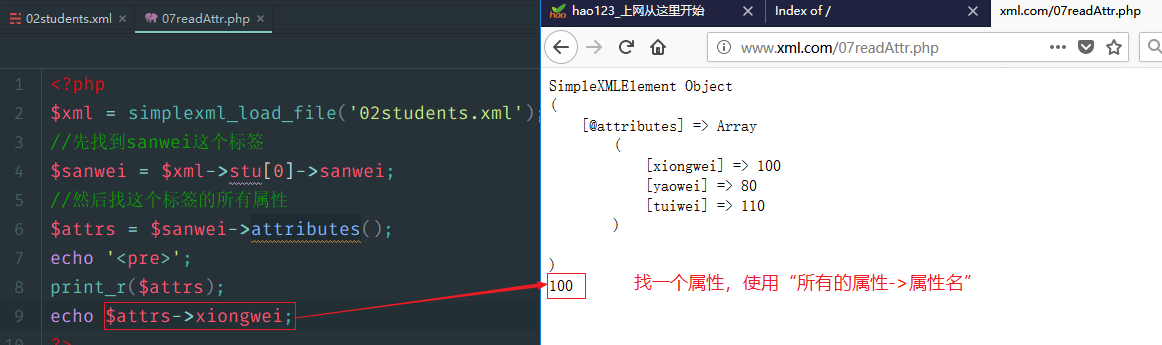
4、对属性节点进行增删改查

1、读取标签的属性

要读取属性，必须先找到标签。

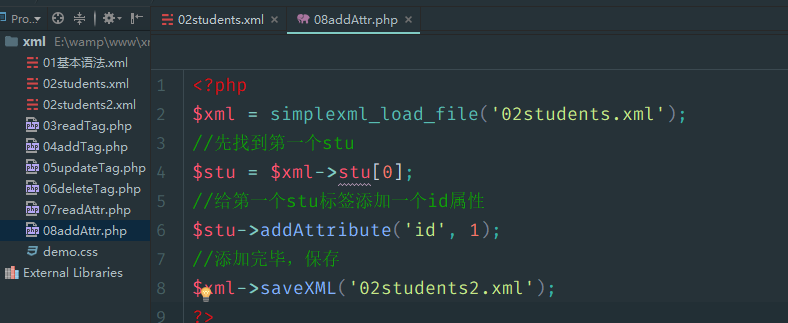


读取单个属性：



2、添加标签的属性

先找到要添加属性的标签，然后通过addAttribute()方法添加即可。



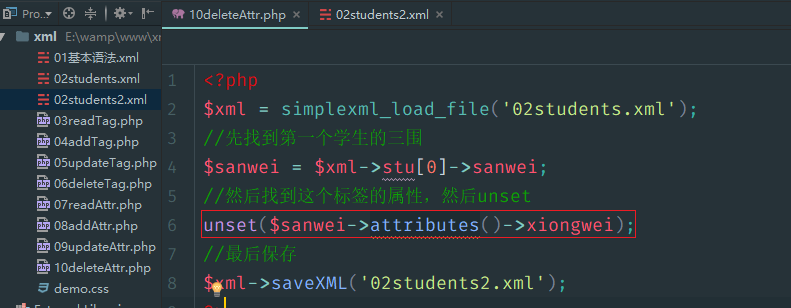
3、修改标签的属性

先找到要修改的属性，然后重新赋值。

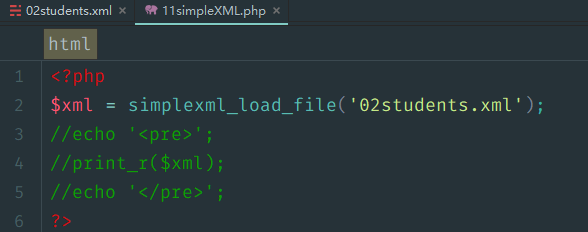


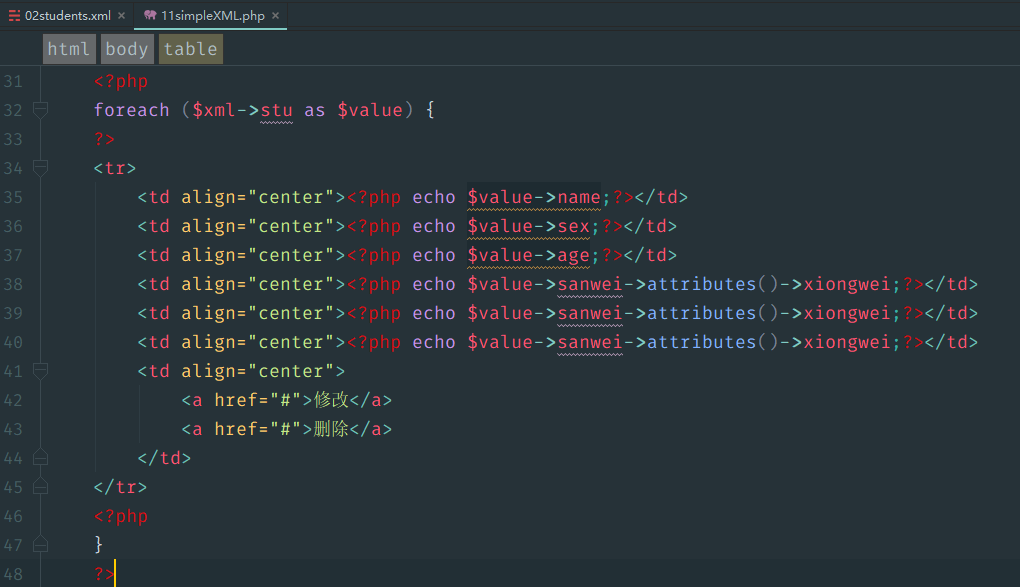
4、删除标签的属性

找到要删除的属性，然后unset



5、综合练习





# 七、案例-获取天气信息（接口调用）

1、接口说明

提供接口的网址：https://www.sojson.com/api/weather.html

XML接口网址：https://www.sojson.com/open/api/weather/xml.shtml?city=北京

接口使用说明

每个IP日调用额度2000次，超过过多，我会直接封掉IP段。

每次请求间隔必须3秒一次，如果多次超过3秒内调用多次，会封掉IP段

2、如何使用接口

访问这个接口，然后获取接口的信息。

# 八、Xpath语言

1、什么Xpath

• XPath 使用路径表达式在 XML 文档中进行导航

• XPath 包含一个标准函数库

• XPath 是 XSLT 中的主要元素

• XPath 是一个 W3C 标准

简言之，**XPath 是一种用来在内存中导航整个 XML 树的语言**。XPath 的设计初衷是作为一种面向 XSLT 和 XPointer 的语言。XPath 1.0 在 1999 年成为了一种 W3C 标准。较新的 XPath 2.0 已于 2007 获得规范的状态。

**PHP目前只支持XPath1.0**

随着 XML 其他规范的出现，XPath 的使用也相应出现。如今，XPath 已成为在 XML 模式、ISO Schematron、XQuery 和 XForms 中导航 XML 的首选语言。具有讽刺意味的是，有可依据的理由证明 XPath 并不是基于的 XML 语法。它使用了其自身的语法来避免与 URI 和其他与 XML 相关的语法的冲突。

XPath是一门在 XML 文档中查找信息的语言。

XPath可用来在 XML 文档中对元素和属性进行遍历。

而且Xpath并不仅仅应用于XML数据查询，其在很多语言中（PHP、Java、.NET以及Javascript），都可以广泛使用。

**XPath提供了一种路径语法，我们只需要按照语法去写，就可以查询到目标节点。XPath就是用于方便，快速查询xml节点的一个语言**。

2、选取节点

XPath 使用路径表达式在 XML 文档中选取节点。节点是通过沿着路径或者 step 来选取的。

下面列出了最有用的路径表达式：

|  |  |
| --- | --- |
| 表达式 | 描述 |
| nodename | 选取此节点的所有子节点。仅限于根节点后的第一个节点 |
| / | 从根节点选取。/后必须跟根节点，最多两层可查 |
| // | 从匹配选择的当前节点选择文档中的节点，而不考虑它们的位置。仅限于根节点后的第一个节点 |
| . | 选取当前节点。 |
| .. | 选取当前节点的父节点。 |
| @ | 选取属性。 |

**实例**

在下面的表格中，我们已列出了一些路径表达式以及表达式的结果：

|  |  |
| --- | --- |
| 路径表达式 | 结果 |
| stu | 选取 stu元素的所有子节点。 |
| /students | 选取根元素 students。  注释：假如路径起始于正斜杠( / )，则此路径始终代表到某元素的绝对路径！ |
| students/stu | 选取属于 students的子元素的所有 stu 元素。 |
| //stu | 选取所有 stu子元素，而不管它们在文档中的位置。 |
| bookstore//book | 选择属于 bookstore 元素的后代的所有 book 元素，而不管它们位于 bookstore 之下的什么位置。 |
| //@lang | 选取名为 lang 的所有属性。 |

3、谓语（Predicates）

谓语是对已经查询出的节点的过滤或者叫做筛选。

谓语用来查找某个特定的节点或者包含某个指定的值的节点。

谓语被嵌在方括号中。

**实例**

在下面的表格中，我们列出了带有谓语的一些路径表达式，以及表达式的结果：

|  |  |
| --- | --- |
| 路径表达式 | 结果 |
| /bookstore/book[1] | 选取属于 bookstore 子元素的第一个 book 元素。 |
| /bookstore/book[last()] | 选取属于 bookstore 子元素的最后一个 book 元素。 |
| /bookstore/book[last()-1] | 选取属于 bookstore 子元素的倒数第二个 book 元素。 |
| /bookstore/book[position()<3] | 选取最前面的两个属于 bookstore 元素的子元素的 book 元素。 |
| //title[@lang] | 选取所有拥有名为 lang 的属性的 title 元素。 |
| //title[@lang='eng'] | 选取所有 title 元素，且这些元素拥有值为 eng 的 lang 属性。 |
| /bookstore/book[price>35.00] | 选取 bookstore 元素的所有 book 元素，且其中的 price 元素的值须大于 35.00。 |
| /bookstore/book[price>35.00]/title | 选取 bookstore 元素中的 book 元素的所有 title 元素，且其中的 price 元素的值须大于 35.00。 |
| //stu[contains(name, "飞")] | 选取名字中包含有“飞”的所有学生 |

4、选取未知节点

XPath 通配符可用来选取未知的 XML 元素。

|  |  |
| --- | --- |
| 通配符 | 描述 |
| \* | 匹配任何元素节点。 |
| @\* | 匹配任何属性节点。 |
| node() | 匹配任何类型的节点。 |

**实例**

在下面的表格中，我们列出了一些路径表达式，以及这些表达式的结果：

|  |  |
| --- | --- |
| 路径表达式 | 结果 |
| /bookstore/\* | 选取 bookstore 元素的所有子元素。 |
| //\* | 选取文档中的所有元素。 |
| //title[@\*] | 选取所有带有属性的 title 元素。 |

5、选取若干路径

通过在路径表达式中使用“|”运算符，您可以选取若干个路径。

**实例**

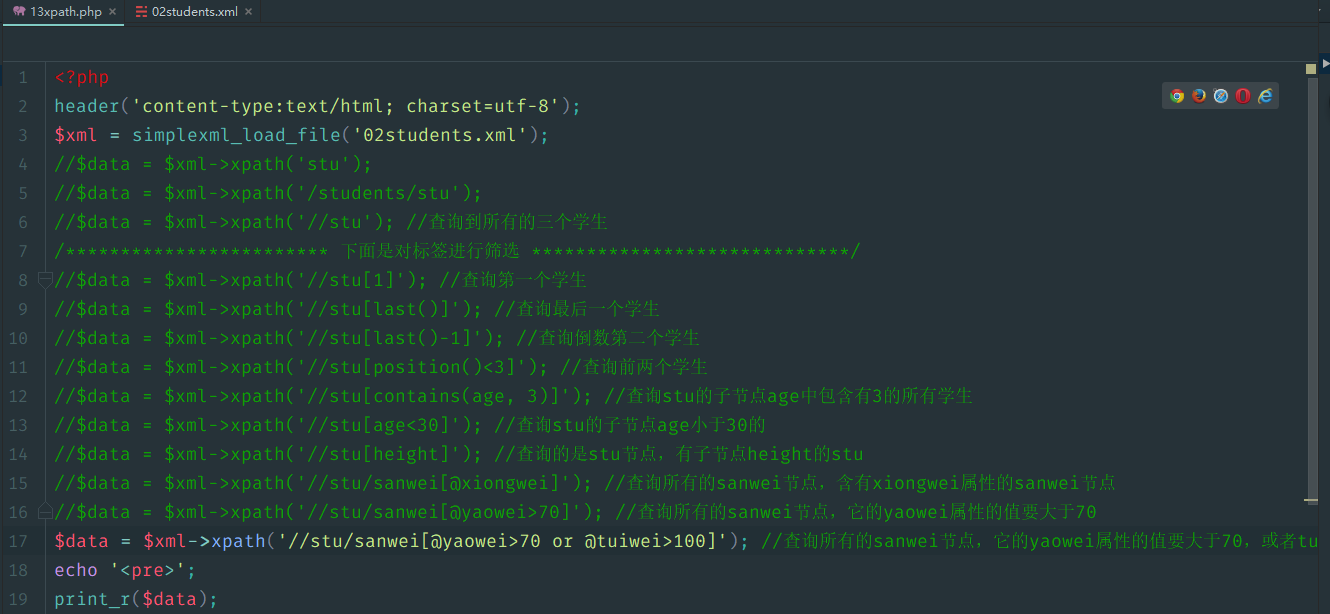
在下面的表格中，我们列出了一些路径表达式，以及这些表达式的结果：

|  |  |
| --- | --- |
| 路径表达式 | 结果 |
| //book/title | //book/price | 选取 book 元素的所有 title 和 price 元素。 |
| //title | //price | 选取文档中的所有 title 和 price 元素。 |
| /bookstore/book/title | //price | 选取属于 bookstore 元素的 book 元素的所有 title 元素，以及文档中所有的 price 元素。 |

6、XPath 运算符

下面列出了可用在 XPath 表达式中的运算符：

|  |  |  |  |
| --- | --- | --- | --- |
| 运算符 | 描述 | 实例 | 返回值 |
| | | 计算两个节点集 | //book | //cd | 返回所有拥有 book 和 cd 元素的节点集 |
| + | 加法 | 6 + 4 | 10 |
| - | 减法 | 6 - 4 | 2 |
| \* | 乘法 | 6 \* 4 | 24 |
| div | 除法 | 8 div 4 | 2 |
| = | 等于 | price=9.80 | 如果 price 是 9.80，则返回 true。  如果 price 是 9.90，则返回 false。 |
| != | 不等于 | price!=9.80 | 如果 price 是 9.90，则返回 true。  如果 price 是 9.80，则返回 false。 |
| < | 小于 | price<9.80 | 如果 price 是 9.00，则返回 true。  如果 price 是 9.90，则返回 false。 |
| <= | 小于或等于 | price<=9.80 | 如果 price 是 9.00，则返回 true。  如果 price 是 9.90，则返回 false。 |
| > | 大于 | price>9.80 | 如果 price 是 9.90，则返回 true。  如果 price 是 9.80，则返回 false。 |
| >= | 大于或等于 | price>=9.80 | 如果 price 是 9.90，则返回 true。  如果 price 是 9.70，则返回 false。 |
| or | 或 | price=9.80 or price=9.70 | 如果 price 是 9.80，则返回 true。  如果 price 是 9.50，则返回 false。 |
| and | 与 | price>9.00 and price<9.90 | 如果 price 是 9.80，则返回 true。  如果 price 是 8.50，则返回 false。 |
| mod | 计算除法的余数 | 5 mod 2 | 1 |



# 九、综合案例-SimpleXML+Xpath电子词典

1、设计xml文档，用于存放中英文

<dict>

<word>

<en>book</en>

<cn>书</cn>

</word>

<word>

<en>bag</en>

<cn>包</cn>

</word>

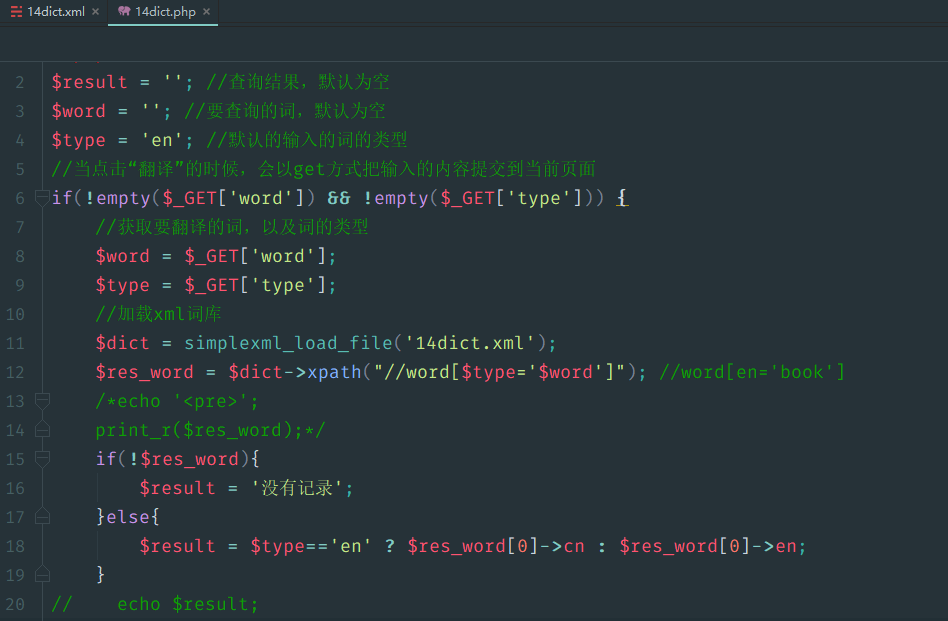
</dict>

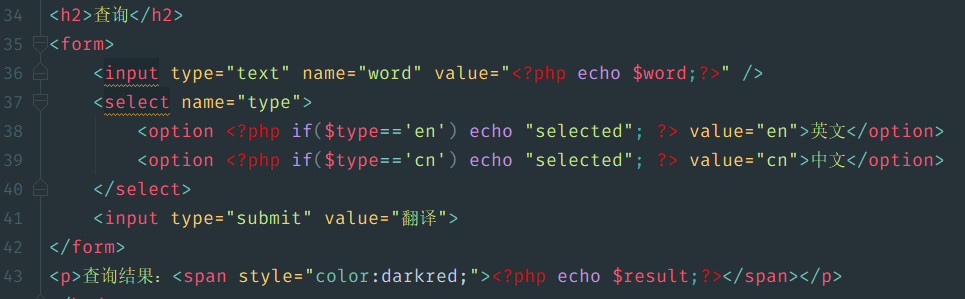
2、设计界面



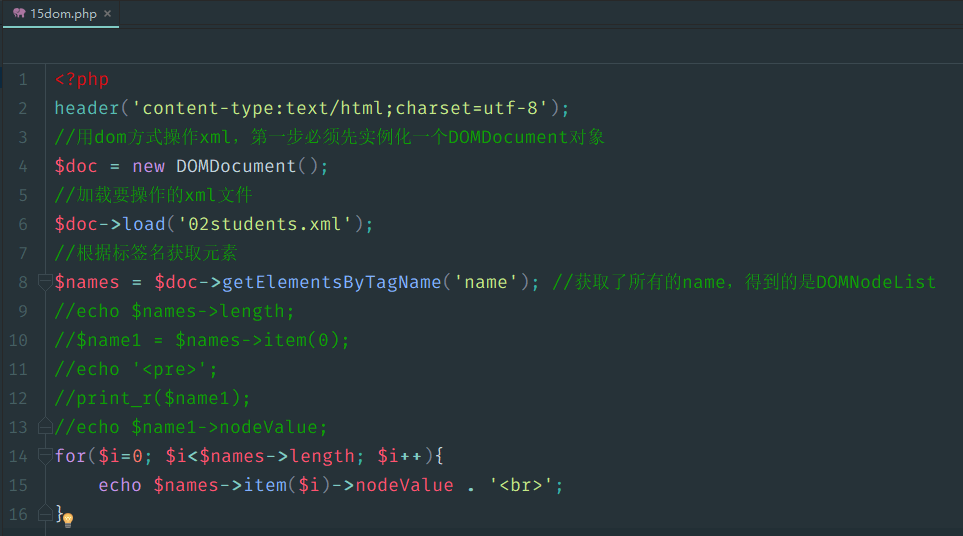


3、实现查询





# 十、DOM的方式操作XML



DOM中的XPath：

DOM中的XPath比simpleXML中的XPath支持的稍微多一点。

