# 一Xml基础

目录

[一Xml基础 1](#_Toc16654)

[1.什么是xml 1](#_Toc32266)

[2.与HTML比较 1](#_Toc12499)

[3.特点： 2](#_Toc1902)

[4.作用： 2](#_Toc29221)

[5.xml用途：XML 应用于 web 开发的许多方面，常用于简化数据的存储和共享。 2](#_Toc29720)

[6.属性： 3](#_Toc16101)

[7.实体： 4](#_Toc26561)

[8.注释 4](#_Toc8154)

[9.总结： 4](#_Toc13366)

[10..解析器： 4](#_Toc794)

[11.命名空间：XML 命名空间提供避免元素命名冲突的方法。 4](#_Toc27827)

[二 xml之CSS技术： 4](#_Toc32392)

[三 xml与dtd 5](#_Toc10461)

[四 xml与schema 8](#_Toc26454)

[五 xml与xsl 9](#_Toc7344)

[六 xml架构 10](#_Toc28351)

1.什么是xml

    XML是一种可扩展标记语言，是HTML的扩展和延伸（也就是说实质上他们是一样的，只不过补充了HTML）。它被用来传输和存储数据。Xml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是标准通用标记语言（SGML）的子集，非常适合Web传输，目前，XML 在 Web 中起到的作用不会亚于一直作为 Web 基石的 HTML。XML 无所不在，XML 是各种应用程序之间进行数据传输的最常用的工具，并且在信息存储和描述领域变得越来越流行。XML提供统一的方法来描述和交换独立于应用程序或供应商的结构化数据。

## 2.与HTML比较

1.设计目的：XML绝对不是HTML的替代，而是HTML的补充和扩展。它被用来传输和存储内容，焦点是数据的内容，它是独立于软件和硬件的信息传输工具。HTML用来显示数据，焦点是数据的外观。

2.XML的标签没有被预定义，用户可以自行定义标签。而HTML是预先定义好的，只可使用。

3.HTML的标记不是所有都需要成对出现，XML则要求所有标记都成对出现。

4.HTML标记不区分大小写，XML标记必须区分大小写。

5.XML文件必须要有根元素，它是一种树形结构，而HTML则不需要。

1. XML元素必须要被正确的嵌套，HTML则不需要。

## 3.特点：

（1）xml与操作系统、编程语言的开发平台都无关；

（2）实现不同系统之间的数据交互。

## 4.作用：

<1>配置应用程序和网站；

<2>数据交互；

<3>Ajax基石。

5.xml用途：XML 应用于 web 开发的许多方面，常用于简化数据的存储和共享。

a. XML 把数据从HTML分离:

如果你需要在 HTML 文档中显示动态数据，那么每当数据改变时将花费大量的时间来编辑 HTML。

通过 XML，数据能够存储在独立的 XML 文件中。这样你就可以专注于使用 HTML 进行布局和显示，并确保修改底层数据不再需要对 HTML 进行任何的改变。

通过使用几行 JavaScript，你就可以读取一个外部 XML 文件，然后更新 HTML 中的数据内容。

b. XML 简化数据共享:

在真实的世界中，计算机系统和数据使用不兼容的格式来存储数据。XML 数据以纯文本格式进行存储，因此提供了一种独立于软件和硬件的数据存储方法。这让创建不同应用程序可以共享的数据变得更加容易。

c. XML 简化数据传输:

通过 XML，可以在不兼容的系统之间轻松地交换数据。

对开发人员来说，其中一项最费时的挑战一直是在因特网上的不兼容系统之间交换数据。

由于可以通过各种不兼容的应用程序来读取数据，以 XML 交换数据降低了这种复杂性。

d. XML简化平台的变更:

升级到新的系统（硬件或软件平台），总是非常费时的。必须转换大量的数据，不兼容的数据经常会丢失。XML 数据以文本格式存储。这使得 XML 在不损失数据的情况下，更容易扩展或升级到新的操作系统、新应用程序或新的浏览器。

e. XML使数据更有用:

由于 XML 独立于硬件、软件以及应用程序，XML 使您的数据更可用，也更有用。不同的应用程序都能够访问您的数据，不仅仅在 HTML 页中，也可以从 XML 数据源中进行访问。

f. XML 用于创建新的 Internet 语言

很多新的 Internet 语言是通过 XML 创建的：

其中的例子包括：

XHTML - 最新的 HTML 版本

WSDL - 用于描述可用的 web service

WAP 和 WML - 用于手持设备的标记语言

RSS - 用于 RSS feed 的语言

RDF 和 OWL - 用于描述资源和本体

SMIL - 用于描述针针对 web 的多媒体

## 6.属性：

（1）语法：

<元素名 属性名=“属性值”/>

例：<Student ID=“S100”>

<Name>Tom</Name>

</Student>

(2)注意：

属性值用双引号包裹；一个元素可以有多个属性，它的基本格式为：

<元素名 属性名=“属性值” 属性名=“属性值”>;

属性值中不能够直接包含<.”,&。

## 7.实体：

在xml中，一些字符拥有特殊的意义。如果把字符“<”放在xml元素中，会发生错误，这是因为解析器会把它当作新元素的开始，这样会产生xml错误：

为了避免这个错误，请用实体引用来代替“<”字符：

xml中5个预定义实体

## 8.注释

注:注释内容中不要出现”--”;不要把注释放在标签中间；注释不能嵌套。

## 9.总结：

（1）xml描述的是文档的内容与语义，而不是文档应当如何显示；

（2）格式正规（well formed）的xml文档

10..解析器：所有现代浏览器都内建了供读取和操作 XML 的 XML 解析器。

解析器把 XML 载入内存，然后把它转换为可通过 JavaScript 访问的 XML DOM 对象。

11.命名空间：XML 命名空间提供避免元素命名冲突的方法。

# 二 xml之CSS技术：

## 1.CSS 概述：

<1>CSS 指层叠样式表 (Cascading Style Sheets)

<2>样式定义如何显示 HTML 元素

<3>样式通常存储在样式表中

<4>把样式添加到 HTML 4.0 中，是为了解决内容与表现分离的问题

<5>外部样式表可以极大提高工作效率

<6>外部样式表通常存储在 CSS 文件中;

<7>多个样式定义可层叠为一;

XML CSS：通过使用CSS,可为XML文档添加显示信息。

XML XSLT: 通过使用XSLT,您可以向XML文档添加显示信息。

XSLT是首选的XML样式表语言。

XSLT远比CSS更加完善。八使用XSLT的方法之一是在浏览器显示XML文件之前；

XML和HTML一样，同样可以嵌入CSS文件。代码如下：

<!--引用css样式文件-->

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

# 三 xml与dtd

## 1.dtd概述

定义XML文件的结构，为XML文件提供语法与规则，因为XML文件需要确定的结构，所以使用DTD.

## 2、DTD用途：

a 有了DTD，每个xml文件可以携带一个自身格式的描述,所谓格式描述就是我的xml文档里面可以写哪些东西，比如元素、属性；

b 有了DTD，不同组织的人可以使用一个通用DTD用来交换数据，xml是随意定义的，有了DTD，我们可以限制多个公司之间按照这种DTD的规则来编写xml文档，由于DTD都是统一格式，所以不同的公司之间，不同的组织之间就可以用这种通用的xml文档格式进行交互数据了；

C 应用程序可以使用一个标准DTD校检从外部接收的xml数据是否有效。

## 3、DTD编写：

DTD分成三大类，第一是内部DTD,第二是外部DTD,第三是内外结合的DTD。

（1）内部DTD文档(Students.xml)：

（2）就是说这个xml文档里面既包含xml定义，又包含DTD的定义。

这个[]里面是DTD的定义的内容，就是第一个叫内部DTD文档。

（3）外部DTD文档（Students.xml Students.dtd）:

所谓外部就是DTD的定义和我们的xml文档是在不同的文件里面。

那么外部的DTD如何使用呢？那就需要在前面的xml文档的根元素的上面这个位置跟刚才内部的是一样的，要写上

这类年改革文档在同一个文件夹下面，就可以了。

有内部DTD了，为什么还要学习外部DTD呢？？

内部的DTD,它只能验证当前的这个xml文档。那假如说，我们有一个DTD需要验证多个xml文档的话，那就直接使用外部DTD就行。

（4）内外部DTD文档结合

4、DTD的具体定义——元素的定义

元素名称：是指xml文档里面的元素名称

元素类型：

（1）EMPTY；

（2）#PCDATA；

（3）纯元素类型的DTD语法；

（4）ANY；

## 5、定义属性的语法结构：

xml文档携带数据的方式，不仅仅是xml的元素的内容，还可以是元素的属性。所以说在DTD里面我们有必要对属性进行限制。

6.实体

实体，简单理解就是C#中的常量，我们在DTD里面定义，在xml文档里面去使用。（实际上就是自定义实体，与前面的预定义实体差不多是一回事）。

## 7 .DTD代码demo

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

<<span style="color:#ff0000;">!ELEMENT </span>书籍列表 (计算机书籍)\*>

<!ELEMENT 计算机书籍 (书名,价格,简要介绍)><!--规定元素顺序-->

<!ELEMENT 书名 (#PCDATA)>

<!ELEMENT 价格 (#PCDATA)>

<!ELEMENT 简要介绍 (#PCDATA)>

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

<!DOCTYPE 书籍列表 SYSTEM <span style="color:#ff0000;">"book.dtd</span>"><!--引用DTD-->

<书籍列表>

<计算机书籍>

<书名>XML的今生今世</书名>

<价格>66.66</价格>

<简要介绍>

该书详细描写XML的从出生到衰老的整个过程

</简要介绍>

</计算机书籍>

</书籍列表>

Schema代码demo

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

<schema xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace="http://www.example.org/email"

xmlns:tns="http://www.example.org/email" elementFormDefault="qualified">

<xs:element name="email">

<xs:complexType>

<xs:sequence>

<xs:element name="<span style="color:#ff0000;">to</span>" type="xs:<span style="color:#000099;">string</span>"></xs:element>

<xs:element name="<span style="color:#ff0000;">from</span>" type="xs:<span style="color:#3333ff;">string</span>"></xs:element>

<xs:element name="<span style="color:#ff0000;">title</span>" type="xs:<span style="color:#3333ff;">string</span>"></xs:element>

<xs:element name="<span style="color:#ff0000;">body</span>" type="xs:<span style="color:#000099;">string</span>"></xs:element>

<xs:element name="<span style="color:#ff0000;">date</span>" type="xs:<span style="color:#000099;">date</span>"></xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

</schema>

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

<xml-body>

<email xmlns:xsi="http://www.w3.org/2001/XMLSchema" xsi:noNamespaceSchemalocation=<span style="color:#ff0000;"><strong>"email.xsd"</strong></span>>

<span style="color:#ff0000;"><to></span>liuweii8809@163.com</to>

<<span style="color:#ff0000;">from></span>wd15350706925@163.com</from>

<span style="color:#ff0000;"><title></span>about story</title>

<span style="color:#ff0000;"><body></span>i miss you </body>

<span style="color:#ff0000;"><date></span>2012-12-24</date>

</email>

</xml-bod>

# 四 xml与schema

## 1 定义

Schema是在DTD的基础上发展而来的，是对DTD的继任。1.保护数据通讯，可扩展，可重用。2.可针对未来的需求进行扩展。3.更完善，功能更强大。4.基于XML编写。5.支持数据类型。

## 2 作用

a DTD的局限性：

DTD不遵守xml语法（写xml文档实例时用一种语法，写DTD的时候用另外一种语法）；DTD数据类型有限（与数据库数据类型不一致）；DTD不可扩展；DTD不支持命名空间（命名冲突）。

b Schema的新特性：

Shema基于xml语法的；Shema可以用能处理xml文档的工具处理；Schema大大扩充了数据类型，可以自定义数据类型；Schema支持元素的继承——Object-Oriented;Shema支持属性组。

3 Schema（模式）

其作用与DTD一样，也是用于验证xml文档的有效性，只不过它提供了比DTD更强大的功能和更细粒度的数据类型，另外Schema还可以自定义数据类型。此外，Schema也是一个xml文件，而DTD则不是。

## 4 Schema的数据类型：

（1）内置的数据类型：

基本的数据类型；

扩展的数据类型；

（2）用户自定义简单类型（通过simpleType定义）。

## 5 Schema的元素类型：

<1>Schema根元素

作用：包含已经定义的Schema

用法：<xs:schema>

属性：xmlns/targetNamespace/elementFormDefault:

<2>element元素

作用：声明一个元素

属性：name/type/ref/minOccurs/maxOccurs

<3>group元素

作用：把一组元素声明组合在一起

属性:name

# 五 xml与xsl

XSL 指扩展样式表语言（**E**Xtensible **S**tylesheet **L**anguage），万维网联盟 (W3C) 开始发展 XSL 的原因是：存在着对于基于 XML 的样式表语言的需求。

### **XSL 包括三部分：**

**XSLT**

一种用于转换 XML 文档的语言。

**XPath**

一种用于在 XML 文档中导航的语言。

**XSL-FO**

一种用于格式化 XML 文档的语言。

# 六 xml架构

## 1 定义

XML 架构是用于定义和验证 XML 数据的内容和结构的文档，就像数据库架构定义和验证组成数据库的表、列和数据类型一样。  
XML 架构通过 XML 架构定义 (XSD) 语言定义和描述某些 XML 数据类型。XML 架构元素（元素、属性、类型和组）用于定义某些 XML 数据类型的有效结构、有效数据内容和关系。XML 架构还可为属性和元素提供默认值  
可使用 XML 架构来保证应用程序和单位之间共享的某些 XML 数据类型的一致性。XML 架构可用作两个应用程序之间进行数据交换的合同。单位可以发布描述其应用程序生成和使用的 XML 格式的架构。这样，希望交换数据的其他单位和应用程序可以围绕这些架构生成他们的应用程序，以便他们的 XML 消息能被理解。  
例如，在买方和卖方之间发送以 XML 表示的订单之前，可以用 XML 架构对其进行验证。该验证校验数据的所有元素（各片段）均存在，都按预期顺序排列，并且均为正确的数据类型。这确保订单收件人在收到它时能够正确解释数据。  
Visual Studio 中的 XML 架构符合基于 XML 架构定义语言 (XSD) 的有关 XML 架构的 W3C 建议。以下是 XML 架构相对于以前的 DTD 之类的技术的一些优越性：  
XML 架构使用 XML 语法，因此您不必学习新语法来定义数据结构。  
XML 架构支持可再次使用的类型，并允许您使用继承创建新类型。  
XML 架构使您可以对元素分组以控制元素和属性的重复。  
有关架构以及它们所代表元素约束数据的方式的更多信息，请参见 XML 的元素、属性和类型。