**XML总结**

1. **XML简介**

XML 指可扩展标记语言（EXtensible Markup Language）

XML 是一种标记语言，很类似 HTML

XML 的设计宗旨是传输数据，而非显示数据

XML 标签没有被预定义。您需要自行定义标签。

XML 被设计为具有自我描述性。

XML 是 W3C 的推荐标准

1. **XM与HTML的主要差异**

XML 不是 HTML 的替代。

XML 和 HTML 为不同的目的而设计：

XML 被设计为传输和存储数据，其焦点是数据的内容。

HTML 被设计用来显示数据，其焦点是数据的外观。

HTML 旨在显示信息，而 XML 旨在传输信息

1. **XML的用途**
2. 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。

1. 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

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。

1. 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。

1. 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。

1. XML 使您的数据更有用

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。

通过 XML，您的数据可供各种阅读设备使用（手持的计算机、语音设备、新闻阅读器等），还可以供盲人或其他残障人士使用。

1. 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

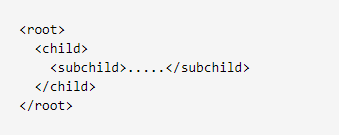
1. **XML树结构**

XML 文档形成了一种树结构，它从“根部”开始，然后扩展到“枝叶”。

XML 文档必须包含根元素。该元素是所有其他元素的父元素。

XML 文档中的元素形成了一棵文档树。这棵树从根部开始，并扩展到树的最底端。

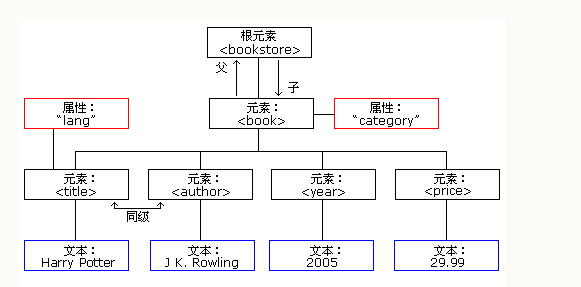
所有元素均可拥有子元素：



父、子以及同胞等术语用于描述元素之间的关系。父元素拥有子元素。相同层级上的子元素成为同胞（兄弟或姐妹）。

所有元素均可拥有文本内容和属性（类似 HTML 中）

实例：



1. **XML语法规则**

所有 XML 元素都须有关闭标签

XML 标签对大小写敏感

XML 必须正确地嵌套

XML 文档必须有根元素

XML 的属性值须加引号

在 XML 中，空格会被保留

XML 以 LF 存储换行

1. **什么是XML元素**

XML 元素指的是从（且包括）开始标签直到（且包括）

束标签的部分。

元素可包含其他元素、文本或者两者的混合物。元素也可以拥有属性。

<bookstore>

<book category="CHILDREN">

<title>Harry Potter</title>

<author>J K. Rowling</author>

<year>2005</year>

<price>29.99</price>

</book>

<book category="WEB">

<title>Learning XML</title>

<author>Erik T. Ray</author>

<year>2003</year>

<price>39.95</price>

</book>

1. **XML命名规则**

XML 元素必须遵循以下命名规则：

名称可以含字母、数字以及其他的字符

名称不能以数字或者标点符号开始

名称不能以字符 “xml”（或者 XML、Xml）开始

名称不能包含空格

1. **XML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 转换为 XML DOM 对象 - 可通过 JavaScript 操作的对象

解析 XML 文档：

方法 1：通过微软的 XML 解析器来加载 XML

方法 2：在 Firefox 及其他浏览器中的 XML 解析器

1. **XML DOM**

XML DOM (XML Document Object Model) 定义了访问和操作 XML 文档的标准方法。

DOM 把 XML 文档作为树结构来查看。能够通过 DOM 树来访问所有元素。可以修改或删除它们的内容，并创建新的元素。元素，它们的文本，以及它们的属性，都被认为是节点。

在下面的例子中，我们使用 DOM 引用从 <to> 元素中获取文本：

xmlDoc.getElementsByTagName("to")[0].childNodes[0].nodeValue

xmlDoc -由解析器创建的 XML 文档

getElementsByTagName("to")[0] - 第一个 <to> 元素

childNodes[0] - <to> 元素的第一个子元素（文本节点）

nodeValue - 节点的值（文本本身）

1. **XML CDATA**

术语 CDATA 指的是不应由 XML 解析器进行解析的文本数据（Unparsed Character Data）。

在 XML 元素中，"<" 和 "&" 是非法的。

"<" 会产生错误，因为解析器会把该字符解释为新元素的开始。

"&" 也会产生错误，因为解析器会把该字符解释为字符实体的开始。

某些文本，比如 JavaScript 代码，包含大量 "<" 或 "&" 字符。为了避免错误，可以将脚本代码定义为 CDATA。

CDATA 部分中的所有内容都会被解析器忽略。

CDATA 部分由 "<![CDATA[" 开始，由 "]]>" 结束：

<script>

<![CDATA[

function matchwo(a,b)

{

if (a < b && a < 0) then

{

return 1;

}

else

{

return 0;

}

}

]]>

</script>

关于 CDATA 部分的注释：

CDATA 部分不能包含字符串 "]]>"。也不允许嵌套的 CDATA 部分。

标记 CDATA 部分结尾的 "]]>" 不能包含空格或折行。

1. **XML DTD**

DTD为英文Document Type Definition，中文意思为“文档类型定义”。

DTD肩负着两重任务:一方面它帮助你编写合法的代码，另一方面它让浏览器正确地显示器代码。

良好的xml文档：符合xml的语法规则。

有效的xml文档：符合xml语法规则的同时还需要符合DTD文档类型定义。

利用DTD可以对xml文档的各个节点进行约束定义，使开发遵循一套“标准”。

DTD可以约束xml文档出现的元素，元素名称，元素的先后顺序，属性等。

DTD文档有四种应用形式：

1.内部DTD文档

<!DOCTYPE 根元素[定义内容]>

2.外部DTD文档

而外部DTD文档又分为两类：外部私有DTD文档，外部共有DTD文档

外部私有DTD：

<!DOCTYPE 根元素 SYSTEM "DTD文件路径">

外部共有DTD：

<!DOCTYPE 根元素 PUBLIC

"注册//组织//类型 标签//定义 语言"

"文档类型定义位置">

3.内外部DTD文档

<!DOCTYPE 根元素 SYSTEM "DTD文件路径" [定义内容]>

4.网络DTD

<!DOCTYPE 根节点 PUBLIC "DTD的名称" "DTD的地址">