**中软国际考题-笔试**

1. 单选题

1、下列说法错误的是（B）

A.xml可以作为不同系统之间传递信息的格式 B.在网页中是无法解析XML文档的

C.java提供多种解析文档的方式 D.以上说法全都正确

2、XML文件的本质是一个（B）。

A.二进制文件B.文本文件C.字节码文件D.以上都不对

3、Java属性文件的后缀名是：（A）。

A.propertiesB.txtC.无后缀名D.xml

4、下列关于XML文档中根元素的说法不正确的是（D）

A.每一个结构完整的XML文档中有且只有一个根元素

B.根元素完全包括了文档中其他所有元素

C.根元素的起始标记要放在其他所有元素的起始标记之前，而根元素的结束标记要放在其他所有元素的结束标记之后

D.根元素不能包含属性节点

5、对xml理解正确的是（B）

A. 它可以跨越互联网任何的平台，受编程语言和操作系统的限制

B. xml有助于在服务器之间穿梭结构化数据

C. xml有助于在客户端之间穿梭结构化数据

D. Xml用于标记电子文件使其具有结构性的标记语言，主要用来定义数据类型

6、 xml命名空间（xml Namespaces）（A）

A.xml命名空间提供避免元素命名冲突的方法。

B.在xml中，元素名称是由W3C定义的

C.当两个不同的文档使用相同的元素名时，不会发生命名冲突

7、对XML文档的显示央视修饰的是下面的（ A ）技术

A. XSL B. XPath C. Xlink D. XHTML

8、名字空间声明有直接定义和缺省定义二种方式,其中直接定义的格式是( A )

A、xmlns:<名字空间前缀>=”<名字空间名>”

B、xmlns:<名字空间名>=”<名字空间前缀>”

C、xmlns<名字空间前缀>=”<名字空间名>”

D、xmlns:<名字空间前缀>=<名字空间名>

9、在XML文档的第一条声明语句中，不能使用的属性是(   D   )

 A、version          B、encoding     C、standalone       D、 name

10、属性（B）用来表示XML文档所使用的字符集。

A.version   B. Encoding   C.  Standalone

11、下列说法中对 XML的描述错误的是?(C)

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

B XML 是一种标记语言，很类似 HTML

C XML 的设计宗旨是显示数据

D XML 标签没有被预定义。您需要自行定义标签。

12、XML默认的编码形式（B）

A.UTF-16 B.UTF-8 C.ASCII D.Unieode

1. 多选题
2. 下面属于XML解析技术的是？（A、B、E）
3. DOM B. SAX C. HTMLDOM
4. LOG4J E. STAX

2、以下说法正确的是：（A、B、C、D、E）。

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

B.XML 是一种很像HTML的标记语言。

C.XML 的设计宗旨是传输数据，而不是显示数据。

D.XML 标签没有被预定义。您需要自行定义标签。

E.XML 被设计为具有自我描述性。

3、对xml理解正确的是（ABCD）

A.它可以跨越互联网任何的平台，不受编程语言和操作系统的限制

B.xml是当前处理结构化文档信息中相当给力的技术

C.xml有助于在服务器之间穿梭结构化数据

D.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

4、XML元素的命名规则(ABC)

A.名称中可以包含字母、数字或者其他的字符；

B.名称不能以数字或者标点符号开始；

C.名称中不能包含空格

5、1XML声明语句：<?xml ( D )=”1.0” ( A )=”UTF-8”?>

A. encoding B. Encording C. Standalone D. version

6、Schema至少需要具有一个属性Xmlns,用以指定该Scehma包含的( ABC )

A、名字空间 B、元素作用域 C、属性作用域 D、变量

7、xml有哪些解析技术？（A、B）

A.DOM B.SAX C.BOM D.STAX

8、以下html标记中可以被引入XML中的有( ABCD )

A.各种静态标签 B.超链接

C.表单 D.脚本

9、XML 与 HTML 的主要差异有哪些？(ABCD)

A.XML 和 HTML 为不同的目的而设计：

B.XML 被设计为传输和存储数据，其焦点是数据的内容。

CHTML 被设计用来显示数据，其焦点是数据的外观。

D.HTML 旨在显示信息，而 XML 旨在传输信息。

10、每个XML文档分为两个部分？（A、B）

A.序言 B. 文档元素 C. 空元素 D.含有其他有元素的子元素

1. 填空题（每空1分，共10分）
2. 定义XML时有\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_两种定义方式。(dtd、schema)
3. XML采用\_\_\_\_\_状数据组织结构。(树)
4. XML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”>
5. XML属性值中不能够直接包含<.”,&。
6. DTD用来描述xml文档的结构，一个DTD文档包含：  元素的定义规则；元素之间的关系规则；属性的定义规则
7. XML的解析方式分为四种：\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_。（DOM解析；SAX解析；JDOM解析；DOM4J解析）
8. XML是一种\_\_\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_\_\_语言，XML具有自描述性、内容和显示相分离、可扩张、独立于平台等特点。（描述数据、数据结构）
9. 重复元素符号“+”和“\*”的区别仅在于是否出现\_\_\_\_\_\_\_\_\_\_\_次。( 0 )
10. XML文档定义有几种形式?\_\_\_\_\_\_\_\_\_\_分别是：\_\_\_\_\_\_\_\_\_\_。(2、dtd , schema)
11. 一个元素开始的分隔符被称为\_\_\_\_\_\_\_\_\_\_,一个元素结束的分隔符被称为\_\_\_\_\_\_\_\_\_\_(起始标记，结束标记)
12. XML文档定义有几种方式？解析XML文档有哪几种方式？XML文档定义方式：\_\_\_\_\_\_\_\_\_\_,\_\_\_\_\_\_\_\_，(dtd.schema模式)
13. 解析XML文档方式：\_\_\_\_\_\_,\_\_\_\_\_\_,\_\_\_\_.(DOM,SAX,STAX)
14. \_\_\_\_\_\_\_\_\_\_是解决XML元素多义性和名字冲突的方案。(命名空间)
15. 判断题
16. DOM适合解析容量很大的XML文件，检索速度快。（ F ）
17. 在实际开发中经常使用xml来配置开发属性。（ T ）
18. XML 被设计用来传输和存储数据。（T）
19. 属性类型—CDATA是表示属性的值，可以是任何字符。(T)
20. 命名空间是解决XML元素多义性和名字冲突问题的方案。（ T ）
21. Schema与DTD的扩展性是一致的。（ F ）
22. 在XML中，在Schema中，声明一个元素的属性的attribute元素有一个常用的属性use,use 的取值不包括fixed。（T）
23. 在DTD中,定义元素属性时,其缺省值是IMPLIED时,是可有可无的属性.（T）
24. 在schema中的simpleType与complexType的区别之一是complexType的元素可以含有attribute。（T）
25. XML相对于HTML的主要优点是标记是固定的。（F）
26. web.xml 的作用是配置欢迎页，servlet，filter，listener 等的。（ F ）
27. 每个文档都被分为两个部分，序文部分和文档元素 （T）
28. 树状结构使得XML文件可以有多个根元素，每个根元素可有自己的树状结构( F )
29. XPath是用于从XML文档检索元素的XML技术（ T ）
30. XML是SGM的一个子集。（ T ）

五、简答题

1、XSLT是什么?

答案：

XSLT是常用的XML技术，用于将一个XML文件转换为另一种XML，HTML或者其他的格式。

XSLT为转换XML文件详细定义了自己的语法，函数和操作符。通常由XSLT引擎完成转换，XSLT引擎读取XSLT语法编写的XML样式表或者XSL文件的指令。

XSLT大量使用递归来执行转换。一个常见XSLT使用就是将XML文件中的数据作为HTML页面显示。XSLT也可以很方便地把一种XML文件转换为另一种XML文档。

2、请简述XML命名空间是什么，以及重要性？

答案：XML命名空间与Java的package类似，用来避免不同来源名称相同的标签发生冲突。XML命名空间在XML文档顶部使用xmlns属性定义，语法为xmlns:prefix=’URI’。prefix与XML文档中实际标签一起使用

3、XML最常见的文本类型有哪些，并分别简单介绍

答案：

(1)元素：元素是 XML 的基本构件。元素可以有其它元素、文本节点或两者兼有来作为其子节点。元素节点还是可以有属性的唯一类型的节点。

(2)属性：属性节点包含关于元素节点的信息，但实际上，不认为它是元素的子节点

(3)文本：文本节点是：文本。它可以包含许多信息或仅仅是空白。

(4)文档（根节点）：文档节点是整个文档中所有其它节点的父节点。

# 4、为什么我们需要使用XML。(分别写出2条即可)（5分）

答案：

①配置文件(XML能够非常清晰描述出程序之间的关系

②程序间数据的传输，XML的格式是通用的，能够减少交换数据时的复杂性！

③XML 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

5、DOM和SAX解析器有什么区别（5分）

答案：DOM和SAX解析器的主要区别在于它们解析XML文档的方式。使用DOM解析时，XML文档以树形结构的形式加载到内存中，而SAX是事件驱动的解析器。

* 1. 对于空标记和空元素，他们的相同和不同之处在什么地方？

答案：空元素和空标记在XML解析器中的处理是相同得，也就是说他们的作用是等价的，两者的不同之处在于语法结构。

7、什么是XML？XML的特点有哪些？

答案：ML（eXtensible Markup Language，可扩展标记语言），是由万维网联盟（World Wide Web Consortium，W3C）定义的一种数据交换的规范，是Web上的数据通用语言，使用一系列简单的标记描述数据。

XML的特点有：

（1）XML是一种简单、与平台无关并被广泛采用的标准，可用于Internet上的各种应用。

（2）XML兼容SGML，所以多数SGML应用可转化为XML。

（3）XML文件同HTML文件一样易于创建。

（4）XML文件内容、结构简单，可以更加灵活地进行编程，减少了服务器的工作量。

（5）XML文件结构严谨，可以轻松被计算机程序解析。

（6）XML相对于HTML的优点是它将用户界面与结构化数据分隔开来。这种数据与显示的分离使得集成来自不同源的数据成为可能。

8、DTD与XML Schema有什么区别？（5分）

答案：DTD与XML Schema有以下区别：DTD不使用XML编写而XML Schema本身就 是xml文件，这意味着XML解析器等已有的XML工具可以用来处理XML Schema。而且XML Schema 是设计于DTD之后的，它提供了更多的类型来映射xml文件不同的数据类型。DTD即文档类型描述(Document Type definition)是定义XML文件结构的传统方式。

9、xml有哪些解析技术?区别是什么?

答案：DOM:处理大型文件时其性能下降的非常厉害。这个问题是由DOM的树结构所造成的，这种结构占用的内存较多，而且DOM必须在解析文件之前把整个文档装入内存,适合对XML的随机访问

SAX:不现于DOM,SAX是事件驱动型的XML解析方式。它顺序读取XML文件，不需要一次全部装载整个文件。当遇到像文件开头，文档结束，或者标签开头与标签结束时，它会触发一个事件，用户通过在其回调事件中写入处理代码来处理XML文件，适合对XML的顺序访问

10、简述XML与HTML的区别

答案：

（1）HTML语言是用来格式化web数据的语言，有固定的标记，每一个标记都有其固定的用法。XML允许自定义标记，用户可以自己定义标记来描述自己领域的信息。

（2）HTML的语法也相当宽松，标记不一定要配对使用，名称不区分大小写。XML语言有其严格的语法规则，标记必须成对使用，严格区分名称的大小写。

（3）不能够很好的描述数据的结构，本质上是一种格式显示语言。XML语言的特点就是将信息的内容和它们的显示样式区分开来，焦点是数据的内容。

11、你在项目中用到了xml技术的哪些方面?如何实现的?

答案：用到了数据存贮，信息配置两方面。在做数据交换平台时，将不能数据源的数据组装成XML文件，然后将XML文件压缩打包加密后通过网络传送给接收者，接收解密与解压缩后再同XML文件中还原相关信息进行处理。在做软件配置时，利用XML可以很方便的进行，软件的各种配置参数都存贮在XML文件中。

12、XML是什么？

答：XML即可扩展标记语言（Extensible Markup language），你可以根据自己的需要扩展XML。XML中可以轻松定义<books>, <orders>等自定义标签，而在HTML等其他标记语言中必须使用预定义的标签，比如<p>，而不能使用用户定义的标签。使用DTD和XML Schema标准化XML结构。XML主要用于从一个系统到另一系统的数据传输，比如企业级应用的客户端与服务端。

13：DTD与XML Schema有什么区别？

答：DTD与XML Schema有以下区别：DTD不使用XML编写而XML Schema本身就是xml文件，这意味着XML解析器等已有的XML工具可以用来处理XML Schema。而且XML Schema 是设计于DTD之后的，它提供了更多的类型来映射xml文件不同的数据类型。DTD即文档类型描述(Document Type definition)是定义XML文件结构的传统方式。

14：XPath是什么？

答：XPath是用于从XML文档检索元素的XML技术。XML文档是结构化的，因此XPath可以从XML文件定位和检索元素、属性或值。从数据检索方面来说，XPath与SQL很相似，但是它有自己的语法和规则。了解更多查看怎样使用XPath从XML文档中检索数据。

15：XSLT是什么?

答：XSLT也是常用的XML技术，用于将一个XML文件转换为另一种XML，HTML或者其他的格式。XSLT为转换XML文件详细定义了自己的语法，函数和操作符。通常由XSLT引擎完成转换，XSLT引擎读取XSLT语法编写的XML样式表或者XSL文件的指令。XSLT大量使用递归来执行转换。一个常见XSLT使用就是将XML文件中的数据作为HTML页面显示。XSLT也可以很方便地把一种XML文件转换为另一种XML文档。

16：什么是XML元素和属性

答：最好举个例子来解释。下面是简单的XML片断。

<Orders>

<Order id=”123″>

<Symbol>6758.T</Symbol>

<Price>2300</Price>

<Order>

<Orders>

例子中id是元素的一个属性，其他元素都没有属性。

17：什么是格式良好的XML

答：这个问题经常在电话面试中出现。一个格式良好的XML意味着该XML文档语法上是正确的，比如它有一个根元素，所有的开放标签合适地闭合，属性值必须加引号等等。如果一个XML不是格式良好的，那么它可能不能被各种XML解析器正确地处理和解析。

18：XML命名空间是什么？它为什么很重要？

答：XML命名空间与Java的package类似，用来避免不同来源名称相同的标签发生冲突。XML命名空间在XML文档顶部使用xmlns属性定义，语法为xmlns:prefix=’URI’。prefix与XML文档中实际标签一起使用。下面例子为XML命名空间的使用。

<root xmlns:inst=”http://instruments.com/inst”

<inst:phone>

<inst:number>837363223</inst:number>

</inst:phone>

</root>

29、什么是JSON和XML？

答：JSON和XML都是数据交换格式，JSON轻量级的数据交换格式、XML标记电子文件使其具有结构性的标记语言。

20：XML CDATA是什么

答：这道题很简单也很重要，但很多编程人员对它的了解并不深。CDATA是指字符数据，它有特殊的指令被XML解析器解析。XML解析器解析XML文档中所有的文本，比如<name>This is name of person</name>，标签的值也会被解析，因为标签值也可能包含XML标签，比如<name><firstname>First Name</firstname></name>。CDATA部分不会被XML解析器解析。CDATA部分以“

21：Java的XML数据绑定是什么

答：Java的XML绑定指从XML文件中创建类和对象，使用Java编程语言修改XML文档。XML绑定的Java API，JAXB提供了绑定XML文档和Java对象的便利方式。另一个可选的XML绑定方法是使用开源库，比如XML Beans。Java中XML绑定的一个最大的优势就是利用Java编程能力创建和修改XML文档。

由此，XML面试问答题对于XML技术的每个人都是有用的。由于XML具有平台独立的特性，XPath，XSLT，XQuery等XML技术越来越重要，XML广泛用于跨平台数据传输。尽管XML有冗余和文档体积大等缺点，但它在web服务以及带宽、速率作为次要考虑因素的系统间数据传输起很大作用。

## 22、XML是什么？

答：XML即可扩展标记语言（Extensible Markup language），你可以根据自己的需要扩展XML。XML中可以轻松定义<books>, <orders>等自定义标签，而在HTML等其他标记语言中必须使用预定义的标签，比如<p>，而不能使用用户定义的标签。使用DTD和XML Schema标准化XML结构。XML主要用于从一个系统到另一系统的数据传输，比如企业级应用的客户端与服务端。

## 23.举例说明什么是XML元素和属性

<Orders>

    <Order id="123">

        <Symbol>6758.T</Symbol>

        <Price>2300</Price>

    <Order>

<Orders>

24、什么是格式良好的XML

答：一个格式良好的XML意味着该XML文档语法上是正确的，比如它有一个根元素，所有的开放标签合适地闭合，属性值必须加引号等等。如果一个XML不是格式良好的，那么它可能不能被各种XML解析器正确地处理和解析。

25.DOM和SAX解析器有什么区别

　　答：这又是一道常见面试题，不仅出现在XML面试题中，在Java面试中也会问到。DOM和SAX解析器的主要区别在于它们解析XML文档的方式。使用DOM解析时，XML文档以树形结构的形式加载到内存中，而SAX是事件驱动的解析器。

26.Java的XML数据绑定是什么

　　答：Java的XML绑定指从XML文件中创建类和对象，使用Java编程语言修改XML文档。XML绑定的Java API，JAXB提供了绑定XML文档和Java对象的便利方式。另一个可选的XML绑定方法是使用开源库，比如XML Beans。Java中XML绑定的一个最大的优势就是利用Java编程能力创建和修改XML文档。

## 27、HTML和XML有什么区别?

答案：HTML用于显示页面，而XML用于描述页面内容的数据或数据的结构。HTML把数据和显示合在一起，在页面中把这些数据显示出来，而XML则将数据和显示分开。

## 28、SGML、HTML和XML的关系?

答案：SGML是描述电子文档的国际标准，是用于书写其他语言的语言。HTML是用SGML所定义的，即HTML是SGML的一个应用。XML是SGML的一个子集。

## 29、XML文档也属于纯文本文件，该文档的组成?

答案：XML文档分为四个部分。1）XML文档的声明 ； 2）XML处理指令 ； 3）XML文档注释 ； 4）XML标识及其内容 。

## 30、XML架构的基本特点?

答案：1）一个XML架构文档是对某一类XML文档结构和内容的抽象描述。2）XML架构本身也是XML文档，通过架构元素的嵌套和引用关系表达其所描述的XML 文档结构、取值方式。。

## 31、简述复杂元素的分类?

答案：复杂元素按内容的复杂程度可以分为两类：简单内容元素（ simpleContent ）和复杂内容元素（complexContent）。说明：1) simpleContent元素是指元素内容中仅包含文本，不含子元素的复杂元素。  注意：该元素带有属性。2) 除simpleContent外的所有复杂元素，都是complexContent元素。

六、程序实现题

1、DOM解析：使用JDOM解析xml文本中的内容

（10分）

xml文件格式如下：

|  |
| --- |
| <?xml version="1.0" encoding="gb2312"?>  <messages>  <message id="1">  <title>11</title>  <content>  <name>lvpingyu</name>  <age>23</age>  </content>  <email>33</email>  </message>  <message id="2">  <title>44</title>  <content>  <name>lvpingyu</name>  <age>23</age>  </content>  <email>66</email>  </message>  </messages> |

|  |
| --- |
| import java.io.FileInputStream;  import java.util.List;  import org.jdom2.Document;  import org.jdom2.Element;  import org.jdom2.input.SAXBuilder;  public class Jdom {  public static void main(String[] args) throws Exception {  SAXBuilder sb = new SAXBuilder();//建立构造器  String path="D://abc.xml";  Document doc = sb.build(new FileInputStream(path));//读入指定文件  Element root = doc.getRootElement();//获得根节点  List list = root.getChildren();//将根节点下的所有子节点放入List中  for (int i = 0; i < list.size(); i++) {  System.out.println("---------------------------");  Element item = (Element) list.get(i);//取得节点实例  String id = item.getAttribute("id").getValue();//取得属性值  System.out.println("id-->" + id);  Element sub = item.getChild("title");//取得当前节点的子节点  String text = sub.getText();//取得当前节点的值  System.out.println("Title-->" + text);  if (item.getChild("content").getChildren().size() > 0) {  Element sub2 = item.getChild("content").getChild("name");  String text2 = sub2.getText();  System.out.println("name-->" + text2);  }  Element sub3 = item.getChild("email");  String text3 = sub3.getText();  System.out.println("Email-->" + text3);    }  System.out.println("---------------------------");  }  } |

2、有如下XML文件（src/ahjava/config.xml），编码读取出class的值。

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <beans>  <bean id="id1" class="com.Cat">  <property name="id" value="1" />  <property name="username" value="英短" />  </bean>  </beans> |

参考答案：

|  |
| --- |
| import java.io.\*;  import java.lang.reflect.Constructor;  import javax.xml.parsers.DocumentBuilder;  import javax.xml.parsers.DocumentBuilderFactory;  import org.w3c.dom.Document;  import org.w3c.dom.Element;  import org.w3c.dom.Node;  import org.w3c.dom.NodeList;  public class Main {  public static void main(String args[]) {  Element element = null;  // 可以使用绝对路劲  File f = new File("src/ahjava/config.xml");  // documentBuilder为抽象不能直接实例化(将XML文件转换为DOM文件)  DocumentBuilder db = null;  DocumentBuilderFactory dbf = null;  try {  dbf = DocumentBuilderFactory.newInstance();  db = dbf.newDocumentBuilder();  Document dt = db.parse(f);// 得到一个DOM  element = dt.getDocumentElement();  // 根元素下的所有子节点  NodeList childNodes = element.getChildNodes();  // 遍历这些子节点  for (int i = 0; i < childNodes.getLength(); i++) {  // 获得每个对应位置i的结点  Node node1 = childNodes.item(i);  if ("bean".equals(node1.getNodeName())) {  String sClass = node1.getAttributes().getNamedItem("class").getNodeValue();  System.out.println(sClass);  }  }  } catch (Exception e) {  e.printStackTrace();  }  }  } |

## 3、XML解析操作

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

<bookstore>

<book category="COOKING">

<author>Rose</author>

<year>2005</year>

<price>80.00</price>

</book>

<book category="CHILDREN">

<author>David</author>

<year>2018</year>

<price>25.3</price>

</book>

</bookstore>

DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

Document document = dBuilder.parse("src/books.xml");

//找到根节点

Element root = document.getDocumentElement();

System.out.println("------开始解析-------");

NodeList booksList = root.getElementsByTagName("book");

System.out.println("一共有" + booksList.getLength() + "本书");

//遍历book节点的所有节点

for (int i = 0; i < booksList.getLength(); i++){

book = new Books();

System.out.println("-----现在开始遍历第" +(i + 1) + "本书的内容-----");

//通过item(i)方法获取一个book节点

Node nodeBook = booksList.item(i);

//获取book节点的所有属性集合

NamedNodeMap attrs = nodeBook.getAttributes();

System.out.println("第" + (i + 1) + "本书有"+ attrs.getLength() + "个属性");

//遍历book属性

for (int j = 0; j < attrs.getLength(); j++){

//通过item(index)方法获取book节点的某一个属性

Node attr = attrs.item(j);

//获取属性名和属性值

System.out.println("属性名：" + attr.getNodeName() + "， 属性值：" + attr.getNodeValue());

book.setCategory(attr.getNodeValue());

}

4、使用DOM解析，解析如下XML文件，得到书籍列表。

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

<bookstore>

<book category="children">

<title lang="en">Harry Potter</title>

<author>J K. Rowling</author>

<year>2005</year>

<price>29.99</price>

</book>

<book category="cooking">

<title lang="en">Everyday Italian</title>

<author>Giada De Laurentiis</author>

<year>2005</year>

<price>30.00</price>

</book>

<book category="web">

<title lang="en">Learning XML</title>

<author>Erik T. Ray</author>

<year>2003</year>

<price>39.95</price>

</book>

<book category="web">

<title lang="en">XQuery Kick Start</title>

<author>James McGovern</author>

<year>2003</year>

<price>49.99</price>

</book>

</bookstore>

（10分）

|  |
| --- |
| public class DOMParse {  public static void main(String[] args) throws Exception {  List<Book> books = new ArrayList<Book>();    //获得DOM解析器工厂  DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();    //创建DOM解析器  DocumentBuilder db = dbf.newDocumentBuilder();    //解析xml，获取到Document对象  Document doc = db.parse(new FileInputStream("books.xml"));    Element root = doc.getDocumentElement();    NodeList nodes = root.getElementsByTagName("book");//就是所有book节点  System.out.println(nodes.getLength());    for(int i=0;i<nodes.getLength();i++){  Book book = new Book();  Element element = (Element)nodes.item(i);  book.setCategory(element.getAttribute("category"));    NodeList childs = element.getChildNodes();  for(int j=0;j<childs.getLength();j++){  if(childs.item(j).getNodeType()==Node.ELEMENT\_NODE){  Element child = (Element)childs.item(j);  if("title".equals(child.getNodeName())){  book.setTitle(child.getTextContent());  }else if("author".equals(child.getNodeName())){  book.setAuthor(child.getTextContent());  }  }  }  books.add(book);  }    for(Book b:books){  System.out.println(b);  }    }  } |

5、根据需求完成以下要求。

（10分）

下面有一些DTD的语句，将它转化为Schema语言。

<?xmlversion=1.0 encoding=”UTF-8”?>

<!ELEMENT book(title,author,price)>

<!ELEMENT title (#PCDATA)>

<!ELEMENT author(#PCDATA)>

<!ELEMENT price(#PCDATA)>

答案：

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

<xsd:schemaxmlns:xsd=http://www.w3.org/2001/XMLschema>

<xsd:element name=”book” type=”BookType”/>

<xsd:complexType name=”BookType”>

<xsd:sequnce>

<xsd:element name=”title” type=”xsd:string”/>

<xsd:element name=”author” type=”xsd:string”/>

<xsd:element name=”price” type=”xsd:decimal”/>

</xsd:sequnce>

</xsd:compexType>

</xsd:schema>

6、 下面XML中有几处错误，请找出，并改正。

<!--一个简单的XML文件-->

<? xml version=”1.1” ?>

<fruits>

<fruit>

<name>orange<price>

</name>1.2</price>

</fruit>

<fruit>

<name>banana<price>

</name>1.8</price>

</fruit>

</Fruits>

答案：

有5处错误：注释位置不正确；声明中xml前不能有空格；版本号应为1.0；<name>与<price>标记出现了交叉；根标记不匹配。正确的文件：

<?xml version=”1.0” ?>

<!--一个简单的XML文件-->

<fruits>

<fruit>

<name>orange</name>

<price>1.2</price>

</fruit>

<fruit>

<name>banana</name>

<price>1.8</price>

</fruit>

</fruits>

7、用jdom解析xml文件时如何解决中文问题?如何解析?.（10分）

|  |
| --- |
| package test;  import java.io.\*;  public class DOMTest  {  private String inFile = "c:\\people.xml"  private String outFile = "c:\\people.xml"  public static void main(String args[])  {  new DOMTest();  }  public DOMTest()  {  try  {  javax.xml.parsers.DocumentBuilder builder =  javax.xml.parsers.DocumentBuilderFactory.newInstance().newDocumentBuilder();  org.w3c.dom.Document doc = builder.newDocument();  org.w3c.dom.Element root = doc.createElement("老师");  org.w3c.dom.Element wang = doc.createElement("王");  org.w3c.dom.Element liu = doc.createElement("刘");  wang.appendChild(doc.createTextNode("我是王老师"));  root.appendChild(wang);  doc.appendChild(root);  javax.xml.transform.Transformer transformer =  javax.xml.transform.TransformerFactory.newInstance().newTransformer();  transformer.setOutputProperty(javax.xml.transform.OutputKeys.ENCODING, "gb2312");  transformer.setOutputProperty(javax.xml.transform.OutputKeys.INDENT, "yes");  transformer.transform(new javax.xml.transform.dom.DOMSource(doc),  new  javax.xml.transform.stream.StreamResult(outFile));  }  catch (Exception e)  {  System.out.println (e.getMessage());  }  }  } |

8、实现新增用户信息，存储到用户的xml文件中

users.xml文件如下：

<?xml version="1.0" encoding="UTF-8" standalone="no"?><users>

<users>

<user id="111">

<pwd>111</pwd>

<name>张三</name>

</user>

<user id="222">

<pwd>222</pwd>

<name>李四</name>

</user>

<user id="333">

<pwd>333</pwd>

<name>王柳</name>

</user>

</users>

（10分）

|  |
| --- |
| 答案：  public class BaseXml {  /\*\*  \* 保存xml文件  \* @param doc 文档  \* @param path 路径  \* @throws Exception  \*/  public void saveXml(Document doc, String path) throws Exception { // 把dom树对象与目标xml文件作为参数  TransformerFactory transformerFactory = TransformerFactory  .newInstance();  Transformer transformer = transformerFactory.newTransformer();  DOMSource domSource = new DOMSource(doc);  transformer.setOutputProperty(OutputKeys.ENCODING, "UTF-8"); // 设置编码类型  StreamResult result = new StreamResult(new FileOutputStream(path));  transformer.transform(domSource, result); // 把DOM树转换为XML文件  }  }  public class UserXmlAction extends BaseXml{  public boolean saveUser(Users user) {  try {  Document doc = this.getDoc("users.xml");  //得到所有user标签  NodeList list = doc.getElementsByTagName("user");  for(int i=0;i<list.getLength();i++){  Node item = list.item(i);//得到集合里的每一个子节点  Element ele = (Element) item;  //得到user标签的id属性值  String id = ele.getAttribute("id");  if(user.getId().equals(id)){  System.out.println("帐号重复");  return false;  }  }  //没有重复id  //创建user标签  Element eleuser = doc.createElement("user");  //为user标签赋属性值---属性名，属性值  eleuser.setAttribute("id", user.getId());  //<user id=""></user>      //创建pwd标签  Element elepwd = doc.createElement("pwd");  //为pwd标签赋内容值  elepwd.setTextContent(user.getPwd());  //<pwd>值</pwd>    //创建pwd标签  Element elename = doc.createElement("name");  //为pwd标签赋内容值  elename.setTextContent(user.getName());  //<name>值</name>    //把pwd标签放入到user标签中  eleuser.appendChild(elepwd);  eleuser.appendChild(elename);  // <user id=""><pwd></pwd></user>    //把user标签放入到根标签users下  Element root = doc.getDocumentElement();  root.appendChild(eleuser);      this.saveXml(doc, "users.xml");  return true;  } catch (Exception e) {  // TODO Auto-generated catch block  e.printStackTrace();  }  return false;  }  } |

9、什么是XML元素和属性

（10分）

|  |
| --- |
| <Orders>、  　　<Order id="123">  　　<Symbol>6758.T</Symbol>  　　<Price>2300</Price>  　　<Order>  　　<Orders>  例子中id是元素的一个属性，其他元素都没有属性。 |

10、完成属性文件写入和读取用户名和密码信息。

|  |
| --- |
| public class TestProperties {  public static void main(String[] args) {  File file = new File(" test.properties ");//后缀名  if(!file.exists()){  file.createNewFile();  }  //属性写入文件  FileOutputStream fos = new FileOutputStream(file,true);  Properties props = new Properties();  props.setProperty("name", "zhangsan");  //通过流保存写入属性  props.store(fos, null);  fos.close();  //读取文件属性  FileInputStream fis = new FileInputStream(file);  props.load(fis); //加载流以便读取属性  String name = props.getProperty("name");  String password = props.getProperty("password");  System.out.println("name="+name);  System.out.println("password="+password);  fis.close();  }  } |