# **Xml**

eXtendsible  markup language  可扩展的标记语言

# **Xml作用**

1.保存数据

2.配置文件

3.\*\*\*\*\*\*数据传输载体

# **定义xml**

一个后缀名为.xml的文件

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

version:解析这个.xml文档时候解析器的版本

encoding:解析器的解析方式

standalone:文档是否独立

# **元素定义**

1.其实就是里面的标签，<>起来都叫元素，如<stu></stu>，一般成对出现。

2.文档声明下来第一个元素叫做根元素。

3.标签里面可以镶嵌标签。

4.空标签，既是开始也是结束，一般配合属性来使用。<hr/>

5.标签可以自定义

    标签命名规则：

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

       命名尽量简单，做到见名知义

# **简单元素/复杂元素**

简单元素里面包含了普通文字，复杂元素里面可以嵌套其他元素

# **属性的定义**

定义再元素里面， <元素名称  属性名称="属性的值"></元素名称>

<students>

    <student id="001">

        <name>张无忌</name>

        <age>20</age>

    </student>

</students>

# **xml注释**

与html的注释一样 <!-- -->

\*\*\*\*\*\*需要放在声明下方

# **CDATA区**

\*非法字符

严格地讲，在 XML 中仅有字符 "<"和"&" 是非法的。省略号、引号和大于号是合法的，但是把它们替换为实体引用是个好的习惯。

  <   &lt;  
  &   &amp

\*CDATA

如果某段字符串里面有过多的字符， 并且里面包含了类似标签或者关键字的这种文字，不想让xml的解析器去解析。 那么可以使用CDATA来包装。  不过这个CDATA 一般比较少看到。 通常在服务器给客户端返回数据的时候。

    <des><![CDATA[<a href="http://www.baidu.com">点我跳转到百度</a>]]></des>

# **xml解析**

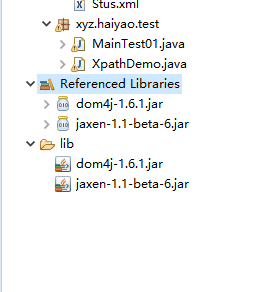
有很多中解析方式，但是最常用的为下面两种

1.\*\*\*\*\*\*DOM:document object model整个xml文档读取到内存中，形成树状结构。整个文档称为Document，元素为element,属性为attribute,字符内容则为文本数据text,这四个对象统称为node节点，如果xml特别大，将会造成内存溢出。可以对文档经行增删的操作

2.SAX:simple API for Xml 基于事件驱动。读取一行，解析一行。不会造成内存溢出，不可以经行增删，只能查询。

# **dom4j基本用法**

1.导入dom4j的包



1. 引入xml文件，例如一个两个简单的学生信息

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

<stus>

<stu>

    <name>张无忌</name>

    <age>20</age>

</stu>

<stu>

    <name>周芷若</name>

    <age>20</age>

</stu>

</stus>

1. 再java中操作xml

import java.io.File;

import java.util.List;

import javax.sound.sampled.LineListener;

import org.dom4j.Document;

import org.dom4j.DocumentException;

import org.dom4j.Element;

import org.dom4j.io.SAXReader;

public class MainTest01 {

    public static void main(String[] args) throws DocumentException {

        //1.创建sax读取对象

        SAXReader reader = new SAXReader();

        //2.读取xml文件

        Document document = reader.read(new File("src/xml/Stus.xml"));

        //3.//得到元素

        System.out.println(document.getName());  //得到xml文件的目录

        System.out.println(document.getRootElement().getName());  //得到根元素

        Element rootElement = document.getRootElement();

        //4.根据根元素获取下面的子元素或者子孙元素

        System.out.println(rootElement.element("stu").element("name").getStringValue());//得到第一个学生的姓名

        List<Element> elements = rootElement.elements(); //获得根元素下的所有子元素

        System.out.println(elements.size());

        for (Element element : elements) {

            System.out.println(element.getName());

            String name = element.element("name").getText();

            String age = element.element("age").getText();

            System.out.println("姓名："+name+"年龄"+age);

        }

    }

}

# **dom4j的xpath用法**

# IMG_256

1e /\*

\* 1.添加架包依赖\* 2.再查找节点的时候\*\*/

public class XpathDemo {

public static void main(String[] args)throws DocumentException{

//1.创建sax读取对象

SAXReader reader = new SAXReader();

Document document=reader.read(new File("src/xml/Stus.xml"));//3.要使用Xpath，还需要添加支持的jar

Element rootElement = document.getRootElement();

Element nameElement=(Element)rootElement.selectSingleNode("//name"); System.out.println(nameElement.getText());

//获取文档中所有的name元素

List<Element>list = rootElement.selectNodes("//name");//打印

for (Element element : list){ System.out.println(element.getText());

Problems @Javadoc 图 Dedaraton日 conole × 国Coverage

sterminted>XpatDemo UavaAppliaton DNabJavsSEdkbinNivaw.exe2019年4月1日下午34602）

WARNING: Please consider reporting this to the maintainers of org.dom4j.io.SAXContentHandler WARNING: Use --illegal-access=warn toenable warnings of further illegal reflective access operations WARNING: All illegal access operations willbe denied in a future release

张无忌

张无忌

周芷若

# **xml中的约束**

1.引入网上的DTD来约束这个xml

<!--    文档类型  根标签名字 网络上的dtd   dtd的名称   dtd的路径  
       <!DOCTYPE stus PUBLIC "//UNKNOWN/" "unknown.dtd"> -->

2.引入本地的DTD

      <!-- 引入本地的DTD  ： 根标签名字 引入本地的DTD  dtd的位置 -->  
      <!-- <!DOCTYPE stus SYSTEM "stus.dtd"> -->

3.直接再xml中嵌入约束规则

       <!-- xml文档里面直接嵌入DTD的约束法则 -->  
       <!DOCTYPE stus [  
           <!ELEMENT stus (stu)>  
           <!ELEMENT stu (name,age)>  
           <!ELEMENT name (#PCDATA)>  
           <!ELEMENT age (#PCDATA)>  
       ]>

        <!ELEMENT stus (stu)>  : stus 下面有一个元素 stu  ， 但是只有一个  
        <!ELEMENT stu (name , age)>  stu下面有两个元素 name  ,age  顺序必须name-age  
        <!ELEMENT name (#PCDATA)>   
        <!ELEMENT age (#PCDATA)>  
        <!ATTLIST stu id CDATA #IMPLIED> stu有一个属性 文本类型， 该属性可有可无

        元素的个数：

            ＋　一个或多个  
            \*  零个或多个  
            ? 零个或一个  
      
        属性的类型定义   
      
            CDATA : 属性是普通文字  
            ID : 属性的值必须唯一

        <!ELEMENT stu (name , age)>        按照顺序来

        <!ELEMENT stu (name | age)>   两个中只能包含一个子元素