<http://javarevisited.blogspot.com.by/2011/12/parse-xml-file-in-java-example-tutorial.html>

How to Parse or Read XML File in Java >> XML Tutorial Example

How to parse xml file in Java or **how to read xml file in java** is one of common need of a Java Developer working with enterprise Java application which uses XML for data representation, messaging and data transfer. Java has good support to handle XML files and XML Documents and you can **read XML File in Java**, **create or** **write to XML file in Java** by using various **XML parsers** available. Reading XML file is little bit different than [reading text or binary file in Java](http://javarevisited.blogspot.com/2011/12/read-and-write-text-file-java.html) but it uses same concept of File class.

[how to read or parse xml file in java](http://javarevisited.blogspot.com/2011/10/how-substring-in-java-works.html)Universal acceptability of XML and Java has helped them to grow together and they have lot of things common in between just like *Java is platform independence*, XML provides data which is also platform independent. You can use XML to transfer data between a legacy application written in C or C++ and Java.

What is important to work with XML in Java is correct understanding of XML Parser, Basic knowledge of XML document etc. In this Java XML Tutorial we will see *how to parse and XML File* by using both DOM  XML Parser. We will also see [**difference between DOM and SAX parser in XML**](http://javarevisited.blogspot.com/2011/12/difference-between-dom-and-sax-parsers.html) and other basics related to XML parsing in Java. I thought about this article after sharing my [xpath notes in Java](http://javarevisited.blogspot.com/2011/11/xpath-tutorials-examples-for-beginners.html).

## How to read XML File in Java

### JAXP - Java API for XML Parsing

Java provides extensive support for reading XML file, writing XML file and accessing any element from

XML file. All XML parsing related classes and methods are inside JAXP. Though DOM related code comes from org.w3c.dom package. All XML parsers are in javax.xml.parsers package.we will see example of parsing xml files using JAXP API in next section.

### Parse XML File in Java using DOM Parser

In this section we will see **how to parse xml files** or **how to read xml file using DOM XML Parser**.DOM is quick and easy way to parse xml files in Java and if you are doing it for testing its way to go. Only thing to concern is that XML files which need to be parsed must not be too large. You can also *create xml file by using DOM parser* and DocumentFactory [Class in Java](http://javarevisited.blogspot.com/2011/10/class-in-java-programming-general.html).

**XML file for parsing in Java**

Here is xml file Stocks.xml which contains some stocks and there price, quantity we will use this in our xml parsing example in Java.

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

<stocks>

       <stock>

              <symbol>Citibank</symbol>

              <price>100</price>

              <quantity>1000</quantity>

       </stock>

       <stock>

              <symbol>Axis bank</symbol>

              <price>90</price>

              <quantity>2000</quantity>

       </stock>

</stocks>

**Code Example of Parsing XML File in Java using DOM Parser**

Here is a code example of parsing above xml file in Java using DOM parser:

**import** java.io.File;

**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** DOMExampleJava {

**public** **static** **void** main(String args[]) {

**try** {

File stocks = **new** File("Stocks.xml");

DocumentBuilderFactory dbFactory = DocumentBuilderFactory.*newInstance*();

DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();

Document doc = dBuilder.parse(stocks);

doc.getDocumentElement().normalize();

System.*out*.println("root of xml file" + doc.getDocumentElement().getNodeName());

NodeList nodes = doc.getElementsByTagName("stock");

System.*out*.println("==========================");

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

Node node = nodes.item(i);

**if** (node.getNodeType() == Node.*ELEMENT\_NODE*) {

Element element = (Element) node;

System.*out*.println("Stock Symbol: " + *getValue*("symbol", element));

System.*out*.println("Stock Price: " + *getValue*("price", element));

System.*out*.println("Stock Quantity: " + *getValue*("quantity", element));

}

}

} **catch** (Exception ex) {

ex.printStackTrace();

}

}

**private** **static** String getValue(String tag, Element element) {

NodeList nodes = element.getElementsByTagName(tag).item(0).getChildNodes();

Node node = (Node) nodes.item(0);

**return** node.getNodeValue();

}

}

**Output:**

root of xml file stocks

==========================

Stock Symbol: Citibank

Stock Price: 100

Stock Quantity: 1000

Stock Symbol: Axis bank

Stock Price: 90

Stock Quantity: 2000

That’s all on **xml parsing in java** for now. We have seen how to read and write xml file in Java and now familiar with both **DOM and SAX Parser in java**. We will see more on xml on coming days like reading xml elements via xpath and using xml beans etc. let me know if you have any doubt on xml parsing example or in general with xml and Java.

**Related Java Tutorials**

[How to create File and Directory in Java](http://javarevisited.blogspot.com/2011/12/create-file-directory-java-example.html)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<http://javarevisited.blogspot.com.by/2015/07/how-to-read-xml-file-as-string-in-java-example.html>

How to Read XML File as String in Java? 3 Examples

Suppose you have an XML file and you just want to read and display the whole file as String in Java, may be for debugging purpose. If you are wondering how to do that in Java, well there are many ways to read XML as String in Java. You can do it in one line if you are fine with using an open source library or you can do it in couple of lines of code in core Java as well. Since an XML file is also a file, you can use BufferedReader or FileInputStream to read content of an XML file as String by using the techniques, I have discussed in my earlier post [3 ways to convert InputStream to String in Java](http://java67.blogspot.sg/2014/05/3-examples-to-read-inputstream-as-String-Java-Guava-Commons.html). But this post is about a new library called jcabi-xml which makes working with XML file really easy. You can parse XML file using XPath expression, you can do XSL transformation, XSD schema validation and you can even parse whole XML file as String in just couple of lines of code. I was playing with this new XML library and thought to share couple of examples of XML processing to show how powerful it is. By the way, while reading XML file as String one thing you must remember is character encoding, which is specified in header of XML file. If you use any XML library or even XML parser like DOM and SAX, you don't need to make any additional adjustment, they will take care of reading String in correct encoding, but if you use BufferedReader or any general Stream reader, you must ensure that correct character encoding is used. In this article, you will learn three ways to read XML file as String in Java, first by using FileReader and BufferedReader, second by using DOM parser and third by using open source XML library jcabi-xml. BTW, if you are new to XML processing in Java, I also suggest you to take a look at [Core Java Volume II - Advanced Features, 9th Edition](http://www.amazon.com/Volume-II-Advanced-Features-Edition-Series/dp/013708160X?tag=javamysqlanta-20)  by Cay S. Horstmann. It will help you to learn and understand more advanced features of Java e.g. JDBC, XML, JMX, JMS etc.

## 3 Ways to Read XML into String in Java

There are multiple ways to read and process XML file and generate String out of it and we will see three of them. Two of those approach are based upon standard classes and interfaces from JDK itself and 3rd approach will make use of open source library. For our example purpose, we will read following XML file as String :

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

<banks>

<bank id="1">

<name>Barclays Bank</name>

<headquarter>London</headquarter>

</bank>

<bank id="2">

<name>Goldman Sachs</name>

<headquarter>NewYork</headquarter>

</bank>

<bank id="3">

<name>ICBC</name>

<headquarter>Beijing</headquarter>

</bank>

</banks>

## Java Program to read XML into String

Here is our Java program to read XML as String. It contains three example, first one uses BufferedReader and read XML like a text file. It's not great because you need to specify character encoding by yourself, while XML parser can read it directly from XML header. In this approach, you can build XML string by [reading file line by line](http://javarevisited.blogspot.sg/2012/07/read-file-line-by-line-java-example-scanner.html).  
  
Second example is about DOM parser, which is great for reading small XML files because it load them entirely in memory and then parse it by creating a DOM tree. It's good for parsing huge XML file because that would require large memory and still may end up in [java.lang.OutOfMemoryError : Java heap space](http://java67.blogspot.sg/2013/08/guide-of-javalangoutofmemoryerror-java-heap-space-tomcat-eclipse-minecraft-jboss.html).  Interesting point is, toString() method of Document object will not return String representation of XML, which means this is not suitable for job in hand but you can do a lot by calling various method of DOM API.  
  
Third example is interesting, it uses an open source library called [jcabi-xml](http://www.jcabi.com/), which provides a convenient class called XMLDocument to represent an XML file in memory. This class has overridden toString() method to return String representation of XML file itself. So you can read entire XML as String by using just two lines.  
  
Here is our sample Java program to demonstrate all three methods :

import java.io.BufferedReader;

import java.io.File;

import java.io.FileReader;

import java.io.IOException;

import java.io.Reader;

import javax.xml.parsers.DocumentBuilder;

import javax.xml.parsers.DocumentBuilderFactory;

import javax.xml.parsers.ParserConfigurationException;

import org.w3c.dom.Document;

import org.xml.sax.SAXException;

import com.jcabi.xml.XML;

import com.jcabi.xml.XMLDocument;

/\*\*

\* Java Program to read XML as String using BufferedReader, DOM parser and jCabi-xml

\* open source library.

\*/

public class XmlAsStringInJava {

public static void main(String[] args) throws ParserConfigurationException, SAXException, IOException {

// our XML file for this example

File xmlFile = new File("info.xml");

// Let's get XML file as String using BufferedReader

// FileReader uses platform's default character encoding

// if you need to specify a different encoding, use InputStreamReader

Reader fileReader = new FileReader(xmlFile);

BufferedReader bufReader = new BufferedReader(fileReader);

StringBuilder sb = new StringBuilder();

String line = bufReader.readLine();

while( line != null){

sb.append(line).append("\n");

line = bufReader.readLine();

}

String xml2String = sb.toString();

System.out.println("XML to String using BufferedReader : ");

System.out.println(xml2String);

bufReader.close();

// parsing XML file to get as String using DOM Parser

DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();

DocumentBuilder docBuilder = dbFactory.newDocumentBuilder();

Document xmlDom = docBuilder.parse(xmlFile);

String xmlAsString = xmlDom.toString(); // this will not print what you want

System.out.println("XML as String using DOM Parser : ");

System.out.println(xmlAsString);

// Reading XML as String using jCabi library

XML xml = new XMLDocument(new File("info.xml"));

String xmlString = xml.toString();

System.out.println("XML as String using JCabi library : " );

System.out.println(xmlString);

}

}  
  
Output  
XML to String using BufferedReader :

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

<banks>

<bank id="1">

<name>Barclays Bank</name>

<headquarter>London</headquarter>

</bank>

<bank id="2">

<name>Goldman Sachs</name>

<headquarter>NewYork</headquarter>

</bank>

<bank id="3">

<name>ICBC</name>

<headquarter>Beijing</headquarter>

</bank>

</banks>

XML as String using DOM Parser :  
[#document: null]  
  
XML as String using JCabi library :

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

<banks>

<bank id="1">

<name>Barclays Bank</name>

<headquarter>London</headquarter>

</bank>

<bank id="2">

<name>Goldman Sachs</name>

<headquarter>NewYork</headquarter>

</bank>

<bank id="3">

<name>ICBC</name>

<headquarter>Beijing</headquarter>

</bank>

</banks>

That's all about **how to read XML file as String in Java**. You have seen all three approaches to get XML as String and you can compare them easily. BufferedReader doesn't take XML encoding defined in header, you have to specify it manually if you want to read an XML file encoded in a different character encoding. In our example, since we use [FileReader](http://javarevisited.blogspot.sg/2014/04/difference-between-fileinputstream-and-filereader-in-java.html), we don't have that option, but if you need to specify a different encoding then platform's default character encoding then please use InputStreamReader. Also when we use BufferedReader you also need to take are of new line, which could be different in different platform e.g. UNIX and Windows uses different characters for new line. DOM parser is the right way to parse XML files but unfortunately its toString() doesn't return what you want. Now if you look at our two line code using new XML library **jcabi-xml**, its a breeze. That's why if you can use open source library, use this one, it will make your XML parsing, validation and transformation easy.

If you like this Java XML tutorial and interested to learn more about XML processing in Java, you can also check out following tutorials :

* How to convert Java object to XML document using JAXB? [[example](http://javarevisited.blogspot.sg/2015/06/how-to-converts-java-object-to-xml-jaxb.html" \t "_blank)]
* Java guide to parse XML file using DOM parser? [[example](http://javarevisited.blogspot.sg/2011/12/parse-xml-file-in-java-example-tutorial.html" \t "_blank)]
* Step by Step guide to read XML using SAX parser in Java? [[guide](http://javarevisited.blogspot.sg/2011/12/parse-read-xml-file-java-sax-parser.html" \t "_blank)]
* How to select XML element using XPath in Java? [[solution](http://javarevisited.blogspot.sg/2012/12/xpath-tutorial-example-how-to-select-elements.html" \t "_blank)]
* Difference between DOM and SAX parser in XML? [[answer](http://java67.blogspot.sg/2012/09/dom-vs-sax-parser-in-java-xml-parsing.html" \t "_blank)]
* How to escape XML special characters in Java String? [[answer](http://javarevisited.blogspot.sg/2012/09/how-to-replace-escape-xml-special-characters-java-string.html" \t "_blank)]
* How to marshall and unmarshall Java to XML using JAXB? [[example](http://javarevisited.blogspot.sg/2013/01/jaxb-xml-binding-tutorial-marshalling-unmarshalling-java-object-xml.html" \t "_blank)]
* XPath Tutorials for Beginner Java Developers [[tutorial](http://javarevisited.blogspot.sg/2011/11/xpath-tutorials-examples-for-beginners.html)]
* How to process XML file using JDOM parser in Java? [[example](http://java67.blogspot.sg/2012/10/jdom-example-reading-and-parsing-xml-java.html" \t "_blank)]
* How to evaluate XPath expression in Java? [[example](http://javarevisited.blogspot.sg/2012/12/create-and-evaluate-xpath-java-example-tutorial-program.html" \t "_blank)]

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*