

JAVA DOM PARSER - OVERVIEW

http://www.tutorialspoint.com/java_xml/java_dom_parser.htm

Copyright © tutorialspoint.com

The Document Object Model is an official recommendation of the World Wide Web Consortium W3C. It defines an interface that enables programs to access and update the style, structure, and contents of XML documents. XML parsers that support the DOM implement that interface.

When to use?

You should use a DOM parser when:

- You need to know a lot about the structure of a document
- You need to move parts of the document around *you might want to sort certain elements, for example*
- You need to use the information in the document more than once

What you get?

When you parse an XML document with a DOM parser, you get back a tree structure that contains all of the elements of your document. The DOM provides a variety of functions you can use to examine the contents and structure of the document.

Advantages

The DOM is a common interface for manipulating document structures. One of its design goals is that Java code written for one DOM-compliant parser should run on any other DOM-compliant parser without changes.

DOM interfaces

The DOM defines several Java interfaces. Here are the most common interfaces:

- **Node** - The base datatype of the DOM.
- **Element** - The vast majority of the objects you'll deal with are Elements.
- **Attr** Represents an attribute of an element.
- **Text** The actual content of an Element or Attr.
- **Document** Represents the entire XML document. A Document object is often referred to as a DOM tree.

Common DOM methods

When you are working with the DOM, there are several methods you'll use often:

- **Document.getDocumentElement** - Returns the root element of the document.
- **Node.getFirstChild** - Returns the first child of a given Node.
- **Node.getLastChild** - Returns the last child of a given Node.
- **Node.getNextSibling** - These methods return the next sibling of a given Node.
- **Node.getPreviousSibling** - These methods return the previous sibling of a given Node.
- **Node.getAttribute** *attrName* - For a given Node, returns the attribute with the requested name