# **Convert To JSON**

**Aim:** converting XML format to JSON format

**Input:** XML format as a string

**Output:** JSON format as a string

### Overall

## Background

JSON (JavaScript Object Notation) is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute-value pairs and arrays (or other serializable values). It is a common data format with a diverse range of functionality in data interchange including communication of web applications with servers. JSON is a language-independent data format.

## Main Idea

First, we loop on the XML string and convert it to a tree data structure. Each tag is represented as a node in this tree which has tag name, attributes, and content. Then we use recursion to print tags and their value as it is in JSON format.

## Main steps in Huffman Compression

1. Build an XML Tree from input string
2. Traverse the XML Tree and print it in JSON format.

### Steps to build XML Tree

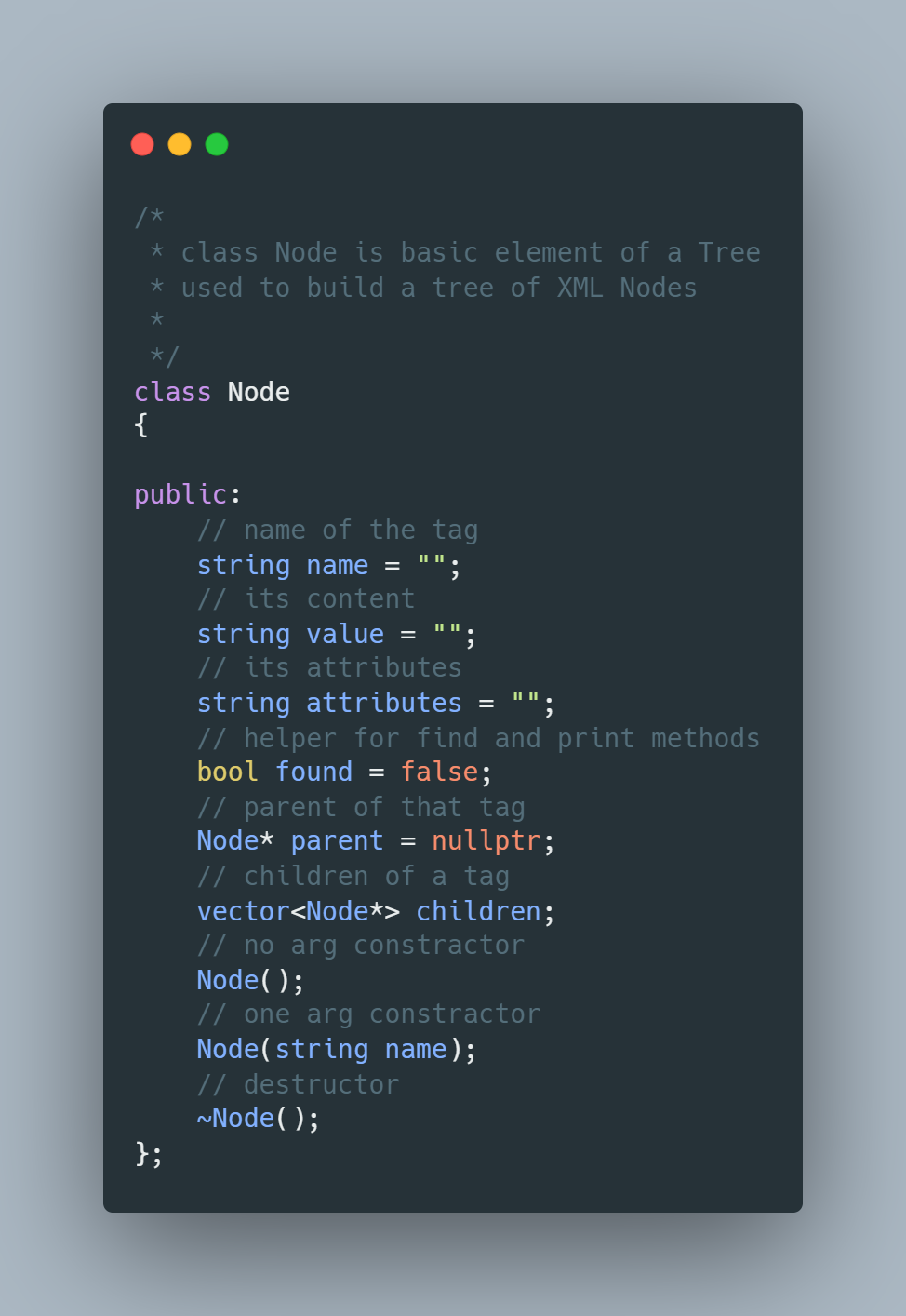
1. We loop until we find an opening tag, attach it to the root node.
2. Search about nested tags if it has we update current root to this node that we are having now.
3. We attach content to this node.
4. Repeat doing this until we reach the end of the string.

### Implementation details

There are many auxiliary functions, but I will focus on the main functions.

**Classes**

1. Node class



1. Tree

The tree data structure (a data organization, management, and storage format that enables efficient access and modification) is a widely used abstract data type that simulates a hierarchical tree structure, with a root value and subtrees of children with a parent node, represented as a set of linked nodes.

**Complexity**

1. parse\_and\_bulid\_xml\_tree method

* Time Complexity **,** because we need to loop over the whole string.
* Space Complexity **,** where M is the number of tags because we store the string in Tree

1. convert\_to\_json method

* Time Complexity **,** where M is the number of tags (Nodes) because we recursively print each node name and value
* Space Complexity **,** because we need to store the string in JSON format.