# Java Coding Assignment

The following is a small coding exercise. You have to provide a working Java program that fulfils a number of requirements. The assessment of your answer will be based on fulfilment of functionality requirements and the quality of the implementation.

## Hand-in

The result of the assignment can be supplied as a zipped project sources.

## Set up

Your solution must be written in Java version at least 1.7. Your application must be able to read from standard input and write onto standard output. Feel free to present your ability to use common tools, libs or frameworks (wiring, testing, logging, and building).

## Functionality

The basic requirement is to provide a program that can convert text into either XML formatted data or comma separated (CSV) data. The text is to be parsed, broken into sentences and words and the words have to be sorted.

Create a domain model class Sentence.java, which will be used as a key in a Map.

Code should be clean and testable. Unit tests must be provided as a formal verification of functional requirements.

Consider huge input when dealing IO operations (entire dataset cannot fit in heap). For testing purposes try running your code against our file large.in with the following JVM argument: -Xmx32m

The program must be able to read input text like:

“Mary had a little lamb. Peter called for the wolf, and Aesop came.

Cinderella likes shoes.”

and parse this text into relevant model classes, and be able to convert the structure to both XML and CSV format.

The parsing must break the text into sentences and words. The parser should allow some whitespace around words and delimiters, e.g. the following is allowed as input and should produce the same result as the first example:

“ Mary had a little lamb .

Peter called for the wolf , and Aesop came .

Cinderella likes shoes..”

In both cases the XML result should be like:

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

<text>

<sentence>

<word>a</word>

<word>had</word>

<word>lamb</word>

<word>little</word>

<word>Mary</word>

</sentence>

<sentence>

<word>Aesop</word>

<word>and</word>

<word>called</word>

<word>came</word>

<word>for</word>

<word>Peter</word>

<word>the</word>

<word>wolf</word>

</sentence>

<sentence>

<word>Cinderella</word>

<word>likes</word>

<word>shoes</word>

</sentence>

</text>

And the CSV result should likewise be:

, Word 1, Word 2, Word 3, Word 4, Word 5, Word 6, Word 7, Word 8

Sentence 1, a, had, lamb, little, Mary

Sentence 2, Aesop, and, called, came, for, Peter, the, wolf

Sentence 3, Cinderella, likes, shoes

Make sure to check sample files:

|  |  |  |
| --- | --- | --- |
| Sample input file | Sample output file | Comment |
| large.in |  | Lengthy input |
| small.in | small.xml  small.csv | Functional cases |