**hadoop jar WordCountPerBook.jar org.apache.hadoop.examples.WordCountPerBook /user/itv002099/mr\_files/input /user/itv002099/test1/mrout**

**toString**

public [String](https://docs.oracle.com/javase/6/docs/api/java/lang/String.html) **toString**()

Returns a string representation of the object. In general, the toString method returns a string that "textually represents" this object. The result should be a concise but informative representation that is easy for a person to read. It is recommended that all subclasses override this method.

The toString method for class Object returns a string consisting of the name of the class of which the object is an instance, the at-sign character `@', and the unsigned hexadecimal representation of the hash code of the object. In other words, this method returns a string equal to the value of:

getClass().getName() + '@' + Integer.toHexString(hashCode())

**Returns:**

a string representation of the object.

**getInputSplit**

public [InputSplit](https://hadoop.apache.org/docs/r1.0.4/api/org/apache/hadoop/mapreduce/InputSplit.html) **getInputSplit**()

Get the input split for this map.

**Protected** Access Modifier - **Protected**. Variables, **methods**, and constructors, which are declared **protected** in a superclass can be accessed only by the subclasses in other package or any class within the package of the **protected**members' class. The **protected** access modifier cannot be applied to class and interfaces.

First you need to get the input split, using the newer mapreduce API it would be done as follows:

context.getInputSplit();

But in order to get the file path and the file name you will need to first typecast the result into FileSplit.

So, in order to get the input file path you may do the following:

Path filePath = ((FileSplit) context.getInputSplit()).getPath();

String filePathString = ((FileSplit) context.getInputSplit()).getPath().toString();

Similarly, to get the file name, you may just call upon getName(), like this:

String fileName = ((FileSplit) context.getInputSplit()).getPath().getName();

[**getPath**](https://hadoop.apache.org/docs/r2.4.1/api/org/apache/hadoop/mapreduce/lib/input/FileSplit.html#getPath())()   
          The file containing this split's data

## Class StringTokenizer

The string tokenizer class allows an application to break a string into tokens. The tokenization method is much simpler than the one used by the StreamTokenizer class. The StringTokenizer methods do not distinguish among identifiers, numbers, and quoted strings, nor do they recognize and skip comments.

The set of delimiters (the characters that separate tokens) may be specified either at creation time or on a per-token basis.

The following is one example of the use of the tokenizer. The code:

StringTokenizer st = new StringTokenizer("this is a test");

while (st.hasMoreTokens()) {

System.out.println(st.nextToken());

}

prints the following output:

this

is

a

test

Methods:

[**hasMoreTokens**](https://docs.oracle.com/javase/7/docs/api/java/util/StringTokenizer.html#hasMoreTokens())()

Tests if there are more tokens available from this tokenizer's string

[**nextToken**](https://docs.oracle.com/javase/7/docs/api/java/util/StringTokenizer.html#nextToken())()

Returns the next token from this string tokenizers

## Class IntWritable

[**set**](https://hadoop.apache.org/docs/r2.7.4/api/org/apache/hadoop/io/IntWritable.html#set-int-)(int value)

Set the value of this IntWritable.

[**get**](https://hadoop.apache.org/docs/r2.7.4/api/org/apache/hadoop/io/IntWritable.html#get--)()

Return the value of this IntWritable.

[**GenericOptionsParser**](https://hadoop.apache.org/docs/r1.2.1/api/org/apache/hadoop/util/GenericOptionsParser.html#GenericOptionsParser(java.lang.String[]))([String](http://java.sun.com/javase/6/docs/api/java/lang/String.html?is-external=true)[] args)   
          Create an options parser to parse the args.

method: [**getRemainingArgs**](https://hadoop.apache.org/docs/r1.2.1/api/org/apache/hadoop/util/GenericOptionsParser.html#getRemainingArgs())()   
          Returns an array of Strings containing only application-specific arguments

GenericOptionsParser is a utility to parse command line arguments generic to the Hadoop framework. GenericOptionsParser recognizes several standarad command line arguments, enabling applications to easily specify a namenode, a jobtracker, additional configuration resources etc.