

Excel based data-source for JMETER



1286463 oNTARIO INC.

2016

The objectives 2

Motives 2

Implementation 2

AbstractExtention class 2

AbstractExcelDataProvider 3

ExcelDataReader 3

ExcelUtils 3

How does it work. 5

Conclusion. 8

## The objectives

The objective of this project is to make possible to use data stored as an MS Excel document from the JMeter run-time environment.

## Motives

Currently, the only available data source for JMeter scripts is a comma separated values (CSV) files. For simple data sets where only alpha numeric characters are used it is good enough: the JMeter reads a csv file, splits values using configurable delimiter and assigns those values to variables in a context of JMeter.

However, for data sets with more than 10 columns and values with commas, quotes (double or single) using CSV files becomes problematic.

A MS Excel document provides ability to store and easy navigate complex data sets. It makes Excel an excellent data storage for testing purposes.

Using the Apache POI libraries (https://poi.apache.org/) and JMeter extensibility it makes sense to design and develop an utility that makes an MS Excel document to be a data source for a JMeter script.

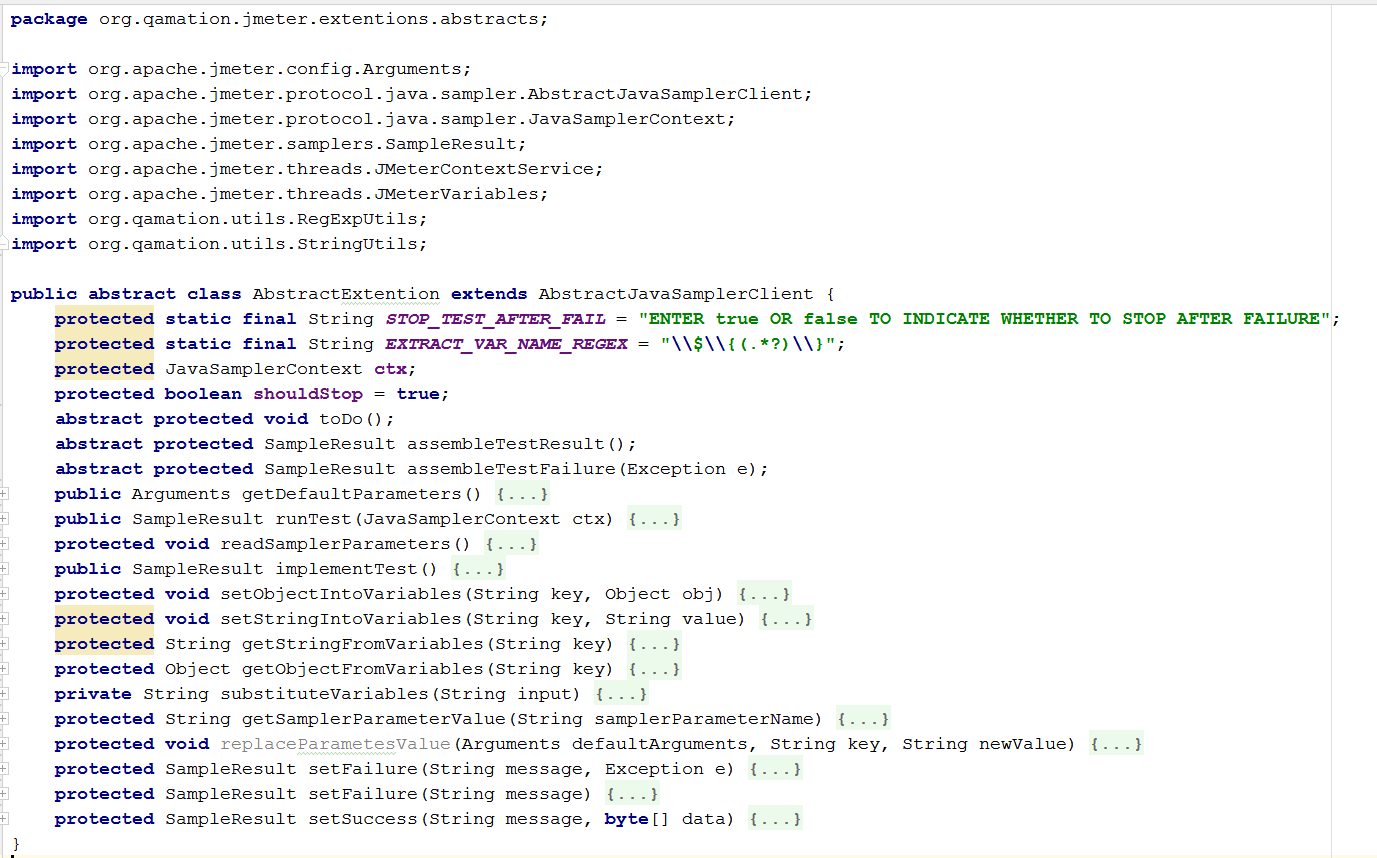
## Implementation

The goal of the implementation is to hide from a user all complexities when dealing with MS Excel documents using POI library and the same time to release a simple interface to exchange data between an excel document and JMeter.

The JMeter is able to instantiate an external class if one is implementing the JavaSamplerClient interface or extends AbstractJavaSamplerClient class. That gives any external class access to the JMeter run-time environment (context) including system and user variables. Also, it gives control over a test execution flow.

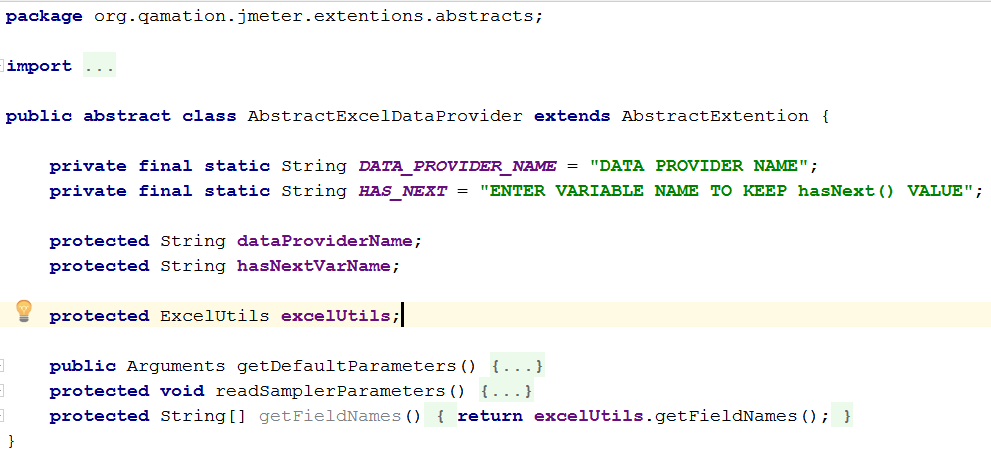
### AbstractExtention class

The main purpose of the class is to isolate communication with JMeter run-time environment from all other functionality. AbstractExtention class extends “*AbstractJavaSamplerClient”* that implements *JavaSamplerClient* interface from JMeter API.



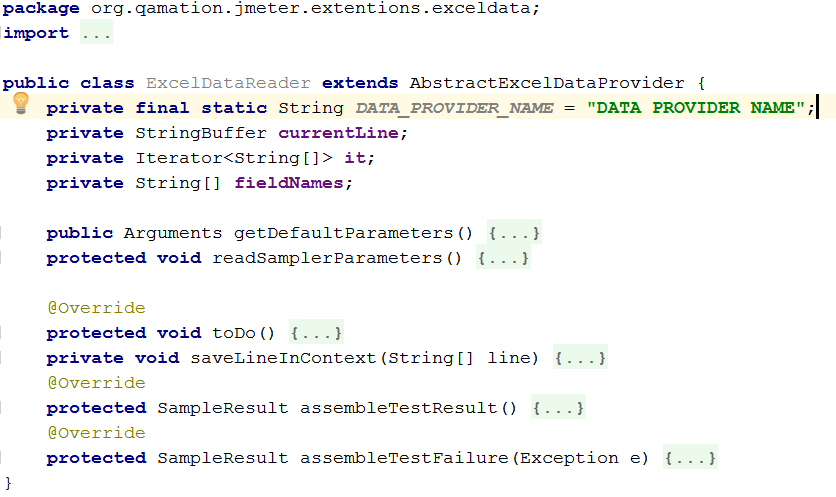
### AbstractExcelDataProvider

*AbstractExcelDataProvider* class is the second layer of abstraction. It’s main responsibility to collect basic information about a file with Excel data that will be read or created.



### ExcelDataReader

*ExcelDataProvider* class works as a bridge between JMeter and POI library. Information received from an Excel document goes into JMeter context through this class.



### ExcelUtils

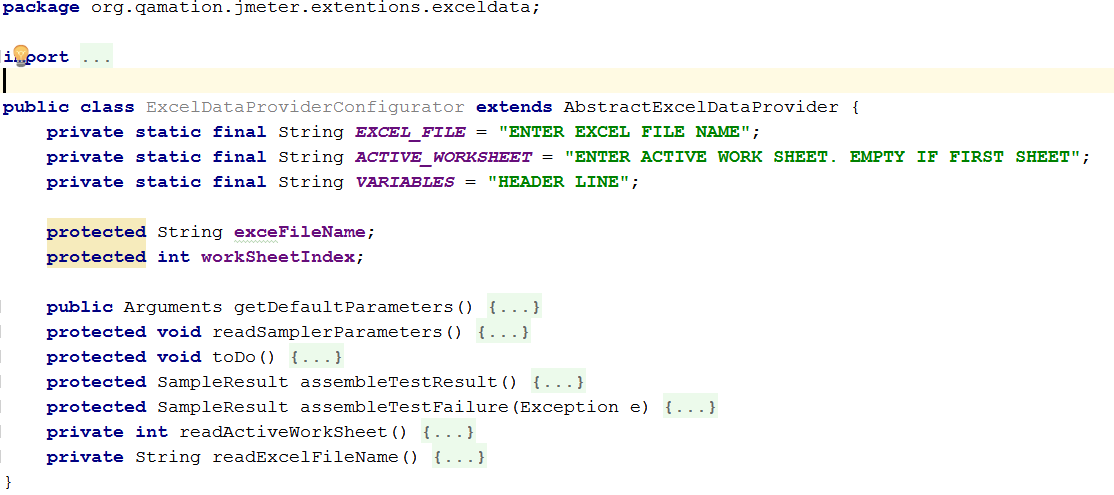
*ExcelUtils* class is the place where an excel document is opened and read. The class supports xls and xlsx file formats. As parameters, *ExcelUtils* class takes a file name of an excel document and an index number of the active work sheet. The first row of the excel sheet should be a header.This class lets users to read excel data by rows where each row is represented as a array of strings with length equal to length of the header.

ExcelDataProviderConfigurator

The last, but not least class that should be included in a JMeter script to read data from an Excel document is *ExcelDataProviderConfigurator*. This class is the first step in a process of reading data from an Excel document. This class expects a user to provide an URL to an Excel document and work sheet number to read data from. *ExcelDataProviderConfigurator* also informs its user that the first line in the work sheet to be read from should be a header.

ExcelDataProviderConfigurator opens the Excel document using provided URL, creates an Excel document iterator object - an object that can do only two operations: check if the iterator has more elements to iterate through via hasNext() method and extract array of strings representing “next” line from the excel document using method next().

The iterator then is stored in JMeter run-time context to be used by ExceDataReader object.



### How does it work.

In order to use data from an Excel document at least two java samplers need to be included into a JMeter script. First, the Java Sampler with *ExcelDataProviderConfigurator* class. Second, at any place where the next line from Excel is required insert the Java Sampler with *ExcelDataProvider* class.

During the first step the desired Excel document will be opened. An iterator through data in the active worksheet will be created and saved in the JMeter context.

A call to the Java Sampler with *ExcelDataProvider* will read the next line from the Excel file and save each field value in the JMeter context using variable name defined in corresponding header line field.

For example, if the first line in an Excel document has six fields with values Field\_1, Field\_2, …, Field\_6 than each call to the *ExcelDataProvider* class from JMeter will product six variables with names Field\_1, Field\_2, …, Field\_6 for each line read.

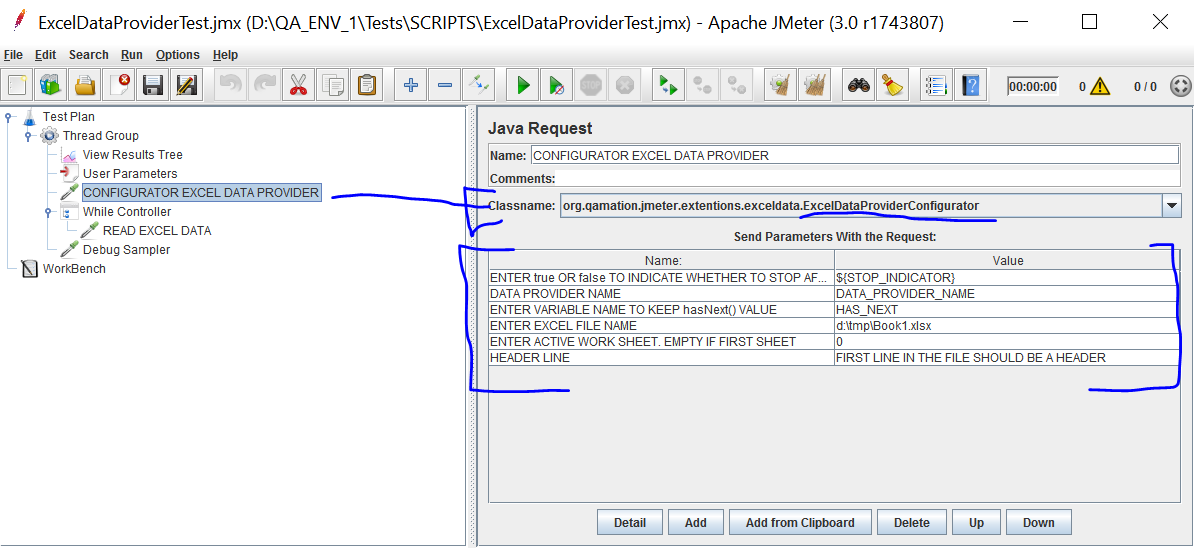
A call to *ExcelDataProvider* will fail if it asked to read beyond the available data.

The best way to use the *ExceDataProvider* is using it within JMeter While Loop configurator with a reference to ${HAS\_NEXT} variable in the loop.

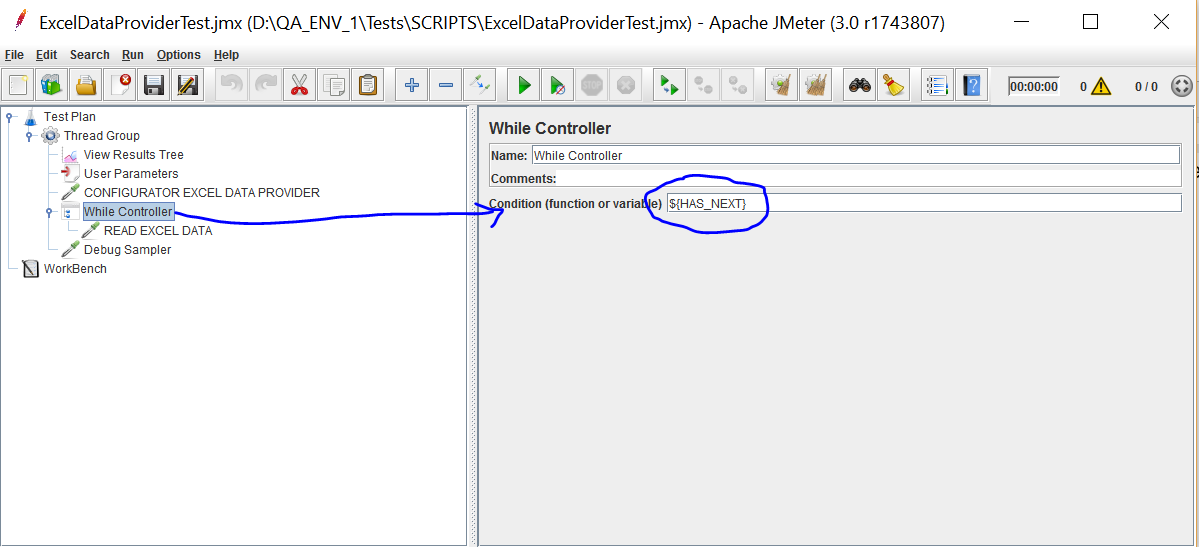
Below are several illustrations how *ExcelDataProviderConfigurator* and *ExcelDataProvider* are used in JMeter.

1. Excel Data Provider Configuration.

It creates iterator through the excel document based on information provided by user.



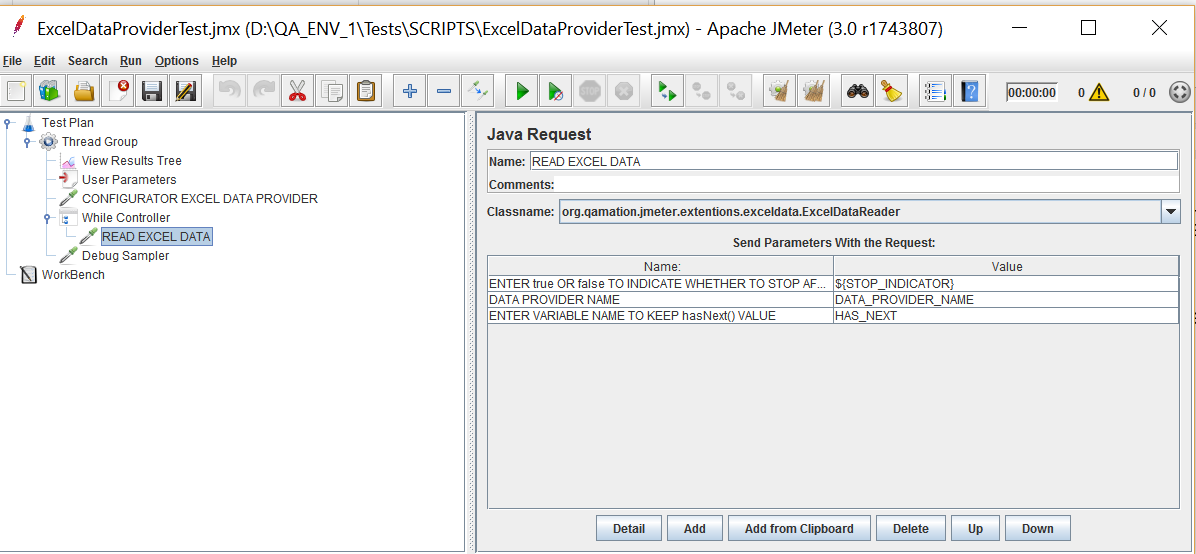
1. While loop will repeat as long as variable ${HAS\_NEXT} returns “true”. In the example below the loop has only one step – a call to *ExcelDataReader*.

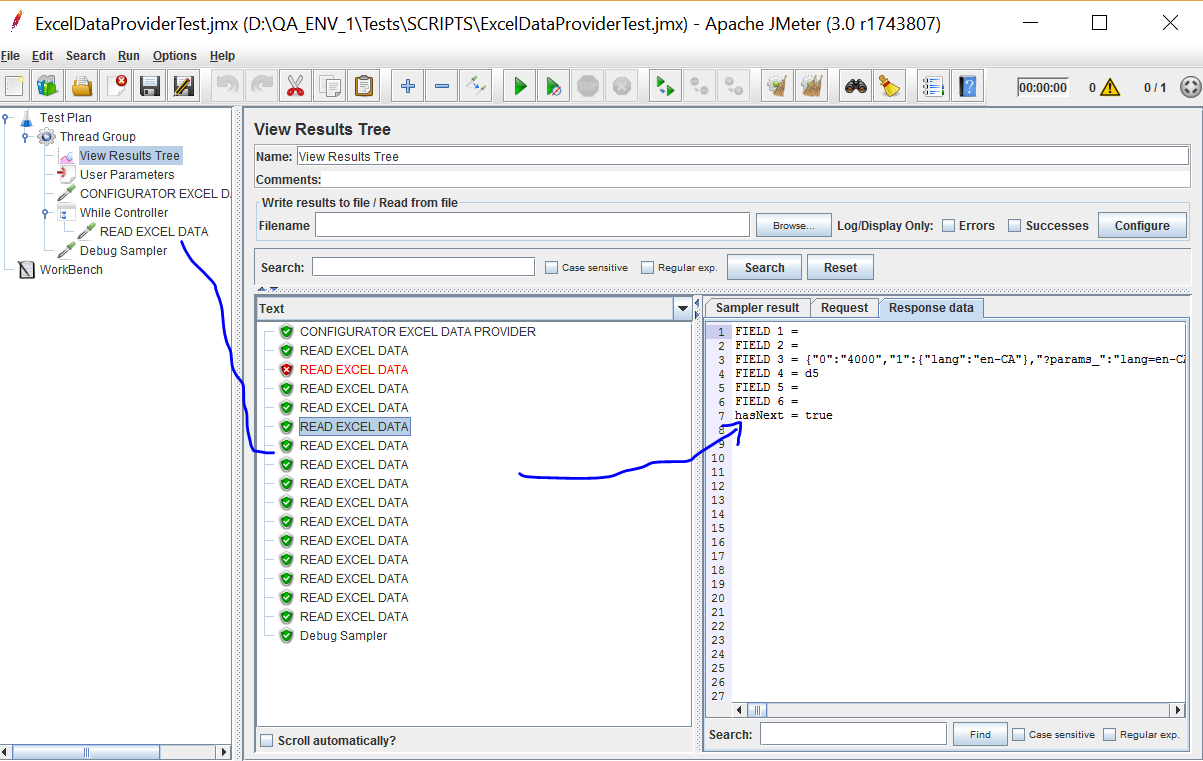


1. ExcelDataReader

A call to this class reads next available data row from excel, sets values from each field from the read row into a variable with name specified in corresponding column of the first (header) row. Finally, it updates value of the “HAS\_NEXT” variable.

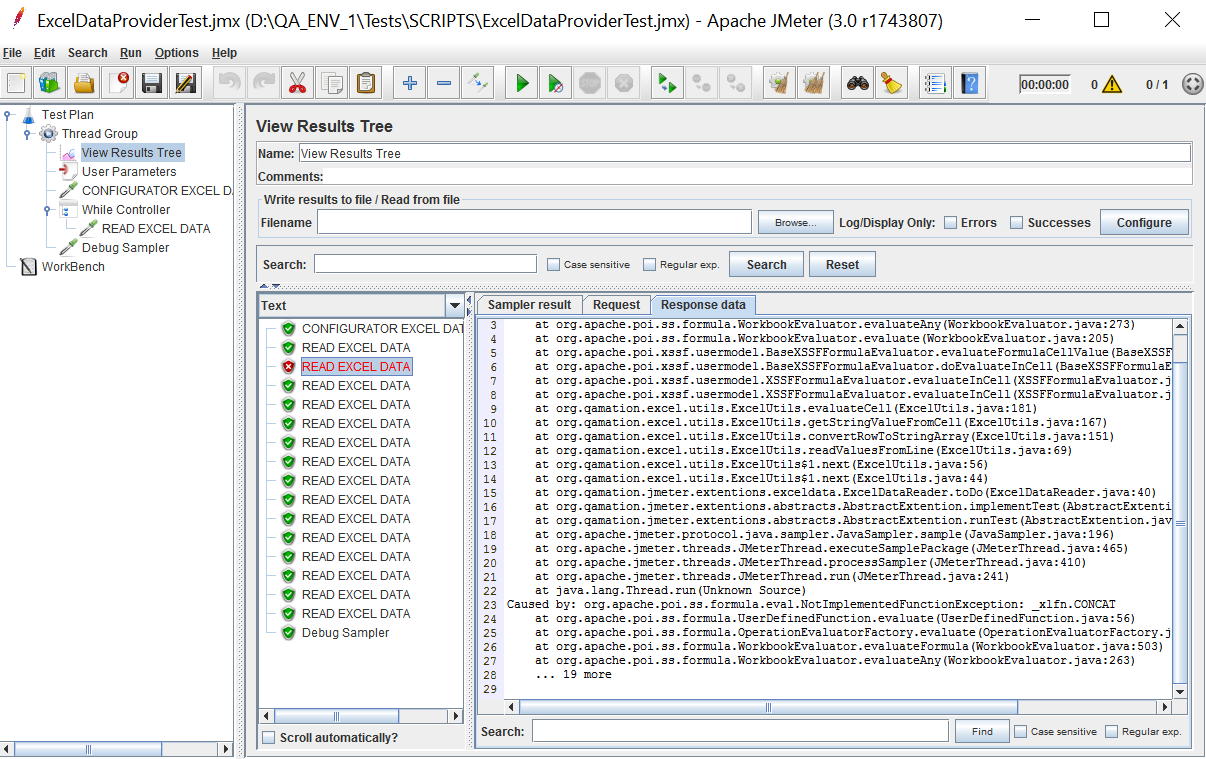
While loop stops when “HAS\_NEXT” variable becomes “false”.





1. Evaluation excel formulas.

ExcelUtils asks POI library to evaluate cell’s value if required. Unfortunately, not all functions can be evaluated. Here is the example of failure when POI library is unable to execute Excel CONCAT() function.



## Conclusion.

Excel Data Provider comes up with several benefits:

* It is more convenient to work with an excel document than with CSV file.
* Excel file support multi-line data in a cell. It especially important for testing when it is necessary to use JSON or SQL.
* This data provider does not depend on a field delimiter, like in a CSV file. It leads to the ability to store data that may have coma, quotes or other special symbols.

Having Excel Data Provider for JMeter decouples test data from a test and makes quality assurance development process easier and more efficient.