



Performance Testing Course : JMeter - Assertion

Author: Hien HOANG



Agenda

- Introduction about several popular assertion.
- Showcases.
- Exercises.

Definition

An assertion element is used as a validation component in JMeter. It validates the response on the basis of pre-defined condition and based on that make a decision to pass or fail the sampler.

If you need to apply assertion on a particular sampler, then add it as a child of that sampler.

Response Assertion

Attribute:

- Name: Name of assertion.
- Comments: Any description.
- Apply to: What do you want to assert:
 - Main samples and sub-samples:

(Search given string pattern in main request and re-directed request)

- Main samples(Only on main request)
- Sub-samples only(Only in re-directed ones)
- JMeter variable(Assert against a variable)

The screenshot shows the 'Response Assertion' configuration window in JMeter. The 'Name' field is set to 'Response Assertion'. The 'Comments' field is empty. In the 'Apply to' section, the 'Main sample and sub-samples' radio button is selected. The 'Field to Test' section has 'Text Response' selected. The 'Pattern Matching Rules' section has 'Substring' selected. The 'Patterns to Test' list is empty. At the bottom, there are buttons for 'Add', 'Add from Clipboard', and 'Delete'. A 'Custom failure message' field is also present at the very bottom.

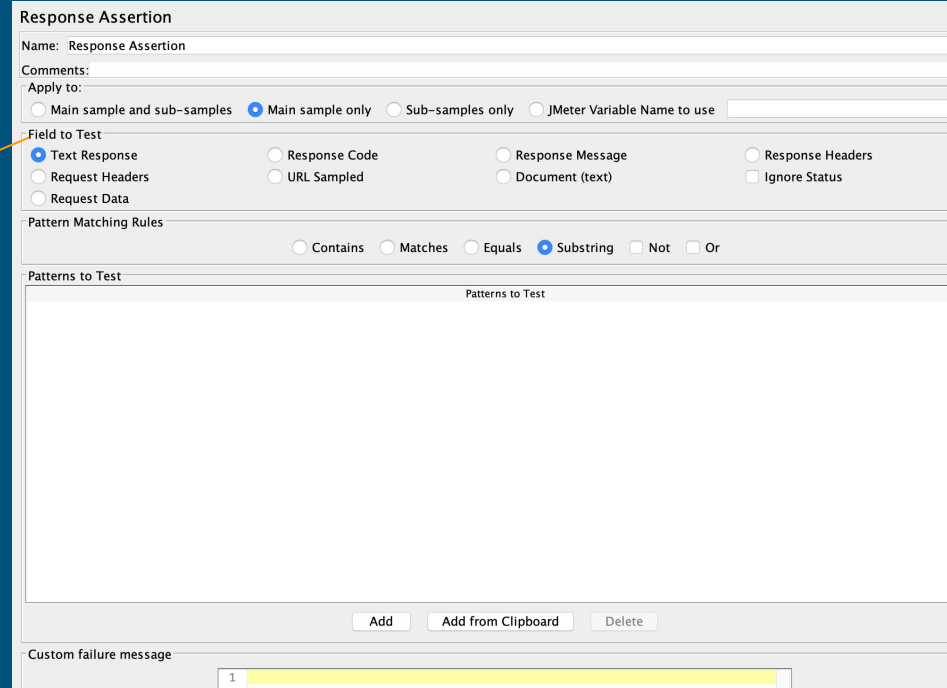
Response Assertion

Attribute:

- Field to Test: Name of assertion.
 - Text Response: Search the given pattern

string in body of response.

- Response Code : Validate response code.
- Response Message: Validate response message.
- Response Headers: Search given pattern string in response headers.
- Request Headers: Search in the header part of the request.
- URL Sampled: Search only in URL.
- Document(text): Search in document returned by server.
- Request Data: Validate string in request body sent to server. It doesn't include request header.
- Ignore Status: Force response status to successful.



Response Assertion

Name: Response Assertion

Comments:

Apply to:

☐ Main sample and sub-samples ☒ Main sample only ☐ Sub-samples only ☐ JMeter Variable Name to use

Field to Test

☒ Text Response ☐ Response Code ☐ Response Message ☐ Response Headers

☐ Request Headers ☐ URL Sampled ☐ Document (text) ☐ Ignore Status

☐ Request Data

Pattern Matching Rules

☐ Contains ☐ Matches ☐ Equals ☒ Substring ☐ Not ☐ Or

Patterns to Test

Patterns to Test

Add Add from Clipboard Delete

Custom failure message

1

Response Assertion

Attribute:

- Pattern Matching Rules: Rule to match.
 - Contains: “Field to Test” should contains

“Patterns to Test” string.(Regex OK)

- Matches: “Field to Test” should have string

matched fully with “Patterns to Test” string.(Regex NOT OK)

- Equals: Match exact text with case-sensitive feature.(Regex NOT OK)
- Substring: Jmeter checks and validate the substring from captured string.(Regex NOT OK)
- Not: String pattern should not be in the response.
- Or: In case of multiple pattern, pass the sampler if any one of the pattern matched.

The screenshot shows the 'Response Assertion' configuration window in JMeter. The 'Name' field is set to 'Response Assertion'. The 'Comments' field is empty. The 'Apply to' section has four radio buttons: 'Main sample and sub-samples', 'Main sample only' (selected), 'Sub-samples only', and 'JMeter Variable Name to use'. The 'Field to Test' section has eight radio buttons: 'Text Response' (selected), 'Response Code', 'Response Message', 'Response Headers', 'Request Headers', 'URL Sampled', 'Document (text)', and 'Ignore Status'. The 'Pattern Matching Rules' section has five radio buttons: 'Contains', 'Matches', 'Equals', 'Substring' (selected), 'Not', and 'Or'. The 'Patterns to Test' section is empty. At the bottom, there are three buttons: 'Add', 'Add from Clipboard', and 'Delete'. The 'Custom failure message' section is also empty.

Response Assertion

Attribute:

- **Patterns to Test:**
 - Write a list of patterns to be tested.
 - Each pattern is tested separately.
 - If a pattern fails, then further patterns are not

checked (if "Or" is not marked). There is no

difference between setting up one Assertion with multiple patterns and setting up multiple Assertions with one pattern each (assuming the other options are the same).

- **Custom failure message:** A custom message can be written in this field which is displayed in case of the assertion failure.

The screenshot shows the 'Response Assertion' configuration window in JMeter. The 'Name' field is set to 'Response Assertion'. The 'Comments' field is empty. The 'Apply to' section has four radio buttons: 'Main sample and sub-samples', 'Main sample only' (selected), 'Sub-samples only', and 'JMeter Variable Name to use'. The 'Field to Test' section has eight radio buttons: 'Text Response' (selected), 'Response Code', 'Response Message', 'Response Headers', 'Request Headers', 'URL Sampled', 'Document (text)', and 'Ignore Status'. The 'Pattern Matching Rules' section has five radio buttons: 'Contains', 'Matches', 'Equals', 'Substring' (selected), 'Not', and 'Or'. The 'Patterns to Test' section is a large text area with a title bar 'Patterns to Test'. Below it are three buttons: 'Add', 'Add from Clipboard', and 'Delete'. The 'Custom failure message' section has a text area with a tab labeled '1'.

Response Assertion

Showcase:

- Use Response Assertion to assert if we search weather for city London

Exercise:

- Use Response Assertion to assert if we found Amsterdam as Netherlands capital city.

JSON Assertion

Definition:

- JSON Assertion is used to assert any particular string against JSON response data.

JSON Assertion

Attribute:

- Name: Name of the assertion
- Comment: Description of the assertion.
- Assert JSON Path exists: Input JSON Path here.
- Additional Assert Value: if checked,

also assert the value is equal to the **Expected Value**.

- Match as regular expression: Verify if response

value is matched against regular expression.

- Expected Value: Declare the expected value here.

JSON Assertion

Name: JSON Assertion

Comments:

Assert JSON Path exists: `$.*.subregion`

☒ Additionally assert value

☒ Match as regular expression

Expected Value:

Western Asia

☐ Expect null

☐ Invert assertion (will fail if above conditions met)

JSON Assertion

Attribute:

- Expect null: Verify if response data is null.
- Invert assertion: Revert result of assertion.

(if passed -> make it to failed and vice versa).

JSON Assertion

Name: JSON Assertion

Comments:

Assert JSON Path exists: `$.*.subregion`

- ☒ Additionally assert value
- ☒ Match as regular expression

Expected Value:

Western Asia

- ☐ Expect null
- ☐ Invert assertion (will fail if above conditions met)

JSON Assertion

Showcase:

- User JSON Assertion to verify if all countries belong to subregion.

Exercise:

- Do your own exercise.

Duration Assertion

Description:

The Duration Assertion is very simple. Used alongside the Response Assertion, it covers 90 percent of use cases where assertions are required. The usage is very straightforward: It provides the maximum duration in milliseconds, and, if any request lasts longer than the value specified, the sample is marked as failed.

Duration Assertion

Description:

- Name: Name of assertion
- Comments: Any description.
- Apply to:
- Main samples and

sub-samples:(Assert the load time
against main and re-directed requests)

- Main samples(Only on main request)
- Sub-samples only(Only in re-directed ones)

Duration Assertion

Name: Duration Assertion

Comments:

Apply to:

☐ Main sample and sub-samples ☒ Main sample only ☐ Sub-samples only

Duration to Assert

Duration in milliseconds: 1000

Duration Assertion

Showcase:

- Use Duration Assertion to assert load time of 4 APIs: Get list of languages, get capital city of country knowing country ISO code, get list of countries using currencies and get country info by ISO code.

Exercise:

- Using this WSDL : <http://webservices.daehosting.com/services/isbnservice.wso?WSDL> , verify duration of each API IsValidISBN10 and IsValidISBN13 is less than 600 ms.

JSR223 Assertion

Description:

Same as JSR223 Pre-processor, JSR223 Post-processor and JSR223 Sampler, it helps to implements your custom assertion.

JSR223 Assertion

Attribute:

- Language : Choose your scripting language.
- Script file: Name of a file to be used as a JSR223 script.

The screenshot shows the 'JSR223 Assertion' configuration window. It includes fields for 'Name' (set to 'JSR223 Assertion'), 'Comments', 'Script language' (set to 'groovy'), and 'Parameters to be passed to script'. There is a 'Script file (overrides script)' section with a 'File Name' field and a 'Browse...' button. A 'Script compilation caching' section has a checked 'Cache compiled script if available' checkbox. At the bottom, a 'Script' text area is visible with a line number '1' on the left.

JSR223 Assertion

Name: JSR223 Assertion

Comments:

Script language (e.g. beanshell, javascript, jexl):

Language: groovy (Groovy 2.4.16 / Groovy Scripting Engine 2.0)

Parameters to be passed to script (=> String Parameters and String []args)

Parameters:

Script file (overrides script)

File Name: Browse...

Script compilation caching

Cache compiled script if available: ☒

Script (variables: ctx vars props SampleResult (aka prev) AssertionResult sampler log Label Filename Parameters args[] OUT)

Script:

1

- Parameters: List of parameters to be passed to the script file or the script.
- Cache compiled script if available: If checked and language supports Compilable interface(for now we only have Groovy in the list),JMeter will compile the script and cache it.
- Scripts(variables: xxx) : List supported variables by JMeter.

JSR223 Assertion

Showcase:

- Script file: Use JSR223 Assertion to assert all countries returned by API belongs to our expected sub-region.

Exercise:

- Use JSR223 Assertion to assert that API get country info by ISO code returns Amsterdam as capital city and euro as currency.

MD5Hex Assertion

Description:

The MD5Hex assertion checks the MD5 checksum of the actual response against the expected MD5 hash. Content of any length, whether it's one character or a full HD video file, will be represented as a 32-digit hexadecimal number. It is particularly useful for large data-integrity checks(especially when you need to load test file downloading feature).

MD5Hex Assertion

Attribute:

- Name: Name of assertion
- Comments: Any description.
- MD5Hex: Expected MD5Hex value

MD5Hex Assertion	
Name:	MD5Hex Assertion
Comments:	
MD5Hex to Assert	
MD5Hex	\${expected_hash}

MD5Hex Assertion

Showcase:

- Use MD5Hex Assertion to verify against this API:
https://file-examples.com/wp-content/uploads/2017/10/file-example_PDF_1MB.pdf

Exercise:

- Do it by your own.

Size Assertion

Description:

Size Assertion checks the response length to see if it's equal/not equal/greater/less than the expected size in bytes. It can be applied to:

1. Full response (body and headers)
2. Response headers
3. Response body
4. Response code
5. Response message

Size Assertion

Attribute:

- Name: Name of assertion
- Comments: Any description.
- Apply to: What do you want to assert:
 - Main samples and sub-samples:

(Assert response size of both main and re-directed samples)

- Main samples(Only on main request)
- Sub-samples only(Only in re-directed ones)
- JMeter variable(Assert against a variable)

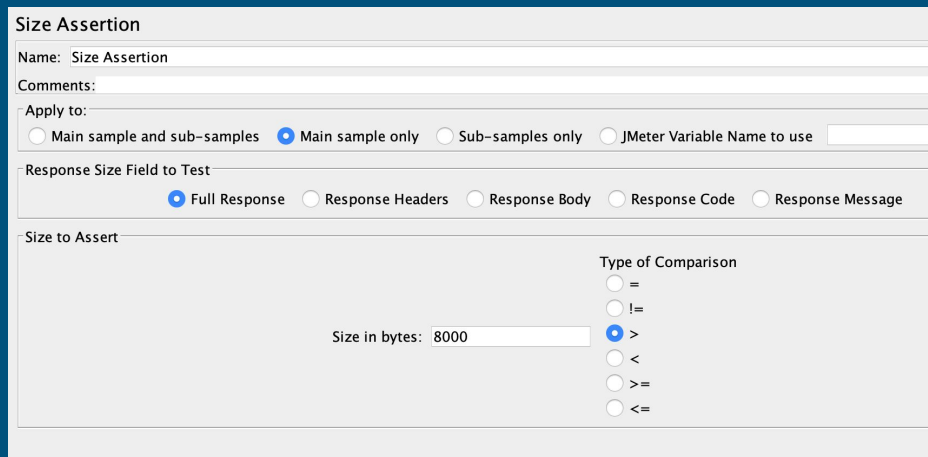
The screenshot shows the 'Size Assertion' configuration window in JMeter. It includes fields for 'Name' (set to 'Size Assertion') and 'Comments'. The 'Apply to:' section has four radio buttons: 'Main sample and sub-samples', 'Main sample only' (selected), 'Sub-samples only', and 'JMeter Variable Name to use'. The 'Response Size Field to Test' section has five radio buttons: 'Full Response' (selected), 'Response Headers', 'Response Body', 'Response Code', and 'Response Message'. The 'Size to Assert' section contains a text field 'Size in bytes' with the value '8000'. To the right, the 'Type of Comparison' section has six radio buttons: '=', '!=', '>' (selected), '<', '>=', and '<='.

Size Assertion

Attribute:

- Size to Assertion:

Input expected value in bytes and select appropriate comparison operation.



The screenshot shows the 'Size Assertion' configuration window in JMeter. It includes fields for 'Name' (set to 'Size Assertion') and 'Comments'. The 'Apply to' section has four radio buttons: 'Main sample and sub-samples', 'Main sample only' (selected), 'Sub-samples only', and 'JMeter Variable Name to use'. The 'Response Size Field to Test' section has five radio buttons: 'Full Response' (selected), 'Response Headers', 'Response Body', 'Response Code', and 'Response Message'. The 'Size to Assert' section contains a text field 'Size in bytes' with the value '8000'. To the right, the 'Type of Comparison' section has six radio buttons: '=', '!=', '>' (selected), '<', '>=', and '<='.

Size Assertion	
Name: Size Assertion	
Comments:	
Apply to:	
<input type="radio"/> Main sample and sub-samples	<input checked="" type="radio"/> Main sample only
<input type="radio"/> Sub-samples only	<input type="radio"/> JMeter Variable Name to use
Response Size Field to Test	
<input checked="" type="radio"/> Full Response	<input type="radio"/> Response Headers
<input type="radio"/> Response Body	<input type="radio"/> Response Code
<input type="radio"/> Response Message	
Size to Assert	
Size in bytes: 8000	Type of Comparison
	<input type="radio"/> =
	<input type="radio"/> !=
	<input checked="" type="radio"/> >
	<input type="radio"/> <
	<input type="radio"/> >=
	<input type="radio"/> <=

Size Assertion

Showcase:

- Assert size of full response of API get all languages.

Exercise:

- Apply it to any API.

Some tips

- **The Cost of JMeter Assertions**

- All assertions come with a cost, in terms of CPU or memory consumption. However, some assertions carry a greater cost than others. According to the JMeter Performance and Tuning Tips guide, the Response Assertion and the Duration Assertion are typically lower-impact choices, whereas Compare Assertion and other XML-based ones like the XPath Assertion consume more CPU and memory.

- **The Scope of JMeter Assertions**

- You must also consider the scope when setting assertions. Assertions can be applied to a main sample and its subsamples, or only to subsamples. Some assertions, like the Response Assertion or the Size Assertion, can also be used against a JMeter Variable. Code-based assertions (such as Beanshell, BSF and JSR223) don't have the GUI element that identifies scope. This means you must manually implement all assertion logic – including scope.

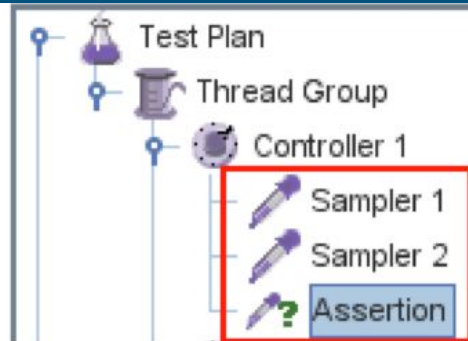
Some tips

- Performance of each assertion:

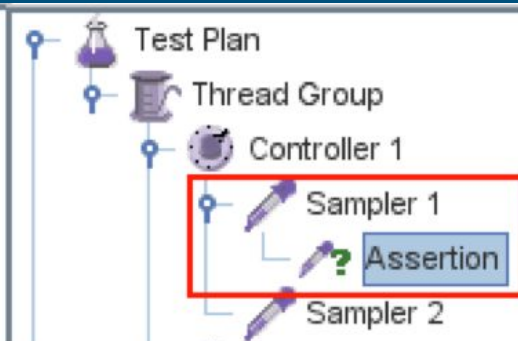
Assertion	CPU/Memory Usage	Notes
Response Assertion	Moderate	Regular Expressions
Duration Assertion	Low	
Size Assertion	Low	
XML Assertion	High	Builds XML DOM Documents
Beanshell Assertion	Variable	Depends on the script logic
MD5Hex Assertion	Low	
HTML Assertion	High	Parses the HTML Response
XPath Assertion	High	Builds XML DOM Documents
XML Schema Assertion	High	Builds XML DOM Documents
JSR223 Assertion	Variable	Depends on the script logic
Compare Assertion	High	Parses responses and compares them
SMIME Assertion	Moderate	
Json Assertion	High	Parses the Json document

Some tips

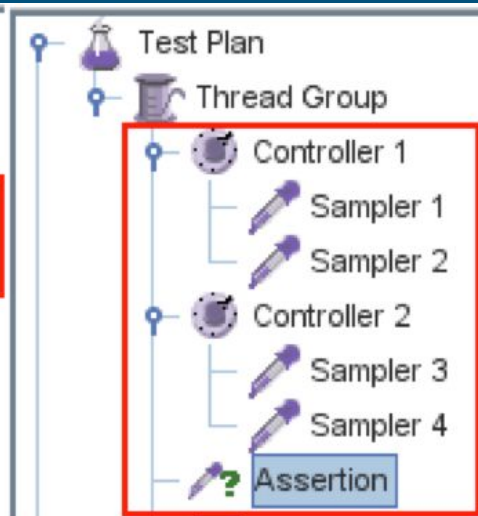
Example of scope:



Assertion applies to
Sampler 1 and Sampler 2



Assertion applies to
Sampler 1 only



Assertion applies to
ALL Samplers

Some tips

- **Combining Assertions**
 - You can add more than one assertion to the sampler, controller, thread group, or test plan. Failed assertions will cause all affected samples to fail, so caution is essential.