JMETER - FUNCTIONS

JMeter Functions and User Variables

JMeter functions are special values that can populate fields of any Sampler or other element in a test tree.

- A function call looks like this −

```
${__functionName(var1,var2,var3)}
```

- *_functionName* matches the name of a function. For example **${__threadNum}**.
- If a function parameter contains a comma, then make sure you escape this with "\" as shown below −

```
${__time(EEE\, d MMM yyyy)}
```

- Variables are referenced as −

```
${VARIABLE}
```

List of Functions

Following table lists a group of functions loosely grouped into types −

Function Type	Name	Comment
Information	threadNum	Get thread number.
Information	samplerName	Get the sampler name *label*.
Information	machineIP	Get the local machine IP address.
Information	machineName	Get the local machine name.
Information	time	Return current time in various formats.
Information	log	Log *ordisplay* a message *andreturnthevalue*.
Information	logn	Log *ordisplay* a message *emptyreturnvalue*.
Input	StringFromFile	Read a line from a file.
Input	FileToString	Read an entire file.
Input	CSVRead	Read from CSV delimited file.
Input	XPath	Use an XPath expression to read from a file.
Calculation	counter	Generate an incrementing number.
Calculation	intSum	Add int numbers.
Calculation	longSum	Add long numbers.
Calculation	Random	Generate a random number.
Calculation	RandomString	Generate a random string.
Calculation	UUID	Generate a random type 4 UUID.

Scripting	BeanShell	Run a BeanShell script.
Scripting	javaScript	Process JavaScript *MozillaRhino*.
Scripting	jexl, jexl2	Evaluate a Commons Jexl expression.
Properties	property	Read a property.
Properties	P	Read a property *shorthandmethod*.
Properties	setProperty	Set a JMeter property.
Variables	split	Split a string into variables.
Variables	V	Evaluate a variable name.
Variables	eval	Evaluate a variable expression.
Variables	evalVar	Evaluate an expression stored in a variable.
String	regexFunction	Parse previous response using a regular expression.
String	escapeOroRegexpChars	Quote meta chars used by ORO regular expression.
String	char	Generate Unicode char values from a list of numbers.
String	unescape	Process strings containing Java escapes e.g. \n & \t.
String	unescapeHtml	Decode HTML-encoded strings.
String	escapeHtml	Encode strings using HTML encoding.
String	TestPlanName	Return name of current test plan.

- There are two kinds of functions −
  - User-defined static values *orvariables*
  - Built-in functions
- User-defined static values allow the user to define variables to be replaced with their static value when a test tree is compiled and submitted to be run.
- The variables cannot be nested; i.e **{Var** **{N}}** does not work.
- The __V *variable* function *versionsafter*2.2 can be used to do this − {__V(Var{N})}.
- This type of replacement is possible without functions, but it is less convenient and less intuitive.

Where to Use Functions And Variables

Functions and variables can be written into any field of any test component.

The following functions should work well in a test plan −

- intSum
- longSum
- machineName
- BeanShell

- javaScript
- jexl
- random
- time
- property functions
- log functions

Functions which are used on the Test Plan have some restrictions. JMeter thread variables will have not been fully set up when the functions are processed, so variable names passed as parameters will not be set up and variable references will not work. Hence, *split* and *regex* and the variable evaluation functions will not work. The *threadNum* function will not work and it does not make sense at test plan level.

Referencing Variables and Functions

- Referencing a variable in a test element is done by bracketing the variable name with '${' and '}'.
- Functions are referenced in the same manner, but by convention, the names of functions begin with "__" to avoid conflict with user value names.
- Some functions take arguments to configure them, and these go in parentheses, comma-delimited. If the function takes no arguments, the parentheses can be omitted. For example −

```
${__BeanShell(vars.put("name"\,"value"))}
```

- Alternatively, you can define your script as a variable, e.g. on the Test Plan −

```
SCRIPT     vars.put("name","value")
```

- The script can then be referenced as follows −

```
${__BeanShell(${SCRIPT})}
```

The Function Helper Dialog

The Function Helper Dialog is available from JMeter's **Options** tab.

- Using the Function Helper, you can select a function from the pull down, and assign values for its arguments. The left column in the table provides a brief description of the argument, and the right column is where you write the value for that argument. Different functions take different arguments.
- Once you have done this, click the “Generate" button, and the appropriate string is generated, which you can copy-paste into the test plan wherever you need to.

Pre-defined Variables

Some variables are defined internally by JMeter. They are −

- COOKIE_cookiename − contains the cookie value.
- JMeterThread.last_sample_ok − whether or not the last sample was OK − true/false. Note − this is updated after PostProcessors and Assertions have been run.
- START variables.

Pre-defined Properties

Some built-in properties are defined by JMeter. These are listed below. For convenience, the

START properties are also copied to variables with the same names.

- START.MS − JMeter start time in milliseconds.
- START.YMD − JMeter start time as yyyyMMdd.
- START.HMS − JMeter start time as HHmmss.
- TESTSTART.MS − test start time in milliseconds.

Note that the START variables / properties represent JMeter startup time, not the test start time. They are mainly intended for use in file names etc.