UserManual GettingStarted

h1. Getting Started The easiest way to begin using JMeter is to first [download the latest production release|http://jakarta.apache.org/site/downloads/downloads_jmeter.cgi] and install it. The release contains all of the files you need to build and run Web, FTP, JDBC, and JNDI tests. If you want to perform JDBC testing, then you will, of course, need the appropriate JDBC driver from your vendor. JMeter does not come with any JDBC drivers. Next, start JMeter and go through the [Building a Test Plan|../BuildTestPlan] section of the User Guide to familiarize yourself with JMeter basics (for example, adding and removing elements). Finally, go through the appropriate section on how to build a specific type of Test Plan. For example, if you are interested in testing a Web application, then see the section [Building a Web Test Plan|../BuildWebTest]. The other specific Test Plan sections are for [JDBC|../BuildDbTest] , [FTP|../BuildFtpTest], [WebServices|../BuildWSTest], and [LDAP|../BuildLdapTest] . Once you are comfortable with building and running JMeter Test Plans, you can look into the various configuration elements (timers, listeners, assertions, and others) which give you more control over your Test Plans. {anchor:requirements} h2. Requirements JMeter requires your computing environment meets some minimum requirements. {anchor:javaversion} h3. Java Version JMeter version 1.8 requires a fully compliant JDK1.4 or higher. JMeter 1.8.1 and 1.9 requires a fully compliant JDK1.3 or higher. This may seem odd, but we are making a strong effort currently to be compatible with the 1.3 JDK's, though it is expected JMeter performs best with 1.4 or better. Because JMeter uses only standard Java APIs (java.*), please do not file bug reports if your JRE fails to run JMeter because of JRE implementation issues. {{JDK1.4.1Beta appears to be buggy, and there are some GUI elements that won't work correctly in this JVM.}} {anchor:operatingsystems} h3. Operating Systems JMeter has been tested and works under Unix (Solaris, Linux, etc) and Windows (98, NT, 2000, XP). Also works on OpenVMS. JMeter is a 100% Java application and should run correctly on any system that has a compliant Java implementation. {anchor:optional} h2. Optional If you plan on doing JMeter development or want to use Sun's Java Standard Extension packages, then you will need one or more optional packages listed below. {anchor:javacompiler} h3. Java Compiler If you want to build the JMeter source or develop JMeter plugins, then you will need a fully compliant JDK1.3 (1.4 for JMeter 1.8) or higher compiler. {anchor:saxparser} h3. SAX XML Parser JMeter comes with [Apache's Xerces XML parser|http://xml.apache.org]. You have the option of telling JMeter to use a different XML parser. To do so, include the classes for the third-party parser in JMeter's [classpath|#classpath], and update the [jmeter.properties|#configuring] file with the full classname of the parser implementation. {anchor:email} h3. Email Support JMeter has limited Email capabilities (it can send email based on test results). To enable Email support, add Sun's [JavaMail] packages and the activation packages to JMeter's [classpath|#classpath]. {anchor:ssl} h3. SSL Encryption To test a web server using SSL encryption (HTTPS), JMeter requires that an implementation of SSL be provided (such as Sun's [Java Secure Sockets Extension -- JSSE|http://java.sun.com/products/jsse/index.jsp]). Include the necessary encryption packages in JMeter's [classpath|#classpath] . Also, update [jmeter.properties|#configuring] by registering the SSL Provider. There is also the SSL Manager , for greater control of certs. {{If you are running JDK1.4, then you do not have to download JSSE because Sun integrated it into JDK1.4 as a standard library.}} {anchor:jdbc} h3. JDBC Driver You will need to add your database vendor's JDBC driver to the classpath if you want to do JDBC testing. {anchor:soap} h3. Apache SOAP Apache SOAP requires mail.jar and activation.jar. You need to download and copy these two jar files to your jmeter/lib directory. Once the files are in there, JMeter will automatically pick them up. {anchor:installation} h2. Installation Installing JMeter is a snap. Specifics depend on which release file you downloaded. {anchor:release} h3. Downloading the Latest Release We recommend that most users run the [latest release|http://jakarta.apache.org/site/downloads/downloads_jmeter.cgi]. To install a release build, simply unzip the zip/tar file into the directory where you want JMeter to be installed. Provided that you have a JDK correctly installed and the JAVA_HOME environment variable set, there is nothing more for you to do. {anchor:nightly} h3. Downloading Nightly Builds If you do not mind working with beta-quality software, then you can download and run the latest nightly build. To install a nightly build, unzip the zip/tar file into the directory where you want JMeter to be installed. Then, open a shell or command prompt and change to JMeter's top-level directory. Next, type "build install". Provided that you have a JDK correctly installed and the JAVA_HOME environment variable set, JMeter should be installed successfully. {anchor:running} h2. Running JMeter To run JMeter, run the jmeter.bat (for Windows) or jmeter (for Unix) file. {anchor:classpath} h3. JMeter's Classpath JMeter automatically finds classes from jars in its /lib and /lib/ext directories. If you want to add other JAR files to JMeter's classpath, then you must copy them to JMeter's "lib" directory. If you have developed new JMeter specific components, then you should jar them and copy the jar into JMeter's /lib/ext directory. JMeter will automatically find JMeter components in any jars found here. You can also install utility Jar files in $JAVA_HOME/jre/lib/ext Note that setting the CLASSPATH environment variable will have no effect. This is because JMeter is started with "java -jar", and the java command silently ignores the CLASSPATH variable, and the -classpath/-cp options when -jar is used. \[This occurs with all Java programs, not just JMeter.\] {anchor:proxy} h3. Using a Proxy Server If you are testing from behind a firewall/proxy server, you may need to provide JMeter with the firewall/proxy server hostname and port number. To do so, run the jmeter.bat/jmeter file from a command line with the following parameters: -H \[proxy server hostname or ip address\] -P \[proxy server port\] -u \[username for proxy authentication - if required\] -a \[password for proxy authentication - if required\] {{jmeter -H my.proxy.server -P 8000 -u username -a password}} Alternatively, you can use --proxyHost, --proxyPort, --username, and --password {anchor:nongui} h3. Non-GUI Mode For non-interactive testing, you may choose to run JMeter without the GUI. To do so, use the following command options -n This specifies JMeter is to run in non-gui mode -t \[name of JMX file that contains the Test Plan\]. -l \[name of JTL file to log sample results to\]. -r Run all remote servers specified in jmeter.properties (or remote servers specified on command line by overriding properties) The script also lets you specify the optional firewall/proxy server information: -H \[proxy server hostname or ip address\] -P \[proxy server port\] {{jmeter -n -t my_test.jmx -l log.jtl -H my.proxy.server -P 8000}} {anchor:servermode} h3. Server Mode For distributed testing , run JMeter in server mode, and then control each server from the GUI. {{The jmeter-server/jmeter-server.bat script should start rmiregistry for you with the appropriate classpath. if it fails to do so, \[:../Remote: read the details\] about starting the jmeter server.}} Run jmeter-server/jmeter-server.bat, plus these optional commands: The script also lets you specify the optional firewall/proxy server information: -H \[proxy server hostname or ip address\] -P \[proxy server port\] {{jmeter-server -H my.proxy.server -P 8000}} {anchor:propertyoverride} h3. Overriding Properties Via The Command Line Java system properties, JMeter properties, and logging properties can be overriden directly on the command line (instead of modifying jmeter.properties). To do so, use the following options: -D\[prop_name\]=\[value\] - defines a java system property value. -J\[prop name\]=\[value\] - overrides a JMeter property. -L\[category\]=\[priority\] - overrides a logging setting, setting a particular category to the given priority level. The -L flag can also be used without the category name to set the root logging level. *Examples:* {{jmeter -Duser.dir=/home/mstover/jmeter_stuff -Jremote_hosts=127.0.0.1 -Ljmeter.engine=DEBUG}} {{jmeter -LDEBUG}} The command line properties are processed early in startup, but after the logging system has been set up. Attempts to use the -J flag to update log_level or log_file properties will have no effect. {anchor:logging} h3. Logging and error messages If JMeter detects an error, a message will be written to the log file. The log file name is defined in the jmeter.properties file. It is normally defined as jmeter.log , and will be found in the JMeter startup directory, i.e. bin. When running on Windows, the file may appear as just jmeter unless you have set Windows to show file extensions. \[Which you should do anyway, to make it easier to detect viruses and other nasties that pretend to be text files...\] As well as recording errors, the jmeter.log file records some information about the test run. For example: {{{10/17/2003 12:19:20 PM INFO - jmeter.JMeter: Version 1.9.20031002 10/17/2003 12:19:45 PM INFO - jmeter.gui.action.Load: Loading file: c:\mytestfiles\BSH.jmx 10/17/2003 12:19:52 PM INFO - jmeter.engine.StandardJMeterEngine: Running the test! 10/17/2003 12:19:52 PM INFO - jmeter.engine.StandardJMeterEngine: Starting 1 threads for group BSH. Ramp up = 1. 10/17/2003 12:19:52 PM INFO - jmeter.engine.StandardJMeterEngine: Continue on error 10/17/2003 12:19:52 PM INFO - jmeter.threads.JMeterThread: Thread BSH1-1 started 10/17/2003 12:19:52 PM INFO - jmeter.threads.JMeterThread: Thread BSH1-1 is done 10/17/2003 12:19:52 PM INFO - jmeter.engine.StandardJMeterEngine: Test has ended}}} The log file can be helpful in determining the cause of an error, as JMeter does not interrupt a test to display an error dialogue. {anchor:configuring} h2. Configuring JMeter If you wish to modify the properties with which JMeter runs you need to either modify the jmeter.properties inside of the /bin directory or create your own copy of the jmeter.properties and specify it in the command line (jmeter \[properties file\]). *Parameters* | *Attribute* | *Description* | *Required* | | ssl.provider | You can specify the class for your SSL implementation. If you are using the JSSE from sun, then it is: com.sun.net.ssl.internal.ssl.Provider. JMeter, by default, should provide https support is you are using JDK1.4 or if you use JDK1.3 with the JSSE class jars in your JMeter classpath. | No | | xml.parser | You can specify an implementation as your XML parser. The default value is: org.apache.xerces.parsers.SAXParser | No | | user.dir | The directory JMeter will first go to for saving and loading test scripts. | No | | remote_hosts | Comma-delimited list of remote JMeter hosts. If you are running JMeter in a distributed environment, list the machines where you have JMeter remote servers running. This will allow you to control those servers from this machine's GUI | No | | not_in_menu | A list of components you do not want to see in JMeter's menus. As JMeter has more and more

components added, you may wish to customize your JMeter to show only those components you are interested in. You may list their classname or their class label (the string that appears in JMeter's UI) here, and they will no longer appear in the menus. | No |