UserManual BuildTestPlan

h1. Building a Test Plan A test plan describes a series of steps JMeter will execute when run. A complete test plan will consist of one or more Thread Groups, logic conrollers, sample generating controllers, listeners, timers, assertions, and configuration elements. {anchor:adding} h2. Adding and Removing Elements Adding elements to a test plan can be done by right-clicking on an element in the tree, and choosing a new element from the "add" list. Alternatively, elements can be loaded from file and added by choosing the "open" option. To remove an element, make sure the element is selected, right-click on the element, and choose the "remove" option. {anchor:loading} h2. Loading and Saving Elements To load an element from file, right click on the existing tree element to which you want to add the loaded element, and select the "open" option. Choose the file where your elements are saved. JMeter will load the elements into the tree. To save tree elements, right click on an element and choose the "save" option. JMeter will save the element selected, plus all child elements beneath it. In this way, you can save test tree fragments, individual elements, or the entire test plan. {{{The test plan file format changed with version 1.7. Versions later than 1.8 are no longer capable of reading test plans in the old format. If you need to use an old test plan, use JMeter 1.7 or 1.8 to load and re-save it in the new format.}}} {anchor:configuring} h2. Configuring Tree Elements Any element in the test tree will present controls in JMeter's right-hand frame. These controls allow you to configure the behavior of that particular test element. What can be configured for an element depends on what type of element it is. The Test Tree itself can be manipulated by dragging and dropping components around the test tree. {anchor:running} h2. Running a Test Plan To run your test plan, choose "start" from the "run" menu item. To stop your test plan, choose "stop" from the same menu. A small box in the top right corner of JMeter turns green when JMeter is running. Another way to check if JMeter is running is to check the "run" menu. If "start" is disabled, and "stop" is enabled, JMeter is running (or at least trying to run) your test plan. h2. Scoping Rules The JMeter test tree contains elements that are both hierarchical and ordered. Some elements in the test trees are strictly hierarchical (Listeners, Config Elements, Post-Procesors, Pre-Processors, Assertions, Timers), and some are primarily ordered (controllers, samplers). When you create your test plan, you will create an ordered list of sample request (via Samplers) that represent a set of steps to be executed. These requests are often organized within controllers that are also ordered. Given the following test tree: {{http://jakarta.apache.org/jmeter/images/screenshots/scoping2.png}} Example test tree The order of requests will be, One, Two, Three, Four. Some controllers affect the order of their subelements, and you can read about these specific controllers in the component reference . Other elements are hierarchical. An Assertion, for instance, is hierarchical in the test tree. If its parent is a request, then it is applied to that request. If its parent is a Controller, then it affects all requests that are descendants of that Controller. In the following test tree: {{http://jakarta.apache.org /jmeter/images/screenshots/scoping2.png}} Hierarchy example Assertion #1 is applied only to Request One, while Assertion #2 is applied to Requests Two and Three. Another example, this time using Timers: {{http://jakarta.apache.org/jmeter/images/screenshots/scoping3.png}} complex example In this example, the requests are named to reflect the order in which they will be executed. Timer #1 will apply to Requests Two, Three, and Four (notice how order is irrelevant for hierarchical elements). Assertion #1 will apply only to Request Three. Timer #2 will affect all the requests. Hopefully these examples make it clear how configuration (hierarchical) elements are applied. If you imagine each Request being passed up the tree branches, to its parent, then to its parent's parent, etc, and each time collecting all the configuration elements of that parent, then you will see how it works. The Configuration elements Header Manager, Cookie Manager and Authorization manager are treated differently from the Configuration Default elements. The settings from the Configuration Default elements are merged into a set of values that the Sampler has access to. However, the settings from the Managers are not merged. If more than one Manager is in the scope of a Sampler, only one Manager is used, but there is currently no way to specify which is used.