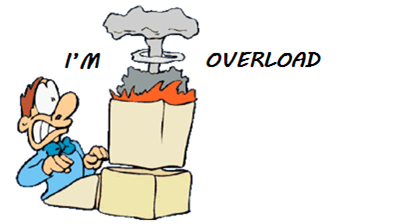
**Jmeter Timers: Constant, Gaussian Random, Uniform [Example]**

**What are Timers?**

By default, JMeter sends the request **without pausing** between each request. In that case, JMeter could **overwhelm** your test server by making too many requests in a short amount of times.

Let imagine that you send **thousands** request to a web server under test in a few seconds. This is what happens!

[](https://www.guru99.com/images/TimerOverload.png)

Timers allow JMeter to **delay** between each request which a thread makes. A timer can solve the server **overload** problem.

Also,**in real life visitors do not arrive at a website all at the same time, but at different time intervals. So Timer will help mimic the real-time behavior.**

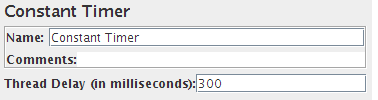
In this tutorial, you will learn

* [Constant Timer](https://www.guru99.com/timers-jmeter.html#1)
* [Gaussian Random Timer](https://www.guru99.com/timers-jmeter.html#2)
* [Uniform Random Timer](https://www.guru99.com/timers-jmeter.html#3)
* [BeanShell Timer](https://www.guru99.com/timers-jmeter.html#4)
* [BSF Timer](https://www.guru99.com/timers-jmeter.html#5)
* [JSR223 Timer](https://www.guru99.com/timers-jmeter.html#6)
* [How to Use Constant Timer](https://www.guru99.com/timers-jmeter.html#7)

Following are some **common** types of a timer in JMeter

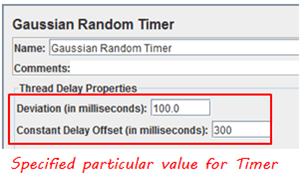
**Constant Timer:**

Constant timer delays each user request for the **same** amount of time.

[](https://www.guru99.com/images/ConstantTimer.png)

**Gaussian Random Timer:**

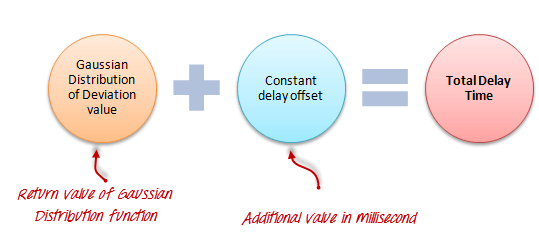
[Gaussian](http://hyperphysics.phy-astr.gsu.edu/hbase/math/gaufcn.html) random timer delays each user request for a **random** amount of time.

[](https://www.guru99.com/images/RamdomTimer.png)

**Parameters:**

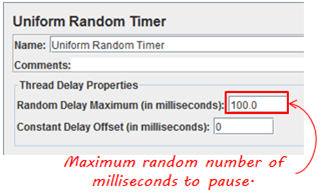
|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Name | Descriptive name for this timer that is shown in the tree |
| Deviations (milliseconds) | A **parameter** of Gaussian Distribution Function |
| Constant Delay Offset (milliseconds) | **Additional** value in milliseconds |

So the total delay is described as below figure:

[](https://www.guru99.com/images/TotalDelay.png)

**Uniform Random Timer:**

Uniform random timer delays each user request for a random amount of time.

[](https://www.guru99.com/images/UniformRandomTimer(1).png)

**Parameters:**

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Name | Descriptive name for this timer that is shown in the tree |
| Random Delay Maximum | Maximum random number of milliseconds to delay. |
| Constant Delay Offset (milliseconds) | **Additional** value in milliseconds |

**The total delay is the sum of the random value and the offset value.**

**BeanShell Timer**

The [BeanShell](http://www.beanshell.org/) Timer can be used to **generate** a delay time between each user request.

**BSF Timer**

The BSF Timer can be used to generate a delay between each user request using a [BSF](http://en.wikipedia.org/wiki/Bean_Scripting_Framework) scripting language.

**JSR223 Timer**

The JSR223 Timer can be used to generate a delay between each user request using a [JSR223](http://en.wikipedia.org/wiki/JSR223) scripting language

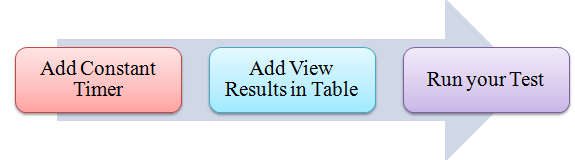
**How to Use Constant Timer**

In this example, you will use **Constant Timer** to set **a fixed delay** between user requests to google.com.

Let start with a simple test script

1. JMeter creates **one** user request to [http://www.google.com](http://www.google.com/) **100** times
2. **Delay** between each user request is **5000** ms

Here is the **roadmap**for this practical example:

[](https://www.guru99.com/images/FlowTimer.png)

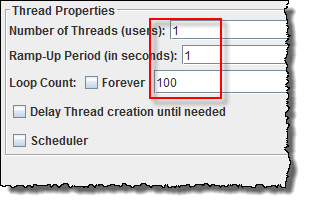
**Pre-condition:**

We **re-use** the Step 1 and Step 2 in tutorial [JMeter Performance Testing](https://www.guru99.com/jmeter-performance-testing.html).

**Step 1) Add Thread Group**

Right click on the[Test Plan](https://www.guru99.com/what-everybody-ought-to-know-about-test-planing.html)and add a new thread group: **Add**-> **Threads (Users)** ->**Thread Group**

In Thread Group control panel, enter Thread Properties as following

[](https://www.guru99.com/images/ThreadProperty.png)

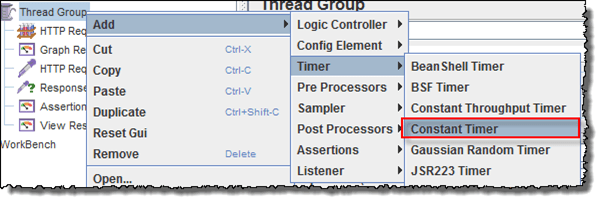
This setting lets JMeter create **one** user request to [http://www.google.com](http://www.google.com/) in **100** times

**Step 2) Add JMeter elements**

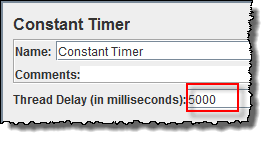
* Add HTTP request default
* Add HTTP request

**Step 3) Add Constant Timer**

Right-click **Thread Group *->* Timer *->* Constant Timer**

[](https://www.guru99.com/images/AddConstantTimer.png)

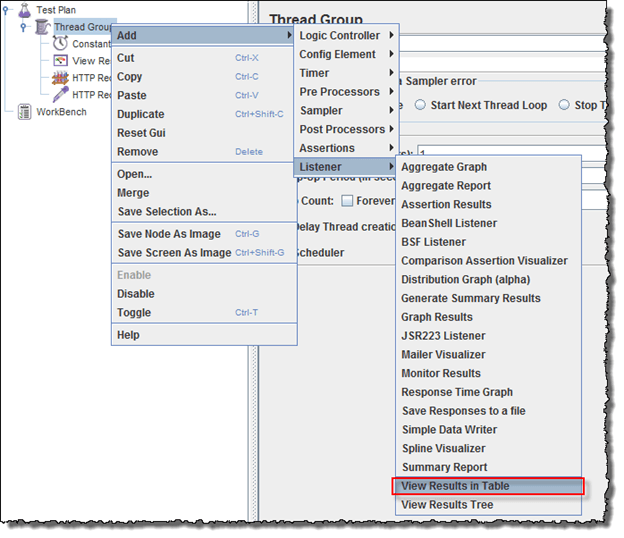
Configuring Thread Delay of 5000 milliseconds

[](https://www.guru99.com/images/ThreadDelayConstantTimer.png)

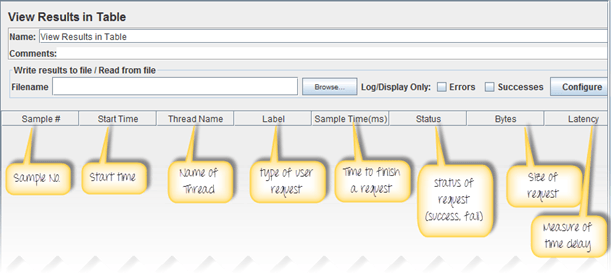
**Step 4) Add View Results in Table**

View Results in Table displays the test result in table format.

Right click **Add *->* Listener *->*View Result in Table**

[](https://www.guru99.com/images/AddViewResult.png)

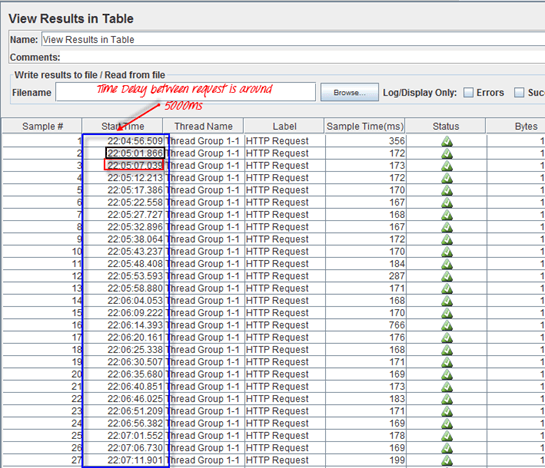
View Results in Table displays as below figure

[](https://www.guru99.com/images/AddViewResultTable.png)

**Step 5) Run your test**

When you ready to run a test, click **the Run** button on the menu bar, or short key **Ctrl+R**

This is the result of this test

[](https://www.guru99.com/images/RunYourTest.png)

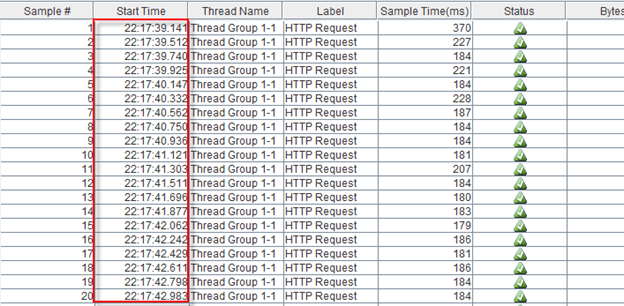
For example, in the above figure, let analyze the **Sample 2**

* **Start time**is 22:05:01.866
* **Sample Time** of Sample 2 is 172 ms
* **Constant Timer**: 5000 ms (as configured)
* **End Time** of this sample is = 22:05:01.866 + 172 + 5000 = 22:05:07.038

So the Sample 3 should start at the time is **22:05:07.039**( As shown in the above figure)

The **delay** of each sample is **5000** ms

If you change the Constant Timer is **zero**, you will see the result is changed

[](https://www.guru99.com/images/SampleTimer.png)

Let analyze the **Sample 1**

* **Start time**is 22:17:39.141
* **Sample Time** of Sample 2 is 370 ms
* **Constant Timer** : 0 ms (as configured)
* **End Time** of this sample is = 22:17:39.141+ 370 + 0 = 22:17:39.511

So the **Sample 2** should start at the time is **22:17:39.512** (Shown in the above figure)