

Getting started

Table of contents

1 The Grinder processes.....	2
2 How do I start The Grinder?.....	2
2.1 Network addresses.....	3
3 Output.....	3

Note:

Please read [Should I use The Grinder 2 or The Grinder 3](#) (../faq.html#g2vsg3) ?

1. The Grinder processes

The Grinder is composed of three key processes:

- **Worker processes**
 - Perform the tests using a plug-in
- **Agent processes**
 - Manage worker processes
 - A single agent process runs on each client machine
- **The console**
 - Coordinates the other processes
 - Collates and displays statistics

As The Grinder is written in Java, each of these processes is a Java Virtual Machine (JVM).

For heavy duty testing, you can start an agent process on each of several client machines. The worker processes they launch can be controlled and monitored using the console. There is little reason to run more than one agent on a single machine, but you can if you wish.

2. How do I start The Grinder?

Its easy:

1. Set your CLASSPATH to include the `grinder.jar` file in the `lib` directory.
2. Start the console:


```
java net.grinder.Console
```
3. Create a `grinder.properties` file which defines the test you want to perform. See the `examples` directory for inspiration.
4. Start an agent process:

```
java net.grinder.Grinder
```

The agent process forks child Java processes to do the work. You can also specify an explicit properties file as the first argument. For example:

```
java net.grinder.Grinder myproperties
```

The console does not read the `grinder.properties` file. It has its own options dialog (choose the *File/Options* menu option), which you should use to set the communication addresses and ports to match those in the `grinder.properties` files.

When the child processes start, they inform the console of the tests they will run. If you start the console after the agent process, you should press the *Reset processes* button. This will cause the existing worker processes to exit and the agent process to start new child

processes, which will update the console with the new test information.

2.1. Network addresses

The worker processes listen for console signals on a multicast address, by default this is 228.1.1.1:1235. Each worker process sets up a TCP network connection to the console to report statistics. By default, the console listens on port 6372 on all local network interfaces of the machine running the console.

If the default multicast addresses are not valid, alter following properties in the `grinder.properties` file before starting the Grinder agents

```
grinder.consoleAddress (Address of machine running console)
grinder.consolePort
grinder.grinderAddress (Multicast address)
grinder.grinderPort
```

3. Output

Each worker process writes logging information to a file called `out-host-n.log`, where `host` is the machine host name and `n` is the worker process number. Errors will be written to `error-host-n.log`. If no errors occur, an error file will not be created.

Data about individual test invocations is written into a file called `data-host-n.log`. This can be imported into a spreadsheet tool such as Microsoft Excel™ for further analysis.

The final statistics summary (in the `out-*` files of each process) looks something like this:

Final statistics for this process:

	Successful Transactions	Errors	Average (ms)	
Test 0	25	0	255.52	
Test 1	25	0	213.40	
Test 2	25	0	156.80	"Image"
Test 3	25	0	90.48	
Test 4	25	0	228.68	"Login page"
Test 5	25	0	86.12	"Security check"
Test 6	25	0	216.20	
Test 7	25	0	73.20	
Test 8	25	0	141.92	
Test 9	25	0	104.68	"Logout page"
Totals	250	0	156.70	

The console displays a similar dynamic display of information collected from all the worker processes.

Each test has one of two possible outcomes:

1. Success. The number of *Successful Transactions* for that test is incremented. The time taken to perform the test is added to the *Total*.
2. Error. The exact interpretation of an error depends on the plug-in. The number of *Errors* for the test is incremented.

The *Total* and *Average* figures are calculated based only on successful transactions.

