

The Grinder To Do list

Table of contents

1 Enhancements.....	3
1.1 Console API.....	3
1.1.1 API.....	3
1.1.2 what about access from grinder scripts?.....	3
1.1.3 sample log file.....	3
1.1.4 feedback from travis.....	3
1.2 Script Distribution.....	3
1.2.1 Model.....	4
1.3 Console.....	5
1.3.1 Stop the recording when last thread terminates.....	5
1.3.2 Refactoring.....	5
1.3.3 Add log panel.....	5
1.3.4 Future editor features.....	5
1.4 Engine.....	6
1.4.1 Interactive mode.....	6
1.4.2 Instrumentation API.....	6
1.5 Communication.....	6
1.6 Statistics.....	6
1.6.1 Aligning data files.....	6
1.6.2 Separate out console statistics views from summary statistics views.....	6
1.6.3 max(), min().....	7
1.7 TCPProxy.....	7
1.8 HTTP Plugin.....	7
1.9 Scripting.....	7
1.9.1 Alternate test languages.....	8
1.9.2 Events.....	8
1.9.3 Error reporting.....	8
1.10 Reports.....	8
1.11 Code.....	8
1.12 Other HTTP/HTML libraries.....	8
1.12.1 Summary.....	9

2 Bugs.....	9
2.1 Console/agent.....	9
2.2 TCPProxy.....	10
3 Build.....	10
4 Documentation.....	10
4.1 Examples.....	10
4.2 Forrest TODO.....	10

This is The Grinder TODO list. Its a collection of my thoughts on the future of The Grinder.

If you want to implement something off this wish list, let me know first by discussion your ideas on the grinder-development list. There are instructions on how to contribute at <http://grinder.sourceforge.net/development/contributing.html>.

The TODO list is part of the binary distribution. I'll accept patches against it too :-).

- Philip Aston

1. Enhancements

1.1. Console API

1.1.1. API

Process control

```
void startWorkerProcess(some id) void stopWorkerProcess(some id) List<ProcessStatus>  
getProcessStatus() - NB return value void  
addProcessEventListener(ProcessEventListener)
```

```
void setScript(...?)
```

Recording control (done).

```
void startRecording() void stopRecording() void resetRecording()
```

1.1.2. what about access from grinder scripts?

1.1.3. sample log file

1.1.4. feedback from travis

Comments on the API: -

- * For sure there will need to be an ability to retrieve stats, as you mentioned.
- * A file transfer API, or some other configuration management API will be needed so that a centrally managed config, such as might be included in `grinder.properties`, can be pushed out to the agents.
- * Right now I'm using scripted ssh calls to start the agents on the remote machines. It would be better if there was a good, generic, cross-platform way the controller/console could initiate a connection to a given agent. This might mean having the agent process running as a server/service on the agent machine, so that it would always be available.

1.2. Script Distribution

1.2.1. Model

Agent process to manage file cache. (Need files as Jython can't have python path into memory). Agent is responsible for local file store below a directory based on the agent name.

File cache should be up to date with directory structure below directory chosen by console. The root directory is part of python path.

1.2.1.1. Tasks

Release 1 (done)

Task: Agent receives asynchronously. Task: Console to broadcast. Remove ScriptDistributionFiles work. Task: Allow editing in the console. Task: DistributeFilesMessage -> DistributeFileMessage (send a single file). Send in a background thread with a progress bar. Task: Console to maintain map of remote agent address to last update time. For now assume that agent cache should be overwritten. For now, find the oldest time and broadcast since then. Null -> send the lot. Task: Console to clear cache state on root change -> broadcast full cache on distribution. Need a special "clear cache" message. Task: Console to allow selection of script. Warn on distribution if no script selected. Task: Agent to receive selection of script. Task: Agent FileStore to update copy of cache asynchronously. Main agent thread to move it into place at good point in lifecycle to prevent locking of stale files by worker processes. Task: Agent shouldn't create filestore directory until necessary. Task: Setting a new directory should require agents to refresh their cache.

Release 2 (done)

Task: File Store directory should have a README describing what its for. Task: Guard against distributing agent cache directory. Can mark with a special file - maybe the README? Also prevent distribution of temporary files, CVS directories, grinder log files.

Release 3 (done)

Task: Mature FileDistributionHandler model. Extract from comms package, leaving the comms bit of it behind. Task: Model agent cache state. Task: Console to indicate global dirty state. (Distinct from buffer "dirty" state). Task: Console to check dirty state on play and warn. Task: Console should not enable worker process controls if no agent is connected. Task: If dirty buffers on play, optionally warn.

Release 4 (done)

Task: Merge worker and agent process status models. Update process tab to show both.

Release 5 (done)

Task: Add warning if saving outside of distribution. Task: Console to watch local file system for edits, trigger need to distribute. Task: Regularly update the tree to match file system. Task: Write some tests for FileTree. Task: Warnings for buffers whose file is

modified outside of editor.

Release 6 (done)

Task: External editor integration. Task: The agent should pick up secondary grinder.properties from start message. Task: Console to optionally auto broadcast on play.

Release 7

Task: Addressed messaging. Every Acceptor.SocketResource to know its remote EndPoint. Add FilteredFanOutServerSender (Extend FanOutServerSender, allow resourceToOutputStream to return null indicating the resource shouldn't be sent anything. Supplied Filter method is given a SocketResource and the Message and can then veto.)

Task: Add agent initiation channel so agent can send a checksum across file names/sizes. Console compares with its cache and generates a good last update time for the agent. Thus agents starting with a good cache skip sync. Can't rely on remote timestamps as clocks are not sync'd.

Task: Distribution file filter should be dynamically settable.

1.3. Console

1.3.1. Stop the recording when last thread terminates

Requested by Jirgen Weber

1.3.2. Refactoring

Consider moving overwrite / save before close / ... handling to the model. Needs some kind of command pattern to represent choices.

1.3.3. Add log panel

Report log messages that currently go to terminal, plus start, stop test runs etc. Logs should be time stamped. Use log to replace use of System.err for warnings.

1.3.4. Future editor features

Revert file. Status bar. Undo. Copy and Paste menu items.

Specialised grinder properties editor.

1.3.4.1. jEdit

Replace jEdit-syntax with new jEdit syntax package when available, if its license terms are acceptable. Apparently now available.

Clause 2b:

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at

no charge to all third parties under the terms of this License.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

1.4. Engine

There should be an ExternalFilenameFactory, cf ExternalLogger.

1.4.1. Interactive mode

Possible to have an interactive mode for debugging? E.g. a simple UI that can be launched from a thread which drops the user into a console? Would need to launch with `grinder.debug.singleprocess=true`.

1.4.2. Instrumentation API

Could we support a procedural "startTest()", "stopTest()" API?

Perhaps. This procedural style would certainly be more obvious to the average user than "wrapping". OTOH, wrapping is powerful and I'm not sure we want to support two mechanisms to instrument code.

1.5. Communication

Resolve anomaly that Senders throw exceptions but Receivers return null.

1.6. Statistics

Should TPS averages be harmonic means? http://en.wikipedia.org/wiki/Harmonic_mean

Split out serializable kernel of TestStatisticsMap? Look hard at where the message is deserialised.

public ExpressionView ctor should probably be deprecated.

1.6.1. Aligning data files

Requested by Jose Antonio Zapata:

Add an additional "milliseconds since the Epoch" entry to the data files.

1.6.2. Separate out console statistics views from summary statistics views

Needed for custom statistics that use 'period' so are valid in the console but not in the process logs.

1.6.3. max(), min()

(Requested by Venelin Mitov).

Should store max, min values against SampleStatistics, and add max(), min() to expressions.

1.7. TCPProxy

Tim McNerney writes:

- > Obviously, I could edit the scripts by hand. But I'd like to have
- > TCPProxy do this for me. So is there some existing method for doing
- > such filtering? Say on target suffix ("filter=.gif,.js,.css")

Change TCPProxy filters to a stream oriented model. This should cure another one of the TCPSniffer/HTTPPlugin bug when recording large outputs with posts? Also, consider having a filter instance pair per connection.

Internationalise messages.

Support different client certificates for proxied connection.

Meo Bogliolo writes:

- > What do You think about adding the possibility to insert remarks
- > with the TCPProxy Console? Maybe it's usefull in complex
- > navigation... I currently look at think time in the scripts to
- > understand when the user "changes" page.

Allow the filters to optionally parse unknown command line options. They would also then have to be able to contribute to the help text. This would allow the http plugin to add options to specify a different filename, and also to specify stdout.

Failed connection events.

1.8. HTTP Plugin

Remove ParseException, ProtocolNotSuppException from public APIs.

Script support for HTTP "system property" options.

Pablo Estades Fernñ;½ndez says:

- > I need to use client cert on TCPProxy to be able to
- > record the test case but also I need to configure
- > grinder workers to run the test case presenting a
- > client cert.

1.9. Scripting

Consider forcing TestRunner to be registered with grinder.

Add per-run statistics. This would also allow number of aborted runs to be recorded.

Script access to global statistics.

1.9.1. Alternate test languages

Should support Java, Groovy scripts.

Should use AOP proxies to do test instrumentation. CGLib, or raw ASM?

Blocker: Can't wrap around an existing instance that doesn't have a default constructor. This is a general problem. How to create a new proxy that supports all the methods + interfaces - it clearly needs to extend the target. What if we don't care about implementing the interfaces - just create a proxy that supports the same methods as its delegate? CGLib doesn't appear to support this.

1.9.2. Events

Idea from Nurul Choudhury:

- > Event counting - The Jython code can create a named event and fire
- > the event when some condition was met. When the console polls for
- > statistics the events and their count would be sent to the console.

1.9.3. Error reporting

- > It would be possible for certain classes of error (AttributeError
- > being a good example) to spew out just
- >
- > Unknown attribute GETx at "http.py", line 9 in __call__.
- >
- > Is this what you're after?

To do this, might have to behave differently with 1 thread vs many.

1.10. Reports

Perhaps JasperReports?

1.11. Code

Review use of Executor. Probably want to share them.

Remove ThreadLocal from RegisteredPlugin.

1.12. Other HTTP/HTML libraries

DeSouza, Edwin writes:

- > Instead of using:
- > <http://www.innovation.ch/java/HttpClient/index.html>
- > <<http://www.innovation.ch/java/HttpClient/FAQ.html>>
- >
- > How about using Jakarta Commons HttpClient (more popular and Apache
- > License):
- > <http://jakarta.apache.org/commons/httpclient/index.html>

Justin Spears writes:

- > I am new to grinder, however I found the built in HTTPClient a
- > little lacking in functionality. I might suggest using
- > `httpunit` (<http://httpunit.sourceforge.net>) instead. It
- > works well with jython, and has extremely powerful methods
- > for handling links, posts, gets, etc.
- >
- > It uses `nekohtml` to parse malformed HTML documents into
- > valid XML then exposes a useful DOM based on these results.
- >
- > ...

OK, this amounts to a campaign against HTTPClient!

Reasons for HTTPClient:

- Its solid, (and not 'alpha' which is the case for HttpClient).
- Its small and comprehensible.
- It is efficient.
- Its extremely well written.
- Its the incumbent.

Reasons for Commons HttpClient:

- Its actively maintained.
- It is more modular.
- It is richer.

Reasons for HttpUnit:

- `nekohtml`, parsing support

I prefer HttpClient, HTTPClient over HttpUnit for The Grinder as they are "closer to the wire".

1.12.1. Summary

On balance, yes HttpClient looks good and we should use it if it proves to be efficient. I'll add it to the TODO, but its a significant change => low priority.

Update (Oct 05): Doubts about HttpClient's scalability:

http://sourceforge.net/mailarchive/forum.php?thread_id=8372852&forum_id=2649

We perhaps need to look more closely at parsing support for functional assertions, but I don't want to lose The Grinder's efficiency here.

Also http://www.nogoop.com/product_16.html#compare

2. Bugs

(In addition to those on Sourceforge).

2.1. Console/agent

Each time L&F changes, the saveAs dialog gets another All Files filter!

Add reset console action.

2.2. TCPProxy

Should listen on all interfaces if -localhost is not specified.

3. Build

How to build clover reports during full build?

- currently "with.clover" only works if clover is first thing on path, we need to do two "compile", "test" runs with different classes.dir.
- could have with ant-call, modify compile, compile-tests, test
- better to ant ourself?

Include Clover history?

4. Documentation

grinder.debug.singleprocess

HTTP plug-in.

4.1. Examples

HTTPS.

Basic authentication.

<http://thread.gmane.org/gmane.comp.java.grinder.user/186>

4.2. Forrest TODO

Create back links from javadoc. This isn't trivial, e.g. this:

```
<bottom> <![CDATA[ <a class="noborder" href="../whats-new.html"
target="_top"></img></a> ]]> </bottom>
```

only works for top level javadoc.

Create menu links to the javadoc.

How to include arbitrary HTML (e.g. for poll forms?)