Todo List

1. high

- **[code]** Reimplement HTTP with Commons HttpClient, which is hopefully less annoying these days. # open
- **[code]** When an uncaught exception occurs, need a way to shut down Tomcat. There are too many situations where Anteater never exits, and therefore can't be used safely in situations, like cron jobs. # open
- **[code]** Implement a decent test grouping system, so tests can be grouped by timestamp or 'group'. # open
- **[code]** Should the default protocol be 1.0 or 1.1? Won't virtual hosts require 1.1? Currently Anteater doesn't implement chunked transfer, making it an incomplete HTTP 1.1 implementation. This needs fixing. # open
- **[docs]** Links to sections are broken, because they incorrectly use <elementref> links. # open

2. medium

- **[code]** Add PDF support to the Blob matcher. # open
- **[code]** Rework the Matcher API so that the 'matched object' may be set from a parameter, instead of only from the HTTP response. Then we could have multi-level matchers. Eg, for the XML document: `<foo><bar>some new value</bar></foo>` We could have:

```
<matcher>
  <xpath select="/foo/bar" assign="m1"/>
  <regexp applyto="m1" pattern="some (.*) value" group="1"
    assign="m2"/>
  <contentEquals applyto="m2">new</contentEquals>
</matcher>
```

  # open
- **[code]** Avoid strings in XML parsing code wherever possible, and stick to bytes. This avoids lots of sticky encoding and i18n issues. See http://doctypechanger.sourceforge.net/#doc.CharVSByte. # open
- **[code]** Currently, <listener> is a blocking, once-off task. It waits for an incoming request,

serves it, and finishes. It would be useful to have a non-blocking version which responded identically to multiple requests. # open

- **[code]** Write an xmlvalidate task that validates XML against a DTD. Possibly use DoctypeChanger to set the doctype. # open
- **[bugfix]** It is not possible to `<servletContainer action="stop"/<` and then start it again. Without this, stopping is a bit pointless. # open
- **[bugfix]** Bad stuff ("Null incoming request") happens when we declare two identical listeners next to each other. Need to fix this error condition. # open
- **[docs]** Implement stylesheet support for referrals, where an element's documentation inherits from another. Eg, so soapRequest docs don't duplicate httpRequest. Should be able to create 'virtual' elements, eg to model ActionTask, from which real elements inherit. UPDATE: - see expand.xsl, which mostly does this. # open
- **[docs]** Investigate alternative DTDs for documenting XML. Eg http://opensymphony.com/webwork/tag.jsp. Forrest apparently supports alternative DTDs. # open
- **[docs]** Investigate Ant's xdoclet-based method for generating documentation, in order to keep Javadocs and XML docs in synch. # open
- **[code]** Fix up the group instantiation. Currently there is a Group object, and DefaultGroup descendant. It's pretty messy, and I'd like to refactor so there's a GroupFactory which creates either a NormalGroup or a DefaultGroup (both extending BaseGroup), depending on the id attribute. # JT
- **[code]** Add mustMatch attributes to all matchers. This is used to invert the sense of the matcher, ie a logical NOT. Currently only a few have this. # open
- **[logging]** Logging has become a complete mess. Need to redo the whole thing with Logkit or log4j. # open
- **[logging]** Reimplement Xalan write-new-file redirects with XSLT1.1 functions # open
- **[reporting]** Stylesheets need to report log messages somehow, and store the expected and actual text as properties. # open

3. low

- **[code]** Allow us to have nested <match> elements, so we can do: <match> <responseCode value="200"/> <match> <header name="contentType" assign="ct"/> </match> </match> and then the 'ct' variable is only set if responsecode=200. # open
- **[code]** Adopt the Commons Discovery API for finding implementations of APIs, like Loggers and RegexpUtil implementations. # open
- **[code]** Allow logger to take a nested <classpath>, to help when users specify their own loggers. # open
- **[code]** Set http.proxyHost and http.proxyPort system properties through standard Group properties. # open
- **[code]** Think about whether it's a good idea to have a completely new Anteater-specific

XML format: <anteater version="1.0"> <target ../> </anteater> That gets styled into an Ant script. This way, users could omit stuff like the taskdef and typedefs, and we could (in theory) change the underlying engine without affecting existing users, so long as we supply stylesheets to upgrade old scripts. # open

- **[code]** Perhaps add a <log> tag, so users can append to the log file halfway through a test. One motivation for this is procedural-style scripts using <antcall> calls. Here, it is more sensible to report per step than per target. # open
- **[error handling]** Anteater tasks are frequently located inside <parallel> tasks. BuildExceptions do not propagate outside <parallel>'s *until* all <parallel>'ed tasks have finished. This is a big problem when we have:

```
<http debug="1" description="">
  <parallel>
    <listener>
      ...
    </listener>

    <httpRequest>
      ...
    </httpRequest>
  </parallel>
</http>
```

If <httpRequest> fails (eg, we left out the path attribute), then we have a kind of deadlock. <httpRequest> is dead, but <listener> won't exit until it gets a request from the now-dead <httpRequest>. Not sure what to do about this.. # open

4. dream

- **[code]** Integrate graphs of Anteater script targets into the HTML reports, with the VizAnt project. # open
- **[code]** Map a Cocoon webapp's URI space to a virtual file system (Myrmidon has VFS support). Then rewrite the Matcher API so that matchers work on the VFS elements. See http://marc.theaimsgroup.com/?t=102665208500004&r=1&w=2 and http://jakarta.apache.org/ant/ant2/VFS.txt. This would allow us to match on files using a <file> element. # open
- **[code]** Validation and error reporting needs thinking about. Ant offers no sort of task validation to ensure that tasks have had their requisite properties and child nodes set. Therefore all tasks need to check their own attributes before execute()'ing, and throw BuildExceptions if they're incorrect. I don't think we're doing this 100% yet.. # open