Table of Contents

Chapter 1. High Priority Enhancements, Bugs, Features and New API migration

--------------- Submitter: SubmitDate: Class: (bug or enhancement) AssignedTo: AssignedData: CompleteData: Description: (detailed, including modules and how to reproduce). -----------------

ApplicationData make joltpage methods final url-encoding in jolt.

High Priority Enhancements:

- POST handling: - New intefrace for POs
- Querys without values (xxx.po?45,55) cause errors. some browser will do xxx.po?abc=def?45,55 additionalArgs???: ordered array of strings.
- request.getPresentationPath() is poorly defined. It returns the servlet root, not the presentation path. Provide both.

Bugs:

- markd 1 Feb 1998:
  - httpPresentation/DirectoryException was missing. Cause hang, not reported.
- markd 15 Dec 1997: A Exception.getMessage is not enough to describe an exception. In many cases we create new exceptions and only pass the message, but the message tends to be meaningless without the class name of the exception. markd
- markd 15 Dec 1997: Presentations are cached forever.
- markd 15 Dec 1997: Jolt compiler should map error messages.
- paul 15 Dec 1997: currently must explicitly type "bin/DemoApp" from the install directory to launch app server. Should be able to run from any directory.
- paul 15 Dec 1997: makefiles cannot detect which *.jhtml files have been changed. Must do a "make clean" or remove the Presentation.java sub-directory manually.
- paul 15 Dec 1997: Must kill and re-start the server everytime you make a change to *.jhtml files. There's got to be a better way.
- paul 15 Dec 1997: the server doesn't appear to start on Windows 95 if there is not an active TCP/IP session happening.
- markd 16 Dec 1997: Referencing docroot in a URL causes a null pointer exception. Do

  ```
  http://localhost:10000/servlet/harmony
  ```
  using httpPresentation/tests/presrun.
- markd 16 Dec 1997: If a *empty* static files is served under the servelet runner, the browser hangs. This occurs after the static file PO has returned, so may be a servlet runner bug.
- kyle 15 Dec 1997: demoApp can hang on win95. Kristen disovered that the following sequence of po's causes the hang.

  - login
  - logout
  - restricted page
  - home page
  - login (hangs here loading page)
- kyle 15 Dec 1997: need to verify that multi-valued args are being parsed and handed to po's correctly
- markd 16 Dec 1997:
  - BaseUser interface should probably be just User for consistancy with Session.
- markd 17 Dec 1997: POST parameters should be available via the getParameter methods in Jolt pages.
- soheil 14 Jan 1998: Users are logged out at Harmony.SessionManager.SessionMaxIdle time, whether they are active or not.

Features:

- markd 17 Dec 1997: Need utility to explictly process POST parameters in PresObjs.
- markd 17 Dec 1997: Define our own class loader to manage presentations. I can track the size of presentations that are loaded and managing cache flushing. Also will not have to require CLASSPATH to be set for the application.
- paul 15 Dec 1997: Another feature request arose today - Harmony should have a configuration option that turns of request logging to a specified log file. What info is requested should be configurable. This is the same functionality that the web server can furnish, but it is written on the application server rather than the web server.
- markd 15 Dec 1997: Add an application-specific handler for errors.
- kyle 15 Dec 1997: Installation should test for installed jdk and jre (or prompt the user) instead of requiring hand editing of *.mk files.
- kyle 15 Dec 1997: create utilities to shutdown the server (other than ps and kill).
- markd 18 Dec 1997: Should have a logging interface/based object in the same way that the SessionManager is implemented. Allow a site to plug in their own logging or extend ours.
- markd 18 Dec 1997: Need to see how harmony handles dropped connections during a request.
- kyle 13 Feb 1998: HarmonyApplication.startup() should be called at servlet init(). Or we need an equivalent init() method in HarmonyApplication. Allows application to do initial setup before first request comes in.

New API migration:

- MIME I/O routines are needed in server/pres/Http*Stream.java.
- Methods speific to file download in JoltRequest.
- Create mime package.

- change @since jolt1.0 to be just since 1.0, since its not all Jolt.
- Headers needed for a lot of files.
- Need shutdown for session and user managers.
- service and application root are specified both in the servlet config and the servlet.properties. Problely should only come from one place.

Chapter 2. Document

Below is a document written by MarkD, around April 98. Caveat emptor.

There is a new module, `Lutris/LBS/Tracking', which has HTML pages for tracking bugs. Create a new html page just for the multiserver. Prioritizing would be useful, but the most important thing is to get everything in; even if your not sure, just add an `investigate' item. This will be a useful tool to alow me to try and get more of your bandwidth.

- Generally, the class-level documentation, where completed, is very good and has the right level of detail. Its uncesssary to repeat @see references in the methods that are in the class documentation. However @see references can directly reference a methods in other classes. For instance, in a the doc for a getContentLength() method of you have
    - @see javax.servlet.ServletRequest following this link would only take one to the doc one the class, if
    - @see javax.servlet.ServletRequest#getContentLength is used, the link goes directly to the package.
- I suspect that too much addressing happens in symbolic id name space rather than as reference. Symbolic ids are required when interface with things outside of java; i.e. display and external selection. Within Java using symbolic ids is painful and inefficient. Always requiring converting to a reference, with the danger of that failing. There is no compile-time type checking. The overhead is probably dozens if not hunderds of VMI instead of a few. I would like an investigation into a reference based interface This doesn't mean eliminating ids, rather that one converts from ids to references when every required, then everything else is done by references. This means, given an id, there is a way to translate to its object and given an object, to its id.
- Some more investigation need to be put into the why ids are allocated and managed (re explicted vs generate unique).
- General use of `description' fields for objects that have an id would be very useful in building user interfaces to this functionallity. Thus every object that can be displayed/manipulated by through a user interface would have and symbolic id and a description.
- How does a filter discover that the channel it is assoicated with is being deleted? Generally, deleting is hard to do.
- Why is channel status needed? Why not just get the information directly from the channel.
    - Note from Andy: so you get a coherent snapshot of the state of the channel, rather than having to make multiple method calls. The state won't change durring the time it takes to examine the channel.
- Built-in functions to reverse navigate the object tree would be useful (given a servlet, what is its channels), but lower priority, this can be obtained by a search.
- As we discussed, got to follow Sun coding conventions or punishment will be .....
- Most places synchronized statements have been used, synchronized methods should be used. Its more straightforward and provides documentation of the behavior of a method.
- Use of fields name `myTable', doesn't add any value in understanding the code. Fields names should describe what they stored, not how they store it.
- A fair number of classes, such as BasicTransactionFilter, import classes from their own package. You don't do

need to do this in Java and it its confusing, since its not something java programmers do, its a no-op.

- Configuration mechanisn needs to be redone. Need to use the Config object. Use of keyword value tables is inapproriate. All of the should be stored in class with fields for each value. Hash tables or keyword value table can be used to find these objects. The config files should be read in advance and used to create the objects. Most likely, these objects are just the objects being configured.
- Three versions of the configuration files and startup scripts means they get out of sync. They need to be generated or the correct one picked up the the start script. (They where wrong!).
- Need to review and understand all error handling.

You probably know these:

- Diagram needs package names updated and should be updated. Its very valuable.
- Some classes still need doc-ed.
- The mutiserver directory needs reorganized.

connectionMethod:

- ConnectionMethodManager
    - All methods do explict synchronization on `myTable', instead declare these functions as synchronized.
    - `myTable' isn't a very meaningful field name.
    - As the comment says, it should be a hash table (only because its less code to write, performance can't matter since there is so few).
- BasicConnectionMethod
    - Documentation is inconsistent as it says its a sample implmentation and yet it doesn't do anything without being extended.
    - In the tree we have used the `Basic' name for classes to indicate they provide the basic level of functionallity and extending the object is optional. `Base' would be a better name.
- ChannelStatus
    - If its necessary to warn about not modifying contents, provide access (get) methods instead of making fields public. Although the only place I see that there is a danger in modifying it is with the filterIDs array. I would suggest cloning the array. Everything else will not affect the channel; in this case, its a real just a struct and the warnings should be removed. Another thing that might be useful is making the fields `final'.
- Channel
    - For booleans functions, isEnabled() would probably be better than getEnabled().
- ChannelTable

  - Method should be synchronized, not synchronize on a field.
  - Indentation is a mess.
- RMIConnectionMethod RMIConnectionMethodImpl
  - The implementions of the connection methods would be better off in the particularly methods package.
- WrapResult
  - Not a very meaningful name, took a while to figure out how it was used. In generally, there term `warp' is not good because this isn't what your doing. Its not warpping, as a filter is only intercepting one side of object it is wraping. System-V streams is a good anology to what your doing. Really you are pushing and popping filters off of a stack and data is flowing as a stream through the filters. Revisit the use of the name `wrap' in both this object and the method name.
  - Note from Andy: The filters do wrap (not warp :) the objects. They totally contain the next object in the chain, usually as instance variables. They can execute code before and after they pass the call on to the other object. The geometrical analogy is circles within circles within circles, not a linear list of boxes next to each other.