ViewerProxy Design

Contents

This section describes the viewerproxy control resolver, the special viewerproxy access via urls and the observer resolver.

The viewerproxy uses the Jetty HTTP server library to handle connections. Each incoming URL is sent through a pipeline of "resolvers", each of which can either process the URL or pass it on to the next resolver. Each resolver must extend the abstract class **CommandResolver**. The **executeCommand** method must be overridden to handle requests, and should return true if the requests was handled by this resolver. The resolver is responsible for calling **response.setStatus** to set the appropriate HTTP result code.

Incoming URLs are handled by the resolvers in the following order: Viewerproxy control resolver, GetDataResolver, NotifyingURIResolver.

Viewerproxy control resolver

The HTTPControllerServer class manages index setup and missing URL collection for the viewerproxy. It is mainly used through the QA web interface. It has the following commands:

- start recording URIS (**/startRecordingURIs**)
- stop recording URIS (**/stopRecordingURIs**)
- clear collected URIS (**/clearRecordedURIs**)
- get collected URIS (**/getRecordedURIs**)
- change index (**/changeIndex**)
- get status (**/getStatus**)

Special Access

Via URLs

The GetDataResolver class provides some special URLs in the viewerproxy that can be used for more direct access to the stored data. To use them, your browser must be set up to access the viewerproxy in the same way as when browsing harvested data. The general format of the commands are `http://viewerproxy.invalid/<command>?arg1=value1&arg2=value2…`

The commands are:

- getFile - gets a whole file from the archive
- arcFile=<filename> - name of the file (without pathnames)
- getRecord - gets a single ARC record from the archive
- arcFile=<filename> - name of the file to look up a record in (without pathnames)
- arcOffset=<offset> - offset into the file the record starts at
- getMetadata - gets all metadata for a job from the archive
- jobID=<id> - ID (numeric) of the job for which to fetch metadata

Via Command Line

Futhermore it is possible to make getFile and getRecord commands from the command line the following way:

```
usage: java dk.netarkivet.archive.tools.GetFile <filename> [destination-file]
```

This tool retrieves a file from the archive. If the **destination-file** is omitted, the file is stored with the same name. The bitarchive replica the file is retrieved from is chosen based on the setting **settings.archive.common.useReplicaId**.

```
usage: java dk.netarkivet.archive.tools.GetRecord <indexdir> [uri]
```

This tool depends on the existence of a luceneindex, as generated by the index server. It will use this index to lookup the arcfile and offset to get a particular record from. It will then retrieve that record from the archive.

The bitarchive replica the file is retrieved from, is chosen based on the setting *settings.archive.common.useReplicaId. The result is printed to stdout.

Observer resolver

The NotifyingURIResolver class provides means for logging what users access through the viewerproxy. It never processes any URLs itself, merely allows a URIObserver to monitor the URLs. It is currently used to record URLs that are not handled by other resolvers.