

# **Melvyl Log File Design Specifications**

Michael D. Cooper  
School of Information Management and Systems  
University of California  
Berkeley, CA 94720

November 6, 2002

## **Introduction**

The Melvyl log file should serve multiple reporting functions: it should be able to be used as input for standard statistical reporting of system usage (e.g. system use by database, by number of sessions, session length) and also a basis for specialized studies of system activity and performance.

In the second case, the log file should capture the sequence of activities that the user performs during a session. For example, the log file should show the first action the user performed, the response from the system to that action, if any, the second action, and so forth until the completion of the session.

### **About this Specification**

This document is not exhaustive. Its intent is to be indicative of the characteristics of the log file. It does not enumerate all possible items that should be logged. It indicates general categories and leaves it to the person implementing it to realize that something relevant to the general logging concept has been omitted.

### **Physical File Naming and Management**

One physical log file should be created for each day in the year. The logging for the August 24, 2002 file would start at midnight 00:00:00 on the morning of August 24, 2002 and end at 23:59:59 on August 24, 2002. The log file for that date would be given the name

20020824

and then gzipped to become

20020824.gz

If a session begins on day one and ends on day two, the ideal strategy would be for the log records for that session to be written to the log file for day one even though the session ends on day two. This may take more work to implement than it is worth.

### **Sessions, Searches, and Log File Records**

A single physical log file stores all the transactions that took place in the Melvyl system on one day. Within the physical file are a series of log file records, the description of which takes up most of this document. A set of log file records describes one user session with the Melvyl system.

Within a session, the user may conduct many different actions, such as do some searches, display retrieved sets, browse indexes, mail citations, or access another database.

The following example illustrates a hypothetical user session. The details of each line in the example will be discussed later. The example is intended to give an idea of the structure of one session's log records. It is not necessarily complete.

```
2002/08/29|13:45:19|AAB2398456|session=start
2002/08/29|13:45:19|AAB2398456|ip=128.32.226.115
2002/08/29|13:45:19|AAB2398456|browser=Mozilla, v. 3.45
2002/08/29|13:45:22|AAB2398456|campus=ucb
2002/08/29|13:45:23|AAB2398456|db=cat
2002/08/29|13:46:01|AAB2398456|cmdid=1
2002/08/29|13:46:02|AAB2398456|uquery1=XS (chocolate) AND TW (chip# %4 field#)
2002/08/29|13:46:04|AAB2398456|squery1=XS (chocolate) AND TW (chip# %4 field#)
2002/08/29|13:46:29|AAB2398456|hits=24
2002/08/29|13:46:31|AAB2398456|cmdid=2
2002/08/29|13:46:33|AAB2398456|display=uquery1
2002/08/29|13:46:34|AAB2398456|dstart=21;end=30
2002/08/29|13:46:34|AAB2398456|dformat=lon;abs
2002/08/29|13:48:19|AAB2398456|display=end
2002/08/29|13:48:20|AAB2398456|cmdid=3
2002/08/29|13:48:22|AAB2398456|display=uquery1
2002/08/29|13:48:23|AAB2398456|dstart=20
2002/08/29|13:48:26|AAB2398456|dformat=MARC
2002/08/29|13:49:42|AAB2398456|display=end
2002/08/29|13:49:44|AAB2398456|cmdid=4
2002/08/29|13:50:02|AAB2398456|savelist1=uquery1;10-12
2002/08/29|13:51:54|AAB2398456|cmdid=5
2002/08/29|13:51:58|AAB2398456|email=savelist1
2002/08/29|13:52:21|AAB2398456|session=quit
```

In the example above, there is one user search, followed by two user display actions, followed by an action to save two citations to a list, followed by an action to email the citations.

Following this set of 24 log file records would be another set of log file records for another session.

## Record Layout

Each record in the log file has the following fields:

1. Date stamp (YYYY/MM/DD). Example: 2002/03/30.
2. Time stamp (HH:MM:SS). Example: 21:59:03.

Further resolution of the time stamp may be necessary for some purposes, like computer performance evaluation, but generally this level is sufficient.

3. Unique session ID number. The session number should never duplicate, ever. The ID number could be constructed with a few letters at the beginning followed by numbers to extend the range of unique ID's available: AAXR34985216.
4. Data. The data fields are defined in the remainder of this document. Generally data fields will have the format of

variable=value

pairs.

## Data Fields

Data fields are the fourth part of the log record. The data fields are described below. First the field name is given, then a description and/or a motivation for including the field is provided, finally an example of the variable=value pair is provided to indicate what would be written to the log file.

One complete log record has the following structure:

```
2002/08/29|13:45:19|AAB2398456|session=start
```

The date field is followed by a vertical bar, followed by the time stamp, followed by a vertical bar, followed by the unique session id, followed by a vertical bar, followed by the variable=value pair of indicating this is a start-of-session log record. After the last 't' of the word 'start' is a new-line character (backslash n). The vertical bar serves as a field separator.

In the description of some fields, alternatives are given. They are intended to be indicative, NOT exhaustive.

### Session initiation record

The log file has a record in it indicating that the session has been initiated. This record is important for several reasons: It is a physical marker indicating the first log record of the current session. If the 'session termination' marker is missing, this marker serves as a fallback to indicate the end of the previous session. The marker provides valuable information in its time stamp to measure session length.

```
session=start  
session=startscript [an automatic session is run]
```

### Session termination record

The session termination record is the last one for the current session. It is a physical marker of the end of the session in the log file. If the user does not 'log off' or 'quit', writing this record is problematical. If an automatic time-out takes place after, say, 20 minutes of inactivity, the record could then be written. If the record is never written, the session initiation record becomes even more important.

```
session=quit [user logs out]  
session=timeout(20) [timeout period is 20 minutes]
```

Detecting the end of a session is difficult and if there is no end-of-session record in the log file it makes post-processing of the log quite difficult. Can we come up with some relatively foolproof means to insure that an end-of-session indicator is really written?

### **IP Address of the user**

There are privacy concerns about keeping the IP address in the log file. But it provides a very important link between this file of log records and other log files, like Apache. There may be future needs to link the files. With this value and the unique session ID, one can be assured of making the link.

ip=128.32.226.115

### **Campus or other location information**

It is usually possible to infer the Campus from which the user is accessing the system from the IP address. But there may be cases where this is not true. Users with remote access to the system who do not go through a campus portal will probably have to identify themselves. When they do, the log file should record their campus affiliation to facilitate statistical analysis broken down by campus. A good choice for codes for academic institutions would be the National Union Catalog (NUC) symbol.

campus=ucb  
campus=ucop

### **Browser name**

The Apache log provides the browser name. It would be useful to transfer this field to the Melvyl log file to have a unified picture of the type of software (and hardware) being used.

browser=Mozilla, v. 3.45

### **Apache process ID for this session**

This id is used to correlate Apache logs with this log.

apacheid=3497245

## **Interaction Method**

The normal way in which the Melvyl system will be accessed is via a browser running on the users own computer. Two other methods of interaction are possible.

```
access=browser
access=telnet [telnet interface]
access=ccl [common command language]
```

## **Profile or Sign-in Activities**

The user has the option of establishing a profile, modifying an existing profile, and/or activating an existing profile. The log file should record which, if any, of these actions takes place.

```
profile=activated
profile=new
profile=edit
profcampus=ucd
profemail=[hashed]
profid=[hashed]
profpw=[hashed]
```

## **Screen Identification**

One of the most useful pieces of information in analyzing user interaction is to know what the user was viewing on the screen when a particular event took place. The log should identify the name of the screen presented to the user. It is recognized that a named screen will not always have the same information on it over time, but this is too complex a problem to address.

Certain databases may present the results of a search in the form of a screen of data, like a statistical database.

```
screen=<unique screen name>
```

## **Command Sequence Number**

Generally, the user initiates user interaction with the system. Each cycle of interaction is given a unique command interaction number beginning with the number one. This number is recorded in the log. Thus if the user begins with a query, the log file would have an entry indicating the beginning of command number one:

```
cmdid=1
```

followed by the query the user entered. If the user next made a request to display a record, the display request would be prefaced by the entry:

```
cmdid=2
```

followed by the display command the user entered. The motivation for this command sequence number is to facilitate knowing the number of user actions taken during a session, and to delineate the end of one interaction and the beginning of the next. While the uquery command has a sequence number associated with it (e.g. uquery4), not all commands do (for example, 'display'), and there is no logical reason they should.

## **Database**

In legacy Melvyl, the user could select one of many databases for use. In the current system, for the moment there is only one database available. Filtering of that one database for searching takes place through limits in the browse command. In order to preserve the possibility that more than one database might eventually be present in the system, the log file records a session as using one database through the

```
db=cat
```

entry. This default entry will be present for all sessions. Note that when the user employs a browse command, the log entries for that action show the limitations placed on the browsing activity, such as 'music only'.

## **User query**

The log should record the user query to the system as a parenthesized boolean expression in the form the user typed it. If the user typed it in upper case only, the log record would show the exact upper case form. In some cases the current query may be the combination of a previous query and new terms. The log should reflect this. The query should show the indexes specified for each term. The variable name 'uquery' is suffixed with a search statement number.

uquery4=XS (chocolate) AND TW (chip# %4 field#)  
uquery2=XS (chocolate) AND TW (field# !3 chip)

It is possible for a user to re-execute a previous search in the session, combine a previous search with a new search, or combine a previous search with another previously executed search. In this case one needs to know one of two things: either the number of the previous search statement or the full query itself. In either case, the log file should record the active search statement. This may call for making an association between each search and a search number and recording that number and the query in the log file. Another strategy is to simply repeat the previous query when it is reused.

uquery3=uquery2 and AT (UCB)

### **System query**

In the legacy Melvyl system, the system performed a number of transformations of the user query before it was executed. These included normalization, changing of word order, and substitution of spacing. When one examines the log file and the user query it is quite difficult to discern exactly what query the system executed. This specification proposes that the log file record the query executed as well as the query entered by the user.

Here are some examples of ways in which transformations might have occurred to motivate the need for storing a system query.

Example 1:

uquery8=MEDPA (baum mk)  
squery8=MEDPA (baum, m. k.) [alternative 1]  
squery8=MEDPA (baum) OR MEDPA (mk) [alternative 2]  
squery8=MEDPA (baum) AND MEDPA (mk) [alternative 3]

Example 2:

uquery43=KW (pig or rat or tick)  
squery43=KW (pig) or KW (rat) or KW (tick)

Example 3:

uquery5=KW (residential life and university)  
squery5=KW (residential life) AND KW (university)

## Qualifiers

Some databases support the use of qualifiers in a search, like the 'location' qualifier in the catalog database. But there are other examples in other databases like Concept Codes, Supertaxa, and MESH heading identifiers. These need to be included in the full statement of the user query.

## Display commands

The log shows each display request. Among the display variables to be recorded are:

(1) The retrieved set (uquery or uquery number) from which the display is taking place.

display=uquery1

(2) The record number or range of record numbers currently being displayed. If the user requests records 21-30, the log would indicate this. If the user requested record 52, the log would show this.

dstart=21;end=30

dstart=52

A display action can have many formatting options. In the dstart example above, and the dformat example below, a semicolon was used as a delimiter between parameters. I am not sure that this is the best choice.

(3) The formatting used in the display (short, long, review, MARC, UCD-CIRC). If custom formatting is possible, this information can become quite complex to record.

dformat=MARC

dformat=lon;abs

dformat=circstatus

(4) The time at which the display action ends. Display actions are the dominant activity in a session. Even though it is intuitive when the user starts and stops a display event, it is very useful to have a log record indicating the end of the display activity and thus the start of another activity.

display=end

With the 'display=uquery1' and 'display=end' records, it becomes relatively easy to determine the amount of time the user spent viewing results.

## **System response**

The system responds to most commands. The log records this response. Typical responses include the number of hits for a search command.

hits=24

## **Error messages**

If the system issues an error message, the log file records it, along with any user response.

error=Missing search term

error=Database not currently available

error=Connection lost to database. Please try again.

error=Melvyl has suffered an internal error. Please try your search later

## **Help messages**

If the user requests help by clicking on a help button, the log file records the screen that is displayed when the action is requested.

help=<Help Screen Unique Name or Unique Text>

## **User Navigation**

The user has the option to move forward and backward between screens, between record displays, and otherwise navigate the system. The log files records these navigation actions. If there is any inference made by the system that the user has clicked on the 'back' or 'forward' button on the browser, that would be useful information to record in the log.

uaction=back

uaction=forward

There is probably no need to record back and forward actions related to display commands since the log would show which citations were being displayed through the 'dstart' variable.

## **Save Citations and Save Across Sessions Citations**

When a user saves a citation or group of citations, the log needs to record this action along with an indicator of the query from which the citation was retrieved. This indicator could be the query number rather than the query itself. The log should indicate which citation or citations (by citation number in the query) are being saved.

```
savelist1=uquery4;34-39 [save citations 34 through 39 from query 4]  
savelist2=stquery5;6 [save citation 6 from stored query 5]
```

If a search is recalled from a previous session, the log should show this search as part of the search history list, with a tag to identify it was recalled.

```
stquery4=TW (duck) and XS (down--history) [stquery=Stored Query]
```

## **Search History Actions**

As a user continues to perform searches, a list of previous searches is accumulated. If the user deletes one of the previous searches, the log should show the number or full text of the search statement that is deleted.

```
delquery=4 [user deletes query number 4]
```

If a research project involves reconstructing query sequences, that project can reconstruct the active query list after the delete query option has been exercised. I do not think it is worth system time to restate the query history in the log file after a delete operation has taken place.

## **Browse Actions**

The system allows the user to browse many indexes. When the user selects this action the log should record the name of the index selected.

bquery=PA (Muir, John)

bquery=PA [TOPOFLIST] [browse PA index from the top of the list]

bquery=PA [BOTOFLIST]

bquery=XS [NEXT] [move forward one page in the list]

bquery=XS [PREV] [move backward one page in the list]

Limits can be placed on which index entries are displayed to the user.

blimit=sound [limit browsing to sound recordings]

If the user selects an index entry, the log should record the index term selected. If the user displays a record as a result of selecting it from a browse action, the display action would be recorded as it would when the user displays records resulting from a search query.

In the example below, the log shows the user clicking on the link for H.W. Tilman in the personal author index listing. This ultimately leads to a display of materials by Tilman. When the display takes place, 'display' records would be entered in the log file.

bselect=PA (Tilman, H.W.)

When index entries are displayed, a specific number are placed on the screen at one time. Thus there is a start and stop point for the number of records on-screen. The log could record this the same way it records which records are shown with a 'display' command.

## **EMail, Print, and Download Actions**

If the user decides to email, print, or download citations to him/herself, certain information must be supplied. For the email option, this includes the format of the citations, the email address, and any subject line. The log should record that the email action took place, which citations were sent in what format. It should exclude from the log the email address and subject line out of privacy concerns.

I am not sure of the way in which saved citations are managed so it is difficult to specify a log entry for these actions.

```
email=savelist1  
print=yes  
printfORMAT=brief;circ  
download=yes  
downformat=full;holdings
```

### **Portal Activities**

When the user requests access to a database that is not part of the core Melvyl system, the log file should record the URL or other location indicator to which the user is going. If the user returns, and if the system is able to detect the return from the other location, a log record should be generated to indicate the user has returned.

```
portal=[URL of destination site]  
portal=return [user has returned from another site]
```

If the system is passing information to the portal, such as the database name, an index to consult, a query to use, or a citation to consult, this data should be posted to the log.

### **Errors in Managing the Log File**

In the legacy Melvyl log file, records extended beyond a reasonable length (say 2000 characters). Generally this was because a new-line character was not being written. It would be nice if there were some safeguard in the write routine to check for the integrity of each record written. If the write routine detected an error, it could add a message to the log file with an error code. This would simplify post-processing. Other errors may be encountered in writing log records. It would be nice if the program produced error messages when it detected something 'bad', rather than the parsing program trying to figure out a problem after the fact.

```
logerror=badwrite  
logerror=corruptlog  
logerror=badopen  
logerror=linetoolong
```