Acknowledgments

Control Room Logbook (**CRL**) was developed at Fermilab by Gary Roediger and Pat Pomatto. Joy Kyriakopulos, Suzanne Panacek, Philippe Canal and Mark Leininger have participated in the development and support efforts as well.

In July of 2000, physicists and developers from several different experiments and projects were invited to provide guidance to the developers regarding desired features and functionality, and the user interface. The participants included:

D0	Stu Fuess and Geoff Savage
CDF	Jim Patrick, Steve Vejcik and G.P. Yeh
CMS	Vivian O'Dell
BTeV	Erik Gottschalk, Dave Slimmer and Margaret Votava
NUMI/Minos	Liz Buckley-Geer
ROOT	Philippe Canal and Suzanne Panacek

CRL took its shape largely in response to their suggestions. The developers continue to add and change features that the user experiments identify as important. Currently (November 2002), the user experiments include D0, NUMI/Minos, BTeV, and MiniBOONE (all at Fermilab).

The author of this manual wishes to thank Gary, Pat, Joy, Mark and Suzanne for the time they have spent providing and reviewing information. Thanks also to Jae Yu and Andriy Zatserklyaniy of D0.

Table of Contents

Acknowledgments

About this Manual

Purpose and Intended Audiences
Availability
Notational Conventions
Your Comments are Welcome!

Overview and User's Guide

Chapter 1: Overview

1.1 About Control Room Logbook (CRL)
1.2 The CRL Application Window
1.2.1 The Look and Feel
1.2.2 The CRL Toolbar
1.2.3 The CRL Desktop
1.3 Logbook Entries
1.3.1 Entry Format
1.3.2 Entry Features

Chapter 2: Invoking and Logging Into CRL

2.1 Invoking the CRL Application
2.1.1 Windows
2.1.2 Linux
2.2 About Logging In
2.3 Username and Password Configuration
2.3.1 About the Entry Signers Feature
2.3.2 Logging In
2.3.3 Logging Out
2.4 Password and PIN Configuration
2.4.1 About PINs
2.4.2 Logging In
2.4.3 Logging Out
2.5 Changing your Password

Chapter 3: Making Entries in the Logbook

3.1 Selecting Category/Topic and Opening a Container
3.2 Selecting Entry Type and Creating Entry

3.3 Logbook Data Entry by Type
3.3.1 Text
3.3.2 Forms
3.3.3 Online Images
3.3.4 Output Files from External Applications
3.3.5 Freehand Equations, Drawings and Notes
3.3.6 Output from a Program or Command
3.4 Including Current Date and Time in Entry
3.5 Attaching Global Message to All Entries
3.5.1 Turn On a Message
3.5.2 Change or Turn Off a Message
Chapter 4: Editing Logbook Entries
4.1 Input Container Menu Options
4.2 Selecting Logbook Data
4.2.1 Select/Deselect Single or Multiple Entries
4.2.2 Select Text within an Entry
4.3 Modifying Non-archived Logbook Entries
4.3.1 Add to or Change Text in an Entry
4.3.2 Cut, Copy and Paste Text
4.3.3 Undo and Redo Operations
4.3.4 Change Text Font
4.3.5 Delete Entries
4.4 Attaching Keywords to a Logbook Entry
4.4.1 Keywords Tailored to Entry's Topic and Data Type
4.4.2 Global Keywords
4.5 Recovering or Deleting Checkpointed Entries
4.6 Printing Entries
4.7 Sending Entries via Email
Chapter 5: Archiving Logbook Entries to the CRL Database
5.1 About Archiving Logbook Entries
5.1.1 General Information for All Entry Types
5.1.2 Extra Actions Taken for Form Entries
5.2 Archiving Entries
5.3 Removing Archived Logbook Entries from View
5.4 Entries Left Unarchived
Chapter 6: Automated Logbook Entry and Archiving
6.1 Overview of the Autoschedule Feature
6.2 Setting up an Autoscheduled Job
6.2.1 Open a Container and Select Entry Source
6.2.2 Provide Autoscheduling Parameters
6.2.3 Change Scheduling Parameters or Stop Job
Chapter 7: Threading Logbook Entries
7.1 About Threads
7.1.1 The Thread Explorer

7.1.2 Thread Containers
7.2 Creating a Thread
7.2.1 From the Thread Explorer
7.2.2 From an Entry Input Container
7.2.3 From a Thread Container
7.3 Adding Entries to a Thread
7.3.1 From an Entry Input Container
7.3.2 From a Thread Container
7.4 Viewing and Manipulating a Thread
7.4.1 View and Sort Entries in a Thread
7.4.2 Print Entries in a Thread
7.4.3 Send Entries to an Email Recipient
7.4.4 Add Entries from One Thread to Another
7.4.5 Delete Entries from a Thread
7.5 Deleting a Thread

Chapter 8: Working with Archived Entries

8.1 Accessing Archived Entries from within CRL
8.2 Inquiries
8.2.1 Create an Inquiry
8.2.2 Execute an Inquiry and View Results
8.2.3 Halt Execution of an Inquiry
8.2.4 Save an Inquiry
8.2.5 Restore an Inquiry
8.3 LogEntry Explorer
8.4 Accessing Archived Entries on the Web
8.4.1 Define your own Search Criteria
8.4.2 Run a One-Click Search
8.5 Annotating Archived Entries
8.5.1 Annotate Within CRL Application
8.5.2 Annotate from the Web Interface
8.6 Downloading/Viewing Attached Files
8.7 Creating, Editing and Printing Reports in CRL
8.7.1 Pull Up a Report Container
8.7.2 Report Container Menu Options
8.7.3 Insert Entries into Report Container
8.7.4 Export Binary Files from Report Container
8.7.5 Select Report Entries in Container
8.7.6 Remove Entries from Report Container
8.7.7 Send Entries to an Email Recipient
8.7.8 Print Report
8.7.9 Add Entries to a Thread

Chapter 9: Programmer's Guide to the Process Logger

9.1 Introduction
9.2 Guidelines for Programmers
9.2.1 Entry Message Format

9.2.2 Return Messages from Plog
9.2.3 Sample Java Program Excerpt

CRL Administrator's Guide

Chapter 10: CRL Installation
10.1 Space Requirements
10.2 Software Requirements
10.2.1 Requirements for Installing **CRL**
10.2.2 Requirements for Running **CRL**
10.2.3 Requirements for Running **CRL** Web Access
10.3 CRL Installation Components
10.4 Upgrade Instructions (to V1_7_04 or Higher)
10.5 Downloading InstallShield on Windows
10.6 Installing CRL via InstallShield on Windows
10.7 Installing CRL on Linux
10.7.1 Install from Tar File
10.7.2 Install using UPD
10.8 Preparing to Use CRL in Fermilab's AFS Products Area
10.8.1 Instructions
10.8.2 Note for Linux
10.8.3 Notes for Windows
10.8.4 Linux and Windows Combined Setup
10.9 Uninstalling CRL
10.9.1 Uninstall on Windows
10.9.2 Uninstall on Linux

Chapter 11: Configuring and Launching CRL
11.1 Editing the Properties File
11.1.1 The Bare Minimum
11.1.2 Further Edits
11.2 Configuring CRL to Use a Preinstalled JVM (Optional)
11.2.1 Files to Copy
11.2.2 Sample CRL Invocation Script
11.2.3 Modify the CRL Invocation Script
11.3 Launching the CRL Application
11.3.1 Once you're ready for launch...
11.3.2 Windows (Local and AFS Installations)
11.3.3 Linux (Local UPD Installation)
11.3.4 Linux (Local Tar File Installation)
11.3.5 Linux (AFS Installation)
11.4 Configuring Your Desktop
11.4.1 Define Keywords
11.4.2 Define a new Desktop Page (Data Entry or Report)
11.4.3 Define a Menu and/or Submenu on a Desktop Page
11.4.4 Define a ToolBar with ToolButtons for Data Entry Types
11.5 Creating Configuration Files for Forms

11.5.1 Create/Modify the XML Form Definition Files
11.5.2 Enable Automatic Electronic Mailing of Form Entries
11.5.3 Include Tables in a Form
11.5.4 Create Forms with Selected Reloadable Fields
11.6 Enabling Document View plus Thumbnail for PS and PDF File Entries
11.7 Editing the Inquiries Configuration File
11.8 Configuring Print Queues
11.9 Starting the Process Logger Daemon

Chapter 12: Managing User Information

12.1 The Administrator Button
12.2 Adding Users
12.3 Activating/Deactivating Users
12.4 Editing Information on Active Users
12.5 Exiting the Administrative Area

Chapter 13: Configuring Web Access to Logged CRL Entries

13.1 Editing the Properties File
13.2 Installing and Configuring Additional Software
13.3 Obtaining and Editing the JavaServer Page
13.3.1 Configure Keywords Table
13.3.2 Finding Information on JSP
13.4 Configure "One-Click" Searches

CRL Administrator’s Reference Manual

Chapter 14: CRL’s Java Properties

Chapter 15: Introduction to XML and DTD Files

15.1 XML Compared to HTML
15.2 Let’s Get a Feel For It...
15.3 Element Types and Attributes in the DTD File
15.3.1 Element Type Declarations
15.3.2 Attributes and Attribute Lists
15.3.3 Example of Element Type Declaration plus Attribute List

Chapter 16: The CRL Desktop Configuration File

16.1 CRL Configuration DTD File Listing
16.2 Element Type and Attribute Listing
16.2.1 XML Declaration
16.2.2 LogBookConfig
16.2.3 Keyword
16.2.4 InitialTabs
16.2.5 Page
16.2.6 EntryInputPage
16.2.7 Menu and SubMenu
16.2.8 Topic
16.2.9 ToolBar and ToolButton
16.3 The CRL Java Class Libraries

16.3.1 Container (Topic) Classes
16.3.2 Entry type Classes
16.3.3 Desktop Page Classes

Chapter 17: Form and Inquiry Configuration Files

17.1 Form Definition Files
17.1.1 XML Elements Allowed in Forms
17.1.2 Sample "Start of Shift" Form
17.1.3 Sample "End of Shift" Form
17.1.4 Sample Form using a Variety of Form Elements
17.1.5 Sample Table Element in a Form
17.1.6 Sample Line for Running a Script or Command
17.1.7 Sample Form with Embedded Forms
17.2 The Logbook Inquiry Configuration File
17.2.1 DTD File Listing for Inquiries
17.2.2 Default XML File Listing for Inquiries

Index

About this Manual

This chapter provides an introduction to the *Control Room Logbook User's and Administrator's Guide*. In particular you will find:

- the purpose and intended audience
- where to find the manual on-line or obtain a hardcopy
- the typeface conventions and symbols used throughout the manual
- where to send comments and questions

1. Purpose and Intended Audiences

This guide explains and illustrates the setup and use of the **Control Room Logbook** (**CRL**) application, a computer-based logbook intended for use in high energy physics experiment control rooms.

The guide consists of three parts:

Part I: Overview and User's Guide

This part is intended primarily to instruct end users on using the features and functionality of the **CRL** application. The end users are presumed to be experimenters on shift in their experiment's control room.

Secondly, the web interface (optional) portion of **CRL** is described here for experimenters wishing to access logbook entries from anywhere in the world via the internet.

Thirdly, experimenters wishing to configure their external programs to create **CRL** entries will find guidance on using the Process Logger.

Part II: CRL Administrator's Guide

This part guides administrators through installing and configuring the application and the web interface.

Part III: CRL Administrator's Reference Manual

This part illustrates the content and structure of the properties file and the configuration files that the administrator needs to edit.

2. Availability

Copies of the *Control Room LogBook User's Guide* (document number PU0376) can be obtained from the following sources:

On-line	`http://www.fnal.gov/docs/products/crl/`
	Under **Documentation Search** on the Computing Division home page (`http://www.fnal.gov/cd/`), search using any of the following keywords: control, room, logbook, crl, elog, e-log, log

Paper Copies	Each chapter is provided individually in PDF format, and accessible via the *View/print PDF file* link on its web page. The entire manual is provided in PostScript and PDF formats under `http://www.fnal.gov/docs/products/crl/manual/ps/`

3. Notational Conventions

The following notational conventions are used in this document:

bold	Used for product and program names (e.g., **CRL**).
italic	Used to emphasize a word or concept in the text. Also used to indicate logon ids and node names.
`typewriter`	Used for filenames, pathnames, contents of files, output of commands.
`typewriter-bold`	Used to indicate commands and prompts.
<CTRL-CHAR>	Indicates a control character. To enter a control character, hold down the control key (labeled **CTRL**, usually) while pressing the key specified by **CHAR**.
`[ ]`	In command formats, indicates optional command arguments and options.
`%`	Prompt for UNIX C shell family commands (**`%`** is also used throughout this document when a command works for both shell families).
`$`	Prompt for UNIX Bourne shell family commands.
`< >`	In commands, paths and environment variables, indicates strings for which the user must make context-specific substitutions.

All command examples are followed by an implicit carriage return key. The following symbols are used throughout the text to draw your attention to specific items:

A “bomb”; this is used to indicate a potential pitfall.

☞ This symbol is intended to draw your attention to a particularly important piece of information.

4. Your Comments are Welcome!

For **CRL** product support issues, problems, and other application-related comments, use the *crl-dev@fnal.gov* mailing list. Archives are maintained at `http://listserv.fnal.gov/archives/crl-support.html`.

The *Control Room Logbook User's and Administrator's Guide* may contain some errors, however we endeavor to minimize the error count! We encourage all the readers of this document to report back to us:

- errors or inconsistencies that we have overlooked
- any parts of the manual that are confusing or unhelpful -- please offer constructive suggestions!
- other topics to include (keeping in mind the purpose of the manual)
- information that other users might find helpful

Send your documentation-related comments to *cdlibrary@fnal.gov.*

Part I Overview and User’s Guide

Chapter 1: *Overview*

We provide an overview of the user interface, describing the “look and feel”, and identifying the types of objects found in the **CRL** window. We also describe the format and features of logbook entries.

Chapter 2: *Invoking and Logging Into CRL*

In this chapter we describe first how to launch the application, and then we describe the two methods for logging in and out of **CRL**. We discuss the Entry Signer feature and session PINs, and we show how to change your password.

Chapter 3: *Making Entries in the Logbook*

In this chapter we describe how to enter logbook data into **CRL** input containers.

Chapter 4: *Editing Logbook Entries*

Before a logbook entry is archived to the **CRL** database, its content and/or appearance can be modified, or it can be deleted. In this chapter we discuss the menu options available on the entry input containers, and we describe how to select and edit your logbook entries.

Chapter 5: *Archiving Logbook Entries to the CRL Database*

Once a logbook entry has been archived, the entry gets “set in stone”. In this chapter we describe how to archive your logbook entries.

Chapter 6: *Automated Logbook Entry and Archiving*

In this chapter we describe the process of setting up an autoscheduled logbook entry job.

Chapter 7: *Threading Logbook Entries*

A thread is a vehicle to link a series of entries relating to a particular issue. In this chapter, we discuss creating and manipulating *threads*.

Chapter 8: *Working with Archived Entries*

In this chapter we discuss searching, viewing, annotating and printing archived entries from within the **CRL** application and from the Web.

Chapter 9: *Programmer’s Guide to the Process Logger*

This chapter describes the Process Logger and provides some guidance for programmers wishing to configure their external programs to create **CRL** entries.

Chapter 1: Overview

1.1 About Control Room Logbook (CRL)

Control Room Logbook (**CRL**) is a multimedia, remotely-viewable, computer-based logbook for use in high energy physics experiment control rooms. **CRL** was conceived in early 1999. It was developed according to a set of requirements identified by physicists from several different experiments[1]. **Java** and **XML** were used to create this product in order to take advantage of the portability, ubiquitousness and rich functionality that these languages offer. **CRL** has three parts:

- the **CRL** application, installed in the control room to create, manipulate, and log entries
- the optional Process Logger which allows logging of entries created by external programs
- the optional **CRL** Web Access used to browse and annotate logged entries from virtually anywhere in the world

CRL supports a wide variety of entry data types. It can store the entries' content in both XML and HTML on the local disk drive or in a shared file system. In addition, it uses a relational database to store indexed information to query the entries. **CRL** is potentially compatible with any SQL-based relational database management system (RDBMS); currently it supports MySQL. The Web Access portion of **CRL** uses only the relational database and the HTML entries.

For information on implementing and configuring **CRL**, please see Part II: *CRL Administrator's Guide* and Part III: *CRL Administrator's Reference Manual*.

1. The experiments polled include DZero, CDF, CMS, NUMI, and BTeV.

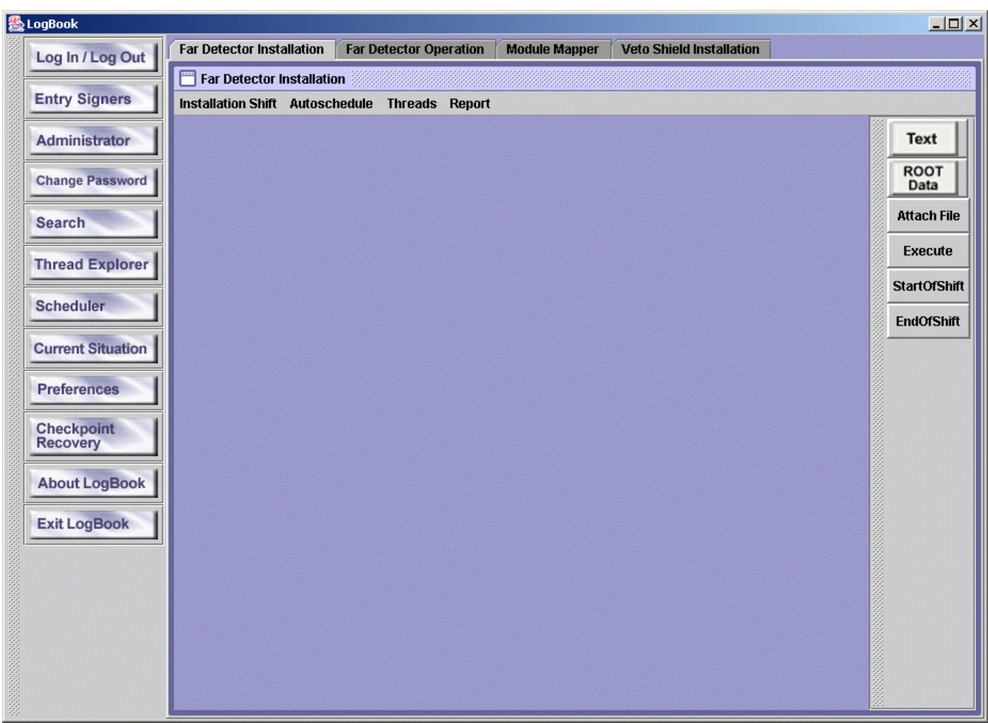
1.2 The CRL Application Window

In this section, we provide an overview of the user interface, describing the "look and feel", and identifying the types of objects found in the **CRL** window. The window in which **CRL** runs is composed of two frames (see image below):

CRL toolbar — The **CRL** toolbar runs vertically down the left side of the window (see image below). This is a column of clickable buttons that control basic operations of **CRL**. It is not configurable, and therefore is the same in all installations.

Desktop — To the right of the **CRL** toolbar, the desktop is the tabbed pane which takes up most of the window (see tabs at top of pane labelled **DETECTOR**, **DAQ**, and **ARCHIVE REPORT PAGE**). Each tab represents a different desktop page. The desktop displays one page at a time, according to the selected tab.

The desktop pages are configured by each experiment's **CRL** administrator(s) according to the needs of the experiment.



1.2.1 The Look and Feel

CRL uses a familiar paradigm for its GUI, and includes features to aid in easy identification of desktop elements and logbook entry status:

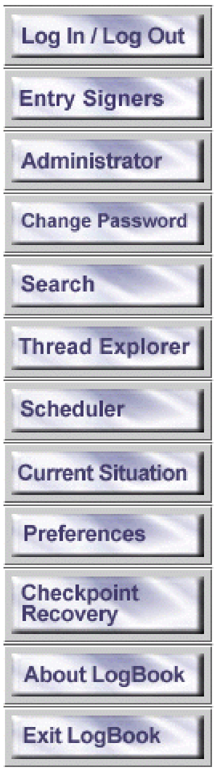
- toolbars, buttons, cascading drop-down menus, and tooltips on mouse rollover
- separate desktop pages with tabbed views
- use of color for finding information or checking status “at-a-glance”
- automatic checkpointing

... and easy entry/manipulation of logbook data:

- drag-and-drop to initiate logbook entry
- double-click, control key and right-click shortcuts
- pop-up windows allowing user to type in or browse for item
- cut/copy/paste functions (within **CRL** and from/to external applications)
- form entry with option to save/reload data
- selection of individual entries or all entries
- “threads” for linking a series of related entries
- automatic, scheduled logbook data entry
- on-line help

1.2.2 The CRL Toolbar

The **CRL** toolbar is a column of clickable buttons that control basic operations of **CRL** independently of the desktop page operations:



LOG IN/LOG OUT Brings up the login window (see section 2.3.2 *Logging In*)

ENTRY SIGNERS Brings up a window in which to add/remove logged in users from the list of entry signers (see section 2.3.1 *About the Entry Signers Feature*)

ADMINISTRATOR Brings up a window for adding/(de)activating/editing user information; administrative password required; (see Chapter 12: *Managing User Information*)

CHANGE PASSWORD Brings up a window allowing users to change their password (see section 2.5 *Changing your Password*)

SEARCH Brings up a window for querying database (see section 8.1 *Accessing Archived Entries from within CRL*)

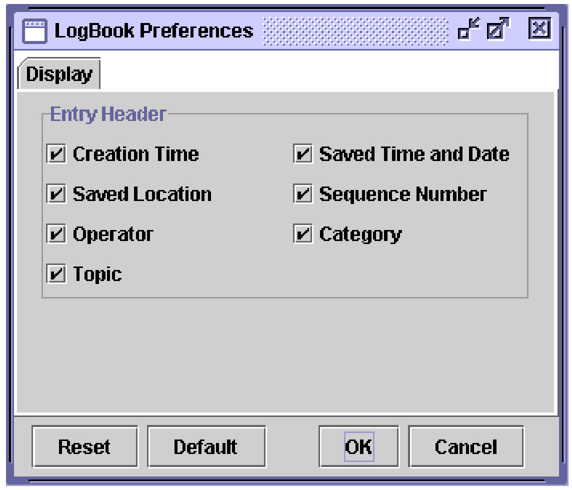
THREAD EXPLORER Brings up a window for managing threads. Threads are used to link related logbook entries (see Chapter 7: *Threading Logbook Entries*).

SCHEDULER Brings up a window indicating all autoscheduled jobs currently running, and allows users to select any or each job for the purpose of changing scheduling parameters or terminating the job (see section 6.2.3 *Change Scheduling Parameters or Stop Job*).

CURRENT SITUATION

Brings up a window for entering, changing or turning off a “current situation” message. A current situation message (also called a “global message”) gets automatically inserted in all subsequent entries until a user turns it off (see section 3.5 *Attaching Global Message to All Entries*).

Preferences Brings up a window for selecting entry header lines to display/hide in container windows.

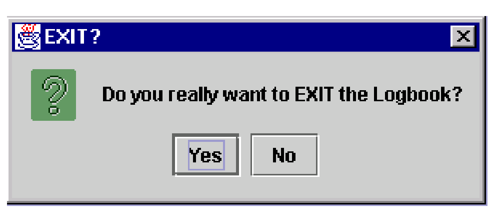


Checkpoint Recovery

Entries not yet archived are checkpointed (saved) periodically to protect against loss. This button brings up a window listing all entries currently checkpointed for the machine (see section 4.5 *Recovering or Deleting Checkpointed Entries*).

About LogBook Brings up a window containing information about the **CRL** application (shown below for the Windows version V1_7_04)

Exit LogBook Exits the **CRL** application (confirmation prompt provided)

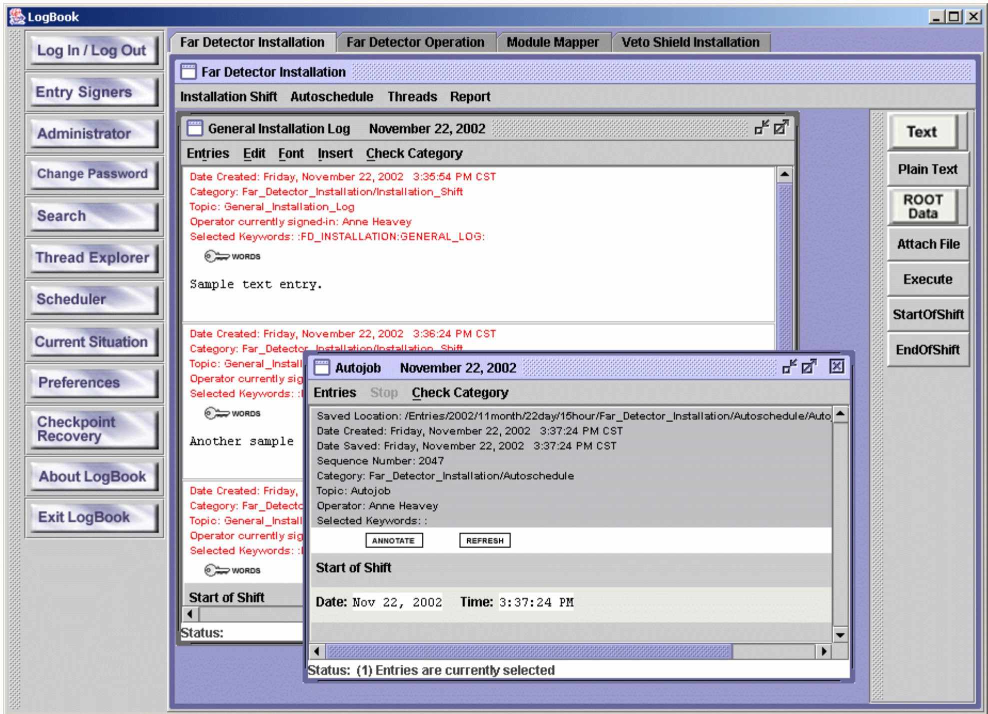


1.2.3 The CRL Desktop

The large window to the right of the **CRL** Toolbar comprises the desktop. The **CRL** desktop is highly customizable, and each experiment configures the desktop pages, the input sources for logbook entries, and the hierarchical structure of categories and topics for logbook data storage according to its needs. The number of items needed on the desktop may therefore become quite large. Complex experiments with lots of monitoring equipment may choose to run **CRL** simultaneously on several computers in the control room, each configured to accept logbook entries from a particular set of sources, and to log the entries into a corresponding set of categories/topics. This allows the desktop on each machine to remain relatively uncluttered.

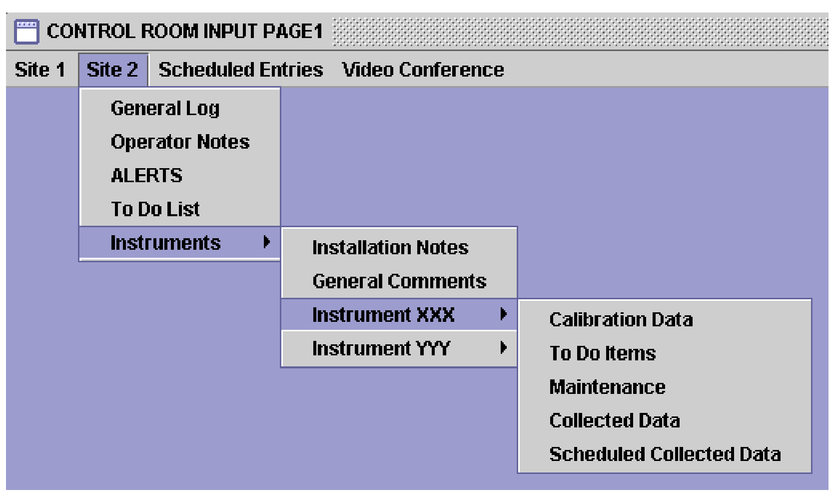
Objects on the Desktop

Page Tabs	Page tabs are displayed horizontally along the top of the desktop. They identify the various desktop pages in your configuration. Click on one to make it the active page on the desktop.
Desktop Page	A desktop page is the work space displayed for the selected tab. There may be several pages to your desktop; pages are configurable by experiment. A desktop page may be configured for data entry and manipulation, or for searching/viewing/manipulating archived entries only. All pages provide one or more menus, and each data entry page also provides a data-entry toolbar running vertically down the right-hand side of the page. Only one page is visible and active at a time. The active desktop page is identified at the top of the page (shown in this image as **TUTORIAL**, a data entry page), underneath the row of page tabs.



Pull-down Menus

Each desktop page has a set of menu headings lined up horizontally underneath the page title. Usually they represent general logbook entry categories. These are pull-down menus which have been configured by your experiment. They may cascade several levels in order to allow precise categorization of entries. The following image shows a desktop page labelled **CONTROL ROOM INPUT PAGE1**, and cascading menus starting from **SITE 2**. The menu options are described under *categories* and *topics* below.



Categories	The menu headings and all the sublevels of categorization except the final one are considered logbook entry *categories*.
Topics	The final level of categorization is considered the *topic* (all menu options without an arrow pointing to the right are topics). Associated with each topic is a *container*, described below.

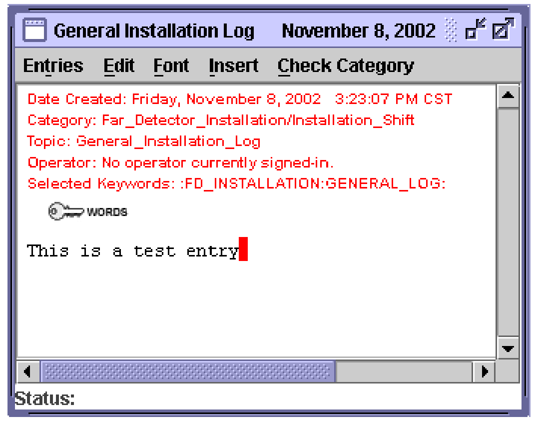
Containers

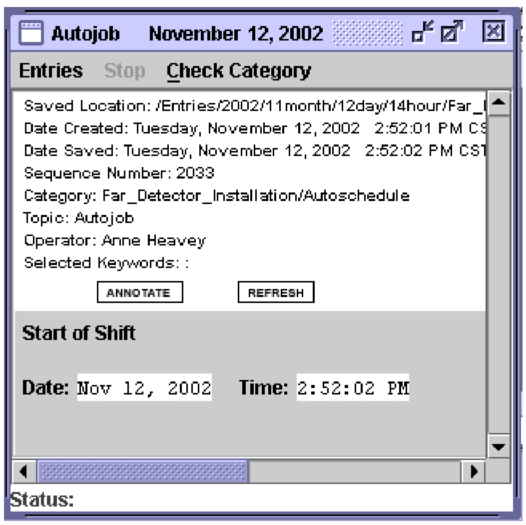
A container is a **CRL** window that can contain one or more logbook entries. Each container represents and is labelled with a particular topic under the selected category hierarchy. You open a container of a given topic by double-clicking the menu item for that topic. Containers can be moved around the desktop, resized, iconized, and closed. They include a status bar at the bottom showing how many entries are selected within the container.

There are several kinds of containers, described below, each with different properties. Each container provides menus for manipulating entries. These menus are not configurable by experiment, and they vary according to the container type.

Input containers

Input containers are containers in which you can add and edit new logbook entries. The input container at right contains entries that get stored under the experiment-configured topic **GENERAL INSTALLATION LOG** (notice the container's title).



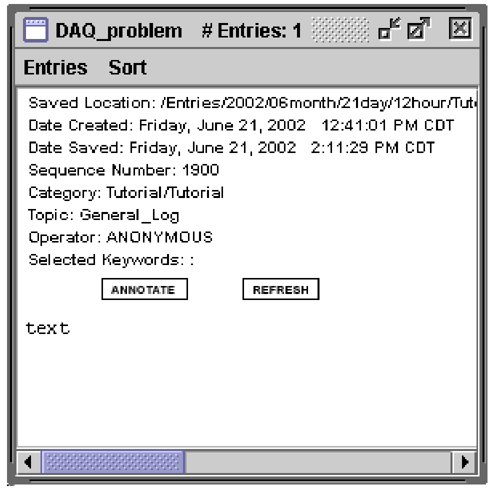


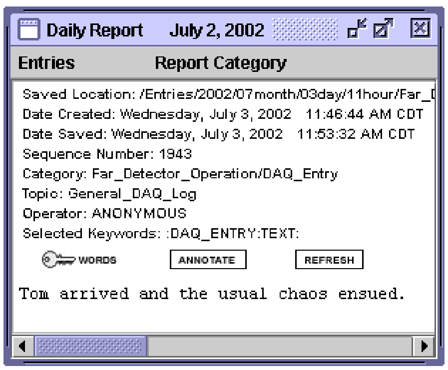
Scheduled Containers

Scheduled containers are input containers for automatically scheduled logbook data entry (see Chapter 6: *Automated Logbook Entry and Archiving*). The container at left has been configured as **AUTOJOB**.

Thread Containers

A *thread* links entries relating to a particular issue (see Chapter 7: *Threading Logbook Entries*). Thread containers are used for collecting threaded archived logbook entries, and are named by the user who creates the thread (e.g., **DAQ_PROBLEM**).



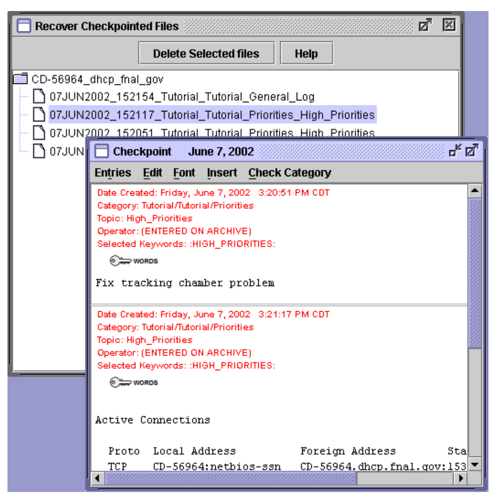


Report Containers

Report containers are used for collecting archived logbook entries to include in reports (see section 8.7 *Creating, Editing and Printing Reports in CRL*). They are configured by the experiment's **CRL** administrators. The one at left is named **DAILY REPORT**.

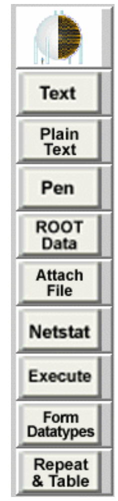
Checkpoint Containers

Checkpoint containers are used for manipulating recovered checkpointed entries. They are configured by the experiment's **CRL** administrators. The one at right is named **CHECKPOINT**. Behind it we show the checkpointed entries.



Logbook Entry Toolbar

On pages that allow logbook data entry, there is a toolbar running vertically down the right-hand side of the page. This toolbar includes a button for each logbook entry type (e.g., text, image binary file, **ROOT** data, forms, etc.), as configured for your installation. This toolbar is scrollable so that it can accommodate many data entry types.



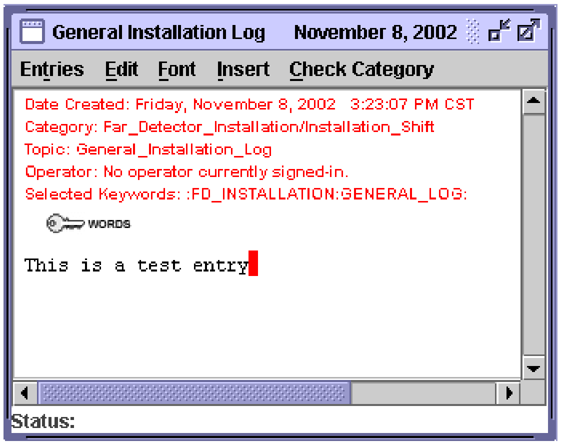
To create an entry of a given type, you either double-click the button to create an entry in the currently selected open input container, or drag the button into any open input container.

1.3 Logbook Entries

1.3.1 Entry Format

Logbook entries get added to input containers or scheduled containers. In either type of input container, each data entry has a header, which identifies the date and time of the entry, the data category and topic, and the operator(s) who are logged in. Depending on the preferences set (see the **PREFERENCES** button under section 1.2.2 *The CRL Toolbar*), some header elements may not be displayed. If there are any keywords that can be attached to the entry (this depends on the configuration), then below the header you'll see a key symbol

(WORDS), which can be clicked to show or set keywords for the entry. The body of the entry is displayed under the key symbol. At the bottom, a status line indicates how many entries are currently selected.



Before a logbook entry is archived to the **CRL** database, its content and/or appearance can be modified, or it can be deleted. At this pre-archived stage, the data header appears in red. Once it is archived, the data entry can be annotated, but no longer modified or deleted, and the data header turns to black.

This color change was designed to let operators tell easily which entries are archived and which are not. In report containers, all headings are black, since all possible entries are already archived.

1.3.2 Entry Features

Data Types — Logbook data can be entered from a wide variety of sources. See section 3.3 *Logbook Data Entry by Type*.

Browse Directories — Several of the data types allow you to browse for a file or a URL to include in an entry. The system remembers the directory most recently browsed, and starts you from there each time. See sections 3.3.3 *Online Images*, 3.3.4 *Output Files from External Applications*.

Global Messages — A global message may be included in all logbook entries as experimental conditions warrant. See 3.5 *Attaching Global Message to All Entries*.

Editing — Before a logbook entry is archived to the **CRL** database, its content and/or appearance can be modified, or it can be deleted. See Chapter 4: *Editing Logbook Entries*.

Hot Key for Date/Time

The key sequence **CTRL-ALT-D** can be used for entering the current date and time into an entry or form field.

Double-click Shortcuts

	Double-click on toolbutton to create entry of corresponding type. Within container, double-click to create entry of type configured by your **CRL** admin.
Control Key and Function Key Shortcuts	
	Short-cut key sequences are defined for several common operations.
Keywords	Keywords can be linked to logbook entries in order to provide an additional dimension for querying the database when attempting to later identify and retrieve particular entries. See section 4.4 *Attaching Keywords to a Logbook Entry*.
Checkpointing	Entries that have not yet been archived or deleted get written out periodically to a checkpoint directory. This protects against significant data loss in the event of a crash or accidental deletion. See section 4.5 *Recovering or Deleting Checkpointed Entries*.
Archiving	Once a logbook entry has been archived (saved), the entry gets "set in stone". See Chapter 5: *Archiving Logbook Entries to the CRL Database*.
Reminders[1]	If an entry is left too long without being archived, your system may send messages periodically to ask you to address it. See section 5.4 *Entries Left Unarchived.*
Threading	A thread links a series of archived entries relating to a particular issue. It provides easy access to the flow of information on an issue, given that the entries may have been entered in different desktop pages, categories and/or topics, and by different operators. See Chapter 7: *Threading Logbook Entries*.
Searching/Retrieving	There are two methods of access to the database of archived entries from within **CRL**: **INQUIRIES** and **LOGENTRY EXPLORER**. See section 8.1 *Accessing Archived Entries from within CRL.* Entries can also be retrieved via the web, see 8.4 *Accessing Archived Entries on the Web*.
Annotating	The only change you can make to an archived entry is to add an annotation. This can be done from within **CRL** (see section 8.5.1 *Annotate Within CRL Application*) and from a web browser (see 8.5.2 *Annotate from the Web Interface*).

1. Also called "nagging".

Chapter 2: Invoking and Logging Into CRL

In this chapter we describe first how to launch the application, and then we describe the two methods for logging in and out of **CRL**. We discuss the Entry Signer feature and session PINs, and we show how to change your password.

2.1 Invoking the CRL Application

Typically **CRL** is up and running at all times in a control room, and users do not have to invoke it; they can just log in (see section 2.2 *About Logging In*). But in case you need to invoke the application, here are instructions:

2.1.1 Windows

On Windows, you will have a desktop icon for **CRL**. Just double-click it to run the program.

2.1.2 Linux

How you invoke **CRL** depends on how it was installed. Find out from your **CRL** administrator what you need to do. Here are typical scenarios:

UPD

Typically, if UPD was used to install **CRL**, you run:

```
% setup crl V<x_y> [-f Linux]
% crl
```

Tar File

For a tar file installation, you may need to add the directory containing the **CRL** script to your $PATH manually, then run the program by entering the script name, e.g.,:

```
% CRL_Linux.bin
```

AFS

If your system runs the **CRL** installation in Fermilab's AFS product area, find out from your sysadmin what command to run. There should be a script on your local machine that runs **CRL** such that it points to local configuration information.

If you are logged into an fnalu Linux node (for example, flxi02.fnal.gov), or any Linux node that has a **UPS** database setup for the AFS products area, it should be as simple as:

```
% setup crl V<x_y> [-f Linux]
% <local_crl_scriptname>
```

If you are on a Linux system with no **UPS** database, you may need to set an environment variable that points to the product, for example:

```
% setenv CRL_DIR /afs/fnal.gov/ups/crl/V<x_y>/Linux
```

and then run the local script that invokes **CRL**:

```
% <local_crl_scriptname>
```

2.2 About Logging In

Your **CRL** administrator has configured your installation according to one of two available login methods[1]:

- The "username and password" login method requires entry of a username and a password only, and uses the Entry Signers feature. Depending on your configuration, you may need to log in to **CRL** in order to archive entries, or you may be able to archive without logging in. Only the names of logged in users that are in the Entry Signers list will appear with the entries as they are archived. See section 2.3.1 *About the Entry Signers Feature*.
- The alternative login method, "password and PIN", requires that you select your username from a list, and provide your password and a PIN (Personal Identification Number). For this method, you cannot archive entries unless you are logged in. You must provide your PIN whenever you give the command to archive an entry. See 2.4.1 *About PINs*.

1. To the administrator: Set the `Logbook.login.class` parameter to the desired value in the `LogbookConfigParms.properties file`; see Chapter 14: *CRL's Java Properties*.

At least a day or two before you will need to log in, you should make sure that you have been entered into the system by your **CRL** administrator and that you get an initial password. Your **CRL** login user name is the “name” part of your email address, e.g., for the email address *joe@fnal.gov*, the **CRL** user name is *joe*.

If the **CRL** program won’t let you log in, a likely cause is that the database is not working. If a **CRL** session is up and running, it can still be used in this condition to create (but not archive) entries; however no one can log in, and no new **CRL** sessions can start.

2.3 Username and Password Configuration

2.3.1 About the Entry Signers Feature

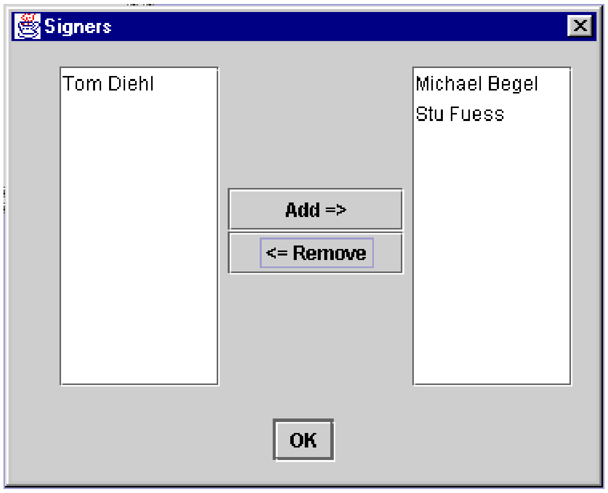
This feature is used only if **CRL** is configured to process “username and password” logins (see section 2.2 *About Logging In*).

The Entry Signer List

When you log in to **CRL**, your name gets added automatically to something called the Entry Signers list. This list is made up of all users that are currently logged in (minus those that have been manually removed from the list). As its name implies, the Entry Signers list provides the name(s) that **CRL** associates with each logbook entry as it is archived; it provides the value for the **OPERATOR** field.

Adding/Removing Users to/from Entry Signer List

While you are logged in, you can remove yourself from the entry signers list, and add yourself back at will. You can also add/remove other logged in users to/from the list.



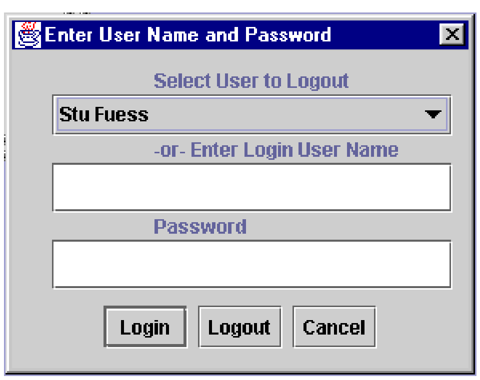
In the right-hand column is the list of entry signers. By default each user that is logged in appears in this column. In the left-hand column is the list of users that have been removed from the list. Together, both columns show all the users currently logged in.

To add entry signer(s):	select from the left-hand list and click **Add=>**
To remove entry signer(s):	select from the right-hand list and click **<=Remove**

To add/remove multiple signers at a time, use the standard method of selecting multiple items for your windowing system. E.g., for Windows, use the **Ctrl** or **Shift** key (**Ctrl**-click individual, nonadjacent items, or **Shift**-click top and bottom of set of adjacent items).

2.3.2 Logging In

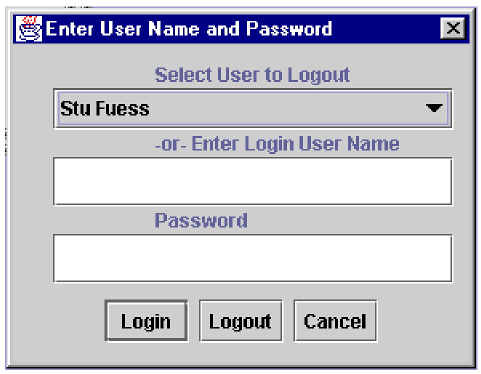
To log in, click the **Log In/Log Out** button on the Logbook Toolbar at the left of the CRL window. This pops up the **Enter User Name and Password** screen.



Type your login user name in **ENTER LOGIN USER NAME**, then enter your password. Click **LOGIN** (or **CANCEL** to cancel out). Once you login, your name appears in the Entry Signers list.

2.3.3 Logging Out

You can log yourself or another user out of **CRL**. To log someone out, click the **LOG IN/LOG OUT** button on the Logbook Toolbar at the left of the **CRL** window. This pops up the **ENTER USER NAME AND PASSWORD** screen.



From the drop-down list, select the user to log out then click **LOGOUT** (or **CANCEL** to cancel out). Once logged out, the user should no longer appear in either column of the **ENTRY SIGNERS** window.

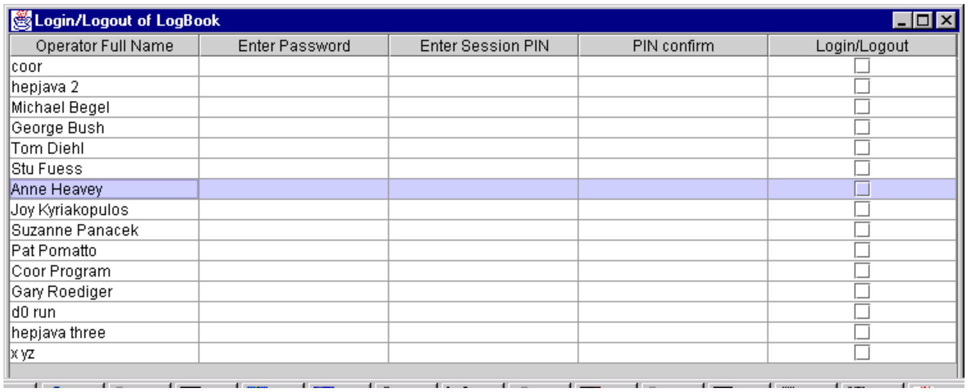
2.4 Password and PIN Configuration

2.4.1 About PINs

PINs are used only for the login configuration that processes "password and PIN" logins (see section 2.2 *About Logging In*). The PIN you choose is good only for your current login session. You can choose a different one each session, or not. You must enter your PIN every time you archive an entry. This determines the username associated with the archived entry in the database; in other words, it fills in the entry's Operator field. To assign multiple names to an entry, all the users whose names are to be assigned must share the same PIN for the session. PINs must be at least 4 characters long.

2.4.2 Logging In

To log in, click the **LOG IN/LOG OUT** button on the Logbook Toolbar at the left of the **CRL** window. This pops up the **LOGIN/LOGOUT OF LOGBOOK** screen:



1) Scroll down to your name. (If it does not appear, contact your **CRL** administrator.)

2) Enter your password (field turns yellow).

3) Choose a session PIN, and enter it (field turns yellow).

4) Enter PIN again, for confirmation (field turns yellow).

5) Click the **LOGIN/LOGOUT** box (a check appears and password and PIN fields turn green to show that you're logged on).

6) Minimize or close the **LOGIN/LOGOUT OF LOGBOOK** window.

2.4.3 Logging Out

1) To log out, click the **LOG IN/LOG OUT** button on the Logbook Toolbar at the left of the **CRL** window. This pops up the **LOGIN/LOGOUT OF LOGBOOK** screen.

2) Scroll down to your name.

3) Enter your session PIN.

4) Enter PIN again, for confirmation.

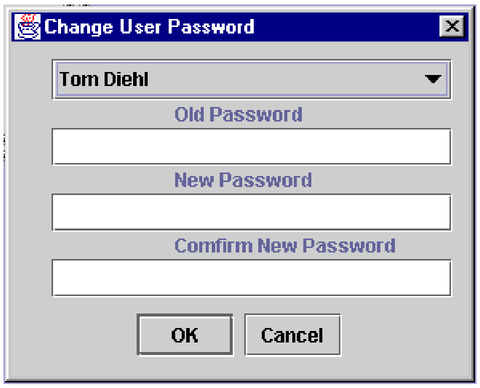
5) Click the **LOGIN/LOGOUT** box (the check disappears and password and PIN fields are no longer green).

2.5 Changing your Password

You should change your initial password at your earliest opportunity. To change your password, you need to know your existing one. If you've forgotten it, contact your **CRL** administrator to get a new one. Passwords are required to contain a minimum of 4 and a maximum of 8 characters. You do not need to be logged on to change your password. Fermilab users: please choose a password that is very different from your Kerberos one!

Select the **CHANGE PASSWORD** button on the **CRL** toolbar at the left of the **CRL** window. This pops up the **CHANGE PASSWORD** screen.

Select your name in the **SELECT USER** drop-down list. Type your old password, then your new one, and finally your new one again for confirmation. If the change is successful, you will get no message. If your old password is wrong, or if there's a problem with your new one, a message will appear, and the change will not take place.



Chapter 3: Making Entries in the Logbook

In this chapter we describe how to enter logbook data into **CRL** input containers.

☞ Besides manual entry, logbook entries can be scheduled and automated. We discuss this topic in Chapter 6: *Automated Logbook Entry and Archiving*. Logbook entries can also be created via the process logger, discussed in Chapter 9: *Programmer's Guide to the Process Logger*.

3.1 Selecting Category/Topic and Opening a Container

Your configuration may have several data input pages, and many menu options and toolbar buttons. You need to understand how your experiment categorizes data, and become familiar with the desktop.

Before you make a logbook entry, you have to have some place to put it. Therefore, the first step is to select an appropriate category and topic for the entry, and open the corresponding input container. To do so:

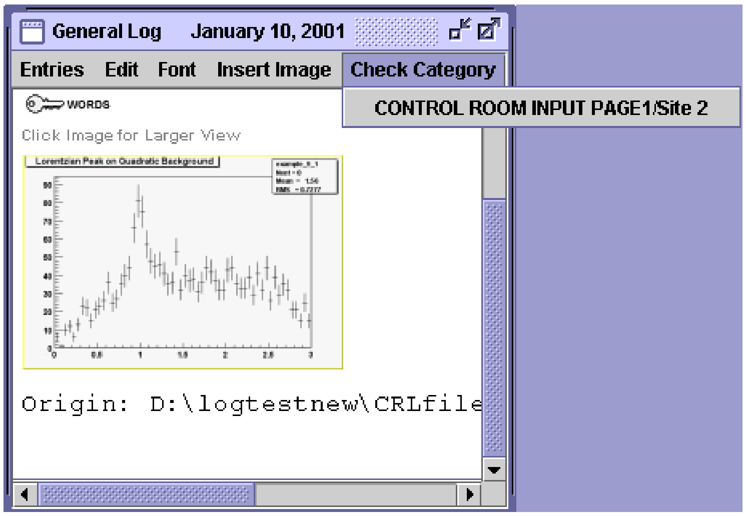
1) Select the page tab at the top of the screen to display the appropriate input page on the desktop. (One or more input container windows may appear automatically on the desktop page.)

2) From the horizontal row of menu headings along the top of the selected page, choose the one that corresponds to the general category of logbook data you plan to enter. Note that there may be menu headings for autoscheduled logbook entry, reports and checkpoints as well as for manual entry.

3) From its pull-down menu (or cascading pull-down menus), choose the category and topic appropriate for your entry. An input container pops up. The container is labelled with the chosen topic. Container windows can be moved around the desktop, resized, iconized, and/or closed.

The “Check Category” Option on the Container

To verify the menu path to the container, click **CHECK CATEGORY**. This displays the path starting with the initial menu option chosen on the active page, and ending one level up from the topic shown on the container.



3.2 Selecting Entry Type and Creating Entry

In the container, you first need to insert your entry:

1) Go to the logbook entry toolbar at the right of the **CRL** desktop. Find the toolbutton corresponding to the data type you want; scroll down if necessary to locate it.

2) Place your cursor in the toolbutton, and either:

a) double-click on the toolbutton[1], or

b) drag-and-drop the selected toolbutton into the container, or...

1. When working over a network, a double-click is sometimes interpreted as two single clicks.

Double-Click Shortcut

There is a shortcut that may be implemented for some or all input containers in your configuration. Any particular input container can be configured to have a default entry type, such that double-clicking *in the container* creates an entry of that type. For example, a container labelled “Operator Comments”, might be configured such that a double-click in the container creates a text entry.

Be aware that when working over a network, a double-click may get interpreted as two single clicks, and thus may not work.

3.3 Logbook Data Entry by Type

Logbook data can be entered from a wide variety of sources:

- plain text from keyboard or file (no formatting available)
- text from keyboard or file (formatting available)
- forms
- online images
- output files from applications (e.g., **ROOT**, MS Word, PDF, PostScript, images, etc.)
- freehand equations, drawings and notes made on Ipen® tablet
- output from a command issued to OS, or from a script or other executable

A few general notes:

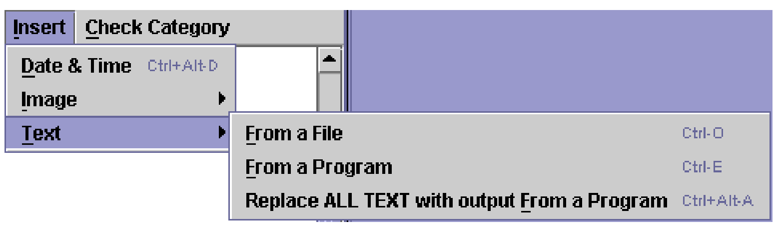
- If you are generating images, histograms or other files to import into **CRL** entries, you should create/store them under the default directories. Ask your **CRL** administrator the locations of the default directories.
- Set your container window width to something reasonable for the data type. Lines wrap, so there is no need to insert carriage returns after each text line.
- You can add text to all logbook entry types except forms.
- You can add images to all logbook entry types except plain text and forms.
- All images appear as thumbnails in **CRL** containers, on the web, and in emailed entries.

3.3.1 Text

There are two types of text entry: (Formattable) Text and Plain Text. Both types allow input from the keyboard and from a copy/paste operation. Tabs are set up so that a tab gives you one space. You can use either type for a plain text entry, but there are some differences:

Text	the default text font can be changed, and images can be inserted into entries
Plain Text	the monospace text font cannot be reformatted, and entries can contain plain text only (no images); use Plain Text for information that requires particular spacing, e.g., tables, line art, program output (text)

Create the entry and underneath the header, you'll see the highlighted text: `Insert Text Here`. Click in the text area; this initial text will disappear. Begin typing, use the **INSERT** menu to insert text from a file, or paste in text copied from an external application. The **INSERT** menu provides three options:



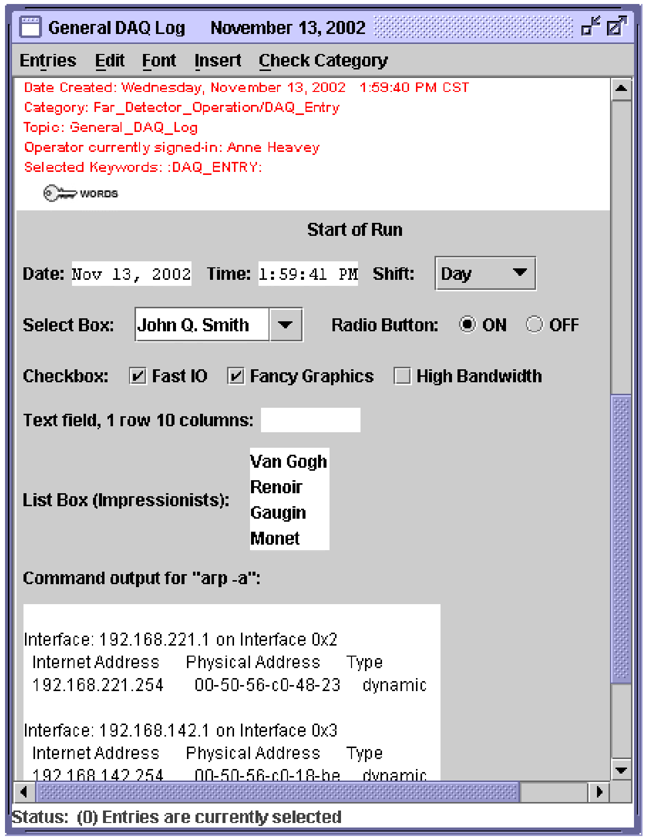
Note that if text includes HTML mark-up, it will appear as HTML source code in the entry as viewed from within **CRL**. When viewed from a browser, it will appear as HTML output.

3.3.2 Forms

To make a form entry, use the toolbutton corresponding to the form you want to use, as configured. Fill in the fields. All text is displayed in a monospaced font.

Forms may contain familiar form elements: check boxes, radio buttons, pull-down menus, and so on. They can also contain script/command output, tables and repeatable blocks. Items in pull-down menus may be configured by your **CRL** administrator to be editable. Data may be reloadable from previously-saved forms. A form may be configured such that it gets emailed to one or more addresses automatically upon archive; this is invisible to the user.

A form illustrating several of the allowed form elements is shown here:

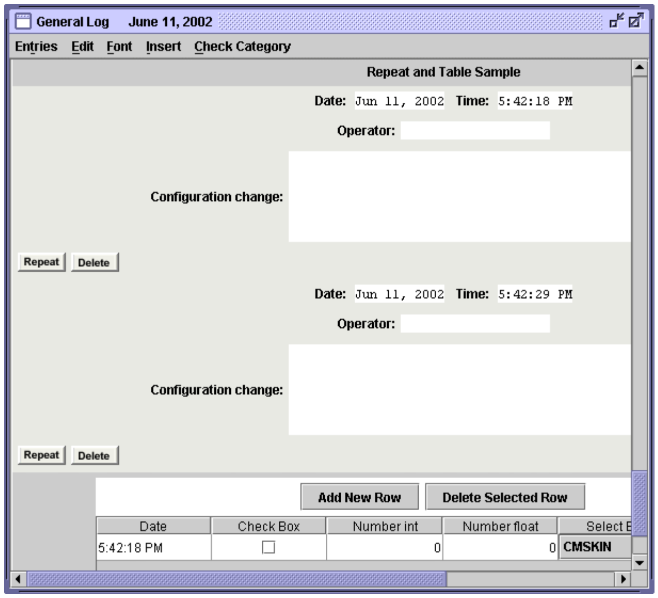


Program Execution within Forms

A form may be configured to run a script, an OS command or some other type of program. The form displays the (text-only) output in a field just as it displays other form data (see above image). The output is editable.

Repeatable Blocks within Form

A form may be configured such that a portion of it is repeatable. If so, that portion is demarcated by a different background shade and displays two buttons below and to the left of the repeatable portion: **REPEAT** and **DELETE**. Each time you repeat it, these buttons appear. **REPEAT** causes the block to be repeated; **DELETE** causes the block to be deleted.

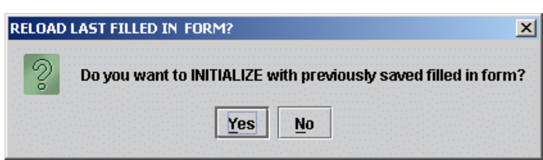


Tables within Forms

A form may contain a table, the cells of which you can edit. The cells remain scrollable after archive. Two buttons are displayed: **ADD NEW ROW** and **DELETE A SELECTED ROW**. These buttons are associated with the table portion of the entry, and appear above it (see bottom portion of above image).

Reloading Data into Form

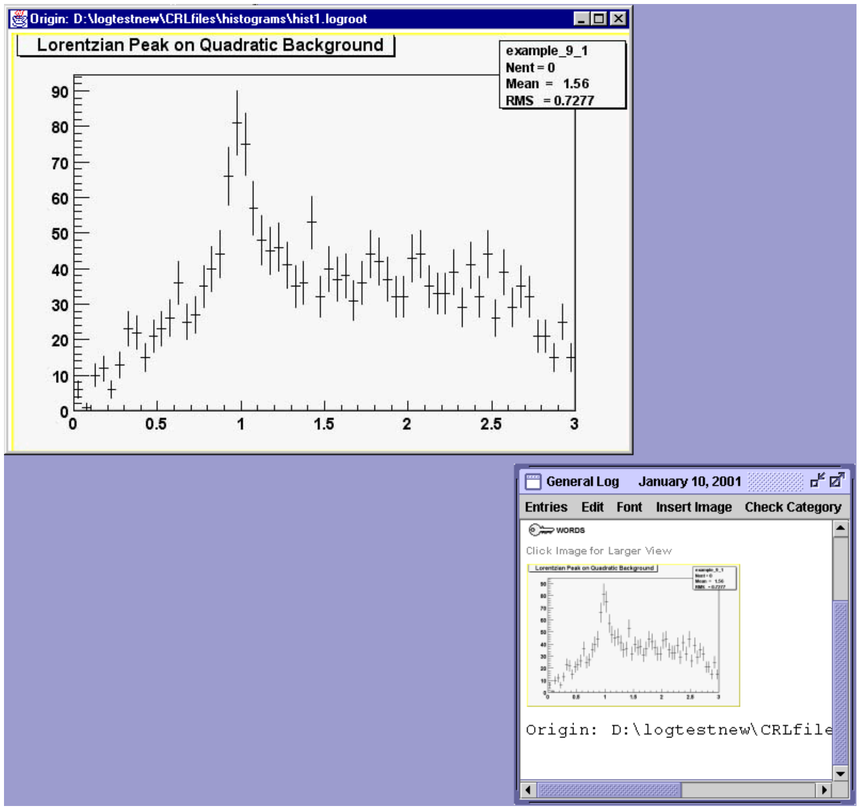
Depending on how or if your system is configured to allow reloading of the previously saved form data, the form may come up with some or all data filled in (which you can edit), and/or a dialog box may appear on which you can choose whether to load the saved data into your new form entry:



If you click **YES**, the text areas restore with their original sizes. You can edit the values after loading them.

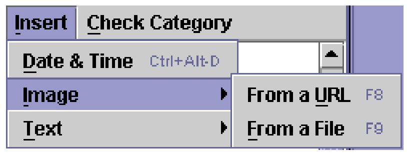
3.3.3 Online Images

Images can come from a variety of sources, depending on your configuration. Gif and jpg images that are too big for the default container window size (or larger than a particular size configured for your installation) will appear as thumbnails. A thumbnail is displayed with the words *Click Image for Larger View* inserted above it, `Origin: path/to/file` below it, and a cursor to allow text entry. Click the thumbnail image to see the larger image.



Possible image sources include:

- Toolbar buttons with preconfigured images
- Toolbar buttons configured to allow entry of output image files from external applications (see section 3.3.4 *Output Files from External Applications*)
- The **INSERT** > **IMAGE** > **FROM A URL/FILE** menu options on the container (available for all entry types except plain text)



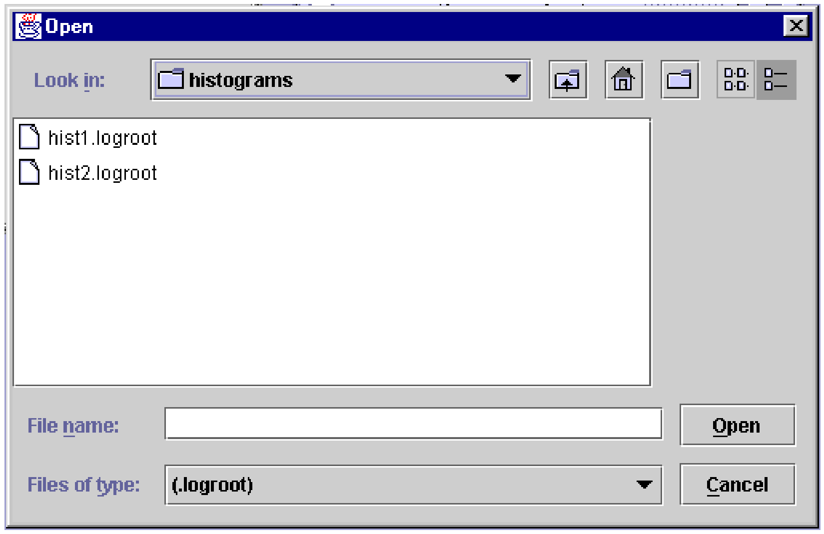
To use the **INSERT** menu (the third bullet point above), place your cursor in the body of the entry in which you want to insert an image. This menu option allows you to choose your image source, either a URL or a file:

- If you select **URL**, a window pops up and asks you to **TYPE THE URL OF THE IMAGE YOU WOULD LIKE TO INSERT:**. Type in the entire URL (it must start with `http://`), then choose **OK** to insert it, or **CANCEL** to cancel out of the operation. From a web page you can only enter gif or jpg files, not web pages themselves.
- If you select **FILE**, a window pops up and asks for the **IMAGE FILE NAME**:. Type it in (using full path if not in default directory) or click the **BROWSE** button to browse for it in the standard way. When you've entered the filename, choose **OK** to insert it, or **CANCEL** to cancel out of the operation.

3.3.4 Output Files from External Applications

CRL supports logbook entry of output files from virtually any software application, e.g., **ROOT** plots, **MS Word** and **Excel** files, images, PDF and PostScript files, etc. (This refers to output files that are created independently of **CRL**, and is distinct from output files that result from script/program execution in **CRL**.) If implemented, you'll see toolbar buttons for items such as **ROOT DATA**, **HISTOGRAMS**, **ATTACH FILE**, or whatever your **CRL** administrator has configured. For entries of this kind, the selected file gets attached to the entry; it does not get inserted. The file content does not appear in the entry.

When you create an entry, a dialog box pops up and asks for the **FILE NAME**:. Type in the name (with path, as necessary) or click the **BROWSE** button to browse for it. There may or may not be file type filters set on the browse window (the sample below shows one filter, `.logroot`).



Once the file name is entered, choose **OK** to attach the file, or **CANCEL** to cancel out of the operation.

If a thumbnail image displays in the entry, click it to open the entire document in a separate application and window, as described in section 3.3.3 *Online Images*. This works whether entry is archived or not. (After archiving the entry, buttons appear in the entry allowing you to download or view the file. See section 8.6 *Downloading/Viewing Attached Files*.)

When does CRL Display a Thumbnail?

You may find that in some cases you get a thumbnail image and in others you don't. When you attach a file to an entry, **CRL** searches the source directory of the file for a `.gif` or `.jpg` image file of the same filename, e.g., `hist1.gif` to go with `hist1.logroot`. If the accompanying image file exists, **CRL** copies it and uses it to provide a thumbnail view of the file in your container window.

PostScript, Encapsulated PostScript and/or PDF files (`.ps`, `.eps`, `.pdf`) don't necessarily need a pre-existing, corresponding image file in the source directory in order for **CRL** to display a thumbnail. If a PostScript or PDF converter is installed on your machine and **CRL** is configured to use it, then **CRL** uses the converter to create a jpg image file of the first page of the file, and displays it in the entry as a thumbnail.

3.3.5 Freehand Equations, Drawings and Notes

CRL supports use of a Cross Ipen® tablet connected via a serial port in order to allow freehand data entry. Typically, **CRL** is configured to display a toolbar button labelled **Pen**, or **DRAW**, or something similar. This toolbar item pops up a drawing frame, and requests entry of the **COM PORT:** to which the tablet is attached. After selecting the port name, click **OPEN PORT**, then begin to draw or write on your tablet. When you've finished, click **SAVE DRAWING** to save your work as a gif file and enter it in the container. Or click **CANCEL** to cancel out of the operation.

3.3.6 Output from a Program or Command

CRL can be configured to run a script, program or command to the operating system in four ways:

1) A toolbar button can be designated for a preconfigured command, and labelled accordingly (e.g., **NETSTAT** or **MYSCRIPT**). The command/script output gets dropped into the entry.

2) A toolbar button can be configured to allow the operator to type in any command string. The button might be labelled **EXECUTE**, or **RUNCMD**, or something similar. It pops up a window requesting you to **ENTER COMMAND LINE:**. Enter the command, and choose **OK** to execute it; its output will get dropped into the container. Or choose **CANCEL** to cancel out of the operation.

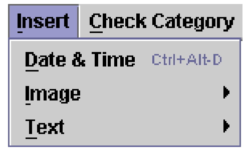
3) You can use the **INSERT > TEXT > FROM A PROGRAM** menu option on a container. It pops up a window requesting you to **ENTER COMMAND LINE:**. Enter the command, and choose **OK** to execute it; its output will get dropped into the container. Or choose **CANCEL** to cancel out of the operation.

4) A form can be configured to run a script, OS command or other program.

3.4 Including Current Date and Time in Entry

The date and time of creation and saving of an entry are recorded in the entry's header. But you still may want intermediate times recorded in an entry. Instead of typing the current date and time in, you can enter it via one of two shortcuts. Both methods insert the date and time in the format `1:54:33 PM CST Feb 13, 2002`. First position your cursor in the desired entry.

One method is the menu option **INSERT > DATE & TIME**:



The other is the control key sequence: **CTRL-ALT-D**. Note that the behavior of this key sequence is affected by your X window manager. Known behaviors:

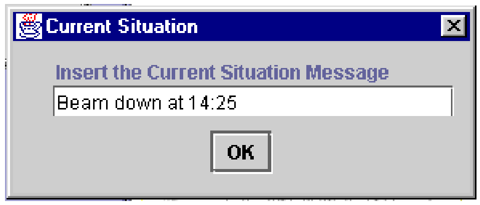
- it works fine under KDE and FVWM
- it doesn't work at all under Gnome
- Reflection X gets stuck on the **CTRL** and you just need to toggle **CTRL** after typing **CTRL-ALT-D** in order to restore proper **CRL** behavior.

3.5 Attaching Global Message to All Entries

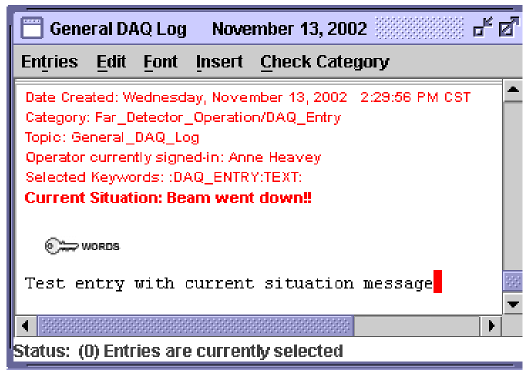
CRL provides a mechanism to turn on a global message as experimental conditions warrant (e.g., "Beam down at 14:25"). Once set, the message gets included with all logbook entries for all data types for that **CRL** installation. When the current situation changes, the operator can turn off or change the message. The **CURRENT SITUATION** button on the logbook toolbar (left side of **CRL** window) is used to turn a global message on and off.

3.5.1 Turn On a Message

To set a message, click the **CURRENT SITUATION** button. A window pops up requesting you to **INSERT THE CURRENT SITUATION MESSAGE**. Type it in, then click **OK**.



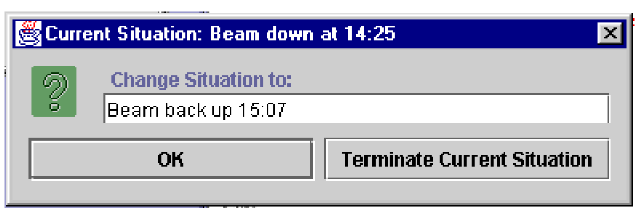
If you choose **OK**, you will see the message appear in red, labelled *Current Situation*, at the bottom of the header in every subsequent logbook entry, manual or automated. (When the entry is archived, the message will stay red.)



3.5.2 Change or Turn Off a Message

When the situation described in the message no longer applies, you should either change the message to reflect the new current situation or turn it off:

- To change the message, click the **CURRENT SITUATION** button, and edit the message. Click **OK**.



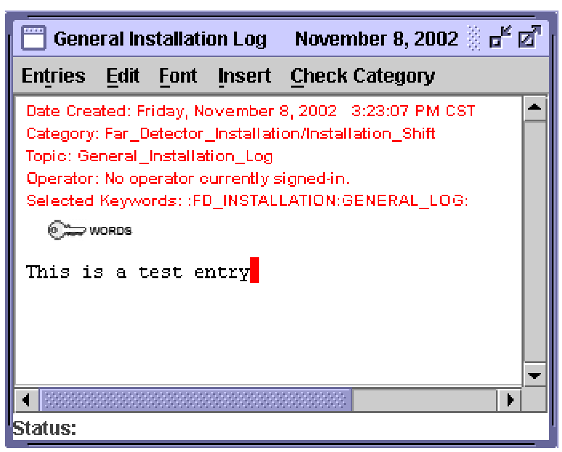
- To turn off the message, click the **CURRENT SITUATION** button, and choose **TERMINATE CURRENT SITUATION**. Subsequent entries do not include the message.

Chapter 4: Editing Logbook Entries

Before a logbook entry is archived to the **CRL** database, its content and/or appearance can be modified, or it can be deleted. In this chapter we discuss the menu options available on the entry input containers, and we describe how to select and edit your logbook entries.

4.1 Input Container Menu Options

☞ The menu options given on the container window operate on existing logbook entries, they do not create new ones. For example, **INSERT IMAGE** inserts an image where the cursor is, assuming you have already inserted an entry; if no entry header exists in the container, the image does not get inserted.



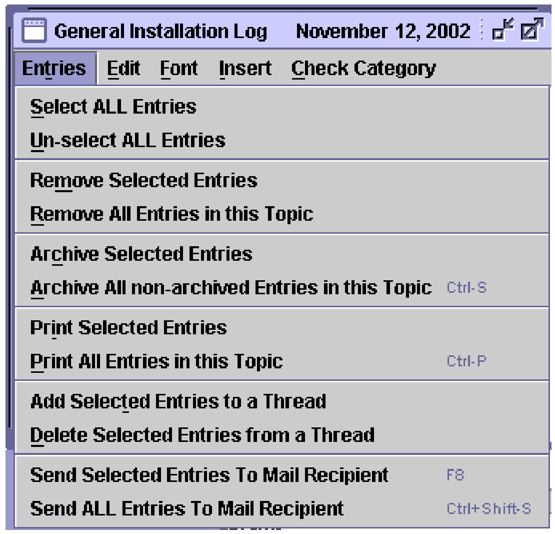
The menu options and their functions include:

Entries

Bring up the **ENTRIES** menu either by clicking on the **ENTRIES** menu option, or by right-clicking anywhere in the container window.

Select/unselect all entries, remove, archive, print, add/delete entries to/from a thread, or send via email.

For non-archived entries, the **REMOVE** options delete the entry, after a confirmation query. For archived entries, these options only remove the entries from the input container; they are no longer displayed in the container window, but they remain unchanged in the database.

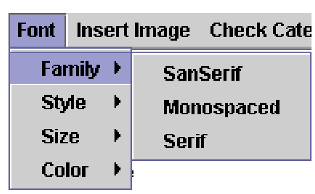


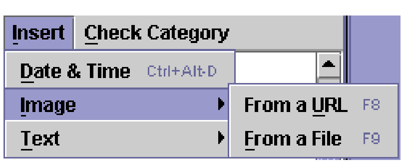
Edit

Edit text in an entry (undo and redo, cut, copy and paste text, select all text).

Font

Set family, style, size, and color of text in entry.





Insert

Insert into an entry: the current date and time[a], an image from a file or URL, or text from a file or program.

Check Category

Display menu path of container.

a. You can also insert date and time using Ctrl-Alt-d.

If there is nothing editable in the container (e.g., all entries are archived), all menus except **ENTRIES** and **CHECK CATEGORY** are grayed out. When the cursor is in the body of a particular entry, the features not available for the current entry type are grayed out.

4.2 Selecting Logbook Data

In order to modify or archive logbook data, you need to be able to select entries. You can select one entry at a time or all the entries in the container at the same time. You can select non-archived and/or archived entries. Nonselected entry headers have a white background. When an entry is selected, its header background turns gray.

The status bar at the bottom of the container displays the number of selected entries.

4.2.1 Select/Deselect Single or Multiple Entries

select single entry	Point anywhere in the header of the desired entry, and click once.
deselect single entry (only one entry selected)	Click once in the entry's header. (The click toggles between select/deselect.)
deselect single entry (only one entry selected) and select different entry simultaneously	Just click in the header of the new entry.
select multiple entries	Select the first entry as described above, then press the **<CTRL>** key and click once in the header of another entry. Continue for additional entries
deselect one of multiple selected entries	Press the **<CTRL>** key and click once in the header of entry to deselect.

select all entries	Open the **ENTRIES** menu (either click on it or right-click anywhere in the container window). Choose the **SELECT ALL ENTRIES** option in the **ENTRIES** menu. This selects non-archived and archived entries.
deselect all selected entries	Click twice (not a double-click; two singles[a]) on the header of one selected entry. Or choose the **UN-SELECT ALL ENTRIES** option from the **ENTRIES** menu.

a. The first click deselects all but the one you're on, the second deselects the current entry.

4.2.2 Select Text within an Entry

You can select text in an entry. To select it, simply drag the cursor over the desired text, or use the shift and arrow keys, as usual.

To select all the text in a single entry, use **CTRL-A** or choose the **SELECT-ALL** option from the **EDIT** menu.

To deselect the text, click once anywhere in the entry.

4.3 Modifying Non-archived Logbook Entries

After an entry is archived, it cannot be changed in the container window or deleted from the **CRL** database. Prior to that, text in entries can be edited, text and images can be added or deleted, and whole entries can be deleted. **UNDO** and **REDO** functions are available from the **EDIT** menu (also implemented as **CTRL-Z** and **CTRL-Y**, respectively); these functions stack successive user operations.

4.3.1 Add to or Change Text in an Entry

To add to text in any entry, just place the cursor where you want it and start typing. To replace a block of text, select the text (see section 4.2.2 *Select Text within an Entry*), and start typing over it.

4.3.2 Cut, Copy and Paste Text

The **EDIT** menu contains options to cut, copy and paste text to and from the clipboard. The **<CTRL>-X**, **<CTRL>-C** and **<CTRL>-V** keys work in the standard way, as well. Text can be copied within the same input container, between different input containers, and from or to external applications.

Within CRL or to/from External non-X Application

Cut/Copy:

- To cut or copy text from a logbook entry, first select the item, as described in section 4.2.2 *Select Text within an Entry*. Choose **CUT-TO-CLIPBOARD** or **COPY-TO-CLIPBOARD** from the **EDIT** menu, or press **<CTRL>-X** or **<CTRL>-C**, respectively.
- To cut or copy text from an external non-X application, use the standard procedure (usually **<CTRL>-X** or **<CTRL>-C**).

Paste:

- To paste text into a logbook entry, place the cursor in the entry where you want to paste the item, and choose **PASTE-FROM-CLIPBOARD** from the **EDIT** menu, or press **<CTRL>-V**.
- To paste text into an external non-X application, use the standard procedure (usually **<CTRL>-V**).

To copy text from the VT100 window into a **CRL** entry:

1) select the text from the window with the left mouse button as usual, then right click to get pop-up menu, and choose **COPY** (or press **CTRL-INSERT**).

2) in the **CRL** entry container, place the cursor where you want the material copied, and press **CTRL-V** or select **PASTE-FROM-CLIPBOARD** from the **EDIT** menu.

Between CRL and X Window Applications (Linux)

To copy text into a **CRL** entry using the standard X Window method:

1) In the **CRL** window, select the target container and create the target entry.

2) In your external X application, highlight the content that you want to paste into your **CRL** entry.

3) In the **CRL** container window, using your middle mouse button, click in the target entry.

To copy text into a **CRL** entry using the clipboard, first make sure your X application uses the clipboard in the standard way, then:

1) In your X application window, highlight the content of interest. Then select **COPY** to place it in the clipboard.

2) In the **CRL** window, select the target container. Then create or click in the target entry.

3) Select **PASTE-FROM-CLIPBOARD** from the **EDIT** menu, or type **CTRL-V**. You should now see the content in your entry.

To copy text from a **CRL** entry to an X Window application:

1) Highlight the content in your **CRL** entry that you want to paste into your external application. Then select **COPY-TO-CLIPBOARD** from the container's **EDIT** menu, or type **CTRL-C**.

2) Position your cursor in your X application window and click the middle mouse button to paste the highlighted content into this window.

4.3.3 Undo and Redo Operations

UNDO and **REDO** functions are available from the **EDIT** menu; these functions stack successive user operations.

Shortcut keys:

UNDO **CTRL-Z**

REDO **CTRL-Y**

4.3.4 Change Text Font

CRL provides choices for font family, style, size and color for text in logbook entries (although not for *plain text*). To change the text font, first select the desired text, as described in section 4.2.2 *Select Text within an Entry*. Then choose the font property(ies) you want from the **FONT** menu.

To undo font properties, you can use the **UNDO** function on the **EDIT** menu or you can just select a different property (note that for **STYLE**, clicking on the same one again toggles the style). **REDO** is available, too.

4.3.5 Delete Entries

There are two menu options for removing logbook entries: **REMOVE SELECTED ENTRIES** and **REMOVE ALL ENTRIES IN THIS TOPIC**. As mentioned in section 4.1 *Input Container Menu Options*, when operating on a non-archived entry, the **REMOVE** function truly deletes the entry (after a confirmation query). Archived entries simply get removed from the container with no prompt (see section 5.3 *Removing Archived Logbook Entries from View*).

To delete a single entry or a set of hand-picked entries, first select each entry as described in section 4.2.1 *Select/Deselect Single or Multiple Entries*. Then choose **REMOVE SELECTED ENTRIES** from the **ENTRIES** menu. **CRL** prompts for confirmation before the deletion of each entry.

To delete all non-archived entries in the container, simply choose **REMOVE ALL ENTRIES IN THIS TOPIC** from the **ENTRIES** menu. **CRL** prompts for confirmation before the deletion of each entry. Any archived entries will be removed from the container as well.

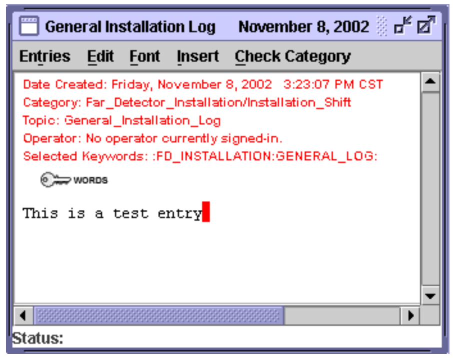
For (non-archived) entries that have been checkpointed automatically, the checkpoint entry will not be removed when you delete the actual entry. To delete these (or other) checkpointed entries, click the **CHECKPOINT RECOVERY** button on the Logbook toolbar. A window comes up containing a list of all the entries that are currently checkpointed. Select the entry you want to delete, and click the **DELETE SELECTED FILES** button at the top of the frame.

4.4 Attaching Keywords to a Logbook Entry

Keywords can be linked to logbook entries in order to provide an additional dimension for querying the database when attempting to later identify and retrieve particular entries. Keywords are stored in UPPERCASE. Each experiment defines its own keywords in accordance with the logbook entry pages, categories, topics and data types it has configured.

Typically the keyword(s) you'd want to attach either come automatically attached or appear in the entry's list of default keywords. However, *any* keyword defined in your **CRL** configuration can be attached to an entry via the **GLOBAL KEYWORDS** feature, discussed in section 4.4.2 *Global Keywords*.

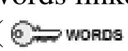
In the container, a data header appears in red. Attached keywords, if any, are listed in the last line of the header. If keywords are available for this entry, the keyword symbol appears under the header.



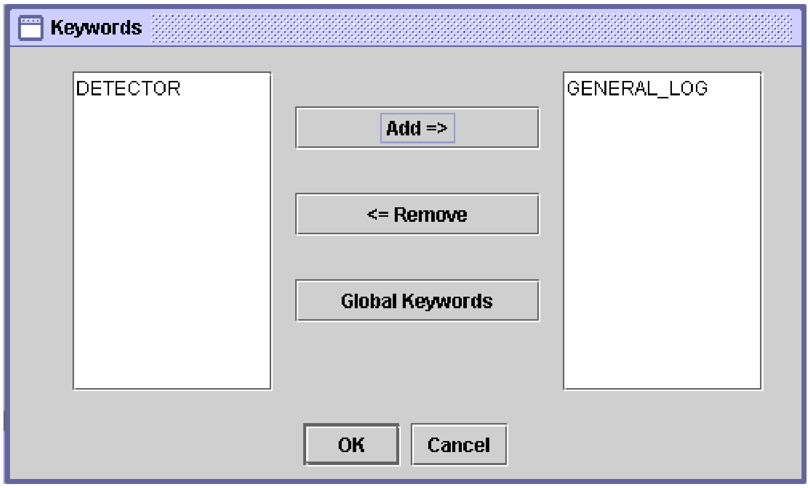
.

4.4.1 Keywords Tailored to Entry's Topic and Data Type

Each input container may have its own set of default, "attachable" keywords pertaining to the container topic. Any or all of these keywords may be configured to link automatically to each entry in the container. Similarly, each data type may have keywords associated with it. Therefore, every logbook entry of a given data type inserted into a given input container has the same set of default, attachable keywords, and you can choose from among them. Some of the automatically linked keywords may be removable, others may not. On any given logbook entry, you can only remove keywords that are removable (to state the obvious!).

To edit the list of keywords linked to a logbook entry, click on the keyword symbol for that entry (

) to pop up the **KEYWORDS** window:



In the left-hand column is the list of default, attachable keywords. In the right-hand column is the list of keywords attached to the entry (initially, this is the set of auto-linked keywords).

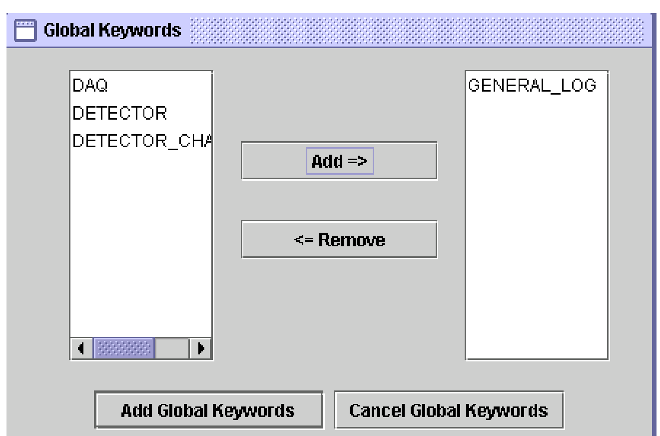
To attach keyword(s): select from the left-hand list and click **ADD=>**

To remove keyword(s): select from the right-hand list and click **<=REMOVE**

To add/remove multiple keywords at a time, use the standard method of selecting multiple items for your windowing system. E.g., for Windows, use the **CTRL** or **SHIFT** key (**CTRL**-click individual, nonadjacent items, or **SHIFT**-click first and last of set of adjacent items).

4.4.2 Global Keywords

If you wish to attach a keyword that does not appear in the list of default attachable keywords for the entry, click **GLOBAL KEYWORDS**. This brings up the **GLOBAL KEYWORDS** box which lists all the keywords defined for the **CRL** installation.



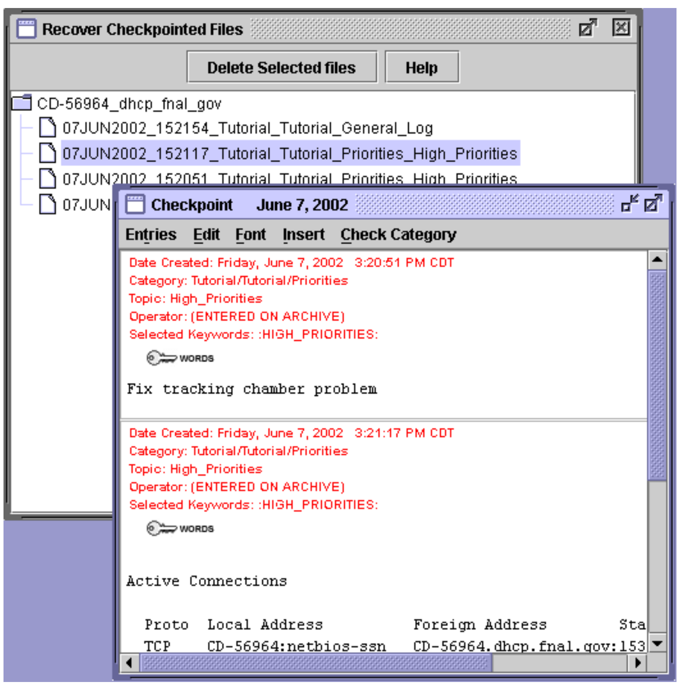
In the same way as for the keywords discussed in section 4.4.1 *Keywords Tailored to Entry's Topic and Data Type*, select any keywords from the left-hand column and add them to the right-hand column. Click **ADD GLOBAL KEYWORDS** when done, and the selected keywords will appear in the right-hand column of the **KEYWORDS** box and be attached to the entry. Click **CANCEL GLOBAL KEYWORDS** to cancel.

Once you're back on the **KEYWORDS** window (described in section 4.4.1 *Keywords Tailored to Entry's Topic and Data Type*), if you remove a global keyword from the right-hand column, it will not appear in the left-hand column since it did not appear there initially.

4.5 Recovering or Deleting Checkpointed Entries

Entries that have not yet been archived or deleted get written out periodically to a checkpoint directory. This protects against significant data loss in the event of a crash or accidental deletion.

To recover checkpointed entries, click the **CHECKPOINT RECOVERY** button on the Logbook toolbar on the left. A window comes up containing a list of all the entries that are currently checkpointed (see the background window in image):



Select the entry or entries you want to recover, and drag it (them) into a Checkpoint Container or into an Input Container of the same Category and Topic. A Checkpoint Container accepts any checkpointed entry, and thus allows you to drag all the entries at once. An Input Container, in contrast, has a particular Category and Topic, and it accepts only entries that match. Both container types allow you to edit, delete and/or archive the recovered entries. The **ENTRIES** menu for a Checkpoint Container is identical to that of an Input Container (see section 4.1 *Input Container Menu Options*).

To delete checkpointed entries, select them in the **RECOVER CHECKPOINTED FILES** window, and click **DELETE SELECTED FILES**. Checkpointed entries may need to be purged now and then.

To select multiple checkpointed entries at a time, use the standard method of selecting multiple items for your windowing system. E.g., for Windows, use the **CTRL** or **SHIFT** key (**CTRL**-click individual, nonadjacent items, or **SHIFT**-click first and last of set of adjacent items).

4.6 Printing Entries

There are two **ENTRIES** menu options for printing entries from a container: **PRINT SELECTED ENTRIES** and **PRINT ALL ENTRIES IN THIS TOPIC** (the latter can also be performed using **CTRL-P**.) When you use one of these options, a standard **PRINT** window for your operating system pops up in which you set the destination printer and printer options. The printed report will contain the entries in the container corresponding to the entries you selected and the menu option you selected.

4.7 Sending Entries via Email

The **ENTRIES** menu contains options to send entries, archived or not, to an email recipient or to a mailing list. Entries can be of any data type (virtually all MIME types are supported), and are sent in HTML format. The two options are:

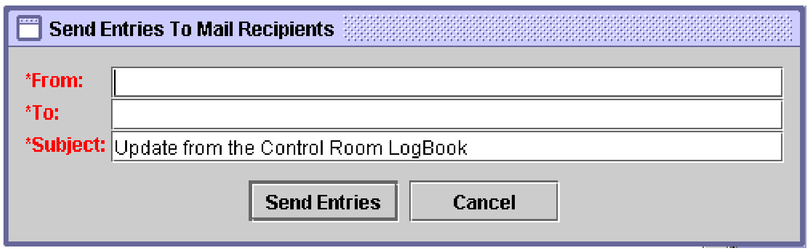
SEND SELECTED ENTRIES TO MAIL RECIPIENT (**F8**)

Select entries to be sent, using a technique from the table in section 4.2.1 *Select/Deselect Single or Multiple Entries*. Then click this option on the **ENTRIES** menu, or press **F8**.

SEND ALL ENTRIES TO MAIL RECIPIENT (**CTRL+SHIFT+S**)

Verify that you want to send all the entries currently in the container, archived and not, to the recipient. Then click this option on the **Entries** menu, or press **Ctrl+Shift+S**.

The **Send Entries to Mail Recipients** dialog box pops up in which you identify yourself, the recipient and a subject (a default subject is provided):



In the **To:** field, enter either a single email address or a mailing list address. As of v1_7_04, multiple addresses are not supported.

Note that **CRL** may be configured such that certain form entries (and *only* form entries) get sent automatically upon archive to designated recipients. This action depends on the configuration of the form's toolbutton. The user is not notified in any way when this happens, so ask your **CRL** administrator if this applies to your installation.

Chapter 5: Archiving Logbook Entries to the CRL Database

Once a logbook entry has been archived, the entry gets “set in stone”. The only change you can make is to add an annotation, as described in section 8.5 *Annotating Archived Entries*. In this chapter we describe how to archive your logbook entries.

5.1 About Archiving Logbook Entries

5.1.1 General Information for All Entry Types

You can select one or more particular entries to archive (entry selection is described in section 4.2 *Selecting Logbook Data*), or you can choose to archive all the entries in the input container that have not been previously archived. Entries that contain no data do not get archived. Each entry gets archived as a whole; you cannot archive part of an entry. When an entry is archived, its checkpoint file, if any, gets deleted. There is no confirmation prompt on archive.

You can choose to add the entry to an existing thread when you archive it. You can create a new thread at archive time and link the entry to it. Threading is described in Chapter 7: *Threading Logbook Entries*.

After an entry is archived, it cannot be changed or deleted; it can be annotated. It can be removed from the input container, but not from the database. Before archiving, verify that the logbook entries you plan to archive are complete and that they explain/display the experimental conditions and events clearly.

Depending on the login configuration of your installation (see Chapter 2: *Invoking and Logging Into CRL*), the operator names that are associated with an entry as it gets archived are:

- for the “username and password” configuration, those in the **ENTRY SIGNERS** list (see section 2.3.1 *About the Entry Signers Feature*). If the list is empty, the entry will be logged under the operator name “Anonymous”.
- for the “password and PIN” configuration, all usernames sharing the PIN that is entered

For form entries, a system may be configured such that when the **CRL** user archives this type of entry:

• a dialog box appears and allows the user to save the form data for eventual inclusion into a later form entry.

• an email message gets sent automatically to one or more preconfigured email addresses. The message contains the entry in html format. The **CRL** user is not alerted if/when email is sent.

For any form entry containing a table, the buttons **ADD NEW ROW** and **DELETE A SELECTED ROW** are no longer displayed after archive. This is also true for **REPEAT** and **DELETE** on a repeat block.

Cells within a table are scrollable when an archived entry is restored via a search in **CRL**.

5.2 Archiving Entries

Frst determine what you want to archive, pull down the **ENTRIES** menu or right-click to pop it up, then choose the appropriate menu option:

Archive individually selected entries:

first select the entries, then choose **ARCHIVE SELECTED ENTRIES**

Archive all non-archived entries:

press **CTRL-S**

or choose **ARCHIVE ALL NON-ARCHIVED ENTRIES IN THIS TOPIC**

If your **CRL** installation uses the “password and PIN” login configuration, you will be prompted for a PIN for each individual entry you attempt to archive. If you’ve selected several entries, the prompt will pop up each time the application is ready to process the next entry.

5.3 Removing Archived Logbook Entries from View

There are two menu options for removing logbook entries: **REMOVE SELECTED ENTRIES** and **REMOVE ALL ENTRIES IN THIS TOPIC**. As mentioned in section 4.1 *Input Container Menu Options*, when operating on an archived entry, the entry is removed from the container only. The archived entry is no longer displayed in the container window, but it remains unchanged in the database.

To remove a single archived entry or a set of hand-picked archived entries, first select each entry as described in section 4.2.1 *Select/Deselect Single or Multiple Entries*. Then choose **REMOVE SELECTED ENTRIES** from the **ENTRIES** menu. **CRL** does not prompt for confirmation before the removal of archived entries.

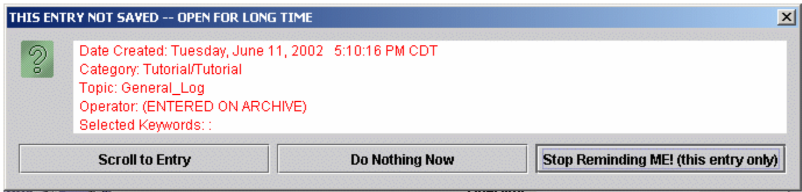
To remove all archived entries from the container, simply choose **REMOVE ALL ENTRIES IN THIS TOPIC** from the **ENTRIES** menu. Note that any non-archived entries in the container will be permanently deleted (see section 4.3.5 *Delete Entries*), subject to confirmation.

5.4 Entries Left Unarchived

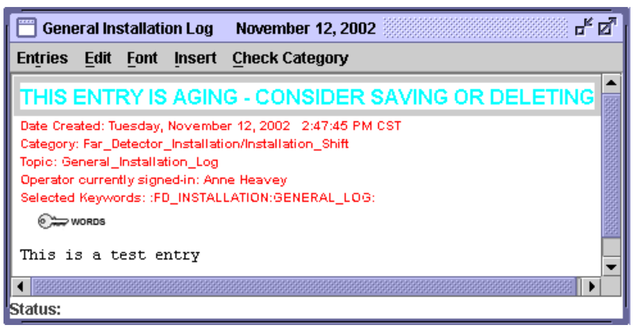
If an entry is left too long without being archived (where "too long" is a value determined and set by your **CRL** admin), your system may send messages periodically to ask you to address it. In this case, a window appears at set intervals listing the header of the entry in question, and it provides three options:

SCROLL TO ENTRY	This takes you to the entry in its input container. The entry may be on your desktop or another.
DO NOTHING NOW	As you’d suspect, nothing happens and the periodic messages continue.
STOP REMINDING ME! (THIS ENTRY ONLY)	

This stops the periodic messages for the given entry, but does nothing else.



The system also tags the unarchived entry in question with a note: THIS ENTRY IS AGING - CONSIDER SAVING OR DELETING:



The note is informational only and does not become part of the entry once it is archived. After the entry is archived or deleted, the messages for that entry cease.

Chapter 6: Automated Logbook Entry and Archiving

In this chapter we describe the process of setting up an autoscheduled logbook entry job.

6.1 Overview of the Autoschedule Feature

Automatic, scheduled logbook data entry is a configurable feature of **CRL**. If your experiment's **CRL** implementation supports it, you should have a scheduled container (described in section 1.2.3 *The CRL Desktop*) available on a desktop page. In this type of container, you can set up a job that generates entries automatically according to a schedule that you define. You must use a data type (a data entry toolbutton) that:

- requires no operator input
- points to a particular file from which it can upload data automatically into an entry (this assumes the file is periodically overwritten by an external program), or runs a program and uploads the output automatically into an entry

To start a job, you drag-and-drop the appropriate toolbutton into a scheduled container. This pops up the **SCHEDULE** dialog box in which you must set the scheduling parameters. Once a job is started, **CRL** performs the following steps automatically:

- for each scheduled file read, it creates an entry in the selected container
- it reads in the file and displays its contents (text or image, as appropriate) under the entry header
- it archives the entry automatically
- it may remove from view earlier entries as newer ones are added, if set

It is possible to set up multiple jobs to run concurrently in a single scheduled container.

6.2 Setting up an Autoscheduled Job

6.2.1 Open a Container and Select Entry Source

To set up scheduled logbook entries, first open the appropriate container:

1) Select the tag at the top of the **CRL** window to display the appropriate input page.

2) From the horizontal row of menu headings along the top of the page, select the one that corresponds to automated or scheduled entries (heading text is configurable, thus varies by experiment).

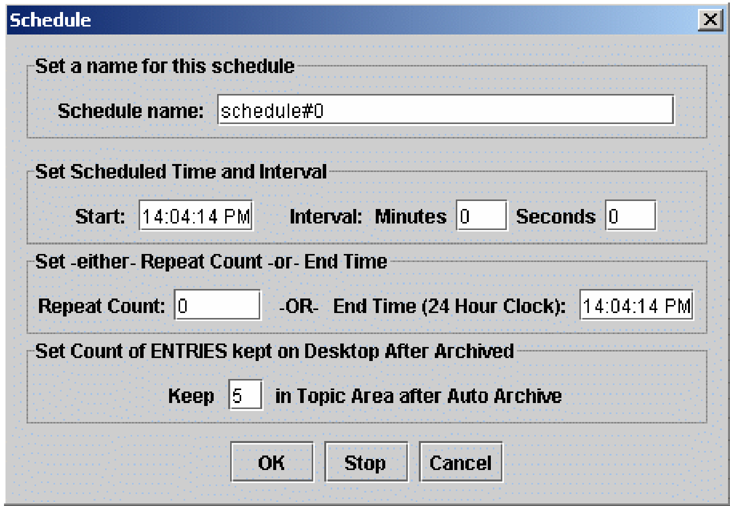
3) From its pull-down menu (or cascading pull-down menus), choose the category/topic appropriate for the logbook entries you intend to add. An input container for scheduled entry pops up.

4) Select the appropriate data type from the logbook entry toolbar at the right of the **CRL** window. (See section 6.1 *Overview of the Autoschedule Feature* for information on the data types that can be used.)

5) Drag-and-drop the toolbar button into the container. A **Schedule** dialog box pops up in which you will need to set up the job, as described in the next section.

6.2.2 Provide Autoscheduling Parameters

In the **Schedule** window, shown below, you must specify the scheduling parameters for the job.

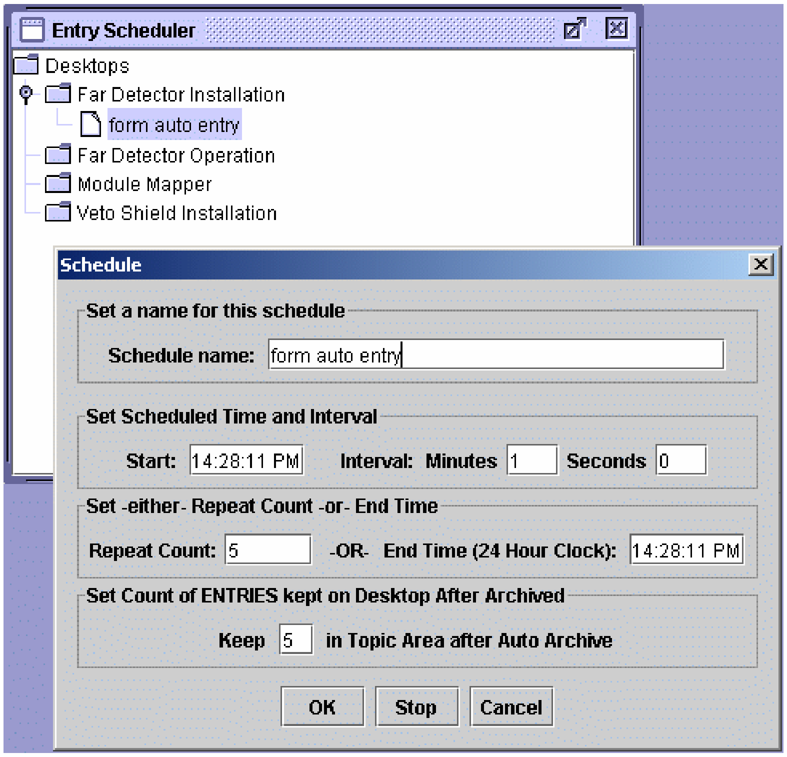


The parameters you need to set include:

- a name for the job (Schedule name), used to identify the job's entries in the **ENTRY SCHEDULER** window (described in section 6.2.3 *Change Scheduling Parameters or Stop Job*)
- the start time (for the initial logbook entry)
- the interval of time (in minutes and/or seconds) to wait before each subsequent entry
- *either* the number of times entry is to be repeated *or* the clock time at which the job is to stop (if a nonzero repeat count is given, **CRL** ignores the end time)
- the number of entries that you want to keep visible in the container as they accumulate (**CRL** drops the earliest entries, leaving the most recent ones visible)

6.2.3 Change Scheduling Parameters or Stop Job

If you want to change any scheduling parameters for a job in progress or terminate a job, click the **SCHEDULER** button on the left-hand side of the **CRL** window. This brings up the **ENTRY SCHEDULER** window, containing a folder icon for each desktop, and a file icon identifying each job in progress (see image below). If you see no file icons, then there are no jobs running. Double-click the appropriate file icon to bring up the **SCHEDULE** dialog box for that job, and change information as desired, or click **STOP** to terminate the job.


Entry Scheduler
Desktops
Far Detector Installation
form auto entry
Far Detector Operation
Module Mapper
Veto Shield Installation
Schedule
Set a name for this schedule
Schedule name:
form auto entry
Set Scheduled Time and Interval
Start:
14:28:11 PM
Interval: Minutes
1
Seconds
0
Set -either- Repeat Count -or- End Time
Repeat Count:
5
-OR- End Time (24 Hour Clock):
14:28:11 PM
Set Count of ENTRIES kept on Desktop After Archived
Keep
5
in Topic Area after Auto Archive
OK
Stop
Cancel

Chapter 7: Threading Logbook Entries

A thread is a vehicle to link a series of entries relating to a particular issue. In this chapter, we discuss creating and manipulating *threads*.

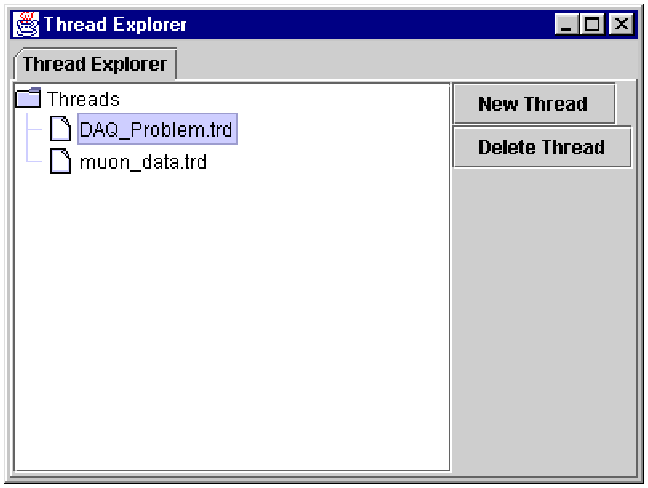
7.1 About Threads

A thread links entries relating to a particular issue. It provides easy access to the flow of information on an issue, given that the entries may have been entered in different desktop pages, categories and/or topics, and by different operators. Only archived entries can be threaded. Entries can be linked to multiple threads. The thread itself is nothing more than an XML file that contains pointers to all the archived entries that have been linked to it. The thread name is used as the XML file name. Threads are designed to create temporary links, they are not maintained in the database. A thread can, but does not have to be removed after the issue it chronicles gets resolved.

7.1.1 The Thread Explorer

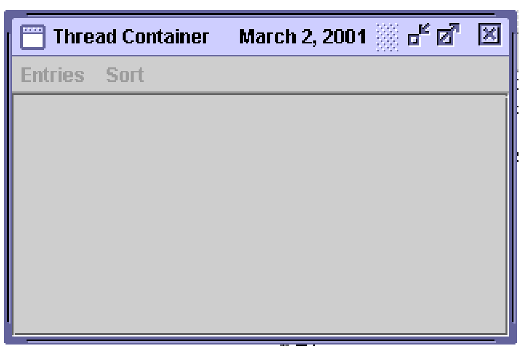
The **THREAD EXPLORER** button on the **CRL** toolbar brings up the **THREAD EXPLORER** window. This window displays the contents of the threads directory, which is the list of thread files. The directory name varies by installation. The image below shows the directory name as `Threads`, and it contains two threads. From this window you can:

- create a new thread
- delete an existing thread
- drop the contents of a thread into a thread container



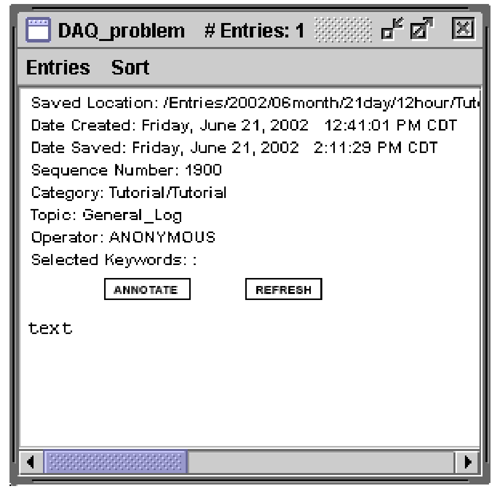
7.1.2 Thread Containers

A thread container provides a window for viewing and manipulating a thread. If configured, you will find a menu option on one or more of your data input desktop pages for threads (typically labelled **THREADS**). This menu item is used to pop up a thread container. Initially, the window is labelled **THREAD CONTAINER**.

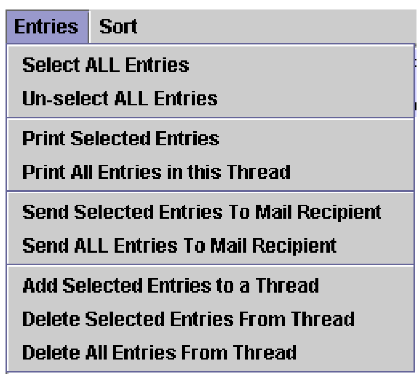


Once a thread is dropped into it, its label changes to the thread name, the number of entries is displayed, and the threaded entries appear in the container. In a thread container, entries can be:

- viewed
- selected and unselected
- printed
- sent via email to a recipient
- added to a different thread
- deleted from a thread
- annotated



as shown in the **ENTRIES** menu:



Via the **SORT** menu (not shown), the entries can be sorted by "date added to thread" (the default) or by "date created".

You can drop only one thread into a thread container.

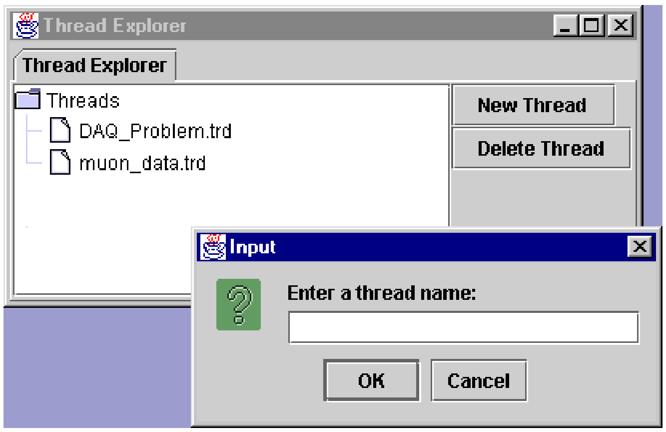
7.2 Creating a Thread

You can create a thread from the **THREAD EXPLORER**, from an entry input container, or from a thread container.

☞ When creating a thread, choose a thread name consistent with the filename constraints of the operating system in use; the XML file containing the thread will be named `<threadname>.trd`.

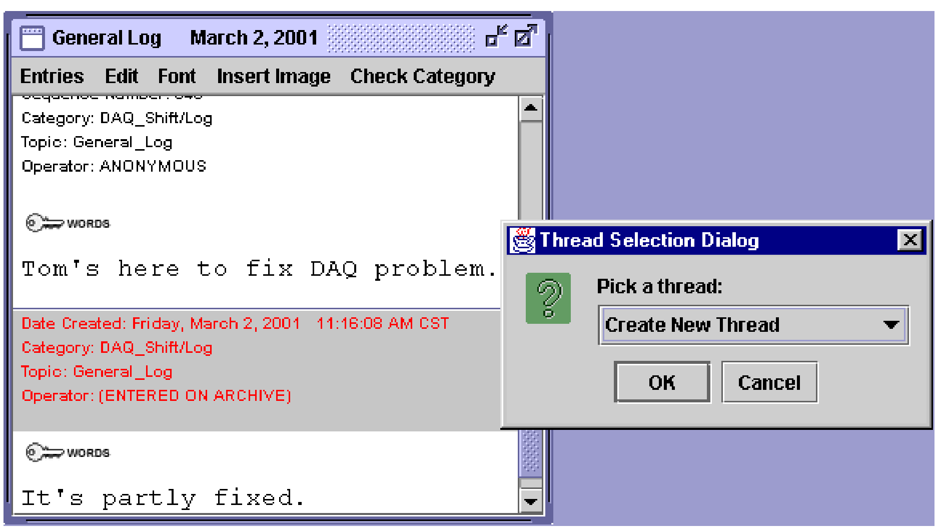
7.2.1 From the Thread Explorer

Click the **THREAD EXPLORER** button on the **CRL** toolbar to bring up the **THREAD EXPLORER** window. On the **THREAD EXPLORER** window, click **NEW THREAD**. You will be prompted for a thread name. Enter a name, and press **OK**.



7.2.2 From an Entry Input Container

First bring up an input container from the desktop page menu. On this container, select one or more entries that you want to include in a new thread. (Selecting entries is discussed in section 4.2 *Selecting Logbook Data*.) From the **ENTRIES** menu, click **ADD SELECTED ENTRIES TO A THREAD**. The system will prompt you for a thread name. Select **CREATE NEW THREAD** and type in a new name to create the thread.

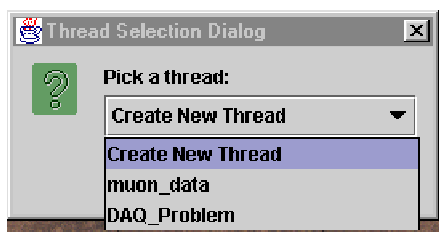


For any entries that are not yet archived, the system will prompt you to archive each one first.

If you respond **YES**, **CRL** will archive the entry and add it to the selected thread. If you respond **NO**, neither action will take place.

7.2.3 From a Thread Container

Bring up a thread container from the desktop page menu and also click the **THREAD EXPLORER** button on the **CRL** toolbar to bring up the **THREAD EXPLORER** window. Drag and drop a thread into the thread container. In the container, select one or more entries that you want to add to a new thread. (Note that all these entries are already linked to a thread; here you're selecting some to link to a second, new thread.) (Selecting entries is discussed in section 4.2 *Selecting Logbook Data*.) From the **ENTRIES** menu, click **ADD SELECTED ENTRIES TO A THREAD**. The system will prompt you for a thread name. Select **CREATE NEW THREAD** and type in a new name to create the thread.



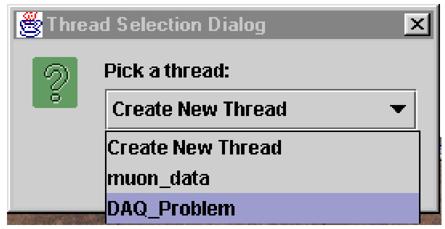
7.3 Adding Entries to a Thread

You can add entries to a thread from an entry input container or from a thread container.

7.3.1 From an Entry Input Container

On an input container, select one or more entries that you want to include in a thread. (Selecting entries is discussed in section 4.2 *Selecting Logbook Data*.) From the **ENTRIES** menu, click **ADD SELECTED ENTRIES TO A THREAD**.

The system will prompt you for a thread name. You can choose an existing thread name, or type in a new one.

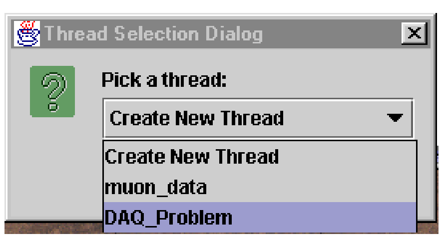


For any entries that are not yet archived, the system will prompt you to archive each one first.

If you respond **YES**, **CRL** will archive the entry and add it to the selected thread. If you respond **NO**, neither action will take place.

7.3.2 From a Thread Container

Bring up a thread container from the desktop page menu. From this container, select one or more entries that you want to add to a (different) thread. (Selecting entries is discussed in section 4.2 *Selecting Logbook Data*.) From the **ENTRIES** menu, click **ADD SELECTED ENTRIES TO A THREAD**. The **THREAD SELECTION DIALOG** box pops up. You can choose an existing thread name, or type in a new one.

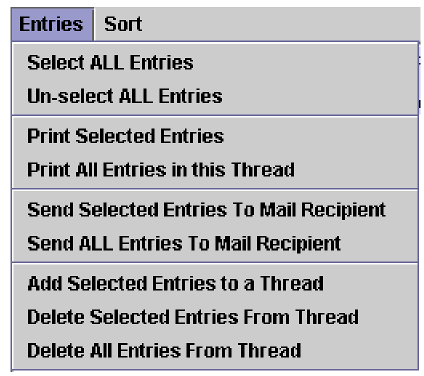


7.4 Viewing and Manipulating a Thread

The entries linked to a thread can be collected and manipulated in a thread container. To work with threads, first click the **THREAD EXPLORER** button on the **CRL** toolbar to bring up the **THREAD EXPLORER** window. Also bring up a thread container from the desktop page menu.

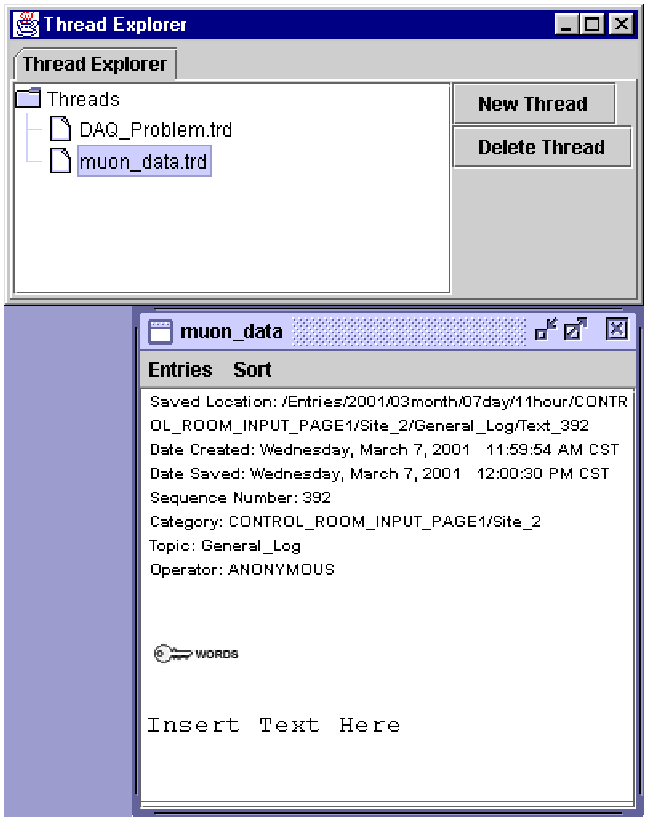
From a thread container you can view and sort entries, print them, send them via email, add them to a different thread, and/or delete them from the thread.

To see the menu options, click the **ENTRIES** menu (or right-click). The sorting options are under the **SORT** menu.



7.4.1 View and Sort Entries in a Thread

On the **THREAD EXPLORER**, select the thread you want to view, and drag it into a thread container. All the entries linked to the thread appear in the container. The default sorting is by date added to thread, in reverse-date order. You can change the sorting to sort by entry creation date, again in reverse-date order. In the image below, we show the **THREAD EXPLORER** window above the thread container. Notice that the thread container is labelled with the name of the thread that's displayed.



7.4.2 Print Entries in a Thread

There are two menu options for printing a report from a thread container: **PRINT SELECTED ENTRIES** and **PRINT ALL ENTRIES IN THIS THREAD**. When you click one of these options, a standard **PRINT** window pops up in which you set the destination printer and printer options. The printed report will contain entries in the container, according to your selection.

7.4.3 Send Entries to an Email Recipient

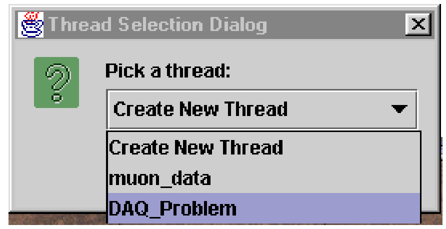
There are two menu options for sending logbook entries: **SEND SELECTED ENTRIES TO MAIL RECIPIENT** and **SEND ALL ENTRIES TO MAIL RECIPIENT**. These work similarly to the **SEND** options for input container menus, described in section 4.7 *Sending Entries via Email*, except that the shortcut keys are not implemented for thread containers.

7.4.4 Add Entries from One Thread to Another

To add a single entry or a set of hand-picked entries to a different thread, first select the entries you want in the thread container, as described in section 4.2.1 *Select/Deselect Single or Multiple Entries*.

Then choose **ADD SELECTED ENTRIES TO THREAD** from the **ENTRIES** menu.

The **THREAD SELECTION DIALOG** box pops up. You can choose an existing thread name, or type in a new one.



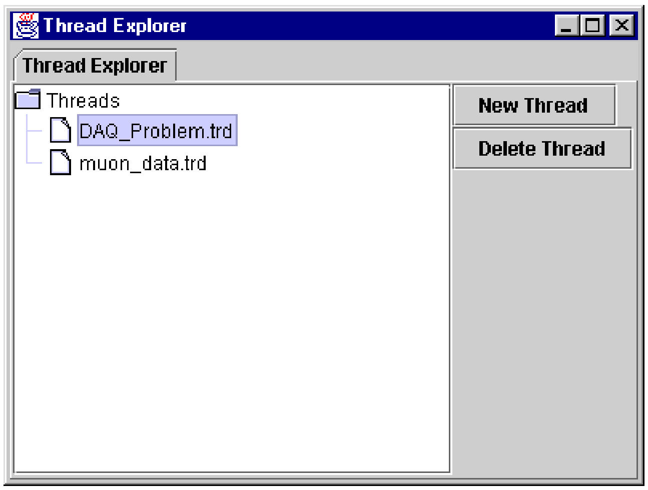
7.4.5 Delete Entries from a Thread

You can delete a single entry or a set of hand-picked entries from a thread. In the thread container, first select the entries that you want to delete, as described in section 4.2.1 *Select/Deselect Single or Multiple Entries*. Then choose **DELETE SELECTED ENTRIES FROM THREAD** from the **ENTRIES** menu. **CRL** prompts for confirmation before deleting.

To delete all entries from the thread, simply choose **DELETE ALL ENTRIES FROM THREAD** from the **ENTRIES** menu.

7.5 Deleting a Thread

Click the **THREAD EXPLORER** button on the **CRL** toolbar to bring up the **THREAD EXPLORER** window. Select the thread you want to delete. On the **THREAD EXPLORER** window, click **DELETE THREAD**. A confirmation prompt is provided. This deletes the XML thread file only; the entries that were in the thread remain archived.



Chapter 8: Working with Archived Entries

In this chapter we discuss searching, viewing, annotating and printing archived entries from within the **CRL** application and from the Web.

8.1 Accessing Archived Entries from within CRL

The **SEARCH** button on the **CRL** toolbar provides access to the database of archived entries. It brings up a window with two tabs: **INQUIRIES** and **LOGENTRY EXPLORER**. These offer two types of access methods (described in sections 8.2 and 8.3, respectively). Use either method to locate the archived entries of interest[1]. To view the search results, pull up a report container[2] (see section 8.7.1 *Pull Up a Report Container*) and drag the found entries into it. From there you can edit the contents of the container, and print reports, as described in section 8.7 *Creating, Editing and Printing Reports in CRL.*

8.2 Inquiries

The **INQUIRIES** feature allows you to search the database by constructing an inquiry (query) using a set of filters. After you configure your inquiry, you can execute it, save the results to a report container, save the inquiry for future use, restore it, and edit it.

1. If any entries get archived when the database is down, and if the database was not updated to reflect these new entries once it was restarted, then these entries will be retrievable only from the **LOGENTRY EXPLORER**, since it uses the filesystem. You will not be able to find them from **INQUIRIES**.
2. You *can* save the results to an input container, but the category/topic of the archived entry(ies) has to match that of the container. It's generally best to just use report containers which accept archived entries of any category/topic.

The search runs in the background, enabling you to continue your work in **CRL** while it's running. A progress bar displays the progress of the search. The cursor changes to show that a search is running, but note that if you click on another **CRL** internal window, the cursor changes back to the pointer. You can cancel a search at any time via the **CANCEL INQUIRY** button.

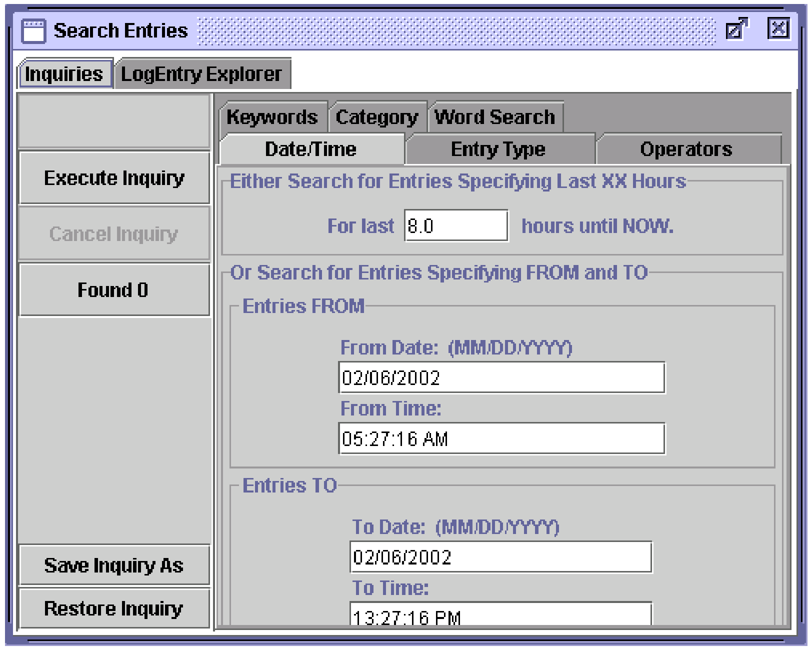
8.2.1 Create an Inquiry

You can construct inquiries to be simple or complex, depending on the number of filters you include. You can construct them to be general or specific, depending on the level of detail you specify for each filter. You can edit your search criteria, and add filters not previously used. Select each filter tab in turn on the right side of the **SEARCH ENTRIES** window, and enter your search criteria.

The *entry type*, *operator*, and *keyword* filters allow you to select or input multiple criteria, separated by a logical AND or OR. **CRL** requires that the criteria be separated by a logical operator. By necessity, the interface does not prohibit you from deleting an operator; but if you don't replace it with another, the system will return an error of the form "You have error in your SQL syntax near '<*filter type*> = <*criterion*>', e.g., ... near 'Entry Type = Ipen'.

Date/Time

The **DATE/TIME** filter allows you to specify a time window in one of two ways. You can specify a number of hours to look back from the present time (e.g., the last 8.0 hours, which is the default), or specify a start and end time. The time of entry creation is used in the search, rather than archive time. If you change the number of hours, the FROM/TO information will change accordingly. If you set either or both of the FROM/TO values, the number of hours in the top field will show 0.0, and become inactive.



An editing note: the fields are type-over. When you've completed this filter, optionally choose another to further restrict your search.

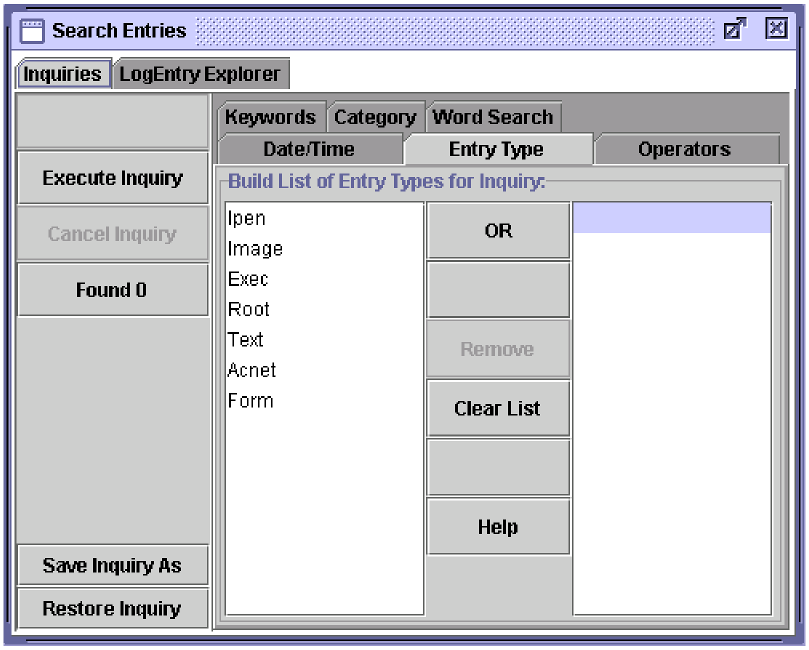
Entry Type

The **ENTRY TYPE** filter allows you to restrict the selection of entries by type. In the left-hand column is the list of possible entry types. You may choose as many as you like. By default an OR is automatically added between entry types so that entries of all selected types are eligible for retrieval.

To select an entry type:	click on a type in the left-hand list; it will appear on the right-hand side
To remove an entry type:	select from the right-hand list and click **REMOVE**

In the same manner, you can also remove any OR in order to create an AND between two adjacent selections. Having no operator between selections results in an error.

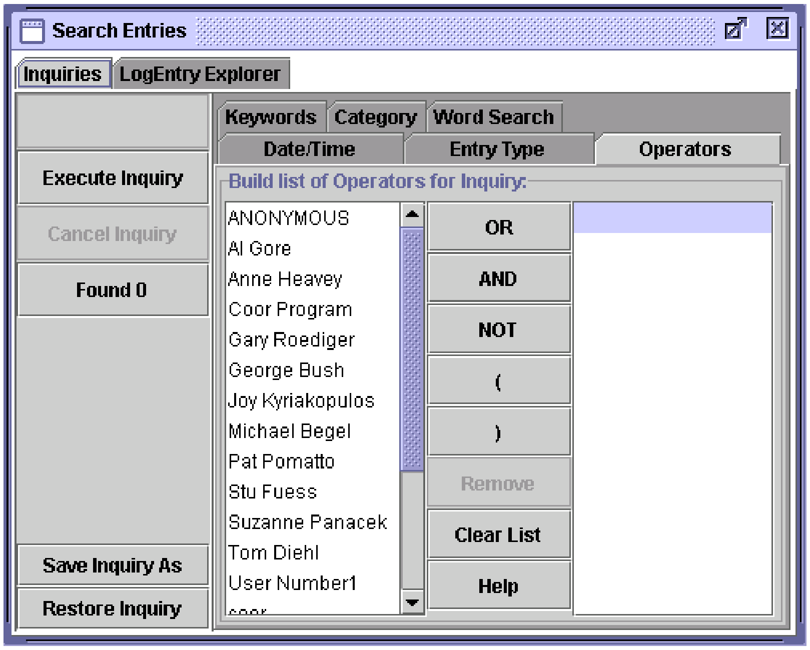
You cannot add/remove multiple entry types at a time.



When you've completed this filter, optionally choose another to further restrict your search.

Operators

The **OPERATOR** filter allows you to restrict the selection of entries by operator. To add an operator to the search list, click the operator's name in the left-hand list. By default, an OR is automatically added between selected operators so that all entries containing any of these operators are eligible for retrieval. To change an OR to AND or NOT, simply remove the OR then click the desired logical token. Use the parentheses for grouping, as needed. Operators and logical tokens are inserted before the selected item in the right-hand search list.

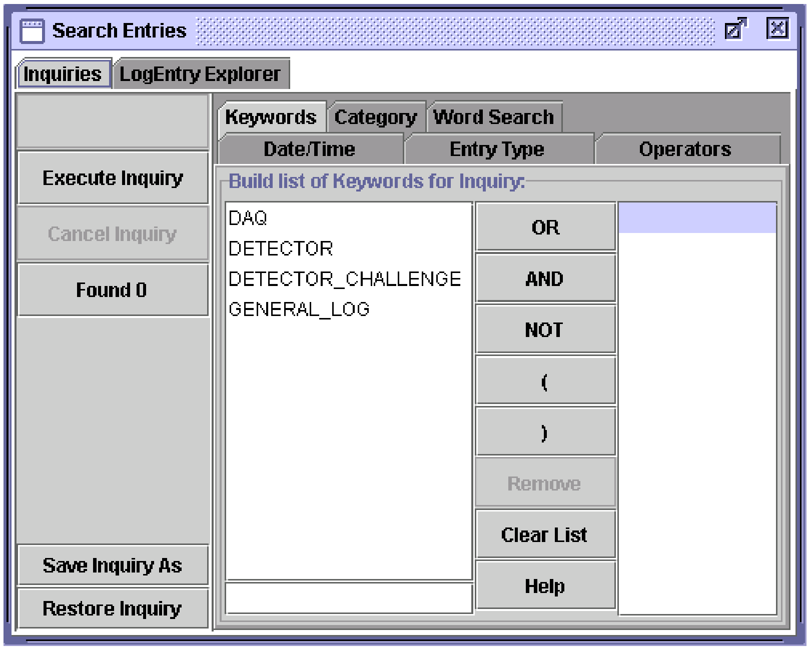


When you've completed this filter, optionally choose another to further restrict your search.

Keywords

The **KEYWORD** filter allows you to restrict the selection of entries by keyword. To add a keyword to the search list, click it in the left-hand list. By default, an OR is automatically added between selected keywords so that all entries containing any of these keywords are eligible for retrieval. To change an OR to AND or NOT, simply remove the OR then click the desired logical token. Use the parentheses for grouping, as needed. To insert a keyword or logical token in front of an item already in the right-hand search list, select that item before inserting the keyword or token.

You can also add a keyword that's not listed on the left. To do so, type the keyword at the bottom of the left-hand keyword list using UPPERCASE letters only, then press **ENTER** to move it to right-hand side.



When you've completed this filter, optionally choose another to further restrict your search.

Category

(not implemented as of V1_7_04)

Word/String Search

The word search filter allows you to retrieve entries based on words or strings occurring in them. Currently, searches can be run as follows:

String Search	select entries containing the entire search string entered
Word Search (Any)	select entries containing ANY of the words entered (separate words with a space)
Word Search (All)	select entries containing ALL of the words entered (separate words with a space)

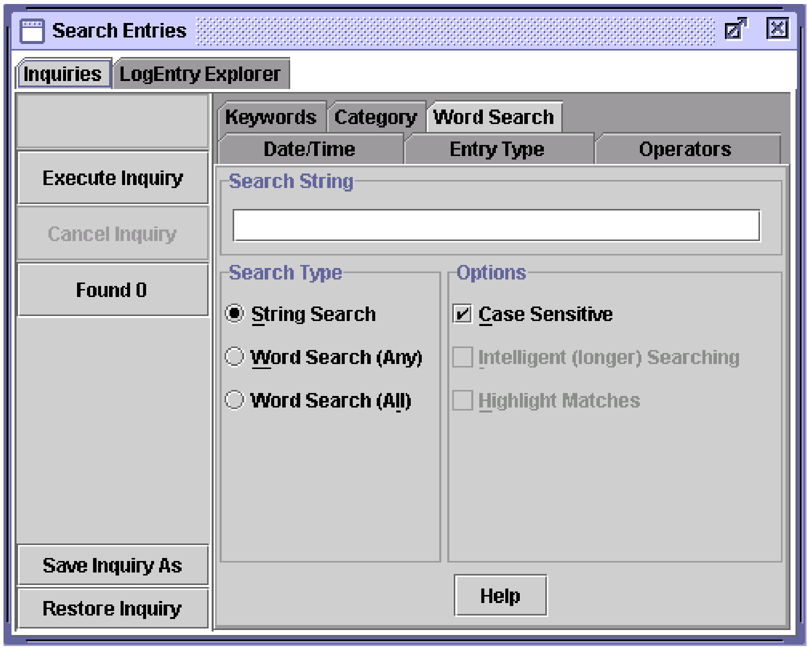
Only one Option is provided (as of V1_7_04):

- Case matching; if **CASE SENSITIVE** is left unchecked, case is ignored.

Planned for future releases:

- Option for the default (faster) search, or the intelligent (slower) search. The default search checks the XML rather than the entry itself (if the search string is anywhere in the XML code, the entry is matched). If you check the **INTELLIGENT (LONGER) SEARCHING** button, **CRL** instantiates each logentry object and provides accurate searching at the expense of search time.

- Option to highlight matched text when retrieved entries are displayed.



When you’ve completed this filter, optionally choose another to further restrict your search.

8.2.2 Execute an Inquiry and View Results

When your inquiry is configured and ready, click the **EXECUTE INQUIRY** button to run the search. As mentioned above, it runs in the background so that can continue to use **CRL** while it runs. A progress bar is displayed for longer inquiries. When execution has completed, the **FOUND <N>** button will indicate the number of entries found that match the inquiry parameters, e.g., **FOUND 7**. Drag the **FOUND <N>** button to a report container to view the results.

Note that you *can* drag the **FOUND <N>** button into an input container, but only the entries of the same category/topic as the container will be included; all other entries will be filtered out. A warning message appears when you do this.

☞ Cells within a table are scrollable when an archived entry is restored via a search in **CRL**.

8.2.3 Halt Execution of an Inquiry

If you wish to cancel the inquiry after you click the **EXECUTE INQUIRY** button but before the search completes, click the **CANCEL INQUIRY** button.

8.2.4 Save an Inquiry

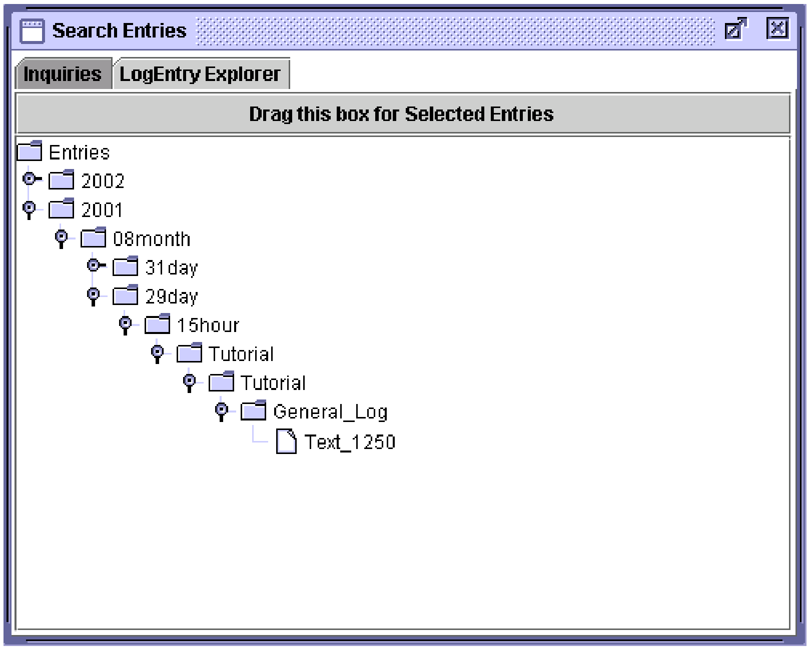
If you wish to use the same inquiry again at a later time (as is or edited), save your inquiry to a file. To do so, click the **SAVE INQUIRY AS** button. This brings up a standard window for choosing a directory and filename.

8.2.5 Restore an Inquiry

To restore a previously saved inquiry, click the **RESTORE INQUIRY** button, and type in or browse for the inquiry file. Run it as is, or edit it first. All the filter panels are made available so that you can add more criteria to your search if you like. You can save a restored inquiry (changed or unchanged) to the same or a different file.

8.3 LogEntry Explorer

The **LOGENTRY EXPLORER** feature allows you to access archived entries using the file system, down the tree by year, month, day and hour (reflecting time of entry creation, not archive time). Within a selected hour time window, you can confine your search to a particular data category and/or topic. Individual entries make up the final branches of the hierarchical tree; they are labelled and numbered.



You can select a folder (e.g., a month, an hour, a category) or an individual entry to drag into a report container in order to view its contents. You can select multiple items (including folders, entries or both) and drag them collectively[1] to the container. Note that you can drag the entries into an input container, but only the entries of the same category as the container will be included; all other entries will be filtered out.

☞ Cells within a table are scrollable when an archived entry is restored via a search in **CRL**.

8.4 Accessing Archived Entries on the Web

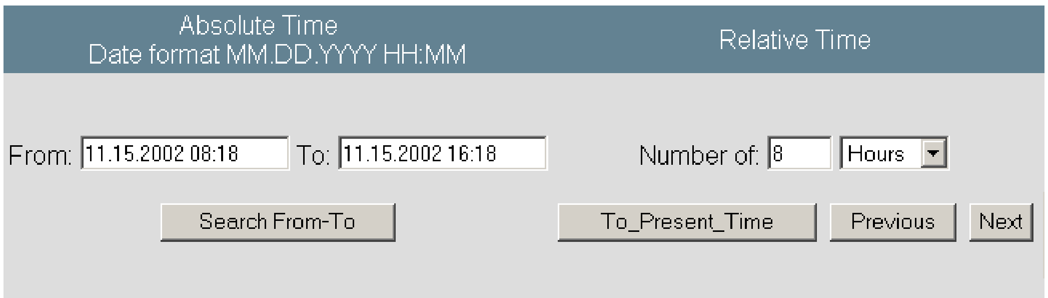
Each experiment can configure its own database inquiry web site from which experimenters anywhere in the world can search, view, annotate and print archived entries. Your experiment will need to provide you with the URL and any related documentation.

1. Note that the down motion of the mouse click selects/deselects. So after all items are selected, move the mouse outside the tree (or into the **DRAG THIS BOX FOR SELECTED ENTRIES**), press the Shift or Ctrl button as appropriate, and then begin dragging.

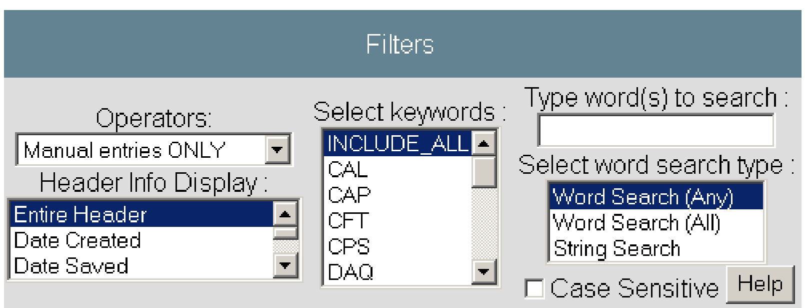
8.4.1 Define your own Search Criteria

The images shown here are for a sample web inquiry page. They illustrate the search features available. To make the pages readable here, we've split the page header onto two lines:

The left-hand portions, labelled *Absolute Time* and *Relative Time* allow you to specify a time frame in one of these two ways:

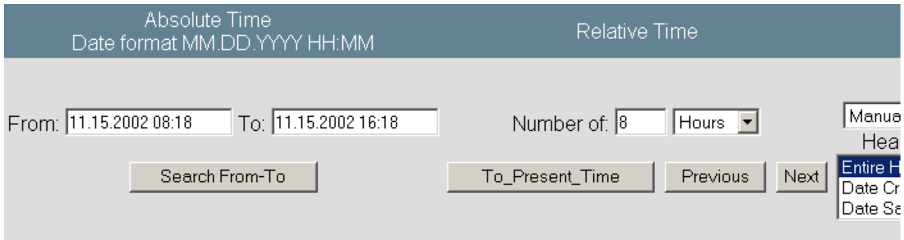


The right-hand portion, labelled *Filters*, allows you to specify one or more of several search criteria, and to select the header information you want to display:



Note that in *Type word(s) to search*, the operator "NOT" doesn't work. However, if under *Select keywords* you find a keyword labelled "NOT_<KEYWORD>", e.g., NOT_CAL, it can be used to eliminate entries with that keyword from the search (e.g., to eliminate entries with the keyword CAL attached)

The inquiry web page as a whole looks like this next image, showing an entry (the right-hand side of the header is cut off):



Annotate This Entry
Date Created: Friday, November 15, 2002 8:23:13 AM CST
Date Saved: Friday, November 15, 2002 10:28:06 AM CST
Category - Topic - sequence number: CAL/Log - CAL_Log - 150111
Operator(s): Wanda Prado
Keyword(s): :CAL:

Run 168133 stopped with 469703 events
no hot cells observed

8.4.2 Run a One-Click Search

Your experiment may also have predefined inquiries set up which you can run just by clicking a link on a web page. For instance, here is a portion of a web page the D0 experiment has set up. It contains several predefined inquiries for a number of subjects:

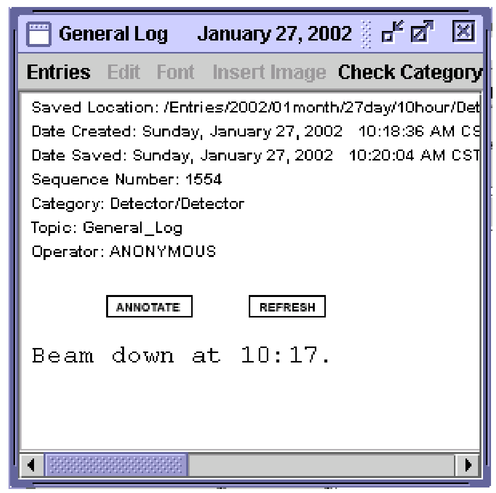
- **All Logbook Entries** *Except COOR Entries & SMT Checklists* 4 hours 8 hours 12 hours 24 hours 36 hours 48 hours 60 hours 72 hours 96 hours 1 Week 2 Weeks
- **CAP + DAQ + MUO Log Entries** 12 hours 24 hours 36 hours 48 hours 72 hours 96 hours 1 Week 2 Weeks 30 Days 60 days 90 days
- **DAQ Shifter Log Entries** 12 hours 24 hours 36 hours 48 hours 60 hours 72 hours 96 hours 1 Week 2 Weeks 30 Days All
- **Captain Log Entries** 12 hours 24 hours 36 hours 48 hours 60 hours 72 hours 96 hours 1 Week 2 Weeks
- **Global Monitor Log Entries** 12 hours 24 hours 36 hours 48 hours 72 hours 1 Week 2 Weeks 30 days
- **Calorimeter Log Entries** 24 hours 48 hours 72 hours 1 week 2 weeks 30 days 60 days
- **SMT Log Entries** 24 hours 72 hours 1 week 2 weeks 30 days 60 days
- **CFT Log Entries** 24 hours 72 hours 1 week 2 weeks 30 days 60 days
- **Muon Log Entries** 24 hours 72 hours 1 week 2 weeks 30 days 60 days

Selecting the "4 hours" link for **ALL LOGBOOK ENTRIES**, the following results page comes up (the entry content has been purposely obscured). It's set up so that users can run predefined inquiries (for the same subject) from this page, too.

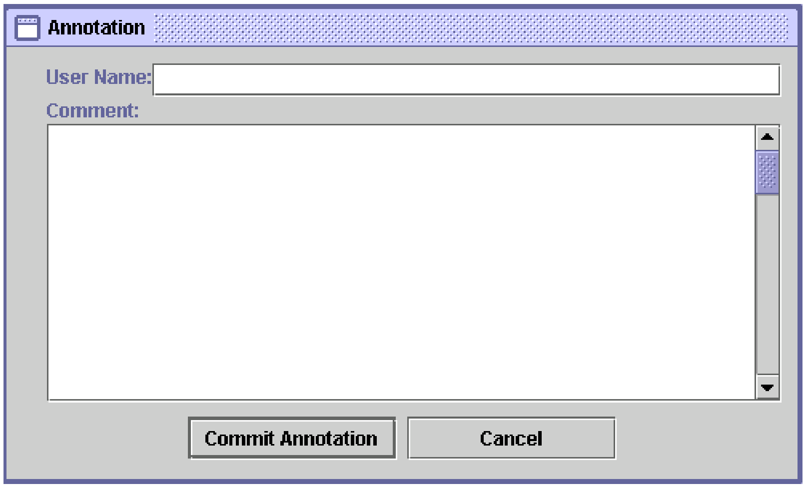
8.5 Annotating Archived Entries

8.5.1 Annotate Within CRL Application

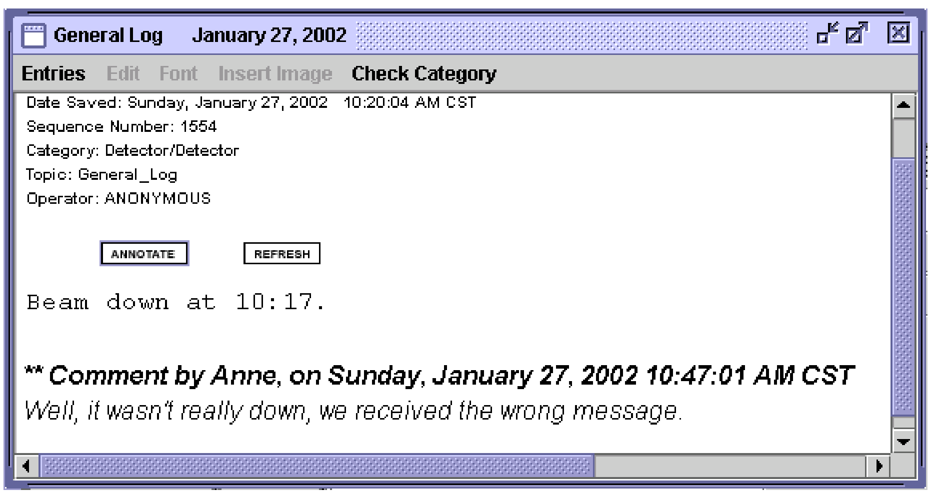
Annotate an Entry

Once a logbook entry has been archived, the entry gets “set in stone”. The only change you can make is to add an annotation. When an entry is archived, the **ANNOTATE** and **REFRESH** buttons appear.	

To annotate the entry, click the **ANNOTATE** button. A window pops up in which you enter your name and type in your comments. When you’re ready, click on the **COMMIT ANNOTATION** or **CANCEL** button, as desired. No checks are made by **CRL** on the contents of the fields before committing the annotation.



Annotations that you make this way are automatically displayed in the entry. Annotations made at other terminals in the control room or via the web interface do not appear until you press the **REFRESH** button; see below. An annotated entry displays the added comments in this format:



Refresh Entry to see Annotations

To see annotations made on an entry at other terminals running **CRL** in the control room or via the web interface, click the **REFRESH** button. This refreshes only the entry in question, it does not refresh all the entries in the container.

8.5.2 Annotate from the Web Interface

Annotate an Entry

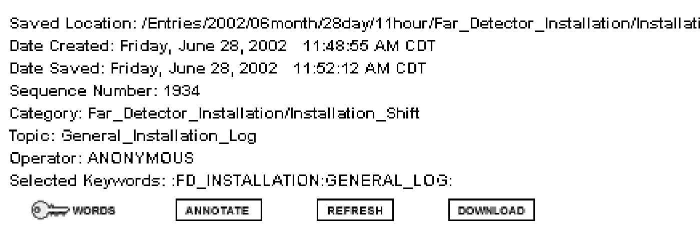
Each entry as it appears on your experiment's web interface should have a link for "Annotate this entry" (see the image at the end of section 8.4.1 *Define your own Search Criteria*). The appearance and location of this link may vary from one implementation to another. Click "Annotate this entry" to add a comment, then commit (or cancel) the comment. Annotations lacking either a username or content do not get saved.

Refresh Entry to see Annotations

When your annotation is complete, click your browser's "Refresh" or "Reload" button. This will update all the entries displayed on the web page (in contrast to the single-entry refresh within the **CRL** application).

8.6 Downloading/Viewing Attached Files

For an output file from an external application that was attached to an entry (see section 3.3.4 *Output Files from External Applications*), the file content does not appear in the container window. After archiving the entry, you can download or view the file. Along with the **ANNOTATE** and **REFRESH** buttons, a **DOWNLOAD** button appears with the entry:



If you click **DOWNLOAD**, you get the following prompt:

If you click **DOWNLOAD** again, a standard dialog box appears allowing you to choose a directory and filename for the download file. If you click **VIEW**, then assuming you have the appropriate viewing software installed on your system, you can view the file.

☞ Note that an attached file works differently from a file inserted into an entry from the **INSERT > TEXT > FROM A FILE** menu option. When inserted, the file does appear in the container window.

8.7 Creating, Editing and Printing Reports in CRL

A *report* is simply a collection of one or more archived entries in a report container.

8.7.1 Pull Up a Report Container

The first step for creating a report is to select an appropriate reporting category and topic, and open the corresponding report container. To do so:

1) Select the page tab at the top of the **CRL** window to display the page that contains the report category you want. This may be a report page or an entry-input page, depending on the configuration.

2) From the horizontal row of menu headings along the top of the selected page, choose a report menu.

3) From its pull-down menu (or cascading pull-down menus), choose the appropriate reporting category and topic. A report container pops up, labelled with the chosen topic.

8.7.2 Report Container Menu Options

The menu options and their functions include:

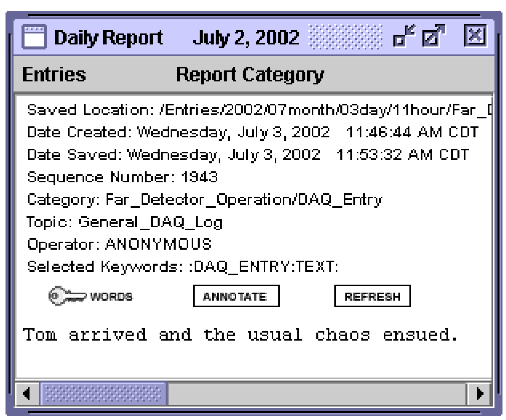
ENTRIES Select/deselect all entries, remove, print entries, send entries via email, or (de)attach them to/from thread. (Bring up the **ENTRIES** menu either by clicking on the **ENTRIES** menu option, or by right-clicking anywhere in the container window.)

☞ The **REMOVE** options remove the entries from the report container; they are no longer displayed in the container window, but they remain unchanged in the database.

REPORT CATEGORY Display hierarchical categorization of container (informational only).

8.7.3 Insert Entries into Report Container

Select entries/folders as described in sections 8.2 *Inquiries* and 8.3 *LogEntry Explorer*, and drag-and-drop them into the report container until you have collected all the entries you want. The entries will order themselves chronologically.



If you insert the entries into a different type of container, e.g., an input container, only the entries of the same category/topic as the target input container will show up; all others will be filtered out. So, depending upon your search criteria, you may see fewer entries than were found by the search. A warning message appears when you do this.

☞ Cells within a table are scrollable when an archived entry is restored via a search in **CRL**.

8.7.4 Export Binary Files from Report Container

From a report container you can export (download) any binary file that has been included in an archived logbook entry. For entries that include a binary file, the **EXPORT FILE** button appears underneath the header. To export a binary file to the file system, click the **EXPORT FILE** button. A window pops up that allows you to type in or browse for a destination directory.

8.7.5 Select Report Entries in Container

You can remove or add entries to a report container in order to produce a printed report containing the exact set of entries you want. **CRL** allows you to select one or more individual entries or all entries in the container. Selected entry headers have a gray background, nonselected ones are white. See section 4.2.1 *Select/Deselect Single or Multiple Entries*.

8.7.6 Remove Entries from Report Container

There are two menu options for removing logbook entries from a report container: **REMOVE SELECTED ENTRIES** and **REMOVE ALL ENTRIES IN THIS TOPIC**. The **REMOVE** function in a report container removes the archived entry from the container *without* any confirmation query.

To remove a single entry or a set of hand-picked entries, first select each entry. Then choose **REMOVE SELECTED ENTRIES** from the **ENTRIES** menu.

To remove all entries from the report container, simply choose **REMOVE ALL ENTRIES IN THIS TOPIC** from the **ENTRIES** menu.

8.7.7 Send Entries to an Email Recipient

There are two menu options for sending logbook entries: **SEND SELECTED ENTRIES TO MAIL RECIPIENT** and **SEND ALL ENTRIES TO MAIL RECIPIENT**. These work similarly to the **SEND** options for input container menus, described in section 4.7 *Sending Entries via Email*, except that the shortcut keys are not implemented for thread containers.

8.7.8 Print Report

There are two menu options for printing a report from the report container: **PRINT SELECTED ENTRIES** and **PRINT ALL ENTRIES IN THIS TOPIC** (the latter can also be performed using **CTRL-P**.) When you use one of these options, a standard **PRINT** window for your operating system pops up in which you set the destination printer and printer options. The printed report will contain the entries in the container corresponding to the entries you selected and the menu option you selected.

8.7.9 Add Entries to a Thread

There are two menu options for adding entries to a thread from a report container: **ADD SELECTED ENTRIES TO A THREAD** and **DELETE SELECTED ENTRIES FROM A THREAD**. These work in the same way as they do for input and thread containers. See section 7.3 *Adding Entries to a Thread* for instructions.

Chapter 9: Programmer's Guide to the Process Logger

This chapter describes the Process Logger and provides some guidance for programmers wishing to configure their external programs to create **CRL** entries.

9.1 Introduction

The Process Logger (Plog) provides a way to create and store **CRL** entries from programs external to **CRL**. This is particularly useful for programs that monitor alarms or devices on the experiment. Plog entries are stored and viewed in the same way as entries inserted from within the **CRL** application.

Plog is run as a standalone daemon process that monitors specified TCP ports for input, interprets the input as **CRL** entries, and creates and logs the entries. Information on starting the Plog daemon is provided in section 11.9 *Starting the Process Logger Daemon*.

There can be multiple back-to-back messages on a single open TCP connection and many concurrent TCP connections on any TCP port. Each experiment must assign and make known the TCP port number(s) for remote program connections.

9.2 Guidelines for Programmers

The Process Logger communication is full duplex. Your program needs to send messages (the entries) to Plog and to read return messages from it. You must write your **CRL** entries to one of the TCP ports (sockets) on the Plog host, as assigned by your **CRL** administrator.

Input for an entry must be furnished in the form of an XML message that identifies the entry's various header elements and body, as shown in section 9.2.1 *Entry Message Format*. The header elements include operator name, category, topic, and keyword(s), all of which are optional but recommended. These element types are described in Chapter 1: *Overview*. Currently, there is no validation of the values of these elements against values already defined for a particular **CRL** installation, so you can choose unique text strings for these items, or not. The search facilities in **CRL** will allow users to search on the operator, category, topic, and keyword(s) for your program's entries, regardless. Make these names descriptive!

☞ **Note that if the header element values you choose don't match values already defined, then you need to communicate to your fellow experimenters exactly what values to search on in order to retrieve these entries!**

Currently, only text or plaintext messages may be included in Plog entries; no binary data is accepted[1]. Text and plaintext are compared and contrasted in section 3.3.1 *Text*.

9.2.1 Entry Message Format

The format of the messages Plog receives from your program must be as shown below.

Notes:

- Use upper case for the element tags.[2]
- The `MESSAGE TYPE` (first line) must be set to `TEXT` or `PLAINTEXT`.
- The element tags `<MESSAGE>... </MESSAGE>` and `<TEXT>...</TEXT>` are required; all other element tags are optional.
- The logged text within `<TEXT>...</TEXT>` must either be contained within a `<![CDATA[...]]>` construction or it must conform to valid XML standards (e.g., `<P>` must be used as `<P/>`, `<BR>` must be `<BR/>`, and so on, and all tags with attributes must have the attribute value enclosed in double quotes, e.g., `<FONT size="10">...</FONT>`).
- The logged text within `<TEXT><![CDATA[...]]></TEXT>` can contain[3]:
  - · newlines
  - · carriage returns
  - · HTML tags (If the message type is plaintext, your browser should treat HTML tags properly, but **CRL** will treat the tags as text.)

---

1. In the future, binary data could be added by using base64 encoding and creating an XML tag for the encoded data.
2. <MESSAGE> and </MESSAGE> are the only element tags that are required to be upper case.
3. The CDATA construction is not strictly necessary, however it ensures that the text will be interpreted as a character string and will not be parsed.

Format

```
<MESSAGE TYPE="plaintext">
      <OPERATOR>Name of program or responsible person
      </OPERATOR>
      <CATEGORY>Category/subcategory/sub-subcategory/...
      </CATEGORY>
      <TOPIC>Topic
      </TOPIC>
      <KEYWORD>Keyword1
      </KEYWORD>
      <KEYWORD>Keyword2
      </KEYWORD>
      <TEXT><![CDATA[The logged text goes here.]]>
      </TEXT>
</MESSAGE>
```

9.2.2 Return Messages from Plog

Every time Plog receives an entry, it returns a message to the sending program. The return message is one of the following three:

`<SUCCESS/>`	successful entry
`<FAIL/>`	entry not saved, but no syntax error detected; may succeed if tried in future (e.g., occurs if database application is not currently running or filesystem not available)
`<ERROR/>`	message had a syntax error and will never result in a saved entry

9.2.3 Sample Java Program Excerpt

You should configure your program to run input and output threads, as illustrated in this annotated **Java** test program excerpt (text enclosed in brackets, e.g., `<text>`, indicates replacement by context-sensitive data):

```
// Start the program:
public static void main (String[] args) {
  try {

// Define target host and TCP port (socket):
     Socket s = new Socket ("<IP_address_of_Plog_host>",<TCP_port_number>);

// Open "write" target file (set to PrintWriter here):
     PrintWriter pw = new PrintWriter (s.getOutputStream(), true);

// Start thread to read return messages from Plog, BEFORE starting write thread:
     InputStreamReader isr = new InputStreamReader ( s.getInputStream () );
```

```
// Start thread to write messages to PrintWriter; this test program accepts
// input from keyboard, writes to pw, then flushes pw. Test program does
// not enforce entry message format -- yours must!
     new ClientTester ().start();
// Thread to send to Plog
     while ( true ) {
        char c = (char)System.in.read();
        pw.write(c);
        pw.flush();
     }

// Thread to read return messages from Plog
public void run () {
   char[] cbuffer = new char[20]
   while ( true ) {
      try {
         System.out.println("ISSUE READ" );
         int count = isr.read(cbuffer);

         System.out.println("READ: "+new String (cbuffer,0,count));
      }  catch ( Exception e) {}
   }
...
}
```

Part II CRL Administrator's Guide

Chapter 10: *CRL Installation*

This chapter lists the components needed on your machine in order to run the **CRL** program and **CRL** Web Access. It also describes how to install **CRL**.

Chapter 11: *Configuring and Launching CRL*

This chapter describes the configuration tasks that are necessary for running the application, for customizing the desktop environment to your experiment, and for enabling particular features of **CRL**. It also discusses invoking the application.

Chapter 12: *Managing User Information*

This chapter describes how to add, activate and deactivate users, as well as how to change information about them.

Chapter 13: *Configuring Web Access to Logged CRL Entries*

This chapter lists tasks that are necessary for creating a web site for your **CRL** entries, in order that collaboration members anywhere in the world can search, view, and annotate your experiment's logbook entries.

Chapter 10: CRL Installation

This chapter lists the system requirements and the software components needed on your machine in order to install and run the **CRL** program and **CRL** Web Access. We also describe how to install **CRL**.

The **CRL** product can be installed on Windows NT, 98, and 2000 using InstallAnywhere (this is still in progress as of Nov 2002). **CRL** is also available in tar format from the KITS database for Linux[1] (product name **crl**), and is installed in the Fermilab AFS area for both Linux and Windows under `/afs/fnal.gov/ups/crl/`. At the time of this writing (again, Nov 2002), the current version of **CRL** for Windows and Linux is 1_7_04.

10.1 Space Requirements

We recommend that you have at least 128 Mb of RAM on the machine (256 is better), and a "good" graphics card (lots of memory and high-end graphics capability). For Linux installations we recommend the **KDE** window manager over **FVWM**, although either will work.

You need about 42 Mb of space to install **CRL** v1_7_04 on Windows, or 64 Mb to install version 1_7_04 on Linux (the space required may go up or down as versions change). **Java Virtual Machine** (**JVM** v1.4.1) is bundled in the installation. You may need additional space to configure the application, since it loads dynamically, and extra objects take extra space.

For the logbook entry database, you need to install a relational database management system (RDBMS). The database can reside on a different computer, if desired. Requirements may vary; a **MySQL** installation initially requires about 25 Mb. The database itself grows as you add entry pointers. Each entry pointer adds about 0.5 kb. The entries themselves require varying amounts of space; figure on about 1 to 3 kb for a text entry.

1. Solaris can be made available upon request.

10.2 Software Requirements

10.2.1 Requirements for Installing CRL

This will be changing soon (as of June 2002); InstallShield will no longer be used.

Windows	the InstallShield executable for installing **CRL** (comes bundled with the product download from `http://www-cpd.fnal.gov/CRL/downloads.htm`; see section 10.5 *Downloading InstallShield on Windows*)
Linux (via **UPD**)	**UPS** and **UPD** must be installed on target machine
Linux (tar file)	none

10.2.2 Requirements for Running CRL

- Java 1.4.1 (comes bundled with the **CRL**; installed in `_jvm` subdirectory) Note that this program will not interfere or conflict with any pre-existing installation of Java on your system.
- MySQL database or other RDBMS (MySQL is available for download at `http://www.mysql.com/downloads/index.html` and is also in KITS.)

10.2.3 Requirements for Running CRL Web Access

All these are available for download from `http://www-cpd.fnal.gov/CRL/downloads.htm`.

- a web server (e.g., Apache Web Server, available for download from `http://httpd.apache.org`)
- Tomcat (download latest version from `http://jakarta.apache.org`)
- mod_jk Apache-to-Tomcat plugin[1]

1. mod_jk provides support for Dynamic Shared Object (DSO); it tells Apache to send JSPs to Tomcat.

Installation instructions for the web server components can be found at `http://www-cpd.fnal.gov/CRL/Apache.html`, `http://www-cpd.fnal.gov/CRL/Tomcat.html`, and `http://www-cpd.fnal.gov/CRL/Tomcat_Apache_Plugin.html`, respectively.

10.3 CRL Installation Components

CRL components installed by InstallShield or tar/**UPD** include:

- **Java Virtual Machine** (**JVM**) version 1.4.1[1]
- **CRL** jar file (the application)
- `CRL_Windows.bat` on Windows, or `crl` script on Linux for running **CRL**
- DTD and XML CRL configuration files
- DTD and XML inquiry configuration files
- DTD file for forms and XML forms definition files
- images for toolbars, sample histograms, sample form entry config files
- JDBC driver for MySQL[2]
- Xerces jar file (XML parser)
- Email-related jar files for sending entries to email recipients
- Communications jar files and other miscellaneous files needed in order to use the Ipen Drawing pad
- Sample html and jsp (Java Server Pages) files for web access to logged entries
- uninstall directory (called `uninst`) containing the file `uninstall.jar`, which when run, uninstalls **CRL**
- installation documentation (README file) and release notes

1. JVM includes comm jar files and other miscellaneous files needed to run the Ipen drawing pad.
2. JDBC is an interface that allows Java to use MySQL.

10.4 Upgrade Instructions (to V1_7_04 or Higher)

In order to upgrade to **CRL** V1_7_04 or higher on Linux or Windows, copy over the new **CRL** application jar file to replace the old one. Copy over the new properties file (or look through Chapter 14: *CRL's Java Properties*) and add the new properties to your existing file.

If you don't do a complete reinstall of **CRL**, you'll need to upgrade your JVM (JRE or JDK) to version 1.4.1 on your own. Once you do that, you can make the necessary changes to JVM 1.4.1 so that it works with the latest **CRL**. Follow the instructions in section 11.2 *Configuring CRL to Use a Preinstalled JVM (Optional)*. Also see `http://www-cpd.fnal.gov/CRL/downloads.htm`.

10.5 Downloading InstallShield on Windows

This will be changing soon (as of June 2002); InstallShield will no longer be used.

The **CRL** product is packaged for download in an InstallShield executable for Windows. You need to log on to a Windows account with local administrator privileges.

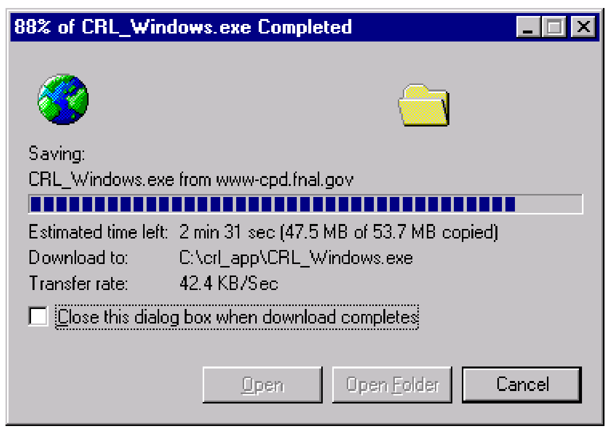
Before you download the executable and install **CRL**, you can create a directory for the application to reside in, or you can let the InstallShield wizard create one during the installation process.

On your web browser, navigate to `http://www-cpd.fnal.gov/CRL/`, and select "Downloads" to arrive at the *CRL - Downloads* page.

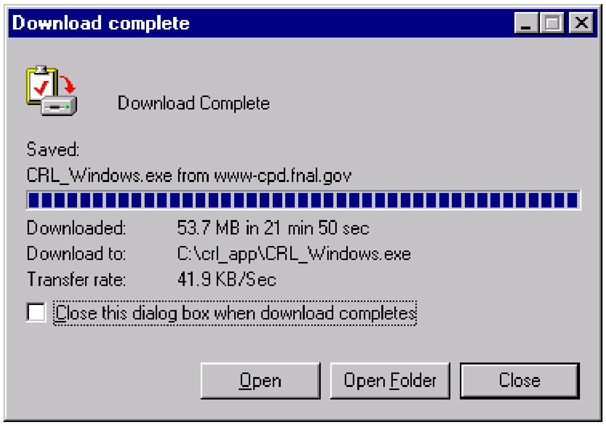
Click on *Windows NT/2000*. The **FILE DOWNLOAD** window appears:[1]

1. The download and installation documented here was performed on a Windows NT machine, and **CRL** version 1.5.01 was installed.

`Save this program to disk` is selected by default. Leave it as is, and click **OK**. A status screen appears for the duration of the download:



When done, the screen changes to indicate that the download of the InstallShield executable for **CRL** is complete:



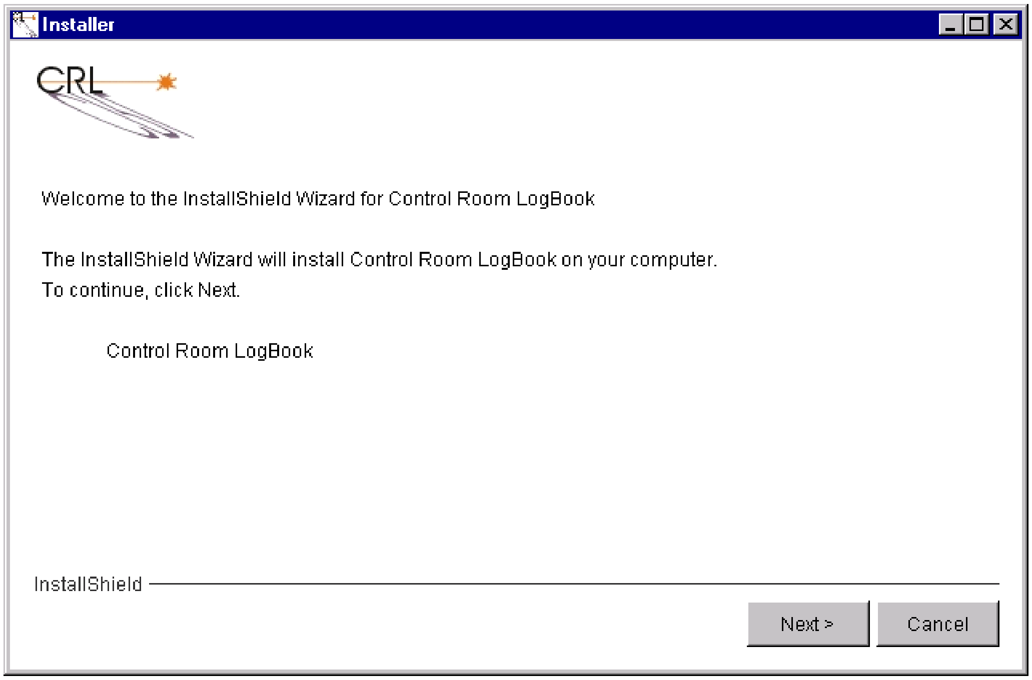
To install **CRL** right away, click **OPEN**, and proceed with the instructions in section 10.6 *Installing CRL via InstallShield on Windows*.

10.6 Installing CRL via InstallShield on Windows

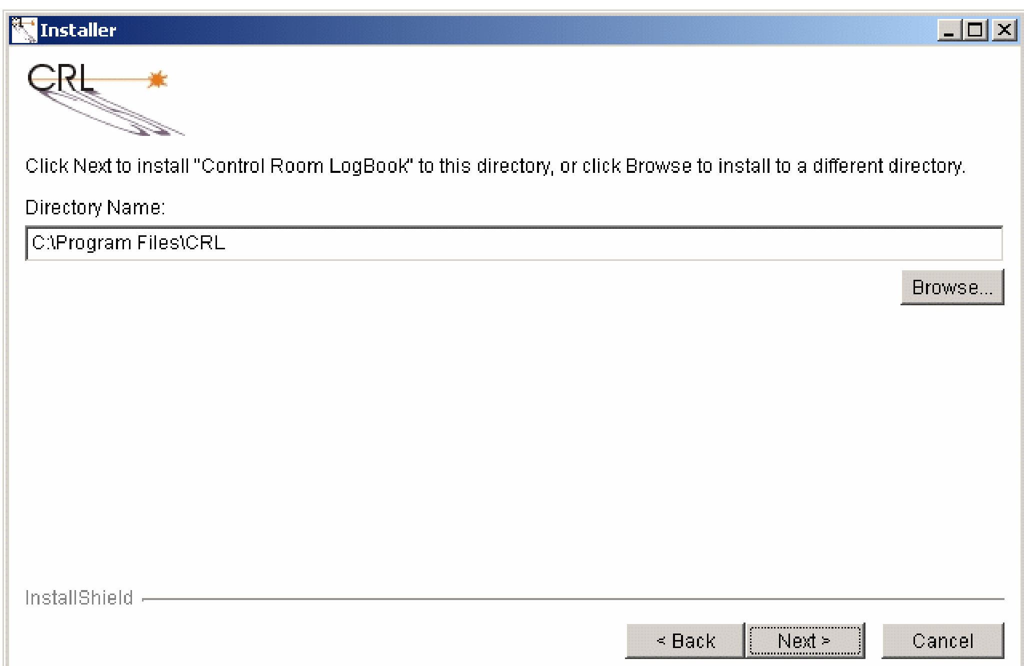
This will be changing soon (as of June 2002); InstallShield will no longer be used.

To install **CRL**, run the executable downloaded in section 10.5 *Downloading InstallShield on Windows*. If you just downloaded the executable and the **DOWNLOAD COMPLETE** window is still visible, click **OPEN**. Otherwise, navigate to the downloaded executable, and run it.

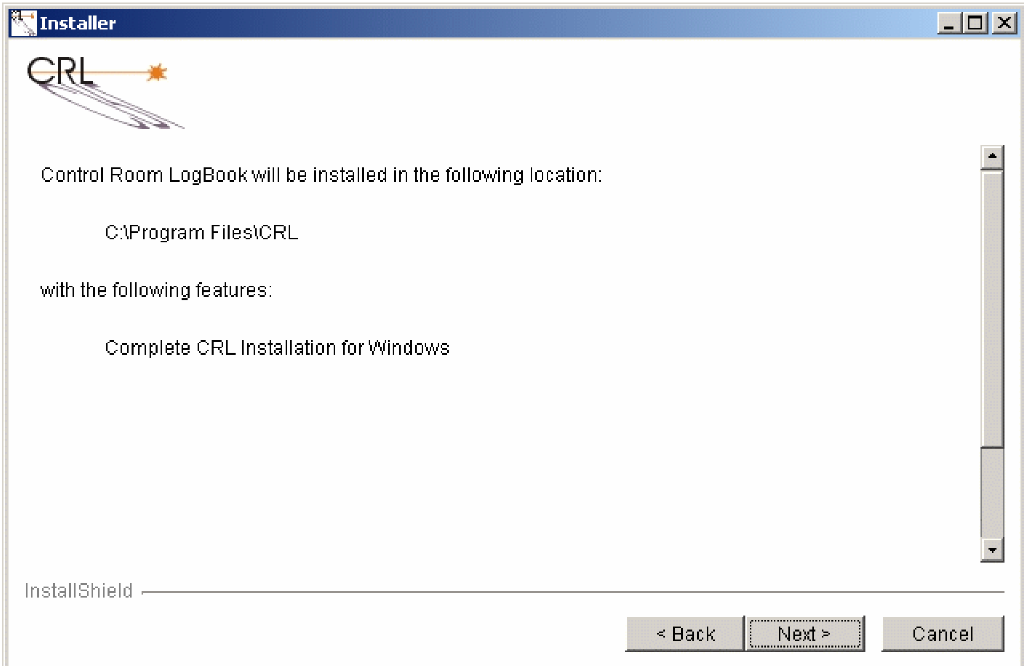
The InstallShield Wizard uses a graphical interface for the **CRL** install. On the initial screen that appears first, just click **NEXT >**:



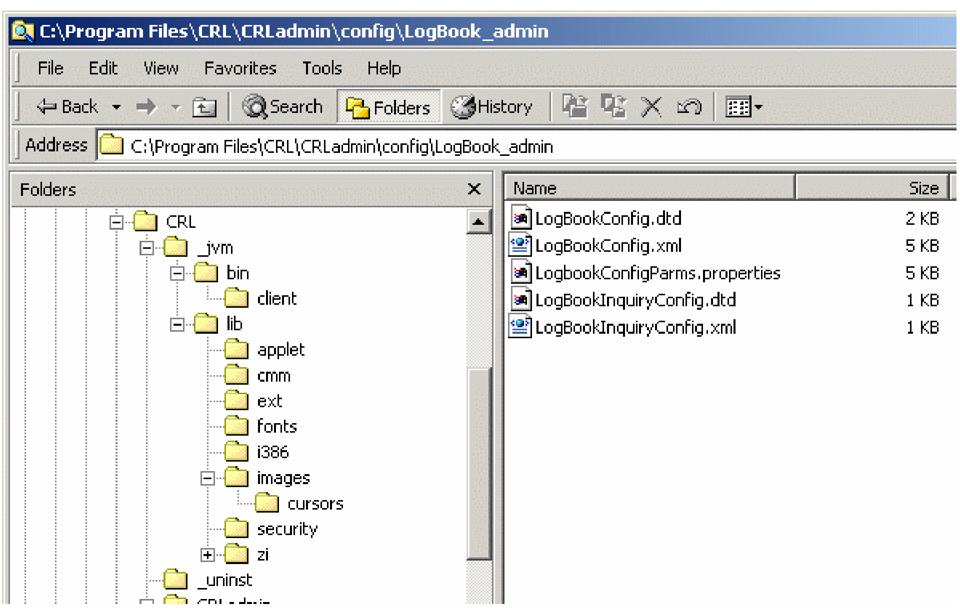
You will be asked to choose a directory in which to install **CRL**. You can type in or browse for the directory. If the directory you give doesn't exist, the wizard will create one. The path to the directory must already exist. After entering the information, click **NEXT >**.



Next the screen displays the information you have selected. If it's correct, click **NEXT >**.



A status window will appear for the duration of the installation. After the installation is complete, verify the directory contents (shown for Windows):



Before you run the program, you must edit the `LogbookConfigParms.properties` file so that it points to your database. See section 11.1 *Editing the Properties File* and Chapter 14: *CRL's Java Properties* for details.

10.7 Installing CRL on Linux

Not yet updated for CRL v1_7_04.

☞ For installing on Linux, you need to log in as *root* only if you plan to install in an area requiring *root* privileges, e.g., `/opt`, `/usr/local`, etc. Otherwise you can perform the installation under another account. Create a target directory for **CRL** before installing it.

10.7.1 Install from Tar File

If you do not have a **UPS** product environment on your Linux machine, you can download the latest **crl** product tar file from the **CRL** web site (`http://www-cpd.fnal.gov/CRL/`), and install by hand, according to the directions provided in the INSTALL_NOTE that comes with the product.

10.7.2 Install using UPD

If you are running UPS on your Linux machine, install the **crl** v1_7_04 product from KITS via UPD. Follow the instructions in the README file (also on the web at `http://www-cpd.fnal.gov/CRL/release_1_7_04/Readme.txt`).

10.8 Preparing to Use CRL in Fermilab's AFS Products Area

10.8.1 Instructions

CRL is installed under `/afs/fnal.gov/ups/crl/` for Linux and Windows, and may be used from there with appropriate setup on your local machine. Instructions are provided at `http://www-cpd.fnal.gov/CRL/afs.html`. In addition, README files are provided with the product installed in the AFS product area.

Here is a checklist of the things to do:

1) Make sure AFS is installed on your local machine.

2) Create a MySQL database, if necessary (see `http://www-cpd.fnal.gov/CRL/LinuxInstallExample.`

htm).

3) Copy the following files from the **CRL** product area in AFS space to your local machine (we recommend creating a similar directory hierarchy):

   · Copy the **CRL** invocation script (afs/fnal.gov/ups/crl/v<x_y>/Linux/crl for Linux; afs/fnal.gov/ups/crl/v<x_y>/Windows/CRLadmin/bin/crlWin32.bat for Windows) to a local directory with read-execute permissions.

   · Copy all files under the LogBook_admin directory to a local directory that's directly under your local "top" directory, and that's write-protected.

   · Copy any desired files under the images directory (includes images not in the jar file) to a local directory that's write-protected.

   · Copy any desired .winq files under the CRLinquiries directory to a local directory that's write-protected.

   · Copy any desired xml files under the CRLforms directory to a local directory that's write-protected.

   · Copy any desired files from the CRLfiles, CRLmaillists, CRLannotations, and CRLexportedFiles directories; at least create analogous local directories to house appropriate files.

4) Edit the **CRL** invocation script (see notes below for Linux and Windows specifics).

5) Edit your local copy of the properties file (/local/path/to/crl_top_directory/LogBook_admin/LogbookConfigParms.properties) so that it works with your database, etc.

6) Edit LogBookConfig.xml to use images in the local images directory, as necessary (/local/path/to/crl_top_directory/LogBook_admin/LogBookConfig.xml).

7) Complete other customization as desired.

10.8.2 Note for Linux

There is no icon supplied for start-up of the application. You will need to edit the crl script in order to specify the database and a local "top" directory (the directory directly above the LogBook_admin directory, which contains the properties file and configuration files). The script uses environment variables.

☞ Communicate to your users where you've put the local **CRL** invocation script and what you've called it!

10.8.3 Notes for Windows

An icon is supplied for start-up of the application.

The `crlWin32.bat` script is set by default to look for the "top" directory (the one above `LogBook_admin`) in the same directory as the script itself is located. If your configuration is different, edit the script to set the location of the "top" directory. Also in the script, set the drive letter that points to the **CRL** executable.

10.8.4 Linux and Windows Combined Setup

To set up a configuration that can be used from Linux or Windows, set up areas for each OS separately, but maintain only one copy of all unique files. Use symbolic links where necessary to avoid duplicate files. Note that you must maintain separate copies of the properties file because paths are written differently in the two operating systems.

10.9 Uninstalling CRL

10.9.1 Uninstall on Windows

Navigate to the top directory where you installed **CRL** on your system, and remove all files.

10.9.2 Uninstall on Linux

UPD Installs

For UPD installs, use the **`ups undeclare -y`** command to remove the product (see section 11.5.1 of the UPS/UPD manual at `http://www.fnal.gov/docs/products/ups/ReferenceManual/html/maintdb.html#16198`). E.g.,:

```
% ups undeclare [-f <flavor>] -y crl v<x_y>
```

Tar File Installs

For **CRL** tar file installs, just remove all the **CRL** files, e.g.,:

```
% rm -r /path/to/crl/version
```

Chapter 11: Configuring and Launching CRL

This chapter describes the configuration tasks that are necessary for running the application, for customizing the desktop environment to your experiment, and for enabling particular features of **CRL**. It also discusses invoking the application.

11.1 Editing the Properties File

11.1.1 The Bare Minimum

Before you can even invoke an installed instance of **CRL**, you must set a crucial subset of the parameters in the properties file. The properties file, `LogbookConfigParms.properties`, is stored in the `LogBook_admin` subdirectory[1].

In this section we just list the properties you need to edit. Look up each property as needed in Chapter 14: *CRL's Java Properties* to get information about setting its value.

Before editing the properties file, you must have information about your directory structure and your database. You're welcome to set all the parameters at this point, but here's the bare minimum that MUST be set:

```
Logbook.file_location.entry_directory
Logbook.database.enabled
Logbook.database.vendor
Logbook.database.version
Logbook.database.driver
```

If the connection URL (location of the database) is not of the form <protocol>//<machine_name>/<database>?user=<username>&password=<password>, then set the parameter:

```
Logbook.database.connection_url
```

1. In the **CRL** 1_7_04 release for Windows, the default **CRL** top level directory on Windows is `CRL\CRLadmin\config`. The `LogBook_admin` subdirectory should be under this.

If you set the above parameter, ignore the following five. Else, ignore above parameter and instead set the following parameters:

```
Logbook.database.protocol
Logbook.database.server
Logbook.database.dbms_name
Logbook.database.username
Logbook.database.password
```

Once you've made the initial edits to the properties file, launch the **CRL** application (see section 11.3 *Launching the CRL Application*) to see what it looks like and how it works. It comes with a sample configuration. Looking it over and trying things out will help you visualize and plan the desktop configuration for your experiment. Then go to section 11.4 *Configuring Your Desktop*.

11.1.2 Further Edits

After you become somewhat familiar with the desktop and operation of **CRL**, come back to the properties file and set additional parameters as you are ready to enable or change various features. Refer to Chapter 14: *CRL's Java Properties* for help on setting values for these parameters.

11.2 Configuring CRL to Use a Preinstalled JVM (Optional)

The JVM is included as part of the **CRL** release. We have extended the functionality of the JVM we ship in order to enable the IPen® functionality in the **CRL**. You can use another installation of the JVM (either JDK or JRE[1], v1.4.1 or higher), but to have IPen® functionality, you need to copy several files from the shipped JVM to the one you plan to use. You also need to modify the **CRL** invocation script so that it runs your JVM.

11.2.1 Files to Copy

All the files you'll need to copy from the shipped JVM reside under `$CRL_DIR/jdk<x.y>/jre/`, where `<x.y>` refers to the version. First **`cd`** to that directory:

1. The JDK stands for Java Development Kit. The JRE stands for Java Runtime Environment. Both include a Java Virtual Machine (JVM) JVM. The JDK is a superset of the JRE, and adds files that the developer needs in order to develop Java programs, not just run them.

```
% cd $CRL_DIR/jdk<x.y>/jre/
```

Now start the copies:

1) Copy the files `comm.jar` and `jcl.jar` from `$CRL_DIR/jdk<x.y>/jre/lib/ext/` to the `jre/lib/ext/` directory under your JVM, e.g.,:

```
% cp lib/ext/comm.jar \
  /path/to/your/java/jdk<x.y>/jre/lib/ext/
```

```
% cp lib/ext/jcl.jar \
  /path/to/your/java/jdk<x.y>/jre/lib/ext/
```

2) Copy the file `javax.comm.properties` from `$CRL_DIR/jdk<x.y>/jre/lib/` to the `jre/lib/` directory under your JVM, e.g.,:

```
% cp lib/javax.comm.properties \
  /path/to/your/java/jdk<x.y>/jre/lib/
```

3) Copy the files `libSerial.so` and `libParall.so` from `$CRL_DIR/jdk<x.y>/jre/lib/i386` to the `jre/lib/i386` directory under your JVM and change mode to 777, e.g.,:

```
% cp lib/i386/libSerial.so \
  /path/to/your/java/jdk<x.y>/jre/lib/i386

% cp lib/i386/libParall.so \
  /path/to/your/java/jdk<x.y>/jre/lib/i386

% cd /path/to/your/java/jdk<x.y>/jre/lib/i386

% chmod 777 libSerial.so

% chmod 777 libParall.so
```

11.2.2 Sample CRL Invocation Script

A simplified script for **CRL** is listed here (the last line has been abbreviated, and is described below; `<x.y>` is used in place of the JRE version.) It is included for your information:

```
#!/bin/bash

INSTALLDIR=$CRL_DIR
VERSION=V<x_y>
JARLIB=$INSTALLDIR/LogBook$VERSION/

LOGBOOK_ADMIN_DIR=$HOME/.crl

echo Running Logbook version $VERSION in $INSTALLDIR

JARS="$JARLIB/LogBook_logentry$VERSION.jar:$JARLIB/LogBook_xmlbeans$VERSION.
jar:$JARLIB/LogBook_xmldatatypes$VERSION.jar"
MAIN_JAR=$JARLIB/LogBook$VERSION.jar
COMM_JAR=$CRL_DIR/jre<x.y>/lib/ext/comm.jar
ALL_JARS=$JARS:$MAIN_JAR:$COMM_JAR
```

```
JVM = $CRL_DIR/jre<x.y>/bin/java

$JVM <options> -classpath $ALL_JARS logbook/LogBook $LOGBOOK_ADMIN_DIR
```

The last line of the script is the command that invokes the JVM (and thereby, **CRL**). The five strings on the command line are the following items:

$JVM — the path to the JVM installation

<options> — various options to JVM

-classpath $ALL_JARS

the last option to JVM; this specifies a search path for application classes and resources, which the **CRL** developers have put into various jar files. (A java jar file is an archive of many java classes put into one file.)

logbook/LogBook

the java class to execute

$LOGBOOK_ADMIN_DIR

a parameter which tells **CRL** the parent directory of the directory containing **CRL**'s properties file

11.2.3 Modify the CRL Invocation Script

The second-to-last line of the script is an environment variable definition that sets the path to the JVM that gets invoked:

```
JVM = $CRL_DIR/jre<x.y>/bin/java
```

It's easiest to simply change the JVM environment variable definition and leave the command line (the last line) unchanged. This will make the command run your JVM installation rather than the one shipped with **CRL**.

11.3 Launching the CRL Application

Before running **CRL**, you must perform at least a minimum of configuration and the database must be running. VERY IMPORTANTLY, YOU MUST EDIT THE PROPERTIES FILE! See section 11.1 *Editing the Properties File*. **CRL** may not even run if you neglect this step. And if it does run, it most likely will not be set up the way you want it.

11.3.1 Once you're ready for launch...

An "FYI": The command you use to launch the application actually runs a java command (usually as part of a script) which sets the java class path, optionally defines variable names for use in the properties file, and lastly furnishes the **CRL** top level directory path (as set in the installation) to the application.

11.3.2 Windows (Local and AFS Installations)

On Windows, you will have a desktop icon for **CRL**. Just double-click it to run the program.

11.3.3 Linux (Local UPD Installation)

Run **`setup`** to add the directory containing the **CRL** script to your $PATH, then run the **`crl`** command:

```
% setup crl V1_<x_y> [-f Linux]
% crl
```

11.3.4 Linux (Local Tar File Installation)

For a Linux tar file installation, manually add the directory containing the **CRL** script to your $PATH, then run the program by entering the script name:

```
% <crlscriptname>
```

11.3.5 Linux (AFS Installation)

Running the **CRL** installation in Fermilab's AFS product area assumes that the setup described in section 10.8 *Preparing to Use CRL in Fermilab's AFS Products Area* has been done. Your machine must be running AFS. There should be a script on your local machine that runs **CRL** such that it points to local configuration information.

If you are logged into an fnalu Linux node (for example, flxi02.fnal.gov), or any Linux node that has a **UPS** database setup for the AFS products area, launching the application should be as simple as:

```
% setup crl V<x_y> [-f Linux]
% <local_crl_scriptname>
```

If you are on a Linux system with no **UPS** database, you may need to set an environment variable that points to the product in AFS space, for example:

```
% setenv CRL_DIR /afs/fnal.gov/ups/crl/V<x_y>/Linux
```

and then run the local script that invokes **CRL**:

```
% <local_crl_scriptname>
```

11.4 Configuring Your Desktop

To configure the various elements of the desktop, you need to edit the `LogBookConfig.xml` configuration file. There are also XML configuration files for inquiries and forms.

Never edit the DTD file that goes with an XML file! The **CRL** application code depends on the DTD structure.

`LogBookConfig.xml` is found in the `LogBook_admin` directory. All the XML you need to know in order to understand and edit the configuration file can be found in Chapter 15: *Introduction to XML and DTD Files*. The files themselves are described in Chapter 16: *The CRL Desktop Configuration File*.

11.4.1 Define Keywords

Keywords may not be the first thing you want to configure, but we place this topic first in this section because you can attach keywords to many of the items that follow. If you read about keywords first, then at least you'll know how and where to go back and enter them.

Keywords can be configured to link to logbook entries in order to provide an additional dimension for querying the database when attempting to later identify and retrieve particular entries. Keywords are stored in UPPERCASE.

Each **input container** may have its own set of default, "attachable" keywords pertaining to the container topic. The keywords may be configured for the topic itself, or for a menu/submenu that leads to it. Any or all of these keywords may be configured to link automatically to each entry in the container. You can configure each automatically linked keyword such that it is removable, or not.

Similarly, each **data type** may have keywords associated with it. In this case, every logbook entry of a given data type inserted into a given input container would have the same set of default keywords, and users can choose from among them.

A **desktop page** may also have keywords configured for it; these keywords would be available for all containers, and for all data types on the given page. (These are only meaningful on entry input pages.)

The **CRL application as a whole** may also have keywords configured; these keywords would be available to the user for all containers, and for all data types on all pages.

See section 16.2.3 *Keyword* for configuration information.

11.4.2 Define a new Desktop Page (Data Entry or Report)

A desktop page is a work space in **CRL**. There may be several pages to your desktop; pages are configurable by experiment. Only one page is visible and active at a time.

A desktop page may be configured for data entry and manipulation, or for searching/viewing/manipulating archived entries only. The former is typically called a "data entry page" or "entry-input page", and the latter a "report page". All pages provide one or more menus for the user, and each data entry page also provides a data-entry toolbar.

- The element type `EntryInputPage` is used to define each of the desktop pages on which logbook entries can be made; see section 16.2.6 *EntryInputPage* for information on constructing the XML code.
- The element type `Page` is used to define each of the non-data-input desktop pages in the application, e.g., a page for reports only; see section 16.2.5 *Page* for information on constructing the XML code.

11.4.3 Define a Menu and/or Submenu on a Desktop Page

Each desktop page has a set of menu headings lined up horizontally underneath the page title. These are pull-down menus. These pull-down menus may cascade several levels in order to allow precise categorization of entries or reports.

Menus may be defined for both types of pages, data entry and report (see section 11.4.2 *Define a new Desktop Page (Data Entry or Report)*. On entry input pages, menus are intended to represent general logbook entry categories. You can also include report menus on an entry-input page, but not vice-versa. Report menus represent general reporting categories, e.g., daily report.

☞ Make the menu names and options descriptive!

The menu headings and all the sublevels of categorization except the final one correspond to logbook entry *categories*. The final level of menu categorization (i.e., an option on the lowest-level submenu, or on the menu itself in the absence of submenus) is considered the *topic*. A *container* is associated with a topic.

- The element type `Menu` is used to define a top level menu on a desktop page; see section 16.2.7 *Menu and SubMenu*.

- The element type `Submenu` is used to define each of the submenus coming off a menu or a higher-level submenu; see section 16.2.7 *Menu and SubMenu*.
- The element type `Topic` is used to define each of the menu options on the lowest-level submenu, or on the menu itself in the absence of submenus; see section 16.2.8 *Topic*.

11.4.4 Define a ToolBar with ToolButtons for Data Entry Types

You must configure a data entry toolbar for each desktop page that allows logbook data entry. A toolbar must therefore appear in the declaration of each `EntryInputPage` element. It will display vertically down the right-hand side of the page.

A toolbar must include at least one toolbutton for each logbook entry type that you want to make available on the associated page (e.g., text, plain text, execute command, application output file, form(s), etc.).

The element type `ToolBar` is used to define a toolbar on an entry input desktop page and the element type `ToolButton` is used to define a button on the toolbar. The images that come with the default configuration are included in the **CRL** jar file. See section 16.2.9 *ToolBar and ToolButton* for information on constructing the XML code.

Creating New Toolbutton Images

You can create additional images for toolbuttons. Toolbuttons may be graphic images (`.gif` files) or plain text. We recommend `.gif` files because they look nicer. First make sure you've got a directory to contain these images. It must be located under the same directory that contains the `Logbook_admin` directory, and may be called anything (the default is `images/entryinputpages`). The template file `Button.gif` is provided in this default directory; edit it to make other buttons that match the default ones. Include the whole path in the XML configuration file when pointing to one of these image files.

11.5 Creating Configuration Files for Forms

CRL comes with some ready-made forms that you can use as is, modify, or delete, as you like. You can also create new forms. Each form entry type you add to your **CRL** installation has its own XML form definition file. The `form.dtd` file and the XML form definition files must be located in a directory defined by the `Logbook.file_location.forms_directory` parameter in the Properties file (described in Chapter 14: *CRL's Java Properties*).

The XML elements allowed in a form definition file are listed in section 17.1 *Form Definition Files*, along with examples.[1] To refresh your memory on XML elements and attributes, see section 15.3 *Element Types and Attributes in the DTD File*.

11.5.1 Create/Modify the XML Form Definition Files

There are several things you need to know up front about creating new forms:

- For each new form, a corresponding data entry toolbutton must be added to the toolbar. See section 11.4.4 *Define a ToolBar with ToolButtons for Data Entry Types*.
- You can create forms with text areas, radio buttons, check boxes, selects, tables and lists. You can combine these elements on the same line, if you like.
- With the exceptions of `<Form>` and `<RepeatBlock>`, all form elements (tags) must be contained within a `<Line> ... </Line>` tag; further, a `<Line>` element must not contain either of the above-mentioned element types.
- The entire form and/or individual form elements can be aligned center (the default), right, or left. Both the `<Form>` and `<Line>` elements can use the `align` attribute.
- Forms can be configured such that the entry gets automatically emailed (in HTML format) to one or more individual addresses and/or to one or more mail lists at the time it is archived (see section 11.5.2 *Enable Automatic Electronic Mailing of Form Entries*).
- You can set up a table in your form definition file by specifying the columns and rows of data. See section 11.5.3 *Include Tables in a Form*. On a form entry containing a table, two buttons are displayed for the user: **ADD NEW ROW** and **DELETE A SELECTED ROW**. These buttons are associated with the table portion of the entry, and appear above it.
- You can have your form run a program, the output of which will appear in a text area on the form when a user creates an entry using this form.
- Forms may contain embedded forms via the element `<insertform>`. This enables you to create end forms in which some fields are reloadable and others are not. This technique is described in section 11.5.4 *Create Forms with Selected Reloadable Fields*.

---

1. Unlike the other XML configuration files in **CRL**, the form definition files are not strictly governed by a DTD file; however the file is referenced and must be present.

- Forms may contain "repeat blocks". From the user's point of view a repeat block is a portion of the form entry that is demarcated and displayed along with a **REPEAT** button, which when clicked causes that portion of the form to be duplicated in the entry. From the administrator's point of view, a repeat block is a portion of the form enclosed between `<REPEATBLOCK> ...</REPEATBLOCK>`, intended for said purpose.

  Repeat blocks may contain one or more `<LINE>...</LINE>` elements only; they cannot contain the `<insertform>` element.

11.5.2 Enable Automatic Electronic Mailing of Form Entries

Entries can be sent to email recipients automatically upon archive, or manually by the user. The latter method applies to all data entry types, and is discussed in section 4.7 *Sending Entries via Email*. Automatic mailing applies only to form entries.

In order to enable automatic electronic mailing of form entries, you need to do one of the following:

1) create a mail list file (format and location given below) and insert the filename into the configuration for the toolbutton that corresponds to the form, or

2) insert a destination email address directly into the configuration for the toolbutton that corresponds to the form. It must contain the @ symbol. Optionally, also include a "from" email address and a subject line.

See section 16.2.9 *ToolBar and ToolButton* for the XML format (there is an example of this in the sample code given there). If a mail list file exists and is specified, it will take precedence over email information contained in the **CRL** configuration file.

If you wish to create a mailing list for any of your forms, first create a text file in the directory specified by the parameter `Logbook.file_location.mail_list_directory` in the properties file (Chapter 14: *CRL's Java Properties*). The text file can have any name, but its contents must conform to the format of the following sample file and to the constraints listed below:

```
<MAILLIST>
 <TO>user1@fnal.gov</TO>
 <TO>user2@fnal.gov</TO>
 <TO>userxyz@myuniv.edu</TO>
 <TO>listxyz@myuniv.edu</TO>
 <FROM>user3@fnal.gov</FROM>
 <FROM>listabcd@fnal.gov</FROM>
 <CC>powersthatbe@fnal.gov</CC>
```

```
 <BCC>mefistofele@underworld.org</BCC>
 <SUBJECT>Muon chambers update $D $T</SUBJECT>
</MAILLIST>
```

Your mail list file may contain any number of any of these elements (`<TO>...</TO>`, `<FROM>...</FROM>`, `<CC>...</CC>`, and `<BCC>...</BCC>`) in any order. Each element can contain only one email address. An email address may be an individual address or a mailing list.

The file may also contain one subject element (`<SUBJECT>...</SUBJECT>`), in any position with respect to the other elements. The macros `$D` and `$T` may be used in the subject line; they get replaced by the current date and time, respectively, when the form is archived and the email is sent.

11.5.3 Include Tables in a Form

You can set up a table in your form definition file by specifying the columns and rows of data. The data types that can be inserted into cells of a table include:

DateAndTime	current date and/or time
CheckBox	boolean
Integer	whole number
Double	floating point number
Select	pull-down, editable or noneditable selection box
Field	text

The syntax for setting up a table within a form is as follows (where `datatype_<n>` refers to one of the above data types):

```
<Line>
  <Table>
    <ColumnLabel name="title of column1">
      <datatype_1 ... />
    </ColumnLabel>
    <ColumnLabel name="title of column2">
      <datatype_2 ... />
    </ColumnLabel>
    ...
  </Table>
</Line>
```

For example:

```
<Line>
  <Table>
```

```
      <ColumnLabel name="Date">
        <DateAndTime Date="yes" Time="no" />
      </ColumnLabel>

      <ColumnLabel name="XYZ Status">
        <CheckBox name="XYZ" checked="on" />
      </ColumnLabel>

      <ColumnLabel name="Integer Value">
        <Integer />
      </ColumnLabel>

      <ColumnLabel name="Floating Value">
        <Double />
      </ColumnLabel>

      <ColumnLabel name="Who?">
        <Select editable="yes">
          <Option name="me">
          <Option name="you">
        </Select>
      </ColumnLabel>

      <ColumnLabel name="some text">
        <Field columns="30" rows="1" />
      </ColumnLabel>

    </Table>
  </Line>
```

There is more information on constructing the XML code and a more detailed example in section 17.1 *Form Definition Files*.

11.5.4 Create Forms with Selected Reloadable Fields

What?

You can create a form that is a composite of its own elements and of one or more other forms. The component forms may in turn be composites of yet other forms, ad infinitum. You can configure component forms to reload previously saved data or not. You can configure component forms such that their data get saved to a reload area each time an entry of the end form type is archived, or such that the data are not saved each time.

Why?

Why would you want to embed forms within forms? This technique allows you to collect all the information your experiment needs in the entry while minimizing data input by the user. You can create forms that are simple for the user with some fields initially filled in with new data (e.g., current date and time), other fields containing previously saved information (either a constant value or data from the previous entry of the form), and still other fields blank. All the fields remain editable.

How?

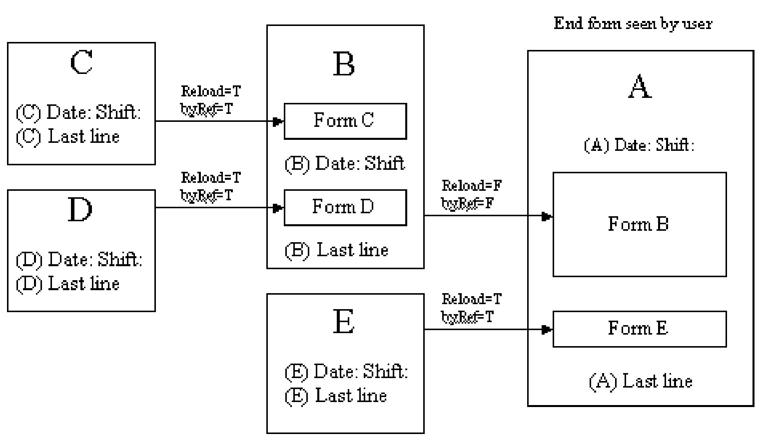
To embed one (source) form inside another (target), you create a separate XML form definition file for each, then in the target file, use the `<insertform>` element (in place of a `<Line>` element), e.g.,:

```
<insertform
  name="source_form.xml"
  reload="false"
  byReference="false"
/>
```

The name attribute is the source form definition filename. The reload and byReference attributes both take values of true or false. If reload="true", then the source form is inserted into the target with reloaded data; if "false", it’s inserted blank. We’ll discuss byReference further on; its function comes into play when the end form is either archived or checkpointed.

The `<insertform>` element will not work inside a repeat block.

Let’s take an example, illustrated by the schematic below. The XML code and other details are given in section 17.1.7 *Sample Form with Embedded Forms*. Here we discuss the concepts.



Form A is the end form that the user sees. It is composed of some native elements (**DATE:** and **SHIFT:** line at the top, and the **LAST LINE** at the bottom) and two inserted (source) forms, B and E. E is a simple form with native elements only, whereas B (as a target) contains (source) forms C and D in addition to some native elements.

Forms C and D are both inserted into B with reload set to true. Form B is inserted into A with `reload="false"`. Form E is inserted into A with `reload="true"`. Form A is marked `reload="false"` (in its definition file). End result:

- A's native elements are either blank or filled in with new data (e.g., Date and Time). (If A were set to `reload="true"`, then user would get a prompt asking whether to reload or not.)
- B's native elements are either blank or filled in with new data (e.g., Date and Time) since reload is false. (An intermediate target form such as B should always be set to `reload="false"`.)
- The portion of B that is form C gets passed to A as it appears in B, i.e., with reloaded data. If there is no previously saved data for one or more of C's fields, those fields will be blank.
- The portion of B that is form D gets passed to A as it appears in B, i.e., with reloaded data. If there is no previously saved data for one or more of D's fields, those fields will be blank.
- The portion of A that is form E contains data from E's reload file, since reload is true. If there is no previously saved data for one or more of its fields, those fields will be blank.

Now the user edits form A. Then user archives the entry, and pulls up a new one. What will the new form entry display? In this case, (`reload="false"` for A) the portions native to A are either new or blank. The non-native portions depend upon the value of the byReference attribute set on insert:

- If `byReference="true"` for a component (or subcomponent) form of A, then when A is archived or checkpointed, the data for that component gets written to a reload file[1] in the forms directory, overwriting any previous instance of the file.
- If `byReference="false"` for a component (or subcomponent) form of A, then data from that component is never written to a reload file.

1. The filename of a reload file is the same as the corresponding form definition file with `RELOAD` prepended. For example, a reload file for the form `c.xml` would be `RELOADc.xml`.

Create Field with Constant Default Value

Say you want the same default value to always appear in a particular field of your end form. You want the user to be able to change it on any given entry, but not to overwrite the reload value. To accomplish this, include that field in a (source) form that gets inserted into the (target) end form (or into one of the end form's component forms). Follow the procedure outlined here:

1) In the target form's definition file, set `reload="true"` and `byReference="true"` upon insertion of source form.

2) Invoke **CRL**, and create an entry using the end form. Either archive the entry or allow it to checkpoint, in order to create the necessary reload files in the forms directory. The field is now initialized.

3) Re-edit the target form definition file, and change byReference to false (keep reload set to true). The reload file for the source form, and hence the field's default value, will never get overwritten now.

Helpful Hints

- All the form definition files should be maintained in the forms directory as defined in the properties file.
- Typically, you want the end form's native elements not to be reloadable; use inserted forms to implement reloadable fields. To do so, set `reload="false"` in the end form's definition file.
- Any time byReference is true, then reload should also be true. Otherwise the information in the reload file will never get used.
- Within a chain of embedded forms (e.g., C inserted into B inserted into A), whenever an intermediate source form is inserted into a target form with `reload="true"` and there is a copy to reload, the source form's inserted forms are ignored; just the latest image of the intermediate source form is used. So, you should insert forms as reloadable only at the start of the chain (e.g., C inserted as reloadable into B, B inserted as NOT reloadable into A).

11.6 Enabling Document View plus Thumbnail for PS and PDF File Entries

CRL gives you the option to include a PostScript or PDF document (with extension `.ps`, `.eps`, or `.pdf`) in a `LogEntryRoot` type entry, similarly to a **ROOT** file. It can be configured to create[1] and display a thumbnail image of the document in the entry. The user would click on the displayed thumbnail to view the entire document in a PS or PDF viewer, in a separate window. The thumbnail image is saved with the entry.

To enable these features, you must specify values for the following four properties in the `LogbookConfigParams.properties` file. See Chapter 14: *CRL's Java Properties* for details on each of these properties:

`Logbook.utils.imagemagick`

defines the path to ImageMagick's convert utility which converts the PS or PDF file to a `.gif` or `.jpg` thumbnail image file

`Logbook.utils.psviewer`

defines the path to a PostScript viewer which is used to display the PS file when user clicks the thumbnail

`Logbook.utils.pdfviewer`

defines the path to the Acrobat PDF viewer which is used to display the PDF file when user clicks the thumbnail

`Logbook.file_location.temp_directory`

defines the directory where the temporary files for the PostScript /PDF convert utility are stored

Two notes:

- If these parameters are left blank or left out all together, **CRL** will still save the PostScript or PDF file and operate fine, but the thumbnail feature and PS/PDF viewing will not be available.
- All temporary files created in the directory specified by the property `Logbook.file_location.temp_directory` upon creation of a thumbnail are removed upon closure of the PS or PDF viewer application.

11.7 Editing the Inquiries Configuration File

The Logbook inquiry XML configuration file governs the fields on which you can query when using the inquiry feature described in section 8.2 *Inquiries*. The DTD and default XML files are listed in section 17.2 *The Logbook Inquiry Configuration File*.

☞ Most **CRL** administrators will have no need to edit these files.

1. It's actually a separate application that creates the thumbnail.

Only edit the XML file if you want to disable/re-enable any of the filters used for inquiries. The inquiry DTD and XML files must be kept in the directory defined in the properties file (see Chapter 14: *CRL's Java Properties*) by the parameter:

```
Logbook.file_location.inquiries_directory
```

11.8 Configuring Print Queues

There are no issues regarding printing from **CRL** installed on a Windows OS.

☞ Fermilab flpr queues are not available; if running **CRL** on Linux, printing requires the **lp** print service to be setup. For KDE, see the Computing Division web page *KDE How-To - Using Printers* at `http://www-oss.fnal.gov/projects/fermilinux/611/adminclass/printers.full.html`.

Other Linux windowing systems provide similar interfaces. There are man pages describing how to do the configuration from the command line using the **`lpadmin`** command.

11.9 Starting the Process Logger Daemon

The Process Logger, described in Chapter 9: *Programmer's Guide to the Process Logger*, is run as a standalone daemon process that monitors specified TCP ports for input, interprets the input as **CRL** entries, and creates and logs the entries. There can be multiple back-to-back messages on a single open TCP connection and many concurrent TCP connections on any TCP port. Each experiment must assign and make known the TCP port number(s) for remote program connections. If your experiment has multiple processes that will create entries in **CRL**, you may want to consider running each on a separate TCP port. The advantage is that you can turn off one process at a time by simply restarting the daemon without that process' assigned port.

The **CRL** administrator needs to first obtain the Plog software from the **CRL** development group (contact *crl-dev@fnal.gov*). There is no Plog-related configuration required in the **CRL** configuration file. Start the Plog daemon at any time before running a program that sends messages to it. Use the following command to start it. (We recommend that you create a script that takes the port number(s) as argument(s), and runs this command.)

```
% java -jar LogBookProcessLogger.jar <CRL_admin_directory>\
  <TCP_port_number> [<additional_TCP_ports>]
```

The command arguments are defined as follows:

`LogBookProcessLogger.jar`	the executable from which Plog program is run
<CRL_admin_directory>	the path to the directory containing **CRL** configuration files and properties file
<TCP_port_number>	the TCP port number assigned to Plog
[<additional_TCP_ports>]	a space separated list of additional TCP port numbers assigned to Plog

The command uses the configuration information in the **CRL** admin directory to determine the **CRL** installation in which the new XML entries and their corresponding HTML web pages are to be stored. At least one port number is required on the command line.

Chapter 12: Managing User Information

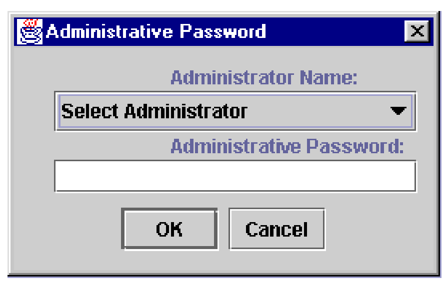
This chapter describes how to add, activate and deactivate users, as well as how to change information about them.

12.1 The Administrator Button

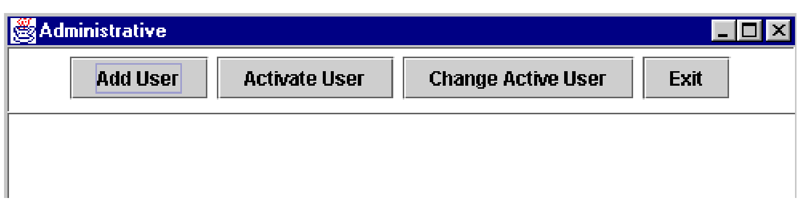
The **ADMINISTRATOR** button on the **CRL** toolbar is used to access the screens on which you can add, activate and deactivate users, and change administrative information about them. An administrative account and password are required to access this feature. Note that if the database is not accessible, the **ADMINISTRATOR** button will not work.

CRL comes with one account that's set up to be an administrator. The account name is *admin1*, and its initial password is set to *logbook*. **Change the password before you do anything else!** (See section 2.5 *Changing your Password*.)

When you click the **ADMINISTRATOR** button, you are prompted for your own administrator name which you select from the drop-down list, and your corresponding administrator password:



Click **OK**. This brings you to the menu of administrative tasks:



12.2 Adding Users

To add a new user, click the **ADD USER** button. The image below shows the information that **CRL** stores for each user. The required fields appear in red, and include: first and last name, password (minimum of 4, maximum of 8 characters), password confirmation, and email address.

There are three user categories:

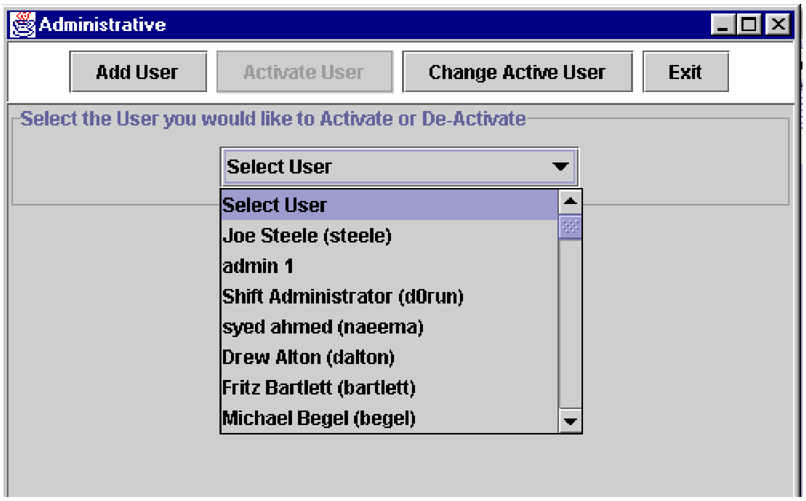
OPERATOR	user of the **CRL** application; can log in and create/edit/archive/annotate/retrieve entries
ADMINISTRATOR	like operator, but also has access to user information and can change it
REMOTE USER	user of the web interface to **CRL**; can retrieve and annotate entries

By default, **OPERATOR** and **REMOTE USER** are checked. Check/uncheck each box as appropriate for the user. Then, if desired, activate the new user before clicking **ADD** to add the user (or **CANCEL** to cancel out). (You can opt to activate the user later.) Once activated, the user will be able to use **CRL** according to the user categories checked.

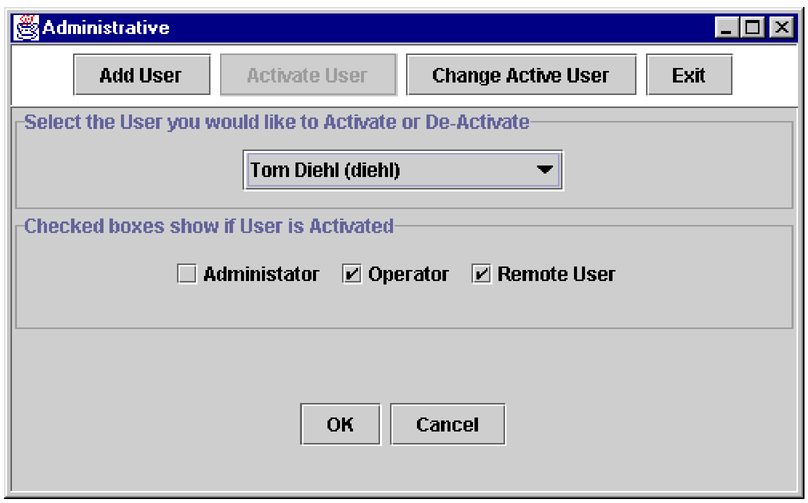
☞ Since operator names are associated with entries in the logbook entry database, once a user is added, he or she can never be deleted from the user list, only deactivated. All operator names must remain "searchable".

12.3 Activating/Deactivating Users

Users can be activated when they are added, or later. If you want to activate or deactivate a user that has already been added, click the **ACTIVATE USER** button.

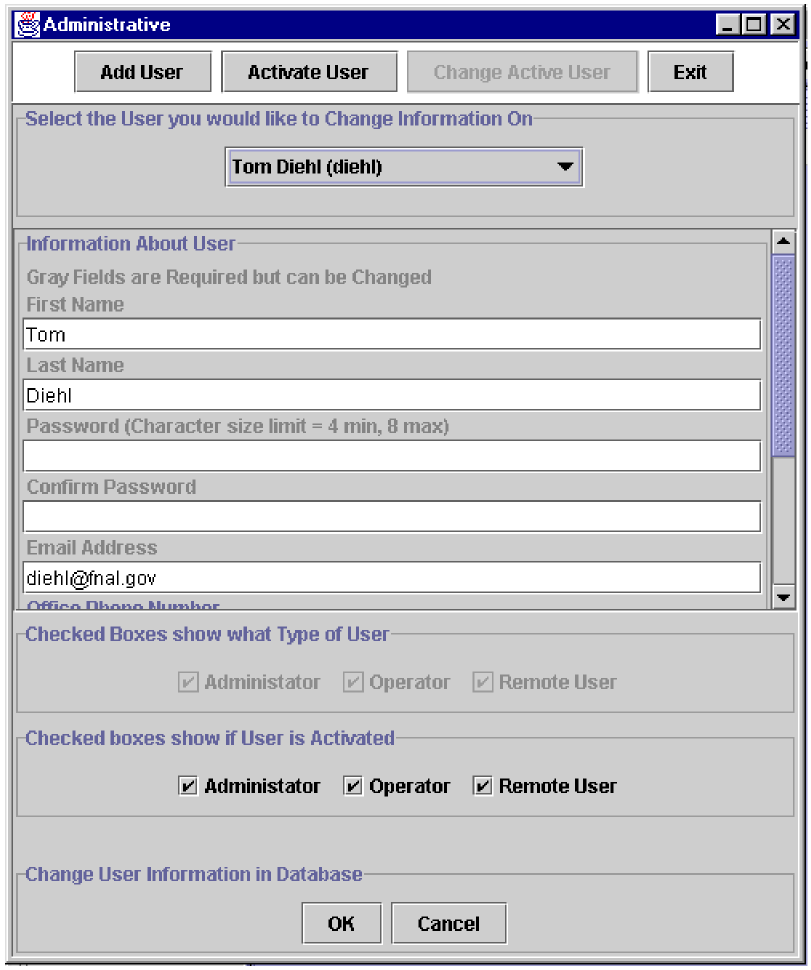


Select the user that you want to activate or deactivate from the pull-down list. If the box corresponding to a user category is checked, the user is activated as that type. Check or uncheck each box as needed, and click **OK**.



12.4 Editing Information on Active Users

For any user that has been activated in at least one user category, you can change the user's information. To do so, click the **CHANGE ACTIVE USER** button. Select the user that you want to change from the pull-down list. The image below shows the top part of the information screen; scroll down to see the bottom part. All fields may be changed, the required fields appear in gray.



☞ If the user has ever been active as a particular user type, you cannot remove that type. Keeping this restriction in mind, you can activate/deactivate the user as one or more user types.

Click **OK** or **CANCEL**.

12.5 Exiting the Administrative Area

To exit out of the Administrative area, click **EXIT**.

Chapter 13: Configuring Web Access to Logged CRL Entries

This chapter lists tasks that are necessary for creating a web site for your **CRL** entries, in order that collaboration members anywhere in the world can search, view, and annotate your experiment's logbook entries. This web access portion of **CRL** is not required for proper functioning of **CRL**.

Your **CRL** installation should be fully configured and operational before you set up the web portion. Without any special configuring on your part, the application automatically saves a copy of each log entry in HTML format. The web access portion uses the relational data base, the HTML entries, and can be made to use inquiries defined from within **CRL**.

Review section 8.4 *Accessing Archived Entries on the Web* before continuing; you need to envision how the end user will interact with your web interface.

13.1 Editing the Properties File

First make sure the following parameter in the properties file is set properly (see Chapter 14: *CRL's Java Properties*):

```
Logbook.file_location.www_directory
```

13.2 Installing and Configuring Additional Software

Next, install the software you'll need. Go to the web page *CRL - Data Base and Web Access* at `http://www-cpd.fnal.gov/CRL/data_base_web_access.htm`.

For web access, additional software requirements are:

- A webserver of your choice. We have a link to the Apache webserver in case you need one: `http://httpd.apache.org/`
- The tomcat product, used to enable JSPs on the webserver: `http://jakarta.apache.org/`

We provide online instructions for the following tasks:

- Getting, Installing, and Running the Apache Webserver for Linux at `http://www-cpd.fnal.gov/CRL/Apache.html`
- Getting, Installing, and Running Tomcat at `http://www-cpd.fnal.gov/CRL/Tomcat.html`
- Installing the mod_jk Apache to Tomcat plugin at `http://www-cpd.fnal.gov/CRL/Tomcat_Apache_Plugin.html`

13.3 Obtaining and Editing the JavaServer Page

Under the `webapps` directory in the **tomcat** subdirectory of the **CRL** installation, you will find `crlsearch.jsp`, a generic JavaServer page (JSP) that you can modify to do web searches on your **CRL** database. There are some comments in this file which describe simple changes you can make to the web search JSP. To edit this file for more complicated changes and reconfigure it for your experiment, you need to understand JavaServer pages.

13.3.1 Configure Keywords Table

In order to allow users to search on entries by keyword, you must enter the list of keywords into the JSP file. The portion of the file in which to insert this list is shown in the sample below, as is the syntax to use:

```
// vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv <==
// Edit/Add Keywords Below for Your Configuration:  <==
// display_value, value, selected?                  <==
// Keep "KEYWORD_ALL" as FIRST item in list         <==
// ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^ <==

Object keywords[][] = {
{KEYWORD_ALL, KEYWORD_ALL, new Boolean(true)},
{"Keyword1", "keyword1", new Boolean(false)},
{"Keyword2", "keyword2", new Boolean(false)},
{"Keyword3", "keyword3", new Boolean(false)},
{"Keyword4", "keyword4", new Boolean(false)},
{"Keyword5", "keyword5", new Boolean(false)},
{NOT+"Keyword1", NOT+"keyword1", new Boolean(false)},
};
```

The last line shows how to define a "NOT" keyword, which can be thought of as the opposite of another keyword in the list (or at least in the database), e.g., NOT_KEYWORD1. This type of keyword makes it possible for users to run a search in which entries linked to a particular keyword can be eliminated. For example, if a user selects NOT_KEYWORD1, then his search will pass over any entry which is linked to KEYWORD1.

13.3.2 Finding Information on JSP

Information about JavaServer Pages dynamically generated web content can be found at the Sun site:

`http://java.sun.com/products/jsp/index.html`

There are lots of other links off of that page. Sun's Technical Resources page for JSP is located at:

`http://java.sun.com/products/jsp/technical.html`

You can find part of a tutorial that includes JSP information at:

`http://java.sun.com/j2ee/tutorial/1_3-fcs/doc/JSPIntro.html`

A resources page at:

`http://java.sun.com/products/jsp/resources.html`

And a good article:

`http://developer.iplanet.com/viewsource/kuslich_jsp/kuslich_jsp.html`

13.4 Configure "One-Click" Searches

First, review sections 8.2 *Inquiries* and 8.4.2 *Run a One-Click Search*.

Now, define an inquiry within the **CRL**, and save it. This creates two files, one ending in `.inq` (for inquiry; this one can be used over again within **CRL**), and one ending in `.winq` (for web inquiry). Both files get stored in the directory defined by the parameter `Logbook.file_location.inquiries_directory` in the properties file.

In the code for your web page, link to the `.winq` file via an HREF command that uses the following type of syntax (do not include ".winq" in the filename):

```
<A HREF="Jsp2.jsp?inquiry=         <!-- Type as is -->
path/to/inquiries_dir/filename     <!-- Substitute path -->
">
Name of Search                     <!-- Substitute name -->
</A>
```

Part III CRL Administrator's Reference Manual

Chapter 14: *CRL's Java Properties*

CRL's Java properties are parameter-value pairs which must contain appropriate values in order for a **CRL** installation to operate correctly. Properties must be defined for database access information, for setting paths to target directories for various data and default browse paths, for turning on or off special features, and so on. The properties are contained in a file called `LogbookConfigParms.properties` which is stored under the `LogBook_admin` directory. The installation's **CRL** administrator must edit this file before **CRL** is invoked.

Chapter 15: *Introduction to XML and DTD Files*

The configuration files for the **CRL** desktop as well as for the inquiry and forms features are provided in **XML** format. In this chapter we provide the information you need in order to understand **CRL**'s **XML** configuration files. This chapter is not intended as a general **XML** reference.

Chapter 16: *The CRL Desktop Configuration File*

In this chapter we illustrate the structure of the **CRL** desktop configuration file (in XML) as dictated by its corresponding DTD file. We discuss each element type and its associated attributes as a unit.

Chapter 17: *Form and Inquiry Configuration Files*

The configuration files for the inquiry and forms features are provided in **XML** format. In this chapter we illustrate the structure of these configuration files.

Chapter 14: CRL's Java Properties

CRL's Java properties are parameter-value pairs which must contain appropriate values in order for a **CRL** installation to operate correctly. Properties must be defined for database access information, for setting paths to target directories for various data and default browse paths, for turning on or off special features, and so on. The properties are contained in a file called `LogbookConfigParms.properties` which is stored under the `LogBook_admin` directory. The installation's **CRL** administrator must edit this file before **CRL** is invoked.

In this chapter, we list the properties with sample values, and provide descriptions. A few notes:

- In the file, each property definition is of the form `property_name = value`.
- The pound symbol (#) at the start of a line in the file indicates a comment line.
- Use the forward slash (/) as a directory delimiter for all the file path names; if the backward slash (\) is needed, a replacement will be done for you automatically.
- The command used to invoke the program (which is part of a `.bat` command file on Windows, or a script on Linux) must include a parameter that defines the top directory for the **CRL** program. **CRL** assumes that a `LogBook_admin` subdirectory resides directly under this top directory, and that the `LogbookConfigParms.properties` file resides in the `LogBook_admin` directory.

Parameter = Sample Value, and Description
`Logbook.debug_value = 0` Set debugging value. This is for development use. Set the value to 0 for "no debugging".
`Logbook.text.charset = UTF-8` Select the character set for writing out XML. (Use UTF-8.)
`Logbook.entry.text.fontsize = 12` Set the default character fontsize for entry text display.

Parameter = Sample Value, and Description
`Logbook.login.class = logbook.Login (or logbook.Login1D0)` Two methods for logging into **CRL** are available, one of which must be selected. The first method (implemented via `logbook.Login`) requires a PIN entry at login, the second method (originally customized for D0 and implemented via `logbook.Login1D0`) does not. These methods are described in Chapter 2: *Invoking and Logging Into CRL.*
`Logbook.login.timeout_in_hours = 9` The entry *Logbook.login.timeout_in_hours = x* will cause operators to be prompted after x hours to make one of three choices. They may either logout, ignore this warning but not future warnings, or ignore this and all future warnings. The prompt will be displayed until the user makes one of the three choices.
`Logbook.login.allow_anonymous = true` The entry *Logbook.login.allow_anonymous = true* will allow operators who did not log in to archive entries. These entries show "anonymous" in the entry header. *True* is the default value if this property is not set. If set to false, CRL allows only logged-in operators to archive entries, and thus all entries are "signed". This prevents creating anonymous entries.
`Logbook.container.filter = true` The entry *Logbook.container.filter = true* applies to all containers **except report containers**. It controls whether entries dropped into the container are filtered (to match category/topic of container). If the value is true, the container will filter archived entries before adding them to the container. If the value is false, there will be no filtering and any type of archived entry may be added to the container.
`Logbook.timezone = America/Chicago` Set the time zone appropriate for the location of your experiment's control room. A list of valid values can be found at `http://www.fnal.gov/docs/products/crl/misc/Time_Zone_Listing.txt`.
`Logbook.checkpoint_interval_in_minutes = 2.5` Entries are checkpointed at the time interval set here. This allows recovery of entries in the event of a crash or accidental deletion. The default interval is 2.5 minutes. At the end of every interval, all entries that have not yet been archived or deleted get written out into a Checkpoint directory. This directory is at the same directory level as Entries.
`Logbook.time_to_start_nagging_in_minutes = 120` "Nagging" alerts users to unarchived entries that have been active for the time period set here. The default period is 120.0 minutes. To disable nagging, set this value to 0.
`Logbook.nag_interval_in_minutes = 25.0` "Nagging" alerts users to unarchived entries that have been active for a long time. Once the time period (above) has elapsed, a message appears on the user's desktop periodically. The time interval between successive messages is set here. The default interval is 25.0 minutes.
`Logbook.entry.image.maxwidth = 200` This value sets the maximum width in pixels of an image in a container. If the width exceeds this, then the image appears as a thumbnail.

Parameter = Sample Value, and Description
`Logbook.entry.header.ShowKeywords = true` If set true, the keywords selected for each entry are displayed in the entry header. If false, then user must click the keyword symbol in the entryto see the selected keywords.
The following seven entries determine how much of the entry header is displayed by default in a CRL container window. The default values specified here can be changed via the Preferences button in the application. Valid values are true and false.
`Logbook.entry.header.ShowCategory = true`
`Logbook.entry.header.ShowCreateTime = true`
`Logbook.entry.header.ShowOperator = true`
`Logbook.entry.header.ShowSavedLocation = true`
`Logbook.entry.header.ShowSavedTime = true`
`Logbook.entry.header.ShowSequenceNumber = true`
`Logbook.entry.header.ShowTopic = true`
The following nine entries define the individual data directories. They can be absolute or relative paths, or an environment variable (use of environment variables is described in Chapter 11: *Configuring and Launching CRL*)

`Logbook.file_location.entry_directory =`
`    D:/path/to/CRLdata (Windows) or`
`    /path/to/CRLdata (Linux)`
Set this to the directory in which to store the archived logbook entry files in XML format. CRL users need write access to this directory in order to log entries. This directory gets created on first launch of CRL.

`Logbook.file_location.forms_directory =`
`    D:/path/to/CRLforms (Windows) or`
`    /path/to/CRLforms (Linux)`
Set this to the directory in which to store the DTD and XML files that define the logbook entry forms. CRL users need read access to files under this directory.

`Logbook.file_location.inquiries_directory =`
`    D:/path/to/CRLinquiries (Windows) or`
`    /path/to/CRLinquiries (Linux)`
Set this to a default location for storing inquiries (see section 8.2 *Inquiries*). (A user can type in or browse for a different location in which to save an inquiry.) CRL users need write access to this directory in order to save inquiries.

`Logbook.file_location.files_directory`
`    D:/path/to/CRLfiles (Windows) or`
`    /path/to/CRLfiles (Linux)`
Set this to the top of a default file browse path (displayed when a user attempts to import a file into a **CRL** entry). For convenience, any files that are generated for the purpose of including them in logbook entries should reside underneath the directory specified here.

Parameter = Sample Value, and Description

`Logbook.file_location.histograms_directory =`
`    D:/path/to/CRLhistograms (Windows) or`
`    /path/to/CRLhistograms (Linux)`
Set this to the top of a default histogram browse path (displayed when a user attempts to import a histogram file into a **CRL** entry). For convenience, any histograms that are generated for the purpose of including them in the logbook should reside underneath the directory specified here.

`Logbook.file_location.annotate_directory =`
`    D:/path/to/CRLannotations (Windows) or`
`    /path/to/CRLannotations (Linux)`
Set this to the directory in which to store the annotation files. CRL users and remote users need write access to this directory in order to annotate entries.

`Logbook.file_location.exported_files_directory =`
`    D:/path/to/CRLexportedFiles (Windows) or`
`    /path/to/CRLExportedFiles (Linux)`
Set this to the top of a default binary file export browse path (displayed when a user attempts to export a binary file from a previously archived entry to the file system). CRL users need write access to this directory.

`Logbook.file_location.www_directory =`
`    D:/path/to/CRLwww (Windows) or`
`    /path/to/CRLwww (Linux)`
The XML entries are automatically copied and converted to HTML for web browsing. Set this to the directory in which to store the HTML entry files. CRL users need write access to this directory. This directory gets created on first launch of CRL.

`Logbook.file_location.mail_list_directory =`
`    D:/path/to/CRLmaillists (Windows) or`
`    /path/to/CRLmaillists (Linux)`
Form entries may be configured to get mailed to a list of recipients when the entries are archived. This parameter gives the path to the directory containing the mail list files, if used.

`Logbook.file_location.temp_directory =`
`    D:/path/to/Temp (Windows) or`
`    /path/to/Temp (Linux)`
This defines the directory where the temporary files for the PostScript /PDF convert utilities are stored. If this is not specified, a thumbnail will not be created and the user will not be able to view PostScript or PDF files directly.

`Logbook.mail.from = joe@fnal.gov (or from_list.txt)` This defines a "from" email address for entries that are emailed out. This may be an individual address, a mailing list or a file. If file, the file must reside in the directory specified by: Logbook.file_location.mail_list_directory
`Logbook.mail.smtp_host = smtp.fnal.gov` This defines the outgoing mail server.

Parameter = Sample Value, and Description

Logbook.utils.imagemagick =
    D:/path/to/imagemagick.exe (Windows), or
    /path/to/imagemagick (Linux)
This defines the path to ImageMagick's convert utility. Install this utility on your system. This utility is used to generate a small jpg (a thumbnail) of a PostScript or PDF file. The thumbnail itself gets saved with the entry. If this property is not specified, no thumbnail will be generated.

Logbook.utils.psviewer =
    D:/path/to/gsview32.exe (Windows), or
    /path/to/ghostview (Linux)
This defines the path to a PostScript viewer, e.g., GhostView. Install this utility on your system. It is used to view PostScript files in archived entries.

Logbook.utils.pdfviewer
    D:/path/to/AcroRd32.exe (Windows), or
    /path/to/Acroread (Linux)
This defines the path to the Acrobat PDF viewer. Install this utility on your system. It is used to view PDF files in archived entries.

The following ten entries define the database information.
Logbook.database.enabled = true Set to "true" (no quotes) to enable database code (the usual case). Only set to "false" if for some reason the database is temporarily unavailable. When set to "false", logbook entries are still collected in the file system as XML, but pointers to them are not included in the SQL database.
Logbook.database.vendor = mysql The database vendor
Logbook.database.version = 3.22 The database version

Logbook.database.driver =
    org.gjt.mm.mysql.Driver (for MySQL), or
    sun.jdbc.odbc.JdbcOdbcDriver (for ODBC)
The database driver appropriate for the database vendor and version.

Logbook.database.connection_url =   jdbc:odbc:<filename> (Optional) If the connection URL (location of the database) is not of the form <protocol>//<machine_name>/<database>?user=<username>&password=<password>, then set this value to the connection URL. If set, the value must start with jdbc:. The sample value shows ODBC with local file. If left unset, then **CRL** uses the values of Logbook.database.protocol, Logbook.database.server, Logbook.database.dbms_name, Logbook.database.username, and Logbook.database.password to build the value.

Parameter = Sample Value, and Description
`Logbook.database.protocol = jdbc:mysql: (or jdbc:odbc:)` The database protocol. It must start with `jdbc:`. If `Logbook.database.connection_url` is set, this is ignored.
`Logbook.database.server = xyz.fnal.gov` The database server is the IP address of the machine running database. If `Logbook.database.connection_url` is set, this is ignored.
`Logbook.database.dbms_name = crl_dbms` The name of the database set up for use by CRL. If `Logbook.database.connection_url` is set, this is ignored.
`Logbook.database.username = myname` The database username to use for CRL's access to the database. If `Logbook.database.connection_url` is set, this is ignored.
`Logbook.database.password = mypassword` The username's password. If `Logbook.database.connection_url` is set, this is ignored.

Chapter 15: Introduction to XML and DTD Files

The configuration files for the **CRL** desktop as well as for the inquiry and forms features are provided in **XML** format. In this chapter we provide the information you need in order to understand **CRL**'s **XML** configuration files. This chapter is not intended as a general **XML** reference.

15.1 XML Compared to HTML

XML is most easily described by starting with **HTML**. **HTML** markup consists of *tags* that define text structures, formatting, and so on, for documents (e.g., <H1> ... </H1> for top level header), and *attributes* that set properties for those tags (e.g., <H1 ALIGN="CENTER">). **HTML** is limited to documentation. It is not case-sensitive.

XML is a configurable, extensible language with syntax similar to **HTML**. **XML** *is* case-sensitive. **XML** can be used for documentation, but it is more versatile than just that. In particular, the **CRL** code, written in **Java**, is designed to read in a configuration file supplied in **XML** format. *Elements* in **XML** are analogous to tags in **HTML**; they represent structures or desired behavior. *Attributes* define properties of elements. Elements are used with attributes in **XML** the same way as tags are in **HTML**:

```
<ELEMENT ATTRIBUTE="value">data</ELEMENT>
```

The configuration files for **CRL** are provided in *valid* **XML** (as opposed to *well formed* **XML**)[1]. A file written in valid **XML** is based upon a document type definition (DTD) file. When you install **CRL**, the following DTD and **XML** files for **CRL** itself and for its inquiry feature (described in section 8.2 *Inquiries*) are stored under the LogBook_admin directory:

- LogBookConfig.dtd
- LogBookConfig.xml
- LogBookInquiryConfig.dtd
- LogBookInquiryConfig.xml

1. There are many references on the Web which discuss valid and well formed XML. In writing this section, the author referenced http://pdbeam.uwaterloo.ca/~rlander/XML_Tutorial/.

The DTD and **XML** files for forms[1] are located under a separate directory, set by the `Logbook.file_location.forms_directory` in the properties file.

The DTD file defines the *element types* and the *attribute lists* that can be used in the corresponding **XML** file as elements and attributes, respectively, in markup declarations.

In order to edit an **XML** configuration file effectively, you need to understand both it and the DTD file so that you know what elements/attributes you're allowed to include, and what they mean. The DTD file and the **CRL** code have been developed in an integrated fashion and are highly dependent upon each other.

Do not edit the DTD files! It will cause CRL to malfunction or become inoperable!

The **XML** configuration files can be edited at will as long as the changes conform to the DTD files, and as long as information corresponding to hard-coded items/functions in the **CRL** code are not changed. Use a good **XML** authoring tool rather than a straight text editor in order to ensure that your edits are valid.

15.2 Let's Get a Feel For It...

As we said, **XML** can be thought of as **HTML** with user-configurable elements and attributes. So, before we get too deep, here's a sample scenario intended to give you a feel for how these entities are defined and used.

In the DTD file, we have the following element type declaration for `EntryInputPage`, which corresponds to a page on the **CRL** desktop.

```
<!ELEMENT EntryInputPage (Keyword?,Menu+,ToolBar?)>
<!ATTLIST EntryInputPage
        class CDATA #REQUIRED
        title CDATA #REQUIRED
        tooltip CDATA #REQUIRED>
```

Don't worry about all the syntactic details yet, just notice that it defines `EntryInputPage`, that `Keyword`, `Menu` and `Toolbar` appear in the declaration (it turns out they are also element types, defined in the same DTD file), and that `EntryInputPage` has a list of attributes (`class`, `title` and `tooltip`).

1. The forms definition files are not written in "valid XML"; they do not strictly follow a DTD file grammar.

In the **XML** configuration file, this `EntryInputPage` element type can be used to define a page on the desktop that allows data entry. The `EntryInputPage` element shown below (taken from the default **CRL** configuration file and edited for brevity) includes the listed attributes. It also includes `Keyword`, `Menu` and `Toolbar` elements.

```
<EntryInputPage class="logbook.Page_EntryInput" title="Tutorial"  tooltip="Tutorial">

  <Keyword  type="page">
    Tutorial,
  </Keyword>

  <Menu name="Tutorial" >
    <Topic class="logbook.MyInternalTopicFrame" name="General Log"
      double-click="logbook.logentry.LogEntryText"  size="90%,90%"
      offset="5%,5%" command="add_to_Menu_Page">
        <Keyword type="category">
          General_Log
        </Keyword>
    </Topic>
...

  <ToolBar>
    <ToolButton class="logbook.logentry.LogEntryImage" tip="Drag and Drop
      MiniBooNE Detector Button to Selected Container"
      image="images/entryinputpages/boone.gif"
      parm1="http://www-boone.fnal.gov/images/illustration.gif" command1="" />
    <ToolButton class="logbook.logentry.LogEntryText" tip="Drag and Drop Text
      Button to Selected Container"  image="images/entryinputpages/Text.gif"
      command1=""  />
...
  </ToolBar>

</EntryInputPage>
```

15.3 Element Types and Attributes in the DTD File

15.3.1 Element Type Declarations

In **CRL**, elements typically define items on the desktop, e.g., a desktop page, a menu, a toolbar, and so on. Each element type is defined in the DTD file in the following form:

```
<!ELEMENT NAME (CONTENT1-plus-OPERATOR,CONTENT2-plus-OPERATOR)>
```

In the **CRL** DTD file, the content area consists of either a sequence of other element types (these element types must also be defined in the DTD file), or the string `#PCDATA`, which indicates text. An operator (as in `CONTENT-plus-OPERATOR`) indicates whether the content is optional or required, and repeatable or not. The operators used in the **CRL** DTD file include:

`*`	optional and repeatable
`?`	optional, not repeatable
`+`	required and repeatable
no operator	required, not repeatable

Multiple content elements are separated by a comma and must be used in the order listed. Here is an example element type declaration from the **CRL** DTD file:

```
<!ELEMENT EntryInputPage (Keyword?,Menu+,ToolBar?)>
```

This tells us that when used in the **XML** file, the element type `Page` can include the three element types `Keyword`, `Menu` and `Toolbar` as content (see example below). The operator symbols with them indicate that the element `Page`:

- *can* include one and only one occurrence of the element `Keyword`; if included, it must come before any occurrences of `Menu`
- *must* include at least one occurrence of `Menu`
- *can* include one and only one occurrence of the element `Toolbar`; if included, it must come after occurrences of `Menu`

Every element in the XML configuration file must be specifically closed at the end. E.g., every `<ElementName>` has a corresponding `</ElementName>`. If an element has no subelements, it may be constructed as `<ElementName ... />`, where the slash-angle bracket (`/>`) closes it.

15.3.2 Attributes and Attribute Lists

Each element type in the DTD file may have an associated *attribute list* defined. This list determines set of possible attributes that element can take in the **XML** file. In **CRL**, attributes typically define titles, roll-over text, file locations, and so on, and it is straightforward to assign values to them. A few attributes are more exacting; for example, values for the `class` attribute are restricted to those in the **CRL** class library, listed in section 16.3 *The CRL Java Class Libraries*.

The **CRL** DTD file includes only CDATA type attributes[1]. This type of attribute is also called a *string attribute* since in **XML** markup it takes a text string as a value. A CDATA attribute list takes the form:

```
<!ATTLIST ELEMENT-TYPE
        ATTRIBUTE1 CDATA DEFAULT
        ATTRIBUTE2 CDATA DEFAULT
        ...>
```

1. There are several different kinds of attributes; defining them is beyond the scope of this document.

where (in the **CRL** DTD file) the `DEFAULT` value is one of the following:

- `#REQUIRED` (required)
- `#IMPLIED` (optional)

15.3.3 Example of Element Type Declaration plus Attribute List

Here is an example from the **CRL** DTD file of a declaration for the element type `EntryInputPage` followed by its corresponding attribute list:

```
<!ELEMENT EntryInputPage (Keyword?,Menu+,Toolbar?)>
<!ATTLIST EntryInputPage
        class CDATA #REQUIRED
        title CDATA #REQUIRED
        tooltip CDATA #REQUIRED>
```

This tells us that in the **XML** file, whenever the element `EntryInputPage` is used, it must provide values for all three listed attributes, e.g.,:

```
<EntryInputPage
  class="logbook.Page_EntryInput"
  title="Tutorial"
  tooltip="Tutorial">
  ...
</EntryInputPage>
```

What these particular attributes do is discussed in Chapter 16: *The CRL Desktop Configuration File*.

Chapter 16: The CRL Desktop Configuration File

In this chapter we illustrate the structure of the **CRL** desktop configuration file (in XML) as dictated by its corresponding DTD file. We discuss each element type and its associated attributes as a unit.

We recommend that you read and understand the material in Chapter 15: *Introduction to XML and DTD Files* before delving into the material presented here. See Chapter 11: *Configuring and Launching CRL* for a task-oriented approach to configuring the application; it refers you to task-appropriate sections in the current chapter.

16.1 CRL Configuration DTD File Listing

The DTD file defines the *element types* and the *attribute lists* that can be used in the corresponding **XML** file as elements and attributes, respectively, in markup declarations. Before the first element type declaration in the DTD file, there is an **XML** declaration which identifies the file to the parser as type **XML** and specifies the character encoding used. Immediately after this statement, the element type declarations and attribute lists begin.

Any changes to the DTD file could cause CRL to malfunction or become inoperable. This file should not be edited!

```
<?xml encoding="US-ASCII"?>
<!ELEMENT LogBookConfig (Keyword*,InitialTabs)>
<!ELEMENT Keyword (#PCDATA)>
<!ATTLIST Keyword
  type CDATA #REQUIRED
  class CDATA #IMPLIED>
<!ELEMENT InitialTabs (EntryInputPage*,Page*,EntryInputPage*)>
<!ELEMENT Page (Keyword?,Menu+)>
<!ATTLIST Page
  class CDATA #REQUIRED
  title CDATA #REQUIRED
  tooltip CDATA #REQUIRED>
<!ELEMENT EntryInputPage (Keyword?,Menu+,ToolBar?)>
<!ATTLIST EntryInputPage
  class CDATA #REQUIRED
  title CDATA #REQUIRED
  tooltip CDATA #REQUIRED>
<!ELEMENT Menu (Keyword?,Topic*,SubMenu*,Topic*,SubMenu*)>
<!ATTLIST Menu name CDATA #REQUIRED>
<!ELEMENT SubMenu (Keyword?,Topic*,SubMenu*,Topic*,SubMenu*)>
```

```
<!ELEMENT Topic (Keyword?) >
<!ATTLIST Topic
  class CDATA #REQUIRED
  name CDATA #REQUIRED
  command CDATA #REQUIRED
  inquiryfile CDATA #IMPLIED
  double-click CDATA #IMPLIED
  double-click-parm1 CDATA #IMPLIED
  double-click-parm2 CDATA #IMPLIED
  size CDATA #IMPLIED
  offset CDATA #IMPLIED
  parm1 CDATA #IMPLIED
  parm2 CDATA #IMPLIED
  auto-entry CDATA #IMPLIED
  auto-entry-parm1 CDATA #IMPLIED
  auto-entry-parm2 CDATA #IMPLIED>
<!ATTLIST SubMenu
  name CDATA #REQUIRED>
<!ELEMENT ToolBar (ToolButton*)>
<!ELEMENT ToolButton EMPTY>
<!ATTLIST ToolButton
  class CDATA #REQUIRED
  tip CDATA #REQUIRED
  text CDATA #IMPLIED
  image CDATA #IMPLIED
  parm1 CDATA #IMPLIED
  parm2 CDATA #IMPLIED
  keyword CDATA #IMPLIED
  maillist CDATA #IMPLIED
  mailsubject CDATA #IMPLIED
  mailfrom CDATA #IMPLIED
  command1 CDATA #REQUIRED>
```

16.2 Element Type and Attribute Listing

We don't list the entire default **XML** configuration file for the **CRL** application in the manual; it is quite long. We believe it's more instructive to describe each of the elements and attributes listed in it.

Note that for elements containing no nested elements, the following two constructions are equivalent:

```
<ElementName AttributeList> </ElementName>
```

and

```
<ElementName AttributeList />
```

XML is a case-sensitive language, however the code that uses this **CRL** configuration file ignores case. Therefore, you can too!

16.2.1 XML Declaration

An **XML** declaration always appears in the form `<?xml ... ?>`. In the **XML** file (as opposed to the DTD), the **XML** declaration takes the form:

```
<?xml version="1.0"?>
```

This is followed by a line that identifies the DTD file on which the **XML** is based:

```
<!DOCTYPE LogBookConfig SYSTEM "LogBookConfig.dtd">
```

16.2.2 LogBookConfig

The `LogBookConfig` element type is defined in the DTD file as follows:

```
<!ELEMENT LogBookConfig (Keyword*,InitialTabs)>
```

This statement declares the element type `LogBookConfig`. `LogBookConfig` represents the overall **CRL** application, and is called the *root* element type.

In the **XML** configuration file, all the elements are included within it (i.e., the entire configuration file content is enclosed within `<LogBookConfig>` and `</LogBookConfig>`, with the exceptions of the `<?xml...?>` and `<!DOCTYPE...>` declarations at the top of the configuration file).

`LogBookConfig` allows content of the element types `Keyword` (optional and repeatable) and `InitialTabs` (required and non-repeatable). Therefore, all the other element types that may appear in the configuration file (listed in the DTD file) must be defined within one of these two element types, or within element types listed in their content, or further down the chain.

There is no attribute list for `LogBookConfig` and thus it has no attributes.

So far then, the **XML** configuration file is constrained by the DTD file to have a structure like the following (keeping in mind the keyword elements are optional, and must be of type “overall”; see section 16.2.3 *Keyword*):

```
<?xml...?>
<!DOCTYPE...>

<LogBookConfig>
   <Keyword type="overall">
     *KEYWORD1, ... ,KEYWORDN
   </Keyword>
   ... (more Keyword elements)

   <InitialTabs (no attributes)>

      ... (bulk of configuration) ...

   </InitialTabs>

</LogBookConfig>
```

16.2.3 Keyword

The `Keyword` element type is defined in the DTD file as follows:

```
<!ELEMENT Keyword (#PCDATA)>
    <!ATTLIST Keyword
    type CDATA #REQUIRED
    class CDATA #IMPLIED>
```

The `Keyword` element type is used to define the keywords in the application. **CRL** supports four types of keywords:

Type	Element Types which may contain Keyword Elements	Description
overall	LogBookConfig	These keywords are associated with the **CRL** application as a whole. They are available to be attached to any logbook entry.
entry	LogBookConfig	These keywords are associated with particular logbook data entry types (e.g., text, image, and so on). The `class` attribute on the `Keyword` element is used to specify the entry type. For a given entry, **CRL** reads in the class and makes available the keywords corresponding to the entry type. See section 16.3 *The CRL Java Class Libraries* for information on determining the `class` attribute value.
page	EntryInputPage, Page	These keywords are associated with particular desktop data entry pages. A keyword of this type is available to be attached to any entries made on its corresponding desktop page.
category	Menu, SubMenu, Topic	Keywords can be associated with particular menus, submenus and topics. A keyword of this type is available to be attached to any entries made in containers whose menu hierarchy includes the menu, submenu or topic for which the keyword was defined.

All keyword elements contain a text string (contents is listed as `#PCDATA`) which is a comma-separated list of keywords. Keywords should be entered in UPPERCASE only; **CRL** stores them as uppercase. Each keyword in the list may be preceeded by either one asterisk (`*`) or two (`**`):

- One asterisk (`*`) means the keyword is automatically inserted into associated entries, but the keyword is removable.
- Two asterisks (`**`) mean the keyword is automatically inserted into associated entries, and it is not removable.

• If no asterisks precede the keyword, it is not inserted automatically into any associated entry.

All keyword elements must have a `type` attribute, but the `class` attribute is used only for the `entry` type, as described in the box above.

The keyword elements listed above under the `LogBookConfig` element could be expanded as follows (note the uppercase used for keywords):

```
<Keyword type="overall">
     KEYWORD1, **KEYWORD2, ... ,KEYWORDN
</Keyword>

<Keyword type="entry" class="logbook.logentry.LogEntryText">
     *KEYWORD1, ... ,KEYWORDN
</Keyword>

<Keyword type="entry" class="logbook.logentry.LogEntryImage">
     KEYWORD1, ... ,*KEYWORDN
</Keyword>

<Keyword  type="page">
     **KEYWORD1, ... ,KEYWORDN
</Keyword>

<Keyword  type="category">
     KEYWORD1, ... ,**KEYWORDN
</Keyword>
```

16.2.4 InitialTabs

The `InitialTabs` element type is defined in the DTD file as follows:

```
<!ELEMENT InitialTabs (EntryInputPage*,Page*,EntryInputPage*)>
```

It is used to define the page tabs in the application. It has no attributes. It allows content of the element types `EntryInputPage`, `Page`, and again `EntryInputPage` (all of them optional and repeatable, within the sequence specified). So the `InitialTabs` element shown in section 16.2.2 *LogBookConfig* can be expanded to:

```
   <InitialTabs>

     <EntryInputPage attribute1="value" ...>  ... </EntryInputPage>
     ... (more EntryInputPage elements)

     <Page attribute1="value" ...>  ... </Page>
     ... (more Page elements)

     <EntryInputPage attribute1="value" ...>  ... </EntryInputPage>
     ... (more EntryInputPage elements)

   </InitialTabs>
```

16.2.5 Page

The `Page` element type is defined in the DTD file as follows:

```
<!ELEMENT Page (Keyword?,Menu+)>
    <!ATTLIST Page
            class CDATA #REQUIRED
            title CDATA #REQUIRED
            tooltip CDATA #REQUIRED>
```

The element type `Page` is used to define each of the non-data-input desktop pages in the application, i.e., report pages.

The element type `Page` has an associated attribute list:

- `class` should be set to the value `logbook.Page_General`
- `title` specifies the page title displayed at the top of the page.
- `tooltip` specifies the text that appears on mouse rollover of the page tab.

EntryInputPage is similar to **Page** but more versatile. We provide an example in section 16.2.6 *EntryInputPage*.

16.2.6 EntryInputPage

The `EntryInputPage` element type is defined in the DTD file as follows:

```
<!ELEMENT EntryInputPage (Keyword?,Menu+,ToolBar?)>
    <!ATTLIST EntryInputPage
            class CDATA #REQUIRED
            title CDATA #REQUIRED
            tooltip CDATA #REQUIRED>
```

The element type `EntryInputPage` is used to define each of the desktop pages on which logbook entries can be made. Notice that it differs from `Page` (immediately above) only by the addition of the (optional) `ToolBar` content type. (The toolbar is used for entry input, and is discussed below.) Any keyword specified for an entry input page must be of the type “page” (see section 16.2.3 *Keyword*).

The `Page` and `EntryInputPage` elements (shown above in the box in section 16.2.4 *InitialTabs*) can be expanded to:

```
<InitialTabs>

<EntryInputPage class="logbook.Page_EntryInput" title="title1" tooltip="tooltip1">
   <Keyword type="page"> KEYWORD1, ... ,KEYWORDN </Keyword>
   <Menu attribute1="value" ...>     ... </Menu>
   ... (more Menu elements)
   <ToolBar (no attributes)>  ... </ToolBar>
</EntryInputPage>

... (more EntryInputPage elements)

<Page class="logbook.Page_General" title="title2" tooltip="tooltip2">
   <Keyword type="page"> KEYWORD1, ... ,KEYWORDN </Keyword>
   <Menu attribute1="value" ...>     ...   </Menu>
   ... (more Menu elements)
</Page>

... (more Page elements)

<EntryInputPage class="logbook.Page_EntryInput" title="title3" tooltip="tooltip3">
   <Keyword type="page"> KEYWORD1, ... ,KEYWORDN </Keyword>
   <Menu attribute1="value" ...>     ... </Menu>
   ... (more Menu elements)
   <ToolBar (no attributes)>  ... </ToolBar>
</EntryInputPage>

... (more EntryInputPage elements)

</InitialTabs>
```

16.2.7 Menu and SubMenu

The `Menu` and `Submenu` element types are defined in the DTD file as follows:

```
<!ELEMENT Menu (Keyword?,Topic*,SubMenu*,Topic*,SubMenu*)>
    <!ATTLIST Menu
            name CDATA #REQUIRED>

<!ELEMENT SubMenu (Keyword?,Topic*,SubMenu*,Topic*,SubMenu*)>
    <!ATTLIST SubMenu
            name CDATA #REQUIRED>
```

`Menu` appears in the content lists of the two page element types, described in sections 16.2.5 *Page* and 16.2.6 *EntryInputPage*. It is used to define entry-input menus on entry-input pages, and report menus on non-entry-input pages[1]. Notice that it can include submenus and/or topics (topics are the lowest-level menu items), and keywords. Any keyword specified for a menu or submenu must be of the type “category” (see section 16.2.3 *Keyword*). `Menu` requires a `name` attribute, which takes a text string giving the menu name. `SubMenu` is defined similarly; `Topic` is a little different, see section 16.2.8 *Topic*. Submenu and topic names appear as menu options on their next highest level menu.

The `Menu` element can be expanded to:

```
<Menu name="value1">

   <Keyword type="category"> KEYWORD1, ... ,KEYWORDN </Keyword>
   <Topic attribute1="value1" ...> ... </Topic>

   ... (more Topic elements)

   <SubMenu name="value2">
     <Keyword type="category"> KEYWORD1, ... ,KEYWORDN </Keyword>
     <SubMenu name="value3">
       <SubMenu name="value4">
         <Topic type="category" attribute2="value5" ...>
         ...
         </Topic>
         ... (more Topic elements)
       </SubMenu>
     </SubMenu>
   </SubMenu>

   ... (more SubMenu and/or Topic elements)

</Menu>
```

16.2.8 Topic

The `Topic` element type is defined in the DTD file as follows:

```
<!ELEMENT Topic (Keyword?) >

      <!ATTLIST Topic
        class CDATA #REQUIRED
        name CDATA #REQUIRED
        command CDATA #REQUIRED
        inquiryfile CDATA #IMPLIED
        double-click CDATA #IMPLIED
        double-click-parm1 CDATA #IMPLIED
        double-click-parm2 CDATA #IMPLIED
        size CDATA #IMPLIED
        offset CDATA #IMPLIED
        parm1 CDATA #IMPLIED
        parm2 CDATA #IMPLIED
        auto-entry CDATA #IMPLIED
```

1. Note that you can add a report menu or menu item to an entry-input page as well. The class value for each topic (menu item) determines whether an entry-input container or a report container comes up when the user clicks on the menu item. See section 16.2.8 *Topic*.

```
auto-entry-parm1 CDATA #IMPLIED
auto-entry-parm2 CDATA #IMPLIED>
```

```
auto-entry CDATA #IMPLIED

auto-entry-parm1 CDATA #IMPLIED

auto-entry-parm2 CDATA #IMPLIED>
```

`Topic` is used with menu items as shown in section 16.2.7 *Menu and SubMenu*. There is a one-to-one correspondence between topics and containers. Any keyword specified for a topic must be of the type "category" (see section 16.2.3 *Keyword*).

The element type `Topic` has the attribute list as shown above, some attributes are required, others are optional (#IMPLIED). Here we provide descriptions of the attributes:

`class` determines what type of container pops up when the topic is selected. The values recognized by **CRL** are listed in section 16.3.1 *Container (Topic) Classes*.

`name` specifies the topic name; it is displayed as the title of the corresponding container

`command` All classes use either `command="add_to_Menu"` to add the topic to the menu, or `command="add_to_Menu_Page"` to add it to the menu and to pop up a container of this topic automatically on the page

`inquiryfile` Currently not implemented.

`double-click` specifies the data entry type to create when a user double-clicks in container window. The `double-click` attribute takes the same values as the the `class` attribute of the ToolButton corresponding to the desired entry type; see section 16.2.9 *ToolBar and ToolButton*.

`double-click-parm1`

specifies the `parm1` (as defined in section 16.2.9 *ToolBar and ToolButton*) that corresponds to the logentry type specified by `double-click`, above. (Replaces the `parm1` attribute for **Topic**.)

`double-click-parm2`

Not currently used. Replaces the `parm2` attribute for **Topic**.

`size` specifies the default size of the container window for the corresponding topic. Specify the horizontal and vertical dimensions as `size="x,y"` in absolute number of pixels (e.g., `size="400,500"`) or as percent of **CRL** desktop size (e.g., `size="85%,90%"`). `x` and `y` can be specified in different units, e.g., `size="90%,500"` is acceptable.

`offset` specifies the default offset of the container window for the corresponding topic, measured from the top left corner. Specify the horizontal and vertical offsets as `offset="x,y"` in absolute number of pixels (e.g., `offset="4,5"`) or as percent of **CRL** desktop size (e.g., `offset="5%,10%"`). `x` and `y` can be specified in different units, e.g., `offset="5%,10"` is acceptable.

`parm1`	This will be phased out; it is replaced by `double-click-parm1`.
`parm2`	This will be phased out; it is replaced by `double-click-parm2`.
`auto-entry`	Not currently implemented.
`auto-entry-parm1`	Not currently implemented.
`auto-entry-parm2`	Not currently implemented.

So the `Topic` element shown above can be expanded to (notice, one keyword element only is allowed; it's optional):

```
<Topic class="logbook.MyInternalTopicFrame" name="Comments" command="add_to_Menu"
 double-click="logbook.logentry.LogEntryText" size="400,500" offset="10,10">

    <Keyword type="category"> KEYWORD1, ... ,KEYWORDN </Keyword>

</Topic>
```

This container is defined such that it does not automatically pop up on the window, but if user double-clicks in it a text entry will be created.

16.2.9 ToolBar and ToolButton

The `ToolBar` and `ToolButton` element types are defined in the DTD file as follows:

```
<!ELEMENT ToolBar (ToolButton*)>
```

<!ELEMENT ToolButton EMPTY>

```
<!ELEMENT ToolButton EMPTY>
<!ATTLIST ToolButton
         class CDATA #REQUIRED
         tip CDATA #REQUIRED
         text CDATA #IMPLIED
         image CDATA #IMPLIED
         parm1 CDATA #IMPLIED
         keyword CDATA #IMPLIED
         maillist CDATA #IMPLIED
         mailsubject CDATA #IMPLIED
         mailfrom CDATA #IMPLIED
         command1 CDATA #REQUIRED>
```

The `ToolBar` element type appears as content in the declaration of all `EntryInputPage` pages. It indicates that a toolbar for data entry is to appear on these pages. Its content consists of zero, one, or more `ToolButton` elements, described below. It has no attribute list.

The `ToolButton` element type appears as content in the declaration of `ToolBar`, described above. Each tool button represents a type or source of logbook data entry, definable via its attribute list:

`class`	specifies the data entry type. The allowed values are listed and described in section 16.3.2 *Entry type Classes*.
`tip`	defines the text that appears upon mouse rollover of the toolbutton.
`text`	specifies the text of the button (not used if image file specified)
`image`	specifies the file containing the image of the button (specified relative to the directory above `LogBook_admin`, unless full path is given). If image file is not specified, the text (above) is used to specify the button. It is generally preferred to use an image because it looks nicer.
`parm1`	specifies a parameter that depends on the entry type (which is specified by the `class` parameter). For a script, program, or OS command `parm1` is the command string. For a form it's the corresponding XML form definition file name.

	For an output file from an external application, it's the file extension filter(s) (in **BROWSE** window, `parm1` provides the value for **FILES OF TYPE**). To include multiple file filters, separate them using plus signs (+), e.g., `parm1="doc+ppt+xls"`. For a preset image `parm1` is either a URL or a path.
`keyword`	specifies one or more keywords that get associated by default with each logbook entry of the corresponding type (`class` attribute, above) when the entry is created. Specify in a format like the following: `keyword="**KEYWORD1,*KEYWORD2,KEYWORD3"`, with or without asterisks, as desired. Keywords and asterisks are described in section 16.2.3 *Keyword*.
`maillist`	If this attribute is set, then every time a form entry is archived, a message containing the form in html format gets sent to one or more recipients. `maillist` is used with form type entries only, and specifies the recipient(s) in one of the following ways: a. An email address (which may be an individual address or a mail list address) in the format `<name>@<domain>`, e.g., `xyz-users@fnal.gov`. b. The filename of a file containing a mail list (see section 11.5.2 *Enable Automatic Electronic Mailing of Form Entries*). If there is no `@` in the value specified, **CRL** looks for a mail list file. The file must be located in the directory specified by the parameter `Logbook.file_location.mail_list_directory` set in the properties file.
`mailfrom`	is used with form type entries to specify the sender of the emailed form. It is a text string used as the "from" line in all email messages generated by the form, unless a `<FROM>...</FROM>` line is specified in the maillist file used, the latter taking precedence. `Mailfrom` must be a valid email address (which may be an individual address or a mail list address) in the format `<name>@<domain>`, e.g., `xyz-users@fnal.gov`.
`mailsubject`	is used with form type entries. It is a text string used as the subject line in all email messages generated by the form, unless a subject is included in the maillist file used, the latter taking precedence. The `mailsubject` string may contain the substrings `$T`

and/or `$D` which get replaced by the time and date, respectively, in the formats HH:MM (24 hr clock) and MM.DD.YYYY, e.g., `mailsubject= "muon update $D $T"` generates the subject: `muon update 01.27.2002 17:32`.

`command1` is not currently used.

```
<EntryInputPage class="class1" title="title1" tooltip="tooltip1">
  ...
  <ToolBar>
     <ToolButton class="logbook.logentry.LogEntryText" tip="tip1"
                image="image1.gif" keyword="KEYWORD1"
     />

     <ToolButton class="logbook.logentry.LogEntryForm" tip="test form"
                text="test form" parm1="TestObjectsForm.xml"
                maillist="mylistfile.txt" mailsubject="Test Form from CRL"
                mailfrom="owlshifters@fnal.gov"
     />

     <ToolButton ... />
     ...

  </ToolBar>
  ...
</EntryInputPage>
```

16.3 The CRL Java Class Libraries

16.3.1 Container (Topic) Classes

Standard Input Container	logbook.MyInternalTopicFrame
Autoscheduled Input Container	logbook.MyInternalScheduledTopicFrame
Thread Container	logbook.logthreads.MyInternalThreadFrame
Report Container	logbook.MyInternalSearchFrame
Checkpoint Container	logbook.MyInternalCheckpointFrame

16.3.2 Entry type Classes

Text	`logbook.logentry.LogEntryText` Used for (formattable) text insertion via keyboard. Allows text formatting. Allows insertion of images, date and time, and/or text from a file.
Plain Text	`logbook.logentry.LogEntryPlainText` Used for text insertion via keyboard. No formatting, no insertion of images; allows insertion of date and time, and/or plain text from a file.
Output file from external application	`logbook.logentry.LogEntryRoot` Used to insert output file from virtually any external application, and having any virtually any file extension. Allows insertion of (formattable) text, date and time, and/or text from a file. (This class was originally created for ROOT output files, hence its name. It has been generalized since to allow all file types).
Program out-put (text)	`logbook.logentry.LogEntryExec` Used to insert output from virtually any OS command output (DOS or UNIX). Allows insertion of (formattable) text, date and time, and/or text from a file.
Form	`logbook.logentry.LogEntryForm` Used to insert a form. A separate toolbutton of this class is required for each form. Allows insertion into form text fields of plain text, date and time, and/or text from a file.
IPen® free-hand drawing	`logbook.logentry.LogEntryIpen` Used to insert a free-hand sketch or writing done on an Ipen® tablet and saved as a gif file.
Preconfigured image	`logbook.logentry.LogEntryImage` Inserts a particular image file of type gif or jpg into the entry it creates. Allows insertion of (formattable) text, date and time, and/or text from a file.

It is possible to define additional classes for data input; request information from *crl-dev@fnal.gov.*

16.3.3 Desktop Page Classes

Entry Input Page	logbook.Page_EntryInput
Report Page (non-entry-input)	logbook.Page_General

Chapter 17: Form and Inquiry Configuration Files

The configuration files for the inquiry and forms features are provided in **XML** format. In this chapter we illustrate the structure of these configuration files.

We recommend that you read and understand the material in Chapter 15: *Introduction to XML and DTD Files* before delving into the material presented here. Also see Chapter 11: *Configuring and Launching CRL* for a task-oriented approach to configuring the application; it refers you to task-appropriate sections in the current and other chapters.

Some of the XML configuration files listed in this chapter are provided with **CRL** as samples. They may be used as is, edited, copied, or ignored!

17.1 Form Definition Files

The Logbook form definition files govern the form entry types. These files must be located in a directory defined by the `Logbook.file_location.forms_directory` parameter in the properties file (see Chapter 14: *CRL's Java Properties*). The `form.dtd` file must also be present in that directory. The **CRL** administrator can create new forms and add them to the same directory, as long as the XML code contains only valid elements. See section 11.5 *Creating Configuration Files for Forms*.

17.1.1 XML Elements Allowed in Forms

Unlike the other configuration files in CRL, the form definition XML files are not strictly governed by the DTD file, but they do reference it. We have turned off case-sensitivity for form elements and attributes. For consistency and completeness, we list the valid elements in the form of a DTD file. See section 15.3 *Element Types and Attributes in the DTD File* to understand the relationship between DTD and XML files.

```
<?xml version="1.0" encoding="UTF-8" ?>

<!ELEMENT Form ( Line+, REPEATBLOCK*, insertform* ) >
<!ATTLIST Form name CDATA #IMPLIED >
<!ATTLIST Form reload NMTOKEN #IMPLIED >
```

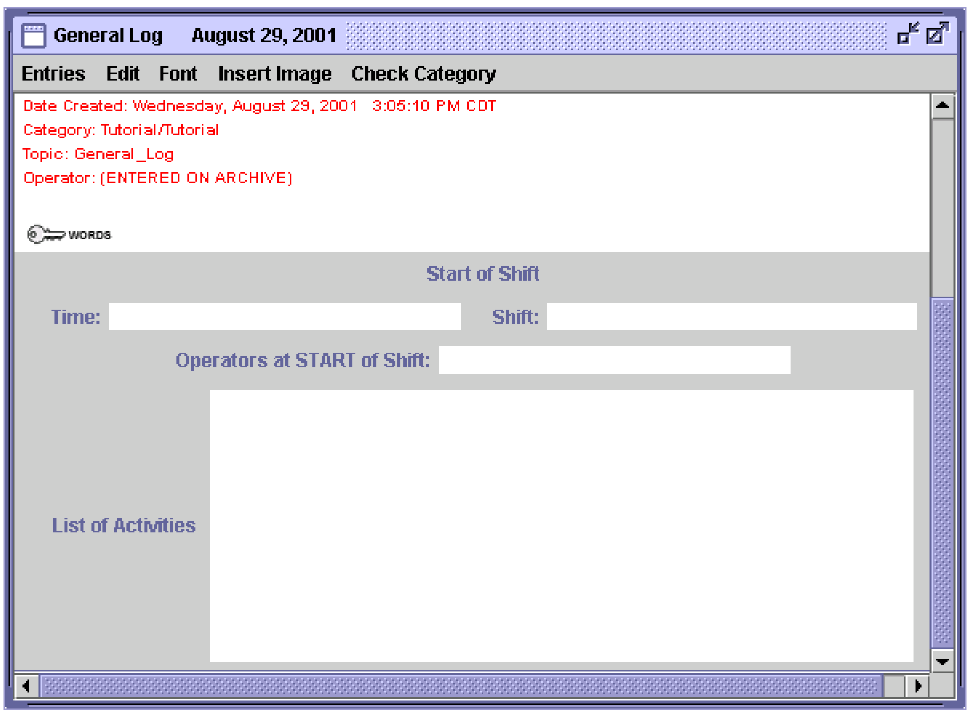
```
<!ELEMENT insertform EMPTY >
<!ATTLIST insertform name NMTOKEN #REQUIRED >
<!ATTLIST insertform reload NMTOKEN #IMPLIED >
<!ATTLIST insertform byReference NMTOKEN #IMPLIED >

<!ELEMENT Line ( Text*, Field*, DateAndTime*, CheckBox*, RadioButtonGroup*, Select*,
List*, TABLE*, EXEC* ) >
<!ATTLIST Line align NMTOKEN #IMPLIED >

<!ELEMENT Text ( #PCDATA ) >

<!ELEMENT CheckBox EMPTY >
<!ATTLIST CheckBox Name CDATA #REQUIRED >
<!ATTLIST CheckBox checked NMTOKEN #REQUIRED >

<!ELEMENT DateAndTime EMPTY >
<!ATTLIST DateAndTime Date NMTOKEN #REQUIRED >
<!ATTLIST DateAndTime Time NMTOKEN #REQUIRED >

<!ELEMENT List ( Text*, Item* ) >
<!ATTLIST List Multiple NMTOKEN #REQUIRED >

<!ELEMENT Item EMPTY >
<!ATTLIST Item Name CDATA #REQUIRED >

<!ELEMENT Field EMPTY >
<!ATTLIST Field Columns NMTOKEN #IMPLIED >
<!ATTLIST Field Rows NMTOKEN #IMPLIED >

<!ELEMENT Option EMPTY >
<!ATTLIST Option Name CDATA #REQUIRED >

<!ELEMENT Select ( Text*, Option* ) >
<!ATTLIST Select Editable NMTOKEN #IMPLIED >

<!ELEMENT RadioButtonGroup ( Text*, RadioButton* ) >
<!ELEMENT RadioButton EMPTY >
<!ATTLIST RadioButton Name NMTOKEN #REQUIRED >
<!ATTLIST RadioButton checked NMTOKEN #REQUIRED >


<!ELEMENT TABLE ( ColumnLabel* ) >
<!ATTLIST Field width CDATA #IMPLIED >
<!ATTLIST Field height CDATA #IMPLIED >

<!ELEMENT EXEC EMPTY >
<!ATTLIST Program CDATA #REQUIRED >

<!ELEMENT ColumnLabel ( DateAndTime*, CheckBox*, Integer*, Double*, Select*, Field* )
>
<!ATTLIST ColumnLabel name CDATA #REQUIRED >

<!ELEMENT Integer EMPTY >

<!ELEMENT Double EMPTY >

<!ELEMENT REPEATBLOCK ( Line* ) >
```

17.1.2 Sample “Start of Shift” Form

```
<?xml version="1.0"?>
```

```
<!DOCTYPE Form SYSTEM "form.dtd">
<Form name="Start of Shift">
      <Line>
         <Text>Time:</Text>
         <Field Rows="1" Columns="20" />
         <Text>     Shift:</Text>
         <Field Rows="1" Columns="21" />
      </Line>
      <Line>
         <Text>Operators at START of Shift:</Text>
         <Field Rows="1" Columns="20" />
      </Line>
      <Line>
         <Text>List of Activities </Text>
         <Field Rows="10" Columns="40" />
      </Line>
</Form>
```

This XML file produces the following **START OF SHIFT** form:

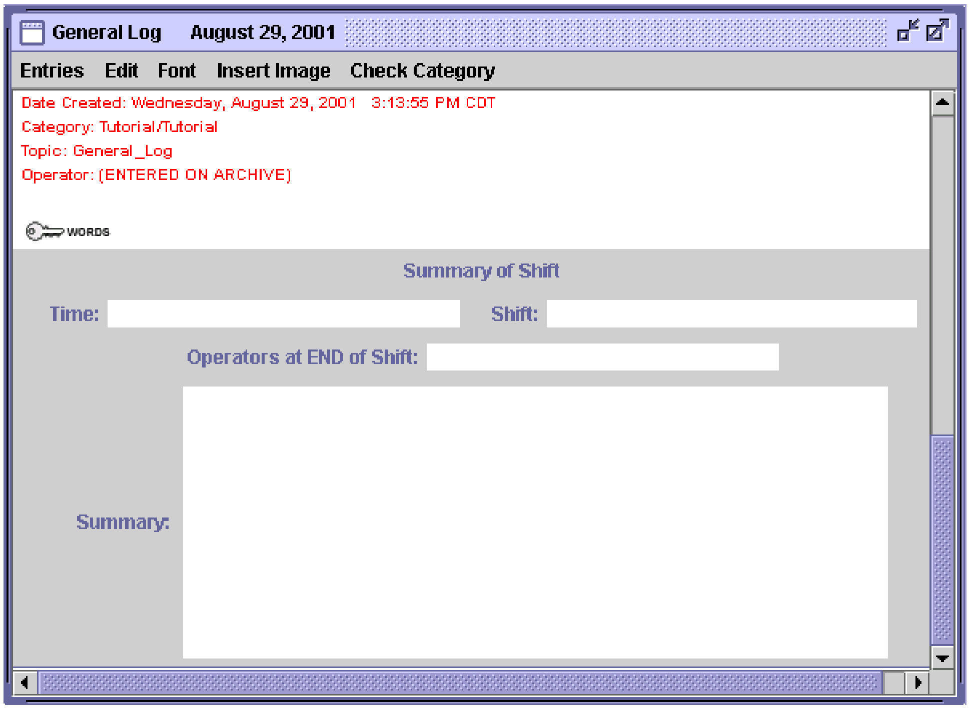


17.1.3 Sample “End of Shift” Form

```
<?xml version="1.0"?>
<!DOCTYPE Form SYSTEM "form.dtd">
<Form name="Summary of Shift">
      <Line>
```

```
            <Text>Time:</Text>
            <Field Rows="1" Columns="20" />
            <Text>     Shift:</Text>
            <Field Rows="1" Columns="21" />
         </Line>
         <Line>
            <Text>Operators at END of Shift:</Text>
            <Field Rows="1" Columns="20" />
         </Line>
         <Line>
            <Text>Summary: </Text>
            <Field Rows="10" Columns="40" />
         </Line>
</Form>
```

This XML file produces the following **SUMMARY OF SHIFT** form:



17.1.4 Sample Form using a Variety of Form Elements

This example shows how to implement list boxes, select boxes (dropdown lists), check boxes, radio buttons, program output and date and time in your forms. This file does not come with the default configuration.

Insert the initial lines, plus Form name:

```
<?xml version="1.0"?>
<!DOCTYPE Form SYSTEM "form.dtd">
<Form name="Start of Run">
```

Insert two Date/Time fields; one for date only and one for time only:

```
<Line align="left">
   <Text>Date:</Text>
   <DateAndTime Date="yes" Time="no" />
   <Text>  Time:</Text>
   <DateAndTime Date="no" Time="yes" />
```

On the same line, insert a select box for Shift:

```
   <Select>
      <Text>  Shift:   </Text>
      <Option Name="Day" />
      <Option Name="Evening" />
      <Option Name="Night" />
   </Select>
</Line>
```

On the next line, insert an editable select box:

```
<Line align="left">
   <Select Editable="yes">
      <Text>Select Box:   </Text>
      <Option Name="John Q. Smith" />
      <Option Name="Pierre Auguste Renoir" />
      <Option Name="Claude Monet" />
      <Option Name="Edouard Manet" />
      <Option Name="Edgar Degas" />
   </Select>
```

On the same line, insert two radio buttons, and set one to a default initial value of “on”, the other to “off”; user can select only one of the set:

```
   <RadioButtonGroup>
      <Text>     Radio Button:   </Text>
      <RadioButton Name="ON" checked="on" />
      <RadioButton Name="OFF" checked="off" />
   </RadioButtonGroup>
</Line>
```

Insert some more check boxes, and set each to a default initial value of “on” or “off”:

```
<Line align="left">
   <Text>Checkbox:   </Text>
   <CheckBox Name="Fast IO" checked="on" />
   <CheckBox Name="Fancy Graphics" checked="on" />
   <CheckBox Name="High Bandwidth" checked="off" />
</Line>
```

Insert a text box:

```
<Line align="left">
   <Text>Text field, 1 row 10 columns:</Text>
   <Field Rows="1" Columns="10" />
```

```
      </Line>
```

Insert a list box:

```
      <Line align="left">
         <List Multiple="yes">
            <Text>List Box (Impressionists):   </Text>
            <Item Name="Van Gogh"/>
            <Item Name="Renoir"/>
            <Item Name="Gaugin"/>
            <Item Name="Monet"/>
         </List>
      </Line>
```

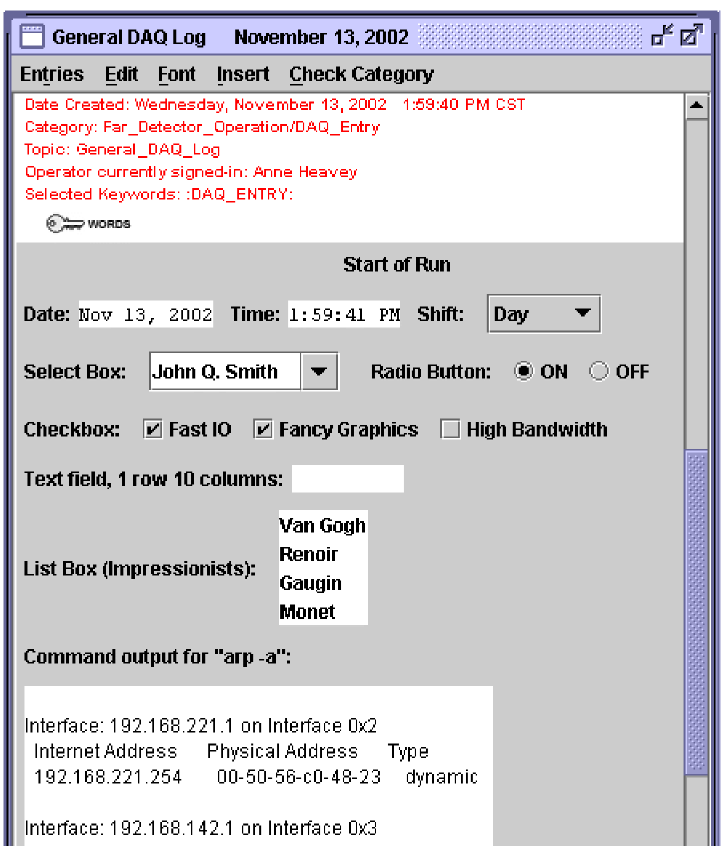
Run a program and insert the (text) output into a text area which sizes itself automatically (insert a header line first):

```
      <Line align="left">
         <Text>Command output for "arp -a":</Text>
      </Line>
      <Line align="left">
         <exec Program="arp -a" />
      </Line>
```

Close form:

```
   </Form>
```

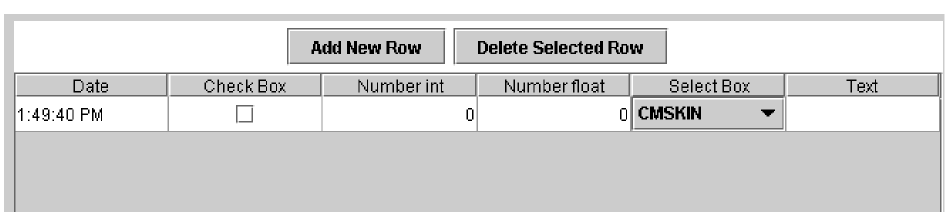
This XML file produces the following **Start of Run** form:



17.1.5 Sample Table Element in a Form

Note that the table element is contained within a line element. The table height and width default to 650 and 150, respectively.

```
<Line>
  <Table height="50" width="200">
    <ColumnLabel name="Date">
      <DateAndTime Date="yes" Time="no" />
    </ColumnLabel>
    <ColumnLabel name="Check Box">
      <CheckBox checked="on" />
    </ColumnLabel>
    <ColumnLabel name="Number int">
      <Integer/>
    </ColumnLabel>
    <ColumnLabel name="Number float">
      <Double/>
    </ColumnLabel>
    <ColumnLabel name="Select Box">
      <Select Editable="yes">
        <Option Name="CMSKIN"/>
        <Option Name="CMSIM"/>
        <Option Name="OOHIT"/>
        <Option Name="OODIGIS"/>
        <Option Name="NTUPLE"/>
      </Select>
    </ColumnLabel>
    <ColumnLabel name="Text">
      <Field/>
    </ColumnLabel>
  </Table>
</Line>
```



17.1.6 Sample Line for Running a Script or Command

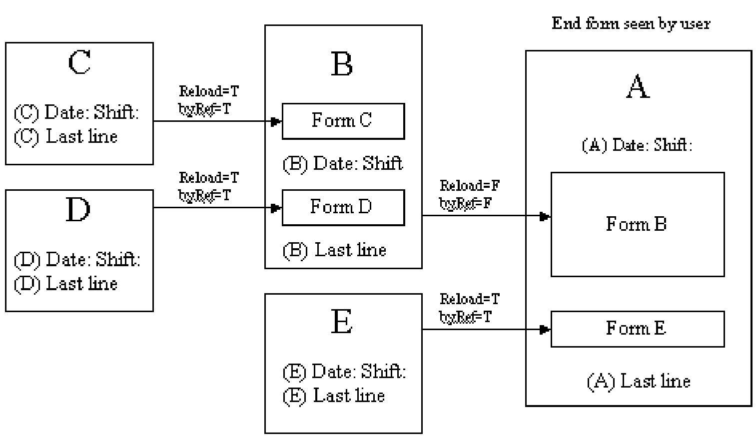
You can run a program within a form using the `EXEC` element with its `Program` attribute. The program can be virtually any type of script (e.g., a Python script), or an OS command. In this example, we execute the command **`arp -a`**:

```
<?xml version="1.0"?>
<!DOCTYPE Form SYSTEM "form.dtd">
<Form name="...">
    ...
    <Line>
        <EXEC Program="arp -a" />
    </Line>
    ....
</Form>
```

The form automatically sizes the field to accommodate the output.

17.1.7 Sample Form with Embedded Forms

These sample forms provide details for the example discussed in section 11.5.4 *Create Forms with Selected Reloadable Fields*. We recommend that you read that section first. The schematic below illustrates our example:



Form A (The End Form)

This form has two native elements, the Date/Time and Shift line and the LAST LINE. Inbetween these two elements, it inserts two forms, b.xml and e.xml (for B and E), the former with reload and byReference false, and the latter with reload and byReference true.

```
<?xml version="1.0"?>
<!DOCTYPE Form SYSTEM "form.dtd">
```

```
<Form name="AAA">
  <Line>
    <Text>Date:</Text>
    <DateAndTime Date="yes" Time="no"/>
    <Text>     Shift:</Text>
    <Field Rows="1" Columns="21" />
  </Line>
<insertform name="b.xml" reload="false" byReference="false"/>
<insertform name="e.xml" reload="true" byReference="true"/>
  <Line>
    <Text>LAST LINE OF AAA</Text>
    <Field Rows="1" Columns="20" />
  </Line>
</Form>
```

Form B (b.xml)

This form has two native elements, the Date/Time and Shift line and the LAST LINE. Above and between these two elements, it inserts two forms, c.xml and d.xml (for C and D), both with reload and byReference true. Form B gets inserted into A with reload and byReference both false.

```
<?xml version="1.0"?>
<!DOCTYPE Form SYSTEM "form.dtd">
<Form name="BBB">
<insertform name="c.xml" reload="true" byReference="true"/>
  <Line>
    <Text>Date:</Text>
    <DateAndTime Date="yes" Time="yes"/>
    <Text>     Shift:</Text>
    <Field Rows="1" Columns="21" />
  </Line>
<insertform name="d.xml" reload="true" byReference="true"/>
  <Line>
    <Text>LAST LINE OF BBB</Text>
    <Field Rows="1" Columns="20" />
  </Line>
</Form>
```

Form C (c.xml)

This form has two native elements only, the Date/Time and Shift line and the LAST LINE. It gets inserted into B with reload and byReference both true.

```
<?xml version="1.0"?>
<!DOCTYPE Form SYSTEM "form.dtd">
<Form name="CCC">
  <Line>
    <Text>Date:</Text>
    <DateAndTime Date="yes" Time="yes"/>
```

```
      <Text>     Shift:</Text>
      <Field Rows="1" Columns="21" />
    </Line>
    <Line>
      <Text>LAST LINE OF CCC</Text>
      <Field Rows="1" Columns="20" />
    </Line>
  </Form>
```

Form D (d.xml)

This form has two native elements only, the Date/Time and Shift line and the LAST LINE. It gets inserted into B with reload and byReference both true.

```
<?xml version="1.0"?>
<!DOCTYPE Form SYSTEM "form.dtd">
<Form name="DDD">
  <Line>
    <Text>Date:</Text>
    <DateAndTime Date="yes" Time="yes"/>
    <Text>     Shift:</Text>
    <Field Rows="1" Columns="21" />
  </Line>
  <Line>
    <Text>LAST LINE OF DDD</Text>
    <Field Rows="1" Columns="20" />
  </Line>
</Form>
```

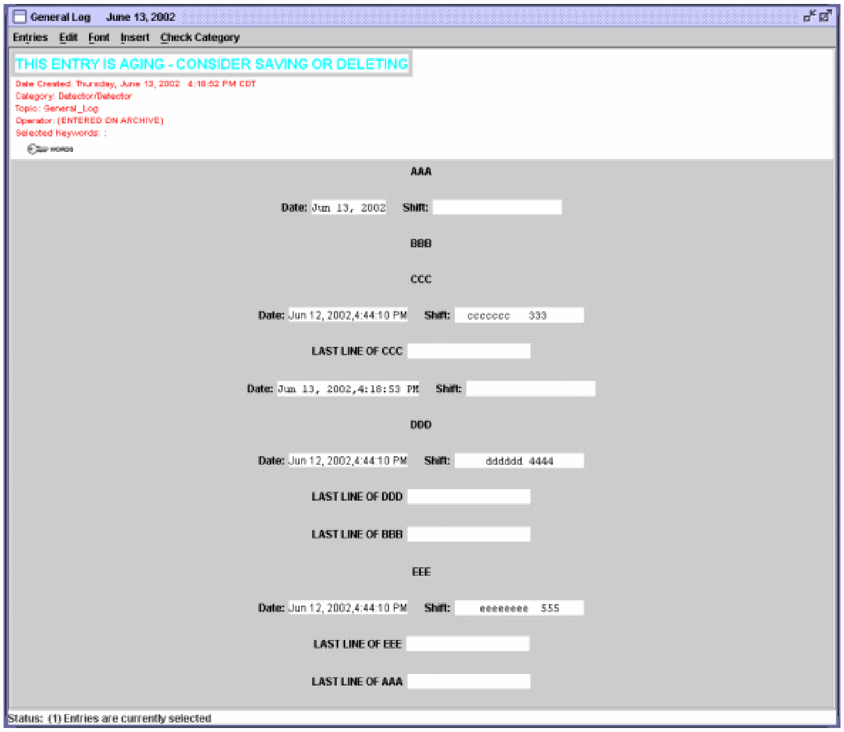
Form E (e.xml)

This form has two native elements only, the Date/Time and Shift line and the LAST LINE. It gets inserted into A with reload and byReference both true.

```
<?xml version="1.0"?>
<!DOCTYPE Form SYSTEM "form.dtd">
<Form name="EEE">
   <Line>
    <Text>Date:</Text>
    <DateAndTime Date="yes" Time="yes"/>
    <Text>     Shift:</Text>
    <Field Rows="1" Columns="21" />
  </Line>
  <Line>
    <Text>LAST LINE OF EEE</Text>
    <Field Rows="1" Columns="20" />
    </Line>
</Form>
```

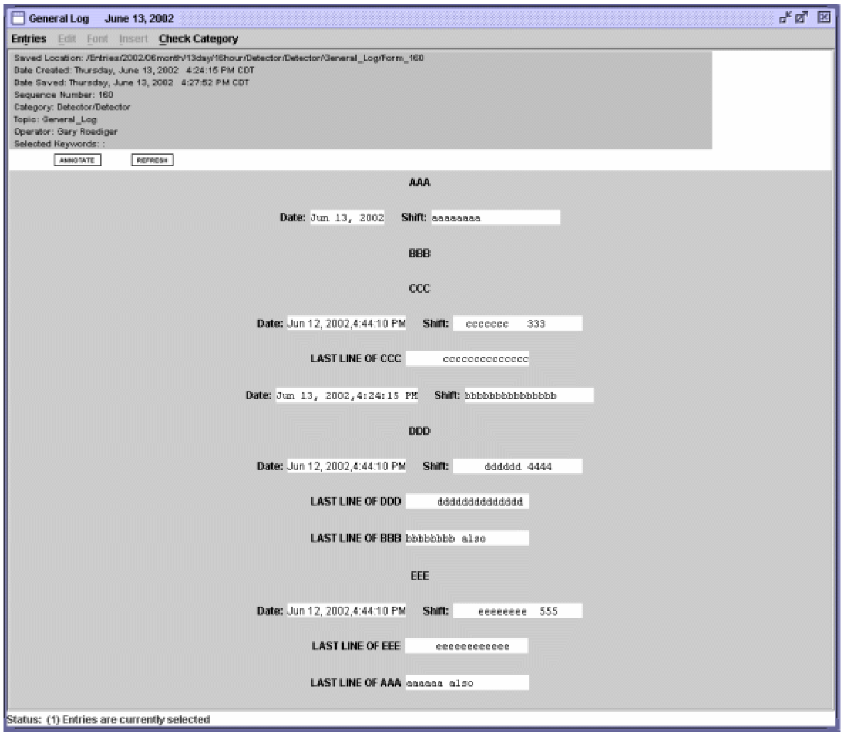
How does this look to the user?

First, user creates a logbook entry of the form A type on June 13 at 4:18 p.m. Initially, it looks like this (ignore the aging message):

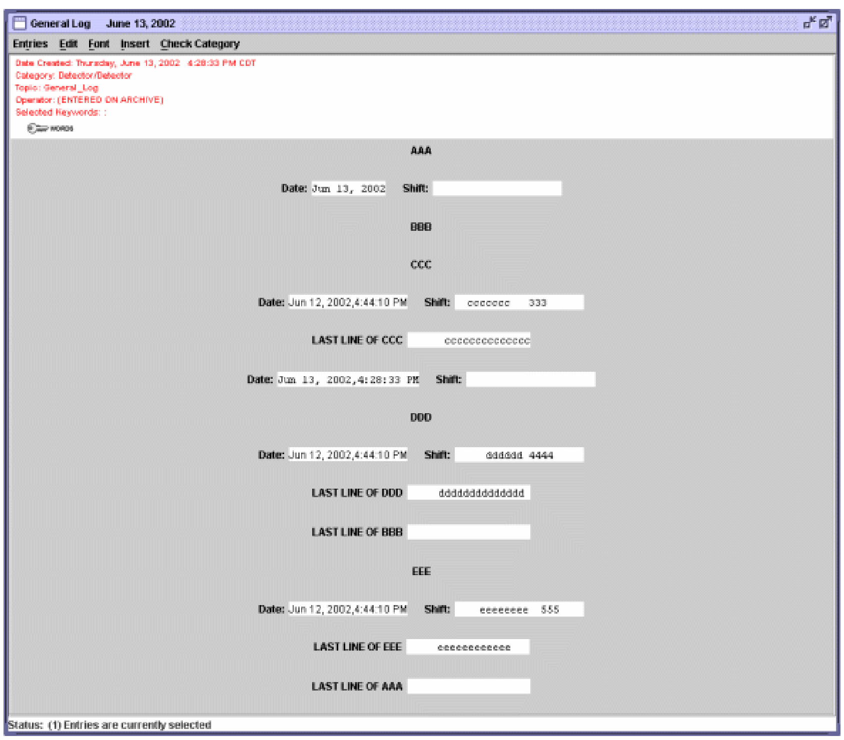


Look at the date/time entries. Forms A and B have the current date/time (B was inserted with reload=false). Forms C, D, and E have an earlier time (they were each reloaded upon insertion into target form).

The user now edits the fields (leaving the date/time fields alone), and archives. The form now looks like this:



Now, user creates another form A. It looks like this initially:



Notice that A and B again reflect an updated time (well, you can't actually tell for A since the time doesn't show) and that the other A and B fields are cleared. E, C, and D all wrote their values to the reload area upon archive (byReference=true) and reloaded the new information (reload=true) into A or B.

17.2 The Logbook Inquiry Configuration File

The Logbook inquiry configuration file governs the fields on which you can query when using the inquiry feature described in section 8.2 *Inquiries*.

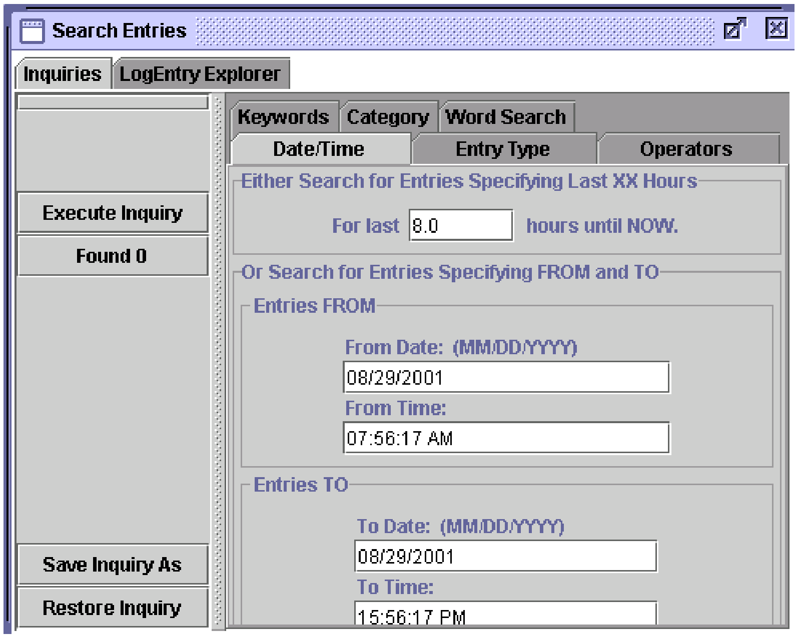
17.2.1 DTD File Listing for Inquiries

```
<?xml encoding="US-ASCII"?>
<!ELEMENT LogBookReports (filter*)>
<!ELEMENT filter EMPTY>
<!ATTLIST filter
     class CDATA #REQUIRED
     name CDATA #REQUIRED
     type CDATA #REQUIRED>
```

17.2.2 Default XML File Listing for Inquiries

```
<?xml version="1.0"?>
<!DOCTYPE LogBookInquiryConfig SYSTEM "LogBookInquiryConfig.dtd">
<LogBookReports >

<filter type="entry" name="Operators" class="logbook.logdbms.FilterOperator"/>
<filter type="entry" name="Entry Type" class="logbook.logdbms.FilterEntryType"/>
<filter type="entry" name="Date/Time" class="logbook.logdbms.FilterDate"/>
<filter type="entry" name="Keywords" class="logbook.logdbms.FilterKeyword"/>
<filter type="entry" name="Category" class="logbook.logdbms.FilterCategory"/>
<filter type="entry" name="Word Search" class="logbook.logdbms.FilterWordSearch"/>

</LogBookReports >
```

Notice that the names in the above code (e.g., `name="Operators"`) match the search categories in the **INQUIRIES** window:


Search Entries
Inquiries
LogEntry Explorer
Execute Inquiry
Found 0
Save Inquiry As
Restore Inquiry
Keywords
Category
Word Search
Date/Time
Entry Type
Operators
Either Search for Entries Specifying Last XX Hours
For last
8.0
hours until NOW.
Or Search for Entries Specifying FROM and TO
Entries FROM
From Date: (MM/DD/YYYY)
08/29/2001
From Time:
07:56:17 AM
Entries TO
To Date: (MM/DD/YYYY)
08/29/2001
To Time:
15:56:17 PM

Index

Symbols

#
    in properties file 14-1
*
    keyword 16-4
    operator on XML element content 15-3
**
    keyword 16-4
+
    operator on XML element content 15-3
/
    in properties file 14-1
?
    operator on XML element content 15-3

A

About LogBook button 1-4
Activate User button 12-3
activating users 12-3
Add Selected Entries to a Thread menu option 8-19
Add User button 12-2
adding users 12-2
administrative password 12-1
administrative tasks
    activating a user 12-1, 12-3
    adding a user 12-1, 12-2
    editing user information 12-1, 12-4
Administrator button 1-3, 12-1
AFS
    launching CRL from 2-2, 11-5
    prep on local machine 10-9
aging entries 5-3
annotate archived entry 8-13
    from web 8-15
Annotate button in archived entry 8-13
annotations 1-12
    set directory for storing 14-4
anonymous entries 14-2
Apache Web Server 10-2
    installation 13-1
application output
    including in logbook entry 3-8
    viewing thumbnail 3-8
application output files
    entry type class 16-16
Archive All non-archived Entries in this Topic menu option 5-2
Archive Selected Entries menu option 5-2
archived entries
    annotate 8-13, 8-15
    search 1-3
    search via Inquiries 8-1
    search via LogEntry Explorer 8-1, 8-8
    search via web 8-10
    set directory for storing 14-1
archiving 1-12
    and threading entry 5-1
    entry signers 5-1
    form entry features 5-1
    reminders 5-3
    setting operator name 5-1
attaching messages to entries 1-4, 3-11
attributes in DTD file 15-4
auto-linked keyword 4-7
automating logbook entry 6-1
autoscheduled container 6-2
    class 16-16
    definition 1-8
autoscheduled logbook entry
    description 6-1
    parameters 6-2
    terminate job 6-3

B

binary files
    as logbook entry 3-8
    export from report 8-18
    set export directory 14-4
browse paths 1-11
    configuring 14-1, 14-3
    locating files to insert in entries 3-3

C

Cancel Inquiry button 8-8
category
    checking 3-2
    define 11-7
    definition 1-7
    keyword config (category type) 16-4
    selecting from pull-down menu 3-1
Category filter for Inquiries 8-6
Change Active User button 12-4
change administrative password 12-1
Change Password button 1-3
changing your password 2-7
character set for writing XML, define 14-1
Check Category menu option 4-3
checkpoint 1-12
    configure interval 14-2
    delete entry 4-7, 4-10
    recover entry 4-10
checkpoint container 1-9
    class 16-16
    editing an entry 4-11
    menu options 4-11
    purging 4-11

Checkpoint Recovery button 1-4, 4-7, 4-10
class libraries for CRL 16-15
command output as logbook entry 3-10
command, executing in form 17-8
configuration files
- * operator 15-3
- + operator 15-3
- ? operator 15-3
- class libraries 15-4, 16-15
- default inquiry XML file listing 17-14
- DTD
  - attribute lists 15-2, 15-4, 16-1
  - attributes 15-4
  - element type content and operators 15-3
  - element type declarations 15-3
  - element types 15-2, 16-1
  - sample attribute list 15-5
  - sample element type declaration 15-5
- element types
  - EntryInputPage 16-7
  - InitialTabs 16-5
  - Keyword 16-4
  - LogBookConfig 16-3
  - Menu 16-8
  - Page 16-6
  - SubMenu 16-8
  - ToolBar, LogBook entry 16-12
  - ToolButton 16-12
  - Topic 16-9
- element types (for forms) 17-1
- element types (for inquiries)
  - filter 17-14
  - LogBookReports 17-14
- form objects, sample XML 17-4
- forms 17-1
- LogBookConfig.dtd 15-1
  - file listing 16-1
- LogBookConfig.xml 15-1
- LogBookInquiryConfig.dtd 15-1
  - file listing 17-14
- LogBookInquiryConfig.xml 15-1
- table element in form, sample XML 17-7
- XML
  - See XML

configuring CRL
- define keywords 11-6
- define new desktop page 11-7
- define new forms 11-8
- define new menu 11-7
- define toolbar 11-8
- define toolbutton images 11-8
- define toolbuttons 11-8
- desktop 11-6
- JVM installation, preinstalled 11-2

configuring web access 13-1
container
- (See input, report, thread, autoscheduled, checkpoint container)
- checkpoint 1-9
- define topic for 11-7
- definition 1-7
- Edit menu 4-5
- input 1-7, 1-8
- menu options (checkpoint container) 4-10, 4-11
- menu options (input container) 4-1
- menu options (report container) 8-16
- menu options (thread containers) 7-2
- report 1-9, 8-1, 8-16
- scheduled 1-8, 6-2
- select multiple entries 4-3
- select single entry 4-3
- status bar 1-7
- thread 1-9, 7-2
- verifying category 3-2

container class
- autoschedule 16-16
- checkpoint 16-16
- input 16-15
- report 16-16
- thread 16-16

container size
- configure default 16-11
- setting 3-3

control key shortcuts 1-12
conventions, notational 1-3
copy text from VT100 window 4-5
copy/cut/paste/ 4-5
copy-to-clipboard menu option 4-5
creating a report 8-16
creating keywords 11-6
CRL
- components 10-2
  - installed via InstallShield 10-3
- configure desktop 11-6
- define new forms 11-8
- desktop 1-2
  - description 1-5
- desktop page 1-5
- development overview 1-1
- distribution platforms 10-1
- feature overview 1-2
- Fermi AFS installation 11-5
- installation requirements 10-2
- installing from tar file 10-9
- installing on Windows 10-6
- installing via UPD 10-9
- invocation script, modifying 11-4
- invocation script, sample 11-3
- invoking application 11-4
- launch from AFS 2-2
- launch from Linux 2-1
- launch from Windows 2-1
- launching application 11-4
- parts of the application 1-1
- toolbar 1-2
  - description 1-3
- uninstalling 10-11
- use with preinstalled JVM 11-2
- workspace 1-5

CRL class libraries 16-15
CRL configuration file 11-6, 16-1
CRL Java properties 14-1
CRL properties file
- (See properties file)

CRL toolbar buttons
- About LogBook 1-4
- Administrator 1-3
- Change Password 1-3
- Checkpoint Recovery 1-4
- Current Situation 1-4

Entry Signers 1-3
Exit LogBook 1-5
Log In/Log Out 1-3
Preferences 1-4
Scheduler 1-4
Search 1-3
Thread Explorer 1-4
CRL Web Access
See web access
Cross Ipen® support 3-9
Ctrl-Alt-D for date&time 1-11, 3-10
Current Situation 3-11
changing message 3-12
turning off message 3-12
turning on message 3-11
Current Situation button 1-4
cut/copy/paste 4-5
cut-to-clipboard menu option 4-5

D

data entry
binary files 3-8
command output 3-10
form 3-4
freehand drawings 3-9
general info 3-3
images 3-6
output from applications 3-8
program output 3-10
ROOT plots 3-8
text from keyboard 3-4
data entry type 1-11
configuring 16-13
supported types 3-3
data loss
protection against 4-10
data types for logbook entry 1-10
database
information in properties file 11-1
queries (see Inquiries)
requirement for running CRL 10-2
set dbms name in properties file 14-6
set driver in properties file 14-5
set location in properties file 14-5
set pointers from XML entries to db 14-5
set protocol in properties file 14-6
set server IP address in properties file 14-6
set username in properties file 14-6
set username's password in properties file 14-6
set vendor in properties file 14-5
set version in properties file 14-5
date&time shortcut key 1-11, 3-10
Date/Time filter for Inquiries 8-2
debugging value, set 14-1
defining desktop page in CRL 11-7
defining keywords 11-6
application-wide 11-7
category/topic 11-6
data type 11-6
desktop page 11-6
input container 11-6
defining menu in CRL 11-7
defining new form in CRL 11-8
defining toolbar in CRL 11-8
defining toolbuttons in CRL 11-8
Delete Selected Entries from a Thread menu option 8-19
deleting CRL application 10-11
deleting entries 4-6
checkpoint deletion 4-7
desktop 1-2
description 1-5
menus 1-6
objects 1-5
opening input container on 3-1
desktop page 1-5
configuring 16-6, 16-7
keyword config (page type) 16-4
page tab 1-5
desktop page class
entry input page 16-17
report page 16-17
directory
annotation storage 14-4
binary file export 14-4
files to be inserted into entries 3-3, 14-3
form config files 14-3
histogram storage 14-4
HTML entry storage 14-4
inquiry storage 11-2, 14-3
logbook entry storage 11-1
mail list files 14-4
temporary dir for PS/PDF files 14-4
XML entry storage 14-3
directory for html entry storage 13-1
directory for inquiry storage 11-2
directory for logbook entry storage 11-1
display of entry header 1-4
double-click shortcuts 1-11
DTD files
attribute lists 15-2, 16-1
element type declarations 15-3
element types 15-2, 16-1
file listing
CRL application 16-1
forms 17-1
Inquiries 17-14
introduction 15-1
operators used in 15-3

E

Edit menu 4-2, 4-5
copy-to-clipboard menu option 4-5
cut-to-clipboard menu option 4-5
paste-from-clipboard menu option 4-5
Redo menu option 4-4, 4-6
Select-all menu option 4-4
Undo menu option 4-4, 4-6
editing properties file 11-1
element syntax for XML 16-2
element type declarations (XML) 15-3
email forms 3-4
configuring 11-10, 11-11
maillist files 11-10, 11-11
email id as username 2-3

embedded forms 11-9, 11-12
encapsulated PS file
as entry, configuring 11-15
entering data
binary files 3-8
command output 3-10
external program 9-1
form 3-4
freehand drawings 3-9
images 3-6
output from applications 3-8
process logger 9-1
ROOT plots 3-8
text from keyboard 3-4
Entries menu 4-2, 8-16
Add Selected Entries to a Thread menu option 8-19
Archive All non-archived Entries in this Topic menu option 5-2
Archive Selected Entries menu option 5-2
Delete Selected Entries from a Thread menu option 8-19
Print All Entries in this Topic 4-11
Print All Entries in this Topic menu option 8-18, 8-19
Print Selected Entries 4-11
Print Selected Entries menu option 8-18
Remove All Entries in this Topic menu option 4-6, 5-3, 8-18
Remove Selected Entries menu option 4-6, 5-3, 8-18
Select ALL Entries menu option 4-4
Send ALL Entries To Mail Recipient 4-11
Send ALL Entries To Mail Recipient menu option 8-18
Send Selected Entries To Mail Recipient 4-11
Send Selected Entries To Mail Recipient menu option 8-18
Un-select ALL Entries menu option 4-4
entry category definition 1-7
entry container
definition 1-7
scheduled entries, definition 1-8
See also input container
entry format description 1-10
entry header
creating 3-2
display 1-4
set preferences 1-10
entry header properties 14-3
entry input container
creating entry in 3-2
definition 1-8
opening 3-1
entry input page
configuring 11-7, 16-7
definition 1-5
desktop page class 16-17
displaying 3-1
making logbook entries on 3-3
Entry Signers 2-2, 2-4
add/remove usernames 2-3
archiving an entry 5-1
description 2-3
list 2-3
Operator field of entry 2-3
Entry Signers button 1-3
entry topic
configuring 11-7, 16-9
input type classes 16-11
definition 1-7
selecting for use 3-1
entry type class
form 16-16
image 16-16
IPen 16-16
OS command 16-16
output from external app 16-16
text 16-16
Entry Type filter for Inquiries 8-3
entry type, selecting 3-2
Execute Inquiry button 8-7
Exit LogBook button 1-5
export of binary files
export directory 14-4
from report 8-18
external application output in entry 3-8
external application, copy content
non-X 4-5
X app 4-5

F

features of CRL 1-2
features of logbook entries 1-11
file name for thread 7-1
filter entries 14-2
filters for logbook entry search
Category 8-6
Date/Time 8-2
Entry Type 8-3
Keywords 8-5
Operators 8-4
Word Search 8-6
find archived entries 8-1
flpr 11-17
Font menu 4-2, 4-6
font size
of logbook entry text 4-6
setting the default 14-1
form entry
actions on archive 5-2
configuration 11-8
alignment 11-9
element types 11-9
email completed form 11-9, 11-10
embedded forms 11-9, 11-12
helpful hints 11-15
Line tag rules 11-9
reload selected fields 11-9, 11-12
repeat blocks 11-10
tables 11-9, 11-11
toolbutton 11-9
configuration for embedded forms 17-8
configuration for script/command execution 17-8
configuration sample 17-4
configure destination mail address 16-14
configure mail subject 16-15
configure sender of emailed form 16-14
configuring automatic email feature 11-10, 11-11
creating 3-4

default XML file for end of shift form 17-3
default XML file for start of shift form 17-3
DTD file 17-1
embedded forms 11-9, 11-12
entering data 3-4
program execution within forms 3-5
program/script execution 11-9
reloadable fields 11-9, 11-12
reloading data 3-6
repeatable blocks 3-5
saving data for reload 5-2
script execution within forms 3-5
send to mail list 5-2
set directory for storing 14-3
specify XML config file 16-14
tables within forms 3-6
form objects 17-4
Found "n" button 8-7
freehand drawing
creating entries 3-9
function key shortcuts 1-12

G

global messages 1-4, 1-11, 3-11
changing 3-12
turning off 3-12
turning on 3-11

H

halt inquiry during execution 8-8
header for logbook entry 1-10
display 1-4
set preferences 1-4
histograms
including in logbook entry 3-8
set directory for storing 14-4
viewing thumbnail 3-8
host for email, configuring 14-4
HTML
comparison to XML 15-1
configure storage directory for HTML archives 14-4
entry storage 1-1
define directory 13-1

I

image entry 3-6
configure max image size 14-2
from file 3-8
from toolbutton 3-8
from URL 3-8
image file
entry type class 16-16
file types 3-6
specifying image file 16-14
imagemagick executable path, configuring 14-5
images
configure max image size 14-2
images, thumbnail 3-6
for PS/PDF files, configuring 11-15
input container
adding entries to thread 7-6
Archive All non-archived Entries in this Topic menu option 5-2
Archive Selected Entries menu option 5-2
Check Category menu option 4-3
class 16-15
copy-to-clipboard menu option 4-5
creating a thread 7-4
cut-to-clipboard menu option 4-5
definition 1-7, 1-8
Edit menu 4-2
editing an entry 4-1
Entries menu 4-2
Font menu 4-2, 4-6
Insert menu 4-2
inserting logbook entry 3-2
keywords 4-7
menu options 4-1
opening 3-1
paste-from-clipboard menu option 4-5
Remove All Entries in this Topic menu option 5-3
Remove Selected Entries menu option 5-3
Select ALL Entries menu option 4-4
shortcut for adding date/time 3-10
shortcut for creating entry 3-3
short-cut keys 3-3
Un-select ALL Entries menu option 4-4
verifying category 3-2
verifying category and topic 4-3
Inquiries 8-1
cancelling 8-8
Category filter 8-6
configuration 11-16
configuration files 17-14
creating 8-2
Date/Time filter 8-2
description 8-1
directory for storing 14-1, 14-3
editing 8-8
Entry Type filter 8-3
executing 8-7
Keywords filter 8-5
Operators filter 8-4
restoring 8-8
saving 8-8
web-based 8-9
Word Search filter 8-6
insert form entry 3-4
Insert menu 4-2
Date &Time 3-10
image from file 4-2
image from URL 4-2
replace text by program output 3-4
text from file 4-2
text from program output 3-4
insert PostScript/PDF file 3-8
insert text from program 3-10
installation requirements 10-1
installing CRL 10-1
components 10-2
from tar file 10-9
on Linux 10-9

on Windows 10-6
requirements 10-2
supported OS 10-1
via UPD 10-9
InstallShield Multi-Platform Edition 10-1
interface description 1-2
internet access
See web access
invoking CRL 2-1, 11-4
from Linux 2-1
from Windows 2-1
in AFS 2-2
IPen file
entering into logbook 3-9
entry type class 16-16
Ipen functionality, enabling 11-2

J

Java 1-1, 10-1, 10-2
java classes 16-15
container 16-15
desktop page 16-17
entry types 16-16
topic 16-15
Java properties, CRL 14-1
Java Server 13-2
JDK 11-2
JRE 10-1, 11-2
JSP 13-2
JVM 10-3
using your own installation 11-2

K

keys, shortcut 1-11, 1-12, 3-3
date&time 1-11, 3-10
keyword icon 1-10, 3-2, 4-8
keywords 1-12
attaching to entry 4-7
automatically linked 4-7
category type
configuring for menu/submenu 16-8
configuring for topic 16-12
class attribute for entry type 16-4
configure as auto insert 16-4
configure as nonremovable 16-4
configure as removable 16-4
configuring 16-4
default 4-7
defining in config file 11-6
definition of 4-7
editing list (for an entry) 4-8
entry data type, configuring for toolbutton 16-14
nonremovable 4-7
overall type, configuring 16-3
page type, configuring 16-7
removable 4-7
type definitions
cateogry type 16-4
entry type 16-4
overall type 16-4
page type 16-4
Keywords filter for Inquiries 8-5

L

launching CRL 2-1, 11-4
from Linux 2-1
from Windows 2-1
in AFS 2-2
location of database 14-5
Log In/Log Out 2-5
Entry Signers config 2-3
PIN and password 2-5
username and password 2-3
Log In/Log Out button 1-3
logbook entry
add/change/replace text 4-4
adding entries from one thread to another 7-9
adding entries to a thread 7-5
allow/prevent anonymous 14-2
allow/prevent filtering 14-2
annotating 8-13
attaching keywords 4-8
global 4-9
automating 6-1
autoscheduled 6-1
scheduling parameters 6-2
terminate job 6-3
category 1-7
changing text font 4-6
checkpoint 14-2
configure max image size 14-2
configuring web access 13-1
control key shortcuts 1-12
create pointers in SQL database 14-5
creating 3-2
Ctrl-Alt-D for date&time 1-11, 3-10
cut/copy/paste text 4-5
data types 1-10, 3-3
date&time entry key 1-11, 3-10
default settings for header display 14-3
deleting (unarchived) 4-6
deleting checkpoint entry 4-7
deleting entries from thread 7-9
double-click shortcut 1-11
double-click to create 3-3
entries left unarchived 5-3
features 1-11
form archive features 5-1
format description 1-10
function key shortcuts 1-12
header description 1-10
keyword config (entry type) 16-4
keywords attached to 4-7
keywords automatically displayed 14-3
mailing entries in a thread 7-8
printing 4-11
printing entries in a thread 7-8
PS/PDF file entry, configuration 11-15
recover checkpointed entry 4-10
refresh to see annotations 8-14, 8-15
reminder to archive 14-2
reminder to archive or delete 1-12

removing from container 5-3
removing from view (archived) 5-3
removing keywords 4-8
global 4-9
scheduled (see autoscheduled)
searching for
in CRL 8-1
on the web 8-9
select/deselect in container 4-3
selecting entries 4-3
selecting text within 4-4
send to email recipient 4-11
setting browse directory for input files 14-3
setting default font size 14-1
setting directory for entry storage 14-3
signing 2-3
summary of features 1-11
threading entries 7-1
at archive time 5-1
topic 1-7
view/sort entries in a thread 7-7
viewing archived entries 8-1
logbook entry database
creating an inquiry 8-2
executing an inquiry 8-7
restoring an inquiry 8-8
saving an inquiry 8-8
searching via Inquiries 8-1
searching via LogEntry Explorer 8-1, 8-8
LogBook Entry toolbar 1-10
LogBook_admin directory 14-1
LogBookConfig.xml file 11-6
LogbookConfigParms.properties file See properties file
LogEntry Explorer 8-1
description 8-8
logging in 2-2, 2-4
Entry Signer config 2-3
login id 2-3
PIN config 2-6
logging out
Entry Signers config 2-5
PIN config 2-7
login methods 2-2
configuring 14-2
login timeout prompt 14-2
logout prompting 14-2
lpr 11-17

M

mail list directory, configuring 14-4
mail list files for emailing forms
contents and format 11-10, 11-11
mailing entries
from user interface 4-11
mailing form entry to mail list 5-2
from user interface 4-11
mailing forms
return address 14-4
menu
configuration 11-7, 16-8
define 11-7
description 1-6
menu items
configuring lowest level (topic) 16-9
configuring varying levels 16-8
message, attaching to entries 3-11
message, global 1-4
methods for login 2-2
mouse rollover on toolbar, config 16-13
MySQL database 10-2

N

nagging 5-3
See reminder to archive
set repeat interval 14-2
set start interval 14-2
nonremovable keyword 4-7
notational conventions 1-3

O

objects usable in forms 17-4
Operator field of entry
entry signer list 5-1
Entry Signers config 2-3
PIN config 2-5
operators (for XML elements) 15-3
Operators filter for Inquiries 8-4
OS command entry
adding logbook entry 3-10
entering user-specified command 3-10
entry type class 16-16
specifying preset command 16-14

P

page
configuring (data-input) 16-7
configuring (non-data-input) 16-6
defining a new page 11-7
definition 1-5
entry input 1-5
no entry input 1-5
page tab 1-5
defining 11-7, 16-5
select 3-1
parameters for CRL, setting 14-1
password
administrative 12-1
changing (user) 1-3, 2-7
for login 2-2
length 2-3, 2-7
user 2-3
paste/cut/copy 4-5
paste-from-clipboard menu option 4-5
PDF files
adding 3-8
as entry, configuration 11-15
set temporary directory in properties file 14-4
set viewer in properties file 14-5
viewing 3-9

PIN 2-2, 14-2
length 2-5
Operator field of entry 2-5
using at archive time 2-5
using at login 2-5
plain text entries
creating 3-4
definition 3-4
Plog (See process logger)
PostScript files
adding 3-8
as entry, configuration 11-15
set temporary directory in properties file 14-4
viewer, set in properties file 14-5
viewing 3-9
Preferences button 1-4
preferences for entry header lines 1-4
preparation for use of CRL 11-1
AFS 10-9
Print All Entries in this Topic menu option 8-18, 8-19
print queues, configuring 11-17
Print Selected Entries menu option 8-18
printing a report 8-16
printing entries 4-11
process logger 1-1, 9-1
definition 9-1
entry content 9-1
entry message elements 9-2
input format 9-2
programmer instructions 9-1
return messages 9-3
sample program 9-3
starting the daemon 11-17
properties file
configuration parameters 14-1
description 14-1
edit before initial launch 11-4
guidance on editing 11-1
required edits 11-1
set database information 11-1
PS files See PostScript files
PS/PDF file view configuration 11-15
PS/PDF files
set temporary directory in properties file 14-4
pull-down menu
description 1-6

R

Redo menu option 4-4, 4-6
refresh browser window to see annotations 8-15
Refresh button in archived entry 8-13
refresh entry to see annotations 8-14
reloadable form fields 11-9, 11-12
reminders to archive 1-12, 5-3
removable keyword 4-7
Remove All Entries in this Topic menu option 4-6, 5-3, 8-18
Remove Selected Entries menu option 4-6, 5-3, 8-18
removing CRL application 10-11
repeat block in form entry 11-10
report
creating 8-16
description 8-16
export binary files 8-18
printing 8-16, 8-18
Report Category menu 8-17
report container 8-1, 8-16
Add Selected Entries to a Thread menu option 8-19
class 16-16
definition 1-9
Delete Selected Entries from a Thread menu option 8-19
Entries menu 8-16
inserting archived entries into 8-17
menu options 8-16
Print All Entries in this Topic menu option 8-18, 8-19
Print Selected Entries menu option 8-18
Remove All Entries in this Topic menu option 8-18
Remove Selected Entries menu option 8-18
Report Category menu 8-17
Send ALL Entries To Mail Recipient option 8-18
Send Selected Entries To Mail Recipient menu option 8-18
report entry
removing from container 8-18
selecting 8-18
report page
desktop page class 16-17
requirements
CRL installation and running 10-2
CRL Web Access 10-2
Restore Inquiry button 8-8
root element type (XML) 16-3
ROOT plots
entry 3-8

S

Save Inquiry button 8-8
scheduled container (see autoscheduled container)
scheduled logbook entry (see autoscheduled logbook entry)
Scheduler button 1-4
scheduling parameters (see autoscheduled logbook entry)
script for invoking CRL 11-3
modify 11-4
script, executing in form 17-8
search archived entries 1-12
Inquiries 8-1
LogEntry Explorer 8-1
Search button 1-3, 8-1
Select ALL Entries menu option 4-4
select multiple entries 4-3
select single entry 4-3
Select-all menu option 4-4
selecting entries 4-3, 8-18
Send ALL Entries To Mail Recipient menu option 8-18
Send Selected Entries To Mail Recipient menu option 8-18
server for email, configuring 14-4
setting container size 3-3
shortcut
create entry 1-11
date&time 1-11, 3-10
double-click to create entry 3-3
signing logbook entries 2-3
SQL database information 14-1
starting process logger daemon 11-17

status bar on container 1-7
stmp mail host, configuring 14-4
submenu configuration 11-7, 16-8
syntax for XML elements 16-2
system requirements 10-1

T

table in form entry 11-9, 11-11
scrollable fields after archive 3-6, 8-7, 8-8, 8-17
tar file of CRL, installing 10-9
TCP ports for plog 11-17
temporary directory for PS/PDF files 14-4
text entries 3-4
creating 3-4
entry type class 16-16
thread 1-12
adding entries from input container 7-6
adding entries from thread container 7-6
adding entries to (general) 7-5
adding entries to thread at archive time 5-1
adding entry from another thread 7-9
creating (general) 7-3
creating from input container 7-4
creating from thread container 7-5
creating from Thread Explorer 7-4
creating new at archive time 5-1
deleting entries from 7-9
deleting thread 7-10
description 7-1
file name 7-1, 7-3
listing current threads 7-1
mailing entries 7-8
printing entries 7-8
sorting entries 7-7
Thread Explorer 7-1
viewing entries 7-7
thread container 1-9
adding entries to thread 7-6
adding entry to different thread 7-2
class 16-16
creating a thread 7-5
deleting entry from thread 7-2
description 7-2
mailing entries 7-2
printing thread 7-2
selecting entries 7-2
sort entries 7-2
viewing a thread 7-2
Thread Explorer
creating a thread 7-1, 7-4
deleting a thread 7-1
description 7-1
dropping thread into container 7-1
selecting thread to view 7-7
Thread Explorer button 1-4
thumbnail for PS/PDF
configuring 11-15
thumbnail images
entering gif or jpg files 3-6
for binary file 3-8
PS/PDF file 3-9
time zone, define 14-2
time/date shortcut key 3-10
timeout prompt for login 14-2
Tomcat 10-2, 13-2
installation 13-1
toolbar button (CRL)
About LogBook 1-4
Administrator 1-3
Change Password 1-3
Checkpoint Recovery 1-4
Current Situation 1-4
Entry Signers 1-3
Exit LogBook 1-5
Log In/Log Out 1-3
Preferences 1-4
Scheduler 1-4
Search 1-3
Thread Explorer 1-4
toolbar button (LogBook entry)
configuring 16-12
text vs image 16-13
toolbar, CRL 1-2
description 1-3
toolbar, LogBook entry 1-10
configure entry types 16-12
configuring 16-12
data input page configuration 16-7
defining 11-8
using 3-2
toolbutton configuration 11-8
image for button 11-8
topic
(of logbook entry) definition 1-7
as configured in menus and submenus 16-8
attribute list 16-9
configuring 16-9
define 11-7
selecting from pull-down menu 3-1

U

unarchived entries
reminder to archive 5-3
Undo menu option 4-4, 4-6
uninstalling CRL 10-11
Un-select ALL Entries menu option 4-4
UPS/UPD 10-9
launch CRL 2-1
UPD kit for CRL 10-1
user categories
administrator 12-2, 12-4
operator 12-2, 12-4
remote user 12-2, 12-4
user interface description 1-2
username 2-3

V

verify container category 3-2
view annotations on entry 8-14
view annotations via web 8-15
view PostScript/PDF file 3-9
viewer for PDF files, configuring 14-5

viewer for PS files, configuring 14-5
VT100 window, copy text 4-5

W

web access 1-1
    additional software installation 13-1
    canned inquiries 8-11
    canned inquiry, configure 13-3
    configuring 13-1
    define search criteria 8-10
    inquiry page features 8-10
    javaserver page, configure 13-2
    JSP info sources 13-3
    keywords table, configure 13-2
    NOT_<keyword>, configure 13-2
    one-click searches 8-11
    one-click searches, configure 13-3
    requirements for 10-2
    to logbook entry database 8-9
Word Search filter for Inquiries 8-6
workspace description 1-5

X

X application 4-5
Xerces 10-3
XML 1-1
    attributes in DTD file 15-4
    comparison to HTML 15-1
    declaration in file 16-2
    element content 15-3
    element operators 15-3
    element syntax 16-2
    elements in DTD file 15-3
    entry storage 1-1
    introduction 15-1
    reference for CRL 15-1
    sample element with attributes 15-2
    valid vs. well-formed 15-1
XML file for end of shift form 17-3
XML file for form showing embedded forms 17-8
XML file for form showing object use 17-4
XML file for form showing table element 17-7
XML file for inquiry feature 17-14
XML file for start of shift form 17-3