



Chicago Interface Group, Inc.

---

*FastLIST*

Batch User Guide

Chicago Interface Group, Inc.  
858 West Armitage Avenue #286  
Chicago, IL 60614 USA

Phone: (773) 524-0998  
Fax: (815) 550-6088  
Internet: [www.cigi.net](http://www.cigi.net)  
Email: [support@cigi.net](mailto:support@cigi.net)

*FastLIST release 7.0*

*Copyright © 2007 Chicago Interface Group, Inc. All Rights Reserved  
Documentation version April 3, 2006*

*FastLIST is a registered trademark of Chicago Interface Group, Inc.  
CA-ENDEVOR is a registered trademark of Computer Associates International, Inc.*

Table of Contents

<b>Chapter 1: The FLIST Utility .....</b>	<b>1-1</b>
.....	1-2
Introduction to FastLIST Utility.....	1-3
Overview .....	1-3
Querying Against The Database Step-by-Step.....	1-3
JCL to Execute FLIST.....	1-4
JCL for FLIST .....	1-4
FastLIST DDNames .....	1-5
Associated ddnames, Descriptions and Attributes .....	1-5
FastLIST Utility in a Processor .....	1-6
FLIST Return Codes.....	1-7
FastLIST Execution Log .....	1-8
FastLIST Syntax .....	1-9
FastLIST Syntax Summary .....	1-9
FastLIST Statement Rules .....	1-10
Element Clauses .....	1-10
Where CCID Statements .....	1-11
Where Component Statements.....	1-11
Where Object Statement.....	1-12
Where Processor Group Statement .....	1-12
Where Generate Failed Statement.....	1-13
Where Date/Time Statement .....	1-13
Build Statements .....	1-13
Option Statements .....	1-14
Aliases .....	1-15
Examples .....	1-16
CCID Examples.....	1-16
Component Examples .....	1-17
Mixed Search Examples.....	1-18
Output Examples .....	1-20
Output from Default NO BUILD Statement.....	1-20
Output from BUILD ACTION MOVE Statement.....	1-21
User Action Support .....	1-22
CIGUSER0.....	1-22
Build Report Option .....	1-25
Output from BUILD REPORT Statement .....	1-25
Build Data Option.....	1-26
Output From Build DATA Statement .....	1-26
<b>Chapter 2: Configuration XREF Reports.....</b>	<b>2-1</b>
.....	2-2

Message Log.....	2-3
Introduction to FastLIST XREF Reports .....	2-4
Overview .....	2-4
Submitting Reports.....	2-5
Using ISPF Front-end .....	2-5
Reports Syntax.....	2-6
Report Syntax Summary .....	2-6
Statement Rules .....	2-7
Report Selection Clause .....	2-7
Element Filter Clause .....	2-8
Where CCID Statements .....	2-8
Where Component Statements .....	2-8
Where Object Statement.....	2-9
Where Processor Group .....	2-9
Aliases .....	2-10
Sample Reports.....	2-11
Report Elements By CCID.....	2-11
Report CCIDs By Element.....	2-13
Report Elements By Component.....	2-14
Report Components By Element.....	2-15
Report Elements By Dataset .....	2-16
Report Datasets By Element .....	2-17
Elements By Processor By Processor Group .....	2-18
Elements By Processor Group By Processor .....	2-19
<b>Chapter 3: Query Tips &amp; Techniques .....</b>	<b>3-1</b>
.....	3-2
Query Tips and Techniques.....	3-3
FastLIST Database Concepts .....	3-3
The Records.....	3-3
Building the Optimum Search.....	3-4
Layering Syntax .....	3-5
Stacking Syntax.....	3-5
DELTA DATE and DELTA CCIDs .....	3-5
DELTA Versus ANY .....	3-7
Using the DATE and TIME Ranges .....	3-8
<b>Chapter 4: Record Layouts .....</b>	<b>4-1</b>
.....	4-2
File Layouts .....	4-3
FST@EFP: The Mini-element Record.....	4-4
FST@ICMP: The Input Component Record.....	4-5
FST@OBJT: The Object Record .....	4-6
FST@OCMP: The Output Component.....	4-7
FST@PCMP: The Processor Component Record .....	4-8

Table of Figures.....	4-9
Index.....	4-10



# ***FastLIST***

## Chapter 1:           The FLIST Utility

This chapter contains:

- A step-by-step explanation of how to use the FLIST utility.
- FLIST Syntax.
- Syntax Examples.
- JCL for FLIST.



# Introduction to FastLIST Utility

---

## Overview

The FLIST utility extracts data stored in the FastLIST database, and produces:

- Output in the form of SCL.
- A simple report.
- Formatted data for export to other user applications.

You can run FLIST outside of Endeavor by using JCL or a CLIST, or FLIST can be run as a program inside of an Endeavor processor.

The first part of this chapter explains the mechanics of running the FLIST utility. JCL and syntax rules are presented in detail. The latter part of this chapter provides usage examples, and illustrates the various FLIST outputs.

## Querying Against the Database Step-by-Step

Use the following five steps as a guideline while learning how to use the FLIST utility. Each of the steps is discussed in this chapter.

❶	Review dataset allocations and dataset attributes.
❷	Modify JCL.
❸	Build FLIST syntax.
❹	Submit JCL, CLIST or run processor.
❺	Review output messages.

*Figure 1.1  
FLIST Step-By-Step*

## JCL to Execute FLIST

---

### JCL for FLIST

Tailor and submit the following JCL to invoke the FLIST utility. This JCL is in the JCLLIB Dataset offloaded from the CIG product installation tape under member name CIGJCL05.

```
/** (JOB CARD)
/**
/** *****
/**
/** THIS IS THE JCL TO RUN THE TEST FLIST AND REPORT UTILITIES.
/**
/** ALSO, INSERT A VALID JOB CARD AND MODIFY FLHQ1 AND FLHQ2 AS PER *
/** YOUR WORKSHEET.
/** *****
//CIGLIST EXEC PGM=CIGFLIST
//STEPLIB DD DSN=FLHQ1.FLHQ2.STEPLIB,DISP=SHR
//CIGLOG DD SYSOUT=*
//CIGIN DD DSN=FLHQ1.FLHQ2.SAMPLIB(FLISTIVP),DISP=SHR
//CIGRPT DD SYSOUT=*
```

*Figure 1.2  
CIGJCL03 FLIST Utility*

**You must execute the program CIGFLIST if running the FLIST utility in an Endeavor processor. Do not execute the program CIGFEEXEC.**

## FastLIST DDNames

---

### Associated ddnames, Descriptions and Attributes

The following are ddnames used by FLIST. Ensure that all required datasets are allocated with the proper attributes.

ddname	Description	Attributes
CIGIN	<b>*REQUIRED*</b> The input dataset. If the dataset is not found or cannot be opened, FLIST will fail in initialization.	LRECL=80 RECFM=FB DSORG= PS or PO
CIGLOG	<b>*REQUIRED*</b> FLIST messages are written to this dataset. If the dataset is not found or cannot be opened, FLIST will fail in initialization. This dd is also the default target of the FLIST action for ACTION building.	DSORG=PS LRECL=132 BLOCKSIZE=13200 RECFM=FBA Or SYSOUT=*
CIGRPT	<b>*OPTIONAL*</b> Required only if using BUILD REPORT syntax. FLIST writes to this dataset if the syntax BUILD REPORT is specified.	LRECL=132 BLOCKSIZE=13200 DSORG=PS RECFM=FBA Or SYSOUT=*
CIGDATA	<b>*OPTIONAL*</b> Required only if using BUILD DATA syntax. FLIST writes to this dataset if the syntax BUILD DATA is specified.	LRECL=250 BLOCKSIZE=25000 DSORG=PS RECFM=FB
CIGINI	<b>*OPTIONAL*</b> Required only if initializing from a CIGINI01 module. Must contain a CIGINI01 module.	DSORG=PO LRECL=0 RECFM=U BLOCKSIZE=6233
CIGTRACE	<b>*OPTIONAL*</b> Allocating this dataset activates the CIGTRACE facility. CIGTRACE writes detailed informational and debugging messages to the CIGLOG DD.	DUMMY

*Figure 1.3  
Associated ddnames, Descriptions, and Attributes*

## FastLIST Utility in a Processor

---

The following is an example of how to use FastLIST in a processor. This sample can be found in the SAMPLIB dataset offloaded from the CIG product installation tape under member name FLISTPGA.

```
//FLISTPGA PROC SYSOUT=H
//* -----
//* EXAMPLE OF FLIST IN A PROCESSOR
//* -----
//* -----
//* TO BE USED WITH MACRO TYPE ELEMENTS INSTEAD OF *NOPROC*
//* THIS IS A SAMPLE, SIMPLE PROCESSOR FOR INVOKING FLIST IN
//* A PROCESSOR. OUTPUT IS IN THE FORMAT OF THE DEFAULT &&ACTION
//* STATEMENTS. USER MAY WANT TO FURTHER TAILOR THIS PROCESSOR
//* THROUGH THE USE OF SYMBOLICS.
//* -----
//* -----
//* THE FLIST ELEMENT OUTPUT IS DIRECTED TO THE DEFAULT *
//* DESTINATION OF CIGLOG. USERS WILL NEED TO MODIFY THIS
//* PROCESSOR IF THEY WANT TO SEND OUTPUT TO A DIFFERENT DDNAME
//* OR SPECIFIC DSNAME. PLEASE REVIEW CHAPTER 6, FLIST SYNTAX FOR
//* BUILD SCL OUTPUT OPTIONS.
//* -----
//CIGLIST EXEC PGM=CIGFLIST
//CIGLOG DD SYSOUT=&SYSOUT
//CIGIN DD *
FLIST ELEMENTS *
    FROM ENV '&&C1ENVMT'
           SYS '&&C1SYSTEM'
           SUB *
           TYPE ASM
           STAGE NUMBER *
           WHERE INPUT COMPONENT EQUAL '&&C1ELEMENT'
```

*Figure 1.4*  
*Sample FLIST in a Processor*

**You must execute the program CIGLIST if running the FLIST utility in an Endeavor processor. Do not execute the program CIGFEXEC.**

## FLIST Return Codes

---

Return Code	Meaning
00	All internal steps completed without errors.
04	Warning messages were issued. See accompanying messages for more details.
12	A severe error has occurred with load process or with MVS access services. See accompanying messages for more details.

All messages sent from the utility and/or any of the database pieces will begin with the prefix FST and will be written out to CIGLOG DD.

# FastLIST Execution Log

---

The following is a standard FastLIST execution summary report, found in CIGLOG after every FastLIST execution. Note the first messages printed are the initialization messages.

```
17:22:24 FST0245I PRIMARY VSAM DATASET NAME 'CIGT.TESTDB'.
17:22:24 FST0246I ALTERNATE INDEX VSAM DATASET NAME 'CIGT.TESTDB'.
17:22:24 FST0244I OPTIONS IN EFFECT. COMPS='Y' AND CCIDS='Y'.
17:22:24 FST0254I      FOREGROUND EXECUTION=Y      ALTERNATE CIGINI ALLOWED=Y
17:22:24 FST0251I FASTLIST APPLICATION MODULES BEING LOADED FROM
DATE 97/06/24 TIME 17:22:26      F A S T L I S T U T I L I T Y, RELEASE 7.0
      E X E C U T I O N R E P O R T

17:22:26 FST1102I  PARSER BEGINS
17:22:26 FST0020I  FLIST ELEMENTS ZTEST*
17:22:26 FST0020I      FROM STAGE NUM 1
17:22:26 FST0020I      WHERE INPUT COMPONENT EQUAL FLWORK
17:22:26 FST0020I  .
17:22:26 FST0020I  FLIST ELEMENTS ZTEST*
17:22:26 FST0020I      FROM STAGE NUM 1
17:22:26 FST0020I      WHERE INPUT COMPONENT EQUAL FLWORK
17:22:26 FST0020I  BUILD ACTION MOVE
17:22:26 FST0020I  .
17:22:26 FST0020I  FLIST ELEMENTS ZTEST*
17:22:26 FST0020I      FROM STAGE NUM 1
17:22:26 FST0020I      WHERE INPUT COMPONENT EQUAL FLWORK
17:22:26 FST0020I  BUILD REPORT
17:22:26 FST0020I  .
17:22:26 FST0020I  FLIST ELEMENTS ZTEST*
17:22:26 FST0020I      FROM STAGE NUM 1
17:22:26 FST0020I      WHERE INPUT COMPONENT EQUAL FLWORK
17:22:26 FST0020I  BUILD DATA
17:22:26 FST0020I  .
17:22:27 FST1103I  PARSER ENDS, RC=0000
17:22:27 FST1111I  STARTING DATABASE QUERY.
17:22:27 FST0500I  DATABASE QUERY RESULTED IN 0007 INPUT RECORDS SENT
      SET FROM ENV 'TEST' SYS 'SYSA' SUBSYS 'SUBA' TYPE 'ASM' STAGE NUM 1.
      &&ACTION ELEMENT ZTEST1      VERSION 01 LEVEL 02 .
```

*Figure 1.5  
Execution Log Example*

# FastLIST Syntax

---

## FastLIST Syntax Summary

The following table shows all FLIST syntax.

FLIST	ELEMENTS	<i>element-name</i>	
	THROUGH	<i>element-name</i>	
	FROM	ENVIRONMENT	<i>env-name</i>
		SYSTEM	<i>system-name</i>
		SUBSYSTEM	<i>subsystem-name</i>
		TYPE	<i>type-name</i>
		STAGE NUMBER	<i>stage-number</i>
	TO	<u>CIGLOG</u>	
		FILE	<i>ddname</i>
		DSNAME	' <i>dataset-name</i> '
		[MEMBER]	<i>member-name</i>
	WHERE	CCID	OF {CURRENT LAST RET GEN DELTA ANY}
		EQUAL	<i>CCID</i>
			{ [RELATED] <u>INPUT</u>   [RELATED] OUTPUT   PROCESSOR   ANY }
		EQUAL	COMPONENT
			<i>component-name</i>
		INDIRECT	
		THROUGH	<i>component-name</i>
		ENVIRONMENT	<i>env-name</i>
		SYSTEM	<i>system-name</i>
		SUBSYSTEM	<i>subsystem-name</i>
		TYPE	<i>type-name</i>
		STAGENUM	<i>stage-number</i>
		VERSION	<i>version</i>
		LEVEL	<i>level</i>
		FILE	<i>ddname</i>
		DSNAME	' <i>dataset-name</i> '
		INCLUDE RELATED	
		RELATED ONLY	
		OBJECT EQUAL	' <i>object-name</i> '
		COMMENT EQUAL	' <i>comment</i> '
		PROCESSOR GROUP EQUAL	<i>processor-group</i>
		GENERATE FAILED	
			{CURRENT GENERATE LASTMOD DELTA}
			[FROM]
			DATE EQUAL YY/MM/DD [TIME EQUAL HH:MM]
			[THROUGH DATE EQUAL YY/MM/DD [TIME EQUAL HH:MM]]
	BUILD	{ACTION {&& <u>ACTION</u>   <i>action-name</i> [WITH COMPONENTS]}	
		REPORT DATA}	
	OPTIONS	MATCH EQUAL { <u>ALL</u>  ANY}	
		NOMATCH EQUAL { <u>0</u>  4 12}	
		SHOW LEVELS	
		REPLACE MEMBER	

Figure 1.6  
FastLIST Syntax Summary

# FastLIST Statement Rules

---

## Element Clauses

### **FLIST ELEMENTS** *element-name*

Indicates the 1-10 character element name to be loaded, and may be wild carded. FLIST ELEMENTS is the only clause required to run the FLIST utility. All of the following clauses are optional. Element clauses omitted are treated as wild cards. Default values for all other options are indicated below.

<b>FROM</b>	<b>ENVIRONMENT</b>	<i>env-name</i>
	<b>SYSTEM</b>	<i>system-name</i>
	<b>SUBSYSTEM</b>	<i>subsystem-name</i>
	<b>ELEMENT</b>	<i>element-name</i>
	<b>TYPE</b>	<i>type-name</i>
	<b>STAGE NUMBER</b>	<i>stage-number</i>
	<b>VERSION</b>	<i>version</i>
	<b>LEVEL</b>	<i>level</i>

The FROM parameters indicate the Endeavor inventory location queried in the FastLIST database. The ENVIRONMENT, SYSTEM, SUBSYSTEM, ELEMENT, and TYPE are 1-8 characters. The STAGE NUMBER is one character limited to 1, 2 or \*. Version and level accept one- or two-character numeric values. Version accepts values from 1-99, and level accepts values from 0-99. All of these values may be wild carded.

### **THROUGH** *element-name*

Use of this option limits data extraction to a range of elements from first element name up to and including the through element value. This field accepts wild carding.

### **TO CIGLOG**

This control statement indicates the target destination of any SCL produced. If no TO statement is coded, then the default is CIGLOG. Other options include:

<b>FILE</b>	<i>ddname</i>
<b>DSNAME</b>	<i>dataset-name</i>
<b>[MEMBER]</b>	<i>member-name</i>

FILE is a ddname 1-8 characters, DSNAME is a 1-44 character dataset name, and MEMBER is an optional 1-8 character member name for partitioned datasets.

## Where CCID Statements

**WHERE CCID OF  
{CURRENT|LAST|RETRIEVE|GENERATE|DELTA|ANY} EQUALS  
CCID**

Indicates that only records with a valid CCID from the specified CCID type be listed. If no CCID type is indicated, then CURRENT is the default.

## Where Component Statements

**WHERE {[RELATED] INPUT | [RELATED] OUTPUT | PROCESSOR |  
ANY} COMPONENT EQUALS *component-name***

Indicates that only records that meet the specified component criteria be returned. For users of Endeavor 3.7, you may use the optional RELATED INPUT and OUTPUT syntax.

### INDIRECT

The INDIRECT keyword allows indirectly impacted elements to be found and included in the search.

To further tailor the WHERE COMPONENT criteria, specify more detailed information about the component's Endeavor or dataset location using the following optional component key words:

<b>ENVIRONMENT</b>	<i>env-name</i>
<b>SYSTEM</b>	<i>system-name</i>
<b>SUBSYSTEM</b>	<i>subsystem-name</i>
<b>ELEMENT</b>	<i>element-name</i>
<b>TYPE</b>	<i>type-name</i>
<b>STAGE NUMBER</b>	<i>stage-number</i>
<b>VERSION</b>	<i>version</i>
<b>LEVEL</b>	<i>level</i>

The ENVIRONMENT, SYSTEM, SUBSYSTEM, ELEMENT, and TYPE are 1-8 characters. The STAGE NUMBER is one character limited to 1, 2 or \*. Version and level accept one or two character numeric values. Version accepts values from 1-99, and level accepts values from 0-99. All of these values may be wild carded.

**FILE *ddname***

The use of the FILE keyword indicates that only components that use the 1-8 character *ddname* provided will meet search criteria.

**DSNAME '*dataset-name*'**

The use of the DSNAME keyword limits the database search to only the input components that come from or the output components that go to the specified *dataset-name*. The *dataset-name* must consist of 1-44 characters, and it must be enclosed in single quotation marks if it contains an embedded period. No validation is done on this field as it is considered to be a data field, not an active dataset.

**INCLUDE RELATED**

The use of this keyword will include related inputs and/or outputs in the search criteria. Note this keyword is only valid for release 3.7 and above.

**RELATED ONLY**

The use of this keyword will limit the search criteria to related inputs and/or outputs. Note this keyword is only valid for release 3.7 and above.

Where Object Statement

The use of the following WHERE OBJECT clause will create a list of elements based on the correlation of an object to an Endeavor element.

**WHERE            OBJECT EQUALS            '*object name*'**  
**COMMENT EQUALS        '*comment*'**

*Object name* and *comment* are 1-70 character values.

Where Processor Group Statement

**WHERE            PROCESSOR GROUP EQUALS *processor-group***

Use of this parameter indicates that the returned element list should be further tailored by processor group name. The group name is a 1-8 character value. If this statement is not coded, processor groups will be ignored in the search criteria.

## Where Generate Failed Statement

### **WHERE      Generate Failed**

Use of this parameter will search for elements having a return code of 12 or higher from the generate processor.

## Where Date/Time Statement

### **WHERE {CURRENT|GENERATE|LASTMOD|DELTA} [FROM]DATE EQUAL YY/MM/DD [TIME EQUAL HH:MM] [THROUGH DATE EQUAL YY/MM/DD [TIME EQUAL HH:MM]]**

Use of this parameter will search for elements that meet the date, time, and type criteria provided.

## Build Statements

The FLIST utility will create output in the form of SCL, a simple report, or formatted data for export to other applications. To direct the FLIST utility to create one of those outputs, you must code one of the following three BUILD statements. If no BUILD statement is coded, then the output will default to BUILD ACTION &&ACTION format of Endeavor. Note BUILD ACTION, BUILD REPORT, and BUILD DATA are mutually exclusive.

### **BUILD      ACTION      *action-name***

#### **[WITH COMPONENTS]**

Use of this parameter tells FastLIST to build ACTION SCL for the action provided. Coding BUILD ACTION without including an ACTION name results in FastLIST output in the format of partially built SCL, with an action of &&ACTION. To create SCL for another action, code that action name after the ACTION keyword. The actions supported by FastLIST are &&ACTION, MOVE, GENERATE, RETRIEVE, DELETE and TRANSFER. The WITH COMPONENTS control statement directs FastLIST to write out the input components of the parent element along with the element.

### **BUILD      REPORT**

This parameter directs FLIST to build a formatted list of elements that meet the search criteria.

## **BUILD DATA**

Build DATA formats records for export to other user applications. Assembler DSECTs for this data can be found in the SAMPLIB dataset offloaded from the CIG product installation tape in members FST@EFP through FSTPCMP.

### Option Statements

#### **OPTIONS MATCH EQUALS {ALL|ANY}**

Coding this option will affect the search based on all of the "WHERE" input clauses. Coding the ALL option requires records to contain all WHERE criteria provided. Coding the ANY option means that records with any of the given WHERE criteria will be included in the element list. The default is ALL.

#### **NOMATCH EQUALS {0|4|12}**

Coding this option affects how return codes are set when a no match condition has been found. FLIST will continue execution of following FLIST statements with a return code of 0 or 4. FastLIST will terminate processing immediately if it receives a return code of 12. The default is 0.

#### **SHOW LEVELS**

This option pertains only to the situation where the FastLIST syntax includes either DELTA CCIDS or DELTA DATE/TIME requests. If you code SHOW LEVELS, then the output list will contain both the current level and the qualifying delta levels.

#### **REPLACE MEMBER**

This option applies only when you are writing out ACTION SCL using the BUILD ACTION statement. If you are writing this ACTION SCL out to a partitioned dataset member, and this member already exists, FastLIST will not overwrite the file and will fail with a return code of 12. If you code the REPLACE MEMBER option, FastLIST will completely replace the member.

# Aliases

---

FastLIST will accept all of the syntax listed above, as well as an abbreviation consisting of the first three letters only of any keyword. All other valid aliases are listed in table 1.7.

Keyword	Valid Aliases
COMPONENT	COMP, COMPONENTS
ELEMENTS	ELEMENT
EQUAL	EQUALS, EQ, =
FLIST	LIST
FILE	DDNAME
THROUGH	THRU

*Figure 1.7  
FastLIST Aliases*

# Examples

---

## CCID Examples

These examples show various syntax examples for element searches based solely on the WHERE CCID clause. The purpose of these examples is to depict some of the different ways in which keyword searches can be built. FastLIST also supports more complex combinations of WHERE statements.

Example 1 shows a search that will result in a list of all elements that start with ZTEST\*, reside in the inventory location stated and have a Current CCID beginning with QA\*. The output format is the default &&ACTION.

### EXAMPLE 1

```
FLIST ELEMENTS ZTEST*
FROM ENV TEST
    SYS SYSA
    SUB SUBA
    TYPE ASM
    STAGE NUM 1
WHERE CCID OF CURRENT EQUAL 'QA'
```

*Figure 1.8  
Example 1*

Example 2 requests a list of all elements that start with ZTEST\* in any inventory location and that have a Current CCID that starts with QA\* or ADD\*. The output is formatted as a line report.

### EXAMPLE 2

```
FLIST ELEMENTS ZTEST*
    WHERE CCID OF CURRENT EQUAL (QA*,ADD*)
BUILD REPORT
.
```

*Figure 1.9  
Example 2*

In Example 3, the request is the same as in Example 2, except the search criteria is set for an ALL condition, by default (see above, MATCH EQUALS {ANY/ALL}). The resultant list will contain only elements that have both a Current CCID of QA\* **and** a Generate CCID that starts with GEN\*. The output format will be MOVE ACTION SCL.

### EXAMPLE 3

```
FLIST ELEMENTS ZTEST*
  WHERE CCID OF CURRENT EQUAL (QA*)
  WHERE CCID OF GENERATE EQUAL (GEN*)
BUILD ACTION MOVE
.
```

*Figure 1.10  
Example 3*

## Component Examples

The following example shows syntax for searching for elements based solely on the WHERE COMPONENT clause.

Example 4 requests a list of all elements in the FastLIST database that have an Input Component of FLWORK. This is a common search that is the equivalent of asking "show me everything that uses macro FLWORK as an input."

### EXAMPLE 4

```
FLIST ELEMENTS *
  WHERE INPUT COMPONENT EQUAL FLWORK.
```

*Figure 1.11  
Example 4*

Example 5 requests a list of all elements in any inventory location that use SYS1.MACLIB for inputs or CIGT.OBJLIB1 for outputs. Note the MATCH EQUAL ANY clause. This tells FastLIST to include elements which satisfy any one of the WHERE clauses. This type of syntax can be used to research which elements use a particular dataset if, for example, you plan to rename all of the system datasets and want to see which production elements will be affected.

### EXAMPLE 5

```
FLIST ELEMENTS *
  WHERE INPUT COMPONENT EQUAL (*)
        DSNAME 'SYS1.MACLIB'
  WHERE OUTPUT COMPONENT EQUAL (*)
        DSNAME 'CIGT.OBJLIB1'
OPTIONS MATCH EQUAL ANY
.
```

Figure 1.12  
Example 5

Example 6 shows the prototypical use of the FLIST utility. You want to change a processor, perhaps during a major conversion. The following syntax will list all PROD elements generated using the PGAGEN1 processor. As a complement to this information, you may want to run a REPORT listing all elements per processor and, alternatively, all processors per element. See Chapter 2 *Configuration XREF Reports*, for information on how to run these reports.

### EXAMPLE 6

```
FLIST ELEMENTS *
  ENV PROD
  WHERE PROCESSOR COMPONENT EQUAL 'PGAGEN1'
```

Figure 1.13  
Example 6

Example 7 shows the usage of the new INDIRECT statement. In this example, the user requests a list of elements that use “DATECOPY” as a copy book, both directly and indirectly. The output will be GENERATE syntax.

### EXAMPLE 7

```
FLIST ELEMENTS *
  WHERE INPUT COMPONENT EQUALS 'DATECOPY' INDIRECT
  BUILD ACTION GENERATE.
```

Figure 1.14  
Example 7

## Mixed Search Examples

The following examples show syntax for performing searches with different mixes of WHERE clauses.

Example 8 requests a list of all elements in the FastLIST database that have an input component of FLWORK and a Current CCID of ADD\*. In addition, the list is limited to elements in processor group PGRP.

### EXAMPLE 8

```
FLIST ELEMENTS *
  WHERE INPUT COMPONENT EQUAL FLWORK
  WHERE CCID EQUAL ADD*
  WHERE PROCESSOR GROUP EQUAL PGRP
.
```

*Figure 1.15*  
*Example 8*

Example 9 requests a list of all elements in any inventory location with an input component that uses dataset SYS1.MACLIB that were generated between 1/1/92 and 10/20/94.

### EXAMPLE 9

```
FLIST ELEMENTS *
  WHERE INPUT COMPONENT EQUAL (*)
  DSNAME 'SYS1.MACLIB'
  WHERE GENERATE FROM DATE EQUAL 92/01/01 THROUGH 94/10/20
.
```

*Figure 1.16*  
*Example 9*

Example 10 requests a list of elements where delta CCID equals ADD-LVL01, where the delta date is between 10/1/94 and 11/1/94. Because SHOW LEVELS has not been coded, this search will show only the most recent delta level. A typical use of this type of syntax would be to plan for packaging a release at the highest level that contains a fix.

### EXAMPLE 10

```
FLIST ELEMENTS *
  WHERE CCID OF DELTA EQUAL 'ADD-LVL01'
  WHERE DELTA FROM DATE EQUAL 94/10/01 THROUGH 94/11/01
.
```

*Figure 1.17*  
*Example 10*

## Output Examples

The following example shows the different output options available with FLIST syntax. The syntax stream contains the same search arguments with each of the four different output coding options. The outputs that follow the syntax stream show the formats resulting from these four different options. Also remember that regardless of the output option, a standard FLIST execution report will be written to SYSOUT=\*.

```
FLIST ELEMENTS ZTEST*
  FROM STAGE NUM 1
  WHERE INPUT COMPONENT EQUAL FLWORK
.
FLIST ELEMENTS ZTEST*
  FROM STAGE NUM 1
  WHERE INPUT COMPONENT EQUAL FLWORK
BUILD ACTION MOVE
.
FLIST ELEMENTS ZTEST*
  FROM STAGE NUM 1
  WHERE INPUT COMPONENT EQUAL FLWORK
BUILD REPORT
.
FLIST ELEMENTS ZTEST*
  FROM STAGE NUM 1
  WHERE INPUT COMPONENT EQUAL FLWORK
BUILD DATA
.
```

*Figure 1.18  
Syntax Stream*

## Output from Default NO BUILD Statement

```
SET FROM ENV 'TEST' SYS 'SYSA' SUBSYS 'SUBA' TYPE 'ASM'
STAGE NUM 1.
  &&ACTION ELEMENT ZTEST1      VERSION 01 LEVEL 02 .
  &&ACTION ELEMENT ZTEST2      VERSION 01 LEVEL 00 .
  &&ACTION ELEMENT ZTEST4      VERSION 01 LEVEL 01 .
  &&ACTION ELEMENT ZTEST5      VERSION 01 LEVEL 03 .
  &&ACTION ELEMENT ZTEST6      VERSION 01 LEVEL 00 .
  &&ACTION ELEMENT ZTEST7      VERSION 01 LEVEL 01 .
  &&ACTION ELEMENT ZTEST8      VERSION 01 LEVEL 00 .
17:09:50 FST1106I FLIST - SCL BUILD FOR FLIST COMPLETED.
```

*Figure 1.19  
Default "NO BUILD Statement" Output*

## Output from BUILD ACTION MOVE Statement

```
SET FROM ENV 'TEST' SYS 'SYSA' SUBSYS 'SUBA' TYPE 'ASM'  
STAGE NUM 1.  
MOVE      ELEMENT  ZTEST1      VERSION 01 LEVEL 02 .  
MOVE      ELEMENT  ZTEST2      VERSION 01 LEVEL 00 .  
MOVE      ELEMENT  ZTEST4      VERSION 01 LEVEL 01 .  
MOVE      ELEMENT  ZTEST5      VERSION 01 LEVEL 03 .  
MOVE      ELEMENT  ZTEST6      VERSION 01 LEVEL 00 .  
MOVE      ELEMENT  ZTEST7      VERSION 01 LEVEL 01 .  
MOVE      ELEMENT  ZTEST8      VERSION 01 LEVEL 00 .  
17:10:16  FST1106I  FLIST - SCL BUILD FOR FLIST COMPLETED.
```

*Figure 1.20*  
*BUILD ACTION MOVE Output Example*

# User Action Support

---

The purpose of the User Action is to extend FastLIST to support user-built functionality. Once a list of Endeavor elements has been built, the users can enter the name of a user function ( an ISPF skeleton name) provided by their systems administrator. The model User Action skeleton is called CIGUSER0. This skeleton has no functionality, is it used only as a model. Examples of User Action processing would include FDELETE, FLIST, FLOAD, and more.

The User Action is only supported in Batch mode. If the User Action is selected in ISPF, then FastLIST automatically sets the execution mode to batch, regardless of the prompt panel setting.

The model User Action skeleton is displayed below. Note that it is meant only as a starting point for building other User Action, i.e., it has no current functionality.

## CIGUSER0

```
)CM                                02/15/99
)CM
)CM  NAME.....: CIGUSER0
)CM  FUNCTION: MODEL USER ACTION SKELETON.
)CM  INVOKED FROM THE FASTLIST USER ACTION GROUP COMMAND.
)CM  THIS SKELETON ASSUMES:
)CM  1. A LIST HAS BEEN CREATED BY FASTLIST
)CM
)CM  -----
)CM  * * *   N O T I C E   * * *
)CM  THIS PROGRAM IS A PROPRIETARY PRODUCT OF CHICAGO INTERFACE
)CM  GROUP, INC. @ COPYRIGHT 1999 CHICAGO INTERFACE GROUP, INC.
)CM  ALL RIGHTS RESERVED.
)CM  -----
)CM  **                                **
)CM  ** PRODUCT INSTALLATION/SETUP ISSUES **
)CM  **                                **
)CM  THE FOLLOWING IS A LIST OF MODIFICATIONS REQUIRED DURING PRODUCT
)CM  INSTALLATION AND INITIAL SETUP:
)CM  1. MODIFY THE HIGH LEVEL QUALIFIERS CALLED
)CM  FLHQ1.FLHQ2 TO REFLECT THE NAMING STANDARD USED DURING
)CM  PRODUCT INSTALLATION.
)CM  2. THE FOLLOWING LOAD MODULES MUST BE LOCATED IN EITHER THE
)CM  STEPLIB DD STATEMENTS OR IN A LINK LISTED LIBRARY:
)CM  CIGFEXEC AND CIGINI.
)CM  3. IF CIGFEXEC AND CIGINI ARE LOCATED IN A LINK LISTED LIBRARY
)CM  THEN THE STEPLIB DD STATEMENTS SHOULD BE REMOVED FROM THIS
)CM  JCL.
)CM  -----
)CM  THE FOLLOWING IS A LIST OF SYMBOLIC VARIABLES USED IN FILE TAILORING:
)CM  1. ACTION NAME = &VACTION
)CM  2. SEGMENT ID = &VSEGID
)CM  3. TABLE INPUT = ENDEVOR: &VLENV, &VLSYS, &VLSBS, &VLTY, &VLSTG
)CM  DATASET: &VLDSN
)CM  4. VERSION/LEVEL = &LLL, &LVV
)CM  5. MEMBER NAME = &VLELE
)CM  6. JOB CARDS = &C1BJC1 (2,3,4)
)CM  continued on next page...
```

```

)CM -----
)SEL &C1BJC1 -= &Z

&C1BJC1
)ENDSEL
)SEL &C1BJC2 -= &Z
&C1BJC2
)ENDSEL
)SEL &C1BJC3 -= &Z
&C1BJC3
)ENDSEL
)SEL &C1BJC4 -= &Z
&C1BJC4
)ENDSEL
/* -----*
/* NAME: CIGUSER0 *
/* PURPOSE: MODE USER ACTION SKELETON *
/* -----*
//GCL EXEC PGM=USERPRGM <===== PROVIDE PROGRAM NAME
//STEPLIB DD DSN=USER1.USER2.LOADLIB,DISP=SHR
//SYSIN DD SYSOUT=*
* -----*
* FORMAT THE DESIRED SYNTAX USING TABLE VARIABLES *
* -----*
)DOT &VTBL4DOT
)SEL &VPWORK -= EXCLUDED
XXXX ELEMENTS &VLELE
FROM ENV &VLENV
SYSTEM &VLSYS
SUBSYSTEM &VLSBS
TYPE &VLTYP
STAGE NUMBER *
.
)ENDSEL
)ENDDOT

```

Figure 1.21 Sample User Action Skeleton



# Build Report Option

---

Output from BUILD REPORT Statement

DATE 94/10/24 TIME 17:10:16 FASTLIST UTILITY, RELEASE 7.0								
ELEMENT	ENVIRONMENT	SYSTEM	SUBSYS	TYPE	STGNUM	VV	LL	
ZTEST1	TEST	SYSA	SUBA	ASM	1	01	02	
ZTEST2	TEST	SYSA	SUBA	ASM	1	01	00	
ZTEST4	TEST	SYSA	SUBA	ASM	1	01	01	
ZTEST5	TEST	SYSA	SUBA	ASM	1	01	03	
ZTEST6	TEST	SYSA	SUBA	ASM	1	01	00	
ZTEST7	TEST	SYSA	SUBA	ASM	1	01	01	
ZTEST8	TEST	SYSA	SUBA	ASM	1	01	00	
17:10:16	FST1107I	FLIST - REPORT PROCESSING COMPLETED.						

*Figure 1.22*  
*BUILD REPORT Output Example*

# Build Data Option

---

## Output From Build DATA Statement

There are very few messages associated with the data extraction request. The data will be written into the dataset pointed to by the CIGDATA DD.

```
17:12:57  FST1109I  FLIST - DATA EXTRACTION COMPLETED.
```

*Figure 1.23*  
*BUILD DATA Output Example*





# *FastLIST 7.0*

## Chapter 2:

### Configuration XREF Reports

This chapter contains:

- A step-by-step explanation of how to use the FastLIST XREF Reports utility.
- Reports syntax.
- Syntax and report format examples.
- JCL for Reports.



## Message Log

---

The following is an example of the XREF Reports execution report. It is the same as the FLIST execution report and will be generated when you run any report.

```
13:17:08 FST0242I'CIGIN01'INITIZATION MODULE LOADED FROM DD //CIGINI
13:17:08 FST0245I PRIMARY VSAM DATASET NAME 'CIGT.QADATA'.
13:17:08 FST0246I ALTERNATE INDEX VSAM DATASET NAME 'CIGT.QADATA'.
13:17:08 FST0244I OPTIONS IN EFFECT. COMPS='Y' AND CCIDS='Y'.
13:17:08 FST0251I FASTLIST APPLICATION MODULES BEING LOADED FROM

DATE 96/05/13 TIME 13:17:10 FASTLIST UTILITY, RELEASE 7.0 PAGE 1

13:17:10 FST1102I PARSER BEGINS
13:17:10 FST0020I REPORT CCIDS XREF
13:17:10 FST0020I .
13:17:10 FST0020I REPORT COMPONENTS XREF
13:17:10 FST0020I .
13:17:10 FST0020I REPORT DATASETS XREF
13:17:10 FST0020I .
13:17:10 FST1103I PARSER ENDS, RC=0000
13:17:10 FST1111I STARTING DATABASE QUERY.
13:19:24 FST1220I REPORT OR REPORTS SUCCESSFULLY CREATED.
13:21:43 FST1220I REPORT OR REPORTS SUCCESSFULLY CREATED.
13:21:43 FST1220I REPORT OR REPORTS SUCCESSFULLY CREATED.
```

*Figure 2.1*  
*XREF Reports Execution Report*

# Introduction to FastLIST XREF Reports

---

## Overview

FastLIST's XREF Reports utility creates ten different reports that cross-reference part or all of your inventory with key component and element footprint data. These reports are:

- Element by CCID and CCID by element.
- Element by component and component by element
- Element by dataset and dataset by element.
- Element by objects and objects by element.
- Elements by processor by processor group.
- Elements by processor group by processor.

The Reports utility provides a tool for impact analysis that goes beyond a unique element list. You can list CCIDs, components, objects, and dataset names as a function of each individual element. You can also get a three-way report depicting elements as a function of processors and processor groups. The comprehensive information provided by these reports allows better configuration and release management by showing you how your inventory fits together.

The first part of this chapter explains the mechanics of running the XREF Reports utility. JCL and syntax rules are presented in detail. The latter part of this chapter provides a usage example and sample formats for most of the various reports.

# Submitting Reports

---

## Using ISPF Front-end

```
----- FASTLIST WORKBENCH -----
OPTION ==>                                SCROLL ==> PAGE
                                     ENTER "/" TO SELECT OPTION:
LIST ELEMENT. .                          LIST TYPE . E (E/X/B)  _ SHOW FILTERS  _ APPEND
WHERE COMPONENT.                         COMP TYPE . X (I/O/P/X)
=====
ENTER "/" TO SELECT FUNCTION BELOW OR LINE COMMAND:
_ GROUP ACTION    _ JCL    _ MSGS    _ REPORTS    _ PREFERENCES
=====
"/" ELEMENT          TYPE          ENVIRON  S  SYSTEM    SUBSYS    (VV.LL)
-----
```

*Figure 2.2  
The Main Panel*

This is the first panel that users see upon invoking the FastLIST ISPF application. From this panel users can perform:

- impact analysis,
- SCL creation,
- element edits,
- report requests, and
- Endeavor line commands.

Regardless of which function is performed, the user will always be returned to this main panel.

# Reports Syntax

---

## Report Syntax Summary

The following table shows all of the XREF Reports syntax. See Chapter 3 *FastLIST Query Tips and Techniques*, for more information on how to code syntax.

REPORT	CCIDS	{BY ELEMENT   XREF}
	COMPONENTS	{BY ELEMENT   XREF}
	DATASETS	{BY ELEMENT   XREF}
	OBJECTS	{BY ELEMENT   XREF}
	PROCESSORS	{BY PROCESSOR GROUP   XREF }
	PROCESSOR GROUP	{BY PROCESSOR   XREF}
	ELEMENTS	{BY CCIDS   BY COMPONENTS   BY DATASETS   BY OBJECTS   XREF}
FILTERS	ELEMENTS	<i>element-name</i>
	THROUGH	<i>element-name</i>
	FROM	ENVIRONMENT <i>env-name</i>
		SYSTEM <i>system-name</i>
		SUBSYSTEM <i>subsystem-name</i>
		TYPE <i>type-name</i>
		STAGENUM <i>stage-number</i>
	WHERE	CCID OF {CURRENT/LAST/RET/GEN/DELTA/ALL} EQUAL <i>ccid</i>
	WHERE	{[RELATED]INPUT [RELATED]OUTPUT PROCESSOR ALL} COMPONENT EQUAL <i>component-name</i>
		THROUGH <i>component-name</i>
	ENVIRONMENT <i>env-name</i>	
	SYSTEM <i>system-name</i>	
	SUBSYSTEM <i>subsystem-name</i>	
	TYPE <i>type-name</i>	
	STAGENUMBER <i>stage-number</i>	
	VERSION <i>version</i>	
	LEVEL <i>level</i>	
	FILE <i>ddname</i>	
	DSNAME <i>dataset-name</i>	
WHERE	OBJECT EQUAL ' <i>object-name</i> ' COMMENT EQUAL ' <i>comment</i> '	
WHERE	PROCESSOR GROUP EQUAL ' <i>processor-group</i> '	
	.	

Figure 2.3  
XREF Reports Syntax Summary

# Statement Rules

---

## Report Selection Clause

<b>REPORT</b>	<b>CCIDS</b>	<b>{BY ELEMENT   XREF}</b>
	<b>COMPONENTS</b>	<b>{BY ELEMENT   XREF}</b>
	<b>DATASETS</b>	<b>{BY ELEMENT   XREF}</b>
	<b>OBJECTS</b>	<b>{BY ELEMENT   XREF}</b>
	<b>PROCESSORS</b>	<b>{BY PROCESSOR GROUP   XREF}</b>
	<b>PROCESSOR GROUP</b>	<b>{BY PROCESSOR   XREF}</b>
	<b>ELEMENTS</b>	<b>{BY CCIDS  </b>
		<b>BY COMPONENTS  </b>
		<b>BY DATASETS  </b>
		<b>BY OBJECTS  </b>
		<b>XREF}</b>

The REPORT syntax statement indicates to the XREF Reporter which reports to create. This is the only syntax required to run the Reports utility. All other syntax is optional. There are ten separate possible reports.

- Element by CCID and CCID by element.
- Element by component and component by element
- Element by dataset and dataset by element.
- Element by objects and objects by element.
- Elements by processor group by processor.
- Elements by processor by processor group.

You can create a single report, for example by coding REPORT DATASETS BY ELEMENT, or you can use an XREF couple. Using an XREF couple in any REPORT statement except a REPORT ELEMENTS statement will generate the two related reports. For instance, coding REPORT CCIDS XREF will result in two CCID reports: CCID by element and element by CCID. Coding ELEMENTS XREF creates the entire set of eight reports that reference elements.

FastLIST treats each report as a separate entity, independent of other lists in other reports. For instance, if a user requests both REPORT CCIDS BY ELEMENT and REPORT COMPONENTS BY ELEMENT, these two reports will be processed separately. Elements may appear on either or both of the reports.

## Element Filter Clause

### **FILTER ELEMENTS** *element-name*

Indicates the 1 -10 character element name or mask to be reported. May be wild carded.

**FROM**           **ENVIRONMENT**   *env-name*  
**SYSTEM**        *system-name*  
**SUBSYSTEM**     *subsystem-name*  
**ELEMENT**      *element-name*  
**TYPE**           *type-name*  
**STAGE NUMBER** *stage-number*

The FROM parameters limit the elements reported to the specified Endeavor inventory location. All of these parameters are optional, and all of them accept wild carding.

## Where CCID Statements

**WHERE CCID OF {CURRENT | LAST | RET | GEN | DELTA | ANY}**  
**EQUALS** *ccid*

Indicates that only records with a valid CCID from the specified level be reported. If no "where CCID" clause is coded, the CCID type of ANY is in effect. All CCIDs will be reported.

## Where Component Statements

**WHERE**        {**[RELATED] INPUT** | **[RELATED]OUTPUT** | **PROCESSOR** | **ANY**} **COMPONENT EQUALS** *component-name*

Indicates that only records that meet the component criteria be reported. If no "where component" clause is coded, then the component type of ANY is in effect. All components will be reported. For users of Endeavor 3.7 and beyond, component type may also be related components.

To tailor the WHERE COMPONENT criteria further, you may specify more detailed information about the component's Endeavor or dataset location using the following optional component key words:

<b>ENVIRONMENT</b>	<i>env-name</i>
<b>SYSTEM</b>	<i>system-name</i>
<b>SUBSYSTEM</b>	<i>subsystem-name</i>
<b>ELEMENT</b>	<i>element-name</i>
<b>TYPE</b>	<i>type-name</i>
<b>STAGE NUMBER</b>	<i>stage-number</i>
<b>VERSION</b>	<i>version</i>
<b>LEVEL</b>	<i>level</i>
<b>FILE</b>	<i>ddname</i>
<b>DSNAME</b>	<i>dataset-name</i>

### Where Object Statement

The use of the WHERE OBJECT or COMMENT clause directs Reports to report elements correlated to the specified object. The *object name* and *comment* are 1 to 70 character values.

**WHERE      OBJECT EQUAL '*object name*'**  
**COMMENT EQUAL '*comment*'**

### Where Processor Group

The use of the WHERE PROCESSOR GROUP statement is used only by the PROCESSOR by PROCESSOR GROUP reports. This statement will be ignored for other report types. Input to this statement is a one to eight character processor group name.

**WHERE      PROCESSOR GROUP EQUAL '*processor-group*'**

# Aliases

---

FastLIST XREF Reports accepts all of the syntax listed above, as well as an abbreviation consisting of the first three letters only of any keyword. All other valid aliases are listed in table 2.4.

Keyword	Valid Aliases
<b>CCIDS</b>	<b>CCID</b>
<b>COMPONENT</b>	<b>COMP, COMPONENTS</b>
<b>DATASETS</b>	<b>DATASET</b>
<b>ELEMENTS</b>	<b>ELEMENT</b>
<b>EQUAL</b>	<b>EQUALS, EQ, =</b>
<b>FILTER</b>	<b>FILTERS</b>
<b>FILE</b>	<b>DDNAME</b>
<b>PROCESSOR</b>	<b>PROCESSORS</b>
<b>OBJECTS</b>	<b>OBJECT</b>
<b>THROUGH</b>	<b>THRU</b>

*Figure 2.4  
XREF Reports Aliases Syntax and Format Examples*

# Sample Reports

## Report Elements By CCID

This lists each unique CCID along with all of the associated elements that satisfy syntax criteria.

```
1 DATE 98/01/02 TIME 17:19:31 FASTLIST UTILITY, RELEASE 2.2 PAGE 2
      E L E M E N T S   B Y   C C I D S
```

ELEMENT	VV.LL	ENVIRONMENT	SYSTEM	SUBSYSTEM	TYPE	STG
ELEMENTS USING CCID 'ADD1013'						
ZTEST1	01.03	TEST	SYSA	SUBA	ASM	1
ZTEST4	01.01	TEST	SYSA	SUBA	ASM	1
ZTEST8	01.00	TEST	SYSA	SUBA	ASM	1
ELEMENTS USING CCID 'ALKDJFAL'						
ZTESTMAC	01.00	TEST	SYSA	SUBA	MAC	1
ELEMENTS USING CCID 'COLLTEST'						
ZTEST1	01.03	TEST	SYSA	SUBA	ASM	1
ZTEST1	01.01	TEST	SYSA	SUBA	ASM	2
ELEMENTS USING CCID 'COLLTEST0906'						
ZTEST2	01.00	TEST	SYSA	SUBA	ASM	1
ZTEST2	01.00	TEST	SYSA	SUBA	ASM	2
ZTEST3	01.00	TEST	SYSA	SUBA	ASM	1
ZTEST3	01.00	TEST	SYSA	SUBA	ASM	2
ZTEST4	01.01	TEST	SYSA	SUBA	ASM	1
ZTEST4	01.00	TEST	SYSA	SUBA	ASM	2
ZTEST5	01.03	TEST	SYSA	SUBA	ASM	1
ELEMENTS USING CCID 'COLLTEST0907'						
ZTEST6	01.00	TEST	SYSA	SUBA	ASM	1
ZTEST6	01.00	TEST	SYSA	SUBA	ASM	2
ZTEST7	01.01	TEST	SYSA	SUBA	ASM	2
ELEMENTS USING CCID 'FLOADR11CP'						
ZTEST2	01.00	TEST	SYSA	SUBA	ASM	1
ZTEST3	01.00	TEST	SYSA	SUBA	ASM	1
ZTEST4	01.01	TEST	SYSA	SUBA	ASM	1
ZTEST5	01.03	TEST	SYSA	SUBA	ASM	1
ZTEST6	01.00	TEST	SYSA	SUBA	ASM	1
ZTEST7	01.01	TEST	SYSA	SUBA	ASM	1
ZTEST8	01.00	TEST	SYSA	SUBA	ASM	1
ELEMENTS USING CCID 'MOVE0926'						
ZTEST2	01.00	TEST	SYSA	SUBA	ASM	2
ZTEST3	01.00	TEST	SYSA	SUBA	ASM	2
ZTEST4	01.00	TEST	SYSA	SUBA	ASM	2
ZTEST6	01.00	TEST	SYSA	SUBA	ASM	2
ZTEST7	01.01	TEST	SYSA	SUBA	ASM	2
ELEMENTS USING CCID 'QA0922'						
ZTEST7	01.01	TEST	SYSA	SUBA	ASM	1
ZTEST7	01.01	TEST	SYSA	SUBA	ASM	2

ELEMENTS USING CCID 'QA0924'						
	ZTESTMAC	01.00 TEST	SYSA	SUBA	MAC	1
1	DATE 98/01/02 TIME 17:19:31		F A S T L I S T U T I L I T Y, RELEASE 2.2 PAGE			3
			E L E M E N T S B Y C C I D S			
ELEMENT	VV.LL ENVIRONMENT	SYSTEM	SUBSYSTEM	TYPE	STG	
ELEMENTS USING CCID 'QA0926'						
	ZTEST5	01.03 TEST	SYSA	SUBA	ASM	1
ELEMENTS USING CCID 'REL11'						
	ZTEST1	01.01 TEST	SYSA	SUBA	ASM	2
ELEMENTS USING CCID 'TEST'						
	ZTEST1	01.01 TEST	SYSA	SUBA	ASM	2
ELEMENTS USING CCID 'XFER0924'						
	ZTEST7	01.01 TEST	SYSA	SUBA	ASM	2

*Figure 2.5  
Elements By CCID Report Example*

## Report CCIDs By Element

This report lists each element that satisfies the Reports syntax criteria along with all associated CCIDs.

```
DATE 97/06/13 TIME 14:22:38 F A S T L I S T U T I L I T Y, RELEASE 7.0 PAGE 1
                                C C I D S U S E D B Y E L E M E N T S
                                CCID          LEVEL          DATE          TIME          USERID
CCIDS USED BY ELEMENT ZTEST1 01.02 TEST/SYSA/SUBA/ASM/1
ADD1013          01.02 (C)      94/10/13      09:04          CIG01X
COLLTEST         01.01          94/09/04      21:52          CIG01E
REL11            GENERATE, LAST ACTION
CCIDS USED BY ELEMENT ZTEST1 01.01 TEST/SYSA/SUBA/ASM/2
COLLTEST         01.01 (C)      94/09/04      21:52          CIG01E
01.00            94/09/04      21:35          CIG01E
REL11            GENERATE, LAST ACTION
CCIDS USED BY ELEMENT ZTEST2 01.00 TEST/SYSA/SUBA/ASM/1
COLLTEST0906     01.00 (C)      94/09/06      19:42          CIG01
FLOADR11CP       GENERATE, LAST ACTION
CCIDS USED BY ELEMENT ZTEST2 01.00 TEST/SYSA/SUBA/ASM/2
COLLTEST0906     GENERATE
01.00 (C)        94/09/06      19:42          CIG01
MOVE0926         LAST ACTION
CCIDS USED BY ELEMENT ZTEST3 01.00 TEST/SYSA/SUBA/ASM/1
COLLTEST0906     01.00 (C)      94/09/06      20:44          CIG01
FLOADR11CP       GENERATE, LAST ACTION
```

*Figure 2.6*  
*CCIDs By Element Report Example*

## Report Elements By Component

This lists each component along with all of the elements that satisfy the syntax criteria and that use the component.

1 DATE 98/01/02 TIME 17:19:31 F A S T L I S T U T I L I T Y, RELEASE 2.2 PAGE 1							
ELEMENT	ENVIRONMENT	SYSTEM	SUBSYS	TYPE	STG	VV.LL	USERID
ZTEST1	TEST	SYSA	SUBA	ASM	1	01.03	CIG03ABG
ZTEST2	TEST	SYSA	SUBA	ASM	1	01.00	CIG01N
ZTEST3	TEST	SYSA	SUBA	ASM	1	01.00	CIG01N
ZTEST4	TEST	SYSA	SUBA	ASM	1	01.01	CIG01N
ZTEST5	TEST	SYSA	SUBA	ASM	1	01.03	CIG01N
ZTEST6	TEST	SYSA	SUBA	ASM	1	01.00	CIG01N
ZTEST7	TEST	SYSA	SUBA	ASM	1	01.01	CIG01N
ZTEST8	TEST	SYSA	SUBA	ASM	1	01.00	CIG01N
ZTEST1	TEST	SYSA	SUBA	ASM	2	01.01	CIG03A
ZTEST2	TEST	SYSA	SUBA	ASM	2	01.00	
ZTEST3	TEST	SYSA	SUBA	ASM	2	01.00	
ZTEST4	TEST	SYSA	SUBA	ASM	2	01.00	
ZTEST6	TEST	SYSA	SUBA	ASM	2	01.00	
ZTEST7	TEST	SYSA	SUBA	ASM	2	01.01	
ZTESTMAC	TEST	SYSA	SUBA	MAC	1	01.00	CIG01A

*Figure 2.7  
Elements by Component Report Example*

## Report Components By Element

This report lists each element that satisfies the Reports syntax criteria along with all associated components.

```

DATE 97/05/13 TIME 12:12:34  F A S T L I S T  U T I L I T Y,  RELEASE 7.0  PAGE 1
                                C O M P O N E N T S   U S E D   B Y   E L E M E N T S

ELEMENT          VV.LL ENVIRONMENT SYSTEM      SUBSYSTEM  TYPE      STG
COMPONENTS USED BY ELEMENT ZTEST1 01.02  TEST/SYSA/SUBA/ASM/1
$CALL           (I) 01.01 TEST      SYSA      SUBA      MAC       1
$CHAIN0         (I) 01.00 TEST      SYSA      SUBA      MAC       1
$CHAIN1         (I) 01.00 TEST      SYSA      SUBA      MAC       1
$CHAIN3         (I) 01.00 TEST      SYSA      SUBA      MAC       1
$CHAIN4         (I) 01.00 TEST      SYSA      SUBA      MAC       1
$CHAIN9         (I) 01.00 TEST      SYSA      SUBA      MAC       1
$CPOOL          (I) 01.00 TEST      SYSA      SUBA      MAC       1
$DYNAM          (I) 01.00 TEST      SYSA      SUBA      MAC       1
$ENTRY          (I) 01.00 TEST      SYSA      SUBA      MAC       1
$EXIT           (I) 01.01 TEST      SYSA      SUBA      MAC       1
$GETLEN         (I) 01.00 TEST      SYSA      SUBA      MAC       1
$REGS           (I) 01.00 TEST      SYSA      SUBA      MAC       1
$STACK          (I) 01.00 TEST      SYSA      SUBA      MAC       1
$STAMP          (I) 01.00 TEST      SYSA      SUBA      MAC       1
$STKDS         (I) 01.01 TEST      SYSA      SUBA      MAC       1
$TITLE          (I) 01.00 TEST      SYSA      SUBA      MAC       1
ASMLNKT1        (P) 01.00 TEST      SYSA      SUBA      PROCESS   1
FLWORK          (I) 01.01 TEST      SYSA      SUBA      MAC       1
INIDSECT        (I) 01.00 TEST      SYSA      SUBA      MAC       1
LVLDSECT        (I) 01.00 TEST      SYSA      SUBA      MAC       1
ZTEST1          (I) 01.02 TEST      SYSA      SUBA      ASM       1
ZTEST1          (O) 01.02 TEST      SYSA      SUBA      ASM       1
$$ENTRY         (I) CIGT.ENDTEST.OBJLIB1
$$EXIT          (I) CIGT.ENDTEST.OBJLIB1
ZTEST1          (I) CIGT.ENDTEST.LOADLIB1
    
```

Figure 2.8  
Components by Element Report Example

## Report Elements By Dataset

This report lists each dataset along with all elements which use each dataset and which meet the Reports syntax criteria.

```
DATE 97/05/13 TIME 12:20:50  F A S T L I S T  U T I L I T Y,  RELEASE 7.0  PAGE 1
                                E L E M E N T S   B Y   D S N A M E S
ELEMENT          VV.LL ENVIRONMENT SYSTEM      SUBSYSTEM  TYPE          STG
ELEMENTS LOADED FROM INPUT DATASET CIGT.ENDTEST.LOADLIB1
ZTEST1
MEMBERS WRITTEN TO OUTPUT DATASET CIGT.ENDTEST.LOADLIB1
ZTEST1          01.02 TEST          SYSA          SUBA          ASM           1
ZTEST2          01.00 TEST          SYSA          SUBA          ASM           1
ZTEST4          01.01 TEST          SYSA          SUBA          ASM           1
ZTEST5          01.03 TEST          SYSA          SUBA          ASM           1
ZTEST6          01.00 TEST          SYSA          SUBA          ASM           1
ZTEST7          01.01 TEST          SYSA          SUBA          ASM           1
ZTEST8          01.00 TEST          SYSA          SUBA          ASM           1
ELEMENTS LOADED FROM INPUT DATASET CIGT.ENDTEST.OBJLIB1
$$ENTRY
$$EXIT
ZTEST1          01.02 TEST          SYSA          SUBA          ASM           1
ZTEST2          01.00 TEST          SYSA          SUBA          ASM           1
ZTEST3          01.00 TEST          SYSA          SUBA          ASM           1
ZTEST4          01.01 TEST          SYSA          SUBA          ASM           1
```

*Figure 2.9  
Elements by Dataset Report Example*

## Report Datasets By Element

This report lists each element that satisfies the Reports syntax criteria along with all datasets used by each element.

```
DATE 97/05/13 TIME 12:28:59   F A S T L I S T   U T I L I T Y,  RELEASE 7.0   PAGE 1
                        D S N A M E S   U S E D   B Y   E L E M E N T S

DATASETS USED BY ELEMENT ZTEST1 01.02  TEST/SYSA/SUBA/ASM/1
(I)  CIGT.ENDTEST.LOADLIB1
(I)  CIGT.ENDTEST.OBJLIB1
(I)  CIGT.ENDTEST.SRCLIB1
(O)  CIGT.ENDTEST.LOADLIB1
(O)  CIGT.ENDTEST.OBJLIB1

DATASETS USED BY ELEMENT ZTEST1 01.01  TEST/SYSA/SUBA/ASM/2
(I)  CIGT.ENDTEST.SRCLIB1
(O)  CIGT.ENDTEST.OBJLIB1

DATASETS USED BY ELEMENT ZTEST2 01.00  TEST/SYSA/SUBA/ASM/1
(I)  CIGT.ENDTEST.OBJLIB1
(I)  CIGT.ENDTEST.SRCLIB1
(O)  CIGT.ENDTEST.LOADLIB1
(O)  CIGT.ENDTEST.OBJLIB1

DATASETS USED BY ELEMENT ZTEST2 01.00  TEST/SYSA/SUBA/ASM/2
(I)  CIGT.ENDTEST.SRCLIB1
(O)  CIGT.ENDTEST.OBJLIB1
```

*Figure 2.10  
Datasets by Element Report Example*

## Elements By Processor By Processor Group

This report lists elements by processor and processor group.

ELEMENT	VV.LL ENVIRONMENT	SYSTEM	SUBSYSTEM	TYPE	STG	GEN_DATE	TIME
DATE 97/05/13 TIME 11:39:05 FASTLIST UTILITY, RELEASE 7.0 PAGE 1							
PROCESSORS BY PROCESSOR GROUP							
PROCESSOR: ASMLNKT1 PROCESSOR GROUP: PGASM 01.00 TEST/SYSA/SUBA/PROCESS/1							
ZTEST1	01.02 TEST	SYSA	SUBA	ASM	1	94/11/25	21:43
ZTEST2	01.00 TEST	SYSA	SUBA	ASM	1	94/11/25	21:43
ZTEST4	01.01 TEST	SYSA	SUBA	ASM	1	94/11/07	19:49
ZTEST5	01.03 TEST	SYSA	SUBA	ASM	1	94/11/07	19:49
ZTEST6	01.00 TEST	SYSA	SUBA	ASM	1	94/11/07	19:49
ZTEST7	01.01 TEST	SYSA	SUBA	ASM	1	94/11/07	19:49
ZTEST8	01.00 TEST	SYSA	SUBA	ASM	1	94/11/07	19:49
PROCESSOR: PGAGEN1 PROCESSOR GROUP: ASM31REN 01.04 TEST/SYSA/SUBA/PROCESS/1							
ZTEST3	01.00 TEST	SYSA	SUBA	ASM	1	94/11/07	19:49
PROCESSOR: PGASMG PROCESSOR GROUP: PGASM 01.12 TEST/SYSA/SUBA/PROCESS/1							
ZTEST1	01.01 TEST	SYSA	SUBA	ASM	2	94/08/08	18:40
ZTEST2	01.00 TEST	SYSA	SUBA	ASM	2	94/08/08	18:40
ZTEST3	01.00 TEST	SYSA	SUBA	ASM	2	94/08/08	18:40
ZTEST4	01.00 TEST	SYSA	SUBA	ASM	2	94/08/08	18:40
ZTEST6	01.00 TEST	SYSA	SUBA	ASM	2	94/08/08	18:40
ZTEST7	01.01 TEST	SYSA	SUBA	ASM	2	94/08/08	18:40

*Figure 2.11  
Elements by Processor by Processor Group Example*

# Elements By Processor Group By Processor

This report lists elements as a function of processor group and then processors.

DATE 97/05/13 TIME 11:39:05 FASTLIST UTILITY, RELEASE 7.0 PAGE 2							
PROCESSOR GROUP BY PROCESSOR							
ELEMENT	VV.LL	ENVIRONMENT	SYSTEM	SUBSYSTEM	TYPE	STG	GEN_DATE
TIME							
PROCESSOR GROUP: ASM31REN PROCESSOR: PGAGEN1 01.04 TEST/SYSA/SUBA/PROCESS/1							
ZTEST3	01.00	TEST	SYSA	SUBA	ASM	1	94/11/07 19:49
PROCESSOR GROUP: PGASM PROCESSOR: ASMLNKT1 01.00 TEST/SYSA/SUBA/PROCESS/1							
ZTEST1	01.02	TEST	SYSA	SUBA	ASM	1	94/11/25 21:43
ZTEST2	01.00	TEST	SYSA	SUBA	ASM	1	94/11/25 21:43
ZTEST4	01.01	TEST	SYSA	SUBA	ASM	1	94/11/07 19:49
ZTEST5	01.03	TEST	SYSA	SUBA	ASM	1	94/11/07 19:49
ZTEST6	01.00	TEST	SYSA	SUBA	ASM	1	94/11/07 19:49
ZTEST7	01.01	TEST	SYSA	SUBA	ASM	1	94/11/07 19:49
ZTEST8	01.00	TEST	SYSA	SUBA	ASM	1	94/11/07 19:49
PROCESSOR GROUP: PGASM PROCESSOR: PGASMG 01.12 TEST/SYSA/SUBA/PROCESS/1							
ZTEST1	01.01	TEST	SYSA	SUBA	ASM	2	94/08/08 18:40
ZTEST2	01.00	TEST	SYSA	SUBA	ASM	2	94/08/08 18:40
ZTEST3	01.00	TEST	SYSA	SUBA	ASM	2	94/08/08 18:40
ZTEST4	01.00	TEST	SYSA	SUBA	ASM	2	94/08/08 18:40
ZTEST6	01.00	TEST	SYSA	SUBA	ASM	2	94/08/08 18:40
ZTEST7	01.01	TEST	SYSA	SUBA	ASM	2	94/08/08 18:40

*Figure 2.12*  
*Elements by Processor Group by Processor Example*



# ***FastLIST 7.0***

## Chapter 3: Query Tips & Techniques

This chapter contains information on optimizing your queries against the FastLIST database. The examples provided use batch syntax, but the suggestions apply to the ISPF interface as well.



# Query Tips and Techniques

---

## FastLIST Database Concepts

FastLIST uses six different types of records to answer queries. These six records are (1) a mini-element record type; (2) a CCID type; (3) input component type; (4) output component type; (5) processor type; and (6) object type. The search criteria specified by the user determines how many calls FLIST will make to the database.

The FastLIST Database is a single-keyed VSAM file. The index allows FastLIST to search for only the necessary record type. This indexing is the key to the "fast" in FastLIST. If the CIGTRACE is active, FastLIST will print an informational message every time it returns from the query engine. The message will have the following format:

**FST0500I DATABASE QUERY RESULTED IN 9999 '*rec-type*'  
RECORDS SENT BACK TO APPLICATION.**

Where 9999 is the number of records returned and *rec-type* is element, CCID, input, output, processor, or object. The number of records returned is not necessarily the number of elements that qualify for the list. FLIST and the ISPF front end must process the records returned into a single qualified list.

## The Records

The following is a short description of each of the record types built and processed by FastLIST:

- The mini-element record is a record that contains the element footprint, all the date and time stamps, and key data about the element record.
- The CCID record contains a unique CCID value, the CCID attributes, level data, and the parent element footprint.
- The input component record contains all information about an input component such as component data and footprint, dataset where it was loaded from, if it is a related element or member, and the parent element footprint.
- The output component record contains all information about an output component such as component name and footprint, dataset where it was stored, if it is a related element or member, and the parent element footprint.
- The processor component record contains all information about the processor used by an element. The contents of the record include what processor processed the element and the parent element footprint.

- The object record contains the object value, type, and the parent element footprint.

## Building the Optimum Search

FLIST syntax provides you with a comprehensive set of query options. When building "where used" syntax, keep in mind the structure of the database. Limit the search criteria to minimize the number of record types queried, and be as specific as possible on the criteria provided. For example, if you are trying to build a list of elements that have a particular input component, use the following search:

```
FLIST ELEMENTS *  
WHERE INPUT COMPONENT EQUAL 'SOMEMAC'.
```

Because the only "where used" data included is INPUT COMPONENT, FastLIST calls the database once. Compare this to the following search:

```
FLIST ELEMENT *  
WHERE INPUT COMPONENT EQUAL (SOMEMAC)  
WHERE CCID EQUAL *.
```

This syntax will perform an indexed search of all elements that use SOMEMAC and then it will perform a sequential search of all CCID records. The two lists will then be combined to build a list containing elements that were on both lists. If every record has some CCID assigned to it, then the list will be the same as the optimum search above, but the extraneous "where CCID" clause caused FastLIST to call the database a second time.

WHERE clauses should rarely contain an \*. It might be necessary when using, for example, a WHERE CCID EQUAL \* if you want only elements that have a current CCID.

**When building the optimum search, the rule is "IF YOU DON'T NEED IT, DON'T CODE IT."**

Another feature of the FastLIST database is that the more fully qualified your syntax information is, the fewer database reads FastLIST will perform. For example, if you want records for element ZTEST1, you can include it in your list by coding ELEMENTS Z\* or ELEMENTS ZTEST1. If you code ELEMENTS ZTEST1, FastLIST will do only one database read. It knows exactly where to go. The less specific ELEMENTS Z\* will result in more reads.

**This leads to the second rule of code optimization. "IF YOU KNOW IT, DON'T WILDCARD IT."**

In summary, two guidelines for building an optimum search:

Always provide FLIST with as much qualified information as possible.  
Avoid '\*' in a WHERE clause, whenever possible.

## Layering Syntax

FLIST allows for complex search combinations, when required. The following is an example of a complex, layered search:

```
FLIST ELEMENTS CIG*
  WHERE INPUT COMPONENT EQ (MAC01,MAC02)
  WHERE OUTPUT COMPONENT EQ (XYZ,ABC)
  WHERE CCID EQUAL (ABC*,FIX*)
  WHERE LASTMOD DATE 94/10/01 THROUGH DATE 94/11/01.
```

This set of syntax would cause several passes through the query engine. The net result of this query would be to ask the following question: what elements start with CIG and use INPUT component MAC01 or MAC02 and create output components XYZ or ABC and have current CCIDS that start with ABC\* or FIX\* and have a last modified date between the given dates.

It is important to understand the difference between the syntax

```
FLIST ELEMENTS CIG*
  WHERE INPUT COMPONENT EQUAL (P1, P2).
```

which asks what elements use input components P1 or P2, and

```
FLIST ELEMENTS CIG*
  WHERE INPUT COMPONENT EQUAL P1
  WHERE INPUT COMPONENT EQUAL P2.
```

which ask what elements use input components P1 and P2. To make these two sets of syntax equal, use the OPTION MATCH EQUAL ANY option on the second set of syntax.

## Stacking Syntax

Like the standard Endeavor LIST action, users can input multiple FLIST sets of syntax. Each of these blocks of syntax are processed as separate entities.

DELTA DATE and DELTA CCIDS

If the database is built collecting delta information, FLIST supports delta date and delta CCID query arguments. If your installation has chosen not to collect deltas, then the query will come back with zero valid records. Also note that FastLIST only collects delta information for delta levels that have an associated CCID. If a delta level does not have a CCID then that information is not stored in the FastLIST database and will not appear on any reports.

**Delta CCID works the same even if you are using the Reverse Delta Endeavor format.**

The purpose of the delta DATE and delta CCID search keywords is to allow you to ask information about source elements that may or may not be current. You may want to build a list of elements where the delta CCID is equal to a release name. The resultant list would contain the highest delta level that contains the CCID. The following syntax is an example of such a request:

```
FLIST ELEMENTS *  
WHERE CCIDS OF DELTA EQUALS REL0001.
```

Note the addition of the OPTIONS SHOW LEVELS would result in a list that shows all the DELTA levels that qualified. All but the most recent DELTA level is commented out of SCL by FastLIST.

## DELTA Versus ANY

There are two ways in which the historical CCID values can be queried, the DELTA or the ANY CCID value. The difference between the two CCID searches is as follows:

1. ANY CCID search will include searching the MCF type CCIDS as well as the DELTA levels where DELTA CCID will only search DELTA levels.
2. The output produced from the ANY CCID search will always include the most current level, and will only include historic levels if OPTIONS SHOW LEVELS has been coded.

For instance, if you want to build a list of elements at the current level that contain the CCID of FIX0001, use the following syntax:

```
FLIST ELEMENTS *  
  WHERE CCID OF ANY EQUALS FIX0001  
  OPTIONS SHOW LEVELS.
```

This will produce a list of elements at the current level that use FIX0001 at ANY level, DELTA or MCF. All lower levels that contain the CCID will be listed but commented out of the SCL.

However, if you want to build a list of elements that contains only the highest level element with a FIX0001, then the appropriate syntax is:

```
FLIST ELEMENTS *  
  WHERE CCID OF DELTA EQUALS FIX0001.
```

The resulting list will contain elements at the actual highest level that contain FIX0001 in the delta. You can then RETRIEVE this list for release management.

## Using the DATE and TIME Ranges

The DATE and TIME search capabilities are powerful FastLIST capabilities. You can now build searches based on GENERATE, LAST MODIFICATION, CURRENT, and DELTA date and time values. The rules for date and time ranges are as follows:

- 1) You must always code at least a DATE value.
- 2) The date must be in the YY/MM/DD format.
- 3) The time must be in the HH:MM format.
- 4) The through date value must be later than the from date value.

The following is an example of various ways to use the WHERE DATE search argument:

```
FLIST ELEMENTS ZTEST1
      WHERE CURRENT DATE EQUAL 04/10/13 TIME 09:04
.
FLIST ELEMENTS ZTEST1
      WHERE GENERATE DATE EQUAL 99/10/01
      THROUGH DATE 04/10/20
.
FLIST ELEMENTS ZTEST1
      WHERE LASTMOD DATE EQUAL 99/10/01 TIME 09:04
      THROUGH DATE 04/10/20
.
FLIST ELEMENTS ZTEST1
      WHERE DELTA DATE EQUAL 99/10/01 TIME 09:04
      THROUGH DATE 04/10/20 TIME 15:00
```

# *FastLIST 7.0*

## Chapter 4: Record Layouts

This chapter contains record layouts used by the Build Data command.



## File Layouts

---

The following file layouts match the five different record types produced by an FLIST BUILD DATA command. The output from the BUILD DATA command is written into the sequential file pointed to by the CIGDATA ddname. There are five different layouts representing the five different types of records. These layouts can be found in the flhq1.flhq2.SAMPLIB, downloaded during the FastLIST install. The members in this dataset are:

- FST@EFP
- FST@ICMP
- FST@OCMP
- FST@PCMP
- FST@OBJ

There is no CCID type layout. All CCID information is contained in the FST@EFP record. If the level value in the record is current, then the EFPCFLAG will contain a 'y'. All dates and CCIDs in the record will represent the current, MCF values. If the EFPCFLAG is blank, then the record represents a delta level. In this case, the current date/time will actually contain the delta date/time for the lower level. The Current CCID field will represent the DELTA CCID value.



## FST@ICMP: The Input Component Record

```
MACRO
FST@ICMP
*****
* INPUT COMPONENT DATABASE EXTRACTION LAYOUT *
*****
FST@ICMP DSECT
*
ICMPEYE DS CL2 RECORD ID = C'04'
ICMPEYEC EQU C'04'
ICMPREL# DS CL2 FASTLIST RELEASE NUMBER
ICMPR1M0 EQU C'10' RECORD RELEASE FORMAT
ICMPNAME DS CL10 COMPONENT NAME
*
=== START OF COMP INFO ===
ICMPCENV DS CL8 COMP ENVIRONMENT
ICMPCSYS DS CL8 COMP SYSTEM
ICMPCSBS DS CL8 COMP SUBSYSTEM
ICMPCTYP DS CL8 COMP TYPE
ICMPCSTG DS CL1 COMP STAGE NUMBER
ICMPCVV DS CL2 COMP VERSION
ICMPCLL DS CL2 COMP LEVEL
ICMPREL DS CL1 COMP RELATED(Y/N)
ICMPVAL DS CL1 COMP VALIDATED (Y/N)
ICMPDSN DS CL44 COMPONENT DATASET NAME
ICMPMBR DS CL10 COMPONENT MEMBER NAME
*
=== START OF ELEMENT INFO ===
ICMPELE DS CL10 ELEMENT
ICMPENV DS CL8 ENVIRONMENT
ICMPSYS DS CL8 SYSTEM
ICMPSBS DS CL8 SUBSYSTEM
ICMPTYP DS CL8 TYPE
ICMPSTG# DS CL1 STAGE NUMBER
ICMPVV DS CL2 VERSION NUMBER
ICMPLL DS CL2 LEVEL NUMBER
ICMPDSLN EQU *-FST@ICMP DSECT SIZE
MEND
```

Figure 4.2 The Input Component Record

## FST@OBJT: The Object Record

```
MACRO
    FST@OBJT
*****
*   OBJECT      COMPONENT DATABASE RECORD LAYOUT      *
*****
FST@OBJT DSECT
*
OBJTEYE DS      CL2          RECORD ID = C'07'
OBJTEYEC EQU    C'07'
OBJTREL# DS     CL2          RECORD RELEASE FORMAT
OBJTR1M0 EQU    C'10'
OBJTNAME DS     CL70         MAXIMUM LENGTH IS SEVENTY
OBJTTYPE DS     CL1          OBJECT TYPE
OBJT$OBJ EQU    C'O'         RELATED OBJECT
OBJT$COM EQU    C'C'         RELATED COMMITMENTS
*
OBJTELE DS     CL10         ELEMENT
OBJTENV DS     CL8          ENVIRONMENT
OBJTSYS DS     CL8          SYSTEM
OBJTSBS DS     CL8          SUBSYSTEM
OBJTTYP DS     CL8          TYPE
OBJTSTG# DS    CL1          STAGE NUMBER
OBJTVV DS     CL2          VERSION NUMBER
OBJTLL DS     CL2          LEVEL    NUMBER
*
OBJTDSLN EQU    *-FST@OBJT   DSECT SIZE
MEND
```

*Figure 4.3 The Object Record*

## FST@OCMP: The Output Component

```
MACRO
FST@OCMP
*****
*   OUTPUT COMPONENT DATABASE EXTRACTION LAYOUT   *
*****
FST@OCMP DSECT
*
OCMPEYE DS    CL2          RECORD ID = C'05'
OCMPEYEC EQU  C'05'
OCMPREL# DS   CL2          RECORD RELEASE FORMAT
OCMPR1M0 EQU  C'10'
OCMPNAME DS   CL10        COMPONENT NAME
*
OCMPENV DS    CL8          === START OF COMP INFO ===
OCMPENV DS    CL8          COMP ENVIRONMENT
OCMPCSYS DS   CL8          COMP SYSTEM
OCMPCSBS DS   CL8          COMP SUBSYSTEM
OCMPCTYP DS   CL8          COMP TYPE
OCMPCSTG DS   CL1          COMP STAGE NUMBER
OCMPCVV DS    CL2          COMP VERSION
OCMPCLL DS    CL2          COMP LEVEL
OCMPREL DS    CL1          COMP RELATED(Y/N)
OCMPVAL DS    CL1          COMP VALIDATED (Y/N)
OCMPDSN DS    CL44        COMPONENT DATASET NAME
OCMPMBR DS    CL10        COMPONENT MEMBER NAME
*
OCMPELE DS    CL10        === START OF ELEMENT INFO ===
OCMPELE DS    CL10        ELEMENT
OCMPENV DS    CL8          ENVIRONMENT
OCMPSYS DS    CL8          SYSTEM
OCMP SBS DS   CL8          SUBSYSTEM
OCMP TYP DS   CL8          TYPE
OCMPSTG# DS   CL1          STAGE NUMBER
OCMPVV DS    CL2          VERSION NUMBER
OCMPLL DS    CL2          LEVEL NUMBER
OCMPDSL N EQU *-FST@OCMP  DSECT SIZE
MEND
```

Figure 4.4 The Output Component

## FST@PCMP: The Processor Component Record

```
MACRO
FST@PCMP
*****
* PROCESSOR COMPONENT DATABASE RECORD LAYOUT *
*****
FST@PCMP DSECT
*
PCMPEYE DS CL2 RECORD ID = C'06'
PCMPEYEC EQU C'06'
PCMPREL# DS CL2 RECORD RELEASE FORMAT
PCMPR1M0 EQU C'10'
PCMPNAME DS CL10 PROCESSOR NAME
*
=== START OF PROCESOR INFO =
PCMPCENV DS CL8 COMP ENVIRONMENT
PCMPCSYS DS CL8 COMP SYSTEM
PCMPCSBS DS CL8 COMP SUBSYSTEM
PCMPCTYP DS CL8 COMP TYPE
PCMP CSTG DS CL1 COMP STAGE NUMBER
PCMP CVV DS CL2 COMP VERSION
PCMP CLL DS CL2 COMP LEVEL
*
=== START OF ELEMENT INFO ===
PCMPELE DS CL10 ELEMENT
PCMP ENV DS CL8 ENVIRONMENT
PCMP SYS DS CL8 SYSTEM
PCMP SBS DS CL8 SUBSYSTEM
PCMP TYP DS CL8 TYPE
PCMP STG# DS CL1 STAGE NUMBER
PCMP VV DS CL2 VERSION NUMBER
PCMP LL DS CL2 LEVEL NUMBER
PCMP DSLN EQU *-FST@PCMP DSECT SIZE
MEND
```

*Figure 4.5 The Processor Component Record*

# Table of Figures

---

Figure 1.1	FLIST Step-By-Step.....	1-3
Figure 1.2	CIGJCL03 FLIST Utility.....	1-4
Figure 1.3	Associated ddnames, Descriptions, and Attributes.....	1-5
Figure 1.4	Sample FLIST in a Processor .....	1-6
Figure 1.5	Execution Log Example .....	1-8
Figure 1.6	FastLIST Syntax Summary .....	1-9
Figure 1.7	FastLIST Aliases.....	1-15
Figure 1.8	Example 1.....	1-16
Figure 1.9	Example 2.....	1-16
Figure 1.10	Example 3.....	1-17
Figure 1.11	Example 4.....	1-17
Figure 1.12	Example 5.....	1-18
Figure 1.13	Example 6.....	1-18
Figure 1.14	Example 7.....	1-18
Figure 1.15	Example 8.....	1-19
Figure 1.16	Example 9.....	1-19
Figure 1.17	Example 10.....	1-19
Figure 1.18	Syntax Stream .....	1-20
Figure 1.19	Default "NO BUILD Statement" Output .....	1-20
Figure 1.20	BUILD ACTION MOVE Output Example .....	1-21
Figure 1.21	Sample User Action Skeleton.....	1-23
Figure 1.22	BUILD REPORT Output Example .....	1-25
Figure 1.23	BUILD DATA Output Example .....	1-26
Figure 2.1	XREF Reports Execution Report.....	2-3
Figure 2.2	The Main Panel.....	2-5
Figure 2.3	XREF Reports Syntax Summary.....	2-6
Figure 2.4	XREF Reports Aliases Syntax and Format Examples .....	2-10
Figure 2.5	Elements By CCID Report Example.....	2-12
Figure 2.6	CCIDs By Element Report Example.....	2-13
Figure 2.7	Elements by Component Report Example .....	2-14
Figure 2.8	Components by Element Report Example .....	2-15
Figure 2.9	Elements by Dataset Report Example.....	2-16
Figure 2.10	Datasets by Element Report Example.....	2-17
Figure 2.11	Elements by Processor by Processor Group Example .....	2-18
Figure 2.12	Elements by Processor Group by Processor Example .....	2-19
Figure 4.1	The Mini Element Record.....	4-4
Figure 4.2	The Input Component Record.....	4-5
Figure 4.3	The Object Record.....	4-6
Figure 4.4	The Output Component.....	4-7
Figure 4.5	The Processor Component Record.....	4-8

## Index

- Aliases, 2-10*
    - FLIST, 1-15
    - Reports, 2-10
  - Associated ddnames, Descriptions, and Attributes, 1-5*
  - BUILD ACTION MOVE Output Example, 1-21*
  - BUILD ACTION MOVE Statement, 1-21*
  - BUILD DATA*
    - File Layouts, 4-3, 4-4
  - BUILD DATA Output Example, 1-27*
  - BUILD DATA Statement, 1-26*
  - BUILD REPORT Output Example, 1-25*
  - BUILD REPORT Statement, 1-25*
  - Build Statements, 1-13*
  - CCID Example 1, 1-16*
  - CCID Example 10, 1-19*
  - CCID Example 2, 1-16*
  - CCID Example 3, 1-17*
  - CCID Example 4, 1-17*
  - CCID Example 5, 1-18*
  - CCID Example 6, 1-18*
  - CCID Example 7, 1-18*
  - CCID Example 8, 1-19*
  - CCID Example 9, 1-19*
  - CCID Examples, 1-16*
  - CCIDs By Element Report Example, 2-13*
  - CIGJCL05, 1-4*
  - CIGTRACE*
    - with FLIST, 1-5
  - Components by Element Report Example, 2-15*
  - Datasets by Element Report Example, 2-17*
  - Date and time searches*
    - FLIST, 1-13, 3-8
  - ddnames*
    - FLIST, 1-5
  - Default "NO BUILD Statement" Output, 1-20*
  - Default NO BUILD Statement, 1-20*
  - DELTA*
    - Date and CCIDs, 3-5
  - DELTA Versus ANY, 3-7*
  - Element By CCID Report Example, 2-12*
  - Element by Dataset Report Example, 2-16*
  - Element Clauses, 1-10*
  - Element Filter Clause, 2-8*
  - Elements by Component Report Example, 2-14*
  - Elements by Processor by Processor Group Example, 2-18*
  - Elements by Processor Group by Processor Example, 2-19*
  - Execution Report Example, 1-8*
  - FLIST*
    - Aliases, 1-15
    - ddnames, 1-5
    - JCL to run, 1-4
    - Overview, 1-3
    - Sample FLIST in a Processor, 1-6
    - sample JCL of FLIST in a processor, 1-6
    - Syntax, 1-9
    - Syntax and execution report examples
      - CCID examples, 1-16
      - Component examples, 1-17
      - Mixed search examples, 1-18
      - Output examples, 1-20
    - FLIST Aliases, 1-15*
    - FLIST ELEMENTS Syntax, 1-10***
    - FLIST Execution Report, 1-8*
    - FLIST Step-By-Step, 1-3*
    - FLIST Syntax Summary, 1-9*
    - FST@EFP, 4-4*
    - FST@ICMP, 4-5*
    - FST@OBJT, 4-6*
    - FST@OCMP, 4-7*
    - FST@PCMP, 4-8*
  - JCL*
    - FLIST (CIGJCL05), 1-4
    - sample JCL of FLIST in a processor, 1-6
  - Option Statements, 1-14*
  - Processors*
    - sample JCL of FLIST in a processor, 1-6
  - Records, description, 3-3*
  - Reports*
    - Aliases, 2-10
    - Overview, 2-4
    - Syntax, 2-6
  - Statement Rules, 1-10***
    - FLIST, 1-10***
    - Reports, 2-7
  - Syntax*
    - FLIST, 1-9
    - Reports, 2-6
    - search optimization, 3-4
  - Syntax Stream, 1-20*
  - Syntax, layering, 3-5*
  - Syntax, stacking, 3-5*
  - The Main Panel, 2-5*
  - Time searches*
    - FLIST, 1-13
  - Where CCID Statements, 1-11, 2-8*
  - Where Component Statements, 1-11, 2-8*
-

*Where Generate Failed Statement, 1-13*  
*Where Object Statement, 1-12, 2-9*  
*Where Processor Group, 2-9*  
*Where Processor Group Statement, 1-12*

*XREF Reports Aliases, 2-10*  
*XREF Reports Execution Report, 2-3*  
*XREF Reports Syntax Summary, 2-6*