
Sirius Mods Release Notes Version 6.2

May, 2002



Sirius Software, Inc.
875 Massachusetts Avenue, Suite 21
Cambridge, MA 02139

Telephone: (617) 876-6677
FAX: (617) 234-1200
E-mail: support@sirius-software.com
World Wide Web: <http://sirius-software.com>

November 17, 2004

© 2004 Sirius Software, Inc.

Proprietary Notices

The following products:

- *Fast/Reload*
- *Janus TCP/IP Base*
- *Janus SOAP*
- *Janus Sockets*
- *Janus Web Server*
- *SirFact*

are proprietary products of Sirius Software, Inc.:

Sirius Software, Inc.
875 Massachusetts Avenue, Suite 21
Cambridge, Massachusetts 02139
USA

Model 204™ is a proprietary product of Computer Corporation of America:

Computer Corporation of America
500 Old Connecticut Path
Framingham, Massachusetts 01701
USA

SoftSpy™ is a proprietary product of Information Technology Systems:

Information Technology Systems
95 Wells Avenue
Newton, Massachusetts 02459-3216
USA

Contents

Proprietary Notices	ii
Contents	iii
Chapter 1: Introduction	1
Chapter 2: All or Multiple Products	3
Support for Model 204 Version 5.1	3
Commands now available as operator commands	3
Comments before user 0 parameter line	3
CASE Mixed/Upper command in Full Screen Editor	3
Chapter 3: Janus TCP/IP Base	5
Chapter 4: Janus Sockets	7
Chapter 5: Janus Web Server	9
\$WEB_BROWSER function	9
Janus Web Legacy Support Enhancements	9
WRAPJS parameter for ON rules	10
EXPIRE parameter for ON rules	10
Chapter 6: Janus SOAP	11
Chapter 7: SirFact	13
SIRFACT command	13
SIRFACT DUMP procedure substitution	13
TRACE statement	13
LONG REQUEST dumps	14
Chapter 8: Sirius Functions	15
Character entity substitution enhancements	15
\$FIELD_IMAGE and \$FIELD_LISTI	16
\$list size restrictions eased	16
\$LISTNEWI and \$LISTNEWAI	16
\$LIST_PRINT	16
LONGSTRING datatype	16
SEQ parameter for \$SIRJGET	18

\$STRAND, \$STROR and \$STRXOR	18
Single line HTML/TEXT statement	18
Chapter 9: Compatibility/Bug fixes	19
SirFact	19
Sequence substitution on SIRFACT DUMP	19
Fixes in Sirius Mods 6.2 but not in 6.1	19
Fast/Reload	19
Version co-requisites	20

CHAPTER 1 ***Introduction***

This document lists the enhancements and other changes contained in the newest release of *Sirius Mods*: version 6.2. The previous generally released version of *Sirius Mods*, 6.1, was released in July, 2001.

2.1 Support for Model 204 Version 5.1

All products packaged in the *Sirius Mods* now support *Model 204* Version 5.1.

2.2 Commands now available as operator commands

The *SIRIUS*, *JANUS*, *SIRFACT* and *BUMPSNAP* commands are now available as operator commands, that is as responses to the *Model 204* HALT command from either a real or virtual console under OS/390 or as a virtual console command under CMS.

2.3 Comments before user 0 parameter line

In addition to the various commands which are supported before the user zero parameter line (such as MSGCTL), you may now place *Model 204* comments (that is, lines which begin with an asterisk and a space) before the user zero parameter line. Note that in this instance, the “full syntax” of *Model 204* comments (which sometimes allows an asterisk followed by a non-space character) is not allowed. For example, the following is a legal comment before the user zero parameter line:

```
* Suppress some messages for standard file reorg:
```

And the following is **not** a legal comment before the user zero parameter line:

```
*Suppress some messages for standard file reorg:
```

2.4 CASE Mixed/Upper command in Full Screen Editor

You may issue either of the following commands in the *Model 204* Full Screen Editor:

CASE Mixed This command indicates that terminal input is accepted “as is”, without translation to upper case. This is similar to invoking the *LOWER command prior to entering the Full Screen Editor. The second word of this command “Mixed”, may be abbreviated to any prefix.

CASE Upper This command indicates that terminal input is translated to upper case. This is similar to invoking the *UPPER command prior to entering the

Full Screen Editor. The second word of this command “Upper”, may be abbreviated to any prefix.

Note that:

- This command will not affect the case translation setting (*UPPER/*LOWER) that was in effect “outside” the editor; that is, when you exit the editor, the case translation will be as it was prior to entering the editor.
- In addition to the CASE command, the initial case translation setting within the editor can be automatically set to *LOWER, using the '1' bit of the SIREDDIT parameter.
- Neither the CASE command nor the *UPPER/*LOWER setting affect the input on the editor command line.
- The matching of strings (for example, “/subroutine”) can be done in a case-insensitive manner, using the '2' bit of the SIREDDIT parameter.
- This feature is available using the same rules as those controlling the ability to reset the SIREDDIT parameter. Note, however, that since both *Janus Web Server* and *Limited Janus Web Server* allow it, any customer with a currently authorized *Sirius Mods* product can use the editor CASE command.

CHAPTER 3 *Janus TCP/IP Base*

The following features are new or changed in *Janus TCP/IP Base*:

JANUS command The JANUS command can now be issued as an operator command, that is on the Online virtual console under VM or as a response to the HALT message under OS/390.

The following features are new or changed in *Janus Sockets*:

\$SOCK_URL_ENCODE Does URL encoding for an EBCDIC string so that special characters that can't be part of a URL or other parts of an HTTP request such as form fields are represented as their URL encoded equivalents which is simply a percent character followed by the hexadecimal representation of the ASCII character to which the EBCDIC character maps except for spaces which are encoded as pluses.

The following features are new or changed in *Janus Web Server*.

5.1 \$WEB_BROWSER function

Returns the release of specified browser or 0 if not that browser. Valid browsers are IE for Microsoft Internet Explorer™, NS for Netscape Navigator™ and OPERA for, you guessed it, Opera™. This function is useful for testing browser type and release — an

```
IF $WEB_BROWSER('IE') THEN
```

for example, is useful for testing if the browser is any release of Internet Explorer and a

```
IF $WEB_BROWSER('NS') GE 6 THEN
```

is useful for testing if the browser is a Netscape Navigator release 6 or later.

5.2 Janus Web Legacy Support Enhancements

Janus Web Server legacy support has been enhanced to use CSS. This greatly improves the appearance and alignment of fields on a 3270 screen mapped to HTML. In addition it is now possible to set the color of a field that's BRIGHT but without an explicit color and it's possible to specify PF key positions on a legacy screen as either TOP, LEFT, RIGHT or BOTTOM. *Janus Web Server* legacy support also uses the accesskey attribute for PF keys to allow `Alt-1` through `Alt-=` to be used as mouseless function keys with many browsers.

The *Janus Web Server* CSS for legacy support also allows users to specify their own style sheets for further customization of the appearance of legacy screens. Finally, a facility is provided to have *Janus Web Server* embed some JavaScript before and after a legacy screen which makes it possible to have legacy screens contained in pages with site specific graphics and text and also to specify images for the function keys. The *Janus Web Server* legacy support has also been enhanced to generate well-formed XHTML, that is all `<input>` tags are now terminated with a `/>` and all attributes on all tags are enclosed in double quotes. This makes it possible to treat a *Janus Web Server* legacy screen as XML for the purposes say of doing web-based 3270 screen-scraping.

5.3 WRAPJS parameter for ON rules

HTML does not allow a page to embed HTML from another URL but it does allow a page to embed JavaScript from another URL. This JavaScript can simply issue *document.write* calls to add HTML to the page at the point it was embedded and so can be used as the logical equivalent of an embedded HTML document. To eliminate the need for such embedded documents to contain all the *document.write* calls and their parentheses and quotes and to avoid the need to worry about "escaping" double quotes, the WRAPJS parameter is provided in JANUS WEB ON rules to indicate that *Janus Web Server* should automatically wrap each line of text in *document.write* calls and to precede every double quote with a slash character to prevent it from being treated as a close quote.

5.4 EXPIRE parameter for ON rules

The EXPIRE parameter in JANUS WEB ON rules now allows rules based setting of either absolute or relative expire times on pages as in

```
JANUS WEB WEBPORT ON /MUMBLE/* OPEN FILE MUMBLE -  
SEND * EXPIRE 021131  
JANUS WEB WEBPORT ON /FOO/* OPEN FILE FOO -  
SEND * EXPIRE +3600
```

CHAPTER 6 ***Janus SOAP***

Sirius Mods Version 6.2 introduces general availability of *Janus SOAP*. The XML API provided by *Janus SOAP* has been adapted to use the latest version of the XPath navigation language. In addition the storage model used by *Janus SOAP* has been simplified, in order to simplify debugging and make XMLDoc and nodelist instantiation more obvious. For more information, refer to the ***Janus SOAP Reference Manual***.

CHAPTER 7 *SirFact*

The following features are new or changed in *SirFact*:

7.1 SIRFACT command

The SIRFACT command can now be issued as an operator command, that is on the Online virtual console under VM or as a response to the HALT message under OS/390.

7.2 SIRFACT DUMP procedure substitution

A “+D” in a dump procedure name on a JANUS DUMP command is now replaced with the date in YYYYMMDD format. Note that this feature was actually added as ZAP6140 in *Sirius Mods* 6.1. A “+M” in a dump procedure name on a JANUS DUMP command is now replaced with the message number (minus the period) causing the request cancellation or the word “SNAP” if the *SirFact* dump was the result of a SIRFACT SNAP command. A plus character (“+”) followed by any digit from 1 to 9 will be replaced by a single digit sequence number (starting at 0) up to the specified number. That is, a “+4” will be replaced by 0, 1, 2, 3 or 4, whichever produces a procedure name that does not exist. Note that this means that the behavior for “+1”, “+2”, “+3” has changed from previous versions.

7.3 TRACE statement

A new User Language statement, TRACE, has been added that acts very much like a PRINT or AUDIT statement with the exception that the target for the TRACE statement can be dynamically changed by RESET'ing the ULTRACE parameter. The ULTRACE parameter is a standard bit-oriented user parameter where the bits mean:

- X'01'** Send TRACE output to terminal.
- X'02'** Send TRACE output to audit trail.
- X'04'** Send TRACE output to a CCATEMP wrap-around trace table.

The default setting for ULTRACE is X'01' which means that the TRACE statement will act pretty much like a PRINT statement. The bits can be combined so that output could be sent to both the terminal and the audit trail or to the CCATEMP wrap-around trace

table and to the terminal and to the audit trail. If a TRACE statement is executed when ULTRACE is set to 0, the request is cancelled.

The CCATEMP wrap-around trace table is a user-specific trace table that consists of a set of CCATEMP pages up to a maximum specified by the ULTRACEP user parameter. If ULTRACEP pages are already in use when a TRACE statement is issued and output is being routed to the trace table (ULTRACE X'04' set), the trace data on the oldest page is discarded and the oldest page is re-used for the new trace data. The default value for ULTRACEP is 2.

The wrap-around trace table is dumped in *SirFact* dumps and can be viewed when examining the dump with the command:

```
D[DISPLAY] T[RACE][.{* | N}]
```

where *N* specifies that the last *N* entries are to be displayed. A simple “D T” will display all trace entries and a “D T.20” will display the last 20.

ULTRACE and ULTRACEP can both be RESET via the \$RESETN function.

7.4 LONG REQUEST dumps

SirFact now produces dumps when a request is cancelled as the result of a “M204.1332” message when the user is not prompted with a “DO YOU REALLY WANT TO CONTINUE” message.

The \$functions presented in this chapter are new or have changed in version 6.2.

8.1 Character entity substitution enhancements

There is a new \$function called \$ENT which does character entity substitution, that is replaces characters that have special meaning to an HTML or XML processor with their character entity representation. For example, the “<” is represented as the “<” character entity. A default character entity substitution table is provided which does the basic required character entity mappings for HTML and XML of “&” to “&”, “<” to “<” and the double quote character to “"”. Another new \$function, \$ENT_TAB makes it possible to modify or examine the default or current character entity substitution table.

\$ENT_PRINT is provided to control automatic character entity substitution of PRINT'ed data including data PRINT'ed via an HTML statement. \$ENT_PRINT can set character entity substitution to “OFF” (the default), “ON” which causes all PRINT'ed data to undergo character entity substitution or “VAR” which causes only non-constant, that is non-literal and non-static variable data to undergo character entity substitution.

The HTML statement has also been enhanced to allow an ENT_PRINT ON, OFF or VAR parameter to control the effect of character entity substitution just over the scope of the HTML statement. Finally, three special characters have been defined that when they immediately follow the expression start character(s) in an HTML block they cause special processing for the printing of the expression results:

- &** Perform character entity translation regardless of the current \$ENT_PRINT setting and the ENT_PRINT setting on the HTML statement.
- !** Don't perform character entity translation regardless of the current \$ENT_PRINT setting and the ENT_PRINT setting on the HTML statement.
- *** Don't print the results of expression.

So for example in

```
HTML
  <input type="text" name="foo" value="{&%FOO}">
END HTML
```

%FOO would undergo character entity translation before being printed.

8.2 \$FIELD_IMAGE and \$FIELD_LISTI

Allow retrieval of multiple fields from a record in a single call using an image as a template for the fields to be retrieved. \$FIELD_LISTI allows multiple occurrences of a repeating group to be retrieved to a \$list, again using an image as a template for the fields to be retrieved.

8.3 \$list size restrictions eased

The maximum size of a \$list item has been increased from 4096 bytes to 6124. This maximum \$list item size can now be retrieved with the \$LIST_MAXIL function.

In addition, \$lists have been enhanced to allow \$lists to grow to a three-level tree structure rather than their former two-level limit. This means that the quantity of data that can be held in a \$list has been increased from the former limit of about 4.6 megabytes of data to about 3.5 gigabytes (billions of bytes) of data.

8.4 \$LISTNEWI and \$LISTNEWAI

Combine the functionality of \$LISTNEW and \$LISTIMG or \$LISTNEWA and \$LISTIMG into a single call. These functions make it possible to create a new \$list or array of \$lists and associate these \$lists with an image in a single call.

8.5 \$LIST_PRINT

Displays the contents of a \$list to the current output target, probably the terminal. This function is useful in displaying \$list contents when debugging code but might prove useful in contexts where data is stashed on a \$list and then needs to be sent to a USE dataset or a web browser.

8.6 LONGSTRING datatype

It is now possible to declare a %variable as a type LONGSTRING. LONGSTRING's largely act as regular STRING's with a few exceptions;

- They can be longer than 255 bytes, in fact can be as long as $2^{31}-1$ bytes long. The first 255 bytes of a LONGSTRING always resides in the server and any more resides in CCATEMP.

- A WITH operation involving a LONGSTRING either as an operand or the target will not truncate at 255 bytes but will produce a LONGSTRING result up to $2^{31}-1$ bytes long.
- They will not be automatically truncated on assignment to a regular STRING variable the way *Model 204* normally silently truncates values on assignment. An assignment from a LONGSTRING to STRING variable that requires truncation will cause a request cancellation. Similarly, the use of a LONGSTRING that is longer than 255 bytes as input to a \$function or subroutine that takes regular STRING's as input will also result in request cancellation.
- They can't be passed as parameters to subroutines that have the parameter defined as STRING OUTPUT and STRING's cannot be passed as parameters to subroutines that have the parameter defined as LONGSTRING OUTPUT.

There are several new \$functions that are specifically geared toward manipulating LONGSTRING's including

\$LSTR_LEN	Returns length of a LONGSTRING.
\$LSTR_SUBSTR	Returns a sub-string of a LONGSTRING.
\$LSTR_LEFT	Returns the left-most N characters of a LONGSTRING padding on the right if the requested length is greater than the length of the input LONGSTRING.
\$LSTR_RIGHT	Returns the right-most N characters of a LONGSTRING padding on the left if the requested length is greater than the length of the input LONGSTRING.
\$LSTR_INDEX	Returns the position of a STRING inside a LONGSTRING or 0 if the STRING can't be found in the LONGSTRING.
\$LSTR_UNBLANK	Removes leading, trailing and extra intermediate blanks from a LONGSTRING.

LONGSTRING's can be defined as arrays, common variables and as parameters in subroutine definitions. A few other \$functions have been added to facilitate movement of data to/from LONGSTRING's:

\$LISTINF_LSTR	Returns a \$list item to a LONGSTRING.
\$LISTADD_LSTR	Adds a LONGSTRING to a \$list.
\$LISTINS_LSTR	Inserts a LONGSTRING into a \$list.
WEB_GET_COOKIE_LSTR	Returns the value of an HTTP cookie to a LONGSTRING.

WEB_SET_COOKIE_LSTR Sets an HTTP cookie from a LONGSTRING.

LONGSTRING's are dumped in *SirFact* dumps and can be displayed at dump examination time exactly as one displays the value of any other %variable, namely with the "D %variable" command.

8.7 **SEQ parameter for \$SIRJGET**

An option that requests that sequence numbers be placed in front of the journal entries formatted by \$SIRJGET. This is useful for maintaining context and position when subsetting the formatted journal lines. *SirScan* has been enhanced to take advantage of this facility to improve subsetting and positioning associated with the ALL command.

8.8 **\$STRAND, \$STROR and \$STRXOR**

String oriented logical bit manipulation functions. Allow AND'ing, OR'ing and exclusive OR'ing the bits in two strings. \$STRXOR, in particular, can be useful as a low-grade two-way encryption technology since two-way encryption often simply involves the exclusive OR'ing of a string with an encryption mask.

8.9 **Single line HTML/TEXT statement**

The HTML/TEXT statement now has a DATA option that must be the last option on the HTML/TEXT statement and indicates that all text that follows the word DATA is to be treated as if it were inside an HTML block and that the HTML block is considered to end at the end of the logical line. This provides a single line HTML or TEXT statement capability that can be very useful as a PRINT statement alternative as in

```
FOR %I FROM 1 TO 10
  FOR %J FROM 1 TO 10
    TEXT DATA {%I} + {%J} = {%I + %J}
  END FOR
  TEXT DATA Isn't this prettier than PRINT?
END FOR
```

Compatibility/Bug fixes

This chapter lists any compatibility issues with prior versions of the *Sirius Mods* and any bugs which have been fixed in this version of the *Sirius Mods* but had not, as of the date of this release, been fixed in the immediately prior version (6.1).

The first sections (that is, all sections before “Fixes in *Sirius Mods* 6.2 but not in 6.1”) list, by product, any backwards incompatibility issues, that is, any differences in processing that result from successful execution with *Sirius Mods* version 6.1, as compared with the same inputs to *Sirius Mods* version 6.2.

9.1 SirFact

9.1.1 Sequence substitution on SIRFACT DUMP

Substitution strings “+1”, “+2” and “+3” in the procedure name on a SIRFACT DUMP command used to substitute sequence numbers 0-9, 00-99 and 000-999 respectively. Under *Sirius Mods* 6.2 they now substitute sequence numbers 0-1, 0-2 and 0-3 respectively. In addition the digits 4 through 9 are now also available as sequence substitution characters with their obvious interpretations.

9.2 Fixes in *Sirius Mods* 6.2 but not in 6.1

This section lists other fixes to functionality existing in the *Sirius Mods* version 6.1 but which, in absence of customer problems, have not, as of the date of the release, been fixed in that version.

9.2.1 Fast/Reload

D statement and max record parameter

The D statement now handles the first parameter of the FLOD or FILELOAD command; previously it was ignored on the D statement.

The first FLOD parameter indicates the maximum number of *Model 204* records to load.

Previously, for example, you could not use this parameter, together with the PAI FLOD program documented in the *Model 204 File Manager's Guide*, to limit the size of file being loaded.

9.3 Version co-requisites

This section lists any restrictions on usage of various products (including *Sirius Mods* itself) which will be imposed by use of version 6.2 of *Sirius Mods*.

There are no corequisites associated with *Sirius Mods* 6.2.