

# SASRX user documentation

## Introduction

SASRX is a REXX program that you can use to invoke SAS. It is provided as an alternative to the SAS CLIST. SASRX supports the same command-line syntax as the SAS CLIST. It also supports the following additional features:

- mixed case option values
- additional options
- Unix-style option specification
- direct specification of SAS system options
- USS file and directory names as option values.

## Option syntax

You can specify 0 or more options on the SASRX command line. Separate the option specifications by one or more blank spaces.

Two different categories of options are processed by SASRX, SASRX options and SAS system options. SASRX options are defined internally to SASRX, which recognizes these options and acts on them. See the individual tables under “SASRX Options” beginning on page 4 for information about these options. All other options that SASRX does not recognize are assumed to be SAS system options. SASRX does not validate or act on these options but only passes them to SAS.

Options of either category are further classified as either *switch* options or *value* options. Switch options are specified only by a keyword; they do not take any value. Value options are specified as a keyword followed by a value. Refer to the decision tree on page 2 for details on how SASRX identifies an option as either a switch option or a value option, and how this differs between SASRX options and SAS system options.

SASRX also supports two different styles of option specifications, *CLIST-style* and *Unix-style*. CLIST-style option specifications are set off from adjacent option specifications only by blank spaces. If the option is a value option, the keyword is followed by optional blank spaces and a value specification that is in parentheses.

Unix-style option specifications are also set off by blank spaces, but the keyword is also prefixed with a hyphen (-). If the option is a value option, the value follows after the next blank space. The value is not enclosed in parentheses.

There are certain unusually-named options that cannot be directly specified as SAS system options with a Unix-style specification. For example `sasrx -fileblksize(3340) 8368` is not accepted. You can however specify this option as `sasrx -options "fileblksize(3340)=8368"`.

CLIST-style and Unix-style option specifications can be freely intermingled on the command line, with a few special cautions that are noted below. Because the two styles can be intermingled, and because SAS system options are accepted, SASRX does not support the Unix convention that an argument that does not begin with “-” is an input file name. It is necessary to specify `-input` explicitly to identify a SAS program input file even if the Unix-style option specification is used exclusively.

SASRX option names can be truncated to the minimum unique abbreviation. For example O is the minimum abbreviation for the OPTIONS option because no other SASRX option begins with O. SASU is the minimum abbreviation for the SASUSER option because other options begin with the string SAS. SAS system options generally must not be abbreviated.

## Examples of option types and specification styles

1. **sasrx o(dms)** is functionally equivalent to **sasrx dms**. In the first case, **o** is a SASRX option known to be a value option, and **dms** is its value. In the second case, **dms** is a SAS system option that is assumed to be a switch option.
2. **sasrx -linesize 72**  
**sasrx linesize(72)**  
**sasrx o(linesize=72)**  
 are all equivalent. The first example is a Unix-style specification of the SAS system value option **linesize** with a value of **72**. The second example is a CLIST-style specification of the same option. The third example is a CLIST-style specification of the SASRX value option **OPTION** with a value of **linesize=72**. The first two examples illustrate new functionality in SASRX, while the third example illustrates compatibility with the SAS CLIST.

**sasrx -linesize(72)** is invalid because it mixes Unix-style (hyphen prefix) and CLIST-style (value in parentheses) within an option specification.

These examples illustrate that when a SAS system value option is specified directly on the SASRX command line in Unix-style, its form must follow the Unix convention of blank space separating the option keyword from the value. When a SAS system value option is specified indirectly via the **OPTION** option, its form must follow the SAS system convention of joining the keyword and value with an equals sign. Thus, for example, **sasrx -linesize=72** would be invalid (Unix-style but value not separated with a blank), as would **sasrx o(linesize 72)** (indirect option specification through **OPTIONS** option but value not separated with an equals sign).

## Option classification

There is some imprecision in the way that SASRX classifies a SAS system option as a switch option or a value option. Note that you can avoid any ambiguity by using either exclusively Unix-style or exclusively CLIST-style option specifications. But if you mix both styles in the same command line, you need to understand the following decision tree that SASRX uses to classify an option specification.

Is the keyword a SASRX option?

- Yes. Its definition specifies whether it is a value option or switch option.
- No. Is the keyword specified in Unix-style? (Is it prefixed with a hyphen?)
  - Yes. Is the keyword either at the end of the command line, or followed by another token that is also prefixed with a hyphen?
    - Yes. The keyword is not followed by a value so it is a switch option.
    - No. Does the next token begin with a single or double quote?
      - Yes. The current token is a value option with a quoted value
      - No. Does the next token contain an embedded left parenthesis?
        - Yes. The next token is a CLIST-style value option, so the current token is a switch option.
        - No. The current token is a value option; the next token is its value. This is where the possibility of misclassification occurs; see examples below.
- No. Is the keyword followed by a string that is in parentheses?

- o Yes. It is a value option.
- o No. It is a switch option.

### Examples of option classification

1. **sasrx dms -memrpt** is valid because SASRX recognizes **-memrpt** as a Unix-style option specification and, therefore, classifies **dms** as a SAS system switch option. However **sasrx -dms memrpt** is not valid because, although **-dms** is classified as a SAS system option, **memrpt** appears to be its value rather than another option. The command is parsed like **sasrx dms(memrpt)**. **sasrx -nostae memrpt** is parsed correctly because **nostae** is a SASRX switch option, thus it is recognized immediately.
2. **sasrx -dms sortpgm(best)** is parsed correctly because the "(" flags the token following **-dms** as being another option, not a value for the DMS option.

### Quoting

For backward compatibility with the SAS CLIST, SASRX requires CLIST-style quoting for CLIST-style option specifications. For Unix-style option specifications, SASRX requires the minimum number of quotes.

Any option value can be enclosed in quotes, but the quotes are required only in certain circumstances.

- For Unix-style option specifications, option values that contain internal blanks, single quotes ('), or other special characters are required to be enclosed in an outer pair of quotes. If the value contains internal single quotes, the outer quotes must be double ("); otherwise they can be either single or double.
- For CLIST-style option specifications the value is required to be enclosed in an outer pair of quotes only if it contains internal quotes. These outer quotes must be single quotes (').

### Examples of quoting

1. In many cases quotes are optional. **sasrx -work "500,200"** and **sasrx -work 500,200** are equivalent.
2. **sasrx -input "'myid.my.sas'"** is equivalent to **sasrx input(''myid.my.sas''')**  
but  

<b>sasrx -input '''myid.my.sas'''</b>	is invalid (not minimum quotes)
<b>sasrx input('myid.my.sas')</b>	is accepted but implies omitted prefix
<b>sasrx -input 'myid.my.sas'</b>	also is accepted but implies omitted prefix
3. **sasrx -log "'~/%Y%m%d.log'" -logparm "rollover=session"**  
extra quotes because of % character
4. **sasrx O('NEWS="SAS.NEWS(ENW0)" nodms')**  
**sasrx O(NEWS="SAS.NEWS(ENW0)" nodms)**  
**sasrx O(NEWS='SAS.NEWS(ENW0)'** **nodms)**  
**sasrx O('NEWS='''SAS.NEWS(ENW0)''' nodms')**  
**sasrx -options "NEWS='SAS.NEWS(ENW0)'** **nodms"**  
All of the above are equivalent.
5. **sasrx o('sysparm="Don''''t use too many quotes"')**  
**sasrx -sysparm "Don't use too many quotes"**  
**sasrx -options "sysparm='Don't use too many quotes'"**  
All of the above are equivalent. Note that in the rare case where a third level of quoting is

required in a Unix-style specification (last example above), the third level of quotes must be doubled.

- When using the explicit mode of invoking the rexx exec, the entire parameter string must be enclosed in single quotes, requiring any internal single quotes to be doubled:

```
exec rexx.exec(sasrx) '-input "'myid.my.sas'"' exec
```

### Option priority

The order in which options are passed to SAS is not necessarily the same as the order in which they are specified on the SASRX command line. When an option is specified more than once, the effective specification is the final one as passed to SAS. The options string passed to SAS contains in this order:

- Option values generated by SASRX
- The list of directly specified SAS system options in the order in which they were specified
- The value of the OPTIONS option
- The value of the SRVOPTS option

### Example of option ordering

```
sasrx srvopts(nomemrpt) o(nodms) dms input(my.sas)
```

yields the following options parameter that is passed to SAS:

```
O('SYSIN=SYSIN DMS nodms nomemrpt')
```

where **SYSIN=SYSIN** is generated by processing of the INPUT option, **DMS** is a directly specified SAS system option, **nodms** is the value of the OPTIONS option, and **nomemrpt** is the value of the SRVOPTS option.

### Site customizations

A site can customize the SASRX exec to meet local requirements. SASRX is coded in such a way that it should be possible to make most or all customizations in the definitions of options, not in the REXX code. All SASRX options are defined in a comment block at the beginning of the exec, using CLIST-style specification. Default option values can be changed by specifying the desired default here.

### SASRX options

#### *Dataset options*

The set of SASRX options in the table below are used to specify datasets that are to be allocated for SAS. For each option, the name of an MVS dataset is a valid value. Options pertaining to input files accept multiple datasets to be concatenated. For some of these options, a size specification is also valid. For example, the default value for the WORK option is '500,200' which means to allocate a temporary WORK file with the ALLOC option SP(500,200). Options with "file" or "dir" indicated in the USS column also accept a USS file or directory as a value, for example **-work /tmp**. A value is recognized as being a USS file or directory name only if it includes at least one "/".

Dataset option name	Default value	Description	USS
AUTOEXEC()		AUTOEXEC= dataset name	file
CLOG()	*	If the value is numeric, it specifies the value for the SPACE operand for the SASCLOG file allocation. If the value is a single letter, it specifies the SYSOUT class for SASCLOG. If SASRX is running in the background and the CLOG	

Dataset option name	Default value	Description	USS
		value is *, then SASCLOG is allocated to DUMMY. In all other cases, the value specifies the DSN for the SASCLOG file.	
CONFIG()		User configuration file to concatenate with system configuration files	
DBMSLIBS()		Database library to concatenate with SASLOAD	
HOTFIX()		Hot Fix library to concatenate with SASLOAD	
INPUT()		SYSIN= dataset name	file
LOAD()		User load library to concatenate with SASLOAD	
LOG()	*	If the value is numeric, it specifies the value for the SPACE operand for the LOG output file allocation. If the value is a single letter, it specifies the SYSOUT class for LOG output. If SASRX is running in the background and the LOG value is *, then the LOG output file is allocated to DUMMY. In all other cases, the value specifies the dataset name or USS file name for the LOG output file	file
MAUTS()	Set by installation procedure	System macro autocall library	
MTKMVS()	Set by installation procedure	System TKMVSENV dataset	
PRINT()	*	If the value is numeric, it specifies the value for the SPACE operand for the PRINT output file allocation. If the value is a single letter, it specifies the SYSOUT class for PRINT output. If SASRX is running in the background and the PRINT value is *, then the PRINT output file is allocated to DUMMY. In all other cases, the value specifies the dataset name or USS file name for the PRINT output file	file
<a href="#">SAMPPIO()</a>	<a href="#">Set by installation procedure</a>	<a href="#">Sample SAS data library</a>	
SASAUTOS()		User macro autocall library	
SASHELP()	Set by installation procedure	SASHELP library dataset name	
SASLOAD()	Set by installation procedure	SAS load library dataset name	
SASMSG()	Set by installation procedure	SAS message library dataset name	
SASUSER()	&syspref.SAS9.SASUSER	SASUSER library dataset name	dir
SORTLDSN()	SYS1.SORT.LINKLIB	system sort library dataset name	
<a href="#">SYSTEMCONFIG()</a>	<a href="#">Set by installation procedure</a>	<a href="#">System config file or files.</a>	
<a href="#">SYSTCPD</a>		<a href="#">SYSTCPD dataset</a>	
TKMVSENV()		User TKMVSENV dataset	
TRANSHelp()		Name of translated SASHELP library dataset to be concatenated in front of the English SASHELP.	

<b>Dataset option name</b>	<b>Default value</b>	<b>Description</b>	<b>USS</b>
WORK()	'500,200'	WORK library dataset name, or size if numeric.	dir

### ***Alternate DDname options***

Options in this group allow you to use DDnames other than defaults. See the discussion under Dataset-related options and alternate DDnames for more information.

<b>Alternate ddname option</b>	<b>Default value</b>	<b>Description</b>
DDAUTOEX()	SASEXEC	AUTOEXEC= file DDname
DDCONFIG()	CONFIG	CONFIG= file DDname
DDLOG()	SASLOG	LOG= file DDname
DDPARMCD()	SASPARM	PARMCARDS= file DDname
DDPRINT()	SASLIST	PRINT= file DDname
DDSASAUT()	SASAUTOS	SASAUTOS= library DDname
DDSASHLP()	SASHELP	SASHELP= data lib DDname
DDSASMSG()	SASMSG	SASMSG= library DDname
DDSASUSR()	SASUSER	SASUSER= data lib DDname
DDSYSIN()	SYSIN	SYSIN= file DDname
DDWORK()	WORK	WORK= data lib DDname

### ***Miscellaneous value options***

<b>Miscellaneous value options</b>	<b>Default value</b>	<b>Description</b>
BLOCK()	264	PRINT BLOCKSIZE
DBMSCONCAT()	LAST	DBMSLIB concatenation order: FIRST or LAST
ENTRY()	SAS, SASB, or SASLPA	SAS Entry point name
LRECL()	260	PRINT LRECL
OPTIONS()		SAS system options
PARMCARD()	1	parmcard file size
SORTLINK()	*	Put system sort lib inTASKLIB? (* means No, () means Yes
SRVOPTS()		SAS system options for server
UNITS()	CYL	alloc unit for LOG, CLOG, PRINT, PARMCARD

### ***Environment variable options***

These SASRX options set the values of environment variables that are used by SAS. NETENCALG is a special case because SAS can use it as an environment variable and as a SAS system option. When specified to SASRX, both the environment variable and the SAS system option are set.

<b>Environment variables</b>	<b>Default value</b>	<b>Description</b>
inherit()		
netencalg()		
sasclientport()		
sasdaemonport()		
tcpdebug()		

## Switch options

For the switch options that are on by default, a NO-prefixed alternative is provided. For example, **STAE** is on by default, but **NOSTAE** overrides the default.

The switch options **STAE**, **NOSTAE**, **STAX**, and **NOSTAX** exist as both SASRX options and SAS system options, with different meanings. Generally these options are used only at the direction of SAS Technical Support. The SAS system option is more frequently used, so these options should usually be specified via the **OPTIONS** option, e.g. **-options nostae**.

Switch options	Description
FLUSH NOFLUSH	Flush stack if error
GO	Continue previous session. When GO is specified, SASRX sets the NOWORKINIT SAS system option and skips processing of either specified or default values for the CONFIG, WORK, SASMSG, PRINT, SASHELP (if not concatenated), PARMCARD, SASUSER, SASAUTOS, and TKMVSENV SASRX options. That is, no allocations are done for these files; it is assumed that the datasets controlled by these options have already been allocated by a previous SAS run.
NOSASUSER	Do not allocate SASUSER dataset
SHARE NOSHARE	Share subpool 78
STACK NOSTACK	Create new i/p stack
STAE NOSTAE	Trap main task abends
STAI NOSTAI	Trap subtask abends
STAX NOSTAX	Trap attentions
SYSDUMP	Allocate SYSDUMP
TRACE	REXX tracing

## Dataset-related options and alternate DDnames

The table below lists dataset-related options for which there is both a SASRX option and a SAS system option with the same name (the INPUT/SYSIN option pair is included because of its similar behavior, although the options have different names). In each case the SASRX option takes the name of a dataset or USS file or directory as a value, or in some cases a size value as described above under "Dataset Options". The corresponding SAS system option can take the same values, or alternatively a DDname. (The CONFIG SAS system option is slightly different because it only takes a DDname.) In the SASRX implementation of these options, if the value is a dataset, USS file or directory, or size, SASRX allocates a file, and then generates the DDname form of the SAS system option. There are three exceptions to this: For the INPUT, LOG, and AUTOEXEC options, if the value is a USS file, SASRX passes the option unchanged to SAS and does not issue an ALLOC command.

SASRX option name	SAS system option set by SASRX	Default SAS system option value	SASRX option for alternate ddname
AUTOEXEC	AUTOEXEC=	SASEXEC	DDAUTOEX
CONFIG	CONFIG=	CONFIG	DDCONFIG
INPUT	SYSIN=	SYSIN	DDSYSIN
LOG	LOG=	SASLOG	DDLOG
PRINT	PRINT=	SASLIST	DDPRINT
SASAUTOS	SASAUTOS=	SASAUTOS	DDSASAUT

SASRX option name	SAS system option set by SASRX	Default SAS system option value	SASRX option for alternate ddname
SASHELP	SASHELP=	SASHELP	DDSASHLP
SASMSG	SASMSG=	SASMSG	DDSASMSG
SASUSER	SASUSER=	SASUSER	DDSASUSR
WORK	WORK=	WORK	DDWORK

### Examples of file allocation

If you specify **AUTOEXEC(MY.SAS(AUTOX))** SASRX will issue the command **ALLOC FILE(SASEXEC) DA(MY.SAS(AUTOX)) SHR REU** and SAS will use the default option **AUTOEXEC=SASEXEC**.

If you need to use a DDname other than SASEXEC, you can also specify the **DDAUTOEX** SASRX option. For example if you specify **AUTOEXEC(MY.SAS(AUTOX)) DDAUTOEX(MYAUTX)** SASRX will issue the command

**ALLOC FILE(MYAUTX) DA(MY.SAS(AUTOX)) SHR REU**  
and pass the option **AUTOEXEC=MYAUTX** to SAS.

If your autoexec file is a USS file named sasrx/autox.sas, you can specify the SASRX option **-autoexec sasrx/autox.sas**. SASRX will pass the SAS system option **autoexec=sasrx/autox.sas** to SAS, it will not issue an ALLOC command.

The other options in the table work in a corresponding manner.