Ad

Adams 2021.3

Configuring Adams

HEXAGON

Corporate
MSC Software Corporation
5161 California Ave, Suite 200
University Research Park
Irvine, CA 92617
Telephone: (714) 540-8900
Email: americas.contact@mscsoftware.com

Europe, Middle East, Africa
MSC Software GmbH
Am Moosfeld 13
81829 Munich, Germany
Telephone: (49) 89 431 98 70
Email: europe@mscsoftware.com

Japan
MSC Software Japan Ltd.
KANDA SQUARE 16F
2-2-1 Kanda Nishikicho, Chiyoda-ku
Tokyo 101-0054, Japan
Telephone: (81)(3) 6275 0870
Email: MSCJ.Market@mscsoftware.com

Asia-Pacific
MSC Software (S) Pte. Ltd.
100 Beach Road
#16-05 Shaw Tower
Singapore 189702
Telephone: 65-6272-0082
Email: APAC.Contact@mscsoftware.com

Worldwide Web
www.mscsoftware.com

Support
http://www.mscsoftware.com/Contents/Services/Technical-Support/Contact-Technical-Support.aspx

ADAM:V2021.3:Z:Z:Z:DC-CONFIG

Documentation Feedback

At MSC Software, we strive to produce the highest quality documentation and welcome your feedback. If you have comments or suggestions about our documentation, write to us at: documentation-feedback@mscsoftware.com.

Please include the following information with your feedback:

- Document name
- Release/Version number
- Chapter/Section name
- Topic title (for Online Help)
- Brief description of the content (for example, incomplete/incorrect information, grammatical errors, information that requires clarification or more details and so on).
- Your suggestions for correcting/improving documentation

You may also provide your feedback about MSC Software documentation by taking a short 5-minute survey at: http://msc-documentation.questionpro.com.

Note: The above mentioned e-mail address is only for providing documentation specific feedback. If you have any technical problems, issues, or queries, please contact Technical Support.

Learning the Basics

Running and Configuring Adams

This section of the online help provides you with instructions for running and setting up the Adams® suite of products. It describes:

- Running Adams products.
- Creating user dynamic-link libraries that extend the functionality of Adams and run them with the associated Adams products.
- Configuring Adams products so they work the way you want them to.

Installing Adams

The install guides for Linux and Windows can be accessed through MSC Software Documentation Center https://help.mscsoftware.com/.

Adams Interfaces

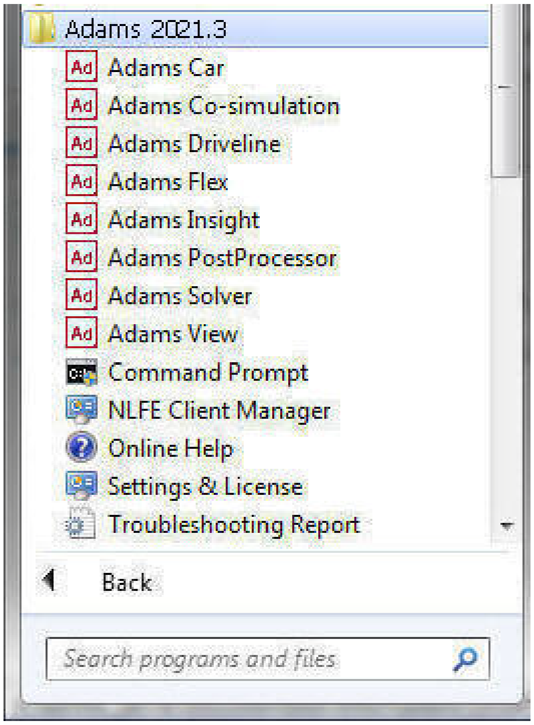
There are four types of interfaces that you can use to access Adams products and create libraries of custom operations for the products:

- Adams Toolbar on Linux only- The Adams Toolbar is your starting point to using Adams products. The Toolbar gives you access to the major products you installed. It uses a registry service that maintains values and settings that you need when running Adams.
- Adams Program Folder on Windows only - The Adams program folder is your main starting point for running and customizing Adams products.
- Adams Selection Menu - The Selection Menu is a menu- and text-based interface that allows you to enter information on the command line. You can enter all of the commands on one line to quickly start the product. Note that you cannot access all products nor change all settings from the Selection Menu.
- Command-Line Parameters - You can enter command-line parameters at the command prompt to run Adams.

Although you can use the Adams Selection Menu, on Linux we recommend that you use the Adams Toolbar because it provides an easy-to-use interface and lets you run all products. Because of its advantages, this online help focuses on how to run and configure Adams products using the Adams Toolbar. For information on how to use the Adams Selection Menu, see Selection Menu.

Adams Program Folder on Windows

The Adams program folder on the Start menu is your starting point for running and customizing Adams products. It provides you with access to all the Adams products which you installed. Note that if you are not licensed to run a product, it will not run even though it is in the program folder.



Each of the Adams products is listed in the Adams program folder. You may have fewer items than shown in figure above, depending on which products you installed. The program folder also contains options for creating and running user libraries, for setting Application defaults, and for generating problem reports. The following table groups the options by their function. For example, it groups all options that run products.

The option:	Does the following:
Product group: Adams Car Adams Co-simulation Adams Driveline Adams Flex Adams Insight Adams PostProcessor Adams Solver Adams View	Runs Adams products. The run options are described in Running Adams Products on Windows and Running Adams Products on Linux.
Command Prompt	Opens the Adams Selection Window in a Windows command shell. ■ Creates user libraries of user-written subroutines to customize Adams products. For more information, see Creating User Libraries.
Online Help	Starts the Adams online help.
Settings & License	Sets up passwords and a license server to manage access to Adams products. See the Adams Installation and Operations Guide for more information. Allows you to control the look of your screen through the graphics drivers and select the amount of memory you need for your model. For more information, see Setting Preferences for Adams on Windows or on Linux.
Troubleshooting Report	Creates a troubleshooting report that examines your system configuration and places important debugging information in the file adams_problem_report.txt. For more information, see Troubleshooting.

Adams Toolbar on Linux

The Adams Toolbar provides access to many of the Adams major products, which all have tools on the Toolbar as shown in the figure below. Adams also has add-on modules or plugins, which expand the functionality of the major products, such as Adams Controls, Adams Durability, and Adams Vibration. You run these products from within the major products. For example, to run Adams Vibration, you first run Adams Car or Adams View and then select the command to run Adams Vibration.

- Displaying the Toolbar
- Setting Up the Toolbar
- Exiting the Adams Toolbar
- Viewing Licensing Information

Displaying the Toolbar

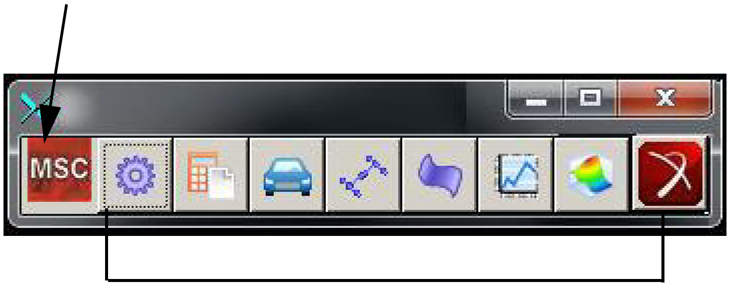
Your system administrator typically sets the command required to run the Adams Toolbar. Therefore, before you begin, you need to know the command to display the Adams Toolbar. It is also helpful to know where the software is installed so you can troubleshoot and access examples, demonstrations, and templates that come with Adams products. See your system administrator for assistance.

To start the Adams Toolbar:

- Enter the command to display the Adams Toolbar (the standard command that MSC Software provides is **adamsx** where ***x*** is the version number, for example adams2021_3).

  The Adams Toolbar appears as shown next:

Adams Toolbar tool - Right click to set up Toolbar, manage memory models, access online help and Technical Support resources and more



Product tools - Click to run product or right-click to configure products and create user libraries.

Hold the cursor over a tool to see the name of the associated product.

Setting Up the Toolbar

You can change the orientation of the Adams Toolbar and remove tools representing products that you do not use frequently. Deleting a tool does not remove it permanently; you can still restore it.

To change the orientation of the Toolbar:

- Right-click the Adams Toolbar tool, and select Horizontal or Vertical depending on the current orientation of the Toolbar.

To remove a product tool from the Toolbar:

- Right-click the tool you want to remove, and then select Remove from Toolbar. For example, to remove the Adams Insight tool, right-click the tool, and then select Remove Adams Insight from Toolbar.

Note: If you would like to remove a product tool from the Toolbar, you must first be sure that the tool is not set to run the product with a user library. If it is, you can only select to remove the library. To check what a product tool is set to run, see Adams View Preferences.

To restore all tools to the Toolbar:

- Right-click the Adams Toolbar tool, and then select Reset Icons.

Exiting the Adams Toolbar

To exit the Adams Toolbar:

- Right-click the Adams Toolbar tool, and then select **Exit**.

Viewing Licensing Information

Using the Adams Toolbar, you can generate a listing of the software for which you are licensed. The listing includes the major version of the products you are running, the number of licenses you have purchased, and the date when the licenses expire.

To view licensing information:

1. Right-click the Adams Toolbar tool, point to **Support**, and then select **License Information.**

   The Adams License Information dialog box appears.

2. After viewing the information, select **Close**.

Selection Menu

The Adams Selection Menu is a command-line interface for building and running Adams products. It provides you with access to all the Adams products for which you are licensed.

- Displaying the Selection Menu
- Entering Selection Codes
- Getting Help
- Exiting the Selection Menu

Displaying the Selection Menu

Your system administrator can change the command to run the Adams Selection Menu to access Adams products. Before you begin, therefore, you need to know:

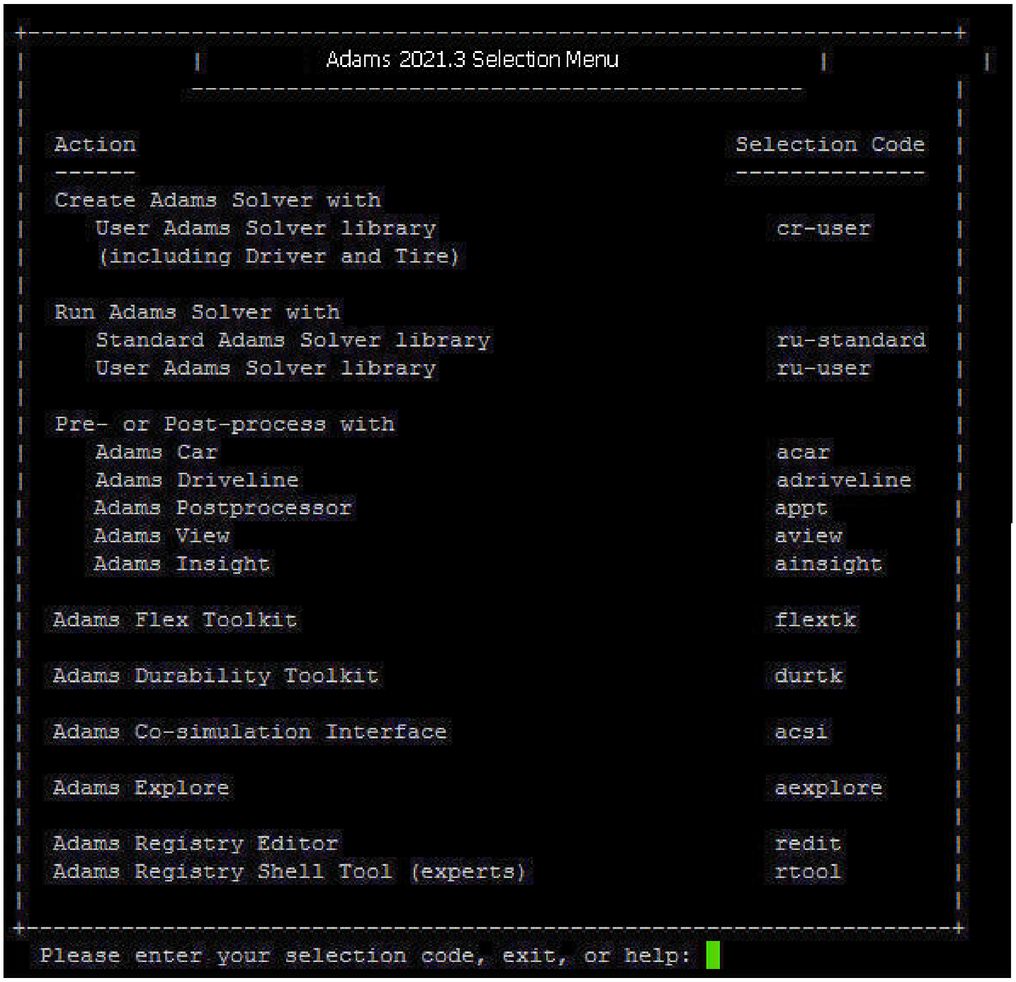
- The command to run the software.
- Where the software is installed.

To start the Selection Menu:

- **On Windows**:
    - From the **Start** menu, point to **Programs**, point to **Adams *x*** (where *x* is the release number), and then select **Command Prompt**.
    - Enter the command to run the Adams Selection menu.
- **On Linux:** Enter the command to run the Adams Selection Menu.

  The standard command that MSC Software provides is **adamsx -c** where *x* is the version number, for example `adams2021_3`.

  The Adams Selection Menu appears. It contains a list of options, or selection codes, for the products for which you are licensed. A Windows example is shown next.



```
+-------------------------------------------------------------------------+
|                 |          Adams 2021.3 Selection Menu          |        |
|                 -------------------------------------------------        |
|                                                                          |
|  Action                                              Selection Code     |
|  ------                                              --------------     |
|  Create Adams Solver with                                               |
|     User Adams Solver library                            cr-user        |
|     (including Driver and Tire)                                         |
|                                                                          |
|  Run Adams Solver with                                                  |
|     Standard Adams Solver library                        ru-standard    |
|     User Adams Solver library                            ru-user        |
|                                                                          |
|  Pre- or Post-process with                                              |
|     Adams Car                                            acar           |
|     Adams Driveline                                      adriveline     |
|     Adams Postprocessor                                  appt           |
|     Adams View                                           aview          |
|     Adams Insight                                        ainsight       |
|                                                                          |
|  Adams Flex Toolkit                                     flextk          |
|                                                                          |
|  Adams Durability Toolkit                               durtk           |
|                                                                          |
|  Adams Co-simulation Interface                          acsi            |
|                                                                          |
|  Adams Explore                                          aexplore        |
|                                                                          |
|  Adams Registry Editor                                  redit           |
|  Adams Registry Shell Tool (experts)                    rtool           |
|                                                                          |
+-------------------------------------------------------------------------+
  Please enter your selection code, exit, or help:
```

- The following table explains all the possible selection codes available from the Adams Selection Menu:

The code:	Does the following:
cr-user	Leads you through the creation of an Adams analysis product user library. Creates a library for the Adams analysis products: Adams Solver and Adams Tire.
ru-standard	Runs standard Adams Solver.
ru-user	Runs Adams Solver with a user library. A user library is one or more user-written subroutines.
acar, adriveline, appt, aview, ainsight	Runs Adams products with and without user libraries, and for certain products, leads you through the creation of the user libraries.
flextk	Runs the Adams Flex toolkit, which lets you view and work with a modal neutral file (MNF), which describes a flexible body. For more information on the Adams Flex toolkit and MNF files, refer to the Adams Flex online help.
durtk	Runs Adams Durability toolkit. For more information see Adams Durability online help.
acsi	Runs Adams Co-Simulation Interface. For more information, see Adams Co-Simulation Interface (ACSI).
aexplore	Runs Adams Explore. For more information, see Adams Explore.
redit	Runs the MSC Software registry in stand-alone mode.
rtool	Runs the MSC Software registry shell tool.
cmm	*Windows only* Runs a Python tool prompting you to enter various uconfg parameter sizes. A new uconfg_user.dll is created.
exit	Returns to the system level; exiting at any of the other prompts returns you to this menu.
help	Displays information about each of the other choices.

Entering Selection Codes

To perform operations through the Adams Selection Menu, you enter a selection code, after the prompt at the bottom of the screen. You can use either upper- or lower-case letters, and you can abbreviate to the shortest unique abbreviation (except when entering file names). For example, you can use e to indicate exit, h to indicate help, ru-s to indicate ru-standard, and av to indicate aview. If you use an incorrect abbreviation and get an error message, press Enter to continue. Whenever you complete or interrupt a program, the Adams Selection Menu returns.

Getting Help

Anytime that you are using the Adams Selection Menu, you can enter **help** or **h** to get help. Help text appears in the window. To exit from the help, press **Enter**.

Exiting the Selection Menu

You can enter **exit** or its abbreviation (**e**) at any prompt to exit from the prompt and return to the Selection Menu. If you enter **exit** at the Selection Menu, it returns to the operating system prompt.

Command-Line Parameters

In addition to using the Adams Selection Menu and entering selection codes and responding to prompts, you can enter command-line parameters. You enter command-line parameters at the operating system prompt.

To enter command line parameters, you enter the command to run Adams Selection Menu, followed by a series of strings representing prompt selections. You can enter the complete selection codes or abbreviations. To indicate pressing Enter, use -**none** or its abbreviation (-**n**). Use spaces to separate the various parameters. As the command executes, menus or prompts do not appear.

If you make an error in a command-line parameter, an error message appears and prompts you to enter exit. If you enter **exit**, the command aborts and displays the Adams Selection Menu.

The following table lists information about the keywords and parameters:

Keyword	If LINUX	Options	Parameters
/top_dir/mdi		exit	Action: exit command menu
/top_dir/mdi	-c	ru-user	Action: run solver with standard executable Arguments: abc.acf : adams command file *Example* *LINUX: mdi -c ru-st i abc.acf ex* *Windows: mdi ru-st abc.acf ex*
/top_dir/mdi	-c	ru-user	Action: run solver with user executable Arguments: abc.dll or abc.so : user executable abc.acf : adams command file *Example* *LINUX: mdi -c ru-u abc.so i abc.acf ex* *Windows: mdi ru-u abc.dll abc.acf ex*

Keyword	If LINUX	Options	Parameters
/top_dir/mdi	-c	cr-user	Action: create user executable Argument: y/n : yes/no option to link in debug mode .o : object file compiled with FORTRAN, C or C++ or .c, .cxx,.f : source file or .lst : list of object or source files -n : (none) to mark end for list of arguments abc.dll or abc.so : name of library to be created e : exit *Example* *LINUX: mdi -c cr-u n sub.f -n cam.so ex* *Windows: mdi cr-u n sub.f -n cam.dll ex*
/top_dir/mdi	-c	aview	Action: launch aview Arguments: cr-user : refer to cr-u option listed above ru-standard : run standard aview executable i/b - option for interactive or batch mode ru-user : run aview with user executable i/b - option for interactive or batch mode abc.dll - view user library xyz.dll - solver user library *Example* *LINUX: mdi -c aview ru-u i abc.so xyz.so ex* *Windows: mdi aview ru-s i abc.dll xyz.dll ex*

Keyword	If LINUX	Options	Parameters
/top_dir/mdi	-c	acar	Action: launch acar -- Create -- Options cr-acarprivate : acar private library (use / MDI_ACAR_PRIVATE_DIR for lib) cr-acarsite : acar site library (use /MDI_ACAR_SITE for lib) cr-solverprivate: acar solver library (use /MDI_ACAR_PRIVATE_DIR for lib) cr-solversite : acar solver site library (use /MDI_ACAR_SITE for lib) cr-privatebin : Create custom binary file cr-sitebin : Create custom binary file Arguments: y/n : yes/no option to link in debug mode .o : object file compiled with FORTRAN, C or C++ or .c, .cxx,.f : source file or .lst : list of object or source files -n : (none) to mark end for list of arguments acar_solver.dll or acar_solver.so : name of user library to be created at above mentioned directory for the library e : exit

Keyword	If LINUX	Options	Parameters
			For more details check: Running Adams Products > On Windows --- Run --- ru-acar : with either the Private, Site or Std binary ru-private : private acar.bin ru-site : site acar.bin ru-standard : standard acar.bin ru-solver : run adams car with private, site and standard solver *Example* *LINUX: mdi -c acar ru-sol i abc.acf ex* *Windows: mdi acar ru-sol abc.acf ex*
/top_dir/mdi	-c	acsi	Action: launch co-simulation interface *Example* *LINUX: mdi -c* *Windows: mdi acsi*

Keyword	If LINUX	Options	Parameters
/top_dir/mdi	-c	adriveline	Action: launch adriveline Options: -- Create --- cr-adrivelinepriv : adriveline private library(use /MDI_ADRV_PRIVATE_DIR for lib) cr-adrivelinesite : adriveline site library (use /MDI_ADRIVELINE_SITE for lib) cr-solverprivate : adriveline solver library (use /MDI_ADRV_PRIVATE_DIR for lib) cr-solversite : adriveline solver site library (use /MDI_ADRIVELINE_SITE for lib) cr-privatebin : Create custom binary file cr-sitebin : Create custom binary file Arguments: y/n : yes/no option to link in debug mode .o : object file compiled with FORTRAN, C or C++ or .c, .cxx,.f : source file or .lst : list of object or source files -n : (none) to mark end for list of arguments adriveline_solver.dll or adriveline_solver.so : name of user library to be created at above mentioned directory for the library e : exit

Keyword	If LINUX	Options	Parameters
			For more details check: Running Adams Products > On Windows -- Run -- ru-adriveline : with either the Private, Site or Std binary ru-private : private adriveline.bin ru-site : site adriveline.bin ru-standard : standard adriveline.bin ru-solver : run Adams car with private, site and standard solver *Example* *LINUX: mdi -c adriveline ru-sol i abd.acf ex* *Windows: mdi adriveline ru-sol abd.acf ex*
/top_dir/mdi	-c	aexplore	Action: launch aexplore job-server and web-server Arguments: ru-jobserver : starts job server ru-webserver : starts webserver *Example* *LINUX: mdi -c aexplore ru-jobserver* *Windows: mdi aexplore ru-jobserver*

Keyword	If LINUX	Options	Parameters
/top_dir/mdi	-c	drill	Action: launch adrill Arguments: ru-startserver : run adrill servers (LINUX only) 2202: a port number for server 'scratch/nkdm/adrill_pvt' : ADRILL_DATA_DIR data directory *Example* *LINUX: mdi -c drill 2202 'scratch/drill_dir'* *Windows: mdi drill*
/top_dir/mdi	-c	ainsight	Action: launch ainsight *Example* *LINUX: mdi -c ainsight* *Windows: mdi ainsight*
/top_dir/mdi	-c	appt	Action: launch post processor *Example* *LINUX: mdi -c appt* *Windows: mdi appt*
/top_dir/mdi	-c	redit	Action: Launch msc registry editor *Example* *LINUX: mdi -c redit* *Windows: mdi redit*

Keyword	If LINUX	Options	Parameters
/top_dir/mdi			Action: msc registry tools Supported commands: cd : change directory; absolute or relative paths example: cd /vendor/tools/interface/settings pwd : print working directory set : set property - absolute or relative path to property name example: set /graphics/perf/DoubleBuffering TRUE set ../../loads/Max_settings 12 unset : unset property - absolute or relative path to property name example: unset /graphics/perf/DoubleBuffering get : print individual property's value - absolute or relative path example: get /graphics/driver/HPUX11 ls : prints all property names and values, datatypes, ranges in current directory only dir : prints all property names and values, datatypes, ranges in current directory and subdirectories, recursively files : lists locations of the implementation registry files, number of lines last parsed from each prompt : changes prompt example: prompt pwd - changes prompt to display current directory prompt normal - changes prompt to display default 'REG>>' string

Keyword	If LINUX	Options	Parameters
			verbose : toggles verbose mode on and off. default=on exit : saves changes and exits quit : exits without changes *Example* *LINUX: mdi -c rtool* *set //TOP_DIR/MDI/ACar/Preferences/privateDir /usr/people/someone/new_private* *WINDOWS: mdi rtool* *set D://TOP_DIR/MDI/ACar/Preferences/privateDir D:/people/someone/new_private*
/top_dir/mdi	-c	python	Actions: Launch python command prompt *Example* *LINUX: mdi -c python* *WINDOWS: mdi python*
/top_dir/mdi	-c	help	Actions: Launch help options

Keyword	If LINUX	Options	Parameters
/top_dir/mdi	-c	flextk	Actions: launch flex toolkit Options: -- Translate -- mnf2mtx : MNF file to Matrix file msc2mnf : MSC/NASTRAN file to MNF file mnfload : Add Load-cases to MNF file unv2mnf : Universal file to MNF file mnfxform : MNF file to XFORM file mnfres : Export flex body nodal simulation results abq2nas : Abaqus input file to NASTRAN input file *Example* *LINUX: mdi -c flextk msc2mnf NASTRAN_OUT2_file* *WINDOWS: mdi flextk msc2mnf NASTRAN_OUT2_file*
/top_dir/mdi	-c	durtk	Actions: launch durability toolkit -- Options -- durfilter : Filter durability files mnf2fes : Translate MNF to nCode FES file res2dur : Translate results to durability files *Example* *LINUX: mdi -c durtk* *WINDOWS: mdi durtk*

Keyword	If LINUX	Options	Parameters
/top_dir/mdi	-c	cmm	Actions: update custom memory model For more details, see Setting Custom Memory Model Size. *Example* *LINUX: mdi -c cmm* *WINDOWS: mdi cmm* *-> Update 8 values prompted*
/top_dir/mdi	-c	-meplot	Actions: Displays standard plots *Example* *LINUX: mdi -c meplot plot_temp_name res_name* *WINDOWS: mdi meplot plot_temp_name res_name*
/top_dir/mdi	-c	-makeleaf	Actions: Displays standard plots *Example* *LINUX: mdi -c makeleaf leaf_template_file* *WINDOWS: mdi makeleaf leaf_template_file*

where:

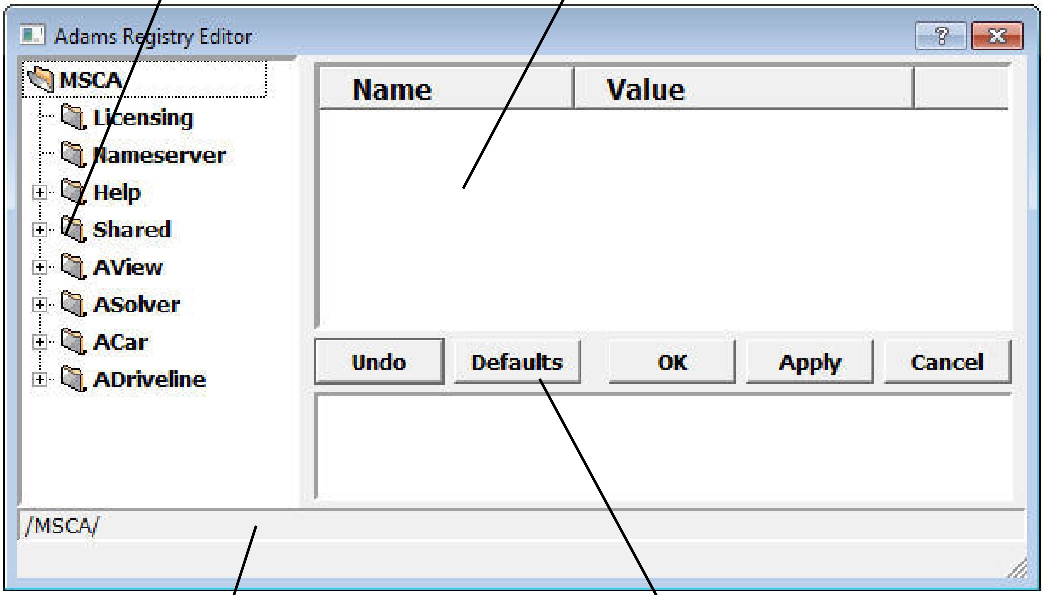
- The -**n** indicates pressing **Enter** in response to the second subroutine file prompt and e represents exit. Because the last **e** is a response to the menu, the operating system prompt returns after creating the library file.
- The **y** answers the question about working in debug mode and the -**n** indicates a return in response to the second subroutine file prompt. The system level returns after linking the library file. You must use spaces to separate the various parameters. When entering source files, be sure the **.f** or **.c** extension is lowercase.

Adams Registry Editor

The Registry Editor lets you edit the settings that the Adams Toolbar stores for the different Adams products. It appears whenever you create a user library or set preferences. The elements of the Registry Editor are shown in the figure below, as they appear when you select to view all elements in the Registry Editor. To help you find files, the Registry Editor provides a file browser where you can enter a specific file name or browse for a file or directory.

Treeview - Displays folders of preferences

Resgistry area - Lists settings for the folder selected in treeview and lets you edit its value



Resets parameters to default settings

Message Area - Displays messages

To make a choice in the Registry Editor:

1. In the treeview, click a folder that you want to change or view.

   The Registry area displays the contents of the folder you selected.

2. Click an item listed in the Registry area.

   The value of the item appears as a text box, option menu, or check box to let you change or set the value. If the registry item that you selected cannot be changed, nothing changes.

3. Select another item, select **OK**, or select a different folder in the treeview.

Note:	If you modify a registry item, and then select a new folder to browse in the treeview, the Editor prompts you to save your modifications.

4. Select **No** and then **OK** to save your changes or select **Yes** to exit without saving the changes. You can also select **Always throw away changes and switch directories** so you no longer receive this message.

To browse for a file or files:

1. Right-click a text box that requires a file, and then select **Select A File** or **Select Files**.

   A file browser appears.

   To help narrow your search, you can browse for a specific file type using a file extension. For example, list of object or source files have the extension *.lst. Therefore, to search for only list of object or source files, set File Type in the browser to *.lst. You can select multiple files by clicking each file name you want.

2. Select **OK**.

   The dialog box closes and the current text box displays the files you selected.

To browse for a directory:

1. Right-click a text box that requires a directory to be specified, and then select **Select A Directory**.

   A directory browser appears.

2. Browse through the directories and click the one you want.
3. Select **OK**.

   The dialog box closes and the current text box displays the directory you selected.

Troubleshooting

If you have questions or problems with any of the Adams products, get help by viewing the Simcompanion web page at http://simcompanion.mscsoftware.com/. Our Knowledge Base articles and Frequently Asked Questions (FAQs) are especially useful. If the problems persist, contact Technical Support. Before contacting Technical Support, read the topics described next.

- Windows Problem Reports
- Linux Problem Reports
- Technical Support

Windows Problem Reports

On supported Windows operating systems there are two varieties of problem reports:

- **Troubleshooting Report:** this examines your system configuration and places important debugging information in the file adams_problem_report.txt. If you have problems running Adams products, you can generate this report and send it to MSC Software. After you generate the report, you may want to save it under a different name because Adams overwrites the file each time you generate a report.

- **Crash Dump File:** in the event that the Adams View or Adams Car interfaces crashes (terminate unexpectedly) then often a dump file (.dmp) will be exported to the working directory. This is a non-human-readable binary file containing diagnostic information which MSC developers can use to investigate and fix problems in Adams causing such crashes. It is currently only supported on Windows. Note that the dump file does **not** contain any potentially confidential data like model files, other user data and so on. It stores only information pointing to where in the software specifically Adams crashes and the conditions that might have caused the crash.

To generate a Troubleshooting Report on Windows:

1. From the **Start** menu, point to **Programs**, point to **Adams x** (where ***x*** is the release number), and then select **Troubleshooting Report**.

   A message appears informing you that the report is being created and providing the name of the report.
2. Open the **adams_problem_report.txt** file using a text editor and edit the first page of the report to include your name and phone and fax numbers. Add any comments in the comments section.

   If you are an experienced Adams user, you may want to look over the report for error messages before sending it to MSC Software.
3. Send the file using e-mail or print the file and fax it to the technical support for your area.
4. To close the window, press **Enter**.

To find the Crash Dump File on Windows:

If the Adams View or Adams Car graphical interfaces ever crash, a .dmp file is often (but not always) written to the current working directory of Adams. The file name will typically start with "SCAKernel" followed by an 8-digit date of the form "yyyymmdd" and then a series of other numbers. Please attach this .dmp file in any communication with MSC Technical Support about the crash. Note that the dump file does **not** contain any potentially confidential data like model files, other user data and so on. It stores only information pointing to where in the software specifically Adams crashes and the conditions that might have caused the crash.

Linux Problem Reports

Using the Adams Toolbar, you can examine your system configuration and place important debugging information in the file PROBLEM.RPT. If you have problems running Adams products, you can generate this report and send it to MSC Software.

Before sending the report, open it into a text editor and edit the first page of the report to include your name and telephone and fax numbers. You can also add any comments in the comments section. You should save the file under a different name since Adams overwrites the file each time.

Either send the file using e-mail or print the file and fax it to Technical Support for your area.

Note:	For correct results, you must run the hotline (problem) report from the license server, not the client machine.

To generate a troubleshooting report on Linux:

1. Right-click the Adams Toolbar tool, point to **Support**, and then select **Technical Support**.
2. Select **Generate Hotline Report**.

   An Alert dialog box informs you that the report has been created and gives you the name of the report. This file uses the default file name of PROBLEM.RPT and places the file in the directory $HOME, where $HOME is your home directory.
3. Select **Close**.

Technical Support

To obtain technical support, contact your local support center, which can be found at http://simcompanion.mscsoftware.com/KB8019304

Before contacting Technical Support, have the following information available:

- Version of FORTRAN, if any.
- Copy of all error messages (you can send it by fax or through e-mail).
- Copy of the Adams View log file (aview.log) and Adams Solver message file (.msg).
- Hardware type.
- Version of the Linux operating system, if applicable (for example, 11 on the HP 9000/700).
- Troubleshooting report.

To find out more about contacting Technical Support on Linux:

1. Right-click the Adams Toolbar tool, point to **Support**, and then select **Technical Support**.

   Information about the various ways in which you can obtain technical support appears.
2. After viewing the information, select **Close**.

Running Adams Products

On Linux

On Linux, use the Adams Toolbar to run Adams products and libraries of user-written subroutines

Modes in Which You Can Run Products

You can run Adams products in the following modes:

- **Interactive mode** - The product starts (a graphical interface is displayed or at least would be displayed if a monitor is hooked up to the machine) and waits for you to enter commands. Typically this means using the graphical interface to interactively work with the product and any commands are actually entered automatically by the product behind the scenes. But, one can use the interface's command window to issue commands directly. Also, one can provide files containing commands (aka "scripts") to issue a series of commands at once, including files that are automatically run during startup of the product.
- **Scripted** - The product runs with a command file that you specify (aka a "script"). A command file is either a set of Adams View commands (.cmd) or Adams Solver (.acf) commands, depending on the product that you are running. The command file helps you automate the creation of a model, perform a simulation, or investigate simulation results. "Scripted" mode is really nothing more than executing a command file in either Interactive mode or Batch mode.
- **Batch mode** - No graphical interface product is ever displayed. You must provide a file of commands (aka a "script") to be run. Information about the batch run is collected in a batch log file that you specify. Batch mode differs only slightly from importing a command file during an interactive session in the graphical interface in that the graphical interface is never actually displayed. Be advised that many commands available in the Adams View command language may require the graphical interface to be displayed in order to execute. So, not all command scripts which work interactively can be expected to work in batch mode.

You can set the same modes when you run a product with a user library.

In addition, you can set debug mode (also referred to as interactive debug mode) when you run a user library. Debug mode runs a debug utility, a system-level program, usually dbx, that steps you through, or isolates parts of, the subroutines in the user library. The debug utility helps you detect and locate any problems in the user libraries. You must have created the library in debug mode. To learn how to create a user library in debug mode, see Creating User Libraries.

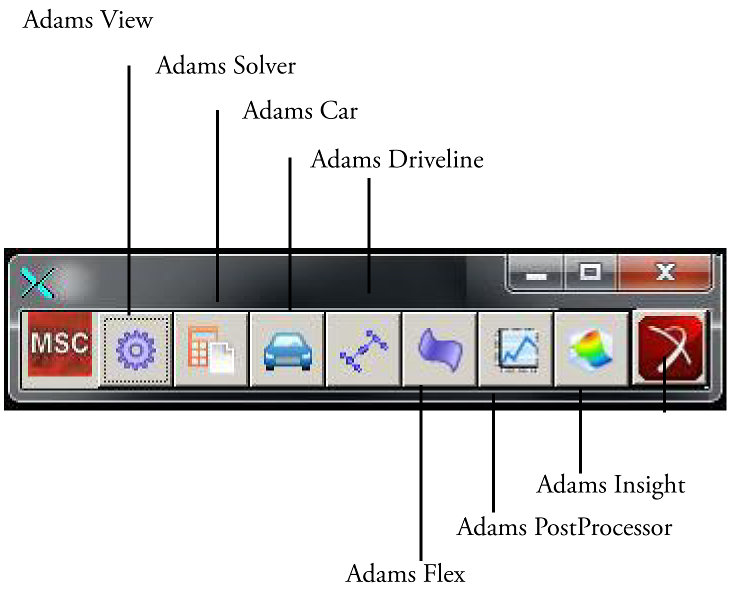
Standard Products

You can use the Adams Toolbar to run standard products by clicking the associated tool. Each Adams product runs using its default preferences.

Note: To run add-on modules or plugins, such as Adams Durability, Adams Flex, or Adams Vibration, you must first run the product in which the plugin runs. For example, to run Adams Vibration, first run Adams Car or Adams View, and then select the command to run Adams Vibration. For more information on running plugins, see their online help.

To run a standard product:

- Click the product tool.



Selecting the product's icon, with a left mouse click, starts the product. If the product default is to run in scripted mode, the product runs the specified command file.

For more information on running Adams products, refer to the online help for your product.

Template-Based Products

MSC Software provides several products built on Adams View, referred to as Adams template-based products. The products are:

- Adams Car lets you create, catalogue, and analyze suspension and full-vehicle assemblies.
- Adams Driveline lets you model drivelines to create and analyze virtual prototypes of driveline subsystems.

The template-based products let you select to run their user interface or their version of Adams Solver. In addition, you can select a binary file that the product reads at startup to change the look of menus, dialog boxes, and commands.

When you run a template-based product, the Toolbar searches the product's private, site, and standard location, in that order, for a user library to run. It runs the first library that it finds. If you want to run just the standard version of these products, be sure that there are no libraries in your private or site location.

The template-based product then reads the binary file that you have specified. You can specify that it read the binary file that is in either the private, site, or standard locations. You can also specify that it search the private, site, and standard location and read the first binary it finds.

To run template-based products with their interface and a particular binary:

- Right-click the product tool, point to **Run**, point to [**Product**] - **Interface**, and then select a binary. For example, for Adams Car, right-click the **Adams Car** tool, point to **Run**, point to **ACar - Interface**, and then select a binary.

To run template-based products with Adams Solver:

- Right-click the product tool, point to **Run**, and then select [**Product**] - **Solver**. For example, for Adams Car, right-click the **Adams Car** tool, point to **Run**, and then select **ACar - Solver**.

Running User Libraries

User libraries are subroutines that extend the functionality of Adams products to meet individual needs. For example, you can use a library of subroutines that define functions for motion or force magnitudes. You run a user library by selecting it from Adams Toolbar and then running it with its associated product. For more on user libraries, see User Library Overview.

When you select to run a user library with Adams View and Adams Solver, the associated product tool on the Toolbar changes to indicate that you are working with a user library. For example, for Adams View, the tool changes from to . You can then select to run the product with the user library as the default by clicking the tool.

You can also run Adams View with an Adams Solver user library by setting it as a default preference, as explained in Adams View Preferences.

For Adams Car, you cannot specifically select a user library. Instead, the Adams Toolbar searches your private, site, and standard locations, in that order, for a user library to run.

To run Adams View or Adams Solver with a user library:

1. Right-click the product tool, point to **Select Library**, and then select a user library.

   For example, for Adams Solver, right-click the **Adams Solver** tool , point to **Select Library**, and select a library, such as **Asolver1**.

The tool changes to indicate that the default is to run the product with the selected user library. For example, **Adams Solver** tool changes to .

2. Click the product tool again.

On Windows

On Windows, use the Adams program folder and Selection Menu.

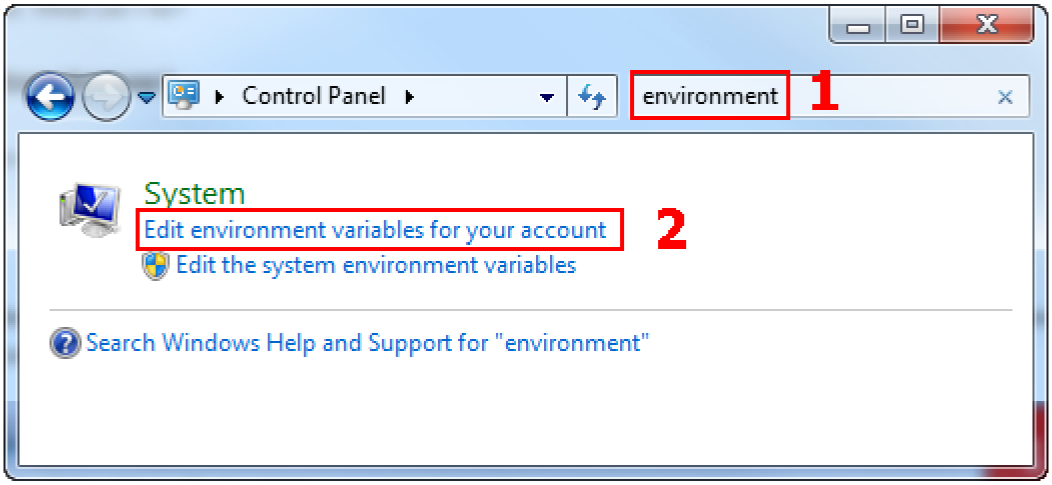
Running Adams on Windows

The Adams program folder and Selection Menu provide access to many of the Adams major products. Adams also has add-on modules or plugins, which expand the functionality of the major products, such as Adams Flex, Adams Controls, Adams Durability, and Adams Vibration. You run these products from within the major products. For example, to run Adams Vibration, you first run Adams Car or Adams View and then select the command to run Adams Vibration.

For instructions on running Adams products, refer to the online help for your product.

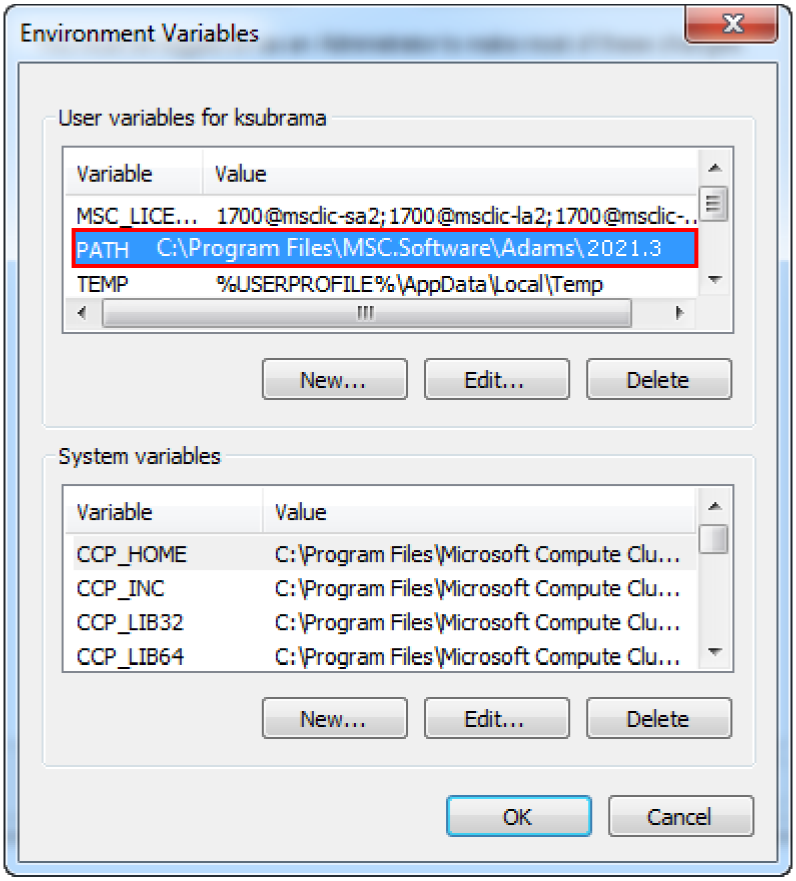
To start a product from DOS shell in Windows:

- **Start** → **Adams 2021.3** → **Command Prompt**
- An alternative is to add the **/bin** directory under the Adams installation directory to your user PATH environment variable. You can do this via the Control Panel:
    a. Type **environment** in the Control Panel search box.
    b. Select the **Edit environment variables for your account** pick.


Control Panel
environment
1
System
Edit environment variables for your account
2
Edit the system environment variables
Search Windows Help and Support for "environment"

- Check to see if you already have a PATH variable defined under "**User variables**". If it exists then edit it, add a semicolon ";" to the end of the existing value, and then add the "bin" directory under the Adams installation directory. Otherwise select **New...**, use PATH for the variable name and add the "bin" directory under the Adams installation directory as the value. The default location of this directory is:

```
Win64 Adams:       C:\Program Files\MSC.Software\Adams\2021_3\bin
```



Environment Variables

User variables for ksubrama

Variable	Value
MSC_LICE...	1700@msclic-sa2;1700@msclic-la2;1700@msclic-...
PATH	C:\Program Files\MSC.Software\Adams\2021.3
TEMP	%USERPROFILE%\AppData\Local\Temp

System variables

Variable	Value
CCP_HOME	C:\Program Files\Microsoft Compute Clu...
CCP_INC	C:\Program Files\Microsoft Compute Clu...
CCP_LIB32	C:\Program Files\Microsoft Compute Clu...
CCP_LIB64	C:\Program Files\Microsoft Compute Clu...

- Once you are done click **OK**.

Note:	Windows appends your User PATH to the System PATH environment variable, so there is no need to copy the existing System PATH variable to your user PATH variable. This behavior is unique to the User PATH variable. For all other environment variables, a User variables definition overrides a System Variables definition.

Running Adams Solver

You can run either standard Adams Solver or Adams Solver with a user library, as explained below.

Standard Adams Solver

You can select to run Adams Solver without the Adams View graphical interface. Adams Linear and Adams Tire are analysis modules of Adams Solver. You access them by selecting the Adams Solver option, if you are licensed to do so.

To run Adams Solver:

1. Do one of the following:
    - From the **Start** menu, point to **Programs**, point to **Adams *x*** (where *x* is the release number), and then select **Adams Solver**.
    - From the Adams Selection Menu, enter **ru-standard**.

      A window appears prompting you for information.
2. Enter the name of the Adams command file (.acf), if you have created one, or press **Enter**.

   Adams Solver starts running.

Note:	The Adams command file (.acf) cannot reside in a directory whose path includes either an exclamation point (!) or an ampersand (&).

For more information on the commands and execution of Adams Solver, see the Adams Solver online help.

User Libraries with Analysis Products

You can run user Adams Solver libraries from either the Program Folder or the Selection Menu, as well as run user libraries with Adams Tire.

To run a user library:

1. From the **Start** menu, point to **Programs**, point to **Adams *x*** (where *x* is the release number), and then select **Command Prompt**.
2. Enter **ru-user**.
3. Enter the name of the custom library that you want to run.
4. Enter the name of the Adams command file (.acf), if you have one, or press **Enter**.

   Adams Solver starts running.

For more information on the commands and execution of Adams Solver, see the Adams Solver online help.

Running Adams View

Adams View is a powerful modeling and simulating environment, which helps you solve your design problems. You can use Adams View to build, simulate, and refine virtual models of any mechanical system. You can run Adams View from the program folder or the selection menu, as well as run it with user libraries. Running Adams View from the Selection Menu lets you select the mode in which to run Adams View.

For information about working with Adams View, refer to its online help.

- Adams View from the Program Folder
- Standard Adams View from the Selection Menu
- User Libraries with Adams View

Adams View from the Program Folder

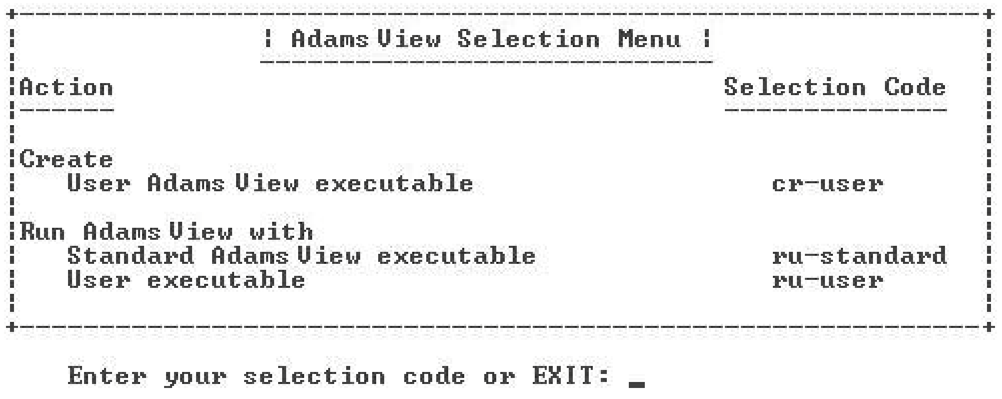
To run Adams View from the program folder:

- From the **Start** menu, point to **Programs**, point to **Adams *x*** (where *x* is the release number), and then select **Adams View**.

  The Adams View main window appears.

Standard Adams View from the Selection Menu

You use the Adams View Selection Menu that appears when you select aview from the main Selection Menu to run and create Adams View user libraries.



```
+----------------------------------------------------------------------+
|                     | Adams View Selection Menu |                    |
|                     -----------------------------                    |
|Action                                                 Selection Code |
|------                                                 -------------- |
|                                                                      |
|Create                                                                |
|   User Adams View executable                              cr-user    |
|                                                                      |
|Run Adams View with                                                   |
|   Standard Adams View executable                          ru-standard|
|   User executable                                         ru-user    |
|                                                                      |
+----------------------------------------------------------------------+

    Enter your selection code or EXIT: _
```

This selection code:	Provides this action:
cr-user	Leads you through the creation of an Adams View library.
ru-standard	Selects standard Adams View.
ru-user	Runs Adams View with a user library.

When you run a Adams View, you can select the following modes:

- **Interactive mode** - The product starts (a graphical interface is displayed or at least would be displayed if a monitor is hooked up to the machine) and waits for you to enter commands. Typically this means using the graphical interface to interactively work with the product and any commands are actually entered automatically by the product behind the scenes. But, one can use the interface's command window to issue commands directly. Also, one can provide files containing commands (aka "scripts") to issue a series of commands at once, including files that are automatically run during startup of the product.
- **Batch mode** - No graphical interface product is ever displayed. You must provide a file of commands (aka a "script") to be run. Information about the batch run is collected in a batch log file that you specify. Batch mode differs only slightly from importing a command file during an interactive session in the graphical interface in that the graphical interface is never actually displayed. Be advised that many commands available in the Adams View command language may require the graphical interface to be displayed in order to execute. So, not all command scripts which work interactively can be expected to work in batch mode.

To run standard Adams View through the Selection Menu:

1. In the Adams Selection Menu, enter **aview**.
2. Enter **ru-standard**.
3. Select the mode in which you want to run the executable.
   - To run in interactive mode, select **i** or press **Enter**.
   - To run in batch mode, select **b**.
4. If you selected batch mode, enter the name of a file containing Adams View commands.

   The Adams View main window appears.

User Libraries with Adams View

You can run Adams View with user libraries that you created, including Adams Solver user libraries.

Note:	You can also set the Adams Solver user library after you've started Adams View. For information on setting the Adams Solver user library after starting Adams View, select **Settings** -> **Solver** -> **Executable**, and then press **F1** for help.

To run Adams View with a user library:

1. From the **Start** menu, point to **Programs**, point to **Adams** ***x*** (where *x* is the release number), and then select **Command Prompt**.

   The Selection Menu displays.
2. Enter **aview**.
3. Enter **ru-user**.
4. Enter the name of the Adams View user library.

5. Enter the name of an Adams Solver user library.

   Adams View begins running.

Running Template-Based Products

MSC Software provides products built on Adams View, referred to as Adams template-based products. The products are:

- Adams Car lets you create, catalogue, and analyze suspension and full-vehicle assemblies.
- Adams Driveline lets you model drivelines to create and analyze virtual prototypes of driveline subsystems.

You can run the template-based products alone or with different user libraries in your private, site, and standard locations. Adams Car, or Adams Driveline runs the first user library that it finds as it searches your private, site, and, finally, the standard location. If you use the Selection Menu to run a product, you can also choose the binary (for example, acar.bin) you want to use. A binary contains the database information for a customized interface.

For information on the different products, see the online help for that product. For information on creating user libraries, see Custom Template-Based Product Libraries.

Template-Based Products from the Program Folder

The Adams program folder lets you run the template-based products through Adams View or Adams Solver. When you enter a command to run a template-based product, the product searches the private, site, and standard locations for a user library to run. It runs the first library it finds.

Tip:	If you want to run the standard version of the product without a user library, be sure that there are no user libraries in your private or site location.

To run template-based products from the program folder:

- Depending on whether you want to run the product through Adams Solver or Adams View, do one of the following:
  - From the **Start** menu, point to **Programs**, point to **Adams *x*** (where *x* is the release number), and then select the appropriate command. For example, select **Adams Car (solver)**.

    The Adams Car, Adams Driveline (Solver) main command window appears.
  - From the **Start** menu, point to **Programs**, point to **Adams *x*** (where *x* is the release number), and then select the appropriate command. For example, select **Adams Car (view)**.

    The Adams Car, or Adams Driveline, (View) main window appears.

Template-Based Products from the Selection Menu

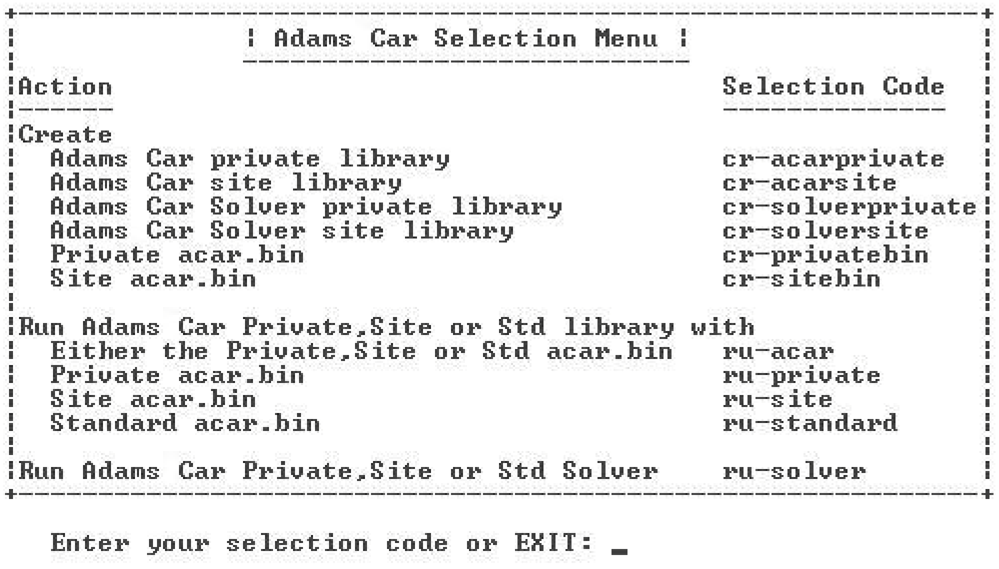
The Selection Menu lets you run the template-based products and choose the binary that you'd like to use. The following sections explain how to use the Selection Menu to choose the version for the product to run:

Displaying the Selection Menu

To display the template-based product's selection menu:

- At the Adams Selection Menu, enter a**car**, or **adriveline**.

  A Selection Menu appears. The following figure applies to Adams Car. The Adams Driveline, Selection menu contain the same type of options.



```
+-------------------------------------------------------------+
!               ! Adams Car Selection Menu !                  !
!               ----------------------------                  !
!Action                                       Selection Code  !
!------                                       --------------  !
!Create                                                       !
!  Adams Car private library                  cr-acarprivate  !
!  Adams Car site library                     cr-acarsite     !
!  Adams Car Solver private library           cr-solverprivate!
!  Adams Car Solver site library              cr-solversite   !
!  Private acar.bin                           cr-privatebin   !
!  Site acar.bin                              cr-sitebin      !
!                                                             !
!Run Adams Car Private,Site or Std library with               !
!  Either the Private,Site or Std acar.bin    ru-acar         !
!  Private acar.bin                           ru-private      !
!  Site acar.bin                              ru-site         !
!  Standard acar.bin                          ru-standard     !
!                                                             !
!Run Adams Car Private,Site or Std Solver     ru-solver       !
+-------------------------------------------------------------+

  Enter your selection code or EXIT: _
```

The code(s):	Do(es) the following:
cr-acarprivate, cr-acarsite, cr-solverprivate, cr-solversite	Creates a user Adams Car library and places it in your home (private) or site directory. The string after the cr- identifies where Adams Car places the library and whether the version runs with Adams View or Adams Solver.
cr-privatebin, cr-sitebin	Creates a custom Adams Car binary file that Adams Car runs when it starts up. The string after the cr- identifies where Adams Car places the binary. **Note:** We strongly encourage you to review your product's log file (acar.log, aride.log, and so on) for any warnings or error messages that may have occurred during the building of the binary file.
ru-acar	Runs the first private, site, or standard library of Adams Car that it finds. It also uses the first version of the Adams Car binary that it finds as it searches the private, site, and standard locations.
ru-private, ru-site, ru-standard	Runs the first private, site, or standard library of Adams Car that it finds and then uses the specified private, site, or standard binary. The string after the ru- identifies which binary to use.
ru-solver	Runs the Adams Car version of Adams Solver that it finds as it searches your private, site, and standard locations.

Running Template-Based Products with Adams View

When you enter a selection code to run a template-based product, the product searches the private, site, and standard locations for a library to run. It runs the first library that it finds.

Tip:	If you want to run the standard version of the product without a user library, be sure that there are no user libraries in your private or site location.

The template-based product then reads the binary file that you have specified. You can specify a binary file in the private, site, or standard locations. You can also specify to search the private, site, and standard locations and read the first binary found.

To run template-based products with Adams View:

1. At the Adams Selection Menu, enter **acar**, or **adriveline**.
2. At the Selection Menu, enter one of the selection codes depending on the binary file that you want to read. For more information on the selection codes, refer to the table above.

Running Template-Based Products with Adams Solver

You can run the standard version of a template-based product with Adams Solver or run user libraries that are in the private or site locations. The Selection Menu searches the private, site, and standard locations and runs the first user library that it finds.

Tip:	If you want to run the standard version of the product without a user library, be sure that there are no user libraries in your private or site location.

To run template-based products with Adams Solver:

1. At the Adams Selection Menu, enter **acar**, or **adriveline**.
2. Enter **ru-solver**.
3. At the command file prompt, do one of the following:
   - Press **Enter**.

     A main command window appears.
   - To launch the product with a command file, type the name of the command file and press **Enter**.

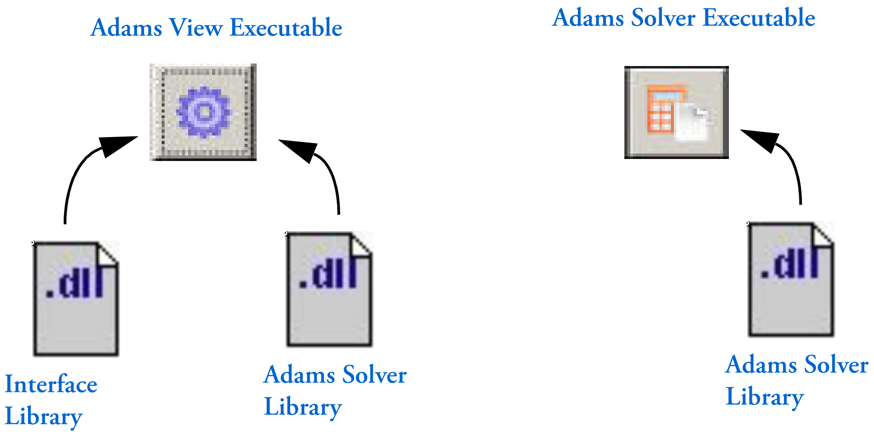
     The product reads in your command file and its main command window appears.

Creating User Libraries

User Library Overview

You can create user libraries and run them with your Adams products. There are two types of user libraries:

- Adams Interface or Adams View libraries - These libraries let you change the definition of your model by adding compiled design-time functions for use in Adams View expressions. You can also change post-processing by adding or changing functions that determine how simulation results are computed. You can use these functions in the same way you would use the built-in design-time functions. For more information, see the Adams View Function Builder online help.
- Adams Solver libraries - These libraries let you add run-time functions for motion or force magnitudes to directly define the behavior of your model and change the way the simulation is performed. They let you take advantage of existing software to define complex modeling relationships, such as hydraulic actuators or tire forces. For more information on writing user-written subroutines, refer to the Adams Solver online help.



Products, such as Adams View, and Adams Car, can run both an interface library and an Adams Solver library because they run Adams Solver internally. To set the Adams Solver user library to run internally with these products, set the preference solverUserLibrary as explained in Adams View Preferences.

Once you create a library, you need to set up your modeling entities, such as motions or forces, to reference these subroutines and use the library whenever you perform a simulation on models referencing these subroutines. For information on using the subroutines in your model definition, refer to the Adams Solver and Adams View online help.

When you run an Adams Solver library, the associated Adams product only loads the library when a user triggers it, such as when a user selects a command that executes the subroutines in the library. This is because Adams Solver libraries are demand-loaded. When you run an interface library, the associated Adams product loads the library at startup so that design-time functions can be registered and available for immediate use.

Creating User Libraries

You can create user libraries for the products listed in the table shown next. The table also lists the type of user library you can create (interface or Adams Solver) for that product and where you can place the user library.

For the product:	You can create the user library:	Its location:
Adams Car	Interface and Adams Solver	Private or site
Adams Driveline	Interface and Adams Solver	Private or site
Adams Solver	Adams Solver	Any directory
Adams View	Interface and Adams Solver	Any directory

Learn how to:

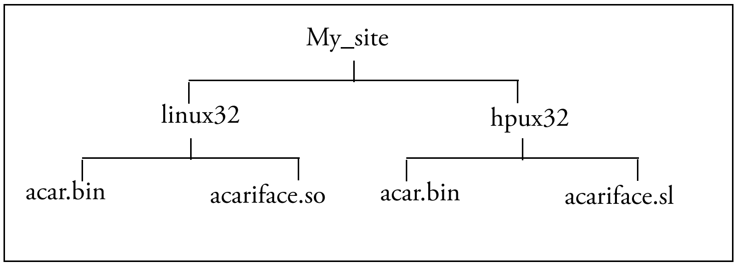
- Create User Libraries on Linux
- Create User Libraries on Windows

Windows will compile C code for Adams Solver, but Linux does not. Currently, the scripts on Linux do not support the compiling of C code. Once the C code is compiled for Linux, it can be included in an Adams Solver user library.

For Adams Car, and Adams Driveline, you can only specify either your home (private) directory or the site location, to which all users have access. On Linux, the Adams Toolbar automatically creates a directory structure to help manage the different libraries. If you are creating a site library, be sure that you set up the site directory as explained in Directory Structure for Template-Based Products.

Directory Structure for Template-Based Products

To create libraries and binaries for template-based products, you need to have the proper directory structure. An example of the structure of a site location for Adams Car with two different platform libraries (Linux and HP) is shown next.



The process for creating the directory structure varies by platform.

- Linux
- Windows

Linux

To help you create binaries and libraries for the template-based products, Adams Car, Adams Driveline, the Adams Toolbar creates a directory structure for you as you create libraries and binaries. For each platform for which you create a library or binary, the Toolbar creates a different subdirectory in which to store the library or binary.

- For the private location, by default, Adams Toolbar uses the directory that it creates when you first run a template-based product. You can also specify its location as explained in Template-Based Product Preferences. You must have permission to write to the directories. If you do not have write permission, the product returns an error.
- For a site location, you must create the base site directory and specify its location as explained in Template-Based Product Preferences.

Windows

Before creating a site library, you need to create a site directory and specify its location, as explained in the online help for your template-based product. If you are creating a private library, Adams Car, and Adams Driveline, uses the private directory that it creates when you first run the product. You must have permission to write to the directories. If you do not have write permission, the product returns an error.

The product stores the binaries and libraries in the directory at the same level.

Debugging User Libraries

Debugging on Linux

You can choose to create the user library in debug mode. Running a user library in debug mode invokes the *debug* utility, a system-level program, that steps you through the subroutines or isolates parts of them. The debug utility helps you detect and locate any problems in the user-written subroutines.

Debugging on Windows

To debug an Adams Solver user library:

1. Create a library file with debugging information as explained in User Libraries on Windows. From the command window, you would enter:

   ```
   adams2021_cr-user y user.f -n mysolver.dll
   ```

   Make sure the .f extension is lowercase.
2. Run the library file in debug mode as explained in Running Adams Solver. From the command window, you would enter:

   ```
   adams2021_ru-user mysolver.dll user.acf
   ```

   Microsoft Visual Studio.Net automatically starts up.
3. In MS Visual Studio, from **Solution Explorer**, right-click on **solver**, and then select **Properties**.

4. In the General settings ensure that **Executable** is set to be:

   `install_dir/solver/win64/solver.exe`

   where ***`install_dir`*** is the actual installation directory, that is, **C:\Program Files\MSC.Software\Adams\2021_3w.**
5. Set the **Working Directory** to the location where the ***.acf***, ***.dll***, and ***.adm*** files are located.
6. Set the **Arguments** to **user.acf** and then select **Apply**.
7. Open **user.f** in MS Visual Studio and set a breakpoint on a line in your user subroutine.
8. Run the project, using:

   Debug → Start

Note:	If this is your first time debugging, you will have to create a solutions file by pressing **Save**. A command window appears, showing Adams Solver initialization commands. The program execution then halts in the debugger at the breakpoint that you've specified.

To debug an Adams View interface library:

- Follow the instructions above for Adams Solver libraries, except:
  - The command to create the library is:

    adams2021_3 aview cr-user y user.c -n myview.dll
  - The command to run in debug mode is:

    adams2021_3 aview ru-u i myview.dll -n
  - You browse for:

    ***install_dir*/aview/win32/aview.exe**

Linux

User Libraries on Linux

To create a user library on Linux:

1. Do one of the following:
   - For a product for which you can only create one type of user library, right-click the product tool, point to **New**, and then select **[Product] User Library**.
   - For a product for which you can create either an interface or Adams Solver library, right-click the product tool, point to **New**, point to **[Product] - Interface** or **[Product]-Solver**, and, if required, select the location for the library you want to create.

The Enter Parameters for User Library list appears in the Registry area with several parameters that you can modify.

2. Click the parameters you want to change and enter their values as needed.

Parameter:	Meaning:
name	A project name identifies the user library, such as AView1. It also determines the file name of the library, such as AView1.so. The project name must be unique. You cannot create two libraries with the same name. Not available for Adams Car, and Adams Driveline, because you cannot run the resulting library directly from the Adams Toolbar.
description	Text that describes the library, such as what it does or how it was created. Not available for Adams Car,and Adams Driveline, because you cannot run the resulting library directly from the Adams Toolbar.
libPath	A directory indicating where to store the resulting library. Not available for Adams Car, and Adams Driveline, because you specify the library location when you first create it as either your private or site location.
sourceFiles	One or more individual C or FORTRAN files, source or object files, or a list file (*.lst) that contains a list of C or FORTRAN files. You can mix C and FORTRAN source files. The list file must contain one file per line.
debug	Selects whether or not to build the library in debug mode.

The parameters, name and libPath, have default values based on a simple counting scheme and the name of the product for which the user library is being created, such as AView1, AView2, ASolver1, and so on. You can use other values that have more significance.

Note: You must enter one or more source files for the parameter sourceFiles. If you do not enter source files, the Adams Toolbar does not create a library.

3. Select **OK**.

A window appears, showing the compilation and linking process that builds your custom library. When the creation is complete, the following occurs:

- A new entry appears on the Select Library menu from the product icon on the Toolbar. The name of the file is the name of the menu.
- If the directory you specified for libPath does not exist, Adams Toolbar creates it. It places the new library and a log file about the creation in the directory.
- Adams Toolbar creates or changes the appropriate registry entries to reflect the new entity.

Deleting User Libraries

You can delete any library registered in the Adams Toolbar. When you delete the library, Adams Toolbar asks you if you want to just remove the library from the Toolbar or delete the actual library file from its directory. If you select to just remove the library from the Toolbar, you can add it back into the Toolbar as explained in Importing Existing User Libraries.

To delete a library:

- Right-click the tool for the product whose library you want to import, and then select **Remove Custom:** [**customName**]. For example, right-click the **Adams View** tool, and then select **Remove Custom: Aview1**.

Importing Existing User Libraries

You can import user libraries that have been created by others so you can run them from your Adams Toolbar.

To import user libraries:

1. Right-click the tool for the product whose library you want to import, and then select **Import Existing Custom Library**. For example, if you are importing an Adams View library, right-click the **Adams View** tool, and then select **Import Existing Custom Library**.
2. Click name and enter the name of the library file.
3. Click description and enter text describing the library.
4. Select **OK**.

   Adams Toolbar adds the library to the list of libraries available to run with that product.

Windows

User Libraries on Windows

On Windows, you can create user libraries for the following:

- Adams View
- Template-based products
- Analysis products
- NO_FORTRAN
- Linking Custom Libraries

Custom Adams View Libraries

You can add your own C routines to create a user Adams View library. Sample C functions are in the file vc_init_usr.c in the directory /install_dir/aview/usersubs/, where install_dir is the directory in which you installed Adams software.

Note:	You can also use a list file (.lst) to enter the names of several object files at once. You must place an @ sign before the name of the list file (for example, @names.lst).

To create an Adams View library:

1. From the **Start** menu, point to **Programs**, point to **Adams x** (where *x* is the release number), and then select **Command Prompt**.
2. Enter **aview**.
3. Enter **cr-user**.
4. At the debug prompt, enter one of the following options:
   - To run in debug mode, enter **y**.
   - To run in standard mode, enter **n** or press **Enter**.
5. Enter the name of the first object, source, or list file containing your subroutines.

   You can enter an object file (for example, mysubs.obj), a source file (.f or .c), or a .lst file (for example, @sub_list.lst). If entering a source file, be sure the .f or .c extension is lowercase.
6. Repeat Step 2 as many times as necessary by either:
   - Entering an object file containing one or more user-written subroutines.
   - Pressing **Enter** to end the list of subroutines linked to Adams View.
7. Enter a name for the Adams View user library that you want to create.

   You've now created a user library that Adams View can run.

Custom Template-Based Product Libraries

For Adams Car, and Adams Driveline, you can create user interface or Adams Solver libraries.

You can create the libraries in your private (home) directory where only you can access them or place them in the site location for all users to access. Learn about the Directory Structure for Template-Based Products.

For information on running the libraries, see Running Template-Based Products. For more information on customizing these products, see their online help.

Creating Libraries for Template-Based Products

The following steps explain how to create interface or Adams Solver user libraries that run with Adams Car, Adams Driveline.

You can choose to place any of the libraries in your private or site location. If you are creating a library in your site location, be sure to set up the directory, as explained in Directory Structure for Template-Based Products.

To create an interface user library:

1. From the **Start** menu, point to **Programs**, point to **Adams x** (where *x* is the release number), and then select **Command Prompt**.
2. Enter **acar**, or **adriveline**.
3. Enter the code to create a private or site library (for example, cr-acarprivate or **cr-acarsite**).
4. At the debug prompt, enter one of the following options:
   - To run in debug mode, enter **y**.
   - To run in standard mode, enter **n** or press **Enter**.
5. Enter the name of the first object, source, or list file containing your subroutines.

   You can enter an object file (for example, mysubs.obj), a source file (.f or .c), or a .lst file (for example, @sub_list.lst). If entering a source file, be sure the .f or .c extension is lowercase.
6. Repeat Step 4 as many times as necessary by either:
   - Entering an object file containing one or more user-written subroutines.
   - Pressing **Enter** to end the list of subroutines linked to Adams View.

To create an Adams Solver library:

1. From the **Start** menu, point to **Programs**, point to **Adams x** (where *x* is the release number), and then select **Command Prompt**.
2. Enter **acar**, or **adriveline**.
3. Enter the code to create a private or site library (for example, enter **cr-solverprivate** or **cr-solversite**).
4. At the debug prompt, enter one of the following options:
   - To run in debug mode, enter **y**.
   - To run in standard mode, enter **n** or press **Enter**.
5. Enter the name of the first object, source, or list file containing your subroutines.

   You can enter an object file (for example, mysubs.obj), a source file (.f or .c), or a .lst file (for example, @sub_list.lst). If entering a source file, be sure the .f or .c extension is lowercase.
6. Repeat Step 4 as many times as necessary, by either:
   - Entering an object file containing one or more user-written subroutines.
   - Pressing **Enter** to end the list of subroutines.

User Libraries for Analysis Products

You can create user libraries for the Adams analysis products, which include Adams Solver, Adams Controls, and Adams Tire. You can also run the user libraries from within Adams View when running Adams View with an integrated Adams Solver.

Sample user FORTRAN subroutines are in the **tire.f** file in the directory c:/*install_dir*/solver/samples/, where *install_dir* is the directory in which you installed Adams software.

Note: You can also use a list file (.lst) to enter the names of several object files at once. You must place an @ sign before the name of the list file (for example, @names.lst).

To create a user library:

1. From the **Start** menu, point to **Programs**, point to **Adams x** (where *x* is the release number), and then select **Command Prompt**.
2. Enter **cr-user**.
3. At the debug prompt, enter one of the following options:
   - To run in debug mode, enter **y**.
   - To run in standard mode, enter **n** or press **Enter**.
4. Enter the name of the first object, source, or list file containing your subroutines.

   You can enter an object file (for example, mysubs.obj), a source file (.f or .c), or a .lst file (for example, @sub_list.lst). If entering a source file, be sure the .f or .c extension is lowercase.
5. Repeat Step 2 as many times as necessary, by either:
   - Entering an object file containing one or more user-written subroutines.
   - Pressing **Enter** to end the list of subroutines linked to Adams Solver.
6. Enter a name for the Adams Solver library you want to create and press **Enter** to begin the linking process.

NO_FORTRAN

Adams Solver supports user subroutines that are written in C. When all of your user subroutines are written in C then NO_FORTRAN compiler is required to compile them. However, on windows when Adams builds user libraries that are to be run with the Solver, Adams assumes FORTRAN is installed on and includes the FORTRAN libraries to its link command. To communicate to the linking scripts that NO_FORTRAN libraries are required, set the environment variable NO_FORTRAN to 1.

For example:

```
C:\temp>set NO_FORTRAN=1
C:\temp>adams2021_3 cr-u n myfile.c -n mylib.dll
```

Linking Custom Libraries

If you need to include one or more custom libraries when you create your user library, you can set an environment variable to instruct the linker which libraries to include.

For the library type:	Use the environment variable:
Interface	MSC_CUSTOM_LINK_LIBS
Adams Solver	MYSLIBS

For example, if you are creating an interface library, and your custom libraries are named my_custom_library1_imp.lib and my_custom_library2_imp.lib, set the environment variable as follows:

```
C:\temp>set MSC_CUSTOM_LINK_LIBS="my_custom_library1_imp.lib
my_custom_library2_imp.lib"
```

If you are creating a Solver library, and your custom libraries are named my_solver_lib1_imp.lib and my_solver_lib2_imp.lib, set the environment variable as follows:

```
C:\temp>set MYSLIBS="my_solver_lib1_imp.lib my_solver_lib2_imp.lib"
```

Once the user library has been created, the environment variable is no longer needed.

Managing Files

Binary Files

You can create a binary file, acar, or adriveline, that contains database information for the customized menus, dialog boxes, and buttons that appear in template-based products (Adams Car, or Adams Driveline). When you start a template-based product with your binary file, the product reads in your customized interface instead of the standard binary file.

You can create the binary files in your private (home) directory where only you can access them or place them in the site location for all users to access.

To specify the interface changes that you want in the binary file, you create a command file containing Adams View commands for modifying the interface. For example, the following commands create a new menu named **New Menu** and a button named **DO IT** under the menu:

```
interface menu create &
   menu_name=.gui.main.aca_sta_mbar.new_menu &
   enabled=yes &
   label ="New Menu"

interface push_button create &
   push_button_name = .gui_man.aca_sta_mbar_new_menu.do_it &
   enabled = yes &
   label = "DO IT" &
   command = "list_info entity=.ACAR"
```

The name of the command file must be acar_build.cmd, or adriveline_build.cmd, (depending on your product) and must be in your private or site directory.

For information on directory structure for storing binary files, see Directory Structure for Template-Based Products. For more on creating acar_build.cmd, see the Customize tab in your template-based product's online help. For more on command files, see the Adams View online help.

There is no restriction on the size of the binary file. While there is no theoretical limit to .bin file size, practically speaking, the size of a .bin file will be limited to the amount of RAM available to the Adams session.

To create a binary file on Windows:

1. From the **Start** menu, point to **Programs**, point to **Adams *x*** (where *x* is the release number, for example Adams 2021.3), and then select **Command Prompt**.
2. Enter **acar**, or **adriveline**.
3. Enter the code to create the binary (for example, enter **cr-privatebin** or **cr-sitebin**).

To create a binary file on Linux:

1. Right-click the **Adams Car**, **Adams Driveline**, tool, point to **New**, point to [**Product**] **Binary**, and then select the location for the binary file. For example, for Adams Car, point to **Adams Car**, point to **New**, point to **ACar Binary**, and then select the location for the binary file.
2. Select **OK**.

Note:	We strongly encourage you to review your product's log file (acar.log, and so on) for any warning or error messages that may have occurred during the building of the binary file.

Language and Compiling

You can submit your subroutines as any of the following types of files:

- FORTRAN and/or C source files.
- Object file containing one or more user-written subroutines.
- File containing a list of source files, with one name per line.

To find out what type of compiler you need, read the hardware and software specifications that come with your Adams product (you can also see them at http://support.mscsoftware.com). Also, for further instructions, refer to your compiler documentation and the hardware and software specifications.

Linux

You can either let Adams Toolbar run your compiler by supplying source files, or you can compile and link your subroutines on your own and create object files. We recommend, however, that you let Adams Toolbar run your compiler since there is less chance for errors. It also ensures that the correct compile flags are set so that your files are compiled and linked using the same options that were used for Adams product code. We also recommend that you purchase a debugger.

To view the list of compiler options valid on your platform, use the following command:

```
adams2021_3 -c cr-user n
```

Windows

You can input C or FORTRAN source files to create Adams Solver user libraries and only C source files to create Adams View user libraries. You must use the following C and FORTRAN compiler command and options for your user subroutines. The options are explained in the table shown below.

For FORTRAN, the command and options are:

ifort/c /architecture:pn4 /MD /Ob2 /automatic /Z7

Note:	/Z7 should only be used for debug mode.

For C, the command and options are:

cl /c /Ox /MD /Z7

This variable:	Provides the following:
/c	No linking
/architecture:pn4	Pentium 4 optimization (FORTRAN compiler)
/MD	Multi-threaded applications (Adams is a multi-threaded application)
/Ob2 or /Ox	Automatic inlining
/automatic	Puts local variables on the run-time stack
/Z7	Adds debugging information

The linking procedure allows you to supply FORTRAN or C files (depending on the Adams product).

Sample Source Files

For Adams Car, Adams Solver, and Adams View, we've provided sample source files for creating user libraries in the directory /install_dir/adams_product/usersubs/, where /install_dir/ is the directory in which you installed Adams software and adams_product is the name of the product.

Adams File Types

Adams View, Solver, Linear and PostProcessor

Extension	Description	Product
cmd	Adams View command script. Defines a model, can also define macros and GUI customization.	Adams View
bin	Adams View binary file. Typically contains a model, can also contain macros and GUI customization.	Adams View
py	Python command script. Can define a model, can also extend the Python interface in Adams View	Adams View
adm	Adams Dataset Model - the most basic Adams Solver model definition	Adams Solver
acf	Adams Command File - Adams Solver commands for simulating a model	Adams Solver
req	Output Request file. Contains all data defined in REQUEST elements	Adams Solver

Extension	Description	Product
gra	Output Graphics file. Contains part transformations for animating with Adams PostProcessor	Adams Solver
res	Output Results file. A comprehensive listing of all output time histories from a simulation. Can contain REQUEST data, Measure data, CONTACT incidents and all part kinematics.	Adams Solver
msg	Adams Solver output message file.	Adams Solver
dac	Test data, can be imported or exported	Adams View, Solver
txt	System modal information tables.	Adams Solver, Linear
nas	Adams2Nastran exported data file	Adams Solver, Linear
dat	Nastran exported data file	Adams Solver, Linear
log	Nastran exported log file	Adams Solver, Linear
jac	Jacobian system matrix	Adams Solver
rhs	Right-hand side solution vector	Adams Solver
mtx	Solver matrix file	Adams Solver
plt	Plot configuration file. Stores plot layouts for later reconstruction in Adams PostProcessor	Adams PostProcessor
htm	Output report format, replicating plots and reports in PostProcessor.	Adams PostProcessor
Geometry File		
xmt_txt, x_t	ASCII Parasolid model file	Adams View
xmt_bin, x_b	Binary Parasolid model file	Adams View
igs, iges	CAD data file format	Adams View
stp, step	CAD data file format	Adams View
sat, asat	ACIS Geometry Kernel file format	Adams View
model	CATIA V4 model file	Adams View
session	CATIA V4 session file	Adams View
CATPart	CATIA V5 part file	Adams View
CATProduct	CATIA V5 assembly file	Adams View
3dxml	CATIA V6 model file	Adams View
ipt	Autodesk Inventor part file	Adams View
iam	Autodesk Inventor assembly file	Adams View
prt	ProE part file	Adams View
asm	ProE assembly file	Adams View

Extension	Description	Product
sldprt	Solidworks part file	Adams View
sldasm	Solidworks assembly file	Adams View
jt	Jupiter Tessellation 3D data format	Adams View

Adams Car, Driver, Explore, PostProcessor, Smartdriver and Tire

Extension	Description	Product
asy	Model file, class assembly, in table assemblies.tbl	Adams Car
tpl	Model file, class template, in table templates.tbl	Adams Car
sub	Model file, class subsystem, in table subsystems.tbl	Adams Car
aer	Property file, class aero_force, in table aero_forces.tbl	Adams Car
asp	Property file, class airspring, in table airsprings.tbl	Adams Car
bum	Property file, class bumpstop, in table bumpstops.tbl	Adams Car
bus	Property file, class bushing, in table bushings.tbl	Adams Car
fmu	Property file, class fmu_bushing, in table bushings.tbl	Adams Car
dpr	Property file, class damper, in table dampers.tbl	Adams Car
dif	Property file, class differential, in table differentials.tbl	Adams Car
edb	Property file, class edm_bushing, in table bushings.tbl	Adams Car
edd	Property file, class edm_damper, in table dampers.tbl	Adams Car
mnf	Property file, class flex_body, in table flex_bodys.tbl	Adams Car
MASTER	Property file, class MD_database, in table flex_bodys.tbl	Adams Car
bdf	Property file, class NL_flexbody, in table flex_bodys.tbl	Adams Car
afi	Property file, class viewflex_input, in table flex_bodys.tbl	Adams Car
gea	Property file, class gear_element, in table gear_elements.tbl	Adams Car
fgf	Property file, class adv_3d_gear_element, in table gear_elements.tbl	Adams Car
lbf	Property file, class linear_bushing, in table bushings.tbl	Adams Car
ldf	Property file, class linear_damper, in table dampers.tbl	Adams Car
lsf	Property file, class linear_spring, in table springs.tbl	Adams Car
pwr	Property file, class powertrain, in table powertrains.tbl	Adams Car
gpf	Property file, class gas_pressure_force, in table powertrains.tbl	Adams Car
reb	Property file, class reboundstop, in table reboundstops.tbl	Adams Car

Extension	Description	Product
gfo	Property file, class gear_force, in table gear_stiffness.tbl	Adams Car
gfs	Property file, class rotational_stiffness, in table gear_stiffness.tbl	Adams Car
gcp	Property file, class adv_3d_gear_contact, in table gear_stiffness.tbl	Adams Car
fgp	Property file, class adv_3d_gear_force, in table gear_stiffness.tbl	Adams Car
shl	Property file, class shell_graphic, in table shell_graphics.tbl	Adams Car
x_t	Property file, class parasolid, in table shell_graphics.tbl	Adams Car
spr	Property file, class spring, in table springs.tbl	Adams Car
tsf	Property file, class torsion_spring, in table complex_springs.tbl	Adams Car
tbf	Property file, class torsion_beam_spring, in table springs.tbl	Adams Car
ste	Property file, class steering_assist, in table steering_assists.tbl	Adams Car
tcf	Property file, class torque_converter, in table torque_converters.tbl	Adams Car
ltf	Property file, class leafspring, in table leafsprings.tbl	Adams Car
mtr	Property file, class motor, in table motors.tbl	Adams Car
bldc	Property file, class motor_bldc, in table motors.tbl	Adams Car
rtp	Property file, class report, in table report_templates.tbl	Adams Car
stpr	Property file, class motor_stepper, in table motors.tbl	Adams Car
spl	Property file, class spline, in table gen_splines.tbl	Adams Car
gsp	Property file, class gen_spline, in table gen_splines.tbl	Adams Car
str	Property file, class strut, in table dampers.tbl	Adams Car
dcf	Analysis types, class driver_controls, in table driver_controls.tbl	Adams Car
sdf	Analysis types, class smart_driver_file, in table driver_controls.tbl	Adams Car
dcd	Analysis types, class driver_data, in table driver_data.tbl	Adams Car
asc	Analysis types - Input file, class asc_driver_data, in table driver_data.tbl	Adams Car
dri	Analysis types - Input file, class driver_loadcase, in table driver_loadcases.tbl	Adams Car
lcf	Analysis types - Input file, class loadcase, in table loadcases.tbl	Adams Car
drv	Analysis types - Input file, class rpc_input, in table loadcases.tbl	Adams Car

Extension	Description	Product
wen	Analysis types - output file, class wheelenv, in table wheel_envelopes.tbl	Adams Car
vsf	Analysis types, class vehicle_setup_file, in table vehicle_setups.tbl	Adams Car
wnd	Analysis types - Input file, class cross_wind, in table winds.tbl	Adams Car
aci	Generic actuation analysis, class actuation_input, in table actuation_inputs.tbl	Adams Car
rrm	Generic actuation analysis, class req2rpc_map, in table req2rpc_maps.tbl	Adams Car
ssf	Solver settings , class solver_settings, in table solver_settings.tbl. Over-rides model Solver settings.	Adams Car
nam	Request and component name for .req file	Adams Car
sdf	Smartdriver driver input file	Adams Car, Smartdriver
sdl	Smartdriver log file	Adams Car, Smartdriver
xml	Smartdriver event file	Adams Car, Smartdriver
_runTime.log	Smartdriver debug log file	Adams Car, Smartdriver
_bsp.pth	Smartdriver debugging file for input path	Adams Car, Smartdriver
_raw.pth	Smartdriver debugging file for input path	Adams Car, Smartdriver
_tra.pth	Smartdriver debugging file for input path	Adams Car, Smartdriver
plt	Postprocessing template, class plot_config, in table plot_configs.tbl	Adams Car, Postprocessor
cmd	Adams command file, class script, in table scripts.tbl	Adams Car, Explore
xlsx	Model files, class workbook, in table workbooks.tbl	Adams Car, Explore
drd	Driver Analysis, class driver_road, in table driver_roads.tbl	Adams Car, Driver
kno	Driver Analysis, class driver_knowledge, in table driver_knowledge.tbl	Adams Car, Driver
din	Driver Analysis, class driver_input, in table driver_inputs.tbl	Adams Car, Driver
tir	Tires Property file, class tire, in table tires.tbl	Adams Car, Tire
rdf	Roads Property file, class road, in table roads.tbl	Adams Car, Tire
crg	Roads Property file, class grid_road, in table roads.tbl	Adams Car, Tire
Geometry File		
xmt_txt, x_t	ASCII Parasolid model file	Adams Car
xmt_bin, x_b	Binary Parasolid model file	Adams Car
igs, iges	CAD data file format	Adams Car
stp, step	CAD data file format	Adams Car

Extension	Description	Product
sat, asat	ACIS Geometry Kernel file format	Adams Car
model	CATIA V4 model file	Adams Car
session	CATIA V4 session file	Adams Car
CATPart	CATIA V5 part file	Adams Car
CATProduct	CATIA V5 assembly file	Adams Car
3dxml	CATIA V6 model file	Adams Car
ipt	Autodesk Inventor part file	Adams Car
iam	Autodesk Inventor assembly file	Adams Car
prt	ProE part file	Adams Car
asm	ProE assembly file	Adams Car
sldprt	Solidworks part file	Adams Car
sldasm	Solidworks assembly file	Adams Car
jt	Jupiter Tessellation 3D data format	Adams Car

Adams Flex, MaxFlex and Durability

Extension	Description	Product
mnf	Modal neutral file	Flex
MASTER	MSC Nastran database (MD DB)	Flex, MaxFlex
mtx	Matrix file for flexible bodies	Flex, MaxFlex
dat	Run-ready input file to MSC Nastran	Flex, MaxFlex
bdf	Input file to MSC Nastran for bulk data (mesh)	Flex, MaxFlex
f06	Output file from MSC Nastran	Flex, MaxFlex
f04	Output file from MSC Nastran	Flex, MaxFlex
log	Log file from MSC Nastran	Flex, MaxFlex
op2	OUTPUT2 file from MSC Nastran	Flex, MaxFlex
sts	Conversion summary from MSC Nastran	MaxFlex
cslog	Log file from MSC Nastran	MaxFlex
mdf	Modal Displacement File	Flex, Durability
unv	Universal file	Flex, Durability
inp	Input file to Abaqus	Flex
rsp	Data file in RPCIII format	Durability

Extension	Description	Product
rpc	Data file in RPCIII format	Durability
dac	Data file in DAC format	Durability
asc	Input file to FE-Fatigue	Durability
fef	Output file from MSC Fatigue or FE-Fatigue	Durability
laf	Load association file for FE-Fatigue	Durability
ndf	Nodal deformation data file	Durability
nsf	Nodal stress/strain data file	Durability
csv	File in comma separated value format	Durability
tab	Table data file of hot spots	Durability

Adams Driveline and Ride

Extension	Description	Product
bea	Property file, class bearing, in table bearings.tbl	Adams Driveline
csp	Property file, class complex_spring, in table complex_springs.tbl	Adams Driveline
tsf	Property file, class torsion_spring, in table complex_springs.tbl	Adams Driveline
wcf	Property file, class wet_clutch, in table clutch_forces.tbl	Adams Driveline
clu	Property file, class clutch_force, in table clutch_forces.tbl	Adams Driveline
clf	Property file, class clutch_connector, in table clutch_forces.tbl	Adams Driveline
rti	Property file, class ride_tire, in table ride_tires.tbl	Adams Driveline
tor	Property file, class torque_loadcase, in table torque_loadcases.tbl	Adams Driveline
hyp	Property file, class hypoid_gear, in table hypoid_gears.tbl	Adams Driveline
tcf	Property file, class torque_converter, in table torque_converters.tbl	Adams Driveline
sta	Analysis types - Input file, class static_loadcase, in table static_loadcases.tbl	Adams Driveline
dyn	Property file, class dyno, in table dynos.tbl	Adams Driveline
dav	Property file, class dyno_curve_angle, in table dynos.tbl	Adams Driveline
dti	Property file, class dyno_curve_time, in table dynos.tbl	Adams Driveline
fdb	Property file, class fd_bushing, in table fd_bushings.tbl	Adams Driveline
pgf	Property file, class planetary_gear, in table gear_elements.tbl	Adams Driveline
rgf	Property file, class ravigneaux, in table gear_elements.tbl	Adams Driveline
hbu	Property file, class hydro_bushing, in table hydro_bushing.tbl	Adams Ride

Extension	Description	Product
fbu	Property file, class frequency_bushing, in table frequency_bushing.tbl	Adams Ride
gsd	Property file, class gse_damper, in table gse_damper.tbl	Adams Ride
rsp	Road Profile data in the RPC III file format- Input file, class road_profile_rpc, in table road_profiles.tbl	Adams Ride
rpt	Road Profile data in table, class road_profile_table, in table road_profiles.tbl	Adams Ride
sfd	Property file, class single_f_d_element, in table dampers.tbl	Adams Ride
gfd	Property file, class general_f_d_element, in table bushings.tbl	Adams Ride
gbu	Property file, class general_bushing, in table general_bushing.tbl	Adams Ride

Adams Controls

Extension	Description
m	Matlab configuration file. Created by Adams Controls for a particular Adams model to define inputs, outputs, installation location and so on.
inf	Easy5 configuration file. Created by Adams Controls for a particular Adams model to define inputs, outputs, installation location and so on.
dll	Compiled library created from Matlab or Easy5 specifically for Adams Controls. Contains equations describing the external system model.
fmu	Functional Mockup Unit file. Implements the Functional Mockup Interface (FMI) protocol. File contains model files.
_adams_master_fmi_plant_info	Used by Adams Controls during FMI simulations. It does not have a standard extension but it ends with the string in the extension column.
.adme	Encrypted model file (.adm file).

Adams Machinery

Extension	Description
bldc	Brushless DC motor property file
dpr	Damper property file for guides
fgf	Gear profile data for Advanced 3D gear

Extension	Description
fgp	Gear contact property file for Advanced 3D gear
gcp	Precomputed contact data for Advanced 3D gear
gea	Gear property file
gfo	3D gear force property file
gfs	Simplified gear force property file
lpt	Load pattern file for Advanced 3D gear
mtr	Motor property file
spr	Spring property file for guides
stpr	Stepper motor property file
wzd	Wizard save restore file

Adams Drill

Extension	Description	Table
acc	Accelerator property file	accelerators.tbl
agn	Agitator property file	agitators.tbl
bre	Blade_reamer property file	blade_reamers.tbl
col	Drill_collar property file	drill_collars.tbl
crs	Crossover property file	crossovers.tbl
dat	Hole property file	NA
djr	Jar property file	jars.tbl
drt	Dart property file	darts.tbl
evt	Event property file	events.tbl
flp	Flex_pipe property file	flex_pipes.tbl
gnl	Generic_long property file	generic_longs.tbl
gnm	Generic_medium property file	generic_mediums.tbl
gns	Generic_short property file	generic_shorts.tbl
hol	Hole property file	holes.tbl
hwp	Hw_pipe property file	hw_pipes.tbl
ins	Instrumentation_sub property file	instrumentation_subs.tbl
lwd	Lwd_tool property file	lwd_tools.tbl
mfr	Mfr_tool property file	mfr_tools.tbl

Extension	Description	Table
mot	Motor property file	motors.tbl
mwd	Mwd_tool property file	mwd_tools.tbl
pdc	Pdc_bit property file	pdc_bits.tbl
pip	Drillpipe property file	drill_pipes.tbl
pip	Equivalent_upper_string property file	drill_pipes.tbl
plt	Plot_config property file	plot_configs.tbl
rcb	Roller_cone_bit property file	roller_cone_bits.tbl
rsd	Rss property file	rss.tbl
sco	Short_collar property file	short_collars.tbl
shk	Shock_sub property file	shock_subs.tbl
ssf	Solver_setting property file	solver_settings.tbl
sta	Stabilizer property file	stabilizers.tbl
str	Assembly file	drill_strings.tbl
tdr	Top_drive property file	top_drives.tbl
xlam	Addin file for macros for Ribbon and menus	NA
xlsm	Spreadsheet with utility macros	NA
xlsx	adril_comp.xlsx holds properties of components	NA

Acoustics

Extension	Description
a2ac	Post processing file from Actran
dat	Header property file for Actran
edat	Actran output file
finished	Actran processing output
info	Actran output file
mdf	Durability output file for Actran input
mic	Mic property file
par	Input parameter file for Actran
plt	Plot configuration file

Extension	Description
resources	Actran output file
sess	Session file for Actran input
wav	Sound file from Actran output

See section 'Exchanging Data in Adams' and 'Supported File Formats for Import and/or Export' for more details.

Adams Configuration Files

You can set up the following files to change the way in which the interface products, including Adams Car, and Adams View, work:

- Path Files
- Startup Files
- Log Files
- Binary Files

Path Files

An Adams interface product creates an Adams default aview.pth file at installation time. It is in the aview product subdirectory, underneath the installation directory, install_dir, on your machine.

The aview.pth file allows you to specify search paths for the different types of files that can be loaded into Adams. This is useful for files that are shared with other users, and are not in your current working directory. Below is an example of an aview.pth file:

```
.bin    /install_dir/aview/
.idb    $workdir/idbfiles/
.adm    ~jsmith/Adams datasets/
.cmd    /usr/aview/cmd_files/
.dat    /usr/Adams test_data/
.gra    /staff/my_home_dir/Adams output/
.req    /staff/my_home_dir/Adams output/
.res    /staff/my_home_dir/Adams output/
.igs    /staff/my_home_dir/iges_files/
.dct    /install_dir/aview/
```

Each line in the file specifies a search path for a specific file extension. The first line in the example file above specifies a search path for files with an extension of .bin. You can have multiple paths for each extension by adding another line with the same file extensions and an alternate path. Remember each line must begin with a file extension.

Any file extension can be put in the Adams path file. Therefore, you do not have to name your files with the standard Adams Solver and Adams View extensions. Use the following conventions to add a file extension to the path file:

- Begin each line in column one with the desired file extension.

- Begin each file extension with a . (period).
- Use the correct case for file extensions in the Path file. Adams is case sensitive with regard to file extensions. All Linux systems are case sensitive for the path.
- Separate the extension and search path for the extension with a space. You can use a single space and tab or combine spaces and tabs to create separation between the extension and search path.
- Complete each search path, starting from the root ('/') of the file system. You can begin a search path with the ~ character to indicate a Linux home directory, or with a $ character to indicate a Linux environment variable.
- End each search path with a trailing /.
- Ensure that each line is less than 255 characters in length.

If you want to make the changes to the path file affect all Adams users, edit the default aview.pth file mentioned above.

To make the changes only affect you:

1. Copy the default Adams path file to your current working directory.
2. Add, delete, or modify the file to specify your file search paths.
3. From the Adams Toolbar, right-click the **Adams View** tool , and then select **Change AView Settings**.

   The **searchPath** is visible in the treeview.
4. Click **searchPath** and specify the full path to your local aview.pth file, including the file name. You can type in the path or use the browser to select it.

Startup Files

The three files that interface products read when they start up, where product is the name of the product, (aview, acar, and adriveline) are:

- product.cmd
- productBS.cmd
- productAS.cmd

In the file names, BS stands for Before Startup, meaning that Adams reads it before it reads other setup files, and AS stands for After Startup, meaning that Adams reads it after it reads other setup files.

The startup files contain Adams View commands that Adams executes on startup. You can edit any of the files to execute Adams as desired when it starts up. Note that many products overwrite the BS and AS files when you select to save options, so you may lose the edits you make. For more information on the content of command files and how to create them, refer to the Adams View online help.

Adams first searches for the command files in the current working directory. If a command file does not exist locally, Adams searches for it in the directories specified for .cmd files in the aview.pth file.

The command file names are hard-coded, and you cannot change them.

Log Files

The aview.log file is a text file containing a log of your commands for an Adams session. Adams creates this file in your local directory.

While running Adams, you can change the name of the log file for that session with the command:

```
FILE LOG_FILE FILE_NAME = "filename"
```

Adams overwrites any existing aview.log file each time it starts.

For more information on the content of log files and how to create them, refer to the Adams View online help.

Binary Files

Note:	When changing binary files for Adams Car, we recommend that you follow the instructions in Binary Files.

The aview.bin file is a binary file containing the Adams View database. Adams View reads it on startup. Adams View first searches for aview.bin in the current working directory. If aview.bin does not exist locally, Adams View searches for it in directories specified for .bin files in the aview.pth file.

The aview.bin provided with Adams View contains database information for the standard menus, panels, and buttons. You can create your own copy of aview.bin to save a customized interface (menus, panels, and buttons you have modified). When you start Adams View again in the directory containing your version of aview.bin, Adams View reads in your customized interface and ignores the standard aview.bin file.

You can also create an aview.bin to save the state of your modeling session. If you use the name aview.bin for the file, the next time you start Adams View in that directory, Adams View automatically reads the aview.bin file and restores your modeling session. Alternatively, you can use a different name for the binary file you save. To restore the state of you modeling session, issue the Adams View command:

```
FILE BINARY READ FILE_NAME = "filename"
```

The name aview.bin is hard-coded in Adams View as the default binary file it reads upon startup. You cannot change it.

Using Memory Models

Memory Models

If you have a large model and are using the Adams FORTRAN based solver, you may need to change your memory allocation to avoid program faults during simulations.

Remember that creating or using a large memory model size uses significant system resources and can potentially impact other processes running on the system. Consult with the other users before changing the allocated size.

Note:	Memory size changes do not take effect until you restart the Adams product.

Memory Model Sizes

The following table shows the variable and array sizes for the memory options.

Variable	MDSIZE	OSSIZ	CDSIZ	LDSIZE	LCEXPR	FPSIZE	NINSIZ	MXSTAK
Array	MD	OPRM	CD	LDSIZE	CEXPR	FPVARS	NINSTR	MSTAK1
Standard	1100000	100000	4000	2000	100	250	1000	50
Large	3000000	100000	4000	2000	200	250	1000	100
Extra Large	20000000	100000	8000	5000	300	1000	5000	300
Huge	60000000	200000	16000	50000	1000	3000	15000	300

The following table explains each variable and array.

This array or variable:	Is:
MD array	Main storage array for Adams Solver analysis.
OPRM array	The memory section for interactive modification of function expressions, user-written subroutines, arrays, splines, and more. This array is specified as a subset of the MD Array, and does not increase the size of the MD Array.
CD array	Used to store column titles and spline types.
LD array	Used to store character strings, request comments, function expressions, and more.
LCEXPR	The maximum number of lines in any function expression.

This array or variable:	Is:
FPSIZE	The maximum number of real constants in any single expression. Real constants are decimal place numbers that cannot be represented as integers; 2.1 for example but not 2.0.
NINSIZ	The maximum number of RPN (Reverse Polish Notation) instructions in all function expressions.
MXSTAK	The maximum size of the RPN stack.

Viewing Memory Model Size from Adams View

From within Adams View, you can use a system command to determine what your environment settings are used for memory model size.

To display the memory model size:

1. Open Adams View.
2. From the **Tools** menu, select **System Command**.
3. Perform one of the following:
    - On Linux: In the **Command Text box**, enter the following Linux command. See your operating system documentation for more information.

      printenv|grep MEM
    - On Windows: In the **Command Text box**, enter **set.**
4. Select to write the output to the information window.
5. Select **OK**.

   Adams View displays your environment settings in the information window. For example, you may see the following:

```
MDI_AVIEW_MEMSIZE=LARGE
MDI_SOLVER_MEMSIZE=HUGE
```

Setting Custom Memory Model Size

Custom Memory Models

You must have a FORTRAN compiler to create a custom memory model.

MSC Software provides several standard memory models for you to use as explained in Memory Model Sizes. Often, when you are simulating very large models, however, you receive the following message:

```
! ERROR: ***** ADAMS MEMORY OVERFLOW *****
! A larger MD array is required.
! The error occurred in subroutine "OPMAIN" while allocating "DOUB" space
! from "TEMP" for the "IZ".
! Space Available: 348496 Space Requested: 7651872
```

In the message, **Space Available** is the amount of space that is left to run with the current memory model. **Space Requested** is the space that is needed for the simulation to continue. Therefore, the space needed to run the simulation is:

```
Space Requested - Space Available + Current Memory Model
```

For example, if you are running with the memory model set to HUGE and you get the following error message:

```
! ERROR: ***** ADAMS MEMORY OVERFLOW *****
! A larger MD array is required.
! The error occurred in subroutine "OPMAIN" while allocating "DOUB" space
! from "TEMP" for the "IZ".
! Space Available: 348496 Space Requested: 7651872
```

The amount of memory necessary to run the simulation is:

7651872-348496 + 20000000 = 27.4e6

If you already have your memory model set to HUGE, you need to create a custom memory model to accommodate the large memory requirements.

Remember that selecting a memory size affects the machine and every user. Consult with other users before decreasing the allocated size.

On Windows, by default, Adams places the custom memory model in the directory install_dir/win64/uconfg_user. When Adams starts, it searches the following locations in the following order for a custom memory model:

- The location where the user executable is found, if used.
- install_dir/win64/uconfg_user directory.

On Linux, when you create a custom memory model, the Adams Toolbar saves the parameters in the Adams registry, and creates a FORTRAN subroutine for the platform on which you are currently running the Adams Toolbar. The Adams Toolbar uses a directory structure under $HOME/.msca/cmm to store the FORTRAN subroutines and corresponding libraries. Do not change the contents or structure of the memory model directory.

Maintaining Custom Memory Models on Linux

This topic explains how to maintain your custom memory models. Select one of the following topics:

- Creating a custom memory model
- Deleting a memory model
- Rebuilding a memory model
- Changing a memory model

Creating a custom memory model

To create a memory model:

1. Right-click the **Adams Toolbar** tool, point to **Manage Custom Memory**, and then select **Create Memory Model**.
2. Click **name**, and enter a name for the memory model.
3. Adjust the array values in the options. Refer to the tables in Memory Model Sizes for explanations of each of the array values.
4. Select **OK** to build the necessary directories, source file, and library.

   A window displays a list of build parameters as the library creation takes place.

Deleting a memory model

To delete a memory model:

1. Right-click the **Adams Toolbar** tool, point to **Manage Custom Memory**, and then select **Delete Memory Model**.
2. In the treeview, click the name of the memory model that you want to delete.
3. Select **OK**.

Note:	This deletes the model parameters in the registry, the model directory, and all libraries.

Rebuilding a memory model

Occasionally, you have to rebuild your custom memory models. For example, you may have to rebuild if you switch to a new architecture or if something goes wrong with your current model directory.

To rebuild a memory model:

- Right-click the **Adams Toolbar** tool, point to **Manage Custom Memory**, and then select **Rebuild Memory Models**.

  A shell window displays each custom memory model being rebuilt as needed.

Changing a memory model

To change a memory model:

1. Right-click the **Adams Toolbar** tool, point to **Manage Custom Memory**, and then select either **View/Change Memory Model**.
2. In the treeview, click the name of the memory model that you want to change.
3. Adjust the array values in the options. Refer to the tables in Memory Model Sizes for explanations of each of the array values.

4. Select **OK**.

   A shell window displays each custom memory model being rebuilt as needed.

Setting Custom Memory Model Size on Linux

To set a custom memory size on Linux:

1. The '**cmm**' is a non-visible option in the start script on Linux.
2. Syntax: /top_dir/mdi -c cmm

   Update the 8 values when prompted, and create a name for the memory model. Help is available by specifying ' -h' (note that, this help argument cannot be used on Windows, it is only available on Linux).
3. mdi -c cmm -h

   This routine will create a custom memory model library.
4. The order of the inputs are:

   MDSIZ OSSIZ CDSIZ LDSIZ LCEXPR FPSIZE NINSIZ MXSTAK model_name

   For example,

   cmm 60000001 200002 16003 50004 1005 3006 15007 308 name1

   If less than the full list of variables are input, then defaults will be used, for example to update CDSIZ:

   cmm 60000000 200000 16001

Setting Custom Memory Model Size on Windows

To set a custom memory size on Windows:

1. From the **Start** menu, point to **Programs**, point to **Adams *x*** (where *x* is the release number, for example **Adams 2021.3**), and then select **Settings & License**.

   The Adams registry editor appears.
2. Set the memory model size for both Adams View and Adams Solver to Custom.
3. Select **OK**.
4. Run **adams2021_3 cmm** and select your uconfg parameter sizes.

   A new *uconfg_user.dll* is created in the *install_dir/win64/uconfg_user* directory.

   Your memory model size is set.

   Set the custom memory setting as the default as explained in Adams Environment.

Applying Advanced Settings

Environment Variables

Adams contains many advanced settings. Some are common to all Adams products, and others apply specifically to one product. In the past, these have commonly been referred to as “environment variables”. For those that influence the solving of Adams models they can be set via the ENVIRONMENT statement in the .adm file or via “Solver Settings - Advanced” in the Adams View and Adams Car interfaces. These are termed “Advanced Settings” and are listed in the Adams Solver Advanced Settings section below. For those that do not directly influence Adams Solver execution, they can continue to be set as environment variables. Using the system registry, you can change these variables to further customize your use of Adams. You can also set these environment variables for your current executable only via the PUTENV function.

Adams Controls Environment Variables

The following are Adams Controls environment variables:

Variable name:	Variable value:	What it does:
ADAMS_CONTROLS_WTIME	An integer number such as 5 or 10.	Sets the total wait time (that is, until time-out) in start-up to create a pipe for a pipes-based co-simulation or function evaluation when using Adams Controls on Windows. Controls will test for the pipe every half second and may finish before the end time. Options are integers. Default = 60 (seconds).
ADAMS_FMI_LOG	An integer number such as 1 for on. To switch off the functionality, unset the environment variable.	When Adams is acting as FMI parent, setting this environment variable would allow logging of any FMI messages into a file called fmilog.txt which would be available in the current working directory of Adams
ADAMS_PATH_FOR_FMU	A string pointing to the path of the Adams installation folder which is to be used by the Adams FMU	If the user has multiple Adams installations or has exported the FMU using one version of Adams and wants to run it in another version of Adams, this environment variable has to be specified so that it forces the FMU to use that version of Adams. Example of how the environment variable is set is given below. `ADAMS_PATH_FOR_FMU= C:\Program Files\MSC.Software\Adams\2021_3`

Variable name:	Variable value:	What it does:
DEBUG_CONTROLS	An integer number such as 1 for on. To switch off the functionality, unset the environment variable.	During a co-simulation or function evaluation run, this will create two files (one for the Adams server side, and one for the client) in the working directory with information about the data passed between client and server.
MSC_ACONTROLS_DISCRETE_STATES	An integer number such as 1 for on. To switch off the functionality, unset the environment variable.	When unset, only a single dummy discrete state is displayed for the GSE. When set, this will report all discrete states found in the External System Library. Default = unset.
MSC_ADAMS_COSIM_SIM_DYN	An integer number such as 1 for on. To switch off the functionality, unset the environment variable.	Adams Controls will run as a standard dynamic analysis instead of an event based analysis when an Adams Car model is used as a plant model.
MSC_ADAMS_SKIP_FMUCHECK	An integer number such as 1 for on. To switch off the functionality, unset the environment variable.	When importing an FMU (Adams Controls System Import), CheckFMU is executed to ensure the FMU is valid prior to creating the GSE for the External System Library. Setting this environment variable also you to skip CheckFMU.
SHOW_CMD_WINDOW	An integer number such as 1 for on. To switch off the functionality, unset the environment variable.	When Adams is acting as a child during a parent child co-simulation, setting this environment variable allows the command window to be shown. This is helpful during debugging to find out any error messages being printed by the child. Note that this could slow down the co-simulation under ordinary circumstances due to the potentially large volume of messages that could be printed here and hence should be enabled only for debugging.

Adams Durability Environment Variables

The following are Adams Durability environment variables:

Variable name:	Variable value:	What it does:
DUR_HOTSPOT_LIMIT	Integer	Sets an upper limit on the number of hot spots to report. Default is 200. **Note:** This environment variable is applicable only when you select the Adams Durability hot spot type as "**threshold**" but not "**count**".
DUR_MSR_MB_LIMIT	Real	Sets the maximum swap space (in megabytes) when performing a modal stress recovery (MSR). Default is 400 MB.
MDI_ADAMS_EQUALTIMES	**keep** (default) - Keep all time steps **first** - Take the value of the first equal time step **last** - Take the value of the last equal time step	Writing of DAC files can be problematic if multiple solutions exist at the same time. These extra (sometimes unwanted) time blocks are created when processing certain simulation commands. This environment variable removes equal time steps.
MDI_DUR_DEFNODE		Recovers data for all nodes in the flexible body if no nodes or hot spots are specified in the FEMDATA statement.
MDI_DUR_LOADHEADER		Prints header information to the loadmap file.

Variable name:	Variable value:	What it does:
MDI_DUR_LOADMAP		Generates an ASCII file that maps the loads from MBD to FE for DAC output only. The name of this file is <*job_name*>.map, where <*job_name*> is the name given on the FILE argument of the first FEMDATA statement.
MDI_DUR_LOADOFFSET	Integer	Creates a gap in the sequential naming of DAC channel numbers. For example, without an offset set, DAC files start with the names: job_name_001.dac job_name_002.dac If MDI_DUR_LOADOFFSET is set to 300, the DAC file names start with: job_name_301.dac job_name_302.dac
MSC_DUR_NUM_REPEATS	Integer	Specifies the required number of repeats to the RPC III file. Default is 0.

Adams Flex Environment Variables

The following are Adams Flex environment variables:

Variable name:	Variable value:	What it does:
MDI_FLEX_BODY_TESTING_OPTIONS	**gamma**: Scale factor; useful range is 0 to 1.0 **max**: When step size changes, cratio changes by max 1% **cratio**: Initial value of cratio across all modes **debug**: Only applicable when using Adams Solver (C++)	The variable for auto_damping overrides the traditional flex-body default damping. Variable: MDI_FLEX_BODY_TESTING_OPTIO NS Value: auto_damping(gamma, max_increase, cratio_initial) debug Linux example: setenv MDI_FLEX_BODY_TESTING_OPTIO NS "auto_damping(1.0, 0.01, 1.0)"
MSC_FLEXBODY_MASS_SCALING	**id**: id of a flexbody **f_cutoff**: "cutoff" frequency **f_delta**: frequency increment	All other frequencies above the "cutoff" frequency will be equally spaced using "delta". Speed improvements can range from 5% to 30%. MSC_FLEXBODY_MASS_SCALING= Id, f_cutoff, f_delta, id, f_cutoff, f_delta,...
MNF_MAX_MODE_MEMORY	Integer	The MNF_MAX_MODE_MEMORY setting specifies how much memory *each* MNF can use to keep the modes in memory. Therefore you should set the value based on the "biggest" MNF you're using (that is, whichever one has the highest mode x node) and all of the "smaller" MNFs will be fine. The default value is 200MB. The value should be a number specifying the amount of memory in MB (Mega Bytes). For example, MNF_MAX_MODE_MEMORY=400, indicating 400 MBytes. Again, it should be a number, 400 is good and 400MB is not.

Adams Solver Advanced Settings

The following are Adams Solver advanced settings:

Variable name:	Variable value:	What it does:
ADAMS_SPARSE_SOLVER_SWITCH_AT	Integer	When the linear system solver is set to "auto" (that is, using the command: LSOLVER/AUTO) this environment variable specifies the number of equations (degrees of freedom) above which Adams will switch from the Calahan solver to the UMF solver. Valid for Adams Solver (C++) only. See the LSOLVER command for details about the UMF solver.
MDI_ADAMS_CONTACT_COR		Specifies the corrector to be used for models involving contacts. **Not set:** Modified corrector is used **Set to any non-null value:** Original corrector is used **Note:** INTEGRATOR/CORRECTOR=MODIFIED overrides whatever the value of MDI_ADAMS_CONTACT_COR is. The modified corrector is always invoked with this statement.
MDI_ADAMS_CONTACT_OUT	on/off	Controls the export of intermediate contact events in the data for a contact model. The intermediate data will only be written to **.res* file. **On**: Enables intermittent contact output and also the default value. **Off**: Disables intermittent contact output.
MSC_ADAMS_LINEAR_FILTER_STATE_MATRICES	double > 0.0	Forces the Adams C++ Solver to filter all entries of the linearization matrix. All entries with an absolute value smaller than the set value, are set to zero. Filtering the linearization matrix may be helpful for ill-conditioned linearization matrices.

Variable name:	Variable value:	What it does:
MDI_ADAMS_SHELL_FILE_PATTERN		Exports, as shell files, tessellated geometry representations used during contact. It creates a shell file at the start of the simulation with the naming prefix you specify. **Note:** This variable must be set before the simulation begins. For example: var set var=test string=(PUTENV("MDI_ADAMS_SHELL_FILE_PATTERN","msc")) Will result in files named as follows: msc1.shl msc2.shl msc3.shl
MDI_PRINT_UCO_SIZ	True or False	Used to check the size of memory model you are using. If this is set to **True** then the current memory model size is printed. You can see a confirmation message when Adams Solver starts up, indicating the size of the current memory model. Valid for Adams Solver (FORTRAN) only.
MDI_SOLVER_SELECT	**CXX** or **F77**	Specifies the Adams Solver type to use. **CXX** - Use Adams Solver (C++) **F77** - Use Adams Solver (FORTRAN)
MSC_ADAMS_CONSTRAINT_DELTA	>0	Specifies the length of I-J marker separation before a warning message is issued. Joints such as CONVEL, FIXED, HOOKE, REVOLUTE, SPHERICAL and UNIVERSAL, and Jprim ATPOINT should have their I and J markers coincident during model input. Solver will issue a warning message for each Joint/Jprim that exceeds the specified value. This option can be used to disable excessive warnings when marker non-coincidence is expected. During Initial Condition analysis, Solver will attempt to satisfy all constraints to within the IC displacement ERROR. The default value is 1.0e-03 in model length units. Valid for Adams Solver (C++) only.

Variable name:	Variable value:	What it does:
MSC_ADAMS_CORNER_TREATMENT	> 0	Enables enhanced handling of 3D contact detection in the RAPID Geometry Engine. In certain situations, such as a box touching a wall and floor at the same time, the contact intersection will wrap around the box. Setting this environment variable will help RAPID handle these situations. This handling is **off** by default because it can slow down models that do not encounter these situations.
MSC_ADAMS_DISCRETE_GSE_TEMPORAL_ERROR		Sets the error threshold for when GSE_update is called. If the difference between the time of the next required call (sample time) to GSE_update and the current simulation time is within this error, then GSE_update will be called at the current simulation time. If it is not set, the system uses 1.0e-6 time units. Making this error very small will cause numerical integration to become unstable. If you model has a GSE with very high frequency discrete states, a better approach (than setting this environment to a smaller value) is to change the time units of the model to milliseconds, or even smaller time units via UCF (see the UNITS statement)
MSC_ADAMS_FLEXCONTACT_PRECISE_PARTIALS	1	Forces the C++ Solver to compute high-precision partial derivatives for Flex Body Contact. By default, the C++ Solver computes approximate partial derivatives for Flex Body Contact to improve performance. This variable can help models that run poorly. In some special cases, it can even improve performance because accurate partials allow the integrator to take larger time steps. Valid for Adams Solver (C++) only.
MSC_ADAMS_FLEXCONTACT_USE_MODAL_VELOCITY	1	Specifies that the time derivative of the Flex Body modal amplitudes are used in computing the damping for Flex Body Contact. The default is to not use the modal velocities when computing the damping because the number of states for which partial derivatives must be computed is increased. However, there may be cases where it is advantageous (or even necessary) to include the modal velocities. Valid for Adams Solver (C++) only.

Variable name:	Variable value:	What it does:
MSC_ADAMS_HHT_FORCED_JACOBIAN_HMAX_THRESHOLD	double	Enforces Adams C++ Solver to force the Jacobian evaluation every output step in HHT (PAT=F) if DTOUT or HMAX is smaller than the set threshold. The default value of this threshold is 5.0E-5. Users can set the value to 0 to turn off this feature. **Note:** This feature is introduced in Adams 2021 to improve the accuracy of HHT for simulations with DTOUT or HMAX <= 5.0 E-5. Users should use this environment variable if the HHT performance is adversely affected due to this feature for simulations with DTOUT or HMAX <= 5.0 E-5.
MSC_ADAMS_NO_INFO_ON_SHELL	True or False	It will turn off all the message on shell. However, Adams will still write the message into the .msg file. By doing so, models with sensors trigger at each steps and models with debug/eprint on will run much faster than before.
MSC_ADAMS_NODE_PENETRATION_THRESHOLD	>0	Specifies the maximum penetration (in millimeters) of a node in Flex Body Contact. Setting this variable overrides the default value (which is proportional to the intersection volume) used by the C++ Solver. This variable is most useful with open shell geometry, where it is difficult for the Solver to compute an accurate intersection volume. Valid for Adams Solver (C++) only.
MSC_ADAMS_NODE_THRESHOLD	>0	Specifies the minimum number of nodes that can be in a single incident for Flex Body Contact. Setting this variable overrides the default value (which is three) used by the C++ Solver. If the number of nodes found is less than the value set, then no contact force will be generated for that incident. Therefore, this variable should be used with caution. This variable is most useful in situations where it is known in advance that there will be fewer than three nodes in a contact incident. Valid for Adams Solver (C++) only.

Variable name:	Variable value:	What it does:
MSC_ADAMS_NOLIST_SERIAL		By default, at the start of a dynamic simulation, when using the NTHREADS option for parallel execution, Adams Solver (C++) prints a list of all modeling elements that are not thread safe. The said list shows all elements that will be evaluated serially. Set this environment variable to avoid printing the list. Valid for Adams Solver (C++) only.
MSC_ADAMS_PARALLEL_CALAHAN	1	Forces the solver to use a multithreaded linear algebra code. It only helps large models (approx. 6000 equations or more) or models that have dense Jacobians (models with flexible bodies and modal forces). Speed improvements can range from 5% to 50%. This environment variable is deprecated in version 2019.
MSC_ADAMS_REAL_TIME	The value for this environment variable is the same as the value you may enter for the Adams FMU parameter msc_adams_realtime. See section Options in the msc_adams_realtime parameter.	Configures the real-time simulation allowing changes in the default behavior, modify integration parameters, control output generation and so on. See section Options in the msc_adams_realtime parameter.
ADAMS_THREAD_AFFINITY_SET0	String values that contain numbers (indicating cores/processor)	Specifies the processor affinity for set0 threads as comma separated individual values or a range. For example, if five threads are to run on cores 1,2,3,5,7 this can be specified as 1-3,5,7. See section Thread Affinity Settings in Adams for more information.
ADAMS_THREAD_AFFINITY_SET1	String values that contain numbers (indicating cores/processors) separated by - or ,	Specifies the processor affinity for set1 threads as comma separated individual values or a range. Note that this parameter is currently not being utilized by Adams Solver in real time.
MSC_ADAMS_SENSOR_SLEEP_COUNT	Integer bigger than zero	By default, Sensors sleep 3 time steps after triggering. During the sleep period, Adams Solver C++ will not query the Sensor that just triggered. However, in real time simulations with mini maneuvers shorter than 3 times the current simulation step, Adams Sensor fails to detect the end of the mini maneuvers. Setting the sleep count to 1 will force the Sensor to be queried at every time step.

Variable name:	Variable value:	What it does:
MSC_ADAMS_SERIAL_CALAHAN		Starting version 2017.2, Adams Solver (C++) uses a second generation parallel version of the Calahan linear algebra solver using both pthreads and OpenMP technologies. Setting this environment variable enforces Adams Solver (C++) to use a serial version of the linear algebra code.
MSC_ADAMS_SOLVER_AXAYAZ_INIT	P or N	If the value of AX or AY or AZ measure is close to a bifurcation point, that is, 180 degrees (+PI) or -180 degrees (-PI), the value will be set to +PI or -PI depending on the variable value. (P ≡ Positive and N ≡ Negative)
MSC_ADAMS_SOLVER_FLEXBODY_BOOST_THRESHOLD	Integer bigger than zero	Changes the default threshold value that triggers the use of a flexible body boost performance code. By default the threshold is 40. See MSC_ADAMS_SOLVER_NO_FLEXBODY_BOOST to disable the usage of the boost performance code.
MSC_ADAMS_SOLVER_INTERPOLATE_ADAPTIVE	1	Forces the solver to use an experimental interpolation technique. Using this option, the Integrator Output Step is simply a reference. The integrator may decide to skip output blocks if the results are smooth. Output steps are not equally spaced. This option should not be used if the model has the discrete GSEs. Speed improvements can be up to a factor of 20 for some models. Available for GSTIF and HHT.
MSC_ADAMS_SOLVER_NO_FLEXBODY_BOOST		By default, when a flexible body has more active modes than a default threshold value of 40, a special code is used to boost performance of the simulations. Setting this environment variable will disable the use of the boost performance code. See MSC_ADAMS_SOLVER_FLEXBODY_BOOST_THRESHOLD to change the default threshold value.

Variable name:	Variable value:	What it does:
MSC_ADAMS_SOLVER_NODAMP_DIFF		When using both the EIGENSOL and NODAMPIN options in a linearization analysis, the velocity terms in the linearization matrix are flushed to zero by default. However, if there are coupling terms between velocity states and differential elements, that coupling is not considered damping and it is left untouched. Setting this environment variable enforces Adams Solver (C++) to flush the coupling terms between velocity and differential states to zero. This option was created to match the behavior of the Adams Solver (FORTRAN).
MSC_ADAMS_SOLVER_USE_DIFFS		Enforce Adams solver to use additional DIFF objects when processing holonomic velocity and acceleration constraints (velocity/acceleration MOTION) which is what F77 does. Setting this variable will help when velocity or acceleration constraints introduce high frequencies. However, you will lose the ability to perform a kinematic analysis on a zero degree-of-freedom model containing motions with velocity/acceleration arguments as described in the C++ motion statement.

MSC_ADAMS_SENSOR_MESSAGES	ON or OFF, ALL or a list of Adams IDs (id1, id2,..., id3-id4,...)	SENSORs in the model may generate large amounts of messages. To disable or enable messages from SENSORs set the value of this environment variable to words ON, OFF, ALL and a list of Adams IDs. For example:
		
		`MSC_ADAMS_SENSOR_MESSAGES=OFF, 4, 7, 11-20`
		will disable sending messages from SENSORs with Adams IDs provided in the list.
		
		`MSC_ADAMS_SENSOR_MESSAGES=OFF, ALL`
		will disable sending messages from all SENSORs in the model.
		
		`MSC_ADAMS_SENSOR_MESSAGES=ON, 3`
		will enable sending messages from SENSOR/3.

Variable name:	Variable value:	What it does:
MSC_ADAMS_USE_MOMENTA	1	Enables the use of Angular Momenta states for Rigid Bodies. By default Rigid Bodies have 12 states (X, Y, Z, PSI, THETA, PHI, VX, VY, VZ, WX, WY, WZ). The angular momenta states are angular momentum about X, Y and Z axes. These additional states can benefit Rigid Bodies that undergo rapid rotation about multiple axes. Valid for Adams Solver (C++) only.
MSC_ADAMS_USE_PARDISO_STATICS	Optional integer bigger than zero	Enforce Adams solver to use the Intel MKL Pardiso linear solver during statics simulations only. The optional value is the number of iterative refinements used when solving the linear set of equations. The default value is zero (no iterative refinements).
MSC_ADAMS_USER_PARTIAL_METHOD	ANALYTICAL, NUMERICAL or ADAPTIVE	Specifies which method the C++ Solver uses to compute partial derivatives for user-written subroutines. User-written subroutines establish dependencies on Solver states by calling SYSFNC and/or SYSARY. The NUMERICAL method computes partial derivatives by directly perturbing the states and evaluating the user subroutine with DFLAG=true. The ANALYTICAL method perturbs the measures (for example, DISP, VM, VX and so on) that the user subroutine depends on, evaluates the user subroutine, and then uses the chain rule to compute the partial derivative. This usually gives superior partials. The ADAPTIVE method (which is the default) uses an internal algorithm to decide whether the ANALYTICAL or NUMERICAL method is more efficient for a given subroutine. Valid for Adams Solver (C++) only.

Variable name:	Variable value:	What it does:
MSC_IFORT_DIR and MSC_ICC_DIR		Adams assumes the Intel compilers are installed under */opt/intel*. Use this variables to tell Adams the compilers location when they are installed in a non-standard location. When set Adams looks for: ▪ *ifort* under $MSC_IFORT_DIR/bin ▪ *icc* under $MSC_ICC_DIR/bin. **Note:** These environment variables only applies for the Linux platforms.
MSC_RAPID_SKIP_SHELL_VERIFICATION	1	Disables checking shell geometry for holes (water tight check). Contact geometry read from shell files (.shl) is checked for holes. This check can be time consuming, if the number of triangles is large (more than 100000). For cases where it is known that shell geometry does not contain holes this variable can be used to disable checking, which improves performance.
PRINT_IC_RESULTS	ALWAYS or NEVER	Enables the output of IC results in *.msg* file for dynamic analysis. Valid for Adams Solver (C++) only. **Note:** The IC results are not written when using a kinematic analysis, therefore you should use SIM/DYNAMIC instead of SIM/TRANSIENT.

Adams Machinery Environment Variables

The following are Adams Machinery environment variables:

Variable name:	Variable value:	What it does:
AMACH_BEARING_DUTY_CYCLE_ORIGINAL	1	This variable if set, will allow you to revert the duty-cycle life calculation method. $$\frac{1}{L_{tot}} = \sum_{i=1}^{N} \frac{(t_i \times r_i)}{L_i}$$ Where: ■ L_{tot} : single-value service life estimate provided for each bearing by this "Adams Machinery Bearing Duty-Cycle Service Life" tool. ■ t_i : the time-share of the given output step's life prediction value. This amounts to the fraction of the total time range covered by all rows of the duty cycle for which the given output step accounts. For example, if a 1 second simulation of 10 steps is run and the entire simulation is covered by the duty cycle specification, then t_i for each output step would be 0.1. ■ r_i : the number of repeats applied to the given output step as defined in the duty cycle specification. ■ L_i : the given output step's life prediction value.

Adams Vibration Environment Variables

The following are Adams Vibration environment variables:

Variable name:	Variable value:	What it does:
ADAMS_VIBRATION_MIN_FLEX_PARTICIPATION		Same as ADAMS_VIBRATION_MIN_PARTICIPATION, except valid for flex modes.
ADAMS_VIBRATION_MIN_PARTICIPATION		Currently, during a vibration animation of flexible bodies, Adams Vibration finds the largest modal coordinate and ignores any other coordinates that are less than 10% of that value. This environment variable changes the percentage that is ignored. Use this variable for system modes. (Use ADAMS_VIBRATION_MIN_FLEX_PARTICIPATION for flex modes.) To include all system modes in the animation, set this environment variable to 0.0. For example, to ignore all modal coordinates that are less than 5% of the largest coordinate, on Linux enter: **setenv ADAMS_VIBRATION_MIN_PARTICIPATION 0.05**

Adams View Environment Variables

The following are Adams View environment variables:

Variable name:	Variable value:	What it does:

ADAMS_GUI_LOCALE = english
 	
or
 	
ADAMS_GUI_LOCALE = japanese
 	
or
 	
ADAMS_GUI_LOCALE = chinese
		Setting the language. Default locale is english. If set to japanese or chinese, the default font used for the Windows platforms is Arial Unicode MS, and for Linux platforms it is Helvetica [Sazanami].

MDI_AVIEW_BITMAPS MDI_AUX_BITMAPS		Search path(s) for bitmap files. Entries are separated with a colon (:) on Linux and semicolon (;) on Windows.
MDI_AVIEW_DDE		Windows only When set, makes available several language functions used for DDE.
MDI_AVIEW_EXPERIMENTAL		When set, experimental code becomes functional. Contact Technical Support for more information.
MDI_COURIERFONT_SIZE		Size of text in multi-line fields and selection lists, specified in pixels.
MDI_GUI_FONT_FAMILY	Any font available on your system.	Changes the default font used in Adams products. Examples include Rockwell, Arial, and Bookman.
MDI_GUI_FONT_SIZE	An integer number such as 11 or 12.	Changes the default font size used in Adams products. Setting this may cause text on certain dialog boxes not to be displayed, so it is recommended to use values such as 10, 11, or 12.

Variable name:	Variable value:	What it does:
MDI_IGNORE_CONNECTIONS_WARNING	1	Suppresses the popup prompt asking if the user would like to continue with the conversion of a shell body geometry type to Parasolid despite the fact that it may take a long time due to the amount of data defining the shell geometry. This can be particularly helpful when exporting an Adams View command file (.cmd) with the option "Use Parasolid" = "Full" for a model with multiple geometries that are such large shell bodies. Suppressing the warnings will allow the user to walk away from the machine while the lengthy conversion process is underway and not fear that the .cmd export process will keep pausing awaiting for user input about each and every large shell.
MDI_MOTIF_STYLE		**X11 only** If set, runs the interface in native look and feel (usually Motif on SGI, the native SGI style).
MDI_MENUBAR_FONTSIZE		Size of text in menubar, specified in pixels.
MDI_MENUITEM_FONTSIZE		Size of text in menus, specified in pixels.
MDI_NO_SHELL_OPT		Turn off the internal shell optimization code by setting to any value.
MDI_ONLINE_BROWSER		Specifies the path to the browser to use for online help on Linux and windows platforms. Do not add the path in string quote.
MDI_P_SCHEMA		Specifies the path for the Parasolid Schema files.
MDI_PARASOLID_IMPORT		For solid bodies in parasolid files, Adams View tessellates them to create a shell. By default Adams View will try to reduce the number of connections if some points are very close to each other. But, for very large parasolid files, this process can take a noticeable amount of time. Setting the variable MDI_PARASOLID_IMPORT disables this process and might improve import speeds of parasolid files.
MDI_PROGRESS_METER		If set, enables the progress meter dialog box when needed. If not set, the dialog will always be hidden.
MDI_SHOW_NOWARN		Set to any value other than null to stop displaying warning messages.

Variable name:	Variable value:	What it does:
MDI_STEREO	0 or 1	If set, enables stereo viewing.
MDI_USE_DESKTOP_SETTINGS		If set to **0** or **disabled**, Adams View no longer tries to use colors and fonts that can be defined by the user under the **Appearance** tab in the Windows **Display Properties** dialog. This will help Adams View not to freeze for minutes on Windows, after such events as using a screen saver. There are two possible causes for Adams View to freeze occasionally: either a device driver (video cards are the usual suspects) causes this problem, or Windows sends out lots of unexpected events that keep Adams busy. Typically, the CPU usage is 0 in the first case, while the aview process hogs one CPU completely in the latter case. That CPU usage is caused by an unexpectedly high number of a certain system event that is repeatedly sent by Windows.
MDI_WARN_RECURSIVE		Turns on dump of expressions in which recursion has been detected. This is a debug aid for very advanced users.
MDI_X11_FONT_PATTERN		X11 only Allows you to specify the general interface font with an X11 font pattern.

Variable name:	Variable value:	What it does:
MSC_FLATTEN_ADM	0 or 1	Specifies the level of hierarchy in the adams_view_name tag. Currently, all datasets are written with full hierarchical names in the adams_view_name tags. This allows XML results files created with an external solver to be read in while retaining the same hierarchy as the model. This benefits applications that use model hierarchy (UDEs, submodels, and so on) and simulate with the external solver. To restore the previous behavior, set the environment variable to 1.
MSC_HELP_WIN_OPEN		Opens a new browser window on Linux. Set to **new** (for Netscape) or **newtab** (for Mozilla).
MSC_RESULTS_DOM	1	Specifies that the DOM xml result reader is used whenever XML result files are read into Adams. Absence of this environment variable will cause the SAX result file reader to be invoked. The SAX result file reader reduces the memory footprint and the time required to read XML result files. Note that the end result of using both the DOM and the SAX parsers is the same, the only difference will be the time required for reading in the files and memory consumed for the reading process.

Note:	Some environment variables (MDI_PROGRESS_METER, for example) are only considered set if their value is either **1** or **enabled** (without quotes, case insensitive).

Template-Based Environment Variables

The following environment variables are used in various Adams template-based products.

Variable name:	Variable value:	What it does:
MDI_ACAR_ANL_LOG1 MDI_ACAR_ANL_LOG2 MDI_ACAR_ANL_LOG3	Component name. For example: hardpoints and parts. See a list of valid variable values.	Writes additional information, about the components you specify, to the analysis log file. You can select up to three components to correspond to log1, log2, and log3.
MDI_ACAR_INITMODE	template_builder/standard_interface	If you have expert-user access, sets the default selection in the Welcome dialog box.
MDI_ACAR_LOGO_BMP	Path to the image you want to use.	Replaces the image on the Welcome and Exit dialog boxes. Image size should be 192x192 pixels.
MDI_ACAR_MODEPROMPT	yes/no	Sets your template-based product so that it does not display the Welcome dialog box at start up.
MDI_ACAR_PLUS_AVIEW	yes/no	Gives you access to Adams View.
MDI_ACAR_SIDE_PREF	right/left	Defines the preferred side for creating/modifying symmetric components. The default is left.
MDI_ACAR_USERMODE	expert/standard	Sets the user mode.
MDI_ACAR_USE_EDITOR	any value	Opens external files in a text editor. If this environment variable is not set, external files will be shown in the info window. If set to any value, ***notepad.exe*** is used to open any file. The default editor can be changed by specifying the path via the Adams Settings utility (**Adams registry editor** → **AView** → **Preferences** → **textEditor**). For example: `C:/Program Files/Windows NT/Accessories/wordpad.exe`.
MDI_ACAR_VEHICLE_LEFT	Direction cosines. For example: 0,-1,0.	Sets the orientation of the global reference frame using direction cosines.
MDI_ACAR_VEHICLE_REAR	Direction cosines. For example: 1,0,0.	Sets the orientation of the global reference frame using direction cosines.

Variable name:	Variable value:	What it does:
MDI_ACAR_WRITE_OUT	on/off	Set the print option of the output command in Adams Car. If set to '**off**', the resulting *.adm* should contain OUTPUT/NOPRINT which prevents request tables to be written to the *.out* file. **Note:** The print option does not work with the `common_init`.
MDI_ACAR_WRITE_RES	yes/no	Outputs an analysis results file. If the model includes a flexible body, your template-based product automatically outputs an analysis results file.
MDI_ACAR_WRITE_RPC	yes/no	Outputs RPC III file(s) following analysis. Use the Request Map Editor to control the filename and file contents.
MDI_ACAR_XRF	on/off	Toggles results file being written in xrf format. The default is off.
MDI_AENG_HHT_ERROR	Any positive number	Sets the HHT integrator as the default integrator, with a default error tolerance of the value specified.
MDI_CDB_EXPAND_PATH	yes/no	Writes the expanded file names to the adm deck.
MDI_CDB_SEARCH	yes/no	Restores version 12.0 CDB search algorithm. Use is discouraged.
MDI_<product name>_PRIVATE		
MDI_ACAR_PRIVATE_CFG+ MDI_ADRV_PRIVATE_CFG+ MDI_AENG_PRIVATE_CFG* MDI_RAIL_PRIVATE_CFG* MDI_AIR_PRIVATE_CFG*		Defines the path to your private configuration file.
MDI_ACAR_PRIVATE_DIR+ MDI_ADRV_PRIVATE_DIR+ MDI_AENG_PRIVATE_DIR* MDI_RAIL_PRIVATE_DIR* MDI_AIR_PRIVATE_DIR*		Defines the path to the private repository.
MDI_ACAR_SITE+ MDI_ADRIVELINE_SITE+ MDI_AENGINE_SITE* MDI_ARAIL_SITE* MDI_AIRCRAFT_SITE*		Defines the path to the site repository.

Variable name:	Variable value:	What it does:
MDI_UPDATE_CFG	yes/no	This environment variable tells Adams Car whether the private configuration file should be automatically updated during the session to reflect CDB Database changes. This auto-update of the private configuration file is also referred to as the 'live' configuration file. Adams Car defaults to the 'live' private configuration file, and the MDI_UPDATE_CFG environment variable allows a user to turn off the auto-update feature, essentially returning to the v2003 release behavior. To disable the auto-update of the private configuration file, simply add the following line into your private configuration file: ENVIRONMENT MDI_UPDATE_CFG no
MSC_ADAMS_VDM_DEBUG	0,1,2,3	Causes additional diagnostic information during full vehicle simulations. -- 0- No additional output. 1 and 2 - Additional output in *.sdf* and *.sdl* files. 3 - Same as 1, plus _runTime.log file with extensive driving machine diagnostics. This setting produces a large output file and slows execution speed.
MSC_ADAMS_VDM_SI2FLAG	0,1,2,3	Allows the user to override default adaptations of the steering actuator when used with certain integrators. 0 - No adaptations (default for I3 integrator.) 1 - Switch motions to forces in closed loop situations (default for HHT integrator.) 2 - Apply functions directly to motions in open loop situations. 3 - Both adaptations (default for SI2 and SI1 integrators.

Variable name:	Variable value:	What it does:
MSC_SD_DEFAULT_XML		You can use this environment variable to set the path to a custom SmartDriver Setting Template file (*.smartdriver.xml*). The SmartDriver Setting Template file (basically an *.xml* event file without events) is used to set default values for Event files. Those defaults are used when: ■ converting *.dcf* format Event files to .xml format Event files. ■ creating new Event files using the Event Builder. If `MSC_SD_DEFAULT_XML` is not set, the SmartDriver Setting Template file path search order is as follows: 1. Home directory (**$HOME*.smartdriver.xml***) **2. <Adams Installation>\\<platform> ($topdir\\$MDI_CPU\*.smartdriver.xml***)
MSC_ACAR_LIST_COMMENTS	yes/no	If this environment variable is set to "**NO**", the assembly and subsystem files will not include comment lines in the parameter variable and hardpoint tables.
MSC_ACAR_ERROR_ON_ MISSING_NODE_ID	yes/no	If this environment variable is set to "yes", reading a subsystem will be aborted if a flexible body node ID specified in the subsystem cannot be found. If this variable is not set, or set to "no", and a flexible body node ID specified in the subsystem cannot be found, a warning will be issued and the closest node will be used instead.

*MSC Partner products.

$^{+}$Superseded by the Adams registry

Note:	The environment variables marked with a '+' must be set in the Adams registry. Environment variables set in the OS (for example, in the Windows registry), or the shell environment prior to starting Adams, will be overwritten by those registry values. Once in Adams, environment variables can be set or changed using the putenv() design time function.

List of Variable Values for MDI_ACAR_ANL_LOG*

assembly	flexible_bodies	general_variables	nsprings	subsystem_all
aforces	friction	hardpoints	parts	switch_parts
bumpstops	gears	interface_parts	parameters	testrig
bushings	general_parameters	model	property_files	user_entities
dampers	general_splines	nrods	reboundstops	wheels

Setting Preferences

Linux Preferences

Some infrequently used configuration options for Adams View and Adams Solver require that you set environment variables by modifying startup scripts, such as your .cshrc file. These options include as setting license sharing, setting default colors for plotting in Adams Solver, and others.

You can also use the Adams toolbar to set preferences for Adams products.

Important:	Memory size changes do not take effect until you restart the Adams product.

Adams Solver License Sharing

When you use Adams View to perform a simulation, it uses an Adams Solver license. Once the simulation is complete, by default, it does not check the license back in for another user. You must exit Adams View before other users can use the Adams Solver license. You can, however, set a simulation preference, hold_solver_license. You can change the simulation preference using the Adams View command:

```
simulation set hold_solver_license=yes/no
```

- If you set hold_solver_license to yes, then Adams View checks out the necessary licenses when you perform a model verify operation (because of the degrees of freedom calculation, which uses Adams Solver) or any type of simulation using the internal, or integrated, Adams Solver. It only releases the licenses when you exit Adams View or when you run a simulation using the external Adams Solver.
- If you set hold_solver_license to no, Adams View releases all Adams Solver licenses (static, kinematic, and dynamic), and all module licenses (Adams Tire) in these cases:
  - You run a simulation using the external Adams Solver (as before).
  - After a model verify operation.
  - When you reset after a single simulation (transient, static, and so on) using the integrated Adams Solver.
  - After a design study, design of experiment, or optimization analysis (licenses are held throughout the parametric analysis).

To change the default for only one Adams View session, use the Command Navigator to enter the commands **simulation set.** A dialog box appears in which you can temporarily set the default.

Adams Solver Preferences

You can use environment variables to do the following for Adams Solver:

- Modifying the Default Colors in Adams Solver
- Setting Remote X11 Access to MSC Software Applications

Default Colors in Adams Solver

You can modify the default colors for interactive graphics in Adams Solver by setting the values of the environment variables, shown below, in a startup script, such as .cshrc. You can set the screen so that it is monochrome or you can set the colors of background, line, text, and plots.

The following is an example of the Adams Solver color settings:

```
# Modify these variables to change the default colors for Adams Solver
#
setenv ADAMSPP_SCREEN_RENDITION       "TYPE"
setenv ADAMSPP_BACKGROUND_COLOR       "BLACK"
setenv ADAMSPP_LINE_COLOR             "GREEN"
setenv ADAMSPP_TEXT_COLOR             "WHITE"
setenv ADAMSPP_PLOT_GRID_COLOR        "ORANGE YELLOW"
setenv ADAMSPP_PLOT_CURVE1_COLOR      "CYAN"
setenv ADAMSPP_PLOT_CURVE2_COLOR      "GREEN"
setenv ADAMSPP_PLOT_CURVE3_COLOR      "BLUE"
setenv ADAMSPP_PLOT_CURVE4_COLOR      "YELLOW"
```

Each command includes the instruction, the name of a variable, and the name of the color or type for the preceding variable.

Setting Monochrome Display

If you have a monochrome screen or if you want the screen image to appear in monochrome, you can set the following environment variable:

```
setenv ADAMSPP_SCREEN_RENDITION    TYPE
```

In the environment variable, you can replace *TYPE* (in uppercase letters) with COLOR for color or MONOCHROME for a black and white.

Setting this variable to monochrome overrides any settings for color even if you set one or more environment variables for colors in the same Adams Solver session.

Setting Color Display

You can modify default colors for displays and plots by setting environment variables. To change the colors for some or all of the variables, set one or more of the following variables:

```
setenv ADAMSPP_BACKGROUND_COLOR "COLOR NAME"
setenv ADAMSPP_LINE_COLOR "COLOR NAME"
setenv ADAMSPP_TEXT_COLOR "COLOR NAME"
setenv ADAMSPP_PLOT_GRID_COLOR "COLOR NAME"
setenv ADAMSPP_PLOT_CURVE1_COLOR "COLOR NAME"
setenv ADAMSPP_PLOT_CURVE2_COLOR "COLOR NAME"
setenv ADAMSPP_PLOT_CURVE3_COLOR "COLOR NAME"
setenv ADAMSPP_PLOT_CURVE4_COLOR "COLOR NAME"
```

You can replace *COLOR NAME* (in uppercase letters) with any of the available 65 colors. You must use double quotation (" ") marks to enclose a type name with embedded spaces, such as "BURNT ORANGE". You do not need to use the double quotation marks to enclose a type name if it is a single word, such as GOLD.

Available colors are:

APRICOT	GREEN/YELLOW	PLUM
AQUAMARINE	INDIAN/RED	RAW SIENNA
BITTERSWEET	LAVENDER	RAW UMBER
BLACK	LEMON/YELLOW	RED
BLUE	MAGENTA	RED ORANGE
BLUE GREEN	MAHOGANY	RED VIOLET
BLUE GREY	MAIZE	SALMON
BLUE VIOLET	MAROON	SEA GREEN
BRICK RED	MELON	SEPIA
BROWN	MIDNITE/BLUE	SILVER
BURNT ORANGE	MULBERRY	SKY BLUE
BURNT SIENNA	NAVY BLUE	SPRING GREEN
CADET BLUE	OLIVE GREEN	TAN
COPPER	ORANGE	THISTLE
CORNFLOWER	ORANGE RED	TURQUOISE BLUE
CYAN	ORANGE YELLOW	VIOLET/PURPLE
FOREST GREEN	ORCHID	VIOLET/BLUE
GOLD	PEACH	VIOLET/RED
GOLDENROD	PERIWINKLE	WHITE
GREY (GRAY)	PINE GREEN	YELLOW
GREEN	PINK (CARNATION PINK)	SPRING GREEN
GREEN/BLUE		

The color names in parentheses are alternative names for the preceding color. If you misspell a variable, Adams Solver uses the default color for that variable; it does not display an error message. If you misspell a COLOR NAME, however, Adams Solver displays an error message.

If you do not set a color for a given variable, Adams Solver uses the default color.

Remote X11 Access to MSC Software Applications

You can use a remote X11 device to display the MSC Software applications graphics by setting the DISPLAY environment variable. For example, if you are using an X terminal to log onto a workstation licensed to run MSC Software applications, you can set the DISPLAY environment variable at the operating system prompt to include the name of the X terminal:

```
X terminal name is: xcitement
DISPLAY variable: setenv DISPLAY xcitement: 0.0
```

For more information on accessing Adams View through remote X11 devices, see Adams View Preferences.

Adams Product License Checkout

You can specify the number of times an Adams product attempts to check out a license. The default is 60 attempts and the time interval between each check, which you cannot change, is 60 seconds. Adams Toolbar checks the registry entry licenseRetry to determine how many times it should try to check out its license.

To set the number of times to check out a license:

1. Right-click the **Adams Toolbar** tool, and then select **Start Registry Editor**.

   The Registry Editor appears with options for setting all products.
2. In the treeview, click **MSCA**.
3. In the Registry area, click **licenseRetry**.
4. Set the number of times, and then select **OK**.

Temporary File Location

You can change where Adams places temporary files it generates during a simulation. By default, it places them in /var/tmp. You can use the environment variable TMPDIR in your .cshrc to set where Adams places the temporary files. You can direct Adams to place the temporary files in any directory, as long as you have the appropriate permissions to that directory.

For example, to direct Adams to place temporary files in the directory /usr/users/me/tmp, enter the following in your .cshrc:

```
setenv TMPDIR /usr/users/me/tmp
```

In the Adams Toolbar

Preferences in the Adams Toolbar

You can set preferences for all products at once or for a particular product. Not all products have preferences that you can set. In addition, many products share preferences with other products, and the Adams Registry Editor displays any shared preferences for each product.

To set all product preferences at once:

- Right-click the **Toolbar** tool MSC, and then select **Start Registry Editor**.

  The Registry Editor appears with options for setting all products.

To set preferences for a particular product:

1. Right-click the product's tool, and then select the **Change Settings command**.
2. Change the preferences as listed in the next sections:
   - Adams View Preferences
   - Adams Solver Preferences
   - Adams PostProcessor Preferences
   - Adams Car Preferences
   - Template-Based Product Preferences
3. Select **OK**.

Adams PostProcessor Preferences

Adams PostProcessor shares graphical settings with Adams View. For your convenience, the Graphics folder appears when you select **Change Adams PostProcessor Settings** from the Toolbar. For more information, see Adams View Preferences.

Note:	When you change graphical preferences for Adams Processor, you also affect the graphics preferences for Adams Car, and Adams View.

Adams Solver Preferences

You can set preferences, listed in the table below, for standard Adams Solver and for user libraries that you created to run with Adams Solver. Each library has its own set of preferences. For more information about user libraries, see Creating User Libraries.

Before setting your preferences, be sure that you are setting the preferences for the desired library or standard Adams Solver.

To verify Adams Solver:

- On the Adams Toolbar, check the appearance of the **Adams Solver** tool:

  indicates that you are working with the standard Adams Solver.

  indicates that you are running Adams Solver with a user library.

Note:	You may have several user libraries. To verify which one is active, move your mouse over the tool until the tip text appears. The name you assigned to the library appears in the tip text. Alternatively, right-click on the tool, and select Info to see information about the library.

Parameter:	Description:
solverSelection	You can select either: ■ **Fortran Solver** - Our commercially available solver (F77). ■ **C++ Solver** - Our C++-based solver, which is faster, provides new linear analysis capabilities, and has an improved methodology for identifying and handling redundant constraints. Currently, it does not support all modeling elements that the Adams Solver (FORTRAN) supports. For more information on the different solvers, see the online help for Adams Solver.
remoteHost	Name of the remote host where you run Adams Solver. **Note:** Network access and multi-task package licenses allow you to submit tasks to Adams Solver while Adams View runs on one of many desktop workstations. To see if you can submit multiple tasks to Adams Solver, check the licensing information on the password certificate that is included in the Installer's Kit. If you do not know the type of license you have, check with the administrator who installed the Adams products.
remoteHostWD	Specifies a directory that Adams Solver uses to write out its files and search for input files. The directory is optional. You need to specify it only if the user's file system is not automounted on the remote machine upon log in.
remoteHostID	The name of the Adams Solver installation directory on the remote machine.
memSize	Sets the memory model Adams Solver uses to simulate a model. You can select the standard memory models provided, as well as any custom models you created. For information on creating a memory model, see Creating a custom memory model.
custMemModel	Sets the name of a custom memory model that you have previously created. For this to take effect, memSize must be set to custom.

Parameter:	Description:
inputFile	Specifies that Adams Solver use the command language file (.acf) when running in scripted or batch mode. You can enter the path name of your command file or right-click in the text box and select one using the File Browser.
workingDirectory	Specifies a directory where Adams Solver writes output files and searches for input files. By default, the parameter is empty and does not affect the location in which the output files are generated, or change the location where Adams Solver looks for input files.
runMode	Sets the mode in which Adams runs: ■ **Interactive** - Adams runs and waits for user input. ■ **Scripted** - Adams runs with the command file that you specify in the parameter inputFile. (If you have not specified a command file and you run in scripted mode, Adams Solver launches in interactive mode.) ■ **Batch** - Adams Solver runs with the command file specified in the inputFile. If you have not specified a command file, and run in batch mode, Adams Solver launches interactively.

Adams View Preferences

You can set two types of preferences in Adams View:

- Preferences for Adams View and any user library - The parameters can be different for each user library that you create to run with Adams View and for standard Adams View.
- Preferences for Adams View only - The parameters are global graphics settings. They affect all Adams products that have a user interface, such as Adams Car. You can view and change them only when you are working with standard Adams View. They appear in the Registry area as a subfolder, Graphics.

Before setting your preferences, be sure that you are setting the preferences for the desired library or standard Adams View.

To verify Adams View libraries:

- On the Adams Toolbar, check the appearance of the product tool:

  indicates that you are working with standard Adams View.

  indicates that you are running Adams View with a user library.

Note:	If you have many user libraries, verify which one is active by moving your cursor over the tool until the tip text appears. The name you assigned to the user library appears in the tip text. Alternatively, right-click on the tool, and select Info to see information about the user library.

	General Preference Settings in Adams View
Parameter:	**Description:**
workingDirectory	Specifies a directory where Adams View writes output files and searches for input files. By default, the parameter is empty and does not affect the location in which the output files are generated, or change the location where Adams View looks for input files.
runMode	Sets the mode in which Adams View runs: ■ **Interactive** - Adams View runs and waits for user input. ■ **Scripted** - Adams View runs with the command file that you specify in the parameter inputFile. (If you have not specified a command file and you run in scripted mode, Adams View launches in interactive mode.)
custMemModel	Sets the name of a custom memory model that you have previously created. For this to take effect, memSize must be set to custom.
solverUserLibrary	Sets the Adams Solver user library to run from within Adams View.
memSize	Sets the memory model Adams View uses to simulate a model. You can select the standard memory models provided as well as any custom models you created. For information on creating a memory model, see Creating a custom memory model.
inputFile	Specifies that Adams View use a specified command language file (*.cmd) when running a product in scripted mode.
searchPath	Allows you to specify the path file that contains search paths for different types of files that can be loaded into Adams View. For more information on path files, see Modifying the Adams Path Files.
description	Available for user libraries only. Text that describes the library, such as what it does or how it was created.

	General Preference Settings in Adams View
Parameter:	**Description:**
shortName	Available for user libraries only. A project name that appears on the Adams Toolbar and identifies the user library, such as AView1.
fileCaching	Available for standard Adams View only. When performing animations that contain flexible bodies, Adams View often produces a large cache of data before it can perform the animation. The cache can be as large as several hundred megabytes. The default is to produce these caches on disk if there is enough disk space available. If the disk space is not available, Adams View places the cache in memory. Note that storing the cache on disk results in longer animation setup time (up to two times). You can disable file caching to increase animation setup speed but if there is not enough memory to produce a cache, Adams View may exit with an Out of Memory message.
textEditor	Available for standard Adams View only. Specifies the text editor to be used by default.

	Global Graphics Settings in Adams View
Parameter:	**Description:**
Graphics Driver	On all platforms, Adams View supports the X11 window driver as well as the one for the native graphics capabilities of the system. The X11 driver lets Adams View run from remote X terminals, and the native driver lets Adams View use any specialized or high-performance graphics hardware on the system. If the machine on which you are displaying Adams View does not have the hardware to meet Adams specifications, a warning message appears. Select X11 to resolve this problem.
OpenGL Software Assisted	If you selected the option Native Open GL for Graphics Driver, it uses the hardware acceleration to render overlay planes. The overlay planes are used for the: ▪ Rubber band box drawn for selection or zoom. ▪ Temporary geometry (arrows) displayed when creating joints. ▪ Temporary geometry when sketching curves, extrusions, and revolutions. ▪ Object names when in selection mode. Some graphics cards do not offer hardware acceleration for overlay planes. If you have one of these graphics cards, then you should use OpenGL Software Assisted. Software assistance simply means that Adams View draws the overlay plane geometry instead of relying on the hardware. OpenGL Software Assisted is slower than Graphics Driver because the screen refresh rate for the overlay plane will be affected by the other geometry in the view.
Double Buffering	Double buffering of screen updates in Adams View provides fluid animation updates. If you are using the X11 Window driver in Adams View, double-buffering uses more memory and generally runs slower than double-buffering with native graphics.
Hardcopy Resolution	Specifies the resolution for shaded images written to a postscript file. Hardcopy Resolution does not affect wireframe images, such as xy plots. The default setting is 300 dots per inch (DPI). Note that earlier versions of Adams View (10.1 and earlier) used a default resolution of 75 DPI. We do not recommend that you use a resolution lower than 75 or higher than 600 DPI. Note also that larger values produce larger postscript files, and will likely increase processing time for the printer.
Overlay Backgrounds	Controls whether or not a background appears behind pop-up text, such as the names of modeling elements and position coordinates. Because some graphics cards do not fully support OpenGL with overlay planes, which the background is, you may not be able to see the text. If this occurs, disable this option to remove the background and leave the text.

	Global Preferences in Shared
Parameter:	**Description:**
minSpaceMB	Enter the amount of disk storage space (MB) available to the working directory below which certain file writing functions of Adams will not start. Specific actions which this will influence: ■ Adams Solver execution

Adams Car Preferences

The following preferences can be set for Adams Car, in addition to the preferences shown in the Template-Based Product Preferences table below.

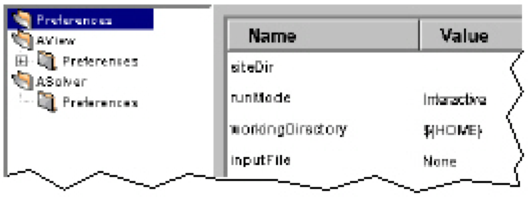
Parameter:	Description:
windowTop	Specifies the location of the main Adams Car window at startup. Positive values indicate the number of pixels from the top of the screen. Negative values indicate the upper location of the window as a percentage of the screen size. For example, windowTop = 0 specifies that the main window will be located at the top of the screen.
windowLeft	Specifies the location of the main Adams Car window at startup. Positive values indicate the number of pixels from the left of the screen. Negative values indicate the horizontal location of the window as a percentage of the screen size. For example, windowLeft = 0 specifies that the main window will be located at the left of the screen.
windowHeight	Specifies the default window height. Positive values indicate the height in pixels. Negative values indicate the height as a percentage of screen size. For example, windowHeight = 100 specifies that the main window will use all of the available screen height.
windowWidth	Specifies the default window width. Positive values indicate the width in pixels. Negative values indicate the width as a percentage of screen size. For example, windowWidth = 100 specifies that the main window will use all of the available screen width.

Template-Based Product Preferences

The template-based products have a private set of preferences, shown in the table below, and a shared set with Adams Solver and Adams View.

Parameter:	Description:
siteDir	Specifies the location of the site repository in which you have created user libraries or binary files. For more information, see User Library Overview and Binary Files.
privateDir	Specifies the location of your private repository. For more information, see the online help for your template-based product.
privateCfg	Specifies the name and location of your private configuration file. For more information, see the online help for your template-based product.
runMode	Sets the mode in which the product runs: ■ Interactive - Runs and waits for user input. ■ Scripted - Runs with the command file that you specify in the parameter inputFile. (If you have not specified a command file and you run in scripted mode, the template-based product launches in interactive mode.) ■ Batch - Runs with the command file specified in the parameter inputFile. If you have not specified a command file, and run in batch mode, the template-based product launches in interactive mode.
workingDirectory	Specifies a directory where the template-based product writes output files and searches for input files. By default, the parameter is empty and does not affect the location in which the output files are generated, or change the location where the product looks for input files.
inputFile	Specifies that the template-based product use a specified command language file (*.cmd)when running in scripted mode. ■ For Adams Solver: specifies the command file (.acf) when the Adams Solver executable runs in scripted or batch mode. ■ For Adams View: specifies the command file (.cmd) when the Adams View executable runs in scripted mode.

For your convenience, the preferences for Adams View and Adams Solver are included with the preferences for the template-based product in the Registry Editor, as shown next:



Name	Value
siteDir	
runMode	Interactive
workingDirectory	$(HOME)
inputFile	None

For information on preferences specific to Adams Solver and Adams View, see Adams Solver Preferences and Adams View Preferences.

Windows Preferences

Adams Solver Pausing

You can set the variable MDI_PAUSE to set a pause after Adams Solver starts running in interactive mode. The default setting is 0 for no pause.

To activate the pause:

1. From the **Start** menu, point to **Settings**, and then select **Control Panel.**
2. From the Control Panel folder, double-click **System**, and then select the tab **Environment**.
3. Enter the variable **MDI_PAUSE** and set it to **1**.
4. Select **OK**.

Adams Environment

To set preferences:

1. From the **Start** menu, point to **Programs**, point to **Adams *x*** (where *x* is the release number), and then select **Settings & License.**

   The Adams registry editor appears.
2. Set the options, as necessary.
3. Select **OK**.

Licensing

Parameter:	Description:
licenseFile	The full path to the MSC License file, or the port and server name. Right mouse button for the file browser menu.
licenseRetry	Enter the number of times an Adams product attempts to check out a license. The interval is 60 seconds and cannot be changed.

Note:	The `MDI_ADAMS_RETRY` environment variable is not valid anymore. Please use the licenseRetry feature above.

Help Preferences

Parameter:	Description:
docsDirectory	Directory that contains your Adams Online Help files. This can be a CD-ROM location.

Shared Graphics

Parameter:	Description:
Graphics_Picture	Only select Null for the OpenGL graphics driver.
Double_Buffering	Selecting "enabled" smooths the display of animations.
OpenGL_Software_Assisted	Sets the Native OpenGL mode to use software emulation of overlay planes. This should be used for graphics cards which do not fully support hardware accelerated Native OpenGL. Overlay planes are used for the: ■ Rubber band box drawn for selection or zoom. ■ Temporary geometry (arrows) displayed when creating joints. ■ Temporary geometry when sketching curves, extrusions, and revolutions. ■ Object names when in selection mode. Software assistance simply means that Adams View draws the overlay plane geometry instead of relying on the hardware. Native OpenGL with software assistance is slower because the screen refresh rate for the overlay plane will be affected by the other geometry in the view.
Overlay_Backgrounds	Controls whether or not a background is added to pop-up text such as names of modeling elements and position coordinates. Because some graphics cards do not fully support OpenGL overlay planes, you may not be able to see pop-up text in Adams applications. Select disabled if you cannot see pop-up text during when trying to select modeling elements. The default is enabled.
Hardcopy_Resolution	Specifies the resolution for shaded images written to a postscript file, in Dots Per Inch (DPI). The higher the DPI number the greater the resolution. Hardcopy Resolution does not affect wireframe images, such as xy plots. The default setting is 300 dots per inch (DPI). We do not recommend that you use a resolution lower than 75 or higher than 600 DPI. Note also that larger values produce larger postscript files, and will likely increase processing time for the printer.

Shared Preferences

Parameter:	Description:
startInWorkingDirectory	Enables or disables Adams from changing to the workingDirectory. Defaults to disabled.
workingDirectory	Specifics the working directory Adams changes to when “startInWorkingDirectory” is enabled.
minSpaceMB	Enter the amount of disk storage space (MB) available to the working directory below which certain file writing functions of Adams will not start. Specific actions which this will influence: ■ Adams Solver execution

AView Preferences

Parameter:	Description:
memSize	Select the Adams Solver (F77) memory model size from the list of choices. You can also define a custom memory model as explained in Setting Custom Memory Model Size on Windows. The Memory Model Sizes table shows the variable and array sizes for the memory options and explains each variable and array.
searchPath	Use this to specify a custom Adams View path file.
textEditor	Specify a text editor. Select the standard Windows Notepad editor, or enter another editor in the Other text box (for example, vi.exe). Note that if you enter the name of another editor, it must be found by the PATH environment variable in Windows.
fileCaching	Controls how caching is performed for flex bodies. Enable to cache flex bodies on disk and decrease memory footprint during animation runs.

ASolver Preferences

Parameter:	Description:
memSize	Select the Adams Solver (F77) memory model size from the list of choices. You can also define a custom memory model as explained in Setting Custom Memory Model Size on Windows. The Memory Model Sizes table shows the variable and array sizes for the memory options and explains each variable and array.
solverSelection	CXX selects the standard commercial Adams Solver. F77 selects the older technology Fortran based Adams Solver. CXX is the default if (none) is selected.
numThreads	Enter the default number of parallel threads that you want Adams Solver to run with (CXX only).

Adams Car Preferences

The following preferences can be set for Adams Car, in addition to the preferences shown in the Template-Based Product Preferences table below.

Parameter:	Description:
windowTop	Specifies the location of the main Adams Car window at startup. Positive values indicate the number of pixels from the top of the screen. Negative values indicate the upper location of the window as a percentage of the screen size. For example, windowTop = 0 specifies that the main window will be located at the top of the screen.
windowLeft	Specifies the location of the main Adams Car window at startup. Positive values indicate the number of pixels from the left of the screen. Negative values indicate the horizontal location of the window as a percentage of the screen size. For example, windowLeft = 0 specifies that the main window will be located at the left of the screen.
windowHeight	Specifies the default window height. Positive values indicate the height in pixels. Negative values indicate the height as a percentage of screen size. For example, windowHeight = 100 specifies that the main window will use all of the available screen height.
windowWidth	Specifies the default window width. Positive values indicate the width in pixels. Negative values indicate the width as a percentage of screen size. For example, windowWidth = 100 specifies that the main window will use all of the available screen width.

Template-Based Product Preferences

The template-based products have a private set of preferences, shown in the table below, and a shared set with Adams Solver and Adams View.

Parameter:	Description:
siteDir	Specifies the location of the site repository in which you have created user libraries or binary files. For more information, see User Library Overview and Binary Files.
privateDir	Specifies the location of your private repository. For more information, see the online help for your template-based product.
privateCfg	Specifies the name and location of your private configuration file. For more information, see the online help for your template-based product.

Search Paths

At installation, Adams products, including Adams Car, Adams Driveline, and Adams View, create a default aview.pth file. It is in the aview subdirectory, beneath the installation directory, *install_dir*.

The aview.pth file allows you to specify search paths for the different types of files that can be opened in Adams View. This is useful for files that are shared with other users, and are not in your current working directory. Below is an example of an aview.pth file:

```
.bin    c:/install_dir/aview/
.adm    c:/install_dir/datasets/
.cmd    c:/aview/cmd_files/
.dat    c:/adams/test_data/
.gra    c:/staff/my_home_dir/adams/output/
.req    c:/staff/my_home_dir/adams/output/
.res    c:/staff/my_home_dir/adams/output/
.igs    c:/staff/my_home_dir/adams/iges_files/
.dct    c:/usr/install_dir/aview/
```

Each line in the file specifies a search path for a specific file extension. The first line in the example file above specifies a search path for files with an extension of .bin. You can have multiple paths for each extension by adding another line with the same file extensions and an alternate path. Remember that each line must begin with a file extension.

Any file extension can be put in the Adams View path file. Therefore, you do not have to name your files with the standard Adams Solver and Adams View extension. Use the following conventions to add a file extension to the path file:

- Begin each line in column one with the desired file extension.
- Begin each file extension with a . (period).
- Separate the extension and search path for the extension with space. You can use a single space, tab, or a combination of these.
- Complete each search path, starting from the root (/) of the file system.
- End each search path with a trailing / (forward slash).
- Ensure that each line is less than 255 characters in length.

  If you want to make the changes to the Adams View path file affect all Adams View users, edit the default aview.pth file mentioned above.