COMMAND
User Guide

www.transmagic.com

Table of Contents

COMMAND Interface

Release Notes

Release Notes

Note: The TransMagic COMMAND interface is an optional component and requires a special license. Please contact TransMagic for more information at: info@transmagic.com

If you are a new TransMagic COMMAND user then you may bypass this section and proceed to COMMAND Description topic. Existing users will want to read the Release Notes and make the appropriate changes to their developments.

-o-

TransMagic R10

TransMagic R10 sp0

- New Translators
  - DWG Read\Write, up to version 2013
  - New image output formats: PNG, JPG & BMP

- TMCmd now supports UNICODE and thus:
  - Multi-byte character directory structures
  - Multi-byte character file names
  - Multi-byte character part names
  - Multi-byte character languages

- New format version support
  - Upgraded to CATIA V5 R23
  - Upgraded to ACIS R23
  - Upgraded to Creo 2.0
  - Upgraded to Parasolid 25.0
  - Upgraded to JT 9.5
  - Upgraded to Inventor 2013
  - Upgraded to SolidWorks 2013

- New functionality
  - Added reading of "Last Save State" or in V5 nomenclature "In Work Object" to the V5 reader. The option is -iwo
  - Removed all external initialization from TMCmd. This mainly pertained to TMLicenseManager.exe & TMLicenseManagerNodeLock.exe. Previously these two executables needed to be specially handled and copied over to the %ALLUSERSPROFILE%\TransMagic\Licensing directory & then registered before TransMagic licensing would run. These executables are now distributed with TMCmd and initialized automatically upon application start-up, removing the need for OEMs to manage these components.
  - Like the license managers, the NGRAIN initialization is now handled by TMCmd automatically.
  - When writing out a file to a directory that doesn't exist, TMCmd will automatically create the directory structure.
  - Added new XML output options:

-xmlasm = XML Assembly File. Sending this command line option will break apart an assembly into individual files with an XML file that contains the assembly structure and part transform matrices.
-xmlatom = XML Atomize. This command is designed to be sent along with the -xmlasm command above. The atomize operation will further break down part files that contain multiple bodies into discretized part files that each contain only one single body per file. Following is a break-down of the behaviors of this option: • For part files with multiple bodies, this option will instruct TMCmd to create individual files for each body in the part. Each new file created will be named the same as the "top-level" part file appended with -01.*, -02.*, -03.*, and so on.

- You can currently specify -xmlasm even for a non-assembly or flattened file such as SAT. The current behavior is that a single XML file will get created along with the single output file format. If the -xmlatom option is specified in conjuction with the -xmlasm option then TMCmd will create the -01.*, -02.*, -03.* files as above for each body in the file.
- If a part file contains only a single body then TMCmd will create that single file with no appended -01.*, -02.*...
- When the -xmlatom option is specified, add a new sub-level "Body", "/Body" is added to the XML assembly file which lists the body by both the appended file name with -01.*, -02.*, etc. and also by the actual attribute name of the body in that file.

-xmlbbox = XML Bounding Box. This command is designed to be sent along with the -xmlasm command above. The -xmlbbox command will calculate the minimum Bounding Box size relative to the global origin for each part (or body) in a file. It will create a new sub-category in the XML file: Bounding_Box

-xmlmass = XML Mass Properties. This command is designed to be sent along with the -xmlasm command above. The -xmlmass command will calculate the the mass properties for each part (or body) in a file including Volume, Center Of Mass, Principal Axes, Inertia Tensor and Principal Moments. It will create a new sub-category in the XML file: Mass

-xmlsurf = XML Surface Area. This command is designed to be sent along with the -xmlasm command above. The -xmlsurf command will calculate the surface area for each part (or body) in a file. It will create a new sub-category in the XML file: Surface_Area

- Other Improvements
  - Improved translation performance by 2-3X across the range of translators
  - Several general bug fixes

-o-

TransMagic R9

TransMagic R9 sp2

- TMCmd now supports UNICODE and thus:
  - Multi-byte character directory structures
  - Multi-byte character file names
  - Multi-byte character part names
  - Multi-byte character languages

- New language support:
  - Japanese

- New format version support
  - Upgraded to ACIS R23
  - Upgraded to Creo 2.0
  - Upgraded to Parasolid 25.0
  - Upgraded to JT 9.5
  - Upgraded to Inventor 2013
  - Upgraded to SolidWorks 2012

- Other Improvements
  - Improved translation performance by 2-3X across the range of translators
  - Several general bug fixes

-o-

TransMagic R9 sp1

- New language support:
  - French

- New format version support
  - Upgraded to CATIA V5 R22
    - Updated default CATIA V5 write version to R18 (up to R22)
  - Upgraded to NX 8.0
  - Upgraded JT Open Toolkit, doesn't affect version but addresses some JT bugs

- New functionality
  - Added reading of hidden entities for Pro/E (Creo Parametric)

- Stability improvements
  - Fixed -flatten option bug
  - Fixed a few option settings that were ignoring their command line directives

-o-

TransMagic R9 sp0

- Removed the -vxf VX flavor option from the Parasolid Write settings as this option is no longer required for writing Parasolid files for use in VX.
- Added TIF image output with full size and rendering options.
- Added sophisticated new -skin option for skinning the options interface and progress meter to match your interface style.
- Stabilized all third-party CAD kernel initialization and environmental set-up to occur at run-time which reduces unnecessary initialization until it's needed. These checks also occur before each translation so in the case that a CAD kernel was unitialized or an environment setting altered, these will be properly reset before translation without any need for user intervention.

TransMagic OEM Partner Changes

- TMCmdLic.lib is no longer used so there is no longer a need to use the TMCmdInit() function in your code. In fact starting with R9sp0.0, you can no longer use this library or function as it will not work with R9sp0.0 and beyond. This is good news though as initialization is much simpler. OEM licensing is now done via the -tmcmdinit command line function.
- A new sample application, "TMCmdLicTestNoLib", was created that demonstrates usage of the new -tmcmdinit function. For more documentation see the section Partner Specific COMMAND Syntax.
- A new TransMagic OEM Partner section has been added to the TransMagic COMMAND docs that describes all information relevant specifically to TransMagic OEM partners.
- Added Multi-language User Interface (MUI) support for multiple languages. For more documentation see the "Global Options" in the Advanced COMMAND Syntax section.
- Added support for the reading of suppressed parts in Pro/E (Creo Parametric). This applies to assemblies with suppressed parts. This does not currently apply to parts with suppressed bodies.
- Multi-language User Interface (MUI) support for multiple languages. For more documentation see the "Global Options" in the Advanced COMMAND Syntax section.

- TransMagic Windows Environment Abstraction:
  - TMCmd has been completely abstracted from it's native Windows environment so that you can install any version of TransMagic under your application install directory and call TMCmd.exe from that directory.
  - Multiple versions of TransMagic\TMCmd can be copied to multiple directories and run independently. This allows multiple versions of your app to be installed integrated with multiple versions of TransMagic.
  - To facilitate this you need to add additional information directly attached to the -tmoem command. This information is used to identify at the very least your application by name but you can also use it to identify your company, application, & application version. This information is stored in the registry at run-time and is used by our licensing system to keep track of the multiple versions.
  - For more information see the section Partner Specific Installation Options.

- TransMagic OEM Installer Changes:
  - Added new option to skip prerequisite installation.
  - Added new single progress meter for prerequisite install that eliminiates the "flashing windows" behavior of previous installs.
  - For complete installation options see the section Partner Specific Installation Options.

-o-

TransMagic R8

TransMagic R8 sp5

- Upgraded Translator: Parasolid v23
- Upgraded Translator: JT 9.4
- Upgraded Translator: SolidWorks 2011
- Upgraded Translator: NX 7.5

-o-

TransMagic R8 sp4

- Removed "Legacy Satellite Interface" documentation. The current timeframe for End Of Lifing this iterface will be TransMagic R10.
- 64-bit Native. Allows for virtually unlimited file size translation when running on XP, Vista or Win 7 64-bit systems. Adequate RAM recommended!
- New Read format: CATIA V5 Graphics *.cgr
  Limitations:
  - Reading: Supports only the CATIA V5 scene graph data saved out to the CGR file. This means that original assembly transforms will appear correct but if parts are transformed again within the assembly then the CGR support will still reflect only the original assembly transforms as this is how the scene graph is stored in the CGR file.
  - Writing: Does not support the writing of colors.
- Upgraded Translator: ACIS R21
- Upgraded Translator: Parasolid v22
- Upgraded Translator: JT Open 5.3
- Upgraded Translator: Pro/E WildFire 5.0
- Upgraded Translator: SolidWorks 2010
- Upgraded Translator: HOOPS 18.0
  - Note: An associated change to the HSF format has been implemented for this release. Previously you would set the version from 1 to 18 corresponding to HSF versions 1.05 through 18.00. However, TMCmd users have requested finer version control to support point releases of the HSF format. For this reason the HSF version is now specified from 100 - 1800. If you want to write out whole HSF versions then use 100, 200, 300, and so on. If you want to write out point versions then the latter two digits represent the specific point version. For example if you wanted to output version 6.30 then you would send 630 as the version or 17.10 then you would send 1710 as the version.
- Upgraded Translator: CATIA V5 R20
- Upgraded Translator: UG/NX 7
- Upgraded Translator: NGRAIN 4.1
- Implemented OEM Licensing which allows for unattended licensing at the command line. This option is available per the appropriate contractual arrangement with TransMagic, Inc.
- Added -dlgmsg option which tells TM COMMAND to output dialog messaging for any errors encountered. This option is off by default. It is convenient for OEM partners implementing TM COMMAND in an end-user GUI as it simplifies error trapping and displays meaningful messaging to the end-user. This option is described in Advanced COMMAND Syntax.
- Improved Translation Settings (-optiondlg) layout. Started adding ToolTips to busy forms to reduce text, simplify options and improve readability.
- Added new NGRAIN Translation Options, see the NGRAIN options for more details:
  - -resbnds Option = Resolution Boundaries.
  - -minres Option = Minimum Resolution.
  - -maxres Option = Maximum Resolution.
  - -pa Option = Part Axis Alignment.
  - -gi Option = Geometry Instancing.
  - -sep Option = Special Entity Processing has been changed to -ngrps or NGRAIN Groups. This option now relies on a file (GroupName.grp) that contains the NGRAIN Group settings. This change has necessarily deprecated the -sn & -fn options.
- Added the ability to check the TransMagic License properties in the registry. This allows an application that calls TMCmd to verify which read or write formats the TransMagic license

currently on the system has the ability to call. See the topic Check The TransMagic License Properties.

-o-

TransMagic R8 sp3

- Cleaned up TM Batch Status Window output for better updates and readability.
- Added elapsed time to Batch Status output.
- TM V5 Environment is now established on a per user account basis for all operating systems. This allows any account type to establish the TM V5 Environment automatically without any user rights issue.
- Fixed JT resolution bug where output resolution was always "Normal" regardless of resolution specified.
- Added more JT output resolutions. The JT Facet Resolution can now be one of five settings: Lowest, Low, Normal, High, Highest. These settings refer to the common facet resolution setting "normal deviation" and correspond to the following settings: Lowest = 45 degrees, Low = 30 degrees, Normal = 15 degrees, High = 10 degrees, Highest = 5 degrees.
- Fixed TM COMMAND issue where any option that started with "-h" triggered help output at the command window.
- Changed TM COMMAND Hide Construction -hc option to -ch.
- Changed TM COMMAND Translate Hidden Entity -he option to -eh.
- Changed TM COMMAND "inopts" field to "globalopts" as this is a more appropriate name since these options are global and apply to the incoming file and thus apply to all outgoing formats. Docs and command line output -h option have been changed to reflect this new field name.
- Added Advanced Faceting Functionality for JT output to TransMagic and TM COMMAND. This new functionality allows for virtually unlimited Viz-Rep resolution of JT files. One such use would be in high-end large format renderings where super-fine polygonal control is required.
- Added TM COMMAND -optiondlg page modifier option which allows the option dialog to only open with a specific page such as IGES Write Options or CATIA V5 Read Options, etc. This makes for even better third party application integration and reduces end-user confusion when only the option dialog they're looking for pops up.
- Added the ability to run Lite Repair and Full Repair successively. You can send -lr or -fr in any order. Lite Repair will always be applied first.
- Added command line licensing with no GUI for OEM/Enterprise licensing automation. See Advanced Licensing Options for more details.

-o-

TransMagic R8 sp2

- Upgraded Translator Version: ACIS R20
- Upgraded Translator Version: NX6
- Upgraded Translator Version: Parasolid v21 (Default now bumped up from v17 to v18)
- Upgraded Translator Version: JT 9.3
- Added version check to ASAT format as they are only supported for ACIS R17 and up.
- Resolved Save As *.sat assembly flattening issue where some assemblies transforms were being changed.
- Resolved issue where some parameters sent in TM COMMAND were not being processed while settings established in the -optiondlg were.
- Fixed TM Batch output logs. They are now copied to the output directory instead of left in the user's TEMP directory.
- Fixed Inventor Single Part scaling. It was set to mm instead of cm so parts were coming in 1/10th of the scale they were supposed to be.
- Added Minimize icon to Batch interface.
- Fixed HTML output of TM COMMAND. Previously it was throwing an "File is wrong version" error.
- Added HTML output to TM Batch.
- TM Batch now remembers where it was last placed on the desktop. Previously it always came up in the center of the screen.
- Fixed bug in German version where TransMagic would report that the TransMagic V5 Environment file was not found - even when it was present.
- Fixed bug with TM COMMAND where local foreign decimal symbols in the computer's "Regional Settings" were being switched from "," to "." and vice versa for domestic decimal symbols.
- Fixed SAT flattening issue.

-o-

TransMagic R8 sp1

- R8 sp0 went out based on the latest available version at the time which was JT 4.3.2.0 (which corresponds to JT 9.1). Siemens informed us that this version had a bug in it that affected the viewing of JT files in Teamcenter. Siemens issued a 4.3.2.1 patch which we built into TM R8sp1.
- JT B-Rep entities were scaled up by 1000. This is because the default Parasolid unit – which is Meters internally was not being set. We've now set the proper internal Parasolid unit for JT B-Rep entities and now the scaling is correct.
- Fixes Parasolid application conflicts – especially with M-COSMOS and MasterCAM. Both of those aforementioned apps use poor put their DLLs in the global Windows “PATH” statement. So sp1 actually has a work-around implemented where we're checking for these and inserting our own at run-time.
- Fixes a V5 product issue where if an existing V5 installation was present then the user would get warnings about missing V5 components. Here again, it’s not a TransMagic bug but a V5 bug that we've implemented an automatic work-around for.
- Installer fix: The R8 sp0 installer was not looking for specific 32-bit “emulation” directories or registry entries in case old versions of TM were installed on 64-bit machines. This issue came up quite a bit. So now we're looking for 32-bit directories\registry entries on 64-bit machines so any platform will uninstall previous versions of TransMagic correctly. Also, if a 64-bit OS is detected then the appropriate Inventor Viewer 64-bit will be downloaded if necessary.

-o-

TransMagic R8 sp0

Welcome to TransMagic COMMAND! This is the inaugural release of this sophisticated translation, Repair, and automation toolkit. TransMagic COMMAND is the result of years of feedback from our previous TransMagic Satellite Interface. Here are the top requests:

- Eliminate the need to use the intermediate TransMagic TMR file for data exchange.
  - This cuts coding by 50% for one translation and if you implemented all TransMagic translators, this cuts coding by 90%. With TransMagic COMMAND you can input a part or assembly and not only output to your desired format but even every single format offered by TransMagic simultaneously!
- Full Assembly Hierarchy support for all formats that support it.
- Simultaneous Lite and Full Repair.
  - Both Repair methods can be performed in a single command. This allows for example an IGES file to be run through Lite Repair for stitching and minor geometric Repair and then through Full Repair for surface simplification, surface Repair and improving the model precision.
- A simpler method to handle the massive amount of potential translation options.
  - In one single command (TMCmd -optiondlg) TransMagic has captured all of our years of industry knowledge and best practices automatically. There is a fully developed GUI interface (see the Description heading for more info) that can be launched which eliminates literally man years of development and also gives your users access to TransMagic's translation options for fine tuning.
  - When TransMagic COMMAND is installed all our "default settings" which address the lowest common denominator of uses and addresses the vast majority of translations are populated in the system registry. These default settings are accessible and changeable by the end user via this "Translation Options" dialog.
  - Of course your can still override any of these default settings by sending options via the command line, though again, no options are required as the default settings in the registry will address the majority of users.
  - All of these translation options are fully documented! This further reduces implementation time and research time and gives the end user the best possible information regarding the effects of setting various options. Help docs are launchable via the "Translation Options" dialog.
- Traditional command line parameters.
  - Instead of the Legacy Satellite Interface's *nn* option format, TransMagic COMMAND now uses the traditional -nn parameters that so many other command line applications use.
- All TransMagic GUI formats are supported via TransMagic COMMAND.
  - This includes all geometric formats, 3D Visualization formats and image formats. Again all formats can be output in one single command string.

Q: What does this mean for the TransMagic Satellite Interface?
A: The TransMagic Satellite Interface is now called the "TransMagic Legacy Satellite Interface". A deprecation date has not been set and will not occur during the TransMagic R8 series. It will be deprecated at some point in the future but not until we've talked to all of our partners and helped to get them upgraded to the TransMagic COMMAND interface. So this new interface will not break any of your existing code. However, starting with the release of the TransMagic COMMAND interface there are good reasons to upgrade. The Legacy Satellite Interface:

- Does not support Assembly Hierarchy.
- Does not support as many formats already and will not support new formats that are added in the future.

- Will not support new capabilities that are added to the TransMagic COMMAND interface.
- In short the Legacy Satellite Interface will be upgraded with new translator version from release to release, it will be tested as always, but it will not receive any new development as that focus is now going to be devoted to the TransMagic COMMAND interface.

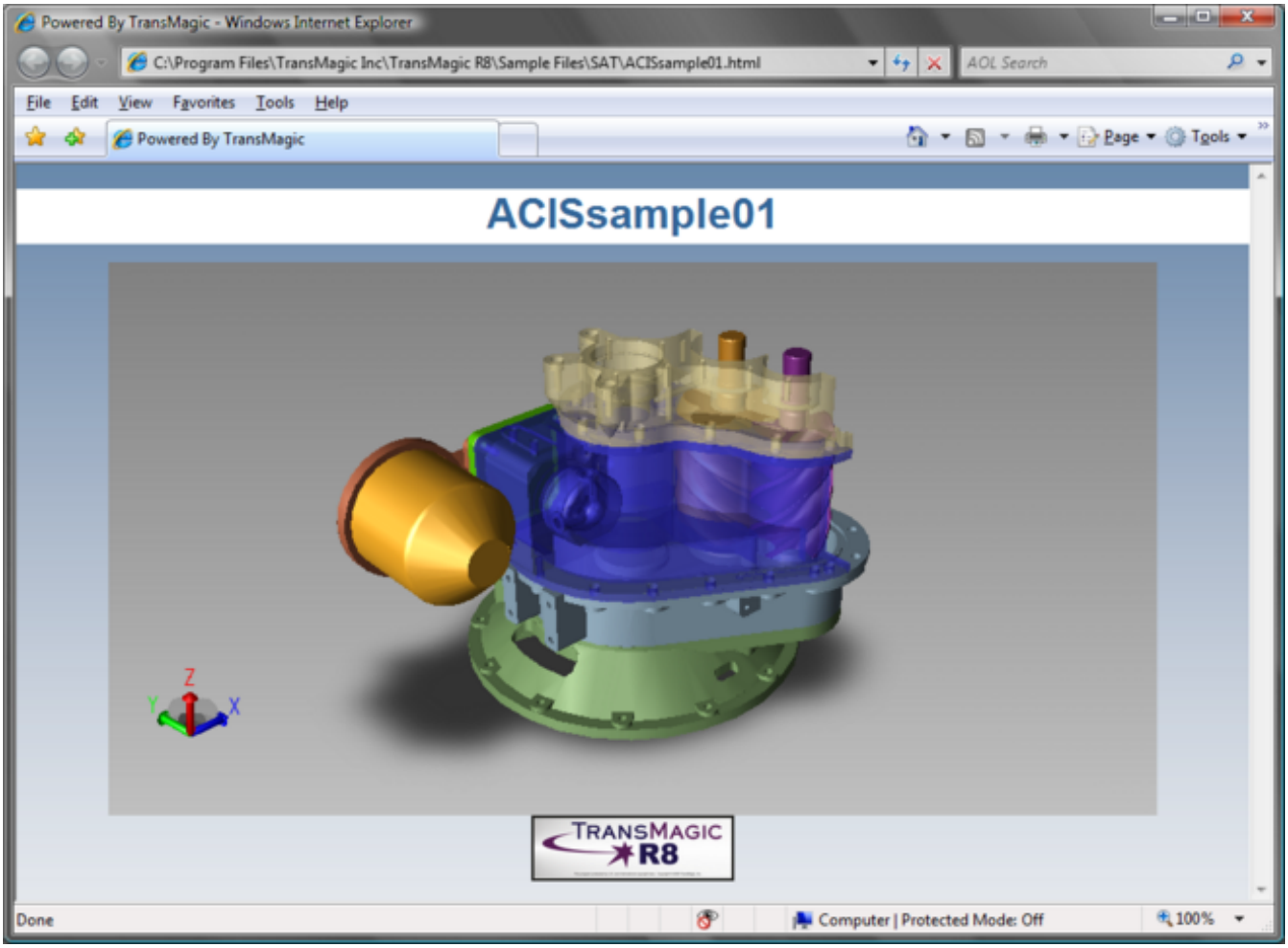
-o-

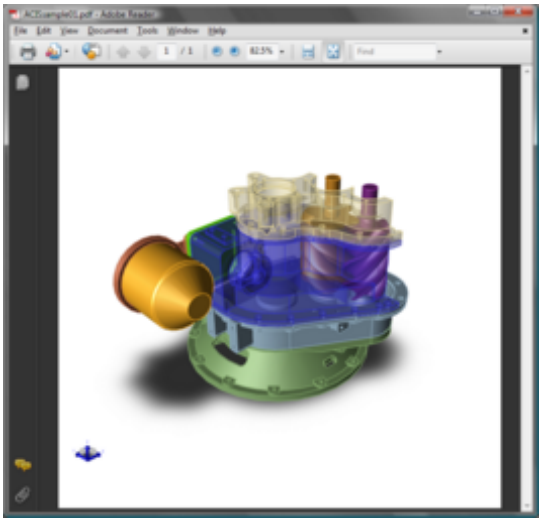
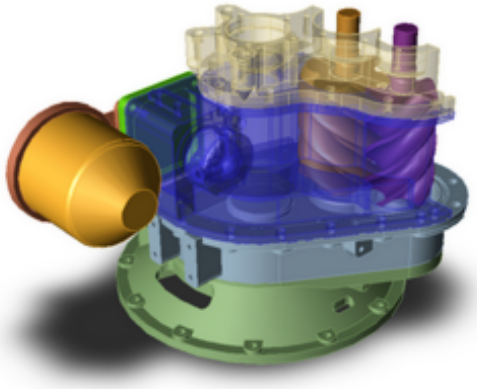
COMMAND Usage

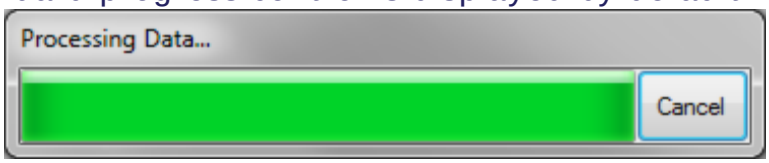
COMMAND Description

The TransMagic COMMAND interface is more than just a command line translator, it is one of the most powerful, flexible and easy to use Engineering Data Exchange applications in the world. TransMagic COMMAND offers the following benefits:

- Callable from any programming language that can run a Windows executable such as C#, VB, HTML, Python, C++, LISP, etc., etc., etc.
- Any supported CAD/CAM/CAE/Viz format in, all supported formats out in one single command.
- Automated and staged geometric Repair.
    - With TM COMMAND Lite and Full Repair can be run successively.
- Simple yet powerful PLM/PDM centric outputs such as:
    - 3D HTML View. This is a full 3D viewer that is launchable in Internet Explorer by loading a simple *.html file. This includes standard 3D viewing controls pan, zoom, rotate, window and more. It also includes advanced controls such as background color, shadows, shaded view, wireframe view and hidden line view. Users can even export additional 3D Viz files from this viewer.
    - HTML View Example:

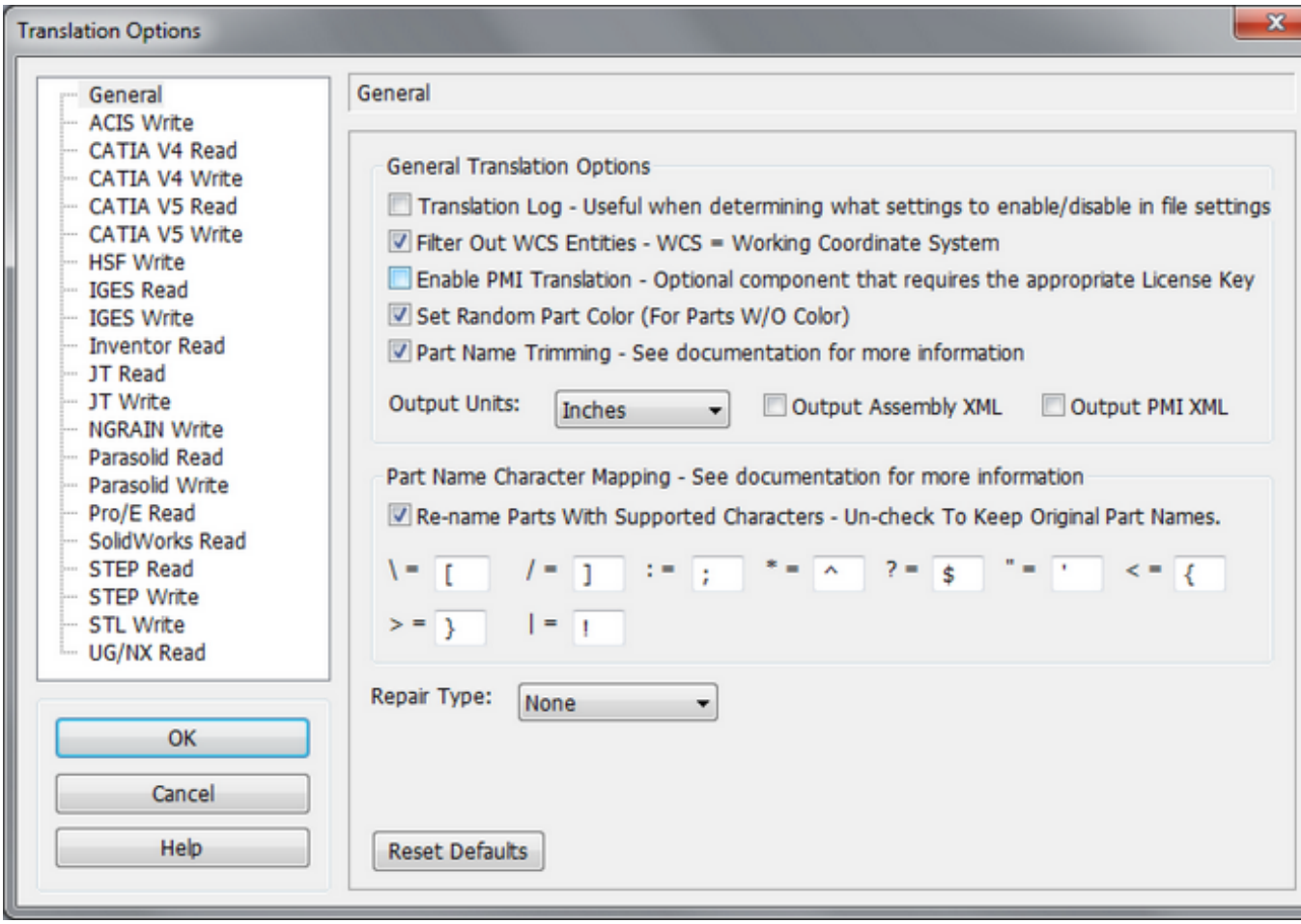
- 

- Advanced image output. This includes PDF, TIFF, EMF & PS. Image size and resolution can be specified to provide high-res images or simple thumbnails.
- PDF View and 256x256 Thumbnail Example:
- 

- All of these formats can be output with the engineering data in one single command.
- Advanced Progress Control:

- A Standard progress control is displayed by default:
  - 

  - See the GUI skinning options to change the progress meter style.
- Or serial progress file is output to the %temp% directory that follows the naming convention: <filename>.prog
  - If you are implementing a server process or have some other interface such as an HTML dashboard, etc. where actual progress dialogs don't apply you can always ping this progress file and update your own progress interface.

- Advanced GUI Options Interface. This is one of the most significant features of TransMagic COMMAND. TransMagic's industry knowledge is captured in it's default translator settings (flavors) and these successfully and continually address the majority of all user cases. However, sometimes these options need to be adjusted for one reason or another. Instead of implementing the hundreds of potential translator options yourself you can send in the simple command: TMCmd -optiondlg
  - This launched the TransMagic Options Dialog which is a very high-level sophisticated option setting interface which enables your users to tweak the options settings saving you untold development effort:



- Of course as TransMagic adds new translators and options, this dialog will always reflect these changes.
- See the GUI skinning options to change the options dialog style.

-o-

Basic COMMAND Syntax

The following is the basic TM COMMAND syntax:

TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td>tmcmdopts</td><td>Options specific to the TM COMMAND program. These options are not translator specific. For TM COMMAND specific options see the TMCmd Options topic.</td></tr>
<tr><td>globalopts</td><td>Global options are options that apply to the incoming file after it has been read "in" and thus affect all output files. For specific global options see the Advanced COMMAND Syntax heading.</td></tr>
<tr><td>inputpath</td><td>The full path to the to the input file. This is a required parameter.

Note: The inputpath and -of<outputspec> parameters are the only required parameters. No inopts or outopts are required.</td></tr>
<tr><td>-of<outputspec></td><td>-of = Output Format. Sending the -of parameter should be followed by the desired output file extension. There are three different ways to send the output format:

• Extension Only. For example: -ofCATProduct. Sending the format using this method has the following effects:
  ○ The CATProduct assembly will be created in the same directory as the input file.
  ○ The CATProduct assembly file will have the same name as the input file.
• Name + Extension. For example: -of"TestFile.CATProduct". Sending the option this way allows you to specify a new file name for the file. The file will also be created in the same directory as the input file.
• Path + Name + Extension. For example: -of"c:\test files\TestFile.CATProduct". Sending the option this way allows you to specify a different output directory than that of the input file.

The allowable output types are:

<table>
<tr><th>outputspec</th><th>Description</th></tr>
<tr><td>model</td><td>CATIA V4 file</td></tr>
<tr><td>CATProduct, CATPart, cgr</td><td>CATIA V5 files</td></tr>
<tr><td>hmf, hsf, asf</td><td>HOOPS files</td></tr>
<tr><td>igs, iges</td><td>IGES files</td></tr>
<tr><td>jt, j_t</td><td>JT files</td></tr>
<tr><td>3ko, ngw</td><td>NGRAIN files</td></tr>
<tr><td>x_t, x_b, xmt_txt, xmt_bin</td><td>Parasolid files</td></tr>
<tr><td>sat, sab</td><td>ACIS part files</td></tr>
<tr><td>asat, asab</td><td>ACIS assembly files</td></tr>
</table>
</td></tr>
</table>

	<table><tr><td>stp, step</td><td>STEP files</td></tr><tr><td>stl</td><td>Stereo Lithography files</td></tr><tr><td>html, obj, ply</td><td>3D visualization files</td></tr><tr><td>tif, ps, emf, pdf</td><td>2D visualization files</td></tr></table> Note: The inputpath and -of<outputspec> parameters are the only required parameters. No globalopts or outopts are required.
outopts	Output specific format options. See the Translator Specific Options heading for output format specific options.

Basic usage example:

TMCmd "C:\Program Files\TransMagic Inc\TransMagic RX\Sample Files\CATIA V5\V5Sample01.CATPart" -ofjt

- Sending this command would: Read the file "V5Sample01.CATPart" from the directory "C:\Program Files\TransMagic Inc\TransMagic RX\Sample Files\CATIA V5" and output a file named "V5Sample01.jt" into that same directory.
- You can actually enter this exact command into Start->Run(XP) or WinKey+R (Vista). Edit "TransMagic RX" to reflect your version, i.e. "TransMagic R8", etc.

As mentioned in the Description topic, TransMagic COMMAND offers much more flexibility and many more options. For more examples and advanced options see the Advanced COMMAND Syntax topic.

-o-

Advanced COMMAND Syntax

Even if you don't send Translator Specific Options, TM COMMAND itself has several advanced options you may wish to use:

TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

[tmcmdopts]:
These options apply to TMCmd itself and should be called directly after the "TMCmd" command.:

Parameter	Description
TMCmd	Simply sending the TMCmd command by itself with no parameters will print to the command window the various options available to you. They won't be as descriptive as this help document but they are useful as a reminder.
-h[type]	-h = Help. Sending the -h parameter followed by the "outputspec" will print to the command window the various options available for that specific output type. Example: -hCATProduct The allowable types are any supported by the "outputspec" option.
>	> = Redirection Parameter. The > parameter is actually a Windows command line parameter that redirects the command window output to a disk based log file. This option breaks the convention for "tmcmdopts" in that it should be used at the very end of a command string. It is followed by a path string that specifies the output log file. For example: TMCmd "C:\Input Files\Test.CATPart" -ofstp >"C:\Input Files\Output.txt" This would have the effect of translating the above file from CATIA V5 to STEP and outputting both the STEP file and the "Output.txt" file that contains the command window output to the "C:\Input Files" directory. The reason this option breaks the "tmcmdopts" convention is that it is a Windows command line parameter and not actually a TMCmd option. There are many additional options available for Redirection. For more information see the following: http://www.robvanderwoude.com/redirection.php
	Advanced Licensing Options If desired the TransMagic GUI licensing dialogs can be bypassed by using the following options. These options apply to Stand-Alone (Node-Locked) licenses only. Network (Floating) licenses require the use of the TransMagic GUI license dialogs. Using these options, the task of licensing can be automated. Contact your TransMagic technical contact for more information.

-getrefcode	getrefcode = Get Reference Code. Sending this option will output the current Reference Code. The Reference Code is a machine signature that is required to generate a TransMagic License Key. This option can be used in conjunction with the redirection parameter to output the Reference Code to a file for the purposes of automation. For example: TMCmd -getrefcode >%TEMP%\LicenseFile.log Sending this command will output the Reference Code to a file named "LicenseFile.log" to the user's "TEMP" directory.
-setkey<License Key>	setkey = Set License Key. You can submit the Reference Code on-line to the TransMagic website or via some other means if you have a special agreement with TransMagic, Inc. Example usage is as follows: TMCmd -setkey3015-6161-9717-7385-3843-0363-5581-3122-3379-9979 If you want to upgrade an existing license then this option can be sent any number of times to overwrite the existing License Key with a new one.
-delkey	delkey = Delete License Key. This option will delete the existing License Key from the system. This option is useful if for some reason a License Key becomes corrupt or otherwise invalid, sometimes the best approach is to simply delete the existing License Key and create a new one using the -getrefcode and -setkey options. Note that when a License Key has been deleted the Reference Code will change and the previous Reference Code cannot be used again. The new Reference Code must be used to generate a new License Key.
-genlicinfofile	genlicinfofile = Generate License Info File. This option will generate a file to the users %TEMP% directory that provides information about the license, if any, that is currently registered with the system. The name of the file is "tm_licprops.lic", it's a text file and it has the form: TransMagic License Properties Version: 9.00.000 Expiration Date: 4/13/2020 Translation Units: Undefined Configuration: TransMagic Expert + JT RW + TM COMMAND Interface Type: Node-Locked Where: • Version is the version of TransMagic installed. The leading number is the major version, i.e. R8, R9, etc., the middle number is the service pack, i.e. sp1, sp2, etc. The trailing number is currently unused. • Expiration date is the expiration date of the current license. This applies to both Node-Locked or Floating. If No license has been registered with the system yet then this will read: Undefined. • Translation Units is for a special "Units" based license which indicates the number of output translations left. This license type is rarely used. • Configuration is the type of license registered with the system. The formats supported with each configuration are listed on-line. • Type is either Node-Locked (stand-alone) or Floating (network) depending on the type of license registered with the system.

[globalopts]:
Global options apply to the incoming file after it has been read in and thus affect all output files. These options should be applied in the command line string directly after the "tmcmdopts" and before "inputpath":

Parameter	Description
-od<directory>	-od = Output Directory. Sending the -od parameter followed by a directory path in quotes specifies an output directory in which to place the output file or files. This option is handy if you are specifying multiple output formats using just the output extensions but want to put them all in the same specified directory. If you wanted to specify multiple output directories when translating multiple files then you would simply specify the path in the -of<outputspec> option for each output file. You would not use this option in this case. This option will always override individual paths. For path strings with spaces you should put quotes "" around the path. Example: -od"C:\CAD Files" In fact, it doesn't hurt to always use quotes. You can also send paths such as environment variables. For example the user "temp" directory is quite common. Example: -od%TEMP% or -od"%TEMP%" both work the same.
-otd	-otd = Output Type Directories. Sending the -otd option will create sub-directories named for the various output types being created. TMCmd can output multiple output types simultaneously and when this is done it is sometimes desirable to put these output files in their own folders. These folders will be created under the -od directory if specified or under the input file's directory if -od is not specified. See the "Advanced usage examples" below for more information.
-[no]pm	-pm = Progress Meter. This refers to the progress meter dialog that users can see and visually monitor. By default a progress meter dialog is on so, even though it exists, sending the -pm parameter to TMCmd does not have a practical use. However, sending the -nopm parameter will turn the progress meter off. TMCmd is very useful for PLM/PDM type integrations that are installed on a server and in these cases turning off visual progress reporting is a necessity. If you are implementing a PLM/PDM type of TMCmd integration, please note that TMCmd always creates a serial progress file that is output to the user "temp" directory. The naming convention of the file is: <filename>.prog The contents of the file is a simple string value from 1 to 100. Any application can ping this file for it's contents and update their own progress meter.
-tl	-tl = Translation Log. Sending the -tl option will tell TMCmd to process a translation log file. This log file is always output to the user "temp" directory. The naming convention of the file is: <filename>_read.log or

	<filename>_write.log
-ipm	-ipm = Input Progress Message. TM COMMAND is often used as an OEM translator into or out of enginnering systems. At TransMagic both our Inventor and SolidWorks Add-Ins call TM COMMAND to translate files. This means there is often an intermediate file that is used to bring data into or out of the engineering system. For example in TransMagic's own SolidWorks Add-In we use the SolidWorks native Parasolid kernel format *.x_t. So instead of having the progress message read "Reading PSFile.x_t, Please Wait..." we send in the -ipm and set the progress message to "Processing SolidWorks Geometry, Please Wait..." as it's more clear to the end user.
-opm	-opm = Output Progress Message. TM COMMAND is often used as an OEM translator into or out of enginnering systems. At TransMagic both our Inventor and SolidWorks Add-Ins call TM COMMAND to translate files. This means there is often an intermediate file that is used to bring data into or out of the engineering system. For example in TransMagic's own SolidWorks Add-In we use the SolidWorks native Parasolid kernel format *.x_t. So instead of having the progress message read "Writing PSFile.x_t, Please Wait..." we send in the -opm and set the progress message to "Processing SolidWorks Geometry, Please Wait..." as it's more clear to the end user.
-dlgmsg	-dlgmsg = Dialog Messaging. This option which tells TM COMMAND to output error dialog messaging for any errors encountered. This option is off by default. It is convenient for OEM partners implementing TM COMMAND in an end-user GUI as it simplifies error trapping and displays meaningful error messaging to the end-user.
-lang<Language>	-lang = Language. By default the language will be the language of the OS. If the language of the OS is not supported then the default language will be English. However the default language can be overridden by sending the -lang switch followed by the language parameter. Allowed choices are: • eng = English • deu = Deutsch • fra = French • jpn = Japanese • esp = Spanish • ita = Italian • ptb = Portuguese Default Setting: Language of the OS.
-xmlasm	-xmlasm = XML Assembly File. Sending this command line option will break apart an assembly into individual files with an XML file that contains the assembly structure and part transform matrices.
-xmlatom	-xmlatom = XML Atomize. This command is designed to be sent along with the -xmlasm command above. The atomize operation will further break down part files that contain multiple bodies into discretized part files that each contain only one single body per file. Following is a break-down of the behaviors of this option: • For part files with multiple bodies, this option will instruct TMCmd to create individual files for each body in the part. Each new file created will be named the same as the "top-level" part file appended with -01.*, -02.*, -03.*, and so on. • You can currently specify -xmlasm even for a non-assembly or flattened file such as SAT. The current behavior is that a single XML file will get created

	along with the single output file format. If the -xmlatom option is specified in conjuction with the -xmlasm option then TMCmd will create the -01.*, -02.*, -03.* files as above for each body in the file. • If a part file contains only a single body then TMCmd will create that single file with no appended -01.*, -02.*... • When the -xmlatom option is specified, add a new sub-level "Body", "/Body" is added to the XML assembly file which lists the body by both the appended file name with -01.*, -02.*, etc. and also by the actual attribute name of the body in that file.
-xmlbbox	-xmlbbox = XML Bounding Box. This command is designed to be sent along with the -xmlasm command above. The -xmlbbox command will calculate the minimum Bounding Box size relative to the global origin for each part (or body) in a file. It will create a new sub-category in the XML file: Bounding_Box
-xmlmass	-xmlmass = XML Mass Properties. This command is designed to be sent along with the -xmlasm command above. The -xmlmass command will calculate the the mass properties for each part (or body) in a file including Volume, Center Of Mass, Principal Axes, Inertia Tensor and Principal Moments. It will create a new sub-category in the XML file: Mass
-xmlsurf	-xmlsurf = XML Surface Area. This command is designed to be sent along with the -xmlasm command above. The -xmlsurf command will calculate the surface area for each part (or body) in a file. It will create a new sub-category in the XML file: Surface_Area

Advanced usage examples:

Multiple output formats:

TMCmd "C:\Program Files\TransMagic Inc\TransMagic RX\Sample Files\CATIA V5\V5Sample01.CATPart" -ofmodel -ofigs -ofjt -of3ko -ofx_t -ofsat -ofstp -ofstl -ofhsf

- Sending this command would: Read the file "V5Sample01.CATPart" from the directory "C:\Program Files\TransMagic Inc\TransMagic RX\Sample Files\CATIA V5" and output into that same directory files named:
  - V5Sample01.model
  - V5Sample01.igs
  - V5Sample01.jt
  - V5Sample01.3ko
  - V5Sample01.x_t
  - V5Sample01.sat
  - V5Sample01.stp
  - V5Sample01.stl
  - V5Sample01.hsf

Multiple output formats into multiple directories:

TMCmd -od"%TEMP%" -otd "C:\Program Files\TransMagic Inc\TransMagic RX\Sample Files\CATIA V5\V5Sample01.CATPart" -ofmodel -ofigs -ofjt -of3ko -ofx_t -ofsat -ofstp -ofstl -ofhsf

- Sending this command would: Read the file "V5Sample01.CATPart" from the directory "C:\Program Files\TransMagic Inc\TransMagic RX\Sample Files\CATIA V5" and output

translated files into the user "temp" directory and also create sub-directories named after the outputspec parameter. The output for this command would create the following files and directories:

- %TEMP%\CATIA V4\V5Sample01.model
- %TEMP%\IGES\V5Sample01.igs
- %TEMP%\JT\V5Sample01.jt
- %TEMP%\NGRAIN\V5Sample01.3ko
- %TEMP%\Parasolid\V5Sample01.x_t
- %TEMP%\ACIS\V5Sample01.sat
- %TEMP%\STEP\V5Sample01.stp
- %TEMP%\STL\V5Sample01.stl
- %TEMP%\HOOPS\V5Sample01.hsf

Multiple output formats into multiple directories with options for "globalopts" and "outopts":

For this example we're going to specify the reading and writing of "free curves" using the -fc option and "free points" using the -fp option. For the input translation the -fc and -fp option only needs to be set once. However, for each output format (only geometric formats support these options), the -fc and -fp option need to be set each time. This offers the ultimate in flexibility for specifying output options for each format. The command would appear as follows:

TMCmd -od"%TEMP%" -otd "C:\Program Files\TransMagic Inc\TransMagic RX\Sample Files\CATIA V5\V5Sample01.CATPart" -ofmodel -fc -fp -ofigs -fc -fp -ofjt -fc -fp -ofx_t -fc -fp -ofsat -fc -fp -ofstp

- Sending this command would: Read the file "V5Sample01.CATPart" from the directory "C:\Program Files\TransMagic Inc\TransMagic RX\Sample Files\CATIA V5" and output translated files into the user "temp" directory and also create sub-directories named after the outputspec parameter. The output for this command would create the following files and directories:
  - %TEMP%\CATIA V4\V5Sample01.model (with free curves and free points saved to the file)
  - %TEMP%\IGES\V5Sample01.igs (with free curves and free points saved to the file)
  - %TEMP%\Parasolid\V5Sample01.x_t (with free curves and free points saved to the file)
  - %TEMP%\ACIS\V5Sample01.sat (with free curves and free points saved to the file)
  - %TEMP%\STEP\V5Sample01.stp (no free curves or free points will be saved to the file - notice -fc & -fp weren't specified)

-o-

COMMAND GUI Elements

TransMagic COMMAND has some important GUI elements that can be launched at the command line that you may wish to use:

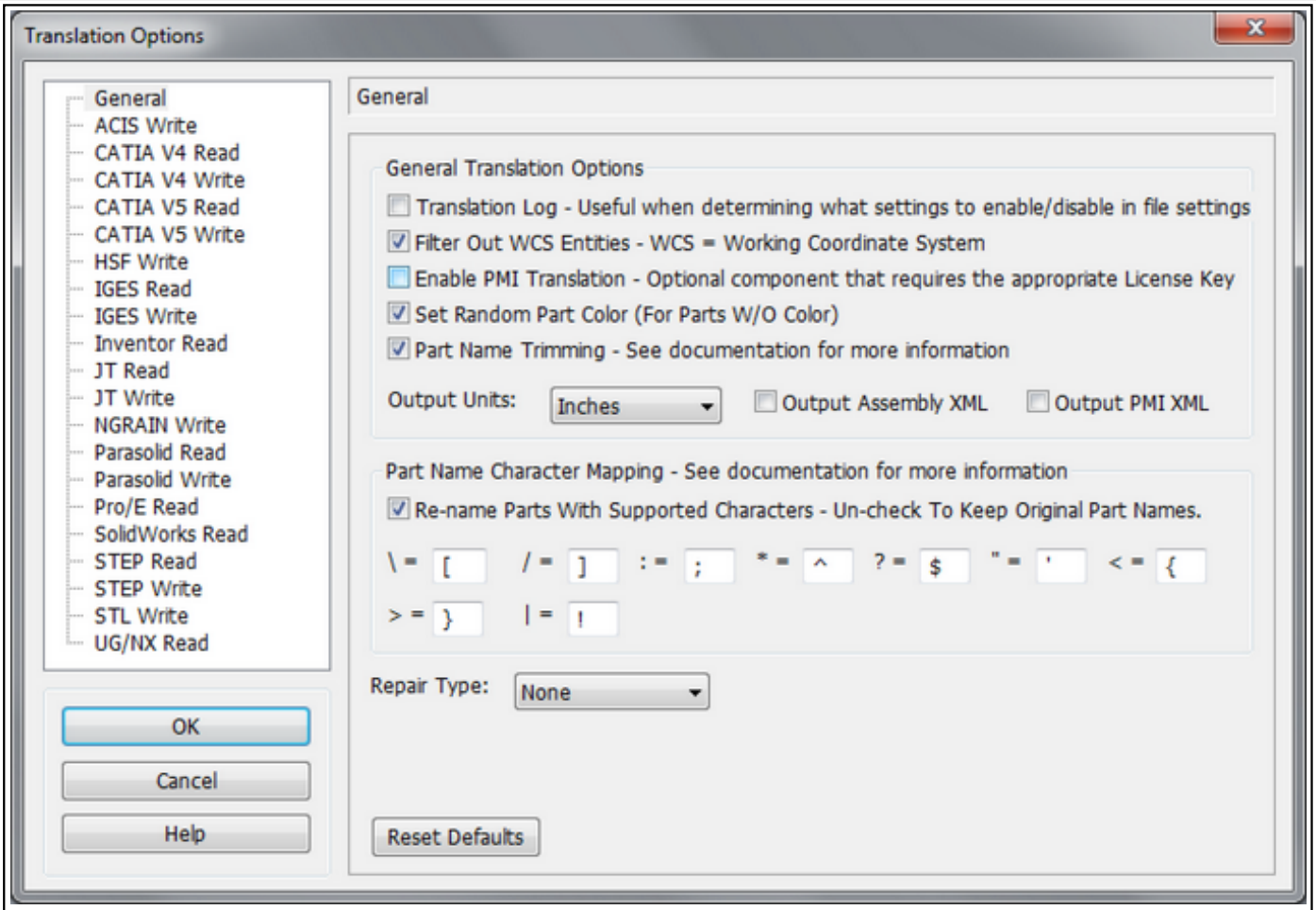
TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

TMCmd GUI Elements:

Parameter	Description
TMCmd	If TM COMMAND has just been installed on the user's system either by itself or with TransMagic, it will need to be licensed before it can be used. Simply sending the command "TMCmd" by itself or with any parameters will launch the TransMagic licensing system if no license has yet been established. You don't technically need to account for this in your development. If through some user action TMCmd is launched for any reason then the user will be prompted to go through the TransMagic license protocol to establish their license. As soon as the License Key is entered the translation will continue. The TransMagic licensing protocol is automatic and again, there is no need to account for this in your development, but for informational purposes the licensing course as experienced by the end-user is as follows: 1) No TransMagic license found, the "Register TransMagic" dialog is launched to inform the user and prompts user to click the "Register" Button:  2) The user clicks "Register", this launches the "License Key" dialog and prompts the user to follow

	the 2 licensing steps to obtain a License Key. User enters his License Key, clicks "OK" and translation (etc.) continues.... 
-optiondlg <page>	-optiondlg = Translation Options Dialog. One of the most significant benefits of using TMCmd is how incredibly simple it makes the complex task of "tweaking" translations. When TMCmd is installed it populates the Windows Registry with "default settings" for all options and translators. See the Translator Specific Options heading for these defaults. The most important aspect of these default settings is that they encapsulate literally tens of man-years of research and development by the translation experts at TransMagic. We refer to these default settings as the "LCD Settings" where LCD = Lowest Common Denominator. These LCD Settings address the 80 to 90 percentile of all translations. It is very likely you or your user will never have to touch these settings. However, there may come a time when a user needs to "tweak" a setting or two. For example, a common need would be to save out an earlier version of a file format, i.e. say the user needed to set the CATIA V5 output to CATIA V5 R15. To give the user access to the TransMagic Options dialog simply send the command: TMCmd -optiondlg This will launch the TransMagic Options Dialog:



Make note of a few things with this dialog:

- In addition to the encapsulation of TransMagic's wealth of industry knowledge, there are literally hundreds of options and scenarios that would require more man-years of development just to create and perfect a dialog such as this.
- All options are described in layman's terms. In suit with TransMagic's philosophy of making translation simpler for everyone including and especially non-engineering users, all options are briefly described in the dialog itself. However, notice in the lower left corner of the dialog, the "Help" button. This dialog also includes full documentation that exhaustively describes the cause and effect of every single option setting, again, in as simple terms possible. This again saves countless hours of development and perfecting the message for the TMCmd implementer.
- These options are saved to the specific user accounts in the Windows Registry and user changes are saved only for that user.

Once the user sets their desired option and clicks "OK", these changes are then saved to the Windows Registry and these changes now become the default settings for any successive translations. There's no need for the TMCmd developer to send Translator Specific Options options to the command line EVER if the Translation Options dialog is used.

Note: Any default setting/option can be over-ridden by sending the appropriate option in the command line. For example: Even though the "Translation Log" is shown unchecked (above) by default - you could send the -tl option in the command line and a Translation Log would be generated.

Page Modifier

Another convenient setting for the -optiondlg parameter is the page modifier. This allows you to open the option dialog to a specific page in the option dialog. For example, you could send the command:

-optiondlgv5write

Sending this option would launch the Translation Options dialog with only the CATIA V5 Write page specifically. You can also open multiple specific pages. For example, you could send the command:

-optiondlgigesread -optiondlgigeswrite

To open both the IGES Read and Write option pages only.

Acceptable page names are:

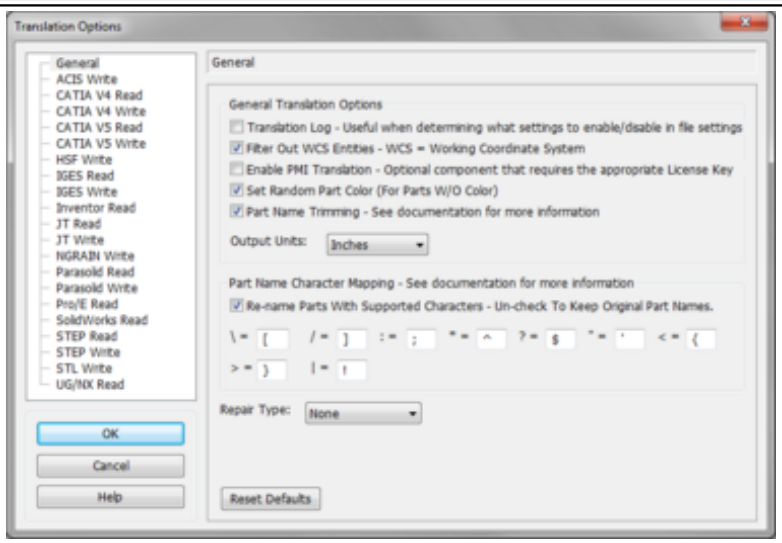
Modifier	Description
aciswrite	ACIS Write Options
v4read	CATIA V4 Read Options
v4write	CATIA V4 Write Options
v5read	CATIA V5 Read Options
v5write	CATIA V5 Write Options
hsfwrite	HSF Write Options
igesread	IGES Read Options
igeswrite	IGES Write Options
invread	Inventor Read Options
jtread	JT Read Options
jtwrite	JT Write Options
ngrainwrite	NGRAIN Write Options
ppsread	Parasolid Read Options
pswrite	Parasolid Write Options
proeread	Pro/E Read Options
swread	SolidWorks Read Options
stepread	STEP Read Options
stepwrite	STEP Write Options
stlwrite	STL Write Options
ugread	UG/NX Read Options

-skin <val>

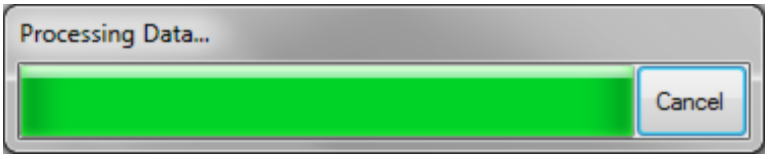
-skin = GUI Skin. This option allows you to apply several different styles to the option dialog or progress meter to more closely match your interface style. Allowed choices are:

- 0 (-skin0) = System default, no skin.

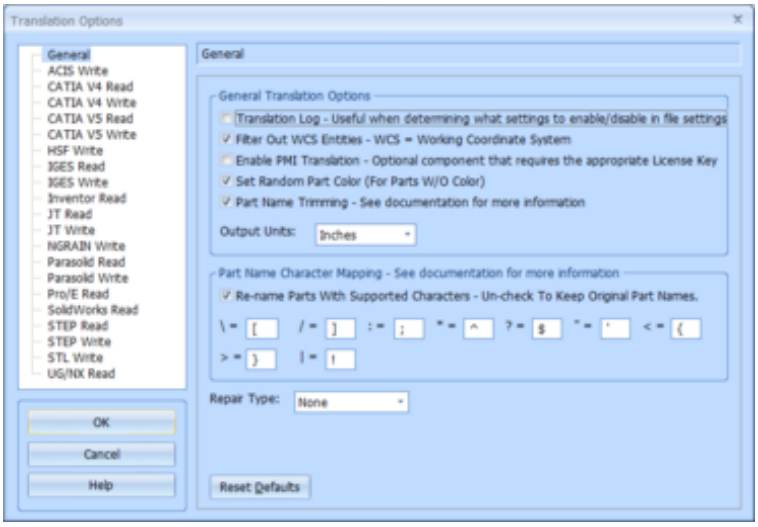
- Options Dialog:



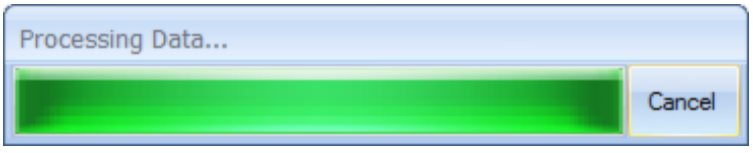
- Progress Meter:



- 1 (-skin1) = Blue skin.
  - Options Dialog:

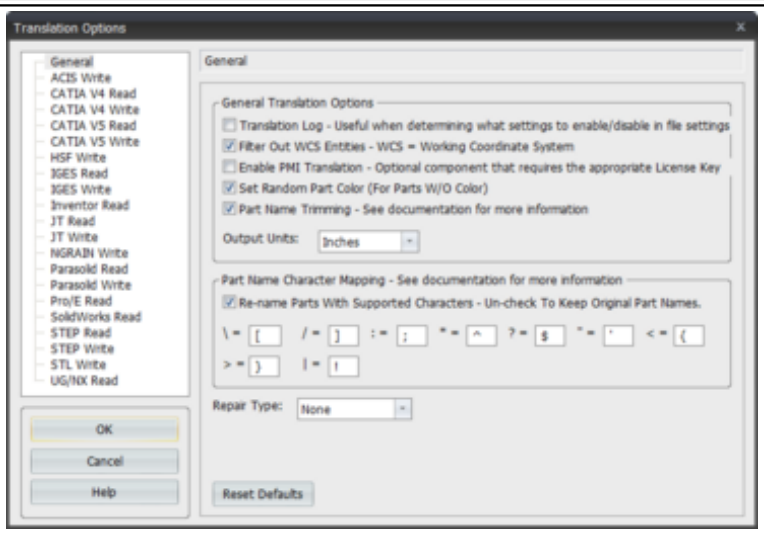


  - Progress Meter:

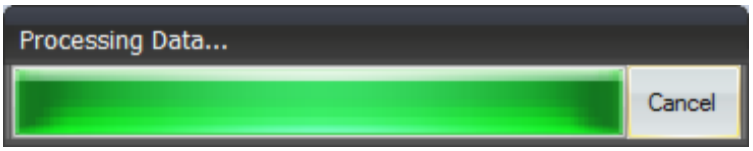


- 2 (-skin2) = Black skin.

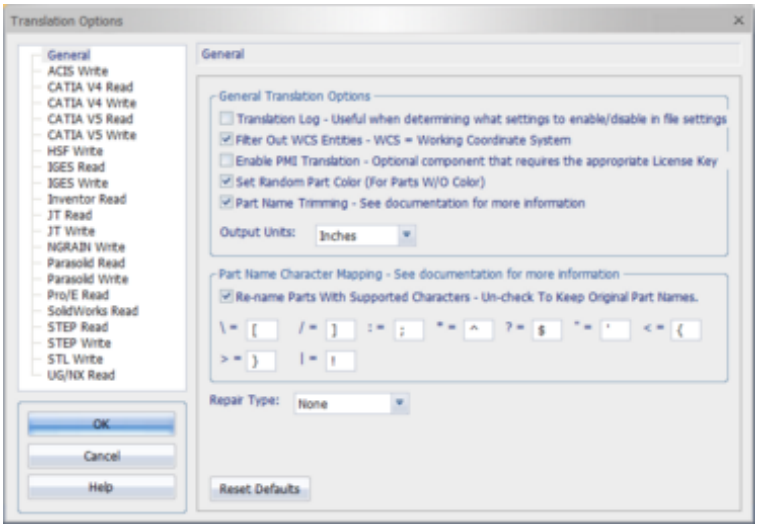
- Options Dialog:



- Progress Meter:

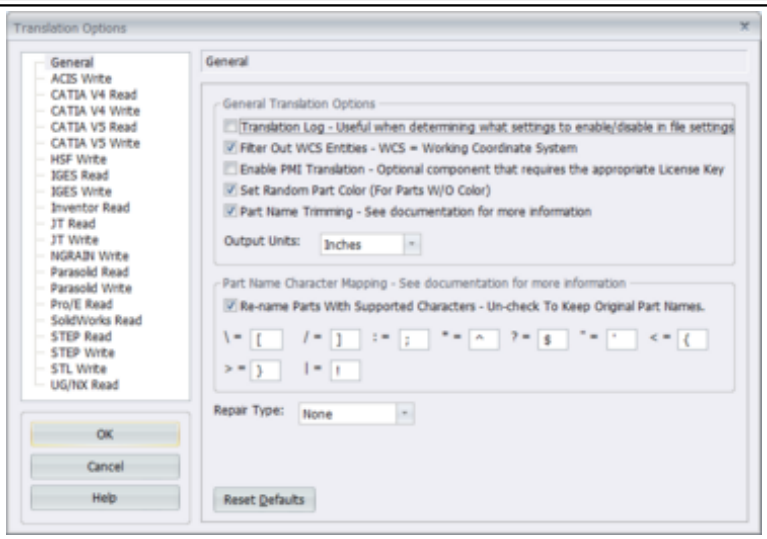


- 3 (-skin3) = Aqua skin.
  - Options Dialog:



  - Progress Meter:

- 4 (-skin4) = Silver skin.

- Options Dialog: 

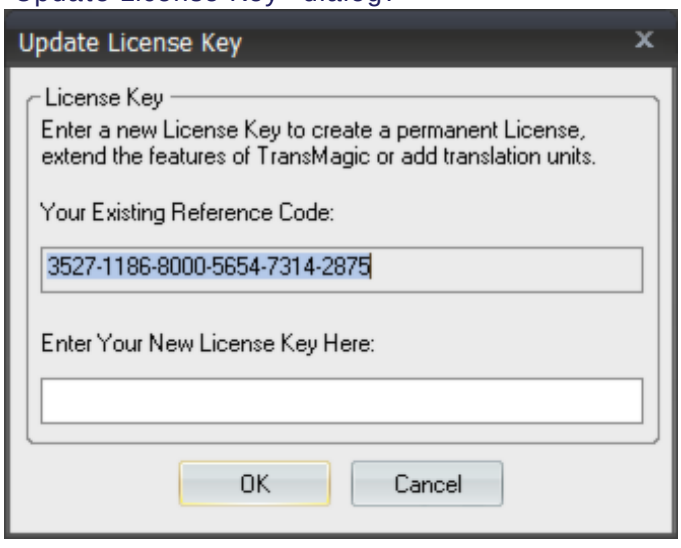
- Progress Meter: 

- 5 (-skin5) = Same skin as specified by TransMagic settings in the registry. This depends on what the user has specified as their preference.

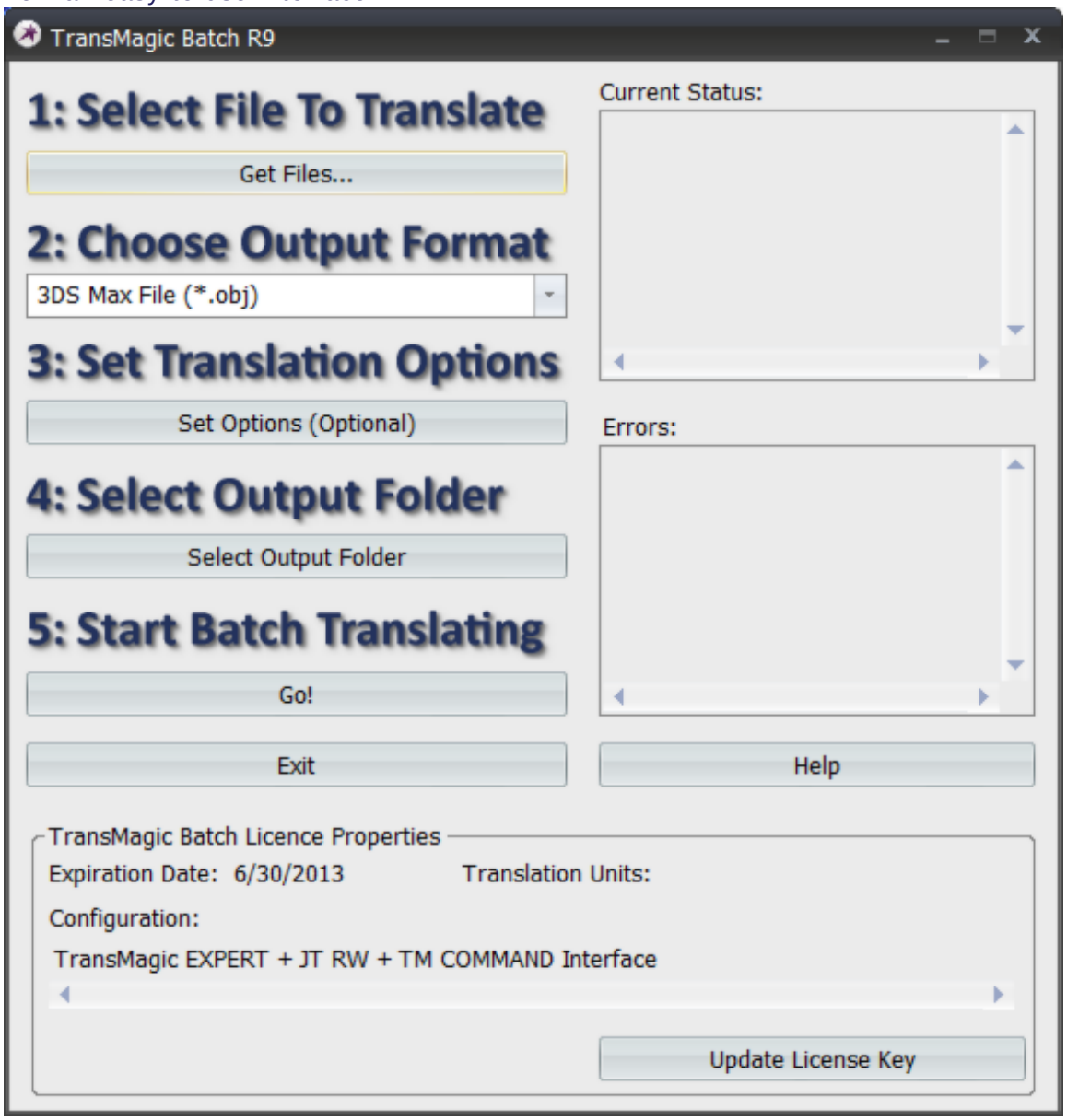
Default Setting: Default skin is "skin0" style which is the system default. If you just want the system default style then just simply don't specify this option.

-update

-update = Update TransMagic License. TransMagic licenses can be configured in a myriad of ways depending on which option the user purchased. Let's say the user purchased a TransMagic BASIC license, which doesn't include the JT format (among others). They have the option of adding just JT to their TransMagic BASIC license by purchasing this translator from TransMagic or a reseller. In this case their license will need to be updated. When they purchase a new translator or other upgrade, TransMagic will proactively send them a new License Key but they need to enter this new License Key before they can use their new feature. Sending the -update parameter will trigger the "Update License Key" dialog:



They can enter their new License Key in the appropriate box and click "OK". Their license has now been updated.

-batch	-batch = TransMagic Batch. Sending the -batch parameter launch the "TransMagic Batch" Interface. TransMagic Batch is a unique application unto itself and an icon for launching is already installed with a TransMagic installation. It also requires it's own unique license key. Launching TM Batch will prompt the same license course outlined above for TMCmd. Once licensed the TM Batch interface will be launched and as the name implies allow the user to perform batch translation processing from an easy to use interface:  Note: The -batch switch can also be followed by the -skin<val> switch to customize the dialog.

-o-

Documentation Conventions

All TransMagic COMMAND options use the following documentation conventions:

TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Documentation Conventions:

[]	Brackets mean a parameter can be modified with a pre-defined value. A good example of this is the -h (Help) parameter documented below. It is defined as -h[type] which means "type" is a predefined value. See below for more information but if you wanted to print out help for the CATIA V5 translator to the command window you could send the command: TMCmd -hCATPart
-[no]	"no" in brackets means that particular option has an "on/off" value. A good example of this is the -pm option. -pm tells TMCmd to show a translation progress meter (which is on by default anyway). However, -nopm tells TMCmd to turn off the progress meter which will be needed for PLM/PDM type server installations. Here again note the brackets indicating that a pre-defined value should be sent. For options with the "no" modifier, the pre-defined value is always "no".
< >	Greater than/less than characters mean this parameter should be modified with a user-defined value. A good example of this is the -od (output directory) parameter documented below. It is define as -od<directory> where "directory" should be given by the user. For example: -od"C:\CAD Files" would tell TMCmd to output translated files to the "C:\CAD Files" directory.

-o-

Check The TransMagic License Properties

When a TransMagic License is generated on the system several registry entries are created which allow you to check the functionality rights of the TransMagic license. The two primary benefits are that these entries allow you to quickly:

1) Check the license properties such as configuration, expiration date, and whether a license is node-locked or networked.

2) Determine which formats the license on the system has the ability to read or write.

However, there are some other variables that may be useful. The complete list is below.

Notes:

- All of these TransMagic License Property values are stored under the following registry key: HKEY_CURRENT_USER\Software\TransMagic\Licenses\Apps\TransMagic R9\PrimaryLicense
- The "PrimaryLicense" key will NOT exist until a TransMagic license is either generated on the system (node-locked) or obtained from the network (floating).
- This key set gets regenerated every time TMCmd is run, if the TM License Manager is run, or if the license is updated.
- So long as a TransMagic license does exist this key will be created for each user the first time TMCmd is run. In fact if you want to generate this key you can simply call TMCmd with no parameters.
- These values are TransMagic version independent so it is not necessary to keep track of the TransMagic version installed on the system.
- See the sample code for examples on obtaining this information from the registry:
    - C++ | Check TransMagic License Properties
    - VB | Check TransMagic License Properties

TM License Properties:

Registry Entry	Type	Description
TM License Config	REG_SZ	The TransMagic License Key configuration description. This is the base TransMagic License plus any optional components. It follows the following example form: TransMagic EXPERT + JT RW + TM COMMAND Interface
TM License Ver	REG_SZ	The version of TransMagic currently installed on the system. It follows the form: Main Version.Service Pack.Build Number, for example 8.40.000
TM License Exp Date	REG_SZ	The expiration date of the license. It follows the form: Month/Day/Year, for example 6/4/2012
TM License	REG_SZ	The number of translation "units" left. This license type is rarely used

Units		and will usually be "Undefined". It follows the form: XXXX, for example 1000
TM License Type	REG_SZ	The type of license currently in use. It will either be: "Node-Locked" or "Floating"

Format Specific Values:

Registry Entry	Type	Description
Viz3DWriteRight	REG_DWORD	1 = true, 0 = false. The ability to write the currently supported 3D Viz-Rep formats other than HSF, STL or JT which have their own values.
ACISReadRight	REG_DWORD	1 = true, 0 = false. The ability to read the currently supported ACIS formats.
ACISWriteRight	REG_DWORD	1 = true, 0 = false. The ability to write the currently supported ACIS formats.
HSFWriteRight	REG_DWORD	1 = true, 0 = false. The ability to write the currently supported HOOPS formats including HSF, HMF & HTML.
IGESReadRight	REG_DWORD	1 = true, 0 = false. The ability to read the currently supported IGES formats.
IGESWriteRight	REG_DWORD	1 = true, 0 = false. The ability to write the currently supported IGES formats.
JTReadRight	REG_DWORD	1 = true, 0 = false. The ability to read the currently supported JT formats.
JTWriteRight	REG_DWORD	1 = true, 0 = false. The ability to write the currently supported JT formats.
INVReadRight	REG_DWORD	1 = true, 0 = false. The ability to read the currently supported Inventor formats.
PSReadRight	REG_DWORD	1 = true, 0 = false. The ability to read the currently supported Parasolid formats.
PSWriteRight	REG_DWORD	1 = true, 0 = false. The ability to write the currently supported Parasolid formats.
ProEReadRight	REG_DWORD	1 = true, 0 = false. The ability to read the currently supported Pro/E formats.
STEPReadRight	REG_DWORD	1 = true, 0 = false. The ability to read the currently supported STEP formats.
STEPWriteRight	REG_DWORD	1 = true, 0 = false. The ability to write the currently supported STEP formats.
STLWriteRight	REG_DWORD	1 = true, 0 = false. The ability to write the currently supported STL formats.
SWReadRight	REG_DWORD	1 = true, 0 = false. The ability to read the currently supported SolidWorks formats.
TMRReadRight	REG_DWORD	1 = true, 0 = false. The ability to read the currently supported TransMagic formats.
TMRWriteRight	REG_DWORD	1 = true, 0 = false. The ability to write the currently supported TransMagic formats.
UGReadRigh	REG_D	1 = true, 0 = false. The ability to read the currently supported UG\NX

t	WORD	formats.
V4ReadRight	REG_DWORD	1 = true, 0 = false. The ability to read the currently supported CATIA V4 formats.
V4WriteRight	REG_DWORD	1 = true, 0 = false. The ability to write the currently supported CATIA V4 formats.
V5ReadRight	REG_DWORD	1 = true, 0 = false. The ability to read the currently supported CATIA V5 formats.
V5WriteRight	REG_DWORD	1 = true, 0 = false. The ability to write the currently supported CATIA V5 formats.

Other Values:

Registry Entry	Type	Description
TMCmdInterfaceRight	REG_DWORD	1 = true, 0 = false. The ability to call the TransMagic COMMAND interface.
CADInterfaceRight	REG_DWORD	1 = true, 0 = false. The ability to use the TransMagic native CAD Add-Ins.
MagicSurfaceRight	REG_DWORD	1 = true, 0 = false. The ability to use the MagicSURFACE toolkit.
PMIRight	REG_DWORD	1 = true, 0 = false. The ability to read PMI data. Currently PMI data is only read into and stored in the TMR format.
Support64	REG_DWORD	1 = true, 0 = false. The ability to run the product in 64-bit mode.

-o-

Tanslator Specific Options

Common Options

The following options are common to most translations:

TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

Parameter	Read	Write	Description
-[no]at	X	X	-at = Attributes. Translate attributes such as Part Name, Part Layer and Part Color. Default Setting: Attributes are ON by default.
-[no]fc	X	X	-fc = Free Curves. Translate free curve entities, also known as wire-frame entities. Default Setting: Free curves are OFF by default on Read and ON by default on Write. The theory being if the desire is to read curves, then it is more likely that the desire is also to write curves out to other formats. If no curves are read in by default anyway, then no curves are written out even if the writing of curves is turned ON.
-[no]fp	X	X	-fc = Free Points. Translate free point entities, also known as vertex entities. Default Setting: Free points are OFF by default on Read and ON by default on Write. The theory being if the desire is to read points, then it is more likely that the desire is also to write points out to other formats. If no points are read in by default anyway, then no points are written out even if the writing of points is turned ON.
-[no]fs	X	X	-fs = Free Surfaces. Translate free surfaces. Default Setting: Free surfaces are ON by default for most formats but not all. Refer to the -optiondlg Help Docs for specific format settings.
-[no]eh	X	X	-eh = Hidden Entities. Translate hidden entities, also known as blanked entities. Default Setting: Hidden Entities are OFF by default on Read and ON by default on Write. The theory being if the desire is to read hidden entities, then it is more likely that the desire is also to write hidden entities out to other formats. If no hidden entities are read in by default anyway, then no hidden entities are written out even if the

			writing of hidden entities is turned ON.
-[no]wp	X	X	-wp = Work Planes. Translate work planes. Work planes a typically geometric reference plane. They usually represent the typical XY, XZ & YZ planes but could also be user defined. Default Setting: Work planes are OFF by default.
-[no]flatten	X	X	-flatten = Flatten Assembly Structure. Use this option if you want to flatten the assembly structure either on the incoming assembly or outgoing assembly. Flattening an assembly will not change the position of the parts in space. It will simply flatten the assembly structure itself with the result being a simple list of parts. Flattening an assembly could result in file size growth as it will also cause any instanced components to become individual parts. For example, if you had 1 bolt copied in 1000 locations, in an assembly structure it's simply one geometric bolt shape and 1000 positions where this bolt is referenced. After flattening an assembly you would then have 1000 geometric bolts. Default Setting: Flatten is OFF by default.
-[no]se	X		-se = Read Suppressed Entities. Default Setting: Default is not to Read Suppressed Entities.
-[no]wcs	X		-wcs = Working Coordinate Systems. Working coordinate systems are typically reference entities that indicate local coordinate systems for sub-assemblies and parts. They are non-geometric entities and are not supported by all output formats. However, if they are supported they will be written out to those formats. Default Setting: Working Coordinate Systems are ON by default.
-lr -norepair	X		-lr = Lite Repair. Apply Lite Repair to incoming geometry. Refer to the -optiondlg Help Docs for a complete description of Lite Repair. Lite Repair and Full Repair can be run successively by choosing the "-lfr" option. Note: As there are three repair option switches available in order to override any one of them and disable anyone of them use the switch: -norepair Default Setting: Lite Repair is OFF by default.
-fr -norepair	X		-fr = Full Repair. Apply Full Repair to incoming geometry. Refer to the -optiondlg Help Docs for a complete description of Full Repair. Lite Repair and Full Repair can be run successively by choosing the "-lfr" option. Note: As there are three repair option switches available in order to override any one of them and disable anyone of them use the switch: -norepair Default Setting: Full Repair is OFF by default.
-lfr -norepair	X		-lfr = Both Lite AND Full Repair. Apply Both Lite and Full Repair successively to incoming geometry. Refer to the -optiondlg Help Docs for a complete description of Lite and Full Repair.

			Note: As there are three repair option switches available in order to override any one of them and disable anyone of them use the switch: -norepair Default Setting: Lite and Full Repair is OFF by default.
-cm<mapchar><string>	X		cm = Part Name Character Mapping. Many CAD systems will give automatic individual part names to each part in a file if the user did not explicitly do so. For example, let's say you have a CATIA V4 file with 3 solid parts in it. CATIA V4 will automatically name the parts *SOL1, *SOL2, *SOL3. Now let's say you want to translate this CATIA V4 file with the 3 parts in it into a Parasolid file so you can bring the file into SolidWorks. SolidWorks will create a 3 part assembly from this file. Each individual part will be saved as a *.sldprt file automatically upon reading into SolidWorks. SolidWorks and presumably other CAD systems that operate in this way will create a file name based on the part name. So when SolidWorks automatically tries to save a file with a name of *SOL.sldprt, Windows will not allow this because "*" is an illegal file name character - as well as "\/:*?"<>|". None of these characters can be used in a file name. To avoid this scenario send the -cm option. Sending just the -cm option will result in a default mapping of the following characters: • \ = [• / =] • : = ; • * = ^ • ? = $ • " = ' • < = { • > = } • | = ! If your CAD system does not save individual files by part names then you don't have to worry about setting this option. If you wish to not use the default TMCmd mappings you can substitute your own mappings by sending -cm<mapchar><string> where <mapchar> is one of the illegal Windows characters and <string> is it's substitution character. Example: -cm"?%" Sending this option would tell TMCmd to map the illegal Windows character '?' to the legal character '%' where-ever it is found in a part name. Any unprovided characters will fall back to the default mapping above. To map additional characters simply continue to add following the formula <mapchar><string> to the end of the option. Again, any unprovided characters will fall back to the default mapping above. Here's an example of changing two illegal character maps: Example: -cm"?%\"8" Note in this example, to modify the " character it must be sent as \"

			so Windows won't think your terminating the quote on the overall string. The option above will map '?' to '%' and '"' to '8'. Default Setting: Part Name Character Mapping is OFF by default.
-[no]pnt	X		-pnt = Part Name Trimming. Trim part names longer than 233 characters minus file extension. Many CAD systems will import a file, say a STEP assembly, and when they do they will create individual files for each part in the assembly. They will also name the file based on the part name. So if the part name is longer than 233 characters and they try to use a Windows function to save it, the function will fail as Windows will not allow parts to be created that are longer than 233 characters. The part names will also be given an arbitrary "(1)", "(2)", "(3)" based on the order they were read in. This is also simply prevents duplicate file names and it helps to identify that name trimming has taken place. If your CAD system does not save individual files by part names then you may want to turn this off by sending the -nopnt option. Default Setting: Part Name Trimming is ON by default.
-[no]rc	X		-rc = Set Random Part Color. By default if TransMagic read in a part or assembly of parts with no color attribute attached - it will apply a random color to each part. Uncheck to disable this function. This is very useful when you bring a large assembly into TransMagic and all of the sub-components are one single color. Changing the color of all the parts helps to clearly differentiate between parts in the assembly. Default Setting: Set Random Part Color is ON by default.
-unit[val]		X	- unit = File Unit. The file unit option must be sent with the "val" modifier. If just -unit is sent nothing will happen. Allowable unit types are: • in (inch) • ft (feet) • mm (millimeters) • cm (centimeters) • dm (decimeters) • m (Meters) Default Setting: By default the outgoing units will be set to the global File Settings->General page "Output Units". This is initially inches.

-o-

CATIA V4

TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

Parameter	Read	Write	Description
-[no]be	X		-be = Read NoShow Entities. Even though the -be option has already been documented in the Common Options topic, we are listing it again here as the -be option for CATIA V4 means "NoShow". "NoShow" in CATIA V4 nomenclature is analogous to "Blanked" or "Hidden" in other CAD systems. Default Setting: NoShow\Blanked entities are OFF by default.
-np	X		-np = Read NoPick Entities. Default Setting: NoPick entities are NOT read by default.
-ra	X		-ra = Read all non-Root and Root entities. Default is to read only Root entities. Default Setting: Only Root entities are read by default.
-aw	X		-aw = Read all Workspaces. Default Setting: The default is to read in only the Master Workspace.
-cr	X		-cr = Check for Extra Root entities to read in. Default Setting: The default is not to read in Extra Root entities.
-ch	X		-ch = DO NOT Hide Construction Geometry. Note: The default setting simply marks the incoming entities as "hidden", it does not eliminate them from the translation. Default Setting: The default is to hide construction geometry.

-afa
-afc
-afu<filter>
	X		-af = Apply Layer Filter. Only one of these -af filters can be applied:
			• -afa = Translate All Layers or more specifically, any layers with geometry on them (default).
			• -afc = Translate Current Layer, the layer that was last active when the file was saved.
			• -afu<filter> = Translate User Defined Layer. Here you sent the -af option followed by a layer by name. This option will only translate geometry on this specified layer:
			  ○ Example: -afu"Engine Block"
			  ○ Sending this option would translate all geometric entities on the layer named "Engine Block"
			
			Default Setting: The default layer filter is set to translate all layers

			with geometry.
-tsolide		X	-tsolide = Translate SOLIDE Output Type. When -tsolide is sent, a multi-lump solid is written as a single SOLIDE having many SOLIDDEF. Default Setting: The default is to translate the VOLUME output type. VOLUME output type is when a multi-lump solid is written as many VOLUMEs.
-ver<version>		X	-ver = Set CATIA V4 Output Version. This can be set from 19 to 24, where 19 = 4.1.9, 20 = 4.2.0, 21 = 4.2.1 and so on. Example: -ver23 Sending this option would set the output version to CATIA V4 4.2.3. Default Setting: The default CATIA V4 output version is 4.2.4 (-ver24).

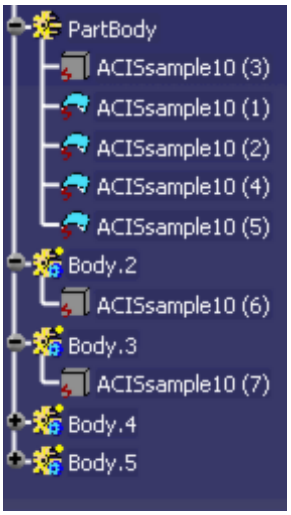
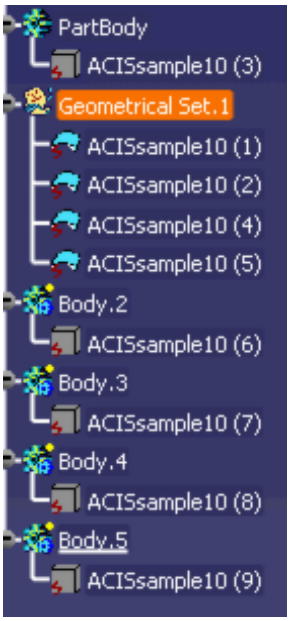
-o-

CATIA V5

TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

Parameter	Read	Write	Description
-[no]pmi	X		-pmi = Product Manufacturing Information. Currently this information is only readable by a TransMagic authored application but if you're converting to a *.sat or *.tmr file then this information will be saved to that format. As PMI data is only stored at the part level, this option will only be invoked if a part document is re. The TransMagic License Key must include the PMI option as well. Default Setting: Product manufacturing information is OFF by default.
-[no]pmirefgeom	X		-pmirefgeom = Retain PMI Reference Geometry. Some PMI data is attached to reference geometry that is not attached to the actual part itself. If the reference data is translated it will come into the document as blanked/hidden entities. Default Setting: Retain PMI reference geometry is OFF by default.
-[no]tan	X		-tan = Trim Assembly Names. Turning this option off will append part names with the assembly structure. Assembly levels will be delimited by the "|" character if Part Name Character Mapping is OFF. Otherwise it will be mapped to what-ever is set in Part Name Character Mapping. By default "|" is remapped to "!". See the Common Options section for more information on Part Name Character Mapping. This option is most often used in conjuction with NGRAIN Write Special Entity Processing. See the NGRAIN docs for more information. Default Setting: Retain Trim Assembly Names is ON by default.
-[no]iwo	X		-iwo = In Work Object. The In Work Object is CATIA nomenclature for the "Last Save State". In most cases the user wants the file as it was last saved by the engineer who last "touched" the file. However, you can also retrieve the "Final State" version of the file by turning this option off. Default Setting: In Work Object is ON by default.
-ver<version>		X	-ver = Set CATIA V5 Output Version. This can be set from 6 to 22. Example: -ver15 Sending this option would set the output version to CATIA V5 R15. Default Setting: The default CATIA V5 output version is R18.

			TransMagic is consistently ahead of the curve in terms of version support and for that reason our default output versioning is usually a couple versions behind the latest supported version. This approach consistently addresses the most widely used versions.
-bth		X	-bth = Body Type Hybrid. By default TMCmd will produce a Non-Hybrid Body Type which means the part geometry gets created in a Geometrical Set in V5. Sending the -bth option will instruct TMCmd to create a Hybrid Body Type which means that the part geometry gets created under a PartBody. • Hybrid:  Non-Hybrid:  Default Setting: Body type hybrid is OFF by default.
-v5wf		X	-v5wf = Disable Illegal V5 Character Warning. CATIA V5 does not accept file names with the "±" character in them. By default TMCmd will automatically change the ± to _ so that the translation will complete. If the -v5wf option is sent then TMCmd will not warn the user and will not create the file. Default Setting: Disable illegal character warning is OFF by default.

-o-

Creo | Pro/E

TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

Parameter	Read	Write	Description
-[no]rs	X	n/a	-rs = Re-Surface Erroneous Spline Surfaces. Default Setting: The default is to Re-Surface Erroneous Spline Surfaces.

-o-

HSF

TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

Parameter	Read	Write	Description
-acisver		X	-acisver = Set ACIS Output Version. This can be set from 1 to 23. This option is relative to the ACIS Streaming Format (*.asf) only. Example: -ver15 Sending this option would set the output version to ACIS R15. 1 actually equals version 1.6 as this was the first shipping version Default Setting: The default ACIS output version is 7.0. TransMagic is consistently ahead of the curve in terms of version support and for that reason our default output versioning is usually a couple versions behind the latest supported version. This approach consistently addresses the most widely used versions.
-[no]topol		X	-topol = Include Topology. The HSF file is a polygonal format with the ability to represent topological data as well (Face, Edge, Vertex, & Connectivity). For this reason the HSF file is much more suited to engineering visualization than other polygonal formats such as *.stl and *.wrl (VRML). This is why Topology data is ON by default. To write a pure polygonal HSF file, first send -notopol, then send -nolines. This will result in lighter weight yet less informative HSF file. Default Setting: Write Topolgy is ON by default.
-[no]lines		X	-lines = Include Line/Edge Data. The HSF file is a polygonal format with the ability to represent topological data as well (Face, Edge, Vertex, & Connectivity). For this reason the HSF file is much more suited to engineering visualization than other polygonal formats such as *.stl and *.wrl (VRML). TransMagic uses the HSF file to offer a lightweight engineering visualization format. This is why Line/Edge data is ON by default. To write a pure polygonal HSF file, first send -notopol, then send -nolines. This will result in lighter weight yet less informative HSF file. Default Setting: Write Line/Edge Data is ON by default.
-[no]cmpnorm		X	-cmpnorm = Compress Normals. By default TransMagic applies advanced compression to normal's to reduce their file size. Though there is really no reason to disable this, you may. You will not notice a difference in the HSF file though the file size will grow. Default Setting: Write Compressed Normals is ON by default.

-[no]cmpvert		X	-cmpvert = Compress Vertices. By default TransMagic applies advanced compression to vertices to reduce their file size. Though there is really no reason to disable this, you may. You will not notice a difference in the HSF file though the file size will grow. Default Setting: Write Compressed Vertices is ON by default.
-[no]cmpadv		X	-cmpadv = Use Advanced Compression. By default TransMagic applies Advanced Compression to the HSF file to reduce the file size. Though there is really no reason to disable this, you may. You will not notice a difference in the HSF file though the file size will grow. Default Setting: Write with Advanced Compression is ON by default.
-[no]dict		X	-dict = Include File Dictionary. The file dictionary is a piece of data in the HSF file that contains entries for the file locations of various representations of entities. If this is enabled then the location of the original file is written to the HSF file. Though there is really no reason to disable this, you may. Default Setting: Write File Dictionary is ON by default.
-vbits<numbits>		X	-vbits = Bits Per Vertex. The Bits Per Vertex setting controls the level of compression for vertices. Though there is really no reason to change this setting, you may. You will not notice a difference in the HSF file though this setting will affect the file size. The value of vertex bits can be anything in the range of 8-72. Default Setting: Default Bits Per Vertex is 40.
-nbits<numbits>		X	-nbits = Bits Per Normal. The Bits Per Normal setting controls the level of compression for vertices. Though there is really no reason to change this setting, you may. You will not notice a difference in the HSF file though this setting will affect the file size. The value of normal bits can be anything in the range of 8-72. Default Setting: Default Bits Per Normal is 20.
-ver<version>		X	-ver = Set HSF Output Version. This can be set from 100 to 1900. HSF files can often have specific point releases that need to be used such as version 1.05 or 6.30, etc. If you want to write out whole HSF versions then use 100, 200, 300, and so on. If you want to write out point versions then the latter two digits represent the specific point version. For example if you wanted to output version 6.30 then you would send 630 as the version or 17.10 then you would send 1710 as the version. Example: -ver1600 Sending this option would set the output version to HSF version 16.00. Default Setting: The default HSF output version is 19.00. TransMagic is consistently ahead of the curve in terms of version support and for that reason our default output versioning is usually a couple versions behind the latest supported version. This approach consistently addresses the most widely used versions.

-o-

IGES

TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

Parameter	Read	Write	Description
-[no]sr	X		-sr = Apply Individual Surface Repair. Default Setting: Surface Repair is ON by default.
-[no]im	X		-im = Read IGES Solids (MSBOs). Default Setting: Read IGES Solids is ON by default.
-[no]ae	X		-ae = Read Associativity Entities. Default Setting: Read Associativity Entities is ON by default.
-[no]ag	X		-ag = Read Annotations As Geometry. Default Setting: Read Annotations As Geometry is ON by default.
-[no]sf	X		-sf = Read Subfigure Entities. Default Setting: Read Subfigure Entities is ON by default.
-rtc<choice>	X		-rtc = Read Trim Curve Preference. Allowed choices are: • 0 = As Specified In File. • 1 = Use 2D Parameter Curves. • 2 = Use 3D Edge Curves. • 3 = Use Both 2D Parameter And 3D Edge Curves. Default Setting: Default Read Trim Curve Preference is As Specified In File.
-[no]ts	X	X	-ts = Translate Trimmed Surfaces. Default Setting: Translate Trimmed Surfaces is ON by default.
-if<choice>	X	X	-if = IGES Flavor. Allowed choices are: • a = Autocad • s = SolidWorks • j = JAMA • v = VX (Write Only) Default Setting: Default is no IGES Flavor specified.
-[no]ss		X	-ss = Write All Surfaces As Splines (IGES 128 Entity). If an application has an issue reading an IGES file out of TMCmd the first two things to try are to set this -ss option and the -sc option below. Sending these two options represent the most benign IGES file

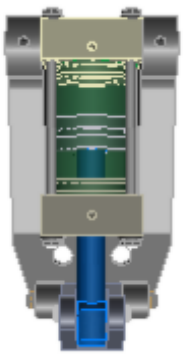
			possible as the importing application only needs to support one surface and curve type, the spline, which all IGES readers support. Default Setting: Default is to write analytic surface types such as sphere, cone, plane, cylinder, torus and spline surfaces for freeform surfaces.
-[no]sc		X	-sc = Write All Curves As Splines (IGES 126 Entity). If an application has an issue reading an IGES file out of TMCmd the first two things to try are to set this -sc option and the -ss above. Sending these two options represent the most benign IGES file possible as the importing application only needs to support one surface and curve type, the spline, which all IGES readers support. Default Setting: Default is to write analytic curve types such as arcs and lines and spline curves for freeform curves.
-ot<type>		X	-ot = Solid Output Type. Allowed choices are: • 0 = Surfaces. • 1 = Solids (IGES MSBO). • 2 = Wireframes. Default Setting: Default is Surfaces.
-sot<type>		X	-sot = Surface Output Type. Allowed choices are: • 0 = IGES Trimmed Surfaces (IGES 144 Entity). • 1 = IGES Bounded Surfaces (IGES entity 143). Default Setting: Default is Trimmed Surfaces (IGES 144 Entity).
-wtc<choice>		X	-wtc = Write Trim Curve Preference. Allowed choices are: • 0 = Both 2D & 3D Trim Curves With No Preference. • 1 = Both 2D & 3D Trim Curves With 2D Preferred. • 2 = Both 2D & 3D Trim Curves With 3D Preferred. • 3 = 3D Trim Curves Only. • 4 = 2D Trim Curves Only. • 5 = Both 2D & 3D Trim Curves With Equal Preference. Default Setting: Default Write Trim Curve Preference is 3D Trim Curves Only.

-o-

IMAGES

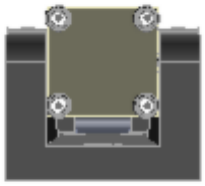
TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

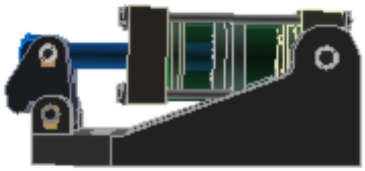
Parameter	Read	Write	Description
-of<outputspec>	n/a	X	TMCmd currently supports the following image output formats: -oftif | Uncompressed TIF -ofbmp | Uncompressed Bitmap -ofpng | Portable Network Graphics -ofjpg | JPEG All of the image output settings below apply to any image output format.
-imgw<val>	n/a	X	-imgw = Image Width. Pixel width for image. Default Setting: Default Image Width is 600.
-imgw<val>	n/a	X	-imgh = Image Height. Pixel width for image. Default Setting: Default Image Height is 600.
-view<val>	n/a	X	-view = View Orientation. Allowed choices are: • top (-viewtop) = Top View plane orientation:  • • bott (-viebott) = Bottom View plane orientation.

- front (-viewfront) = Front View plane orientation.

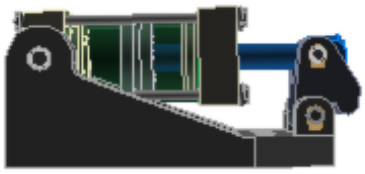
- back (-viewback) = Back View plane orientation.



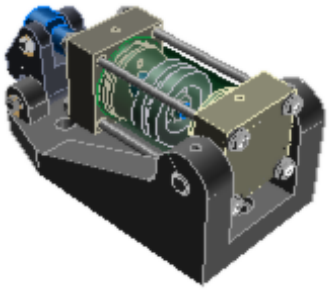
- right (-viewright) = Right View plane orientation.



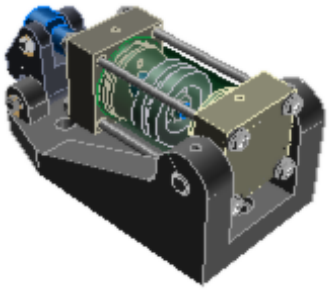
- left (-viewleft) = Left View plane orientation.



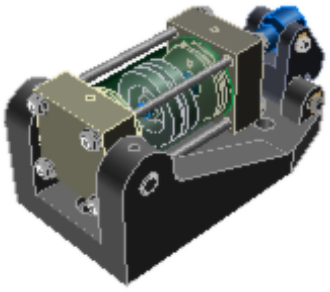
-
- iso (-viewiso) = Isometric View orientation.



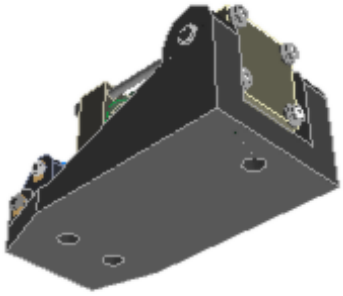
-
- frt (-viewfrt) = Front Right Top View orientation.



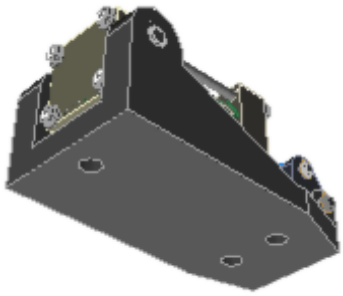
-
- flt (-viewflt) = Front Left Top View orientation.



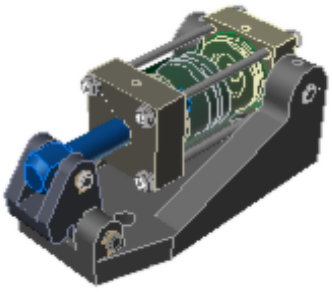
- frb (-viewfrb) = Front Right Back View orientation.

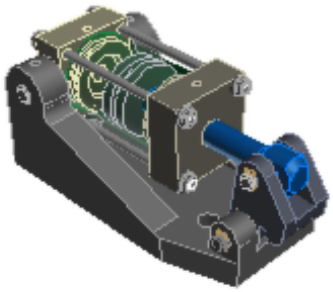
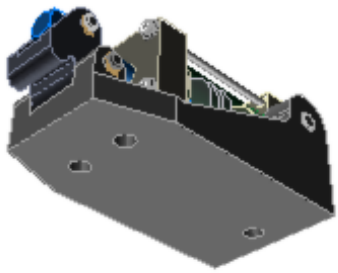
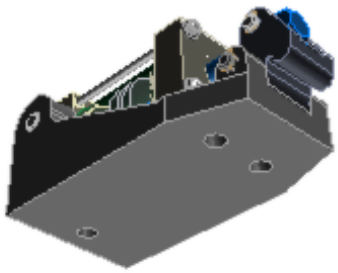


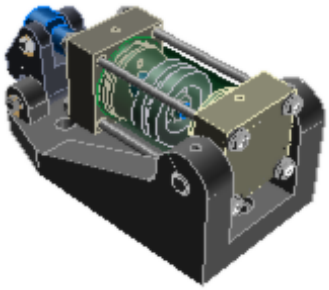
- flb (-viewflb) = Front Left Back View orientation.



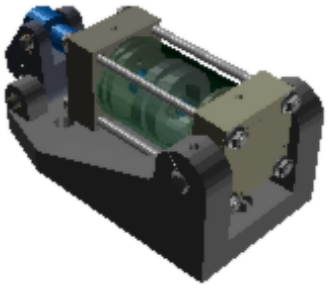
- brt (-viewbrt) = Back Right Top View orientation.



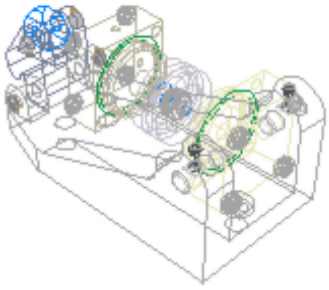
			• blt (-viewblt) = Back Left Top View orientation.  • • brb (-viewbrb) = Back Right Bottom View orientation.  • • blb (-viewblb) = Back Left Bottom View orientation.  • Default Setting: Default View Orientation is isometric.
-rend<val>	n/a	X	-rend = Rendering Mode. Allowed choices are: • shade (-rendshade) = Shaded Rendering Mode.



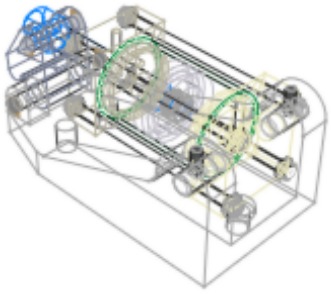
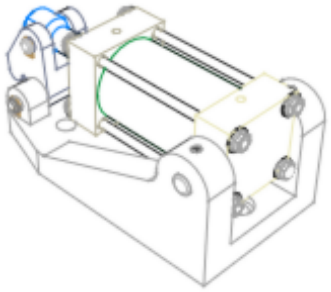
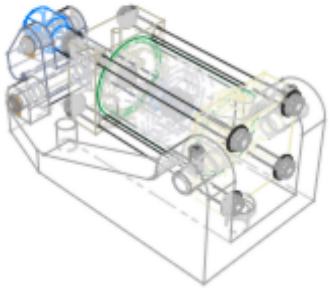
- flat (-rendflat) = Flat (polygon) Shading Rendering Mode.

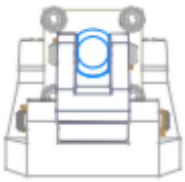
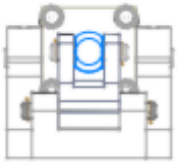


- wf (-rendwf) = Wire Frame Rendering Mode.



- sil (-rendsil) = Silhouette Rendering Mode = Wireframe with silhouette edges.

			 • • hl (-rendhl) = Hidden Line Rendering Mode.  • • hlwd (-rendhlwd) = Hidden Line Rendering Mode w/Dashed lines for occluded entities.  • Default Setting: Default Rendering Mode is Shaded.
-rend<val>	n/a	X	-proj = Projection Mode. Allowed choices are: • per (-projper) = Perspective Projection:

			 • • ort (-projort) = Orthograpic (parallel) Projection:  • Default Setting: Default Projection Mode is Perspective Projection.

-o-

Inventor

TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

Currently there are no specific Inventor options outside of the Common Options.

Note: The TransMagic Inventor translator requires that either Inventor or Inventor View be installed for proper operation. Inventor View is a FREE downloadable Inventor Viewing application. This application can be downloaded from the TransMagic download page: http://www.transmagic.com/support/install/downloads

-o-

JT

TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

Parameter	Read	Write	Description
-[no]brep	X	X	-brep = Read/Write JT B-Rep (CAD geometry). Sending the -nobrep option will read/write JT 3D Visualization only parts/assemblies. Default Setting: The default is to read/write JT B-Rep geometry.
-ver<version>		X	-ver = Set JT Output Version. This can be set from 64 to 95. These correspond to the JT versions 6.4 to 9.5. Example: -ver70 Sending this option would set the JT output version to 7.0. Default Setting: The default JT output version is 8.0. TransMagic is consistently ahead of the curve in terms of version support and for that reason our default output versioning is usually a couple versions behind the latest supported version. This approach consistently addresses the most widely used versions.
-dt	X	X	-dt = Direct Translation. This option can be specified when creating a JT file with B-Rep data directly from a Parasolid file and also when creating a JT + B-Rep to a Parasolid file. Since the B-Rep data inside a JT file is in fact Parasolid data, and if you have a Parasolid file you need to convert to a JT + B-Rep or a JT + B-Rep file you need to convert to a Parasolid file then there's no need to go through TransMagic's own intermediate "TMR" B-Rep translation. This will greatly speed up the conversion of Parasolid to JT + B-Rep and JT + B-Rep to Parasolid as there really is no conversion of B-Rep data but rather just a file format organizational conversion. Unlike other Translator Specific Options which get set immediately after the format has been specified, this option needs to be sent right after the call to TMCmd.exe. For example: TMCmd -dt "C:\CADData\SampleFile.jt" -ofx_t This is because TMCmd needs to know to "turn off" it's internal B-Rep. Also, this option has to be sent alone in a single translation. When using this option you can't translate out to multiple formats in one command line call. If you do they will simply fail to write. This option must be sent alone after TMCmd and only for X_T->JT and JT->X_T translations.
-fot<val>		X	-fot = File Output Translation. This option accepts the following

			integer values: • 0 = Single File Assembly | Entire assembly in one file. • 1 = Typical Assembly | Assembly JT files + part JT files. • 2 = Typical Assembly w/Directory | Typical assembly JT file + directory. • 3 = All Assembly Files | Each assembly is a JT file including parts. Default Setting: The default JT File Output Type is 1 or a Single File Assembly.

JT Faceting Options:
Please note that ANY faceting option listed here is relative to Visualization Representation (V-Rep) JT files ONLY. For JT files with B-Reps we use the JT Open recommended option "JtkSMART_LODS" in our translator, which automatically creates several LODs based on the B-Rep data. With B-Rep JT files it is presumed that the applications importing them will use the B-Rep data to generate a facet mesh on the importing application side and thus faceting options are irrelevant.

Standard Faceting Option:
The majority of all JT faceting needs will be handled by the standard Faceting option.

Parameter	Read	Write	Description
-fct<tol>		X	-fct = Facet Resolution. The JT Facet Resolution can be one of five settings: lowest, low, normal, high, highest. These settings refer to the common facet resolution setting "normal deviation" and correspond to the following settings: lowest = 45 degrees, low = 30 degrees, normal = 15 degrees, high = 10 degrees, highest = 5 degrees. Default Setting: The default JT Facet Resolution setting is "normal".

Advanced Faceting Options:
Sending ANY Advanced Faceting option will override the Standard Faceting Option -fct. You can send any ONE Advanced Faceting Option and the defaults will be used for the others or you may send all options.

Parameter	Read	Write	Description
-nd<val>	n/a	X	-nd = Max Normal Deviation. The normal deviation specifies that no two adjacent polygon normals can deviate by more than this value. Default Setting: Default Max Normal Deviation is 15 degrees.
-sd<val>	n/a	X	-sd = Max Surface Tolerance. The surface tolerance specifies that for any given facet, the distance from the facets centroid to it's surface may not exceed this value. Default Setting: Default Max Normal Deviation is .020 inch (or the equivalent in any other unit).

-ep<val> or -lu<val>	n/a	X	Max Edge Length Preference. The Max Edge Length refinement specifies that any polygon edge cannot exceed this value. There are two ways to set the Max Edge Length Preference; however, only one of these can be specified. If you send both options, which would be incorrect, which-ever option was sent last in the command string would be the option that was used. -ep = Edge Length By Percent. This is defined by generating a virtual bounding box around the entity being output, then taking the diagonal of that box. This percentage is the percentage of length of the part's bounding box diagonal. This method provides a very uniform method for specifying a Max Edge Length for any variety of part in any variety of sizes. Default Setting: Default Edge Length By Percent is 10%. -lu = Edge Length By Unit. This refinement is much more meaningful when you know the size of the part being saved to STL. For example you would not want to specify a Edge Length By Unit of 10.0 for something the size of a football field. This would result in an excessive and unnecessary amount of polygons. Default Setting: Default Max Normal Deviation is 10.0 inches (or the equivalent in any other unit). Default Setting: Default is an Edge Length By Percent of 10%.

-o-

NGRAIN

TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

Parameter	Read	Write	Description
-nd<val>	n/a	X	-nd = Max Normal Deviation. The normal deviation specifies that no two adjacent polygon normals can deviate by more than this value. Before geometry is converted to the NGRAIN voxel representation, it must first be faceted. This is one of the facet refinements used to generate this facet mesh before converting it to a voxel representation. Default Setting: Default Max Normal Deviation is 10 degrees.
-sd<val>	n/a	X	-sd = Max Surface Tolerance. The surface tolerance specifies that for any given facet, the distance from the facets centroid to it's surface may not exceed this value. Before geometry is converted to the NGRAIN voxel representation, it must first be faceted. This is one of the facet refinements used to generate this facet mesh before converting it to a voxel representation. Default Setting: Default Max Normal Deviation is .1 unit.
-vc<val>	n/a	X	-vc = Voxel Cube Size. This value is also known as the Dimension value. The Dimension determines the overall size of the box of the 3D model. The higher the number, the more detailed the model and the larger the resulting NGRAIN file. Default is: 500 VSpace Units. This means the VSpace (Voxel Space) has a dimension of 500x500x500 Units – an over-all Voxel Space resolution – think of this as an overall bitmap resolution only in 3 dimensions. A larger value = higher quality, a smaller value = lower quality. The maximum value is 10000 Units. Default Setting: Default is 500 VSpace Units.
-vos<val>	n/a	X	-vos = Voxel Oversampling. This value is also known as the Visual Quality value. The Visual Quality determines the accuracy of the model. The higher the setting, the higher the fidelity, and the longer the conversion process will take. Default is 4. This option allows you to set the level of oversampling. Oversampling is a quality setting during the conversion process that involves scaling the resolution of the model upwards by a specified factor to determine more accurate settings of which voxels are occupied, what their color values are, and what their normal value is when the model is scaled back down to its intended size. The range of values for this setting is 1 to 7. Default Setting: Default Voxel Oversampling is 4.
-ver<val	n/a	X	-ver = NGRAIN Output Version. This can be set from 32 or 40. These

>			correspond to the JT versions 3.2 or 4.0. Example: -ver32 Sending this option would set the NGRAIN output version to 3.2. Default Setting: The default NGRAIN output version is 4.0. TransMagic is consistently ahead of the curve in terms of version support and for that reason our default output versioning is usually a couple versions behind the latest supported version. This approach consistently addresses the most widely used versions.

3KO Specific Options

Parameter	Read	Write	Description
-minres<val>	n/a	X	-minres = Minimum Resolution. The minimum size that the longest axis of a converted part should be. This value should be between 0 and the "Max Resolution". Default Setting: Minimum Resolution is OFF by default.
-maxres<val>	n/a	X	-maxres = Maximum Resolution. The maximum size that the longest axis of a converted part should be. This value should be between the "Min Resolution" and the "Voxel Cube Size". Default Setting: Maximum Resolution is OFF by default.
-pa	n/a	X	-pa = Part Axis Alignment. Applies To Version: 4.1. Optimize orientation of parts. Convert parts using their local orientation rather than model orientation. Default Setting: Part Axis Alignment is OFF by default.
-gi	n/a	X	-gi = Geometry Instancing. Applies To Version: 4.1. Duplicate parts will be converted once and reused where possible. Enable "Part Axis Alignment" to make sure parts are instanced regardless of their orientation in the model. Default Setting: Geometry Instancing is OFF by default.

NGW Specific Options:

Parameter	Read	Write	Description
-rpd	n/a	X	-rpd = Include Raw Poly Data. Enabling this option allows for special post-processing of surface data. This feature is not relative to "typical" usage. Sending this option will force the NGRAIN format to NGW and the NGRAIN version to 4.0. Contact NGRAIN Support for more information. Default Setting: Include Raw Poly Data is OFF by default.

-ngrps			-ngrps = NGRAIN Groups. Note, when -ngrps options are used the output format is forced to NGW regardless of which format is specified in the -f or -of option. Note also that SEP applies more specifically to CATIA V5 as the input format. In order to properly use SEP you MUST set the following CATIA V5 Read options: • Read Free Curves, V5 Read Option: -fc • Add Assembly Structure To Part Names, V5 Read Option: -notan To specify NGRAIN Groups you must create a file named "GroupName.grp" and it should reside in the user folder: XP: %USERPROFILE%\My Documents\TransMagic NGRAIN Groups Visa & Win7: %USERPROFILE%\Documents\TransMagic NGRAIN Groups The file identifies groups using the < GroupName></GroupName> delmiters and <SearchString></SearchString> delimeters. Here is an example of a "GroupName.grp" file: <GroupName>Seams</GroupName> <SearchString>Search String</SearchString> <GroupName>Fasteners</GroupName> <SearchString>Search String 1;Search String 2</SearchString> - Sending the -ngrps option will tell TMCmd to look for Additional Notes: • You can search for multiple strings by adding a semicolon ";" between search strings. • The same entity can be added to more than one group. Default Setting: NGRAIN Groups are OFF by default.
-sdp	n/a	X	-sdp = Large Part Sub-Division. This is a special functionality of the NGRAIN translator that instructs it to sub-divide regions of the voxel space for more efficient large part handling. The -sdp option will automatically set the following values for the related -pst and -msd

			options if none are specified: -pst4000 -pst1000 These options are explained in more detail in below and my be overridden with user specified values. Sending this option will force the NGRAIN version to 4.0. Default Setting: Large Part Subdivision is OFF by default.
-pst<val>	n/a	X	-pst<val> = Part Size Threshold. This is the criteria for applying sub-division on a particular part: if max. dimension (width, height, or depth) of a part is larger than or equal to this number, the part will be sub-divided. A default value of 4000 will be entered by default; however, optionally one can send the -pst option enter any value between 1 and 10,000 as the part size threshold. Default Setting: Large Part Size Threshold is OFF by default. If the -sdb option is sent then the default is -pst4000.
-msd<type>	n/a	X	-msd<val> = Maximum Sub-Division Size. This is the maximum dimension of a sub-division. This value specifies that the max sub-division's dimension will not exceed this number. A default value of 1000 will be entered; however, optionally one can send the -msd option enter any value between 1 and 10,000 as the maximum sub-division size threshold. Default Setting: Maximum Sub-Division Size is OFF by default. If the -sdb option is sent then the default is -msd1000.

-o-

Parasolid

TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

Parameter	Read	Write	Description
-x_b		X	-x_b = Output Parasolid Binary File. Default Setting: The default is to output to an ASCII *.x_t file.
-ver<version>		X	-ver = Set Parasolid Output Version. This can be set from 10 to 25. Example: -ver11 Sending this option would set the Parasolid output version to 11. Default Setting: The default Parasolid output version is 17. TransMagic is consistently ahead of the curve in terms of version support and for that reason our default output versioning is usually a couple versions behind the latest supported version. This approach consistently addresses the most widely used versions.

-o-

SAT

TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

Parameter	Read	Write	Description
-ver<version>		X	-ver = Set ACIS Output Version. This can be set from 1 to 23. Example: -ver15 Sending this option would set the output version to ACIS R15. 1 actually equals version 1.6 as this was the first shipping version Default Setting: The default ACIS output version is 7.0. TransMagic is consistently ahead of the curve in terms of version support and for that reason our default output versioning is usually a couple versions behind the latest supported version. This approach consistently addresses the most widely used versions.

-o-

SolidWorks

TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

Currently there are no specific SolidWorks options outside of the Common Options.

-o-

STEP

TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

Parameter	Read	Write	Description
-ap203		X	-ap203 = Use Application Protocol 203. Default Setting: The default is to use AP 214.

-o-

STL

TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

Parameter	Read	Write	Description
-nd<val>	n/a	X	-nd = Max Normal Deviation. The normal deviation specifies that no two adjacent polygon normals can deviate by more than this value. Default Setting: Default Max Normal Deviation is 5 degrees.
-sd<val>	n/a	X	-sd = Max Surface Tolerance. The surface tolerance specifies that for any given facet, the distance from the facets centroid to it's surface may not exceed this value. Default Setting: Default Max Normal Deviation is .005 unit (or the equivalent in any other unit).
-ep<val> or -lu<val>	n/a	X	Max Edge Length Preference. The Max Edge Length refinement specifies that any polygon edge cannot exceed this value. There are two ways to set the Max Edge Length Preference; however, only one of these can be specified. If you send both options, which would be incorrect, which-ever option was sent last in the command string would be the option that was used. -ep = Edge Length By Percent. This is defined by generating a virtual bounding box around the entity being output, then taking the diagonal of that box. This percentage is the percentage of length of the part's bounding box diagonal. This method provides a very uniform method for specifying a Max Edge Length for any variety of part in any variety of sizes. Default Setting: Default Edge Length By Percent is 10%. -lu = Edge Length By Unit. This refinement is much more meaningful when you know the size of the part being saved to STL. For example you would not want to specify a Edge Length By Unit of 10.0 for something the size of a football field. This would result in an excessive and unnecessary amount of polygons. Default Setting: Default Max Normal Deviation is 10.0 inches (or the equivalent in any other unit). Default Setting: Default is an Edge Length By Percent of 10%.
-[no]apf	n/a	X	-apf = Procedural Faceting Mode. Normally all facets are based on spline approximations of surfaces - this results in a higher performance faceting; however, it's not as accurate as possible. Procedural Faceting will cause the faceter will facet surfaces based on

			their procedural definition where possible (i.e. spheres planes, cylinders, tori, cones, etc.) which is more accurate. Sending the -noapf option will turn the Procedural Faceting mode off which will result in faster spline approximated faceting. In addition this will invoke Adaptive Faceting: Adaptive Faceting is employed to lay a grid of non-equidistant lines, rather than a regular grid, for surfaces. This non-regular grid is based on the maximum surface deviation. Higher curvature areas get a greater number of grid lines, while lower curvature areas get fewer lines. In all cases, this setting tries to satisfy the facet settings to generate more uniform triangle sizing. This is a processor intensive option and is well suited to generating uniformly triangulated *.stl files for the purposes of generating an FEA mesh vs. a Rapid Prototype mesh. Default Setting: Default is Procedural Faceting Mode.
-[no]stlb	n/a	X	-stlb = Binary STL Output File. Binary STL files are more widely used and are smaller in size than an ASCII STL file by a factor of 10. Sending the -nostlb option will generate an ASCII text based STL file. Default Setting: Default is a Binary STL Output File.

-o-

UG / NX

TM COMMAND Basic Syntax: TMCmd [tmcmdopts] [globalopts] inputpath [inopts] -of<outputspec> [outopts] ...

Please see the Documentation Conventions topic for details on common option conventions.

Parameter	Read	Write	Description
-[no]il	X	n/a	-il = Read Invisible Layers. *SE* = Read suppressed entities. Default is not to read in suppressed entities. Default Setting: Default is to read visible layers only.
-[no]rs	X	n/a	-rs = Re-Surface Erroneous Spline Surfaces. Default Setting: The default is to Re-Surface Erroneous Spline Surfaces.

-o-

Sample Code

C++

C++ | Get TM Install Directory From Registry

The following example checks the Windows Registry for the TransMagic install directory and returns the directory string:

Function Declaration:

```
CString GetTMInstallDir();
```

Function Body:

```
CString GetTMInstallDir()
{
	// Set variables
	unsigned long length;
	auto  HKEY executable_key;
	auto  DWORD type;
	char  ExePath[MAX_PATH];
	CString ReturnString;

	// This gets the root key
	if (RegOpenKeyEx (HKEY_LOCAL_MACHINE,
				"Software\\Microsoft\\Windows\\CurrentVersion\\App
Paths\\TransMagic.exe",
				0L, //Reserved
				KEY_QUERY_VALUE,
				&executable_key) != ERROR_SUCCESS)
	{
		length = 0; // Explicit failure
		ReturnString.Format("TransMagic has not been installed on the system.
Please Install TransMagic.");
	}
	else
	{
		// This get's a subkey's value and converts it to a string
		int error;
		length = sizeof (ExePath);
		error = RegQueryValueEx (executable_key, "InstallDir", NULL, &type,
				(LPBYTE)ExePath,
				&length);
		ReturnString.Format("%s", ExePath);
	}
	RegCloseKey(executable_key);

	return ReturnString;
}
```

-o-

C++ | Run TM COMMAND Translation

The following example uses the Windows SDK ShellExecute function for running a simple TransMagic COMMAND translation:

See the C++ | Get TM Install Directory From Registry topic for the GetTMInstallDir() function used below.

```
// Get the systems "Common Documents" directory as that's where TransMagic puts its Sam ple Files
CString CommonDocsFolder;
TCHAR szPath[MAX_PATH];
SHGetFolderPath(NULL, CSIDL_COMMON_DOCUMENTS, NULL, 0, szPath);
CommonDocsFolder.Format( _T("%s"), szPath);

// Create Sample Command in your code somewhere
char CmdLine[MAX_PATH];
sprintf(CmdLine, "%s\\TMCmd \"%s\\TransMagic\\Sample Files\\CATIA V5\\V5Sample01.CATPart\" -of\"jt\"", GetTMInstallDir(), CommonDocsFolder);
if(RunShell(CmdLine))
	// Do Something
```

Function Declaration:

```
bool RunShell(CString CommandString);
```

Function Body:

```
void RunShell(CString CommandString);
{
	ShellExecute(NULL, "open", CommandString, NULL, NULL, SW_SHOWDEFAULT);
}
```

The following example uses the more flexible Windows SDK CreateProcess function for running a TransMagic COMMAND translation:

See the C++ | Get TM Install Directory From Registry topic for the GetTMInstallDir() function used below.

```
// Create Sample Command in your code somewhere
char CmdLine[MAX_PATH];
sprintf(CmdLine, "%s\\TMCmd \"%s\\TransMagic\\Sample Files\\CATIA V5\\V5Sample01.CATPart\" -of\"jt\"", GetTMInstallDir(), CommonDocsFolder);
if(RunProcess(CmdLine))
	// Do Something
```

Function Declaration:

```
bool RunProcess(CString CommandString);
```

Function Body:

```
bool RunProcess(CString CommandString)
{
	bool success = true;

	STARTUPINFO si;
	PROCESS_INFORMATION pi;
	ZeroMemory( &si, sizeof(si) );
```

```
	si.cb = sizeof(si);
	si.dwFlags = STARTF_USEPOSITION | STARTF_USEFILLATTRIBUTE |
STARTF_USESHOWWINDOW;

	// Show or hide the command window
	si.wShowWindow = SW_HIDE;
	// si.wShowWindow = SW_SHOW;
	si.dwX = 0;
	si.dwY = 0;

	CString TitleText = _T( "Running process, please wait...");
	si.lpTitle = TitleText.GetBuffer();

	// Start the child process.
	if( !CreateProcess(NULL,		// No module name (use command line).
		CommandString.GetBuffer(),	// Command line.
		NULL,				// Process handle not inheritable.
		NULL,				// Thread handle not inheritable.
		FALSE,				// Set handle inheritance to FALSE.
		0,				// No creation flags.
		NULL,				// Use parent's environment block.
		NULL,				// Use parent's starting directory.
		&si,				// Pointer to STARTUPINFO structure.
		&pi ))				// Pointer to PROCESS_INFORMATION
structure.
	{
		AfxMessageBox( "Error running process.", MB_OK|MB_ICONEXCLAMATION);
		success = false;
	}

	WaitForSingleObject( pi.hProcess, INFINITE );

	// Close process and thread handles.
	CloseHandle( pi.hProcess );
	CloseHandle( pi.hThread );

	return success;
}
```

-o-

C++ | Check TransMagic License Properties

The following example is the main application .cpp file for a Windows Console application + MFC support. To build this example very simply do the following:

- In Visual Studio, create a new Windows Console application named "TMLicProps" and add MFC support.
- Copy the contents below into the main application TMLicProps.cpp file, build and run!

```
// TMLicProps.cpp : Defines the entry point for the console application.
//

#include "stdafx.h"
#include "TMLicProps.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#endif


// The one and only application object

CWinApp theApp;

using namespace std;

// Helper Function Declarations
/*************************************************************************/
CString ReadRegString( HKEY KeyHandle, CString KeyPath, REGSAM KeyAccess, CString
KeyEntry );
int ReadRegInt(HKEY KeyHandle, CString KeyPath, REGSAM KeyAccess, CString
KeyEntry);
CString AddMessage( CString KeyName);

int _tmain(int argc, TCHAR* argv[], TCHAR* envp[])
{
	int nRetCode = 0;

	// initialize MFC and print and error on failure
	if (!AfxWinInit(::GetModuleHandle(NULL), NULL, ::GetCommandLine(), 0))
	{
		// TODO: change error code to suit your needs
		_tprintf(_T("Fatal Error: MFC initialization failed\n"));
		nRetCode = 1;
	}
	else
	{
		// Key Handle
		HKEY KeyHandle = HKEY_CURRENT_USER;
		// Key Path
		CString KeyPath;
		KeyPath.Format( _T("Software\\TransMagic\\Licenses\\Apps\\TransMagic
R9\\PrimaryLicense"));
		// Key Security Access Mask
		REGSAM KeyAccess = KEY_QUERY_VALUE;

		// Display the license configuration and the format rights
```

```
            CString msg, tmp;
            msg.Format( _T("TM License Configuration:\n"));
            msg += ReadRegString( KeyHandle, KeyPath, KeyAccess, _T("TM License
Config"));
            msg += _T("\nTM License Version: ");
            msg += ReadRegString( KeyHandle, KeyPath, KeyAccess, _T("TM License
Ver"));
            msg += _T("\nTM License Expiration Date: ");
            msg += ReadRegString( KeyHandle, KeyPath, KeyAccess, _T("TM License
Exp Date"));
            msg += _T("\nTM License Units: ");
            msg += ReadRegString( KeyHandle, KeyPath, KeyAccess, _T("TM License
Units"));
            msg += _T("\nTM License Type: ");
            msg += ReadRegString( KeyHandle, KeyPath, KeyAccess, _T("TM License
Type"));
            msg += _T("\n\nFunctionality Rights:");
            msg += AddMessage( _T("Viz3DWriteRight"));
            msg += AddMessage( _T("ACISReadRight"));
            msg += AddMessage( _T("ACISWriteRight"));
            msg += AddMessage( _T("DWGReadRight"));
            msg += AddMessage( _T("DWGWriteRight"));
            msg += AddMessage( _T("HSFWriteRight"));
            msg += AddMessage( _T("IGESReadRight"));
            msg += AddMessage( _T("IGESWriteRight"));
            msg += AddMessage( _T("JTReadRight"));
            msg += AddMessage( _T("JTWriteRight"));
            msg += AddMessage( _T("INVReadRight"));
            msg += AddMessage( _T("PSReadRight"));
            msg += AddMessage( _T("PSWriteRight"));
            msg += AddMessage( _T("ProEReadRight"));
            msg += AddMessage( _T("STEPReadRight"));
            msg += AddMessage( _T("STEPWriteRight"));
            msg += AddMessage( _T("STLWriteRight"));
            msg += AddMessage( _T("SWReadRight"));
            msg += AddMessage( _T("TMRReadRight"));
            msg += AddMessage( _T("TMRWriteRight"));
            msg += AddMessage( _T("UGReadRight"));
            msg += AddMessage( _T("V4ReadRight"));
            msg += AddMessage( _T("V4WriteRight"));
            msg += AddMessage( _T("V5ReadRight"));
            msg += AddMessage( _T("V5WriteRight"));
            msg += AddMessage( _T("PMIRight"));
            msg += AddMessage( _T("XMLWriteRight"));
            AfxMessageBox( msg, MB_OK);
      }

      return nRetCode;
}

// Helper Functions
/*****************************************************************************/
CString ReadRegString( HKEY KeyHandle, CString KeyPath, REGSAM KeyAccess, CString
KeyEntry )
{
      unsigned long length;
      auto HKEY KeyAddress;
      auto DWORD type;
      char GetValue[MAX_PATH]; // If "" returned then function failed

      if( RegOpenKeyEx( KeyHandle, KeyPath, 0L, KeyAccess, &KeyAddress) ==
ERROR_SUCCESS )
      {
```

```
            // Key Opened, get value
            length = sizeof (GetValue);
            if( RegQueryValueEx( KeyAddress, KeyEntry, NULL, &type,
(LPBYTE)GetValue, &length ) != ERROR_SUCCESS )
                  sprintf( GetValue, "" );
      }
      else
            sprintf( GetValue, "" );
      RegCloseKey (KeyAddress);

      CString RetStr;
      RetStr.Format( _T("%s"), GetValue);

      return RetStr;
}

int ReadRegInt(HKEY KeyHandle, CString KeyPath, REGSAM KeyAccess, CString
KeyEntry)
{
      auto HKEY KeyAddress;
      auto DWORD type;
      int GetValue = 0; // If 0 returned then function failed
      DWORD length = sizeof(BOOL);

      if( RegOpenKeyEx( KeyHandle,  KeyPath, 0L, KeyAccess, &KeyAddress ) ==
ERROR_SUCCESS )
      {
            // Key Opened, get value
            if( RegQueryValueEx (KeyAddress, KeyEntry, NULL, &type, (BYTE
*)&GetValue, &length) != ERROR_SUCCESS)
                  GetValue = 0;
      }
      RegCloseKey (KeyAddress);

      return GetValue;
}

CString AddMessage( CString KeyName)
{
      // Key Handle
      HKEY KeyHandle = HKEY_CURRENT_USER;
      // Key Path
      CString KeyPath;
      KeyPath.Format( _T("Software\\TransMagic\\PrimaryLicense"));
      // Key Security Access Mask
      REGSAM KeyAccess = KEY_QUERY_VALUE;

      CString tmp;
      tmp.Format( _T("\n%s: %d"), KeyName, ReadRegInt( KeyHandle, KeyPath,
KeyAccess, KeyName));
      return tmp;
}
```

-o-

VB

VB | Get TM Install Directory From Registry

The following example checks the Windows Registry for the TransMagic install directory and displays the directory in a message box:

```
Dim TMInstallDir, svProgramKey, svRegInstDir
'Create Windows shell object
Set WshShell = CreateObject("wscript.Shell")
'Access Windows Environment
Set objEnv = WshShell.Environment("PROCESS")

'Get TransMagic install directory from registry
svProgramKey = "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\App
Paths\TransMagic.exe\InstallDir"
On Error resume next
TMInstallDir = WshShell.RegRead(svProgramKey)
If TMInstallDir <> 0 Then
        msgBox TMInstallDir, 1, "TRANSMAGIC INSTALL DIRECTORY"
Else
        msgBox "TransMagic is not installed on the system.  Please install TransMagic.", 1,
"TRANSMAGIC NOT INSTALLED"
End if
```

-o-

VB | Run TM COMMAND Translation

The following example checks the Windows Registry for the TransMagic install directory and runs a TM COMMAND translation:

```
Dim TMInstallDir, svProgramKey, svRegInstDir, svTMCmdString, CustomDocsFolder
'Create Windows shell object
Set WshShell = CreateObject("wscript.Shell")
'Access Windows Environment
Set objEnv = WshShell.Environment("PROCESS")

'Get TransMagic install directory from registry
svProgramKey = "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\App
Paths\TransMagic.exe\InstallDir"
On Error resume next
TMInstallDir = WshShell.RegRead(svProgramKey)
If TMInstallDir <> 0 Then
	'Get the systems "Common Documents" directory as that's where TransMagic puts its
Sample Files
	svProgramKey =
"HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Shell
Folders\Common Documents"
	CustomDocsFolder = WshShell.RegRead(svProgramKey)
	svTMCmdString = """" + TMInstallDir + "System\TMCmd"" """ + CustomDocsFolder +
"\TransMagic\Sample Files\SolidWorks\SWSample01\ujoint.sldasm"" -ofjt -od""%TEMP%"""
	return=WshShell.Run(svTMCmdString, 1, true)
Else
	msgBox "TransMagic is not installed on the system.  Please install TransMagic.", 1,
"TRANSMAGIC NOT INSTALLED"
End if
```

-o-

VB | Check TransMagic License Properties

The following example checks the TransMagic License Properties from the registry and displays the result:

```
Dim TMInstallDir, svProgramKey, svRegInstDir, regKey, configMsg
'Create Windows shell object
Set WshShell = CreateObject("wscript.Shell")
'Access Windows Environment
Set objEnv = WshShell.Environment("PROCESS")

'Get TransMagic install directory from registry
svProgramKey = "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\App
Paths\TransMagic.exe\InstallDir"
On Error resume next
TMInstallDir = WshShell.RegRead(svProgramKey)
If Err <> 0 Then
      msgBox "TransMagic is not installed on the system.  Please install
TransMagic.", 1, "TRANSMAGIC NOT INSTALLED"
End if
Err.Clear

'Display the license properties and the format rights
configMsg = "TransMagic License Properties:" + vbNewLine + _
+ "TM License Config: " + _
+ ReadRegVal("TM License Config") + vbNewLine + _
+ "Version: " + _
+ ReadRegVal("TM License Ver") + vbNewLine + _
+ "Expiration Date: " + _
+ ReadRegVal("TM License Exp Date") + vbNewLine + _
+ "Type: " + _
+ ReadRegVal("TM License Type") + vbNewLine + _
+ vbNewLine + "Functionality Rights" + vbNewLine + _
+ AddMessage("Viz3DWriteRight") + vbNewLine + _
+ AddMessage("ACISReadRight") + vbNewLine + _
+ AddMessage("ACISWriteRight") + vbNewLine + _
+ AddMessage("DWGReadRight") + vbNewLine + _
+ AddMessage("DWGWriteRight") + vbNewLine + _
+ AddMessage("IGESReadRight") + vbNewLine + _
+ AddMessage("IGESWriteRight") + vbNewLine + _
+ AddMessage("JTReadRight") + vbNewLine + _
+ AddMessage("JTWriteRight") + vbNewLine + _
+ AddMessage("INVReadRight") + vbNewLine + _
+ AddMessage("PSReadRight") + vbNewLine + _
+ AddMessage("PSWriteRight") + vbNewLine + _
+ AddMessage("ProEReadRight") + vbNewLine + _
+ AddMessage("STEPReadRight") + vbNewLine + _
+ AddMessage("STEPWriteRight") + vbNewLine + _
+ AddMessage("STLWriteRight") + vbNewLine + _
+ AddMessage("SWReadRight") + vbNewLine + _
+ AddMessage("TMRReadRight") + vbNewLine + _
+ AddMessage("TMRWriteRight") + vbNewLine + _
+ AddMessage("UGReadRight") + vbNewLine + _
+ AddMessage("V4ReadRight") + vbNewLine + _
+ AddMessage("V4WriteRight") + vbNewLine + _
+ AddMessage("V5ReadRight") + vbNewLine + _
+ AddMessage("V5WriteRight") + vbNewLine + _
+ AddMessage("PMIRight") + vbNewLine + _
+ AddMessage("XMLWriteRight")
msgBox configMsg, 1, "TRANSMAGIC LICENSE PROPERTIES"
```

```
'Helper Functions
Function ReadRegVal(regKey)
    Dim key, val, msg
    key = "HKEY_CURRENT_USER\SOFTWARE\TransMagic\Licenses\Apps\TransMagic
R9\PrimaryLicense\" + regKey
      On Error resume next
      val = WshShell.RegRead(key)
      If Err <> 0 Then
            msg = "The following registry entry could not be found:" + vbNewLine +
_
            + key + vbNewLine + vbNewLine + _
            + "This means that TransMagic has not yet been licensed on the
system."
            msgBox msg, 1, "TRANSMAGIC NOT LICENSED"
      Else
            ' Set the function's return value.
            ReadRegVal = val
      End if
      Err.Clear
End Function

Function AddMessage(regKey)
   AddMessage = regKey + ": " + CStr(ReadRegVal(regKey))
End Function
```

-o-

TransMagic OEM Partner

Partner Specific COMMAND Syntax

TransMagic COMMAND can be used as an integrated translation bundle to your own development.

Usage of TM COMMAND in this context requires a special contract with TransMagic and may also include the usage of some special TransMagic OEM Partner specific command options.

In order to use your TransMagic OEM license you need to call a special -tmcmdinit command once per application initialization. You don't want to call this function for every translation.

This function is what generates your run-time TM OEM license. The TM OEM licensing syntax is as follows:

<Path To TMCmd>\tmcmd -tmoem<App Registry Path> -tmcmdinit<TM Customer ID|FAK>

The required TM OEM Licensing arguments are as follows:

Parameter	Description
<Path To TMCmd>	You could opt to use your own install directory by using the -OEMINSTALL option with the TransMagic installer. See the Partner Specific Installation Options section for more details. If you have done this then only you will know where your install directory is as no registry entries will exist. However, if you are relying on an end-user seat of TransMagic traditionally installed than you can get the TransMagic install directory from the following registry location: Key: HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\App Paths\TransMagic.exe Value Name: InstallDir
-tmoem<App Registry Path>	-tmoem = In order to use your TransMagic OEM license you always need to call tmcmd with the -tmoem switch always following the tmcmd executable. This tells TransMagic where to look for your license. The base TransMagic license registry path is: HKEY_CURRENT_USER\Software\TransMagic\Licenses\Apps This <App Registry Path> specifies where underneath this base TransMagic license registry path your own application key will be stored. This path could be as simple as an application name, for example: "Application X" This would store your license here in the registry: HKEY_CURRENT_USER\Software\TransMagic\Licenses\Apps\Application X

	...or you might want to create a more comprehensive registry structure if you may need to have multiple versions of your product installed along with multiple versions of TMCmd. To specifiy a deeper registry structure simply add more paths separated by "\\". The TM OEM license will be stored under the final entry. For example let's say you wanted to store your TM OEM license under <Company Name>\\<Application Name>\\<Application Version> then your <App Registry Path> would look something like this this: "Company X\\Application X\\Version X.X" This would store your license here in the registry: HKEY_CURRENT_USER\Software\TransMagic\Licenses\Apps\Company X\Application X\Version
-tmcmdinit<TM Customer ID|FAK>	-tmcmdinit = TransMagic COMMAND OEM License Initialization. Your TransMagic Customer ID will have a standard GUID form: FE47612D-D4A8-4d47-98A0-5B0DC407A6FC Your FAK, or Feature Access Key, identifies your product configuration. Refer to the TMCmdLicTestNoLib sample application for a best practices usage example.

After you have established your TransMagic OEM Partner run-time license key you are ready to make TM COMMAND calls using your own license. The only difference from any other TMCmd.exe command line translation is that you must always follow the TMCmd.exe with the -tmoem<App Registry Path> directive. The basic TM OEM Partner translation syntax is as follows:

<Path To TMCmd>\tmcmd -tmoem<App Registry Path> <Input File> -of<Output Format>

This will perform a basic translation that reads <Input File>, translates it out to <Output Format> to the same directory as <Input File>. You can control this translation in countless ways using more advanced command line options. For more details on these see the Advanced COMMAND Syntax section.

-o-

Partner Specific Installation Options

TransMagic COMMAND has advanced distribution options built into it's standard installer which support a multitude of distribution scenarios.

Usage of TM COMMAND in this context requires a special contract with TransMagic and may also include the usage of some special TransMagic OEM Partner specific command options.

For reference simply running the standard TransMagic installer has the following effects:

- Displays the TransMagic End User License Agreement (EULA).
- Removes any previous version of TransMagic that may be present on the system.
- Installs all necessary TransMagic redistributables.
- Installs itself on the system and integrates into Windows in the following ways:
    - Installs all core program functionality into the %ProgramFiles% directory by default, which will have the form:
        - %ProgramFiles%\TransMagic Inc\TransMagic RX
    - Installs the TransMagic Sample Files into the CSIDL_COMMON_DOCUMENTS directory, which will have the form:
        - CSIDL_COMMON_DOCUMENTS\TransMagic\Sample Files
        - This directory structure is optional.
    - Populates the Windows application paths registry key:
        - HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\App Paths\TransMagic.exe
        - Values (Value name \ Value data):
            - (Default) \ %ProgramFiles%\TransMagic Inc\TransMagic R9\System\TransMagic.exe
            - InstallDir \ %ProgramFiles%\TransMagic Inc\TransMagic R9\
        - You can query the InstallDir value for the current TransMagic end-user install path.
    - All component registration, environment establishment and registry entries are created at run-time by TransMagic.exe\TMCmd.exe.

However, the TransMagic installer will accept command line switches for fully automated installations. The following are the list of installers switches currently available for standard non-OEM installations:

Parameter	Description
-SILENT	Use to specify a silent install. Progress meters will still be displayed but no use input is required. Usage: -SILENT=1
-INSTALLDIR	Use to specify the the install directory to anything other than the default %ProgramFiles% directory. The actual directory needs to be between the and delimiters to identify the actual directory. For example, to specify the directory “D:\Unique Install Dir” as the new install directory this option would look like the following: -INSTALLDIR="Any Install Directory" The actual directory needs to be between the < DIR> and </DIR> delimiters to identify the actual directory. For example, to specify the

	directory “D:\Unique Install Dir” as the new install directory this option would look like the following: -INSTALLDIR<DIR>D:\Unique TransMagic Dir</DIR>
-LICENSETYPE	Specify a "Workgroup" license or a NodeLock license. Usage: -LICENSETYPE=NodeLock or -LICENSETYPE=Floating
-SWADDIN	Installs the SW Add-Ins. Usage: -SWADDIN=1
-INVADDIN	Installs the Inv Add-Ins. Usage: -INVADDIN=1
-KEEPOLD	Do not remove old versions of TransMagic. Usage: -KEEPOLD=1

TransMagic installer command line usage:
Let's say we wanted a simple SILENT install. The command line would like like the following:

TransMagicR9sp30_x64.exe /l"1033" /v"/qb SILENT=1"

Let's say we wanted a SILENT install of the SolidWorks Add-In. Let's also say this is a floating network license. Usage would be as follows:

TransMagicR9sp30_x64.exe /l"1033" /v"/qb SILENT=1 LICENSETYPE=Floating SWADDIN=1"

Let’s say we wanted a SILENT install, floating network license, and to specify the unique install directory “D:\Unique Install Dir”. Usage would be as follows:

TransMagicR9sp30_x64.exe /l"1033" /v"/qb SILENT=1 INSTALLDIR=“D:\Unique Install Dir” LICENSETYPE=Floating"

NOTE: When running the installer in silent mode the prerequisites are skipped. This is simply an InstallShield behavior, note necessarily a TransMagic installer behavior. As of this writing, TransMagic R9 sp3 requires the following redistributables and silent command lines:

TMPRQ_C++2008sp1_vcredist_x64.exe /qn

TMPRQ_C++2008sp1_ATL_Update_vcredist_x64.exe /qn

TMBootstrapperR9sp30_x64.exe /s /v"/qb"

Partner Specific Installation Options

TMCmd can be run without being installed so long as it's prerequisites are installed. It can then simply be run from it's "System" directory using the Partner Specific COMMAND Syntax. If you have an interest in this option please contact us at: support@transmagic.com for details.

-o-

Index

- A -

- B -

- C -

- D -

- H -

- I -

- J -

- N -

- P -

- R -

- S -

- T -

- U -

- V -