LT-Extender and LTE Kernel Customization Guide

Contact Information

TM-CAD Ingenieurbüro
Torsten Moses
Landjaegerstrasse 34
12555 Berlin / Germany
Phone/Fax: ++49-30-5530620
www.LT-Extender.com
Developer@LT-Extender.com

0. License Agreement

All documentation is provided under the LT-Extender license agreement, that includes the permission to modify the default LT-Extender installation and related files as long as all copyright statements are respected and will remain unchanged. All developers are encouraged to use the documentation provided here in order to customize LT-Extender to meet their specific requirements.
The LT-Extender license agreement allows all registered licensees to modify any files provided, except the files listed below :

LT-Extender.arx, LT-Extender-2K4.arx, all *.exe files, all *.dll files, LTE2000.dll
LTE2000.sys, LTE2000.mod, LTE2000.msg, LTE2000.ldr
LTE2004.sys, LTE2004.mod, LTE2004.msg, LTE2004.ldr
LTE2004Lisp.mod, LTE2004Lisp.ext

The "LT-Extender Developer License Agreement" covers both the usage of LT-Extender and LTE 2000 Kernel – developers can choice which system they would prefer to use with their applications.

1. Introduction

This reference will document all abilities to modify the default LT-Extender installation and features, so all licensees should be able to adjust LT-Extender's features and behaviour due to their specific requirements. It will also document the minimum required files, optional files, and possible changes to configuration and settings files. Especially, this documentation will give assistance on how to integrate/bundle the developer's application with LT-Extender.

Using LTE 2000 Kernel instead of (full/customized) LT-Extender is a great alternative for application developers – using the LTE 2000 Kernel allows developers to load & run their application(s) inside LT without using LT-Extender. Therefore, this documentation also explains how to use & integrate this LTE 2000 Kernel into the developers application(s).

Additionally, this documentation contains an extra chapter on bugs, defects, issues, limitations and 'strange effects' for the ADS/ARX/Lisp API when running under AutoCAD LT. Please see the chapter **"Bugs, Issues, Defects and 'Strange Effects' with ADS/ARX/Lisp APIs under LT"** for further details ….

If there are any questions, ideas, suggestions or special problems that are not covered with this documentation, or if there are any open problems, please feel free to contact the author at every time and for any reason – we will be glad to assist you in any way and as fast as possible – therefore, please send an enamil to

Developer@LT-Extender.de or Develop@LT-Extender.com

2. Installing LT-Extender

There is a separate documentation **"Developer-Tools.pdf"** that describes how to customize the installation process. Please refer to that documentation for all aspects of installing LT-Extender. Additionally, there are 2 tools (Acad_Reg.exe and LTSetup.exe) that may be used to create a comfortable multi-mode installation of your application (multi-mode: single installation, that works with both AutoCAD and LT).

3. Minimum required Files for LT-Extender

To ensure LT-Extender is running properly, the following files are absolutely nessecary and must be installed (standalone or bundled with other applications) under all conditions :

LT-Extender.arx	the main program file for LT 2000-2002
LT-Extender-2k.arx	the main program file for LT 2004
LT-Extender*.msg	one message file must be supplied (see Single/Multi-Language support)
LTSetup.exe	Setup tool to register LT-Extender with AutoCAD LT
LTUninst.exe	related Unregister tool

LTE2004.sys, LTE2004.mod, LTE2004.msg, LTE2004.ldr	Kernel files for LT 2004
LTE2000.sys, LTE2000.mod, LTE2000.msg, LTE2000.ldr	Kernel files for LT 2000-2002
LTE2000.mns, LTE2000.mnr, LTE2000.mnc, LTE2000.dll	LTE 2000 kernel menu files
LTE2004Lisp.mod, LTE2004Lisp.ext	Lisp processor for LT 2004
Unc.dll, Unc95.dll, PsApi.dll	helper for path & process management

Any other files, provided with the default LT-Extender installation, are optionally and may be provided if desired and/or nessecary; additionally, those other files may also be customized to meet the developers needs.

Minimum required Files under 'foreign' LTE2000 Environment

If your applications already uses its own LTE 2000 kernel in oder to be loaded under AutoCAD LT, you will not need to have LT-Extender installed the usual way (via LTSetup.exe) – in such configuration your application may load LT-Extender.arx like any other 'normal' Arx program file)via "ads_arxload()/acedArxLoad()".
LT-Extender will still provide all features and functionality when loaded via "ads_arxload/acedArxLoad" without any differences. Also, LT-Extender may still be customized as documented in the following chapters.

Because LT-Extender will not need to be registered to LT (via LTSetup.exe), you will only need the following required LT-Extender files as minimum :

LT-Extender.arx	the main program file for LT 2000-2002
LT-Extender-2k.arx	the main program file for LT 2004
LT-Extender*.msg	one message file must be supplied (see Single/Multi-Language support)
LTE2004Lisp.*	Lisp processor for LT 2004

4. Optional Files for LT-Extender

The default LT-Extender installation uses a number of additional files, that provide most of the LT-Extender's features, and may be configured due to developer's and application's needs.

Files that may be optionally installed :

LT-Extender * .mnu/mns/mnr/mnc	PullDown & Toolbars (single/multi language based)
LT-Extender * .mnl	related Menu Lisp file (single/multi language based)
LT-Extender * .chm	Online help/documentation (single/multi language based)
LT-Extender * .cmd	localized 'hidden' commands (single/multi language based)
LT-Extender.acf	Support files Reference table (language independent)
LT-Extender.prg	AutoLoad program files (language independent)
LT-Extender.ini	default settings file (language independent)

Optional Files for ExpressTools support

If you would like to support ExpressTools with your application, you will need the following file :

under .\LT-Extender\Support:	AcetUtil.lsp	provides nessecary functions for ExpressTools

Optional Files for LayerTools

If you would like to include ***LayerTools*** with your application, you will need the following files :

under .\LT-Extender\Support:	Lm_Tools.arx, Lmmfccmd.arx	(language independent)
under .\LT-Extender\Support-De:	Lm_Tools.msg, Lm_Tools.dlg	(language dependent)
under .\LT-Extender\Support-En:	Lm_Tools.msg, Lm_Tools.dlg	(language dependent)
etc. for each supported language		

(all folders are also provided with –2k4 postfix : these folders contain all arx files as 2004 based version)
If you don't need multiple-language-support (fixed-language only), you will only need one ".\Support" folder then.
LT-Extender automatically detects the precence of ***LayerTools*** and will load it at startup.

5. Multi-Language Installations and Support

LT-Extender is designed to provide easy customization, especially to provide easy localization ot all LT-Extender files. There are 5 language-dependent files used by LT-Extender (asteriks '*' indicates the language) :

Messages file	(LT-Extender*.msg)
Menu file	(LT-Extender*.mnu/mns/mnr/mnc)
Menu Lisp file	(LT-Extender*.mnl)
Command names	(LT-Extender*.cmd)
Online Help file	(LT-Extender*.chm)

All language dependent files will use the postfix "**-xx**" with **xx** indicating the language used. By default. the LT-Extender installation supplies **German (-de)** and **English (-en)** files; at runtime, the LT-Extender-Setup ("_ltextender_setup") allows to manually select the appropriate files, even in mixed language mode.

Suggestions:
If any developer would like to add further language(s), it is **strongly** recommended to follow the given rules : use a separate postfix **"-xx"** for each language to add; the following language postfixes are internally pre-defined for automatic language selection :

Bulgarian	-bg	Chinese	-ch
Croatian	-cr	Czech	-cz
Danish	-dn	Dutch	-nl
Finnish	-fi	French	-fr
Greek	-gr	Hungarian	-hu
Icelandic	-ic	Italian	-it
Japanese	-jp	Korean	-ko
Norwegian	-no	Polish	-pl
Portuguese	-pg	Romanian	-ro
Russian	-ru	Slovakian	-sv
Slovenian	-sl	Spanish	-cst (castellano) -cat (catalan)
Swedish	-sw	Turkish	-tu

For other languages not listed here, feel free to use any other postfixes (even with more than 2 characters).

Tips:
- if a particular file is not available in localized version, the English file is automatically used (if present)
- if you need to have a particular language to be pre-selected/preset, then add the appropriate entries to the settings file "LT-Extender.ini" and provide this settings file with your installation

The currently used language selection is saved to the settings file **"LT-Extender.ini"**; if nessecary, it may be useful to ship a pre-defined "LT-Extender.ini" with your application, if the application's language is not one of the internal pre-defined languages (see above).

If you don't need multiple languages support (fixed language installation), then refer to the following chapter.

6. Single-Language Installations and Support

If your installation does not need to support multiple languages, you may supply LT-Extender with a single, fixed language.

There are 2 major possibilities, how to provide the specific language:
1. if your specific language is among LT-Extender's known languages (see above), you may follow the rules for naming language-dependent files, and provide only those files; optionally, you may preset the "LT-Extender.ini" settings file with appropriate entries, and deliver the settings file with your application
2. you may also name the language dependent files **without "-xx" postfix**: this means, you may supply all language dependent files using "LT-Extender,*" rule (LT-Extender.msg, LT-Extender.mnu etc.)

At startup, LT-Extender first searches for all nessecary files using a 3-step-algorithm:
Step 1: if a settings file "LT-Extender.ini" is available, all filenames are taken from the settings file
Step 2: files are searched in language-independent mode without "-xx" postfixes (LT-Extender.*)
Step 3: files are searched in language-dependent mode, using "-xx" postfixes (LT-Extender-xx.*)

Optionally:
You may edit the settings file "LT-Extender.ini" to preset all fixed-language files, and provide this prepared "LT-Extender.ini" file with your installation.

7. Recommended LT-Extender's Installation Kits

As described in chapters 5. and 6., LT-Extender may be included into your application as either "Single-Language" or "Multi-Language" package. For these packages, the nessecary files are listed here.

Single-Language package:

The following files must be supplied :

- LTSetup.exe, LTUninst.exe (#) — the registration/unregistration tools
- LTE2000.mns, LTE2000.mnr, LTE2000.mnc, LTE2000.dll (#) — LTE 2000 menu files
- Unc.dll, Unc95.dll, PsApi.dll (#) — UNC and Process management helper files
- LT-Extender.chm (*) — Online Help (localized)
- LT-Extender.acf (*) — control settings for AutoCAD-style environment

Files for LT 2000-2002

- LTE2000.sys, LTE2000.mod, LTE2000.msg, LTE2000.ldr (#) — LTE 2000 Kernel files
- LT-Extender.arx (#) — main program
- LT-Extender.msg (#) — messages file (localized)
- LT-Extender.prg (*) — startup programs list
- LT-Extender.cmd (*) — settings for 'Hidden' commands (localized)
- LT-Extender.mns/mnr/mnc/mnl (*) — menu files (localized)

Files for LT 2004

- LTE2004.sys, LTE2004.mod, LTE2004.msg, LTE2004.ldr (#) — LTE 2000 Kernel files
- LT-Extender-2k4.arx (#) — main program
- LT-Extender-2k4.msg (#) — messages file (localized)
- LT-Extender-2k4.prg (*) — startup programs list
- LT-Extender-2k4.cmd (*) — settings for 'Hidden' commands (localized)
- LT-Extender-2k4.mns/mnr/mnc/mnl (*) — menu files (localized)
- LTE2004Lisp.mod/ext — Lisp processor for LT 2004

(*) these files are optionally and/or may be modified; please refer to chapter 8. for details on optional files
(#) these files are not nessecary, if your application uses its own LTE2000 kernel and LT registration – under this configuration, your base application's arx file may load LT-Extender.arx via "acedArxLoad()"

Multi-Language package:

The following files must be supplied :

Language independent files

- LTSetup.exe, LTUninst.exe (#) — the registration/unregistration tools
- LTE2000.mns, LTE2000.mnr, LTE2000.mnc, LTE2000.dll (#) — LTE 2000 menu files
- Unc.dll, Unc95.dll, PsApi.dll (#) — UNC and Process management helper files
- LT-Extender.acf (*) — control settings for AutoCAD-style environment

Files for LT 2000-2002

- LTE2000.sys, LTE2000.mod, LTE2000.msg, LTE2000.ldr (#) — LTE 2000 Kernel files
- LT-Extender.arx (#) — main program
- LT-Extender.prg (*) — startup programs list

Files for LT 2004

- LTE2004.sys, LTE2004.mod, LTE2004.msg, LTE2004.ldr (#) — LTE 2000 Kernel files
- LT-Extender-2k4.arx (#) — main program
- LT-Extender-2k4.prg (*) — startup programs list
- LTE2004Lisp.mod/ext — Lisp processor for LT 2004

Language dependent files – Language xx

Files for LT 2000-2002

- LT-Extender-xx.msg (#) — Message file (localized to xx)
- LT-Extender-xx.cmd (*) — settings for 'Hidden' commands (localized to xx)
- LT-Extender-xx.mns/mnr/mnc/mnl (*) — menu files (localized to xx)
- LT-Extender-xx.chm (*) — Online Help (localized to xx)

Files for LT 2004

- LT-Extender-2k4-xx.msg (#) — Message file (localized to xx)
- LT-Extender-2k4-xx.cmd (*) — settings for 'Hidden' commands (localized to xx)
- LT-Extender-2k4-xx.mns/mnr/mnc/mnl (*) — menu files (localized to xx)
- LT-Extender-2k4-xx.chm (*) — Online Help (localized to xx)

Language dependent files – Language yy and so on

(*) these files are optionally and/or may be modified; please refer to chapter 8. for details on optional files
(#) these files are not nessecary, if your application uses its own LTE2000 kernel and LT registration – under this configuration, your base application's arx file may load LT-Extender.arx via "acedArxLoad()"

8. Enabling, Disabling, Customizing LT-Extender features

Most of all LT-Extender features may be separately enabled or disabled due to your application's needs. Except the minimum files, that are nessecaey for LT-Extender to work properly, all other files are optionally and may be used to control LT-Extender's features.

Adding "Support Pathes" and defining "Template Path" for your applications

During installation and each time when "LTSetup.exe" is executed, LT-Extender is registered with AutoCAD LT. "LTSetup.exe" checks for the optional files **"LT-Extender.pl"** (LT2000-2002) and **"LT-Extender-2k4.pl"** (LT2004), which define additionaly "Support Pathes", additional menu and the "Template Path" for AutoCAD LT. During Unregistration using "LTUninst.exe", all those added pathes and the menu are removed.
If your application needs some pathes to be added to AutoCAD LT's "Support Pathes", or additional menu, or if the "Template Path" needs to be re-defined, you may edit the "LT-Extender.pl/LT-Extender-2k4.pl" file and add your support/template pathes; then add the file "LT-Extender.pl" to your installation.
Please refer to "Developer Tools.pdf" for more informations and look at "LT-Extender.pl/LT-Extender-2k4.pl" files.

Status: "LTSetup.pl/LTSetup-2k4.pl" is optional

Emulating the AutoCAD environment under AutoCAD LT

Many applications are using references to some special AutoCAD support files, that are usually not available under AutoCAD LT. Therefore, those applications will fail or not be working properly. One of the major LT-Extender's features is to emulate or re-build the original AutoCAD environment, including those AutoCAD-like support files.

LT-Extender uses an optional settings file "LT-Extender.acf" that controls how to emulate the AutoCAD-like environment. This file defines which files are to be copied from LT-style to AutoCAD-style; each time, LT-Extender is started, it will search for all defined AutoCAD-style files, and if nessecary, it will then copy the defined LT-style files to AutoCAD-style files for any missing files.

LT-Extender is shipped with a pre-defined "LT-Extender.acf" file, to manage to most common AutoCAD-style files; those predefined file relations are listed below (Syntax : LT-style file = AutoCAD-style file).

Aclt.lin = Acad.lin
AcltIso.lin = AcadIso.lin
Aclt.pat = Acad.pat
AcltIso.pat = AcadIso.pat
Aclt.dcl = Acad.dcl
Aclt.pgp = Acad.pgp
Aclt.psf = Acad.psf
Aclt.slb = Acad.slb
Aclt.unt = Acad.unt

If a particular relation is missing in that reference list, you may add further entries, as nessecary. In that case, please give a notice to TM-CAD, to include your addition to the default LT-Extender installation.

Status: "LT-Extender.acf" is optionally, **but highly recommended**

Automatic application loading at LT startup

All programs that shall be loaded automatically at LT startup can be defined with file **"LT-Extender.prg/LT-Extender-2k4.prg"**. Usually, the user adds desired program files by using the LT-Extender command "_ltextender_bootapps". All those programs defined here are loaded at LT startup.
If you need some of your application's programs to be loaded automatically, you may ship a pre-defined file "LT-Extender.prg" with your application. There are some other methods available to have your application loaded automatically together with LT-Extender – please see the following chapter.

Status: using LT-Extender startup program list is optionally; may be used to integrate your application into LT

Exposing 'Hidden' AutoCAD commands under LT

One of LT-Extender's most interesting feature is the so-called "Command Activation". Usually, there are more than 60 AutoCAD commands considered to be none-existing under LT – but in fact, nearly all original AutoCAD commands are also implemented with LT. With LT-Extender, all those 'Hidden' commands may be re-activated under LT, so the user will get the full AutoCAD command set to work with, just like as in original AutoCAD.

There is a special settings file **"LT-Extender-xx.cmd/LT-Extender-2k4-xx.cmd"** (language-dependent mode) or **"LT-Extender.cmd/LT-Extender-2k4.cmd"** (language-independent mode), that will define, which commands are to be re-activated and localized to the current LT language version.

If this file is NOT supplied with your installation, most of the 'Hidden' commands are not re-activated.
Additionally, because the "LT-Extender-xx.cmd/LT-Extender-2k4-xx.cmd" or "LT-Extender.cmd/LT-Extender-2k4.cmd" file also contains the localized names for re-activated commands, you should provide the correct language version (correct localized command names).

Status: using "*.cmd" files is optionally, **but highly recommended**

Using and Customizing the default LT-Extender menu file

LT-Extender is shipped with a user-friendly menu file, that provides access to re-activated 'Hidden' commands and it also provides AutoCAD-compatible toolbars for all those commands. Additionally, the LT-Extender menu provides access to some of LT-Extender's functions and features (like Appload, Lisp prompt, AutoStart programs, Setup, Licensing, Help, Web access etc.).

Depending on your installation needs, you may use and install either **"LT-Extender-xx.mnu/LT-Extender-2k4-xx.mnu"** (language-dependent mode) or **"LT-Extender.mnu/LT-Extender-2k4.mnu"** (language-independent mode) together with your application. The LT-Extender menu is automatically loaded & shown at startup, if not loaded and/or shown; if the menu file is not available, no error will occur.

Of course, you may modify the original LT-Extender menu too meet your specific requirements – but it is strongly recommended to preserve LT-Extender's own features, as listed above. In case you would like to modify the original LT-Extender menu, please take care for correct localization and file naming rules.

Status: using LT-Extender menu files is optionally, **but recommended**

Using and Customizing the default LT-Extender menu lisp file

Together with the menu file, LT-Extender is shipped with the appropriate menu lisp file **"LT-Extender-xx.mnl/LT-Extender-2k4-xx.mnl"** (language-dependent mode) or **"LT-Extender.mnl/LT-Extender-2k4.mnl"** (language-independent mode), that is loaded at startup and for each drawing created/loaded.

Currently, LT-Extender makes no use any code inside the menu lisp file, that means, the menu lisp file is empty (except from comments). Therefore, developers may use the LT-Extender menu lisp file to adjust LT-Extender to their specific requirements : you may load application's program files at startup, make

Please keep in mind, that LT-Extender **loads the menu lisp file(s)** (from all currently active menus) **for each drawing** – therefore, please take care, that the lisp code placed into the menu lisp file(s) is intelligent enough not to re-load applications, that are already loaded (especially for *.arx files).
Additionally, you may load your menu file (*.mnu/*.mns/*.mnc) from inside the menu lisp file (using the usual Lisp functions) – the menu lisp file is not loaded twice, because LT will not load any menu lisp files when loading the menu.

Status: using LT-Extender menu lisp files is optionally, **but recommended**

Enabling and Disabling ExpressTools Support under LT

Usually, ExpressTools versions Vol1-8/9 are using some "AcetUtil.fas" (AcetUtil2.fas – AcetUtil4-fas), which can currently not be loaded under LT. Therefore, LT-Extender uses its own emulation of those basic ET functions from its "AcetUtil.lsp" file located in the ".\LT-Extender\Support" and ".\LT-Extender\Support-2k4" folder.

All ExpressTools are completely managed by LT-Extender – this means, LT-Extender will detect the presence of ExpressTools (by some of its specific files), and then load & show the ExpressTools menu and menu lisp as well as the "AcerUtil.lsp" file. If you would like to disable ExpressTools support, simply remove the file "AcetUtil.lsp" from the installation package – then, LT-Extender will completely ignore ExpressTools under LT.

Status: "AcetUtil.lsp" is optionally, **but recommended for ExpressTools Support under LT**

9. Pre-Loading developer's application with LT-Extender (Bundling)

This chapter will describe all possibilities on how to bundle your application with LT-Extender, and will give some assistance and notes.

Basically, "Bundling" consists of 2 major preparations:

1. your installation must include at least the minimum required LT-Extender files; in other words, your installation must also install the LT-Extender files (**Installation Requirements**)
2. you must prepare one of the "AutoStart" mechanisms to have LT-Extender loading your application at LT startup; those prepared LT-Extender files should be installed (**Preparing the Application's AutoLoad**)

Both aspects are described in the following chapters.

Installation Requirements

As described in chapter "3. Minimum required files for LT-Extender" you will need to include those minimum LT-Extender files into your application's installation – but it is recommended to include the optional files too. Of course, you may modify the default LT-Extender files as nessecary (please refer to chapters 4. – 8.).

Recommendations:

1. It is strongly recommended, that you should place and install all LT-Extender files into the root of your application's installation folder (called 'home folder')
2. at the end of your installation process, your installer should run "LTSetup.exe" to register with LT
3. if your application uses a "startmenu group", you should add 2 icons for both "LTSetup.exe" and "LTUninst.exe" – this is nessecary for the user to be able to register/unregister with LT manually; you may also add an icon for LT-Extenders Online help file "LT-Extender-xx.chm" / "LT-Extender.chm"

Preparing the Application's AutoLoad

Usually, your application should be loaded & activated at LT startup, just like as with original AutoCAD. At LT startup, the only application loaded by LTE2000 kernel is LT-Extender – therefore, you will need to have LT-Extender loading your application at LT startup ….

There are several methods available, to include your application into LT-Extender. All methods are recommended in the order as listed below :

Method 1: using the applications Menu & MenuLisp file

If your application uses a menu, you may place all your application's startup program file(s) into the MenuLisp file and load/activate all program code from inside MenuLisp. Your application's MenuLisp file is automatically loaded following the menu file, but may be optionally placed into the AutoLoad list "LT-Extender.prg", to ensure your application will be loaded, even if the menu is not yet loaded in LT.

Integration:

- a) because the MenuLisp is automatically loaded following the Menu, there is no further preparation nessecary; **the Menu is attached to LT by using "LTSetup"**
- b) optionally, place the MenuLisp file into the AutoLoad list "LT-Extender.prg/LT-Extender-2k4.prg"

Suggestion:

- you may also load & show the Menu from inside MenuLisp file, if the Menu is not shown
- take care not to load *.arx files twice from inside MenuLisp – this will create a Lisp error (because the MenuLisp files are loaded for each drawing !)

Regardless of whether you use method a) or b) to activate your application, the MenuLisp file is never loaded multiple times for one drawing – LT-Extender takes special care about loading MenuLisp files !

Also, this method works identical as in AutoCAD, and therefore offers best compatibility for applications simultaneously running with both AutoCAD and AutoCAD LT.

Special notes: All MenuLisp files are loaded for each drawing when opened/created ! Take care with your application's initialization procedure – trying to "(arxload "program").for the already loaded program.arx will result in errors, and therefore, menu lisp processing is canceled.

Method 2: Using LT-Extender's AutoLoad program list

Another method is based on LT-Extender's AutoLoad program list "LT-Extender.prg/LT-Extender-2k4.prg". Developers may place one or more basic startup program file(s) into this AutoLoad list . At LT startup, LT-Extender will process this list and load all defined program file(s).
You may place any types of program files into the AutoLoad list (*.arx, *.dbx, *.lsp, *.mnl and others) – LT-Extender will automatically detect, which way a given program file needs to be loaded.

Some notes: The program files may use all extensions, even unusual ones – LT-Extender will "look into" the file to detect the method, which a particular program file is to be loaded (all arx/dbx/dll files are loaded via "arxload", all other files are loaded as Lisp files via "load"). Therefore, you may use any extension as needed.

Special notes: The AutoLoad program list, and therefore, all defined AutoLoad programs, is processed for each drawing when opened/created ! Take care with your application's initialization procedure – trying to "(arxload "program").for the already loaded program.arx will result in errors, and therefore, menu lisp processing is canceled.

Method 3: Using LT-Extender's MenuLisp file

Similar to method 2, the application's startup program file(s) may be placed into LT-Extender's MenuLisp file "LT-Extender-xx.mnl/LT-Extender-2k4-xx.mnl" (language-dependent mode) or "LT-Extender.mnl/LT-Extender-2k4.mnl" (language-independent mode).

Some notes: To use this method. it is nessecary to ship LT-Extender's Menu together with your application; the LT-Extender Menu is automatically loaded & activated, if present – your application does not need to care about the LT-Extender Menu.

Special note: All MenuLisp files are loaded for each drawing when opened/created ! Take care with your application's initialization procedure – trying to "(arxload "program").for the already loaded program.arx will result in errors, and therefore, menu lisp processing is canceled.

Method 4: Using Classical AutoCAD support files

As in AutoCAD, you may place your applications startup code into the files "Acad.lsp"/"Acad2000.lsp" and/or "AcadDoc.lsp"/Acad2000Doc.lsp". But you should take care about the different ways that these files are handles : "Acad.lsp"/"Acad2000.lsp" are loaded once per AutoCAD/LT session, while "AcadDoc.lsp"/ Acad2000Doc.lsp" are loaded once for each drawing.
Additionally, the file "Acad.rx" may also be used to load Arx program files.

Uninstallation Notes

When uninstalling your application, the uninstallation process must care about included LT-Extender. There is a serious risk to damage those LT installation(s) that your application & LT-Extender were registered to.

The problem: LT-Extender includes our LTE kernel, that will integrate a special dll file (LTE2000.*) into AutoCAD LT. If your application (and therefore, LT-Extender with LTE) is uninstalled, all those LT-Extender and LTE files are removed from disk, for safety, your application should be un-registered from LT before.

Solutions: there are 2 different solutions available to prevent thar conflict :

1. when installing your application, define all LT-Extender and LTE2000 files to be "un-installable"; this should effect these files :**" LT-Extender*.*"** and **"LTE2000*.*"** and **"LTSetup.exe/LTUninst.exe"**
2. your uninstallation process should execute this command line :
   "LTSetup.exe /A /U /S",
   which runs the LT-Extender Un-Registration in Silent mode, un-registering LT-Extender from all LT versions, in generally

The secons method 2.) is recommended, but depends on your Uninstaller's capabilities.
In case, that AutoCAD LT is damaged, it is recommended to simply re-install or repair LT from CD again.

10. Handling LT-Extender License for bundled Applications

This chapter will describe all possibilities on how to bundle your application with LT-Extender and how to manage the LT-Extender Registered/Trial license status, and will give some assistance and notes.

LT-Extender manages its license using the license data file "LT-Extender.dax" – if this file is not present or if the license file does not contain a valid license, LT-Extender will run in trial mode for 200 starts.

There are 2 basic possibilities for bundled applications:

1. **The application is shipped with LT-Extender in Trial mode** – this means, no license file "LT-Extender.dax" is included with the applications package
2. **The application is shipped with LT-Extender in Registered mode** – this means, the "LT-Extender.dax" license file is included with the applications package

LT-Extender in Trial mode

If no license data file "LT-Extender.dax" is shipped with the application, LT-Extender will run in Trial mode for 200 starts. There are several possibilities available to provide a valid LT-Extender license to the user ….

1. LT-Extender can be registered by providing the Registration code, based on Serial number & Program Key from LT-Extender's license dialog – the developer is provided with the "Registration code formula" on demand, to be able to manage their customers LT-Extender license
2. The developer may also send the specially prepared license data file "LT-Extender.dax" (via email etc.), to convert LT-Extender from Trial into Registered mode.

Both the Registration code formula abd/or the specially prepared license data file "LT-Extender.dax" will be provided on demand by developer. This is usually a part of the "Developer / Reseller License Agreement".

LT-Extender in Registered mode

The special license data file "LT-Extender.dax" must be shipped with the application's package – at first start, the license data file will be "dongled" to that machine then.

All developers with a valid "LT-Extender Developer/Reseller License Agreement" are provided with that specially prepared license data file on demand.

11. Using LTE 2000 Kernel system

As an alternative for using (full or customized) LT-Extender, developers can also choice to use LTE 2000 Kernel system for their applications. The LTE 2000 Kernel is an intermediate software layer that enables AutoCAD LT to offer ARX and LISP interfaces for applications – from this point of view, also LT-Extender is a normal application only.

What is the difference ?
If an application is using LTE 2000 Kernel, it is directly connected to AutoCAD LT via LTE 2000 Kernel – there is no LT-Extender involved at all. Therefore, the user will have his AutoCAD LT and the developers application.
Nevetherless, there is no problem to install any other LTE based applicatios like LT-Extender, LayerManager or whatever applications – LTE 2000 Kernel is designed to support unlimited number of applications.

What does LTE 2000 Kernel offer ?
Basically, LTE 2000 Kernel offers these major features :

1. providing the ARX interface for applications
2. providing the LISP interface for applications
3. providing DemandLoad feature for ARX applications
4. LTE manages unlimited number of applications simultaneously
5. comfortable registration and un-registration with AutoCAD LT with LTSetup.exe/LTUninst.exe tools with full command line support

LTE 2000 Kernel basics

The LTE 2000 Kernel is an intermediate software layer that activates ARX and LISP interfaces in AutoCAD LT; additionally, LTE 200 Kernel manages all installed applications based on LTE Kernel.

At LT startup, the LTE menu is loaded and the appropriate menu DLL file.
Then, the integrated version management will detect all installed LTE 2000 Kernels (via Registry) and analyze their versions – then, the highest version of LTE 2000 Kernel (LTE2000.sys, LTE2000.mod, LTE2000.msg resp. LTE2004.sys, LTE2004mod, LTE2004.msg) is loaded, which takes control of ARX and LISP interfaces.

At last step, LTE 2000 Kernel will analyse all installed LTE based applications, and will load all files defined by all "Loader Definition" files from all applications.

LTE 2000 Kernel is designed to manage an unlimited number of applications – this means, there is no logical or technical conflict if LTE 2000 Kernel is installed several times in different folders. The internal version management will manage the LTE 2000 Kernel files on disk

LTE 2000 Kernel Licensing system

LTE Kernel does **not** use any license system nor does it use trial period or any other restrictions at all !
Because LTE Kernel is 'attached' to the application with a specific "Loader Definition" file, there is no need to use any license system for the LTE Kernel. This means, the application specific LTE Kernel can not be used with any other software package – it is limited to that particular application.

Therefore, applications can still use their normal licensing system as with regular AutoCAD, without any differences.

LTE 2000 Kernel files

Application-independent files :

LTE2000.mns/mnr/mnc, LTE2000.dll	LTE 2000 menu files
LTE2000.sys, LTE2000.mod, LTE2000.msg	LTE 2000 Kernel files (LT 2000-2002)
LTE2004.sys, LTE2004.mod, LTE2004.msg	LTE 2000 Kernel files (LT 2004)
Unc..dll, Unc95.dll, PsApi.dll	UNC and process management helper files
LTSetup.exe, LTUninst.exe	Register/Un-Register tool
- LTE2004Lisp.mod/ext	Lisp processor for LT 2004

Application-specific-files :

LTE2000.ldr	applications loader definition (LT2000-2002)
LTE2004.ldr	applications loader definition (LT2004)
LTSetup.msg	optional: message definition
LTSetup.pl	**optional:** support path & menu definition (LT 2000-2002)
LTSetup-2k4.pl	**optional:** support path & menu definition (LT 2004)

All LTE 2000 Kernel files should be located in the same folder as the applications main program files – exactly : the LTE 2000 Kernel files must be located in the same folder as the files defined in the "Loader Definition" file. Alternatively, the LTE 2000 Kernel will also search the standard support pathes list to locate the applications's defined startup program files.

The application-specific "Loader Definition" LTE2000.ldr/LTE2004.ldr is provided by TM-CAD Engineering on developers request, based on the "Developers License Agreement".

Connecting LTE 2000 Kernel to the application

The application-independent LTE 2000 Kernel always uses an application-specific "Loader Definition" files LTE2000.ldr/LTE2004.ldr, which connect that particular application with its LTE Kernel. That loader definition files control which arx/lisp program files shall be loaded by LTE Kernel at LT startup.

The "Loader Definition" contains informations about the applications name and the startup program file(s) to be loaded – both arx and lisp applications can be used.
…
Those application's startup file(s) can be both Arx and Lisp files, in exact sequence – usually, we suggest to use the menu lisp file (if a menu is used by the application) or any other "boot-up" lisp file : this way, the developer has all freedom to modify the startup process without requesting a new "Loader Definition" file; all modifications to the application's startup process can be made via the lisp file.

So each application will need its own "Loader Definition" file, which is provided by TM-CAD Engineering on request of developers, based on their "Developer License Agreement".

General notes and hints related to LTE 2000 Kernel

Installation :
when installing the developer's application, the installer should call "LTSetup.exe /A /S" - this will automatically register the application with all existing LT installations, in silent mode

Un-Installation :
when un-installing the developer's application, the un-installer should first call "LTSetup.exe /U /A /S" or "LTUninst.exe /A /S" - this will automatically un-register the application from all existing LT installations, in silent mode

Command line switches :
all command line switches for LTSetup.exe/LTUninst.exe are described in a separate chapter above – please have a look there.

Startmenu program group :
We usually suggest to include 2 shortcuts to both "LTSetup.exe" and "LTUninst.exe" – this way, the user can manually activate and deactivate the application from specified LT versions.

Adding / removing application specific support pathes :
The LTSetup.exe and LTUninst.exe provide a very comfortable method to automatically add support search pathes to that LT versions, where the application registers to, resp. to remove those support search pathes when the application un-registers from LT. These support pathes are defined by the optional "LTSetup.pl/LTSetup-2k4.pl" files.
This is described in above chapter – please have a look there; alternatively, the syntax is also explained in "LTSetup.pl/LTSetup-2k4.pl" files as comments.

Adding / removing application specific menu :
The LTSetup.exe and LTUninst.exe also provide a very comfortable method to automatically add the application specific menu (if provided) to that LT versions, where the application registers to, resp. to remove that menu when un-registering from LT. This menu is defined by the optional "LTSetup.pl/LTSetup-2k4.pl" files.
This is described in above chapter – please have a look there; alternatively, the syntax is also explained in "LTSetup.pl/LTSetup-2k4.pl" files as comments.

Using different language for LTSetup.exe / LTUninst.exe :
Internally, LTSetup.exe and LTUninst.exe offer German and English language.
If needed, any other language can be used by providing the (optional) file LTSetup.msg – developers can translate the text entries to any other language.

12. Bugs, Issues, Defects and 'Strange Effects' with ADS/ARX/Lisp APIs under LT

This chapter decribes known bugs, issues, defects and other 'strange effects' with the ADS/ARX/Lisp APIs when running under AutoCAD LT. Primarily, this defect lists are based on our own experiences and those of third party developers involved with LT-Extender …..
This list is permanently maintained – so if any developer will find those bugs, issues and defects, please contact us at Developer@LT-Extender.com , so we can improve and extend these lists – thank you !

Common Bugs, Issues, Defects with APIs under LT

Here some common or unspecified issues with programming APIs in LT and with LT in generally are documented.

CmdActive system variable returns undocumented values
in example : (getvar "CmdActive") sometimes returns 32 instead of 0.
The only documented values are these bit-values 1, 2, 4, 8, 16 (Bit 0…4). If programs will check for Zero, they will usually fail …
Solution: always mask (bitwise-AND, &, logand) the return value with 31, which is the sum of 1+2+4+8+16; for Lisp programs, LTE kernel overrides (getvar…) function and will automatically adjust the return value by masking with 31 – for performance reasons, LTE kernel does hook into acedGetVar functions for all loaded applications (for both ARX and LISP), but applications should mask the return value by their own code to be most stable if LTE Kernel mistakenly did not hook into all modules.

Missing Commands under LT

There are a number of AutoCAD Commands missing under LT which can therefore not be reactivated – below, you will find a list of those commands :

_Group, _-Group : the Group commands are normally defined in "Acad.exe", but missing in "Aclt.exe";
under LT, the "Group" and "-Group" commands seems to be defined in *.fas file – therefore, it is not possible to use (command "_group") from Lisp and acedCommand(RTSTR, "_Group", NULL) from Ads/Arx
Solution:
Group commands are implemented as "_PkFstGroup" and "_-PkFstGroup" in Aclt.exe – so applications can detect wether they are running under LT and then use PkFstGroup instead of Group commands

Missing AutoCAD System Variables under LT

There are a number of AutoCAD Ssstem variables missing under LT – here is the list of those system variables:

- **AcadPrefix** – returns the Support Path list as string (devided by ';')
- **LoginName** – returns the name of the current user logged-in

LTE Kernel 'overrides' the **getvar** function for both Lisp and ARX interfaces, in order to check & emulate those missing system variables. Therefore, the following system variables are emulated and work as in AutoCAD :

- **AcadPrefix**
- **LoginName**

Missing AutoCAD Environment Variables under LT

There are some AutoCAD environment variables missing under LT – here is the list of those variables:

- **AcadCfg** – returns the folder of current configuration file

LTE Kernel 'overrides' the **getenv** function for both Lisp and ARX interfaces, in order to check & emulate those missing environment variables. Therefore, these environment variables are emulated and work as in AutoCAD :

- **AcadCfg** – returns the folder of Aclt.exe (where the CFG file normally resides)

Bugs, Issues, Defects with Lisp under LT

1. ActiveX and VBA are not available

Currently, the main limitation is the unavailability of ActiveX and VBA under AutoCAD LT. This causes all those advanced VisualLisp functions, based on ActiveX/VBA to be not present. Therefore, these functions are not present:

- all **(vlr-…..)** functions for reactor management
- all **(vla-….)** ActiveX based functions
- all **(vlax-...)** ActiveX based functions

Because VisualLisp IS active when running LT, all other **(vl-…)** VisualLisp functions may be used without any limitations – they work as they do with AutoCAD.

Functions, which are emulated by LT-Extender:

- (vl-load-com), (vl-load-reactors)
  basically, this is a "No-Operation" to prevent Lisp programs from being cancelled when loaded
- (vlax-product-key)
  this function return the Registry key for LT exactly like it works in AutoCAD

Bugs, Issues, Defects with Ads/Arx under LT

Basically, there are no major problems with Ads/Arx applications running under LT. Only a very less number of defects and 'strange effects' are currently known ….

General notes:
Of course, all developers should use the Ads/Arx API from AutoCAD 2000 – because the APIs from AutoCAD 2000i and 2002 are slightly different from the 2000 API. Otherwise, any Ads/Arx programs developed using the 2000i and 2002 API will not work properly under AutoCAD+LT 2000. In other words, the APIs are upward compatible, but not downward compatible.

1. Determing the applications HINSTANCE handle

Applies to : all LT versions

A very common mistake happens, when an Ads/Arx application needs to know its own HINSTANCE handle to be used with MFC.

From AutoCAD R14, the following method is often taken into R2000 based applications, to determine the HINSTANCE :

```
HINSTANCE  myInstance = ::GetModuleHandle("myProgram.arx");
```

This method will fail, because Ads/Arx applications loaded into AutoCAD LT are specially prepared and will therefore use a different filename for the module to load: **myProgram.ltx**. Because no file "myProgram.arx" is loaded, the returned handle is NULL.

There are 2 solutions available to correct this problem :
Method 1:
This will additionally check for both applications' filenames – so it works under AutoCAD and LT.

```
HINSTANCE  myInstance = ::GetModuleHandle("myProgram.arx");
if (myInstance == NUL) myInstance = ::GetModuleHandle("myProgram.ltx");
```

Method 2: (suggested)
Since AutoCAD 2000 API, all Ads/Arx programs are normal DLL files, which includes the **"DllMain"** function, that is called by Windows after the file is loaded into memory. This DllMain functions provied that HINSTANCE value for the loaded Ads/Arx file – therefore, the application only needs to save that hInstance value.

```
extern "C" int APIENTRY DllMain (HINSTANCE hInstance, DWORD dwReason, LPVOID lpReserved)
{
	UNREFERENCED_PARAMETER(lpReserved);

	if (dwReason == DLL_PROCESS_ATTACH) arxmfcDLL.AttachInstance(hInstance);
	if (dwReason == DLL_PROCESS_ATTACH) _hdllInstance=hInstance; // stored into global variable

	if (dwReason == DLL_PROCESS_DETACH) arxmfcDLL.DetachInstance();

	return 1;
}
```

2. AcRx::kPreQuitMsg message not sent to acrxEntryPoint

Applies to : LT 2000 only

This message in not sent to the application, when LT is closed (either by command "_quit", or menu item "Close" or "Close icon" from LT mainframe window).

Therefore, do not use this place for un-initialization of your applications objects and memory ! Use the AcRx::kQuitMsg for your cleanup tasks.

3. AcEditorReactor with different Case-Sensitivity in Reactor functions

Applies to : all LT version

From modulCAD (www.modulcad.com) we got the following 'mystery' with AcEditorReactor issues :

The "Command related" functions of AcEditorReactor shows command names in both lower and upper case – differently to AutoCAD, which always uses uppercase names !

Therefore, always check command names with case-insensitive code like "CString::CompareNoCase()" resp. ""stricmp()"; additionally, it seems to be possible, that AutoCAD LT will also use uppercase/lowercase mixture in many other functions, where command names, entity types, or any other ASCII parameter are used !

4. AcProfileManager and related Classes are not present

Applies to : all LT 2000-2002 version, does not apply to lT 2004

Under all LT versions, the AcProfileManager and all related Classes (like profile reactor etc.) are not present at all. Because the applications .arx/.ltx will not be loaded (for unresolved dependencies to AcProfileManager etc.) there is no other chance than to remove all code related to AcProfileManager and related classes from your applications code.

5. using acrxLoadModule() instead of acrxDynamicLinker->loadModule()

Applies to : all LT version

LTE Kernel monitors and hooks into several ARX API calls to manage loading and executing ARX applications. For several technical reasons, there are some problems related to acrxDynamicLinker->loadModule() which can not be monitored completely (this is a "virtual" declared function).

Therefore: We strongly suggest using acrxLoadModule() instead. This API function is a wrapper function for acrxDynamicLinker->loadModule() with same argument syntax; LTE Kernel can monitor this wrapper function without any problems.