Installing Logtalk

This page provides an overview of Logtalk installation requirements and instructions and a description of the files contained on the Logtalk distribution. For detailed, up-to-date installation and configuration instructions, please see the `README.txt`, `INSTALL.txt`, and `CUSTOMIZE.txt` files distributed with Logtalk. The broad compatibility of Logtalk, both with Prolog compilers and operating-systems, together with all the possible user scenarios, means that installation can vary from very simple by running an installer or a couple of scripts to the need of patching both Logtalk and Prolog compilers to workaround the lack of strong Prolog standards.

The preferred installation scenario is to have Logtalk installed in a system-wide location, thus available for all users, and a local copy of user-modifiable files on each user home directory (even when you are the single user of your computer). This scenario allows each user to independently customize Logtalk and to freely modify the provided programming examples. Logtalk installers, installation shell scripts, and Prolog integration scripts favor this installation scenario, although alternative installation scenarios are always possible.

Hardware & software requirements

Computer and operating system

Logtalk is compatible with almost any computer/operating-system with a modern Prolog compiler available. Currently, the main development environment is an Apple MacBook Pro running MacOS X. Being written in Prolog and distributed in source form, the only issue regarding operating system compatibility are the end-of-line codes in the source text files!

Prolog compiler

In writing Logtalk I have tried to follow the current ISO Prolog Part 1 standard whenever possible. Capabilities needed by Logtalk that are not defined in the ISO standard are:

- access to predicate properties (`dynamic`, `static`, `built_in`)

Logtalk needs access to the predicate property `built_in` to properly compile objects and categories that contain Prolog built-in predicates calls. In addition, some Logtalk built-ins need to know the dynamic/static status of predicates to ensure correct application. The ISO standard for Prolog modules defines a `predicate_property/2` predicate that is already implemented by most Prolog compilers. Note that if these capabilities are not built-in the user cannot easily define them.

For optimal performance, Logtalk requires that the Prolog compiler supports **first-argument indexing** for both static and dynamic code (most modern compilers support this feature).

Since most Prolog compilers are moving closer to the ISO Prolog standard [ISO 95], it is advisable that you try to use the most recent version of your favorite Prolog compiler.

Logtalk installers

Logtalk installers are available for MacOS X, Linux, and Microsoft Windows 2000/XP/2003. Depending on the chosen installer, some tasks (e.g. setting environment variables or integrating Logtalk with some Prolog compilers) may need to be performed manually.

Source distribution

Logtalk sources are available in a `tar` archive compressed with `gzip`, `lgt2xxx.tgz`. You may expand the archive by using a decompressing utility or by typing the following commands at the command-line:

```
% tar -zxvf lgt2xxx.tgz
```

This will create a sub-directory named `lgt2xxx` in your current directory. Almost all files in the Logtalk distribution are text files. Different operating-systems use different end-of-line codes for text files. Ensure that your decompressing utility convert the end-of-lines of all text files to match your operating system.

Directories and files organization

In the Logtalk installation directory, you will find the following files and directories:

- `BIBLIOGRAPHY.bib` - Logtalk bibliography in BibTeX format
- `INSTALL.txt` - Logtalk installation instructions
- `LICENSE.txt` - Logtalk user license
- `QUICK_START.txt` - Quick start instructions for those that do not like to read manuals
- `README.txt` - several useful information
- `RELEASE_NOTES.txt` - release notes for this version
- `UPGRADING.txt` - instructions on how to upgrade your programs to the current Logtalk version
- `compiler`
    - `NOTES.txt` - notes on the current status of the compiler
    - `...` - compiler source files
- `configs`
    - `NOTES.txt` - notes on the provided configuration files
    - `template.config` - template configuration file
    - `...` - specific configuration files
- `contributions`
    - `NOTES.txt` - notes on the user-contributed code
    - `...` - user-contributed code files
- `examples`
    - `NOTES.txt` - short description of the provided examples
    - `bricks`
        - `NOTES.txt` - example description and other notes
        - `SCRIPT.txt` - step by step example tutorial
        - `loader.lgt` - loader utility file for the example objects
        - `...` - bricks example source files
    - `...` - other examples
- `libpaths`
    - `NOTES.txt` - description on how to setup library and examples paths
    - `libpaths.pl` - default library and example paths
- `library`
    - `NOTES.txt` - short description of the library contents
    - `all_loader.lgt` - loader utility file for all library entities
    - `...` - library source files
- `manuals`
    - `NOTES.txt` - notes on the provided documentation
    - `bibliography.html` - bibliography
    - `glossary.html` - glossary
    - `index.html` - root document for all documentation
    - `...` - other documentation files
- `scripts`
    - `NOTES.txt` - notes on scripts for Logtalk packaging, installation, and integration with Prolog compilers
    - `...` - packaging, installation, and Prolog integration scripts
- `wenv`
    - `NOTES.txt` - notes on the text editor syntax configuration files providing syntax coloring for editing Logtalk source files
    - `...` - syntax coloring configuration files
- `xml`
    - `NOTES.txt` - notes on the automatic generation of XML documentation files
    - `logtalk.css` - cascade style sheet file for the HTML output of the XSLT conversion of the XML files

`logtalk.dtd` - Document Type Description file describing the structure of the XML files
`lgthtml.xsl` - transformation style sheet to output HTML code from the XML files
`...` - other XSL files
`lgt2html.sh` - shell script for converting XML documenting files to PDF files
`...` - other script files

Configuration files

Configuration files provide the glue code between the Logtalk preprocessor/runtime and a Prolog compiler. Each configuration file contains two sets of predicates: ISO Prolog standard predicates and directives not built-in in the target Prolog compiler and Logtalk-specific predicates.

Logtalk already includes ready to use configuration files for most academic and commercial Prolog compilers. If a configuration file is not available for the compiler that you intend to use, then you need to build a new one, starting from the included `template.config` file. Start by making a copy of the template file. Carefully check (or complete if needed) each listed definition. If your Prolog compiler conforms to the ISO standard, this task should only take you a few minutes. In most cases, you can borrow code from some of the predefined configuration files. If you are unsure that your Prolog compiler provides all the ISO predicates needed by Logtalk, try to run the system by setting the unknown predicate error handler to report as an error any call to a missing predicate. Better yet, switch to a modern, ISO compliant, Prolog compiler. If you send me your configuration file, with a reference to the target Prolog compiler, maybe I can include it in the next release of Logtalk.

The configuration files specifies default values for all the Logtalk compiler flags. Most of these compiler flags are described in the next section. A few of these flags have read-only values which cannot be changed at runtime. These are:

`startup_message`
: Controls the messages printed by Logtalk at startup. Possible flag values are `flags(verbose)` (the usual default; prints the Logtalk banner and a list of all default compiler flags), `flags(compact)` (a more compact listing of the default compiler flags), `banner` (prints only the Logtalk banner), and `none` (suppress all startup messages; useful for batch processing).

`altdirs`
: Allows the use of alternative directories for storing the Prolog files and the XML documenting files resulting from entity compilation. This flag can only be set for Prolog compilers providing the necessary operating-system access predicates for implementing this feature (specifically, a built-in predicate for creating new directories). Possible option values are `on` and `off` (the usual default). The names of the alternative directories are specified using the predicate `'$lgt_alt_directory'/2`, which is also defined in the configuration files. Make sure the directory names are valid for your operating system before setting this flag on.

`supports_break_predicate`
: Informs Logtalk if the Prolog compiler supports a `break/0` predicate (which can be used by the Logtalk built-in debugger). Possible flag values are `true` and `false`.

`supports_encoding_dir`
: Informs Logtalk if the Prolog compiler supports the `encoding/1` directive. This directive is used for declaring the text encoding of source files. Possible flag values are `true` and `false`.

`threads`
: Informs Logtalk if the Prolog compiler provides suitable multi-threading programming support. Possible flag values are `unsupported`, `on`, and `off`. The default flag vale for compatible Prolog compilers is `off`, i.e. multi-threading programming support is disabled by default.

Logtalk compiler and runtime

The `compiler` sub-directory contains the Prolog source file(s) that implement the Logtalk preprocessor/compiler and the Logtalk runtime. The compiler and the runtime may be split in two (or more) separate files or combined in a single file, depending on the Logtalk release that you are installing.

Library

Starting from version 2.7, Logtalk contains a standard library of useful objects, categories, and protocols. Read the corresponding `NOTES.txt` file for details about the library contents.

Examples

Logtalk 2.x contains new implementations of some of the examples provided with previous 1.x versions. The sources of each one of these examples can be found included in a subdirectory with the same name, inside the directory examples. The majority of these examples include a file named `SCRIPT.txt` that contains cases of simple utilization. Some examples may depend on other examples and library objects to work properly. Read the corresponding `NOTES.txt` file for details before running an example.

Logtalk source files

Logtalk source files are text files containing entity definitions (objects, categories, or protocols). The extension `.lgt` is normally used. The Logtalk preprocessor compiles these files to plain Prolog, replacing the `.lgt` extension with `.pl` (the default Prolog extension). If your Prolog compiler expects the Prolog source filenames to end with a specific, different extension, you can set it in the corresponding configuration file.

Last updated on: October 26, 2005