

## User Manual

- Logtalk features
- Logtalk nomenclature
- Message sending
- Objects
- Protocols
- Categories
- Predicates
- Inheritance
- Event-driven programming
- Multi-threading programming
- Error handling
- Documenting Logtalk programs
- Installing Logtalk
- Running and debugging Logtalk programs
- Programming in Logtalk

## Logtalk features

- Integration of logic and object-oriented programming
- Integration of event-driven and object-oriented programming
- Support for component-based programming
- Support for both prototype and class-based systems
- Support for multiple object hierarchies
- Separation between interface and implementation
- Private, protected, and public inheritance
- Private, protected, and public object predicates
- Parametric objects
- Smooth learning curve
- Compatibility with most Prologs and the ISO standard
- Performance

## Logtalk nomenclature

- C++ nomenclature
- Java nomenclature

## Message sending

- Operators used in message sending
- Sending a message to an object
- Broadcasting
- Sending a message to *self*
- Calling an overridden predicate definition
- Message sending and event generation
- Message sending performance

## Objects

- Objects, prototypes, classes, and instances
- Defining a new object
- Parametric objects
- Finding defined objects
- Creating a new object in runtime
- Abolishing an existing object
- Object directives
  - Object initialization
  - Dynamic objects
  - Object dependencies
  - Object documentation
- Object relationships
- Object properties
- The pseudo-object user
- The pseudo-object debugger

## Protocols

- Defining a new protocol
- Finding defined protocols
- Creating a new protocol in runtime
- Abolishing an existing protocol
- Protocol directives
  - Protocol initialization
  - Dynamic protocols
  - Protocol documentation
- Protocol relationships
- Protocol properties
- Implementing protocols

## Categories

- Defining a new category
- Finding defined categories
- Creating a new category in runtime
- Abolishing an existing category
- Category directives
  - Category initialization
  - Dynamic categories
  - Category dependencies
  - Category documentation
- Category relationships
- Category properties
- Importing categories

## Predicates

Declaring predicates

Scope directives

Mode directive

Meta-predicate directive

Discontiguous directive

Dynamic directive

Operator directive

Uses directive

Alias directive

Documenting directive

Defining predicates

Object predicates

Category predicates

Meta-predicates

Definite clause grammars

Built-in object predicates (methods)

Execution context methods

Database methods

Meta-call methods

All solutions methods

Reflection methods

Definite clause grammar parsing methods

Term expansion methods

Predicate properties

Finding declared predicates

Calling Prolog built-in predicates

Calling Prolog non-standard meta-predicates

## Inheritance

- Protocol inheritance
  - Search order for prototype hierarchies
  - Search order for class hierarchies
- Implementation inheritance
  - Search order for prototype hierarchies
  - Search order for class hierarchies
  - Inheritance versus predicate redefinition
- Public, protected, and private inheritance
- Composition versus multiple inheritance

## Event-driven programming

- Definitions
  - Event
  - Monitor
- Event generation
- Communicating events to monitors
- Performance concerns
- Monitor semantics
- Activation order of monitors
- Event handling
  - Finding defined events
  - Defining new events
  - Abolishing defined events
  - Defining event handlers

## Multi-threading programming

- Enabling multi-threading support
- Object threads
- Multi-threading built-in predicates
  - Proving goals asynchronously using threads
  - Retrieving asynchronous goal proof results
  - One-way asynchronous calls
  - Atomic goals and asynchronous calls
- Competing goals
- Atomic predicates

## Error handling

- Compiler warnings and errors
  - Unknown entities
  - Singleton variables
  - Redefinition of Prolog built-in predicates
  - Redefinition of Logtalk built-in predicates
  - Redefinition of Logtalk built-in methods
  - Misspell calls of local predicates
  - Portability warnings
  - Other warnings and errors
- Runtime errors
  - Logtalk built-in predicates
  - Logtalk built-in methods
  - Message sending

## Documenting Logtalk programs

- Documenting directives
  - Entity directives
  - Predicate directives
- Processing and viewing documenting files

## Installing Logtalk

- Installing Logtalk
- Hardware & software requirements
  - Computer and operating system
  - Prolog compiler
- Logtalk installers
- Source distribution
- Directories and files organization
  - Configuration files
  - Logtalk compiler and runtime
  - Library
  - Examples
  - Logtalk source files

## Running and debugging Logtalk programs

- Running a Logtalk session
  - Starting Logtalk
  - Compiling and loading your programs
  - Compiler flags
  - Smart compilation of source files
- Using Logtalk for batch processing
- Debugging Logtalk programs
  - Compiling objects in debug mode
  - Logtalk Procedure Box model
  - Defining spy points
  - Tracing program execution
  - Debugging using spy points
  - Debugging commands

## Programming in Logtalk

- Writing programs
  - Source files
  - Loader utility files
  - Libraries of source files
  - Portable programs
  - Avoiding common errors
  - Coding style guidelines
- Logtalk scope

Copyright © Paulo Moura — Logtalk.org  
XHTML + CSS Last updated on: September 17, 2006