Jekejeke Runtime Installation

Version 1.0.10, October 20th, 2015

XLOG Technologies GmbH

Jekejeke Prolog

Runtime Library 1.0.10

Installation Guide

Author: XLOG Technologies GmbH
Jan Burse
Freischützgasse 14
8004 Zürich
Switzerland

Date: October 20th, 2015
Version: 0.25

Participants: None

Warranty & Liability

To the extent permitted by applicable law and unless explicitly otherwise agreed upon, XLOG Technologies GmbH makes no warranties regarding the provided information. XLOG Technologies GmbH assumes no liability that any problems might be solved with the information provided by XLOG Technologies GmbH.

Rights & License

Trademarks

Table of Contents

Change History

Jan Burse, April 17th, 2011, 0.1:
- Derived from development environment installation guide.

Jan Burse, Mai 6th, 2011, 0.2:
- Release notes for 0.8.9 added and documentation section included.

Jan Burse, June 15th, 2011, 0.3:
- Release notes for 0.9.0 added and samples section included.

Jan Burse, September 16th, 2011, 0.4:
- Release notes for 0.9.1 added.

Jan Burse, November 4th, 2011, 0.5:
- Release notes for 0.9.2 added.

Jan Burse, February 23th, 2012, 0.6:
- Release notes for 0.9.3 added.

Jan Burse, June 4th, 2012, 0.7:
- License conversation moved to here and release notes for 0.9.4 added.

Jan Burse, August 24th, 2012, 0.8:
- Release notes for 0.9.5 added and Android top-level sub section introduced.

Jan Burse, September 6th, 2012, 0.9:
- Release notes for 0.9.6 added.

Jan Burse, October 30th, 2012, 0.10:
- Release notes for 0.9.7 added.

Jan Burse, February 20th, 2013, 0.11:
- Release notes for 0.9.8 added and system requirements sections moved.

Jan Burse, May 7th, 2013, 0.12:
- Release notes for 0.9.9 added.

Jan Burse, June 9th, 2013, 0.13:
- Release notes for 0.9.10 added.

Jan Burse, August 10th, 2013, 0.14:
- Release notes for 0.9.11 added and known issues sections moved.

Jan Burse, November 11th, 2013, 0.15:
- Release notes for 0.9.12 added and automatic discovery introduced.

Jan Burse, March 6th, 2014, 0.16:
- Release notes for 1.0.0.

Jan Burse, March 24th, 2014, 0.17:
- Release notes for 1.0.1.

Jan Burse, June 4th, 2014, 0.18:
- Release notes for 1.0.3 and known issues section updated.

Jan Burse, August 22sd, 2014, 0.19:
- Release notes for 1.0.4.

Jan Burse, January 1st, 2015, 0.20:
- Release notes for 1.0.5.

Jan Burse, April 22nd, 2015, 0.21:
- Release notes for 1.0.6 and title page introduced.

Jan Burse, July 10th, 2015, 0.22:
- Release notes for 1.0.7.

Jan Burse, August 9th, 2015, 0.23:
- Release notes for 1.0.8.

Jan Burse, September 1st, 2015, 0.24:
- Release notes for 1.0.9.

Jan Burse, October 20th, 2015, 0.25:
- Release notes for 1.0.10.

1 Introduction

The Jekejeke Prolog runtime library is available as a platform independent archive. Customers can also download documentation and samples. In the following we describe the download contents and its most basic use.

- **Release 1.0:** This section lists the changes concerning the Jekejeke Prolog runtime library, documentation and samples.

- **Release 0.9:** This section lists the changes concerning the Jekejeke Prolog runtime library, documentation and samples.

- **Release 0.8:** This section lists the changes concerning the Jekejeke Prolog runtime library, documentation and samples.

- **Library Installation:** Here we describe how the Jekejeke Prolog runtime library can be installed and used.

- **Support Files Installation:** Here we describe how the Jekejeke Prolog runtime library support files can be installed.

- **License Conversations:** The Jekejeke Prolog runtime library provides character terminal based interactions. Among the interactions we find license management.

2 Release 1.0

This section lists the changes concerning the Jekejeke Prolog runtime library executable, documentation and samples:

2.1 Release 1.0.10

The following features and bug fixes were provided for the Jekejeke Prolog runtime library of version 1.0.10:

Language

- New predicate sys_register_file/1 in Prolog text load introduced.
- New predicate current_resource/1 in Prolog text source introduced.
- New predicate sys_load_resource/1 in Prolog text module introduced.
- New predicate sys_add_resource/1 in Prolog text module introduced.
- New Prolog flag sys_belongs_to introduced.
- New predicate sys_term_eq_list/2 introduced.
- The autoloader for Java classes now also re-exports the implemented interfaces.
- Foreign predicate parameter type Term now checks for callable/1.
- Foreign predicate parameter type Object now gives String instead TermAtom.
- Foreign predicate return type Object now accepts String besides TermAtom.
- The predicate special_bridge/2 removed.
- New predicate special_bridge/3 introduced.
- New predicates set_atom_property/3 and reset_atom_property/3 introduced.
- Experimental module prefix allowed in predicate reexport/1 now.
- .

Frequent Predicates

- New module proxy introduced.
- .

Programming Interface

- The reference datatype is comparable if the wrapped object implements comparable.
- The class TermRef does not anymore provide a comparator constructor.
- The class TermRef does not anymore provide a comparator getter.
- The methods unfoldFirst() and unfoldNext() removed from class Interpreter.
- The method unfoldClose() removed from class Interpreter.
- The methods unfoldCut() and unfoldCleanup() removed from class Interpreter.
- New methods iterator() with and without result in class Interpreter.

- New methods hasNext(), next() and close() in class CallIn introduced.
- New methods cut() and cleanup()in class CallIn introduced.
- New constructor that takes another interpreter in class Interpreter introduced.
- The method createCompound() in the class TermAtom is now deprecated.
- New constructors in class TermCompound with and without interpreter.
- New method iterator() without goal in class Interpreter introduced.
- The class CallFrame has been removed.
- The method unfoldChecked() removed from class Interpreter.
- The method unifyTerm() in the class Interpreter is now deprecated.
- New method unifyTerm() in class Term introduced.
- New method copyTerm() in class Term introduced.
- New methods nextClose() and hasNextClose() in class CallIn introduced.
- New convenience atom ATOM_TRUE in class Knowledgebase introduced.
- The initialization methods from Interpreter class moved to Knowledgebase class.
- New constructor that takes a hint in class Knowledgebase introduced.
- The constructor Interpreter from class Interpreter is not anymore public.
- New method iterable() in class Knowledgebase introduced.
- New static method currentInterpreter() in class Interpreter introduced.
- New class Controller introduced.
- The signal methods from Interpreter class moved to Controller class.
- New method getController() in class Interpreter introduced.
- New method iterable() in class Interpreter introduced.
- The parse methods from Interpreter class moved to Term class.
- The classes TermInteger, TermFloat and TermDecimal eliminated.
- Methods for integers, floats and decimals moved to class TermAtom.
- The class TermRef eliminated.
- The class TermVar not anymore decadent from class Term.
- API methods now accept Strings besides TermAtom.
- API methods now return Strings.
- New API xxxWrapped() methods introduced that return TermAtom.
- Java package jekpro.tools.api eliminated.
- New Java package jekpro.tools.term introduced.
- New Java package jekpro.tools.call introduced.
- ArithmeticException now thrown by failed guardFloat() and guardDouble().
- ArithmeticException now thrown by failed compare().
- ArithmeticException thrown by foreign function now mapped to evaluation_error/1.
- New exception class RuntimeWrap introduced.
- .

2.2 Release 1.0.9

The following features and bug fixes were provided for the Jekejeke Prolog runtime library of version 1.0.9:

Language

- Indexes are now built by analysing the clause body for var/1 guards.
- Indexes are now accessed by respecting the guard analysis.
- The wrapper sys_capture/1 is not anymore needed for clause/2, retract/1, etc...
- The wrapper sys_capture/1 is not anymore needed for predicate_property/2, etc..
- The wrapper sys_capture/1 is not anymore needed for current_predicate/1, etc..
- New predicate sys_parent_goal/1 to dynamically analyse the call-site introduced.
- New directive foreign_constructor/3 for registering a constructor as a predicate.
- New directive foreign_setter/3 for registering a field write as a predicate.
- New directive foreign_getter/3 for registering a field read as a predicate.
- The foreign directives now work for non-static members.
- The foreign directives now work for primitive data types.
- The foreign directives now work the Number class.
- New directive foreign_function/3 for registering a method as an evaluable function.
- New directive foreign_constant/3 for registering a field as an evaluable function.
- The directive override/1 now also accepts evaluable functions.
- The override style check is now also performed for evaluable functions.
- The implementation style check is now also performed for evaluable functions.
- New Prolog flag sys_random introduced.
- New directive special_bridge/2 introduced.
- New directive auto_load/1 introduced.
- The source property import/1 removed.
- The source property export/1 removed.
- New source property sys_link/2 introduced.
- The predicate abolish/1 now also allows removing evaluable functions.
- The predicate abolish/1 now also allows removing syntax operators.
- Override style check for syntax operators during loading introduced.
- New operator property override/0 introduced.
- The directive override/1 now also accepts syntax operator indicators.
- Implementation style check for syntax operators after load introduced.
- The directive sys_source_preload/0 removed.
- New automatic preload marking during capability load.
- Text file read is now by default done with BOM detection.
- Consult is now by default done with BOM detection.
- The predicate listing/1 now also allows listing evaluable functions.
- The colon notation (:)/2 can now be used in evaluable expressions.
- Binary and unary rounding ops on decimals give now decimals.
- New data type small float with prefix “0f” introduced.
- New predicate float32/1 introduced.
- New evaluable function float32/1 introduced.

Frequent Predicates

- Missing context variable sharing in predicate foreach/2 fixed.
- New module experiment/surrogate introduced.
- New module advanced/sequence introduced.
- New module basic/hyper introduced.

Programming Interface

- The class TermDecimal now provides some widening and norming methods.
- The class TermFloat now provides some widening and norming methods.
- The class TermInteger now provides some widening and norming methods.
- Use of the Prolog flag sys_context was eliminated from the interpreter.
- New multi-file predicate sys_foreign_suffix/1 introduced.
- The absolut_file_name/[2,3] predicates now accept verbatim/1 specifier.
- The absolut_file_name/[2,3] predicates now accept foreign/1 specifier.
- Loading the verbatim/1 specifier results in an empty synthetic module.
- Loading the foreign/1 specifier results in a synthetic module from the Java class.
- New auto loader first tries library/1 from module name.
- New auto loader then tries foreign/1 from module name.
- New auto loader finally tries verbatim/1 from module name.
- New predicate integer32/1 introduced.
- New predicate integer64/1 introduced.
- New predicate integer_or_float32/1 introduced.
- New predicate integer_or_float/1 introduced.
- Foreign function interface now understands CharSequence data type.
- New predicate atom_or_reference/1 introduced.
- Auto loader now generates branching code for overloaded Java class members.
- Auto loader now re-exports the super class of the Java class.
- Auto loader now prefers more specific over less specific Java class members.
- Auto loader now combines Java specificity and Java inheritance.
- Unscaled value and scale constructor for class TermDecimal introduced.
- Unscaled value accessor for class TermDecimal introduced.
- Scale accessor for class TermDecimal introduced.

Android Interface

- Predictive text disabled since it conflicts with boundary and colour filtering.

2.3 Release 1.0.8

The following features and bug fixes were provided for the Jekejeke Prolog runtime library of version 1.0.8:

Language

- The predicate sys_advised/2 removed.
- The predicate sys_unadvise/2 removed.
- The predicate sys_advisea/1 removed.
- The predicate sys_advisez/1 removed.
- The predicate sys_advised_ref/3 removed.
- The predicate sys_evict_ref/1 removed.
- The predicate clause_ref/3 moved into module experiment/ref.
- The predicate atom_property/2 now understands slash and colon notation.
- The predicate atom_property/2 now returns orphan for a context less atom.
- The predicate sys_capture/1 can now be placed anywhere in query or clause.
- New predicate property sys_style_check/1 introduced.
- New predicate property sys_public/1 and sys_private/1 introduced.
- New style check verifies that public/1 declaration is repeated multiple files.
- New predicate property sys_meta_predicate/1 and sys_meta_function/1 introduced.
- New style check verifies that meta_predicate/1 declaration is repeated multiple files.
- New style check verifies that meta_function/1 declaration is repeated multiple files.
- New predicate property sys_dynamic/1 and sys_thread_local/1 introduced.
- New style check verifies that dynamic/1 declaration is repeated multiple files.
- New style check verifies that thread_local/1 declaration is repeated multiple files.
- New error verifies that public/1 doesn't promote a non-public predicate.
- New error verifies that meta_predicate/1 doesn't promote a non-meta predicate.
- New error verifies that meta_function/1 doesn't promote a non-meta function.
- New error verifies that multifile/1 doesn't promote a non-multi-file predicate.
- New error verifies that abolish/1 can remove all clauses.
- New error verifies that assertz/1 and assertza/1 don't redefine predicate.
- The module/1 directive now accepts the module name 'user' for nameless modules.
- The directive sys_source_package/0 removed.
- The directive sys_source_name/1 removed.

Frequent Predicates

- The predicate phrase/3 doesn't use the flag sys_barrier anymore.
- New multi-file predicate phrase_abnormal/1 introduced.
- The DCG expansion doesn't use the flag sys_barrier anymore.
- New multi-file predicate phrase_expansion_abnormal/1 introduced.
- Module experiment/tecto now uses new DCG programming interface.
- New module experiment/ref created from minlog extension system/ref.
- The predicate clause_ref/3 now also works for static predicates.
- The predicate consultable_ref/3 now also works for dynamic/thread_local predicates.
- New predicate above/2 introduced in module advanced/arith.

2.4 Release 1.0.7

The following features and bug fixes were provided for the Jekejeke Prolog runtime library of version 1.0.7:

Language

- The query-answer loop is now a little REPL.

Frequent Predicates

- New predicates error_make/[3,4] introduced into module system/locale.
- The predicates print_exception/[1,2] removed from module stream/console.
- New predicates print_error/[1,2] introduced into module stream/console.
- New predicates print_stack_trace/[1,2] introduced into module stream/console.

2.5 Release 1.0.6

The following features and bug fixes were provided for the Jekejeke Prolog runtime library of version 1.0.6:

Language

- Qualified predicates now searched in the re-exported sources of the module.
- Qualified operators now searched in the re-exported sources of the module.
- New predicate atom_list_concat/3 introduced.
- New predicate absolute_resource_name/2 introduced.
- New multi-file predicate sys_resource_suffix/1 introduced.
- New Prolog flag sys_locale introduced.
- Error messages removed from documentation.
- Error messages are now multi-lingual, English and German.
- Error messages are now found on open source web site.
- The documentation has now a title page.

Programming Interface

- New method getErrorProperties() in class Knowledgebase introduced.
- New method getDescriptionProperties() in class Capability introduced.
- Property PROP_FAMILY_DESCR from class Capability removed.
- Property PROP_PRODUCT_DESCR from class Capability removed.
- Property PROP_LANGUAGE_DESCR from class Capability removed.
- Property PROP_PLATFORM_DESCR from class Capability removed.
- Property PROP_LICENSE_DESCR from class Capability removed.
- New property PROP_LANGUAGE_CODE in class Capability introduced.
- New property PROP_INSTALL_CODE in class Capability introduced.
- New property PROP_LICENSE_CODE in class Capability introduced.
- Property PROP_HELP_DOCS from class Capability removed.
- Property PROP_SHOP_URL from class Capability removed.
- Foreign function parameter type Float introduced.
- Foreign function result type Float introduced.
- New method getDoubleValue() in class TermFloat introduced.
- New method getFloatValue() in class TermFloat introduced.
- Method getValue() in class TermFloat now returns number value.
- New methods toString() with Writer argument in class Term introduced.
- New flag FLAG_IGNORE_MOD in class Term introduced.
- Method getBase() and setBase() from class Knowledgebase removed.
- Property PROP_BASE_URL in class ToolkitLibrary introduced.
- Property PROP_SYS_LOCALE in class ToolkitLibrary introduced.

Frequent Predicates

- The predicate accumulate/3 removed in module advanced/aggregate.
- New predicate aggregate_all/3 introduced in module advanced/aggregate.
- New predicates aggregate_all/4, aggregate/4 and sys_collect/4 introduced.
- New predicate make_spec/4 introduced in module system/uri.
- Fixed missing I/O exception handling in sys_find_read/2.
- Fixed missing scenario when content type is null in stream reading.
- New module system/locale introduced.
- New module stream/console introduced.
- Error messages removed from documentation.

- Error messages are now multi-lingual, English and German.
- Error messages are now found on open source web site.
- The documentation has now a title page.

Android Interface

- Android interface is now multi-lingual, English and German.
- New language settings panel introduced.

Swing Interface

- Fixed missing text scaling in problem report alert dialog.
- Swing interface is now multi-lingual, English and German.
- New language settings panel introduced.
- Window icon scaling on high-res displays.

2.6 Release 1.0.5

The following features were provided for the Jekejeke Prolog runtime library of version 1.0.5:

Language

- The multi-file predicate sys_print_eq/1 removed.
- New multi-file predicate sys_unwrap_eq/2 introduced.
- Top-level displays those constraints related to the query variables.
- Printing of lists respect meta-directives now.
- New meta_function/1 directive introduced.
- Printing respects meta_function/1 directive now.
- The predicate sys_copy_term_site/2 removed.
- The predicate sys_findall_site/3 removed.
- The predicate sys_modfunc_site/2 removed.
- New predicate sys_functor/3 introduced.
- The predicate sys_moduniv_site/3 removed.
- New predicate sys_univ/2 introduced.
- New predicate atom_property/2 introduced.
- New predicate sys_callable/1 introduced.
- New predicate sys_var/1 introduced.
- New predicate sys_make_indicator/3 introduced.

Frequent Predicates

- New predicate plus/3 introduced into module advanced/arith.
- The predicate nth/3 removed from module basic/lists.
- New predicates nth0/3 and nth1/3 introduced into module basic/lists.
- New predicates nth0/4 and nth1/4 introduced into module basic/lists.
- New predicate (*->)//2 introduced into module standard/dcg.
- The predicate sys_bagof_site/3 removed.
- New predicate last/3 introduced into module basic/lists.

Swing Interface

- Heading scaling on high-res displays.
- Icon scaling on high-res displays.
- Table scaling on high-res displays.

2.7 Release 1.0.4

The following features were provided for the Jekejeke Prolog runtime library of version 1.0.4:

Language

- The Prolog flag sys_clause_conpand removed.
- New Prolog flag sys_body_convert introduced.
- New Prolog flag sys_clause_expand introduced.
- The predicate property sys_meta_operator/1 removed.
- The predicate property sys_noframe/0 removed.
- New predicate property sys_traverse/0 introduced.
- Operator property newr/0 removed.
- Pretty printing for newr/0 inferred from meta-predicate declaration.
- Operator property lowr/0 removed.
- Pretty printing for lowr /0 inferred from meta-predicate and operator level declaration.
- Operator property indr/0 removed.
- Pretty printing for indr /0 inferred from meta-predicate and operator level declaration.
- Pretty printing does not anymore place newline in situations such as p :- !.
- Pretty printing does not anymore place newline in situations such as p :- !, q.
- Pretty printing respects end-user newlines in compounds and lists.
- Pretty printing can preserve chars or codes notation.
- New read term option annotation/1 introduced.
- New write term option annotation/1 introduced.
- New read term option source/1 introduced.
- New write term option source/1 introduced.
- New read term option line_no/1 introduced.
- New write term option line_no/1 introduced.
- The foreign/3 directive now allows defining foreign predicates local to a module.
- Foreign predicates are now allowed to return some well-known Java exceptions.
- New Prolog text load option buffer/1 introduced.
- New source property buffer/1 introduced.
- Predicate sys_define_neutral/1 removed.
- New predicates sys_neutral_predicate/1 and sys_neutral_evaluable/1 introduced.
- New Prolog flag dialect introduced.
- New Capability Property company_descr introduced.
- New Prolog flag version_data introduced.
- Source property private/0 removed.
- Predicate property private/0 removed.
- Evaluable function property private/0 removed.
- Syntax operator property private/0 removed.
- New source property visible/1 introduced.
- New predicate property visible/1 introduced.
- New evaluable function property visible/1 introduced.
- New syntax operator property visible/1 introduced.
- Package local visibility for members now supported.
- New predicate sys_slash_atom/2 introduced.
- Structured module names now supported.
- New predicate sys_get_variable_names/1 introduced.
- New multi-file predicate sys_current_eq/1 introduced.
- New multi-file predicate sys_print_eq/1 introduced.
- Answer set now shows instantiation terms with variable names.
- Answer set now doesn’t show trivial instantiation terms anymore.

Programming Interface

- The InterpreterException constructor removed that fetches back trace.
- The InterpreterException constructor removed that fetches back trace and location.
- New InterpreterException constructor introduced with arbitrary context.
- New InterpreterException constructor introduced with arbitrary context and type.
- New method fetchStack() in class InterpreterException introduced.
- New method fetchLocation() in class InterpreterException introduced.
- New method fetchPos() in class InterpreterException introduced.
- Method absoluteWriteName() in class Knowledgebase removed.
- Method absoluteReadName() in class Interpreter removed.
- New predefined atoms ATOM_ON and ATOM_OFF introduced.
- New predefined atom ATOM_SUB introduced.
- Constant FLAG_VISIBLE_PRIVATE in class Interpreter removed.
- New method parseNumber() in class Interpreter introduced.
- New method prepareStream() in class Capability introduced.
- New method findLibrary() in class Knowledgebase introduced.
- Method defineForeign() in class Interpreter removed.
- New method getProperties() in class Capability introduced.

Frequent Predicates

- New module library(simp) introduced.
- New module library(expand) introduced.
- New module library(file) and class ForeignFile introduced.
- New module library(pairlist) introduced.
- New module library(uri) and class ForeignUri introduced.
- New module library(xml) and class ForeignXml introduced.
- New pre-loaded library(char) and class ForeignChar introduced.
- New pre-loaded library(byte) and class ForeignByte introduced.
- New pre-loaded library(term) and class ForeignTerm introduced.
- New pre-loaded library(stream) and class ForeignStream introduced.
- New stream open option buffer/1 introduced.
- New stream property buffer/1 introduced.
- New module library(random) introduced.
- New pre-loaded library(toolkit) and class ForeignToolkit introduced.
- New module library(shell) and class ForeignShell introduced.

Swing User Interface

- More performant input colouring.

Android User Interface

- More performant input colouring.
- Failed editing of protected text area now preserves character attributes.

2.8 Release 1.0.3

The following features were provided for the Jekejeke Prolog runtime library of version 1.0.3:

Language

- New predicates current_oper/1 and oper_property/2 introduced.
- New predicates set_oper_property/2 and reset_oper_property/2 introduced.
- New sys_load_file/2 option export/1 introduced.
- The source property sys_deps/1 removed.
- New source property sys_deps/2 introduced.
- New predicate reexport/1 introduced.
- The predicates sys_copy_term_site/[4,6] removed.
- New predicate sys_findall_site/3 introduced.
- Directive static/0 now performs multifile check.
- The predicate property discontiguous/0 and multifile/0 removed.
- New predicate property discontiguous/1 and multifile/1 introduced.
- The discontiguous check is now performed local to a Prolog text.
- The multifile check is now performed local to a Prolog text.
- The predicate property full_name/1 now returns a predicate indicator.
- The source property preload/0, private/0 and name/1 removed.
- New source property sys_source_preload/0 introduced.
- New source property sys_source_private/0 introduced.
- New source property sys_source_name/1 introduced.
- The predicate property sys_noframe/0 does not anymore imply cut transparency.
- New predicate property sys_nobarrier/0 for defined predicates introduced.
- The predicate listing/1 now groups clauses by their source context.
- The predicate listing/0 now groups predicates and clauses by their source context.
- The predicates listing/[0,1] now show some source properties.
- New multifile predicate sys_file_suffix/1 introduced.
- The predicates absolute_file_name/[2,3] now do file extension probing.
- ISO apply and bag predicates not anymore preloaded.
- ISO DCG expansion not anymore preloaded.
- Interpreter now keeps a mapping of module names to source files.
- Better interface/implementation cache invalidation for reexport/1.
- Better interface/implementation cache invalidation for public/1 and private/1.
- Experimental aggregate predicates not anymore preloaded.
- Experimental abstract predicates not anymore preloaded.
- Statistics property variables replaced by new Prolog flag sys_variables.
- Statistics property choices replaced by new Prolog flag sys_choices.
- New user session predicate time/1 introduced.
- Interpreter now keeps a list of unnamed source files.
- Better implementation cache invalidation for non-module predicates.
- Refined syntax operator and predicate lookup for non-module predicates.
- New override/1 predicate property introduced.
- New override warning for module and non-module predicates.
- Predicate sys_term_goal/2 renamed to predicate sys_goal_kernel/2.
- Predicate sys_term_witness/2 renamed to predicate sys_goal_globals/2.
- Syntax operator (&)/2 removed.
- Multiple directives and clauses can now be joined by the syntax operator (/\)/2.
- The advising operator (^)/1 for clauses has been removed.
- The local variables operator (^)/2 for queries has been removed.
- New syntax operator (*->)/2 introduced.
- New logical predicate (*->)/2 introduced.

- New logical predicate (*->)/2 in connection with (;)/2 introduced.
- Syntax operators ‘user%<name>’ are now directly resolved to ‘<name>’.
- Predicates ‘user%<name>’ are now directly resolved to ‘<name>’.
- Predicate current_op/3 now understands syntax operator patterns.

Programming Interface

- Method absoluteReadName() now probes library/1 via Class.getResource().
- Method absoluteReadName() now probes path/1 via ClassLoader.getResource().
- Method absoluteReadName() now probes file schema via File.exists().
- Method absoluteReadName() now probes http schema via HEAD method.
- Method absoluteReadName() now probes other schemas by open/close stream.
- Method absoluteReadName() now returns null if probe fails.

Frequent Predicates

- Flag example moved to here from language reference.
- Palindrome example moved to here from language reference.
- Fruits example moved to here from language reference.
- Grammar rule format description moved to here from language reference.
- Syntax operators for dcg and apply moved to here from language reference.
- New module library(apply) introduced.
- New module library(bags) introduced.
- New module library(lists) introduced.
- New module library(abstract) introduced.
- New module library(aggregate) introduced.
- New module library(arith) introduced.
- New module library(ordsets) introduced.
- New module library(sets) introduced.
- New module library(dcg) introduced.
- New module library(tecto) introduced.

2.9 Release 1.0.1

The following features were provided for the Jekejeke Prolog runtime library of version 1.0.1:

Language

- The atom polymorphic cache extended so that it respects the call-site.
- The atom polymorphic cache refined so that it respects source dependencies.
- New source property name/1 introduced.
- New syntax operator sys_source name/1 introduced.
- New directive sys_source_name/1 introduced.
- New directive module/2 introduced.
- New directive use_module/1 introduced.
- New predicate property full_name/1 introduced.
- New syntax operator (:)/2 introduced.
- The predicate indicator extended so that a module name can be qualified.
- The predicate sys_functor_site/4 removed.
- New predicate sys_modfunc_site/2 introduced.
- The atom polymorphic cache varied so that it can extend names.
- New predicate sys_callable_colon/2 introduced.
- New predicate (:)/2 introduced.
- The predicates asserta/1 and assertz/1 do now understand the colon (:)/2.
- The predicates clause/2 and retract/1 do now understand the colon (:)/2.
- New predicate sys_indicator_colon/2 introduced.
- The predicate listing/1 now understands the colon (:)/2.
- The directive private/0 renamed to sys_source_private/0.
- New syntax operator private/1 introduced.
- New directive private/1 introduced.
- New predicate op/4 introduced.
- The directives public/1 and private/1 now accept op/3.
- The second argument of module/2 now accepts op/3.
- The atom polymorphic cache now reflects predicate visibility.
- Predicate directives now accept colon notation.
- Listing now shows colon notation for predicate directives.
- Error messages now show colon notation.
- Stack trace now shows colon notation.
- The predicates absolute_file_name/[2,3] don't search in classes by default anymore.
- The predicates absolute_file_name/[2,3] can now resolve against the current source.
- Support of path/1 in predicates absolute_file_name/[2,3] introduced.
- Support of library/1 in predicates absolute_file_name/[2,3] introduced.
- Support of absolute and relative jar schema paths introduced.
- New predicate sys_source_preload/0 introduced.
- New source property preload/0 introduced.
- The predicate sys_elevate/1 has been removed.
- New predicates current_evaluable/1 and evaluable_property/2 introduced.
- New predicate set_evaluable_property/2 introduced.
- New predicate reset_evaluable_property/2 introduced.
- Clause expansion now recognizes the colon notation (:)/2.
- Clause rebuild now recognizes the colon notation (:)/2.
- The predicate sys_univ_site/3 removed.
- New predicate sys_moduniv_site/2 introduced.
- The predicates sys_extend_args/[3..9] removed.
- New predicates sys_modext_args/[3..9] introduced.

- New predicate sys_replace_site/3 introduced.
- The DCG translation now recognizes the colon notation (:)/2.

Programming Interface

- Method getBaseUri() in Knowledgebase renamed to getBase().
- Method setBaseUri() in Knowledgebase renamed to setBase().
- Method getBase() in Knowledgebase now returns a String again.
- Method setBase() in Knowledgebase now accepts a String again.
- Method absoluteWriteName() in Knowledgebase now returns a String again.
- Method forName() in Knowledgebase renamed to stringToCapability().
- Method capabilityToString() in Knowledgebase introduced.
- Method absoluteReadName() moved from class Knowledgebase to class Interpreter.
- Method absoluteReadName() in Interpreter now returns a Term.
- Method absoluteReadName() in Interpreter now accepts a Term.

Swing User Interface

- New add path menu item introduced.
- New add path menu removed.

2.10 Release 1.0.0

The following features were provided for the Jekejeke Prolog runtime library of version 1.0.0:

Language

- New source property sys_timing/1 introduced.
- New source property sys_deps/1 introduced.
- Option predlude/1 from predicate sys_load_file/2 removed.
- Option scope/1 from predicate sys_load_file/2 removed.
- New option timeout/1 for predicate sys_load_file/2 introduced.
- New option verbose/1 for predicate sys_load_file/2 introduced.
- New predicate sys_detach_file/2 introduced.
- New predicate sys_import_file/2 introduced.
- New path specification +P for []/2 introduced.
- New predicates make/0 and rebuild/0 introduced.
- New option bom/1 for predicate sys_load_file/2 introduced.
- New option bom/1 for predicates open/[3,4] introduced.
- New property bom/1 in predicate stream_property/2 introduced.
- The predicates absolute_file_name/[2,3] now remove the authority for schema file.
- The predicates absolute_file_name/[2,3] now remove the fragment.
- The predicates absolute_file_name/[2,3] now encode paths.
- The predicates sys_load_file/2 and sys_import_file/2 handle coded paths.
- The predicates open/[3,4] now handle coded paths.
- Predicate property sys_private/0 removed.
- New predicate property sys_owner/1 introduced.
- New option single_quotes/1 for predicates write_term/[2,3] introduced.
- New option single_quotes/1 for predicates read_term/[2,3] introduced.
- New Prolog property single_quotes/1 introduced.

Programming Interface

- New representation for a sub range of BigDecimals introduced.

Android User Interface

- The command history is now saved upon closing a tab.
- New load menu item.
- New make menu item.

Swing User Interface

- The command history is now saved upon closing a tab.
- New file selector button in paths panel.
- New consult and load menu item.
- New unload menu item.
- New make and reload menu item.

3 Release 0.9

This section lists the changes concerning the Jekejeke Prolog runtime library executable, documentation and samples:

3.1 Release 0.9.12

The following features were provided for the Jekejeke Prolog runtime library of version 0.9.12:

Language

- New directive foreign/4 introduced.
- New command line option -a introduced.

Programming Interface

- New method classToString() in class Knowledgebase introduced.
- New method getCommittedLoader() in class Knowledgebase introduced.
- New method forName() in class Knowledgebase introduced.
- New constant FLAG_VISIBLE_PRIVATE in class Interpreter introduced.
- New method definedForeign() in class Interpreter introduced.
- The method getBaseURL() in class Knowledgebase now returns an URI.
- The method setBaseURL() in class Knowledgebase now accepts an URI.
- The method absoluteWriteName() in class Knowledgebase now returns an URI.
- The method getClassPaths() in class Knowledgebase now returns URIs.
- Method addClassPath() moved to class Interpreter.

Android User Interface

- Capability icons introduced.
- Paths and capability newly detected panel introduced.

Swing User Interface

- Paths and capability newly detected panel introduced.

3.2 Release 0.9.11

The following features were provided for the Jekejeke Prolog runtime library of version 0.9.11:

Language

- The reference data type now inherits the equals() and hashCode() of the object.
- The reference data type is now orderable by the compare() of a comparator.

Programming Interface

- New constructor TermRef(Object, Comparator) introduced.
- New method getComparator() in class TermRef introduced.

Android User Interface

- Click outside of dialogs to dismiss disabled.

3.3 Release 0.9.10

The following features were provided for the Jekejeke Prolog runtime library of version 0.9.10:

Language

- Foreign function argument type Character not anymore supported.
- Foreign function return type Character not anymore supported.
- Predicate atom_chars/2 and atom_codes/2 now handles surrogate pairs.
- Predicate number_chars/2 and number_codes/2 now handles surrogate pairs.
- Predicate char_code/2 now handles surrogate pairs.
- Predicate atom_length/2 now handles surrogate pairs.
- Predicate sub_atom/5 now handles surrogate pairs.
- New predicate sub_atom/4 introduced.
- Predicates put_char/1 and put_code/1 now handle surrogate pairs.
- Predicates get_char/1 and get_code/1 now handle surrogate pairs.
- Predicates peek_char/1 and peek_code/1 now handle surrogate pairs.
- Predicate read_term/2 now handles surrogate pairs.
- Predicate write_term/2 now handles surrogate pairs.
- New double_quotes and back_quotes flag value error introduced.
- The value error is now the default for the back_quotes flag.
- The double_quotes and back_quotes flags are now settable.
- New double_quotes/1 and back_quotes/1 option for read_term/[2,3] introduced.
- New double_quotes/1 and back_quotes/1 option for write_term/[2,3] introduced.
- New predicate property private/0 introduced.
- New source property private/0 introduced.
- The predicates asserta/1 and assertz/1 now respect public/private.
- The predicates clause/2 and restract/1 now respect public/private.
- The predicates listing/0 and listing/1 now respect public/private.
- The predicate current_predicate/1 now respects public/private.
- The predicate predicate_property/2 now respects public/private.
- The predicate set_predicate_property/2 now respects public/private.
- The predicate reset_predicate_property/2 now respects public/private.

Programming Interface

- New constructor TermAtom(int) introduced.
- New method getCodePointValue() in class TermAtom introduced.

3.4 Release 0.9.9

The following features were provided for the Jekejeke Prolog runtime library of version 0.9.9:

Language

- New capability property sys_notrace introduced.
- New abstraction predicates (\)/n introduced.
- The predicates (^)/n now solely define local variables.
- New abstraction predicates (\)/n (grammar) introduced.
- The predicates (^)/n (grammar) now solely define local variables.
- New predicates sys_copy_term_site/[2,4,6] introduced.

Programming Interface

- New capability property PROP_SYS_NOTRACE introduced.

3.5 Release 0.9.8

The following features were provided for the Jekejeke Prolog runtime library of version 0.9.8:

Language

- New evaluable function random/0 introduced.
- New evaluable function random/1 introduced.
- Indexing of large data sets improved.
- The predicates asserta/1 and assertz/1 now use Java body conversion.
- The predicates sys_advisea/1 and sys_advisea/1 now use Java body conversion.
- The predicates sys_elevate/1 and sys_capture/1 now use Java body conversion.
- The predicates call/1, once/1 and \+/1 now use Java body conversion.
- The predicates catch/3 and sys_trap/3 now use Java body conversion.
- The predicate sys_atomic/1 now uses Java body conversion.
- The predicates sys_cleanup/1 and sys_unbind/1 now use Java body conversion.
- The predicates findall/3, bagof/3 and sys_heapof/3 now use Java body conversion.
- The Prolog flag sys_clause_expand has been renamed to sys_clause_conpand.
- The predicate property sys_stable_predicate/1 has been removed.
- The predicate property sys_instable_predicate/1 has been removed.
- The predicate property sys_cut_stable_predicate/1 has been removed.
- New predicate properties sys_notraverse/0 and sys_nowork/0 introduced.
- The predicates asserta/1 and assertz/1 don't understand (&)/2 and unit/0 anymore.
- The predicate sys_advisea/1 doesn't understand (&)/2 and unit/0 anymore.
- The predicate sys_advisea/1 doesn't understand (&)/2 and unit/0 anymore.
- The predicate retract/1 doesn't understand (&)/2 and unit/0 anymore.
- The predicate sys_advised/1 doesn't understand (&)/2 and unit/0 anymore.

Programming Interface

- The Interpreter method unfoldFirst() now uses Java body conversion.
- The Interpreter method unfoldChecked() now uses Java body conversion.

3.6 Release 0.9.7

The following features were provided for the Jekejeke Prolog runtime library of version 0.9.7:

Language

- New capability predicate sys_check_license/1 introduced.
- New capability predicate sys_check_licenses/0 introduced.
- New control predicate false/0 for TC2 compatibility introduced.
- New evaluable function +/1 for TC2 compatibility introduced.
- New capability property shop_url/1 introduced.
- New access mode append in open/[3,4] introduced.
- New access mode append in access/1 option of absolute_file_name/3 introduced.
- New access mode append in mode/1 property of stream_property/2 introduced.
- New length/1 property in stream_property/2 introduced.
- New stream predicate set_stream_length/2 introduced.
- New call-site transfer predicates sys_univ_site/3 and sys_copy_site/3 introduced.
- New call-site transfer predicate sys_extend_args/3 introduced.
- New dynamic predicate clause_ref/3 and sys_advised_ref/3 introduced.
- New dynamic predicate sys_evict_ref/1 introduced.
- New call-site transfer predicate sys_functor_site/3 introduced.
- Call-site transfer for the body conversion and the stability analysis introduced.
- Call-site transfer for the higher order predicates call/n and ^/n introduced.
- Call-site transfer for the DCG predicate phrase/3 introduced.

Programming Interface

- New Toolkit methods checkLicense() and checkLicenses() introduced.
- New Capability constant PROP_SHOP_URL introduced.

Android User Interface

- New text panel introduced.
- New information panel and email panel introduced.

Swing User Interface

- New backward text search button.
- New information panel introduced.

3.7 Release 0.9.6

The following features were provided for the Jekejeke Prolog runtime library of version 0.9.6:

Language

- New capability property bundle_dir introduced.
- New capability property language_descr introduced.
- New capability property family_descr introduced.
- Capability property predefined removed.
- New capability property image_icon introduced.
- New capability property big_image_icon introduced.
- New capability property help_docs introduced.
- New Prolog flag sys_act_status introduced.
- Compound parenthesis must now directly precede atom (“(“). [ISO 6.3.3]
- Operator closing symbol (“)],|}”) must now not directly follow atom. [ISO 6.3.1.3]
- New set predicate sys_distinct/2 introduced.
- New set predicate sys_keygroup/2 introduced.
- New set predicate sys_heapof/3 introduced.
- Priority clash now also detected for left arguments that are prefix. [ISO 6.3.4.2]
- Arbitrary tokens are not anymore allowed as operators. [ISO 6.3.4.3]
- New database predicates sys_advisea/1 and sys_advisez/1 introduced.
- New database predicates sys_advised/2 and sys_unadvise/1 introduced.
- Source specific operator definitions now undone initially in consult/1.
- User operator definitions now shown during listing/0.
- New aggregate predicate accumulate/3 introduced.
- New aggregate predicate aggregate/3 and sys_collect/3 introduced.
- New Prolog flag sys_cur_input and sys_cur_output introduced.
- New Prolog flag sys_disp_error and sys_cur_error introduced.
- New stream predicate current_error/1 and set_error/1 introduced.
- New predicate property sys_cut_stability_predicate/1 introduced.
- Prolog flag sys_owner replaced by new Prolog flag sys_capability.
- Source property system replaced by new source property sys_capability.
- Meta-argument specifiers ‘:’ and ‘@’ replaced by 0 and -1.
- Closure-integer meta-argument specifiers introduced.
- Pretty printing now respects closure arguments of meta-predicates.
- Meta predicate declarations for higher order predicates introduced.
- Meta predicate declarations for grammar rules introduced.
- The DCG translation now generates list equations for terminals instead of ‘C’/3.
- The DCG translation can now merge equations into the head or body goals.

Programming Interface

- New capability method getProperty() introduced.
- Interpreter method initCapability() moved to capability.
- Knowledgebase method finiCapability() moved to capability.
- New Toolkit method getInitCapabilities() introduced.
- New Toolkit method getBrandCapability() introduced.
- New Toolkit method checkLicense() introduced.
- New Interpreter method getProperty() introduced.
- New Interpreter method setProperty() introduced.
- TermDecimal method getValue() now returns Number.
- New TermDecimal method getBigDecimal() introduced.

- The Interpreter methods setXXX() and getXXX() for streams and GUI removed.
- The ToolkitLibrary has now declaration of the bootstrap Interpreter properties.
- The Interpreter methods setStatus() and getStatus() removed.
- Atoms ATOM_NIL and ATOM_CONS moved to Knowledgebase class.
- Type parameters visible in final .jar/.zip and in documentation.

Android User Interface

- Console window now shows input, output and error streams in different colours.
- The window and stream colours can now be configured in the settings dialog.

Swing User Interface

- New backward string search menu item and toolbar action introduced.
- Console window now shows error streams in a different text colour and text style.
- The error stream colour and style can now be configured in the settings dialog.

3.8 Release 0.9.5

The following features were provided for the Jekejeke Prolog runtime library of version 0.9.5:

Language

- Issue fixed with concurrent path adding.
- New Prolog flag sys_attached_to introduced.

Programming Interface

- New Interpreter methods setAttachedTo() and getAttachedTo() introduced.

Android User Interface

- The runtime library is now available as an activity with an Android console.
- The settings dialog has now a class path panel.
- The settings dialog has now an enlist panel.
- New menu item abort introduced.

Swing User Interface

- The main method of the runtime library now starts by default a Swing console.
- The main method of the runtime library has now the –h option.
- The about and register dialog show capability specific icons.

3.9 Release 0.9.4

The following features were provided for the Jekejeke Prolog runtime library of version 0.9.4:

Language

- New syntax operator (&)/2 introduced.
- New predicate property static introduced.
- New directive static introduced.
- Just in time indexing hash tables now resize.
- Assert/retract scales better for large clause sets.
- Just in time indexing creates flatter index structures now.
- Just in time indexing now returns more negative information.
- New system predicate advice/1 introduced.
- New system predicate simplify_term/2 introduced.
- New multi-file system predicate term_simplification/2 introduced.
- New evaluable functions min/2 and max/2 introduced.
- New system flag sys_last_pred introduced.
- New system predicate sys_capture/1 introduced.
- No more choice point creation for neck cut.
- If-then-else is already detected at compile time.
- More time and memory efficient call/n implementation.
- The call/n implementation now carries over the call-site.

Programming Interface

- CapabilityLibrary moved to package jekpro.platform.headless.
- ToolkitLibrary moved to package jekpro.platform.headless.

3.10 Release 0.9.3

The following features were provided for the Jekejeke Prolog runtime library of version 0.9.3:

Language

- New system predicate sys_fini_capability/1 introduced.
- New system predicate sys_capability_property/2 introduced.
- New capability initialization option prompt/1 introduced.
- New write term option context/1 introduced.
- New write term option format/1 introduced.
- New write term option operand/1 introduced.
- New stream property file_name/1 introduced.
- New stream property input/0 introduced.
- New stream property output/0 introduced.
- Opening a directory now yields file not found (Finding U. Neumerkel).
- A period (.) is now shown in quotes during write (Finding U. Neumerkel).
- New statistics value runtime introduced.
- New statistics value choices introduced.
- New system predicate sys_unbind/1 introduced.
- New system predicate simplify_goal/2 introduced.
- New multi-file system predicate goal_simplification/2 introduced.

Programming Interface

- New class ToolkitLibrary introduced.
- New class CapabilityRuntime introduced.
- Method initThreshold() in class Toolkit removed.
- Method addThresholdListener() in class Toolkit removed.
- Method removeThresholdListener() in class Toolkit removed.
- Method getClassLoader() in class Knowledgebase removed.
- New method stringToClass() in class Knowledgebase introduced.
- New method finiCapability() in class Knowledgebase introduced.
- New method finiKnowledgebase() in class Knowledgebase introduced.
- New constant FLAG_IGNORE_OPS in class Term.
- Method compareTo() in class Term removed.
- New method compare() in class Interpreter.
- Method createVars() moved to class TermVar from class Interpreter.
- New method getValue() in class TermVar.
- New method setSignal() in class Interpreter introduced.
- New method setSignalAndWait() in class Interpreter introduced.

3.11 Release 0.9.2

The following features were provided for the Jekejeke Prolog runtime library of version 0.9.2:

Language

- New system predicate set_stream_position/2 introduced.
- New open option reposition/1 introduced.
- New stream property reposition/1 introduced.
- New stream property position/1 introduced.
- The evaluable predicate sign/1 now returns float for a float argument.
- The evaluable predicate sign/1 now returns decimal for a decimal argument.
- New directive meta_predicate/1 introduced.
- New predicate property meta_predicate/1 introduced.
- New predicate property sys_noexpand/0 introduced.
- New predicate property sys_stable_predicate/1 introduced.
- New predicate property sys_instable_predicate/1 introduced.
- New write option variable_names/1 introduced.
- New predicate property sys_meta_operator/1 introduced.
- New write option priority/1 introduced.
- New Prolog flag sys_break_level introduced.
- New Prolog flag sys_disp_input and sys_disp_output introduced.
- New system predicates call/2 .. call/8 introduced.
- New system predicates ^/3 … ^/9 introduced.
- New DCG expansion call/2 .. call/8 introduced.
- New DCG expansion ^/3 … ^9 introduced.
- Character constants are now parsed. (Finding U. Neumerkel)
- Comma and bar needs quotes. (Finding U. Neumerkel)
- Comma and bar operator cannot be modified. (Finding U. Neumerkel)
- Parenthesis in writing of same level operator. (Finding U. Neumerkel)
- Parsing does now respect minimal number borders. (Finding U. Neumerkel)
- Predefine string escapes not anymore case insensitive. (Finding U. Neumerkel)
- Bug in block comment after graphic character fixed. (Finding U. Neumerkel)
- Bug in negative number with non-decimal radix fixed. (Finding U. Neumerkel)
- Bug in exception for failed clean-up fixed. (Finding U. Neumerkel)
- Bug in operator priority for query answer bindings fixed. (Finding U. Neumerkel)
- New system predicate sys_term_singletons/2 introduced.
- New system predicate sys_number_variables/4 introduced.
- New condition value never in consult option introduced.
- New system predicate unload_file/2 introduced.

Programming Interface

- TermFloat constructor does not anymore accept NaNs or Infinite.
- New methods setDispInput() and getDispInput() in class Interpreter introduced.
- New methods setDispOutput() and getDispOutput() in class Interpreter introduced.
- New method initThreshold() in class Toolkit introduced.
- New methods addThresholdListener() in class Toolkit introduced.
- New methods removeThresholdListener() in class Toolkit introduced.

Compliance

- New call/n test cases introduced.

- New DCG test cases introduced.
- New needs quotes test cases introduced. (Finding U. Neumerkel)
- New cannot be modified test cases introduced. (Finding U. Neumerkel)
- New same level operator test cases introduced. (Finding U. Neumerkel)
- New minimal number borders test cases introduced. (Finding U. Neumerkel)
- New escape case sensitivity test cases introduced. (Finding U. Neumerkel)

3.12 Release 0.9.1

The following features were provided for the Jekejeke Prolog runtime library of version 0.9.1:

Language

- New system predicate sys_atomic_call/1 introduced.
- New system predicate sys_on_cleanup/1 introduced.
- New system predicate setup_call_cleanup/3 introduced.
- New Prolog flag sys_choices introduced.

Programming Interface

- New method swapBarrier() in CallFrame class introduced.
- New method getCleanup() in CallOut class introduced.
- New method setCutter() in CallOut class introduced.
- New method setSpecial() in CallOut class introduced.
- The method unfoldClose() in Interpreter now might throw an InterpreterException.
- The method unfoldCut() in Interpreter now takes an InterpreterException argument.
- The method unfoldCut() in Interpreter now returns an InterpreterException.
- New constructors InterpreterException() with an InterpreterMessage introduced.
- New constructor InterpreterException() with two InterpreterExceptions introduced.
- New method causeChainRest() in InterpreterException class introduced.
- New method getException() in CallOut class introduced.
- New method setException() in CallOut class introduced.

3.13 Release 0.9.0

The following features were provided for the Jekejeke Prolog runtime library of version 0.9.0:

Language

- New system predicate thread_sleep/1 introduced.
- New directive thread_local introduced.
- New stream open option type(binary) introduced.
- New system predicates put_byte/1 and put_byte/2 introduced.
- New system predicates peek_byte/1 and peek_byte/2 introduced.
- New system predicates get_byte/1 and get_byte/2 introduced.
- The system predicate with_input_from now supports binary streams.
- The system predicate with_output_to now supports binary streams.
- Compound predicate cache removed and atom predicate cache enhanced.
- Predicate cache also used as evaluable function cache.
- New evaluable functions pi/0 and e/0 introduced.
- Crash in is/2 for reference type in expression fixed.
- New multi-file predicate 'C'/3 with standard definition introduced.
- DCG terminals now make use and expanded to 'C'/3.
- Missing expansion for the DCG action {}/1 added.
- phrase/2 now calls phrase/3 with the output argument loose.
- Head variable elimination by temporary variable analysis enhanced.
- New system predicate sub_atom/5 introduced.
- The character '\xFFFD\' now detected as invalid Unicode during parsing.
- The character type SURROGATE now detected as invalid Unicode during parsing.
- The character type UNASSIGNED now detected as invalid Unicode during parsing.
- The character type PRIVATE_USE now detected as invalid Unicode during parsing.
- Back quoted strings now parsed as Prolog variables.
- New evaluable functions asin/1, acos/1 and tan/1 introduced.
- New evaluable function xor/2 introduced.
- New Prolog flag sys_clause_index introduced.
- Dynamic multi-argument indexing introduced.

Programming Interface

- New constants ATOM_NIL and ATOM_CONS in TermAtom class.
- Method getLongValue() in TermDecimal class renamed to getMoneyValue().
- New long based constructor in TermInteger class.
- New method getLongValue() in TermInteger class introduced.
- Attribute input/output renamed to curInput/curOutput in Interpreter class.
- Type of attribute curInput/curOutput in Interpreter class generalized to Object.
- Character based constructor removed from TermAtom class.
- Method getCharValue() removed from TermAtom class.
- New convenience methods checkXXX() in InterpreterMessage class introduced.

4 Release 0.8

This section lists the changes concerning the Jekejeke Prolog runtime library executable, documentation and samples:

4.1 Release 0.8.9

The following features were provided for the Jekejeke Prolog runtime library of version 0.8.9:

Language

- New system predicate sys_activate_capability/2 introduced.
- New system predicate sys_init_capability/1 introduced.
- New Prolog flag base_url introduced.
- New system predicate sys_calc_install_id/2 introduced.
- New system predicate sys_reg_license_text/2 introduced.
- New Prolog flag sys_context introduced.
- New predicate property sys_context/1 introduced.
- Output properties of open/4 moved to stream_property/2.
- New system predicate sys_init_finally/2.
- System predicates sys_clause_term/3 moved to development environment.
- System predicates sys_retract_term/2 moved to development environment.
- System predicates sys_assertz_term/2 moved to development environment.
- System predicates sys_asserta_term/2 moved to development environment.
- Prolog flag source_file moved to development environment.
- Prolog flag line_no moved to development environment.
- New system predicates set_predicate_property/2 introduced.
- New system predicates reset_predicate_property/2 introduced.
- Prolog flag sys_mask now modifiable,
- System predicate sys_mask_call/2 removed.
- Prolog flag sys_owner now modifiable.
- New predicate property system/0 introduced.
- Predicate property sys_invisible/0 renamed to sys_notrace/0.
- New source file property system/0 introduced.
- New source file property sys_notrace/0 introduced.
- New system predicate set_source_property/2 introduced.
- New system predicate reset_source_property/2 introduced.
- New user definable predicate term_expansion/2 introduced.
- Predicate property sys_nosource/0 removed.
- System predicate sys_elevate/1 now also changes the context.
- System access matrix refined by multifile predicate property.
- The multifile property now detects conflict with single file.
- New Prolog flag sys_clause_expand.

- New system predicate expand_term/2 and expand_goal/2 introduced.
- Definite clause grammars now make use of clause expansion.
- New system predicate sys_add_path/1 introduced.
- New system predicate sys_current_path/1 introduced.
- New system predicate sys_current_capability/1 introduced.
- Fixed an issue with cut transparency and end of body.
- Fixed an issue with cut transparency and catch.

Programming Interface

- Method addCapability() from the class Toolkit removed.
- New method initCapability() in the class Interpreter introduced.
- New method activateCapability() in the class toolkit introduced.
- The method toString() of the class InterpreterMessage now takes a Toolkit parameter.
- New method systemErrorType() the class InterpreterException introduced.
- New method calcInstallID() in the class Toolkit introduced.
- New method regLicenseText() in the class Toolkit introduced.
- The method getFlag() in class Interpreter has been renamed to getStatus().
- The method setFlag() in class Interpreter has been renamed to setStatus().
- New method addClassPath() in class Knowledgebase introduced.
- New method getClassPaths() in class Knowledgebase introduced.
- New method getCapabilities() in class Knowledgebase introduced.

4.2 Release 0.8.8

The following features were provided for the Jekejeke Prolog runtime library of version 0.8.8:

Language

- Text input streams now deliver end of line always as '\n'.
- All ASCII control characters now count as space.
- Terminating period now also detected when directly followed by a line comment.
- Block comments now not anymore detected when preceded by graphic character.
- New section that documents the Unicode extension, 16 bit and version 4.0.
- Automatic escape of character codes above or equal 512 as hex codes.
- The open option charset/1 renamed to encoding/1.
- Predicate stream_property/2 with properties encoding/1 and line_no/1 introduced.
- The open option encoding/1 changed from output to input for sources.
- Predicate sys_load_file/2 with options condition/1, prelude/1 and scope/1 introduced.
- Predicate source_property/2 now returns also encoding/1 and short_name/1.
- The empty set {} can now be parsed as a Prolog term.
- No more quoting is done when canonically writing the empty set {} or the empty list.
- Infix and postfix operators are now held in a single table.
- No more level error when defining the comma (,) or the vertical bar (|).
- The token syntax now supports also back quoted strings.
- The token syntax now also supports the escape of quotes.
- Permission type private_procedure now also supported.
- Stricter implementation of predicate_property/2 for bound first argument.
- Optimization technique flags prefixed with sys_.
- New Prolog flag sys_mask introduced.
- New system predicates sys_mask_call/2 introduced.
- New Prolog flag sys_owner introduced.
- New system predicate sys_elevate/1 introduced.
- New Prolog flag source_file and line_no introduced.
- New system predicate sys_nosource/1 introduced.
- System predicates sys_nobox/1 and sys_nosig/1 removed.
- Stream consult now suppresses display of system exceptions.
- Stream consult now reacts on user exit by leaving the consult loop.
- Stream consult now reacts on all other system exceptions by leaving all loops.
- New system predicate sys_catch_system/3 introduced.
- System predicate sys_finally/2 removed.
- New Prolog flag back_quotes introduced.
- New Prolog flag max_code introduced.
- Layout character in string now detected as syntax error.
- New system predicate sys_setup_cleanup/3 introduced.

Programming Interface

- Interpreter methods get/setPrintwriter renamed to get/setOutput.
- Interpreter methods get/setBuffered renamed to get/setInput.
- Interpreter methods get/setInput now based on LineNumberReader.
- New Term method toString() with variable map parameter.
- New Term method toString() with write option flags parameter.

4.3 Release 0.8.7

The following features were provided for the Jekejeke Prolog runtime library of version 0.8.7:

Language

- Character type can now also be declared for a Java foreign predicate.
- Text stream open option encoding/1 renamed to charset/1.
- Character set encoding now detected from content type.
- New source open options use_caches/1 and if_modified_since/1.
- New source open options last_modified/1 and expiration/1.
- Predicate source_file/1 renamed to current_source/1.
- New predicate source_property/2 introduced.
- New predicate ensure_loaded/1 introduced.
- New clause option line_no/1.
- Internal footprint of dynamically created variables reduced.
- Internal footprint of dynamically created atoms reduced.
- Line number now also shown for errors not based on callable.
- Problem fixed with stack frame elimination and cut transparency.
- New predicate atom_concat/3 introduced.
- New predicate include/1 introduced.
- Problem fixed in recursively calling of the top level.
- New predicate prolog/0 introduced.
- New statistics keys variables/0 and choices/0 introduced.
- The predicates functor/3 and (=..)/2 are now recognized as deterministic.
- New Prolog flag trace/0 introduced.
- Problem fixed with stack frame elimination and multiple cuts.
- The flags trace/0 and debug/0 are now changeable.
- New predicate property sys_nosig/0 introduced.
- New predicate sys_finally/2 introduced.

Programming Interface

- Class TermInteger has now method getIntValue().
- Class TermDecimal has now method getLongValue().
- Class TermAtom has now character based constructor.
- Class TermAtom has now method getCharValue().
- New class Toolkit.
- New constructor for the class Knowledgebase.
- New constructor for the class Interpreter.
- New method absoluteWriteName() in class Knowledgebase.
- New method absoluteReadName() in class Knowledgebase.
- New methods getFlag() and setFlag() in class Interpreter.
- New methods getSignal() and setSignal() in class Interpreter.
- New methods xxxError() in class InterpreterMessage.
- Custom Mutex object example completed.

4.4 Release 0.8.6

The following features were provided for the Jekejeke Prolog runtime library of version 0.8.6:

Language

- New syntax for references, integer with reference indicator (“R” or “r”) at end.
- New syntax for decimals, float with decimal indicator (“D” or “d”) at end.
- Syntax now clarifies variable, name, integer and float.
- New standard operator (**)/2.
- New test predicate decimal/1 and reference/1.
- Test predicates atomic/1 and number/1 adapted.
- New evaluable predicate decimal/1.
- New internal representation Long (for bounded decimal)
- New internal representation Double (for floats).
- Basic operations extended to floats, decimals we had already.
- Rounding operations extended to floats, decimals we had already.
- New evaluable functions for the trigonometric operations.
- Additional errors for widening, narrowing, domain and range problems.
- Arithmetic comparison extended to floats, decimals we had already.
- Lexical comparison extended to floats, decimals we had already.
- Lexical comparison extended to references, will throw error.
- Fixed problem hanging of the numbervars/3 predicate for atomic arguments.
- No terminating period when preceded by a relator.
- No terminating period when preceded by an incomplete comment.
- Binary, octal and hexadecimal numbers now parsed.
- End of line sequence in strings not anymore allowed.
- Escape sequences now detected inside strings.
- New predicates peek_char/1, get_char/1, put_char/1.
- New predicates atom_chars/2, number_chars/2, char_code/2.
- Basic stream control predicates introduced.
- Character input/output predicates now also stream term or alias argument.
- Term input/output predicates now also stream term or alias argument.
- Fixed some issue with clause less directives.
- New directive sys_nobox/1 introduced.
- New predicate term_variables/2 introduced.
- New predicate absolute_file_name/2 introduced.

Programming Interface

- Section about special objects written.
- Remark that compareTo() in Term might throw a runtime exception.
- New class TermRef for references.
- New class TermDecimal for decimals.
- The method getValue() of the class TermFloat now returns double.
- Foreign java predicates now accept reference arguments.
- Foreign java predicates now accept Long as decimal arguments.
- Foreign java predicates now accept Double as float arguments.
- New methods get/setBaseUrl() in class Knowledgebase.
- New methods get/setClassPath() in class Knowledgebase.
- New method defineForeign() in class Interpreter.

4.5 Release 0.8.5

The following features were provided for the Jekejeke Prolog runtime library of version 0.8.5:

Language

- Parsing of arguments now starts with level 999.
- Operator definitions for - + now have level 200.
- Directives dynamic, discontiguous and multifile are now prefix operators.
- Bitwise and existential standard operators introduced.
- Operators escape now by all stop characters of context.
- Parenthesizing of operators now only in operator context.
- Comments are now allowed also inside input terms.
- Term syntax now also recognizes negative numbers.
- Operator type yfy not anymore available.
- New syntax errors end_of_clause and end_of_file.
- New syntax errors specification_infix_postfix and specification_prefix_postfix.
- New syntax errors priority_infix_postfix_comma and priority_infix_postfix_bar.
- Directive op now takes a list argument.
- Directives dynamic, discontiguous and multifile now take list arguments.
- Directives dynamic, discontiguous and multifile now take comma arguments.
- Kernel predicate sys_local_cut/0 introduced.
- Logical predicate (->)/2 behaviour fixed.
- Grammar rules (->)/4 behaviour fixed.
- Predicate property sys_invisible and sys_noframe introduced.
- Database predicate clause_term/3 and retract_term/2 introduced.
- Database predicate assertz_term/2 and asserta_term/2 introduced.
- Input/output predicate write_term/2 and read_term/2 introduced.
- Input/output predicate flush_output/0 and put_code/1 introduced.
- Input/output predicate peek_code/1 and get_code/1 introduced.
- Evaluable functions abs/1, sign/1 and float/1 introduced.
- Evaluable functions float_integer_part/1 and float_fractional_part/1 introduced.
- Evaluable functions ceiling/1 and round/1 introduced.
- Evaluable functions bitwise operations introduced.
- Type testing predicate ground/1 introduced.
- Comparison predicate compare/3 and term_hash/2 introduced.
- Set predicates keysort/2 and sort/2 introduced.
- Set predicates bagof/3 and setof/3 introduced.
- Predicate testing run_tests/1 introduced.
- Predicate testing with_input_from/2 and with_output_to/2 introduced.

Programming Interface

- TermFrame, Callin, Callout renamed to CallFrame, CallIn, CallOut.
- CallIn now inherits from CallFrame for improved interaction protocol.
- Method hashCode() for Term introduced.
- Method hasChoices() for CallIn introduced.
- Method unfoldCut() for Interpreter introduced.

4.6 Release 0.8.4

The following features were provided for the Jekejeke Prolog runtime library of version 0.8.4:

Language

- Evaluable function (/)/2 introduced, but with decimal semantics.
- Evaluable functions div/2 and rem/2 semantics fixed.
- System predicate predicate_property/2 does not return foreign/2 anymore.
- System predicate set_prolog_flag/2 introduced.
- System predicate current_prolog_flag/2 introduced.
- Rework of the arithmetic unit, now creates BigInteger only when necessary.
- Body variable elimination optimization introduced.
- Stack frame elimination optimization introduced.
- Head variable elimination optimization introduced.
- System predicate print_exception/1 introduced.
- DCG rules introduced.
- System predicate foreign/1 and foreign/2 introduced.
- Stack trace now always available, not only in debug or trace mode.

Programming Interface

- Additional constructors for the class TermInteger introduced.
- Additional factory methods for the class TermFloat introduced.

4.7 Release 0.8.3

We started the beta-testing campaign with this release. This is our first public release.

5 Library Installation

The library comes in two flavours. There is a version for the Swing Java virtual machine and a version for the Android Java virtual machine.

Concerning the headless use there are the following differences:

Table 1: Headless Differences

Feature	Hotspot	Dalvik
Memory Low	Yes	No
GC Statistics	Yes	No

Concerning the graphic user interface use there are the following differences:

Table 2: Graphical Interface Differences

Feature	Swing	Android
Multi-Window	Yes	No

The code of the libraries is also mostly identical except for some places where different system packages are used. Because of these differences the Android library will not run in a Swing virtual machine, and the Swing library will not run in a Android virtual machine.

In the following we will give more details on the installation of the different versions:

5.1 Swing Installation

A manual package is available for any Swing Java virtual machine. The package only includes the Jekejeke Prolog runtime library. You will download the following archive file:

```
interpreter.jar # Top-Level and Embedding
```

You might copy the archive file to the destination directory `<dest>` of your choice. The archive file can be used for the following purposes:

- **Activation:** The archive file can be used to activate licenses.
- **Top-Level:** The archive file can be used to execute a Prolog query answer loop.
- **Embedding:** The archive file can be embedded into Swing applications.
- **Automatic Discovery:** The archive file discovers class paths and capabilities.
- **System Requirements:** The system requirements of the actual version.

Activation

The archive file can be used to activate licenses. The library archive file itself does not need a license, but additional capabilities might need a license. The activation can be done either with the original archive file or when the archive file has been unpacked and included in a new archive file. The following methods are available for activation:

- **Non-Graphical License Manager:** The non-graphical top level automatically queries the end-user via prompt lines for the activation of licenses. The subsequent top-level section provides more information on how to start the non-graphical top-level.
- **Graphical License Manager:** The graphical top level automatically queries the end-user via dialog windows for the activation of licenses. The subsequent top-level section provides more information on how to start the graphical top-level.
- **Custom License Manager:** Applications and libraries that embed the archive file can code their own license management interactions. The subsequent embedding section provides more information on building applications and libraries.

Top-Level

The archive file can be used to execute a Prolog query answer loop. You will need a Java runtime environment so that you have a java command available. The following command will then execute the archive file and start the Prolog query answer loop without a graphical user interface.

```
java <options> –jar interpreter.jar –h <arguments>
```

To start the Prolog query answer loop with a graphical user interface the following command can be used. On Windows one might also use javaw instead of java:

```
java <options> –jar interpreter.jar <arguments>
```

The following options are recommended:

```
-Duser.language=<language_code>     # Locale language
-Duser.region=<country/area code>   # Locale country
-mx<size>                           # Available memory
-Dapple.laf.useScreenMenuBar=true   # On Mac OS only.
-Dapple.awt.brushMetalLook=true     # On Mac OS only.
-Xdock:name=Jekejeke                # On Mac OS only.
```

Alternatively one can also double click the `interpreter.jar` which executes the archive file with the current default Java runtime and without any options or arguments. This works mostly for Windows and Mac OS, but might fail on Linux.

The archive file accepts further arguments. A detailed documentation of the archive file arguments can be found in the programming interface document for the class ToolkitLibrary.

Embedding

The archive file can be embedded into variety of Hotspot applications. Let's look at the case of an embedding inside a Java standalone application. Assume that your Java class `<main>` has a static method main() and that it resides inside the destination directory `<dest>`. Assume further that this class will use the Hotspot runtime library of Jekejeke Prolog.

You will first need a Java development kit so that you have a Java compiler available. Your Java class `<main>` can be compiled by the following command form the destination directory `<dest>`. Note the different path separators on the different platforms:

```
javac -cp interpreter.jar;. <main>.java    # on windows
javac -cp interpreter.jar:. <main>.java    # on linux and mac
```

You will then need a Java runtime environment so that you have a Java runtime available. Your Java class <main> can be executed by the following command from the destination directory <dest>. Note again the different path separators on the different platforms:

```
java -cp interpreter.jar;. <main>           # on windows
java -cp interpreter.jar:. <main>           # on linux and mac
```

Alternatively you can use an integrated development environment to compile and execute your Java class. All you probably have to do is create an appropriate project and then register the archive file of the Jekejeke Prolog runtime library in the class path of the project.

You might also unpack the Jekejeke Prolog runtime library and include it in a .jar together with your compiled byte code and then execute this .jar.

Further you might want to deploy the Jekejeke Prolog runtime library together with your applets or servlets. In the case of applets all you need to do is mention the archive file in the applet tag and copy the archive file to the web server together with the applet. In case of servlets all you need to do is copy the archive file into the WEB-INF/lib directory. For more details see the deployment study document.

Automatic Discovery

Since release 0.9.12 of the Jekejeke Runtime Library we have facilitated the selection of class paths. This feature is only available for the graphic invocation of the Jekejeke Runtime Library. Upon start-up the interpreter will first check the following directory for additional class path elements:

```
    <working directory>/apk
```

If this directory contains class path elements which are not yet listed in the class path settings or which have not yet been added by the new -a command line option a graphical dialog will be shown to the end-user. The end-user can then decide which additional class path elements from the above directory should be included or excluded.

Since release 0.9.12 of the Jekejeke Runtime Library we have also facilitated the selection of capabilities. This feature is also only available for the graphic invocation of the Jekejeke Runtime Library. Upon start-up of the interpreter and when the class path elements have been registered, the class path elements are search for package slips.

If the package slips contain capabilities which are not yet listed in the capabilities settings or which have not yet been added by the -e command line option a graphical dialog will be shown to the end-user. The end-user can then decide which additional capabilities from the package slips should be included or excluded.

System Requirements

The Jekejeke Prolog runtime library of version 0.9.8 requires at least:

Graphic interface

- Swing 1.6 [4]

Headless

- Hotspot 1.5 [1]

5.2 Android Installation

Manual packages are available for any Android Java virtual machine. The packages only include the Jekejeke Prolog runtime library. You will download the following archive files:

```
interpreter.apk # Top-Level
interpreter.zip # Embedding
```

You might copy the archive files to the destination directory `<dest>` of your choice. The archive files can be used for the following purposes:

- **Activation:** The archive file can be used to activate licenses.
- **Top-Level:** The archive file can be used to execute a Prolog query answer loop.
- **Embedding:** The archive file can be embedded into Android applications.
- **Automatic Discovery:** The archive file discovers class paths and capabilities.
- **System Requirements:** The system requirements of the actual version.

Activation

The archive file can be used to activate licenses. The library archive file itself does not need a license, but additional capabilities might need a license. The activation can be done either with the original archive file or when the archive file has been unpacked and included in a new archive file. The following methods are available for activation:

- **Graphical License Manager:** The graphical top level automatically queries the end-user via dialog windows for the activation of licenses. The subsequent top-level section provides more information on how to start the graphical top-level.
- **Custom License Manager:** Applications and libraries that embed the archive file can code their other license management interactions. The subsequent embedding section provides more information on building applications and libraries.

Top-Level

The archive file can be used to execute a Prolog query answer loop. The archive file can either directly or indirectly be deployed on a device.

For direct deployment change the application preferences on your device to allow download from arbitrary locations. Open a browser on the device and then navigate to the download page of our sales system. Some devices might work better when the sales system is browsed without frames. Finally click on the corresponding download link. This will initiate first a local download and then a local deployment of the archive file on the device.

For indirect deployment you might copy the archive file to the destination directory `<dest>` of your choice and then remotely deploy it to a device. You will then need an Android development kit so that you have a deployment tool. The following step might then do the remote deployment:

- **adb:** Install your Android package on a device.

The above works for an Android device connected via USB or for an Android emulator present on the download platform.

There is no need to unpack the archive file.

Embedding

The archive file can be embedded into variety of Dalvik applications. Let's look at the case of an embedding inside an Android activity. Assume that your Java class `<activity>` derives from the class android.app.Activity and that it resides inside the destination directory `<dest>`. Assume further that this class will use the Dalvik runtime library of Jekejeke Prolog. Further assume that we do cross compilation on a traditional Java platform for an Android emulator or a remote Android device.

You will first need a Java development kit so that you have a Java compiler available. You will also need the Android development kit so that the Android libraries are available. Before you can start compiling your classes the following step might be necessary:

- **aapt:** Compile the manifest and your Android resources.
- **aidl:** Compile your Android interface definitions.

Your Java class `<activity>` can be compiled by the following command form the destination directory `<dest>`. Note the different path separators on the different platforms:

```
# on windows
javac -bootclasspath android.jar \
      -cp interpreter.zip;. \
      <activity>.java
# on linux and mac
javac -bootclasspath android.jar \
      -cp interpreter.zip:. \
      <activity>.java
```

Further steps that are necessary in the process of building an Android package are:

- **dex:** Convert the class files to Dalvik byte code.
- **apkbuilder:** Create an Android package.
- **Jarsigner:** Sign the Android package.
- **zipalign:** Align the Android package.
- **adb:** Install your Android package on a device.

Alternatively you can use an integrated development environment to compile and execute your Java class. All you probably have to do is create an appropriate project and then register the archive file of the Jekejeke Prolog runtime library in the class path of the project. The integrated development environment might invoke the installation for you.

The above works for an Android device connected via USB or for an Android emulator started from the integrated development environment or manually. Alternatively you can upload your Android package to an internet store or to an internet site. Then point your device to the internet store or to the internet site to launch the package.

Automatic Discovery

Since release 0.9.12 of the Jekejeke Runtime Library we have facilitated the selection of class paths. Upon start-up the interpreter will first check the Android package manager for additional class path elements:

```
    packages with the same user id
```

If this list contains class path elements which are not yet listed in the class path settings a graphical dialog will be shown to the end-user. The end-user can then decide which additional class path elements from the above directory should be included or excluded.

Since release 0.9.12 of the Jekejeke Runtime Library we have also facilitated the selection of capabilities. Upon start-up of the interpreter and when the class path elements have been registered, the class path elements are search for package slips.

If the package slips contain capabilities which are not yet listed in the capabilities settings a graphical dialog will be shown to the end-user. The end-user can then decide which additional capabilities from the package slips should be included or excluded.

System Requirements

The Jekejeke Prolog runtime library of version 0.9.8 requires at least:

Graphic interface

- Android 2.2 (API 8) [3]

Headless

- Dalvik 1.6 (API 4) [2]

6 Support Files

Download of the support files is available for all platforms that have a ZIP extractor. The download includes the support files for the Jekejeke Prolog runtime library. You will download the following archive file:

```
suprun.zip        # The support files archive
```

You can use a GUI tool or a commend line tool of your choice that is able to deal with `.zip` files. If all else fails you can use the jar utility that comes with a Java development kit installation. The archive file can be extracted with the following jar utility command. Make sure that you are inside destination directory `<dest>`:

```
jar xf suprun.zip
```

After unpacking the archive one can easily explore its contents with a HTML browser.

The support files archive contains the following kind of support files:

- **Documentation:** The documentation for the runtime library is provided as HTML split files or as full PDF documents.
- **Example Sources:** The source files for the runtime library example programs are provided as ZIP archive files.
- **Interpreter Sources:** The partial source files for the runtime library interpreter are provided as a ZIP archive files.

6.1 Documentation

The documentation for the runtime library is provided as HTML split files or as full PDF documents. The HTML split files can be view with a HTML browser. To view the PDF files a PDF reader needs to be available.

The support files archive contains the following documentation:

```
05_run            # Runtime Library
 +--- 10_docu           # Documentation
 |     +--- 00_android        # User Manual Android
 |     +--- 01_swing          # User Manual Swing
 |     +--- 02_reference      # Language Reference
 |     +--- 03_interface      # Programming Interface
 |     +--- 04_installation   # Installation Guide
 |     +--- 05_frequent       # Frequent Predicates
 +--- 15_stdy           # Studies
       +--- 06_bench          # Benchmark Results
       +--- 07_compliance     # Compliance Results
       +--- 08_deploy         # Deployment Methods
```

The full PDF documents are located in the files called package.pdf in the above directories.

6.2 Example Sources

The source files for the runtime library example programs are provided as source archive files. The source files mainly include Prolog texts and Java classes. But they might also include other types of artefacts.

The support files archive contains the following sources:

```
05_run            # Runtime Library
 +--- 10_docu           # Documentation
 |     +--- 01_swing          # User Manual Swing
 |     +--- 02_reference      # Language Reference
 |     +--- 03_interface      # Programming Interface
 |     +--- 05_frequent       # Frequent Predicates
 +--- 15_stdy           # Studies
       +--- 06_bench          # Benchmark Results
       +--- 07_compliance     # Compliance Results
       +--- 08_deploy         # Deployment Methods
```

The source archive files are located in the files package.zip in the above directories.

You can easily run the programs by means of the Java command line or from within an integrated development environment. Some programs from the deployment methods document demand a web server, an SQL database, a HTML browser or an applet runner for execution. For more details see the corresponding documentation.

6.3 Interpreter Sources

The partial source files for the runtime library interpreter programs are provided as source archive files. The source files mainly include Prolog texts and Java classes. But they might also include other types of artefacts.

The support files archive contains the following sources:

```
05_run                  # Runtime Library
 +--- 02_reference            # Language Reference
 +--- 05_frequent             # Frequent Predicates
```

The source archive files are located in the files package.zip in the above directories.

The sources are mainly there to give a more detailed documentation of the inner working of the interpreter. But the sources can also be used to create derivative work, except for special predicates, which currently come without source.

7 License Conversations

The Jekejeke Prolog runtime library provides character terminal based interactions. Among the interactions we find license management.

- **License Activation:** The console allows activating capabilities on demand either by service or by e-mail.

- **License Registry:** The console allows the management of the enlisted capabilities in case one of them gets invalidated.

7.1 License Activation

The console allows activating capabilities on demand either by service or by e-mail. We can distinguish two possible points in time when license management gets invoked. The first point in time is when a capability is initialized and its license validation fails. The second point in time is when a capability is in use and its license validation fails. The later might happen when the license expires or when the license store is tempered.

Let's first consider the initialization of a capability. This might either happen implicitly during the initialization of a knowledge base for its default capabilities or explicitly by calling for the initialization of a specific capability. If user interaction has been disabled via setting the prompt flag to false both methods will simply throw a license error. If user interaction is enabled via setting the prompt flag to true the license management gets invoked.

In the case of the initialization of a capability the license management will show a prompt with the validation failure and the product description. Here is a possible outcome:

```
Minimal Logic 0.1.0, English
The license could not be found. ?
```

The end-user can choose upon the following options:

```
s = Service,      e = Email,
c = Cancel,       a = Abort,
EOF = Exit,       w = Close ?
```

The meaning of the options is as follows:

- **Service:** When the end-user enters "s" the license manager will allow the activation of the capability via the internet. The license manager will first ask for the license key:

```
Minimal Logic 0.1.0, English ?s
License Key:
```

  The end-user can then enter the license key that he received together with the product. The license manager will the contact the product server via the internet and activate the license.

- **Email:** When the end-user enters "e" the license manager will allow the activation of the capability via e-mail. This method can be used when there is no direct internet access. The license manager will show the install ID. Here is a possible outcome:

```
Minimal Logic 0.1.0, English ?e
Install ID: FgOfxdzUYJYulJhIlySxotZNjukMxNkX83LO1VD6
dQZ6o8IsmEWm0t/FdCEg3mpO8BIMd04t3rF+N1R6
eCSLiw==
```

The end-user should copy the install ID into an e-mail and send that e-mail to the product provider. The product provider then responds with another e-mail that will contain the license text. The license manager will ask for the license text:

```
License Text:
```

The end-user can then enter the license text he received from the product provider. The license manager will then activate the license.

- **Cancel:** When the end-user enters “c” the license manager will stop the activation of the product and return from the initialization with the last license error.
- **Abort:** When the end-user enters “a” the license manager will stop the activation of the product and return to the surrounding query answer loop.
- **Exit:** When the end-user issues an end of file (^D on Mac and Linux, ^Z on Windows) the license manager will stop the activation of the product and leave the surrounding query answer loop.
- **Close:** When the end-user enters “w” the license manager will stop the activation of the product and leave all surrounding query answer loops.

7.2 License Registry

The console allows the management of the enlisted capabilities in case one of them gets invalidated. Let's now consider the situation when capabilities are already in use and the validation of one of the capabilities fails. The license manager will then issue a message stating that at least one capability has turned invalid and prompt the end-user:

```
There are invalid licenses that need an activation. ?
```

The end-user can choose upon the following options:

```
= Continue,       l = List,
a# = Activate,    q = Quit,
EOF = Halt ?
```

The meaning of the options is as follows:

- **Continue:** When the end-user enters "" the license manager will stop the activation of multiple products and resume the execution of the Prolog system.

- **List:** When the end-user enters "l" the license manager will list all currently enlisted capabilities. Capabilities that are invalid are marked with a star (*). Capabilities that can be activated are marked with a period (.). Here is a possible outcome:

```
There are invalid licenses that need an activation. ?l
  0: * Minimal Logic 0.1.0, English
  1:   Runtime Library 0.9.3, English
  2: . Development Environment 0.9.3, English
```

- **Activate:** When the end-user enters "a" and the number of a product the license manager will show a menu so that the product can be activated.

- **Quit:** When the end-user enters "q" the license manager will stop the activation of multiple products and signal all Prolog threads to leave their query loops.

- **Halt:** When the end-user issues an end of file (^D on Mac and Linux, ^Z on Windows) the license manager will stop the activation of multiple products and shutdown the execution of the Prolog system.

8 Known Issues

The issues are grouped as follows:

8.1 Runtime Issues

The following issues are known for the Jekejeke Prolog runtime library of version 1.0.10:

Language

- Should distinguish bookkeeping and non-bookkeeping choice points.
- The throw/1 system predicate should be a cutter, so that no Java stack is needed.
- Should have stream close option “force”.
- Should have memory streams.
- The listing/[0,1] predicates should show operator properties.
- The listing/[0,1] predicates should show pretty source imports.
- Should have a meta-argument specifier for evaluable expression.
- The answer loop REPL should do goal expansion.
- Should have help/[0,1] predicates which shows only predicate declarations.
- The help/[0,1] predicates should show static predicates independent of clauses.
- Should have re-export facades for various Prolog systems such as SWI-Prolog.
- Should have mechanism to switch on and off the library path for re-export facades.
- The (:)/2 operator should allow Prolog objects.
- The module system should have inner modules.
- The parallel loader should block module calls until module complete.
- .

Programming Interface

- The foreign function interface should support Java array types.
- The (:)/2 operator should allow auto loaded Java objects.
- Distinguish primitive and non-primitive number types for specificity.
- Introduce instance of test predicates for method dispatching.
- Take into account inheritance distance for specificity.
- Auto loader should also probe relative path and class loader path.
- Module prefixes should use auto loader to assure module is there.
- Should have a Prolog constant that can be mapped to the Java null value.
- .

Frequent Predicates

- The proxy interface should support Java array types.
- The proxy interface should support Iterator interface members.
- The (:)/2 operator should allow proxy generated Java objects.
- The proxy interface needs un-mapping of Prolog IOException to Java.
- The proxy interface needs un-mapping of Prolog Throwable to Java.
- .

8.2 Installation Issues

The following issues are known for the Jekejeke Prolog runtime library of version 1.0.4:

Installation

- HTML documentation should use style sheets.
- Cross reference tool should find modules auto load style.
- .

8.3 Compliance Issues

The following issues are known for the Jekejeke Prolog runtime library of version 1.0.4:

Compliance

- The push-back is missing in the DCG test cases.
- The meta-call is missing in the DCG test cases.
- The type error is missing in the DCG test cases.
- .

8.4 Swing Issues

The following issues are known for the Jekejeke Prolog runtime library of version 1.0.9:

Swing Interface

- Should use context menus for example for tabs.
- Beep missing when currently disabled accelerators are pressed.
- The hold menu item should suspend running thread.
- Not yet a dock icon on the Mac platform.
- Not yet correct about menu and preferences menu on the Mac platform.
- Soft signal interrupted reads are shown on same line.
- Should combine type characters and backspace into one undo action.
- Text pane search should be done case insensitive.
- Should have line wrapping like CSS pre-wrap.
- Should have an API for a later invoke on the Swing thread.
- .

8.5 Android Issues

The following issues are known for the Jekejeke Prolog runtime library of version 1.0.9:

Android Interface

- Should update activity title when locale changes.
- Should use if room menu items in action bar.
- Should use context menus for example for tabs.
- Console text should optimize runs of identical character styles.
- Console text should react to arrival of newline from input channel.
- Email panel should have email link.
- Should have an API for a later invoke on the Android thread.
- .

Pictures

Es konnten keine Einträge für ein Abbildungsverzeichnis gefunden werden.

Tables

References

[1] Java 2 Platform Standard Edition 5.0, Tiger, Sun Microsystems, 2004
http://www.oracle.com/technetwork/java/javase/index-jsp-135232.html
[2] Android 1.6 Platform, Donut, Google Inc., September 2009
http://developer.android.com/sdk/android-1.6.html
[3] Android 2.2 Platform, Froyo, Google Inc., May 2010
http://developer.android.com/sdk/android-2.2.html
[4] Java 2 Platform Standard Edition 6.0, Mustang, Sun Microsystems, 2006
http://www.oracle.com/technetwork/java/javase/overview/index-jsp-136246.html