Nim Standard Library 0.14.3

Andreas Rumpf

June 29, 2016

Contents

"The good thing about reinventing the wheel is that you can get a round one."

Though the Nim Standard Library is still evolving, it is already quite usable. It is divided into *pure libraries*, *impure libraries* and *wrappers*.

Pure libraries do not depend on any external `*.dll` or `lib*.so` binary while impure libraries do. A wrapper is an impure library that is a very low-level interface to a C library.

Read this document for a quick overview of the API design.

The bottom of this page includes a list of 3rd party packages created by the Nim community. These packages are a useful addition to the modules in the standard library.

1 Pure libraries

1.1 Core

- system Basic procs and operators that every program needs. It also provides IO facilities for reading and writing text and binary files. It is imported implicitly by the compiler. Do not import it directly. It relies on compiler magic to work.
- threads Nim thread support. **Note**: This is part of the system module. Do not import it explicitly.
- channels Nim message passing support for threads. **Note**: This is part of the system module. Do not import it explicitly.
- locks Locks and condition variables for Nim.
- rlocks Reentrant locks for Nim.
- macros Contains the AST API and documentation of Nim for writing macros.
- typeinfo Provides (unsafe) access to Nim's run time type information.
- typetraits This module defines compile-time reflection procs for working with types.
- threadpool Implements Nim's spawn.
- cpuinfo This module implements procs to determine the number of CPUs / cores.

1.2 Collections and algorithms

- algorithm Implements some common generic algorithms like sort or binary search.
- tables Nim hash table support. Contains tables, ordered tables and count tables.
- sets Nim hash and bit set support.
- lists Nim linked list support. Contains singly and doubly linked lists and circular lists ("rings").
- queues Implementation of a queue. The underlying implementation uses a `seq`.
- intsets Efficient implementation of a set of ints as a sparse bit set.
- critbits This module implements a *crit bit tree* which is an efficient container for a set or a mapping of strings.
- sequtils This module implements operations for the built-in seq type which were inspired by functional programming languages.

1.3 String handling

- strutils This module contains common string handling operations like changing case of a string, splitting a string into substrings, searching for substrings, replacing substrings.
- strmisc This module contains uncommon string handling operations that do not fit with the commonly used operations in strutils.
- parseutils This module contains helpers for parsing tokens, numbers, identifiers, etc.
- strscans This module contains a `scanf` macro for convenient parsing of mini languages.
- strtabs The `strtabs` module implements an efficient hash table that is a mapping from strings to strings. Supports a case-sensitive, case-insensitive and style-insensitive mode. An efficient string substitution operator `%` for the string table is also provided.
- unicode This module provides support to handle the Unicode UTF-8 encoding.
- encodings Converts between different character encodings. On UNIX, this uses the `iconv` library, on Windows the Windows API.
- pegs This module contains procedures and operators for handling PEGs.
- ropes This module contains support for a *rope* data type. Ropes can represent very long strings efficiently; especially concatenation is done in O(1) instead of O(n).
- matchers This module contains various string matchers for email addresses, etc.
- subexes This module implements advanced string substitution operations.

1.4 Generic Operating System Services

- os Basic operating system facilities like retrieving environment variables, reading command line arguments, working with directories, running shell commands, etc.
- osproc Module for process communication beyond `os.execShellCmd`.
- times The `times` module contains basic support for working with time.
- dynlib This module implements the ability to access symbols from shared libraries.
- streams This module provides a stream interface and two implementations thereof: the *FileStream* and the *StringStream* which implement the stream interface for Nim file objects (*File*) and strings. Other modules may provide other implementations for this standard stream interface.
- marshal Contains procs for serialization and deseralization of arbitrary Nim data structures.
- terminal This module contains a few procedures to control the *terminal* (also called *console*). The implementation simply uses ANSI escape sequences and does not depend on any other module.
- memfiles This module provides support for memory mapped files (Posix's `mmap`) on the different operating systems.
- fsmonitor This module implements the ability to monitor a directory/file for changes using Posix's inotify API.

  Warning: This module will likely be moved out to a Nimble package soon.
- asyncfile This module implements asynchronous file reading and writing using `asyncdispatch`.

1.5 Math libraries

- math Mathematical operations like cosine, square root.
- complex This module implements complex numbers and their mathematical operations.
- rationals This module implements rational numbers and their mathematical operations.
- fenv Floating-point environment. Handling of floating-point rounding and exceptions (overflow, zero-devide, etc.).
- basic2d Basic 2d support with vectors, points, matrices and some basic utilities.
- basic3d Basic 3d support with vectors, points, matrices and some basic utilities.
- mersenne Mersenne twister random number generator.
- random Fast and tiny random number generator.
- stats Statistical analysis

1.6 Internet Protocols and Support

- cgi This module implements helpers for CGI applications.
- scgi This module implements helpers for SCGI applications.
- browsers This module implements procs for opening URLs with the user's default browser.
- httpserver This module implements a simple HTTP server.
- httpclient This module implements a simple HTTP client which supports both synchronous and asynchronous retrieval of web pages.
- smtp This module implement a simple SMTP client.
- cookies This module contains helper procs for parsing and generating cookies.
- mimetypes This module implements a mimetypes database.
- uri This module provides functions for working with URIs.
- asyncdispatch This module implements an asynchronous dispatcher for IO operations.
- asyncnet This module implements asynchronous sockets based on the `asyncdispatch` module.
- asynchttpserver This module implements an asynchronous HTTP server using the `asyncnet` module.
- asyncftpclient This module implements an asynchronous FTP client using the `asyncnet` module.
- net This module implements a high-level sockets API. It will replace the `sockets` module in the future.
- nativesockets This module implements a low-level sockets API.
- selectors This module implements a selector API with backends specific to each OS. Currently epoll on Linux and select on other operating systems.

1.7 Parsers

- parseopt The `parseopt` module implements a command line option parser.
- parseopt2 The `parseopt2` module implements a command line option parser. This supports long and short command options with optional values and command line arguments.
- parsecfg The `parsecfg` module implements a high performance configuration file parser. The configuration file's syntax is similar to the Windows `.ini` format, but much more powerful, as it is not a line based parser. String literals, raw string literals and triple quote string literals are supported as in the Nim programming language.
- parsexml The `parsexml` module implements a simple high performance XML/HTML parser. The only encoding that is supported is UTF-8. The parser has been designed to be somewhat error correcting, so that even some "wild HTML" found on the Web can be parsed with it.
- parsecsv The `parsecsv` module implements a simple high performance CSV parser.
- parsesql The `parsesql` module implements a simple high performance SQL parser.
- json High performance JSON parser.
- lexbase This is a low level module that implements an extremely efficient buffering scheme for lexers and parsers. This is used by the diverse parsing modules.
- highlite Source highlighter for programming or markup languages. Currently only few languages are supported, other languages may be added. The interface supports one language nested in another.
- rst This module implements a reStructuredText parser. A large subset is implemented. Some features of the markdown wiki syntax are also supported.
- rstast This module implements an AST for the reStructuredText parser.
- rstgen This module implements a generator of HTML/Latex from reStructuredText.
- sexp High performance sexp parser and generator, mainly for communication with emacs.

1.8 XML Processing

- xmldom This module implements the XML DOM Level 2.
- xmldomparser This module parses an XML Document into a XML DOM Document representation.
- xmltree A simple XML tree. More efficient and simpler than the DOM. It also contains a macro for XML/HTML code generation.
- xmlparser This module parses an XML document and creates its XML tree representation.
- htmlparser This module parses an HTML document and creates its XML tree representation.
- htmlgen This module implements a simple XML and HTML code generator. Each commonly used HTML tag has a corresponding macro that generates a string with its HTML representation.

1.9 Cryptography and Hashing

- hashes This module implements efficient computations of hash values for diverse Nim types.
- md5 This module implements the MD5 checksum algorithm.
- base64 This module implements a base64 encoder and decoder.
- securehash This module implements a sha1 encoder and decoder.

1.10 Multimedia support

- colors This module implements color handling for Nim. It is used by the `graphics` module.

1.11 Miscellaneous

- events This module implements an event system that is not dependent on external graphical toolkits.
- oids An OID is a global ID that consists of a timestamp, a unique counter and a random value. This combination should suffice to produce a globally distributed unique ID. This implementation was extracted from the Mongodb interface and it thus binary compatible with a Mongo OID.
- endians This module contains helpers that deal with different byte orders.
- logging This module implements a simple logger.
- options Types which encapsulate an optional value.
- future This module implements new experimental features. Currently the syntax sugar for anonymous procedures.
- coro This module implements experimental coroutines in Nim.
- unittest Implements a Unit testing DSL.

1.12 Modules for JS backend

- dom Declaration of the Document Object Model for the JS backend.

1.13 Deprecated modules

- asyncio This module implements an asynchronous event loop for sockets. **Deprecated since version 0.11.2:** Use the asyncnet together with the asyncdispatch module instead.
- ftpclient This module implements an FTP client. **Deprecated since version 0.11.3:** Use the asyncftpclient module instead.
- sockets This module implements a simple portable type-safe sockets layer. **Deprecated since version 0.11.2:** Use the net or the rawsockets module instead.
- rawsockets **Deprecated since version 0.11.4:** This module has been renamed to nativesockets.

2 Impure libraries

2.1 Regular expressions

- re This module contains procedures and operators for handling regular expressions. The current implementation uses PCRE.
- nre Another implementation of procedures for using regular expressions. Also uses PCRE.

2.2 Database support

- db_postgres A higher level PostgreSQL database wrapper. The same interface is implemented for other databases too.
- db_mysql A higher level MySQL database wrapper. The same interface is implemented for other databases too.
- db_sqlite A higher level SQLite database wrapper. The same interface is implemented for other databases too.

2.3 Other

- ssl This module provides an easy to use sockets-style Nim interface to the OpenSSL library.

3 Wrappers

The generated HTML for some of these wrappers is so huge that it is not contained in the distribution. You can then find them on the website.

3.1 Windows specific

- winlean Contains a wrapper for a small subset of the Win32 API.

3.2 UNIX specific

- posix Contains a wrapper for the POSIX standard.

3.3 Regular expressions

- pcre Wrapper for the PCRE library.

3.4 GUI libraries

- iup Wrapper of the IUP GUI library.

3.5 Database support

- postgres Contains a wrapper for the PostgreSQL API.
- mysql Contains a wrapper for the mySQL API.
- sqlite3 Contains a wrapper for SQLite 3 API.
- odbcsql interface to the ODBC driver.

3.6 Network Programming and Internet Protocols

- libuv Wrapper for the libuv library used for async I/O programming.
- joyent_http_parser Wrapper for the joyent's high-performance HTTP parser.
- libcurl Wrapper for the libcurl library.
- openssl Wrapper for OpenSSL.

3.7 Scientific computing

- libsvm Low level wrapper for lib svm.

4 Nimble

Nimble is a package manager for the Nim programming language. For instructions on how to install Nimble packages see its README.

4.1 Official packages

These packages are officially supported and will therefore be continually maintained to ensure that they work with the latest versions of the Nim compiler.

4.2 Unofficial packages

These packages have been developed by independent Nim developers and as such may not always be up to date with the latest developments in the Nim programming language.