NAME

CPAN::Meta::History::Meta_1_1 - Version 1.1 metadata specification for META.yml

PREFACE

This is a historical copy of the version 1.1 specification for *META.yml* files, copyright by Ken Williams.

Modifications from the original:

- Conversion from the original HTML to POD format
- Include list of valid licenses from Module::Build 0.18 rather than linking to the module.

DESCRIPTION

This document describes version 1.1 of the *META.yml* specification.

The *META.yml* file describes important properties of contributed Perl distributions such as the ones found on CPAN <http://www.cpan.org>.
It is typically created by tools like Module::Build and ExtUtils::MakeMaker.

The fields in the *META.yml* file are meant to be helpful to people maintaining module collections (like CPAN), for people writing installation tools (like CPAN or CPANPLUS), or just people who want to know some stuff about a distribution before downloading it and starting to install it.

Format

META.yml files are written in the YAML <http://www.yaml.org/> format. The reasons we chose YAML instead of, say, XML or Data::Dumper are discussed in this thread <http://archive.develooper.com/makemaker@perl.org/msg00405.html> on the MakeMaker mailing list.

The first line of a *META.yml* file should be a valid YAML document header <http://www.yaml.org/spec/#.Document> like `"--- #YAML:1.0"`

Fields

The rest of the META.yml file is one big YAML mapping <http://www.yaml.org/spec/#.-syntax-mapping-Mapping->, whose keys are described here.

name
: Example: `Module-Build`

  The name of the distribution. Often created by taking the ''main module'' in the distribution and changing ''::'' to ''-''. Sometimes it's completely different, however, as in the case of the libwww-perl <http://search.cpan.org/author/GAAS/libwww-perl/> distribution.

version
: Example: `0.16`

  The version of the distribution to which the META.yml file refers. This is a mandatory field.

  The version is essentially an arbitrary string, but *must* be only ASCII characters, and *strongly should* be of the format integer-dot-digit-digit, i.e. `25.57`, optionally followed by underscore-digit-digit, i.e. `25.57_04`.

  The standard tools that deal with module distribution (PAUSE, CPAN, etc.) form an identifier for each distribution by joining the 'name' and 'version' attributes with a dash (`-`) character. Tools who are prepared to deal with distributions that have no version numbers generally omit the dash as well.

license
: Example: `perl`

  a descriptive term for the licenses ... not authoritative, but must be consistent with licensure statements in the READMEs, documentation, etc.

  The license under which this distribution may be used and redistributed.

Must be one of the following licenses:

perl
: The distribution may be copied and redistributed under the same terms as perl itself (this is by far the most common licensing option for modules on CPAN). This is a dual license, in which the user may choose between either the GPL or the Artistic license.

gpl
: The distribution is distributed under the terms of the Gnu General Public License (<http://www.opensource.org/licenses/gpl-license.php>).

lgpl
: The distribution is distributed under the terms of the Gnu Lesser General Public License (<http://www.opensource.org/licenses/lgpl-license.php>).

artistic
: The distribution is licensed under the Artistic License, as specified by the Artistic file in the standard perl distribution.

bsd
: The distribution is licensed under the BSD License (<http://www.opensource.org/licenses/bsd-license.php>).

open_source
: The distribution is licensed under some other Open Source Initiative-approved license listed at <http://www.opensource.org/licenses/>.

unrestricted
: The distribution is licensed under a license that is **not** approved by www.opensource.org <http://www.opensource.org> but that allows distribution without restrictions.

restrictive
: The distribution may not be redistributed without special permission from the author and/or copyright holder.

license_uri
: This should contain a URI where the exact terms of the license may be found.

  (change ''unrestricted'' to ''redistributable''?)

distribution_type
: Example: `module`

  What kind of stuff is contained in this distribution. Most things on CPAN are `modules` (which can also mean a collection of modules), but some things are `scripts`.

  This field is basically meaningless, and tools (like Module::Build or MakeMaker) will likely stop generating it in the future.

private
: WTF is going on here?

  index_ignore: any application that indexes the contents of distributions (PAUSE, search.cpan.org) ought to ignore the items (packages, files, directories, namespace hierarchies).

requires
: Example:

```
Data::Dumper: 0
File::Find: 1.03
```

  A YAML mapping <http://www.yaml.org/spec/#.-syntax-mapping-Mapping-> indicating the Perl modules this distribution requires for proper operation. The keys are the module names, and the values are version specifications as described in the Module::Build.

Note: the exact nature of the fancy specifications like `">= 1.2, != 1.5, < 2.0"` is subject to change. Advance notice will be given here. The simple specifications like `"1.2"` will not change in format.

recommends
: Example:

```
Data::Dumper: 0
File::Find: 1.03
```

A YAML mapping <http://www.yaml.org/spec/#.-syntax-mapping-Mapping-> indicating the Perl modules this distribution recommends for enhanced operation.

build_requires
: Example:

```
Data::Dumper: 0
File::Find: 1.03
```

A YAML mapping <http://www.yaml.org/spec/#.-syntax-mapping-Mapping-> indicating the Perl modules required for building and/or testing of this distribution. These dependencies are not required after the module is installed.

conflicts
: Example:

```
Data::Dumper: 0
File::Find: 1.03
```

A YAML mapping <http://www.yaml.org/spec/#.-syntax-mapping-Mapping-> indicating the Perl modules that cannot be installed while this distribution is installed. This is a pretty uncommon situation.

- possibly separate out test-time prereqs, complications include: can tests be meaningfully preserved for later running? are test-time prereqs in addition to build-time, or exclusive?

- make official location for installed *distributions*, which can contain tests, etc.

dynamic_config
: Example: `0`

A boolean flag indicating whether a *Build.PL* or *Makefile.PL* (or similar) must be executed, or whether this module can be built, tested and installed solely from consulting its metadata file. The main reason to set this to a true value if that your module performs some dynamic configuration (asking questions, sensing the environment, etc.) as part of its build/install process.

Currently Module::Build doesn't actually do anything with this flag - it's probably going to be up to higher-level tools like CPAN to do something useful with it. It can potentially bring lots of security, packaging, and convenience improvements.

generated_by
: Example: `Module::Build version 0.16`

Indicates the tool that was used to create this *META.yml* file. It's good form to include both the name of the tool and its version, but this field is essentially opaque, at least for the moment.

Ingy's suggestions

short_description
: add as field, containing abstract, maximum 80 characters, suggested minimum 40 characters

description
: long version of abstract, should add?

maturity
: alpha, beta, gamma, mature, stable

author_id, owner_id
categorization, keyword, chapter_id
URL for further information
: could default to search.cpan.org on PAUSE

namespaces
: can be specified for single elements by prepending dotted-form, i.e. ''com.example.my_application.my_property''. Default namespace for META.yml is probably ''org.cpan.meta_author'' or something. Precedent for this is Apple's Carbon namespaces, I think.

History

- **March 14, 2003**(Pi da y) - created version 1.0 of this document.
- **May 8, 2003**- added the ''dynamic_config'' field, whic h was missing from the initial version.