NAME

Lintian::Tags – Manipulate and output Lintian tags

SYNOPSIS

```
my $tags = Lintian::Tags->new;
my $proc = Lintian::Processable::Package->new ('/path/to/file');
$tags->file_start ($proc);
$tags->file_overrides ('/path/to/an/overrides-file');
$tags->tag ('lintian-tag', 'extra tag information');
tag ('other-lintian-tag', 'with some extra data');
tag ('third-lintian-tag'); # with no extra).
my %overrides = $tags->overrides ($proc);
my %stats = $tags->statistics;
if ($tags->displayed ('lintian-tag')) {
    # do something if that tag would be displayed...
}
```

DESCRIPTION

This module stores metadata about Lintian tags, stores configuration about which tags should be displayed, handles displaying tags if appropriate, and stores cumulative statistics about what tags have been seen. It also accepts override information and determines whether a tag has been overridden, keeping override statistics. Finally, it supports answering metadata questions about Lintian tags, such as what references Lintian has for that tag.

Each Lintian::Tags object has its own tag list, file list, and associated statistics. Separate Lintian::Tags objects can be maintained and used independently. However, as a convenience for Lintian's most typical use case and for backward compatibility, the first created Lintian::Tags object is maintained as a global default. The **tag()** method can be called as a global function instead of a method, in which case it will act on that global default Lintian::Tags object.

CLASS METHODS

new()

Creates a new Lintian::Tags object, initializes all of its internal statistics and configuration to the defaults, and returns the newly created object.

tag(TAG, [EXTRA, ...])

Issue the Lintian tag TAG, possibly suppressing it or not displaying it based on configuration. EXTRA, if present, is additional information to display with the tag. It can be given as a list of strings, in which case they're joined by a single space before display.

This method can be called either as a class method (which is exported by the Lintian::Tags module) or as an instance method. If called as a class method, it uses the first-constructed Lintian::Tags object as its underlying object.

This method throws an exception if it is called without **file_start()** being called first or if an attempt is made to issue an unknown tag.

INSTANCE METHODS

Configuration

display(OPERATION, RELATION, SEVERITY, CERTAINTY)

Configure which tags are displayed by severity and certainty. OPERATION is + to display the indicated tags, − to not display the indicated tags, or = to not display any tags except the indicated ones. RELATION is one of <, <=, =, >=, or >. The OPERATION will be applied to all pairs of severity and certainty that match the given RELATION on the SEVERITY and CERTAINTY arguments. If either of those arguments are undefined, the action applies to any value for that variable. For example:

```
$tags->display('=', '>=', 'important');
```

turns off display of all tags and then enables display of any tag (with any certainty) of severity important or higher.

```
$tags->display('+', '>', 'normal', 'possible');
```

adds to the current configuration display of all tags with a severity higher than normal and a certainty higher than possible (so important/certain and serious/certain).

```
$tags->display('-', '=', 'minor', 'possible');
```

turns off display of tags of severity minor and certainty possible.

This method throws an exception on errors, such as an unknown severity or certainty or an impossible constraint (like `> serious`).

show_experimental(BOOL)
: If BOOL is true, configure experimental tags to be shown. If BOOL is false, configure experimental tags to not be shown.

show_overrides(BOOL)
: If BOOL is true, configure overridden tags to be shown. If BOOL is false, configure overridden tags to not be shown.

sources([SOURCE [, ...]])
: Limits the displayed tags to only those from the listed sources. One or more sources may be given. If no sources are given, resets the Lintian::Tags object to display tags from any source. Tag sources are the names of references from the Ref metadata for the tags.

profile(PROFILE)
: Use the PROFILE (Lintian::Profile) to determine which tags are suppressed, the severity of the tags and which tags are non-overridable.

File Metadata

file_start(PROC)
: Adds a new file from a processable, initializes the data structures used for statistics and overrides, and makes it the default file for which tags will be issued. Also call **Lintian::Output::print_end_pkg()** to end the previous file, if any, and **Lintian::Output::print_start_pkg()** to start the new file.

This method throws an exception if the file being added was already added earlier.

file_overrides(OVERRIDE−FILE)
: Read OVERRIDE-FILE and add the overrides found there which match the metadata of the current file (package and type). The overrides are added to the overrides hash in the info hash entry for the current file.

file_start() must be called before this method. This method throws an exception if there is no current file and calls **fail()** if the override file cannot be opened.

load_overrides
: Loads overrides for the current file. This is basically a short-hand for finding the overrides file in the lab and calling files_overrides on it if it is present.

file_end()
: Ends processing of a file.

This does two things. First it emits ‘‘unused-override’’ tags for all unused overrides. Secondly, it calls Lintian::Output::print_end_pkg to mark the end of the package.

Note that this method is called by file_start if it detects another entry is already active.

Statistics

overrides(PROC)
: Returns a reference to the overrides hash for the given processable. The keys of this hash are the tags for which are overrides. The value for each key is another hash, whose keys are the extra data matched by that override and whose values are the counts of tags that matched that override. Overrides matching any tag by that name are stored with the empty string as metadata, so:

```
my $overrides = $tags->overrides('/some/file');
print "$overrides->{'some-tag'}{''}\n";
```

will print out the number of tags that matched a general override for the tag some-tag, regardless of what extra data was associated with it.

statistics([PROC])

Returns a reference to the statistics hash for the given processable or, if PROC is omitted, a reference to the full statistics hash for all files. In the latter case, the returned hash reference has as keys the file names and as values the per-file statistics.

The per-file statistics has a set of hashes of keys to times seen in tags: tag names (the `tags` key), severities (the `severity` key), certainties (the `certainty` key), and tag codes (the `types` key). It also has an `overrides` key which has as its value another hash with those same four keys, which keeps statistics on overridden tags (not included in the regular counts).

Tag Reporting

displayed(TAG)

Returns true if the given tag would be displayed given the current configuration, false otherwise. This does not check overrides, only whether the tag severity, certainty, and source warrants display given the configuration.

suppressed(TAG)

Returns true if the given tag would be suppressed given the current configuration, false otherwise. This is different than **displayed()** in that a tag is only suppressed if Lintian treats the tag as if it's never been seen, doesn't update statistics, and doesn't change its exit status. Tags are suppressed via **profile()**.

ignored_overrides()

Returns a hash of tags, for which overrides have been ignored. The keys are tag names and the value is the number of overrides that has been ignored.

AUTHOR

Originally written by Russ Allbery <rra@debian.org> for Lintian.

SEE ALSO

lintian (1), **Lintian::Output** (3), **Lintian::Tag::Info** (3)