NAME

config_data - Query or change configuration of Perl modules

SYNOPSIS

```
# Get config/feature values
config_data --module Foo::Bar --feature bazzable
config_data --module Foo::Bar --config magic_number

# Set config/feature values
config_data --module Foo::Bar --set_feature bazzable=1
config_data --module Foo::Bar --set_config magic_number=42

# Print a usage message
config_data --help
```

DESCRIPTION

The `config_data` tool provides a command-line interface to the configuration of Perl modules. By "configuration", we mean something akin to "user preferences" or "local settings". This is a formalization and abstraction of the systems that people like Andreas Koenig (`CPAN::Config`), Jon Swartz (`HTML::Mason::Config`), Andy Wardley (`Template::Config`), and Larry Wall (perl's own Config.pm) have developed independently.

The configuration system emplyed here was developed in the context of `Module::Build`. Under this system, configuration information for a module `Foo`, for example, is stored in a module called `Foo::ConfigData`) (I would have called it `Foo::Config`, but that was taken by all those other systems mentioned in the previous paragraph...). These `...::ConfigData` modules contain the configuration data, as well as publically accessible methods for querying and setting (yes, actually re-writing) the configuration data. The `config_data` script (whose docs you are currently reading) is merely a front-end for those methods. If you wish, you may create alternate front-ends.

The two types of data that may be stored are called `config` values and `feature` values. A `config` value may be any perl scalar, including references to complex data structures. It must, however, be serializable using `Data::Dumper`. A `feature` is a boolean (1 or 0) value.

USAGE

This script functions as a basic getter/setter wrapper around the configuration of a single module. On the command line, specify which module's configuration you're interested in, and pass options to get or set `config` or `feature` values. The following options are supported:

module

Specifies the name of the module to configure (required).

feature

When passed the name of a `feature`, shows its value. The value will be 1 if the feature is enabled, 0 if the feature is not enabled, or empty if the feature is unknown. When no feature name is supplied, the names and values of all known features will be shown.

config

When passed the name of a `config` entry, shows its value. The value will be displayed using `Data::Dumper` (or similar) as perl code. When no config name is supplied, the names and values of all known config entries will be shown.

set_feature

Sets the given `feature` to the given boolean value. Specify the value as either 1 or 0.

set_config

Sets the given `config` entry to the given value.

eval

If the `--eval` option is used, the values in `set_config` will be evaluated as perl code before being stored. This allows moderately complicated data structures to be stored. For really complicated structures, you probably shouldn't use this command-line interface, just use the Perl API instead.

help

Prints a help message, including a few examples, and exits.

AUTHOR

Ken Williams, kwilliams@cpan.org

COPYRIGHT

SEE ALSO

Module::Build(3), perl(1).