NAME

CPANPLUS::Dist::MM

SYNOPSIS

```
$mm = CPANPLUS::Dist::MM->new( module => $modobj );

$mm->create;        # runs make && make test
$mm->install;       # runs make install
```

DESCRIPTION

`CPANPLUS::Dist::MM` is a distribution class for MakeMaker related modules. Using this package, you can create, install and uninstall perl modules. It inherits from `CPANPLUS::Dist`.

ACCESSORS

parent()

Returns the `CPANPLUS::Module` object that parented this object.

status()

Returns the `Object::Accessor` object that keeps the status for this module.

STATUS ACCESSORS

All accessors can be accessed as follows: $mm->status->ACCESSOR

makefile ()

Location of the Makefile (or Build file). Set to 0 explicitly if something went wrong.

make ()

BOOL indicating if the `make` (or `Build`) command was successful.

test ()

BOOL indicating if the `make test` (or `Build test`) command was successful.

prepared ()

BOOL indicating if the `prepare` call exited succesfully This gets set after `perl Makefile.PL`

distdir ()

Full path to the directory in which the `prepare` call took place, set after a call to `prepare`.

created ()

BOOL indicating if the `create` call exited succesfully. This gets set after `make` and `make test`.

installed ()

BOOL indicating if the module was installed. This gets set after `make install` (or `Build install`) exits successfully.

uninstalled ()

BOOL indicating if the module was uninstalled properly.

_create_args ()

Storage of the arguments passed to `create` for this object. Used for recursive calls when satisfying prerequisites.

_install_args ()

Storage of the arguments passed to `install` for this object. Used for recursive calls when satisfying prerequisites.

METHODS

$bool = $dist->format_available();

Returns a boolean indicating whether or not you can use this package to create and install modules in your environment.

Sets up the `CPANPLUS::Dist::MM` object for use. Effectively creates all the needed status accessors.

Called automatically whenever you create a new `CPANPLUS::Dist` object.

`prepare` preps a distribution for installation. This means it will run `perl Makefile.PL` and determine what prerequisites this distribution declared.

If you set `force` to true, it will go over all the stages of the `prepare` process again, ignoring any previously cached results.

When running `perl Makefile.PL`, the environment variable `PERL5_CPANPLUS_IS_EXECUTING` will be set to the full path of the `Makefile.PL` that is being executed. This enables any code inside the `Makefile.PL` to know that it is being installed via CPANPLUS.

Returns true on success and false on failure.

You may then call `$dist->create` on the object to create the installable files.

$href = $dist->_find_prereqs(file => '/path/to/Makefile', [verbose => BOOL])

Parses a `Makefile` for `PREREQ_PM` entries and distills from that any prerequisites mentioned in the `Makefile`

Returns a hash with module-version pairs on success and false on failure.

$bool = $dist->create([perl => '/path/to/perl', make => '/path/to/make', makeflags => 'EXTRA=FLAGS', prereq_target => TARGET, skiptest => BOOL, force => BOOL, verbose => BOOL])

`create` creates the files necessary for installation. This means it will run `make` and `make test`. This will also scan for and attempt to satisfy any prerequisites the module may have.

If you set `skiptest` to true, it will skip the `make test` stage. If you set `force` to true, it will go over all the stages of the `make` process again, ignoring any previously cached results. It will also ignore a bad return value from `make test` and still allow the operation to return true.

Returns true on success and false on failure.

You may then call `$dist->install` on the object to actually install it.

$bool = $dist->install([make => '/path/to/make', makemakerflags => 'EXTRA=FLAGS', force => BOOL, verbose => BOOL])

`install` runs the following command: make install

Returns true on success, false on failure.

$bool = $dist->write_makefile_pl([force => BOOL, verbose => BOOL])

This routine can write a `Makefile.PL` from the information in a module object. It is used to write a `Makefile.PL` when the original author forgot it (!!).

Returns 1 on success and false on failure.

The file gets written to the directory the module's been extracted to.