NAME

CPANPLUS::Dist::Base - Base class for custom distribution classes

SYNOPSIS

```
package CPANPLUS::Dist::MY_IMPLEMENTATION

use base 'CPANPLUS::Dist::Base';

sub prepare {
    my $dist = shift;

    ### do the 'standard' things
    $dist->SUPER::prepare( @_ ) or return;

    ### do MY_IMPLEMENTATION specific things
    ...

    ### don't forget to set the status!
    return $dist->status->prepared( $SUCCESS ? 1 : 0 );
}
```

DESCRIPTION

CPANPLUS::Dist::Base functions as a base class for all custom distribution implementations. It does all the mundane work CPANPLUS would have done without a custom distribution, so you can override just the parts you need to make your own implementation work.

FLOW

Below is a brief outline when and in which order methods in this class are called:

```
$Class->format_available;   # can we use this class on this system?

$dist->init;                # set up custom accessors, etc
$dist->prepare;             # find/write meta information
$dist->create;              # write the distribution file
$dist->install;             # install the distribution file

$dist->uninstall;           # remove the distribution (OPTIONAL)
```

METHODS

@subs = $Class->methods

Returns a list of methods that this class implements that you can override.

$bool = $Class->format_available

This method is called when someone requests a module to be installed via the superclass. This gives you the opportunity to check if all the needed requirements to build and install this distribution have been met.

For example, you might need a command line program, or a certain perl module installed to do your job. Now is the time to check.

Simply return true if the request can proceed and false if it can not.

The `CPANPLUS::Dist::Base` implementation always returns true.

$bool = $dist->init

This method is called just after the new dist object is set up and before the `prepare` method is called. This is the time to set up the object so it can be used with your class.

For example, you might want to add extra accessors to the `status` object, which you might do as follows:

```
$dist->status->mk_accessors( qw[my_implementation_accessor] );
```

The `status` object is implemented as an instance of the `Object::Accessor` class. Please refer to its documentation for details.

Return true if the initialization was successful, and false if it was not.

The `CPANPLUS::Dist::Base` implementation does not alter your object and always returns true.

$bool = $dist->prepare

This runs the preparation step of your distribution. This step is meant to set up the environment so the `create` step can create the actual distribution(file). A `prepare` call in the standard `ExtUtils::MakeMaker` distribution would, for example, run `perl Makefile.PL` to find the dependencies for a distribution. For a `debian` distribution, this is where you would write all the metafiles required for the `dpkg-*` tools.

The `CPANPLUS::Dist::Base` implementation simply calls the underlying distribution class (Typically `CPANPLUS::Dist::MM` or `CPANPLUS::Dist::Build`).

Sets `$dist->status->prepared` to the return value of this function. If you override this method, you should make sure to set this value.

$bool = $dist->create

This runs the creation step of your distribution. This step is meant to follow up on the `prepare` call, that set up your environment so the `create` step can create the actual distribution(file). A `create` call in the standard `ExtUtils::MakeMaker` distribution would, for example, run `make` and `make test` to build and test a distribution. For a `debian` distribution, this is where you would create the actual `.deb` file using `dpkg`.

The `CPANPLUS::Dist::Base` implementation simply calls the underlying distribution class (Typically `CPANPLUS::Dist::MM` or `CPANPLUS::Dist::Build`).

Sets `$dist->status->dist` to the location of the created distribution. If you override this method, you should make sure to set this value.

Sets `$dist->status->created` to the return value of this function. If you override this method, you should make sure to set this value.

$bool = $dist->install

This runs the install step of your distribution. This step is meant to follow up on the `create` call, which prepared a distribution(file) to install. A `create` call in the standard `ExtUtils::MakeMaker` distribution would, for example, run `make install` to copy the distribution files to their final destination. For a `debian` distribution, this is where you would run `dpkg --install` on the created `.deb` file.

The `CPANPLUS::Dist::Base` implementation simply calls the underlying distribution class (Typically `CPANPLUS::Dist::MM` or `CPANPLUS::Dist::Build`).

Sets `$dist->status->installed` to the return value of this function. If you override this method, you should make sure to set this value.

$bool = $dist->uninstall

This runs the uninstall step of your distribution. This step is meant to remove the distribution from the file system. A `uninstall` call in the standard `ExtUtils::MakeMaker` distribution would, for example, run `make uninstall` to remove the distribution files the file system. For a `debian` distribution, this is where you would run `dpkg --uninstall PACKAGE`.

The `CPANPLUS::Dist::Base` implementation simply calls the underlying distribution class (Typically `CPANPLUS::Dist::MM` or `CPANPLUS::Dist::Build`).

Sets `$dist->status->uninstalled` to the return value of this function. If you override this method, you should make sure to set this value.