NAME

File::Path - create or remove directory trees

SYNOPSIS

```
use File::Path;

mkpath(['/foo/bar/baz', 'blurfl/quux'], 1, 0711);
rmtree(['foo/bar/baz', 'blurfl/quux'], 1, 1);
```

DESCRIPTION

The `mkpath` function provides a convenient way to create directories, even if your `mkdir` kernel call won't create more than one level of directory at a time. `mkpath` takes three arguments:

- the name of the path to create, or a reference to a list of paths to create,
- a boolean value, which if TRUE will cause `mkpath` to print the name of each directory as it is created (defaults to FALSE), and
- the numeric mode to use when creating the directories (defaults to 0777), to be modified by the current umask.

It returns a list of all directories (including intermediates, determined using the Unix '/' separator) created.

If a system error prevents a directory from being created, then the `mkpath` function throws a fatal error with `Carp::croak`. This error can be trapped with an `eval` block:

```
eval { mkpath($dir) };
if ($@) {
  print "Couldn't create $dir: $@";
}
```

Similarly, the `rmtree` function provides a convenient way to delete a subtree from the directory structure, much like the Unix command `rm -r`. `rmtree` takes three arguments:

- the root of the subtree to delete, or a reference to a list of roots. All of the files and directories below each root, as well as the roots themselves, will be deleted.
- a boolean value, which if TRUE will cause `rmtree` to print a message each time it examines a file, giving the name of the file, and indicating whether it's using `rmdir` or `unlink` to remove it, or that it's skipping it. (defaults to FALSE)
- a boolean value, which if TRUE will cause `rmtree` to skip any files to which you do not have delete access (if running under VMS) or write access (if running under another OS). This will change in the future when a criterion for 'delete permission' under OSs other than VMS is settled. (defaults to FALSE)

It returns the number of files successfully deleted. Symlinks are simply deleted and not followed.

NOTE: There are race conditions internal to the implementation of `rmtree` making it unsafe to use on directory trees which may be altered or moved while `rmtree` is running, and in particular on any directory trees with any path components or subdirectories potentially writable by untrusted users.

Additionally, if the third parameter is not TRUE and `rmtree` is interrupted, it may leave files and directories with permissions altered to allow deletion (and older versions of this module would even set files and directories to world-read/writable!)

Note also that the occurrence of errors in `rmtree` can be determined *only* by trapping diagnostic messages using `$SIG{__WARN__}`; it is not apparent from the return value.

DIAGNOSTICS

- On Windows, if `mkpath` gives you the warning: **No such file or directory**, this may mean that you've exceeded your filesystem's maximum path length.

AUTHORS

Tim Bunce <*Tim.Bunce@ig.co.uk*> and Charles Bailey <*bailey@newman.upenn.edu*>