NAME

Env - perl module that imports environment variables as scalars or arrays

SYNOPSIS

```
use Env;
use Env qw(PATH HOME TERM);
use Env qw($SHELL @LD_LIBRARY_PATH);
```

DESCRIPTION

Perl maintains environment variables in a special hash named `%ENV`. For when this access method is inconvenient, the Perl module `Env` allows environment variables to be treated as scalar or array variables.

The `Env::import()` function ties environment variables with suitable names to global Perl variables with the same names. By default it ties all existing environment variables (`keys %ENV`) to scalars. If the `import` function receives arguments, it takes them to be a list of variables to tie; it's okay if they don't yet exist. The scalar type prefix '$' is inferred for any element of this list not prefixed by '$' or '@'. Arrays are implemented in terms of `split` and `join`, using `$Config::Config{path_sep}` as the delimiter.

After an environment variable is tied, merely use it like a normal variable. You may access its value

```
@path = split(/:/, $PATH);
print join("\n", @LD_LIBRARY_PATH), "\n";
```

or modify it

```
$PATH .= ":.";
push @LD_LIBRARY_PATH, $dir;
```

however you'd like. Bear in mind, however, that each access to a tied array variable requires splitting the environment variable's string anew.

The code:

```
use Env qw(@PATH);
push @PATH, '.';
```

is equivalent to:

```
use Env qw(PATH);
$PATH .= ":.";
```

except that if `$ENV{PATH}` started out empty, the second approach leaves it with the (odd) value `:.`", but the first approach leaves it with "`.`.

To remove a tied environment variable from the environment, assign it the undefined value

```
undef $PATH;
undef @LD_LIBRARY_PATH;
```

LIMITATIONS

On VMS systems, arrays tied to environment variables are read-only. Attempting to change anything will cause a warning.

AUTHOR

Chip Salzenberg <*chip@fin.uucp*> and Gregor N. Purdy <*gregor@focusresearch.com*>