



Rocky Enterprise Linux 9.2 Manual Pages on command 'Clone.3pm'

C:\>man Clone.3pm

Clone(3pm)

User Contributed Perl Documentation

Clone(3pm)

NAME

Clone - recursively copy Perl datatypes

SYNOPSIS

```
use Clone 'clone';
```

```
my $data = {  
    set => [ 1 .. 50 ],  
    foo => {  
        answer => 42,  
        object => SomeObject->new,  
    },  
};
```

```
my $cloned_data = clone($data);
```

```
$cloned_data->{foo}{answer} = 1;  
print $cloned_data->{foo}{answer}; # '1'  
print $data->{foo}{answer};      # '42'
```

You can also add it to your class:

```
package Foo;
use parent 'Clone';
sub new { bless {}, shift }

package main;

my $obj = Foo->new;
my $copy = $obj->clone;
```

DESCRIPTION

This module provides a "clone()" method which makes recursive copies of nested hash, array, scalar and reference types, including tied variables and objects.

"clone()" takes a scalar argument and duplicates it. To duplicate lists, arrays or hashes, pass them in by reference, e.g.

```
my $copy = clone (\@array);
```

or

```
my %copy = %{ clone (\%hash) };
```

SEE ALSO

Storable's "dclone()" is a flexible solution for cloning variables, albeit slower for average-sized data structures. Simple and naive benchmarks show that Clone is faster for data structures with 3 or fewer levels, while "dclone()" can be faster for structures 4 or more levels deep.

COPYRIGHT

Copyright 2001-2019 Ray Finch. All Rights Reserved.

This module is free software; you can redistribute it and/or modify it under the same terms as Perl itself.

AUTHOR

Ray Finch "<rdf@cpan.org>"

Breno G. de Oliveira "<garu@cpan.org>" and Florian Ragwitz "<rafl@debian.org>"
perform routine maintenance releases since 2012.

perl v5.30.0

2019-11-24

Clone(3pm)