

## NAME

Dumpvalue - provides screen dump of Perl data.

## SYNOPSIS

```
use Dumpvalue;
my $dumper = new Dumpvalue;
$dumper->set(globPrint => 1);
$dumper->dumpValue(\*::);
$dumper->dumpvars('main');
my $dump = $dumper->stringify($some_value);
```

## DESCRIPTION

### Creation

A new dumper is created by a call

```
$d = new Dumpvalue(option1 => value1, option2 => value2)
```

Recognized options:

`arrayDepth`, `hashDepth`

Print only first N elements of arrays and hashes. If false, prints all the elements.

`compactDump`, `veryCompact`

Change style of array and hash dump. If true, short array may be printed on one line.

`globPrint`

Whether to print contents of globs.

`dumpDBFiles`

Dump arrays holding contents of debugged files.

`dumpPackages`

Dump symbol tables of packages.

`dumpReused`

Dump contents of "reused" addresses.

`tick`, `quoteHighBit`, `printUndef`

Change style of string dump. Default value of `tick` is `auto`, one can enable either double-quotish dump, or single-quotish by setting it to `"` or `'`. By default, characters with high bit set are printed as *is*. If `quoteHighBit` is set, they will be quoted.

`usageOnly`

rudimentally per-package memory usage dump. If set, `dumpvars` calculates total size of strings in variables in the package.

`unctrl`

Changes the style of printout of strings. Possible values are `unctrl` and `quote`.

`subdump`

Whether to try to find the subroutine name given the reference.

`bareStringify`

Whether to write the non-overloaded form of the stringify-overloaded objects.

**quoteHighBit**

Whether to print chars with high bit set in binary or "as is".

**stopDbSignal**

Whether to abort printing if debugger signal flag is raised.

Later in the life of the object the methods may be queried with `get()` method and `set()` method (which accept multiple arguments).

**Methods****dumpValue**

```
$dumper->dumpValue($value);  
$dumper->dumpValue([$value1, $value2]);
```

Prints a dump to the currently selected filehandle.

**dumpValues**

```
$dumper->dumpValues($value1, $value2);
```

Same as `$dumper->dumpValue([$value1, $value2]); >`.

**stringify**

```
my $dump = $dumper->stringify($value [, $noticks] );
```

Returns the dump of a single scalar without printing. If the second argument is true, the return value does not contain enclosing ticks. Does not handle data structures.

**dumpvars**

```
$dumper->dumpvars('my_package');  
$dumper->dumpvars('my_package', 'foo', '~bar$', '!.....');
```

The optional arguments are considered as literal strings unless they start with `~` or `!`, in which case they are interpreted as regular expressions (possibly negated).

The second example prints entries with names `foo`, and also entries with names which ends on `bar`, or are shorter than 5 chars.

**set\_quote**

```
$d->set_quote('');
```

Sets `tick` and `unctrl` options to suitable values for printout with the given quote char. Possible values are `auto`, `'` and `"`.

**set\_unctrl**

```
$d->set_unctrl('');
```

Sets `unctrl` option with checking for an invalid argument. Possible values are `unctrl` and `quote`.

**compactDump**

```
$d->compactDump(1);
```

Sets `compactDump` option. If the value is 1, sets to a reasonable big number.

**veryCompact**

```
$d->veryCompact(1);
```

Sets `compactDump` and `veryCompact` options simultaneously.

set

```
$d->set(option1 => value1, option2 => value2);
```

get

```
@values = $d->get('option1', 'option2');
```