

#### NAME

Hash::Util - A selection of general-utility hash subroutines

#### **SYNOPSIS**

# **DESCRIPTION**

Hash::Util contains special functions for manipulating hashes that don't really warrant a keyword.

By default Hash::Util does not export anything.

#### **Restricted hashes**

5.8.0 introduces the ability to restrict a hash to a certain set of keys. No keys outside of this set can be added. It also introduces the ability to lock an individual key so it cannot be deleted and the value cannot be changed.

This is intended to largely replace the deprecated pseudo-hashes.

```
lock_keys
unlock_keys
lock_keys(%hash);
lock_keys(%hash, @keys);
```

Restricts the given %hash's set of keys to @keys. If @keys is not given it restricts it to its current keyset. No more keys can be added. delete() and exists() will still work, but will not alter the set of allowed keys. **Note**: the current implementation prevents the hash from being bless()ed while it is in a locked state. Any attempt to do so will raise an exception. Of course you can still bless() the hash before you call lock\_keys() so this shouldn't be a problem.

```
unlock_keys(%hash);
```

Removes the restriction on the %hash's keyset.



Locks and unlocks an individual key of a hash. The value of a locked key cannot be changed. %hash must have already been locked for this to have useful effect.

#### lock hash

#### unlock hash

```
lock hash(%hash);
```

lock\_hash() locks an entire hash, making all keys and values readonly. No value can be changed, no keys can be added or deleted.

```
unlock_hash(%hash);
```

unlock\_hash() does the opposite of lock\_hash(). All keys and values are made read/write. All values can be changed and keys can be added and deleted.

### hash\_seed

```
my $hash_seed = hash_seed();
```

hash\_seed() returns the seed number used to randomise hash ordering. Zero means the "traditional" random hash ordering, non-zero means the new even more random hash ordering introduced in Perl 5.8.1.

**Note that the hash seed is sensitive information**: by knowing it one can craft a denial-of-service attack against Perl code, even remotely, see "Algorithmic Complexity Attacks" in perlsec for more information. **Do not disclose the hash seed** to people who don't need to know it. See also "PERL\_HASH\_SEED\_DEBUG" in perlrun.

# **CAVEATS**

Note that the trapping of the restricted operations is not atomic: for example

```
eval { %hash = (illegal_key => 1) }
```

leaves the %hash empty rather than with its original contents.

# **AUTHOR**

Michael G Schwern <schwern@pobox.com> on top of code by Nick Ing-Simmons and Jeffrey Friedl.

# **SEE ALSO**

Scalar::Util, List::Util, Hash::Util, and "Algorithmic Complexity Attacks" in perlsec.