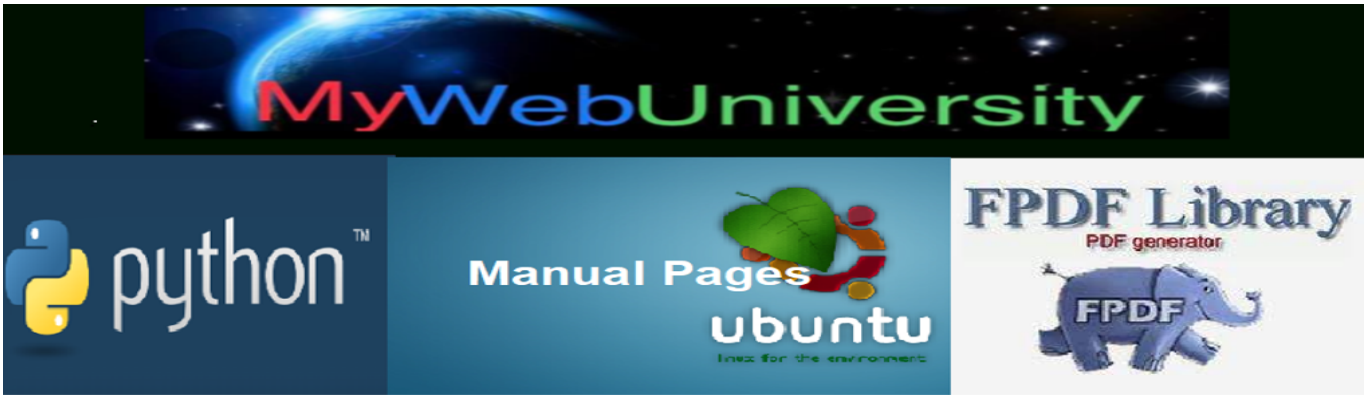

Rocky Enterprise Linux 9.2 Manual Pages on command 'Crypt::Digest.3pm'

$ man Crypt::Digest.3pm

Crypt::Digest(3pm) User Contributed Perl Documentation Crypt::Digest(3pm)

NAME

Crypt::Digest - Generic interface to hash/digest functions

SYNOPSIS

```
### Functional interface:
use Crypt::Digest qw( digest_data digest_data_hex digest_data_b64 digest_data_b64u
                      digest_file digest_file_hex digest_file_b64 digest_file_b64u );

# calculate digest from string/buffer
$digest_raw  = digest_data('SHA1', 'data string');
$digest_hex  = digest_data_hex('SHA1', 'data string');
$digest_b64  = digest_data_b64('SHA1', 'data string');
$digest_b64u = digest_data_b64u('SHA1', 'data string');
# calculate digest from file
$digest_raw  = digest_file('SHA1', 'filename.dat');
$digest_hex  = digest_file_hex('SHA1', 'filename.dat');
$digest_b64  = digest_file_b64('SHA1', 'filename.dat');
$digest_b64u = digest_file_b64u('SHA1', 'filename.dat');
# calculate digest from filehandle
$digest_raw  = digest_file('SHA1', *FILEHANDLE);
$digest_hex  = digest_file_hex('SHA1', *FILEHANDLE);
```

```
$digest_b64  = digest_file_b64('SHA1', *FILEHANDLE);
$digest_b64u = digest_file_b64u('SHA1', *FILEHANDLE);

### OO interface:
use Crypt::Digest;

$d = Crypt::Digest->new('SHA1');
$d->add('any data');
$d->addfile('filename.dat');
$d->addfile(*FILEHANDLE);
$result_raw  = $d->digest;     # raw bytes
$result_hex  = $d->hexdigest;  # hexadecimal form
$result_b64  = $d->b64digest;  # Base64 form
$result_b64u = $d->b64udigest; # Base64 URL Safe form
```

DESCRIPTION

Provides an interface to various hash/digest algorithms.

EXPORT

Nothing is exported by default.

You can export selected functions:

```
use Crypt::Digest qw( digest_data digest_data_hex digest_data_b64 digest_data_b64u
                      digest_file digest_file_hex digest_file_b64 digest_file_b64u );
```

Or all of them at once:

```
use Crypt::Digest ':all';
```

FUNCTIONS

Please note that all functions take as its first argument the algorithm name, supported values are:

'CHAES', 'MD2', 'MD4', 'MD5', 'RIPEMD128', 'RIPEMD160',

'RIPEMD256', 'RIPEMD320', 'SHA1', 'SHA224', 'SHA256',

'SHA384', 'SHA512', 'SHA512_224', 'SHA512_256', 'Tiger192', 'Whirlpool',

'SHA3_224', 'SHA3_256', 'SHA3_384', 'SHA3_512',

'BLAKE2b_160', 'BLAKE2b_256', 'BLAKE2b_384', 'BLAKE2b_512',

'BLAKE2s_128', 'BLAKE2s_160', 'BLAKE2s_224', 'BLAKE2s_256'

(simply any <NAME> for which there is Crypt::Digest::<NAME> module)

digest_data

Logically joins all arguments into a single string, and returns its SHA1 digest encoded as a binary string.

```
$digest_raw = digest_data('SHA1', 'data string');
#or
$digest_raw = digest_data('SHA1', 'any data', 'more data', 'even more data');
```

digest_data_hex

Logically joins all arguments into a single string, and returns its SHA1 digest encoded as a hexadecimal string.

```
$digest_hex = digest_data_hex('SHA1', 'data string');
#or
$digest_hex = digest_data_hex('SHA1', 'any data', 'more data', 'even more data');
```

digest_data_b64

Logically joins all arguments into a single string, and returns its SHA1 digest encoded as a Base64 string, with trailing '=' padding.

```
$digest_b64 = digest_data_b64('SHA1', 'data string');
#or
$digest_b64 = digest_data_b64('SHA1', 'any data', 'more data', 'even more data');
```

digest_data_b64u

Logically joins all arguments into a single string, and returns its SHA1 digest encoded as a Base64 URL Safe string (see RFC 4648 section 5).

```
$digest_b64url = digest_data_b64u('SHA1', 'data string');
#or
$digest_b64url = digest_data_b64u('SHA1', 'any data', 'more data', 'even more data');
```

digest_file

Reads file (defined by filename or filehandle) content, and returns its digest encoded as a binary string.

```
$digest_raw = digest_file('SHA1', 'filename.dat');
#or
$digest_raw = digest_file('SHA1', *FILEHANDLE);
```

digest_file_hex

Reads file (defined by filename or filehandle) content, and returns its digest encoded as a hexadecimal string.

```
$digest_hex = digest_file_hex('SHA1', 'filename.dat');
#or
$digest_hex = digest_file_hex('SHA1', *FILEHANDLE);
```

BEWARE: You have to make sure that the filehandle is in binary mode before you pass it as argument to the addfile() method.

digest_file_b64

Reads file (defined by filename or filehandle) content, and returns its digest encoded as a Base64 string, with trailing '=' padding.

```
$digest_b64 = digest_file_b64('SHA1', 'filename.dat');
```

```
#or
$digest_b64 = digest_file_b64('SHA1', *FILEHANDLE);
```

digest_file_b64u

Reads file (defined by filename or filehandle) content, and returns its digest encoded as a Base64 URL Safe string (see RFC 4648 section 5).

```
$digest_b64url = digest_file_b64u('SHA1', 'filename.dat');
#or
$digest_b64url = digest_file_b64u('SHA1', *FILEHANDLE);
```

METHODS

new

Constructor, returns a reference to the digest object.

```
$d = Crypt::Digest->new($name);
# $name could be: 'CHAES', 'MD2', 'MD4', 'MD5', 'RIPEMD128', 'RIPEMD160',
#                 'RIPEMD256', 'RIPEMD320', 'SHA1', 'SHA224', 'SHA256', 'SHA384',
#                 'SHA512', 'SHA512_224', 'SHA512_256', 'Tiger192', 'Whirlpool'
#
# simply any <FUNCNAME> for which there is Crypt::Digest::<FUNCNAME> module
```

clone

Creates a copy of the digest object state and returns a reference to the copy.

```
$d->clone();
```

reset

Reinitialize the digest object state and returns a reference to the digest object.

```
$d->reset();
```

add

All arguments are appended to the message we calculate digest for. The return value is the digest object itself.

```
$d->add('any data');
#or
$d->add('any data', 'more data', 'even more data');
```

Note that all the following cases are equivalent:

```
# case 1
$d->add('aa', 'bb', 'cc');

# case 2
$d->add('aa');
$d->add('bb');
$d->add('cc');

# case 3
$d->add('aabbcc');

# case 4
$d->add('aa')->add('bb')->add('cc');
```

addfile

The content of the file (or filehandle) is appended to the message we calculate digest for. The return value is the digest object itself.

```
$d->addfile('filename.dat');
#or
$d->addfile(*FILEHANDLE);
```

BEWARE: You have to make sure that the filehandle is in binary mode before you pass it as argument to the addfile() method.

add_bits

This method is available mostly for compatibility with other Digest::SOMETHING modules on CPAN, you are very unlikely to need it. The return value is the digest object itself.

```
$d->add_bits($bit_string);   # e.g. $d->add_bits("111100001010");
#or
$d->add_bits($data, $nbits); # e.g. $d->add_bits("\xF0\xA0", 16);
```

BEWARE: It is not possible to add bits that are not a multiple of 8.

hashsize

Returns the length of calculated digest in bytes (e.g. 32 for SHA-256).

```
$d->hashsize;
#or
Crypt::Digest->hashsize('SHA1');
#or
Crypt::Digest::hashsize('SHA1');
```

digest

Returns the binary digest (raw bytes).

```
$result_raw = $d->digest();
```

hexdigest

Returns the digest encoded as a hexadecimal string.

```
$result_hex = $d->hexdigest();
```

b64digest

Returns the digest encoded as a Base64 string, with trailing '=' padding (BEWARE: this padding style might differ from other Digest::<SOMETHING> modules on CPAN).

```
$result_b64 = $d->b64digest();
```

b64udigest

Returns the digest encoded as a Base64 URL Safe string (see RFC 4648 section 5).

```
$result_b64url = $d->b64udigest();
```

SEE ALSO

- CryptX
- Crypt::Digest tries to be compatible with Digest interface.
- Check subclasses like Crypt::Digest::SHA1, Crypt::Digest::MD5, ...