## NAME

EVP_MD_meth_new, EVP_MD_meth_dup, EVP_MD_meth_free, EVP_MD_meth_set_input_blocksize, EVP_MD_meth_set_result_size, EVP_MD_meth_set_app_datasize, EVP_MD_meth_set_flags, EVP_MD_meth_set_init, EVP_MD_meth_set_update, EVP_MD_meth_set_final, EVP_MD_meth_set_copy, EVP_MD_meth_set_cleanup, EVP_MD_meth_set_ctrl, EVP_MD_meth_get_input_blocksize, EVP_MD_meth_get_result_size, EVP_MD_meth_get_app_datasize, EVP_MD_meth_get_flags, EVP_MD_meth_get_init, EVP_MD_meth_get_update, EVP_MD_meth_get_final, EVP_MD_meth_get_copy, EVP_MD_meth_get_cleanup, EVP_MD_meth_get_ctrl - Routines to build up legacy EVP_MD methods

## SYNOPSIS

```
#include <openssl/evp.h>
```

The following functions have been deprecated since OpenSSL 3.0, and can be hidden entirely by defining **OPENSSL_API_COMPAT** with a suitable version value, see **openssl_user_macros**(7):

```
EVP_MD *EVP_MD_meth_new(int md_type, int pkey_type);
void EVP_MD_meth_free(EVP_MD *md);
EVP_MD *EVP_MD_meth_dup(const EVP_MD *md);

int EVP_MD_meth_set_input_blocksize(EVP_MD *md, int blocksize);
int EVP_MD_meth_set_result_size(EVP_MD *md, int resultsize);
int EVP_MD_meth_set_app_datasize(EVP_MD *md, int datasize);
int EVP_MD_meth_set_flags(EVP_MD *md, unsigned long flags);
int EVP_MD_meth_set_init(EVP_MD *md, int (*init)(EVP_MD_CTX *ctx));
int EVP_MD_meth_set_update(EVP_MD *md, int (*update)(EVP_MD_CTX *ctx,
                                                     const void *data,
                                                     size_t count));
int EVP_MD_meth_set_final(EVP_MD *md, int (*final)(EVP_MD_CTX *ctx,
                                                   unsigned char *md));
int EVP_MD_meth_set_copy(EVP_MD *md, int (*copy)(EVP_MD_CTX *to,
                                                 const EVP_MD_CTX *from));
int EVP_MD_meth_set_cleanup(EVP_MD *md, int (*cleanup)(EVP_MD_CTX *ctx));
int EVP_MD_meth_set_ctrl(EVP_MD *md, int (*ctrl)(EVP_MD_CTX *ctx, int cmd,
                                                 int p1, void *p2));

int EVP_MD_meth_get_input_blocksize(const EVP_MD *md);
int EVP_MD_meth_get_result_size(const EVP_MD *md);
int EVP_MD_meth_get_app_datasize(const EVP_MD *md);
```

```
unsigned long EVP_MD_meth_get_flags(const EVP_MD *md);
int (*EVP_MD_meth_get_init(const EVP_MD *md))(EVP_MD_CTX *ctx);
int (*EVP_MD_meth_get_update(const EVP_MD *md))(EVP_MD_CTX *ctx,
                                                const void *data,
                                                size_t count);
int (*EVP_MD_meth_get_final(const EVP_MD *md))(EVP_MD_CTX *ctx,
                                               unsigned char *md);
int (*EVP_MD_meth_get_copy(const EVP_MD *md))(EVP_MD_CTX *to,
                                              const EVP_MD_CTX *from);
int (*EVP_MD_meth_get_cleanup(const EVP_MD *md))(EVP_MD_CTX *ctx);
int (*EVP_MD_meth_get_ctrl(const EVP_MD *md))(EVP_MD_CTX *ctx, int cmd,
                                              int p1, void *p2);
```

## DESCRIPTION

All of the functions described on this page are deprecated. Applications should instead use the OSSL_PROVIDER APIs.

The **EVP_MD** type is a structure for digest method implementation. It can also have associated public/private key signing and verifying routines.

**EVP_MD_meth_new()** creates a new **EVP_MD** structure. These **EVP_MD** structures are reference counted.

**EVP_MD_meth_dup()** creates a copy of **md**.

**EVP_MD_meth_free()** decrements the reference count for the **EVP_MD** structure. If the reference count drops to 0 then the structure is freed.

**EVP_MD_meth_set_input_blocksize()** sets the internal input block size for the method **md** to **blocksize** bytes.

**EVP_MD_meth_set_result_size()** sets the size of the result that the digest method in **md** is expected to produce to **resultsize** bytes.

The digest method may have its own private data, which OpenSSL will allocate for it. **EVP_MD_meth_set_app_datasize()** should be used to set the size for it to **datasize**.

**EVP_MD_meth_set_flags()** sets the flags to describe optional behaviours in the particular **md**. Several flags can be or'd together. The available flags are:

EVP_MD_FLAG_ONESHOT
: This digest method can only handle one block of input.

EVP_MD_FLAG_XOF
: This digest method is an extensible-output function (XOF) and supports the **EVP_MD_CTRL_XOF_LEN** control.

EVP_MD_FLAG_DIGALGID_NULL
: When setting up a DigestAlgorithmIdentifier, this flag will have the parameter set to NULL by default. Use this for PKCS#1. *Note: if combined with EVP_MD_FLAG_DIGALGID_ABSENT, the latter will override.*

EVP_MD_FLAG_DIGALGID_ABSENT
: When setting up a DigestAlgorithmIdentifier, this flag will have the parameter be left absent by default. *Note: if combined with EVP_MD_FLAG_DIGALGID_NULL, the latter will be overridden.*

EVP_MD_FLAG_DIGALGID_CUSTOM
: Custom DigestAlgorithmIdentifier handling via ctrl, with **EVP_MD_FLAG_DIGALGID_ABSENT** as default. *Note: if combined with EVP_MD_FLAG_DIGALGID_NULL, the latter will be overridden.* Currently unused.

EVP_MD_FLAG_FIPS
: This digest method is suitable for use in FIPS mode. Currently unused.

**EVP_MD_meth_set_init()** sets the digest init function for **md**. The digest init function is called by **EVP_Digest()**, **EVP_DigestInit()**, **EVP_DigestInit_ex()**, EVP_SignInit, **EVP_SignInit_ex()**, **EVP_VerifyInit()** and **EVP_VerifyInit_ex()**.

**EVP_MD_meth_set_update()** sets the digest update function for **md**. The digest update function is called by **EVP_Digest()**, **EVP_DigestUpdate()** and **EVP_SignUpdate()**.

**EVP_MD_meth_set_final()** sets the digest final function for **md**. The digest final function is called by **EVP_Digest()**, **EVP_DigestFinal()**, **EVP_DigestFinal_ex()**, **EVP_SignFinal()** and **EVP_VerifyFinal()**.

**EVP_MD_meth_set_copy()** sets the function for **md** to do extra computations after the method's private data structure has been copied from one **EVP_MD_CTX** to another. If all that's needed is to copy the data, there is no need for this copy function. Note that the copy function is passed two **EVP_MD_CTX ***, the private data structure is then available with **EVP_MD_CTX_get0_md_data()**. This copy function is called by **EVP_MD_CTX_copy()** and **EVP_MD_CTX_copy_ex()**.

**EVP_MD_meth_set_cleanup()** sets the function for **md** to do extra cleanup before the method's private data structure is cleaned out and freed. Note that the cleanup function is passed a **EVP_MD_CTX ***, the private data structure is then available with **EVP_MD_CTX_get0_md_data()**. This cleanup function is called by **EVP_MD_CTX_reset()** and **EVP_MD_CTX_free()**.

**EVP_MD_meth_set_ctrl()** sets the control function for **md**. See **EVP_MD_CTX_ctrl**(3) for the available controls.

**EVP_MD_meth_get_input_blocksize()**, **EVP_MD_meth_get_result_size()**, **EVP_MD_meth_get_app_datasize()**, **EVP_MD_meth_get_flags()**, **EVP_MD_meth_get_init()**, **EVP_MD_meth_get_update()**, **EVP_MD_meth_get_final()**, **EVP_MD_meth_get_copy()**, **EVP_MD_meth_get_cleanup()** and **EVP_MD_meth_get_ctrl()** are all used to retrieve the method data given with the EVP_MD_meth_set_*() functions above.

## RETURN VALUES

**EVP_MD_meth_new()** and **EVP_MD_meth_dup()** return a pointer to a newly created **EVP_MD**, or NULL on failure. All EVP_MD_meth_set_*() functions return 1. **EVP_MD_get_input_blocksize()**, **EVP_MD_meth_get_result_size()**, **EVP_MD_meth_get_app_datasize()** and **EVP_MD_meth_get_flags()** return the indicated sizes or flags. All other EVP_CIPHER_meth_get_*() functions return pointers to their respective **md** function.

## SEE ALSO

**EVP_DigestInit**(3), **EVP_SignInit**(3), **EVP_VerifyInit**(3)

## HISTORY

All of these functions were deprecated in OpenSSL 3.0.

The **EVP_MD** structure was openly available in OpenSSL before version 1.1. The functions described here were added in OpenSSL 1.1. The **EVP_MD** structure created with these functions became reference counted in OpenSSL 3.0.

## COPYRIGHT