

**NAME**

EVP\_KEM\_fetch, EVP\_KEM\_free, EVP\_KEM\_up\_ref, EVP\_KEM\_get0\_name, EVP\_KEM\_is\_a, EVP\_KEM\_get0\_provider, EVP\_KEM\_do\_all\_provided, EVP\_KEM\_names\_do\_all, EVP\_KEM\_get0\_description, EVP\_KEM\_gettable\_ctx\_params, EVP\_KEM\_settable\_ctx\_params - Functions to manage EVP\_KEM algorithm objects

**SYNOPSIS**

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

```
EVP_KEM *EVP_KEM_fetch(OSSL_LIB_CTX *ctx, const char *algorithm,
                       const char *properties);
void EVP_KEM_free(EVP_KEM *kem);
int EVP_KEM_up_ref(EVP_KEM *kem);
const char *EVP_KEM_get0_name(const EVP_KEM *kem);
int EVP_KEM_is_a(const EVP_KEM *kem, const char *name);
OSSL_PROVIDER *EVP_KEM_get0_provider(const EVP_KEM *kem);
void EVP_KEM_do_all_provided(OSSL_LIB_CTX *libctx,
                             void (*fn)(EVP_KEM *kem, void *arg), void *arg);
int EVP_KEM_names_do_all(const EVP_KEM *kem,
                         void (*fn)(const char *name, void *data), void *data);
const char *EVP_KEM_get0_description(const EVP_KEM *kem);
const OSSL_PARAM *EVP_KEM_gettable_ctx_params(const EVP_KEM *kem);
const OSSL_PARAM *EVP_KEM_settable_ctx_params(const EVP_KEM *kem);
```

**DESCRIPTION**

**EVP\_KEM\_fetch()** fetches the implementation for the given **algorithm** from any provider offering it, within the criteria given by the **properties** and in the scope of the given library context **ctx** (see **OSSL\_LIB\_CTX(3)**). The algorithm will be one offering functions for performing asymmetric kem related tasks such as key encapsulation and decapsulation. See "ALGORITHM FETCHING" in **crypto(7)** for further information.

The returned value must eventually be freed with **EVP\_KEM\_free()**.

**EVP\_KEM\_free()** decrements the reference count for the **EVP\_KEM** structure. Typically this structure will have been obtained from an earlier call to **EVP\_KEM\_fetch()**. If the reference count drops to 0 then the structure is freed.

**EVP\_KEM\_up\_ref()** increments the reference count for an **EVP\_KEM** structure.

**EVP\_KEM\_is\_a()** returns 1 if *kem* is an implementation of an algorithm that's identifiable with *name*,

otherwise 0.

**EVP\_KEM\_get0\_provider()** returns the provider that *kem* was fetched from.

**EVP\_KEM\_do\_all\_provided()** traverses all EVP\_KEMs implemented by all activated providers in the given library context *libctx*, and for each of the implementations, calls the given function *fn* with the implementation method and the given *arg* as argument.

**EVP\_KEM\_get0\_name()** returns the algorithm name from the provided implementation for the given *kem*. Note that the *kem* may have multiple synonyms associated with it. In this case the first name from the algorithm definition is returned. Ownership of the returned string is retained by the *kem* object and should not be freed by the caller.

**EVP\_KEM\_names\_do\_all()** traverses all names for *kem*, and calls *fn* with each name and *data*.

**EVP\_KEM\_get0\_description()** returns a description of the *kem*, meant for display and human consumption. The description is at the discretion of the *kem* implementation.

**EVP\_KEM\_gettable\_ctx\_params()** and **EVP\_KEM\_settable\_ctx\_params()** return a constant **OSSL\_PARAM(3)** array that describes the names and types of key parameters that can be retrieved or set by a key encapsulation algorithm using **EVP\_PKEY\_CTX\_get\_params(3)** and **EVP\_PKEY\_CTX\_set\_params(3)**.

## RETURN VALUES

**EVP\_KEM\_fetch()** returns a pointer to an **EVP\_KEM** for success or **NULL** for failure.

**EVP\_KEM\_up\_ref()** returns 1 for success or 0 otherwise.

**EVP\_KEM\_names\_do\_all()** returns 1 if the callback was called for all names. A return value of 0 means that the callback was not called for any names.

**EVP\_KEM\_gettable\_ctx\_params()** and **EVP\_KEM\_settable\_ctx\_params()** return a constant **OSSL\_PARAM(3)** array or **NULL** on error.

## SEE ALSO

"ALGORITHM FETCHING" in **crypto(7)**, **OSSL\_PROVIDER(3)**

## HISTORY

The functions described here were added in OpenSSL 3.0.

**COPYRIGHT**

Copyright 2020-2021 The OpenSSL Project Authors. All Rights Reserved.

Licensed under the Apache License 2.0 (the "License"). You may not use this file except in compliance with the License. You can obtain a copy in the file LICENSE in the source distribution or at <https://www.openssl.org/source/license.html>.