NAME

EC_KEY_new, EC_KEY_get_flags, EC_KEY_set_flags, EC_KEY_clear_flags, EC_KEY_new_by_curve_name, EC_KEY_free, EC_KEY_copy, EC_KEY_dup, EC_KEY_up_ref, EC_KEY_get0_group, EC_KEY_set_group, EC_KEY_get0_private_key, EC_KEY_set_private_key, EC_KEY_get0_public_key, EC_KEY_set_public_key, EC_KEY_get_enc_flags, EC_KEY_set_enc_flags, EC_KEY_get_conv_form, EC_KEY_set_conv_form, EC_KEY_get_key_method_data, EC_KEY_insert_key_method_data, EC_KEY_set_asn1_flag, EC_KEY_precompute_mult, EC_KEY_generate_key, EC_KEY_check_key, EC_KEY_set_public_key_affine_coordinates - Functions for creating, destroying and manipulating EC_KEY objects.

SYNOPSIS

```
#include <openssl/ec.h>
#include <openssl/bn.h>

EC_KEY *EC_KEY_new(void);
int EC_KEY_get_flags(const EC_KEY *key);
void EC_KEY_set_flags(EC_KEY *key, int flags);
void EC_KEY_clear_flags(EC_KEY *key, int flags);
EC_KEY *EC_KEY_new_by_curve_name(int nid);
void EC_KEY_free(EC_KEY *key);
EC_KEY *EC_KEY_copy(EC_KEY *dst, const EC_KEY *src);
EC_KEY *EC_KEY_dup(const EC_KEY *src);
int EC_KEY_up_ref(EC_KEY *key);
const EC_GROUP *EC_KEY_get0_group(const EC_KEY *key);
int EC_KEY_set_group(EC_KEY *key, const EC_GROUP *group);
const BIGNUM *EC_KEY_get0_private_key(const EC_KEY *key);
int EC_KEY_set_private_key(EC_KEY *key, const BIGNUM *prv);
const EC_POINT *EC_KEY_get0_public_key(const EC_KEY *key);
int EC_KEY_set_public_key(EC_KEY *key, const EC_POINT *pub);
point_conversion_form_t EC_KEY_get_conv_form(const EC_KEY *key);
void EC_KEY_set_conv_form(EC_KEY *eckey, point_conversion_form_t cform);
void *EC_KEY_get_key_method_data(EC_KEY *key,
void *(*dup_func)(void *), void (*free_func)(void *), void (*clear_free_func)(void *));
void EC_KEY_insert_key_method_data(EC_KEY *key, void *data,
void *(*dup_func)(void *), void (*free_func)(void *), void (*clear_free_func)(void *));
void EC_KEY_set_asn1_flag(EC_KEY *eckey, int asn1_flag);
int EC_KEY_precompute_mult(EC_KEY *key, BN_CTX *ctx);
int EC_KEY_generate_key(EC_KEY *key);
int EC_KEY_check_key(const EC_KEY *key);
int EC_KEY_set_public_key_affine_coordinates(EC_KEY *key, BIGNUM *x, BIGNUM *y);
```

DESCRIPTION

An EC_KEY represents a public key and (optionaly) an associated private key. A new EC_KEY (with no associated curve) can be constructed by calling EC_KEY_new. The reference count for the newly created EC_KEY is initially set to 1. A curve can be associated with the EC_KEY by calling EC_KEY_set_group.

Alternatively a new EC_KEY can be constructed by calling EC_KEY_new_by_curve_name and supplying the nid of the associated curve. Refer to *EC_GROUP_new(3)* for a description of curve names. This function simply wraps calls to EC_KEY_new and EC_GROUP_new_by_curve_name.

Calling EC_KEY_free decrements the reference count for the EC_KEY object, and if it has dropped to zero then frees the memory associated with it.

EC_KEY_copy copies the contents of the EC_KEY in **src** into **dest**.

EC_KEY_dup creates a new EC_KEY object and copies **ec_key** into it.

EC_KEY_up_ref increments the reference count associated with the EC_KEY object.

EC_KEY_generate_key generates a new public and private key for the supplied **eckey** object. **eckey** must have an EC_GROUP object associated with it before calling this function. The private key is a random integer (0 < priv_key < order, where order is the order of the EC_GROUP object). The public key is an EC_POINT on the curve calculated by multiplying the generator for the curve by the private key.

EC_KEY_check_key performs various sanity checks on the EC_KEY object to confirm that it is valid.

EC_KEY_set_public_key_affine_coordinates sets the public key for **key** based on its affine co-ordinates, i.e. it constructs an EC_POINT object based on the supplied **x** and **y** values and sets the public key to be this EC_POINT. It will also performs certain sanity checks on the key to confirm that it is valid.

The functions EC_KEY_get0_group, EC_KEY_set_group, EC_KEY_get0_private_key, EC_KEY_set_private_key, EC_KEY_get0_public_key, and EC_KEY_set_public_key get and set the EC_GROUP object, the private key and the EC_POINT public key for the **key** respectively.

The functions EC_KEY_get_conv_form and EC_KEY_set_conv_form get and set the point_conversion_form for the **key**. For a description of point_conversion_forms please refer to *EC_POINT_new(3)*.

EC_KEY_insert_key_method_data and EC_KEY_get_key_method_data enable the caller to associate arbitary additional data specific to the elliptic curve scheme being used with the EC_KEY object. This data is treated as a ‘‘black box’’ by the ec library. The data to be stored by EC_KEY_insert_key_method_data is provided in the **data** parameter, which must have have associated functions for duplicating, freeing and ‘‘clear_freeing’’ the data item. If a subsequent EC_KEY_get_key_method_data call is issued, the functions for duplicating, freeing and ‘‘clear_freeing’’ the data item must be provided again, and they must be the same as they were when the data item was inserted.

EC_KEY_set_flags sets the flags in the **flags** parameter on the EC_KEY object. Any flags that are already set are left set. The currently defined standard flags are EC_FLAG_NON_FIPS_ALLOW and EC_FLAG_FIPS_CHECKED. In addition there is the flag EC_FLAG_COFACTOR_ECDH which is specific to ECDH and is defined in ecdh.h. EC_KEY_get_flags returns the current flags that are set for this EC_KEY. EC_KEY_clear_flags clears the flags indicated by the **flags** parameter. All other flags are left in their existing state.

EC_KEY_set_asn1_flag sets the asn1_flag on the underlying EC_GROUP object (if set). Refer to *EC_GROUP_copy(3)* for further information on the asn1_flag.

EC_KEY_precompute_mult stores multiples of the underlying EC_GROUP generator for faster point multiplication. See also *EC_POINT_add(3)*.

RETURN VALUES

EC_KEY_new, EC_KEY_new_by_curve_name and EC_KEY_dup return a pointer to the newly created EC_KEY object, or NULL on error.

EC_KEY_get_flags returns the flags associated with the EC_KEY object as an integer.

EC_KEY_copy returns a pointer to the destination key, or NULL on error.

EC_KEY_up_ref, EC_KEY_set_group, EC_KEY_set_private_key, EC_KEY_set_public_key, EC_KEY_precompute_mult, EC_KEY_generate_key, EC_KEY_check_key and EC_KEY_set_public_key_affine_coordinates return 1 on success or 0 on error.

EC_KEY_get0_group returns the EC_GROUP associated with the EC_KEY.

EC_KEY_get0_private_key returns the private key associated with the EC_KEY.

EC_KEY_get_conv_form return the point_conversion_form for the EC_KEY.

SEE ALSO

crypto(3), *ec(3)*, *EC_GROUP_new(3)*, *EC_GROUP_copy(3)*, *EC_POINT_new(3)*, *EC_POINT_add(3)*, *EC_GFp_simple_method(3)*, *d2i_ECPKParameters(3)*