EVP_PKEY (OSSL/include/crypto/evp.h)

```
struct evp_pkey_st {
    int type;
    int save_type;
    CRYPTO_REF_COUNT references;
    const EVP_PKEY_ASN1_METHOD *ameth;
    ENGINE *engine;
    ENGINE *pmeth_engine; /* If not NULL public key ENGINE to use */
    union {
 <>     void *ptr;
        struct rsa_st *rsa;     /* RSA */
        struct dsa_st *dsa;     /* DSA */
        struct dh_st *dh;       /* DH */
        struct ec_key_st *ec;   /* ECC */
        ECX_KEY *ecx;           /* X25519, X448, Ed25519, Ed448 */
    } pkey;
    int save_parameters;
    STACK_OF(X509_ATTRIBUTE) *attributes; /* [ 0 ] */
    CRYPTO_RWLOCK *lock;
} /* EVP_PKEY */ ;
```

EVP_PKEY_CTX (OSSL/include/crypto/evp.h)

```
struct evp_pkey_ctx_st {
    /* Method associated with this operation */
    const EVP_PKEY_METHOD *pmeth;
    /* Engine that implements this method or NULL if builtin */
    ENGINE *engine;
    /* Key: may be NULL */
 <> EVP_PKEY *pkey;
    /* Peer key for key agreement, may be NULL */
    EVP_PKEY *peerkey;
    /* Actual operation */
 <> int operation;
    /* Algorithm specific data */
 <> void *data;
    /* Application specific data */
    void *app_data;
    /* Keygen callback */
    EVP_PKEY_gen_cb *pkey_gencb;
    /* implementation specific keygen data */
    int *keygen_info;
    int keygen_info_count;
} /* EVP_PKEY_CTX */ ;
```

EVP_MD (OSSL/include/crypto/evp.h)

```
struct evp_md_st {
    int type;
    int pkey_type;
    int md_size;
    unsigned long flags;
    int (*init) (EVP_MD_CTX *ctx);
    int (*update) (EVP_MD_CTX *ctx, const void *data, size_t count);
    int (*final) (EVP_MD_CTX *ctx, unsigned char *md);
    int (*copy) (EVP_MD_CTX *to, const EVP_MD_CTX *from);
    int (*cleanup) (EVP_MD_CTX *ctx);
    int block_size;
    int ctx_size;               /* how big does the ctx->md_data need to be */
    /* control function */
    int (*md_ctrl) (EVP_MD_CTX *ctx, int cmd, int p1, void *p2);
} /* EVP_MD */ ;
```

EVP_MD_CTX (OSSL/crypto/evp/evp_local.c)

```
struct evp_md_ctx_st {
 <> const EVP_MD *digest;
    ENGINE *engine;             /* functional reference if 'digest' is
                                 * ENGINE-provided */
    unsigned long flags;
    void *md_data;
    /* Public key context for sign/verify */
 <> EVP_PKEY_CTX *pctx;
    /* Update function: usually copied from EVP_MD */
    int (*update) (EVP_MD_CTX *ctx, const void *data, size_t count);
} /* EVP_MD_CTX */ ;
```