

PRAGMA

PRAGMA action

Function

This action enables/disables a transfer IN OUT parameters of the procedure by a **reference**.

Declaration

```
PRAGMA "ENABLE_INOUT_BY_REF"

PRAGMA "DISABLE_INOUT_BY_REF"
```

Description

The action is meaningful only when you use a procedure with IN OUT parameters (a structure type) in context of object [Event](#). It is unimportant for RPC procedures.

When [CALL](#) action is executed, the action **PRAGMA** "ENABLE_INOUT_BY_REF" ensures the real parameter, which inputs to the procedure via its formal parameter, will not be copied to the formal parameter of procedure. It means the formal parameter represents a reference to the real parameter.

Basically, [CALL](#) action creates a copy of the real parameter and uses it as the formal parameter. It causes:

1. A procedure call copies the procedure values uselessly (it is slower).
2. The formal parameter is "separated" from the real parameter.

PRAGMA "ENABLE_INOUT_BY_REF" ensures the formal parameter is "a reference" to the real parameter and the value is not copied. The procedure call is faster.

Example

```
PROCEDURE Proc1(RECORD (SD.Data) _data)
.
.
_data[1]^Col :=2
.
.
.
END Proc1

BEGIN
PRAGMA "ENABLE_INOUT_BY_REF"
RECORD (SD.Data) _rec
REDIM _rec [2]

_rec[1]^Col :=0

CALL Proc1(_rec)

END
```

An application of **PRAGMA** "ENABLE_INOUT_BY_REF" action:

The formal parameter `_data` is the real parameter `_rec`.

When you write `_data[1]^Col := 2`, the value "2" is assigned to the real parameter `_rec[1]^Col`.

An application of [CALL](#) action (**PRAGMA** "ENABLE_INOUT_BY_REF" action is not used):

The formal parameter `_data` is an absolute copy of the real parameter `_rec`.

When you write `_data[1]^Col := 2`, the value "2" is assigned to the formal parameter `_data`.

After **CALL** `proc1(_rec)` is executed, the formal parameter `_data` will be copied to the real parameter `_rec`.



Related pages:

[Script actions](#)