

Does SET ADDRESS OF X TO Y work?

Author: Veryant Support

Saved From: <http://support.veryant.com/support/phpkb/question.php?ID=60>

Question:

I have some code that uses M\$ALLOC and then assigns the address of an array (in the linkage area) to the space just allocated. Code fragment below. There is a note in the migration section that "SET X TO ADDRESS OF Y" requires compiling with the -cp option to enable full POINTER support. But I am using only "SET ADDRESS OF X TO Y" and am not calling C functions.

```
000419      CALL "M$ALLOC" USING IMG-ROW-SIZE,
000420          IMG-ROW-PTR,
000421      SET ADDRESS OF IMG-ROW TO IMG-ROW-PTR,
Where IMG-ROW is an array in the linkage section.
```

Answer:

The answer is yes. SET ADDRESS OF X TO Y is supported. Use the -ca compiler option.

Note: If you plan to exchange pointers with C functions then enable full POINTER support by compiling with the -cp option.

If you are using a USAGE POINTER item internally, within COBOL (i.e. not to pass to C functions), SET X TO HANDLE OF Y works as desired and gives the same behavior as SET X TO ADDRESS OF Y in ACUCOBOL.

In addition, "SET ADDRESS OF X TO Y" works as expected.

Because of Java constraints, with isCOBOL, M\$ALLOC returns a handle, not an actual memory address. When you set the address of a linkage item to the handle value, the program behaves as desired because internally Java uses handles to identify objects, not pointers.

If you compile with -ca, then the program you described will compile and run as it did with ACUCOBOL. If you don't compile with -ca then you simply need to change USAGE POINTER to USAGE HANDLE wherever it occurs in the data division.

Here is an example program:

```
id division.
program-id. malloctest.
data division.
working-storage section.
77 img-row-ws pic x(50).
77 img-row-ptr usage handle.
linkage section.
01 img-row.
   03 filler pic x occurs 50.
procedure division.
main-logic.
   set address of img-row to address of img-row-ws.
   move "Message 1" to img-row.
   display img-row.
```

```
move "Message 2" to img-row.  
display img-row-ws.  
move "Message 3" to img-row-ws.  
display img-row.  
move "Message 5" to img-row-ws.  
CALL "M$ALLOC" USING length of img-row,  
    IMG-ROW-PTR,  
SET ADDRESS OF IMG-ROW TO IMG-ROW-PTR,  
  
move "Message 4" to img-row.  
display img-row.  
display img-row-ws.  
move "Message 6" to img-row-ws.  
move "Error" to img-row.  
display img-row-ws.
```

The output of program is:Â

Message 1
Message 2
Message 3
Message 4
Message 5
Message 6