

MÁS SOBRE LAS FUNCIONES ALV

1.- OBJETIVO.....	3
2.- INCORPORAR CHECK BOXES A UN ALV REPORT.	3
3.- PROCESAMIENTO DE MULTIPLES REGISTROS.	6
4.- EJEMPLO.....	7

1.- OBJETIVO.

El objetivo de este documento es el de profundizar un poco más en detalle en el uso de las funciones ALV, en concreto en la incorporación de checkboxes en dichos reports que permita seleccionar múltiples líneas para su posterior procesamiento.

Conviene leer previamente el documento "Funciones ALV manual en castellano.doc".

2.- INCORPORAR CHECK BOXES A UN ALV REPORT.

Para incorporar checkboxes en un alv report hay que seguir los pasos siguientes:

1. Incluir en la tabla de datos del alv una columna adicional TYPE C, que almacenará la información de si la línea esta o no seleccionada.

```
DATA: BEGIN OF I_ALV_DATA OCCURS 0,
      CHECKED TYPE C,           "Columna extra para los check boxes
      COLUM1 TYPE C,
      COLUM2 TYPE C,
      END OF I_TAB.
```

2. En la estructura utilizada para la definición del layout del report hay que informar el campo BOX_FIELDNAME con el nombre el campo de Checkboxes del apartado anterior (en este caso 'CHECKED').

```
IS_LAYOUT-BOX_FIELDNAME      = 'CHECKED'.
```

3. Se crea el fieldcatalog. (Ver manual "Funciones ALV manual en castellano.doc")

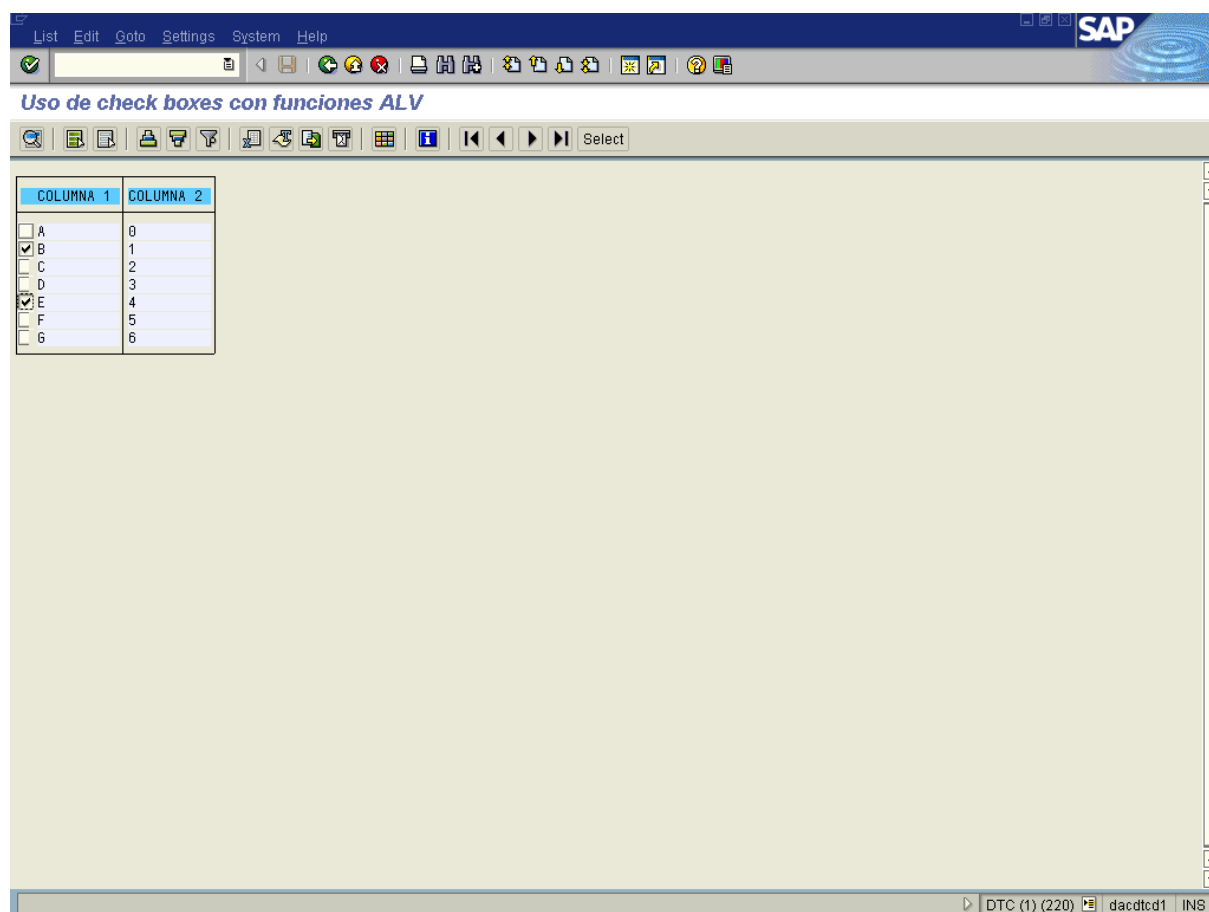
4. Se llama a la función ALV.

```
CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
  EXPORTING
    IS_LAYOUT      = IS_LAYOUT
    IT_FIELDCAT    = I_FIELDCAT
  TABLES
    T_OUTTAB       = I_ALV_DATA
  EXCEPTIONS
    PROGRAM_ERROR  = 1
    OTHERS         = 2.
```

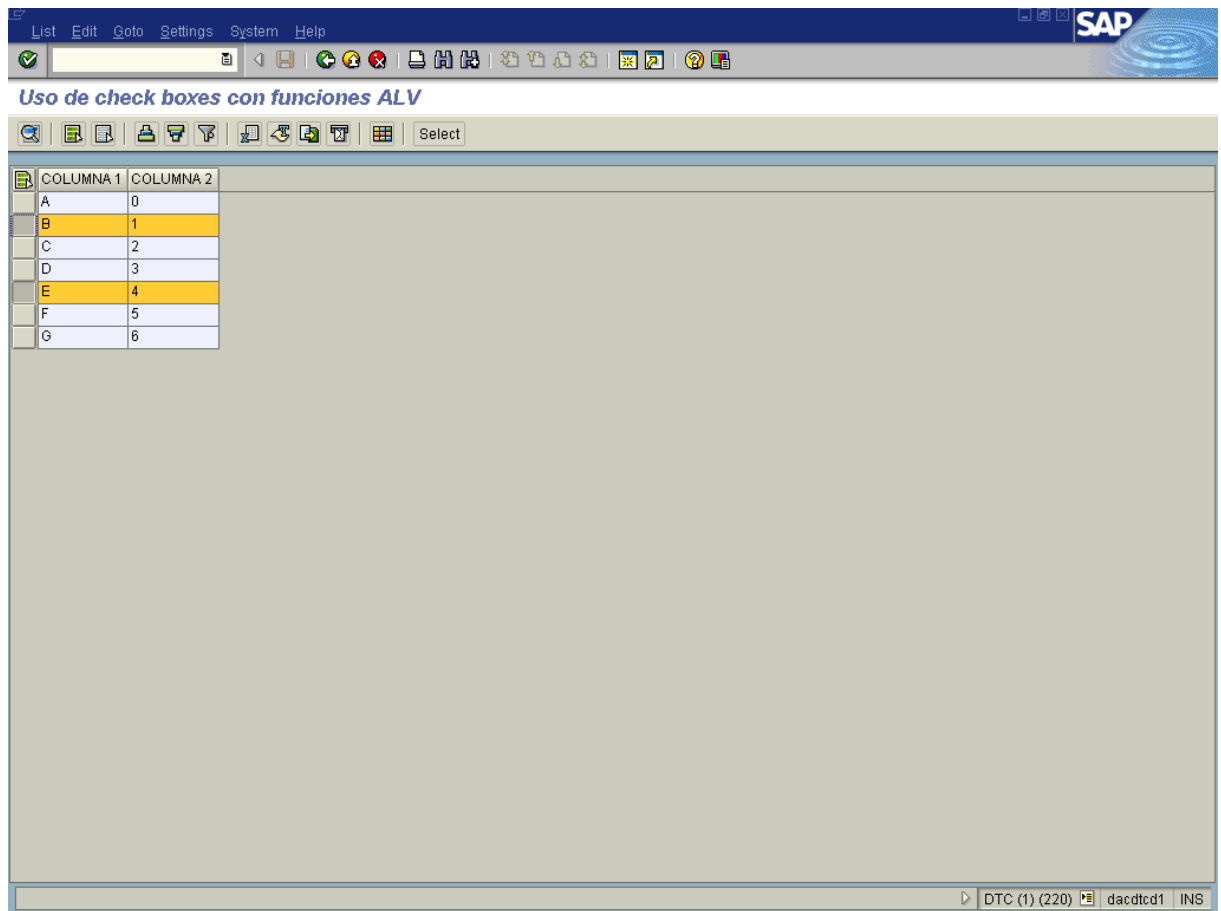
Esto es suficiente para visualizar en el report un check box al comienzo de cada línea del report. Si se selecciona una de esas líneas, automáticamente cambia el valor del campo de checkboxes (en este caso 'Checked') en la tabla de datos (en este caso I_ALV_DATA), que pasa a tener el valor 'X'. Esa información permanece en la tabla de datos aún después de haber salido del report con F3 (BACK). De modo que podemos realizar cualquier tipo de procesamiento posterior con esos registros.

El aspecto visual de esos check boxes difiere según se esté utilizando la función "REUSE_ALV_LIST_DISPLAY" o "REUSE_ALV_GRID_DISPLAY", como se puede observar a continuación:

Usando "REUSE_ALV_LIST_DISPLAY":



Usando "REUSE_ALV_GRID_DISPLAY":



3.- PROCESAMIENTO DE MULTIPLES REGISTROS.

Como se ha comentado en el apartado anterior, se puede realizar a posteriori el procesamiento de los registros seleccionados, pero también es posible realizarlo mientras aún tenemos el report visualizado.

En este caso necesitamos hacer uso de los eventos del ALV. Por ejemplo, si se desea disponer de un botón que inicie el procesamiento se emplearán los eventos 'PF_STATUS_SET' y 'USER_COMMAND'. El primero para cambiar la Application Toolbar añadiendo el botón, y el segundo para capturar la pulsación del mismo.

Para ello se crea una tabla de eventos de la siguiente forma.

```
REFRESH I_EVENTS.
* User_command event
CLEAR I_EVENT.
I_EVENT-NAME = 'USER_COMMAND'.
I_EVENT-FORM = 'F_USER_COMMAND'. (u otro nombre con el que se llamara al
                                FORM que manejará el evento)
APPEND I_EVENT TO I_EVENTS.

* Pf_status_set event
CLEAR I_EVENT.
I_EVENT-NAME = 'PF_STATUS_SET'.
I_EVENT-FORM = 'F_STATUS_SET'. (u otro nombre con el que se llamara al
                                FORM que manejará el evento).
APPEND I_EVENT TO I_EVENTS.
```

Las funciones de manejo de estos eventos tendrán la forma:

```
FORM F_USER_COMMAND USING PI_UCOMM    LIKE SY-UCOMM
                        PI_SELFIELD TYPE SLIS_SELFIELD.
    CASE PI_UCOMM.
        WHEN 'BUTTON1'.

            *   AQUÍ se incluirá el código para el procesamiento de los registros
            *   seleccionados
            *
        WHEN OTHERS.
            *   Nothing is done
        ENDCASE.
    ENDFORM.

FORM F_STATUS_SET USING PI_EXTAB TYPE SLIS_T_EXTAB.
* Set the new status
    SET PF-STATUS 'Nombre de Status' EXCLUDING PI_EXTAB.
ENDFORM.
```

Hay que crear un status para el programa con el botón que queremos.

La llamada a la función ALV se realizará utilizando los parámetros siguientes:

```
L_PROGRAM_NAME = SY-REPID.

CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
  EXPORTING
    I_CALLBACK_PROGRAM           = L_PROGRAM_NAME
    I_CALLBACK_PF_STATUS_SET    = 'F_STATUS_SET'
    I_CALLBACK_USER_COMMAND     = 'F_USER_COMMAND'
    IS_LAYOUT                   = IS_LAYOUT
    IT_FIELDCAT                 = I_FIELDCAT
    IT_EVENTS                   = I_EVENTS
  TABLES
    T_OUTTAB                   = I_ALV_DATA
  EXCEPTIONS
    PROGRAM_ERROR              = 1
    OTHERS                     = 2.
```

4.- EJEMPLO

```
REPORT Y_OG733_CHECKBOX .
```

```
* DATA definition
```

```
TYPE-POOLS: SLIS.
```

```
DATA: IS_FIELDCAT TYPE SLIS_FIELDCAT_ALV,
      I_FIELDCAT  TYPE SLIS_T_FIELDCAT_ALV,
      IS_LAYOUT   TYPE SLIS_LAYOUT_ALV,
      I_EVENTS    TYPE SLIS_ALV_EVENT OCCURS 0,
      I_EVENT     TYPE SLIS_ALV_EVENT.
```

```
DATA: BEGIN OF I_ALV_DATA OCCURS 0,
      CHECKED TYPE C,           "Columna extra para los check boxes
      COLUM1 TYPE C,
      COLUM2 TYPE C,
      END OF I_ALV_DATA.
```

```
DATA: I_ALV_DATA_AUX LIKE I_ALV_DATA OCCURS 0 WITH HEADER LINE.
```

```
DATA: V_PROGRAM LIKE SY-REPID.
```

```
*****
* EVENTO START-OF-SELECTION
*****
```

```
START-OF-SELECTION.
```

```
* Selección de datos
```

```
I_ALV_DATA-COLUM1 = 'A'.
I_ALV_DATA-COLUM2 = 0.
APPEND I_ALV_DATA.
```

```
I_ALV_DATA-COLUM1 = 'B'.
I_ALV_DATA-COLUM2 = 1.
APPEND I_ALV_DATA.
```

```
I_ALV_DATA-COLUM1 = 'C'.
I_ALV_DATA-COLUM2 = 2.
```

```

APPEND I_ALV_DATA.

I_ALV_DATA-COLUM1 = 'D'.
I_ALV_DATA-COLUM2 = 3.
APPEND I_ALV_DATA.

I_ALV_DATA-COLUM1 = 'E'.
I_ALV_DATA-COLUM2 = 4.
APPEND I_ALV_DATA.

I_ALV_DATA-COLUM1 = 'F'.
I_ALV_DATA-COLUM2 = 5.
APPEND I_ALV_DATA.

I_ALV_DATA-COLUM1 = 'G'.
I_ALV_DATA-COLUM2 = 6.
APPEND I_ALV_DATA.

*****
* EVENTO END-OF-SELECTION
*****
END-OF-SELECTION.

* LAYOUT
PERFORM F_BUILD_LAYOUT.

* FIELDATALOG
PERFORM F_BUILD_CATALOG.

* Events
PERFORM F_GET_EVENTS.

V_PROGRAM = SY-REPID.

* Show alv report
CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
EXPORTING
  I_CALLBACK_PROGRAM           = V_PROGRAM
  I_CALLBACK_PF_STATUS_SET    = 'F_STATUS_SET'
  I_CALLBACK_USER_COMMAND     = 'F_USER_COMMAND'

  IS_LAYOUT                   = IS_LAYOUT
  IT_FIELDCAT                 = I_FIELDCAT
  IT_EVENTS                   = I_EVENTS
  TABLES
    T_OUTTAB                   = I_ALV_DATA
  EXCEPTIONS
    PROGRAM_ERROR              = 1
    OTHERS                     = 2.

*&-----*
*&      Form  F_GET_EVENTS.
*&-----*
*      text
*-----*
*      --> p1      text
*      <-- p2      text
*-----*

FORM F_GET_EVENTS.

* Fill the internal table I_EVENT with all needed events
* User_command event
CLEAR I_EVENT.

```

```

REFRESH I_EVENTS.
I_EVENT-NAME = 'USER_COMMAND'.
I_EVENT-FORM = 'F_USER_COMMAND'.
APPEND I_EVENT TO I_EVENTS.

* Pf_status_set event
CLEAR I_EVENT.
I_EVENT-NAME = 'PF_STATUS_SET'.
I_EVENT-FORM = 'F_STATUS_SET'.
APPEND I_EVENT TO I_EVENTS.

ENDFORM.                " F_GET_EVENTS

*&-----*
*&      Form  F_USER_COMMAND
*&-----*
*      text
*-----*
* --> p1      text
* <-- p2      text
*-----*

FORM F_USER_COMMAND USING PI_UCOMM    LIKE SY-UCOMM
                        PI_SELFIELD TYPE SLIS_SELFIELD.

CASE PI_UCOMM.
  WHEN 'SELECT'.
    IF I_ALV_DATA_AUX[] IS INITIAL.
      * Almacenamiento de los registros seleccionados
      I_ALV_DATA_AUX[] = I_ALV_DATA[].
      DELETE I_ALV_DATA_AUX WHERE CHECKED NE 'X'.

      CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
        EXPORTING
          I_CALLBACK_PROGRAM           = V_PROGRAM
          I_CALLBACK_PF_STATUS_SET    = 'F_STATUS_SET'
          I_CALLBACK_USER_COMMAND     = 'F_USER_COMMAND'

          IS_LAYOUT                    = IS_LAYOUT
          IT_FIELDCAT                  = I_FIELDCAT
          IT_EVENTS                    = I_EVENTS
          TABLES
            T_OUTTAB                   = I_ALV_DATA_AUX
          EXCEPTIONS
            PROGRAM_ERROR              = 1
            OTHERS                    = 2.

      REFRESH I_ALV_DATA_AUX.
    ENDIF.

  WHEN OTHERS.
      * Nothing is done
ENDCASE.
ENDFORM.                " F_USER_COMMAND

*&-----*
*&      Form  F_STATUS_SET
*&-----*
*      text
*-----*
* --> p1      text
* <-- p2      text
*-----*

```

```
FORM F_STATUS_SET USING PI_EXTAB TYPE SLIS_T_EXTAB.
```

```
* Set the new status
SET PF-STATUS 'Y_OG733_CHECK_STATUS' EXCLUDING PI_EXTAB.
```

```
ENDFORM. " F_STATUS_SET
```

```
*&-----*
*&      Form  F_BUILD_LAYOUT
*&-----*
*      text
*-----*
* --> p1      text
* <-- p2      text
*-----*
```

```
FORM F_BUILD_LAYOUT.
```

```
IS_LAYOUT-BOX_FIELDNAME = 'CHECKED'.
```

```
ENDFORM. " F_BUILD_LAYOUT
```

```
*&-----*
*&      Form  F_BUILD_CATALOG
*&-----*
*      text
*-----*
* --> p1      text
* <-- p2      text
*-----*
```

```
FORM F_BUILD_CATALOG.
```

```
CLEAR IS_FIELDCAT.
REFRESH I_FIELDCAT.
```

```
IS_FIELDCAT-TABNAME = 'I_ALV_DATA'.
IS_FIELDCAT-FIELDNAME = 'COLUM1'.
IS_FIELDCAT-REPTTEXT_DDIC = 'COLUMNA 1'.
IS_FIELDCAT-DATATYPE = 'C'.
APPEND IS_FIELDCAT TO I_FIELDCAT.
```

```
IS_FIELDCAT-TABNAME = 'I_ALV_DATA'.
IS_FIELDCAT-FIELDNAME = 'COLUM2'.
IS_FIELDCAT-REPTTEXT_DDIC = 'COLUMNA 2'.
IS_FIELDCAT-DATATYPE = 'I'.
APPEND IS_FIELDCAT TO I_FIELDCAT.
```

```
ENDFORM. " F_BUILD_CATALOG
```