

BASICS FOR INLINE DECLARATIONS WITH SAP-ABAP

Author : Pradyut S. Vyas

Mentored by : Pranesh Kumar

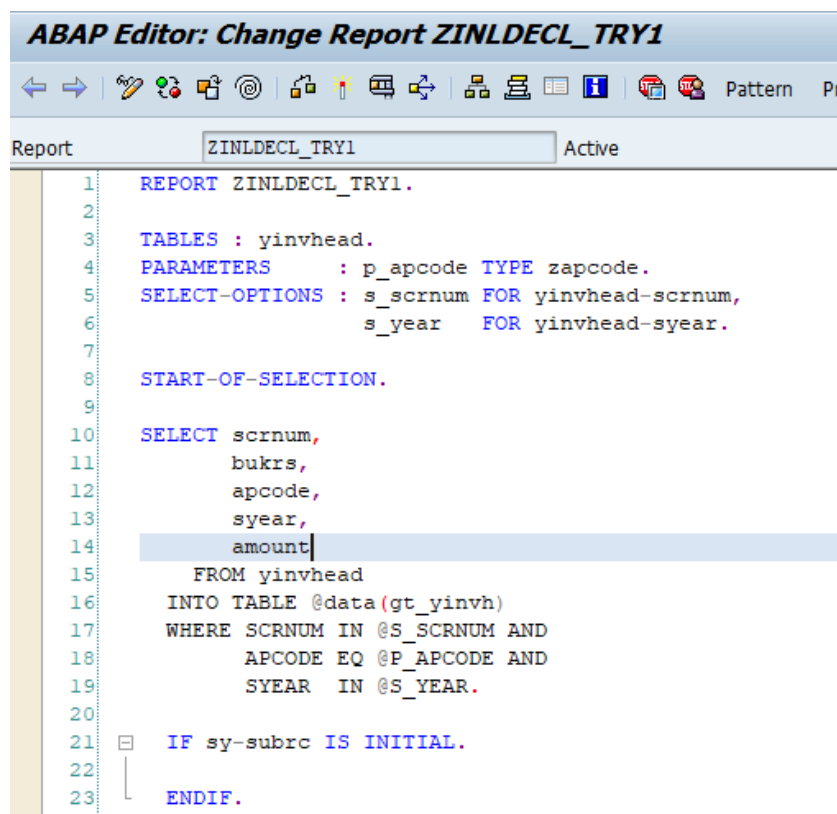
One of the most useful concept introduced in SAP NetWeaver 7.4 is ABAP inline declarations, this will help an ABAP Developer to simplify programming help to reduce unnecessary data declarations in a program.

Inline declarations are most useful in reducing onetime and local data declarations in a ABAP Object, for example If you want to store some value inside in a Function Module/Perform and you don't need to access them globally, you can declare a inline variable and use it locally.

For further clarity on this topic, I created a reference program locally parallel to this documentation and tried to put up as much as I can of the recent learnings on this topic, feel free to refer at the same:

Server: D90; Client: 551(ABAP Client) / 570(Test Client); Program Name: ZINLDECL_TRY1.

To start with, writing a basic report with a selection screen and a select query to fetch data based on user input and noting the observation for the same, that ways we can see **how to declare an internal table in the program.**

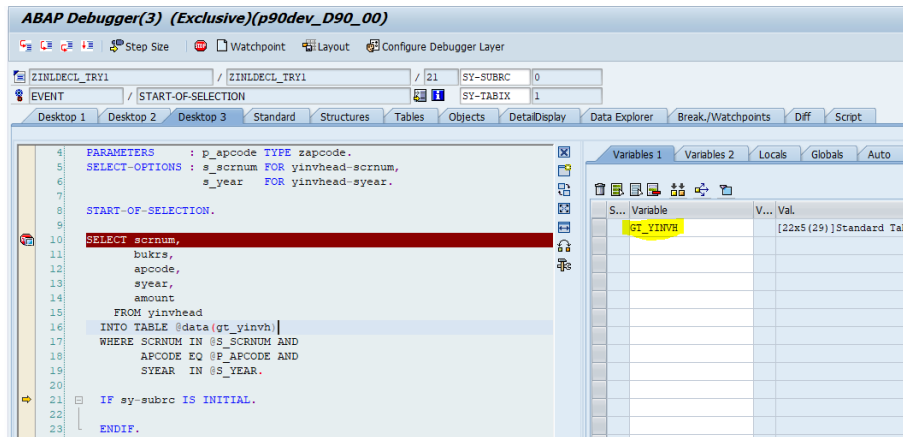


```
ABAP Editor: Change Report ZINLDECL_TRY1
Report: ZINLDECL_TRY1 Active
1  REPORT ZINLDECL_TRY1.
2
3  TABLES : yinvhead.
4  PARAMETERS : p_apcode TYPE zapcode.
5  SELECT-OPTIONS : s_scrnum FOR yinvhead-scrnum,
6                  s_year   FOR yinvhead-syear.
7
8  START-OF-SELECTION.
9
10 SELECT scrnum,
11        buhrs,
12        apcode,
13        syear,
14        amount
15 FROM yinvhead
16 INTO TABLE @data(gt_yinvh)
17 WHERE SCRNUM IN @S_SCRNUM AND
18        APCODE EQ @P_APCODE AND
19        SYEAR  IN @S_YEAR.
20
21 IF sy-subrc IS INITIAL.
22
23   ENDIF.
```

For the code written above, following things are to be observed:

- 1- No type/data declaration is being used for the internal table in which the data is being fetched, prior to select query. The table is directly declared in the select query at **"INTO TABLE"** clause by using **@data(internal_table)**.
- 2- A comma(,) between each two fields that are required to fetch in the select statement.
- 3- **WHERE CLAUSE**, "@" used before writing every parameter/select-option.

Running the same program in debugger mode, to observe we get to see:



ABAP Debugger(3) (Exclusive)(p90dev_D90_00)

EVENT / START-OF-SELECTION / 21 SY-SUBRC 0 / 1 SY-TABIX 1

Tables Table Contents

Table GT_YINVH

Attributes Standard [22x5(29)]

Insert Column Columns ...

Row	SCRNUM [N(11)]	BUKRS [C(4)]	APCODE [C(3)]	SYEAR [N(4)]	AMOUNT [P(7)]...
1	00011000017	5075	RC3	2019	118.00
2	00011000018	5075	RC3	2019	118.00
3	00011000015	5075	RC3	2019	1180.00
4	00011000016	5075	RC3	2019	118.00
5	00011000012	5075	RC3	2019	70800.00
6	00011000002	5075	RC3	2019	8260.00
7	00011000009	5075	RC3	2019	70800.00
8	00011000006	5075	RC3	2019	70800.00
9	00011000014	5075	RC3	2019	11800.00
10	00011000003	5075	RC3	2019	169000.00

Further, if we are required to dump data from one internal table to another for some operation an internal table can be declared by simply using the declaration operator **"DATA"**.

```

19 SYEAR IN @s_year.
20
21 IF sy-subrc IS INITIAL.
22 DATA(gt_yinvd) = gt_yinvh.
23 DELETE gt_yinvd WHERE syear IS INITIAL.
24 ENDIF.

```

Desktop 1 Desktop 2 Desktop 3 Standard Structures Tables Objects DetailDis					
Tables Table Contents					
Table GT_YINVD					
Attributes Standard [22x5(29)]					
Insert Column Columns ...					
Row	SCRNUM [N(11)]	BUKRS [C(4)]	APCODE [C(3)]	SYEAR [N(4)]	AMOUNT [P(7)...
1	00011000017	5075	RC3	2019	118.00
2	00011000018	5075	RC3	2019	118.00
3	00011000015	5075	RC3	2019	1180.00
4	00011000016	5075	RC3	2019	118.00
5	00011000012	5075	RC3	2019	70800.00
6	00011000002	5075	RC3	2019	8260.00
7	00011000009	5075	RC3	2019	70800.00

Moving forward, similarly we can use the “**DATA**” declaration operator for further declaring a **work area as well as variables**, below shown code are several examples for the same:

```

19      SYEAR IN @S_YEAR.
20
21  IF sy-subrc IS INITIAL.
22  *   DATA(gt_yinvd) = gt_yinvh.
23  *   DELETE gt_yinvd WHERE syyear IS INITIAL.
24  LOOP AT gt_yinvh INTO DATA(gw_yinvh).
25
26      ENLOOP.
27  ENDIF.

```

```

28      READ TABLE gt_yinvh INDEX 1 INTO DATA(gw_yinvh).
29      DATA(lv_var1) = gw_yinvh-scrnum .
30      CONCATENATE gw_yinvh-scrnum gw_yinvh-bukrs INTO DATA(lv_var2) SEPARATED BY '///'.

```

ZINDECL_TRY1 / ZINDECL_TRY1 / 31 SY-SUBRC 0

EVENT / START-OF-SELECTION SY-TABIX 1

Desktop 1 Desktop 2 Desktop 3 Standard Structures Tables Objects DetailDisplay Data Explorer Break/Watchpoints Diff Script

```

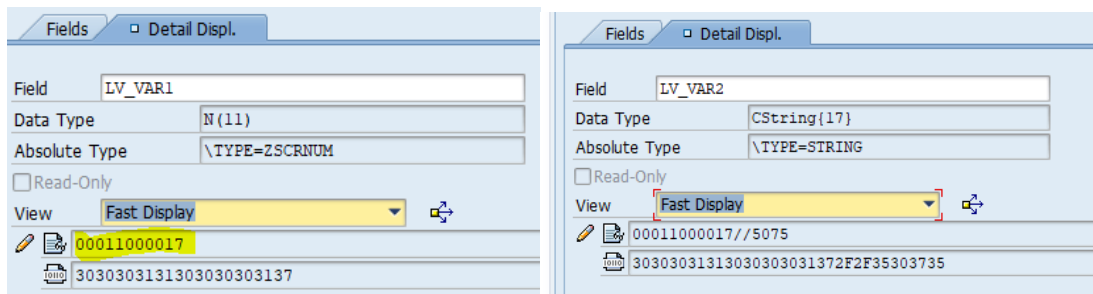
19      SYEAR IN @S_YEAR.
20
21  IF sy-subrc IS INITIAL.
22  *   DATA(gt_yinvd) = gt_yinvh.
23  *   DELETE gt_yinvd WHERE syyear IS INITIAL.
24  *   LOOP AT gt_yinvh INTO DATA(gw_yinvh).
25
26      ENLOOP.
27  ENDIF.
28      READ TABLE gt_yinvh INDEX 1 INTO DATA(gw_yinvh).
29      DATA(lv_var1) = gw_yinvh-scrnum .
30      CONCATENATE gw_yinvh-scrnum gw_yinvh-bukrs INTO DATA(lv_var2) SEPARATED BY '///'.
31      PERFORM subrout_1.
32  ENLOOP.
33  ENDIF.

```

Variables 1 Variables 2 Locals Globals Auto

S...	Variable	V...	Val.
	LV_VAR2		00011000017//5075
	LV_VAR1		00011000017
	GW_YINVH		Structure: flat, not

Structures Fld.list					
Struct. GW_YINVH					
Struc. Type Structure: flat, not charlike(29)					
Exp.	Component	Val...	Val.	Ch...	Technical Type
	SCRNUM		00011000017		N(11)
	BUKRS		5075		C(4)
	APCODE		RC3		C(3)
	SYEAR		2019		N(4)
	AMOUNT		118.00		P(7) DECIMALS 2



Further for **declaring a field symbol**, SAP has provided a new declaration operator **FIELD-SYMBOL(...)**.

```

LOOP AT gt_yinvh ASSIGNING FIELD-SYMBOL <fs_yinvh> .

  DATA(lv_var1) = <fs_yinvh>-scrnum .
  CONCATENATE <fs_yinvh>-scrnum <fs_yinvh>-bukrs INTO DATA(lv_var2) SEPARATED BY '///'.

  READ TABLE gt_yinvh INDEX 1 INTO DATA(gw_yinvh).

  DATA(lv_var1) = gw_yinvh-scrnum .
  CONCATENATE gw_yinvh-scrnum gw_yinvh-bukrs INTO DATA(lv_var2) SEPARATED BY '///'.
  PERFORM subrout_1.
ENDLOOP.
UNASSIGN <fs_yinvh> .

```

SOME POINTS TO REMEMBER:

- **SORTING** can be done on database level using **ORDER BY** keyword in select query right after **WHERE** clause.

```

SELECT scrnum,
       bukrs,
       apcode,
       syear,
       amount
FROM yinvhead
INTO TABLE @DATA(gt_yinvh)
WHERE scrnum IN @s_scrnum AND
       apcode EQ @p_apcode AND
       syear IN @s_year
ORDER BY scrnum, bukrs, apcode, syear.

```

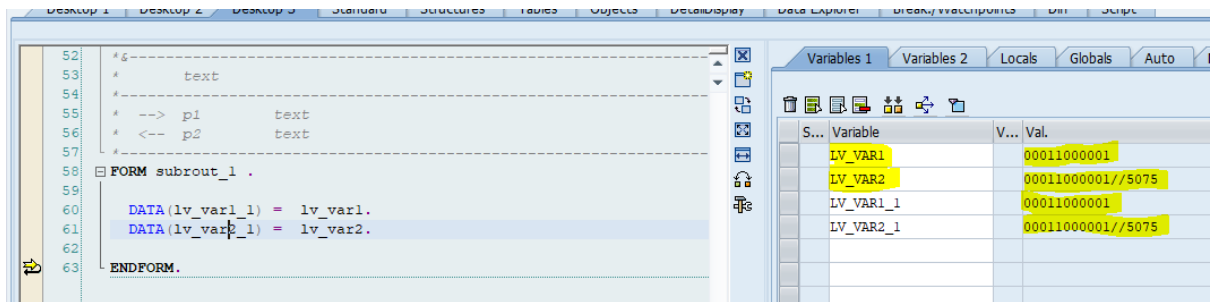
- Observations of using inline declarations in subroutines:
(a) Declaring a variable in main program and using it in it's subroutine.

```

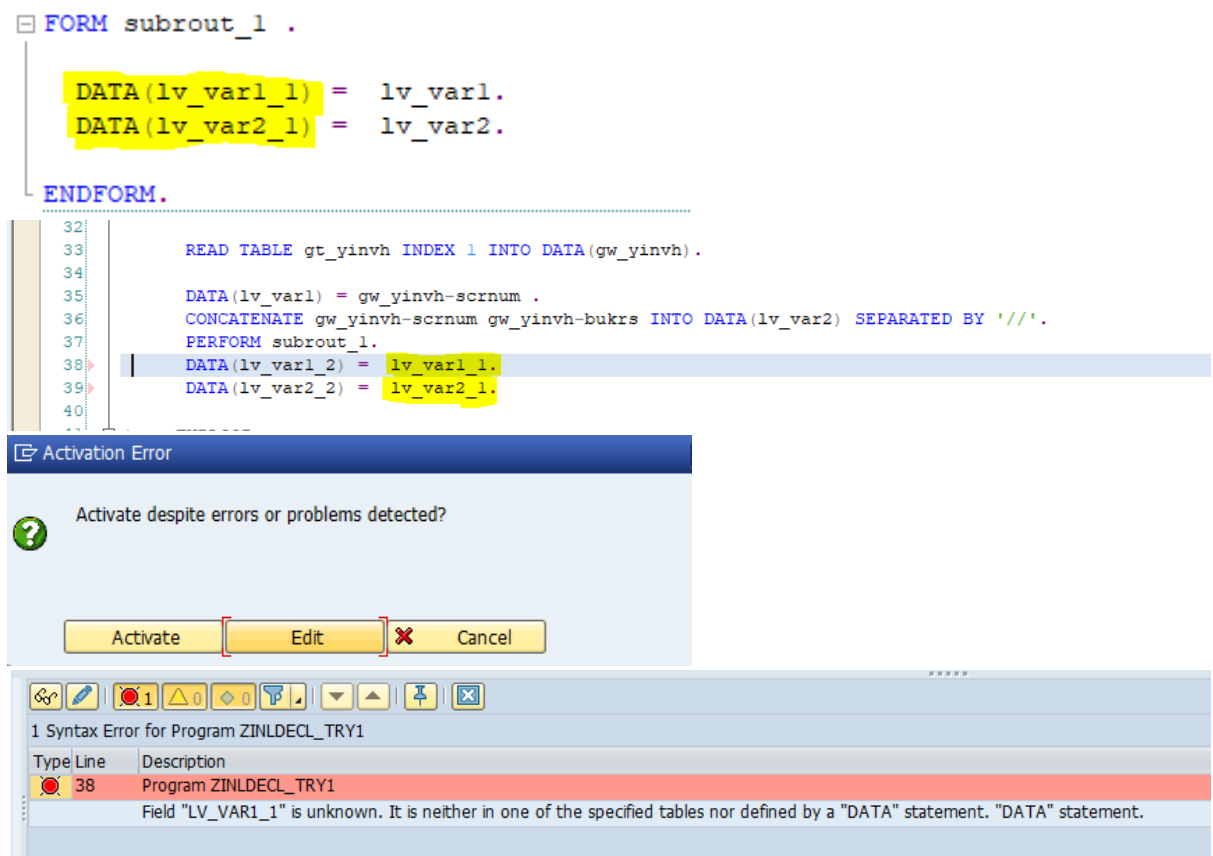
DATA(lv_var1) = gw_yinvh-scrnum .
CONCATENATE gw_yinvh-scrnum gw_yinvh-bukrs INTO DATA(lv_var2) SEPARATED BY '///'.
PERFORM subrout_1.

58 FORM subrout_1 .
59
60   DATA(lv_var1_1) = lv_var1.
61   DATA(lv_var2_1) = lv_var2.
62
63   ENDFORM.

```



(b) Declaring a variable in subroutine and trying to use it outside post the subroutine's execution.



Hence, objects declared in subroutines are unusable outside it.