SAP Summer School

Setup

internal

value – inspired by people

.msg

1. Packages and Reports
2. Development Objects. Active and Inactive
3. How to debug ABAP Programs

1 Packages and reports

➔ navigate to your $TMP package (local objects in se80)

1. Create a new package using the naming convention **TEST_INTERNSHIP2023_*<your initials>*** (by right-clicking on the super-package and selecting "*New -> Package*"). In this package you will store all your development artifacts during the course.

2. In a new session (transaction /ose80) ,open the ABAP Program **Z01_ABAP2023_SAMPLE** and copy it to you own package, under the name **Z01_ABAP2023_*< your initials >***

3. Activate the program.

The program can be activated using Ctrl+F3 or 

4. Execute the program:

a. By clicking on the wrench () symbol, or

b. By pressing F8

What does the program do? Experiment with the program and have a look at the code to get an understanding of how it operates.

2 Development objects. Active and inactive versions

While it is being implemented, any ABAP development artifact is in an ***inactive*** state. However, in order to be executable in the ABAP runtime environment, the object has to be ***activated*** first. During activation syntax and other consistency checks are performed, after which a runtime object is generated.
If the object is edited once again, an inactive copy is created. This copy can be edited without affecting the active runtime object (until a new activation takes place).
Display & Edit Mode. Activating Objects.
1. Open the source code of program **Z01_ABAP2023_*< your initials >***in the Object Navigator

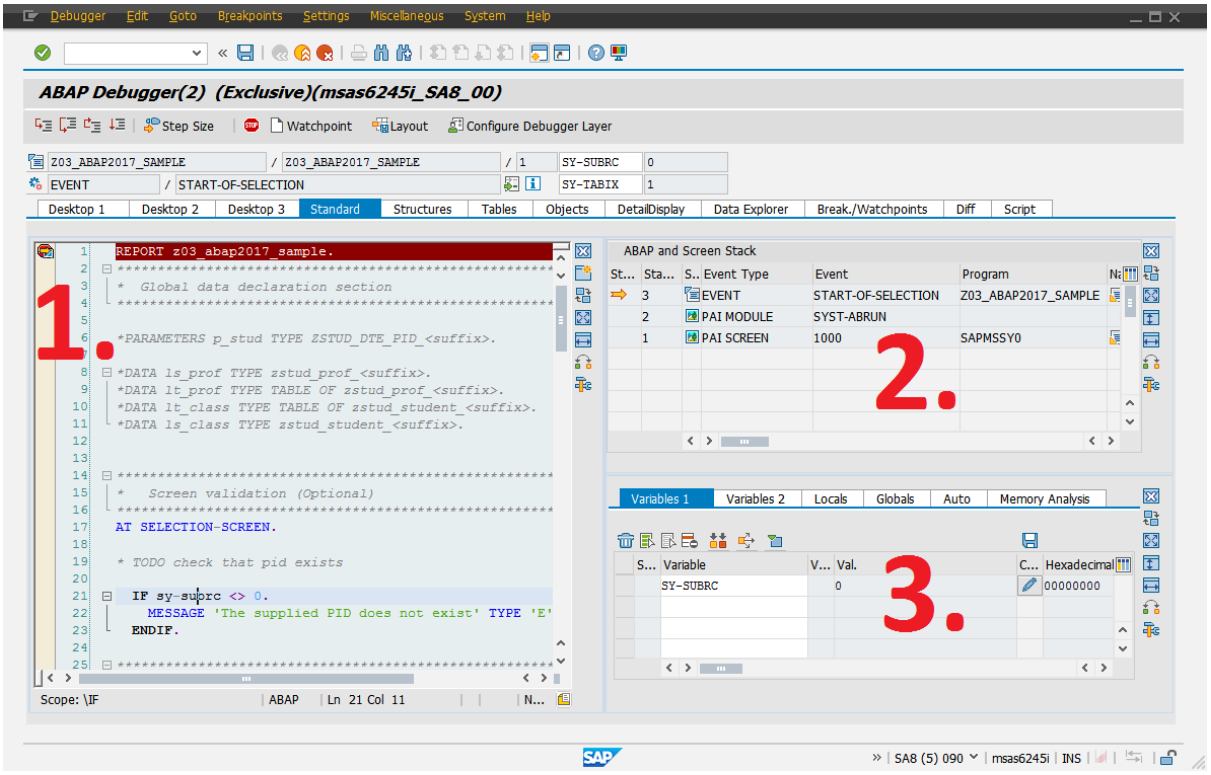
2. Switch from *Display* to *Edit Mode* by clicking the pen () icon, or by pressing *Ctrl+F1*

3. On line 8, change the name of variable *p_ord2* to *p_ord*. Save your changes (*Ctrl+S*) and perform a syntax check with *Ctrl+F2* or using the icon 

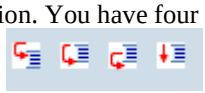
4. Correct any remaining syntax error. Afterwards the program can be activated using *Ctrl+F3* or 

3 How to debug ABAP programs

The ABAP debugger is a straightforward tool that helps you analyze what is wrong with your programs. The layout is divided in different tabs: (1) standard desktop with your source code, (2) the call stack and (3) the variables and their values. By double clicking on the variables in (3), you can analyze their content. We navigate back to the standard desktop using the Back button. ( )



In order to enter this perspective, you simply have to add a breakpoint to your active source code and run your application. You have four options for tracing your code:

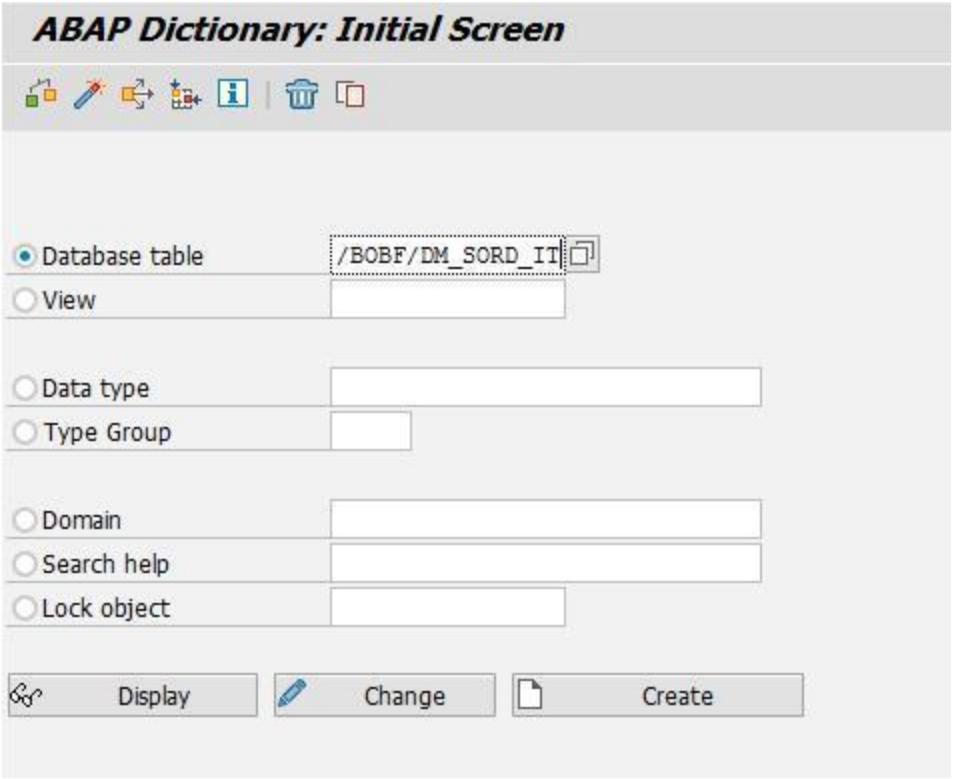


- F5 – single step mode
- F6 – execute
- F7 – return
- F8 – continue.

4 Assignment

The provided sample program lists sales orders available in the system. Every sales order is in fact associated with multiple sales order items. The program will be enhanced to reflect this fact.

1. Open two sessions and run the transaction SE11 (Data Dictionary) in each one of them. Enter the name of the 2 database tables (sales order header **- /BOBF/DM_SORD_RT** and items **- /BOBF/DM_SORD_IT**) in the first input field ("Database table") and select "Display". This will provide details regarding how the tables are defined.



2. Analyze how the two tables are connected to each other. To make things easier, the table contents can be visualized by clicking on the icon.(Answer: The tables are connected through a parent-child relationship, in the following way -> the value of the field **DB_KEY** from the parent entity will be stored in the field **PARENT_KEY** of the child entity).

Select a value of the field DB_KEY from the contents of the table **/BOBF/DM_SORD_RT,** and use it to filter the contents of the table **/BOBF/DM_SORD_IT,** to prove this relationship.
3. After having studied the program source code **Z01_ABAP2023_*< your initials >*** , perform the following enhancements and test to check the results:

a. Implement the three GET methods (HINT: use F1 help to see the correct SELECT syntax)
METHOD get_all_sales_orders. -> returns a table containing all information from the sales order table
METHOD get_orders_by_id. -> returns a table containing all orders having a giver ORDER_ID
METHOD get_items_for_order. -> returns a table containing all items for a specific order (HINT: use the DB_KEY->PARENT_KEY relationship)

b. When a sales order line is double-clicked, on the next screen the list of corresponding sales order items should be displayed. The code that will be executed when double-clicking a line is the following. Re-write the WRITE statement using the corresponding method call (get_items_for_order).

```
AT LINE-SELECTION.

  WRITE: 'selected order:'(001), ls_order-order_id.
```

c. A message should be displayed if no items were found for the selected sales order (check the link for information: https://wiki.scn.sap.com/wiki/display/ABAP/Number+of+ways+to+display+the+messages)

Optional assignment:
4. Open the ABAP Program **Z01_ABAP2023_INS_SAMPLE** and copy it to you own package, under the name **Z01_ABAP2023_INS_*< your initials >***. Using the information from the report, resolve all the assignments.