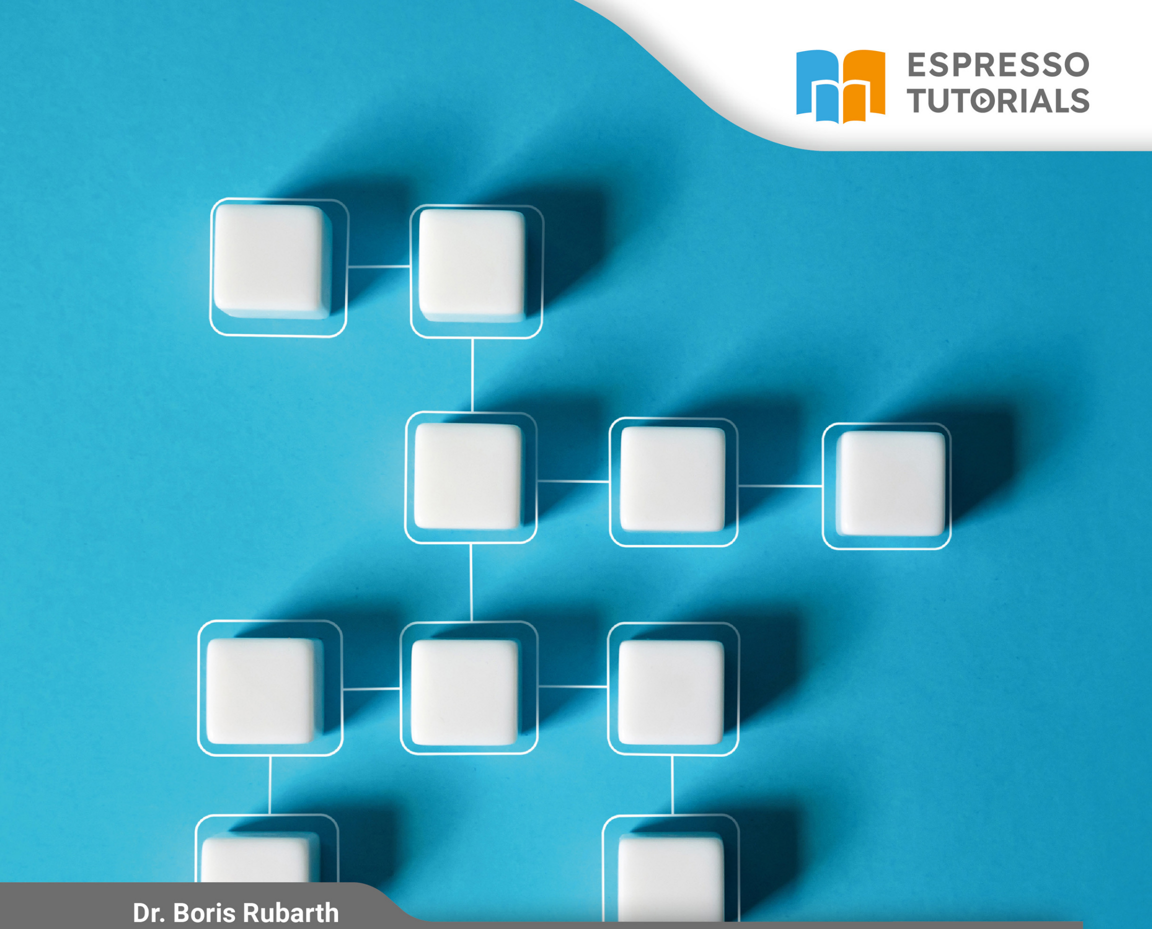
ESPRESSO TUTORIALS



Dr. Boris Rubarth

# First Steps in SAP® ABAP®

Second Edition

- Step-by-step instructions for beginners
- Comprehensive descriptions and code examples
- A guide to creating your first ABAP application
- Tutorials that provide answers to the most commonly asked programming questions

# Table of Contents

# 3 Using the ABAP Workbench

**This chapter introduces the ABAP Workbench and explains the relationship between the ABAP Workbench and the development tools. We will take a closer look at the ABAP Editor, one of the most widely used tools in the ABAP Workbench. We will also look at how to customize some of the Workbench features.**

You can start a report without using the ABAP Editor – if you know how to define a transaction code. Let’s briefly cover how to define a transaction code.

## 3.1 Starting the ABAP Workbench

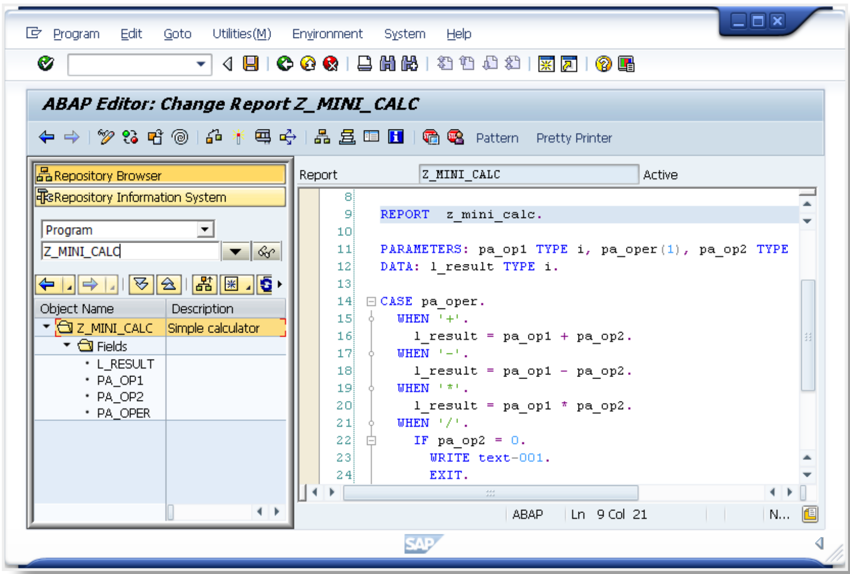
There are two ways to open the ABAP Workbench (also known as the *Object Navigator*).

One option is to start the ABAP Workbench directly with transaction code SE80. Select the object you want to work on (for example, your report) and then the Workbench will launch the appropriate tool (for reports, the ABAP Editor opens).

It works the other way as well: when you are working on your report using the ABAP Editor, open the ABAP Workbench using the menu path UTILITIES • DISPLAY OBJECT LIST. This will open a separate area on the left-hand side – as displayed in Figure 3.1 – which is called the *object list*.

The object list for a report shows the data objects used in the report.

You can open the object list not only in the ABAP Editor, but also from each ABAP tool. The object list is a separate window section with its own navigation; it is not always synchronized with the object displayed on the right-hand side. You can always synchronize both sections in one direction or the other: either use the menu path listed above to show the object list for an object displayed in the tool (right-hand side); alternatively, select an object in the object list (left-hand side) by double-clicking it and the appropriate tool will be displayed on the right-hand side of the screen. Later, we will see that it can be useful not to have both sections synchronized.



```
REPORT  z_mini_calc.

PARAMETERS: pa_op1 TYPE i, pa_oper(1), pa_op2 TYPE
DATA: l_result TYPE i.

CASE pa_oper.
  WHEN '+'.
    l_result = pa_op1 + pa_op2.
  WHEN '-'.
    l_result = pa_op1 - pa_op2.
  WHEN '*'.
    l_result = pa_op1 * pa_op2.
  WHEN '/'.
    IF pa_op2 = 0.
      WRITE text-001.
      EXIT.
```

*Figure 3.1: The object list for a report*

Above the object list, three buttons with icons and text are visible: MIME REPOSITORY, REPOSITORY BROWSER (highlighted), and REPOSITORY INFOSYSTEM. The REPOSITORY BROWSER is currently selected and below that, a dropdown box shows *Program* and the name of our report. Our report is part of the *ABAP Repository*, like many other ABAP object types (for example, ABAP classes) as well.

### The ABAP Repository

All ABAP Repository objects are stored in the database. When we save any changes for our report, the changes are written to the database, into the repository area in the database.

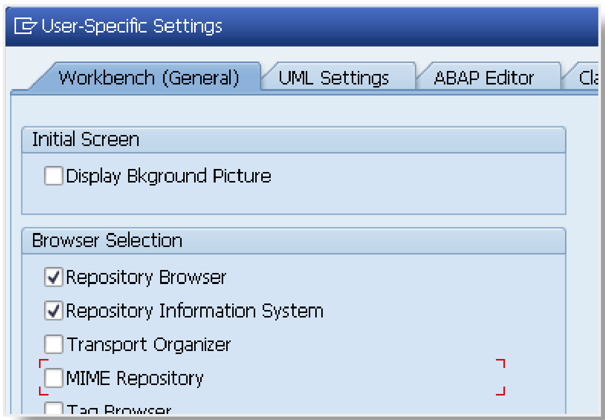
Business data in the database is client-specific data, since a client is a unit that is self-contained in terms of business, organization, and data. However, repository objects are client-independent: a report created in one client can be used in the other clients of that system as well. More than likely, the business results of the report are different when started in different clients, since the report will be working on different business data (which is client-specific data).

You can change user-specific settings to determine which tools are displayed in the ABAP Workbench, as described in the following section.

## 3.2 Configuring ABAP Workbench options

The ABAP Workbench can display more tools than you will need in your daily work, so it is a good idea to display only the tools you need.

Select UTILITIES • SETTINGS to change your user-specific settings. The first tab WORKBENCH (GENERAL) allows you to set the BROWSER SELECTION of your choice. Use this tab to select only the REPOSITORY BROWSER and the REPOSITORY INFORMATION SYSTEM, as shown in Figure 3.2 below.



Initial Screen

- [ ] Display Bkground Picture

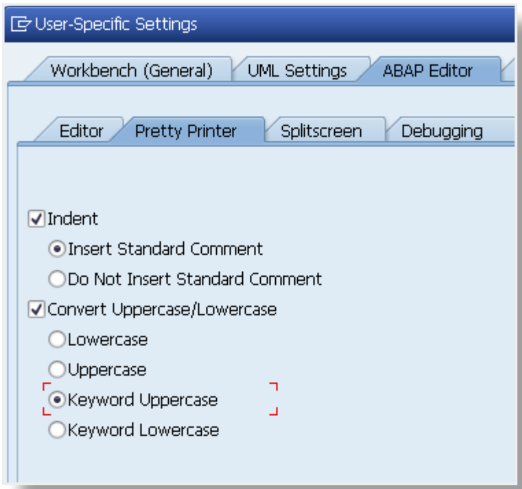
Browser Selection

- [✓] Repository Browser
- [✓] Repository Information System
- [ ] Transport Organizer
- [ ] MIME Repository
- [ ] Tag Browser

*Figure 3.2: ABAP Workbench Settings*

As you can see on the third tab, there is an area for user-specific settings for the ABAP EDITOR as well. Select that tab to see the PRETTY PRINTER tab and use the CONVERT UPPERCASE/LOWERCASE option to select KEYWORD UPPERCASE, as shown in Figure 3.3.

After closing the popup window, use the PRETTY PRINTER button to automatically convert ABAP keywords to upper case (and to properly indent the lines, if you selected that option). These options optimize the display of your report according to your preferences, they are not relevant for the runtime of your report.


User-Specific Settings

Workbench (General) | UML Settings | ABAP Editor

Editor | Pretty Printer | Splitscreen | Debugging

[✓] Indent
(●) Insert Standard Comment
( ) Do Not Insert Standard Comment
[✓] Convert Uppercase/Lowercase
( ) Lowercase
( ) Uppercase
(●) Keyword Uppercase
( ) Keyword Lowercase

*Figure 3.3: ABAP Editor settings for Pretty Printer*

Now back to the selection *Program* in the REPOSITORY BROWSER: let's take a look at the dropdown menu options.

## 3.3 Working with packages and transport requests

When you open the dropdown menu, you will see some of the existing repository object types, like *Program* and *Package* as displayed in Figure 3.4.

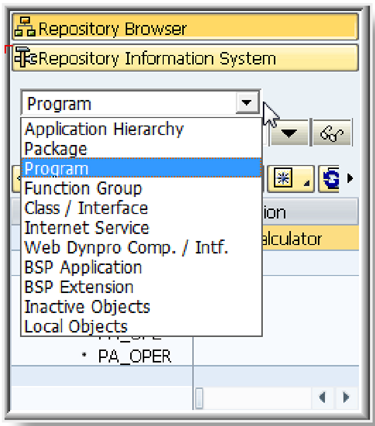


*Figure 3.4: Repository object types in the Repository Browser*

When we created our first report in Chapter 1, we skipped over an explanation of packages (Figure 1.3). Let's now examine packages in ABAP. Each object is assigned to a package, along with other objects that pertain to the same area of development or business functionality. A package bundles different types of objects, like reports, UI objects, and ABAP classes, as long as they belong together from the application view. Several developers may work on the objects in one package.

Each ABAP object has another assignment that we previously skipped over: transport requests. A transport request bundles objects that will be transported together along the transport path.

**ABAP transports**

ABAP objects are developed in a development system. When a development phase is done, the objects are transported to the next system (typically a quality assurance or test system). The third step in a typical transport landscape is a production system, in which no development is allowed.

All three systems are separate installations of AS ABAP, with their own system ID.

A transport request may include objects from different packages, and vice versa. Generally, as a developer, you do not have to worry about transports. As soon as you are done with the current development cycle, you will have to change all of your objects to status *Final* (which is called *Release your transport request*). Typically, the project lead on your team will determine when to transport the objects. The transport is technically carried out by AS ABAP (and its tools).

To create a package, select the menu entry *Package* in the Repository Browser, and enter the name of your package, for example, ZFIRSTPACKAGE. Press `Enter`: the ABAP Workbench will then display a popup, asking whether you would like to create the (not yet existing) package. Confirm with YES.

Add a short description of the package, as displayed in Figure 3.5.

# B Index