ABAP Programming for SAP

ABAP (Advanced Business Application Programming) is the name of SAP's proprietary, fourth-generation programming language. It was specifically developed to allow the mass-processing of data in SAP business applications.

By working with ABAP in SAP NetWeaver, companies running the SAP ERP and SAP S/4HANA business solutions have the opportunity to customize those systems to better meet their needs.

ABAP is a multi-paradigm programming language, meaning programmers can utilize procedural, object-oriented, and other programming principles. While it is SAP's primary programming language, programs written with ABAP can run alongside those based on other programming languages such as Java, JavaScript, and SAPUI5.

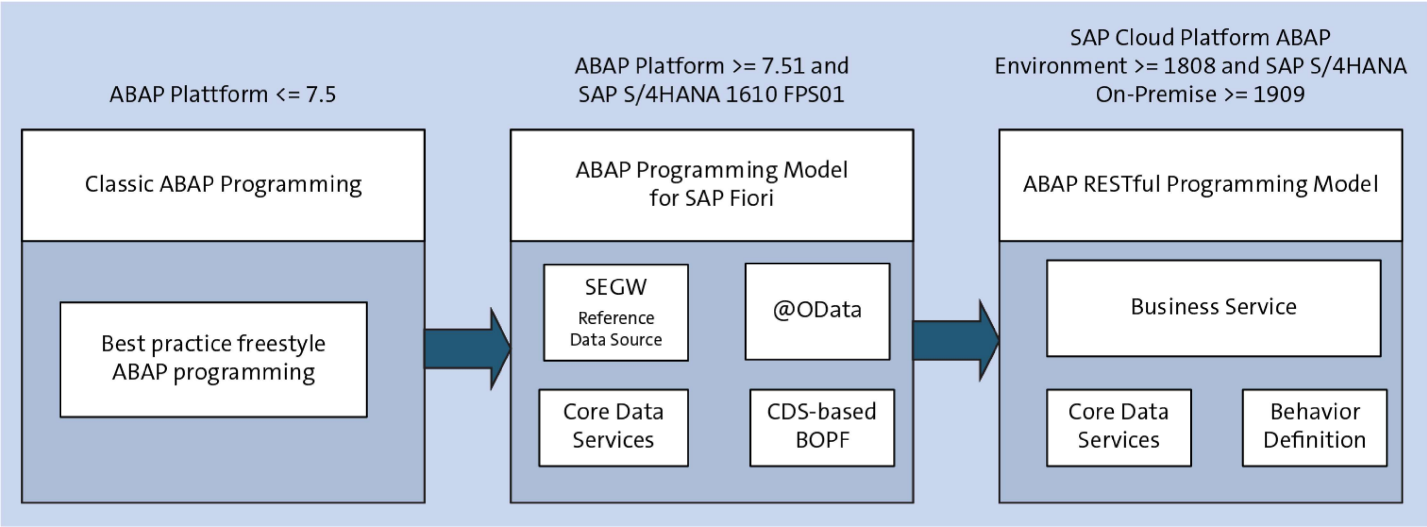
Table of Contents

1. A Brief History of ABAP
2. New vs. Old ABAP
3. When to Use ABAP
4. The ABAP Workbench
    A. ABAP Editor
    B. ABAP Dictionary
    C. ABAP Painter
    D. Function Builder
    E. Class Builder
    F. Web Application Builder
5. Other Key ABAP Terms
6. Additional Resources
    A. Blog Posts
    B. Books by SAP PRESS
    C. Videos

A Brief History of ABAP

ABAP was first introduced by SAP in the 1980s. Throughout the years, various enhancements to the language increased what programmers could do with it. For example, through April 2000 programs could only be created *procedurally*, meaning a program had to follow a set of pre-defined "procedures" to perform a certain task successfully.

In May 2000, SAP changed ABAP with release 4.6C, allowing for object-oriented programming (OOP). This programming strategy involves multiple individual "objects" interacting with one another, allowing programs to grow more complex with the use of ABAP design patterns and other OOP practices.



With the release of ABAP 7.4 and 7.5 in the early to mid-2010s, SAP gave object-oriented programmers using ABAP some powerful new features to play around with, vastly reducing the amount of code needed for common tasks. The end result is that code ends up being up to 50% shorter plus both cleaner and clearer—making both programmers and end users' lives easier.

Other new features made available to ABAP programmers in the 2010s were extended syntax for Open SQL, ABAP Managed Database Procedures (AMDP), and core data services (CDS) Views.

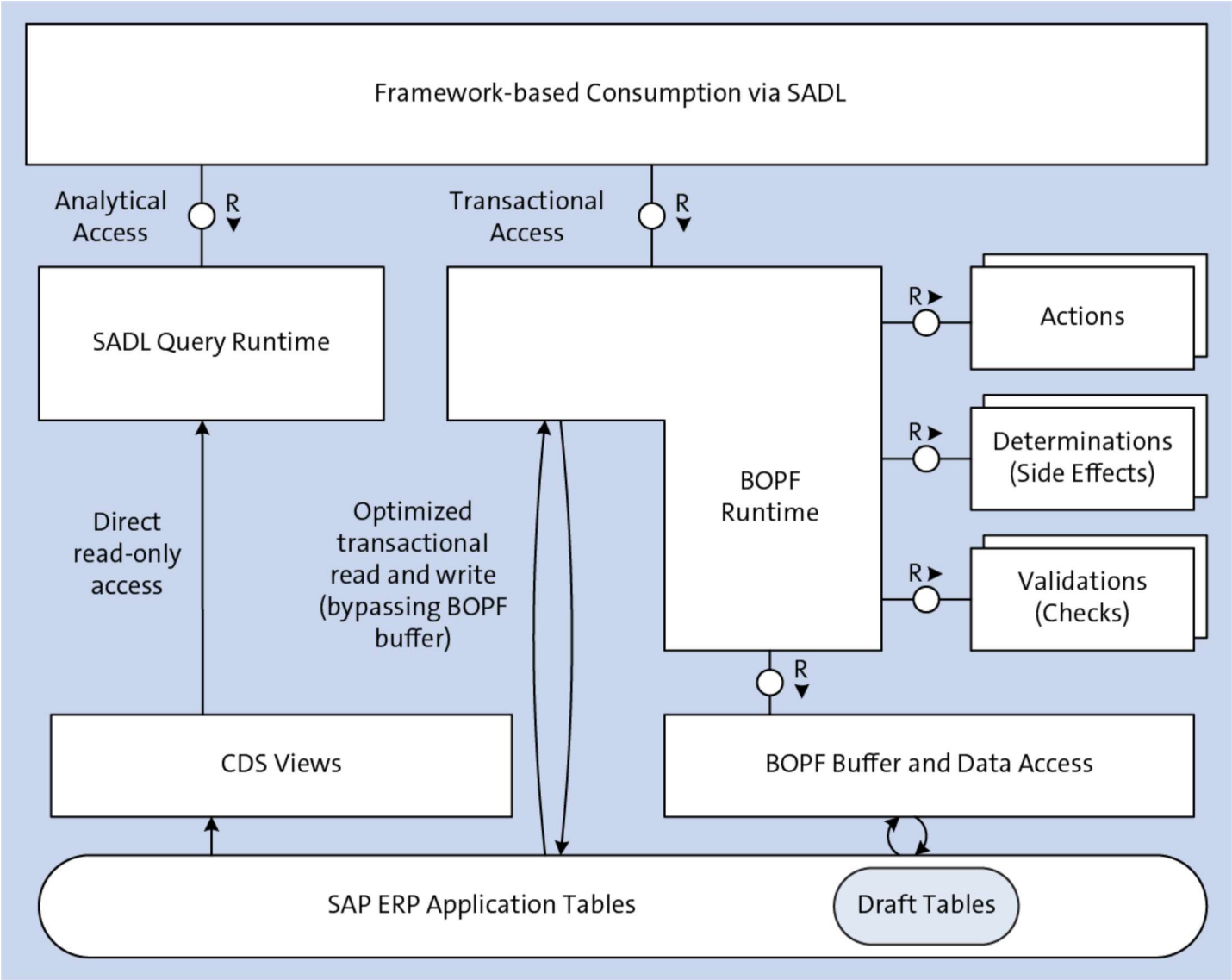
Perhaps the biggest and most important change to ABAP programming came with the invention and release of the SAP HANA platform in the summer of 2011. Due to the in-memory architecture of this database, processing that formerly happened on the application layer now could be done on the database layer.

With the traditional, row-based database architecture of SAP R/3, it was important to have ABAP code run in the application layer rather than the database layer to save memory usage for further tasks. But SAP HANA allowed, and even encouraged, tasks to be completed in real-time by using in-memory technology.

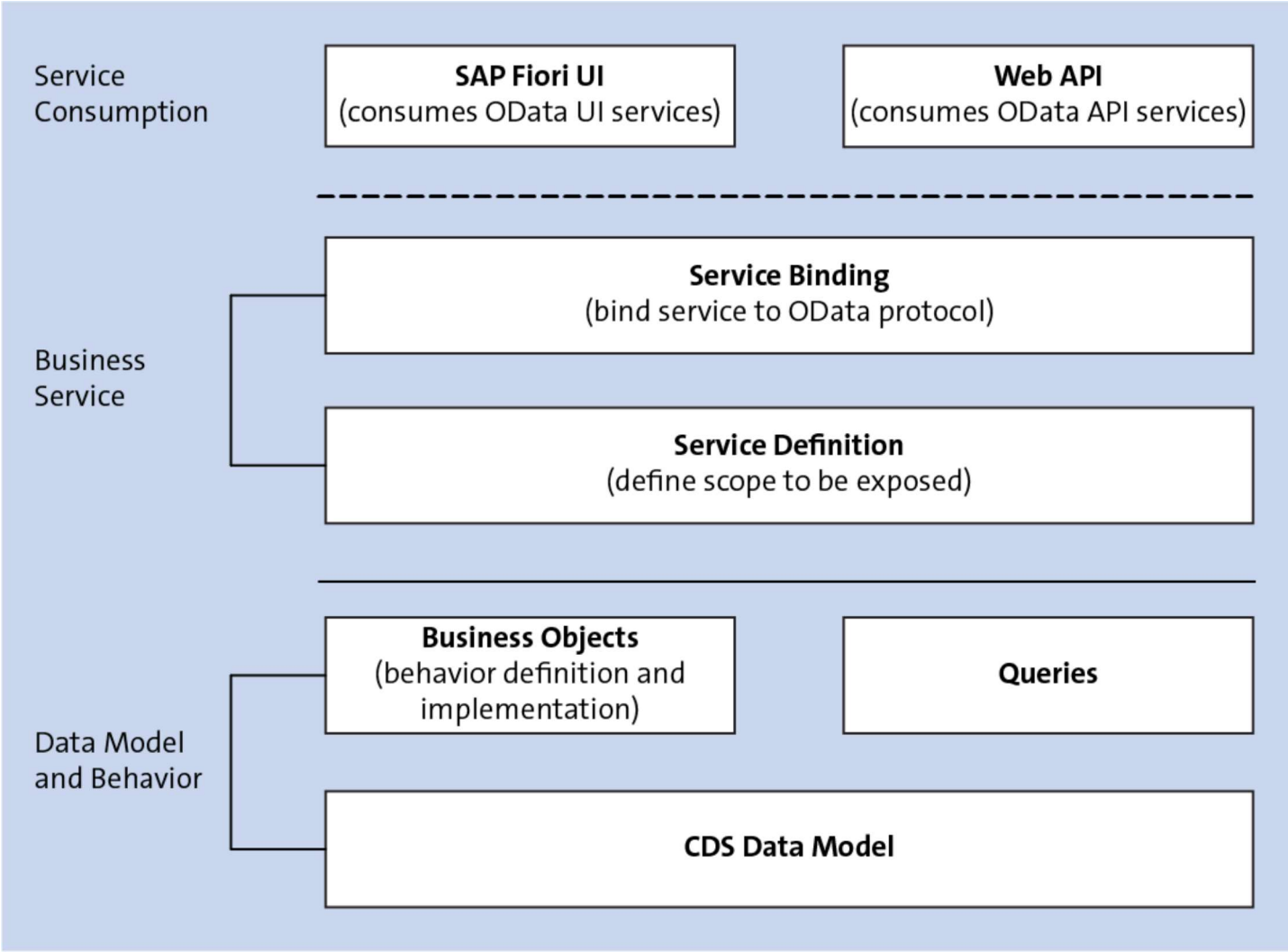
This meant code could be developed and utilized in the database itself. For companies running the new ERP solution, SAP S/4HANA, this meant a whole slew of new programming opportunities. There are a few things to know when programming ABAP on SAP HANA, but it is very doable and quite powerful.

During the early 2010s, many developers wondered if ABAP was to become increasingly obsolete as SAP acquired multiple cloud, non-ABAP-based solutions and pivoted existing products towards the cloud. But with the advent of SAP S/4HANA, and more importantly ABAP in the Cloud, the language was given new life, leading many to proclaim "ABAP's not dead." Programmers went into the second half of the decade with a clear idea of how to use ABAP to code for SAP S/4HANA, SAP Business Technology Platform, and more.

These new platforms led to the creation of additional ABAP programming models. The first, the **ABAP programming model for SAP Fiori**, is used when developing SAP HANA-optimized OData services for SAP Fiori applications. These are based on core data services views and cover three application scenarios: analytical, transaction, and search.



The **ABAP RESTful programming model** is a very new paradigm based on the model for SAP S/4HANA, but eschews Business Object Processing Framework (BOPF) in place of a more advanced concept.



(Back to ToC.)

New vs. Old ABAP

When working with a seasoned ABAP developer or scouring SAP Community blogs you may come across the terms “new,” “modern,” “old,” “classic,” “old-fashioned,” or “legacy” ABAP. What all does this mean?

Simply put, ABAP programming techniques that were used from the 1980s through April 2000 is typically what developers mean when they refer to “old,” “old-fashioned,” “classic,” or “legacy” ABAP. The advent of object-oriented programming in ABAP and the change in syntax that arrived with release 7.4 unofficially mark the changeover to “new” or “modern” ABAP.

(Back to ToC.)

When to Use ABAP

All SAP solutions—from R/1 through SAP S/4HANA—can be modified with ABAP code. While some solutions, such as SAP Business One, SAP Ariba, and acquired products such as SAP Concur and SAP SuccessFactors, run primarily on other languages, ABAP will still play a role when these solutions interface with a central, ABAP-based SAP ERP or SAP S/4HANA system.

(Back to ToC.)

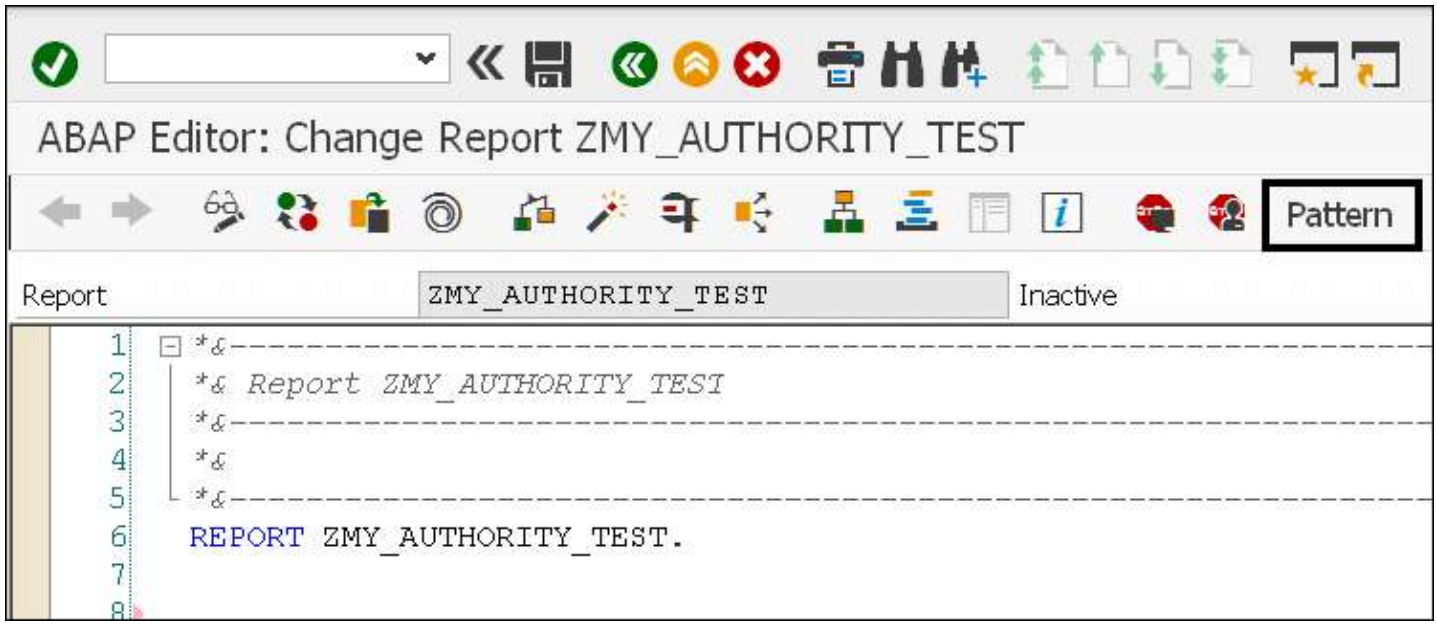
The ABAP Workbench

There are half a dozen important tools that programmers can use when working with ABAP code. They can be found in a development environment named the **ABAP Workbench**. This environment contains a number of needed development tools, the most commonly used of which accessible through the Object Navigator. You can access the Object Navigator with t-code SE80.

Here's a breakdown of these key ABAP Workbench tools:

ABAP Editor

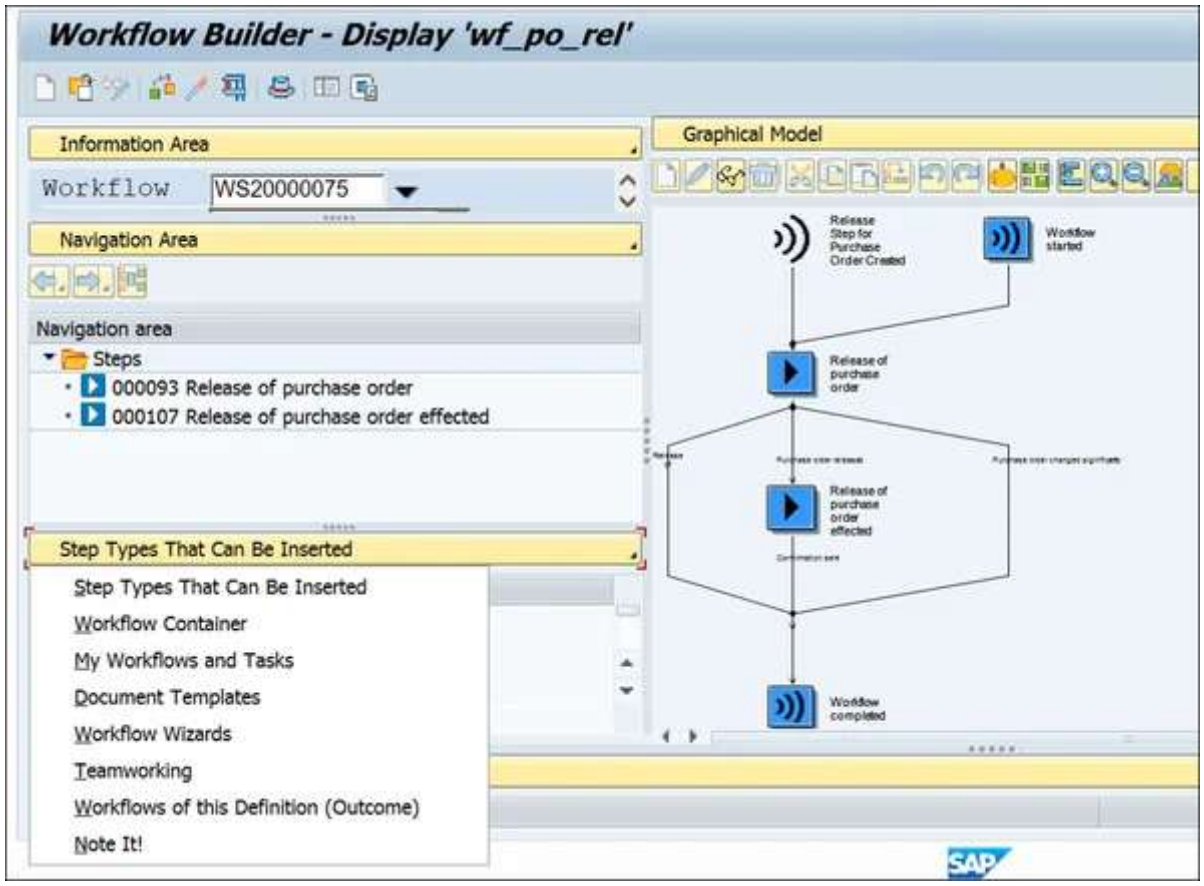
ABAP coding can be done in a special tool called the ABAP Editor, which has three different modes to work within—two versions of the Front-End Editor, and the Back-End Editor. The three editors are fully compatible and interchangeable. The source code created in one editor can be viewed by all other modes.



You can access the ABAP Editor with t-code SE38.

ABAP Dictionary

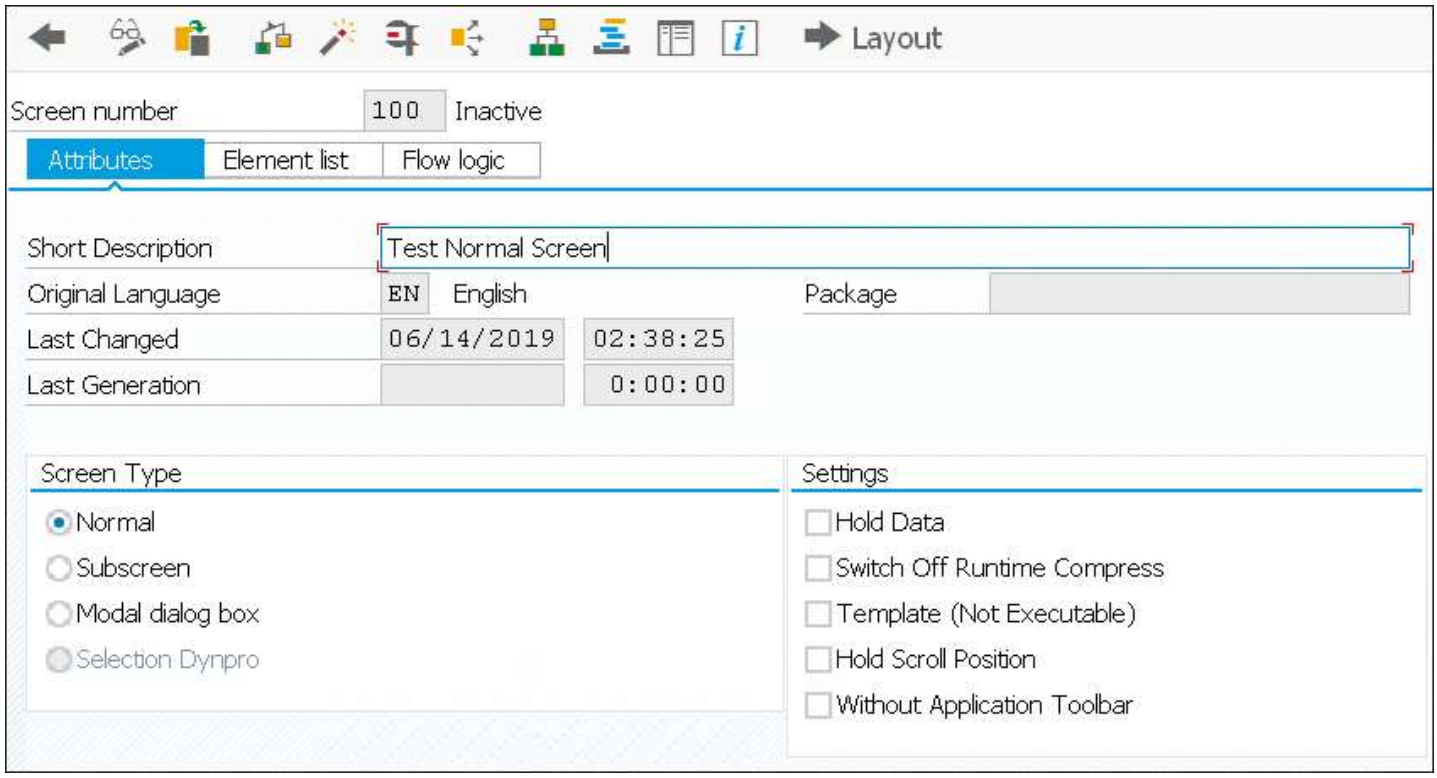
Also called the Data Dictionary, DDIC, or sometimes just “Dictionary,” this is a system-wide repository where database objects such as domains, data elements, and transparent tables are created and maintained. Programs will query the ABAP Dictionary to ensure that all sides are working with a single definition of an object.



You can access the ABAP Dictionary with t-code SE11.

ABAP Painter

The ABAP Painter is a set of two tools used to create GUI statuses and dynpros. The **Menu Painter** creates the GUI status and components, while the **Screen Painter** creates dynpros via text and screen editors.



You can access the Menu Painter with t-code SE41 and the Screen Painter with t-code SE51.

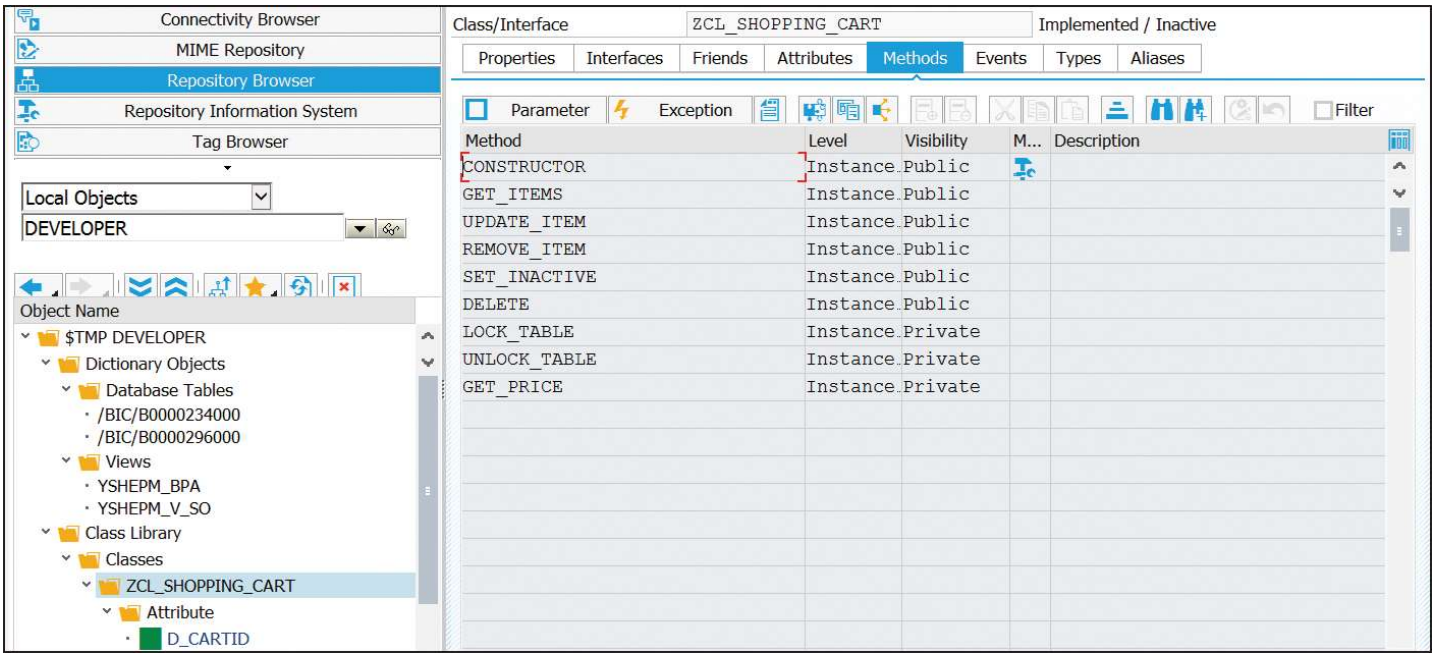
Function Builder

The Function Builder is a tool that can create and maintain function modules. These are universal procedures that start with **FUNCTION** and end with **ENDFUNCTION**.

You can access the Function Builder with t-code SE37.

Class Builder

The Class Builder is a specialized tool that creates and maintains class pools. A class pool is a repository object that stores global classes along with related definitions that will help the program implement the class.



You can access the Class builder with t-code SE24.

Web Application Builder

The Web Application Builder is a tool that allows programmers to create web applications.

You can access the Web Application Builder by following this menu path: **Create > BSP Library > BSP Application**.

(Back to ToC.)

Other Key ABAP Terms

While we've laid out much of the important terminology you'll run into when working with ABAP, there are a handful more that will be helpful to you. Let's take a look at twelve such terms and concepts.

- **ABAP Debugger**: A tool for performing functional troubleshooting in programs.
- **ABAP Development Guidelines**: A set of general and ABAP-specific guidelines meant to help programmers creating applications with ABAP.
- **ABAP Development Tools**: A set of downloadable plugins that allow programmers running Eclipse to develop ABAP. Formerly known as ABAP in Eclipse

or ABAP Development Tools in Eclipse.

- **ABAP Managed Database Procedures**: A way to execute complicated code inside a database via stored procedure. Specific to SAP HANA and its in-memory processing.
- **ABAP Objects**: The official name for OOP in ABAP.
- **ABAP Unit**: A testing tool used to check functions of code sections.
- **CDS Views**: Core data service Views allow programmers to take full advantage of the SAP HANA database. They enhance information integration with cloud apps and other UIs via OData. These serve as the basis of SAP Fiori apps.
- **Design Patterns**: Tried and true solutions to common software requirements that can be "recycled" and used as the basis of a new program.
- **Repository Information System**: A source used to search repository objects; accessible via the Object Navigator or t-code SE15.
- **SAP GUI (SAP Graphical User Interface)**: The interface in the presentation layer of applications built with ABAP code that run on the desktop as opposed to in a browser.
- **SAP NetWeaver AS ABAP**: Part of a client server which allows for creating ABAP programs and consists of at least three layers, including the presentation, application, and database layers. The **AS** stands for **Application Server**.
- **Two-Track Method Development**: The simultaneous development of two versions of a method. An example of which might be implementing a method using both ABAP and AMDP.

(Back to ToC.)

Additional Resources

Feel you need more help learning ABAP programming? These books, blog posts, and videos will help, as well as this ABAP learning journey.

Blog Posts

- "3 Advanced ABAP Testing Things to Know"