Comprehensive SAP Demo Programs Reference for ABAPers

A Complete, Humanized, and Practical Guide to Learning from SAP's Own Example Programs

---

1. Introduction – Why SAP Demo Programs Matter

Every ABAP system contains a wealth of knowledge hidden in plain sight — the **SAP demo programs**. These are not theoretical samples or internet tutorials; they are **authentic examples built and shipped by SAP engineers themselves**.

When you explore demo programs, you are effectively learning directly from SAP's core development team — studying how they structure code, name objects, use frameworks, and apply new syntax.

The demo repository acts as:

- A **live coding library** to test and learn ABAP features.
- A **reference base** for project design patterns.
- A **teaching tool** for training teams and onboarding new developers.
- A **troubleshooting reference** when your code behaves differently from expected SAP logic.

Knowing how to explore and interpret these demos separates a capable ABAPer from a seasoned expert.

---

2. How SAP Structures Its Demo Programs

SAP organizes its demo programs using naming conventions and package structures that follow consistent patterns. Recognizing these patterns helps you search efficiently.

Prefix	Purpose / Area	Description
DEMO_*	Core ABAP syntax and features	These are the modern, official examples.
BCALV_*	ALV Grid demos	Classic ALV examples showing reports, events, and editable grids.
SALV_*	Simple ALV Object (OO) demos	Object-oriented ALV reports using `CL_SALV_TABLE`.
RSDEMO_*	Legacy R/3 demos	Older ABAP examples used in early releases.
SABAPDEMOS*	Main demo package	Central package that groups all demo reports, classes, and DDIC objects.

smart varghese smart varghese smart varghese

SABAPOOP*	Object-Oriented Programming	Focused on ABAP Objects, inheritance, and polymorphism.
SABAPEXPRESSIONS*	Modern syntax	Inline declarations, `COND`, `SWITCH`, and table expressions.
SABAPDEMOS_RAP	RAP demos	RESTful ABAP Programming Model examples.
SABAPDEMOS_CDS	CDS view demos	Examples of annotations, joins, and associations.
SABAPDEMOS_AMDP	AMDP demos	Database procedures and SQLScript examples.

---

3. Discovering Demo Programs in the System

3.1 Using SE38 (ABAP Editor)

1. Go to **SE38**.
2. In *Program Name*, enter a wildcard such as `DEMO_*`.
3. Press **F4 Help**.
4. Choose "Programs containing pattern".
5. Execute to get a full list of demo programs.
6. Double-click on any program → *Display → Execute (F8)* to run.

This method is best for quickly locating specific ABAP examples such as internal tables, loops, or string operations.

3.2 Using SE80 (Object Navigator)

1. Open **SE80**.
2. From the dropdown, select "Program".
3. Type in a prefix (like `SALV*` or `BCALV*`).
4. Press Enter to see all matching demo programs.
5. You can then explore the includes, GUI status, and dynpros if any.

SE80 helps you explore the architecture of larger demos like ALV or screen-based examples.

3.3 Using SE24 (Class Builder)

SAP provides demo classes that represent object-oriented patterns.

1. Go to **SE24**.
2. Enter `CL_DEMO_*` → press **F4 Help**.
3. Select a class like `CL_DEMO_OUTPUT` or `CL_DEMO_EXPRESSIONS`.
4. Open the *Methods* tab to see individual executable examples.

smart varghese smart varghese smart varghese

Most of these classes have a `RUN`, `EXECUTE`, or `MAIN` method that you can execute directly.

3.4 Using SE37 (Function Builder)

To see function module-based demos:

1. Open **SE37**.
2. Enter `DEMO_*`.
3. Press **F4 Help** to browse function modules.
4. Examples include `DEMO_READ_TEXT`, `DEMO_FUNCTION_MODULE_CALL`, and `DEMO_DATE_CONVERSION`.

Each demo function module is executable and demonstrates modular coding.

3.5 Using ABAP Keyword Documentation

Every ABAP keyword comes with examples directly linked to demo programs.

In **ADT (Eclipse)**:

1. Place the cursor on a keyword (e.g., `SELECT`, `LOOP`, `COND`).
2. Press **F1**.
3. In the documentation window, scroll to **Example**.
4. Click **Open Example in Project** to load the demo into your workspace.

This integration is extremely powerful — it connects theory with working, debuggable code.

---

4. Exploring Demo Packages

The SAP system contains structured demo packages. You can navigate them through **SE80 → Package**.

Package	Description
SABAPDEMOS	Primary repository of ABAP demo programs.
SABAPDOCU	Documentation and code examples used in help content.
SABAPDIC	Data dictionary demos for tables, views, and domains.
SABAPOOP	Object-oriented ABAP learning materials.
SABAPEXPRESSIONS	Inline syntax and expression-based examples.
SABAPDEMOS_RAP	RAP BO and OData demos.
SABAPDEMOS_AMDP	AMDP methods and SQLScript examples.
SABAPDEMOS_CDS	CDS annotations and joins examples.

smart varghese smart varghese smart varghese

Each package is a curated learning library covering a distinct layer of ABAP technology.

---

5. Core Demo Programs to Study

Domain	Program	What It Demonstrates	
Internal Tables	`DEMO_TABLES`	Declaring, appending, and reading internal tables.	
Inline Declarations	`DEMO_INLINE_DECLARATIONS`	Modern syntax for local variable declarations.	
String Templates	`DEMO_STRING_TEMPLATE`	Using `	` for dynamic string concatenation.
Expressions	`DEMO_EXPRESSIONS`	`COND`, `SWITCH`, `REDUCE` examples.	
RTTS / RTTC	`DEMO_RTTI`	Dynamic type creation and inspection.	
ALV Grid (Classic)	`BCALV_GRID_DEMO`	Using `REUSE_ALV_GRID_DISPLAY`.	
ALV OO (SALV)	`SALV_DEMO_TABLE`	`CL_SALV_TABLE` based display.	
File Handling	`DEMO_OPEN_DATASET`	Read/write files on the application server.	
Regular Expressions	`DEMO_REGEX_MATCH`	Pattern searching with regex.	
JSON Serialization	`DEMO_JSON_SERIALIZE`	Convert ABAP data to JSON and back.	
Exception Handling	`DEMO_CX_STATIC_CHECK`	Checked and unchecked exceptions.	
Parallel Processing	`DEMO_PARALLEL_PROCESSING`	Using background RFCs.	
CDS / RAP	`SABAPDEMOS_RAP_TRAVEL`	Full RAP business object example.	

Each of these programs represents a well-documented, executable reference that you can adapt into your own project templates.

smart varghese smart varghese smart varghese

---

6. Using Demo Classes for Object-Oriented Learning

Class	Focus Area	What You Learn
CL_DEMO_OUTPUT	Output Formatting	Render formatted HTML-like output for reports.

CL_DEMO_EXPRESSIONS	Inline Syntax	Use of modern operators and expressions.
CL_DEMO_CALCULATOR	Inheritance	Demonstrates virtual and redefined methods.
CL_DEMO_FILE	File I/O	File handling operations.
CL_DEMO_TABLE	Internal Table Manipulation	Dynamic operations, filters, and aggregations.
CL_DEMO_JSON	JSON Conversion	Serialize and deserialize JSON structures.
CL_DEMO_RTTI	Runtime Type Information	Demonstrates RTTS concepts.

These classes provide working models of encapsulated design and serve as excellent study material for ABAP OO.

---

7. Real-World Usage: SALV OO Example

1. Open `SALV_DEMO_TABLE` in SE38.
2. Execute and observe how the table data displays in ALV format.
3. Review the source code:
   - Data preparation
   - Factory method:
   - `cl_salv_table=>factory(`
   - `  IMPORTING r_salv_table = DATA(lo_alv)`
   - `  CHANGING  t_table      = lt_data`
     `).`

   - Layout, display, and function settings
4. Copy the program to `ZSALV_DEMO_TABLE`.
5. Replace the sample table with your data and execute.

You now have a ready ALV framework that follows SAP's design principles.

---

8. Working with Demos in ADT (Eclipse)

ADT provides enhanced demo handling:

- Use **Ctrl + F1** to open documentation.
- Import examples into your workspace.
- Debug, refactor, and adapt demos safely in your own Z-package.

For RAP and CDS, ADT-based demos are far more detailed and better documented than GUI-based examples.

smart varghese smart varghese smart varghese

---

9. Practical Usage Guidelines

1. **Never modify SAP demos directly.** Always copy to your own namespace (e.g., `ZDEMO_*`).
2. **Study the comments carefully.** Most demos explain each step in plain text.
3. **Debug interactively.** Observing variable changes provides deeper understanding.
4. **Use as code patterns, not production-ready logic.** These demos illustrate features, not performance tuning.
5. **Build your own index.** Create a personal or team-wide catalog of useful demo programs for quick reference.
6. **Stay version-aware.** New demos appear with each S/4HANA or ABAP release.

---

10. Expert Tips to Maximize Learning

- Use **package-level browsing** to explore related artifacts (classes, tables, CDS).
- Combine demos with **ABAP Keyword Documentation** to correlate syntax and results.
- Integrate demos in your team's knowledge sessions.
- Keep your own "ZDEMO_LIBRARY" package where you store copied and enhanced versions.
- Before using a new ABAP keyword, always run the SAP demo for it — this ensures accuracy.

---

11. Why Demos Are Better Than Internet Code Samples

While blogs and forums show working examples, SAP demo programs are:

- **Authoritative** – written by SAP's internal developers.
- **Version-consistent** – aligned with your current ABAP release.
- **Executable** – run and debug directly in your environment.
- **Comprehensive** – cover syntax, data handling, UI, and database integration.

They serve as **trusted references** for both learning and quality assurance.

---

12. Summary

Demo programs are the most direct way to learn how SAP intends ABAP to be written. They teach syntax, design, and framework usage in a consistent, tested, and maintainable form.

By regularly exploring demos:

smart varghese smart varghese smart varghese

- You internalize SAP's own coding standards.
- You reduce trial-and-error during development.
- You enhance your understanding of both classical and modern ABAP paradigms.

A developer who can interpret and extend demo programs becomes an **independent learner and solution architect** within the SAP landscape.

Recommendation:
Start exploring these packages in this sequence:

1. **SABAPDEMOS** → Core syntax.
2. **SABAPEXPRESSIONS** → Inline and modern syntax.
3. **SABAPOOP** → Object-Oriented ABAP.
4. **SABAPDEMOS_CDS / AMDP / RAP** → Modern S/4HANA development.

Use these as a structured progression path from classical ABAP to full ABAP Cloud competence.

smart varghese smart varghese smart varghese