SPANBIX TRAINING INSTITUTE

SAP ABAP on S/4HANA — Job-Ready Course Outline

Designed by a 20+ Year SAP ABAP Practitioner · S/4HANA-Native · Beginner to Developer-Ready · 12 Weeks

COURSE AT A GLANCE

Duration: 12 Weeks (3 Months)	**Level:** Beginner → ABAP Developer-Ready
Live Hours: 84 hrs instruction + 60 hrs sandbox coding practice	**SAP Platform:** S/4HANA 2023 — live system, ADT (Eclipse) + SAP GUI
Capstone Projects: 3 (ALV Report, BAdI Enhancement, RAP Fiori App)	**Certification:** After all capstones + mock interview + TS document graded
Batch 1: Mon / Wed / Fri — 1 hr/class (timing flexible)	**Batch 2:** Tue / Thu / Sat — 1 hr/class (timing flexible)
Batch 3: Sat / Sun — 1.5 hrs/class (timing flexible)	**PD Classes:** Every Saturday & Sunday — resume + interview training

WHY THIS COURSE — THE SPANBIX ABAP ADVANTAGE

★ **S/4HANA-Native ABAP** Eclipse ADT, CDS Views, RAP — not legacy SE38-only teaching	★ **ABAP OOP from Day 1** Classes, interfaces, inheritance woven into every module
★ **CDS + OData + RAP** Modern app development stack that every client expects	★ **Live Project Training** Real enhancement scenarios: BAdI, user exit, VOFM routines
★ **Sandbox Access** Hands-on S/4HANA system — write & test code from Day 1	★ **Data Migration Coverage** BDC, LSMW, and file handling — Day 1 support task essentials
★ **Performance Tuning Module** SQL Trace, Runtime Analysis, short dump debugging	★ **Placement Assistance** Until you sign the offer letter — Spanbix own company network
★ **Functional Spec Writing** Write FS for ABAP developers — a skill that gets you hired	★ **Personality Development** Every Sat & Sun — resume, communication, mock interviews

12-WEEK JOB-READY CURRICULUM — SAP ABAP ON S/4HANA

Week	Mod	Topic	Key Content Covered
PHASE 1 — SAP & S/4HANA Foundation *Weeks 1–2*			
1	**M01**	**SAP & S/4HANA Fundamentals**	› ERP concept: what it is, why enterprises run it, major players › SAP evolution: R/1 → R/2 → R/3 → ECC → S/4HANA — key differences › S/4HANA architecture: in-memory DB (HANA), column store, ACDOCA universal journal › SAP system landscape: DEV → QAS → PRD, client concept, transport management › SAP GUI & Fiori Launchpad: login, navigation, transaction codes, multitasking › ABAP structure overview: workbench, repository, development objects › SAP Activate / ASAP methodology: phases, consultant's role in each phase › T-Codes: SE38, SE80, SE11, STMS, SE10
1–2	**M02**	**ABAP Basics & Data Dictionary**	› ABAP editor (SE38): creating, activating, and executing programs › ABAP/4 syntax: DATA, PARAMETERS, CONSTANTS — types and classification › Built-in data types: C, N, I, F, P, D, T, STRING, XSTRING — with use cases › Output statements: WRITE, ULINE, SKIP, MESSAGE; system fields (SY-*) › Control statements: IF/ELSEIF, CASE, DO, WHILE, LOOP — structured coding › String operations: CONCATENATE, SPLIT, REPLACE, FIND, string templates › ABAP Dictionary (SE11): Domains, Data Elements, Database Tables, Structures › Views: Database view, Projection view, Maintenance view, Join view › Search Helps (F4) and Lock Objects — data integrity essentials › Table Maintenance Generator: creating Z-table maintenance dialogs
PHASE 2 — Core ABAP Programming *Weeks 2–4*			
2–3	**M03**	**Internal Tables & Open SQL**	› Internal table types: Standard, Sorted, Hashed — when to use each › Declaring, populating, reading, modifying, and deleting from internal tables › Control break statements: AT FIRST / AT NEW / AT END OF / AT LAST › Open SQL: SELECT, INSERT, UPDATE, MODIFY, DELETE — full syntax › SELECT with WHERE, GROUP BY, ORDER BY, HAVING — practical exercises › Inner joins vs FOR ALL ENTRIES — performance implications (a key interview topic) › Field symbols: ASSIGN, FIELD-SYMBOLS — dynamic data access pattern › S/4HANA SQL: strict mode, SELECT SINGLE vs READ TABLE, deprecated syntax
3	**M04**	**Modularization Techniques**	› INCLUDE programs: TOP, I01, O01, F01 — module pool structure › Subroutines (FORM/PERFORM): passing by reference vs by value › Function Groups and Function Modules (SE37): parameters, exceptions, tables › ABAP Memory vs SAP Memory: GET/SET PARAMETER — data passing between programs › Message classes (SE91): message types A/E/W/I/S — correct usage in programs › Packages: creating packages, local objects vs transportable objects › Why modularization matters in a real project: reuse, testing, transport
3–4	**M05**	**Report Programming (Classical, Interactive & ALV)**	› Classical reports: TOP-OF-PAGE, END-OF-PAGE, RESERVE — print-ready output › Interactive reports: AT LINE-SELECTION, AT USER-COMMAND, HIDE, GET CURSOR › Hotspot, double-click drill-down — building an O2C drill-down report (SD flow) › Selection screens: PARAMETERS, SELECT-OPTIONS, SELECTION-SCREEN blocks › Dynamic screen modification: MODIF ID, screen-table fields (SCREEN-ACTIVE) › ALV using Function Modules: REUSE_ALV_GRID_DISPLAY — the industry standard › ALV OOP: CL_SALV_TABLE — clean, modern approach; field catalogue, layout › Variants: creating, saving, scheduling via background jobs (SM36) › Real exercise: write a billing report using VBRK/VBRP tables with ALV output
PHASE 3 — OOP, CDS Views & S/4HANA Development Stack *Weeks 4–6*			
4–5	**M06**	**Object-Oriented ABAP (OOP)**	› Why OOP: encapsulation, reuse, maintainability — and why S/4HANA mandates it › Classes (SE24): instance vs static methods and attributes, constructor › Inheritance: INHERITING FROM, method redefinition, abstract classes › Interfaces: INTERFACES keyword, polymorphism in practice › Exception classes: CX_ROOT, CX_DYNAMIC_CHECK, CX_STATIC_CHECK — raising and handling › Events: RAISE EVENT, SET HANDLER — event-driven programming pattern › Design patterns used in SAP: Factory, Singleton, Observer — real ABAP examples › Writing clean OOP code that a TCS/Accenture code review would approve

Week	Mod	Topic	Key Content Covered
5	**M07**	**CDS Views & Eclipse ADT**	› Why CDS views exist: push-down to HANA layer, code-to-data paradigm › ABAP Development Tools (ADT) in Eclipse: setup, project connection, navigation › Creating basic CDS views: @AbapCatalog, @EndUserText, DEFINE VIEW › CDS associations vs SQL JOIN — when each is appropriate › VDM (Virtual Data Model): Raw → Basic → Consumption view layers › CDS annotations: @Analytics, @UI, @Consumption — Fiori app enablement › CDS views with parameters, aggregations, currency conversion › Difference between CDS DDIC views (SE11) and CDS views — migration context › T-Codes/Tools: ADT (Eclipse), SE11, transaction DCLS for access controls
5–6	**M08**	**OData Services & RAP Framework**	› What is OData? Why Fiori apps consume OData — REST API fundamentals › Creating OData services via SEGW (Gateway Service Builder): entity sets, associations › OData on CDS: @OData.publish annotation — the S/4HANA way › RESTful ABAP Programming (RAP): managed vs unmanaged scenario › Behavior Definition (BDEF): create, update, delete, actions › Behavior Implementation (BILM class): handler class, saver class › Service Definition and Service Binding: publish as UI2 or Web API › BOPF framework: introduction for legacy projects still using BOPF › AMDP (ABAP Managed Database Procedures): calling HANA procedures from ABAP › Real exercise: build a simple Fiori Elements list report using RAP end-to-end
PHASE 4 — Enhancements, Forms & Output *Weeks 6–7*			
6–7	**M09**	**Enhancements & User Exits**	› Enhancement framework overview: why standard SAP code is never modified › User Exits (SMOD/CMOD): finding exits in a transaction, implementing in projects › Customer Exits: function module exits, menu exits, screen exits › BAdIs (Business Add-Ins): classic BAdI (SE18) vs new Enhancement Spot BAdI › Finding the right BAdI for a requirement: where-used, SPRO path › Implicit and explicit enhancements: source code plug-ins, pre/post-exits › VOFM routines (SD pricing): requirements, condition base value, condition value › Writing a Technical Specification (TS) for an enhancement — project deliverable › Transport request management for enhancements: SE10, release, STMS import
7	**M10**	**Forms: Smart Forms & Adobe Forms**	› SAPscript: structure, text elements, windows — legacy context only › Smart Forms (SMARTFORMS): layout, graphic management, paragraph/character formats › Writing a print program: calling a Smart Form function module from ABAP › Adobe Forms (SFP): why they replaced Smart Forms in modern projects › Adobe Form structure: layout set (PDF), interface, context mapping › Calling Adobe Forms from ABAP: FP_FUNCTION_MODULE_NAME, FP_JOB_OPEN/CLOSE › Output Management in S/4HANA: BRF+ rules replacing NACE — developer perspective › Real exercise: create an invoice form with dynamic line items using Adobe Forms
PHASE 5 — Interfaces, IDocs & Data Migration *Weeks 8–9*			
8	**M11**	**BAPIs, RFCs & ALE/IDocs**	› Remote Function Call (RFC): types — synchronous, asynchronous, transactional (tRFC) › RFC destination configuration (SM59): connecting systems for interfaces › BAPIs: what they are, BAPI_SALESORDER_CREATEFROMDAT2 example walkthrough › IDoc architecture: message types, basic types, segments, control record, data record › ALE configuration: logical systems, distribution model (BD64), partner profiles (WE20) › Inbound IDoc processing: ORDERS05 → sales order; function module + posting › Outbound IDoc processing: INVOIC02 output from billing › IDoc monitoring and error handling: BD87, WE05, MONI — Day-1 support task › Workflow basics: work items, tasks, SAP Business Workplace — awareness level
8–9	**M12**	**Data Migration: BDC & LSMW**	› Data migration strategy in SAP projects: when to use BDC vs LSMW vs BAPI › BDC recording method: SHDB transaction — capturing field names and sequence › Call Transaction method: synchronous BDC, error handling, log review › Session method: asynchronous BDC for large data volumes — SM35 monitoring › Handling table controls in BDC: index-based loop logic › LSMW (LSMW transaction): 4 object types, file structure, field mapping, conversion rules › Flat file creation: CSV/TXT format requirements, character encoding considerations › Uploading and verifying migrated data — cutover checklist for go-live › File handling in ABAP: Application Server (OPEN DATASET) vs Presentation Server (GUI_UPLOAD)
PHASE 6 — Dialog Programming, Performance & Debugging *Weeks 9–10*			

Week	Mod	Topic	Key Content Covered
9–10	**M13**	**Dialog Programming (Module Pool)**	› Screen Painter (SE51): layout editor, screen elements, attributes › Flow Logic: PBO (Process Before Output) and PAI (Process After Input) events › POV (Process On Value Request) and POH (Process On Help Request) › Table controls and step loops: tabular data entry in screen programs › Tabstrip controls and subscreens: multi-tab UI design › CALL SCREEN, SET SCREEN, LEAVE SCREEN — screen navigation logic › Menu Painter (SE41): GUI status, GUI title, function keys assignment › Transaction code creation (SE93): linking dialog program to a T-code › Real exercise: build a Z-transaction for material master data entry
10	**M14**	**Performance Tuning & Debugging**	› ABAP Debugger (classic + new): breakpoints, watchpoints, field display › Static vs dynamic breakpoints — using /h for production debugging › Changing internal table contents live in debugger — support project technique › Short dump analysis (ST22): reading ABAP runtime errors, fixing root cause › SQL Trace (ST05): identifying table buffer bypass, missing indexes, slow SELECTs › Runtime Analysis (SE30 / SAT): identifying performance bottlenecks in programs › Top 10 ABAP performance rules: SELECT *, FOR ALL ENTRIES pitfalls, buffering › ABAP Unit Testing basics: CREATE OBJECT for test class, assert methods › Code Inspector (SCI) and ATC (ABAP Test Cockpit): S/4HANA strict mode checks
PHASE 7 — Project Lifecycle, Consulting Skills & Placement *Weeks 11–12*			
11	**M15**	**Functional Spec, Auth & Project Lifecycle**	› Full SAP project lifecycle: Prepare → Explore (BBP) → Realize → Deploy → Run › AS-IS and TO-BE documentation — writing your section as an ABAP developer › Functional Specification (FS) document: structure, logic tables, field mapping › Reading an FS and translating into technical design — the developer's job › Technical Specification (TS): program logic, database changes, test cases › Authorization Management: SU01, roles (PFCG), profiles, authorization objects › Transport requests (SE10): creating, adding objects, releasing, STMS import queue › Cutover activities: Z-table data, number ranges, master data upload sequence › Ticket handling: priority classification (P1–P4), SLA, root cause analysis format
11–12	**M16**	**Interview Prep & Capstone Project**	› Top 60 SAP ABAP interview questions — fresher and 2-year experience pattern › Deep-dive Q&A: internal tables, CDS views, BAdI vs user exit, RAP, OOP › CAPSTONE PROJECT 1: write an ALV report pulling SD/MM data with drill-down › CAPSTONE PROJECT 2: implement a BAdI enhancement for a pricing requirement › CAPSTONE PROJECT 3: end-to-end RAP app — CDS → Behavior → Fiori Elements › Resume writing: how to present sandbox projects, GitHub, TS documents › Mock interviews: video-recorded with panel feedback (2 rounds minimum) › Post-placement: what your first week on a project looks like — real war stories › Personality development, communication & client-facing consulting skills

RECOMMENDED DELIVERY MODEL

SAP System: S/4HANA 2023 — mandatory. ECC supplementary reference only.	**Dev Tools:** Eclipse ADT for CDS/RAP; SAP GUI for SE38, SE11, SE24, SM59
Instructor: 8+ years SAP ABAP project experience minimum	**Guest Sessions:** Active ABAP developers from Tier-1 SIs (TCS, Capgemini, Infosys)
Mock Interviews: Minimum 2 sessions per student, Weeks 11–12, video-recorded + feedback	**Placement Support:** Continuous until offer letter signed — Spanbix own company network

Enroll Now — Limited Seats per Batch

WhatsApp: +91 93107 93790 | www.spanbix.com | @spanbix93 | Greater Noida