

BMC AMI DevX Code Debug for TSO 17: Getting Started Using ISPF (WBT)

[Learning Path >](#)

Course Code: CPCD-GSIS-U1700

Modality

Web-based Training (WBT)

Duration

4 Hours

Applicable Version(s)BMC AMI DevX Code Debug
for TSO 17.02**Target Audience**

- Application Developers

Course Overview

BMC AMI DevX Code Debug for TSO is a source-level testing and debugging solution that helps developers test, diagnose, and resolve issues in mainframe applications. It supports interactive and batch debugging across multiple languages and environments, including COBOL, Assembler, PL/I, IMS, and Db2.

In this web-based course, you'll learn how to use BMC AMI DevX Code Debug for TSO 17.x to debug programs effectively from the ISPF interface. You'll start with the essential concepts and navigation, then build the foundation for source-level debugging by preparing your programs with the TSO Compile Facility and DDIO.

Next, you'll set up and run both interactive and batch debugging sessions by working with profiles and execution environments, including Batch Connect and Multi-Batch options. You will learn how to take control of execution using breakpoints, stepping, tracing, and other flow-control commands, and you'll learn how to inspect and update data while the program runs.

Finally, you will learn techniques to modify program logic during a debug session, use Review mode to analyze execution paths, and apply coverage tools to validate what ran.

In addition to English, the textual content of this course is available in Chinese (Simplified), French, German, Italian, Japanese, Korean, Portuguese (Brazil), and Spanish (Mexico).

Prerequisites

- BMC AMI DevX Code Debug 17.x: Fundamentals Concepts (WBT)

Recommended Knowledge/Skills

- NA

Learning Objectives

- Explain Code Debug for TSO concepts, capabilities, supported languages and applications, and interactive versus batch modes
- Navigate the Code Debug ISPF interface and configure a standard test environment
- Prepare programs for source-level debugging using the TSO Compile Facility and DDIO
- Configure and run interactive and batch debugging sessions using profiles and execution environments
- Control program execution using breakpoints, stepping, tracing, and flow-control commands
- Inspect and modify program data during execution
- Analyze program logic using dynamic analysis features and Review mode
- Modify execution flow using SKIP, GOTO, MOVE, and INSERT commands

Course Modules

Module 1: Getting Started with BMC AMI DevX Code Debug for TSO

- Describe BMC AMI DevX Code Debug for TSO at a high level
- Identify key capabilities and benefits of Code Debug TSO
- Identify supported languages and application types for debugging
- Explain interactive and batch debugging modes in Code Debug TSO
- Identify inputs and outputs used during a Code Debug TSO session
- Explain the high-level debugging workflow using Code Debug TSO
- Navigate the Code Debug TSO user interface
- Use the PRIMARY MENU and its key options
- Use the SETUP MENU and its configuration options
- Use the ENVIRONMENTS MENU to select supported execution environments
- Configure the STANDARD test environment and key fields
- Navigate the SOURCE panel and its key sections
- Access the Utilities menu and available utility functions
- Explain breakpoint concepts and execution control
- Use PF key assignments and navigation support
- Use command entry and processing methods in Code Debug

Module 2: Compile the Program Using TSO Compile Facility

- Explain how to prepare programs for source-level debugging
- Describe the precompile, compile, and link-edit process using Code Debug TSO
- Explain the role of BMC AMI Common Shared Services (CSS) in program preparation
- Navigate the Program Preparation Menu
- Use the CreateJCL Menu options
- Convert existing compile JCL to use the Language Processor (LP)
- Create new compile JCL using the Compile Facility
- Explain CSS Preprocessor and Postprocessor concepts and use cases
- Generate and store symbolic information for debugging
- Explain the DDIO file concept and its role in debugging
- Use DDIO File Facility utilities
- Describe the DDIO file structure (shared directory and database relationship)
- Create a DDIO file
- Prepare programs for interactive source-level debugging

Module 3: Starting a Code Debug TSO Debugging Session

- Describe Code Debug TSO debugging capabilities at a high level
- Start an interactive debugging session using ISPF
- Use profiles to store and manage debugging session information
- Select and configure execution environments
- Navigate and use the Code Debug TSO SOURCE panel

- Execute and control programs using Code Debug TSO commands
- Debug batch programs using the Batch Connect Facility
- Convert and submit execution JCL for batch debugging
- Select job steps for interactive and unattended execution
- Connect to and debug intercepted batch jobs
- Debug multiple jobs using the Multi-Batch Facility
- Manually modify execution JCL to enable debugging
- Understand temporary data sets created during JCL conversion

Module 4: Controlling Program Execution

- Describe program execution control in BMC AMI DevX Code Debug for TSO and IMS
- Identify execution states (running, paused, stopped, and abended)
- Set unconditional breakpoints to gain control of program execution
- Set and manage BEFORE and AFTER breakpoints
- Explain breakpoint behavior across statements, paragraphs, modules, and nested programs
- Use the GO command to start, resume, and control execution flow
- Step through code using GO n, GO 1, and GO HALF
- Apply conditional breakpoints to control execution based on logic and events
- Use WHEN to pause execution based on variable conditions
- Use PAUSE to temporarily suspend execution within inserted logic
- Trace program execution using TRACE and control trace behavior
- Navigate execution flow using GOTO and RUNTO
- Interpret execution flow indicators, system messages, and session output
- Complete an end-to-end debugging walkthrough using execution-control commands

Module 5: Inspecting Program Data

- Inspect and interpret program data during execution
- Understand the Keep window and Automatic Keep behavior
- Use the KEEP command to continuously monitor variable values
- Use the PEEK command and its variants (PEEK, PEEKE, PEEKH)
- Manage and modify variable values during a debug session
- Control session logging for KEEP, PEEK, and Automatic Keep entries
- Examine data sets allocated during a Code Debug TSO test session
- Use the SHOW ALLOCATE command to verify allocated files
- Use the LIST ALLOCATES utility to review and manage allocations
- Browse data sets during debugging using the BROWSE command

Module 6: Analyzing and Modifying Program Logic

- Modify program logic during debugging using BMC AMI DevX Code Debug for TSO
- Use SKIP commands to bypass single statements and blocks of code
- Redirect execution flow using the GOTO command
- Change data values using the MOVE command to influence program behavior
- Insert statements dynamically during debugging using the INSERT facility
- Analyze program structure and control flow using dynamic analysis features
- Locate data definitions, usage, and modifications using the FIND command and keywords
- Understand data relationships through aliases and group data structures
- Monitor program execution paths using Review mode
- Trace and reverse execution using MONITOR, REVERSE, and RESUME commands
- Evaluate statement-level execution coverage using COUNT and SHOW COUNT

Course Assessment

Discount Options

Have multiple students? Contact us to discuss hosting a private class for your organization.

Contact us for additional information 