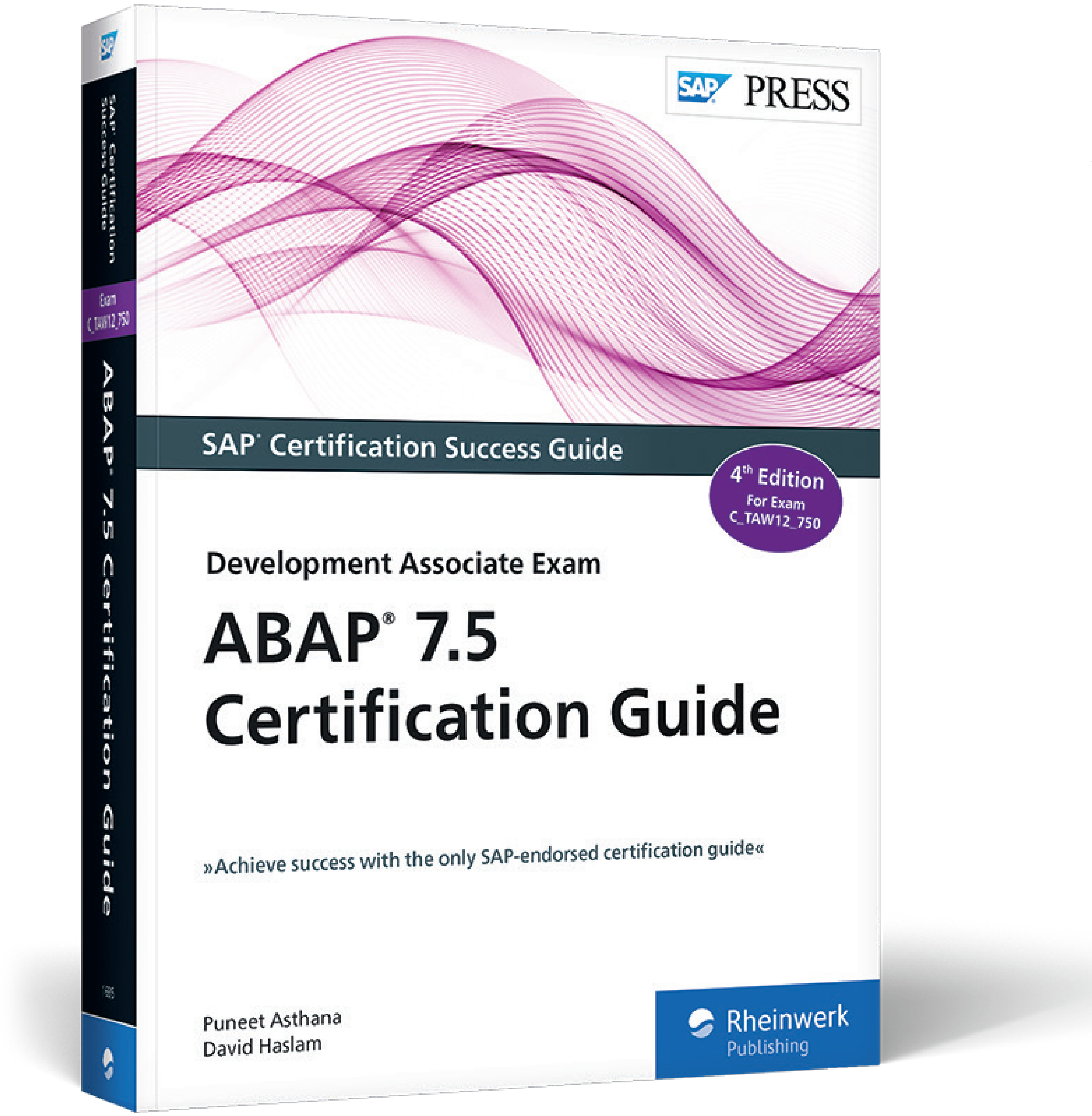

SAP PRESS
SAP Certification Success Guide
4th Edition For Exam C_TAW12_750
Development Associate Exam
ABAP® 7.5 Certification Guide
»Achieve success with the only SAP-endorsed certification guide«
Puneet Asthana
David Haslam
Rheinwerk Publishing

Reading Sample

In this chapter, authors Puneet Asthana and David Haslam review ABAP Workbench usage, including the repository browser, the ABAP Editor, the transport organizer, and more. Once you're comfortable with these key concepts, test your knowledge with practice questions designed to help your ace your upcoming ABAP certification exam!

"ABAP Workbench Usage"

Contents

Index

The Authors

Puneet Asthana, David Haslam

ABAP 7.5 Certification Guide: Development Associate Exam

636 Pages, 2018, $69.95
ISBN 978-1-4932-1685-7

www.sap-press.com/4605

Chapter 4
ABAP Workbench Usage

Techniques You'll Master:

- Describe the features and capabilities of the ABAP Workbench
- Explain the structure of the Object Navigator
- Use the Repository Browser to edit repository objects
- Access the Repository Information System to search for repository objects
- Understand the Enhancement Information System and its use
- Work with the ABAP Workbench tools
- Identify the Front-End Editor and the various settings to improve productivity
- Determine the concept of the Development Package and the Transport Organizer

The ABAP Workbench is the development environment for the SAP NetWeaver Application Server ABAP and is available in every ABAP-based SAP system. The ABAP Workbench provides the development tool needed for application development. The Object Navigator, the central entry point into the ABAP Workbench, contains the most commonly used development tools and is the recommended development environment.

In this chapter, we will cover the ABAP Workbench tool and the Object Navigator in detail. We will explain the features and capabilities of the ABAP Workbench, discuss the structure of the Object Navigator, and discuss the various browsers available in the Object Navigator. We will cover the Front-End Editor, the Repository Information System, the Enhancement Information System, and ABAP Workbench tools such as the ABAP Editor, Screen Painter, Menu Painter, ABAP Dictionary, Class Builder, Function Builder, and Web Application Builder. Finally, we will discuss packages and their attributes and the Transport Organizer.

Real-World Scenario

You have started on a customer project as an SAP development lead and have been asked to explain SAP development tools and the various features of the ABAP Workbench and the Transport Organizer to the project team.

As a development lead, you have to organize the technical development for the project. Your job is to define the development standard and the tools to be used for the various developments. You also have to define the development project, package, system landscape, and transport strategy for all of the developments in the project.

Objectives of this Portion of the Test

One of the objectives of the ABAP certification examination is to verify your knowledge of the ABAP Workbench and the various tools associated with it. You need to understand the following to be a successful ABAP developer:

- ABAP Workbench and development tools
- ABAP Workbench settings
- The use of various browsers in the ABAP Workbench
- The Front-End Editor and the settings to improve productivity
- Development Packages and Transport Organizer

Key Concepts Refresher

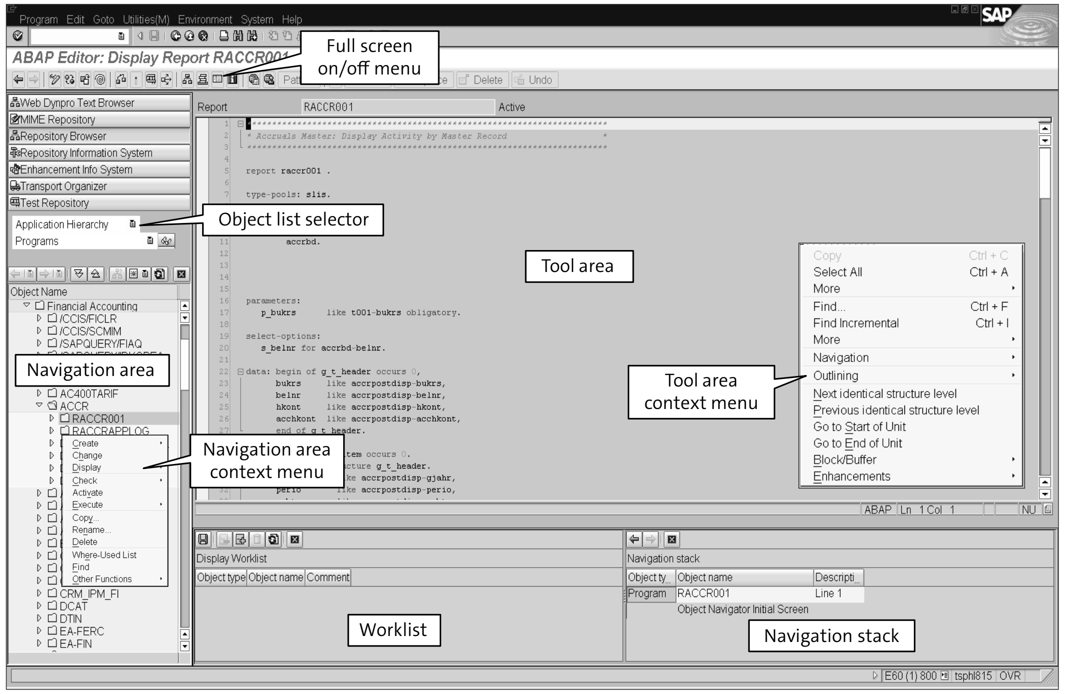
The ABAP Workbench is the integrated development environment for the application developer. The Workbench has most of the tools a developer needs to develop an application. Some of the most commonly used Workbench tools are:

- ABAP Editor
- Screen Painter
- Menu Painter
- ABAP Dictionary
- Web Dynpro development tools
- Package Builder and Transport Organizer
- Repository Browser
- Repository Information System
- Enhancement Information System

ABAP Workbench

ABAP stands for *Advanced Business Application Programming* and is the language in which most of the SAP business applications are written. The ABAP Workbench is the development environment for the ABAP developer. The ABAP Workbench consists of the development tools required for creating repository objects and can be started via Transaction SE80. The Object Navigator is the main entry point to the ABAP Workbench and can be accessed from the menu path **Overview • Object Navigator**.

The main screen of the ABAP Workbench is divided into the navigation area and the tool area (see Figure 4.1). The navigation area displays the object list as a hierarchy tree. The object list consists of all of the objects within an application area, package, program, global class, module pool, function group, and so on. The benefit of editing or displaying the object within the Object Navigator is that you can view all of the related objects for the program, class, module pool, or package as a tree structure and can access them from the navigation area by just double-clicking on the object. If you work with the repository object with the individual tool, then you only have the option to work with one type of object at a time with one tool.



```
Report RACCR001 Active
*********************************************************************
* Accruals Master: Display Activity by Master Record               *
*********************************************************************

report raccr001 .

type-pools: slis.

           accrbd.




parameters:
   p_bukrs      like t001-bukrs obligatory.

select-options:
   s_belnr for accrbd-belnr.

data: begin of g_t_header occurs 0,
        bukrs     like accrpostdisp-bukrs,
        belnr     like accrpostdisp-belnr,
        hkont     like accrpostdisp-hkont,
        acchkont  like accrpostdisp-acchkont,
      end of g_t_header.
```

Figure 4.1 ABAP Workbench Screen

The tool area is used to display or edit the repository object using the relevant tool editor. The navigation area or the tool area can be resized. You can hide the navigation area by clicking on the **Full Screen On/Off** icon on the application toolbar. The object navigation history with the ABAP Workbench is stored in the navigation stack. The Object Navigator has a menu option to display the navigation history. You can display the navigation window from the menu path **Utilities • Display Navigation Window** or by clicking on the **Display Navigation Window** icon, and an additional window for the navigation stack will be displayed under the tool area of the workbench. You can scroll back through the navigation stack by clicking on the blue left arrow or forward by clicking on the right arrow icon in the navigation stack window. This tool helps you scroll through the objects you have viewed during the logon session.

You can create your own worklist within the ABAP Workbench. A worklist is useful to help you manage the development objects on which you need to work. To create your worklist, display or edit the repository object first in the tool area of the ABAP Workbench. Then select **Utilities • Worklist • Insert Current Object** from the menu. This inserts the object in the worklist. You can display the worklist via the menu path **Utilities • Worklist • Display**, and an additional window for the worklist will be displayed under the tool area of the ABAP Workbench.

You can access the Reuse Library from the workbench menu path **Environment • Reuse Library**. The Reuse Library provides you with a set of reusable objects with example code and documentation. You can execute the code to display the result and copy the reusable code for your application development. The Reuse Library is displayed in the browser, and you can display the individual development objects within the Reuse Library by double-clicking on the product and reading the example code. You can also access the Reuse Library via Transaction SE83.

Similar to the Reuse Library, you can access the ABAP examples via the menu path **Environment • Example • ABAP Example**. The ABAP example programs are displayed in the browser, and you can display individual programs by double-clicking on the example program in the navigation area. The actual program is displayed in the tool area of the Object Navigator. ABAP examples demonstrate most of the keywords and syntax based on the ABAP documentation. You can also access the ABAP documentation from the ABAP examples screen. ABAP examples can also be displayed by executing Transaction ABAPDOCU.

You can display the context menu for an object within both the navigation area and the tool area by right-clicking on the object. The context menu offers only the functions that are relevant for the selected repository objects. You have the option to add your development objects to your favorites list within the Object Navigator by clicking on the **Favorites** icon in the navigation window.

You can access the individual tools to create the repository objects using individual transaction codes or access the most commonly used tools from the Object Navigator. The initial screen of the Object Navigator displays the Browser list on the top-left section of the screen. The Object Navigator has several browsers that you can use for various development needs. You have the option to navigate to the relevant tool based on the selected browser from the browser list (see Figure 4.2). You can add or remove browsers from the list via the menu path **Utilities • Settings**.

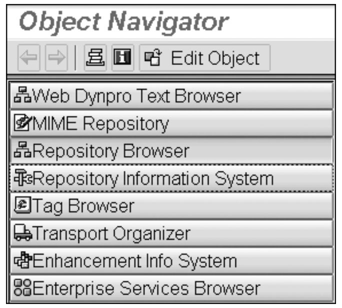


Figure 4.2 Object Navigator

The Object Navigator list has the following browser selection options in the object navigation:

- **Web Dynpro Text Browser**
  You use the Web Dynpro Text Browser to edit the text of the Web Dynpro UI elements that are created or managed in the Online Text Repository (OTR). This tool allows you to change OTR texts associated with Web Dynpro views that have been stored in the OTR or inserted from the OTR in the Web Dynpro application.
- **MIME Repository Browser**
  You use the MIME (Multipurpose Internet Mail Extensions) Repository Browser to browse or import MIME objects such as style sheets, icons, graphics, and so on from the ABAP Workbench. The MIME Repository stores these MIME objects in the SAP system. The MIME Repository Browser automatically creates a folder for BSP applications. You can import MIME objects from the Repository Browser or view the MIME objects by double-clicking on them.
- **Tag Browser**
  The Tag Browser provides the documentation of the user interface elements that can be used for web page development for BSP applications and Internet Transaction Server (ITS) templates. You have the option to filter the tags for the BSP application or the tags for the ITS templates. The Tag Browser provides documentation for HTMLB, BSP extension, BSP Directives, HTML, WML, and XHTML.
- **Test Repository Browser**
  You use the Test Repository Browser to manage test cases. You can display eCATT test scripts, manual test cases, and external test cases. You also have the option to create test cases within the Test Repository Browser. It's a good tool for you to manage the test cases within the Test Repository.

- **ABAP Unit Browser**
  This ABAP Unit testing tool for developers is included in SAP NetWeaver 7.0 EHP 2 and is integrated with the Object Navigator. The ABAP Unit tool provides the ABAP developer with an easy-to-use software verification and validation mechanism. It is highly integrated with the ABAP Workbench, which makes it easier for an ABAP developer to get accustomed to an ABAP Unit. Using the service method of class `cl_unit_assert`, you can check many conditions; the system diverges to the ABAP Unit result display if some condition is not fulfilled. In the result display, you can navigate to the position where the error occurred that caused the incorrect result. The tests are implemented in plain ABAP Objects as methods of local classes.

  The test classes are part of the program being tested. This can be an executable program, a module pool, a function group, or a class pool. The program being tested can include many test classes, and the test class can contain many methods. All unit tests belonging to a program being tested are organized in a test task. You can test whether the test condition or the modularization unit works properly by using `ASSERT` methods. For functions and methods with returning, exporting, or changing parameters, you simply test with the relevant parameter to determine if the modularization unit returns the expected value.
- **ATC Result Browser**
  ABAP Test Cockpit is a new integrated tool for quality checking of the ABAP repository objects and is available as of SAP NetWeaver 7.0 EHP 2 Support Package 2. The ATC Result Browser displays the findings of the ABAP Test Cockpit quality check run. The ATC Result Browser is integrated in Object Navigator, and it is used by the developer to resolve the issues reported in the ATC findings. The active ATC result is marked with a lightbulb icon and should be used by the developer to resolve the issues with the ABAP code. The analysis and the correction features of the ATC findings are displayed in the browser. The ATC browser opens with a **Quick Filter** option, and you can specify the filter criteria to filter the ATC findings. You can filter the objects for which you are the responsible developer and work on the list to resolve the issues with the code; you also have the option to specify the **Advance Filter** option and can switch between the Advance and Quick Filter options by clicking on the **Switch Filter** push button.

  You can double-click on the message in the browser to display the details and edit the program to resolve the reported issues or apply for exemption.

  The ABAP Test Cockpit is integrated with the ABAP development tools as well as ABAP Workbench, and can be used by developers to check their code or the content of the development package before migration of the code to the quality or test system. The ATC can be set up to check the content of the transport request automatically upon release of the transport request.

The ABAP Test Cockpit is a good tool for ABAP quality management and can be used by the quality assurance team to check the quality of the ABAP programs in the ABAP consolidation system or the test or quality system. Typically, your consolidation or quality system is the ATC Master system, in which you perform testing and quality checks. The ATC central check run can be performed from the ATC administration, Transaction ATC in the ATC Master system.

The quality manager schedules the regular central ATC check run in the ATC Master system to catch the problems with the ABAP repository objects. The quality manager then designates the ATC result as the active result and publishes it to the development system. The published result from central ATC quality check run will appear in the ATC Result Browser, and the developer should work on the active result to resolve the problem with the ABAP repository objects or request for exemptions.

ABAP Test Cockpit is also used to check functional correctness and performance optimization potential for SAP HANA migration.

Other browsers such as the Repository Browser, the Repository Information System, the Enhancement Information System, and the Transport Organizer are discussed in detail in the following sections.

Repository Browser

The Repository Browser is one of the menu options in the Object Navigator and is started by default when you execute the Object Navigator (Transaction SE80). You can create and edit repository objects in the Repository Browser. The Repository Browser is the central and most commonly used tool for managing your development objects within the Object Navigator.

When we talk about repository objects, we mean all of the SAP-delivered objects and customer-developed objects. These repository objects consist of programs, classes, ABAP Dictionary objects, function modules, screens, menus, and so on.

The object list displayed in the Repository Browser is a hierarchical tree structure. You can navigate to a repository object by the application hierarchy, whereby you can list the objects within each of the SAP application components. The objects can also be listed in a hierarchical tree structure within a package, program, class or interface, BSP application, Web Dynpro application, local object, and so on.

You can display the object from the navigation area by double-clicking on it. The object will be displayed in the tool area of the object browser in the editor that has been used to create the object. The ABAP development objects are displayed in the ABAP Editor, whereas the ABAP Dictionary objects are displayed in the ABAP Dictionary tool. Similarly, you display the screens and the menus in the tool area in the Screen Painter or Menu Painter by double-clicking on the screen or menu in the navigation area. You can use the context menus in the navigation area to create or edit the object in the application area or package (see Figure 4.3).

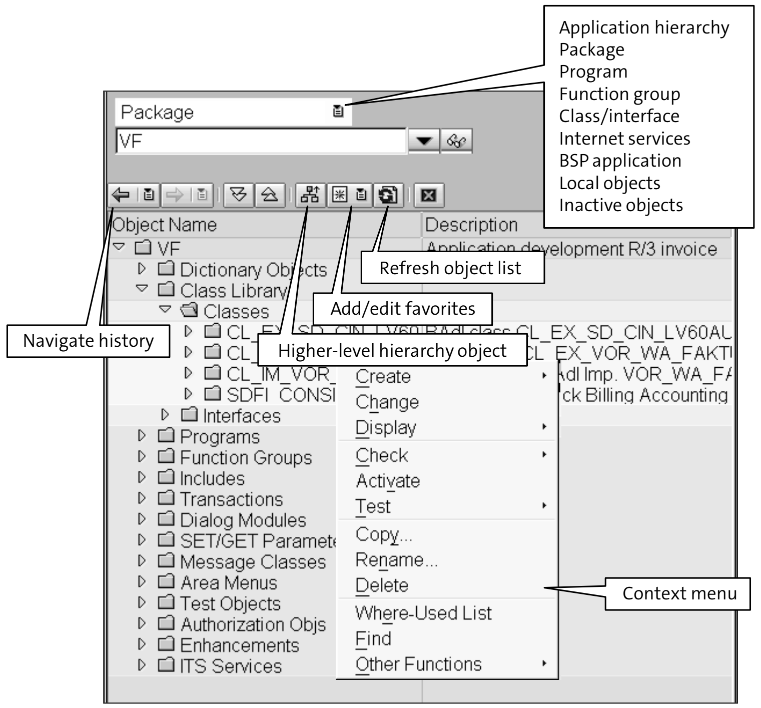


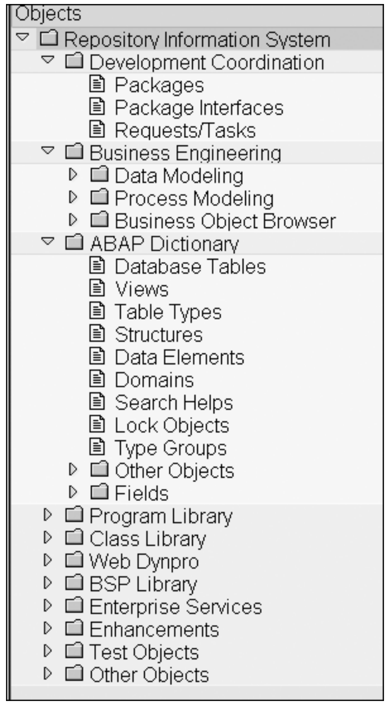
Figure 4.3 Repository Browser Navigation Area

You can create or edit repository objects only if you have the appropriate developer authorization and your user ID has been registered in the SAP Service Marketplace as a developer. You need to provide your developer access key the first time you create any repository object. You only have to provide the access key once because the system stores it for you in the table `DEVACCESS`.

Some of the common context menu functions for repository objects are to check repository objects, activate an object, copy or rename objects, and change the package assignment of an object or write the object to the change request. You also have the where-used list function of the object from the context menu.

Repository Information System

The Repository Information System Browser is used to search for repository objects in the SAP system. The initial screen of the Repository Information System displays a hierarchical list of the categories of repository objects in the SAP system (see Figure 4.4).

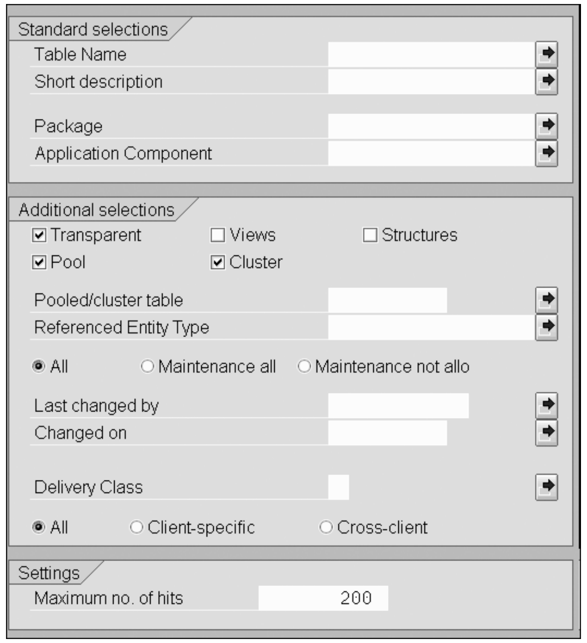


Objects
- Repository Information System
  - Development Coordination
    - Packages
    - Package Interfaces
    - Requests/Tasks
  - Business Engineering
    - Data Modeling
    - Process Modeling
    - Business Object Browser
  - ABAP Dictionary
    - Database Tables
    - Views
    - Table Types
    - Structures
    - Data Elements
    - Domains
    - Search Helps
    - Lock Objects
    - Type Groups
    - Other Objects
    - Fields
  - Program Library
  - Class Library
  - Web Dynpro
  - BSP Library
  - Enterprise Services
  - Enhancements
  - Test Objects
  - Other Objects

Figure 4.4 Repository Information System

The Repository Information System has a selection screen to provide the search criteria for the repository objects. You can search for repository objects within a package or search for ABAP Dictionary objects or programs by specifying the search criteria on the selection screen. Figure 4.5 displays the Repository selection screen for database tables within ABAP Dictionary objects.

You can access the selection screen for the repository object by double-clicking on the type of repository object in the object navigation area; the selection screen relevant for the object will then be displayed in the tool area of the Object Navigator.



Standard selections

Table Name
Short description

Package
Application Component

Additional selections

[✓ Transparent] Views Structures
[✓ Pool] [✓ Cluster]

Pooled/cluster table
Referenced Entity Type

[✓ All] Maintenance all Maintenance not allo

Last changed by
Changed on

Delivery Class

[✓ All] Client-specific Cross-client

Settings

Maximum no. of hits 200

Figure 4.5 Repository Selection Screen for ABAP Dictionary Objects

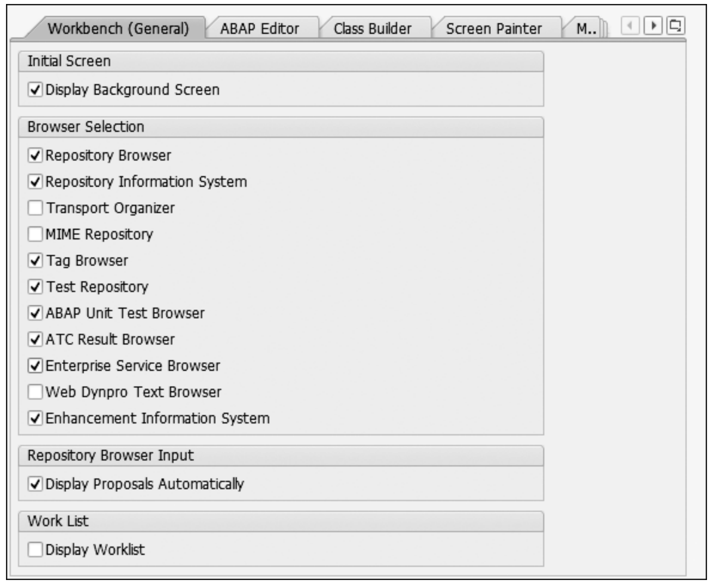
You can customize the selection screen of the Repository Information System by selecting **Edit • Settings** from the initial screen of the Repository Information System Browser. You can specify in the customization if you want to see the default selection screen or the screen with all of the available selection criteria. You can also display all of the selection criteria for the object by clicking on the icon on the application toolbar.

You can specify the search criteria for the repository object in the selection screen and execute. The system displays the search results in the tool area of the Object Navigator. You can also select the object from the search result and display the where-used list of the object if required, and you can double-click on it to look at the object itself in the appropriate tool.

The Repository Information System is a useful tool to display the available enhancements within the SAP system. You can search for both the definition and the implementations of the enhancement with this tool. You can filter your selection of enhancements by application component or package. You can search for business add-ins, customer exits, enhancement implementations, and enhancement definitions. For a detailed explanation of enhancements and modifications, refer to Chapter 18.

Workbench Settings

You can configure the ABAP Workbench to change its look and functionality. The workbench is configured from the menu path **Utilities • Settings** (see Figure 4.6). The workbench settings are user-specific and therefore do not affect anyone else in the system.



Initial Screen
- [✓] Display Background Screen

Browser Selection
- [✓] Repository Browser
- [✓] Repository Information System
- [ ] Transport Organizer
- [ ] MIME Repository
- [✓] Tag Browser
- [✓] Test Repository
- [✓] ABAP Unit Test Browser
- [✓] ATC Result Browser
- [✓] Enterprise Service Browser
- [ ] Web Dynpro Text Browser
- [✓] Enhancement Information System

Repository Browser Input
- [✓] Display Proposals Automatically

Work List
- [ ] Display Worklist

Figure 4.6 ABAP Workbench Customization Screen

The Object Navigator can display only eight browser lists at a time. You can unselect the browser selection checkbox for the browsers that you are not going to use very often.

Within the Workbench settings, you also have the option to configure the ABAP Editor settings. You can select the new Front-End Editor, the old Front-End Editor, or the standard Back-End Editor and customize its settings. You have the option to set your default ABAP Debugger to either the classic or new ABAP Debugger. You

can change your pretty printer settings, split screen settings, and the pattern settings. You can maintain the settings for the Class Builder, Screen Painter, Menu Painter, Function Builder, Repository Information System, Data Browser, Internet Transaction Server, Business Server Pages, Web Dynpro, Transport Organizer, SAPscript, eCATT, and proxy generation from the Workbench settings as well.

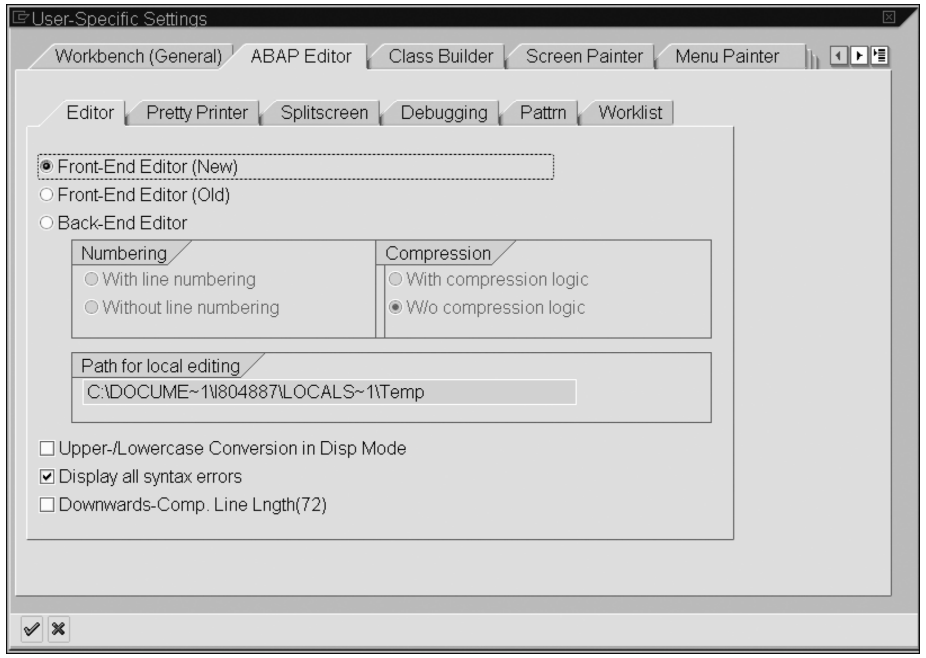
ABAP Editor and Workbench Settings

There are three different modes for the ABAP Editor:

- Front-End Editor (source code mode—new)
- Front-End Editor (plain text mode—old)
- Back-End Editor (line-based mode)

The three editors are fully compatible and interchangeable. The source code created in one editor can be viewed by all other modes.

The choice of the editor is based on the user-specific settings made in the ABAP Workbench. The editor can be configured within the ABAP Editor via the menu path **Utilities • Settings • ABAP Editor** (see Figure 4.7).



- [✓ Front-End Editor (New)]
- Front-End Editor (Old)
- Back-End Editor

Numbering
- With line numbering
- Without line numbering

Compression
- With compression logic
- [✓ W/o compression logic]

Path for local editing: C:\DOCUME~1\I804887\LOCALS~1\Temp

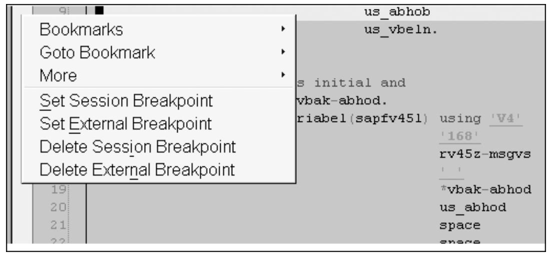
- [ ] Upper-/Lowercase Conversion in Disp Mode
- [✓] Display all syntax errors
- [ ] Downwards-Comp. Line Lngth(72)

Figure 4.7 ABAP Editor Customization Screen

The **Front-End Editor (New)** option provides the latest editor and comes with SAP GUI for Windows 7.0. The new editor is an ActiveX control and is fully integrated into the SAP NetWeaver 7.0 environment.

The new editor has all of the modern code editing features:

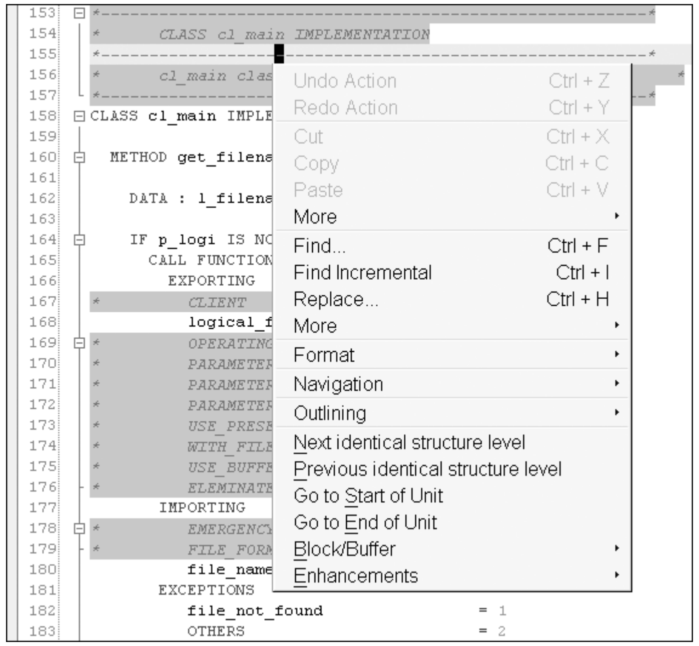
- The left margin of the main screen of the editor displays any bookmarks and breakpoints. Breakpoints are displayed with a red stop sign, and bookmarks are displayed as a blue flag in the editor margin. You can set up to nine numbered bookmarks and an unlimited number of bookmarks that are not numbered on the editor for fast navigation within the code.
- The editor has a line number margin next to the editor margin, where the line number is displayed.
- The code changes are marked with a red triangle against the line number.
- The status bar displays the current status of the code.
- The vertical scroll tip provides information about the current scroll position within the code, current function, class, or method.
- You can split the code editor screen horizontally by double-clicking on the splitter line on the vertical scroll bar. You can also just drag the splitter line to split the editor screen horizontally.
- You can collapse or expand blocks of code such as `IF-ENDIF` or `CASE-ENDCASE`.
- The status bar of the editor displays the current status of `CapsLock` and `NumLock` and the line number of the cursor position. `CapsLock` and `NumLock` can be changed by double-clicking on them. Double-clicking on the line number displays the **Go to Line** dialog.
- The Front-End Editor has two types of context menu; the context menu options depend on the area selected for the context. The margin context menu is displayed by right-clicking on the left margin and has the option to set breakpoints, delete breakpoints, set bookmarks, clear bookmarks, or navigate to a bookmark (see Figure 4.8).



```
                                      us_abhob
                                      us_vbeln.

                    s initial and
                    vbak-abhod.
                    riabel(sapfv451) using 'V4'
                                          '168'
                                          rv45z-msgvs
                                          ' '
                                          *vbak-abhod
                                          us_abhod
                                          space
```

Figure 4.8 Margin Context Menu

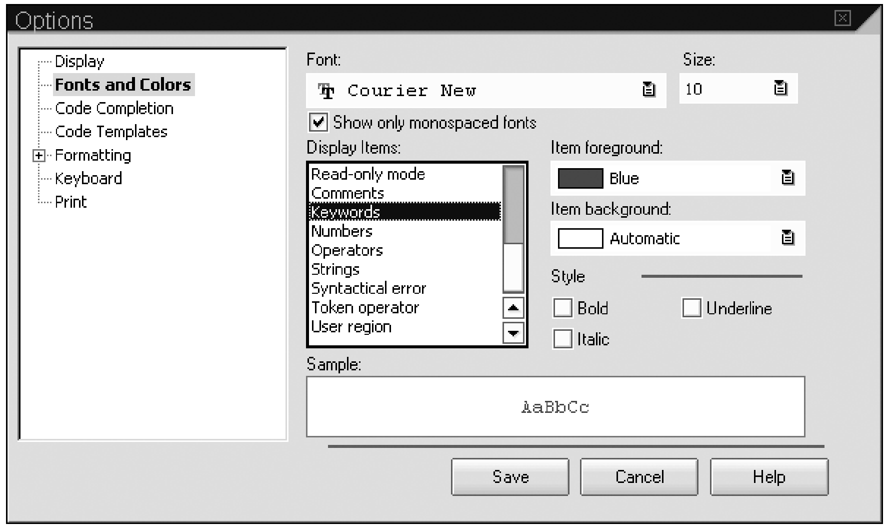
The context menu in the editing area displays the menu options for the ABAP code. The editing area context menu has various formatting, editing, and navigation options (see Figure 4.9).



```
*---------------------------------------------------------------*
*      CLASS cl_main IMPLEMENTATION
*---------------------------------------------------------------*
*      cl_main clas                                              *
*---------------------------------------------------------------*
CLASS cl_main IMPLE

  METHOD get_filena

    DATA : l_filena

    IF p_logi IS NO
      CALL FUNCTION
        EXPORTING
*         CLIENT
          logical_f
*         OPERATING
*         PARAMETER
*         PARAMETER
*         PARAMETER
*         USE_PRESE
*         WITH_FILE
*         USE_BUFFE
*         ELEMINATE
        IMPORTING
*         EMERGENCY
*         FILE_FORM
          file_name
        EXCEPTIONS
          file_not_found              = 1
          OTHERS                      = 2
```

Figure 4.9 Context Menu for Text Formatting

- The editor provides code hints at runtime as you type by suggesting keyword hints, block templates, and so on. You can accept the code hint by pressing the `Tab` key or insert a block template by pressing `Ctrl` + `Enter`. Furthermore, the editor supports WYSIWYG export functionality and exports to HTML, PDF, and RTF formats.
- The new ABAP Editor is a fully integrated development environment (IDE) for ABAP programming. It supports syntax highlighting, outlining language structures, real-time code hints, and auto-completion of language structures.
- With the new code editor you can customize the highlighting for the keyword, strings, and comments. You can customize the font, color, and size for ABAP keywords and comments. Similarly, you can customize font, color, and size for strings, breakpoints, and other display items in the code editor (see Figure 4.10).


Options

- Display
- Fonts and Colors
- Code Completion
- Code Templates
- Formatting
- Keyboard
- Print

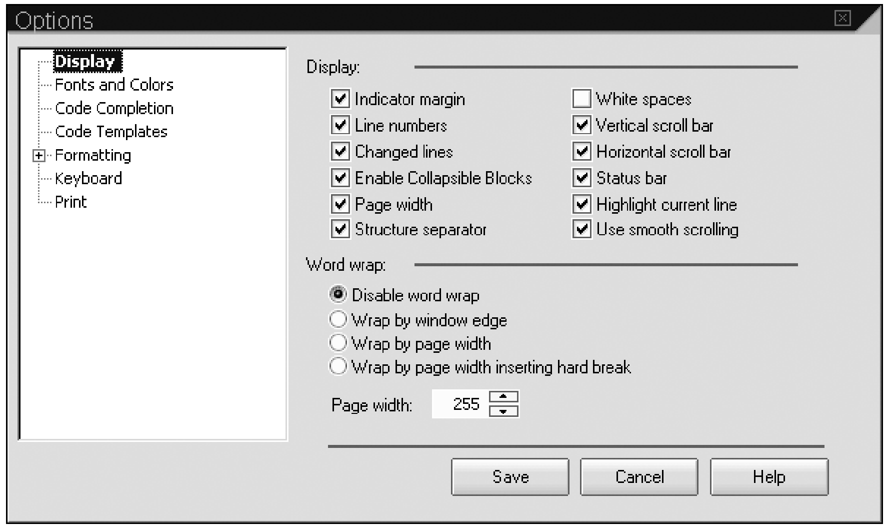
Font: Courier New
Size: 10
✓ Show only monospaced fonts

Display Items: Read-only mode, Comments, Keywords, Numbers, Operators, Strings, Syntactical error, Token operator, User region

Item foreground: Blue
Item background: Automatic
Style: ☐ Bold ☐ Underline ☐ Italic
Sample: AaBbCc

Figure 4.10 Code Editor Customization for Color Schemas

- You can customize the display settings of your editor. Figure 4.11 shows the display customization screen. The display and the word wrap options can be switched on or off according to your preference.


Options

- Display
- Fonts and Colors
- Code Completion
- Code Templates
- Formatting
- Keyboard
- Print

Display:

✓ Indicator margin	☐ White spaces
✓ Line numbers	✓ Vertical scroll bar
✓ Changed lines	✓ Horizontal scroll bar
✓ Enable Collapsible Blocks	✓ Status bar
✓ Page width	✓ Highlight current line
✓ Structure separator	✓ Use smooth scrolling

Word wrap:

- ● Disable word wrap
- ○ Wrap by window edge
- ○ Wrap by page width
- ○ Wrap by page width inserting hard break

Page width: 255

Figure 4.11 Display Customization for the Editor

- Code completion has been added to new the ABAP Front-End edition in SAP NetWeaver 7.0 EHP 2. The tool proposes appropriate ABAP words and operands for where you are in your code. To insert the proposed text, press `Ctrl`+`Space`.
- The other feature that's new in SAP NetWeaver 7.0 EHP 2 is that with the new code-based view in the ABAP Editor, you can toggle between the classic form-based view in Class Builder and the new code-based view. This new view allows you to see a global class's whole code, such as a program in the ABAP Editor, and you can edit the code. When you save, the modified source code is then split among the corresponding include programs.
- Finally, you can customize the code completion options for the editor. This option allows you to complete the available keyword from the dictionary or complete the class, method, or variable name within the scope of the visibility.

ABAP Workbench Tools in Detail

With the Object Navigator, you work directly with the repository object. The relevant tool for the repository object is automatically selected when you double-click on the repository object. The SAPscript Editor, Smart Forms Editor, and Customer Enhancement Projects are some of the development tools that are not available within the Object Navigator. The following are the most commonly used ABAP Workbench development tools:

- **ABAP Editor**
  The ABAP Editor is used to develop programs and can be executed via Transaction SE38. You use the editor to write ABAP programs.
- **Class Builder**
  The Class Builder allows you to create ABAP classes within the ABAP Workbench. You can create ABAP classes and interfaces, implement inheritance relationships, and create attributes, methods, and events for the global classes and interfaces that you build. You can access the Class Builder via Transaction SE24.
- **Function Builder**
  You use the Function Builder to write function modules and define function groups within the ABAP Workbench. You use the Function Builder to create, change, display, and test function modules. You can access the Function Builder directly via Transaction SE37.

- **Screen Painter**
  The Screen Painter is an ABAP Workbench tool used to create screens for SAP GUI transactions. You use the Screen Painter to create screens and write the flow logic for screens. You can access the Screen Painter via Transaction SE51.

  The Screen Painter layout editor used to design the screen has two versions: the alphanumeric layout editor and the graphic layout editor. By default, the Screen Painter layout editor is alphanumeric, but the mode can be changed to the graphical layout editor by changing the Screen Painter setting by following the menu path **Utilities • Screen Painter** and selecting **Graphical Layout Editor**.

  You define the screen attributes and the screen field attributes for the screen fields and the flow logic using the Screen Painter. You can create tab strips, table controls, subscreens, custom containers, and other screen elements using the Screen Painter.
- **Menu Painter**
  The Menu Painter is an ABAP Workbench tool to create the user interface for your program or transaction. You can create the menu bar, standard toolbar, application toolbar, and GUI title using the Menu Painter. You also assign the function keys to the functions you create in the Menu Painter. You can execute the Menu Painter outside the ABAP Workbench via Transaction SE41.
- **ABAP Dictionary**
  The ABAP Dictionary tool is an integral part of the ABAP Workbench. The ABAP Dictionary is used to create, change, and display transparent tables, pooled and cluster tables, views, types, domains, data elements, structures, table types, search helps, and lock objects. The database utility is also an integral part of the ABAP Dictionary tool. You can directly access the ABAP Dictionary tool outside the ABAP Workbench via Transaction SE11.
- **Web Application Builder**
  You can build web applications within the ABAP Workbench. The ABAP Workbench delivers the tool to develop ITS web applications or BSP applications.
  - ITS was the first approach by SAP to extend business applications to web browsers by converting the SAP Dynpro screen into HTML format, making it possible to access SAP transactions via web browsers. For ITS applications you can create Internet services for existing SAP transactions, HTML templates for the transaction screens, and MIME objects to add icons and graphics to the web screen layout.

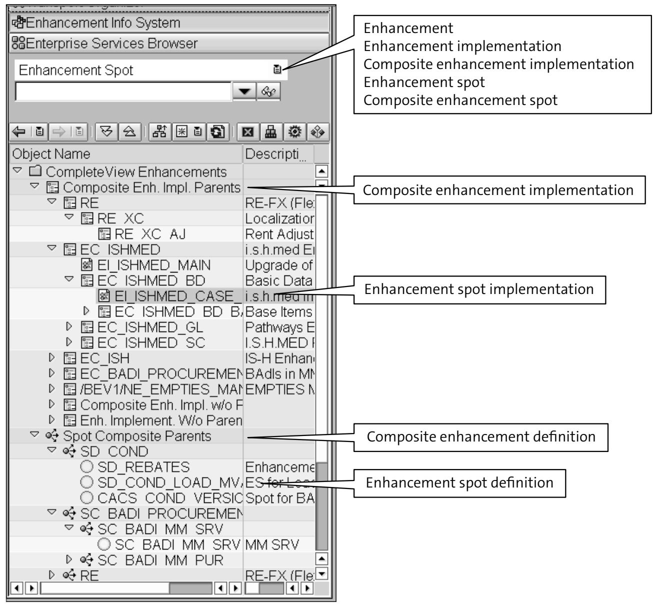
  - You can also develop BSP applications with the Web Application Builder. You can design your web page for the BSP application using HTML and ABAP or Java Scripting language. You can define the page flow and implement the event handler with ABAP with the Web Application Builder for BSP. The Web Application Builder also allows you to define the theme for your layout and integrate the MIME objects into the web application.
- **Web Dynpro Explorer**
  The workbench has a tool for development of Web Dynpro ABAP applications (as of SAP NetWeaver 7.0). The tool consists of the runtime environment and the graphical tool to design the Web Dynpro views within the ABAP development environment.

Enhancement Information System

The Enhancement Information System provides an overview of the defined enhancements and the enhancement implementations within the Enhancement Framework. The Enhancement Framework is the technical basis for SAP's enhancement concept (for detailed information, refer to Chapter 18). It enables you to see the enhancement definitions and implementations in the system.

The Enhancement Information System is a tree display structure for enhancement definitions and enhancement implementations. The enhancement element definition is displayed under the enhancement spot definition, which in turn is displayed under the composite enhancement spot. Similarly, the enhancement element implementation is displayed under the simple enhancement implementation, which in turn is displayed under the composite enhancement implementation. The enhancement implementation node is not displayed if the enhancement has not been implemented. Also, it is not necessary that all enhancement spot definitions or implementations are assigned to a composite enhancement definition. Enhancement definitions and implementations are discussed in detail in Chapter 18.

You can filter the display of the enhancements within the Enhancement Information System. You have the option to display any or all of the following: the composite enhancement, enhancement spot, enhancement implementation, or composite enhancement implementation within the Enhancement Information System (see Figure 4.12).

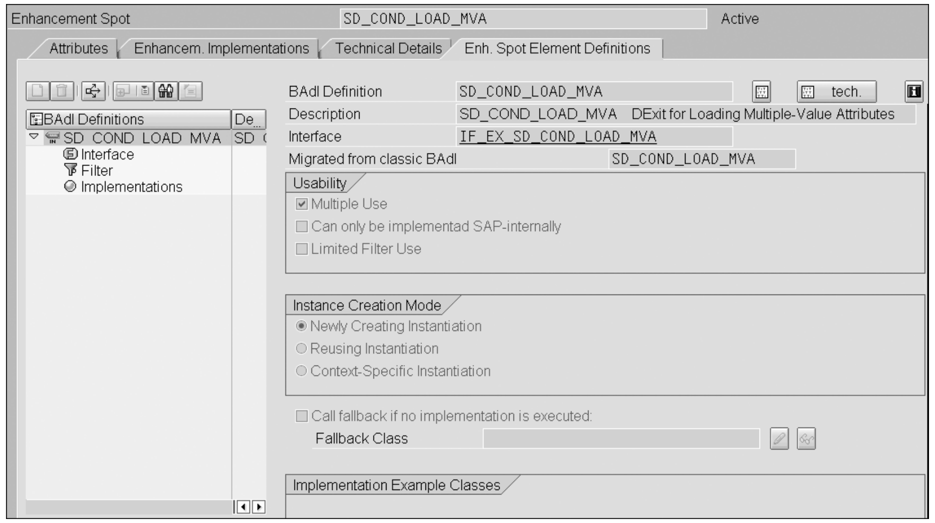

Enhancement Info System
Enterprise Services Browser
Enhancement
Enhancement implementation
Composite enhancement implementation
Enhancement spot
Composite enhancement spot
Composite enhancement implementation
Enhancement spot implementation
Composite enhancement definition
Enhancement spot definition

Enhancement Spot

Object Name	Descripti...
CompleteView Enhancements	
Composite Enh. Impl. Parents	
RE	RE-FX (Fle
RE XC	Localizatior
RE XC AJ	Rent Adjust
EC ISHMED	i.s.h.med Ei
EI_ISHMED_MAIN	Upgrade of
EC ISHMED BD	Basic Data
EI_ISHMED_CASE_	i.s.h.med In
EC ISHMED BD B/	Base Items
EC_ISHMED_GL	Pathways E
EC ISHMED SC	I.S.H.MED I
EC_ISH	IS-H Enhan
EC_BADI_PROCUREMEN	BAdIs in MM
/BEV1/NE_EMPTIES_MAN	EMPTIES N
Composite Enh. Impl. w/o F	
Enh. Implement. W/o Paren	
Spot Composite Parents	
SD COND	
SD_REBATES	Enhanceme
SD_COND_LOAD_MV/	ES for Loa
CACS COND VERSIC	Spot for BA
SC BADI PROCUREMEN	
SC BADI MM SRV	
SC BADI MM SRV	MM SRV
SC BADI MM PUR	
RE	RE-FX (Fle

Figure 4.12 Enhancement Information System Navigation Screen

The enhancement spot definition can be displayed in the ABAP Workbench tool area by double-clicking on the enhancement spot definition in the navigation area. Figure 4.13 displays the **Enhancement Spot** definition screen in the tool area.



Enhancement Spot SD_COND_LOAD_MVA Active

BAdI Definition SD_COND_LOAD_MVA
Description SD_COND_LOAD_MVA DExit for Loading Multiple-Value Attributes
Interface IF_EX_SD_COND_LOAD_MVA
Migrated from classic BAdI SD_COND_LOAD_MVA

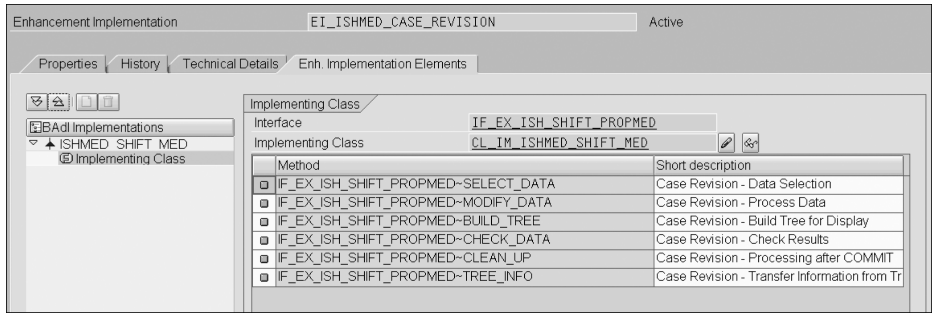
Usability
- ☑ Multiple Use
- ☐ Can only be implementad SAP-internally
- ☐ Limited Filter Use

Instance Creation Mode
- ● Newly Creating Instantiation
- ○ Reusing Instantiation
- ○ Context-Specific Instantiation

☐ Call fallback if no implementation is executed:
Fallback Class

Figure 4.13 Enhancement Spot Definition Screen

Similarly, you can display the **Enhancement Implementation** screen in the tool area of the ABAP Workbench by double-clicking on the enhancement spot implementation (see Figure 4.14).



Enhancement Implementation EI_ISHMED_CASE_REVISION Active

Implementing Class
Interface IF_EX_ISH_SHIFT_PROPMED
Implementing Class CL_IM_ISHMED_SHIFT_MED

Method	Short description
IF_EX_ISH_SHIFT_PROPMED~SELECT_DATA	Case Revision - Data Selection
IF_EX_ISH_SHIFT_PROPMED~MODIFY_DATA	Case Revision - Process Data
IF_EX_ISH_SHIFT_PROPMED~BUILD_TREE	Case Revision - Build Tree for Display
IF_EX_ISH_SHIFT_PROPMED~CHECK_DATA	Case Revision - Check Results
IF_EX_ISH_SHIFT_PROPMED~CLEAN_UP	Case Revision - Processing after COMMIT
IF_EX_ISH_SHIFT_PROPMED~TREE_INFO	Case Revision - Transfer Information from Tr

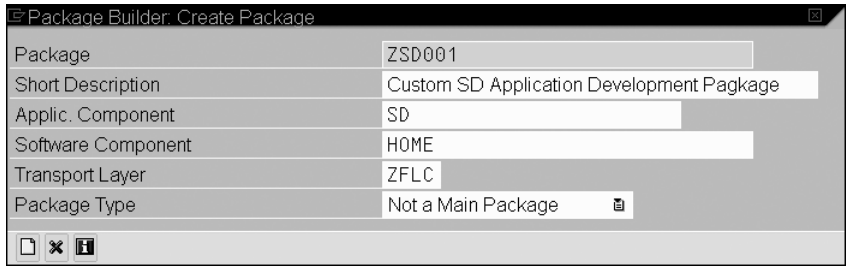
Figure 4.14 Enhancement Spot Implementation Screen

Packages and Their Attributes

Prior to release 4.6C, all ABAP development objects were assigned to development classes that were used to group related development objects. Packages were assigned to the application components. There can be multiple packages for an application component. As of release 4.6C and beyond, all ABAP development objects are to be assigned to the package. Packages are containers for the development objects within the transport layer that allow the objects to be transported. You can create or edit packages in the Package Builder in the Repository Browser. You can display all of the objects assigned to the package in the Repository Browser by selecting a package for the object list type. The SAP system has a predefined package (`$TMP`) to which all of the local development objects are assigned. Local objects assigned to the `$TMP` package cannot be transported. Nontransportable package names start with `$`.

The Package Builder can also be accessed via Transaction SE21 or SPACKAGE. The Package Builder is used to create and assign attributes to packages.

To create a package for your development project, you can create a main package and the subpackage, specify the package hierarchy, define the package interface, add elements to the package interface, and then define the use access for the package user. You create main packages only if you want to create a package hierarchy. Normally you create a package and specify the package type as **Not a Main Package**. You create the package from the Repository Browser (see Figure 4.15).



Package	ZSD001
Short Description	Custom SD Application Development Pagkage
Applic. Component	SD
Software Component	HOME
Transport Layer	ZFLC
Package Type	Not a Main Package

Figure 4.15 Create Package Screen

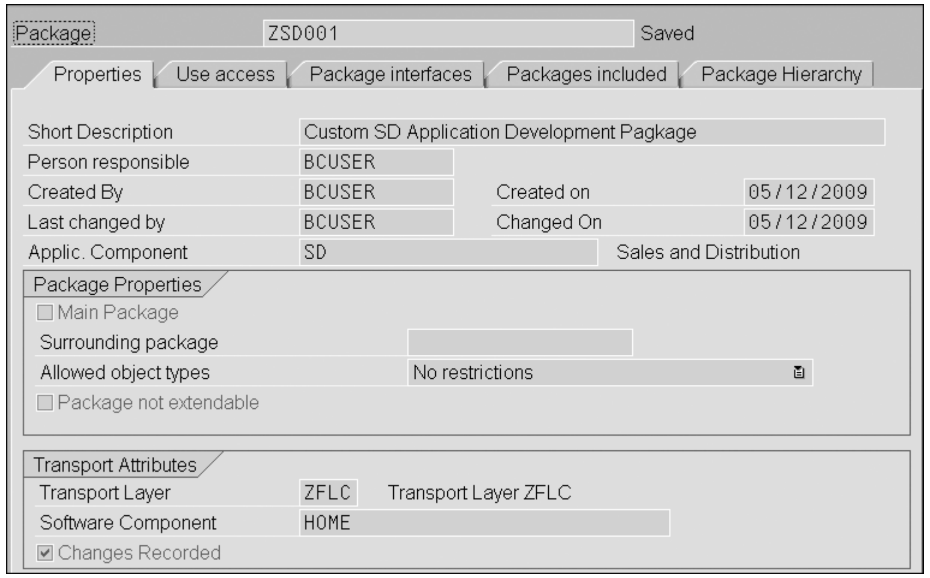
You specify the package name, short description, application component, software component, transport layer, and package type and then click on the **Create Package** icon. Customer package names should start with the letter *Z* or *Y*. The software component for a customer package is always `HOME`. The transport layer determines if the object assigned to the package is local or transportable. Normally, you carry out all of your development activity in the development system and then move your objects to the quality assurance system and the production system. Your system administration team will set up a transport layer for your development system, for which the transport route is defined to move the object from one system to another. Based on the transport layer, the objects assigned to the package are moved to the different systems defined in the transport route for the transport layer.

You can specify the package properties, use access, interface, and package hierarchy in the Package Builder screen in change mode. However, you are not required to specify the package interface or use access unless you want to protect the object assigned to the package from being used arbitrarily by other packages. Figure 4.16 displays the Package Builder screen where you can specify the package attributes.

Packages have attributes such as nesting, package interfaces, visibility, and use access.

- Nesting allows you to embed packages in other packages, thus allowing you to split larger units of repository objects and structure them in a hierarchy.
- Visibility is a property of the package element. Elements within the package are visible to all other elements within the same package and are always invisible to the elements outside of the subpackage.

- Elements within a package can be made visible outside the package by defining a package interface.
- Use access is the right of one package to use the visible elements of the other package interface.



Package ZSD001 Saved

Short Description	Custom SD Application Development Pagkage		
Person responsible	BCUSER		
Created By	BCUSER	Created on	05/12/2009
Last changed by	BCUSER	Changed On	05/12/2009
Applic. Component	SD	Sales and Distribution	

☐ Main Package	
Surrounding package	
Allowed object types	No restrictions
☐ Package not extendable	

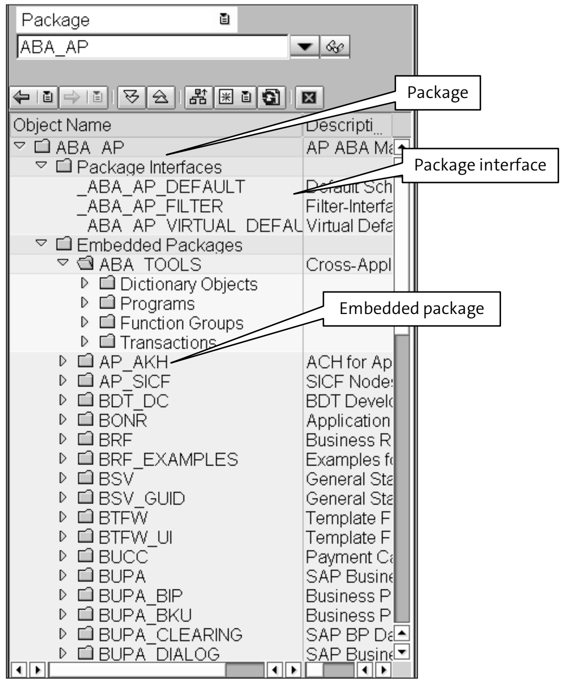
Transport Layer	ZFLC	Transport Layer ZFLC
Software Component	HOME	
☑ Changes Recorded		

Figure 4.16 Package Builder Screen for Package Attributes

Packages use interfaces and visibility to make their elements visible to other packages. All visible elements of a package can be used by the other package. Use access restricts the use of the visible elements of the package. The visible elements of the provider package can only be used by the package if the use access of the interface has been created in the package (see Figure 4.17).

Package interface and visibility are useful if you want to make an element of your subpackage visible to the higher-level package or main package.

The system checks that your package complies with these rules based on the entry in the table `PAKPARAM`. The package check is switched off by default in the customer system. The entry `GLOBAL_SWITCH` for the key field `NAME` in the table `PAKPARAM` controls the behavior of the package check. By default, the `GLOBAL_SWITCH` key is set to `OFF` in the table `PARPARAM`. To switch on package check, set the `GLOBAL_SWITCH` key to `RESTRICTED` or `R3ENTERPRISE`.



Object Name	Descripti...
ABA_AP	AP ABA Ma
Package Interfaces	
_ABA_AP_DEFAULT	Default Sch
_ABA_AP_FILTER	Filter-Interfa
_ABA_AP_VIRTUAL_DEFAU	Virtual Defa
Embedded Packages	
ABA_TOOLS	Cross-Appl
Dictionary Objects	
Programs	
Function Groups	
Transactions	
AP_AKH	ACH for Ap
AP_SICF	SICF Node
BDT_DC	BDT Develo
BONR	Application
BRF	Business R
BRF_EXAMPLES	Examples f
BSV	General Sta
BSV_GUID	General Sta
BTFW	Template F
BTFW_UI	Template F
BUCC	Payment C
BUPA	SAP Busine
BUPA_BIP	Business P
BUPA_BKU	Business P
BUPA_CLEARING	SAP BP Da
BUPA_DIALOG	SAP Busine

Figure 4.17 Package Structure in Object Navigator of ABAP Workbench

Table 4.1 displays the values for `GLOBAL_SWITCH` and their effects on package check.

Value	Behavior
`RESTRICTED`	The package check is only performed if you have selected the **Package Check** as **Client** or **Package Check** as **Server Attribute** in the package.
`R3ENTERPRISE`	The same checks that are performed for entry `RESTRICTED` are performed here. With objects belonging to structure packages, an additional check is performed to determine whether use access has been defined between them. Note that in this case an additional entry has to be made in the table `PAKPARAM` with the key field `NAME` equal to `SAP_DEV_SYSTEM` and `VALUE` set to `X`.
`OFF`	Switches off package check.

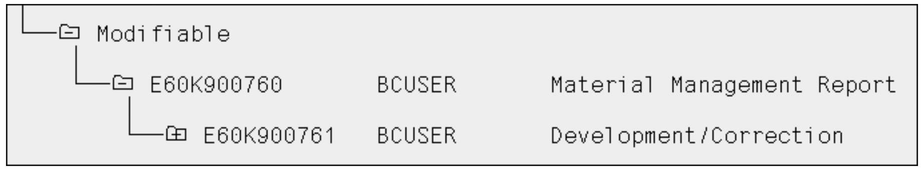
Table 4.1 Valid Values for Package Check

Transport Organizer

The Change and Transport System (CTS) provides you with a tool to organize your ABAP Workbench, cross-client customization, and customization work and then transport the changes through your system landscape. The CTS records all of the changes in the change request. The change request is also referred to as the transport request. You can release your task and the change request once you have completed and tested your development. You have to ensure that your development object is syntactically correct and active before you release the change request. The change request is then used to transport the changes to other systems or clients based on the transport route.

Repository objects and cross-client customizations are assigned to a workbench request, whereas the client-specific customization objects are assigned to a Customizing request. Each change request can have one or multiple tasks, and the repository objects or the customization objects are assigned to one of these tasks. The task is assigned to a user, and only the owner of the task can record his changes to the task.

There are two types of workbench tasks: development/correction and repair tasks. The repository object changes are recorded in the development/correction task if the current system is the original system of the object. The object is recorded in the repair task if the current system is not the original system of the object. All SAP standard object modifications are recorded in the repair task of the workbench request. Figure 4.18 displays the change request structure. The top level is the change request, and the lower-level number is the task of the change request.



```
Modifiable
   E60K900760    BCUSER     Material Management Report
      E60K900761 BCUSER     Development/Correction
```

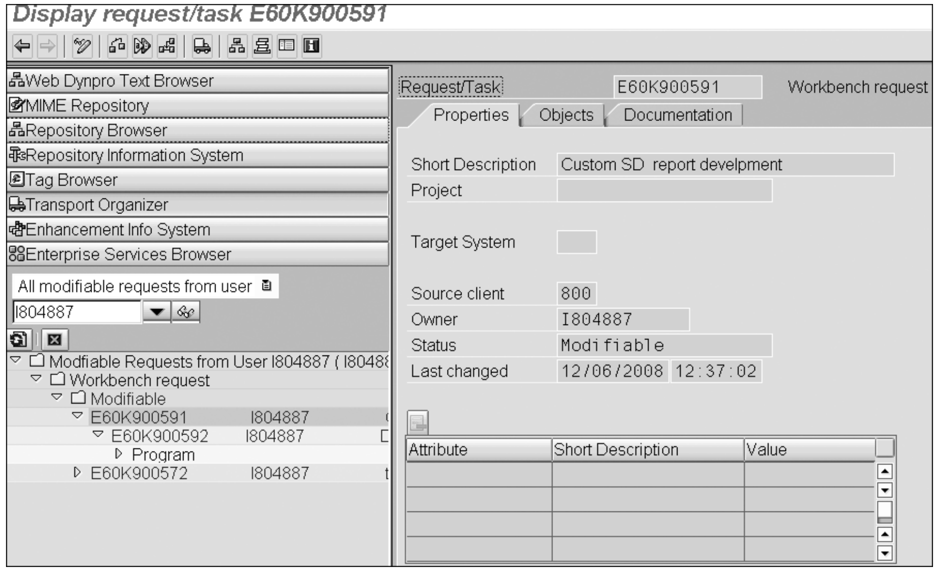
Figure 4.18 Transport Request Structure

Each repository object is assigned to a package, and the package is assigned to the transport layer. If the route for the transport layer is defined in the Transport Management System (TMS), then the object recorded in the transport task is transportable; otherwise, the task belongs to the local change request.

The system administration team creates the transport layer in the TMS to transport the objects in your system landscape. The transport layer is assigned to the development system. The administration team needs to set up a transport route after creating the transport layer.

There are two types of transport routes: Your administration team will set up the consolidation route for each transport layer and then the delivery route. The development system is the source system for the consolidation route, and the quality assurance system is the target system. You define the delivery route to transport the objects from the quality system to the other target system, which can be production and other training systems. So once the object is imported to the quality system, you need delivery routes to transport the objects from the quality system to other systems.

The Transport Organizer is integrated with the Object Navigator or can be accessed via Transaction SE10. You can display the transport request and the associated tasks of the request from the workbench by double-clicking on the request in the navigation area of the workbench. The request will be displayed in the tool area, whereby you can display the properties, objects, and documentation of the request. The request editor displays all of the recorded changes within the request (see Figure 4.19).


Display request/task E60K900591

I804887

Modfiable Requests from User I804887 (I80488
Workbench request
Modifiable
E60K900591 I804887
E60K900592 I804887
Program
E60K900572 I804887

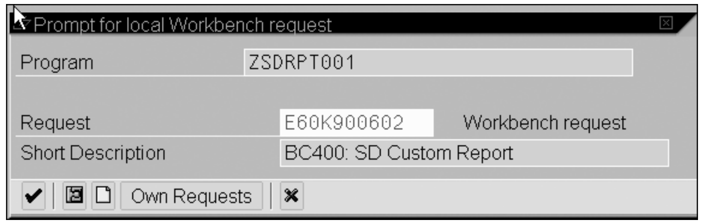
Request/Task	E60K900591	Workbench request
Short Description	Custom SD report develpment	
Project		
Target System		
Source client	800	
Owner	I804887	
Status	Modifiable	
Last changed	12/06/2008 12:37:02	

Attribute	Short Description	Value

Figure 4.19 Request Editor in ABAP Workbench

You can create change requests by creating the repository object. When you are creating the repository object, a dialog window appears to allow you to create a

new request or assign the object to an existing request. The transport attributes are populated automatically based on the attributes of the repository object. Figure 4.20 displays the initial transport request screen.

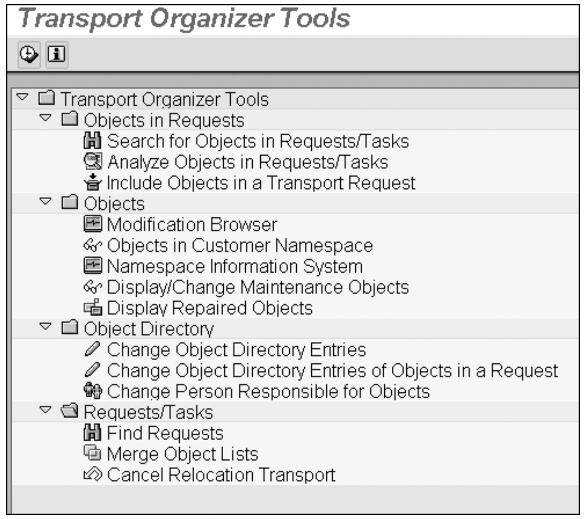

Prompt for local Workbench request

Program	ZSDRPT001	
Request	E60K900602	Workbench request
Short Description	BC400: SD Custom Report	

Figure 4.20 Transport Request Screen

You can also create the transport request from the transport organizer, but you have to populate the transport attributes on your own. You release all of the tasks within the change request and then the change request itself to transport the objects through your system landscape. The system administrator imports the objects to the target system. You can display the transport log by selecting the transport log menu option.

The Transport Organizer tools are integrated with the Transport Organizer. They are a set of programs that help you work with the Transport Organizer. You can start the tool from the initial screen of the Transport Organizer via the menu path **Goto • Transport Organizer Tools** (see Figure 4.21).


Transport Organizer Tools

- Transport Organizer Tools
  - Objects in Requests
    - Search for Objects in Requests/Tasks
    - Analyze Objects in Requests/Tasks
    - Include Objects in a Transport Request
  - Objects
    - Modification Browser
    - Objects in Customer Namespace
    - Namespace Information System
    - Display/Change Maintenance Objects
    - Display Repaired Objects
  - Object Directory
    - Change Object Directory Entries
    - Change Object Directory Entries of Objects in a Request
    - Change Person Responsible for Objects
  - Requests/Tasks
    - Find Requests
    - Merge Object Lists
    - Cancel Relocation Transport

Figure 4.21 Transport Organizer Tools

The tools can be used to search for objects in transport requests or include objects in the transport requests. You can search the request or task based on different search criteria. Similarly, you can display all of the modifications in the customer system using the modification browser.

Practice Questions

These practice questions will help you evaluate your understanding of the topic. The questions shown are similar in nature to those found on the certification examination. Although none of these questions will be found on the exam itself, they allow you to review your knowledge of the subject. Select the correct answers and then check the completeness of your answers in the following solution section. Remember that you must select all correct answers and only correct answers on the exam to receive credit for the question.

1. The Object Navigator incorporates a total of 11 browsers.

☐ **A.** True
☐ **B.** False

2. The Repository Browser is started by default when you execute Transaction SE80 for the Object Navigator.

☐ **A.** True
☐ **B.** False

3. You can list a maximum of six browsers in the Object Navigator.

☐ **A.** True
☐ **B.** False

4. You can maintain SAPscript forms and SAP Smart Forms within the ABAP Workbench.

☐ **A.** True
☐ **B.** False

5. The Repository Information System is a useful tool to search for customer exits/function exits and BAdIs in the SAP system.

☐ **A.** True
☐ **B.** False

6. Which of the following statements about the Object Navigator are true? Select all that apply.

☐ **A.** ABAP programs can be displayed and edited in the Object Navigator.
☐ **B.** Screens can be displayed and edited in the Object Navigator.
☐ **C.** Menus can be displayed and edited in the Object Navigator.
☐ **D.** You can create BAdI implementations in the Object Navigator.
☐ **E.** You can create customer projects (Transaction CMOD) in the Object Navigator.
☐ **F.** The ABAP Dictionary can be maintained in the Object Navigator.

7. Enhancement definitions and implementations can be displayed in the Enhancement Information System.

☐ **A.** True
☐ **B.** False

8. Which of the following is a true statement? Select all that apply.

☐ **A.** All customer repository objects have to be assigned to a package.
☐ **B.** Packages use interfaces and visibility to make their elements visible to other packages.
☐ **C.** The transport layer is a mandatory input field for the package.
☐ **D.** A package can be nested.

9. The software component for a customer package can be...

☐ **A.** `HOME`
☐ **B.** Any SAP software component (i.e., `SAP_APPL`, `SAP_BASIS`, `SAP_HR`, etc.)

10. Which of the following is a true statement? Select all that apply.

☐ **A.** All transportable objects have to be assigned to a package.

☐ **B.** Local repository objects can be transported.

☐ **C.** Repository objects and cross-client customization objects are assigned to the workbench request.

☐ **D.** Client-specific customization objects are assigned to the customizing request.

☐ **E.** Inactive objects can be transported.

11. Which of the following is true? Select all that apply.

☐ **A.** The repository objects and cross-client customization objects are recorded in a task belonging to a local change request if there is no consolidation route leading from the current system defined in the Transport Management System for the transport layer.

☐ **B.** The repository objects and the cross-client customization are recorded in a task belonging to the transportable request if the consolidation route is defined in the Transport Management System.

12. There are _____ versions of the ABAP Editor.

☐ **A.** 3

☐ **B.** 4

☐ **C.** 2

13. Repository objects are client-specific.

☐ **A.** True

☐ **B.** False

14. ABAP Unit Test Browser is included in Object Navigator with SAP NetWeaver 7.0 EHP 2.

☐ **A.** True

☐ **B.** False

Practice Question Answers and Explanations

1. Correct answer: **A**
   The Object Navigator has a total of 11 browsers, as an additional ABAP Unit Browser has been added in SAP NetWeaver 7.0 EHP 2. In earlier releases, it was nine. The following are the object browsers available in the ABAP Workbench:
   – Repository Browser
   – Repository Information System
   – Transport Organizer
   – MIME Repository
   – Tag Browser
   – Test Repository
   – Enterprise Service Browser
   – Web Dynpro Text Browser
   – Enhancement Information System
   – ABAP Unit Browser
   – ABAP Test Cockpit
2. Correct answer: **A**
   The Repository Browser is the default browser for the ABAP Workbench.
3. Correct answer: **B**
   You can display up to eight browsers in the ABAP Workbench.
4. Correct answer: **B**
   You cannot maintain SAPscript forms or SAP Smart Forms in the ABAP Workbench.
5. Correct answer: **A**
   You can search for enhancements including business add-ins and customer exits with the Repository Information System.
6. Correct answers: **A, B, C, F**
   You can edit or display ABAP programs in the ABAP Workbench in the ABAP Editor. You can display or edit screens within the ABAP Workbench in the Screen Painter. You can display and edit menus within the ABAP Workbench in the Menu Painter. The BAdI implementation tool is not integrated within the Object Navigator, so it's not possible to create BAdI implementations in the Object Navigator. The customer project (CMOD) is not integrated within the ABAP Workbench, so it's not possible to create customer projects with the ABAP

Workbench. The ABAP Dictionary is integrated within the ABAP Workbench, so it can be maintained in ABAP Workbench.

7. Correct answer: **A**
   Enhancement definitions and implementations can be displayed in the Enhancement Information System.
8. Correct answers: **A, B, D**
   You can create a local object and assign it to package `$TMP`, but you have to assign the object to a package (whose name does not start with '`$`') if you want to transport the object from one system to another. A package has to define a package interface and visibility to make its elements visible to other packages. The transport layer is not a mandatory input field for the package. The transport layer is assigned to the package if it is defined for the system. A package can be nested.
9. Correct answer: **A**
   The software component of the customer package should always be `HOME`.
10. Correct answers: **A, C, D**
    The repository object has to be assigned to the package to transport the object to another system within the system landscape. You cannot transport a local repository object. Repository objects and cross-client customization objects are assigned to a workbench request. Client-specific customization objects are assigned to the customization request and are not assigned to the package. Inactive objects can be transported.
11. Correct answers: **A**, **B**
    A local change request is created if the consolidation route is not defined; otherwise, a transportable change request is created if the consolidation route for the transport layer is defined.
12. Correct answer: **A**
    There are three modes of ABAP Editor: the new Front-End Editor, the old Front-End Editor, and the Back-End Editor.
13. Correct answer: **B**
    Repository objects are cross-client objects, and therefore they are system wide.
14. Correct answer: **A**
    ABAP Unit Test Browser is the new tool in Object Navigator in SAP NetWeaver 7.0 EHP 2.

Takeaway

You should now understand the ABAP Workbench features and be able to navigate within it. You should be able to use the available browsers within the ABAP Workbench and complete your development in an efficient manner. You should know how to configure the ABAP Workbench and the various development tools.

To be a successful developer, you should know about all the available development tools and their uses. It is important to understand the features of the new ABAP Editor because it will be very helpful for your application development and will increase the productivity of your development team.

Finally, you should understand the package concept and the use of the Transport Organizer. You should understand the concept of the transport request and the different types of transport request and their use for the migration of development objects from the development environment to the production environment.

Refresher

Table 4.2 shows key concepts about the ABAP Workbench.

Key Concept	Definition
ABAP Workbench	The ABAP Workbench is an integrated development environment for the ABAP developer.
Repository Browser	The Repository Browser is a tool within the Object Navigator and is used by the developer to access the repository objects and workbench tools to create repository objects.
Repository Information System Browser	The Repository Information System Browser is a search tool to search for repository objects.
Enhancement Information System	The Enhancement Information System is a tool to search for enhancement definitions and implementations in the SAP system.
Transport Organizer	The Transport Organizer is a tool to work with the change request and the objects within the change request. Using this tool, you can view the objects assigned to the change request and the transport log of the request.

Table 4.2 Key Concept Refresher

In this chapter we covered the ABAP Workbench and its use in detail. We covered some of the most commonly used browsers such as the Repository Browser, Repository Information System, Enhancement Information System, and Transport Organizer. Furthermore, we discussed the ABAP Workbench, the new ABAP Editor, and the concepts of package and transport requests in detail. This knowledge will allow you to easily pass this topic on the certification examination.

Contents

Acknowledgments 17
Preface 19

PART I General Introduction

1 ABAP Development Certification Track: Overview 29

Certification Exam Scoring Changes 30
Certification Levels 30
Advanced Certification Levels 32
Becoming an SAP Certified ABAP Development Associate: Overview 33
Associate Examination Specifics 33
Competency Areas 34

2 Courses and Experience 37

Training Courses for ABAP 38
Sources of Information 39
Strategic Understanding 39
SAP Examination Strategies 41
General Examination Strategies 42

PART II Exam Preparation

3 SAP NetWeaver: Overview 47

Objectives of This Portion of the Test 48
Key Concepts Refresher 48

SAP Products in a Nutshell 49
Product Evolution 50
SAP NetWeaver Architecture 56
Kernel and Administration Services 58
Software-Oriented View 59
User-Oriented View 62
Structure of a Work Process 64
Important Terminology 68
Practice Questions 69
Practice Question Answers and Explanations 71
Takeaway 73
Refresher 73
Tips 74

4 ABAP Workbench Usage 75

Objectives of this Portion of the Test 76
Key Concepts Refresher 77
ABAP Workbench 77
Repository Browser 82
Repository Information System 84
Workbench Settings 86
ABAP Editor and Workbench Settings 87
ABAP Workbench Tools in Detail 91
Enhancement Information System 93
Packages and Their Attributes 95
Transport Organizer 99
Practice Questions 102
Practice Question Answers and Explanations 105
Takeaway 107
Refresher 107

5 ABAP Debugger Program Usage 109

Objectives of this Portion of the Test 111
Key Concepts Refresher 111
New and Classic Debugger 112
New Debugger Tools and UI 116
Assertions and Breakpoints 125
New Debugger Customization and Settings 126
Important Terminology 132
Practice Questions 133
Practice Question Answers and Explanations 136
Takeaway 138
Refresher 139
Tips 139

6 ABAP Types and Data Objects 141

Objectives of This Portion of the Test 142
Key Concepts Refresher 143
ABAP Types and Data Objects 143
ABAP Data Types 151
Local Data Types 155
Global Data Types 158
Data Object Visibility 160
Important Terminology 160
Practice Questions 161
Practice Question Answers and Explanations 164
Takeaway 166
Refresher 166

7 Internal Table Definition and Use 169

Objectives of this Portion of the Test ... 170

Key Concepts Refresher ... 171
- Internal Table Definition and Use ... 171
- Defining ABAP Internal Tables ... 175
- Using ABAP Internal Tables ... 184

Important Terminology ... 201

Practice Questions ... 201

Practice Question Answers and Explanations ... 204

Takeaway ... 206
- Refresher ... 207

8 SQL Statements Including Update Strategies 209

Objectives of This Portion of the Test ... 211

Key Concepts Refresher ... 211
- Data Modeling ... 211
- Data Retrieval ... 214
- Arrangement of SELECT and FROM Clauses ... 217
- Unions ... 217
- Performance of Database Access ... 218
- Logical Units of Work ... 221
- Enqueue and Dequeue ... 223
- Inline Updates ... 228
- Perform on Commit ... 229
- Update Modules ... 229
- Open SQL ... 235
- SQL Parser ... 236
- SQL Expressions ... 237
- SQL Expressions Expanded ... 238
- SQL Functions Expanded ... 238
- Access to CDS Entities ... 241

Practice Questions ... 241

Practice Question Answers and Explanations ... 246

Takeaway ... 249
- Refresher ... 250
- Tips ... 250

9 Basic ABAP Programs and Interface Creation 251

Objectives of this Portion of the Test ... 253

Key Concepts Refresher ... 253
- Organizing Development ... 254
- ABAP Programming Overview ... 255
- ABAP Event Blocks ... 261
- Basic ABAP Statements ... 266
- ABAP Subroutine ... 269
- ABAP Function Module ... 270
- ABAP Classes and Methods ... 274
- ABAP Selection Screen ... 277
- Authorization Checks ... 279
- ABAP Dynpros ... 281
- Dialog Messages ... 283

Important Terminology ... 289

Practice Questions ... 291

Practice Question Answers and Explanations ... 296

Takeaway ... 299
- Refresher ... 300
- Tips ... 301

10 ABAP Dictionary 303

Objectives of This Portion of the Test ... 304

Key Concepts Refresher ... 305
- Overview ... 305

Basic and Complex Data Types ... 307
Transparent Tables ... 320
Global Temporary Table ... 326
Replacement Objects ... 327
Search Helps ... 328
Append Search Helps ... 332
Lock Objects ... 332
View Types and Maintenance ... 335

Important Terminology ... 338

Practice Questions ... 338

Practice Question Answers and Explanations ... 342

Takeaway ... 344
Refresher ... 344

11 Unicode 347

Objectives of This Portion of the Test ... 349

Key Concepts Refresher ... 350
Unicode Compliance ... 350
Unicode Tools ... 351

Important Terminology ... 362

Practice Questions ... 362

Practice Question Answers and Explanations ... 365

Takeaway ... 366
Refresher ... 366
Tips ... 367

12 Classical Screens 369

Objectives of this Portion of the Test ... 371

Key Concepts Refresher ... 371

Screen Design ... 372
GUI Status and Title Design ... 391
Table Control Programming ... 400

Important Terminology ... 408

Practice Questions ... 408

Practice Question Answers and Explanations ... 411

Takeaway ... 413
Refresher ... 413

13 Selection Screens 415

Objectives of this Portion of the Test ... 416

Key Concepts Refresher ... 416
Selection Screens ... 417
Selection Screen Design ... 417

Important Terminology ... 433

Practice Questions ... 434

Practice Question Answers and Explanations ... 436

Takeaway ... 438
Refresher ... 438

14 ABAP Object-Oriented Programming 439

Objectives of this Portion of the Test ... 441

Key Concepts Refresher ... 441
Object-Oriented Programming Concepts ... 441
ABAP Objects ... 443
ABAP Class ... 443
Objects ... 451
Attributes ... 451
Methods ... 453

Events ... 463

Practice Questions ... 467

Practice Question Answers and Explanations ... 470

Takeaway ... 472

Refresher ... 473

15 ALV Grid Control 475

Objectives of this Portion of the Test ... 476

Key Concepts Refresher ... 477

Overview of ALV Programming ... 477

ALV Grid Programming (CL_GUI_ALV_GRID) ... 478

ALV Object Model ... 488

Important Terminology ... 498

Practice Questions ... 500

Practice Question Answers and Explanations ... 503

Takeaway ... 504

Refresher ... 504

Tips ... 505

16 User Interfaces (Web Dynpro) 507

Objectives of this Portion of the Test ... 508

Key Concepts Refresher ... 509

Web Dynpro Design ... 509

Controllers ... 514

Contexts ... 516

Events ... 517

Web Dynpro Application ... 518

Graphical Elements ... 519

Important Terminology ... 525

Practice Questions ... 526

Practice Question Answers and Explanations ... 530

Takeaway ... 531

Refresher ... 531

Tips ... 532

17 Class Identification Analysis and Design 533

Objectives of This Portion of the Test ... 534

Key Concepts Refresher ... 535

Functional Methods ... 535

Inheritance ... 540

Interfaces ... 546

Constructor Expressions ... 548

Class Identification ... 549

Important Terminology ... 551

Practice Questions ... 552

Practice Question Answers and Explanations ... 556

Takeaway ... 558

Refresher ... 558

Tips ... 558

18 Enhancements and Modifications 559

Objectives of this Portion of the Test ... 560

Key Concepts Refresher ... 561

Enhancing SAP Applications ... 561

Enhancement Techniques ... 562

Enhancement Framework ... 588

Modification ... 591

Practice Questions ... 593

Practice Question Answers and Explanations 596

Takeaway 598

Refresher 599

19 Table Relationships 601

Objectives of this Portion of the Test 602

Key Concepts Refresher 603

Table Relationships 603

Value Help 610

Practice Questions 615

Practice Question Answers and Explanations 618

Takeaway 620

Refresher 620

Tips 620

The Authors 623

Index 625

Index

A

ABAP
- *basic statement* 266
- *data object* 143
- *data type* 142, 151, 166
- *data type,domain* 339
- *definition* 77, 255
- *event blocks* 261
- *global declaration* 256, 290
- *global definition* 256
- *obsolete additions* 256
- *procedural* 256
- *processing block* 258
- *processor* 65
- *runtime environment* 255, 261, 266, 544, 545
- *statement* 266, 536
- *subroutine* 269
- *syntax rules* 266
- *text translation* 256
- *type* 313
- *type of program* 256

ABAP application server 48, 61, 63, 66, 214
- *resource optimization* 67

ABAP class 443
- *attribute* 451
- *class events* 463
- *constructor* 460
- *create object* 451
- *event* 463
- *global class* 445
- *local class* 443
- *method* 453
- *private* 444
- *private instantiation* 449
- *private section* 447
- *protected* 444
- *protected instantiation* 449
- *protected section* 448
- *public* 444
- *public section* 447
- *raise exception* 459

ABAP class (Cont.)
- *read-only* 452
- *returning* 459
- *static method* 462

ABAP Debugger 86, 110, 112
- *architecture* 111
- *classic* 112, 113, 115, 117
- *flow control* 127
- *new* 113, 115–117, 126, 133
- *process call* 268
- *start methods* 114
- *starting* 111, 113, 115
- *stopping* 115
- *switching* 115
- *work areas/desktops* 117

ABAP Dictionary 66, 92, 303, 359, 525, 602, 611
- *activate table definition* 212
- *append structure* 359
- *check table* 602–604, 606, 607, 610, 611
- *create view* 611
- *customizing include* 359
- *data element* 283, 486, 491, 525, 602, 603, 611, 613
- *data element, individual field* 213
- *data type* 142, 162
- *domain* 213, 603
- *fixed value* 604, 606
- *foreign key* 602, 606–608, 610
- *input field* 277
- *interface parameters* 271
- *key concepts* 345
- *overview* 305
- *program development support* 603
- *search help* 602–604, 611
- *search helps* 328
- *structure enhancement* 359
- *table* 345
- *text table* 609, 612
- *tool* 307
- *transparent table* 212
- *value table* 602, 604, 606, 608
- *value table to check table* 607

ABAP Dictionary type ... 313
ACCP ... 313
CLNT ... 313
DATS ... 313
FLTP ... 313
predefined ... 158
ABAP Editor ... 91, 122, 127
configure settings ... 86
new ABAP Editor ... 75, 87, 89
open code ... 112
ABAP List Viewer → ALV
ABAP messaging channels (AMC) ... 54
ABAP Objects ... 111, 119, 256, 268, 274, 440
definition ... 443
reference ... 111
ABAP program
development tasks ... 253
ABAP push channels (APC) ... 55
ABAP Test Cockpit ... 81
ABAP Unit Browser ... 81
ABAP Workbench ... 76, 107, 510
change request creation ... 254
introductory statement ... 257
main screen ... 77
settings ... 86
tools ... 77, 91
Abstraction ... 442
Access mode ... 318
Alias ... 548
ALV ... 476
change layout ... 484
container ... 476–478, 480, 488, 496
currency ... 486
custom container ... 478
data sorting ... 484
display data table ... 481, 482, 490
display variant ... 482–484
event handling ... 476, 477, 486, 496
event registration ... 487, 488, 497
field catalog ... 481–485, 491, 498
field group ... 482
field label ... 604
filtering ... 482
full screen ... 476, 477, 481, 488
grid control ... 476, 478, 481
layout ... 482
object model ... 477, 493, 494
ALV (Cont.)
print output ... 482
sorting ... 482, 492
unit of measure ... 486
ALV Grid
sort data ... 481
ALV Grid Control
create instance ... 480
data order ... 484
ALV Object Model ... 488
American Standard Code for Information Interchange → ASC
Append ... 184, 185, 206
Application layer ... 61, 219, 223, 255
ASC ... 350
ASCII ... 350
Assert ... 111, 123
Assertions ... 125
Assigning ... 190, 191
Associate certification ... 31
Associate examination
competency area ... 34
specifics ... 33
At exit-command ... 393
At selection-screen ... 432
on ... 433
on block ... 433
on help-request for ... 433
on radio button ... 433
on value-request ... 433
output ... 424
ATC Result Browser ... 81
Attribute ... 274, 453, 473
instance ... 478, 488, 537
static ... 275, 489, 537, 543
Authorization check ... 252, 279
Authorization object ... 280

B

Back-End Editor ... 87
BAdI ... 561, 578
definition ... 579
Filter-depend ... 578
implementation ... 584
menu enhancement ... 586
multiple use ... 578, 580

BAdI (Cont.)
new ... 589
screen enhancement ... 587
searching ... 580
BAPI ... 59
definition ... 318
Begin of block
title ... 427
with frame ... 427
Binary search ... 192, 203
Breakpoint ... 111, 119, 123, 125, 132, 260
types ... 124
BSP ... 52
BTE ... 561, 575
FIBF ... 577
process ... 577
process interface ... 575
publish and subscribe ... 575
search ... 578
Buffer ... 66
Buffering ... 603
Open SQL ... 219
type ... 324
Business Application Programming Interface → BAPI
Business object ... 441
Business Process Platform ... 49
Business Server Pages → BSP
Business Transaction Event → BTE
Byte and character string processing ... 360
Byte type ... 350, 354, 356, 360, 362

C

Call method ... 460
Cardinality ... 609
Central instance ... 224
Certification
associate ... 31
professional ... 31
Certification exam
scoring ... 30
Certification level ... 30
advanced ... 32
Change and Transport System (CTS) ... 99
Change request ... 252, 254, 289
assign objects ... 255
project timeline ... 254
Change request (Cont.)
task ... 254, 289
Character type ... 151, 350, 354, 355, 357, 358, 360, 362
Check table example ... 608
Checkbox
define in program ... 420
Checkpoint ... 123, 125, 133
assertions ... 125
group ... 126
Class ... 440, 443, 473, 539
constructor ... 460
definition ... 444
friends ... 539, 540
inheriting from ... 541
public, private, and protected components ... 448
visibility ... 447
Class Builder ... 91, 453
method definition ... 457
Class Identification ... 549
CLASS-DATA ... 150
Class-events ... 463
Class-methods ... 454
Client ... 58
Cluster table ... 343
Code execution ... 111, 127
information ... 127
Code page ... 361
Collect ... 188
Component
lifetime ... 512, 515
Composite enhancement spot ... 590
Constructor ... 542
instance ... 481, 542
static ... 538, 542
Constructor Expressions ... 548
Context mapping ... 516, 517, 524
Control Framework ... 478
Controller
component controller ... 515
configuration controller ... 515
context controller ... 512
custom controller ... 515
view controller ... 516
window controller ... 516
Conversion routine ... 342

Course
BC400 ... 38
BC401 ... 38
BC402 ... 38
BC405 ... 38
BC414 ... 38
BC425 ... 38
BC430 ... 38
NET310 ... 38
NW001 ... 38
TAW10 ... 38
TAW11E ... 38
TAW12 ... 38
Create object ... 473
Customer exit ... 561, 566
Customer-function ... 568

D

Data
binding ... 517
encapsulation ... 268
modeling ... 210, 211, 280
process large volume ... 172
retrieval ... 210, 211, 214
Data element ... 158, 303, 305, 306, 315, 345
create ... 316
default component name ... 318
field label ... 316
parameter ID ... 316
search help ... 316, 604
technical attribute ... 308
Data object ... 143, 166
classic ... 112
definition ... 145
fixed value ... 145
form...endform ... 160
internal table ... 171
literal ... 145
local ... 162
modifying ... 111, 115, 121
module...endmodule ... 160
offset ... 356–358
predefined ... 147
viewing ... 111, 115, 118–120, 122, 128
visibility ... 160
Data type ... 143, 144, 166, 305, 309
basic ... 305
complex ... 155, 305, 307
declare incomplete ... 154
define structure ... 156
elementary ... 142, 153, 158
generic ... 141, 143
global ... 158, 162, 166
LCHR ... 310
local ... 155, 166
LRAW ... 311
MESH ... 157
parameters ... 144
predefined ... 143, 309
QUAN ... 311
RAWSTRING ... 311
select-options ... 144
SSTRING ... 312
STRING ... 311
string ... 153
TIMS ... 312
types ... 144
VARC ... 312
XSTRING ... 153
Database access ... 418
Database interface ... 65–67, 74, 214, 216, 220
Database layer ... 58–60, 64, 210, 219, 223, 255
Database lock ... 68, 111, 224
Database update ... 210, 211
asynchronous ... 233
bundled ... 223, 225, 228, 229
inline update ... 225, 228
local ... 233
synchronous ... 233
update task ... 225, 230
Database view ... 220, 306, 337
Debugger ... 110
old vs. new ... 120
scripting ... 129
Software Layer Aware Debugging ... 129
step size ... 129
tool ... 118
Decimal mantissa ... 152
Declaration position ... 148
Deep structure ... 141, 166, 342
Delete ... 196
Delivery and maintenance ... 320, 326

Dequeue ... 333, 338
Desktop ... 126, 128
tool ... 117, 118
Dialog Message → Message
Dialog program ... 370
Dialog programming concept ... 371
Dialog screen ... 413
event ... 413
Dialog step ... 63, 68, 223, 228
Diff tool ... 119, 128
Dispatcher ... 64
Domain ... 303, 305, 308, 345, 603
data element ... 315, 605
equality ... 608
fixed values ... 613
format ... 309
output property ... 309, 313
value table ... 606
Down cast ... 545
Dynpro ... 281

E

Empty table key ... 176
Encapsulation ... 442
Enhancement ... 93
ABAP Dictionary ... 587
append structure ... 587
BAdI ... 578
BTE ... 575
concept ... 599
customer exit ... 566
Customizing include ... 588
explicit enhancement point ... 588, 589
implicit enhancement point ... 589
menu exit ... 574
option ... 589
screen exit ... 572
search tool ... 599
spot ... 590
technique ... 562, 599
user exit ... 562
Enhancement Framework ... 93, 560, 561, 588, 599
composite enhancement ... 590
enhancement option ... 589
enhancement point ... 589
Enhancement Framework (Cont.)
enhancement spot ... 93, 590
new BAdI ... 589
Enhancement Information System ... 75, 93, 107, 599
Enqueue ... 333, 338
Enterprise extension ... 50
Entity relationship model (ERM) ... 211
Event ... 463, 473
declaration and implementation ... 464
sender ... 463
Event block ... 252, 262, 290
at line-selection ... 265, 288
at selection-screen ... 262, 265, 286, 287
at user-command ... 266, 288
details ... 262
end-of-page ... 265, 286
end-of-selection ... 264, 286
get node ... 262, 264, 286
initialization ... 263, 286
load-of-program ... 262, 286
start-of-selection ... 262, 263, 286
top-of-page ... 265, 286, 288
Examination
Strategy ... 42
Examine
processing logic ... 111
result ... 111
source code ... 111, 115, 118–120, 122, 123, 127, 128
Exclusive lock ... 334
Exclusive mode ... 116, 117, 133
External mode ... 113, 115, 133
External view ... 235

F

Field ... 381
OK_CODE ... 283
Ref_field ... 485, 486
Ref_table ... 485, 486
File interface ... 361
open dataset ... 361
Fixed point arithmetic ... 152
Flat structure ... 141, 166, 342
type ... 155
Flow logic ... 281

Foreign key 322, 606
domain equality 608
field types 609
relationship cardinality 609
table 607
Fragment view 350, 354, 357, 358, 362
Free 199
Friendship → class
Front-End Editor 88
Full buffering 324
Function Builder 91
Function keys 395
Function module 270, 306
exits 568
messages 289
Functional method 535
Functional type 392

G

Gateway 64
Generic buffering 324
Global class 443, 445
public instantiation 449
Graphical Screen Painter 376
GUI
creating GUI status 395
creating GUI title 395
setting GUI status 398
setting GUI title 398
status 370, 391, 414
title 370
title design 391

H

Hashed table 174, 178, 182, 318, 319
internal 207
Help view 338, 343
Hexadecimal type 152
Hook methods 516

I

Identifying error 110, 111
Index 306, 324
Index table 174, 178

Inheritance 442, 492, 540, 546, 551
friend 540
semantic difference 546
tree 541
Initialization 423, 431
Inline declaration 148
Inner join 335
Input check 306, 603
in ABAP dialog 386
Insert 185
lines 186
lines in internal table 185
table 177
Instance 57
attribute 453
component 450
Integer type 151
Interface 546, 552
Internal table 170, 207, 270
access mode 177
append 174
append line to 184
append summarized lines 188
append/modify 191
assigning 190
at...endat 194
binary search 192
clear 198
collect 188
column 172
define 175
define hashed 182
define in program, syntax 180
define sorted/hashed 182
definition 171
delete adjacent duplicates 197
delete lines 196
empty table key 176
endloop 193
features 171
hashed 270
hashed table 174
header line 173
index access 175
initial size 181
insert lines 185
key access 175

Internal table (Cont.)
line type 173
loop 193
modify 195
modify lines 195
MOVE-CORRESPONDING statement 187
non-unique key 180
occurs 202
populate 174
process lines sequentially 193
read lines 189
read table 189
refresh 198
secondary key 176
sort 197
sorted table 174, 270
standard 270
standard key 173
standard table 174
table key 173
table row 172
table type 177, 318
transporting 190
uniqueness of key 175
user-defined key 173
with header line 181, 183
without header line 173
work area 173
Internet Communication Manager 52
Internet Transaction Server 52
IS NOT BOUND condition 480

J

Java Connector 53
Join 335

K

Kernel and administration services 58
Key 318

L

Layout
FlowLayout 520
GridLayout 522

Layout (Cont.)
MatrixLayout 521
RowLayout 521
Like 150
Line type 318
Literal
constant 146
numeric 145
string 146
Literals 145
Local class 443
Lock object 48, 210, 211, 224, 304, 306, 332, 603
Dequeue 603
dequeue 223, 225, 228, 233
Enqueue 603
enqueue 223, 225, 228
enqueue work process 68
function modules 333
local lock container 226
lock mode 334
maintain 225
Lock parameter
collect 226
mode 226, 227
scope 226, 228
wait 226, 228
Log data change 324
LUW 48, 210, 211, 229
commit 222, 224, 229, 232, 233
database 64, 67, 221–224
definition 221
implicit commit 222–224
lock database 234
place changes in dialog step 223
rollback 221, 222, 231, 232
SAP 67, 231
update module error 234

M

Maintenance 335
Maintenance view 306, 337, 343
Memory analysis 119
Menu exit 574
Menu item, activate/deactivate 392

Menu Painter ... 92
create GUI status ... 392
Meshes ... 157
Message ... 252, 259, 283
class ... 284
number ... 284
placeholder ... 284
server ... 58, 61–63, 68
type ... 284
Method ... 441, 453, 473, 546
changing ... 459
default ... 459
exporting ... 459
importing ... 459
instance ... 275, 537, 541
optional ... 459
static ... 537, 543
Model View Controller → MVC
Modification ... 591
Modification adjustment ... 562
Modification Assistant ... 591
Modification Browser ... 592
Modify ... 195, 221
Modify table ... 177
Modularization unit ... 252, 256, 267, 268, 290, 496, 536
function module ... 256, 258, 261, 268, 270, 274, 289, 536
method ... 256, 258, 261, 268, 274, 289, 536
subroutine ... 256, 258, 259, 261, 268, 269, 271, 274, 496, 536
Module ... 381
Module pool program, create ... 406
MOVE-CORRESPONDING statement ... 187
Multiple inheritance ... 541, 546, 552
MVC ... 509
application structure ... 510
controller ... 510
model ... 510
view ... 510

N

Narrowing cast → Down cast
Nested structure ... 166
type ... 156
Node cardinality ... 516
Non-cumulative lock ... 334
Non-Unicode system ... 351
Numeric literal ... 145
Numeric type ... 151

O

Object ... 451, 473
context menu ... 79
instantiation ... 275
Object Navigator ... 78, 79
Object-oriented programming ... 440
concept ... 441
On ... 381
Open SQL
buffering ... 219
read database records ... 215
SQL Parser ... 236
statements ... 220
Outer join ... 335
Overloading ... 542

P

Package ... 95
global_switch ... 97
interface ... 97
nesting ... 96
pakparam ... 97
visibility ... 96
Package Builder ... 95, 96
PAI ... 371, 373
Parameter ... 144, 150, 259, 268
as checkbox ... 419
changing ... 258, 261, 269, 271, 274, 535
default ... 419
input ... 258, 261, 269, 271, 274, 487, 613
memory ID ... 419
obligatory ... 418, 419
output ... 258, 261, 269, 271, 274, 535, 613
parameter ID ... 317, 419
pass by reference ... 259, 261, 269, 290
pass by value ... 259, 261, 269, 273, 275, 290
radiobutton group ... 419
returning ... 258, 274, 277, 535
table ... 258, 269, 271
value check ... 420

Parameters statement ... 418
PBO ... 371, 373
Personalization ... 561, 562
Physical lock ... 224
Plug ... 513, 514
POH ... 373, 414
Polymorphism ... 442, 544
Pooled table ... 343
POV ... 373, 414
Predefined function for internal table ... 200
Presentation layer ... 61, 222, 223, 255
user view ... 62
Primary index ... 324
Private attribute ... 452
Procedure
block ... 262
exception ... 258, 270, 272, 274, 546
parameter ... 258
Process After Input → PAI
Process Before Output → PBO
Process interface ... 577
Process On Help-Request → POH
Process On Value-Request → POV
Processing block ... 256, 258, 286, 290
dialog module ... 258, 614
event block ... 258, 261
procedure ... 258
Professional certification ... 31
Program attribute ... 351
Program context ... 223
Program exits ... 568
Program type ... 285
class ... 257, 274
executable ... 256, 282
function group ... 256, 257, 270, 272, 282
interface ... 257
module pool ... 256, 282
type group ... 257
Projection view ... 337, 341, 343
Public attribute ... 452

Q

Quality management ... 82

R

Radio button ... 420
Read lock ... 334
Read table ... 177, 189, 191, 193
Redefinition ... 541
Refactoring Assistant ... 541
Reference data type ... 343
Reference table ... 322
Reference type ... 158
REFRESH ... 198
Relational operator ... 360
Remote Function Call → RFC
Repository Browser ... 82, 107
create package ... 95
Repository Information System ... 75, 84, 107
custommize selection screen ... 85
Repository object ... 60, 69, 74, 82, 254, 255
create/edit ... 83
search ... 84
Reuse component ... 215, 267, 270, 510
Reuse Library ... 79
RFC ... 59, 222, 256
Root component ... 519

S

SAP application server ... 48, 223, 224
SAP applications
enhance ... 561
SAP Business ByDesign ... 49
SAP Business One ... 49
SAP Business Suite ... 49
SAP Certified ABAP Development Associate ... 33
SAP Community Network (SCN) ... 39
SAP examination
Strategy ... 41
SAP GUI ... 59, 61, 64, 65
SAP HANA ... 61
as standard database ... 235
primary database ... 61
SQLScript ... 235
view ... 235
SAP Help Portal ... 39
SAP List Viewer → ALV
SAP Logon ... 62

SAP NetWeaver ... 49
server architecture ... 52
SAP NetWeaver Application Server ... 48, 351, 477, 510
ABAP ... 53, 56, 59, 61–64, 67
ABAP + Java ... 52, 56
Java ... 53, 56
SAP Notes Assistant ... 592
SAP Products ... 49
SAP S/4HANA ... 49
SAP Software Change Registration (SSCR) 591
SAP Web Application Server ... 50, 52
Screen ... 62, 271, 281, 405, 477
chain ... 381
container control ... 478, 481, 496
control ... 478, 481
define layout ... 375
design ... 372
Dynpro 64, 252, 277, 281, 478, 480, 496, 509
element attribute ... 376
endchain ... 381
endloop ... 381
exit ... 572
field ... 383
field help ... 387
function code ... 379
function type ... 392
loop ... 381
modal dialog box ... 252, 283
modify dynamically ... 405
module calls ... 379
on chain-input ... 384
on input ... 384
on request ... 384
process after input ... 371
process before output ... 371
processing ... 381
processor ... 65
show element list ... 378
start sequence ... 282
structure ... 405
type ... 375
value help ... 387
values ... 381
Screen flow logic ... 379, 380
at cursor-selection ... 384
at exit-command ... 385
Screen flow logic (Cont.)
F1 Help with dialog module ... 389
keywords ... 381
process after input ... 380
process before output ... 380
table control ... 401
Screen Painter ... 329, 370
create screen for dialog programming ... 372
design screen layout ... 373
screen attribute definition ... 374
Search help ... 304, 306, 328, 343
append ... 329, 332
attach to data element ... 613
attach to field ... 613
attach to table field ... 321
attachment ... 611, 613
collective ... 329, 331
data element ... 317
dialog behavior ... 611, 612
elementary ... 329
interface ... 611, 613
parameter ... 329, 331
selection method ... 611, 614
value list ... 611, 612
Secondary index ... 218, 324
Secondary table key
hash/sorted key ... 176
Select ... 215, 216, 219, 220, 614
array fetch ... 217
field list ... 216
from clause ... 215
into clause ... 215–217
join column ... 220
join condition ... 220
join table ... 220
loop ... 215, 216
select clause ... 215
single ... 215
where clause ... 215, 216, 218
Selection method ... 329, 330
Selection screen ... 144, 150, 252, 277, 372, 416, 417, 420
as subscreen ... 429
begin of block ... 427
begin of line ... 428
call selection-screen ... 427

Selection screen (Cont.)
code ... 278
design ... 417
end of block ... 428
end of line ... 428
formatting ... 427
high ... 421
low ... 421
memory ID ... 424
nesting level ... 431
no intervals ... 424
no-display ... 424
no-extension ... 423
obligatory ... 424
option ... 421
parameter ... 277, 416–418
processing ... 432
selection text ... 418
select-options ... 277, 416, 417
sign ... 421
skip ... 429
tab strip control ... 430
text element ... 418
uline ... 429
variant ... 427
SELECT-OPTIONS statement ... 420
Server ... 58
Session ... 62
Set handler ... 466
Shared lock ... 334
Shared memory ... 64
Singleton ... 538, 540
SLAD →Debugger, Software Layer Aware Debugging
Sort key, specify dynamically ... 197
Sorted table ... 174, 178, 182, 318, 319
internal ... 207
Specialization ... 540, 546
SQL ... 210
access method ... 218
ADBC ... 214
buffering ... 66, 213, 214, 219, 255
CDS ... 237
key field ... 212, 213, 215, 218
Native SQL ... 66
native SQL ... 67, 214
Open SQL ... 66, 214, 255
optimize ... 218
SQL (Cont.)
process rows ... 221
secondary index ... 213, 218
SQL expressions ... 237
SQL parser ... 236
Standard table ... 174, 178, 319
internal ... 207
Static attribute ... 453
Static component ... 450
String ... 153
String literal ... 146
Structure ... 213, 216, 306, 308, 318
Syntactically forbidden ... 550
System ... 58

T

Table ... 160, 280, 306
buffering ... 324, 326
DBTABPRT ... 324
Field ... 320, 327
join ... 220
primary key ... 213
secondary key ... 213, 218
transparent ... 212
transparent vs structure type ... 213
Table control
attribute ... 401
programming ... 400
screen flow logic ... 401
structure ... 403
Table Control Wizard ... 401
Table expression ... 199
using table key ... 199
Table type ... 156, 177, 306, 318
key category ... 179
line type ... 177
Tag Browser ... 80
Target → Select
Technical platform ... 50, 51
Technical settings ... 323, 326
Test Repository Browser ... 80
Text field literal ... 146
Text symbol ... 146
Tool ... 118, 123
change how opened ... 128
in work areas ... 126
object history ... 120

Tool (Cont.)
service 119
Variable Fast Display 128
Transaction
CMOD 567
FIBF 577
SE51 373
SE80 82
SMOD 567
UCCHECK 350, 352
Transparent table 305, 320
attributes 213
deep structure 339
definition 213
Transport Organizer 99, 101, 107
Transport → Change request
Transporting 193
Type 150, 155

U

Understanding
Comprehension 40
Effective memory 39
Mnemonics 40
Testing 40
Unicode 266, 348
alignment 349, 354, 356, 358
byte order 356
character set 351
check 348, 350, 351
compliant 349
enforce check 349
fragment view 358, 359
program 256, 351, 352, 354, 355, 359
syntax 349, 350, 352
system 351, 352, 362
tool 350, 352
Up cast 543
Update module 67, 229, 232
code 232
programming rules 234
V1 232, 234
V2 232, 234
User exit 561, 564
view 565

V

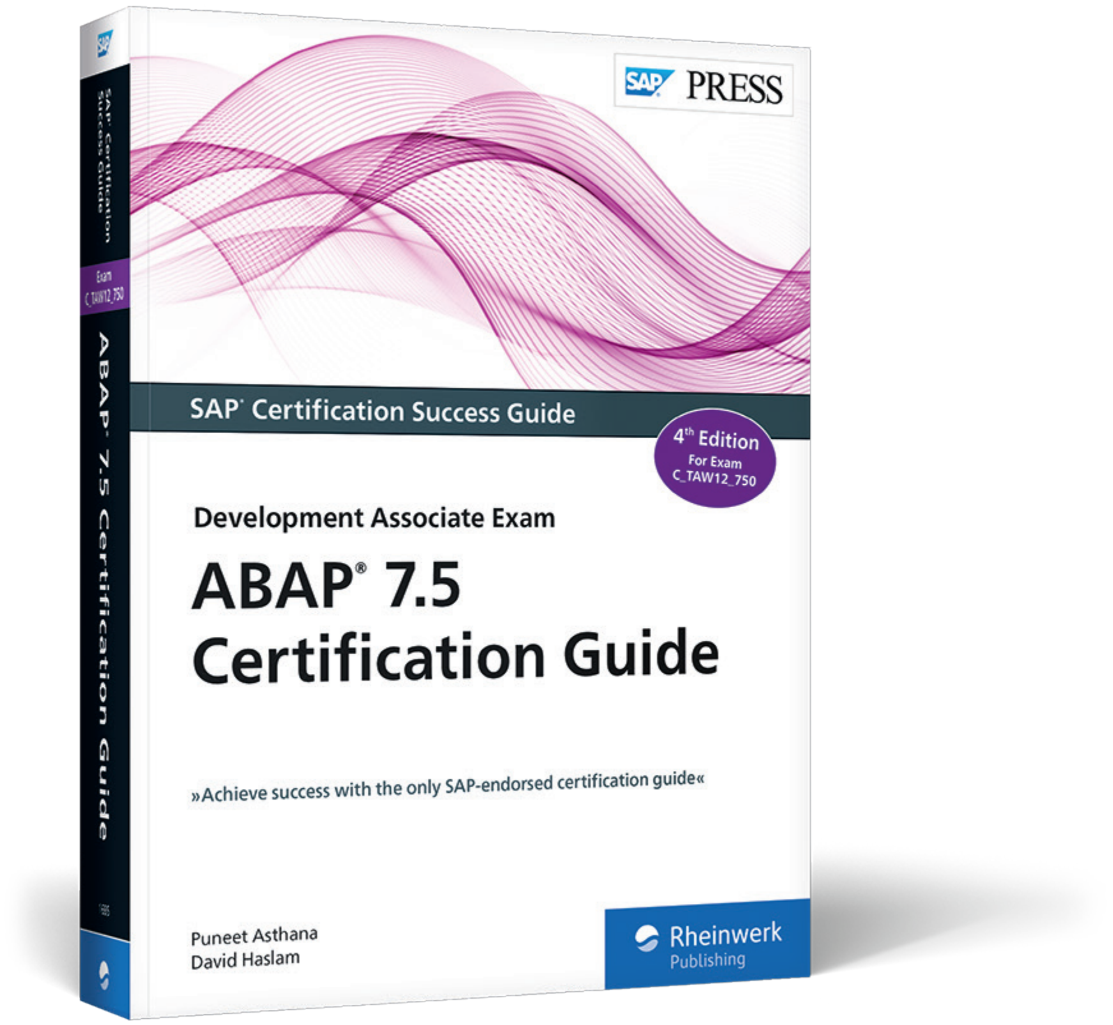
Value 150
Value help 603, 610
Variable 148
inline declaration 148
STATICS keyword 149
View
container 520
context 513, 515, 524
controller 513, 515, 523, 524
define 335
layout 519, 520, 524
type 335
UI element 511, 517, 519, 520, 522, 524
Visibility section 542, 547
private 538, 539, 542, 543
protected 539, 543
public 538, 539, 543, 547

W

Warning message 386
Watchpoint 111, 123, 132
Web Application Builder 92
Web Dynpro 508
application 518
component 510, 514
component interface 511, 518
controller 511, 512, 514, 515
event handling 513, 517
Explorer 93
Text Browser 80
view 511, 513, 514
window 511, 514
WebDynpro 53, 62
Widening cast → Up cast
Work process 58, 63, 64, 223, 233
determine number of 67
register 64
Worklist 79
Write lock 334

X

XSTRING 153


SAP PRESS
SAP Certification Success Guide
4th Edition For Exam C_TAW12_750
Development Associate Exam
ABAP 7.5 Certification Guide
»Achieve success with the only SAP-endorsed certification guide«
Puneet Asthana
David Haslam
Rheinwerk Publishing

Puneet Asthana, David Haslam

ABAP 7.5 Certification Guide: Development Associate Exam

636 Pages, 2018, $69.95
ISBN 978-1-4932-1685-7

www.sap-press.com/4605

Puneet Asthana is a UX and mobility architect and a recognized expert in ABAP development; he has more than 18 years of development experience, and has been working for SAP for over 15 years. Apart from ABAP development, he is also an expert in SAP Fiori, mobility, SAP Cloud Platform, SAP Cloud Platform Integration, IDocs, ABAP on SAP HANA development, ALE, EDI, workflow, and process integration.

David Haslam is a recognized expert in ABAP development, having worked for SAP for more than 23 years. David has led or participated in more than seven full lifecycle implementations of SAP, which include several multiple-phase projects and four large development projects. He was awarded the prestigious status of SAP Platinum Consultant in 2001, and he enjoys helping others through workshops and white papers, and sharing his knowledge and experience.