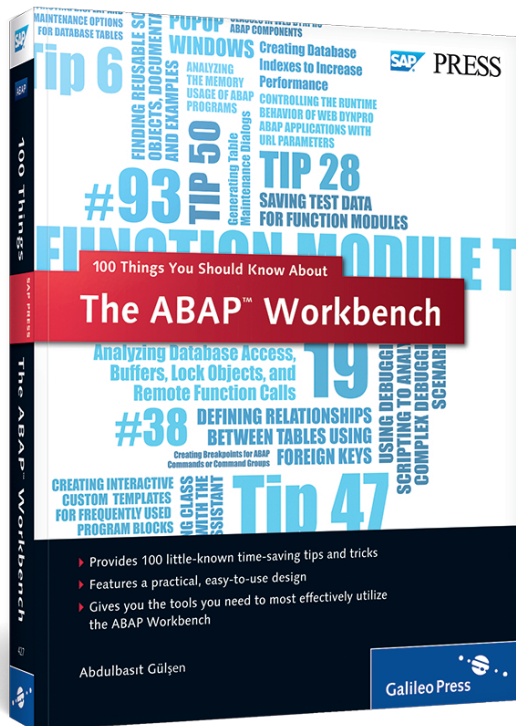


Abdulbasit Gülşen

100 Things You Should Know About ABAP® Workbench



 Galileo Press

Bonn • Boston

Contents at a Glance

1	Object Navigator	15
2	ABAP Editor	69
3	Function Builder	97
4	Class Builder	107
5	ABAP Debugger	121
6	Analysis Tools	169
7	ABAP Data Dictionary	201
8	Enhancements	271
9	Web Dynpro ABAP	313

Contents

Acknowledgments	11
Introduction	13
PART 1 Object Navigator	15
1 Building Package Hierarchies to Organize Development Objects	17
2 Using the Reuse Library to Find Reusable Software Objects, Documentation, and Examples	21
3 Accessing Your Previous Navigation Steps with the Navigation Stack ...	25
4 Inserting Statement Patterns in ABAP Programs with Drag and Drop ...	28
5 Using Worklists to Group Development Objects	31
6 Managing Your Frequently Used Objects with a Favorites List	33
7 Comparing ABAP Programs between Two Systems	36
8 Modifying and Testing Programs with Inactive Versions of Development Objects	39
9 Creating Local Objects for Test Purposes	43
10 Creating and Accessing Documentation for Development Objects	45
11 Reserving Namespaces with SAP for Third-Party Objects	48
12 Using the Application Hierarchy Tool to Organize Applications	50
13 Searching for Objects in Transport Requests with Transport Organizer Tools	52
14 Searching for Development Objects using the Repository Information System	55
15 Using OO Transactions to Link Class Methods to Transaction Codes	58
16 Using Forward Navigation to Create Objects	61
17 Uploading/Downloading User-Specific Settings to a Different System ...	63
18 Using Package Interfaces to Create a Set of Visible Development Objects	65
PART 2 ABAP Editor	69
19 Comparing ABAP Programs with the Splitscreen Editor	70
20 Viewing and Modifying Two Parts of the Same Code at Once	73
21 Using Interactive Code Templates for Frequently Used Code Blocks	75
22 Using Enhanced Copy and Paste Functionalities	79
23 Searching in Real Time with Incremental Search	81
24 Using Improved Navigation Features in the ABAP Editor	83
25 Creating Custom Statement Patterns	86

26	Formatting Source Code with Pretty Printer	89
27	Using Code Hints as Prompts When Writing Code	91
28	Using Code Completion to Complete Statements	94
PART 3 Function Builder		97
29	Saving Test Data for Function Modules	98
30	Running Function Modules Successively with Test Sequences	100
31	Creating and Using Remote-Enabled Function Modules	102
32	Using Predefined RFC Destination BACK to Call Function Modules	105
PART 4 Class Builder		107
33	Maintaining Classes with the Source Code-Based Class Builder	108
34	Renaming Methods of Classes Consistently with the Refactoring Assistant	111
35	Using Persistent Classes to Access Database Tables	114
36	Managing Exceptions with Exception Classes	117
PART 5 ABAP Debugger		121
37	Using SAP GUI Shortcuts to Debug Popup Windows	123
38	Debugging Background Jobs	126
39	Setting Breakpoints for ABAP Commands and Command Groups	129
40	Using External Breakpoints to Debug External Calls	132
41	Customizing the ABAP Debugger Desktop Tabs	134
42	Using the Diff Tool to Compare Complex ABAP Data Structures	138
43	Viewing and Manipulating Internal Tables Using the Table Tool	141
44	Saving Test Data for Function Modules in the ABAP Debugger	144
45	Using Watchpoints to Monitor Variable Changes	146
46	Using Debugger Scripting to Analyze Complex Debugging Scenarios ...	150
47	Debugging Specific Program Areas Using the Software Layer-Aware Debugger	154
48	Using Conditional Breakpoints to Check Specific Conditions	159
49	Using Forward Navigation Features in the ABAP Debugger	162
50	Analyzing Deep Nested Objects in the Main Object	165
PART 6 Analysis Tools		169
51	Performing Detailed Checks on ABAP Programs with Extended Program Check	170
52	Checking ABAP Programs for Naming Conventions with the Code Inspector	174
53	Testing and Improving the Quality of ABAP Programs with Unit Tests ...	178

54	Using the ABAP Runtime Analysis to Measure the Performance of an ABAP Program	180
55	Using Checkpoint Groups to Activate and Deactivate Checkpoints	183
56	Analyzing the Memory Consumption of ABAP Programs	187
57	Analyzing Database Accesses in ABAP Programs Using the Performance Trace Tool	190
58	Finding the Right Event to Trigger a Workflow	194
59	Using the Work Item Selection Tool to Analyze Workflow Logs	197
PART 7 ABAP Data Dictionary		201
60	Configuring Display and Maintenance Options for Database Tables	203
61	Generating Table Maintenance Dialogs for Database Tables or Views ...	205
62	Creating and Using Foreign Keys to Define Relationships between Database Tables	208
63	Using Foreign-Key Relationships to Create Maintenance Views	212
64	Assigning Value Tables to Domains to Propose Foreign Keys for Database Fields	215
65	Adjusting Screen Elements with Conversion Routines	218
66	Creating a Secondary Index to Improve Table Access Performance	221
67	Extending Table Maintenance Dialogs with Events	225
68	Creating View Clusters to Group Maintenance Dialogs Together for Better Maintenance	228
69	Using Delivery Classes to Control the Transport Behavior of the Database Table Data	232
70	Displaying and Analyzing Table Relationships in a Graphic	235
71	Logging Data Changes in a Database Table	238
72	Linking Text Tables to Main Tables to Use Multi-Language Applications	241
73	Using Buffering Options for Database Tables to Improve System Performance	244
74	Using Lock Objects to Control Multi-User Access to Table Records	248
75	Creating Alternative Search Help Paths with Collective Search Helps	251
76	Using Domains to Define Value Ranges for Database Tables and Structure Components	254
77	Attaching Search Helps Directly to Data Elements for Global Availability	257
78	Adding Date Fields to Make Time-Sensitive Table Maintenance Dialogs	259
79	Using the Database Utility to Transfer Structural Changes to the Database System	262

80	Defining Ranges Using Range Table Types	265
81	Using the Data Modeler to Create Data Models According to the SAP SERM Method	267
PART 8	Enhancements	271
82	Enhancing Standard Objects with Implicit Enhancement Options	273
83	Creating Composite Enhancement Implementations to Group Enhancement Implementations Hierarchically	277
84	Using Nested Enhancements in Existing Enhancement Implementations	280
85	Using Enhancements When Modifying Standard ABAP Programs	283
86	Activating or Deactivating Enhancements with the Switch Framework	285
87	Adjusting Enhanced Objects When Upgrading the SAP System	288
88	Using the Enhancement Category to Restrict Table and Structure Enhancements	290
89	Creating Multiple-Use Business Add-Ins	293
90	Using Filters to Select Between Multiple BAdI Implementations	296
91	Finding BAdIs in SAP Transactions Using the ABAP Debugger	299
92	Creating Customized Transactions with Transaction Variants	302
93	Using Parameter Transactions to Create a Transaction for Table Maintenance Dialogs	306
94	Using SET/GET Parameters to Assign Default Values for Screen Elements	308
PART 9	Web Dynpro ABAP	313
95	Controlling the Runtime Behavior of Web Dynpro ABAP Applications with Application Parameters	314
96	Tailoring Web Dynpro ABAP Applications to User Groups with Application and Component Configuration	317
97	Customizing Logon Screens for Web Dynpro ABAP Applications	321
98	Enhancing Web Dynpro ABAP Applications in the Enhancement Framework	325
99	Debugging Web Dynpro-Specific Program Entities	328
100	Assigning a Transaction Code to a Web Dynpro Application Using a Parameter Transaction	330
	The Author	333
	Index	335

Introduction

SAP is continuously adding new products and solutions into the product portfolio to keep the platform up to date and meet the latest technology trends. Although some of these new products need to be developed in different programming languages, ABAP® still stands as a major programming language for software development in the SAP platform. It is a simple but powerful language for developers to build enterprise applications.

Several tools are available in the ABAP Workbench that make the development process easy and efficient. In this book, you'll find 100 little-known tips and tricks that teach you quick and practical techniques to solve your problems and increase your productivity while using the ABAP Workbench tools. This book neither gives detailed tutorials explaining how to use these ABAP Workbench tools nor teaches the ABAP programming language. Each tip has only 3-5 pages that focus on the specific problem or topic that you can immediately use in your daily work. If you want to get more background detail about any tool explained in the book, we recommend searching the SAP PRESS catalog for another book focused on the tool or refer to the SAP Community Network (<http://scn.sap.com>) or official documentation provided by SAP (<http://help.sap.com>).

The ABAP Workbench has many tools and features that cannot be limited to 100 tips. Tips that are included in the book are selected from the wide range of topics mostly to give you workarounds to perform your daily tasks in a more efficient way. Some tips in the book require expert-level knowledge, and some can be very simple for ABAP experts, but generally they are selected to address both beginner and advanced ABAP programmers. You can put this book into your library and consult it when you want to read tips about specific tools, or you can read from start to end. I was excited to learn many of these tips, and I hope you'll feel the same when you read tips that will save you time and effort on a daily basis.

Tips in the book are divided into nine different parts. In the first four parts, you'll find tips that mostly help you to improve your ABAP development capabilities by providing practical, alternative workaround solutions while you're using the ABAP Workbench tools to develop an ABAP program, such as the ABAP Editor, the Function Builder, and the Class Builder.

In Part 5, we move into the analysis phase of ABAP programming. In this part, you'll find very useful tips on the ABAP Debugger to help you find bugs in your ABAP programs. You'll see how the new ABAP Debugger is improved after it's launched. You'll also notice the latest improvements of the ABAP Debugger in SAP NetWeaver 7.3. We've added a footnote about the version restrictions in the individual tip title in all that apply.

You might also see some differences in the step-by-step procedures or screenshots because of the version difference. We've used the SAP NetWeaver 7.3 system to describe the step-by-step procedures and provide the screenshots. If you're using a different version, please note that there may be small differences on the tools.

Part 6 is also focused on problem analysis in ABAP programs. You'll find tips for various analysis tools in the ABAP Workbench. You'll learn practical ways of improving the code quality and find performance bottlenecks arising from wrong database or memory usages. There are also useful tips that will help you to analyze SAP Business Workflow problems.

In Part 7, you'll learn practical ways of using the ABAP Data Dictionary tools. The ABAP Data Dictionary has very broad features that can't be limited to a small part in a book. This part lists the most useful and practical tips on using the ABAP Data Dictionary tools.

Enhancement tips listed in Part 8 are included to give you practical ways of using the Enhancement Framework, which is also one of the most useful parts of the ABAP Workbench. In this part, you'll see different usages of the Enhancement Framework that will save a lot of time during patches or system upgrades.

Finally, in Part 9, you'll see some tips on Web Dynpro ABAP. Of course, it's impossible to fit tips about Web Dynpro ABAP in a small part of this book. We've added some tips about Web Dynpro ABAP that are mostly related to the other ABAP Workbench tools. If you want more details, we highly recommend the books *Getting Started with Web Dynpro ABAP* (SAP PRESS, 2010) and *Web Dynpro ABAP—The Comprehensive Guide* (SAP PRESS, 2013).

Part 5

ABAP Debugger

Things You'll Learn in this Section

37	Using SAP GUI Shortcuts to Debug Popup Windows	123
38	Debugging Background Jobs	126
39	Setting Breakpoints for ABAP Commands and Command Groups	129
40	Using External Breakpoints to Debug External Calls	132
41	Customizing the ABAP Debugger Desktop Tabs	134
42	Using the Diff Tool to Compare Complex ABAP Data Structures	138
43	Viewing and Manipulating Internal Tables Using the Table Tool	141
44	Saving Test Data for Function Modules in the ABAP Debugger	144
45	Using Watchpoints to Monitor Variable Changes	146
46	Using Debugger Scripting to Analyze Complex Debugging Scenarios	150
47	Debugging Specific Program Areas Using the Software Layer-Aware Debugger	154
48	Using Conditional Breakpoints to Check Specific Conditions ...	159
49	Using Forward Navigation Features in the ABAP Debugger	162
50	Analyzing Deep Nested Objects in the Main Object	165

Debuggers allow developers to locate and fix any bugs or problems that exist in the source code. You have two options available to find these problems:

1. Go through the logical steps of the program to see the state of variables step by step.
2. Use the debugger to analyze and extract the logic of the program.

In SAP, you can debug any ABAP program, whether it's custom developed or a standard program. This section provides tips, tricks, and techniques to improve your experience when using the ABAP Debugger and make it easier to analyze and find bugs in your ABAP programs.

Using SAP GUI Shortcuts to Debug Popup Windows

You can switch on debugging for popup windows by creating an SAP shortcut file for command /h and dragging this file onto a popup window.


Normally, if you know the place in the source code that you want to debug, you can easily place a breakpoint there, and a debugger will be triggered when you execute the program and the position that you marked is reached. You can also switch on debugging at any screen by entering the /h command into the command field on the SAP GUI toolbar. However, you can't use the /h command to stop popup windows because there is no command field. In this tip, we'll show you how to bypass this problem by using SAP GUI shortcuts to trigger the /h command on popup windows.

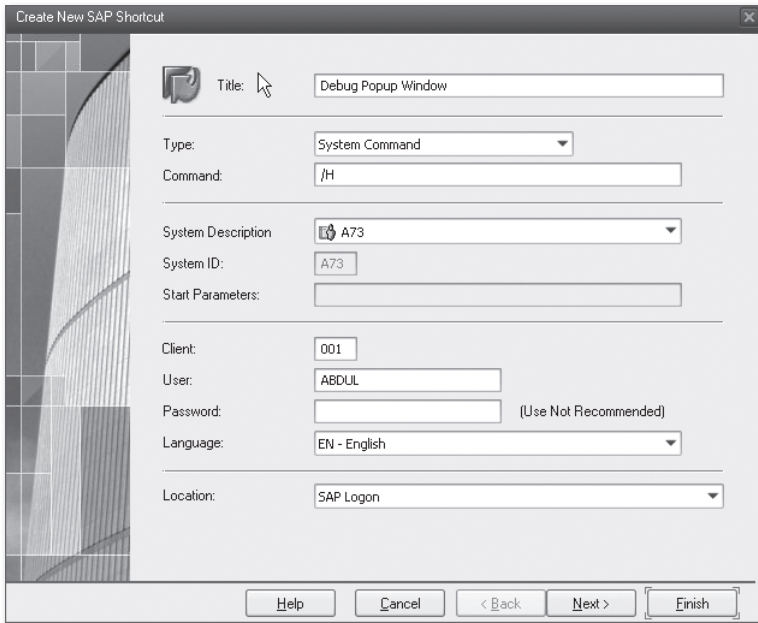
✓ And Here's How ...

To switch on debugging for popup windows, you need to create an SAP GUI shortcut file. The SAP GUI shortcut is a file that can be used to execute transactions or system commands directly from a file.

You can use three different methods to create an SAP GUI shortcut, which we discuss in the following sections.

Method 1

Click the GENERATE A SHORTCUT button  on the SAP GUI toolbar. The window in Figure 1 appears; enter the details of the shortcut.



📌 *Figure 1* Creating an SAP GUI Shortcut

Follow these steps:

1. Enter a title in the TITLE field.
2. Select SYSTEM COMMAND from the TYPE dropdown.
3. Enter "/h" in the COMMAND field.
4. Click FINISH.

A shortcut file with an extension .sap is created on the desktop.

Method 2

1. Right-click on the Windows desktop.
2. Select NEW – SAP GUI SHORTCUT.
3. Enter the name of the file and press .
4. Right-click on the file, and select EDIT from the context menu.
5. Select SYSTEM COMMAND from the TYPE dropdown.
6. Enter "/h" in the COMMAND field.
7. Click OK.

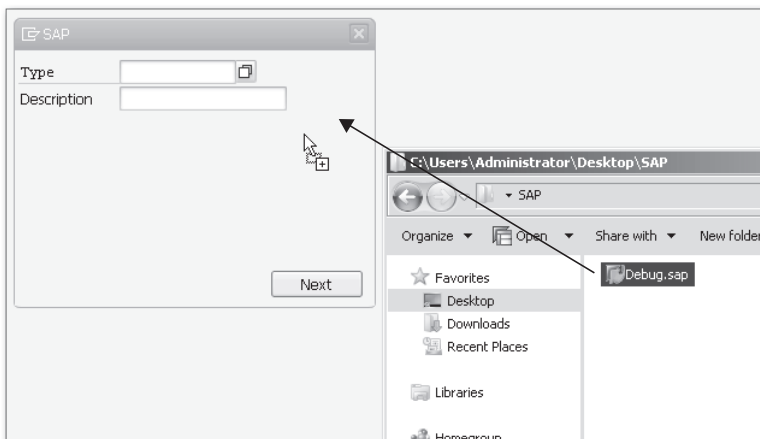
Method 3

Create a text file named *Debugger.sap*, open it in any text editor (e.g., Notepad), and write the following text into the file:

```
[Function]
Command=/H
Title=Debugger
Type=SystemCommand
```

Debug the Popup

Now you have created a SAP GUI shortcut file. Open the popup window that you want to debug, and drag the shortcut file onto the popup as shown in Figure 2.



⤴ **Figure 2** Dragging a Shortcut File onto a Popup Window

When you drop the file on the popup window, you'll see the **DEBUGGING SWITCHED ON** message on the status bar. Now you can start the debugger tool by pressing the key.

Index

A

ABAP Data Dictionary, 202
ABAP Debugger, 187, 299
 desktop, 134
ABAP layout standards, 89
ABAP object, 165
ABAP object-oriented statements, 150
ABAP programs, 70
 compare, 36
 improve quality, 178
Action, 325
Adjust database, 264
Adjust enhancement implementation, 288
Adjustment state, 289
Agent class, 115
ALPHA conversion, 220
ALV functions, 198
ALV Grid Control, 22
Append structure, 290
Application configuration, 317
Application Hierarchy
 custom, 50
 SAP, 50
 tool, 50
Application parameters, 314
Application server, 133, 244
Assert statement, 183
Attribute, 325
 change for all applications, 315
 change for single application, 314
Auto completion, 96

B

Backup, 63
BAPI, 56, 100
BAPI_TRANSACTION_COMMIT, 100
Base class, 115

Bookmark, 31, 83, 84
Bottom-up approach, 268
Breakpoint, 152, 300
 at statement, 129
Break-point statement, 183
Buffer, 244
Buffering
 generic area, 246
 options, 245
 single record, 246
Business Add-In (BAI), 94, 293, 299
 classic, 299
 create definition, 294
 create subobject, 296
 filter, 296
 new, 299
Business function, 286
 reverse, 286
Bypassing buffer, 247

C

Capture Active Job, 127
Cardinality, 210, 212, 269
Change and transport management system,
 52
Check field, 211
Checkpoint group, 183
Check table, 208, 215, 235
cl_abap_memory_utilities, 188
Class, 62
 CL_ICF_SYSTEM_LOGIN, 323
 CL_EXITHANDLER, 299
Class Builder, 58
Client copy, 232
Clipboard, 77
 buffers, 80
 ring, 79
Close tool, 135

Code

- blocks*, 75
 - completion*, 94
 - duplicate*, 73
 - Hints*, 91
 - modify*, 73
 - patterns*, 86
- Code templates
- create*, 75
 - view/modify*, 77
- Collective search help, 251
- Collision check, 250
- Command group, 129
- Comment blocks, 76
- Commit, 100
- Compare ABAP data structures, 138
- Compare selected systems, 71
- Comparison, 72, 138
- Component configuration, 317
- Component usage, 326
- Composite enhancement implementation, 277
- Condition
- link*, 156
 - stop*, 156
- Conditional breakpoint, 159
- Configuration editor, 318
- Context node, 325
- Controller usage, 326
- Conversion process, 264
- Conversion routine, 218, 220
- Create Breakpoint Condition option, 159
- Customer, 324
- hierarchy*, 51
 - namespace*, 291
- Customizing, 53
- include structure*, 290
 - table*, 205
- Custom parameter, 315
- Custom statement patterns, 86

D

- Database access, 190
- Database server, 244

- Database tables, 114
- Database utility, 262
- Data element, 62, 257
- Data records, 238
- Debug, 122, 328
 - background job*, 126
 - popup window*, 123
 - running background jobs*, 127, 128
 - scheduled or finished background jobs*, 126
- Debugger Desktop, 328
- Debugger.sap, 125
- Debugger Script Services, 152
- Declare, 192
- Delivery class, 232
- Delivery options, 203
- Dependency control list, 67
- Dequeue, 250
- Desktop, 134
- Development license, 49
- Development node, 51
- Development object, 65
 - documentation*, 45
 - group*, 31
 - status*, 39
- Development package, 17
- Display and maintenance options, 203, 205
- Display format, 219
- Documentation, 45
- Domain, 62, 215, 218, 254
- Download, 63
- Drag and drop, 28
- Drill-down navigation, 25

E

- End user, 169
- Enhanced copy and paste, 79
- Enhancement
 - implementation*, 277
 - layer*, 282
 - nested*, 280
 - point*, 283
 - spot*, 284, 293, 296
- Enhancement Category property, 290
- Enhancement Framework, 272, 325

Enqueue, 250
 Environment Analysis, 57
 Error message, 119
 customize, 210
 Evaluate logs, 239
 Event, 225, 325
 block, 60, 89
 customize, 225
 subroutine, 226
 Exception, 117
 classes, 117
 handling, 117
 texts, 118
 Exclusive lock, 249
 extended, 249
 Exec, 192
 Explicit enhancement option, 273
 Explicit enhancement point, 293
 Exposed object, 66
 Extended paste, 79
 Extended Program Check, 170
 Extended table maintenance, 234
 event, 225
 Extended trace list, 192
 External breakpoint, 132, 328

F

Favorites, 31, 33
 list, 33
 Fetch, 192
 Field dependency, 230
 Filter, 296
 combination, 297
 Foreign key, 208, 212, 215
 relationship, 212, 235, 242
 Form-based class builder, 108
 Form statements, 90
 Forward navigation, 61, 162
 Full screen, 135
 Fully Buffered option, 246
 Function Builder, 97, 144
 Function module, 56, 144
 call, 102

G

GET BADI, 298, 300
 GET parameter, 310
 Global class, 108
 Global runtime environment, 42
 Go to last change, 84
 Go to Line, 83
 Graphical PC Editor, 45

H

Hierarchical list, 43
 Hierarchical tree, 28
 HTML Viewer, 22
 HTTP, 132

I

Implicit enhancement
 explicit enhancement, 280
 Implicit enhancement option, 273
 explicit enhancement option, 283
 show, 275
 Inactive Object list, 40
 Inbound plug, 326
 Incremental search, 81
 Initial screen
 skip, 330
 Input conversion
 output conversion, 218
 Insert statement patterns, 86
 Inspection, 174
 Interactive templates, 75
 Interface, 294
 create, 66
 Internal format, 219
 Internal table, 141

J

Job overview, 126

K

Keywords, 75

L

Language import, 232
Language key, 242
Line editor, 46
Line number, 83
Local object, 43
Local runtime environment, 39, 42
Lock object, 248
Log data changes, 238
Logging indicator, 238
Logical unit of work, 100
Logon language, 241
Logon screen
 customize, 321
Log-Point statement, 183

M

Main package, 17
Maintain options, 203
Maintenance cluster, 228
Maintenance view, 212
Mapping Assistant, 115
Marker, 31
Maximize horizontally, 136
Maximize vertically, 135
Memory snapshot, 187
Modularize, 17
Module statements, 90
Monitor attribute changes, 148
Monitor the change
 monitors, 238
Multi-language application, 241
Multi-user access, 248

N

N
 1 dependency, 213
 C cardinality, 214

Namespace, 48
Naming conventions, 174
Naming standards, 48
Navigation link, 325
 tool, 25
Navigation window, 26
Nested enhancements, 280
Network load, 244
Non-unique index radio button, 223

O

Object Navigator, 16, 25, 28, 31, 33, 43, 51, 114, 277, 309
Object-oriented (OO) application, 267
Object-oriented (OO) programming, 58, 107
 model, 114, 117
Object-oriented (OO) statements
 ABAP, 150
Object profile, 155
Object set, 154
One-step maintenance, 206
OO transaction model, 58
Open SQL, 190
Optimistic lock, 249
Outbound plug, 326
Out-of-the-box functionality, 272

P

Package, 43, 50, 277, 285
 \$TMP, 43
 hierarchy, 17, 50, 277
 interface, 65
Package Builder, 17, 18
Parameter
 configure globally, 315
 transaction, 23, 24, 306
Partner, 324
Pattern, 28
 settings, 30
Persistent class, 114
Point of entry, 158
Point of exit, 158

Popup window, 123
 Predefined object set, 157
 Prepare, 192
 Primary index, 221
 Primary key, 221
 Private library, 24
 Process overview, 126
 Program
 bcalv_grid_01, 26
 DEMO_OO_TRANSACTION, 59
 Programming interface, 67
 Pseudo comment, 173
 Public library, 23

R

Range, 265
 Readability, 174
 Real-time searches, 81
 Reexec, 192
 Refactoring Assistant, 111
 Relational database, 208
 design, 212
 Remote Comparison tool, 36
 Remote-enabled function module, 102
 Remote system, 36, 105
 Rename class methods, 111
 Reopen, 192
 Repair license, 49
 Repository Information System, 55
 Restore, 63
 Reuse, 86
 Reuse Library, 21
 create, 23
 Reuse product, 21
 create, 23
 RFC, 102, 132
 destination, 36, 71, 102, 105
 function, 56, 105
 interface, 105
 Runtime behavior, 314
 Runtime object, 262, 329

S

SAP Business Workflow, 194, 197
 SAP GUI, 328, 330
 shortcut, 123
 SAP NetWeaver 7.0, 299
 SAP NetWeaver 7.3, 108, 150, 154, 159,
 162, 280
 SAP shortcut file, 123
 SAP Structured Entity Relationship Model
 (SERM), 267
 SAP Support Portal, 48
 Save layout, 137
 Screen fields
 adjust, 218
 Screen variant, 304
 Script, 150
 Script Wizard, 152
 Search help, 62, 251, 257
 Secondary index, 221
 Selection set, 155
 Semantic attributes, 210
 Services of the Tool, 136, 143
 SET/GET parameters, 308
 Shared lock, 249
 Simultaneous access, 248
 Single sign-on (SSO), 330
 Source code, 70
 check, 170
 format, 89
 surround with comments, 75
 Source code-based Class Builder, 108, 111
 Special tools, 328
 Split bar, 73
 Splitscreen Editor, 37, 70, 289
 SQL Trace, 190
 S_TABU_DIS, 206
 Standard ABAP program
 enhance, 273
 modify, 283
 Structure, 62
 package, 17
 Structured row type, 266
 Subpackage, 20
 Superordinate composite enhancement
 implementation, 278

Superpackage, 19
 Support package, 288
 Suspend navigation, 162
 Swap, 136
 Switch Framework, 285
 Sy-batch, 127
 Syntax Check function, 170
 System command, 124
 System logon configuration, 321
 System parameters, 238
 System performance, 244
 System upgrades, 325

T

Table, 62
 auditing, 238
 maintenance, 203
 maintenance dialog, 214, 225, 228, 259, 306
 type, 265
 Tag, 76
 CursorPosition, 76
 SurroundedText, 77
 Technical documentation, 45
 Technical settings, 238
 Templates, 86
 Test data, 98
 directory, 99, 144
 Test interface, 98
 Test sequence, 100
 Test tools, 169
 Text edit, 23
 Text table, 241
 Theme, 323
 Time-dependent database table, 259
 Time-dependent record, 259
 Title node, 51
 Toggle bookmark, 85
 Tool
 ABAP Runtime Analysis, 180
 ABAP Unit, 178
 Analysis, 169
 Code Inspector, 174
 Compare Variables, 149

Tool (Cont.)
 Data Browser, 203
 Data Explorer, 162, 163, 165
 Data Modeler, 267
 Debugger Scripting, 150
 Detailed Display, 162
 Diff, 138, 149
 Diff, history, 140
 Event Trace, 194
 Generate Table Maintenance Dialog, 205
 Graphical representation, 235
 Maintain Table Views, 203
 Memory Analysis, 187
 Memory Inspector, 187
 Objects, 165
 Performance Trace, 190
 Pretty Printer, 89
 Replace, 135
 Software Layer-Aware Debugger, 154
 Split View, 73
 Structures, 142, 165
 Table, 141
 Table Maintenance Generator, 205, 225
 Variable Fast Display, 138, 142, 144, 162, 166
 Web Dynpro, 328
 Work Item Selection, 197
 Top-down approach, 268
 Trace, 194
 Transaction
 customize, 302
 DEMO_OO_METHOD, 59
 SAAB, 183
 SAT, 180
 SCI, 174
 SCU3, 239
 SD11, 268
 SE03, 49
 SE11, 205, 209, 212, 218, 232, 242, 245, 248, 252, 254, 257, 263, 291
 SE13, 238
 SE14, 264
 SE16, 203
 SE18, 293, 296, 300
 SE24, 111, 114
 SE37, 145

Transaction (Cont.)

- SE39*, 70
 - SE54*, 205, 229
 - SE80*, 66, 318, 326
 - SE83*, 21
 - SE84*, 55
 - SE93*, 24, 58, 306, 331
 - SFW1*, 285
 - SFW2*, 286
 - SFW5*, 286
 - SHD0*, 303
 - SICF*, 322
 - SLAD*, 154
 - SLIBN*, 23
 - SLIBP*, 23
 - SLIN*, 171
 - SM30*, 207, 214, 229, 234, 260
 - SM34*, 231
 - SM37*, 126
 - SM50*, 126
 - SM59*, 102
 - S_MEMORY_INSPECTOR*, 188
 - SPAU_ENH*, 288
 - ST05*, 190
 - SWEL*, 196
 - SWI1*, 197
 - WDYID*, 330
 - ZMM01_RAW*, 305
- Transaction code
- assign to Web Dynpro application*, 330
- Transaction variant, 302
- Transport behavior, 232
- Transport management system, 36
- Transport Organizer tools, 49, 52
- Tree controls, 23
- Two-step maintenance, 206

U

- UI element, 325
- Unicode check, 291
- Unique index, 223
- Unit test, 178
- Upgrade, 232, 288
- Upload, 63
- User-specific settings, 63

V

- Validity period, 259
- Value range, 215, 254
- Value table property, 215
- Variant, 56, 175
- Version management, 36
- View, 325
 - cluster*, 228

W

- Watchpoint, 146, 152
 - at Object Attribute*, 146
 - at Variable*, 146
 - local variable*, 147
- Web Dynpro ABAP, 313
 - application*, 314, 321
 - enhance components*, 325
 - framework*, 315
- Web Dynpro component, 329
- Where-used list, 57
- Window, 325
- Workflow, 194
 - graphic*, 199
 - log*, 198
- Work item, 197
 - container*, 199
- Worklist, 22, 28, 31
 - comments*, 32

X

- X Buffer, 80

Y

- Y Buffer, 80

Z

- Z Buffer, 80