

Horst Keller, Wolf Hagen Thümmel

Official ABAP™ Programming Guidelines


Galileo Press

Bonn • Boston

Contents

Foreword	13
Acknowledgments	15
1 Introduction	17
1.1 What Are Programming Guidelines?	17
1.2 Why Programming Guidelines?	18
1.3 Which Guidelines Are Involved Here?	18
1.4 Target Audience	19
1.5 How to Use This Book	20
2 General Basic Rules	23
2.1 Separation of Concerns	23
2.2 KISS Principle	32
2.3 Correctness and Quality	34
3 ABAP-Specific Basic Rules	41
3.1 ABAP Objects as a Programming Model	41
3.2 Program Type and Program Attributes	50
3.2.1 Program Type	51
3.2.2 Program Attributes	55
3.2.3 Original Language	60
3.3 Modern ABAP	62
3.4 Checks for Correctness	65
3.4.1 Syntax Check	65
3.4.2 Extended Program Check	69
3.4.3 Code Inspector	72
3.4.4 ABAP Test Cockpit	76

4 Structure and Style 79

- 4.1 Source Code Formatting 80
 - 4.1.1 Case Sensitivity 80
 - 4.1.2 Statements per Program Line 83
 - 4.1.3 Using the Pretty Printer 86
 - 4.1.4 Line Width 89
- 4.2 Naming 91
 - 4.2.1 Selecting the Language 92
 - 4.2.2 Descriptive Names 94
 - 4.2.3 Names of Repository Objects 101
 - 4.2.4 Program-Internal Names 106
- 4.3 Comments 115
 - 4.3.1 Selecting the Language 115
 - 4.3.2 Content 117
 - 4.3.3 Arrangement in the Source Code 120
- 4.4 Program and Procedure Structure 123
 - 4.4.1 Global Declarations of a Program 124
 - 4.4.2 Local Declarations 127
- 4.5 Source Code Organization 130
 - 4.5.1 Source Code Modularization 130
 - 4.5.2 Multiple Use of Include Programs 132
- 4.6 Alternative Notations 135
 - 4.6.1 Alternative Language Constructs in Statements 135
 - 4.6.2 Chained Statements 137
 - 4.6.3 Method Calls 141
 - 4.6.4 Assignments and Calculations 143
 - 4.6.5 Calculation Expressions 145
- 4.7 Complexity 146
 - 4.7.1 Expressions 147
 - 4.7.2 Nesting Depth 149
 - 4.7.3 Procedure Volume 150
 - 4.7.4 Class Size 151
 - 4.7.5 Dead Code 153

5 Architecture 155

- 5.1 Object-Oriented Programming 155
 - 5.1.1 Encapsulation 156

5.1.2	Modularization	157
5.1.3	Static Classes and Singletons	161
5.1.4	Inheritance	166
5.1.5	Class References and Interface References	167
5.1.6	Local Types for Global Classes	169
5.1.7	Instance Constructor	171
5.2	Error Handling	172
5.2.1	Reaction to Error Situations	172
5.2.2	Classical and Class-Based Exceptions	174
5.2.3	Exception Categories	178
5.2.4	Exception Texts	180
5.2.5	Using Exception Classes	183
5.2.6	Handling and Propagating Exceptions	185
5.2.7	Cleanup After Exceptions	186
5.2.8	Catchable Runtime Errors	188
5.2.9	Assertions	190
5.2.10	Messages	191
5.3	User Interfaces	195
5.3.1	Selecting the User Interface Technology	195
5.3.2	Encapsulating Classical User Interfaces	199
5.3.3	Lists	204
5.3.4	Accessibility	207
5.4	Data Storage	208
5.4.1	Persistent Data Storage	208
5.4.2	Database Accesses	210
5.4.3	Client Handling	211
5.4.4	Using the Shared Memory	213

6 Secure and Robust ABAP **217**

6.1	Data Types and Data Objects	217
6.1.1	Bound and Standalone Data Types	218
6.1.2	Declaration of Data Types and Constants	220
6.1.3	Declaration of Variables	224
6.1.4	Including Structures	226
6.1.5	Using Types	228
6.1.6	Referring to Data Types or Data Objects	230
6.1.7	Table Work Areas	232
6.1.8	Literals	233

6.1.9	Strings	236
6.1.10	Start Values	238
6.1.11	Data Objects for Truth Values	239
6.2	Assignments, Calculations, and Other Accesses to Data	241
6.2.1	Assignments Between Different Types	241
6.2.2	Avoiding Invalid Values	243
6.2.3	Using Conversion Rules	245
6.2.4	Specification of Numbers	247
6.2.5	Selecting the Numeric Type	249
6.2.6	Rounding Errors	253
6.2.7	Division by Zero	255
6.2.8	Casting	255
6.2.9	Runtime Errors When Accessing Data Objects	257
6.2.10	Anonymous Containers	259
6.2.11	Passing Global Data by Reference	260
6.3	System Fields	262
6.3.1	Access	262
6.3.2	Obsolete and Internal System Fields	264
6.3.3	Evaluation	265
6.3.4	Return Value	267
6.3.5	Using System Fields as Actual Parameters	268
6.3.6	Using System Fields on the User Interface	270
6.3.7	Using System Fields in Operand Positions	272
6.4	Internal Tables	273
6.4.1	Selecting the Table Category	274
6.4.2	Secondary Keys	276
6.4.3	Initial Memory Allocation	280
6.4.4	Sorted Filling	281
6.4.5	Aggregated Filling	283
6.4.6	Output Behavior	284
6.4.7	Loop Processing	286
6.5	Modularization Units	287
6.5.1	Function Modules and Subroutines	288
6.5.2	Type of the Formal Parameters of Procedures	289
6.5.3	Transfer Type of Formal Parameters	292
6.5.4	Passing Output Parameters by Reference	294
6.5.5	Typing of Formal Parameters	296
6.5.6	Internal and External Procedure Calls	298
6.5.7	Exiting Procedures	302

6.5.8	Dialog Modules and Event Blocks	304
6.5.9	Macros	306
6.6	<i>Dynamic Programming Techniques</i>	309
6.6.1	Using Dynamic Programming Techniques	309
6.6.2	Runtime Errors During Dynamic Processing	311
6.6.3	Using Dynamic Data Objects	313
6.6.4	Memory Consumption of Dynamic Memory Objects	315
6.6.5	Administration Costs of Dynamic Memory Objects	318
6.6.6	Accessing Data Objects Dynamically	321
6.6.7	Generic Programming	324
6.7	<i>Internationalization</i>	329
6.7.1	Storing System Texts	329
6.7.2	Translation-Friendly Message Texts	331
6.7.3	Text Environment	333
6.7.4	Character Set of Source Code	335
6.7.5	Splitting Texts	336
6.7.6	Codepages for Files	337

Appendices	339
-------------------------	------------

A	<i>Obsolete Language Constructs</i>	341
A.1	Procedures	342
A.2	Declarations	343
A.3	Object Generation	347
A.4	Calls and Exits	347
A.5	Program Flow Control	349
A.6	Assignments	350
A.7	Calculation Statements	352
A.8	Processing Character and Byte Strings	353
A.9	Internal Tables	355
A.10	Dynpro Flow Logic	357
A.11	Classical List Processing	358
A.12	Data Storage	360
A.13	Contexts	362
A.14	External Interfaces	363
B	<i>Automatic Check of Naming Conventions</i>	365
B.1	Naming Conventions in the Code Inspector	365
B.2	Type-Specific Prefix Components	366
B.3	Prefixes for Procedure-Local Declarations	367

B.4	Structured Programming	369
B.5	Object-Oriented Programming	370
B.6	Assessment of the Naming Conventions	371
C	Table of Rules	373
D	Recommended Reading	377
E	The Authors	379
	Index	381