



POLITECNICO DI MILANO
SCUOLA DI INGEGNERIA INDUSTRIALE E
DELL'INFORMAZIONE
M.Sc. IN COMPUTER SCIENCE AND ENGINEERING

GlassFish Server

Code Inspection

Authors:

Angelo GALLARELLO
Edoardo LONGO
Giacomo LOCCI

January 3, 2016
Version 1.0

Contents

Contents	1
1 Assigned Classes	6
2 Functional Role of Classes	6
Documentation sources	7
3 List of Issues	8
3.1 preInvokeNoTx	8
Method	8
Naming Convention	8
Indention	8
Braces	8
File Organizaion	9
Wrapping Lines	9
Comments	9
Java Source Files	9
Package and Import Statements	9
Class and Interface Declarations	9
Initialization and Declarations	10
Method Calls	10
Arrays	11
Object Comparison	11
Output Format	11
Computation, Comparisons and Assignments	11
Exceptions	12
Flow of control	12
Files	12
3.2 isHomeFinder	12
Method	12
Naming Convention	12
Indention	13
Braces	13
File Organizaion	13
Wrapping Lines	13
Comments	13
Java Source Files	14
Package and Import Statements	14

	Class and Interface Declarations	14
	Initialization and Declarations	15
	Method Calls	15
	Arrays	15
	Object Comparison	15
	Output Format	15
	Computation, Comparisons and Assignments	16
	Exceptions	16
	Flow of control	16
	Files	16
3.3	run	17
	Method	17
	Naming Convention	17
	Indention	17
	Braces	17
	File Organizaion	18
	Wrapping Lines	18
	Comments	18
	Java Source Files	18
	Package and Import Statements	18
	Class and Interface Declarations	18
	Initialization and Declarations	19
	Method Calls	19
	Arrays	20
	Object Comparison	20
	Output Format	20
	Computation, Comparisons and Assignments	20
	Exceptions	21
	Flow of control	21
	Files	21
3.4	passivateEJB	21
	Method	21
	Naming Convention	21
	Indention	22
	Braces	22
	File Organizaion	22
	Wrapping Lines	22
	Comments	22
	Java Source Files	23
	Package and Import Statements	23
	Class and Interface Declarations	23

	Initialization and Declarations	24
	Method Calls	24
	Arrays	24
	Object Comparison	24
	Output Format	24
	Computation, Comparisons and Assignments	25
	Exceptions	25
	Flow of control	25
	Files	25
3.5	internalGetEJBLocalObjectImpl	26
	Method	26
	Naming Convention	26
	Indention	26
	Braces	26
	File Organizaion	27
	Wrapping Lines	27
	Comments	27
	Java Source Files	27
	Package and Import Statements	27
	Class and Interface Declarations	28
	Initialization and Declarations	28
	Method Calls	29
	Arrays	29
	Object Comparison	29
	Output Format	29
	Computation, Comparisons and Assignments	29
	Exceptions	30
	Flow of control	30
	Files	30
3.6	getEJBObjectStub	30
	Method	30
	Naming Convention	31
	Indention	31
	Braces	31
	File Organizaion	31
	Wrapping Lines	31
	Comments	32
	Java Source Files	32
	Package and Import Statements	32
	Class and Interface Declarations	32
	Initialization and Declarations	33

	Method Calls	33
	Arrays	33
	Object Comparison	34
	Output Format	34
	Computation, Comparisons and Assignments	34
	Exceptions	34
	Flow of control	35
	Files	35
3.7	internalGetEJBObjectImpl	35
	Method	35
	Naming Convention	35
	Indentation	36
	Braces	36
	File Organizaion	36
	Wrapping Lines	36
	Comments	36
	Java Source Files	36
	Package and Import Statements	37
	Class and Interface Declarations	37
	Initialization and Declarations	38
	Method Calls	38
	Arrays	38
	Object Comparison	38
	Output Format	38
	Computation, Comparisons and Assignments	39
	Exceptions	39
	Flow of control	39
	Files	39
4	Other Problems	41
4.1	Indentation	41
4.2	Line length	45
4.3	Bracket position	51
4.4	Tab character	53
4.5	Missing brackets	69
4.6	Naming patterns	71
	Appendices	74

Abstract

This document is the Code Inspection.

1 Assigned Classes

- **Name:** EntityContainer.java
- **Location:** appserver/persistence/entitybean-container/src/main/java/org/glassfish/persistence/ejb/entitybean/container/EntityContainer.java

2 Functional Role of Classes

The *EntityContainer* class implements the *Container* interface for BMP (Bean Managed Persistence) and CMP (Container Managed Persistence) EntityBeans managing their instance and lifecycle.

The EntityContainer implements option B of the commit-time options described in the EJB2.0 spec section 10.5.

The Container caches a ready instance between transactions. In contrast to Option A, in this option the Container does not ensure that the instance has exclusive access to the state of the object in the persistent storage. Therefore, the Container must synchronize the instances state from the persistent storage at the beginning of the next transaction.

It also implements optimistic concurrency (i.e. multiple non-exclusive bean instances per primary key) when there are multiple concurrent transactions on a EntityBean.

The *EntityContainer* can perform different action for each *EJB* lifecycle stage (*Creation, Finding, Invocation*).

The EntityContainer manages collections of EJBs in 5 different states.

1. **POOLED** : the EB does not have an identity.
2. **READY** : the EB is ready for invocation and there are no transactions in progress.
3. **INVOKING** : the EB is processing an invocation.
4. **INCOMPLETE_TX** : the EB is ready for invocation and there are transactions in progress.
5. **DESTROYED** : the EB does not exist.

Documentation sources

- *EntityContainer.java* Javadoc
- *Enterprise JavaBeans™ Specification, Version 2.0* (http://www.dbs.ethz.ch/education/oho/SS_02/additional-info/ejb-2_0-fr2-spec.pdf)

3 List of Issues

3.1 preInvokeNoTx

Method

- **Name:** preInvokeNoTx(EjbInvocation inv)
- **Start Line:** 1599
- **Location:** appserver/persistence/entitybean-container/src/main/java/org/glassfish/persistence/ejb/entitybean/container/EntityContainer.java

Naming Convention

1. *Name meaningfulness:* No issues found.
2. *One char for loops:* In **line 1609** a single char name is used for a not throwaway var.
3. *Classes are nouns in camelCase:* No issues found.
4. *Interfaces are nouns in camelCase:* No issues found.
5. *Methods are verbs in camelCase:* No issues found.
6. *Class variables in camelCase (may be preceded by “_”):* No issues found.
7. *Constants all UPPERCASE (with words separated by “_”):* No issues found.

Indention

8. *Three or four spaces:* No issues found.
9. *No tabs:* No issues found.

Braces

10. *Consistent bracing style:* No issues found.
11. *Single statements sorrounded by curly braces:* No issues found.

File Organizaion

- 12. *Blank lines and optional comments to separate sections:* No issues found.
- 13. *Line length ≤ 80 characters (when practical):* No issues found.
- 14. *If it must exceed 80 characters, then they are ≤ 120 :* No issues found.

Wrapping Lines

- 15. *Line breaks after comma or operator:* No issues found.
- 16. *Higher-level breaks are used:* No issues found.
- 17. *New statements aligned with same level expressions:* No issues found.

Comments

- 18. *Comments adequately explain the purpose of the code:* No issues found.
- 19. *Commented code has reason to exist and contains date:* No issues found.

Java Source Files

- 20. *Each java source file contains only one public class or interface:* No issues found.
- 21. *The public class is the first class or interface in the file:* No issues found.
- 22. *External program interfaces are implemented consistently w.r.t. the javadoc:* No issues found.
- 23. *Javadoc is complete:* **Method has not a Javadoc comment.**

Package and Import Statements

- 24. *If any package statements are needed, they should be the first non-comment statements:* No issues found.

Class and Interface Declarations

- 25. *Class or interface follows the declaration template:*
 - (a) Class/interface documentation comment
 - (b) Class/interface statement
 - (c) Class/interface implementation comment, if necessary

- (d) Class (static) variables
 - i. first *public* class variables
 - ii. next *protected* class variables
 - iii. next package level (no access modifier)
 - iv. last *private* class variables
- (e) Instance variables
 - i. first *public* class variables
 - ii. next *protected* class variables
 - iii. next package level (no access modifier)
 - iv. last *private* class variables
- (f) Constructors
- (g) Methods

No issues found.

26. *Methods grouped by functionality*: No issues found.

27. *Code is free of duplicates, long methods, big classes, breaking encapsulation, also, coupling and cohesion are adequate*: No issues found.

Initialization and Declarations

28. *Correct type of variables and class members (right visibility)*: No issues found.

29. *Variables are declared in the proper scope*: No issues found.

30. *Constructors are called when a new object is desired*: No issues found.

31. *All object references are initialized before use*: No issues found.

32. *Variables are initialized where they are declared, unless dependent upon a computation*: No issues found.

33. *Declarations appear at the beginning of blocks*: No issues found.

Method Calls

34. *Parameters are presented in the correct order*: No issues found.

35. *Correct method is being called*: No issues found.

36. *Method returned variables are used properly*: No issues found.

Arrays

- 37. *Required array elements are accessed through the index:* No issues found.
- 38. *Indexes have been prevented from going out-of-bounds:* No issues found.
- 39. *Constructors are called when a new array item is desired:* No issues found.

Object Comparison

- 40. *All objects are compared with “equals”:* No issues found.

Output Format

- 41. *Displayed output is free of spelling and grammatical errors:* No issues found.
- 42. *Error messages are comprehensive and provide guidance as to how to correct the problem:* No issues found.
- 43. *Output is formatted correctly in terms of line stepping and spacing:* No issues found.

Computation, Comparisons and Assignments

- 44. *Implementation avoids “brutish programming”¹:* No issues found.
- 45. *Order of computation/evaluation, operator precedence and parenthesizing:* No issues found.
- 46. *Liberal use of parenthesis is used to avoid operator precedence problems.:* No issues found.
- 47. *All denominators of a division are prevented from being zero:* No issues found.
- 48. *Integer arithmetic, especially division, are used appropriately to avoid causing unexpected truncation/rounding:* No issues found.
- 49. *Comparison and Boolean operators are correct:* No issues found.
- 50. *Throw-catch expressions, and their error condition is actually legitimate:* No issues found.
- 51. *Code is free of any implicit type conversions:* No issues found.

¹See <http://users.csc.calpoly.edu/~jdalbey/SWE/CodeSmells/bonehead.html>.

Exceptions

- 52. *Relevant exceptions are caught:* No issues found.
- 53. *Appropriate action are taken for each catch block:* No issues found.

Flow of control

- 54. *In a switch statement, all cases are addressed by break or return:* No issues found.
- 55. *All switch statements have a default branch:* No issues found.
- 56. *All loops are correctly formed, with the appropriate initialization, increment and termination expressions:* No issues found.

Files

- 57. *All files are properly declared and opened:* No issues found.
- 58. *All files are closed properly, even in the case of an error:* No issues found.
- 59. *EOF conditions are detected and handled correctly:* No issues found.
- 60. *All file exceptions are caught and dealt with accordingly:* No issues found.

3.2 isHomeFinder

Method

- **Name:** isHomeFinder(Method method)
- **Start Line:** 1652
- **Location:** appserver/persistence/entitybean-container/src/main/java/org/glassfish/persistence/ejb/entitybean/container/EntityContainer.java

Naming Convention

- 1. *Name meaningfulness:* No issues found.
- 2. *One char for loops:* No issues found.
- 3. *Classes are nouns in camelCase:* No issues found.
- 4. *Interfaces are nouns in camelCase:* No issues found.

5. *Methods are verbs in camelCase*: No issues found.
6. *Class variables in camelCase (may be preceded by “_”)*: No issues found.
7. *Constants all UPPERCASE (with words separated by “_”)*: No issues found.

Indentation

8. *Three or four spaces*: No issues found.
9. *No tabs*: No issues found.

Braces

10. *Consistent bracing style*: No issues found.
11. *Single statements surrounded by curly braces*: No issues found.

File Organizaion

12. *Blank lines and optional comments to separate sections*: No issues found.
13. *Line length ≤ 80 characters (when practical)*: No issues found.
14. *If it must exceed 80 characters, then they are ≤ 120* : No issues found.

Wrapping Lines

15. *Line breaks after comma or operator*: No issues found.
16. *Higher-level breaks are used*: No issues found.
17. *New statements aligned with same level expressions*: No issues found.

Comments

18. *Comments adequately explain the purpose of the code*: No issues found.
19. *Commented code has reason to exist and contains date*: No issues found.

Java Source Files

- 20. *Each java source file contains only one public class or interface:* No issues found.
- 21. *The public class is the first class or interface in the file:* No issues found.
- 22. *External program interfaces are implemented consistently w.r.t. the javadoc:* No issues found.
- 23. *Javadoc is complete:* **Method has not a Javadoc comment.**

Package and Import Statements

- 24. *If any package statements are needed, they should be the first non-comment statements:* No issues found.

Class and Interface Declarations

- 25. *Class or interface follows the declaration template:*

- (a) Class/interface documentation comment
- (b) Class/interface statement
- (c) Class/interface implementation comment, if necessary
- (d) Class (static) variables
 - i. first *public* class variables
 - ii. next *protected* class variables
 - iii. next package level (no access modifier)
 - iv. last *private* class variables
- (e) Instance variables
 - i. first *public* class variables
 - ii. next *protected* class variables
 - iii. next package level (no access modifier)
 - iv. last *private* class variables
- (f) Constructors
- (g) Methods

No issues found.

- 26. *Methods grouped by functionality:* No issues found.
- 27. *Code is free of duplicates, long methods, big classes, breaking encapsulation, also, coupling and cohesion are adequate:* No issues found.

Initialization and Declarations

- 28. *Correct type of variables and class members (right visibility):* No issues found.
- 29. *Variables are declared in the proper scope:* No issues found.
- 30. *Constructors are called when a new object is desired:* No issues found.
- 31. *All object references are initialized before use:* No issues found.
- 32. *Variables are initialized where they are declared, unless dependent upon a computation:* No issues found.
- 33. *Declarations appear at the beginnig of blocks:* No issues found.

Method Calls

- 34. *Parameters are presented in the correct order:* No issues found.
- 35. *Correct method is being called:* No issues found.
- 36. *Method returned variables are used properly:* No issues found.

Arrays

- 37. *Required array elements are accessed through the index:* No issues found.
- 38. *Indexes have been prevented from going out-of-bounds:* No issues found.
- 39. *Constructors are called when a new array item is desired:* No issues found.

Object Comparison

- 40. *All objects are compared with “equals”:* **In line 1657 it’s not used equals to compare two objects.**

Output Format

- 41. *Displayed output is free of spelling and grammatical errors:* No issues found.
- 42. *Error messages are comprehensive and provide guidance as to how to correct the problem:* No issues found.
- 43. *Output is formatted correctly in terms of line stepping and spacing:* No issues found.

Computation, Comparisons and Assignments

- 44. *Implementation avoids “brutish programming”²*: No issues found.
- 45. *Order of computation/evaluation, operator precedence and parenthesizing*: No issues found.
- 46. *Liberal use of parenthesis is used to avoid operator precedence problems.*: No issues found.
- 47. *All denominators of a division are prevented from being zero*: No issues found.
- 48. *Integer arithmetic, especially division, are used appropriately to avoid causing unexpected truncation/rounding*: No issues found.
- 49. *Comparison and Boolean operators are correct*: No issues found.
- 50. *Throw-catch expressions, and their error condition is actually legitimate*: No issues found.
- 51. *Code is free of any implicit type conversions*: No issues found.

Exceptions

- 52. *Relevant exceptions are caught*: No issues found.
- 53. *Appropriate action are taken for each catch block*: No issues found.

Flow of control

- 54. *In a switch statement, all cases are addressed by break or return*: No issues found.
- 55. *All switch statements have a default branch*: No issues found.
- 56. *All loops are correctly formed, with the appropriate initialization, increment and termination expressions*: No issues found.

Files

- 57. *All files are properly declared and opened*: No issues found.
- 58. *All files are closed properly, even in the case of an error*: No issues found.
- 59. *EOF conditions are detected and handled correctly*: No issues found.

²See <http://users.csc.calpoly.edu/~jdalbey/SWE/CodeSmells/bonehead.html>.

60. *All file exceptions are caught and dealt with accordingly:* No issues found.

3.3 run

Method

- **Name:** run()
- **Start Line:** 1699
- **Location:** appserver/persistence/entitybean-container/src/main/java/org/glassfish/persistence/ejb/entitybean/container/EntityContainer.java

Naming Convention

1. *Name meaningfulness:* No issues found.
2. *One char for loops:* No issues found.
3. *Classes are nouns in camelCase:* No issues found.
4. *Interfaces are nouns in camelCase:* No issues found.
5. *Methods are verbs in camelCase:* No issues found.
6. *Class variables in camelCase (may be preceded by “_”):* No issues found.
7. *Constants all UPPERCASE (with words separated by “_”):* No issues found.

Indention

8. *Three or four spaces:* **The method is poorly formatted and a bit messy.**
9. *No tabs:* No issues found.

Braces

10. *Consistent bracing style:* No issues found.
11. *Single statements sorrounded by curly braces:* No issues found.

File Organizaion

- 12. *Blank lines and optional comments to separate sections:* No issues found.
- 13. *Line length ≤ 80 characters (when practical):* No issues found.
- 14. *If it must exceed 80 characters, then they are ≤ 120 :* No issues found.

Wrapping Lines

- 15. *Line breaks after comma or operator:* No issues found.
- 16. *Higher-level breaks are used:* No issues found.
- 17. *New statements aligned with same level expressions:* No issues found.

Comments

- 18. *Comments adequately explain the purpose of the code:* No issues found.
- 19. *Commented code has reason to exist and contains date:* No issues found.

Java Source Files

- 20. *Each java source file contains only one public class or interface:* No issues found.
- 21. *The public class is the first class or interface in the file:* No issues found.
- 22. *External program interfaces are implemented consistently w.r.t. the javadoc:* No issues found.
- 23. *Javadoc is complete:* **Javadoc doesn't exist on this method.**

Package and Import Statements

- 24. *If any package statements are needed, they should be the first non-comment statements:* No issues found.

Class and Interface Declarations

- 25. *Class or interface follows the declaration template:*
 - (a) Class/interface documentation comment
 - (b) Class/interface statement
 - (c) Class/interface implementation comment, if necessary

- (d) Class (static) variables
 - i. first *public* class variables
 - ii. next *protected* class variables
 - iii. next package level (no access modifier)
 - iv. last *private* class variables
- (e) Instance variables
 - i. first *public* class variables
 - ii. next *protected* class variables
 - iii. next package level (no access modifier)
 - iv. last *private* class variables
- (f) Constructors
- (g) Methods

No issues found.

26. *Methods grouped by functionality*: No issues found.

27. *Code is free of duplicates, long methods, big classes, breaking encapsulation, also, coupling and cohesion are adequate*: No issues found.

Initialization and Declarations

28. *Correct type of variables and class members (right visibility)*: No issues found.

29. *Variables are declared in the proper scope*: No issues found.

30. *Constructors are called when a new object is desired*: No issues found.

31. *All object references are initialized before use*: No issues found.

32. *Variables are initialized where they are declared, unless dependent upon a computation*: No issues found.

33. *Declarations appear at the beginning of blocks*: No issues found.

Method Calls

34. *Parameters are presented in the correct order*: No issues found.

35. *Correct method is being called*: No issues found.

36. *Method returned variables are used properly*: No issues found.

Arrays

- 37. *Required array elements are accessed through the index:* No issues found.
- 38. *Indexes have been prevented from going out-of-bounds:* No issues found.
- 39. *Constructors are called when a new array item is desired:* No issues found.

Object Comparison

- 40. *All objects are compared with “equals”:* No issues found.

Output Format

- 41. *Displayed output is free of spelling and grammatical errors:* No issues found.
- 42. *Error messages are comprehensive and provide guidance as to how to correct the problem:* No issues found.
- 43. *Output is formatted correctly in terms of line stepping and spacing:* No issues found.

Computation, Comparisons and Assignments

- 44. *Implementation avoids “brutish programming”³:* No issues found.
- 45. *Order of computation/evaluation, operator precedence and parenthesizing:* No issues found.
- 46. *Liberal use of parenthesis is used to avoid operator precedence problems.:* No issues found.
- 47. *All denominators of a division are prevented from being zero:* No issues found.
- 48. *Integer arithmetic, especially division, are used appropriately to avoid causing unexpected truncation/rounding:* No issues found.
- 49. *Comparison and Boolean operators are correct:* No issues found.
- 50. *Throw-catch expressions, and their error condition is actually legitimate:* No issues found.
- 51. *Code is free of any implicit type conversions:* No issues found.

³See <http://users.csc.calpoly.edu/~jdalbey/SWE/CodeSmells/bonehead.html>.

Exceptions

- 52. *Relevant exceptions are caught:* No issues found.
- 53. *Appropriate action are taken for each catch block:* No issues found.

Flow of control

- 54. *In a switch statement, all cases are addressed by break or return:* No issues found.
- 55. *All switch statements have a default branch:* No issues found.
- 56. *All loops are correctly formed, with the appropriate initialization, increment and termination expressions:* No issues found.

Files

- 57. *All files are properly declared and opened:* No issues found.
- 58. *All files are closed properly, even in the case of an error:* No issues found.
- 59. *EOF conditions are detected and handled correctly:* No issues found.
- 60. *All file exceptions are caught and dealt with accordingly:* No issues found.

3.4 passivateEJB

Method

- **Name:** passivateEJB(ComponentContext ctx)
- **Start Line:** 1762
- **Location:** appserver/persistence/entitybean-container/src/main/java/org/glassfish/persistence/ejb/entitybean/container/EntityContainer.java

Naming Convention

- 1. *Name meaningfulness:* No issues found.
- 2. *One char for loops:* No issues found.
- 3. *Classes are nouns in camelCase:* No issues found.
- 4. *Interfaces are nouns in camelCase:* No issues found.

5. *Methods are verbs in camelCase*: No issues found.
6. *Class variables in camelCase (may be preceded by “_”)*: No issues found.
7. *Constants all UPPERCASE (with words separated by “_”)*: No issues found.

Indentation

8. *Three or four spaces*: No issues found.
9. *No tabs*: No issues found.

Braces

10. *Consistent bracing style*: No issues found.
11. *Single statements surrounded by curly braces*: No issues found.

File Organizaion

12. *Blank lines and optional comments to separate sections*: No issues found.
13. *Line length ≤ 80 characters (when practical)*: No issues found.
14. *If it must exceed 80 characters, then they are ≤ 120* : No issues found.

Wrapping Lines

15. *Line breaks after comma or operator*: No issues found.
16. *Higher-level breaks are used*: No issues found.
17. *New statements aligned with same level expressions*: No issues found.

Comments

18. *Comments adequately explain the purpose of the code*: **Not totally explicative and quite confusing.**
19. *Commented code has reason to exist and contains date*: No issues found.

Java Source Files

- 20. *Each java source file contains only one public class or interface:* No issues found.
- 21. *The public class is the first class or interface in the file:* No issues found.
- 22. *External program interfaces are implemented consistently w.r.t. the javadoc:* No issues found.
- 23. *Javadoc is complete:* **Javadoc doesn't exist on this method.**

Package and Import Statements

- 24. *If any package statements are needed, they should be the first non-comment statements:* No issues found.

Class and Interface Declarations

- 25. *Class or interface follows the declaration template:*

- (a) Class/interface documentation comment
- (b) Class/interface statement
- (c) Class/interface implementation comment, if necessary
- (d) Class (static) variables
 - i. first *public* class variables
 - ii. next *protected* class variables
 - iii. next package level (no access modifier)
 - iv. last *private* class variables
- (e) Instance variables
 - i. first *public* class variables
 - ii. next *protected* class variables
 - iii. next package level (no access modifier)
 - iv. last *private* class variables
- (f) Constructors
- (g) Methods

No issues found.

- 26. *Methods grouped by functionality:* No issues found.
- 27. *Code is free of duplicates, long methods, big classes, breaking encapsulation, also, coupling and cohesion are adequate:* No issues found.

Initialization and Declarations

- 28. *Correct type of variables and class members (right visibility):* No issues found.
- 29. *Variables are declared in the proper scope:* No issues found.
- 30. *Constructors are called when a new object is desired:* No issues found.
- 31. *All object references are initialized before use:* No issues found.
- 32. *Variables are initialized where they are declared, unless dependent upon a computation:* No issues found.
- 33. *Declarations appear at the beginning of blocks:* No issues found.

Method Calls

- 34. *Parameters are presented in the correct order:* No issues found.
- 35. *Correct method is being called:* No issues found.
- 36. *Method returned variables are used properly:* No issues found.

Arrays

- 37. *Required array elements are accessed through the index:* No issues found.
- 38. *Indexes have been prevented from going out-of-bounds:* No issues found.
- 39. *Constructors are called when a new array item is desired:* No issues found.

Object Comparison

- 40. *All objects are compared with “equals”:* No issues found.

Output Format

- 41. *Displayed output is free of spelling and grammatical errors:* No issues found.
- 42. *Error messages are comprehensive and provide guidance as to how to correct the problem:* No issues found.
- 43. *Output is formatted correctly in terms of line stepping and spacing:* No issues found.

Computation, Comparisons and Assignments

- 44. *Implementation avoids “brutish programming”*⁴: No issues found.
- 45. *Order of computation/evaluation, operator precedence and parenthesizing*: No issues found.
- 46. *Liberal use of parenthesis is used to avoid operator precedence problems.*: No issues found.
- 47. *All denominators of a division are prevented from being zero*: No issues found.
- 48. *Integer arithmetic, especially division, are used appropriately to avoid causing unexpected truncation/rounding*: No issues found.
- 49. *Comparison and Boolean operators are correct*: No issues found.
- 50. *Throw-catch expressions, and their error condition is actually legitimate*: No issues found.
- 51. *Code is free of any implicit type conversions*: No issues found.

Exceptions

- 52. *Relevant exceptions are caught*: No issues found.
- 53. *Appropriate action are taken for each catch block*: No issues found.

Flow of control

- 54. *In a switch statement, all cases are addressed by break or return*: No issues found.
- 55. *All switch statements have a default branch*: No issues found.
- 56. *All loops are correctly formed, with the appropriate initialization, increment and termination expressions*: No issues found.

Files

- 57. *All files are properly declared and opened*: No issues found.
- 58. *All files are closed properly, even in the case of an error*: No issues found.
- 59. *EOF conditions are detected and handled correctly*: No issues found.

⁴See <http://users.csc.calpoly.edu/~jdalbey/SWE/CodeSmells/bonehead.html>.

60. *All file exceptions are caught and dealt with accordingly:* No issues found.

3.5 internalGetEJBLocalObjectImpl

Method

- **Name:** internalGetEJBLocalObjectImpl(Object primaryKey , boolean incrementRefCount , boolean cacheEJBO)
- **Start Line:** 1849
- **Location:** appserver/persistence/entitybean-container/src/main/java/org/glassfish/persistence/ejb/entitybean/container/EntityContainer.java

Naming Convention

1. *Name meaningfulness:* **Method name it's not auto-explicative and doesn't follow the standard naming convention.**
'getInternalEJBLocalObjectImplementation' would be better.
2. *One char for loops:* No issues found.
3. *Classes are nouns in camelCase:* No issues found.
4. *Interfaces are nouns in camelCase:* No issues found.
5. *Methods are verbs in camelCase:* No issues found.
6. *Class variables in camelCase (may be preceded by "_"):* No issues found.
7. *Constants all UPPERCASE (with words separated by "_"):* No issues found.

Indention

8. *Three or four spaces:* No issues found.
9. *No tabs:* No issues found.

Braces

10. *Consistent bracing style:* No issues found.
11. *Single statements sorrounded by curly braces:* No issues found.

File Organizaion

- 12. *Blank lines and optional comments to separate sections:* No issues found.
- 13. *Line length ≤ 80 characters (when practical):* No issues found.
- 14. *If it must exceed 80 characters, then they are ≤ 120 :* No issues found.

Wrapping Lines

- 15. *Line breaks after comma or operator:* No issues found.
- 16. *Higher-level breaks are used:* No issues found.
- 17. *New statements aligned with same level expressions:* No issues found.

Comments

- 18. *Comments adequately explain the purpose of the code:* **Not totally explicative and quite confusing.**
- 19. *Commented code has reason to exist and contains date:* No issues found.

Java Source Files

- 20. *Each java source file contains only one public class or interface:* No issues found.
- 21. *The public class is the first class or interface in the file:* No issues found.
- 22. *External program interfaces are implemented consistently w.r.t. the javadoc:* No issues found.
- 23. *Javadoc is complete:* **Javadoc doesn't exist on this method.**

Package and Import Statements

- 24. *If any package statements are needed, they should be the first non-comment statements:* No issues found.

Class and Interface Declarations

25. *Class or interface follows the declaration template:*

- (a) Class/interface documentation comment
- (b) Class/interface statement
- (c) Class/interface implementation comment, if necessary
- (d) Class (static) variables
 - i. first *public* class variables
 - ii. next *protected* class variables
 - iii. next package level (no access modifier)
 - iv. last *private* class variables
- (e) Instance variables
 - i. first *public* class variables
 - ii. next *protected* class variables
 - iii. next package level (no access modifier)
 - iv. last *private* class variables
- (f) Constructors
- (g) Methods

No issues found.

26. *Methods grouped by functionality:* No issues found.

27. *Code is free of duplicates, long methods, big classes, breaking encapsulation, also, coupling and cohesion are adequate:* No issues found.

Initialization and Declarations

28. *Correct type of variables and class members (right visibility):* No issues found.

29. *Variables are declared in the proper scope:* No issues found.

30. *Constructors are called when a new object is desired:* No issues found.

31. *All object references are initialized before use:* No issues found.

32. *Variables are initialized where they are declared, unless dependent upon a computation:* No issues found.

33. *Declarations appear at the beginning of blocks:* No issues found.

Method Calls

- 34. *Parameters are presented in the correct order:* No issues found.
- 35. *Correct method is being called:* No issues found.
- 36. *Method returned variables are used properly:* No issues found.

Arrays

- 37. *Required array elements are accessed through the index:* No issues found.
- 38. *Indexes have been prevented from going out-of-bounds:* No issues found.
- 39. *Constructors are called when a new array item is desired:* No issues found.

Object Comparison

- 40. *All objects are compared with “equals”:* No issues found.

Output Format

- 41. *Displayed output is free of spelling and grammatical errors:* No issues found.
- 42. *Error messages are comprehensive and provide guidance as to how to correct the problem:* No issues found.
- 43. *Output is formatted correctly in terms of line stepping and spacing:* No issues found.

Computation, Comparisons and Assignments

- 44. *Implementation avoids “brutish programming”⁵:* No issues found.
- 45. *Order of computation/evaluation, operator precedence and parenthesizing:* No issues found.
- 46. *Liberal use of parenthesis is used to avoid operator precedence problems.:* No issues found.
- 47. *All denominators of a division are prevented from being zero:* No issues found.

⁵See <http://users.csc.calpoly.edu/~jdalbey/SWE/CodeSmells/bonehead.html>.

- 48. *Integer arithmetic, especially division, are used appropriately to avoid causing unexpected truncation/rounding:* No issues found.
- 49. *Comparison and Boolean operators are correct:* No issues found.
- 50. *Throw-catch expressions, and their error condition is actually legitimate:* No issues found.
- 51. *Code is free of any implicit type conversions:* No issues found.

Exceptions

- 52. *Relevant exceptions are caught:* No issues found.
- 53. *Appropriate action are taken for each catch block:* No issues found.

Flow of control

- 54. *In a switch statement, all cases are addressed by break or return:* No issues found.
- 55. *All switch statements have a default branch:* No issues found.
- 56. *All loops are correctly formed, with the appropriate initialization, increment and termination expressions:* No issues found.

Files

- 57. *All files are properly declared and opened:* No issues found.
- 58. *All files are closed properly, even in the case of an error:* No issues found.
- 59. *EOF conditions are detected and handled correctly:* No issues found.
- 60. *All file exceptions are caught and dealt with accordingly:* No issues found.

3.6 getEJBObjectStub

Method

- **Name:** getEJBObjectStub(Object primaryKey , byte [] streamKey)
- **Start Line:** 1877
- **Location:** appserver/persistence/entitybean-container/src/main/java/org/glassfish/persistence/ejb/entitybean/container/EntityContainer.java

Naming Convention

1. *Name meaningfulness*: No issues found.
2. *One char for loops*: No issues found.
3. *Classes are nouns in camelCase*: No issues found.
4. *Interfaces are nouns in camelCase*: No issues found.
5. *Methods are verbs in camelCase*: No issues found.
6. *Class variables in camelCase (may be preceded by “_”)*: No issues found.
7. *Constants all UPPERCASE (with words separated by “_”)*: No issues found.

Indentation

8. *Three or four spaces*: No issues found.
9. *No tabs*: **Line 1877 contains a tab character (even if it is just at the end of the line).**

Braces

10. *Consistent bracing style*: No issues found.
11. *Single statements surrounded by curly braces*: No issues found.

File Organizaion

12. *Blank lines and optional comments to separate sections*: **Line 1894 is blank for no reason.**
13. *Line length ≤ 80 characters (when practical)*: No issues found.
14. *If it must exceed 80 characters, then they are ≤ 120* : No issues found.

Wrapping Lines

15. *Line breaks after comma or operator*: No issues found.
16. *Higher-level breaks are used*: No issues found.
17. *New statements aligned with same level expressions*: No issues found.

Comments

- 18. *Comments adequately explain the purpose of the code:* No issues found.
- 19. *Commented code has reason to exist and contains date:* No issues found.

Java Source Files

- 20. *Each java source file contains only one public class or interface:* No issues found.
- 21. *The public class is the first class or interface in the file:* No issues found.
- 22. *External program interfaces are implemented consistently w.r.t. the javadoc:* No issues found.
- 23. *Javadoc is complete:* **Method does not have a Javadoc comment.**

Package and Import Statements

- 24. *If any package statements are needed, they should be the first non-comment statements:* No issues found.

Class and Interface Declarations

- 25. *Class or interface follows the declaration template:*
 - (a) Class/interface documentation comment
 - (b) Class/interface statement
 - (c) Class/interface implementation comment, if necessary
 - (d) Class (static) variables
 - i. first *public* class variables
 - ii. next *protected* class variables
 - iii. next package level (no access modifier)
 - iv. last *private* class variables
 - (e) Instance variables
 - i. first *public* class variables
 - ii. next *protected* class variables
 - iii. next package level (no access modifier)
 - iv. last *private* class variables
 - (f) Constructors

(g) Methods

No issues found.

26. *Methods grouped by functionality*: No issues found.

27. *Code is free of duplicates, long methods, big classes, breaking encapsulation, also, coupling and cohesion are adequate*: No issues found.

Initialization and Declarations

28. *Correct type of variables and class members (right visibility)*: No issues found.

29. *Variables are declared in the proper scope*: No issues found.

30. *Constructors are called when a new object is desired*: No issues found.

31. *All object references are initialized before use*: No issues found.

32. *Variables are initialized where they are declared, unless dependent upon a computation*: No issues found.

33. *Declarations appear at the beginning of blocks*: **Declaration at line 1892 should be at the beginning of ‘try’ block.**

Method Calls

34. *Parameters are presented in the correct order*: **In each call of the method, the second parameter ‘byte[] streamKey’ is always passed as ‘null’.**

35. *Correct method is being called*: No issues found.

36. *Method returned variables are used properly*: No issues found.

Arrays

37. *Required array elements are accessed through the index*: No issues found.

38. *Indexes have been prevented from going out-of-bounds*: No issues found.

39. *Constructors are called when a new array item is desired*: No issues found.

Object Comparison

- 40. *All objects are compared with “equals”*: No issues found.

Output Format

- 41. *Displayed output is free of spelling and grammatical errors*: No issues found.
- 42. *Error messages are comprehensive and provide guidance as to how to correct the problem*: No issues found.
- 43. *Output is formatted correctly in terms of line stepping and spacing*: No issues found.

Computation, Comparisons and Assignments

- 44. *Implementation avoids “brutish programming”⁶*: No issues found.
- 45. *Order of computation/evaluation, operator precedence and parenthesizing*: No issues found.
- 46. *Liberal use of parenthesis is used to avoid operator precedence problems.*: No issues found.
- 47. *All denominators of a division are prevented from being zero*: No issues found.
- 48. *Integer arithmetic, especially division, are used appropriately to avoid causing unexpected truncation/rounding*: No issues found.
- 49. *Comparison and Boolean operators are correct*: No issues found.
- 50. *Throw-catch expressions, and their error condition is actually legitimate*: No issues found.
- 51. *Code is free of any implicit type conversions*: No issues found.

Exceptions

- 52. *Relevant exceptions are caught*: No issues found.
- 53. *Appropriate action are taken for each catch block*: No issues found.

⁶See <http://users.csc.calpoly.edu/~jdalbey/SWE/CodeSmells/bonehead.html>.

Flow of control

- 54. *In a switch statement, all cases are addressed by break or return:* No issues found.
- 55. *All switch statements have a default branch:* No issues found.
- 56. *All loops are correctly formed, with the appropriate initialization, increment and termination expressions:* No issues found.

Files

- 57. *All files are properly declared and opened:* No issues found.
- 58. *All files are closed properly, even in the case of an error:* No issues found.
- 59. *EOF conditions are detected and handled correctly:* No issues found.
- 60. *All file exceptions are caught and dealt with accordingly:* No issues found.

3.7 internalGetEJBObjectImpl

Method

- **Name:** internalGetEJBObjectImpl(Object primaryKey , byte [] streamKey , boolean incrementRefCount , boolean cacheEJBO)
- **Start Line:** 1920
- **Location:** appserver/persistence/entitybean-container/src/main/java/org/glassfish/persistence/ejb/entitybean/container/EntityContainer.java

Naming Convention

1. *Name meaningfulness:* **Method name it's not auto-explicative and doesn't follow the standard naming convention. 'getInternalEJBObjectImplementation' would be better.**
2. *One char for loops:* No issues found.
3. *Classes are nouns in camelCase:* No issues found.
4. *Interfaces are nouns in camelCase:* No issues found.
5. *Methods are verbs in camelCase:* No issues found.

6. *Class variables in camelCase (may be preceded by “_”)*: No issues found.
7. *Constants all UPPERCASE (with words separated by “_”)*: No issues found.

Indentation

8. *Three or four spaces*: No issues found.
9. *No tabs*: No issues found.

Braces

10. *Consistent bracing style*: **Line 1974 should be on the previous line.**
11. *Single statements surrounded by curly braces*: No issues found.

File Organizaion

12. *Blank lines and optional comments to separate sections*: **Lines 1926, 1948, 1958 are blank for no reason.**
13. *Line length ≤ 80 characters (when practical)*: No issues found.
14. *If it must exceed 80 characters, then they are ≤ 120* : No issues found.

Wrapping Lines

15. *Line breaks after comma or operator*: No issues found.
16. *Higher-level breaks are used*: No issues found.
17. *New statements aligned with same level expressions*: No issues found.

Comments

18. *Comments adequately explain the purpose of the code*: No issues found.
19. *Commented code has reason to exist and contains date*: No issues found.

Java Source Files

20. *Each java source file contains only one public class or interface*: No issues found.
21. *The public class is the first class or interface in the file*: No issues found.

22. *External program interfaces are implemented consistently w.r.t. the javadoc:* No issues found.

23. *Javadoc is complete:* **Method does not have a Javadoc comment.**

Package and Import Statements

24. *If any package statements are needed, they should be the first non-comment statements:* No issues found.

Class and Interface Declarations

25. *Class or interface follows the declaration template:*

- (a) Class/interface documentation comment
- (b) Class/interface statement
- (c) Class/interface implementation comment, if necessary
- (d) Class (static) variables
 - i. first *public* class variables
 - ii. next *protected* class variables
 - iii. next package level (no access modifier)
 - iv. last *private* class variables
- (e) Instance variables
 - i. first *public* class variables
 - ii. next *protected* class variables
 - iii. next package level (no access modifier)
 - iv. last *private* class variables
- (f) Constructors
- (g) Methods

No issues found.

26. *Methods grouped by functionality:* No issues found.

27. *Code is free of duplicates, long methods, big classes, breaking encapsulation, also, coupling and cohesion are adequate:* No issues found.

Initialization and Declarations

- 28. *Correct type of variables and class members (right visibility):* No issues found.
- 29. *Variables are declared in the proper scope:* No issues found.
- 30. *Constructors are called when a new object is desired:* No issues found.
- 31. *All object references are initialized before use:* No issues found.
- 32. *Variables are initialized where they are declared, unless dependent upon a computation:* No issues found.
- 33. *Declarations appear at the beginnig of blocks:* **Declaration at line 1952 should be at the beginning of ‘try’ block.**

Method Calls

- 34. *Parameters are presented in the correct order:* No issues found.
- 35. *Correct method is being called:* No issues found.
- 36. *Method returned variables are used properly:* No issues found.

Arrays

- 37. *Required array elements are accessed through the index:* No issues found.
- 38. *Indexes have been prevented from going out-of-bounds:* No issues found.
- 39. *Constructors are called when a new array item is desired:* No issues found.

Object Comparison

- 40. *All objects are compared with “equals”:* **Line 1965 uses ‘!=’.**

Output Format

- 41. *Displayed output is free of spelling and grammatical errors:* No issues found.
- 42. *Error messages are comprehensive and provide guidance as to how to correct the problem:* No issues found.
- 43. *Output is formatted correctly in terms of line stepping and spacing:* No issues found.

Computation, Comparisons and Assignments

- 44. *Implementation avoids “brutish programming”*⁷: No issues found.
- 45. *Order of computation/evaluation, operator precedence and parenthesizing*: No issues found.
- 46. *Liberal use of parenthesis is used to avoid operator precedence problems.*: No issues found.
- 47. *All denominators of a division are prevented from being zero*: No issues found.
- 48. *Integer arithmetic, especially division, are used appropriately to avoid causing unexpected truncation/rounding*: No issues found.
- 49. *Comparison and Boolean operators are correct*: No issues found.
- 50. *Throw-catch expressions, and their error condition is actually legitimate*: No issues found.
- 51. *Code is free of any implicit type conversions*: No issues found.

Exceptions

- 52. *Relevant exceptions are caught*: No issues found.
- 53. *Appropriate action are taken for each catch block*: No issues found.

Flow of control

- 54. *In a switch statement, all cases are addressed by break or return*: No issues found.
- 55. *All switch statements have a default branch*: No issues found.
- 56. *All loops are correctly formed, with the appropriate initialization, increment and termination expressions*: No issues found.

Files

- 57. *All files are properly declared and opened*: No issues found.
- 58. *All files are closed properly, even in the case of an error*: No issues found.
- 59. *EOF conditions are detected and handled correctly*: No issues found.

⁷See <http://users.csc.calpoly.edu/~jdalbey/SWE/CodeSmells/bonehead.html>.

60. *All file exceptions are caught and dealt with accordingly.* No issues found.

4 Other Problems

This section contains formatting problems found in all the class.

4.1 Indentation

- – line:“773”
 - error:“*if* child have incorrect indentation level 16, expected level should be 20.”
- – line:“773”
 - error:“—— have incorrect indentation level 16, expected level should be 20.”
- – line:“853”
 - error:“*EjbInvocation* have incorrect indentation level 5, expected level should be 8.”
- – line:“881”
 - error:“*!* have incorrect indentation level 8, expected level should be 12.”
- – line:“881”
 - error:“*if* child have incorrect indentation level 8, expected level should be 12.”
- – line:“1035”
 - error:“*method def lparen* have incorrect indentation level 8, expected level should be 4.”
- – line:“1050”
 - error:“(have incorrect indentation level 2, expected level should be 9.”
- – line:“1053”
 - error:“*activeTxCache* have incorrect indentation level 3, expected level should be 6.”
- – line:“1688”
 - error:“*method call* child have incorrect indentation level 11, expected level should be 12.”

- – line:“1688”
 - error:“*try* child have incorrect indentation level 11, expected level should be 12.”
- – line:“1713”
 - error:“*method def modifier* have incorrect indentation level 24, expected level should be one of the following: 28, 32.”
- – line:“1714”
 - error:“*method call* child have incorrect indentation level 28, expected level should be one of the following: 32, 36.”
- – line:“1714”
 - error:“*method def* child have incorrect indentation level 28, expected level should be one of the following: 32, 36.”
- – line:“1715”
 - error:“*method def* child have incorrect indentation level 28, expected level should be one of the following: 32, 36.”
- – line:“1716”
 - error:“*method def rcurly* have incorrect indentation level 24, expected level should be one of the following: 28, 32.”
- – line:“1717”
 - error:“*object def rcurly* have incorrect indentation level 20, expected level should be one of the following: 24, 28.”
- – line:“1727”
 - error:“*if* child have incorrect indentation level 26, expected level should be 32.”
- – line:“1750”
 - error:“*method def modifier* have incorrect indentation level 24, expected level should be one of the following: 28, 32.”
- – line:“1751”
 - error:“*method call* child have incorrect indentation level 28, expected level should be one of the following: 32, 36.”
- – line:“1751”

- error:“*method def* child have incorrect indentation level 28, expected level should be one of the following: 32, 36.”
- – line:“1752”
 - error:“*method def* child have incorrect indentation level 28, expected level should be one of the following: 32, 36.”
- – line:“1753”
 - error:“*method def rcurlly* have incorrect indentation level 24, expected level should be one of the following: 28, 32.”
- – line:“1754”
 - error:“*object def rcurlly* have incorrect indentation level 20, expected level should be one of the following: 24, 28.”
- – line:“1837”
 - error:“*method def lparen* have incorrect indentation level 8, expected level should be 4.”
- – line:“1843”
 - error:“*method def lparen* have incorrect indentation level 8, expected level should be 4.”
- – line:“1850”
 - error:“*method def lparen* have incorrect indentation level 8, expected level should be 4.”
- – line:“2144”
 - error:“*ejbContainerUtilImpl* have incorrect indentation level 2, expected level should be 9.”
- – line:“2147”
 - error:“*if* child have incorrect indentation level 8, expected level should be 12.”
- – line:“2461”
 - error:“*method call lparen* have incorrect indentation level 28, expected level should be 24.”
- – line:“2909”

- error:“*method def modifier* have incorrect indentation level 24, expected level should be one of the following: 28, 32.”
- – line:“2910”
 - error:“*method call* child have incorrect indentation level 28, expected level should be one of the following: 32, 36.”
- – line:“2910”
 - error:“*method def* child have incorrect indentation level 28, expected level should be one of the following: 32, 36.”
- – line:“2911”
 - error:“*method def* child have incorrect indentation level 28, expected level should be one of the following: 32, 36.”
- – line:“2912”
 - error:“*method def rcurly* have incorrect indentation level 24, expected level should be one of the following: 28, 32.”
- – line:“2913”
 - error:“*object def rcurly* have incorrect indentation level 20, expected level should be one of the following: 24, 28.”
- – line:“2946”
 - error:“*method def modifier* have incorrect indentation level 24, expected level should be one of the following: 28, 32.”
- – line:“2947”
 - error:“*method call* child have incorrect indentation level 28, expected level should be one of the following: 32, 36.”
- – line:“2947”
 - error:“*method def* child have incorrect indentation level 28, expected level should be one of the following: 32, 36.”
- – line:“2948”
 - error:“*method def* child have incorrect indentation level 28, expected level should be one of the following: 32, 36.”
- – line:“2949”

- error:“*method def rcurly* have incorrect indentation level 24, expected level should be one of the following: 28, 32.”
- – line:“2950”
 - error:“*object def rcurly* have incorrect indentation level 20, expected level should be one of the following: 24, 28.”
- – line:“2976”
 - error:“*implements* have incorrect indentation level 1, expected level should be 8.”
- – line:“3020”
 - error:“*method def modifier* have incorrect indentation level 4, expected level should be 8.”

4.2 Line length

- – line:“267”
 - error:“Line is longer than 80 characters (found 89).”
- – line:“272”
 - error:“Line is longer than 80 characters (found 118).”
- – line:“312”
 - error:“Line is longer than 80 characters (found 91).”
- – line:“321”
 - error:“Line is longer than 80 characters (found 91).”
- – line:“325”
 - error:“Line is longer than 80 characters (found 88).”
- – line:“346”
 - error:“Line is longer than 80 characters (found 89).”
- – line:“437”
 - error:“Line is longer than 80 characters (found 89).”
- – line:“438”

- error:“Line is longer than 80 characters (found 82).”
- - line:“440”
 - error:“Line is longer than 80 characters (found 87).”
- - line:“466”
 - error:“Line is longer than 80 characters (found 86).”
- - line:“486”
 - error:“Line is longer than 80 characters (found 83).”
- - line:“499”
 - error:“Line is longer than 80 characters (found 94).”
- - line:“786”
 - error:“Line is longer than 80 characters (found 85).”
- - line:“824”
 - error:“Line is longer than 80 characters (found 88).”
- - line:“827”
 - error:“Line is longer than 80 characters (found 99).”
- - line:“834”
 - error:“Line is longer than 80 characters (found 94).”
- - line:“837”
 - error:“Line is longer than 80 characters (found 101).”
- - line:“1004”
 - error:“Line is longer than 80 characters (found 99).”
- - line:“1005”
 - error:“Line is longer than 80 characters (found 87).”
- - line:“1006”
 - error:“Line is longer than 80 characters (found 96).”
- - line:“1007”
 - error:“Line is longer than 80 characters (found 96).”

- – line:“1008”
 - error:“Line is longer than 80 characters (found 97).”
- – line:“1009”
 - error:“Line is longer than 80 characters (found 96).”
- – line:“1013”
 - error:“Line is longer than 80 characters (found 97).”
- – line:“1015”
 - error:“Line is longer than 80 characters (found 99).”
- – line:“1016”
 - error:“Line is longer than 80 characters (found 87).”
- – line:“1017”
 - error:“Line is longer than 80 characters (found 88).”
- – line:“1018”
 - error:“Line is longer than 80 characters (found 96).”
- – line:“1019”
 - error:“Line is longer than 80 characters (found 97).”
- – line:“1020”
 - error:“Line is longer than 80 characters (found 82).”
- – line:“1022”
 - error:“Line is longer than 80 characters (found 105).”
- – line:“1024”
 - error:“Line is longer than 80 characters (found 99).”
- – line:“1025”
 - error:“Line is longer than 80 characters (found 87).”
- – line:“1026”
 - error:“Line is longer than 80 characters (found 92).”
- – line:“1030”

- error:“Line is longer than 80 characters (found 96).”
- - line:“1031”
 - error:“Line is longer than 80 characters (found 90).”
- - line:“1036”
 - error:“Line is longer than 80 characters (found 108).”
- - line:“1045”
 - error:“Line is longer than 80 characters (found 82).”
- - line:“1050”
 - error:“Line is longer than 80 characters (found 81).”
- - line:“1118”
 - error:“Line is longer than 80 characters (found 91).”
- - line:“1154”
 - error:“Line is longer than 80 characters (found 83).”
- - line:“1184”
 - error:“Line is longer than 80 characters (found 96).”
- - line:“1191”
 - error:“Line is longer than 80 characters (found 95).”
- - line:“1206”
 - error:“Line is longer than 80 characters (found 89).”
- - line:“1254”
 - error:“Line is longer than 80 characters (found 82).”
- - line:“1481”
 - error:“Line is longer than 80 characters (found 81).”
- - line:“1640”
 - error:“Line is longer than 80 characters (found 90).”
- - line:“1691”
 - error:“Line is longer than 80 characters (found 89).”

- – line:“1751”
– error:“Line is longer than 80 characters (found 85).”
- – line:“1774”
– error:“Line is longer than 80 characters (found 84).”
- – line:“1868”
– error:“Line is longer than 80 characters (found 91).”
- – line:“1975”
– error:“Line is longer than 80 characters (found 97).”
- – line:“2011”
– error:“Line is longer than 80 characters (found 92).”
- – line:“2070”
– error:“Line is longer than 80 characters (found 81).”
- – line:“2161”
– error:“Line is longer than 80 characters (found 81).”
- – line:“2171”
– error:“Line is longer than 80 characters (found 101).”
- – line:“2189”
– error:“Line is longer than 80 characters (found 81).”
- – line:“2199”
– error:“Line is longer than 80 characters (found 101).”
- – line:“2294”
– error:“Line is longer than 80 characters (found 88).”
- – line:“2295”
– error:“Line is longer than 80 characters (found 92).”
- – line:“2297”
– error:“Line is longer than 80 characters (found 94).”
- – line:“2301”

- error:“Line is longer than 80 characters (found 85).”
- - line:“2305”
 - error:“Line is longer than 80 characters (found 83).”
- - line:“2309”
 - error:“Line is longer than 80 characters (found 95).”
- - line:“2326”
 - error:“Line is longer than 80 characters (found 90).”
- - line:“2335”
 - error:“Line is longer than 80 characters (found 82).”
- - line:“2338”
 - error:“Line is longer than 80 characters (found 95).”
- - line:“2341”
 - error:“Line is longer than 80 characters (found 89).”
- - line:“2344”
 - error:“Line is longer than 80 characters (found 85).”
- - line:“2371”
 - error:“Line is longer than 80 characters (found 88).”
- - line:“2518”
 - error:“Line is longer than 80 characters (found 83).”
- - line:“2587”
 - error:“Line is longer than 80 characters (found 85).”
- - line:“2668”
 - error:“Line is longer than 80 characters (found 81).”
- - line:“2689”
 - error:“Line is longer than 80 characters (found 81).”
- - line:“2759”
 - error:“Line is longer than 80 characters (found 86).”

- – line:“2779”
 - error:“Line is longer than 80 characters (found 106).”
- – line:“2787”
 - error:“Line is longer than 80 characters (found 89).”
- – line:“2788”
 - error:“Line is longer than 80 characters (found 88).”
- – line:“2799”
 - error:“Line is longer than 80 characters (found 85).”
- – line:“2805”
 - error:“Line is longer than 80 characters (found 84).”
- – line:“2814”
 - error:“Line is longer than 80 characters (found 84).”
- – line:“2817”
 - error:“Line is longer than 80 characters (found 81).”
- – line:“2889”
 - error:“Line is longer than 80 characters (found 93).”
- – line:“2947”
 - error:“Line is longer than 80 characters (found 85).”
- – line:“3025”
 - error:“Line is longer than 80 characters (found 84).”

4.3 Bracket position

- – line:“190”
 - error:“{ at column 1 should be on the previous line.”
- – line:“456”
 - error:“{ at column 5 should be on the previous line.”
- – line:“566”

- error:“{ at column 5 should be on the previous line.”
- - line:“573”
 - error:“{ at column 5 should be on the previous line.”
- - line:“817”
 - error:“{ at column 5 should be on the previous line.”
- - line:“855”
 - error:“{ at column 5 should be on the previous line.”
- - line:“929”
 - error:“{ at column 5 should be on the previous line.”
- - line:“1084”
 - error:“{ at column 5 should be on the previous line.”
- - line:“1100”
 - error:“{ at column 5 should be on the previous line.”
- - line:“1151”
 - error:“{ at column 5 should be on the previous line.”
- - line:“1583”
 - error:“{ at column 2 should be on the previous line.”
- - line:“1697”
 - error:“{ at column 5 should be on the previous line.”
- - line:“1844”
 - error:“{ at column 5 should be on the previous line.”
- - line:“1912”
 - error:“{ at column 5 should be on the previous line.”
- - line:“2310”
 - error:“{ at column 17 should be on the previous line.”
- - line:“2383”
 - error:“{ at column 5 should be on the previous line.”

- – line:“2398”
 - error:“{ at column 5 should be on the previous line.”
- – line:“2529”
 - error:“{ at column 5 should be on the previous line.”
- – line:“2535”
 - error:“{ at column 5 should be on the previous line.”
- – line:“2657”
 - error:“{ at column 5 should be on the previous line.”
- – line:“2860”
 - error:“{ at column 5 should be on the previous line.”
- – line:“2963”
 - error:“{ at column 5 should be on the previous line.”
- – line:“2977”
 - error:“{ at column 5 should be on the previous line.”

4.4 Tab character

- – line:“216”
 - error:“Line contains a tab character.”
- – line:“222”
 - error:“Line contains a tab character.”
- – line:“243”
 - error:“Line contains a tab character.”
- – line:“244”
 - error:“Line contains a tab character.”
- – line:“245”
 - error:“Line contains a tab character.”
- – line:“254”

- error:“Line contains a tab character.”
- - line:“255”
 - error:“Line contains a tab character.”
- - line:“257”
 - error:“Line contains a tab character.”
- - line:“268”
 - error:“Line contains a tab character.”
- - line:“269”
 - error:“Line contains a tab character.”
- - line:“273”
 - error:“Line contains a tab character.”
- - line:“467”
 - error:“Line contains a tab character.”
- - line:“472”
 - error:“Line contains a tab character.”
- - line:“477”
 - error:“Line contains a tab character.”
- - line:“478”
 - error:“Line contains a tab character.”
- - line:“479”
 - error:“Line contains a tab character.”
- - line:“480”
 - error:“Line contains a tab character.”
- - line:“481”
 - error:“Line contains a tab character.”
- - line:“482”
 - error:“Line contains a tab character.”

- – line:“483”
 – error:“Line contains a tab character.”
- – line:“484”
 – error:“Line contains a tab character.”
- – line:“489”
 – error:“Line contains a tab character.”
- – line:“508”
 – error:“Line contains a tab character.”
- – line:“509”
 – error:“Line contains a tab character.”
- – line:“513”
 – error:“Line contains a tab character.”
- – line:“514”
 – error:“Line contains a tab character.”
- – line:“515”
 – error:“Line contains a tab character.”
- – line:“516”
 – error:“Line contains a tab character.”
- – line:“517”
 – error:“Line contains a tab character.”
- – line:“533”
 – error:“Line contains a tab character.”
- – line:“534”
 – error:“Line contains a tab character.”
- – line:“535”
 – error:“Line contains a tab character.”
- – line:“536”

- error:“Line contains a tab character.”
- - line:“537”
 - error:“Line contains a tab character.”
- - line:“538”
 - error:“Line contains a tab character.”
- - line:“540”
 - error:“Line contains a tab character.”
- - line:“544”
 - error:“Line contains a tab character.”
- - line:“548”
 - error:“Line contains a tab character.”
- - line:“552”
 - error:“Line contains a tab character.”
- - line:“556”
 - error:“Line contains a tab character.”
- - line:“557”
 - error:“Line contains a tab character.”
- - line:“558”
 - error:“Line contains a tab character.”
- - line:“803”
 - error:“Line contains a tab character.”
- - line:“853”
 - error:“Line contains a tab character.”
- - line:“854”
 - error:“Line contains a tab character.”
- - line:“856”
 - error:“Line contains a tab character.”

- – line:“857”
 – error:“Line contains a tab character.”
- – line:“858”
 – error:“Line contains a tab character.”
- – line:“899”
 – error:“Line contains a tab character.”
- – line:“903”
 – error:“Line contains a tab character.”
- – line:“904”
 – error:“Line contains a tab character.”
- – line:“905”
 – error:“Line contains a tab character.”
- – line:“906”
 – error:“Line contains a tab character.”
- – line:“907”
 – error:“Line contains a tab character.”
- – line:“908”
 – error:“Line contains a tab character.”
- – line:“909”
 – error:“Line contains a tab character.”
- – line:“910”
 – error:“Line contains a tab character.”
- – line:“911”
 – error:“Line contains a tab character.”
- – line:“912”
 – error:“Line contains a tab character.”
- – line:“1049”

- error:“Line contains a tab character.”
- - line:“1050”
 - error:“Line contains a tab character.”
- - line:“1052”
 - error:“Line contains a tab character.”
- - line:“1053”
 - error:“Line contains a tab character.”
- - line:“1054”
 - error:“Line contains a tab character.”
- - line:“1055”
 - error:“Line contains a tab character.”
- - line:“1056”
 - error:“Line contains a tab character.”
- - line:“1057”
 - error:“Line contains a tab character.”
- - line:“1058”
 - error:“Line contains a tab character.”
- - line:“1059”
 - error:“Line contains a tab character.”
- - line:“1075”
 - error:“Line contains a tab character.”
- - line:“1153”
 - error:“Line contains a tab character.”
- - line:“1362”
 - error:“Line contains a tab character.”
- - line:“1363”
 - error:“Line contains a tab character.”

- – line:“1365”
– error:“Line contains a tab character.”
- – line:“1366”
– error:“Line contains a tab character.”
- – line:“1367”
– error:“Line contains a tab character.”
- – line:“1386”
– error:“Line contains a tab character.”
- – line:“1537”
– error:“Line contains a tab character.”
- – line:“1538”
– error:“Line contains a tab character.”
- – line:“1539”
– error:“Line contains a tab character.”
- – line:“1548”
– error:“Line contains a tab character.”
- – line:“1581”
– error:“Line contains a tab character.”
- – line:“1582”
– error:“Line contains a tab character.”
- – line:“1583”
– error:“Line contains a tab character.”
- – line:“1584”
– error:“Line contains a tab character.”
- – line:“1585”
– error:“Line contains a tab character.”
- – line:“1586”

- error:“Line contains a tab character.”
- - line:“1588”
 - error:“Line contains a tab character.”
- - line:“1589”
 - error:“Line contains a tab character.”
- - line:“1590”
 - error:“Line contains a tab character.”
- - line:“1591”
 - error:“Line contains a tab character.”
- - line:“1735”
 - error:“Line contains a tab character.”
- - line:“1739”
 - error:“Line contains a tab character.”
- - line:“1877”
 - error:“Line contains a tab character.”
- - line:“2072”
 - error:“Line contains a tab character.”
- - line:“2073”
 - error:“Line contains a tab character.”
- - line:“2074”
 - error:“Line contains a tab character.”
- - line:“2142”
 - error:“Line contains a tab character.”
- - line:“2143”
 - error:“Line contains a tab character.”
- - line:“2144”
 - error:“Line contains a tab character.”

- – line:“2145”
 – error:“Line contains a tab character.”
- – line:“2146”
 – error:“Line contains a tab character.”
- – line:“2148”
 – error:“Line contains a tab character.”
- – line:“2149”
 – error:“Line contains a tab character.”
- – line:“2150”
 – error:“Line contains a tab character.”
- – line:“2161”
 – error:“Line contains a tab character.”
- – line:“2171”
 – error:“Line contains a tab character.”
- – line:“2172”
 – error:“Line contains a tab character.”
- – line:“2173”
 – error:“Line contains a tab character.”
- – line:“2174”
 – error:“Line contains a tab character.”
- – line:“2175”
 – error:“Line contains a tab character.”
- – line:“2177”
 – error:“Line contains a tab character.”
- – line:“2199”
 – error:“Line contains a tab character.”
- – line:“2200”

- error:“Line contains a tab character.”
- - line:“2201”
 - error:“Line contains a tab character.”
- - line:“2202”
 - error:“Line contains a tab character.”
- - line:“2227”
 - error:“Line contains a tab character.”
- - line:“2228”
 - error:“Line contains a tab character.”
- - line:“2229”
 - error:“Line contains a tab character.”
- - line:“2230”
 - error:“Line contains a tab character.”
- - line:“2320”
 - error:“Line contains a tab character.”
- - line:“2496”
 - error:“Line contains a tab character.”
- - line:“2976”
 - error:“Line contains a tab character.”
- - line:“2978”
 - error:“Line contains a tab character.”
- - line:“2979”
 - error:“Line contains a tab character.”
- - line:“2981”
 - error:“Line contains a tab character.”
- - line:“2982”
 - error:“Line contains a tab character.”

- – line:“2983”
 – error:“Line contains a tab character.”
- – line:“2984”
 – error:“Line contains a tab character.”
- – line:“2986”
 – error:“Line contains a tab character.”
- – line:“2987”
 – error:“Line contains a tab character.”
- – line:“2988”
 – error:“Line contains a tab character.”
- – line:“2989”
 – error:“Line contains a tab character.”
- – line:“2991”
 – error:“Line contains a tab character.”
- – line:“2992”
 – error:“Line contains a tab character.”
- – line:“2993”
 – error:“Line contains a tab character.”
- – line:“2994”
 – error:“Line contains a tab character.”
- – line:“2996”
 – error:“Line contains a tab character.”
- – line:“2997”
 – error:“Line contains a tab character.”
- – line:“2998”
 – error:“Line contains a tab character.”
- – line:“3000”

- error:“Line contains a tab character.”
- - line:“3001”
 - error:“Line contains a tab character.”
- - line:“3002”
 - error:“Line contains a tab character.”
- - line:“3004”
 - error:“Line contains a tab character.”
- - line:“3005”
 - error:“Line contains a tab character.”
- - line:“3006”
 - error:“Line contains a tab character.”
- - line:“3008”
 - error:“Line contains a tab character.”
- - line:“3009”
 - error:“Line contains a tab character.”
- - line:“3010”
 - error:“Line contains a tab character.”
- - line:“3012”
 - error:“Line contains a tab character.”
- - line:“3013”
 - error:“Line contains a tab character.”
- - line:“3014”
 - error:“Line contains a tab character.”
- - line:“3016”
 - error:“Line contains a tab character.”
- - line:“3017”
 - error:“Line contains a tab character.”

- – line:“3018”
 – error:“Line contains a tab character.”
- – line:“3021”
 – error:“Line contains a tab character.”
- – line:“3022”
 – error:“Line contains a tab character.”
- – line:“3023”
 – error:“Line contains a tab character.”
- – line:“3024”
 – error:“Line contains a tab character.”
- – line:“3025”
 – error:“Line contains a tab character.”
- – line:“3026”
 – error:“Line contains a tab character.”
- – line:“3035”
 – error:“Line contains a tab character.”
- – line:“3036”
 – error:“Line contains a tab character.”
- – line:“3039”
 – error:“Line contains a tab character.”
- – line:“3040”
 – error:“Line contains a tab character.”
- – line:“3044”
 – error:“Line contains a tab character.”
- – line:“3045”
 – error:“Line contains a tab character.”
- – line:“3047”

- error:“Line contains a tab character.”
- - line:“3048”
 - error:“Line contains a tab character.”
- - line:“3049”
 - error:“Line contains a tab character.”
- - line:“3050”
 - error:“Line contains a tab character.”
- - line:“3051”
 - error:“Line contains a tab character.”
- - line:“3052”
 - error:“Line contains a tab character.”
- - line:“3053”
 - error:“Line contains a tab character.”
- - line:“3055”
 - error:“Line contains a tab character.”
- - line:“3059”
 - error:“Line contains a tab character.”
- - line:“3060”
 - error:“Line contains a tab character.”
- - line:“3061”
 - error:“Line contains a tab character.”
- - line:“3062”
 - error:“Line contains a tab character.”
- - line:“3066”
 - error:“Line contains a tab character.”
- - line:“3067”
 - error:“Line contains a tab character.”

- – line:“3069”
 – error:“Line contains a tab character.”
- – line:“3070”
 – error:“Line contains a tab character.”
- – line:“3071”
 – error:“Line contains a tab character.”
- – line:“3072”
 – error:“Line contains a tab character.”
- – line:“3073”
 – error:“Line contains a tab character.”
- – line:“3074”
 – error:“Line contains a tab character.”
- – line:“3075”
 – error:“Line contains a tab character.”
- – line:“3076”
 – error:“Line contains a tab character.”
- – line:“3077”
 – error:“Line contains a tab character.”
- – line:“3078”
 – error:“Line contains a tab character.”
- – line:“3079”
 – error:“Line contains a tab character.”
- – line:“3080”
 – error:“Line contains a tab character.”
- – line:“3081”
 – error:“Line contains a tab character.”
- – line:“3083”

- error:“Line contains a tab character.”
- - line:“3088”
 - error:“Line contains a tab character.”
- - line:“3089”
 - error:“Line contains a tab character.”
- - line:“3091”
 - error:“Line contains a tab character.”
- - line:“3092”
 - error:“Line contains a tab character.”
- - line:“3093”
 - error:“Line contains a tab character.”
- - line:“3094”
 - error:“Line contains a tab character.”
- - line:“3095”
 - error:“Line contains a tab character.”
- - line:“3096”
 - error:“Line contains a tab character.”
- - line:“3097”
 - error:“Line contains a tab character.”
- - line:“3098”
 - error:“Line contains a tab character.”
- - line:“3099”
 - error:“Line contains a tab character.”
- - line:“3100”
 - error:“Line contains a tab character.”
- - line:“3101”
 - error:“Line contains a tab character.”

- – line:“3102”
– error:“Line contains a tab character.”
- – line:“3103”
– error:“Line contains a tab character.”
- – line:“3105”
– error:“Line contains a tab character.”
- – line:“3109”
– error:“Line contains a tab character.”
- – line:“3113”
– error:“Line contains a tab character.”

4.5 Missing brackets

- – line:“621”
– error:“*if construct must use {}s.*”
- – line:“623”
– error:“*if construct must use {}s.*”
- – line:“639”
– error:“*if construct must use {}s.*”
- – line:“641”
– error:“*if construct must use {}s.*”
- – line:“645”
– error:“*if construct must use {}s.*”
- – line:“697”
– error:“*if construct must use {}s.*”
- – line:“772”
– error:“*if construct must use {}s.*”
- – line:“775”

- error:“*else construct must use {}s.*”
- - line:“818”
 - error:“*if construct must use {}s.*”
- - line:“940”
 - error:“*if construct must use {}s.*”
- - line:“942”
 - error:“*else construct must use {}s.*”
- - line:“959”
 - error:“*if construct must use {}s.*”
- - line:“961”
 - error:“*else construct must use {}s.*”
- - line:“971”
 - error:“*if construct must use {}s.*”
- - line:“973”
 - error:“*else construct must use {}s.*”
- - line:“1433”
 - error:“*if construct must use {}s.*”
- - line:“1786”
 - error:“*if construct must use {}s.*”
- - line:“2011”
 - error:“*if construct must use {}s.*”
- - line:“2360”
 - error:“*if construct must use {}s.*”

4.6 Naming patterns

- – line:“152”
– error:“Local variable name *l* must match the right pattern.”
- – line:“510”
– error:“Local variable name *k16_1* must match the right pattern.”
- – line:“511”
– error:“Local variable name *k16_2* must match the right pattern.”
- – line:“227”
– error:“Member name *DEFAULT_LOAD_FACTOR* must match the right pattern.”
- – line:“228”
– error:“Member name *DEFAULT_CACHE_SIZE* must match the right pattern.”
- – line:“332”
– error:“Local variable name *e* must match the right pattern.”
- – line:“607”
– error:“Method name *_getContext* must match the right pattern.”
- – line:“856”
– error:“Local variable name *pKeys* must match the right pattern.”
- – line:“933”
– error:“Local variable name *e* must match the right pattern.”
- – line:“952”
– error:“Local variable name *c* must match the right pattern.”
- – line:“1101”
– error:“Local variable name *i* must match the right pattern.”
- – line:“1448”
– error:“Local variable name *e* must match the right pattern.”
- – line:“1491”

- error:“Local variable name *e* must match the right pattern.”
- – line:“1609”
 - error:“Local variable name *e* must match the right pattern.”
- – line:“69”
 - error:“Member name *nCallsInProgress* must match the right pattern.”
- – line:“115”
 - error:“Parameter name *b* must match the right pattern.”
- – line:“120”
 - error:“Parameter name *s* must match the right pattern.”
- – line:“134”
 - error:“Parameter name *b* must match the right pattern.”
- – line:“275”
 - error:“Method name *_setNext* must match the right pattern.”
- – line:“279”
 - error:“Method name *_getNext* must match the right pattern.”
- – line:“283”
 - error:“Method name *_getPKHashCode* must match the right pattern.”
- – line:“125”
 - error:“Member name *RELATIVE_TIME_CHECK_MODE* must match the right pattern.”
- – line:“236”
 - error:“Method name *_getContext* must match the right pattern.”
- – line:“425”
 - error:“Local variable name *e* must match the right pattern.”
- – line:“827”
 - error:“Local variable name *e* must match the right pattern.”

- – line:“835”
 - error:“Local variable name *c* must match the right pattern.”
- – line:“893”
 - error:“Parameter name *v* must match the right pattern.”
- – line:“930”
 - error:“Parameter name *o* must match the right pattern.”
- – line:“84”
 - error:“Method name *_refresh_com_sun_ejb_containers_read_only_bean_* must match the right pattern.”
- – line:“92”
 - error:“Method name *_refresh_All* must match the right pattern.”
- – line:“52”
 - error:“Method name *_refresh_com_sun_ejb_containers_read_only_bean_* must match the right pattern.”
- – line:“55”
 - error:“Method name *_refresh_All* must match the right pattern.”

Appendices