Item 224
git_comments:

1. * Licensed to the Apache Software Foundation (ASF) under one * or more contributor license agreements. See the NOTICE file * distributed with this work for additional information * regarding copyright ownership. The ASF licenses this file * to you under the Apache License, Version 2.0 (the * "License"); you may not use this file except in compliance * with the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, * software distributed under the License is distributed on an * "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY * KIND, either express or implied. See the License for the * specific language governing permissions and limitations * under the License.
2. NOI18N
3. the array is not sufficiently homogeneous.
4. skipped fix for invalid array member and for parameterized array member.
5. * * * @author arusinha
6. * * * @author arusinha
7. * Licensed to the Apache Software Foundation (ASF) under one * or more contributor license agreements. See the NOTICE file * distributed with this work for additional information * regarding copyright ownership. The ASF licenses this file * to you under the Apache License, Version 2.0 (the * "License"); you may not use this file except in compliance * with the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, * software distributed under the License is distributed on an * "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY * KIND, either express or implied. See the License for the * specific language governing permissions and limitations * under the License.
8. NOI18N
9. * Licensed to the Apache Software Foundation (ASF) under one * or more contributor license agreements. See the NOTICE file * distributed with this work for additional information * regarding copyright ownership. The ASF licenses this file * to you under the Apache License, Version 2.0 (the * "License"); you may not use this file except in compliance * with the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, * software distributed under the License is distributed on an * "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY * KIND, either express or implied. See the License for the * specific language governing permissions and limitations * under the License.
10. * * Replaces invalid var type array initialization statement with explicit * array type. * * @param arrayType : target explicit array type. * @throws IOException
11. * * Tests conversion of invalid var type variable to explicit array type. * * @author arusinha
12. converts var type to explicit array type
13. Correcting lower/upper bounds for oldT.vartype tree.
14. adding back all whitespaces/block-comment/javadoc-comments present in OldTree before variable type token.
15. * * For erroneous variable type sometime diffTree function return * wrong position. In that scenario only old variable type will * be replaced with new variable type leaving out succeeding * comments tokens present(if any). Below code copies successive * tokens after excluding variable type token which will be the * first token.
16. moving to variable type token
17. copying modifiers from oldTree
18. copying tokens from vartype bounds after excluding variable type token.
19. * * Extracting modifier from oldTree using symbol position and * modifier positions when oldT.type is error and vartype * upperbound is not proper.
20. moving to first token after variable type token.
21. returns -1 if modifiers not present.

git_commits:

1. **summary:** [NETBEANS-481] JDK10-LVTI: Added new ErrorRule to fix compiler error on initialization of var type variable with array (#519)
   message: [NETBEANS-481] JDK10-LVTI: Added new ErrorRule to fix compiler error on initialization of var type variable with array (#519) * netbeans-481: Added new ErrorRule to fix compiler error on initialization of var type variable with array * netbeans-481: Added setup() method in

ConvertInvalidVarToExplicitArrayTypeTest * netbeans-481: Refactored code of CasualDiff class * netbeans-481: Refactored code of ConvertInvalidVarToExplicitArrayType class * [NETBEANS-481] Refactored code of ConvertInvalidVarToExplicitArrayType class * [NETBEANS-481] JDK10-LVTI: Added tests for CasualDiff class changes * [NETBEANS-481] JDK10-LVTI: Refactored Test class for CasualDiff changes * [NETBEANS-481] JDK10-LVTI: 1.Refactored Test File for CasualDiff changes 2. Corrected Code for Skipping hint for Parameterized type array members * [NETBEANS-481] JDK10-LVTI: Corrected fix label text * [NETBEANS-481] JDK10-LVTI: Refactored CasualDiff to handle any tokensequence overshoot scenario * [NETBEANS-481] JDK10-LVTI: Handled scenarios related to empty array/invalid array as var initializer

github_issues:

github_issues_comments:

github_pulls:

1. **title:** [NETBEANS-481] JDK10-LVTI: Added new ErrorRule to fix compiler error on initialization of var type variable with array
   body: Jira Id : https://issues.apache.org/jira/browse/NETBEANS-481 Below statement throws compiler error. var arr = {1,2}; : error: cannot infer type for local variable var k = { 1 , 2 }; ^ (array initializer needs an explicit target-type) Proposed fix: int[] arr = { 1, 2 }; The fix is provided for numeric primitive type array or for array with homogeneous members. a)Homogeneous type 1. var arr = { new Object(), new Object()}; fix: Object[] arr = { new Object(), new Object()}; 2. var arr = {"hello", "world"}; fix: String[] arr = {"hello", "world"}; b)Primitive type numeric array var arr = {1,2.2}; fix: double[] arr = {1,2.2};

github_pulls_comments:

1. Great, thanks for the work on this and for the reviewers.

github_pulls_reviews:

1. The change was made to run TestCase in JDK10, otherwise getting error can't access package java.lang
2. You can use setup() method to avoid duplicate code of sourceLevel = "1.10"; JavacParser.DISABLE_SOURCE_LEVEL_DOWNGRADE = true; @Override protected void setUp() throws Exception { super.setUp(); sourceLevel = "1.10"; JavacParser.DISABLE_SOURCE_LEVEL_DOWNGRADE = true; }
3. Do not use `printer.print` directly, `copyTo()` handles copying with respect to guarded blocks.
4. use `copyUpTo()`, it checks for the boundaries already.
5. Corrected in latest commit
6. Corrected in latest commit
7. This looks good I think.
8. **body:** General comment: please minimize calls to Trees.getScope. That calls are inevitable fairly slow.
   label: code-design
9. (Unless I am missing something) this is re-attributing the live tree (the tree that is under CompilationInfo.getCompilationUnit(), effects of which would see all subsequent tasks. This should not be done, the attributed tree either needs to be a copy, or (ideally) compilationInfo.getTrees().getTypeMirror could be used (passing in "new TreePath(new TreePath(treePath, arrayTree), tree)" - hoisting the "new TreePath(treePath, arrayTree)" out of the loop).
10. Better: tree.getKind() == Tree.Kind.NEW_CLASS.
11. **body:** This check appears to be unnecessary, right? If maxTypePriority == -1, then maxTypePriority < arrayElementPriority, and the next if statement will set the maxTypePriority?
    label: code-design
12. Poulation of arrayElementPriority/maxTypePriority starts when it found out array is hetrogeneous. In the below scenarion it will detect array is hetrogenous when it traverse 2nd element {2.1F,1} it that point maxTypePriority =-1,arrayElementPriority = 3(int),arrayTypeMirror = float type If we use only the below check,this will populate maxTypePriority = 3 (int) which is incorrect. if(maxTypePriority < arrayElementPriority) maxTypePriority = arrayElementPriority; For this reason using the (maxTypePriority == -1) check
13. corrected in latest commit.
14. Corrected in latest commit.

15. Corrected in latest commit.
16. Ah, right, sorry I got confused with the names. I was thinking on how to simplify the logic. And this might work (untested): TypeMirror arrayType = null; for (ExpressionTree et : currentValues) { TypeMirror etType = ...; if (arrayType == null) { arrayType = etType; } else if (!types.isAssignable(etType, arrayType)) { if (types.isAssignable(arrayType, etType)) { //here I am not completely sure if isAssignable is transitive arrayType = etType; } else { return null; //the array is not sufficiently homogeneous. } } } Would this work?
17. Corrected in latest commit.
18. **body:** I think the limitation to skip on parameterized types is OK at this time. But relying on tree structure appears to be less ideal, what if the expression is e.g. "list.get(0)" - is there a chance the etType could be checked? (e.g. getKind() == TypeKind.DECLARED && !DeclaredType.getTypeArguments.isEmpty()). **label:** code-design
19. Had implemented the review comment in last commit
20. if `move` moves tokensequence after last token, `moveNext` fails - what will be the result of `tokenSequence.token()` ? An exception ?
21. if `moveNext` fails to move the token sequence (end of stream), the `tokenId` will remain the same (see below) and the wile will loop endlessly. Maybe not because of `offset` check, but still.
22. Handled tokensequence overshoot issue.
23. Handled tokensequence overshoot issue
24. The former code in commit #96003dd59aa2003dcbf086b231f8a5d5403e818a checked for `JavaTokenId.WHITESPACE` at `modsUpperBound`, and if there was whitespace did not copy it over. This update changes that behaviour, copying up to `modsUpperBound`. If that change is OK, then the tokenSequence work on lines 1479-1482 is probably useless, the sequence is positioned, but no token is consumed - and regardless of execution branch taken, the `tokenSequence` is always repositioned to a new place at line 1489.
25. **body:** check for JavaTokenId.WHITESPACE at modsUpperBound was removed in subsequent commit as check was not required after the code refactoring. Plz consider below scenario. Scenario1: /*comment1*/var/*comment2*/ a = {2,3.1f}; For above case getCommentCorrectedEndPos(oldT.mods) will be -1 . So Netbeans will not run code inside the check (modsUpperBound > -1) present at line 1478 In that case it will loop at line 1492 to add all the whitespaces/comments. Scenario2: @NotNull var j = {new Object(),new Object()}; in this case code inside getCommentCorrectedEndPos(oldT.mods) >1 check will add '@NotNull' to the final statement string. But we need to add the whitespace present after '@NotNull' to the final string. For that reason while loop at line 1492 is required. it loops at line 1492 to add back the whitespace. Scenario3: final/*comment1*/var /*comment2*/ a = {2,3.1f}; check (modsUpperBound > -1) at line 1478 will add 'final/*comment1*/' to the final string. In this scenario modsUpperBound offset is pointing to VAR token. As we want to skip adding VAR token in final String tokenSequence.move(localPointer) is required at line 1489. Otherwise it will add wrongly 'var /*comment2*/' to the final string after looping at line 1492
    label: code-design
26. OK, I get why the change was done. And sorry, `copyTo` does not handle after-end-of-text position gracefully, my fault.
27. What will happen if the initializer array is empty? (I.e. "var v = {};".)
28. The scenario was not handled, Had corrected in last commit scenarios related to ' empty array /invalid array' as var initializer

jira_issues:

1. **summary:** New ErrorRule to fix compiler error on initialization of var type variable with array **description:** Below statement throws compiler error. var arr = \{1,2}; : error: cannot infer type for local variable var k = \{ 1 , 2 }; ^ (array initializer needs an explicit target-type) Proposed fix would be int[] arr = \{ 1, 2 }; The fix will be provided for Numeric primitive type array or for array with homogeneous members. a)Homogeneous type 1. var arr = \{ new Object(), new Object()}; fix: Object[] arr = \{ new Object(), new Object()}; 2. var arr = \{"hello", "world"}; fix: String[] arr = \{"hello", "world"}; b)Primitive Type Numeric array var arr = \{1,2.2}; fix: double[] arr = \{1,2.2}; PR Link: [https://github.com/apache/incubator-netbeans/pull/519]

jira_issues_comments:

1. The link to the PR is missing on this issue.

2. Closed as the issue has been resolved more than 180 days.