

OccurrencesSearchGroup.java - ReadReferencesSearchGroup.java

```

1/*****
2 * Copyright (c) 2000, 2008 IBM Corporation and others.
3 * All rights reserved. This program and the accompanying materials
4 * are made available under the terms of the Eclipse Public License v1.0
5 * which accompanies this distribution, and is available at
6 * http://www.eclipse.org/legal/epl-v10.html
7 *
8 * Contributors:
9 *   IBM Corporation - initial API and implementation
10 *****/
11 package org.eclipse.jdt.ui.actions;
12
13 import org.eclipse.core.runtime.Assert;
14
15 import org.eclipse.jface.action.Action;
16 import org.eclipse.jface.action.MenuListener;
17 import org.eclipse.jface.action.IMenuManager;
18 import org.eclipse.jface.action.MenuManager;
19
20 import org.eclipse.jface.viewers.ISelection;
21 import org.eclipse.jface.viewers.ISelectionChangedListener;
22 import org.eclipse.jface.viewers.ISelectionProvider;
23 import org.eclipse.jface.viewers.IStructuredSelection;
24
25 import org.eclipse.jface.text.IDocument;
26 import org.eclipse.jface.text.ITextSelection;
27
28 import org.eclipse.ui.IActionBars;
29 import org.eclipse.ui.IWorkbenchSite;
30 import org.eclipse.ui.PlatformUI;
31 import org.eclipse.ui.actions.ActionGroup;
32 import org.eclipse.ui.keys.IBindingService;
33
34 import org.eclipse.ui.texteditor.ITextEditorActionConstants;
35
36 import org.eclipse.jdt.core.ITypeRoot;
37
38 import org.eclipse.jdt.ui.IContextMenuConstants;
39 import org.eclipse.jdt.ui.JavaUI;
40
41 import org.eclipse.jdt.internal.ui.actions.SelectionConverter;
42 import org.eclipse.jdt.internal.ui.javaeditor.JavaEditor;
43 import org.eclipse.jdt.internal.ui.javaeditor.JavaTextSelection;
44 import org.eclipse.jdt.internal.ui.search.SearchMessages;
45
46 /**
47 * Action group that adds the occurrences in file actions
48 * to a context menu and the global menu bar.
49 *
50 * This class may be instantiated; it is not intended to be subclassed.
51 */
52

```

```

1/*****
2 * Copyright (c) 2000, 2008 IBM Corporation and others.
3 * All rights reserved. This program and the accompanying materials
4 * are made available under the terms of the Eclipse Public License v1.0
5 * which accompanies this distribution, and is available at
6 * http://www.eclipse.org/legal/epl-v10.html
7 *
8 * Contributors:
9 *   IBM Corporation - initial API and implementation
10 *****/
11 package org.eclipse.jdt.ui.actions;
12
13 import java.util.Iterator;
14
15 import org.eclipse.core.runtime.Assert;
16
17 import org.eclipse.jface.action.IAction;
18
19 import org.eclipse.jface.action.IMenuManager;
20 import org.eclipse.jface.action.MenuManager;
21 import org.eclipse.jface.action.Separator;
22 import org.eclipse.jface.viewers.ISelection;
23 import org.eclipse.jface.viewers.ISelectionChangedListener;
24 import org.eclipse.jface.viewers.ISelectionProvider;
25 import org.eclipse.jface.viewers.IStructuredSelection;
26
27 import org.eclipse.ui.IActionBars;
28 import org.eclipse.ui.IWorkbenchSite;
29 import org.eclipse.ui.IWorkingSet;
30 import org.eclipse.ui.actions.ActionGroup;
31
32 import org.eclipse.ui.texteditor.ITextEditorActionConstants;
33
34 import org.eclipse.jdt.ui.IContextMenuConstants;
35
36 import org.eclipse.jdt.internal.ui.javaeditor.JavaEditor;
37 import org.eclipse.jdt.internal.ui.search.SearchMessages;
38 import org.eclipse.jdt.internal.ui.search.SearchUtil;
39
40 /**
41 * Action group that adds the search for read references actions to a
42 * context menu and the global menu bar.
43 *
44 * This class may be instantiated; it is not intended to be subclassed.
45 */
46

```

OccurrencesSearchGroup.java - ReadReferencesSearchGroup.java

```

53 * @since 3.1
54 *
55 * @noextend This class is not intended to be subclassed by clients.
56 */
57 public class OccurrencesSearchGroup extends ActionGroup {
58
59     private IWorkbenchSite fSite;
60     private JavaEditor fEditor;
61     private IActionBars fActionBars;
62
63     private String fGroupId;
64
65     private FindOccurrencesInFileAction fOccurrencesInFileAction;
66     private FindExceptionOccurrencesAction fExceptionOccurrencesAction;
67     private FindImplementOccurrencesAction fFindImplementorOccurrencesAction;
68     private FindBreakContinueTargetOccurrencesAction
69     fBreakContinueTargetOccurrencesAction;
69     private FindMethodExitOccurrencesAction fMethodExitOccurrencesAction;
70
71     /**
72      * Creates a new <code>ImplementorsSearchGroup</code>. The group
73      * requires that the selection provided by the site's selection provider
74      * is of type <code>org.eclipse.jface.viewers.IStructuredSelection</code>.
75      *
76      * @param site the view part that owns this action group
77      */
78     public OccurrencesSearchGroup(IWorkbenchSite site) {
79         this(site, null);
80     }
81
82     /**
83      * Creates a new <code>OccurrencesSearchGroup</code>. The group requires
84      * that the selection provided by the given selection provider is of type
85      * {@link IStructuredSelection}.
86      *
87      * @param site the site that will own the action group.
88      * @param specialSelectionProvider the selection provider used instead of the
89      * sites selection provider.
90      *
91      * @since 3.4
92      */
93     public OccurrencesSearchGroup(IWorkbenchSite site, ISelectionProvider
94     specialSelectionProvider) {
95         fSite= site;
96         fGroupId= IContextMenuConstants.GROUP_SEARCH;
97
98         fOccurrencesInFileAction= new FindOccurrencesInFileAction(site);
99         fOccurrencesInFileAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_OCCURRENCES_IN_FILE);
100         // Need to reset the label

```

```

47 * @since 2.0
48 *
49 * @noextend This class is not intended to be subclassed by clients.
50 */
51 public class ReadReferencesSearchGroup extends ActionGroup {
52
53     private static final String MENU_TEXT= SearchMessages.group_readReferences;
54
55     private IWorkbenchSite fSite;
56     private JavaEditor fEditor;
57     private IActionBars fActionBars;
58
59     private String fGroupId;
60
61     private FindReadReferencesAction fFindReadReferencesAction;
62     private FindReadReferencesInProjectAction fFindReadReferencesInProjectAction;
63     private FindReadReferencesInHierarchyAction
64     fFindReadReferencesInHierarchyAction;
64     private FindReadReferencesInWorkingSetAction
65     fFindReadReferencesInWorkingSetAction;
66
67     /**
68      * Creates a new <code>ReadReferencesSearchGroup</code>. The group requires
69      * that the selection provided by the site's selection provider is of type
70      * <code>org.eclipse.jface.viewers.IStructuredSelection</code>.
71      *
72      * @param site the view part that owns this action group
73      */
74     public ReadReferencesSearchGroup(IWorkbenchSite site) {
75         this(site, null);
76     }
77
78     /**
79      * Creates a new <code>ReadReferencesSearchGroup</code>. The group requires
80      * that the selection provided by the given selection provider is of type
81      * {@link IStructuredSelection}.
82      *
83      * @param site the site that will own the action group.
84      * @param specialSelectionProvider the selection provider used instead of the
85      * sites selection provider.
86      *
87      * @since 3.4
88      */
89     public ReadReferencesSearchGroup(IWorkbenchSite site, ISelectionProvider
90     specialSelectionProvider) {
91         fSite= site;
92         fGroupId= IContextMenuConstants.GROUP_SEARCH;
93
94         fFindReadReferencesAction= new FindReadReferencesAction(site);
95         fFindReadReferencesAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_WRITE_ACCESS_IN_WORKSPACE);

```

OccurrencesSearchGroup.java - ReadReferencesSearchGroup.java

```

100 fOccurrencesInFileAction.setText(SearchMessages.Search_FindOccurrencesInFile_shortLabel);
101
102     fExceptionOccurrencesAction= new FindExceptionOccurrencesAction(site);
103     fExceptionOccurrencesAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_EXCEPTION_OCCURRENCES_IN_FILE);
104
105     fFindImplementorOccurrencesAction= new FindImplementorOccurrencesAction(site);
106     fFindImplementorOccurrencesAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_IMPLEMENT_OCCURRENCES_IN_FILE);
107
108     fBreakContinueTargetOccurrencesAction= new FindBreakContinueTargetOccurrencesAction(site);
109     fBreakContinueTargetOccurrencesAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_BREAK_CONTINUE_TARGET_OCCURRENCES);
110
111     fMethodExitOccurrencesAction= new FindMethodExitOccurrencesAction(site);
112     fMethodExitOccurrencesAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_METHOD_EXIT_OCCURRENCES);
113
114
115     // register the actions as selection listeners
116     ISelectionProvider provider= specialSelectionProvider == null ?
117     fSite.getSelectionProvider() : specialSelectionProvider;
118     registerAction(fOccurrencesInFileAction, provider, selection, specialSelectionProvider);
119     registerAction(fExceptionOccurrencesAction, provider, selection, specialSelectionProvider);
120     registerAction(fFindImplementorOccurrencesAction, provider, selection, specialSelectionProvider);
121     registerAction(fBreakContinueTargetOccurrencesAction, provider, selection, specialSelectionProvider);
122     registerAction(fMethodExitOccurrencesAction, provider, selection, specialSelectionProvider);
123 }
124
125 /**
126  * Note: This constructor is for internal use only. Clients should not call this constructor.
127  *
128  * @param editor the Java editor
129  *
130  * @noreference This constructor is not intended to be referenced by clients.
131  */

```

```

94
95     fFindReadReferencesInProjectAction= new FindReadReferencesInProjectAction(site);
96     fFindReadReferencesInProjectAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_READ_ACCESS_IN_PROJECT);
97
98     fFindReadReferencesInHierarchyAction= new FindReadReferencesInHierarchyAction(site);
99     fFindReadReferencesInHierarchyAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_WRITE_ACCESS_IN_HIERARCHY);
100
101     fFindReadReferencesInWorkingSetAction= new FindReadReferencesInWorkingSetAction(site);
102     fFindReadReferencesInWorkingSetAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_WRITE_ACCESS_IN_WORKING_SET);
103
104     // register the actions as selection listeners
105     ISelectionProvider provider= specialSelectionProvider == null ?
106     fSite.getSelectionProvider() : specialSelectionProvider;
107     registerAction(fFindReadReferencesAction, provider, selection, specialSelectionProvider);
108     registerAction(fFindReadReferencesInProjectAction, provider, selection, specialSelectionProvider);
109     registerAction(fFindReadReferencesInHierarchyAction, provider, selection, specialSelectionProvider);
110     registerAction(fFindReadReferencesInWorkingSetAction, provider, selection, specialSelectionProvider);
111 }
112
113 /**
114  * Note: This constructor is for internal use only. Clients should not call this constructor.
115  *
116  * @param editor the Java editor
117  *
118  * @noreference This constructor is not intended to be referenced by clients.
119  */

```

OccurrencesSearchGroup.java - ReadReferencesSearchGroup.java

```

132 public OccurrencesSearchGroup(JavaEditor editor) {
133     fEditor= editor;
134     fSite= fEditor.getSite();
135     fGroupId= ITextEditorActionConstants.GROUP_FIND;
136
137     fOccurrencesInFileAction= new FindOccurrencesInFileAction(fEditor);
138     fOccurrencesInFileAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_OCCURRENCES_IN_FILE);
139     // Need to reset the label
140     fOccurrencesInFileAction.setText(SearchMessages.Search_FindOccurrencesInFile_shortLabel);
141     fEditor.setAction("SearchOccurrencesInFile", fOccurrencesInFileAction);
142     //$NON-NLS-1$
143
144     fExceptionOccurrencesAction= new FindExceptionOccurrencesAction(fEditor);
145     fExceptionOccurrencesAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_EXCEPTION_OCCURRENCES_IN_FILE);
146     fEditor.setAction("SearchExceptionOccurrences", fExceptionOccurrencesAction);
147     //$NON-NLS-1$
148
149     fFindImplementorOccurrencesAction= new FindImplementOccurrencesAction(fEditor);
150     fFindImplementorOccurrencesAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_IMPLEMENT_OCCURRENCES_IN_FILE);
151     fEditor.setAction("SearchImplementOccurrences", fFindImplementorOccurrencesAction);
152     //$NON-NLS-1$
153
154     fBreakContinueTargetOccurrencesAction= new FindBreakContinueTargetOccurrencesAction(fEditor);
155     fBreakContinueTargetOccurrencesAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_BREAK_CONTINUE_TARGET_OCCURRENCES);
156     fEditor.setAction("BreakContinueTargetOccurrences", fBreakContinueTargetOccurrencesAction);
157     //$NON-NLS-1$
158
159     fMethodExitOccurrencesAction= new FindMethodExitOccurrencesAction(fEditor);
160     fMethodExitOccurrencesAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_METHOD_EXIT_OCCURRENCES);
161     fEditor.setAction("ExitOccurrencesAction", fMethodExitOccurrencesAction);
162     //$NON-NLS-1$
163
164     ISelectionProvider provider= fSite.getSelectionProvider();

```

```

119 public ReadReferencesSearchGroup(JavaEditor editor) {
120     fEditor= editor;
121     fSite= fEditor.getSite();
122     fGroupId= ITextEditorActionConstants.GROUP_FIND;
123
124     fFindReadReferencesAction= new FindReadReferencesAction(fEditor);
125     fFindReadReferencesAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_WRITE_ACCESS_IN_WORKSPACE);
126     fEditor.setAction("SearchReadAccessInWorkspace", fFindReadReferencesAction);
127     //$NON-NLS-1$
128
129     fFindReadReferencesInProjectAction= new FindReadReferencesInProjectAction(fEditor);
130     fFindReadReferencesInProjectAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_READ_ACCESS_IN_PROJECT);
131     fEditor.setAction("SearchReadAccessInProject", fFindReadReferencesInProjectAction);
132     //$NON-NLS-1$
133
134     fFindReadReferencesInHierarchyAction= new FindReadReferencesInHierarchyAction(fEditor);
135     fFindReadReferencesInHierarchyAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_WRITE_ACCESS_IN_HIERARCHY);
136     fEditor.setAction("SearchReadAccessInHierarchy", fFindReadReferencesInHierarchyAction);
137     //$NON-NLS-1$
138
139     fFindReadReferencesInWorkingSetAction= new FindReadReferencesInWorkingSetAction(fEditor);
140     fFindReadReferencesInWorkingSetAction.setActionDefinitionId(IJavaEditorActionDefinitionIds.SEARCH_WRITE_ACCESS_IN_WORKING_SET);
141     fEditor.setAction("SearchReadAccessInWorkingSet", fFindReadReferencesInWorkingSetAction);
142     //$NON-NLS-1$

```

OccurrencesSearchGroup.java - ReadReferencesSearchGroup.java

```

161//      ISelection selection= provider.getSelection();
162//
163//      registerAction(fOccurrencesInFileAction, provider, selection);
164//      registerAction(fExceptionOccurrencesAction, provider, selection);
165//      registerAction(fFindImplementorOccurrencesAction, provider, selection);
166    }
167
168    private void registerAction(SelectionDispatchAction action, ISelectionProvider
169    provider, ISelection selection, ISelectionProvider specialSelectionProvider) {
170        action.update(selection);
171        provider.addSelectionChangedListener(action);
172        if (specialSelectionProvider != null)
173            action.setSpecialSelectionProvider(specialSelectionProvider);
174    }
175
176    /*
177     * Method declared on ActionGroup.
178     */
179    public void fillContextMenu(IMenuManager manager) {
180        String menuText= SearchMessages.group_occurrences;
181        String shortcut= getShortcutString();
182        if (shortcut != null) {
183            menuText= menuText + '¥t' + shortcut;
184        }
185
186        MenuManager javaSearchMM= new MenuManager(menuText,
187        IContextMenuConstants.GROUP_SEARCH);
188        javaSearchMM.add(new Action() {
189        });
190        javaSearchMM.addMenuListener(new IMenuListener() {
191            public void menuAboutToShow(IMenuManager mm) {
192                mm.removeAll();
193                updateActionsInJavaEditor();
194                addAction(fOccurrencesInFileAction, mm);
195                addAction(fFindImplementorOccurrencesAction, mm);
196                addAction(fExceptionOccurrencesAction, mm);
197                addAction(fMethodExitOccurrencesAction, mm);
198                addAction(fBreakContinueTargetOccurrencesAction, mm);
199                if (mm.isEmpty()) {
200                    mm.add(new
201                    Action(SearchMessages.group_occurrences_quickMenu_noEntriesAvailable) {
202                        public boolean isEnabled() {
203                            return false;
204                        }
205                    });
206                }
207            }
208        });
209
210        private void addAction(Action action, IMenuManager mm) {
211            if (action.isEnabled())
212                mm.add(action);
213        }
214    }
215
216    manager.appendToGroup(fGroupId, javaSearchMM);
217    }
218
219    }
220
221    }
222
223    }
224
225    }
226
227    }
228
229    }
230
231    }
232
233    }
234
235    }
236
237    }
238
239    }
240
241    }
242
243    }
244
245    }
246
247    }
248
249    }
250
251    }
252
253    }
254
255    }
256
257    }
258
259    }
260
261    }
262
263    }
264
265    }
266
267    }
268
269    }
270
271    }
272
273    }
274
275    }
276
277    }
278
279    }
280
281    }
282
283    }
284
285    }
286
287    }
288
289    }
290
291    }
292
293    }
294
295    }
296
297    }
298
299    }
300
301    }
302
303    }
304
305    }
306
307    }
308
309    }
310
311    }
312
313    }
314
315    }
316
317    }
318
319    }
320
321    }
322
323    }
324
325    }
326
327    }
328
329    }
330
331    }
332
333    }
334
335    }
336
337    }
338
339    }
340
341    }
342
343    }
344
345    }
346
347    }
348
349    }
350
351    }
352
353    }
354
355    }
356
357    }
358
359    }
360
361    }
362
363    }
364
365    }
366
367    }
368
369    }
370
371    }
372
373    }
374
375    }
376
377    }
378
379    }
380
381    }
382
383    }
384
385    }
386
387    }
388
389    }
390
391    }
392
393    }
394
395    }
396
397    }
398
399    }
400
401    }
402
403    }
404
405    }
406
407    }
408
409    }
410
411    }
412
413    }
414
415    }
416
417    }
418
419    }
420
421    }
422
423    }
424
425    }
426
427    }
428
429    }
430
431    }
432
433    }
434
435    }
436
437    }
438
439    }
440
441    }
442
443    }
444
445    }
446
447    }
448
449    }
450
451    }
452
453    }
454
455    }
456
457    }
458
459    }
460
461    }
462
463    }
464
465    }
466
467    }
468
469    }
470
471    }
472
473    }
474
475    }
476
477    }
478
479    }
480
481    }
482
483    }
484
485    }
486
487    }
488
489    }
490
491    }
492
493    }
494
495    }
496
497    }
498
499    }
500
501    }
502
503    }
504
505    }
506
507    }
508
509    }
510
511    }
512
513    }
514
515    }
516
517    }
518
519    }
520
521    }
522
523    }
524
525    }
526
527    }
528
529    }
530
531    }
532
533    }
534
535    }
536
537    }
538
539    }
540
541    }
542
543    }
544
545    }
546
547    }
548
549    }
550
551    }
552
553    }
554
555    }
556
557    }
558
559    }
560
561    }
562
563    }
564
565    }
566
567    }
568
569    }
570
571    }
572
573    }
574
575    }
576
577    }
578
579    }
580
581    }
582
583    }
584
585    }
586
587    }
588
589    }
590
591    }
592
593    }
594
595    }
596
597    }
598
599    }
600
601    }
602
603    }
604
605    }
606
607    }
608
609    }
610
611    }
612
613    }
614
615    }
616
617    }
618
619    }
620
621    }
622
623    }
624
625    }
626
627    }
628
629    }
630
631    }
632
633    }
634
635    }
636
637    }
638
639    }
640
641    }
642
643    }
644
645    }
646
647    }
648
649    }
650
651    }
652
653    }
654
655    }
656
657    }
658
659    }
660
661    }
662
663    }
664
665    }
666
667    }
668
669    }
670
671    }
672
673    }
674
675    }
676
677    }
678
679    }
680
681    }
682
683    }
684
685    }
686
687    }
688
689    }
690
691    }
692
693    }
694
695    }
696
697    }
698
699    }
700
701    }
702
703    }
704
705    }
706
707    }
708
709    }
710
711    }
712
713    }
714
715    }
716
717    }
718
719    }
720
721    }
722
723    }
724
725    }
726
727    }
728
729    }
730
731    }
732
733    }
734
735    }
736
737    }
738
739    }
740
741    }
742
743    }
744
745    }
746
747    }
748
749    }
750
751    }
752
753    }
754
755    }
756
757    }
758
759    }
760
761    }
762
763    }
764
765    }
766
767    }
768
769    }
770
771    }
772
773    }
774
775    }
776
777    }
778
779    }
780
781    }
782
783    }
784
785    }
786
787    }
788
789    }
790
791    }
792
793    }
794
795    }
796
797    }
798
799    }
800
801    }
802
803    }
804
805    }
806
807    }
808
809    }
810
811    }
812
813    }
814
815    }
816
817    }
818
819    }
820
821    }
822
823    }
824
825    }
826
827    }
828
829    }
830
831    }
832
833    }
834
835    }
836
837    }
838
839    }
840
841    }
842
843    }
844
845    }
846
847    }
848
849    }
850
851    }
852
853    }
854
855    }
856
857    }
858
859    }
860
861    }
862
863    }
864
865    }
866
867    }
868
869    }
870
871    }
872
873    }
874
875    }
876
877    }
878
879    }
880
881    }
882
883    }
884
885    }
886
887    }
888
889    }
890
891    }
892
893    }
894
895    }
896
897    }
898
899    }
900
901    }
902
903    }
904
905    }
906
907    }
908
909    }
910
911    }
912
913    }
914
915    }
916
917    }
918
919    }
920
921    }
922
923    }
924
925    }
926
927    }
928
929    }
930
931    }
932
933    }
934
935    }
936
937    }
938
939    }
940
941    }
942
943    }
944
945    }
946
947    }
948
949    }
950
951    }
952
953    }
954
955    }
956
957    }
958
959    }
960
961    }
962
963    }
964
965    }
966
967    }
968
969    }
970
971    }
972
973    }
974
975    }
976
977    }
978
979    }
980
981    }
982
983    }
984
985    }
986
987    }
988
989    }
990
991    }
992
993    }
994
995    }
996
997    }
998
999    }
1000
1001    }
1002
1003    }
1004
1005    }
1006
1007    }
1008
1009    }
1010
1011    }
1012
1013    }
1014
1015    }
1016
1017    }
1018
1019    }
1020
1021    }
1022
1023    }
1024
1025    }
1026
1027    }
1028
1029    }
1030
1031    }
1032
1033    }
1034
1035    }
1036
1037    }
1038
1039    }
1040
1041    }
1042
1043    }
1044
1045    }
1046
1047    }
1048
1049    }
1050
1051    }
1052
1053    }
1054
1055    }
1056
1057    }
1058
1059    }
1060
1061    }
1062
1063    }
1064
1065    }
1066
1067    }
1068
1069    }
1070
1071    }
1072
1073    }
1074
1075    }
1076
1077    }
1078
1079    }
1080
1081    }
1082
1083    }
1084
1085    }
1086
1087    }
1088
1089    }
1090
1091    }
1092
1093    }
1094
1095    }
1096
1097    }
1098
1099    }
1100
1101    }
1102
1103    }
1104
1105    }
1106
1107    }
1108
1109    }
1110
1111    }
1112
1113    }
1114
1115    }
1116
1117    }
1118
1119    }
1120
1121    }
1122
1123    }
1124
1125    }
1126
1127    }
1128
1129    }
1130
1131    }
1132
1133    }
1134
1135    }
1136
1137    }
1138
1139    }
1140
1141    }
1142
1143    }
1144
1145    }
1146
1147    }
1148
1149    }
1150
1151    }
1152
1153    }
1154
1155    }
1156
1157    }
1158
1159    }
1160
1161    }
1162
1163    }
1164
1165    }
1166
1167    }
1168
1169    }
1170
1171    }
1172
1173    }
1174
1175    }
1176
1177    }
1178
1179    }
1180
1181    }
1182
1183    }
1184
1185    }
1186
1187    }
1188
1189    }
1190
1191    }
1192
1193    }
1194
1195    }
1196
1197    }
1198
1199    }
1200
1201    }
1202
1203    }
1204
1205    }
1206
1207    }
1208
1209    }
1210
1211    }
1212
1213    }
1214
1215    }
1216
1217    }
1218
1219    }
1220
1221    }
1222
1223    }
1224
1225    }
1226
1227    }
1228
1229    }
1230
1231    }
1232
1233    }
1234
1235    }
1236
1237    }
1238
1239    }
1240
1241    }
1242
1243    }
1244
1245    }
1246
1247    }
1248
1249    }
1250
1251    }
1252
1253    }
1254
1255    }
1256
1257    }
1258
1259    }
1260
1261    }
1262
1263    }
1264
1265    }
1266
1267    }
1268
1269    }
1270
1271    }
1272
1273    }
1274
1275    }
1276
1277    }
1278
1279    }
1280
1281    }
1282
1283    }
1284
1285    }
1286
1287    }
1288
1289    }
1290
1291    }
1292
1293    }
1294
1295    }
1296
1297    }
1298
1299    }
1300
1301    }
1302
1303    }
1304
1305    }
1306
1307    }
1308
1309    }
1310
1311    }
1312
1313    }
1314
1315    }
1316
1317    }
1318
1319    }
1320
1321    }
1322
1323    }
1324
1325    }
1326
1327    }
1328
1329    }
1330
1331    }
1332
1333    }
1334
1335    }
1336
1337    }
1338
1339    }
1340
1341    }
1342
1343    }
1344
1345    }
1346
1347    }
1348
1349    }
1350
1351    }
1352
1353    }
1354
1355    }
1356
1357    }
1358
1359    }
1360
1361    }
1362
1363    }
1364
1365    }
1366
1367    }
1368
1369    }
1370
1371    }
1372
1373    }
1374
1375    }
1376
1377    }
1378
1379    }
1380
1381    }
1382
1383    }
1384
1385    }
1386
1387    }
1388
1389    }
1390
1391    }
1392
1393    }
1394
1395    }
1396
1397    }
1398
1399    }
1400
1401    }
1402
1403    }
1404
1405    }
1406
1407    }
1408
1409    }
1410
1411    }
1412
1413    }
1414
1415    }
1416
1417    }
1418
1419    }
1420
1421    }
1422
1423    }
1424
1425    }
1426
1427    }
1428
1429    }
1430
1431    }
1432
1433    }
1434
1435    }
1436
1437    }
1438
1439    }
1440
1441    }
1442
1443    }
1444
1445    }
1446
1447    }
1448
1449    }
1450
1451    }
1452
1453    }
1454
1455    }
1456
1457    }
1458
1459    }
1460
1461    }
1462
1463    }
1464
1465    }
1466
1467    }
1468
1469    }
1470
1471    }
1472
1473    }
1474
1475    }
1476
1477    }
1478
1479    }
1480
1481    }
1482
1483    }
1484
1485    }
1486
1487    }
1488
1489    }
1490
1491    }
1492
1493    }
1494
1495    }
1496
1497    }
1498
1499    }
1500
1501    }
1502
1503    }
1504
1505    }
1506
1507    }
1508
1509    }
1510
1511    }
1512
1513    }
1514
1515    }
1516
1517    }
1518
1519    }
1520
1521    }
1522
1523    }
1524
1525    }
1526
1527    }
1528
1529    }
1530
1531    }
1532
1533    }
1534
1535    }
1536
1537    }
1538
1539    }
1540
1541    }
1542
1543    }
1544
1545    }
1546
1547    }
1548
1549    }
1550
1551    }
1552
1553    }
1554
1555    }
1556
1557    }
1558
1559    }
1560
1561    }
1562
1563    }
1564
1565    }
1566
1567    }
1568
1569    }
1570
1571    }
1572
1573    }
1574
1575    }
1576
1577    }
1578
1579    }
1580
1581    }
1582
1583    }
1584
1585    }
1586
1587    }
1588
1589    }
1590
1591    }
1592
1593    }
1594
1595    }
1596
1597    }
1598
1599    }
1600
1601    }
1602
1603    }
1604
1605    }
1606
1607    }
1608
1609    }
1610
1611    }
1612
1613    }
1614
1615    }
1616
1617    }
1618
1619    }
1620
1621    }
1622
1623    }
1624
1625    }
1626
1627    }
1628
1629    }
1630
1631    }
1632
1633    }
1634
1635    }
1636
1637    }
1638
1639    }
1640
1641    }
1642
1643    }
1644
1645    }
1646
1647    }
1648
1649    }
1650
1651    }
1652
1653    }
1654
1655    }
1656
1657    }
1658
1659    }
1660
1661    }
1662
1663    }
1664
1665    }
1666
1667    }
1668
1669    }
1670
1671    }
1672
1673    }
1674
1675    }
1676
1677    }
1678
1679    }
1680
1681    }
1682
1683    }
1684
1685    }
1686
1687    }
1688
1689    }
1690
1691    }
1692
1693    }
1694
1695    }
1696
1697    }
1698
1699    }
1700
1701    }
1702
1703    }
1704
1705    }
1706
1707    }
1708
1709    }
1710
1711    }
1712
1713    }
1714
1715    }
1716
1717    }
1718
1719    }
1720
1721    }
1722
1723    }
1724
1725    }
1726
1727    }
1728
1729    }
1730
1731    }
1732
1733    }
1734
1735    }
1736
1737    }
1738
1739    }
1740
1741    }
1742
1743    }
1744
1745    }
1746
1747    }
1748
1749    }
1750
1751    }
1752
1753    }
1754
1755    }
1756
1757    }
1758
1759    }
1760
1761    }
1762
1763    }
1764
1765    }
1766
1767    }
1768
1769    }
1770
1771    }
1772
1773    }
1774
1775    }
1776
1777    }
1778
1779    }
1780
1781    }
1782
1783    }
1784
1785    }
1786
1787    }
1788
1789    }
1790
1791    }
1792
1793    }
1794
1795    }
1796
1797    }
1798
1799    }
1800
1801    }
1802
1803    }
1804
1805    }
1806
1807    }
1808
1809    }
1810
1811    }
1812
1813    }
1814
1815    }
1816
1817    }
1818
1819    }
1820
1821    }
1822
1823    }
1824
1825    }
1826
1827    }
1828
1829    }
1830
1831    }
1832
1833    }
1834
1835    }
1836
1837    }
1838
1839    }
1840
1841    }
1842
1843    }
1844
1845    }
1846
1847    }
1848
1849    }
1850
1851    }
1852
1853    }
1854
1855    }
1856
1857    }
1858
1859    }
1860
1861    }
1862
1863    }
1864
1865    }
1866
1867    }
1868
1869    }
1870
1871    }
1872
1873    }
1874
1875    }
1876
1877    }
1878
1879    }
1880
1881    }
1882
1883    }
1884
1885    }
1886
1887    }
1888
1889    }
1890
1891    }
1892
1893    }
1894
1895    }
1896
1897    }
1898
1899    }
1900
1901    }
1902
1903    }
1904
1905    }
1906
1907    }
1908
1909    }
1910
1911    }
1912
1913    }
1914
1915    }
1916
1917    }
1918
1919    }
1920
1921    }
1922
1923    }
1924
1925    }
1926
1927    }
1928
1929    }
1930
1931    }
1932
1933    }
1934
1935    }
1936
1937    }
1938
1939    }
1940
1941    }
1942
1943    }
1944
1945    }
1946
1947    }
1948
1949    }
1950
1951    }
1952
1953    }
1954
1955    }
1956
1957    }
1958
1959    }
1960
1961    }
1962
1963    }
1964
1965    }
1966
1967    }
1968
1969    }
1970
1971    }
1972
1973    }
1974
1975    }
1976
1977    }
1978
1979    }
1980
1981    }
1982
1983    }
1984
1985    }
1986
1987    }
1988
1989    }
1990
1991    }
1992
1993    }
1994
1995    }
1996
1997    }
1998
1999    }
2000
2001    }
2002
2003    }
2004
2005    }
2006
2007    }
2008
2009    }
2010
2011    }
2012
2013    }
2014
2015    }
2016
2017    }
2018
2019    }
2020
2021    }
2022
2023    }
2024
2025    }
2026
2027    }
2028
2029    }
2030
2031    }
2032
2033    }
2034
2035    }
2036
2037    }
2038
2039    }
2040
2041    }
2042
2043    }
2044
2045    }
2046
2047    }
2048
2049    }
2050
2051    }
2052
2053    }
2054
2055    }
2056
2057    }
2058
2059    }
2060
2061    }
2062
2063    }
2064
2065    }
2066
2067    }
2068
2069    }
2070
2071    }
2072
2073    }
2074
2075    }
2076
2077    }
2078
2079    }
2080
2081    }
2082
2083    }
2084
2085    }
2086
2087    }
2088
2089    }
2090
2091    }
2092
2093    }
2094
2095    }
2096
2097    }
2098
2099    }
2100
2101    }
2102
2103    }
2104
2105    }
2106
2107    }
2108
2109    }
2110
2111    }
2112
2113    }
2114
2115    }
2116
2117    }
2118
2119    }
2120
2121    }
2122
2123    }
2124
2125    }
2126
2127    }
2128
2129    }
2130
2131    }
2132
2133    }
2134
2135    }
2136
2137    }
2138
2139    }
2140
2141    }
2142
2143    }
2144
2145    }
2146
2147    }
2148
2149    }
2150
2151    }
2152
2153    }
2154
2155    }
2156
2157    }
2158
2159    }
2160
2161    }
2162
2163    }
2164
2165    }
2166
2167    }
2168
2169    }
2170
2171    }
2172
2173    }
2174
2175    }
2176
2177    }
2178
2179    }
2180
2181    }
2182
2183    }
2184
2185    }
2186
2187    }
2188
2189    }
2190
2191    }
2192
2193    }
2194
2195    }
2196
2197    }
2198
2199    }
2200
2201    }
2202
2203    }
2204
2205    }
2206
2207    }
2208
2209    }
2210
2211    }
2212
2213    }
2214
2215    }
2216
2217    }
2218
2219    }
2220
2221    }
2222
2223    }
2224
2225    }
2226
2227    }
2228
2229    }
2230
2231    }
2232
2233    }
2234
2235    }
2236
2237    }
2238
2239    }
2240
2241    }
2242
2243    }
2244
2245    }
2246
2247    }
2248
2249    }
2250
2251    }
2252
2253    }
2254
2255    }
2256
2257    }
2258
2259    }
2260
2261    }
2262
2263    }
2264
2265    }
2266
2267    }
2268
2269    }
2270
2271    }
2272
2273    }
2274
2275    }
2276
2277    }
2278
2279    }
2280
2281    }
2282
2283    }
2284
2285    }
2286
2287    }
2288
2289    }
2290
2291    }

```

OccurrencesSearchGroup.java - ReadReferencesSearchGroup.java

```

214 private void updateActionsInJavaEditor() {
215     if (fEditor == null)
216         return;
217
218     ITypeRoot element= SelectionConverter.getInput(fEditor);
219     if (element == null)
220
221         return;
222     ITextSelection textSelection= (ITextSelection)
fEditor.getSelectionProvider().getSelection();
223     IDocument document=
JavaUI.getDocumentProvider().getDocument(fEditor.getEditorInput());
224     JavaTextSelection javaSelection= new JavaTextSelection(element, document,
textSelection.getOffset(), textSelection.getLength());
225
226     fExceptionOccurrencesAction.update(javaSelection);
227     fOccurrencesInFileAction.update(javaSelection);
228     fFindImplementorOccurrencesAction.update(javaSelection);
229     fBreakContinueTargetOccurrencesAction.update(javaSelection);
230     fMethodExitOccurrencesAction.update(javaSelection);
231 }
232
233 private String getShortcutString() {
234     IBindingService bindingService=
(IBindingService)PlatformUI.getWorkbench().getAdapter(IBindingService.class);
235     if (bindingService == null)
236         return null;
237     return
bindingService.getBestActiveBindingFormattedFor(IJavaEditorActionDefinitionIds.SEAR
CH_OCCURRENCES_IN_FILE_QUICK_MENU);
238 }
239
240 /*
241  * Method declared on ActionGroup.
242  */
243
154 private void addWorkingSetAction(IWorkingSet[] workingSets, IMenuManager
manager) {
155     FindAction action;
156     if (fEditor != null)
157         action= new WorkingSetFindAction(fEditor, new
FindReadReferencesInWorkingSetAction(fEditor, workingSets),
SearchUtil.toString(workingSets));
158     else
159         action= new WorkingSetFindAction(fSite, new
FindReadReferencesInWorkingSetAction(fSite, workingSets),
SearchUtil.toString(workingSets));
160     action.update(getContext().getSelection());
161     addAction(action, manager);
162 }
163
164 /* (non-Javadoc)
165  * Method declared on ActionGroup.
166  */
167
168 public void fillContextMenu(IMenuManager manager) {
169     MenuManager javaSearchMM= new MenuManager(MENU_TEXT,
IContextMenuConstants.GROUP_SEARCH);
170     addAction(fFindReadReferencesAction, javaSearchMM);
171     addAction(fFindReadReferencesInProjectAction, javaSearchMM);
172     addAction(fFindReadReferencesInHierarchyAction, javaSearchMM);
173
174     javaSearchMM.add(new Separator());
175
176     Iterator iter= SearchUtil.getLRUWorkingSets().sortedIterator();
177     while (iter.hasNext()) {
178         addWorkingSetAction((IWorkingSet[]) iter.next(), javaSearchMM);
179     }
180     addAction(fFindReadReferencesInWorkingSetAction, javaSearchMM);
181
182     if (!javaSearchMM.isEmpty())
183         manager.appendToGroup(fGroupId, javaSearchMM);
184 }
185
186 /*
187  * Method declared on ActionGroup.
188  */

```

OccurrencesSearchGroup.java - ReadReferencesSearchGroup.java

```

243 public void fillActionBars(IActionBars actionBars) {
244     Assert.isNotNull(actionBars);
245     super.fillActionBars(actionBars);
246     fActionBars= actionBars;
247     updateGlobalActionHandlers();
248 }
249
250 /*
251  * Method declared on ActionGroup.
252  */
253 public void dispose() {
254     ISelectionProvider provider= fSite.getSelectionProvider();
255     if (provider != null) {
256         disposeAction(fFindImplementorOccurrencesAction, provider);
257         disposeAction(fExceptionOccurrencesAction, provider);
258         disposeAction(fOccurrencesInFileAction, provider);
259         disposeAction(fMethodExitOccurrencesAction, provider);
260         disposeAction(fBreakContinueTargetOccurrencesAction, provider);
261     }
262     super.dispose();
263     fFindImplementorOccurrencesAction= null;
264     fExceptionOccurrencesAction= null;
265     fOccurrencesInFileAction= null;
266     fMethodExitOccurrencesAction= null;
267     fBreakContinueTargetOccurrencesAction= null;
268     updateGlobalActionHandlers();
269 }
270
271 private void updateGlobalActionHandlers() {
272     if (fActionBars != null) {
273         fActionBars.setGlobalActionHandler(JdtActionConstants.FIND_OCCURRENCES_IN_FILE,
274             fOccurrencesInFileAction);
275         fActionBars.setGlobalActionHandler(JdtActionConstants.FIND_EXCEPTION_OCCURRENCES,
276             fExceptionOccurrencesAction);
277         fActionBars.setGlobalActionHandler(JdtActionConstants.FIND_IMPLEMENT_OCCURRENCES,
278             fFindImplementorOccurrencesAction);
279         fActionBars.setGlobalActionHandler(JdtActionConstants.FIND_BREAK_CONTINUE_TARGET_OC
280             CURRENCES, fBreakContinueTargetOccurrencesAction);
281         fActionBars.setGlobalActionHandler(JdtActionConstants.FIND_METHOD_EXIT_OCCURRENCES,
282             fMethodExitOccurrencesAction);
283     }
284 }
285
286 private void disposeAction(ISelectionChangedListener action,
287     ISelectionProvider provider) {
288     if (action != null)
289         provider.removeSelectionChangedListener(action);
290 }

```

```

189 public void fillActionBars(IActionBars actionBars) {
190     Assert.isNotNull(actionBars);
191     super.fillActionBars(actionBars);
192     fActionBars= actionBars;
193     updateGlobalActionHandlers();
194 }
195
196 /*
197  * Method declared on ActionGroup.
198  */
199 public void dispose() {
200     ISelectionProvider provider= fSite.getSelectionProvider();
201     if (provider != null) {
202         disposeAction(fFindReadReferencesAction, provider);
203         disposeAction(fFindReadReferencesInProjectAction, provider);
204         disposeAction(fFindReadReferencesInHierarchyAction, provider);
205         disposeAction(fFindReadReferencesInWorkingSetAction, provider);
206     }
207     fFindReadReferencesAction= null;
208     fFindReadReferencesInProjectAction= null;
209     fFindReadReferencesInHierarchyAction= null;
210     fFindReadReferencesInWorkingSetAction= null;
211     updateGlobalActionHandlers();
212     super.dispose();
213 }
214
215 private void updateGlobalActionHandlers() {
216     if (fActionBars != null) {
217         fActionBars.setGlobalActionHandler(JdtActionConstants.FIND_READ_ACCESS_IN_WORKSPACE,
218             fFindReadReferencesAction);
219         fActionBars.setGlobalActionHandler(JdtActionConstants.FIND_READ_ACCESS_IN_PROJECT,
220             fFindReadReferencesInProjectAction);
221         fActionBars.setGlobalActionHandler(JdtActionConstants.FIND_READ_ACCESS_IN_HIERARCHY,
222             fFindReadReferencesInHierarchyAction);
223         fActionBars.setGlobalActionHandler(JdtActionConstants.FIND_READ_ACCESS_IN_WORKING_S
224             ET, fFindReadReferencesInWorkingSetAction);
225     }
226 }
227
228 private void disposeAction(ISelectionChangedListener action,
229     ISelectionProvider provider) {
230     if (action != null)
231         provider.removeSelectionChangedListener(action);
232 }

```