001/\*  
002 \* Licensed to the Apache Software Foundation (ASF) under one or more  
003 \* contributor license agreements. See the NOTICE file distributed with  
004 \* this work for additional information regarding copyright ownership.  
005 \* The ASF licenses this file to You under the Apache License, Version 2.0  
006 \* (the "License"); you may not use this file except in compliance with  
007 \* the License. You may obtain a copy of the License at  
008 \*  
009 \* http://www.apache.org/licenses/LICENSE-2.0  
010 \*  
011 \* Unless required by applicable law or agreed to in writing, software  
012 \* distributed under the License is distributed on an "AS IS" BASIS,  
013 \* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
014 \* See the License for the specific language governing permissions and  
015 \* limitations under the License.  
016 \*/  
017package org.apache.commons.collections4.sequence;  
018  
019import java.util.ArrayList;  
020import java.util.List;  
021  
022/\*\*  
023 \* This class handles sequences of replacements resulting from a comparison.  
024 \* <p>  
025 \* The comparison of two objects sequences leads to the identification of common  
026 \* parts and parts which only belong to the first or to the second sequence. The  
027 \* common parts appear in the edit script in the form of <em>keep</em> commands,  
028 \* they can be considered as synchronization objects between the two sequences.  
029 \* These synchronization objects split the two sequences in synchronized  
030 \* sub-sequences. The first sequence can be transformed into the second one by  
031 \* replacing each synchronized sub-sequence of the first sequence by the  
032 \* corresponding sub-sequence of the second sequence. This is a synthetic way to  
033 \* see an {@link EditScript edit script}, replacing individual  
034 \* {@link DeleteCommand delete}, {@link KeepCommand keep} and  
035 \* {@link InsertCommand insert} commands by fewer replacements acting on  
036 \* complete sub-sequences.  
037 \* </p>  
038 \* <p>  
039 \* This class is devoted to perform this interpretation. It visits an  
040 \* {@link EditScript edit script} (because it implements the  
041 \* {@link CommandVisitor CommandVisitor} interface) and calls a user-supplied  
042 \* handler implementing the {@link ReplacementsHandler ReplacementsHandler}  
043 \* interface to process the sub-sequences.  
044 \* </p>  
045 \*  
046 \* @see ReplacementsHandler  
047 \* @see EditScript  
048 \* @see SequencesComparator  
049 \*  
050 \* @since 4.0  
051 \*/  
052public class ReplacementsFinder<T> implements CommandVisitor<T> {  
053  
054 private final List<T> pendingInsertions;  
055 private final List<T> pendingDeletions;  
056 private int skipped;  
057  
058 /\*\* Handler to call when synchronized sequences are found. \*/  
059 private final ReplacementsHandler<T> handler;  
060  
061 /\*\*  
062 \* Simple constructor. Creates a new instance of {@link ReplacementsFinder}.  
063 \*  
064 \* @param handler handler to call when synchronized sequences are found  
065 \*/  
066 public ReplacementsFinder(final ReplacementsHandler<T> handler) {  
067 pendingInsertions = new ArrayList<>();  
068 pendingDeletions = new ArrayList<>();  
069 skipped = 0;  
070 this.handler = handler;  
071 }  
072  
073 /\*\*  
074 \* Add an object to the pending insertions set.  
075 \*  
076 \* @param object object to insert  
077 \*/  
078 @Override  
079 public void visitInsertCommand(final T object) {  
080 pendingInsertions.add(object);  
081 }  
082  
083 /\*\*  
084 \* Handle a synchronization object.  
085 \* <p>  
086 \* When a synchronization object is identified, the pending insertions and  
087 \* pending deletions sets are provided to the user handler as subsequences.  
088 \*  
089 \* @param object synchronization object detected  
090 \*/  
091 @Override  
092 public void visitKeepCommand(final T object) {  
093 if (pendingDeletions.isEmpty() && pendingInsertions.isEmpty()) {  
094 ++skipped;  
095 } else {  
096 handler.handleReplacement(skipped, pendingDeletions, pendingInsertions);  
097 pendingDeletions.clear();  
098 pendingInsertions.clear();  
099 skipped = 1;  
100 }  
101 }  
102  
103 /\*\*  
104 \* Add an object to the pending deletions set.  
105 \*  
106 \* @param object object to delete  
107 \*/  
108 @Override  
109 public void visitDeleteCommand(final T object) {  
110 pendingDeletions.add(object);  
111 }  
112  
113}