```
diff /org/gjt/sp/jedit/actions.xml
324
> <ACTION NAME="find-and-mark-all" NO_RECORD="TRUE">
>   <CODE>
>       SearchAndReplace.markAll(view);
>       </CODE>
> </ACTION>
>

diff /org/gjt/sp/jedit/Buffer.java
1269
<       if(getMarkerAtLine(line) != null)
---
>       if(!toggleMarkersRemovable && getMarkerAtLine(line) != null)
1664
>   private static boolean toggleMarkersRemovable = false;
1995
>   /**
>    * disableMarkersRemovable(true) to ensure a call to addMarker()
>    * will only result in markers being added.                    ↵
disableMarkersRemovable(false)
>    * returns the addMarker() behaviour to its default.
>    *
>    * @param toggle true disengages marker removal process. false
>    * means addMarker() will remove markers, at its discretion.
>    */
>   public void disableMarkersRemovable(boolean toggle){
>       toggleMarkersRemovable = toggle;
>   }

diff /org/gjt/sp/jedit/jedit_gui.props
371
>        find-and-mark-all \
392
> find-and-mark-all.label=Mark All

diff /org/gjt/sp/jedit/search/SearchAndReplace.java
28
> import java.util.ArrayList;
422
>   /**
>    * place a marker at each line where the highlighted search
>    * pattern exists in the current buffer.
>    */
>   public static void markAll(View view){
>       try{
>           int[] matches = findAll(view);
>           int i=0;
>           //prevent calls to addMarker() from toggling off       ↵
existing markers
>           view.getBuffer().disableMarkersRemovable(true);
>
>           if(matches!=null){ // highlighted string was found at  ↵
least once
>               for(int integer:matches){
>                   // place the abstract 'caret' to beginning of  ↵
each line
>                   // where a match was
>                   view.getTextArea().setCaretPosition(
>                                                                  ↵
view.getTextArea().getLineStartOffset(integer)
```

```
57  >                       );
58  >                   // place marker to show the line as marked
59  >                   view.getEditPane().addMarker();
60  >               }
61  >           }
62  >           //restore buffer.toggleMarkersRemovable to its default
63  >           view.getBuffer().disableMarkersRemovable(false);
64  >
65  >       }
66  >       catch(Exception e){
67  >           // display red message in console if user is confused.
68  >           System.err.println("exception in find_and_mark caused
    by "+e+".\n"
69  >                   + "You must have selected some text to begin
    with..");
70  >       }
71  >   }
72  >   /**
73  >    * find lines where the highlighted text is found in the buffer
74  >    * @param view The View object
75  >    * @return a list of the line numbers where matches are found
76  >    * @throws Exception SearchAndReplace.findAll() exception is
    thrown
77  >    * when no text was highlighted prior at time of method's calling
78  >    */
79  >   public static int[] findAll(View view) throws Exception
80  >   {
81  >       String text = view.getTextArea().getSelectedText();
82  >       if(text==null)
83  >           throw new Exception("SearchAndReplace.findAll(view)");
84  >       caretStart = 0;
85  >
86  >       // method variables to ready findAll() enhancements so all
    files
87  >       // in the active session can be marked vs. the active buffer
88  >       // if this enhancement is desired in future. otherwise, we
    could
89  >       // retrieve the buffer associated with
    fileset.getFiles(view)[0]
90  >       // to keep with SearchAndReplace original authors' style
    or just
91  >       // use view.getBuffer()
92  >
93  >       int fileIndex = 0; //
94  >       SearchFileSet fileset; // set of searchable files for this
    .jEdit session
95  >       String[] files = null; // names of files in the SearchFileSet
96  >       ArrayList<Integer> hits = new ArrayList<Integer>();
97  >       int i=0; // generic counter
98  >
99  >       try{
100 >           fileset = SearchAndReplace.getSearchFileSet();
101 >           files = fileset.getFiles(view);
102 >           if(files!=null){
103 >               Buffer b =
    jEdit.openTemporary(null,null,files[fileIndex],false);
104 >               if(b != null){
105 >                   String s = b.getText(0, b.getLength());//all
    text in the
106 >                   // current buffer
107 >                   if(text!=null){ // some text was highlighted
```

```
in the buffer
108 >                         if(s.contains(text)){ //and the
highlighted text was
109 >                             // found as a match in the buffer
1(+?) times
110 >                             for(i=0;i<b.getLineCount();i++){
//iterate by lines
111 >                                 // of text, the current buffer
112 >                                 String line=b.getLineText(i);
113 >                                 if(line.contains(text)){
114 >                                     hits.add(new Integer(i));
115 >                                     // can call println(i) here to
see lines
116 >                                     // where searchMatches are found
117 >                                 }
118 >
119 >                             }
120 >                             int[] searchMatches = new
int[hits.size()];
121 >                             i=0;
122 >                             for(Integer bigInt:hits){
123 >                                 searchMatches[i++]=bigInt; //java
unboxing will handle
124 >                                 // the cast for us from big
Integer to little integer
125 >                             }
126 >                             return searchMatches;
127 >                         }
128 >                     } // text!=null is false, therefore:
129 >                         else{
130 >                             // no text was highlighted
131 >                             throw new
Exception("SearchAndReplace.findAll(View)");
132 >                         }
133 >                     }
134 >             } // there may have been a problem getting the
SearchFileSet
135 >             // which would confuse the author of this method.
136 >         } catch (Exception e) {
137 >
138 >         }
139 >         return null;
140 >     }
141 >
142 1010
143 >     private static int caretStart;
```