victoryforphil / compsci-2018-lessons

Branch: master ▾ compsci-2018-lessons / src / projects / magpie / act4 / Magpie4.java Find file Copy path

victoryforphil Mag Pie Done 0313d3a 2 minutes ago

1 contributor

265 lines (234 sloc) 6.58 KB

```
1  /**
2   * A program to carry on conversations with a human user.
3   * This version:
4   *<ul><li>
5   *             Uses advanced search for keywords
6   *</li><li>
7   *             Will transform statements as well as react to keywords
8   *</li></ul>
9   * @author Laurie White
10  * @version April 2012
11  *
12  */
13 public class Magpie4
14 {
15        /**
16         * Get a default greeting
17         * @return a greeting
18         */
19        public String getGreeting()
20        {
21                return "Hello, let's talk.";
22        }
23
24        /**
25         * Gives a response to a user statement
26         *
27         * @param statement
28         *            the user statement
29         * @return a response based on the rules given
30         */
31        public String getResponse(String statement)
32        {
33                String response = "";
34                if (statement.length() == 0)
35                {
36                        response = "Say something, please.";
37                }
38
39                else if (findKeyword(statement, "no") >= 0)
40                {
41                        response = "Why so negative?";
42                }
43                else if (findKeyword(statement, "mother") >= 0
44                                || findKeyword(statement, "father") >= 0
45                                || findKeyword(statement, "sister") >= 0
46                                || findKeyword(statement, "brother") >= 0)
47                {
48                        response = "Tell me more about your family.";
49                }
50
51                // Responses which require transformations
52                else if (findKeyword(statement, "I want to", 0) >= 0)
53                {
54                        response = transformIWantToStatement(statement);
55                }
```

```
56              else if (findKeyword(statement, "I want", 0) >= 0)
57              {
58                      response = transformIWantStatement(statement);
59              }
60              else
61              {
62                      // Look for a two word (you <something> me)
63                      // pattern
64                      int psn = findKeyword(statement, "you", 0);
65
66                      if (psn >= 0 && findKeyword(statement, "me", psn) >= 0)
67                      {
68                              response = transformYouMeStatement(statement);
69                      }else if(findKeyword(statement, "I", psn) >= 0){
70                              response = transformYouIStatement(statement);
71                      }
72                      else
73                      {
74                              response = getRandomResponse();
75                      }
76              }
77              return response;
78      }
79
80      private String transformIWantToStatement(String statement)
81      {
82              //  Remove the final period, if there is one
83              statement = statement.trim();
84              String lastChar = statement.substring(statement
85                              .length() - 1);
86              if (lastChar.equals("."))
87              {
88                      statement = statement.substring(0, statement
89                                      .length() - 1);
90              }
91              int psn = findKeyword (statement, "I want to", 0);
92              String restOfStatement = statement.substring(psn + 9).trim();
93              return "What would it mean to " + restOfStatement + "?";
94      }
95
96      private String transformIWantStatement(String statement)
97      {
98              //  Remove the final period, if there is one
99              statement = statement.trim();
100             String lastChar = statement.substring(statement
101                             .length() - 1);
102             if (lastChar.equals("."))
103             {
104                     statement = statement.substring(0, statement
105                                     .length() - 1);
106             }
107             int psn = findKeyword (statement, "I want to", 0);
108             String restOfStatement = statement.substring(psn + 9).trim();
109             return "Would you be really happy if you had " + restOfStatement + "?";
110     }
111
112
113     private String transformYouMeStatement(String statement)
114     {
115             //  Remove the final period, if there is one
116             statement = statement.trim();
117             String lastChar = statement.substring(statement
118                             .length() - 1);
119             if (lastChar.equals("."))
120             {
121                     statement = statement.substring(0, statement
122                                     .length() - 1);
```

```
123             }
124
125             int psnOfYou = findKeyword (statement, "you", 0);
126             int psnOfMe = findKeyword (statement, "me", psnOfYou + 3);
127
128             String restOfStatement = statement.substring(psnOfYou + 3, psnOfMe).trim();
129             return "What makes you think that I " + restOfStatement + " you?";
130         }
131
132
133         private String transformYouIStatement(String statement)
134         {
135             //  Remove the final period, if there is one
136             statement = statement.trim();
137             String lastChar = statement.substring(statement
138                     .length() - 1);
139             if (lastChar.equals("."))
140             {
141                 statement = statement.substring(0, statement
142                         .length() - 1);
143             }
144
145             int psnOfYou = findKeyword (statement, "you", 0);
146             int psnOfMe = findKeyword (statement, "me", psnOfYou + 3);
147
148             String restOfStatement = statement.substring(psnOfYou + 3, psnOfMe).trim();
149             return "Why do you" + restOfStatement + " me?";
150         }
151
152
153
154         /**
155          * Search for one word in phrase. The search is not case
156          * sensitive. This method will check that the given goal
157          * is not a substring of a longer string (so, for
158          * example, "I know" does not contain "no").
159          *
160          * @param statement
161          *            the string to search
162          * @param goal
163          *            the string to search for
164          * @param startPos
165          *            the character of the string to begin the
166          *            search at
167          * @return the index of the first occurrence of goal in
168          *         statement or -1 if it's not found
169          */
170         private int findKeyword(String statement, String goal,
171                 int startPos)
172         {
173             String phrase = statement.trim().toLowerCase();
174             goal = goal.toLowerCase();
175
176             // The only change to incorporate the startPos is in
177             // the line below
178             int psn = phrase.indexOf(goal, startPos);
179
180             // Refinement--make sure the goal isn't part of a
181             // word
182             while (psn >= 0)
183             {
184                 // Find the string of length 1 before and after
185                 // the word
186                 String before = " ", after = " ";
187                 if (psn > 0)
188                 {
189                     before = phrase.substring(psn - 1, psn);
```

```
                }
                if (psn + goal.length() < phrase.length())
                {
                        after = phrase.substring(
                                        psn + goal.length(),
                                        psn + goal.length() + 1);
                }

                // If before and after aren't letters, we've
                // found the word
                if (((before.compareTo("a") < 0) || (before
                                .compareTo("z") > 0)) // before is not a
                                                                                        // letter
                                && ((after.compareTo("a") < 0) || (after
                                                .compareTo("z") > 0)))
                {
                        return psn;
                }

                // The last position didn't work, so let's find
                // the next, if there is one.
                psn = phrase.indexOf(goal, psn + 1);

        }

        return -1;
    }

    /**
     * Search for one word in phrase.  The search is not case sensitive.
     * This method will check that the given goal is not a substring of a longer string
     * (so, for example, "I know" does not contain "no").  The search begins at the beginning of the string.
     * @param statement the string to search
     * @param goal the string to search for
     * @return the index of the first occurrence of goal in statement or -1 if it's not found
     */
    private int findKeyword(String statement, String goal)
    {
        return findKeyword (statement, goal, 0);
    }



    /**
     * Pick a default response to use if nothing else fits.
     * @return a non-committal string
     */
    private String getRandomResponse()
    {
        final int NUMBER_OF_RESPONSES = 4;
        double r = Math.random();
        int whichResponse = (int)(r * NUMBER_OF_RESPONSES);
        String response = "";

        if (whichResponse == 0)
        {
                response = "Interesting, tell me more.";
        }
        else if (whichResponse == 1)
        {
                response = "Hmmm.";
        }
        else if (whichResponse == 2)
        {
                response = "Do you really think so?";
        }
        else if (whichResponse == 3)
```

```
257                 {
258                         response = "You don't say.";
259                 }
260 
261                 return response;
262         }
263 
264 }
```