Keenan Kunzelman & Gonzalo Reyes

First the http connection parameters are created through the setURLGrid(int game, int column, int row). This method takes the parameters of the game to be played and the letter to be requested from the server.

```
private static String setURLGrid(int game, int column, int row) {
       HashMap<Integer, Character> charColumn = new HashMap<>();
       charColumn.put(1, 'a');
       charColumn.put(2, 'b');
       charColumn.put(3, 'c');
       charColumn.put(4, 'd');
       charColumn.put(5, 'e');

       return "https://wordfinder-001.appspot.com/wordfinder?game=" + game +
"&pos=" + charColumn.get(column) + row;
   }
```

Next the actual connection and error handling occur through the use fo the getURL(String url) method. This method takes the url created by setURLGrid and feeds it to the code responsible for generating the HTTP request. Then the exceptions of UnknownHost, MalformedURL, and IO are handled with a try, catch block

```
static Character getURL(String url) {
       // Get character
       String inputLine;
       URL urlLook;
       int statusCode = 0;

       try {
           // create new URL object
           urlLook = new URL(url);
           // Open new HTTP connection
           HttpURLConnection connection = (HttpURLConnection)
urlLook.openConnection();
           // Get the response code from connection
           statusCode = connection.getResponseCode();
           // Check status code and perform action
           switch(statusCode) {
               // When status code is 200 == OK
               case 200:
```

```
                    // Create new BufferedReader object to store content
                    BufferedReader in = new BufferedReader(new
InputStreamReader(urlLook.openStream()));
                    // Read content
                    while ((inputLine = in.readLine()) != null) {
                        return inputLine.charAt(0);
                    }
                    // Close buffer reader
                    in.close();
                default:
                    // break when status code is not 200, e.g. 404, 403 or 500
                    break;
            }
        }

        // Handle exceptions
        catch (UnknownHostException u) {
            System.out.print("Unknown host");
        }
        catch (MalformedURLException e) {
            System.out.print("Malformed URL");
        }
        catch (IOException i) {
            System.out.print("IO Exception");
        }
        return null;
    }
```

Then the program makes HTTP request to pull all the valid words to search each game for a match. This method is nearly identical to the method that retrieves the characters from the gameboard.

```
private static ArrayList<String> getWords(String url) {
        // Get character
        String inputLine;
        URL urlLook;
        int statusCode = 0;

        try {
            // create new URL object
            urlLook = new URL(url);
            // Open new HTTP connection
```

```
            HttpURLConnection connection = (HttpURLConnection)
urlLook.openConnection();
            // Get the response code from connection
            statusCode = connection.getResponseCode();
            ArrayList<String> arr = new ArrayList<>();
            // Check status code and perform action
            switch(statusCode) {
                // When status code is 200 == OK
                case 200:
                    // Create new BufferedReader object to store content
                    BufferedReader in = new BufferedReader(new
InputStreamReader(urlLook.openStream()));
                    // Read content
                    while ((inputLine = in.readLine()) != null) {
                        arr.add(inputLine);
                    }
                    // Close buffer reader
                    in.close();
                    return arr;
                default:
                    // break when status code is not 200, e.g. 404, 403 or 500
                    break;
            }
        }
    // Handle exceptions
      catch (UnknownHostException u) {
         System.out.print("Unknown host");
      }
      catch (MalformedURLException e) {
         System.out.print("Malformed URL");
      }
      catch (IOException i) {
         System.out.print("IO Exception");
      }
      return null;
    }
```

Lastly the findWord method is where the logic for finding the word is executed. We created a hashmap to map integer values to alphabetic column values. This ensures quick look up when necessary. Next we use a quadratic algorithm to loop through all of

the columns and rows as chunks or words. So instead of having to loop through every letter we only loop through every row and column as a whole and check for the possible work in the row or column. Lastly if a match is made we output the start and end position of the word in the grid.

```
private static void findWord(int game) {
	ArrayList<String> wordsToSearchFor =
getWords("https://wordfinder-001.appspot.com/word.txt");
	ArrayList<String> wordsToSearchFrom = setGrid(game);

	HashMap<Integer, Character> alphaNumPosition = new HashMap<>();
	alphaNumPosition.put(0, 'A');
	alphaNumPosition.put(1, 'B');
	alphaNumPosition.put(2, 'C');
	alphaNumPosition.put(3, 'D');
	alphaNumPosition.put(4, 'E');

	for (int i = 0; i < wordsToSearchFor.size(); i++) {
	    for (int j = 0; j < wordsToSearchFrom.size(); j++) {
	        if (wordsToSearchFrom.get(j).contains(wordsToSearchFor.get(i)))
{

	            int idx =
wordsToSearchFrom.get(j).indexOf(wordsToSearchFor.get(i));
	            /*if (j > 4) {
	                System.out.println("game: " + game + " word: " +
wordsToSearchFor.get(i) + " location: " + alphaNumPosition.get(j - 5) + idx + "
: " + alphaNumPosition.get(j - 5) + (idx + 2));
	            }
	            else {
	                System.out.println("game: " + game + " word: " +
wordsToSearchFor.get(i) + " location: " + alphaNumPosition.get(idx) + j + " : "
+ alphaNumPosition.get(idx + 2) + j);
	            }*/
	            if (j <= 4) {
	                System.out.println("game: " + game + " word: " +
wordsToSearchFor.get(i) + " location: " + alphaNumPosition.get(j) + (idx+ 1) +
" : " + alphaNumPosition.get(j) + (idx + 3));
	            }
	            else {
	                System.out.println("game: " + game + " word: " +
wordsToSearchFor.get(i) + " location: " + alphaNumPosition.get(idx) + (j - 4) +
" : " + alphaNumPosition.get(idx + 2) + (j - 4));
	            }
```

```
            }
        }
    }
}
```