Mad Libs

1 Overview

A Mad Lib is a game where you use a template to tell a story. The game works by having someone fill words into a story template with minimal context or information about the story. The Mad Lib might ask the player for an adjective or a proper name, with no hint of how the word will become part of the story. The player must decide on all the words before the story is read. The fun and humor of the game is discovering how those words are used in the story.

2 Learning Outcomes

By the end of this project students should be able to:

- write, save, and evaluate simple programs;
- read and write programs with string literals;
- read and write programs with simple function calls (e.g., input, print);
- break up simple problems into multiple steps;
- work effectively with a partner using pair-programming;
- write an effective report that describes the students' problem solving process.

3 Pre-Lab Instructions

Do this part before you come to lab:

While the Mad Lib game should be written in the lab with your partner, there are a few things you should do to prepare before coming to the lab:

- If you have never played a Mad Lib before, try one before class:
  `http://www.eduplace.com/tales/`
- Come up with your own Mad Lib and bring it to lab. Be prepared to show this to the lab aide at the beginning of lab.

4 Lab Instructions

Do this part in lab:

Step 1. Discuss your Mad Lib from your pre-lab with your partner. What are the important features? What makes it fun?

Step 2. Use a sheet of paper to design a new Mad Lib with your partner. This should look like a Mad Lib you might play in a book. This template is your plan for this project.

Step 3. Create a file called "MadLib.java" and convert your Mad Lib into Java code. There's really only three things you need to know:

1. how to get user input and store it in a variable,
2. how to concatenate (join) strings, and
3. how to print strings.

Here's an example that illustrates these processes:

```
import java.util.Scanner;
class MadLib {
    public static void main(String[] args){
        Scanner scanner = new Scanner(System.in);
        System.out.print("Type␣an␣adjective:␣");
        String first_adjective = scanner.next();
        System.out.print("Type␣a␣plural␣noun:␣");
        String first_noun = scanner.next();
        System.out.print("Type␣a␣friend's␣name:␣");
        String first_name = scanner.next();

        System.out.println("My␣First␣Mad␣Lib␣Story");
        System.out.println("=========================");
        System.out.println("There␣once␣was␣a␣" + first_adjective + "␣boy␣" +
             "␣who␣liked␣" + first_noun + ".");
        System.out.println("His␣best␣friend␣was␣named␣" + first_name);
    }
}
```

Your Mad Lib should be a lot longer. You should have at least 12 questions and 10 lines to your story. But feel free to add more and most importantly be creative!

Step 5. Run the application to make sure your Mad Lib works. If it doesn't, see if you can figure out how to fix it. Your partner and the lab staff are here to help.

Step 6. Take a screenshot of your window. In Microsoft Windows you can do this by selecting the window and press ALT and Print Screen.

5 Lab Report

Each pair of students will write a single lab report together and each student will turn in that same lab report on BBLearn. Submissions from each student on a pair should be identical.

Your lab report should begin with a preamble that contains:

- The lab assignment number and name
- Your name(s)
- The date
- The lab section

It should then be followed by four numbered sections:

1. Problem Statement

In this section you should describe the problem in **your** own words. The problem statement should answer questions like:

- What are the important features of the problem?
- What are the problem requirements?

This section should also include a reasonably complete list of requirements in the assignment. Following your description of the problem, include a bulleted list of specific features to implement. If there are any specific funtions, classes or numeric requirements given to you, they should be represented in this bulleted list.

2. Planning

In the second section you should describe what planning you did in order to solve the problem. You should include planning artifacts like sketches, diagrams, or pseudocode you may have used. You should also describe your planning process. List the specific data structures or techniques you plan on using, and why.

3. Implementation and Testing

In the third section you should describe how you implemented your plan. As directed by the lab instructor you should (as appropriate) include:

- a copy of your source code (Submitted in BBLearn as a .java files)
- a screen shot of your running application / solution
- results from testing

4. Reflection

In the last section you should reflect on the project. Consider different things you could have done to make your solution better. This might include code organization improvements, design improvements, etc.

You should also ask yourself what were the key insights or features of your solution? Were there alternative approaches or techniques you could have employed? How would these alternatives have impacted a different solution?

5. Partner Rating

Every assignment you are required to rate your partner with a score -1, 0 or +1. This should be submitted in the comment section of the BBLearn submission, and not in the report document. If you don't want to give your partner a negative rating making sure not to use a dash before listing the number! You do not have to tell your partner the rating you assign them. A rating of 1 indicates that your partner was particularly helpful or contributed exceptional effort. A rating of 0 indcates that your partner met the class expectations of them. Rating your partner at -1 means that they refused contribute to the project, failed to put in a resonable effort or actively blocked you from participating. If a student recieves three ratings of -1 they must attend a mandatory meeting with the instructor to dicuss the situation, and recieving additional -1 ratings beyond that, the student risks losing a letter grade, or even failing the course.

Colophon

This project was developed by Richard Lester and Dr. James Dean Palmer of Northern Arizona University. Except as otherwise noted, the content of this document is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License.

MadLibs rubric (40pts)

Report	10 pts
Code compiles and runs	10 pts
Correct use of Scanner	5 pts
Correct use of System.out	5 pts
Observance of 12 questions and 10 lines	5 pts
Free	5 pts