Instructional GUI Projects (for level I and II CS)

By D. Poe, 2017

Source files for these projects may be found in the accompanying folder.

Word Cloud (interactive display for level 2 CS students)

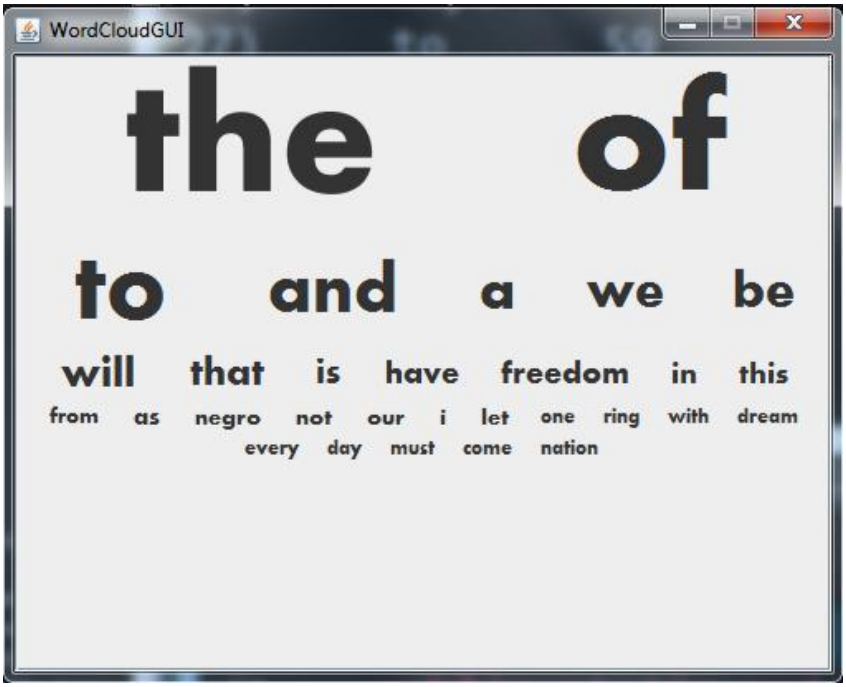
The file 'WordCloudGUI.java' can be downloaded and placed into the source folder along with the WordCloud, Word, and Runner classes. It requires no external files or classes.

Becuase the *main()* method is in the Runner class and not in the WordCloudGUI, the students must call the static *run()* method in the WordCloudGUI class in order to run the GUI. The *run()* method takes a single parameter, the ArrayList of words to be used in the Word Cloud (which can be obtained by calling the *getTopHits()* method in the WordCloud class). Below is a suggested setup for a runner class:

```
public class Runner {
    public static void main(String[] args) throws Exception{
        WordCloud wc = new WordCloud("dream.txt"); //load and initialize Word cloud
        wc.printTop30(); //print top 30 words, as instructed in the lab
        WordCloudGUI.run(wc.getTopHits()); //run the GUI
    }
}
```

*Note that the run() method is static, so students should refrain from creating instances of the WordCloudGUI class.

The program uses a dynamic GUI that runs through the list of words and assigns each word a font size proportional to its frequency. The final output should look something like this:



Adventure Time (interactive display for level 2 CS students)

The file 'AdventureTimeGUI.java' can be downloaded and placed into the source folder along with the Room, Command, Game, and Runner classes.

The Adventure Time GUI was originally composed of 3 different classes; however I condensed them into a single class for simplicity. It functions as requested in the instruction document, running on top of the other logic and taking input from the console. This GUI requires 2 minor changes to the starter code of Adventure Time. First, two lines must be added to the Game class in the play() method. An AdventureTimeGUI object should be declared and initialized outside of the loop. Inside the loop, the *updateUI()* method must be called on the AdventureTimeGUI object in order to refresh the GUI each time the loop runs. This method should be supplied with the current room that the player is in. It is important that this is placed at the first line of the loop, as placing it at the end causes the program to lag in certain situations.

```
public void play() {
    AdventureTimeGUI atg = new AdventureTimeGUI(); //Declare and initialize a GUI object
    boolean finished = false;
    while (!finished) {
        atg.updateUI(player.getCurrentRoom()); //This must be the first line in the loop
        //All of the other logic goes here
    }
}
```

Because Adventure Time is an open-ended lab after the base code is written, I created an open-ended GUI. I avoided hard-coding a text field to display items because customizing the game could involve adding sprites, characters, and keys to the rooms. Instead, students have the option to add custom text to each room. This can be done using the *getRoomMeta()* method, which should be added to the Room class. I realized that this method would only be added after the game is customized, so the GUI only calls this method when it is present. **If the Room class does not contain this method, the GUI will still work**, however the 'other info' section will be hidden. Once the student begins customizing each room, he or she can add the *getRoomMeta()* method in the Room class. The method should return a single string that contains all of the room's information. When the GUI is run, the 'Other Info' section will appear and display the string that the *getRoomMeta()* method returns for the current room.

```
public String getRoomMeta(){
    //return a single string containing further details about the room
    return "{your custom text here}";
}
```

*The string that this method returns should be formatted, if needed. For example, if a student wants to display a list of items, this method should return "Items: \n Item 1 \n Item 2 \n Item 3".

Game of Pig (demo for level 1 CS students)

The Game of Pig is contained in its own file and can be run by launching 'GameOfPig.jar'. The UI includes buttons, images, and 3D animations. There is an option for 2-Player mode and an option to play against the AI. The AI uses the aggressive-rolling strategy, rolling more frequently when its opponent's score reaches a certain threshold. Using the GUI is largely self-explanatory; however it is important to note that the pig icon with sunglasses indicates the player currently taking a turn. The roll/hold buttons of the player who is not taking a turn are temporarily disabled. Below are screenshots of the game: