**Assignment 4**

**Due, Sunday, June 25, 2017 for maximum 100**

**Monday, June 26, 2017 for maximum 90**

**Tuesday, June 27, 2017 for maximum 80**

**Wednesday, June 28, 2017 for maximum 70**

**Assignment Scope**

1. Adding member variables or instance variables to a class
2. Using Java API classes from javax.swing.\* package
3. Developing a User Interface

**Deliverables**

To complete this assignment you must submit your **compressed Netbeans project** to Webcourses.

**Tasks and Rubric**

|  |  |  |
| --- | --- | --- |
| Activity | | |
| Boggle.java | 1. Add to method **main()**    1. Instantiate an instance of class **BoggleUi**, passing the reference object of class **Board** as an argument |
| userInterface package | 1. Create class **BoggleUi** |
| BoggleUi.java | 1. Add member variables of type    1. JFrame    2. JMenuBar    3. JMenu    4. JMenuItem    5. JMenuItem    6. JPanel    7. JButton[][]    8. JPanel    9. JScrollPane    10. JTextPane    11. JLabel    12. JButton    13. JPanel    14. JLabel    15. JButton    16. JLabel 2. A custom constructor should be defined that receives a parameter of type Board class    1. Set member variable of type class **Board** to the parameter passed in    2. call method **initComponents()** 3. A method **initComponents()** should initialize all the components for the UI and called from the constructor    1. Set the default size of the **JFrame**    2. Set the default close operation of the **JFrame**    3. Use default layout manager **BorderLayout**    4. Set up the **JMenuBar**       1. **JMenu** Boggle should be added to the **JMenuBar**       2. **JMenuItems** New Game and Exit should be added to the **JMenu** Boggle    5. A **JPanel** should be created to hold the current word being created by the user       1. Recommend using default Layout Manager **FlowLayout**       2. A **JLabel** for the current word being created       3. A **JButton** to submit the current word created       4. A **JLabel** for the player’s score    6. A **JPanel** should be created to hold the **JButtons** of dice       1. The **JPanel** for the Boggle board should be a 4 x 4 grid that displays a **JButton** for each die       2. Recommend using Layout Manager **GridLayout**    7. A **JPanel** should be created to hold       1. Recommend using Layout Manager **BoxLayout**       2. the **JTextArea** for the user to enter their words       3. the **JScrollPane** for the **JTextArea** to scroll as necessary       4. the **JLabel** that displays time left for the current round of play       5. the **JButton** to shake the dice    8. **JMenuBar** should be set on the **JFrame**    9. Add all the components to the **JFrame**    10. Set the visibility of the **JFrame** |
| Boggle application |  |
| Test Case 1 | Test Case 1 passes |
| Test Case 2 | Test Case 2 passes |
| Test Case 3 | Test Case 3 passes |
|  | Source compiles with no errors |
|  | Source runs with no errors |
|  | Source includes comments |
| Total |  |

**Perform the following test cases**

|  |  |  |
| --- | --- | --- |
| Test Cases | | |
|  | **Action** | **Expected outcome** |
| Test Case 1 | **User interface displays** | User interface is similar to figure 1 |
| Test Case 2 | **Boggle menu** | Boggle menu looks like figure 2 |
| Test case 3 | **Project view** | Project view matches figure 3 |

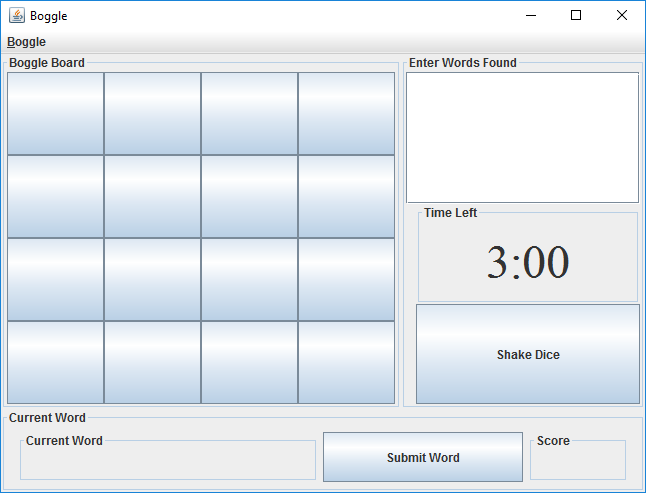


Figure 1 Test Case 1

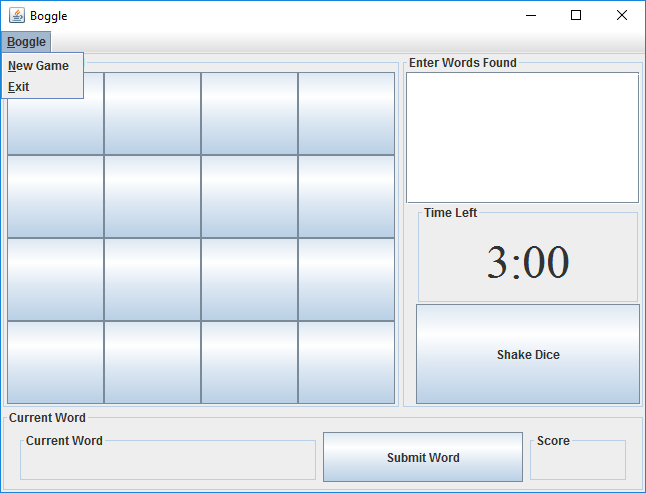


Figure 2 Menu Boggle

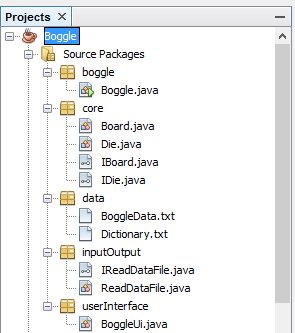


Figure 3 Project View