Guided Project - Iteration #4

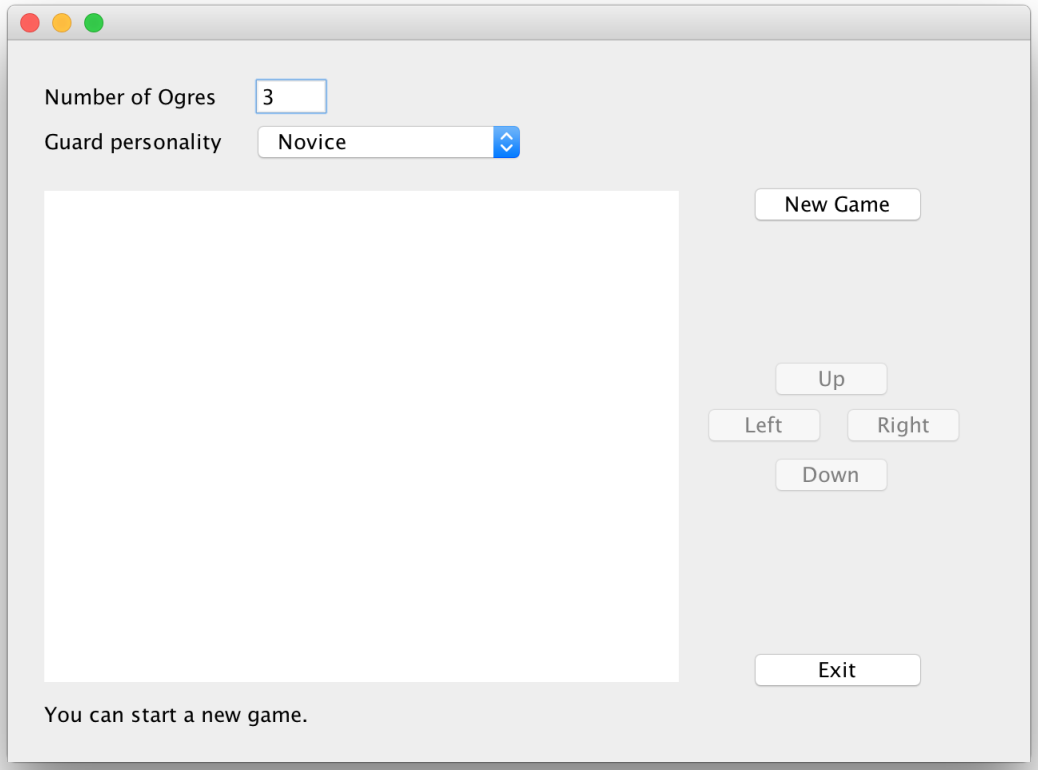
Iteration #4

A. Graphical User Interface (part I)

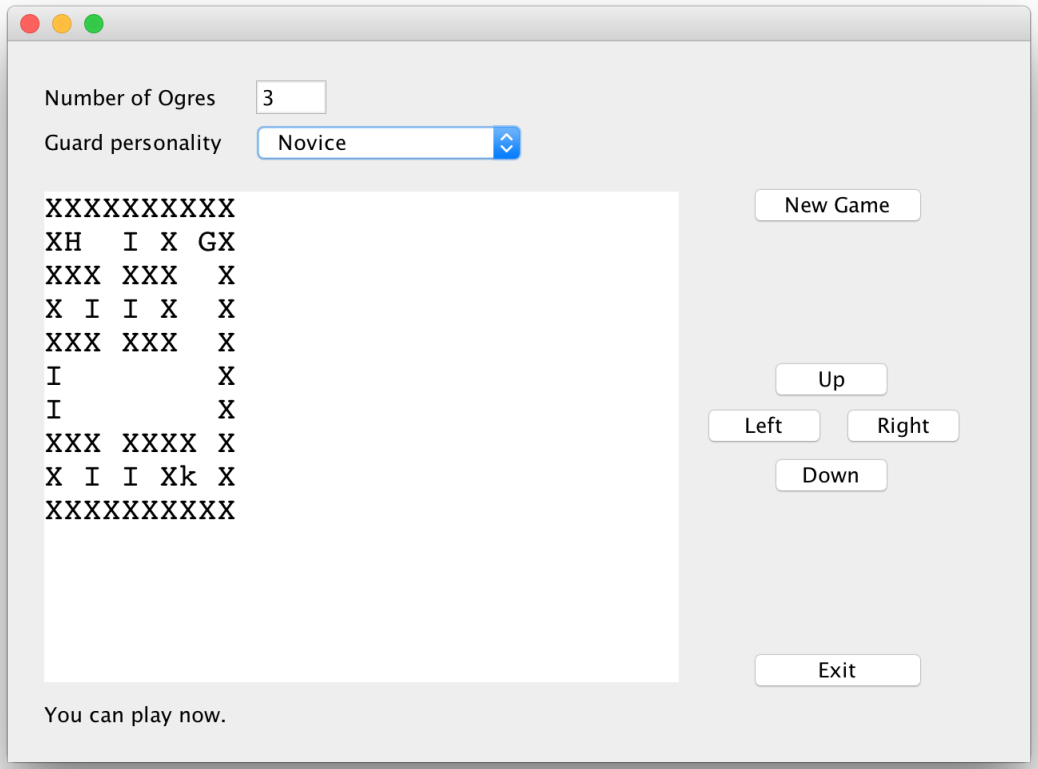
Tired of playing the game on Console? Now its time to give it a Graphical User Interface (GUI), using JAVA SWING. All GUI source code should be placed in a separate package called **dkeep.gui**. Despite this, It should be possible to still run the game on console mode (choosing the main program class at the dkeep.cli package). In this iteration, the GUI will be very simple, having the game shown in a text area and the hero movement issued by buttons (not keys yet).

- **Task #1 [at class].** Using WindowBuilder plug-in, create a top-level window (*Application Window*), containing the following components (or others that provide the same functionality):
  - Text box (*JLabel + JTextField*) for entering number of Ogres on the Keep level.
  - Drop-down box (*JLabel + JComboBox*) to select the Guard's personality.
  - Button (*JButton)* for starting a new game.
  - Button (*JButton*) for exiting the game and closing the window (ending the program).
  - Non-editable text area (*JTextArea*) to shown the current game status (like the Console), configured with a proportional *font,* such as *Courier New.*
  - Buttons (*JButton*) for moving the hero (initially disabled).
  - Label (*JLabel*) to indicate the game status and/or messages to the player.

Below, you can see an example (merely indicative) (MAC look-and-feel).



- **Task #2 [at class].** At WindowBuilder Design "tab", double-click on the "New Game" button, and add the respective code to the *listener's* method body. The *listener* should create a new game according to the configuration values (number of Ogres and Guard's personality) entered and selected on the respective interface elements. It should show the game on the text area (*setText*), enable the hero's movement buttons (*setEnable(true)*), update the game status label. Test it, by trying multiple times to create a new game. Below is a, merely indicative, example:



- **Task #3 [at class].** Create a *listener* for the "Exit" button. The *System.exit(0)* instruction should suffice.
- **Task #4 [at class].** Create *listeners* for the hero's movement buttons. Besides moving the hero and updating the game status, it should also update the status label with the game state accordingly. Try not to duplicate code. If the game ends (win or loose), not only should the status label de updated, but the movement buttons disabled (*setEnabled(false)*).
- **Task #5 [at class or at home].** Make your GUI more robust, by handling invalid entry values on the number of Ogres (must be a positive integer, no greater than 5).

Última alteração: Segunda, 6 Março 2017, 12:14

ADMINISTRAÇÃO DA PÁGINA/UNIDADE

Nome de utilizador: Pedro Miguel Sousa Fernandes (Sair)