Application User Interfaces

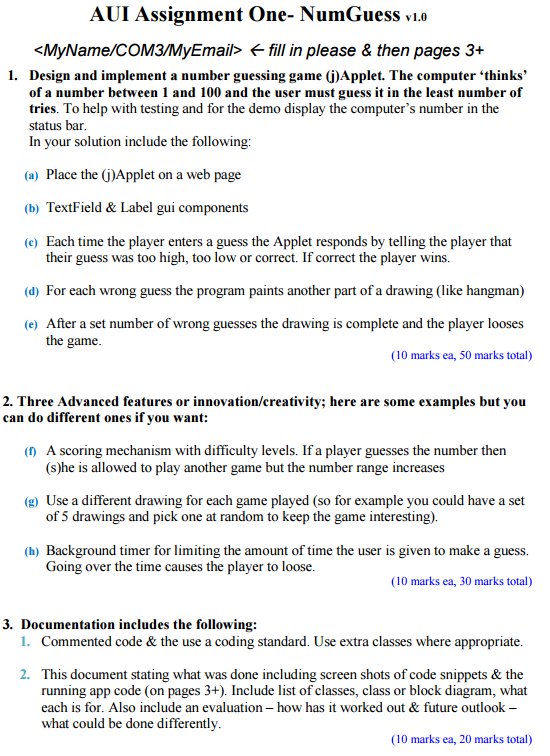
Assignment 1

Design and implementation of a number guessing game (j)Applet.

**Written by:** John Byrne – R00050076

**Due by:**  23:59 on Mon 13th April 2015

# Assignment Requirements



**DONE**

**DONE**

**DONE**

**DONE**

**DONE**

**DONE**

**DONE**

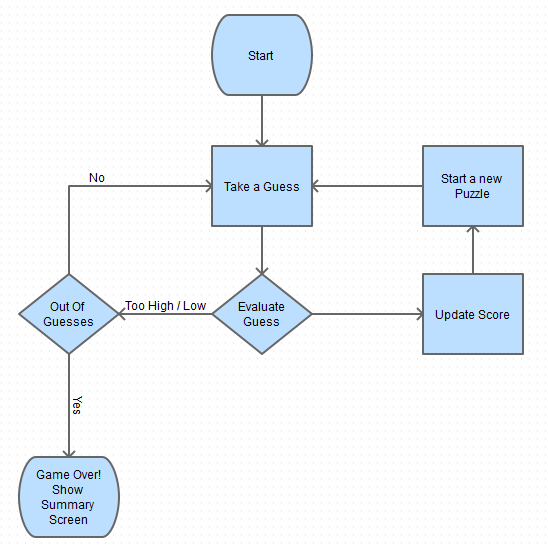
**DONE**

**DONE**

**DONE**

# List of Classes and Block Diagram

Class: NumGuess.class which contains two ActionListener Inner Classes.



# Source Code

import javax.swing.\*;

import java.applet.Applet;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.Random;

public class NumGuess extends Applet {

/\*\*

\* NumGuess Applet - by John Byrne R00050076

\*

\* RUN IN APPLET SIZE OF 600 x 400 PIXELS!!!!

\*

\* Applet Pics from

\* https://www.cs.colostate.edu/~cs192/Labs/AppletExamples.html

\*

\* Simple Instructions:

\* The game starts at Difficulty Level 1 and you are asked to guess a number between 1 and 100.

\* As you solve each puzzle, your score is calculated and you start the next puzzle with an

\* increased difficulty, where you will be asked to guess a number from a larger range.

\* The score is based on the number of unused guesses when the puzzle is solved and each unused

\* guess is worth more as the difficulty increases.

\*

\* For your entertainment, there are 4 Pictures that are built-up as you make wrong guesses.

\* The pictures are picked at random at the start of each puzzle, so you might get the same

\* picture twice in a row!

\*

\* Oh, and just so I'm not waiting all day, you have 10 seconds to enter a guess or you'll

\* lose a try!!

\*

\*/

////My CONSTANTS

final Color PINK4 = new Color ( 255, 128, 128 ) ;

final Color GREEN2 = new Color ( 0, 208, 0 ) ;

final Color LTBLUE = new Color ( 35, 206, 255 ) ;

final Color REDBROWN = new Color ( 182, 100, 110 ) ;

final Color DARKBROWN = new Color ( 150, 70, 80 ) ;

final Color DARKGOLD = new Color ( 240, 220, 0 ) ;

final int FIGUREHEIGHT = getHeight();

final int FIGUREWIDTH = getWidth();

final int DELTA\_Y = 50;

////My Global Variables

int difficultyLevel; //The difficulty of the puzzle

int upperLimit; //The Upper-limit of the rang of numbers that I can pick from.

int picChoice; //The chosen picture I'm going to use for this puzzle.

Label lblDifficultyLevel; //A label to tell the user what Difficulty Level they are on.

Label lblInput; //A label for the User Input Text Box

TextField txtInput; //The aforementioned, User Input Text Box

Random random; //'An instance of Random.' Sounds like a James Bond movie!

int randomNumber; //The Random Number I've chosen for the puzzle

int userGuess; //The user's guess

int guessesTaken; //How many guesses the user has used in this puzzle

int timeRemaining; //Time remaining per guess

Timer timer; //A Timer object!!!

boolean playerHasLost; //Has the user lost the game yet?

int score; //The user's score

/\*\*NumGuess Constructor

\* This is the same as init(), but I prefer to treat it as a Constructor

\* It initialises all my global variables.

\*/

public NumGuess() {

difficultyLevel = 0;

timeRemaining = 10;

playerHasLost = false;

score = 0;

lblDifficultyLevel = new Label();

lblInput = new Label();

txtInput = new TextField(10);

random = new Random();

startNewPuzzle();

add(lblDifficultyLevel);

add(lblInput);

add(txtInput);

txtInput.addActionListener(new TextInputListener());

timer = new Timer(1000, new TimerListener());

timer.start();

}

/\*\*

\* startNewPuzzle

\* This contains the variables that require re-initializing when a new puzzle starts.

\*/

public void startNewPuzzle() {

difficultyLevel++;

upperLimit = difficultyLevel \* 100; //Sets the Upper-Limit of the range from which I pick my random number.

randomNumber = random.nextInt(upperLimit) + 1; //Pick my Random number

picChoice = random.nextInt(4) + 1; //Randomly pick one of the 4 pictures

//Setup the Label and Text Boxes for this puzzle...

lblDifficultyLevel.setText("Difficulty Level: " + difficultyLevel + " !");

lblInput.setText("I'm thinking of a number between 1 and " + upperLimit + "! Take a Guess...");

//Yes, both of these are required to clear a TextBox due to a Java Applet Bug!

txtInput.setText(" "); //clear data entry field

txtInput.setText(""); //clear data entry field

guessesTaken = 0; //At the start of a puzzle, the number of guesses taken will be ZERO!

if (timer != null) { //If this isn't the first level, the Timer object will already exist.

timer.restart();

}

}

/\*\*

\* paint

\*

\* @param graphics Instance of the Graphics object

\*/

public void paint( Graphics graphics )

{

if(!playerHasLost) { //Is te user still in the game?

updateGame(graphics);

}else{

drawEndGameSummary(graphics);

}

}

/\*\*

\*

\* @param graphics The Original instance of rht Graphics object

\*/

private void updateGame(Graphics graphics) {

if (guessesTaken == 0) { //First attempt at this puzzle

setBackground(Color.WHITE); //Just in case a previous puzzle had a different background.

}

if (picChoice == 1) { //picChoice is randomly picked in the startNewPuzzle method

drawStickman(guessesTaken, graphics);

} else if (picChoice == 2) {

drawSnowman(guessesTaken, graphics);

} else if (picChoice == 3) {

drawCar(guessesTaken, graphics);

} else if (picChoice == 4) {

drawFaceWithHat(guessesTaken, graphics);

}

//Save the original Color, so that I can restore it after I update the remaining time and score

Color originalColor = graphics.getColor();

graphics.setFont(new Font("default", Font.BOLD, 16));

updateTime(graphics);

updateScore(graphics);

graphics.setColor(originalColor);

}

/\*\*

\*

\* @param graphics The Original instance of rht Graphics object

\*/

private void drawEndGameSummary(Graphics graphics) {

setBackground(Color.RED);

graphics.setFont(new Font("default", Font.BOLD, 50));

graphics.drawString("You Lose!!!", getWidth() / 10, getHeight() / 5);

graphics.drawString("The Number was " + randomNumber + "!", getWidth() / 10, (getHeight() / 5) \* 2);

graphics.drawString("Your Score was " + score + "!", getWidth() / 10, (getHeight() / 5) \* 3);

if(score < 100){

graphics.drawString("Better luck next time!", getWidth() / 10, (getHeight() / 5) \* 4);

}else if (score < 300){

graphics.drawString("Nice Try!", getWidth() / 10, (getHeight() / 5) \* 4);

}else{

graphics.drawString("You're AWESOME!", getWidth() / 10, (getHeight() / 5) \* 4);

}

lblDifficultyLevel.setVisible(false);

lblInput.setVisible(false);

txtInput.setVisible(false);

showStatus("Goodbye!!");

}

/\*\*

\*

\* @param graphics The Original instance of rht Graphics object

\*/

private void updateScore(Graphics graphics) {

graphics.setColor(Color.GREEN);

graphics.drawString("Score: " + score, getWidth() - 100, getHeight() - 5);

}

/\*\*

\*

\* @param graphics The Original instance of rht Graphics object

\*/

private void updateTime(Graphics graphics) {

if (timeRemaining > 3) {

graphics.setColor(Color.GREEN);

} else {

graphics.setColor(Color.RED);

}

graphics.drawString("Time Remaining: " + timeRemaining, 0, getHeight() - 5);

}

//////////////////////////

//// Action Listeners ////

//////////////////////////

public class TimerListener implements ActionListener{

@Override

public void actionPerformed(ActionEvent e) {

timeRemaining--;

if(timeRemaining < 1){ //User out of time for this guess?

guessesTaken++;

showStatus("You ran out of time!! You have " + (10 - guessesTaken) + " guesses remaining.");

timeRemaining = 10; //Reinitialize the timer

}

if (guessesTaken > 9){ //Check if the user is out of guesses.

playerHasLost = true;

}

repaint(10); // Repaint in 10 ms. (Required to allow the Timer a moment.)

}

}

public class TextInputListener implements ActionListener{

@Override

public void actionPerformed(ActionEvent e) {

guessesTaken++;

try {

userGuess = Integer.parseInt(e.getActionCommand()); //get number

}catch (RuntimeException exception){

//User loses a guess for being a bit silly!

}

if(userGuess < randomNumber){

showStatus("Your guess of " + userGuess + " is too Low. You have " +

(10 - guessesTaken) + " guesses remaining.");

}else if(userGuess > randomNumber){

showStatus("Your guess of " + userGuess + " is too High. You have " +

(10 - guessesTaken) + " guesses remaining.");

}else{ //Well, if it isn't too Low and it isn't too High, then it must be equal, right!!

showStatus("Well done! You are right. My number was " +

randomNumber + " and you got it in " + guessesTaken + " guesses!");

//Calculate the score...

score = score + (10 - guessesTaken) \* (difficultyLevel \* 10);

timer.stop();

startNewPuzzle(); //Start new puzzle, which will be at the next difficulty level.

}

//Reset TextField... Yes, it requires both of these!!!

txtInput.setText(" "); //clear data entry field

txtInput.setText(""); //clear data entry field

timeRemaining = 10; //Reset the timer

if (guessesTaken > 9){ //Check if the user is out of guesses

playerHasLost = true;

}

repaint();

}

}

/////////////////////////

//// Applet Pictures ////

/////////////////////////

private void drawFaceWithHat(int guessesTaken, Graphics graphics) {

if(guessesTaken > 0){ //After 1st Guess

// background for figure

graphics.setColor ( Color.white ) ;

graphics.fillRect ( 0, DELTA\_Y, FIGUREWIDTH, FIGUREHEIGHT ) ;

}

if(guessesTaken > 1) { //After 2nd Guess

// outline of face

graphics.setColor(Color.black);

graphics.drawOval(20, 54 + DELTA\_Y, 166, 166);

}

if(guessesTaken > 2) { //After 3rd Guess

// mouth

graphics.setColor(PINK4);

graphics.fillOval(91, 160 + DELTA\_Y, 24, 24);

}

if(guessesTaken > 3) { //After 4th Guess

// two eyes

graphics.setColor(LTBLUE);

graphics.fillOval(66, 108 + DELTA\_Y, 16, 16);

}

if(guessesTaken > 4) { //After 5th Guess

graphics.setColor(LTBLUE);

graphics.fillOval(124, 108 + DELTA\_Y, 16, 16);

}

if(guessesTaken > 5) { //After 6th Guess

// hat

graphics.setColor(GREEN2);

graphics.fillRect(20, 35 + DELTA\_Y, 166, 42);

}

if(guessesTaken > 6) { //After 7th Guess

graphics.setColor(GREEN2);

graphics.fillRect(62, 15 + DELTA\_Y, 83, 22);

}

if(guessesTaken > 7) { //After 8th Guess

// nose

graphics.setColor(Color.black);

graphics.drawLine(103, 125 + DELTA\_Y, 103, 151 + DELTA\_Y);

graphics.fillOval(96, 150 + DELTA\_Y, 3, 3);

graphics.fillOval(107, 150 + DELTA\_Y, 3, 3);

}

if(guessesTaken > 8) { //After 9th Guess

// eyebrows

graphics.setColor(Color.black);

graphics.drawLine(74, 100 + DELTA\_Y, 84, 104 + DELTA\_Y);

graphics.drawLine(74, 100 + DELTA\_Y, 59, 110 + DELTA\_Y);

graphics.drawLine(132, 100 + DELTA\_Y, 122, 104 + DELTA\_Y);

graphics.drawLine(132, 100 + DELTA\_Y, 147, 110 + DELTA\_Y);

}

}

private void drawCar(int guessesTaken, Graphics graphics) {

if (guessesTaken > 0) { //After 1st Guess

// background for figure

graphics.setColor(Color.white);

graphics.fillRect(0, DELTA\_Y, FIGUREWIDTH, FIGUREHEIGHT);

graphics.setColor(Color.black);

// front tire

graphics.setColor(Color.black);

graphics.fillOval(20, 170 + DELTA\_Y, 100, 100);

graphics.setColor(Color.white);

graphics.fillOval(30, 180 + DELTA\_Y, 80, 80);

graphics.setColor(Color.black);

graphics.drawOval(40, 190 + DELTA\_Y, 60, 60);

}

if (guessesTaken > 1) { //After 2nd Guess

// back tire

graphics.setColor(Color.black);

graphics.fillOval(270, 170 + DELTA\_Y, 100, 100);

graphics.setColor(Color.white);

graphics.fillOval(280, 180 + DELTA\_Y, 80, 80);

graphics.setColor(Color.black);

graphics.drawOval(290, 190 + DELTA\_Y, 60, 60);

}

if (guessesTaken > 2) { //After 3rd Guess

// car hood

graphics.setColor(DARKBROWN);

graphics.fillRect(10, 113 + DELTA\_Y, 122, 12);

graphics.setColor(REDBROWN);

graphics.fillRect(10, 123 + DELTA\_Y, 122, 82);

}

if (guessesTaken > 3) { //After 4th Guess

// car hood ornament

graphics.setColor(DARKGOLD);

graphics.fillOval(10, 105 + DELTA\_Y, 10, 10);

// car window

graphics.setColor(LTBLUE);

graphics.fillRect(130, 15 + DELTA\_Y, 130, 100);

}

if (guessesTaken > 4) { //After 5th Guess

// car door

graphics.setColor(REDBROWN);

graphics.fillRect(130, 113 + DELTA\_Y, 130, 92);

// car backseat

graphics.setColor(REDBROWN);

graphics.fillRect(258, 15 + DELTA\_Y, 122, 190);

}

if (guessesTaken > 5) { //After 6th Guess

// car trunk

graphics.setColor(REDBROWN);

graphics.fillRect(378, 80 + DELTA\_Y, 57, 125);

}

if (guessesTaken > 6) { //After 7th Guess

// car running board

graphics.setColor(DARKBROWN);

graphics.fillRect(118, 205 + DELTA\_Y, 154, 10);

}

if (guessesTaken > 7) { //After 8th Guess

// visor

graphics.setColor(Color.black);

graphics.drawLine(131, 15 + DELTA\_Y, 110, 30 + DELTA\_Y);

graphics.drawLine(131, 16 + DELTA\_Y, 110, 31 + DELTA\_Y);

graphics.drawLine(131, 17 + DELTA\_Y, 110, 32 + DELTA\_Y);

}

if (guessesTaken > 8) { //After 9th Guess

// door handle

graphics.setColor(Color.black);

graphics.drawLine(145, 125 + DELTA\_Y, 170, 125 + DELTA\_Y);

graphics.drawLine(145, 124 + DELTA\_Y, 170, 124 + DELTA\_Y);

graphics.drawLine(145, 123 + DELTA\_Y, 170, 123 + DELTA\_Y);

}

}

private void drawStickman(int guessesTaken, Graphics graphics) {

if(guessesTaken > 0) { //After 1st Guess

setBackground(Color.black);

graphics.setColor(Color.red);

}

if(guessesTaken > 1) { //After 2nd Guess

// draw a boundary

graphics.drawRect(5, 5 + DELTA\_Y, 190, 190);

}

if(guessesTaken > 2) { //After 3rd Guess

// the head

graphics.drawOval(90, 60 + DELTA\_Y, 20, 20);

}

if(guessesTaken > 3) { //After 4th Guess

// the body

graphics.drawLine(100, 80 + DELTA\_Y, 100, 120 + DELTA\_Y);

}

if(guessesTaken > 4) { //After 5th Guess

// the hands

graphics.drawLine(100, 100 + DELTA\_Y, 80, 100 + DELTA\_Y);

}

if(guessesTaken > 5) { //After 6th Guess

graphics.drawLine(100, 100 + DELTA\_Y, 120, 75 + DELTA\_Y);

}

if(guessesTaken > 6) { //After 7th Guess

// the legs

graphics.drawLine(100, 120 + DELTA\_Y, 85, 135 + DELTA\_Y);

}

if(guessesTaken > 7) { //After 8th Guess

graphics.drawLine(100, 120 + DELTA\_Y, 115, 135 + DELTA\_Y);

}

if(guessesTaken > 8) { //After 9th Guess

// the greeting

graphics.drawString("Last chance!!!", 20, 180 + DELTA\_Y);

}

}

private void drawSnowman(int guessesTaken, Graphics graphics) {

int middle = 150; // middle of the snowman

int top = 50 + DELTA\_Y; // top of the snowman

if(guessesTaken > 0) { //After 1st Guess

setBackground(Color.cyan);

}

if(guessesTaken > 1) { //After 2nd Guess

// color the ground

graphics.setColor(Color.blue);

// the ground is a blue rectangle

graphics.fillRect(1, 175 + DELTA\_Y, 300, 50);

}

if(guessesTaken > 2) { //After 3rd Guess

// draw three large snowballs to make up snowman

graphics.setColor(Color.white);

// draw head

graphics.fillOval(middle - 20, top, 40, 40);

}

if(guessesTaken > 3) { //After 4th Guess

// draw middle (upper torso)

graphics.fillOval(middle - 35, top + 35, 70, 50);

}

if(guessesTaken > 4) { //After 5th Guess

// draw base (lower torso)

graphics.fillOval(middle - 50, top + 80, 100, 60);

}

if(guessesTaken > 5) { //After 6th Guess

// draw in features of snowman

graphics.setColor(Color.black);

// draw eyes

// draw left eye

graphics.fillOval(middle - 10, top + 10, 5, 5);

// draw right eye

graphics.fillOval(middle + 5, top + 10, 5, 5);

}

if(guessesTaken > 6) { //After 7th Guess

// draw mouth

graphics.drawArc(middle - 10, top + 20, 20, 10, 190, 160);

}

if(guessesTaken > 7) { //After 8th Guess

// draw arms

// draw left arm

graphics.drawLine(middle - 25, top + 60, middle - 50, top + 40);

// draw right arm

graphics.drawLine(middle + 25, top + 60, middle + 55, top + 60);

}

if(guessesTaken > 8) { //After 9th Guess

// draw hat

// draw brim of hat

graphics.drawLine(middle - 20, top + 5, middle + 20, top + 5);

// draw top of hat

}

}

}

# Conclusion

Not sure what I can say here to justify 10 marks….

If I was to do it again, I’d probably put more work into the Pictures and in particular how the pictures are started, finished and transition from one puzzle to another.

In terms of what I have learnt…

* I now know how to use Java Applets

This was an enjoyable and fruitful assignment, with valuable experience and knowledge gained.

Thank you. END John Byrne – R00050076