/\*

\* HangmanGUI.as

\* Nicole Fella

\* Assignment 7

\*/

package

{

//import statements

import flash.display.MovieClip;

import flash.events.MouseEvent;

/\*

\* description of class

\* "extends" statements would go here

\*/

public class HangmanGUI extends MovieClip

{

/\*\*\*\*\*\*\*\*\*\*Instance Properties\*\*\*\*\*\*\*\*\*\*/

//user defined properties for class

var hangmanGame: HangmanGame;

/\*\*\*\*\*\*\*\*\*\*Instance Methods\*\*\*\*\*\*\*\*\*\*/

//Constructor

public function HangmanGUI()

{

hangmanGame = new HangmanGame();

initGUI();

}

//Additional instance methods go here

/\* initialize user interface elements (e.g., add event listeners, hide win/lose screens)

Make sure current display is up to date (refreshed)

\*/

public function initGUI (): void

{

//add event listener for when guessButton is clicked

guessButton.addEventListener(MouseEvent.CLICK, userGuessed);

//start with invisible winner and loser symbols

win.visible = false;

lose.visible = false;

//invoke refreshDisplay function to set up interface

refreshDisplay ();

}

/\*This function is invoked when the user guesses a letter

(should be added as an event listener in the initGUI function)

If the game isn't over, call the guessLetter function to update "logic"

and refresh display to reflect current state of game

\*/

public function userGuessed (e: MouseEvent): void

{

//when button is clicked, invoke the guessLetter function with string input

guessLetter (letterGuessField.text);

//update currentGuessField

currentGuessField.text = getCurrentGuess ();

//update remainingStrikesField

remainingStrikesField.text = intToString (numberOfRemainingStrikes());

//invoke revealAnswer function to check gameOver, gameWon, and getAnswer

revealAnswer ();

}

/\*refresh under interface (GUI), e.g., text fields to reflect

(1) current guess

(2) number of strikes

\*/

public function refreshDisplay (): void

{

//currentGuessField

currentGuessField.text = getCurrentGuess();

//remainingStrikesField

remainingStrikesField.text = intToString (numberOfRemainingStrikes());

}

/\*update GUI to reveal the correct answer\*/

public function revealAnswer (): void

{

//check if game over

if (gameOver() == true){

//if game over, get answer -- trace to OUTPUT window and answerField

trace ("The word was: " + getAnswer ());

answerField.text = "The word was: " + getAnswer ();

}

//check if game won

if (gameWon() == true){

//if game won, display game win screen

win.visible = true;

//get answer -- trace to OUTPUT window and answerField

trace ("The word was: " + getAnswer());

answerField.text = "The word was: " + getAnswer ();

}

//if game over and game lost

if (gameOver()==true && gameWon()==false){

//display game lose screen

lose.visible = true;

//get answer -- trace to OUTPUT window and answerField

trace("The word was: " + getAnswer ());

answerField.text = "The word was: " + getAnswer ();

}

}

//function intToString to be used for remainingStrikesField text

public function intToString( i : int ) : String

{

// You should not understand this right now!

return i.toString();

}

}

}

/\*

\* HangmanGame.as

\* Nicole Fella

\* Assignment 7

\*/

package

{

//import statements

/\*

\* description of class

\* "extends" statements would go here

\*/

public class HangmanGame

{

/\*\*\*\*\*\*\*\*\*\*Instance Properties\*\*\*\*\*\*\*\*\*\*/

//user defined properties for class

private var NUMBER\_STRIKES: int = 7; //allowed number of strikes

private var currentWordArray: Array; //each index contains single letter of hidden word

private var currentGuessArray: Array; //index I is true if user has correctly guessed letter at that index

private var strikes: int; //holds current number of strikes against the player

private var dictionary: Dictionary;

private var arrayFunctions: ArrayFunctions;

/\*\*\*\*\*\*\*\*\*\*Instance Methods\*\*\*\*\*\*\*\*\*\*/

//Constructor

public function HangmanGame()

{

dictionary = new Dictionary();

arrayFunctions = new ArrayFunctions();

initGame();

}

//Additional instance methods go here

/\*this funciton intializes a game of "logic"

A word should be randomly chosen from dictionary

Variables should be assigned accordingly\*/

public function initGame (): void

{

//define new Array for currentWordArray

currentWordArray = new Array();

//assign Array of randomWord String --> changed with string2Array function

currentWordArray = string2Array(randomWord ());

//invokde initGuess function, to initialize currentWordArray

initGuess ();

}

/\*this "helper" function creates a new Array for currentGuess variable and initializes all slots to be false

Think about the for-loops we have been doing in class about initialization of Arrays\*/

public function initGuess (): void

{

//length of the initGuess will be a long as length of currentWordArray

//give currentGuessArray the legnth of currentWordArray

currentGuessArray = new Array (currentWordArray.length);

//use for-loop to initialize all slots to be false

for (var i: int = 0; i<currentWordArray.length; i++){

//initialize all slots to be false

currentGuessArray[i] = 0;

}

//invoke getCurrentGuess function

getCurrentGuess ();

}

/\*this funciton will update currentGuess variable by checking currentWord variable

if the letter is not found, number of strikes should be updated\*/

public function guessLetter (p: String): void

{

//use for-loop to check each value of currentWordArray

for (var i:int=0; i<currentWordArray.length; i++){

//check if parameter p is found in currentWordArray

if (p == currentWordArray[i]){

//if true, replace currentGuess[i] with letter from parameter p

currentGuessArray[i] = p;

}

}

}

/\*this function should return String "?" for unrevealed letters

this function will "walk over" currentWord and currentGuess arrays and create strings accordingly

if false at index i in currentGuess, then ? should appear

if true at index i in currentGuess, then value in ith slot in currentWord should appear

\*/

public function getCurrentGuess (): String

{

//define currentGuessString

var currentGuessString: String = "";

//use for-loop to "walk over" currentGuessArray

for (var i:int=0; i<currentGuessArray.length; i++){

//if false at index i in curretnGuess, then ? should appear

if (currentGuessArray[i] == 0){

//add ? in place for that character spot

currentGuessString = currentGuessString + "?";

}

//if true at index i in currentGuess, then value in ith slot of currentWord should appear

else{

//add value in ith slot of currentWord

currentGuessString = currentGuessString + currentWordArray[i];

}

}

//if currentGuessField.text (the last guessed word) is equal to the currentGuessString

if (currentGuessField.text == currentGuessString){

//invoke numberOfStrikes function to update number of strikes

numberOfStrikes();

}

//trace currentGuessString to OUTPUT window

trace ("Your current Guess is: " + currentGuessString);

//return currentGuessString to function

return currentGuessString;

}

/\*returns number of strikes against player thus far

strikes variable will be updated each time this function in invoked

\*/

public function numberOfStrikes (): int

{

//update strikes variable --> add 1 to strikes variable

strikes = strikes+1;

//return number of strikes against player so far

return strikes;

}

/\*this function will return the reaming number of strikes a player has left

Subtract current number of Strikes from NUMBER\_STRIKES variable\*/

public function numberOfRemainingStrikes (): int

{

//define variable remaningStrikes

var remainingStrikes: int;

//remainingStrikes is equal to NUMBER\_STRIKES minus strikes

remainingStrikes = NUMBER\_STRIKES - strikes;

//return remainingStrikes variable

return remainingStrikes;

}

/\*this function will return the hidden word chosen from original dictionary Arrary\*/

public function getAnswer (): String

{

//change currentWordArray into a string which is returnable to function

return array2String(currentWordArray);

}

/\*this function will return true whether the game is won or lost

The game must only be over to return true\*/

public function gameOver (): Boolean

{

//true if remaining strikes equals zero

if (NUMBER\_STRIKES == strikes){

return true;

}

//if not, then false

else{

return false;

}

//true if currentGuess is same as currentWord

if (array2String(currentGuessArray) == array2String(currentWordArray)){

return true;

}

//if not, then false

else{

return false;

}

}

/\*this function will return true if player won game (guessed all letters in word)

\*/

public function gameWon (): Boolean

{

//true if currentGuess=currentWord AND number of remaining strikes doesn't equal 0

if ((array2String(currentGuessArray)==array2String(currentWordArray)) && (NUMBER\_STRIKES != strikes)){

return true;

}

//if not, then false

else{

return false;

}

}

}

}

/\*

\* Dictionary.as

\* Nicole Fella

\* Assignment 7

\*/

package

{

//import statements

/\*

\* description of class

\* "extends" statements would go here

\*/

public class Dictionary

{

/\*\*\*\*\*\*\*\*\*\*Instance Properties\*\*\*\*\*\*\*\*\*\*/

//user defined properties for class

//create an array with five words of MY choosing

private var dictionary: Array = ["piano", "beach", "daisy", "library", "travel"];

/\*\*\*\*\*\*\*\*\*\*Instance Methods\*\*\*\*\*\*\*\*\*\*/

//Constructor

public function Dictionary()

{

randomWord();

}

//Additional instance methods go here

/\* this function will return a random word from the dictionary Array

Should use a random number generator, and return array value of dictionary[#]

\*/

public function randomWord(): String

{

//get a random number between 0 and 1

var randNum: Number = Math.random();

//use randomNum to choose a word from dictionary Array -- each choice has equal chance of being chosen

if (randNum < (1/5)){

//return the first value in dictionary Array

return dictionary[0];

}

else if (randNum < (2/5)){

//return second value in dictionary Array

return dictionary[1];

}

else if (randNum < (3/5)){

//return third value in dictionary Array

return dictionary [2];

}

else if (randNum < (4/5)){

//return fourth value in dictionary Array

return dictionary [3];

}

else{

//return fifth (final) value in dictionary Array

return dictionary [4];

}

}

}

}

/\*

\* ArrayFunctions.as

\* Nicole Fella

\* Assignment 7

\*/

package

{

//import statements

/\*

\* description of class

\* "extends" statements would go here

\*/

public class ArrayFunctions

{

/\*\*\*\*\*\*\*\*\*\*Instance Properties\*\*\*\*\*\*\*\*\*\*/

//user defined properties for class

/\*\*\*\*\*\*\*\*\*\*Instance Methods\*\*\*\*\*\*\*\*\*\*/

//Constructor

public function ArrayFunctions()

{

}

//Additional instance methods go here

/\*the parameter of this function is assumed to be an Array of Strings

Should return a String in concatenation of elements of Array (summing up Array strings)

\*/

public function array2String (inputA: Array): String

{

//define String variable concatString, initalize to contain nothing

var concatString: String = "";

//use for-loop to go through each element of Array

for (var i: int = 0; i<inputA.length; i++){

//"sum" up elements of Array into concatString

concatString = concatString + inputA[i];

}

//return concatString

return concatString;

}

/\*create Array that puts each letter of passed parameter into own slot\*/

public function string2Array (inputS: String): Array

{

//define concatArray variable

var concatArray: Array = new Array(inputS.length);

//use for-loop to go through each element of String

for (var i: int = 0; i<inputS.length; i++){

//put letter at certain place in String into spot in Array

concatArray[i] = inputS.substr(i,1);

}

//return concatArray

return concatArray;

}

}

}