IST400/600 Scripting for Games, Spring 2010
Lab 4
Instructor: Keisuke Inoue

Description:

In this lab, you will create the "Scrabble" game that was discussed in the previous class. The following instructions will help you to create the game, but will not tell you all the small steps. The lab is designed to encourage your learning and enhance your understanding by working independently. **You may ask questions to instructors and/or friends (but keep it quiet) and look up previous lecture materials, textbook, and other resources. Do not copy-and-paste any resources other than you created.**

Peer Survey:

In this lab, we will be doing a peer survey instead of the peer evaluation we have been doing previously. You will be in one of the two groups: Group 1 for undergraduate students and Group 2 for the graduate or non-matriculated students. After you finish your lab, review other people's games, and select one person (other than yourself), who you think made the best game. Your lab's evaluation will be done by the instructor (10 points) but you will receive extra 2 points, if your game was chosen as the best game. All the students must vote for someone! (Your lab will not be graded, until you finish your lab and survey.)

Game Description:

The player will be given a list of letters in the alphabetical order. In order to win the game, the player has to guess a word by reordering the letters in a certain time.

Implementation Outline:

1. Interface: Displaying an instruction, a start button, sorted letters, and the input text field.
2. Logic: Implementing functions to start the game and check the answers.
3. Timer: Incorporating the timer functionality to the game.

Lab Instructions:

1. Interface

1.1) Game Instruction
Create a new file on your text editor, and save it with a name "Lab4.html" (or whatever you like). As always, insert <script> and <body> elements.

Now, insert a description of the game in the beginning of the <body> element. Say something like "Guess a name of … by reordering the letters below. Click the Start button to play." Put the line break element ("
") or two, if you like, at the end.

1.2) Start button
Below the question text, insert a <form> element and name it "f1" (or whatever you like). In the form element, insert a button. Set the value attribute to "Start" (or whatever you like) and the onClick attribute to "startGame();". Again, put the line break element or two, if you like.

1.3) Sorted letters
Below the question text, insert a text field that will display the sorted letters. Set the name attribute to "letters" (or whatever you like). Depending on the length of the answer(s), you may want to set the size attribute. For example, if your answer is 30 letters, you may want to say size="30". (That would be a really tough question, though.) Insert a string "Letters: " (or something) if you would like, in front of the text field.

1.4) Input text field.

Below the sorted letters, insert a text saying “Your guess: “ (or something like that). Then put a text field element. Set the name attribute to “guess”.

1.5) Submit Button
Below the text field, insert a button with the value attribute “Submit” (or something like that) and the onClick attribute to “checkAnswer();”.

If you would like, you can put everything in the <body> element in a element. (In other words, you can put everything above between and in the <body> element.)

At this point, open the file with the web browser and make sure everything is displayed properly. It should look like below:

2. Logic

2.1) Answer String
Now, move up to the <script> element, and define a global variable gName, and initialize it with the answer. Do you understand why we need to use a global variable to store the answer? Because the answer needs to be accessed by both the startGame() function and the checkAnswer() function.

2.2) startGame() Function
Now, define the startGame() function. The purpose of this function is to setup the game to start. (Let’s not worry about the timer for now.) Our strategy is to use an array to sort the answer string and display the letters. Below are the step-by-step processes of the function:

1. Declare an array, nameArray, and initialize it by calling “new Array();”.
2. Declare a string variable sortedName, and initialize it with “” (empty string).
3. Copy gName to nameArray, using a for-loop statement. You need to use the charAt() method to access each character in gName.
4. Sort the nameArray, using the sort() method.
5. Copy nameArray to sortedName, using a for-loop statement.
6. Set the value of “letters” text field to sortedName.
7. Set the value of “guess” text field to “” (empty string).

Save the file and reload the browser. Click the star button, and make sure the sorted letters are displayed. If not, this might be a good time to ask for help!

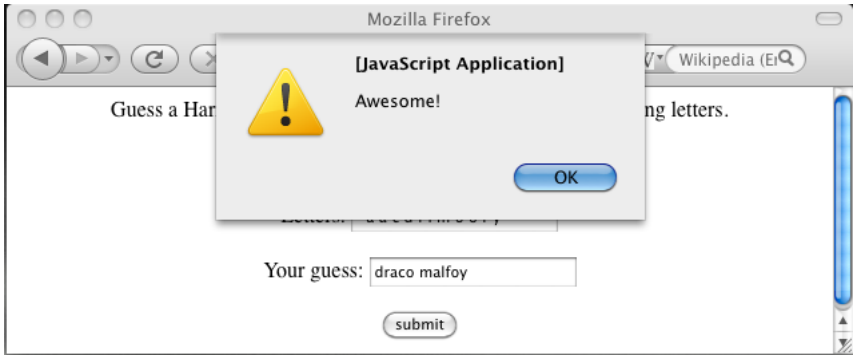
2.2) checkAnswer() Function

Now, define the checkAnswser() function. The purpose of this function is to check if the submitted guess is right and notify the player. This is a pretty simple function involving the following step:

Check if the value of the “guess” element is the same as the gName global variable. Display a message accordingly (e.g. “Way to go!” for the right answer and “You’re wrong!” for the wrong one, etc.) If the answer was wrong, display the correct answer as well.

Save the file and reload the browser. At this point, your game should be working perfectly fine except the timer. Try starting the game and submitting some answer. Make sure the startGame() and checkAnswer() functions are working.

Screenshot with the right answer:



With a wrong answer:

3. Timer

3.1) timeLeft Text Field

In the “f1” <form> element, insert a line to display a remained time. You need to mix simple texts and text field. Remember to use <input> element with type=”text”. Name it “timeLeft”.

3.2) gTimeLeft and gTimer Global Variables
In the beginning of the <script> element, declare a global variable, “gTimeLeft” and “gTimer”.

3.3) startGame() Function
In the beginning of the startGame() function, inivialize the two variables. Set the gTimeLeft variable to whatever the time you want to set (in seconds). Set gTimer by starting a timer. You should remember by now how to start the timer, but just in case, it should look like below:

```
gTimer = setInterval(updateTimer, 1000);
```

Again, this line specifies the function “updateTimer()” should be called every 1000 milliseconds (= 1 second).

3.4) updateTimer() Function
Now, let’s define the “updateTimer()” function that you specified just now. The function needs to do the following:

1. Set the value of the timeLeft text field element to gTimeLeft. (Displaying the time left.)

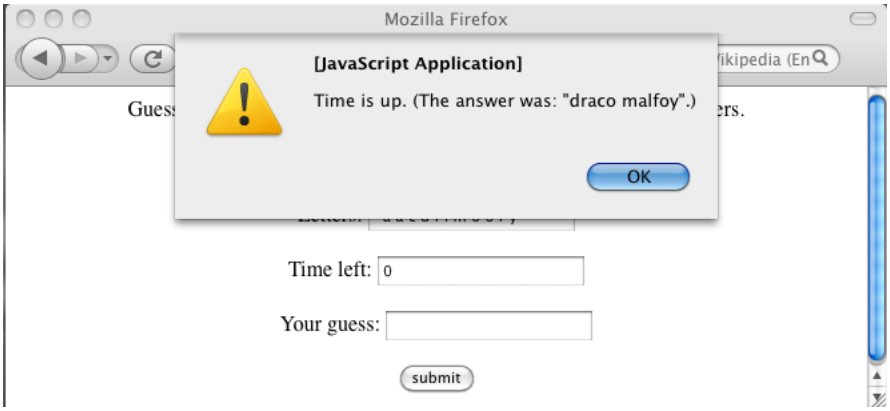
2. Check if the timeLeft is smaller or equal to 0, and if so, stop the timer, and notify the player that the time is up and the correct answer. You should remember by now how to stop the timer, but just in case, it should look like below:

   ```
   clearInterval(gTimer);
   ```

3. Decrease the value of gTimeLeft by 1.

3. 5) checkAnswer() Function
In the definition of the checkAnswer() function, add a line to stop the timer. You have two choices: 1) stop the timer whenever answer is submitted; or 2) stop the timer only when the answer is right. The second choice will give another chance to submit a guess, when the current guess is wrong (if there is still time left.) Choose you whichever you like.

Congratulations! At this point, you finished the implementation of the Scrabble game! Save the file and reload the browser. Make sure your game is functioning correctly, **if you are undergraduate, move on the “Submission” Section in the next page. If you are a graduate student, move on to the “Extra Task for Graduate Students” Section.**



Extra Task for Graduate Students

In the current implementation, the game will provide only one question, which is pretty lame. As an extra task, graduate students need to modify the game so that the player can play the game multiple times (with different answers, of course!) The basic strategy to do this is following:

1. Use an array variable (nameList) to keep the list of possible answers. (Use a local variable.)
2. Use a global variable (gNumPlayed) to keep track how many games have been played.
3. The startGame() funciton initializes the current answer (gName), using gNumPlayed and nameList.
4. The startGame() function also needs to make sure if gNumPlayed cannot exceed the length of nameList. (In other words, startGame() should not start a game after it used all the answers.)
5. The checkAnswer() function now needs to call startGame().

Submission

In order to handle multiple files (including images), from this lab, we will use the Web hosting service that SU provides, called MySite. The Instruction of MySite is found at http://its.syr.edu/mysite/.

Once you setup your MySite account, you will be able to copy your lab assignment to the MySite admin folder, and it will be accessible from the Web.

You will only need to submit the URL to page you created to the discussion board on the ILMS. Create a hyperlink, if you would like. For example, I would submit:

http://kinoue.mysite.syr.edu/IST400-600/Scrabble.html

Grading Criteria

The lab assignment must be submitted by Tuesday Feb. 23rd midnight. The grading will be based on the following criteria:

- Game presentation: 2 points
- Logic: 2 points
- Array syntax: 2 points
- For-loop Syntax: 2 points
- HTML syntax: 2 points

Late submission will be accepted within one week from the original due date, with 2 points deduction. 2 points will be deducted for improper submissions (e.g. not using MySite) or for not submitting the peer survey as well.