**ASSIGNMENT 10**

Your task is to modify the virtual pet application from a previous assignment and create any additional file you need so that the game is available when the user is offline.

To test the game availablility online and offline,  make sure that you serve virtualpet.html through the localhost using our Node server.  Do NOT just open virtualpet.html from the file system.   You should make sure that the application works offline, not just that the web page is loaded.

The virtual pet application html document and associated files are available under resources.

**Submit your modified document virtualpet.html and any additional  file you create**.

**Answer**

* text/html[virtualpet.html](https://myetudes.org/access/mneme/content/private/mneme/09ae2205-2717-4bfc-00cf-33f5bdcd7b48/submissions/15429406/bdd8c42a-48c2-47e4-007a-17da3c1fc274/virtualpet.html)
* application/octet-stream[virtualpet.appcache](https://myetudes.org/access/mneme/content/private/mneme/09ae2205-2717-4bfc-00cf-33f5bdcd7b48/submissions/15429406/b26ce4fe-a60d-4476-006d-0d06a6aecbd1/virtualpet.appcache)

[[https://myetudes.org/ambrosia_library/icons/collapse.gif](https://myetudes.org/portal/tool/acd42055-9bd4-4630-8071-c0425c2388c3/review/15429406/list) Model Answer](https://myetudes.org/portal/tool/acd42055-9bd4-4630-8071-c0425c2388c3/review/15429406/list)

Two attachments are required here:

virtualpet.appcache, a new file with the following content:

**CACHE MANIFEST**

**virtualpet.html**

**../scripts/pet.js**

**../scripts/jquery-1.11.1.js**

**pet.css**

**fish.gif**

**dog.gif**

virtualpet.html, modified to specify the cache manifest as follows:

<!DOCTYPE html>

**<html manifest = "virtualpet.appcache">**

<html>

<head>

<meta charset="utf-8">

<title>Virtual Pet App</title>

<link rel = "stylesheet" type = "text/css" href = "pet.css" media = "all">

</head>

<body class = "day">

<h1 id="adopt">Please adopt a pet by clicking on an image below:</h1>

<div id = "choice">

<input id = "dog" type="image" src="dog.gif" alt="Dog">

<input id = "fish" type="image" src="fish.gif" alt="Fish">

</div>

<div id = "action">

<input id = "feed" type ="button" value = "FEED" class = "disappear">

<input id = "walk" type ="button" value = "WALK" class = "disappear">

<input id = "play" type ="button" value = "PLAY" class = "disappear">

<input id = "clean" type ="button" value = "CLEAN" class = "disappear">

</div>

<script defer src="../scripts/jquery-1.11.1.js"></script>

<script defer src="../scripts/pet.js"></script>

</body>

</html>

**Question 2 of 2     Score: 35   (of possible 35 points)**

Consider our word guessing game.

Your task is to modify the JavaScript program guess.js so that the game stores the number of wins and the total numbers of words played in the local storage.

When the user wins, a message showing the number of wins out of the total number of words played so far is shown:

So if the user played 5 words and won 3, the following is displayed:

Congratulations! You win.    (3/5 wins so far)

You do not have to modify the message displayed when the user loses.

These numbers should be preserved when the user closes and  reopens the browser.  Make sure you don't reinitialize them.

The word guessing game source document and associated files are available under Resources.

Upload your modified JavaScript file guess.js.

**Answer**

* application/javascript[guess.js](https://myetudes.org/access/mneme/content/private/mneme/09ae2205-2717-4bfc-00cf-33f5bdcd7b48/submissions/15429406/f2a679b3-d7b1-42b3-8038-ef2a94caaa6e/guess.js)

[[https://myetudes.org/ambrosia_library/icons/collapse.gif](https://myetudes.org/portal/tool/acd42055-9bd4-4630-8071-c0425c2388c3/review/15429406/list) Model Answer](https://myetudes.org/portal/tool/acd42055-9bd4-4630-8071-c0425c2388c3/review/15429406/list)

The additions specific to this assignment are highlighted below:

/\*

 \* Word Guessing Game - Solution

 \*

 \*/

'use strict';

// Define a container for the game variables

var game = {

  answerPosition: 0,   // position of the answer in the wordList

  display: '',         // the current dash/guessed letters display

  wrong: '',           // all the wrong letters guessed so far

  answer: '',          // the correct answer

  wrongCount: 0,       // the number of wrong guesses so far

  over: false,         // is the game over?

  answersList: [       // list of answers to cycle through

    'JavaScript',

    'document',

    'element',

    'ajax',

    'property',

    'event',

    'propagation',

    'listener',

    'transition',

    'animation'

  ]

  };

function dashes(number) {

    // this function takes a number as a parameter

    // and returns a string with that many dashes

    var result = '';

    for (var i = 1; i <= number; i++)  {

        result = result + '-';

    }

    return result;

}

function check(letter) {

    // Checks all occurrences of the letter guessed against game.answer.

    // Returns true if the guess is correct and false otherwise.

    // Updates the game dash display variable game.display if applicable.

    var position;

    var result = false;

    if (letter) {   // check that guess is not the empty string

        // Find the first occurrence of guess in the answer

        position = game.answer.indexOf(letter);

        // if the guessed letter is found in the answer

        if (position > - 1) {

            result = true;

        }

        while (position >= 0) {

            // update the dash display and find all remaining occurrences

            game.display = game.display.substring(0, position) + letter + game.display.substring(position + 1);

            // get the next occurrence

            position = game.answer.indexOf(letter, position + 1);

        }

    }

    return result;

}

function restart() {

    // Initialize the game at the beginning or after restart

    // Initialize the game variables - the model

    game.answer = game.answersList[game.answerPosition].toLowerCase(); // get the word for this round

    // use the modulo operator to cycle through the wordList

    game.answerPosition = (game.answerPosition + 1) % game.answersList.length;

    game.display = dashes(game.answer.length);

    game.wrong = '';

    game.wrongCount = 0;

    game.over = false;

    // Initialize the web page - the view

    document.getElementById('indicator').value = 0; // initialize the progress bar

    document.getElementById('display').textContent = game.display;

    document.getElementById('wrong').textContent = '';

    document.getElementById('guessedletter').value = '';

    document.getElementById('guessedletter').focus();

}

function play() {

    // Invoked when the user clicks on GUESS

    if (game.over) {// if the game is over

        document.getElementById('wrong').textContent = 'Press RESTART to play again.';  // user has to restart

    } else {

        //if the game is not over yet

        var guess = document.getElementById('guessedletter').value.toLowerCase();

        if (check(guess)) {

            // if the guess is valid

            document.getElementById('display').textContent = game.display;

        } else if (guess) {

            // If it's a wrong non-empty guess

            game.wrong = guess + ' ' + game.wrong;

            game.wrongCount++;

            document.getElementById('wrong').textContent = game.wrong;

            document.getElementById('indicator').value = game.wrongCount;

        }

        // reinitialize the guess

        document.getElementById('guessedletter').value = '';

        document.getElementById('guessedletter').focus();

        // check for a win or loss

        gameOutcome();

    }

}

function gameOutcome() {

    // check if the game is won or lost

    if (game.answer === game.display) {

**updateTotal();**

**updateWins();**

**document.getElementById('wrong').textContent = 'Congratulations!  You win (' +**

**localStorage['Guessing Game Wins'] +**

**'/' +**

**localStorage['Guessing Game Words Played'] +**

**' so far)';**

        game.over = true;  // game is over.  User has to restart to play again

    } else if (game.wrongCount >= 10) {  // 10 guesses only

**updateTotal();**

        document.getElementById('wrong').textContent = 'No more guesses - the answer was ' + game.answer;

        game.over = true;  // game is over.  User has to restart to play again

    }

}

**function updateTotal() {**

**// update the total count of words played in local storage**

**var totalPlayed = Number(localStorage['Guessing Game Words Played']);**

**totalPlayed ++;**

**localStorage.setItem('Guessing Game Words Played', totalPlayed);**

**}**

**function updateWins() {**

**// update the count of words won  in local storage**

**var currentWins = Number(localStorage['Guessing Game Wins']);**

**currentWins ++;**

**localStorage.setItem('Guessing Game Wins', currentWins);**

**}**

**function setupStorage() {**

**// Initialize the guessing game properties in local storage**

**if (! ('Guessing Game Wins' in localStorage)) {**

**localStorage.setItem('Guessing Game Wins', 0)**

**}**

**if (! ('Guessing Game Words Played' in localStorage)) {**

**localStorage.setItem('Guessing Game Words Played', 0)**

**}**

**}**

// Main program starts here

restart();

**setupStorage()**

document.getElementById('guessbutton').addEventListener('click', play, false);

document.getElementById('restart').addEventListener('click', restart, false);

**Comments**

Even though this implementation works, it is best to store and update the actual numbers.

 1 of 1