# Λογισμικό Διαχείρισης Μάθησης

Απαλλακτική εργασία για το ακαδημαϊκό έτος 2023-2024


Λεωνίδας Πάστρας
π20155

Καστανάς Φοίβος Γεώργιος
π21224

Άγγελος Μανουσάκης
π20120

Δημήτρης Γιαγίας
π21018

# Περιεχόμενα

# Εκφώνηση Άσκησης

3) Να αναπτύξετε λογισμικό διαχείρισης ηλεκτρονικού ερωτηματολογίου ερωτήσεων κλειστού τύπου χρησιμοποιώντας μια γλώσσα προγραμματισμού της επιλογής σας. Οι τύποι ερωτήσεων θα είναι i) Σωστού – Λάθους, ii) Πολλαπλής επιλογής με μία ορθή απάντηση, iii) Πολλαπλής επιλογής με περισσότερες της μίας ορθές απαντήσεις, iv) Συμπλήρωσης κενού, v) Αντιστοίχισης, vi) Διάταξης. Στο λογισμικό θα συμπεριλάβετε τουλάχιστον τρεις ερωτήσεις κάθε τύπου (δηλ. συνολικά τουλάχιστον 18 ερωτήσεις) για ένα διδακτικό αντικείμενο της επιλογής σας. Κατά την εκτέλεσή του το λογισμικό θα επιλέγει και θα παρουσιάζει με τυχαίο τρόπο 6 από τις διαθέσιμες ερωτήσεις. Κάθε ερώτηση θα έχει μέγιστο χρόνο επεξεργασίας, για τον οποίο θα πληροφορείται ο χρήστης με αντίστροφη μέτρηση. Κατά την λήξη του διαθέσιμου χρόνου, η ερώτηση θα κλειδώνεται, δηλαδή ο χρήστης θα μπορεί να την δει, αλλά όχι να την επεξεργαστεί πλέον. Επιπλέον, θα εμφανίζεται ο συνολικός υπολειπόμενος χρόνος του ερωτηματολογίου. Στους τύπους ii και iii οι προτεινόμενες απαντήσεις μπορεί να είναι σε μορφή λεκτική ή εικόνων. Στους τύπους v και vi η αντιστοίχιση / διάταξη θα γίνεται με drag’n’drop. Η εφαρμογή θα περιλαμβάνει κουμπί υποβολής του ερωτηματολογίου προς αξιολόγηση. Η διαδικασία υποβολής θα ενεργοποιείται αυτόματα με την λήξη του διαθέσιμου χρόνου. Κατά την αξιολόγηση η εφαρμογή θα υπολογίζει την επίδοση του χρήστη, υπό μορφή βαθμού επί τοις εκατό. Κατά την παράδοση, η εφαρμογή να συνοδεύεται και από ένα έγγραφο που θα τεκμηριώνει και θα περιγράφει αναλυτικά την ανάπτυξη και την λειτουργία της.

Extra βαθμοί:

♦ Ο απομένων χρόνος (ανά ερώτηση και συνολικά) θα κωδικοποιείται χρωματικά, είτε ως οριζόντια μπάρα διαρκώς μειούμενου μήκους ή ως περιφέρεια κύκλου που θα μειώνεται μέχρις εξαφάνισης.

♦ κατά την αξιολόγηση η εφαρμογή θα εμφανίζει ένα κατάλογο με τις 6 ερωτήσεις, τις απαντήσεις του χρήστη καθώς και τις ορθές απαντήσεις, εφόσον αυτές είναι διαφορετικές,

♦ στους τύπους ii και iii θα εμφανίζεται κουμπί βοήθειας, το οποίο θα μπορεί να χρησιμοποιηθεί άπαξ ανά ερώτηση και θα εξαφανίζει μία από τις λανθασμένες απαντήσεις, με ταυτόχρονη μείωση του μέγιστου βαθμού που δικαιούται ο χρήστης,

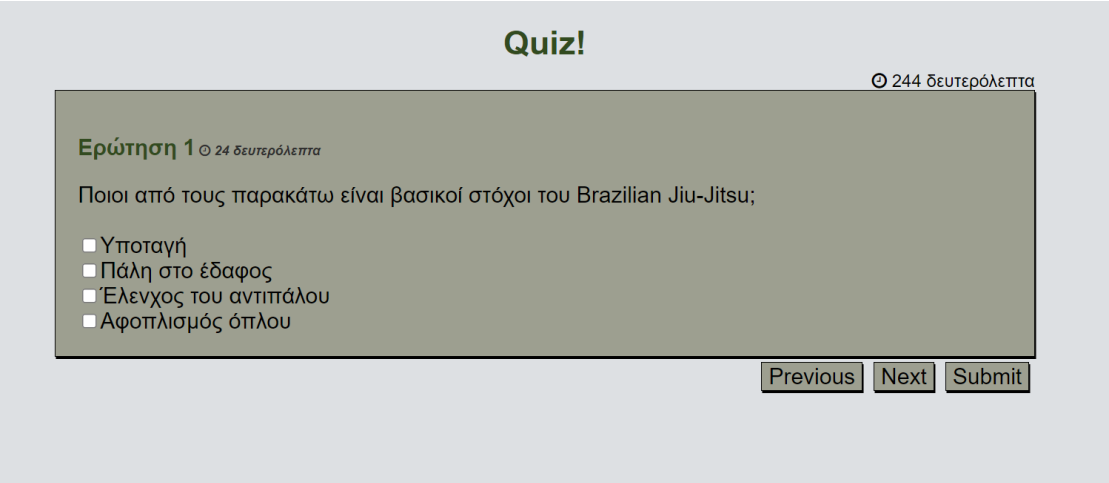
♦ στον τύπο iv το λογισμικό θα αναγνωρίζει μια απάντηση ως ορθή είτε ο χρήστης την έχει πληκτρολογήσει με πεζά, με κεφαλαία ή οποιαδήποτε ανάμειξή τους, με τόνους ή χωρίς και (σε περίπτωση περισσότερων της μίας λέξεων που αντιστοιχούν σε ένα κενό) με ένα ή περισσότερα κενά διαστήματα,

♦ στον τύπο iv το λογισμικό θα αναγνωρίζει μια απάντηση ως ορθή αν έχει πληκτρολογηθεί ανορθόγραφα (π.χ. ‘ω’ αντί για ‘ο’, ή ‘ι’ αντί για ‘οι’ κ.λπ.,

♦ στον τύπο vi το λογισμικό θα αναγνωρίζει την απάντηση ως ορθή είτε η διάταξη έχει γίνει ορθά από αριστερά προς τα δεξιά είτε αντίστροφα. Οι extra βαθμοί είναι ανάλογοι του πλήθους των παραπάνω πρόσθετων χαρακτηριστικών που θα ενσωματώσετε στην εργασία σας. Μπορείτε να συμπεριλάβετε προσθήκες δικής σας επινόησης (τις οποίες θα περιγράφετε συνοπτικά σε ξεχωριστή ενότητα του συνοδευτικού εγγράφου τεκμηρίωσης).

## Λίγα λόγια για το πρόγραμμα

Το λογισμικό διαχείρισης ηλεκτρονικού ερωτηματολογίου αναπτύχθηκε σε HTML , JavaScript και CSS, και αποτελείται απο τα τρία αντίστοιχα αρχεία **«index.html»** , **«QuestionsScript.js»** και **«style.css»** τα οποία βρίσκονται στον φάκελο "code". Έχουν υλοποιηθεί όλα τα βασικά χαρακτηριστικά καθώς και δύο επιπρόσθετα.




## Περιληπτική λειτουργία του προγράμματος

Κατα την εκτέλεση του προγράμματος δημιουργείται μια λίστα με 6 τυχαία νούμερα απο το 0 μεχρι το 19 τα οποία αντιστοιχούν σε κάθε μία απο τις 20 ερωτήσης (π.χ [5, 0, 13, 8, 19, 7]) και προβάλεται στην οθόνη του χρήστη η ερώτηση που αντιστοιχεί στον αριθμό του πρώτου στοιχείου της λίστας (στο παράδειγμα το 5).

Πατώντας το κουμπί ''Next" ή "Previous" για να αλλάξει ερώτηση, το πρόγραμμα πηγάινει αντίστοιχα στο επόμενο ή στο προηγούμενο νούμερο του πίνακα και προβάλει στην οθόνη του χρήστη την αντίστοιχη ερώτηση.

Κάθε ερώτηση έχει ορισμένο χρόνο μέσα στον οπόιο ο χρήστης πρέπει να την απαντήσει. Ο χρόνος κάθε ερώτησης μετράει αντίστροφα μόνο όταν η αντίστοιχη ερώτηση είναι επιλεγμένη απο το χρήστη. Οταν τελειώσει ο χρόνος ο χρήστης δεν έχει πλέον δυνατότητα να επεξεργαστεί την ερώτηση αλλά μπορεί ακόμα να την δεί. Επιπλέον υπάρχει και συνολικός χρόνος για την επεξεργασία του ερωτηματολογίου κατα την λήξη του οπόιου ενεργοποιείται αυτόματα η υποβολή του ερωτηματολογίου.

Κατα την υποβολή του ερωτηματολογίου, είτε συμβεί λόγο της λήξης του διαθέσιμου χρόνου είτε επείδη ο χρήστης πάτησε το κουμπί "Submit", εμφανίζεται στην οθόνη η επίδοση του χρήστη υπό μορφή βαθμού επί τοις εκατό. Επίσης εμφανίζονται όλες οι ερώτησεις καθώς και οι αντίστοιχες απαντήσεις του χρήστη. Αν οι απαντήσεις του χρήστη ήταν λανθασμένες τότε εμφανίζονται επιπρόσθετα και οι σωστές απαντήσεις.

## Αναλυτική τεκμηρίωση ανάπτυξης του κώδικα

Ο κώδικας του αρχείου **«index.html»** περιέχει 6 βασικα στοιχεία.

- Το στοιχείο *<p>* με *id="timer"* που εμφανίζει την αντίθετη χρονομέτρηση του συνόλικου χόνου
- Το στοιχείο *<article>* με *id="mainForm"* μέσα στο οποίο εμφανίζονται δυναμικά οι ερωτήσης
- Τα κουμπιά "Next", "Previous" και "Submit" τα οπόια πατώντας τα καλούνται αντίστοιχα οι συναρτήσεις *NextQuestion()*, *PrevQuestion()* και *ShowResults()*.
- Το στοιχείο *<script>* που καλεί το JavaScript αρχείο **«QuestionsScript.js»** και ξεκινάει την εκτέλεση του ερωτηματολογίου όπως προαναφέρθηκε στην περιλιπτική λειτουργία του προγράμματος.

Ο κώδικας του αρχείου **«QuestionsScript.js»** θα αναλυθεί με την σειρά που αναφέρεται κάθε λειτουργία στην περιλιπτική λειτουργία του προγράμματος.

### ➢ Επιλογή τυχαίων ερωτήσεων

Ορίζουμε την συνάρτηση getRandomInt(min, max) η οποία επιστρέφει έναν τυχαίο ακέραιο αριθμό απο το σύνολο [min, max – 1]. Στην συνέχεια ορίζουμε την σταθέρα «*NUMBER_OF_QUESTIONS*» η οπόια αντιπροσωπεύει τον συνολικό αριθμό των διαθέσιμων ερωτήσεων. Ορίζουμε επίσης την λίστα *questionsIndex[]* και τοποθετούμε στην πρώτη θέση της έναν τυχαίο αριθμό καλώντας την getRandomInt(0, NUMBER_OF_QUESTIONS). Τέλος, με την βοήθεια της βοηθητικής μεταβλητής *randomInt*, γεμίζουμε άλλες 5 θέσης της λίστας *questionsIndex[]* ελέγχοντας κάθε φορά πως δεν υπάρχουν επαναλαμβανόμενοι αριθμοί.

```
2   // Get a random integer between two values
3   // source: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Ref
4   function getRandomInt(min, max) {
5       min = Math.ceil(min);
6       max = Math.floor(max);
7       return Math.floor(Math.random() * (max - min) + min); // The maxim
8   }
9
10  const NUMBER_OF_QUESTIONS = 20;
11  // Fill and array with i number of random numbers that range between
12  // the two numbers inside the getRandomInt(min, max) method
13  let questionsIndex = [];
14  // Define the first number to use as a point of reference
15  questionsIndex[0] = getRandomInt(0, NUMBER_OF_QUESTIONS);
16  let randomInt;
17  let i = 1;
18  do {
19      // get a random integer between 0 and the number of questions
20      randomInt = getRandomInt(0, NUMBER_OF_QUESTIONS);
21      // make sure there are no duplicates
22      if (!questionsIndex.includes(randomInt)) {
23          questionsIndex[i] = randomInt;
24          i++
25      }
26  } while (i < 6)
```

Στο τέλος αυτού του κομματιού κώδικα η λίστα *questionsIndex[]* περιέχει 6 τυχαίους αριθμούς οι οπόιοι ειναι διαφορετικοί μεταξύ τους και ανήκουν στο σύνολο [0, 19]. Πριν αναφέρουμε πως αντιστοιχούμε κάθε ερώτηση σε έναν αριθμό πρέπει πρώτα να εξηγήσουμε πως φτίαχνουμε και επεξεργαζόμαστε τις ερωτήσεις του ερωτηματολογίου.

## ➢ Κλάση QuizQuestion

Κάθε μια απο τις ερωτήσεις του ερωτηματολογίου είναι και ένα αντικείμενο της κλάσης **QuizQuestion.** Δηλαδή υπάρχουν 20 αντικείμενα της κλάσης **QuizQuestion** τα οποία τα έχουμε ονομάσει {q0, q1,…, q19}. Για την δημιουργία κάθε ερώτησης του ερωτηματολογίου κάλουμε ανάλογα με τον τύπο της ερώτησης και διαφορετικό (ψευτό)constructor. Με άλλα λόγια ενώ κάθε αντικείμενο {q0, q1,…, q19} προέρχεται απο την κλάση **QuizQuestion,** ανάλογα με τον τύπο της ερώτησης *(Σωστού – Λάθους, Πολλαπλής επιλογής με μία ορθή απάντηση, Πολλαπλής επιλογής με περισσότερες της μίας ορθές απαντήσεις, Συμπλήρωσης κενού, Αντιστοίχισης, Διάταξης)* το αντικείμενο έχει διαφορετικές ιδιότητες. Συγκεκριμένα κάθε τύπος ερώτησης έχει τις εξείς ιδιότητες

| | Σωστού – Λάθους | Πολλαπλής επιλογής με μία ορθή απάντηση | Πολλαπλής επιλογής με περισσότερες της μίας ορθές απαντήσεις | Συμπλήρωσης κενού | Αντιστοίχισης | Διάταξης |
|---|---|---|---|---|---|---|
| question | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| userAnswers | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| availableWords | | | | | ✓ | ✓ |
| choice1 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| choice2 | ✓ | ✓ | ✓ | | ✓ | ✓ |
| choice3 | | ✓ | ✓ | | ✓ | ✓ |
| choice4 | | | ✓ | | ✓ | ✓ |
| questionType | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| correctAnswers | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| local_time_left | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

- Η ιδιότητα *question* είναι κοινή σε όλους τους τύπους ερωτήσεων διότι είναι η εκφώνηση της ερώτησης.
- Αντίστοιχα κοινή σε όλους τους τύπους ερωτήσεων ειναι η ιδιότητα *userAnswers* που είναι η λίστα με τις απαντήσεις του χρήστη.
- Η ιδιότητα *availableWords* είναι ένα HTML στοιχείο τύπου  το οπόιο περιέχει τις λέξεις που χρησιμοποιούνται σε drag'n'drop ερωτήσεις εξου και μοιράζετε μόνο απο τις ερωτήσεις τύπου *αντίστοιχης* και *διάταξης*.
- Οι ιδιότητες *choice1*, *choice2*, *choice3* και *choice4* αντιστοιχούν στις απαντήσεις/επιλογές που μπορεί να κάνει ο χρήστης. Για παράδειγμα στις ερωτήσεις τύπου *Σωστού – Λάθους* ο χρήστης έχει να διαλέξει ανάμεσα σε δύο επιλογές (σωστό ή λάθος), εξου και αυτές οι ερωτήσεις έχουν μόνο *choice1* και *choice2*.
- Η ιδιότητα *questionType* είναι μια σταθερά που γράφει τον τύπο της ερώτησης (π.χ "MultipleAnswers" για τις ερωτήσης τύπου *Πολλαπλής επιλογής με περισσότερες της μίας ορθές απαντήσεις*). Η ιδιότητα αυτή είναι πολυ βοηθητική δίοτι οι μέθοδοι της κλάσης **QuizQuestion** επεξεργάζονται διαφορετικά κάθε ερώτηση ανάλογα με τον τύπο της.
- Η ιδιότητα *correctAnswers* είναι η λίστα με τις ορθές απαντήσεις της κάθε ερώτησης και συγκρίνεται στο τέλος του quiz με την αντίστοιχη λίστα *userAnswers* για να βαθμολογηθεί η ερώτηση.

- Τέλος η ιδιότητα *local_time_left* είναι ο χρόνος που απομένει σε κάθε ερώτηση. Ο χρόνος αυτός μπορεί να διαφέρει ανάλογα με τον τύπο της ερώτησης.

Για να φτιάξουμε μία ερώτηση κάποιου τύπου καλούμε τον αντίστοιχο (ψευτο)constructor της κλάσης **QuizQuestion**, και περνάμε τα ανάλογα ορίσματα.

```
370 let q0 = QuizQuestion.TrueOrFalse
371 (
372     'Οι ζώνες στο Brazilian Jiu-Jitsu συμβολίζουν το επίπεδο επιδεξιότητας του αθλητή.', // question
373     '<input type="radio" id="q0ans1" name="question0" onclick="q0.SaveAnswer(0)"><label for="q0ans1">Αλήθές</label> ', // choice 1
374     '<input type="radio" id="q0ans2" name="question0" onclick="q0.SaveAnswer(1)"><label for="q0ans2">Ψευδές</label>', // choice 2
375     [1 , 0] // correct answers
376 )
377
463 let q10 = QuizQuestion.MultipleAnswer
464 (
465     'Ποιοι από τους παρακάτω αποτελούν μέρος του "guard" στο Brazilian Jiu-Jitsu;',
466     '<input type="checkbox" id="q10ans1" name="q10" onclick="q10.SaveAnswer(0)"><label for="q10ans1">De la Riva</label><br>', // choice 1
467     '<input type="checkbox" id="q10ans2" name="q10" onclick="q10.SaveAnswer(1)"><label for="q10ans2">Half Guard</label><br>', // choice 2
468     '<input type="checkbox" id="q10ans3" name="q10" onclick="q10.SaveAnswer(2)"><label for="q10ans3">Kneebar</label><br>', // choice 3
469     '<input type="checkbox" id="q10ans4" name="q10" onclick="q10.SaveAnswer(3)">Όλα τα παραπάνω</label><br>', // choice 4
470     [1, 1, 0, 0] // correct answers
471 )
```

*Η δημιουργία της πρώτης και της εντέκατης ερώτησης*

## ➢ Εμφάνιση ερωτήσεων

Αφού φτιαχτούν οι 20 ερωτήσεις τις αποθηκεύουμε στην λίστα questionDataBase[] με την σειρα που δημιουργήθηκαν. Δηλαδή στην πρώτη θέση του πίνακα θα βρίσκεται η πρώτη ερώτηση, στην δεύτερη θέση η δεύτερη ερώτηση κτλ. (ή αλλίως, $\forall i \in [0, 19]$ ισχύει ότι questionDataBase[i] = qi).

Στην συνέχεια ορίζουμε την μεταβλητή *questionShown* μέσα στην οποία αποθηκεύουμε το αντικείμενο/την ερώτηση που είναι κάθε στιγμή στην οθόνη του χρήστη. Ορίζουμε επίσης την μεταβλητή *questionNum* της οποίας η τιμή δηλώνει το νούμερο της ερώτησης που είναι κάθε στιγμή στην οθόνη του χρήστη, και την αρχικοποιούμε με το 0. Είμαστε έτοιμοι πλέον να εμφανίσουμε την πρώτη ερώτηση στον χρήστη.

Οι ερωτήσεις του ερωτηματολογίου εμφανίζονται στην οθόνη του χρήστη μέσα απο την συνάρτηση showQuestion(questionNum). Στην περίπτωση που το πρόγραμμα εκτελείτε για πρώτη φορά και η μεταβλητή *questionNum* είναι 0, συμβαίνει το εξής. Επιλέγεται απο την λίστα *questionsIndex[]* το πρώτο στοιχείο της (*questionsIndex[questionNum]*, *questionNum=0)*. Στην συνέχεια, το στοιχείο αυτο (το οποίο στην πραγματικότητα είναι ένας τυχαίος αριθμός απο το 0 μέχρι το 19) χρησιμοποιήται ως δείκτης της λίστας questionDataBase[] για να επιλεχθεί η ανάλογη ερώτηση, και η ερώτηση αποθηκευεται στην μεταβλητή *questionShown.*

```
function showQuestion(questionNum) {
    questionShown = questionDataBase[questionsIndex[questionNum]];
```

Τέλος καλώντας την *getter function* Question() της κλάσης **QuizQuestion** (η οποία επιστρέφει συγχωνευμένες της κατάλληλες ιδιότητες του αντικειμένου για την κατασκευή της ερώτησης) στο αντικείμενο *questionShown*, εισάγουμε δυναμικά τον HTML κώδικα στο στοιχείο *<article>* με *id="mainForm"*. Έτσι εμφανίζεται στην οθόνη του χρήστη η πρώτη ερώτηση του ερωτηματολογίου.

## ➢ Κουμπιά “Next”και “Previous”

Πατώντας το κουμπί ‘’Next” και “Previous” καλούνται αντίστοιχα οι συναρτήσεις *NextQuestion()* και *PrevQuestion()* οι οποίες αυξομοιώνουν ανάλογα την μεταβλητή *questionNum* και καλούν την συνάρτηση showQuestion(questionNum). Επιπλέον γίνεται έλεγχος πως η τιμή της *questionNum* δεν φεύγει απο το [0, 5] αφού έχουμε πεί πως η *questionsIndex[]* έχει 6 στοιχεία.

```
681  // Change to the next question
682  function NextQuestion() {
683      questionNum == 5 ? questionNum = 0 : questionNum++;
684      showQuestion(questionNum);
685  }
686
687  // Change to the previous question
688  function PrevQuestion() {
689      questionNum == 0 ? questionNum = 5 : questionNum--;
690      showQuestion(questionNum);
691  }
692
```

## ➢ Υπολειπόμενος χρόνος ερωτήσεων

Η αντίστροφη μέτρηση του διαθέσιμου χρόνου της κάθε ερώτησης γίνεται με την βοήθεια της μεθόδου setInterval(). Η μέθοδος setInterval() είναι μια μέθοδος ενσωματτωμένη στην JavaScript που παίρνει ως ορίσματα μία μέθοδο και έναν ακέραιο αριθμό. Με το που την καλέσουμε η setInterval() καλεί την μέθοδο που της περάσαμε ως όρισμα σε τακτά διαστήματα μεγέθους ίσα με του ακεραίου που του ορίσαμε σε milliseconds. Στο δικό μας πρόγραμμα κάλουμε την setInterval() με ορίσματα την μέθοδο UpdateTimeLeft() και τον αριθμό *1000* (1000 milliseconds δηλαδή 1 δευτερόλεπτο). Πριν αναλύσουμε πως λειτουργεί η UpdateTimeLeft() πρέπει να αναφερθούν δύο πράγματα.

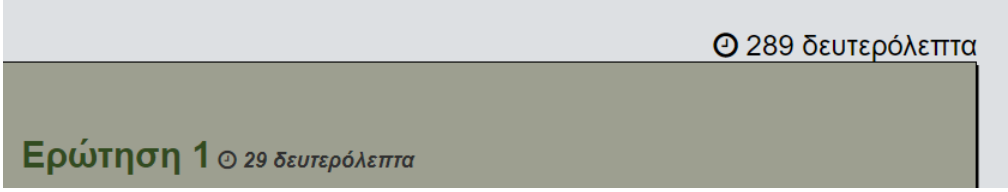
1. Κάθε αντικέιμενο της κλάσης **QuizQuestion**, δηλαδή κάθε ερώτηση, έχει την ιδιότητα *local_time_left* που είναι ο χρόνος σε δευτερόλεπτα που της απομένει. Επιπλέον έχει την μέθοδο UpdateLocalTimeLeft() η οποία μειώνει την τιμή της *local_time_left* κατα 1 και την επιστρέφει.

2. Ο Συνολικός χρόνος του ερωτηματολογίου είναι το άθροισμα των χρόνων όλων των ερωτήσεων και υπολογίζεται απο την μέθοδο getTotalTimeLeft().

Κάθε φορά που καλείται η UpdateTimeLeft() (δηλαδή κάθε 1 δευτερόλεπτο) γίνονται τα παρακάτω. Ελέγχει με την μέθοδο getTotalTimeLeft() τον συνολικό χρόνο που απομένει στο ερωτηματολόγιο και εμφανίζει τα αποτελέσματα του ερωτηματολογίου αν έχει τελειώσει. Αν δεν έχει τελειώσει ο συνολικός χρόνος τότε ελέγχει αν η τιμη της *local_time_left* της ερώτησης που έχει εκείνη την στιγμή επιλεγμένη ο χρήστης είναι 0, δηλαδή ελέγχει αν έχει τελειώσει ο διαθέσιμος χρόνος της. Στην περίπτωση που ο υπολειπόμενος χρόνος της ερώτησης έχει λείξει:

1) Κλειδώνει την ερώτηση με την μέθοδο DisableAnswers(), η οπόια βάζει σε μία λίστα όλα τα HTML στοιχεία που βρίσκονται εκείνη την στιγμή στην οθόνη του χρήστη και καταργεί όποιο στοιχείο δέχεται είσοδο.
2) Αλλάζει το μήνημα στην θέση του διαθέσιμου χρόνου της ερώτησης σε *«Τέλος χρόνου ερώτησης»*.
3) Μειώνει την τιμή της ιδιοτητα *local_time_left* με την μέθοδο UpdateLocalTimeLeft(). Δηλαδή ο συνολικός υπολειπόμενος χρόνος συνεχίζει να μειώνεται αφού ισούται με το άθροισμα των χρόνων όλων των ερωτήσεων.

Στην περίπτωση που ο υπολειπόμενος χρόνος της ερώτησης δεν έχει τελειώσει τότε απλώς τον μείώνει με την μέθοδο UpdateLocalTimeLeft() και ενημερώνει το μήνημα στην θέση του διαθέσιμου χρόνου. Τέλος ενημερώνει το μήνημα στην θέση του συνολικού υπολειπόμενου χρόνου.




Στην περίπτωση που ο συνολικός χρόνος τελειώσει ή αν ο χρήστης έχει πατήσει το κουμπί "Submit" καλείται η μέθοδος clearInterval() που καταργεί τη λειτουργία της setInterval() και η αντίστροφη μέτρηση τελειώνει.

## ➢ Υποβολή ερωτηματολογίου

Όταν τελειώνει ο συνολικός υπολειπόμενος χρόνος η όταν ο χρήστης πατήσει το κουμπί "Submit" καλείται η μέθοδος ShowResults().

1) Αρχικά ορίζουμε την μεταβλητή *resultPage* ως το HTML στοιχείο με *id=*"mainForm" μέσα στο οποίο προηγουμένος εμφανιζοντουσαν οι ερωτήσεις. Στην συνέχεια αν η ShowResults() κλήθηκε λόγο της λείξης του συνολικού υπολειπόμενου χρόνου αρχικοποιούμε την τιμή της *resultPage* να λεέι «Τέλος Χρόνου», αλλίως την αρχικοποιούμε με το κενό.

```
// Creates the Results page that appears
// after pressing the "Submit" button
function ShowResults(time_is_up=false) {
    let resultPage = document.getElementById("mainForm");
    if (time_is_up) {
        resultPage.innerHTML = "<h3>Τέλος Χρόνου!<h3>";
    } else {
        resultPage.innerHTML = "";
    }
```

2) Υπολογίζουμε τον βαθμό του χρήστη με τις μεταβλητές *totalScore* και *percentageResult*. Για κάθε μια απο τις 6 ερωτήσεις καλούμε την μέθοδο CalculateScore(). Η CalculateScore() συγκρίνει την ιδιότητα *userAnswers* (τις απαντήσεις του χρήστη) με την *correctAnswers* (τις σωστές απαντήσεις) και βαθμολογέι με άριστα το 1. Δηλαδή η CalculateScore() επιστρέφει έναν αριθμό απο το 0 μέχρι το 1 τον οποίο προσθετουμε στην *totalScore*. Έτσι μετά τον υπολογισμό των 6 ερωτήσεων η τιμή της *totalScore* είναι ένας πραγματικός αριθμός μεταξύ του 0 και του 6. Άρα για να υπολογίσουμε το ποσοστο επι της εκατό διαιρούμε την *totalScore* με το 6, την πολλαπλασιάζουμε με το 100 και αποθηκεύουμε το αποτέλεσμα στην *percentageResult*. Τέλος εμφανίζουμε την *percentageResult* στην οθόνη του χρήστη με την βοήθεια της *resultPage*.

```
    let totalScore = 0;
    let currentQuestion;
    // Calculate the score for each of the 6 questions
    for (let i = 0; i < 6; i++) {
        currentQuestion = questionDataBase[questionsIndex[i]];
        totalScore += currentQuestion.CalqulateScore();
    }
    // Convert the result into a percentage and show it
    let pecentageResult = Math.ceil(totalScore/6 * 100);
    resultPage.innerHTML += `<h3>Αποτέλεσμα: ${pecentageResult}%</h3>`;
```

3) Μετά που εμφανίσουμε την επίδοση του χρήστη υπό μορφή βαθμού επί τοις εκατό εμφανίζουμε τις ίδιες τις ερωτήσεις μαζί με τις αντίστοιχες απαντήσεις που έδωσε ο χρήστης. Επιπλέον σε όποια ερώτηση ο χρήστης έκανες λάθος εμφανίζονται οι σωστές απαντήσης απο κάτω της καλώντας την μέθοδο της κλάσης **QuizQuestion**, ShowCorrectAnswers().

```
// Show each questions along with the answers of the user.
// Also show the correct answer(s) if the user was wrong.
for (let i = 0; i < 6; i++) {
    currentQuestion = questionDataBase[questionsIndex[i]];
    // Show silly icon next to each question
    if (currentQuestion.CalqulateScore() == 1) {
        resultPage.innerHTML += `<h3>Ερώτηση ${i + 1} <i class="fa fa-check"></i></h3>`;
    } else {
        resultPage.innerHTML += `<h3>Ερώτηση ${i + 1} <i class="fa fa-times"></i></h3>`;
    }
    resultPage.innerHTML += questionDataBase[questionsIndex[i]].Question + "<br>";
    // Show correct answer(s) if the user was wrong
    if (currentQuestion.CalqulateScore() != 1) {
        resultPage.innerHTML += currentQuestion.ShowCorrectAnswers();
    }
}
```

4) Τέλος αλλάζει την τιμή της μεταβλητής *quizOver* σε true (μια βοηθητική μεταβλητή που δηλώνει στην UpdateTimeLeft() πως πρέπει να σταματήσει την αντίστροφη μέτρηση), καλέι την DisableAnswers() διότι διαφορετικά ο χρήστης θα μπορουσε να κάνει αλλαγές στην απαντήσης του στην σελίδα των αποτελεσμάτων, και κρύβει τα κουμπία “Next”, “Previous” και “Submit” αφού πλέον δεν χρειάζονται.

```
    // quiz is over!
    quizOver = true;
    // Disable the ability for the user to change his answers
    DisableAnswers();
    // Hide the buttons that are no longer needed
    document.getElementById("prevBtn").hidden = true;
    document.getElementById("nextBtn").hidden = true;
    document.getElementById("subBtn").hidden = true;
}
```

## ➢ Extra Βαθμοί

- Όταν βοθμολογούνται απαντήσεις τύπου συμπλήρωσης κενού το πρόγραμμα αναγνωρίζει μια απάντηση ως ορθή είτε ο χρήστης την έχει πληκτρολογήσει με πεζά, με κεφαλαία ή οποιαδήποτε ανάμειξή τους και με ένα ή περισσότερα κενά διαστήματα.

  Μέσα στην μέθοδο CalqulateScore(), Όταν ο τύπος την ερώτησης είναι συμπλήρωση κενού, πρίν συγκρίνει την απάντηση του χρήστη με την ορθή απάντηση, κάνει την εξής προεργασία. Αρχικά μετατρέπει και τις δύο απαντήσεις σε κεφαλαία με την μέθοδο toUpperCase() και στην συνέχεια απαλείφει τα κένα διαστήματα με την μέθοδο replace() με ορίσματα “ “ & “” (δηλαδή αντικαθιστά τα κενά διαστήματα με το τίποτα). Έτσι για παράδειγμα αν η σωστή απάντηση είναι «Guard Pass» και ο χρήστης δώσει ως απάντηση «guardpass» το πρόγραμμα θα συγκρίνει τις τιμές «GUARDPASS» και «GUARDPASS», που είναι προφανός ίδιες.

- Κατά την αξιολόγηση η εφαρμογή εμφανίζει ένα κατάλογο με τις 6 ερωτήσεις, τις απαντήσεις του χρήστη καθώς και τις ορθές απαντήσεις, εφόσον αυτές είναι διαφορετικές.

  Η εκτέλεση αυτής της ερώτησης ήταν μακράν το δυσκολότερο κομμάτι την άσκησης. Ο τρόπος με τον οποίο ο HTML κώδικας εισάγεται δυναμικά στο **«index.html»** σημαίνει πώς κάθε αλλάγη που κάνει ο χρήστης σε μια ερώτηση πατώντας κουμπία, κάνωντας drag λέξεις, συμπληρώνοντας κενά κτλ. Πρέπει να σώζεται μέσα στον HTML κώδικα έτσι ώστε την επόμενη φορά που θα εμφανιστεί η ερώτηση, ο HTML κώδικας που θα εισαχθεί στο **«index.html»** να έχει τις ανάλογες αλλάγες που αντιστοιχούν στις απαντήσεις του χρήστη.

  Ο κώδικας για αυτή την λειτουργία είναι όλος μέσα στην κλάση **QuizQuestion**, και μοιράζεται μεταξύ 5 μεθόδων.

  i. FormatTrueOrFalseQuestions()
  ii. FormatMultipleChoiceQuestions()
  iii. FormatMultipleAnswersQuestions()
  iv. FormatCompleteSentenceQuestions()
  v. FormatWordsQuestions()

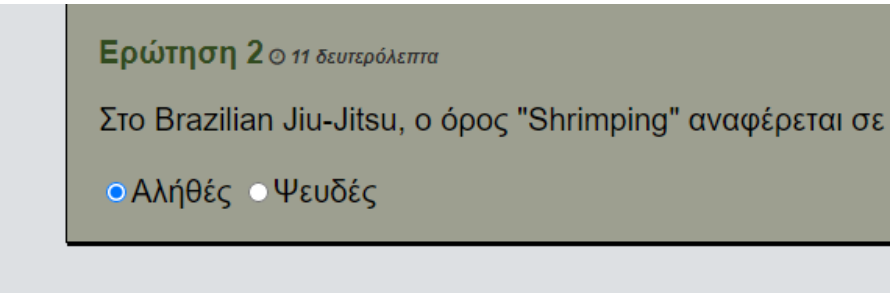
  Οι 5 αυτές μέθοδοι καλούνται έμεσα η άμεσα μέσω της μεθόδου SaveAnswer() η οποία ανάλογα με τον τύπο της ερώτησης καλέι την αντίστοιχη μέθοδο (ειναι 5 διότι οι ερτωτήσεις *Αντιστοίχισης και Διάταξης* χρησιμοποιούν την ίδια μέθοδο FormatWordsQuestions()). Η SaveAnswer() καλείται κάθε φορά που ο χρήστης συμπληρώσει/επιλέξει κάποια απάντηση, συγκεκριμένα:

  Όλα τα κουμπία των ερώτήσεων *Σωστού – Λάθους, Πολλαπλής επιλογής με μία ορθή απάντηση και Πολλαπλής επιλογής με περισσότερες της μίας ορθές απαντήσεις* έχουν στον κώδικά τους την ιδιότητα onclick=”SaveAnswer()” . Δήλαδή κάθε φορά που ο χρήστης πατάει ένα κουμπί καλείται η μέθοδος SaveAnswer().

  Επίσης τα πλαίσια των ερτωτήσεων *Συμπλήρωσης κενού* έχουν στον κώδικά τους την ιδιότητα oninput=”SaveAnswer()” . Δήλαδή κάθε φορά που ο χρήστης αλλάζει την τιμή της εισόδου καλείται η μέθοδος SaveAnswer().

  Τέλος οι drag’n’drop λέξεις των ερτωτήσεων *Αντιστοίχισης και Διάταξης* έχουν στον κώδικά τους κάτι παρόμοιο που καλέι την SaveAnswer() όταν ο χρήστης αλλάξει κάποια απάντηση.

## Παράδειγμα




Ας πάρουμε ως παράδειγμα μια ερώτηση σωστού λάθους. Πριν αναλύσουμε τις ενέργειες που γίνονται όταν ο χρήστης επιλέγει μια απάντηση, να θυμηθούμε πρώτα τις ιδιότητες αυτής της ερώτησης.

- *question*: 'Στο Brazilian Jiu-Jitsu, ο όρος "Shrimping" αναφέρεται σε ένα είδος χτυπήματος.'
- *userAnswers*: [0, 0]
- *choice1*: '<input type="radio" id="q2ans1" name="question2" onclick="q2.SaveAnswer(0)"><label for="q2ans1">Αλήθές</label> '
- *choice2*: '<input type="radio" id="q2ans2" name="question2" onclick="q2.SaveAnswer(1)"><label for="q2ans2">Ψευδές</label>'
- *questionType*: "TrueOrFalse"
- *correcrAnswers*: [1, 0]
- *local_time_left*: 20

Όταν ο χρήστης επιλέξει την επιλογή «Αληθές» πραγματοποιούνται οι παρακάτω ενέργειες.

1) Πυροδοτείτε η μέθοδος onclick="q2.SaveAnswer(0)" που φαίνεται στην *choice1*. Αυτή η μέθοδος καλεί την SaveAnswer() του αντικειμένου **q2** με όρισμα 0 (διαφορετικά το όρισμα θα ήταν 1 αν ο χρήστης διάλεγε «Ψευδές»).
2) Η SaveAnswer() ελέγχει το *questionType* που είναι "TrueOrFalse", και αποθηκεύει την απάντηση του χρήστη στην λίστα *userAnswers*. Σε αυτό το παράδειγμα η *userAnswers* θα γίνει [1, 0] αφού η SaveAnswer() κλήθηκε με όρισμα 0.
3) Τέλος μέσα από την SaveAnswer() καλείτε η FormatTrueOrFalseQuestions(). Η μέθοδος αυτή παίρνει σαν όρισμα το ίδιο όρισμα με την SaveAnswer() δηλαδή σε αυτό το παράδειγμα το 0, και αλλάζει τον HTML κώδικα μέσα στα *choice1* και *choice2* έτσι ώστε να αντιστοιχούν στην απάντηση του χρήστη. Συγκεκριμένα:
   a) Διχοτομεί με την μέθοδο substr() την *choice1* και της προσθέτει την ιδιότητα "checked" που δηλώνει στην HTML πως η είσοδος αυτή είναι επιλεγμένη.
   b) Επιπλέον με την μέθοδο replace() αφαιρεί από την *choice2* την ιδιότητα "checked" σε περίπτωση που υπάρχει (διότι στις ερωτήσεις *Σωστού – Λάθους* μπορεί να υπάρχει μόνο μια απάντηση).

Έτσι όταν ξαναεμφανιστεί αυτή η ερώτηση στον κατάλογο με όλες τις ερωτήσεις κατά την αξιολόγηση, ο κώδικας μέσα στην ερώτηση θα είναι παραμετροποιημένος με τέτοιο τρόπο που θα φαίνονται οι απαντήσεις του χρήστη!

**Με ανάλογο τρόπο λειτουργούν και οι υπόλοιπες 4 μέθοδοι.**

Οι ορθές απαντήσεις κάτω από ερωτήσεις που ο χρήστης έχει απαντήσει λανθασμένα εμφανίζονται μέσω την μεθόδου ShowCorrectAnswers() η οποία απλώς καλείται όταν κάποια απάντηση είναι λάθος (δηλαδή όταν η CalculateScore() δεν είναι 1) και εμφανίζει με όμορφο τρόπο την λίστα *correctAnswers*.

### ➢ Drag'n'Drop

Δουλεύει.

## Πηγές

- Μέθοδος getRandomInt(): https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Math/random
- Drag'n'Drop foundation: https://jsfiddle.net/xptLU/