ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ
ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΤΜΗΜΑΤΟΣ ΕΦΑΡΜΟΣΜΕΝΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΣΧΕΔΙΑΣΗ ΚΑΙ ΑΝΑΠΤΥΞΗ ΠΑΙΧΝΙΔΙΟΥ ΣΟΒΑΡΟΥ ΣΚΟΠΟΥ ΜΕ ΣΤΟΧΟ ΤΗΝ ΕΚΜΑΘΗΣΗ HTML, CSS ΚΑΙ JAVASCRIPT

Διπλωματική Εργασία

της

Ευγενίας Πλιάτσικα

Θεσσαλονίκη, Ιούνιος 2022

ΣΧΕΔΙΑΣΗ ΚΑΙ ΑΝΑΠΤΥΞΗ ΠΑΙΧΝΙΔΙΟΥ ΣΟΒΑΡΟΥ ΣΚΟΠΟΥ ΜΕ ΣΤΟΧΟ ΤΗΝ ΕΚΜΑΘΗΣΗ HTML, CSS ΚΑΙ JAVASCRIPT

Ευγενία Πλιάτσικα

Πτυχίο Εφαρμοσμένης Πληροφορικής, Πανεπιστήμιο Μακεδονίας, 2017

Διπλωματική Εργασία

υποβαλλόμενη για τη μερική εκπλήρωση των απαιτήσεων του

ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΤΙΤΛΟΥ ΣΠΟΥΔΩΝ ΣΤΗΝ ΕΦΑΡΜΟΣΜΕΝΗ ΠΛΗΡΟΦΟΡΙΚΗ

Επιβλέπων Καθηγητής
Ξυνόγαλος Στυλιανός

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την

ΞΥΝΟΓΑΛΟΣ ΣΤΥΛΙΑΝΟΣ	ΓΕΩΡΓΙΑΔΗΣ ΧΡΗΣΤΟΣ	ΚΑΣΚΑΛΗΣ ΘΕΟΔΩΡΟΣ
.....	.....	.....

Πλιάτσικα Ευγενία

.....

Περίληψη

Η παρούσα διπλωματική εργασία έχει ως σκοπό τη σχεδίαση και ανάπτυξη ενός διαδικτυακού παιχνιδιού σοβαρού σκοπού για την εκμάθηση των HTML, CSS και JavaScript. Το παιχνίδι, το οποίο ονομάζεται "Super Dwarf: Quest for the prototype hammer", απευθύνεται σε παίκτες που δεν έχουν έρθει σε επαφή με αυτές τις τεχνολογίες, αλλά και σε όσους τις γνωρίζουν και θέλουν να τις κατανοήσουν καλύτερα. Η υλοποίησή του έγινε με το εργαλείο Phaser, που αποτελεί πλαίσιο ανάπτυξης δισδιάστατων παιχνιδιών.

Εκτός από την υλοποίηση του παιχνιδιού στο πλαίσιο της εργασίας παρουσιάζονται και αναλύονται και άλλα παιχνίδια σοβαρού σκοπού που έχουν ως στόχο τη διδασκαλία της γλώσσας JavaScript, αλλά και του CSS. Επίσης, αναλύεται με βάση το πλαίσιο σχεδίασης Educational Games Design Model (Ibrahim & Jaafar, 2009), ο τρόπος σχεδίασης του παιχνιδιού και οι τεχνολογίες που χρησιμοποιήθηκαν για την υλοποίηση του.

Στη συνέχεια γίνεται μια αναλυτική παρουσίαση του εκπαιδευτικού περιεχομένου, των διεπαφών του χρήστη καθώς και της συνολικής υλοποίησής του παιχνιδιού. Τέλος, παρουσιάζονται τα αποτελέσματα από την αξιολόγηση των χρηστών καθώς και τα συμπεράσματα που προέκυψαν από αυτή.

Λέξεις Κλειδιά: παιχνίδια σοβαρού σκοπού, HTM, CSS, JavaScript, Phaser

Abstract

The aim of this master thesis is to design and develop an online serious game for learning HTML, CSS and Javascript. This game, called "Super Dwarf: Quest for the prototype hammer", is both for users who have not been exposed to these technologies and also users who have a previous experience on them and they want to understand them even better. The game was implemented with the Phaser tool, which is a framework for the development of two-dimensional games.

In addition to the game's implementation, in this thesis other serious games that aim to teach the JavaScript language and CSS are presented and analyzed . Moreover, the design of the new game is analyzed based on the design framework Educational Games Design Model (Ibrahim & Jaafar, 2009), and the technologies used to implement it are presented.

Then there is a detailed presentation of the educational content, user interfaces and the overall implementation of the game. Finally, the results from the users' evaluation and the conclusions that emerged from it are presented.

Keywords: serious games, HTM, CSS, JavaScript, Phaser

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή κ. Ξυνόγαλο Στέλιο, για τη βοήθεια και τις συμβουλές που μου έδωσε για την ολοκλήρωση της διπλωματικής μου εργασίας. Τέλος, ευχαριστώ όσους δοκίμασαν και αξιολόγησαν το παιχνίδι που υλοποιήθηκε.

Περιεχόμενα

1 Εισαγωγή 1
1.1 Πρόβλημα – Σημαντικότητα του θέματος 1
1.2 Σκοπός – Στόχοι 1
1.3 Συνεισφορά 2
1.4 Διάρθρωση της μελέτης 2
2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο 3
2.1 Εισαγωγή 3
2.2 Παιχνίδια σοβαρού σκοπού 4
2.2.1 CodeCombat 4
2.2.2 Crunchzilla 6
2.2.3 CSS Diner 8
2.2.4 Flexbox Froggy 9
2.2.5 Grid Garden 10
2.3 Πλαίσιο σχεδίασης 11
2.4 Ανάλυση και σύγκριση παιχνιδιών σοβαρού σκοπού 13
3 Μεθοδολογία 23
3.1 Σκοπός του παιχνιδιού 23
3.2 Ανάλυση και σχεδίαση του παιχνιδιού 23
3.3 Τεχνολογίες υλοποίησης του παιχνιδιού 27
3.3.1 Πλαίσιο εργασίας Phaser 27
3.3.2 Τεχνολογίες για το συντάκτη κώδικα 28
3.3.3 Πρόγραμμα σχεδίασης χαρτών Tiled map editor 29
3.3.4 Nodejs και modules 30
3.3.5 MongoDB Atlas 32
4 Παρουσίαση του παιχνιδιού 33
4.1 Περιγραφή 33
4.2 Εκπαιδευτικό περιεχόμενο - Επίπεδα παιχνιδιού 35
4.2.1 Επίπεδο 1: Εισαγωγή στην HTML 35
4.2.2 Επίπεδο 2: Attributes και παρουσίαση elements 36
4.2.3 Επίπεδο 3: Δομή αρχείου html και δημιουργία συνδέσμων 36
4.2.4 Επίπεδο 4: Δημιουργία πίνακα και φόρμας 36

4.2.5 Επίπεδο 5: Σημασιολογική HTML 37
4.2.6 Επίπεδο 6: Εισαγωγή στο CSS, selectors και δημιουργία κανόνων 37
4.2.7 Επίπεδο 7: Ιδιότητες κειμένου 37
4.2.8 Επίπεδο 8: Κατανόηση του box model 38
4.2.9 Επίπεδο 9: Τοποθέτηση elements με χρήση CSS 38
4.2.10 Επίπεδο 10: Εισαγωγή στο Flexbox 38
4.2.11 Επίπεδο 11: Εισαγωγή στο CSS Grid 38
4.2.12 Επίπεδο 12: Εισαγωγή στην JavaScript 39
4.2.13 Επίπεδο 13: Οι συναρτήσεις στη JavaScript 39
4.2.14 Επίπεδο 14: Πίνακες, βρόχοι και iterators 39
4.2.15 Επίπεδο 15: Αντικείμενα (objects) και η λέξη κλειδί this 40
4.2.16 Επίπεδο 16: Εισαγωγή στο Document Object Model (DOM) 40
4.2.17 Επίπεδο 17: Τροποποίηση του document 41
4.2.18 Επίπεδο 18: DOM events, φάσεις πυροδότησης και event object 41
4.2.19 Επίπεδο 19: Mouse, keyboard, scroll events και event delegation 41
4.3 Τελικές οθόνες χρήστη 42
4.3.1 Είσοδος και εγγραφή χρήστη 42
4.3.2 Επαναφορά συνθηματικού 43
4.3.3 Αρχική οθόνη και κεντρικό μενού 44
4.3.4 Οθόνη σεναρίου και οδηγιών 45
4.3.5 Οθόνη επιπέδων 46
4.3.6 Βασικά στοιχεία επιπέδου 47
4.3.7 Οθόνη ολοκλήρωσης επιπέδου 50
4.3.8 Οθόνη παρουσίασης θεωρίας 52
4.3.9 Οθόνη ερώτησης πολλαπλής επιλογής 53
4.3.10 Οθόνη συντάκτη κώδικα 56
5 Υλοποίηση του παιχνιδιού 62
5.1 Υποδομή του server 62
5.2 Ανάλυση της βάσης δεδομένων 73
5.3 Υποδομή του client 76
5.3.1 Διεπαφές χρήστη 76
5.3.2 Σκηνές του παιχνιδιού 84
5.3.3 Αντικείμενα του παιχνιδιού 99

5.4 Γραφικά και ήχοι 106
5.5 Σχεδίαση χαρτών 106
6 Αξιολόγηση του παιχνιδιού 108
6.1 Μεθοδολογία 108
6.2 Αποτελέσματα αξιολόγησης 108
6.2.1 Δημογραφικά αποτελέσματα 108
6.2.2 Εμπειρία χρήστη 111
6.2.3 Μαθησιακά αποτελέσματα 116
6.2.4 Ερωτήσεις ανοιχτού τύπου 116
6.2.5 Συμπεράσματα αξιολόγησης 118
7 Επίλογος 119
7.1 Σύνοψη και συμπεράσματα 119
7.2 Όρια και περιορισμοί της έρευνας 119
7.3 Μελλοντικές Επεκτάσεις 120
Βιβλιογραφία 121
Παράρτημα 123

Κατάλογος Εικόνων

Εικόνα 1: Επιλογή γλώσσας προγραμματισμού και ήρωα στο παιχνίδι CodeCombat 5
Εικόνα 2: Επίπεδο στο παιχνίδι GameCombat 6
Εικόνα 3: Οι θεματικές ενότητες του παιχνιδιού Crunchzilla 7
Εικόνα 4: Επίπεδο στο παιχνίδι Crunchzilla 8
Εικόνα 5: Το παιχνίδι CSS Diner 9
Εικόνα 6: Το παιχνίδι Flexbox Froggy 10
Εικόνα 7: Το παιχνίδι Grid Garden 11
Εικόνα 8: Πλαίσιο σχεδίασης Educational Games Design Model (Ibrahim & Jaafar, 2009) 12
Εικόνα 9: Λογότυπο Phaser 27
Εικόνα 10: Λογότυπο Ace 28
Εικόνα 11: Λογότυπο Tiled Map Editor 30
Εικόνα 12: Λογότυπο Node.js 30
Εικόνα 13: Λογότυπο MongoDB 32
Εικόνα 14: Είσοδος χρήστη 42
Εικόνα 15: Εγγραφή χρήστη 43
Εικόνα 16: Ανάκτηση συνθηματικού χρήστη 43
Εικόνα 17: Επαναφορά συνθηματικού χρήστη 44
Εικόνα 18: Αρχική οθόνη και κεντρικό μενού 45
Εικόνα 19: Σενάριο και οδηγίες του παιχνιδιού 46
Εικόνα 20: Οθόνη επιπέδων του παιχνιδιού 47
Εικόνα 21: Έναρξη επιπέδου 48
Εικόνα 22: Επιβεβαίωση αποχώρησης από το επίπεδο 49
Εικόνα 23: Σημείο τερματισμού επιπέδου 50
Εικόνα 24: Επιτυχής ολοκλήρωση επιπέδου 51
Εικόνα 25: Μη επιτυχής ολοκλήρωση επιπέδου 52
Εικόνα 26: Παρουσίαση θεωρίας 53
Εικόνα 27: Ερώτηση πολλαπλής επιλογής 54
Εικόνα 28: Ερώτηση πολλαπλής επιλογής με λάθος επιλογή απάντησης 55
Εικόνα 29: Εμφάνιση θεωρίας από το κουμπί “Θεωρία” 56
Εικόνα 30: Εμφάνιση συντάκτη κώδικα 57

Εικόνα 31: Εμφάνιση μηνύματος ανατροφοδότησης..... 58
Εικόνα 32: Επανατοποθέτηση βράχων μετά την εκτέλεση κώδικα..... 59
Εικόνα 33: Εμφάνιση κρυφού μηνύματος μετά την εκτέλεση κώδικα..... 59
Εικόνα 34: Έδαφος που δεν επιτρέπει τη συνέχεια του ταξιδιού..... 60
Εικόνα 35: Εμφάνιση πλατφόρμας μετά την εκτέλεση κώδικα..... 60
Εικόνα 36: Σημείο με αδιέξοδο στο επίπεδο..... 61
Εικόνα 37: Εισαγωγή HTML elements στον κόσμο του παιχνιδιού..... 61
Εικόνα 38: Δημογραφικά αποτελέσματα..... 110
Εικόνα 39: Δημογραφικά αποτελέσματα σχετικά με το παιχνίδι..... 111
Εικόνα 40: Αποτελέσματα για τη χρηστικότητα..... 112
Εικόνα 41: Αποτελέσματα για την αυτοπεποίθηση..... 113
Εικόνα 42: Αποτελέσματα για την πρόκληση..... 113
Εικόνα 43: Αποτελέσματα για την ικανοποίηση..... 114
Εικόνα 44: Αποτελέσματα για την διασκέδαση..... 114
Εικόνα 45: Αποτελέσματα για την εστίαση της προσοχής..... 115
Εικόνα 46: Αποτελέσματα για τη συνάφεια..... 115
Εικόνα 47: Μαθησιακά αποτελέσματα..... 116
Εικόνα 48: Επίπεδο 1 - HTML στιγμιότυπο θεωρίας..... 123
Εικόνα 49: Επίπεδο 1 - HTML στιγμιότυπο στο παιχνίδι..... 123
Εικόνα 50: Επίπεδο 2 - HTML στιγμιότυπο στο παιχνίδι..... 124
Εικόνα 51: Επίπεδο 3 - HTML στιγμιότυπο στο παιχνίδι..... 124
Εικόνα 52: Επίπεδο 4 - HTML στιγμιότυπο στο παιχνίδι..... 125
Εικόνα 53: Επίπεδο 5 - HTML στιγμιότυπο στο παιχνίδι..... 125
Εικόνα 54: Επίπεδο 5 - HTML στιγμιότυπο εξόδου..... 126
Εικόνα 55: Επίπεδο 6 - CSS στιγμιότυπο με εμπόδια..... 126
Εικόνα 56: Επίπεδο 6 - CSS στιγμιότυπο δραστηριότητας κώδικα..... 127
Εικόνα 57: Επίπεδο 7 - CSS στιγμιότυπο με μηνύματα λάθους..... 127
Εικόνα 58: Επίπεδο 7 - CSS στιγμιότυπο με εμφάνιση κρυφού μηνύματος..... 128
Εικόνα 59: Επίπεδο 8 - CSS στιγμιότυπο με δραστηριότητα κώδικα..... 128
Εικόνα 60: Επίπεδο 8 - CSS στιγμιότυπο προσθήκης πλατφόρμας..... 129
Εικόνα 61: Επίπεδο 9 - CSS στιγμιότυπο παιχνιδιού..... 129
Εικόνα 62: Επίπεδο 10 - CSS στιγμιότυπο θεωρίας..... 130
Εικόνα 63: Επίπεδο 10 - CSS στιγμιότυπο δραστηριότητας κώδικα..... 130

Εικόνα 64: Επίπεδο 10 - CSS στιγμιότυπο μετά την εκτέλεση κώδικα 131
Εικόνα 65: Επίπεδο 11 - CSS στιγμιότυπο παιχνιδιού 131
Εικόνα 66: Επίπεδο 12 - JavaScript στιγμιότυπο παιχνιδιού 132
Εικόνα 67: Επίπεδο 13 - JavaScript στιγμιότυπο παιχνιδιού 132
Εικόνα 68: Επίπεδο 13 - JavaScript στιγμιότυπο προσθήκης πλατφόρμας 133
Εικόνα 69: Επίπεδο 14 - JavaScript στιγμιότυπο παιχνιδιού 133
Εικόνα 70: Επίπεδο 14 - JavaScript στιγμιότυπο ερώτησης πολλαπλής επιλογής 134
Εικόνα 71: Επίπεδο 15 - JavaScript στιγμιότυπο παιχνιδιού 134
Εικόνα 72: Επίπεδο 16 - JavaScript στιγμιότυπο παιχνιδιού 135
Εικόνα 73: Επίπεδο 17 - JavaScript στιγμιότυπο παιχνιδιού με εμπόδιο 135
Εικόνα 74: Επίπεδο 17 - JavaScript στιγμιότυπο παιχνιδιού αφαίρεση εμποδίου 136
Εικόνα 75: Επίπεδο 18 - JavaScript στιγμιότυπο παιχνιδιού 136
Εικόνα 76: Επίπεδο 19 - JavaScript στιγμιότυπο συμπλήρωσης κώδικα 137
Εικόνα 77: Επίπεδο 19 - JavaScript στιγμιότυπο μετά την εκτέλεση κώδικα 137
Εικόνα 78: Τερματισμός παιχνιδιού 138

Κατάλογος Πινάκων

Πίνακας 1: Ο άξονας του σχεδιασμού παιχνιδιού (Game design) 13
Πίνακας 2: Ο άξονας των παιδαγωγικών θεμάτων (Pedagogy) 19
Πίνακας 3: Μοντελοποίηση μαθησιακού περιεχομένου (Learning Content Modeling).. 21
Πίνακας 4: Ο άξονας του σχεδιασμού παιχνιδιού (Game design) 24
Πίνακας 5: Ο άξονας των παιδαγωγικών θεμάτων (Pedagogy) 26
Πίνακας 6: Μοντελοποίηση μαθησιακού περιεχομένου (Learning Content Modeling).. 27
Πίνακας 7: Node.js modules για την υλοποίηση του παιχνιδιού 31
Πίνακας 8: Αντικείμενα και χαρακτήρες του παιχνιδιού 34
Πίνακας 9: Συναρτήσεις αρχείου start-scene.js 84
Πίνακας 10: Συναρτήσεις αρχείου HUD.js 86
Πίνακας 11: Συναρτήσεις αρχείου about-scene.js 87
Πίνακας 12: Συναρτήσεις αρχείου about-scene.js 87
Πίνακας 13: Συναρτήσεις αρχείου platformer-scene.js 88
Πίνακας 14: Συναρτήσεις αρχείου dialog-scene.js 94
Πίνακας 15: Συναρτήσεις αρχείου knowledege-scene.js 98
Πίνακας 16: Συναρτήσεις αρχείου gameover-scene.js 99
Πίνακας 17: Συναρτήσεις της κλάσης Player 101
Πίνακας 18: Συναρτήσεις της κλάσης Fireball 101
Πίνακας 19: Συναρτήσεις της κλάσης Enemy 103
Πίνακας 20: Συναρτήσεις της κλάσης PickUp 104
Πίνακας 21: Συναρτήσεις της κλάσης Door 106
Πίνακας 22: Απαντήσεις σχετικά με παρατηρήσεις και βελτιώσεις 117

Κώδικες

Κώδικας 1: Αρχείο app.js 63
Κώδικας 2: Αρχείο auth.js 65
Κώδικας 3: Αρχείο main.js 67
Κώδικας 4: Αρχείο asyncMiddleware.js 68
Κώδικας 5: Αρχείο secure.js 69
Κώδικας 6: Αρχείο password.js, γραμμές 1 - 45 71
Κώδικας 7: Αρχείο password.js, γραμμές 47 – 81 72
Κώδικας 8: Αρχείο password.js, γραμμές 83 – 117 73
Κώδικας 9: Αρχείο userModel.js 75
Κώδικας 10: Αρχείο index.html 77
Κώδικας 11: Αρχείο signup.html 78
Κώδικας 12: Αρχείο forgot-password.html 79
Κώδικας 13: Αρχείο reset-password.html 80
Κώδικας 14: Αρχείο game.html 81
Κώδικας 15: Αρχείο refreshToken.js 81
Κώδικας 16: Αρχείο game.js 83
Κώδικας 17: Συνάρτηση initRegistry στο αρχείο start-scene.js 85
Κώδικας 18: Επιλογή καιρού, μουσικής και δημιουργία του κόσμου 91
Κώδικας 19: Προσθήκη ήρωα, αντικειμένων στον κόσμο και συγκρούσεων μεταξύ τους 92
Κώδικας 20: Αντικείμενο θεωρίας στο αρχείο content.json 93
Κώδικας 21: Αντικείμενο ερώτησης πολλαπλής επιλογής στο αρχείο content.json 93
Κώδικας 22: Αντικείμενο συντάκτη κώδικα στο αρχείο content.json 94
Κώδικας 23: Συνάρτηση searchForInfiniteLoops στο αρχείο dialog-scene.js 98
Κώδικας 24: Κατασκευαστής της κλάσης Player 100
Κώδικας 25: Κατασκευαστής της κλάσης Enemy 103
Κώδικας 26: Συνάρτηση collected της κλάσης PickUp 105

1 Εισαγωγή

1.1 Πρόβλημα – Σημαντικότητα του θέματος

Τα παιχνίδια ηλεκτρονικών υπολογιστών στις μέρες μας αποτελούν αναπόσπαστο κομμάτι στη ζωή πολλών ανθρώπων και αρκετοί από αυτούς στον προσωπικό τους χρόνο παίζουν βιντεοπαιχνίδια μόνοι τους ή με παρέα. Επίσης. αρκετοί επιστήμονες υποστηρίζουν ότι τα ψηφιακά παιχνίδια θεωρούνται εκτός από ψυχαγωγικά και εκπαιδευτικά εργαλεία υψηλού επιπέδου τα οποία μπορούν να προσφέρουν νέες ευκαιρίες μάθησης (Bellotti et al., 2011).

Ο όρος "παιχνίδι σοβαρού σκοπού", γνωστός προγενέστερα ως "Edutainment", που αποτελείται από την ένωση των λέξεων "education" (εκπαίδευση) και "entertainment" (διασκέδαση), έκανε την εμφάνιση του στις αρχές της δεκαετίας του 90 μαζί με την άνοδο της δημοτικότητας των προσωπικών ηλεκτρονικών υπολογιστών. Ο όρος αυτός αν και αναφέρεται σε όλες τις μορφές μάθησης που συνδυάζονται με τη διασκέδαση έχει επικρατήσει πιο πολύ να χρησιμοποιείται για βιντεοπαιχνίδια με εκπαιδευτικούς στόχους. Μελέτες έχουν δείξει ότι τα παιχνίδια αποτελούν αποδοτικά εκπαιδευτικά εργαλεία και σύμφωνα με έρευνες πιστεύεται ότι τα παιχνίδια σοβαρού σκοπού μελλοντικά θα αποτελούν μέρος της εκπαιδευτικής διαδικασίας (Michael and Chen, 2006).

Η συνεχής εξέλιξη του διαδικτύου έχει αυξήσει την ανάπτυξη εφαρμογών που βασίζονται στις τεχνολογίες ιστού, καθιστώντας έτσι την εκμάθηση προγραμματισμού ιστού αναγκαία. Αρκετές σχολές έχουν εισάγει τη συγκεκριμένη ενότητα στο πρόγραμμα σπουδών, διότι αποτελεί έναν ταχύτατα αναπτυσσόμενο τομέα στην αγορά εργασίας (Liu and Phelps, 2011). Ωστόσο, η ραγδαία εξέλιξη και μεταβολή των τεχνολογιών ιστού, ο μεγάλος όγκος πληροφορίας που καλύπτουν αλλά και το διαφορετικό υπόβαθρο γνώσεων των μαθητών αποτελούν σημαντικές προκλήσεις στην εκμάθησή τους (Yue and Ding, 2004). Στα πλαίσια της διπλωματικής εργασίας θα δημιουργηθεί ένα παιχνίδι που θα προσπαθήσει να συγκεντρώσει τις βασικές έννοιες του προγραμματισμού ιστού, σχετικά με τις τεχνολογίες HTML, CSS και JavaScript.

1.2 Σκοπός – Στόχοι

Σκοπός της διπλωματικής εργασίας είναι η σχεδίαση και ανάπτυξη ενός παιχνιδιού σοβαρού σκοπού. Το εκπαιδευτικό περιεχόμενο του παιχνιδιού αφορά την εκμάθηση και κατανόηση των βασικών εννοιών των HTML, CSS και JavaScript, που αποτελούν τις

βασικές τεχνολογίες ανάπτυξης ιστοτόπων. Στόχος είναι ο παίκτης μέσα από το παιχνίδι να γνωρίσει και να εξοικειωθεί με αυτές τις τρεις τεχνολογίες ιστού.

1.3 Συνεισφορά

Η παρούσα εργασία αφορά το σχεδιασμό και την ανάπτυξη του παιχνιδιού σοβαρού σκοπού “Super Dwarf: Quest for the prototype hammer”. Μέσα από το παιχνίδι αυτό ο παίκτης θα έχει τη δυνατότητα να γνωρίσει και να εξοικειωθεί με τις βασικές έννοιες των HTML, CSS και JavaScript, συνδυάζοντας μάθηση και διασκέδαση. Το βασικό κοινό στο οποίο απευθύνεται το παιχνίδι είναι άτομα που δεν γνωρίζουν τις τρεις αυτές τεχνολογίες. Ωστόσο το παιχνίδι μπορεί να χρησιμοποιηθεί και από άτομα που ήδη γνωρίζουν αυτούς τους τομείς και να αποτελέσει ένα μέσο επιβεβαίωσης των γνώσεων τους αλλά και ψυχαγωγίας. Το παιχνίδι ακολουθεί τη φιλοσοφία των παιχνιδιών πλατφόρμας που είναι αρκετά δημοφιλή και εισάγει το εκπαιδευτικό περιεχόμενο μέσα στη ροή του με την προσθήκη δραστηριοτήτων που αφορούν ερωτήσεις πολλαπλής επιλογής και συμπλήρωση κώδικα.

1.4 Διάρθρωση της μελέτης

Στο Κεφάλαιο 2, παρουσιάζεται η βιβλιογραφική επισκόπηση και γίνεται η συγκριτική ανάλυση παιχνιδιών σοβαρού σκοπού με βάση το πλαίσιο σχεδίασης Educational Games Design Model των Ibrahim & Jaafar (2009).

Στο Κεφάλαιο 3, περιγράφεται η μεθοδολογία που ακολουθήθηκε για την σχεδίαση του παιχνιδιού καθώς και οι τεχνολογίες που θα χρησιμοποιηθούν για την ανάπτυξή του.

Στο Κεφάλαιο 4, πραγματοποιείται η παρουσίαση του παιχνιδιού, η οποία περιέχει την ανάλυση του εκπαιδευτικού περιεχομένου σε κάθε κεφάλαιο και την προβολή των τελικών οθονών.

Στο Κεφάλαιο 5, γίνεται η περιγραφή της υλοποίησης του παιχνιδιού παρουσιάζοντας την αρχιτεκτονική και τις τεχνολογίες του διακομιστή και στη συνέχεια αναλύεται η υποδομή του παιχνιδιού.

Στο Κεφάλαιο 6, πραγματοποιείται η αξιολόγηση του παιχνιδιού και παρουσιάζονται τα αποτελέσματα και τα συμπεράσματα που εξήχθησαν.

Τέλος, στο Κεφάλαιο 7, παρουσιάζονται τα συμπεράσματα που προκύπτουν καθώς και μελλοντικές επεκτάσεις.

2 Βιβλιογραφική Επισκόπηση – Θεωρητικό Υπόβαθρο

2.1 Εισαγωγή

Ο όρος "παιχνίδι σοβαρού σκοπού" αναφέρθηκε για πρώτη φορά το 1970 από τον ερευνητή Clark Abt στο βιβλίο "Serious Games" και είναι ο εξής:

"Τα παιχνίδια σοβαρού σκοπού έχουν ένα συγκεκριμένο και προσεκτικά σχεδιασμένο εκπαιδευτικό στόχο και δεν έχουν ως πρωταρχικό στόχο τη διασκέδαση. Αυτό όμως, δεν σημαίνει ότι τα παιχνίδια σοβαρού σκοπού δεν είναι ή δεν πρέπει να είναι διασκεδαστικά." (Abt, 1970; p. 9)

Σύμφωνα με τον Mike Zyda (2005), ένα παιχνίδι σοβαρού σκοπού είναι ένας διανοητικός διαγωνισμός, ο οποίος παίζεται με τον υπολογιστή και έχει συγκεκριμένους κανόνες, ενώ χρησιμοποιεί τη διασκέδαση έχοντας ως στόχο την εκπαίδευση σε θέματα παιδείας, υγείας, δημόσιας πολιτικής και επικοινωνιακών στόχων. Τα παιχνίδια σοβαρού σκοπού εκτός από την ιστορία, το οπτικοακουστικό υλικό και το λογισμικό, τα οποία αποτελούν κύρια χαρακτηριστικά των βιντεοπαιχνιδιών, εμπεριέχουν και εκπαιδευτικά στοιχεία. Δηλαδή, μέσω των δραστηριοτήτων του παιχνιδιού θα πρέπει να μεταδίδεται η γνώση και να αποκτώνται οι δεξιότητες (Zyda, 2005).

Η νέα γενιά μαθητών και εκπαιδευόμενων έχει μεγαλώσει παίζοντας βιντεοπαιχνίδια, οπότε είναι πιο πιθανό να μαθαίνουν μέσα από αυτά. Τα παιχνίδια παρουσιάζουν ένα θέμα ή ένα πρόβλημα και δίνουν τη δυνατότητα στους παίκτες να αναπτύξουν στρατηγικές, να πάρουν αποφάσεις και να αντιμετωπίσουν το πρόβλημα χωρίς αυτό να έχει συνέπειες στον πραγματικό κόσμο και λαμβάνοντας πίσω σχόλια σχετικά με τις κινήσεις τους. Ο στρατός, οι εκπαιδευτικοί όλων των βαθμίδων, οι επιχειρήσεις, οι Μ.Κ.Ο και ο καλλιτεχνικός χώρος είναι κάποιες από τις κατηγορίες που έχουν δείξει ενδιαφέρον για τα παιχνίδια σοβαρού σκοπού (Michael and Chen, 2006).

Ο σχεδιασμός ενός επιτυχημένου παιχνιδιού σοβαρού σκοπού θα πρέπει να γίνει με τέτοιο τρόπο ώστε να υπάρχει μία ισορροπία μεταξύ του ψυχαγωγικού κομματιού αλλά και του κύριου σκοπού του παιχνιδιού, διότι η διασκέδαση αποτελεί το μέσο με το οποίο μπορεί να επιτευχθεί αυτός ο στόχος (Laamarti, Eid & El Saddik, 2014). Τα περισσότερα παιχνίδια σοβαρού σκοπού βασίζονται στο ότι ο παίκτης πρέπει να αποκτά τη γνώση ανακαλύπτοντας τον κόσμο και επιλύοντας προβλήματα που συναντά σε αυτόν. Όμως, μελέτες έχουν δείξει ότι τα παιχνίδια που έχουν μικρή καθοδήγηση, αν και δημοφιλή έχουν

χαμηλότερη αποτελεσματικότητα από παιχνίδια που βασίζονται στην ισχυρή καθοδήγηση του μαθητή (Bellotti et al., 2011).

2.2 Παιχνίδια σοβαρού σκοπού

Παρακάτω παρουσιάζονται παιχνίδια σοβαρού σκοπού για τη γλώσσα προγραμματισμού JavaScript, αλλά και παιχνίδια τα οποία καλύπτουν κάποια θεματολογία της γλώσσας CSS, η οποία χρησιμοποιείται για την εμφάνιση ενός εγγράφου HTML.

2.2.1 CodeCombat

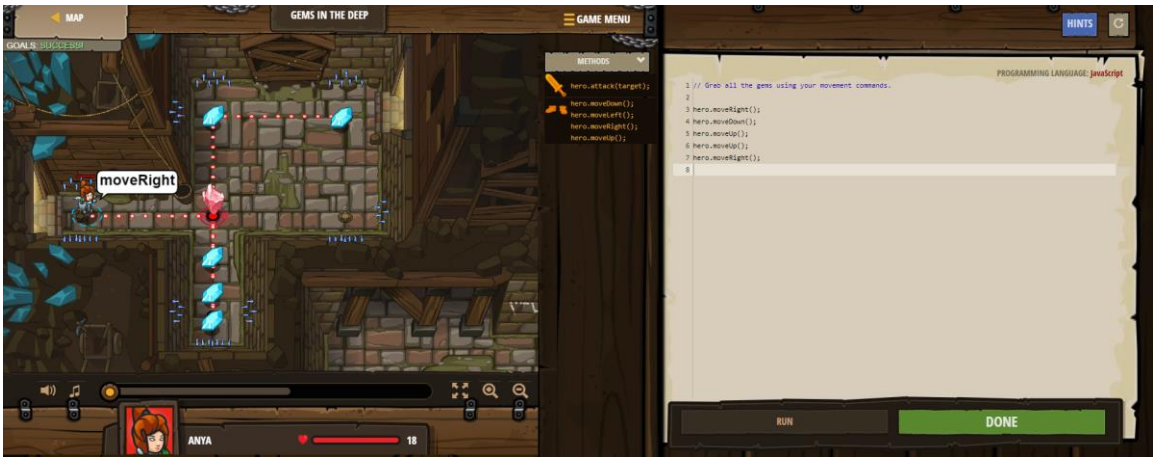
Το CodeCombat (https://codecombat.com/) είναι ένα εκπαιδευτικό παιχνίδι που έχει ως στόχο την εκμάθηση συγκεκριμένων γλωσσών προγραμματισμού και εννοιών της επιστήμης των υπολογιστών. Οι γλώσσες προγραμματισμού που υποστηρίζει είναι η JavaScript, η Python και η C++, η οποία μπορεί να επιλεχθεί μόνο από εγγεγραμμένους χρήστες. Επίσης, είναι διαθέσιμες οι γλώσσες CoffeeScript, Lua και Java, οι οποίες βρίσκονται σε πειραματικό στάδιο. Είναι ένα διαδικτυακό παιχνίδι, το οποίο απευθύνεται σε μαθητές ηλικίας 9 - 16 ετών και είναι μεταφρασμένο σε πάρα πολλές γλώσσες, καθώς και στην ελληνική. Το παιχνίδι λαμβάνει χώρα σε έναν φανταστικό κόσμο, ο οποίος είναι χωρισμένος σε επίπεδα ανάλογα με το θέμα που προσεγγίζει και ο παίκτης μαθαίνει μέσα από αυτό γράφοντας κώδικα. Για να έχει πρόσβαση σε όλα τα επίπεδα ένας παίκτης θα πρέπει να κάνει μηνιαία ή ετήσια συνδρομή.

Αρχικά, ο παίκτης επιλέγει τη γλώσσα προγραμματισμού και τον ήρωα που επιθυμεί για να συμμετέχει στο παιχνίδι (Εικόνα 1). Οι ήρωες διαφέρουν μεταξύ τους ως προς τα στατιστικά στοιχεία τους, για παράδειγμα την υγεία και την ταχύτητα, αλλά και τον εξοπλισμό, ο οποίος βελτιώνει τα χαρακτηριστικά του ήρωα. Για να είναι διαθέσιμες όλες οι γλώσσες προγραμματισμού και όλοι οι ήρωες ο χρήστης θα πρέπει να έχει ενεργή συνδρομή. Αφού γίνει η επιλογή, ξεκινάει το επίπεδο παρουσιάζοντας στο χρήστη τους στόχους που πρέπει να επιτύχει για να το ολοκληρώσει και να προχωρήσει στο επόμενο.

Εικόνα 1: Επιλογή γλώσσας προγραμματισμού και ήρωα στο παιχνίδι CodeCombat

Η περιοχή του επιπέδου χωρίζεται σε δύο βασικά κομμάτια. Αριστερά, παρουσιάζεται ο ήρωας και ο κόσμος μέσα στον οποίο κινείται, για παράδειγμα ένα μπουντρούμι. Στο δεξί μέρος, βρίσκεται ο συντάκτης κειμένου μέσω του οποίου ο παίκτης καθοδηγεί τον ήρωά του γράφοντας κώδικα για να εκτελέσει τους στόχους κάθε επιπέδου. Δίπλα από τον συντάκτη κειμένου, στο αριστερό μέρος, είναι διαθέσιμη μία λίστα με τις εντολές προγραμματισμού που μπορεί να χρησιμοποιήσει ο χρήστης. Όσο ανεβαίνει επίπεδα ο παίκτης, αυτή η λίστα εμπλουτίζεται με περισσότερες εντολές. Αφού γράψει τις εντολές που θεωρεί σωστές μπορεί να τις τρέξει πατώντας το κουμπί “RUN” και να παρακολουθήσει τις κινήσεις του ήρωά του. Όταν εκτελεστούν όλοι οι στόχοι εμφανίζεται το κουμπί “DONE”, το οποίο ο παίκτης πατάει για να δηλώσει την ολοκλήρωση του επιπέδου. Στο τέλος του επιπέδου εμφανίζεται η πρόοδος του παίκτη και τα βραβεία που κέρδισε. Σε περίπτωση συντακτικών σφαλμάτων, αλλά και λάθος εκτέλεσης κώδικα το

παιχνίδι επιστρέφει τα κατάλληλα σχόλια για να βοηθήσει το χρήστη να κάνει τις απαραίτητες διορθώσεις. Επίσης, ο χρήστης μπορεί να πατήσει το κουμπί “HINTS” που βρίσκεται πάνω δεξιά για να ζητήσει επιπλέον βοήθεια (Εικόνα 2).



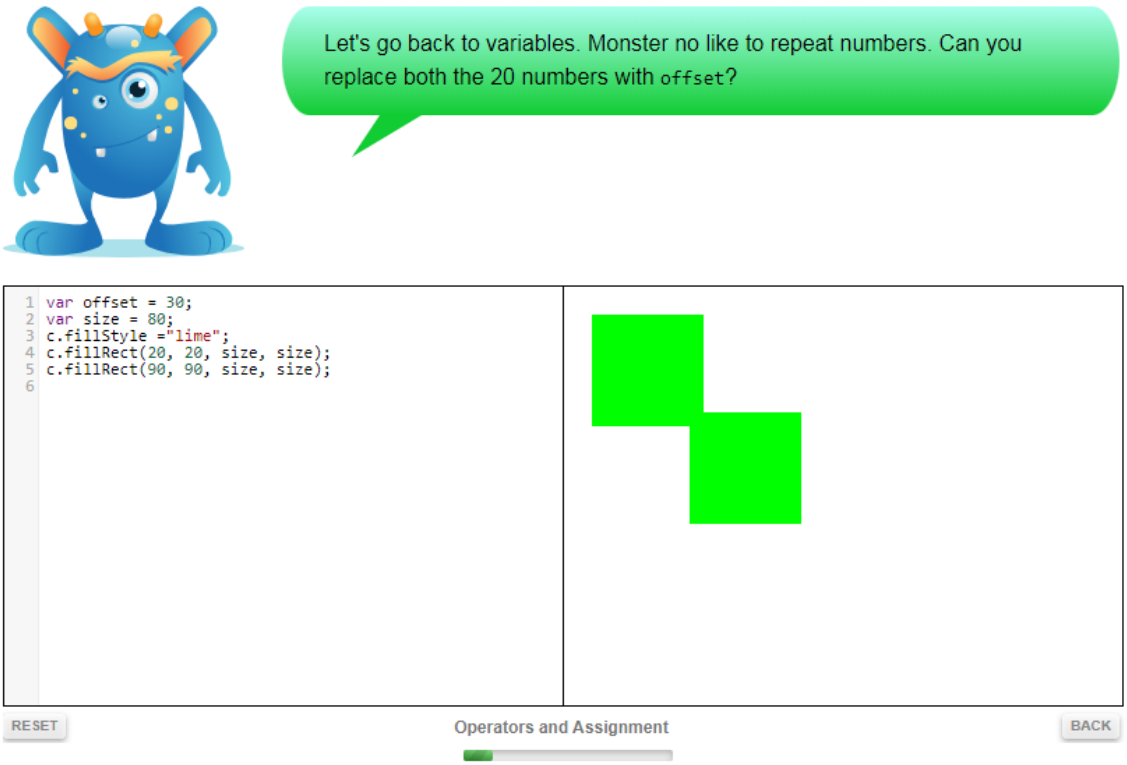
Εικόνα 2: Επίπεδο στο παιχνίδι GameCombat

2.2.2 Crunchzilla

Το Crunchzilla (https://www.crunchzilla.com/) είναι ένα παιχνίδι που έχει ως στόχο την εκμάθηση της γλώσσας προγραμματισμού JavaScript με διαδραστικό τρόπο. Ο χρήστης μπορεί να παίξει το παιχνίδι χρησιμοποιώντας έναν φυλλομετρητή, χωρίς να απαιτείται εγγραφή και συνδρομή σε αυτό καθώς είναι δωρεάν. Η γλώσσα που που χρησιμοποιείται στη γραφική διασύνδεση χρήστη είναι η αγγλική. Το παιχνίδι χωρίζεται σε τρεις μαθησιακές ενότητες με βάση το επίπεδο δυσκολίας, οι οποίες διακρίνονται κατά την επίσκεψη στην ιστοσελίδα (Εικόνα 3). Το Code Monster είναι η πρώτη και πιο βασική ενότητα από τις τρεις και είναι κατάλληλη για μικρές ηλικίες. Αποτελείται από 59 επίπεδα και η δυσκολία τους ανεβαίνει σταδιακά όσο προχωράει ο χρήστης. Στην συνέχεια, ακολουθεί το Code Maven το οποίο απευθύνεται σε εφήβους και ενήλικες. Έχει την ίδια ροή και θεματολογία με το Code Monster αλλά το επίπεδό του είναι πιο προχωρημένο. Τέλος, το Game Maven είναι η ενότητα με το μεγαλύτερο βαθμό δυσκολίας και είναι κατάλληλο για εφήβους και ενήλικες. Τα επίπεδα της είναι 37 στο σύνολο και δίνει την ευκαιρία για εξερεύνηση πιο προηγμένων εννοιών της JavaScript, που αποσκοπούν στη δημιουργία παιχνιδιών, χωρίς να καλύπτει τις βασικές έννοιες, καθώς θεωρεί ότι ο χρήστης γνωρίζει ότι διδάσκεται στο Code Maven.

Εικόνα 3: Οι θεματικές ενότητες του παιχνιδιού Crunchzilla

Αφού ο χρήστης επιλέξει τη μαθησιακή ενότητα που επιθυμεί, οδηγείται στην επόμενη σελίδα. Στο πάνω μέρος της οθόνης εμφανίζεται ο χαρακτήρας και η φούσκα ομιλίας μέσα στην οποία ο χρήστης διαβάζει τις επεξηγήσεις που αφορούν τη θεωρία και τις ερωτήσεις κάθε επιπέδου. Από κάτω υπάρχουν δύο κουτιά, ένα στα αριστερά με το συντάκτη κειμένου, στο οποίο ο παίκτης γράφει τις εντολές του στην JavaScript και ένα στα δεξιά όπου εμφανίζεται το αποτέλεσμα του κώδικα που υπάρχει στον συντάκτη (Εικόνα 4). Σε κάποια επίπεδα υπάρχει ήδη συμπληρωμένος κώδικάς στον συντάκτη και ο παίκτης καλείται να τον επεξεργαστεί ανάλογα με τις οδηγίες που του δίνονται, ενώ σε άλλα καλείται να συντάξει κώδικα από την αρχή. Ο κώδικας εκτελείται όταν σταματήσει να πληκτρολογεί ο παίκτης χωρίς να χρειάζεται να πατήσει κάποιο κουμπί. Σε περίπτωση που υπάρξει κάποιο σφάλμα εμφανίζεται ένα μήνυμα πάνω από την περιοχή του συντάκτη για να βοηθήσει τον παίκτη. Για να προχωράει η ροή του παιχνιδιού ο παίκτης θα πρέπει να κάνει κλικ με το ποντίκι του στη φούσκα ομιλίας για να αλλάζουν οδηγίες. Εάν θέλει να επιστρέψει στην προηγούμενη οδηγία τότε θα πρέπει να χρησιμοποιήσει το κουμπί “BACK”, που βρίσκεται κάτω δεξιά. Η μετάβαση από το ένα επίπεδο στο άλλο δεν απαιτεί να έχουν ολοκληρωθεί συγκεκριμένοι στόχοι και ο παίκτης είναι ελεύθερος να μεταβεί σε όποιο επίπεδο θέλει χωρίς περιορισμούς. Επίσης, μπορεί να βγει από το παιχνίδι και όταν επιστρέψει να το βρει στο επίπεδο που το άφησε σε περίπτωση που χρησιμοποιεί τον ίδιο υπολογιστή.



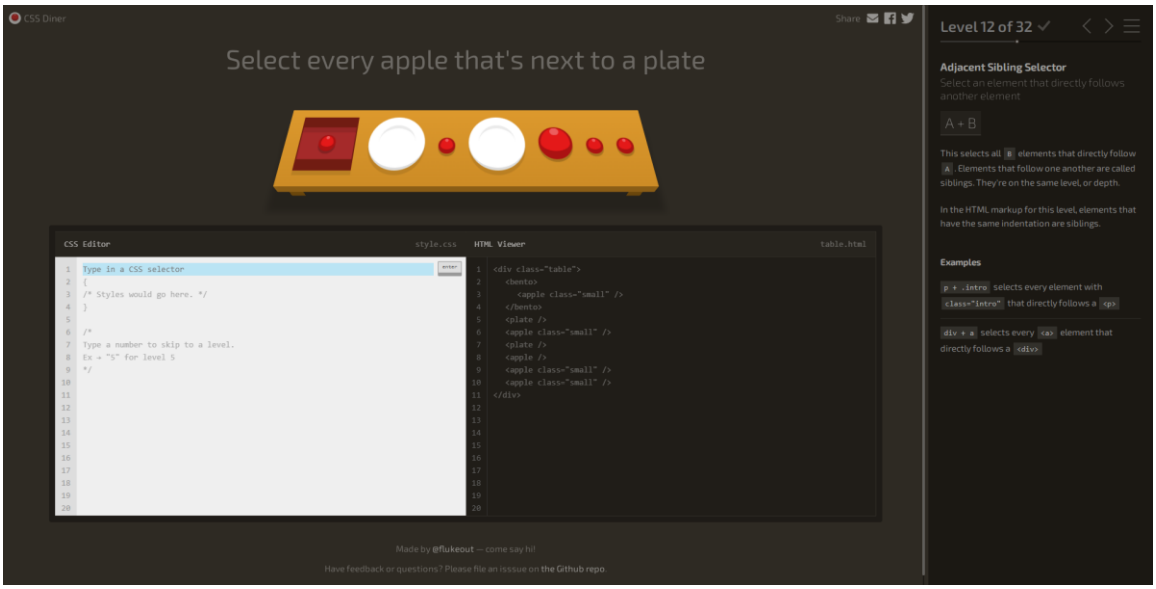
Εικόνα 4: Επίπεδο στο παιχνίδι Crunchzilla

2.2.3 CSS Diner

Το CSS Diner (https://flukeout.github.io/) είναι ένα δισδιάστατο single-player βιντεοπαιχνίδι, που μπορεί κάποιος να παίξει online σε κάποιον από τους δημοφιλείς φυλλομετρητές. Σκοπός του είναι να βελτιώσει τις δεξιότητες του παίκτη σχετικά με τους CSS selectors. Το παιχνίδι είναι δωρεάν και είναι διαθέσιμο στην αγγλική γλώσσα. Αποτελείται από 32 επίπεδα, τα οποία είναι αλληλένδετα μεταξύ τους και διδάσκουν στον παίκτη μία ποικιλία από DOM στοιχεία. Καθώς τα επίπεδα προχωράνε ο βαθμός δυσκολίας ανεβαίνει σταδιακά.

Η βασική οθόνη του παιχνιδιού περιλαμβάνει ένα τραπέζι πάνω στο οποίο είναι τοποθετημένα πιάτα και διάφορα φαγητά (Εικόνα 5). Κάθε φορά ο παίκτης καλείται να επιλέξει κάποιο ή κάποια από τα αντικείμενα που βρίσκονται πάνω στο τραπέζι με τη βοήθεια των CSS selectors. Η πρόκληση εμφανίζεται γραμμένη πάνω από το τραπέζι. Στο δεξί μέρος της οθόνης ο χρήστης μπορεί να διαβάσει τη θεωρία που διδάσκεται σε κάθε επίπεδο, σε συνδυασμό με κάποια παραδείγματα. Όπως φαίνεται στην Εικόνα 5, κάτω από το τραπέζι βρίσκεται ο CSS editor (αριστερά) και ο HTML viewer (δεξιά). Ο παίκτης γράφει τον CSS selector που θεωρεί ότι είναι κατάλληλος για το επίπεδο και πατάει το πλήκτρο “Enter”. Όταν η απάντηση του είναι σωστή μεταβαίνει στο επόμενο επίπεδο, ενώ

όταν είναι λάθος παραμένει στο ίδιο και τρεμοπαίζει ο editor χωρίς να παίρνει κάποια άλλη πληροφορία. Στον HTML viewer παρουσιάζεται ο κώδικας HTML που αναπαριστά το τραπέζι και τα αντικείμενα που υπάρχουν πάνω σε αυτό. Εάν τοποθετηθεί ο κέρσορας του ποντικιού σε ένα HTML στοιχείο τότε δίνεται έμφαση στο αντικείμενο που αντιπροσωπεύει πάνω στο τραπέζι με τη χρήση ενός περιγράμματος. Αντίστοιχα, εάν ο κέρσορας του ποντικιού τοποθετηθεί σε ένα αντικείμενο πάνω στο τραπέζι ο κώδικας HTML γίνεται πιο διακριτός. Ο παίκτης μπορεί να μεταβεί σε όποιο επίπεδο θέλει χωρίς να ολοκληρώσει τα προηγούμενα.



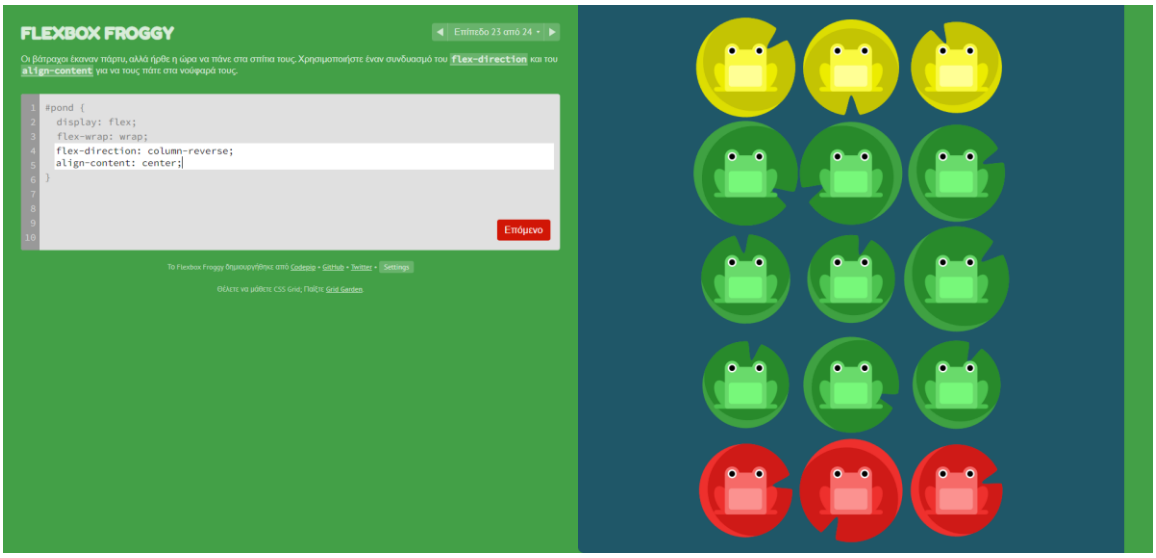
Εικόνα 5: Το παιχνίδι CSS Diner

2.2.4 Flexbox Froggy

Το Flexbox Froggy (https://flexboxfroggy.com/) είναι ένα διαδραστικό διαδικτυακό παιχνίδι, που διανέμεται δωρεάν και είναι μεταφρασμένο σε 41 γλώσσες μαζί με την ελληνική. Σκοπός του είναι ο παίκτης να διδαχθεί και να εξοικειωθεί με τις ιδιότητες του CSS flexbox τοποθετώντας τους βατράχους πάνω στα νούφαρα τους. Συνολικά διαθέτει 24 επίπεδα και η δυσκολία τους ανεβαίνει σταδιακά. Το παιχνίδι είναι κατάλληλο για αρχάριους, αλλά και για έμπειρους web developers που δεν είναι γνώστες του αντικειμένου ή θέλουν να επιβεβαιώσουν τις γνώσεις τους.

Το παιχνίδι διαδραματίζεται σε μία οθόνη, η οποία είναι χωρισμένη σε δύο μέρη (Εικόνα 6). Αριστερά, βρίσκεται το σενάριο του παιχνιδιού με τις οδηγίες που πρέπει να ακολουθήσει ο παίκτης για να ολοκληρωθεί το επίπεδο. Κάτω από αυτές υπάρχει ο συντάκτης κειμένου, όπου ο παίκτης μπορεί να γράψει τους κατάλληλους CSS κανόνες.

Δεξιά, παρουσιάζεται η λίμνη μέσα στην οποία βρίσκονται οι βάτραχοι και τα νούφαρα τους. Κάθε φορά που ο χρήστης γράφει κάποιο κανόνα η δεξιά περιοχή αλληλεπιδρά και οι βάτραχοι κινούνται μέσα στη λίμνη. Όταν οι βάτραχοι τοποθετηθούν στο σωστό νούφαρο, τότε ενεργοποιείται το κουμπί "Επόμενο" και η ροή του παιχνιδιού μπορεί να συνεχίσει. Δεν υπάρχει συγκεκριμένος αριθμός προσπαθειών σε κάθε επίπεδο, οπότε ο παίκτης μπορεί να κάνει όσες δοκιμές επιθυμεί. Το επίπεδο που ολοκληρώνεται χρωματίζεται πράσινο στη λίστα των επιπέδων. Επίσης, ο παίκτης μπορεί να επιλέξει όποιο επίπεδο θέλει από τη λίστα, καθώς δεν υπάρχει κάποιος αυστηρός περιορισμός.



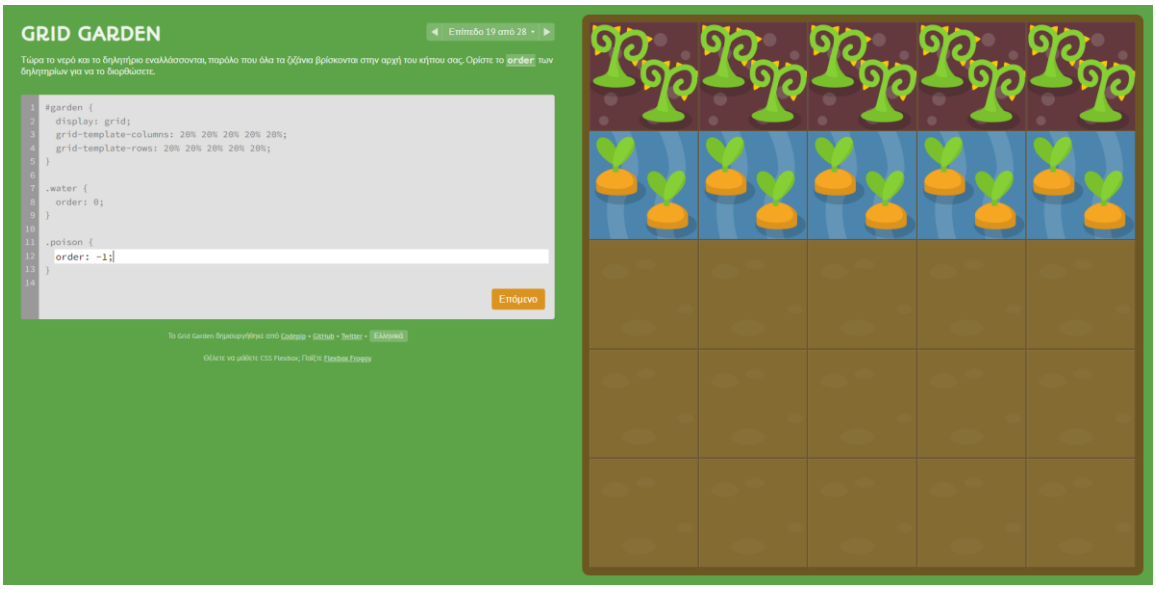
Εικόνα 6: Το παιχνίδι Flexbox Froggy

2.2.5 Grid Garden

Το Grid Garden (https://cssgridgarden.com/) είναι ένα παιχνίδι που μπορεί να παιχτεί ελεύθερα στο διαδίκτυο και έχει ως στόχο να διδάξει τις βασικές ιδιότητες του CSS grid layout. Δημιουργήθηκε από τον προγραμματιστή που έφτιαξε το Flexbox Froggy, οπότε ακολουθεί παρόμοιο τρόπο διδασκαλίας και χρήσης. Συνολικά έχει μεταφραστεί σε 31 γλώσσες μαζί με την ελληνική. Σε κάθε επίπεδο ο παίκτης συμπληρώνει CSS κανόνες σχετικούς με το CSS Grid, είτε για να ποτίσει τα καρότα, είτε για να σκοτώσει τα ζιζάνια που βρίσκονται στον κήπο του. Το παιχνίδι αποτελείται από 28 επίπεδα.

Όπως και στο Flexbox Froggy, αριστερά βρίσκονται οι οδηγίες κάθε επιπέδου και ο CSS συντάκτης και δεξιά αναπαριστάται ο κήπος με τα καρότα (Εικόνα 7). Όταν εκτελείται ο κώδικας που συμπληρώνει ο χρήστης στον συντάκτη, ο κήπος στα δεξιά αλληλεπιδρά και φαίνονται οι αλλαγές. Αφού εκπληρωθεί ο στόχος του επιπέδου

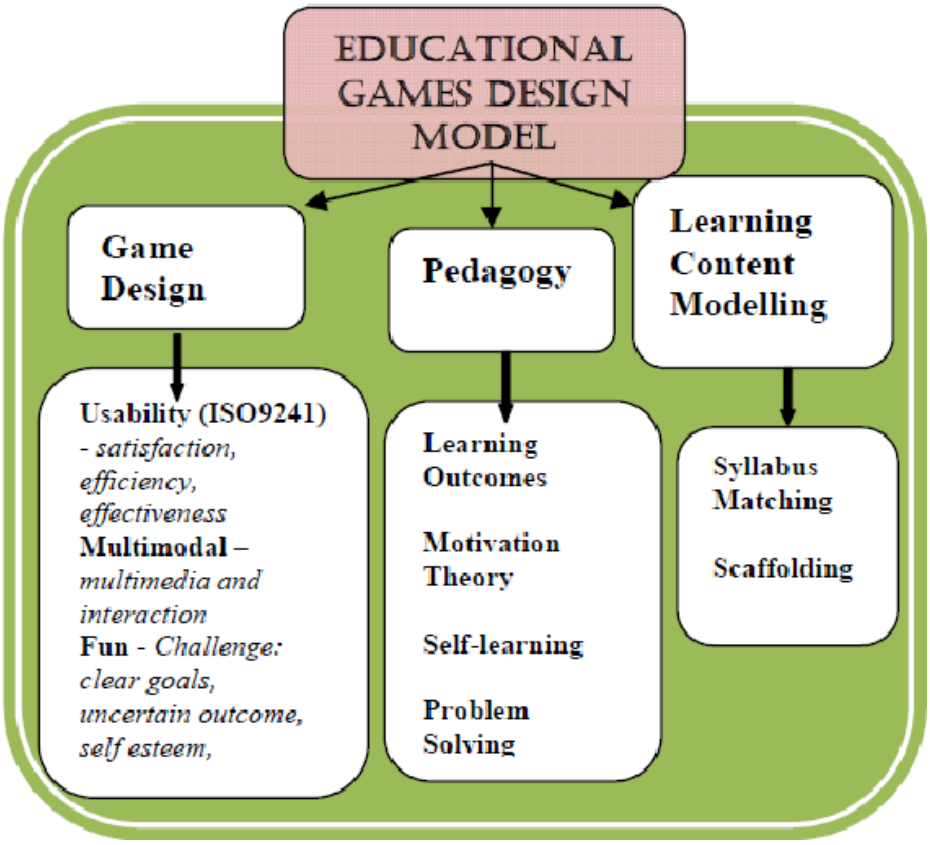
ενεργοποιείται το κουμπί “Επόμενο” και η μετάβαση στο επόμενο επίπεδο. Ο παίκτης μπορεί να παραλείψει κάποιο επίπεδο και να επιλέξει κάποιο άλλο από τη λίστα. Τα επίπεδα που έχουν χρωματιστεί με πράσινο χρώμα είναι αυτά που έχουν ολοκληρωθεί.



Εικόνα 7: Το παιχνίδι Grid Garden

2.3 Πλαίσιο σχεδίασης

Για την ανάπτυξη και την υλοποίηση παιχνιδιών σοβαρού σκοπού έχουν δημιουργηθεί αρκετά πλαίσια σχεδίασης και αξιολόγησης, τα οποία θέτουν συγκεκριμένα κριτήρια και άξονες τους οποίους πρέπει να ακολουθήσουν οι σχεδιαστές των παιχνιδιών. Ένα από αυτά τα πλαίσια είναι το Educational Games Design Model που παρουσιάστηκε από τους Ibrahim & Jaafar (2009) (Εικόνα 8). Οι τρεις βασικοί του άξονες είναι ο σχεδιασμός του παιχνιδιού (Game Design), οι παιδαγωγικοί στόχοι (Pedagogy) και η μοντελοποίηση του μαθησιακού περιεχομένου (Learning Content Modeling) (Ibrahim & Jaafar, 2009).



Εικόνα 8: Πλαίσιο σχεδίασης Educational Games Design Model (Ibrahim & Jaafar, 2009)

Ο άξονας του σχεδιασμού παιχνιδιού περιλαμβάνει την ευχρηστία, την πολυτροπικότητα και την ψυχαγωγία. Η ευχρηστία θα πρέπει να ελέγχεται με βάση το πρότυπο ISO 9241 (Pinelle, D. and N. Wong, 2008) και αφορά την αποτελεσματικότητα και την αποδοτικότητα του παιχνιδιού, καθώς και την ικανοποίηση του παίκτη. Η αποτελεσματικότητα συνδέεται με την ακρίβεια και την επίτευξη των στόχων που έχουν τεθεί από τον χρήστη, η αποδοτικότητα σχετίζεται με τους πόρους που δαπανώνται για την επίτευξη των στόχων αυτών, ενώ η ικανοποίηση σχετίζεται με τη διάθεση των χρηστών. Η πολυτροπικότητα είναι ο συνδυασμός των πολυμέσων, όπως είναι το κείμενο, τα γραφικά, ο ήχος, το βίντεο και οι κινούμενες εικόνες. Η διάδραση αναφέρεται στον τρόπο με τον οποίο αλληλεπιδρούν οι παίκτες με το παιχνίδι και και λαμβάνουν άμεση ανατροφοδότηση από αυτό. Η ψυχαγωγία αποτελεί το κυριότερο χαρακτηριστικό, καθώς παρέχει κίνητρο και αφοσίωση στους παίκτες.

Ο δεύτερος άξονας του πλαισίου, οι παιδαγωγικοί στόχοι, επικεντρώνεται στον βαθμό στον οποίο το παιχνίδι ανταποκρίνεται στα μαθησιακά αποτελέσματα. Για να επιτευχθεί αυτό, ο σχεδιασμός του παιχνιδιού θα βασιστεί στα τρία πρώτα επίπεδα της

ταξινόμησης Bloom, δηλαδή στη γνώση, την κατανόηση και την εφαρμογή. Επίσης, το μοντέλο υποστηρίζει την αυτόνομη μάθηση και την αξιολόγηση μέσω των κινήτρων που παρέχει η θεωρία. Ένα ακόμη χαρακτηριστικό είναι η επίλυση προβλημάτων μέσα από την οποία οι παίκτες αναπτύσουν τις δεξιότητές τους (Ibrahim & Jaafar, 2009).

Η μοντελοποίηση του μαθησιακού περιεχομένου αποτελεί ένα σημαντικό παράγοντα, αφού το παιχνίδι σχεδιάζεται για να βοηθήσει τους παίκτες να μάθουν μόνοι τους τα θέματα που πραγματεύεται. Το μαθησιακό περιεχόμενο θα πρέπει να είναι όσο γίνεται πιο κατάλληλο προκειμένου να επιτευχθούν οι μαθησιακοί στόχοι (Ibrahim & Jaafar, 2009).

2.4 Ανάλυση και σύγκριση παιχνιδιών σοβαρού σκοπού

Με βάση το πλαίσιο σχεδίασης Educational Games Design Model που αναλύθηκε στην ενότητα 2.3 θα γίνει αξιολόγηση των παιχνιδιών που παρουσιάστηκαν στην αρχή του κεφαλαίου. Στον Πίνακα 1 αναλύονται τα παιχνίδια με βάση τον πρώτο παράγοντα του σχεδιασμού παιχνιδιού. Στον Πίνακα 2 παρουσιάζονται τα παιχνίδια με βάση τον άξονα των παιδαγωγικών στόχων και στον Πίνακα 3 με βάση τη μοντελοποίηση του μαθησιακού περιεχομένου.

Πίνακας 1: Ο άξονας του σχεδιασμού παιχνιδιού (Game design)

Παιχνίδια	**Χαρακτηριστικά**
	Ευχρηστία (Usability)
	Ικανοποίηση (Satisfaction)
CodeCombat	Όταν ολοκληρώνεται ένα επίπεδο παρουσιάζονται στον παίκτη οι πόντοι εμπειρίας που συγκέντρωσε σε αυτό, τα πετράδια που μάζεψε στο επίπεδο καθώς και άλλα αντικείμενα που μπορεί να κέρδισε, όπως για παράδειγμα εξοπλισμό, δίνοντάς του έτσι την ικανοποίηση ότι η προσπάθειά του ανταμείβεται. Επίσης, έχοντας κάνει λογαριασμό στο παιχνίδι μπορεί να βλέπει σε ποιο επίπεδο βρίσκεται, σύμφωνα με τους συνολικούς πόντους εμπειρίας του, τα διαμάντια που έχει στην κατοχή του, και έχει τη δυνατότητα να παίξει όποτε διαθέτει ελεύθερο χρόνο. Το παιχνίδι προσφέρει στον παίκτη επιπλέον επιτεύγματα που έχουν σχέση με τη συνολική του απόδοση στο παιχνίδι ενισχύοντας έτσι το αίσθημα ικανοποίησης του.
Crunchzilla	Ο παίκτης ικανοποιείται καθώς βλέπει τις εντολές κώδικα που γράφει στον συντάκτη να παίρνουν μορφή με την εμφάνιση

	σχημάτων στην περιοχή δεξιά του συντάκτη. Επίσης, η ικανοποίηση του ενισχύεται βλέποντας την μπάρα προόδου να γεμίζει καθώς προχωράει στα επίπεδα. Δεν είναι εμφανές με κάποιο άλλο τρόπο πόσα επίπεδα έχει ολοκληρώσει, ειδικά εάν ακολουθήσει τυχαία σειρά ή παραλείψει κάποιο από αυτά.
CSS Diner	Η επιλογή και η απομάκρυνση των αντικειμένων από το τραπέζι δίνουν χαρά στον παίκτη καθώς επιβεβαιώνεται ότι η προσπάθεια του είναι σωστή. Επίσης, αφού ολοκληρωθεί το επίπεδο μαρκάρεται με ένα πράσινο εικονίδιο στη λίστα των επιπέδων. Η πρόοδος του παίκτη αποθηκεύεται μόνο στο συγκεκριμένο υπολογιστή που ξεκίνησε το παιχνίδι και μελλοντικά θα χαθεί.
Flexbox Froggy	Η ικανοποίηση του παίκτη ενισχύεται καθώς βλέπει τους βατράχους να τοποθετούνται στα σωστά νούφαρα της λίμνης μετά την εκτέλεση των εντολών CSS που έχει πληκτρολογήσει. Όταν οι εντολές του είναι σωστές ενεργοποιείται το κουμπί "Επόμενο" και μαρκάρετε με πράσινο χρώμα το επίπεδο στην αντίστοιχη λίστα. Η πρόοδος του δεν αποθηκεύεται σε κάποια βάση δεδομένων, αλλά τοπικά στον φυλλομετρητή που παίζει το παιχνίδι.
Grid Garden	Ο παίκτης βλέπει τα αποτελέσματα των πράξεων του άμεσα στην δεξιά περιοχή της οθόνης. Σε περίπτωση που οι απαντήσεις του είναι σωστές ενεργοποιείται το κουμπί "Επόμενο" για να συνεχίσει στο επόμενο επίπεδο. Στη λίστα των επιπέδων τα ολοκληρωμένα επίπεδα φαίνονται με πράσινο χρώμα.
	Αποδοτικότητα (Efficiency)
CodeCombat	Σε κάθε επίπεδο παρουσιάζονται με μορφή απλού κειμένου οι οδηγίες και οι στόχοι που απαιτείται να πραγματοποιήσει ο παίκτης. Επίσης, είναι εμφανής η λίστα εντολών που γνωρίζει και μπορεί να χρησιμοποιήσει. Κάθε εντολή συνοδεύεται με την επεξήγησή της, που μπορεί να την διαβάσει πατώντας πάνω της.
Crunchzilla	Η θεωρία του παιχνιδιού παρουσιάζεται με απλό και περιεκτικό τρόπο μέσω κειμένου και μέσω παραδειγμάτων στον συντάκτη που εφαρμόζουν τη θεωρία. Το παιχνίδι βρίσκεται δωρεάν στο διαδίκτυο και δεν απαιτεί εγγραφή.
CSS Diner	Η θεωρία εμφανίζεται με λιτό και κατανοητό κείμενο. Το παιχνίδι βοηθάει τον παίκτη με την προσθήκη παραδείγματος σε κάθε επίπεδο. Μπορεί να παιχτεί δωρεάν με πρόσβαση στους δημοφιλείς περιηγητές.
Flexbox Froggy	Σε κάθε επίπεδο, προβάλλεται η θεωρία του παιχνιδιού με τη μορφή απλού κειμένου. Ο παίκτης έχει τη δυνατότητα να δοκιμάσει αυτά που διδάσκεται χωρίς περιορισμούς, να βλέπει άμεσα τα αποτελέσματα και να κατανοεί πιο εύκολα τις έννοιες.

Grid Garden	Ακολουθεί την ίδια λογική με το Flexbox Froggy και βρίσκεται και αυτό δωρεάν στο διαδίκτυο.
	Αποτελεσματικότητα (Effectiveness)
CodeCombat	Η θεωρία που προβάλλεται σε κάθε επίπεδο εισάγει κάθε φορά μία νέα έννοια. Όλες οι προηγούμενες έννοιες που έχουν διδαχθεί συνεχίζουν να χρησιμοποιούνται από τον παίκτη. Η λίστα με τις εντολές που έχει διδαχθεί και η προβολή της σημασίας τους είναι διαθέσιμη συνέχεια. Ο έλεγχος των απαντήσεων, αλλά και η συχνή ανατροφοδότηση βοηθούν στην καλύτερη κατανόηση του χρήστη και στην εύρεση των λαθών του.
Crunchzilla	Κάθε επίπεδο επικεντρώνεται σε μία έννοια η οποία όμως παρουσιάζεται μέσα από αρκετά παραδείγματα. Οι εντολές που γράφει ο παίκτης στον κειμενογράφο παίρνουν μορφή στο viewer που βρίσκεται δεξιά του. Σε περίπτωση συντακτικού λάθους εμφανίζεται το σφάλμα που προέκυψε δίνοντας έτσι βοήθεια στον παίκτη. Το παιχνίδι δεν κάνει κανένα έλεγχο στις απαντήσεις του παίκτη για να βεβαιωθεί ότι οι στόχοι που έχουν τεθεί ολοκληρώνονται και μπορεί να προχωρήσει χωρίς να απαιτείται η συμπλήρωση κάποιας απάντησης.
CSS Diner	Η προβολή μιας και μόνο έννοιας σε κάθε επίπεδο βοηθάει στην αποτελεσματικότερη εκμάθηση της. Επίσης, ο παίκτης μπορεί να ανατρέξει σε προηγούμενα επίπεδα και να ξαναδιαβάσει τη θεωρία χωρίς όμως να έχει τη δυνατότητα να δει την προηγούμενη απάντησή του. Σε περίπτωση που δεν δώσει σωστή απάντηση το παιχνίδι δεν βοηθάει τον παίκτη να καταλάβει που βρίσκεται το λάθος.
Flexbox Froggy	Κάθε επίπεδο επικεντρώνεται σε μία έννοια για την καλύτερη κατανόησή της. Υπάρχουν όμως και επίπεδα όπου καλούν τον παίκτη να χρησιμοποιήσει έννοιες που έχουν διδαχθεί σε κάποιο από τα προηγούμενα. Οποιαδήποτε στιγμή ο παίκτης μπορεί να ανατρέξει σε προηγούμενα επίπεδα για να διαβάσει τη θεωρία και να δει την απάντηση που έχει δώσει. Με την άμεση αλληλεπίδραση με το δεξί μέρος της οθόνης ο παίκτης παίρνει την κατάλληλη ανατροφοδότηση ώστε να οδηγηθεί στην σωστή απάντηση.
Grid Garden	Σε κάθε επίπεδο εισάγεται μία καινούργια έννοια ώστε η θεωρία να είναι πιο κατανοητή από τον παίκτη. Κάθε φορά που ο παίκτης γράφει κάτι στο συντάκτη υπάρχει αλληλεπίδραση και παρουσιάζει στον παίκτη τις συνέπειες της απάντησής του. Όπως και στο Flexbox Froggy μπορεί να επισκεφθεί οποιοδήποτε επίπεδο θέλει ανα πάσα στιγμή.
	Πολυτροπικότητα (Multimodal)

	Πολυμέσα και αλληλεπιδράσεις (multimedia and interaction)
CodeCombat	Ο κόσμος και τα επίπεδα που λαμβάνει χώρα το παιχνίδι είναι δύο διαστάσεων. Περιέχει πλούσιο γραφικό περιβάλλον ακολουθώντας τη φιλοσοφία ενός RPG παιχνιδιού και δίνει τη δυνατότητα στον παίκτη να επιλέξει το χαρακτήρα που θέλει μέσα από μία σειρά διαθέσιμων avatars. Ο παίκτης καλείται να μαζέψει διαμάντια, να αντιμετωπίσει εχθρούς, να αποφύγει και να καταστρέψει εμπόδια. Επίσης, έχει τη δυνατότητα να παρακολουθήσει με μορφή κινούμενων εικόνων το αποτέλεσμα του κώδικα που έγραψε. Η προσθήκη ηχητικών εφέ κατά την αλληλεπίδραση με αντικείμενα και εχθρούς, αλλά και για την ανατροφοδότηση του παίκτη είτε σε περίπτωση επιτυχίας είτε αποτυχίας διατηρούν το ενδιαφέρον του χρήστη σε υψηλά επίπεδα. Επιπλέον, η ύπαρξη μουσικής στο υπόβαθρο, που αλλάζει ανάλογα με την περίσταση, εισάγει τον παίκτη πιο βαθιά στον κόσμο του παιχνιδιού.
Crunchzilla	Το γραφικό περιβάλλον του παιχνιδιού είναι φτωχό καθώς περιέχει απλά μία εικόνα, που αντιπροσωπεύει το βασικό χαρακτήρα και τη φούσκα ομιλίας, η οποία περιέχει τη θεωρία και τις οδηγίες για τον παίκτη. Η παρουσία του HTML viewer, όπου εμφανίζεται το αποτέλεσμα από τη συγγραφή κώδικα ενισχύει σημαντικά το ενδιαφέρον του παίκτη. Ωστόσο, η έλλειψη ανατροφοδότησης και ελέγχου των απαντήσεων αποτελούν ένα σημαντικό πρόβλημα στην αλληλεπίδραση με τον παίκτη. Τελος, η προσθήκη γραφικών, ηχητικών εφέ και επιβράβευσης, που τώρα απουσιάζουν είναι παράγοντες που θα μπορούσαν να αυξήσουν το ενδιαφέρον του παίκτη.
CSS Diner	Το παιχνίδι διαθέτει ένα λιτό και καλαίσθητο γραφικό περιβάλλον. Η ενσωμάτωση εικόνων όπως πιάτα, διάφορα φρούτα και λαχανικά για την αναπαράσταση της HTML, αλλά και η προσθήκη κινούμενων εικόνων που δείχνουν στον παίκτη ποια αντικείμενα τον ενδιαφέρουν σε κάθε επίπεδο διατηρούν το ενδιαφέρον του αμείωτο. Διακριτή είναι η μη ισχυρή ανατροφοδότηση προς τον παίκτη καθώς δεν υπάρχει βοήθεια όταν κάνει κάποιο λάθος. Τέλος, αν το παιχνίδι είχε κάποιο μουσικό υπόβαθρο θα βοηθούσε τον παίκτη να επικεντρωθεί στην επίλυση των προβλημάτων.
Flexbox Froggy	Το παιχνίδι παρουσιάζει ένα πολύχρωμο περιβάλλον και διαθέτει αντικείμενα (βάτραχοι και νούφαρα) με τα οποία ο παίκτης αλληλεπιδρά με την εκτέλεση του κώδικα που συμπληρώνει στον συντάκτη. Η επιτυχημένη απάντηση ξεχωρίζει με την ενεργοποίηση του κουμπιού “Επόμενο”, αφού ο παίκτης τοποθετήσει όλους τους βατράχους στα σωστά νούφαρα. Μειονέκτημά του αποτελεί η έλλειψη μουσικής και ηχητικών εφέ.

Grid Garden	Η προβολή εικόνων, όπως καρότα και ζιζάνια, πλαισιώνουν την πρόκληση που έχει να αντιμετωπίσει ο παίκτης σε κάθε επίπεδο αυξάνοντας το αίσθημα ενδιαφέροντος του παίκτη προς το παιχνίδι. Με την προσθήκη μουσικής και ήχων θα μπορούσε να ενισχυθεί το ενδιαφέρον του παίκτη ως προς αυτό.
	<u>Ψυχαγωγία - Πρόκληση (Fun - Challenge)</u>
	Ξεκάθαροι στόχοι (Clean goals)
CodeCombat	Σε κάθε επίπεδο παρουσιάζονται ξεκάθαρα οι στόχοι του όταν αυτό ξεκινάει, αλλά και στην περιοχή των οδηγιών. Κάποιοι από τους στόχους που εμφανίζονται είναι η συλλογή διαμαντιών, η αποφυγή παγίδων και η αντιμετώπιση εχθρών.
Crunchzilla	Το παιχνίδι παρουσιάζει τους στόχους του στο επίπεδο μέσω της φούσκας ομιλίας του κεντρικού χαρακτήρα. Ο παίκτης θα πρέπει κατανοώντας αυτά που διαβάζει να σχεδιάσει αυτά που του ζητούνται κάθε φορά.
CSS Diner	Ο στόχος που ορίζεται σε κάθε επίπεδο είναι ο παίκτης να επιλέξει τα κατάλληλα αντικείμενα πάνω από το τραπέζι με τον κατάλληλο CSS selector κάθε φορά. Εμφανίζεται κάθε φορά πάνω από το τραπέζι.
Flexbox Froggy	Το παιχνίδι έχει ως βασικό στόχο σε όλα τα επίπεδα να τοποθετηθούν οι βάτραχοι στα σωστά νούφαρα της λίμνης με την κατάλληλη σύνταξη CSS. Αυτό φαίνεται κάθε φορά στο κείμενο οδηγιών.
Grid Garden	Ο κύριος στόχος του παιχνιδιού είναι με την κατανόηση του CSS grid ο παίκτης να ποτίσει τα καρότα στον κήπο του και να εξολοθρεύσει τα ζιζάνια, συμπληρώνοντας τον κατάλληλο κώδικα CSS.
	Αβέβαιο αποτέλεσμα (Uncertain outcome)
CodeCombat	Στο παιχνίδι είναι απαραίτητο ο παίκτης να βρει τη σωστή λύση και να δει το αποτέλεσμα αυτής για να συνεχίσει στα επόμενα επίπεδα.
Crunchzilla	Δεν υπάρχει κάποια δέσμευση για να προχωρήσει ο παίκτης στο επόμενο επίπεδο. Η συμπλήρωση λανθασμένης απάντησης αλλά και η έλλειψη αυτής δεν αποτελούν εμπόδιο στο να μεταβεί στο επόμενο επίπεδο, καθώς και στο να επιλέξει τα επίπεδα με όποια σειρά θέλει.
CSS Diner	Ο παίκτης θα πρέπει να συμπληρώσει τον κατάλληλο CSS selector για να πάει στο επόμενο επίπεδο. Ωστόσο, μπορεί να παραλείψει κάποιο από αυτά μέσω της λίστας επιπέδων.

Flexbox Froggy	Για να μεταβεί στο επόμενο επίπεδο σύμφωνα με τη ροή του παιχνιδιού και να το ολοκληρώσει ο παίκτης θα πρέπει να συμπληρώσει τη σωστή λύση. Παρόλα αυτά, μπορεί να επιλέξει κάποιο επίπεδο χωρίς να έχει ολοκληρώσει τα προηγούμενα.
Grid Garden	Με τη σωστή συγγραφή κανόνα CSS ο παίκτης ολοκληρώνει το επίπεδο και προχωράει στο επόμενο. Η παράλειψη κάποιου επιπέδου είναι εφικτή.
	Αυτοπεποίθηση (Self esteem)
CodeCombat	Η κίνηση του χαρακτήρα μέσω του κώδικα που έχει συντάξει ο παίκτης ενισχύει σημαντικά την αυτοπεποίθηση του. Επίσης, σε αυτό συμβάλλουν η παρουσίαση των αντικειμένων που συλλέγει σε κάθε επίπεδο, η αύξηση των πόντων της εμπειρίας του και η δυνατότητα επανεκκίνησης του επιπέδου χωρίς κάποια ποινή.
Crunchzilla	Η ύπαρξη του HTML viewer μέσω του οποίου παίρνουν ζωή οι εντολές που γράφει ο παίκτης αποτελεί τον βασικό παράγοντα ενίσχυσης της αυτοπεποίθησης του παίκτη. Είναι φανερή η έλλειψη κάποιου σκορ, αλλα και μηνυμάτων που θα μπορούσαν να βοηθήσουν τον παίκτη.
CSS Diner	Το μαρκάρισμα των επιπέδων ως ολοκληρωμένα, η απομάκρυνση των αντικειμένων από το τραπέζι και ο μη περιορισμένος αριθμός προσπαθειών αυξάνουν την αυτοπεποίθηση του. Ωστόσο, η προσθήκη κάποιας επιπλέον επιβράβευσης και μηνυμάτων για την καθοδήγηση της βέλτιστης λύσης θα αποτελούσαν ένα σημαντικό πλεονέκτημα.
Flexbox Froggy	Ο παίκτης έχει τη δυνατότητα να δοκιμάσει πολλές φορές την απάντησή του μέχρι να βρει τη λύση. Αυτό και η εμφάνιση της επιτυχούς ολοκλήρωσης του επιπέδου στην αντίστοιχη λίστα βοηθούν στην αύξηση της αυτοπεποίθησής του. Εμφανής είναι και σε αυτό το παιχνίδι η απουσία μηνυμάτων προς τον παίκτη και στην περίπτωση επιτυχίας αλλά και αποτυχίας.
Grid Garden	Ο χρωματισμός των ολοκληρωμένων επιπέδων με πράσινο χρώμα βοηθάει στην αύξηση της αυτοπεποίθησης του παίκτη. Επίσης, ο χρήστης έχει την ελευθερία να προσπαθήσει πολλές φορές μέχρι να βρει την σωστή απάντηση.

Πίνακας 2: Ο άξονας των παιδαγωγικών θεμάτων (Pedagogy)

	Μαθησιακά αποτελέσματα (Learning Outcomes)
CodeCombat	Με αυτό το παιχνίδι ο παίκτης υποστηρίζεται στην εκμάθηση βασικών αρχών και δομών του προγραμματισμού στην γλώσσα που έχει επιλέξει. Οι μεταβλητές, οι εντολές if, οι βρόγχοι επανάληψης και οι αντικειμενοστρεφείς έννοιες είναι κάποια από τη θεματολογία που διδάσκεται.
Crunchzilla	Μέσα από το παιχνίδι ο χρήστης αποκτά γνώσεις πάνω στη γλώσσα προγραμματισμού JavaScript. Ο σχεδιασμός κουτιών, τα χρώματα, οι μεταβλητές, οι βρόχοι, οι προϋποθέσεις, και οι συναρτήσεις αποτελούν κάποιες από τις έννοιες που μαθαίνει.
CSS Diner	Σε αυτό το παιχνίδι ο παίκτης θα μάθει και θα εξασκήσει τις γνώσεις του πάνω στους CSS selectors.
Flexbox Froggy	Ο παίκτης εισάγεται στις βασικές έννοιες του CSS flexbox για την τοποθέτηση στοιχείων σε μία ιστοσελίδα.
Grid Garden	Ο χρήστης παίζοντας θα μάθει τις βασικές αρχές του CSS Grid για τη δημιουργία διατάξεων σε μία ιστοσελίδα.
	Θεωρία παρακίνησης (Motivation Theory)
CodeCombat	Ο παίκτης σε κάθε επίπεδο μαθαίνει νέες εντολές ή επεκτείνει τις γνώσεις του πάνω σε αυτές που έχει ήδη μάθει. Όλα τα επίπεδα του παιχνιδιού εκτός από το πρώτο είναι κλειδωμένα. Ο παίκτης πρέπει να ολοκληρώνει ένα επίπεδο για να προχωρήσει στο επόμενο, πράγμα που τον παρακινεί να παίζει και να μαθαίνει μέσα από αυτό.
Crunchzilla	Σε κάθε επίπεδο εισάγονται νέες έννοιες της Javascript, το οποίο παρακινεί τον παίκτη να θέλει να προχωρήσει για να μάθει περισσότερα. Η θεωρία εξελίσσεται σταδιακά και οι έννοιες που έχει ήδη μάθει σε προηγούμενα επίπεδα χρησιμοποιούνται σε συνδυασμό με τα νέα στοιχεία που παρουσιάζονται στο επίπεδο.
CSS Diner	Ο χρήστης σε κάθε επίπεδο γνωρίζει και μία νέα έννοια σχετική με τους CSS selectors. Το ενδιαφέρον του παίκτη και η θέληση του να ολοκληρώσει το παιχνίδι ενισχύεται διότι όσο τα επίπεδα προχωράνε δυσκολεύουν και παρουσιάζονται πιο σύνθετες έννοιες.
Flexbox Froggy	Η παρουσίαση μιας νέας ιδιότητας του CSS flexbox σε κάθε επίπεδο παρακινεί τον παίκτη να θέλει να ολοκληρώσει το παιχνίδι για να μάθει περισσότερα. Καθώς εξελίσσεται το παιχνίδι ο παίκτης χρησιμοποιεί συνδυαστικά παλιά και καινούργια στοιχεία που διδάσκεται.

Grid Garden	Κάθε επίπεδο διδάσκει ένα καινούργιο στοιχείο του CSS grid. Η σταδιακή και ομαλή εξέλιξη του παιχνιδιού παρακινούν τον παίκτη να τερματίσει το παιχνίδι για να εμπλουτίσει τις γνώσεις του σχετικά με το CSS grid.
	Αυτοεκμάθηση (Self - learning)
CodeCombat	Η θεωρία είναι χωρισμένη σε ενότητες και επίπεδα βοηθώντας έτσι τον παίκτη να κατανοήσει και να αφομοιώσει καλύτερα τις έννοιες που διδάσκεται. Σε κάθε επίπεδο είναι διαθέσιμη η θεωρία που έχει διδαχθεί ο χρήστης στα προηγούμενα επίπεδα. Επίσης, ο παίκτης έχει τη δυνατότητα να παίξει οποιοδήποτε προηγούμενο επίπεδο για να διαβάσει ξανά τη θεωρία και να επαναλάβει τις δραστηριότητες. Με τον προσωπικό λογαριασμό που διαθέτει ο παίκτης έχει τη δυνατότητα να οργανώσει τη μελέτη του και να γνωρίζει το επίπεδο που βρίσκεται.
Crunchzilla	Η θεωρία του παιχνιδιού είναι οργανωμένη σε επίπεδα και κάθε επίπεδο επικεντρώνεται σε μία έννοια, η οποία παρουσιάζεται μέσω παραδειγμάτων. Ο χρήστης έχει τη δυνατότητα να αφιερώσει όσο χρόνο επιθυμεί στο επίπεδο για να κατανοήσει σωστά την έννοια που διδάσκεται. Επίσης, το παιχνίδι τον προτρέπει να εκτελεί τα παραδείγματα που του παρουσιάζονται σε κάθε επίπεδο βοηθώντας τον έτσι να αφομοιώσει σε μεγαλύτερο βαθμό τη θεωρία. Τέλος, μπορεί να ανατρέξει οποιαδήποτε στιγμή στα προηγούμενα επίπεδα και να τα ξαναπαίξει.
CSS Diner	Κάθε επίπεδο του παιχνιδιού είναι αφιερωμένο σε μία έννοια και η θεωρία εξελίσσεται σταδιακά. Αυτό επιτρέπει στον παίκτη να επικεντρωθεί και να κατανοήσει τις πληροφορίες που λαμβάνει σε κάθε επίπεδο πριν προχωρήσει στο επόμενο. Το παιχνίδι δίνει τη δυνατότητα στον παίκτη να ανατρέξει σε οποιοδήποτε επίπεδο επιθυμεί και να διαβάσει πολλές φορές τη θεωρία.
Flexbox Froggy	Η θεωρία που παρουσιάζει το παιχνίδι είναι οργανωμένη σε επίπεδα και ο παίκτης μπορεί να αφιερώσει όσο χρόνο επιθυμεί για να την κατανοήσει πριν προχωρήσει στο επόμενο. Επίσης, ο παίκτης μπορεί να παίξει ένα επίπεδο όσες φορές θέλει δίνοντάς του έτσι τη δυνατότητα να εξοικειωθεί σε μεγαλύτερο βαθμό με τη θεωρία.
Grid Garden	Η θεωρία είναι διαχωρισμένη σε επίπεδα δίνοντας έτσι τη δυνατότητα στον παίκτη να κατανοήσει και να αφομοιώσει κάθε έννοια με το δικό του ρυθμό. Ο παίκτης μπορεί να επαναλάβει ένα επίπεδο χωρίς κάποιο περιορισμό μέχρι να νιώσει ότι έχει εξοικειωθεί με την έννοια που διδάσκεται.
	Επίλυση προβλημάτων (Problem Solving)
CodeCombat	Ο χρήστης σε κάθε επίπεδο γράφει τις κατάλληλες εντολές ώστε

	να ολοκληρώσει επιτυχημένα τη λίστα με τους στόχους και να καθοδηγήσει τον χαρακτήρα του στην έξοδο. Έχοντας τη δυνατότητα να εκτελέσει των κώδικα του όσες φορές θέλει μπορεί να κατανοήσει καλύτερα τις εντολές. Επίσης, τα περιεκτικά μηνύματα λάθους που παρέχει το παιχνίδι τον βοηθούν να καταλάβει τα λάθη του και να οδηγηθεί στη σωστή απάντηση.
Crunchzilla	Ο παίκτης κάνει εφαρμογή της θεωρίας και των παραδειγμάτων που διδάσκεται με σκοπό να σχεδιάσει ότι του ζητείται σε κάθε επίπεδο. Η οπτικοποίηση των εντολών του στον HTML viewer βοηθάει στο να καταλάβει εάν εκτελεί επιτυχημένα τους στόχους του καθώς το παιχνίδι δεν έχει αυστηρό έλεγχο.
CSS Diner	Σε κάθε επίπεδο ο χρήστης χρησιμοποιεί τη θεωρία που διδάσκεται για να επιλέξει τα αντικείμενα που πρέπει πάνω από το τραπέζι.
Flexbox Froggy	Σε αυτό το παιχνίδι, ο παίκτης καλείται να τοποθετεί τους βατράχους πάνω στα νούφαρα κάνοντας εφαρμογή της θεωρίας που μαθαίνει.
Grid Garden	Ο παίκτης χρησιμοποιεί τη θεωρία που διδάχθηκε σε κάθε επίπεδο για να ποτίσει τα καρότα και να εξολοθρεύσει τα ζιζάνια.

Πίνακας 3: Μοντελοποίηση μαθησιακού περιεχομένου (Learning Content Modeling)

	Πρόγραμμα σπουδών (Syllabus Matching)
CodeCombat	Το παιχνίδι παρέχει ένα ολοκληρωμένο πρόγραμμα μάθησης. Αρχικά καλύπτει τις βασικές έννοιες όπως το βασικό συντακτικό, οι παράμετροι, οι μέθοδοι και οι συμβολοσειρές και φτάνει μέχρι το σημείο να καλύπτει προχωρημένες τεχνικές προγραμματισμού. Ο παίκτης μπορεί να δει αυτό το πρόγραμμα εξερευνώντας τον κεντρικό χάρτη του παιχνιδιού.
Crunchzilla	Το παιχνίδι παρουσιάζει ένα πλήρες πρόγραμμα μέσω του οποίου διδάσκει σημαντικές έννοιες της JavaScript, όπως μεταβλητές, εκφράσεις, συνθήκες, συναρτήσεις και βρόχοι. Όμως, δεν σταματάει μόνο σε αυτές καθώς διδάσκει τον μαθητή πως θα σχεδιάζει αντικείμενα με τη βοήθεια του canvas object και πως μπορεί να αλληλεπιδρά με τα αντικείμενα αυτά.
CSS Diner	Το παιχνίδι αυτό καλύπτει ένα μεγάλο μέρος των CSS selectors. Ο παίκτης παρατηρώντας τη λίστα των επιπέδων μπορεί να δει τι διδάσκεται σε κάθε επίπεδο.
Flexbox Froggy	Η διδασκαλία του παιχνιδιού είναι προσανατολισμένη στους CSS κανόνες που έχουν ως στόχο την τοποθέτηση αντικειμένων με τη χρήση CSS flexbox. Μέσα από τα μαθήματα καλύπτει πλήρως τη

	θεματολογία του. Ωστώσο, ο παίκτης δεν μπορεί να δει τι περιλαμβάνει κάθε επίπεδο εάν δεν το ξεκινήσει.
Grid Garden	Το παιχνίδι παρουσιάζει ολοκληρωμένα τους CSS κανόνες του CSS grid που είναι σχετικοί για τη δημιουργία διατάξεων σε μια ιστοσελίδα. Το περιεχόμενο κάθε επιπέδου είναι ορατό μόνο όταν ο παίκτης εισέλθει σε αυτό.
	Πλαίσιο στήριξης (Scaffolding)
CodeCombat	Υπάρχει μια ομαλή ροή στην παρουσίαση των εννοιών από το παιχνίδι. Όσο ο παίκτης προχωράει μαθαίνει νέα πράγματα τα οποία συνδυάζει με αυτά που έχει διδαχθεί ήδη. Σε κάθε επίπεδο είναι διαθέσιμη η λίστα των εντολών που μπορεί να χρησιμοποιήσει και μπορεί να διαβάσει για αυτές τις εντολές χωρίς πίεση χρόνου.
Crunchzilla	Αρχικά το παιχνίδι ξεκινάει με απλά επίπεδα και καταλήγει σε πιο σύνθετα εξηγώντας πώς λειτουργεί μια έννοια και παρουσιάζοντας παραδείγματα. Υπάρχουν ενσωματωμένα κουίζ, ώστε οι μαθητές να μπορούν να ελέγξουν ό,τι γνωρίζουν.
CSS Diner	Στα πρώτα επίπεδα το παιχνίδι ξεκινάει διδάσκοντας τους πιο απλούς και δημοφιλείς selectors. Σταδιακά, όσο ανεβαίνουν τα επίπεδα η δυσκολία αυξάνεται παρουσιάζοντας πιο δύσκολους selectors.
Flexbox Froggy	Το παιχνίδι εισάγει τον παίκτη στις ιδιότητες του CSS flexbox κλιμακωτά, καλώντας τον παίκτη να δοκιμάσει όλες τις πιθανές ιδιότητες και να τις χρησιμοποιήσει συνδυαστικά.
Grid Garden	Οι ιδιότητες του CSS grid παρουσιάζονται με καλώς ορισμένη σειρά βοηθώντας τον παίκτη να εξελίξει τη σκέψη του πάνω στο θέμα.

3 Μεθοδολογία

3.1 Σκοπός του παιχνιδιού

Ο σκοπός του παιχνιδιού που θα δημιουργηθεί είναι η εκμάθηση των HTML, CSS και JavaScript, που αποτελούν τις τρεις κύριες γλώσσες για την κατασκευή ιστοτόπων. Στόχος είναι μέσα από τη ροή και τις δραστηριότητες του παιχνιδιού ο παίκτης να γνωρίσει και να εξοικειωθεί με τις βασικές έννοιες αυτών των τριών τεχνολογιών ιστού. Αρχικά ο παίκτης θα έρθει σε επαφή με τη δομή και τις ετικέτες της HTML. Στη συνέχεια, θα διδαχθεί πως με τη χρήση του CSS μπορεί να μορφοποιήσει κατάλληλα την HTML. Έπειτα, θα ακολουθήσει η παρουσίαση της γλώσσας προγραμματισμού JavaScript, όπου διδάσκονται οι βασικές λειτουργίες, το συντακτικό της και η δομή της. Τέλος, θα δοθεί έμφαση στο Document Object Model (DOM), το οποίο είναι μια διεπαφή προγραμματισμού εφαρμογών (API), που χρησιμοποιείται για το χειρισμό HTML και XML εγγράφων.

Ακόμη, ένα στόχος είναι η θεωρία που θα διδάσκεται να παρουσιάζεται αρμονικά μέσα από τη ροή του παιχνιδιού και να αποτελεί κομμάτι του. Ο παίκτης θα πρέπει να αλληλεπιδρά με τον κόσμο του παιχνιδιού απαντώντας ερωτήσεις πολλαπλής επιλογής και γράφοντας το δικό του κώδικα CSS και JavaScript για να επιβεβαιώνει ότι έχει κατανοήσει το εκπαιδευτικό περιεχόμενο και να μπορέσει να ολοκληρώσει το παιχνίδι.

3.2 Ανάλυση και σχεδίαση του παιχνιδιού

Η αρχική σχεδίαση του παιχνιδιού πραγματοποιήθηκε με βάση το πλαίσιο σχεδίασης Educational Games Design Model (Ibrahim & Jaafar, 2009) το οποίο παρουσιάστηκε στην ενότητα 2.3. Στους Πίνακες 4, 5 και 6 παρουσιάζονται οι σχεδιαστικές αποφάσεις σύμφωνα με τους τρεις βασικούς άξονες του πλαισίου οι οποίοι είναι ο σχεδιασμός του παιχνιδιού (Game Design), οι παιδαγωγικοί στόχοι (Pedagogy) και η μοντελοποίηση μαθησιακού περιεχομένου (Learning Content Modeling).

Πίνακας 4: Ο άξονας του σχεδιασμού παιχνιδιού (Game design)

Χαρακτηριστικά	**Σχεδιαστική απόφαση**
<u>Ευχρηστία (Usability)</u>	
Ικανοποίηση (Satisfaction)	Όταν ο παίκτης ολοκληρώνει ένα επίπεδο εμφανίζονται σε αυτόν οι συνολικοί πόντοι που έχει συγκεντρώσει, ο χρόνος που κατάφερε να τερματίσει το παιχνίδι και μηνύματα επιβράβευσης τα οποία χαροποιούν τον παίκτη αφού ανταμείβεται η προσπάθειά του. Επίσης, η πρόοδος του παίκτη αποθηκεύεται σε βάση δεδομένων, οπότε μπορεί να παίξει το παιχνίδι όποτε επιθυμεί και να βελτιώσει την απόδοσή του σε κάθε επίπεδο. Τα επίπεδα ανοίγουν σταδιακά συμβάλλοντας έτσι στην ικανοποίηση του παίκτη. Τέλος, δίνοντας σωστή απάντηση στις προκλήσεις που του εμφανίζονται σε κάθε επίπεδο μαζεύει πόντους και ενδυναμώνει το χαρακτήρα του παιχνιδιού.
Αποδοτικότητα (Efficiency)	Η θεωρία παρουσιάζεται σταδιακά σε κάθε επίπεδο μέσω απλού και κατανοητού κειμένου, συνοδευόμενη από παραδείγματα. Όταν ο παίκτης διαβάσει κάποιο τμήμα θεωρίας τότε αυτό είναι διαθέσιμο κάθε φορά που ο παίκτης καλείται να απαντήσει σε κάποια ερώτηση. Επιπλέον, μέσα από τις δραστηριότητες που ολοκληρώνει ο παίκτης στο παιχνίδι μπορεί να επιβεβαιώσει αυτά που έχει μάθει. Τέλος, το παιχνίδι μπορεί να παιχτεί ελεύθερα σε κάποιον από τους δημοφιλείς browsers.
Αποτελεσματικότητα (Effectiveness)	Κάθε επίπεδο διδάσκει έννοιες οι οποίες καλύπτουν μία συγκεκριμένη ενότητα. Αυτές κατανέμονται στο επίπεδο χωριστά ώστε να είναι πιο αποτελεσματική η εκμάθηση τους. Ο έλεγχος των απαντήσεων που δίνει ο χρήστης συμβάλλει στην αποτελεσματική κατανόηση τους. Επίσης, τα μηνύματα ανατροφοδότησης που λαμβάνει βοηθούν στην εύρεση του λάθους του.
<u>Πολυτροπικότητα (Multimodal)</u>	
Πολυμέσα και αλληλεπιδράσεις (multimedia and interaction)	Το παιχνίδι έχει την αισθητική και τα γραφικά ενός platformer παιχνιδιού δύο διαστάσεων. Ο παίκτης έρχεται σε επαφή και μαζεύει αντικείμενα, όπως σεντούκια και φίλτρα, και αντιμετωπίζει εχθρούς, όπως Όρκς, διατηρώντας έτσι το ενδιαφέρον του. Επιπλέον, η προσθήκη ηχητικών εφέ στις παραπάνω διαδικασίες αποτελούν σημαντική συμβολή στο παιχνίδι. Ο παίκτης αλληλεπιδρά με το παιχνίδι απαντώντας σε ερωτήσεις πολλαπλής επιλογής, αλλά και με τη συμπλήρωση κώδικα σε ορισμένα σημεία. Στις ερωτήσεις πολλαπλής

	επιλογής χρωματίζεται με πράσινο η σωστή απάντηση και με κόκκινο η λανθασμένη. Επίσης, εμφανίζεται κείμενο το οποίο βοηθάει στην κατανόηση της απάντησης. Στη συμπλήρωση κώδικα, εμφανίζονται μηνύματα σε φούσκα ομιλίας σε περίπτωση που προκύψουν σφάλματα και ο κώδικας δεν εκτελεστεί. Στην περίπτωση που ο κώδικας εκτελεστεί με επιτυχία εμφανίζονται πλατφόρμες ή μετακινούνται βράχια στον κόσμο του παιχνιδιού για να συνεχίσει ο ήρωας το ταξίδι του και αφαιρούνται εμπόδια που του κλείνουν το δρόμο. Στο τέλος κάθε επιπέδου εμφανίζονται τα κατάλληλα μηνύματα και ακούγεται η ανάλογη μουσική σε περίπτωση νίκης ή ήττας. Επίσης, κάθε επίπεδο διαδραματίζεται σε διαφορετική ώρα της ημέρας και αυτό φαίνεται στον παίκτη από το χρωματισμό του ουρανού, τη φωτεινότητα αλλά και τη μουσική που παίζει στο υπόβαθρο.
Ψυχαγωγία - Πρόκληση (Fun - Challenge)	
Ξεκάθαροι στόχοι (Clean goals)	Ο βασικός στόχος κάθε επιπέδου είναι να εκτελέσει σωστά ο χρήστης τρεις δραστηριότητες, είτε αυτό είναι ερώτηση πολλαπλής επιλογής είτε συμπλήρωση κώδικα και να προχωρήσει στο επόμενο.
Αβέβαιο αποτέλεσμα (uncertain outcome)	Ο παίκτης θα πρέπει να απαντήσει σωστά σε τρεις τουλάχιστον ερωτήσεις για να προχωρήσει στο επόμενο επίπεδο. Καθώς προχωράει στα επίπεδα η δυσκολία ανεβαίνει αφού αυξάνονται οι εχθροί του και οι δραστηριότητες γίνονται πιο απαιτητικές.
Αυτοπεποίθηση (Self esteem)	Η δυνατότητα να επανεκκινήσει ο παίκτης το επίπεδο όσες φορές επιθυμεί, αλλά και η δυνατότητα να βελτιώσει το σκορ και το χρόνο σε κάθε επίπεδο ενισχύουν την αυτοπεποίθηση του παίκτη. Επίσης, τα κατάλληλα μηνύματα που προβάλλονται και η επανεμφάνιση των ερωτήσεων που έχει απαντήσει λάθος σε κάθε επίπεδο συμβάλλουν σε αυτό.

Πίνακας 5: Ο άξονας των παιδαγωγικών θεμάτων (Pedagogy)

Χαρακτηριστικά	**Σχεδιαστική απόφαση**
Μαθησιακά αποτελέσματα (Learning Outcomes)	Σκοπός του παιχνιδιού είναι ο χρήστης να διδαχθεί τις βασικές έννοιες και τη δομή των HTML, CSS και JavaScript. Σχετικά με την HTML θα μάθει για τη δομή ενός αρχείου HTML, τα elements και πως ομαδοποιούνται, τα attributes, τη παρουσίαση κειμένου, τη δημιουργία λίστας, πίνακα και φόρμας, την εισαγωγή συνδέσμων, εικόνας και βίντεο, καθώς και για τη semantic HTML. Στην ενότητα που αφορά το CSS μαθαίνει για τους selectors, τη σύνταξη κανόνων, τις ιδιότητες που αφορούν την μορφοποίηση κειμένου, το box model, το box-sizing, την τοποθέτηση elements, το flexbox και το grid. Τέλος, η JavaScript περιλαμβάνει θέματα όπως οι μεταβλητές, ο τύποι δεδομένων, οι εντολές if, οι βρόχοι, οι συναρτήσεις, η έννοια του scope, οι πίνακες και οι μέθοδοι τους, τα objects και η παρουσίαση και διαχείριση του DOM.
Θεωρία παρακίνησης (Motivation Theory)	Ο παίκτης σε κάθε επίπεδο μαθαίνει νέες έννοιες σε σχέση με τις τρεις τεχνολογίες ιστού που διδάσκεται κάτι το οποίο τον παρακινεί για να μάθει περισσότερα για να εμπλουτίσει τις γνώσεις του. Επίσης, ξεκινώντας το παιχνίδι όλα τα επίπεδα εκτός από το πρώτο είναι κλειδωμένα, παρακινόντας έτσι τον παίκτη να θέλει να τερματίσει το παιχνίδι.
Αυτοδιδασκαλία (Self - learning)	Η θεωρία του παιχνιδιού είναι ομαδοποιημένη σε ενότητες και επίπεδα δίνοντας έτσι τη δυνατότητα στον παίκτη να κατανοήσει τις εννοιες κάθε επιπέδου πριν προχωρήσει στο επόμενο. Επίσης, μπορεί να παίξει ένα επίπεδο όσες φορές θέλει αφού το ολοκληρώσει για να αφομοιώσει καλύτερα τη θεωρία του. Τέλος, με τη δημιουργία λογαριασμού και την αποθήκευση της προόδου του μπορεί να οργανώσει τη μελέτη του σύμφωνα με το δικό του ρυθμό.
Επίλυση προβλημάτων (Problem Solving)	Σε κάθε επίπεδο ο παίκτης πρέπει να ολοκληρώσει επιτυχώς τρεις δραστηριότητες που έχουν σχέση με τη θεωρία που διάβασε για να ξεκλειδώσει την πόρτα του επιπέδου και να οδηγήσει το χαρακτήρα του στο επόμενο. Οι δραστηριότητες αυτές μπορεί να είναι είτε ερωτήσεις πολλαπλής επιλογής είτε σύνταξη κώδικα. Σε περίπτωση που ο παίκτης απαντήσει λάθος σε κάποια ερώτηση πολλαπλής επιλογής δεν μπορεί να ολοκληρώσει το επίπεδο, αλλά αφού ο χαρακτήρας ακουμπήσει την πόρτα στο τέλος του επιπέδου τα αντικείμενα που εξαφανίστηκαν έπειτα από λανθασμένη απάντηση θα εμφανιστούν ξανά. Επίσης, κατά την σύνταξη κώδικα λαμβάνει κατάλληλα μηνύματα που τον καθοδηγούν προς τη σωστή επίλυση του προβλήματος.

Πίνακας 6: Μοντελοποίηση μαθησιακού περιεχομένου (Learning Content Modeling)

Χαρακτηριστικά	**Σχεδιαστική απόφαση**
Πρόγραμμα σπουδών (Syllabus Matching)	Το παιχνίδι καλύπτει το βασικό μέρος και των τριών τεχνολογιών ιστού HTML, CSS και JavaScript. Η επιλογή της θεωρίας και των ασκήσεων και η σειρά με την οποία αυτά παρουσιάζονται έγιναν με τη βοήθεια και τη μελέτη των παρακάτω ιστοσελίδων: ● mdn - https://developer.mozilla.org/ ● CSS tutorials - https://webplatform.github.io/docs/css/tutorials/ ● Speaking JavaScript - http://speakingjs.com/es5/
Προοδευτικότητα - Εξέλιξη (Scaffolding)	Το παιχνίδι περιέχει ομαλή παρουσίαση της θεωρίας και η δυσκολία του ανεβαίνει σταδιακά. Αρχικά, ο παίκτης ξεκινά με τα επίπεδα που αφορούν την HTML. Στη συνέχεια, ακολουθεί η διδασκαλία του CSS, που συμβάλλει στην μορφοποίηση της HTML και τέλος, μαθαίνει τις βασικές λειτουργίες της JavaScript. Επιπλέον, και το επίπεδο δυσκολίας κάθε ενότητας ανεβαίνει σταδιακά.

3.3 Τεχνολογίες υλοποίησης του παιχνιδιού

3.3.1 Πλαίσιο εργασίας Phaser

Για την ανάπτυξη του παιχνιδιού επιλέχθηκε το δημοφιλές framework Phaser (Εικόνα 9). Το Phaser είναι ένα framework παιχνιδιών δύο διαστάσεων για τη δημιουργία παιχνιδιών HTML5 για υπολογιστές και κινητά. Αναπτύχθηκε από την Photon Storm και αποτελεί μία πλατφόρμα ανοιχτού λογισμικού. Ο κώδικας του framework είναι διαθέσιμος στο GitHub (https://github.com/photonstorm/phaser).

Εικόνα 9: Λογότυπο Phaser

Η επιλογή του συγκεκριμένου framework έγινε διότι υποστηρίζει τη δημιουργία παιχνιδιών στη γλώσσα JavaScript, η οποία αποτελεί τη γλώσσα στην οποία θα υλοποιηθεί το παιχνίδι. Ένας ακόμη αξιοσημείωτος λόγος είναι ότι αποτελεί λογισμικό ανοιχτού κώδικα και παρέχεται δωρεάν. Επίσης, είναι εύκολο στην εγκατάσταση και παρέχει πληθώρα από οδηγούς και παραδείγματα αφού είναι αρκετά δημοφιλές και η κοινότητα του παιχνιδιού είναι αρκετά μεγάλη. Τέλος, σημαντικό λόγο επιλογής της αποτελούν τα χαρακτηριστικά της, τα οποία είναι η ενσωματωμένη Physics και Sound engine, η εύκολη φόρτωση και διαχείριση αρχείων, όπως JSON και CSV, γραφικών και ήχων μέσω των συναρτήσεων που περιέχει και ο σαφής διαχωρισμός των καταστάσεων του παιχνιδιού.

3.3.2 Τεχνολογίες για το συντάκτη κώδικα

3.3.2.1 Ace (Ajax.org Cloud9 Editor)

Για την εισαγωγή συντάκτη κώδικα στο παιχνίδι επιλέχθηκε ο επεξεργαστής κώδικα Ace (Ajax.org Cloud9 Editor) (Εικόνα 10). Το Ace αποτελεί έναν επεξεργαστή κώδικα, ο οποίος μπορεί να ενσωματωθεί σε οποιαδήποτε ιστοσελίδα και εφαρμογή JavaScript. Είναι ένα project ανοιχτού κώδικα που φιλοξενείται στο GitHub (https://github.com/ajaxorg/ace) και είναι γραμμένο στη γλώσσα JavaScript. Το Ace αναπτύχθηκε ως ο κύριος επεξεργαστής για το Cloud9 IDE και πλέον αποτελεί τον κύριο διάδοχο του έργου Mozilla Skywriter. Βασικοί λόγοι επιλογής του συγκεκριμένου επεξεργαστή κώδικα αποτελούν η συγγραφή του σε JavaScript, η ευκολία ενσωμάτωσής του, η δυνατότητα σύνταξης κώδικα σε περισσότερες από 120 γλώσσες, συμπεριλαμβανομενων των HTML, CSS και JavaScript που θα χρησιμοποιηθούν στο παιχνίδι και η παροχή όλων των δυνατοτήτων που περιέχει ένας επεξεργαστής κώδικα.

Εικόνα 10: Λογότυπο Ace

3.3.2.2 Βιβλιοθήκες για τον συντάκτη κώδικα

Σε αυτό το σημείο παρουσιάζονται βιβλιοθήκες οι οποίες θα χρησιμοποιηθούν στη διαμόρφωση και στην επεξεργασία του κώδικα που θα εισάγει ο χρήστης στον συντάκτη. Η επιλογή των παρακάτω έγινε διότι αποτελούν εργαλεία ανοιχτού κώδικα, είναι εύκολα στη χρήση τους ιδιαίτερα σε εφαρμογές JavaScript, διαθέτουν μεγάλη κοινότητα και ενημερώνονται συνέχεια.

Οι βιβλιοθήκες αυτές είναι:

- Η βιβλιοθήκη JS Beautifier (https://github.com/beautify-web/js-beautify) είναι ανοιχτού λογισμικού και χρησιμοποιείται για να επαναδιαμορφώνει και να βελτιώνει τον κώδικα ως προς την ανάγνωσή του.
- Το JSHint (https://github.com/jshint/jshint) είναι ένα εργαλείο ανάλυσης κώδικα το οποίο χρησιμοποιείται για τον εντοπισμό σφαλμάτων και πιθανών προβλημάτων στον κώδικα που είναι γραμμένος σε γλώσσα JavaScript.
- Το Esprima (https://github.com/jquery/esprima) είναι ένας parser γραμμένος σε JavaScript, ο οποίος χρησιμοποιείται για την λεξιλογική και συντακτική ανάλυση προγραμμάτων γραμμένων σε JavaScript.

3.3.3 Πρόγραμμα σχεδίασης χαρτών Tiled map editor

Το πρόγραμμα που θα χρησιμοποιηθεί για τη δημιουργία των χαρτών όλων των επιπέδων είναι το Tiled Map Editor (https://www.mapeditor.org) (Εικόνα 11). Το πρόγραμμα είναι δωρεάν και είναι κατάλληλο για την ανάπτυξη περιεχομένου σε παιχνίδια δύο διαστάσεων που βασίζονται σε πλακίδια, όπως RPGs και platformers. Ένας σημαντικός λόγος για την επιλογή του είναι η ευελιξία που προσφέρει καθώς επιτρέπει τη δημιουργία χαρτών διαφορετικού μεγέθους, χωρίς να περιορίζει το μέγεθος των πλακιδίων και τον αριθμό των επιπέδων ή των πλακιδίων που μπορεί κάποιος να χρησιμοποιήσει. Επίσης, δίνει τη δυνατότητα για την προσθήκη αντικειμένων, τα οποία διαφέρουν από τα πλακίδια και την εκχώρηση αυθαίρετων ιδιοτήτων σε αυτά. Τέλος, σημαντικό πλεονέκτημα αποτελεί ότι υποστηρίζεται από πολλά πλαίσια ανάπτυξης παιχνιδιών, όπως και το Phaser, καθώς μπορεί να αποθηκευτεί είτε σε αρχείο TMX είτε σε αρχείο JSON.

Εικόνα 11: Λογότυπο Tiled Map Editor

3.3.4 Node js και modules

Το Node.js (Εικόνα 12) είναι ένα δωρεάν ανοιχτού κώδικα περιβάλλον διακομιστή, το οποίο τρέχει σε όλες τις πλατφόρμες (Windows, Linux, Unix, Mac OS X, κ.λπ.) και εκτελεί κώδικα JavaScript από την πλευρά του διακομιστή. Ο σχεδιασμός και η αρχιτεκτονική της την καθιστούν κατάλληλη για τη δημιουργία διαδικτυακών εφαρμογών αλλά και εφαρμογών επικοινωνίας και παιχνιδιών. Οι λόγοι που επιλέχθηκε είναι ο γρήγορος χρόνος εκτέλεσης αλλά και για τη χρήση της JavaScript τόσο από την πλευρά του πελάτη όσο και από την πλευρά του διακομιστή.

Εικόνα 12: Λογότυπο Node.js

Εκτός, από τη JavaScript το Node.js χρησιμοποιεί και μια μεγάλη συλλογή από modules τα οποία παρέχουν διάφορες βασικές λειτουργίες και θα μπορούσαν να θεωρηθούν παρόμοια με τις βιβλιοθήκες της JavaScript. Στον Πίνακα 7 παρουσιάζονται τα modules που θα χρησιμοποιηθούν.

Πίνακας 7: Node.js modules για την υλοποίηση του παιχνιδιού

Module	**Περιγραφή**
Express	Είναι το πιο δημοφιλές framework του Node για τη δημιουργία εφαρμογών ιστού και API. Το module αυτό θα χρησιμοποιηθεί για να δημιουργηθεί το API με το οποίο θα ανταλλάσσει δεδομένα ο server με τον client, σχετικά με τη σύνδεση/εγγραφή του χρήστη και την αποθήκευση των δεδομένων του. Σύνδεσμος: https://expressjs.com/
DotEnv	Είναι ένα module το οποίο φορτώνει αυτόματα τις μεταβλητές περιβάλλοντος από ένα αρχείο .env στο αντικείμενο process.env. Θα χρησιμοποιηθεί για να μεταβλητές όπως το url της βάσης δεδομένων. Σύνδεσμος: https://github.com/motdotla/dotenv
Mongoose	Είναι μία βιβλιοθήκη η οποία διευκολύνει τη σύνδεση και επικοινωνία διακομιστή με βάση δεδομένων τύπου MongoDB. Προσφέρει δυνατότητες όπως η δημιουργία καθορισμένου σχήματος (schema-based) για το μοντέλο της βάσης, η εύκολη δημιουργία ερωτημάτων στη βάση δεδομένων, καθώς και την εισαγωγή, τροποποίηση και διαγραφή αρχείων στη βάση δεδομένων. Η βιβλιοθήκη αυτή θα χρησιμοποιηθεί για τη δημιουργία του μοντέλου του χρήστη της εφαρμογής, για τη δημιουργία ερωτημάτων στη βάση, αλλά και για τη δημιουργία και ενημέρωση των χρηστών στη βάση. Σύνδεσμος: https://mongoosejs.com/
Bcrypt	Αποτελεί μια βιβλιοθήκη η οποία παρέχει μηχανισμούς για τον κατακερματισμό κωδικών πρόσβασης. Θα χρησιμοποιηθεί για τον κατακερματισμό του συνθηματικού του χρήστη της εφαρμογής πριν αυτός αποθηκευτεί στη βάση δεδομένων. Σύνδεσμος: https://github.com/kelektiv/node.bcrypt.js
Passport	Είναι ένα module το οποίο προσθέτει λειτουργίες ελέγχου ταυτότητας στους ιστοτόπους και στις εφαρμογές. Θα γίνει χρήση του συγκεκριμένου εργαλείου για να γίνεται αυθεντικοποίηση των αιτημάτων που ζητούν ή ανανεώνουν δεδομένα του χρήστη. Σύνδεσμος: https://www.passportjs.org/
Passport-local	Αποτελεί μία στρατηγική του module passport που χρησιμοποιεί το όνομα χρήστη και τον κωδικό πρόσβασης για τον έλεγχο ταυτότητας.
Passport-jwt	Αποτελεί μία στρατηγική του module passport που χρησιμοποιεί ένα JSON web token για τον έλεγχο ταυτότητας.
Cookie-parser	Με το module αυτό αποκτούμε πρόσβαση στη επικεφαλίδα cookie που υπάρχει στα αιτήματα πελάτη - διακομιστή. Θα χρησιμοποιηθεί για την διακίνηση του JSON web token. Σύνδεσμος: https://github.com/expressjs/cookie-parser

Nodemailer	Είναι ένα από τα πιο δημοφιλή modules για την αποστολή email μέσω εφαρμογών που είναι φτιαγμένες με το Node.js. Το κομμάτι στο οποίο θα χρησιμοποιηθεί είναι η αποστολή email για την ανάκτηση συνθηματικού από το χρήστη. Σύνδεσμος: https://nodemailer.com/

3.3.5 MongoDB Atlas

Η MongoDB (https://www.mongodb.com/) (Εικόνα 13) είναι μία ανοιχτού κώδικα document-oriented βάση δεδομένων, που μπορεί να εκτελεστεί σε διάφορα λειτουργικά συστήματα. Κυκλοφόρησε το 2007 και είναι η πιο δημοφιλής NoSQL βάση δεδομένων. Χρησιμοποιεί μοντέλο JSON-like για να αποθηκεύσει τα δεδομένα και δίνει τη δυνατότητα στους προγραμματιστές να αποθηκεύουν δομημένα ή μη δεδομένα. Επίσης, είναι απλή στη χρήση της και μπορεί να διαχειριστεί μεγάλο όγκο δεδομένων. Για τη φιλοξενία της βάσης δεδομένων θα χρησιμοποιηθεί η cloud υπηρεσία Atlas που παρέχεται από τη MongoDB. Η MongoDB Atlas παρέχει έναν εύκολο τρόπο διαχείρισης των δεδομένων και προσφέρει ένα δωρεάν cluster, αποθηκευτικού χώρου 512 MB στους χρήστες που δημιουργούν λογαριασμό. Η συγκεκριμένη επιλογή έγινε διότι οι NoSQL βάσεις δεδομένων δεν χρησιμοποιούν κάποιο προκαθορισμένο σχήμα (schema-less) και μπορούν να επεκταθούν πιο εύκολα και η υπηρεσία Atlas είναι cloud και προσφέρει εύκολη διαχείριση των δεδομένων.

Εικόνα 13: Λογότυπο MongoDB

4 Παρουσίαση του παιχνιδιού

4.1 Περιγραφή

Το παιχνίδι σοβαρού σκοπού που αναπτύχθηκε και παρουσιάζεται ονομάζεται «Super Dwarf: Quest for the prototype hammer». Είναι ένα παιχνίδι περιπέτειας φαντασίας, που ανήκει στην κατηγορία παιχνιδιών πλατφόρμας, καθώς περιλαμβάνει στοιχεία όπως ο ελιγμός του χαρακτήρα του παιχνιδιού σε πλατφόρμες, η αντιμετώπιση εχθρών, η αποφυγή εμποδίων και η συνεχής γρήγορη κίνηση του χαρακτήρα μπροστά.

Το σενάριο του παιχνιδιού λαμβάνει χώρα σε ένα φανταστικό κόσμο και ακολουθεί τις περιπέτειες του νάνου κληρικού Super Dwarf. Μια μέρα, Όρκς εισέβαλαν στο ναό του θεού Moradin και έκλεψαν το ιερό σφυρί του. Ο νάνος αρχιερέας κάλεσε το γενναίο κληρικό Super Dwarf και του ζήτησε να αναζητήσει το ιερό όπλο και να το φέρει πίσω στο ναό αλλιώς τα Όρκς θα κυριαρχήσουν. Έτσι ξεκίνησε το ταξίδι και η περιπέτεια του Super Dwarf.

Ο παίκτης πρέπει να καθοδηγήσει τον ήρωα και να τερματίσει όλα τα επίπεδα για να βρει το ιερό σφυρί. Σε κάθε επίπεδο όμως θα πρέπει να αντιμετωπίσει τα εχθρικά Ορκς, χρησιμοποιώντας τη μαγική δύναμη του Super Dwarf και να ολοκληρώσει τις δραστηριότητες που του παρουσιάζονται για να τον δυναμώσει και να τον βοηθήσει να συνεχίσει το ταξίδι του. Μέσα από το παιχνίδι ο παίκτης θα έρθει σε επαφή με έννοιες της HTML, του CSS και της JavaScript και για να φέρει εις πέρας τις δραστηριότητες θα πρέπει να έχει κατανοήσει τη θεωρία που συναντά σε διάφορα σημεία του επιπέδου και να τη χρησιμοποιήσει κατάλληλα. Για να μπορέσει να προχωρήσει στο επόμενο επίπεδο θα πρέπει να ολοκληρώσει τρεις δραστηριότητες.

Στο παιχνίδι υπάρχουν διάφορα αντικείμενα με τα οποία ο παίκτης αλληλεπιδρά και του παρουσιάζεται το εκπαιδευτικό περιεχόμενο. Τα αντικείμενα αυτά είναι τα σεντούκια, τα μαγικά φίλτρα και η φωτιά. Μόλις ο παίκτης ακουμπήσει ένα σεντούκι και αφού πατήσει το πλήκτρο F εμφανίζεται μια περγαμηνή η οποία περιέχει τη σχετική θεωρία και παραδείγματα που ενισχύουν την κατανόησή της. Όταν ο παίκτης έρθει σε επαφή με ένα μαγικό φίλτρο και πατήσει το F τότε εμφανίζεται σε αυτόν μία περγαμηνή με μία ερώτηση πολλαπλής επιλογής με τέσσερις διαθέσιμες απαντήσεις. Η ερώτηση μπορεί να είναι είτε θεωρητικού περιεχομένου είτε να σχετίζεται με την εκτέλεση του κομματιού ενός κώδικα. Εάν ο παίκτης απαντήσει σωστά χρωματίζεται η απάντηση με πράσινο χρώμα και ενισχύεται η ζωή (κόκκινο ή πράσινο φίλτρο) ή η μαγική δύναμη (μπλε

φίλτρο) του ήρωα. Σε περίπτωση που απαντήσει λάθος χρωματίζεται με κόκκινο χρώμα η απάντησή του και με πράσινο η σωστή. Και στις δύο περιπτώσεις εμφανίζεται κείμενο σχετικό με την απάντηση και στη συνέχεια το φίλτρο εξαφανίζεται. Τέλος, όταν ο ήρωας έρχεται σε επαφή με τη φλόγα που βρίσκεται στο έδαφος εμφανίζεται στο κάτω μέρος της οθόνης ένας συντάκτης κώδικα. Πάνω από το συντάκτη βρίσκονται οι οδηγίες σύμφωνα με τις οποίες ο παίκτης πρέπει να συμπληρώσει και να εκτελέσει τον κώδικα του. Ο κώδικας που θα συμπληρώσει το χρήστης μπορεί να αφορά είτε το CSS είτε τη JavaScript. Κάθε φορά που εκτελείται ο κώδικας θα γίνονται άμεσα φανερά τα αποτελέσματα στον κόσμο του παιχνιδιού και ο ήρωας θα μπορεί να συνεχίσει το ταξίδι του. Σε περίπτωση που ο κώδικας δεν είναι εκτελέσιμος θα εμφανίζεται στον παίκτη η κατάλληλη ανατροφοδότηση μέσα σε μία φούσκα ομιλίας. Οι ερωτήσεις πολλαπλής επιλογής και η συμπλήρωση κώδικα αποτελούν τις δραστηριότητες που πρέπει να εκπληρωθούν σε κάθε επίπεδο. Εάν ο ήρωας φτάσει στο τέλος του επιπέδου χωρίς να απαντήσει σωστά σε τρεις δραστηριότητες τότε εμφανίζεται σχετικό μήνυμα και επανατοποθετούνται ξανά στον κόσμο του παιχνιδιού τα αντικείμενα που εξαφανίστηκαν δίνοντας ευκαιρία στον παίκτη να δοκιμάσει ξανά τις γνώσεις του. Στον Πίνακα 8 παρουσιάζονται οι χαρακτήρες και τα αντικείμενα που υπάρχουν σε κάθε επίπεδο.

Πίνακας 8: Αντικείμενα και χαρακτήρες του παιχνιδιού

Όνομα	**Περιγραφή**	**Εικόνα**
Super Dwarf	Είναι ο βασικός χαρακτήρας του παιχνιδιού, τον οποίο χειρίζεται ο παίκτης.	
Εχθρός 1	Είναι ένας από τους εχθρούς του Super Dwarf. Εάν έρθει σε επαφή με τον Super Dwarf τότε του αφαιρεί μια ζωή.	
Εχθρός 2	Ακολουθεί την ίδια λογική με τον εχθρό 1.	
Σεντούκι	Αποτελεί το σημείο το οποίο παρουσιάζει τη θεωρία στον παίκτη όταν έρθει σε επαφή με τον Super Dwarf και πατήσει το πλήκτρο F.	
Κόκκινο φίλτρο	Η συλλογή του κόκκινου φίλτρου αυξάνει τη ζωή του Super Dwarf κατά 1. Σε περίπτωση που ενεργοποιείται μία ερώτηση πολλαπλής	

	επιλογής πρέπει να απαντηθεί σωστά για να αυξηθεί η ζωή του.	
Μπλε φίλτρο	Με το μπλε φίλτρο αυξάνονται οι μαγικές φωτιές του Super Dwarf κατά 5. Σε κάποιες περιπτώσεις το φίλτρο απαιτεί τη σωστή απάντηση μιας ερώτησης για να δώσει το bonus.	
Πράσινο φίλτρο	Η συλλογή του πράσινου φίλτρου αυξάνει τη συνολική ζωή του ήρωα κατά ένα. Ακολουθεί την ίδια λογική με τα προηγούμενα φίλτρα.	
Φωτιά	Με αυτό ενεργοποιείται η δραστηριότητα του συντάκτη κώδικα.	
Πόρτα	Η πόρτα σηματοδοτεί το τέλος του επιπέδου και μεταφέρει τον ήρωα στο επόμενο επίπεδο	

Τέλος, ο χρήστης για να παίξει το παιχνίδι θα πρέπει να δημιουργήσει προσωπικό λογαριασμό. Η δημιουργία λογαριασμού έχει ως στόχο να δημιουργήσει μια πιο εξατομικευμένη εμπειρία στο χρήστη αποθηκεύοντας την πρόοδο του στη βάση δεδομένων και δίνοντάς του έτσι τη δυνατότητα να συνεχίσει το παιχνίδι όποτε το επιθυμεί και έχει ελεύθερο χρόνο. Επίσης, μπορεί να παρακολουθεί τις επιδόσεις του σε κάθε επίπεδο και να τις βελτιώνει.

4.2 Εκπαιδευτικό περιεχόμενο - Επίπεδα παιχνιδιού

Το παιχνίδι έχει ως στόχο την εκμάθηση των HTML, CSS και JavaScript και την εξοικείωση του παίκτη με αυτές. Ο παίκτης δεν χρειάζεται να γνωρίζει κάποιες από αυτές τις γλώσσες για να ξεκινήσει το παιχνίδι. Το εκπαιδευτικό περιεχόμενο είναι χωρισμένο σε τρεις ενότητες και το παιχνίδι αποτελείται συνολικά από 19 επίπεδα. Στο παράρτημα Α παρουσιάζονται στιγμιότυπα από τα επίπεδα του παιχνιδιού. Η πρώτη ενότητα αφορά την HTML και περιλαμβάνει τα πρώτα πέντε επίπεδα του παιχνιδιού. Η δεύτερη ενότητα σχετίζεται με το CSS και περιέχει τα επίπεδα 6 έως 11. Τέλος, η τρίτη ενότητα που αποτελείται από τα επίπεδα 12 έως 19 παρουσιάζει τη θεωρία της γλώσσας JavaScript με τα επίπεδα 16 έως 19 να είναι αφιερωμένα στο Document Object Model (DOM).

4.2.1 Επίπεδο 1: Εισαγωγή στην HTML

Στο πρώτο επίπεδο η θεωρία ξεκινάει με μία εισαγωγή της HTML. Έπειτα περιέχει πληροφορίες για τη σημαντική ετικέτα body και την οργάνωση της δομής της HTML. Στη

συνέχεια, ακολουθεί η παρουσίαση των heading elements (επικεφαλίδες) και τέλος, εισάγει τις έννοιες των block και inline level elements. Στο συγκεκριμένο επίπεδο περιλαμβάνονται τέσσερις δραστηριότητες πολλαπλής επιλογής που σχετίζονται με τα κομμάτια της θεωρίας.

4.2.2 Επίπεδο 2: Attributes και παρουσίαση elements

Το επίπεδο ξεκινάει με την επεξήγηση της έννοιας attribute αλλά και την παρουσίαση των πιο δημοφιλών attributes. Στην συνέχεια, ακολουθεί η εισαγωγή elements που είναι υπεύθυνα για την ένταξη και τη μορφοποίηση κειμένου στην HTML. Έπειτα, αναφέρεται ο τρόπος με τον οποίο δημιουργούνται ταξινομημένες (ol) και μη (ul) λίστες. Στο τελευταίο κομμάτι θεωρίας αυτού του επιπέδου προβάλλεται ο τρόπος με τον οποίο μπορεί να εισαχθεί εικόνα (img) και video (video) σε ένα αρχείο HTML. Το επίπεδο αυτό περιλαμβάνει συνολικά τρεις δραστηριότητες πολλαπλής επιλογής που έχουν ως σκοπό να επιβεβαιώσουν ότι ο χρήστης κατανόησε τη θεωρία.

4.2.3 Επίπεδο 3: Δομή αρχείου html και δημιουργία συνδέσμων

Σε αυτό το επίπεδο ο χρήστης μαθαίνει για τη δομή και τους κανόνες που πρέπει να ακολουθεί ένα html αρχείο, καθώς και για τα html, head και title elements. Έπειτα, ακολουθεί η επεξήγηση της δημιουργίας συνδέσμων σε μία σελίδα και τέλος παρουσιάζεται ο τρόπος που εισάγονται σχόλια στην HTML. Οι δραστηριότητες του επιπέδου είναι συνολικά τρεις και είναι όλες ερωτήσεις πολλαπλής επιλογής πάνω στη θεωρία του επιπέδου.

4.2.4 Επίπεδο 4: Δημιουργία πίνακα και φόρμας

Η θεωρία του επιπέδου 4 σχετίζεται με τους πίνακες και τις φόρμες. Αρχικά προβάλλεται ο τρόπος με τον οποίο δημιουργείται ένας πίνακας στην HTML και παρουσιάζονται κάποια χρήσιμα attributes όπως το rowspan και το colspan. Στη συνέχεια, ακολουθεί η διδασκαλία για τη δημιουργία φόρμας με το form element και παρουσιάζονται τα πιο δημοφιλή inputs elements που χρησιμοποιούνται σε αυτή. Στο τελευταίο κομμάτι θεωρίας του επιπέδου παρουσιάζονται τα attributes της HTML τα οποία βοηθούν στην επικύρωση μιας φόρμας. Οι τρεις δραστηριότητες του επιπέδου είναι ερωτήσεις πολλαπλής επιλογής που αφορούν τη θεωρία.

4.2.5 Επίπεδο 5: Σημασιολογική HTML

Το επίπεδο αυτό αποτελεί το τελευταίο της ενότητας που αφορά την HTML. Η θεωρία αυτού του επιπέδου επικεντρώνεται στη σημασιολογική (semantic) HTML και παρουσιάζει τα elements header, nav, main και footer σύμφωνα με τα οποία δημιουργείται η βασική δομή μιας ιστοσελίδας. Επίσης, παρουσιάζει τα section, article, aside, figure και figcaption elements και την ενσωμάτωση media όπως video. Και σε αυτό το επίπεδο εμπεριέχονται τρεις δραστηριότητες ερωτήσεων πολλαπλής επιλογής.

4.2.6 Επίπεδο 6: Εισαγωγή στο CSS, selectors και δημιουργία κανόνων

Το επίπεδο 6 είναι το πρώτο επίπεδο της ενότητας του CSS οπότε η θεωρία ξεκινάει με μία εισαγωγή και τους τρόπους που μπορεί να εισαχθεί το CSS στην HTML. Στη συνέχεια, παρουσιάζονται οι selectors που μπορούν να χρησιμοποιηθούν για να επιλεχθούν συγκεκριμένα html elements και οι κανόνες προτεραιότητας τους. Έπειτα, η θεωρία εξηγεί τον τρόπο με τον οποίο μπορούν να γραφούν οι CSS κανόνες. Οι δραστηριότητες που περιέχει το επίπεδο είναι τρεις ερωτήσεις πολλαπλής επιλογής.

4.2.7 Επίπεδο 7: Ιδιότητες κειμένου

Στο έβδομο κεφάλαιο αρχικά παρουσιάζονται οι ιδιότητες με τις οποίες μπορεί να μορφοποιηθεί ένα κείμενο γράφοντας κανόνες CSS. Κάποιες από τις ιδιότητες είναι ο ορισμός γραμματοσειράς (font-family), ο ορισμός μεγέθους των γραμμάτων (font-size), η στοίχιση του κειμένου (text-align) και η απόσταση γραμμάτων (letter-spacing). Έπειτα ακολουθεί η επεξήγηση ιδιοτήτων που είναι σχετικές με τον ορισμό χρώματος σε ένα element και τις τιμές που μπορεί να πάρουν αυτές οι ιδιότητες. Στην συνέχεια, παρουσιάζεται η δυνατότητα τοποθέτησης εικόνας ως φόντο σε ένα element και ιδιότητές μορφοποίησης της εμφάνισης της εικόνας. Το συγκεκριμένο επίπεδο περιλαμβάνει τρεις δραστηριότητες στις οποίες ο χρήστης πρέπει να συντάξει κώδικα. Εάν οι δραστηριότητες ολοκληρωθούν σωστά τότε εμφανίζονται μηνύματα στον κόσμο του παιχνιδιού και τα οποία βοηθούν στη συνέχεια του παιχνιδιού. Η πρώτη δραστηριότητα αφορά τη συμπλήρωση κανόνων με τις ιδιότητες font-size και letter-spacing. Η δεύτερη συνδέεται με το αποτέλεσμα της πρώτης και αφορά την ιδιότητα opacity. Τέλος, η τρίτη δραστηριότητα συνδέεται και αυτή με τη δεύτερη και απαιτεί συγκεκριμένη συμπλήρωση του κανόνα background-image.

4.2.8 Επίπεδο 8: Κατανόηση του box model

Η θεωρία αυτού του επιπέδου εισάγει τον χρήστη στην έννοια του box model και στις ιδιότητες που το συνθέτουν. Στη συνέχεια, παρουσιάζονται οι ιδιότητες overflow και visibility και το τελευταίο κομμάτι της θεωρίας αναφέρεται στην ιδιότητα box-sizing. Η πρώτη και τρίτη δραστηριότητα είναι ερωτήσεις πολλαπλής επιλογής σχετικές με τη θεωρία, ενώ η δεύτερη είναι δραστηριότητα σύνταξης κώδικα και είναι σχετική με την ιδιότητα visibility.

4.2.9 Επίπεδο 9: Τοποθέτηση elements με χρήση CSS

Σε αυτό το επίπεδο η θεωρία σχετίζεται με τις ιδιότητες CSS που αφορούν την τοποθέτηση και την σειρά εμφάνισης των elements σε HTML έγγραφο. Η θεωρία ξεκινάει με την ιδιότητα position και οι τιμές που δέχεται. Έπειτα, ακολουθεί η ιδιότητα display και οι τιμές της και τέλος οι ιδιότητες float και clear. Οι δραστηριότητες του επιπέδου είναι συνολικά τρεις και έχουν όλες τη μορφή ερωτήσεων πολλαπλής επιλογής.

4.2.10 Επίπεδο 10: Εισαγωγή στο Flexbox

Το επίπεδο αυτό είναι επικεντρωμένο στο flexbox που αποτελεί ένα εργαλείο της CSS3. Μέσα από τη θεωρία του επιπέδου παρουσιάζονται όλες οι ιδιότητες που περιλαμβάνει, όπως είναι οι flex-direction, flex-wrap, justify-content, align items, align-content, flex-grow, flex-shrink και flex-basis. Στο επίπεδο αυτό οι δραστηριότητες αφορούν τη σύνταξη κώδικα. Σε κάθε δραστηριότητα εμφανίζονται βράχοι και ο χρήστης γράφοντας κανόνες CSS θα πρέπει να τους τοποθετήσει κατάλληλα για να προχωρήσει στο επίπεδο. Η πρώτη δραστηριότητα αφορά την ιδιότητα flex-direction, η δεύτερη την align-items και η τρίτη την justify-content.

4.2.11 Επίπεδο 11: Εισαγωγή στο CSS Grid

Το επίπεδο 11 είναι το τελευταίο επίπεδο της ενότητας του CSS και η θεωρία του αφορά το εργαλείο CSS grid. Στο επίπεδο παρουσιάζεται ο τρόπος με τον οποίο ένα element ορίζεται ως grid και οι ιδιότητες που σχετίζονται με αυτό. Αρχικά καλύπτονται ιδιότητες που σχετίζονται με τη δημιουργία του πλέγματος του grid οι οποίες είναι οι grid-template-columns, grid-template-rows, grid-template, grid gap, grid area και grid template areas. Τέλος, παρουσιάζονται άλλες ιδιότητες όπως οι justify-items, align-items, justify-content και align-content. Το επίπεδο έχει συνολικά τρεις δραστηριότητες σύνταξης κώδικα που έχουν ως στόχο την τοποθέτηση βράχων όπως στο επίπεδο 10. Στην πρώτη

δραστηριότητα καλύπτεται η ιδιότητα grid-area. Στη συνέχεια, η δεύτερη αφορά την ιδιότητα grid-template-areas και τέλος, η τρίτη επισημαίνει την ιδιότητα align-items.

4.2.12 Επίπεδο 12: Εισαγωγή στην JavaScript

Αυτό το επίπεδο αποτελεί την εισαγωγή στην ενότητα της γλώσσας JavaScript. Η θεωρία ξεκινάει με μία εισαγωγή και τους τρόπους με τους οποίους μπορεί να εκτελεστεί η JavaScript. Έπειτα περιλαμβάνει τον τρόπο δημιουργίας μεταβλητών, την παρουσίαση των τύπων δεδομένων που υποστηρίζει η γλώσσα, την επεξήγηση των λογικών τελεστών και της ισότητας και τέλος, ακολουθεί ο τρόπος με τον οποίο μπορούν να γραφούν conditional statements. Σε αυτό το επίπεδο περιλαμβάνονται τρεις δραστηριότητες πολλαπλής επιλογής που βασίζονται στη θεωρία που διδάσκεται.

4.2.13 Επίπεδο 13: Οι συναρτήσεις στη JavaScript

Η θεωρία του επιπέδου 13 επικεντρώνεται στη διδασκαλία των συναρτήσεων. Αρχικά παρουσιάζεται ο ορισμός μιας συνάρτησης και ο τρόπος που καλείται. Στη συνέχεια, παρουσιάζονται ο παράμετροι με τους οποίους εισάγονται δεδομένα σε μια συνάρτηση και η λέξη κλειδί return με την οποία επιστρέφει μία τιμή η συνάρτηση. Έπειτα, η θεωρία περιέχει την επεξήγηση των function expressions, arrow functions και IIFE (Immediately Invoked Function Expression). Τέλος, παρουσιάζεται η έννοια του scope με το οποίο καθορίζεται που μπορούν να προσπελαστούν ή να αναφερθούν οι μεταβλητές. Το επίπεδο περιλαμβάνει δύο δραστηριότητες σύνταξης κώδικα και μία ερώτηση πολλαπλής επιλογής σχετική με το scope. Η πρώτη δραστηριότητα αφορά τη δημιουργία μιας συνάρτησης, η οποία δέχεται ως παραμέτρους δύο αριθμούς και επιστρέφει τον μεγαλύτερο. Η δεύτερη περιλαμβάνει τη δημιουργία μιας arrow expression function, η οποία δέχεται ως παραμέτρους δύο αριθμούς και επιστρέφει true αν το άθροισμά τους είναι μικρότερο από 50. Οι δραστηριότητες εμφανίζονται σε σημεία του επιπέδου που μπλοκάρουν τη συνέχεια της πορείας του ήρωα. Σε περίπτωση που ο χρήστης απαντήσει σωστά τις δραστηριότητες εμφανίζονται πλατφόρμες στον κόσμο του παιχνιδιού για να μπορέσει να συνεχίσει.

4.2.14 Επίπεδο 14: Πίνακες, βρόχοι και iterators

Στο επίπεδο αυτό η θεωρία ξεκινάει με την ανάλυση του τρόπου δημιουργίας ενός πίνακα. Έπειτα παρουσιάζονται κάποιες από τις πιο σημαντικές μεθόδους των πινάκων με τις οποίες μπορούν να εισαχθούν και να αφαιρεθούν στοιχεία σε αυτούς. Στη συνέχεια

παρουσιάζεται ο τρόπος συγγραφής των βρόχων και τέλος, οι ενσωματωμένες μέθοδοι των πινάκων που βοηθούν στην επανάληψη (iterators). Οι δραστηριότητες του επιπέδου είναι συνολικά τρεις και αποτελούνται από μία ερώτηση πολλαπλής επιλογής που αφορά την εισαγωγή και εξαγωγή στοιχείων σε πίνακα και δύο που απαιτούν συμπλήρωση κώδικα. Στην πρώτη δραστηριότητα συμπλήρωσης κώδικα απαιτείται η δημιουργία μιας συνάρτησης, η οποία δέχεται ως παράμετρο έναν πίνακα με αριθμούς και επιστρέφει το άθροισμα τους, η χρήση βρόχων while, for ή do while. Η δεύτερη δραστηριότητα ζητάει τη δημιουργία μιας συνάρτησης, η οποία δέχεται ως παράμετρο έναν πίνακα με αριθμούς και επιστρέφει το γινόμενό τους και τη χρήση του iterator forEach. Όπως και στο επίπεδο 13 με τη σωστή λύση των δραστηριοτήτων εμφανίζονται πλατφόρμες στον κόσμο του παιχνιδιού για να μπορέσει ο παίκτης να συνεχίσει και να τερματίσει το επίπεδο.

4.2.15 Επίπεδο 15: Αντικείμενα (objects) και η λέξη κλειδί this

Η θεωρία στο επίπεδο 15 ξεκινάει με την παρουσίαση των objects στη JavaScript και περιλαμβάνει την δημιουργία τους, τον τρόπο που μπορούν να προσπελαστούν, προστεθούν και διαγραφούν οι ιδιότητες τους, την εμφώλευση ενός object σε ένα άλλο και το βρόχο επανάληψης για τα objects. Στην συνέχεια, αναλύεται η σημασία της λέξης κλειδί this και τέλος παρουσιάζονται χρήσιμες ιδιότητες και μέθοδοι των objects. Η πρώτη δραστηριότητα είναι σύνταξης κώδικα και περιλαμβάνει τη δημιουργία μιας συνάρτησης που δέχεται ως παράμετρο ένα object και επιστρέφει το κλειδί με το μεγαλύτερο πλήθος γραμμάτων και τη χρήση της επαναληπτικής δομής for in. Η δεύτερη δραστηριότητα είναι ερώτηση πολλαπλής επιλογής και αφορά στη λέξη κλειδί this. Τέλος, η τρίτη δραστηριότητα είναι συμπλήρωσης κώδικα και το περιεχόμενό της αφορά τη δημιουργία μιας συνάρτησης που δέχεται ως παράμετρο ένα object και ένα string και επιστρέφει true εάν αυτό αποτελεί property του object κάνοντας χρήση της μεθόδου hasOwnProperty.

4.2.16 Επίπεδο 16: Εισαγωγή στο Document Object Model (DOM)

Αυτό το επίπεδο αποτελεί την εισαγωγή στην υποενότητα που είναι αφιερωμένη στο Document Object Model (DOM). Αρχικά γίνεται μια παρουσίαση του DOM και των nodes που αποτελούν βασικά συστατικά του. Έπειτα αναλύονται οι μέθοδοι με τις οποίους επιτυγχάνεται η επιλογή των elements και οι ιδιότητες με τις οποίες μπορεί κάποιος να βρει τα συγγενικά elements ενός element. Στο επίπεδο αυτό υπάρχουν συνολικά τρεις ερωτήσεις πολλαπλής επιλογής που στοχεύουν στην κατανόηση της θεωρίας που διδάχθηκε.

4.2.17 Επίπεδο 17: Τροποποίηση του document

Η θεωρία του επιπέδου ξεκινάει με την παρουσίαση των μεθόδων που τροποποιούν το document. Οι μέθοδοι αυτοί δημιουργούν elements, προσθέτουν περιεχόμενο σε αυτά, προσθέτουν, αφαιρούν και αντικαθιστούν elements στο document. Στη συνέχεια, γίνεται αναφορά στα properties των elements και στον τρόπο με τον οποίο αποκτάται η πρόσβαση σε αυτά καθώς και με ποιες μεθόδους γίνεται η επεξεργασία τους. Τέλος, αναλύεται η διαχείριση των CSS styles των elements και με ποιες μεθόδους προσθαφαιρούνται οι κλάσεις. Η πρώτη δραστηριότητα του επιπέδου απαιτεί τη συμπλήρωση κώδικα και ζητάει από τον χρήστη τη δημιουργία μιας συνάρτησης που δέχεται ως παράμετρο την κλάση ενός element, βρίσκει στο document το element αυτό και αφαιρεί τα παιδιά χρησιμοποιώντας τη μέθοδο removeChild. Η δεύτερη δραστηριότητα είναι πολλαπλής επιλογής και αφορά τη διαχείριση των properties. Τέλος, η τρίτη δραστηριότητα έχει ως στόχο τη δημιουργία μιας συνάρτησης που δέχεται ως παράμετρο την κλάση ενός element, βρίσκει στο document το element αυτό και αφαιρεί από τα παιδιά του μία συγκεκριμένη κλάση χρησιμοποιώντας τη μέθοδο remove.

4.2.18 Επίπεδο 18: DOM events, φάσεις πυροδότησης και event object

Το επίπεδο 18 εισάγει τον χρήστη στην έννοια του συμβάντος στη JavaScript. Εξηγείται πως ορίζεται ένας handler ενός event, η ροή του event όταν αυτό πυροδοτείται και οι φάσεις πυροδότησής του. Έπειτα αναλύεται το event object που αποτελεί το όρισμα του handler και τέλος παρουσιάζονται τα συμβάντα που πυροδοτούνται κατά τη φόρτωση της σελίδας. Σε αυτό το επίπεδο υπάρχουν τρεις δραστηριότητες πολλαπλής επιλογής σχετικές με τη θεωρία που αναφέρθηκε.

4.2.19 Επίπεδο 19: Mouse, keyboard, scroll events και event delegation

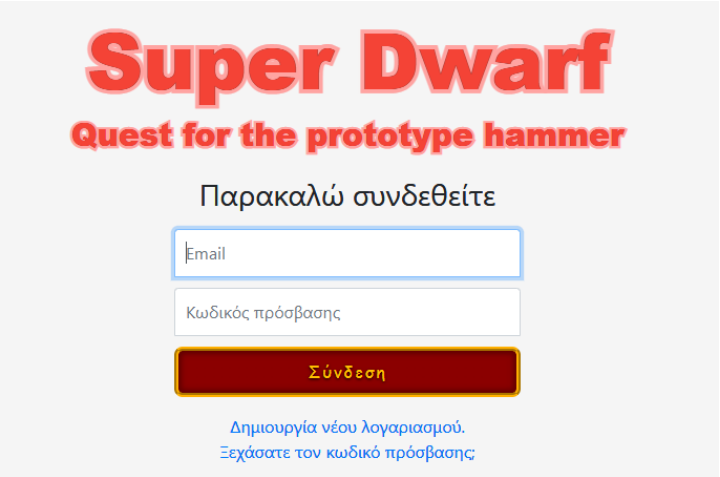
Η θεωρία αυτού του επιπέδου παρουσιάζει τα συμβάντα που προκαλούνται με τη χρήση του ποντικιού, με τη χρήση του πληκτρολογίου και σκρολάροντας σε μια σελίδα. Επίσης, εξηγείται και η λογική του event delegation που αποτελεί ένα από τα πιο ισχυρά pattern χειρισμού συμβάντων. Το επίπεδο περιέχει δύο δραστηριότητες σύνταξης κώδικα και μια ερώτηση πολλαπλής επιλογής, η οποία αφορά τα συμβάντα του πληκτρολογίου. Η αρχική δραστηριότητα συμπλήρωσης κώδικα απαιτεί τη δημιουργία μιας συνάρτησης με την οποία θα προσθέσει ένα listener χρησιμοποιώντας τη μέθοδο addEventListener σε ένα συγκεκριμένο element. Για να βρει ο χρήστης το element θα πρέπει να περνάει ως όρισμα στη συνάρτηση το id που αναφέρεται στην εκφώνηση της άσκησης. Στη συνέχεια,

κάνοντας κλικ σε αυτό το κουμπί θα πρέπει να γίνεται άσπρο το background color μιας συγκεκριμένης παραγράφου. Η τελευταία δραστηριότητα έχει ως στόχο την εφαρμογή του event delegation και ζητάει από τον χρήστη τη δημιουργία μιας συνάρτησης η οποία βρίσκει το table element που εμφανίζεται στον κόσμο του παιχνιδιού και προσθέτει σε αυτό ένα listener με σκοπό την αλλαγή του background color του target του event σε άσπρο.

4.3 Τελικές οθόνες χρήστη

4.3.1 Είσοδος και εγγραφή χρήστη

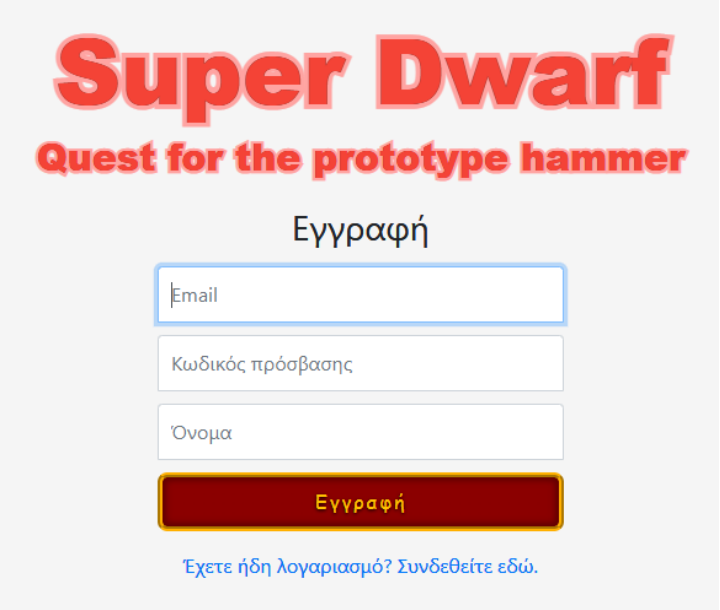
Αρχικά ο χρήστης κατευθύνεται στην οθόνη εισόδου (Εικόνα 14) στην οποία του ζητείται να συμπληρώσει τη διεύθυνση ηλεκτρονικού ταχυδρομείου και το συνθηματικό του. Εάν η σύνδεσή του είναι επιτυχής τότε οδηγείται στην αρχική οθόνη του παιχνιδιού. Σε περίπτωση που ο χρήστης δώσει λάθος στοιχεία τότε εμφανίζονται τα κατάλληλα μηνύματα σφάλματος σε αυτόν. Εάν ο χρήστης δεν έχει λογαριασμό τότε θα πρέπει να δημιουργήσει πατώντας το σύνδεσμο "Δημιουργία νέου λογαριασμού".



Εικόνα 14: Είσοδος χρήστη

Το παιχνίδι απαιτεί από το χρήστη να δημιουργήσει λογαριασμό συμπληρώνοντας τη φόρμα εγγραφής (Εικόνα 15). Στόχος αυτής της απαίτησης αποτελεί η αποθήκευση της προόδου του παίκτη και η δυνατότητα να ασχολείται με το παιχνίδι όποτε το επιθυμεί. Τα στοιχεία που είναι απαραίτητα να συμπληρώσει είναι η διεύθυνση ηλεκτρονικού ταχυδρομείου, ένα ψευδώνυμο και ένα συνθηματικό. Αφού ολοκληρωθεί η εγγραφή ο

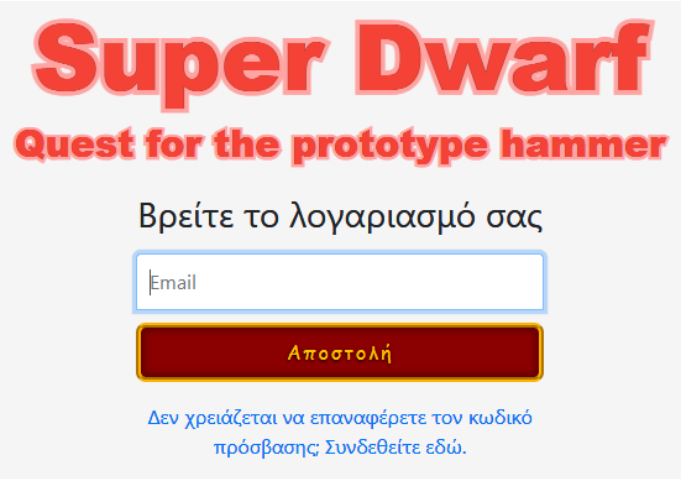
χρήστης κατευθύνεται στην οθόνη εισόδου και συμπληρώνει τα στοιχεία σύνδεσης για να εισέλθει στο παιχνίδι.



Εικόνα 15: Εγγραφή χρήστη

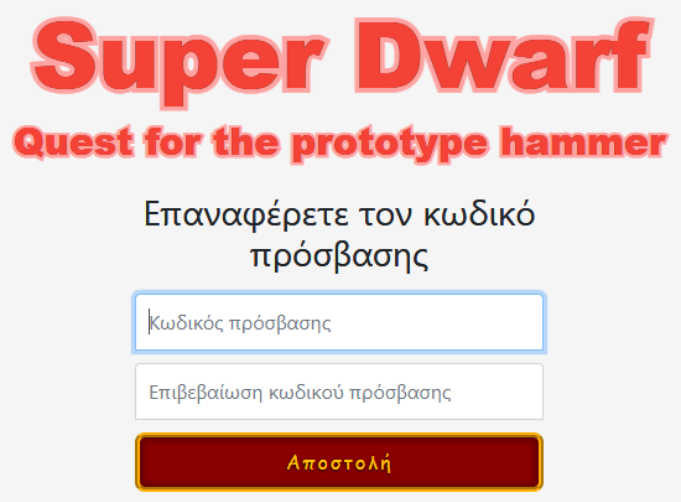
4.3.2 Επαναφορά συνθηματικού

Το παιχνίδι δίνει τη δυνατότητα στον χρήστη να επαναφέρει τον κωδικό του σε περίπτωση που τον χάσει. Πατώντας το σύνδεσμο "Ξεχάσατε τον κωδικό πρόσβασης" κατευθύνεται στη σελίδα ανάκτησης συνθηματικού (Εικόνα 16) όπου του ζητάει τη διεύθυνση ηλεκτρονικού ταχυδρομείου για να αποσταλεί ο σύνδεσμος επαναφοράς.



Εικόνα 16: Ανάκτηση συνθηματικού χρήστη

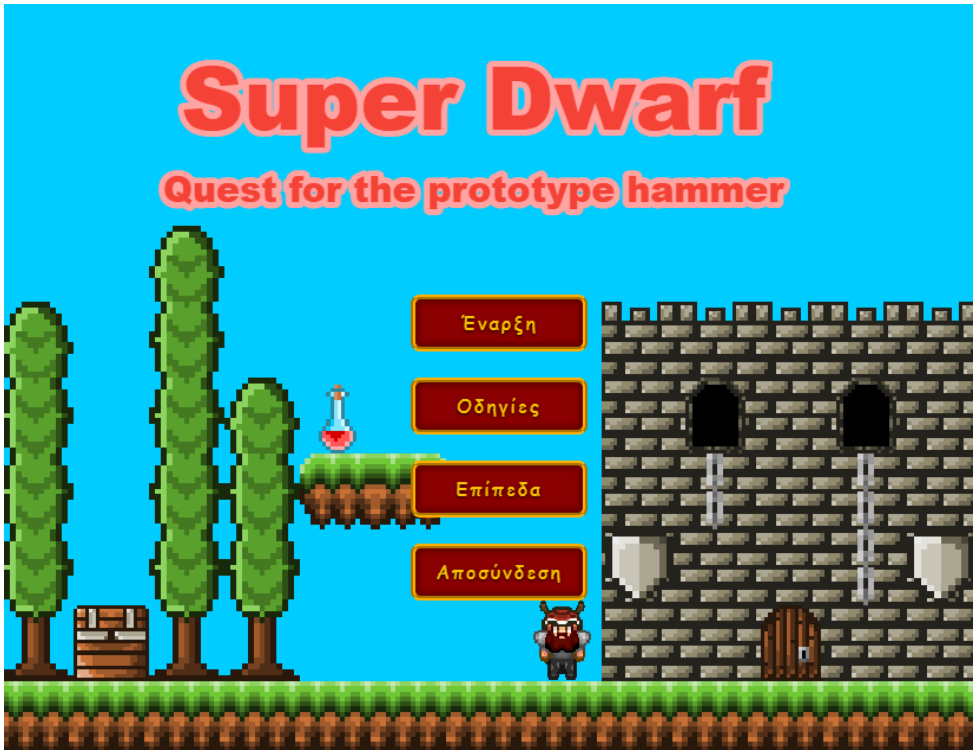
Χρησιμοποιώντας το σύνδεσμο επαναφοράς που του έχει σταλεί ο χρήστης κατευθύνεται στην οθόνη επαναφοράς (Εικόνα 17) και συμπληρώνει το νέο συνθηματικό στα πεδία “Κωδικός πρόσβασης” και “Επιβεβαίωση κωδικού πρόσβασης”. Με την ολοκλήρωση της επαναφοράς θα οδηγηθεί στην οθόνη σύνδεσης.



Εικόνα 17: Επαναφορά συνθηματικού χρήστη

4.3.3 Αρχική οθόνη και κεντρικό μενού

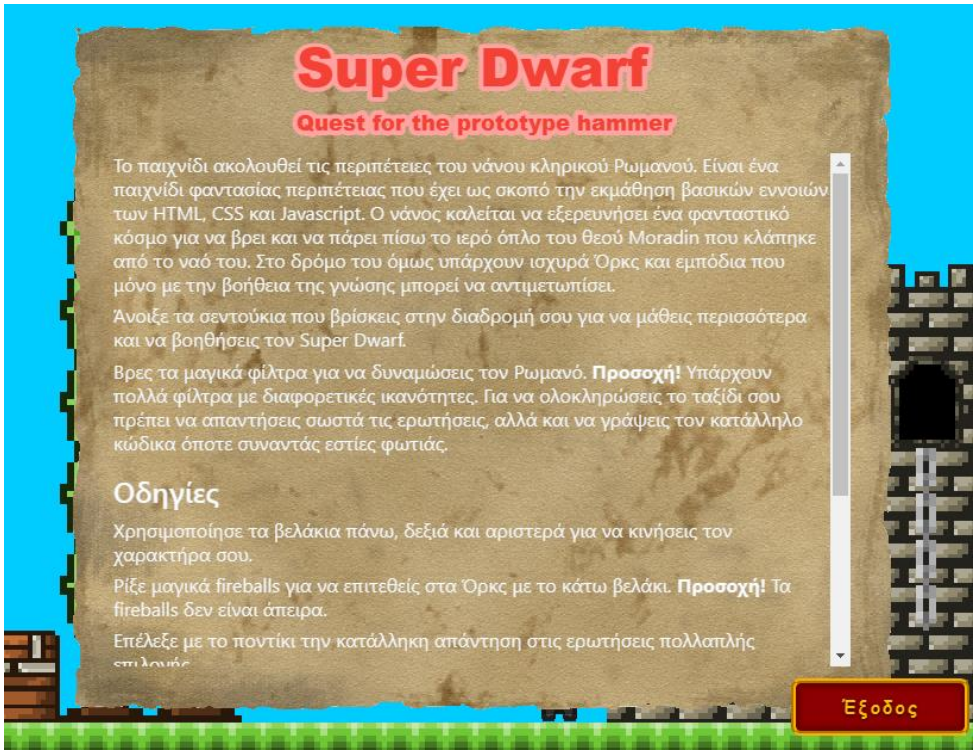
Αφού συνδεθεί ο χρήστης στο παιχνίδι κατευθύνεται στην αρχική οθόνη (Εικόνα 18), όπου εμφανίζεται το κεντρικό μενού του παιχνιδιού. Η πρώτη επιλογή στο μενού είναι το κουμπί έναρξη με το οποίο ξεκινάει το παιχνίδι. Όταν εισέρχεται ο χρήστης για πρώτη φορά το παιχνίδι ξεκινάει από το επίπεδο ένα, ενώ τις επόμενες φορές ξεκινάει από το επίπεδο που το είχε σταματήσει ο χρήστης. Σε περίπτωση που ο χρήστης αποχωρήσει ενώ ολοκλήρωσε ένα επίπεδο τότε θα ξεκινήσει από το επόμενο. Το δεύτερο κουμπί εμφανίζει την οθόνη με την περιγραφή και τις οδηγίες του παιχνιδιού και το τρίτο κουμπί οδηγεί στην οθόνη με τα επίπεδα. Αυτές οι οθόνες θα περιγραφούν στη συνέχεια του κεφαλαίου. Τέλος, με το κουμπί αποσύνδεση ο χρήστης βγαίνει από το παιχνίδι.



Εικόνα 18: Αρχική οθόνη και κεντρικό μενού

4.3.4 Οθόνη σεναρίου και οδηγιών

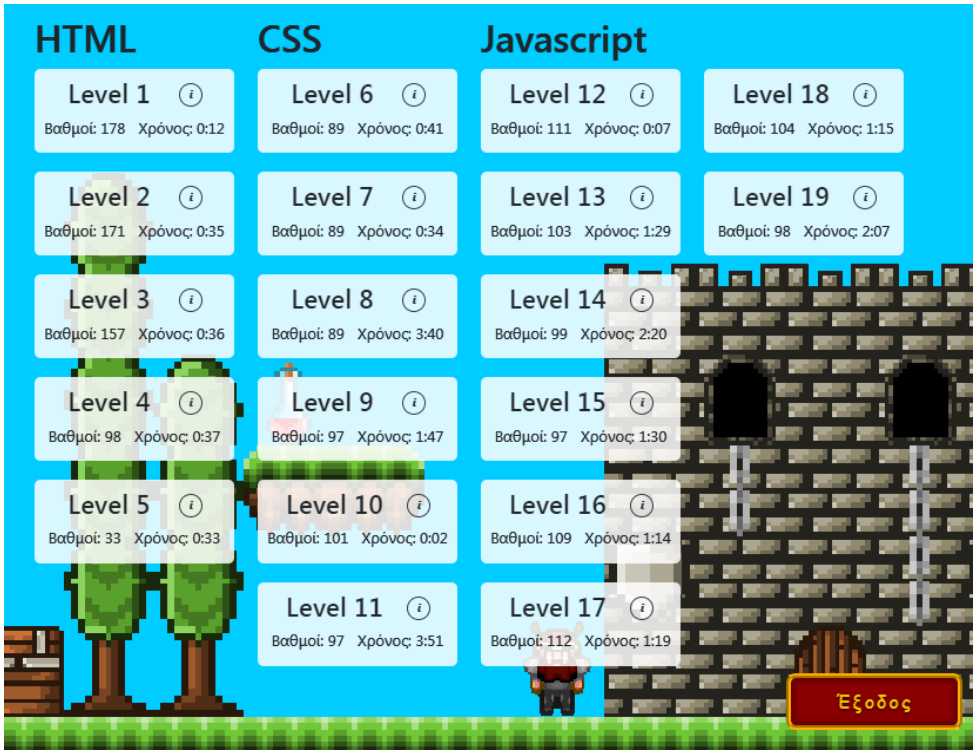
Στην οθόνη αυτή (Εικόνα 19) παρουσιάζεται το σενάριο του παιχνιδιού και οι οδηγίες με τις οποίες παίζεται το παιχνίδι. Μέσα από το σενάριο περιγράφεται ο διδακτικός σκοπός του παιχνιδιού και ο στόχος του παίκτη που είναι να βοηθήσει τον ήρωα του παιχνιδιού να βρει το ιερό όπλο του θεού Moradin που κλάπηκε από τα εχθρικά Όρκς. Στην συνέχεια ακολουθούν οι οδηγίες που ενημερώνουν το χρήστη για τον τρόπο χρήσης του παιχνιδιού. Η πλοήγηση του χαρακτήρα πραγματοποιείται με τα βελάκια πάνω, δεξιά και αριστερά, ενώ με το κάτω βελάκι πραγματοποιείται η μαγική επίθεση. Επίσης, ενημερώνει το χρήστη για τις δραστηριότητες του παιχνιδιού οι οποίες θα περιγραφούν στη συνέχεια. Με το κουμπί έξοδος μπορεί να επιστρέψει στην αρχική οθόνη.



Εικόνα 19: Σενάριο και οδηγίες του παιχνιδιού

4.3.5 Οθόνη επιπέδων

Στην οθόνη αυτή παρουσιάζονται τα επίπεδα και οι διδακτικοί στόχοι που έχει το καθένα. Όπως φαίνεται στην εικόνα 20 το παιχνίδι περιλαμβάνει συνολικά 19 επίπεδα τα οποία χωρίζονται σε τρεις διακριτές ενότητες. Η πρώτη ενότητα η οποία αφορά την HTML περιλαμβάνει τα πρώτα πέντε επίπεδα. Η δεύτερη ενότητα που καλύπτει τη διδασκαλία του CSS περιλαμβάνει τα επίπεδα 6 έως 11. Η τελευταία ενότητα της JavaScript περιέχει τα επίπεδα 12 έως 19. Αρχικά όλα τα επίπεδα εκτός από το πρώτο είναι κλειδωμένα. Για να ανοίξουν τα επίπεδα θα πρέπει ο παίκτης να ολοκληρώσει το αμέσως προηγούμενο.



Εικόνα 20: Οθόνη επιπέδων του παιχνιδιού

Κάθε επίπεδο αναπαρίσταται ως ένα λευκό πλακίδιο πάνω στο οποίο αναγράφεται ο τίτλος αυτού, η καλύτερη βαθμολογία που συγκέντρωσε ο χρήστης παίζοντας και ο καλύτερος χρόνος που σημείωσε κατά την ολοκλήρωση του επιπέδου. Επίσης, υπάρχει και το κουμπί με το χαρακτήρα i το οποίο όταν πατηθεί παρουσιάζει τους διδακτικούς στόχους του επιπέδου. Το επίπεδο ξεκινάει όταν ο χρήστης κάνει κλικ πάνω στον τίτλο αυτού. Αφού, ο χρήστης ξεκλειδώσει ένα επίπεδο αυτό παραμένει διαθέσιμο για πάντα και μπορεί να το ξαναπαίξει οποιαδήποτε στιγμή. Τα μη ενεργά επίπεδα ξεχωρίζουν διότι απεικονίζονται με πιο αχνό χρώμα. Με το κουμπί έξοδος μπορεί να επιστρέψει στην αρχική οθόνη.

4.3.6 Βασικά στοιχεία επιπέδου

Με την έναρξη του επιπέδου ο παίκτης κατευθύνεται στον κόσμο του παιχνιδιού και βλέπει τον ήρωα του στην αρχική του θέση (Εικόνα 21). Πάνω από τον ήρωα εμφανίζεται μία φούσκα ομιλίας που τον πληροφορεί για τον στόχο που πρέπει να εκπληρώσει σε κάθε επίπεδο. Ο στόχος είναι ο ίδιος σε όλα τα επίπεδα και σύμφωνα με τη φούσκα πρέπει να ολοκληρώσει σωστά τρεις δραστηριότητες για να προχωρήσει στο επόμενο. Το σκηνικό των πρώτων 11 επιπέδων που αφορούν τις ενότητες HTML και CSS διαδραματίζεται σε δάσος ενώ των υπολοίπων που ανήκουν στην ενότητα της JavaScript

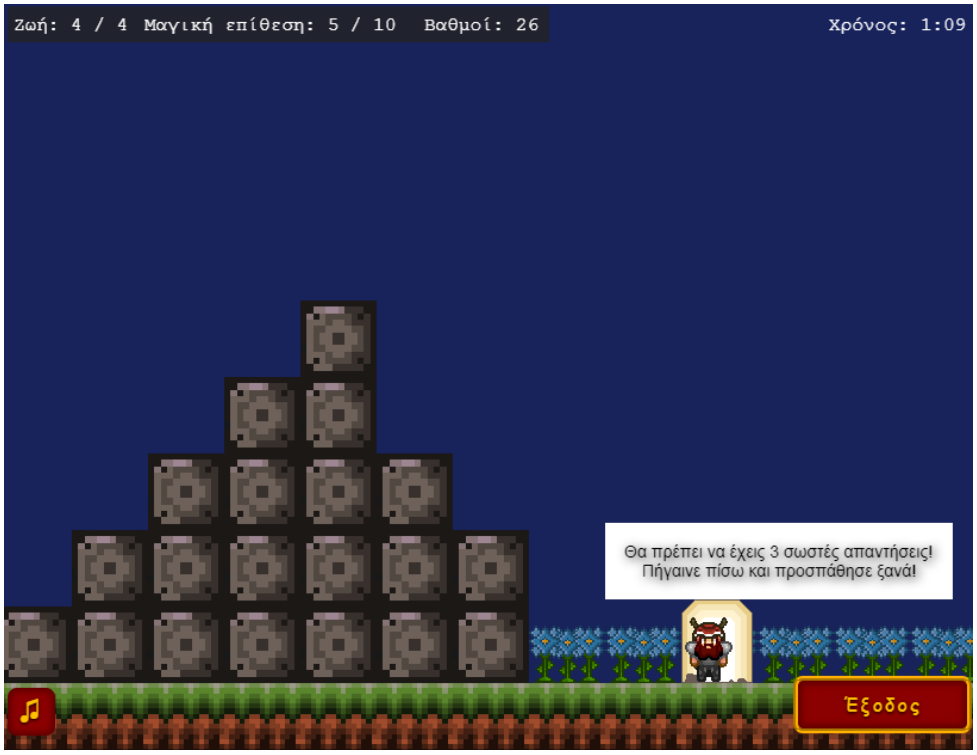
σε κάστρο. Τα επίπεδα του δάσους λαμβάνουν χώρα σε διαφορετικές στιγμές της ημέρας και αυτό διακρίνεται από το χρώμα του ουρανού και την φωτεινότητα του επιπέδου. Για παράδειγμα, στο επίπεδο 1 που διαδραματίζεται το πρωί ο ουρανός είναι ροζ και η φωτεινότητα κανονική, ενώ το επίπεδο 4 που είναι βράδυ ο ουρανός είναι μαύρος και η φωτεινότητα χαμηλή. Επίσης, η μουσική στα επίπεδα αλλάζει ανάλογα με τη μαθησιακή ενότητα που βρίσκεται ο παίκτης.

Εικόνα 21: Έναρξη επιπέδου

Όπως φαίνεται στην Εικόνα 21, πάνω αριστερά στην οθόνη του επιπέδου εμφανίζονται πληροφορίες σχετικά με τη διαθέσιμη ζωή του ήρωα, τις διαθέσιμες μαγικές επιθέσεις και τους βαθμούς που συλλέγει. Πάνω δεξιά παρουσιάζεται το χρονόμετρο το οποίο δείχνει πόσο χρόνο αφιερώνει στο συγκεκριμένο επίπεδο ο παίκτης. Επίσης, κάτω αριστερά υπάρχει ένα κουμπί με το οποίο μπορεί να απενεργοποιήσει και να ενεργοποιήσει τη μουσική και κάτω δεξιά υπάρχει το κουμπί έξοδος με το οποίο μπορεί να αποχωρήσει από το επίπεδο. Εάν πατήσει το κουμπί έξοδος εμφανίζεται στον χρήστη ένα μήνυμα που τον ενημερώνει ότι σε περίπτωση που αποχωρήσει από το παιχνίδι θα χάσει την πρόοδό του και ζητάει την επιβεβαίωση του (Εικόνα 22).

Εικόνα 22: Επιβεβαίωση αποχώρησης από το επίπεδο

Στο τέλος του επιπέδου βρίσκεται η πόρτα εξόδου που οδηγεί τον ήρωα στο επόμενο επίπεδο. Εάν ο παίκτης έχει ολοκληρώσει τον στόχο του επιπέδου τότε οδηγείται στην οθόνη επιτυχίας. Σε περίπτωση όμως που δεν έχει ολοκληρώσει σωστά τρεις δραστηριότητες εμφανίζεται μήνυμα που τον ενημερώνει και τον παροτρύνει να πάει πίσω και να προσπαθήσει ξανά (Εικόνα 23).



Εικόνα 23: Σημείο τερματισμού επιπέδου

4.3.7 Οθόνη ολοκλήρωσης επιπέδου

Όταν ο παίκτης τερματίσει με επιτυχία το παιχνίδι οδηγείται στην οθόνη τερματισμού η οποία περιέχει σχετικά μηνύματα (Εικόνα 24). Αρχικά, δίνει στον παίκτη συγχαρητήρια και τον ενημερώνει ότι έχει ολοκληρώσει το επίπεδο, για τους συνολικούς πόντους που μάζεψε και για το χρόνο που κατάφερε να το τερματίσει. Εμφανίζει σε αυτόν ένα κουμπί που θα τον οδηγήσει στο επόμενο επίπεδο με τον τίτλο του επόμενου επιπέδου και ένα κουμπί που τον κατευθύνει στο κύριο μενού. Η εικόνα που υπάρχει στο φόντο αλλάζει ανάλογα με την τοποθεσία που διαδραματίζονται τα επίπεδα. Δηλαδή, παρουσιάζεται δάσος ή κάστρο ανάλογα με τα επίπεδα.

Εικόνα 24: Επιτυχής ολοκλήρωση επιπέδου

Εάν ο παίκτης δεν κατάφερε να ολοκληρώσει το επίπεδο είτε γιατί έχασε όλες του τις ζωές είτε γιατί έπεσε πάνω στο εμπόδιο με τα καρφιά τότε εμφανίζεται η οθόνη τερματισμού που τον ενημερώνει ότι το παιχνίδι έχει τελειώσει και ότι τα Όρκς τον έχουν αιχμαλωτίσει. Σε αυτή την οθόνη ο παίκτης έχει τη δυνατότητα να ξαναπαίξει το ίδιο επίπεδο πατώντας στο κουμπί με τον τίτλο του επιπέδου ή να επιστρέψει στο αρχικό μενού μέσα από το κουμπί “Μενού”. Επίσης, και σε αυτή την περίπτωση το φόντο αλλάζει ανάλογα το επίπεδο.

Εικόνα 25: Μη επιτυχής ολοκλήρωση επιπέδου

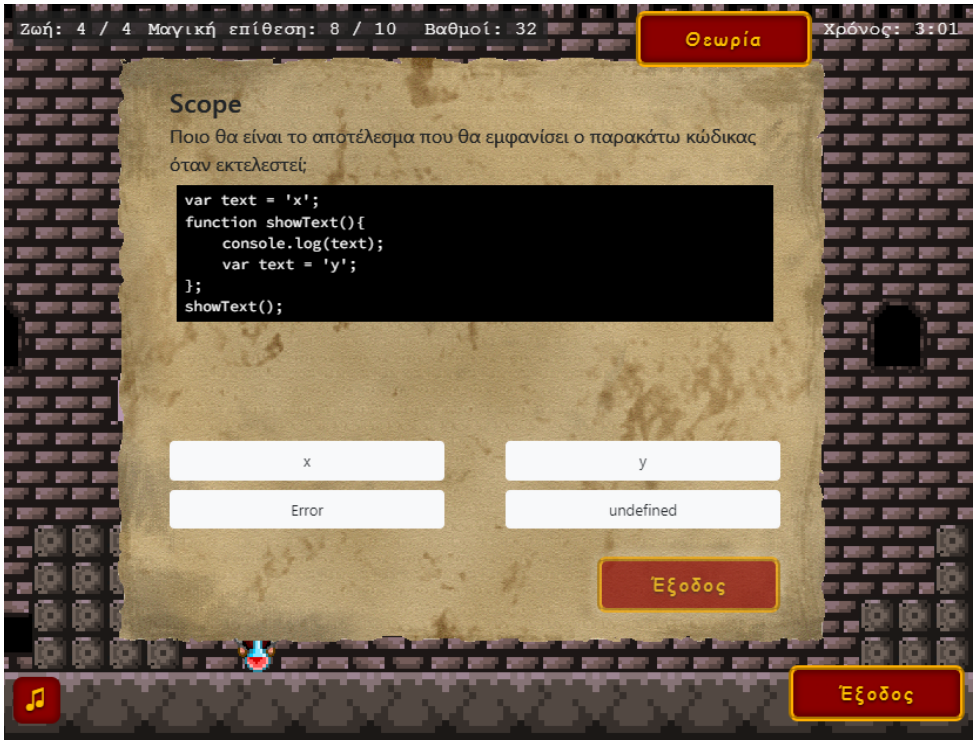
4.3.8 Οθόνη παρουσίασης θεωρίας

Η οθόνη που προβάλλει τη θεωρία σε διάφορα σημεία του επιπέδου εμφανίζεται όταν ο παίκτης έρχεται σε επαφή με ένα σεντούκι και πατάει το πλήκτρο F. Αρχικά, παρουσιάζεται ο τίτλος και έπειτα ακολουθεί η θεωρία σε μορφή κειμένου με φόντο την εικόνα μιας περγαμηνής (Εικόνα 26). Τα παραδείγματα κώδικα που περιέχονται σε αυτή εμφανίζονται σε μαύρο πλαίσιο με λευκά γράμματα. Για να κλείσει τη θεωρία ο χρήστης μπορεί να πατήσει το κουμπί “Έξοδος” κάτω δεξιά.

Εικόνα 26: Παρουσίαση θεωρίας

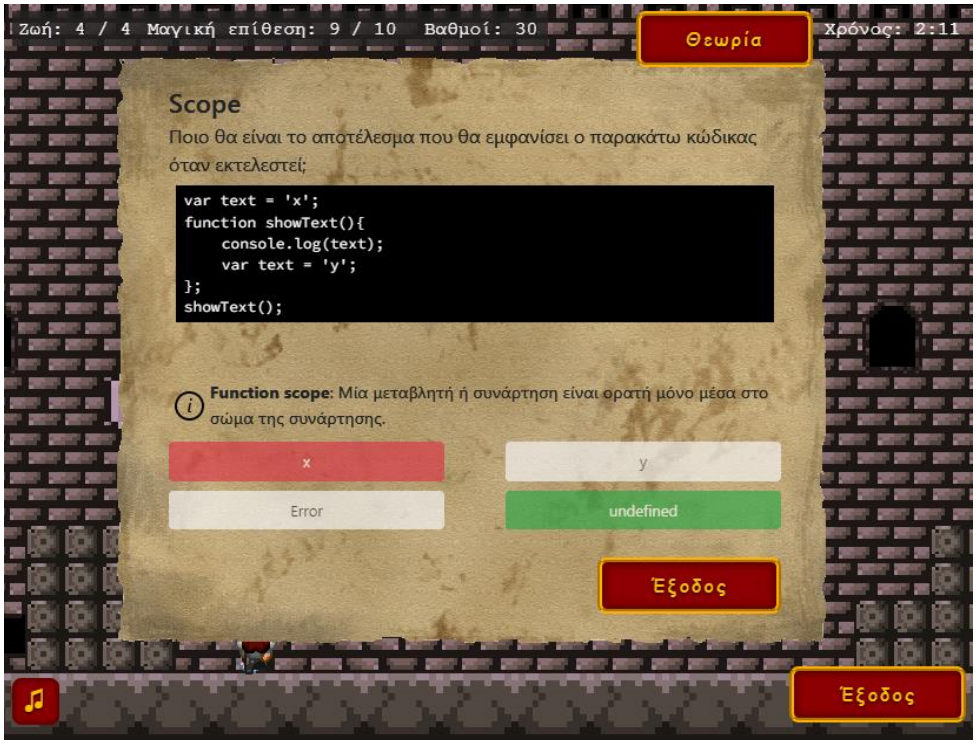
4.3.9 Οθόνη ερώτησης πολλαπλής επιλογής

Μία από τις δραστηριότητες του παιχνιδιού είναι η ερώτηση πολλαπλής επιλογής, η οποία εμφανίζεται όταν ο παίκτης πατήσει το πλήκτρο F κατά την επαφή με κάποιο μαγικό φίλτρο. Αρχικά εμφανίζεται ο τίτλος της ερώτησης που σχετίζεται με το περιεχόμενο και ακολουθεί η περιγραφή του ζητούμενου της ερώτησης. Το περιεχόμενο μπορεί να αφορά τη θεωρία ή την εκτέλεση του κομματιού ενός κώδικα. Έπειτα ακολουθούν με τη μορφή λευκών κουμπιών οι διαθέσιμες απαντήσεις. Τέλος, κάτω δεξιά υπάρχει το κουμπί “Έξοδος”, το οποίο κλείνει την οθόνη αλλά είναι απενεργοποιημένο μέχρι να δοθεί κάποια απάντηση (Εικόνα 27).



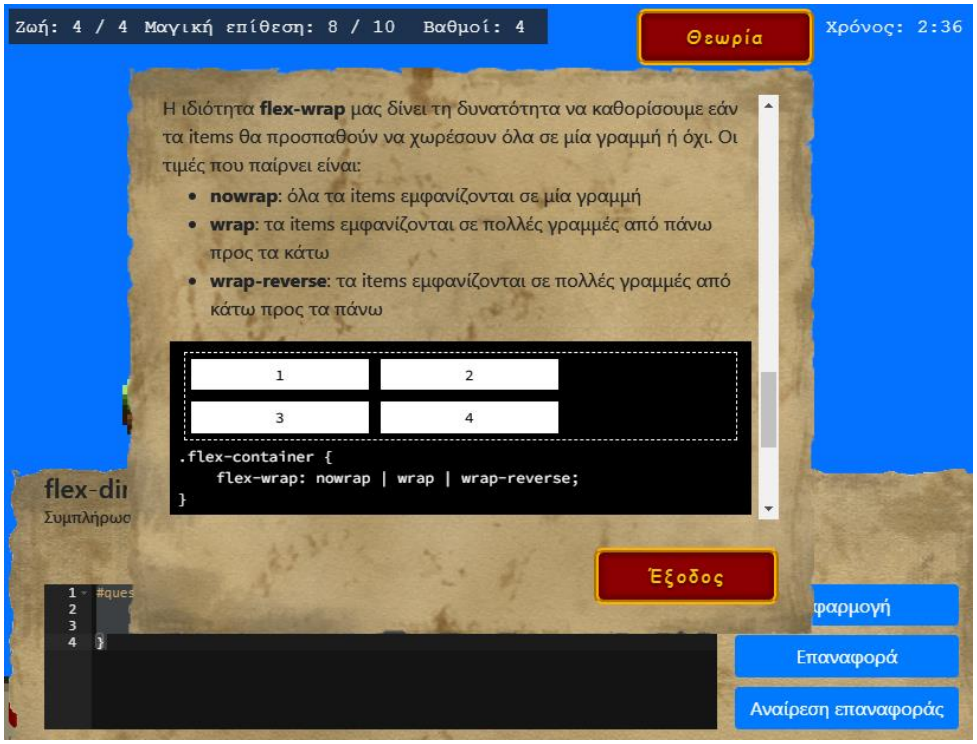
Εικόνα 27: Ερώτηση πολλαπλής επιλογής

Αφού επιλέξει κάποια απάντηση ο χρήστης απενεργοποιούνται τα κουμπιά με τις απαντήσεις, εμφανίζεται πάνω από αυτές μία πρόταση που περιέχει επιπλέον πληροφορίες για τη σωστή απάντηση και ενεργοποιείται το κουμπί έξοδος. Σε περίπτωση που ο χρήστης επιλέξει τη σωστή απάντηση τότε το κουμπί χρωματίζεται με πράσινο χρώμα. Σε περίπτωση που η επιλογή του είναι λάθος τότε αυτή χρωματίζεται με κόκκινο χρώμα και η σωστή απάντηση με πράσινο (Εικόνα 28).



Εικόνα 28: Ερώτηση πολλαπλής επιλογής με λάθος επιλογή απάντησης

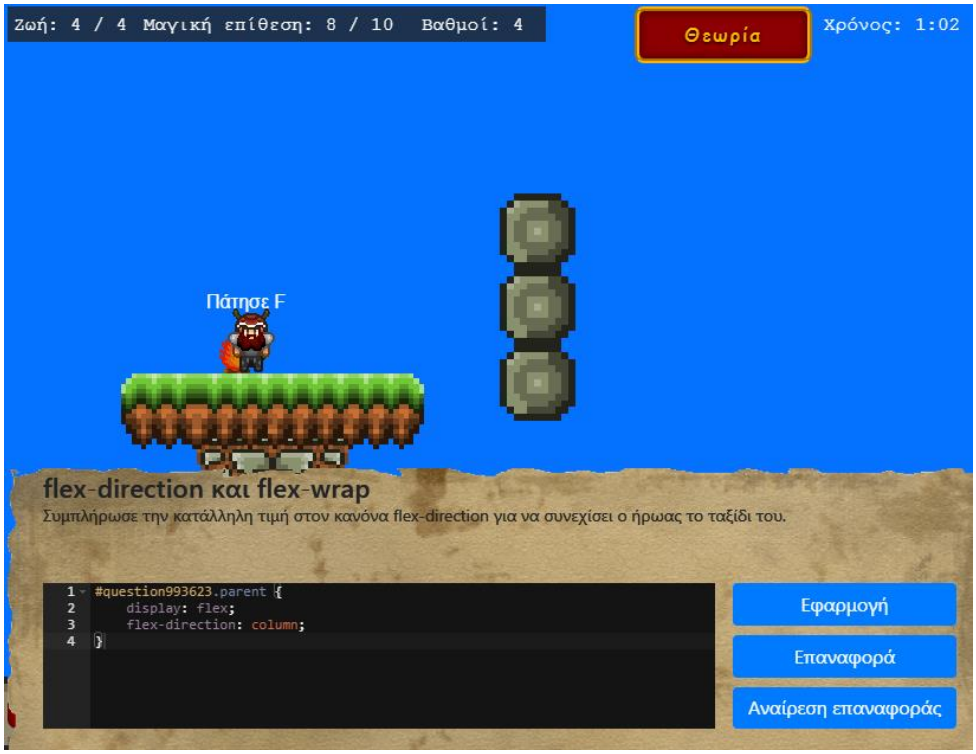
Ένας ακόμη στοιχείο που εμφανίζεται είναι το κουμπί “Θεωρία” που είναι τοποθετημένο δίπλα στο χρονόμετρο. Πατώντας ο χρήστης μπορεί να διαβάσει ξανά τη θεωρία που έχει ήδη μελετήσει στο επίπεδο που βρίσκεται (Εικόνα 29). Με το κουμπί “Έξοδος” μπορεί να επιστρέψει στη δραστηριότητα. Το ίδιο ισχύει και για τη δραστηριότητα σύνταξης κώδικα.



Εικόνα 29: Εμφάνιση θεωρίας από το κουμπί "Θεωρία"

4.3.10 Οθόνη συντάκτη κώδικα

Η δεύτερη δραστηριότητα του παιχνιδιού αφορά τη συμπλήρωση κώδικα και ενεργοποιείται όταν ο ήρωας έρχεται σε επαφή με τη φωτιά στο έδαφος σε συνδυασμό με το πάτημα του πλήκτρου F. Όταν ξεκινάει η δραστηριότητα εμφανίζεται στο κάτω μέρος της οθόνης μια περγαμηνή η οποία περιέχει τις οδηγίες που πρέπει να ακολουθήσει ο χρήστης, ο συντάκτης κώδικα και τα κουμπιά εκτέλεσης (Εικόνα 30). Σε κάποιες περιπτώσεις υπάρχει ήδη κώδικας στο συντάκτη και ο χρήστης πρέπει να κάνει αλλαγές, ενώ σε άλλες πρέπει να συμπληρώσει κώδικα εξ αρχής.



Εικόνα 30: Εμφάνιση συντάκτη κώδικα

Ο κώδικας εκτελείται μόλις ο παίκτης πατήσει το κουμπί “Εφαρμογή”. Εάν ο κώδικας είναι εκτελέσιμος και περάσει τους απαραίτητους ελέγχους τότε φαίνονται άμεσα τα αποτελέσματα στον κόσμο του παιχνιδιού. Στην αντίθετη περίπτωση εμφανίζονται μηνύματα ανατροφοδότησης μέσα σε μια φούσκα ομιλίας πάνω από το συντάκτη (Εικόνα 31). Το κουμπί “Επαναφορά” αντικαθιστά τον κώδικα στον συντάκτη με τον αρχικό κώδικα, ενώ με το κουμπί “Αναίρεση επαναφοράς” ο κώδικας επιστρέφει στην κατάσταση που βρισκόταν πριν πατηθεί το κουμπί “Επαναφορά”.

Εικόνα 31: Εμφάνιση μηνύματος ανατροφοδότησης

Όπως αναφέρθηκε, κατά την εκτέλεση του κώδικα γίνονται εμφανείς αλλαγές στον κόσμο του παιχνιδιού. Αυτές οι αλλαγές ομαδοποιούνται σε συνολικά πέντε κατηγορίες. Στην πρώτη μορφή δραστηριότητας οι φλόγες βρίσκονται σε σημεία όπου δεν υπάρχει διαδρομή να συνεχίσει ο ήρωας το ταξίδι του και όταν εκκινείται η δραστηριότητα εμφανίζονται στον κόσμο του παιχνιδιού βράχια τα οποία ο παίκτης πρέπει να τοποθετήσει κατάλληλα (Εικόνα 30). Με την εκτέλεση του κώδικα αλλάζει και η τοποθέτηση αυτών (Εικόνα 31). Στη δεύτερη περίπτωση υπάρχουν κρυφά μηνύματα στον κόσμο του παιχνιδιού τα οποία εμφανίζονται με την εκτέλεση του κατάλληλου κώδικα και συνδέουν τις ερωτήσεις μεταξύ τους (Εικόνας 33). Η τρίτη ομάδα περιλαμβάνει δραστηριότητες που βρίσκονται σε σημεία όπου διακόπτεται η διαδρομή του παιχνιδιού (Εικόνα 34) και με τη σωστή εκτέλεση κώδικα εμφανίζονται πλατφόρμες πάνω στις οποίες μπορεί να κινηθεί ο ήρωας (Εικόνα 35). Στην τέταρτη μορφή οι φλόγες είναι τοποθετημένες μπροστά σε αδιέξοδα τα οποία ο παίκτης πρέπει να εξαφανίσει γράφοντας κώδικα (Εικόνα 36). Τέλος, η πέμπτη κατηγορία περιλαμβάνει δραστηριότητες οι οποίες εισάγουν HTML elements στον κόσμο του παιχνιδιού και ο παίκτης πρέπει να τα χειριστεί κατάλληλα γράφοντας κώδικα (Εικόνα 37).

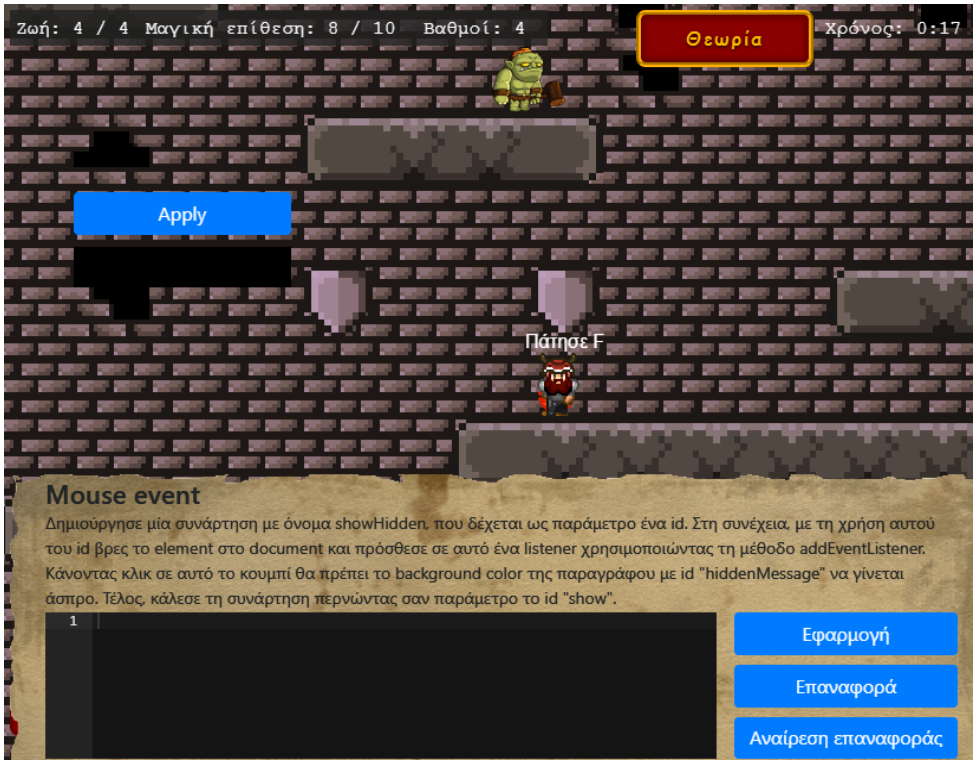
Εικόνα 32: Επανατοποθέτηση βράχων μετά την εκτέλεση κώδικα

Εικόνα 33: Εμφάνιση κρυφού μηνύματος μετά την εκτέλεση κώδικα

Εικόνα 34: Έδαφος που δεν επιτρέπει τη συνέχεια του ταξιδιού

Εικόνα 35: Εμφάνιση πλατφόρμας μετά την εκτέλεση κώδικα

Εικόνα 36: Σημείο με αδιέξοδο στο επίπεδο



Εικόνα 37: Εισαγωγή HTML elements στον κόσμο του παιχνιδιού

5 Υλοποίηση του παιχνιδιού

5.1 Υποδομή του server

Για τη δημιουργία του server και του API που είναι υπεύθυνο για την εγγραφή/είσοδο του χρήστη αλλά και την ενημέρωση των δεδομένων της προόδου του μελετήθηκαν αρκετοί οδηγοί σχετικοί με τις τεχνολογίες που αναφέρθηκαν στην ενότητα 3.3. Οι οδηγοί που αποτέλεσαν κύρια πηγή μελέτης είναι ο “Building a REST API with Express, Node, and MongoDB” που παρέχεται από τη MongoDB και ο “How To Implement API Authentication with JSON Web Tokens and Passport” που γράφτηκε από τον Obielum Godson, προγραμματιστή στη DigitalOcean.

Στον Κώδικα 1 παρουσιάζεται το αρχείο app.js το οποίο αποτελεί το κύριο σημείο της εφαρμογής για τον Node.js server. Αρχικά, στη γραμμή 1 γίνεται η δήλωση του module dotenv, το οποίο χρησιμοποιείται για την ανάγνωση του αρχείου .env που περιέχει τα ευαίσθητα δεδομένα της εφαρμογής, όπως για παράδειγμα το URI σύνδεσης στη βάση δεδομένων. Μετά την ανάγνωση του .env αυτές οι τιμές είναι διαθέσιμες ως μεταβλητές περιβάλλοντος. Στις γραμμές 2 έως 5 δηλώνονται οι υπόλοιπες βιβλιοθήκες που θα χρησιμοποιηθούν στο αρχείο και στις γραμμές 7 έως 9 εισάγονται τα αρχεία που περιέχουν τα routes της εφαρμογής. Στο αρχείο main.js περιέχονται τα routes που αφορούν βασικές λειτουργικότητες όπως η σύνδεση, στο αρχείο secure.js αυτά που σχετίζονται με την ανανέωση των δεδομένων και στο password.js αυτά που αφορούν την επαναφορά κωδικού. Ακολουθεί η σύνδεση με τη βάση δεδομένων με τη βοήθεια των μεθόδων του module mongoose και κάνοντας χρήση του URI της βάσης που έρχεται από το αρχείο .env. Στη γραμμή 24 αρχικοποιείται η εφαρμογή και ορίζονται οι κατάλληλες ρυθμίσεις σε αυτή ώστε να γίνεται πιο εύκολα η ανάλυση των δεδομένων που έρχονται από τα POST αιτήματα. Έπειτα, εισάγεται το αρχείο auth.js το οποίο είναι υπεύθυνο για την αυθεντικοποίηση του χρήστη. Στη συνέχεια, προστίθεται το route που οδηγεί στη βασική σελίδα του παιχνιδιού (γραμμές 32 - 34) που είναι η game.html, ενημερώνεται η εφαρμογή με τη χρήση του express.static ότι για την προβολή στατικών αρχείων χρησιμοποιείται ο φάκελος public και ορίζεται το route για το root path της εφαρμογής. Στις γραμμές 42 έως 44 δηλώνονται στην εφαρμογή τα routes που θα χρησιμοποιήσει. Επίσης, ορίζονται δύο ενδιάμεσοι χειριστές, ένας που χειρίζεται τα σφάλματα που προκύπτουν και ένας που ενεργοποιείται όταν ο χρήστης καλέσει κάποιο route που δεν έχει οριστεί. Τέλος, δηλώνεται η θύρα στην οποία ακούει ο διακομιστής.

app.js

```
require("dotenv").config();
const express = require("express");
const mongoose = require("mongoose");
const cookieParser = require("cookie-parser");
const passport = require("passport");

const routes = require("./routes/main");
const secureRoutes = require("./routes/secure");
const passwordRoutes = require("./routes/password");

// setup mongodb connection
const uri = process.env.MONGO_CONNECTION_URL;
mongoose.connect(uri, { useNewUrlParser : true, useUnifiedTopology: true, useCreateIndex: true });
mongoose.connection.on("error", (error) => {
    console.log(error);
    process.exit(1);
});
mongoose.connection.on("connected", function () {
    console.log("connected to mongo");
});
mongoose.set("useFindAndModify", false);

// an express app instance
const app = express();
app.use(express.urlencoded({ extended: true }));
app.use(express.json());
app.use(cookieParser());

// require passport auth
require("./auth/auth");

app.get("/game.html", passport.authenticate("jwt", { session : false }), function (req, res) {
  res.sendFile(__dirname + "/public/game.html");
});

app.use(express.static(__dirname + "/public"));
app.get("/", function (req, res) {
  res.sendFile(__dirname + "/index.html");
});

// main routes
app.use("/", routes);
app.use("/", passwordRoutes);
app.use("/", passport.authenticate("jwt", { session : false }), secureRoutes);

// catch other routes
app.use((req, res, next) => {
  res.status(404).json({ message: "404 - Not Found" });
});

// errors handler
app.use((err, req, res, next) => {
  console.log(err.message);
  res.status(err.status || 500).json({ error: err.message });
});

app.listen(process.env.PORT || 3000, () => {
  console.log(`Server started on port ${process.env.PORT || 3000}`);
});
```

Κώδικας 1: Αρχείο app.js

Πριν την ανάλυση των αρχείων που περιέχουν τα routes της εφαρμογής θα παρουσιαστεί το αρχείο auth.js που περιέχει κώδικα για τη προσθήκη στρατηγικών στο module passport (Κώδικας 2). Αρχικά, ορίζονται τα modules passport, passport-local και passport-jwt και το αρχείο userModel που περιέχει τη δομή της βάσης δεδομένων. Στη συνέχεια ορίζεται στο passport μια νέα τοπική στρατηγική, η οποία χρησιμοποιείται κατά την κλήση του route εγγραφής. Η συνάρτηση αυτή δέχεται ως ορίσματα ένα αντικείμενο με τα πεδία usernameField, passwordField και passReqToCallback, και μία callback συνάρτηση. Το πεδίο passReqToCallback με την τιμή true δηλώνει ότι το αντικείμενο του αιτήματος θα μεταβιβαστεί στην callback συνάρτηση. Μέσα στη συνάρτηση καλείται η μέθοδος create της UserModel η οποία είναι υπεύθυνη για τη δημιουργία νέου χρήστη στη βάση δεδομένων και τέλος εκτελείται η συνάρτηση done που μεταβιβάστηκε ως όρισμα. Με την ίδια λογική ορίζεται στο passport και η τοπική στρατηγική που χρησιμοποιείται όταν καλείται το route της σύνδεσης χρήστη. Σε αυτή την callback συνάρτηση γίνεται αναζήτηση του χρήστη στη βάση δεδομένων με τη μέθοδο findOne και αν βρεθεί ο χρήστης καλείται η μέθοδος isValidPassword που επαληθεύει το συνθηματικό που έδωσε με το συνθηματικό που είναι αποθηκευμένο. Σε περίπτωση που το συνθηματικό δεν ισχύει ή δεν βρεθεί ο χρήστης τότε επιστρέφονται μηνύματα λάθους. Τέλος, ορίζεται μια JWT στρατηγική η οποία χρησιμοποιείται για την επαλήθευση του token. Τα ορίσματα που δέχεται είναι ένα αντικείμενο με τα πεδία secretOrKey και jwtFromRequest και μία callback συνάρτηση. Το πεδίο secretOrKey χρησιμοποιείται για την υπογραφή του JWT που δημιουργείται, ενώ το jwtFromRequest είναι μια συνάρτηση που χρησιμοποιείται για να πάρει το jwt που είναι τοποθετημένο στο cookie του αιτήματος. Τέλος, εκτελείται η συνάρτηση done που μεταβιβάστηκε ως όρισμα στην callback συνάρτηση.

auth.js

```
const passport = require("passport");
const localStrategy = require("passport-local").Strategy;
const JWTstrategy = require("passport-jwt").Strategy;
const UserModel = require("../models/userModel");

// user registration handler
passport.use("signup", new localStrategy({
  usernameField: "email",
  passwordField: "password",
  passReqToCallback: true
}, async (req, email, password, done) => {
  try {
    const { name } = req.body;
    const user = await UserModel.create({ email, password, name});
    return done(null, user);
  } catch (error) {
    done(error);
  }
}));

// user login handler
passport.use("login", new localStrategy({
  usernameField: "email",
  passwordField: "password"
}, async (email, password, done) => {
  try {
    const user = await UserModel.findOne({ email });
    if(!user)
      return done(null, false, { message: "User not found" });
    const validate = await user.isValidPassword(password);
    if(!validate)
      return done(null, false, { message: "Wrong Password" });
    return done(null, user, { message: "Logged in Successfully" });
  } catch (error) {
    return done(error);
  }
}));

// verify if token is valid
passport.use(new JWTstrategy({
  secretOrKey: "top_secret",
  jwtFromRequest: function (req) {
    let token = null;
    if(req && req.cookies)
      token = req.cookies["jwt"];
    return token;
  }
}, async (token, done) => {
  try {
    return done(null, token.user);
  } catch (error) {
    done(error);
  }
}));
```

Κώδικας 2: Αρχείο auth.js

Στον Κώδικα 3 ορίζονται τα routes που αφορούν την εγγραφή, σύνδεση και αποσύνδεση του χρήστη και την ανανέωση του token. Ξεκινώντας εισάγονται τα modules που είναι απαραίτητα και με το express.Router δημιουργείται ένα σύστημα δρομολόγησης το οποίο λειτουργεί σαν μια μικρή εφαρμογή express που μπορεί να εισαχθεί στην κύρια εφαρμογή. Αρχικά, προστίθεται το signup route το οποίο αφορά την εγγραφή και χρησιμοποιώντας το passport.authenticate και τη στρατηγική signup (Κώδικας 2) δημιουργεί το νέο χρήστη στη βάση δεδομένων. Στην συνέχεια ορίζεται το login route μέσα στο οποίο εκτελείται το passport.authenticate χρησιμοποιώντας τη στρατηγική login και εισάγοντας μία callback συνάρτηση. Σε αυτή τη συνάρτηση αρχικά ελέγχεται αν υπήρξε κάποιο σφάλμα και αν δεν βρέθηκε χρήστης και επιστρέφει τα κατάλληλα μηνύματα. Σε περίπτωση που δεν προκύψει κάτι από αυτά καλείται η μέθοδος login, η οποία προστίθεται αυτόματα από το module passport, με ορίσματα το αντικείμενο που αντιπροσωπεύει το χρήστη, ένα αντικείμενο με επιλογές και μία callback συνάρτηση. Μέσα στη συνάρτηση με τη βοήθεια της βιβλιοθήκης jsonwebtoken δημιουργούνται δύο JSON web tokens τα οποία χρησιμοποιούν ως φορτίο το αναγνωριστικό και το email του χρήστη. Το κύριο token λήγει σε πέντε λεπτά, ενώ το refreshToken λήγει σε μία ημέρα. Τέλος, τα tokens αποθηκεύονται στο σώμα της απάντησης με τη μέθοδο cookie και στη μνήμη και αποστέλλονται με την απάντηση του αιτήματος (Κώδικας 3 γραμμή 40). Το route token είναι υπεύθυνο για τον έλεγχο της εξουσιοδότησης του χρήστη. Αφού ψάξει και βρει το token που είναι αποθηκευμένο στη μνήμη με βάση αυτό που σταλθηκε στο αντικείμενο του αιτήματος, το ανανεώνει και ενημερώνει τη λίστα στη μνήμη, το αντικείμενο απάντησης και το επιστρέφει. Σε περίπτωση που δεν βρεθεί επιστρέφει το μήνυμα “Unauthorized”. Τέλος, στο logout route που αφορά την αποσύνδεση του χρήστη ελέγχεται εάν στο αντικείμενο του αιτήματος υπάρχουν cookies και αν υπήρχαν τότε αφαιρούνται και ολοκληρώνεται η διαδικασία.

main.js

```
const passport = require("passport");
const express = require("express");
const jwt = require("jsonwebtoken");

const tokenList = { };
const router = express.Router();

router.post("/signup", passport.authenticate("signup", { session: false }), async (req, res, next) => {
  res.status(200).json({ message: "signup successful" });
});

router.post("/login", async (req, res, next) => {
  passport.authenticate("login", async (err, user, info) => {
    try {
      if (err)
        return next(new Error("An Error occured"));
      if(!user)
        return res.status(401).json({ message: "Authentication failed: " + info.message });
      req.login(user, { session: false }, async (error) => {
        if (error)
          return next(error);
        const body = {
          _id: user._id,
          email: user.email
        };

        const token = jwt.sign({ user: body }, "top_secret", { expiresIn: 300 });
        const refreshToken = jwt.sign({ user: body }, "top_secret_refresh", { expiresIn: 86400 });

        res.cookie("jwt", token);
        res.cookie("refreshJwt", refreshToken);

        tokenList[refreshToken] = {
          token,
          refreshToken,
          email: user.email,
          _id: user._id
        };

        return res.status(200).json({ token, refreshToken });
      });
    } catch (error) {
      return next(error);
    }
  })(req, res, next);
});

router.post("/token", (req, res) => {
  const { refreshToken } = req.body;
  if (refreshToken in tokenList) {
    const body = { email: tokenList[refreshToken].email, _id: tokenList[refreshToken]._id };
    const token = jwt.sign({ user: body }, "top_secret", { expiresIn: 300 });

    res.cookie("jwt", token);
    tokenList[refreshToken].token = token;
    res.status(200).json({ token });
  } else {
    res.status(401).json({ message: "Unauthorized" });
  }
});

router.post("/logout", (req, res) => {
  if (req.cookies) {
    const refreshToken = req.cookies["refreshJwt"];
    if (refreshToken in tokenList) delete tokenList[refreshToken]
    res.clearCookie("refreshJwt");
    res.clearCookie("jwt");
  }

  res.status(200).json({ message: "logged out" });
});

module.exports = router;
```

Κώδικας 3: Αρχείο main.js

Το αρχείο secure.js που παρουσιάζεται στον Κώδικα 4 περιλαμβάνει τα routes του API που σχετίζονται με την ανάγνωση και ενημέρωση των δεδομένων του χρήστη. Αρχικά ορίζεται το module express για να χρησιμοποιηθεί και πάλι το express.Router. Έπειτα, εισάγεται το αρχείο asyncMiddleware το οποίο περιέχει μόνο μία συνάρτηση με το όνομα asyncMiddleware η οποία δέχεται ως όρισμα μια συνάρτηση για να την εκτελέσει με τη χρήση Promise (Κώδικας 4). Επίσης, εισάγεται και το userModel το οποίο αποτελεί δίαυλο επικοινωνίας με τη βάση δεδομένων. Το πρώτο route που ακούει στην κλήση του αιτήματος "/current" επιστρέφει τον χρήστη τον οποίο βρίσκει στη βάση χρησιμοποιώντας τη μέθοδο findOne στο UserModel. Το πρώτο όρισμα στην findOne αποτελεί τα κριτήρια με βάση τα οποία γίνεται η αναζήτηση και το δεύτερο, που δεν είναι υποχρεωτικό, είναι μία συμβολοσειρά που ορίζονται τα πεδία που θα επιστραφούν. Με αυτή τη μέθοδο επιστρέφεται ένα μοναδικό αποτέλεσμα. Με το επόμενο route, το "submit-score" ενημερώνεται η βαθμολογία και ο χρόνος κάθε επιπέδου. Αρχικά βρίσκει τον χρήστη στη βάση με τη findOne. Στη συνέχεια, αναζητεί αν υπάρχει αποθηκευμένο το επίπεδο για το οποίο πραγματοποιήθηκε το αίτημα. Σε περίπτωση που βρεθεί ανανεώνει τη βαθμολογία και το χρόνο του συγκεκριμένου επιπέδου. Στην αντίθετη περίπτωση προσθέτει το νέο επίπεδο και το εισάγει στη λίστα επιπέδων του χρήστη. Έπειτα, σε περίπτωση που το επίπεδο έχει ολοκληρωθεί με επιτυχία ενημερώνει και το πεδίο completedLevel. Τέλος, ενημερώνει τη βάση δεδομένων με τη μέθοδο updateOne. Επιπλέον, ορίζεται το route "update-level", το οποίο ενημερώνει την τιμή του πεδίου currentLevel στη βάση και το route "user-levels", το οποίο βρίσκει και επιστρέφει το πεδίο levels που περιέχει τα επίπεδα που έχει παίξει ο χρήστης.

asyncMiddleware.js

```
1  const asyncMiddleware = fn =>
2    (req, res, next) => {
3      Promise.resolve(fn(req, res, next))
4        .catch(next);
5    };
6
7  module.exports = asyncMiddleware;
```

Κώδικας 4: Αρχείο asyncMiddleware.js

secure.js

```
const express = require("express");
const asyncMiddleware = require("../middleware/asyncMiddleware");
const UserModel = require("../models/userModel");
const router = express.Router();

router.get("/current", asyncMiddleware(async (req, res, next) => {
  const user = await UserModel.findOne({ email: req.user.email }, "-password");
  res.status(200).json({ status: "ok", user: user });
}));

router.post("/submit-score", asyncMiddleware(async (req, res, next) => {
  const { level, score, time, completed } = req.body;
  const email = req.user.email;
  const user = await UserModel.findOne({ email: email });
  let userLevels = user.levels;
  let indexLevel;
  if(userLevels.some((userLevel, index) => {
      indexLevel = index;
      return userLevel.level.split("_")[0] === level;
    })) {
    let currentLevel = userLevels[indexLevel];
    currentLevel["time"] = time;
    currentLevel["score"] = score;
  } else {
    userLevels.push({
      "level": level + "_" + user._id.toString().substring(user._id.toString().length - 6),
      "score": score,
      "time": time
    });
  }
  let data = { levels: userLevels };
  if(completed !== "lose")
    data["completedLevel"] = level;
  await UserModel.updateOne({ email }, data);
  res.status(200).json({ status: "ok" });
}));

router.post("/update-level", asyncMiddleware(async (req, res, next) => {
  const { level } = req.body;
  const email = req.user.email;
  await UserModel.updateOne({ email }, { currentLevel: level });
  res.status(200).json({ status: "ok" });
}));

router.get("/user-levels", asyncMiddleware(async (req, res, next) => {
  const levels = await UserModel.findOne({ email: req.user.email }, "levels -_id");
  res.status(200).json(levels["levels"]);
}));

module.exports = router;
```

Κώδικας 5: Αρχείο secure.js

Τα τελευταία routes της εφαρμογής αφορούν την ανάκτηση του συνθηματικού του χρήστη και βρίσκονται στο αρχείο password.js. Στην αρχή ορίζονται τα απαραίτητα modules και αρχεία που θα χρησιμοποιηθούν. Έπειτα δημιουργείται ένα SMTP transport

αντικείμενο για την αποστολή email καλώντας την createTransport της nodemailer. Στη μέθοδο αυτή δηλώνεται ότι θα χρησιμοποιηθεί η υπηρεσία “gmail” για την αποστολή και η αυθεντικοποίηση γίνεται μέσω του πρωτοκόλλου OAuth 2.0. Στη συνέχεια, ακολουθεί ο ορισμός των προτύπων του περιεχομένου των emails που θα αποσταλούν (Κώδικας 6).

Το πρώτο route είναι το “forgot-password” το οποίο καλείται όταν ο χρήστης δεν γνωρίζει το password του και του αποστέλλεται ένα email που περιέχει το url μαζί με το token επαναφοράς. Αρχικά, γίνεται αναζήτηση στη βάση δεδομένων με τη διεύθυνση ηλεκτρονικού ταχυδρομείου που έδωσε ο χρήστης. Σε περίπτωση που βρεθεί στη βάση δεδομένων ο χρήστης δημιουργείται ένα token με τη μέθοδο randomBytes της crypto και ενημερώνονται στη βάση τα πεδία resetToken και resetTokenExp. Τέλος, δημιουργείται και στέλνεται το email και επιστρέφεται το κατάλληλο μήνυμα στο σώμα απάντησης του αιτήματος (Κώδικας 7). Το route “reset-password” λειτουργεί με παρόμοια λογική. Αφού βρεθεί ο χρήστης ελέγχοντας ότι το token που έχει σταλεί είναι ίδιο με το πεδίο resetToken και η ημερομηνία του αιτήματος δεν είναι μεγαλύτερη από αυτή που είναι αποθηκευμένη στο πεδίο resetTokenExp, επιβεβαιώνει ότι οι κωδικοί πρόσβασης είναι όμοιοι και ενημερώνει τον χρήστη στη βάση. Τέλος, ενημερώνει τον χρήστη με την αποστολή email (Κώδικας 8).

password.js

```
const express = require("express");
const hbs = require("nodemailer-express-handlebars");
const nodemailer = require("nodemailer");
const path = require("path");
const crypto = require("crypto");
const { google } = require("googleapis");
const OAuth2 = google.auth.OAuth2;

const asyncMiddleware = require("../middleware/asyncMiddleware");
const UserModel = require("../models/userModel");

const username = process.env.EMAIL;
const clientId = process.env.CLIENT_ID;
const clientSecret = process.env.CLIENT_SECRET;
const refreshToken = process.env.REFRESH_TOKEN;
const oauth2Client = new OAuth2(clientId, clientSecret, "https://developers.google.com/oauthplayground");
oauth2Client.setCredentials({
  refresh_token: refreshToken
});
const accessToken = oauth2Client.getAccessToken();
const smtpTransport = nodemailer.createTransport({
  service: "gmail",
  auth: {
       type: "OAuth2",
       user: username,
       clientId: clientId,
       clientSecret: clientSecret,
       refreshToken: refreshToken,
       accessToken: accessToken
  },
  tls: {
    rejectUnauthorized: false
  }
});
const handlebarsOptions = {
  viewEngine: {
    extName: ".html",
      partialsDir: path.resolve("./templates"),
      defaultLayout: false,
  },
  viewPath: path.resolve("./templates"),
  extName: ".html",
};

smtpTransport.use("compile", hbs(handlebarsOptions));
```

Κώδικας 6: Αρχείο password.js, γραμμές 1 - 45

password.js

```
47  const router = express.Router();
48  router.post("/forgot-password", asyncMiddleware(async (req, res, next) => {
49    const { email } = req.body;
50    const user = await UserModel.findOne({ email });
51    if (!user) {
52      res.status(400).json({ "message": "invalid email" });
53      return;
54    }
55
56    // create user token
57    const buffer = crypto.randomBytes(20);
58    const token = buffer.toString("hex");
59
60    // update user reset password token and exp
61    await UserModel.findByIdAndUpdate({ _id: user._id },
62      { resetToken: token, resetTokenExp: Date.now() + 600000 });
63
64    const data = {
65      to: user.email,
66      from: username,
67      template: "forgot-password",
68      subject: "Super Dwarf: Επαναφορά κωδικού",
69      context: {
70        url: process.env.NODE_ENV === "production" ?
71          `https://superdwarf.herokuapp.com/reset-password.html?token=${token}` :
72          `http://localhost:${process.env.PORT || 3000}/reset-password.html?token=${token}`,
73        name: user.name
74      }
75    };
76    await smtpTransport.sendMail(data);
77
78    res.status(200).json({
79      message: "Το email έχει σταλεί. Ο σύνδεσμος επαναφοράς κωδικού πρόσβασης ισχύει μόνο για 10 λεπτά."
80    });
81  }));
```

Κώδικας 7: Αρχείο password.js, γραμμές 47 – 81

password.js

```
83  router.post("/reset-password", asyncMiddleware(async (req, res, next) => {
84    const user = await UserModel.findOne({
85      resetToken: req.body.token, resetTokenExp: { $gt: Date.now() } });
86    if (!user) {
87      res.status(400).json({ "message": "Μη έγκυρο token" });
88      return;
89    }
90
91    if (req.body.password !== req.body.verifiedPassword) {
92      res.status(400).json({ "message": "Οι κωδικοί πρόσβασης δεν ταιριάζουν" });
93      return;
94    }
95
96    // update user model
97    user.password = req.body.password;
98    user.resetToken = undefined;
99    user.resetTokenExp = undefined;
100   await user.save();
101
102   // send password update email
103   const data = {
104     to: user.email,
105     from: username,
106     template: "reset-password",
107     subject: "Super Dwarf: Επιβεβαίωση επαναφοράς κωδικού πρόσβασης",
108     context: {
109       name: user.name
110     }
111   };
112   await smtpTransport.sendMail(data);
113
114   res.status(200).json({ message: "Ο κωδικός πρόσβασης ενημερώθηκε" });
115 }));
116
117 module.exports = router;
```

Κώδικας 8: Αρχείο password.js, γραμμές 83 – 117

5.2 Ανάλυση της βάσης δεδομένων

Για τη δημιουργία της βάσης δεδομένων MongoDB χρησιμοποιήθηκε η cloud υπηρεσία Atlas της MongoDB, που περιγράφηκε στην ενότητα 3.3.5. Αρχικά, πραγματοποιήθηκε εγγραφή στην υπηρεσία και στη συνέχεια δημιουργήθηκε το cluster με όνομα “Cluster0” το οποίο θα φιλοξενήσει τα δεδομένα της εφαρμογής. Έπειτα παράχθηκε το URI με βάση το οποίο γίνεται η σύνδεση της βάσης δεδομένων με την εφαρμογή (Κωδικας 1, γραμμές 12 - 13).

Για την αποθήκευση και ανάκτηση των δεδομένων δημιουργήθηκε με τη βοήθεια του module Mongoose το μοντέλο user και καθορίστηκε το schema που θα έχουν τα δεδομένα. Πιο συγκεκριμένα το schema περιέχει τα πεδία:

- email: αποτελεί τη διεύθυνση ηλεκτρονικού ταχυδρομείου με την οποία κάνει εγγραφή ο χρήστης, είναι υποχρεωτικό και μοναδικό κλειδί
- password: είναι το κατακερματισμένο συνθηματικό του χρήστη, είναι υποχρεωτικό
- username: είναι υποχρεωτικό και αποτελεί το ψευδώνυμο που συμπληρώνει ο χρήστης κατα την εγγραφή
- levels: είναι ένας πίνακας που περιέχει τα επίπεδα που έχει παίξει ο χρήστης. Κάθε επίπεδο είναι ένα αντικείμενο το οποίο περιέχει το υποχρεωτικό πεδίο level που αντιπροσωπεύει το κάθε επίπεδο, το score με την υψηλότερη βαθμολογία του χρήστη και το time με τον χαμηλότερο χρόνο του χρήστη στο επίπεδο.
- currentLevel: αποθηκεύει το τελευταίο επίπεδο το οποίο είχε παίξει ο χρήστης
- completedLevel: αποθηκεύει το τελευταίο επίπεδο που ολοκλήρωσε ο χρήστης
- resetToken: αποθηκεύει το token που δημιουργείται όταν ζητείται επαναφορά κωδικού
- resetTokenExp: αποθηκεύει την ημερομηνία λήξης του token επαναφοράς κωδικού

Στον Κώδικα 9 αρχικά δηλώνονται τα modules mongoose και bcrypt που θα χρησιμοποιηθούν και έπειτα ορίζονται τα LevelSchema και UserSchema με τα οποία καθορίζονται τα πεδία που θα έχει το μοντέλο user, καθώς και τον τύπο κάθε πεδίου και αν θα είναι υποχρεωτικό. Όπως έχει αναφερθεί το συνθηματικό του χρήστη δεν θα αποθηκεύεται σαν απλό κείμενο αλλά θα κατακερματίζεται πριν αποθηκευτεί. Για να επιτευχθεί αυτό προστέθηκε ένα hook στο UserSchema που ενεργοποιείται πριν από την αποθήκευση του χρήστη στη βάση και με τη χρήση της μεθόδου hash που παρέχει η bcrypt αλλάζει κατακερματίζει τον κωδικό. Μέσα σε αυτό το hook επίσης προστίθεται και το πρώτο level στο πεδίο levels. Επίσης, δημιουργήθηκε η μέθοδος isValidPassword με την οποία γίνεται η επικύρωση του συνθηματικου όταν ο χρήστης προσπαθεί να συνδεθεί.

Τέλος, δημιουργείται το μοντέλο με τη μέθοδο mongoose.model δίνοντας ως ορίσματα το όνομα “user” και το βασικό σχήμα που ορίστηκε.

userModel.js

```
const mongoose = require("mongoose");
const bcrypt = require("bcrypt");
const Schema = mongoose.Schema;
const LevelSchema = new Schema({
  level: {
    type: String,
    required: true
  },
  score: {
    type: Number
  },
  time: {
    type: Number
  }
});
const UserSchema = new Schema({
  email: {
    type: String,
    required: true,
    unique: true
  },
  password: {
    type: String,
    required: true
  },
  name: {
    type: String,
    required: true
  },
  levels: [ LevelSchema ],
  currentLevel: {
    type: String,
  },
  completedLevel: {
    type: String,
  },
  resetToken: {
    type: String
  },
  resetTokenExp: {
    type: Date
  }
});

UserSchema.pre("save", async function (next) {
  const user = this;
  const hash = await bcrypt.hash(this.password, 10);
  this.password = hash;
  this.levels.push({
    "level": "level1_" + user._id.toString().substring(user._id.toString().length - 6),
    "score": 0,
    "time": 0
  });
  next();
});
UserSchema.methods.isValidPassword = async function (password) {
  const user = this;
  const compare = await bcrypt.compare(password, user.password);
  return compare;
}

const UserModel = mongoose.model('user', UserSchema);
module.exports = UserModel;
```

Κώδικας 9: Αρχείο userModel.js

5.3 Υποδομή του client

5.3.1 Διεπαφές χρήστη

Όλα τα αρχεία που αφορούν τον client, δηλαδή τη διεπαφή που θα βλέπει ο τελικός χρήστης βρίσκονται στο φάκελο public. Αρχικά, ο χρήστης κατευθύνεται στη σελίδα σύνδεσης (Κώδικας 10) όπου περιέχεται μία φόρμα με τα πεδία “Email”(email), “Κωδικός πρόσβασης” (password), το κρυφό πεδίο “userKey” το οποίο χρησιμοποιείται για τον αποκλεισμό των bots που θα μπορούσαν να επιτεθούν στη φόρμα και το κουμπί “Σύνδεση” που όταν πατηθεί εκτελεί τη συνάρτηση signIn. Μέσα σε αυτή τη συνάρτηση αρχικά ελέγχεται εάν έχει συμπληρωθεί το πεδίο userKey και σε περίπτωση που έχει συμπληρωθεί διακόπτεται η εκτέλεση της. Στην αντίθετη περίπτωση εκτελεί ένα POST αίτημα σύνδεσης (login) στέλνοντας τα δεδομένα email και password που ανακτήθηκαν από τα πεδία της φόρμας. Εάν το αίτημα είναι επιτυχές, ο χρήστης ανακατευθύνεται στη σελίδα του παιχνιδιού, ενώ εάν είναι ανεπιτυχές εμφανίζονται τα κατάλληλα μηνύματα. Στην σελίδα αυτή υπάρχουν επίσης ο σύνδεσμος “Δημιουργία νέου λογαριασμού” που οδηγεί στη σελίδα εγγραφής (Κώδικας 11) και ο σύνδεσμος “Ξεχάσατε τον κωδικό πρόσβασης” που οδηγεί στην αντίστοιχη σελίδα (Κώδικας 12).

index.html

```
<form class="form-signin">
    <h3 class="h3 mb-3 font-weight-normal">Παρακαλώ συνδεθείτε</h3>
    <label for="email" class="sr-only">Email</label>
    <input type="email" id="email" class="form-control" placeholder="Email" required autofocus>
    <label for="password" class="sr-only">Κωδικός πρόσβασης</label>
    <input type="password" id="password" class="form-control" placeholder="Κωδικός πρόσβασης" required>
    <input id="userKey" type="text" name="userKey" tabindex="-1" value="" autocomplete="off" />
    <div id="error"></div>
    <a id="loginBtn" class="btn-home" style="width: 100%; margin-bottom: 1rem;" onClick="signIn()">Σύνδεση</a>
    <a href="/signup.html">Δημιουργία νέου λογαριασμού.</a>
    <a href="/forgot-password.html">Ξεχάσατε τον κωδικό πρόσβασης;</a>
</form>
<script>
    function signIn() {
        const userKey = document.getElementById("userKey").value;
        if(userKey.length)
            return false;
        const data = {
            email: document.getElementById("email").value,
            password: document.getElementById("password").value
        };
        $.ajax({
            type: "POST",
            url: "/login",
            data,
            success: data => {
                window.location.replace("/game.html");
            },
            error: xhr => {
                if(xhr.status !== 401) {
                    window.alert(JSON.stringify(xhr));
                    return;
                }
                let errorElement = document.getElementById("error");
                let message = xhr.statusText;
                if(xhr.responseJSON != null && xhr.responseJSON.message != null)
                    message = xhr.responseJSON.message;
                errorElement.innerHTML = "<p style='color: red;'>"+ message + "</p>";
            }
        });
    }
    const onKeypress = function(event) {
        if (event.keyCode === 13) {
            event.preventDefault();
            document.getElementById("loginBtn").click();
        }
    }
    document.getElementById("email").addEventListener("keyup", onKeypress);
    document.getElementById("password").addEventListener("keyup", onKeypress);
</script>
```

Κώδικας 10: Αρχείο index.html

Η σελίδα εγγραφής είναι υπεύθυνη για τη δημιουργία λογαριασμού χρήστη στο παιχνίδι. Περιέχει μία φόρμα με τα πεδία “Email” (email), “Κωδικός πρόσβασης” (password), “Όνομα” (username) και το κρυφό πεδίο “userKey” και το κουμπί “Εγγραφή” που εκτελεί τη συνάρτηση signUp. Κατά την εκτέλεση της αρχικά ελέγχεται εάν έχει συμπληρωθεί το πεδίο userKey και εάν ο έλεγχος αποτύχει εκτελείται το POST αίτημα εγγραφής (signup) στέλνοντας τα δεδομένα της φόρμας. Εάν η εγγραφή είναι επιτυχής και δημιουργηθεί ο χρήστης στη βάση τότε ανακατευθύνεται στη σελίδα σύνδεσης. Εάν

προκύψει κάποιο σφάλμα τότε ο χρήστης ενημερώνεται και παραμένει στη σελίδα εγγραφής. Σε περίπτωση που έχει ήδη λογαριασμό μπορεί να χρησιμοποιήσει το σύνδεσμο "Έχετε ήδη λογαριασμό? Συνδεθείτε εδώ." που οδηγεί στη σελίδα σύνδεσης.

signup.html

```
<form class="form-signin">
    <h1 class="h3 mb-3 font-weight-normal">Εγγραφή</h1>
    <label for="email" class="sr-only">Email</label>
    <input type="email" id="email" class="form-control" placeholder="Email" required autofocus>
    <label for="password" class="sr-only">Κωδικός πρόσβασης</label>
    <input type="password" id="password" class="form-control" placeholder="Κωδικός πρόσβασης" required>
    <label for="name" class="sr-only">Όνομα</label>
    <input type="text" id="name" class="form-control" placeholder="Όνομα" required>
    <input id="userKey" type="text" name="userKey" tabindex="-1" value="" autocomplete="off" />
    <a id="signInBtn" class="btn-home" style="width: 100%; margin-bottom: 1rem;" onClick="signUp()">Εγγραφή</a>
    <a href="/index.html">Έχετε ήδη λογαριασμό? Συνδεθείτε εδώ.</a>
</form>
<script>
    function signUp() {
        const userKey = document.getElementById("userKey").value;
        if(userKey.length)
            return false;
        const data = {
            email: document.getElementById("email").value,
            password: document.getElementById("password").value,
            name: document.getElementById("name").value
        };
        $.ajax({
            type: "POST",
            url: "/signup",
            data,
            success: data => {
                window.alert("user created successfully");
                window.location.replace("/index.html");
            },
            error: xhr => {
                window.alert(JSON.stringify(xhr));
                window.location.replace("/signup.html");
            }
        });
    }
    const onKeypress = event => {
        if (event.keyCode === 13) {
            event.preventDefault();
            document.getElementById("signInBtn").click();
        }
    };
    document.getElementById("email").addEventListener("keyup", onKeypress);
    document.getElementById("password").addEventListener("keyup", onKeypress);
    document.getElementById("name").addEventListener("keyup", onKeypress);
</script>
```

Κώδικας 11: Αρχείο signup.html

Ο χρήστης έχει τη δυνατότητα να αλλάξει κωδικό σε περίπτωση που τον ξεχάσει πηγαίνοντας στην ανάλογη σελίδα "Βρείτε το λογαριασμό σας". Σε αυτή τη σελίδα αφού συμπληρώσει τη διεύθυνση ηλεκτρονικού ταχυδρομείου και πατήσει το κουμπί "Αποστολή" θα εκτελεστεί ένα POST αίτημα στο endpoint "forgot-password" στέλνοντας σαν δεδομένο στο διακομιστή το email (Κώδικας 12). Στη συνέχεια, θα λάβει ένα μήνυμα

στο email του με το σύνδεσμο επαναφοράς κωδικού που θα τον κατευθύνει στη κατάλληλη σελίδα (Κώδικας 13). Εκεί θα πρέπει να συμπληρώσει τα πεδία “Κωδικός πρόσβασης” (password) και “Επιβεβαίωση κωδικού πρόσβασης” (verifiedPassword). Με το πάτημα του κουμπιού “Αποστολή” θα κληθεί η συνάρτηση resetPassword η οποία θα εκτελέσει ένα POST αίτημα στο endpoint “reset-password” στέλνοντας ως δεδομένα τους κωδικούς και το token που υπήρχε στο σύνδεσμο επαναφοράς.

forgot-password.html

```
<form class="form-signin">
  <h1 class="h3 mb-3 font-weight-normal">Βρείτε το λογαριασμό σας</h1>
  <label for="email" class="sr-only">Email</label>
  <input type="email" id="email" class="form-control" placeholder="Email" required autofocus>
  <input id="userKey" type="text" name="userKey" tabindex="-1" value="" autocomplete="off" />
  <a class="btn-home" style="width: 100%; margin-bottom: 1rem;" onClick="forgotPassword()">Αποστολή</a>
  <a href="/index.html">Δεν χρειάζεται να επαναφέρετε τον κωδικό πρόσβασης; Συνδεθείτε εδώ.</a>
</form>
<script>
  function forgotPassword() {
    const userKey = document.getElementById("userKey").value;
    if(userKey.length)
        return false;
    const data = { email: document.getElementById("email").value };
    $.ajax({
      type: "POST",
      url: "/forgot-password",
      data,
      success: data => {
        window.alert(data.message);
        window.location.replace("/index.html");
      },
      error: xhr => {
        window.alert(JSON.stringify(xhr));
        window.location.replace("/forgot-password.html");
      }
    });
  }
</script>
```

Κώδικας 12: Αρχείο forgot-password.html

reset-password.html

```
<form class="form-signin">
  <h1 class="h3 mb-3 font-weight-normal">Επαναφέρετε τον κωδικό πρόσβασης</h1>
  <label for="password" class="sr-only">Κωδικός πρόσβασης</label>
  <input type="password" id="password" class="form-control" placeholder="Κωδικός πρόσβασης" required autofocus>
  <label for="verifiedPassword" class="sr-only">Επιβεβαίωση κωδικού πρόσβασης</label>
  <input type="password" id="verifiedPassword" class="form-control" placeholder="Επιβεβαίωση κωδικού πρόσβασης" required>
  <input id="userKey" type="text" name="userKey" tabindex="-1" value="" autocomplete="off" />
  <a class="btn-home" style="width: 100%; margin-bottom: 1rem;" onClick="resetPassword()">Αποστολή</a>
</form>
<script>
  function resetPassword() {
    const userKey = document.getElementById("userKey").value;
    if(userKey.length)
        return false;
    const token = document.location.href.split("token=")[1];
    const password = document.getElementById("password").value;
    const verifiedPassword =  document.getElementById("verifiedPassword").value;

    if (password !== verifiedPassword) {
      window.alert("passwords do not match");
    } else {
      const data = {
        password: password,
        verifiedPassword: verifiedPassword,
        token: token
      };
      $.ajax({
        type: "POST",
        url: "/reset-password",
        data,
        success: data => {
          window.alert(data.message);
          window.location.replace("/index.html");
        },
        error: xhr => {
          window.alert(JSON.stringify(xhr));
          window.location.replace("/reset-password.html");
        }
      });
    }
  }
</script>
```

Κώδικας 13: Αρχείο reset-password.html

Όταν ο χρήστης συνδεθεί με επιτυχία ανακατευθύνεται στη σελίδα του παιχνιδιού (Κώδικας 14). Σε αυτή τη σελίδα εισάγονται όλες οι απαραίτητες βιβλιοθήκες που θα χρησιμοποιηθούν στο παιχνίδι (Κώδικας 14, γραμμές 17-23), το αρχείο refreshToken.js που είναι υπεύθυνο για τη λήψη του token ανανέωσης από το cookie και την εκτέλεση POST αιτήματος για τη λήψη νέου (Κώδικας 15) και το αρχείο game.js που είναι υπεύθυνο για τη σχεδίαση της περιοχής παιχνιδιού.

game.html

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Super Dwarf</title>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
      integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
  <link rel="preconnect" href="https://fonts.gstatic.com">
  <link href="assets/css/main.css" rel="stylesheet">
  <link href="https://fonts.googleapis.com/css2?family=Source+Code+Pro&display=swap" rel="stylesheet">
</head>
<body id="game-page">
  <div class="container">
      <div id="game-container"></div>
  </div>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/ace/1.4.14/ace.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/js-beautify/1.14.3/beautify.min.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/js-beautify/1.14.3/beautify-css.min.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/js-beautify/1.14.3/beautify-html.min.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/jshint/2.13.4/jshint.min.js"></script>
  <script src="https://unpkg.com/esprima@4.0.1/dist/esprima.js"></script>
  <script src="//cdn.jsdelivr.net/npm/phaser@3.52.0/dist/phaser.min.js"></script>
  <script src="js/refreshToken.js"></script>
  <script src="js/game.js" type="module"></script>
  <script nomodule>
    alert("This browser doesn't support JS modules, so the example won't work as is.");
  </script>
</body>
</html>
```

Κώδικας 14: Αρχείο game.html

refreshToken.js

```
function getCookie(cname) {
  const name = cname + "=";
  const decodedCookie = decodeURIComponent(document.cookie);
  const ca = decodedCookie.split(";");
  for(let i = 0; i <ca.length; i++) {
    let c = ca[i];
    while(c.charAt(0) == ' ')
      c = c.substring(1);
    if(c.indexOf(name) == 0)
      return c.substring(name.length, c.length);
  }
  return "";
}

setInterval(function() {
  $.ajax({
    type: "POST",
    url: "/token",
    data: { refreshToken: getCookie("refreshJwt") },
    success: data => { },
    error: xhr => {
      window.alert(JSON.stringify(xhr));
      window.location.replace("/index.html");
    }
  });
}, 10000);
```

Κώδικας 15: Αρχείο refreshToken.js

Το αρχείο game.js παρουσιάζεται στον Κώδικα 16 και ξεκινάει με την εισαγωγή των αρχείων JavaScript που περιέχουν των κώδικα των σκηνών του παιχνιδιού. Έπειτα ορίζεται ένα αντικείμενο που περιέχει τις ρυθμίσεις του παιχνιδιού που θα περαστούν ως όρισμα στον κατασκευαστή του παιχνιδιού new Phaser.Game(config). Βασικές ρυθμίσεις αποτελούν η parent (γραμμή 14), η οποία καθορίζει σε ποιο element της σελίδας game.html θα φορτώσει το παιχνίδι, η dom (γραμμές 17-19) με την οποία δηλώνεται ότι είναι επιθυμητή η χρήση html elements στο παιχνίδι, η scene (γραμμές 20-29) που περιέχει τις διάφορες σκηνές του παιχνιδιού και η physics (γραμμές 30-35) η οποία αφορά την ενσωμάτωση μηχανής που διαχειρίζεται τους κανόνες φυσικής. Τέλος, μετά τη δημιουργία του παιχνιδιού ακολουθεί μια προσθήκη βελτίωσης για τον συντάκτη κώδικα που υπάρχει στο παιχνίδι.

game.js

```
import StartScene from "./scenes/start-scene.js";
import AboutScene from "./scenes/about-scene.js";
import LevelsScene from "./scenes/levels-scene.js";
import GameOverScene from "./scenes/gameover-scene.js";
import HUD from "./scenes/HUD.js";
import PlatformerScene from "./scenes/platformer-scene.js";
import DialogScene from "./scenes/dialog-scene.js";
import KnowledgeScene from "./scenes/knowledge-scene.js";

const config = {
    type: Phaser.AUTO,
    width: 832,
    height: 640,
    parent: "game-container",
    pixelArt: true,
    backgroundColor: "#1d212d",
    dom: {
        createContainer: true
    },
    scene: [
        StartScene,
        AboutScene,
        LevelsScene,
        PlatformerScene,
        HUD,
        DialogScene,
        KnowledgeScene,
        GameOverScene,
    ],
    physics: {
        default: "arcade",
        arcade: {
            gravity: { y: 500 }
        }
    }
};

const game = new Phaser.Game(config);

// Fix for ace tooltip
ace.require("ace/tooltip").Tooltip.prototype.setPosition = (x, y) => {
    y -= $(this.$parentNode).offset().top;
    x -= $(this.$parentNode).offset().left;
    this.getElement().style.left = x + "px";
    this.getElement().style.top = y + "px";
};

```

Κώδικας 16: Αρχείο game.js

Όπως γίνεται αντιληπτό από τη ρύθμιση scene στον Κώδικα 16 το παιχνίδι αποτελείται από συνολικά οκτώ σκηνές. Η σκηνή StartScene περιέχει το βασικό μενού του παιχνιδιού και φορτώνει τα απαραίτητα γραφικά και ήχους. Αυτή είναι η σκηνή που εκκινεί το παιχνίδι καθώς έχει δηλωθεί πρώτη στη σειρά. Η σκηνή AboutScene παρουσιάζει την ιστορία και τις οδηγίες και η LevelsScene όλα τα επίπεδα του παιχνιδιού. Η PlatformerScene αποτελεί μια από τις πιο σημαντικές σκηνές η οποία αφορά την κατασκευή κάθε επιπέδου. Η HUD είναι μία βοηθητική σκηνή που περιέχει γενικά στοιχεία του παιχνιδιού. Η σκηνή DialogScene περιέχει όλη τη λογική των δραστηριοτήτων που περιέχει το παιχνίδι. Με τη σκηνή KnowledgeScene παρουσιάζεται η θεωρία κατά τη διάρκεια μιας δραστηριότητας. Τέλος, η GameOverScene είναι η σκηνή που αφορά το σχεδιασμό της οθόνης ολοκλήρωσης επιπέδου.

5.3.2 Σκηνές του παιχνιδιού

5.3.2.1 Σκηνή StartScene

Η σκηνή αυτή είναι υπεύθυνη για το σχεδιασμό του βασικού μενού που περιέχει τα κουμπιά “Έναρξη”, “Οδηγίες”, “Επίπεδα” και “Αποσύνδεση”. Επίσης, όταν ξεκινάει αυτή η σκηνή φορτώνονται στο παιχνίδι τα γραφικά, όπως οι εικόνες και τα sprite sheets και τα αρχεία ήχου, όπως τα ηχητικά εφέ και η μουσική υποβάθρου που θα χρησιμοποιηθούν στο παιχνίδι. Στον Πίνακα 9 παρουσιάζονται οι συναρτήσεις της σκηνής.

Πίνακας 9: Συναρτήσεις αρχείου start-scene.js

Όνομα	**Περιγραφή**
preload	Γραμμές Κώδικα: 11-55 Είναι υπεύθυνη για τη φόρτωση όλων των αρχείων που θα χρησιμοποιηθούν στο παιχνίδι. Τα αρχεία αυτά αποτελούν εικόνες, sprites, εφέ ήχου και μουσική.
create	Γραμμές Κώδικα: 58-84 Δημιουργεί την οθόνη του βασικού μενού ορίζοντας την εικόνα φόντου, τον τίτλο και σχεδιάζοντας τα κουμπιά.
startGame	Γραμμές Κώδικα: 86 - 105 Καλείται κατά το πάτημα του κουμπιού “Έναρξη”. Αρχικά σταματάει τη μουσική της αρχικής οθόνης. Έπειτα στέλνει ένα αίτημα GET στον server για να βρεθεί το επίπεδο που βρίσκεται ο χρήστης και αρχικοποιεί τις μεταβλητές που είναι διαθέσιμες σε όλες τις σκηνές με την

	initRegistry. Στη συνέχεια, εκκινεί την σκηνή HUD εκτελώντας τη μέθοδο launch που επιτρέπει στη σκηνή να τρέχει παράλληλα και τέλος με τη μέθοδο start εκκινεί την PlatformerScene. Η μέθοδος start τερματίζει τη σκηνή που τρέχει και εκκινεί αυτή που ορίζεται.
showLevels	Γραμμές Κώδικα: 107 - 110 Σταματάει την τρέχουσα σκηνή και ξεκινάει την "LevelsScene" όταν πατηθεί το κουμπί "Επίπεδα".
aboutScene	Γραμμές Κώδικα: 112 - 115 Σταματάει την τρέχουσα σκηνή και ξεκινάει την "AboutScene" όταν πατηθεί το κουμπί "Οδηγίες".
exitGame	Γραμμές Κώδικα: 117 - 130 Καλείται όταν ο χρήστης πατήσει την "Αποσύνδεση". Στέλνει το POST αίτημα logout στον διακομιστή και ανακατευθύνει τον χρήστη στη σελίδα σύνδεσης.
initRegistry	Γραμμές Κώδικα: 132 - 141 Είναι υπεύθυνη για τον αρχικό ορισμό των δεδομένων που πρέπει να είναι προσβάσιμα και διαχειρίσιμα από όλες τις σκηνές του παιχνιδιού. Τα δεδομένα αυτά είναι η μέγιστη ζωή του ήρωα (health_max), η τρέχουσα ζωή του ήρωα (health_current), ο μέγιστος αριθμός μαγικών επιθέσεων του ήρωα (fireball_max), ο τρέχων αριθμός των μαγικών επιθέσεων (fireball_current), η συνολική βαθμολογία (score), τα αναγνωριστικά της θεωρίας που έχει διαβάσει ο χρήστης στο επίπεδο (quotes_ids) και τέλος ο κώδικας που συμπληρώνει ο χρήστης στο επίπεδο (user_code) (Κώδικας 17).

start-scene.js

```
132     initRegistry() {
133         //the game registry provides a place accessible by all scenes to set and get data.
134         this.registry.set("health_max", 4);
135         this.registry.set("health_current", 4);
136         this.registry.set("fireball_max", 10);
137         this.registry.set("fireball_current", 10);
138         this.registry.set("score", 0);
139         this.registry.set("quotes_ids", [ ]);
140         this.registry.set("user_code", { });
141     }
```

Κώδικας 17: Συνάρτηση initRegistry στο αρχείο start-scene.js

5.3.2.2 Σκηνή HUD

Η HUD είναι μία σκηνή η οποία τρέχει παράλληλα με την PlatformerScene και είναι υπεύθυνη για την παρουσίαση της ζωής του ήρωα, των μαγικών επιθέσεων, της βαθμολογίας που συγκεντρώνει ο παίκτης, του χρονόμετρου καθώς και των κουμπιών που

είναι υπεύθυνα για την έξοδο από το επίπεδο και την ενεργοποίηση ή μη της μουσικής. Στον Πίνακα 10 παρουσιάζονται οι συναρτήσεις της.

Πίνακας 10: Συναρτήσεις αρχείου HUD.js

Όνομα	**Περιγραφή**
create	Γραμμές Κώδικα: 8-45 Ζωγραφίζει στο επίπεδο τα δεδομένα που αφορούν τη ζωή, τις μαγικές επιθέσεις, τη συνολική βαθμολογία και το χρονόμετρο. Επίσης, προσθέτει συμβάντα σε αυτά τα δεδομένα τα οποία ενεργοποιούνται κάθε φορά που υπάρχει κάποια αλλαγή. Τέλος, προσθέτει το κουμπί με το σύμβολο της νότας για την απενεργοποίηση της μουσικής και το κουμπί "Εξοδος".
onEvent	Γραμμές Κώδικα: 47-51 Ενημερώνει το δεδομένο time και το κείμενο που αφορά το χρονόμετρο του επιπέδου.
updateHealth	Γραμμές Κώδικα: 53-55 Ανανεώνει το κείμενο που αναφέρει τις συνολικές ζωές του χρήστη. Εκτελείται όταν υπάρχει αλλαγή στο "health_current" ή "health_max"
updateFireball	Γραμμές Κώδικα: 57-59 Αλλάζει το κείμενο που εμφανίζει τις μαγικές επιθέσεις. Εκτελείται όταν αλλάζει το δεδομένο "fireball_current"
updateScore	Γραμμές Κώδικα: 61-63 Ενημερώνει το κείμενο της βαθμολογίας που εμφανίζεται στον παίκτη.
muteMusic	Γραμμές Κώδικα: 65-67 Αφορά την ενεργοποίηση ή όχι της μουσικής.
exitGame	Γραμμές Κώδικα: 69-91 Είναι υπεύθυνη για την έξοδο του παίκτη από το επίπεδο. Δημιουργεί το παράθυρο που ζητά την επιβεβαίωση του παίκτη για την έξοδο από το επίπεδο και εάν απαντήσει θετικά σταματάει τις σκηνές που τρέχουν και ξεκινάει στην StartScene.

5.3.2.3 Σκηνή AboutScene

Η σκηνή αυτή παρουσιάζει την ιστορία του παιχνιδιού και τις οδηγίες που πρέπει να ακολουθήσει ο παίκτης για να παίξει το κάθε επίπεδο. Στον Πίνακα 11 παρουσιάζονται οι συναρτήσεις που περιέχει.

Πίνακας 11: Συναρτήσεις αρχείου about-scene.js

Όνομα	**Περιγραφή**
preload	Γραμμές Κώδικα: 11-13 Είναι υπεύθυνη για τη φόρτωση του αρχείου about.html που περιέχει το κείμενο της ιστορίας και τις οδηγίες.
create	Γραμμές Κώδικα: 15-30 Σχεδιάζει τη σκηνή προσθέτοντας την εικόνα φόντου και την εικόνα της περγαμηνής πάνω στην οποία θα εμφανιστεί το κείμενο. Έπειτα προσθέτει στη σκηνή το αρχείο html που φορτώθηκε, σχεδιάζει τον τίτλο και τα τοποθετεί στη σκηνή. Τέλος, προστίθεται το κουμπί "Εξοδος".
exitAbout	Γραμμές Κώδικα: 32-35 Σταματάει την παρούσα σκηνή και εκκινεί τη σκηνή StartScene.

5.3.2.4 Σκηνή LevelsScene

Η σκηνή αυτή σχεδιάζει την οθόνη που παρουσιάζει όλα τα επίπεδα του παιχνιδιού. Στον Πίνακα 12 περιγράφονται οι συναρτήσεις της.

Πίνακας 12: Συναρτήσεις αρχείου about-scene.js

Όνομα	**Περιγραφή**
init	Γραμμές Κώδικα: 11-13 Αρχικοποιεί τη μεταβλητή που αφορά τη μουσική.
preload	Γραμμές Κώδικα: 15-17 Είναι υπεύθυνη για τη φόρτωση του αρχείου levels.html που περιέχει τον κώδικα HTML που παρουσιάζει τα επίπεδα.
create	Γραμμές Κώδικα: 19-55 Αρχικά προσθέτει την εικόνα φόντου και το αρχείο html που φορτώθηκε. Έπειτα καθορίζονται οι ενέργειες που θα εκτελεστούν όταν επιλεχθεί ένα συγκεκριμένο επίπεδο. Στη συνέχεια καλείται η συνάρτηση getUserLevels και αφού επιστρέψει τα αποτελέσματα ενημερώνονται οι πληροφορίες του κάθε επιπέδου.
initRegistry	Γραμμές Κώδικα: 57-67 Είναι υπεύθυνη για τον αρχικό ορισμό των δεδομένων που πρέπει να είναι προσβάσιμα και διαχειρίσιμα από όλες τις σκηνές του παιχνιδιού. Είναι αντίστοιχη με τη συνάρτηση initRegistry της σκηνής StartScene.
getUserLevels	Γραμμές Κώδικα: 69-80 Η συγκεκριμένη συνάρτηση στέλνει ένα POST αίτημα στον διακομιστή για να λάβει τις πληροφορίες που αφορούν τα δεδομένα των επιπέδων.
exit	Γραμμές Κώδικα: 82-85

	Σταματάει την παρούσα σκηνή και εκκινεί την σκηνή StartScene.

5.3.2.5 Σκηνή PlatformerScene

Η PlatformerScene αποτελεί την πιο βασική σκηνή και είναι αρμόδια για τη δημιουργία του επιπέδου και την προσθήκη των αντικειμένων όπως ο ήρωας του παιχνιδιού και οι εχθροί. Τα αρχεία κώδικα που αφορούν τα αντικείμενα εισάγονται στην αρχή του αρχείου και θα περιγραφούν στην Ενότητα 5.3.3. Στον Πίνακα 13 θα περιγραφούν οι συναρτήσεις του αρχείου platformer-scene.js.

Πίνακας 13: Συναρτήσεις αρχείου platformer-scene.js

Όνομα	**Περιγραφή**
preload	Γραμμές Κώδικα: 16-21 Είναι υπεύθυνη για τη φόρτωση της εικόνας που περιέχει τα tiles και του αρχείου json που περιέχει της πληροφορίες για τον χάρτη που θα σχεδιαστεί. Επίσης, εκχωρεί στη μεταβλητή level το τρέχων επίπεδο και καλεί τη συνάρτηση updateCurrentLevel.
create	Γραμμές Κώδικα: 23-123 Δημιουργεί τον κόσμο του επιπέδου με βάση τις πληροφορίες που περιέχει το αρχείο json του επιπέδου. Αρχικά επιλέγεται η μουσική και οι καιρικές συνθήκες σύμφωνα με τον αριθμό του επιπέδου. Έπειτα σχεδιάζεται ο κόσμος του παιχνιδιού με βάση τις οδηγίες που έχουν οριστεί στο αρχείο και ανάλογα με τις καιρικές συνθήκες επηρεάζεται και η φωτεινότητά τους (Κώδικας 18). Στη συνέχεια, καλείται η συνάρτηση convertObjects, η οποία προσθέτει όλα τα αντικείμενα εκτός από αυτό του ήρωα. Ακολουθεί η δημιουργία και προσθήκη του ήρωα και των επιθέσεων του. Μετά την προσθήκη όλων των παραπάνω ορίζονται οι συγκρούσεις του ήρωα με το έδαφος, τα εμπόδια, το έδαφος που αποτελείται από καρφιά και τους εχθρούς. Τέλος, ρυθμίζεται κατάλληλα η κάμερα του παιχνιδιού (Κώδικας 19).
update	Γραμμές Κώδικα: 125-143 Εκτελείται σε κάθε εικόνα (frame) του παιχνιδιού και για κάθε αντικείμενο που υπάρχει στο επίπεδο εκτελεί την αντίστοιχη συνάρτηση update που τα αφορά.
convertTimeToBonusScore	Γραμμές Κώδικα: 145-148 Είναι η συνάρτηση που μετατρέπει το χρόνο τον οποίο ολοκλήρωσε ο παίκτης το επίπεδο σε επιπλέον βαθμούς.

addBox	Γραμμές Κώδικα: 150-155 Με αυτή τη συνάρτηση προστίθενται αντικείμενα που απεικονίζονται ως βράχια στον κόσμο του παιχνιδιού.
createSpeechBubble	Γραμμές Κώδικα: 157-213 Η συνάρτηση εκτελείται στην αρχή του επιπέδου και δημιουργεί τη φούσκα ομιλίας που παρουσιάζει το κείμενο με τον στόχο.
removeTile	Γραμμές Κώδικα: 215-217 Αφαιρεί συγκεκριμένα πλακίδια από τον κόσμο του παιχνιδιού με βάση τις συντεταγμένες που ορίζονται όταν καλείται.
updateCorrectQuestionsMap	Γραμμές Κώδικα: 219-221 Ανανεώνει το αντικείμενο που περιέχει τις πληροφορίες για την ορθότητα των απαντήσεων του παίκτη.
getNumberOfCorrectQuestions	Γραμμές Κώδικα: 223-225 Επιστρέφει τον αριθμό των σωστών απαντήσεων που έδωσε ο παίκτης σε ένα επίπεδο.
playerTouchedByEnemy	Γραμμές Κώδικα: 227-239 Αποτελεί τη συνάρτηση που εκτελείται όταν έρθει σε επαφή ο ήρωας με έναν εχθρό. Αφαιρεί από τη ζωή του ήρωα ένα πόντο και για κάποια δευτερόλεπτα ο ήρωας ξεθωριάζει και αλλάζει χρώμα για να είναι διακριτό ότι τραυματίστηκε.
playerFallDown	Γραμμές Κώδικα: 241-243 Εκτελείται όταν ο παίκτης πέσει σε έδαφος με καρφιά και μειώνει τη ζωή του ήρωα κατά 10 πόντους. Έπειτα οδηγεί στη λήξη του επιπέδου.
updateCurrentLevel	Γραμμές Κώδικα: 245-258 Η συνάρτηση αυτή στέλνει ένα POST αίτημα “update-level” με την πληροφορία του τρέχοντος επιπέδου για να ενημερωθεί στη βάση το πεδίο “currentLevel”.
gameOver	Γραμμές Κώδικα: 260-312 Καλείται όταν το επίπεδο ολοκληρωθεί είτε με επιτυχία είτε όχι. Είναι υπεύθυνη για την ανανέωση της βαθμολογίας και του χρόνου του χρήστη στο συγκεκριμένο επίπεδο. Τέλος, ξεκινά τη σκηνή GameOverScene περνώντας τα απαραίτητα δεδομένα σε αυτή.
setDialog	Γραμμές Κώδικα: 314-317 Ξεκινά τη σκηνή DialogScene περνώντας τις κατάλληλες πληροφορίες και σταματάει την εκτέλεση της update της σκηνής PlatformerScene.

convertObjects	Γραμμές Κώδικα: 319-362 Η συνάρτηση αυτή διαβάζει από το json αρχείο τις πληροφορίες που έχουν οριστεί για τις ομάδες αντικειμένων και δημιουργεί στις κατάλληλες συντεταγμένες τα αντικείμενα που αφορούν τους εχθρούς, τα σημεία θεωρίας και δραστηριοτήτων και την πόρτα εξόδου από το επίπεδο.
createPickUps	Γραμμές Κώδικα: 364-390 Δημιουργεί ξανά τα αντικείμενα που αφορούν δραστηριότητες ερώτησης πολλαπλής επιλογής και ο παίκτης έχει δώσει λανθασμένη απάντηση.

platformer-scene.js

```
24      this.stop = false;
25      const numOflevel = parseInt(this.level.substring(5));
26      this.music = this.sound.add(numOflevel < 6 ? "levelHTML" : numOflevel > 11 ? "levelJavascript" : "levelCSS" );
27      this.music.setLoop(true);
28      this.music.volume = 0.7;
29      this.music.play();
30
31      // weathers
32      const weathers = {
33        "morning": {
34          "color": 0xecdccc,
35          "bgColor": 0xF8c3aC,
36        },
37        "afternoon": {
38          "color": 0xffffff,
39          "bgColor": 0x0571FF,
40        },
41        "twilight": {
42          "color": 0xccaacc,
43          "bgColor": 0x18235C
44        },
45        "night": {
46          "color": 0x777777,
47          "bgColor": 0x000000
48        },
49        "underground": {
50          "color": 0xccaacc,
51          "bgColor": 0x000000
52        }
53      }
54
55      // change weather
56      let weather;
57      if(numOflevel > 11)
58        weather = "underground";
59      else {
60        const arr = ["night", "morning", "afternoon", "twilight"];
61        weather = arr[numOflevel%4];
62      }
63      let { color, bgColor } = weathers[weather];
64      this.cameras.main.setBackgroundColor(bgColor);
65
66      this.map = this.make.tilemap({ key: this.level });
67      const tiles = this.map.addTilesetImage("mapTiles", "mapTiles");
68      const foreground = this.map.createLayer("Foreground", tiles);
69      foreground.getTilesWithin().forEach((tile) => { tile.tint = color; } );
70      const frontForeground = this.map.createLayer("FrontForeground", tiles);
71      frontForeground.getTilesWithin().forEach((tile) => { tile.tint = color; } );
72      this.barrier = this.map.createLayer("Barrier", tiles);
73      this.barrier.getTilesWithin().forEach((tile) => { tile.tint = color; } );
74      this.groundLayer = this.map.createLayer("Ground", tiles);
75      this.groundLayer.getTilesWithin().forEach((tile) => { tile.tint = color; } );
76      this.deathGroundLayer = this.map.createLayer("DeathGround", tiles);
77      if(this.deathGroundLayer != null) {
78        this.deathGroundLayer.getTilesWithin().forEach((tile) => { tile.tint = color; } );
79        this.deathGroundLayer.setCollisionByProperty({ collides: true });
80      }
81
82      this.groundLayer.setCollisionByProperty({ collides: true });
83      this.barrier.setCollisionByProperty({ collides: true });
```

Κώδικας 18: Επιλογή καιρού, μουσικής και δημιουργία του κόσμου

platformer-scene.js

```
85        this.doors = this.add.group();
86        this.enemies = this.add.group();
87        this.pickups = this.add.group();
88        this.correctQuestionsMap = { };
89        this.recreatePickups = true;
90        this.convertObjects();
91
92        // player
93        this.playerX = 50, this.playerY = 40;
94        this.map.getObjectLayer("player").objects.forEach(obj => {
95          if(obj["type"] === "StartingPosition") {
96            this.playerX = obj.x;
97            this.playerY = obj.y;
98          }
99        });
100       this.player = new Player({
101         scene: this,
102         x: this.playerX,
103         y: this.playerY
104       });
105       // player attack
106       this.playerAttack = this.add.group({
107         classType: Fireball,
108         maxSize: 100,
109         runChildUpdate: true
110       });
111
112       this.physics.world.addCollider(this.player, this.groundLayer);
113       this.physics.world.addCollider(this.player, this.barrier);
114       this.collider = this.physics.add.collider(this.player, this.enemies, this.playerTouchedByEnemy, null, this);
115       if(this.deathGroundLayer != null)
116         this.physics.add.collider(this.player, this.deathGroundLayer, this.playerFallDown.bind(this));
117
118       this.cameras.main.setBounds(0, 0, this.map.widthInPixels, this.map.heightInPixels);
119       this.physics.world.setBounds(0, 0, this.map.widthInPixels, this.map.heightInPixels);
120       this.cameras.main.startFollow(this.player, true, 0.5, 0.5, -200, 0);
121
122       this.createSpeechBubble(parseInt(this.playerX) - 40, this.playerY - 128);
```

Κώδικας 19: Προσθήκη ήρωα, αντικειμένων στον κόσμο και συγκρούσεων μεταξύ τους

5.3.2.6 Σκηνή DialogScene

Η DialogScene είναι υπεύθυνη για τον σχεδιασμό και τη λειτουργικότητα των οθονών της θεωρίας και των δραστηριοτήτων. Όλα τα περιεχόμενα της θεωρίας και των δραστηριοτήτων υπάρχουν στο αρχείο content.json που φορτώνεται όταν δημιουργείται η σκηνή. Το αρχείο αυτό είναι μορφής JSON και αποτελείται από ένα σύνολο αντικειμένων που έχουν ως κλειδί ένα μοναδικό αλφαριθμητικό. Το αντικείμενο που αντιπροσωπεύει τη θεωρία περιέχει τις ιδιότητες title και content που αφορούν τον τίτλο και το περιεχόμενο αντίστοιχα (Κώδικας 20). Η ερώτηση πολλαπλής επιλογής είναι ένα αντικείμενο που περιέχει τις ιδιότητες title και content όπως το αντικείμενο της θεωρίας και επιπλέον τις answers, correctAnswer και info (Κώδικας 21). Η ιδιότητα answers αποτελεί τον πίνακα με τις διαθέσιμες απαντήσεις, η correctAnswer δείχνει τη σωστή απάντηση της ερώτησης και η info περιέχει πληροφορίες σχετικές με την απάντηση. Το αντικείμενο που αντιπροσωπεύει τη δραστηριότητα κώδικα περιέχει τις ιδιότητες mode, title, content, editorContent, requirements, testCases (Κώδικας 23). Οι διαφορετικές ιδιότητες που έχει

είναι η mode που δηλώνει τον τύπο της δραστηριότητας π.χ. “JavaScript”, η requirements η οποία περιλαμβάνει τις απαιτήσεις που πρέπει να πληροί ο κώδικας για να εκτελεστεί και αποτελείται από αντικείμενα που περιέχουν το regular expression (test) με βάση το οποίο γίνεται ο έλεγχος και το μήνυμα (message) που θα εμφανιστεί στον χρήστη και η testCases που περιέχει διάφορες κλήσεις (test) της συνάρτησης που θα κατασκευαστεί στη δραστηριότητα και το αποτέλεσμά τους (result). Το αρχείο αυτό αποτελεί βασικό στοιχείο της συγκεκριμένης σκηνής. Στον Πίνακα 14 περιγράφονται οι συναρτήσεις της.

content.json

```
379  "quote293041": {
380      "title": "Σχόλια",
381      "content": [
382          "<p>Στην HTML μπορούν να προστεθούν και σχόλια τα οποία δεν θα εμφανιστούν στην σελίδα.</p>",
383          "<p>Η σύνταξη τους είναι:</p>",
384          "<p class=\"code-snippet\">&lt;!-- Αυτό είναι ένα σχόλιο --&gt;</p>"
385      ]
386  },
```

Κώδικας 20: Αντικείμενο θεωρίας στο αρχείο content.json

content.json

```
3227  "question108261": {
3228      "title": "Events κατα το φόρτωμα της σελίδας",
3229      "content": [
3230          "Ποιο από τα παρακάτω event trigger υποδηλώνει ότι έχει φορτωθεί και ολοκληρωθεί η δημιουργία του DOM;"
3231      ],
3232      "answers": [
3233          {
3234              "content": "DocumentReady"
3235          },
3236          {
3237              "content": "DOMReady"
3238          },
3239          {
3240              "content": "DOMContentLoaded"
3241          },
3242          {
3243              "content": "DocumentLoad"
3244          }
3245      ],
3246      "correctAnswer": 2,
3247      "info": "Το DOMContentLoaded ενεργοποιείται όταν το περιεχόμενο DOM έχει ολοκληρωθεί χωρίς να περιμένει να φ
3248  }
```

Κώδικας 21: Αντικείμενο ερώτησης πολλαπλής επιλογής στο αρχείο content.json

content.json

```
"question135181": {
    "mode": "javascript",
    "title": "Functions",
    "content": [
        "Δημιούργησε τη συναρτηση (function) με όνομα findTheLargerNumber.",
        "Η  συνάρτσ θα δέχεται ως παραμέτρους δύο αριθμούς και επιστρέφει τον μεγαλύτερο.",
        "Τα ονόματα των παραμέτρων είναι num1 και num2."
    ],
    "editorContent": [ ],
    "requirements": [
        {
            "test": "\\b(findTheLargerNumber)\\b\\s*\\(\\w*\\s*\\,\\s*\\w*\\)",
            "message": "Το όνομα της συνάρτησης πρέπει να είναι 'findTheLargerNumber'."
        },
        {
            "test": "\\(\\b(num1)\\b\\s*\\,\\s*\\b(num2)\\b\\)",
            "message": "Ο κώδικας πρέπει να περιέχει τις παραμέτρους num1 και num2."
        },
        {
            "test": "\\s*\\b(return)\\b.*\\;\\s*\\}",
            "message": "Ο κώδικας πρέπει να περιέχει τη λέξη κλειδί 'return'."
        }
    ],
    "testCases": [
        {
            "test": "findTheLargerNumber(5, 4)",
            "result": 5
        },
        {
            "test": "findTheLargerNumber(10, 20)",
            "result": 20
        },
        {
            "test": "findTheLargerNumber(1, 1)",
            "result": 1
        }
    ]
},
```

Κώδικας 22: Αντικείμενο συντάκτη κώδικα στο αρχείο content.json

Πίνακας 14: Συναρτήσεις αρχείου dialog-scene.js

Όνομα	**Περιγραφή**
init	Γραμμές Κώδικα: 14-23 Αρχικοποιεί τις απαραίτητες μεταβλητές με τα δεδομένα που ήρθαν κατά την εκκίνηση της σκηνής.
preload	Γραμμές Κώδικα: 25-31 Είναι υπεύθυνη για τη φόρτωση του αρχείου ace-editor.html που περιέχει τον συντάκτη κώδικα, του αρχείο dialog.html με βάση το οποίο διαμορφώνεται η παρουσίαση της θεωρίας,

	του αρχείου content.json που περιέχει όλες τις πληροφορίες για τη θεωρία και τις δραστηριότητες του παιχνιδιού. Επίσης, εάν η σκηνή αφορά δραστηριότητα που περιέχει συνάκτη κώδικα φορτώνει και το αρχείο που περιέχει τα html elements που θα προστεθούν στον κόσμο του παιχνιδιού.
create	Γραμμές Κώδικα: 33-171 Ανάλογα με τον τύπο του αντικειμένου με βάση το οποίο ξεκίνησε η σκηνή δημιουργεί την οθόνη θεωρίας ή την οθόνη δραστηριότητας. Σε περίπτωση που είναι θεωρία ή ερώτηση πολλαπλής επιλογής προσθέτει την εικόνα της περγαμηνής και τη φόρμα του αρχείου dialog.html ενημερώνοντας τα κατάλληλα σημεία με το περιεχόμενο της θεωρίας ή της ερώτησης. Εάν αφορά δραστηριότητα κώδικα προστίθεται η φόρμα του αρχείου ace-editor.html και η φόρμα που περιέχει τα HTML elements που θα προστεθούν στον κόσμο. Επίσης, αρχικοποιείται η τιμή του συντάκτη κώδικα και τοποθετούνται οι απαραίτητοι listeners στα κουμπιά που περιέχει το αρχείο ace-editor.html.
applyCode	Γραμμές Κώδικα: 173-283 Είναι υπεύθυνη για την εκτέλεση του κώδικα που εισάγει ο χρήστης. Εάν καλείται στα css επίπεδα τότε αρχικά εκτελεί την checkForRequirements για να ελέγξει εάν καλύπτονται οι απαιτήσεις. Μετά καλεί τη συνάρτηση deleteStyleSheet για να αφαιρέσει τα προηγούμενα css styles. Έπειτα με την addStyleSheet προσθέτει τους css κανόνες του χρήστη. Τέλος, εισάγει στην καθολική μεταβλητή children τα html elements της φόρμας και την ακριβή τοποθεσία τους και για κάθε ένα από αυτά καλεί την addElementToWorld. Εάν εκτελείται κώδικας JavaScript τότε αρχικά εκτελεί την checkForJshintErrors που ελέγχει για συντακτηκά λάθη και την checkForRequirements. Σε περίπτωση που η δραστηριότητα είναι λειτουργίας (mode) “javascript” ελέγχει εάν κατά την εκτέλεσή του περνάει τους ελέγχους που ορίζει η δραστηριότητα και είτε εμφανίζει μηνύματα λάθους είτε προσθέτει πλατφόρμα στον κόσμο του παιχνιδιού. Εάν η δραστηριότητα είναι λειτουργίας (mode) “javascriptDom”, τότε εκτελείται ο κώδικας και μεταβάλλονται τα elements στον κόσμο του παιχνιδιού. Τέλος, αν η δραστηριότητα είναι λειτουργίας (mode) “javascriptDomEvents”, τότε εκτελείται ο κώδικας και ανδεν υπάρχουν λάθη εμφανίζονται οι αλλαγές στον κόσμο του παιχνιδιού. Όταν εκτελείται ο κώδικας JavaScript καλείται και η συνάρτηση searchForInfiniteLoops που ελέγχει για ατέρμονες βρόχους.
addHtmlElementsToScene	Γραμμές Κώδικα: 285-289 Εισάγει στο επίπεδο το html αρχείο που φορτώνεται με κάθε δραστηριότητα κώδικα.

initializeAceEditorValue	Γραμμές Κώδικα: 291-294 Αρχικοποιεί την τιμή του συντάκτη κώδικα. Η τιμή αυτή μπορεί να είναι είτε ο κώδικας που συμπλήρωσε ο χρήστης, είτε ο κώδικας που υπάρχει στο αρχείο content.json.
checkForRequirements	Γραμμές Κώδικα: 296-308 Ελέγχει ότι ο κώδικας του χρήστη καλύπτει όλες τις απαραίτητες απαιτήσεις που έχουν οριστεί για τη συγκεκριμένη άσκηση. Εάν δεν πληρούνται οι απαιτήσεις καλεί την createSpeechBubble για την εμφάνιση των κατάλληλων μηνυμάτων.
checkForJshintErrors	Γραμμές Κώδικα: 310-318 Η συνάρτηση αυτή ελέγχει εάν υπάρχουν συντακτικά λάθη με τη χρήση της βιβλιοθήκης jsHint. Σε περίπτωση που βρεθούν σφάλματα καλεί την createSpeechBubble για την εμφάνισή τους.
processAst	Γραμμές Κώδικα: 320-332 Ελέγχει εάν το στοιχείο (ast) που δόθηκε ως όρισμα είναι τύπου for, while ή do while και εισάγει τις πληροφορίες στο αντικείμενο (map) που αποτελεί το δεύτερο όρισμα της. Τέλος, εάν το στοιχείο (ast) περιέχει εμφωλευμένα στοιχεία καλείται ξανά η συνάρτηση.
searchForInfiniteLoops	Γραμμές Κώδικα: 334-357 Είναι υπεύθυνη για την αναζήτηση ύπαρξης ατέρμονα βρόχου στον κώδικα που εισάγει ο χρήστης στον συντάκτη. Καλύπτει τις βασικές περιπτώσεις που αφορούν τα for, while και do while. Αρχικά, διαμορφώνει κατάλληλα τον κώδικα αντικαθιστώντας συγκεκριμένα σημεία με νέα γραμμή και δημιουργεί έναν πίνακα με τον κώδικα χωρισμένο με βάση τις νέες γραμμές. Έπειτα καλώντας τη συνάρτηση processAst και με τη βοήθεια της βιβλιοθήκη esprima, η οποία υλοποιεί ανάλυση κώδικα, δημιουργεί ένα χάρτη που περιέχει πληροφορίες για τις γραμμές του κώδικα που αφορούν βρόχους. Τέλος, συγκεντρώνοντας τις παραπάνω πληροφορίες ανανεώνει τον κώδικα του χρήστη προσθέτοντας ένα μετρητή και πραγματοποιώντας ένα έλεγχο ο οποίος θα οδηγήσει σε τερματισμό εκτέλεσης του κώδικα αν υπερβεί τα όρια (Κώδικας 23).
addElementToWorld	Γραμμές Κώδικα: 359-367 Προσθέτει ένα html element στον κόσμο του παιχνιδιού και του προσθέτει ιδιότητες σύγκρουσης.
findFirstDivId	Γραμμές Κώδικα: 369-381 Βρίσκει το id του βασικού περιέκτη (div) του αρχείου που περιέχει τα HTML elements που θα εισαχθούν στον κόσμο του παιχνιδιού.

findStyleSheetIndex	Γραμμές Κώδικα: 383-392 Ελέγχει εάν υπάρχει ένα συγκεκριμένο style element στο document με βάση το αναγνωριστικό που δέχεται ως όρισμα.
addStyleSheet	Γραμμές Κώδικα: 394-400 Δημιουργεί css style element και το προσθέτει στο head element του document.
deleteStyleSheet	Γραμμές Κώδικα: 402-406 Αφαιρεί τα css styles που ισχύουν σε ένα συγκεκριμένο element.
checkTheAnswer	Γραμμές Κώδικα: 408-423 Ελέγχει εάν η απάντηση που δόθηκε στην ερώτηση πολλαπλής επιλογής είναι σωστή, χρωματίζει ανάλογα τα κουμπιά που αποτελούν τις απαντήσεις και ανανεώνει τη βαθμολογία καλώντας τη συνάρτηση updateScores.
createSpeechBubble	Γραμμές Κώδικα: 425-484 Η συνάρτηση αυτή δημιουργεί τη φούσκα ομιλίας που παρουσιάζει τα μηνύματα σφάλματος κατά την εκτέλεση το κώδικα.
showKnowledgeScene	Γραμμές Κώδικα: 486-489 Εκτελείται όταν πατηθεί το κουμπί “Θεωρία”. Σταματάει με τη χρήση της pause την τρέχουσα σκηνή και ξεκινάει την KnowledgeScene.
exitDialog	Γραμμές Κώδικα: 491-499 Εκτελείται όταν πατηθεί το κουμπί “Έξοδος” ή εκτελεστεί με επιτυχία ο κώδικας του χρήστη. Σταματάει τη συγκεκριμένη σκηνή και συνεχίζει την PlatformerScene.

dialog-scene.js

```
334  searchForInfiniteLoops(code) {
335      let updatedCode = code.replace(/\{[^\S\r\n]/g, "{\n");
336      let codeLines = updatedCode.split(/[\r\n]/);
337      codeLines = codeLines.filter(n => n);
338      let newCode = "";
339      let map = { };
340      this.processAst(esprima.parse(updatedCode, { range: true, loc: true }), map);
341      codeLines.forEach((line, index) => {
342          let type = map["line" + (index + 1)];
343          if(type && (type === "ForStatement" ||
344                  type === "WhileStatement" ||
345                  type === "DoWhileStatement")) {
346              newCode += line +
347                  "\ncmRunStepCount++; if (cmRunStepCount > 100000) {" +
348                  " throw new Error('Execution limit exceeded. Infinite loop?'); }\n";
349          } else
350              newCode += line;
351      });
352      newCode = "var cmRunStepCount = 0;\n" +
353          "var cmExecMonId = setInterval(function() { cmRunStepCount = 0; }, 10000);\n" +
354          "document.stopExecutionMonitor = function() { clearInterval(cmExecMonId); }\n" +
355          newCode;
356      return newCode;
357  }
```

Κώδικας 23: Συνάρτηση searchForInfiniteLoops στο αρχείο dialog-scene.js

5.3.2.7 Σκηνή KnowledgeScene

Με τη σκηνή αυτή σχεδιάζεται η οθόνη που εμφανίζει τη συνολική θεωρία που έχει διαβάσει ο παίκτης σε ένα επίπεδο. Οι συναρτήσεις της παρουσιάζονται στον Πίνακα 15.

Πίνακας 15: Συναρτήσεις αρχείου knowledege-scene.js

Όνομα	**Περιγραφή**
preload	Γραμμές Κώδικα: 11-14 Είναι υπεύθυνη για τη φόρτωση του αρχείου knowledge.html που περιέχει τον σκελετό παρουσίασης της θεωρίας και του αρχείου content.json που περιέχει όλα τα κείμενα της θεωρίας.
create	Γραμμές Κώδικα: 16-36 Προσθέτει στη σκηνή τη φόρμα html που φορτώθηκε. Στη συνέχεια διαβάζει τα περιεχόμενα του αρχείου και με βάση τα αναγνωριστικά της θεωρίας που είναι αποθηκευμένα στο "quotes_ids" προσθέτει το κατάλληλο κείμενο.
exit	Γραμμές Κώδικα: 41-44 Σταματάει την παρούσα σκηνή και εκκινεί την σκηνή StartScene.

5.3.2.8 Σκηνή GameOverScene

Η σκηνή αυτή είναι υπεύθυνη για το σχεδιασμό της οθόνης ολοκλήρωσης επιπέδου. Στον Πίνακα 16 περιγράφονται οι συναρτήσεις της.

Πίνακας 16: Συναρτήσεις αρχείου gameover-scene.js

Όνομα	**Περιγραφή**
init	Γραμμές Κώδικα: 11-16 Αρχικοποιεί τις μεταβλητές condition, time, score, level με τα δεδομένα που έρχονται από τη σκηνή που ξεκίνησε την GameOverScene.
create	Γραμμές Κώδικα: 18-61 Είναι υπεύθυνη για τον σχεδιασμό της σκηνής τοποθετώντας τα κατάλληλα μηνύματα και προσθέτοντας την κατάλληλη λειτουργικότητα στα κουμπιά που περιέχει. Τα μηνύματα επιλέγονται με βάση τη μεταβλητή condition που μπορεί να έχει τις τιμές “win”, “lose” και “nextLevel”.
nextLevel	Γραμμές Κώδικα: 63-77 Η συνάρτηση αυτή σταματάει την παρούσα σκηνή, αρχικοποιεί τα δεδομένα που είναι διαθέσιμα σε όλες τις σκηνές και εκκινεί τις σκηνές HUD και PlattformerScene.
exit	Γραμμές Κώδικα: 79-85 Εκτελείται όταν πατηθεί το κουμπί “Έξοδος” και σταματάει τη σκηνή που τρέχει και την HUD. Τέλος, ξεκινάει τη σκηνή StatScene.

5.3.3 Αντικείμενα του παιχνιδιού

5.3.3.1 Κεντρικός χαρακτήρας - ήρωας

Η κλάση Player με βάση την οποία δημιουργείται ο ήρωας, αλλά και οι υπόλοιπες κλάσεις των αντικειμένων αποτελούν επέκταση της Phaser.GameObjects.Sprite που προσφέρει το Phaser. Στον κατασκευαστή της κλάσης αρχικά δημιουργούνται τα animations του ήρωα. Το πρώτο που παρουσιάζει τον ήρωα να στέκεται σταθερός και το άλλο που χρησιμοποιείται όταν κινείται. Στη συνέχεια ορίζονται τα πλήκτρα με τα οποία θα κινείται ο ήρωας. Έπειτα, αρχικοποιούνται τιμές που περιορίζουν τον ήρωα στα όρια του χάρτη, ορίζουν το μέγεθος εμφάνισης του και το ηχητικό εφέ τραυματισμού του και τέλος προστίθεται στον κόσμο (Κώδικας 24). Στον Πίνακα 17 περιγράφονται οι συναρτήσεις της κλάσης.

player.js

```
5   constructor(config) {
6         super(config.scene, config.x, config.y, "player");
7
8         config.scene.physics.world.enable(this);
9         this.scene = config.scene;
10
11        const anims = this.scene.anims;
12        anims.create({
13          key: "player-idle",
14          frames: anims.generateFrameNumbers("player", { start: 6, end: 8 }),
15          frameRate: 3,
16          repeat: -1
17        });
18        anims.create({
19          key: "player-run",
20          frames: anims.generateFrameNumbers("player", { start: 3, end: 5 }),
21          frameRate: 3,
22          repeat: -1
23        });
24        this.anims.play("player-idle");
25
26        const { LEFT, RIGHT, UP, DOWN } = Phaser.Input.Keyboard.KeyCodes;
27        this.keys = this.scene.input.keyboard.addKeys({
28          left: LEFT,
29          right: RIGHT,
30          up: UP,
31          down: DOWN
32        });
33
34        this.body.collideWorldBounds = true;
35        this.body.setMaxVelocity(300, 600);
36        this.body.setDrag(1000, 0);
37        this.body.setSize(40, 58).setOffset(6, 5);
38        this.sound = this.scene.sound.add("damageSFX");
39        this.sound.setVolume(.2);
40
41        this.alive = true;
42        this.damaged = false;
43        this.fireCoolDown = 0;
44        this.scene.add.existing(this);
45      }
```

Κώδικας 24: Κατασκευαστής της κλάσης Player

Πίνακας 17: Συναρτήσεις της κλάσης Player

Όνομα	**Περιγραφή**
update	Γραμμές Κώδικα: 47-94 Η συνάρτηση αυτή αρχικά ελέγχει εάν η τρέχουσα ζωή του ήρωα είναι μικρότερη ή ίση με το 0. Εάν ισχύει ο έλεγχος καλεί τη gameOver της PlatformerScene. Επίσης, είναι υπεύθυνη για την κίνηση του ήρωα με τα πλήκτρα που έχουν οριστεί και για την επιλογή εμφάνισης του κατάλληλου animation, καθώς και την πυροδότηση της μαγικής επίθεσης πατώντας το κάτω βελάκι.
damage	Γραμμές Κώδικα: 99-111 Με αυτή τη συνάρτηση μειώνεται η ζωή του ήρωα όταν τραυματίζεται και δημιουργείται το αντίστοιχο εφέ, το οποίο περιλαμβάνει τρεμόπαιγμα της κάμερας, ξεθώριασμα και αλλαγή χρώματος στην εικόνα του ήρωα και αναπαραγωγή ηχητικού εφέ.
normalize	Γραμμές Κώδικα: 113-119 Η συνάρτηση αυτή επαναφέρει τον ήρωα στην κατάσταση που βρισκόταν πριν εφαρμοστούν τα εφέ τραυματισμού.

5.3.3.2 Μαγική επίθεση

Η κλάση αυτή δημιουργεί το αντικείμενο που αντιπροσωπεύει τη μαγική επίθεση του ήρωα. Στον κατασκευαστή της δηλώνονται η εικόνα και το μέγεθος της, το ηχητικό εφέ και η ιδιότητα που αφορά τη σύγκρουση με τα όρια του κόσμου. Ο Πίνακας 18 παρουσιάζει τις συναρτήσεις της κλάσης.

Πίνακας 18: Συναρτήσεις της κλάσης Fireball

Όνομα	**Περιγραφή**
update	Γραμμές Κώδικα: 14-25 Μέσα σε αυτή τη συνάρτηση ορίζονται οι συγκρούσεις του αντικειμένου της επίθεσης με το έδαφος και τα εμπόδια του και η επικάλυψη του αντικειμένου με αυτό του εχθρού. Κατά τη σύγκρουση καλείται η collided, ενώ κατά την επικάλυψη η explode και η startKilled της κλάσης του εχθρού.
fire	Γραμμές Κώδικα: 27-36 Είναι υπεύθυνη για την εμφάνιση του αντικειμένου στον κόσμο και την αναπαραγωγή του ηχητικού εφέ.
collided	Γραμμές Κώδικα: 38-43 Ελέγχει την ταχύτητα του αντικειμένου στο άξονα y και αν ισούται με το 0 τη μειώνει για να δημιουργηθεί αναπήδηση. Ελέγχει την ταχύτητα στον άξονα x και αν ισούται με 0 καλεί την explode για να την αφαιρέσει.

explode	Γραμμές Κώδικα: 45-52 Όταν εκτελείται, αφαιρεί το αντικείμενο από τον κόσμο.

5.3.3.3 Εχθρός

Η κλάση Enemy υλοποιεί τον εχθρό του παιχνιδιού. Στον κατασκευαστή της κλάσης αρχικά δημιουργούνται τα animations του εχθρού, τα οποία αποτελούνται από δύο διαφορετικά sprites και ορίζεται το μέγεθός τους. Η επιλογή του sprite γίνεται τυχαία. Στη συνέχεια, για ορισμένα αντικείμενα δημιουργείται ένα εφέ που προσφέρει το Phaser με βάση το οποίο οι εχθροί κινούνται επαναλαμβανόμενα σε μια συγκεκριμένη περιοχή. Οι εχθροί που δεν έχουν αυτό το εφέ κινούνται μέχρι να βρουν κάποιο εμπόδιο και να αλλάξουν κατεύθυνση. Τέλος, αρχικοποιούνται οι βοηθητικές μεταβλητές alive και direction και προστίθεται στον κόσμο (Κώδικας 25). Στον Πίνακα 19 περιγράφονται οι συναρτήσεις της κλάσης.

enemy.js

```
5   constructor(config) {
6         super(config.scene, config.x, config.y, "orcs");
7         config.scene.physics.world.enable(this);
8         this.scene = config.scene;
9
10        const anims = this.scene.anims;
11        anims.create({
12          key: "orc-run1",
13          frames: anims.generateFrameNumbers("orcs", { start: 0, end: 1 }),
14          frameRate: 3,
15          repeat: -1
16        });
17        anims.create({
18          key: "orc-run2",
19          frames: anims.generateFrameNumbers("orcs", { start: 2, end: 3 }),
20          frameRate: 3,
21          repeat: -1
22        });
23
24        this.animOrc = Math.floor(Math.random() * 10) % 2 === 0 ? "orc-run1" : "orc-run2";
25        if(this.animOrc === "orc-run1")
26          this.body.setSize(52, 56).setOffset(10, 7);
27        else
28          this.body.setSize(52, 50).setOffset(10, 13);
29
30        this.tweens = config.tweens;
31        if(this.tweens) {
32          let tweenX = config.x + 180;
33          this.scene.tweens.add({
34            targets: this,
35            x: tweenX,
36            ease: "Linear",
37            duration: 3500,
38            flipX: true,
39            repeat: -1,
40            yoyo: true
41          });
42        }
43
44        this.alive = true;
45        this.direction = -50;
46        this.scene.add.existing(this);
47      }
```

Κώδικας 25: Κατασκευαστής της κλάσης Enemy

Πίνακας 19: Συναρτήσεις της κλάσης Enemy

Όνομα	**Περιγραφή**
update	Γραμμές Κώδικα: 49-59 Είναι υπεύθυνη για την εφαρμογή του animation και την αλλαγή κατεύθυνσης του εχθρού. Επίσης, ορίζονται οι συγκρούσεις με το έδαφος και τα εμπόδια.
startKilled	Γραμμές Κώδικα: 61-70 Η συνάρτηση αυτή καλείται όταν ο εχθρός τραυματίζεται από τη μαγική επίθεση. Αυξάνει τη συνολική βαθμολογία κατά 2 πόντους και καταστρέφει το αντικείμενο.

5.3.3.4 Σημείο θεωρίας και δραστηριότητας

Η κλάση PickUp είναι υπεύθυνη για τη δημιουργία των αντικειμένων που αναπαριστούν τα σεντούκια, τα μαγικά φίλτρα και τις φωτιές. Στον κατασκευαστή αρχικά επιλέγεται το ηχητικό εφέ ανάλογα με τον τύπο του αντικειμένου και ορίζονται όλες οι βασικές μεταβλητές σύμφωνα με τις ιδιότητες (config) που πέρασαν ως όρισμα. Έπειτα σε περίπτωση που το αντικείμενο είναι τύπου code, αφορά δηλαδή δραστηριότητα τύπου συμπλήρωσης κώδικα δημιουργείται το animation της φωτιάς. Στη συνέχεια προστίθεται ένα συμβάν κατά το πάτημα του πλήκτρου F στο οποίο η μεταβλητή openDialog ορίζεται ως αληθής όταν ο παίκτης βρίσκεται εκτός της οθόνης θεωρίας ή δραστηριότητας και ο ήρωας έρχεται σε επαφή με το συγκεκριμένο αντικείμενο. Η μεταβλητή openDialog χρησιμοποιείται για να ελέγχεται εάν έχει ανοίξει η οθόνη θεωρίας ή δραστηριότητας. Τέλος, προστίθεται στον κόσμο το κείμενο “Πάτησε F” με βάση τις συντεταγμένες του αντικειμένου. Οι συναρτήσεις της κλάσης παρουσιάζονται στον Πίνακα 20.

Πίνακας 20: Συναρτήσεις της κλάσης PickUp

Όνομα	**Περιγραφή**
update	Γραμμές Κώδικα: 56-66 Ελέγχει εάν το αντικείμενο αφαιρείται και το αφαιρεί από τον κόσμο. Επίσης, ορίζονται η σύγκρουση με το έδαφος και η επικάλυψη του αντικειμένου με τον ήρωα.
collected	Γραμμές Κώδικα: 68-115 Η συνάρτηση καλείται όταν ο ήρωας έρχεται σε επαφή με το αντικείμενο (Κώδικας 26). Σε περίπτωση που το αντικείμενο είναι τύπου “bonus” ανανεώνει τη βαθμολογία, τη μεταβλητή alpha που καθορίζει την εμφάνιση του αντικειμένου και καλεί την updateCorrectQuestionsMap της σκηνής PlatformerScene. Σε αντίθετη περίπτωση ελέγχει τη μεταβλητή openDialog και εάν είναι αληθής αναπαράγει το ηχητικό εφέ και καλεί την setDialog της PlatformerScene δίνοντας της τα απαραίτητα δεδομένα για να ξεκινήσει η κατάλληλη σκηνή. Σε περίπτωση που είναι τύπου “potion”, δηλαδή αντιπροσωπεύει ερώτηση πολλαπλής επιλογής καλεί την updateCorrectQuestionsMap.

pickUp.js

```
68    collected(pickUp) {
69        if(pickUp.type === "bonus") {
70          Helper.updateScores(pickUp.updateType, this.scene);
71          pickUp.alpha = 0;
72          this.scene.updateCorrectQuestionsMap(this.objectId, true);
73          if(!this.scene.recreatePickups)
74            this.scene.recreatePickups = true;
75          return;
76        }
77        if(pickUp.openDialog) {
78          this.sound.play();
79          if(pickUp.type === "potion") {
80            this.scene.setDialog({
81                type: "question",
82                objectId: this.objectId,
83                displayX: this.displayX,
84                displayY: this.displayY,
85                updateType: pickUp.updateType,
86                platformerScene: this.scene,
87                pickUp: pickUp
88            });
89            this.scene.updateCorrectQuestionsMap(this.objectId, false);
90            if(!this.scene.recreatePickups)
91              this.scene.recreatePickups = true;
92          } else if(pickUp.type === "chest")
93            this.scene.setDialog({
94              type: "quote",
95              objectId: this.objectId,
96              displayX: this.displayX,
97              displayY: this.displayY,
98              updateType: pickUp.updateType,
99              platformerScene: this.scene,
100             pickUp: pickUp
101           });
102         else if(pickUp.type === "code")
103           this.scene.setDialog({
104             type: "code",
105             objectId: this.objectId,
106             displayX: this.displayX,
107             displayY: this.displayY,
108             updateType: pickUp.updateType,
109             platformerScene: this.scene,
110             pickUp: pickUp
111           });
112         pickUp.exitDialogue = false;
113         pickUp.openDialog = false;
114       }
115     }
```

Κώδικας 26: Συνάρτηση collected της κλάσης PickUp

5.3.3.5 Έξοδος επιπέδου

Η κλάση Door δημιουργεί το αντικείμενο το οποίο αποτελεί την έξοδο από το επίπεδο. Στον κατασκευαστή δημιουργείται το animation του και προστίθεται στον κόσμο. Επίσης, ορίζεται και η ύπαρξη σύγκρουσης με το έδαφος. Οι συναρτήσεις της κλάσης παρουσιάζονται στον Πίνακα 21.

Πίνακας 21: Συναρτήσεις της κλάσης Door

Όνομα	**Περιγραφή**
update	Γραμμές Κώδικα: 28-30 Ορίζει την επικάλυψη του αντικειμένου με τον ήρωα και σε περίπτωση που συμβει καλείται η insert.
insert	Γραμμές Κώδικα: 32-67 Ελέγχει εάν έχουν απαντηθεί σωστά λιγότερες από τρεις δραστηριότητες. Σε περίπτωση που ισχύει εμφανίζει σχετικό μήνυμα και καλεί την createPickUps για να προστεθούν εκ νέου τα αντικείμενα με τις λάθος απαντήσεις. Στην αντίθετη περίπτωση καλεί την gameOver δίνοντας σαν δεδομένο το επόμενο επίπεδο.

5.4 Γραφικά και ήχοι

Τα γραφικά και οι ήχοι που επιλέχθηκαν για το παιχνίδι διατίθενται δωρεάν στο διαδίκτυο από τις παρακάτω ιστοσελίδες.

Γραφικά παιχνιδιού:

- https://opengameart.org/
- https://craftpix.net/

Ήχοι παιχνιδιού:

- https://opengameart.org/
- https://freesound.org/

Επίσης, για την επεξεργασία των γραφικών χρησιμοποιήθηκε το εργαλείο Paint.NET (https://www.getpaint.net/) και πιο συγκεκριμένα για την επεξεργασία των sprites η ιστοσελίδα https://ezgif.com/sprite-cutter.

5.5 Σχεδίαση χαρτών

Για το σχεδιασμό των επιπέδων χρησιμοποιήθηκε το πρόγραμμα Tiled Map Editor που παρουσιάστηκε στην Ενότητα 3.3.3. Η δημιουργία έγινε με τη χρήση ενός συγκεκριμένου tileset που αποτελείται από τετράγωνα πλακίδια που έχουν πλάτος 64px

και ύψος 64px. Κατά τη δημιουργία του χάρτη ορίζεται το μέγεθος του με βάση τον αριθμό των πλακιδίων. Ο οριζόντιος χάρτης που δημιουργήθηκε έχει διαστάσεις πλάτους 150 πλακιδίων και ύψους 10 πλακιδίων, ενώ ο κάθετος έχει πλάτος 13 πλακίδια και ύψος 50 πλακίδια. Επίσης, δημιουργήθηκαν ξεχωριστά στρώματα πλακιδίων (tile layers), όπως για παράδειγμα το Ground για την αναπαράσταση πλατφορμών και εδάφους και το Barrier για την αναπαράσταση εμποδίων, για να υπάρχει σαφής διαχωρισμός και καλύτερη διαχείριση στην υλοποίηση του κόσμου του παιχνιδιού. Επιπλέον, για την προσθήκη των αντικειμένων στον κόσμο του παιχνιδιού δημιουργήθηκαν στρώματα αντικειμένων (object layers) τα οποία περιέχουν συγκεκριμένες ιδιότητες. Τα αντικείμενα που ανήκουν στο pickUps στρώμα περιέχουν πληροφορίες όπως το όνομά τους, τον τύπο τους, το αναγνωριστικό που συνδέεται με το περιεχόμενό τους στο αρχείο content.json και επιπλέον συντεταγμένες. Οι πληροφορίες αυτές χρησιμοποιούνται για τη δημιουργία του αντικειμένου και της οθόνης θεωρίας και δραστηριότητας αντίστοιχα. Το αντικείμενο με βάση το οποίο τοποθετούνται οι εχθροί περιέχει την ιδιότητα tweens με βάση την οποία κινείται σε συγκεκριμένο χώρο ή όχι. Τέλος, το αντικείμενο που απεικονίζει την πόρτα εξόδου περιέχει την πληροφορία του επόμενου επιπέδου.

6 Αξιολόγηση του παιχνιδιού

6.1 Μεθοδολογία

Για την αξιολόγηση του παιχνιδιού δημιουργήθηκε ένα ερωτηματολόγιο με βάση το μοντέλο αξιολόγησης εκπαιδευτικών παιχνιδιών MEEGA+ (Petri et al., 2016). Το ερωτηματολόγιο αυτό χωρίζεται σε δύο άξονες. Ο πρώτος αφορά την εμπειρία του χρήστη (player experience) και περιλαμβάνει τις υποκατηγορίες χρηστικότητα (usability), αυτοπεποίθηση (confidence), πρόκληση (challenge), ικανοποίηση (satisfaction), κοινωνική αλληλεπίδραση (social interaction), διασκέδαση (fun), εστίαση προσοχής (focus attention) και σχετικότητα (relevance). Ο δεύτερος παράγοντας αφορά τα μαθησιακά αποτελέσματα όπως τα αντιλαμβάνεται ο παίκτης (perceived learning).

Το ερωτηματολόγιο που δημιουργήθηκε αποτελείται από συνολικά 3 ενότητες. Η πρώτη αφορά τα δημογραφικά στοιχεία και περιέχει 12 ερωτήσεις. Η δεύτερη ενότητα αφορά την εμπειρία του χρήστη και περιλαμβάνει 31 ερωτήσεις. Η τρίτη ενότητα περιέχει 10 ερωτήσεις σχετικές με τα μαθησιακά αποτελέσματα και μία ερώτηση ανοιχτού τύπου για παρατηρήσεις και προτάσεις βελτίωσης. Οι ερωτήσεις κλειστού τύπου των ενοτήτων 2 και 3 που βασίζονται στο μοντέλο MEEGA+ χρησιμοποιούν την κλίμακα Likert από το -2 έως το 2, όπου -2 = Διαφωνώ κάθετα, -1 = Διαφωνώ, 0 = Ούτε συμφωνώ ούτε διαφωνώ, 1 = Συμφωνώ, 2 = Συμφωνώ απόλυτα.

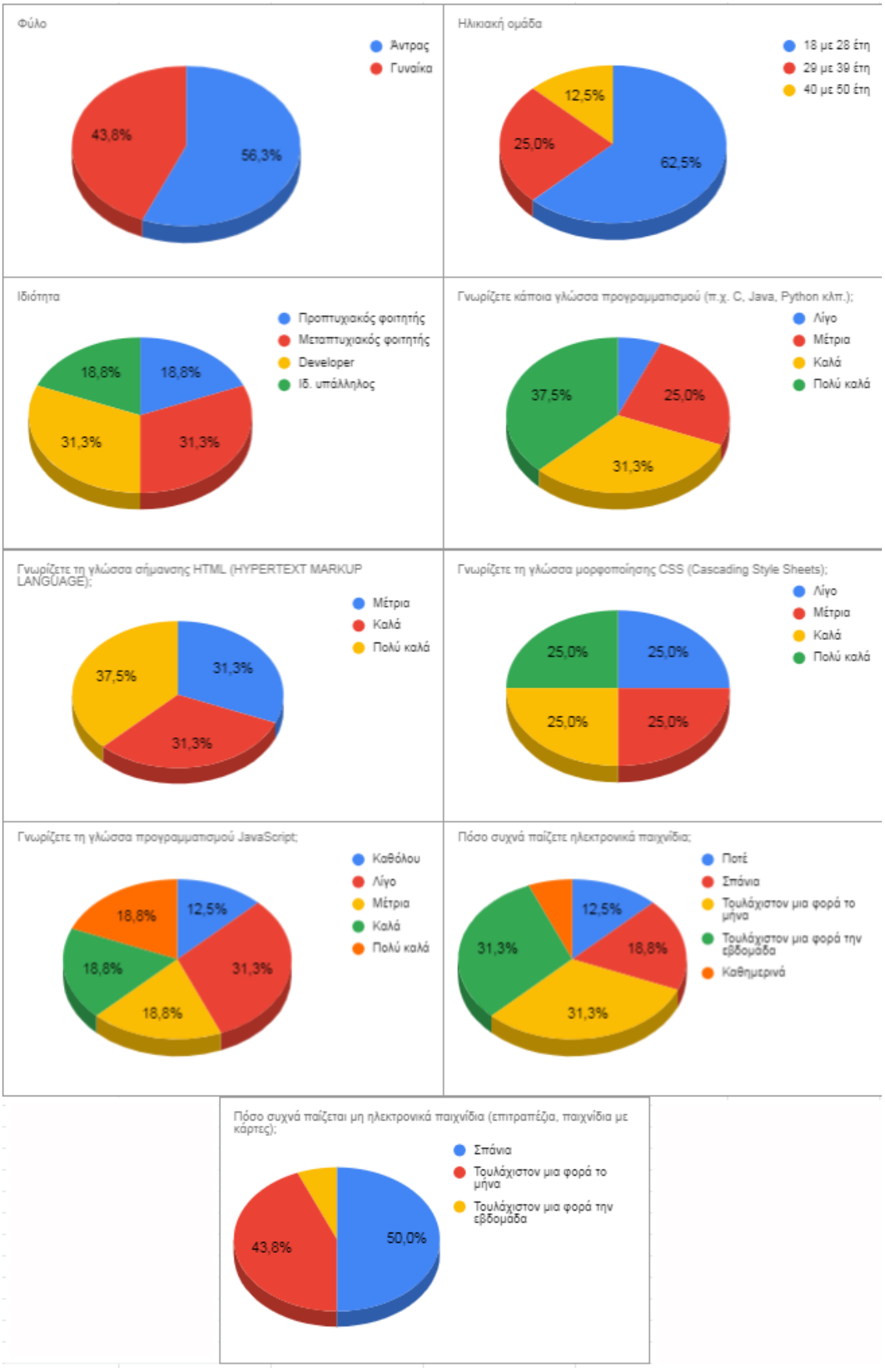
6.2 Αποτελέσματα αξιολόγησης

Το παιχνίδι αξιολογήθηκε συνολικά από 16 χρήστες με τη χρήση ερωτηματολογίου που δημιουργήθηκε με το Google Forms. Τα αποτελέσματα των απαντήσεων παρουσιάζονται αναλυτικά στις επόμενες ενότητες.

6.2.1 Δημογραφικά αποτελέσματα

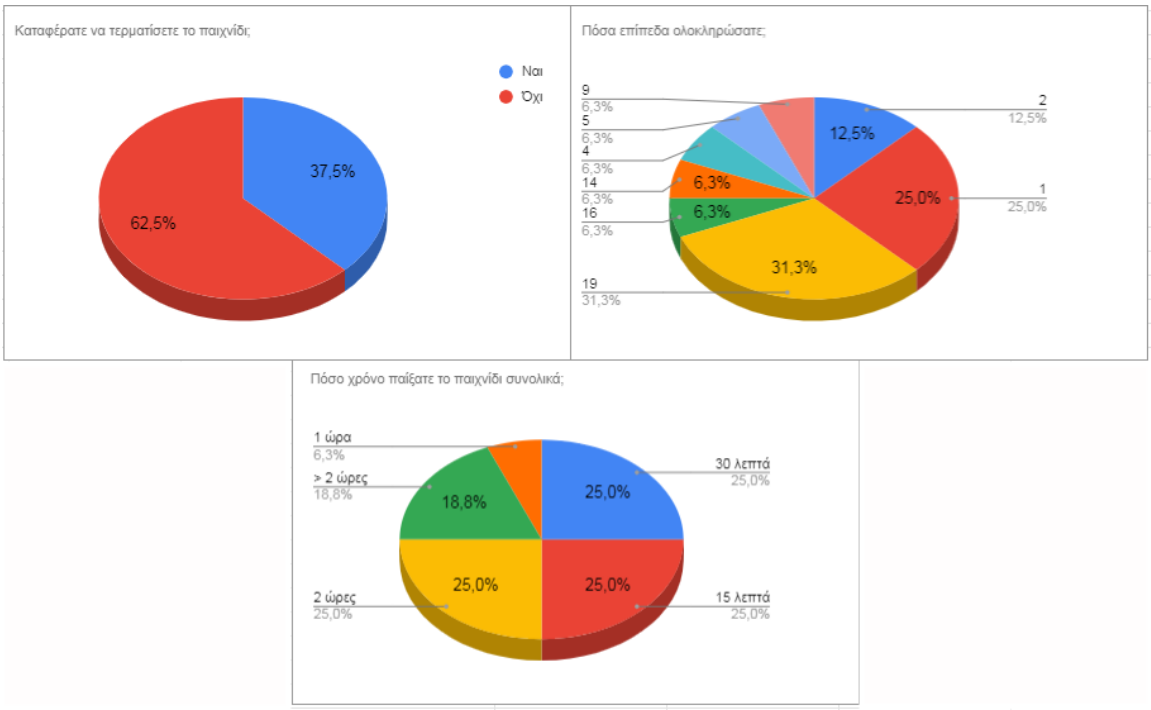
Σύμφωνα με τα αποτελέσματα οι περισσότεροι χρήστες ανήκουν στην ηλικιακή ομάδα 18 με 28 έτη (ποσοστό 62,5%). Το 50% είναι εργαζόμενοι και το μεγαλύτερο ποσοστό εργάζεται σε θέσης σχετικές με την πληροφορική (31,3%). Επίσης είναι αξιοσημείωτο ότι αρκετά μεγάλο ποσοστό από τους συμμετέχοντες (ποσοστό 68,9%) παίζει βιντεοπαιχνίδια τουλάχιστον μία φορά το μήνα, οπότε διαθέτει εμπειρία σχετική με αυτά. Στη συνέχεια παρατηρείται ότι το μεγαλύτερο μέρος των συμμετεχόντων γνωρίζει κάποια γλώσσα προγραμματισμού (ποσοστό 68,8%), είτε καλά είτε πολύ καλά, που σημαίνει ότι είναι εξοικειωμένοι με τη λογική συγγραφής κώδικα. Έπειτα, όλοι οι

συμμετέχοντες έχουν ορισμένες γνώσεις HTML, καθώς όλες οι απαντήσεις συγκεντρώνονται στο μέτρια, καλά και πολύ καλά. Οι γνώσεις που αφορούν το CSS είναι μοιρασμένες σε όλες τις απαντήσεις εκτός από την επιλογή καθόλου και τέλος, η ερώτηση που αφορά τη γνώση της γλώσσας JavaScript συγκεντρώνει ένα μεγάλο ποσοστό (ποσοστό 43,8%) που τη γνωρίζει από καθόλου έως λίγο (Εικόνα 38).



Εικόνα 38: Δημογραφικά αποτελέσματα

Όπως φαίνεται στην Εικόνα 39, το μεγαλύτερο ποσοστό των συμμετεχόντων (ποσοστό 62,5%) δεν κατάφερε να ολοκληρώσει το παιχνίδι. Επίσης, από την ερώτηση "Πόσα επίπεδα ολοκληρώσατε;" παρατηρείται ότι αρκετά μεγάλο ποσοστό των παικτών επαιξε είτε μέχρι το πρώτο επίπεδο (ποσοστό 12,5%) είτε μέχρι το δεύτερο (ποσοστό 25%). Βέβαια μεγάλο ποσοστό αποτελεί και το 37,6% που ολοκλήρωσε από 16 και πάνω επίπεδα. Τέλος, τα ποσοστά του χρόνου που αφιερώθηκε στο παιχνίδι συμβαδίζουν με τα ποσοστά ολοκλήρωσεις καθώς το 50% ασχολήθηκε λιγότερο από 30 λεπτά ενώ το 43,8% περισσότερο από 2 ώρες.

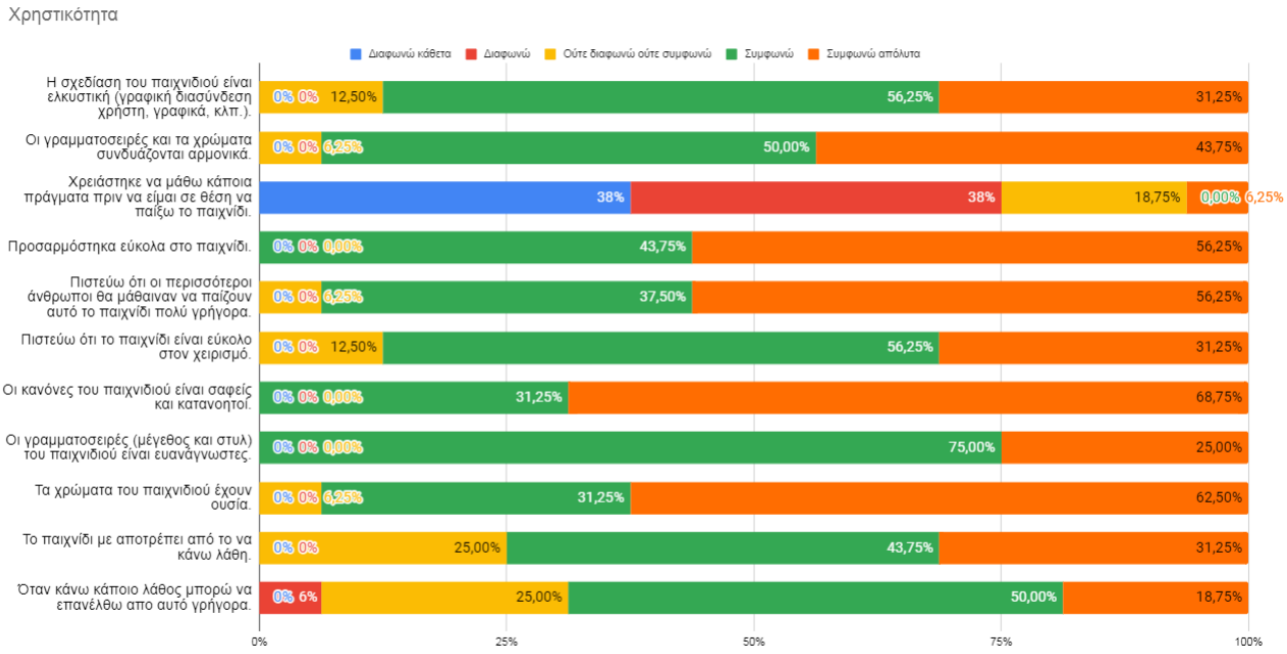


Εικόνα 39: Δημογραφικά αποτελέσματα σχετικά με το παιχνίδι

6.2.2 Εμπειρία χρήστη

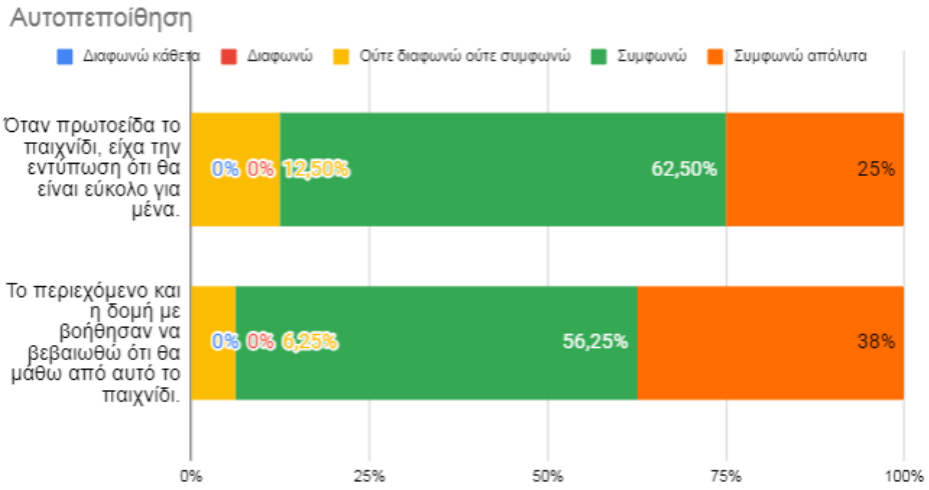
Τα αποτελέσματα της χρηστικότητας που παρουσιάζονται στην Εικόνα 40 είναι αρκετά θετικά καθώς τα ποσοστά όλων των απαντήσεων συγκεντρώνονται στις απαντήσεις "Συμφωνώ" και "Συμφωνώ απόλυτα". Οι συμμετέχοντες βρίσκουν ελκυστικό το παιχνίδι ως προς την σχεδίαση του και συμφωνούν ότι συνδυάζονται αρμονικά οι γραμματοσειρές και τα χρώματα. Το μεγαλύτερο ποσοστό (ποσοστό 76%) υποστηρίζει ότι δεν χρειάστηκε να μάθει κάτι πριν για να είναι σε θέση να παίξει το παιχνίδι. Επίσης, οι περισσότεροι πιστεύουν ότι το παιχνίδι είναι κατανοητό με σαφής κανόνες και εύκολο στο χειρισμό του. Τέλος, ενθαρυντικές είναι και οι απαντήσεις που σχετίζονται με την

αποτροπή λάθους και την επαναφορά από αυτό με το μεγαλύτερο ποσοστό να έχει δώσει την απάντηση “Συμφωνώ”.

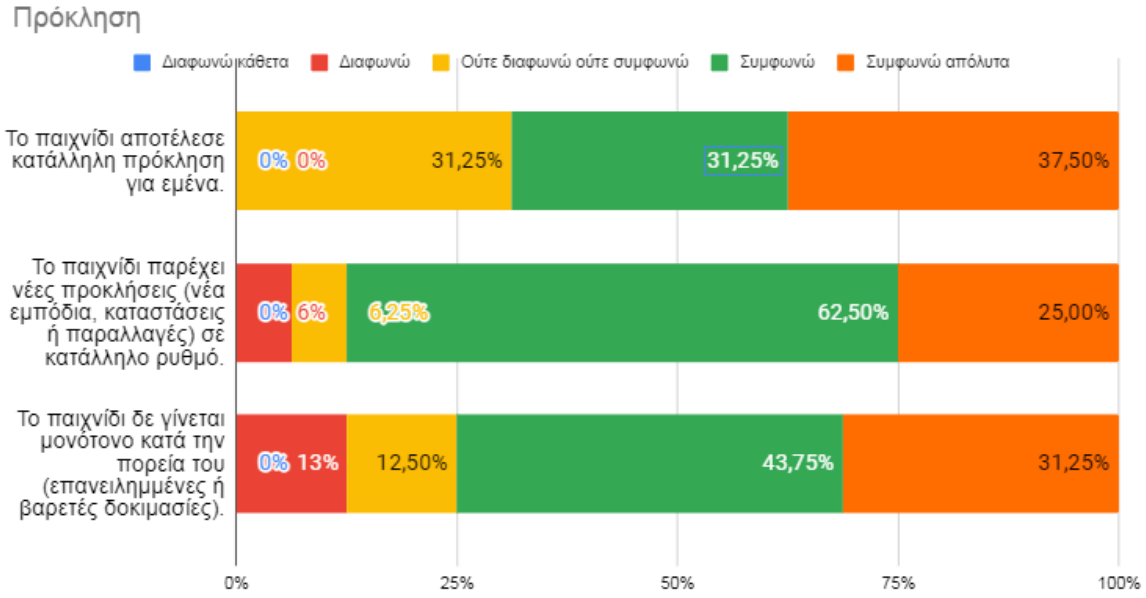


Εικόνα 40: Αποτελέσματα για τη χρηστικότητα

Σύμφωνα με τις Εικόνες 41 και 42 που σχετίζονται με την αυτοπεποίθηση και την πρόκληση παρατηρείται ότι έχουν δοθεί αρκετά θετικές απαντήσεις. Οι συμμετέχοντες σε ποσοστό 77,5% θεώρησαν το παιχνίδι εύκολο όταν το πρωτο είδαν και το 94,25% απάντησε θετικά για το περιεχόμενο και τη δομή του παιχνιδιού. Βέβαια, στην κατηγορία της πρόκλησης ενώ οι περισσότερες απαντήσεις είναι θετικές υπάρχει ένα σημαντικό ποσοστό (ποσοστό 31,25%) που έχει απαντήσει ουδέτερα διαλέγοντας την επιλογή “Ούτε συμφωνώ ούτε διαφωνώ” στην ερώτηση εάν το παιχνίδι αποτέλεσε κατάλληλη πρόκληση για αυτούς.

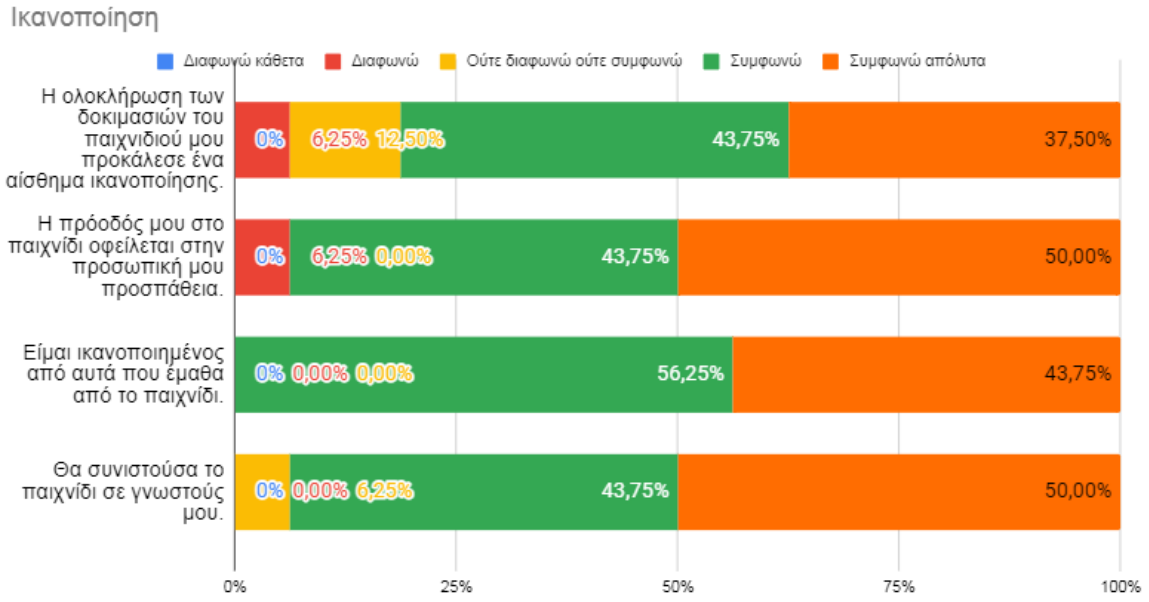


Εικόνα 41: Αποτελέσματα για την αυτοπεποίθηση



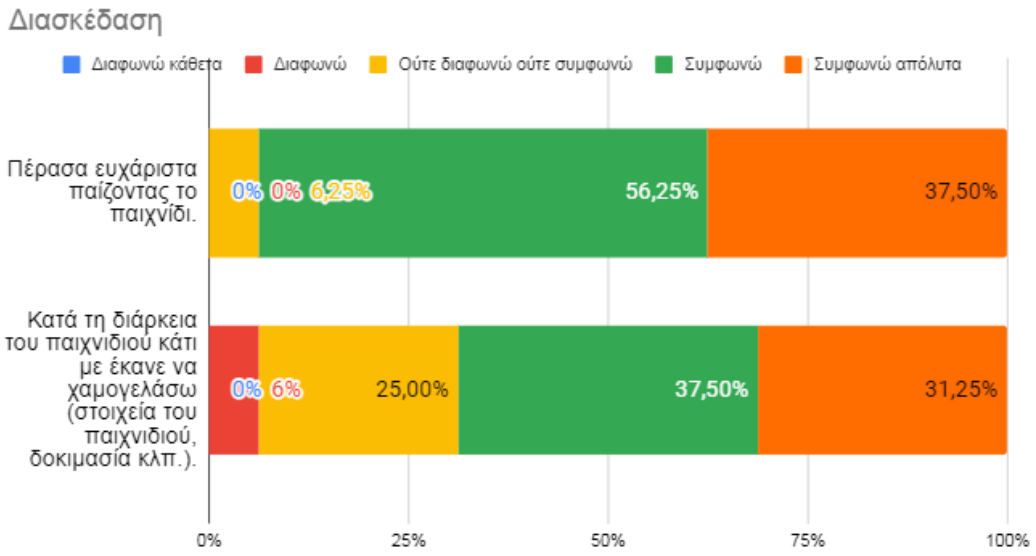
Εικόνα 42: Αποτελέσματα για την πρόκληση

Στην κατηγορία της ικανοποίησης του παίκτη (Εικόνα 43) η πλειοψηφία των συμμετεχόντων αξιολόγησε θετικά όλα τα επιμέρους στοιχεία. Είναι ιδιαίτερα αξιοσημείωτο ότι 93,75% των συμμετεχόντων θα συνιστούσε το παιχνίδι σε γνωστούς του.

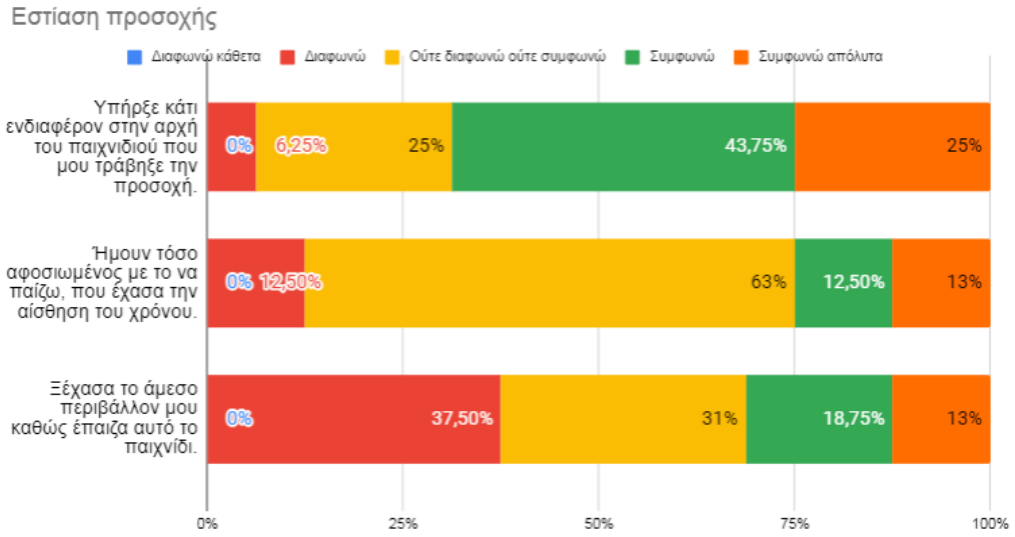


Εικόνα 43: Αποτελέσματα για την ικανοποίηση

Όσον αφορά το κομμάτι της διασκέδασης, του οποίου τα αποτελέσματα παρουσιάζονται στην Εικόνα 44, ένα αρκετά μεγάλο ποσοστό (ποσοστό 93,75%) δηλώνει ότι πέρασε ευχάριστα παίζοντας το παιχνίδι, ενώ το 68,75% ότι υπήρξε κάτι που το έκανε να γελάσει. Ενδιαφέρον παρουσιάζει και η κατηγορία που αφορά την εστίαση της προσοχής (Εικόνα 45), όπου ενώ οι συμμετέχοντες σε μεγάλο ποσοστό δηλώνουν ότι υπήρξε κάτι που τους τράβηξε την προσοχή, στη συνέχεια δηλώνουν ότι δεν παρέμειναν αφοσιωμένοι.

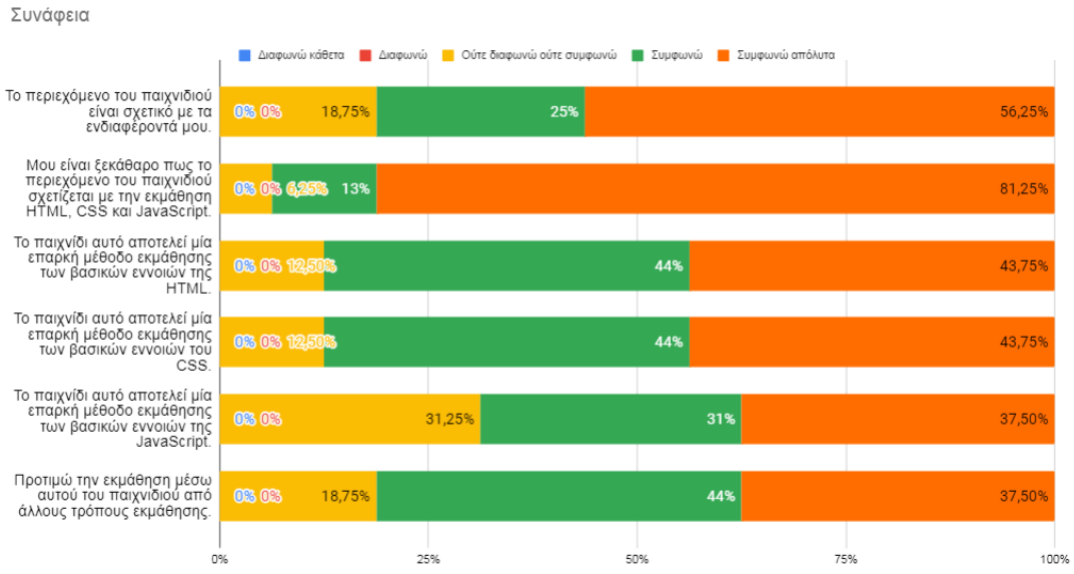


Εικόνα 44: Αποτελέσματα για την διασκέδαση



Εικόνα 45: Αποτελέσματα για την εστίαση της προσοχής

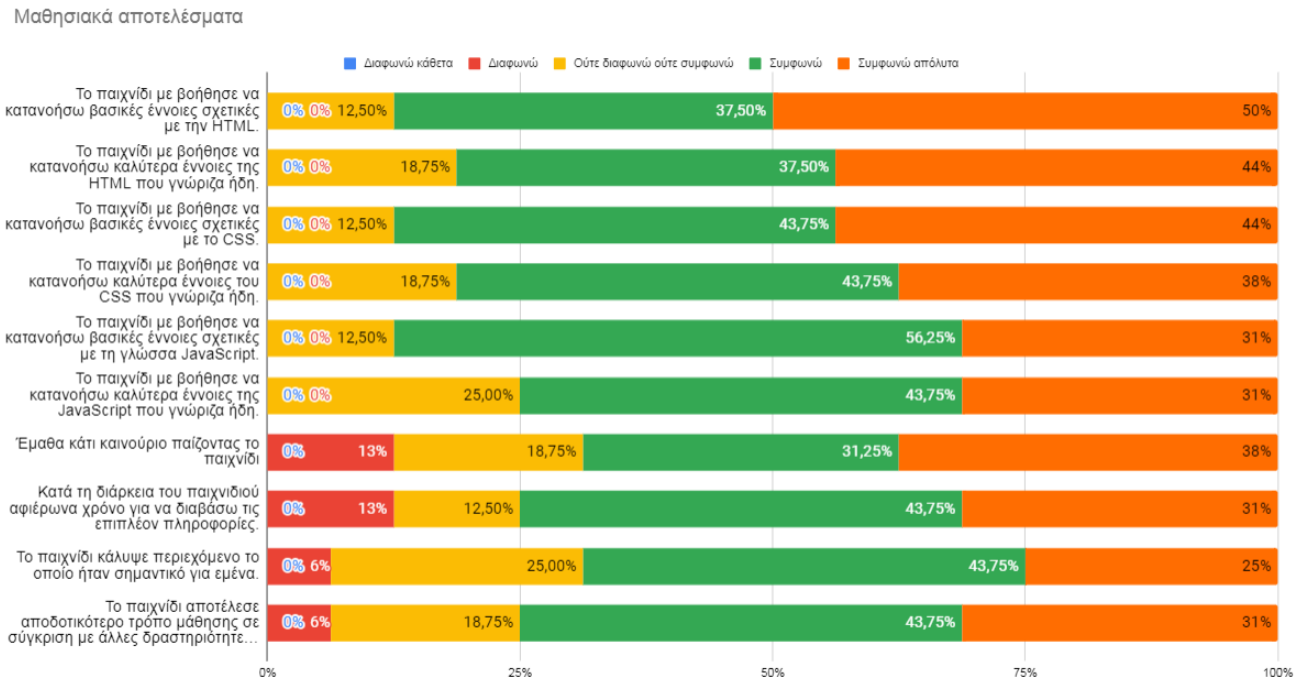
Τέλος, στην κατηγορία που αφορά τη συνάφεια (Εικόνα 46), το 93,75% δηλώνει ότι του είναι ξεκάθαρο ότι το περιεχόμενο του παιχνιδιού σχετίζεται με την εκμάθηση HTML, CSS και JavaScript έχοντας επιλέξει τις απαντήσεις "Συμφωνώ" ή "Συμφωνώ απόλυτα" στην αντίστοιχη ερώτηση. Αρκετά ενθαρρυντικό είναι ότι η πλειονότητα των συμμετεχόντων πιστεύει ότι το παιχνίδι αποτελεί επαρκή μέθοδο και για τις τρεις τεχνολογίες, ωστόσο υπάρχουν και χρήστες που παραμένουν ουδέτεροι (ποσοστό HTML 12,5%, ποσοστό CSS 12,5%, ποσοστό JavaScript 31,25%). Τέλος, αρκετά μεγάλο ποσοστό δηλώνει ότι προτιμάει την εκμάθηση μέσω παιχνιδιού.



Εικόνα 46: Αποτελέσματα για τη συνάφεια

6.2.3 *Μαθησιακά αποτελέσματα*

Τα μαθησιακά αποτελέσματα που προέκυψαν από το ερωτηματολόγιο και παρουσιάζονται στην Εικόνα 47 είναι αρκετά ενθαρρυντικά. Το μεγαλύτερο ποσοστό των ερωτηθέντων δίνει θετική απάντηση στις ερωτήσεις που σχετίζονται με την κατανόηση βασικών εννοιών στην HTML, CSS και JavaScript, ωστόσο κάποιοι χρήστες (ποσοστό HTML 12,5%, ποσοστό CSS 12,5%, ποσοστό JavaScript 12,5%) παραμένουν ουδέτεροι. Επίσης, το 81,5% συμφωνεί ή συμφωνεί απόλυτα ότι κατανόησε καλύτερα έννοιες που γνώριζε ήδη στις HTML και CSS, ενώ το 75% στην JavaScript. Επίσης, θετικό αποτελεί ότι το 69,25% των συμμετεχόντων δήλωσε ότι έμαθε κάτι καινούριο παίζοντας το παιχνίδι και το 69% ότι το παιχνίδι κάλυψε σημαντικό περιεχόμενο για αυτούς. Οι απαντήσεις στην ερώτηση για το αν οι παίκτες αφιέρωσαν χρόνο να διαβάζουν επιπλέον πληροφορίες είναι αρκετά θετικές, ενώ το 25,5% είναι ουδέτερο ή διαφωνεί. Τέλος, το μεγαλύτερο ποσοστό (ποσοστό 74,75%) θεωρεί ότι το παιχνίδι αποτέλεσε αποδοτικότερο τρόπο μάθησης σε σύγκριση με άλλες δραστηριότητες, με το 6% να διαφωνεί με αυτό και το 18,75% να είναι ουδέτερο.



Εικόνα 47: Μαθησιακά αποτελέσματα

6.2.4 *Ερωτήσεις ανοιχτού τύπου*

Το ερωτηματολόγιο στο τέλος περιελάμβανε μία ερώτηση ανοιχτού τύπου όπου οι συμμετέχοντες καλούνταν να συμπληρώσουν τις παρατηρήσεις τους και προτάσεις

βελτίωσης. Συνολικά 5 άτομα απάντησαν στην ερώτηση, με τις περισσότερες απαντήσεις να αναφέρονται στο συντάκτη κώδικα και μερικές στα πλήκτρα χειρισμού του ήρωα. Στον Πίνακα 22 παρουσιάζονται τα σχόλια.

Πίνακας 22: Απαντήσεις σχετικά με παρατηρήσεις και βελτιώσεις

Γράψτε τις παρατηρήσεις σας, προτάσεις βελτίωσης κλπ.
Το μόνο σοβαρό comment έχει να κάνει με τις ερωτήσεις που έχουν ACE editor, όταν γράφεις τη σωστή απάντηση και την υποβάλεις, θα βοηθούσε κάποιο reaction από τη μεριά του παιχνιδιού (ένας ήχος; ένα μικρό εφέ;) για να είσαι σίγουρος πως πας καλά. Στο 1ο τέτοιο quiz δεν πήρε το μάτι μου πως τα βράχια άλλαζαν θέση και νόμιζα πως πρέπει να ξαναπροσπαθήσω Στα minor nitpicks: - Τρώει αρκετή cpu όταν "κάθεται", μήπως υπάρχει δυνατότητα frame rate capping? - Θέλει λίγο ακόμα proof reading (πχ >head><title>Title of the HTML page</title></head<) - Μήπως τα πλήκτρα WASD θα ήταν πιο ξεκούραστα; - Εγώ δεν βάζω πάντα μουστάκια στους βρόγχους (ασυγχώρητος!) - Κάπου στα instructions "Ποιο θα είναι το αποτέλεσμα που θα εμφανίσει ο παρακάτω κώδικας" πρέπει να προστεθεί "όταν κληθεί η myObject.func()" - Κανά-δυο φορές, όταν ήμουν στον αέρα και πετούσα fireball, αυτό πήγαινε προς τα κάτω αντί για ίσια Γενικότερη κουβέντα - Μπράβο! ΦΟΒΕΡΗ δουλειά!!!! Και pixel art που λατρεύω! Και platformer σε στιλ super Mario που λατρεύω επίσης! - Εύγε για την εναλλαγή μουσικής σε chase scene από horror movie με του που μπαίνει η CSS, ταιριάζει απόλυτα :-) - Το παράδειγμα "var text = 'x'; function showText(){ console.log(text); var text = 'y'; }; showText();" μου έκανε φοβερή εντύπωση! Άτιμη JavaScript... - Το ότι εμφάνισα το "Apply" και μετά το πάτησα για να πάρω το hint με κούφανε
Θα μπορούσαν ίσως να υπάρχουν hints (αν και η θεωρία είναι παρόμοιας λογικής)
Θεωρώ ότι χρειάζεται βελτιώσει στα σημεία που ο παίκτης γράφει κώδικα όσο αφορά την διαδικασία του debug της συνάρτησης που έγραψε. Η προσθήκη κονσόλας για εμφάνιση περιεχομένου των στοιχείων ή η χρήση του inspect του browser σε κάποια σημεία πιστεύω θα βοηθούσε τον χρήστη για την καλύτερη κατανόηση της θεωρίας.
Το παιχνίδι είναι αρκετά ενδιαφέρον βοηθώντας με ένα διασκεδαστικό τρόπο, να προσφέρει γνώσεις σχετικές με τα αντικείμενα που πραγματεύεται. Ένα μικρό tutorial στο πρώτο επίπεδο, πιθανό να βοηθούσε για την πιο ομαλή ένταξη του παίκτη στους κανόνες του παιχνιδιού. Επίσης ένα μικρό θέμα, τουλάχιστον μέχρι να το συνηθίσει ο χρήστης, είναι το κάτω πλήκτρο που χρησιμοποιείται για την επίθεση. Ίσως η επιλογή

κάποιου διαφορετικού πλήκτρου να ήταν μια λύση.
στα επίπεδα με την html να υπήρχε κάποιο task με συμπλήρωση κώδικα και όχι μόνο πολλαπλής επιλογής, η δυνατότητα χειρισμού του παίκτη και με άλλα πλήκτρα (πχ awds)

6.2.5 Συμπεράσματα αξιολόγησης

Σύμφωνα με τα αποτελέσματα της αξιολόγησης το παιχνίδι συγκέντρωσε αρκετά θετικά σχόλια τόσο σε θέματα ψυχαγωγίας όσο και σε θέματα χρηστικότητας. Επίσης, αρκετά ενθαρρυντικά ήταν τα σχόλια που αφορούν το εκπαιδευτικό περιεχόμενο και στις τρεις τεχνολογίες που παρουσιάζει το παιχνίδι. Ωστόσο, η κατηγορία που αφορά την εστίαση της προσοχής του παίκτη και έχει να κάνει με την αφοσίωση του στο παιχνίδι επιδέχεται βελτιώσεις καθώς είναι αυτή που συγκέντρωσε τα χαμηλότερα ποσοστά.

7 Επίλογος

7.1 Σύνοψη και συμπεράσματα

Η συγκεκριμένη διπλωματική εργασία είχε ως στόχο τη σχεδίαση και την ανάπτυξη ενός παιχνιδιού σοβαρού σκοπού που διδάσκει τις τρεις βασικές τεχνολογίες ανάπτυξης ιστοτόπων. Οι τεχνολογίες αυτές αποτελούνται από τη γλώσσα σήμανσης HTML, τη γλώσσα CSS που χρησιμοποιείται για την εμφάνιση των ιστοτόπων και τη γλώσσα προγραμματισμού JavaScript. Σκοπός ήταν οι παίκτες που δεν είχαν κάποιο υπόβαθρο σε σχέση με τις τεχνολογίες αυτές να γνωρίσουν τις βασικές έννοιες τους, αλλά και οι παίκτες που έχουν ήδη κάποια εμπειρία να εξοικειωθούν με αυτές.

Κατά το σχεδιασμό του παιχνιδιού δόθηκε αρκετή έμφαση στη δημιουργία δραστηριοτήτων που θα επηρεάζουν τον κόσμο του παιχνιδιού με στόχο να προσφέρει μεγαλύτερη ικανοποίηση στον χρήστη. Οπότε εκτός από τις ερωτήσεις πολλαπλής επιλογής που συμπληρώνει ο χρήστης παίζοντας, δημιουργήθηκαν και δραστηριότητες σύνταξης κώδικα στα επίπεδα που αφορούν το CSS και τη JavaScript. Οι δραστηριότητες αυτές βρίσκονται σε κομβικά σημεία του επιπέδου που εμποδίζουν τη συνέχεια του, οπότε η σωστή πληροφόρηση και ανατροφοδότηση του χρήστη αποτέλεσε σημαντικό κομμάτι.

Τέλος, με βάση την πιλοτική αξιολόγηση που πραγματοποιήθηκε τα αποτελέσματα ήταν αρκετά θετικά καθώς στις περισσότερες κατηγορίες συγκεντρώθηκαν αρκετά υψηλά ποσοστά στις θετικές απαντήσεις. Εξαίρεση βέβαια αποτελεί η κατηγορία που αφορά την εστίαση της προσοχή του παίκτη η οποία δείχνει ότι το παιχνίδι δεν κατάφερε να προκαλέσει μεγάλο εμβύθισης.

7.2 Όρια και περιορισμοί της έρευνας

Το παιχνίδι που δημιουργήθηκε στα πλαίσια της διπλωματικής εργασίας έχει ως στόχο τη διδασκαλία των HTML, CSS και JavaScript. Επειδή το περιεχόμενο των ενοτήτων αυτών είναι αρκετά ευρύ και συνεχώς εξελίσσεται, το παιχνίδι είναι αδύνατο να το καλύψει οπότε περιορίζεται στις βασικές έννοιες τους.

Επιπλέον ένας ακόμη περιορισμός της παρούσας εργασίας είναι ότι είναι αδύνατη η αναπαραγωγή του σε κινητές συσκευές, παρότι έχει δημιουργηθεί με τεχνολογίες που υποστηρίζονται από όλους τους δημοφιλείς φυλλομετρητές. Αυτό συμβαίνει διότι η καθοδήγηση του ήρωα του παιχνιδιού πραγματοποιείται με τη χρήση πληκτρολογίου και

ο συντάκτης κώδικα που υπάρχει στην εφαρμογή δεν θα είναι εύκολα διαχειρίσιμος από αυτές τις συσκευές.

7.3 Μελλοντικές Επεκτάσεις

Μία εύκολη βελτίωση που αναφέρθηκε στις παρατηρήσεις του ερωτηματολογίου και ενσωματώθηκε στον κώδικα του παιχνιδιού είναι η προσθήκη ηχητικού εφέ κατά την εκτέλεση του κώδικα που εισάγει ο χρήστης καθώς μπορεί να μην παρατηρήσει την αλλαγή που επέφερε ο κώδικάς του.

Επίσης, μία ακόμη μελλοντική επέκταση που αναφέρθηκε από κάποιους χρήστες στο ερωτηματολόγιο είναι ο χειρισμός του ήρωα να γίνεται και με τα πλήκτρα WASD εκτός από τα βελάκια, αλλά και η επίθεση του ήρωα να γίνεται με διαφορετικό πλήκτρο από το κάτω βελάκι καθώς ορισμένοι το θεώρησαν δύσχρηστο. Αυτή η αλλαγή αν και αρχικά φαίνεται εύκολα υλοποιήσιμη επιφέρει προβλήματα στη λειτουργία του συντάκτη του κώδικα οπότε για την πραγματοποίηση της απαιτείται περαιτέρω μελέτη.

Επιπλέον, μία ακόμη σημαντική επέκταση του παιχνιδιού αποτελεί η βελτίωση του συντάκτη κώδικα. Αρχικά, θα μπορούσαν να προστεθούν περισσότερα και πιο στοχευμένα μηνύματα προς τον χρήστη κατά την εκτέλεση του κώδικα ώστε να υπάρχει καλύτερη καθοδήγηση και να αποφεύγονται τα λάθη. Στα επίπεδα που βρίσκονται στην ενότητα της JavaScript θα μπορούσε να προστεθεί κονσόλα η οποία θα επιτρέπει στο χρήστη να τρέξει και να δει τα αποτελέσματα του κώδικα που έχει γράψει πριν πατήσει το κουμπί “Εφαρμογή”.

Μία ακόμη παρατήρηση που αναφέρθηκε στο ερωτηματολόγιο και αποτελεί επέκταση του παιχνιδιού είναι η προσθήκη δραστηριοτήτων που αφορούν τη σύνταξη κώδικα στα επίπεδα της ενότητας που αφορά την HTML, καθώς τώρα περιέχει μόνο ερωτήσεις πολλαπλής επιλογής. Μελλοντικά το παιχνίδι θα μπορούσε να αποτελείται μόνο από δραστηριότητες σύνταξης κώδικα.

Τέλος, καλό θα ήταν να αποθηκεύονται περισσότερες πληροφορίες στη βάση δεδομένων που αφορούν τις ενέργειες του χρήστη για να έχει καλύτερη ανατροφοδότηση σχετικά με την πρόοδό του. Κάποια από αυτά τα δεδομένα είναι πόσες φορές προσπάθησε να παίξει ένα επίπεδο μέχρι να το ολοκληρώσει, το χρόνο που αφιέρωσε στις δραστηριότητες των επιπέδων καθώς και να αποθηκεύονται όλοι οι χρόνοι των προσπαθειών του για να βλέπει την πρόοδό του.

Βιβλιογραφία

Abt, Clark C. (1970). Serious games, New York: Viking Press.

Bellotti, F., Berta, R., De Gloria, A., Ott, M, Arnab, S., et al. (2011). Designing Serious Games for education: from Pedagogical principles to Game Mechanisms. Proceedings 5th European Conference on Game-Based Learning, Athens, Greece. pp.26-34

Building a REST API with Express, Node, and MongoDB. Ανακτήθηκε Μάιος 25, 2022 από https://www.mongodb.com/languages/express-mongodb-rest-api-tutorial

CodeCombat. Ανακτήθηκε Απρίλιος 4, 2022 από https://codecombat.com/

Crunchzilla. Ανακτήθηκε Απρίλιος 4, 2022 από https://www.crunchzilla.com/

CSS Diner. Ανακτήθηκε Απρίλιος 4, 2022 από https://flukeout.github.io/

Flexbox Froggy. Ανακτήθηκε Απρίλιος 4, 2022 από https://flexboxfroggy.com/#el

GRID GARDEN. Ανακτήθηκε Απρίλιος 4, 2022 από https://cssgridgarden.com/

How To Implement API Authentication with JSON Web Tokens and Passport. Ανακτήθηκε Μάιος 25, 2022 από https://www.digitalocean.com/community/tutorials/api-authentication-with-json-web-tokensjwt-and-passport

Ibrahim, R., & Jaafar, A. (2009). Educational Games (EG) Design Framework: Combination of Game Design, Pedagogy and Content Modeling, International Conference on Electrical Engineering and Informatics, Selangor, Malaysia. pp. 293-298

Laamarti, F., Eid, M., and El Saddik, A., (2014) An Overview of Serious Games. International Journal of Computer Games Technology, vol 2014, 15 pages

Liu, Y. and Phelps, G. (2011). Challenges and professional tools used when teaching web programming. Journal of Computing Sciences in Colleges, 26(5), pp.116-121

Makzan (2015). HTML5 Game Development by Example: Beginner's Guide - Second Edition. Birmingham: Packt Publishing

Micheal, D., Chen, S. (2006). Serious Games: Games that Educate, Train and INform, Boston: Thomson.

Node.js. Ανακτήθηκε Μάιος 25, 2022 από https://nodejs.dev/

Petri, G., Wangenheim., G. C., & Borgatto F. A. (2018). MEEGA+: A Method for the Evaluation of Educational Games for Computing Education. Brazilian Institute for Digital Convergence, Federal University of Santa Catarina, Brazil.

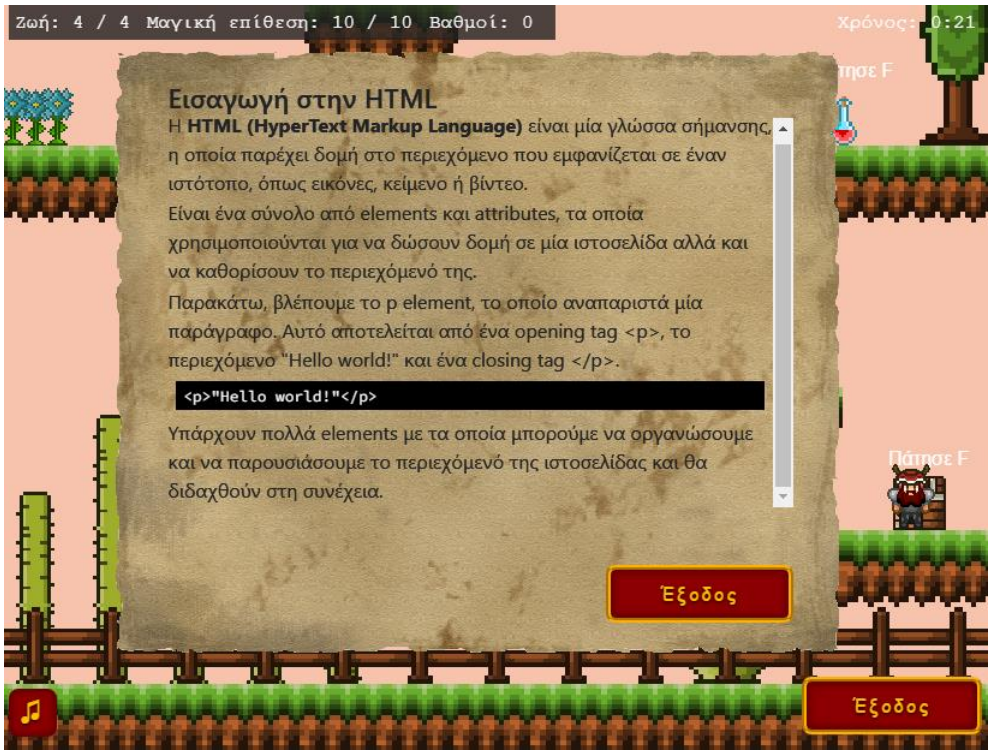
Phaser. Ανακτήθηκε Μάρτιος 15, 2022 από https://phaser.io/

Rauschmayer, A. (2014). Speaking JavaScript, Sebastopol: O'Reilly Media.

Yue, K. B. and Ding, W. (2004). Design and evolution of an undergraduate course on web application development. ACM SIGCSE Bulletin, pp.22-26

Zyda, M. (2005). From Visual Simulation to Virtual Reality to Games. IEEE Computer Society, Information Sciences Institute, California. pp.25-32

Παράρτημα



Εικόνα 48: Επίπεδο 1 - HTML στιγμιότυπο θεωρίας

Εικόνα 49: Επίπεδο 1 - HTML στιγμιότυπο στο παιχνίδι

Εικόνα 50: Επίπεδο 2 - HTML στιγμιότυπο στο παιχνίδι

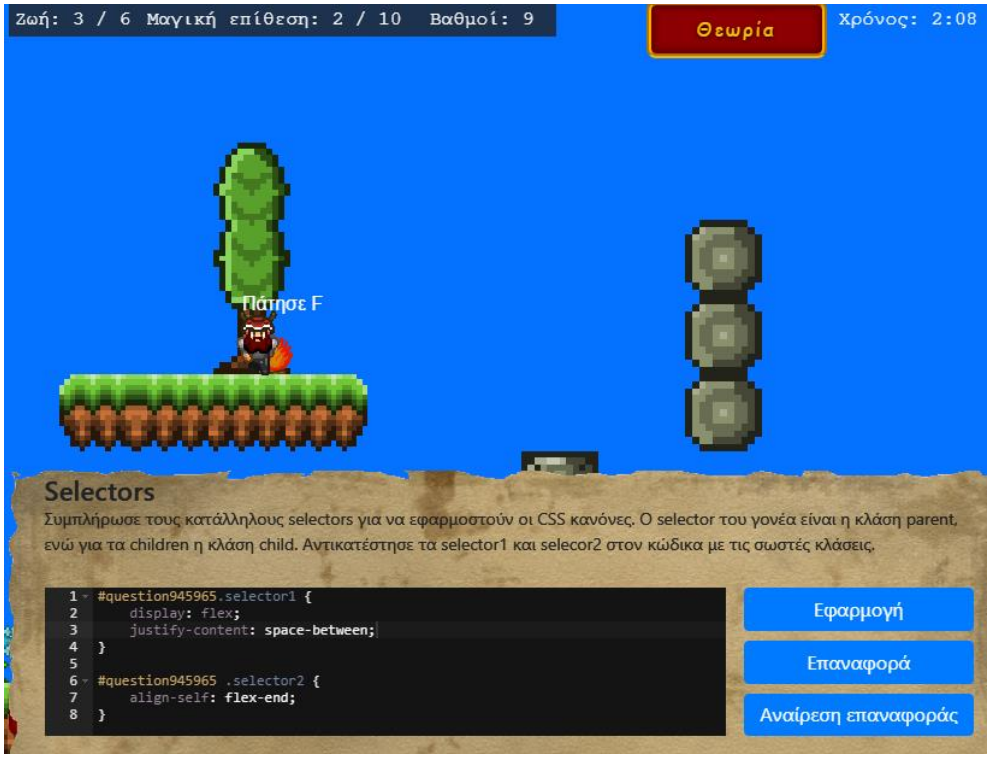
Εικόνα 51: Επίπεδο 3 - HTML στιγμιότυπο στο παιχνίδι

Εικόνα 52: Επίπεδο 4 - HTML στιγμιότυπο στο παιχνίδι

Εικόνα 53: Επίπεδο 5 - HTML στιγμιότυπο στο παιχνίδι

Εικόνα 54: Επίπεδο 5 - HTML στιγμιότυπο εξόδου

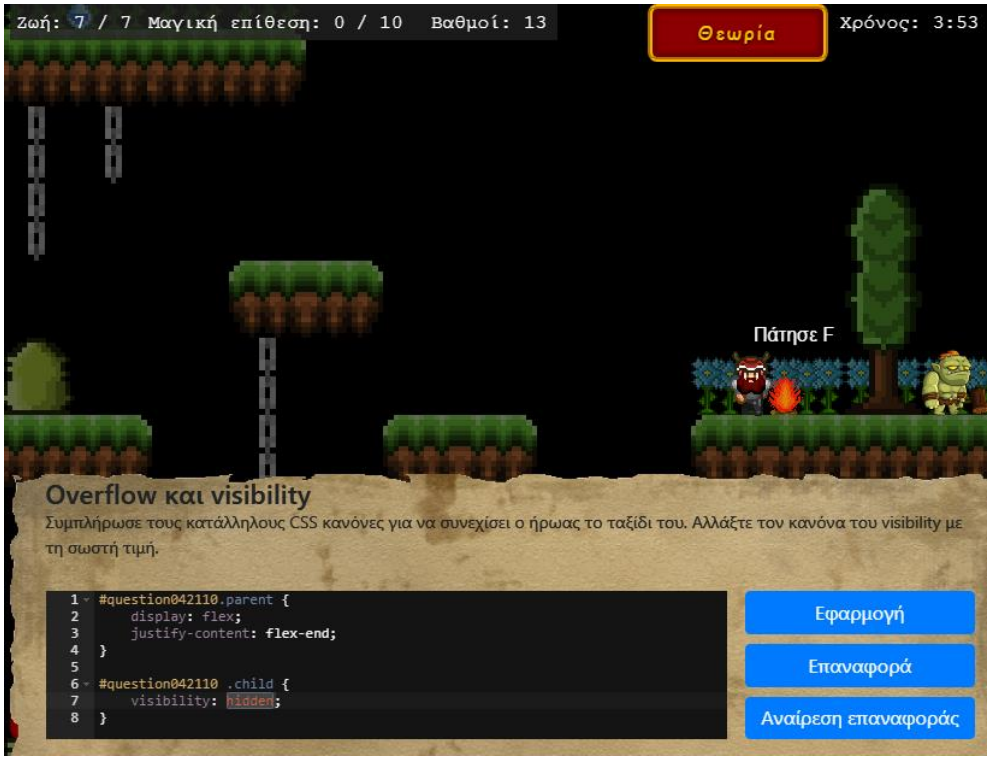
Εικόνα 55: Επίπεδο 6 - CSS στιγμιότυπο με εμπόδια



Εικόνα 56: Επίπεδο 6 - CSS στιγμιότυπο δραστηριότητας κώδικα

Εικόνα 57: Επίπεδο 7 - CSS στιγμιότυπο με μηνύματα λάθους

Εικόνα 58: Επίπεδο 7 - CSS στιγμιότυπο με εμφάνιση κρυφού μηνύματος

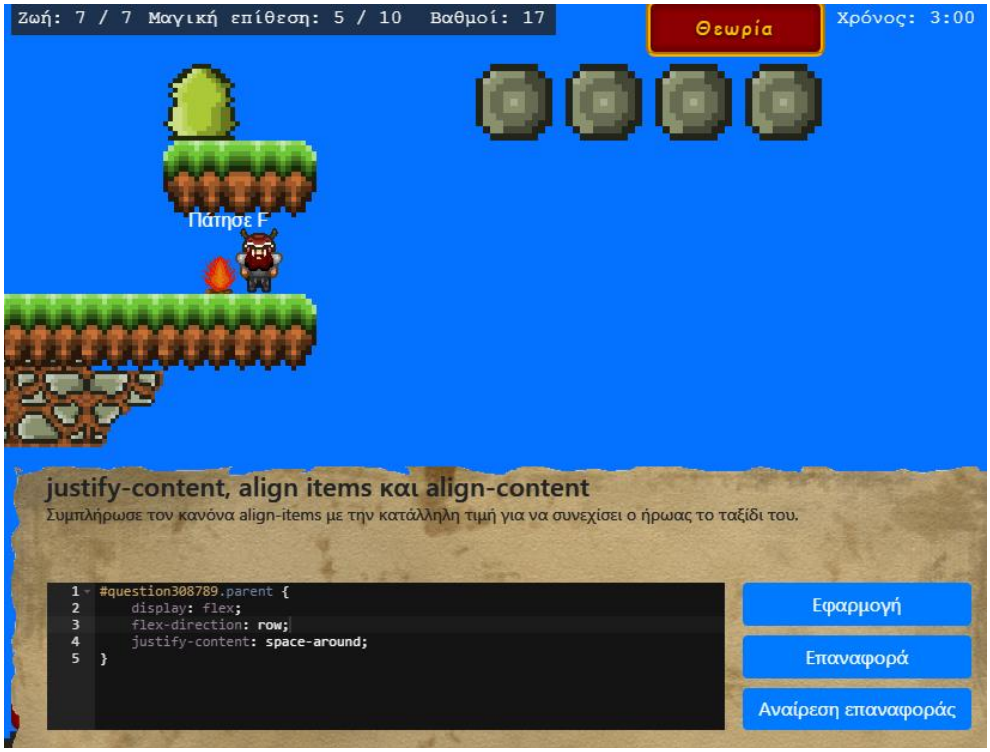


Εικόνα 59: Επίπεδο 8 - CSS στιγμιότυπο με δραστηριότητα κώδικα

Εικόνα 60: Επίπεδο 8 - CSS στιγμιότυπο προσθήκης πλατφόρμας

Εικόνα 61: Επίπεδο 9 - CSS στιγμιότυπο παιχνιδιού

Εικόνα 62: Επίπεδο 10 - CSS στιγμιότυπο θεωρίας



Εικόνα 63: Επίπεδο 10 - CSS στιγμιότυπο δραστηριότητας κώδικα

Εικόνα 64: Επίπεδο 10 - CSS στιγμιότυπο μετά την εκτέλεση κώδικα

Εικόνα 65: Επίπεδο 11 - CSS στιγμιότυπο παιχνιδιού

Εικόνα 66: Επίπεδο 12 - JavaScript στιγμιότυπο παιχνιδιού

Εικόνα 67: Επίπεδο 13 - JavaScript στιγμιότυπο παιχνιδιού

Εικόνα 68: Επίπεδο 13 - JavaScript στιγμιότυπο προσθήκης πλατφόρμας

Εικόνα 69: Επίπεδο 14 - JavaScript στιγμιότυπο παιχνιδιού

Εικόνα 70: Επίπεδο 14 - JavaScript στιγμιότυπο ερώτησης πολλαπλής επιλογής

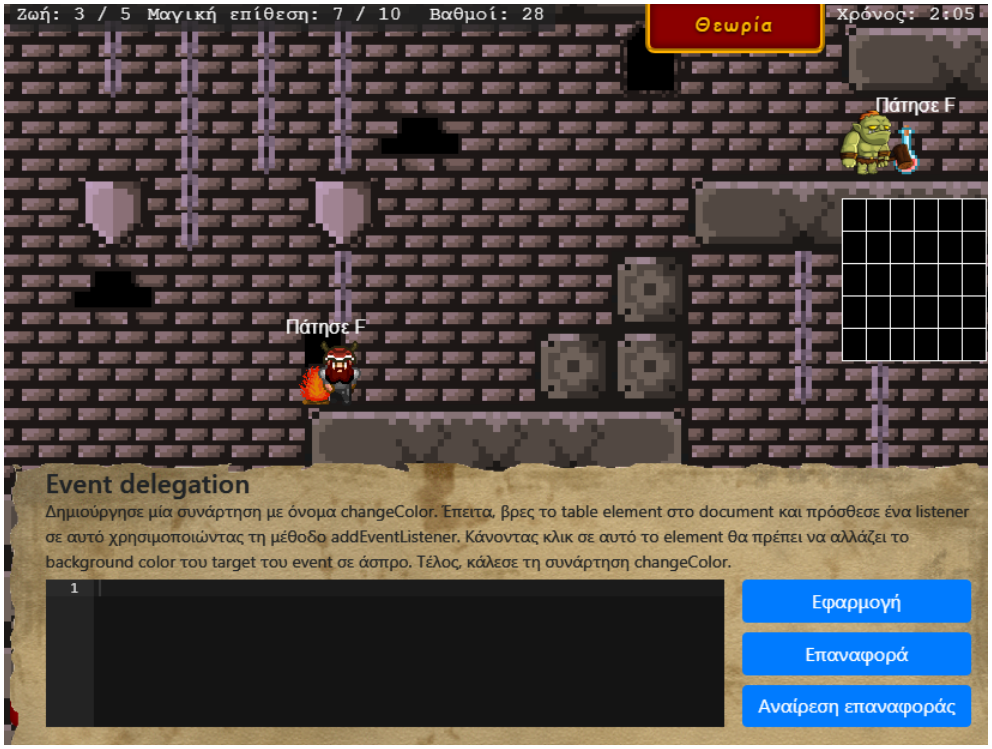
Εικόνα 71: Επίπεδο 15 - JavaScript στιγμιότυπο παιχνιδιού

Εικόνα 72: Επίπεδο 16 - JavaScript στιγμιότυπο παιχνιδιού

Εικόνα 73: Επίπεδο 17 - JavaScript στιγμιότυπο παιχνιδιού με εμπόδιο

Εικόνα 74: Επίπεδο 17 - JavaScript στιγμιότυπο παιχνιδιού αφαίρεση εμποδίου

Εικόνα 75: Επίπεδο 18 - JavaScript στιγμιότυπο παιχνιδιού



Εικόνα 76: Επίπεδο 19 - JavaScript στιγμιότυπο συμπλήρωσης κώδικα

Εικόνα 77: Επίπεδο 19 - JavaScript στιγμιότυπο μετά την εκτέλεση κώδικα

Εικόνα 78: Τερματισμός παιχνιδιού