ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΥΠΟΛΟΓΙΣΤΩΝ

Σχεδιασμός και Ανάπτυξη ενός Εκπαιδευτικού Παιχνιδιού για Εκμάθηση Αγγλικών

Design and Development of an Educational Game for Learning English

Διπλωματική Εργασία

ΝΙΚΟΛΑΟΣ ΤΑΤΣΗΣ

ΑΜ 151050

ΕΠΙΒΛΕΠΩΝ

ΓΕΩΡΓΙΟΣ ΜΠΑΡΔΗΣ

ΕΠΙΚΟΥΡΟΣ ΚΑΘΗΓΗΤΗΣ

ΑΘΗΝΑ

2023

Διπλωματική Εργασία

Σχεδιασμός και Ανάπτυξη ενός Εκπαιδευτικού Παιχνιδιού για Εκμάθηση Αγγλικών

Design and Development of an Educational Game for Learning English

ΝΙΚΟΛΑΟΣ ΤΑΤΣΗΣ

ΕΠΙΒΛΕΠΩΝ

ΓΕΩΡΓΙΟΣ ΜΠΑΡΔΗΣ

ΕΠΙΚΟΥΡΟΣ ΚΑΘΗΓΗΤΗΣ

ΑΜ 151050

Α/Α	ΟΝΟΜΑ ΕΠΩΝΥΜΟ	ΒΑΘΜΙΔΑ/ΙΔΙΟΤΗΤΑ/ΤΜΗΜΑ	ΥΠΟΓΡΑΦΗ
1.	Γεώργιος Μπαρδής	Επίκουρος Καθηγητής	Georgios Bardis Digitally signed by Georgios Bardis Date: 2023.04.05 19:44:12 +03'00'
2.	Χρήστος Τρούσσας	Επίκουρος Καθηγητής	Christos Troussas Digitally signed by Christos Troussas Date: 2023.04.06 08:15:34 +03'00'
3.	Παναγιώτα Τσελέντη	Ε.ΔΙ.Π.	Panagiota Tselenti Digitally signed by Panagiota Tselenti Date: 2023.04.05 20:26:08 +03'00'

Ημερομηνία εξέτασης: 10 / 3 /2023

Δήλωση συγγραφέα

Ο κάτωθι υπογεγραμμένος Τάτσης Νικόλαος του Παναγιώτη, με αριθμό μητρώου 151050 φοιτητής του Πανεπιστημίου Δυτικής Αττικής της Σχολής Μηχανικών του Τμήματος Μηχανικών Πληροφορικής και Υπολογιστών, δηλώνω υπεύθυνα ότι:

«Είμαι συγγραφέας αυτής της διπλωματικής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφεί από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος.

Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του πτυχίου μου».

Ο Δηλών

(Υπογραφή)

Πίνακας Περιεχομένων

Πίνακας Εικόνων

1. Εισαγωγή

Στόχος της παρούσας εργασίας είναι ο σχεδιασμός και η υλοποίηση ενός παιχνιδιού με εκπαιδευτικό χαρακτήρα το οποίο προσφέρει στον χρήστη μια καλύτερη εμπειρία εκμάθησης, καθώς ταυτόχρονα προσπαθεί να πετύχει μια πιο ευχάριστη απασχόληση, αξιοποιώντας τα διάφορα συστήματά του για να κάνει την εμπειρία ελκυστική. Πριν από κάποια χρόνια η εκμάθηση των αγγλικών γινόταν αποκλειστικά μέσω βιβλίων και εντατικών δια ζώσης μαθημάτων. Τότε τα παιχνίδια που κάλυπταν τέτοιες ανάγκες δεν ήταν ανεπτυγμένα με το σκεπτικό της διασκέδασης σε αρμονία με την μάθηση, πράγμα το οποίο φανερώνεται από την απλοϊκή τους κατασκευή και τον τρόπο αλληλεπίδρασης με τον χρήστη. Ως απάντηση σε αυτό το είδος παιχνιδιών δημιουργήθηκε η παρούσα εργασία για να αποδείξει ότι ένα εκπαιδευτικό παιχνίδι δεν χρειάζεται να "θυσιάζει" την διασκέδαση για μάθηση. Ο λόγος που επιλέχτηκε αυτό το είδος παιχνιδιού ως εκπαιδευτικό είναι το γεγονός ότι οι χρήστες, που βρίσκονται στην ηλικιακή προ εφηβείας ομάδα όπου δεν έχουν καθόλου ή πολύ περιορισμένη γνώση της αγγλικής γλώσσας, θα προτιμήσουν να αλληλεπιδράσουν με ένα παιχνίδι περισσότερο από ένα βιβλίο με το ίδιο πληροφοριακό περιεχόμενο. Αξιοποιώντας την μηχανή δημιουργίας παιχνιδιών Unity για την δημιουργία της εργασίας η οποία προσφέρει πολλά θετικά στοιχεία σε αυτή όπως είναι η καλύτερη απόδοση κατά την περίοδο της λειτουργίας της ή ο σχεδιασμός πιο πολύπλοκων συστημάτων τα οποία χρησιμοποιούνται για την καλύτερη εμπειρία του χρήστη.

Με το σκεπτικό αυτό σαν βάση, πάρθηκε η επιλογή να σχεδιαστεί η εργασία ως ένα παιχνίδι λαβυρίνθου από το οποίο ο παίχτης πρέπει να δραπετεύσει, ενσωματώνοντας παζλ ερωτήσεων γνώσης και ερωτήσεων σε διαφορετικές μορφές αναπαράστασης και απάντησης προκειμένου να διατηρηθεί το ενδιαφέρον των νεαρών χρηστών. Προκειμένου να μην θεωρηθεί το εκπαιδευτικό κομμάτι του παιχνιδιού ως δευτερεύον, επιλέχτηκε να ενσωματωθεί το σύστημα των ερωτήσεων με τον τρόπο ανάκτησης των βασικών πόρων τού παίχτη "δένοντας" τα δύο συστήματα. Επίσης, για να μην καταντήσει παρασιτική αυτή η ένωση συστημάτων λόγω έλλειψης ερωτήσεων, τοποθετήθηκαν στον κόσμο του παιχνιδιού κατάλληλα παζλ τα οποία ανταμείβουν τον παίχτη για την σωστή ολοκλήρωσή τους. Για να αυξηθεί η χρησιμότητα των παζλ αυτών αποφασίστηκε να γίνεται έλεγχος του πόρου που ο παίχτης έχει την μεγαλύτερη έλλειψη κατά την ολοκλήρωση του παζλ για να τον ανταμείψει με τον καλύτερο δυνατό τρόπο, δίνοντας γνώση ότι η επίλυση των παζλ δεν θα δίνει την αίσθηση της μη επιβράβευσης που ήταν μία δυσκολία που παρατηρήθηκε σε αρχικά στάδια της δημιουργίας. Επίσης πάρθηκε απόφαση να πρέπει να ενσωματώνει τα παιχνίδια που βρίσκονται υλοποιημένα σε διαφορετικές σκηνές στην κύρια σκηνή, προκειμένου να δημιουργηθεί συνεκτικότητα μεταξύ των διαφορετικών σκηνών του παιχνιδιού, προσφέροντας μια πιο καλή ολοκληρωμένη εικόνα λειτουργικότητας.

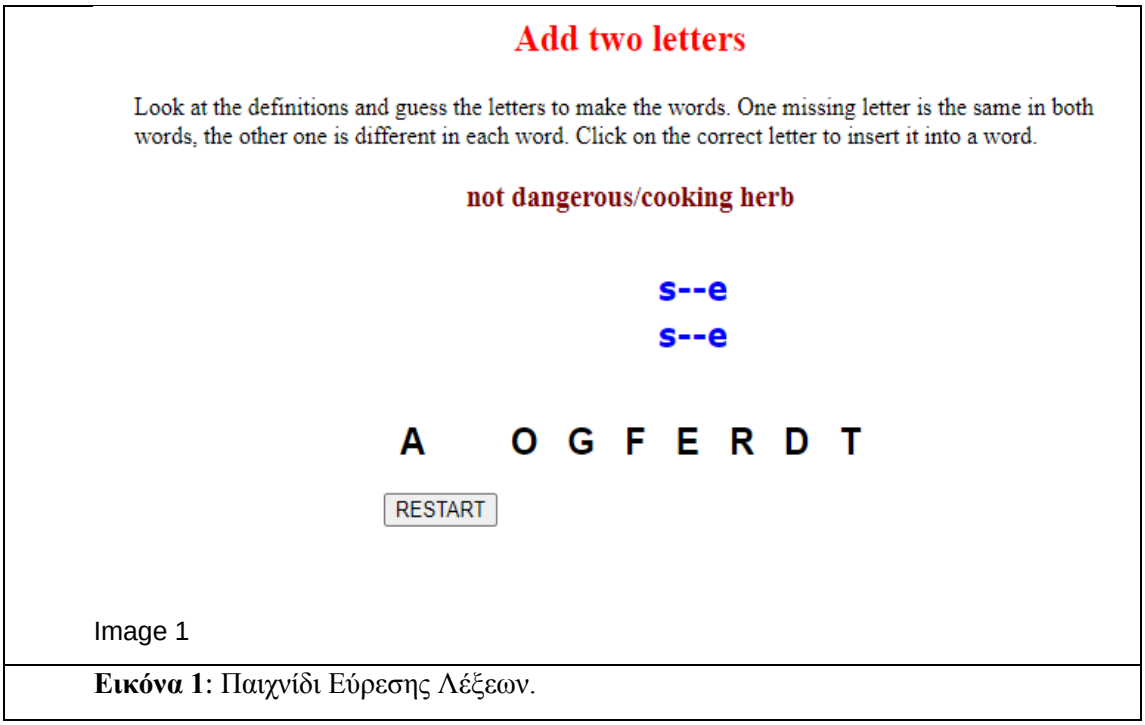
Συμπεράσματα που προέκυψαν κατά την διάρκεια της εκπόνησης της εργασίας αυτής είναι το γεγονός ότι δημιουργώντας ένα τέτοιο παιχνίδι απαιτείται η εξισορρόπηση μεταξύ των δύο βασικών στοιχείων του που είναι η διασκέδαση και η μάθηση σε τέτοιο βαθμό ώστε να μην υπάρχει υπερκάλυψη του ενός από το άλλο. Επίσης, προκύπτουν καλύτερα αποτελέσματα στο τελικό "προϊόν" όταν ο δημιουργός συνεχώς αναθεωρεί τις ιδέες που έχει για το παιχνίδι και δεν φοβάται να το αλλάξει προς το καλύτερο, φυσικά έχοντας στο μυαλό του πάντα πρώτα την διασκέδαση ως καθοριστικό παράγοντα αποφάσεων.

Ξεκαθαρίζοντας τα παραπάνω η παρούσα εργασία είναι οργανωμένη ως εξής: Το παρόν πρώτο κεφάλαιο αποτελεί μία μικρή εισαγωγή στο στόχο της εργασίας, στο σκεπτικό της επιλογής του και στο σχετικό υπόβαθρο. Αναφέρεται και αιτιολογείται η επιλογή πλατφόρμας υλοποίησης και παρουσιάζεται μία σύνοψη της εξέλιξης και των συμπερασμάτων. Στο δεύτερο κεφάλαιο παρουσιάζονται σχετικές προσεγγίσεις εκπαιδευτικών λογισμικών με στόχο την εκμάθηση αγγλικών και επισημαίνονται στοιχεία τους που η παρούσα εργασία επιχειρεί να βελτιώσει. Στο ίδιο κεφάλαιο παρουσιάζεται το περιβάλλον και η σχετική πλατφόρμα υλοποίησης και το σκεπτικό επιλογής τους. Στο τρίτο κεφάλαιο περιγράφεται αναλυτικά ο λειτουργικός σχεδιασμός της εφαρμογής και οπτικοποιείται μέσω διαγραμμάτων ροής. Στο τέταρτο κεφάλαιο περιγράφεται ο τεχνικός σχεδιασμός και η υλοποίηση της εφαρμογής και επισημαίνονται τα κυριότερα σημεία. Στο πέμπτο κεφάλαιο παρουσιάζεται μία σύνοψη των συμπερασμάτων που εξάγονται από την παρούσα εργασία σε συνάρτηση με την πορεία του σχεδιασμού και της υλοποίησης καθώς και της δοκιμαστικής χρήσης της από χρήστες που εμπίπτουν στην ηλικιακή ομάδα στην οποία απευθύνεται.

2. Υπόβαθρο

2.1. Σχετικές Εφαρμογές

Όπως είναι ευρέως γνωστό υπάρχει πληθώρα παιχνιδιών διαθέσιμα στο διαδίκτυο τα οποία μπορεί κάποιος να κατεβάσει δωρεάν ή επί πληρωμή. Βλέποντας όμως τον τρόπο με τον οποίο αυτά τα παιχνίδια χειρίζονται την παρουσίαση και δόμηση των συστημάτων τους και της πληροφορίας που θέλουν να περάσουν, παρατηρήθηκε μία απλοϊκότητα στον τρόπο δημιουργίας τους. Ειδικότερα, μετά από έρευνα σε διάφορα είδη δωρεάν παιχνιδιών με εκπαιδευτικό χαρακτήρα για την παιδική ηλικία, παρατηρήθηκε ότι το παιχνίδι έχει μόνο ένα σύστημα το οποίο χρησιμοποιεί ως βάση και ελάχιστη, αρκετές φορές καθόλου[16]([**Εικόνα 1**]), κίνηση στους χαρακτήρες που έχει για να μπορέσει να κρατήσει την προσοχή των παιδιών σε αυτό.



Εικόνα 1: Παιχνίδι Εύρεσης Λέξεων.

Σε πολλές περιπτώσεις το εκάστοτε παιχνίδι θα επιχειρήσει να ενσωματώσει φωνητικά στοιχεία προκειμένου να ανεβάσει την ποιότητά του. Παρόλα αυτά ο τρόπος παρουσίασής του παραμένει ελλιπής ή υπερβολικά απλοϊκός[15]([**Εικόνα 2**]).

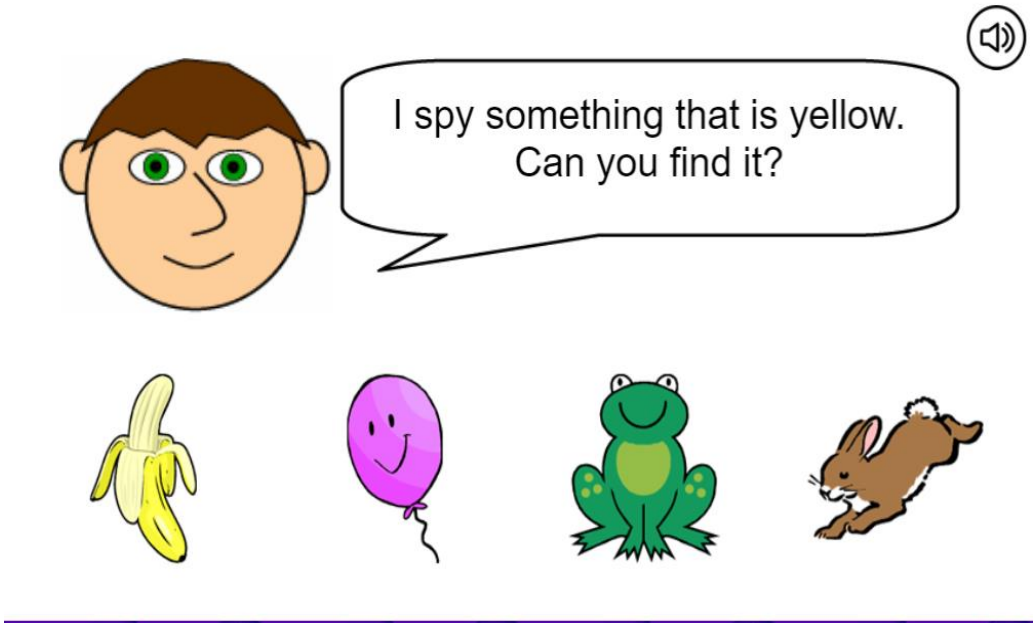


Image 2

Εικόνα 2: Παιχνίδι Κατανόησης Χρωμάτων.

Επίσης, συχνό φαινόμενο σε αυτά τα παιχνίδια είναι η κάλυψη μόνο ενός τομέα μάθησης(π.χ. εκμάθηση conditionals[14]([**Εικόνα 3**])). Συχνά αγνοώντας άλλους τομείς που θα βοηθούσαν στην καλύτερη εκμάθηση της γλώσσας(π.χ. word-spelling ή γραμματική εξάσκηση), για τους οποίους θα έπρεπε να αλλάξουν παιχνίδι προκειμένου να εξασκηθούν σε αυτούς.

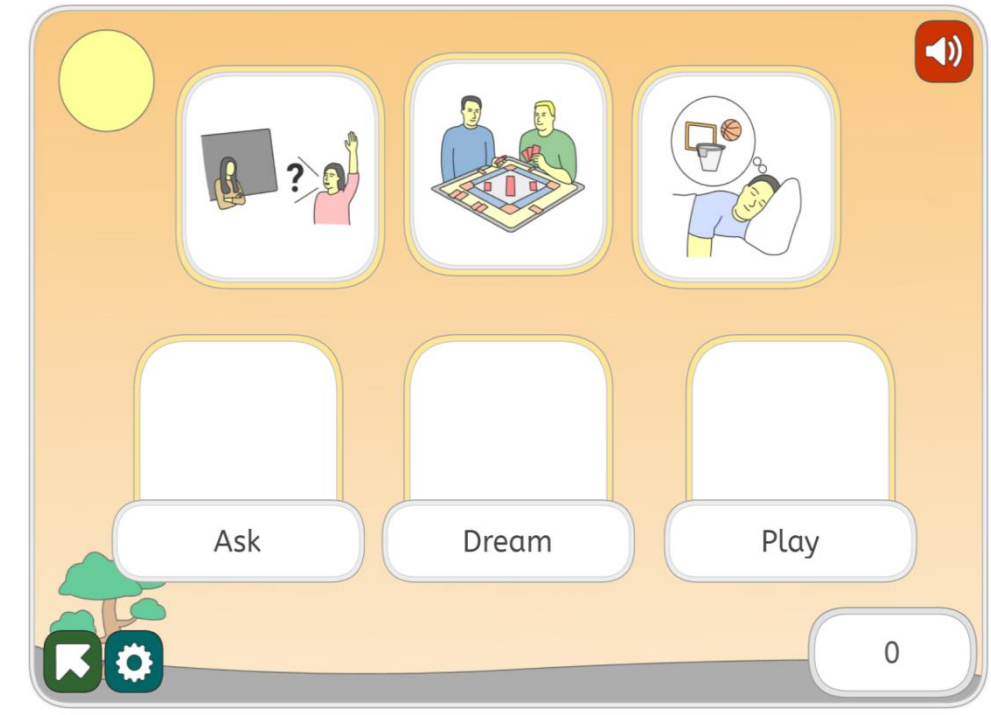


Image 3

Εικόνα 3: Παιχνίδι Εκμάθησης Ενεργειών.

Για παράδειγμα, σε ένα παιχνίδι σταυρολέξου [13] όπου πρέπει να τοποθετηθούν τα γράμματα που δίνονται στην σωστή σειρά για να σχηματιστεί μία λέξη θα παρατηρούνταν ότι όλο το παιχνίδι θα ακολουθούσε αυτήν την λογική χωρίς να αλλάζει ή να προσφέρει άλλους διαδραστικούς τρόπους για να "σπάσει" την μονοτονία (αν αυτή υπάρχει, εφόσον τέτοιου είδους παιχνίδια δεν είναι μεγάλα σε μέγεθος χρόνου που καταναλώνουν. [**Εικόνα 4**]).

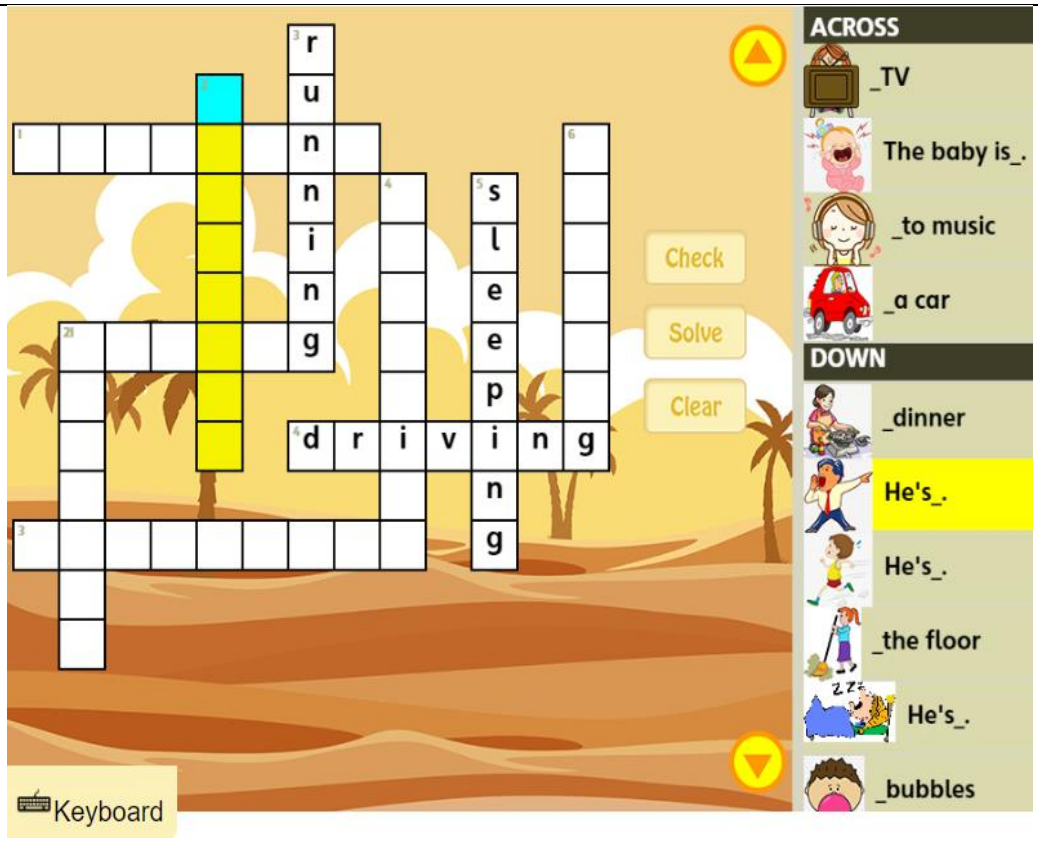


Image 4

Εικόνα 4: Παιχνίδι Σταυρολέξου.

Πολύ συχνά για να διατηρήσουν το ενδιαφέρον των νεαρών χρηστών τους τα παιχνίδια αυτά, έχουν κίνηση στοιχείων της οθόνης κατά την διάρκεια αλληλεπίδρασης των χρηστών με αυτά, αλλά ακόμα και αυτό πολλές φορές δεν είναι αρκετό για να διατηρήσει το ενδιαφέρον ή ακόμα καλύτερα να κάνει τους χρήστες του να θέλουν να δοκιμάσουν να παίξουν ξανά, εφόσον έχοντας παίξει μία φορά γίνεται φανερό πόσο απλό είναι[17]([**Εικόνα 5**]).


Actions, Nouns
Treasure
0/17
2
3
4
5
2 Steps Back
6
7
2 Steps Forward
8
9
10
Start Again
11
12
13
14
2 Steps Back
15
16
17
Start Again
18
19
20
21
22
2 Steps Forward
23
24
25
26
Start Again
27
28
3 Steps Back
29

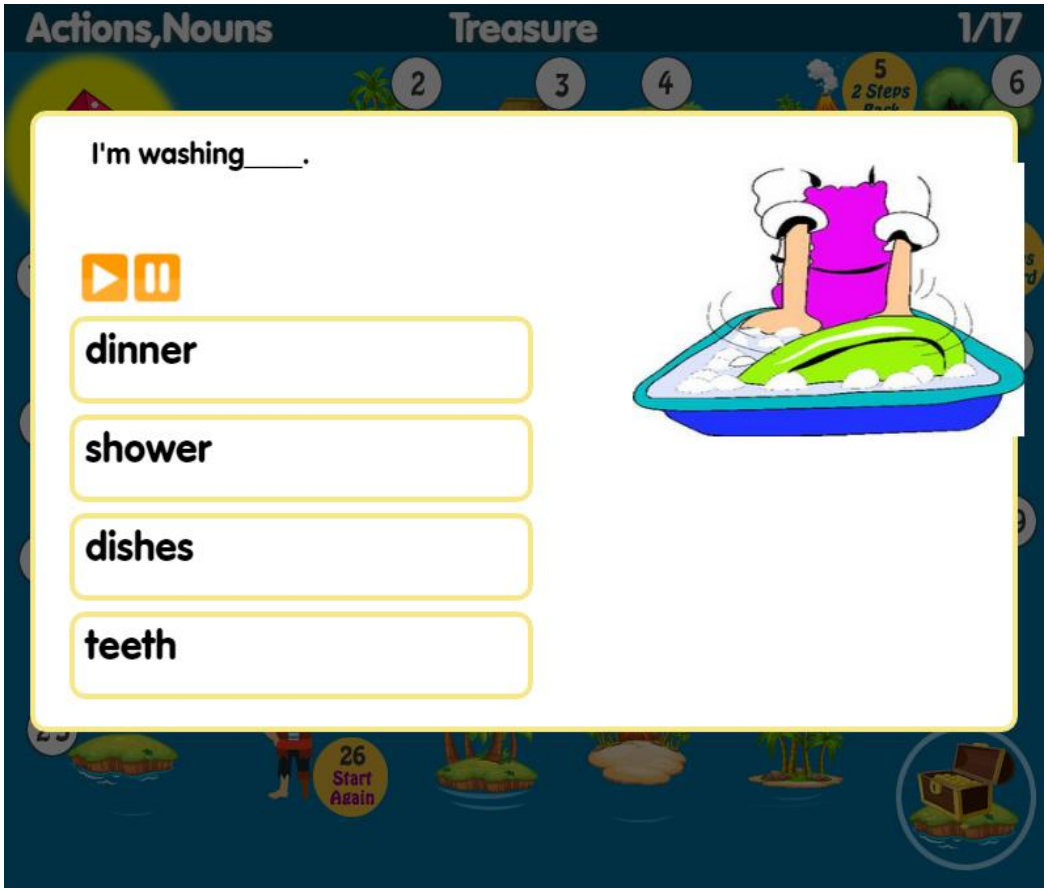

Actions, Nouns
Treasure
1/17
I'm washing____.
dinner
shower
dishes
teeth
26
Start Again

Image 5

Εικόνα 5: Παιχνίδι Συμπλήρωσης Προτάσεων.

Με αναφορά αυτού του είδους τα παιχνίδια [18] δόθηκε αφορμή για την δημιουργία του παιχνιδιού στο οποίο βασίζεται αυτή η διπλωματική εργασία, για να δείξει:

1. Ότι ένα εκπαιδευτικό παιχνίδι δεν είναι αναγκαστικό να ακολουθήσει την ίδια μεθοδολογία κατασκευής στον τρόπο με τον οποίο επιλέγει να εκπαιδεύσει τον χρήστη για την πιο αποτελεσματική εκμάθηση μίας ξένης γλώσσας.
2. Μπορεί να χρησιμοποιήσει περισσότερα και σύνθετα συστήματα για να εκτελέσει την διαδικασία μάθησης και να την κάνει πιο ευχάριστη και ελκυστική για τον χρήστη και ότι τελικά,
3. Το παιχνίδι βοηθάει τον χρήστη με ενδιαφέροντα συστήματα και προσθέτει παραπάνω τρόπους οι οποίοι τον ενθαρρύνουν να δοκιμάσει να ξαναπαίξει το παιχνίδι με διαφορετικό τρόπο, δίνοντάς του απόλυτο έλεγχο για αποφάσεις στον κόσμο του παιχνιδιού.

Τα παραπάνω στοιχεία παρατηρούνται σε παιχνίδια τα οποία κατασκευάζονται από μεγάλες εταιρίες αξιοποιώντας μία ομάδα προγραμματιστών για την δημιουργία τους και είναι επί πληρωμή(τα παιχνίδια αυτά ονομάζονται AAA games ή αλλιώς triple A games).

2.2. Περιβάλλον Υλοποίησης

2.2.1. Συσκευές και Λειτουργικό

Το παιχνίδι σχεδιάστηκε έχοντας ως κύρια πλατφόρμα λειτουργίας τους υπολογιστές (pc). Αυτό οφείλεται σε αρκετούς λόγους που επηρεάζουν όχι μόνο αυτό το παιχνίδι αλλά και γενικότερα παιχνίδια από μεγάλες εταιρίες (γνωστά ως AAA games). Συγκεκριμένα για αυτό το παιχνίδι ευνοούν τα παρακάτω:

1. Το παιχνίδι είναι εύκολα διανεμημένο μέσω ειδικών ψηφιακών πλατφορμών(π.χ. Steam) που βοηθάει στην εξάλειψη της ανάγκης δημιουργίας φυσικού αντίγραφου προκειμένου να είναι διαθέσιμο στους χρήστες.

2. Προσφέρει αυξημένη απόδοση, εφόσον τα παιχνίδια διαμορφώνουν την λειτουργικότητά τους με βάση το υλικό του υπολογιστή κάθε χρήστη για να παρέχουν πιο ομαλή και ευχάριστη εμπειρία.

3. Η πλατφόρμα είναι πιο δημοφιλής από όλες τις άλλες (σύμφωνα με την πηγή [45]) πράγμα που βοηθάει στην έκθεση του παιχνιδιού σε περισσότερους ανθρώπους.

Λόγοι που απορρίφθηκε η ιδέα της επέκτασης σε κινητά τηλέφωνα είναι οι εξής:

Ως προς το πρακτικό κομμάτι, παρά την πιθανή διευκόλυνση του χρήστη και την αύξηση των εσόδων που προσφέρει στον δημιουργό του παιχνιδιού, το κοινό στο οποίο προορίζεται το παιχνίδι δεν βρίσκεται σε ηλικία που επιτρέπεται να έχει κινητό τηλέφωνο από τους γονείς του.

Ως προς το ηθικό κομμάτι, το παιχνίδι χάνει την εμπιστοσύνη των γονέων λόγω της δυσφήμισης γενικά των παιχνιδιών σε κινητά εξαιτίας των διαβόητων πρακτικών που χρησιμοποιούν για να αναγκάσουν τον καταναλωτή να πληρώσει χρήματα(πάνω σε αντικείμενα που προσφέρει το παιχνίδι, χρησιμοποιώντας την τεχνική γνωστή ως FOMO or Fear of Missing out(περισσότερα στην πηγή [47]) η οποία αναγκάζει τους καταναλωτές να πληρώσουν προκειμένου να μην βρεθούν χαμένοι αργότερα και αισθανθούν άσχημα με τον εαυτό τους).

Ως προς το τεχνικό κομμάτι, ένα παιχνίδι που σχεδιάζεται για ένα κινητό τηλέφωνο θα παρέχει λιγότερες δυνατότητες στον τομέα των γραφικών(εφόσον πρέπει να γίνουν οι απαραίτητες απλοποιήσεις προκειμένου να υπάρχει αρκετός χώρος μνήμης στο κινητό, χωρίς να υπάρχει υπερθέρμανση της μπαταρίας ή μεγάλος χρόνος αναμονής για την εμφάνιση των γραφικών στοιχείων του παιχνιδιού). Επίσης, οι ενέργειες που μπορεί να εκτελέσει ο χρήστης είναι περιορισμένες στο "πάτημα" της οθόνης του κινητού, οπότε και το παιχνίδι θα πρέπει να απλοποιηθεί στον τομέα των γραφικών αλλά και της πολυπλοκότητας πράγμα που είναι αντιφατικό με τις αρχές της διπλωματικής εργασίας που ειπώθηκαν παραπάνω(Κεφάλαιο 2.1. Σχετικές Εφαρμογές).

2.2.2. Πλατφόρμα

Η μηχανή ανάπτυξης παιχνιδιών που χρησιμοποιήθηκε είναι η Unity, η οποία επιλέχτηκε για τους εξής λόγους:

1. Είναι δωρεάν ανοιχτό λογισμικό το οποίο πολλές φορές προτιμάται έναντι άλλων επί πληρωμή λογισμικών όπως είναι η μηχανή Unreal Engine η οποία απαιτεί συνεχή συνδρομή.
2. Επειδή το λογισμικό είναι ανοιχτού τύπου, υπάρχουν περισσότερες πληροφορίες και εκπαιδευτικό υλικό στο διαδίκτυο, βοηθώντας έτσι στην ευκολότερη εκμάθηση της χρήσης του λογισμικού για την δημιουργία παιχνιδιών.
3. Προσφέρει πολλές δυνατότητες για ένα παιχνίδι, όπως είναι η επιλογή ray tracing η οποία ενδυναμώνει τον φωτισμό προκειμένου να προσθέσει ρεαλισμό σε ένα παιχνίδι(π.χ. ένα σώμα νερού (λίμνη,θάλασσα) μπορεί να αντικατοπτρίζει στοιχεία όπως και στην πραγματική ζωή).
4. Υπάρχουν πρόσθετα(plugins) στοιχεία τα οποία μπορούν να χρησιμοποιηθούν για την διευκόλυνση της δημιουργίας συστημάτων του παιχνιδιού, χωρίς την απαίτηση εκτενούς χειροκίνητης κωδικοποίησης(scripting code).
5. Το λογισμικό δέχεται συνεχώς νέες ανανεώσεις και αναβαθμίσεις προκειμένου να βοηθήσει τους χρήστες του να δημιουργήσουν καλύτερης ποιότητας παιχνίδια και προσθέτει υποστήριξη καινούριων πλατφορμών(περισσότερα στην πηγή [48]).

Για την δημιουργία ενός παιχνιδιού είναι σημαντικό να γίνουν κατανοητές κάποιες έννοιες που θα χρησιμοποιηθούν παρακάτω.

1. **Σκηνή:** Η σκηνή του παιχνιδιού είναι ένα σύνολο στοιχείων τα οποία δημιουργούν τον κόσμο του παιχνιδιού και βρίσκονται σε λειτουργία κατά την έναρξη του παιχνιδιού. Μια πρακτική καλύτερου σχεδιασμού είναι ο χωρισμός του παιχνιδιού σε πολλαπλές σκηνές προκειμένου να βελτιωθεί η απόδοσή του και να είναι ευκολότερη η συντήρησή του.(ένα παιχνίδι μπορεί να έχει πολλές σκηνές σε ταυτόχρονη λειτουργία αλλά μόνο στην πρώτη ιεραρχικά σκηνή γίνεται ο εστιασμός από το παιχνίδι.)
2. **Στοιχεία:** Ως στοιχεία του παιχνιδιού θεωρούνται τα αντικείμενα τα οποία βρίσκονται μέσα στην σκηνή. Τα στοιχεία αυτά μπορεί να είναι αντικείμενα τα οποία βλέπει ο παίχτης(π.χ. χαρακτήρες,πόρτες,αντίπαλοι,παγίδες) αλλά και πράγματα τα οποία δεν βλέπει(π.χ. το αντικείμενο της κάμερας, το αντικείμενο της αναπαραγωγής μουσικής) και ορισμένοι μηχανισμοί με όνομα triggers οι οποίοι εκτελούν κάποια λειτουργία που επιθυμεί ο προγραμματιστής(π.χ. όταν ο παίχτης "πατήσει" ένα μοχλό τότε ανοίγει μία πόρτα).
3. **Scripts:** Τα scripts είναι κομμάτια κώδικα τα οποία τοποθετούνται πάνω σε στοιχεία της σκηνής προκειμένου να εκτελεστούν ενέργειες. Ειδικότερα, για να μπορέσει ένας εχθρός να έχει την κατάλληλη συμπεριφορά προς τον παίχτη θα πρέπει να γραφεί κατάλληλος κώδικας της λογικής λειτουργίας του και να προστεθεί στο αντικείμενο του εχθρού για να λάβει δράση κατά την διάρκεια του παιχνιδιού.

3. Λειτουργικός Σχεδιασμός

3.1. Επεξήγηση της ροής του Παιχνιδιού

Το πρώτο πράγμα το οποίο θα δει ένας χρήστης όταν ανοίγει το παιχνίδι θα είναι το αρχικό μενού, το οποίο έχει σχεδιαστεί με απλοϊκό τρόπο λαμβάνοντας υπόψιν το κοινό στο οποίο απευθύνεται το παιχνίδι αλλά ταυτόχρονα θέτει το και ύφος του παιχνιδιού.Το μενού δίνει στον παίχτη την επιλογή της έναρξης του παιχνιδιού, ή τον τερματισμό του. Αφού ο παίχτης επιλέξει να ξεκινήσει το παιχνίδι, αρχίζει από το πρώτο επίπεδο και του δίνεται ο έλεγχος του χαρακτήρα του. Το πρώτο πράγμα που παρατηρεί είναι το UI του παιχνιδιού. Αποφασίστηκε η μπάρα ένδειξης των πόρων του παίχτη να δημιουργηθεί με διαφορετικό εμφανισιακό τρόπο από αυτόν των εχθρών (έχοντας ως οδηγό την πηγή [20] και προσαρμόζοντας ανάλογα). Το κουμπί μουσικής με το οποίο ο παίχτης μπορεί να ελέγξει την μουσική του παιχνιδιού (σχεδιάστηκε σύμφωνα με την πηγή [21]).

Πέρα από τα παραπάνω στοιχεία, ο παίχτης θα ακολουθήσει τα βοηθητικά μηνύματα που βρίσκονται μέσα στον κόσμο του παιχνιδιού προκειμένου να μάθει πώς να χειρίζεται τον χαρακτήρα του. Κάθε φορά που ο παίχτης επιχειρεί να αντιδράσει με ένα αντικείμενο θα λάβει κατάλληλο μήνυμα για την ενέργεια που εκτελεί(χρησιμοποιώντας την πηγή [22] και εμβαθύνοντας στο σύστημα αυτό αλλάχτηκε η εμφάνιση αριθμών σε εμφάνιση μηνυμάτων), έχοντας έτσι μια καθαρή εικόνα της κατάστασης στην οποία βρίσκεται. Πολεμώντας τον πρώτο του εχθρό(ο οποίος είναι απλός σε λειτουργία) θα παρατηρήσει ότι κάθε φορά που επιτίθεται(τρόπος επίθεσης με βάση την πηγή [28]) ο πόρος της δύναμής του μειώνεται. Όταν η αντίστοιχη μπάρα φτάσει στο 0 δεν θα μπορεί να εκτελέσει άλλες επιθέσεις μέχρι να βρει τρόπο να επαναφέρει τους χαμένους αυτούς πόντους. Υπάρχουν τρεις τρόποι για να επαναφερθούν οι πόντοι δύναμης(και υγείας):

- Μέσω της εύρεσης ενός potion δύναμης το οποίο μπορεί να βρεθεί σε συγκεκριμένα σημεία του επιπέδου.

- Με τυχαίο τρόπο, κερδίζοντας εχθρούς(υλοποιημένο με βάση την πηγή [23], διευρύνοντας την λειτουργία του. [**Εικόνα 6**])

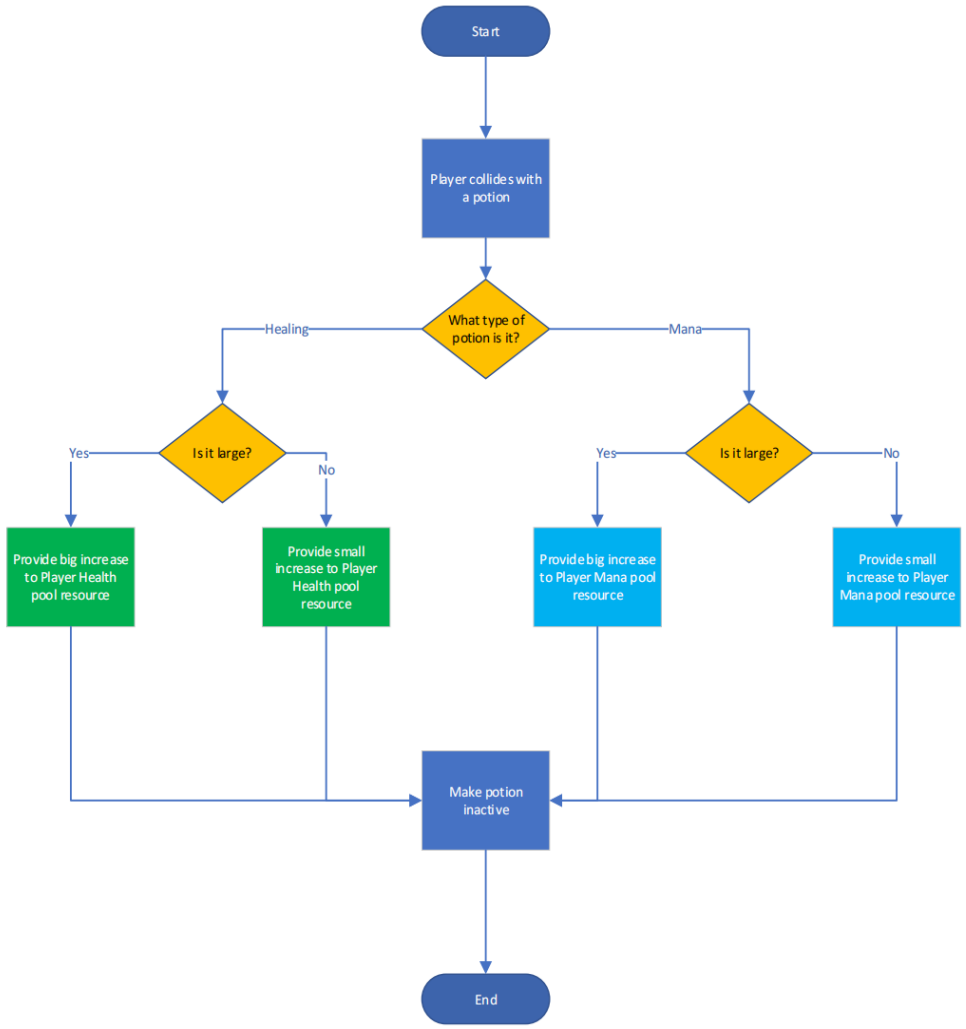


Image 6

Εικόνα 6: Η ροή λειτουργίας των διαφόρων potions του παιχνιδιού.

- Απαντώντας σε ερωτήσεις στα αγγλικά όπου για κάθε σωστή απάντηση επανέρχεται το 25% της δύναμής του και αποτελεί τον κύριο τρόπο αποκατάστασης αυτού του πόρου(παιχνίδι ερωτήσεων υλοποιημένο με βάση την πηγή [24]. **Εικόνα 7**). Αν επιλέξει να χρησιμοποιήσει το σύστημα στο οποίο απαντάει σε ερωτήσεις, τότε πρέπει να σιγουρευτεί ότι βρίσκεται μακριά από εχθρούς, εφόσον μπορεί να δέχεται επιθέσεις κατά την διάρκεια αλληλεπίδρασης με αυτό.

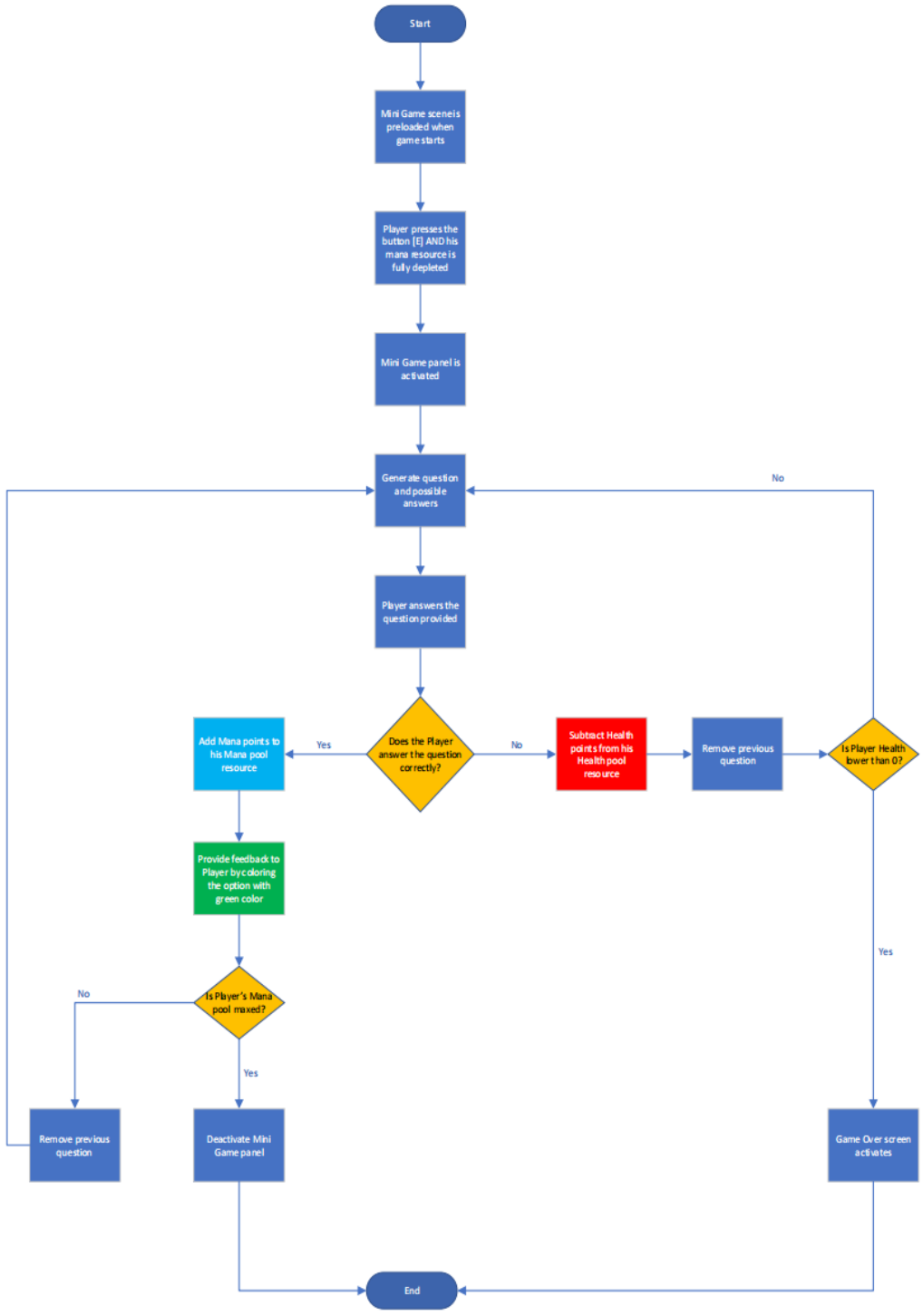


Image 7

Εικόνα 7: Η ροή λειτουργίας του παιχνιδιού πολλαπλών ερωτήσεων.

Κερδίζοντας τον πρώτο εχθρό ο παίχτης θα ανταμειφθεί με ένα κλειδί με το οποίο μπορεί να επιλέξει να ανοίξει μία πόρτα(χρησιμοποιήθηκαν οι πηγές [25], [26] και [27]. **Εικόνα 8**), ξεκλειδώνοντας νέες διαδρομές προκειμένου να φτάσει στο δεύτερο επίπεδο.

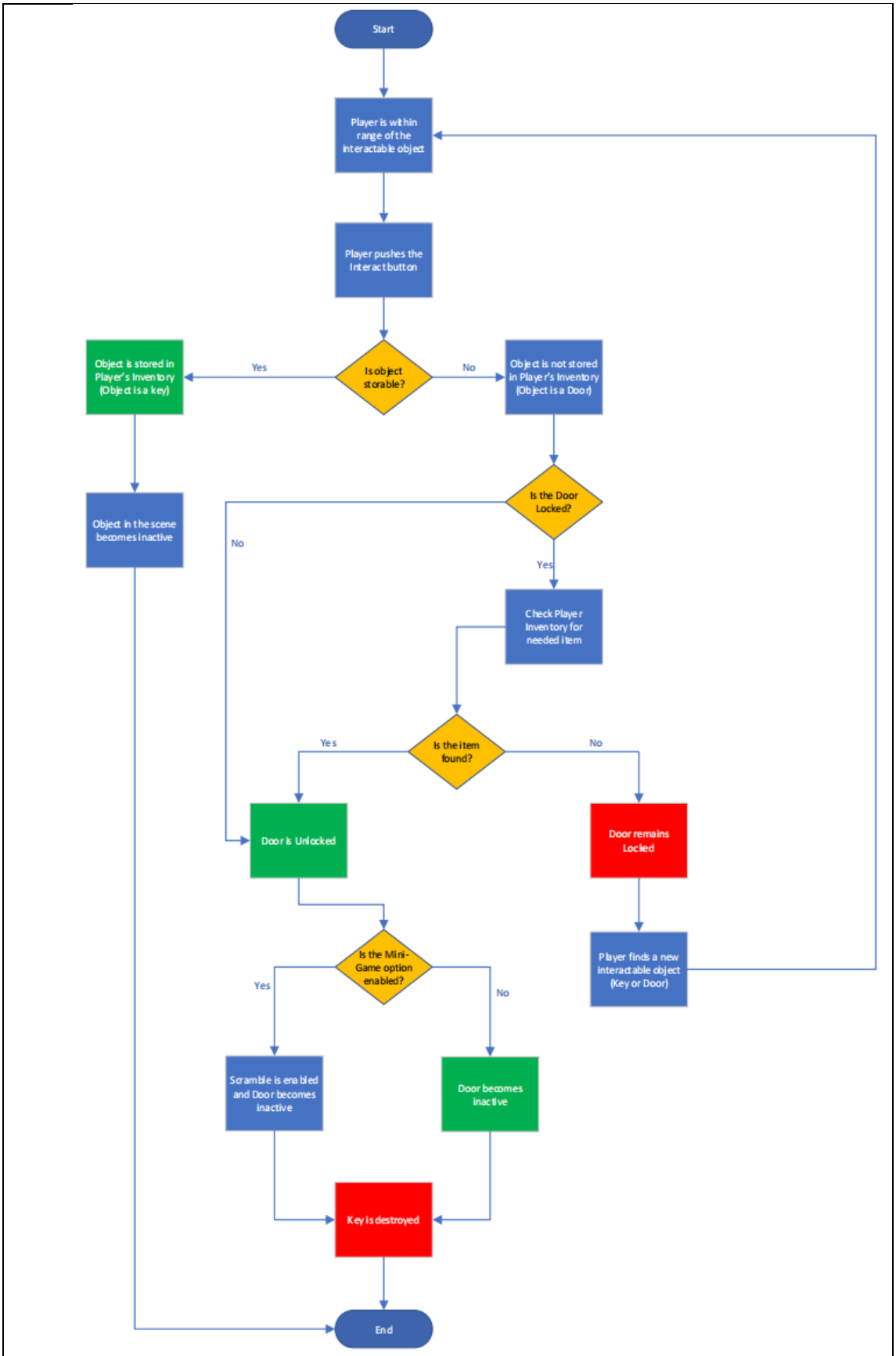


Image 8

Εικόνα 8: Η ροή λειτουργίας των Interactable objects του παιχνιδιού.

Κατά την διάρκεια εξερεύνησης του επιπέδου, ο παίχτης μπορεί να βρει παγίδες(πηγή [46]) στον δρόμο του, που μπορούν να του προκαλέσουν ζημιά στους πόντους υγείας του. Είτε παγίδες ανασυρόμενων καρφιών, που απαιτούν έγκαιρη αποφυγή(**Εικόνα 9 / Εικόνα 10**).

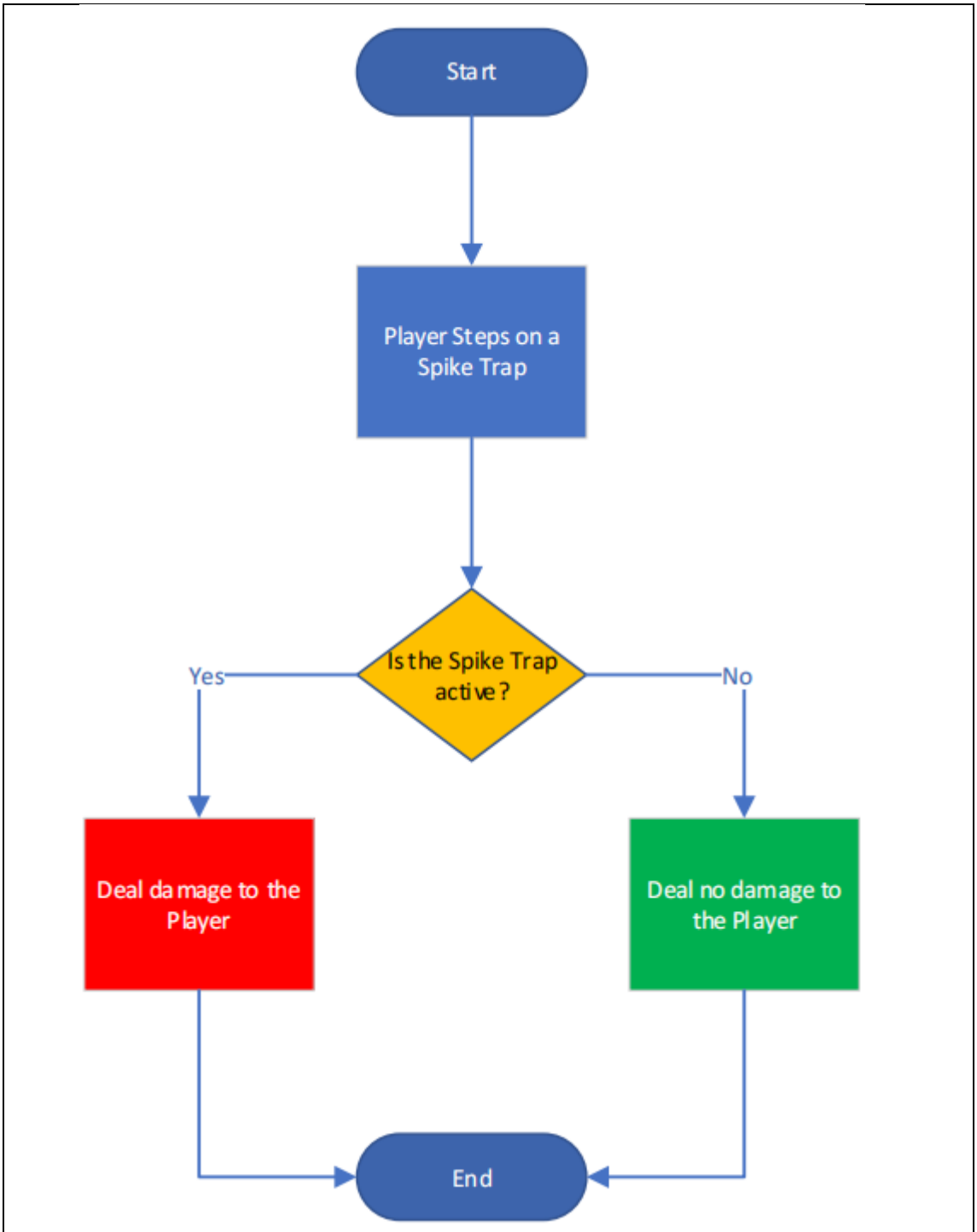


Image 9

Εικόνα 9: Η ροή λειτουργίας της παγίδας με καρφιά.

Image 10

Εικόνα 10: Η παγίδας με καρφιά στον κόσμο του παιχνιδιού.

Είτε παγίδες που εκτοξεύουν βέλη προκειμένου να τραυματίσουν τον παίχτη(**Εικόνα 11 / Εικόνα 12**).

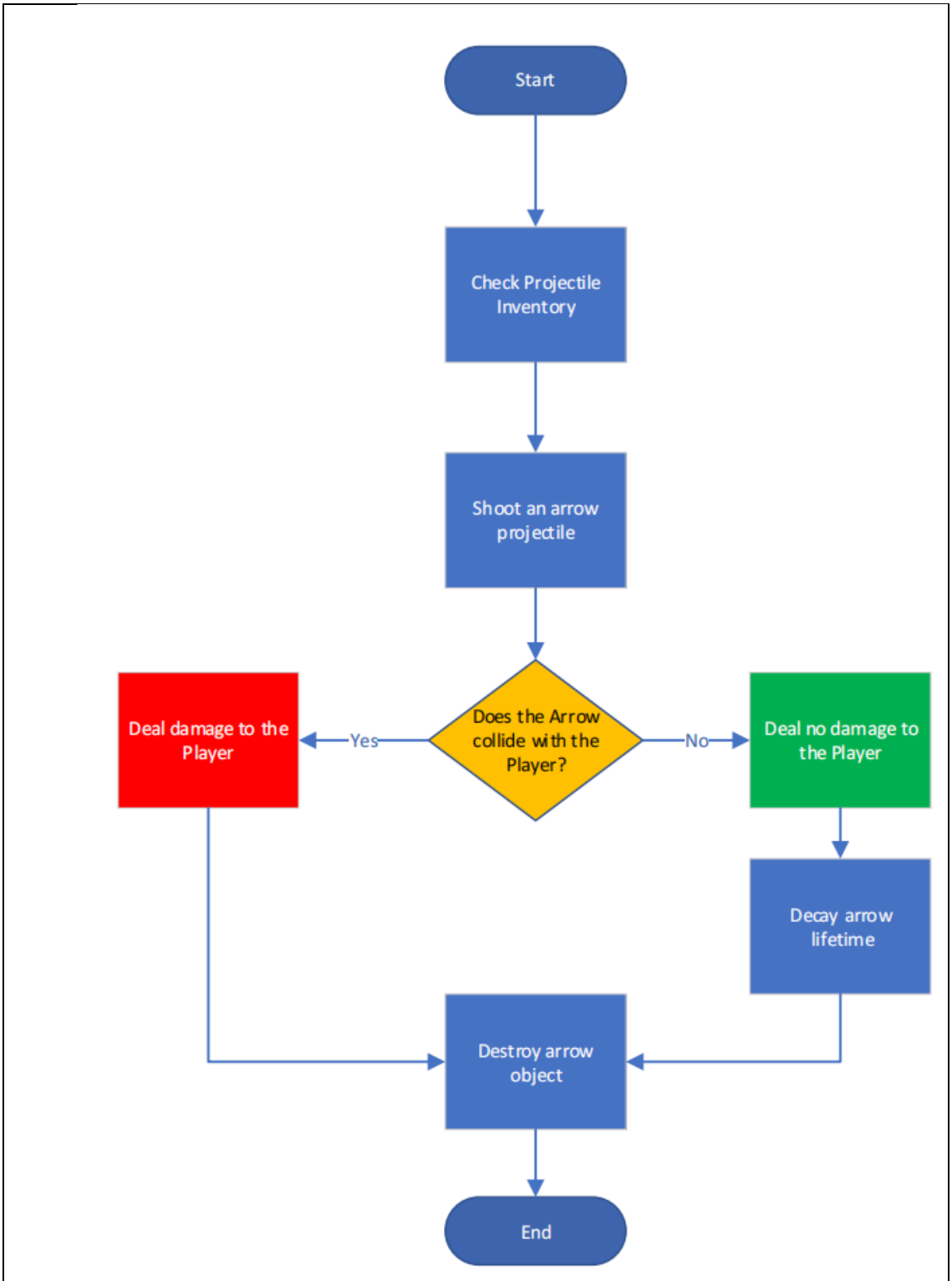


Image 11

Εικόνα 11: Η ροή λειτουργίας της παγίδας που εκτοξεύει βέλη.

Image 12

Εικόνα 12: Παγίδες εκτόξευσης βελών στον κόσμο του παιχνιδιού.

Επίσης ίσως συναντήσει και ξεχωριστά παζλ τα οποία μπορεί επιλεκτικά να λύσει προκειμένου να βοηθηθεί στην αποκατάσταση του προς στιγμήν χαμηλότερου πόρου του(**Εικόνα 13**).

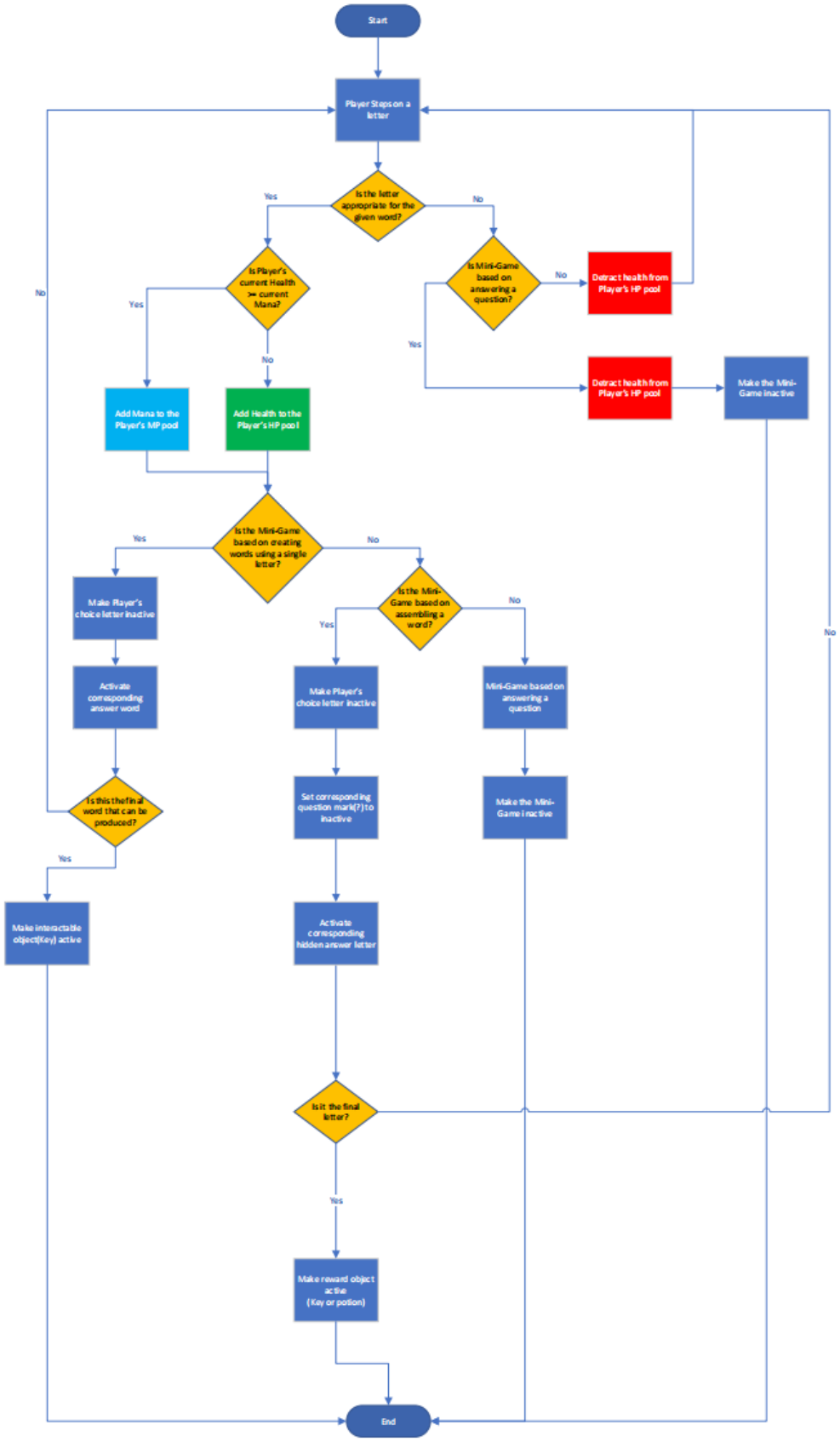


Image 13

Εικόνα 13: Η ροή λειτουργίας των παιχνιδιών Letter on the Ground.

Ο παίχτης πέρα από τους αρχικούς, θα βρει και δυνατότερους εχθρούς οι οποίοι προκαλούν περισσότερη ζημιά(σκελετοί με τσεκούρια δημιουργημένοι με βάση τις πηγές [29] , [30] και [31] καθώς και σκελετοί που ρίχνουν μπάλες φωτιάς με έμπνευση της πηγής [32] και απλοποίηση [**Εικόνα 14 / Εικόνα 15 / Εικόνα 16**]).

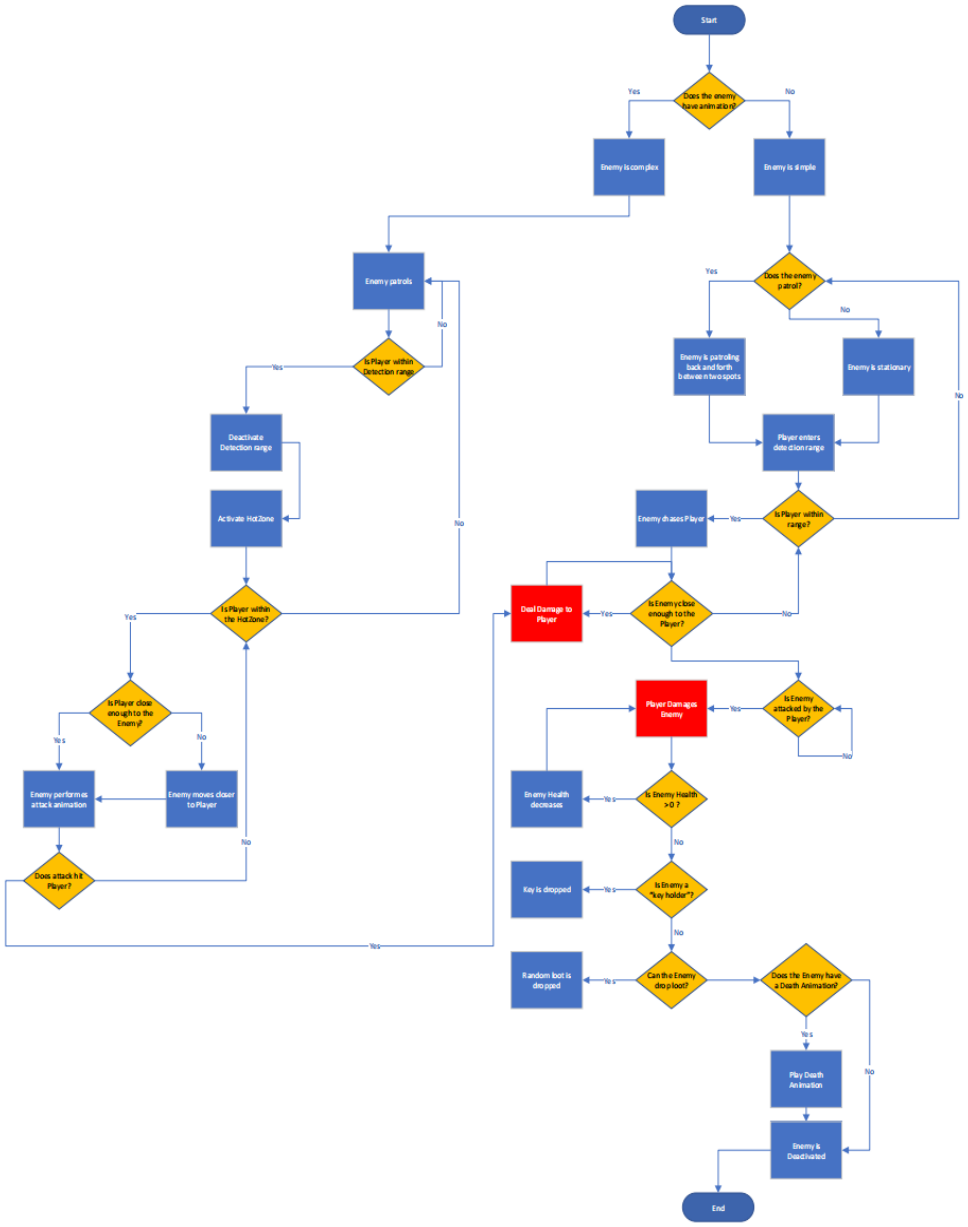


Image 14

Εικόνα 14: Η ροή της λειτουργίας των εχθρών του παιχνιδιού.

Image 15

Εικόνα 15: Μάχη με έναν εχθρό που ρίχνει μπάλες φωτιάς.

Image 16

Εικόνα 16: Μάχη με έναν εχθρό κοντινής απόστασης.

Προχωρώντας και εξερευνώντας με αυτό τον τρόπο μέσα στο πρώτο επίπεδο του λαβύρινθου(τα δύο επίπεδα δημιουργήθηκαν με βάση την πηγή [42]), ο παίχτης επιδιώκει να προχωρήσει στο επόμενο επίπεδο, προκειμένου να πολεμήσει τελικό αρχηγό . Μεταβαίνει, συνεπώς στο επόμενο επίπεδο(χρήση της πηγής [33],[39] και [40]), όπου για άλλη μια φορά πρέπει να καταφέρει να φτάσει στην "καρδιά" του λαβύρινθου και να ανοίξει την τελική πόρτα με χρυσό χρώμα προκειμένου να φτάσει στον τελικό αρχηγό. Η πόρτα αυτή όταν ξεκλειδωθεί θα παρουσιάσει στον παίχτη ένα τελικό παιχνίδι εύρεσης της λέξης(το παιχνίδι

κατασκευάστηκε με την χρήση των πηγών [34],[35] και [36]. **Εικόνα 17**). Λύνοντάς το ο παίχτης μπορεί να προχωρήσει στην αίθουσα του αρχηγού προκειμένου να αρχίσει η μάχη.

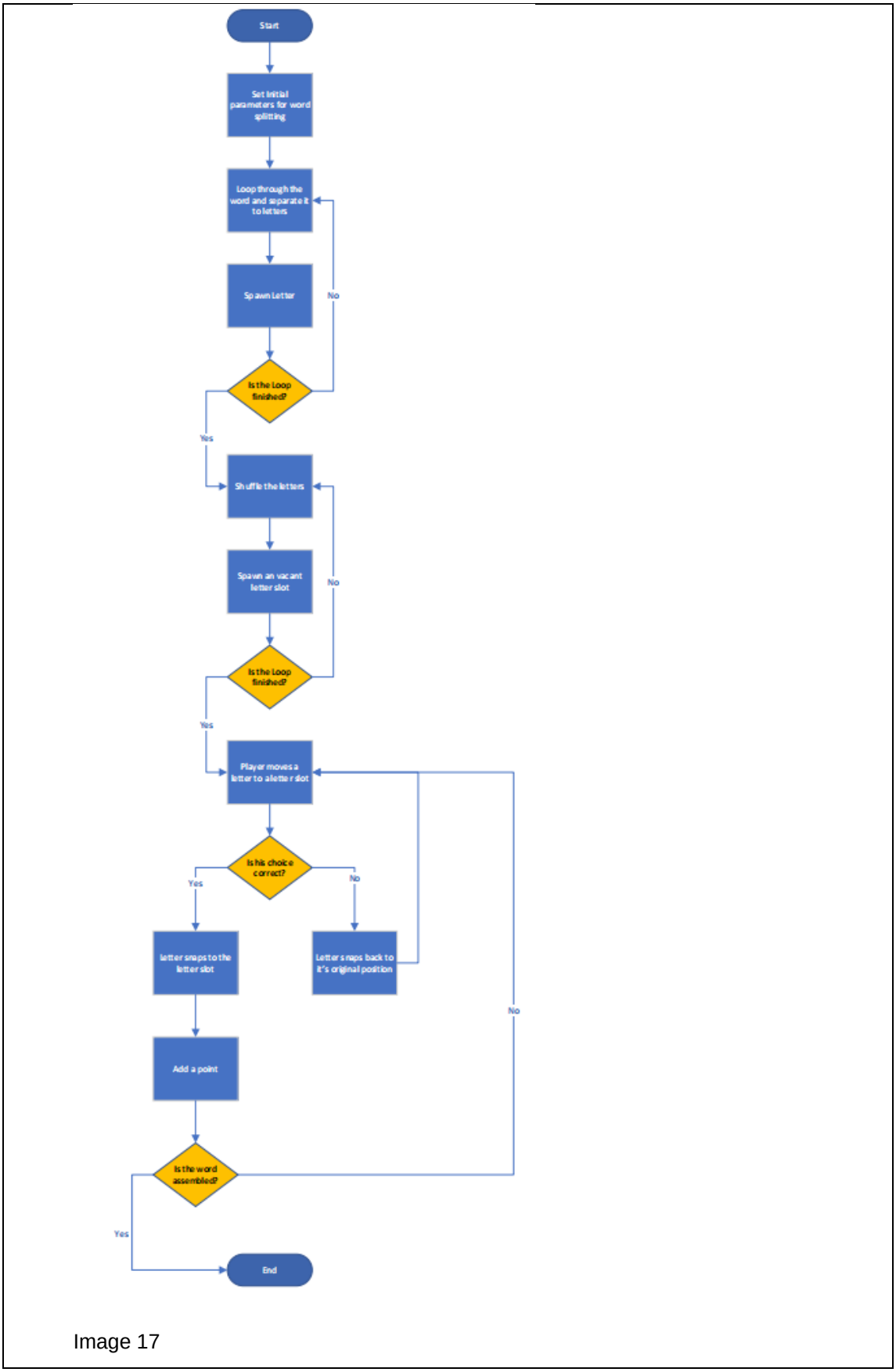


Image 17

Εικόνα 17: Η ροή λειτουργίας του FSM στο οποίο στηρίζεται το παιχνίδι Word Scramble.

Ο αρχηγός έχει 2 διαφορετικά στάδια τα οποία ενεργοποιούνται όταν φτάσει η υγεία του σε ένα συγκεκριμένο ποσοστό(50 % και 25 %). Τότε ενεργοποιείται μία ασπίδα η οποία μπλοκάρει τις επιθέσεις του παίχτη και ταυτόχρονα δημιουργεί έναν ειδικό εχθρό(ή δύο αν βρίσκεται στο 25 %) ο οποίος πρέπει να ηττηθεί για να καταστραφεί η ασπίδα(δημιουργία της ασπίδας στην πηγή [37]). Κατά την αρχή της μάχης με τον αρχηγό θα δημιουργηθούν τρεις εχθροί σε τυχαία σημεία της αίθουσας οι οποίοι μπορούν να ηττηθούν από τον παίχτη ή να καταστραφούν όταν ηττηθεί ο αρχηγός(για την δημιουργία της διαδικασίας μάχης με τον αρχηγό ακολουθήθηκε η πηγή [38] και πηγή [44] για την καταστροφή μετά το animation. **Εικόνα 18 / Εικόνα 19**). Όταν κερδηθεί ο αρχηγός το παιχνίδι θεωρείται ολοκληρωμένο.

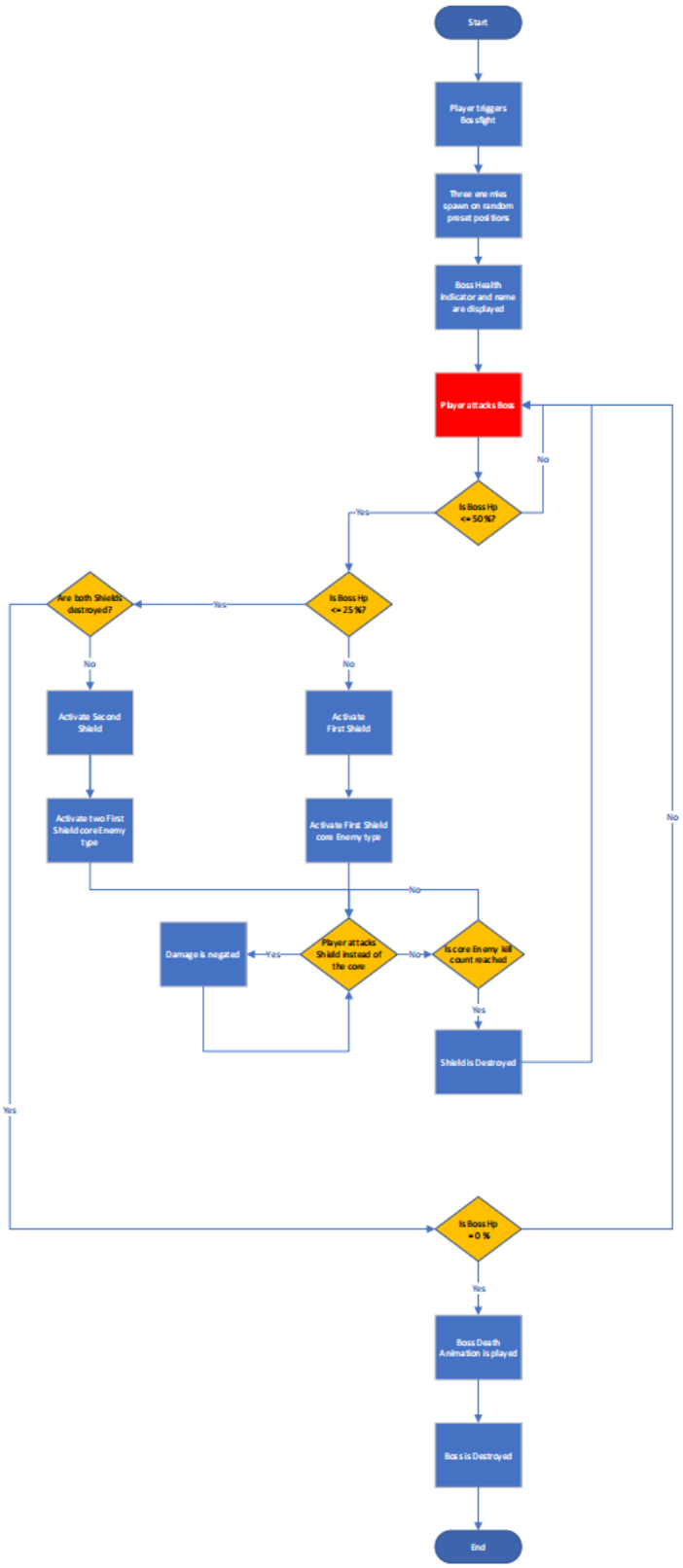


Image 18

Εικόνα 18: Η ροή λειτουργίας της τελικής μάχης με τον αρχηγό.

Image 19

Εικόνα 19: Η τελική μάχη με τον αρχηγό όταν ενεργοποιείται η ασπίδα.

Επιπλέον, κατά την διάρκεια του παιχνιδιού ο παίχτης μπορεί να αξιοποιήσει τα διάφορα μενού που προσφέρει το παιχνίδι για να αποθηκεύσει την κατάσταση του παιχνιδιού του(σύστημα Saving χρησιμοποιώντας την πηγή [41]), να σταματήσει την ροή του παιχνιδιού, να ξαναρχίσει από την αρχή το παιχνίδι ή να επαναφέρει μία προηγούμενη κατάσταση του παιχνιδιού ή να αλλάξει την ένταση της μουσικής μέσω του κύριου μενού(υλοποίηση με χρήση της πηγής [43], [**Εικόνα 20**]).

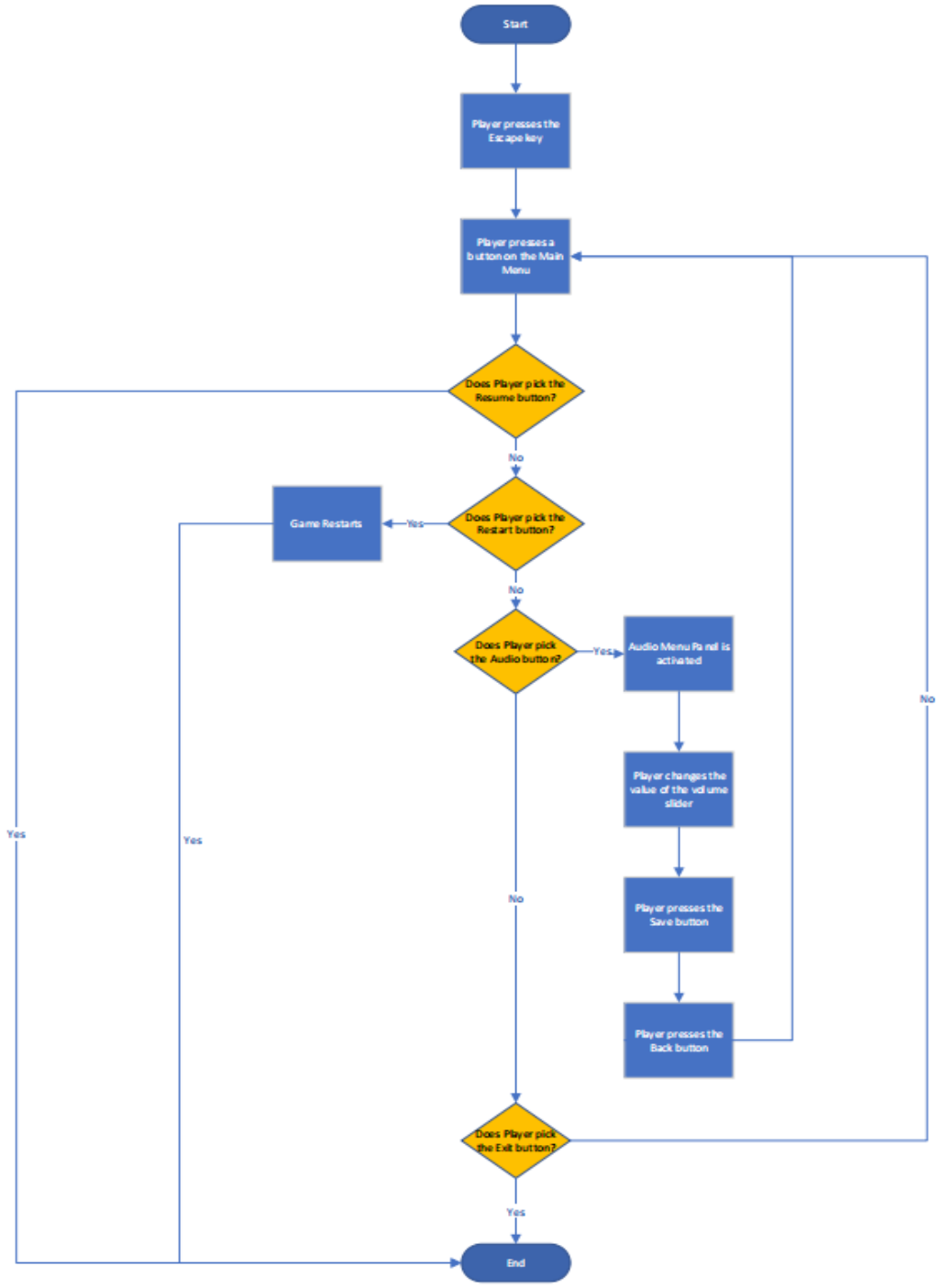


Image 20

Εικόνα 20: Η ροή λειτουργίας του κύριου μενού.

3. 2 Στοιχεία για τα Mini Games

Τα παραπάνω Mini Games έχουν σχεδιαστεί με στόχο την εκπαίδευση του χρήστη στην αγγλική γλώσσα, μέσω μίας ποικιλίας διαφορετικών τρόπων παρουσίασης και λειτουργίας. Τα παιχνίδια αυτά κατηγοριοποιούνται ως εξής:

1. Σε παιχνίδια που λαμβάνουν μέρος εντός του χώρου του παιχνιδιού και λύνονται με την χρήση του χαρακτήρα του παίχτη.
2. Σε παιχνίδια που λαμβάνουν μέρος σε διαφορετική σκηνή από αυτή του κύριου παιχνιδιού και βασίζονται σε επιλογές που πρέπει να κάνει ο παίχτης.

Ξεκινώντας από το παιχνίδι το οποίο βρίσκεται εντός του κόσμου του παιχνιδιού, ζητάτε από τον παίχτη να χρησιμοποιήσει τον χαρακτήρα του, για να "πατήσει" πάνω σε γράμματα προκειμένου να ολοκληρωθεί η πρόταση ή η λέξη που ορίζει το παιχνίδι αυτό. Για διευκόλυνση των χρηστών, όταν επιλέξουν ένα σωστό γράμμα, τότε αυτό εμφανίζεται στην λέξη ή πρόταση και εξαφανίζεται από τα πιθανά γράμματα που μπορεί να επιλέξει ο χρήστης. Οι λέξεις που χρησιμοποιούνται είναι απλού χαρακτήρα (π.χ. η εύρεση της λέξης: La?? → Land) και αφορούν την εύρεση ουσιαστικών. Από τον τρόπο που σχεδιάστηκε το παιχνίδι αυτο, δημιουργήθηκαν και ερωτήσεις απάντησης, εντός του κόσμου του παιχνιδιού, όπου ο χαρακτήρας του παίχτη πρέπει να "πατήσει" σε μια απάντηση "Yes/No" ή "True/False" για την ερώτηση αυτή. Αυτές οι ερωτήσεις βασίζονται στην κοινή λογική για να απαντηθούν. Έτσι αξιολογείται και η γνώση του παίχτη αλλά και η κριτική του ικανότητα. Για να υπάρξει τρόπος ένδειξης της απάντησης στον χρήστη, επιλέχτηκε, κάθε φορά που "πατάει" ο χαρακτήρας του παίχτη σε λάθος γράμμα, να αφαιρούνται πόντοι υγείας από αυτόν ενώ στην αντίθετη περίπτωση να αναπληρώνεται ο χαμηλότερος πόρος που έχει εκείνη την περίοδο.

Image 21

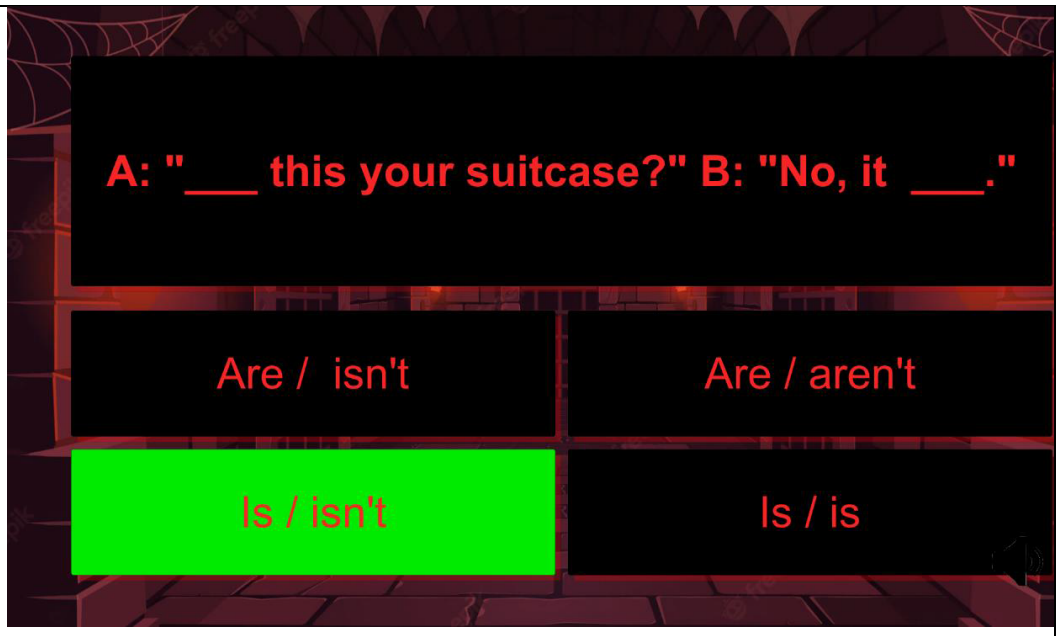
Εικόνα 21: Το παιχνίδι letter on the ground.(Επιλογή σωστής / λάθος απάντησης)

Image 22

Εικόνα 22: Το παιχνίδι επιλογής σωστής απάντησης.

Συνεχίζοντας, ακολουθούν τα παιχνίδια τα οποία βρίσκονται σε διαφορετική σκηνή από την κύρια. Το πρώτο από τα παιχνίδια αυτά είναι η απάντηση πολλαπλών ερωτήσεων, στο οποίο έχουν επιλεγεί ερωτήσεις γραμματικής και λεξιλογίου για καλύτερη εξάσκηση του παίχτη σε αυτά τα πεδία, εφόσον θα αλληλεπιδρά με το παιχνίδι αυτό επανειλημμένα. Αποτελείται δε, από μία ερώτηση και τέσσερις πιθανές απαντήσεις στις οποίες πρέπει να

επιλέξει την σωστή. Σε περίπτωση που επιλέξει, με το αριστερό κλικ του ποντικιού του σωστά, το χρώμα της απάντησης αλλάζει σε πράσινο για να φανεί ότι απάντησε σωστά αλλιώς αλλάζει σε κόκκινο για την περίπτωση που απάντησε λάθος.



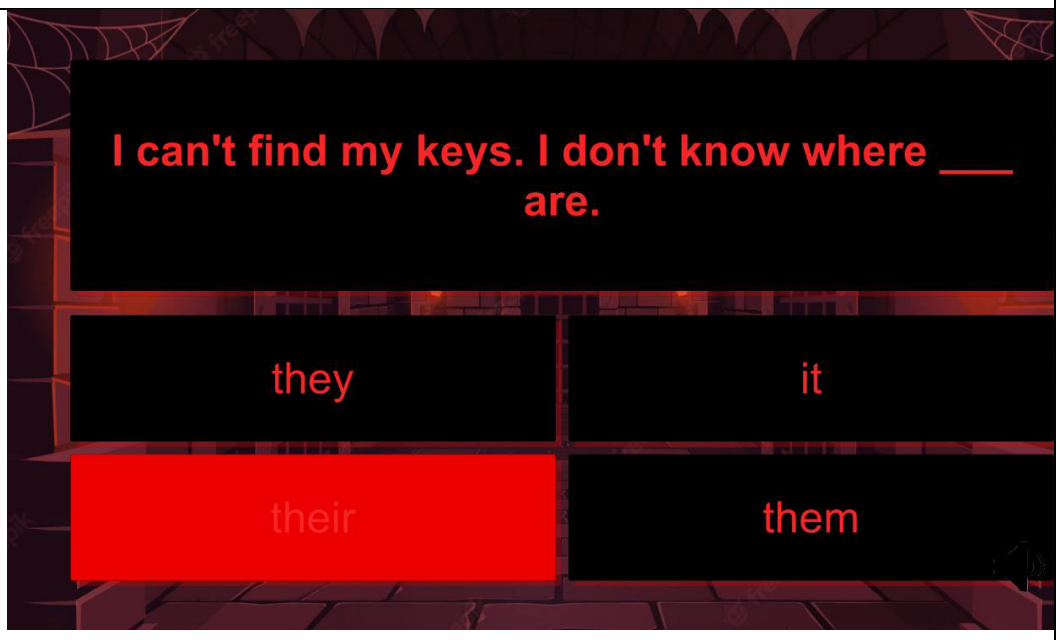


Image 23

Εικόνα 23: Το παιχνίδι πολλαπλών ερωτήσεων. (Επιλογή σωστής / λάθος απάντησης)

Το δεύτερο παιχνίδι είναι το Scramble Game στο οποίο ο παίχτης πρέπει να τοποθετήσει τα γράμματα που του δίνονται σε σωστή σειρά, προκειμένου να δημιουργηθεί μία λέξη. Έτσι ο παίχτης εκπαιδεύεται στον τομέα της ορθογραφίας των λέξεων της αγγλικής γλώσσας. Για να ολοκληρώσει το παιχνίδι, πρέπει να "πατήσει" πάνω σε ένα από τα γράμματα και να το σύρει στην σωστή θέση, τοποθετώντας τελικά τα γράμματα στις κατάλληλες θέσεις, ολοκληρώνοντας την λέξη. Σε αυτή την περίπτωση για να υπάρχει feedback της επιλογής του, αποφασίστηκε κάθε φορά που το γράμμα εισέρχεται στην σωστή θέση να παραμένει. Αλλιώς επιστρέφει πίσω μαζί με τις άλλες επιλογές γραμμάτων.

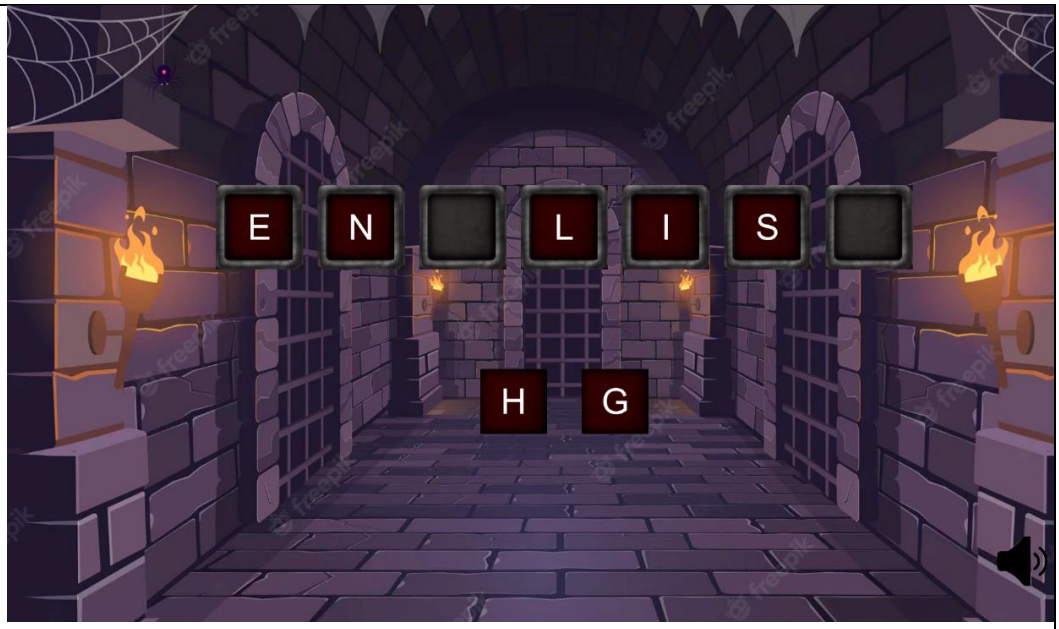


Image 24

Εικόνα 24: Το παιχνίδι scramble κατά την εξέλιξή του.

4. Υλοποίηση

4.1. Menu

Εφόσον το μενού παίζει σημαντικό ρόλο, αποφασίστηκε να γίνει η υλοποίησή του σε διαφορετική σκηνή. Ένας παραπάνω λόγος είναι ότι σε περίπτωση που δεν σχεδιαστεί έτσι θα πρέπει να ενταχθεί με την αρχική σκηνή, στην οποία λαμβάνει μέρος το παιχνίδι πράγμα το οποίο όχι μόνο θα επιβάρυνε την σκηνή του παιχνιδιού γιατί θα έπρεπε να διατηρεί παραπάνω στοιχεία κάθε φορά που χρησιμοποιόταν αλλά και θα μπορούσαν να εμφανιστούν προβλήματα όπως η εμφάνιση του μενού πάνω στην σκηνή του παιχνιδιού υπερκαλύπτοντάς την. Ένα επίσης πλεονέκτημα του να βρίσκεται η σκηνή του μενού ξεχωριστά από την σκηνή του παιχνιδιού είναι η ευκολότερη συντήρηση.

Το εν λόγω μενού απαρτίζεται από έναν τίτλο με το όνομα του παιχνιδιού το οποίο αποτελεί αντικείμενο τύπου text, δύο επιλογές στην μορφή κουμπιών με ονόματα “Play” και “Exit” αντίστοιχα, μία μπάρα η οποία είναι αρχικά κρυφή στα μάτια του χρήστη αλλά θα χρησιμοποιηθεί αργότερα και ένα παρασκήνιο με μία εικόνα για να ολοκληρωθεί η εμφάνισή του.

4.1.1. Main Menu.cs

Στην συνέχεια ακολουθεί η λειτουργικότητα του μενού. Για να μπορέσει να εκτελέσει τις κατάλληλες λειτουργίες που του αναλογούν, δημιουργήθηκε το script με όνομα Main Menu. Το script αυτό δέχεται δύο αντικείμενα τύπου GameObject τα οποία είναι:

1. Το στοιχείο **Loading Interface** το οποίο είναι η μπάρα η οποία έχει δύο στοιχεία μία background μαύρη εικόνα και μία foreground κόκκινη εικόνα, έτσι ώστε όταν απαιτείται να κινηθεί από το script να αντικατοπτρίζει την πρόοδο της φόρτωσης των σκηνών του παιχνιδιού.

2. Το στοιχείο του μενού που αποτελεί το σύνολο όλων των αντικειμένων από τα οποία δημιουργείται, για να χρησιμοποιηθεί κυρίως για την απόκρυψη και επαναφορά όποτε κρίνεται απαραίτητο.

Επίσης, δέχεται την εικόνα τύπου Image που έχει τεθεί ως foreground και με τις απαραίτητες αλλαγές όπως την χρήση Πηγής Εικόνας , τίθεται ο τύπος εικόνας ως γεμάτη, τίθεται ο τύπος γεμίσματος ως οριζόντιου και προέλευση γεμίσματος ως αριστερή, γιατί θα

πρέπει μέσω scripting να υπάρχει πρόσβαση και αλλαγή του πεδίου της ποσότητας του γεμίσματος της εικόνας.

Εφόσον πρέπει να μπορέσουν να φορτωθούν πάνω από μία σκηνές, για να έχει την δυνατότητα το παιχνίδι να αντλήσει τις κατάλληλες πληροφορίες για τις σκηνές αυτές, θα χρησιμοποιηθεί μία λίστα η οποία θα ορίζει την σειρά με την οποία θα πρέπει να φορτώνονται οι σκηνές.

Αρχικά, στην συνάρτηση Start τίθεται η κατάσταση του loadingInterface σε ψευδής, με αποτέλεσμα να μην είναι ορατό από τον χρήστη και να μην εκτελεί την λειτουργία του περιμένοντας.

Στην συνέχεια ακολουθεί η συνάρτηση με όνομα StartGame στην οποία καλούνται δύο άλλες συναρτήσεις η HideMenu, η οποία θα θέσει την κατάσταση του αντικειμένου του menu ως ψευδή για να μπορέσει να εξαφανιστεί το μενού όταν φορτωθούν τα απαραίτητα στοιχεία και η ShowLoadingScreen, η οποία εμφανίζει την μπάρα φόρτωσης για να παρέχεται στον χρήστη οπτική ανατροφοδότηση για την κατάσταση του παιχνιδιού, θέτοντας την κατάσταση του αντικειμένου loadingInterface σε αληθή. Για να υπάρχει δυνατότητα φόρτωσης και διαχείρισης σκηνών, θα γίνει χρήση της βιβλιοθήκης SceneManagement, έτσι φορτώνονται ασύγχρονα οι σκηνές του παιχνιδιού στο παρασκήνιο με αρχική την Level_0 η οποία είναι η αρχική σκηνή και στην συνέχεια οι υπόλοιπες προστίθεται στην λίστα των σκηνών που πρέπει να φορτωθούν. Για να μην παραγράψουν την πρώτη σκηνή φορτώνονται με προσθετικό τρόπο με την χρήση της δεύτερης επιλογής Additive. Στην συνέχεια καλείται η συνάρτηση StartCoroutine, η οποία προσφέρει την δυνατότητα να εκτελεστεί μια ενέργεια κατά την διάρκεια της πρώτης ενέργειας, για την συνάρτηση LoadingScreen η οποία περιέχει μία float μεταβλητή για την πρόοδο της φόρτωσης και ένα for loop το οποίο ελέγχει αν για κάθε στοιχείο της λίστας των σκηνών έχει ολοκληρωθεί η φόρτωση. Όσο η εν λόγω σκηνή δεν έχει τελειώσει να φορτώνει, προσθέτει στην μεταβλητή totalProgress την πρόοδό της (αυτό γίνεται μέσω του τρόπου Async με τον οποίο φορτώθηκαν οι σκηνές). Στην συνέχεια για την σωστή λειτουργία της μπάρας φόρτωσης το πεδίο fillAmount ορίζεται ως η πρόοδος φόρτωσης προς τον αριθμό των σκηνών που έχει το παιχνίδι. Έτσι, η μπάρα δείχνει την πρόοδο φόρτωσης των σκηνών του παιχνιδιού. Τέλος, έχει οριστεί η συνάρτηση ExitGame η οποία θα σταματάει την εφαρμογή όταν καλεστεί.

Έχοντας αναλύσει την λειτουργία του script θα πρέπει να τεθεί η σωστή λειτουργία στο κατάλληλο κουμπί. Αυτό μπορεί να επιτευχθεί στις ρυθμίσεις του κάθε κουμπιού επιλέγοντας το OnClick() event το οποίο θα εκτελέσει την επιθυμητή λειτουργία που θα του τεθεί όταν το κουμπί πατηθεί από τον χρήστη. Επομένως, για τα δύο κουμπιά δίνεται ως αναφορά το αντικείμενο που έχει το απαραίτητο script και επιλέγεται η κατάλληλη συνάρτηση για κάθε κουμπί, δηλαδή για το Play κουμπί η συνάρτηση StartGame και για το Exit κουμπί η συνάρτηση ExitGame.

Τέλος, για να μπορέσει το μενού να λειτουργήσει κανονικά και να μπορέσει να εκτελέσει την λειτουργικότητά του είναι απαραίτητη η ύπαρξη του αντικειμένου με όνομα EventSystem στην παρούσα σκηνή. Χωρίς αυτό τα κουμπιά δεν θα μπορούνε να είναι αλληλεπιδράσιμα με αποτέλεσμα να μην μπορεί ο χρήστης να ξεπεράσει την αρχική οθόνη.

4.2. Level Design

Αφού δημιουργηθεί μια νέα σκηνή επιλέγεται η χρήση του δισδιάστατου αντικειμένου Tilemap το οποίο θα δημιουργήσει ένα Grid πάνω στην σκηνή. Για την καλύτερη οργάνωσή του επιλέχτηκε η χρήση τεσσάρων Tilemaps (Shadow,Ground, Wall, Decorations) και η ανάθεση layer(Backgorund,Middleground,Foreground) στο καθένα ώστε να υπάρχει μεγαλύτερη ευελιξία στην κατασκευή του. Στην συνέχεια με την βοήθεια των [1],[2] επιλέγεται ένα σετ από εικόνες για την δημιουργία του χάρτη και των στοιχείων του. Αφού γίνει χωρισμός των sprite σε υπό-εικόνες ξεκινάει η διαδικασία δημιουργίας του χάρτη.

Αφού ολοκληρωθεί η αισθητική εικόνα του χάρτη πρέπει να προστεθεί το TilemapCollider2D αντικείμενο στους τοίχους για να περιορίζουν την κίνηση του χαρακτήρα του χρήστη. Αλλά παρατηρείται ότι το αποτέλεσμα εκτελεί παραπάνω υπολογισμούς για σημεία που ο χρήστης δεν έχει την δυνατότητα να πατήσει. Για να λυθεί αυτό το πρόβλημα θα πρέπει να προστεθεί το στοιχείο CompositeCollider2D.

Αφού χρησιμοποιηθεί το παραπάνω στοιχείο πρέπει να τεθεί ως 0 η βαρύτητα του Rigidbody2D που προστέθηκε αλλιώς το τοίχος θα βρεθεί εκτός ορίων όταν αρχίσει το παιχνίδι.

.

4.2.1. Entrance.cs

Στο script γίνεται ο έλεγχος του κωδικού του χαρακτήρα σε σχέση με τον κωδικό εισόδου της σκηνή που θέλει να μεταβεί. Αυτό γίνεται για να επιλεγεί σωστά η σκηνή στην οποία προσπαθεί να εισέλθει. Εφόσον οι κωδικοί είναι ίδιοι, επιτρέπεται η μεταφορά προς την σκηνή στόχο και μεταφέρεται στο σημείο εισόδου αυτής της σκηνής.

4.2.2. Exit.cs

Η δουλειά του Exit script είναι να κρατάει έναν κωδικό(σε αυτή την περίπτωση το όνομα της σκηνής που ο χαρακτήρας θέλει να πάει) και να ελέγξει με την βοήθεια της συνάρτησης OnTriggerEnter2D αν το αντικείμενο που συγκρούστηκε μαζί του είναι ο χαρακτήρας του χρήστη, ελέγχοντας την ετικέτα του αντικειμένου. Αν είναι τότε αλλάζει την τιμή του κωδικού του χαρακτήρα και τον μεταβιβάζει στην επιθυμητή σκηνή καλώντας την συνάρτηση FadeTo από το scprit SceneFade το οποίο παρέχει τη δυνατότητα αλλαγής σκηνής με μία μαύρη εικόνα που χρησιμοποιείται για fade-in και fade-out της σκηνής (αξιοποιώντας το άλφα κανάλι μίας εικόνας που είναι υπεύθυνο για την διαφάνεια).

4.2.3. SceneFade.cs

Αρχικά, ο λόγος που η συνάρτηση FadeIn ορίστηκε ως IEnumerator ήταν η ανάγκη της ταυτόχρονης εκτέλεσης της συνάρτησης και της μεταβίβασης του χαρακτήρα σε

διαφορετική σκηνή, αλλιώς θα υπήρχε πρόβλημα μεταβίβασης στην νέα σκηνή και ολοκλήρωσης της συνάρτησης στο ίδιο frame, πράγμα που σημαίνει ότι δεν θα προλάβει να γίνει ορατή η αλλαγή της fade-in εικόνας στον χρήστη.

Συνεπώς, στην συνάρτηση FadeIn αρχικά ορίζεται η τιμή του άλφα καναλιού ως 1 (χωρίς διαφάνεια) και στην συνέχεια γίνεται έλεγχος (όσο η τιμή του A είναι θετική) αλλάζει το χρώμα της στο νέο άλφα το οποίο μειώνεται με βάση τον χρόνο και τελειώνει την κλήση χωρίς καμία χρονική καθυστέρηση. Η συνάρτηση αυτή καλείται όταν ο χρήστης μεταβεί στην σκηνή του παιχνιδιού από την σκηνή του μενού, με την χρήση της εντολής StartCoroutine για να εκτελεστούν οι δύο ενέργειες ταυτόχρονα.

Ακολουθεί η συνάρτηση FadeOut η οποία θα χρησιμοποιηθεί όταν ο χρήστης επανέλθει από μία νέα σκηνή πίσω στην προηγούμενη. Αρχικά η τιμή του καναλιού άλφα ορίζεται ως 0(δηλαδή η εικόνα είναι πλήρως διαφανής) και σταδιακά αυξάνεται μέχρι η τιμή του να γίνει 1. Στην συνέχεια, αλλάζει η σκηνή σε αυτήν που πρέπει να βρεθεί ο χαρακτήρας. Τέλος, η συνάρτηση αυτή καλείται από την συνάρτηση FadeTo η οποία παρέχει το όνομα της σκηνής προς μεταβίβαση.

Να σημειωθεί ότι η είσοδος και η έξοδος πρέπει να έχουν συγκεκριμένες θέσεις μέσα στην σκηνή προκειμένου να αποφευχθεί το πρόβλημα της ατελείωτης αλλαγής μεταξύ δυο σκηνών.

4.3. Camera

Η κάμερα είναι το πιο σημαντικό στοιχείο του παιχνιδιού. Υπάρχουν πολλοί τρόποι για την λειτουργία και την συμπεριφορά της κάμερας όπως είναι η χρήση διαφορετικών στατικών καμερών ή η χρήση μοναδικής κάμερας η οποία ακολουθεί τον παίχτη. Για να γίνει πιο ενδιαφέρον αποφασίστηκε η κάμερα να μην είναι κολλημένη πάνω στον χαρακτήρα του χρήστη(επειδή ο τρόπος δεν είναι απόλυτα φυσικός) αλλά να επιχειρεί να τον ακολουθεί με ομαλό τρόπο.

4.3.1. CameraController.cs

Πριν ξεκινήσει η διαδικασία της κατασκευής της κάμερας το πρώτο πράγμα που πρέπει να ληφθεί υπόψιν είναι το γεγονός ότι η κάμερα που είναι ενεργή σε μία σκηνή ανά πάσα πιθανότητα πρέπει να είναι μοναδική, σε περίπτωση πολλών δημιουργείται πρόβλημα προτεραιότητας μεταξύ τους.

Για τον λόγο αυτό θα μετατραπεί η κύρια κλάση του script σε τύπο Singleton, δηλαδή θα υπάρχει μόνο μία εκδοχή ενεργή σε οποιαδήποτε σκηνή καθ' όλη την διάρκεια του παιχνιδιού πράγμα το οποίο επιτυγχάνεται στην συνάρτηση Awake.

Ειδικότερα, ορίζεται ένα στατικό στιγμιότυπο του script και ελέγχεται αν υπάρχουν παραπάνω από ένα ενεργά αντικείμενα που το περιέχουν και κρατιέται μόνο ένα ενώ καταστρέφονται τα υπόλοιπα.

Ξεκινώντας, η συνάρτηση Start θέτει ως στόχο της κάμερας το αντικείμενο το οποίο περιέχει το script PlayerController, με άλλα λόγια τον χαρακτήρα τον οποίο χειρίζεται ο χρήστης. Στην συνέχεια ακολουθεί η συνάρτηση LateUpdate η οποία προτιμήθηκε έναντι της Update γιατί έχει την δυνατότητα παρακολούθησης αντικειμένων που μπορεί να έχουν μετακινηθεί μέσα στο Update. Αυτό συμβαίνει γιατί κατά την εκτέλεση και ειδικότερα στο πρώτο frame η κάμερα δεν έχει την δυνατότητα να "βρει" το PlayerController στο ίδιο frame με αποτέλεσμα να μην εμφανίζει σωστά αποτελέσματα.

Το πρόβλημα βρίσκεται στον τρόπο με τον οποίο το Unity χειρίζεται την σειρά εκτέλεσης των event συναρτήσεων[3].

Έπειτα, αλλάζει η τοποθεσία της κάμερας χρησιμοποιώντας την συνάρτηση Lerp η οποία δέχεται τις συντεταγμένες που έχει αυτή την στιγμή η κάμερα ως πρώτο όρισμα, τις συντεταγμένες του στόχου ως δεύτερο όρισμα και την ταχύτητα που πρέπει να έχει για να φτάσει στον στόχο ως τρίτο.

Τέλος, για να γίνει καλύτερη η λειτουργικότητα της κάμερας πάρθηκε η απόφαση να σταματάει σε κάποιο όριο του χάρτη για να μην εμφανίσει σημεία τα οποία δεν είναι μέρος του παιχνιδιού.

Αυτό γίνεται θέτοντας τα όρια στα οποία σταματάει η κάμερα μετά από εξονυχιστικές δοκιμές. Η ίδια διαδικασία πρέπει να επαναληφθεί για όλες τις σκηνές ξεχωριστά.

4.4. Traps

Σε αυτό το πρότζεκτ χρησιμοποιούνται δύο είδη παγίδων, μια παγίδα με καρφιά και μία παγίδα που εκτοξεύει τόξα.

4.4.1. SpikeTrap.cs

Αρχικά πρέπει η παγίδα να χτυπάει τον χαρακτήρα όταν πατήσει πάνω της και είναι ενεργή. Για αυτό θα χρησιμοποιηθεί ως βάση η συνάρτηση OnTriggerEnter2D που υπάρχει ήδη υλοποιημένη στο script DamagePlayer μέσω κληρονομικότητας.

Στην συνέχεια στην συνάρτηση Start βρίσκεται το αντικείμενο animator για να μπορέσει να αλλάξει η τιμή της bool μεταβλητής isWorking η οποία ελέγχει την κατάσταση της παγίδας.

Τέλος, στην συνάρτηση Update αφαιρείται χρόνος από την μεταβλητή cooldownTimer κατά την διάρκεια της οποίας η παγίδα δεν είναι ενεργή. Αν τελειώσει ο χρόνος αναμονής τότε η μεταβλητή isWorking αντιστρέφεται και ορίζεται νέος χρόνος αναμονής.

4.4.2. ArrowTrap.cs

Στην συνάρτηση FindArrow επιστρέφονται οι θέσεις των αντικειμένων των βελών. Συνεχίζοντας, ακολουθεί η συνάρτηση Attack η οποία ορίζει το χρόνο αναμονής για την απελευθέρωση του βέλους σε 0 και θέτει την τοποθεσία του στην σωστή θέση κάνοντάς το ενεργό.

Μετά την ολοκλήρωση της παραπάνω διαδικασίας το βέλος κινείται στην κατεύθυνση που του ορίζει το δικό του script το οποίο προκαλεί ζημιά στον χαρακτήρα του παίχτη όταν βρεθούν σε επαφή και εξαφανίζει το βέλος.

4.5. Enemies

Οι εχθροί που εμφανίζονται στο παιχνίδι είναι δύο ειδών, ένας χωρίς animations και ένας που έχει πιο μεγάλη πολυπλοκότητα γιατί περιέχει animations.

4.5.1. Απλοί Εχθροί χωρίς animation

Για την λειτουργία των εχθρών επιλέχτηκε η ικανότητα περιπολίας μεταξύ δύο σημείων. Ο εχθρός σταματάει την περιπολία για να κυνηγήσει τον παίχτη όσο βρίσκεται μέσα σε ένα επιτρεπτό όριο απόστασης από τον εαυτό του. Σε περίπτωση που ο παίχτης βγει εκτός του ορίου ο εχθρός επιστρέφει στην περιπολία του. Αποφασίστηκε σε αυτή την περίπτωση τα scripts τα οποία χρησιμοποιούνται, να χωριστούν σε διαφορετικά κομμάτια παρά να βρίσκονται σε ένα.

AIPatrol.cs

Για να μπορέσει να αρχίσει η περιπολία του εχθρού, επιλέγονται τα σημεία τα οποία θα τεθούν ως οι συντεταγμένες στις οποίες πρέπει να κινηθεί και τοποθετούνται δύο αντικείμενα τα οποία θα δράσουν ως τερματικά σημεία. Έτσι, στην συνάρτηση Start ορίζεται ως αρχικός στόχος το σημείο 1 και λαμβάνεται το στοιχείο SpriteRenderer για να χρησιμοποιηθεί κατά την αλλαγή κατεύθυνσης που κοιτάει ο εχθρός όταν αλλάζει ο στόχος περιπολίας.

Με αυτό το σκεπτικό, στην συνάρτηση Update κατευθύνεται ο εχθρός προς ένα από τα σημεία χρησιμοποιώντας την συνάρτηση MoveTowards η οποία δέχεται ως ορίσματα τις συντεταγμένες θέσης που έχει το αντικείμενο κάθε frame, τις συντεταγμένες που έχει ο στόχος στον οποίο επιχειρεί να φτάσει ο εχθρός και την ταχύτητα με την οποία θα φτάσει στον στόχο. Να σημειωθεί ότι χρησιμοποιήθηκε Vector2 και όχι Vector3 εφόσον το παιχνίδι είναι δύο διαστάσεων(2D) καθιστώντας τον άξονα των z ως περιττό. Στην συνέχεια γίνεται έλεγχος αν η απόστασή του από τον τελικό στόχο είναι η ελάχιστη. Ο λόγος που επιλέχτηκε να γίνει ο έλεγχος με αυτόν το τρόπο και όχι με την ακριβή τιμή της απόστασης είναι η παρουσίαση προβλημάτων στην κατεύθυνση που βλέπει ο εχθρός. Έχοντας ικανοποιηθεί η παραπάνω συνθήκη, αλλάζει ο στόχος του στο δεύτερο σημείο και εφόσον αλλάζει κατεύθυνση χρησιμοποιείται το στοιχείο SpriteRenderer για να γυρίσει το sprite του στον άξονα των X με αποτέλεσμα να γυρνάει κατά 180° . Με τον ίδιο τρόπο όταν ο εχθρός βρεθεί στο σημείο 2 αλλάζει ξανά ο στόχος του πίσω στο σημείο 1 και αυτή την φορά δεν είναι απαραίτητη η αλλαγή προσανατολισμού του εχθρού.

Enemy.cs

Αρχικά εφόσον αποφασίστηκε ότι το script για την περιπολία θα βρίσκεται σε ξεχωριστό script, πρέπει να ληφθεί υπόψιν το ενδεχόμενο να μην υπάρχει τοποθετημένο πάνω σε κάποιο από τα άλλα αντικείμενα των εχθρών έτσι ώστε να μην δημιουργηθούν

προβλήματα αργότερα που θα γίνει χρήση του συγκεκριμένου script. Έτσι, στην συνάρτηση Awake γίνεται ο έλεγχος αν υπάρχει το script στο αντικείμενο.

Συνεχίζοντας, ακολουθεί η συνάρτηση Update στην οποία βρίσκεται και αποθηκεύεται το script του χαρακτήρα του παίχτη και σε περίπτωση που βρίσκεται εντός μίας ζώνης επιρροής του εχθρού τότε τον κυνηγάει. Ακολουθεί η συνάρτηση OnTriggerEnter2D στην οποία όσο ο χαρακτήρας του παίχτη βρίσκεται μέσα στον κύκλο επιρροής του εχθρού τότε ορίζεται ως ο νέος στόχος μετακίνησης. Σε περίπτωση που περιέχει το script περιπολίας, τίθεται ως μη ενεργό για να μην υπάρξουν προβλήματα στην κατεύθυνση στην οποία κοιτάει ο εχθρός. Σε περίπτωση που ο χαρακτήρας του παίχτη βγει από την ζώνη επιρροής τότε ο στόχος του εχθρού χάνεται. Στην περίπτωση που περιέχει το script περιπολίας τότε επανέρχεται σε λειτουργία. Τέλος, όταν φτάσει τον χαρακτήρα τού παίχτη τότε του προκαλεί ζημιά στους πόντους υγείας του.

4.5.2. Εχθροί με Animation

Για τους εχθρούς που έχουν Animations αποφασίστηκε να χρησιμοποιηθούν τα παρακάτω scripts για την πλήρη επίδειξη της λειτουργικότητάς του.

TriggerAreaCheck.cs

Αρχικά ορίζεται ένα αντικείμενο τύπου script με το όνομα skeletonParent. Στην Awake συνάρτηση λαμβάνεται το script Skeleton_Behaviour το οποίο υπάρχει στο αντικείμενο γονέα. Στην συνέχεια στην OnTriggerEnter2D αν βρεθεί σε επαφή με τον χαρακτήρα του χρήστη τότε η περιοχή που έλεγχε για την είσοδο του παίχτη απενεργοποιείται, αλλάζει η μεταβλητή inRange του Skeleton_Behaviour script σε true, τίθενται ως στόχος οι συντεταγμένες του παίχτη και στην συνέχεια ενεργοποιείται το αντικείμενο hotZone μαζί με το στοιχείο εμβέλειας το οποίο θα ελέγχει την ύπαρξη του παίχτη μέσα σε αυτό.

HotZoneCheck.cs

Στην Awake συνάρτηση λαμβάνονται τα στοιχεία Skeleton_Behaviour script και το animator από το πατρικό αντικείμενο. Ακολουθεί η συνάρτηση Update στην οποία ελέγχεται αν ο παίχτης βρίσκεται αρκετά κοντά στον εχθρό ώστε να εκτελέσει μία επίθεση και ταυτόχρονα δεν εκτελείται το animation της επίθεσης τότε ο εχθρός μπορεί να αλλάξει την κατεύθυνση που βλέπει. Αυτό γίνεται για την περίπτωση που ο παίχτης κατά την διάρκεια που εκτελείται το animation της επίθεσης βρεθεί από πίσω από τον εχθρό κάνοντάς τον να αλλάξει την κατεύθυνση της επίθεσης πράγμα το οποίο δεν θεωρείται δίκαιο.

Στην συνέχεια στην συνάρτηση OnTriggerEnter2D όσο ο παίχτης βρίσκεται μέσα στην ζώνη επιρροής του εχθρού τότε θεωρείται ως στόχος και μπορεί να δεχτεί επιθέσεις από αυτόν.

Τελειώνοντας, με την συνάρτηση OnTriggerExit2D σε περίπτωση που ο παίχτης βρεθεί εκτός της ζώνης επιρροής τότε δεν θεωρείται ως στόχος για τον εχθρό. Απενεργοποιείται η HotZone περιοχή, ενεργοποιείται ξανά η περιοχή η οποία ελέγχει για την ύπαρξη του παίχτη και ο εχθρός επιλέγει ως νέο σημείο το πιο κοντινό προκαθορισμένο σημείο από αυτά που είχε για την περιπολία του.

Skeleton_Behaviour.cs

Ξεκινώντας με την συνάρτηση Awake αρχικά καλείται η συνάρτηση SelectTarget στην οποία θέτονται τα όρια της περιπολίας και με βάση την μεγαλύτερη απόσταση τίθεται ανάλογα ο στόχος για την αρχή του κύκλου της περιπολίας. Στην συνέχεια γίνεται ο ορισμός τιμών όπως του χρόνου για το animation της επίθεσης και της έναρξης του χρόνου αναμονής για την εκκίνηση μίας νέας επίθεσης, καθώς και η αποθήκευση του στοιχείου Animator και του script που ελέγχει την υγεία του εχθρού για χρήση παρακάτω.

Ακολουθεί η συνάρτηση Update στην οποία γίνεται έλεγχος της υγείας του εχθρού και σε περίπτωση που γίνει μικρότερη από 0, οι τιμές που ελέγχουν τα animations για την επίθεση, κατάσταση του εχθρού και κίνηση αλλάζουν σε ψευδή και η ταχύτητα αλλάζει σε 0. Συνεχίζοντας, αν ο εχθρός δεν βρίσκεται σε κατάσταση επίθεσης τότε καλείται η συνάρτηση Move η οποία θέτει την μεταβλητή του animation για περιπολία ως αληθή, ελέγχοντας αν δεν εκτελείται το animation της επίθεσης για να μπορέσει να κινηθεί χωρίς να γίνεται επικάλυψη των animations μεταξύ τους προκαλώντας προβλήματα. Ακολουθώντας την ροή της Update συνάρτησης, στην συνέχεια γίνεται έλεγχος για το αν ο εχθρός δεν βρίσκεται μέσα στην απόσταση ανάμεσα στα δύο σημεία που περιπολεί και δεν βρίσκεται ο παίχτης σε επαρκή απόσταση για επίθεση και δεν εκτελείται το animation της επίθεσης, τότε επιλέγει το σημείο περιπολίας που βρίσκεται πιο μακριά.

Προχωρώντας, σε περίπτωση που ο παίχτης βρίσκεται μέσα στην ζώνη επιρροής τού εχθρού τότε καλείται η συνάρτηση EnemyLogic στην οποία ορίζεται η απόσταση μεταξύ τού παίχτη και του εχθρού. Στην συνέχεια ελέγχεται αν η απόσταση των δύο είναι μεγαλύτερη από την απαιτούμενη απόσταση που επιτρέπει να εκτελεστεί μία επίθεση. Αν ο έλεγχος είναι αληθής τότε καλείται η συνάρτηση StopAttack η οποία θέτει την κατάσταση της επίθεσης ως ψευδή.

Στην συνέχεια, ελέγχεται αν ο εχθρός βρίσκεται αρκετά κοντά στον παίχτη και αν η επίθεσή του δεν βρίσκεται σε αναμονή, τότε καλείται η συνάρτηση Attack η οποία σταματάει το animation της περιπολίας και επιτρέπει την κατάσταση επίθεσης στην οποία μπορεί να εκτελεστεί το animation της επίθεσης ορίζοντας τον χρόνο αναμονής.

Ολοκληρώνοντας την Update συνάρτηση, ο τελικός έλεγχος που γίνεται είναι αν ο εχθρός βρίσκεται σε κατάσταση αναμονής επίθεσης. Τότε καλείται η συνάρτηση Cooldown η οποία μειώνει τον χρόνο αναμονής και ο εχθρός βρίσκεται σε κατάσταση επίθεσης αλλά δεν μπορεί να εκτελέσει επίθεση λόγω της κατάστασης αναμονής από εκτέλεση προηγούμενης επίθεσης. Όταν ο χρόνος αναμονής γίνει 0 τότε γίνεται επιτρεπτή η εκτέλεση της νέα επίθεσης όπου αργότερα τίθεται νέος χρόνος αναμονής ολοκληρώνοντας την διαδικασία επίθεσης-αναμονής.

4.6. Interactable Objects

Ως αλληλεπιδράσιμα αντικείμενα θεωρούνται τα αντικείμενα τα οποία μπορεί να χρησιμοποιήσει ο παίχτης. Υπάρχουν τέσσερα αντικείμενα με τα οποία μπορεί να αλληλεπιδράσει ο παίχτης που δεν θεωρούνται mini-games, τα οποία θα σχολιαστούν σε επόμενη παράγραφο. Αυτά είναι τα δύο είδη potions τα οποία βοηθούν τον παίχτη δίνοντάς του πίσω πόρους που έχει χρησιμοποιήσει(mana ή health), τα διάφορα κλειδιά τα οποία χρησιμοποιούνται για να ανοιχτούν τα τελικά αντικείμενα με τα οποία μπορεί να αλληλεπιδράσει ο παίχτης στην μορφή των πορτών που υπάρχουν στις σκηνές του παιχνιδιού.

Τα potions κατηγοριοποιούνται σε επίπεδο(layer), στο οποίο δεν μπορούν να καταστραφούν από επιθέσεις εχθρών ή τις επιθέσεις του χαρακτήρα του παίχτη. Επίσης χρησιμοποιείται ένα μικρό animation πηδήματος σε loop το οποίο θα βοηθήσει στον τομέα της ορατότητάς του μέσα στην σκηνή.

4.6.1. HealthPotion.cs

Η βασική ιδέα του script είναι ο έλεγχος της ετικέτας του αντικειμένου για τον χειρισμό της λειτουργίας του και το είδος ηχητικού εφέ το οποίο πρέπει να έχει όταν έρχεται σε επαφή με τον χαρακτήρα του χρήστη.

Επομένως, στην συνάρτηση OnTriggerEnter2D ελέγχεται αν βρίσκεται σε επαφή με τον χαρακτήρα του παίχτη και καθορίζεται με βάση την ετικέτα του(tag), τι είδους πόρο θα αυξήσει για τον παίχτη και το ηχητικό εφέ που θα έχει όταν χρησιμοποιηθεί. Αφού χρησιμοποιηθεί το αντικείμενο κρύβεται.

4.6.2. InteractionObject.cs

Χρησιμοποιείται η συνάρτηση FixedUpdate έναντι της Update λόγω της ιδιότητάς της να εκτελείται λιγότερες φορές ([5][6] περίπου κάθε 0.02 δευτερόλεπτα ή αλλιώς 50 φορές το δευτερόλεπτο σε αντίθεση των 60 φορών/δευτερόλεπτο της Update) μειώνοντας το φόρτο του παιχνιδιού για ελέγχους που δεν είναι απαραίτητοι να εκτελούνται σε κάθε frame.

Στην FixedUpdate συνάρτηση ελέγχεται αν υπάρχει στην ενεργή σκηνή ή σκηνές, αντικείμενο πόρτας το οποίο δεν μπορεί να ανοιχτεί επειδή το αναγκαίο αντικείμενο κλειδί δεν υπάρχει και το καταστρέφει. Αυτό συμβαίνει σε περίπτωση επανεμφάνισης του ίδιου αντικειμένου όταν ο παίχτης ξαναφορτώνει την ίδια σκηνή μετά από Game Over, όπου στην ίδια σκηνή υπάρχουν την ίδια στιγμή δύο αντίγραφα μίας πόρτας στοιβαγμένα το ένα πάνω στο άλλο με την διαφορά ότι το ένα έχει αναφορά τού απαραίτητου αντικειμένου για να ανοίξει ενώ το άλλο δεν έχει εφόσον έχει καταστραφεί πριν την επανέναρξη του παιχνιδιού.

Στην συνέχεια, ακολουθεί η συνάρτηση SetItem η οποία επιστρέφει το απαραίτητο αντικείμενο που χρειάζεται η πόρτα για να ανοιχτεί. Ακολουθεί η συνάρτηση DoInteraction η

οποία ενεργοποιείται όταν ο παίχτης επιχειρήσει να ανοίξει την πόρτα, ασχέτως αν έχει το απαραίτητο κλειδί για το άνοιγμά της.

Σε περίπτωση που έχει το σωστό κλειδί, καλείται η συνάρτηση Open στην οποία ελέγχεται αν το Mini-Game που πρέπει να παιχτεί υπάρχει και αν έχει καταστραφεί, απλά απενεργοποιεί το αντικείμενο της πόρτας χωρίς να προσπαθήσει να το καλέσει, οδηγώντας σε προβλήματα, αλλά αν είναι διαθέσιμο τότε λαμβάνεται το όνομα της σκηνής στην οποία βρίσκεται ο χαρακτήρας του παίχτη, κρατιέται η κατάσταση στην οποία βρίσκεται το μίνι παιχνίδι και γίνεται ενεργή η σκηνή του μίνι παιχνιδιού για τον παίχτη. Τελειώνοντας το Mini-Game απενεργοποιείται το αντικείμενο της πόρτας, ανοίγοντας τον δρόμο για τον παίχτη όταν επιστρέψει στην σκηνή αφού τελειώσει με το μίνι παιχνίδι.

Τα αντικείμενα κλειδιών και πορτών χρησιμοποιούν το ίδιο script για την λειτουργία τους με την διαφορά ορισμένων παραμέτρων για την σωστή κατηγοριοποίηση τους.

4.7. Mini Games

Τα mini Games είναι τα παιχνίδια τα οποία προσφέρουν στον χρήστη έναν τρόπο για την ανανέωση του στοιχείου της δύναμης(mana) έναντι της αναζήτησης potions από ηττημένους εχθρούς. Κάθε φορά που ο παίχτης απαντά σωστά στις ερωτήσεις που του εμφανίζονται ή λύνει με επιτυχία κάποιο παζλ τότε ανταμείβεται με αύξηση της δύναμής του.

4.7.1. Find Word Mini Game

Αυτό το παιχνίδι βρίσκεται αποκλειστικά μέσα στον κόσμο του παιχνιδιού. Όταν ο χαρακτήρας πατήσει σε ένα γράμμα τότε αν η λέξη που προκύπτει πρόκειται για έγκυρη λέξη ανταμείβεται με αύξηση των πόντων υγείας του εξαφανίζοντας το γράμμα που έδωσε ως απάντηση και εμφανίζοντας την ολοκληρωμένη λέξη. Αντίθετα αν το γράμμα που πάτησε δεν καταλήγει σε έγκυρη λέξη τότε το γράμμα παραμένει στην τοποθεσία του και αφαιρεί πόντους υγείας από τον παίχτη.

LetterOnTheGround.cs

Αρχικά στην συνάρτηση Awake αποθηκεύεται σε μία μεταβλητή το αντικείμενο του χαρακτήρα του παίχτη για να χρησιμοποιηθούν οι απαραίτητες συναρτήσεις. Στην συνέχεια ακολουθεί η συνάρτηση Start στην οποία βρίσκεται και αποθηκεύεται το script από το αντικείμενο πατέρα τού αντικειμένου που το περιέχει, καλώντας το CorrectChoiceEvent κάθε φορά που ο χαρακτήρας του παίχτη πατάει πάνω σε ένα σωστό γράμμα. Αυτό γίνεται για έλεγχο της λειτουργίας του παιχνιδιού.

Ακολουθεί η συνάρτηση CorrectChoice η οποία εκτελείται κάθε φορά που ο παίχτης επιλέγει ένα σωστό γράμμα για να τον ανταμείψει με αποκατάσταση των πόντων του πόρου που έχει μεγαλύτερη έλλειψη. Αντίθετα, η συνάρτηση WrongChoice εκτελείται όταν ο παίχτης πατήσει ένα λάθος γράμμα και του αφαιρεί πόντους υγείας.

Στην συνέχεια ακολουθεί η συνάρτηση OnTriggerEnter2D(όπου καλείται όταν ο παίχτης "πατήσει" σε ένα γράμμα) στην οποία γίνεται ο απαραίτητος έλεγχος για την ορθότητα της απάντησης του παίχτη και σε περίπτωση που είναι σωστή και σχηματίζεται έγκυρη λέξη, τότε καλείται η συνάρτηση CorrectChoice, εξαφανίζεται το γράμμα που επέλεξε ο παίχτης και εμφανίζεται η απάντηση. Αξιοποιώντας το script SpawnObject αυξάνει τον αριθμό των σωστών απαντήσεων που βρήκε ο παίχτης και αφαιρεί το event για να μην καλείται παραπάνω φορές λόγο του απ ενεργοποιημένου αντικειμένου. Σε περίπτωση που ο παίχτης επιλέξει λάθος γράμμα καλείται η συνάρτηση WrongChoice αφαιρώντας πόντους υγείας για την λανθασμένη επιλογή του.

SpawnObject.cs

Η λειτουργία αυτού του script είναι η εμφάνιση του αντικειμένου κλειδιού που ξεκλειδώνει την πόρτα που σχετίζεται με το Mini Game και μπλοκάρει την πρόοδο του παίχτη. Η συνάρτηση Spawn εμφανίζει το αντικείμενο που ξεκλειδώνει την πόρτα όταν η συνάρτηση CountCorrects φτάσει στον κατάλληλο αριθμό λύσεων και επιστρέφει την τιμή των σωστών απαντήσεων για testing. Η συνάρτηση CountCorrects καλείται κάθε φορά που ο παίχτης επιλέξει σωστό γράμμα εκμεταλλευόμενος του event του παραπάνω script. Στην συνέχεια αποθηκεύεται ο αριθμός των σωστών απαντήσεων μέσω του συστήματος αποθήκευσης που θα αναλυθεί με μεγαλύτερη λεπτομέρεια παρακάτω.

4.7.2. Multiple Question Game

Το παιχνίδι αυτό επιλέχτηκε να λαμβάνει μέρος σε ξεχωριστή σκηνή από την σκηνή του κύριου παιχνιδιού. Ο κυριότερος λόγος της απόφασης αυτής είναι για να γίνει ελκυστικότερο για τον παίχτη του εκπαιδευτικού παιχνιδιού.

AnswerScript.cs

Το script θέτει το χρώμα σε πράσινο αν η επιλογή του παίχτη είναι σωστή και κόκκινο σε περίπτωση που είναι λάθος.

Ξεκινώντας, στην συνάρτηση Start λαμβάνεται το στοιχείο χρώματος της εικόνας για να γίνουν αργότερα οι κατάλληλες αλλαγές. Ακολουθούν οι συναρτήσεις IsCorrect η οποία επιστρέφει αν η μεταβλητή isCorrect είναι αληθής ή ψευδής, για την χρήση της σε άλλα scripts και η συνάρτηση SetCorrect η οποία θέτει στην μεταβλητή isCorrect την σωστή απάντηση. Τέλος, ακολουθεί η συνάρτηση Answer η οποία με βάση την επιλογή της απάντησης του χρήστη στο ερώτημα που του τίθεται αποδίδει το κατάλληλο χρώμα στο αντικείμενο που επέλεξε.

QuestionsAndAnswers.cs

Σε αυτό το script περιέχονται τα απαραίτητα στοιχεία που θα χρησιμοποιηθούν από το mini game όπως είναι οι ερωτήσεις, οι πιθανές απαντήσεις και η σωστή απάντηση. Τέθηκαν ως δημόσια στοιχεία προκειμένου να δημιουργηθούν πιο εύκολα οι ερωτήσεις χωρίς την χρήση πολλών συναρτήσεων παραλαβής δεδομένων όπως θα συνέβαινε αν ήταν ιδιωτικά. Συνεπώς, αποτελείται από μια μοναδική κλάση με όνομα QuestionsAndAnswers η οποία περιέχει το όνομα της ερώτησης, έναν πίνακα(τεσσάρων θέσεων) για τις πιθανές απαντήσεις και τον αριθμό της σωστής απάντησης.

QuizManager.cs

Το script αυτό χειρίζεται την σειρά εμφάνισης του set των ερωτήσεων, καθώς και την επιβράβευση της επιλογής του παίχτη σε περίπτωση που είναι σωστή. Επίσης, θέτει την λογική που πρέπει να ακολουθήσει το παιχνίδι σε περίπτωση που τελειώσουν οι ερωτήσεις.

Ξεκινώντας με την συνάρτηση Start στην οποία καλείται η συνάρτηση GenerateQuestion, όπου από τις διαθέσιμες ερωτήσεις που υπάρχουν επιλέγεται μία με τυχαίο τρόπο και θέτει το κείμενο του panel σε αυτό της ερώτησης. Στην συνέχεια, καλεί την συνάρτηση SetAnswers προκειμένου να χρησιμοποιηθούν οι κατάλληλες πιθανές απαντήσεις στο συγκεκριμένο ερώτημα και να οριστεί η σωστή απάντηση ανάμεσα στις επιλογές. Στην περίπτωση που δεν υπάρχουν διαθέσιμες ερωτήσεις καλείται η συνάρτηση EndOfQuestions η οποία απενεργοποιεί το παιχνίδι(Mini Game) και επαναφέρει το κύριο παιχνίδι στο αρχικό μενού.

Ακολουθεί η συνάρτηση Update στην οποία λαμβάνονται και αποθηκεύονται τα στοιχεία του χαρακτήρα του παίχτη που έχουν σχέση με την δύναμη(mana) για χρήση αργότερα. Γίνεται έλεγχος για το αν ο παίχτης έχει αποκτήσει όλη την δύναμή του πίσω απαντώντας σωστά στις ερωτήσεις και ταυτόχρονα αν η τιμή της μεταβλητής flag είναι ψευδής, που σημαίνει ότι δεν βρίσκεται στην αρχική έναρξη του παιχνιδιού, προκειμένου να απενεργοποιήσει το παιχνίδι και να επιστρέψει στην σκηνή του παιχνιδιού. Συνεχίζοντας, ακολουθεί η συνάρτηση ActivateMiniGame η οποία ενεργοποιεί το παιχνίδι. Προχωρώντας στις συναρτήσεις Correct και Wrong οι οποίες αφαιρούν την ερώτηση και τις πιθανές απαντήσεις της από την λίστα, επιβραβεύοντας ή τιμωρώντας τον παίχτη με βάση την επιλογή του. Ταυτόχρονα καλούν την συνάρτηση WaitForNext η οποία επιλέγει μια άλλη τυχαία ερώτηση από τις απομένουσες με ένα δευτερόλεπτο καθυστέρησης ώστε να εμφανιστεί το χρώμα της επιλογής του παίχτη(κόκκινο αν πρόκειται για λανθασμένη απάντηση και πράσινο για σωστή απάντηση).

4.8. Scramble Game

Στο παιχνίδι αυτό ο παίχτης πρέπει να σχηματίσει μια λέξη με τα γράμματα που του παρέχονται. Όπως και το προηγούμενο παιχνίδι δημιουργήθηκε σε ξεχωριστή σκηνή. Για αυτό το παιχνίδι επιλέχτηκε η χρήση ενός plugin με όνομα Playmaker (περισσότερα στα

[10],[11],[12]) το οποίο βοηθάει στην δημιουργία παιχνιδιών. Επιλέχτηκε λόγω της μεγάλης ποσότητας scripting που περιέχει το πρότζεκτ και επίσης λόγω της επιθυμίας για χρήση όσο το δυνατόν πιο πολλών διαφορετικών μεθόδων υλοποίησης παιχνιδιών.

Drag Letter object & Letter Slot object

Τα δύο αυτά αντικείμενα αποτελούν την βάση του παιχνιδιού. Το αντικείμενο Drag Letter θα χρησιμοποιηθεί ως template για τα γράμματα που θα μπορεί να μετακινήσει ο παίχτης και αντίστοιχα το αντικείμενο Letter Slot θα χρησιμοποιηθεί ως το template για τις θέσεις στις οποίες πρέπει να εκχωρηθούν τα γράμματα. Για την λειτουργία τους θα δημιουργηθεί μία νέα κατάσταση(state) στο FSM που ορίζει την λειτουργία που πρέπει να εκτελεστεί από το καθένα.

Ειδικότερα, για το αντικείμενο Drag Letter θα δημιουργηθεί μία νέα κατάσταση με όνομα On Drag και θα γίνεται μετάβαση(transition), εφόσον το αντικείμενο πρέπει, να μπορεί να αναγνωρίζει ποιες στιγμές επιχειρεί να το μετακινήσει ο παίχτης, με όνομα UI DRAG. Προκειμένου το αντικείμενο να μπορεί να μετακινηθεί πρέπει επιπλέον να ληφθούν τα απαραίτητα στοιχεία μέσω της πράξης(action) με όνομα UIGetLastPointerDataInfo από όπου είναι απαραίτητη η τιμή Delta, οπότε δημιουργείται νέα τιμή με όνομα Delta Drag. Στην συνέχεια, προστίθεται η πράξη Translate όπου αφαιρούνται οι επιλογές Per Second και Every Frame εφόσον δεν χρησιμεύουν και στην τιμή Vector γίνεται convert της τιμής Delta Drag σε Vector3 από Vector2 που ήταν αρχικά.

Όμοια, για την δημιουργία του Letter Slot ορίζεται η κατάσταση On Drop και χρησιμοποιείται η μετάβαση UI Drop η οποία ελέγχει αν υπάρχει κάποιο αντικείμενο πάνω σε αυτό. Επίσης, χρησιμοποιείται ξανά η πράξη UIGetLastPointerDataInfo προκειμένου να δημιουργηθεί μία μεταβλητή τύπου Pointer id με το όνομα Dragged Letter, με αποτέλεσμα το αντικείμενο να γίνεται αξιοποιήσιμο από την μεταβλητή Pointer Drag. Με αυτόν τον τρόπο μπορεί να γίνει ο έλεγχος μεταξύ των γραμμάτων των δύο αντικειμένων.

Answer & Shuffle Slots

Τα αντικείμενα αυτά καθορίζουν που θα βρίσκονται οι θέσεις για τις απαντήσεις και τα μπερδεμένα γράμματα αντίστοιχα. Ορίζοντας τις κατάλληλες μεταβλητές του ώστε να υπάρχει κενό μεταξύ των γραμμάτων και διατήρηση των διαστάσεων του αντικειμένου, τοποθετούνται τα κατάλληλα αντικείμενα ως παιδιά στο καθένα.

Στην συνέχεια για το Drag Letter χρησιμοποιείται ένα αντικείμενο εικόνας, προκειμένου να εμφανίζει το γράμμα στον παίχτη και δημιουργείται νέο καθολικό event με όνομα Set Letter, δημιουργώντας νέα κατάσταση με όνομα Set the Letter στην οποία θα οριστεί ως μετάβαση. Σε αυτή την κατάσταση δημιουργείται νέα πράξη με όνομα UITextSetText για να χρησιμοποιηθεί το γράμμα του αντικειμένου σε μεταγενέστερο στάδιο.

Word Manager

Το στοιχείο Word Manager είναι αυτό που χειρίζεται την διαδικασία χωρισμού των λέξεων σε γράμματα, τον ορισμό της σωστής σειράς της λέξης που πρέπει να δημιουργηθεί και την αλλαγή θέσης των γραμμάτων που χρησιμοποιεί ο παίχτης σε τυχαίες θέσεις.

Μεγάλο ενδιαφέρον έχει το FSM που πρέπει να δημιουργηθεί προκειμένου να εκτελεί όλες αυτές τις λειτουργίες.

Αρχικά στο Initial state χωρίζεται η λέξη με βάση τον διαχωριστή που επιλέχτηκε και στην συνέχεια μεταβαίνει στο loop των καταστάσεων Loop Word και Spawn Letter όπου για κάθε γράμμα που περιέχει η λέξη δημιουργείται νέο αντικείμενο έχοντας ως εικόνα το κάθε γράμμα. Όταν ολοκληρωθεί η διαδικασία αυτή γίνεται μετάβαση στο τελικό loop στο οποίο γίνεται τυχαία αλλαγή των θέσεων των γραμμάτων από την κατάσταση Loop Word Shuffle και δημιουργείται νέο αντικείμενο θέσης στο οποίο πρέπει να τοποθετηθεί το γράμμα μέσω της κατάστασης Spawn Letter 2.

Εφόσον έχουν ολοκληρωθεί οι παραπάνω διαδικασίες, ξεκινάει το παιχνίδι. Κάθε φορά που τοποθετείται το σωστό γράμμα στην κατάλληλη θέση γίνεται μετάβαση στην κατάσταση Add Point όπου όταν συλλεγούν οι απαραίτητοι πόντοι(ολοκληρώνοντας την λέξη επιτυχώς) τότε ολοκληρώνεται ο κύκλος αυτού του μίνι παιχνιδιού και επιστρέφει ο παίχτης στο στάδιο του κύριου παιχνιδιού που βρισκότανε.

4.9. The UI Elements

4.9.1. Mute Button

Ο κύριος λόγος ύπαρξης αυτού του κουμπιού είναι η διευκόλυνση του χρήστη, για να μην υπάρχει ανάγκη να πρέπει να αλληλεπιδράσει με το μενού. Αυτού του είδους η πρακτική συνήθως εμφανίζεται σε παιχνίδια κινητού για τους ακριβώς ίδιους λόγους και θεωρήθηκε αρκετά χρήσιμη με αποτέλεσμα να χρησιμοποιηθεί σε αυτό το πρότζεκτ.

FastMute.cs

Το script δέχεται δύο εικόνες, τις οποίες θα χρησιμοποιήσει στην συνέχεια για να υπάρξει αλλαγή της εικόνας όταν ο χρήστης πατήσει το κουμπί και κλείσει τον ήχο, αλλά και να επαναφέρει την αρχική εικόνα όταν ξαναπατήσει το κουμπί και επανέλθει ο ήχος. Πέρα από αυτές τις δύο εικόνες δέχεται ένα αντικείμενο τύπου κουμπιού το οποίο θα λαμβάνει υπόψιν τα κλικ του χρήστη, μία μεταβλητή τύπου bool με την οποία θα ελέγχεται η

κατάσταση του κουμπιού και μία μεταβλητή τύπου AudioSource με την οποία θα μπορεί να ελέγχει την κατάσταση του ήχου.

Αρχικά για να μην υπάρχουν πολλά διαφορετικά ίδια αντικείμενα στην σκηνή, θα πρέπει να δημιουργηθεί ένα στιγμιότυπο του script το οποίο θα παραμένει σταθερό στην σκηνή και θα διαγράφει οποιαδήποτε αντίγραφα δημιουργηθούν.

Στην συνάρτηση Awake γίνεται ακριβώς αυτή η διαδικασία, δηλαδή γίνεται έλεγχος αν υπάρχουν πάνω από ένα scripts με αυτό το όνομα και σε περίπτωση που υπάρχουν, διαγράφονται τα αντίγραφα, διατηρείται το αρχικό και ορίζεται ως αρχική εικόνα η εικόνα που έχει το αντικείμενο στην σκηνή.

Στην συνέχεια στην συνάρτηση ButtonClicked γίνεται έλεγχος αν η μεταβλητή bool είναι αληθής. Αν είναι αληθής και ο χρήστης πατήσει το κουμπί τότε αλλάζει η εικόνα του κουμπιού, η μεταβλητή αλλάζει σε false και η πηγή ήχου σταματάει να παράγει ήχο. Αλλιώς αν η μεταβλητή bool είναι ψευδής και πατήσει το κουμπί ο χρήστης, τότε η εικόνα αλλάζει στην αρχική, αλλάζει η τιμή του σε true και η πηγή αναπαραγωγής ήχου αρχίζει να αναμεταδίδει ήχο.

Να σημειωθεί ότι όταν σταματάει να ακούγεται ο ήχος, αυτό δεν σημαίνει ότι όταν επανέλθει, θα βρίσκεται στο σημείο που βρισκόταν όταν σταμάτησε να αναπαράγεται.

4.9.2. SFX Manager

Η λειτουργία του είναι να προσθέτει ήχο όποτε κρίνεται απαραίτητο, για την δημιουργία καλύτερης αίσθησης του περιβάλλοντος και των αντικειμένων, καθώς και να παρέχει στον χρήστη μία ηχητική ένδειξη για το αν χρησιμοποίησε ένα αντικείμενο ή έχασε και πρέπει να ξαναρχίσει το επίπεδο από την αρχή. Στο συγκεκριμένο πρότζεκτ θα χρησιμοποιηθεί αυτή η μέθοδος, κάθε φορά που ο χρήστης χρησιμοποιεί ένα αντικείμενο που του επαναφέρει την υγεία(health - HP) ή τη δύναμη(mana - MP).Για να μην χρησιμοποιείται συνέχεια το ίδιο clip ήχου και για να γίνει λίγο πιο ενδιαφέρον, προστέθηκε ένα τυχαίο στοιχείο για τον ήχο ο οποίος χρησιμοποιείται κάθε φορά.

SFXManager.cs

Στο script αυτό αρχικά, εφόσον θέλουμε να είναι μοναδικό σε οποιαδήποτε κατάσταση του παιχνιδιού βρίσκεται ο χρήστης, θα δημιουργηθεί ένα στιγμιότυπό του όπου στην συνάρτηση Awake θα ελέγχει την μοναδικότητά του και θα δρα ανάλογα(βλέπε FastMute script). Επίσης, θα δέχεται μία πηγή αναπαραγωγής ήχου audioSrc για να μπορεί να αναπαράγει ήχο όταν χρειάζεται, αξιοποιώντας έναν πίνακα Sounds με όλους τους πιθανούς ήχους.

Στην συνάρτηση Start τίθεται ως sfx το ίδιο στιγμιότυπο του script. Ως πηγή αναπαραγωγής ήχου επιλέγεται να χρησιμοποιηθεί η μέθοδος με όνομα GetComponent η οποία βρίσκει το αντικείμενο που της επισημάνθηκε, χωρίς να είναι απαραίτητο να της δοθεί

αναφορά για αυτό από τον επιθεωρητή κατά την διάρκεια δημιουργίας του παιχνιδιού. Στην συνέχεια φορτώνονται στον πίνακα Sounds όλοι οι ήχοι οι οποίοι βρίσκονται στον φάκελο με όνομα SFXSounds. Τέλος, η συνάρτηση PlaySound είναι αυτή που θα καλεστεί για την αναπαραγωγή του ήχου, στην οποία η μεταβλητή randomSound παίρνει μία τυχαία τιμή στο διάστημα [0,2] και στην συνέχεια καλείται η πηγή ήχου με υπό ρουτίνα την PlayOneShot η οποία αναπαριστά το κομμάτι ήχου μόνο μία φορά, επιλέγοντας έναν τυχαίο ήχο από τον πίνακα Sounds.

Στο παράθυρο επιθεωρητή, πρέπει στο στοιχείο της πηγής (Audio Source) να μην χρησιμοποιηθεί η επιλογή επανάληψης Loop εφόσον ο ήχος πρέπει να χρησιμοποιηθεί μόνο μία φορά. Επίσης, σημαντικό στοιχείο για την σωστή αναγνώριση των ήχων και για την ένταξή τους στον αντίστοιχο πίνακα είναι ότι θα πρέπει να βρίσκονται στον φάκελο με το όνομα Resources αλλιώς δεν θα γίνονται ορατά στο Unity οδηγώντας σε σφάλμα. Αυτό συμβαίνει λόγω του τρόπου κατασκευής του Unity ως Game Engine. Για να μπορέσει δε, να χρησιμοποιήσει το Unity παραπάνω από ένα clip ήχου πρέπει αυτά να βρίσκονται στον παραπάνω φάκελο.

4.9.3. Pause Menu

Το κύριο μενού αποτελεί αναντικατάστατο στοιχείο για ένα παιχνίδι, οποιουδήποτε είδους και μεγέθους. Σε αυτό ο χρήστης έχει την δυνατότητα να επιλέξει:

- Αν θέλει να αλλάξει ορισμένες από τις ρυθμίσεις του παιχνιδιού.
- Μπορεί να το χρησιμοποιήσει για να μεταβεί σε κάποια προηγούμενη κατάσταση παιχνιδιού που έχει σώσει.
- Μπορεί να σταματήσει την ροή του παιχνιδιού όποτε αυτός θελήσει και
- φυσικά μπορεί να κλείσει την εφαρμογή.

Όλες αυτές οι δυνατότητες παρέχονται από ένα και μοναδικό script το οποίο περιέχει όλη την λειτουργικότητα που χρειάζεται, από την εμφάνιση του μενού στην οθόνη έως τον έλεγχο των διαφορετικών μενού διαφορετικών αντικειμένων όπως π.χ. είναι το μενού υπεύθυνο για το τέλος του παιχνιδιού. Παρακάτω, θα γίνει εκτενής ανάλυση της λειτουργίας του και των στοιχείων από τα οποία αποτελείται για την πλήρη κατανόησή του.

Main Menu Panel: Το μενού που ενεργοποιείται όταν ο χρήστης πατήσει το κουμπί Escape στο πληκτρολόγιο. Περιέχει το κουμπί με όνομα Resume με το οποίο μπορεί ο χρήστης να κλείσει την καρτέλα και να επιστρέψει στο παιχνίδι συνεχίζοντας από την στιγμή που είχε ενεργοποιήσει το μενού. Στην συνέχεια ακολουθεί το κουμπί Restart το οποίο παρέχει την δυνατότητα επαναφοράς του παιχνιδιού στην αρχική του κατάσταση. Ακολουθεί το κουμπί Audio το οποίο ενεργοποιεί την καρτέλα Audio Menu από όπου μπορεί να γίνει αλλαγή της έντασης του ήχου. Τέλος, υπάρχει το κουμπί Exit με το οποίο μπορεί να κλείσει την εφαρμογή.

Audio Menu: Σε αυτήν την καρτέλα ο χρήστης, μπορεί να προσαρμόσει την ένταση του ήχου στο επίπεδο που είναι της αρεσκείας του, με την χρήση της ρυθμιζόμενης μπάρας η οποία στο άκρο της έχει μία ένδειξη για την τωρινή ένταση. Για να σωθεί η αλλαγή της έντασης υπάρχει το κουμπί Save και τέλος το κουμπί Back το οποίο απενεργοποιεί την καρτέλα και επιστρέφει στο Main Menu.

Save/Load Menu: Το μενού που εμφανίζεται όταν όταν ο χρήστης πατήσει το κουμπί Space, για να μπορέσει να αποθηκεύσει το παιχνίδι με την χρήση του κουμπιού Save. Για να μπορέσει να φορτώσει την κατάσταση του παιχνιδιού παρέχεται το κουμπί Load.

Game Over Screen: Η καρτέλα που εμφανίζεται όταν ο χαρακτήρας που βρίσκεται υπό τον έλεγχο του χρήστη, χάσει όλους τους πόντους υγείας και ουσιαστικά χάσει το παιχνίδι. Του δίνεται η ευκαιρία να ξεκινήσει ξανά το παιχνίδι από την αρχή, μέσω του κουμπιού Restart και η επιλογή του να κλείσει το παιχνίδι μέσω του κουμπιού Exit.

Menu Manager.cs

Ξεκινώντας από την συνάρτηση Awake αποφασίστηκε να οριστεί αυτή ως protected virtual για να μπορέσει να χρησιμοποιηθεί από άλλες κλάσεις οι οποίες κληρονομούν την κλάση MenuManager. Στην συνάρτηση αρχικά όλα τα μενού και καρτέλες που χρησιμοποιούνται γίνονται μη ορατά για τον χρήστη όταν αρχίζει το παιχνίδι, στην συνέχεια αποθηκεύονται οι συντεταγμένες των θέσεων του παίχτη και τέλος τίθεται ο χρόνος του παιχνιδιού ως 1. Ο χρόνος του παιχνιδιού μπορεί να χρησιμοποιηθεί για να επηρεάσει την ροή του παιχνιδιού, ανάλογα με την τιμή που του ανατίθεται. Αν γίνει 0 τότε ο χρόνος του παιχνιδιού σταματάει, με αποτέλεσμα να μην μπορεί να γίνει οποιαδήποτε ενέργεια. Αντίθετα αν γίνει 1 τότε ο χρόνος του παιχνιδιού κυλάει κανονικά επιτρέποντας κάθε ενέργεια.

Όταν καλείται η συνάρτηση SaveOrLoad για να μπορέσει ο χρήστης να έχει πρόσβαση στο μενού αποθήκευσης, γίνεται το κατάλληλο μενού ορατό και τίθεται ο χρόνος παιχνιδιού ως 0. Στην συνέχεια ο χρήστης πρέπει να επιλέξει ανάμεσα στα Save και Load κουμπιά. Αν επιλέξει να αποθηκεύσει την κατάσταση του παιχνιδιού του, τότε θα καλεστεί η συνάρτηση SaveButton η οποία χρησιμοποιεί την συνάρτηση Save από την κλάση SaveLoadSystem. Επιλέχτηκε αυτός ο τρόπος, για να μειωθεί το μέγεθος του περιττού επαναλαμβανόμενου κώδικα και για να επιδείξει την δύναμη της κληρονομικότητας, κάνοντας τον κώδικα πιο αποδοτικό και εύκολο στην κατανόηση(ο τρόπος λειτουργίας του Save θα επεξηγηθεί παρακάτω). Αφού τελειώσει η διαδικασία ακολουθεί η απόκρυψη του μενού και η επιστροφή της τιμής του χρόνου σε 1. Όμοια, αν επιλέξει να επιστρέψει σε μία προηγούμενη κατάσταση του παιχνιδιού τότε επιλέγοντας το κουμπί Load θα εκτελεστεί η συνάρτηση LoadButton στην οποία γίνεται η επίκληση του στιγμιότυπου του παιχνιδιού μέσω κληρονομικότητας από την κληρονομούμενη κλάση. Μετά ολοκληρώνεται η εκτέλεσή της και κρύβεται το μενού επιστρέφοντας την τιμή του χρόνου σε 1.

Ακολουθεί η συνάρτηση Update η οποία είναι ειδικά φτιαγμένη και παρέχεται από το Unity για να χρησιμοποιείται κάθε frame του παιχνιδιού, όσο αυτό βρίσκεται σε λειτουργία ή ο χρόνος του παιχνιδιού δεν έχει σταματήσει. Η χρήση αυτής της συνάρτησης είναι, κατά κύριο λόγο, η αναμονή για κάποια ενέργεια από τον χρήστη, όπως για παράδειγμα το πάτημα ενός κουμπιού του πληκτρολογίου. Συγκεκριμένα, γίνεται έλεγχος αν το κουμπί που πατήθηκε είναι το Escape όπου εκτελείται η συνάρτηση Pause, η οποία εμφανίζει το κύριο μενού και σε περίπτωση που ο χαρακτήρας του χρήστη έχει χάσει και έχει εμφανιστεί το μενού Game Over τότε το κρύβει, για να αποφευχθούν προβλήματα προτεραιότητας μεταξύ

των δύο μενού, σταματώντας ταυτόχρονα τον χρόνο του παιχνιδιού. Εναλλακτικά, αν ο χρήστης πατήσει το πλήκτρο Space τότε καλείται η συνάρτηση SaveOrLoad η οποία έχει αναλυθεί παραπάνω.

Η συνάρτηση Restart με την ενεργοποίησή της θα χρησιμοποιήσει την βιβλιοθήκη SceneManagement του Unity, αξιοποιώντας την κλάση SceneManager η οποία μπορεί να ελέγχει σκηνές κατά την διάρκεια του τρεξίματος του παιχνιδιού. Στο συγκεκριμένο πρότζεκτ, επειδή στην σκηνή με το αρχικό μενού γίνεται φόρτωση παραπάνω από μίας σκηνής, για την ομαλή λειτουργία του παιχνιδιού, αποφασίστηκε ο χρήστης να παραπέμπεται στο αρχικό μενού παρά στην αρχή του πρώτου επιπέδου, γιατί σε αυτή την περίπτωση δεν θα έχουν φορτώσει απαραίτητα στοιχεία, τα οποία θα προσπαθεί να επισημοποιήσει το σύστημα ανεπιτυχώς οδηγώντας σε σφάλματα. Στη συνέχεια πρέπει να γίνει λήψη των στοιχείων του χαρακτήρα του χρήστη και να επανέλθουν οι τιμές που είχαν τα στοιχεία αυτά όταν βρίσκονταν στην αρχή του παιχνιδιού. Για τον σκοπό αυτό θα χρησιμοποιηθεί η συνάρτηση GetComponentInChilden η οποία βρίσκει το αντικείμενο που της δόθηκε σαν όρισμα, λαμβάνοντας υπόψιν τα στοιχεία τα οποία περιέχονται στο αντικείμενο και αφού εντοπίσει τα κατάλληλα scripts τότε εκτελεί την συνάρτηση Start και σε αυτά, επαναφέροντας τις αρχικές τιμές τους. Συνεχίζοντας, πρέπει να γίνει επαναφορά όχι μόνο των στοιχείων του χαρακτήρα αλλά και της θέσης στην οποία βρίσκεται. Αυτό επιτυγχάνεται μέσω της συνάρτησης Awake όπου κρατήθηκαν οι αρχικές τιμές της θέσης του για να χρησιμοποιηθούν από αυτήν την συνάρτηση και να τεθούν ως οι νέες συντεταγμένες του χαρακτήρα.

Τέλος, σε περίπτωση που η ενεργοποίηση της συνάρτησης έχει προέλθει από το κύριο μενού ή από την καρτέλα Game Over, εφόσον χρησιμοποιούν τον ίδιο κώδικα στα κουμπιά τους, κρύβονται και τα δύο μενού σε κάθε περίπτωση.

Στην συνέχεια ακολουθεί η συνάρτηση AudioSettings, η οποία κρύβει το κύριο μενού για να μην υπάρχει υπερκάλυψη των στοιχείων και ενεργοποιεί το υπεύθυνο μενού για την προσαρμογή του ήχου. Μετά την επιλογή των ρυθμίσεων, ο χρήστης θα πατήσει το κουμπί Back το οποίο θα εκτελέσει την συνάρτηση Back όπου θα κρύψει το μενού για την ρύθμιση ήχου και θα επαναφέρει το κύριο μενού. Ακολουθεί η συνάρτηση Exit η οποία θα καλεστεί όταν ο χρήστης πατήσει το κουμπί Exit, προκειμένου να τερματίσει το παιχνίδι. Επειδή ο τερματισμός του παιχνιδιού δεν μπορεί να προσομοιωθεί από το Unity, έγινε χρήση της εντολής Debug.Log η οποία εμφανίζει ένα μήνυμα με το κατάλληλο κείμενο για την κατάσταση του παιχνιδιού. Τέλος, η συνάρτηση GameOver καλείται όταν ο χρήστης χάσει το παιχνίδι και πρέπει να αρχίσει από την αρχή.

4.9.4. Player UI

Σε αυτό το κομμάτι, θα δοθεί μεγαλύτερη έμφαση στον τρόπο κατασκευής και τοποθέτησης των στοιχείων που απαρτίζουν το σύστημα των πόρων του χαρακτήρα του χρήστη. Τα λειτουργικά κομμάτια των script που έχει το σύστημα αυτό, βρίσκονται τοποθετημένα στο αντικείμενο του χαρακτήρα, οπότε θα γίνει περαιτέρω ανάλυση για αυτά στην παράγραφο επεξήγησης των στοιχείων και λειτουργιών του χαρακτήρα που χειρίζεται ο χρήστης.

Αρχικά, στο αντικείμενο PlayerUI της σκηνής δημιουργείται ένα αντικείμενο τύπου Panel ως παιδί σε αυτό και ως εικόνα αυτού χρησιμοποιείται το περίγραμμα που θα

απαιτείται ως η βάση για την διαδικασία. Στην συνέχεια, προσαρμόζεται στο μέγεθος που είναι επιθυμητό και γίνεται πλήρως ορατό αλλάζοντας το A κανάλι του στην μέγιστη τιμή.

Έχοντας τελειώσει με την βάση, γίνεται αντιγραφή του αντικειμένου ως δεύτερο παιδί, αλλάζοντας την εικόνα στο στοιχείο που θα αποτελέσει την μπάρα υγείας με όνομα FrontHealthbar. Αλλάζοντας το μέγεθός της ώστε να βρίσκεται στα πλαίσια της εικόνας του πρώτου παιδιού αντικειμένου και τοποθετώντας το δεύτερο αντικείμενο πάνω από το πρώτο για να έχει προτεραιότητα εμφάνισης η εικόνα της βάσης. Με αυτό το κομμάτι ολοκληρωμένο γίνεται αντιγραφή του αντικειμένου FrontHealthbar, αλλαγή του ονόματός του σε BackHealthbar και της εικόνας του σε αυτή χωρίς χρώμα εφόσον θα γίνει η ρύθμισή του μέσω κώδικα στο script, τοποθετώντας το πρώτο στο ιεραρχικό επίπεδο για να έχουν προτεραιότητα τα παρακάτω από αυτό στοιχεία. Τέλος, αντιγράφεται το αντικείμενο BackHealthbar μετονομάζοντας το σε HealthBackground, αλλάζοντας το χρώμα του σε μαύρο και τοποθετώντας το ως πρώτο παιδί στην ιεραρχία.

Για να μπορέσει η εικόνα να αλλάξει όταν ο χαρακτήρας δέχεται επιθέσεις πρέπει να αλλάξει ο τύπος των εικόνων σε γέμισμα(Filled) και ο τρόπος γεμίσματος σε οριζόντιο αποκλείοντας την εικόνα HealthbarBackground. Στην συνέχεια για να παρέχεται στον χρήστη η πληροφορία της κατάστασης του χαρακτήρα, δημιουργήθηκε ένα επιπλέον αντικείμενο τύπου text με όνομα HpValue το οποίο τοποθετήθηκε τελευταίο ώστε να εμφανίζεται πάνω από τα υπόλοιπα στοιχεία.

Η ίδια διαδικασία ακολουθείται για την κατασκευή της μπάρας δύναμης(MP), αλλάζοντας το αρχικό χρώμα σε μπλε όπως συνηθίζεται σε άλλα παιχνίδια.

4.9.5. DamagePopup.cs

Σε αυτό το script θα δημιουργηθεί ένα αντικείμενο τύπου UI, το οποίο θα προσφέρει οπτική πληροφορία για τον παίχτη όταν χάνει πόντους υγείας ή δύναμης ή αντίστοιχα την επαναφορά αυτών, καθώς και ορισμένες ενέργειες που εκτελούνται στο παρασκήνιο και σχετίζονται με αυτόν.

Έτσι, αρχικά δημιουργείται μια συνάρτηση τύπου DamagePopup η οποία φτιάχνει και επιστρέφει ένα τέτοιο αντικείμενο με βάση την χρήση που πρέπει να πραγματοποιεί, καλώντας διαφορετικές συναρτήσεις για κάθε περίπτωση. Στην συνάρτηση Awake λαμβάνεται το στοιχείο TextMeshPro το οποίο θα αλλάξει τις παρακάτω συναρτήσεις προκειμένου να αναδείξει σωστά τα απαραίτητα στοιχεία.

Στην συνάρτηση SetupDamage μετατρέπεται το text στον αριθμό της ζημιάς των πόντων υγείας του αντικειμένου και αλλάζει το χρώμα του σε κόκκινο εφόσον πρόκειται για αριθμό ζημιάς. Ακολουθεί η συνάρτηση SetupHealth η οποία με την ίδια λογική αλλάζει το χρώμα σε πράσινο για να επιδείξει την επαναφορά πόντων υγείας. Στην συνέχεια ακολουθεί η συνάρτηση SetupMana η οποία ασχολείται με το στοιχείο δύναμης του χαρακτήρα του παίχτη, αλλάζει την τιμή του text στον αριθμό της δύναμης που ανακτήθηκε καθώς επίσης και το χρώμα του κειμένου σε μπλε διευκρινίζοντας ότι πρόκειται για το στοιχείο της δύναμης του χαρακτήρα. Έτσι γίνεται καλύτερα αντιληπτό από τον παίχτη ο τύπος του στοιχείου που ανακτάται(υγείας ή δύναμης).

Προχωρώντας, ακολουθούν οι συναρτήσεις ConfirmActionText και InvalidActionText οι οποίες εμφανίζουν με κατάλληλο χρώμα την ενέργεια του παίχτη.

Τέλος, στην συνάρτηση Update ορίζεται η ταχύτητα με την οποία θα ανεβαίνει το text στοιχείο και επίσης ορίζεται η θέση στην οποία θα εμφανιστεί.

4.10. Player Character

4.10.1. Player Healthbar.cs

Ξεκινώντας από την συνάρτηση Start τίθεται ως αρχικό health η μέγιστη δυνατή τιμή. Έτσι κάθε φορά που ξεκινάει το παιχνίδι ο χαρακτήρας του χρήστη έχει πάντα πλήρη την μπάρα υγείας του. Στη συνέχεια, ακολουθεί η συνάρτηση CurrentHealth η οποία επιστρέφει την τιμή της υγείας του χαρακτήρα. Η χρήση αυτής της συνάρτησης βρίσκεται στην αξιοποίησης της τιμής που επιστρέφει από διαφορετικά scripts.

Στην συνάρτηση Update αρχικά θέτονται τα όρια που μπορεί να πάρει η μεταβλητή health μέσω της συνάρτησης Clamp από την κλάση Mathf, επίσης καλούνται οι συναρτήσεις CurrentHealth και UpdateHealthUI εφόσον πρέπει να υπάρχει δυνατότητα αλλαγής των τιμών τους κατά την εξέλιξη του παιχνιδιού.

Η συνάρτηση UpdateHealthUI ορίζει τις τοπικές μεταβλητές fillFront και fillBack με τις οποίες παίρνει τις τιμές γεμίσματος από τα αντικείμενα front και backHealthbars. Στην συνέχεια, για να μπορέσουν να γίνουν εύκολα έλεγχοι, ορίστηκε η μεταβλητή hFraction ως η τωρινή τιμή της υγείας του χαρακτήρα προς την μέγιστη τιμή που μπορεί να έχει, οδηγώντας σε μία τιμή από 0 έως 1.

Συνεχίζοντας, αρχικοποιείται και αλλάζει σε τύπο string η μεταβλητή textNumber για να μπορέσει να εμφανιστεί σαν text.

Ακολουθεί έλεγχος. Σε περίπτωση που η τιμή του fillBack είναι μεγαλύτερη από το hFraction τότε ο χαρακτήρας έχει χάσει υγεία και πρέπει η εικόνα της πρώτης μπάρας να συρρικνωθεί, καθώς το χρώμα της πίσω μπάρας αλλάζει σε κόκκινο, για να δειχθεί ότι ο χαρακτήρας δέχτηκε ζημιά. Στην συνέχεια πρέπει να τεθεί ένα χρονικό διάστημα για το οποίο θα περιμένει η πίσω μπάρα για να "κυνηγήσει" την μπροστινή μπάρα. Για να επιτευχθεί αυτό θα χρησιμοποιηθεί η συνάρτηση Lerp της κλάσης Mathf από την μεταβλητή fillBack προς την hFraction με τιμή καθυστέρησης percentComplete . Η διαδικασία επαναλαμβάνεται για την περίπτωση που ο χαρακτήρας έχει "αναρρώσει" με τη διαφορά του χρώματος σε πράσινο για το ποσό που επανήλθε και σε αυτή την περίπτωση πρέπει να αυξηθεί η τιμή της frontHealthbar για να φτάσει την τιμή της backHealthbar.

Στην συνέχεια του script ακολουθεί η συνάρτηση TakeDamage, η οποία δέχεται ως όρισμα έναν αριθμό τύπου float τον οποίο αφαιρεί από την υγεία του χαρακτήρα όταν δέχεται μία επίθεση. Για να προσφέρει το παιχνίδι ένδειξη ότι ο χαρακτήρας χτυπήθηκε από μια επίθεση, κάθε φορά δημιουργείται ένα αντικείμενο text με τον αριθμό της επίθεσης το οποίο

ανεβαίνει ως προς τον ψ άξονα και εξαφανίζεται μετά από προκαθορισμένο χρονικό διάστημα. Σε αυτή την περίπτωση έχει κόκκινο χρώμα, αλλιώς σε περίπτωση που επαναφέρεται υγεία ή δύναμη το χρώμα αλλάζει σε πράσινο και μπλε αντίστοιχα(παραπάνω για την κατασκευή και την λειτουργία τους παρακάτω). Έπειτα, ελέγχεται αν σε κάποιο από τα χτυπήματα η υγεία του χαρακτήρα φτάσει στο 0. Αν ισχύει τότε καλείται η GameOver οθόνη του παιχνιδιού .

Ακολουθεί η συνάρτηση RestoreHealth η οποία εκτελεί τις ίδιες λειτουργίες αλλά για την αντίθετη περίπτωση, αυτή που ο χαρακτήρας "αναρρώνει".

Οι συναρτήσεις που ακολουθούν, αποτελούν κομμάτι του συστήματος saving της κατάστασης του παιχνιδιού. Θα αναλυθεί το σύστημα σε επόμενη παράγραφό, αλλά τα ακόλουθα στοιχεία πρέπει να βρίσκονται στα scripts για τα οποία πρέπει να σωθεί η κατάσταση για κάποιο ή κάποια από αυτά.

Ειδικότερα, δημιουργείται μία δομή SaveData με τα στοιχεία τα οποία πρέπει να διατηρηθούν(σε αυτή την περίπτωση health και maxhealth). Στην συνέχεια, δημιουργείται το αντικείμενο SaveState το οποίο επιστρέφει ένα στιγμιότυπο των τιμών αυτών. Τέλος, για να ολοκληρωθεί η διαδικασία διαβάζονται οι τιμές από το αρχείο στο οποίο έχουν σωθεί και αντικαθιστούν τις αρχικές τιμές που θα είχε το παιχνίδι αν άρχιζε ξανά από την αρχή. Το σύστημα αυτό έχει πολλές δυνατότητες, εφόσον μόνο με την δημιουργία τριών scripts μπορεί να σωθεί οποιοδήποτε αντικείμενο χωρίς να δίνεται σημασία στη δομή ή την πολυπλοκότητα που υπάρχει σε οποιαδήποτε σκηνή. Για να λειτουργήσει ένα τέτοιο σύστημα, υπάρχουν ορισμένοι κανόνες στους οποίους πρέπει να υπακούσουν τα εν λόγω αντικείμενα για να μπορέσουν να είναι μέρος αυτής της διαδικασίας, όπως η υλοποίηση αυτών των συναρτήσεων σε κάθε αντικείμενο ξεχωριστά και η χρήση διαφορετικών IDs για να ξεχωρίζουν μεταξύ τους σε περίπτωση που επαναλαμβάνονται μέσα στην ίδια σκηνή κ.λ.π.

4.10.2. Player Manabar.cs

Αρχικά, στην συνάρτηση Start γίνεται η ανάθεση του στοιχείου της δύναμης στην μέγιστη τιμή της και αποθηκεύεται το όνομα της σκηνής που βρίσκεται ο παίχτης προκειμένου να χρησιμοποιηθεί αργότερα.

Ακολουθούν οι συναρτήσεις SetMana και SetMaxMana οι οποίες επιστρέφουν τις τιμές της δύναμης και την μέγιστης τιμής της για να χρησιμοποιηθούν από άλλα scripts.

Στην Update συνάρτηση ορίζονται τα όρια που μπορεί να πάρει η τιμή της δύναμης, γίνεται έλεγχος αν έχει πατηθεί το πλήκτρο P και ταυτόχρονα αν η τιμή της δύναμης δεν είναι μηδέν. Τότε αποθηκεύεται το όνομα της σκηνής σε μία μεταβλητή και καλείται η συνάρτηση Save από το script του SaveLoadSystem(το οποίο θα αναλυθεί παρακάτω), αξιοποιείται η λειτουργία εύρεσης αντικειμένου με συγκεκριμένο όνομα και ξεκινάει το Mini Game.

Στην συνέχεια ακολουθεί η συνάρτηση UpdateManaUI. Η λειτουργία και υλοποίηση αυτής είναι ίδια με το αντίστοιχο στοιχείο του PlayerHealthbar script. Ακολουθεί η συνάρτηση LoseMana στην οποία αφαιρείται η δύναμη που χρησιμοποίησε ο παίχτης, με ακριαίο τρόπο, για να υπάρχει καθαρή ένδειξη της δύναμης που χρησιμοποιείται. Κατόπιν ακολουθεί η συνάρτηση RestoreMana στην οποία ο παίχτης ανανεώνει τους πόντους δύναμής του, με την εμφάνιση ενός στοιχείου Popup το οποίο δείχνει το ποσό που ανανεώθηκε. Προχωρώντας, η συνάρτηση GetCurrentSceneName επιστρέφει το όνομα της σκηνής που βρίσκεται ο χαρακτήρας του παίχτη για την χρήση του σε άλλα scripts. Τέλος, ακολουθούν

οι απαραίτητες συναρτήσεις για την αποθήκευση και ανάκτηση των τιμών της δύναμης από το σύστημα SavingSystem.

4.10.3. PlayerController.cs

Η κύρια λειτουργία αυτού του script είναι να προσθέσει το στοιχείο της κίνησης στον χαρακτήρα που χειρίζεται ο παίχτης. Εφόσον το παιχνίδι χειρίζεται έναν παίχτη κατά την διάρκεια της εκτέλεσής του και όχι πολλούς, το script αυτό πρέπει να είναι μοναδικό.

Συνεπώς, στην συνάρτηση Awake μετατρέπεται σε script τύπου Singleton για να τηρεί τις προδιαγραφές του παιχνιδιού.

Συνεχίζοντας, στην συνάρτηση Start λαμβάνεται ως στοιχείο το αντικείμενο Rigidbody2D το οποίο θα χρησιμοποιηθεί για την κίνηση. Ακολουθεί η συνάρτηση Update στην οποία ορίζονται οι κινήσεις που μπορεί να εκτελέσει ο χαρακτήρας του παίχτη και γίνεται έλεγχος στα εξής:

- αν δεν υπάρχει μεγάλη διαφορά μεταξύ των τιμών [0,moveH] και
- αν ο χρόνος του παιχνιδιού δεν έχει σταματήσει,

τότε αν η τιμή του moveH είναι αρνητική (δηλαδή ο χαρακτήρας κινείται προς τα αριστερά), το sprite του χαρακτήρα αλλάζει κατεύθυνση προς τα αριστερά προκειμένου να είναι συμφασικός με την κατεύθυνση της κίνησης που έχει. Τελικά, ορίζεται ο περιορισμός μεταξύ των στρωμάτων(layers) του χαρακτήρα με τα projectiles που δημιουργεί, προκειμένου να επιτεθεί στους εχθρούς του. Επιλέχτηκε ο τρόπος αυτός για την αποφυγή του προβλήματος της λανθασμένης μικρής μετατόπισης του χαρακτήρα προς τα πίσω λόγω του σημείου της δημιουργίας των projectiles.

Στην συνέχεια ακολουθεί η συνάρτηση FixedUpdate η οποία μετατοπίζει τον χαρακτήρα με βάση τις δύο κατευθύνσεις που ορίστηκαν στην συνάρτηση Update και η συνάρτηση SetSpeed η οποία ορίζει την ταχύτητα με την οποία θα κινηθεί.

Όσο αφορά στο πιο κομμάτι πρέπει να σωθεί, θεωρήθηκε ότι αυτό είναι οι συντεταγμένες που απέκτησε ο χαρακτήρας του παίχτη όταν αυτός εκτέλεσε την διαδικασία αποθήκευσης της κατάστασης του παιχνιδιού. Επομένως, μέσω των συναρτήσεων και στοιχείων SaveState, LoadState και SaveData που παρέχει το σύστημα αποθήκευσης, τα απαραίτητα στοιχεία αποθηκεύονται και αργότερα ανακτώνται αντικαθιστώντας τα παλαιά στοιχεία με αποτέλεσμα ο χαρακτήρας να μεταφέρεται στο σημείο το οποίο βρισκόταν κατά την διάρκεια της προηγούμενης εκτέλεσης του παιχνιδιού.

4.10.4. PlayerInteract.cs

Σε αυτό το script θα γίνει ο χειρισμός των διάφορων αλληλεπιδράσιμων αντικειμένων του παιχνιδιού από τον παίχτη. Ξεκινώντας, με την συνάρτηση Update γίνεται ο έλεγχος αν έχει πατηθεί το κουμπί για την αλληλεπίδραση του παίχτη με το αντικείμενο, καθώς και αν το αντικείμενο βρίσκεται αρκετά κοντά για την ενέργεια αυτή. Αν ναι τότε το αντικείμενο προστίθεται στο inventory του παίχτη όπως είναι για παράδειγμα ένα αντικείμενο κλειδί για να ανοίξει μία κλειδωμένη πόρτα. Επειδή αυτή η λογική χρησιμοποιείται και από αντικείμενα τα οποία κανονικά δεν θα μπορούσαν να δράσουν λειτουργικά ως κλειδιά(όπως είναι τα αντικείμενα πορτών) προστέθηκε ως checkbox σε κάθε αντικείμενο για το αν προορίζονται

για τοποθέτηση στο inventory του χρήστη ή πρέπει να παραμείνουν στην θέση τους όπως οι πόρτες. Συνεχίζοντας, εφόσον πρόκειται για αντικείμενο πόρτας πρέπει να γίνει έλεγχος για το αν μπορεί να ανοιχτεί και αν ναι, για το αν είναι κλειδωμένο που σημαίνει ότι θα χρειαστεί κάποιο κλειδί για να ανοίξει. Προκειμένου να ανοίξει γίνεται έλεγχος για την ύπαρξη του απαιτούμενου αντικειμένου σε όλο το inventory του παίχτη. Αν βρεθεί τότε το αντικείμενο δεν θεωρείται πλέον κλειδωμένο και ανοίγει(ή στην περίπτωση αυτού του πρότζεκτ εξαφανίζεται) ανοίγοντας τον δρόμο για τον παίχτη και καταστρέφοντας το απαιτούμενο κλειδί από το inventory του χαρακτήρα του παίχτη εφόσον δεν θα υπάρχει άλλη χρήση για αυτό στην εξέλιξη του παιχνιδιού.

Ακολουθούν οι συναρτήσεις OnTriggerEnter2D και OnTriggerExit2D οι οποίες ελέγχουν αν το αντικείμενο που εισέρχεται στην ζώνη εμβέλειας έχει την απαραίτητη ταμπέλα(αν είναι δηλαδή αλληλεπιδράσιμο αντικείμενο). Τότε παρέχει τα στοιχεία του (αν πρόκειται για πόρτα ή κλειδί) για τον καθορισμό της σωστής ενέργειας μεταξύ παίχτη και αντικειμένου. Αν ο παίχτης εξέλθει από την ζώνη εμβέλειας αλληλεπίδρασης με το αντικείμενο, χωρίς να το χρησιμοποιήσει τότε αλλάζει η τιμή της μεταβλητής επιθεώρησης του τωρινού αντικειμένου σε κενή.

4.10.5. Inventory.cs

Καταρχήν, ο ρόλος της συνάρτησης AddItem είναι να προσθέσει ένα αντικείμενο στο inventory του παίχτη εφόσον υπάρχει χώρος. Ακολουθεί έλεγχος για αν το αντικείμενο έχει προστεθεί στο inventory του παίχτη.

Στην συνέχεια ακολουθεί η συνάρτηση FindItem η οποία ελέγχει αν το απαιτούμενο αντικείμενο βρίσκεται σε κάποια από τις θέσεις του inventory του παίχτη, επιστρέφοντας κατάλληλο αποτέλεσμα bool σε κάθε περίπτωση. Ακολουθεί η συνάρτηση FindKeyItem όπου σε περίπτωση που το αντικείμενο υπάρχει μέσα στο inventory επιστρέφει το αντικείμενο.

Τέλος, στην συνάρτηση RemoveItem γίνεται έλεγχος για το αντικείμενο που χρησιμοποιήθηκε, προκειμένου να διαγραφεί και να ελευθερώσει χώρο στο inventory του παίχτη.

4.11. Boss

Ο αρχηγός αποτελεί απαραίτητο στοιχείο για κάθε παιχνίδι όπου ο παίχτης μπορεί να μονομαχήσει εναντίον του ως το τελικό εμπόδιο μεταξύ αυτού και του τερματισμού του παιχνιδιού.

4.11.1. TriggerBoss.cs

Αυτό το script είναι υπεύθυνο για την καθιέρωση της αρχής της μάχης ενάντια του αρχηγού. Θα τοποθετηθεί σε ένα αντικείμενο τύπου collider το οποίο θα τεθεί ως trigger

προκειμένου να ελεγχθεί, αν ο χαρακτήρας του παίχτη έχει εισχωρήσει στην περιοχή που περιβάλλει για να ενεργοποιήσει την μάχη. Αποτελείται από μία συνάρτηση τύπου OnTriggerEnter2D η οποία κατά την επαφή με τον παίχτη λαμβάνει το script PlayerController και όσο βρίσκεται ο παίχτης εντός των ορίων του collider τότε καλείται το event [7],[8] OnPlayerEnterTrigger, το οποίο καλεί την συνάρτηση OnTriggerEnter2D. Με αποτέλεσμα να παρακολουθείται η κατάσταση του παίχτη όσο βρίσκεται μέσα στο collider.

4.11.2. EnemyBossManager.cs

Όταν ο παίχτης ενεργοποιήσει τον αρχηγό, το πρώτο πράγμα που θα προσέξει είναι η εμφάνιση μίας μπάρας στο πάνω μέρος της οθόνης του, με την υγεία που έχει ο αρχηγός και το όνομά του. Το συγκεκριμένο script τοποθετεί το απαραίτητο όνομα στην μπάρα του αρχηγού όταν ενεργοποιηθεί στην συνάρτηση Start.

4.11.3. ShieldField.cs

Κατά την διάρκεια της μάχης με τον παίχτη, ο αρχηγός θα χάνει υγεία ως αποτέλεσμα των επιθέσεων του. Για να μην είναι μονότονη η μονομαχία επιλέχτηκε, κάθε φορά που ο αρχηγός φτάνει σε κάποιο ποσοστό της υγείας του να ενεργοποιείται μία προστατευτική ασπίδα, η οποία μπλοκάρει τις επιθέσεις του παίχτη και διατηρείται από εχθρούς οι οποίοι επιτίθενται με επιθέσεις από απόσταση(ranged attacks).

Η συνάρτηση DestroyShield ελέγχει αν βρίσκονται εν υγεία οι εχθροί που διατηρούν την ασπίδα. Αν έχουν ηττηθεί όλοι τότε η ασπίδα διαλύεται μετά από 0.1 δευτερόλεπτα. Η συνάρτηση αυτή καλείται σε κάθε frame.

4.11.4. BossHealthbar.cs

Το συγκεκριμένο script χειρίζεται την λειτουργία της υγείας του αρχηγού με την χρήση events. Τα events χρησιμοποιούνται για να μειωθεί ο υπολογιστικός όγκος κατά την διάρκεια της εκτέλεσης του παιχνιδιού, εφόσον με την χρήση τους δεν θα χρησιμοποιηθεί η συνάρτηση Update η οποία καλείται σε κάθε frame.

Αρχίζοντας, στην συνάρτηση Awake ορίζονται οι απαραίτητες τιμές προκειμένου να τεθούν κατάλληλα 1.το όνομα του αρχηγού, 2.να ξεκινήσει η υγεία του στην μέγιστη δυνατή τιμή και 3.να γίνει αναφορά στο στοιχείο Animator που χρησιμοποιεί για τα animations του.

Στην συνέχεια ακολουθεί η Start συνάρτηση στην οποία κάθε φορά που αλλάζει η τιμή της υγείας του αρχηγού τότε καλείται η συνάρτηση UpdateBossUI μέσω της συνάρτησης UpdateBossUI_OnHealthChanged.

Στην Update συνάρτηση ορίζονται τα όρια που μπορεί να έχει η υγεία του αρχηγού, εξαιρώντας τις αρνητικές τιμές ή τιμές που ξεπερνούν το ανώτατο όριο της maxHealth. Σε περίπτωση που το αντικείμενο HP_UI δεν έχει καταστραφεί κατά την διάρκεια της εκτέλεσης

του παιχνιδιού, όταν χάσει πόντους υγείας για πρώτη φορά, τότε εμφανίζεται η μπάρα υγείας του. Στην συνέχεια αποθηκεύονται τα αντικείμενα τύπου GameObject για να χρησιμοποιηθούν παρακάτω.

Συνεχίζοντας, ακολουθούν οι συναρτήσεις ReturnHealth και HealthNormalized οι οποίες επιστρέφουν τιμές για την αξιοποίησή τους από άλλα scripts.

Ακολουθώντας, την λογική της κατασκευής της μπάρας υγείας του χαρακτήρα του παίχτη, γίνεται έλεγχος για την τιμή που έχει. Σε περίπτωση που είναι 0 τότε η τιμή IsAlive αλλάζει σε ψευδής με αποτέλεσμα να παίζεται το animation του θανάτου του αρχηγού και να καταστρέφονται τα στοιχεία που σχετίζονται με αυτόν όπως είναι το HP_UI, το BossBattleManager και το BossTrigger εφόσον η μάχη μεταξύ παίχτη και αρχηγού έχει τελειώσει. Για να μην είναι απαραίτητο να γίνεται συνεχώς έλεγχος για την κατάσταση της υγείας του αρχηγού επιλέχτηκε να γίνει και σε αυτή την περίπτωση χρήση event το οποίο θα καλέσει την συνάρτηση UpdateBossUI μόνο μία φορά όταν η κατάσταση την οποία χειρίζεται το event επιτευχθεί.

Στην συνέχεια του script ακολουθεί η συνάρτηση SetBossName η οποία αλλάζει το αντικείμενο τύπου text πάνω από την μπάρα υγείας του αρχηγού στο όνομά του.

Ακολουθεί στην συνέχεια η συνάρτηση TakeDamage η οποία δέχεται ως όρισμα μία τιμή για να καθοριστεί, πόσο θα αφαιρείται από την υγεία του κάθε φορά που δέχεται επίθεση από τον παίχτη. Για να υπάρχει μια ένδειξη για τον χρήστη, εμφανίζεται ένα στοιχείο(Pop up) με τον αριθμό της επίθεσης που προκάλεσε στον αρχηγό. Ακολουθεί η χρήση των event όπου εκτελούν την συνάρτηση TakeDamage κάθε φορά που δέχεται επιθέσεις ο εχθρός χωρίς να χρειάζεται να γίνεται συνεχής έλεγχος μέσω της συνάρτησης Update.

Ακολουθούν οι συναρτήσεις SetUIHealthBarToActive και SetHealthBarToInactive προκειμένου να ελέγχεται η κατάσταση της μπάρας του αρχηγού όταν αυτό είναι απαραίτητο, για κάποια μετάβαση της κατάστασης της μάχης. Τέλος, η συνάρτηση UpdateBossUI_OnHealthChanged καλεί την UpdateBossUI όταν αυτό κρίνεται απαραίτητο από την εκκίνηση του event.

4.11.5. BossBattle.cs

Αρχικά, δημιουργείται ένα αντικείμενο Stage τύπου enumerator το οποίο θα περιέχει τα διαφορετικά στάδια στα οποία θα εξελίσσεται η μάχη μεταξύ παίχτη και αρχηγού.

Στην συνάρτηση Awake αρχικά λαμβάνεται το script για την λειτουργία του BossHeakthbar αντικειμένου της μπάρας υγείας και ορίζονται δύο λίστες οι οποίες θα χρησιμοποιηθούν για να κρατήσουν τις θέσεις(συντεταγμένες) στις οποίες θα εμφανιστούν οι εχθροί καθώς και την διατήρησή τους σε μία ομάδα για την ευκολότερη διαχείριση κατά την διάρκεια της μάχης με τον αρχηγό αλλά και για την ολοκλήρωσή της. Τέλος, ως αρχικό στάδιο ορίζεται το WaitingToStart για να μην ενεργοποιηθεί η κατάσταση μάχης με τον αρχηγό και να μην εμφανιστούν οι εχθροί κατά την εκκίνηση του παιχνιδιού αλλά μόνο όταν ο παίχτης φτάσει στο κατάλληλο σημείο του επιπέδου.

Στην συνέχεια ακολουθεί η συνάρτηση Start στην οποία αρχικά κάθε φορά που ο αρχηγός χάνει πόντους υγείας καλείται η συνάρτηση BossBattle_OnDamaged (μέσω του event OnDamaged) η οποία ελέγχει σε πιο στάδιο της μάχης βρίσκεται ο παίχτης με βάση το πόσο επί τοις εκατό υγεία βρίσκεται ο αρχηγός. Με την αξιοποίηση μίας δομής switch ελέγχεται αν ο αρχηγός βρίσκεται κάτω από το 50% της συνολικής του υγείας για να γίνει

αλλαγή κατάστασης καλώντας την συνάρτηση StartBextStage. Η συνάρτηση StartNextStage αξιοποιεί επίσης μία δομή switch για να εκτελέσει την κατάλληλη λειτουργία για κάθε διαφορετικό στάδιο, όπου αν ο αρχηγός βρεθεί στο 50 % της υγείας του τότε ξεκινάει η δεύτερη φάση όπου ενεργοποιείται η πρώτη ασπίδα(shield_1) σε περίπτωση που δεν έχει ήδη καταστραφεί. Αντίστοιχα, όταν ο αρχηγός φτάσει στο 25 % της υγείας του τότε ενεργοποιείται η δεύτερη ασπίδα(shield_2) σηματοδοτώντας την έναρξη του τρίτου σταδίου. Προχωρώντας, ακολουθεί το event OnDead το οποίο καλεί την συνάρτηση BossBattle_OnDead όταν ο αρχηγός χάσει όλους τους πόντους υγείας του. Αυτή με την σειρά της καλεί την συνάρτηση DestroySpawnedEnemies η οποία ελέγχει για τους εχθρούς, οι οποίοι είναι μέρος της μάχης με τον αρχηγό, αν δεν έχουν ηττηθεί από τον παίχτη και τους καταστρέφει εφόσον ο αρχηγός έχει ηττηθεί. Τελικά, ακολουθεί το event OnPlayerEnterTrigger το οποίο καλεί την συνάρτηση TriggerBoss_OnPlayerEnterTrigger (η οποία καλείται όταν ο χαρακτήρας του παίχτη βρίσκεται μέσα στην εμβέλεια ενεργοποίησης της μάχης), η οποία καλεί την συνάρτηση StartBattle, η οποία αλλάζει το στάδιο της μάχης από αναμονή στο στάδιο 1 και τελικά καλείται η συνάρτηση SpawnEnemy για να δημιουργήσει τρεις εχθρούς.

Η συνάρτηση αυτή δημιουργεί τρεις εχθρούς σε τυχαίες θέσεις ανάμεσα στα πιθανά σημεία που έχουν οριστεί, προσθέτοντας δυναμικότητα στην μάχη. Λόγω του τρόπου κατασκευής των εχθρών, πρέπει να αλλάξουν κατεύθυνση κατά 180° προκειμένου να βλέπουν προς την κατεύθυνση του παίχτη. Τελικά, τοποθετούνται στην λίστα για να μπορέσουν να καταστραφούν όταν τελειώσει η μάχη.

4.12. Saving System

Ένα από τα πιο σημαντικά στοιχεία για ένα παιχνίδι είναι η δημιουργία ενός συστήματος, το οποίο παρέχει στον χρήστη την ικανότητα να αποθηκεύει την κατάσταση του παιχνιδιού για την συνέχισή του κάποια άλλη στιγμή. Το ακόλουθο σύστημα που επιλέχτηκε να χρησιμοποιηθεί, αποτελεί εύχρηστο εργαλείο με το οποίο αποθηκεύονται ξεχωριστά κομμάτια από αντικείμενα έως τιμές από scripts αξιοποιώντας διαφορετικούς κωδικούς αναγνώρισης μοναδικούς για κάθε στοιχείο σε μορφή binary.

4.12.1. SaveableEntity.cs

Σε αυτό το script θα γίνεται ο χειρισμός της ταυτοποίησης των αντικειμένων που παίρνουν μέρος στην διαδικασία της αποθήκευσης. Αρχικά, ορίζεται μια ιδιωτική μεταβλητή id και μία δημόσια μεταβλητή Id. Η δημόσια μεταβλητή θα παρέχει πρόσβαση στην ιδιωτική. Για την διευκόλυνση των τεστ δημιουργήθηκαν κατάλληλα promts με την χρήση του ContextMenu. Στην συνέχεια στην συνάρτηση GenerateId με την χρήση της κλάσης Guid δημιουργείται ένας τυχαίος αλφαριθμητικός αριθμός ο οποίος μετατρέπεται σε string και ορίζεται ως το νέο id. Ο αριθμός αυτός δεν είναι μοναδικός, αλλά η πιθανότητα επανεμφάνισής του είναι τόσο μικρή που θεωρείται ως μοναδικός.

Συνεχίζοντας, ακολουθεί η συνάρτηση SaveState στην οποία θα αποθηκεύονται οι τιμές των επιθυμητών δεδομένων και το είδος τους. Για να επιτευχθεί αυτό θα δημιουργηθεί ένα Dictionary το οποίο θα έχει ως ορίσματα, το όνομα του αντικειμένου(string) και το SaveData αντικείμενο(object) το οποίο περιέχει τα δεδομένα που πρέπει να σωθούν. Έτσι, εφόσον υπάρχουν πολλά διαφορετικά στοιχεία σε κάθε σκηνή, θα χρησιμοποιηθεί μία δομή επανάληψης για κάθε στοιχείο όπου θα δημιουργείται νέα είσοδος στο Dictionary με την κατάσταση των αντικειμένων και των τιμών τους και στην συνέχεια θα επιστρέφει την κατάσταση του Dictionary.

Προχωρώντας, στην συνάρτηση LoadState η οποία θα χρησιμοποιήσει το αντικείμενο Dictionary που δημιουργήθηκε προηγουμένως και για κάθε στοιχείο του, θα φορτωθούν οι αποθηκευμένες τιμές αντί για τις αρχικές.

4.12.2. ISaveable.cs

Το script ISaveable ορίζει μία κλάση τύπου interface η οποία αναγκάζει τα script που την χρησιμοποιούν(μέσω κληρονομικότητας) να αποθηκεύσουν τα στοιχεία τους και να υλοποιήσουν τις συναρτήσεις SaveState και LoadState στο εσωτερικό τους.

4.12.3. SaveLoadSystem.cs

Ξεκινώντας, στην συνάρτηση SaveState, κάθε script που χρησιμοποιεί στοιχεία από το SaveableEntity πρέπει:

- να επιλέξει το κατάλληλο Id και
- να αλλάξει τα δεδομένα του σε αυτά που είχαν αποθηκευτεί.

Με την ίδια λογική στην συνάρτηση LoadState ελέγχεται αν μπορούν να χρησιμοποιηθούν οι αποθηκευμένες τιμές και αλλάζει τις τιμές του παιχνιδιού στις κατάλληλες ανακτημένες τιμές.

Στην συνέχεια ακολουθεί η συνάρτηση SaveFile, η οποία είναι υπεύθυνη για την δημιουργία ενός αρχείου που περιέχει τις αποθηκευμένες τιμές. Συνεχίζοντας, για να μπορέσουν να ανακτηθούν οι πληροφορίες από το αρχείο στην κατάλληλη μορφή για επεξεργασία, θα οριστεί η συνάρτηση LoadFile ως τύπου Dictionary εφόσον αυτός ο τύπος δεδομένων χρησιμοποιήθηκε και παραπάνω. Στην LoadFile γίνεται έλεγχος αν υπάρχει το αρχείο στο μονοπάτι που έχει οριστεί και αν δεν υπάρχει, τότε δημιουργεί και επιστρέφει νέο Directory. Σε περίπτωση που το αρχείο βρεθεί τότε το ανοίγει, θα αποκωδικοποιήσει τις τιμές του αρχείου και θα τις μετατρέψει σε μορφή που μπορούν να χρησιμοποιηθούν από το σύστημα.

Ακολουθεί η συνάρτηση Save, η οποία διαβάζει το αρχείο και καλώντας την SaveState αναγκάζει τα στοιχεία που πρέπει να αποθηκευτούν να εκτελέσουν την λειτουργία τους και αποθηκεύει τις νέες τιμές στο αρχείο. Αντίστοιχα η συνάρτηση Load ελέγχει για την ύπαρξη του αρχείου και καλεί την συνάρτηση LoadState για να αντικατασταθούν οι τιμές του παιχνιδιού με τις ανακτημένες που βρέθηκαν στο αρχείο.

Μέσω του ContextMenu με τα κατάλληλα ονόματα γίνεται ο έλεγχος της λειτουργίας των συναρτήσεων.

4.12.4. StateManager.cs

Σε αυτό το script θα γίνει χρήση του συστήματος αποθήκευσης για να αποθηκευτεί η κατάσταση ορισμένων αντικειμένων, τα οποία έχουν ιδιαίτερη σημασία κατά την διαδικασία της ανάκτησης της αποθηκευμένης κατάστασης του παιχνιδιού. Τέτοια αντικείμενα είναι τα παζλ που έχει να αντιμετωπίσει ο παίχτης και πόρτες τις οποίες έχει ανοίξει, που πρέπει να διατηρούν την κατάστασή τους και να μην επανεμφανιστούν όταν ο παίχτης ανακτά την αποθηκευμένη κατάσταση, εφόσον μπορούν να αποτελέσουν εμπόδια για τον παίχτη.

Ειδικότερα, στην συνάρτηση Awake ορίζεται το μέγεθος του πίνακα ObjState το οποίο θα περιέχει την κατάσταση του κάθε αντικειμένου που βρίσκεται στον πίνακα Obj. Ακολουθεί η συνάρτηση Update στην οποία γίνεται έλεγχος για κάθε στοιχείο του πίνακα Obj και σε περίπτωση που το αντικείμενο υπάρχει, τότε με βάση την κατάστασή του τοποθετείται ανάλογα αληθής ή ψευδής τιμή στον πίνακα των καταστάσεων. Σε περίπτωση που δεν υπάρχει το αντικείμενο επειδή έχει καταστραφεί κατά την διάρκεια της εξέλιξης του παιχνιδιού, τοποθετείται η τιμή ψευδής στον πίνακα καταστάσεων.

Στην συνέχεια υλοποιούνται οι απαραίτητες συναρτήσεις που επιβάλει το σύστημα προκειμένου να σωθεί η κατάσταση των αντικειμένων του πίνακα.

4.13. Specs

Με βάση την ιστοσελίδα [19] που προσφέρει το Unity (**Εικόνα 25**), το κάθε παιχνίδι σχεδιασμένο σε αυτή την πλατφόρμα μπορεί να λειτουργήσει στα περισσότερα διαφορετικά μοντέλα υπολογιστών. Δοκιμασίες για το τρέξιμο της εφαρμογής έγιναν σε υπολογιστή με Windows 7 64-bit, Gpu: (intel)i7 3770 καθώς και σε υπολογιστή με Windows 11 Pro 64-bit,

Gpu: AMD Ryzen 5 5600G with Radeon Graphics. Και στις δύο περιπτώσεις το παιχνίδι λειτουργούσε χωρίς προβλήματα, συνεπώς τα παρακάτω στοιχεία καλύπτουν τις απαραίτητες προδιαγραφές για το συγκεκριμένο παιχνίδι.

System	Minimum requirements
Desktop	
Operating system	Windows 7 SP1+ macOS 10.12+ Ubuntu 16.04+
CPU	SSE2 instruction set support.
GPU	Graphics card with DX10 (shader model 4.0) capabilities.
iOS	iOS 9.0 or higher.
Android	OS 4.1 or later ARMv7 CPU with NEON support or Atom CPU OpenGL ES 2.0 or later.
WebGL	Any recent desktop version of Firefox, Chrome, Edge or Safari.
Universal Windows Platform	Windows 10 and a graphics card with DX10 (shader model 4.0) capabil

Image 25

Εικόνα 25: Οι ελάχιστες απαιτήσεις για να λειτουργήσει ένα παιχνίδι κατασκευασμένο

σε Unity.

5. Συμπεράσματα & Μελλοντικοί Στόχοι

5.1. Συμπεράσματα

Προκειμένου να γίνει ένα παιχνίδι πιο ευχάριστο για τον παίχτη, πρέπει να έχει δημιουργηθεί με σκοπό την διασκέδασή του. Αυτό ισχύει και στα παιχνίδια που έχουν εκπαιδευτικό χαρακτήρα. Για να θεωρηθούν ως επιτυχή, πέρα από το να έχουν επαρκή συστήματα και ισχυρή βάση για το θεματικό περιεχόμενο που πρέπει να μεταδώσουν, είναι απαραίτητο να ελκύουν τον παίχτη με τον τρόπο λειτουργίας τους. Εν κατακλείδι, ένα παιχνίδι μάθησης δεν αρκεί ούτε να είναι απλό σε φύση αλλά να παρέχει αυτούσια την πληροφορία που πρέπει να μαθευτεί, ούτε να έχει πολύπλοκα συστήματα που να λειτουργούν απαγορευτικά στην διασκέδαση και κατ' επέκταση στην μάθηση, αλλά να προσπαθεί να ισορροπεί ανάμεσα στην μάθηση και την διασκέδαση προκειμένου να επιτευχθούν τα μέγιστα δυνατά αποτελέσματα εκμάθησης.

5.2. Μελλοντικοί στόχοι

Θα ήταν επιθυμητό να προστεθούν cutscenes τα οποία θα μπορούσαν να εμπλουτίζουν την εμπειρία των χρηστών είτε προϊδεάζοντας τους για επικείμενους κινδύνους ή δείχνοντάς τους τον δρόμο σε περίπτωση που έχουν δυσκολία καθοδήγησης του παιχνιδιού. Θα έπρεπε να δημιουργηθούν περισσότεροι τύποι εχθρών, νέων επιπέδων και δημιουργία συστήματος δυσκολίας, η οποία θα αυξάνει την πολυπλοκότητα των εχθρών και θα ανεβάζει το επίπεδο των ερωτήσεων από (A1) μέχρι και (Lower). Επίσης, θα μπορούσε να υποστηρίζει συστήματα που διευκολύνουν άτομα με ειδικές ανάγκες(όπως είναι η αχρωματοψία).

Ευχαριστίες

Χαράλαμπος Ραφαήλ Δημητρίου (Ε.Κ.Π.Α Πληροφορική και Τηλεπικοινωνίες)

Γιώργος Σαφαλίδης(Σχολή Τεχνολογικών Εφαρμογών - Τμήμα Μηχανικών Πληροφορικής ΤΕΙ Αθήνας)

Πηγές

[1] “Top Free Game Assets.” Itch.io, itch.io/game-assets/free.

[2] “OpenGameArt.org.” OpenGameArt.org, 2000, opengameart.org/.

[3] ---. “Unity - Manual: Order of Execution for Event Functions.” Docs.unity3d.com, docs.unity3d.com/Manual/ExecutionOrder.html.

[4] “Use of English.” Test-English, test-english.com/use-of-english/.

[5] Carrillo, Uriel. “Update vs. FixedUpdate vs. LateUpdate in Unity.” LogRocket Blog, 12 Oct. 2022, blog.logrocket.com/update-vs-fixedupdate-vs-lateupdate-in-unity/. Accessed 31 Jan. 2023.

[6] Unity Technologies. “Unity - Scripting API: MonoBehaviour.FixedUpdate().” Docs.unity3d.com, docs.unity3d.com/ScriptReference/MonoBehaviour.FixedUpdate.html.

[7] Unity Technologies. “Unity - Scripting API: UnityEvent.” Unity3d.com, 2019, docs.unity3d.com/ScriptReference/Events.UnityEvent.html. Accessed 24 Aug. 2019.

[8] Unity Technologies. “Unity - Manual: UnityEvents.” Docs.unity3d.com, docs.unity3d.com/Manual/UnityEvents.html.

[9] Unity Technologies. “How Shader Graph Can Be Applied to Your 2D or 3D Game | Unity.” Unity.com, unity.com/features/shader-graph.

[10] Unity Playmaker: Learn the Basics + Playmaker Tutorials. 27 Aug. 2020, www.gamedesigning.org/learn/unity-playmaker/#:~:text=Frequently%20Asked%20Questions-. Accessed 31 Jan. 2023.

[11] “Hutong Games.” Hutong Games, hutonggames.com/.

[12] “Playmaker Manual.” Hutonggames.fogbugz.com, hutonggames.fogbugz.com/. Accessed 31 Jan. 2023.

[13] Action Verbs Interactive Crossword Ending with -Ing. www.kidslearningville.com/action-verbs-interactive-crossword-ending-with-ing/. Accessed 16 Feb. 2023.

[14] “Present Simple.” Www.gamestolearnenglish.com, www.gamestolearnenglish.com/present-simple/.

[15] “I Spy on PrimaryGames.com.” PrimaryGames, www.primarygames.com/langarts/ispy/. Accessed 16 Feb. 2023.

[16] “English for Everybody - English Online Games - Add Two Letters.” Www.english-Online.org.uk, www.english-online.org.uk/games/twolettersframe.htm. Accessed 16 Feb. 2023.

[17] PLUS, ESL Games. Action Verbs Nouns Collocation ESL Vocabulary Pirate Board Game. 18 Aug. 2012, www.eslgamesplus.com/action-verbs-nouns-collocation-esl-vocabulary-pirate-board-game/. Accessed 16 Feb. 2023.

[18] Vincent, Daniel. “10 Great Sites with Free Games for Practising English.” ELT Learning Journeys, 9 June 2015, eltlearningjourneys.com/2015/06/09/10-great-sites-with-free-games-for-practising-english/.

[19] ---. “Unity - Manual: System Requirements for Unity 2019.1.” Docs.unity3d.com, docs.unity3d.com/2019.1/Documentation/Manual/system-requirements.html.

[20] “How to Make a Better Health Bar in Unity : Chip Away Tutorial.” Www.youtube.com, www.youtube.com/watch?v=CFASjEuhyf4&list=PLGUw8UNswJEPL5NuTGjh-K__V75CkoHAK. Accessed 20 Feb. 2023.

[21] “Creating a Mute Button in Unity.” Www.youtube.com, www.youtube.com/watch?v=ZmQAHhZ7784&loop=0. Accessed 20 Feb. 2023.

[22] “POLISH Your Game with Damage Popups! (Unity Tutorial).” Www.youtube.com, www.youtube.com/watch?v=iD1_JczQcFY&t=216s. Accessed 20 Feb. 2023.

[23] “Health Pickups in Unity.” Www.youtube.com, www.youtube.com/watch?v=TQhzBAKaOTE&loop=0. Accessed 20 Feb. 2023.

[24] “How to Make Quiz Game with Multiple Choices in Unity | Part 2.” Www.youtube.com, www.youtube.com/watch?v=POUemIGCyr0. Accessed 20 Feb. 2023.

[25] “Unity 2D Game Basics - Using an Item from Inventory.” Www.youtube.com, www.youtube.com/watch?v=ZIR8Kk5F13Y&loop=0. Accessed 20 Feb. 2023.

[26] “Unity 2D Game Basics - Unlocking and Opening Doors.” Www.youtube.com, www.youtube.com/watch?v=xFSAvnlmreE. Accessed 20 Feb. 2023.

[27] “Creating Game Mechanics - Enemy Loot Drop and Coin Pickup - 2D Roguelike.” Www.youtube.com, www.youtube.com/watch?v=nSXkiRdwiGQ&loop=0. Accessed 20 Feb. 2023.

[28] “Unity 2D Aim and Shoot at Mouse Position Tutorial.” Www.youtube.com, www.youtube.com/watch?v=-bkmPm_Besk&loop=0. Accessed 20 Feb. 2023.

[29] “2D ENEMY Melee Combat in UNITY - Setting up Character - Part 1.” Www.youtube.com, www.youtube.com/watch?v=qgX941I-YqE. Accessed 20 Feb. 2023.

[30] “2D ENEMY Melee Combat in UNITY - Coding Attack Pattern - Part 2.” Www.youtube.com, www.youtube.com/watch?v=waj6i9cQ6rM&t=449s. Accessed 20 Feb. 2023.

[31] “Unity 2D Enemy Melee Combat - Improving AI - Part 4.” Www.youtube.com, www.youtube.com/watch?v=j4036aBB4Mo. Accessed 20 Feb. 2023.

[32] “SHOOTING/FOLLOW/RETREAT ENEMY AI with UNITY and C# - EASY TUTORIAL.” Www.youtube.com, www.youtube.com/watch?v=_Z1t7MNk0c4&loop=0. Accessed 20 Feb. 2023.

[33] “How to Save Objects or Characters between Scenes in Unity - Easy Tutorial Beginner.” Www.youtube.com, www.youtube.com/watch?v=EnMUpV9Ueg4&loop=0. Accessed 20 Feb. 2023.

[34] “Word Scramble Game with Playmaker - Unity.” Www.youtube.com, www.youtube.com/watch?v=C_Ix8tpwbnk&t=56s. Accessed 20 Feb. 2023.

[35] “Accessing a PlayMakerFSM in Scripts.” Hutonggames.fogbugz.com, hutonggames.fogbugz.com/default.asp?W329. Accessed 20 Feb. 2023.

[36] “How Do I Get the Current FSM and Its State in Script? [SOLVED].” Hutonggames.com, hutonggames.com/playmakerforum/index.php?topic=8567.0. Accessed 20 Feb. 2023.

[37] “Shield Force Field - Shader Graph Tutorial.” Www.youtube.com, www.youtube.com/watch?v=6AKR4b4C5jo&t=111s. Accessed 20 Feb. 2023.

[38] “Awesome Boss Fight in Unity with Stages and Increasing Difficulty!” Www.youtube.com, www.youtube.com/watch?v=qZC1VYWnHZ8. Accessed 20 Feb. 2023.

[39] “Making Scene Transitions in Unity.” Www.youtube.com, www.youtube.com/watch?v=ACkVh3quMzU. Accessed 20 Feb. 2023.

[40] “FADE in and FADE OUT.” Www.youtube.com, www.youtube.com/watch?v=BM2Hav6iPQ0. Accessed 20 Feb. 2023.

[41] “How to Make a Save & Load System in Unity | 2022.” Www.youtube.com, www.youtube.com/watch?v=aUi9aijvpgs&t=405s. Accessed 20 Feb. 2023.

[42] “Making One DUNGEON Level Map by Using TILEMAP2D in Unity.” Www.youtube.com, www.youtube.com/watch?v=HcFlVYKiWlc&t=874s. Accessed 20 Feb. 2023.

[43] “How to Make Pause/Resume Menu in Unity.” Www.youtube.com, www.youtube.com/watch?v=J1x6cSTGQO8&loop=0. Accessed 20 Feb. 2023.

[44] “How to Destroy a GameObject after the Animation.” Www.youtube.com, www.youtube.com/watch?v=ygIC7DcXRCk. Accessed 20 Feb. 2023.

[45] Hagedoorn, Hilbert. “PC Was the Most Popular Platform among Gamers in 2022.” Guru3D.com, www.guru3d.com/news-story/pc-was-the-most-popular-platform-among-gamers-in-2022.html#:~:text=According%20to%20the%20data%20provided. Accessed 21 Feb. 2023.

[46] “How to Make a Trap in Unity.” Www.youtube.com, www.youtube.com/watch?v=diHNwcoYUO0&loop=0. Accessed 21 Feb. 2023.

[47] “What Is FOMO (Fear of Missing Out)? Definition from WhatIs.com.” WhatIs.com, www.techtarget.com/whatis/definition/FOMO-fear-of-missing-out#:~:text=The%20fear%20of%20missing%20out.

[48] “What Platforms Are Supported by Unity?” Unity, support.unity.com/hc/en-us/articles/206336795-What-platforms-are-supported-by-Unity-.