ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΥΠΟΛΟΓΙΣΤΩΝ
ΡΟΗ ΛΟΓΙΣΜΙΚΟΥ ΚΑΙ ΠΛΗΡΟΦΟΡΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΣΧΕΔΙΑΣΜΟΣ ΚΑΙ ΑΝΑΠΤΥΞΗ ΕΚΠΑΙΔΕΥΤΙΚΟΥ ΠΑΙΧΝΙΔΙΟΥ ΑΝΑΚΥΚΛΩΣΗΣ

“The Recycle Game”

του φοιτητή

Κυριακόπουλου Άγγελου (71347242)

Επιβλέπων Καθηγητής
Γεώργιος Μπαρδής, Επίκουρος Καθηγητής

Αιγάλεω, Μάρτιος 2024

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΥΠΟΛΟΓΙΣΤΩΝ
ΡΟΗ ΛΟΓΙΣΜΙΚΟΥ ΚΑΙ ΠΛΗΡΟΦΟΡΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΣΧΕΔΙΑΣΜΟΣ ΚΑΙ ΑΝΑΠΤΥΞΗ ΕΚΠΑΙΔΕΥΤΙΚΟΥ ΠΑΙΧΝΙΔΙΟΥ ΑΝΑΚΥΚΛΩΣΗΣ

“The Recycle Game”

Ον/μο: **Κυριακόπουλος Άγγελος**

Αριθμός Μητρώου: **71347242**

Επιβλέπων Καθηγητής: **Γεώργιος Μπαρδής, Επίκουρος Καθηγητής**

Εγκρίθηκε από την 3μελή εξεταστική επιτροπή των:

Georgios Bardis Digitally signed by Georgios Bardis Date: 2024.03.26 15:42:14 +02'00'	Christos Troussas Digitally signed by Christos Troussas Date: 2024.03.27 08:08:08 +02'00'	Panagiota Tselenti Digitally signed by Panagiota Tselenti Date: 2024.03.26 16:06:21 +02'00'
Γεώργιος Μπαρδής Επίκουρος Καθηγητής	Χρήστος Τρούσσας Επίκουρος Καθηγητής	Παναγιώτα Τσελέντη Ε.ΔΙ.Π.

Αιγάλεω, Μάρτιος 2024

UNIVERSITY OF WEST ATTICA

SCHOOL OF ENGINEERING

DEPARTMENT OF INFORMATICS AND COMPUTER ENGINEERING

SOFTWARE AND INFORMATION SYSTEMS FLOW

DESIGN AND DEVELOPMENT OF A RECYCLING EDUCATION COMPUTER GAME

“The Recycle Game”

Full Name: **Kyriakopoulos Angelos**

ID Number: **71347242**

Supervisor: **Georgios Bardis, Assistant Professor**

Egaleo, March 2024

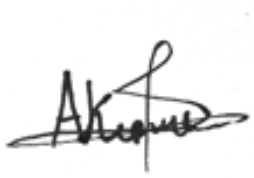
ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος **Κυριακόπουλος Άγγελος**, με αριθμό μητρώου **71347242**, 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,

Κυριακόπουλος Άγγελος



(υπογραφή)

Περίληψη

Οι καθημερινές ανάγκες των ανθρώπων στις μέρες μας όλο και αυξάνονται αφού καθημερινά εφευρίσκονται νέοι τρόποι αυτοματοποίησης και διευκόλυνσης της ζωής μας, ωστόσο έχουν δημιουργήσει ανοδική τάση στην παραγωγή προϊόντων με αποτέλεσμα την επιβάρυνση του περιβάλλοντος στο οποίο ζούμε με τα κατάλοιπα που αφήνουν. Γι' αυτό τον λόγο έχει γίνει αναγκαία η διαδικασία ανακύκλωσης αντικειμένων στην καθημερινότητα μας διότι υπάρχει μεγάλο πρόβλημα στις χωματερές όλου του πλανήτη αφού ο πληθυσμός της Γης αυξάνεται με ραγδαίους ρυθμούς με αποτέλεσμα οι ανάγκες για παραγωγή προϊόντων να αυξάνονται μαζί του. Τα παιδιά μαθαίνουν από μικρή ηλικία την ανάγκη της ανακύκλωσης στη ζωή μας, όμως αυτό συνήθως γίνεται μέσω προφορικών ενημερώσεων και ασκήσεων που δεν παράγουν τα καλύτερα αποτελέσματα συγκράτησης αυτής της γνώσης.

Με την τεχνολογία να έχει εδραιωθεί σαν βασικό και αναγκαίο αντικείμενο στην καθημερινότητα μας άτομα όλο και μικρότερης ηλικίας έχουν την δυνατότητα χρήσης κάποιου ψηφιακού μέσου κατά την ανάπτυξη τους. Έτσι έχει αρχίσει να προσαρμόζεται και η εκπαίδευση σε αυτή την τάση της κοινωνίας, ωστόσο ακόμα βασιζόμαστε αρκετά στην προφορική εκπαίδευση. Γνωρίζοντας τα παραπάνω θα εστιάσουμε στην χρήση ενός ψηφιακού και διαδραστικού μέσου, τα βιντεοπαιχνίδια, για την ανάπτυξη μιας σωστής νοοτροπίας ως προς την βιωσιμότητα και την ανακύκλωση. Παρόλο που τα βιντεοπαιχνίδια δεν είναι κάτι νέο στην κοινωνία μας, έχουν αποδειχθεί πολύ καλά και διαδραστικά μέσα εκπαίδευσης τα τελευταία χρονιά καθώς κεντρίζει το ενδιαφέρον του παίχτη μέσω ψυχαγωγίας αυτού, κάτι που βοηθά στη συγκράτηση της γνώσης που προβάλλεται σε αυτό.

Επομένως, με την παρούσα εργασία θα ερευνήσουμε το περιεχόμενο που οφείλει να έχει ένα εκπαιδευτικό βιντεοπαιχνίδι ανακύκλωσης με σκοπό να κεντρίσει το ενδιαφέρον του παίχτη, ώστε να έχει ως αποτέλεσμα την εκπαίδευση και κατανόηση της ανάγκης για ανακύκλωση στη ζωή μας. Ταυτόχρονα θα εμβαθύνουμε στις τεχνολογίες που κάναμε χρήση για την επίτευξη του στόχου αυτού αλλά και πως αυτές μπορούν να εξελιχθούν μακροπρόθεσμα.

Λέξεις-Κλειδιά

Ανακύκλωση, Περιβάλλον, Εκπαίδευση, Τεχνολογία, Βιντεοπαιχνίδια, Βιωσιμότητα, Ψυχαγωγία.

Abstract

The daily demands of individuals in our era are on the rise due to the continuous development of new ways to automate and streamline our lives. However, this progress has led to an increasing trend in the production of new products, contributing to the environmental burden through the waste they generate. Consequently, recycling has become an imperative aspect of our daily lives. The global issue of overflowing landfills is exacerbated by the Earth's rapidly growing population, intensifying the demand for product creation. From an early age, children are taught about the importance of recycling, often through verbal instructions and exercises that may not yield optimal results in knowledge retention.

In a world where technology has firmly established itself as an indispensable tool, individuals of increasingly younger ages have access to digital media during their developmental years. Consequently, education has adapted to this societal shift, although oral instruction still plays a significant role. Recognizing this, we aim to explore the use of a digital and interactive medium—video games—to cultivate a mindset that emphasizes sustainability and recycling. While video games are not a recent addition to our society, they have proven to be effective and interactive educational tools in recent years. By engaging players through entertainment, video games facilitate the retention of knowledge presented within them.

Thus, this work seeks to investigate the essential content that an educational recycling video game should encompass to capture the player's interest, fostering an understanding of the importance of recycling in our lives. Simultaneously, we will delve into the technologies employed to achieve this goal and explore their potential long-term evolution.

Keywords

Recycling, Environment, Education, Technology, Video Games, Sustainability, Entertainment.

Ευχαριστίες

Η παρούσα διπλωματική εργασία ολοκληρώθηκε μετά από επίμονες προσπάθειες, σε ένα ενδιαφέρον γνωστικό αντικείμενο. Την προσπάθειά μου αυτή υποστήριξε ο επιβλέπων καθηγητής μου, τον οποίο θα ήθελα να ευχαριστήσω.

Ακόμα θα ήθελα να ευχαριστήσω την οικογένειά μου για τη συμπαράσταση κατά τη διάρκεια των σπουδών μου.

Πίνακας Περιεχομένων

Κατάλογος Εικόνων

1. Εισαγωγή

Τα βιντεοπαιχνίδια έχουν γίνει ολοένα πιο δημοφιλή και εξέχον μέρος της ζωής πολλών ανθρώπων τα τελευταία χρονιά. Προσφέρουν έναν διασκεδαστικό αλλά και συναρπαστικό τρόπο χαλάρωσης, μπορούν επίσης να προσφέρουν ένα συναίσθημα ολοκλήρωσης και επίτευξης στόχων. Εκτός από την παροχή ψυχαγωγίας, τα βιντεοπαιχνίδια μπορούν επίσης να βοηθήσουν στη βελτίωση του συντονισμού χεριού-ματιού και των δεξιοτήτων επίλυσης προβλημάτων. Με τα βιντεοπαιχνίδια να πληθαίνουν πολλοί επαγγελματικοί και εκπαιδευτικοί φορείς αποφάσισαν να εκμεταλλευτούν αυτή την άνοδο δημοτικότητας και να δημιουργήσουν παιχνίδια που έχουν σκοπό να εκπαιδεύουν προσωπικό με έναν πιο διαδραστικό τρόπο με την χρήση των βιντεοπαιχνιδιών. Έτσι δημιουργήθηκε μια νέα κατηγορία βιντεοπαιχνιδιών, αυτή που γνωρίζουμε μέχρι και σήμερα ως «σοβαρά παιχνίδια» (“serious games”).

Ειδικότερα στην παιδεία, οπού παιδιά όλο και μικρότερων ηλικιών αποκτούν ευκολότερη πρόσβαση στα ηλεκτρονικά μέσα ψυχαγωγίας και ενημέρωσης, έχουν γίνει ραγδαίες εξελίξεις ως προς την απομάκρυνση του ρολού του μαθητή ως παθητικός δέκτης και στην όδευση της διαδραστικότητας μέσω παιχνιδιών, με σκοπό την ευκολότερη κατανόηση και αποστήθιση της γνώσης από τους μαθητές. Ένα παράδειγμα θα ήταν η ενημέρωση των μαθητών για την κατάσταση που βρίσκεται αυτή την στιγμή το περιβάλλον του πλανήτη μας και τι μπορεί να κάνει ο κάθε μαθητής για να μειωθεί η περιβαλλοντική μόλυνση. Κάτι τέτοιο γινόταν μέχρι στιγμής μέσω παρουσιάσεων καθηγητών και βιβλίων τα οποία καθιστούν τον μαθητή παθητικό δέκτη χωρίς να κινούν την προσοχή του, ωστόσο, τελευταία, έχουν αναπτυχθεί διάφορες εκπαιδευτικές πλατφόρμες στον διαδίκτυο που προσπαθούν να πετύχουν παρόμοια ενημέρωση μέσω απλών διαδραστικών βιντεοπαιχνιδιών.

1.1 Στόχος Εργασίας

Όπως αναφέρθηκε παραπάνω η ανάγκη διαφοροποίησης των επιμορφωτικών παιχνιδιών σε σχέση με τα παιχνίδια διασκέδασης έδωσαν ζωή σε ένα νέο ορό παιχνιδιών που αφορούσαν την επιμόρφωση του κοινού, τα “serious games”, πολλές φορές λοιπόν αυτά τα εκπαιδευτικά παιχνίδια αποτυγχάνουν να κινήσουν το ενδιαφέρον στις μικρές ηλικίες γιατί τους λείπει το κομμάτι της διασκέδασης. Συχνά αναπτύσσονται εκπαιδευτικά παιχνίδια που παρόλο που υπάρχει διαδραστικότητα με τους μαθητές στους οποίους απευθύνονται, δεν έχουν το υπόβαθρο της διασκέδασης που κάνει ένα βιντεοπαιχνίδι ενδιαφέρον, με αποτέλεσμα οι μαθητές να δοκιμάζουν για μερικά λεπτά την εφαρμογή και ύστερα να μην την ξαναχρησιμοποιούν. Επομένως, στόχος μας είναι η ανάπτυξη ενός σοβαρού εκπαιδευτικού βιντεοπαιχνιδιού που θα ανταποκρίνεται στις προσδοκίες που έχουν οι μαθητές για ένα βιντεοπαιχνίδι με κύριο θέμα την ανακύκλωση και σωστή εκμάθηση περιβαλλοντικής αγωγής και βιωσιμότητας. Εντός αυτού θα προσπαθήσουμε να εντάξουμε διάφορα περιβαλλοντικά στοιχεία όπως διαδραστικά παζλ με σκοπό την εκμάθηση σημασίας μείωσης των απορριμμάτων και παιχνίδια επιλογών με επεξήγηση συνέπειων της επιλογής που έκανε ο παίχτης στο περιβάλλον. Τέλος θα αξιολογήσουμε την προσπάθεια μας αλλά και θα προτείνουμε τρόπους βελτίωσης του παιχνιδιού εκ μέρους μας αλλά και μέσα από αξιολογήσεις από τρίτους.

1.2 Ανάλυση Δομής Εργασίας

Η εργασία αυτή θα είναι χωρισμένη σε τέσσερα κεφάλαια. Αρχικά, στο πρώτο κεφάλαιο θα υπάρχει μια γενική εισαγωγή για τον σκοπό της εργασίας με σκοπό την προετοιμασία του αναγνώστη για τα επερχόμενα. Κατόπιν στο δεύτερο κεφάλαιο θα επεξηγήσουμε τι είναι ένα βιντεοπαιχνίδι και πως αυτό αναπτύσσεται μέσω ιστορικών αναδρομών και προβολής παραδειγμάτων βιντεοπαιχδιών του παρελθόντος. Ύστερα θα αναλύσουμε γιατί είναι σημαντικό να έχουμε περιβαλλοντική συνείδηση και να ανακυκλώνουμε μέσω προβολής τρεχόντων περιβαλλοντικών γεγονότων και ζητημάτων. Επακολούθως, το τρίτο κεφάλαιο θα περιέχει την ανάπτυξη του δικού μας εκπαιδευτικού βιντεοπαιχνιδιού, αναλύοντας τους τρόπους ανάπτυξης κάθε πτυχής του αλλά και των προβλημάτων που αντιμετωπίσαμε. Τέλος στο τέταρτο και τελευταίο κεφάλαιο θα εκτελέσουμε το παιχνίδι και θα αναφέρουμε τρόπους που θα μπορούσε να γίνει καλύτερο.

2. Θεωρητικό Μέρος

2.1 Βιντεοπαιχνίδια

Ο όρος βιντεοπαιχνίδι αναφέρεται σε οποιοδήποτε παιχνίδι πραγματοποιείται μέσω κάποιας ηλεκτρονικής συσκευής όπως ηλεκτρονικός υπολογιστής, κονσόλα βιντεοπαιχνιδιών ή πιο πρόσφατα, κινητά τηλέφωνα [1]. Όλα τα βιντεοπαιχνίδια διαθέτουν έναν τρόπο αλληλεπίδρασης με το χρήστη όπως πληκτρολόγιο και ποντίκι ή gamepad (χειριστήριο βιντεοπαιχνιδιών συνήθως διαθέσιμο σε κονσόλες) καθώς και έναν τρόπο αναπαραγωγής του περιεχομένου στον χρήστη, συνήθως μέσω κάποιου είδους οθόνης και ηχείων.

Όπως αναφέρθηκε παραπάνω υπάρχουν πολλές πλατφόρμες αναπαραγωγής βιντεοπαιχνιδιών με τις πιο δημοφιλείς στις μέρες μας να είναι οι παιχνιδοκονσόλες[2]. Ως παιχνιδοκονσόλα ορίζεται οποιαδήποτε ηλεκτρονική συσκευή που έχει ως μοναδικό σκοπό την αναπαραγωγή βιντεοπαιχνιδιών για την ψυχαγωγία του χρήστη, αυτή μπορεί να είναι είτε σταθερή κονσόλα σπιτιού (όπως PlayStation, XBOX, Nintendo Wii κ.α.) είτε φορητή (Nintendo 3DS, PlayStation Portable, Game Boy) αλλά και ένα υβριδικό είδος σταθερής και φορητής κονσόλας που έχει αναπτυχθεί πρόσφατα (Nintendo Switch, Nintendo Wii U) όπου έχει τα προνόμια και των δυο ειδών.

2.1.1 Μια Ιστορική αναδρομή

Τα πρώτα βιντεοπαιχνίδια αναπτυχθήκαν και είδαν άνθιση την δεκαετία του ‘70 με την κυκλοφορία του πρώτου βιντεοπαιχνιδιού arcade με όνομα «Computer Space», ένα παιχνίδι διαστημικής μάχης, το 1971. Ένα χρόνο αργότερα η Atari κυκλοφόρησε το πρώτο επιτυχημένο βιντεοπαιχνίδι, το «Pong», ένα απλό βιντεοπαιχνίδι που αναπαριστά το παιχνίδι του ping-pong σε ηλεκτρονική μορφή μέσω απλών δισδιάστατων γραφικών που διανεμήθηκε μέσω 19.000 arcade cabinets. Αργότερα η κυκλοφορία του «Space Invaders» το 1978 έδωσε ζωή σε μια νέα «χρυσή» εποχή για τα βιντεοπαιχνίδια με την χρήση νέων τεχνολογιών ανάπτυξης τους με λιγότερο κόστος αλλά και την προσθήκη χρώματος σε αυτά.

Την δεκαετία του ‘80 τα βιντεοπαιχνίδια είχαν φτάσει στο αποκορύφωμα της χρυσής εποχής τους ξεπερνώντας σε χρηματικά έσοδα την ποπ μουσική και τις ταινίες του Hollywood μαζί, με το πιο επιτυχημένο βιντεοπαιχνίδι της εποχής να είναι φυσικά το «Pac-Man» της Namco με περισσότερες από 350.000 πωλήσεις σε arcade cabinets. Την ίδια εποχή αναπτυχθήκαν υπολογιστές σπιτιού με σκοπό την αναπαραγωγή βιντεοπαιχνιδιών όπως ο Commodore 64 και ο ZX Spectrum με τα βιντεοπαιχνίδια να μπορούν να παιχτούν στο σπίτι του κάθε χρήστη μέσω κασετών σε αντίθεση με το σύνηθες της εποχής όπου ήταν αναγκαίο ένα cabinet.

Το 1983, η βιομηχανία της Βόρειας Αμερικής κατέρρευσε λόγω της παραγωγής πάρα πολλών κακώς ανεπτυγμένων παιχνιδιών (ποσότητα πάνω από ποιότητα). Ο κλάδος θα αναζωογονηθεί τελικά με την κυκλοφορία του Nintendo Entertainment System (NES), το οποίο είχε ως αποτέλεσμα η αγορά οικιακών κόνσολών να κυριαρχείται από ιαπωνικές εταιρείες όπως η Nintendo, ενώ μια επαγγελματική ευρωπαϊκή βιομηχανία βιντεοπαιχνιδιών

άρχισε επίσης να διαμορφώνεται με εταιρείες όπως η Ocean Software και η Gremlin Interactive.

Την επόμενη δεκαετία τα βιντεοπαιχνίδια εξελίχθηκαν δραστικά μιας και για πρώτη φορά από τα δισδιάστατα γραφικά πέρασαν στα τρισδιάστατα, με την χρήση των οποίων δημιουργήθηκαν πολλά βιντεοπαιχνίδια της εποχής που είναι ευρέως γνωστά μέχρι και σήμερα, όπως το Doom, Legend of Zelda και το Super Mario 64. Σε αυτή την δεκαετία εδραιώθηκε η θέση της Sony στην αγορά των παιχνιδοκονσολών με την κυκλοφορία του PlayStation το 1994.

Αργότερα στην δεκαετία του 2000 είδαμε και την κυκλοφορία του XBOX για πρώτη φορά μια νέα κονσόλα που θα δημιουργήσει ανταγωνιστικότητα στο διπώλιο της σκηνής των βιντεοπαιχνιδιών με κύριους παίχτες μέχρι στιγμής την Nintendo με το GameCube και την Sony με το PlayStation 2. Μαζί με την ραγδαία ανάπτυξη του ανερχόμενου τότε Internet για πρώτη φορά είδαμε την ανάπτυξη παιχνιδιών πολλών παιχτών μέσω αυτού, κάτι το οποίο έγινε δεδομένο στις επόμενες γενιές παιχνιδιών αλλά και στην καθημερινότητα μας.

Τέλος στις επόμενες δεκαετίες υπήρξε μια λιγότερο ραγδαία ανάπτυξη της βιομηχανίας των βιντεοπαιχνιδιών μιας και είχαν ήδη εδραιωθεί τα τρισδιάστατα γραφικά στον χώρο και με κάθε νέα έκδοση των παιχνιδοκονσολών, αλλά και νέων τεχνολογιών στη σκηνή των προσωπικών υπολογιστών, είχε ως αποτέλεσμα την βελτιστοποίηση των γραφικών των παιχνιδιών. Με δυο νέους ανταγωνιστές να εμφανίζονται στην αγορά, τα βιντεοπαιχνίδια κινητών smartphone, αλλά και των παιχνιδιών εικονικής πραγματικότητας έχουμε μια ολοκληρωμένη σκηνή πλατφόρμων των βιντεοπαιχνιδιών που εξακολουθούν να υπάρχουν μέχρι και σήμερα.

2.1.2 Κατηγορίες Βιντεοπαιχνιδιών

Κάθε βιντεοπαιχνίδι έχει συνήθως ένα συγκεκριμένο είδος στο οποίο ανήκει βάση του περιεχομένου του, την πλατφόρμα στην οποία έχει κυκλοφορήσει ή αν είναι ενός ή πολλών παιχτών. Βέβαια όλοι αυτοί οι διαχωρισμοί σε είδη αναφέρονται στα βιντεοπαιχνίδια ψυχαγωγίας, έτσι έχουμε την πιο σημαντική κατηγοριοποίηση που είναι φυσικά τα παιχνίδια ψυχαγωγίας (entertainment games) με κύριο σκοπό την ευχαρίστηση του παίχτη και τα σοβαρά παιχνίδια (serious games) που έχουν σαν στόχο την εκπαίδευση/ενημέρωση του παίχτη.

Τα παιχνίδια ψυχαγωγίας αποτελούν την πλειοψηφία των βιντεοπαιχνιδιών μιας και παρέχουν ευχαρίστηση στον παίχτη μέσω των δραστηριοτήτων τους με σκοπό την αποστασιοποίηση του από την πραγματικότητα έστω και για λίγο χρονικό διάστημα παρέχοντας μια διέξοδο σε ένα πιο ενδιαφέρον εικονικό περιβάλλον. Αυτό συνήθως επιτυγχάνεται μέσω ενδιαφερουσών ιστοριών ή/και δημιουργικών τροπών παιχνιδιού (gameplay).

2.1.3 Τι είναι τα serious games;

Η χρήση παιχνιδιών στις εκπαιδευτικές διαδικασίες ασκείται από τον 20ο αιώνα. Συγκεκριμένα η χρήση εκπαιδευτικών παιχνιδιών με βάση το χαρτί έγινε δημοφιλής τις δεκαετίες του 1960 και 1970 μιας και η βιομηχανία των βιντεοπαιχνιδιών δεν είχε ακόμη εδραιωθεί. Με τον ερευνητή Clark C. Abt να επινοεί τον όρο «σοβαρό παιχνίδι» την δεκαετία του 1970, ορίστηκε ως τα παιχνίδια που «έχουν ένα σαφή και προσεκτικά μελετημένο εκπαιδευτικό σκοπό και δεν προορίζονται κυρίως για διασκέδαση», κάτι το οποίο βέβαια δεν σημαίνει ότι δεν πρέπει να είναι διασκεδαστικά πάραυτα[6].

Έτσι η ανάγκη διαφοροποίησης των επιμορφωτικών παιχνιδιών σε σχέση με τα παιχνίδια διασκέδασης έδωσαν ζωή σε ένα νέο ορό παιχνιδιών που αφορούσαν την επιμόρφωση του κοινού. Αυτό ακριβώς είναι και ένα serious game, ένα παιχνίδι που έχει σκοπό την επιμόρφωση του παίχτη σε αντίθεση με τα κλασσικά βίντεο παιχνίδια που έχουν σκοπό την διασκέδαση. Με τα σοβαρά παιχνίδια (serious games) ο παίχτης ναι μεν διασκεδάζει παίζοντας τα αλλά τα παιχνίδια αυτά έχουν ένα πιο βαθύ υπόβαθρο με στόχο την εκμάθηση του. Με αυτά τα παιχνίδια μπορεί κάποιος να ενημερωθεί ή/και να επιμορφωθεί σε μείζονα και ανερχόμενα θέματα του κόσμου μας ακόμα και από μικρή ηλικία μιας και παρουσιάζονται σε μορφή παιχνιδιού.

Ένα πολύ καλό παράδειγμα εκπαιδευτικού παιχνιδιού αποτελεί το Flight Simulator [7] της Microsoft μιας και έχει πολλές εκδόσεις από το 1982 έως το 2020, και αποτελεί ένα παιχνίδι προσομοίωσης πτήσης για ερασιτέχνες με πολλά αεροσκάφη και ακριβείς τοποθεσίες σε όλο τον κόσμο μιας και προσομοιώνει την τοπογραφίας όλης της γης αλλά παρέχει και στοιχεία σε πραγματικό χρόνο όπως καιρικές συνθήκες και εναέρια κυκλοφορία. Με τον παίχτη να αντιμετωπίζει πολλές προκλήσεις όπως η απογείωση και η προσγείωση ενός αεροσκάφους με ρεαλιστικά φυσικά φαινόμενα μέσω της μηχανής φυσικής του παιχνιδιού αλλά και ρεαλιστικών μοντέλων και οργάνων κάθε αεροσκάφους, αποτελεί ένα καλό πρώτο βήμα για οποιονδήποτε θέλει να αποκτήσει δίπλωμα πιλότου.

Εικόνα 2.1 Microsoft Flight Simulator 2020 – Ένα παράδειγμα πτήσης στον ουρανό της Νέας Υόρκης

2.1.4 Κατηγορίες Σοβαρών Βιντεοπαιχνιδιών

Όπως τα βιντεοπαιχνίδια ψυχαγωγίας έχουν κατηγορίες, έτσι και τα σοβαρά βιντεοπαιχνίδια μπορούν να κατηγοριοποιηθούν ανάλογα με το περιεχόμενο τους και τον τελικό σκοπό που έχουν στην εκμάθηση του παίχτη. Μερικά παραδείγματα κατηγοριών είναι:

- **Εκπαιδευτικά παιχνίδια (Educational games):** Παιχνίδια που έχουν σχεδιαστεί για να διδάξουν ή να ενισχύσουν έννοιες σε μια συγκεκριμένη θεματική περιοχή, όπως μαθηματικά ή επιστήμες.

- **Παιχνίδια υγείας (Health games):** Παιχνίδια που έχουν σχεδιαστεί για να προωθούν υγιεινές συμπεριφορές ή να βοηθούν στη διαχείριση μιας κατάστασης υγείας, όπως παιχνίδια άσκησης ή παιχνίδια που βοηθούν άτομα με χρόνιες ασθένειες να διαχειριστούν τη θεραπεία τους.

- **Προπονητικά παιχνίδια (Training games):** Παιχνίδια που έχουν σχεδιαστεί για να διδάξουν ή να εξασκήσουν συγκεκριμένες δεξιότητες, όπως η οδήγηση ή οι πρώτες βοήθειες.

- **Παιχνίδια πειθούς (Persuasive games):** Παιχνίδια που έχουν σχεδιαστεί για να επηρεάζουν στάσεις ή συμπεριφορές, όπως παιχνίδια που προωθούν την περιβαλλοντική συνείδηση ή ενθαρρύνουν τους ανθρώπους να ψηφίσουν.

- **Παιχνίδια προσομοίωσης (Simulation games):** Παιχνίδια που έχουν σχεδιαστεί για να μιμούνται πραγματικές καταστάσεις ή διαδικασίες, όπως στρατιωτικές προσομοιώσεις ή παιχνίδια οικοδόμησης πόλεων.

Εκτός από τη κατηγοριοποίηση με βάση το περιεχόμενο, τα serious games μπορούν να χωριστούν και σε δυο βασικές κατηγορίες με βάση τη δομή και τον στόχο τους. Συγκεκριμένα χωρίζονται σε παιχνίδια:

- **Προσανατολισμένα στη διαδικασία (Process-oriented games),** όπου δεν έχουν κάποιο συγκεκριμένο τελικό στόχο όπως τα παιχνίδια λήψης αποφάσεων, εξερεύνησης και προσομοίωσης. Με αυτόν τον τύπο παιχνιδιών ο παίχτης καθ' όλη την διάρκεια του παιχνιδιού αντλεί την γνώση πάνω στο αντικείμενο του παιχνιδιού απλά εκτελώντας τις δραστηριότητες του. Για παράδειγμα κατά την διάρκεια της πτήσης του αεροπλάνου στο Microsoft Flight Simulator που αναφέρθηκε παραπάνω.

- **Προσανατολισμένα στο αποτέλεσμα (Outcome-oriented games),** όπου μέσω των δραστηριοτήτων του το παιχνίδι οδηγεί τον παίχτη στο αποτέλεσμα όπου είναι η κύρια εστίαση. Οι δραστηριότητες αυτές επικεντρώνονται στην απόκτηση μιας δεξιότητας με στόχο ο εκπαιδευόμενος να επιδείξει μια κατανόηση των πληροφοριών στον τέλος της δραστηριότητας. Σε αυτά τα παιχνίδια συμπεριλαμβάνονται τα παιχνίδια εκπαίδευσης και πειθούς.

2.2 Ανάπτυξη Βιντεοπαιχνιδιών

Για την ανάπτυξη ενός βιντεοπαιχνιδιού απαιτείται συνδυασμός και συντονισμός πολλών τομέων επιστήμης και τέχνης. Από την ανάπτυξη κώδικα για τις λειτουργίες του βιντεοπαιχνιδιού, την ηχοληψία, την δημιουργία μοντέλων και περιβάλλοντος από καλλιτέχνες παιχνιδιών (video game artist) αλλά και την ανάπτυξη των επιπέδων του παιχνιδιού, όλα αυτά συνάγουν στην δημιουργία και σωστή λειτουργία ενός βιντεοπαιχνιδιού. Γι' αυτό τον λόγο πρέπει να υπάρχει τέλεια συνεργασία μεταξύ των παραπάνω τομέων ώστε να είναι ανταποκρίνεται στην ανάγκη των παιχτών για ευχαρίστηση και διασκέδαση. Μέχρι ένα στάδιο της εξέλιξης των βιντεοπαιχνιδιών η σχετικά μειωμένη πολυπλοκότητά τους έδινε τη δυνατότητα ανάπτυξης παιχνιδιών ακόμα και από μεμονωμένους προγραμματιστές. Ωστόσο η εξέλιξή τους οδήγησε στην ανάγκη τέλειας συνεργασίας και αυξημένων τεχνικών μέσων, κάτι που σήμαινε ότι από ένα σημείο και μετά και για πολλά χρονιά η ανάπτυξη βιντεοπαιχνιδιών ήταν κάτι δύσκολο έως και ακατόρθωτο από μικρές ομάδες προγραμματιστών με αποτέλεσμα μόνο τα μεγάλα στούντιο να μπορούν να παράγουν βιντεοπαιχνίδια με μεγάλη απήχηση. Αυτό άλλαξε με την εμφάνιση των μηχανών παιχνιδιών (game engines) κάνοντας την βασική ανάπτυξη ενός βιντεοπαιχνιδιού αρκετά πιο εύκολη για μικρές ομάδες. Έτσι με την βοήθεια των μηχανών όπως της Unity και της Unreal Engine υπήρξε ένα κύμα ραγδαίας ανάπτυξης ανεξάρτητων βιντεοπαιχνιδιών χιλιάδες από τα οποία είδαν επιτυχία. Υπάρχουν ακόμα και περιπτώσεις βιντεοπαιχνιδιών που αναπτύχθηκαν από ένα μόνο άτομο, μιας και οι γνώσεις προγραμματισμού και η βοήθεια της μηχανής βιντεοπαιχνιδιού ήταν αρκετές για την ανάπτυξη του αποτελεσματικά και ευκολότερα από ποτέ.

2.2.1 Δομή Βιντεοπαιχνιδιών

Τα βιντεοπαιχνίδια αποτελούνται από πολλά αλληλένδετα στοιχεία, οπού το καθένα από αυτά συμβάλλει στην εμπειρία του παιχνιδιού. Θα αναφέρουμε τα κυριότερα από αυτά και την συνεισφορά του καθενός στο σύνολο του παιχνιδιού.

Αρχικά έχουμε την εμπειρία παιχνιδιού (gameplay) οπού συμπεριλαμβάνονται οι μηχανισμοί του παιχνιδιού, όπως για παράδειγμα οι κανόνες και οι αλληλεπιδράσεις που καθορίζουν το πώς θα παιχτεί αλλά και ποιος είναι ο σκοπός του. Οι κανόνες περιλαμβάνουν συχνά νόμους της φυσικής που εξασφαλίζουν ρεαλισμό στην κίνηση και τις αλληλεπιδράσεις μεταξύ των αντικειμένων και των χαρακτήρων. Ένα συχνά χρησιμοποιούμενο εργαλείο για παράδειγμα αποτελεί ο "Collider", όπου είναι ένα αόρατο πλέγμα γύρω από τα μοντέλα αντικειμένων ή/και χαρακτήρων το οποίο ορίζει το σχήμα τους για την ανίχνευση των φυσικών συγκρούσεων μεταξύ τους.

Εικόνα 2.2 Παράδειγμα κυλινδρικού Collider σε μοντέλο παίκτη

Ύστερα έχουμε το κομμάτι των γραφικών που αποτελούν την κυριά οπτική αλληλεπίδραση στην εμπειρία του παίκτη αλλά και το κομμάτι που θα συμβάλλει στην αξέχαστη εμπειρία του παιχνιδιού. Συμπεριλαμβάνει τα οπτικά στοιχεία όπως χαρακτήρες περιβάλλοντα και εφέ κίνησης αλλά και το καλλιτεχνικό ύφος (art style) του παιχνιδιού οπού καθορίζει την οπτική ταυτότητα του παιχνιδιού.

Επακολούθως έχουμε το ακουστικό κομμάτι του παιχνιδιού, κάτι το οποίο είναι στενά συνδεδεμένο με το οπτικό κομμάτι που αναφέραμε παραπάνω. Σε αυτή την κατηγορία συμπεριλαμβάνεται η μουσική του παιχνιδιού όπου δίνει την σωστή ατμόσφαιρα σε αυτό αλλά και τα ακουστικά εφέ που συμβάλλουν στην δραματικότητα κάθε ενέργειας του παίκτη αλλά και στη εμβύθιση (immersion) του.

Ένα ακόμα σημαντικό στοιχείο είναι η ιστορία του παιχνιδιού και ο τρόπος με τον οποίο την αφηγείται, συμπεριλαμβάνεται η πλοκή του παιχνιδιού, οι χαρακτήρες που συναντάει ο παίκτης, συνήθως με τη μορφή NPC (Non Playable Characters), δηλαδή χαρακτήρων που δεν χειρίζεται ο παίκτης και της αλληλεπιδράσεις με αυτόν μέσω διαλόγων.

Τέλος έχουμε τον σχεδιασμό επιπέδων (Level Design) και αλληλεπίδραση του παιχνιδιού, που αποτελούνται από τα περιβάλλοντα που εκτυλίσσεται το παιχνίδι, κάτι αρκετά αλληλένδετο με τα γραφικά και την ιστορία του παιχνιδιού αλλά και την ροή του. Στην αλληλεπίδραση αντίστοιχα συμπεριλαμβάνεται ο χειρισμός του χαρακτήρα του παίκτη με την χρήση μέσων όπως πληκτρολόγια, ποντίκια και χειριστήρια αλλά και την απόκριση του παιχνιδιού στις ενέργειες που κάνει ο παίκτης.

Όλα τα παραπάνω δένονται μεταξύ τους μέσω της διεπαφής χρήστη (UI) όπου αποτελεί το κύριο αντικείμενο που βλέπει και αλληλεπιδρά ο παίκτης όπως τα μενού που συμπεριλαμβάνουν στοιχεία πλοήγησης ρυθμίσεις και λειτουργίες εντός του παιχνιδιού αλλά και το HUD (Heads Up Display), δηλαδή τα στοιχεία που εμφανίζονται στην οθόνη κατά την διάρκεια του παιχνιδιού και παρέχουν απαραίτητες πληροφορίες στον παίκτη για την εκπλήρωση της αποστολής του.

2.2.2 Μηχανή Βιντεοπαιχνιδιών (Game Engine)

Η μηχανή παιχνιδιών είναι ένα περιβάλλον ανάπτυξης λογισμικού που έχει σχεδιαστεί από τους ανθρώπους για την δημιουργία βιντεοπαιχνιδιών [10]. Παρέχει ένα πλαίσιο στους προγραμματιστές, επιτρέποντάς τους να επικεντρωθούν στη δημιουργία των μοναδικών πτυχών του παιχνιδιού τους, ενώ η μηχανή χειρίζεται τις πιο τεχνικές πτυχές, όπως η απόδοση γραφικών και ο χειρισμός της λογικής του παιχνιδιού. Μερικά παραδείγματα μηχανών παιχνιδιών περιλαμβάνουν τη Unity, τη Unreal Engine και την CryEngine. Συνήθως περιέχει μια σουίτα οπτικών εργαλείων με σκοπό την απλοποιημένη υλοποίηση του βιντεοπαιχνιδιού στους τομείς των γραφικών σε δισδιάστατο (2D) ή και τρισδιάστατο (3D) χώρο, ήχου, δικτύωσης και φυσικής παιχνιδιού (in game physics). Τέλος πάντα συνοδεύονται από μια γλώσσα προγραμματισμού, συνήθως C++, C# ή JavaScript, στην οποία είναι βασισμένη η μηχανή και με αυτή την γλώσσα υλοποιούνται διαφορά σενάρια (scripts) από τον προγραμματιστή, συνήθως για την αλληλεπίδραση του παίχτη με το περιβάλλον.

Στις αρχές της βιομηχανίας των βιντεοπαιχνιδιών σχεδόν κάθε παιχνίδι που κυκλοφορούσε είχε την δικιά του μηχανή βιντεοπαιχνιδιού που ήταν σχεδιασμένη για αυτό το συγκεκριμένο βιντεοπαιχνίδι ή και για κάποιο επακόλουθο (sequel) της ίδιας σειράς. Όσο περνούσαν τα χρόνια και η βιομηχανία εξελισσόταν κυκλοφορούσαν όλο και πιο πολλά βιντεοπαιχνίδια με αποτέλεσμα να υπάρχει μια πληθώρα μηχανών που βρισκόνταν σε αχρηστία μετά την παραγωγή του βιντεοπαιχνιδιού για το οποίο προορίζονταν, για παράδειγμα η μηχανή “id Tech” με την οποία αναπτύχθηκε το γνωστό βιντεοπαιχνίδι “Doom” το 1993 και το επακόλουθο “Doom II” το 1994. Οπότε το 1998 δημιουργήθηκε και κυκλοφόρησε η “Unreal Engine” [11], μια μηχανή βιντεοπαιχνιδιών που είχε σαν στόχο την δημιουργία πολλών διαφορετικών βιντεοπαιχνιδιών οπού αντί να χρειάζεται να δουλέψουν οι προγραμματιστές από την αρχή και να δημιουργήσουν όλες τις πτυχές του νέου βιντεοπαιχνιδιού με αυτή την μηχανή απλά αδειοδοτούσαν τα βασικά τμήματα του λογισμικού και σχεδίαζαν δικά τους γραφικά, χαρακτήρες όπλα και επίπεδα, δηλαδή το περιεχόμενο του παιχνιδιού.

Εικόνα 2.3 Unreal (1998) – Παράδειγμα χρήσης της Unreal Engine 1

Με το πέρασμα του χρόνου όλο και πιο πολλές μηχανές βιντεοπαιχνιδιών εμφανίζονταν στην αγορά όπως η Unity Engine το 2005, η Gamemaker το 1999 που βασίζονταν κυρίως στα δισδιάστατα γραφικά και η CryEngine της “Crytek” το 2002 γνωστή για το βιντεοπαιχνίδι “Crysis”. Αυτή τη φορά όμως είχαν σαν σκοπό την γενική ανάπτυξη βιντεοπαιχνιδιών όπως είχε κάνει η Unreal Engine το 1998 και θεωρείται ακόμα ίσως η κορυφαία μηχανή βιντεοπαιχνιδιών αφού πολλά βιντεοπαιχνίδια βασίζονται σε αυτήν μιας και συνεχίζει να αναπτύσσεται μέχρι και σήμερα έχοντας φτάσει στην έκδοση 5.1 (15 Νοέμβριου 2022).

2.2.3 Γραφικά υπολογιστών

Τα γραφικά υπολογιστή είναι οπτικές αναπαραστάσεις δεδομένων που παράγονται από έναν υπολογιστή [14]. Μπορούν να είναι στατικά, όπως μια φωτογραφία, ή δυναμικά, όπως μια ταινία. Τα γραφικά υπολογιστών χρησιμοποιούνται σε διάφορους τομείς, όπως βιντεοπαιχνίδια, ταινίες, τηλεόραση, επιστημονική οπτικοποίηση και οπτικοποίηση πληροφοριών και μπορούν να είναι είτε δισδιάστατα (2D) είτε τρισδιάστατα (3D).

Τα δισδιάστατα γραφικά υπολογιστή είναι οπτικές αναπαραστάσεις δεδομένων που δημιουργούνται σε ένα δισδιάστατο επίπεδο. Αυτό σημαίνει ότι έχουν πλάτος και ύψος, αλλά όχι βάθος και χρησιμοποιούνται συνήθως σε εφαρμογές όπως η εκτύπωση, η σχεδίαση ιστοσελίδων (web design) και η σχεδίαση διεπαφής χρήστη (user interface design) αλλά και βασικό αρχιτεκτονικό και μηχανολογικό σχέδιο.

Τα τρισδιάστατα γραφικά υπολογιστών είναι οπτικές αναπαραστάσεις δεδομένων που δημιουργούνται σε τρισδιάστατο χώρο. Αυτό σημαίνει ότι έχουν πλάτος, ύψος και βάθος. Χρησιμοποιούνται συνήθως σε εφαρμογές όπως ταινίες, τηλεόραση, βιντεοπαιχνίδια και επιστημονική οπτικοποίηση.

Υπάρχουν πολλές διαφορετικές τεχνικές που χρησιμοποιούνται για τη δημιουργία γραφικών υπολογιστών και οι συγκεκριμένες τεχνικές που χρησιμοποιούνται μπορούν να εξαρτηθούν από την εφαρμογή και το επιθυμητό τελικό αποτέλεσμα. Μερικές κοινές τεχνικές περιλαμβάνουν:

- **Γραφικά ράστερ (Raster Graphics):** Αυτός είναι ένας τύπος γραφικών υπολογιστή που χρησιμοποιεί ένα πλέγμα εικονοστοιχείων (pixel) για να αναπαραστήσει μια εικόνα. Τα γραφικά ράστερ εξαρτώνται από την ανάλυση, πράγμα που σημαίνει ότι χάνουν την ποιότητα όταν αλλάζουν το μέγεθός τους. Οι συνήθεις μορφές αρχείων για στατικά γραφικά ράστερ περιλαμβάνουν JPEG, PNG και GIF. Αξίζει να σημειωθεί ότι, πρακτικά, με δεδομένο ότι οι σύγχρονες οθόνες στις οποίες προβάλλονται τα γραφικά είναι δομημένες σε pixels, όλοι οι άλλοι τύποι γραφικών, μετατρέπονται σε ράστερ για την τελική τους οπτικοποίηση.

- **Διανυσματικά γραφικά (Vector Graphics):** Πρόκειται για ένα είδος γραφικών υπολογιστή που χρησιμοποιεί μαθηματικές εξισώσεις για να σχεδιάσει γραμμές και σχήματα. Τα διανυσματικά γραφικά είναι ανεξάρτητα από την ανάλυση, που σημαίνει ότι μπορούν να αλλάξουν το μέγεθός τους χωρίς απώλεια ποιότητας. Οι συνήθεις μορφές αρχείων για στατικά διανυσματικά γραφικά περιλαμβάνουν SVG και EPS.

- **Γραφικά με ίχνη ακτίνων (Ray Traced Graphics):** Μια νέα τεχνική απεικόνισης γραφικών υπολογιστή που χρησιμοποιείται για τη δημιουργία εικόνων υψηλής ποιότητας προσομοιώνοντας τον τρόπο με τον οποίο το φως ταξιδεύει στον πραγματικό κόσμο. Περιλαμβάνει τον εντοπισμό της διαδρομής του φωτός καθώς αναπηδά από διάφορα αντικείμενα σε μια σκηνή και τον

υπολογισμό του τελικού χρώματος κάθε εικονοστοιχείου (pixel) στην οθόνη με βάση αυτές τις πληροφορίες.

- **Τρισδιάστατη μοντελοποίηση (3D Modeling):** Είναι η διαδικασία δημιουργίας ενός εικονικού τρισδιάστατου μοντέλου ενός αντικειμένου ή κόσμου. Τα τρισδιάστατα μοντέλα χρησιμοποιούνται συχνά σε κινούμενα σχέδια, ειδικά εφέ και βιντεοπαιχνίδια. Η τρισδιάστατη μοντελοποίηση παρουσιάζει το πλεονέκτημα της δυνατότητας παραγωγής εναλλακτικών όψεων του ίδιου αντικειμένου ή κόσμου, αλλάζοντας υπολογιστικά το σημείο και τον τρόπο θέασης.

- **Προγραμματισμός Shader (Shader Programming):** Αυτός είναι ένας τύπος προγραμματισμού υπολογιστών που χρησιμοποιείται για τη δημιουργία πολύπλοκων οπτικών εφέ σε γραφικά υπολογιστή. Οι Shaders είναι προγράμματα που εκτελούνται στη μονάδα επεξεργασίας γραφικών (GPU) και χρησιμοποιούνται για τον υπολογισμό του τελικού χρώματος κάθε εικονοστοιχείου (pixel) στην οθόνη επεμβαίνοντας πρακτικά στα τελικά στάδια οπτικοποίησης και με δυνατότητα για αυξημένο βαθμό λεπτομέρειας.

2.2.3.1 Οπτικοποίηση των Γραφικών

Οι παραπάνω μορφές οπτικοποίησης μέσω υπολογιστή εμπλέκουν σε διαφορετικούς βαθμούς το λεγόμενο graphics pipeline, δηλαδή την ακολουθία εργασιών για την τελική οπτικοποίηση στην οθόνη του εκάστοτε υπολογιστικού μέσου. Η διαδικασία αυτή ξεκινά με τη δημιουργία, ίσως και εκτός υπολογιστή, ενός μοντέλου του αντικειμένου, της εικόνας ή του εικονικού κόσμου που πρόκειται να απεικονιστεί. Στη συνέχεια, το μοντέλο αυτό καταχωρείται στον υπολογιστή, ανάλογα και με τη μορφή του, ως ένα σύνολο δεδομένων, είτε, στην απλούστερη εκδοχή, απευθείας ως χρωματική πληροφορία για pixels (εικόνα ράστερ) είτε, στην πιο πολύπλοκη, ως ένα μαθηματικό μοντέλο των αντικειμένων που συνθέτουν τον εικονικό κόσμο, με τη μορφή μιας μαθηματική αναπαράσταση που χρησιμοποιεί διάφορα γεωμετρικά σχήματα όπως τρίγωνα ή γραμμές. Στη συνέχεια, ο υπολογιστής διαβάζει τα δεδομένα και, ανάλογα με το είδος τους, τα επεξεργάζεται κατάλληλα για να παράγει την εικόνα που πρέπει να απεικονιστεί στην οθόνη του υπολογιστή. Κατά την επεξεργασία αυτή, πιθανώς αξιοποιούνται τεχνικές υπολογισμού φωτισμού και σκίασης όπως η ιχνηλάτηση ακτίνων (ray tracing) που αναφέρθηκε, καταπολέμησης ειδωλισμού (anti-aliasing) για την αποφυγή φαινομένων που προκύπτουν από την ψηφιοποίηση της εικόνας κλπ. λαμβάνοντας υπόψη το επιθυμητό σημείο και τον τρόπο θέασης του εικονικού κόσμου.

Για το τελικό αποτέλεσμα, λαμβάνεται υπόψη ο διαχωρισμός της οθόνης σε ένα πλέγμα εικονοστοιχείων (pixels) και αποφασίζεται ποια εικονοστοιχεία πρέπει να συμπληρωθούν για την δημιουργία της τελικής εικόνας. Ύστερα για κάθε εικονοστοιχείο ο υπολογιστής υπολογίζει την χρωματική τιμή μέσω της μαθηματικής απεικόνισης της εικόνας με βάση την προηγούμενη επεξεργασία και πιθανές τελικές επεμβάσεις μέσω προγραμματισμού shaders.

Τέλος, τα χρωματικά δεδομένα στέλνονται σε κάθε εικονοστοιχείο της οθόνης η οποία με τη σειρά της εμφανίζει την εικόνα φωτίζοντάς τα κατάλληλα.

Αυτή η διαδικασία συμβαίνει πολύ γρήγορα με την βοήθεια του κύριου επεξεργαστή (CPU) του συστήματος όπου είναι υπεύθυνος για την εκτέλεση των εντολών υπολογισμού που γίνονται κατά τη διαδικασία προβολής της εικόνας και της μονάδας επεξεργασίας γραφικών (GPU) που είναι εξειδικευμένη για εργασίες προβολής γραφικών, επιτρέποντας έτσι την εμφάνιση της εικόνας σε πραγματικό χρόνο. Η ποιότητα της εικόνας εξαρτάται από την ανάλυση της οθόνης, που είναι ο αριθμός των pixel που περιέχει. Μια οθόνη υψηλότερης ανάλυσης θα έχει περισσότερα pixel, με αποτέλεσμα μια πιο καθαρή και λεπτομερέστερη εικόνα αλλά και αυξημένες υπολογιστικές απαιτήσεις.

2.2.4 Η Γλώσσα Προγραμματισμού C#

Η C# είναι μια δημοφιλής γλώσσα προγραμματισμού που χρησιμοποιείται ευρέως σε διάφορα περιβάλλοντα, συμπεριλαμβανομένης της ανάπτυξης παιχνιδιών με τη μηχανή Unity. Είναι μια ισχυρή γλώσσα υψηλού επιπέδου που είναι εύκολη στην εκμάθηση και στη χρήση, και έχει μια μεγάλη και ενεργή κοινότητα προγραμματιστών. Αποτελεί μια μοντέρνα αντικειμενοστραφή γλώσσα βασισμένη στην γλώσσα προγραμματισμού C της Microsoft αλλά με παραπάνω λειτουργίες και βελτιώσεις. Η πιο σημαντική βελτίωση στη γλώσσα είναι η δυνατότητα αντικειμενοστρέφειας, ένα δημοφιλές πρότυπο προγραμματισμού που επιτρέπει στους προγραμματιστές να δημιουργούν και να χειρίζονται αντικείμενα που αντιπροσωπεύουν οντότητες του πραγματικού κόσμου. Υποστηρίζει κληρονομικότητα, πολυμορφισμό και ενθυλάκωση, που είναι όλες βασικές έννοιες στον αντικειμενοστραφή προγραμματισμό.

Εκτός από την αντικειμενοστρέφεια η C# έχει υποστήριξη για προγραμματισμό με συνιστώσες λογισμικού (component oriented), όπου είναι ένα υψηλότερο επίπεδο προγραμματισμού από τον αντικειμενοστραφή μιας και επιτρέπει στους προγραμματιστές να δημιουργούν εφαρμογές λογισμικού χρησιμοποιώντας ήδη υπάρχουσες συνιστώσες αντί να πρέπει να δημιουργηθούν νέες από το μηδέν. Αυτές οι συνιστώσες αντιπροσωπεύονται από κλάσεις που μπορούν να επαναχρησιμοποιηθούν σε διαφορετικά κομμάτια μιας εφαρμογής επιτρέποντας την δημιουργία πολύπλοκων εφαρμογών πιο γρηγορά και ευκολά κάνοντας χρήση του ήδη υπάρχοντος κώδικα.

Κατά την κατασκευή εφαρμογών η C# παρέχει χειρισμό εξαιρέσεων (exception handling) και συλλογή απορριμμάτων (garbage collection) για την απελευθέρωση μνήμης από αντικείμενα που βρίσκονται σε αχρησία. Όλοι οι τύποι μεταβλητών της γλώσσας βασίζονται σε μια ρίζα αντικειμένου που κληρονομείται, με αποτέλεσμα να μοιράζεται το σύνολο των λειτουργιών αποθήκευσης και μεταφοράς τιμών. Υπάρχει και υποστήριξη μεταβλητών τύπου αναφοράς (reference type) όπου αποθηκεύουν αναφορές στα δεδομένα τους σε αντίθεση με τις κοινές μεταφορές τύπου τιμής (value type) που αναφέρθηκαν παραπάνω όπου περιέχουν απευθείας τα δεδομένα τους. Αυτό έχει ως αποτέλεσμα με τη χρήση των αναφορών να μπορούν δύο μεταβλητές να αναφέρονται στο ίδιο αντικείμενο, επομένως οι πράξεις πάνω σε μια μεταβλητή να επηρεάζουν το αντικείμενο που αναφέρεται στην άλλη μεταβλητή, κάτι το οποίο δεν συμβαίνει στις μεταβλητές τύπου τιμής (value type).

Τέλος όσον αφορά την ανάπτυξη βιντεοπαιχνιδιών, η γλώσσα προγραμματισμού C# περιέχεται στην μηχανή βιντεοπαιχνιδιών “Unity” για την ανάπτυξη σεναρίων (scripts) που μπορούν να προσαρτηθούν σε αντικείμενα του παιχνιδιού προκειμένου να τους προσδώσουν λειτουργικότητα, με την δυνατότητα να μπορούν να ενσωματωθούν και με αλλά εργαλεία και δυνατότητες της μηχανής.

2.3 Ανακύκλωση και καθημερινότητα

Κατά την πάροδο του χρόνου η ανθρωπότητα κατορθώνει να βρίσκει όλο και περισσοτέρους τρόπους εκμετάλλευσης των πόρων της Γης, αυτό από την μια βοηθάει στην βελτίωση της καθημερινής μας ζωής αλλά από την άλλη μέσω αυτής της εκμετάλλευσης σπάνιων μετάλλων και ορυκτών καυσίμων που βρίσκονται στο υπέδαφος δημιουργούμε μια σταδιακή κλιματική αλλαγή στον πλανήτη μας.

2.3.1 Κλιματική αλλαγή

Η κλιματική αλλαγή αναφέρεται σε μακροπρόθεσμες αλλαγές στις θερμοκρασίες και τα καιρικά μοτίβα [15]. Αυτές οι μετατοπίσεις μπορεί να είναι φυσικές, όπως μέσω διακυμάνσεων στον ηλιακό κύκλο. Αλλά από το 1800, οι ανθρώπινες δραστηριότητες ήταν ο κύριος μοχλός της αλλαγής του κλίματος, κυρίως λόγω της καύσης ορυκτών καυσίμων όπως ο άνθρακας, το πετρέλαιο και το φυσικό αέριο. Κατά την καύση ορυκτών καυσίμων παράγονται εκπομπές αέριων θερμοκηπίου (greenhouse gasses) που λειτουργούν σαν ένα επιπλέον στρώμα της ατμοσφαίρας που παγιδεύει τη θερμότητα του ηλίου, αυξάνοντας έτσι τη θερμοκρασία της Γης.

Μερικά παραδείγματα εκπομπών αυτών των αέριων θερμοκηπίου που συμβάλλουν στην κλιματική αλλαγή περιλαμβάνουν το διοξείδιο του άνθρακα και το μεθάνιο. Αυτά προέρχονται από τη χρήση βενζίνης για την οδήγηση αυτοκινήτου, παραγωγή προϊόντων από εργοστάσια ή άνθρακα για τη θέρμανση ενός κτιρίου, για παράδειγμα. Η αποψίλωση της γης και των δασών μπορεί επίσης να απελευθερώσει διοξείδιο του άνθρακα. Οι χώροι υγειονομικής ταφής απορριμμάτων αποτελούν σημαντική πηγή εκπομπών μεθανίου. Η ενέργεια, η βιομηχανία, οι μεταφορές, τα κτίρια, η γεωργία και η χρήση γης συγκαταλέγονται στους κύριους εκπομπούς. Αυτές οι εκπομπές συνεχίζουν να αυξάνονται, ως αποτέλεσμα η θερμοκρασία της γης έχει αυξηθεί κατά 1.1°C σε σχέση με τα τέλη του 1800, με την τελευταία δεκαετία (2011-2020) να αποτελεί την θερμότερη που έχει καταγραφεί μέχρι στιγμής.

Πολλοί πιστεύουν ότι η κλιματική αλλαγή σημαίνει μόνο αύξηση της θερμοκρασίας, αλλά αυτό είναι μόνο η αρχή μιας και η Γη αποτελείται από ένα ενιαίο οικοσύστημα οπού είναι όλα συνδεδεμένα μεταξύ τους, έτσι με την αύξησης της θερμοκρασίας δημιουργούνται νέα προβλήματα όπως μεταξύ άλλων, έντονες ξηρασίες, λειψυδρία, σοβαρές πυρκαγιές, άνοδο της στάθμης της θάλασσας, πλημμύρες, λιώσιμο των πολικών πάγων, καταστροφικές καταιγίδες και μείωση της βιοποικιλότητας.

Η κλιματική αλλαγή μπορεί να επηρεάσει την υγειά μας, την καλλιέργεια τροφίμων, την στέγαση αλλά και την ασφάλεια μας με πολλούς από εμάς να βρισκόμαστε ήδη σε πιο ευάλωτη θέση λόγω των κλιματικών επιπτώσεων. Άτομα που μένουν σε νησιά ή σε αναπτυσσόμενες χώρες βρίσκονται σε μεγαλύτερο κίνδυνο αφού συνθήκες όπως η άνοδος της στάθμης της θάλασσας και η διείσδυση του αλμυρού νερού διακινδυνεύουν την ασφάλεια των κάτοικων, με αποτέλεσμα την ανάγκη μετέγκαταστασης της κοινότητας τους. Τέτοιες περιπτώσεις «κλιματικών προσφύγων» θα συνεχίσουν να αυξάνονται όσο δεν αναλαμβάνεται δράση για την ανατροπή του φαινομένου της κλιματικής αλλαγής

2.3.2 Συμβολή της ανακύκλωσης

Με βάση τα παραπάνω καταλαβαίνουμε ότι η κλιματική αλλαγή αποτελεί ένα μεγάλο πρόβλημα στην κοινωνία μας αφού επηρεάζει το τρόπο ζωής μας αλλά και την βιωσιμότητα του πλανήτη μας. Με την εισαγωγή της ανακύκλωσης στη ζωή μας μπορούμε να μειώσουμε την παραγωγή αυτών των βλαβερών αέριων που την προκαλούν [**Error! Reference source not found.**]. Αυτό συμβαίνει διότι ανακυκλώνοντας τα αντικείμενα αυτά αντί να καταλήγουν στην χωματερή γίνεται συλλογή τους από τους ειδικούς κάδους ανακύκλωσης.

Ύστερα αυτά τα υλικά μεταφέρονται από τον συλλέκτη σε μια εγκατάσταση που γίνεται επεξεργασία για ανάκτηση υλικών οπού αυτά διαλέγονται, καθαρίζονται από ρύπους και προετοιμάζονται για μεταφορά σε άλλες μονάδες άλεσης υλικών ή και απευθείας σε μονάδες παραγωγής. Άλλα αντικείμενα μπορεί να χρειάζονται περισσότερη επεξεργασία και διαλογή, όπως ο γυαλί και το πλαστικό μιας και γίνεται εμπλουτισμός ή επεξεργασία αυτών για να καταλήξουν σε μορφές έτοιμες για άλεση.

Τέλος αφού υποστούν επεξεργασία τα υλικά αυτά γίνονται κομμάτι νέων προϊόντων σε μονάδες παραγωγής ανακυκλωμένων προϊόντων οπού πολλές φορές μπορεί να είναι και κάτι εντελώς διαφορετικό από την αρχική του μορφή. Για παράδειγμα με τη συλλογή και επεξεργασία γυαλιού μπορεί να παραχθεί άσφαλτος για την ασφάλτωση δρόμων ή με την ανακύκλωση πλαστικού μπορεί να γίνει κατασκευή χαλιών ή παγκάκια πάρκων.

Με αυτόν τον τρόπο αντιμετώπισης απορριμμάτων έχουμε όχι μόνο ασφαλή και αποδοτική διαχείριση απορριμμάτων μειώνοντας την ποσότητα τους σε δομές ΧΥΤΗ/ΧΥΤΑ, και σταδιακής κατάργησης αυτών, αλλά υπάρχει και μείωση κατανάλωσης ενέργειας κατά την παραγωγή νέων αντικειμένων εκ των οποίων τα βασικά στοιχεία υπάρχουν ήδη σε ανακυκλωμένα προϊόντα. Έτσι υπάρχει μείωση των ρίπων που επιβαρύνουν το περιβάλλον παραγωγής των εκ των εργοστασίων.

2.3.3 Ανακύκλωση και εκπαίδευση

Για να υπάρχει κάποιος αντίκτυπος στην γενική εικόνα του πλανήτη μας θα πρέπει να γίνει μια μεγάλη προσπάθεια αλλαγής της συμπεριφοράς μας ως προς την απόρριψη άχρηστων αντικειμένων. Ενώ υπάρχει ήδη μια δυναμική μείωσης των απορριμμάτων και ανακύκλωσης μπορεί να γίνει μεγαλύτερη με την βοήθεια της εκπαίδευσης.

Ενσωματώνοντας στην διδασκαλία την σημασία της ανακύκλωσης εμπνέουμε τους μαθητές να σκεφτούν πώς οι προσωπικές τους ενέργειες επηρεάζουν τον πλανήτη αλλά και το μελλοντικά δικό τους περιβάλλον. Μαθαίνοντάς τους να ανακυκλώνουν από μικρή μαθητική ηλικία υπάρχουν μεγαλύτερες πιθανότητες να το υιοθετήσουν σαν συνήθεια βοηθώντας τους να γίνουν ευσυνείδητοι ενήλικες. Αυτό ενθαρρύνει τους μαθητές να εφαρμόσουν αυτές τις τακτικές ανακύκλωσης στην καθημερινότητα τους, βοηθώντας στην αντιμετώπιση απορριμμάτων στο σπίτι.

Κατά την διδασκαλία της περιβαλλοντικής αγωγής ενθαρρύνεται η κριτική σκέψη. Οι μαθητές εξερευνούν πολύπλοκα θέματα όπως οι περιβαλλοντικές επιπτώσεις διαφορετικών υλικών, κύκλοι ζωής προϊόντων και συνέπειες ταφής απορριμμάτων σε δομές ΧΥΤΑ, μέσω αυτών των συζητήσεων προάγεται η κριτική σκέψη και οι δεξιότητες επίλυσης προβλημάτων.

Ενώ υπάρχει ήδη ένα υπόβαθρο εκπαίδευσης ως προς την ανακύκλωση και την περιβαλλοντική σκέψη, παρατηρείται ότι δεν είναι αρκετό μιας και οι μαθητές δεν δίνουν ιδιαίτερη σημασία στα μαθήματα αυτά μιας και τα θεωρούν βαρετά, ή ακόμα και «ελεύθερη ώρα» για να μπορούν να κάνουν ότι μπορούν προκειμένου αν ψυχαγωγηθούν. Μια λύση σε αυτό το πρόβλημα θα μπορούσε να αποτελέσει η χρήση των βιντεοπαιχνιδιών, μιας και είναι κάτι πλέον ευρέως διαδεδομένο και γνωστό ως ενδιαφέρον και ψυχαγωγικό.

Με την χρήση των βιντεοπαιχδιών μπορούμε να περάσουμε μηνύματα ευσυνειδησίας ως προς το περιβάλλον και υιοθέτησης τακτικών ανακύκλωσης με διασκεδαστικό τρόπο ενθαρρύνοντας τους χρήστες τους να εξερευνήσουν και να μάθουν. Τέτοιου είδους βιντεοπαιχνίδια υπάρχουν ήδη αλλά σπανίζουν στον τομέα της ανακύκλωσης. Αναζητώντας στο διαδίκτυο τέτοιου είδους βιντεοπαιχνίδια βρίσκουμε αποτελέσματα, συγκεκριμένα τα πιο γνωστά από αυτά είναι το “Litter Critters - Learn to Sort and Recycle” της “ABCYa” που αφορά την σωστή κατανομή ανακυκλώσιμων σε κάδους σε μια γραμμή παραγωγής, και το “Recycle Roundup” του National Geographic που αφορά την συλλογή και ανακύκλωση αντικειμένων σε δυσδιάστατο περιβάλλον.

Εικόνα 2.4 Recycle Roundup video game από το National Geographic

Επιπλέον, μέσω των βιντεοπαιχνιδιών παρέχεται μια διαδραστικά μαθησιακή εμπειρία όπου οι μαθητές μπορούν αν συμμετάσχουν ενεργά σε δραστηριότητες ανακύκλωσης εντός του παιχνιδιού όπως ταξινόμηση εικονικών ανακυκλώσιμων διαχείριση και αποκομιδή αυτών από εικονικά περιβάλλοντα ενισχύονται την μάθηση.

Πέρα από τα παραπάνω τα βιντεοπαιχνίδια ως τρόπος εκπαίδευσης για την ανακύκλωση, και γενικότερα, έχουν επιπρόσθετα πλεονεκτήματα, όπως:

- Βιντεοπαιχνίδια πολλών παιχτών και ανταγωνιστικά: ενθαρρύνουν την συνεργασία και τον ανταγωνισμό μεταξύ των παικτών. Με τον τρόπο αυτό, παιχνίδια με θέμα την ανακύκλωση που προορίζονται για πολλούς παίκτες προάγουν την ομαδική εργασία και συνεργασία αλλά παράλληλα παρακινούν τους μαθητές να ξεπερνούν ο ένας τον άλλον σε εργασίες που σχετίζονται με την ανακύκλωση.

- Διήγηση και αφήγηση: Τα βιντεοπαιχνίδια συχνά παρουσιάζουν συναρπαστικές ιστορίες και αφήγηση. Τα παιχνίδια με θέμα την ανακύκλωση μπορούν να ενσωματώσουν περιβαλλοντικές ιστορίες και χαρακτήρες που διδάσκουν στους παίκτες τη σημασία της ανακύκλωσης, της μείωσης των απορριμμάτων και της βιωσιμότητας μέσω συναρπαστικών αφηγήσεων.

- Ανάλυση δεδομένων: Ορισμένα εκπαιδευτικά παιχνίδια περιλαμβάνουν συλλογή και ανάλυση δεδομένων. Τα παιχνίδια ανακύκλωσης μπορούν να εισάγουν τους παίκτες στη συλλογή δεδομένων που σχετίζονται με τα ποσοστά ανακύκλωσης, τη σύνθεση των απορριμμάτων και τις περιβαλλοντικές επιπτώσεις. Η ανάλυση αυτών των δεδομένων μπορεί να βοηθήσει τους παίκτες να κατανοήσουν τη σημασία των προσπαθειών ανακύκλωσης.

- Εικονικές εκδρομές: Τα εκπαιδευτικά βιντεοπαιχνίδια μπορούν να προσφέρουν εικονικές εκδρομές σε εγκαταστάσεις ανακύκλωσης, κέντρα διαχείρισης απορριμμάτων ή κοινότητες φιλικές προς το περιβάλλον. Αυτές οι εικονικές εμπειρίες μπορούν να παρέχουν πληροφορίες για τις πραγματικές διαδικασίες και πρακτικές ανακύκλωσης.

- Διαχείριση πόρων: Πολλά παιχνίδια στρατηγικής και διαχείρισης πόρων απαιτούν από τους παίκτες να διαχειρίζονται αποτελεσματικά τους πόρους. Τα παιχνίδια με θέμα την ανακύκλωση μπορούν να διδάξουν δεξιότητες διαχείρισης πόρων προσομοιώνοντας την αποτελεσματική χρήση υλικών και στρατηγικές μείωσης των απορριμμάτων.

- Κίνητρα και ανταμοιβές: Τα βιντεοπαιχνίδια συχνά χρησιμοποιούν ανταμοιβές και επιτεύγματα για να παρακινήσουν τους παίκτες. Τα παιχνίδια ανακύκλωσης μπορούν να ενσωματώσουν ανταμοιβές για φιλικές προς το περιβάλλον ενέργειες και επιτεύγματα, ενθαρρύνοντας τους παίκτες να υιοθετήσουν βιώσιμες συνήθειες τόσο στο παιχνίδι όσο και στην πραγματική ζωή.

- Παγκόσμια προοπτική: Ορισμένα παιχνίδια έχουν παγκόσμια ή διεθνή εμβέλεια, επιτρέποντας στους παίκτες να εξερευνήσουν προκλήσεις και λύσεις ανακύκλωσης από διάφορα μέρη του κόσμου. Αυτό μπορεί να διευρύνει τις προοπτικές των παικτών για την ανακύκλωση και τη βιωσιμότητα.

Ενώ τα βιντεοπαιχνίδια μπορούν να αποτελέσουν πολύτιμο εργαλείο για την εκπαίδευση για την ανακύκλωση και την προώθηση της περιβαλλοντικής ευαισθητοποίησης, είναι απαραίτητο να εξισορροπηθεί το παιχνίδι με τις πραγματικές εμπειρίες και το εκπαιδευτικό υλικό για να διασφαλιστεί μια ολοκληρωμένη κατανόηση των εννοιών ανακύκλωσης και βιωσιμότητας. Επιπλέον, οι εκπαιδευτικοί και οι προγραμματιστές παιχνιδιών θα πρέπει να συνεργάζονται για να δημιουργήσουν αποτελεσματικές και εκπαιδευτικές εμπειρίες παιχνιδιού.

3. Σχεδιασμός & Ανάπτυξη Βιντεοπαιχνιδιού

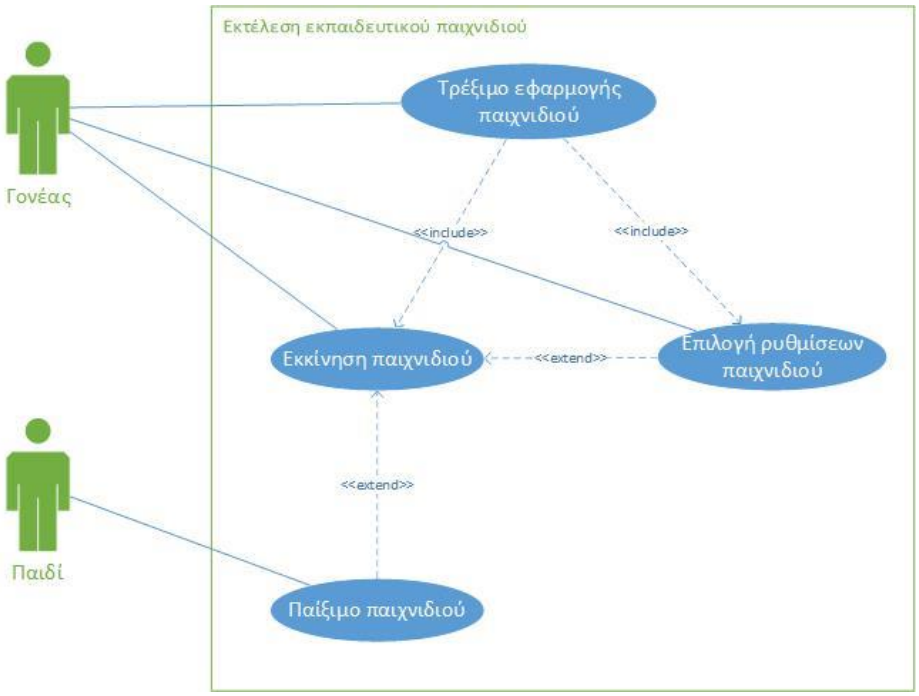
3.1 Περιγραφή

Σκοπός μας είναι η ανάπτυξη ενός εκπαιδευτικού βιντεοπαιχνιδιού εκμάθησης των βασικών σημείων της ανακύκλωσης που απευθύνεται σε μαθητές δημοτικού. Για την υλοποίηση του επιλέξαμε την χρήσης της μηχανής βιντεοπαιχνιδιών “Unity” σε τρισδιάστατη (3D) μορφή. Ο παίκτης με την βοήθεια καθοδήγησης του ίδιου του παιχνιδιού και του γονέα του καλείται να ολοκληρώσει έναν ορισμένο αριθμό αποστολών που αφορούν την διαλογή και ταξινόμηση ανακυκλώσιμων απορριμμάτων που βρίσκονται στον εικονικό χώρο του παιχνιδιού.

Αναλυτικότερα στο σύνολο του το παιχνίδι θα περιλαμβάνει:

- Αρχικό μενού και μενού επιλογής / αλλαγής ρυθμίσεων (Main Menu & Options Menu) για διαχείριση επιλογών εικόνας και ήχου καθώς και γλώσσας διαλόγου, με επιπλέον επιλογής «δύσκολης» λειτουργίας με λιγότερη καθοδήγηση ανακύκλωσης
- Τρεις αποστολές οπού η καθεμία θα περιλαμβάνει και έναν διαφορετικό τύπο ανακυκλώσιμου αντικειμένου, με μια τελική η οποία θα αναφέρεται στην ρύπανση κατά την παραγωγή αυτών των αντικειμένων.
- Μεγάλο εικονικό περιβάλλον με σκοπό την δυνατότητα εξερεύνησής του από μεριάς του παίκτη
- Καθοδήγηση του παίκτη μέσω ενός μη ελεγχόμενου από τον παίκτη χαρακτήρα ο οποίος θα δίνει και συμβουλές ανακύκλωσης.
- Αναβάθμιση λειτουργιών του παίκτη μετά από το πέρας κάποιων αποστολών ώστε να παραμένει ενδιαφέρον το παιχνίδι
- Συμπληρωματικές λεπτομέρειες για την βελτίωση του παιχνιδιού όπως: φωτισμός, μουσική, εφέ ήχου και αλληλεπίδρασης.

Η λειτουργικότητα του παιχνιδιού απεικονίζεται παρακάτω με την χρήση ενός use case diagram στο οποίο αναφέρεται η κάθε διαδικασία που πρέπει να γίνει πριν την έναρξη του παιχνιδιού, όπως η εκκίνηση της ιδίας της εφαρμογής αλλά και οι αλλαγές των ρυθμίσεων εντός αυτής.



Εικόνα 3.1 Use Case Diagram εκτέλεσης του παιχνιδιού

3.1.1 Σχεδιασμός αποστολών

Με την δημιουργία και δοκιμή ορθής λειτουργίας της διεπαφής χρήστη μπορούμε να περάσουμε στην υλοποίηση των τριών αποστολών του παίκτη. Αυτές θα περιλαμβάνουν:

Την ανακύκλωση 10 πλαστικών μπουκαλιών με σκοπό την εκμάθηση των χρωμάτων των κάδων ανακύκλωσης αλλά και ενημέρωση του παίκτη μέσω συμβουλών για το καλό που προσφέρει στον πλανήτη ανακυκλώνοντας ακόμα και αυτόν τον μικρό αριθμό πλαστικών αντικειμένων, κάτι το οποίο θα φανεί εντός παιχνιδιού με τη μείωση της ομίχλης καυσαερίου που υπάρχει στο εικονικό πάρκο που διαδραματίζεται το παιχνίδι.

Την συλλογή και ανακύκλωση 15 κουτιών αλουμινίου με σκοπό την χρήση αυτών για την δημιουργία μιας μεταλλικής γέφυρας (upcycling), έτσι ώστε να μπορέσει ο παίκτης να περάσει πάνω από την λίμνη του πάρκου με σκοπό την απόκτηση της δυνατότητας να εκτελέσει άλματα μέσω του αντικειμένου “μπότες άλματος”. Έτσι διδάσκεται την χρήση ανακυκλώσιμων αντικειμένων για την δημιουργία νέων αντικειμένων από τα κυριά υλικά τους, μειώνοντας την ανάγκη κατανάλωσης παραπάνω ενέργειας για δημιουργία των αντικειμένων αυτών εκ νέου.

Τέλος στην τρίτη αποστολή θα πρέπει να συλλέξει 4 χαρτόκουτα έτσι ώστε να μπορεί να τα επαναχρησιμοποιήσει ως σκαλοπάτια για την απενεργοποίηση παράνομης λειτουργίας ενός εργοστασίου. Με αυτόν τον τρόπο ο παίκτης μαθαίνει την διαδικασία της επαναχρησιμοποίησης αντικειμένων (reuse) και της μείωσης παραγωγής αυτών μαζί με τους ρύπους που συμπεριλαμβάνουν (reduce).

Με αυτόν τον τρόπο ο παίκτης έχει διδαχθεί μέσω του παιχνιδιού μερικές από τις βασικές αρχές περιβαλλοντικής αγωγής συμπεριλαμβανομένου και του κανόνα των τριών "R" της βιωσιμότητας (Reduce, Reuse, Recycle) [18], παρόλο που το παιχνίδι μας δίνει βάση στο τελευταίο από τους κανόνες αυτούς.

Αναλυτικότερα οι αποστολές του παίκτη έχουν ως εξής:

- **1η αποστολή**

- Στόχος:
  Συλλογή και ανακύκλωση 10 πλαστικών μπουκαλιών

- Σκοπός:
  Εκμάθηση χρωματισμού κάδων ανακύκλωσης και την βοήθεια που προσφέρει ή ανακύκλωση στο περιβάλλον όσων αφορά το πλαστικό.

Στην πρώτη αυτή αποστολή ο παίκτης πρέπει απλά να περπατήσει σε μια ευθεία όπου υπάρχουν τοποθετημένα 10 μπουκάλια και να τα συλλέξει. Τα μπουκάλια έχουν τοποθετηθεί με τρόπο έτσι ώστε να καθοδηγηθεί ο παίκτης προς το σημείο που είναι τοποθετημένοι οι κάδοι ανακύκλωσης και έτσι να μάθει για την κατηγοριοποίηση αυτών μέσω του ενημερωτικού πλαισίου κειμένου που του εμφανίζεται στην οθόνη. Τέλος αλληλεπιδρά με τον σωστό κάδο και επιστρέφει στον NPC χαρακτήρα μας, οπού με την αλληλεπίδραση με αυτόν εμφανίζεται ακόμα ένα ενημερωτικό πλαίσιο κειμένου που αφορά την συμβολή της ανακύκλωσης πλαστικού στο περιβάλλον [20].

- **2η αποστολή**

- Στόχος:
  Συλλογή και ανακύκλωση 15 αλουμινένιων κουτιών με σκοπό την κατασκευή μιας γέφυρας για την απόκτηση της δυνατότητας άλματος

- Σκοπός:
  Εκμάθηση της συμβολής ανακύκλωσης αλουμίνιου στο περιβάλλον και της δημιουργικής επαναχρησιμοποίησης (upcycling) για την δημιουργία νέων αντικειμένων από απορρίμματα.

Στην δεύτερη αποστολή θέλουμε να διδάξουμε στον παίκτη μια ακόμα χρήση της ανακύκλωσης στην καθημερινότητα μας δείχνοντας πως με την συλλογή μερικών απορριμμάτων μπορεί να δημιουργηθεί κάτι καινούργιο, πολλές φορές εντελώς διαφορετικό από το αρχικό αντικείμενο το οποίο ανακυκλώσαμε.
Έτσι με την ύπαρξη μιας αναβάθμισης κινήσεων του παίκτη σαν κίνητρο ο παίκτης συλλεγεί και ανακυκλώνει 15 αλουμινένια κουτάκια με σκοπό να φτιάξει μια γέφυρα ώστε να μπορεί να περάσει πάνω από το εμπόδιο λίμνης που υπάρχει μεταξύ αυτού και της αναβάθμισης άλματος (απεικονιζόμενη μέσα στο παιχνίδι ως μπότες άλματος).

- **3η αποστολή**

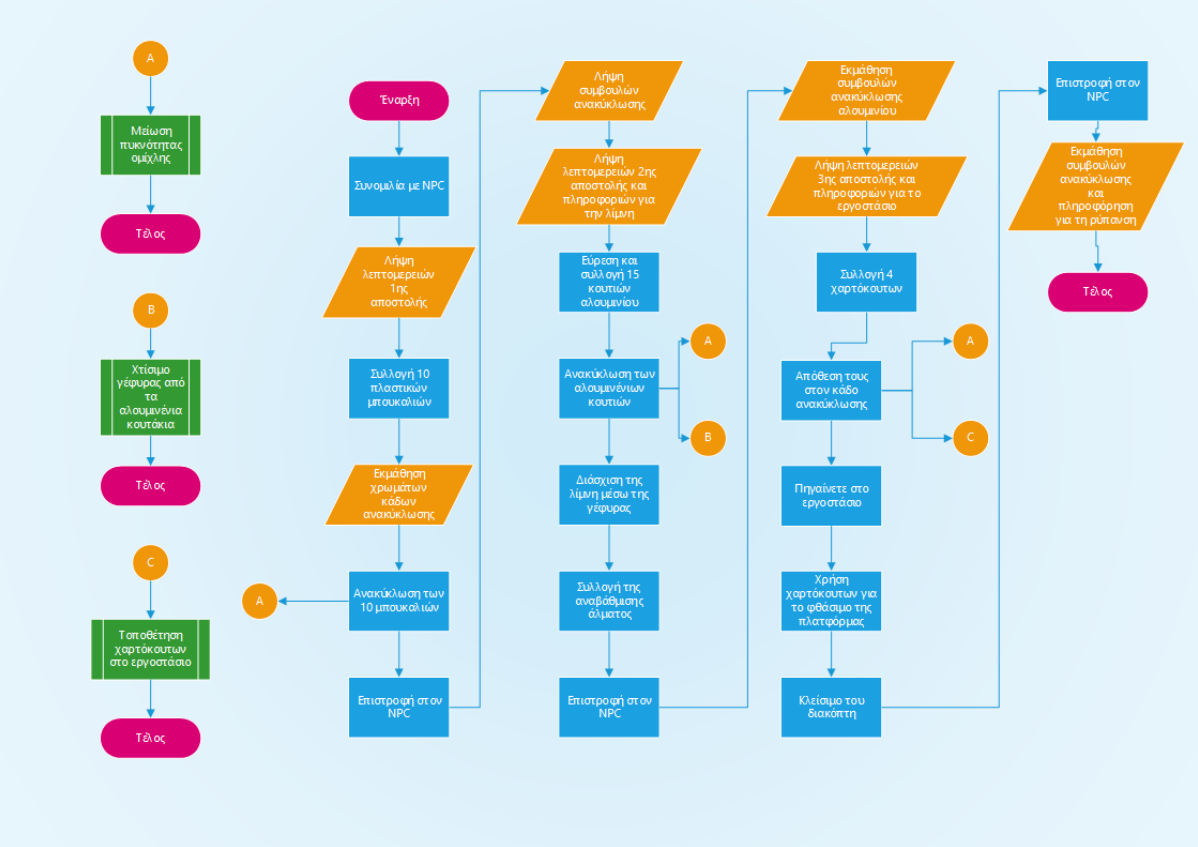
- Στόχος:

  Συλλογή 4 χαρτόκουτων με σκοπό την ανάβαση στον λεβιέ απενεργοποίησης του εργοστασίου

- Σκοπός:
  Επισήμανση ανάγκης επαναχρησιμοποίησης αντικειμένων και αποφυγής υπερβολικής παραγωγής αυτών εφόσον συνεπάγονται με καυσαέρια.

Στην τρίτη και τελευταία αποστολή ο παίκτης μαθαίνει με την συλλογή και επαναχρησιμοποίηση (reuse) των χαρτόκουτων την δυνατότητα διατήρησης κάποιον αντικειμένων για χρήση όπως σε όπως ανάγκες και λειτουργίες όπως στο παιχνίδι σαν σκαλί αναπήδησης για να φτάσει τον χώρο του εργοστασίου. Εκτός αυτού μαθαίνει την ύπαρξη καυσαερίων και παραγωγής αυτών μέσω παραγωγής αντικειμένων αλλά και τη δυνατότητας μείωσης μέσω ανακύκλωσης αφού η τελευταία συνεπάγεται ελάττωση παραγωγής νέων αντικειμένων μέσω χρήσης ήδη υπαρχόντων ανακυκλωμένων πόρων.

Αναλυτικότερα έγινε οπτικοποίηση όπως διαδικασίας ολοκλήρωσης των αποστολών με τη χρήση όπως διαγράμματος ροής (flow chart) όπως φαίνεται παρακάτω



Εικόνα 3.2 Διάγραμμα ροής βιντεοπαιχνιδιού

3.1.2 Σχεδιασμός Περιβάλλοντος

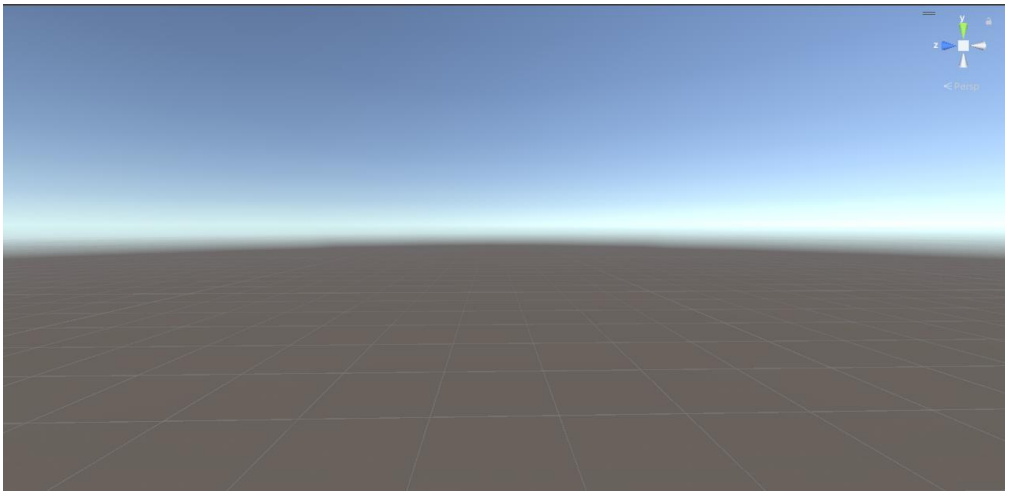
Ο χώρος στον οποίο θα λαμβάνει μέρος το βιντεοπαιχνίδι μας πρέπει να είναι συνδεδεμένος με την φύση έτσι ώστε να ενισχύει την σύνδεση καθημερινών ενεργειών και επιπτώσεων τους στο περιβάλλον στο μυαλό του παίκτη. Γι' αυτό επιλέξαμε σαν περιβάλλον παιχνιδιού ένα πάρκο αφού συνδέεται με τη φύση και το περιβάλλον αλλά επίσης αποτελεί κοινό δημόσιο χώρο που υπάρχει συγκέντρωση ανθρώπων, κάνοντας το πλαίσιο του παιχνιδιού πιο ρεαλιστικό. Το πάρκο αυτό θα έχει κάποιες υψομετρικές διαφορές όπως λόφους και λίμνες αλλά δεν παύει να είναι σημαντική και η προσθήκη οπτικοακουστικών λεπτομερειών όπως δέντρα, άνθη και φυτά αλλά και μουσική περιβάλλοντος όπως ήχους πατημασιών του παίκτη και μουσική υπόκρουση που αντιστοιχεί σε πάρκο όπως αέρας, πουλιά που κελαηδούν κλπ. Για την υποβοήθηση του παίκτη θα προστεθούν και αντίστοιχες πινακίδες που θα αναφέρουν την κατεύθυνσή που θα πρέπει να ακολουθήσει για να βρεθεί στον αντίστοιχο προορισμό.

3.2 Αναλυτική πορεία ανάπτυξης βιντεοπαιχνιδιού

Για την σωστή ανάπτυξη του βιντεοπαιχνιδιού θα πρέπει πρώτα να δημιουργήσουμε το περιβάλλον στο οποίο θα κινείται και αλληλεπιδρά ο παίχτης. Με αυτό τον τρόπο θα έχουμε την βάση για την ανάπτυξη όλων των υπολοίπων λειτουργιών του παιχνιδιού εξοικονομώντας χρόνο με τον τρόπο αυτό. Έπειτα θα γίνει προσθήκη μερικών αντικειμένων περιβάλλοντος όπως δέντρα, λουλούδια και φυτά με σκοπό τον εμπλουτισμό του. Στην συνέχεια θα δημιουργήσουμε έναν χειριστή παίχτη (character controller) για να μπορεί ο παίχτης να χειριστεί την κάμερα (δηλαδή τον κύριο χαρακτήρα εντός του παιχνιδιού) και να εξερευνήσει το εικονικό περιβάλλον. Αυτό συμπεριλαμβάνει μοντέλο χαρακτήρα, προσθήκη κίνησης και σκιών ώστε να φαίνεται αρκετά ρεαλιστική η κίνηση. Ύστερα θα γίνει εισαγωγή αντικειμένων με τα οποία θα αλληλεπιδρά ο παίχτης όπως πλαστικά μπουκάλια, χαρτιά και αλουμίνιο, αλλά και κάδων ανακύκλωσης. Αφού ολοκληρωθεί η εισαγωγή των αντικειμένων αυτών θα πρέπει να υλοποιηθεί το σενάριο κωδικοποίησης (coding script) για την αλληλεπίδραση του παίχτη με τα αντικείμενα αυτά. Αφετέρου θα προστεθεί ένας μη ελεγχόμενος χαρακτήρας (NPC) με σκοπό την καθοδήγηση του παίχτη κατά την διάρκεια των αποστολών και την ανάπτυξη των σεναρίων αλληλεπίδρασής του με τον παίχτη για καθεμιά από αυτές. Επιπρόσθετα αφού ολοκληρωθούν τα παραπάνω θα γίνει η ανάπτυξη των επιπέδων του παιχνιδιού και προστεθεί η αναβάθμιση του παίχτη που θα γίνεται στο δεύτερο επίπεδο – αποστολή αλλά και το περιβάλλον αντικειμένων για την τελευταία αποστολή μαζί με τα αντίστοιχα σενάρια αλληλεπίδρασης. Επιπρόσθετα αφού ολοκληρωθούν όλα τα παραπάνω θα γίνει ανάπτυξη των μενού (αρχικό μενού αλλά και μενού επίλογων και αλλαγής αυτών) για την δια σύνδεση όλων τον παραπάνω λειτουργειών. Τέλος θα γίνει υλοποίηση μερικών συμπληρωματικών λεπτομερών όπως ήχος βαδίσματος αλλά και μουσικής περιβάλλοντος και αρχικού μενού και η πειραματική εκτέλεση του παιχνιδιού για εύρεση και επίλυση τυχόν προβλημάτων. Η πορεία επεξηγείται αναλυτικότερα παρακάτω.

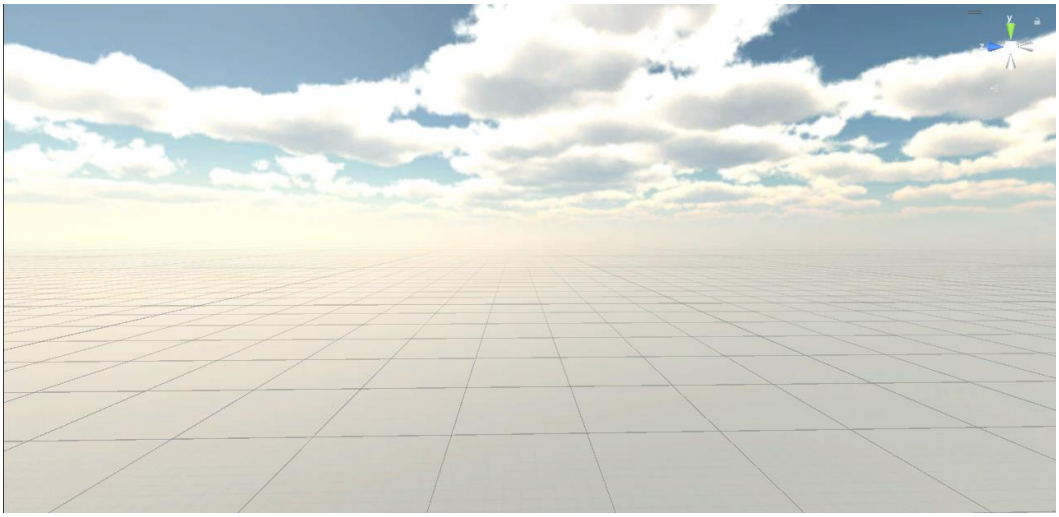
3.2.1 Δημιουργία Περιβάλλοντος Παιχνιδιού

Κατά την εκκίνηση του Unity, δημιουργώντας ένα νέο πρότζεκτ οδηγούμαστε σε μια άδεια σκηνή παιχνιδιού με την διεπαφή χρήστη όπου υπάρχουν όλα τα εργαλεία που χρειαζόμαστε για την ανάπτυξη του βιντεοπαιχνιδιού. Στην αρχική σκηνή το Unity μας δίνει μια πηγή φωτισμού και μια κάμερα κατά την αρχική της κατάσταση.



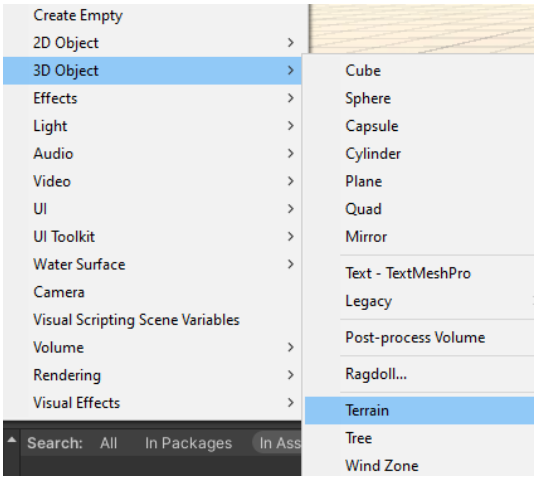
Εικόνα 3.3 Η αρχική σκηνή ενός νέου 3D πρότζεκτ στο Unity

Αρχικά προσθέτουμε ένα Skybox για αλλαγή εμφάνισης του ουρανού σε κάτι πιο ρεαλιστικό και ευπαρουσίαστο.

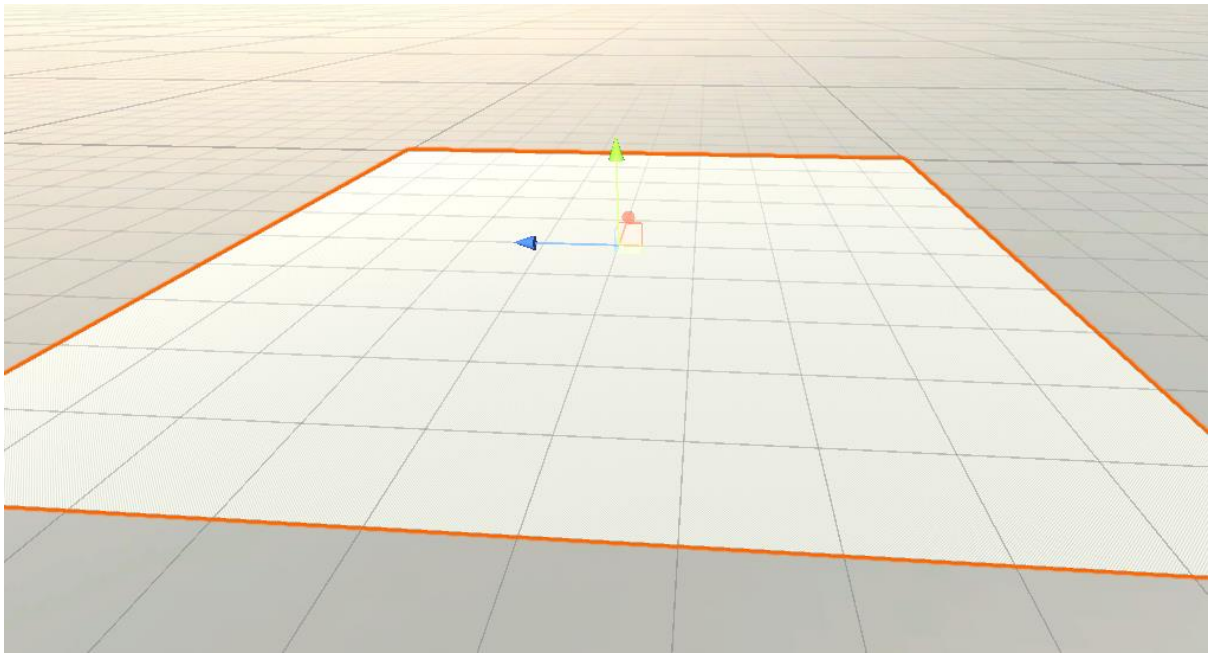


Εικόνα 3.4 Η μετατροπή εμφάνισης της σκηνής με την προσθήκη ενός Skybox

Στη συνέχεια, θα ξεκινήσουμε την αρχική υλοποίηση του περιβάλλοντος του παιχνιδιού δημιουργώντας πρώτα το έδαφος στο οποίο θα κινείται ο παίκτης. Αφετέρου θα κάνουμε μερικές αλλαγές ως προς το υψόμετρο του εδάφους για να υπάρχει έμφαση στα μονοπάτια που θα δημιουργήσουμε για τις αποστολές του παιχνιδιού, κάνοντας το πιο ευπαρουσίαστο με την χρήση λόφων.

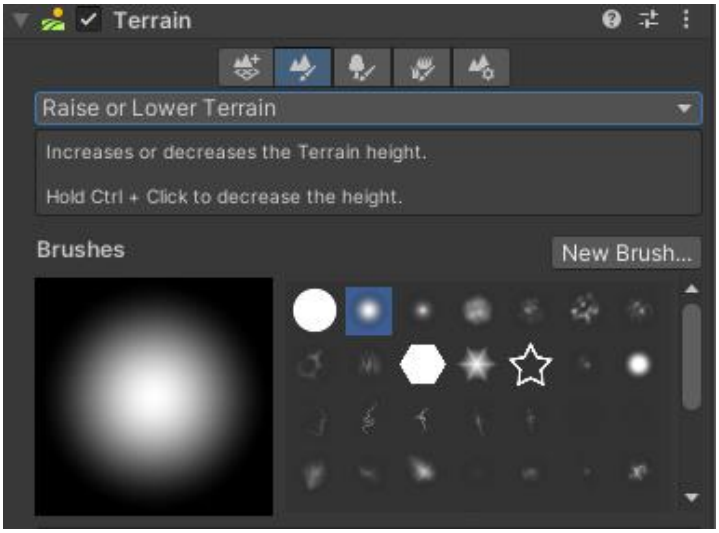


Εικόνα 3.5 Μενού δημιουργίας αντικειμένου εδάφους



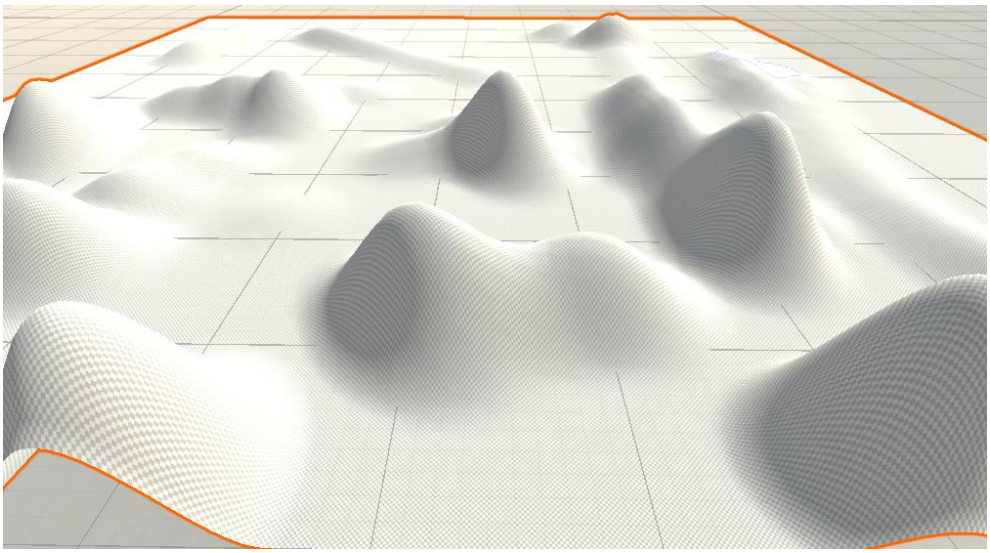
Εικόνα 3.6 Δημιουργία ενός κενού αντικειμένου εδάφους

Για να κάνουμε αλλαγές στο έδαφος χρησιμοποιούμε τα εργαλεία επεξεργασίας που μας παρέχει το Unity όπως το εργαλείο βαφής εδάφους (paint terrain) για την αλλαγή μορφολογίας εδάφους και προσθήκη λόφων ώστε να είναι πιο ρεαλιστικό και ευπαρουσίαστο.

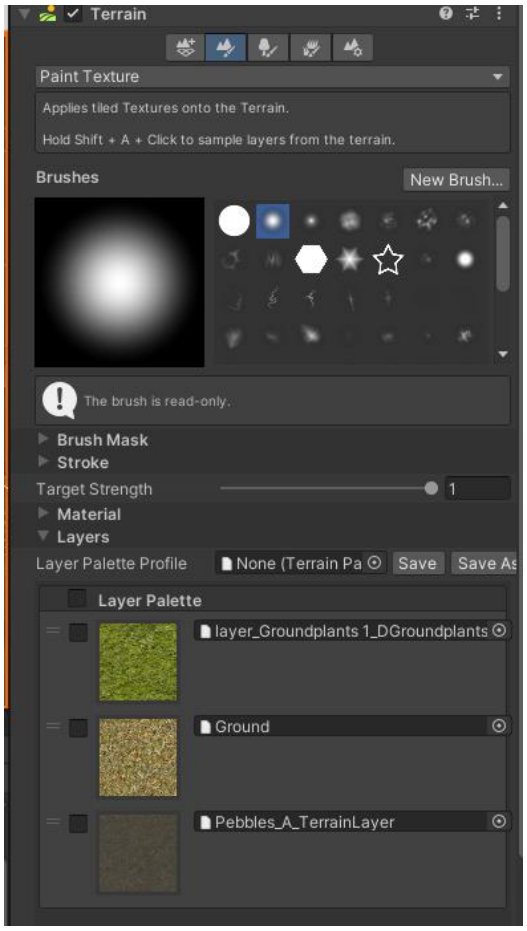


Εικόνα 3.7 Εργαλείο βαφής εδάφους με επιλεγμένη την ανύψωση του

Κάνοντας κλικ με το αριστερό κουμπί του ποντικιού δημιουργούμε ανυψώσεις στο έδαφος μας αλλάζοντας έτσι την μορφολογία του εδάφους. Αντίστοιχα πατώντας το ίδιο πάλι πλήκτρο με ταυτόχρονο πάτημα του πλήκτρου Control μπορούμε να κάνουμε μείωση του εδάφους για δημιουργία λιμνών και λοιπών αντικειμένων.

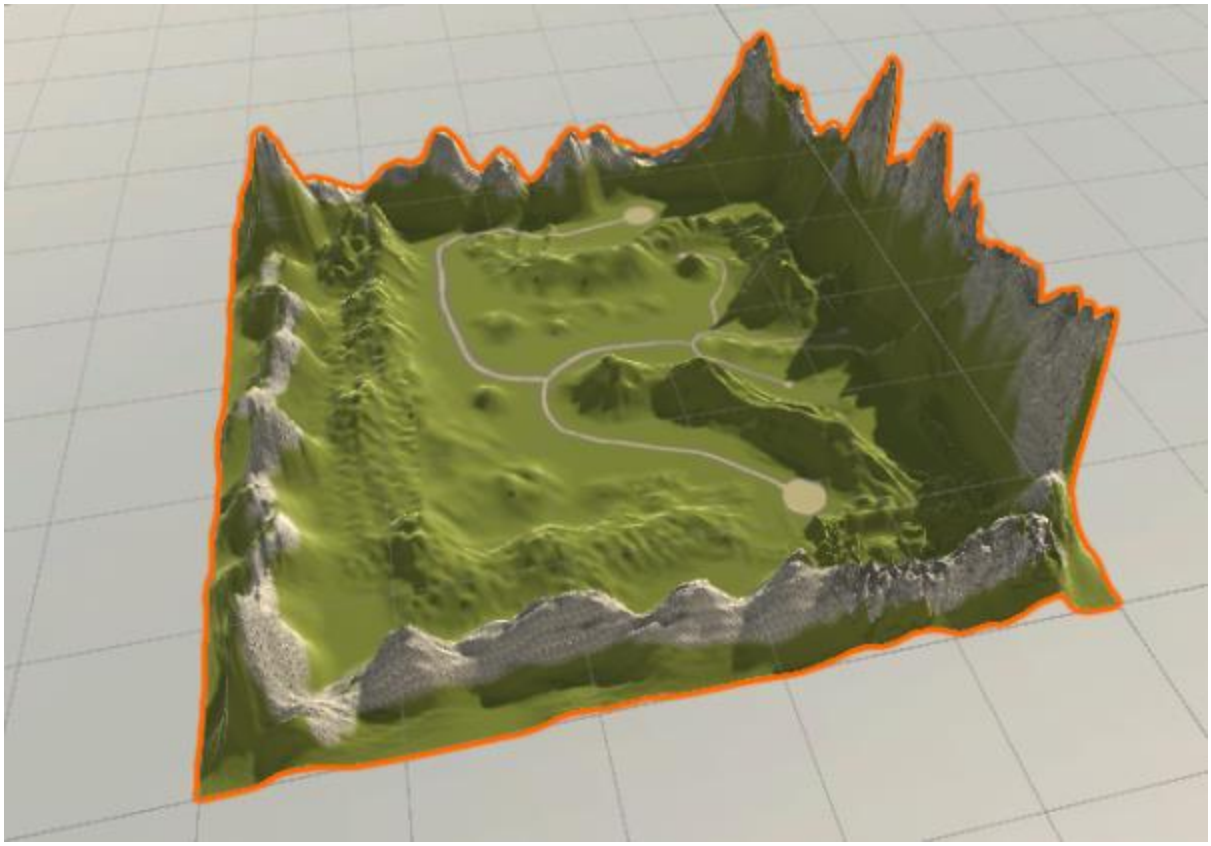


Εικόνα 3.8 Παράδειγμα ανύψωσης εδάφους με τη χρήση του εργαλείου



Εικόνα 3.9 Μενού εργαλείου βαφής υφής εδάφους

Με την δημιουργία και επεξεργασία του αντικειμένου εδάφους έτοιμη βλέπουμε ότι είναι πολύ απλό και μονόχρωμο, σκέτο. Αυτό θα το αλλάξουμε με την χρήση του εργαλείου βαφής του εδάφους και αλλαγής του υλικού του εδάφους σε κάτι πιο ρεαλιστικό, όπως το πράσινο του γρασιδιού με σχεδίαση μονοπατιών σε χρώματα οπού κάνουν αντίθεση με αυτό, όπως το καφέ του χώματος. Έτσι ο παίκτης θα αντιλαμβάνεται ποιο δρόμο πρέπει να ακολουθήσει για να υλοποιήσει την αποστολή του. Επιπρόσθετα ανυψωθήκαν τα άκρα της σκηνής μας έτσι ώστε να μην επιτρέπεται στον παίκτη να βγει εκτός των ορίων του χώρου παιχνιδιού, δίνοντας έτσι την εικόνα ύπαρξης μιας κορυφογραμμής γύρο από τον χώρο αυτό.

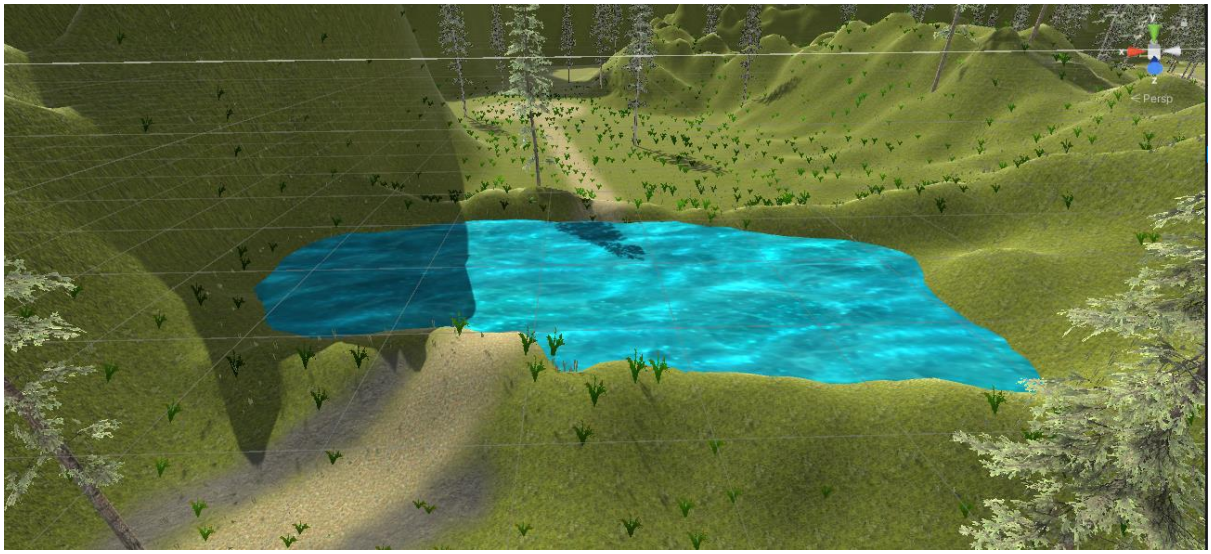


Εικόνα 3.10 Τελική εμφάνιση εδάφους μετά από τις σχεδιαστικές αλλαγές

Θεωρητικά η σχεδίαση του παραπάνω περιβάλλοντος είναι αρκετή για την ομαλή λειτουργία του παιχνιδιού μας αλλά για να το κάνουμε πιο ενδιαφέρον και εύμορφο θα προσθέσουμε, με την βοήθεια του Conifer Pack από το Asset Store του Unity, μερικά παραπάνω στοιχεία περιβάλλοντος όπως γρασίδι, φυτά και δέντρα. Έτσι το περιβάλλον μας μετατρέπεται ως εξής.

Εικόνα 3.11 & 3.12 Μοντέλο δέντρου από το πακέτο κωνοφόρων που χρησιμοποιήσαμε και εμφάνιση του στο περιβάλλον του παιχνιδιού μας

Εικόνα 3.13 Το περιβάλλον μέσα από τα «ματιά» του παίκτη (κάμερα παιχνιδιού)



Εικόνα 3.14 Περιβάλλον λίμνης, εμπόδιο δεύτερης αποστολής

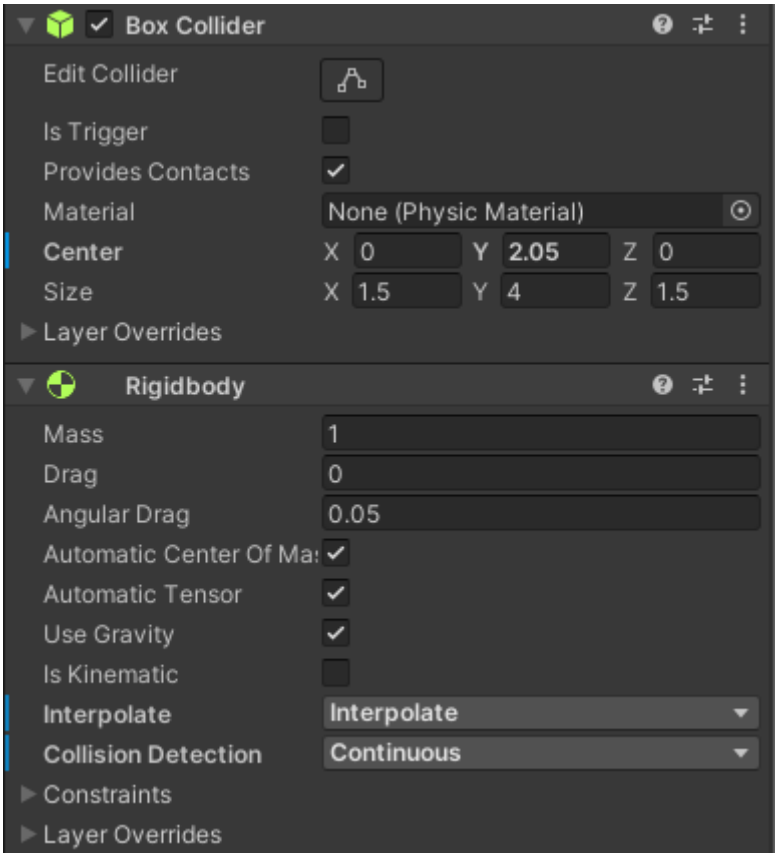
Εικόνα 3.15 Περιβάλλον Εργοστασίου, τελικού προορισμού τρίτης αποστολής

3.2.2 Δημιουργία κύριου χαρακτήρα παίκτη

Στο επόμενο βήμα αφού δημιουργηθεί το περιβάλλον θα πρέπει να δημιουργήσουμε τον κύριο χαρακτήρα που θα ελέγχει ο παίκτης μας μαζί με τις λειτουργίες που θα πρέπει να παρέχονται σε αυτόν, όπως περπάτημα, τρέξιμο και κίνηση κάμερας. Σαν βάση θα χρησιμοποιήσουμε ένα μοντέλο χαρακτήρα από το πακέτο "CharacterPack Lowpoly" του Unity store. Το μοντέλο του χαρακτήρα αποτελείται από το κομμάτι κάτω από την μέση, οπού συμπεριλαμβάνονται ποδιά και γοφοί και από το άνω κομμάτι που περιλαμβάνονται όλα τα υπόλοιπα. Αυτό μας βοηθάει στην ευκολότερη υλοποίηση της κίνησης του μοντέλου κατά το περπάτημα.


Εικόνα 3.16 & 3.17 Αρχικό μοντέλο χαρακτήρα, και μετατροπή σε μοντέλο παίκτη αφαιρώντας τις τρίχες προσώπου

Επιπλέον προσθέτουμε στο σύνολο του μοντέλου τα απαραίτητα εξαρτήματα όπως Collider για να μην μπορεί ο παίκτης να διαπεράσει αντικείμενα αλλά και εξάρτημα Rigidbody όπου συμβάλλει στην αντίδραση του χαρακτήρα μας με την βαρύτητα, δίνοντας έτσι μια πιο ρεαλιστική αντίδραση στις εντολές του παίκτη.

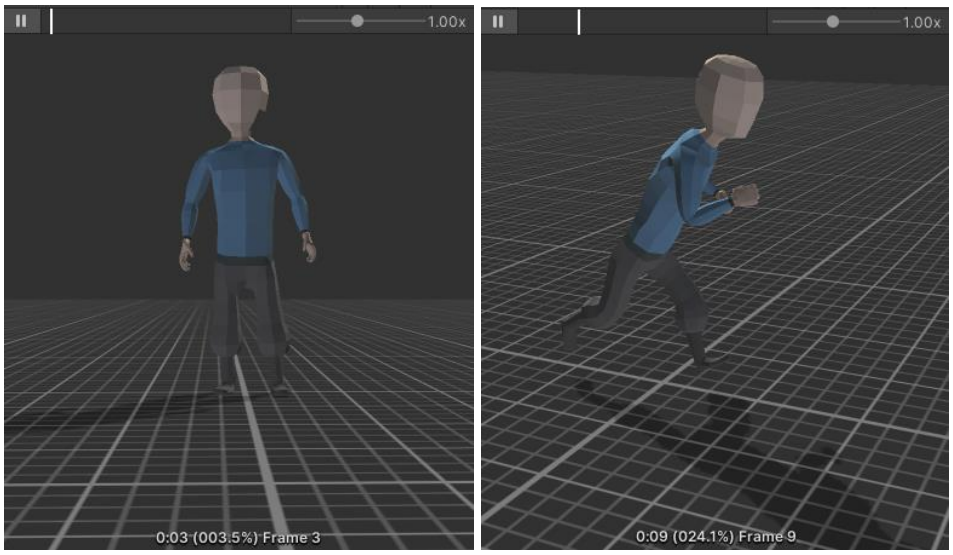


Εικόνα 3.18 Τα εξαρτήματα "Collider" και "Rigidbody"

Ύστερα θα προσθέσουμε την κύρια κάμερα μέσα στο μοντέλο του παίκτη για να έχουμε ένα ρεαλιστικό τρόπο ακολούθησης του παίκτη από την κάμερα. Αυτό θα προκαλούσε προβλήματα από μόνο του έτσι αφαιρούμε την απεικόνιση του μοντέλου από την κυριά κάμερα ώστε να μην βλέπει μέσα στο μοντέλο του παίκτη αλλά στο περιβάλλον. Σαν τελευταίο βήμα πρέπει με την ανάπτυξη ενός C# script με όνομα PlayerMovement να δώσουμε δυνατότητα κίνησης στο μοντέλο του παίκτη αλλά και δυνατότητα περιστροφής της κάμερας μαζί με αυτό. Έτσι δημιουργώντας μεταβλητές για έλεγχο πατήματος στο έδαφος και ταχύτητας αλλά και παρακολούθηση περιστροφής του ποντικιού ο παίκτης θα μπορεί να κινείται ανεξάρτητα με τα πλήκτρα του πληκτρολογίου και το ποντίκι για περιστροφή της κάμερας. Τέλος δημιουργούμε μια περαιτέρω κίνηση "sprint" για να μπορεί ο παίκτης να κινείται με μεγαλύτερη ταχύτητα μέσα στο περιβάλλον του παιχνιδιού με τη χρήση μιας μεταβλητής που διπλασιάζει ουσιαστικά την ταχύτητα του παίκτη κατά την κίνηση.

Με το πέρας υλοποίησης του λειτουργικού κομματιού θα προσθέσουμε την κίνηση του μοντέλου κατά το περπάτημα και τρέξιμο αντίστοιχα. Για να το κάνουμε αυτό δημιουργούμε έναν "Animator" που θα ελέγχει την κίνηση του μοντέλου μας και με μια μεταβλητή ταχύτητας "speed" μεταβάλλουμε την ταχύτητα αντίστοιχα με τη είσοδο που δίνει ο παίκτης, δηλαδή άμα δεν πατάει κανένα κουμπί κίνησης το μοντέλο του εντός του παιχνιδιού μένει στάσιμο σε ένα "idle animation" αλλά κατά την κίνηση με το πάτημα των αντίστοιχων κουμπιών η μεταβλητή αλλάζει τιμή ενεργοποιώντας έτσι την μετακίνηση των ποδιών και των χεριών του μοντέλου όπως ακριβός γίνεται και στην πραγματικότητα.

Περαιτέρω, με το πάτημα του κουμπιού “Shift” σε συνδυασμό με τα παραπάνω ο παίκτης τρέχει και έτσι η κίνηση των ακρών του γίνεται γρηγορότερη.



Εικόνα 3.19 & 3.20 Μοντέλο χαρακτήρα σε ακινησία και σε κίνηση κατά το πάτημα των αντίστοιχων πλήκτρων

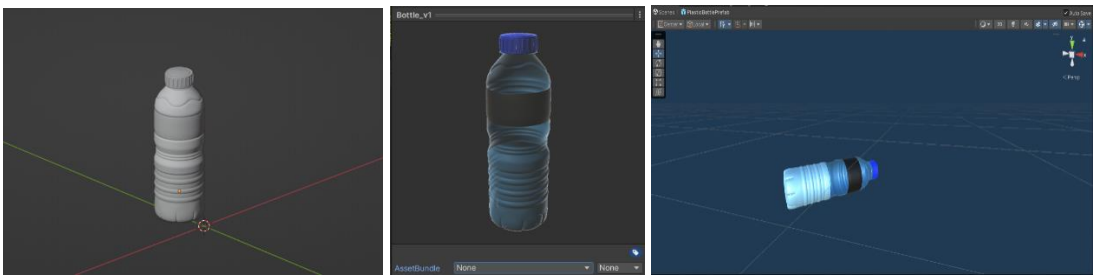
3.2.3 Εισαγωγή ανακυκλώσιμων αντικειμένων

Με την ανάπτυξη του κύριου χαρακτήρα παίκτη έχουμε την δυνατότητα εξερεύνησης του περιβάλλοντος από τον παίκτη, όμως για να υλοποιήσουμε τον στόχο μας, δηλαδή την εκμάθηση σημαντικών συμβουλών ανακύκλωσης στους παίκτες θα πρέπει να προσθέσουμε τα αντικείμενα με τα οποία θα αλληλεπιδρούν. Έτσι αποφασίσαμε να προσθέσουμε τρία ήδη ανακυκλώσιμων υλικών στο παιχνίδι, τόσα όσα και οι αποστολές του παίκτη:

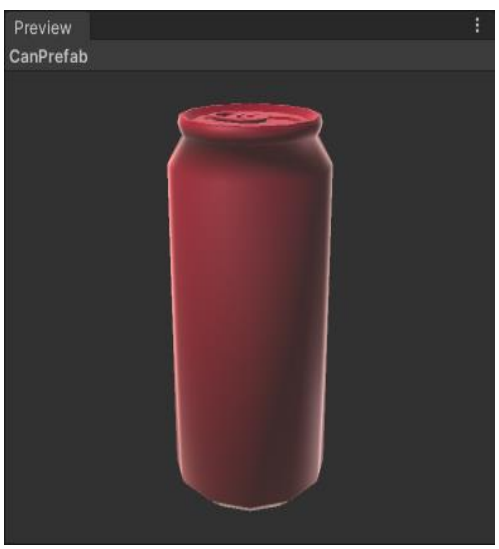
- Πλαστικό, με την μορφή των πλαστικών μπουκαλιών νερού μιας και είναι το πιο συχνό αντικείμενο όπου περιέχει πλαστικό στην ζωή μας.

- Αλουμίνιο, με την μορφή των κουτιών αλουμίνιου που συναντάμε στα αναψυκτικά

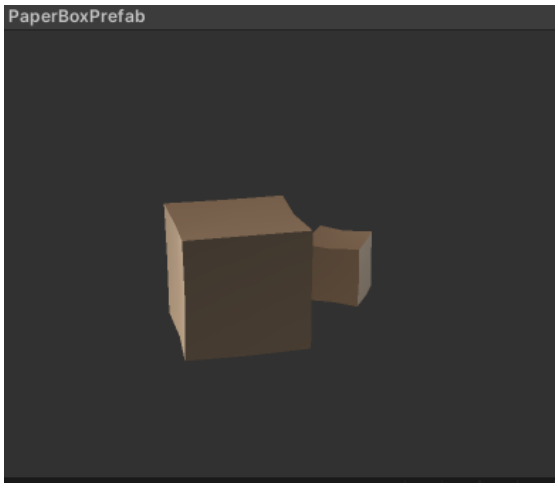
- Χαρτί, με την μορφή χαρτονιού σε κουτές αποθήκευσης.



Εικόνες 3.21 – 3.23 Μοντέλο πλαστικού μπουκαλιού σε διαφορά στάδια μορφοποίησης

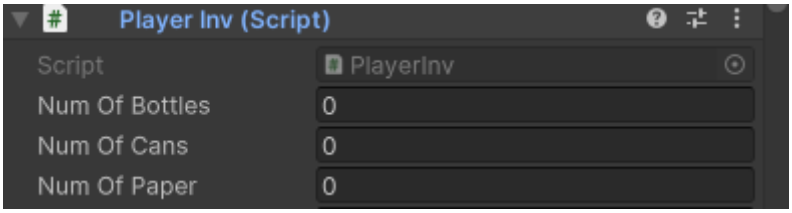


Εικόνες 3.24, 3.25 Μοντέλο κουτιού αλουμίνιου και εισαγωγή περιβάλλον του παιχνιδιού



Εικόνες 3.26, 3.27 Μοντέλο χαρτοκιβώτιου και εισαγωγή στο περιβάλλον του παιχνιδιού

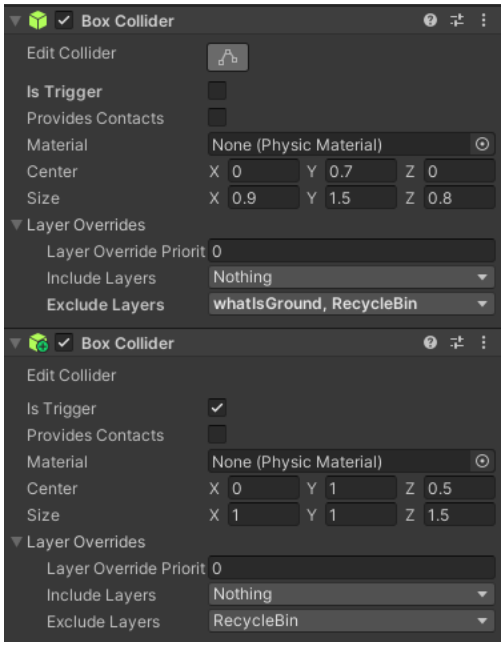
Όλα τα παραπάνω αντικείμενα εισάχθηκαν στο παιχνίδι με δωρεάν μοντέλα που βρέθηκαν στο internet. Αφού έγινε προσθήκη των παραπάνω αντικειμένων θα πρέπει να τοποθετηθούν σε διαφορά σημεία του χάρτη παιχνιδιού με σκοπό την συλλέξει τους από τον παίκτη, κάτι που γίνεται δυνατό με την ανάπτυξη μερικών ακόμα σεναρίων (script) σε C#, με όνομα Plastic Bottles, Alu Can και Paper Box αντίστοιχα για τα αντικείμενα και το Player Inv script για τον παίκτη, όπου συνδυαστικά θα επιτρέπουν στον παίκτη να τα συλλέξει απλά περπατώντας κοντά τους. Αυτό καθίσταται δυνατό με την χρήση των Colliders που έχει κάθε αντικείμενο καθώς και αυτόν που προσθέσαμε στο μοντέλο χαρακτήρα του παίκτη, με την χρήση της ρουτίνας OnTriggerEnter() όταν ο παίκτης βρίσκεται μέσα στον Collider ενός ανακυκλώσιμου αντικειμένου το συλλεγεί πρακτικά διαγράφοντας το αντικείμενο που βρίσκεται στο έδαφος και προσθέτοντας το σε έναν μετρητή συλλογής. Ο τελευταίος ενημερώνεται μετά από κάθε αλληλεπίδραση κρατώντας το σύνολο κάθε αντικειμένου στον κατάλογο αντικειμένων του παίκτη (Player Inventory) όπου θα μπορεί να βλέπει ο παίκτης στη διεπαφή χρήστη (UI) που θα αναπτύξουμε στην συνέχεια.



Εικόνα 3.28 Player Inv script

Εκτός από τα παραπάνω θα προσθέσουμε μερικά μοντέλα κάδων που θα μετατρέψουμε σε κάδους ανακύκλωσης με διαφορετικό χρωματισμό ανάλογα το αντικείμενο που θα αντιπροσωπεύουν. Τα χρώματα που χρησιμοποιούνται πιο συχνά σε κάδους ανακύκλωσης όπου αντιστοιχούν στα υλικά που επιλέξαμε είναι: κίτρινο για το αλουμίνιο, μπλε για το χαρτί και πορτοκαλί για το πλαστικό. Έτσι εισάγουμε και χρωματίζουμε αντίστοιχα τους κάδους ανακύκλωσης που θα χρησιμοποιεί ο παίκτης στο παιχνίδι.

Εικόνες 3.29 – 3.31 Χρωματισμένοι κάδοι ανακύκλωσης παιχνιδιού

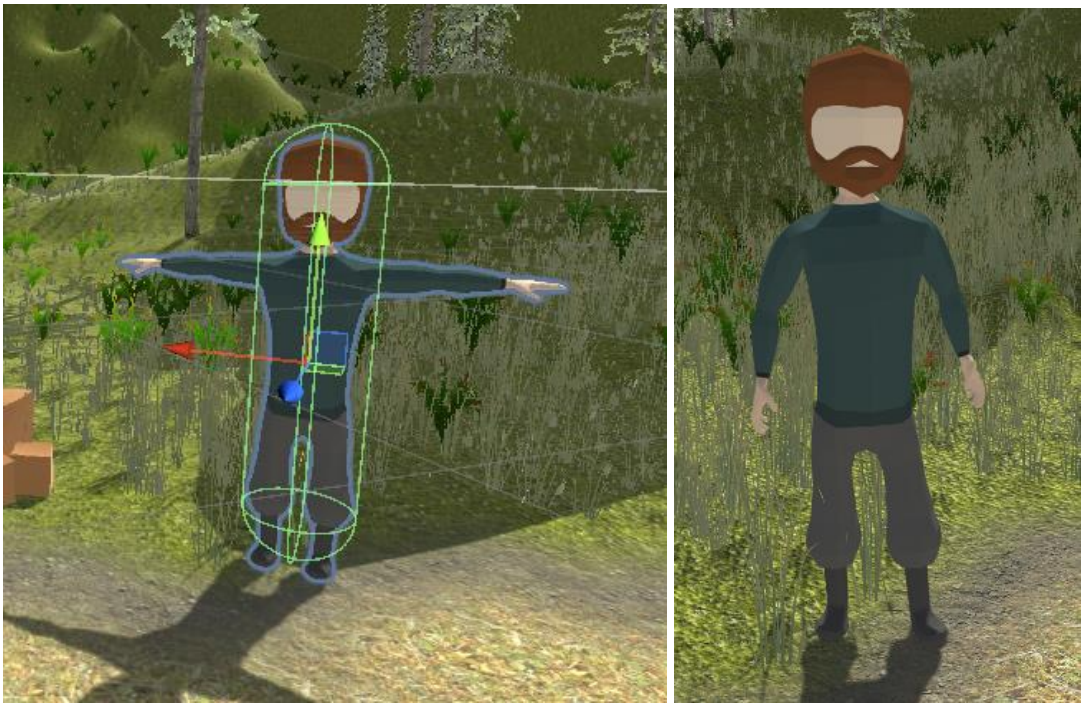


Τέλος για να μπορεί ο παίκτης να αλληλεπιδράσει με τους κάδους αυτούς αλλά και να πετάει τα απορρίμματα αναπτύσσουμε τα αντίστοιχα scripts αλληλεπίδρασης με όνομα Plastic Bin Deposit, Paper Bin Deposit και Alu Bin Deposit, πάλι με την χρήση των Colliders των αντικειμένων και του παίκτη, έτσι ώστε όταν είναι σε κοντά τους να μπορεί με την χρήση του κουμπιού "E" του πληκτρολογίου να ανακυκλώσει τα απορρίμματα που έχει στην κατοχή του όπου θα αφαιρούνται από τον μετρητή που είχαμε υλοποιήσει παραπάνω.

Εικόνα 3.32 Colliders κρούσης και αλληλεπίδρασης κάδων ανακύκλωσης

3.2.4 Υλοποίηση NPC χαρακτήρα και διεπαφής χρήστη

Για την υποβοήθηση του παίκτη στις αποστολές του αλλά και την ανάθεση αυτών στον παίκτη θα δημιουργήσουμε στη συνέχεια έναν μη ελεγχόμενο χαρακτήρα (NPC) ο οποίος θα επεξηγεί στον παίκτη τον σκοπό της επομένης αποστολής του αλλά και την κοινοποίηση συμβουλών και γνώσεων ανακύκλωσης. Αρχικά επιλέγουμε για μοντέλο του NPC το μοντέλο χαρακτήρα που χρησιμοποιήσαμε προηγουμένως και αυτή την φορά δεν αφαιρούμε τις τρίχες προσώπου ώστε να τον διαφοροποιήσουμε από τον κύριο παίκτη μας. Στην συνέχεια αναπτύσσουμε ένα ακόμα σενάριο (script) αλληλεπίδρασης, έτσι ώστε να μπορεί ο παίκτης να επικοινωνεί με τον χαρακτήρα αυτόν μέσω κειμένου που θα εμφανίζεται στην οθόνη και συγκεκριμένα στη διεπαφή του χρήστη (UI). Επιπλέον θα προσθέσουμε έναν Collider σε αυτόν για να έχει σύσταση το μοντέλο του χαρακτήρα ώστε να υπάρχει κρούση με το αντίστοιχο μοντέλο του παίκτη και να μην τον διαπερνά. Επιπρόσθετα χρησιμοποιούμε τον Animator που έχουμε χρησιμοποιήσει παραπάνω στο μοντέλο του παίκτη μας για να δώσουμε ζωή στον χαρακτήρα μέσω χρήσης ενός "idle animation" όπως ακριβώς έχει και ο κύριος χαρακτήρας μας.



Εικόνες 3.33 & 3.34 Μοντέλο χαρακτήρα με την προσθήκη του Collider και του Idle Animation αντίστοιχα

Αφού προσθέσαμε τον NPC χαρακτήρα μας στο παιχνίδι είμαστε σε θέση να αρχίσουμε την υλοποίηση της διεπαφής που θα βλέπει ο χρήστης στην οθόνη του, αυτή θα περιλαμβάνει μετρητές με του συνόλου συλλεγμένων απορριμμάτων ανά τύπο (πλαστικό, αλουμίνιο και χαρτί) οι οποίοι θα μηδενίζονται κατά την ανακύκλωση των αντικειμένων, το πλαίσιο διαλόγου με το οποίο θα αλληλεπιδρά με τον NPC χαρακτήρα, τον τρέχων στόχο του παίκτη για την αποστολή που πρέπει να φέρει εις πέρας αλλά και ένα μετρητή πόντων (σκορ) οπού θα συλλέγει με κάθε κατάθεση απορρίμματος στους κάδους ανακύκλωσης.

Αρχικά θα προσθέσουμε τρία στοιχεία εικόνας για τα αντίστοιχα αντικείμενα που θα συλλέγει ο παίκτης, δηλαδή μια εικόνα για πλαστικά μπουκάλια, μια για αλουμινένια κουτάκια και μια για χαρτόκουτα. Διπλά από τα στοιχεία αυτά δημιουργούμε τρία πλαίσια κειμένου οπού θα λαμβάνουμε με αναφορά τον αριθμό αντικειμένων που υπάρχει στους μετρητές απορριμμάτων που δημιουργήσαμε παραπάνω στο script του παίκτη μας. Έτσι με τη βοήθεια ακόμα ενός script, του Player Inv , μπορούμε να λάβουμε την τιμή των μεταβλητών από τον μετρητή και να τις προβάλουμε στην οθόνη του παίκτη, επιπρόσθετα στο script αυτό θα προσθέσουμε και μετρητές για τα απορρίμματα που έχει ανακυκλώσει στους κάδους ο παίκτης, έτσι ώστε να μπορούμε να εντοπίσουμε ποτέ έχει ολοκληρώσει την κάθε αποστολή μαζί και με έναν μετρητή σκορ στον οποίο θα προστίθενται πόντοι κάθε φορά που ο παίκτης ανακυκλώνει.

Εικόνα 3.35 Διεπαφή χρήστη

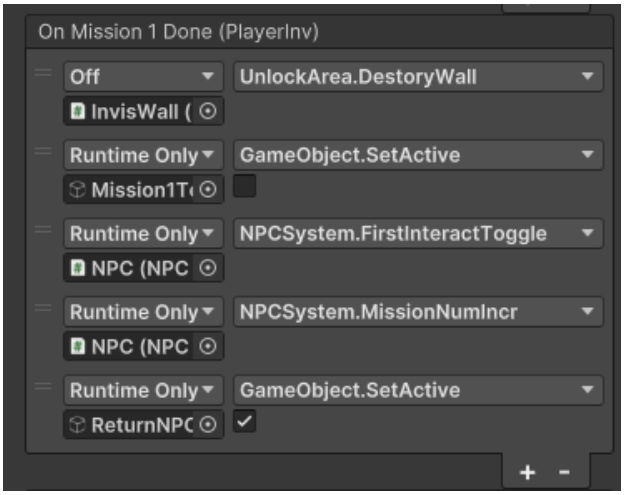
Επιπρόσθετα θα προσθέσουμε μερικά ακόμα πλαίσια κειμένου για την υποβοήθηση του παίκτη και της αλληλεπίδρασης αυτού με τα αντικείμενα του παιχνιδιού, όπως για παράδειγμα υπόδειξη πατήματος του κουμπιού “E” για να ακούσει τον NPC όταν βρίσκεται κοντά του, κάτι που μπορεί να εντοπιστεί με τη χρήση ενός άυλου και αόρατου Collider και τη χρήση της ρουτίνας OnColliderEnter(), αλλά και την ανακύκλωση των απορριμμάτων του στον αντίστοιχο κάδο πάλι με το πάτημα του ιδίου κουμπιού ενώ βρίσκεται στην εμβέλεια των κάδων ανακλώσης.

Εικόνα 3.36 Πλαίσιο κειμένου συμβουλής αλληλεπίδρασης χαρακτήρα

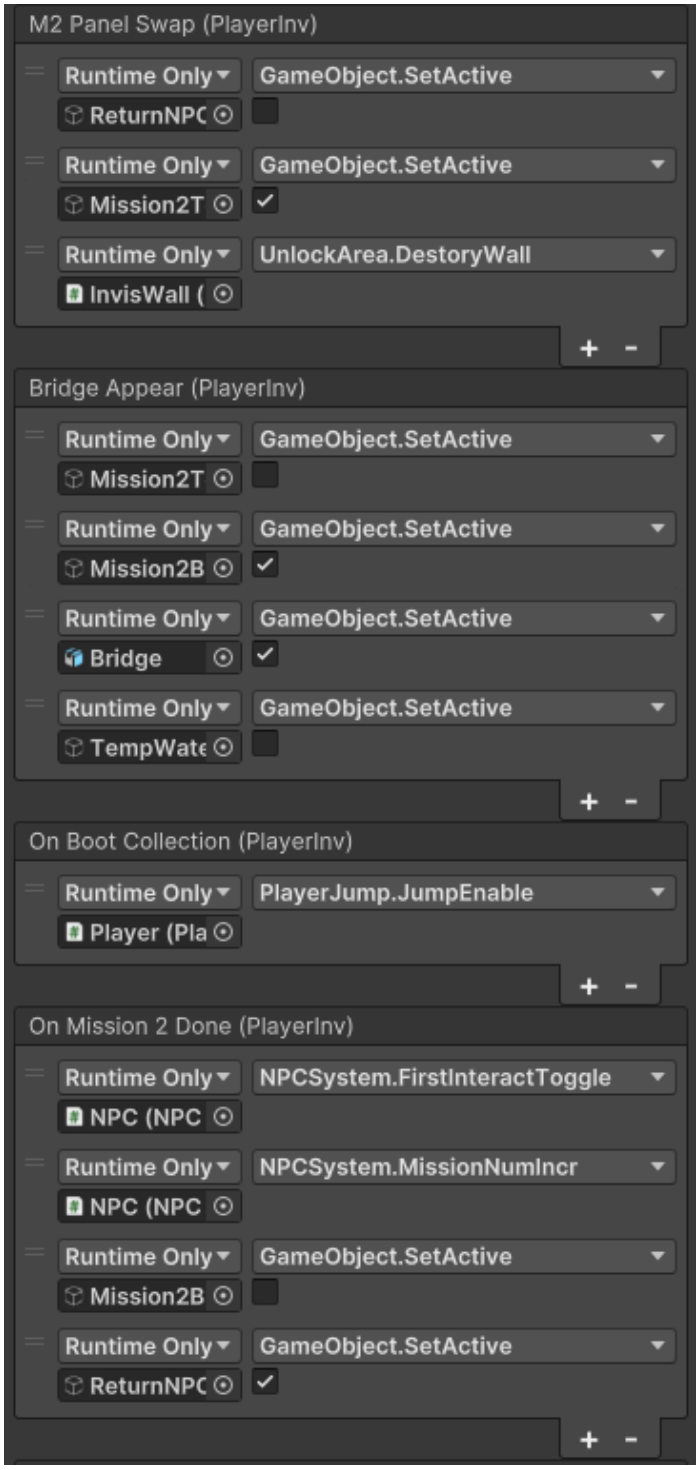
Εικόνα 3.37 Πλαίσιο κειμένου συμβουλής αλληλεπίδρασης κάδου ανακύκλωσης πλαστικού

3.2.5 Υλοποίηση αποστολών

Αρχικά τοποθετούμε τα αντικείμενα που θα συλλέγει ο παίκτης στον εικονικό χώρο μας με μικρή απόσταση μεταξύ τους ώστε να έχουν μια συνοχή συλλογής προς διευκόλυνση του παίκτη. Εφόσον έχουμε ήδη προσθέσει το script για την συλλογή τους όταν περνάει ο παίκτης από κοντά τους το μόνο που απομένει είναι η αύξηση των αντίστοιχων μετρητών και ο έλεγχος ολοκλήρωσης αποστολής, δηλαδή έλεγχος συλλογής του απαιτούμενου ποσού αντικειμένων. Αυτό γίνεται με την δημιουργία ενός συμβάντος της Unity (**Unity Event**) το οποίο αποτελεί ουσιαστικά μια καθολική if statement οπού μπορεί και πυροδοτεί υπό-ρουτίνες από άλλα σενάριά που έχουμε στον εικονικό μας χώρο. Επομένως τοποθετώντας ένα **Unity Event** με έλεγχο στον μετρητή πλαστικών μπουκαλιών του παίκτη μπορούμε κατά την συλλογή και ανακύκλωση 10 μπουκαλιών να πυροδοτούμε την υπό ρουτίνα αλλαγής αποστολής στην διεπαφή του χρήστη, σε αυτή την περίπτωση θα γίνει αλλαγή του κειμένου της αποστολής από συλλογή των μπουκαλιών σε επιστροφή στον NPC μας για την απόδοση της επομένης αποστολής αλλά και μερικών συμβουλών ανακύκλωσης.



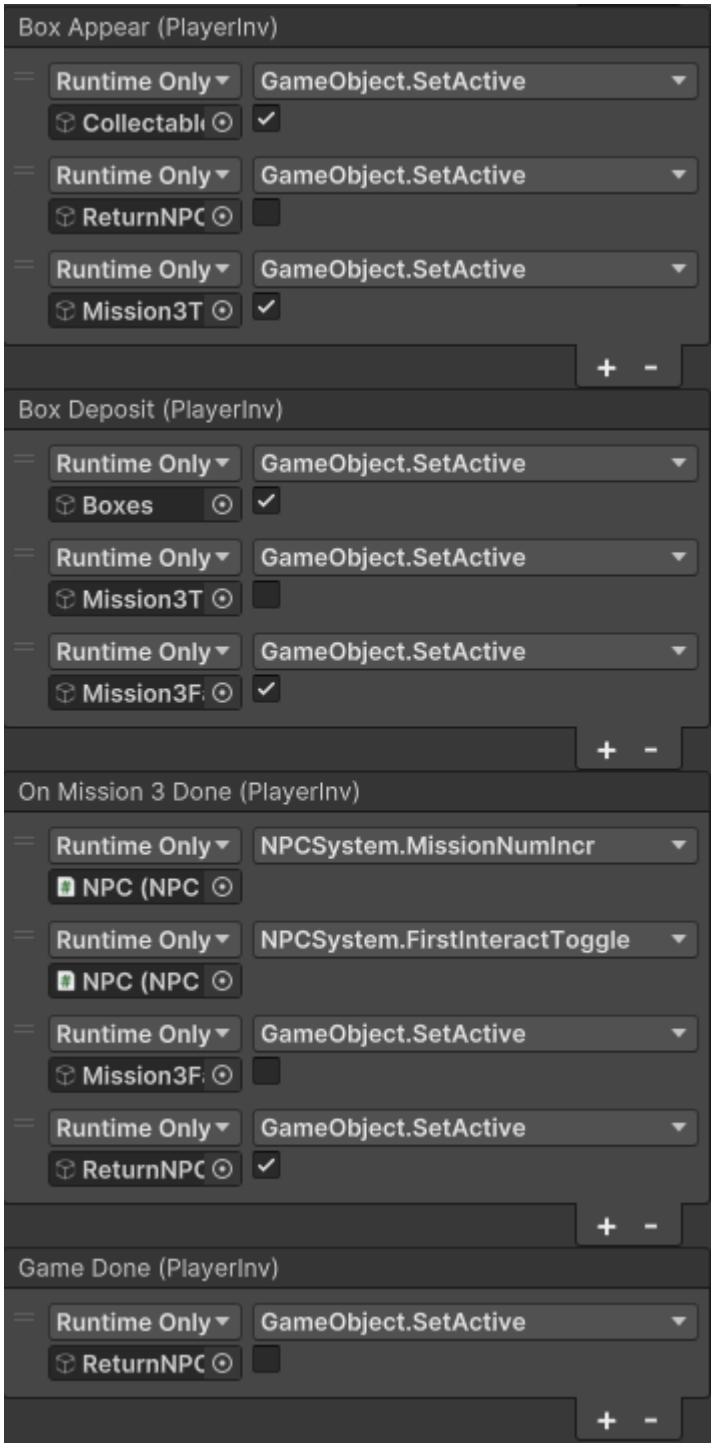
Εικόνα 3.38 Unity Event για την πρώτη αποστολή, εναλλαγή κειμένου αποστολής διεπαφής παίκτη



Έτσι κατά την ανάπτυξη της δεύτερης αποστολής κάνουμε χρήση των αλουμινένιων αντικειμένων του παιχνιδιού μας μαζί με τον μετρητή που τους έχουμε αποδώσει. Σκοπός του παίκτη είναι να περάσει την λίμνη που έχουμε δημιουργήσει σαν εμπόδιο για την απόκτηση της λειτουργίας άλματος, έτσι κατά την συλλογή και ανακύκλωση 12 κουτιών αλουμίνιου ένα ακόμα unity event πυροδοτεί την δημιουργία (εμφάνιση) μιας μεταλλικής γέφυρας πάνω από την λίμνη την οποία μπορεί τώρα να περάσει ο παίκτης και να αποκτήσει τις μπότες αναπήδησης. Κατά την απόκτηση του αντικειμένου αυτού, μέσω πάλι της λειτουργίας συλλογής με Colliders, ενεργοποιούμε άλλο ένα Unity Event για την ενεργοποίηση του άλματος στο script κίνησης του παίκτη και την εναλλαγή κειμένου αποστολής στην διεπαφή του χρήστη ώστε να γυρίσει πίσω στον NPC χαρακτήρα μας.

Εικόνα 3.39 Unity Events δεύτερης αποστολής, διαγραφή αοράτων τοίχων,

εμφάνιση γέφυρας, συλλογή αναβάθμισης άλματος και εναλλαγή κειμένου αποστολής διεπαφής παίκτη



Τέλος για την ανάπτυξη της τρίτης και τελευταίας αποστολής μας τοποθετούμε το τελευταίο είδος ανακυκλώσιμου αντικειμένου στον χώρο μας, τα χαρτόκουτα, με σκοπό την συλλογή τεσσάρων από αυτά προς επαναχρησιμοποίηση τους για να αποκτήσει πρόσβαση ο παίκτης σε έναν μοχλό απενεργοποίησης του εργοστασίου που έχουμε δημιουργήσει. Κατά την συλλογή των κουτιών, για άλλη μια φορά με την χρήση μετρητών και Collider, όταν ο μετρητής φτάσει τον αριθμό που θέλουμε μέσω ενός Unity Event πυροδοτούμε την εμφάνιση των κουτιών στο χώρο του εργοστασίου, δίνοντας έτσι την δυνατότητα στον παίκτη να τα χρησιμοποιήσει σαν πλατφόρμες αναπήδησης για να φτάσει τον μοχλό. Με το τράβηγμα του μοχλού έχουμε και πάλι χρήση ενός Unity Event για την εναλλαγή κειμένου στην διεπαφή του χρήστη έτσι ώστε να οδηγηθεί στον NPC μας για την απόδοση συγχαρητήριων και σύνοψη τελικών συμβουλών ανακύκλωσης.

Εικόνα 3.40 Unity Events τρίτης αποστολής, εμφάνιση χαρτόκουτων,

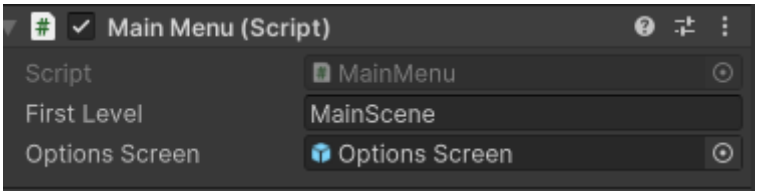
εμφάνιση πλατφόρμων αναπήδησης και εναλλαγή κειμένου αποστολής διεπαφής παίκτη.

3.2.6 Δημιουργία αρχικού μενού, μενού ρυθμίσεων παιχνιδιού και μενού παύσης

Με την ολοκλήρωση λειτουργικότητας του παιχνιδιού είμαστε σε θέση να υλοποιήσουμε τις τελικές πινελιές του παιχνιδιού μας, δηλαδή το αρχικό μενού που θα εμφανίζεται στον χρήστη κατά την εκκίνηση του παιχνιδιού, του μενού ρυθμίσεων για την ρύθμιση εικόνας και έντασης ήχου του παιχνιδιού και το μενού παύσης.

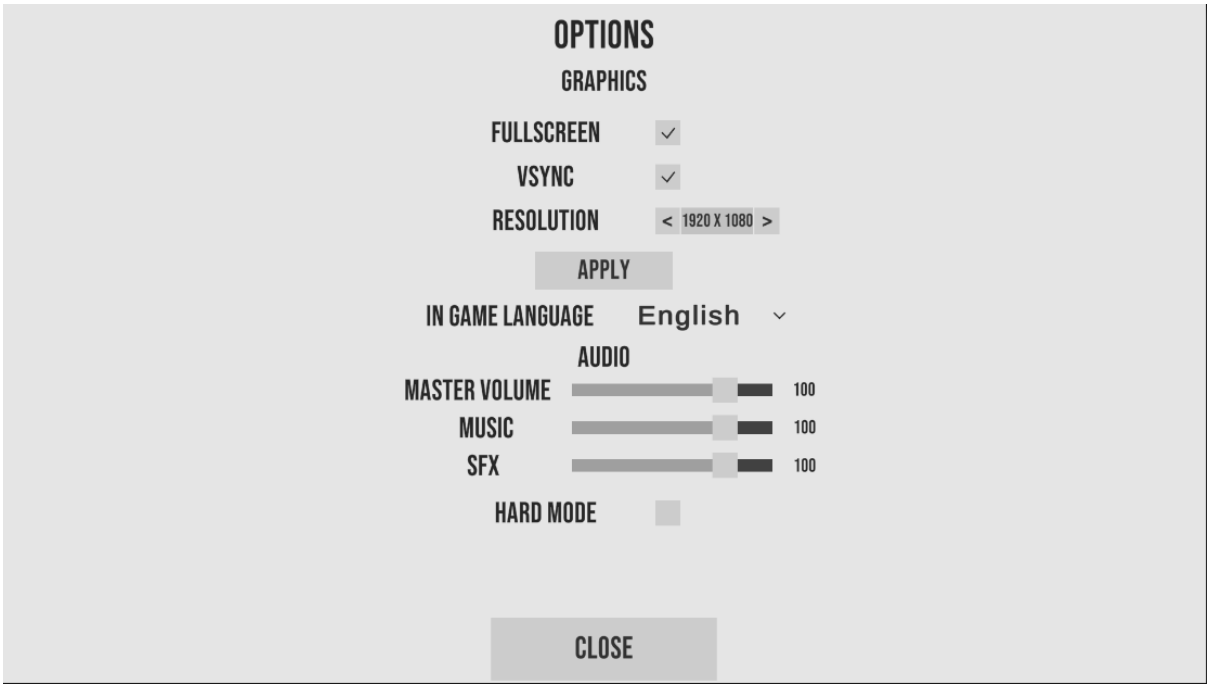
Για το αρχικό μενού δημιουργούμε μια νέα σκηνή παιχνιδιού για να έχουμε έναν κενό φόντο πίσω από το μενού μας. Στην νέα σκηνή αυτή προσθέτουμε έναν κενό καμβά διεπαφής (Canvas UI) και τοποθετούμε ένα πλαίσιο κειμένου σε αυτόν με τον τίτλο του παιχνιδιού μας μαζί με τα απαραίτητα κουμπιά έναρξης, ανοίγματος μενού ρυθμίσεων και εξόδου παιχνιδιού. Αφετέρου προσθέτουμε μια εικόνα μέσα από το παιχνίδι μας σαν φόντο για να μην είναι τόσο κενός ο καμβάς μας και υλοποιούμε την εναλλαγή σκηνών παιχνιδιού στο πάτημα του κουμπιού έναρξης μέσω του Main Menu script και ενός Unity Event. Αντίστοιχα υλοποιούμε την λειτουργικότητα του κουμπιού εξόδου μέσα από το ίδιο script αλλά και την προσθήκη λειτουργικότητας του μενού ρυθμίσεων που θα αναπτύξουμε παρακάτω.

Εικόνα 3.41 Τελειοποιημένο αρχικό μενού παιχνιδιού



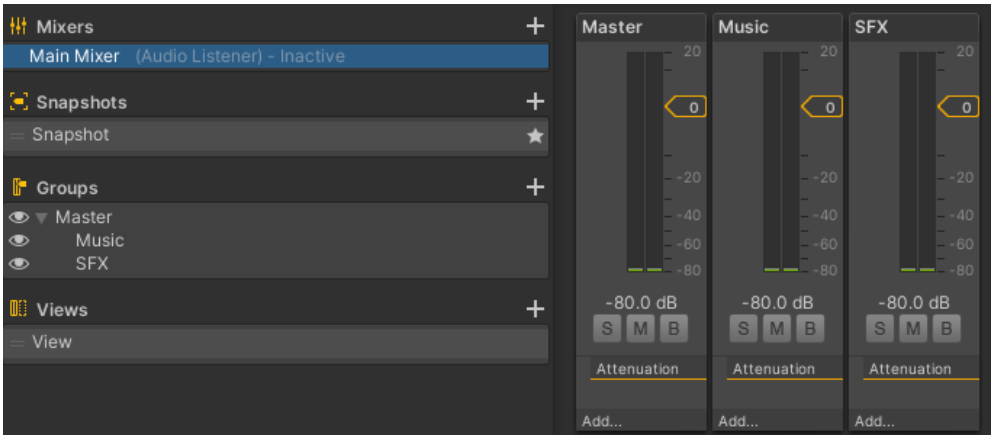
Εικόνα 3.42 Εργαλείο ρύθμισης Main Menu script

Για την υλοποίηση του μενού ρυθμίσεων δημιουργούμε επιπρόσθετο έναν καμβά που θα περιέχει τους ολισθηρές (sliders) έντασης ήχου και τις επιλογές εικόνας του παιχνιδιού, ανάλυση πλήρης οθόνη και vertical synchronization (Vsync), καθως και την επιλογή αύξησης δυσκολίας του παιχνιδιού όπου δεν αναφέρεται ο τύπος ανακυκλώσιμου που δέχεται κάθε κάδος κατά την προσέγγιση αυτού από τον παίκτη αλλά και την επιλογή γλώσσας διαλόγου με τον NPC χαρακτήρα του παιχνιδιού.

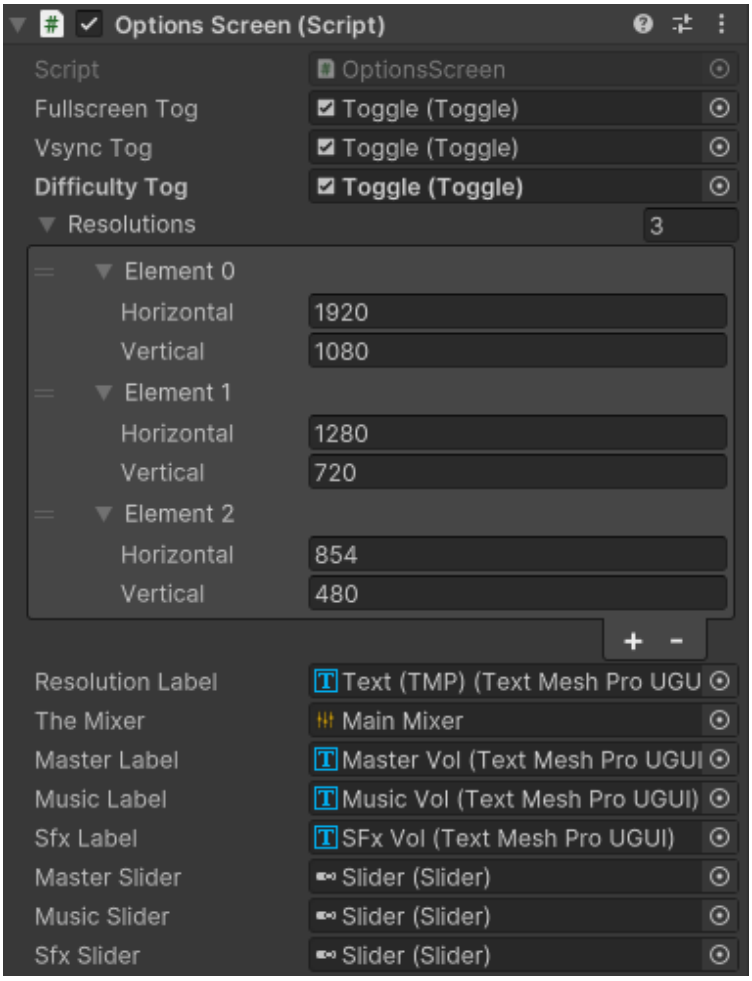


Εικόνα 3.43 Μενού ρυθμίσεων παιχνιδιού

Για την υλοποίηση της λειτουργικότητας του μενού θα δημιουργήσουμε ένα νέο script με όνομα Options Screen όπου θα περιέχει την σύνδεση των κουμπιών με τις κυρίες ρύθμισης της μηχανής Unity για την εναλλαγή ρυθμίσεων εικόνας. Επιπρόσθετα δημιουργούμε έναν διαχειριστή ήχων για το παιχνίδι μας και συνδέουμε όλα τα ηχητικά εφέ και εφέ μουσικής της κυρίας σκηνής μας σε αυτόν. Έτσι μπορούμε να τα διαχωρίσουμε ανά κατηγορία ήχου και να διαχειριστούμε την ένταση της κάθε κατηγορίας μέσω του μενού ρυθμίσεων.



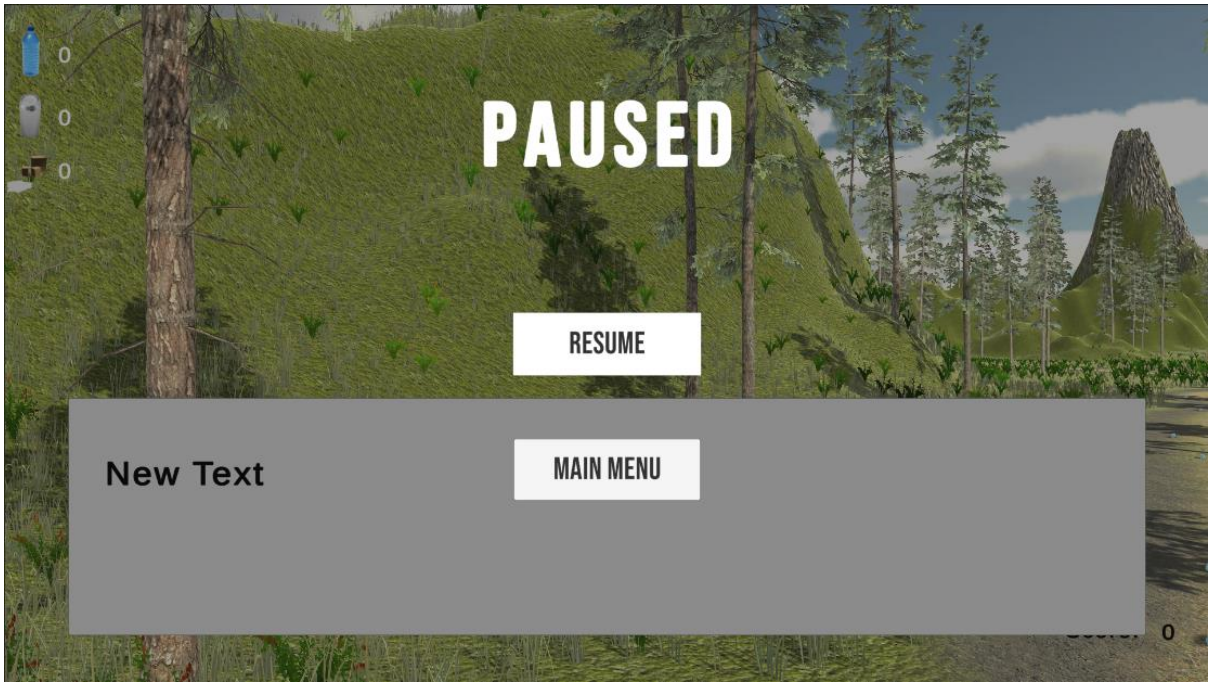
Εικόνα 3.44 Μεικτής διαχειριστή ήχου.



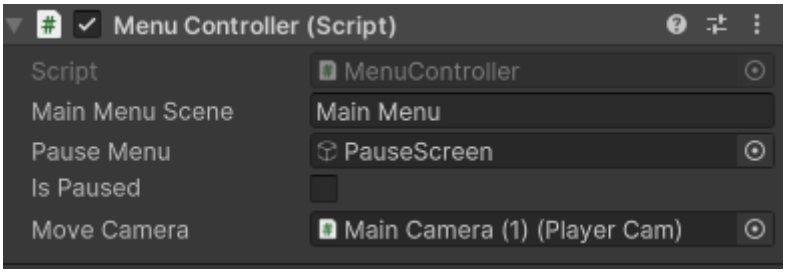
Εικόνα 3.45Διασύνδεση ρυθμίσεων ήχου και εικόνας παιχνιδιού με το μενού ρυθμίσεων μέσω του Options Screen script.

Τέλος προσθέτουμε το μενού παύσης παιχνιδιού όπου θα εμφανίζεται κάθε φορά που ο παίκτης θα πατάει το κουμπί Escape, το οποίο θα θέτει σε παύση το παιχνίδι και θα δίνει την δυνατότητα εξόδου παιχνιδιού ανά πασά στιγμή στον παίκτη. Για την υλοποίηση αυτού δημιουργούμε ένα νέο αντικείμενο εικόνας στον ήδη υπάρχον καμβά της κύριας σκηνής του παιχνιδιού μας το οποίο περιέχει τα καταλληλά κουμπιά για έξοδο ή συνέχιση του παιχνιδιού μαζί με τα απαραίτητα πλαίσια κειμένου. Η λειτουργικότητα αυτών υλοποιείται μέσω του

Menu Controller script και των ρουτίνων OnClick() που υπάρχουν σε κάθε κουμπί κατά την δημιουργία του. Επιπρόσθετα το script μας θα θέτει σε παύση το παιχνίδι και την κίνηση της κάμερας του παίκτη και θα εμφανίζει το ποντίκι του παίκτη στην οθόνη έτσι ώστε να μπορεί να πατήσει το κουμπί της αρεσκείας του.



Εικόνα 3.46 Μενού παύσης παιχνιδιού



Εικόνα 3.47 Εργαλείο επιλογών Menu Controller script

4. Εκτέλεση Βιντεοπαιχνιδιού

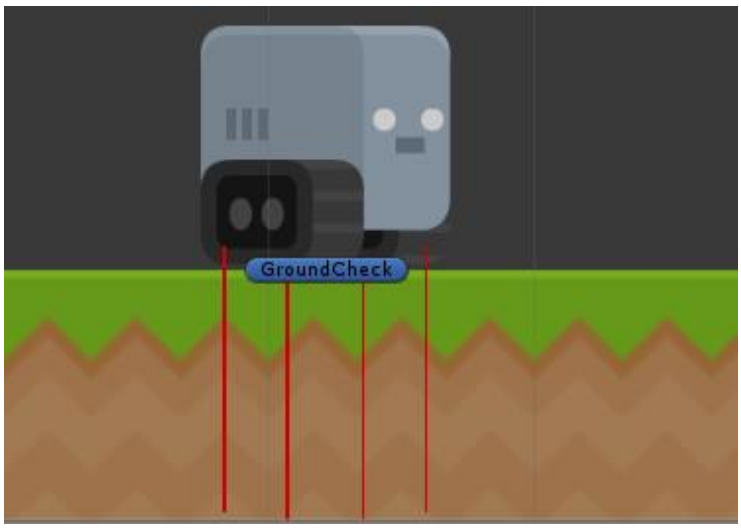
4.1 Ανάλυση αποστολών

Με την ολοκλήρωση ανάπτυξης του παιχνιδιού μπορούμε να δούμε πιο αναλυτικά τπροσφέρει από άποψη περιεχομένου κατά την εκτέλεση του. Το παιχνίδι προσφέρει τρεις αποστολές στον παίκτη οπού αφορούν τα τρία συχνότερα υλικά ανακύκλωσης καθώς και διαφορετικούς τρόπους αντιμετώπισης των απορριμμάτων αυτών.

4.2 Προβλήματα που συναντήθηκαν

Κατά την ανάπτυξη και εκτέλεση του παιχνιδιού υπήρξαν κάποια προβλήματα που θα αναφερθούν και θα αναλυθούν παρακάτω. Αρχικά το πρώτο πρόβλημα αφορά την δυνατότητα του παίκτη να βγει εκτός της εμβελείας του NPC Collider με αποτέλεσμα να μην μπορεί να διαβάσει το μήνυμα στο πλαίσιο κειμένου και την αδυναμία συνέχισης της αλληλεπίδρασης. Αυτό έχει εν μέρει διορθωθεί τοποθετώντας αόρατους τοίχους γύρω από τον παίκτη κατά την αλληλεπίδραση με τον NPC χαρακτήρα αλλά δεν αποτελεί και ολοκληρωτική λύση μιας και κάτι τέτοιο δεν βρέθηκε.

Ένα δεύτερο πρόβλημα που συναντήθηκε ήταν η δυνατότητα αναπήδησης παίκτη απεριόριστες φορές κατά την συλλογή της αναβάθμισης του. Ουσιαστικά με την ύπαρξη δυνατότητας αναπήδησης ο παίκτης μπορούσε να βγει εκτός ορίων του παιχνιδιού απλά πατώντας πολλές φορές το κουμπί αναπήδησης (Space), μια λειτουργία που δεν είναι επιθυμητή. Έτσι με την χρήση εκπομπής ακτίνας (raycast), δηλαδή μιας ευθείας από το μοντέλο του παίκτη και προβολής αυτής προς το έδαφος από το μοντέλο του παίκτη μπορεί να γίνει έλεγχος ύψους και πατήματος εδάφους μέσα από το script κίνησης. Αυτό επιτυγχάνεται μέσω του ελέγχου επικάλυψης της ακτίνας που προβάλλουμε από το κάτω μέρος του μοντέλου του παίκτη, ουσιαστικά ελέγχουμε πότε επικαλύπτεται αυτή από το έδαφος που έχουμε ορίσει και μόνο τότε επιτρέπουμε στον παίκτη να αναπηδήσει.



Εικόνα 4.1 Παράδειγμα χρήσης εκπομπής ακτίνας για έλεγχο επαφής εδάφους

Έτσι ο παίκτης μπορεί να αναπηδήσει μέχρι ένα συγκεκριμένο ύψος από το έδαφος διορθώνοντας το πρόβλημα μας.

Τέλος, υπάρχει ένα γενικό πρόβλημα που γινότανε όλο και πιο εμφανές κατά την ανάπτυξη του παιχνιδιού που αφορά την επιβράδυνση του παιχνιδιού σε υπολογιστές με χαμηλές επιδόσεις. Αυτό γίνεται λόγω της ύπαρξης μεγάλου αριθμού αντικειμένων στον χώρο όπως δέντρα, γρασίδι αλλά και απορριμμάτων, πόσο μάλλον των αντικειμένων του εργοστασίου και της γέφυρας αλλά και λόγω μεγέθους του χάρτη μας. Ως συνέπεια, αυτό περιόρισε τα αντικείμενα που μπορούν να προστεθούν στον χάρτη μας αλλά και το μέγεθος αυτού, περιορίζοντας έτσι και τις ιδέες περαιτέρω ανάπτυξης του παιχνιδιού μας όπως επιπλέον εφέ, πιο πολλά αντικείμενα πάρκων (παγκάκια, παιδικές χαρές κλπ.) αλλά και λειτουργιών παιχνιδιού.

5. Συμπεράσματα και Πιθανές Βελτιώσεις

5.1 Προσωπική αξιολόγηση

Με το πέρας της ανάλυσης του παιχνιδιού που δημιουργήθηκε και έχοντας παρατηρήσει τα προβλήματα τα οποία υπάρχουν σε αυτό, μπορούμε να το ερευνήσουμε με μια πιο αντικειμενική ματιά όσον αφορά την ικανοποίηση του στόχου μας. Αρχικά από το θέμα εκπαιδευτικότητας και διδασκαλίας το παιχνίδι ικανοποιεί το στόχο που δώσαμε μαθαίνοντας στον παίκτη την ανάγκη της ανακύκλωσης και τις διάφορες χρήσεις της στην ζωή μας μέσω ενημερωτικών μηνυμάτων που εμφανίζονται κατά την διάρκεια του παιχνιδιού. Μέσω των μηνυμάτων αυτών ο παίκτης μαθαίνει πράγματα τα οποία θα μπορούσαν να ειπωθούν σε μια τάξη με τον παραδοσιακό τρόπο διδασκαλίας αλλά μέσω της ψυχαγωγίας η αποστήθιση γίνεται πιο εύκολη.

Επακολούθως, όσον αφορά την κατάταξη του παιχνιδιού μας στο σύνολο των σοβαρών παιχνιδιών το παιχνίδι αυτό παρέχει μια ισορροπία ψυχαγωγίας και εκμάθησης μιας και κατά την ολοκλήρωση κάθε αποστολής μετρά το πέρας της ψυχαγωγίας του παίκτη μέσω της συλλογής των αντικειμένων, δίνονται επιβραβεύσεις με την μορφή εκπαιδευτικών συμβουλών ανακύκλωσης. Με την απόκτηση της αναβάθμισης του ο παίκτης διατηρεί το ενδιαφέρον του για το παιχνίδι έχοντας την δυνατότητα να κινηθεί διαφορετικά απ’ ότι προηγουμένως και έτσι οι εκπαιδευτικές συμβουλές μεταδίδονται πιο αποτελεσματικά.

Τέλος όσον αφορά τον χειρισμό του παιχνιδιού, βρίσκεται σε ικανοποιητικό επίπεδο παρ’ όλα τα προβλήματα που συναντήθηκαν κατά την ανάπτυξη της δυνατότητας κίνησης του παίκτη στο χώρο. Τα πλήκτρα χειρισμού είναι τα πιο συνήθη σε παιχνίδια πρώτου προσώπου διευκολύνοντας την κατανόηση κίνησης από τον παίκτη. Τα μενού είναι σχεδιασμένα με απλό τρόπο έτσι ώστε να αποφεύγεται κάθε δυσκολία κατανόησης τους, αλλά και η διεπαφή του παίκτη που εμφανίζεται στην οθόνη περιέχει αρκετές πληροφορίες έτσι ώστε να είναι εύκολη η κατανόηση των αποστολών, η κατάσταση στην οποία βρίσκονται και η πρόοδος του παίκτη προς τον στόχο της κάθε μίας. Σημαντικό είναι να αναφερθεί βέβαια πως παρότι η λειτουργία του παιχνιδιού είναι ικανοποιητική παραμένει σε πρωτότυπο στάδιο και κατά την ανάδειξη των δυνατοτήτων του υπάρχει πάντα η πιθανότητα εμφάνισης σφαλμάτων, όπως αναφέρθηκαν στην παραπάνω ενότητα.

5.2 Αξιολόγηση από τρίτους

- **Αγόρι (Ετών 11)**

Το αγόρι εξέφρασε γενική ικανοποίηση από το παιχνίδι με μερικές συμβουλές πιθανής βελτίωσης. Το αρχικό μενού αρχικά ήταν κατανοητό και εύκολο στην χρήση παρότι ήταν στην αγγλική γλώσσα. Στο κυρίως μέρος του παιχνιδιού βρήκε εύκολη την πλοήγηση αφού γίνεται χρήση κοινών κουμπιών διεπαφής με άλλα παιχνίδια. Εξέφρασε ικανοποίηση από τα γραφικά του παιχνιδιού αν και ανέφερε την μικρή ποικιλία συμπληρωματικών αντικειμένων όπως τα δέντρα και τα φυτά αλλά ότι δεν επηρέασε την εμπειρία του παιχνιδιού. Ενώ του άρεσε το αντικείμενο του παιχνιδιού και οι συμβουλές ανακύκλωσης που αναφέρονταν εξέφρασε παράπονο για την ποικιλία και τον αριθμό αυτών μιας και τις ολοκλήρωσε γρήγορα. Τέλος θεωρεί ότι το παιχνίδι αυτό είναι ένα καλό σημείο εκκίνησης εκμάθησης της σημασίας της ανακύκλωσης σε παιδιά, και πρόσθεσε ότι με μερικές αλλαγές και επεκτάσεις θα μπορούσε να χρησιμοποιηθεί για την εκμάθηση της περιβαλλοντικής αγωγής.

5.3 Πιθανές βελτιώσεις

Το τελικό αποτέλεσμα του παιχνιδιού είναι ικανοποιητικό, παρ’ όλα αυτά μπορούν να γίνουν μελλοντικά κάποιες βελτιώσεις στην αποδοτικότητα και στην ποιότητα του. Αρχικά η λειτουργία του σε υπολογιστές με χαμηλές επιδόσεις μπορεί να βελτιστοποιηθεί με την μείωση ποιότητας μοντέλων και ανάλυσης αυτών ανάλογα με τις δυνατότητες του συστήματος στο οποίο εκτελείται. Έτσι το παιχνίδι θα έχει την μέγιστη δυνατή ανάλυση αντικειμένων σε υπολογιστές υψηλών επιδόσεων αλλά θα υπάρχει και η δυνατότητα ομαλής εκτέλεσης αυτού από υπολογιστές χαμηλών επιδόσεων αντίστοιχα. Επακολούθως σημαντική βελτίωση αποτελεί η εμφάνιση του χάρτη μιας και το πάρκο στο οποίο διαδραματίζεται το παιχνίδι φαίνεται λιτό και άδειο. Κάτι τέτοιο μπορεί να υλοποιηθεί χρησιμοποιώντας παραπάνω αντικείμενα τα οποία υπάρχουν στα πραγματικά πάρκα όπως παγκάκια, παιδικές χαρές, λάμπες φωτισμού αλλά και μεγαλύτερο πλήθος χαρακτήρων με τους οποίους μπορεί να αλληλεπιδράσει ο παίκτης προσφέροντας έτσι μια πιο ψυχαγωγική εμπειρία ενώ ταυτόχρονα μπορεί να μαθαίνει περισσότερα μέσω διαφορετικών συμβουλών που θα αναφέρει κάθε χαρακτήρας. Τέλος το παιχνίδι θα μπορούσε να εμπλουτιστεί με περισσότερες λειτουργίες όπως προσθήκη μερικών ακόμα αποστολών αλλά και λειτουργίας εξερεύνησης του χάρτη αναζητώντας κρυμμένα αντικείμενα που θα βρίσκονται διάσπαρτα σε όλη την έκταση του χάρτη, κάτι το οποίο θα μπορούσε να ξεκλειδώσει ο παίκτης κατά την ολοκλήρωση των κύριων αποστολών.

Βιβλιογραφία:

1. Twinkl. (n.d.). What are Video Games? Teaching Wiki. Retrieved February 8, 2024, from https://www.twinkl.gr/teaching-wiki/video-games

2. New World Encyclopedia. (n.d.). Video game. Retrieved February 8, 2024, from https://www.newworldencyclopedia.org/entry/Video_game

3. Donovan, T., & Garriott, R. (2010). Replay: The history of video games (1st ed.). Yellow Ant.

4. Cleverism. (n.d.). Gaming industry: An introduction. Retrieved February 8, 2024, from https://www.cleverism.com/gaming-industry-introduction/

5. Designing Digitally. (n.d.). All about serious games: Types and purposes. Retrieved from https://www.designingdigitally.com/blog/all-about-serious-games-types-and-purposes

6. Djaouti, D., Alvarez, J., & Jessel, J.-P. (n.d.). Classifying Serious Games: The G/P/S model [PDF]. Retrieved June 26, 2015, from http://www.ludoscience.com/files/ressources/classifying_serious_games.pdf

7. Red Bull. (2020, August 26). Microsoft Flight Simulator 2020 Landmarks. Retrieved from https://www.redbull.com/ie-en/microsoft-flight-simulator-best-landmarks

8. Moro, C., Phelps, C., & Stromberga, Z. (2020). Utilizing serious games for physiology and anatomy learning and revision. Advances in Physiology Education, 44(3), 505–507. From https://journals.physiology.org/doi/full/10.1152/advan.00074.2020)

9. Growth Engineering. (2013, November 11). Serious games [Video]. YouTube. https://www.youtube.com/watch?v=JmG3fdptY_k

10. *Arm Limited. (n.d.). Gaming engines. Retrieved February 8, 2024, from https://www.arm.com/glossary/gaming-engines*

11. Codecademy. (n.d.). Unreal Engine: An introduction. Retrieved February 8, 2024, from https://www.codecademy.com/article/unreal-intro/

12. Mod DB. (n.d.). Unreal Engine 1. Retrieved from https://www.moddb.com/engines/unreal-engine-1

13. Cornell University Department of Computer Science. (n.d.). Computer Graphics Tutorial. Retrieved February 8, 2024, from http://www.graphics.cornell.edu/online/tutorial/

14. University of Leeds. (2015, January 6). Overview of computer graphics. Retrieved February 8, 2024, from https://web.archive.org/web/20150106160420/http://iss.leeds.ac.uk/info/306/giraphics/215/overview_of_computer_graphics/2

15. United Nations. (n.d.). What is climate change. Retrieved from https://www.un.org/en/climatechange/what-is-climate-change

16. DOANYS. (n.d.). Πλεονεκτήματα της ανακύκλωσης [Advantages of recycling]. Retrieved from https://www.doanys.gr/%CF%80%CE%BB%CE%B5%CE%BF%CE%BD%CE%B5%CE%BA%CF%84%CE%B7%CE%BC%CE%B1%CF%84%CE%B1-%CF%84%CE%B7%CF%83-%CE%B1%CE%BD%CE%B1%CE%BA%CF%85%CE%BA%CE%BB%CF%89%CF%83%CE%B7%CF%83/

17. National EMS Management Association. (n.d.). Benefits of recycling. Retrieved from https://nems.nih.gov/environmental-programs/pages/benefits-of-recycling

18. Boulder County. (n.d.). Reduce, reuse, recycle. Retrieved from https://bouldercounty.gov/environment/recycle/reduce-reuse-recycle/

19. My Disposal. (n.d.). Cardboard recycling and types of cardboard. Retrieved from https://mydisposal.com/cardboard-recycling-and-types-of-cardboard

20. McKinsey & Company. (2018, October). Climate impact of plastics. Retrieved from https://www.mckinsey.com/industries/chemicals/our-insights/climate-impact-of-plastics

Παράρτημα Κώδικα

PlayerMovement.cs

```
using UnityEngine;

public class PlayerMovement : MonoBehaviour
{
    [Header("Movement")]
    public float moveSpeed;
    public float runSpeed;

    public float groundDrag;

    [Header("Ground Check")]
    public float playerHeight;
    public LayerMask whatIsGround;
    bool grounded;

    public Transform orientation;
    public Transform player;

    float horizontalInput;
    float verticalInput;
    private Animator anim;

    Vector3 moveDirection;
    public AudioSource footstepsSound, runningSound, ambience;

    Rigidbody rb;

    private void Start()
    {
        ambience.enabled = true;
        ambience.Play();
        anim = GetComponentInChildren<Animator>();
        rb = GetComponent<Rigidbody>();
        rb.freezeRotation = true;
    }

    private void Update()
    {
        grounded = Physics.Raycast(transform.position, Vector3.down, playerHeight * 0.5f + 0.2f,
whatIsGround);
        MyInput();
        SpeedControl();

        //drag
        if (grounded)
            rb.drag = groundDrag;
        else
            rb.drag = 0;
    }
```

```
    private void MyInput()
    {
        horizontalInput = Input.GetAxisRaw("Horizontal");
        verticalInput = Input.GetAxisRaw("Vertical");
    }

    private void MovePlayer()
    {
        if (horizontalInput != 0.0f || verticalInput != 0.0f)
        {
            anim.SetFloat("Speed", 0.5f);
            footstepsSound.enabled = true;
        }
        else if (horizontalInput == 0.0f || verticalInput == 0.0f)
        {
            anim.SetFloat("Speed", 0);
            footstepsSound.enabled = false;
        }
        moveDirection = orientation.forward * verticalInput + orientation.right * horizontalInput;
        if (Input.GetKey(KeyCode.LeftShift))
        {
            rb.AddForce(moveDirection.normalized * runSpeed * 10f, ForceMode.Force);
            anim.SetFloat("Speed", 1);
            footstepsSound.enabled = false;
            runningSound.enabled = true;
        }
        else
        {
            rb.AddForce(moveDirection.normalized * moveSpeed * 10f, ForceMode.Force);
            runningSound.enabled = false;
        }
        player.transform.rotation = orientation.rotation;

    }

    private void SpeedControl()
    {
        Vector3 flatVel = new Vector3(rb.velocity.x, 0f, rb.velocity.z);

        //limit velocity if needed
        if (flatVel.magnitude > moveSpeed)
        {
            Vector3 limitedVel = flatVel.normalized * moveSpeed;
            rb.velocity = new Vector3(limitedVel.x, rb.velocity.y, limitedVel.z);
        }
    }

    private void Sprint()
    {
        if (Input.GetKey(KeyCode.LeftShift))
            {

            }
    }

}
```

PlayerCam.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PlayerCam : MonoBehaviour
{
    public float sensX;
    public float sensY;

    public Transform orientation;

    float xRotation;
    float yRotation;

    public bool isPaused;

    private void Start()
    {
        Cursor.lockState = CursorLockMode.Locked;
        Cursor.visible= false;

    }

    private void Update()
    {
        if (isPaused)
        {
        Cursor.lockState = CursorLockMode.None;
        Cursor.visible = true;
        return;
        }
        else if (!isPaused)
        {
            Cursor.lockState = CursorLockMode.Locked;
            Cursor.visible = false;
            //get mouse input
            float mouseX = Input.GetAxisRaw("Mouse X") * Time.fixedDeltaTime * sensX;
            float mouseY = Input.GetAxisRaw("Mouse Y") * Time.fixedDeltaTime * sensY;

            yRotation += mouseX;
            xRotation -= mouseY;

            xRotation = Mathf.Clamp(xRotation, -60f, 60f);

            transform.rotation = Quaternion.Euler(xRotation, yRotation, 0);
            orientation.rotation = Quaternion.Euler(0, yRotation, 0);
        }
    }
}
```

MoveCamera.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MoveCamera : MonoBehaviour
{
    public Transform cameraPosition;


    // Update is called once per frame
    void Update()
    {

        transform.position = cameraPosition.position;

    }
}
```

PlayerInv.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.Events;

public class PlayerInv : MonoBehaviour
{


    public int NumOfBottles, NumOfCans, NumOfPaper, score;
    public int BottlesDeposited, CansDeposited, PaperDeposited;
    public NPCSystem npcsystem;

    public UnityEvent<PlayerInv> OnBottleCollected; //event trigger when picking up a bottle
    public UnityEvent<PlayerInv> OnCanCollected;    //event trigger when picking up an aluminum can
    public UnityEvent<PlayerInv> OnPaperCollected;  //event trigger when picking up a papaer box/sheet

    public UnityEvent<PlayerInv> OnBottleDeposited; //event trigger when depositing up a bottle
    public UnityEvent<PlayerInv> OnCanDeposited;    //event trigger when depositing up a can
    public UnityEvent<PlayerInv> OnPaperDeposited;  //event trigger when depositing up paper
    public UnityEvent<PlayerInv> OnScoreChange; //event trigger when depositing to change (increase) score

    public UnityEvent<PlayerInv> OnMission1Done;    //event trigger when doing mission 1

    public UnityEvent<PlayerInv> M2PanelSwap; //swaps mission panels
    public UnityEvent<PlayerInv> BridgeAppear;  //trigger for bridge to appear when player is done depositing for mission 2
    public UnityEvent<PlayerInv> OnBootCollection;  //event trigger when player has picked up the jump boot upgrade

public UnityEvent<PlayerInv> OnMission2Done;    //event trigger when doing mission 2

    //public UnityEvent<PlayerInv> M3PanelSwap;
    public UnityEvent<PlayerInv> BoxAppear; //makes collectable boxes appear
    public UnityEvent<PlayerInv> BoxDeposit;    //triggers when said boxes are deposited to enable scalable boxes for the factory
    //public UnityEvent<PlayerInv> OnFuseDisable;
```

```
    public UnityEvent<PlayerInv> OnMission3Done;    //event trigger when doing mission 3
    public UnityEvent<PlayerInv> GameDone;  //event trigger when player has finished everything, disables
mission box info

    public void BottleCollected()
    {
        NumOfBottles++;
        OnBottleCollected.Invoke(this);
    }
    public void CanCollected()
    {
        NumOfCans++;
        OnCanCollected.Invoke(this);
    }
    public void PaperCollected()
    {
        NumOfPaper++;
        OnPaperCollected.Invoke(this);
    }

    public void BootCollected()
    {
        OnBootCollection.Invoke(this);
    }

    public void PaperBoxCollected()
    {
        NumOfPaper+=10;
        OnPaperCollected.Invoke(this);
    }

    public void BottleDeposit()
    {
        BottlesDeposited += NumOfBottles;
        if (BottlesDeposited == 10)
        {
            RenderSettings.fogDensity = 0.015f;
            OnM1Done();


        }
        score += NumOfBottles;
        NumOfBottles = 0;
        OnBottleDeposited.Invoke(this);
        OnScoreChange.Invoke(this);
    }
    public void CanDeposit()

    {

        CansDeposited += NumOfCans;

        score+= (NumOfCans*2);

        NumOfCans = 0;

        OnCanDeposited.Invoke(this);

        if (CansDeposited == 12)

        {

            RenderSettings.fogDensity = 0.012f;
```

```
            OnBridgeAppear();
        }
        OnScoreChange.Invoke(this);
    }

    public void PaperDeposit()
    {
        PaperDeposited += NumOfPaper;
        score += NumOfPaper;
        NumOfPaper = 0;
        if (PaperDeposited == 40)
        {
            BoxDeposit.Invoke(this);
            RenderSettings.fogDensity = 0.010f;
        }
        OnPaperDeposited.Invoke(this);
        OnScoreChange.Invoke(this);
    }

    public void OnBoxAppear()
    {
        BoxAppear.Invoke(this);
    }

    public void OnM1Done()
    {
        OnMission1Done.Invoke(this);
    }

    public void OnM2Done()
    {
        OnMission2Done.Invoke(this);
    }

    public void OnM3Done()
    {
        OnMission3Done.Invoke(this);
    }

    public void FuseToggle()
    {
        OnM3Done();
      // Application.Quit();
    }

    public void OnBridgeAppear()
    {
        BridgeAppear.Invoke(this);
    }

    public void OnM2PanelSwap()
    {
        M2PanelSwap.Invoke(this);
    }

    public void OnGameDone()
    {
        GameDone.Invoke(this);
    }
}
```

PlayerJump.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PlayerJump : MonoBehaviour
{
    public Vector3 jump;
    public float jumpForce = 2.0f;
    public bool isEnabled = false;

    public bool isGrounded;
    Rigidbody rb;
    void Start()
    {
        rb = GetComponent<Rigidbody>();
        jump = new Vector3(0.0f, 2.0f, 0.0f);
    }

    void OnCollisionStay()
    {
        isGrounded = true;
    }
    public void JumpEnable()
    {
        isEnabled = true;
    }
    void Update()
    {
        if (isEnabled)
        {
            if (Input.GetKeyDown(KeyCode.Space) && isGrounded)
            {

                rb.AddForce(jump * jumpForce, ForceMode.Impulse);
                isGrounded = false;
            }
        }
    }
}
```

NPCSystem.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.Events;

public class NPCSystem : MonoBehaviour
{
    public PlayerInv playerInv;
    public Dialogue dialogue;
    public GameObject dialoguePanel;
    public GameObject interactText;
    public bool stillTalking;
    public bool FirstInteract;
    public int MissionNum;
    public UnityEvent<NPCSystem> OnMissionChange;
    public AudioAwake aud;


    // Start is called before the first frame update
    void Start()
    {
        aud.MusicStart();
        MissionNum = 1;
        FirstInteract = true;
        stillTalking = false;

    }

    // Update is called once per frame
    void Update()
    {
        if (dialogue.playerCheck && Input.GetKeyDown(KeyCode.E))
            if (!stillTalking)
            {
                stillTalking = true;
                dialoguePanel.gameObject.SetActive(true);
                dialogue.StartDialogue();
            }
    }

    private void OnTriggerEnter(Collider c)
    {
        PlayerInv playerInv = c.GetComponent<PlayerInv>();
        if (c.CompareTag("Player"))
        {
            interactText.SetActive(true);
            dialogue.playerCheck = true;

        }
    }

    private void OnTriggerExit(Collider c)
    {
        PlayerInv playerInv = c.GetComponent<PlayerInv>();
        if (c.CompareTag("Player"))
        {
            interactText.SetActive(false);
            dialogue.playerCheck = false;
            dialoguePanel.SetActive(false);
        }
    }
```

```
    public void MissionComplete()
    {
        OnMissionChange.Invoke(this);
    }

    public void MissionNumIncr()
    {
        MissionNum++;
    }
    public void FirstInteractToggle()
    {
        FirstInteract = true;
    }
}
```

UnlockArea.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class UnlockArea : MonoBehaviour
{
    public PlayerInv playerInv;
    [SerializeField] GameObject invisWall;

    public void DestoryWall(PlayerInv playerInv)
    {
        invisWall.SetActive(false);
    }
}
```

JumpBoot.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class JumpBoot : MonoBehaviour
{
    public PlayerInv plInv;
    [SerializeField] GameObject spotlight;
    private void OnTriggerEnter(Collider c)
    {
        PlayerInv playerInv = c.GetComponent<PlayerInv>();

        if (playerInv)
        {
            playerInv.BootCollected();
            Destroy(gameObject);
            Destroy(GetComponent<Light>());
            plInv.OnM2Done();
        }

    }
```

AudioAwake.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.Events;

public class AudioAwake : MonoBehaviour
{
    [SerializeField]
    private AudioSource m_AudioSource;
    public UnityEvent<AudioAwake> PlayMusic;
    void Start()
    {

        Awake();
    }

    void Update()
    {

    }
    private void Awake()
    {

        AudioSource audioSource = GetComponent<AudioSource>();
        DontDestroyOnLoad(audioSource);
    }

   public void MusicStart()
    {
        PlayMusic.Invoke(this);
    }
}
```

AudioManager.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.Audio;

public class AudioManager : MonoBehaviour
{
    public AudioMixer theMixer;

    void Start()
    {
        if (PlayerPrefs.HasKey("MasterVol")) {
            theMixer.SetFloat("MasterVol", PlayerPrefs.GetFloat("MasterVol"));
        }

        if (PlayerPrefs.HasKey("MusicVol")) {
            theMixer.SetFloat("MusicVol", PlayerPrefs.GetFloat("MusicVol"));
        }

        if (PlayerPrefs.HasKey("SFXVol")) {
            theMixer.SetFloat("SFXVol", PlayerPrefs.GetFloat("SFXVol"));
        }
    }
}
```

Dialogue.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;
public class Dialogue : MonoBehaviour
{
    public bool playerCheck=false;
    public TextMeshProUGUI textComponent;
    public GameObject missionTxt;
    public GameObject talkWall;
    public PlayerInv PInv;
    public string[] lines;
    public float textSpeed;
    private int index;
    public NPCSystem npcsystem;

    void Start()
    {
        gameObject.SetActive(false);
        textComponent.text = string.Empty;
        npcsystem.MissionNum = 1;
        npcsystem.FirstInteract = true;

    }

    void Update()
    {
        if (playerCheck)
        {
            if (Input.GetMouseButtonDown(0) )

            {
                if (textComponent.text == lines[index])
                {
                    NextLine();
                }
                else
                {
                    StopAllCoroutines();
                    textComponent.text = lines[index];
                }
            }
        }
    }

    public void StartDialogue()
    {
        MissionCheck();
        gameObject.SetActive(true);
        index =0;
        StartCoroutine(TypeLine());

    }
    IEnumerator TypeLine()
    {
        foreach (char c in lines[index].ToCharArray())
            {
            textComponent.text += c;
            yield return new WaitForSeconds(textSpeed);
        }
    }
```

```
void NextLine()
    {
        if (index < lines.Length - 1)
        {
            index++;
            textComponent.text = string.Empty;
            StartCoroutine(TypeLine());
        }
        else
        {
            textComponent.text = string.Empty;
            if (npcsystem.MissionNum == 1 && npcsystem.FirstInteract == true)
            {
                missionTxt.SetActive(true);
                talkWall.SetActive(false);
            }
            if (npcsystem.MissionNum == 2 && npcsystem.FirstInteract == true)
            {
                PInv.OnM2PanelSwap();
            }
            if (npcsystem.MissionNum == 3 && npcsystem.FirstInteract == true)
            {
                PInv.OnBoxAppear();
            }
            if (npcsystem.MissionNum == 4 && npcsystem.FirstInteract == true)
            {
                PInv.OnGameDone();
            }
            npcsystem.FirstInteract = false;
            npcsystem.stillTalking = false;
            gameObject.SetActive(false);

        }
    }

    public void MissionCheck()
    {
        switch (npcsystem.MissionNum)
        {
            case 1:
                if (npcsystem.FirstInteract)
                {
                    lines[0] = "Hello, welcome to the park. This was once a great green park for all people to enjoy";
                    lines[1] = "Nowdays with all the factories producing new plastics and new items in general we have a
lot of pollution from their gasses ";
                    lines[2] = "That is why we have such a thick pollution fog, only way we can counteract it is by
reducing production waste by recycling, so please for starters try and get 10 of these plastic bottles scattered
around";

                }

                else
                {
                    lines[0] = "Every year plastic production factories contribute to 3.4% of worldwide greenhouse
pollution" ;
                    lines[1] = "So they create this thick fog, please help us reduce it for the good of the planet";
                    lines[2] = "Collect 10 plastic bottles to reduce the amount of pollution fog";

                }
                break;
```

```
            case 2:
            if (npcsystem.FirstInteract)
            {
               lines[0] = "Great, as you saw the fog dispersed when you recycled those plastic bottles. In general the Orange Bins are for plastic recycling";
               lines[1] = "You might have seen that there are two more colored Bins there, the Yellow one is for aluminum and the Blue one is for paper";
               lines[2] = "Now, there is a lake on the east side that block the way to an upgrade you will need, by collecting 12 aluminum that is scattered around you can upcycle it to make a bridge!";

            }
            else
            {
               lines[0] = "To cross the lake you need to create a bridge from recycled aluminum";
               lines[1] = "Collect 12 aluminum that is scattered around the park to create the bridge";
               lines[2] = "And remember that it takes 95% less energy to recycle aluminum than it does to make it from raw materials.";
            }
            break;

            case 3:
            if (npcsystem.FirstInteract)
            {
               lines[0] = "Great, now you have the jump boots";
               lines[1] = "Now we can proceed to the final mission, you need to get to the illegal factory and turn everything off";
               lines[2] = "But to do that you need to collect and deposit 4 boxes to place in order to reach the switch with your new jumping ability";

            }
            else
            {
               lines[0] = "The Factory is West of here, just follow the road";
               lines[1] = "With upcycling these 4 boxes you can use 'em to jump onto the factory switch";
               lines[2] = "While also recycling useful items that can be almost entirely re-used, recycling boxes also reduces processing pollution by 95%!";
            }
            break;

            case 4:
            if (npcsystem.FirstInteract)
            {
               lines[0] = "Congratulations, you have completed the game! Now you are free to roam around the map if you wish and see the whole thing for yourself";
               lines[1] = "And always remember what you have learnt!, Recycling conserves energy, reduces air and water pollution, reduces greenhouse gases, and conserves natural resources.";
               lines[2] = "By reusing aluminum, paper, glass, plastics, and other materials, we can save production and energy costs, and reduce the negative impacts that the extraction and processing of virgin materials has on the environment";

            }
            else
            {
               lines[0] = " Recycling boxes reduces processing pollution by 95% because the fiber has already been processed.";
               lines[1] = "Throughout their lifecycle, plastics have a significant carbon footprint and emit 3.4% of global greenhouse gas emissions";
               lines[2] = "When depositing materials remember than yellow bins are for plastics, blue for paper and orange for aluminum! Recycling diverts waste away from landfills and incinerators, which reduces the harmful effects of pollution and emissions.";

            }
            break;
```

AluCan.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class AluCan : MonoBehaviour
{
    private void OnTriggerEnter(Collider c)
    {
        PlayerInv playerInv = c.GetComponent<PlayerInv>();

        if (playerInv)
        {
            playerInv.CanCollected();

            Destroy(gameObject);
        }

    }
}
```

PaperBox.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PaperBox : MonoBehaviour
{
    private void OnTriggerEnter (Collider c)
    {
        PlayerInv playerInv = c.GetComponent<PlayerInv>();

        if (playerInv)
        {
            playerInv.PaperBoxCollected();

            Destroy(gameObject);
        }

    }
}
```

PlasticBottles.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PlasticBottles : MonoBehaviour
{
    private void OnTriggerEnter(Collider c)
    {
        PlayerInv playerInv = c.GetComponent<PlayerInv>();

        if (playerInv)
        {
            playerInv.BottleCollected();

            Destroy(gameObject);
        }

    }
}
```

AluBinDeposit.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class AluBinDeposit : MonoBehaviour
{
    [SerializeField] private bool triggerActive = false;
    public PlayerInv playerInv;
    [SerializeField] GameObject aluTxt;

    private void OnTriggerEnter(Collider c)
    {
        PlayerInv playerInv = c.GetComponent<PlayerInv>();

        if ((playerInv) && (c.CompareTag("Player")))
        {

            triggerActive = true;
            aluTxt.SetActive(true);

        }
    }

    private void OnTriggerExit(Collider c)
    {
        PlayerInv playerInv = c.GetComponent<PlayerInv>();
        if (c.CompareTag("Player"))
        {
            triggerActive = false;
            aluTxt.SetActive(false);
        }
    }
```

```
private void Update()
    {

        if (triggerActive && Input.GetKeyDown(KeyCode.E))
        {
            playerInv.CanDeposit();
        }
    }

}
```

PlasticBinDeposit.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PlasticBinDeposit : MonoBehaviour
{
    [SerializeField] private bool triggerActive = false;
    public PlayerInv playerInv;
    [SerializeField] GameObject plasticTxt;


    private void OnTriggerEnter(Collider c)
    {
        PlayerInv playerInv = c.GetComponent<PlayerInv>();

        if ((playerInv) && (c.CompareTag("Player")))
        {

            triggerActive= true;
            plasticTxt.SetActive(true);
        }
    }

    private void OnTriggerExit(Collider c)
    {
        PlayerInv playerInv = c.GetComponent<PlayerInv>();
        if (c.CompareTag("Player"))
        {
            triggerActive = false;
            plasticTxt.SetActive(false);
        }
    }
    private void Update()
    {

        if (triggerActive && Input.GetKeyDown(KeyCode.E))
        {
            playerInv.BottleDeposit();
        }
    }

}
```

PaperBinDeposit.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PaperBinDeposit : MonoBehaviour
{
    [SerializeField] private bool triggerActive = false;
    public PlayerInv playerInv;
    [SerializeField] GameObject paperTxt;

    private void OnTriggerEnter(Collider c)
    {
        PlayerInv playerInv = c.GetComponent<PlayerInv>();

        if ((playerInv) && (c.CompareTag("Player")))
        {

            triggerActive = true;
            paperTxt.SetActive(true);
        }
    }

    private void OnTriggerExit(Collider c)
    {
        PlayerInv playerInv = c.GetComponent<PlayerInv>();
        if (c.CompareTag("Player"))
        {
            triggerActive = false;
            paperTxt.SetActive(false);
        }
    }
    private void Update()
    {

        if (triggerActive && Input.GetKeyDown(KeyCode.E))
        {
            playerInv.PaperDeposit();
```

BinTooltip.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;
public class BinTooltip : MonoBehaviour
{
    public bool playerCheck = false;
    public TextMeshProUGUI binTooltipTxt;
    public BinTxtTrigger binRef;
    public string[] lines;
    public float textSpeed;
    private int index;

    void Start()
    {
        binTooltipTxt.text = string.Empty;

    }

    void Update()
    {
        if (playerCheck)
        {
            if (Input.GetMouseButtonDown(0))

            {
                if (binTooltipTxt.text == lines[index])
                {
                    NextLine();
                }
                else
                {
                    StopAllCoroutines();
                    binTooltipTxt.text = lines[index];
                }
            }
        }
    }

    public void StartDialogue()
    {
        index = 0;
        StartCoroutine(TypeLine());
    }
    IEnumerator TypeLine()
    {
        foreach (char c in lines[index].ToCharArray())
        {
            binTooltipTxt.text += c;
            yield return new WaitForSeconds(textSpeed);
        }
    }
```

```
    void NextLine()
    {
        if (index < lines.Length - 1)
        {
            index++;
            binTooltipTxt.text = string.Empty;
            StartCoroutine(TypeLine());
        }
        else
        {
            binTooltipTxt.text = string.Empty;
            gameObject.SetActive(false);
            binRef.binWalls.SetActive(false);
            binRef.done=true;

        }
    }


}
```

BinTxtTrigger.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class BinTxtTrigger : MonoBehaviour
{
    [SerializeField] BinTooltip binTool;
    [SerializeField] GameObject binToolPanel;
    public GameObject binWalls;
    public bool done;
    // Start is called before the first frame update
    void Start()
    {
        binToolPanel.SetActive(false);
        binWalls.SetActive(false);
    }

    // Update is called once per frame
    void Update()
    {

    }

    private void OnTriggerEnter(Collider c)
    {
        PlayerInv playerInv = c.GetComponent<PlayerInv>();

        if ((playerInv) && (c.CompareTag("Player")))
        {
            if (!done)
            {
                binTool.playerCheck = true;
                binWalls.SetActive(true);
                binToolPanel.SetActive(true);
                binTool.StartDialogue();
            }

        }
    }
```

```
    private void OnTriggerExit(Collider c)
    {
        PlayerInv playerInv = c.GetComponent<PlayerInv>();

        if ((playerInv) && (c.CompareTag("Player")))
        {

            binTool.playerCheck = false;
            binToolPanel.SetActive(false);


        }
    }
}
```

ItemRotation.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ItemRotation : MonoBehaviour
{
    public float speed = 10.0f;
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {
        transform.Rotate(0, speed * Time.deltaTime, 0, Space.Self);
    }
}
```

FuseBoxInteract.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class FuseBoxInteract : MonoBehaviour
{
    [SerializeField] GameObject fuseTxt;
    [SerializeField] bool triggerActive;
    public PlayerInv playerInv;

    void Start()
    {
        triggerActive = false;
    }
```

```
    void Update()
    {
        if (triggerActive && Input.GetKeyDown(KeyCode.E))
        {
            playerInv.FuseToggle();
        }
    }

    private void OnTriggerEnter(Collider c)
    {
        PlayerInv playerInv = c.GetComponent<PlayerInv>();

        if ((playerInv) && (c.CompareTag("Player")))
        {
            triggerActive = true;
            fuseTxt.SetActive(true);
        }
    }
    private void OnTriggerExit(Collider c)
    {
        PlayerInv playerInv = c.GetComponent<PlayerInv>();
        if (c.CompareTag("Player"))
        {
            triggerActive = false;
            fuseTxt.SetActive(false);
        }
    }
}
```

InventoryUI.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;

public class InventoryUI : MonoBehaviour
{
    private TextMeshProUGUI bottleText, canText, paperText, score;

    void Start()
    {
        bottleText= GetComponent<TextMeshProUGUI>();
        canText = GetComponent<TextMeshProUGUI>();
        paperText = GetComponent<TextMeshProUGUI>();
        score = GetComponent<TextMeshProUGUI>();
    }

    public void UpdateBottleText(PlayerInv playerInv)
    {
        bottleText.text= playerInv.NumOfBottles.ToString();
    }
    public void UpdateCanText(PlayerInv playerInv)
    {
        canText.text = playerInv.NumOfCans.ToString();
    }
```

```
    public void UpdatePaperText(PlayerInv playerInv)
    {
        paperText.text = playerInv.NumOfPaper.ToString();
    }

    public void UpdateScoreText(PlayerInv playerInv)
    {
        score.text = playerInv.score.ToString();
    }
}
```

MainMenu.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class MainMenu : MonoBehaviour
{
    public string firstLevel;
    public GameObject optionsScreen;

    // Start is called before the first frame update
    void Start()
    {
        Cursor.lockState = CursorLockMode.None;
        Cursor.visible = true;
    }

    // Update is called once per frame
    void Update()
    {

    }

    public void StartGame()
    {
        SceneManager.LoadScene(firstLevel);
    }

    public void OpenOptions()
    {
        optionsScreen.SetActive(true);
    }

    public void CloseOptions()
    {
        optionsScreen.SetActive(false);
    }

    public void QuitGame()
    {
        Application.Quit();
    }

}
```

OptionsScreen.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using TMPro;
using UnityEngine.Audio;

public class OptionsScreen : MonoBehaviour
{

    public Toggle fullscreenTog, vsyncTog;

    public List<ResItem> resolutions = new List<ResItem>();
    private int selectedResolution;

    public TMP_Text resolutionLabel;

    public AudioMixer theMixer;

    public TMP_Text masterLabel, musicLabel, sfxLabel;
    public Slider masterSlider, musicSlider, sfxSlider;


    void Start()
    {
        fullscreenTog.isOn = Screen.fullScreen;

        if(QualitySettings.vSyncCount == 0 )
        {
            vsyncTog.isOn = false;
                }
        else
        {
            vsyncTog.isOn = true;
        }

        bool foundRes = false;
        for(int i = 0; i < resolutions.Count; i++)
        {
            if(Screen.width == resolutions[i].horizontal && Screen.height == resolutions[i].vertical)
            {
                foundRes = true;

                selectedResolution = i;

                UpdateResLabel();
            }
        }

        if(!foundRes)
        {
            ResItem newRes = new ResItem();
            newRes.horizontal = Screen.width;
            newRes.vertical = Screen.height;

            resolutions.Add(newRes);
            selectedResolution = resolutions.Count-1;
            UpdateResLabel();
        }
```

```
        float vol = 0f;
        theMixer.GetFloat("MasterVol", out vol);
        masterSlider.value = vol;

        theMixer.GetFloat("MusicVol", out vol);
        musicSlider.value = vol;

        theMixer.GetFloat("SFXVol", out vol);
        sfxSlider.value = vol;

        masterLabel.text = Mathf.RoundToInt(masterSlider.value + 80).ToString();
        musicLabel.text = Mathf.RoundToInt(musicSlider.value + 80).ToString();
        sfxLabel.text = Mathf.RoundToInt(sfxSlider.value + 80).ToString();
    }

    void Update()
    {

    }

    public void ReLeft()
    {
        selectedResolution--;
        if( selectedResolution < 0 )
        {
            selectedResolution = 0;
        }

        UpdateResLabel();
    }

    public void ResRight()
    {
        selectedResolution++;
        if (selectedResolution > resolutions.Count-1 )
        {
            selectedResolution = resolutions.Count-1;
        }
        UpdateResLabel();
    }

    public void UpdateResLabel()
    {
        resolutionLabel.text = resolutions[selectedResolution].horizontal.ToString() + " x " +
resolutions[selectedResolution].vertical.ToString();
    }

    public void ApplyGraphics()
    {

        if(vsyncTog.isOn )
        {
            QualitySettings.vSyncCount = 1;
        }
        else
        {
            QualitySettings.vSyncCount = 0;
        }

        Screen.SetResolution(resolutions[selectedResolution].horizontal, resolutions[selectedResolution].vertical,
fullscreenTog.isOn);
    }
```

MenuController.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

public class MenuController : MonoBehaviour
{
    public string mainMenuScene;
    public GameObject pauseMenu;
    public bool isPaused;
    public PlayerCam moveCamera;

    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {
        if(Input.GetKeyUp(KeyCode.Escape))
        {
            if (isPaused)
            {
                ResumeGame();
            }
            else
            {
                isPaused = true;
                pauseMenu.SetActive(true);
                Time.timeScale = 0f;
                moveCamera.isPaused = true;
                Cursor.lockState = CursorLockMode.None;
                AudioListener.pause = true;
            }
        }
    }

    public void ResumeGame()
    {
        isPaused = false;
        pauseMenu.SetActive(false);
        Time.timeScale = 1f;
        moveCamera.isPaused = false;
        Cursor.lockState = CursorLockMode.Locked;
        Cursor.visible = false;
        AudioListener.pause = false;
    }

    public void ReturnToMain()
    {
        Time.timeScale = 1f;
        moveCamera.isPaused = false;
        AudioListener.pause = false;
        SceneManager.LoadScene(mainMenuScene);

    }

}
```