ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ
ΣΧΟΛΗ ΔΙΟΙΚΗΣΗΣ ΚΑΙ ΟΙΚΟΝΟΜΙΑΣ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΜΜΕ

ΔΗΜΙΟΥΡΓΙΑ ΠΑΙΧΝΙΔΙΟΥ ΜΕ ΧΡΗΣΗ ΤΗΣ ΣΧΕΔΙΑΣΤΙΚΗΣ ΠΛΑΤΦΟΡΜΑΣ UNITY

Τσαβαρή Καλλιόπη-Στυλιανή

Εισηγητής : ΚΟΥΤΡΑΣ ΑΘΑΝΑΣΙΟΣ

Πύργος, 2016

ΥΠΕΥΘΥΝΗ ΔΗΛΩΣΗ ΠΕΡΙ ΜΗ ΛΟΓΟΚΛΟΠΗΣ

Βεβαιώνω/ουμε ότι είμαι/είμαστε ο/οι συγγραφέας/εις αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα/είχαμε για την προετοιμασία της, είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία.

Επίσης, έχω/έχουμε αναφέρει τις οποίες πηγές από τις οποίες έκανα /κάναμε χρήση δεδομένων, ιδεών η λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες.

Ακόμη δηλώνω/ουμε ότι αυτή η γραπτή εργασία προετοιμάστηκε από εμένα/εμάς προσωπικά και αποκλειστικά και ειδικά για την συγκεκριμένη πτυχιακή εργασία ότι θα αναλάβω/ουμε πλήρως τις συνέπειες εάν η εργασία αυτή αποδειχτεί ότι δεν μου/μας ανήκει.

ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΣΠΟΥΔΑΣΤΗ 1	ΑΡΙΘ.ΜΗΤΡΩΟΥ	ΥΠΟΓΡΑΦΗ
ΤΣΑΒΑΡΗ ΚΑΛΛΙΟΠΗ-ΣΤΥΛΙΑΝΗ	1915	
ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΣΠΟΥΔΑΣΤΗ 2	ΑΡΙΘ.ΜΗΤΡΩΟΥ	ΥΠΟΓΡΑΦΗ
ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΣΠΟΥΔΑΣΤΗ 3	ΑΡΙΘ.ΜΗΤΡΩΟΥ	ΥΠΟΓΡΑΦΗ

Περιεχόμενα

Εικόνες

Σχεδιαγράμματα

Ευχαριστίες

Ευχαριστώ όλους τους καθηγητές μου για την πολύτιμη βοήθειά τους καθ' όλη την διάρκεια της φοίτησής μου στο τμήμα Πληροφορικής και ΜΜΕ και ιδιαίτερα τον επιβλέποντα καθηγητή της πτυχιακής μου κο Αθανάσιο Κούτρα που με καθοδήγησε έτσι ώστε να μπορέσω να τελειοποιήσω την εργασία μου. Επίσης, θα ήθελα να ευχαριστήσω τους φίλους μου που χάρη σ' αυτούς μπόρεσα να φτάσω μέχρι εδώ. Τέλος, ευχαριστώ πολύ τους γονείς και τον αδερφό μου που με στήριξαν σε οποιαδήποτε απόφαση μου όπως και στις δυσκολίες που αντιμετώπισα κατά την διάρκεια των σπουδών μου.

Περίληψη

Σε αυτή την πτυχιακή, στο πρώτο μέρος, αναφέρεται η θεωρία των video games και του design τους. Βλέπουμε αναλυτικά τα βήματα που πρέπει να ακολουθήσει ο δημιουργός/σχεδιαστής ενός παιχνιδιού, καθώς και τις δυσκολίες και τα προβλήματα που αντιμετωπίζει πριν ξεκινήσει και κατά την διάρκεια του project του. Επιπλέον, αναφέρονται τα βασικά συστατικά και η δομή ενός video game, τι και ποιος είναι ο game world, ποια είναι η χρησιμότητα του και ποιες οι διαστάσεις του. Ακόμη, περιλαμβάνονται οι δημοφιλέστερες κατηγορίες παιχνιδιών από την αρχή της δημιουργίας των games μέχρι και σήμερα.

Τέλος, στο δεύτερο μέρος, αναγράφεται αναλυτικά η διαδικασία που ακολουθήθηκε για να δημιουργηθεί το παιχνίδι με τίτλο "Rabbit Runner", όπως και μία αναφορά στις game engines που είναι διαθέσιμες στην αγορά για την δημιουργία ενός game με κύρια την Unity3D, η οποία χρησιμοποιήθηκε για την δημιουργία αυτού του project.

Abstract

The first part of this dissertation focuses on the theory of video games and design. We trace the steps that need to be followed by the game developer/designer in depth, as well as the difficulties and problems they might face from the conception of an idea to the implementation of it as a full scale project. We also examine the basic contents and structure of a video game, analyze what the term "game world" means and which are its uses and dimensions. Furthermore, some of the most popular genres since the beginning of games are presented.

In the second part of this thesis, we present a step by step guide of the procedure that was followed for the creation of the "Rabbit Runner" platform game. Also, we mention many of the currently available game engines on the market, that allow us to create our own game, focusing more on Unity3D which was used for the completion of this project.

ΜΕΡΟΣ ΠΡΩΤΟ: **ΘΕΩΡΙΑ ΤΩΝ** VIDEO GAMES **ΚΑΙ** DESIGN

1 Εισαγωγή

Με την αυξανόμενη ανάπτυξη της τεχνολογίας στις μέρες μας, το μεγαλύτερο ποσοστό του πληθυσμού έχει εύκολη πρόσβαση στον κόσμο των video games. Οι περισσότεροι από αυτούς μπορούν να παίζουν ένα ή και περισσότερα παιχνίδια σε καθημερινή βάση, ενώ άλλοι να είναι αυτοί οι οποίοι δημιουργούν τα παιχνίδια.

Είναι πάρα πολύ εύκολο και διασκεδαστικό να παίζει κάποιος ένα video game. Μπορεί με ευχαρίστηση να κάτσει ώρες ολόκληρες και να ασχολείται με αυτό, χωρίς κανένα δισταγμό ή παράπονο. Από την άλλη, είναι πάρα πολύ δύσκολο και χρονοβόρο να δημιουργήσει κάποιος ένα από αυτά. Οι δημιουργοί των παιχνιδιών θα πρέπει να κάνουν ατελείωτες μελέτες σε πολλούς τομείς για να αρχίσουν να φτιάχνουν το παιχνίδι τους. Πρέπει να προσέξουν τους κρυμμένους κινδύνους και τα τυχόν τεχνικά προβλήματα που μπορεί να υπάρξουν στον δρόμο τους, καθώς και να μελετήσουν πολύ προσεκτικά την αγορά στην οποία θα το δημοσιεύσουν.

Η διαδικασία την οποία πρέπει να ακολουθήσει ο δημιουργός ενός παιχνιδιού είναι συγκεκριμένη. Αρχικά πρέπει να μελετήσει την συμπεριφορά της αγοράς στην οποία θέλει να δημοσιεύσει το παιχνίδι του. Θα πρέπει να βρει και να συνεργαστεί με κάποια εταιρεία δημοσίευσης παιχνιδιών, αλλιώς μπορεί να το δημοσιεύσει σε δικό του ιστότοπο. Να στοχεύσει σε ένα συγκεκριμένο target group, ώστε να αποφασίσει τι ακριβώς θα περιλαμβάνει το παιχνίδι του. Έπειτα, θα πρέπει να αποφασίσει την ιστορία του παιχνιδιού και να προβλέψει με όσο μεγαλύτερη ακρίβεια μπορεί, τα προβλήματα που θα αντιμετωπίσει στη συνέχεια. Προχωράει στον προγραμματισμό και την δημιουργία των γραφικών και σε περίπτωση που δεν φτιάξει ο ίδιος τα γραφικά του θα πρέπει να συνεργαστεί με κάποιον γραφίστα ώστε να έχει το επιθυμητό αποτέλεσμα. Τέλος, ολοκληρώνει το παιχνίδι και στη συνέχεια το δημοσιεύει σε μία πλατφόρμα όπου επιθυμεί (για παράδειγμα ηλεκτρονικό υπολογιστή).

Το παιχνίδι αυτό που δημιουργήθηκε με τόσο κόπο, μπορεί να το παίξει οποιοσδήποτε απλά και μόνο χρησιμοποιώντας τον υπολογιστή ή το κινητό του.

Μέχρι και τα τελευταία χρόνια ήμαστε συνηθισμένοι στην ιδέα ότι τα video games βγαίνουν απαραίτητα από μεγάλο-εταιρίες και πως αν ήθελε κάποιος να είναι game developer θα έπρεπε να βρει δουλειά σε μία από αυτές. Στις μέρες μας όμως, είναι πολύ διαδεδομένο το είδος indie των video games όπου είναι παιχνίδια τα οποία έχουν φτιαχτεί από stand-alone game developers. Είναι, λοιπόν, ευκολότερο πλέον, με την ανάπτυξη της τεχνολογίας και την διάδοση των μηχανών σχεδιασμού παιχνιδιών (game engines να φτιάξει κάποιος το δικό του video game μόνος του και να το δημοσιοποιήσει ο ίδιος είτε στο διαδίκτυο είτε σε κάποιο store για κινητά (play store, app store κ.λπ.).

2 **Θεωρία του** Game Development

2.1 **Τι πρέπει να λάβει υπόψιν του ένας** game developer **και η ομάδα του για την δημιουργία ενός παιχνιδιού**

Ένας game developer και η ομάδα του θα πρέπει πριν ξεκινήσει την δημιουργία του παιχνιδιού του να λάβει υπόψιν του κάποιους παράγοντες. Πρέπει να ακολουθήσει ένα μονοπάτι πολύ προσεκτικά και να προσέξει να μην παρασυρθεί από λανθασμένες υποθέσεις που μπορούν μόνο να τον αποπροσανατολίσουν από τον αρχικό στόχο του. Θα πρέπει να ελιχθεί και να αποφύγει τις υπερβολικές σκέψεις, καθώς πρέπει να έχει ξεκάθαρα στο μυαλό του ποιες από αυτές είναι άσκοπες και ποιες αφορούν τον στόχο του.

Ένα μεγάλο λάθος που κάνουν πολλοί game developers είναι να δημιουργούν πολύ μεγάλα και πολύπλοκα σχέδια για το μελλοντικό τους παιχνίδι. Δημιουργούν κείμενα σχεδιασμού, τα λεγόμενα Design Documents, και προσπαθούν να γράψουν σε αυτά με κάθε λεπτομέρεια τι ακριβώς θα περιέχει το παιχνίδι τους από τον κώδικα και τα μηχανικά μέχρι τον σχεδιασμό και την παραγωγή του. Αυτό μπορεί να είναι λάθος κυρίως γιατί υπάρχει χρόνος που σπαταλάται άσκοπα χωρίς να έχει κάποιο συγκεκριμένο αποτέλεσμα στον τελικό στόχο. Έπειτα προχωρούν στην υλοποίηση αυτού και μετά στον έλεγχο, όπου εκεί παρουσιάζονται πολλές φορές προβλήματα που δεν επιδέχονται εύκολα λύσεις γιατί είναι συνδεδεμένα με τον κώδικα του παιχνιδιού με τέτοιο τρόπο, που θα πρέπει να ξανά φτιάξουν τα πάντα από την αρχή έτσι ώστε να λυθεί το πρόβλημα.

Ο καλύτερος, λοιπόν, τρόπος για να ακολουθήσει ο game developer ένα πλάνο και να φτιάξει το παιχνίδι του, είναι η λεγόμενη μέθοδος της προσεγγιστικής επανάληψης (Iteration) (Sylvester, 2013). Σύμφωνα με αυτή, καταγράφει το πρώτο κομμάτι του πλάνου του, προχωράει στην υλοποίησή του και στη συνέχεια το δοκιμάζει για να δει αν λειτουργεί έτσι όπως το έχει προγραμματίσει. Έπειτα επαναλαμβάνει την ίδια διαδικασία σταδιακά και μεθοδικά και καταλήγει στο επιθυμητό τελικό αποτέλεσμα.

Άλλο ένα συστατικό για την υλοποίηση ενός video game είναι ο developer να δημιουργεί τρόπους που θα τον βοηθήσουν να αποκτήσει την απαραίτητη γνώση που χρειάζεται για να δημιουργήσει το παιχνίδι του με επιτυχία. Μια σημαντική διαδικασία που βοηθάει σε αυτό είναι γνωστή ως "Rumination", η «αναμάσηση» μιας ιδέας (Sylvester, 2013). «Αναμασώντας» μία ιδέα ξανά και ξανά οδηγεί τον εγκέφαλο σε σκέψεις και λύσεις προβλημάτων πολύ πιο εύκολα .

Οι γενικές γνώσεις και οι γνώσεις που έχουμε από παλιά καθώς και η χαλάρωση είναι τα βασικά συστατικά που μπορούν να κάνουν την διαδικασία αυτή πιο εύκολη. Ο εγκέφαλος μπορεί να συνδέσει καινούριες ιδέες με παλιές γνώσεις και να δώσει λύσεις σε προβλήματα ακόμα και υποσυνείδητα. Επιπλέον, όταν είναι χαλαρός μπορεί να σκεφτεί

κάτι ακαριαία που να τον οδηγήσει επίσης σε λύση κάποιου προβλήματος ή στην δημιουργία μίας καινούργιας ιδέας.

Ένας επιπλέον πολύ σημαντικός παράγοντας στην δημιουργία ενός video game είναι η έρευνα. Εκτός από την έρευνα αγοράς που είναι απαραίτητη για την δημιουργία ενός video game που θα είναι ποθητό και αρεστό από το ευρύ κοινό, πολύ σημαντικό ρόλο παίζει και η έρευνα που κάνει ο developer για να διευρύνει τις γνώσεις του. Για παράδειγμα όταν κάποιος μελετάει καινούργιες μεθόδους δημιουργίας και σχεδιασμού παιχνιδιών ή ψάχνει κομμάτια κώδικα ώστε να τα προσθέσει στο παιχνίδι του. Αυτού του είδους η έρευνα πολλές φορές αγνοείται από τους developers, αλλά είναι εξίσου σημαντική με τους υπόλοιπους παράγοντες.

Για την δημιουργία ενός video game απαραίτητη είναι και η χρήση γραφικών. Οτιδήποτε έχει να κάνει με σχέδια, εικόνες, sprites και textures πρέπει ο developer είτε να τα δημιουργήσει μόνος του, είτε να συνεργαστεί με κάποιον γραφίστα.

Ακόμη, η σύλληψη της ιδέας σε συνδυασμό με την καταγραφή της και η ανάλυση του video game παίζουν καθοριστικό ρόλο στην δημιουργία του.

Επίσης, ο developer πρέπει να λάβει υπόψιν του ότι για να βγάλει το παιχνίδι του στην αγορά, θα πρέπει να το περάσει από μια σειρά δοκιμών όχι μόνο από τον ίδιο, αλλά και από άλλους παίχτες οι οποίοι θα μπορούν να εντοπίσουν τυχών λάθη ή τα λεγόμενα "glitches" που μόνο η συνεχόμενη χρήση ενός παιχνιδιού μπορεί να τα φέρει στην επιφάνεια.

Τέλος, πριν μια εταιρεία ή ένας game developer κυκλοφορήσει το παιχνίδι του στην αγορά θα πρέπει να λάβουν υπόψιν τους τα νομικά πλαίσια μέσα στα οποία θα πρέπει να κινηθούν, έτσι ώστε να αποφύγουν τυχών προβλήματα με ανταγωνιστές ή τους πελάτες τους.

2.2 **Απαιτήσεις**

Η απαίτηση είναι συνηθισμένη λέξη στον χώρο των video games. Ο στόχος και τα εμπόδια που σταματάνε τον δημιουργό από το να φτάσει τον στόχο αυτό, είναι αυτά τα οποία καθορίζουν την απαίτηση. Η αναγνώριση των απαιτήσεων που μπορεί να συναντήσει ο game developer και η ομάδα του, μπορεί να τον βοηθήσει σημαντικά στο να τις φέρει σε σημείο όπου πλέον θα είναι υπό έλεγχο. Μερικές από τις πιο σημαντικές απαιτήσεις είναι οι ακόλουθες:

- **Η απαίτηση του χρόνου**: Οι δημιουργοί πρέπει να έχουν στο μυαλό τους κάθε στιγμή, ότι ο χρόνος συνήθως δεν είναι με το μέρος τους. Για παράδειγμα, μία εταιρεία που έχει δημιουργήσει ένα «δυνατό» franchise και σκοπεύει να βγάλει τον επόμενο τίτλο της, τότε θα πρέπει να έχει λάβει υπόψιν της ότι το κοινό της είναι πολύ απαιτητικό και καθόλου διαθέσιμο να περιμένει παραπάνω χρόνο απ' όσο θα ήταν ανεκτό για να αγοράσει και να παίξει αυτόν τον Τίτλο. Επομένως, θα πρέπει

όσο το δυνατόν πιο σύντομα να δημιουργήσει το παιχνίδι της με όσο το δυνατόν λιγότερα προβλήματα έτσι ώστε να μπορέσει να το βγάλει στην αγορά άμεσα και έπειτα να βρίσκεται σε εγρήγορση ώστε να διορθώνει με patches και updates τα τυχόν προβλήματα που θα εμφανιστούν.

- **Η απαίτηση της δεξιοτεχνίας**: Το κοινό τις περισσότερες φορές, επιλέγει παιχνίδια τα οποία εμπεριέχουν την απαίτηση της σκέψης. Δηλαδή, αυτά που έχουν επίπεδα δυσκολίας και που για να επιτευχθεί ο οποιοσδήποτε στόχος μέσα στο παιχνίδι θα πρέπει ο παίχτης να σκεφτεί για να λύσει πολύπλοκα και περίπλοκα πλάνα. Για παράδειγμα, τα παιχνίδια που περιέχουν γρίφους ή τα παιχνίδια στρατηγικής όπου χρειάζεται ο παίχτης να σκεφτεί τον καλύτερο δυνατό τρόπο ώστε να αναπτύξει το περιβάλλον όπου παίζει. Με βάση αυτό λοιπόν, οι δημιουργοί των παιχνιδιών θα πρέπει να είναι διαθέσιμοι και να μπορούν με ευκολία να δημιουργούν τέτοιου επιπέδου δυσκολίες στα παιχνίδια τους και να μπορούν ακόμα και οι ίδιοι να τις αντιμετωπίζουν.

- **Η απαίτηση της αντοχής**: Άλλο ένα στοιχείο των παιχνιδιών που προτιμάει συνήθως το κοινό είναι η αντοχή. Υπάρχουν επίπεδα σε ένα παιχνίδι ή αλλιώς πίστες, οι οποίες χαρακτηρίζονται δύσκολες, λόγω του ότι ο παίχτης πρέπει να αντέξει μια σειρά από δοκιμασίες για μεγάλο χρονικό διάστημα χωρίς να χάσει. Αυτό, επίσης, οδηγεί τους δημιουργούς των παιχνιδιών στο να είναι διαθέσιμοι και εύστοχοι ώστε να μπορούν να δημιουργούν τέτοια επίπεδα μέσα στο παιχνίδι με σχετική ευκολία.

- **Η απαίτηση της μνήμης και της γνώσης**: Σημαντικό ρόλο στην επιλογή παιχνιδιού από το κοινό παίζει και το πόσο απαιτητικό είναι από άποψη γνώσεων και απαίτηση μνήμης. Πολλοί παίχτες, επιλέγουν παιχνίδια που περιέχουν για παράδειγμα, πραγματικά ιστορικά στοιχεία έτσι ώστε να διευρύνουν τις γνώσεις τους. Επιπλέον, επιθυμητά είναι και τα παιχνίδια που εξασκούν την μνήμη του παίχτη. Αυτά δηλαδή που περιέχουν επίπεδα και πίστες που απαιτούν από τον παίχτη να θυμάται κάποια στοιχεία που συνάντησε σε προηγούμενα επίπεδα και πίστες, έτσι ώστε να τα χρησιμοποιήσει στο τρέχον επίπεδο που βρίσκεται. Οι δημιουργοί τέτοιων παιχνιδιών θα πρέπει να κάνουν μεγάλες έρευνες οι οποίες θα είναι αξιόπιστες έτσι ώστε να χρησιμοποιήσουν ιστορικά στοιχεία ή κάποιες συγκεκριμένες γνώσεις οι οποίες θα πρέπει να είναι αληθείς και πραγματικές κατά ένα μεγάλο βαθμό. Έπειτα δημιουργούν το παιχνίδι τους με βάση αυτές τις έρευνες. Επίσης, θα πρέπει να είναι και σε αυτόν τον τομέα, διαθέσιμοι να δημιουργήσουν επίπεδα δυσκολίας μέσα στο παιχνίδι που θα αφορούν την μνήμη του παίχτη, όπως αναφέραμε πιο πάνω.

- **Η απαίτηση της εξυπνάδας και της λογικής**: Είναι παρόμοια με την προηγούμενη. Σε αυτήν την περίπτωση όμως, ο παίχτης αντιμετωπίζει δυσκολίες που αφορούν εξυπνάδα και λογική. Τέτοια παιχνίδια είναι αυτά που περιέχουν γρίφους ή puzzles, τα οποία για να λυθούν θα πρέπει ο παίχτης να βρει κάποιον συνδυασμό ή κάποια στοιχεία που θα τον οδηγήσουν στην λύση. Τέτοια στοιχεία είναι κρυμμένα μέσα

στο ίδιο επίπεδο ή στην πίστα και δεν δίνονται σε προηγούμενα επίπεδα και πίστες όπως αυτά που πρέπει ο παίχτης να θυμάται από πριν. Οι δημιουργοί αυτών των παιχνιδιών, λοιπόν, θα πρέπει να είναι ικανοί να φτιάχνουν τέτοιου είδους δυσκολίες σε κάθε διαφορετικό επίπεδο και σε κάθε διαφορετική πίστα του παιχνιδιού.

- **Η απαίτηση των αναλώσιμων**: Σε πολλά παιχνίδια ο παίχτης πρέπει να χρησιμοποιεί συγκεκριμένα αναλώσιμα ώστε να περάσει το κάθε επίπεδο. Τέτοιου είδους παιχνίδια είναι συνήθως τα action/adventure και survival. Σε αυτά τα παιχνίδια ο παίχτης πρέπει να μαζεύει από το περιβάλλον του παιχνιδιού σφαίρες, φαρμακευτικά, όπλα κ.λπ. που θα τον βοηθήσουν να περάσει το παιχνίδι και να το τερματίσει. Σε κάθε επίπεδο δυσκολίας, υπάρχουν και τα ανάλογα αναλώσιμα, δηλαδή στο εύκολο επίπεδο θα μπορεί ο παίχτης να βρει πολλά από αυτά στο περιβάλλον, ενώ στο δύσκολο επίπεδο θα μπορεί να βρει πολύ λιγότερα. Επομένως οι δημιουργοί τέτοιων παιχνιδιών θα πρέπει να είναι ικανοί να προσθέσουν και να μπορούν να υπολογίσουν πόσα από αυτά τα αναλώσιμα θα χρειαστεί ο παίχτης για να μπορέσει να τερματίσει το παιχνίδι για κάθε επίπεδο δυσκολίας που του διατίθεται.

3 **Σχεδιάζοντας ένα** Video Game

Η σύλληψη της ιδέας ενός παιχνιδιού, ορίζοντας το πώς θα λειτουργήσει, περιγράφοντας τα στοιχεία που το αποτελούν και μεταφέροντας όλες αυτές τις πληροφορίες στην ομάδα όπου θα το σχεδιάσει, όλα αυτά μαζί αποτελούν την διαδικασία του game design (Adams, 2010). Η δουλειά αυτή αφορά τον game designer. Στην βιομηχανία των video games χρειάζεται τουλάχιστον μία ομάδα ανθρώπων αποτελούμενη από 3 έως 20 άτομα για τον σχεδιασμό ενός παιχνιδιού. Και αυτοί αποτελούν αποκλειστικά την ομάδα σχεδιασμού, τίποτε παραπάνω. Με το πέρασμα των χρόνων είναι πιο προσιτό να μπορεί κάποιος να φτιάξει το δικό του παιχνίδι ακόμη και μόνος του.

Πολλοί θεωρούν πως το game design αφορά περισσότερο τον καλλιτεχνικό τομέα (δημιουργία γραφικών, textures κ.λπ.). Άλλοι πιστεύουν ότι το κομμάτι της δημιουργίας και το μηχανικό κομμάτι (engineering) είναι περισσότερο σημαντικό από το προηγούμενο. Στην πραγματικότητα, όμως, και τα δύο είναι εξίσου σημαντικά και απαραίτητα για την δημιουργία ενός video game.

3.1 **Τα περιεχόμενα-κλειδιά στον σχεδιασμό ενός** video game

Για να φτιάξει κάποιος το gameplay και να το προσφέρει στον παίχτη, θα πρέπει να έχει στο νου του πως τα video games χρειάζονται δύο στοιχεία-κλειδιά για να είναι ολοκληρωμένα. Αυτά τα στοιχεία δεν αφορούν το τεχνικό κομμάτι, αλλά το εννοιολογικό. Αυτά είναι το "core mechanics" και το "user interface" (Adams, 2010).

3.1.1 Core Mechanics

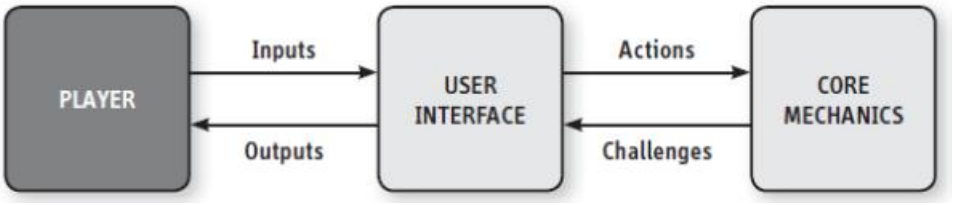
Μια από τις αρμοδιότητες του game designer είναι να μετατρέψει τους γενικούς κανόνες ενός παιχνιδιού σε ένα συμβολικό και μαθηματικό μοντέλο το οποίο μπορεί να εφαρμοστεί με διάφορους αλγόριθμους. Αυτό το μοντέλο αποτελεί τα "core mechanics" του παιχνιδιού και είναι πιο συγκεκριμένο από τους αρχικούς κανόνες. Για παράδειγμα μπορείς να πεις σαν κανόνα ότι ένα σκουλήκι μετακινείται γρηγορότερα από ένα σαλιγκάρι, ενώ το μοντέλο θα σου υπολογίσει ακριβώς την ταχύτητα που έχει το καθένα με ακρίβεια εκατοστού ανά λεπτό.

Τα "core mechanics" είναι η καρδιά κάθε παιχνιδιού επειδή δημιουργούν το gameplay. Ορίζουν τις απαιτήσεις που μπορεί να έχει ένα παιχνίδι, καθώς και τις ενέργειες που μπορεί να κάνει ο παίχτης για να φέρει εις πέρας όλες αυτές τις απαιτήσεις. Επιπλέον καθορίζουν τις επιπτώσεις που θα έχουν οι ενέργειες του παίχτη πάνω στον κόσμο του παιχνιδιού. Ορίζουν τις συνθήκες που πρέπει να υπάρχουν για να επιτευχθούν οι στόχοι μέσα στο παιχνίδι και τι επιπτώσεις θα ακολουθήσουν σε περίπτωση που κερδίσει ή αποτύχει ο παίχτης. Σε ένα video game τα "core mechanics" είναι κρυμμένα από τον παίχτη. Μπορεί μόνο να τα ανακαλύψει σιγά σιγά καθώς προχωράει στο παιχνίδι και ανάλογα με το πόσες

φορές θα προσπαθήσει να το περάσει, θα μάθει και την υπάρχουσα φύση των μηχανισμών μέσα στο παιχνίδι όπου θα τον βοηθήσει να το τερματίσει.

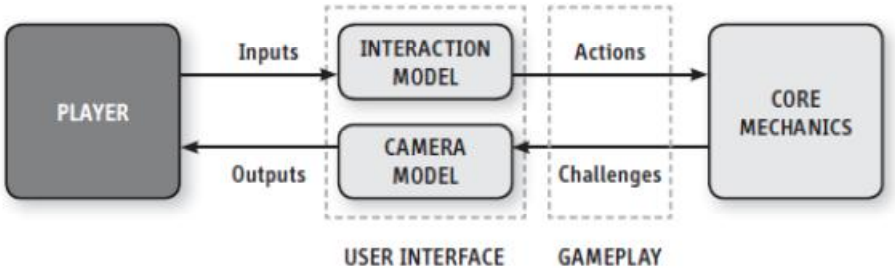
3.1.2 User Interface

Το "user interface" των video games μεσολαβεί ανάμεσα στα "core mechanics" του παιχνιδιού και στον παίχτη (εικόνα 3.1) (Adams, 2010). Παίρνει τις απαιτήσεις οι οποίες δημιουργούνται από τα "core mechanics" (για παράδειγμα, η οδήγηση ενός αγωνιστικού αυτοκινήτου) και τα μετατρέπει σε γραφικά στην οθόνη και σε ήχο στα ηχεία. Επιπλέον, μετατρέπει τα πλήκτρα που πατάει ο παίχτης και τις κινήσεις από το πληκτρολόγιο ή το χειριστήριο σε ενέργειες μέσα στο παιχνίδι. Όσο πιο απαλές και εύκολες είναι οι κινήσεις που πρέπει να ακολουθήσει ο παίχτης, τόσο πιο εύκολα συνδυάζει στον εγκέφαλό του τις κινήσεις αυτές με τα αντίστοιχα πλήκτρα και τότε δεν χρειάζεται να σκέφτεται τι ακριβώς πρέπει να πατήσει για να κάνει την κάθε κίνηση, αλλά απλά πατάει τα πλήκτρα. Το "user interface" αντιλαμβάνεται το πάτημα του κάθε κουμπιού και ενημερώνει τα "core mechanics". Αυτά με την σειρά τους καθορίζουν τι ενέργεια θα κάνει το κουμπί που πατήθηκε και στέλνουν πίσω στο "user interface" μια οδηγία λέγοντάς του να δείξει στον παίχτη το αποτέλεσμα. Όλα αυτά συμβαίνουν σε κλάσματα του δευτερολέπτου.



Εικόνα 3.1 - **Η σχέση μεταξύ των** "core mechanics", **του** "user interface" **και του παίχτη**

Το "user interface" κάνει πολλά περισσότερα από το να δείχνει στην οθόνη τα αποτελέσματα των ενεργειών του παίχτη και να λαμβάνει δεδομένα. Παρουσιάζει επίσης την ιστορία του παιχνιδιού, αν αυτή υπάρχει, και δημιουργεί όλες τις αισθήσεις του κόσμου του παιχνιδιού, δηλαδή όλες τις εικόνες και τους ήχους αυτού του κόσμου, τις δονήσεις του χειριστηρίου αν αυτές είναι διαθέσιμες από το παιχνίδι. Όλη η γραφιστική δουλειά και οι ήχοι του παιχνιδιού αποτελούν κομμάτι του "user interface" , το κομμάτι του φαίνεσθαι. Δύο σημαντικά κομμάτια του "user interface" είναι το "camera model" και το "interaction model" (εικόνα 3.2) (Adams, 2010).



Εικόνα 3.2 - **Το** "Camera Model" **και το** "Interaction model" **του** "User Interface"

3.1.2.1 Interaction Models

Το user interface μετατρέπει τις εντολές που εισχωρεί ο παίχτης από το υλικό (πληκτρολόγιο, χειριστήριο κ.λπ.), σε ενέργειες μέσα στον κόσμο του παιχνιδιού (Adams, 2010). Η σχέση μεταξύ των εντολών που δίνει ο παίχτης και των ενεργειών που δημιουργούνται, προσδιορίζεται από το "Interaction model" του παιχνιδιού. Το μοντέλο αυτό καθορίζει το πώς ο παίχτης θα προβάλλει την θέλησή, τις επιλογές και τις εντολές του στον κόσμο του παιχνιδιού. Πιο συγκεκριμένα, ορίζει τι μπορεί ή όχι να κάνει σε οποιαδήποτε δεδομένη στιγμή ο παίχτης. Τα video games χρησιμοποιούν κάποια συγκεκριμένα interaction models όπως για παράδειγμα το avatar-based model στο οποίο ο παίχτης αντιπροσωπεύεται από έναν χαρακτήρα μέσα στο παιχνίδι και ενεργεί μέσα από αυτόν. Ένα παιχνίδι μπορεί να έχει ένα ή περισσότερα interaction models ανάλογα με το τι ακριβώς συμβαίνει την δεδομένη στιγμή.

3.1.2.2 Camera Models

Όλα τα παιχνίδια περιέχουν μια αναπαράσταση του φυσικού χώρου, τον κόσμο του παιχνιδιού. Είναι δεδομένο, λοιπόν, ότι χρησιμοποιούν γραφικά για να αναπαραστήσουν αυτόν τον κόσμο στον παίχτη. Το user interface θα πρέπει να αναπαράγει αυτόν τον χώρο από μία συγκεκριμένη οπτική γωνία. Οι game designers συνήθως φαντασιώνονται μια υποθετική κάμερα η οποία στοχεύει στον εικονικό χώρο και δημιουργεί μία εικόνα όπου βλέπει ο παίχτης. Το σύστημα που ελέγχει την συμπεριφορά αυτής της κάμερας ονομάζεται camera model (Adams, 2010). Για να ορίσει ο designer αυτό το μοντέλο, θα πρέπει να αποφασίσει το πώς ακριβώς θέλει ο παίχτης να βλέπει τον κόσμο του παιχνιδιού και να ορίσει το σύστημα αυτό μέσα στο design document του.

Οι τύποι των camera models είναι δύο: οι στατικές και οι δυναμικές (Adams, 2010). Το στατικό μοντέλο είναι αυτό που έχουν τα παιχνίδια όταν η προοπτική τους είναι συγκεκριμένη και δεν αλλάζει κατά την διάρκεια του παιχνιδιού. Στο δυναμικό μοντέλο, η κάμερα και η προοπτική μπορεί να αλλάζει ανάλογα με τις κινήσεις που κάνει ο χαρακτήρας μέσα στο παιχνίδι. Αυτό το μοντέλο για να δημιουργηθεί χρειάζεται πολύ δυνατό υλικό (hardware) και έχει περισσότερες απαιτήσεις για να δημιουργηθεί, αλλά μετατρέπει τον κόσμο του παιχνιδιού σε κάτι το οποίο φαίνεται πιο ρεαλιστικό και κινηματογραφικό. Σε περιπτώσεις όπου τα παιχνίδια δεν έχουν κάποιον εικονικό χώρο (virtual space), ο όρος camera model δεν χρησιμοποιείται και πρέπει σε αντικατάσταση αυτού, να υπάρχει μία επεξήγηση της οθόνης που θα εμφανίζεται στο design document.

Τα πιο συνηθισμένα camera models που χρησιμοποιούνται στα παιχνίδια είναι τα πρώτου και τρίτου προσώπου (first person, third person), έτσι ώστε να παρουσιάζεται ο τρισδιάστατος κόσμος και τα top-down, side-scrolling, isometric για να παρουσιάζεται ο δισδιάστατος κόσμος (Adams, 2010).

3.2 **Η δομή ενός** Video Game

Η δομή ενός παιχνιδιού είναι φτιαγμένη από τα "gameplay modes", ένα ζωτικής σημασίας παράγοντα στο game design, τα "shell menus", καθώς και με τις σχέσεις που έχουν μεταξύ

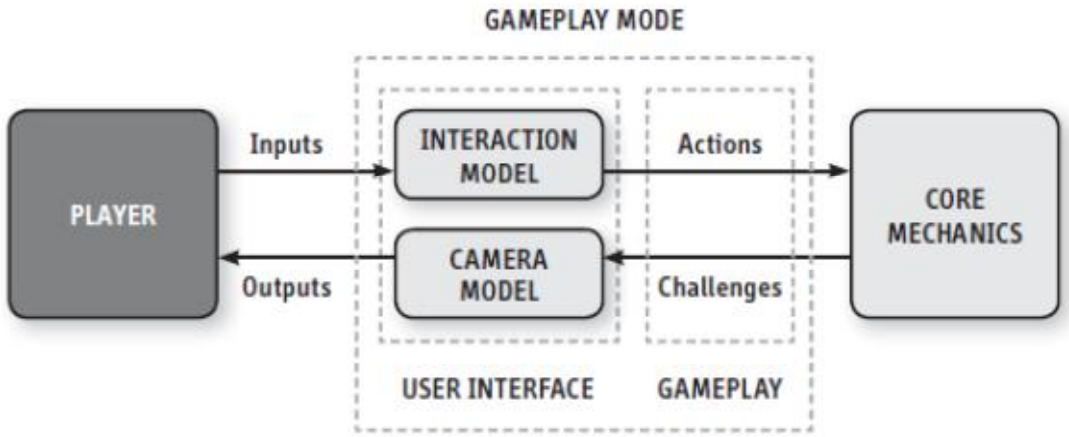
τους (Adams, 2010). Παρακάτω θα δούμε αυτά τα δύο πως έρχονται σε επαφή έτσι ώστε να δημιουργήσουν την δομή.

3.2.1 Gameplay Modes

Για να έχει ένα παιχνίδι συνοχή, οι απαιτήσεις και οι ενέργειες που είναι διαθέσιμες προς τον παίχτη ανά πάσα ώρα και στιγμή, θα πρέπει να είναι συνδεδεμένες μεταξύ τους με μία έννοια. Για παράδειγμα σε μία μάχη σώμα-με-σώμα ο παίχτης θα πρέπει να μπορεί να κινηθεί, να κρατάει το όπλο του και να μπορεί να το χρησιμοποιήσει, να μπορεί να καταναλώσει ένα ειδικό αντικείμενο που του αναπληρώνει την διάρκεια ζωής του (συνήθως αυτό επιφέρει ρίσκο στο να μην προλάβει ο παίχτης να το χρησιμοποιήσει) και ίσως να μπορέσει να το σκάσει ή να παραδοθεί στον αντίπαλο. Δεν θα πρέπει να έχει την δυνατότητα εκείνη την στιγμή να ανοίξει έναν χάρτη του παιχνιδιού ή να ανοίξει κάποιο backpack/inventory και να επεξεργαστεί τα αντικείμενά του ακόμα και αν αυτές τις ενέργειες μπορεί να τις κάνει κατά την διάρκεια του υπόλοιπου παιχνιδιού.

Δεν μπορούν να λειτουργούν πάντα όλες οι απαιτήσεις και οι ενέργειές ενός παιχνιδιού ταυτόχρονα. Ο παίχτης βιώνει μόνο ένα υποσύνολο του gameplay κάθε φορά, αυτό που το παιχνίδι του προσφέρει εκείνη την δεδομένη στιγμή. Επίσης το user interface είναι σχεδιασμένο με τέτοιο τρόπο ώστε να υιοθετεί οποιαδήποτε ενέργεια εκτελείται τη δεδομένη στιγμή, όπως και τα camera και interaction models. Για παράδειγμα, σε ένα παιχνίδι όπου ο χαρακτήρας (avatar) χρησιμοποιεί ένα αυτοκίνητο και ταυτόχρονα πρέπει να πυροβολήσει τους εχθρούς μπροστά του, το gameplay δίνει στον παίχτη την πρόσβαση στο συγκεκριμένο οπτικό πεδίο και στα αντικείμενα που μπορεί να χρησιμοποιήσει εκείνη τη στιγμή, αλλά τα υπόλοιπα μέρη του gameplay που δεν βρίσκονται σε αυτό το οπτικό πεδίο, δεν είναι προσβάσιμα.

Αυτός ο συνδυασμός του διαθέσιμου gameplay και της συνεισφοράς του user interface σε αυτό, κάθε δεδομένη στιγμή μέσα στο παιχνίδι, αποτελούν το λεγόμενο gameplay mode (εικόνα 3.3). Το gameplay mode αποτελείται από συγκεκριμένα κομμάτια και υποσύνολα gameplays ενός παιχνιδιού τα οποία είναι διαθέσιμα ανά πάσα στιγμή μέσα στο παιχνίδι, καθώς και το user interface το οποίο παρουσιάζει αυτά τα κομμάτια και υποσύνολα του gameplay στον παίχτη.



Εικόνα 3.3 - **Το κεντρικό διακεκομμένο κουτί αντιπροσωπεύει το** gameplay mode

Μόνο ένα gameplay mode είναι διαθέσιμο κάθε φορά σε ένα παιχνίδι (Adams, 2010). Όταν αλλάξει σημαντικά το user interface τότε αυτό σημαίνει αυτόματα την αλλαγή από ένα gameplay mode σε ένα άλλο. Τέτοιες αλλαγές στο user interface αλλάζουν την προσοχή του παίχτη από μία ενέργεια σε μία άλλη, κάνοντας τον να σκεφτεί εκ νέου μία διαφορετική δοκιμασία. Επιπλέον, αλλαγή στο mode υπάρχει και όταν το σετ των πλήκτρων που πρέπει να πατήσει ο παίχτης για να κάνει κάποιες ενέργειες, αλλάξει. Για παράδειγμα όταν ο χαρακτήρας μπει σε θέση μάχης, τότε τα διαθέσιμα πλήκτρα και οι συνδυασμοί αυτών είναι διαφορετικοί από αυτούς στο υπόλοιπο gameplay.

3.2.2 **Τα** Shell Menus **και** Shell Screens

Όταν ένας παίχτης επηρεάζει με τις ενέργειές τον κόσμο ενός παιχνιδιού, δηλαδή οποιαδήποτε στιγμή παίζει ένα παιχνίδι, αυτό βρίσκεται σε κάποιο gameplay mode. Υπάρχουν όμως και τα modes όπου ο παίχτης δεν μπορεί να επηρεάσει με τις ενέργειές του τον κόσμο του παιχνιδιού, αλλά μπορεί να κάνει άλλες αλλαγές. Αυτά ονομάζονται Shell menus και τοποθετούνται πριν ή αφού ξεκινήσει να παίζει το παιχνίδι, καθώς και κατά την διάρκεια του παιχνιδιού σε περίπτωση που επιλέξει να το «ανοίξει» ο παίχτης [pause menu] (Adams, 2010). Οι ενέργειες που είναι διαθέσιμες σε αυτά τα menus είναι για παράδειγμα το loading και saving game, οι ρυθμίσεις, όπου επιλέγει ο παίχτης την ένταση του ήχου, την ανάλυση των γραφικών που θα έχει το παιχνίδι κ.λπ.. Σε περιπτώσεις όπου το pause menu επιτρέπει στον παίχτη να αλλάξει κάποια ρύθμιση όπου θα επηρεάσει τον κόσμο του παιχνιδιού όπως για παράδειγμα να αλλάξει το επίπεδο δυσκολίας, τότε αυτό δεν είναι shell menu, αλλά gameplay mode. Επίσης, τα κομμάτια όπου δεν έχουν αλληλεπίδραση με τον παίχτη, όπως η οθόνες τίτλων ή τα credits, ονομάζονται shell screens.

3.3 **Τα στάδια του σχεδιασμού ενός** Video Game

Ανάλογα με το μέγεθος του παιχνιδιού, ακολουθείται και η σχεδιαστική διαδικασία. Σε ένα σχετικά μικρό παιχνίδι, είναι ευκολότερο να οργανώσει ο designer το σχεδιαστικό του κομμάτι και έπειτα να το θέσει σε λειτουργία συνδέοντας το με τον κώδικα. Αν όμως το παιχνίδι είναι μεγάλο, η διαδικασία αυτή γίνεται δυσκολότερη και η κάθε εταιρεία που θέλει να δημοσιεύει ένα τέτοιο παιχνίδι θα πρέπει να είναι οργανωμένη με τέτοιο τρόπο, ώστε να βγει σωστά η δουλειά. Σε μεγάλα παιχνίδια, ο designer και η ομάδα του, θα πρέπει να προσέξουν ένα μεγάλο πρόβλημα που προκύπτει: μερικά κομμάτια του σχεδιασμού δεν μπορούν να αλλάξουν κατά την διάρκεια ή μετέπειτα σε ένα παιχνίδι, από την στιγμή που έχει δημιουργηθεί και συνδεθεί με τον κώδικα. Επομένως, η οργάνωση αυτών των κομματιών θα πρέπει να γίνεται με μεγάλη προσοχή και να υπάρχει πολλαπλός έλεγχος αυτών κατά την διάρκεια της δημιουργίας του παιχνιδιού. Επιπλέον, υπάρχουν τμήματα του παιχνιδιού τα οποία επιλέγονται από την ομάδα πριν ξεκινήσει η δημιουργία και δεν μπορούν να αλλάξουν ποτέ. Αυτά, για παράδειγμα, είναι το είδος του παιχνιδιού, το κοινό στο οποίο θα απευθύνεται, το περιεχόμενό του κ.λπ.. Η διαδικασία σχεδιασμού χωρίζεται σε τρία στάδια:

- Το **αρχικό στάδιο** (concept stage), όπου εδώ αποφασίζει η ομάδα την σειρά που θα εκτελεστούν οι ενέργειες και τα αποτελέσματα αυτής της πράξης δεν αλλάζουν ποτέ.

- Το **στάδιο της επεξεργασίας** (elaboration stage), στο οποίο προστίθενται οι λεπτομέρειες του σχεδιασμού και μέσα από τον έλεγχο και τις πολλαπλές δοκιμές βελτιώνονται οι αποφάσεις που πήρε η ομάδα.

- Το **στάδιο του συντονισμού** (tuning stage), στο οποίο πλέον δεν προστίθενται νέες λειτουργίες, αλλά μπορούν να υπάρξουν μικρές αλλαγές για να τελειοποιηθεί το παιχνίδι.

Κάθε ένα από αυτά τα στάδια περιέχει και τα δικά του διαφορετικά καθήκοντα που πρέπει να ακολουθήσει η ομάδα των designers (Adams, 2010).

3.3.1 **Το Αρχικό Στάδιο** (Concept Stage)

Σε αυτό το στάδιο τις αποφάσεις που θα πάρει η ομάδα των designers θα τις χρησιμοποιεί μέχρι να ολοκληρωθεί το project. Οι αποφάσεις αυτού του σταδίου είναι τόσο καθοριστικές για το παιχνίδι, που αν προσπαθήσουν να τις αλλάξουν κατά την διάρκεια της δημιουργίας του, τότε θα πρέπει να πάει χαμένη όλη η δουλειά που έχουν κάνει μέχρι εκείνη την στιγμή και να ξεκινήσουν, ουσιαστικά, την δημιουργία του παιχνιδιού από την αρχή. Για παράδειγμα, αν επιλεγεί από την αρχή ότι το είδος του παιχνιδιού που θα φτιάξουν είναι ένα third-person action game, δεν μπορούν το μετατρέψουν αργότερα σε ένα first-person-shooting game. Θα πρέπει να το δημιουργήσουν από την αρχή.

Αρχικά, θα πρέπει **να επιλεχθεί το** concept του παιχνιδιού. Η ομάδα, δηλαδή, θα πρέπει να διαλέξει την γενική ιδέα που θα ακολουθήσει το παιχνίδι, με την οποία σκοπεύουν να διασκεδάσουν το κοινό μέσω του gameplay και κατά πόσο αυτή θα είναι μία ακαταμάχητη εμπειρία για τον παίχτη. Πολύ σημαντικό σε αυτό το σημείο είναι να αποφασίσουν τι είδος θα είναι το παιχνίδι που θα δημιουργήσουν.

Έπειτα, θα πρέπει **να καθοριστεί το κοινό** που θα επιλέξει να παίξει αυτό το παιχνίδι. Σε έναν κόσμο όπου κυριαρχείται από την διαφήμιση, πολλοί εκδότες παιχνιδιών αποφασίζουν αρχικά ποιο θα είναι το κοινό τους και στη συνέχεια καθορίζουν οτιδήποτε έχει να κάνει με το παιχνίδι. Όπως και να ‘χει, οι αποφάσεις που παίρνονται σε αυτό το στάδιο είναι πολύ σημαντικές, γιατί θα πρέπει να καταφέρουν τελικά να διασκεδάσουν το κοινό που επιλέχθηκε (target audience).

Ακόμη, θα πρέπει **να καθοριστεί ο ρόλος του παίχτη**. Σε ένα παιχνίδι όπου είναι αφηρημένο (π.χ. Backgammon), ο παίχτης δεν συνδέεται με τον κόσμο του παιχνιδιού και δεν έχει κάποιον συγκεκριμένο ρόλο. Σε παιχνίδια όμως που είναι αντιπροσωπευτικά, ο παίχτης μπαίνει στη θέση του χαρακτήρα του (avatar) και γίνεται ένα με αυτόν. Νιώθει πως είναι ένα με τον κόσμο του παιχνιδιού και συνδέεται η ψυχοσύνθεση του με αυτήν του χαρακτήρα του. Σε τέτοιου είδους παιχνίδια, η ομάδα των designers θα πρέπει να μπορέσει με ευκολία να δημιουργήσει τον ρόλο που θα αντιπροσωπεύσει ο παίχτης μέσω του χαρακτήρα του, και αποτελεί πολύ σημαντικό στάδιο για την δημιουργία ενός παιχνιδιού. Αυτός ο ρόλος θα καθορίσει και το αν τον κοινό τελικά επιλέξει το συγκεκριμένο παιχνίδι.

Τέτοιου είδους παιχνίδια όπου συνδέονται οι παίχτες με τους χαρακτήρες τους μέσα στο παιχνίδι είναι, για παράδειγμα, τα Uncharted, Tomb Raider, Assassin's Creed, Prince of Persia κ.λπ.

Τέλος, αφού αποφασιστεί το ολοκληρωμένο concept του παιχνιδιού, ο ρόλος που θα παίξει ο χαρακτήρας και σε τι κοινό θα απευθύνεται το παιχνίδι, θα πρέπει να αποφασιστεί και ο τρόπος με τον οποίο θα **εκπληρωθούν τα όνειρα** του παίχτη. Κάθε παίχτης που παίζει ένα αντιπροσωπευτικό παιχνίδι θέλει να εκπληρώσει τα όνειρά του. Όνειρα τα οποία περιέχουν στόχους, κατορθώματα, δύναμη, δημιουργία ή απλά να κάνει κάποιες ενέργειες που θα του επιφέρουν συγκεκριμένα συναισθήματα και εμπειρίες. Πρέπει λοιπόν να οριστούν από την ομάδα τα ακόλουθα: η αίσθηση την οποία θα αφήνει το παιχνίδι, οι δοκιμασίες τις οποίες θα αντιμετωπίζει και οι ενέργειες που θα πρέπει να κάνει ο παίχτης. Στα αφηρημένα παιχνίδια δεν υπάρχει όλος αυτός ο κόσμος των συναισθημάτων. Ο παίχτης παίζει απλά και ακολουθεί τους συγκεκριμένους κανόνες που του προσφέρει το παιχνίδι. Ορίζοντας το τι χρειάζεται για να εκπληρώσει ο παίχτης τα όνειρά του μέσα από το παιχνίδι, είναι το πρώτο βήμα που ακολουθεί ο designer στο να ορίσει και το gameplay του.

3.3.2 **Το Στάδιο της Επεξεργασίας** (Elaboration Stage)

Μετά τις αποφάσεις που παίρνονται στο αρχικό στάδιο, η ομάδα προχωράει στο στάδιο της επεξεργασίας. Εδώ οι ιδέες γίνονται πράξεις και το θεωρητικό κομμάτι μετατρέπεται σε πρακτικό. Αρχικά, πρέπει να δημιουργηθεί ένα πρωτότυπο του παιχνιδιού το οποίο θα μπορούν να υποβάλλουν σε έλεγχο όσες φορές χρειαστεί, ώστε να πάρουν το τελικό αποτέλεσμα. Μετά από κάθε έλεγχο και κάθε δοκιμή, το παιχνίδι θα γίνεται καλύτερο και θα διορθώνονται τα πιθανά λάθη που εμφανίζονται σε αυτές τις δοκιμές. Έτσι το παιχνίδι θα ολοκληρωθεί και θα μπορέσει η εταιρεία να το δημοσιεύσει με επιτυχία.

Η πρώτη δουλειά που πρέπει να γίνει σε αυτό το στάδιο είναι να οριστεί το **αρχικό** gameplay mode που θα έχει το παιχνίδι. Αυτό θα είναι το κομμάτι όπου ο παίχτης θα ξοδεύει τον περισσότερο χρόνο του. Σε αυτό το σημείο, δεν χρειάζεται να προστεθεί κάθε λεπτομέρεια του παιχνιδιού, αλλά να δοθεί σημασία στα κύρια τμήματα που θα αποτελούν αυτό το gameplay mode. Αυτά, για παράδειγμα, είναι η προοπτική που θα χρησιμοποιηθεί για να βλέπει ο παίχτης τον κόσμο του παιχνιδιού, το κομμάτι της αλληλεπίδρασης με τον παίχτη, οι δοκιμασίες οι οποίες θα υποβληθούν στον παίχτη και οι ενέργειες με τις οποίες θα μπορεί να ξεπεράσει αυτές τις δοκιμασίες.

Η επόμενη δουλειά της ομάδας είναι να σχεδιάσουν τον **κύριο χαρακτήρα** του παιχνιδιού, τον πρωταγωνιστή. Εκτός από την περίπτωση όπου το παιχνίδι θα είναι first-person (όπου δεν υπάρχει συγκεκριμένο avatar), η ομάδα θα πρέπει να ξοδέψει πολύ χρόνο ώστε να δημιουργήσει τον τέλειο χαρακτήρα για τα δεδομένα του παιχνιδιού τους. Ο παίχτης θα ξοδεύει πολύ χρόνο με τον χαρακτήρα του οπότε θα πρέπει να του δοθούν με προσοχή τα χαρακτηριστικά που θα τον κάνουν να έχει μία συγκεκριμένη προσωπικότητα με την οποία συνήθως ταυτίζεται και ο παίχτης. Τέτοια χαρακτηριστικά είναι για παράδειγμα, η συμπεριφορά που θα έχει, τα όρια που μπορεί να φτάσει, τα συναισθήματα του προσώπου και η γλώσσα του σώματος, το λεξιλόγιο που θα χρησιμοποιεί κ.λπ..

Έπειτα θα πρέπει να ορίσουν τον **κόσμο του παιχνιδιού**. Σε περίπτωση όπου είναι ένας πραγματικός κόσμος, όπως για παράδειγμα μια προσομοίωση πτήσης, η ομάδα θα

μπορέσει να πάρει φωτογραφίες και χάρτες από τον πραγματικό κόσμο και να τις τοποθετήσει στο παιχνίδι. Στην περίπτωση, όμως, που ο κόσμος αυτός είναι αποτέλεσμα φαντασίας, θα πρέπει να σχεδιαστούν οι χάρτες και οι περιοχές (terrains) από την αρχή καθώς και οτιδήποτε άλλο εμπεριέχει ένας τέτοιος κόσμος. Μόνο έτσι θα είναι ολοκληρωμένο το gameplay.

Ένα ακόμη βήμα για την ομάδα των designers είναι να **σχεδιάσουν τα** core mechanics του παιχνιδιού. Εφόσον επιλέχθηκε το κύριο gameplay mode και όλες οι δοκιμασίες και απαιτήσεις που θα έχει το παιχνίδι, μπορούν να ξεκινήσουν να σκέφτονται το πώς μπορούν τα core mechanics να υλοποιήσουν όλα αυτά. Αν για παράδειγμα, ο παίχτης πρέπει να μαζέψει υλικά για τον χαρακτήρα του ώστε να μπορέσει να φτιάξει ένα αντικείμενο από αυτά, θα πρέπει να ορίσουν οι designers που ακριβώς μπορεί να βρει αυτά τα υλικά. Αυτή είναι δουλειά των core mechanics, όπως είδαμε και παραπάνω.

Επιπλέον, θα πρέπει να σκεφτεί η ομάδα αν της **αρκεί το ένα** gameplay mode ή αν θα χρειαστεί και περισσότερα για το παιχνίδι της. Να αποφασιστούν οι προοπτικές που θα έχει το κάθε ένα, οι αλληλεπιδράσεις με τον παίχτη και τα gameplays του καθενός. Αυτό το βήμα μπορεί να οριστεί και νωρίτερα καθώς δημιουργείται το βασικό gameplay mode ή κατά την διάρκεια της δημιουργίας των core mechanics.

Σε αυτό το σημείο, ή και νωρίτερα μερικές φορές, ξεκινάει η δημιουργία του πρώτου λειτουργικού **επιπέδου του παιχνιδιού** (game level). Έτσι μπορεί να ξεκινήσει και ο πρώτος έλεγχος και οι δοκιμές του παιχνιδιού, ώστε να βελτιωθεί κάθε τυχών πρόβλημα ή λάθος. Στη συνέχεια δημιουργούνται και τα επόμενα επίπεδα.

Έπειτα, έρχεται η στιγμή όπου θα πρέπει να γραφτεί **η ιστορία του παιχνιδιού**. Η ιστορία ενός παιχνιδιού είναι ένας από τους κύριους λόγους που επιλέγει το κοινό ένα παιχνίδι. Κατά πόσο του κινεί το ενδιαφέρον και κατά πόσο τον δελεάζει ώστε να επιλέξει το συγκεκριμένο παιχνίδι. Η ιστορία του παιχνιδιού μπορεί να είναι ενσωματωμένη με κώδικα στο παιχνίδι, ή να εμφανίζεται και να δημιουργείται κατά την διάρκεια του παιχνιδιού από τα core mechanics. Μπορεί να εξαρτάται από τις ενέργειες του παίχτη ή μπορεί να είναι συγκεκριμένη, γραμμική από την αρχή του παιχνιδιού μέχρι το τέλος του.

Τέλος, θα πρέπει αυτό το πρωτότυπο να «**χτιστεί**», **να δοκιμαστεί και να επαναληφθεί** όλη αυτή η διαδικασία αρκετές φορές, έτσι ώστε να καταλήξουν στο τελικό παιχνίδι το οποίο θα λειτουργεί και θα έχει το αρχικό επιθυμητό αποτέλεσμα. Αν δεν γίνει αυτό το στάδιο πολλές φορές από την ομάδα των σχεδιαστών και των προγραμματιστών, τότε δεν θα έχουν το επιθυμητό αποτέλεσμα και το παιχνίδι που θα δημοσιεύσουν, πιθανόν, θα έχει αρκετά προβλήματα και λάθη τα οποία θα δυσαρεστήσουν πολύ το κοινό τους.

3.3.3 **Το Στάδιο Του Συντονισμού** (Tuning Stage)

Σε αυτό το στάδιο η ομάδα των designers σταματάει να παίρνει αποφάσεις και να προσθέτει ιδέες. Είναι η στιγμή όπου «κλειδώνουν» τις ήδη υπάρχουσες και προχωράνε με αυτές. Δεν είναι ποτέ ακριβές γνωστό πότε πρέπει να μπουν σε αυτό το στάδιο. Συνήθως αποφασίζεται με βάση το αρχικό τους πρόγραμμα. Το ιδανικό φυσικά είναι να επιλεγεί κατά την διάρκεια, καθώς θα βρίσκονται πλέον οι σχεδιαστές σε σημείο όπου μπορούν να εισέλθουν σε αυτό το στάδιο χωρίς να γίνει κάποια βιαστική παράλειψη ζωτικών σημείων

του παιχνιδιού. Μετά από αυτό, δεν μπορούν να προσθέσουν κάτι καινούργιο όταν έχουν φτάσει στο σημείο του ελέγχου και της διόρθωσης των σφαλμάτων και λαθών του κώδικα. Ωστόσο, μπορεί η ομάδα να κάνει μικρές αλλαγές στα επίπεδα του παιχνιδιού ή στα core mechanics, αρκεί να μην προσθέσει καινούργιες λειτουργίες που θα μπορεί να προσφέρει το παιχνίδι. Αυτό το στάδιο, λοιπόν, πρέπει να επιλεχθεί προσεκτικά, διότι αυτό θα κάνει την διαφορά στο παιχνίδι. Από αυτό το στάδιο θα φανεί αν το παιχνίδι είναι απλά καλό ή πραγματικά τέλειο.

Εικόνα 3.4 - **Διαδικασίες Σχεδιασμού**

3.4 Οι Ρόλοι της Σχεδιαστικής Ομάδας

Ένα μεγάλο video game συνήθως σχεδιάζεται από μία ομάδα ανθρώπων. Οι εταιρείες των παιχνιδιών, ορίζουν τους ρόλους των εργαζομένων τους, τις ομάδες και τι εργασίες θα πάρει η κάθε μία από αυτές, ανάλογα με τις ικανότητες τους και τις ανάγκες του project. Παρ' όλα αυτά, υπάρχουν κάποιοι ρόλοι οι οποίοι πλέον είναι δεδομένοι για κάθε εταιρεία παιχνιδιών. Αυτοί είναι οι ακόλουθοι:

- Lead Designer: Αυτός επιβλέπει όλη την διαδικασία του σχεδιασμού ενός παιχνιδιού και είναι υπεύθυνος για να καθορίσει πότε θα ολοκληρωθεί το παιχνίδι. Αυτός είναι ο «φύλακας του οράματος». Επιπλέον, ενθουσιάζει με την ιδέα του παιχνιδιού, τόσο τους υπαλλήλους της εταιρίας, όσο και όλους αυτούς που είναι εκτός και συνήθως είναι αυτός που θα καλέσουν για οποιαδήποτε ομιλία που αφορά το project. Η δουλειά του lead designer δεν αφορά πάντα την δημιουργικότητα, αλλά ως αρχηγός της ομάδας, ο κύριος ρόλος του είναι να ελέγχει και να επιβλέπει την πρόοδο της διαδικασίας του σχεδιασμού, καθώς και να ελέγχει ότι όλη η υπόλοιπη ομάδα κάνει σωστά την δουλειά της. Κάθε project έχει μόνο ένα lead designer.

- Level Designer/World Builder: Αυτοί παίρνουν τα βασικά συστατικά του παιχνιδιού που τους παρέχει η υπόλοιπη ομάδα, όπως το user interface, τα core mechanics και το gameplay, και σχεδιάζουν το κάθε ξεχωριστό επίπεδο (level) που θα έχει το παιχνίδι και από τα οποία θα περνάει ο παίχτης κατά την διάρκειά του. Παλιότερα

ο ρόλος του level designer θεωρούνταν κατώτερος από τους υπόλοιπους ρόλους του game design, αλλά στις μέρες μας, οι level designers είναι απαραίτητο σε πολλές περιπτώσεις να είναι σε θέση να μπορούν να σχεδιάζουν τρισδιάστατα (3D) μοντέλα, καθώς και να προγραμματίζουν με γλώσσες προγραμματισμού. Έτσι, ο ρόλος του level designer πλέον είναι πολύ σημαντικός και χρειάζεται να διαθέτει συγκεκριμένες δεξιοτεχνίες ώστε να μπορέσει να ανταπεξέλθει στα καθήκοντά του. Συνήθως, ένα project έχει περισσότερους από έναν level designer οι οποίοι λογοδοτούν σε κάποιον lead level designer ή στον lead designer.

- User Interface Designer: Αν ένα project εμπεριέχει τον ρόλο του user interface designer ως ξεχωριστό ρόλο, αυτός αναθέτεται σε έναν ή περισσότερους σχεδιαστές οι οποίοι είναι υπεύθυνοι για την δημιουργία του layout της οθόνης για τα διαφορετικά gameplay modes του παιχνιδιού και για τον ορισμό των λειτουργιών που θα έχουν τα δεδομένα που προέρχονται από τις συσκευές εισόδου (π.χ. πληκτρολόγιο). Σε μεγάλα και πολύπλοκα project ο ρόλος αυτός γίνεται δουλειά full-time και χρειάζεται έναν πολύ καλό user interface designer, έτσι ώστε το τελικό παιχνίδι να βγει σωστά. Στην περίπτωση της δημιουργίας κακού user interface το παιχνίδι μπορεί να καταστραφεί. Μεγάλες εταιρείες, για να αποφύγουν αυτό ακριβώς, απευθύνονται σε ειδικούς από άλλες εταιρείες λογισμικού, ώστε να τους βοηθήσουν στην διαδικασία του ελέγχου και της βελτιστοποίησης των interfaces που έχουν σχεδιάσει.

- Writer: Οι writers είναι υπεύθυνοι στο να συγγράψουν ένα ρεαλιστικό ή φανταστικό περιεχόμενο για το παιχνίδι που θα σχεδιάσει η ομάδα. Αυτό είναι το εισαγωγικό κομμάτι, η ιστορία του παιχνιδιού, οι διάλογοι, τα cut-scenes (είναι τα video clips που συμβάλλουν στην εξέλιξη της ιστορίας του παιχνιδιού και στα οποία δεν υπάρχει αλληλεπίδραση με τον παίχτη, απλά τα παρακολουθεί) κ.λπ.. Γενικά, ο ρόλος του writer περιορίζεται στο συγγραφικό κομμάτι και δεν εμπλέκεται με την καταγραφή του τεχνικού κομματιού. Αυτή είναι δουλειά των game designers. Λίγα παιχνίδια απαιτούν ο ρόλος του writer να είναι συνεχής. Η δουλειά αυτή συνήθως αναθέτεται σε κάποιον ελεύθερο επαγγελματία εκτός εταιρείας ή την αναλαμβάνει κάποιος άλλος από την ομάδα των designers.

Υπάρχουν δύο ακόμα ρόλοι οι οποίοι είναι πολύ σημαντικοί για την δημιουργία ενός παιχνιδιού, αλλά αυτοί, συνήθως, δεν λογοδοτούν στον lead designer. Οι υπόλοιπες ομάδες των designers, όμως, θα πρέπει να συνεργάζονται με αυτούς για μεγάλα χρονικά διαστήματα κατά την διάρκεια της δημιουργίας του project. Αυτοί είναι:

- Art Director: Ο art director ή αλλιώς lead artist διευθύνει την παραγωγή όλων των εικονικών αντικειμένων ενός παιχνιδιού, δηλαδή τα models, τα γραφικά, τα textures, τα sprites, τα animations, τα κομμάτια του user interface κ.λπ.. Επιπλέον, ο art director παίζει μεγάλο ρόλο στην δημιουργία και την ενδυνάμωση των γραφιστικών στοιχείων του παιχνιδιού, καθώς και ολόκληρου του στυλ που διαθέτει το παιχνίδι. Συνήθως, ο ρόλος αυτός είναι ισάξιος με τον ρόλο του Lead

designer και πρέπει οι δύο τους να έχουν μία πολύ καλή συνεργασία και να συμμερίζονται τους στόχους του project.

- Audio Director: Όπως και ο art director, έτσι και ο audio director επιβλέπει οτιδήποτε έχει να κάνει με το ηχητικό κομμάτι ενός παιχνιδιού. Δηλαδή, την μουσική, τους ήχους περιβάλλοντος, τα effects, τους διαλόγους και τις αφηγήσεις. Ο ήχος είναι ένα πολύ σημαντικό κομμάτι σε ένα παιχνίδι, διότι δημιουργεί τον κόσμο των συναισθημάτων που θα λαμβάνει ο παίχτης, καθώς και θα επηρεάζει την διάθεσή του. Ο audio director συνεργάζεται με τον lead designer έτσι ώστε να καθορίσουν τι είδη ήχων χρειάζεται να παραχθούν για το συγκεκριμένο project.

3.5 **Το** Design Document **Ενός Παιχνιδιού**

Κομμάτι της δουλειάς των game designers είναι να δημιουργούν μία σειρά από έγγραφα με τα οποία θα επεξηγούν τα πλάνα τους στα υπόλοιπα μέλη της ομάδας. Ποια είναι αυτά τα έγγραφα και τι αφορούν εξαρτάται από τον καθένα designer και από το κάθε διαφορετικό project που αναλαμβάνουν, αλλά συνήθως ακολουθούν ένα συγκεκριμένο διάγραμμα.

Είναι σημαντικό στα μεγάλα και πολύπλοκα παιχνίδια να καταγράφεται σε έγγραφα οποιαδήποτε λεπτομέρεια σχεδιασμού χρειάζεται το παιχνίδι για να δημιουργηθεί. Με αυτόν τον τρόπο αποφεύγονται σημαντικά λάθη στον σχεδιασμό. Επίσης, είναι ευκολότερο να καταλάβει η κάθε ομάδα τι ζητάει ο lead designer και να υπάρξει καλύτερη συνεννόηση μεταξύ τους. Επιπλέον, τα design documents βοηθάνε στο να μην παραληφθεί κάποιο στοιχείο του παιχνιδιού. Αν για παράδειγμα ένας σχεδιαστής έχει ξεχάσει να καταγράψει ένα κομμάτι που αφορά ένα επίπεδο μέσα στο παιχνίδι, τότε αυτός που θα διαβάζει και θα ακολουθεί το design document θα το προσέξει πιο εύκολα και έτσι θα ζητήσει να συμπληρωθεί, πριν ξεκινήσει η διαδικασία του σχεδιασμού του παιχνιδιού. Σε κάθε άλλη περίπτωση, θα υπήρχε λάθος στον σχεδιασμό του παιχνιδιού και σαφώς θα ήταν δυσκολότερο να το διορθώσουν σε αυτό το σημείο.

Υπάρχουν διάφορα είδη design documents τα οποία μπορεί να γράψει ένας game designer. Αυτά είναι τα λεγόμενα high concept, game treatment, character design, world design, story ή level progression documents, καθώς και τα flowboard και game script.

Τα high concept και game treatment documents χρησιμεύουν κυρίως ως εργαλεία πώλησης του παιχνιδιού και είναι σχεδιασμένα με τέτοιο τρόπο ώστε να εξηγούν το concept του παιχνιδιού σε κάποιον χρηματοδότη ή σε μία εκδοτική εταιρεία. Είναι συνήθως γραμμένα σε κάποιο πρόγραμμα δημιουργίας εγγράφων όπως είναι το Microsoft Word και παρουσιάζονται εκτυπωμένα στον ενδιαφερόμενο. Τα υπόλοιπα documents γράφονται επίσης σε κάποιο πρόγραμμα δημιουργίας εγγράφων, αλλά δημοσιεύονται σε κάποιο εσωτερικό και ασφαλές site της εταιρείας το οποίο είναι προσβάσιμο μόνο από τα μέλη της. Εκεί μπορεί το κάθε μέλος να διορθώσει ή να ανεβάσει κάποιο ανανεωμένο αρχείο, καθώς και να το αποθηκεύσει.

Παρακάτω αναφέρονται το κάθε ένα από τα design documents ξεχωριστά.

- High Concept Document.
  Το high concept document δεν αφορά τον σχεδιασμό ενός παιχνιδιού. Έτσι όπως ένα βιογραφικό είναι σχεδιασμένο για να σε βοηθήσει να βρεις δουλειά, έτσι και ο σκοπός αυτού του εγγράφου είναι να προσελκύσει την προσοχή κάποιου χρηματοδότη ή εκδότη παιχνιδιών. Μέσα σε αυτό εμπεριέχονται οι ιδέες-κλειδιά που κάνουν το project να ξεχωρίζει. Όπως και ένα βιογραφικό, έτσι και αυτό θα πρέπει να είναι σύντομο και όχι μεγαλύτερο από τέσσερις σελίδες. Ο αναγνώστης αυτού, θα πρέπει να μπορεί να το διαβάσει και να το καταλάβει μέσα σε λίγα μόνο λεπτά. Μόνο τότε θεωρείται ότι είναι επιτυχημένο.

- Game Treatment Document.
  Το game treatment document παρουσιάζει το παιχνίδι με ένα γενικό σχεδιάγραμμα σε κάποιον που ενδιαφέρεται ήδη και θέλει να μάθει περισσότερα γι' αυτό. Είναι σχεδιασμένο με τέτοιο τρόπο ώστε να δημιουργεί αρχικά μία περιέργεια για το παιχνίδι και έπειτα να κεντρίζει το ενδιαφέρον όποιου το διαβάζει, έτσι ώστε να θέλει να μάθει περισσότερα γι' αυτό το παιχνίδι. Συνήθως, το game treatment document, το δίνει η ομάδα στον πιθανό εκδότη έτσι ώστε να μπορέσει να το μελετήσει αργότερα και να αποφασίσει αν τελικά θα χρηματοδοτήσει ή εκδώσει το παιχνίδι.
  Το game treatment document είναι ένα απλό έγγραφο το οποίο παρουσιάζει τις βασικές ιδέες του παιχνιδιού. Οπότε ένας καλός τρόπος για να μπορέσει η ομάδα να δημιουργήσει αυτό το έγγραφο είναι να φανταστεί πως δημιουργεί ένα website για να πουλήσει το παιχνίδι της και μετά να το εμπλουτίσει με περισσότερες λεπτομέρειες.

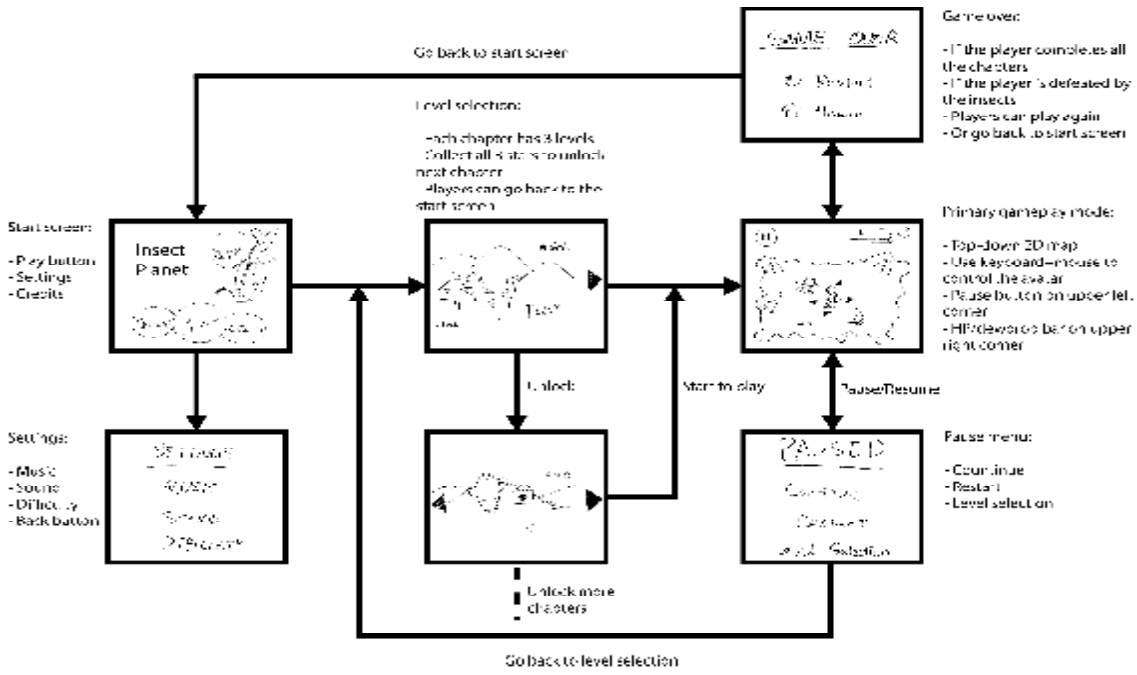
- Character Design Document.
  Αυτό το έγγραφο είναι ειδικά σχεδιασμένο έτσι ώστε να παρουσιάζει έναν χαρακτήρα που θα έχει το παιχνίδι (avatar). Κύριος σκοπός του είναι να παρουσιάσει όλο το σκηνικό παρουσιαστικό του χαρακτήρα καθώς και όλες τις πιθανές κινήσεις που μπορεί να κάνει (move-set), δηλαδή μία λίστα από τα animations που μπορεί να κάνει είτε μόνος του είτε με την βοήθεια του παίχτη. Επίσης, θα πρέπει να εμπεριέχει αρκετά από τα εικαστικά κομμάτια του χαρακτήρα (concept art) όπως και μία λίστα από τις διαφορετικές εκφράσεις του προσώπου του. Τέλος, απαραίτητο είναι να περιέχει πληροφορίες σχετικά με την προσωπικότητα και την ιστορία αυτού του χαρακτήρα, όπως για παράδειγμα τι του αρέσει και τι όχι, τις δυνατότητές του, τις αξίες του, τα ευάλωτα σημεία του κ.λπ.

- World Design Document.
  Το world design document περιέχει όλα τα στοιχεία που αφορούν το περιβάλλον (κτήρια κ.λπ.), το εικαστικό κομμάτι τους και τους ήχους που περιγράφουν τον κόσμο του παιχνιδιού. Δεν είναι μία ακριβής λίστα που καταγράφει τα πάντα μέσα στο παιχνίδι, αλλά καταγράφει τις πληροφορίες του Background που αφορούν όλα αυτά τα αντικείμενα που περιέχει ένας κόσμος. Για παράδειγμα, μπορεί να

εμπεριέχει τον χάρτη των περιοχών ενός landscape (εικονική αναπαράσταση εξωτερικού χώρου) ή cityscape (εικονική αναπαράσταση πόλης). Δεν χρειάζεται η καταγραφή όλων των λεπτομερειών, απλά μία γενική απεικόνιση έτσι ώστε οι level designers να χρησιμοποιήσουν αυτές τις πληροφορίες για να δημιουργήσουν το ακριβές περιεχόμενο. Σημαντικό περιεχόμενο αυτού του εγγράφου είναι και η λίστα των ήχων περιβάλλοντος και τις πηγές τους έτσι ώστε οι audio designers να μπορέσουν να τους δημιουργήσουν στα κατάλληλα σημεία. Επιπλέον, το world design document θα πρέπει να περιέχει όλες τις αισθήσεις που προκαλεί ο κόσμος του παιχνιδιού, την αισθητική του και τον συναισθηματικό του τόνο.

- Flowboard.

  Ένα flowboard είναι κάτι ανάμεσα στο flowchart και το storyboard. Τα storyboards είναι έγγραφα τα οποία χρησιμοποιούν οι δημιουργοί ταινιών έτσι ώστε να καταγράψουν μία σειρά από λήψεις. Τα flowcharts είναι έγγραφα τα οποία χρησιμοποιούν οι προγραμματιστές για να καταγράψουν τους αλγόριθμους τους. Έτσι, το flowboard συνδυάζει την ιδεολογία αυτών των δύο και παρουσιάζει την δομή ενός video game. Στο flowboard καταγράφονται όλα τα gameplay modes και τα shell menus, τα ονόματα αυτών, κάποια πρόχειρα σχέδια που παρουσιάζουν την οθόνη που θα έχει το καθένα από αυτά, την προοπτική της κάμερας, τα κομμάτια του user interface, καθώς και τα αντικείμενα που εμπεριέχονται στο μενού και τα δεδομένα που μπορεί ο παίχτης να εισάγει (inputs) και τι κάνουν αυτά. Επιπλέον, περιέχει όλες τις δοκιμασίες, τις κινήσεις και τις ενέργειες που είναι διαθέσιμες για τον χαρακτήρα σε κάθε mode. Τέλος, εμπεριέχει όλες τις συνθήκες και τις σχέσεις που πρέπει να πραγματοποιηθούν, έτσι ώστε το παιχνίδι να περάσει από το ένα mode ή menu στο άλλο (εικόνα 3.5).



Εικόνα 3.5 - **Παράδειγμα** Flowboard (**Πηγή**: https://junpengiat410.wordpress.com/page/2/)

- Story **και** Level Progression Document.
  Σε αυτό το έγγραφο καταγράφεται όλη η ιστορία του παιχνιδιού, καθώς και το πώς ένα επίπεδο εξελίσσεται και περνάει στο επόμενο. Αν το παιχνίδι είναι απλό και έχει μόνο ένα επίπεδο, τότε δεν χρειάζεται τέτοιο έγγραφο. Αν όμως έχει πάνω από ένα επίπεδο ή ο παίχτης αντιλαμβάνεται σημαντικές αλλαγές να γίνονται καθώς παίζει το παιχνίδι, τότε ένα τέτοιο έγγραφο είναι απαραίτητο. Εδώ καταγράφεται ένα γενικό σχεδιάγραμμα που αφορά τις εμπειρίες του παίχτη από την στιγμή που θα ξεκινήσει το παιχνίδι μέχρι το τέλος. Επίσης, εμπεριέχει όλες τις πιθανές ενέργειες που θα γίνουν αν το παιχνίδι είναι τέτοιο που πρέπει ο παίχτης να επιλέξει κάτι να εκτελεστεί. Επιπλέον, σε αυτό το σημείο καταγράφεται και ο τρόπος με τον οποίο ο παίχτης θα βιώσει την ιστορία του παιχνιδιού, αν για παράδειγμα αυτή βγαίνει μέσα από cut-scenes, αποστολές που πρέπει να εκτελέσει ο παίχτης, διαλόγους ή άλλες αφηγήσεις. Η ιστορία ενός παιχνιδιού είναι τελείως διαφορετικό από την δομή του. Μπορεί σε ένα gameplay mode να υπάρχει κομμάτι της ιστορίας και σε ένα άλλο όχι.

- **Το** Game Script.
  Το game script καταγράφει έναν πολύ σημαντικό τομέα που τα υπόλοιπα documents δεν αναφέρουν: τους κανόνες και τα core mechanics του παιχνιδιού. Καταγράφει και καθορίζει με κάθε λεπτομέρεια όλους τους κανόνες του παιχνιδιού σε τέτοιο βαθμό, που θεωρητικά μπορείς να το παίξεις πριν καν μπει σε υπολογιστή. Δηλαδή, μπορείς να πάρεις αυτά τα έγγραφα και να ανακαλύψεις ακολουθώντας τα, πως μπορείς να παίξεις το παιχνίδι. Με αυτόν τον τρόπο, πολλές φορές εντοπίζονται και κάποια λάθη τα οποία λύνονται πριν αρχίσει η επεξεργασία στον υπολογιστή. Το game script δεν περιέχει τον τεχνικό σχεδιασμό (technical design) του παιχνιδιού, αλλά καταγράφει την πλατφόρμα στην οποία θα παίζεται το παιχνίδι, για παράδειγμα ηλεκτρονικός υπολογιστής (PC), καθώς και τα ελάχιστα χαρακτηριστικά που πρέπει να έχει έτσι ώστε να μπορεί να το «σηκώσει». Αν υπάρχει technical design document, συνήθως γράφεται από τον lead programmer ή τον technical director και είναι βασισμένο στο game script (Adams, 2010).

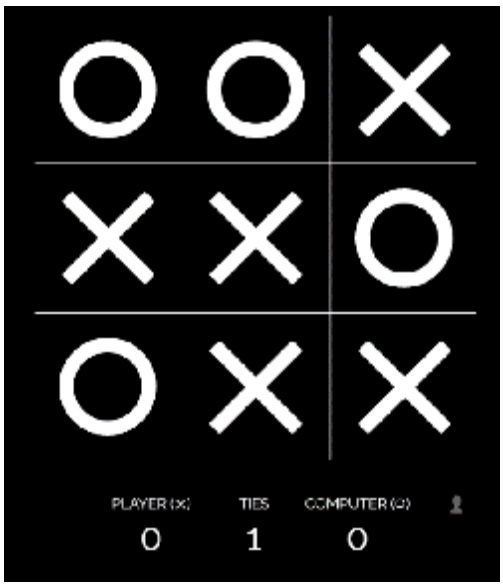
4 **Ο Κόσμος του** Video Game (Game Worlds)

Πολλά παιχνίδια διασκεδάζουν το κοινό τους μέσω του gameplay τους. Πολλά όμως τους διασκεδάζουν με το να τους ταξιδεύουν σε ένα φανταστικό μέρος, τον κόσμο του παιχνιδιού (game world). Ειδικά στα single-player games, ο παίχτης έχει την αίσθηση ότι ζει μέσα στον κόσμο του παιχνιδιού και αλληλεπιδρά με αυτόν. Ο κόσμος του παιχνιδιού περιγράφεται από κάποιες διαστάσεις (dimensions): την φυσική (physical), την χρονική (temporal), την περιβαλλοντική (environmental), την συναισθηματική (emotional) και την ηθική (ethical) (Adams, 2010).

4.1 **Τι είναι ο κόσμος ενός** video game

Ο κόσμος ενός video game αποτελεί ένα τεχνητό σύμπαν, ένα φανταστικό κόσμο στον οποίο συμβαίνουν τα γεγονότα του παιχνιδιού. Όταν ο παίχτης ξεκινάει να παίζει ένα παιχνίδι, τότε αμέσως προσποιείται και νιώθει ότι ο ίδιος βρίσκεται στον κόσμο αυτό.

Δεν έχουν όλα τα παιχνίδια τέτοιο κόσμο. Ένα παιχνίδι ποδοσφαίρου για παράδειγμα, αντιπροσωπεύει έναν αληθινό κόσμο και όχι φανταστικό. Επίσης, τα παιχνίδια όπως είναι η τρίλιζα (tic tac toe), είναι επιτραπέζια παιχνίδια και δεν περιέχουν κάποιο κόσμο, οπότε δεν υπάρχει το φανταστικό στοιχείο καθόλου (εικόνα 4.1). Από την άλλη, υπάρχουν κάποια αφηρημένα παιχνίδια που μπορεί να έχουν μία μικρή έννοια του κόσμου. Για παράδειγμα το σκάκι, δεν έχει καθόλου κάποιο φανταστικό στοιχείο, όμως με μία έννοια σε ταξιδεύει στον μεσαιωνικό κόσμο όπου υπήρχαν οι βασιλιάδες και οι βασίλισσες, οι ιππότες και οι στρατιώτες (εικόνα 4.2).



Εικόνα 4.1 - **Παράδειγμα παιχνιδιού** tic-tac-toe

Εικόνα 4.2 - **Παράδειγμα παιχνιδιού σκάκι**

Τα περισσότερα παιχνίδια παρουσιάζουν τον κόσμο τους με εικόνες, ήχους, το γραφιστικό κομμάτι, τα animations, την μουσική και τα ηχητικά εφέ. Από την άλλη, δεν περιέχουν όλα τα παιχνίδια οπτικό και ακουστικό στοιχείο. Στα λεγόμενα text adventure παιχνίδια, ο παίχτης δημιουργεί όλες τις εικόνες και τους ήχους του κόσμου στην φαντασία του, καθώς διαβάζει τις περιγραφές και τα κείμενα στην οθόνη.

Ο κόσμος ενός παιχνιδιού είναι κάτι πολύ περισσότερο από το σύνολο των εικόνων και των ήχων που τον αποτελούν. Ένας τέτοιος κόσμος μπορεί να περιέχει την δική του κουλτούρα, μια συγκεκριμένη αισθητική, κάποιες ηθικές αξίες και άλλα τέτοια χαρακτηριστικά. Επίσης, ο κόσμος ενός παιχνιδιού έχει άμεση σχέση με την πραγματικότητα είτε αν αυτό είναι ένα αφηρημένο παιχνίδι το οποίο σχετίζεται ελάχιστα με τον πραγματικό κόσμο, είτε αν αυτό είναι ένα αντιπροσωπευτικό παιχνίδι το οποίο είναι όσο το δυνατών περισσότερο όμοιο με τον πραγματικό κόσμο.

4.2 **Ο Σκοπός ενός** Game World

Ένας από τους κύριους στόχους του game world είναι να διασκεδάζει με τον δικό του ξεχωριστό τρόπο το κοινό. Να τον μεταφέρει σε έναν κόσμο φανταστικό, όπου θα μπορεί να εξερευνήσει και να αλληλεπιδράσει με το περιβάλλον του.

Όσο περισσότερο ένας παίχτης καταλαβαίνει τα core mechanics ενός παιχνιδιού, τόσο λιγότερη σημασία έχει ο κόσμος του. Ωστόσο, όσο γίνεται καλύτερος και καταλαβαίνει περισσότερο τα core mechanics, τόσο μειώνεται το στοιχείο της φαντασίας και της προσποίησης. Για παράδειγμα στο παιχνίδι Uncharted 4, όσο πιο πολλές φορές το παίζει κάποιος, τόσο καλύτερος γίνεται σε αυτό και αυτό τον οδηγεί στο να μην σκέφτεται πια ότι ο ίδιος προσποιείται τον χαρακτήρα (avatar). Σκέφτεται μόνο να κρυφτεί, να κινηθεί

γρήγορα, να πυροβολήσει, να νικήσει τους εχθρούς, να μαζέψει πυρομαχικά κ.λπ. Οι παίχτες που παίζουν ένα παιχνίδι με τέτοιο αφηρημένο τρόπο, αδιαφορώντας για τον κόσμο του παιχνιδιού, είναι οι έμπειροι παίχτες. Για κάποιον ο οποίος παίζει πρώτη φορά ένα παιχνίδι, ο κόσμος του παιχνιδιού είναι πολύ σημαντικό να του κρατάει το ενδιαφέρον και να του εξάπτει την φαντασία.

Άλλος ένας σκοπός του κόσμου ενός παιχνιδιού (game world) είναι να βοηθήσει στις πωλήσεις του παιχνιδιού. Συνήθως, το κοινό δεν επιλέγει ένα παιχνίδι για τα mechanics του, αλλά για τον κόσμο που προσφέρει: ποιος θα είναι ο χαρακτήρας, τι θα αντιπροσωπεύει, τι θα νιώθει καθώς θα παίζει το παιχνίδι κ.λπ.

4.3 **Οι διαστάσεις ενός** Game World

Τον κόσμο ενός παιχνιδιού τον καθορίζουν πολλοί παράγοντες. Μερικοί, όπως το μέγεθος του κόσμου αυτού, είναι ποσοτικοί παράγοντες και μπορούν να τους δοθούν αριθμητικές τιμές. Άλλοι, όπως είναι η «διάθεση» του κόσμου, είναι ποιοτικοί παράγοντες και μπορούν να περιγραφούν με λόγια. Οι παράγοντες που έχουν συσχετιζόμενες ιδιότητες συνδέονται μεταξύ τους και σχηματίζουν ομάδες. Αυτές οι ομάδες αποτελούν τις διαστάσεις του game world και περιγράφονται παρακάτω.

4.3.1 **Η Φυσική Διάσταση** (Physical Dimension)

Ο κόσμος ενός video game εφαρμόζεται σχεδόν πάντα σύμφωνα με τα δεδομένα του πραγματικού κόσμου. Ο παίχτης μετακινεί τον χαρακτήρα του, επικοινωνεί με άλλους χαρακτήρες και χειρίζεται τα αντικείμενά μέσα σε αυτόν τον χώρο. Οι φυσικές ιδιότητες αυτού του χώρου καθορίζουν ένα μεγάλο μέρος του gameplay.

Ακόμα και τα text adventures παιχνίδια έχουν μία φυσική διάσταση. Ο παίχτης μετακινείται από μία αφηρημένη τοποθεσία, η οποία συνήθως αποκαλείται «δωμάτιο» ακόμα κι αν είναι εξωτερικός χώρος, σε μία άλλη.

Η φυσική διάσταση ενός παιχνιδιού χαρακτηρίζεται από κάποιες ιδιότητες. Αυτές είναι η χωρική διάσταση (spatial dimensionality), η κλίμακα (scale) και τα όρια της (boundaries).

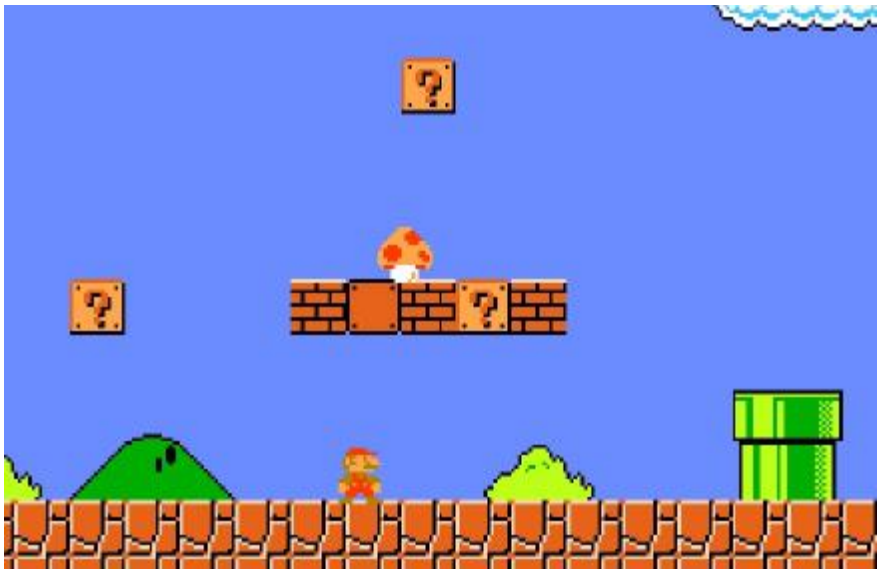
*4.3.1.1 **Χωρικές Διαστάσεις** (Spatial Dimensionalities)*

Ένα από τα σημαντικότερα ερωτήματα κατά την διάρκεια της δημιουργίας ενός παιχνιδιού, είναι το πόσες χωρικές διαστάσεις θα έχει ο φυσικός του χώρος. Είναι πολύ σημαντικό να κατανοηθεί πως οι χωρικές διαστάσεις του φυσικού χώρου ενός παιχνιδιού δεν είναι ίδιο με το πώς εμφανίζεται αυτός ο χώρος (camera model). Το πρώτο έχει να κάνει με τεχνικό σχεδιασμό, ενώ το δεύτερο με σχεδιασμό του user interface.

Όπως και να έχει, όλοι οι χώροι ενός παιχνιδιού θα πρέπει να εμφανίζονται σε μία δισδιάστατη οθόνη.

Οι τυπικές χωρικές διαστάσεις που μπορούν να χαρακτηρίσουν ένα παιχνίδι είναι οι εξής:

- 2D: Πριν από μερικά χρόνια, η πλειοψηφία των video games είχαν μόνο δύο διαστάσεις. Κυρίως, αυτό φαινόταν σε παιχνίδια side-rolling όπου η κάμερα ακολουθεί τον χαρακτήρα από αριστερά προς τα δεξιά και το αντίθετο, όπως για παράδειγμα το Super Mario Bros (εικόνα 4.3). Ο Mario μπορούσε να κινηθεί αριστερά και δεξιά, πάνω και κάτω, αλλά δεν μπορούσε να κινηθεί προς τον παίχτη (προς τα έξω της οθόνης) ή να απομακρυνθεί από αυτόν (προς τα μέσα της οθόνης).



Εικόνα 4.3 - **Το** side-scrolling **παιχνίδι** Super Mario Bros

Οι δισδιάστατοι κόσμοι έχουν ένα μεγάλο πλεονέκτημα όσον αφορά την παρουσίασή τους στην οθόνη: οι δύο διαστάσεις του χώρου ταυτίζονται με τις δύο διαστάσεις της οθόνης. Δεν χρειάζεται να τοποθετηθεί η ψευδαίσθηση του βάθους στον παίχτη. Από την άλλη, αρκετά παιχνίδια με 2D κόσμους χρησιμοποιούν έναν ειδικό μηχανισμό υλικού (hardware) που παρουσιάζει τον δισδιάστατο κόσμο σαν να είναι τρισδιάστατος, ενώ στην πραγματικότητα δεν έχει την τρίτη διάσταση. Στις μέρες μας, τα δισδιάστατα παιχνίδια δεν χρησιμοποιούνται πολύ, αλλά πολλά από αυτά είναι διαθέσιμα για κινητές συσκευές.

- 2.5D: Αναφέρεται στους κόσμους των παιχνιδιών οι οποίοι φαίνονται να είναι τρισδιάστατοι χώροι, αλλά στην πραγματικότητα είναι δισδιάστατα επίπεδα (2D layers) τοποθετημένα το ένα πάνω στο άλλο. Για παράδειγμα το παιχνίδι StarCraft περιέχει οροπέδια και πεδιάδες, καθώς και αεροσκάφη που μπορούν και προσπερνάνε εμπόδια πάνω από το έδαφος. Αυτά, βλέποντάς τα ο παίχτης, έχει την ψευδαίσθηση της τρίτης διάστασης. Ο παίχτης, μπορεί να τοποθετήσει και να μετακινήσει αντικείμενα οριζόντια μέσα σε ένα επίπεδο με κάποια ακρίβεια, αλλά όχι κατακόρυφα, ένα αντικείμενο μπορεί να βρίσκεται είτε σε ένα επίπεδο είτε σε άλλο. Δεν μπορεί να βρίσκεται στο ενδιάμεσό τους. Τα ιπτάμενα αντικείμενα δεν μπορούν να κινηθούν πάνω ή κάτω στον αέρα, απλά βρίσκονται στο επίπεδο που αναπαριστά την ατμόσφαιρα (εικόνα 4.4).

Εικόνα 4.4 - **Το παιχνίδι** StarCraft **που περιέχει οροπέδια και πεδιάδες, καθώς και αντικείμενα**

- 3D: Είναι ο κόσμος που αντιπροσωπεύει τις τρεις πραγματικές διαστάσεις. Χάρει στα σύγχρονα συστήματα που περιέχουν ειδικά υλικά (hardware) και λογισμικά (software), όπως και στα προγράμματα σχεδιασμού μοντέλων 3D, οι τρισδιάστατοι χώροι είναι πλέον πολύ εύκολο να δημιουργηθούν. Τέτοιοι χώροι δίνουν στον παίχτη την αίσθηση ότι βρίσκεται μέσα στον κόσμο του παιχνιδιού, ενώ οι δισδιάστατοι χώροι είναι απλά χώροι τους οποίους παρακολουθεί ο παίχτης από ψηλά. Οι 3D κόσμοι είναι κατάλληλοι για παιχνίδια που περιέχουν χαρακτήρες (avatars) και που δίνουν την δυνατότητα στο παίχτη να εξερευνήσει τον χώρο, όπως για παράδειγμα η σειρά παιχνιδιών Uncharted (εικόνα 4.5). Τα περισσότερα πλέον παιχνίδια είναι τρισδιάστατα.

Εικόνα 4.5 - **Στιγμιότυπο από το παιχνίδι** Uncharted 4, **ένα** 3D **παιχνίδι**

- 4D: Είναι μία δεύτερη εκδοχή του ήδη υπάρχοντος 3D κόσμου του παιχνιδιού. Δηλαδή είναι δύο ή περισσότεροι τρισδιάστατοι χώροι οι οποίοι μοιάζουν μεταξύ τους, αλλά διαφέρουν στις εμπειρίες που προσφέρουν στον παίχτη καθώς ο χαρακτήρας του περιπλανιέται σ αυτούς. Για παράδειγμα το παιχνίδι Legacy of Kain παρουσιάζει δύο εκδοχές του ίδιου τρισδιάστατου κόσμου, τα λεγόμενα Spiritual Realm και το Material Realm, με διαφορετικά gameplay modes για το κάθε ένα. Το τοπίο είναι το ίδιο και για τα δύο, όμως στο πρώτο υπάρχει μπλε φως, ενώ στο δεύτερο λευκό φως. Επίσης στο πρώτο η αρχιτεκτονική είναι παραμορφωμένη (εικόνα 4.6 και εικόνα 4.7).

Εικόνα 4.6 - **Το παιχνίδι** Legacy of Kain: Soul Reaver's **το** material realm

Εικόνα 4.7 - **Το παιχνίδι** Legacy of Kain: Soul Reaver's **το** spiritual realm

4.3.1.2 ***Η κλίμακα*** *(Scale)*

Η κλίμακα αναφέρεται στο απόλυτο μέγεθος του φυσικού χώρου που παρουσιάζεται στον κόσμο του παιχνιδιού και μετράται σε μονάδες που αφορούν αυτόν τον κόσμο (π.χ. μέτρα, μίλια, έτη φωτός), καθώς και με τα σχετικά μεγέθη των αντικειμένων μέσα στο παιχνίδι. Αν το παιχνίδι είναι αφηρημένο και δεν συσχετίζεται με τον πραγματικό κόσμο, τα μεγέθη των αντικειμένων του κόσμου του δεν έχουν μεγάλη σημασία. Στην περίπτωση όμως που το παιχνίδι αντιπροσωπεύει την πραγματικότητα, θα πρέπει ο κόσμος του να έχει το ακριβές μέγεθος που χρειάζεται για να φαίνεται αληθινός, αλλά να είναι και χρηστικός προς τον παίχτη. Σε μερικές περιπτώσεις είναι θεμιτό να παραμορφωθεί η κλίμακα του παιχνιδιού για χάρη του gameplay. Ειδικότερα σε παιχνίδια στρατηγικής η κλίμακα παραμορφώνεται, αλλά σε επίπεδο που δεν χαλάει την ψευδαίσθηση του χώρου που έχει ο παίχτης.

Σε παιχνίδια όπως είναι τα αθλητικά, τα παιχνίδια που περιέχουν οδήγηση οχήματος ή αυτά που είναι προσομοίωση πτήσεων, καθώς και όλα αυτά που περιέχουν σε μεγάλο βαθμό την αληθοφάνεια, δεν υπάρχει περιθώριο για αλλαγή ή παραμόρφωση της κλίμακας. Σοβαρά παιχνίδια που αναπαριστούν τον πραγματικό κόσμο θα πρέπει να είναι όσο το δυνατόν γίνεται πιο ακριβή όσον αφορά τον φυσικό τους κόσμο.

Επίσης, τα first-person παιχνίδια διαθέτουν ακρίβεια στην κλίμακά τους, αφού αντιπροσωπεύουν έναν κόσμο όπου η προοπτική του παίχτη είναι αυτή που έχει ένας άνθρωπος καθώς προχωράει μπροστά σε ένα χώρο και τα αντικείμενα που βλέπει είναι πραγματικού μεγέθους, όπως πόρτες, έπιπλα κ.λπ.. Μερικά αντικείμενα σε τέτοιους κόσμους μπορούν να δεχτούν ελάχιστη παραμόρφωση στην κλίμακά τους έτσι ώστε να μπορούν να είναι ορατά στον παίχτη, όπως για παράδειγμα κλειδιά, όπλα, πυρομαχικά.

Πολλές φορές υπάρχει παραμόρφωση στην κλίμακα ενός παιχνιδιού, όσον αφορά το ύψος των ανθρώπων, των κτηρίων και των λόφων που υπάρχουν στο περιβάλλον. Για παράδειγμα στο παιχνίδι Age of Empires τα κτήρια φαίνονται να είναι λίγο ψηλότερα από τους ανθρώπους που περνάνε δίπλα τους (εικόνα 4.8). Για να μπορέσει ο παίχτης να δει τα κτήρια και τις οροφές τους θα πρέπει η κάμερα να τοποθετηθεί στο ψηλότερο σημείο του game world. Αν όμως τοποθετηθεί πάρα πολύ ψηλά, οι άνθρωποι δεν θα είναι ορατοί σχεδόν καθόλου. Για να λυθεί αυτό το πρόβλημα, το παιχνίδι δεν περιέχει ψηλά κτήρια και λόφους και υπάρχει παραμόρφωση στην κλίμακα των ανθρώπων, αφού έχουν μεγαλύτερο μέγεθος από το κανονικό.

Άλλη μία παραμόρφωση που μπορεί να σημειωθεί αφορά την αίσθηση του χρόνου και αυτό σχετίζεται με την ταχύτητα των αντικειμένων στον χώρο. Για παράδειγμα αν ένας game world περιλαμβάνει αεροσκάφη, οχήματα εδάφους και ανθρώπους που μπορούν και τρέχουν, θα πρέπει να οριστεί μία σχετικά παραμορφωμένη κλίμακα, όπου πάντα θα είναι γρηγορότερο το αεροσκάφος. Δεν γίνεται να φτιαχτεί ένας κόσμος όπου θα αντιστοιχεί η ταχύτητα ενός πραγματικού αεροσκάφους σε αυτό που υπάρχει μέσα στο παιχνίδι, διότι σε αυτή την περίπτωση η ταχύτητα του ανθρώπου θα είναι πολύ αργή σε σχέση με αυτή της πραγματικότητας.

Εικόνα 4.8 - **Το παιχνίδι** Age of Empires. **Ο κόσμος του έχει δεχτεί παραμόρφωση στην κλίμακα των κτηρίων και των ανθρώπων**

*4.3.1.3 **Τα Όρια** (The Boundaries)*

Ο κόσμος ενός παιχνιδιού πρέπει να έχει πάντα όρια. Κύριος λόγος είναι ότι οι υπολογιστές έχουν περιορισμένη μνήμη για να αποθηκεύουν δεδομένα. Για παράδειγμα σε ένα επιτραπέζιο παιχνίδι, τα όρια του board του αποτελούν και τα όρια του κόσμου του. Τα μεγάλα παιχνίδια δουλεύουν διαφορετικά όμως από τα επιτραπέζια. Σε αυτά θα πρέπει να δοθούν όρια στον game world, αλλά ταυτόχρονα θα πρέπει αυτός ο κόσμος να έχει την αμφίεση ότι είναι απεριόριστος.

Σε μερικές περιπτώσεις όπου τα όρια ενός παιχνιδιού έρχονται με φυσικό τρόπο, είναι εύκολο να δημιουργηθούν από την σχεδιαστική ομάδα. Τέτοια παιχνίδια είναι τα αθλητικά, όπου αναπαριστούν μόνο μία αρένα ή στάδιο και δεν υπάρχει η απαίτηση να δημιουργηθεί περεταίρω κόσμος. Επίσης και τα παιχνίδια που περιέχουν αγώνες αυτοκινήτων, περιορίζονται σε δρόμους.

Επιπλέον, σε παιχνίδια που αναπαριστούν εσωτερικούς χώρους ή υπόγειους, είναι εύκολο να δημιουργηθούν τα όρια του κόσμου τους. Το πρόβλημα εμφανίζεται όταν τα παιχνίδια λαμβάνουν χώρα σε εξωτερικούς χώρους όπου οι παίχτες προσμένουν να υπάρχουν μεγάλες και ανοιχτές περιοχές, τις όποιες θα μπορούν να εξερευνήσουν, αλλά ταυτόχρονα να έχουν ξεκάθαρα όρια στους κόσμους τους. Μία λύση για το πρόβλημα αυτό είναι ο κόσμος του παιχνιδιού να αποτελείται από ένα νησί και σε ένα μακρινό σημείο του νερού, μπορούν να καθοριστούν τα όρια αυτού του κόσμου. Επίσης, λύση αποτελεί και η τοποθέτηση βουνών, ερήμων ή οποιασδήποτε μορφής δύσβατου εδάφους έτσι ώστε να είναι εμφανή τα όρια κάθε τέτοιου κόσμου.

4.3.2 **Η Χρονική Διάσταση** (The Temporal Dimension)

Η χρονική διάσταση ενός game world καθορίζει τον τρόπο με τον οποίο θα συμπεριφέρεται ο χρόνος μέσα σε αυτόν τον κόσμο, καθώς και τους τρόπους στους οποίους διαφέρει με τον χρόνο στον πραγματικό κόσμο.

Σε πολλά action παιχνίδια, ο κόσμος τους δεν περιέχει την έννοια του χρόνου με την πραγματική της μορφή. Όλα τα γεγονότα συμβαίνουν με τον χρόνο που σου δείχνει ότι είναι στην αφήγηση της ιστορίας ή στο αν είναι μέρα ή νύχτα, που φαίνεται λόγω γραφικών. Μέχρις ότου ο παίχτης να φτάσει σε σημείο στο παιχνίδι όπου αλλάζει η ιστορία χρονικά, θα παραμένει ο χρόνος σε ένα βρόχο ή απλά σταθερός. Σε μερικές μόνο περιπτώσεις ο παίχτης θα πρέπει να εκτελέσει μία ενέργεια σε ένα παιχνίδι το οποίο του ορίζει αληθινό χρονικό διάστημα. Για παράδειγμα, αν σε μία σκηνή μάχης θα πρέπει ο παίχτης να οδηγήσει τον χαρακτήρα του στην έξοδο πριν εκραγεί μία χειροβομβίδα, τότε το game world ορίζει 10 δευτερόλεπτα πραγματικού χρόνου για να προλάβει να ξεφύγει ο παίχτης.

Σε μερικά παιχνίδια ο χρόνος αποτελεί κομμάτι του κόσμου του αλλά όχι του gameplay. Δηλαδή, καθώς περνάει ο χρόνος μέσα στο παιχνίδι θα υπάρχει αλλαγή στα γραφικά του παιχνιδιού και θα μετατρέπεται σταδιακά η μέρα σε νύχτα και το αντίθετο, αλλά στην πραγματικότητα δεν επηρεάζει το ίδιο το παιχνίδι ούτε τις δοκιμασίες που προσφέρει στον παίχτη. Αυτή η επιλογή είναι καθαρά καλλιτεχνική.

Υπάρχουν όμως και παιχνίδια που επηρεάζεται το gameplay τους από τον χρόνο. Για παράδειγμα στο παιχνίδι Grand Theft Auto: Vice City, όταν βραδιάζει τα μαγαζιά που παρέχει είναι κλειστά και ανοίγουν πάλι το πρωί. Επίσης, κάποιες αποστολές που αφορούν τον χρόνο πρέπει να γίνουν εκείνη τη χρονική στιγμή αλλιώς θα πρέπει ο παίχτης να περιμένει την επόμενη μέρα για να την εκτελέσει. Όμως, η έννοια του χρόνου σε αυτό το παιχνίδι δεν ακολουθεί την πραγματική ροή του χρόνου, αλλά μία δική του όπου σε αντιστοιχία με την πραγματικότητα οι ώρες είναι λεπτά.

4.3.3 **Η Περιβαλλοντική Διάσταση** (The Environmental Dimension)

Αυτή η διάσταση περιγράφει την εμφάνιση και την ατμόσφαιρα του game world. Στην φυσική διάσταση, είδαμε ότι ορίζονται οι ιδιότητες του περιβάλλοντος, ενώ στην περιβαλλοντική διάσταση καθορίζεται το τι υπάρχει μέσα σε αυτό. Τα περιβαλλοντικά χαρακτηριστικά ενός game world ορίζουν και το οπτικό-ακουστικό του περιεχόμενο. Δύο βασικοί παράγοντες αυτής της διάστασης είναι το *πολιτισμικό περιεχόμενο* του κόσμου του παιχνιδιού, καθώς και τα *αντικείμενα που εμπεριέχονται* σε αυτόν (Adams, 2010).

- **Πολιτισμικό Περιεχόμενο**

  Το πολιτισμικό περιεχόμενο ενός παιχνιδιού αναφέρεται στην κουλτούρα του με την ανθρωπολογική έννοια. Δηλαδή, τα πιστεύω, οι αξίες, οι συμπεριφορές των ανθρώπων που ζουν μέσα στον κόσμο ενός παιχνιδιού, καθώς και οι θρησκευτικές, οι κοινωνικές και οι πολιτικές τους πεποιθήσεις. Αυτά συνήθως μεταφέρονται μέσα από το σχεδιαστικό κομμάτι του παιχνιδιού, δηλαδή μέσα από τα ρούχα, τα έπιπλα,

την αρχιτεκτονική, τα τοπία και οποιοδήποτε άλλο αντικείμενο που αφορά την ανθρώπινη κατασκευή.

Επίσης, η κουλτούρα ενός παιχνιδιού κρύβεται και στην προϊστορία που παρουσιάζει, είτε αυτή είναι μεγάλης κλίμακας, όπως για παράδειγμα καταστροφές, πόλεμοι, είτε είναι μικρής κλίμακας, όπως προσωπικά γεγονότα και αλληλεπιδράσεις μεταξύ χαρακτήρων ενός παιχνιδιού, που συνέβησαν πριν ξεκινήσει την χρονική περίοδο όπου παρουσιάζει το παιχνίδι ως παρόν. Αυτό το κομμάτι της ιστορίας συνήθως βοηθάει στο να κατανοήσει ο παίχτης από πού προήλθε αυτή η κουλτούρα.

- **Τα Αντικείμενα που Εμπεριέχονται σε ένα** Game World

  Τα αντικείμενα που εμπεριέχονται στον κόσμο ενός παιχνιδιού ορίζουν και την εμφάνισή του. Ανάλογα με το τι αφορά ένα παιχνίδι, τέτοια αντικείμενα μπορεί να είναι κτήρια, οχήματα, ρουχισμός, όπλα, έπιπλα, κάθε είδους διακόσμηση, κοσμήματα, θρησκευτικά ή μαγικά αντικείμενα, λογότυπα ή εμβλήματα, αξεσουάρ κ.λπ.

  Αντικείμενα ενός κόσμου, επιπλέον, αποτελούν και τα ζώα, τα δέντρα, η γη, οι πέτρες, οι λόφοι, τα φυτά, ακόμα και ο ουρανός.

  Επίσης, τέτοια αντικείμενα είναι οι ήχοι και ο τρόπος εμφάνισής τους. Δηλαδή, η μουσική, οι ήχοι του περιβάλλοντος, ήχοι που κάνουν τα ζώα, οι άνθρωποι, τα μηχανήματα και τα οχήματα, καθώς και το πώς φαίνονται ή παρουσιάζονται όλα αυτά.

  Τα αντικείμενα αυτά παίζουν σημαντικό ρόλο στην δημιουργία ενός τόνου και μιας συγκεκριμένης διάθεσης σε ένα game. Για παράδειγμα το παιχνίδι Super Mario προσφέρει στον παίχτη ένα τόνο χαράς και μία ξέγνοιαστη διάθεση, ενώ το παιχνίδι Uncharted δημιουργεί την αίσθηση της αγωνίας και της δράσης.

4.3.4 **Η Συναισθηματική Διάσταση** (The Emotional Dimension)

Η συναισθηματική διάσταση του κόσμου ενός παιχνιδιού περιλαμβάνει τα συναισθήματα που έχουν οι χαρακτήρες του παιχνιδιού, καθώς και τα συναισθήματα που δημιουργούνται στον παίχτη κατά την διάρκεια του παιχνιδιού. Συνήθως τα παιχνίδια που είναι διαθέσιμα σε multiplayer, προκαλούν τα περισσότερα συναισθήματα στους παίχτες γιατί μέσα από αυτά, τους δίνεται η ευκαιρία να γνωρίσουν καινούριους ανθρώπους και πιθανόν να κάνουν καινούργιους φίλους ή εχθρούς. Τα παιχνίδια που είναι single-player προκαλούν τον συναισθηματικό κόσμο των παιχτών μέσω της ιστορίας και του gameplay τους. Τα παιχνίδια που ανήκουν στην κατηγορία της στρατηγικής, προκαλούν λιγότερα συναισθήματα στον παίχτη.

4.3.5 **Η Ηθική Διάσταση** (The Ethical Dimension)

Η ηθική διάσταση ενός game world καθορίζει τι σημαίνει το σωστό και τι το λάθος σε αυτό το παιχνίδι. Τα περισσότερα παιχνίδια που έχουν ένα φανταστικό περιεχόμενο, έχουν επίσης και ένα σύστημα ηθικής που πρέπει ο παίχτης να ακολουθήσει και να

συμπεριφέρεται βάσει αυτού. Με αυτόν τον τρόπο ορίζονται οι πράξεις του παίχτη αν είναι σωστές ή λάθος, οι ενέργειες που πρέπει να αποφύγει ή όχι κ.λπ.

Η ηθική διάσταση αποτελεί κομμάτι της κουλτούρας και της ιστορίας ενός παιχνιδιού, που αυτά με την σειρά τους αποτελούν κομμάτι της περιβαλλοντικής διάστασης. Η ηθική ενός game world διαφέρει σε πολλά σημεία με αυτή του πραγματικού κόσμου. Τα παιχνίδια επιτρέπουν στον παίχτη να κάνει πράγματα που στην πραγματικότητα δεν μπορεί να κάνει. Για παράδειγμα, στο παιχνίδι Assassin's Creed είναι θεμιτό ο χαρακτήρας να σκοτώνει και να κλέβει, υπό τον όρο, όμως, ότι κάνει αυτές τις ενέργειες μόνο εναντίων των εχθρών και όχι των απλών, αθώων πολιτών.

5 **Τα Είδη των** Video Games

Τα είδη των παιχνιδιών δημιουργήθηκαν έτσι ώστε να ορίζονται ως ένα σύνολο παιχνίδια τα οποία είναι παρόμοια και έχουν κοινά στοιχεία. Το gameplay κάθε παιχνιδιού ορίζει και το είδος που ανήκει. Υπάρχουν παιχνίδια που μπορεί να έχουν ίδιες ή παρόμοιες ιδιότητες, αλλά στην πραγματικότητα να ανήκουν σε διαφορετικό είδος.

Επίσης πολλά παιχνίδια δεν ανήκουν μόνο σε ένα είδος. Μπορούν να περιέχουν στοιχεία από δύο η περισσότερα είδη ταυτόχρονα. Παρακάτω θα δούμε τα πιο γνωστά είδη των video games.

5.1 Action Games

Σε αυτή την κατηγορία ανήκουν τα περισσότερα παιχνίδια. Είναι η πιο διαδεδομένη και η πιο συνηθισμένη στο ευρύ κοινό, αφού όταν κάποιος ακούει την έννοια video game, αμέσως του έρχεται στο μυαλό αυτό το είδος παιχνιδιών. Επιπλέον, περιλαμβάνει πολλά διαφορετικά είδη τα οποία είναι μεν ξεχωριστά, αλλά στην πραγματικότητα αποτελούν κομμάτι του.

Στα action games η πλειοψηφία των ενεργειών που πρέπει να εκτελέσει ο παίχτης αποτελούν δοκιμασία των φυσικών ικανοτήτων και του συντονισμού του. Επίσης εμπεριέχουν γρίφους, διαμάχες, δοκιμές στις οποίες ο παίχτης θα πρέπει να εξερευνήσει τον χώρο κ.λπ.

Αυτού του είδους τα παιχνίδια απαιτούν από τον παίχτη να έχει καλό συντονισμό στα μάτια και τα χέρια του, αφού συνήθως χρειάζονται γρήγορες κινήσεις και αντανακλαστικά για να ξεπεράσουν μία δοκιμασία. Επιπλέον, ο παίχτης θα πρέπει να έχει την ικανότητα να στοχεύει σταθερά, να έχει τον ρυθμό του συγχρονισμού ή την δυνατότητα να εκτελεί ενέργειες που απαιτούν πολλές εντολές πλήκτρων είτε από πληκτρολόγια και ποντίκι, είτε από χειριστήριο.

Τα πιο γνωστά είδη που ανήκουν σε αυτή την κατηγορία είναι τα παρακάτω:

5.1.1 Shooters

Σε αυτό το είδος ο κύριος στόχος του παίχτη είναι να πυροβολεί και να κερδίζει μάχες όπου υπάρχουν άφθονοι πυροβολισμοί μεταξύ του παίχτη και των εχθρών. Υπάρχουν τα first-person shooter (αλλιώς γνωστά και ως FPS) games και τα third-person shooter games. Στα πρώτα, ο παίχτης αντιπροσωπεύεται από κάποιον χαρακτήρα (avatar), αλλά δεν μπορεί να τον δει στην οθόνη του παρά μόνο να δει μέσα από τα μάτια του, όπως είναι το παιχνίδι Call of Duty (εικόνα 5.1), ενώ στα δεύτερα, ο παίχτης αντιπροσωπεύεται από έναν χαρακτήρα όπου τον βλέπει στην οθόνη του και εκτελεί μέσω αυτού τις δοκιμασίες του παιχνιδιού, όπως είναι το παιχνίδι Rise of the Tomb Raider (εικόνα 5.2).

Εικόνα 5.1 - **Στιγμιότυπο από το παιχνίδι** Call of Duty (FPS Game)

Εικόνα 5.2 - **Στιγμιότυπο από το** Rise of the Tomb Raider (Third Person Shooter Game)

5.1.2 Fighting Games

Αυτή η κατηγορία παιχνιδιών δεν έχει πολλά κοινά στοιχεία με τα υπόλοιπα action games. Δεν περιέχουν πυροβολισμούς, εξερεύνηση του περιβάλλοντος, γρίφους κ.λπ.. Απαιτούν όμως από τον παίχτη να μπορεί να ανταπεξέλθει σε δύσκολες δοκιμασίες γρήγορα και να έχει συγχρονισμό.

Αρχικά ήταν βασισμένα σε μία δισδιάστατη πλατφόρμα όπου οι χαρακτήρες μπορούσαν να κινηθούν μόνο πάνω, κάτω, δεξιά και αριστερά και είχαν περιορισμένο αριθμό πιθανών κινήσεων. Ένα χαρακτηριστικό παιχνίδι είναι το Street Fighter series (εικόνα 5.3), το οποίο είναι ένα από τα πρώτα παιχνίδια που δημιουργήθηκαν στο είδος αυτό. Επίσης, το Mortal Kombat είναι ένα πολύ καλό παράδειγμα για την εξέλιξη του είδους αυτού, όσον αφορά τον γραφικό τομέα και το πλήθος των πιθανών κινήσεων που μπορούσε να εκτελέσει ο χαρακτήρας (combos), παραμένοντας όμως στις δύο διαστάσεις (εικόνα 5.4). Τέλος, τα fighting games της εποχής μας έχουν εξελιχθεί σε πολύ μεγάλο βαθμό, αφού πλέον προστέθηκε η τρίτη διάσταση, όπως και πολλοί περισσότεροι συνδυασμοί κινήσεων των χαρακτήρων. Τέτοιο παιχνίδι, για παράδειγμα, είναι το Dragon Ball Z Ultimate Tenkaichi (εικόνα 5.5).

Εικόνα 5.3 - **Στιγμιότυπο από το παιχνίδι** Street Fighter

Εικόνα 5.4 - **Στιγμιότυπο από το παιχνίδι** Mortal Kombat

Εικόνα 5.5 - **Στιγμιότυπο από το παιχνίδι** Dragon Ball Z Ultimate Tenkaichi

5.2 Adventure Games

Πολλά παιχνίδια τα οποία περιέχουν την έννοια της περιπέτειας δεν ανήκουν απαραίτητα στην κατηγορία των adventure games. Αυτή η κατηγορία αναφέρεται σε παιχνίδια που διαφέρουν αρκετά από τα υπόλοιπα παιχνίδια της αγοράς. Δεν περιέχουν το στοιχείο του συναγωνισμού ή της αναπαράστασης, δεν προσφέρει διαδικασίες που πρέπει να κάνει ο παίχτης έτσι ώστε να κερδίσει τους εχθρούς του με κύριο όπλο την στρατηγική και την τακτική. Αντίθετα, τα adventure games παρουσιάζουν την ιστορία του χαρακτήρα που χειρίζεται ο παίχτης. Αυτός ο χαρακτήρας δεν αποτελεί ταύτιση του με τον παίχτη, αλλά είναι ένας φανταστικός άνθρωπος με την δική του προσωπικότητα, ο πρωταγωνιστής και ο ήρωας της ιστορίας. Κύρια συστατικά του gameplay τους αποτελούν η αφήγηση της ιστορίας και η εξερεύνηση, όπως και η επίλυση γρίφων σε συνδυασμό με τις δοκιμασίες που αναθέτονται στον παίχτη. Τα στοιχεία της μάχης, της οικονομικής οργάνωσης και της δράσης είναι σχεδόν ανύπαρκτα. Παράδειγμα τέτοιου παιχνιδιού είναι το Dreamfall (εικόνα 5.6).

Λίγα τέτοια παιχνίδια έχουν δημιουργηθεί. Η πιο συνηθισμένη μορφή αυτής της κατηγορίας παιχνιδιών είναι τα Action-Adventure.

Εικόνα 5.6 - **Στιγμιότυπο από το παιχνίδι** Dreamfall

5.2.1 Action-Adventure

Αυτό αποτελεί συνδυαζόμενο είδος, αφού εμπεριέχει στοιχεία από τα action και από τα adventure παιχνίδια. Και αυτά, όπως τα action, για να τα παίξει ο παίχτης θα πρέπει να διαθέτει συγκεκριμένες ικανότητες, αλλά εμπεριέχουν επίσης ιστορία, πολλούς διαφορετικούς χαρακτήρες, διαλόγους και άλλα στοιχεία που έχουν τα adventure games. Είναι πολύ συνηθισμένος αυτός ο συνδυασμός των δύο ειδών και αποτελεί την πλειοψηφία των action και των adventure games. Τέτοια παιχνίδια είναι το Uncharted Series, το Tomb Raider series, το Assassin's Creed series και πολλά άλλα.

Εικόνα 5.7 - **Στιγμιότυπο από το παιχνίδι** Assassin's Creed Brotherhood

5.3 Strategy Games

Τα παιχνίδια στρατηγικής είναι αυτά που προκαλούν τον παίχτη να κερδίσει το παιχνίδι μέσω του σχεδιασμού, της οργάνωσης και ειδικότερα μέσω της προ-σχεδίασης μιας σειράς ενεργειών οι οποίες θα εκτελεστούν εναντίων ενός ή πολλών εχθρών μαζί. Κύριος στόχος αυτών των παιχνιδιών είναι ο παίχτης να καταφέρει να μειώσει τις δυνάμεις του εχθρού, και γι' αυτό τα περισσότερα strategy games είναι πολεμικά. Από την άλλη, δεν εστιάζουν όλα τα παιχνίδια στρατηγικής στην μάχη και τον πόλεμο. Πολλά από αυτά στοχεύουν περισσότερο στην έννοια της κατάκτησης νέων περιοχών από τον παίχτη, καθώς και στην προσπάθεια να διατηρηθεί η ειρήνη στις περιοχές που κατέκτησε όσο το δυνατόν περισσότερη ώρα γίνεται.

Επομένως, τα παιχνίδια στρατηγικής εμπεριέχουν ως πλειοψηφία δοκιμασίες οι οποίες ξεπερνιούνται μόνο με χρήση της στρατηγικής και ο παίχτης έχει τη δυνατότητα να διαλέξει ανάμεσα από πολλές πιθανές κινήσεις, έτσι ώστε να το καταφέρει. Η νίκη προέρχεται μόνο από την οργάνωση και την κατάστρωση σχεδίων και το στοιχείο της πιθανότητας σε αυτά τα παιχνίδια είναι σχεδόν ανύπαρκτο.

Τα πιο γνωστά παιχνίδια στρατηγικής είναι τα Total War Series (εικόνα 5.8).

Εικόνα 5.8 - **Στιγμιότυπο από το παιχνίδι** Total War Arena

5.4 Role-Playing Games (RPG)

Σε αυτή την κατηγορία ανήκουν τα παιχνίδια όπου ο παίχτης χειρίζεται έναν ή περισσότερους χαρακτήρες, συνήθως σχεδιασμένοι από τον ίδιο (εικόνα 5.9), τους οποίους οδηγεί ανάμεσα από μία σειρά δοκιμασιών τις οποίες πρέπει να ξεπεράσει έτσι ώστε να κερδίσει το παιχνίδι. Κύριο συστατικό τέτοιων παιχνιδιών είναι η εξέλιξη του χαρακτήρα και την σταδιακή αύξηση των ικανοτήτων του καθώς περνάει τις δοκιμασίες. Αυτό ορίζεται ως «άνοδος των level του χαρακτήρα» στην κοινότητα των video games. Οι δοκιμασίες που αντιμετωπίζει ο παίχτης, αποτελούν συνήθως μάχες τακτικής, οικονομική οργάνωση, εξερεύνηση, επίλυση γρίφων κ.λπ..

Τα πιο γνωστά παραδείγματα αυτής της κατηγορίας είναι τα παιχνίδια Elder Scrolls Series και Witcher Series.

Αυτή η κατηγορία αναμιγνύεται με πολλά άλλα είδη παιχνιδιών όπως είναι τα action, τα adventure και τα war games.

Εικόνα 5.9 - **Δημιουργία χαρακτήρα στο παιχνίδι** Skyrim

Εικόνα 5.10 - **Στιγμιότυπο από το παιχνίδι** The Witcher 3

5.5 Sport Games

Τα παιχνίδια αυτά αναπαριστούν κάθε είδους σπορ, είτε αυτό υπάρχει στην πραγματικότητα, είτε αποτελεί αντικείμενο φαντασίας. Κύριος σκοπός τους είναι να μπορεί ένας ή περισσότεροι παίχτες να παίξουν το αντιπροσωπευόμενο άθλημα, να οργανώσουν την διαχείρισή του και να ορίσουν τους ρόλους που θα έχουν οι αθλητές (εικόνα 5.11).

Σε αντίθεση με τα περισσότερα παιχνίδια τα οποία λαμβάνουν χώρα σε έναν τόπο ο οποίος είναι άγνωστος προς τον παίχτη, αυτή η κατηγορία περιλαμβάνει παιχνίδια τα οποία αντιπροσωπεύουν έναν αληθινό κόσμο, όπως για παράδειγμα ένα αληθινό γήπεδο ποδοσφαίρου ή μπάσκετ (εικόνα 5.12).

Η κατηγορία των sport games αποτελεί στις μέρες μας μία από τις πιο δημοφιλείς κατηγορίες και τα γνωστότερα παραδείγματα του είδους είναι το FIFA soccer games, το Pro Evolution Soccer και το NBA 2K.

Εικόνα 5.11 - **Το** FIFA 15 **όπου μπορεί ο παίχτης να οργανώσει τους αθλητές όπως θέλει**

Εικόνα 5.12 - **Το παιχνίδι** NBA 2K 17 **όπου αναπαρίσταται το πραγματικό γήπεδο μπάσκετ**

5.6 Vehicle Simulation Games

Σε αυτό το είδος ανήκουν τα παιχνίδια που περιέχουν προσομοιώσεις οχημάτων, δηλαδή σε αυτά ο παίχτης οδηγεί οχήματα σαν να συμβαίνει στην πραγματικότητα. Τέτοια οχήματα μπορεί να είναι αυτοκίνητα, αεροπλάνα, πλοία κ.λπ.. Ο κόσμος και το περιβάλλον είναι αντίστοιχος της πραγματικότητας και συνήθως το μέρος που αντιπροσωπεύει υπάρχει και είναι ίδιο σχεδιαστικά.

Οι δύο μεγαλύτερες υποκατηγορίες αυτού του είδους είναι οι εξής:

- Flight Simulation, στις οποίες ο παίχτης χειρίζεται αεροπλάνα, είτε επιβατικά, είτε αεροπλάνα της πολεμικής αεροπορίας. Μπορεί να οδηγηθεί σε μάχες μεταξύ άλλων αεροπλάνων, ή να περιπλανηθεί στον αέρα. Παράδειγμα τέτοιου παιχνιδιού είναι το flight simulator (εικόνα 5.13) και το Combat Flight Simulator (εικόνα 5.14).

Εικόνα 5.13 - Flight Simulator 2016

Εικόνα 5.14 - Combat Flight Simulator

- Racing. Αυτό το είδος προσομοιώνει αγώνες αυτοκινήτων σε μεγάλους δρόμους, όπως επίσης πολλές φορές προσομοιώνει και αστυνομική καταδίωξη. Κύριος στόχος του παίχτη είναι να βγει νούμερο ένα στους αγώνες αυτούς και να αποκτήσει το καλύτερο και γρηγορότερο αμάξι που είναι διαθέσιμο στο παιχνίδι. Στα περισσότερα του είδους παιχνίδια, ο παίχτης έχει την δυνατότητα να διαμορφώσει το αμάξι του όπως ο ίδιος θέλει (εικόνα 5.15). Το γνωστότερο παράδειγμα αυτού του είδους είναι τα Need For Speed Series (εικόνα 5.16), καθώς και το Gran Turismo (εικόνα 5.17).

Εικόνα 5.15 - **Διαμόρφωση του οχήματος**, Need for Speed Most Wanted

Εικόνα 5.16 - **Προσομοίωση αγώνα δρόμου**, Need for Speed Most Wanted

Εικόνα 5.17 - Gran Turismo 6

5.7 LAN **και** Online Games

Παλαιότερα, αν ήθελαν κάποιοι να παίξουν multiplayer παιχνίδια, θα έπρεπε να μαζευτούν σε ένα χώρο και να παίξουν όλοι μαζί με χρήση μίας κονσόλας ή πολλών διαφορετικών μηχανημάτων, αλλά μέσω ενός κοινού δικτύου. Τέτοιου είδους παιχνίδια ονομάζονται LAN games και είναι συνήθως τα Sports και τα Fighting. Αυτό συνέβαινε διότι τα υπολογιστικά συστήματα καθώς και το διαδίκτυο δεν είχαν αναπτυχθεί σε τέτοιο βαθμό ώστε να υποστηρίζουν μεγάλες και ταυτόχρονες συνδέσεις από διαφορετικούς υπολογιστές σε όλο τον κόσμο.

Στην εποχή μας, όπου η τεχνολογία έχει εξελιχθεί δραματικά, τα περισσότερα παιχνίδια όλων των ειδών είναι διαθέσιμα και online. Δηλαδή μπορεί ο κάθε παίχτης να συνδεθεί μέσω διαδικτύου και να παίξει το παιχνίδι που επιθυμεί ταυτόχρονα με άλλους παίχτες, είτε αυτοί είναι φίλοι του, είτε είναι παίχτες από όλο τον κόσμο. Είναι η πιο διαδεδομένη κατηγορία παιχνιδιών, αφού εκτός από μαζική διασκέδαση, προσφέρει και την δυνατότητα γνωριμίας καινούργιων ανθρώπων και επιπλέον αυξάνει την κοινωνικοποίηση των παιχτών. Επίσης είναι πολύ σημαντικό ότι προσφέρει μεγάλο επίπεδο ανταγωνισμού μεταξύ των παιχτών και τους δίνει την δυνατότητα να γίνουν καλύτεροι ή ακόμα και να γίνουν επαγγελματίες, αφού οργανώνονται πρωταθλήματα από μεγάλες και μικρές εταιρείες με επιβράβευση κάποιο χρηματικό έπαθλο και μεγάλη αναγνώριση ανά τον πλανήτη.

Τα online games λειτουργούν με βάσεις δεδομένων και ειδικούς servers που παρέχονται από τις εταιρείες που φτιάχνουν τα συγκεκριμένα παιχνίδια, έτσι ώστε να μπορούν να συμβαίνουν ταυτόχρονα πολλές διαδικτυακές συνδέσεις χωρίς να παίζει μεγάλο ρόλο σε ποια περιοχή βρίσκεται ο κάθε παίχτης.

Πολλά action και action-adventure παιχνίδια, εκτός από το single-player που διαθέτουν, έχουν πλέον την επιλογή της χρήσης του online multiplayer. Για παράδειγμα το Uncharted 4

στο single-player μπορεί ο παίχτης να παίξει μόνος του την ιστορία του παιχνιδιού και να το τερματίσει, αλλά μπορεί να επιλέξει από το μενού το online multiplayer, όπου συνδέεται στο παιχνίδι μαζί με άλλους παίχτες ανά τον κόσμο και παίζουν όλοι μαζί τις διάφορες μάχες που παρέχονται.

Η κυριότερη κατηγορία αυτού του είδους είναι η Massively Multiplayer Online Role-Playing Games (MMORPGs), καθώς και η Multiplayer Online Battle Arena Games (MOBA).

5.7.1 Massively Multiplayer Online Role-Playing Games (MMORPGs)

Είναι online multiplayer παιχνίδια στα οποία ο κάθε παίχτης αντιπροσωπεύεται από έναν χαρακτήρα στον game world. Περιλαμβάνουν τα χαρακτηριστικά των RPG Games με την μόνη διαφορά ότι μπορούν πολλοί παίχτες από όλο τον κόσμο να παίζουν ταυτόχρονα με αποτέλεσμα οι αποφάσεις και οι ενέργειες του ενός να επηρεάζουν τον κόσμο του παιχνιδιού και για τους υπόλοιπους. Προωθούν την συνεργασία μεταξύ των παιχτών με σκοπό να ανταπεξέλθουν στις διάφορες δυσκολίες που συναντούν, καθώς και τον υγιή ανταγωνισμό, αφού ο καθένας προσπαθεί να γίνει καλύτερος από τους υπόλοιπους με διάφορους τρόπους. Οι δύο βασικοί τρόποι που προσφέρονται στα περισσότερα MMORPGs είναι είτε μέσω ενός κοινού βαθμολογικού πίνακα που εμπεριέχεται σε κάθε server ξεχωριστά, είτε μέσω των αντικειμένων που ο κάθε παίχτης συλλέγει για τον χαρακτήρα του. Μία ακόμα πολύ σημαντική πτυχή του είδους αυτού είναι ότι προσφέρει την επιλογή δύο διαφορετικών game-style, του Person-versus-Person (PVP) και του Person-versus-Environment (PVE), όπου στο πρώτο ο παίχτης βρίσκεται αντιμέτωπος με άλλους παίχτες, ενώ στο δεύτερο βρίσκεται αντιμέτωπος με τα αντικείμενα του game world. Τέλος, είναι στην ευχέρεια του καθενός να επιλέξει το game-style που επιθυμεί, χωρίς αυτό να σημαίνει ότι δεν μπορεί να ακολουθήσει και τα δύο ταυτόχρονα.

Το πιο χαρακτηριστικό παράδειγμα παιχνιδιού αυτής της κατηγορίας είναι το World of Warcraft (εικόνα 5.18).

Εικόνα 5.18 - World of Warcraft. **Φαίνονται όλοι οι διαφορετικοί παίχτες που πολεμάνε ένα τέρας**

5.7.2 Multiplayer Online Battle Arena Games (MOBA)

Τα MOBA games είναι μία από τις νεότερες και πιο διαδεδομένες κατηγορίες παιχνιδιών και είναι πλήρως βασισμένα στο multiplayer. Σε αντίθεση με τα MMORPGs, προσφέρουν μόνο ένα τρόπο αναγνώρισης της αξίας του παίχτη και αυτό γίνεται μέσω μίας κατάταξης του ranking. Επίσης, δεν είναι βασισμένα σε ένα συνεχές κόσμο στον οποίο ο παίχτης επιστρέφει όποτε ξεκινάει το παιχνίδι. Πιο συγκεκριμένα, ο παίχτης επιλέγει κάθε φορά τον χαρακτήρα που επιθυμεί μέσα από μία πολλή μεγάλη γκάμα που του προσφέρεται, με σκοπό να ταιριάξει με τους χαρακτήρες των συμπαικτών του, ώστε να κερδίσουν την αντίπαλη ομάδα. Το κάθε match συνήθως διαρκεί από είκοσι έως σαράντα πέντε λεπτά και στο τέλος αποδίδονται οι πόντοι ανάλογα με το αν υπήρξε νίκη ή ήττα. Ουσιαστικά, αυτό το είδος προέρχεται από τα strategy games και ένα από τα πρώτα της κατηγορίας του ήταν το DOTA (Defense of the Ancients), το οποίο ήταν ένα mod του Warcraft III.

Τα πιο διαδεδομένα παιχνίδια αυτής της κατηγορίας στην εποχή μας είναι το DOTA 2, το League of Legends και το Smite.

Το μεγαλύτερο επίτευγμα αυτού του είδους είναι η προώθηση του gaming στον επαγγελματικό τομέα, αφού πάρα πολλοί άνθρωποι κερδίζουν αξιόλογα χρηματικά ποσά παίζοντας τέτοια παιχνίδια.

Εικόνα 5.19 - **Στιγμιότυπο από το παιχνίδι** Smite

5.8 Indie Games

Τα independent games ή αλλιώς Indie είναι αυτά τα οποία δεν έχουν εκδοθεί από κάποια εταιρεία εκδόσεων παιχνιδιών, αλλά από μία μικρή ομάδα ανθρώπων ή ακόμα και από ένα μόνο άτομο. Αυτά τα άτομα είναι και οι δημιουργοί των παιχνιδιών. Είναι πολύ εύκολο στις μέρες μας κάποιος να φτιάξει ένα δικό του παιχνίδι χωρίς να χρειάζεται να βρει κάποιον για να το εκδώσει.

Τα indie games περιλαμβάνουν όλη την γκάμα των ειδών των video games, αφού μπορούν να είναι από action μέχρι strategy κ.λπ.

Για να φτιάξει κάποιος ένα τέτοιο παιχνίδι, αρκεί να έχει όρεξη και άπλετο χρόνο στην διάθεσή του, έτσι ώστε να το διαμορφώσει με τον τρόπο που ο ίδιος θέλει ή να συνεργαστεί με μερικά άτομα ακόμα που θα τον βοηθήσουν στην δημιουργία του.

Τα πιο γνωστά Indie παιχνίδια είναι το Stardew Valley, το Hotline Miami και το Banished, τα οποία είναι διαθέσιμα για αγορά μέσω του Steam.

Εικόνα 5.20 - **Στιγμιότυπο από το παιχνίδι** Stardew Valley

5.9 Serious Games

Στην εποχή μας, τα παιχνίδια εκτός από τον τομέα της διασκέδασης, χρησιμοποιούνται σε μεγάλο βαθμό σε τομείς όπως η εκπαίδευση, η ιατρική, η διαφήμιση και άλλοι. Έτσι αναπτύχθηκε η έννοια "Serious Games", δηλαδή σοβαρά παιχνίδια τα οποία δημιουργούνται με συγκεκριμένο σκοπό, συνήθως να βοηθήσουν να ξεπεραστεί κάποιο πρόβλημα. Για παράδειγμα στον τομέα της ιατρικής χρησιμοποιούνται παιχνίδια για να βοηθήσουν ασθενείς κατά την διάρκεια της αποθεραπείας και αποκατάστασης τους, όπου εμπεριέχονται διαδικασίες άκρως επαναληπτικές με αποτέλεσμα να γίνονται βαρετές και οι ίδιοι να μην τις εκτελούν. Έτσι, παρέχοντας διασκέδαση, ένα επίπεδο δυσκολίας και επιβράβευση μέσω ενός παιχνιδιού, οι ασθενείς είναι πολύ πιο πρόθυμοι να ολοκληρώσουν τις ασκήσεις της θεραπείας τους. Επίσης, κατά την διάρκεια μιας επίπονης θεραπείας ασθενών που πάσχουν από χρόνιες ασθένειες όπως π.χ. ο καρκίνος, η χρήση παιχνιδιών έχει αποδειχθεί ότι είναι ένα πολύ καλό ερέθισμα ώστε να νιώθουν λιγότερο τον πόνο. Στον τομέα της εκπαίδευσης, υπάρχουν πολλά διαθέσιμα παιχνίδια που προσφέρουν εκπαιδευτικό υλικό με στόχο την εκμάθηση. Όπως και στην ιατρική, έτσι και σε αυτό τον τομέα, παρέχοντας ποικίλα επίπεδα δυσκολίας και επιβράβευσης μέσω των παιχνιδιών, η εκμάθηση γίνεται κάτι το διασκεδαστικό και προκαλεί το ενδιαφέρον των μαθητών και όχι μόνο.

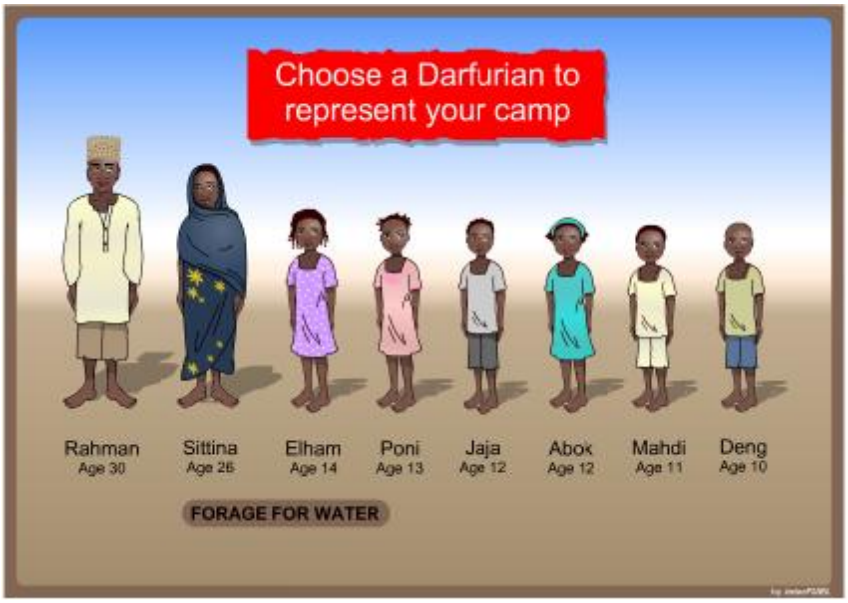
Το πανεπιστήμιο "University of Skövde" ορίζει τα Serious Games ως:

“The term "serious games" refers to games that engage the user, and contribute to the achievement of a defined purpose other than pure entertainment (whether or not the user is consciously aware of it). The additional purposes can, for example, be education, training, health care or marketing.”

(http://www.his.se/en/Prospective-student/education/Masters-Studies/Serious-Games/)

Παράδειγμα serious game αποτελεί το “Darfur is dying” το οποίο δημιουργήθηκε το 2006 με στόχο να προβάλει την ζωή των προσφύγων στην περιοχή Darfur στο Sudan. Οι παίχτες θα πρέπει να προσπαθήσουν να διατηρήσουν σωστή λειτουργία στο στρατόπεδο των προσφύγων σε περίπτωση επίθεσης από τα στρατεύματα των Janjaweed. Επίσης, μαθαίνουν πληροφορίες για την γενοκτονία που υπήρξε σε αυτή την περιοχή, καθώς και τρόπους που μπορούν να βοηθήσουν στην πραγματική ζωή, έτσι ώστε να σταματήσει η καταπάτηση των ανθρώπινων δικαιωμάτων εκεί (εικόνα 5.21 και 5.22).

Ακόμη ένα παράδειγμα αποτελεί το serious game “Re-Mission”. Αυτό δημιουργήθηκε το 2006 από την εταιρεία HopLab και μετά από λίγο καιρό δημιούργησε και το “Re-Mission 2”. Χρησιμοποιώντας επιστημονικές έρευνες, τα παιχνίδια αυτά δημιουργήθηκαν για να υποστηρίζουν τους νέους στον αγώνα τους κατά του καρκίνου δείχνοντάς τους την δύναμη και τον έλεγχο που μπορούν να έχουν, όπως επίσης τους ενθαρρύνει να επιμένουν στις θεραπευτικές διαδικασίες. Κάθε τέτοιο παιχνίδι, τοποθετεί τον παίχτη μέσα σε ένα εικονικό ανθρώπινο σώμα και του παρέχει όπλα και υπερδυνάμεις, έτσι ώστε με την βοήθεια του ανοσοποιητικού συστήματος, των αντιβιοτικών και της χημειοθεραπείας, να καταστρέψει τον καρκίνο μέσα σε αυτό το σώμα (εικόνα 5.23).



Εικόνα 5.21 - **Στιγμιότυπο από το παιχνίδι** Darfur is dying

Εικόνα 5.22 - **Στιγμιότυπο από το παιχνίδι** Darfur is dying

Εικόνα 5.23 - **Στιγμιότυπο από το παιχνίδι** Re-Mission

5.10 PC, Console **και** Mobile Games

Σε αυτό το σημείο είναι πολύ σημαντικό να αναφερθούν οι συσκευές που μπορούν να χρησιμοποιηθούν για να μπορέσει κάποιος να παίξει ένα παιχνίδι. Κάθε μια από αυτές έχει διαφορετικά πλεονεκτήματα και μειονεκτήματα, καθώς και διαφορετική χρήση. Οι τρεις κατηγορίες διαθέσιμων συσκευών στην αγορά είναι οι προσωπικοί ηλεκτρονικοί υπολογιστές ή αλλιώς PC, οι κονσόλες και τα κινητά ή παρόμοιες φορητές συσκευές, όπως για παράδειγμα tablet.

5.10.1 PC Games

Τα παιχνίδια που είναι διαθέσιμα για συσκευές ηλεκτρονικού υπολογιστή έχουν μεγαλύτερες δυνατότητες και περισσότερα πλεονεκτήματα όσον αφορά τις λειτουργίες και δυνατότητες χωρητικής επέκτασης για την δημιουργία καλύτερων γραφικών. Ο ηλεκτρονικός υπολογιστής είναι ένα μηχάνημα το οποίο μπορεί να διαθέσει πολλά διαφορετικά hardware όπου το κάθε ένα έχει διαφορετικές δυνατότητες και χρήσεις. Όλα αυτά μαζί, λοιπόν, αποτελούν ένα δυναμικό σύνολο που δίνει την δυνατότητα στον παίχτη να παίξει παιχνίδια στην υψηλότερη ανάλυση που είναι διαθέσιμη εκείνη τη χρονική στιγμή. Επίσης, το pc μπορεί να «αντέξει» πολλά και υψηλών απαιτήσεων software τα οποία λειτουργούν ταυτόχρονα για να ικανοποιήσουν τον παίχτη.

Ένα σημαντικό μειονέκτημα των ηλεκτρονικών υπολογιστών αφορά το κόστος του υλικού που είναι διαθέσιμο γι' αυτόν. Δηλαδή, αν κάποιος επιθυμεί να αγοράσει έναν υπολογιστή που να «σηκώνει» τα παιχνίδια τελευταίας τεχνολογίας σε πάρα πολύ υψηλή ανάλυση, τότε θα του κοστίσει πολύ ακριβά μόνο ένα κομμάτι hardware όπως για παράδειγμα είναι η κάρτα γραφικών ή ο επεξεργαστής.

5.10.2 Console Games

Το μεγάλο πλεονέκτημα στις κονσόλες είναι ότι αποτελούνται από μία κύρια συσκευή η οποία δεν χρειάζεται κάποιος να ψάξει ξεχωριστά τα υλικά της. Το μόνο που χρειάζεται για να παίξει ένα παιχνίδι σε αυτήν είναι ένα χειριστήριο και μία οθόνη.

Από την άλλη, αυτό αποτελεί και μειονέκτημα. Σε αντίθεση με τους ηλεκτρονικούς υπολογιστές που μπορούν να διαθέσουν είτε πληκτρολόγιο και ποντίκι, είτε χειριστήριο, οι κονσόλες διαθέτουν μόνο χειριστήριο. Αυτό μειώνει σε ορισμένα παιχνίδια, την ποικιλία των κουμπιών που μπορεί να πατήσει ο παίχτης και να εκτελέσει εντολές και λειτουργίες στο παιχνίδι.

Επιπλέον, όσον αφορά τα παιχνίδια της κονσόλας, δεν μπορούν να φτάσουν το υψηλό επίπεδο γραφικών και ποιότητας σε σχέση με τους ηλεκτρονικούς υπολογιστές, διότι περιορίζονται στο συγκεκριμένο hardware που διαθέτει η κάθε κονσόλα.

Στην κατηγορία αυτή ανήκουν και οι hand-held **κονσόλες** οι οποίες είναι οι Nintendo Gameboy, Sony PS Vita κλπ. Αυτές έχουν το πλεονέκτημα της φορητότητας, ο παίχτης μπορεί να τις κουβαλήσει μαζί του όπου θέλει, αλλά ούτε αυτές μπορούν να προσφέρουν τις υψηλότερες ποιότητες και αναλύσεις παιχνιδιών.

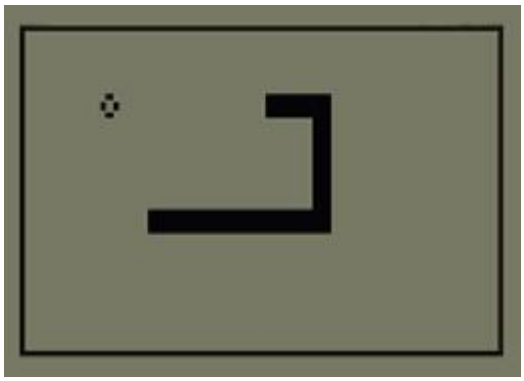
5.10.3 Mobile Games

Τα mobile παιχνίδια είναι πλέον πολύ διαδεδομένα γιατί είναι διαθέσιμα σε οποιοδήποτε tablet ή κινητό στις μέρες μας. Αυτές οι συσκευές χρησιμοποιούνται από το μεγαλύτερο μέρος του πληθυσμού και η κύρια λειτουργία τους είναι το δίκτυο. Με την εξέλιξη τους, μπορεί πλέον ο καθένας να παίξει ένα αξιοπρεπές παιχνίδι οπουδήποτε και οποτεδήποτε, καθώς περιμένει στην στάση του τρένου ή στην αίθουσα αναμονής του αεροδρομίου κλπ.

Σαφώς τα παιχνίδια αυτά δεν έχουν τις ίδιες απαιτήσεις και δυνατότητες που έχουν τα παιχνίδια κονσόλας και PC, όμως για το είδος τους έχουν αναπτυχθεί ραγδαία.

Το πρώτο γνωστότερο παιχνίδι κινητού ήταν το φιδάκι (εικόνα 5.23), μια δημιουργία της Nokia την δεκαετία του '70. Το φιδάκι ήταν ένα απλό παιχνίδι χωρίς ιδιαίτερα γραφικά και απαιτήσεις. Από το 2012 και μετά, το πιο δημοφιλές παιχνίδι κινητών ήταν το Angry Birds (εικόνα 5.24). Είναι προφανής η εξέλιξη που επιδέχτηκαν όσον αφορά το software και αυτό συνέβη λόγο της μεγάλης εξέλιξης του hardware των κινητών.

Μειονέκτημα των Mobile games είναι ότι από κάποιο σημείο και μετά όλα μοιάζουν μεταξύ τους και έχουν πολύ παρόμοιες λειτουργίες. Επίσης μειονέκτημα αποτελεί και η μικρή οθόνη των κινητών, καθώς και η περιορισμένη δυνατότητα επιλογής κουμπιών, αφού πλέον έχουν μόνο touch screen ή ελάχιστα διαθέσιμα πλήκτρα.



Εικόνα 5.24 - **Στιγμιότυπο από το** mobile game "Snake"

Εικόνα 5.25 - **Στιγμιότυπο από το** mobile game "Angry Birds"

<u>ΜΕΡΟΣ ΔΕΥΤΕΡΟ</u>: Η ΔΗΜΙΟΥΡΓΙΑ ΤΟΥ RABBIT RUNNER

6 Game Engines

Παλαιότερα δεν υπήρχε η δυνατότητα να φτιάξει ο καθένας ένα video game. Αυτά τα δημιουργούσαν μόνο μεγάλες ή μικρές εταιρείες με μεγάλα budget και με την βοήθεια πολλών ατόμων στα οποία οριζόταν διαφορετικός ρόλος. Στην εποχή μας πλέον, με την εξέλιξη της τεχνολογίας και την διάδοση του ίντερνετ σε τέτοιο βαθμό έτσι ώστε ο καθένας να μπορεί να έχει σπίτι του τουλάχιστον μία σύνδεση διαδικτύου, μπορεί ο κάθε ενδιαφερόμενος να δημιουργήσει ένα παιχνίδι από την αρχή είτε μόνος του, είτε με την βοήθεια μιας μικρής ομάδας ανθρώπων. Κύριο παράγοντα γι' αυτή τη δυνατότητα αποτελεί το σύνολο των δωρεάν ή χαμηλού κόστους game engines που είναι διαθέσιμες μέσω διαδικτύου στο ευρύ κοινό. Αυτές δίνουν την δυνατότητα στον δημιουργό να μπορέσει να φτιάξει το δικό του παιχνίδι από την αρχή με χρήση κάποιας γλώσσας προγραμματισμού, καθώς και να συνδέσει όλο το καλλιτεχνικό και σχεδιαστικό κομμάτι του παιχνιδιού του με τον κώδικα και να βγάλει ένα ολοκληρωμένο παιχνίδι έτοιμο για δημοσίευση.

Υπάρχουν πολλές μηχανές σχεδιασμού παιχνιδιών (game engines) στην αγορά. Οι πιο διαδεδομένες είναι οι Unity3D, LeadWerks Game Engine, Unreal Engine, CryEngine 3, HeroEngine, Rage Engine, Project Anarchy, GameSalad, GameMaker Studio, App Game Kit, Cocow2D. Εμείς θα ασχοληθούμε κυρίως με την σχεδιαστική πλατφόρμα Unity3D η οποία χρησιμοποιήθηκε για την δημιουργία του platform παιχνιδιού "Rabbit Runner" (https://unity3d.com/).

6.1 Unity3D Game Engine

Με την δωρεάν μηχανή Unity3D μπορεί κάποιος να δημιουργήσει είτε δισδιάστατα, είτε τρισδιάστατα παιχνίδια. Να συνδυάσει όμορφα γραφικά με δυνατά χαρακτηριστικά και όλα αυτά με απόλυτη ευκολία. Η πλατφόρμα αυτή είναι πολύ εύκολη στη χρήση, καθώς περιέχει μεγάλες βιβλιοθήκες εντολών και έτοιμων build-in συναρτήσεων που μπορούν να χρησιμοποιηθούν σε πάρα πολλές διαφορετικές λειτουργίες. Επιπλέον, διαθέτει τομέα Learning, όπου το προσωπικό της εταιρείας παρέχει διδακτικές ενότητες επίσης δωρεάν, έτσι ώστε ο οποιοσδήποτε να μπορέσει να μάθει καλύτερα την χρήση και την λειτουργία αυτής της μηχανής. Στην ίδια ενότητα υπάρχουν και διάφορα προγεγραμμένα βίντεο-tutorials τα οποία δείχνουν βήμα-βήμα την δημιουργία διάφορων μικρών παιχνιδιών, όπως και την ενότητα documentation όπου βρίσκεται κάθε πιθανή εντολή και συνάρτηση που μπορεί να χρησιμοποιηθεί καθώς και η επεξήγηση της λειτουργίας της.

Τα παιχνίδια που φτιάχνονται με την Unity3D είναι διαθέσιμα για τα περισσότερα λειτουργικά συστήματα όπως είναι τα Windows, Linux, Macintosh, Android, iOS κ.λπ.

Οι βασικές της λειτουργίες για να δημιουργηθεί ένα παιχνίδι παρέχονται δωρεάν. Υπάρχει όμως και η έκδοση Unity3D Plus και η Unity3D Pro, η οποία παρέχει ακόμα περισσότερες λειτουργίες με κάποιο μηνιαίο κόστος.

7 Το Storyboard του Rabbit Runner

Ιστορία

Ο χαρακτήρας είναι ένα λαγουδάκι όπου τρέχει από αριστερά προς τα δεξιά ασταμάτητα, με σκοπό να μαζέψει το αγαπημένο του φαγητό, τα καρότα. Πρέπει όμως να αποφύγει τα εμπόδια που εμφανίζονται στον δρόμο του και να προσέξει να μην πέσει στο κενό.

Χρηστικότητα

Το παιχνίδι ξεκινάει με το κεντρικό menu όπου έχει δύο βασικές επιλογές τύπου button functions. Την "play game" η οποία σε οδηγεί στο παιχνίδι και την "quit game" η οποία σε βγάζει εντελώς από την εφαρμογή. Επίσης, περιλαμβάνει μια επιλογή τύπου button, την "credits", η οποία αν επιλεχθεί δείχνει την οθόνη με τους δημιουργούς του παιχνιδιού, καθώς και ένα κουμπί το οποίο σε οδηγεί σε μία οθόνη που περιλαμβάνει τις βασικές πληροφορίες και οδηγίες του παιχνιδιού.

Αφού ο χρήστης επιλέξει να ξεκινήσει το παιχνίδι, τότε μεταφέρεται στην "scene 1" όπου είναι το περιβάλλον του παιχνιδιού. Το σκηνικό είναι βασισμένο σε 2D μηχανή και περιλαμβάνει τα εξής:

- Έναν βασικό χαρακτήρα ο οποίος έχει δύο αυτοματοποιημένες και δύο χειροκίνητες λειτουργίες.

  Οι αυτοματοποιημένες είναι:
  1. Τρέξιμο
  2. Σταδιακή αύξηση ταχύτητας

  Οι χειροκίνητες είναι:

  1. Άλμα
  2. Διπλό άλμα

- Την πλατφόρμα, η οποία θα δημιουργείται αυτόματα στο χώρο, με τυχαία θέση και μέγεθος και θα πολλαπλασιάζεται ανάλογα με την θέση της κάμερας.

- Τα εμπόδια, όπου είναι ένας κάκτος και τα κενά ανάμεσα στις πλατφόρμες, Οι θέσεις όπου δημιουργούνται οι κάκτοι είναι τυχαίες πάνω στην πλατφόρμα.

- Τα collectibles, όπου είναι αντικείμενα που μαζεύει ο χαρακτήρας με στόχο ένα high score

- Το score και high score που συνδέονται και αυξάνονται ανάλογα με τα collectibles που μαζεύει ο χαρακτήρας

- Ένα Pause Button το οποίο εμφανίζει το Pause Menu με τρεις επιλογές:
  1. Restart, όπου μπορεί ο παίχτης να επανεκκινήσει το παιχνίδι

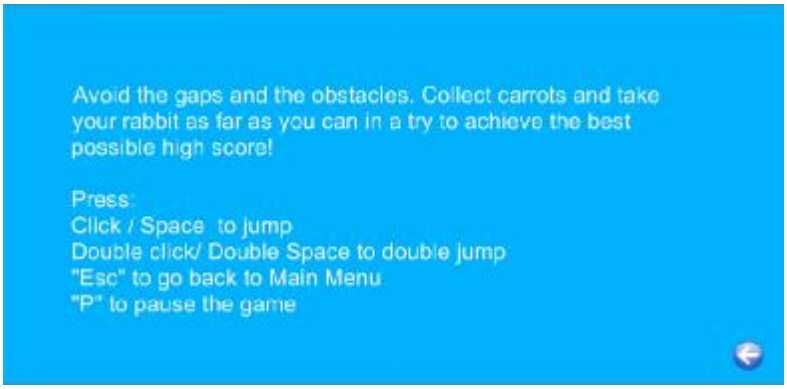
  2. Resume, όπου μπορεί να συνεχίσει από εκεί που έμεινε

3. Quit, όπου μπορείς να γυρίσεις στο αρχικό menu. Σε αυτήν εμπεριέχεται και μία οθόνη στην οποία δίνεται η δυνατότητα στον παίχτη να επιλέξει αν θέλει όντως να αποχωρήσει από το παιχνίδι, ή αν άλλαξε γνώμη.

- Ένα Death Menu, το οποίο εμφανίζεται την στιγμή που θα πεθάνει ο χαρακτήρας και παρέχει τις επιλογές:
    1. Restart, όπου μπορεί ο παίχτης να επανεκκινήσει το παιχνίδι
    2. Quit, η οποία λειτουργεί όπως η quit του Pause Menu

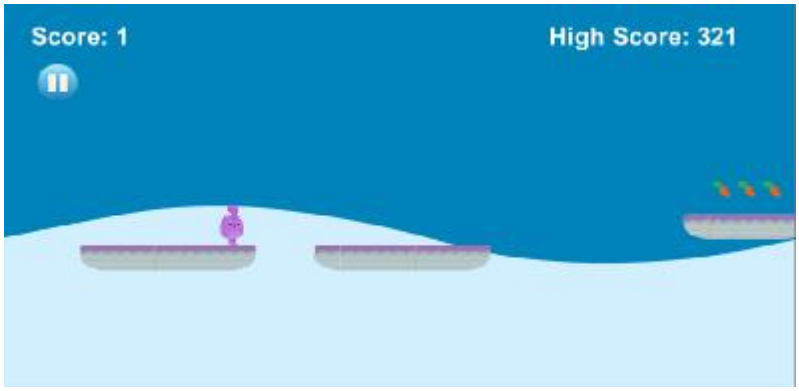
- Τους ήχους, οι οποίοι είναι τρεις ήχοι περιβάλλοντος (όταν πηδά, πεθαίνει και τρώει ο χαρακτήρας) και δύο ήχοι μενού (ένας στο αρχικό μενού και ένας κατά την διάρκεια του gameplay)

Εικόνα 7.1 - Rabbit Runner Main Menu



Εικόνα 7.2 - Rabbit Runner **Οθόνη Πληροφοριών και Οδηγιών**

Εικόνα 7.3 - Rabbit Runner **Οθόνη με** Credits



Εικόνα 7.4 - Rabbit Runner Gameplay

Εικόνα 7.5 - Rabbit Runner **Οθόνη** Pause Menu

Εικόνα 7.6 - Rabbit Runner **Οθόνη** Death Menu

Εικόνα 7.7 – Rabbit Runner **Οθόνη επιλογής εξόδου ή όχι**

8 Το Script του Rabbit Runner

Σε αυτό το κεφάλαιο αναλύεται ο κώδικας που χρησιμοποιήθηκε για την δημιουργία και την σωστή λειτουργία του παιχνιδιού Rabbit Runner, με ποιον τρόπο συνδέονται οι διαφορετικοί κώδικες μεταξύ τους και ποιες λειτουργίες περιέχουν σε αντιστοιχία με τα αντικείμενα του παιχνιδιού (game objects).

8.1 "Player Controller" Script

Σε αυτό το script[1] ενεργοποιούνται όλες οι λειτουργίες που έχουν κάποια σχέση με τον χαρακτήρα του παιχνιδιού, το λαγουδάκι.

Από την γραμμή 1 έως 59 αρχικοποιούνται όλες οι μεταβλητές που χρησιμοποιούνται μέσα στο script. Υπάρχουν δύο μεγάλες κατηγορίες μεταβλητών στην Unity3D. Αυτές είναι οι μεταβλητές public και private. Τις πρώτες, έχει ο δημιουργός την δυνατότητα να τις επεξεργαστεί μέσα στο πρόγραμμα του Unity χωρίς να χρειάζεται να μπει στο script, ενώ οι δεύτερες θα πρέπει να ορίζονται καθαρά μέσα στο κομμάτι του κώδικα. Βλέπουμε, επίσης, ότι υπάρχουν διαφορετικά είδη μεταβλητών. Αυτά είναι τα εξής:

- Float: Οι μεταβλητές που ορίζονται ως float παίρνουν δεκαδικές τιμές.
- Bool: Οι μεταβλητές που ορίζονται έτσι παίρνουν ως τιμή το 0 και το 1, όπου το 0 σημαίνει ψέμα και το 1 αληθές.
- Rigidbody2D: Έτσι ορίζεται η μεταβλητή ως rigidbody2D, ένα στοιχείο του Unity που επιτρέπει να χρησιμοποιηθούν δυνάμεις φυσικής πάνω στο αντικείμενο (game object) που τοποθετείται.
- LayerMask: Έτσι ορίζεται το αντικείμενο που είναι layerMask μέσα στο Unity. Ένα αντικείμενο δηλαδή που ορίζεται με κάποιο συγκεκριμένο όνομα ως layer.
- Transform: Αυτό δηλώνει είτε θέση, είτε περιστροφή, είτε κλιμάκωση. Επομένως οι μεταβλητές που ορίζονται έτσι, παίρνουν τιμές θέσης, περιστροφής και κλιμάκωσης.
- Animator: Έτσι ορίζεται το αντικείμενο που περιέχει animation και χρειάζεται να του επιτραπεί η επεξεργασία μέσω του script, έτσι ώστε να του δοθούν οι απαραίτητες εντολές που χρειάζονται για την δημιουργία του.
- GameManager: Έτσι δίνεται η δυνατότητα σε ένα αντικείμενο να έχει πρόσβαση σε άλλο script με το όνομα GameManager.
- AudioSource: Έτσι δηλώνονται τα αντικείμενα που είναι ήχοι.
- GameObject: Με αυτόν τον τρόπο δηλώνεται ότι αυτή η μεταβλητή αποτελεί κάποιο game object που υπάρχει στο project.
- String: Στην μεταβλητή αυτή μπορεί να δοθεί αλφαριθμητική τιμή

[1] οι γραμμές του script "PlayerController" φαίνονται στο Παράρτημα Ι

Από την 62 μέχρι την 89 φαίνεται το κομμάτι της συνάρτησης void Start() η οποία είναι μία build-in συνάρτηση της μηχανής Unity3D και σε αυτή ορίζεται οτιδήποτε έχει να κάνει με αρχικοποίηση, δηλαδή οι λειτουργίες που θέλει ο developer να συμβούν με το που ξεκινήσει το παιχνίδι. Αυτές συμβαίνουν μία φορά σε αυτό το σημείο, και δεν επαναλαμβάνονται σε κάθε frame του παιχνιδιού. Μέσα στη συνάρτηση αυτή ορίζονται αρχικά τα components που αντιστοιχούν στην κάθε μεταβλητή, όπως για παράδειγμα στην γραμμή 64 η μεταβλητή backgroundSound παίρνει το στοιχείο (component) ήχου του αντικειμένου με όνομα backgroundSound. Στις γραμμές από την 75 μέχρι την 82 ορίζονται οι τιμές των μεταβλητών που θα ισχύουν κάθε φορά που θα ξεκινάει το παιχνίδι από την αρχή. Τέλος, στην γραμμή 87 υπάρχει η εντολή που κάνει τον ήχο του background να παίζει.

Στις γραμμές 91-203 εμπεριέχεται η συνάρτηση void Update(). Όλες οι εντολές που περιλαμβάνονται σε αυτή την συνάρτηση εκτελούνται σε κάθε frame. Στις γραμμές 107-112 περιγράφεται το κομμάτι που συνδυάζει την ταχύτητα του χαρακτήρα καθώς προχωράει στον χώρο, η οποία αυξάνεται σταδιακά, ανάλογα με την απόσταση που διανύει ο χαρακτήρας. Από την 118 έως την 146 φαίνεται η συνθήκη όπου αν πατηθεί το πλήκτρο “space” του πληκτρολόγιου ή το αριστερό κλικ του ποντικιού, μόνο στην περίπτωση που ο χαρακτήρας ακουμπήσει το πάτωμα, δηλαδή τις πλατφόρμες, έχει την δυνατότητα να πηδήξει μία φορά. Επίσης, σε περίπτωση που πατάει σε κάποια από τις πλατφόρμες και έχει την δυνατότητα να πηδήξει δεύτερη φορά συνεχόμενα, με την προϋπόθεση ότι δεν μπορεί να πηδήξει περισσότερες από δύο φορές συνεχόμενα. Σε αυτό το κομμάτι ενεργοποιείται και το ηχητικό του πηδήματος. Στις γραμμές 148-159 περιγράφεται η συνθήκη όπου αν πατηθεί το πλήκτρο “space” και το αριστερό κλικ του ποντικιού, αλλά σε αυτή την περίπτωση κρατηθούν πατημένα, τότε δίνεται η δυνατότητα στον χαρακτήρα να πηδήξει αναλογικά με το πάτημα του κουμπιού. Δηλαδή, όσο ο παίχτης κρατάει πιεσμένο το κουμπί, τόσο ψηλότερα πηδάει ο χαρακτήρας, και αυτό συμβαίνει μέχρι να φτάσει ένα συγκεκριμένο ύψος σε σχέση με τον χρόνο. Έπειτα ξεκινάει να πέφτει, παρ’ όλο που ο παίχτης κρατάει πιεσμένο το πλήκτρο. Στο κομμάτι 162-171, φαίνεται η συνθήκη όπου συμβαίνει αν αφήσει ο παίχτης το κουμπί, τότε σταματάει να πηδάει ο χαρακτήρας. Στις γραμμές 174-193, περιγράφονται οι συνθήκες όπου αν ο χαρακτήρας πατάει σε έδαφος, τότε μπορεί να πηδήξει μέχρι δύο φορές συνεχόμενα, αν ο παίχτης πατήσει το πλήκτρο “p” τότε μεταφέρεται στο pause menu του παιχνιδιού και αν πατήσει το πλήκτρο “escape” τότε γυρίζει στο αρχικό μενού. Τέλος, στις 199 και 201 ορίζεται το animation που διαθέτει ο χαρακτήρας.

Τέλος, στις γραμμές 205-225 περιγράφεται η συνάρτηση void OnCollisionEnter2D() η οποία ελέγχει τις ενέργειες που θα γίνουν σε περίπτωση που το collider του αντικειμένου που περιέχει αυτό το script, σε αυτή την περίπτωση ο χαρακτήρας, έρθει σε επαφή με το collider κάποιου άλλου αντικειμένου μέσα στο project. Σε αυτή την περίπτωση, αν το άλλο αντικείμενο έχει ως tag την λέξη “killbox”, τότε καλείται η συνάρτηση RestartGame(), η οποία εμπεριέχεται στο script με όνομα “GameManager” και δίνει την δυνατότητα να επανεκκινηθεί το παιχνίδι. Επίσης, αρχικοποιούνται οι τιμές της ταχύτητας και της σταδιακής αύξησής της. Τέλος, ξεκινάει να παίζει το ηχητικό του θανάτου, αλλά σταματάει να παίζει ο ήχος του background.

8.2 "Camera Controller" Script

Σε αυτό το script[2] περιγράφονται οι ενέργειες που αφορούν την κίνηση της κάμερας στον χώρο.

Από την γραμμή 6 έως 16 αρχικοποιούνται οι μεταβλητές που χρησιμοποιούνται. Εδώ, φαίνονται ακόμα δύο είδη μεταβλητών, το Vector3, όπου ορίζεται ως τιμή τριών διαστάσεων και το PlayerController, όπου επιτρέπει την είσοδο σε διαφορετικό script το οποίο έχει το ίδιο όνομα.

Στις γραμμές 18-26 περιλαμβάνεται η συνάρτηση void Start() μέσα στην οποία ορίζεται το διαφορετικό script που θέλουμε να χρησιμοποιήσουμε, καθώς και αποθηκεύεται η τιμή της τελευταίας θέσης που βρέθηκε ο χαρακτήρας.

Τέλος, στις 28-39 περιγράφεται η συνάρτηση void Update(), όπου υπολογίζεται η απόσταση που πρέπει να διανύσει η κάμερα, η νέα θέση που θα έχει, καθώς και η θέση του χαρακτήρα στο τέλος κάθε frame.

8.3 "Platform Generator" Script

Σε αυτό το script[3] περιγράφονται οι λειτουργείες που αφορούν την δημιουργία των πλατφόρμων που πατάει ο χαρακτήρας και θεωρούνται ως πάτωμα, καθώς και την δημιουργία των καρότων που μαζεύει για να κερδίσει περισσότερους πόντους, όπως και τον εμποδίων, στην περίπτωση αυτή κάκτων, όπου δυσκολεύουν το επίπεδο του παιχνιδιού. Οι πλατφόρμες δημιουργούνται σε τυχαίες θέσεις όσο κινείται η κάμερα προς τα δεξιά και σε τυχαίο ύψος. Τα καρότα δημιουργούνται πάνω σε τυχαίες πλατφόρμες, όχι όμως σε όλες και οι κάκτοι το ίδιο.

Στις γραμμές 6-48 αρχικοποιούνται οι μεταβλητές που χρησιμοποιούνται σε αυτό το script. Τα καινούργια είδη μεταβλητών που παρουσιάζονται εδώ είναι τα εξής:

- int: οι μεταβλητές που ορίζονται ως int παίρνουν ακέραιες τιμές
- float[]: αποτελεί πίνακα τιμών με μεταβλητές που έχουν δεκαδικές τιμές
- ObjectPooler[]: αποτελεί πίνακα που δέχεται αντικείμενα από το script "ObjectPooler"
- CarrotGenerator: έτσι δίνεται η δυνατότητα σε ένα αντικείμενο να έχει πρόσβαση σε άλλο script με όνομα "CarrotGenerator"
- ObjectPooler: έτσι δίνεται η δυνατότητα σε ένα αντικείμενο να έχει πρόσβαση σε άλλο script με όνομα "ObjectPooler"

[2] Οι γραμμές του script "CameraController" φαίνονται στο παράρτημα II
[3] Οι γραμμές του script "PlatformGenerator" φαίνονται στο παράρτημα III

Στις γραμμές 52-72 περιλαμβάνεται η συνάρτηση void Start() στην οποία ορίζεται το μέγεθος και το αρχικό ύψος που θα έχει η κάθε πλατφόρμα, καθώς και η μεταβλητή που έχει πρόσβαση στο αντικείμενο το οποίο περιλαμβάνει το script με όνομα "CarrotGenerator".

Από την γραμμή 74 μέχρι την 146 περιγράφεται η συνάρτηση void Update(), όπου οι λειτουργίες της εκτελούνται σε κάθε frame και περιλαμβάνει τα εξής:

Στις γραμμές 79 έως 87 ορίζεται τυχαία η απόσταση, η θέση στο ύψος και το μέγεθος της κάθε πλατφόρμας.

Από την 89 μέχρι την 95 εκτελείται έλεγχος ο οποίος ορίζει τις πλατφόρμες με τέτοιο τρόπο έτσι ώστε να βρίσκονται μέσα στα όρια της κάμερας, οπότε σε περίπτωση που δημιουργούνται εκτός ορίων, τότε αλλάζει η τιμή του κάθε ύψους.

Στις γραμμές 100 και 142 ορίζεται η θέση της κάθε πλατφόρμας έτσι ώστε να αποφευχθούν τυχών προβλήματα επικάλυψης της μίας πλατφόρμας με την άλλη. Στην γραμμή 106 ορίζονται τα αντικείμενα που θα μπουν στον πίνακα ObjectPooler[].

Στις 108-111 ορίζεται η θέση και η περιστροφή της πλατφόρμας που θα δημιουργηθεί, καθώς και ενεργοποιείται έτσι ώστε να φαίνεται στην οθόνη.

Από την 113 εως την 121 δημιουργούνται τα καρότα ως αντικείμενα συλλογής σε τυχαίες πλατφόρμες κοντά στο κέντρο της κάθε μίας από αυτές.

Τέλος, στις 124 έως 138 δημιουργούνται οι κάκτοι σε τυχαία θέση πάνω στις πλατφόρμες. Οι πλατφόρμες που θα εμφανιστούν κάκτοι είναι επίσης τυχαίες.

8.4 "Object Destroyer" Script

Σε αυτό το script[4] περιγράφονται οι λειτουργείες που εκτελούνται έτσι ώστε να «καταστρέφονται» τα αντικείμενα που βρίσκονται εκτός εμβέλειας της κάμερας, δηλαδή εκτός οθόνης.

Στην γραμμή 9 αρχικοποιείται η μία μεταβλητή που χρειάζεται εδώ.

Στις γραμμές 13 με 17 περιλαμβάνεται η συνάρτηση void Start() στην οποία ορίζεται το αντικείμενο της μεταβλητής objectDestructionPoint.

Από την γραμμή 19 έως την 31 περιγράφεται η συνάρτηση void Update() στην οποία εκτελείται ο εξής έλεγχος: αν η θέση σε αυτό το frame είναι μικρότερη από την θέση της παραπάνω μεταβλητής, τότε απενεργοποιούνται όλα τα αντικείμενα.

[4] Οι γραμμές του script "Object Destroyer" φαίνονται στο παράρτημα IV

8.5 "Object Pooler" Script

Αυτό το script[5] χρησιμοποιείται για την δημιουργία πολλαπλών αντικειμένων με ίδια ή παρεμφερή χαρακτηριστικά. Τα αντικείμενα αυτά δημιουργούνται αυτόματα χωρίς να χρειάζεται να υπάρχουν πολλά αντίστοιχα αντικείμενα στο project, παρά μόνο ένα το οποίο θα χρησιμοποιηθεί ως πρότυπο κλωνοποίησης των υπόλοιπων. Αυτά τα πρότυπα ονομάζονται prefabs στην Unity3D.

Στις γραμμές 9 έως 19 αρχικοποιούνται οι μεταβλητές. Το είδος της μεταβλητής List<GameObject> αποτελεί λίστα στην οποία τοποθετούνται τα αντικείμενα που κλωνοποιήθηκαν.

Στις γραμμές 21-38 περιλαμβάνεται η συνάρτηση void Start() στην οποία εκτελείται η λειτουργία της δημιουργίας των κλώνων και στην συνέχεια τοποθετούνται στην παραπάνω λίστα. Επιπλέον, ορίζονται και ως ενεργά στον χώρο της κάμερας, ή αλλιώς στην οθόνη.

Στις γραμμές 45 έως 64 περιλαμβάνεται μία νέα συνάρτηση, η public GameObject GetPooledObject() η οποία μετράει τα αντικείμενα τα οποία έχουν δημιουργηθεί, ελέγχει αν υπάρχουν ανενεργά αντικείμενα και αν ναι, τότε τα ενεργοποιεί και τέλος εξασφαλίζει τουλάχιστον ένα ενεργό αντικείμενο στον χώρο, για κάθε περίπτωση που μπορεί να μην υπάρχουν άλλα.

8.6 "Game Manager" Script

Σε αυτό το script[6] περιλαμβάνονται ενέργειες που αφορούν γενικά την λειτουργία του παιχνιδιού όπως για παράδειγμα τι συμβαίνει όταν ο χαρακτήρας πεθάνει ή αν ο παίχτης ξεκινήσει το παιχνίδι από την αρχή.

Στις γραμμές 6-20 αρχικοποιούνται οι μεταβλητές. Η ObjectDestroyer[] αποτελεί πίνακα που δέχεται αντικείμενα από το script "ObjectDestroyer". Η ScoreManager και η DeathMenu επιτρέπουν την πρόσβαση στα script "ScoreManager" και "DeathMenu" αντίστοιχα.

Στις γραμμές 24-35 περιλαμβάνεται η συνάρτηση void Start() στην οποία ορίζονται οι αρχικές θέσεις του παίχτη και της πλατφόρμας, καθώς και ορίζεται το αντικείμενο που περιέχει το "ScoreManager" script.

Από την γραμμή 42 έως την 59 βρίσκεται η συνάρτηση void RestartGame() η οποία σε περίπτωση που κληθεί, σταματάει να μετράει score, εξαφανίσει τον χαρακτήρα από την οθόνη και ενεργοποιεί το μενού του θανάτου (Death Menu).

[5] Οι γραμμές του script "ObjectPooler" φαίνονται στο παράρτημα V
[6] Οι γραμμές του script "GameManager" φαίνονται στο παράρτημα VI

Τέλος, στις γραμμές 61-90 περιλαμβάνεται η συνάρτηση void Reset() η οποία σε περίπτωση που κληθεί, εξαφανίζει το μενού θανάτου, εξαφανίζει τις πλατφόρμες που δημιουργήθηκαν έως εκείνη τη στιγμή, επαναφέρει τις θέσεις του παίχτη και της πλατφόρμας στην αρχή, επανεμφανίζει τον χαρακτήρα και ξεκινάει να μετράει το score από την αρχή.

8.7 "Score Manager" Script

Σε αυτό το script[7] περιγράφονται οι λειτουργίες που αφορούν την καταμέτρηση του score και high score.

Στις γραμμές 7-17 αρχικοποιούνται οι μεταβλητές που χρησιμοποιήθηκαν. Η μεταβλητή τύπου text παίρνει χαρακτήρες ως τιμές.

Από την γραμμή 19 έως την 30 περιγράφεται η συνάρτηση void Start(), όπου αποθηκεύεται η τιμή του high score ακόμα και μετά το restart του παιχνιδιού.

Στις 32-69 περιλαμβάνεται η συνάρτηση void Update(), οι λειτουργίες της οποίας εκτελούνται σε κάθε frame. Στο κομμάτι 35 έως 42 εκτελείται έλεγχος ο οποίος αυξάνει το score ανάλογα με τον χρόνο που διαρκεί το κάθε frame. Στις 44-50 ελέγχεται αν το score γίνει μεγαλύτερο από το high score, όπου τότε το δεύτερο παίρνει την τιμή του πρώτου. Επιπλέον, ορίζονται τα κείμενα και οι τιμές που αντιστοιχούν στο score και high score που θα εμφανίζονται στην οθόνη.

Τέλος, στις γραμμές 73-76 περιλαμβάνεται η συνάρτηση void AddScore(int pointsToAdd) η οποία χρησιμοποιείται για την πρόσθεση των πόντων που θα αντιστοιχούν στο score του παιχνιδιού. Με αυτόν τον τρόπο, μπορεί να χρησιμοποιηθεί σε διαφορετικά scripts χωρίς να χρειάζεται να επαναλαμβάνεται η δημιουργία της. Το μόνο που χρειάζεται είναι να κληθεί αυτή από το εκάστοτε script.

8.8 "Pick Up Points" Script

Σε αυτό το script[8] περιγράφονται οι λειτουργίες και ο τρόπος καταγραφής πόντων στο score του παιχνιδιού. Το script αυτό συνδέεται άμεσα με το script "ScoreManager".

Στις γραμμές 6-12 αρχικοποιούνται οι μεταβλητές που χρησιμοποιούνται. Η μεταβλητή ScoreManager δίνει την δυνατότητα στο αντικείμενο να έχει πρόσβαση στο script "ScoreManager".

[7] Οι γραμμές του script "ScoreManager" φαίνονται στο παράρτημα VII
[8] Οι γραμμές του script "pickUpPoints" φαίνονται στο παράρτημα VIII

Από την γραμμή 15 έως την 23 περιλαμβάνεται η συνάρτηση void Start() στην οποία ορίζεται το αντικείμενο που περιέχει το script "ScoreManager" ως στοιχείο (component), καθώς και ο ήχος που θα αντιστοιχεί στην μεταβλητή "eatSound".

Τέλος, στις γραμμές 30-61 περιγράφεται η συνάρτηση void OnTriggerEnter2D(Collider2D other) η οποία είναι μία build-in συνάρτηση φτιαγμένη από την Unity3D και ορίζει τι θα γίνει σε περίπτωση που το λεγόμενο trigger του αντικειμένου που περιέχει αυτό το script ως στοιχείο (component), έρθει σε επαφή με τον collider ενός άλλου. Η λειτουργία trigger ενός αντικειμένου, του δίνει την δυνατότητα να μπορεί να το διαπεράσει ένα άλλο αντικείμενο όταν είναι ενεργοποιημένη. Στις γραμμές 36-58 ελέγχεται αν το όνομα του άλλου αντικειμένου είναι ορισμένο ως "Player". Αν ναι, τότε χρησιμοποιείται η συνάρτηση AddScore του script "ScoreManager" για να προστεθούν οι πόντοι στο score, εξαφανίζεται το αντικείμενο που περιέχει αυτό το script (σε αυτήν την περίπτωση, είναι τα καρότα) και τέλος, εκτελείται έλεγχος που αφορά τον ήχο που θα παίζει την ώρα αυτή.

8.9 "Carrot Generator" Script

Σε αυτό το script[9] περιγράφονται οι λειτουργίες που εκτελούνται για την αυτόματη δημιουργία των καρότων μέσα στο παιχνίδι.

Στις γραμμές 6-8 αρχικοποιούνται οι μεταβλητές.

Στις γραμμές 12-25 περιλαμβάνεται η συνάρτηση void SpawnCarrots(Vector 3 startPosition) η οποία σε περίπτωση που κληθεί, δημιουργεί τρία πρότυπα καρότων τα οποία έχουν ίδια απόσταση μεταξύ τους και τοποθετούνται πάνω στις πλατφόρμες με την βοήθεια του script "PlatformGenerator" που περιγράφεται πιο πάνω (βλ. ενότητα 8.3).

8.10 "Main Menu" Script

Σε αυτό το script[10] περιγράφονται όλες οι λειτουργίες που εκτελούνται στο κεντρικό μενού του παιχνιδιού.

Στις γραμμές 6-11 αρχικοποιούνται οι μεταβλητές που χρησιμοποιούνται.

Από την γραμμή 14 έως την 21 περιγράφεται η συνάρτηση void Update() στην οποία εκτελείται έλεγχος όπου αν πατηθεί το πλήκτρο escape (esc) του πληκτρολογίου, τότε θα απενεργοποιηθούν οι οθόνες των πληροφοριών (εικόνα 7.2) ή των credits (εικόνα 7.3). Αυτό θα συμβεί μόνο στην περίπτωση που είναι ενεργοποιημένες.

[9] Οι γραμμές του script "CarrotGenerator" φαίνονται στο παράρτημα IX
[10] Οι γραμμές του script "MainMenu" φαίνονται στο παράρτημα X

Στις γραμμές 23-26 περιγράφεται η συνάρτηση void PlayGame() η οποία σε περίπτωση που κληθεί, μεταβιβάζει τον παίχτη στην σκηνή όπου ξεκινάει το παιχνίδι.

Στις γραμμές 28-31 περιγράφεται η συνάρτηση void QuitGame() η οποία όταν κληθεί, εκτελεί έξοδο από το παιχνίδι.

Στις γραμμές 33-36 περιγράφεται η συνάρτηση void Credits() η οποία όταν κληθεί ενεργοποιεί την οθόνη των credits.

Στις γραμμές 38-41 περιγράφεται η συνάρτηση void infoGame() η οποία όταν κληθεί ενεργοποιεί την οθόνη των πληροφοριών και οδηγιών.

Τέλος, στις γραμμές 43-47 περιγράφεται η συνάρτηση void arrowBack() η οποία καλείται όταν ο παίχτης πατήσει το πίσω βελάκι που υπάρχει στις οθόνες πληροφοριών και credits και τις απενεργοποιεί, ανάλογα με το ποια είναι ενεργή εκείνη τη στιγμή.

8.11 "Death Menu" Script

Σε αυτό το script [11] περιγράφονται οι λειτουργίες που εκτελούνται όταν ο χαρακτήρας του παιχνιδιού πεθαίνει ή χάσει.

Από την γραμμή 6 έως την 11 αρχικοποιούνται οι μεταβλητές που χρησιμοποιούνται εδώ.

Στις γραμμές 14-23 περιγράφεται η συνάρτηση void RestartGame() στην οποία ορίζεται ο ήχος του background, εκτελείται η συνάρτηση Reset() του script "GameManager" και έπειτα ενεργοποιείται ο ήχος του background από την αρχή.

Τέλος, στις γραμμές 25-29 περιγράφεται η συνάρτηση areSure() η οποία όταν κληθεί, ενεργοποιεί το μενού όπου ρωτάει τον παίχτη αν είναι σίγουρος για την επιλογή του (εικόνα 7.7) και απενεργοποιείται το μενού του θανάτου (εικόνα 7.6).

8.12 "Pause Menu" Script

Σε αυτό το script[12] περιγράφονται οι λειτουργίες που εκτελούνται στο pause menu (εικόνα 6.5).

Στις γραμμές 6-13 αρχικοποιούνται οι μεταβλητές που χρησιμοποιούνται.

Στις γραμμές 15-18 περιλαμβάνεται η συνάρτηση void Start() στην οποία ορίζεται το αντικείμενο του ήχου για το background.

[11] Οι γραμμές του script "DeathMenu" φαίνονται στο παράρτημα XI
[12] Οι γραμμές του script "PauseMenu" φαίνονται στο παράρτημα XII

Στις γραμμές 24-29 περιγράφεται η συνάρτηση void PauseGame() η οποία όταν κληθεί, σταματάει τον χρόνο του παιχνιδιού έτσι ώστε να «παγώσει» στο σημείο που βρίσκεται, ενεργοποιεί την οθόνη pause (εικόνα 7.5) και βάζει σε παύση τον ήχο του background.

Στις γραμμές 31-36 περιγράφεται η συνάρτηση void ResumeGame() η οποία καλείται όταν ο παίχτης επιλέξει το κουμπί "Resume". Αυτή συνεχίζει το παιχνίδι από εκεί που έμεινε πριν «παγώσει», απενεργοποιεί την οθόνη του pause menu και συνεχίζει να παίζει τον ήχο του background.

Στις γραμμές 39-50 περιγράφεται η συνάρτηση void RestartGame() η οποία καλείται όταν ο παίχτης επιλέξει το κουμπί "Restart". Αυτή επαναφέρει το παιχνίδι σε κανονικούς ρυθμούς, απενεργοποιεί την οθόνη του pause menu, χρησιμοποιεί την συνάρτηση Reset() από το script "GameManager", και παίζει από την αρχή τον ήχο του background.

Τέλος, στις γραμμές 52-57 περιγράφεται η συνάρτηση areSure() η οποία καλείται όταν ο παίχτης επιλέξει το κουμπί "Quit", και του δίνεται η δυνατότητα να επιλέξει αν είναι σίγουρος πως θέλει να κάνει έξοδο από το παιχνίδι. Αυτή, απενεργοποιεί την οθόνη του Pause menu και ενεργοποιεί την οθόνη επιλογής εξόδου (εικόνα 7.7).

8.13 "Are Sure" Script

Σε αυτό το script[13] περιγράφονται οι ενέργειες που εκτελούνται στο μενού επιλογής εξόδου ή όχι (εικόνα 7.7).

Στις γραμμές 6-13 αρχικοποιούνται οι μεταβλητές που χρησιμοποιούνται.

Στις γραμμές 16-20 περιγράφεται η συνάρτηση void noSureDeath() η οποία εκτελείται όταν ο παίχτης επιλέξει το κουμπί "no". Αυτή απενεργοποιεί το μενού επιλογής, και ενεργοποιεί το μενού του θανάτου (εικόνα 7.6).

Στις γραμμές 22-26 περιγράφεται η συνάρτηση void yesSure() η οποία εκτελείται όταν ο παίχτης επιλέξει το κουμπί "yes". Αυτή επαναφέρει τον χρόνο στο κανονικό επίπεδο και μεταφέρεται στην οθόνη του αρχικού μενού (εικόνα 7.1).

Τέλος, στις γραμμές 28-32 περιγράφεται η συνάρτηση void noSurePause() η οποία εκτελείται όταν ο παίχτης επιλέξει το κουμπί "no". Αυτή απενεργοποιεί το μενού επιλογής, και ενεργοποιεί το μενού του pause (εικόνα 7.5).

[13] Οι γραμμές του script "AreSure" φαίνονται στο παράρτημα XIII

9 Ερωτηματολόγιο

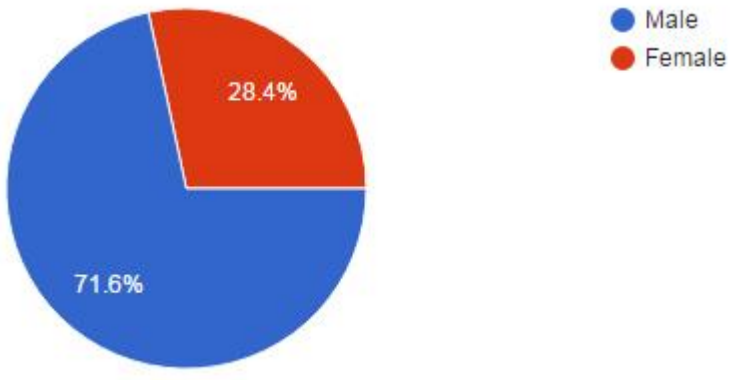
Το ερωτηματολόγιο[14] αυτό χρησιμοποιήθηκε για έρευνα όσον αφορά το κοινό που παίζει video games. Η έρευνα αυτή ξεκίνησε 24 Ιουλίου 2016 και έληξε 10 Οκτωβρίου 2016. Απαντήθηκε από 148 άτομα από Ελλάδα και εξωτερικό, μέσω των social media αλλά και της σελίδας του ιδρύματος Πληροφορικής και ΜΜΕ του ΤΕΙ Δυτικής Ελλάδας.

Σκοπός του ερωτηματολογίου, είναι να δείξει σε κάποιον νέο game developer τις προτιμήσεις και τις τάσεις του κοινού που ασχολείται ιδιαίτερα με τα video games. Τα αποτελέσματα αυτού μπορούν να του δώσουν την ευχέρεια να αποφασίσει τι είδος παιχνιδιού θα φτιάξει, σε ποιες ηλικίες θα απευθύνεται και για ποια συσκευή, ποια game engine είναι δημοφιλέστερη ώστε να χρησιμοποιήσει αυτήν και γενικότερα όλα όσα χρειάζεται να γνωρίζει κάποιος, ώστε να μπορέσει να αποφασίσει τι παιχνίδι θα δημιουργήσει.

9.1 What gender are you?

Η ερώτηση αυτή αφορά το γένος του πληθυσμού που απάντησαν. Όπως φαίνεται στο σχεδιάγραμμα 9.1, το 71,6% είναι άντρες και το υπόλοιπο 28,4% γυναίκες.

Επομένως βλέπουμε ότι το ενδιαφέρων του αντρικού πληθυσμού για τα video games είναι μεγαλύτερο από αυτό των γυναικών.

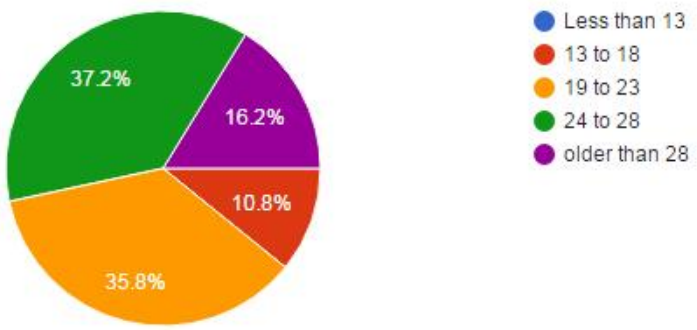


Σχήμα 9.1 – **Σχεδιάγραμμα που αφορά το γένος των ερωτηθέντων**

[14] Βλ. Παράρτημα XIV

9.2 How old are you?

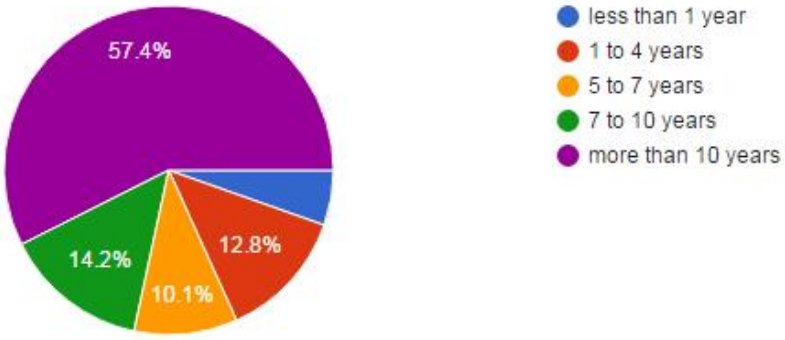
Σε αυτήν την ερώτηση το κοινό είχε την επιλογή να διαλέξει ένα εύρος ηλικιών μέσα στο οποίο θα έπρεπε να βρίσκεται και η δική του. Στο σχεδιάγραμμα 9.2, παρατηρούμε ότι το μεγαλύτερο μέρος των ερωτηθέντων είχαν ηλικία από 24 έως 28 ετών με 37.2%, το αμέσως επόμενο από 19 έως 23 με 35,8%, το 16,2% είχε ηλικία μεγαλύτερη των 28 ετών και τέλος το υπόλοιπο 10,8% το καταλαμβάνουν οι ηλικίες από 13 έως 18 ετών. Παρατηρούμε ότι κανένας δεν υπήρξε κάτω από 13 ετών.



Σχήμα 9.2 - **Σχεδιάγραμμα που αφορά την ηλικία των ερωτηθέντων**

9.3 How many years do you play video games?

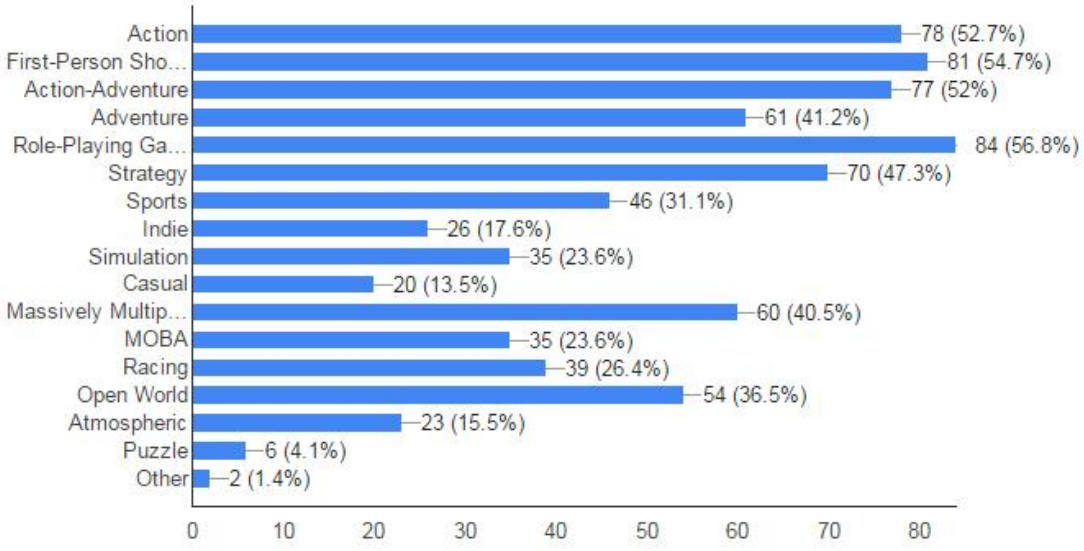
Η ερώτηση αυτή αφορά το πόσα χρόνια εμπειρίας έχουν οι ερωτηθέντες πάνω στα παιχνίδια. Πόσα χρόνια δηλαδή είναι ενεργοί στον χώρο των games. Το 57,4% απάντησε πως παίζει πάνω από 10 χρόνια, το 14,2% από 7 μέχρι 10 χρόνια, το 12,8% από 1 έως 4, το 10,1% από 5 έως 7 χρόνια και το τελευταίο 5,5% λιγότερο από 1 χρόνο (σχεδιάγραμμα 9.3).



Σχήμα 9.3 - **Σχεδιάγραμμα που αφορά την εμπειρία των ερωτηθέντων**

9.4 Which genre of video games do you play?

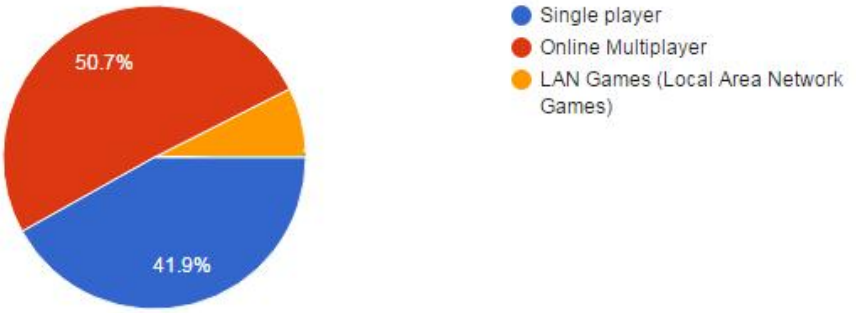
Αυτή η ερώτηση αφορά τα πιο δημοφιλή είδη των video games που επέλεξαν οι ερωτηθέντες. Στο σχεδιάγραμμα 9.4 φαίνεται ότι τα μεγαλύτερα ποσοστά κατέχουν τα Role-Playing-Games, First-Person Shooter, Action, Action-Adventure και Strategy games



Σχήμα 9.4 - **Σχεδιάγραμμα που αφορά τα δημοφιλέστερα είδη των** video games

9.5 Which type of video games do you prefer?

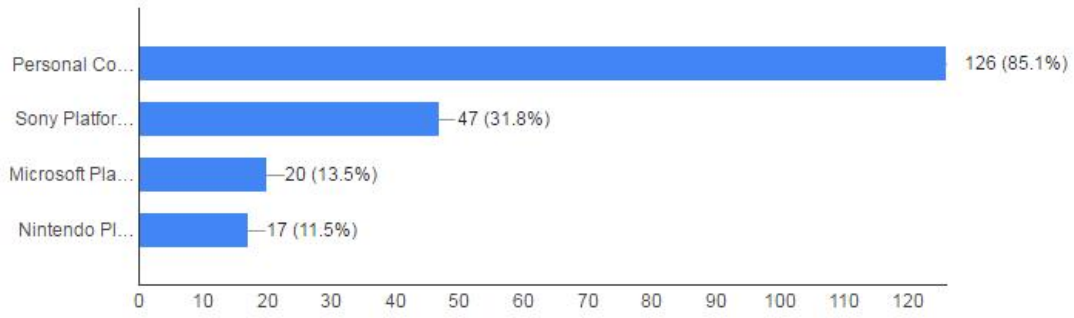
Η ερώτηση αυτή αφορά την προτίμηση που έδειξαν οι ερωτηθέντες στον τύπο των video games. Στο σχεδιάγραμμα 9.5 φαίνεται ότι οι περισσότεροι προτιμούν το online multiplayer με ποσοστό που φτάνει το 50,7%. Αρκετά μεγάλο μέρος του πλήθους επέλεξε το single player με ποσοστό 41,9% και το μικρότερο μέρος προτιμά τα LAN Games με ποσοστό 7,4%.



Σχήμα 9.5 - **Σχεδιάγραμμα που αφορά την προτίμηση των ερωτηθέντων στον τύπο των** games

9.6 In which platform do you prefer to play video games?

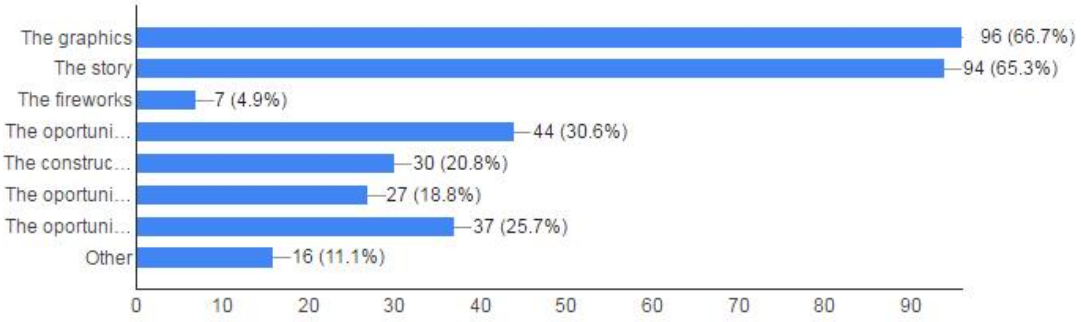
Με αυτήν την ερώτηση φαίνεται η προτίμηση της συσκευής που θέλουν οι χρήστες να διαθέτουν για να παίζουν τα παιχνίδια τους. Η πλειοψηφία των ερωτηθέντων προτιμούν τον δικό τους προσωπικό ηλεκτρονικό υπολογιστή με ποσοστό 85,1%. Το 31,8% δείχνει προτίμηση στις κονσόλες της Sony αφήνοντας το 13,5% να προτιμά τις κονσόλες της Microsoft, και το υπόλοιπο 11,5% αυτές τις Nintendo (σχεδιάγραμμα 9.6).



Σχήμα 9.6 - **Σχεδιάγραμμα που αφορά την προτίμηση πλατφόρμας από τους ερωτηθέντες**

9.7 What do you like most about video games?

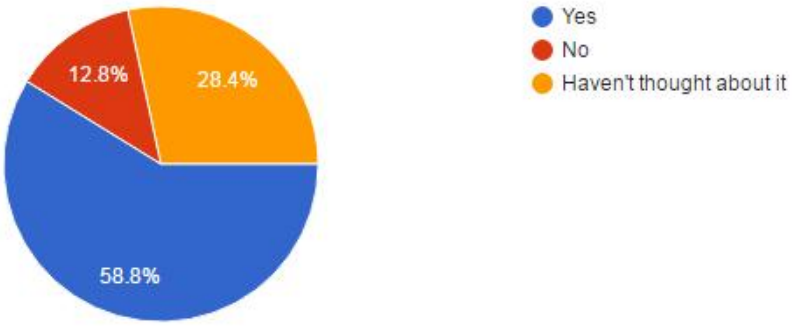
Στην ερώτηση αυτή φαίνεται ποιο από τα διαθέσιμα χαρακτηριστικά των παιχνιδιών κινεί το ενδιαφέρον των ερωτηθέντων. Το μεγαλύτερο ποσοστό απάντησε ότι του αρέσουν τα γραφικά ενός παιχνιδιού. Επίσης μεγάλο ποσοστό κέρδισε και η ιστορία που περιέχει ένα παιχνίδι (Σχεδιάγραμμα 9.7).



Σχήμα 9.7 - **Σχεδιάγραμμα που δείχνει την προτίμηση των ερωτηθέντων στα χαρακτηριστικά των** games

9.8 Would you like the idea of making your own video games?

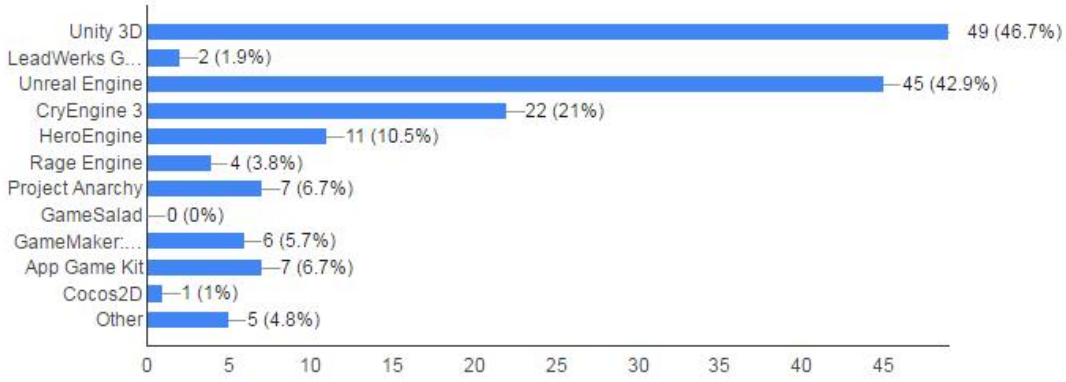
Με αυτή την ερώτηση φαίνεται αν το κοινό που λατρεύει να παίζει τα video games, έχει ποτέ σκεφτεί την ιδέα να φτιάξει και το δικό του παιχνίδι. Στο σχεδιάγραμμα 9,8 βλέπουμε ότι το μεγαλύτερο ποσοστό απαντάει «ναι» με 58,8%. Το 28,4% δεν έχει ασχοληθεί με την ιδέα και το υπόλοιπο 12,8% δεν θα ήθελε.



Σχήμα 9.8 - **Σχεδιάγραμμα που αφορά το αν θα ήθελαν οι ερωτηθέντες να φτιάξουν το δικό τους παιχνίδι**

9.9 If yes, which game engine would you prefer?

Στην περίπτωση που η προηγούμενη ερώτηση απαντήθηκε θετικά, τότε αυτή η ερώτηση αφορά τους πιο ειδικευμένους στον χώρο του game development και οι ερωτηθέντες επέλεξαν την μία ή περισσότερες game engines που θέλουν να χρησιμοποιούν. Το μεγαλύτερο ποσοστό ξεχωρίζει την Unity3D με 46,7% και το αμέσως επόμενο μεγάλο ποσοστό είναι το 42,9% το οποίο προτιμά την Unreal Engine. Το 21% κατακτήθηκε από την CryEngine 3 (σχεδιάγραμμα 9.9).



Σχήμα 9.9 - **Σχεδιάγραμμα που δείχνει την προτίμηση των ερωτηθέντων στις** game engines

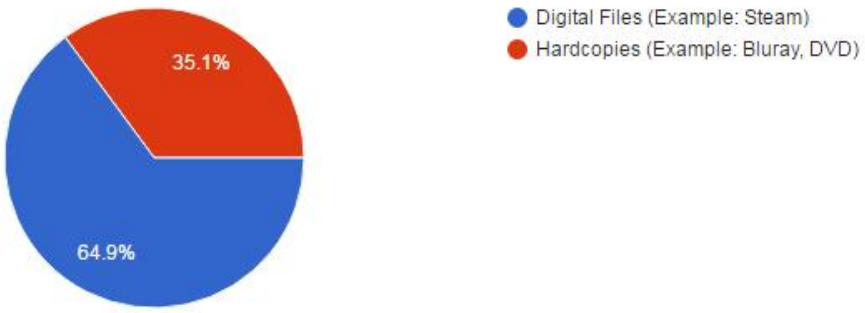
9.10 Which games do you play?

Σε αυτή την ερώτηση το κοινό φάνηκε πολύ πρόθυμο να απαντήσει με πλήθος τίτλων παιχνιδιών. Ανάμεσα στους τίτλους που αναφέρθηκαν, επιλέχθηκαν οι πέντε πιο δημοφιλείς οι οποίοι είναι οι εξής:

- League of Legends
- Assassin's Creed Series
- World of Warcraft
- Total War Series
- The Elder Scrolls Series

9.11 What type of video games do you prefer to own?

Αυτή η ερώτηση αφορά την προτίμηση που έχουν οι ερωτηθέντες σε σχέση με το αν το video game που διαθέτουν θα είναι digital αρχείο ή hardcopy DVD/Bluray. Στο σχεδιάγραμμα 9.10 φαίνεται ότι το μεγαλύτερο ποσοστό προτιμάει τα πρώτα με 64,9%, ενώ το υπόλοιπο 35,1% θέλει τα δεύτερα.



Σχήμα 9.10 - **Σχεδιάγραμμα που αφορά τα** digital games **και τα** hardcopies

9.12 **Συμπεράσματα**

Μετά την διεκπεραίωση αυτής της έρευνας, τα συμπεράσματα που προκύπτουν είναι πως το μεγαλύτερο μέρος του πληθυσμού των gamers κυριαρχεί το αρσενικό φύλο και προτιμάει να παίζει online multiplayer παιχνίδια με τα RPGs και τα FPS να κυριαρχούν. Διαθέτει εμπειρία στον χώρο αυτό πάνω από δέκα χρόνια και ανήκει στις ηλικίες από 24 έως 28. Επίσης, περισσότερο δημοφιλή είναι τα digital αρχεία, με μεγαλύτερη προτίμηση τον ηλεκτρονικό υπολογιστή ως μέσω και το μεγαλύτερο ενδιαφέρον όσον αφορά τα video

games προσφέρουν τα γραφικά και η ιστορία του. Τέλος, οι πιο εξειδικευμένοι προτιμούν να δημιουργούν τα παιχνίδια τους με την μηχανή Unity3D.

Επομένως, σε περίπτωση που ένας νέος δημιουργός παιχνιδιών θέλει να φτιάξει το δικό του παιχνίδι, πρέπει να λάβει υπόψιν του όλες αυτές τις πληροφορίες έτσι ώστε να καθοδηγηθεί και να αποφασίσει τι παιχνίδι θα φτιάξει βάσει τις προτιμήσεις του κοινού.

10 Συμπεράσματα

Αυτή η πτυχιακή αφορά την δημιουργία ενός παιχνιδιού με την χρήση της πλατφόρμας Unity3D. Αρχικά, στο πρώτο μέρος, αναλύεται το θεωρητικό κομμάτι των video games, οι λειτουργίες που εμπεριέχει, τα βήματα που ακολουθεί ένας δημιουργός παιχνιδιών, καθώς και τα εμπόδια που θα πρέπει να αντιμετωπίσει. Αναφέρεται στα περιεχόμενα-κλειδιά ενός παιχνιδιού, στη κύρια δομή του, στα στάδια του σχεδιασμού, όπως επίσης περιλαμβάνει τους ρόλους και τα καθήκοντα που έχουν οι σχεδιαστικές ομάδες. Περιγράφεται αναλυτικά τι είναι ο κόσμος των video games, ποιος ο σκοπός και οι διαστάσεις του. Επιπλέον, περιλαμβάνονται τα είδη των video games, καθώς και οι διαφορετικές πλατφόρμες που μπορούν να χρησιμοποιηθούν για να παίξει κάποιος αυτά τα games.

Το δεύτερο κομμάτι της πτυχιακής, αρχικά αναφέρεται στις μηχανές δημιουργίας παιχνιδιών και έπειτα αναλύεται το κομμάτι δημιουργίας του παιχνιδιού Rabbit Runner. Επεξηγούνται αναλυτικά οι κώδικες (scripts) που δημιουργήθηκαν για την σωστή λειτουργία του παιχνιδιού αυτού. Τέλος, αναφέρεται στο ερωτηματολόγιο που πρέπει να λάβει καθένας νέος δημιουργός παιχνιδιών υπόψιν του πριν ξεκινήσει να δημιουργεί το παιχνίδι του.

10.1 Μελλοντικές επεκτάσεις της εργασίας

Το παιχνίδι Rabbit Runner αποτελείται από κάποια δομικά στοιχεία και συγκεκριμένες λειτουργίες. Για την μελλοντική του χρήση ή και την δημοσίευσή του σε κάποιο store, θα μπορούσαν να προστεθούν περαιτέρω λειτουργίες έτσι ώστε να γίνει ακόμα πιο ενδιαφέρον. Για παράδειγμα, όταν ο παίχτης φτάσει κάποιο συγκεκριμένο Score, τότε να αλλάζει η πίστα και να έχει διαφορετικά textures και γραφικά, καθώς επίσης να αλλάζει το επίπεδο δυσκολίας. Αυτό μπορεί να γίνει με την δημιουργία κάποιον κινούμενων και όχι στατικών εχθρών οι οποίοι καθώς προχωράει ο χαρακτήρας, να έρχονται οι εχθροί από δεξιά και να κατευθύνονται προς αυτόν. Έτσι θα πρέπει να πηδήξει ο χαρακτήρας και να μπορέσει να τους αποφύγει. Επίσης, καθώς ανεβαίνει η δυσκολία, θα δημιουργούνται power-ups τα οποία καθώς τα συλλέγει ο χαρακτήρας θα μπορεί να έχει και διαφορετικές ικανότητες, όπως για παράδειγμα να μπορεί να παραμείνει στον αέρα περισσότερο χρονικό διάστημα ή και να μπορεί να χαμηλώσει την ταχύτητά του για συγκεκριμένο χρονικό διάστημα. Τέλος, θα μπορούσαν να υπάρχουν in-app purchases τα οποία όταν αγοράσει κάποιος θα του δίνει έξτρα προνόμια όπως για παράδειγμα αφαίρεση πιθανών διαφημίσεων, μείωση ή αύξηση του επιπέδου δυσκολίας κλπ.

Τέτοιες και παρόμοιες αλλαγές, κινούν το ενδιαφέρον των παιχτών και αυξάνουν την πιθανότητα να αγοράσει κάποιος το παιχνίδι.

Παράρτημα I: “Player Controller” Script

PlayerController.cs

PlayerController ▸ No selection

```
1  using UnityEngine;
2  using System.Collections;
3
4  public class PlayerController : MonoBehaviour {
5
6      public float moveSpeed;
7
8      //prepei na ginetai restart
9      //o xronos otan pethainei o player
10     private float moveSpeedStore;
11     private float speedMilestoneCountStore;
12     private float speedIncreaseMilestoneStore;
13
14     //gia na afksanetai i taxuthta
15     //stadiaka kathos proxwraei
16     public float speedMultiplier;
17     public float speedIncreaseMilestone;
18     private float speedMilestoneCount;
19
20     public float jumpForce;
21
22     //to make the jumping smooth
23     public float jumpTime;
24     private float jumpTimeCounter;
25
26     //gia na min pidaei kai otan fugei
27     //apo to telow tis platformas
28     private bool stoppedJumping;
29
30     //gia doublejump
31     private bool canDoubleJump;
32
33     // I use the name of the component
34     //that I want to interact with
35     private Rigidbody2D myRigidbody;
36
37     public bool grounded;
38     public LayerMask whatIsGround;
39     public Transform groundCheck;
40     public float groundCheckRadius;
41
42     //private Collider2D myCollider;
43
44     private Animator myAnimator;
45
46     public GameManager theGameManager;
47
48     //for audio
49
50     public AudioSource jumpSound;
51     public AudioSource deathSound;
52
53     private AudioSource backgroundSound;
54
55     public GameObject thePauseMenuScreen;
56
57     public string mainMenuLevel;
58     //gia na mpw stin alli scene
59
60
```

```
61      // Use this for initialization
62      void Start () {
63
64          backgroundSound = GameObject.Find ("backgroundSound").GetComponent<AudioSource> ();
65
66          /* GetComponent: it'll search on the object that this script is attached to, for the
67          component that is inside the <>. */
68
69          myRigidbody = GetComponent<Rigidbody2D> ();
70
71          //myCollider = GetComponent<Collider2D> ();
72
73          myAnimator = GetComponent<Animator> ();
74
75          jumpTimeCounter = jumpTime;
76
77          speedMilestoneCount = speedIncreaseMilestone;
78
79          //prepei na ginetai restart o xronos otan pethainei o player
80          moveSpeedStore = moveSpeed;
81          speedMilestoneCountStore = speedMilestoneCount;
82          speedIncreaseMilestoneStore = speedIncreaseMilestone;
83
84          //gia na min pidaei kai otan fugei apo to telow tis platformas
85          stoppedJumping = true;
86
87          backgroundSound.Play ();
88
89      }
```

```
91      // Update is called once per frame
92      void Update () {
93
94
95
96          //this can return either true or false. It checks if they are touching each other
97          //grounded = Physics2D.IsTouchingLayers (myCollider, whatIsGround);
98
99          /* Physics2D.OverlapCircle(Vector2 point, float radius, int layerMask);
100
101          Checks if a collider falls within a circular area.
102          The circle is defined by its centre coordinate in world space and by its radius.
103          The optional layerMask allows the test to check only for objects on specific layers.*/
104
105         grounded = Physics2D.OverlapCircle(groundCheck.position, groundCheckRadius, whatIsGround);
106
107         if (transform.position.x > speedMilestoneCount)
108         {
109             speedMilestoneCount += speedIncreaseMilestone;
110             speedIncreaseMilestone = speedIncreaseMilestone * speedMultiplier;
111             moveSpeed = moveSpeed * speedMultiplier;
112         }
113
114
115         myRigidbody.velocity = new Vector2 (moveSpeed, myRigidbody.velocity.y);
116
117
118         //otan to patas mia fora
119         if (Input.GetKeyDown (KeyCode.Space) || Input.GetMouseButtonDown(0))
120         {
121
122             // to be able to jump only if touching the ground
123             if (grounded)
124             {
125                 myRigidbody.velocity = new Vector2 (myRigidbody.velocity.x, jumpForce);
126
127                 stoppedJumping = false; //gia na min pidaei kai otan fugei apo to telow tis platformas
128
129                 jumpSound.Play (); //gia na paiksei o ixos toy jump
130             }
131
```

```
132
133             //gia na tsekareis gia doublejump
134             if(!grounded && canDoubleJump)
135             {
136                 myRigidbody.velocity = new Vector2 (myRigidbody.velocity.x, jumpForce);
137
138                 //gia na ftiaksw kai ton xrono sto doublejump
139                 jumpTimeCounter = jumpTime;
140                 stoppedJumping = false;
141                 canDoubleJump = false;
142
143                 jumpSound.Play (); //gia na paiksei o ixos toy jump
144             }
145
146         }
147
148         //otan to kratas patimeno
149         if ((Input.GetKey (KeyCode.Space) || Input.GetMouseButton (0)) && !stoppedJumping)
150             //gia na eimaste sigouroi oti den aioreitai apla
151         {
152             if (jumpTimeCounter > 0)
153             {
154                 myRigidbody.velocity = new Vector2 (myRigidbody.velocity.x, jumpForce);
155                 jumpTimeCounter -= Time.deltaTime; //na meiwnetai kathe fora
156
157                 /*Time.deltaTime: The time in seconds it took to complete the last frame*/
158             }
159         }
160
161
162         //otan to afineis
163         if(Input.GetKeyUp (KeyCode.Space) || Input.GetMouseButtonUp(0))
164         {
165             //arxikopoioume pali ton counter gia na min mporei na
166             //pidaei sunexomena oso einai akoma ston aera
167             jumpTimeCounter = 0;
168
169             stoppedJumping = true;
170             //gia na min pidaei kai otan fugei apo to telow tis platformas
171         }
172
```

```
173
174         if (grounded)
175         {
176             //kai otan patisei to edafos pali na ginei idio me to jumpTime
177             jumpTimeCounter = jumpTime;
178
179             //gia double jump
180             canDoubleJump = true;
181         }
182
183         if (Input.GetKeyDown (KeyCode.P))
184         {
185             backgroundSound.Pause ();
186             Time.timeScale = 0f;
187             thePauseMenuScreen.SetActive (true);
188         }
189
190         if (Input.GetKeyDown (KeyCode.Escape))
191         {
192             Application.LoadLevel (mainMenuLevel);
193         }
194
195
196
197         // Connect the animator settings with the script
198
199         myAnimator.SetFloat ("Speed", myRigidbody.velocity.x);
200
201         myAnimator.SetBool ("Grounded", grounded);
202
203     }
```

```
    //Sent when an incoming collider makes contact with this object's collider
    void OnCollisionEnter2D (Collision2D other)
    {
        //vriskei to sugkekrimeno tag
        if(other.gameObject.tag == "killbox")
        {
            theGameManager.RestartGame();
            //kalei tin sunartisi pou ftiaksame sto gameManager

            //prepei na ginetai restart o xronos otan pethainei o player
            moveSpeed = moveSpeedStore;
            speedMilestoneCount = speedMilestoneCountStore;
            speedIncreaseMilestone = speedIncreaseMilestoneStore;

            deathSound.Play (); // gia na paiksei o ixos otan pethainei

            backgroundSound.Stop ();


        }
    }
}
```

Παράρτημα II: “Camera Controller” Script

CameraController.cs

No selection

```
using UnityEngine;
using System.Collections;

public class CameraController : MonoBehaviour {

    /* we need to know the position of the player
    and we use another script so we access it like that:*/

    public PlayerController thePlayer;


    //to store the position
    private Vector3 lastPlayerPosition;

    //distance we need to move the camera
    private float distanceToMove;

    // Use this for initialization
    void Start () {

        //finding the script we need to access
        thePlayer = FindObjectOfType<PlayerController> ();

        lastPlayerPosition = thePlayer.transform.position;

    }

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

        distanceToMove = thePlayer.transform.position.x - lastPlayerPosition.x;

        transform.position = new Vector3 (transform.position.x + distanceToMove, transform.position.y, transform.position.z);


        //the last position of the player in the end of every frame
        lastPlayerPosition = thePlayer.transform.position;
    }
}
```

Παράρτημα III: "Platform Generator" Script

PlatformGenerator.cs

PlatformGenerator ▸ Update ()

```
1  using UnityEngine;
2  using System.Collections;
3
4  public class PlatformGenerator : MonoBehaviour {
5
6      public GameObject thePlatform;
7
8      //the point ahead of us that we need to
9      //know and generate more platforms
10     public Transform generationPoint;
11
12     public float distanceBetween;
13
14     //we need to know that because we don't
15     //want the platforms to over one another
16     private float platformWidth;
17
18
19     //to randomize the distance between platforms
20     public float distanceBetweenMin;
21     public float distanceBetweenMax;
22
23     //a value to decide which object we gonna be picking
24     private int platformSelector;
25
26     //fix the widths. Avoid the huge gaps between platforms
27     private float[] platformWidths;
28
29     //to access the other script
30     public ObjectPooler[] theObjectPools;
31
32     //making the y position of the platforms
33     private float minHeight;
34     public Transform maxHeightPoint;
35     private float maxHeight;
36     public float maxHeightChange;
37     private float heightChange;
38
39     //gia na kalesw tin sunartisi pou dimiourgei carrots
40     public CarrotGenerator theCarrotGenerator;
41
42     //na ta kanw random
43     public float randomCarrotThreshold;
44
45     //gia na ftiaksw cactus
46     public float randomCactusThreshold;
47
48     public ObjectPooler cactusPool;
49
```

```
52    // Use this for initialization
53    void Start () {
54
55        platformWidths = new float[theObjectPools.Length];
56        //pairnoyme to megethos toy collider gia na kseroyme to makros kathe platformas
57        for (int i = 0; i < theObjectPools.Length; i++)
58        {
59            //the pooledObject is from the scripy "ObjectPooler"
60            platformWidths [i] = theObjectPools [i].pooledObject.GetComponent<BoxCollider2D> ().size.
61        }
62
63        //the current height of the platform
64        minHeight = transform.position.y;
65
66        maxHeight = maxHeightPoint.position.y;
67        //einai idi transform gi'ayto den to vazoyme pali
68
69        theCarrotGenerator = FindObjectOfType<CarrotGenerator> ();
70        //vriskei to object me to carrot generator script
71
72    }
73
74     // Update is called once per frame
75     void Update () {
76
77         if (transform.position.x < generationPoint.position.x)
78         {
79             //to randomize the distance between platforms
80             distanceBetween = Random.Range(distanceBetweenMin,distanceBetweenMax);
81
82             //length=how many platforms we have
83             platformSelector = Random.Range (0, theObjectPools.Length);
84
85
86             //ftiaxnoyme to ypsos na einai tyxaio anamesa stis times poy toy dinw
87             heightChange = transform.position.y + Random.Range (maxHeightChange, -maxHeightChange
88
89             //gia na apofygoume to provlima poy dimourgountai platforms eksw apo ta oria ths kame
90             if (heightChange > maxHeight) {
91                 heightChange = maxHeight;
92             } else if (heightChange < minHeight)
93             {
94                 heightChange = minHeight;
95             }
```

```
96
97
98          //we need to fix the problem of overlaping/touching platforms so we move the generation point to the end of the platform and we divide the width b
99          //Continue at line down
100         transform.position = new Vector3 (transform.position.x + (platformWidths[platformSelector] /2) + distanceBetween, heightChange, transform.position
101
102         //here we use the method from the script ObjectPooler
103
104         //it knows where to look in objectPool and it's able to call from that object pool
105         //(theObjectPools[platformSelector]) that we selected, it's able to GET the objectpool (GetPooledObject();)
106         GameObject newPlatform = theObjectPools[platformSelector].GetPooledObject();
107
108         //we give the position and rotation cause we didn't in instantiate in the other script
109         newPlatform.transform.position = transform.position;
110         newPlatform.transform.rotation = transform.rotation;
111         newPlatform.SetActive (true);
112
113         //the dimiourgoume tuxaia
114         if (Random.Range (0f, 100f) < randomCarrotThreshold)
115         {
116             //pairnoume mesa tin dimiourgia tis neas platformas alla prin metakinithe i
117             //transform.position sto telos tis platformas. Thelw na xrisimopoiisw tin mesh ths
118
119             //xrisimopoio tin sunartisi pou eftiaksa SpawnCarrots apo to CarrotGenerator script
120             theCarrotGenerator.SpawnCarrots (new Vector3 (transform.position.x, transform.position.y + 1.5f, transform.position.z));
121         }
122
123
124         if (Random.Range (0f, 100f) < randomCactusThreshold)
125         {
126
127             GameObject newCactus = cactusPool.GetPooledObject (); //ftiaxnoume kainourgio me to allo script
128
129             float cactusXPosition = Random.Range (-platformWidths [platformSelector] / 2f + 1f, platformWidths [platformSelector] / 2f - 1f);
130             //sto left edge prosthetei ena kai sto right edge afairei ena
131
132             Vector3 cactusPosition = new Vector3 (cactusXPosition, 0.5f, 0f);
133
134             newCactus.transform.position = transform.position + cactusPosition;
135             newCactus.transform.rotation = transform.rotation;
136             newCactus.SetActive (true);
137
138         }
139
140
141         //in continuing: after all is instantiated we add back the other half of the platform without distanceBetween
142         transform.position = new Vector3 (transform.position.x + (platformWidths[platformSelector] /2), transform.position.y, transform.position
143
144
145     }
146 }
147 }
```

Παράρτημα IV: “Object Destroyer” Script

ObjectDestroyer.cs

ObjectDestroyer ▸ No selection

```
1  using UnityEngine;
2  using System.Collections;
3
4  public class ObjectDestroyer : MonoBehaviour {
5
6
7      //the point that we need to set behind the camera,
8      //so we can destroy every object behind it
9      public GameObject objectDestructionPoint;
10
11
12     // Use this for initialization
13     void Start () {
14
15         objectDestructionPoint = GameObject.Find ("ObjectDestructionPoint");
16
17     }
18
19     // Update is called once per frame
20     void Update () {
21
22         if (transform.position.x < objectDestructionPoint.transform.position.x)
23         {
24             //Destroy (gameObject);
25
26
27             //when making an object inactive, it saves more proccess
28             gameObject.SetActive (false);
29         }
30
31     }
32 }
```

Παράρτημα V: “Object Pooler” Script

ObjectPooler.cs

ObjectPooler ▸ No selection

```
1  using UnityEngine;
2  using System.Collections;
3  using System.Collections.Generic;
4  //to use lists we need that last line of using
5
6  public class ObjectPooler : MonoBehaviour {
7
8
9      //the object we need to pool
10     public GameObject pooledObject;
11
12
13     //the number of the clones we need to
14     //create by pooling the object
15     public int pooledAmount;
16
17
18     //creating a list of game objects
19     List<GameObject> pooledObjects;
20
21     // Use this for initialization
22     void Start () {
23
24         pooledObjects = new List<GameObject> ();
25
26         for (int i = 0; i < pooledAmount; i++) {
27
28             //edw kanei kateytheian convert se gameobject
29             //kai den xreiazetai position kai rotation
30             //h diadikasia ayth legetai casting it into something
31             GameObject obj = (GameObject)Instantiate (pooledObject);
32
33             obj.SetActive (false);
34
35             //add the object to the list
36             pooledObjects.Add (obj);
37         }
38     }
```

```
39
40
41          //create a new method so we can get the object
42          //we want to pool and use it whenever we want
43          //from every other script we want
44
45          public GameObject GetPooledObject()
46          {
47              //the size of our array. How many objects it has
48              for(int i=0; i<pooledObjects.Count; i++)
49              {
50                  //search for not active objects
51                  if(!pooledObjects[i].activeInHierarchy) //if is not active
52                  {
53                      return pooledObjects[i];
54                      //it returns them active
55                  }
56              }
57
58              //Ayto to kommati to vazw gia na exw sigoyra ena object active
59              GameObject obj = (GameObject)Instantiate (pooledObject);
60              obj.SetActive (false);
61              pooledObjects.Add (obj);
62              return obj;
63
64          }
65  }
66
```

Παράρτημα VI: “Game Manager” Script

GameManager.cs

GameManager ▸ Reset ()

```
using UnityEngine;
using System.Collections;

public class GameManager : MonoBehaviour {

    //ayto kuriws to xrisimopoioume gia na kanei reset
    //kai tis times toy xronoy klp otan pethainei
    public Transform platformGenerator;
    private Vector3 platformStartPoint;

    public PlayerController thePlayer;
    private Vector3 playerStartPoint;

    private ObjectDestroyer[] platformList;

    //mpainei sto allo script
    private ScoreManager theScoreManager;

    //mpainei se allo script
    public DeathMenu theDeathScreen;



    // Use this for initialization
    void Start () {

        // einai idi transform giayto den to vazw
        platformStartPoint = platformGenerator.position;

        playerStartPoint = thePlayer.transform.position;

        //vriskei to object pou exei ayto to script
        theScoreManager = FindObjectOfType<ScoreManager> ();

    }

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

    }

    //creating new method.
    //It is gonna called whenever the player falls off

    public void RestartGame()
    {
        //twra me to menu den xreiazetai i coroutine
        //ara pairnw ayta poy grafw sto IEnumerator

        //tin stigmi pou tha pethanei o player tha prepei
        //na stamatisei o counter tou score

        theScoreManager.scoreIncreasing = false;
        thePlayer.gameObject.SetActive (false); //svinei ton player

        //tin stigmi pou pethanei tha emfanistei to death screen
        theDeathScreen.gameObject.SetActive (true);

    }

```

```
60
61      //nea function
62      public void Reset()
63      {
64          theDeathScreen.gameObject.SetActive (false);
65          //svinei to death screen otan ginetai pali to reset to paixnidiou
66
67          //me ayto to kommati lunoyme to provlima ton pollwn platforms
68          //poy den katastrefontai meta to restart
69
70          platformList = FindObjectsOfType<ObjectDestroyer> ();
71          for (int i = 0; i < platformList.Length; i++)
72          {
73              platformList [i].gameObject.SetActive (false);
74          }
75
76          thePlayer.transform.position = playerStartPoint;
77          //epanaferei tin thesi stin arxiki
78
79          platformGenerator.position = platformStartPoint;
80          //epanaferei tin thesi stin arxiki
81
82          thePlayer.gameObject.SetActive (true);
83          //ksanaemfanizei ton player
84
85          //otan kanei restart tha prepei na ginei reset sto score
86          //kai na arxisei pali na metraei
87          theScoreManager.scoreCount = 0;
88          theScoreManager.scoreIncreasing = true;
89
90      }
91
92  }
```

Παράρτημα VII: “Score Manager” Script

ScoreManager.cs

ScoreManager ▸ Update ()

```
1  using UnityEngine;
2  using System.Collections;
3  using UnityEngine.UI; //to be able to put text to our application
4
5  public class ScoreManager : MonoBehaviour {
6
7      public Text scoreText;
8      public Text hiScoreText;
9
10     public float scoreCount;
11     public float hiScoreCount;
12
13     public float pointsPerSecond;
14     //h value poy tha ypologizei pote tha exw score.
15     //oso proxwraei o player kai den pethainei diladi
16
17     public bool scoreIncreasing;
18
19     // Use this for initialization
20     void Start () {
21
22         //PlayerPrefs.DeleteKey ("HighScore");
23
24         //gia na kratame to high score akoma kai otan pethanei
25         if (PlayerPrefs.HasKey ("HighScore"))
26         {
27             hiScoreCount = PlayerPrefs.GetFloat ("HighScore");
28         }
29
30     }
```

```
31
32      // Update is called once per frame
33      void Update () {
34
35          if (scoreIncreasing)
36              //to scoreIncreasing to orizoume sto GameManager
37          {
38              scoreCount += pointsPerSecond * Time.deltaTime;
39              //to score pou prosthetoume se kathe frame
40              //Time.deltaTime: The time in seconds
41              //it took to complete the last frame
42          }
43
44          if (scoreCount > hiScoreCount)
45          {
46              hiScoreCount = scoreCount;
47              PlayerPrefs.SetFloat ("HighScore", hiScoreCount);
48              //save a value sto computer pou legetai HighScore
49              //kai exei thn timh toy hiScoreCount
50          }
51
52          /*PlayerPrefs: Stores and accesses player preferences
53           * between game sessions
54          PlayerPrefs.GetFloat: Returns the value corresponding
55          to key in the preference file if it exists.
56          PlayerPrefs.HasKey: Returns true if key exists in
57          the preferences.
58          PlayerPrefs.SetFloat: Sets the value of the preference
59          identified by key.
60          PlayerPrefs.DeleteKey: Removes key and its corresponding
61          value from the preferences*/
62
63
64          //add the text to our screen
65
66          scoreText.text = "Score: " + Mathf.Round (scoreCount);
67          hiScoreText.text = "High Score: " + +Mathf.Round (hiScoreCount);
68
69      }
70
71      //ftiaxnw sunartisi gia na xrisimopoio to AddScore kai se alla scripts
72
73      public void AddScore(int pointsToAdd)
74      {
75          scoreCount += pointsToAdd;
76      }
77  }
```

Παράρτημα VIII: “Pick Up Points” Script

pickUpPoints.cs

pickUpPoints ▸ OnTriggerEnter2D (Collider2D other)

```
1  using UnityEngine;
2  using System.Collections;
3
4  public class pickUpPoints : MonoBehaviour {
5
6      public int scoreToGive;
7
8      private ScoreManager theScoreManager;
9      //gia na mpoume sto allo script
10
11     public AudioSource eatSound;
12     //gia ton ixo pou trwei
13
14
15     // Use this for initialization
16     void Start () {
17
18         theScoreManager = FindObjectOfType <ScoreManager> ();
19         //vriskei ta objects pou exoyn to ScoreManager
20
21         eatSound = GameObject.Find ("EatSound").GetComponent<AudioSource> ();
22         // sundew to arxeio tou ixou mou me tin metavliti
23     }
24
25     // Update is called once per frame
26     void Update () {
27
28     }
29
30     //build-in sunartisi
31     //Checks to see if something enters our trigger.
32     //That smthing is another object that has a collider
33
34     void OnTriggerEnter2D (Collider2D other)
35     {
36         if (other.gameObject.name == "Player")
37             //if that "other" is our player
38         {
39             theScoreManager.AddScore (scoreToGive);
40             //xrisimopoiw tin sunartisi apo to script
41             //ScoreManager me value scoreToGive
42
43             gameObject.SetActive (false);
44             //inactivating to object pou exei
45             //ayto to script. Ta karota diladi
46
47             //gia na min akougetai san enas ixos
48             //otan mazeyei ta krota poly grigora
49
50             //an o ixos paizei idi tote stop kai meta paikse pali
51             if (eatSound.isPlaying) {
52                 eatSound.Stop ();
53                 eatSound.Play ();
54
55             } else
56             {
57                 eatSound.Play ();
58             }
59
60         }
61     }
62 }
63
```

Παράρτημα IX: “Carrot Generator” Script

CarrotGenerator.cs

No selection

```
1  using UnityEngine;
2  using System.Collections;
3
4  public class CarrotGenerator : MonoBehaviour {
5
6      public ObjectPooler carrotPool;
7
8      public float distanceBetweenCarrots;
9
10     //new method
11
12     public void SpawnCarrots(Vector3 startPosition)
13     {
14         GameObject carrot1 = carrotPool.GetPooledObject (); //h sunartisi pou ftiaxnei game objects ston kosmo
15         carrot1.transform.position = startPosition;
16         carrot1.SetActive (true);
17
18         GameObject carrot2 = carrotPool.GetPooledObject (); //h sunartisi pou ftiaxnei game objects ston kosmo
19         carrot2.transform.position = new Vector3(startPosition.x - distanceBetweenCarrots, startPosition.y, startPosition.z)
20         carrot2.SetActive (true);
21
22         GameObject carrot3 = carrotPool.GetPooledObject (); //h sunartisi pou ftiaxnei game objects ston kosmo
23         carrot3.transform.position = new Vector3(startPosition.x + distanceBetweenCarrots, startPosition.y, startPosition.z)
24         carrot3.SetActive (true);
25     }
26
27 }
```

Παράρτημα X: “Main Menu” Script

MainMenu.cs

MainMenu ▸ No selection

```
1  using UnityEngine;
2  using System.Collections;
3
4  public class MainMenu : MonoBehaviour {
5
6      public string playGameLevel;
7      //gia na mporw na xrisimopoiisw onoma
8      //apo alli scene
9
10     public GameObject theInfoScreen;
11     public GameObject theCreditsScreen;
12
13
14     void Update()
15     {
16         if (Input.GetKey (KeyCode.Escape))
17         {
18             theInfoScreen.SetActive (false);
19             theCreditsScreen.SetActive (false);
20         }
21     }
22
23     public void PlayGame()
24     {
25         Application.LoadLevel (playGameLevel);
26     }
27
28     public void QuitGame()
29     {
30         Application.Quit ();
31     }
32
33     public void Credits()
34     {
35         theCreditsScreen.SetActive (true);
36     }
37
38     public void infoGame()
39     {
40         theInfoScreen.SetActive (true);
41     }
42
43     public void arrowBack()
44     {
45         theInfoScreen.SetActive (false);
46         theCreditsScreen.SetActive (false);
47     }
48 }
```

Παράρτημα XI: “Death Menu” Script

DeathMenu.cs

DeathMenu ▸ RestartGame ()

```
1  using UnityEngine;
2  using System.Collections;
3
4  public class DeathMenu : MonoBehaviour {
5
6      public string mainMenuLevel; //gia na kanw access stin mainmenu scene
7
8      private AudioSource backgroundSound;
9
10     public GameObject areSureMenu;
11     public GameObject theDeathMenu;
12
13
14     public void RestartGame ()
15     {
16         backgroundSound = GameObject.Find ("backgroundSound").GetComponent<AudioSource> (
17
18         //i sunartisi pou tha xrisimopoiiso apo to gamemanager
19         FindObjectOfType<GameManager>().Reset();
20
21         backgroundSound.Stop();
22         backgroundSound.Play();
23     }
24
25     public void areSure()
26     {
27         areSureMenu.SetActive (true);
28         theDeathMenu.SetActive (false);
29     }
30 }
```

Παράρτημα XII: "Pause Menu" Script

PauseMenu.cs

PauseMenu ▸ No selection

```
1 using UnityEngine;
2 using System.Collections;
3
4 public class PauseMenu : MonoBehaviour {
5
6     public string mainMenuLevel; // gia na mpw se alli scene
7
8     public GameObject pauseMenu;
9
10    private AudioSource backgroundSound;
11
12    public GameObject areSureMenu;
13    public GameObject theDeathMenu;
14
15    void Start()
16    {
17        backgroundSound = GameObject.Find ("backgroundSound").GetComponent<AudioSource> ();
18    }
19
20
21    /*Time.timeScale: the scale at which the time is passing.
22     This can be used for slow motion effects*/
23
24    public void PauseGame()
25    {
26        Time.timeScale = 0f;
27        pauseMenu.SetActive (true);
28        backgroundSound.Pause ();
29    }
30
31    public void ResumeGame()
32    {
33        Time.timeScale = 1f;
34        pauseMenu.SetActive (false);
35        backgroundSound.UnPause ();
36    }
37
38
39    public void RestartGame ()
40    {
41        Time.timeScale = 1f;
42        pauseMenu.SetActive (false);
43
44        FindObjectOfType<GameManager>().Reset();
45        //i sunartisi pou tha xrisimopoiiso apo to gamemanager
46
47        backgroundSound.Stop();
48        backgroundSound.Play();
49
50    }
51
52    public void areSure()
53    {
54
55        pauseMenu.SetActive (false);
56        areSureMenu.SetActive (true);
57    }
58 }
```

Παράρτημα XIII: "Are Sure" Script

AreSure.cs

AreSure ▸ No selection

```
1  using UnityEngine;
2  using System.Collections;
3
4  public class AreSure : MonoBehaviour {
5
6      //gia na kanw access stin mainmenu scene
7      public string mainMenuLevel;
8
9      private AudioSource backgroundSound;
10
11     public GameObject areSureMenu;
12     public GameObject theDeathMenu;
13     public GameObject thePauseMenu;
14
15
16     public void noSureDeath() //no button
17     {
18         areSureMenu.SetActive (false);
19         theDeathMenu.SetActive (true);
20     }
21
22     public void yesSure() //yes button
23     {
24         Time.timeScale = 1f;
25         Application.LoadLevel (mainMenuLevel);
26     }
27
28     public void noSurePause() //no button
29     {
30         areSureMenu.SetActive (false);
31         thePauseMenu.SetActive (true);
32     }
33
34 }
```

Παράρτημα XIV: **Ερωτηματολόγιο** "For The Gamers"

For the Gamers

A Questionnaire for the Gamers. I will use it for my dissertation, so thank you in advance for your participation.

*Required

What gender are you? *

- ○ Male
- ○ Female

How old are you? *

- ○ Less than 13
- ○ 13 to 18
- ○ 19 to 23
- ○ 24 to 28
- ○ older than 28

How many years do you play video games? *

- ○ less than 1 year
- ○ 1 to 4 years
- ○ 5 to 7 years
- ○ 7 to 10 years
- ○ more than 10 years

Which genre of video games do you play? *

You can choose more than one choice.

- [ ] Action
- [ ] First-Person Shooter
- [ ] Action-Adventure
- [ ] Adventure
- [ ] Role-Playing Games (RPG)
- [ ] Strategy
- [ ] Sports
- [ ] Indie
- [ ] Simulation
- [ ] Casual
- [ ] Massively Multiplayer
- [ ] MOBA
- [ ] Racing
- [ ] Open World
- [ ] Atmospheric
- [ ] Puzzle
- [ ] Other: _____

Which type of Video games do you prefer? *

- ○ Single player
- ○ Online Multiplayer
- ○ LAN Games (Local Area Network Games)

In which platform do you prefer to play video games? *

You can choose more than one choice

- ☐ Personal Computer (PC)
- ☐ Sony Platform (Playstation, ps2, ps3, ps4)
- ☐ Microsoft Platform (Xbox, Xbox One, etc)
- ☐ Nintendo Platform (Gameboy, etc)

What do you like most about video games?

You can choose more than one choice.

- ☐ The graphics
- ☐ The story
- ☐ The fireworks
- ☐ The oportunity to solve riddles
- ☐ The construction
- ☐ The oportunity to build your own communities
- ☐ The oportunity to meet new people
- ☐ Other: _____

Would you like the idea of making your own video games? *

- ○ Yes
- ○ No
- ○ Haven't thought about it

If yes, which game engine would you prefer?

You can choose more than one choice.

- ☐ Unity 3D
- ☐ LeadWerks Game Engine
- ☐ Unreal Engine
- ☐ CryEngine 3
- ☐ HeroEngine
- ☐ Rage Engine
- ☐ Project Anarchy
- ☐ GameSalad
- ☐ GameMaker: Studio
- ☐ App Game Kit
- ☐ Cocos2D
- ☐ Other: _____

Which games do you play? *

Please write the titles separated by comma. Write as many as you like. (Example: Rome Total War, Crash Bandicoot 3, Uncharted 4)

Your answer

What type of video games do you prefer to own? *

- ○ Digital Files (Example: Steam)
- ○ Hardcopies (Example: Bluray, DVD)

Index

Links **των παιχνιδιών που αναφέρονται στην πτυχιακή**

http://playtictactoe.org/

https://www.sparkchess.com/

https://mario.nintendo.com/

http://us.blizzard.com/en-us/games/sc2/

https://www.ubisoft.com/en-US/game/prince-of-persia/

http://www.unchartedthegame.com/en-us/

http://www.dark-chronicle.co.uk/

https://www.ageofempires.com/

http://www.rockstargames.com/vicecity/

http://assassinscreed.ubi.com/en-gb/home/

https://www.callofduty.com/

https://www.tombraider.com/en-us

http://streetfighter.com/

https://www.mortalkombat.com/

http://dragonball.bandainamcogames.com/

http://www.dreamfall.com/

https://www.totalwar.com/

https://elderscrolls.bethesda.net/

http://thewitcher.com/en/witcher3

https://www.easports.com/fifa

https://www.konami.com/wepes/2017/eu/en/

https://www.2k.com/games/nba-2k17

https://www.microsoft.com/games/flight/

https://www.needforspeed.com/

http://www.gran-turismo.com/gr/

https://worldofwarcraft.com/en-gb/

https://www.smitegame.com/

http://blog.dota2.com/

http://play.na.leagueoflegends.com/en_US

http://stardewvalley.net/

http://www.hotlinemiami.com/

http://www.shiningrocksoftware.com/

http://www.darfurisdying.com/

http://www.re-mission2.org/#/about

http://www.puffgames.com/snake/

https://www.angrybirds.com/games/

Βιβλιογραφία

Adams, E., 2010. *Fundamentals of Game Design.* 2nd επιμ. USA: New Riders.

Feil, J. & Scattergood, M., 2005. *Beggining Game Level Design.* USA: Thomson Course Technology.

Moore, M. E., 2011. *Basics of Game Design.* USA: CRP Press.

Rose, M., 2011. *250 Indie Games You Must Play.* USA: CRC Press.

Rouse III, R., 2001. *Game Design: Theory & Practice.* USA: Wordware Publishing.

Sylvester, T., 2013. *Designing Games: A Guide to Engineering Experiences.* USA: O'Reilly.

Wolf, M. J. P., 2012. *Encyclopedia of Video Games.* USA: Greenwood.