ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

Σχολή Τεχνολογιών Πληροφορικής και Επικοινωνιών

Τμήμα Πληροφορικής

Πτυχιακή Εργασία

Τίτλος Πτυχιακής Εργασίας	*Μάχη στο διάστημα: ένα 2D παιχνίδι για πολλούς παίκτες για Android και PC* *Space Battle: a 2D multiplayer game for Android and PC*
Ονοματεπώνυμο Φοιτητή	Καλοϊάν Καλαϊντζίεβ
Πατρώνυμο	Μιλέν
Αριθμός Μητρώου	Π21044
Επιβλέπων	Αλέπης Ευθύμιος, Καθηγητής

Σεπτέμβριος 2025

Copyright

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα.

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν αποκλειστικά τον συγγραφέα και δεν αντιπροσωπεύουν τις επίσημες θέσεις του Πανεπιστημίου Πειραιώς.

Ως συγγραφέας της παρούσας εργασίας δηλώνω πως η παρούσα εργασία δεν αποτελεί προϊόν λογοκλοπής και δεν περιέχει υλικό από μη αναφερόμενες πηγές.

Ευχαριστίες

Θα ήθελα να εκφράσω τις ειλικρινείς μου ευχαριστίες σε όλους όσοι συνέβαλαν στην ολοκλήρωση της πτυχιακής μου εργασίας. Πρώτα απ' όλα, ευχαριστώ θερμά τον επιβλέποντά μου, κ. Ευθύμη Αλέπη, για την πολύτιμη καθοδήγηση καθ' όλη τη διάρκεια της συγγραφής, καθώς και για τις γνώσεις και τη στήριξη που μου παρείχε. Τέλος, ευχαριστώ την οικογένειά μου και τους στενούς μου φίλους για την αδιάκοπη υποστήριξη που μου προσέφεραν, όχι μόνο κατά την εκπόνηση της παρούσας εργασίας, αλλά σε όλη τη ακαδημαϊκή μου πορεία.

Abstract (EN)

This thesis presents the design and implementation of Space Battle, a 2D, team-based (Blue vs Red) multiplayer shooter built with Unity, targeting Android with optional Windows PC cross-play. The work focuses on a mobile-first experience: on-screen controls for portrait and landscape modes and an in-game layout editor. Core gameplay features include three weapons—bullets (hitscan), a re-targetable rocket, and a continuous electric beam—and time-based power-ups that spawn periodically.

Networking relies on Unity Lobby/Relay and Netcode for GameObjects (NGO) in a host-authoritative model. Clients send input commands; the host simulates authoritative state and disseminates snapshots. At match end, the host broadcasts per-player statistics (kills, deaths, etc.), which each client persists locally in JSON, a deliberate choice because the stored data provide no competitive advantage.

The methodology covers requirements, system architecture, gameplay and networking design, and a reference implementation, followed by functional and performance testing. We discuss trade-offs specific to mobile multiplayer—authority, synchronization, bandwidth budget, and UX across orientations—and outline future extensions.

Keywords: Unity, Multiplayer, Android, Networking, Synchronization, Latency, Interpolation

Περίληψη (GR)

Η εργασία παρουσιάζει τον σχεδιασμό και την υλοποίηση του Space Battle, ενός 2D, ομαδικού (Blue vs Red) multiplayer shooter σε Unity, με κύρια πλατφόρμα το Android και προαιρετική υποστήριξη για Windows PC. Δίνεται έμφαση στο mobile-first χειρισμό: on-screen controls για οριζόντιο και κάθετο προσανατολισμό και ενσωματωμένο layout editor. Το gameplay περιλαμβάνει τρία όπλα—σφαίρες, ανακατευθυνόμενο πύραυλο και ηλεκτρικό τόξο—καθώς και power-ups που εμφανίζονται περιοδικά.

Η δικτύωση βασίζεται σε Unity Lobby/Relay και Netcode for GameObjects (NGO) με host-authoritative αρχιτεκτονική. Οι clients αποστέλλουν εντολές, ο host προσομοιώνει την κατάσταση και διανέμει στιγμιότυπα. Στο τέλος του αγώνα, ο host αποστέλλει ανά παίκτη στατιστικά (kills, deaths κ.ά.), τα οποία κάθε client αποθηκεύει τοπικά σε JSON, επιλογή που απλοποιεί την υλοποίηση, αφού τα δεδομένα δεν προσφέρουν ανταγωνιστικό πλεονέκτημα.

Η μεθοδολογία καλύπτει απαιτήσεις, αρχιτεκτονική, σχεδίαση gameplay/δικτύωσης και υλοποίηση, καθώς και λειτουργικές και δοκιμές. Συζητούνται κρίσιμες ισορροπίες για mobile multiplayer—αυθεντία, συγχρονισμός, προϋπολογισμός εύρους ζώνης, UX ανά προσανατολισμό—και προτείνονται επεκτάσεις.

Λέξεις-κλειδιά: Unity, Πολυχρηστικά παιχνίδια, Android, Δικτύωση, Συγχρονισμός, Καθυστέρηση, Παρεμβολή

Πίνακας Περιεχομένων

Κατάλογος Εικόνων

Κατάλογος Πινάκων

Κατάλογος Ροών

Εισαγωγή

Σκοπός και Συνεισφορά

Σκοπός της εργασίας είναι ο σχεδιασμός και η υλοποίηση του Space Battle, ενός 2D, ομαδικού (Κόκκινοι vs Μπλε) multiplayer shooter με προσανατολισμό σε Android συσκευές και προαιρετική συμμετοχή από PC. Κάθε παίκτης χειρίζεται διαστημόπλοιο με τρία διακριτά όπλα: (α) κανονικές βολές, (β) πύραυλο που αλλάζει στόχευση με tap οπουδήποτε στην αρένα, και (γ) ηλεκτρική ακτίνα με συνεχή εφαρμογή ζημιάς. Στο πεδίο μάχης εμφανίζονται power-ups περίπου κάθε δέκα δευτερόλεπτα (π.χ. slow motion health regeneration, invisibility, invincibility), τα οποία τροποποιούν προσωρινά τη ροή του αγώνα και ενισχύουν την ποικιλία στρατηγικών.

Η δικτύωση βασίζεται σε Unity Multiplayer Services (Lobby/Relay) και Netcode for GameObjects (NGO). Υιοθετείται αρχιτεκτονική host-authoritative: οι clients αποστέλλουν εισόδους, ο host προσομοιώνει την κατάσταση και διανέμει snapshots στους clients. Στο τέλος κάθε αγώνα, ο host αποστέλλει στατιστικά (kills, deaths κ.ά.), τα οποία οι clients αποθηκεύουν τοπικά σε JSON· η επιλογή τοπικής αποθήκευσης έγινε εσκεμμένα, διότι τα δεδομένα δεν παρέχουν ανταγωνιστικό πλεονέκτημα και μειώνουν σημαντικά την πολυπλοκότητα του backend.

Το Space Battle δίνει έμφαση σε mobile-first χειρισμό: on-screen πλήκτρα, προσαρμοσμένα για portrait και landscape, καθώς και ενσωματωμένο editor διάταξης για κάθε προσανατολισμό. Η ροή του παιχνιδιού είναι: Main Menu → Host/Join Lobby → Game → Lobby ή Main Menu. Η είσοδος σε lobby γίνεται με κωδικό ή μέσω QR code που μοιράζεται ο host.

Κύριες συνεισφορές:

- Πλήρης δικτυακός σκελετός με host authority, ελαφριά μηνύματα και συγχρονισμό όπλων/power-ups.

- Mobile UX δύο προσανατολισμών και layout editor στο ίδιο το παιχνίδι.

- Απλό matchmaking (code/QR) χωρίς λογαριασμούς ή επίμονο backend.

- Τοπική επιμονή (JSON) για στατιστικά/skins και σύστημα ξεκλειδώματος βάσει κριτηρίων.

Πλαίσιο και Κίνητρα

Τα mobile multiplayer παιχνίδια λειτουργούν κάτω από περιορισμούς υπολογιστικής ισχύος, θερμικών ορίων και δικτυακής μεταβλητότητας. Παράλληλα, η εμπειρία χρήστη πρέπει να παραμένει ομαλή σε διάφορες αναλύσεις/αναλογίες οθόνης και διαφορετικούς προσανατολισμούς. Η αξιοποίηση των Unity Services (Lobby/Relay) και του NGO μειώνει το κόστος ολοκλήρωσης κρίσιμων υποδομών (NAT traversal, session management, replication), επιτρέποντας εστίαση στον σχεδιασμό gameplay και στη βελτιστοποίηση για συσκευές μεσαίας κατηγορίας.

Κίνητρο της εργασίας είναι να παραχθεί ένα συμπαγές παράδειγμα αρχιτεκτονικής και υλοποίησης mobile multiplayer, το οποίο ισορροπεί απλότητα και ρεαλισμό: προσφέρει ουσιαστικό gameplay και πραγματική δικτύωση χωρίς την πολυπλοκότητα πλήρους backend υπηρεσίας.

Υποθέσεις και Περιορισμοί

- Αρχιτεκτονική: Host-authoritative μοντέλο - οι clients θεωρούνται μη αξιόπιστοι.

- Παίκτες ανά αγώνα: 1v1 ή 2v2 (λειτουργικό εύρος 2–8).

- Δίκτυο: Wi-Fi/4G.

- Tickrate: ενδεικτικά 30 Hz με client-side interpolation· δυνατότητα αναπροσαρμογής σε δοκιμές.

- Στόχος απόδοσης: Μέγιστο ποσοστό απόδοσης

- Δεδομένα: Τοπικά JSON αρχεία για τα δεδομένα του παίκτη και την διάταξη των κουμπιών.

- Ασφάλεια: Βασική επικύρωση ενεργειών στον host

- Πλατφόρμες: Android (κύρια), Windows PC (δευτερεύουσα).

Μεθοδολογία και Δομή Εργασίας

Η εργασία οργανώνεται ως εξής:

- Στο Κεφάλαιο 1 αναπτύσσεται το θεωρητικό υπόβαθρο (μοντέλα δικτύωσης, τεχνικές συγχρονισμού, οικοσύστημα Unity, περιορισμοί mobile)
- Στο Κεφάλαιο 2 περιγράφεται η αρχιτεκτονική συστήματος (επιλογές engine/NGO, υπηρεσίες Lobby/Relay, μοντέλο αυθεντίας)
- Στο Κεφάλαιο 3 αναλύονται οι ροές και σχεδίαση παιχνιδιού (game flow, μηχανισμοί όπλων και power-ups)
- Στο Κεφάλαιο 4 περιγράφεται η υλοποίση του έργου
- Στο Κεφάλαιο 5 γίνεται η αξιολόγηση και προτείνονται κατευθύνσεις για επεκτάσεις για την ολοκλήρωση του παιχνιδιού

1. Θεωρητικό Υπόβαθρο

1.1 Βασικές Έννοιες Δικτύωσης σε Παιχνίδια

Client–Server vs P2P: Στα περισσότερα σύγχρονα action/multiplayer παιχνίδια επιλέγεται client–server (ή listen-server/host-authoritative), όπου ένας κόμβος (server ή host-παίκτης) κρατά την αυθεντία της κατάστασης του παιχνιδιού. Οι clients στέλνουν εντολές εισόδου (inputs), ενώ ο host προσομοιώνει τον κόσμο και αποστέλλει στιγμιότυπα (snapshots) της κατάστασης. Το P2P αποφεύγεται όταν απαιτείται ασφάλεια/δίκαιη παιξιμότητα, διότι μοιράζει την αυθεντία σε πολλούς μη αξιόπιστους κόμβους.

Χρονισμός & Tickrate: Η προσομοίωση εξελίσσεται σε διακριτά ticks (π.χ. 30 Hz). Ο client συνήθως παρεμβάλλει (interpolates) τις ληφθείσες καταστάσεις για ομαλή κίνηση, ενώ ο host «κόβει» την κατάσταση σε snapshots. Μεταβλητότητα δικτύου (RTT, jitter, packet loss) αντιμετωπίζεται με buffers παρεμβολής, επαναμεταδόσεις όπου χρειάζεται (reliable channels) και διορθώσεις (reconciliation).

Διαύλοι μετάδοσης: Για συχνές ενημερώσεις θέσεων προτιμάται μη αξιόπιστη μετάδοση (unreliable/UDP) ώστε να αποφεύγεται συμφόρηση από παλιές ενημερώσεις. Για «κρίσιμα» συμβάντα (spawn, score) χρησιμοποιούνται αξιόπιστα κανάλια με επιβεβαίωση.

1.2 Τεχνικές Συγχρονισμού

Παρεμβολή (Interpolation): Ο client διατηρεί buffer πρόσφατων snapshots και προβάλλει την εικόνα λίγο «πίσω στον χρόνο» (π.χ. ~100 ms), ώστε να ανακατασκευάσει ομαλή κίνηση ανάμεσα σε δύο γνωστά σημεία. Έτσι μικρές απώλειες/καθυστερήσεις δεν γίνονται αντιληπτές ως «άλματα».

Πρόβλεψη πελάτη (Client-Side Prediction): Για άμεση απόκριση, ο client εφαρμόζει το δικό του input στο δικό του αντίγραφο της προσομοίωσης χωρίς να περιμένει τον host. Όταν λάβει την επίσημη κατάσταση, ελέγχει αν διαφέρει (misprediction) και αν

χρειάζεται, επαναπαίζει τα inputs πάνω στη διορθωμένη κατάσταση (reconciliation). Η αυθεντία παραμένει στον host, αποτρέποντας μόνιμο desync/cheat.

Lag Compensation: Σε χτυπήματα «στιγμιαίας ανίχνευσης» (hitscan), ο host μπορεί να «γυρίσει πίσω στον χρόνο» για τον στόχο, χρησιμοποιώντας ιστορικό θέσεων, ώστε να αξιολογήσει το χτύπημα όπως το «είδε» ο σκοπευτής με το δικό του latency. Αυτό μειώνει αδικίες λόγω καθυστέρησης.

Συμπίεση/Διαφορική μετάδοση: Επειδή τα συνεχόμενα snapshots μοιάζουν, μεταδίδονται διαφορές (deltas) και μόνο για ενδιαφέροντα αντικείμενα. μειώνοντας το bandwidth/πακέτο.

1.3 Το Οικοσύστημα Unity για Multiplayer

Unity Transport (UTP): είναι το χαμηλού επιπέδου “στρώμα μεταφοράς” δικτύου της Unity. Φροντίζει για το γρήγορο και αποδοτικό πέρα-δώθε πακέτων (κυρίως πάνω από UDP, και WebSocket για WebGL) και δίνει primitives όπως αξιόπιστη/μη αξιόπιστη μετάδοση, σειρά/ταξινόμηση πακέτων, κατακερματισμός, και βασική προσομοίωση καθυστέρησης/απωλειών. Είναι το θεμέλιο πάνω στο οποίο «κάθονται» τα Netcode for GameObjects (NGO) και Netcode for Entities (ECS).

Netcode for GameObjects (NGO): Υψηλού επιπέδου βιβλιοθήκη για παιχνίδια που βασίζονται σε GameObjects. Κύριες έννοιες: NetworkObject (ταυτότητα/ιδιοκτησία), NetworkBehaviour (συγχρονισμένη συμπεριφορά), RPCs (ServerRpc, ClientRpc) για γεγονότα και NetworkVariables για κατάσταση μικρού μεγέθους με αυτόματο replication. Υποστηρίζει host-authoritative μοντέλο, που ταιριάζει με το Space Battle.

Unity Lobby & Relay: Το Lobby προσφέρει «ανακάλυψη» και είσοδο με κωδικό ή Quick Join σε μια συνεδρία, ενώ το Relay λειτουργεί ως διαμεσολαβητής δικτύου (proxy) ώστε οι clients να επικοινωνούν χωρίς άμεσο άνοιγμα θυρών/NAT ρυθμίσεις. Τυπική ροή: ο host δημιουργεί allocation στο Relay και παράγει join code· οι clients συνδέονται με τον κωδικό.

1.4 Mobile-First Περιορισμοί

Απόδοση & Frame Pacing: Τα κινητά και οι συσκευές μεσαίας κατηγορίας υπόκεινται σε θερμικό throttling και μεταβλητές συχνότητες CPU/GPU. Σχεδιαστικές πρακτικές: σταθερό fixed timestep, περιορισμός «ζωγραφικής» στην οθόνη, και ευέλικτος στόχος FPS (60 με πτώση στα 30 όταν χρειάζεται).

Δίκτυο σε Κινητό Περιβάλλον: Οι τιμές RTT/jitter είναι υψηλότερα/ασταθέστερα έναντι ενσύρματου. Επιλέγουμε μικρά, συχνά πακέτα για state και αξιόπιστα μόνο όπου απαιτείται. Για ορατή ομαλότητα, ο buffer παρεμβολής πρέπει να απορροφά τις αιχμές jitter· υπερβολικά μεγάλος buffer αυξάνει την καθυστέρηση ελέγχου.

Χειρισμός & Προσανατολισμός: Touch controls με δύο διατάξεις (portrait/landscape) και editor στο παιχνίδι μειώνουν τριβές και επιτρέπουν εξατομίκευση. UI scaling πρέπει να λειτουργεί σε μεγάλη ποικιλία DPI/αναλύσεων.

1.5 Matchmaking, Lobbies και Ροές Συνεδρίας

Η υποδομή Lobby/Relay καλύπτει: (α) δημιουργία λόμπι από host, (β) διαμοιρασμό join code, (γ) έλεγχο χωρητικότητας/ιδιωτικότητας, (δ) Quick Join για αυτόματη εύρεση διαθέσιμου λόμπι. Στο Space Battle αξιοποιούμε κωδικό, ο οποίος μπορεί να μετατραπεί και σε QR για απλότητα. Με την έναρξη, ο host γίνεται authoritative κόμβος και ξεκινά την αναπαραγωγή μέσω NGO.

1.6 Δεδομένα, Επιμονή και Ιδιωτικότητα

Για μη κρίσιμα δεδομένα (kills, deaths, matchesPlayed, skins, ρυθμίσεις UI), επιλέγεται τοπική αποθήκευση σε JSON στον φάκελο της εφαρμογής. Πλεονεκτήματα: απλότητα, offline λειτουργία, μηδενικό backend. Μειονέκτημα: ευκολία τροποποίησης από τον χρήστη—γι’ αυτό τέτοια δεδομένα δεν επηρεάζουν το matchmaking ή το δίκαιο αποτέλεσμα του αγώνα.

1.7 Ασφάλεια & Δικαιοσύνη Παιξίματος

Βασικές πρακτικές για host-authoritative παιχνίδια:

- Validation server-side για damage, συγκρούσεις, παραλαβή αντικειμένου, respawn.
- Rate limiting σε RPC/inputs για αποτροπή spam.
- Έλεγχοι ορθότητας κατάστασης και περιοδική επανευθυγράμμιση (π.χ. απόρριψη ακραίων αποκλίσεων θέσης).
- (Προαιρετικά) Αντιστάθμιση καθυστέρησης για hitscan, με ιστορικό θέσεων ~1 s.

2. Αρχιτεκτονική Συστήματος

2.1 Επιλογή Game Engine - Unity

Για την υλοποίηση του Space Battle επιλέχθηκε η Unity ως μηχανή παιχνιδιού, διότι προσφέρει γρήγορες και πρακτικές λύσεις στο κομμάτι του multiplayer networking (σύνδεση χρηστών, lobby/relay, ανώτερου επιπέδου API συγχρονισμού) και επιτρέπει παραγωγή εκτελέσιμων για Android/Windows χωρίς αλλαγές στον κώδικα gameplay.

Η Unity είναι μία από τις πιο διαδεδομένες και ευέλικτες μηχανές ανάπτυξης παιχνιδιών σε 2D/3D/VR/AR, Χρησιμοποιεί κυρίως τη γλώσσα C#, προσφέροντας ένα εύχρηστο περιβάλλον εργασίας, μεγάλη κοινότητα υποστήριξης και εργαλεία για ανάπτυξη σε πολλές πλατφόρμες (PC, κινητά, κονσόλες, web).

Οι κύριοι λόγοι που επιλέχθηκε η Unity για την ανάπτυξη της παρούσας πτυχιακής εργασίας είναι:

- **Unity Multiplayer Services**: έτοιμες υπηρεσίες για δημιουργία λόμπι, διαμοιρασμό κωδικού συμμετοχής/QR, επικοινωνία host–clients και διευκόλυνση σύνδεσης πίσω από NAT χωρίς ειδική υποδομή.

- **Netcode for GameObjects**: υψηλού επιπέδου δικτυακά primitives (NetworkObject, NetworkVariable, ServerRpc/ClientRpc) που επιτρέπουν ταχεία υλοποίηση host-authoritative λογικής με λιγότερο boilerplate.

- **Cross-platform by design**: Η Unity υποστηρίζει διάφορα λειτουργικά συστήματα. Χρησιμοποιώντας επιπλέον βιβλιοθήκες, το τελικό πρότζεκτ μπορεί να βγει για οποιοδήποτε λειτουργικό σύστημα.

- **Asset Store**: Το Asset Store είναι ένα online κατάστημα μέσα στο Unity όπου μπορούμε να αγοράσουμε ή να κατεβάσουμε δωρεάν έτοιμα “assets” — δηλαδή μοντέλα, υφές, ήχους, scripts, animations, εργαλεία, plugins, κ.λπ..

2.2 Networking Layer - NGO

Η δικτύωση του Space Battle οργανώνεται σε επιμέρους υποσυστήματα. Τα πιο σημαντικά υποσυστήματα είναι:

- **Lobby/Relay Layer**:
    - Είναι υπεύθυνο για την εγκατάσταση σύνδεσης μεταξύ των χρηστών.
    - Ένας χρήστης μπορεί να δημιουργήσει ή να μπει σε ένα «δωμάτιο» με άλλους παίκτες, χρησιμοποιώντας τον κωδικό που του δίνεται κατά την

δημιουργία του δωματίου.

- Netcode for GameObjects (NGO):
  - Ένα υποσύστημα υπεύθυνο για τον συγχρονισμό μεταξύ δικτυοτών αντικειμένων, μεταβλητών, καταστάσεων κ.λπ. μέσα σε μία συνεδρία.
  - Όλοι οι υπολογισμοί των δικτυοτών αντικειμένων γίνονται στην πλευρά του Host.
  - Οι Clients στέλνουν αιτήματα προς τον Server όταν θέλουν να εκτελέσουν κάποια ενέργεια στο παιχνίδι (π.χ. να πυροβολήσουν).

Η NGO φέρνει τις παρακάτω έννοιες:

- **NetworkVariables**: Μεταβλητές οι οποίες είναι ορατές στο δίκτυο. Τέτοιες μεταβλητές μπορεί να είναι matchDuration, isMovementLocked, flags ικανοτήτων (π.χ. canShoot για τους παίκτες), health κτλ. Είναι οι μεταβλητές, οι οποίες αλλάζονται συνήθως από τον Host και διαβάζονται από όλους.

- **ServerRpc**: Είναι μια συνάρτηση που καλείται από έναν Client και εκτελείται στον Server. Μια τέτοια συνάρτηση είναι η **FireRocketServerRpc** η οποία λέει ο Client στον Host ότι θέλει να πετάξει έναν πύραυλο

- **ClientRpcs**: Είναι μια συνάρτηση που καλείται από τον Server και τρέχει σε όλους τους Clients. Μια τέτοια συνάρτηση είναι η **SendNotificationClientRpc** η οποία στέλνει (εμφανίζει) ειδοποίηση σε όλους τους Clients.

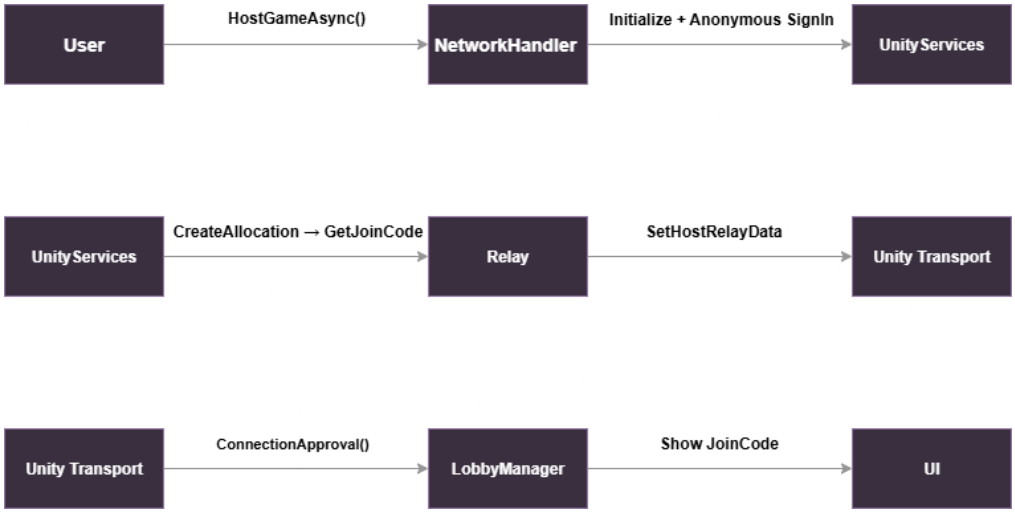
2.3 Σύνοψη & Σύνδεση

Οι επιλογές του έργου (NGO + Lobby/Relay, host-authoritative, interpolation client-side, τοπικό JSON) ευθυγραμμίζονται με βέλτιστες πρακτικές για μικρής κλίμακας mobile τίτλο. Στα επόμενα κεφάλαια μετατρέπουμε τα παραπάνω σε συγκεκριμένη αρχιτεκτονική, σχεδίαση ροών και μετρικές αξιολόγησης.

3. Ροές και Σχεδίαση παιχνιδιού

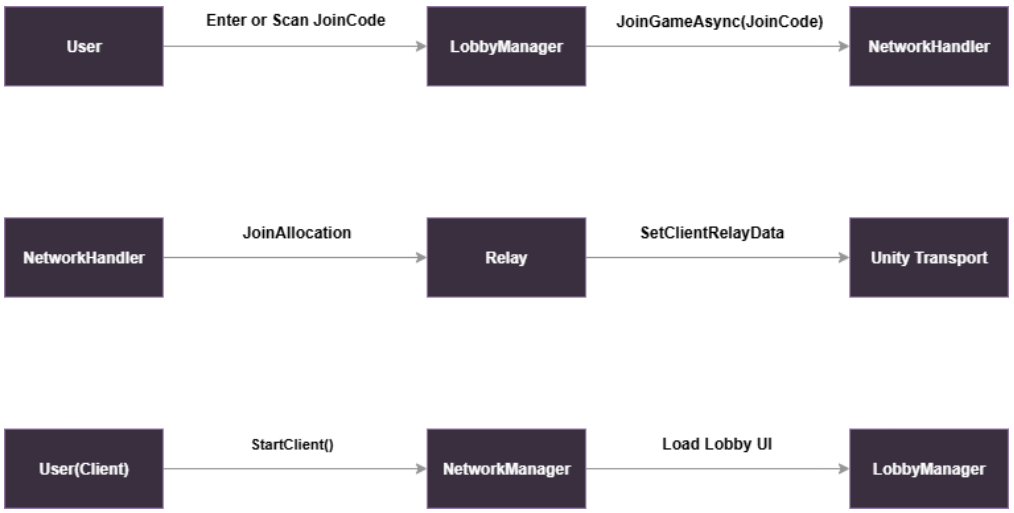
3.1 Ροές στο παιχνίδι

- **Host Game**: Ο παίκτης δημιουργεί μια συνεδρία και λαμβάνει έναν κωδικό. Μοιράζεται τον κωδικό με τους υπόλοιπους παίκτες για να μπούν στην ίδια συνεδρία. Η ροή που ακολουθείται είναι:



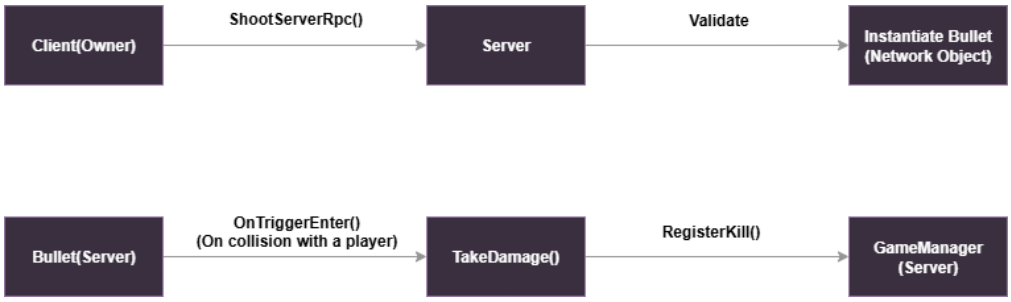
Ροή 1: Host Game

- **Join Game**: Ο παίκτης εισάγει τον κωδικό που του δώθηκε από έναν Host για να μπει στην ίδια συνεδρία. Ακολουθείται η παρακάτω ροή:



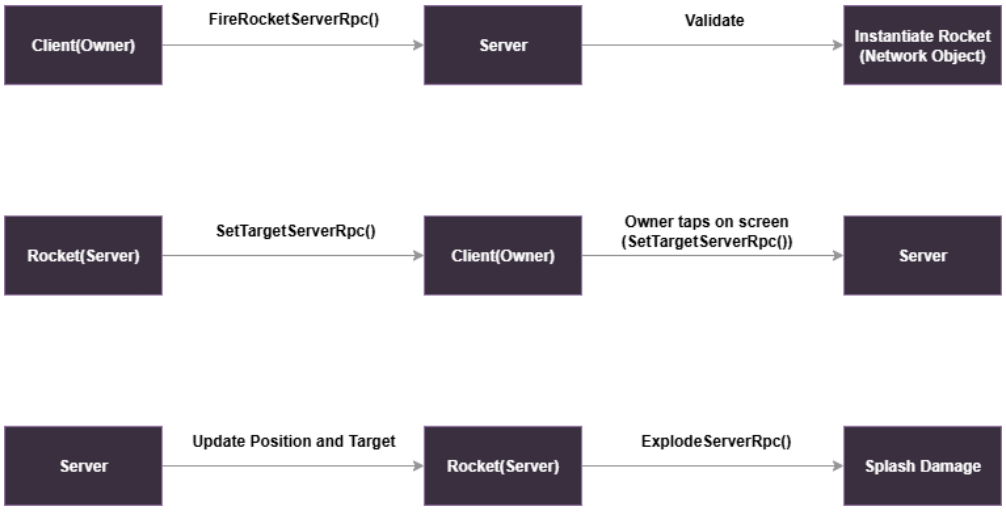
Ροή 2: Join Game

- **Bullets**: Ένας Client για να πυροβολήσει μια σφαίρα, στέλνει αίτημα προς τον Server, ο server εγκρίνει το αίτημα και εμφανίζει την σφαίρα. Παρακάτω φαίνεται η ροή μιας σφαίρας:



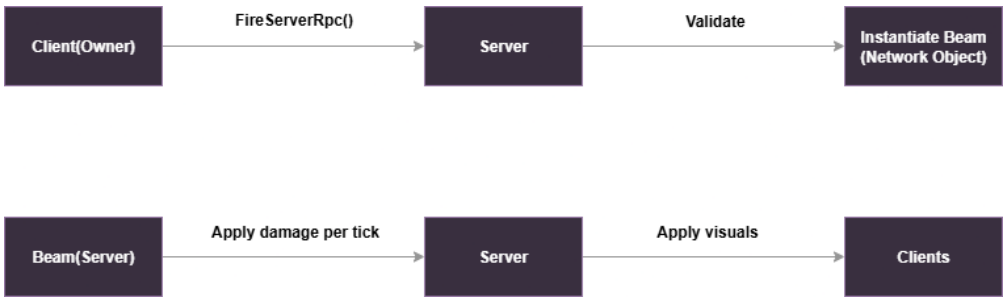
Ροή 3: Εκτόξευση σφαίρας (Bullet)

- **Rocket + Retarget**: Κατά την εκτόξευση του πυραύλου, ο Client στέλνει αίτημα προς τον Server, αυτός εγκρίνει και εμφανίζει τον πύραυλο. Ο πύραυλος μπορεί να ανακατευθυνθεί, πατώντας κάποιο σημείο στον χώρο Κάθε ανακατεύθυνση είναι ένα αίτημα από τον Client στον Server. Παρακάτω φαίνεται η ροή ενός πυραύλου:



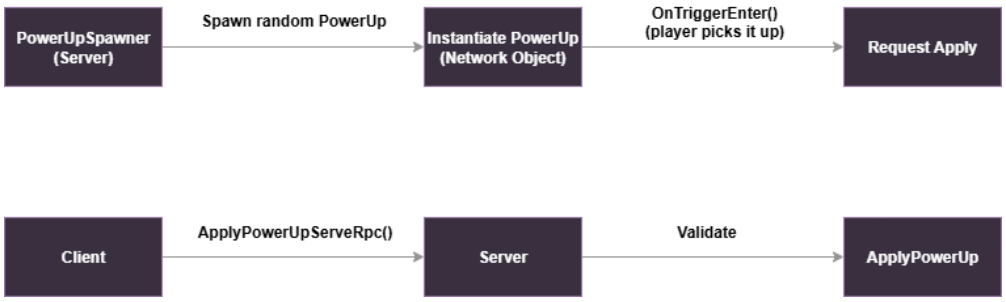
Ροή 4: Εκτόξευση πυραύλου (Rocket)

- **Electric Beam**: Κατά την ενεργοποίηση του ηλεκτρικού τόξου, όπως και στα υπόλοιπα όπλα, ο Client στέλνει αίτημα προς τον Server, αυτός εγκρίνει και εμφανίζει το ηλεκτρικό τόξο. Παρακάτω φαίνεται η ροή ενός ηλεκτρικού τόξου:



Ροή 5: Ενεργοποίηση ηλεκτρικού τόξου (Electric Beam)

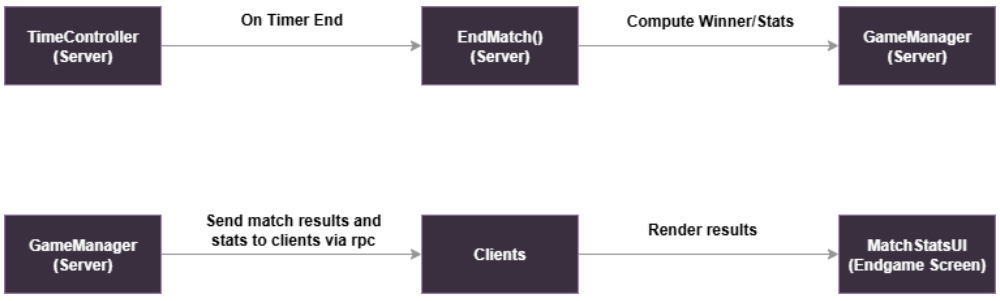
- **Power-Ups**: Τα Power-Ups εμφανίζονται στον χώρο από τον Server. Όταν ένας παίκτης έρθει σε επαφή με αυτά, στέλνει αίτημα προς τον Server και ενεργοποιείται το Power-Up. Παρακάτω φαίνεται η ροή ενός Power-Up:



Ροή 6: Power-Ups

- **GameFlow**: Η γενική ροή ενός παιχνιδιού έχει ως εξής:

1. Οι παίκτες εισέρχονται στον χώρο
2. Πολεμούν μεταξύ τους
3. Ο Server κρατάει στατιστικά και χρόνο που απομένει
4. Όταν τελειώσει ο χρόνος, καλείται η EndMatch(). Ο Server υπολογίζει τα στατιστικά και τον νικητή, και μέσω ClientRpc τα στέλνει τα απαραίτητα δεδομένα στους χρήστες στην συνεδρία. Η ροή φαίνεται παρακάτω:



3.2 Σχεδίαση παιχνιδιού

Οι παίκτες συνδέονται σε lobby, εκκινούν αγώνα 2 ομάδων (Blue vs Red) και ανταγωνίζονται με τρία όπλα (bullets/rocket/beam). Κατά τη διάρκεια εμφανίζονται Power-Ups κάθε ~10s. Στο τέλος νικά η ομάδα με περισσότερα kills. Σε περίπτωση ισοβαθμίας υπερισχύει η ομάδα με λιγότερα deaths. Σε περίπτωση ισοβαθμίας είναι ισοπαλία. Μετά την προβολή αποτελεσμάτων, οι παίκτες επιστρέφουν στο Lobby.

Όπως έχει αναφερθεί και πριν, τα όπλα για κάθε παίκτη είναι 3. Κάθε όπλο έχει διαφορετική λειτουργία, φαίρνοντας στρατιγική στο παιχνίδι:

Όπλο	Περιγραφή
Bullets	Βασική ζημιά: 10 HP ανά βολή Διάρκεια ζωής: 2s.
Rocket	Splash damage, έκρηξη σε σύγκρουση ή μετά από 5δ, Δυνατότητα ανακατεύθυνσης -> Η ζημιά μειώνεται κατά 50%, η ταχύτητα αυξάνεται κατά 50%
Beam	Beam Duration 3; damagePerTick 10; Κατά την διάρκεια ο παίκτης είναι πολύ αργός

Πίνακας 1: Όπλα και Πληροφορίες

Στο παιχνίδι υπάρχουν και πολλά Power-Ups. Τα Power-Ups βοηθάνε τον παίκτη, ή αλλαζουν την ροή του παιχνιδιού, ώστε να φέρνει ενδιαφέρον στο παιχνίδι:

Power-Up	Ενέργεια
SlowMotion	Επιβράδυνση του χρόνου παιχνιδιού ×0.5 για 5s
HealthRegen	+5hp/δευτερόλεπτο για 10s
Invincibility	Απρόσβλητος από ζημιά για 10s
Invisibility	Αόρατος για 10s
SpeedBoost	Ταχύτητας κίνησης x1.5 για 10s
DamageMultiplier	Ζημιά bullets x2 για 10s

Πίνακας 2: Power-Ups και Ενέργειες

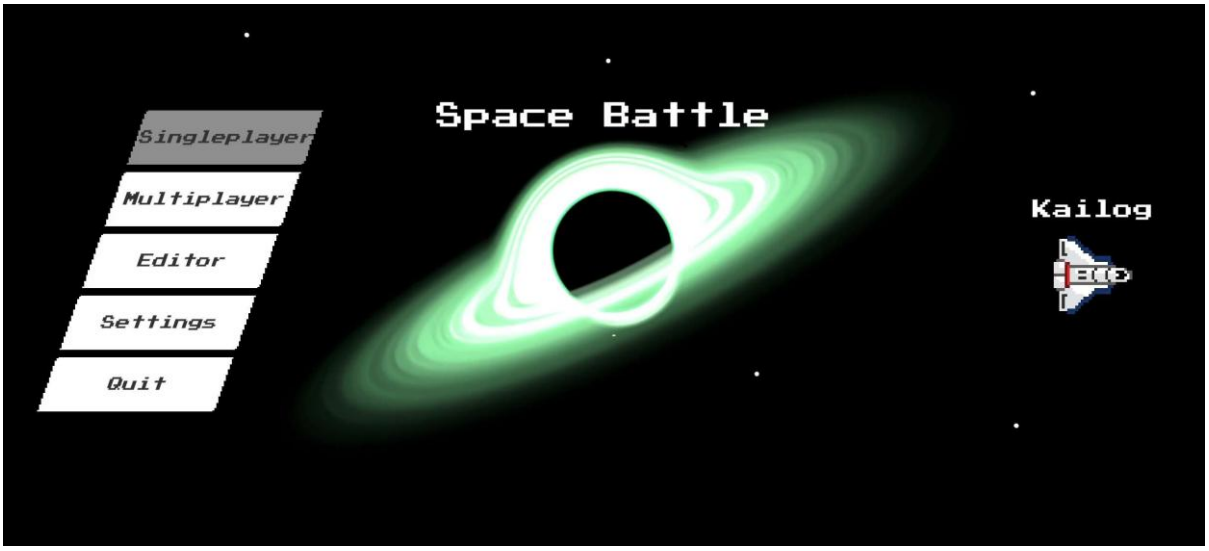
4. Υλοποίηση

Για την υλοποίηση του Space Battle, το πρότζεκτ χωρίζεται σε κομμάτια, όπως σκηνές, αντικείμενα και διαχειριστές.

4.1 Σκηνές

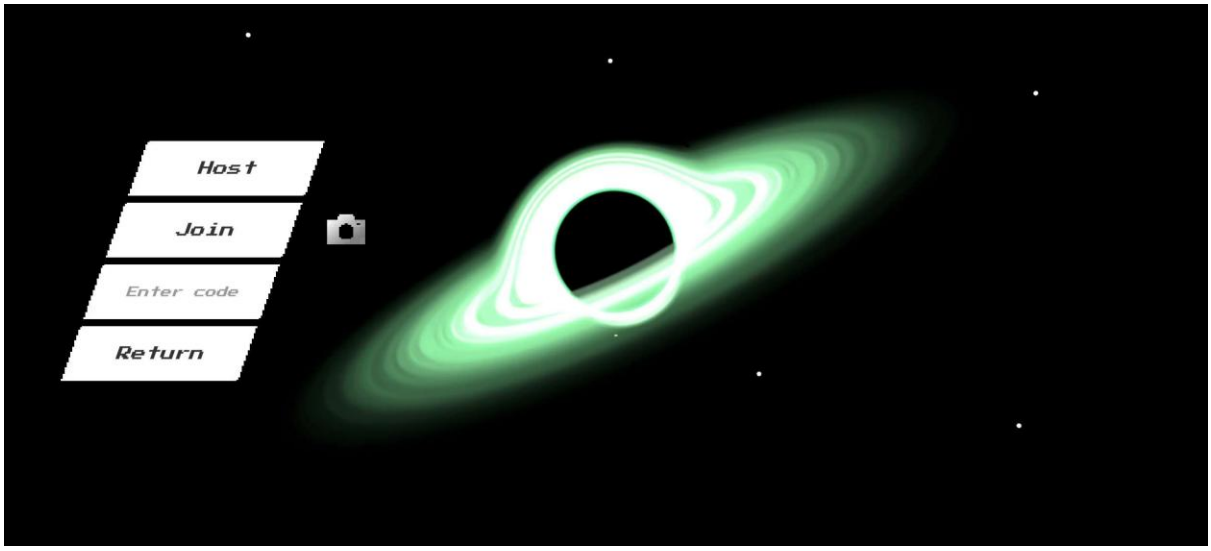
Μια σκηνή (Scene) στο Unity είναι ουσιαστικά ένα κόσμος που περιέχει όλα τα αντικείμενα (GameObjects) που συνθέτουν ένα μέρος του παιχνιδιού. Στο Space Battle έχουμε 4 σκηνές συνολικά:

- Main Menu: Η πρώτη σκηνή που φορτώνεται κατα την εκκίνηση του παιχνιδιού είναι η σκηνή με το μενού πλοήγησης. Σε αυτό έχουμε τις επιλογές
    - Singleplayer: μη διαθέσιμο προς το παρόν, θα επεκταθεί στο μέλλον
    - Multiplayer: Μετάβαση στην σκηνή Lobby
    - Editor: Μετάβαση στην σκηνή Editor
    - Settings: Εμφανίζει μενού με ρυθμίσεις (ρυθμίσεις ήχου, ενεργοποίηση/απενεργοποίηση on-screen controls)
    - Quit: Έξοδος από το παιχνίδι
    - Player customisation: Μενού με τα στατιστικά του παίκτη και προσαρμογή της εμφάνισης του διαστημόπλοιου



Εικόνα 1: Main Menu Scene

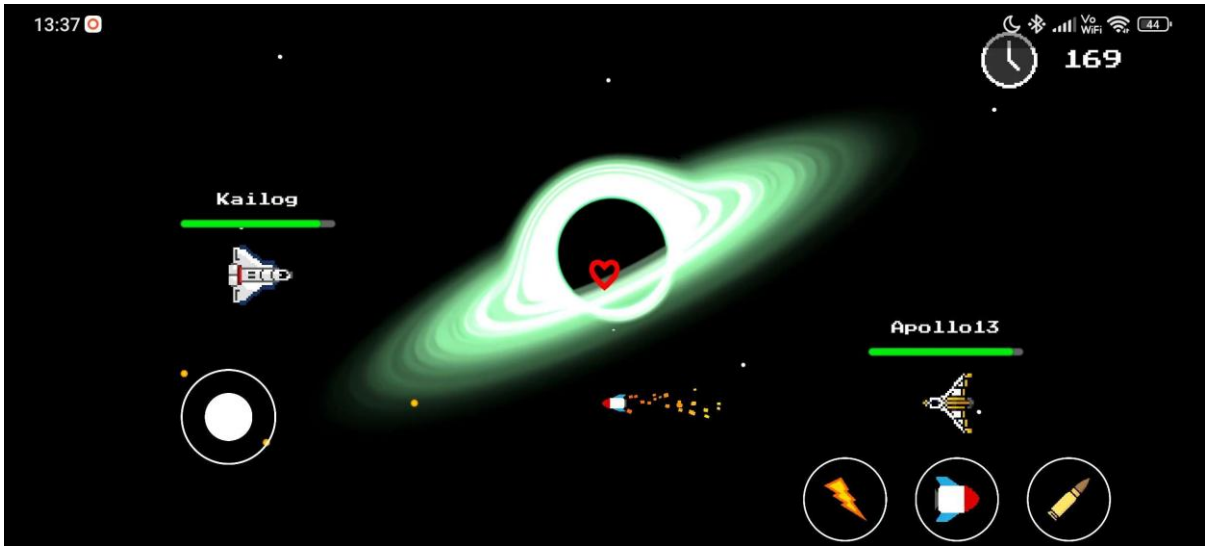
- Lobby Scene: Είναι η σκηνή που δίνει την δυνατότητα να φτιάξουμε ή να μπούμε σε ένα lobby. Για να μπούμε σε ένα lobby πρέπει να εισάγουμε τον κωδικό ή να σκανάρουμε το QR Code που μας δίνει ο Host



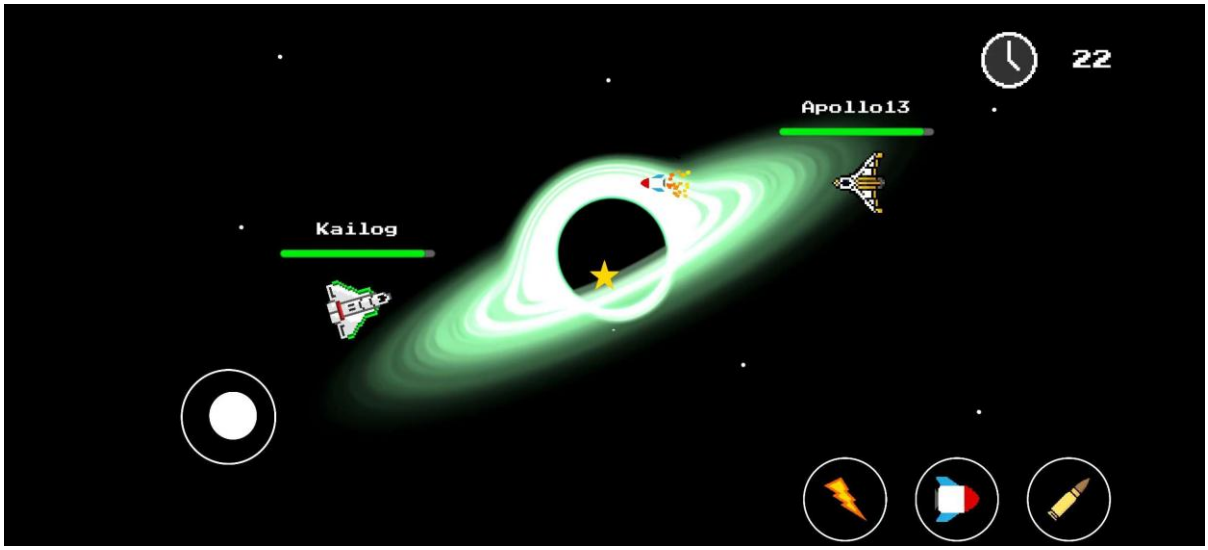
Εικόνα 2: Lobby Scene

Εικόνα 3: Lobby Scene, στο Lobby

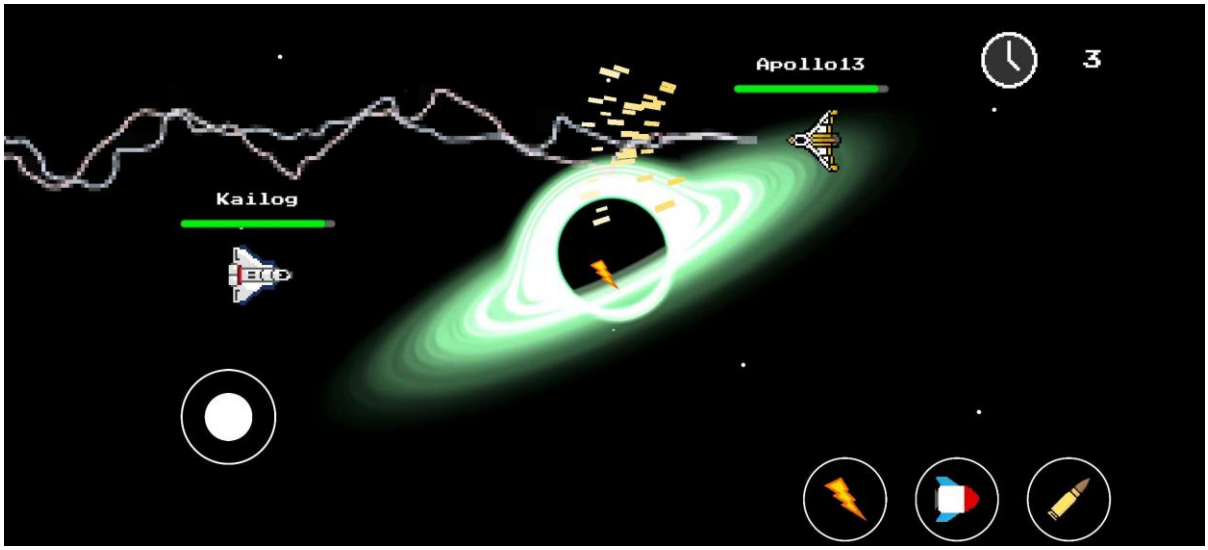
- Game Scene: Σε αυτή την σκηνή εξελίσσεται η μάχη μεταξύ των παικτών



Εικόνα 4: Game Scene



Εικόνα 5: Game Scene (2)



Εικόνα 6: Game Scene (3)

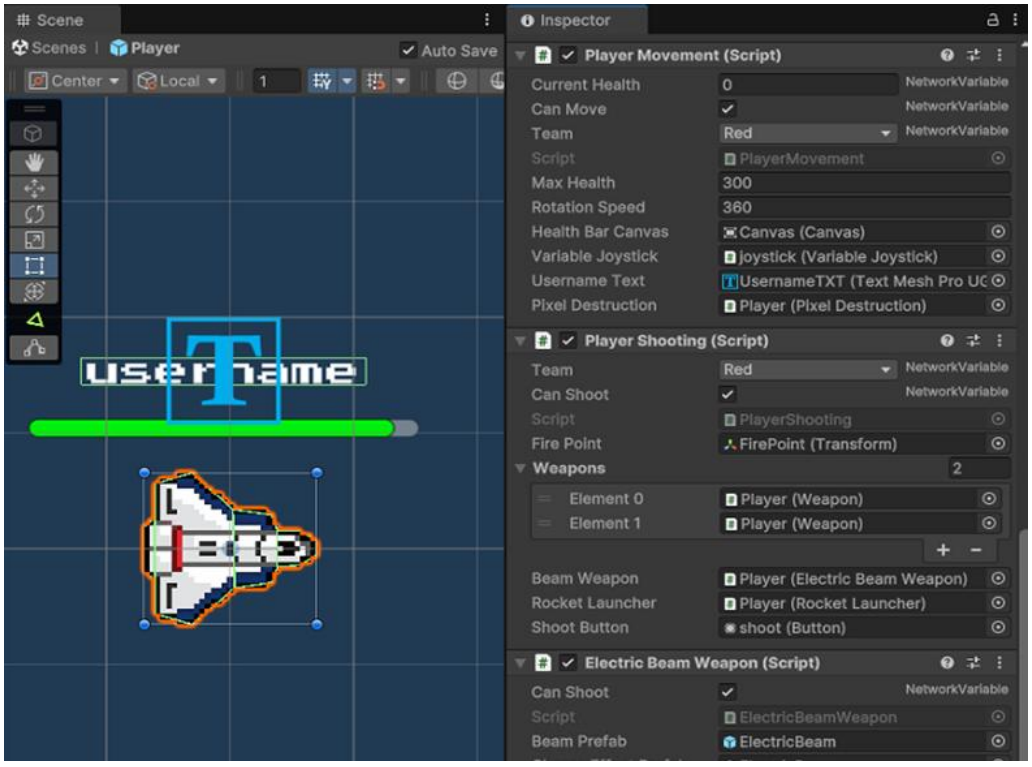
- Editor Scene: Αυτή η σκηνή αφόρα την ρύθμιση των on-screen controls. Τα on-screen controls είναι 4 κουμπιά συνολικά, τα οποία είναι:
    - Joystick: μοχλό για την κίνηση του διαστημόπλοιου
    - ElectricBeam button: κουμπί για την ενεργοποίηση του ηλεκτρικού τόξου
    - Rocket button: κουμπί για την εκτόξευση πυραύλου
    - Fire button: κουμπί για την εκτόξευση σφαιρών

  Για κάθε control μπορούμε να ρυθμίσουμε την τοποθεσία του στην οθόνη και την αδιαφάνεια (ποσοστό διαφάνειας, πόσο διαφανές είναι). Για το joystick μπορούμε επίσης να αλλάξουμε και τον τύπο του.

4.2 Prefabs

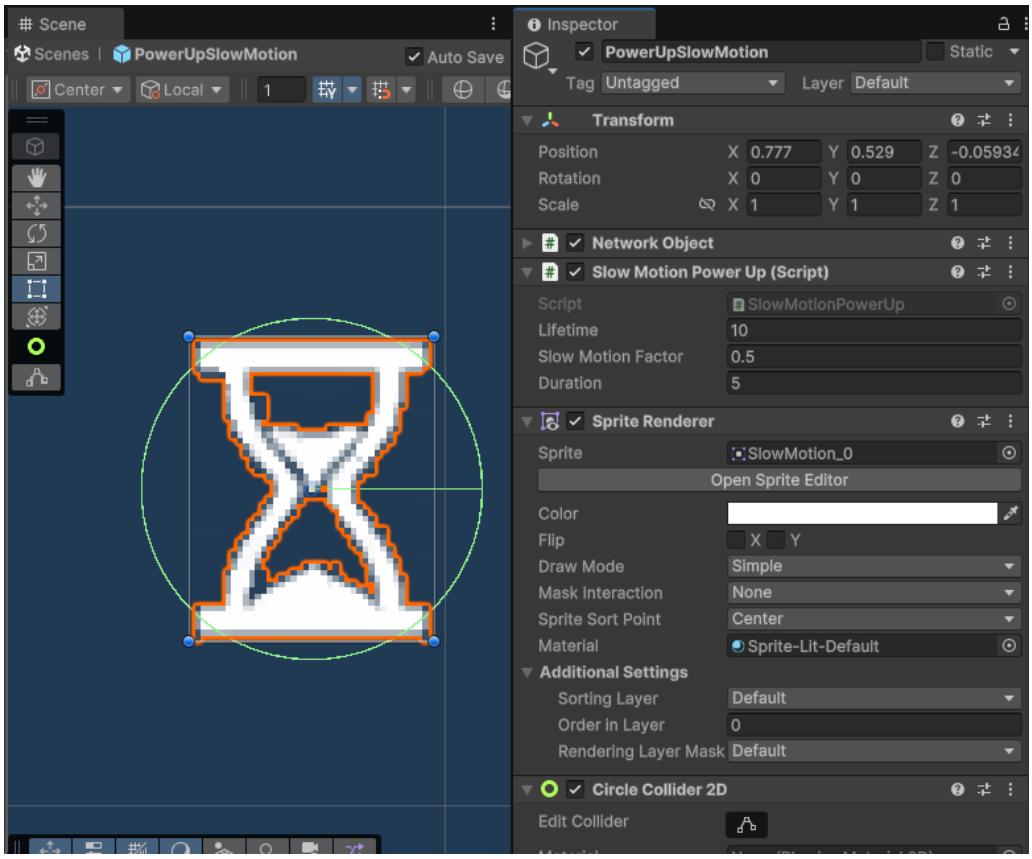
Στο Space Battle υπάρχουν πάρα πολλά prefabs για την δημιουργία του παιχνιδιού. Το prefab είναι ένα έτοιμο πρότυπο αντικειμένου που έχουμε ρυθμίσει (με τα components, scripts, υλικά κ.λπ.) και μπορούμε να το επαναχρησιμοποιήσουμε παντού στο project. Από τα πιο σημαντικα prefabs είναι:

- **Player**: Αυτό είναι το αντικείμενο που εμφανίζεται για κάθε παίκτη που ανήκει σε ομάδα κατά την εκκίνηση του παιχνιδιού. Στην περίπτωση αυτή πρόκειται για μια εικόνα (διαστημόπλοιου), επιπλεόν εξαρτήματα, όπως: collider, healthbar, username text και scripts. Τα scripts είναι μικρά κομμάτια κώδικα συνδεδεμένα σε ένα αντικείμενο. Τα σημαντικότερα scripts του παίκτη είναι:
    - PlayerMovement: Script για την κίνηση του παίκτη στον κόσμο
    - PlayerShooting: Είναι το script υπεύθυνο για τα όπλα του παίκτη



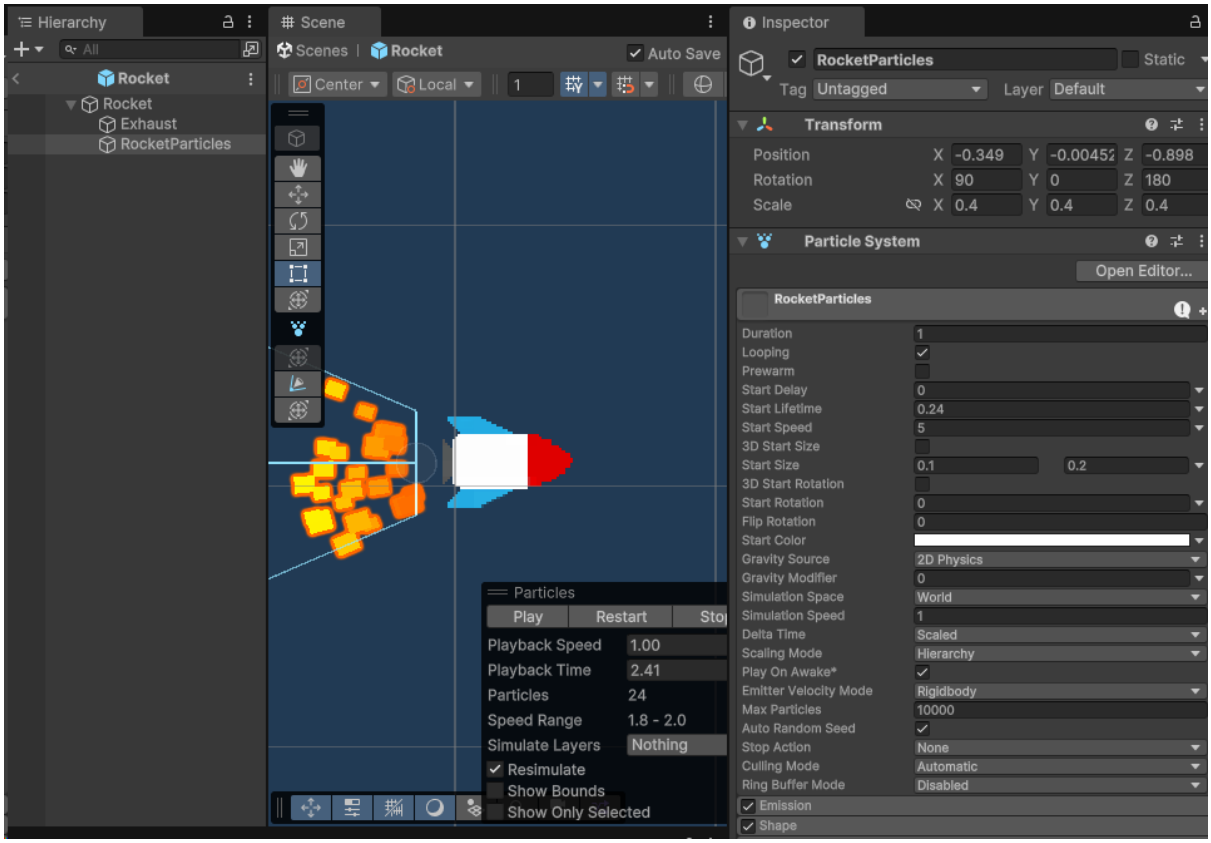
Εικόνα 7: Player Prefab

- **Power-Ups**: Είναι τα βοηθήματα που εμφανίζονται στον κόσμο. Στο Space Battle υπάρχουν πολλά power-ups, όπου όλα έχουν την ίδια δομή συνήθως. Όλα τα power-ups περιέχουν:
    - Network Object: Είναι script της NGO το οποίο καθορίζει το αντικείμενο ως ορατό στο δίκτυο, δηλαδή υπάρχει για όλους
    - Script: Το προσωπικό script για κάθε power-up, το οποίο είναι υπεύθυνο για το αποτέλεσμα που θα φέρει
    - Sprite-Renderer: Είναι η εμφάνιση του αντικειμένου
    - Collider: το “αόρατο σχήμα” ενός αντικειμένου που χρησιμοποιεί η Unity για ανίχνευση σύγκρουσης/επαφής.



Εικόνα 8: SlowMotion Prefab

- **Rocket**: Είναι το αντικείμενο που δημιουργείται κατά την εκτόξευση του πυραύλου. Ο πύραυλος, όπως και ο παίκτης, αποτελείται από πολλά εξαρτήματα. Εκτός από εμφάνιση, scripts κ.λπ. περιέχει επίσης particle system, το οποίο είναι ένα σύστημα που δημιουργεί πολλά μικρά σωματίδια (particles) για να προσομοιώσει φαινόμενα όπως φωτιά, καπνό, σπίθες κ.λπ.



Εικόνα 9: Rocket Prefab

4.3 Διαχειριστές (Managers)

Στο Space Battle υπάρχουν επίσης παρά πολλοι διαχειριστές και είναι πολύ κρίσιμοι. Οι διαχειριστές είναι «άδεια» αντικείμενα, τα οποία δεν έχουν κάποια εμφάνιση και στην συγκεκριμένη περίπτωση σκοπός τους είναι να εκτελέσουν κάποιο/α script/s. Στο παιχνίδι μερικοί διαχειριστές είναι:

- **PlayerDataManager**: Είναι υπεύθυνος για την διαχείριση των δεδομένων του χρήστη. Φορτώνει, κρατάει και αποθηκεύει δεδομένα όπως: username, kills, deaths, level κ.λπ.

- **GameSessionManager**: Είναι υπεύθυνος για την διαχείριση των πληροφοριών των παικτών στην συνεδρία. Κάθε παίκτης κατά την σύνδεσή του σε ένα lobby, μοιράζεται με τους υπόλοιπους πληροφορίες/μεταβλητές όπως: username, level, platform

- **GameManager**: Είναι υπεύθυνος για την διαχείριση της κατάστασης του παιχνιδιού (match) και τα στατιστικά των παικτών

5. Αξιολόγηση, Συμπεράσματα και Μελλοντικές επεκτάσεις

5.1 Αξιολόγηση

Περιβάλλον & Ρυθμίσεις:

- Unity 6000.0.29f1
- Netcode for GameObjects 2.5
- Δίκτυο: Wi-Fi (Starlink)
- Συσκευές:
    - Laptop (Ryzen 7 7530U, Integrated Graphics)
    - Phone (Snapdragon 865, Adreno 650)
- Στόχος: 60FPS
- Διάρκεια αγώνα: 1 λεπτό

Με βάση τα τοπικά αρχεία από τις μετρήσεις κατά την διάρκεια του αγώνα έχουμε:

Συσκευή - Σενάριο	**FPS (μέσο)**	**Frame-time p50 (ms)**	**Frame-time p95 (ms)**
Laptop(Host)	58.1	16.67	16.87
Phone(Host)	59.7	16.73	16.86

Πίνακας 3: Αποτελέσματα απόδοσης

Παρατηρούμε ότι και στα δύο σενάρια ο μέσος ρυθμός καρέ παραμένει ομαλός. Κατά την δοκιμή στο λάπτοπ, παρατηρήθηκε κόλλημα όταν κάποιος παίκτης επαίρνε ζημία από το ηλεκτρικό τόξο. Αυτό οφείλεται στο ότι γίνονται πολλοι υπολογισμοί για το ηλεκτρικό τόξο και το λάπτοπ ήταν σε λειτουργία μπαταρίας (μειωμένη απόδοση). Όταν το λάπτοπ ήταν στην πρίζα, αυτό το φαινόμενο δεν υπήρχε.

Συσκευή - Σενάριο	**Packets Sent**	**Packets Received**	**Packet Loss%**	**RTT Median (ms)**	**RTT p95 (ms)**	**Jitter (ms)**
Phone(Client)	62	62	0	167.32	202.42	22.05
Laptop(Client)	60	60	0	165.43	201.34	21.97

Πίνακας 4: Αποτελέσματα μετρήσεων δικτύου

Παρατηρούμε ότι και στα δύο σενάρια έχουμε εξίσου καλή σύνδεση. Τα νούμερα είναι αρκετά υψηλά, κάτι που δεν είναι ιδανικό για multiplayer shooter. Αυτό οφείλεται στο ότι οι δοκιμές έγιναν σε δίκτυο Starlink, όπου η κίνηση περνάει από δορυφόρους στο διάστημα. Το host-authorative μοντέλο επίσης προσθέτει επιπλέον (μικρή) καθυστέρηση.

Όρος	**Εξήγηση**
FPS	Καρέ κάθε δευτερόλεπτο
Frame-time p50 (ms)	Ο μέσος χρόνος για να εμφανιστεί ένα καρέ
Frame-time p95 (ms)	Ο χρόνος κάτω από τον οποίο βρίσκονται το 95% των καρέ - δείχνει τις πιο “βαριές” στιγμές
Packets Sent	Πλήθος πακέτων που στάλθηκαν
Packets Received	Πλήθος πακέτων που παραλήφθηκαν
Packet Loss %	Ποσοστό πακέτων που χάθηκαν στην διαδρομή
RTT Median (ms)	Μέσος χρόνος μετάβασης και επιστροφής ενός πακέτου
RTT p95 (ms)	Χρόνος κάτω από τον οποίο πέφτει το 95% των μετρήσεων RTT - δείχνει καθυστερήσεις σε φορτίο
Jitter (ms)	Μεταβολή του χρόνου απόκρισης. Όσο πιο κοντά στο 0, τόσο πιο σταθερή σύνδεση

Πίνακας 5: Ορολογία μετρήσεων

5.2 Συμπεράσματα

Το Space Battle δείχνει ότι ένας 2D τίτλος που εστιαζεί κυρίως σε κινητά με host-authoritative μοντέλο μπορεί να διατηρήσει ομαλή εμπειρία ακόμη και με συνθήκες δορυφορικής τάξης μέσω Starlink/Relay. Η καθαρή οριοθέτηση αυθεντίας με Unity Services (Lobby/Relay) + NGO επιτάχυνε την υλοποίηση και διασφάλισε συνεπή κατάσταση.

5.3 Μελλοντικές επεκτάσεις

Για την βελτίωση του Space Battle προτείνονται οι εξής ιδέες:

- **Εμπλουτισμός περιεχομένου**: Προφανώς και σε ένα παιχνίδι υπάρχουν άπειρες ιδέες για επεκτάσεις. Τέτοια επέκταση θα μπορούσε να είναι η προσθήκη singleplayer, όπλων, Power-Ups, διαφορετικους χάρτες, εμφανίσεις κ.λπ.
- **Dedicated Server**: Για σταθερότερους χρόνους.
- **Βάση δεδομένων στο Cloud**: Καθε χρήστης θα μπορούσε να έχει τον δικό του λογαριασμό σε βάση δεδομένων, όπου αυτή την φορά θα αποθηκεύονται και δεδομένα που δίνουν κάποιο ανταγωνιστικό πλεονέκτημα.

Πίνακας Ορολογίας

Ξενόγλωσσος Όρος	Ελληνικός Όρος
Server	Διακομιστής
Host	Διοργανωτής
Client	Πελάτης
Host-authoritative	Μοντέλο όπου ο host κρατά την «πηγή αλήθειας»
Client-side prediction	Πρόβλεψη πελάτη
Reconciliation	Συμφιλίωση
Interpolation	Παρεμβολή
Lag compensation	Αντιστάθμιση καθυστέρησης
Tick rate	Συχνότητα ενημερώσεων
Jitter	Μεταβλητότητα του χρόνου άφιξης πακέτων
Packet Loss	Ποσοστό χαμένων πακέτων
RTT (Round-Trip Time)	Χρόνος μετάβασης πακέτου
RPC (Remote Procedure Call)	Κλήση συνάρτησης απομακρυσμένα
NetworkVariable	Συγχρονιζόμενη μεταβλητή του NGO
Reliable / Unreliable channels	Αξιόπιστα / μη αξιόπιστα κανάλια
Listen server	Μοντέλο όπου ενας παίκτης τρέχει και client και server στον ίδιο κόμβο (ο Host)
Dedicated server	Αυτόνομος server
Lobby	Αίθουσα / χώρος αναμονής
Relay	Μεσολαβητής
Session	Συνεδρία
UTP (Unity Transport)	Βιβλιοθήκη επιπέδου μεταφοράς
Prefab	Πρότυπο αντικειμένου στην Unity

Βιβλιογραφία

- *Unity Technologies, “The Ultimate Advanced Multiplayer Networking Guide,” Dec. 10, 2024. unity.com/resources/ultimate-guide-advanced-multiplayer-networking*

- *Unity Technologies, Netcode for GameObjects — Manual (v2.5). https://docs.unity3d.com/Packages/com.unity.netcode.gameobjects@2.5/manual/index.html*

- *Unity Technologies, Relay — Introduction. docs.unity.com/ugs/manual/relay/manual/introduction*

- *Unity Technologies, Lobby — Manual. docs.unity.com/ugs/manual/lobby/manual/unity-lobby-service*

- *C. Hansen, N. Jurgens, D. Makaroff, D. Callele, and P. Dueck, “Network Performance Measurement Framework for Real-Time Multiplayer Mobile Games”, 2013.*

- *T. Michail and E. Alepis, “Design of Real-Time Multiplayer Word Game for the Android Platform Using Firebase and Fuzzy Logic,” 2023 14th Int. Conf. on Information, Intelligence, Systems & Applications (IISA), 2023.*

- *Hyeong-Gyu Jang, Sung-Yun Park, Sang-Kwang Lee “Predicting Player of the Game in Multiplayer Online Battle Arena Games”, 2025*

- *Steven Schmidt, Saman Zadtootaghaj, Saeed Shafiee Sabet, Sebastian Möller “Modeling and Understanding the Quality of Experience of Online Mobile Gaming Services”, 2021*

Παραρτήματα

Παράρτημα Α - Δεδομένα & Δομές Αποθήκευσης

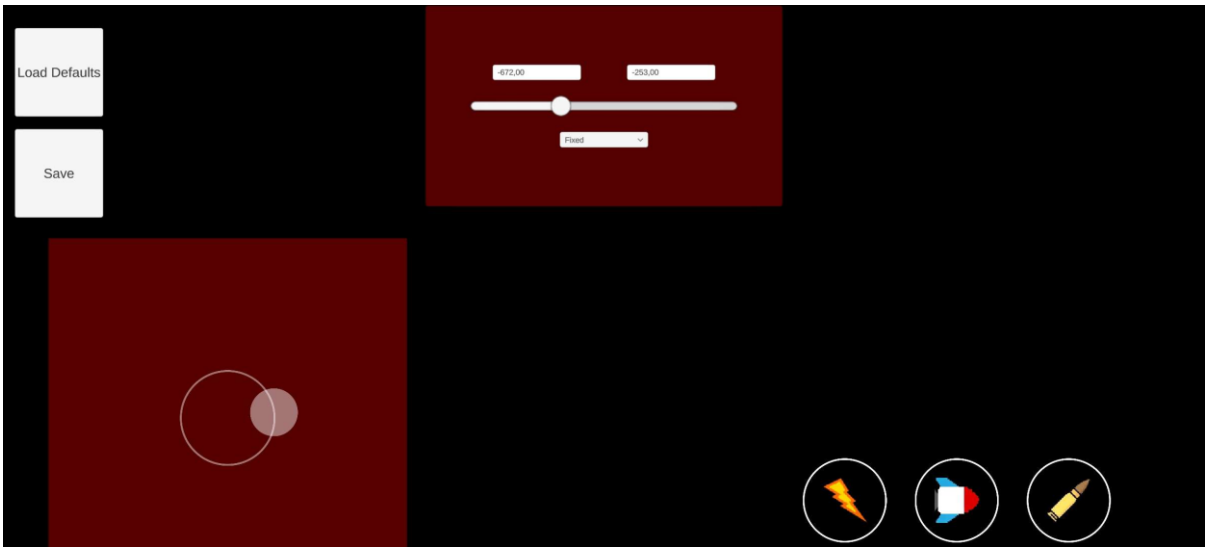

```
{
  "Apollo13": {
    "username": "Apollo13",
    "level": 2,
    "kills": 12,
    "deaths": 3,
    "score": 120,
    "skin": "fighter.png",
    "matchesPlayed": 2
  },
  "Godiak": {
    "username": "Godiak",
    "level": 1,
    "kills": 0,
    "deaths": 0,
    "score": 0,
    "skin": "spaceship.png",
    "matchesPlayed": 0
  },
  "LastSelected": "Apollo13",
  "ShowControls": true,
  "SfxVolume": 0.348684222,
  "MusicVolume": 0.0
}
```


Εικόνα 10: Αρχείο με τα δεδομένα του παίκτη - playerData.json

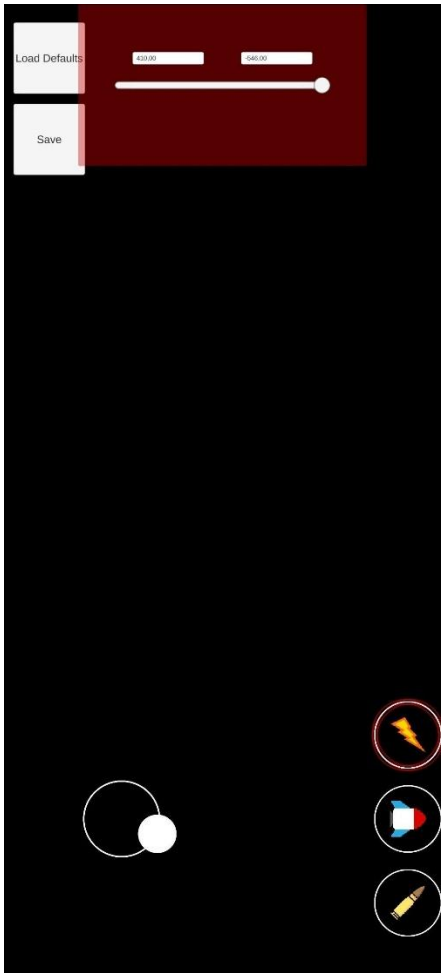
```
{
  "buttonScale": 3,
  "buttonsPortrait": {
    "joystick": {
      "size": 2,
      "buttonSize": 3,
      "type": "Fixed",
      "anchor": "Center",
      "offsetX": -223.0,
      "offsetY": -732.0,
      "opacity": 100
    },
    "shoot": {
      "size": 3,
      "buttonSize": 0,
      "type": null,
      "anchor": "Center",
      "offsetX": 410.0,
      "offsetY": -918.0,
      "opacity": 100
    },
    "rocket": {
      "size": 3,
      "buttonSize": 0,
      "type": null,
      "anchor": "Center",
      "offsetX": 410.0,
      "offsetY": -732.0,
      "opacity": 100
    },
    "beam": {
      "size": 3,
      "buttonSize": 0,
      "type": null,
      "anchor": "Center",
      "offsetX": 410.0,
      "offsetY": -546.0,
      "opacity": 100
    }
  },
  "buttonsLandscape": {
    "joystick": {
      "size": 2,
      "buttonSize": 3,
      "type": "Fixed",
      "anchor": "Center",
      "offsetX": -640.0,
      "offsetY": 95.0,
      "opacity": 100
    },
    "shoot": {
      "size": 3,
      "buttonSize": 0,
      "type": null,
      "anchor": "Center",
      "offsetX": 830.0,
      "offsetY": -400.0,
      "opacity": 100
    },
    "rocket": {
      "size": 3,
      "buttonSize": 0,
      "type": null,
      "anchor": "Center",
      "offsetX": 603.0,
      "offsetY": -96.0,
      "opacity": 100
    },
    "beam": {
      "size": 3,
      "buttonSize": 0,
      "type": null,
      "anchor": "Center",
      "offsetX": 422.0,
      "offsetY": -152.0,
      "opacity": 100
    }
  }
}
```


Εικόνα 11: Αρχείο με πληροφορίες των διατάξεων των κουμπιών - uiSettings.json

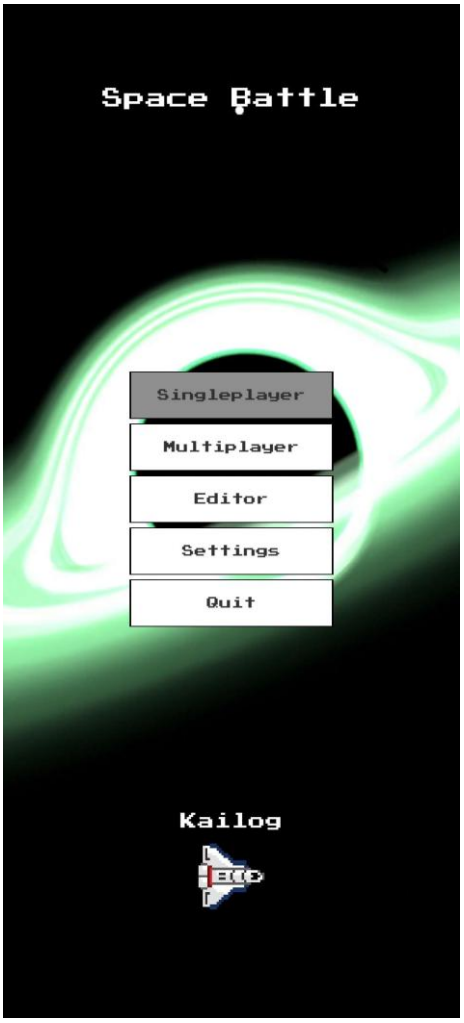
Παράρτημα Β - UI & Διατάξεις



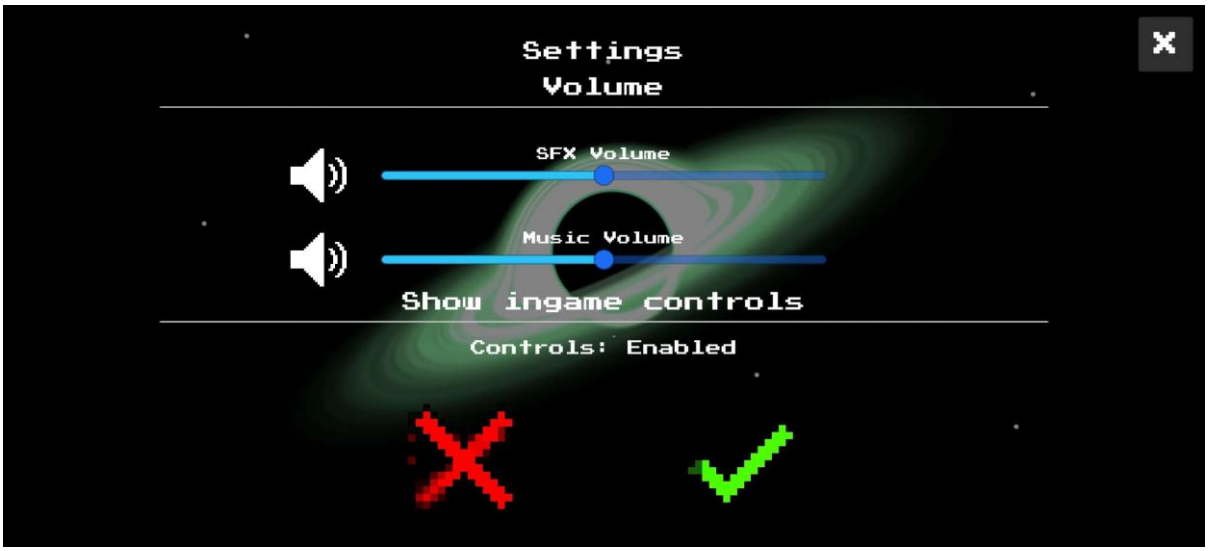
Εικόνα 12: Editor Scene



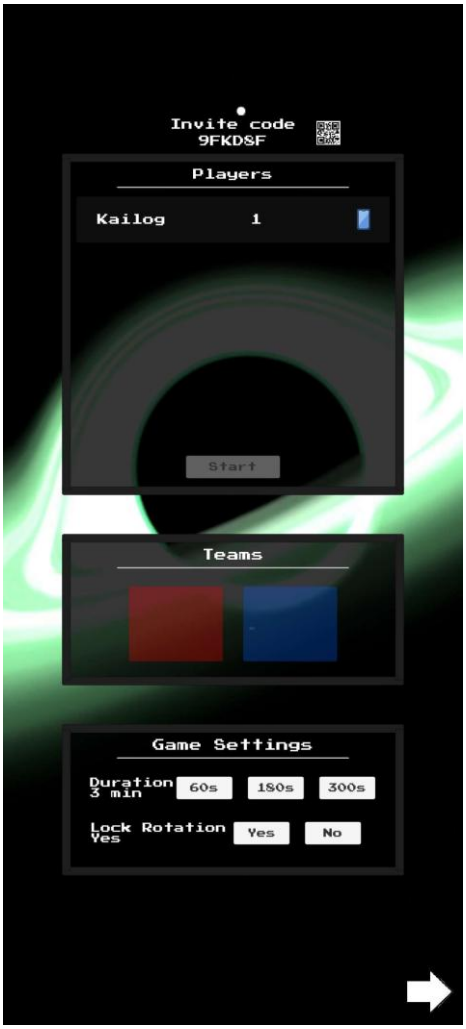
Εικόνα 13: Editor Scene (Κάθετος προσανατολισμός)



Εικόνα 14: Αρχικό μενού (Κάθετος προσανατολισμός)

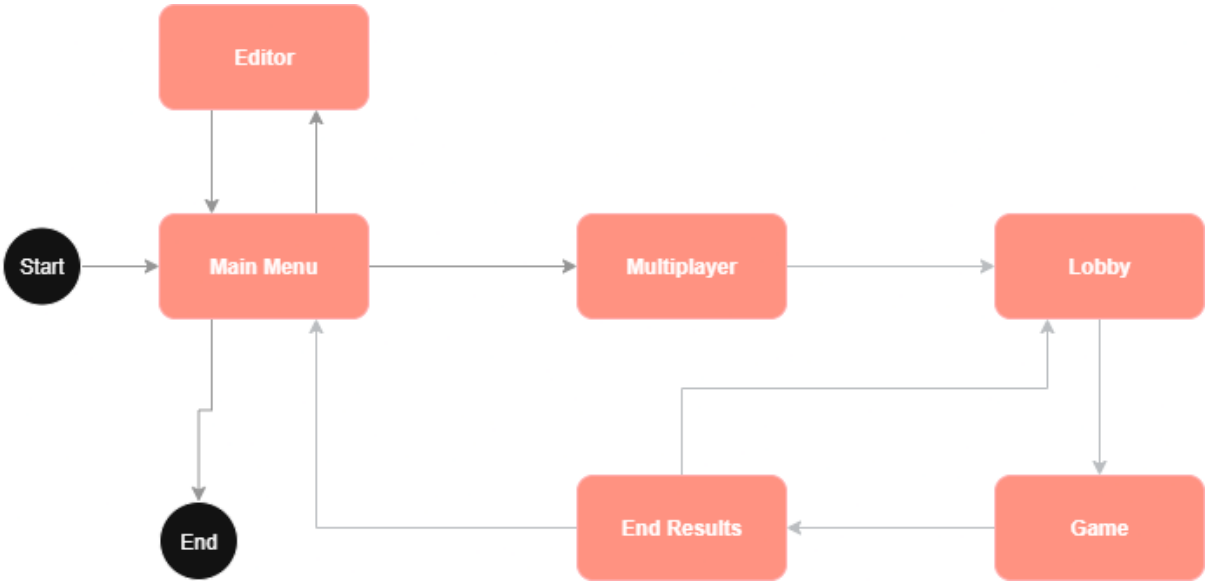


Εικόνα 15: Μενού ρυθμίσεων



Εικόνα 16: Στο Lobby (Κάθετος προσανατολισμός)

Εικόνα 17: Μενού προσαρμογής εμφάνισης και στατιστικών παίκτη



Ροή 7: Διάγραμμα καταστάσεων σκηνών

Παράρτημα Γ - Τρίτα Στοιχεία

Assets:

Joystick Pack: Fenerax Studios – https://assetstore.unity.com/packages/tools/input-management/joystick-pack-107631

Sounds & Music:

Sfxr – Sfxr.me

Pixabay – Pixabay.com:

- Retro Acrade Game Music by MFCC
- 8-bit Arcade Mode by moodmode