Διπλωματική Εργασία

Ανάπτυξη 3D σκοπευτικού παιχνιδιού τρίτου προσώπου με πολλαπλούς παίκτες τύπου Roguelite

Νικολόζι Μετρεβέλι

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΡΟΜΠΟΤΙΚΗΣ, ΕΝΣΩΜΑΤΩΜΕΝΩΝ ΚΑΙ ΟΛΟΚΛΗΡΩΜΕΝΩΝ ΣΥΣΤΗΜΑΤΩΝ

https://arch.ece.uowm.gr

Επιβλέπων καθηγητές:
Δρ. Μηνάς Δασυγένη
Δρ. Κώστας Καρπούζης

Κοζάνη, Οκτώβριος 2023

Περίγραμμα παρουσίασης

- Εισαγωγή
- Ανάλυση παιχνιδιού
- Συγχρονισμός
- Αποτελέσματα
- Συμπεράσματα

Εισαγωγή

Εισαγωγή – Τα παιχνίδια

- Προσφέρουν αμέτρητη ψυχαγωγία και διασκέδαση.
- Βοηθάνε στην αντιμετώπιση του άγχους, της κατάθλιψης και της μοναξιάς.
- Αλλά…

Ειδικά για τα παιχνίδια πολλαπλών παικτών (multiplayer):

- Γίνονται όλο και πιο δημοφιλή
- Διαδραστικός τρόπος κοινωνικοποίησης

Εισαγωγή - Σκοπός

- Ανάπτυξη ενός multiplayer παιχνιδιού για υπολογιστές
- Αντιμετώπιση της δυσκολίας ανάπτυξης multiplayer παιχνιδιών λόγω:
  - Δικτύου
  - Συγχρονισμού καταστάσεων
  - Καθυστέρησης
  - Ανομοιομορφία εξοπλισμού
- Αναλυτική επεξήγηση της διαδικτυακής πλευράς ενός multiplayer παιχνιδιού, λόγω της ελλιπής τεκμηρίωσης που υπάρχει διαδικτυακά

Εισαγωγή – Επεξήγηση τεχνικών όρων του παιχνιδιού

Σκοπευτικό

Εστίαση στην ήττα των εχθρών με την χρήση όπλων.

Τρίτο Πρόσωπο

Ο παίκτης είναι ορατός από την κάμερα.

Πολλαπλοί παίκτες

Δύο ή περισσότερα άτομα παίζουν ταυτόχρονα στο ίδιο περιβάλλον.

Roguelite

Τυχαία δημιουργία γεγονότων, μόνιμη ήττα και ταυτόχρονα μόνιμη πρόοδο.

Εισαγωγή – Τα Roguelike προήλθαν από το παιχνίδι Rogue (1980)

Roguelike

- Τυχαία δημιουργία περιβάλλοντος
- Μόνιμος θάνατος
- Turn based κίνηση σε πλέγμα
- Περιπλοκότητα
- Εστίαση στους εχθρούς
- Έμφαση στην εξερεύνηση

Roguelite

- Πιθανή τυχαία δημιουργία περιβάλλοντος
- Μόνιμος θάνατος με μόνιμη πρόοδο
- Περιπλοκότητα
- Εστίαση στους εχθρούς
- Έμφαση στην εξερεύνηση
- Συνδυασμός με άλλες κατηγορίες παιχνιδιών

Εργαλεία Ανάπτυξης

Τεχνικό κομμάτι:

VS Code

Σχεδιαστικό κομμάτι:

Paint.net

Draw.io

Ανάλυση και περιγραφή

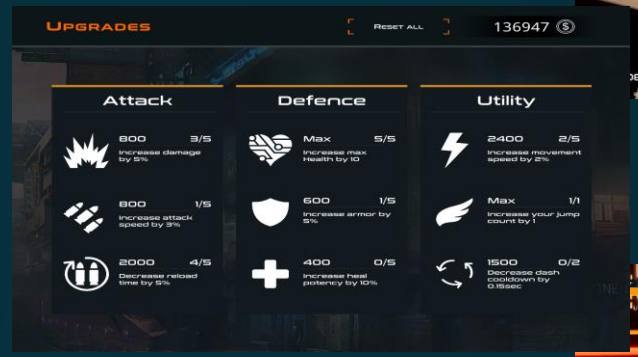
BULLET DRIFTERS

Ανάλυση παιχνιδιού

- 10 γύροι
- 4 διαφορετικά είδη γύρων
- 6 ξεχωριστά όπλα
- Μόνιμες & προσωρινές αναβαθμίσεις

Τυχαιοποίηση των:

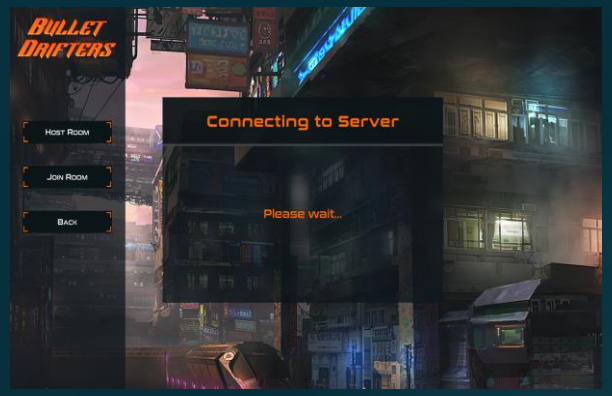
- Εχθρών
- Όπλων και της ποιότητάς τους
- Αναβαθμίσεων και της ποιότητάς τους
- Ειδών γύρου

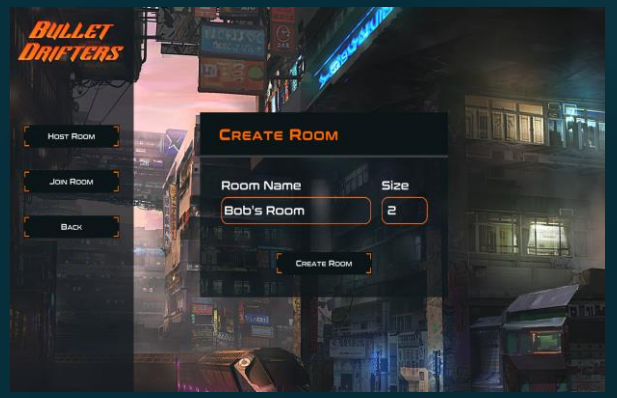


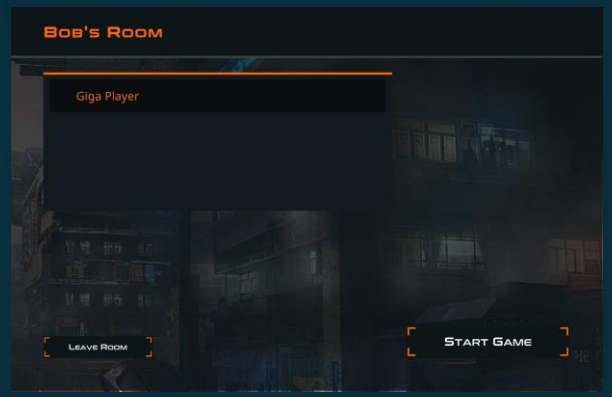
Συγχρονισμός

Διασύνδεση παικτών με διακομιστές

- Σύνδεση σε διακομιστές της Photon
- Οι παίκτες δημιουργούν και συνδέονται σε δωμάτια
- Μόνο ο δημιουργός του δωματίου (host) μπορεί να ξεκινήσει το παιχνίδι
- Μετά την διασύνδεση σε δωμάτια το παιχνίδι είναι peer to peer







Τρόποι συγχρονισμού παικτών

- PhotonView
- Photon Transform View Classic
- Photon Animator View

Unity Editor

- PhotonNetwork.AutomaticallySyncScene
- PhotonView.IsMine
- PhotonNetwork.IsMasterClient

Ιδιότητες κλάσης

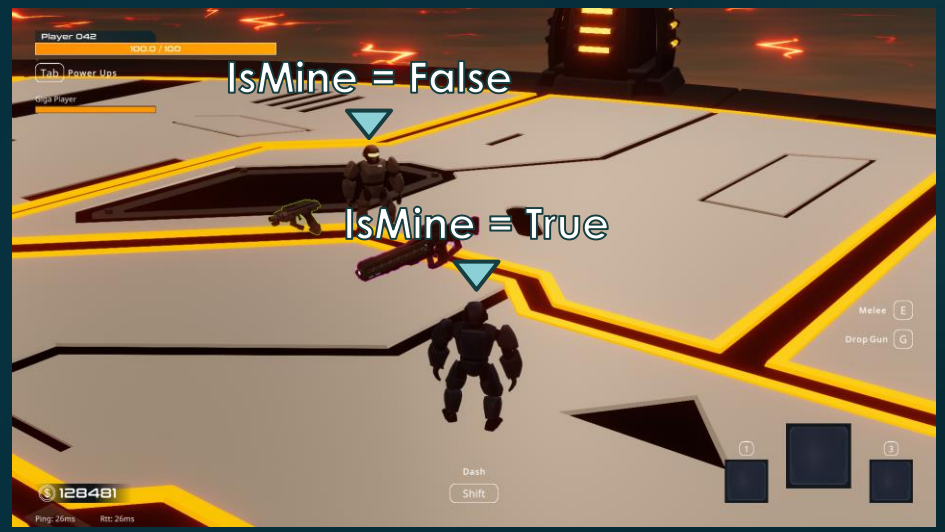
- OnPhotonSerializeView()
- RPC()
- RaiseEvent()

Διαδικτυακές Συναρτήσεις

Χρήση της ιδιότητας PhotonView.IsMine για τον διαχωρισμό του χειρισμού των παικτών

Παίκτης 1

Παίκτης 2



- Αν το PhotonView δεν ανήκει στον παίκτη, τότε επιστρέφουμε από τις συναρτήσεις

```
if (!PlayerPv.IsMine)
    return;
```

Διαδικτυακές συναρτήσεις για την αποστολή μηνυμάτων μεταξύ των παικτών

OnPhotonSerializeView

Καλείται πολλαπλές φορές ανά δευτερόλεπτο για το γράψιμο δεδομένων από τον κάτοχο του PhotonView, και το διάβασμα των δεδομένων από τους υπόλοιπους παίκτες.

Απαραίτητη η ύπαρξη του PhotonView στο αντικείμενο.

RPC

Αποστολή ενός διαδικτυακού μηνύματος με δεδομένα προς τους παίκτες που επιλέγονται, για την σύγχρονη εκτέλεση συναρτήσεων.

Απαραίτητη η ύπαρξη του PhotonView στο αντικείμενο.

RaiseEvent

Αποστολή ενός κωδικού και δεδομένων μέσω ενός διαδικτυακού μηνύματος για την συγχρονισμένη εκτέλεση κώδικα.

Δεν είναι απαραίτητη η ύπαρξη του PhotonView.

Κλήση της OnPhotonSerializeView για τον συχνό συγχρονισμό μεταβλητών


```
public void OnPhotonSerializeView(PhotonStream stream, PhotonMessageInfo info)
{
    if (stream.IsWriting)
    {
        stream.SendNext(health);
        stream.SendNext(maxHealth);
    }
    else
    {
        health = (float)stream.ReceiveNext();
        maxHealth = (float)stream.ReceiveNext();
    }
}
```

Συχνότητα κλήσης συνάρτησης ανά δευτερόλεπτο

- SerializationRate = 10

Συχνότητα αποστολής δεδομένων ανά δευτερόλεπτο

- SendRate = 20

Το IsWriting είναι αληθές μόνο για τον κάτοχο του PhotonView (παρόμοιος έλεγχος με την ιδιότητα IsMine)

- Ο κάτοχος γράφει δεδομένα και τα στέλνει
- Οι υπόλοιποι παίκτες λαμβάνουν και διαβάζουν τα δεδομένα

Κλήση του RPC για την συγχρονισμένη εκτέλεση συνάρτησης

- Δυνατότητα αποστολής δεδομένων και επιλογής στόχου με την ιδιότητα RpcTarget (All, Master, Others)

```
pv.RPC("RPC_PickUpGun", RpcTarget.All, id);
```

- Όποια συνάρτηση καλείται από το RPC πρέπει να χαρακτηριστεί με τον τίτλο [PunRPC]

```
[PunRPC]
0 references
public void RPC_PickUpGun(int id)
{
    GameObject _player = PhotonView.Find(id).gameObject;

    transform.SetParent(_player.transform.GetChild(0));
    transform.localPosition = Vector3.zero;
    transform.localRotation = Quaternion.Euler(Vector3.zero);
    transform.localScale = Vector3.one;
    transform.SetAsFirstSibling();
}
```

Αποστολή και παραλαβή των Events για την συγχρονισμένη εκτέλεση κώδικα

- Αποστολή ενός Event

```
Vector3 randomPosition = new Vector3(Random.Range(-stageRadius, stageRadius), 2.97f, Random.Range(-stageRadius, stageRadius));

object[] content = new object[] { randomPosition };
RaiseEventOptions raiseEventOptions = new RaiseEventOptions { Receivers = ReceiverGroup.All };
PhotonNetwork.RaiseEvent(ActivateHealingPoolCode, content, raiseEventOptions, SendOptions.SendReliable);
```

- Παραλαβή του Event

```
private void OnEvent(EventData photonEvent)
{
    byte eventCode = photonEvent.Code;
    if (eventCode == ActivateHealingPoolCode)
    {
        object[] data = (object[])photonEvent.CustomData;
        Vector3 randomPos = (Vector3)data[0];

        healingPool.transform.position = randomPos;
        healingPool.SetActive(true);
    }
}
```

Παρόμοια με τα RPC:

x Πιο σύνθετη η υλοποίηση των events

✓ Δεν είναι απαραίτητη η χρήση ενός PhotonView

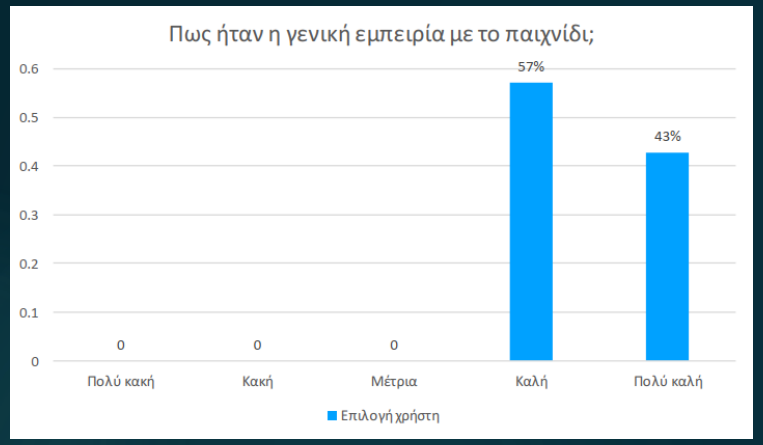
Βελτιστοποίηση επίδοσης

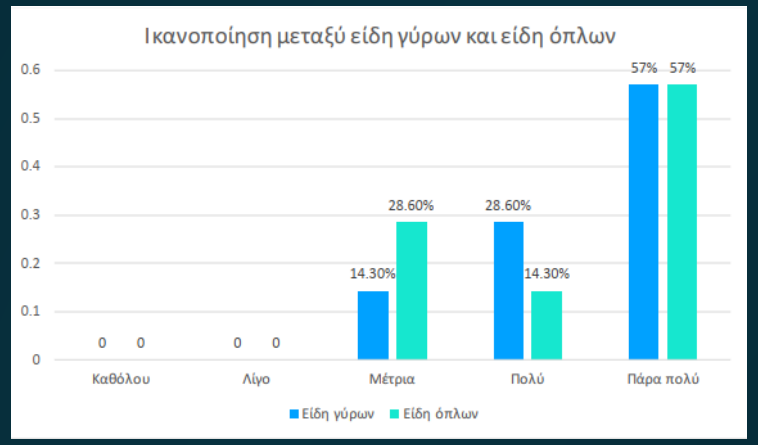
- Αποθηκευτικός χώρος
  - Συμπίεση όλων των textures για την σημαντική μείωση του αποθηκευτικού χώρου από 882 MB στα 334 MB.
- Φωτισμός
  - Χρήση Baked (προϋπολογισμένου) φωτισμού για την αύξηση επιδόσεων
- Object pooling (grouping)
  - Ενεργοποίηση/Απενεργοποίηση για όλες τις σφαίρες για την αύξηση επιδόσεων

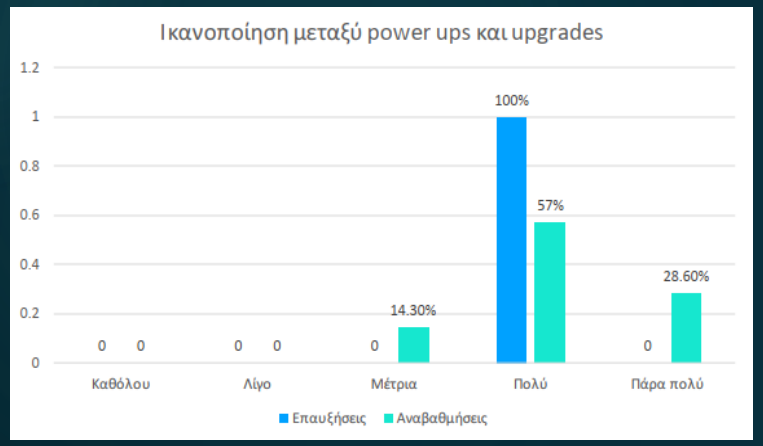
Αποτελέσματα

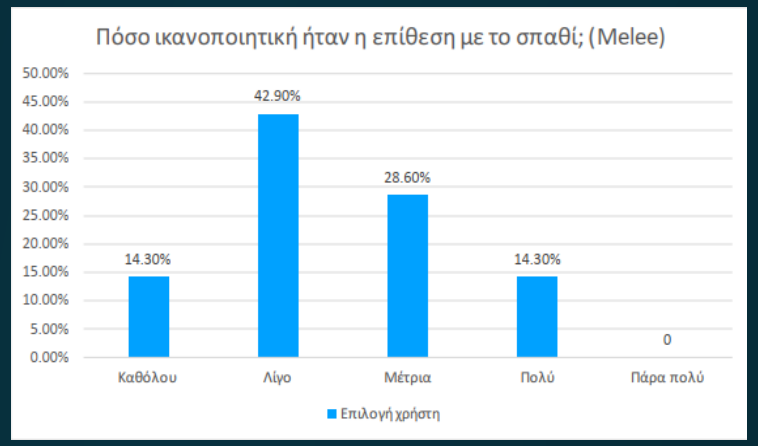
Αποτελέσματα - Σχεδιασμός παιχνιδιού

Επιλέχτηκαν 8 δοκιμαστές ηλικίας 18-34 χρονών κατά την περίοδο Σεπτεμβρίου - Οκτωβρίου 2023





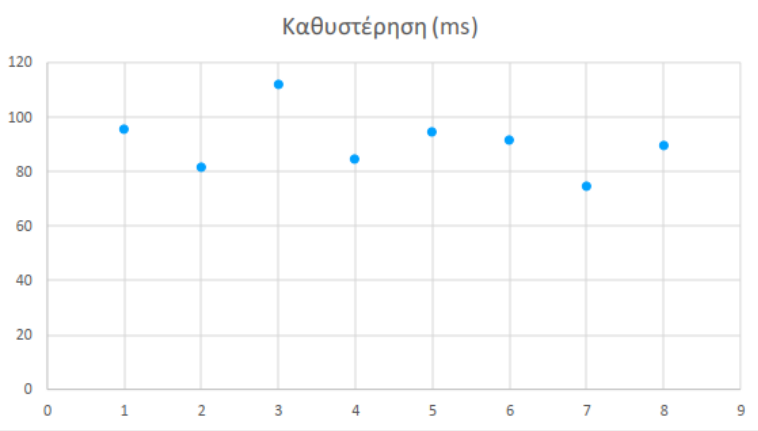


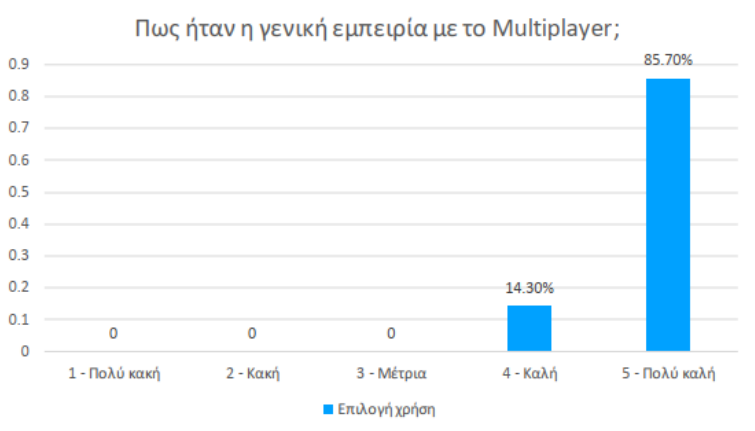


Οι δοκιμαστές δώσανε συναίνεση για τη συμμετοχή τους στην αξιολόγηση και συμφώνησαν με τους κανονισμούς που δίνονται από την Επιτροπή Ηθικής Δεοντολογίας της Έρευνας (Ε.Η.Δ.Ε) του Πανεπιστήμιου Δυτικής Μακεδονίας

Αποτελέσματα - Διαδικτύωση

Μέση καθυστέρηση – 91ms





Συμπεράσματα

- Η ανάπτυξη των ψηφιακών παιχνιδιών μπορεί να γίνει περίπλοκη όσο μεγαλώνει η κλίμακα του παιχνιδιού.
- Η διαδικτύωση ενός ψηφιακού παιχνιδιού προσθέτει επιπλέον βήματα στην ανάπτυξη και εμφανίζονται σύνθετα προβλήματα που καθιστούν την αποσφαλμάτωση μια δύσκολη διαδικασία.
- Το δυσκολότερο κομμάτι του παιχνιδιού ήταν ο συγχρονισμός του AI του εχθρού.
- Η εμπειρία ανάπτυξης ενός ψηφιακού παιχνιδιού και συγκεκριμένα ενός multiplayer παιχνιδιού ήταν εξίσου ευχάριστη όσο και επίπονη.

Μελλοντικές επεκτάσεις

- Περισσότερο περιεχόμενο παιχνιδιού όπως είδη εχθρών, όπλα και αναβαθμίσεις.
- Οι αναβαθμίσεις να παρέχουν πιο ενεργές επιδράσεις στο παιχνίδι.
- Ενσωμάτωση ασφάλειας για την αποφυγή κακόβουλων χρηστών.

Σας ευχαριστώ!

Ακολουθεί επίδειξη