**Απτική αλληλεπίδραση με εικονικά αντικείμενα**

**ΟΜΑΔΑ** 37

Κισκήρα Φωτεινή-Άρτεμις - 1059448

Μάμαλη Σοφία - 1059466

Μαναγούδης Μιχάλης - 1059398

Ματζαράπης Ηλίας - 1063966

Σεραφείμ Άννα - 1064481

Τεπελένας-Αγγελακόπουλος Μάριος - 1059502

**Υπεύθυνος καθηγητής:** Μουστάκας Κωνσταντίνος, Αν. Καθ. ΗΜΤΥ

**Έτος:** 2017**-**2018

Στο πλαίσιο του μαθήματος «Εισαγωγή στην επιστήμη του Ηλεκτρολόγου Μηχανικού» μας ανατέθηκε η ομαδική εργασία με θέμα «Απτική αλληλεπίδραση με εικονικά αντικείμενα» με υπεύθυνο τον καθηγητή τον κύριο Κωνσταντίνο Μουστάκα. ‘Έτσι, απευθυνθήκαμε σε αυτόν με σκοπό την εις βάθος κατανόηση του θέματος. Αφού μας έστειλε αρκετό βοηθητικό υλικό(εκπαιδευτικά βίντεο εκμάθησης της Unity, μιας μηχανής ανάπτυξης βιντεοπαιχνιδιού), το αξιοποιήσαμε μελετώντας ο καθένας μόνος του και δημιουργώντας το παιχνίδι-παράδειγμα από τα αντίστοιχα βίντεο τα οποία μας βοήθησαν να εξοικειωθούμε με το περιβάλλον του Unity. Στη συνέχεια, αρχίσαμε τις ομαδικές συναντήσεις στη σχολή όπου πειραματιζόμασταν με τη συσκευή LeapMotion μέσω του Unity.

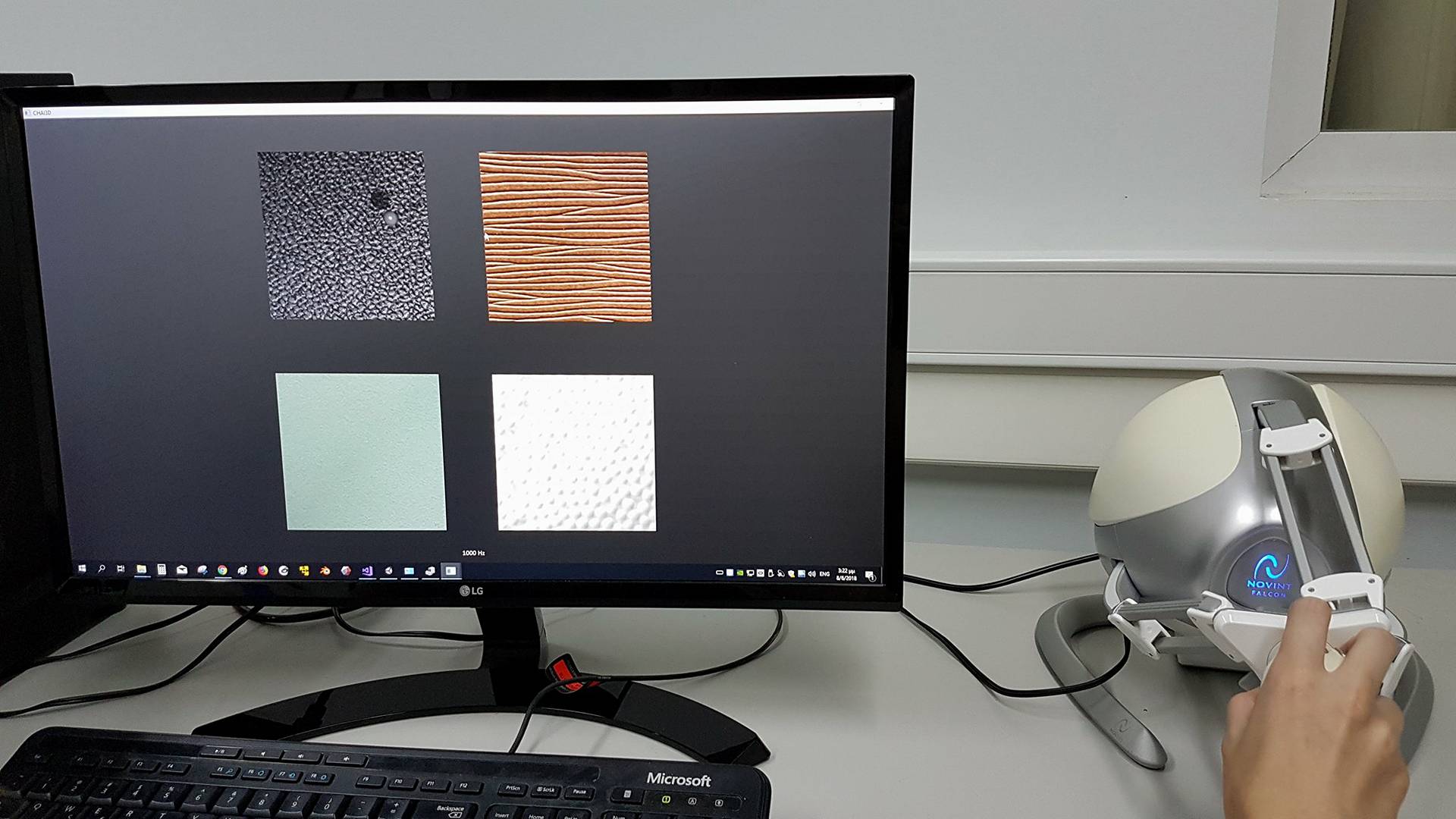
Πρώτο μέλημά μας ήταν να βρούμε πώς μπορούμε μέσω Unity να αναγνωρίζουμε κινήσεις. Για τη λύση αυτού του θέματος αναζητήσαμε στο internet πληροφορίες και tutorials. Έτσι, ανακαλύψαμε τη μέθοδο Extended Finger Detector που χρησιμοποιεί έτοιμο κώδικα(Scripts σε γλώσσα C#). Τα scripts υπάρχουν στα Unity Core Assets(έκδοση 4.3.4) στην ιστοσελίδα του LeapMotion (developer.leapmotion.com). Η μέθοδος αυτή σου επιτρέπει να επιλέξεις ποια δάχτυλα θέλεις να είναι σε προέκταση(extended) και ποια κλειστά (not extended). Όταν, εντοπίζεται η κάθε χειρονομία τότε εκτελείται ο αντίστοιχος κώδικας αναγνώρισής της, κατά τον οποίο εμφανίζεται το κατάλληλο μήνυμα. Στη συνέχεια, εισάγαμε κάποια 3D objects στο Unity τα οποία αντιστοιχίσαμε με κώδικα, στις κινήσεις που είχαμε δημιουργήσει, έτσι ώστε να διαγράφεται συγκεκριμένο αντικείμενο όταν αναγνωρίζεται η κίνηση.

Το επόμενο βήμα, ήταν να δημιουργήσουμε τη σειρά που αναγνωρίζονται οι κινήσεις, ώστε να δημιουργήσουμε γύρους στο παιχνίδι μας. Σκεφτήκαμε να υπάρχει μία public μεταβλητή που θα αυξάνει κάθε φόρα. Ταυτόχρονα αναπτύξαμε κώδικα για να κρατάει το score στο παιχνίδι μας.

Όλοι μας εργαστήκαμε ισάξια και παραβρεθήκαμε στις συγκεντρώσεις της ομάδας ώστε το αποτέλεσμα που πρόεκυψε να είναι «γέννημα» συλλογικής προσπάθειας. Συγκεκριμένα, όλοι ασχοληθήκαμε με τη συγγραφή κώδικα ωστόσο ο καθένας ανέλαβε κατά κύριο λόγο ένα συγκεκριμένο κομμάτι. Αναλυτικότερα, η Φωτεινή, η Σοφία και η Άννα ασχολήθηκαν με τη δημιουργία πρότυπων χειρονομιών, ο Μιχάλης με τον κώδικα που ελέγχει τη σωστή σειρά των χειρονομιών και καταστρέφει τα αντίστοιχα σχήματα, ο Μάριος με τα texts που εμφανίζονται για την ενημέρωση του χρήστη και για την εμφάνιση του score κάθε στιγμή και ο Ηλίας με τη διαχείριση του παιχνιδιού( πότε αρχίζει το παιχνίδι, πότε τελειώνει και πότε γίνεται επανεκκίνηση).

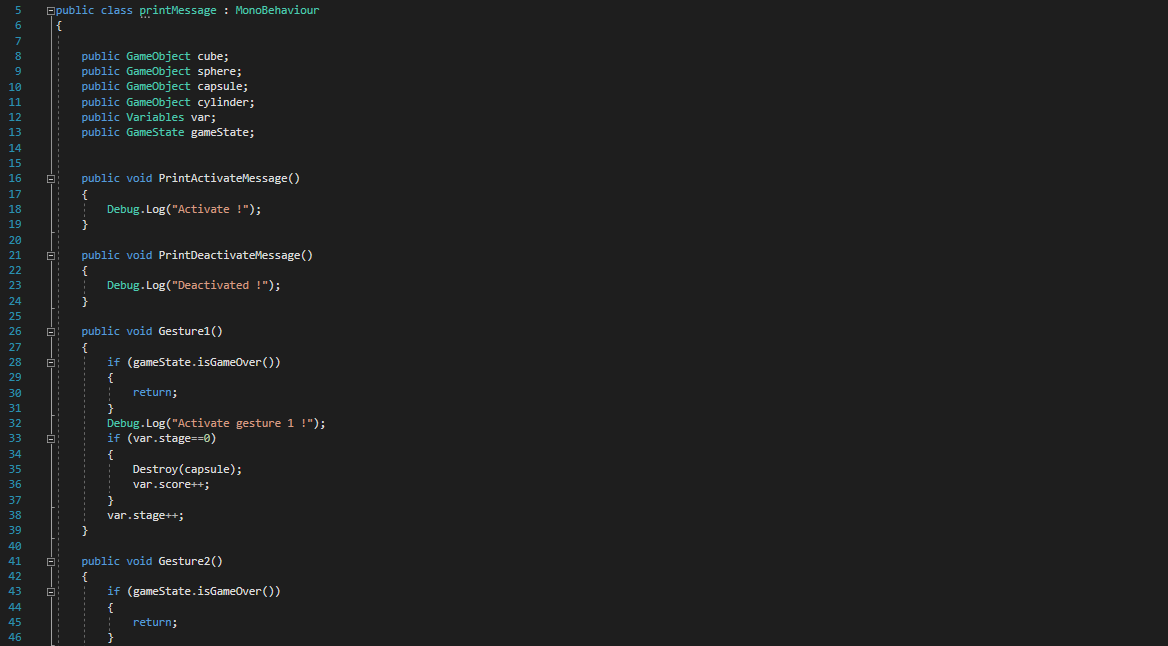
**ΟΔΗΓΙΕΣ ΕΓΚΑΤΑΣΤΗΣΗΣ**

Υπάρχουν στο αρχείο «ΟΔΗΓΟΣ ΕΓΚΑΤΑΣΤΑΣΗΣ.doc»

**ΠΑΡΑΔΕΙΓΜΑ ΧΡΗΣΗΣ ΤΟΥ UNITY**

**ΚΩΔΙΚΑΣ ΠΟΥ ΑΝΑΠΤΥΧΘΗΚΕ**

**A.** Script/class for the messages that are going to be printed in the console according to the movements.

****

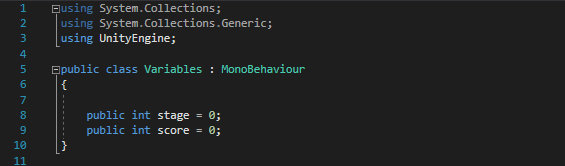
**1**. Debug.Log(); can be used instead of print(). In Unity, Debug.Log is preferred over print because it provides more functionality tailored to Unity's environment. It’s a best practice for better debugging (the process of identifying, analyzing, and fixing bugs or errors in software code) and development within Unity.

**2.** Gesture1() method:

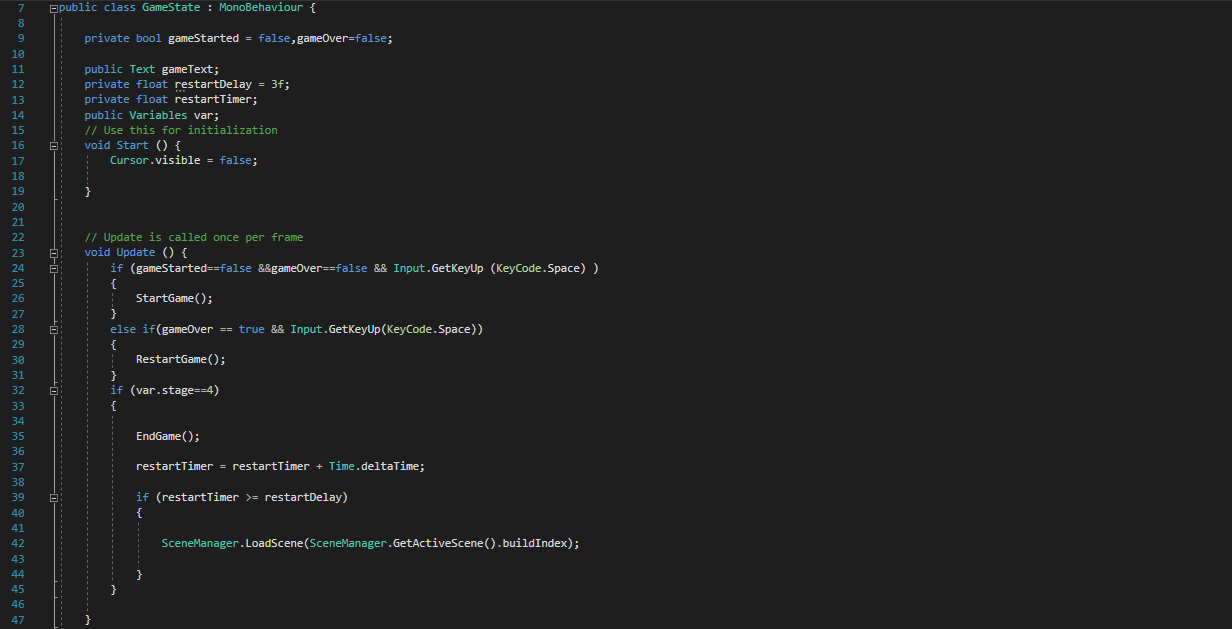
A. isGameOver() is typically a method or function, not a class. It would be used to trigger actions when a game ends.

B. var.stage: αν ισχύει ότι αυτό ίσο με 0, πάει στις μεταβλητές αυξάνει κατά 1 το score και καταστρέφει τη capsule

**B.** Script/class for the variables stage and score that are going to be increased as we pass rounds

****

**C.** Script/class for the state of the game. It is the main script/class of the game

****

**1.** Cursor.visible = false: makes the cursor invisible

**2.** Input.GetKeyUp(KeyCode.Space) checks if the Space key was released after being pressed. It's useful for triggering actions once a key is released

**3.** **SceneManager.GetActiveScene().buildIndex**: Gets the build index of the currently active scene.

**SceneManager.LoadScene()**: Reloads the scene using its build index.

It’s used to **restart the Current Scene**: This effectively resets the scene to its initial state.

**ΗΛΕΚΤΡΟΝΙΚΕΣ ΠΗΓΕΣ**

1. <https://developer.leapmotion.com/releases/detection-examples-101>
2. <https://developer.leapmotion.com/releases/detection-examples-104>
3. <https://docs.unity3d.com/Manual/ScriptingConcepts.html>
4. <https://docs.unity3d.com/ScriptReference/Debug.Log.html>
5. <https://answers.unity.com/questions/602224/destroy-gameobject-variable.html>
6. <https://answers.unity.com/questions/250286/how-do-you-delete-variables-or-arrays.html>
7. https://answers.unity.com/questions/10108/debuglog-or-print-whats-the-difference-and-when-to.html
8. [www.youtube.com/channel/UCBrjdbKlf9f\_3zlpNoTyZ0Q](http://www.youtube.com/channel/UCBrjdbKlf9f_3zlpNoTyZ0Q)
9. <https://www.youtube.com/watch?v=MBbWcniM0BM>
10. https://en.wikipedia.org/wiki/Haptic\_technology