Ελληνικό Μεσογειακό Πανεπιστήμιο
Σχολή Μηχανικών
Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Πρόγραμμα σπουδών: Μηχανικών Πληροφορικής Τ.Ε.

Πτυχιακή Εργασία

Κατασκευή εκπαιδευτικού αυτοματισμού με δυνατότητα προγραμματισμού μέσω γραφικής διεπαφής.

Γαροφαλάκη Ιφιγένεια-Ελευθερία

ΗΡΑΚΛΕΙΟ 2020

ΠΤΥΧΙΑΚΉ ΕΡΓΑΣΙΑ

Κατασκευή εκπαιδευτικού αυτοματισμού με δυνατότητα προγραμματισμού μέσω γραφικής διεπαφής.

Γαροφαλάκη Ιφιγένεια-Ελευθερία
ΑΜ: ΤΠ3973

Επιβλέποντες: Παναγιωτάκης Σπυρίδων, Αναπληρωτής Καθηγητής

ΠΕΡΙΛΗΨΗ

Ο σκοπός της παρούσας Πτυχιακής Εργασίας είναι να δημιουργήσουμε μια εκπαιδευτική εφαρμογή για μαθητές δημοτικού όπου μέσω αυτής να μπορούν να κατανοήσουν κάποιες βασικές έννοιες του προγραμματισμού. Με το νέο σύστημα εκπαίδευσης STEM, το έργο μας διευκολύνεται διότι ήδη υπάρχει, σε κάποια σχολεία στην Ελλάδα, μια πρώτη επαφή με την εκπαιδευτική ρομποτική και την επιστήμη των υπολογιστών. Έτσι με δεδομένο αυτό και με την βοήθεια των γλωσσών οπτικού προγραμματισμού, θα κατασκευάσουμε ένα εκπαιδευτικό αυτόνομο όχημα τύπου ρομπότ όπου θα μπορεί να προγραμματίζεται μέσω γραφικής διεπαφής(Scratch, Blockly) ώστε να είναι φιλικό στον προγραμματισμό από μαθητές δημοτικού. Πιο συγκεκριμένα θα φτιάξουμε ένα DIY όχημα όπου θα κινείται μέσα σε μια αυτοσχέδια πίστα και η λειτουργία του οχήματος θα ελέγχεται μέσω της γραφικής μας πλατφόρμας. Η επικοινωνία του οχήματος με την γραφική μας πλατφόρμα θα γίνεται ασύρματα με την χρήση κάποιων βιβλιοθηκών όπως η firmata, όπου βοηθάει στην επικοινωνία υλικού και λογισμικού, και με τα κατάλληλα εξαρτήματα για να μπορεί το όχημα να κινείται και να διαβάζει την απόσταση για το επόμενο εμπόδιο που βρίσκεται στον δρόμο του. Εν κατακλείδι, συνδυάζοντας όλα τα παραπάνω δηλαδή την λογική την εκπαίδευσης STEM, τις γλώσσες οπτικού προγραμματισμού και την σύνδεση των σωστών εξαρτημάτων έχουμε ένα επιθυμητό και εύκολο στην κατανόηση αποτέλεσμα. Επίσης θα αναπτυχθεί μια σειρά εστιασμένων παραδειγμάτων με σκοπό την εκπαίδευση των μαθητών σε τεχνικές προγραμματισμού.

ΛΕΞΕΙΣ – ΚΛΕΙΔΙΑ: Blockly, Wemos, Εκπαιδευτικό ρομπότ, firmata, Johnny-five, STEM.

ABSTRACT

The purpose of this Thesis is to create an educational application for primary school students where through it they can understand some basic concepts of programming. With the new STEM education system, our work is facilitated because there is already a first contact with educational robotics and computer science in some schools in Greece. So, considering this and with the help of visual programming languages, we will build a robot-type training vehicle where it can be programmed via a graphical interface (Scratch, Blockly) to be programmable-friendly by elementary school students. More specifically, we will build a DIY vehicle where it will move through a makeshift track and the operation of the vehicle will be managed through our graphical platform. The communication of the vehicle with our graphical platform will be done wirelessly using some libraries such as firmata, where it helps in the communication of hardware and software, and with the appropriate components so that the vehicle can move and read the distance to the next obstacle on its way. In conclusion, by combining all the above i.e. the logic of STEM education, the languages of visual programming and the connection of the right components, we have a desired and easy to understand effect. A series of focused examples will also be developed to educate students on programming techniques.

KEY – WORDS: Blockly, Wemos, Educational robot, firmata, Johnny-five, STEM.

ΕΥΧΑΡΙΣΤΙΕΣ

Για τη διεκπεραίωση της παρούσας Πτυχιακής Εργασίας θα ήθελα να ευχαριστήσω αρχικά τον επιβλέποντα καθηγητή μου, κ. Σπύρο Παναγιωτάκη για την ευκαιρία που μου έδωσε να εκπονήσω την εργασία αυτή και τη βοήθεια του στην επιλογή ενός θέματος που μου ταίριαζε και με έμαθε πολλά πράματα. Επίσης, θα ήθελα να ευχαριστήσω τους φίλους και συναδέλφους μου όπου με βοήθησαν και με υποστήριξαν καθ' όλη αυτήν την περίοδο. Τέλος θα ήθελα να ευχαριστήσω ιδιαίτερα τον φίλο μου Βιντσέντσο όπου με βοήθησε αρκετά πάνω στο κομμάτι της πτυχιακής μου και με βοήθησε σε θέματα όπου δεν ήμουν τόσο έμπειρη.

ΠΡΌΛΟΓΟΣ

Η παρούσα Πτυχιακή Εργασία εκπονήθηκε στο Ηράκλειο Κρήτης από το 2018-2019. Αποτελεί αναπόσπαστο κομμάτι για την απόκτηση του πτυχίου.

Πίνακας περιεχομένων

1. ΕΙΣΑΓΩΓΗ

Η τεχνολογία και η πληροφορική είναι ένα αναπόσπαστο κομμάτι της ζωής μας στην σήμερον ημέρα. Παντού υπάρχει κάτι το οποίο έχει δημιουργηθεί για να μας βοηθάει και να μας διευκολύνει την καθημερινότητα. Ένα παράδειγμα είναι οι αυτοματισμοί , οι έξυπνες συσκευές όπου χρησιμοποιούν τον προγραμματισμό για να λειτουργήσουν. Το Internet of Things (IoT) μπήκε πολύ γρήγορα στην καθημερινότητα μας. Η τεχνολογία εξελίσσεται συνεχώς αλλά τι υπάρχει πίσω από αυτήν; Τι υπάρχει πίσω από τις εφαρμογές και πίσω από τα ρομπότ; Υπάρχει κώδικας! Και εφόσον χρησιμοποιούμε την τεχνολογία τόσο συχνά τι καλύτερο από να την μαθαίνουμε και να την κατανοούμε από μικρή ηλικία. Για αυτό τον λόγο υπάρχει η εκπαιδευτική ρομποτική, ένα σχέδιο διεπιστημονικής δραστηριότητας όπου αφορά τα πεδία της τεχνολογίας, της πληροφορικής, της επιστήμης και τα μαθηματικά. Υπάρχει ένα εκπαιδευτικό πλαίσιο όπου βασίζεται στην εκπαιδευτική ρομποτική και έχει εφαρμοστεί παγκοσμίως στα σχολεία, όπου προσπαθεί να διεγείρει το ενδιαφέρον των μαθητών προς αυτές τις επιστήμες. Το εκπαιδευτικό πλαίσιο ονομάζεται STEM (Science,Technology,Engineering and Mathematics). Έτσι και εμείς στα πλαίσια της πτυχιακής εργασίας προσπαθήσαμε να εφαρμόσουμε αυτήν την μέθοδο ώστε να διευκολύνουμε μαθητές μικρής ηλικίας να κατανοήσουν κάποιες βασικές μεθόδους προγραμματισμού μέσω γραφικής διεπαφής. Παρακάτω θα δούμε πως λειτουργεί το εκπαιδευτικό πρόγραμμα STEM συνδυάζοντας το με το IoT και πως μπορούμε να πετύχουμε ένα ευχάριστο και εύχρηστο αποτέλεσμα το οποίο μπορεί να χρησιμοποιηθεί ως εκπαιδευτικό εργαλείο σε μαθητές δημοτικού σχολείου.

Πιο συγκεκριμένα θα δημιουργήσουμε ένα DIY όχημα όπου θα δέχεται εντολές μέσω μιας προσιτής εφαρμογής που έχει δημιουργηθεί με βάση το google blockly. Μέσω αυτής της εφαρμογής που έχουμε δημιουργήσει, ο μικρός μας χρήστης θα δημιουργεί μια επανάληψη εντολών, όπου θα μπορεί να συμβουλευτεί την σειρά μαθημάτων ή τον δάσκαλο του, ώστε να κατευθύνει το όχημα από την έναρξη στον προορισμό του σε μια αυτοσχέδια πίστα. Η εφαρμογή είναι προγραμματισμένη να στέλνει το τελικό αποτέλεσμα όπου έχουμε κάνει πατώντας το κουμπί “εκτέλεση”, στον μικροελεκτή που έχουμε πάνω στο όχημα. Έπειτα ο μικροελεκτής σε

real time παίρνει τα δεδομένα από το αισθητήρα απόστασης και ταυτόχρονα εκτελεί τις εντολές μας. Παραπάνω πληροφορίες σχετικά με το κύκλωμα μας και τον προγραμματισμό του, θα δούμε στα παρακάτω κεφάλαια.

2. ΕΚΠΑΙΔΕΥΤΙΚΗ ΡΟΜΠΟΤΙΚΗ

2.1 Τι είναι η εκπαιδευτική ρομποτική;

Η εκπαιδευτική ρομποτική θεωρείται ως σχέδιο διεπιστημονικής δραστηριότητας κυρίως στην επιστήμη, τα μαθηματικά, την πληροφορική και την τεχνολογία προσφέροντας σημαντικά νέα οφέλη στην εκπαίδευση και γενικότερα σε όλα τα επίπεδα. Η εκπαιδευτική ρομποτική είναι μια ισχυρή και ευέλικτη διδασκαλία μάθησης που ενθαρρύνει τους μαθητές να κατασκευάσουν και να ελέγξουν τα ρομπότ χρησιμοποιώντας συγκεκριμένες γλώσσες προγραμματισμού. Η εκπαιδευτική ρομποτική προάγει έναν ευχάριστο τρόπο μάθησης ενώ παράλληλα προωθεί τα κίνητρα των παιδιών στην συνεργασία, την αυτοπεποίθηση και τη δημιουργικότητα. Πολλοί ερευνητές υποστηρίζουν ότι τα προγράμματα ρομποτικής παρέχουν μια πολύτιμη διαδρομή για την συμμετοχή τους στην επιστήμη, την τεχνολογία, την μηχανική και τα μαθηματικά (STEM).

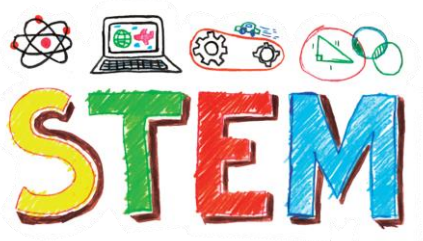
Η εκπαιδευτική ρομποτική αποτελεί ένα σχετικά νέο επιστημονικό κλάδο ο οποίος ασχολείται με την κατασκευή, τον προγραμματισμό και την αξιοποίηση των ρομπότ σε εκπαιδευτικό επίπεδο. Εμπεριέχεται ως διδακτικό αντικείμενο στο μάθημα της πληροφορικής και αποσκοπεί στο σχεδιασμό δραστηριοτήτων για την ενίσχυση δεξιοτήτων υπολογιστικής σκέψης. Τα τελευταία χρόνια υπάρχει μεγάλη απήχηση της εκπαιδευτικής ρομποτικής από τους εκπαιδευτικούς πρωτοβάθμιας και δευτεροβάθμιας εκπαίδευσης, εξαιτίας της αποτελεσματικότητας με την οποία βοηθά τους μαθητές στην απόκτηση δεξιοτήτων επίλυσης προβλημάτων (problem solving skills).

<u>Ανάπτυξη δεξιοτήτων-γνώσεων μέσω της εκπαιδευτικής ρομποτικής :</u>

Η εκπαιδευτική ρομποτική προσφέρεται για την καλλιέργεια πνεύματος ομαδικής συνεργασίας στους μαθητές ενώ τους ενθαρρύνει να πειραματιστούν και να εφαρμόσουν αυτά που έχουν διδαχθεί από τα γνωστικά πεδία των μαθηματικών, της φυσικής, της τεχνολογίας και της πληροφορικής. Επιπλέον έρχονται σε επαφή με τις βασικές έννοιες του προγραμματισμού και ανακαλύπτουν τις υποσυνείδητες χρήσεις των οργάνων του ανθρώπινου σώματος και πως αυτές εφαρμόζονται στη τεχνολογία (π.χ. χρώμα , κίνηση με εμπόδια κτλ).

2.2 STEM

Το ακρώνυμο STEM σημαίνει Science, Technology, Engineering και Mathematics. Το STEM είναι ένα εκπαιδευτικό πλαίσιο που χρησιμοποιείται παγκοσμίως. Αυτό που ξεχωρίζει το STEM από την παραδοσιακή εκπαιδευτική επιστήμη και την μαθηματική παιδεία είναι το περιβάλλον μικτής μάθησης όπου δείχνει στους μαθητές πως η επιστημονική μέθοδος μπορεί να εφαρμοστεί στην καθημερινή ζωή. Σε έναν κόσμο συνεχώς μεταβαλλόμενο, ολοένα και πιο πολύπλοκο, είναι πιο σημαντικό από ποτέ η νεολαία να είναι διατεθειμένη να φέρει γνώσεις και δεξιότητες για την επίλυση προβλημάτων, την αίσθηση της πληροφόρησης και την κατανόηση του τρόπου συλλογής και αξιολόγησης αποδεικτικών στοιχείων για την λήψη αποφάσεων. Αυτά είναι τα είδη δεξιοτήτων που αναπτύσσουν οι σπουδαστές στην επιστήμη, την τεχνολογία, τη μηχανική και τα μαθηματικά όπως επίσης τους διδάσκει στην υπολογιστική σκέψη και επικεντρώνεται στις πραγματικές εφαρμογές της επίλυσης προβλημάτων.



Εικόνα 1: Logo STEM

Η εκπαίδευση STEM στο νηπιαγωγείο:

Η εκπαίδευση STEM στο νηπιαγωγείο έχει ως βασικό στόχο να καλλιεργήσει βασικές δεξιότητες προγραμματισμού μέσα από δραστηριότητες ευχάριστες, με παιγνιώδη χαρακτήρα. Επίσης η εκπαίδευση STEM στην προσχολική ηλικία στοχεύει στην ανάπτυξη των δεξιοτήτων επίλυσης προβλημάτων και συνεργασία σε ομάδες. Η STEM Education προσφέρει ένα πρόγραμμα γνωριμίας με το Kids First Coding & Robotics, μια νέα πλατφόρμα εκπαίδευσης STEM, ιδανική για να κατανοήσουν τα παιδιά από την προσχολική ηλικία τη βασική λογική του

προγραμματισμού. Παραπάνω πληροφορίες για το συγκεκριμένο πρόγραμμα εκπαίδευσης θα βρείτε ανατρέχοντας στην βιβλιογραφία με αριθμό [14].

Η εκπαίδευση STEM στο δημοτικό σχολείο :

Η εκπαίδευση STEM επικεντρώνεται στα εισαγωγικά μαθήματα STEM καθώς και στην ευαισθητοποίηση των πεδίων STEM. Ο στόχος είναι να ωθήσει το ενδιαφέρον των μαθητών σε αυτούς που θέλουν να ακολουθήσουν τα μαθήματα επειδή τους αρέσουν, και όχι επειδή πρέπει. Με την χρήση ηλεκτρονικών υπολογιστών, κινητό ή tablet οι μαθητές μέσω εκπαιδευτικής εφαρμογής πχ όπως την δική μας εφαρμογή ή το Blockly ή το Scratch, δηλαδή μέσω μιας γραφικής διεπαφής, θα μπορούν να κατανοήσουν τις βασικές γνώσεις προγραμματισμού.

Η εκπαίδευση STEM στο γυμνάσιο και λύκειο:

Σε αυτό το στάδιο τα μαθήματα γίνονται πιο αυστηρά και απαιτητικά. Οι μαθητές εξερευνούν την STEM και των σχετικών σταδιοδρομιών. Με τον σωστό εξοπλισμό οι μαθητές ξεκινάνε να επιλύουν προβλήματα κατασκευάζοντας λειτουργικά αυτόνομα ρομπότ και συσκευές ελέγχου, με την βοήθεια των εκπαιδευτικών τους. Δεν είναι απαραίτητη η γνώση προγραμματισμού γιατί σε αυτόν το τομέα βοηθάει η Scratch. Πιο συγκεκριμένα στην Scratch δημιουργείς την λογική που θες να χρησιμοποιήσεις και στην μεταφράζει σε γλώσσα προγραμματισμού. Στο λύκειο όμως οι μαθητές έχουν αρχίσει να διδάσκονται τις γλώσσες προγραμματισμού οπότε μπορούν να χρησιμοποιήσουν τις γνώσεις του και σε συνδυασμό με τις άλλες τρεις επιστήμες να κάνουν πιο περίπλοκα πρότζεκτ.

Γιατί να χρησιμοποιήσουμε την μεθοδολογία STEM ;

Τα παιδιά μαθαίνουν να δουλεύουν σε ομάδες από πολύ μικρή ηλικία. Επικοινωνούν, επιχειρηματολογούν, διαφωνούν και συνεργάζονται. Επίσης μαθαίνουν να επιλύουν προβλήματα γιατί όπως είδαμε και παραπάνω ένα από τα βασικά στοιχεία της STEM είναι το problem solving. Τα παιδιά αναζητούν λύσεις σε καθημερινά προβλήματα. Επιπλέον τα παιδιά κάνουν μια εισαγωγή στις φυσικές επιστήμες, στην μηχανική και τα μαθηματικά. Με το STEM η εισαγωγή στις φυσικές επιστήμες γίνεται με βιωματικά projects όπου τα παιδιά συμμετέχουν ενεργά. Τα

παιδιά κατανοούν τις απλές και σύνθετες μηχανές και την λειτουργία τους. Αντιλαμβάνονται έτσι τη χρησιμότητα των καθημερινών μηχανών. Επίσης μαθαίνουν μέσα από δραστηριότητες να σκέφτονται αλγοριθμικά. Τα μαθηματικά γίνονται πιο ελκυστικά, αφού είναι μέρος της βιωματικής μάθησης.

Εν κατακλείδι, τα παιδιά μέσω της εκπαιδευτικής ρομποτικής μαθαίνουν να κατασκευάζουν και να προγραμματίζουν, μέσω γλώσσα οπτικού προγραμματισμού, ρομπότ, μένοντας συνεχώς ενημερωμένα για τις τρέχουσες τεχνολογικές εξελίξεις. Αυτή η εκπαιδευτική μέθοδος έχει ήδη να χρησιμοποιείται στα ελληνικά σχολεία.

2.2.1 Η χρήση της STEM στην δική μας εφαρμογής

Όπως προαναφέραμε παραπάνω σκοπός της παρούσας εργασίας είναι η δημιουργία ενός εκπαιδευτικού αυτοματισμού όπου θα βοηθάει μαθητές δημοτικού στην κατανόηση βασικών ορολογιών του προγραμματισμού. Κάτι τέτοιο με την χρήση της STEM γίνετε εφικτό. Από την στιγμή που η STEM λειτουργεί ως δραστηριότητα στα ελληνικά σχολεία και είναι ευρέως γνωστή ως μεθοδολογία το εκμεταλλευτήκαμε έτσι ώστε να δημιουργήσουμε κάτι έξυπνο και εύχρηστο για τους μικρούς μαθητές. Η δική μας εφαρμογή έχει σχεδιαστεί και προγραμματιστεί έτσι ώστε οι μαθητές χρησιμοποιώντας το blockly, μια γλώσσα οπτικού προγραμματισμού, έτσι με αυτόν τρόπο και μέσα από τα ενδεικτικά εκπαιδευτικά μαθήματα. Η εφαρμογή μας περιέχει όλα τα πεδία της STEM. Περιέχει την μηχανική μάθηση γιατί ο μαθητής θα πρέπει να αποφασίσει την διαδρομή όπου το αυτόνομο αυτοκίνητό μας θα πρέπει να επιλέξει για να φτάσει στον προορισμού του, ανακαλύπτουν τις υποσυνείδητες χρήσεις των οργάνων του ανθρώπινου σώματος και πως αυτές εφαρμόζονται στη τεχνολογία παραδείγματος χάρη η κίνηση με εμπόδια. Χρησιμοποιούν τα μαθηματικά για να μπορούν να υπολογίσουν πόσα βήματα μπορεί να προχωρήσει το αυτόνομο αυτοκίνητο ή πόσες μοίρες πρέπει να στρίψει για να αποφύγει το εμπόδιο που βρίσκεται μπροστά του. Και τέλος χρησιμοποιεί την μηχανική έτσι ώστε να μάθει με απλά παραδείγματα και ορισμούς τι είναι ένας μκροελεκτής ή ένας αισθητήρας. Παρακάτω θα δούμε παραπάνω λεπτομέρειες σχετικά με το παρών πρότζεκτ.

2.3 Γλώσσες οπτικού προγραμματισμού

Στην επιστήμη υπολογιστών, οπτική γλώσσα προγραμματισμού είναι μια γλώσσα προγραμματισμού που επιτρέπει στο χρήστη την δημιουργία προγραμμάτων μέσα από το γραφικό χειρισμό προγραμματιστικών στοιχείων αντί κειμένου.

Οι οπτικές γλώσσες προγραμματισμού, ανάλογα με τον τύπο και την έκταση της χρήσης των οπτικών εκφράσεων σε γλώσσες βασισμένες στα εικονίδια (icon-based), γλώσσες βασισμένες στις φόρμες (form-based) και σε γλώσσες βασισμένες σε διαγράμματα (diagram languages). Περιβάλλοντα οπτικού προγραμματισμού παρέχουν στοιχεία εικονιδίων ή γραφικών που μπορούν να χρησιμοποιηθούν διαλογικά από το χρήστη σύμφωνα με κάποια χωρική γραμματική (spatial grammar), για την κατασκευή ενός προγράμματος.

2.3.1 Γραφική διεπαφή Blockly

Το blockly είναι βιβλιοθήκη όπου προσθέτει επεξεργαστή οπτικού κώδικα σε εφαρμογές ιστού και εφαρμογές κινητού. Το blockly editor χρησιμοποιεί αλληλεπικαλυπτόμενα γραφικά block για να αναπαραστήσει έννοιες κώδικα όπως μεταβλητές, λογικές εκφράσεις, βρόχους και πολλά άλλα. Επιτρέπει στους χρήστες να εφαρμόζουν αρχές προγραμματισμού χωρίς να χρειάζεται να ανησυχούν για τη σύνταξη τους.

Από την μεριά του χρήστη, το blockly είναι ένας διαισθητικός, οπτικός τρόπος για την κατασκευή κώδικα. Από την μεριά του προγραμματιστή, το blockly είναι ένα έτοιμο user interface για τη δημιουργία μιας οπτικής γλώσσας που εκπέμπει συντακτικά σωστό κώδικα που δημιουργείται από τον χρήστη. Στην δική μας περίπτωση θα χρησιμοποιήσουμε το blockly από την μεριά του προγραμματιστή, δηλαδή θα τροποποιήσουμε τα ήδη υπάρχοντα blocks να λειτουργούν με βάση τις δικές μας ανάγκες έτσι ώστε ο χρήστης να βλέπει μόνο τα blocks και όχι τον κώδικα. Το blockly όμως μπορεί να εξάγει block σε πολλές γλώσσες προγραμματισμού το οποίο είναι χρήσιμο για χρήστες οι οποίοι δεν είναι εξοικειωμένοι με τον προγραμματισμό. Μερικές από τις γλώσσες τις οποίες μπορούν να παράγουν μέσω των blocks είναι οι παρακάτω:

- JavaScript
- Python
- PHP
- Lua

- Dart

Ως μηχανή ενσωματώνεται σε πολλές δημοφιλείς εφαρμογές όπως το Hour Of Code, αλλά μπορεί να χρησιμοποιηθεί και αυτόνομα. Μπορεί να χρησιμοποιηθεί ως εργαλείο γρήγορης ανάπτυξης εφαρμογών από χρήστες που είναι εξοικειωμένοι με τις έννοιες του προγραμματισμού αλλά όχι με συγκεκριμένες γλώσσες. Για παράδειγμα μπορούμε να αναπτύξουμε λογισμικό server σε Python ή PHP στα πλαίσια ενός σχολικού project και να χρησιμοποιήσουμε το Blockly για την ανάπτυξη του κώδικα ή για εκσφαλμάτωση (debugging). Σύντομα, το Blockly θα συνεργάζεται με το Scratch στο project που ονομάζεται Scratch Blocks και θα ελέγχει μέσω Bluetooth αυτοματισμούς και μηχανισμούς του πακέτου (κιτ) ρομποτικής Lego Wedo 2.0. Και ενώ αναμένουμε την τελική έκδοση αυτής της συνεργασίας της Google με το MIT μπορούμε να εντάξουμε, με τη βοήθεια του Blockly, την χρήση υψηλού επιπέδου γλωσσών προγραμματισμού, στην εκπαίδευση.

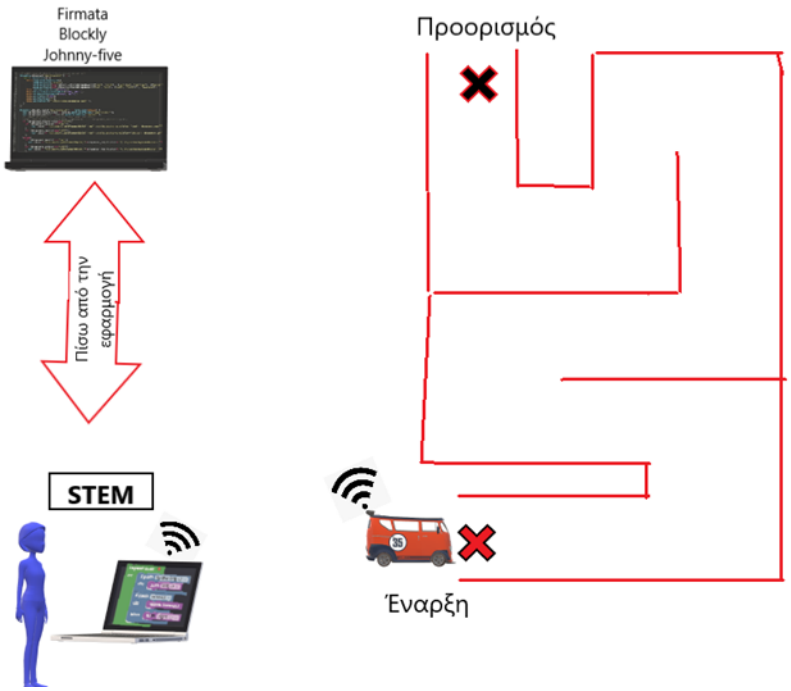
.

3. ΣΧΕΔΙΑΣΜΟΣ ΕΚΠΑΙΔΕΥΤΙΚΟΥ ΑΥΤΟΜΑΤΙΣΜΟΥ

Η ιδέα του παρών πρότζεκτ είναι να παρέχουμε στους μαθητές δημοτικού την δυνατότητα μάθησης βασικών εννοιών προγραμματισμού με έναν εύκολο τρόπο. Έτσι με αυτόν τρόπο κάνουμε και μια μικρή εισαγωγή, στους μαθητές, στο εκπαιδευτικό πλαίσιο STEM, όπου έχουν ήδη αναφέρει πως λειτουργεί παραπάνω. Για να πετύχουμε τον σκοπό μας φτιάξαμε ένα αυτοσχέδιο αυτόνομο αυτοκινητάκι που δέχεται εντολές από τον χρήστη. Στην παρούσα εφαρμογή που δημιουργήσαμε τα παιδιά χειρίζονται τις κινήσεις του αυτόνομου αυτοκινήτου μέσω της γραφικής διεπαφής blockly. Μέσω των blocks που έχουμε ήδη προγραμματίσει με βάση το blockly, ο μαθητής μπορεί να εκτελέσει τις παρακάτω εντολές: Προχώρα μπροστά, Στρίψε αριστερά/δεξιά x°, Σταμάτα, Για ν φορές επανέλαβε, Εάν εμπόδιο μπροστά τότε, Εάν εμπόδιο μπροστά τότε..αλλιώς κάνε και Επανέλαβε μέχρι να βρεις εμπόδιο. Στόχος μας είναι με αυτές τις εντολές σε συνδυασμό σωστά παραδείγματα, οι μαθητές να μπορούν να κατανοήσουν τις βασικές εντολές επανάληψης παίζοντας.

Έτσι λοιπόν για την κατασκευή του αυτόνομου αυτοκινήτου, όπου πληροί τις προϋποθέσεις για να εκτελέσει τις παραπάνω εντολές, χρειαζόμαστε τον σωστό εξοπλισμό. Μετά από την σωστή κατανόηση της εργασίας μας, προβήκαμε σε μια έρευνα για την αγορά του σωστού εξοπλισμού και καταλήξαμε στο ότι χρειαζόμαστε την κατασκευή ενός αυτοκινήτου, μόνο τον σκελετό του αυτοκινήτου, δύο ρόδες και δύο κινητήρες. Επίσης χρειαζόμαστε έναν μικροελεκτή όπου θα μπορεί να λαμβάνει τα δεδομένα από την εφαρμογή μας και να τα επεξεργάζεται έτσι ώστε να μπορεί να εκτελέσει το πρόγραμμα σωστά, έναν αισθητήρα απόστασης όπου θα διαβάζει την απόσταση που υπάρχει μπροστά από το αυτοκίνητο και στην συνέχεια σε συνδυασμό με τον μικροελέκτη θα γίνεται η σωστή επιλογή μονοπατιού παραδείγματος χάρη όταν κάποιος επιλέγει να χρησιμοποιήσει την εντολή “Εάν εμπόδιο μπροστά τότε Στρίψε δεξιά Αλλιώς Προχώρα μπροστά” ο μικροελεκτής θα είναι σε θέση να κρίνει τι πρέπει να κάνει. Και τέλος τελευταίο εξάρτημα που χρησιμοποιήσαμε είναι μια γέφυρα H-bridge όπου επιτρέπει στου κινητήρες μας να κινούνται. Παρακάτω θα δούμε παραπάνω πληροφορίες για τα εξαρτήματα μας και πως τα συνδυάσαμε όλα αυτά μαζί έτσι ώστε να υπάρχει μια λειτουργικότητα. Η ασύρματη τεχνολογία όπου θα χρησιμοποιήσουμε θα είναι το Wi-Fi.

Το αυτοκίνητο μας θα τρέχει πάνω σε μια αυτοσχέδια πίστα. Δεν έχουμε κατασκευάσει κάτι συγκεκριμένο αλλά δεν είναι απαραίτητο να υπάρχει κάτι συγκεκριμένο, δηλαδή θα μπορούσε να οτιδήποτε παραδείγματος χάρη ένας διάδρομος από θρανία ή τσάντες όπου χωράει να κινηθεί το αυτοκίνητο μας και φυσικά να περιέχει εμπόδια. Το ιδανικό δάπεδο θα ήταν ένα επίπεδο δάπεδο χωρίς κάποιο περιορισμό στο υλικό του δαπέδου.



Εικόνα 2: Η γενική ιδέα του project μας

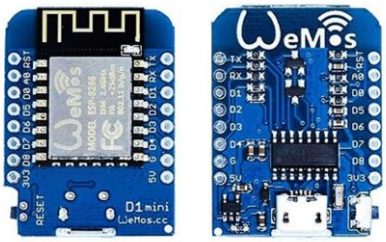
4. HARDWARE ΤΟΥ ΕΚΠΑΙΔΕΥΤΙΚΟΥ ΡΟΜΠΟΤ

4.1 Μικροελεγκτής

4.1.1 Wemos D1 mini

Το Wemos D1 mini είναι μια μικρή wi-fi συσκευή βασισμένη στο ESP8266EX chip. Αυτή η συσκευή είναι μια πολύ συμπαγής λύση για τη δημιουργία πρωτότυπων μικρών έξυπνων αντικειμένων που συνδέονται στο διαδίκτυο, χάρη στις λειτουργίες του Wi-fi Espressif ESP8266. Το Wemos D1 mini διαθέτει 4MB flash memory, 80Hz system clock, περίπου 50k χρησιμοποιήσιμης μνήμης RAM και ασύρματο πομποδέκτη Wi-fi.

Features :



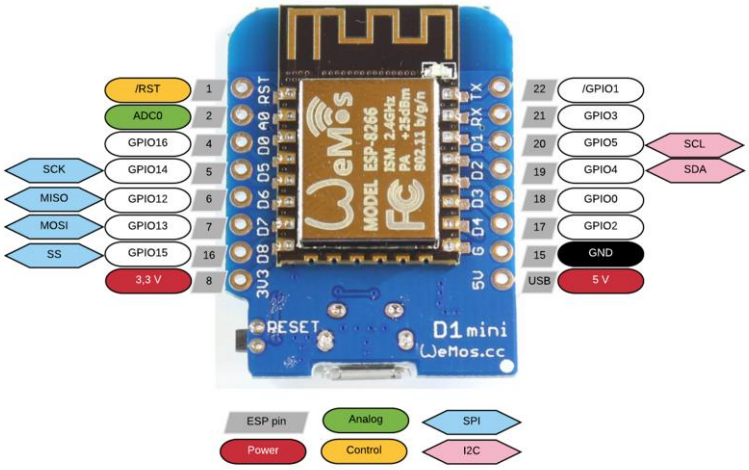
- 11 digital input/output pins, όλα τα pins έχουν interrupt, PWM, I2C, one-wire-supported(εκτός το D0)
- 1 analog input (3.2V max input)
- a Micro USB connection
- Συμβατό με MicroPython, Arduino, nodeMCU

Power:

Η ισχύς στο Wemos D1 mini παρέχεται μέσω USB Micro B connector ή κατευθείαν από το “Vin” pin. Η πηγή ενέργειας διαλέγεται αυτόματα. Η συσκευή μπορεί να λειτουργήσει σε εξωτερική τροφοδοσία από 6V έως 20V. Αν χρησιμοποιήσουμε περισσότερα από 12V, ο ρυθμιστής τάσης μπορεί να υπερθερμανθεί και να βλάψει τη συσκευή. Το συνιστώμενο εύρος τιμών είναι από 7v έως 12V.

Connect , Register , Virtualize , Program:

Το Wemos D1 mini εκθέτει τη σειριακή θύρα της μονάδας ESP8266 μέσω CH340 USB to UART chip.



Εικόνα 2: Wemos D1 mini pins

Επιλέξαμε το wemos D1 mini αρχικά γιατί είναι μια συσκευή βασισμένη στο wi-fi, κάτι που μας εξυπηρετεί για την δημιουργία του πρότζεκτ μας, όπως επίσης είναι συμβατό με την firmata. Παρακάτω θα δώσουμε παραπάνω πληροφορίες σχετικά με αυτό. Επίσης, άλλος ένα λόγος που επιλέξαμε το wemos είναι μικρό σε μέγεθος και ελαφρύ και μας παρέχει όσα pins χρειαζόμαστε για το παρών πρότζεκτ.

Οδηγός εγκατάστασης του wemos D1 mini με το λογισμικό Arduino IDE υπάρχει παρακάτω και περισσότερα τεχνικά χαρακτηριστικά σχετικά με το wemos D1 mini θα βρείτε στον σύνδεσμο που βρίσκεται στη βιβλιογραφία με αριθμό [1].

4.2 Αισθητήρες

4.2.1 Sharp GP2Y0A41SK0F

Ο αισθητήρας GP2Y0A41SK0F είναι ένας αισθητήρας απόστασης (distance measuring sensor), αποτελείται από έναν ολοκληρωμένο συνδυασμό PSD (position sensitive detector), IR-LED (infrared emitting diode) και κύκλωμα επεξεργασίας σήματος. Η ποικιλία της ανακλαστικότητας του αντικειμένου, η θερμοκρασία περιβάλλοντος και η διάρκεια λειτουργίας

δεν επηρεάζονται εύκολα από την ανίχνευση απόστασης λόγω της υιοθέτησης της μέθοδος τριγωνισμού (triangulation method). Αυτή η συσκευή εξάγει την τάση που αντιστοιχεί στην απόσταση ανίχνευσης. Έτσι αυτός ο αισθητήρας μπορεί να χρησιμοποιηθεί ως ένας αισθητήρας εγγύτητας (proximity sensor).

Κόκκινο καλώδιο: 5V
Μαύρο καλώδιο: ground
Κίτρινο καλώδιο: output

Εικόνα 3: Sharp GP2Y0A41SK0F

Χαρακτηριστικά:

1. Εύρος μέτρησης : 4cm to 30cm
2. Αναλογική έξοδος
3. Ο αισθητήρας μέτρησης απόστασης, είναι ενωμένος με το PSD ,υπέρυθρο LED και το κύκλωμα επεξεργασίας σήματος
4. Σύντομος κύκλος μέτρησης : 16.5ms
5. Αναλογίες του αισθητήρα: 29.5mm X 13.0mm X 13.5mm
6. Τάση λειτουργίας: 4.5V – 5.5V
7. Μέση κατανάλωση ρεύματος: 12 mA

Επιλέξαμε τον αισθητήρα της sharp γιατί είναι ένας infrared αισθητήρας και είναι συμβατός με την πλατφόρμα του Johnny-five, που όπως θα εξηγήσουμε παρακάτω μας δίνει την δυνατότητα να χρησιμοποιήσουμε την firmata. Περισσότερες λεπτομέρειές σχετικά με το johnny-five και την firmata υπάρχουν στα επόμενα κεφάλαια.
Περισσότερα τεχνικά χαρακτηριστικά σχετικά με τον αισθητήρα της sharp, GP2Y0A41SK0F θα βρείτε στον σύνδεσμο που βρίσκεται στη βιβλιογραφία με αριθμό[2].

4.3 Actuators

4.3.1 DC Motors

Ένας DC κινητήρας είναι ένας από τους τύπους περιστροφικών ηλεκτρικών μηχανών που μετατρέπει την ηλεκτρική ενέργεια συνεχούς ρεύματος σε μηχανική ενέργεια. Σχεδόν όλοι οι τύποι κινητήρων συνεχούς ρεύματος έχουν κάποιο εσωτερικό μηχανισμό, είτε ηλεκτρομαγνητικό είτε ηλεκτρονικό, για να αλλάζουν περιοδικά την κατεύθυνση της ροής σε μέρος του κινητήρα.

Οι DC κινητήρες ήταν η πρώτη μορφή κινητήρα που χρησιμοποιείται ευρέως, δεδομένου ότι θα μπορούσαν να τροφοδοτηθούν από υπάρχοντα συστήματα διανομής ηλεκτρικού συνεχούς ρεύματος. Η ταχύτητα του DC κινητήρα μπορεί να ελεγχθεί σε ένα ευρύ φάσμα χρησιμοποιώντας είτε μεταβλητή τάση τροφοδοσίας είτε μεταβάλλοντας την ισχύ του ρεύματος στις περιελίξεις του πεδίου. Μικροί DC κινητήρες χρησιμοποιούνται σε εργαλεία, παιχνίδια και συσκευές.

Οι συγκεκριμένοι κινητήρες, που χρησιμοποιούμε και εμείς, είναι 6V ο καθένας.

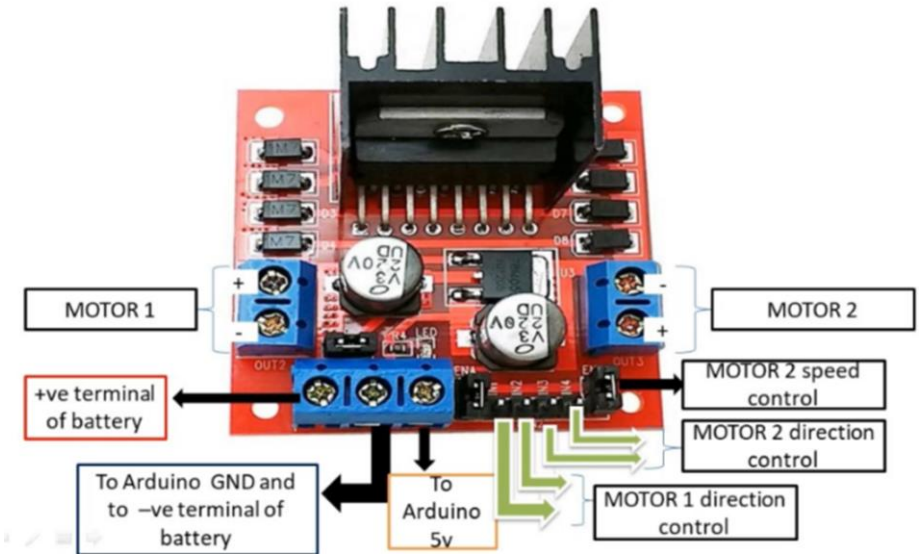
Εικόνα 4: DC motors

4.3.2 L298N Motor Driver

Το L298N είναι ένα ολοκληρωμένο μονολιθικό κύκλωμα σε πακέτα Multiwatt και Power SO20 15-καναλιών. Είναι ένα υψηλής τάσης, υψηλού ρεύματος διπλού οδηγού πλήρους

γέφυρας, το οποίο έχει συμφωνήσει να δέχεται τυπικά επαγωγικά φορτία μονάδας TTL λογικής στάθμης, όπως DC κινητήρες.

Δύο είσοδοι ενεργοποίησης παρέχονται για να ενεργοποιούν ή να απενεργοποιούν την συσκευή ανεξάρτητα από τα σήματα εισόδου. Οι εκπομποί των κατώτερων τρανζίστορ κάθε γέφυρας συνδέονται μεταξύ τους και ο αντίστοιχος εξωτερικός ακροδέκτης μπορεί να χρησιμοποιηθεί για τη σύνδεση μιας εξωτερικής αντίστασης ανίχνευσης. Επίσης παρέχεται μια πρόσθετη είσοδος τροφοδοσίας έτσι ώστε η λογική να λειτουργεί σε χαμηλότερη τάση.



Εικόνα 5: L298N Motor Driver

+5V : Παρέχει 5V εάν έχει αφαιρεθεί ο jumper. Λειτουργεί ως έξοδο 5V εάν ο jumper είναι στην θέση του.
Jumper : εάν τροφοδοτείτε το τσιπ πάω από 12V θα πρέπει να αφαιρέσετε τον Jumper.

Signal on the enable pin: HIGH: Motor enabled
LOW: Motor not enabled
PWM : Motor enabled: speed proportional to duty cycle

ΧΑΡΑΚΤΗΡΙΣΤΙΚΆ

- Υψηλή τάση λειτουργίας, η οποία μπορεί να φτάσει τα 40 volts.

- Μεγάλο ρεύμα εξόδου, το στιγμιαίο ρεύμα κορυφής μπορεί να είναι μέχρι 3A.
- Με ονομαστική ισχύ 25W.
- Δύο ενσωματωμένες γέφυρες H, υψηλής τάσης, μεγάλου ρεύματος, πλήρους οδηγού γέφυρας, οι οποίες μπορούν να χρησιμοποιηθούν για κινητήρες συνεχούς ρεύματος, βηματικούς κινητήρες, πηνία ρελέ και άλλα επαγωγικά φορτία.
- Χρησιμοποιούν τυπικό σήμα λογικής στάθμης για έλεγχο.
- Μπορεί να οδηγήσει έναν 2-phase stepper motor ή 4-phase stepper motor, και 2-phase DC motors.
- Διαθέτει έναν πυκνωτή φίλτρου υψηλής χωρητικότητας και μια δίοδο ελεύθερης περιστροφής που προστατεύει τις συσκευές στο κύκλωμα από βλάβη από το αντίστροφο ρεύμα ενός επαγωγικού φορτίου, ενισχύοντας την αξιοπιστία.
- Η μονάδα μπορεί να χρησιμοποιήσει τον ενσωματωμένο σταθερό σωλήνα 78M05 για να λάβει 5v από την τροφοδοσία. Αλλά για να προστατέψετε το τσιπ του 78M05 από ζημιά, όταν η τάση του κινητήρα είναι μεγαλύτερη από 12v, θα πρέπει να χρησιμοποιηθεί μια εξωτερική τροφοδοσία 5v.
- Drive voltage: 5-35V; logic voltage: 5V.

Πίνακας 1: ΠΙΝΑΚΑΣ ΑΛΗΘΕΙΑΣ ΤΩΝ Pins :

DIRECTION	**INPUT 1**	**INPUT 2**	**INPUT 3**	**INPUT 4**
Forward	0	1	0	1
Backward	1	0	1	0
Right	0	1	0	0
Left	0	0	0	1
Stop	0	0	0	0

INPUT 1 ΚΑΙ INPUT 2: RIGHT MOTOR, INPUT 3 ΚΑΙ INPUT 4: LEFT MOTOR

Επιλέξαμε τον L298N γιατί πρώτον μας εξυπηρετεί λόγω του ότι θα χρησιμοποιήσουμε μόνο 2 τροχούς και δεύτερον είναι μια καλή και οικονομική λύση.

Περισσότερα τεχνικά χαρακτηριστικά σχετικά με τον L298N θα βρείτε στον σύνδεσμο που βρίσκεται στη βιβλιογραφία με αριθμό[4].

4.4 Συνδεσμολογία του συστήματος μας

Σύνδεση wemos D1 mini με L298N και wemos D1 mini με τον αισθητήρα:

Wemos D1 mini	**L298N**
D8	IN1
D7	IN2
D6	IN3
D5	IN4
Wemos D1 mini	**Αισθητήρας**
A0	Κίτρινο καλώδιο (output)

— Τα 5V και το Gnd του Wemos συνδέονται με το + και – στο breadboard αντίστοιχα.

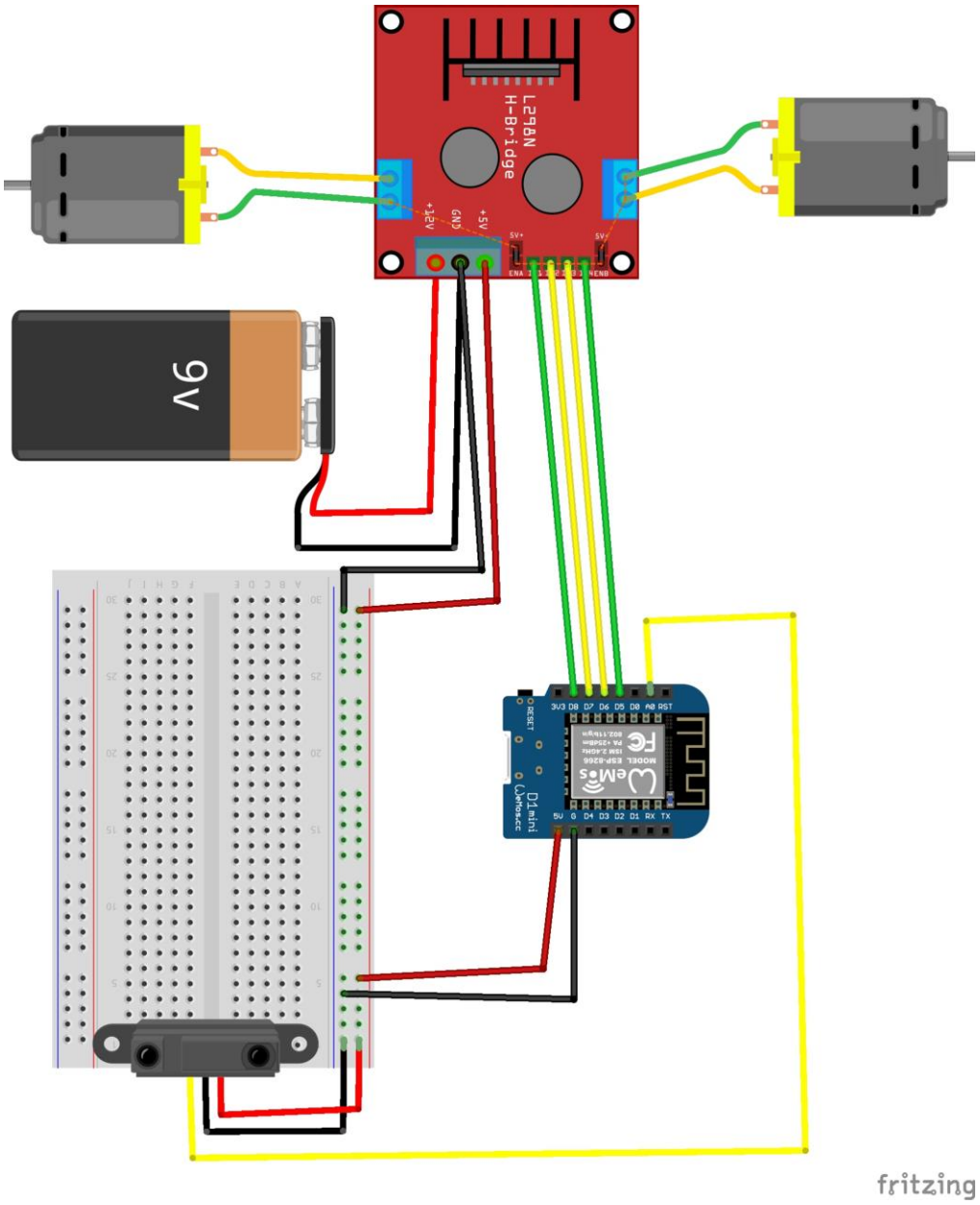
— Τα +5V και το Gnd του L298N συνδέονται με το + και – στο breadboard αντίστοιχα.

— Το μαύρο καλώδιο(γείωση) και το κόκκινο καλώδιο(5V) του αισθητήρα συνδέονται με το + και – στο breadboard αντίστοιχα.

— Η μπαταρία συνδέεται με το l298n , το κόκκινο καλώδιο με τα 12V και το μαύρο καλώδιο με το Gnd .

Σύνδεση L298N με τους κινητήρες:

L298N	**DC motors**
OUT1	Αριστερός κινητήρας πάνω
OUT2	Αριστερός κινητήρας κάτω
OUT3	Δεξιός κινητήρας κάτω
OUT4	Δεξιός κινητήρας πάνω



Εικόνα 6: Συνδεσμολογία του project μας

5. ΛΟΓΙΣΜΙΚΟ ΤΟΥ ΕΚΠΑΙΔΕΥΤΙΚΟΥ ΡΟΜΠΟΤ

5.1 Firmata

5.1.1 Τι είναι η firmata;

Η firmata είναι βιβλιοθήκη η οποία υλοποιεί το πρωτόκολλο firmata για επικοινωνία με λογισμικό στον κεντρικό υπολογιστή. Αυτό μας επιτρέπει να γράψουμε το προσαρμοσμένο firmware χωρίς να χρειάζεται να δημιουργήσουμε το δικό μας πρωτόκολλο και αντικείμενα για το περιβάλλον προγραμματισμού που χρησιμοποιούμε.

Το πρωτόκολλο firmata μπορεί να υλοποιηθεί σε firmware σε οποιαδήποτε αρχιτεκτονική μικροελεγκτών. Η firmata είναι συμβατή με Arduino και Arduino-compatible συσκευές. Αρκετές βιβλιοθήκες firmata έχουν υλοποιηθεί σε διάφορες δημοφιλείς γλώσσες προγραμματισμού όπως:

- Python
- JavaScript
- Java
- Perl
- Ruby
- PHP etc

Υπάρχουν δύο βασικά μοντέλα χρήσης της firmata:

Το πρώτο είναι όταν θέλουμε να το χρησιμοποιήσουμε τις διάφορες μεθόδους που παρέχει η βιβλιοθήκη firmata για την επιλεκτική αποστολή και λήψη δεδομένων μεταξύ της συσκευής Arduino και του λογισμικού που εκτελείται στον κεντρικό υπολογιστή. Για παράδειγμα ένας χρήστης μπορεί να στείλει αναλογικά δεδομένα στον κεντρικό υπολογιστή χρησιμοποιώντας το Firmata.sendAnalog.

Το δεύτερο και ποιο κοινό μοντέλο είναι να φορτώσουμε ένα αρχείο γενικής χρήσης όπου ονομάζεται StandardFirmata (ή μια από τις παραλλαγές του όπως είναι το StandardFirmataWifi ή StandardFirmataEthernet, αναλόγως τις ανάγκες μας) στο Arduino και στην συνέχεια χρησιμοποιούμε τον κεντρικό υπολογιστή αποκλειστικά για να αλληλεπιδράσουμε με το Arduino. Επίσης, παρέχεται ένα εργαλείο από την Firmata όπου, μπορούμε να

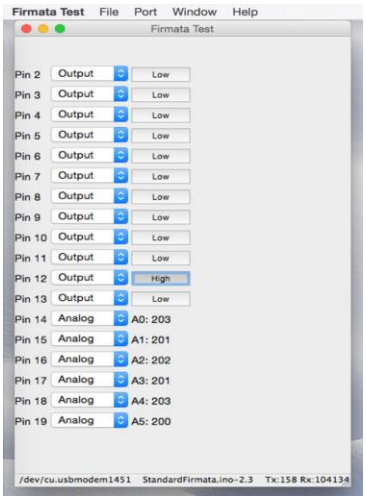
δημιουργήσουμε το δικό μας αρχείο firmata με βάση τους κανόνες τις firmata. Ο σύνδεσμος για το παραπάνω firmata builder βρίσκεται στην βιβλιογραφία με αριθμό[7].

(Αρχείο → Παραδείγματα → Firmata → ...)

Για τις ανάγκες του παρών πρότζεκτ χρησιμοποιήσαμε το αρχείο γενικής χρήσης StandardFirmataWiFi διότι θέλαμε να συνδέσουμε την συσκευή μας (wemos d1 mini) με το λογισμικό, το οποίο θα δούμε παρακάτω, μέσω Wi-Fi.

5.1.2 Διαφορές StandardFirmata και StandardFirmataWifi

Όπως προ αναφέραμε παραπάνω η firmata είναι ένα πρωτόκολλο, δηλαδή ένα σύνολο κανόνων, που χρησιμοποιείτε για την επικοινωνία των εφαρμογών με τους μικροελεκτές. Το Arduino IDE μας παρέχει κάποια βασικά παραδείγματα χρήσης της firmata, όπως και το γενικής χρήσης αρχείο StandardFirmata. Υπάρχουν κάποια πιο εξειδικευμένα παραδείγματα όπου βασίζονται στην StandardFirmata. Μερικά από αυτά είναι η StandardFirmata WiFi, StandardFirmata Ethernet, StandardFirmata Plus και τα λοιπά, καθένα από αυτά χρησιμοποιείται αναλόγως με τις ανάγκες μας. Αυτά τα παραδείγματα θα πρέπει να εγκατασταθούν ξεχωριστά.



Εικόνα 7: Παράδειγμα της StandardFirmata

Παρακάτω υπάρχει ο οδηγός εγκατάστασής αυτών.

Η διαφορά του StandardFirmata με το StandardFirmata WiFi είναι ότι το StandardFirmata είναι βιβλιοθήκη λογισμικού της firmata όπου συνδέεται ενσύρματα με το board μας ενώ το StandardFirmata WiFi επιτρέπει την χρήση της firmata μέσω σύνδεσης TCP και μπορεί να διαμορφωθεί είτε ως TCP Client είτε ως TCP Server.

5.2 Arduino IDE

Το Arduino IDE είναι μια εφαρμογή όπου την χρησιμοποιούμε για να προγραμματίζουμε και να ανεβάζουμε προγράμματα σε συμβατές πλακέτες arduino.

Εικόνα 8: Arduino IDE

5.2.1 Εγκατάσταση wemos στο Arduino IDE(για Windows)

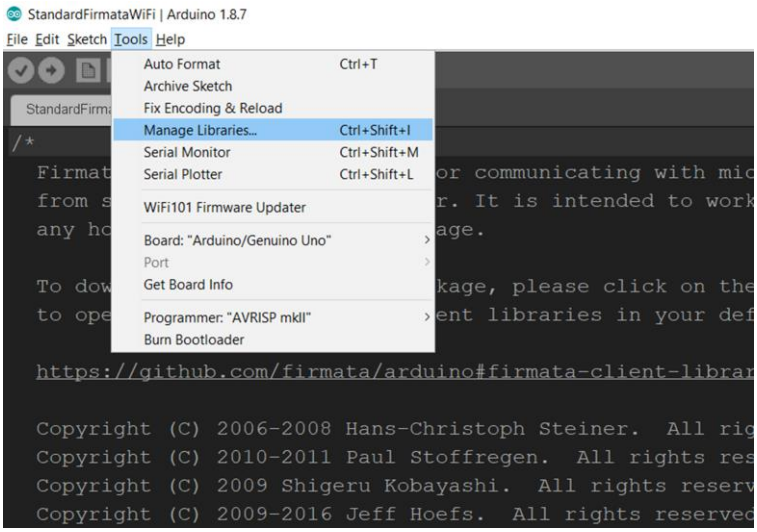
1. Εγκατάσταση του Arduino IDE(έκδοση 1.8 ή νεότερη έκδοση)

   https://www.arduino.cc/en/Main/OldSoftwareReleases

Εικόνα 9: Μενου του arduino IDE

2.Ανοίγουμε το Arduino IDE (για να προσθέσουμε την esp8266 βιβλιοθήκη):

- Από Αρχείο > Προτιμήσεις, επιπλέον URLs διαχειριστή πλακετών και αντιγράφουμε το παρακάτω σύνδεσμο http://arduino.esp8266.com/stable/package_esp8266com_index.json
- Από Εργαλεία > Πλακέτα > Διαχειριστής βιβλιοθηκών, ψάχνουμε για D1 mini και κάνουμε Εγκατάσταση.



Εικόνα 10: Βήμα 1 για την εγκατάσταση του wemos στο Arduino IDE

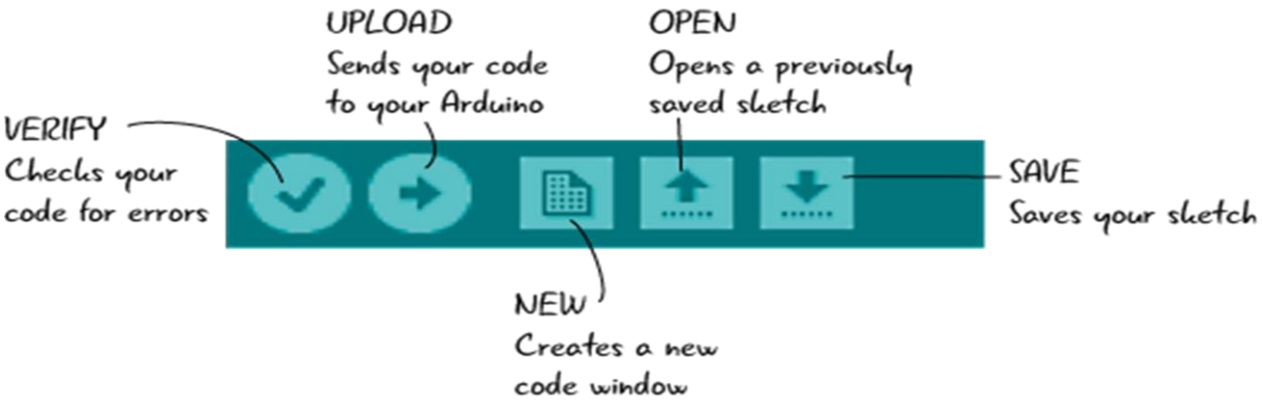
3. Σύνδεση:

- Από Εργαλεία > Θύρα επέλεξε την σωστή COM port, όπως έχει αντιστοιχιστεί στην συσκευή. (Θα πρέπει να έχετε συνδέσει την συσκευή σας στο υπολογιστή σας για να μπορεί να βρεθεί το COM port)

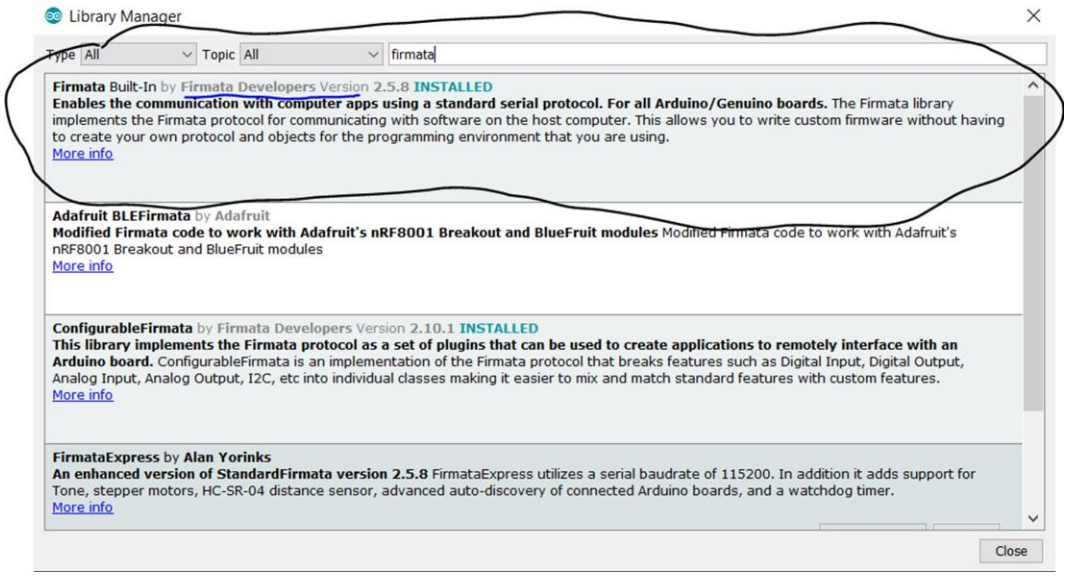
4.Testing:

- Από Αρχείο > Παραδείγματα >ESP8266 ανοίγουμε το αρχείο Blink.

- Ελέγχουμε την αντιστοίχιση των Pins.
- Επικύρωση(verify) το αρχείο.
- Ανέβασμα(upload) το αρχείο στην συσκευή μας.



Εικόνα 11: Επεξήγηση των λειτουργικών κουμπιών του arduino IDE



Εικόνα 12: Βήμα 1 για την εγκατάσταση της Firmata στο Arduino IDE

3.

5.2.2 Εγκατάσταση Firmata στο Arduino IDE

1. Εγκατάσταση του Arduino IDE (version 1.8 ή νεότερη έκδοση).

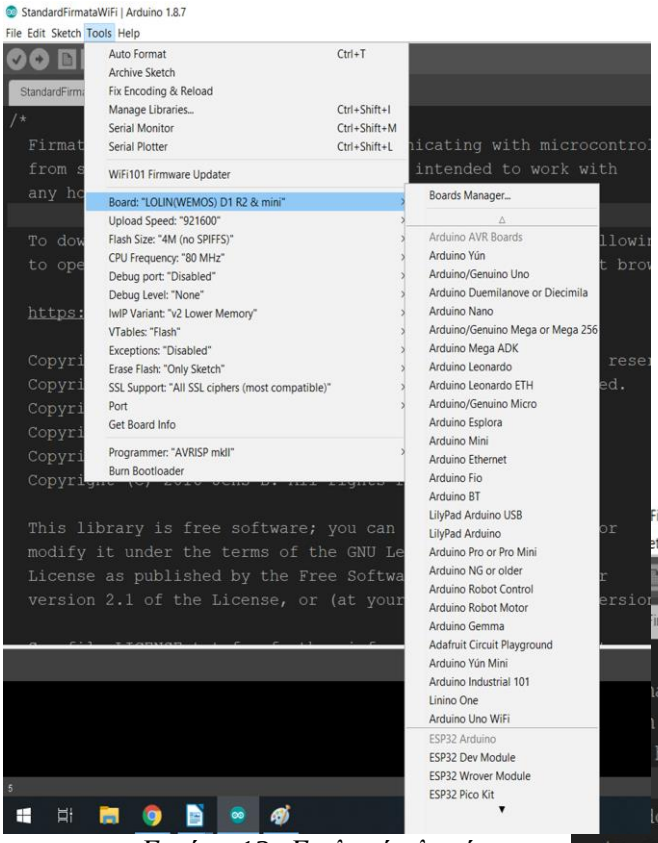
https://www.arduino.cc/en/Main/OldSoftwareReleases

2. Κατεβάζουμε την βιβλιοθήκη firmata.

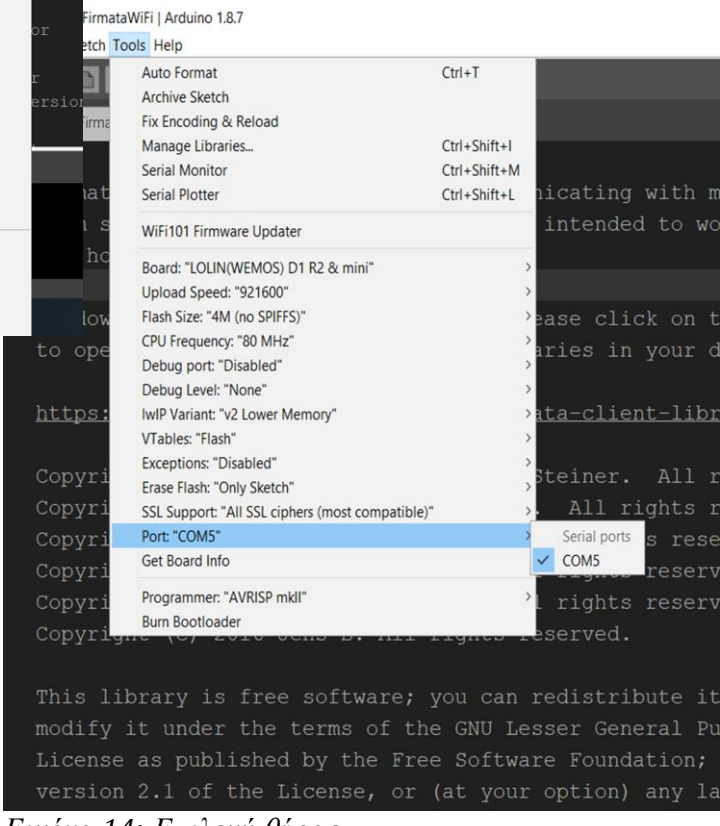
Από Εργαλεία > Διαχείριση βιβλιοθηκών... > Κάνουμε αναζήτηση firmata

3. Συνδέουμε την συσκευή μας στο κεντρικό υπολογιστή.

4. Επιλέγουμε την θύρα και την πλακέτα την οποία θα συνδέσουμε.

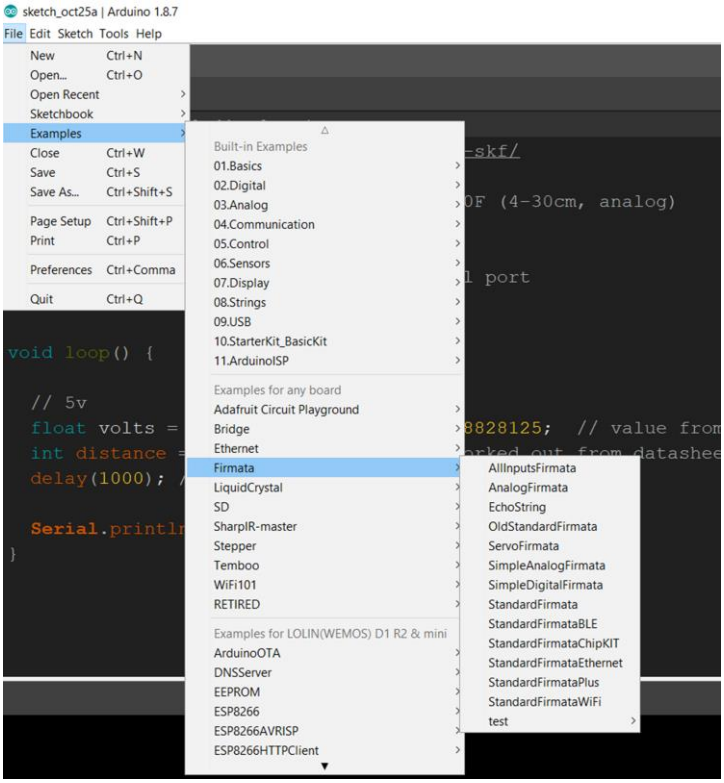


Εικόνα 13: Επιλογή πλακέτας



Εικόνα 14: Επιλογή θύρας

5. Ανεβάζουμε το αρχείο firmata που θέλουμε να χρησιμοποιήσουμε.



Εικόνα 15: Επιλογή αρχείου firmata

5.3 Johnny-five

Το johnny-five είναι μια JavaScript Robotics & IoT πλατφόρμα, βασισμένη στην firmata. Υποστηρίζει ασύγχρονο προγραμματισμό. Αυτό σημαίνει ότι οδηγείται από γεγονότα, επομένως μειώνει την καθυστέρηση και τον "κωδικό αποκλεισμού", που σημαίνει ότι δεν χρειάζεται πλέον να ανησυχείτε για κάποια διαδικασία που διαρκεί τόσο πολύ ώστε να εμποδίζει ένα άλλο κομμάτι κώδικα. Είναι συμβατό με πολλά συστήματα ελέγχου όπως το Arduino UNO και πολλά άλλα. Το johnny-five δημοσιεύθηκε το 2012 από τον Rick Waldron.

Εικόνα 16: Logo Johnny-five

Για να τρέξουμε οποιοδήποτε πρόγραμμα με johnny-five θα πρέπει αν ακολουθήσουμε τα παρακάτω βήματα: (για Windows)

- Εγκαθιστούμε το node.js.
- Ελέγχουμε ότι έχουμε συμβατή πλακέτα. Για να το επιβεβαιώσετε επισκεφτείτε το παρακάτω σύνδεσμο http://johnny-five.io/platform-support/

- Κατεβάσετε την βιβλιοθήκη του johnny-five γράφοντας “npm install johnny-five” στο cmd.
- Τρέξτε το πρόγραμμά σας! node test.js

Επιλέξαμε την πλατφόρμα του johnny-five γιατί είναι μια από τις πλατφόρμες όπου ήταν συμβατή με firmata, όπως και με την βιβλιοθήκη blockly. Από τον ιστότοπο του johnny-five χρησιμοποιήσαμε αρκετά βοηθητικά παραδείγματα που μας παρέχει. Για παραπάνω παραδείγματα επισκεφτείτε τον ιστότοπο του Johnny-five. Ο σύνδεσμος βρίσκεται στην βιβλιογραφία με αριθμό[11]

Για παράδειγμα, για να χρησιμοποιήσουμε τον αισθητήρα απόστασης χρειαστήκαμε τον παρακάτω κώδικα:

Usage

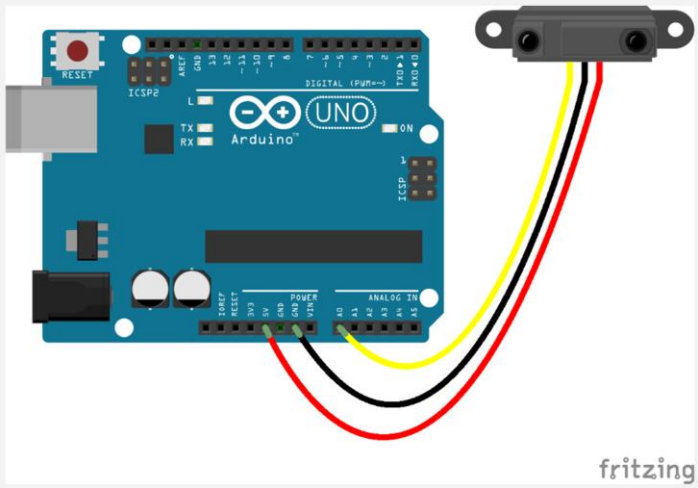
```
var five = require("johnny-five");
var board = new five.Board();

board.on("ready", function() {
  var proximity = new five.Proximity({
    controller: ...,
    pin: "A0"
  });

  proximity.on("data", function() {
    console.log("inches: ", this.inches);
    console.log("cm: ", this.cm);
  });
});
```

Εικόνα 17: Κώδικας από J5 για χρήση αισθητήρα απόστασης.

Component Initialization

Analog GP2Y0A21YK (and friends)

```
/*
  Use with:

  - GP2Y0A21YK
  - GP2D120XJ00F
  - GP2Y0A02YK0F
  - GP2Y0A41SK0F
*/

new five.Proximity({
  controller: "GP2Y0A21YK",
  pin: "A0"
});
```



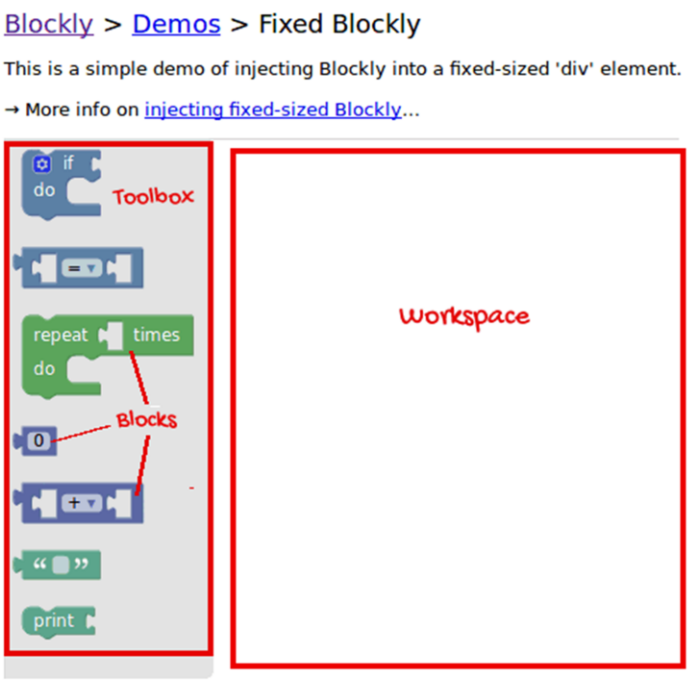
Εικόνα 18: J5-GP2Y0A21YK

5.4 Βιβλιοθήκες και λοιπές πλατφόρμες

5.4.1 Βιβλιοθήκη Blockly

Όπως είδαμε παραπάνω το blockly είναι μια γραφική διεπαφή που βοηθάει χρήστες χωρίς εμπειρία στον προγραμματισμό, να προγραμματίζουν χωρίς να τους απασχολούν τα συντακτικά λάθη. Επίσης είδαμε ότι υπάρχουν δυο τρόποι να χρησιμοποιήσουμε το Blockly, ο ένας είναι αυτός που μόλις αναφέραμε και ο δεύτερος τρόπος είναι ο τρόπος που χρησιμοποιήσαμε εμείς, δηλαδή ως προγραμματιστές για να φτιάξουμε το δικό μας γραφικό περιβάλλον. Αλλά για να υπάρχει λειτουργικότητα των block μας και του γραφικού μας περιβάλλοντος χρειαζόμαστε την βιβλιοθήκη Blockly.

Προγραμματιστικά το Blockly μας επιτρέπει να φτιάξουμε τα δικά μας blocks. Είναι διαθέσιμο ένα εργαλείο από το Blockly το οποίο μας επιτρέπει να δημιουργήσουμε τα δικά μας blocks με την λειτουργικότητα τον πρωτότυπων blocks. Στην παρακάτω εικόνα βλέπουμε το παράδειγμα του περιβάλλοντος Blockly το οποίο έχουμε χρησιμοποιήσει και εμείς στη δική μας εφαρμογή.



Εικόνα 19: Blockly

Για την δημιουργία του γραφικού περιβάλλοντος, όπως και του δικού μας πρότζεκτ, χρησιμοποιήσαμε τα παραδείγματα από την επίσημη σελίδα του Blockly. Παρακάτω φαίνεται ο κώδικας όπου χρησιμοποιήσαμε για την δημιουργία του workspace και του toolbox.

Ο πιο απλός τρόπος για να τοποθετήσετε τον Blockly σε μια ιστοσελίδα είναι να τον εισάγουμε σε μια κενή ετικέτα 'div’.

Χώρος εργασίας σταθερού μεγέθους:

Πρώτον, συμπεριλάβετε το κύριο script Blockly και τα μπλοκ πυρήνα που έχουν οριστεί. Σημειώστε ότι η διαδρομή μπορεί να διαφέρει, ανάλογα με το πού βρίσκεται η σελίδα σας σε σχέση με το Blockly:

```
<script src="blockly_compressed.js"></script>
<script src="blocks_compressed.js"></script>
```

Στη συνέχεια, συμπεριλάβετε τα μηνύματα για τη γλώσσα του χρήστη (σε αυτή την περίπτωση στα Αγγλικά):

```
<script src="msg/js/en.js"></script>
```

Προσθέστε ένα κενό div κάπου στο σώμα της σελίδας και ορίστε το μέγεθός του:

```
<xml id="toolbox" style="display: none">
  <block type="controls_if"></block>
  <block type="controls_repeat_ext"></block>
  <block type="logic_compare"></block>
  <block type="math_number"></block>
  <block type="math_arithmetic"></block>
  <block type="text"></block>
  <block type="text_print"></block>
</xml>
```

Προσθέστε τη δομή της εργαλειοθήκης οπουδήποτε στη σελίδα:

Τέλος, καλέστε τα παρακάτω για να εισάγετε το Blockly σε ένα κενό div. Αυτό το σενάριο πρέπει να βρίσκεται στο κάτω μέρος της σελίδας ή να καλείται από το συμβάν φόρτωσης.

```
<script>
  var workspace = Blockly.inject('blocklyDiv',
      {toolbox: document.getElementById('toolbox')});
</script>
```

Η μεταβλητή χώρου εργασίας δεν χρησιμοποιείται αυτή τη στιγμή, αλλά θα γίνει σημαντική αργότερα όταν κάποιος θέλει να αποθηκεύσει τα μπλοκ ή να δημιουργήσει κώδικα. Εάν στην ίδια σελίδα εισάγονται περισσότερες από μία περιπτώσεις Blockly, βεβαιωθείτε ότι κάθε αποθηκευμένος χώρος εργασίας αποθηκεύεται σε διαφορετική μεταβλητή.

Εργαλειοθήκη:

Η εργαλειοθήκη είναι το πλευρικό μενού από το οποίο ο χρήστης μπορεί να δημιουργήσει νέα μπλοκ. Η δομή της εργαλειοθήκης καθορίζεται με XML, η οποία μπορεί να είναι είτε δέντρο κόμβων, είτε συμβολική συμβολοσειρά. Αυτή η XML διαβιβάζεται στο Blockly όταν εισάγεται στη σελίδα. Αν δεν σας αρέσει να πληκτρολογείτε XML με μη αυτόματο τρόπο, σας συνιστούμε να ελέγξετε τα Εργαλεία προγραμματισμού Blockly. Με αυτό, μπορείτε να δημιουργήσετε μια εργαλειοθήκη και να δημιουργήσετε αυτόματα την εργαλειοθήκη XML χρησιμοποιώντας μια οπτική διεπαφή.

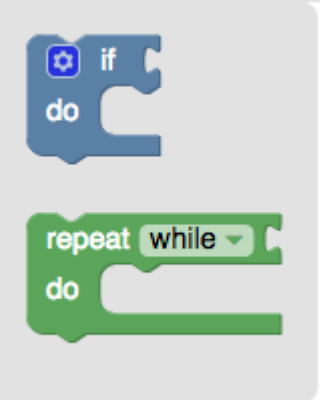
Εδώ είναι ένα ελάχιστο παράδειγμα, χρησιμοποιώντας ένα δέντρο των κόμβων:

```
<xml id="toolbox" style="display: none">
  <block type="controls_if"></block>
  <block type="controls_whileUntil"></block>
</xml>
<script>
  var workspace = Blockly.inject('blocklyDiv',
      {toolbox: document.getElementById('toolbox')});
</script>
```

Εδώ είναι το ίδιο παράδειγμα, χρησιμοποιώντας μια παράσταση :

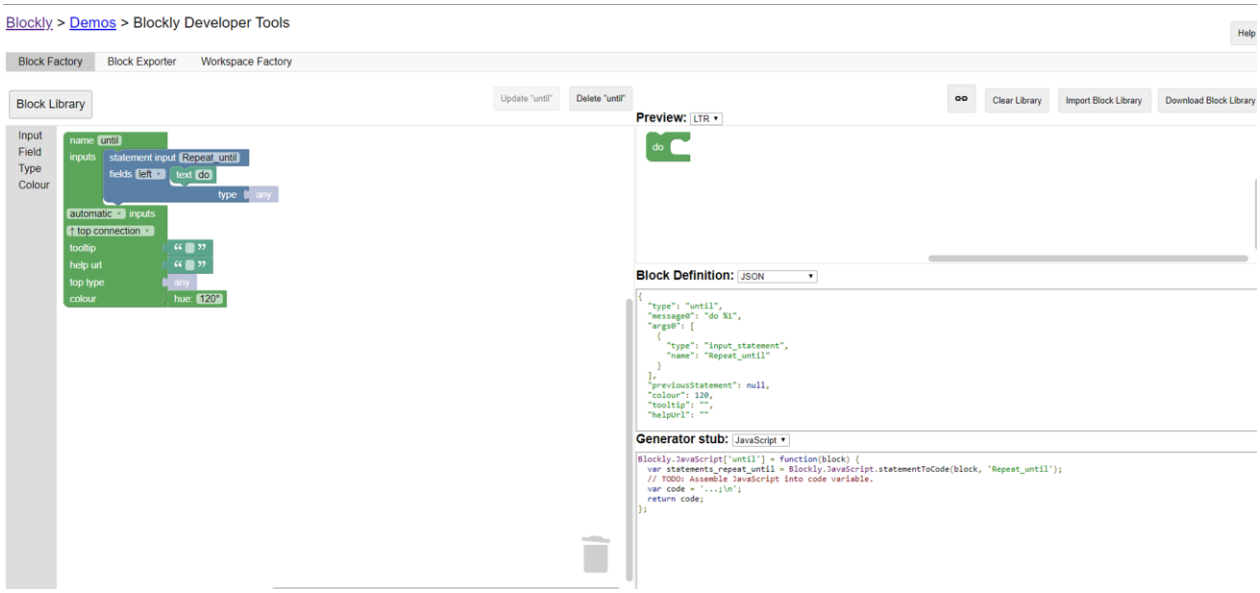
```
<script>
  var toolbox = '<xml>';
  toolbox += '  <block type="controls_if"></block>';
  toolbox += '  <block type="controls_whileUntil"></block>';
  toolbox += '</xml>';
  var workspace = Blockly.inject('blocklyDiv', {toolbox: toolbox});
</script>
```

Και τα δύο παραπάνω δημιουργούν την ίδια εργαλειοθήκη με δύο μπλοκ:



Υπάρχουν πολλά παραδείγματα ακόμα, όπως και έτοιμα πρότυπα block, στην ιστοσελίδα του Blockly. Ο λόγος που δεν χρησιμοποιήσαμε τα πρότυπα block είναι επειδή εμείς θέλουμε συγκεκριμένη λειτουργία των block, όπου είναι βασισμένα σύμφωνα με την λειτουργικότητα του αυτόνομου αυτοκινήτου που έχουμε φτιάξει, για αυτόν αυτό δημιουργήσαμε τα customized block. Ο σύνδεσμος υπάρχει στην βιβλιογραφία με αριθμό[9].

Για την δημιουργία customized block χρησιμοποιήσαμε το προγραμματιστικό εργαλείο του Blockly του οποίου τον σύνδεσμο θα τον βρείτε στην βιβλιογραφία με αριθμό[10]. Παρακάτω μπορούμε να δούμε το προγραμματιστικό εργαλείο των blocks. Όπως φαίνεται στην παρακάτω εικόνα πως μπορούμε να δημιουργήσουμε τα blocks του Blockly με block του Blockly! Επιλέγουμε από την εργαλειοθήκη αριστερά τα κριτήρια που θέλουμε να έχει το block μας και αυτόματα το εργαλείο αυτό το μεταφράζει σε κώδικα, είτε JSON ή JavaScript. Έπειτα αντιγράφουμε τον κώδικα που μας δόθηκε στον δικό μας κώδικα. Προσοχή όμως, θα χρειαστεί να εγκαταστήσουμε την βιβλιοθήκη του Blockly στο πρότζεκτ μας. Παρακάτω θα αναλύσουμε με περισσότερες πληροφορίες ό,τι αφορά την βιβλιοθήκη Blockly.



Εικόνα 20: Blockly developer tool 1

Εικόνα 21: Blockly developer tool 2

5.4.2 Βιβλιοθήκη React

Η react είναι μια javascript βιβλιοθήκη για την δημιουργία user interface. Υποστηρίζεται από το facebook και μια κοινότητα προγραμματιστών και εταιρειών. Με την react μπορούμε να δημιουργήσουμε ενσωματωμένα εξαρτήματα που διαχειρίζονται τη δική τους κατάσταση, στη συνέχεια τα συνθέτουμε όλα μαζί και δημιουργούμε σύνθετα user interfaces.

Στο δικό μας πρότζεκτ έχουμε χρησιμοποιήσει την react για να δημιουργήσουμε το παράθυρό της εφαρμογής μας, όπως και κάποια άλλα εξαρτήματα. Το workspace του Blockly που είδαμε παραπάνω, είναι μέσα στο παράθυρο που δημιουργήσαμε με την react.

Για παραδείγματα react επισκεφτείτε τον σύνδεσμο με αριθμό [17] που βρίσκεται στην βιβλιογραφία.

Εικόνα 22: Logo React

5.4.3 Βιβλιοθήκη EtherPort-Client

Το EtherPort είναι ένα transport layer που λειτουργεί σε συνδυασμό με την firmata για να επιτρέψει την επικοινωνία με μια πλακέτα arduino ή άλλες συμβατές συσκευές. Το etherPort-client χρησιμοποιείται για την υλοποίηση πλακέτων συμβατών με την firmata, συνδέοντας τους διανομείς για τον έλεγχο μιας πλακέτας από μια απομακρυσμένη οντότητα. Η βιβλιοθήκη αυτή είναι συμβατή με το johnny-five και το μόνο που χρειαζόμαστε για να την χρησιμοποιήσουμε είναι να την εγκαταστήσουμε στον υπολογιστή μας. Εμείς την εγκαταστήσαμε με την παρακάτω εντολή, “npm install johnny-five etherport” στο cmd.

```
//Libraries for Wemos
var five = require('johnny-five'); //johnny-five library
var EtherPortClient = require('etherport-client').EtherPortClient; //etherport library
```

Εικόνα 23: Etherport-1

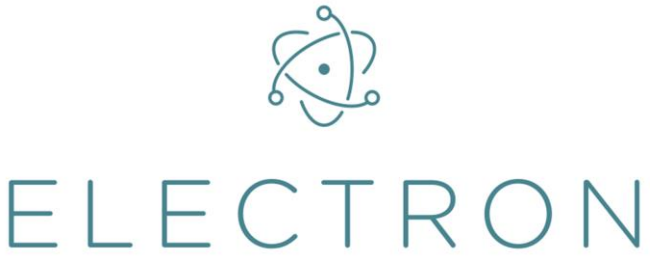
5.4.4 Electron

Η electron θεωρείται ένα software framework ανοιχτού κώδικα σχεδιασμένο από την github. Η electron μας επιτρέπει την ανάπτυξη εφαρμογών GUI για υπολογιστές,

```
98    //Setup the Wemos D1 mini board
99    board = new five.Board({
100     port: new EtherPortClient({
101       host: '192.168.178.77', // IP of Wemos
102       port: 3030
103     }),
104     timeout: 10000,
105     repl: false
106   });
107
108   //Attempt to connect the board
109   ipcMain.on('connectBoard', (event, code) => {
110     board = null;
111     board = new five.Board({
112       port: new EtherPortClient({
113         host: '192.168.178.77', // IP of Wemos
114         port: 3030
115       }),
116       timeout: 10000,
117       repl: false
118     });
119   });
```

Εικόνα 24: Etherport-2

χρησιμοποιώντας web technologies όπως JavaScript, HTML και CSS.



Εικόνα 25: Logo Electron

Εμείς χρησιμοποιήσαμε την electron για να μετατρέψουμε την εφαρμογή της react σε desktop εφαρμογή. Επίσης η electron μας βοηθάει να συνδυάσουμε το node.js με την react. Για να καταφέρουμε να δημιουργήσουμε ένα .exe αρχείο μέσα από την electron, κατεβάσαμε το electron-builder μέσω node.js. Εφόσον εγκαταστήσουμε το electron-builder, μπορούμε να το χρησιμοποιήσουμε στην desktop εφαρμογή μας. Στην προκειμένη περίπτωση είναι για να μετατρέψουμε την react σε desktop εφαρμογή. Για παράδειγμα στον κώδικα μας, μετά που κατεβάσαμε την βιβλιοθήκη, με την παρακάτω εντολή το ενσωματώσαμε στο δικό μας project.

Πιο συγκεκριμένα παραδείγματα χρήσης:

```
1  const electron = require('electron');
2  const app = electron.app;
3  const BrowserWindow = electron.BrowserWindow;
4  const path = require('path');
5  const isDev = require('electron-is-dev');
6  const { ipcMain } = require('electron');
7
```

Εικόνα 26: Χρήση electron στο project μας-1

```
124
125  app.on('ready', createWindow);
126
127  app.on('window-all-closed', () => {
128    if (process.platform !== 'darwin') {
129      app.quit();
130    }
131  });
132
133  app.on('activate', () => {
134    if (mainWindow === null) {
135      createWindow();
136    }
137  });
```

Εικόνα 27: Χρήση electron στο project μας-2

Παραπάνω φαίνονται οι τρεις φάσης της εφαρμογής. Η πρώτη είναι όταν ανοίγει η εφαρμογή και είναι έτοιμη για χρήση, η δεύτερη είναι για να κλείσουμε τελείως την εφαρμογή και η τρίτη όταν είναι ενεργή αλλά στο παρασκήνιο. Στην επόμενη φωτογραφία βλέπουμε την δημιουργία του παραθύρου.

```
14  function createWindow() {
15    mainWindow = new BrowserWindow({
16      width: 1360,
17      height: 768,
18      webPreferences: {
19        nodeIntegration: true
20      }
21    });
```

Εικόνα 28: Χρήση electron στο project μας-3

Για περισσότερα παραδείγματα της electron μπορείτε να επισκεφτείτε τον ιστότοπο της electron. Ο σύνδεσμος βρίσκεται στην βιβλιογραφία με αριθμό[18].

6. ΤΟ ΔΙΚΟ ΜΑΣ FIRMWARE

6.1 Εισαγωγή

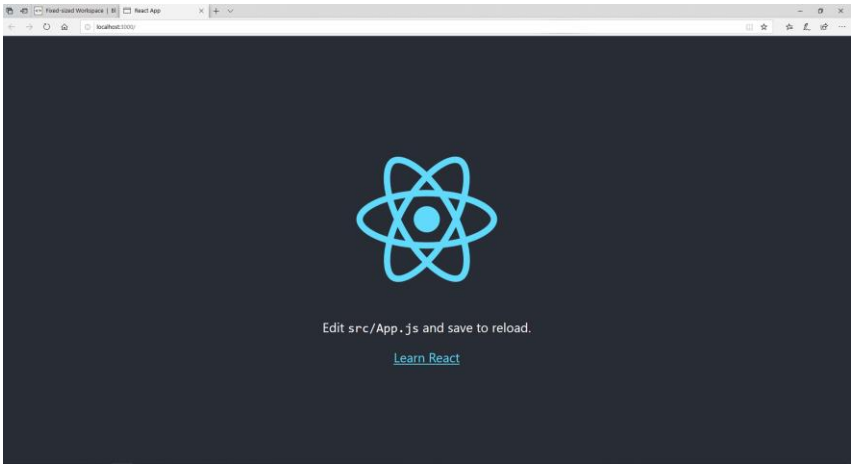
Η βάση του παρών πρότζεκτ είναι ένα εργαλείο της create-react-app, θα μπορούσε κανείς να το πει και boilerplate, με την διαφορά ότι αυτό δεν το κατεβάζεις από κάποιον ιστότοπο κατευθείαν, αλλά το κατεβάζεις μέσω node.js. Το create-react-app είναι ένα εργαλείο, κατασκευασμένο από προγραμματιστές του facebook, μας δίνει το ξεκίνημα για την κατασκευή εφαρμογών react. Μας εξοικονομεί χρόνο στο να κάνουμε κάποια setup και κάποιες ρυθμίσεις. Για περισσότερες πληροφορίες σχετικά με το create-react-app θα βρείτε στην βιβλιογραφία με αριθμό[8]. Για την δημιουργία του create-react-app χρειαζόμαστε το node.js. Εφόσον το έχουμε εγκατεστημένο γράφουμε στο cmd τις παρακάτω εντολές:

npx create-react-app my-app

cd my-app

npm start ή yarn start

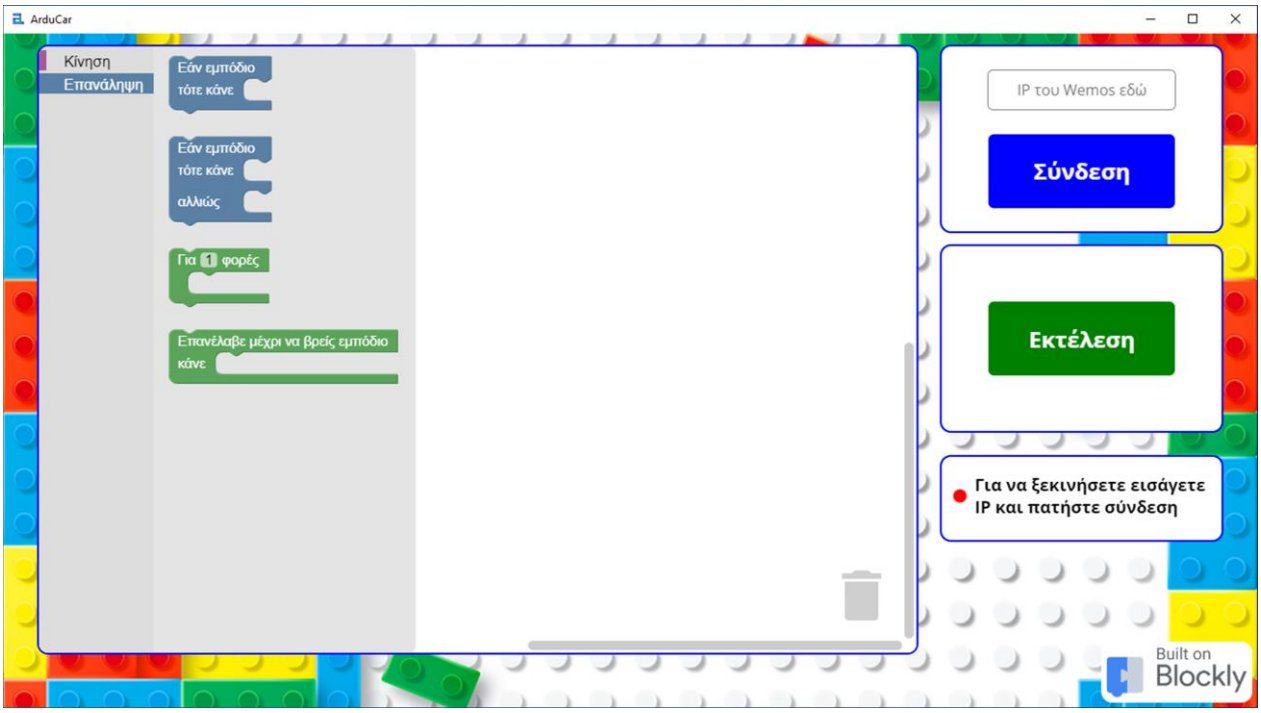
Και μας εμφανίζεται ο παρακάτω ιστότοπος:



Εικόνα 29: Boilerplate

Έπειτα για να αλλάξουμε τον ιστότοπο σε desktop εφαρμογή χρειαστήκαμε την βοήθεια της electron. Όπως αναφέραμε παραπάνω, η electron μας βοηθάει να μετατρέψουμε την react σε

desktop εφαρμογή. Και τέλος με την χρήση του johnny-five, όπου έχει συμβατότητα με firmata και Blockly, δημιουργήσαμε το δικό μας firmware.

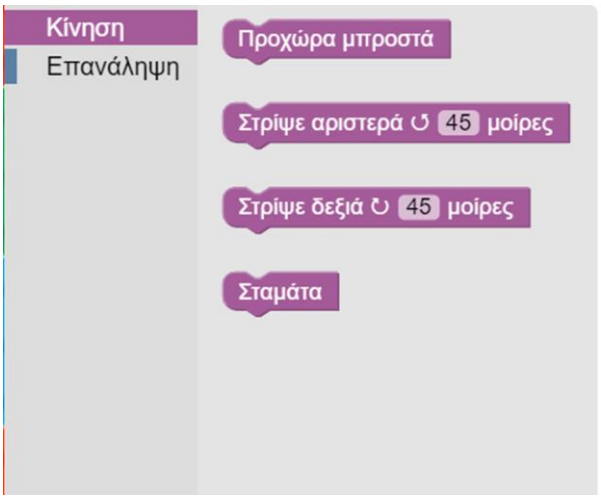


Εικόνα 30: Τελικό γραφικό περιβάλλον της εφαρμογής μας

Στο παραπάνω γραφικό περιβάλλον υπάρχουν δεξιά κάποια χρήσιμα πεδία για να μπορέσουμε να συνδέσουμε το wemos με την παρούσα εφαρμογή. Αρχικά βλέπουμε ένα πεδίο, όπου είναι και η πρώτη κίνηση που πρέπει να κάνουμε, η συμπλήρωση της static ip του wemos. Έπειτα πατάμε "Σύνδεση" και περιμένουμε να δούμε αν η ζεύξη είναι πετυχημένη. Αυτό φαίνεαι στο τρίτο πεδίο όπου υπάρχουν δύο επιλογές, η πράσινη τελεία όπου σημαίνει ότι η ζεύξη έγινε επιτυχώς και η κόκκινη τελεία όπου σημαίνει ότι υπάρχει σφάλμα εμφανίζοντας πιο είναι το σφάλμα αυτό. Και τέλος το μεσαίο πεδίο όπου είναι το κουμπί "Εκτέλεση" και το χρησιμοποιούμε για να εκτελέσουμε τα παραδείγματα μας.

Αριστερά υπάρχει η εργαλειοθήκη με δύο καρτέλες, η μια με τις κινήσεις και η άλλη με τις εντολές επανάληψης και τους βρόγχους. Στο κέντρο βλέπουμε το workspase όπου με drag &

drop από την εργαλειοθήκη μπορούμε να προγραμματίσουμε το "ρόμποτ" μας να κινηθεί σε μια αυτοσχέδια πίστα όπου θα έχουμε δημιουργήσει, και έπειτα δεξιά υπάρχει το κουμπί εκτέλεσης για να τρέξουμε τον κώδικα μας. Όπως βλέπουμε υπάρχει ένα μήνυμα δεξιά εάν έχει συνδεθεί το wemos μας με το διαδίκτυο. Όταν είναι συνδεδεμένο μας εμφανίζει πράσινο και όταν για κάποιον λόγω δεν υπάρχει σύνδεση μας εμφανίζει κόκκινο.

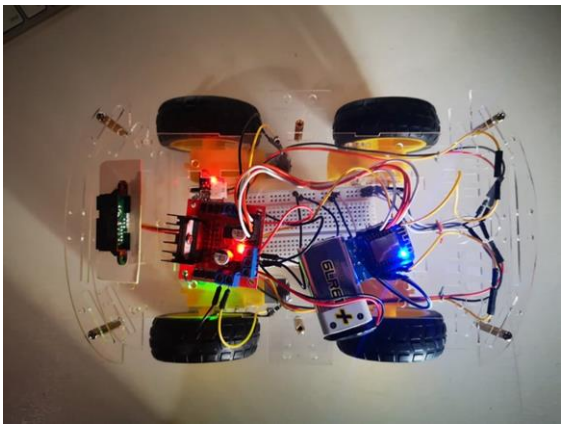


Εικόνα 31: Μενού κίνηση

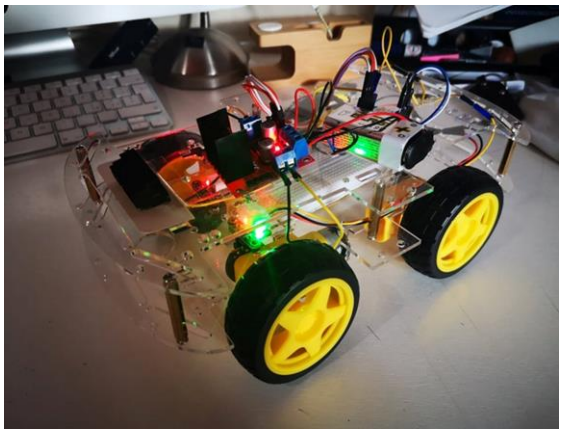
Τα blocks και το γραφικό περιβάλλον έχουν δημιουργηθεί με βάση το blockly. Το γραφικό περιβάλλον, δηλαδή το workspace και η εργαλειοθήκη, είναι ακριβώς τα ίδια όπου μας παρέχει το Blockly αλλά το block, όπως είπαμε παραπάνω, είναι προσαρμοσμένα για το δικό μας πρότζεκτ. Η ανάγκη για την δημιουργία δικών μας blocks προήλθε αρχικά γιατί τα blocks κίνησης δεν υπήρχαν αυτοσχέδια όποτε επειδή πρέπει να τα ρυθμίσουμε με βάση το δικό μας "ρομπότ" με την χρήση του εργαλείου του Blockly, που αναφέραμε στο κεφάλαιο του Blockly ως προγραμματιστικό εργαλείο. Επίσης τα blocks με τις επαναληπτικές εντολές "Εάν, Εάν..τότε και Επανέλαβε μέχρι να βρεις εμπόδιο", χρειάζονται κάποια δεδομένα από τον αισθητήρα απόστασης, οπότε είναι καλύτερο να έχουμε τα δικά μας blocks. Ούτως ή άλλως η ίδια η Blockly προτείνει στην σελίδα της, στην βιβλιογραφία υπάρχει με αριθμό[9], πώς να δημιουργήσουμε τα δικά μας block, οπότε θεωρούμε ότι είναι καλύτερο από να χρησιμοποιήσουμε τα ήδη υπάρχοντα, διότι μπορούμε να τα διαχειριστούμε όπως εμείς πιστεύουμε ότι θα ήταν καλύτερα στο πρότζεκτ μας.

Παρακάτω φαίνεται το όχημα μας πώς είναι στην πραγματικότητα μετά από τον συνδυασμό όλων των παραπάνω όπου αναλύσαμε στην πτυχιακή αυτή. Είναι ένα DIY όχημα με τέσσερις τροχούς, όπου μόνο οι δύο είναι σε λειτουργία, όπως ανέφερα και πιο πάνω. Οι δύο μπροστινοί τροχοί είναι σε λειτουργία και οι άλλοι δύο είναι ανενεργοί, δηλαδή έχουμε αφαιρέσει από το εσωτερικό τους όλα τα εξαρτήματα που περιείχε έτσι ώστε να είναι πιο εύκολο να κυλήσουν χωρίς αντίσταση. Αν θέλαμε να χρησιμοποιήσουμε και τους τέσσερις τροχούς θα έπρεπε να χρησιμοποιήσουμε μια επιπλέον H-bridge. Επιλέξαμε το αυτοσχέδιο όχημα με τους τέσσερις τροχούς αντί για τους δύο, διότι παραμένει πιο σταθερό στις στροφές, και έχει μεγαλύτερο εμβαδόν έτσι ώστε να συνδεθούν όλα τα εξαρτήματα χωρίς να μπερδεύονται. Ένα αντίστοιχο όχημα θα μπορούσε να φτιαχτεί με lego blocks ή να εκτυπωθεί από έναν 3D-printer με το αντίστοιχο σχέδιο.

Στον πρώτο όροφο είναι ένα breadboard όπου συνδέσαμε τις γειώσεις και τις τροφοδοσίες των εξαρτημάτων και στον πρώτο όροφο είναι τα υπόλοιπα εξαρτήματα μας, L289N, Wemos D1 mini και ο αισθητήρας απόστασης. Οι διαστάσεις είναι: 25.8cm x 15cm x 8cm και το βάρος είναι 0.5Kg. Επίσης έχουμε προσθέσει ένα επιπλέον βάρος των 140 γραμμαρίων για να είναι πιο σταθερό στις κινήσεις του.



Εικόνα 32: Το αυτοσχέδιο αυτοκινητάκι μας, όψη 1



Εικόνα 33: Το αυτοσχέδιο αυτοκινητάκι μας, όψη 2

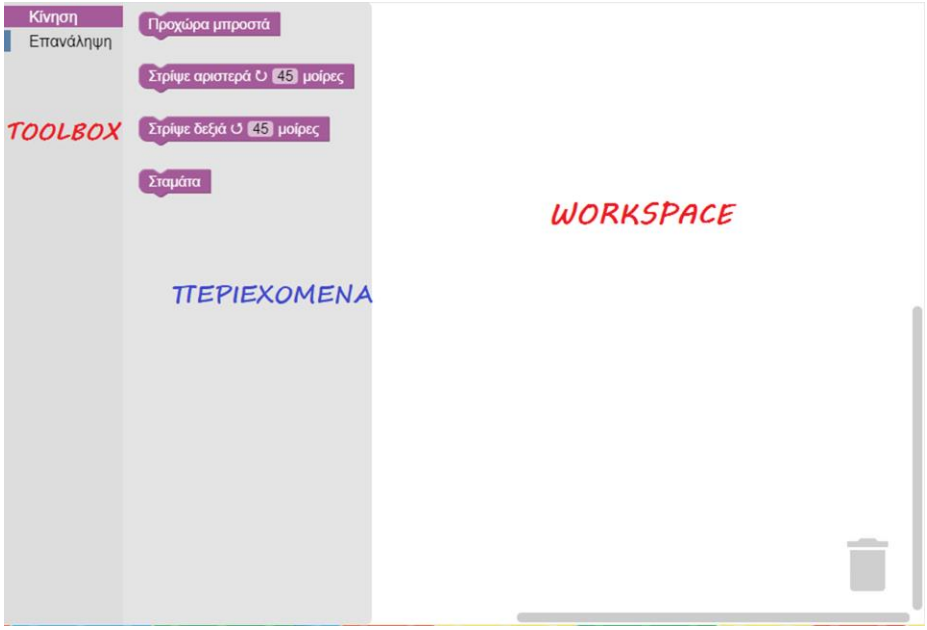
6.2 Δημιουργία και λειτουργικότητα περιβάλλοντος Blockly (Προγραμματιστικό κομμάτι του project)

6.2.1 Περιβάλλον Workspace

Με την χρήση της βιβλιοθήκης Blockly, μπορούμε εύκολα να εισάγουμε εύκολα το πρωτότυπο workspace του Blockly. Όπως φαίνεται και στην παρακάτω εικόνα μέσα στην δήλωση του workspace, εισάγουμε και την εργαλειοθήκη το οποίο μας παρέχετε και αυτό από την βιβλιοθήκη του Blockly.

```
  workspace = Blockly.inject('blockly', {
    toolbox: document.getElementById('toolbox'),
    move: { scrollbars: true, drag: true, wheel: true }
  });
}
```

Εικόνα 34: Δημιουργία του toolbox μέσω blockly



Εικόνα 35: Γραφικό περιβάλλον blockly

6.2.2 Περιβάλλον εργαλειοθήκης

Αφού προσθέσαμε το workspace και μέσα στο workspace εισάγαμε την εργαλειοθήκη, δημιουργήσαμε τα περιεχόμενα του toolbox. Όπως φαίνεται παρακάτω στην είκονα, δημιουργήσαμε το μενού "Κίνηση" και "Επανάληψη" και μέσα σε κάθε κατηγορία δηλώσαμε τα αντίστοιχα block. Παρακάτω φαίνεται ο κώδικας για το πως δημιουργήσαμε τα block και την λειτουργικότητας τους.

```
<div id="blockly" style="height: 90vh; width: 70vw;"></div>

<xml id="toolbox" style="display: none">
  <category name="Κίνηση" colour='310'>
    <block type="move_forward"></block>
    <block type="turn_left"></block>
    <block type="turn_right"></block>
    <block type="stop"></block>
  </category>
  <category name="Επανάληψη" colour="210" expanded="true">
    <block type='if_wall'></block>
    <block type='if_else_wall'></block>
    <block type='for'></block>
    <block type='until'></block>
  </category>
</xml>
```

Εικόνα 36: Δημιουργία του μενού του toolbox στο blockly

6.2.3 Block “Προχώρα μπροστά”

```
/* Instructions to move the car forward */
Blockly.JavaScript['move_forward'] = function(block) {

  var code =
    'console.log("forward");' +
    'rightMotorForward.stop();' +
    'leftMotorForward.stop();' +
    'rightMotorBackward.stop();' +
    'leftMotorBackward.stop();' +
    'rightMotorForward.start(' +
    motorSpeed +
    ');' +
    'leftMotorForward.start(' +
    motorSpeed +
    ');' +
    'sleep(' +
    seconds +
    ');' +
    'rightMotorForward.stop();' +
    'leftMotorForward.stop();' +
    'rightMotorBackward.stop();' +
    'leftMotorBackward.stop();';

  return code;
```

Εικόνα 37: Κώδικας για την δημιουργία της λειτουργικότητας του block “Προχώρα μπροστά”.

```
Blockly.defineBlocksWithJsonArray([
  //Create the customized block 'move_forward'
  {
    type: 'move_forward',
    message0: 'Προχώρα μπροστά',
    previousStatement: null,
    nextStatement: null,
    colour: 310
  },
```

Εικόνα 38: Κώδικας για την δημιουργία του block “Προχώρα μπροστά”.

6.2.4 Block “Στρίψε αριστερά”

```
//Create the customized block 'turn_left'
{
  type: 'turn_left',
  message0: 'Στρίψε αριστερά \u2b6e %1 μοίρες',
  args0: [
    {
      type: 'field_number',
      name: 'DEGREES',
      min: 0,
      max: 360,
      value: 45
    }
  ],
  previousStatement: null,
  nextStatement: null,
  colour: 310
},
```

Εικόνα 39: Κώδικας για την δημιουργία του block "Στρίψε αριστερά".

6.2.5 Block "Στρίψε δεξιά"

```
/* Instructions to turn the car to the left */
Blockly.JavaScript['turn_left'] = function(block) {
  var degree = Number(block.getFieldValue('DEGREES'));
  var seconds = degree / degreeConst;

  var code =
    'console.log("left");' +
    'rightMotorForward.stop();' +
    'leftMotorForward.stop();' +
    'rightMotorBackward.stop();' +
    'leftMotorBackward.stop();' +
    'rightMotorForward.start(' +
    motorSpeed +
    ');' +
    'leftMotorBackward.start(' +
    motorSpeed +
    ');' +
    'sleep(' +
    seconds +
    ');' +
    'rightMotorForward.stop();' +
    'leftMotorForward.stop();' +
    'rightMotorBackward.stop();' +
    'leftMotorBackward.stop();' +
    'sleep(' +
    delay +
    ');';

  return code;
};
```

Εικόνα 40: Κώδικας για την δημιουργία της λειτουργικότητας του block "Στρίψε αριστερά".

```
//Create the customized block 'turn_right'
{
  type: 'turn_right',
  message0: 'Στρίψε δεξιά \u2b6f %1 μοίρες',
  args0: [
    {
      type: 'field_number',
      name: 'DEGREES',
      min: 0,
      max: 360,
      value: 45
    }
  ],
  previousStatement: null,
  nextStatement: null,
  colour: 310
},
```

Εικόνα 41: Κώδικας για την δημιουργία του block "Στρίψε δεξιά".

```
/* Instructions to turn the car to the right */
Blockly.JavaScript['turn_right'] = function(block) {
  var degree = Number(block.getFieldValue('DEGREES'));
  var seconds = degree / degreeConst;

  var code =
    'console.log("right");' +
    'rightMotorForward.stop();' +
    'leftMotorForward.stop();' +
    'rightMotorBackward.stop();' +
    'leftMotorBackward.stop();' +
    'rightMotorBackward.start(' +
    motorSpeed +
    ');' +
    'leftMotorForward.start(' +
    motorSpeed +
    ');' +
    'rightMotorForward.stop();' +
    'leftMotorBackward.stop();' +
    'sleep(' +
    seconds +
    ');' +
    'rightMotorForward.stop();' +
    'leftMotorForward.stop();' +
    'rightMotorBackward.stop();' +
    'leftMotorBackward.stop();' +
    'sleep(' +
    delay +
    ');';

  return code;
};
```

Εικόνα 42: Κώδικας για την δημιουργία της λειτουργικότητας του block "Στρίψε δεξιά".

6.2.6 Block "Σταμάτα"

```
//Create the customized block 'stop'
{
  type: 'stop',
  message0: 'Σταμάτα',
 //Create the customized block 'if'
 {
   type: 'if_wall',
   message0: 'Εάν εμπόδιο %1 τότε κάνε %2',
   args0: [
     {
       type: 'input_dummy'
     },
     {
       type: 'input_statement',
       name: 'wall'
     }
   ],
   previousStatement: null,
   nextStatement: null,
   colour: 210,
   tooltip: '',
   helpUrl: ''
 },
```

Εικόνα 45: Κώδικας για την δημιουργία του block "Εάν εμπόδιο τότε κάνε...".

```
/* Instructions to stop the car */
Blockly.JavaScript['stop'] = function(block) {

  var code =
    'console.log("stop");' +
    'rightMotorForward.stop();' +
    'leftMotorForward.stop();' +
    'rightMotorBackward.stop();' +
    'leftMotorBackward.stop();';

  return code;
};
```

Εικόνα 44: Κώδικας για την δημιουργία της λειτουργικότητας του block "Σταμάτα".

6.2.7 Block "If"

```
/* if object in front conditions code */
Blockly.JavaScript['if_wall'] = function(block) {
  //Statements inside the if
  var statements_wall = Blockly.JavaScript.statementToCode(block, 'wall');

  var code = 'console.log("if"); console.log(proximity.cm); if(proximity.cm <= ' + minimumDistanceFromWall + '){' + statements_wall + '}';

  return code;
};
```

Εικόνα 46: Κώδικας για την λειτουργικότητα της εντολής "Εάν εμπόδιο τότε κάνε...".

6.2.8 Block "If else"

```
//Create the customized block 'if_else'
{
  type: 'if_else_wall',
  message0: 'Εάν εμπόδιο %1 τότε κάνε %2 αλλιώς %3',
  args0: [
    {
      type: 'input_dummy'
    },
    {
      type: 'input_statement',
      name: 'wall'
    },
    {
      type: 'input_statement',
      name: 'no_wall'
    }
  ],
  previousStatement: null,
  nextStatement: null,
  colour: 210,
  tooltip: '',
  helpUrl: ''
}
```

Εικόνα 47: Κώδικας για την δημιουργία του block "Εάν εμπόδιο τότε κάνε...αλλιώς...".

```
/* If_else object in front conditions code */
Blockly.JavaScript['if_else_wall'] = function(block) {
  //Statements inside the do
  var statements_wall = Blockly.JavaScript.statementToCode(block, 'wall');
  //Statements inside the else
  var statements_no_wall = Blockly.JavaScript.statementToCode(block, 'no_wall');

  var code =
    'console.log("if_else");' +
    "console.log(proximity.cm);" +
    'if(proximity.cm <= ' +
    minimumDistanceFromWall +
    '){' +
    statements_wall +
    '} else {' +
    statements_no_wall +
    '}';
```

Εικόνα 48: Κώδικας για την λειτουργικότητα της εντολής "Εάν εμπόδιο τότε κάνε...αλλιώς...".

6.2.9 Block "For"

```
//Create the customized block 'for'
{
  type: 'for',
  message0: 'Για %1 φορές %2 %3',
  args0: [
    {
      type: 'field_number',
      name: 'TIMES',
      value: 1,
      min: 1,
      max: 20
    },
    {
      type: 'input_dummy'
    },
    {
      type: 'input_statement',
      name: 'move_statement'
    }
  ],
  previousStatement: null,
  nextStatement: null,
  colour: 120,
```

Εικόνα 49: Κώδικας για την δημιουργία του block "Για ν φορές κάνε...".

```
/* for n times conditions code */
Blockly.JavaScript['for'] = function(block) {
  //Times to repeat the cycle
  var times = Number(block.getFieldValue('TIMES'));
  //Statements inside the for
  var move_statement = Blockly.JavaScript.statementToCode(block, 'move_statement');

  var code =
  'console.log("for");'+
  'var i = 0;'+
  'for(i=0; i<' +times+ '; i++){'+
    move_statement +
  '}';

  return code;
};
```

Εικόνα 50: Κώδικας για την λειτουργικότητα της εντολής "Για Ν φορές κάνε...".

6.2.10 Block "Until"

```
//Create the customized block 'repeat_until'
{
  type: 'until',
  message0: 'Επανέλαβε μέχρι να βρείς εμπόδιο %1 κάνε %2',
  args0: [
    {
      type: 'input_dummy'
    },
    {
      type: 'input_statement',
      name: 'no_wall'
    }
  ],
  previousStatement: null,
  //nextStatement: null,
  colour: 120,
  tooltip: '',
  helpUrl: ''
}
```

2

Εικόνα 51: Κώδικας για την δημιουργία του block "Επανέλαβε μέχρι να βρεις εμπόδιο".

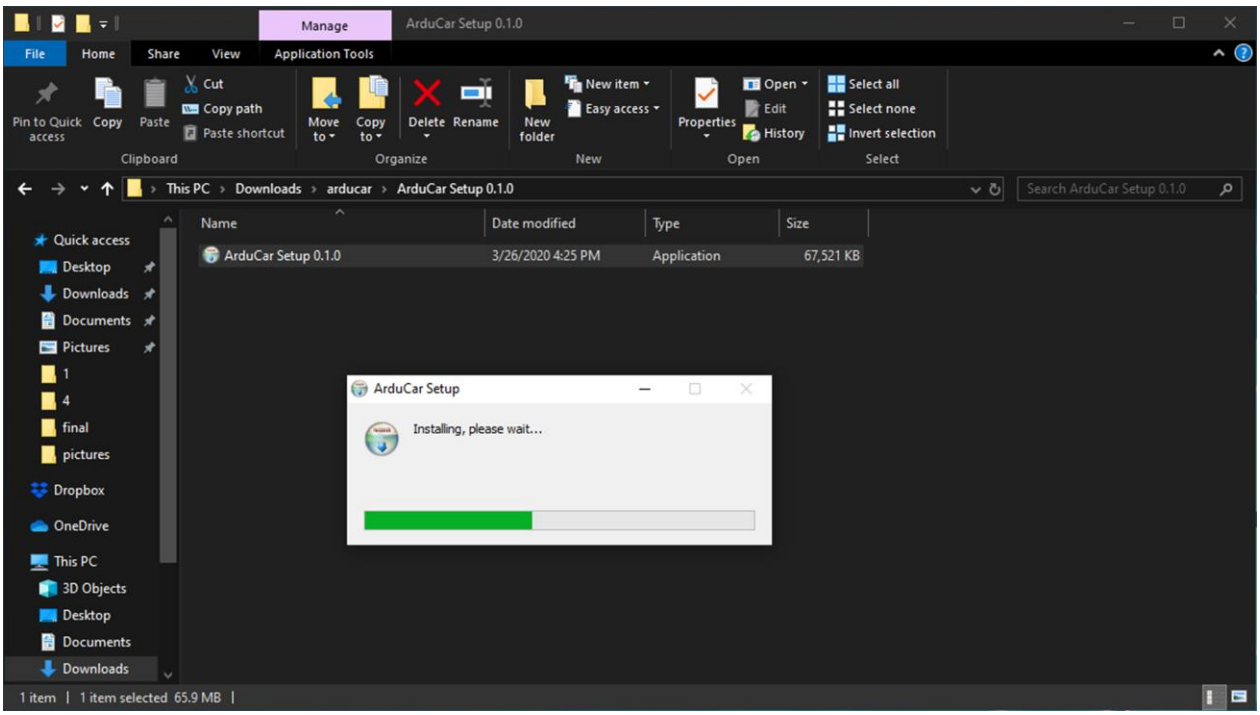
```
/* repeat until find an obstacle  conditions code*/
Blockly.JavaScript['until'] = function(block) {
  //Statements inside the until
  var statements_no_wall = Blockly.JavaScript.statementToCode(block, 'no_wall');

  var code =
    'console.log("until");' +
    "console.log((proximity.cm * 2), 'cm');" +
    'var intervalID = setInterval(function() {' +
    "console.log((proximity.cm * 2), 'cm');" +
    'if ((proximity.cm * 2) <= ' +
    minimumDistanceFromWall +
    ') {' +
    'rightMotorForward.stop();' +
    'rightMotorBackward.stop();'+
    'leftMotorForward.stop();' +
    'leftMotorBackward.stop();'+
    'clearInterval(intervalID);' +
    '} else {' +
    statements_no_wall +
    '}' +
    '},'+(seconds * 1001)+');';

  return code;
};
```

Εικόνα 52: Κώδικας για την λειτουργικότητα της εντολής "Επανέλαβε μέχρι να βρεις εμπόδιο ".

7. ΕΚΤΕΛΕΣΗ ΕΦΑΡΜΟΓΗΣ

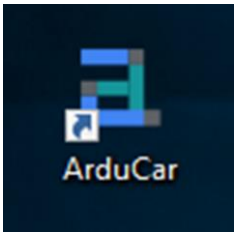
7.1 Εγκατάσταση εφαρμογής

Η εγκατάσταση της εφαρμογής μας είναι πολύ εύκολη. Κάνουμε διπλό κλικ στο .exe αρχείο και είμαστε έτοιμοι.



Εικόνα 53: Εγκατάσταση εφαρμογής

Μετά την εγκατάσταση ανοίγει κατευθείαν την εφαρμογή και δημιουργεί ένα εικονίδιο στην επιφάνεια εργασίας.



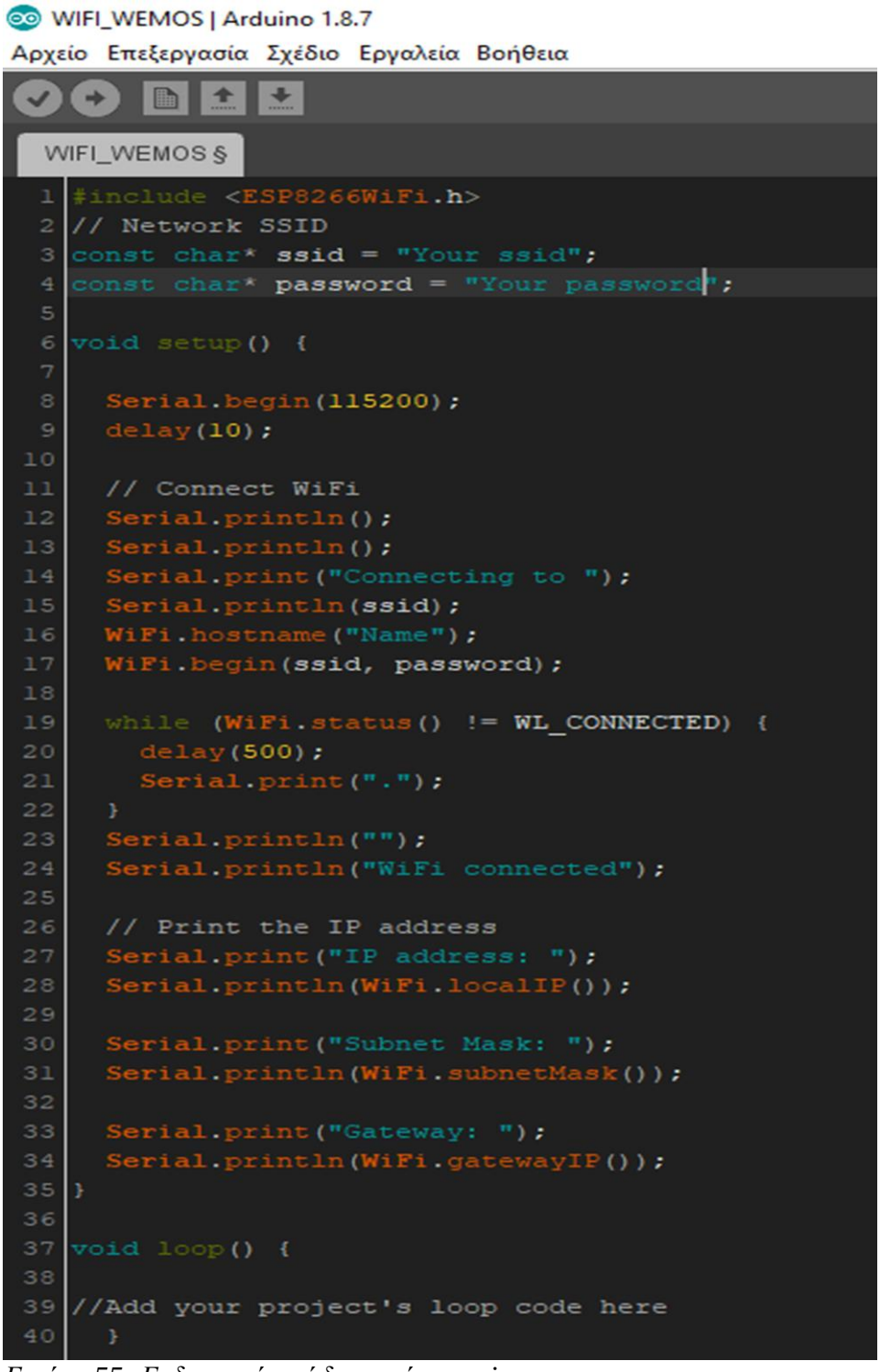
Εικόνα 54: Εικονίδιο εφαρμογής

Έπειτα χρειαζόμαστε κάποια δεδομένα για να μπορεί να γίνει η σύνδεση του wemos με την εφαρμογή μας. Αυτά τα δεδομένα είναι το ssid και password του δικτύου μας, και η static ip και gateway του wemos τα οποία μας παρέχονται από το router μας. Εφόσον έχουμε αυτά τα στοιχεία τα συμπληρώνουμε στο αρχείο της StandardFirmataWifi. Αυτό που χρειαζόμαστε στην παρούσα φάση είναι η εγκατάσταση του arduino IDE, της firmata και του wemos στο Arduino IDE, όπως υποδείξαμε παραπάνω σε προηγούμενο κεφάλαιο. Ανοίγουμε το παράδειγμα της StandardFirmataWifi και εμφανίζονται δυο καρτέλες. Εμείς εισάγουμε τις πληροφορίες μας στο αρχείο με όνομα wifiConfig, όπου υπάρχουν σαφής οδηγίες για το που εισάγουμε τα στοιχεία μας. Έπειτα συνδέουμε το wemos στον υπολογιστή και μέσω του arduino IDE ανεβάζουμε την StandardFirmataWifi στο wemos. Εφόσον αυτό γίνει, εγκαθιστούμε την εφαρμογή μας. Παρακάτω θα δούμε βήμα βήμα πώς γίνεται η εγκατάσταση της εφαρμογής. Η διαδικασία αυτή γίνετε μία φόρα μόνο. Η περίπτωση να χρειαστεί να την ξανακάνουμε είναι εφόσον θελήσουμε να αλλάξουμε δίκτυο, όποτε θα έχουμε και διαφορετική ip.

Αρχικά να δούμε πως βρίσκουμε την static ip του wemos.

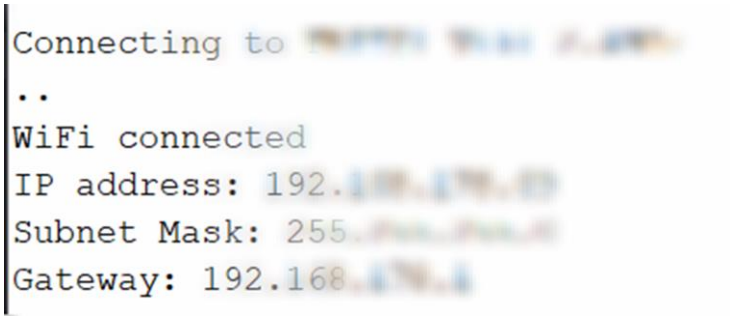
1. Πρώτα συνδέουμε το wemos με τον υπολογιστή μας.
2. Ανοίγουμε το arduino IDE.
3. Δημιουργούμε ένα αρχείο με τον παρακάτω κώδικα.
4. Και τέλος το φορτώνουμε στην σωστή θύρα στην συσκευή μας για να συλλέξουμε τα δεδομένα που θέλουμε.

ΠΡΟΣΟΧΗ! Το upload speed πρέπει να είναι 115200 όπως το έχουμε ορίσει. Για να το επαληθεύσουμε πάμε στα Εργαλεία> Upload speed



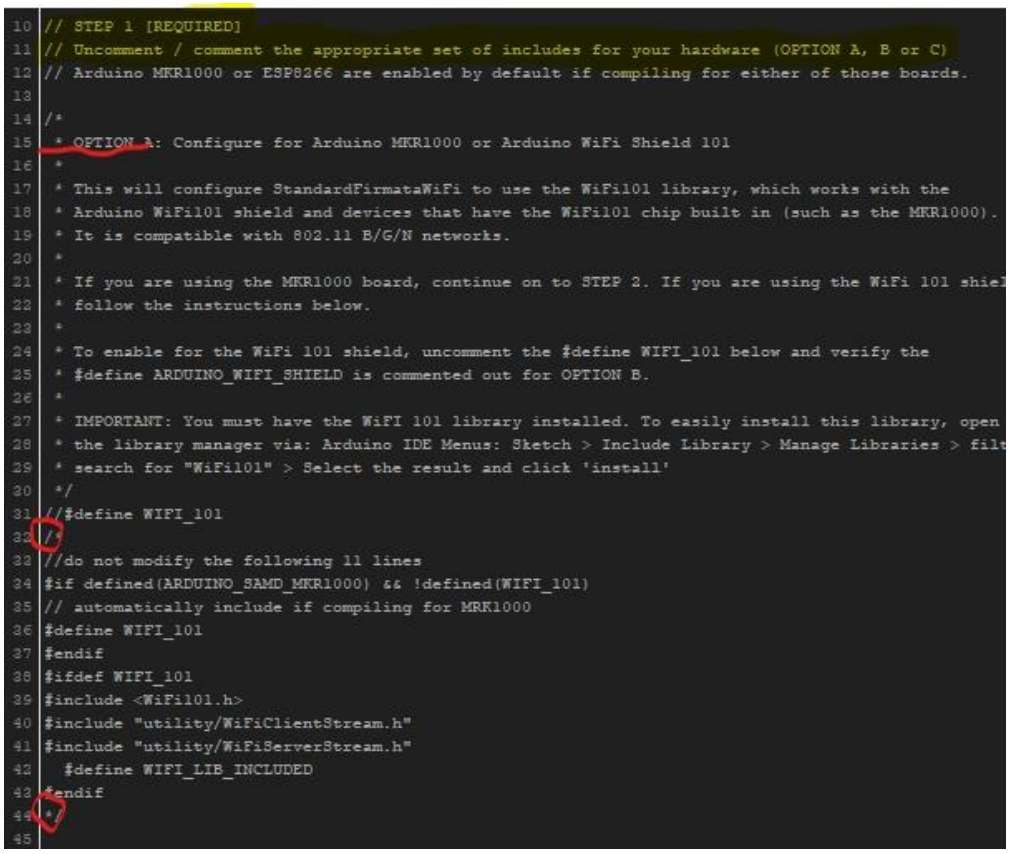
Εικόνα 55: Ενδεικτικός κώδικας εύρεσης ip.

Έπειτα ανοίγουμε Εργαλεία>Παρακολούθηση σειριακής και ανοίγει ένα νέο παράθυρο με τα δεδομένα που ζητήσαμε.



Εικόνα 56: Λήψη ip

Εφόσον έχουμε πλέον την static ip, ανοίγουμε το παράδειγμα της StandardFirmataWifi και ακολουθούμε την ακόλουθη διαδικασία.

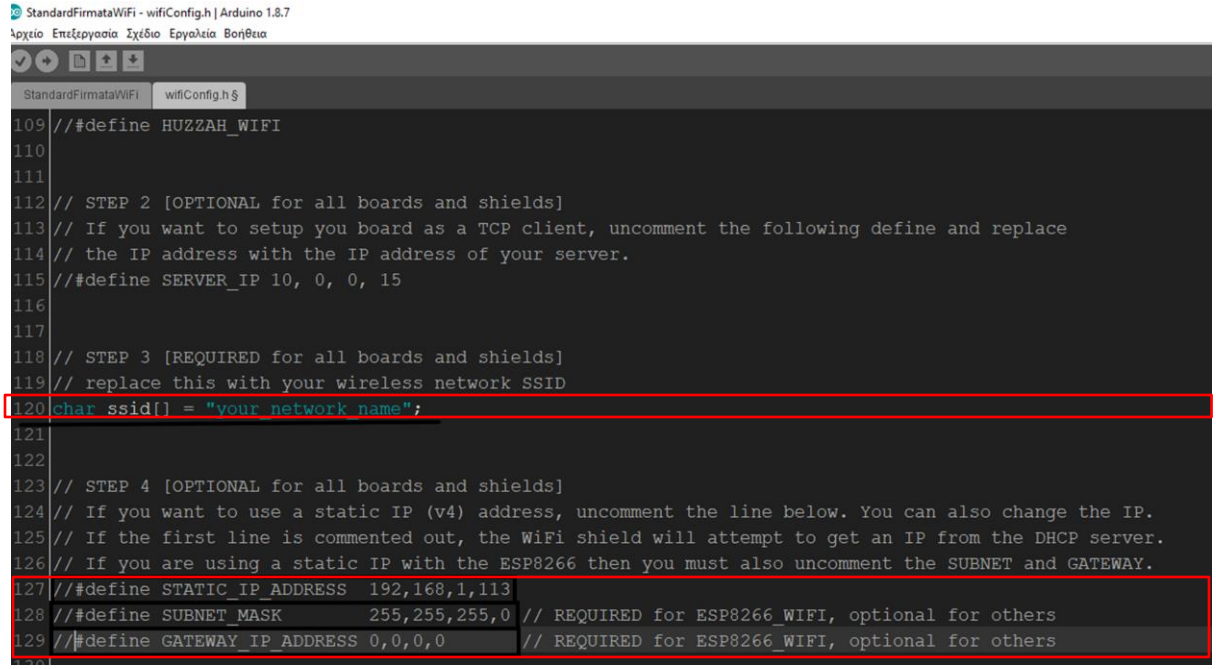


```
 * OPTION B: Configure for legacy Arduino WiFi shield
 *
 * This will configure StandardFirmataWiFi to use the original WiFi library (deprecated) provided
 * with the Arduino IDE. It is supported by the Arduino WiFi shield (a discontinued product) and
 * is compatible with 802.11 B/G networks.
 *
 * To configure StandardFirmataWiFi to use the legacy Arduino WiFi shield
 * leave the #define below uncommented and ensure #define WIFI_101 is commented out for OPTION A.
 */
//#define ARDUINO_WIFI_SHIELD
/*
//do not modify the following 10 lines
#ifdef ARDUINO_WIFI_SHIELD
#include <WiFi.h>
#include "utility/WiFiClientStream.h"
#include "utility/WiFiServerStream.h"
  #ifdef WIFI_LIB_INCLUDED
  #define MULTIPLE_WIFI_LIB_INCLUDES
  #else
  #define WIFI_LIB_INCLUDED
  #endif
#endif
*/
/*
 * OPTION C: Configure for ESP8266
 *
 * This will configure StandardFirmataWiFi to use the ESP8266WiFi library for boards
 * with an ESP8266 chip. It is compatible with 802.11 B/G/N networks.
 *
 * The appropriate libraries are included automatically when compiling for the ESP8266 so
 * continue on to STEP 2.
 *
 * IMPORTANT: You must have the esp8266 board support installed. To easily install this board see
 * the instructions here: https://github.com/esp8266/Arduino#installing-with-boards-manager.
 */
//do not modify the following 14 lines
#ifdef ESP8266
// automatically include if compiling for ESP8266
#define ESP8266_WIFI
#endif
#ifdef ESP8266_WIFI
#include <ESP8266WiFi.h>
#include "utility/WiFiClientStream.h"
#include "utility/WiFiServerStream.h"
  #ifdef WIFI_LIB_INCLUDED
  #define MULTIPLE_WIFI_LIB_INCLUDES
  #else
  #define WIFI_LIB_INCLUDED
  #endif
#endif
```

Εικόνα 57: Βήμα 1ο: Βάζουμε σε σχόλια τα options A & B

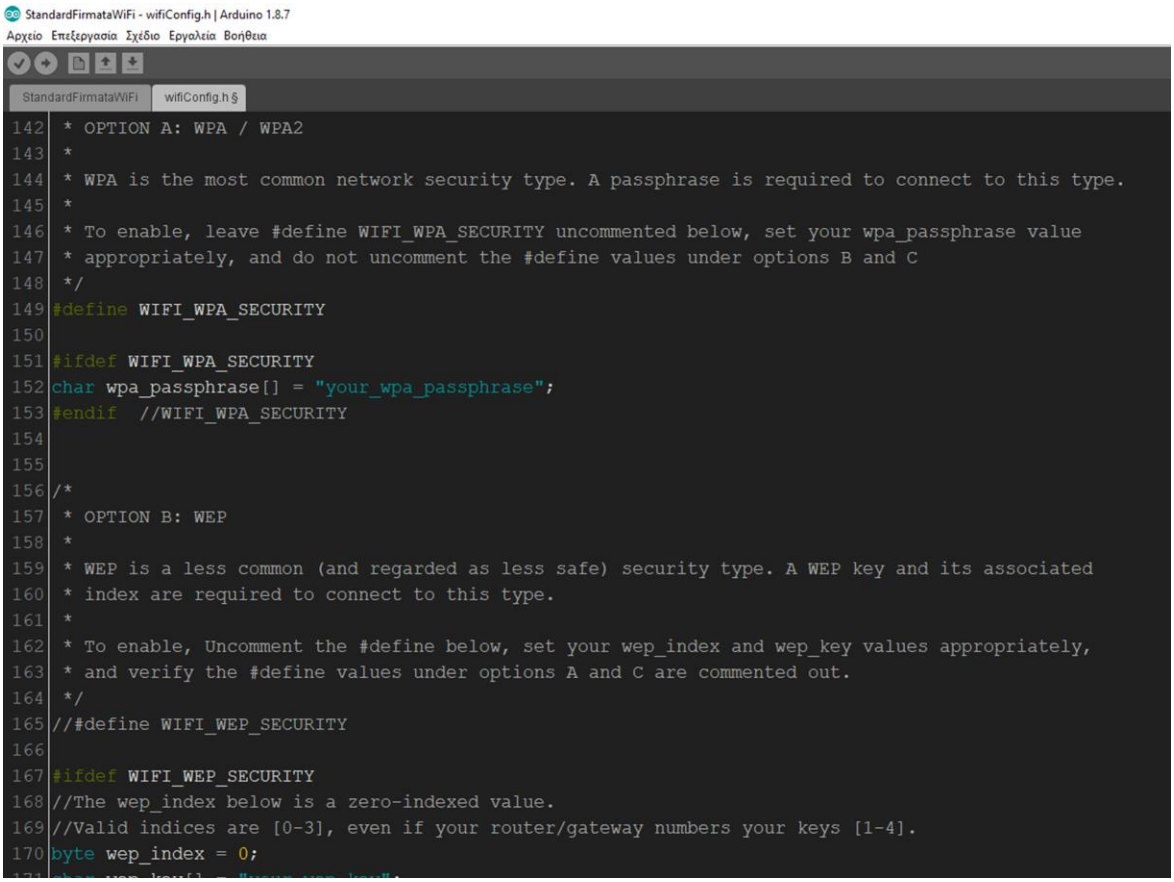
Όπως φαίνεται στην εικόνα 57, μας ζητείται να βάλουμε σε σχόλια τις επιλογές που δεν μας εξυπηρετούν. Εμείς θα χρειαστούμε στο Option C όπου είναι για board με ESP8266 πυρήνα.

Έπειτα εισάγουμε το ssid και την static ip αλλά πρώτα θα πρέπει να αφαιρέσουμε τις τρεις γραμμές από τα σχόλια.



Εικόνα 58: Βήμα 2ο: εισαγωγή ssid και δικτυακών παραμέτρων.

Και τέλος εισάγουμε και τον κωδικό του δικτύου μας.

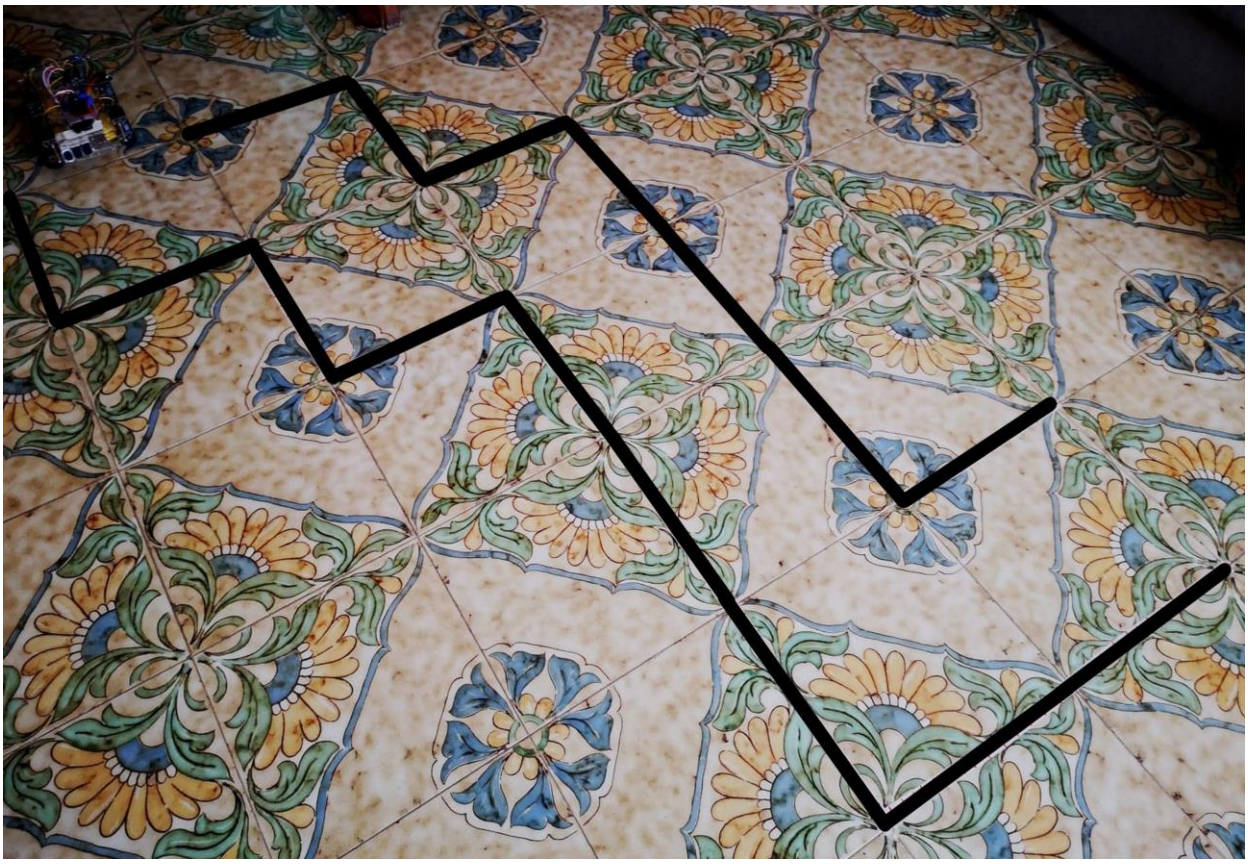


Εικόνα 59: Βήμα 3ο: εισαγωγή του password.

Μετά την εγκατάσταση της εφαρμογής θα χρειαστεί να εισάγουμε στο κατάλληλο πεδίο την static ip του wemos και να πατήσουμε το κουμπί “Σύνδεση”. Εφόσον η ζεύξη είναι επιτυχημένη μπορούμε να τρέξουμε όποιο παράδειγμα θελήσουμε στην εφαρμογή μας. Υπάρχει ένδειξη για το αν η ζεύξη είναι πετυχημένη ή όχι.

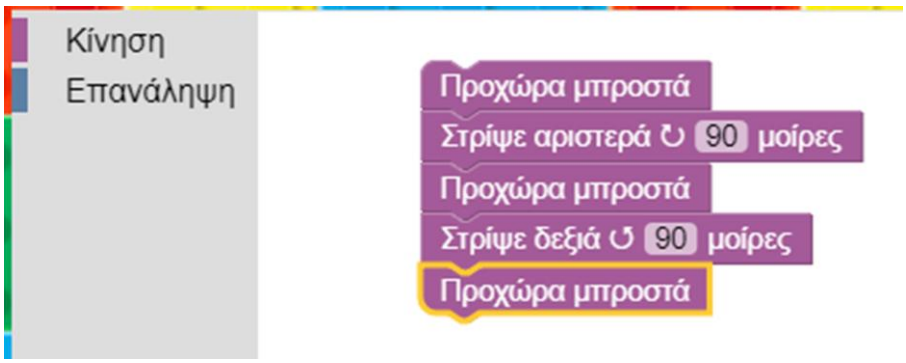
7.2 Παραδείγματα εκτέλεσης εφαρμογής

Παρακάτω φαίνεται μια αυτοσχέδια πίστα που φτιάξαμε για να τρέξουμε τα παραδείγματα μας. Φτιάξαμε τον διάδρομο όπου θα κινείται το αυτοκίνητο με χαρτοταινία και για εμπόδιο βάλαμε ένα μαξιλάρι. Όπως αναφέραμε παραπάνω η πίστα μπορεί να δημιουργηθεί με πολλούς και διάφορους τρόπους και αναλόγως με την πίστα χρησιμοποιούμε και τις εντολές για να τρέξουμε και το αυτοκίνητο μας.



Εικόνα 60: Αυτοσχέδια πίστα

1ο παράδειγμα:



Εικόνα 61: 1ο παράδειγμα

Εκτέλεση πρώτου παραδείγματος βήμα προς βήμα:

Εικόνα 63: 1ο παράδειγμα: Αρχική θέση αυτοκινήτου

Εικόνα 64: 1ο παράδειγμα-βήμα 1: Προχώρα μπροστά

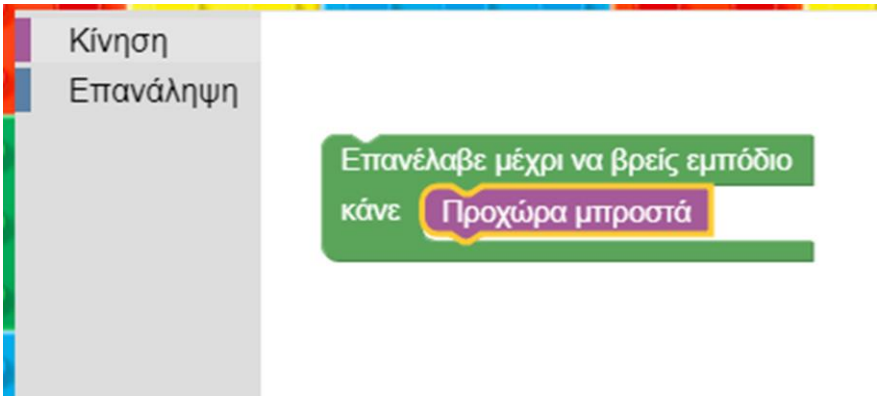
Εικόνα 65: 1ο παράδειγμα-βήμα 2: Στρίψε αριστερά

Εικόνα 66: 1ο παράδειγμα-βήμα 3: Προχώρα μπροστά

Εικόνα 67: 1ο παράδειγμα-βήμα 4: Στρίψε δεξιά

Εικόνα 68: 1ο παράδειγμα-βήμα 5: Προχώρα μπροστά

2ο παράδειγμα



Εικόνα 69: 2ο παράδειγμα

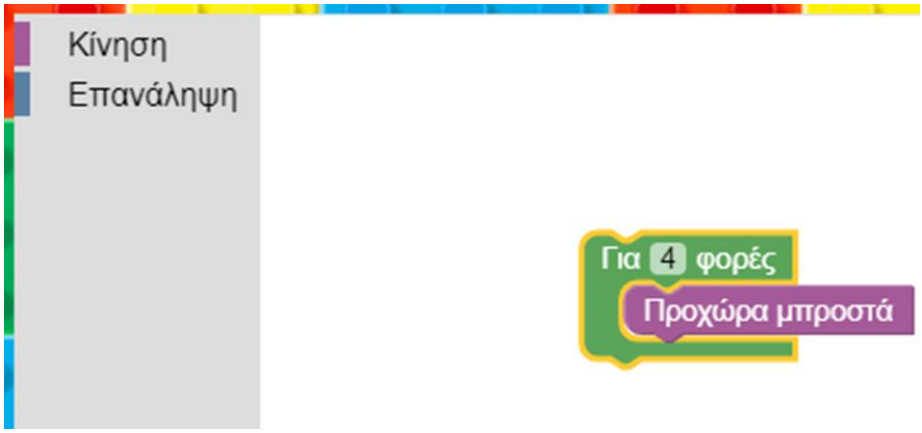
Εκτέλεση δεύτερου παραδείγματος βήμα προς βήμα:

Εικόνα 70: 2ο παράδειγμα-βήμα 1

Εικόνα 71: 2ο παράδειγμα-βήμα 2

Εικόνα 72: 2ο παράδειγμα-βήμα 3

3ο παράδειγμα:



Εικόνα 73: 3ο παράδειγμα

Εκτέλεση τρίτου παραδείγματος βήμα προς βήμα:

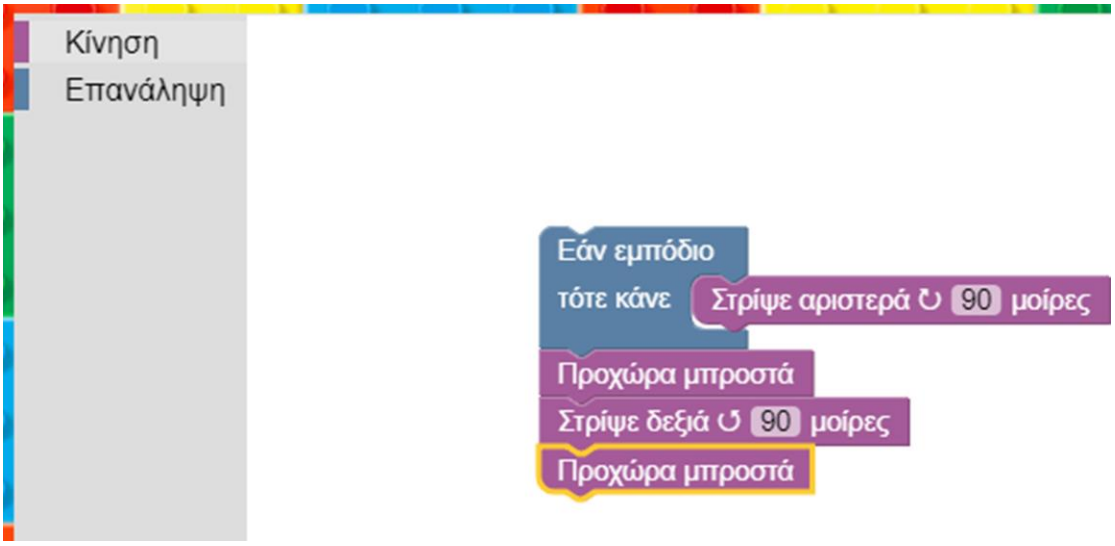
Εικόνα 74: 3ο παράδειγμα-βήμα 1

Εικόνα 75: 3ο παράδειγμα-βήμα 2

Εικόνα 76: 3ο παράδειγμα-βήμα 3

Εικόνα 77: 3ο παράδειγμα-βήμα 4

4ο παράδειγμα:



Εικόνα 78: 4ο παράδειγμα

Εκτέλεση τέταρτου παραδείγματος βήμα προς βήμα:

Εικόνα 79: 4ο παράδειγμα-βήμα 1

Εικόνα 80: 4ο παράδειγμα-βήμα 2

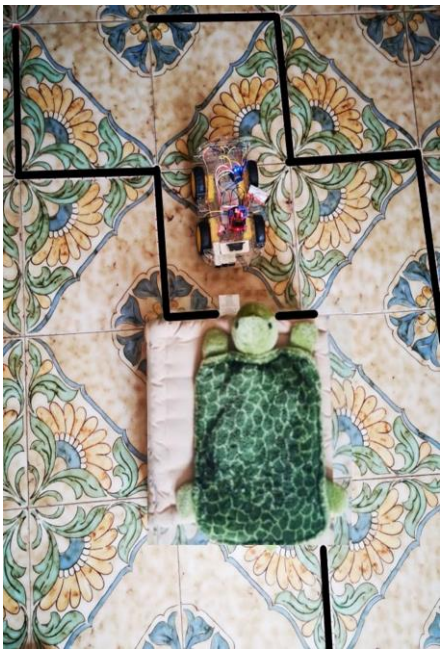
Εικόνα 81: 4ο παράδειγμα-βήμα 3

Εικόνα 82: 4ο παράδειγμα-βήμα 4

5ο παράδειγμα:

Εικόνα 83: 5ο παράδειγμα

Εκτέλεση πέμπτου παραδείγματος βήμα προς βήμα:



Εικόνα 84: 5ο παράδειγμα-βήμα 1: Αρχική θέση

Εικόνα 85: 5ο παράδειγμα-βήμα 2: Στρίψε αριστερά

Εικόνα 86: 5ο παράδειγμα-βήμα 3: Προχώρα μπροστά

8. ΣΥΜΠΕΡΑΣΜΑΤΑ

Μετά από μελέτη για την κατασκευή εκπαιδευτικού αυτοματισμού, στην περίπτωση μας αυτοκίνητο, με δυνατότητα προγραμματισμού μέσω γραφική διεπαφής και μελέτη του εκπαιδευτικού πλαισίου STEM και πως αυτό εφαρμόζεται σε μαθητές δημοτικού καταλήξαμε στο παρών αποτέλεσμα. Σκοπός μας είναι να παρέχουμε στους μαθητές ένα εργαλείο, το οποίο είναι εύκολα κατανοητό σε μικρές ηλικίες, και πληρεί τις προϋποθέσεις του προγράμματος STEM. Έτσι και εμείς δημιουργήσαμε αυτήν την εφαρμογή έτσι ώστε χρησιμοποιώντας την να μπορούν οι μαθητές να κατανοήσουν τις βασικές έννοιες του προγραμματισμού, τις επαναλήψεις. Με την βοήθεια του blockly καταφέραμε να δημιουργήσουμε ένα γραφικό περιβάλλον όπου από την εργαλειοθήκη μπορείτε να κάνετε “drag&drop” τα blocks, τα οποία είναι κατασκευασμένα ειδικά για ο δικό μας πρότζεκτ, στην επιφάνεια εργασίας της εφαρμογής. Έτσι , με τον σωστό συνδυασμό block, αναλόγως την κάθε περίπτωση, το αυτόματο όχημα μετακινείτε σε μια αυτοσχέδια πίστα, για να φτάσει στον προορισμό του αποφεύγοντας τα τυχόν εμπόδια που μπορεί να βρίσκονται μπροστά του. Αρχικά ξεκινάμε με απλά παραδείγματα έτσι ώστε να κατανοήσουμε την εφαρμογή και την λειτουργία των blocks και έπειτα κάνουμε πιο σύνθετα παραδείγματα. Έτσι οι μαθητές βήμα-βήμα κατανοούν και εφαρμόζουν τις απλές έννοιες του προγραμματισμού.

Πίσω, όμως, από το γραφικό κομμάτι υπάρχει και ο κώδικας που γράψαμε για να προγραμματιστεί σωστά η όλη εφαρμογή. Και εκτός το προγραμματιστικό κομμάτι υπάρχει βεβαίως και το κομμάτι του υλικού. Συνδυάζοντας τα σωστά components και σωστό προγραμματισμό ώστε να μας δώσουν ένα άρτιο αποτέλεσμα, χρειάστηκε μελέτη για την επιλογή των σωστών components, όπου θα υπάρχει συμβατότητα μεταξύ τους, όπως και η συμβατότητα με την firmata. Έπειτα από μελέτη καταλήξαμε στα components όπου σας παρουσιάσαμε παραπάνω τα οποία είναι ο μικροελεκτής wemos d1 mini, οι dc motors, η h-bridge L289N και ο αισθητήρας απόστασης της sharp GP2Y0A41SK0F. Μετά από έρευνα καταλήξαμε ότι η πιο χρήσιμη γλώσσα προγραμματισμού για εμάς είναι η JavaScript διότι χρησιμοποιήσαμε την πλατφόρμα Johnny-five, όπου είναι σε js, γιατί πρώτον το Johnny-five έχει συμβατότητα με

firmata και blockly, και δεύτερον το wemos είναι μια πλακέτα συμβατή με το Johnny-five. Με αυτά τα κριτήρια επιλέχθηκαν όλα τα εξαρτήματα και όλες οι πλατφόρμες που χρησιμοποιήσαμε.

Κατά την διάρκεια της εκτέλεσης της παρούσας πτυχιακής υπήρξαν αρκετές δυσκολίες, όπου οι περισσότερες λύθηκαν επιτυχώς. Κάποιες από τις αδυναμίες που παραμένουν είναι η εισαγωγή της ip και η τροφοδότηση του οχήματος. Πιο συγκεκριμένα για το πρώτο, για να συνδεθεί το όχημα με την εφαρμογή χρειάζεται το SSID και την ip του δικτύου όπου είναι συνδεδεμένο το wemos, διότι η ζεύξη γίνεται ασύρματα μέσω wi-fi. Το πρόβλημα στην παρούσα φάση είναι ότι την ip και το SSID τα χρησιμοποιούμε στην firmataStandardWifi, στο αρχείο που πρέπει να ανεβάσουμε στο wemos, αλλά δεν γίνεται αυτόματα,δηλαδή μέσω της εφαρμογής μας, και πρέπει να την προσθέσουμε εμείς χειροκίνητα κάθε φορά που αλλάζουμε δίκτυο. Το δεύτερο πρόβλημα που παραμένει είναι με την τροφοδοσία, όπου ενδεχομένως δεν είναι αρκετή ώστε να τροφοδοτεί σωστά όλο το κύκλωμα. Εμείς χρησιμοποιούμε μια μπαταρία 9Volts, όπου αρχικά φαίνεται να λειτουργεί τέλεια αλλά μετά από κάποια χρήση του οχήματος, η μπαταρία εξαντλείτε πιο γρήγορα από το συνηθισμένο.

Εν κατακλείδι, το όχημα λειτουργεί σωστά όταν είναι συνδεδεμένο με το δίκτυο. Αλλαγές ή προσθήκες που θα μπορούσαν να γίνουν, εκτός από την επίλυση των ελαττωμάτων που προαναφέραμε, θα ήταν η χρήση κάποιων επιπλέον αισθητήρων τύπου photo interrupter για την πιο ακριβή μέτρηση κατά την κίνηση του “Προχώρα εμπρός” και “Στρίψε δεξιά/αριστερά”. Εμείς για να εκτελέσουμε αυτές τις εντολές, υπολογίσαμε σε δευτερόλεπτα την απόσταση που πρέπει να κάνει για να προχωρήσει μπροστά όπως και τις μοίρες που θα πρέπει να στρίψει το όχημα μας, αν και παρόλα αυτά εάν το δάπεδο μας είναι ομαλό οι μετρήσεις που κάναμε είναι κατά 98% σωστές. Με τους photo interrupter θα είναι πιο ακριβής με την έννοια ότι θα μετράμε με πιο ακριβή τρόπο την απόσταση που θα πρέπει να διανύσει το όχημα, μετρώντας σε δυαδικό σύστημα διότι ο συγκεκριμένος αισθητήρας διαβάζει από ένα κυκλικό αντικείμενο που έχει τρύπες και είναι συνδεδεμένος με την ρόδα. Οι επιλογές εισαγωγής είναι δυο, είτε διαβάζει το φως που διαπερνά από το κενό είτε όχι, δηλαδή λειτουργεί σαν διακόπτης.

9. ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Documentation of Wemos D1 mini.
https://docs.zerynth.com/latest/official/board.zerynth.wemos_d1_mini/docs/index.html#pin-mapping
(Πρόσβαση: 25-09-2019)

[2] Data-sheet of sharp GP2Y0A41SK0F.
https://global.sharp/products/device/lineup/selection/opto/haca/diagram.html
(Πρόσβαση: 25-09-2019)

[3] DC motors.
https://en.wikipedia.org/wiki/DC_motor
(Πρόσβαση: 25-09-2019)

[4] L289N.
https://mytectutor.com/dc-motor-control-using-l298n-motor-driver/
(Πρόσβαση: 27-09-2019)

[5] Βιβλιοθήκη firmata.
https://www.arduino.cc/en/Reference/Firmata
(Πρόσβαση: 27-02-2020)

[6] Εγκατάσταση StandardFirmata Wifi σε Arduino IDE.
https://github.com/firmata/arduino
(Πρόσβαση: 27-02-2020)

[7] Firmata Builder.
http://firmatabuilder.com/
(Πρόσβαση: 27-02-2020)

[8] Create-react-app.
https://github.com/facebook/create-react-app
(Πρόσβαση: 27-02-2020)

[9] Δημιουργία περιβάλλοντος Blockly.
https://developers.google.com/blockly/guides/configure/web/fixed-size
(Πρόσβαση: 27-02-2020)

[10] Δημιουργία customized block από το Blockly.
https://blockly-demo.appspot.com/static/demos/blockfactory/index.html
(Πρόσβαση: 27-02-2020)

[11] Πλατφόρμα Johnny-five και παραδείγματα.
http://johnny-five.io/
(Πρόσβαση: 28-02-2020)

[12] Τι είναι η STEM.
https://www.livescience.com/43296-what-is-stem-education.html
(Πρόσβαση: 28-02-2020)

[13] Δραστηριότητες STEM στην Ελλάδα.
https://stem.edu.gr/
(Πρόσβαση: 28-02-2020)

[14] STEM στο νηπιαγωγείο.
https://stem.edu.gr/coding/kids-first-coding-robotics/
(Πρόσβαση: 28-02-2020)

[15] Εκπαιδευτική ρομποτική.
https://el.wikipedia.org/wiki/%CE%95%CE%BA%CF%80%CE%B1%CE%B9%CE%B4%CE%B5%CF%85%CF%84%CE%B9%CE%BA%CE%AE_%CF%81%CE%BF%CE%BC%CF%80%CE%BF%CF%84%CE%B9%CE%BA%CE%AE
(Πρόσβαση: 28-02-2020)

[16] Οπτική γλώσσα προγραμματισμού.
https://el.wikipedia.org/wiki/%CE%9F%CF%80%CF%84%CE%B9%CE%BA%CE%AE_%CE%B3%CE%BB%CF%8E%CF%83%CF%83%CE%B1_%CF%80%CF%81%CE%BF%CE%B3%CF%81%CE%B1%CE%BC%CE%BC%CE%B1%CF%84%CE%B9%CF%83%CE%BC%CE%BF%CF%8D
(Πρόσβαση: 28-02-2020)

[17] Τι είναι η React και παραδείγματα.
https://reactjs.org/
(Πρόσβαση: 29-02-2020)

[18]Τι είναι η Electron.
https://www.electronjs.org/
(Πρόσβαση: 29-02-2020)