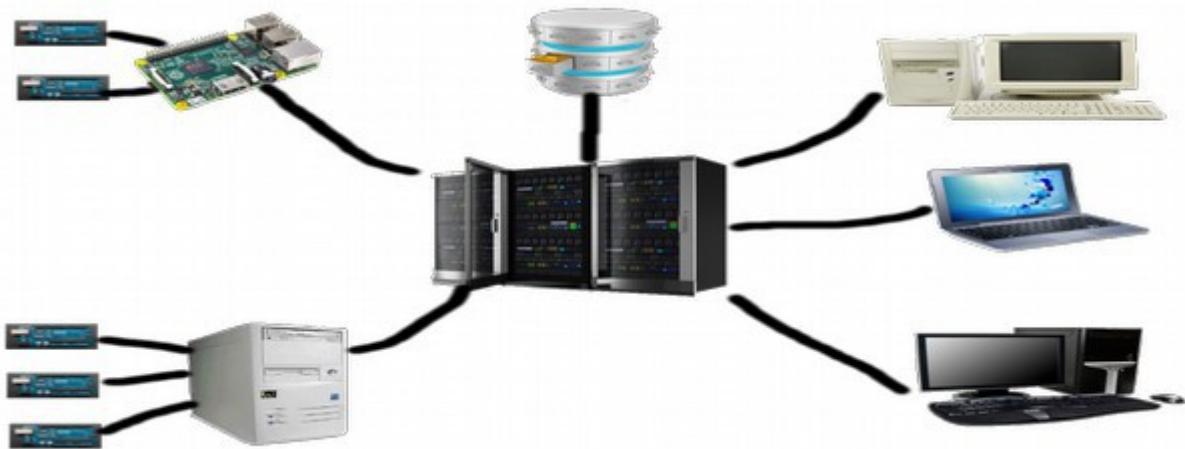




ΤΕΙ Αθήνας
Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Μηχανικών Πληροφορικής ΤΕ
Αθήνα Απρίλιος 2016

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ



με θέμα :

Μελέτη ενός ολοκληρωμένου συστήματος για τη διαχείριση μικρο-ελεγκτών απομακρυσμένα σε πραγματικό χρόνο από εγγεγραμμένους χρήστες (πχ για τον έλεγχο "έξυπνων" σπιτιών).

Πιλοτικά θα υλοποιηθεί client-server εφαρμογή που θα επιτρέπει την διαχείριση διάφορων μονάδων (Αισθητήρες, φώτα, διακόπτες) κάποιων μικρο-ελεγκτών(Arduino).

Εμφαση θα δοθεί στη χρήση τεχνολογιών Java, MySQL,Arduino sketch,php,Javascript,css,html.

**Γαλλιάκης Μιχαήλ
Α.Μ : 081001**

Επιβλέποντες καθηγητές :
**Σκουρλάς Χρήστος
Τσολακίδης Αναστάσιος**

Email : cs081001@teiath.gr & michaelgalliakis@yahoo.gr

Όλα τα αρχεία του συστήματος βρίσκονται στο : <https://github.com/michaelgalliakis/myThesis.git>



Πρόλογος

Αρχικά να διευκρινιστεί ότι το θέμα της πτυχιακής έχει να κάνει με τη μελέτη και την ερεύνα που χρειάζεται να γίνει για να μπορέσει να υλοποιηθεί και να λειτουργήσει στη πράξη ένα “ολοκληρωμένο σύστημα διαχείρισης μικρο-ελεγκτών σε πραγματικό χρόνο από απομακρυσμένους εγγεγραμμένους χρήστες”. Η έμφαση με άλλα λόγια, έχει δοθεί κυρίως στο καθαρά τεχνικό κομμάτι που αφορά την ανάλυση, σχεδίαση και κυρίως ανάπτυξη εμπράκτως του συστήματος και όχι σε θεωρητικό επίπεδο . Επομένως, η εργασία αυτή πιθανόν να διαφέρει κάπως από το παραδοσιακό τρόπο συγγραφής πτυχιακών άλλων σχολών ή ειδικοτήτων . Διότι ο σκοπός ήταν να αξιοποιηθούν ιδίως οι γνώσεις και οι δεξιότητες που αποκτά ένας φοιτητής πληροφορικής του ΤΕΙ Αθήνας στην διάρκεια φοίτησης του στη σχολή. Παρόλο όμως που η πτυχιακή έχει μια διαφορετική προσέγγιση στη δομή της (δεν παρουσιάζει για παράδειγμα μια θεωρητική επιστημονική αναζήτηση) και αποτελείτε ουσιαστικά από μια εφαρμογή (σύστημα) και τη τεκμηρίωση της, δεν έχει λείψει η έρευνα διότι το ίδιο το σύστημα που έχει αναπτυχθεί είναι αποτέλεσμα έρευνας (όπως και κάποια σημεία της τεκμηρίωσης). Επίσης, ήταν απαραίτητη η αναζήτηση και η μελέτη βιβλίων και άλλων πηγών (μέσω διαδικτύου κατά κύριο λόγο) για την περάτωση αυτής της εργασίας.

Όσον αφορά αυτό εδώ το κείμενο , έχει σκοπό να τεκμηριώσει όσο καλύτερα γίνεται , με όσο πιο απλά και λίγα λόγια, την υλοποίηση του εν λόγω συστήματος. Αφενός για να είναι πιο εύκολο να τα καταλάβει κάποιος που δεν γνωρίζει πολλά πάνω στο θέμα και αφετέρου για να είναι γενικότερα όσο το δυνατόν λιγότερο κουραστική η ανάγνωση από αυτόν που το διαβάζει .

Οι λόγοι επιλογής του συγκεκριμένου θέματος ήταν διάφοροι.

Ένας από τους σημαντικότερους λόγους ήταν το γεγονός ότι είναι πολύ ενδιαφέρον και εντυπωσιακό να μπορείς να έχεις την “αίσθηση” ενός χώρου και να τον “διαχειρίζεσαι” απομακρυσμένα και πρακτικά να μπορείς να αλληλεπιδράσεις μαζί του από μακριά .Δηλαδή ακούγεται πολύ ωραίο, να μπορείς να δέχεσαι “ερεθίσματα” από το πραγματικό κόσμο ,να βλέπεις τις καταστάσεις διάφορων συσκευών αλλά και να μπορείς να τις αλλάζεις (πχ να ανοίγεις ή να κλείνεις μια λάμπα) και όλα αυτά από κάποιο άλλο πιθανόν μακρινό μέρος του κόσμου .

Ακόμη είναι πολύ “όμορφο” να βλέπεις να “παντρεύονται” διαφορετικές τεχνολογίες με σκοπό να παραχθεί ένα ενιαίο αποτέλεσμα , όπως για παράδειγμα να συνδυάζονται οι υπολογιστές με τα δίκτυα και το hardware με το software. Με άλλα λόγια και πιο συνοπτικά λοιπόν, είναι πολύ ενδιαφέρον να δει κάποιος στη πράξη και να υλοποιήσει κάτι , που έχει να κάνει με αυτό που λέγεται κοινώς σήμερα **internet of things**.

Ένας άλλος λόγος επιλογής αυτού του θέματος πτυχιακής ήταν επειδή συνδυάζει πολλές τεχνολογίες της πληροφορικής. Κάτι που είναι ιδανικό επειδή υπάρχει ποικιλία και δίνεται η ευκαιρία να ασχοληθεί κάποιος φοιτητής με τόσο όμορφα και συνάμα διαφορετικά “κομμάτια” της επιστήμης των υπολογιστών.



Ευχαριστίες

Αρχικά θα ήθελα να ευχαριστήσω με όλη μου τη καρδιά τη μητέρα μου Μαρία μαζί με όλη την οικογένεια μου ,τους συγγενείς μου αλλά και γενικά όλους τους ανθρώπους που στήριξαν τις σπουδές μου με όποιο τρόπο μπορούσε ο κάθε ένας .

Ακόμη θα ήθελα να ευχαριστήσω όλους τους καθηγητές ,δασκάλους, συμμαθητές , συμφοιτητές και συναδέλφους μου που βοήθησαν και συντέλεσαν για την καλύτερη δυνατή μόρφωση μου .Και βέβαια ιδιαίτερα τον κ. Σκουρλά και τον κ. Τσολακίδη που εκτός από καθηγητές μου στη σχολή επέβλεπαν και αυτήν εδώ την πτυχιακή εργασία.

Πάνω από όλα όμως αισθάνομαι την ευγνωμοσύνη και θα ήθελα να ευχαριστήσω ολόψυχα το Θεό και πιο συγκεκριμένα το Χριστό, την Παναγία, τον Αρχάγγελο Μιχαήλ , την Αγία Μαρίνα και τον Άγιο Ραφαήλ που χάρη σε αυτούς βρίσκομαι ακόμη στη ζωή και που επιπλέον με βοήθησαν σε όλη τη διάρκεια των σπουδών μου και συνέβαλαν στην πραγμάτωση αυτής της εργασίας .

Τέλος, δεν θα ήθελα να λησμονήσω να ευχαριστήσω τη Πολιτεία και το ΤΕΙ Αθήνας που μου έδωσαν την ευκαιρία και τη δυνατότητα να γίνω φοιτητής στο τμήμα πληροφορικής .



Περιεχόμενα

| | |
|-------------------------------------------------------------------------------|----|
| Πρόλογος..... | 2 |
| Ευχαριστίες..... | 3 |
| Περίληψη..... | 5 |
| Εισαγωγή..... | 6 |
| Κάποιες βασικές έννοιες..... | 6 |
| Μονάδες..... | 6 |
| Arduino (Μικρό – ελεγκτής)..... | 7 |
| Υπολογιστής..... | 9 |
| Socket..... | 9 |
| Σενάρια χρήσης Πτυχιακής..... | 10 |
| Βασικά κομμάτια του συστήματος..... | 12 |
| Παρουσίαση συστήματος..... | 13 |
| Website..... | 13 |
| Βάση δεδομένων (Database)..... | 21 |
| Device Client..... | 22 |
| Simulator για το Device Client..... | 23 |
| Arduino..... | 26 |
| Server..... | 27 |
| User Client..... | 28 |
| Μεθοδολογία υλοποίησης του συστήματος..... | 34 |
| Ανάλυση, σχεδίαση και τρόπος σκέψης..... | 34 |
| Βάση δεδομένων(Database)..... | 38 |
| Device Client (Simulator)..... | 41 |
| Device Client..... | 43 |
| Server..... | 45 |
| UserClient..... | 47 |
| Ιστοσελίδα (Website)..... | 49 |
| Arduino..... | 51 |
| Εισαγωγή..... | 51 |
| Λίγα πρακτικά εισαγωγικά για το Arduino (Λαμπάκι που αναβοσβήνει)..... | 52 |
| Διάβασμα και αποστολή από και προς τον υπολογιστή..... | 54 |
| Διαχείριση ηλεκτρικών συσκευών από υπολογιστή(Άνοιγμα-κλείσιμο πορτατίφ)..... | 55 |
| Αναλογικό θερμόμετρο..... | 56 |
| Αισθητήρας Φωτιάς..... | 57 |
| Αισθητήρας υγρασίας και θερμοκρασίας..... | 58 |
| Αισθητήρας απόστασης (Sonar)..... | 59 |
| Φωτοαισθητήρας..... | 60 |
| Αισθητήρας κίνησης..... | 61 |
| Αισθητήρας φωνής (Μικρόφωνο)..... | 62 |
| Ηχειάκι (Buzzer)..... | 63 |
| Κουμπάκι (Button)..... | 64 |
| Άλλες μονάδες και εξαρτήματα..... | 65 |
| Πρότυπο sketch για επικοινωνία με το Device Client..... | 66 |
| Μια περίπτωση πραγματικής χρήσης του συστήματος..... | 71 |
| Πιθανές μελλοντικές επεκτάσεις..... | 79 |
| Συμπεράσματα..... | 80 |
| Βιβλιογραφία..... | 81 |
| Παράρτημα (οδηγίες για το “στήσιμο” του συστήματος)..... | 83 |



Περίληψη

Στην αρχή αυτής εδώ της τεκμηρίωσης παρουσιάζονται κάποιες βασικές έννοιες(όπως τι είναι το arduino & το socket και τι θεωρούνται μονάδες & υπολογιστής) που είναι απαραίτητες για να μπορέσει κάποιος να κατανοήσει την συνέχεια. Ακόμη στην εισαγωγική ενότητα αναφέρονται κάποια σενάρια χρήσης όπως επίσης και τα κομμάτια που αποτελούν το σύστημα όπως αυτά είχαν καθοριστεί σαν προδιαγραφές. Σκοπός είναι να μπορέσει κάποιος που διαβάζει αυτό το κείμενο να καταλάβει τον ακριβή στόχο και δηλαδή το τι ακριβώς πρέπει να πετύχει το σύστημα της εργασίας .

Στην επόμενη ενότητα παρουσιάζεται το υλοποιημένο σύστημα με όλες τις λεπτομέρειες που αφορούν την λειτουργία όλων των εφαρμογών του, που σαν σύνολο το κάνουν να υφίσταται. Ουσιαστικά δηλαδή δείχνονται κάποιες περιπτώσεις χρήσης από τα 7 υλοποιημένα μέρη του συστήματος (Website,Database,DeviceClient,Simulator,Arduino,Server και UserClient). Κανονικά ίσως θα έπρεπε να αναφερθεί πρώτα πως υλοποιήθηκε το σύστημα και στην συνέχεια να παρουσιαστούν τα αποτελέσματα , αλλά για λόγους καλύτερης κατανόησης από τη πλευρά του αναγνώστη και πιο ομαλής ροής της τεκμηρίωσης , προτιμήθηκε να υπάρξει η ανάποδη σειρά .

Η τρίτη ενότητα αφορά την μεθοδολογία υλοποίησης του όλου συστήματος . Δηλαδή πως έγινε η ανάλυση & η σχεδίαση και ποιος ήταν ο τρόπος σκέψης. Επίσης, εμφανίζονται κάποια πιο τεχνικά χαρακτηριστικά και μερικές πιο συγκεκριμένες τεχνολογίες που χρησιμοποιήθηκαν μέσα από τη γλώσσα προγραμματισμού Java . Στην συνέχεια αναφέρονται κάποιες γενικές περιγραφές αλλά και κάποια τεχνικά μέρη που αφορούν την υλοποίηση του κάθε κομματιού του συστήματος (και γίνονται οι ανάλογες παραπομπές στο Github για περισσότερες τεχνικές λεπτομέρειες). Στο κομμάτι όμως που αφορά το arduino γίνεται μια εξαίρεση και έτσι πραγματοποιείται μια πιο εκτενή αναφορά με αρκετά παραδείγματα και μεγαλύτερη ανάλυση.

Προς το τέλος δείχνεται μια περύπτωση πραγματικής χρήσης του υλοποιημένου συστήματος όπως επίσης και διάφορες πιθανές μελλοντικές επεκτάσεις που θα μπορούσαν να γίνουν σε αυτό. Αμέσως μετά παρουσιάζονται τα συμπεράσματα που βγήκαν από όλο το εγχείρημα που ολοκληρώθηκε και γίνεται αναφορά στην βιβλιογραφία που χρησιμοποιήθηκε κατά τη διάρκεια ανάπτυξης του συστήματος της εργασίας. Στο τέλος υπάρχει ένα παράρτημα που έχει χρήσιμες οδηγίες για κάποιον που θέλει να “στήσει” το σύστημα μόνος του .



Εισαγωγή

Κάποιες βασικές έννοιες

Μονάδες

➤ **Αισθητήρες :**

Κάποια μικρά συνήθως εξαρτήματα που κάνουν διάφορες μετρήσεις παίρνοντας τιμές από το πραγματικό κόσμο.

Πχ

Θερμόμετρο

Αισθητήρας υγρασίας

Αισθητήρας καπνού



Αισθητήρας απόστασης

Αισθητήρας φωτιάς



➤ **Διακόπτης :**

Μπορεί να χρησιμοποιηθεί, για να απομονώσει μέρος ενός κυκλώματος και έχει 2 καταστάσεις .
Ανοικτό και κλειστό .

Πχ

Μπορεί να είναι ένα κουμπί για να ανοίξει – κλείσει μια λάμπα :



Η ακόμη έχει την έννοια του διακόπτη αυτό που μπαίνει πχ στα παράθυρα για να γνωρίζουμε αν κάποιο έχει παραβιαστεί .



➤ **Λάμπα (ή Led) :**

Μπορεί να έχει 2 καταστάσεις (on-off) ή να παίρνει διακριτές τιμές (πχ 0 - 255) για την φωτεινότητα.





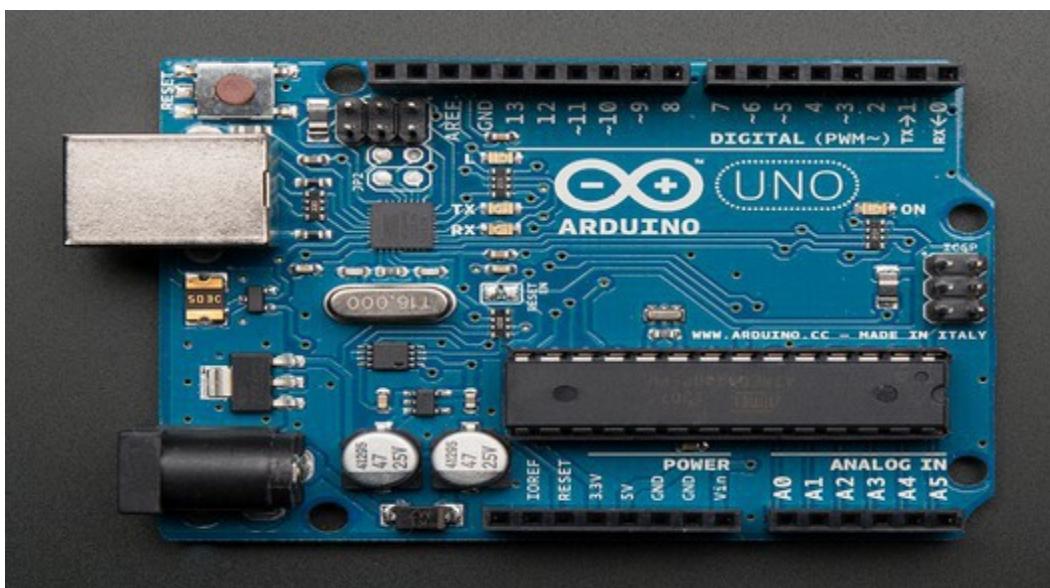
Arduino (Μικρο – ελεγκτής)

- ✓ Στο σημείο αυτό θα γίνει μια αναφορά με απλά λόγια(και όχι με επίσημους όρους) για το τι είναι πρακτικά ένα Arduino – μικροελεγκτής.

Σαν εμφάνιση είναι ένα κομμάτι υλικού (πλακετίτσα) που μπορούν να συνδεθούν πάνω του διάφοροι αισθητήρες, φωτάκια, ρελέ, διακόπτες (και άλλα πιθανόν) με σκοπό να μας κάνει κάτι που θέλουμε με όλα αυτά. Χρησιμοποιούνται βέβαια και μικρά καλώδια, αντιστάσεις και άλλα πιθανόν , για την δημιουργία συνδεσμολογίας όλων αυτών μεταξύ τους.

Ένα arduino μπορεί να προγραμματιστεί και να δουλεύει αυτόνομα και όχι σε συνεργασία με ένα υπολογιστή . Για παράδειγμα, γίνεται σε ένα arduino να συνδέσουμε ένα led και ένα αισθητήρα φωτεινότητας και σε συνάρτηση με τη φωτεινότητα του χώρου να ανάβει ανάλογα και το led (μια έννοια και για το έξυπνο σπίτι). Η ακόμη μπορεί να χρησιμοποιηθεί ένα θερμόμετρο και μια μικρή LCD οθόνη ώστε να βλέπουμε τη θερμοκρασία ενός δωματίου .

Βέβαια, και στις 2 περιπτώσεις πρέπει να έχει προγραμματίσει το arduino με το κατάλληλο sketch(πρόγραμμα) και να παίρνει τροφοδοσία ρεύματος είτε από μια μπαταρία είτε από ένα τροφοδοτικό είτε ακόμη και με την βοήθεια ενός USB καλωδίου (όχι για μεταφορά δεδομένων αλλά μόνο για ρεύμα).



Μια απλή έκδοση του , όπως είναι το arduino Uno , έχει πάνω του :

- Μια θύρα USB για σύνδεση με υπολογιστή(Για προγραμματισμό αλλά και επικοινωνία)
- Μια θύρα για τροφοδοσία 9V .
- Κουμπί για Reset .
- Pins για να συνδέονται διάφορα *εξαρτήματα (όπως αυτά που αναφέρθηκαν πιο πάνω)
 - Τα 6 είναι αναλογικά (από A0 μέχρι το A5).
 - Τα 14 είναι ψηφιακά (από το 0 μέχρι το 13)
 - Τα υπόλοιπα είναι : 3 Gnd(γείωση), 1 5V, 1 3.3V, 1 Reset , 1 Vin , 1 AREF και 1 IOREF .
- Pins (Αρσενικά) για να προγραμματιστεί ο ελεγκτής χωρίς την βοήθεια του USB .
- Τον μικροελεγκτή .



- Η γλώσσα που χρησιμοποιείτε για να προγραμματιστεί το arduino βασίζεται στη γλώσσα Wiring, μια παραλλαγή της C/C++ (για μικροελεγκτές αρχιτεκτονικής AVR όπως ο Atmega) .

Βέβαια, παρόλο που είναι μια **αντικειμενοστρεφής γλώσσα** και θεωρητικά μπορούν να γραφτούν μεγάλα και πολύπλοκα προγράμματα(Sketch) δεν συνιστάται γιατί αφενός είναι **αδύναμη** η επεξεργαστική ισχύ του μικρο – ελεγκτή και αφετέρου η φύση των προβλημάτων που καλούνται να λύσουν οι μικρο-ελεγκτές είναι απλή.

Επίσης, υπάρχουν περιορισμένες **δυνατότητες** και όσον αφορά τη **μνήμη** (που έχει μέσα του ο μικροελεγκτής) και ακόμη δεν υπάρχει η έννοια των threads (τρέχει **μόνο μια ροή** ενός προγράμματος).

Και έτσι ιδιαίτερα πρέπει να γράφονται μικρά sketch και όσο γίνεται πιο απλά .

- Στο περιβάλλον ανάπτυξης ενός arduino sketch (Arduino IDE) υπάρχει η δυνατότητα να γίνει debugging εμφανίζοντας τιμές στην οθόνη . Ουσιαστικά αυτό που συμβαίνει είναι να **στέλνει συμβολοσειρές** το sketch που εκτελείτε σε ένα arduino ,μέσω της θύρας USB σε κάποιον υπολογιστή. Και πρακτικά το Arduino IDE περιβάλλον αναλαμβάνει να διαβάζει τα bytes από το σειριακό καλώδιο και να τα εμφανίζει στην οθόνη .
Παράλληλα με το γράψιμο γίνεται να **διαβάζει** το sketch και **συμβολοσειρές** που έρχονται από την άλλη άκρη του καλωδίου USB πχ από έναν υπολογιστή.
Και στις δυο περιπτώσεις πρέπει να έχει καθορίσει ένα bit rate πχ το 9600 bps .

Ο **ελεγκτής** είναι ένας τύπος επεξεργαστή που λειτουργεί αυτόνομα και έχει μικρό μέγεθος. Ουσιαστικά αυτό που πρέπει να ξέρει κάποιος για να καταλάβει σε πρακτικό επίπεδο (χωρίς μεγάλη ανάλυση) πως δουλεύει (από τη πλευρά ενός προγραμματιστή) είναι ότι ,αν προγραμματιστεί κατάλληλα βλέπει το ρεύμα-σήμα που έχουν τα pin εισόδου του και τα μετατρέπει σε τιμές-αριθμούς . Και αντίστοιχα καθορίζει με βάση κάποιες τιμές-αριθμούς το ρεύμα-σήμα που υπάρχει στα pin εξόδου του .

Και με βάση αυτά που έχουν συνδεθεί πάνω του(αισθητήρες,led ,οθόνες κλπ)&το πρόγραμμα που “τρέχει” ο ελεγκτής, αναλαμβάνει να κάνει κάποια “δουλειά” (όπως στα δυο πιο πάνω παραδείγματα).Μπορεί δηλαδή να προγραμματιστεί κανονικά (με μεταβλητές , πράξεις κλπ) και να υλοποιηθεί στη πράξη κάποιος αλγόριθμος . Επίσης , να αναφερθεί ότι κάθε ελεγκτής έχει αποθηκευμένο και εκτελεί **ένα μόνο sketch**, το οποίο καθορίζει όλη τη λειτουργία του.

- Με απλά λόγια αυτά που πρέπει να κάνει το arduino είναι δυο, στα πλαίσια της πτυχιακής.
 - ✓ Το ένα είναι ανά κάποια χρονική στιγμή (πχ <1 δευτερόλεπτο), να διαβάζει τις τιμές-μετρήσεις από τους αισθητήρες(ανάλογα την περίπτωση πρέπει να γίνουν και κάποιες ενέργειες για να είναι πραγματικές οι τιμές) αλλά και γενικά να “βλέπει” τις καταστάσεις όλων των μονάδων. Και σε επόμενη φάση , εφόσον υπάρξουν κάποιες αλλαγές στις καταστάσεις των μονάδων, αναλαμβάνει να τις στέλνει με μορφή συμβολοσειράς στη θύρα usb για να τα παίρνει ένας υπολογιστής σε επόμενο χρόνο και να τα αξιοποιεί κατάλληλα .
 - ✓ Το δεύτερο πράγμα που πρέπει να κάνει είναι, να βρίσκεται διαρκώς σε θέση να δέχεται μηνύματα μέσω του USB καλωδίου ώστε να μπορούν να αλλάζουν μέσα από έναν υπολογιστή κάποιες τιμές του,ήτοι οι καταστάσεις των μονάδων(πχ φώτα ,διακόπτες κα)



Υπολογιστής

Στην περίπτωση της πτυχιακής εργασίας έχει την έννοια ενός ολοκληρωμένου συστήματος που έχει επεξεργαστή, κύρια και βοηθητική μνήμη, θύρες USB, κάρτα δικτύου και λειτουργικό σύστημα. Θα μπορούσε να είναι ένα παλαιό (ή και καινούριο) pc, ένα laptop ή ένας υπολογιστής “τσέπης” όπως ένα raspberry pi.

- ✓ Το Raspberry pi έχει δυνατότητες και χαρακτηριστικά όπως ενός smartphone. Ακόμη να αναφερθεί ότι γίνεται να συνδεθεί με οθόνη και μέσω USB να έχει πληκτρολόγιο, ποντίκι κα .

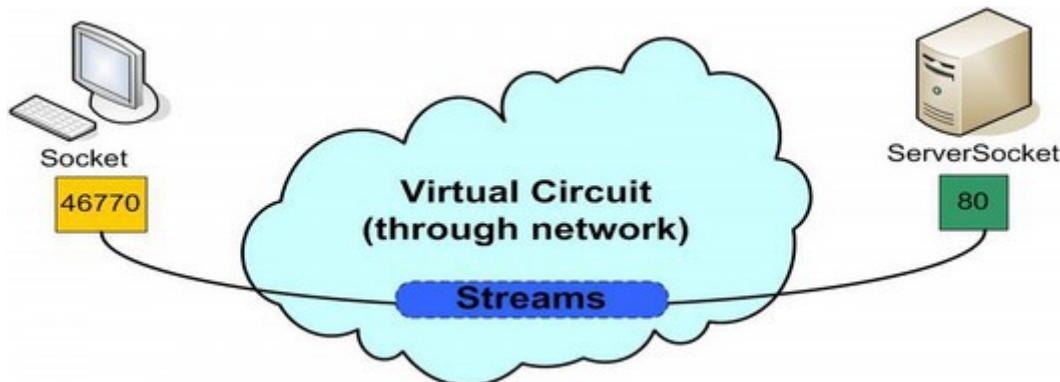


Ο υπολογιστής χρειάζεται για να έχει πάνω του ένα ή περισσότερα arduino (μέσω USB) και για να μπορεί να πραγματοποιηθεί (μέσω της κάρτας δικτύου του) η σύνδεση με έναν άλλον απομακρυσμένο υπολογιστή. Ωστε ουσιαστικά να μπορούν να εποπτεύονται από μακριά οι μονάδες των διαφόρων arduino που είναι συνδεδεμένα “πάνω” του.

Socket

Με λίγα λόγια (και όχι επίσημος ορισμός), είναι μια τεχνολογία που επιτρέπει να συνδεθούν 2 “σημεία” μεταξύ τους (πχ 2 υπολογιστές-προγράμματα) και να επιτευχθεί αμφίδρομη επικοινωνία σε πραγματικό χρόνο μέσα σε δίκτυο ...

Δηλαδή ουσιαστικά υπάρχει η δυνατότητα, με χρήση socket να στηθεί ένα κανάλι μεταξύ 2 προγραμμάτων (που μπορεί το κάθε ένα να τρέχει σε οποιοδήποτε υπολογιστή αρκεί να “βλέπει” το ένα το δίκτυο του άλλου) ώστε να μεταφέρονται bytes με διάφορες πληροφορίες εκατέρωθεν . Στην περίπτωση της εργασίας αυτής θα μεταφέρονται συμβολοσειρές αμφοτέρωθεν . Μια κοινώς γνωστή χρήση των socket είναι στα προγράμματα τύπου chat.





Σενάρια χρήσης Πτυχιακής

Ένα σενάριο Χρήσης (με λίγα λόγια) :

1. Κάποιος δυνητικός χρήστης θα επισκέπτεται την ιστοσελίδα και θα φτιάχνει ένα λογαριασμό χρήστη.
2. Στη συνέχεια θα συνδέετε στην ιστοσελίδα με τα στοιχεία από το λογαριασμό του (που μόλις έφτιαξε) και θα δημιουργήσει ένα λογαριασμό *συσκευής¹(Ένα σύνολο *μονάδων²).
3. Θα κατεβάζει από την ιστοσελίδα ένα template arduino sketch και θα το προσαρμόζει κατάλληλα με βάση τις δικές του *μονάδες που θα έχει (Θα υπάρχουν αναλυτικές οδηγίες βοήθειας).
4. Θα κατεβάζει το Device Client από την ιστοσελίδα και θα το τρέξει στον υπολογιστή (PC, raspberry pi κα) που θα είναι πάνω του συνδεδεμένοι οι μικροελεγκτές-Arduino (σειριακά μέσω usb θύρας) και βέβαια που θα έχει πρόσβαση στο internet . Ο Client θα setáρεται κατάλληλα με τα στοιχεία του λογαριασμού *συσκευής που έφτιαξε ο χρήστης από την ιστοσελίδα.
5. Ο εγγεγραμμένος χρήστης θα μπορεί από οπουδήποτε (στον κόσμο) να κατεβάσει το User Client από τη ιστοσελίδα και με βάση τα στοιχεία του να συνδεθεί στο Server μας (που θα είναι συγχρόνως συνδεδεμένος και ο Device Client) και θα ελέγχει σε πραγματικό χρόνο ανάλογα την περίπτωση την *συσκευή που θα έχει διάφορους arduino πάνω...
6. Προαιρετικά θα μπορεί ο χρήστης μέσω της ιστοσελίδας να προσθέτει (είτε μετά από αίτηση είτε από μόνος ...) άλλους εγγεγραμμένους χρήστες και να τους δίνει δικαιώματα για να βλέπουν και αυτοί την *συσκευή και κατά περίπτωση(ανάλογα τα δικαιώματα) να αλλάζουν και τιμές.

Σημειώσεις – Διευκρινίσεις Σεναρίου :

- Όλα τα παραπάνω θα μπορούν να γίνουν από πολλούς χρήστες και για πολλές *συσκευές .
- Ο κάθε χρήστης θα μπορεί να κάνει αυτά τα βήματα πολλές φορές (εκτός από την εγγραφή του) και κατά επέκταση να “βλέπει” πολλές *συσκευές .
- Επίσης, Θα μπορούσε να μην φτιάξει δικιά του *συσκευή απλά να μπορεί να “βλέπει” *συσκευές από άλλους χρήστες (αν θα έχει πάρει δικαιώματα από τους Ιδιοκτήτες- Διαχειριστές των *συσκευών).

Σημείωση – Διευκρίνιση

- ◆ Θα μπορούν να βλέπουν μια *συσκευή πολλοί χρήστες πχ σε μια οικογένεια ενός σπιτιού κάθε μέλος θα έχει το δικό του λογαριασμό . Επίσης, για κάποιο λόγο θα μπορούσε ανάλογα την περίπτωση να μην έχουν όλοι οι χρήστες δικαίωμα να “επηρεάζουν” την *συσκευή(να αλλάζουν καταστάσεις των μονάδων) αλλά μόνο να την εποπτεύουν . Εκτός από “έξυπνα” σπίτια μπορούν να υπάρχουν και γενικά “έξυπνοι” χώροι .

Εξηγήσεις όρων (*):

1Συσκευή = Ένα σύνολο μονάδων (Με ένα ή παραπάνω μικροελεγκτές).

2Μονάδα = Αισθητήρες, φώτα, διακόπτες κλπ .

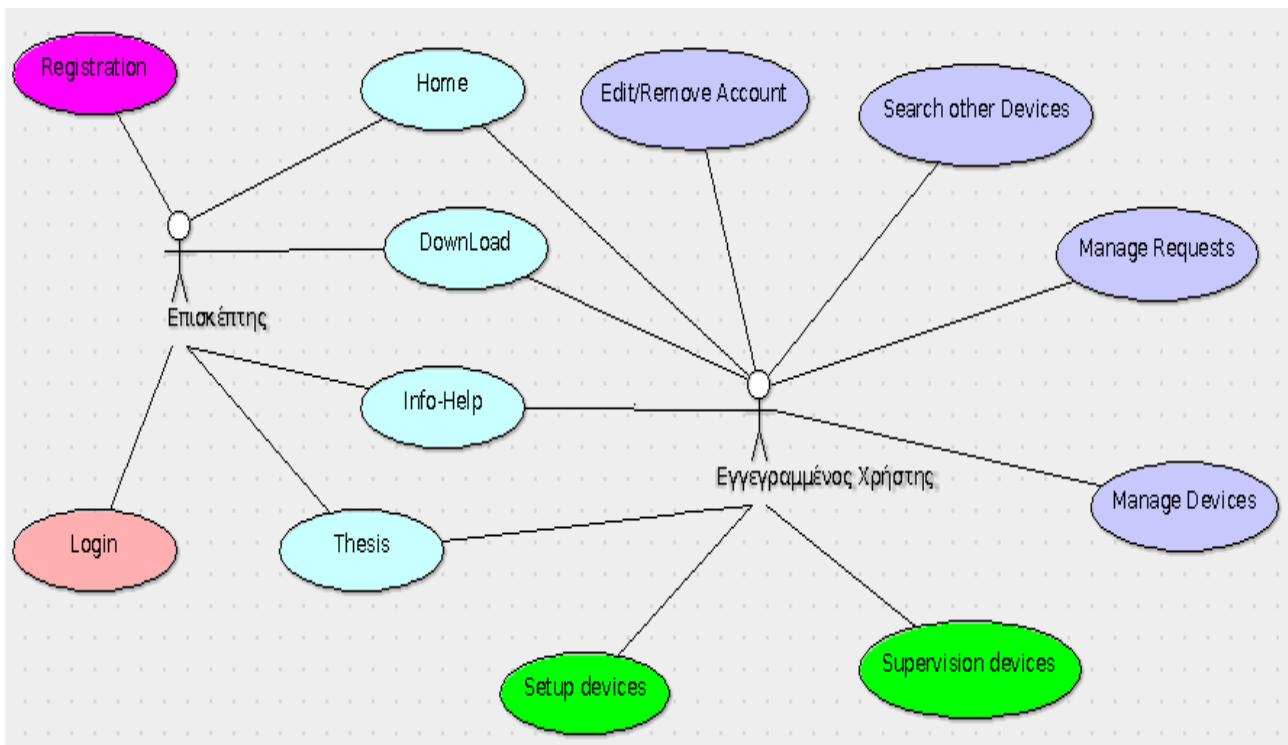


Ένα ακόμη (πιο πρακτικό) σενάριο Χρήστης : (με μαύρα γράμματα αυτά που θα υλοποιηθούν)

- ◆ Για παράδειγμα , έχουμε 1 arduino με 2 λάμπες και ένα θερμόμετρο.
1. Γράφουμε ένα sketch(**Arduino Sketch**) που να διαβάζει και να στέλνει συνέχεια τις τιμές του θερμόμετρου αλλά και τις καταστάσεις των λαμπών μέσω usb σε ένα laptop μας .
 2. Στο laptop μας setάρουμε και τρέχουμε ένα πρόγραμμα (**Device Client σε Java**) που αναλαμβάνει να συνδεθεί σε κάποιο server και να του στέλνει τις τιμές από τα arduino όταν υπάρχουν αλλαγές .
 3. Σε ένα άλλο υπολογιστή (που είτε έχει πραγματική ip είτε βρίσκεται μέσα στο τοπικό δίκτυο του laptop μας) τρέχουμε ένα δεύτερο πρόγραμμα σε Java(**Server**) που έχει το ρόλο του server για να λαμβάνει τις μετρήσεις από το laptop μας αλλά και να κάνει τις κατάλληλες πιστοποιήσεις χρήστη και εφαρμογής. *Πρακτικά το βήμα 3 γίνεται πριν το βήμα 2.*
 4. Στην συνέχεια μπορούμε από οποιοδήποτε άλλο υπολογιστή (εφόσον “βλέπει” το server μας μέσα από το δίκτυο του) να συνδεθούμε , να πιστοποιηθούμε και να δούμε τις μονάδες που έχει πάνω του το arduino μας (και ανάλογα τα δικαιώματα και το τύπο των μονάδων να αλλάζουμε τις τιμές τους...) μέσω ενός τρίτου προγράμματος Java (**User client**) που θα έχει και γραφικό περιβάλλον.
 5. Τα δικαιώματα και οι λογαριασμοί θα βρίσκονται σε μια βάση δεδομένων (**MySql**).
 6. Επίσης, θα υπάρχει μια απλή ιστοσελίδα(**PHP website**) που θα δίνει τη δυνατότητα παραμετροποίησης της βάσης (Λογαριασμοί χρηστών και συσκευών αλλά και δικαιώματα προσπέλασης).

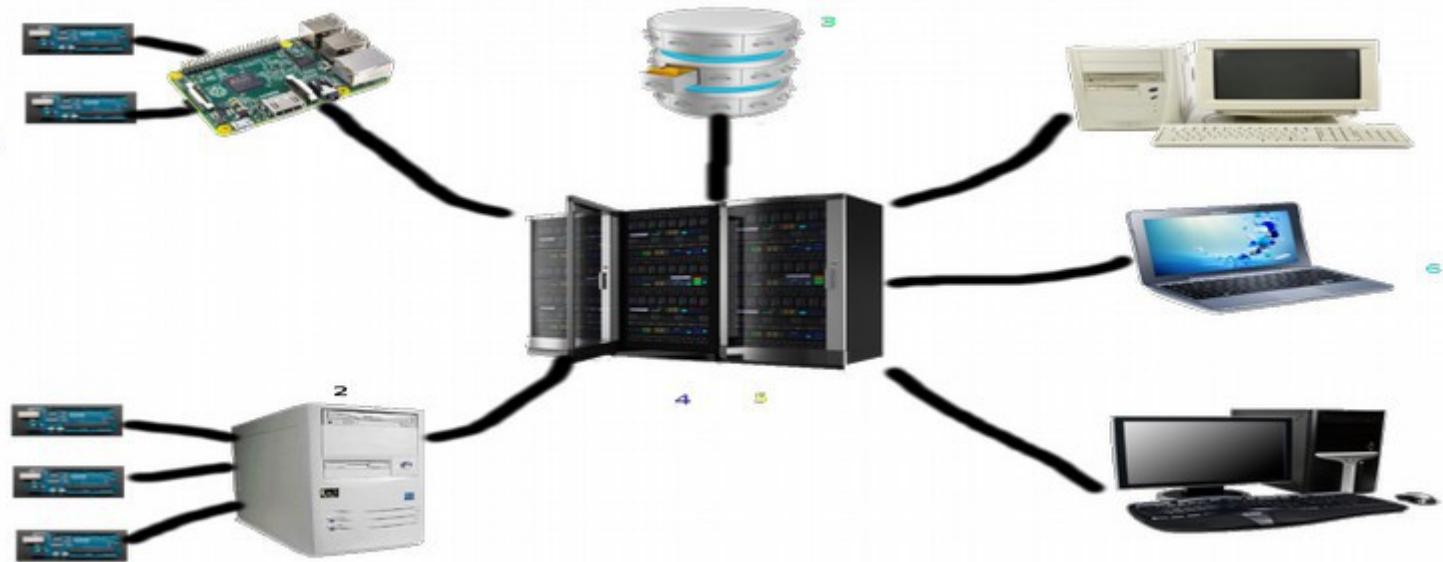
Ένα Use Case (γραφικό) :

Από την πλευρά καθαρά του χρήστη ως προς το σύστημα , δηλαδή χρησιμοποιώντας ο χρήστης άμεσα μόνο τα : UserClient ,DeviceClient ,Website[browser] και Arduino sketch (χωρίς δηλαδή να έχει άμεση σχέση ο χρήστης με Server, Website[server] και mysql) -1ο σενάριο.



Βασικά κομμάτια του συστήματος

Γράφημα συστήματος :



Περισσότερες λεπτομέρειες (συνοπτικά) για τα βασικότερα κομμάτια του συστήματος :

- ✓ (6)User Client (σε Java) για να μπορεί να συνδεθεί στον server κάποιος χρήστης και κατά επέκταση να μπορεί να παίρνει πληροφορίες από τις *μονάδες & να τις βλέπει σε γραφικό περιβάλλον αλλά&να μπορεί να αλλάζει (κατά περίπτωση) τις διάφορες τιμές τους.
Μέσω του client θα μπορεί ο χρήστης να συνδέεται σε πολλές *συσκευές ταυτόχρονα (με τη βοήθεια tabs) στις οποίες βέβαια θα έχει δικαιώματα&θα είναι συνδεδεμένες στο Server.
- ✓ (2)Device Client (σε java) που θα αναλαμβάνει να συνδέεται με το server&θα στέλνει τις πληροφορίες από τους μικρο ελεγκτές που θα είναι και αυτοί συνδεδεμένοι μαζί του .
Θα έχει & επιλογή για αλλαγή μικρο ρυθμίσεων(όπως IP,πόρτα DeviceName,Password κα).
- ✓ (1)Πρόγραμμα (σε Arduino sketch) που θα τρέχει στο Arduino και θα επικοινωνεί με το Device client σειριακά , στέλνοντας του πληροφορίες(σε μορφή συμβολοσειράς) από τις *μονάδες που θα έχει πάνω του αλλά και θα παίρνει μηνύματα για να αλλάξει ανάλογα την περίπτωση τιμές στις *μονάδες του ,αν πχ το ζητήσει κάποιος απομακρυσμένος χρήστης.
Παράλληλα θα φτιαχτούν και ρουτίνες για εικονικούς controllers που θα συνδέονται με το Device Client και θα δίνουν τυχαίες τιμές προγραμματιστικά για test.
- ✓ (5)Website (σε php,css,javascript και html) που θα επιτρέπει να φτιαχτούν διάφοροι λογαριασμοί χρηστών και λογαριασμοί *συσκευών αλλά και για να μπορούν να διαχειριστούν τα δικαιώματα προσπέλασης κάθε *συσκευής κα .
- ✓ (4)Server (σε java) που θα είναι ενδιάμεσος των Device Clients και των User Clients .
Θα βλέπει την ίδια βάση δεδομένων με την ιστοσελίδα για να ανακτά πληροφορίες πιστοποίησης κα .
- ✓ (3)Βάση δεδομένων (σε mysql) Σε πρώτη φάση θα υπάρχουν 4 πίνακες (λογικά συσχετισμένοι μεταξύ τους) και ίσως αργότερα να υπάρχει ανάγκη και για παραπάνω .
Αυτοί οι πίνακες θα είναι : users, devices ,deviceaccess , requests .
Τη βάση θα την βλέπουν μόνο ο Server και το WebSite (server) και όχι απευθείας οι clients.



Παρουσίαση συστήματος

Website

Είναι μια ιστοσελίδα που σκοπό έχει να παρέχει κάποιες βασικές λειτουργίες στους χρήστες του συστήματος , όπως είναι οι παρακάτω :

- ◆ Νέο Account Χρήστη .
- ◆ Login/Logout χρήστη
- ◆ Διαχείριση Devices (Νέο , Διαγραφή, Επεξεργασία κλπ)
- ◆ Αναζήτηση Devices με βάση “Device Name” για request
- ◆ Αναζήτηση χρηστών για πρόσκληση σε κάποιο Device
- ◆ Διαχείριση profile χρήστη
- ◆ Download (Των client κυρίως αλλά και του template sketch για να το προσαρμόσουν)
- ◆ Home (διάφορα νέα)
- ◆ Πληροφορίες-βοήθεια πάνω στην πτυχιακή (documentation)
- ◆ Πληροφορίες που αφορούν την πτυχιακή (θέμα, συντελεστές κλπ)

Ουσιαστικά είναι ένα γραφικό περιβάλλον με το οποίο υπάρχει η δυνατότητα μεταξύ άλλων , να φτιαχτούν λογαριασμοί χρηστών και συσκευών αλλά και να διαχειρίζονται τα δικαιώματα προσπέλασης πάνω στις συσκευές . Δηλαδή ο σημαντικότερος σκοπός ύπαρξης της ιστοσελίδας είναι να έχουν οι χρήστες μια διεπαφή με τη βάση δεδομένων .

- ☛ Με διάφορες αντιπροσωπευτικές κυρίως εικόνες θα παρουσιαστεί η υλοποίηση του Website
- ✓ Η αρχική σελίδα του Website :

The screenshot shows the homepage of the website. At the top, there's a navigation bar with links for HOME, Manage Devices, My Requests, Search other Devices, Download, Info-Help, and Thesis. Below the navigation bar is a large graphic illustrating a network architecture. It features a central server unit connected to multiple client devices (a laptop, a desktop computer, a monitor, and a smartphone). A small cartoon character wearing a blue hoodie and holding a coffee cup is positioned next to the network diagram. The year '2016' is visible at the bottom right of the character. At the very bottom of the page, there's a footer section containing text about the thesis and contact information for the author.

Θέμα πτυχιακής εργασίας :
Μελέτη ενός ολοκληρωμένου συστήματος για τη διαχείριση μικρο-ελεγκτών απομακρυσμένα σε πραγματικό χρόνο από εγχειριδιώνες χρηστών, όπως για τον Έλεγχο "Έντασης" απλών,
Πιλοτικά έχα υλοποιηθεί client-server εφαρμογή που θα εμπλέκει την διαχείριση διάφορων μονάδων
(Αυθεντήσεις, φωτα, διακόπτες) κόποιων μικρο-ελεγκτών (Arduino),
Εμφαση θα δοθεί στη χρήση τεχνολογιών Java, MySQL, Arduino sketch, PHP, JavaScript, CSS, HTML .

ΤΕΙ Athens - IT department
Φοιτητής : Γαλλιάκης Μιχαήλ (081001)
Επίλ. καθηγ. : Σκουρλας Χρήστος & Τσολούδης Αναστάσιος
Email : cs081001@teiaith.gr & michaelgallakatos@yahoo.gr
Όλα τα αρχεία βρίσκονται στο : ([github](#))
Ημερομηνία παρουσίασης : 4/2016



- ✓ Η περίπτωση δημιουργίας νέου λογαριασμού χρήστη :

Δημιουργία Νέου λογαριασμού χρήστη:

Όνομα χρήστη:

Κωδικός:

Επιβεβαίωση κωδικού:

Όνομα:

Επίθετο:

Email:

Αλέργησης:

Already registered ?

Login

Φόρμα πτυχιακής εργασίας :

Μελέτη για την ανάπτυξη μικρο-ελεγκτών απομακρυσμένα σε πραγματικό χρόνο από εγκριμένους χρήστες πληροφοριών που θα είναι τηλεοπτικής φύσης.

Πλοηγή που υλοποιεί client-server εφαρμογή που θα επεξεργάζει την διάχειρη των διάφορων μονάδων (Αυσθητήρες, φάτε, διαχοπτές) κάποιων μικρο-ελεγκτών (Arduino).

Εμφανίζεται στην παρακάτω παραγόμενη πλατφόρμα που παρέχει την διάχειρη των μονάδων.

ΤΕΙ Athens - IT department
Φοιτητής : Γαλλιάκης Μιχαήλ (081001)
Επίδ. καθηγ. : Σκουρλός Χρήστος & Τσαλακίδης Αναστάσιος
Email : cs081001@teiath.gr & michaelgalakakis@yahoo.gr
Όλα τα αρχεία βρίσκονται στο : [GitHub](#)
Ημερομηνία παρουσίασης : 4/2016

- ✓ Η περίπτωση συνδέσεως χρήστη με το σύστημα:

Σύνδεση :

Username:

Password:

Login

Φόρμα πτυχιακής εργασίας :

Μελέτη για την ανάπτυξη μικρο-ελεγκτών απομακρυσμένα σε πραγματικό χρόνο από εγκριμένους χρήστες πληροφοριών που θα είναι τηλεοπτικής φύσης.

Πλοηγή που υλοποιεί client-server εφαρμογή που θα επεξεργάζει την διάχειρη των διάφορων μονάδων (Αυσθητήρες, φάτε, διαχοπτές) κάποιων μικρο-ελεγκτών (Arduino).

Εμφανίζεται στην παρακάτω παραγόμενη πλατφόρμα που παρέχει την διάχειρη των μονάδων.

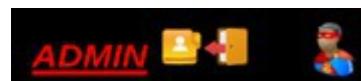
ΤΕΙ Athens - IT department
Φοιτητής : Γαλλιάκης Μιχαήλ (081001)
Επίδ. καθηγ. : Σκουρλός Χρήστος & Τσαλακίδης Αναστάσιος
Email : cs081001@teiath.gr & michaelgalakakis@yahoo.gr
Όλα τα αρχεία βρίσκονται στο : [GitHub](#)
Ημερομηνία παρουσίασης : 4/2016



Να επισημανθεί εδώ ,ότι το εικονίδιο που υπάρχει στο πάνω δεξιό μέρος με το ανθρωπάκι είναι ανάλογα το **τύπο του χρήστη**.

Υπάρχουν 3 τύποι χρηστών, οι του συστήματος(SuperMan), οι VIP και οι απλοί.

Ουσιαστικά δεν έχει ως τώρα κάποια διαφορά ο κάθε χρήστης και έτσι όλοι μπορούν να κάνουν τις ίδιες λειτουργίες. Μελλοντικά ίσως να έχουν επιπλέον δυνατότητες οι System και VIP .



- ✓ Η περίπτωση διαχείρισης συσκευών κάποιου χρήστη:

| Device: | Comment: | Right: | Owner: | Edit Device | Edit Access | Release Device |
|------------|-------------------------|--------|----------|-------------|-------------|----------------|
| kavala | Σπίτι στη Καβάλα | Owner | admin | | | |
| korinthos | Spiti stin korintho! | Owner | admin | | | |
| margarites | Εξοχικό στις Μαργαρίτες | Read | georgios | | | |
| melidoni | Εξοχικό στο μελιδόνι! | Admin | maria | | | |
| perama | To spiti sto perama! | Full | maria | | | |

Θέμα πτυχιακής εργασίας :
Μελέτη για ολοκληρωμένους συστήματος για τη διαχείριση μικρο-ελεγκτών απομακρυσμένων σε πραγματικό χρόνο από εγγυημένους χρήστες που για τον ένανχο "Έντυπων" σπάτιαν.
Πλοηγικά θα υλοποιηθεί client-server πραγματούντα που θα εκπέμψει την διαχείριση διαφορών μονάδων (Ιανοθήρας, φύτα, διακόπτες, αποστολή μηνημάτων Arduino).
Εμφαση θα δοθεί στη χρήση τεχνολογίας Java, MySQL, Arduino, sketch, PHP, JavaScript, CSS, HTML.

ΤΕΙ Athens - IT department
Φοιτητής : Γαλιλαίκης Μιχαήλ (081001)
Επλ. κοθηγ. : Σπουρήδης Κοζάρος & Ιωακώβης Αναστάρης
Email : cs081001@telath.gr & michaeligallikis@yahoo.gr
Όλα τα αρχεία βρίσκονται στο : GitHub
Ημερομηνία παροχούσης : 4/2018

Στην από πάνω φωτογραφία φαίνονται όλες οι συσκευές (5 διαφορετικά “έξυπνα” σπίτια) που έχει ο χρήστης “admin” . Πιο συγκεκριμένα φαίνεται το όνομα ,το σχόλιο και τα δικαιώματα που έχει ο χρήστης για κάθε συσκευή . Επίσης, μέσα από τον πίνακα ο χρήστης έχει την δυνατότητα να αλλάξει τα στοιχεία των συσκευών ή τα δικαιώματα άλλων χρηστών πάνω στις συσκευές εφόσον βέβαια έχει ο ίδιος το δικαίωμα (Owner,Admin). Ακόμη (αν δεν είναι Owner) μπορεί να αφαιρέσει κάποια συσκευή από αυτές που έχει , όπως επίσης δύναται και να δημιουργήσει μια νέα συσκευή.

Υπάρχουν 4 τύποι δικαιωμάτων και ουσιαστικά είναι τριών διαφορετικών ειδών :

1. Να είναι **Owner** δηλαδή ο δημιουργός της συσκευής .Μπορεί να αλλάξει τις τιμές των μονάδων και μπορεί να διαχειριστεί και τα δικαιώματα των χρηστών της συσκευής .
2. Να είναι **Admin** και ουσιαστικά κάνει τα ίδια πράγματα με τον owner .
3. Να είναι **Full Access** . Μπορεί να αλλάξει τιμές της συσκευής αλλά δεν μπορεί να δώσει, να αλλάξει και να αφαιρέσει δικαιώματα της συσκευής σε άλλους χρήστες .
4. Να είναι **Readonly**. Μπορεί μόνο να βλέπει τις τιμές και όχι να τις αλλάξει .
Και βέβαια δεν μπορεί να κάνει διαχείριση δικαιωμάτων .



- ✓ Η περύπτωση δημιουργίας νέας συσκευής από κάποιο χρήστη:

Δημιουργία Νέου Λογαριασμού Συσκευής:

Όνομα συσκευής:
Device name

Κωδικός συσκευής:
Password

Επιβεβαίωση κωδικού:
Confirm Password

Σχόλιο:
Comment

Θέμα πτυχιακής εργασίας :
Μελέτη ενός ολοκληρωμένου συστήματος για τη διαχείριση μικρο-ελεγκτών απομακρυνόμενα σε πραγματικό χρόνο από εγγεργούμενους χρήστες (π.χ για τον έλεγχο "έξυπνης" σπιτών).

TEI Athens - IT department
Φοιτητής : Γαλιλαϊκής Μιχαήλ (081001)
Επίδ. καθηγ. : Ιακουλάς Χρήστος & Τσολοκίδης Αναστάσιος

- ✓ Η περύπτωση αλλαγής στοιχείων μιας συσκευής από κάποιο χρήστη:

Διαχείριση συσκευής:
kavala

Κωδικός συσκευής:

Επιβεβαίωση κωδικού:

Σχόλιο:
Σπίτι στη Καβάλα

Ιδιοκτήτης: admin

Θέμα πτυχιακής εργασίας :
Μελέτη ενός ολοκληρωμένου συστήματος για τη διαχείριση μικρο-ελεγκτών απομακρυνόμενα σε πραγματικό χρόνο από εγγεργούμενους χρήστες (π.χ για τον έλεγχο "έξυπνης" σπιτών).
Πλέοντας σε πλέοντας σημείο στην εργασία που θα γνωρίζει την λειτουργία της συσκευής.

TEI Athens - IT department
Φοιτητής : Γαλιλαϊκής Μιχαήλ (081001)
Επίδ. καθηγ. : Ιακουλάς Χρήστος & Τσολοκίδης Αναστάσιος



- ✓ Η περίπτωση ανάθεσης ή αλλαγής δικαιωμάτων κάποιας συσκευής , σε άλλους χρήστες :

| User: | Current Right: | Owner: | Admin: | Full Access: | Read only: | None: | Option: | Request Status: |
|--------------|----------------|----------------------------------|----------------------------------|----------------------------------|-----------------------|----------------------------------|-----------------------|-----------------|
| admin | Administrator | <input checked="" type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | No Request |
| georgios | Administrator | <input type="radio"/> | <input checked="" type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input checked="" type="radio"/> | <input type="radio"/> | No Request |
| maria | Administrator | <input type="radio"/> | <input checked="" type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | No Request |
| mikeole | Administrator | <input type="radio"/> | <input type="radio"/> | <input checked="" type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | No Request |
| newUser | Administrator | <input type="radio"/> | <input type="radio"/> | <input checked="" type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | No Request |

Θέμα πτυχιακής εργασίας :
Μελέτη ενός ολοκληρωμένου συστήματος για τη διαχείριση μικρο-ελεγκτών απομακρυσμένα σε πραγματικό χρόνο από εγχειριδιώνας χρήστες, με για τον έλεγχο "έξυπνων" απλών.
Πλοκτικά θα υλοποιηθεί client-server εφαρμογή που θα επιτρέψει την διαχείριση διάφορων μονάδων (Αυτομητήρες, φωτιά διακόπτες) κάποιων μικρο-ελεγκτών (Arduino).
Έμφαση θα δοθεί στη χρήση τεχνολογιών Java, MySQL, Arduino sketch, PHP, Javascript, CSS, HTML.

TEI Athens - IT department
Φοιτήτης : Γαλλιάκης Μιχαήλ (081001)
Επίβλ. καθηγ. : Ιωακούλας Χαροπόσ & Τσολακίδης Αναστάριος
Email : cs081001@teliauth.gr & michaelgallaki@yahoo.gr
Όλα τα έργα δημοσιεύονται στο : GitHub
Ημερομηνία παρουσίασης : 4/2016

Στην πάνω φωτογραφία , ο πρώτος χρήστης είναι πάντα ο Owner στον οποίο δεν γίνεται να αλλάξουν τα δικαιώματα του .

Το υπογραμμισμένο όνομα χρήστη είναι αυτουνού που έχει συνδεθεί.

Μπορούν να αλλαχθούν τα δικαιώματα κάνοντας select το αντίστοιχο optioν και πατώντας το αντίστοιχο κουμπί (Αν είναι με “βελάκια” , αλλάζουν αμέσως τα δικαιώματα ενώ με το “ανθρωπάκι” δημιουργείτε ένα invite για να το εγκρίνει ο ανάλογος χρήστης που τον αφορά). Με το τρόπο αυτό γίνεται και να αφαιρεθούν δικαιώματα από κάποιους χρήστες .

Στο “request status” είναι η κατάσταση ενός invite εφόσον έχει ήδη γίνει κάποιο παλαιότερα .

Για να χειρίστει κάποιος μια τέτοια φόρμα αλλαγής δικαιωμάτων πρέπει να είναι Owner ή Admin .

- ✓ Μια περίπτωση αναζήτησης συσκευής άλλων χρηστών, για αποστολή request :

Search Device Name :
Device Name:

Δεν βρέθηκε η συσκευή :[KatiPouDenYparxei]

Θέμα πτυχιακής εργασίας :
Μελέτη ενός ολοκληρωμένου συστήματος για τη διαχείριση μικρο-ελεγκτών απομακρυσμένα σε πραγματικό χρόνο από εγχειριδιώνας χρήστες, με για τον έλεγχο "έξυπνων" απλών.
Πλοκτικά θα υλοποιηθεί client-server εφαρμογή που θα επιτρέψει την διαχείριση διάφορων μονάδων (Αυτομητήρες, φωτιά διακόπτες) κάποιων μικρο-ελεγκτών (Arduino).
Έμφαση θα δοθεί στη χρήση τεχνολογιών Java, MySQL, Arduino sketch, PHP, Javascript, CSS, HTML .

TEI Athens - IT department
Φοιτήτης : Γαλλιάκης Μιχαήλ (081001)
Επίβλ. καθηγ. : Ιωακούλας Χαροπόσ & Τσολακίδης Αναστάριος
Email : cs081001@teliauth.gr & michaelgallaki@yahoo.gr
Όλα τα έργα δημοσιεύονται στο : GitHub
Ημερομηνία παρουσίασης : 4/2016



AM:081001

- ✓ Μια 2η περίπτωση αναζήτησης συσκευής άλλων χρηστών , για αποστολή request :

Βρέθηκε η συσκευή :[andros]

Γράψε κάποιο σχόλιο και κάνε αίτηση για να αποκτήσεις δικαιώματα στην συσκευή :

Θα ήθελα να μου δώσετε δικαιώματα διαχειριστή!

Administrator Full Access Read Only

Θέμα πτυχιακής εργασίας :
Μελέτη ενός ολοκληρωμένου συστήματος για τη διαχείριση μικρο-ελεγκτών απομακρυσμένα σε πραγματικό χρόνο από γεγονολημένους χρήστες (πχ για τον έλεγχο "έντυπων" από την πλατφόρμα ή από οποιοδήποτε client-server εφαρμογή που θα επεξέρχεται την διαχείριση διαφόρων μονάδων). Επιπλέον θα δοθεί στη χρήση πλευράς λογισμικού Java, MySQL, Arduino sketch, PHP, JavaScript, CSS, HTML.

ΤΕΙ Athens - IT department
Φοκτής : Γαλλιάκης Μιχαήλ (081001)
Email: cs081001@teiaith.gr & michaelgalikis@yahoo.gr
Όλα τα αρχεία μπορούνται να τα βρετεις στο : [\[link\]](#)
Ηπειρουνιατικό Πανεπιστήμιο, 472916

Αν βρεθεί η συσκευή τότε μπορεί ο χρήστης να γράψει ένα μήνυμα για να το διαβάσει ο Owner(ή κάποιος admin) που θα κληθεί να τον αποδεχτεί . Όπως επίσης πρέπει να επιλέξει , τι δικαιώματα θέλει να αποκτήσει πάνω στην συσκευή .

- ✓ Η περίπτωση αλλαγής των στοιχείων του λογαριασμού του χρήστη (προφίλ) :

Διαχείριση λογαριασμού χρήστη:

Όνομα χρήστη: georgios

Κωδικός:

Επιβεβαίωση κωδικού:

Όνομα: georgios

Επίθετο: georgios

Email: georgios@pmg.gr

Θέμα πτυχιακής εργασίας :
Μελέτη ενός ολοκληρωμένου συστήματος για τη διαχείριση μικρο-ελεγκτών απομακρυσμένα σε πραγματικό χρόνο από γεγονολημένους χρήστες (πχ για τον έλεγχο "έντυπων" από την πλατφόρμα).

ΤΕΙ Athens - IT department
Φοκτής : Γαλλιάκης Μιχαήλ (081001)
Email: cs081001@teiaith.gr & michaelgalikis@yahoo.gr

- ✓ Η περύπτωση διαχείρισης των διαφόρων αιτήσεων που αφορούν τον εκάστοτε χρήστη :

| Αιτήσεις Χρηστών που θέλουν να αποκτήσουν δικαιώματα στις συσκευές μου : | | | | | |
|--------------------------------------------------------------------------|----------|-----------|----------------------------------------|-------------------------------------|--------------------------|
| Device: | User: | Right: | Request: | Accept: | Reject: |
| kavala | maria | —O (Full) | Θέλεις να με προσθέσεις; | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| kavala | georgios | —O (Read) | Θέλω να έχω αυτή τις συσκευή και σγάω. | <input checked="" type="checkbox"/> | <input type="checkbox"/> |

| Προσκλήσεις από άλλους χρήστες για να δεχτώ δικαιώματα σε νέες συσκευές : | | | | | |
|---------------------------------------------------------------------------|-----------|----------|--------------------------------|-------------------------------------|--------------------------|
| Device: | Right: | User: | Invite: | Accept: | Reject: |
| olxwrio | —O (Full) | georgios | You want to have this device ? | <input checked="" type="checkbox"/> | <input type="checkbox"/> |

| Προσκλήσεις που έχω κάνει για να πάρω δικαιώματα κάποιοι άλλοι χρήστες στις συσκευές μου : | | | | |
|--------------------------------------------------------------------------------------------|------|--------|---------|---------|
| Device: | User | Right: | Invite: | Delete: |
| Δεν υπάρχουν προσκλήσεις! | | | | |

| Αιτήσεις που έχω κάνει για να πάρω δικαιώματα σε μια νέα συσκευή : | | | |
|--------------------------------------------------------------------|------------|------------------------------------------------|---------------------------------------|
| Device: | Right: | Request: | Delete: |
| andros | —O (Admin) | Θα ήθελα να μου δώσεις δικαιώματα διαχειριστή! | <input type="button" value="Delete"/> |

- Πρώτος πίνακας: Αιτήσεις από χρήστες που θέλουν να αποκτήσουν δικαιώματα σε κάποιες από τις συσκευές του admin . Μπορεί να τις δεχθεί ή να τις απορρίψει .
- Δεύτερος πίνακας: Προσκλήσεις που έχει δεχθεί ο χρήστης admin από άλλους χρήστες για νέες συσκευές , ώστε αν τις αποδεχτεί , να μπορεί να τις έχει και αυτός .
- Τρίτος πίνακας: Ενεργές προσκλήσεις που έχει κάνει ο χρήστης admin, για να αποκτήσουν δικαιώματα άλλοι χρήστες πάνω στις δικές του συσκευές . Μπορεί ανά πάσα στιγμή να τις διαγράψει , ώστε να παύσουν να ισχύουν .
- Τέταρτος πίνακας: Αιτήσεις που έχει κάνει (από search) ο χρήστης admin ώστε να αποκτήσει δικαιώματα σε νέες συσκευές . Μπορεί να τα διαγράψει , ώστε να μην ισχύουν .

- ✓ Η περύπτωση που θέλει να κάνει “download” ο χρήστης κάποια πράγματα :

Εδώ παταχαντήστε για να διαγράψετε τα αποτελέσματα από την συσκευή σας με μικρό «εγκύρων απομεμνούματα» σε πραγματικό χρόνο από εγκριθείσαντες χρήστες (περιγραφής: «έξυπνα» αποτελέσματα πλεοντικής υπολογιστής client-server εφερόντων που δε εκτελεῖ τη διεύρυνση των διεύθυνσης διαδικονίας μενούδων (Αυτοματισμές, φωτιά, βιολόντες) κάποιων μικρο-ελεγκτών (Arduino)).

Είμαστε έτοιμοι να διεύθουμε την υποστήση της στην πλατφόρμα MySQL, Arduino Sketch, PHP, JavaScript, CSS, HTML .

TEI Athens - IT department
Φανταστική Γαλλιάκη Μιχαήλ AM:081001
Επίδρ. καθηγ.: Δημήτρης Καζαντζής Τακτικός Αναπληρωτής
Email : am081001@teiath.gr & michalegalakis@hua.gr
Όλα τα αρχεία διαθέσιται στο: tiny.cc/meyarw
Ημερομηνία παροντότητας : 4/2016



AM:081001

- ✓ Η περύπτωση που θέλει ο χρήστης να μάθει διάφορες πληροφορίες ή βοήθειες :

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

με θέμα :

Μελέτη ενός ολοκληρωμένου συστήματος για τη διαχείριση μικρο-ελεγκτών απομακρυνόμενά σε πραγματικό χρόνο από εγγεγραμμένους χρήστες (πχ για τον έλεγχο "έξυπνων" σπιτιών). Πιλοτικά θα υλοποιηθεί client-server εφαρμογή που θα επιτρέπει την διαχείριση διάφορων μονάδων (Αυτομητήρες, φώτα, διακόπτες) κάποιων μικρο-ελεγκτών (Arduino). Έμφαση θα δοθεί στη χρήση τεχνολογιών Java, MySQL, Arduino sketch, PHP, Javascript, CSS, HTML.

- ✓ Η περύπτωση που θέλει ο χρήστης να μάθει πληροφορίες που αφορούν την πτυχιακή :

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

με θέμα :

Μελέτη ενός ολοκληρωμένου συστήματος για τη διαχείριση μικρο-ελεγκτών απομακρυνόμενά σε πραγματικό χρόνο από εγγεγραμμένους χρήστες (πχ για τον έλεγχο "έξυπνων" σπιτιών). Πιλοτικά θα υλοποιηθεί client-server εφαρμογή που θα επιτρέπει την διαχείριση διάφορων μονάδων (Αυτομητήρες, φώτα, διακόπτες) κάποιων μικρο-ελεγκτών (Arduino). Έμφαση θα δοθεί στη χρήση τεχνολογιών Java, MySQL, Arduino sketch, PHP, Javascript, CSS, HTML.

Technological Educational Institute of Athens - IT department.
Φοιτητής : **Γαλλιάκης Μιχαήλ (081001)**
Επιβλέπων καθηγητές : Σκουρλάς Χρήστος & Τσολακίδης Αναστάριος
Email : cs081001@teiath.gr & michaelgallakakis@yahoo.gr
Όλα τα αρχεία βρίσκονται στο : ([Github](#))
Ημερομηνία παρουσίασης : 4/2016

Θέμα πτυχιακής εργασίας :
Μελέτη ενός ολοκληρωμένου συστήματος για τη διαχείριση μικρο-ελεγκτών απομακρυνόμενά σε πραγματικό χρόνο από εγγεγραμμένους χρήστες (πχ για τον έλεγχο "έξυπνων" σπιτιών). Πιλοτικά θα υλοποιηθεί client-server εφαρμογή που θα επιτρέπει την διαχείριση διάφορων μονάδων (Αυτομητήρες, φώτα, διακόπτες) κάποιων μικρο-ελεγκτών (Arduino). Έμφαση θα δοθεί στη χρήση τεχνολογιών Java, MySQL, Arduino sketch, PHP, Javascript, CSS, HTML.

TEI Athens - IT department
Φοιτητής : **Γαλλιάκης Μιχαήλ (081001)**
Επιβλ. καθηγ. : Σκουρλάς Χρήστος & Τσολακίδης Αναστάριος
Email : cs081001@teiath.gr & michaelgallakakis@yahoo.gr
Όλα τα αρχεία βρίσκονται στο : ([Github](#))
Ημερομηνία παρουσίασης : 4/2016



Βάση δεδομένων (Database)

Επειδή η ενότητα αυτή έχει να κάνει μόνο με την παρουσίαση και όχι με λεπτομέρειες για το πως υλοποιήθηκε, θα φανούν μόνο 2 φωτογραφίες με το περιεχόμενο της βάσης δεδομένων, δηλαδή τους 4 λογικά συσχετισμένους πίνακες (users,devices,deviceaccess και requests) :



Device Client

Το “DeviceClient” είναι ένα πρόγραμμα (χωρίς γραφικό περιβάλλον) που αναγνωρίζει, συντονίζει και επικοινωνεί με όλους τους μικροελεγκτές(arduino) που είναι συνδεδεμένοι στον υπολογιστή που “τρέχει”.Σκοπός του είναι να συνδέεται με το Server (και αφού κάνει login με κάποιο λογαριασμό συσκευής) να του στείλει όλες τις πληροφορίες από τα arduino και τις μονάδες τους , που ήδη έχουν αναγνωριστεί πάνω στον υπολογιστή.Τα στοιχεία του λογαριασμού συσκευής όπως και όλες οι υπόλοιπες ρυθμίσεις υπάρχουν σε ένα αρχείο xml (που συμπληρώνει κατάλληλα κάποιο φυσικό πρόσωπο) και διαβάζονται από το πρόγραμμα όταν πρώτο-ξεκινάει (Όπως devicename,password,ip και port Server, κα).Στην συνέχεια μόλις έχει γίνει απόλυτα η αρχικοποίηση και ο πλήρης συντονισμός με το server και τα arduino , αναλαμβάνει να ενημερώνει σε πραγματικό χρόνο το Server με τις όποιες αλλαγές προκύπτουν στις μονάδες των arduino . Παράλληλα βέβαια ο “DeviceClient” είναι έτοιμος να δεχθεί και να αλλάξει τις καταστάσεις των μονάδων που “βλέπει” ,αν το “ζητήσει” κάποιος απομακρυσμένος χρήστης που εποπτεύει “ζωντανά” εκείνη την ώρα την συσκευή (σύνολο από Arduino με τις μονάδες τους) .Ακόμη το πρόγραμμα είναι σε θέση να αναγνωρίζει και να προσθέτει συνεχώς νέα arduino αν συνδεθούν στον υπολογιστή . Όπως επίσης καταλαβαίνει αν βγάλει κάποιο arduino από τον υπολογιστή και ενημερώνει κατάλληλα το Server ώστε να το πληροφορηθούν οι εκάστοτε UserClients.Μαζί με το παραπάνω είναι σε θέση , στην περίπτωση που χαθεί η σύνδεση με το Server , να κάνει ανά διαστήματα προσπάθειες να ξανά-συνδεθεί εκ νέου με τον εξυπηρετητή . Εκτελείτε μέσα από οποιαδήποτε λειτουργικό σύστημα (που έχει java), πχ μέσα από ένα raspberry :

Entomological pest and predator notes received prior to 1980 (see also 1980), by field and area (raspberry).

```
[File Edit View Search Terminal Help]
raspberrypi ~ Desktop/mikeThesis/dist/DeviceClient $ sudo java -jar DeviceClientV1.jar
[PI] www.michaelgallikis.gr -> [http://www.michaelgallikis.gr]
[PI] Michael Gallikis 2009 ]
[PI] TEI Athens - IT department ]
[PI] Email : cs081001@teliauth.gr & michaelgallikis@yahoo.gr . ]
[PI] All files can be found : ]
[PI] "https://github.com/michaelgallikis/myThesis.git" ]
[PI] "https://github.com/michaelgallikis/MyThesisArduino" ]
[PI] Open 'PMGTCDC.config' File Success! ]
[PI] series 1
[PI] series ]
[PI] 192.168.2.200 ]
[PI] 50120 ]
[PI] 68 ]
[PI] 10 ]
[PI] true ]
[PI] certification OK! ]
[PI] DeviceName and Password Correct - Login OK! ]
[PI] Warning: Removing stale lock file: /var/lock/LCK_.ttyACM0
[PI] Warning: Removing stale lock file: /var/lock/LCK_.ttyUSB0
[PI] ./dev/ttyACM0 ]
[PI] serialPort[2,00] ]
[PI] #22000NewValues:1*(Ard2/Mikrofono[3.00]; ]
[PI] #22000NewValues:1*(Ard2/Mikrofono[5.00]; ]
[PI] #22000NewValues:1*(Ard2/Mikrofono[7.00]; ]
[PI] #22000NewValues:1*(Ard2/Mikrofono[9.00]; ]
[PI] #22000NewValues:1*(Ard2/Mikrofono[14.00]; ]
[PI] #22000NewValues:1*(Ard2/Mikrofono[2.00]; ]
[PI] #22000NewValues:1*(Ard2/Fotias[9.00]; ]
[PI] #22000NewValues:12*(Ard2/Mikrofono[1.00]);(Ard2/Fotias[10.00]; )
[PI] #22000NewValues:13*(Ard2/Mikrofono[2.00]; )
[PI] #22000NewValues:13*(Ard2/Mikrofono[1.00]; )
[PI] #22000NewValues:13*(Ard2/Mikrofono[4.00]; )
[PI] stopped! ]
[AI] Version : PMG.V1 ]
[AI] devNameConfirm : 22000 ]
[AI] info #22000NewController:0*(Ard2/Mikrofono[0|0|14.00|300.00|0.00|NL](Fotias[0|0|10.00|255.00|0.00|NL)(Mousiki[1|1|1|0.00|255.00|0.00|128SW)(
[AI] #22000NewController:1*(Ard2/Mikrofono[0|1|14.00|255.00|0.00|NL)(Fotias[0|0|10.00|255.00|0.00|NL)(Mousiki[1|1|1|0.00|255.00|0.00|128SW) ]
[PI] #22000NewController:16*(Ard2/Mikrofono[0|1|14.00|200.00|0.00|NL](Fotias[0|0|10.00|255.00|0.00|NL)(Mousiki[1|1|1|0.00|255.00|0.00|128SW) ]
[PI] #22000NewController:17*(Ard2/Mikrofono[0|1|14.00|255.00|0.00|NL)(Fotias[0|0|10.00|255.00|0.00|NL)(Mousiki[1|1|1|0.00|255.00|0.00|128SW) ]
[PI] #22000NewController:18*(Ard2/Mikrofono[0|1|14.00|255.00|0.00|NL)(Fotias[0|0|10.00|255.00|0.00|NL)(Mousiki[1|1|1|0.00|255.00|0.00|128SW) ]
[File Edit View Search Terminal Help]
GNU nano 2.2.6
File: PMGTCDC.config

<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<pmgThesisConfigDeviceClient>
<DeviceName>series</DeviceName>
<Password>series</Password>
<Address>192.168.2.200</Address>
<Port>50120</Port>
<secondsForNextTryConnect>60</secondsForNextTryConnect>
<secondsForNextSearchArduino>10</secondsForNextSearchArduino>
<viewMessage>True</viewMessage>
</pmgThesisConfigDeviceClient>
```

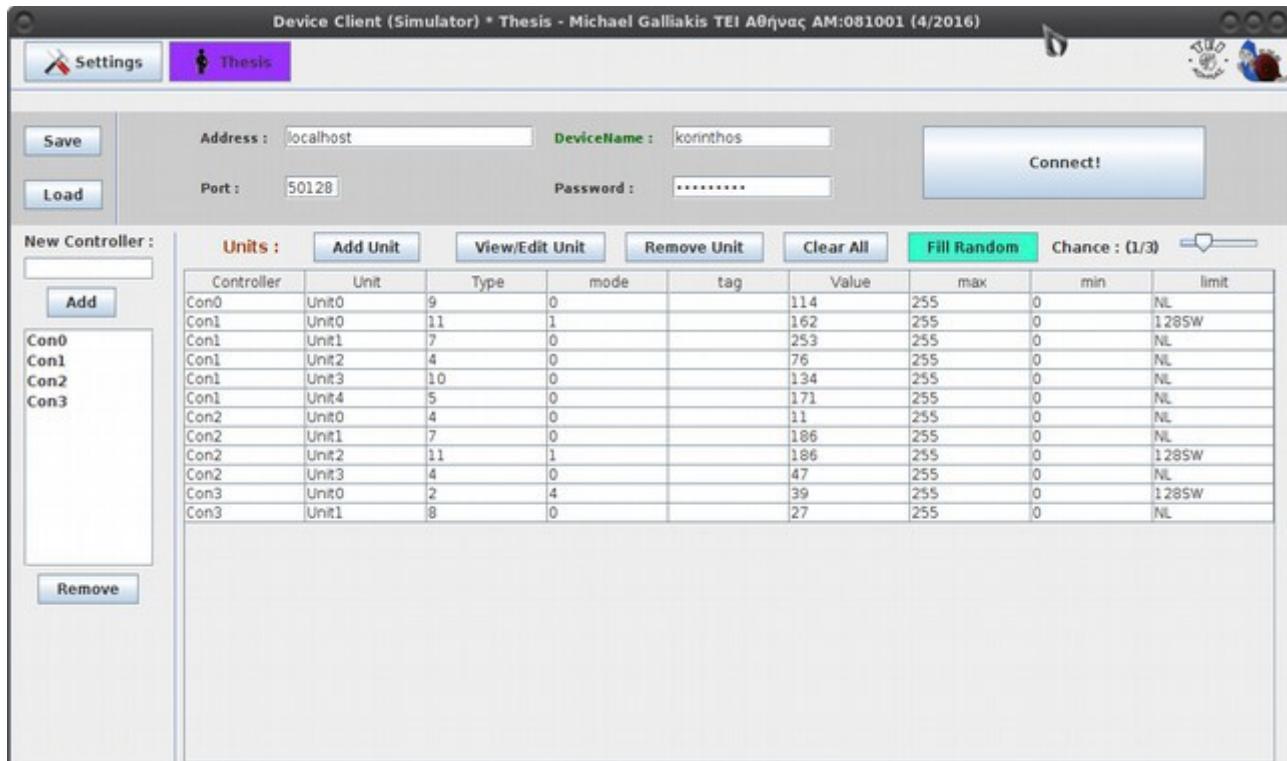
Το παραθυράκι που φαίνεται δεξιά είναι το config αρχείο των ρυθμίσεων .



Simulator για το Device Client

Επειδή στην αρχή δεν υπήρχε διαθέσιμο κάποιο arduino και γενικότερα εξαρτήματα (όπως αισθητήρες ,led αντιστάσεις, καλώδια κ.α), δημιουργήθηκε ένας Simulator με γραφικό περιβάλλον για να προσομοιώνει έναν πραγματικό Device Client .

Στην από κάτω εικόνα φαίνεται μια περύπτωση του “Device Client Simulator” :



Όπως φαίνεται , γίνεται να αλλαχθούν από πάνω τα στοιχεία IP,Port,Devicename,Password και με το Κουμπί Connect να πραγματοποιηθεί μια σύνδεση με το Server.

Με τα κουμπιά save και load γίνεται να αποθηκευθούν και να φορτωθούν κάποια profile με σκοπό ουσιαστικά να συμπληρώνεται αυτόματα με περιεχόμενο η όλη φόρμα που υπάρχει (πχ όταν ανοίγουμε το πρόγραμμα ,μπορούμε να φορτώσουμε ένα παλαιότερο profile για να μην καταχωρούμε εκ νέου στοιχεία).

Παράλληλα υπάρχει η δυνατότητα να γίνονται για παράδειγμα διάφορα σενάρια (με συγκεκριμένες μονάδες , συσκευή,IP ,Port κλπ), και να αποθηκευθούν σαν xml και να ανακτούνται όποτε θέλει ο χρήστης .

Η φόρμα όπως φαίνεται και στη φωτογραφία επιτρέπει στο χρήστη να προσθέτει χειροκίνητα ελεγκτές και διάφορες μονάδες όπως και να τις διαγράφει ή να τις αλλάζει . Επίσης, υπάρχει η δυνατότητα στο χρήστη , να γεμίσει (με το κουμπί “fill Random”) τυχαίες μονάδες και ελεγκτές ώστε να μην χρειάζεται να τις καταχωρεί μία-μία .

Ακόμη δύναται η επιλογή “Chance” (Που αλλάζει και όταν υπάρχει σύνδεση με το Server) στο χρήστη , για να καθορίζει την πιθανότητα αλλαγής τιμών των μονάδων (Πχ 1:1 αλλάζουν πάντα οι τιμές κάθε χρονική στιγμή ενώ 1:8 είναι πιο μικρή η πιθανότητα να αλλάξει κάποια τιμή).



Τώρα, κάθε μονάδα έχει **9 χαρακτηριστικά**:

1. Όνομα Ελεγκτή
 2. Όνομα Μονάδας (Ο συνδυασμός των ονομάτων του Ελεγκτή & Μονάδας πρέπει να είναι μοναδικός)
 3. Τύπος μονάδας (Μέχρι στιγμής υπάρχουν 12 από 0-11)
 - 0) "No Category Dimming"
 - 1) "No Category Switch"
 - 2) "Lamp"
 - 3) "Brightness"
 - 4) "Motion"
 - 5) "Distance"
 - 6) "Sound"
 - 7) "Vibration"
 - 8) "Smoke"
 - 9) "Temperature"
 - 10) "Humidity"
 - 11) "switch"
 4. Mode μονάδας (υπάρχουν 5 από 0-4)
 - 0 : Auto - Η μονάδα παίρνει αυτόματα τιμή και δεν μπορεί αλλαχτεί απομακρυσμένα.
 - 1 : Remote – Η μονάδα αλλάζει μόνο απομακρυσμένα .
 - 2 : Both – Η μονάδα μπορεί να αλλάζει ταυτόχρονα είτε αυτόματα είτε απομακρυσμένα.
 - 3 : Auto – Το ίδιο με το 0 αλλά μπορεί να το ρυθμίσει ο απομακ. χρήστης και να το κάνει Remote .
 - 4 : Remote - Το ίδιο με το 1 αλλά μπορεί να το ρυθμίσει ο απομακ. χρήστης και να το κάνει Auto .
 5. Tag Μονάδας (Για μελλοντική πιθανόν χρήση ,μια ετικέτα-σχόλιο για τη μονάδα)
 6. Value μονάδας (Η τιμή της κάθε μονάδας)
 7. Max value μονάδας (Η μέγιστη τιμή που μπορεί να πάρει μια μονάδα)
 8. Min value μονάδας (Η μικρότερη τιμή που μπορεί να πάρει μια μονάδα)
 9. Limit (Υπάρχουν 3 περιπτώσεις)
 - Να μην έχει Limit (NL)
 - Να έχει έναν αριθμό σαν όριο και στο τέλος τα γράμματα TB -trackbar.
 - Να έχει έναν αριθμό σαν όριο και στο τέλος τα γράμματα SW -switch.

Χρησιμοποιείτε για να ξέρει ο UserClient από την “άλλη πλευρά” πως να φερθεί στην συγκεκριμένη μονάδα. Αν η μονάδα έχει 2 ειδών εικόνες (Πχ ανοιχτή και κλειστή λάμπα) τότε αν το value περάσει το όριο δείχνει ο UserClient ανοιχτή την λάμπα και το ανάποδο .

Τώρα το TB σημαίνει ότι αν επιχειρήσει ένας χρήστης να αλλάξει τη τιμή της μονάδας θα έχει την επιλογή μιας trackbar αλλιώς αν είναι SW θα έχει 2 καταστάσεις η μονάδα και από τη μία κατάσταση θα γίνεται η άλλη πχ κλειστό-ανοιχτό...
- Επιλογή για νέα μονάδα
-
- Επιλογή για αλλαγή μονάδας
-
- ✓ Τα δύο κουμπιά που υπάρχουν στη κορυφή είναι το ένα για ρυθμίσεις (το οποίο δεν έχει υλοποιηθεί και υπάρχει για πιθανόν μελλοντική χρήση) και το άλλο για να δει ο χρήστης διάφορες πληροφορίες που αφορούν την Πτυχιακή.
 - Αν “κλικάρει” κάποιος το κουμπί “settings” θα δει το εξής μήνυμα :
-
- Πτυχιακή εργασία Γαλλιάκη Μιχαήλ * Απρίλιος 2016 * Τμήμα Πληροφορικής ΤΕΙ Αθήνας (24/83)



AM:081001

- Αν “κλικάρει” κάποιος το κουμπί “thesis” θα δει το εξής παράθυρο :

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

με θέμα :

Μελέτη ενός ολοκληρωμένου συστήματος για τη διαχείριση μικρο-ελεγκτών απομακρυσμένα σε πραγματικό χρόνο από εγγεγραμμένους χρήστες (πχ για τον έλεγχο "έξυπνων" σπιτιών). Πλοτικά θα υλοποιηθεί client-server εφαρμογή που θα επιτρέπει την διαχείριση διάφορων μονάδων (Αισθητήρες, φώτα, διακόπτες) κάποιων μικρο-ελεγκτών (Arduino). Εμφαση στη χρήση τεχνολογιών Java, MySQL, Arduino sketch, PHP, Javascript, CSS, HTML.

Technological Educational Institute of Athens - IT department.

Φοιτητής : **Γαλλιάκης Μιχαήλ**

AM : **081001**

Επιβλέπων καθηγητές : **Σκουρλάς Χρήστος & Τσολακίδης Αναστάσιος**

Email : **cs081001@teiath.gr & michaelgalliakakis@yahoo.gr**

Όλα τα αρχεία : **<https://github.com/michaelgalliakakis/myThesis.git>**

Ημερομηνία παρουσίασης : 4/2016

Μια ακόμη περίπτωση του Simulator που όμως έχει επιτευχθεί σύνδεση με το Server είναι η από κάτω :

Device Client (Simulator) * Thesis - Michael Galliakakis TEI Αθήνας AM:081001 (4/2016)

Settings Thesis

| | | | |
|------------------|----------------------------------------------------------------------------------|------------------------|-----------|
| Save | Address : localhost | DeviceName : korinthos | Connected |
| Load | Port : 50128 | Password : | |
| New Controller : | Units : Add Unit View/Edit Unit Remove Unit Clear All Fill Random Chance : (1/3) | | |
| Add | Controller Unit Type mode tag Value max min limit | | |
| Con0 | Unit0 9 0 54.0 255 0 NL | | |
| Con1 | Unit0 11 1 162 255 0 128SW | | |
| Con1 | Unit1 7 0 147.0 255 0 NL | | |
| Con1 | Unit2 4 0 12.0 255 0 NL | | |
| Con1 | Unit3 10 0 145.0 255 0 NL | | |
| Con1 | Unit4 5 0 253.0 255 0 NL | | |
| Con2 | Unit0 4 0 11 255 0 NL | | |
| Con2 | Unit1 7 0 186 255 0 NL | | |
| Con2 | Unit2 11 1 186 255 0 128SW | | |
| Con2 | Unit3 4 0 64.0 255 0 NL | | |
| Con3 | Unit0 2 4 39 255 0 128SW | | |
| Con3 | Unit1 8 0 33.0 255 0 NL | | |

Remove

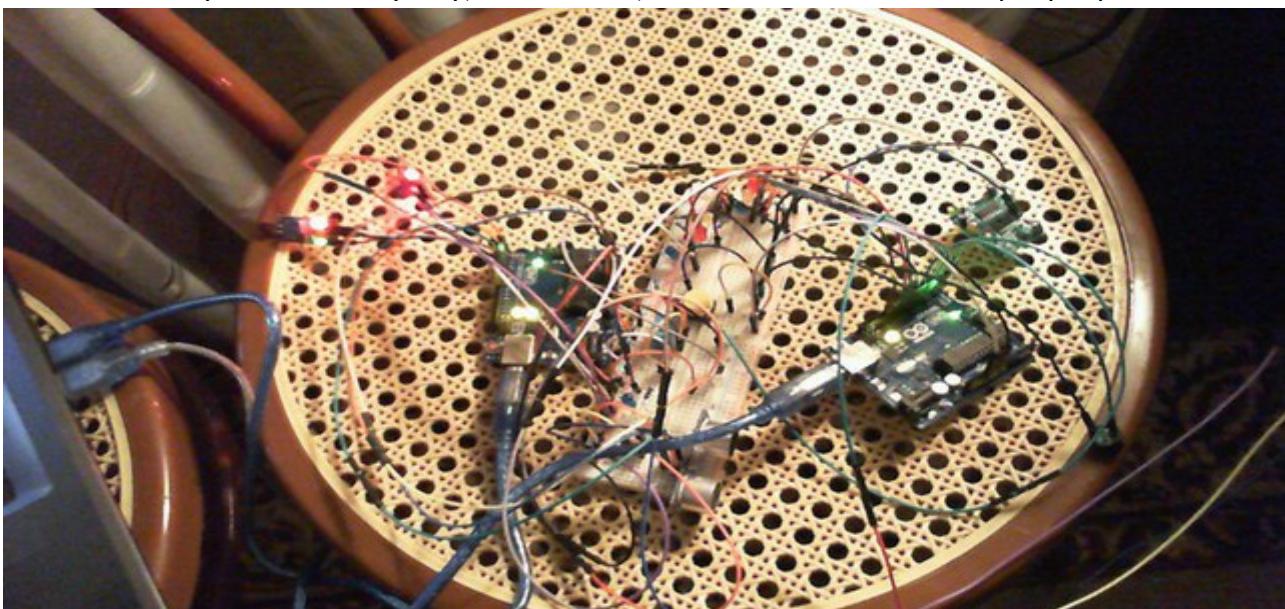


Arduino

Στην πτυχιακή αυτή, το arduino προγραμματίζεται (με βάση ένα συγκεκριμένο πρότυπο πρόγραμμα-sketch) για να μπορεί να επικοινωνήσει αμφίδρομα με έναν υπολογιστή(DeviceClient). Πιο συγκεκριμένα είναι σε θέση να διαβάζει εντολές από το DeviceClient και να πράττει αναλόγως (όπως για παράδειγμα διάφορες εντολές συγχρονισμού και αλλαγής mode-value κάποιων μονάδων) Επίσης , ένα arduino κάνει μετρήσεις από το πραγματικό κόσμο (μέσω διαφόρων αισθητήρων) και παρατηρεί αν έχει αλλάξει κατάσταση πχ κάποιος διακόπτης ή λάμπα (από κάποιο πιθανόν φυσικό πρόσωπο) και στέλνει τις νέες (διαφορετικές)τιμές των μονάδων κάθε φορά στον υπολογιστή .

Για τα πολλά τεχνικά μέρη θα γίνει αναφορά εκτενώς στην επόμενη ενότητα για το πως υλοποιήθηκαν

Μια φωτογραφία (από κάτω) που δείχνει δυο arduino να έχουν πάνω τους διάφορες μονάδες και να επικοινωνούν με έναν υπολογιστή(DeviceClient) ώστε να εποπτεύονται απομακρυσμένα :



Μια φωτογραφία (κάτω) από την έξοδο που βγάζει ένα arduino (με βάση το πρότυπο πρόγραμμα) μετά από την εντολή “cat” στην “συσκευή” όπως την “βλέπει” ένα raspberry (/dev/ttyUSB0) :

```
File Edit View Search Terminal Help
pi@raspberrypi: ~ cat /dev/ttyUSB0
#RC$NewValues:5*(Ard1|thermometro2|-1.00)(Ard1|Ygrasia|-1.00)(Ard1|Mikrofono|126.00)(Ard1|Fotias|212.00)(Ard1|Fotoastitirias|315.00);
35
#RC$NewValues:5*(Ard1|thermometro2|-1.00)(Ard1|Ygrasia|-1.00)(Ard1|Mikrofono|126.00)(Ard1|Fotias|212.00)(Ard1|Fotoastitirias|315.00);

35
#RC$NewValues:5*(Ard1|thermometro2|-1.00)(Ard1|Ygrasia|-1.00)(Ard1|Mikrofono|126.00)(Ard1|Fotias|212.00)(Ard1|Fotoastitirias|315.00);

35
#RC$NewValues:5*(Ard1|Mikrofono|257.00)(Ard1|Fotias|176.00)(Ard1|Fotoastitirias|572.00);
35
62
#RC$NewValues:5*(Ard1|thermometro2|-1.00)(Ard1|Ygrasia|-1.00)(Ard1|Mikrofono|126.00)(Ard1|Fotias|212.00)(Ard1|Fotoastitirias|315.00);

35
#RC$NewValues:3*(Ard1|Mikrofono|398.00)(Ard1|Fotias|198.00)(Ard1|Fotoastitirias|482.00);
35
```

Μέσα από το java πρόγραμμα (DeviceClient) δύναται όχι μόνο να διαβάζουμε (και χωρίς χαμένα bytes) αλλά και να στέλνουμε μηνύματα προς το arduino σε αντίθεση με την εντολή cat .



Server

Κάποιες από τις σημαντικότερες λειτουργίες του Server :

- Ο server αναλαμβάνει αν συνδεθεί πάνω του κάποιος client (Device ή User) να τον αναγνωρίζει και να κάνει την ανάλογη πιστοποίηση χρήστη ή συσκευής με την βοήθεια της βάσης δεδομένων.
- Μετά αν ο client είναι συσκευή (DeviceClient) δημιουργείτε ένα νέο “κανάλι” που μπορούν αργότερα να συνδεθούν πάνω του χρήστες που έχουν στη συσκευή δικαιώματα . Και βέβαια ενημερώνει όλους τους χρήστες που είναι ήδη συνδεδεμένοι και έχουν δικαιώματα στη συσκευή ότι κάποια νέα συσκευή είναι διαθέσιμη για εποπτεία .
- Αν ο client είναι κάποιος χρήστης για εποπτεία (UserClient), τότε ο Server με τη βοήθεια της βάσης στέλνει στο χρήστη τις συσκευές που έχει δικαιώματα και την κατάσταση τους (πχ αν είναι up ή down, δηλαδή αν είναι ήδη συνδεδεμένοι πάνω στο server).
- Τώρα σε επόμενη φάση αν ο χρήστης επιλέξει να κάνει εποπτεία σε μια συσκευή τότε ο Server στέλνει όλες τις απαραίτητες πληροφορίες που έχει ήδη από το αντίστοιχο Device Client και “βάζει στο ίδιο κανάλι” το UserClient με το DeviceClient για να μπορούν να επικοινωνούν ... (Μοιάζει και με τη λογική ενός απλού chat πολλών ταυτόχρονων χρηστών)
- ✓ Μια άλλη δυνατότητα του server είναι να εμφανίζει μια ροή με μηνύματα από διάφορα γεγονότα που συμβαίνουν, όπως σύνδεση νέου χρήστη , αλλαγή τιμής μια μονάδας, κλείσιμο socket ομαλά από κάποιο χρήστη ή και απροσδόκητα κλπ .
(Μέσα από ένα αρχείο xml που διαβάζει ο Server μεταξύ άλλων μπορεί να καθοριστεί το πόσο σημαντικά μηνύματα θα εμφανίζονται ,ώστε να υπάρχει συγκεκριμένου μεγέθους ροή)

Υπάρχουν τεχνικά μέρη στα οποία θα γίνει αναφορά στην επόμενη ενότητα για το πως υλοποιήθηκαν

Ένα screenShot από το server ενώ τρέχει :

```
File Edit View Search Terminal PMG Terminal
[michael@PMG: /] ~ /gitLab/myThesis/ServerV1/dist/[ 18:04:23 Sat Apr 02]
[sudo] password for michael:
[ : ] #####
[ : ] Thesis Michael Galliakis 2016.
[ : ] TEI Athens - IT department.
[ : ] Email : cs081001@telath.gr & michaelgalliakis@yahoo.gr .
[ : ] All files can be found :
[ : ] "https://github.com/michaelgalliakis/myThesis.git"
[ : ] #####
[ : ] ****
[ : ] Starting Server ...
[ : ] Failed : File 'PMGTCGS.config' not Exists!
[ : ] Successfully check Database
[ : ] Successfully connected to the database.
[ : ] Opened successfully the server at the door 50128.
[ : ] Started Server.
[ : ] Listen ...
[ : ] Device Client | temporaryID : 0 , SysID : 150 InitControllers OK!
[ : ] Device Client | temporaryID : 0 , SysID : 150 NewController OK!
[ : ] Device Client | temporaryID : 0 , SysID : 150 NewController OK!
[ : ] Device Client | temporaryID : 0 , SysID : 150 NewController OK!
[ : ] Device Client | temporaryID : 0 , SysID : 150 NewController OK!
[ : ] Device Client | temporaryID : 0 , SysID : 150 NewController OK!
[ : ] Device Client | temporaryID : 0 , SysID : 150 put Initialization Finished!
[ : ] Device Client | temporaryID : 1 , SysID : 210 InitControllers OK!
[ : ] Device Client | temporaryID : 1 , SysID : 210 NewController OK!
[ : ] Device Client | temporaryID : 1 , SysID : 210 NewController OK!
[ : ] Device Client | temporaryID : 1 , SysID : 210 NewController OK!
[ : ] Device Client | temporaryID : 1 , SysID : 210 put Initialization Finished!
[ : ] UserDesktop Client | temporaryID : 3 , SysID : 13U01 | GetDevicesInfo put message :#SSPutDevicesInfo:6*(15|korinthos|A|U)(16|perama|A|D)(17|m
[ : ] ldoni|A|D)(18|olixwrio|F|D)(21|andros|A|U)(22|serres|A|D);
[ : ] UserDesktop Client | temporaryID : 4 , SysID : 13U02 BringMeDevice OK!
[ : ] UserDesktop Client | temporaryID : 5 , SysID : 13U03 BringMeDevice OK!
[ : ] UserDesktop Client | temporaryID : 7 , SysID : 12U01 | GetDevicesInfo put message :#SSPutDevicesInfo:5*(15|korinthos|A|U)(16|perama|F|D)(17|m
[ : ] ldoni|A|D)(18|olixwrio|R|D)(20|kavala|A|D);
[ : ] UserDesktop Client | temporaryID : 8 , SysID : 12U02 BringMeDevice OK!
[ : ( ] ID : 12U02 | Left the Device : 150 ERROR(1)!!!
[ : ( ] ID : 12U02 | Receiver the message :(null) - ERROR(2)!!!
[ : ( ] ID : 15D | Receiver the message :(#150#0$Quit:1"(Bey-bey!)) - ERROR(2)!!!
[ : ( ] ID : 13U02 | Left the Device : 21D ERROR(1)!!!
[ : ( ] ID : 13U02 | Receiver the message :(null) - ERROR(2)!!!
```

Τα μηνύματα με κάποιο σφάλμα έχουν “:(“ και τα αδιάφορα έχουν “:|” στην αρχή τους.



User Client

Στην αρχή όταν ανοίγει κάποιος χρήστης, το User Client πρόγραμμα για να κάνει τις όποιες απομακρυσμένες εποπτείες του, πρέπει να πιστοποιηθεί από το Server .

Γιαυτό συναντάει μια φόρμα για να γίνει το απαραίτητο login :



Όπως φαίνεται στην πάνω φωτογραφία πρέπει να συμπληρωθεί η ip και η πόρτα του Server όπως επίσης το username και password του χρήστη που έχει φτιαχτεί από την ιστοσελίδα .

Επίσης, υπάρχει σαν επιλογή πατώντας το κουμπί “Thesis” να δει ο χρήστης κάποιες πληροφορίες που αφορούν την πτυχιακή. Με το κεντρικό κουμπί (Που έχει ένα laptop μέσα) συνδέεται με το Server και με το κουμπί με το “X” δύναται να κλείσει το πρόγραμμα.

Ακόμη με τα ανάλογα checkbox που υπάρχουν δίνεται η επιλογή να “θυμάται” ο Userclient τα στοιχεία που έχει πληκτρολογήσει ο χρήστης, για την επόμενη φορά που θα “τρέξει” το πρόγραμμα.

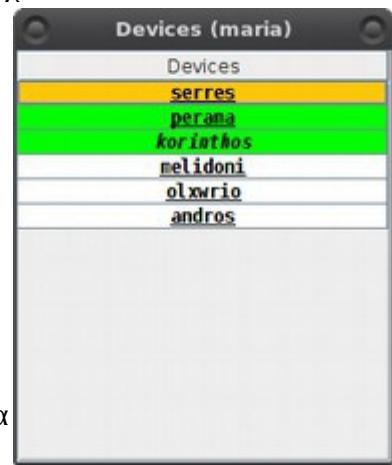
Μια εικόνα από το περιβάλλον μετά την σύνδεση με το server και έχοντας ανοίξει ο χρήστης “maria” δύο ζωντανές εποπτείες απομακρυσμένων συσκευών , είναι η παρακάτω :





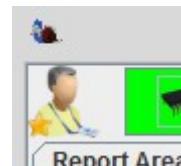
Τα κουμπιά που διακρίνονται στην προηγούμενη εικόνα στο πάνω μέρος, με τη σειρά κάνουν :

- Devices : Εμφανίζεται ένα παράθυρο (πάνω από όλα τα υπόλοιπα) που έχει όλα τα devices του χρήστη (όπως είναι το δίπλα) →
 - Με πράσινο είναι οι online συσκευές με το Server
 - Με πορτοκαλί είναι τα devices με ανοιχτές καρτέλες εποπτείας
 - Με άσπρο είναι οι συσκευές που δεν είναι online με το Server.
 - Στις συσκευές χωρίς υπογράμμιση έχουμε Readonly δικαιώματα .
 - Όποιες συσκευές είναι online είναι πιο ψηλά .
- Settings : Ανοίγει παράθυρο για ρυθμίσεις (Δεν έχει υλοποιηθεί και υπάρχει για μελλοντική χρήση όπως και στο Simulator).
- Console : Εμφανίζεται το περιεχόμενο του report Area σε όλες τις καρτέλες (Δηλαδή διάφορα μηνύματα αναφοράς) .
- Thesis : Εμφανίζεται ένα παράθυρο (που είναι διαρκώς πάνω από όλα τα υπόλοιπα όπως το παραθυράκι των devices) με πληροφορίες για την Πτυχιακή.
- Logout : Κάνει logout του χρήστη.



Το εικονίδιο που υπάρχει στο πάνω μέρος με το ανθρωπάκι είναι ανάλογα το **τύπο του χρήστη**.

- ◆ Αν πάει το ποντίκι του χρήστη πάνω από το ανθρωπάκι θα εμφανιστεί το username του.
(Είναι λίγο διαφορετικό το “παράθυρο” διότι τρέχει σε windows , όπως και η login φώτο)



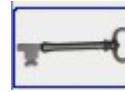
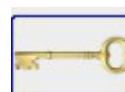
Μέσα στη κάθε καρτέλα εποπτείας υπάρχει στα αριστερά ένα δέντρο με όλες τις μονάδες και στο πάνω μέρος του δέντρου υπάρχουν 5 κουμπιά και στο κάτω 3 label με πληροφορίες της εποπτείας .
Τα κουμπιά της κάθε καρτέλας κάνουν με τη σειρά που βρίσκονται τα εξής :

- (1) Ξεκινάει και σταματάει την εποπτεία.
- (2) Αλλάζει την εικόνα του Background.
- (3) Φέρνει όλες τις μονάδες σε οπτικό πεδίο , σε περίπτωση που δεν φαίνονται μετά από resize.
- (4) Αν είναι “πατημένο” αποθηκεύονται οι ρυθμίσεις στον υπολογιστή του χρήστη .
(όπως οι θέσεις στην οθόνη , αν είναι κλειδωμένα , τα description κ.α των μονάδων)
- (5) Κλείνει τη καρτέλα .



Τα 3 label (με πληροφορίες) μας δείχνουν σε τη σειρά :

- (a) Μια εικόνα κλειδιού ανάλογα τα δικαιώματα προσπέλασης πάνω στην συσκευή



- (b) Ένα μήνυμα κατάστασης. Πχ πότε συνδέθηκε, αν άλλαξε κάτι , αν υπάρχει σφάλμα κλπ .
(c) Ένα εικονίδιο κατάστασης . Για να είναι πιο διακριτό και εμφανές στον χρήστη .





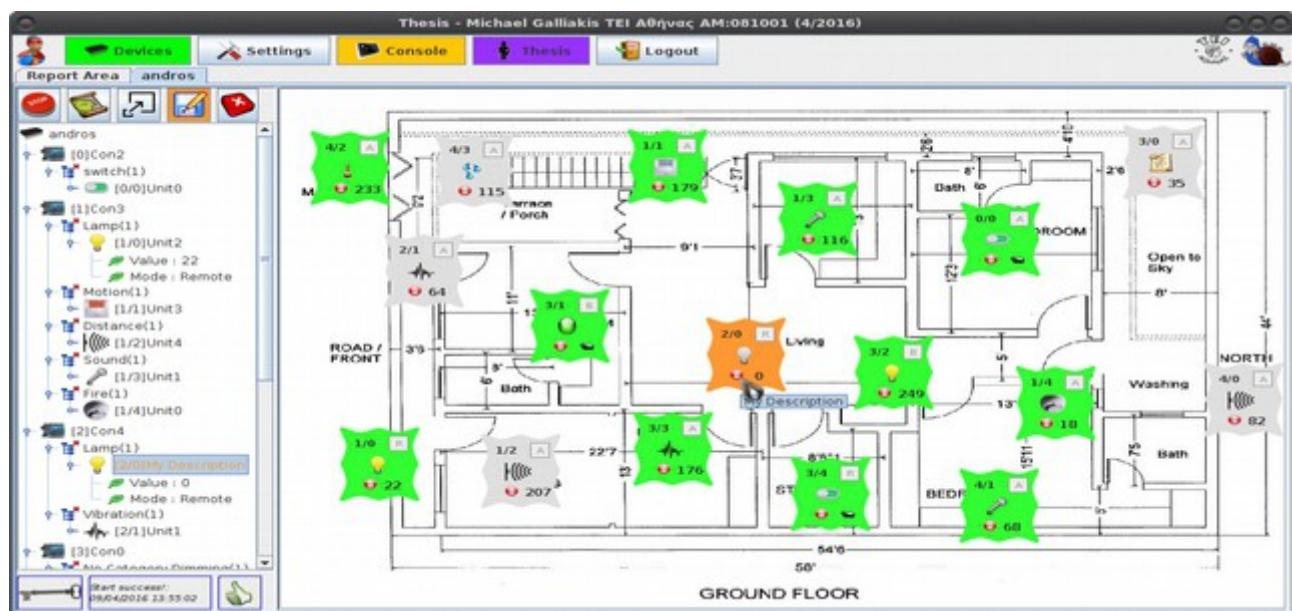
Τώρα, στο **δέντρο που υπάρχει στα αριστερά** διακρίνεται στην ρίζα του το όνομα της συσκευής. Μέσα του εμφανίζονται όλα τα arduino που έχει πάνω του η συσκευή.

Για κάθε arduino υπάρχουν σε κατηγορίες ανάλογα το τύπο όλες οι μονάδες με το όνομα τους αλλά και ένα διακριτικό συστήματος πχ 2/1 (που το φτιάχνει ο UserClient στη δημιουργία της καρτέλας) Μέσα στο κλαδί κάθε μονάδας έχει ακόμη 2 κλαδιά που το ένα είναι η τρέχων τιμή(Value) του και το άλλο η τρέχων λειτουργία(Mode) του .

Δεξιά στο χώρο που είναι οι μονάδες φαίνεται για τη κάθε μία, το διακριτικό συστήματος της (Σε tooltip είναι το όνομα της ή το description αν έχει αλλάξει), ένα εικονίδιο με βάση το τύπο της, ένα κουμπί για το mode, ένα διακριτικό για το αν έχει access πάνω της ο χρήστης και τέλος διακρίνεται η τιμή της (ή αν είναι 2 καταστάσεων ένα μικρό εικονίδιο ενός διακόπτη).

Επίσης, με βάση το χρώμα του (πράσινο, γκρι, πορτοκαλί) μπορεί να καταλάβει κανείς :

- Πράσινο : Ξεκλειδωθη μονάδα (Δηλαδή γίνεται να τραβηγχτεί (drag-drop) με σκοπό να μετακινηθεί μέσα στο χώρο έως ότου κλειδωθεί με το δεξί κλικ) →
- Γκρι : Κλειδωμένη μονάδα (Δεν γίνεται να την μετακινήσει κάποιος, μέχρι να την ξεκλειδώσει πρώτα με δεξί κλικ) →
- Πορτοκαλί : Είναι οι επιλεγμένες μονάδες (Επιλέγεται μια μονάδα είτε με το να την “κλικάρουμε” στην δεξιά μεριά είτε με το να την επιλέξουμε από το δέντρο). Επίσης, αν επιλεχθεί από το δέντρο ένας τύπος μονάδας ή ένας ελεγκτής θα γίνουν πορτοκαλί και όλες οι αντίστοιχες μονάδες στα δεξιά της εποπτείας .



Στην πάνω φωτογραφία διακρίνουμε ξανά κάποια από τα προαναφερθέντα πράγματα.

Για παράδειγμα, μεταξύ άλλων φαίνονται κλειδωμένες και ξεκλειδώτες μονάδες όπως επίσης και μια επιλεγμένη μονάδα (πορτοκαλί).

Ακόμη η φωτογραφία του background είναι αλλαγμένη (μέσω του 2ου κουμπιού καρτέλας) και το παραθυράκι των devices είναι κλειστό (Είναι και πράσινο το αντίστοιχο κουμπί για να είναι διακριτό και εμφανές στο χρήστη στην περίπτωση που θέλει να το ξανά-ανοιξει).

Επιπλέον, φαίνονται τα κλαδιά διάφορων μονάδων όπως και της επιλεγμένης λάμπας 2/0 με το αλλαγμένο description(“My Description”) που έχει value : 0 και Mode : Remote όπως παράλληλα διακρίνονται όλα αυτά και στο εικονίδιο μονάδας που βρίσκεται στα δεξιά .

Παράλληλα παρατηρείται ότι επειδή η συσκευή είναι ReadOnly όλες οι μονάδες ανεξαρτήτως φύσεως έχουν ένα διακριτικό (κόκκινο σημάδι με ένα χεράκι τύπου stop και όχι χέρι με δάκτυλο).



Αν επιθυμεί κάποιος χρήστης να αλλάξει το Description μίας μονάδας ή ελεγκτή πρέπει απλά να κάνει “δεξί κλικ” πάνω του μέσα στο δέντρο και να επιλέξει το “Change Description”



Ετσι θα εμφανιστεί ένα “παραθυράκι” για να το αλλάξει ο εκάστοτε χρήστης :



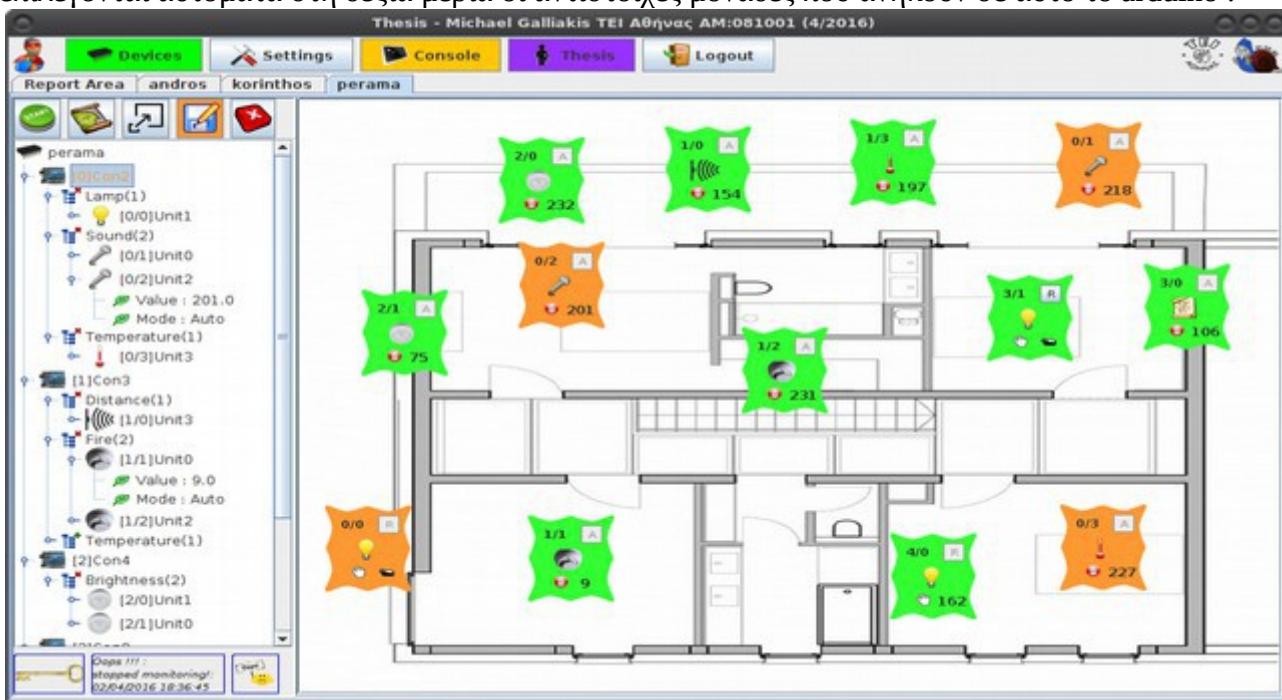
Υπάρχουν 2 τρόποι να αλλάξει κάποιος χρήστης τη τιμή κάποιας μονάδας .

- ✓ Ο ένας είναι να κάνει ο χρήστης “διπλό κλίκ” και κατευθείαν αν η μονάδα είναι δύο καταστάσεων (πχ διακόπτης ή λάμπα) να αλλάξει αυτόματα η τιμή της στην αντίθετη της (Πχ αν είναι ανοικτή η λάμπα θα κλείσει ή αν είναι κλειστή θα ανοίξει).
 - ✓ Ο δεύτερος είναι να κάνει ο χρήστης πάλι “διπλό κλίκ” και να εμφανιστεί ένα “παραθυράκι” που θα επιτρέπει στο χρήστη να αλλάξει την τιμή μιας μονάδας με την βοήθεια είτε μιας μπάρας είτε πληκτρολογώντας απευθείας την νέα τιμή (Που βέβαια θα έχει εύρος μεταξύ του min και του max της μονάδας).
- Η δεύτερη περίπτωση είναι για τις μονάδες που μπορούν να πάρουν διακριτές τιμές ενός καθορισμένου εύρους που έχουν από το Device Client .
(Πχ για μια λάμπα που γίνεται να καθοριστεί η φωτεινότητα της)



Στην από κάτω φωτογραφία φαίνεται μια περίπτωση όταν έχει για κάποιο λόγο χαθεί η σύνδεση την συσκευής που εποπτεύεται με το Server . Ετσι κλείνει η εποπτεία και αλλάζουν τα label κατάστασης ,όπως και το κουμπί που υπήρχε για να σταματήσει η εποπτεία τώρα πια δίνει την δυνατότητα στο user να ξανά-ξεκινήσει την εποπτεία.

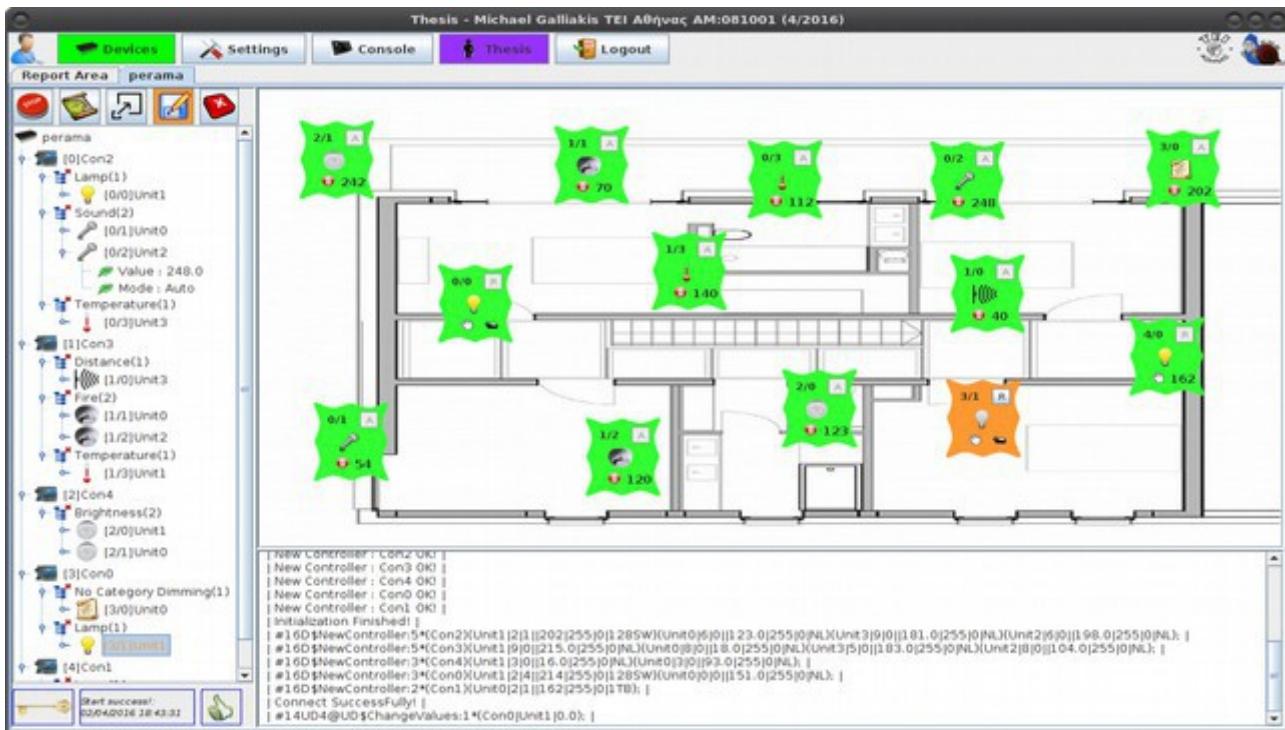
Επίσης, φαίνεται μια περίπτωση που ο χρήστης ενώ έχει επιλέξει έναν ελεγκτή από το δέντρο , επιλέγονται αυτόματα στη δεξιά μεριά οι αντίστοιχες μονάδες που ανήκουν σε αυτό το arduino .



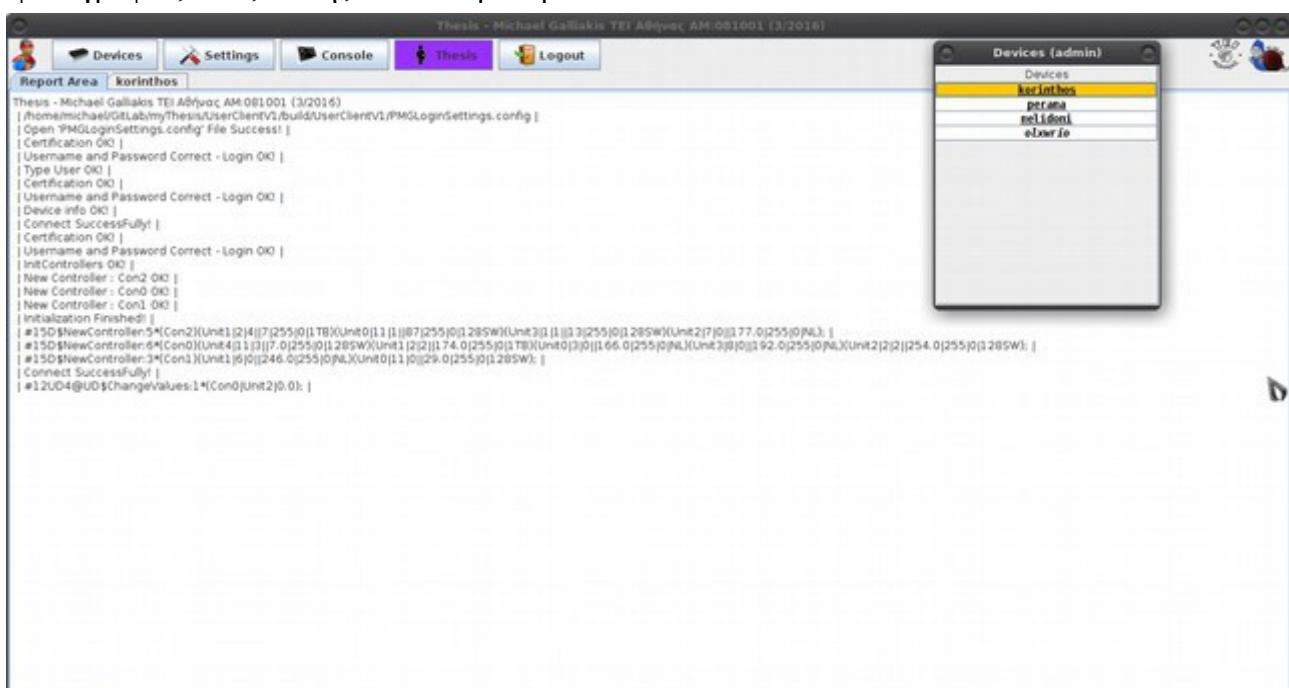


Στην από κάτω φωτογραφία φαίνεται ο UserClient να έχει ανοικτή τη κονσόλα μηνυμάτων αναφοράς (δεν είναι πια πορτοκαλί το κουμπί) σε όλες τις καρτέλες (που στην περίπτωση μας υπάρχει μόνο μια ανοικτή).

Ακόμη διακρίνονται κάποια μηνύματα που τυπώνει ο User Client για ενημέρωση του χρήστη όπως και κάποια μηνύματα (με βάση το πρωτόκολλο) που έλαβε ή έστειλε από και προς το Server.



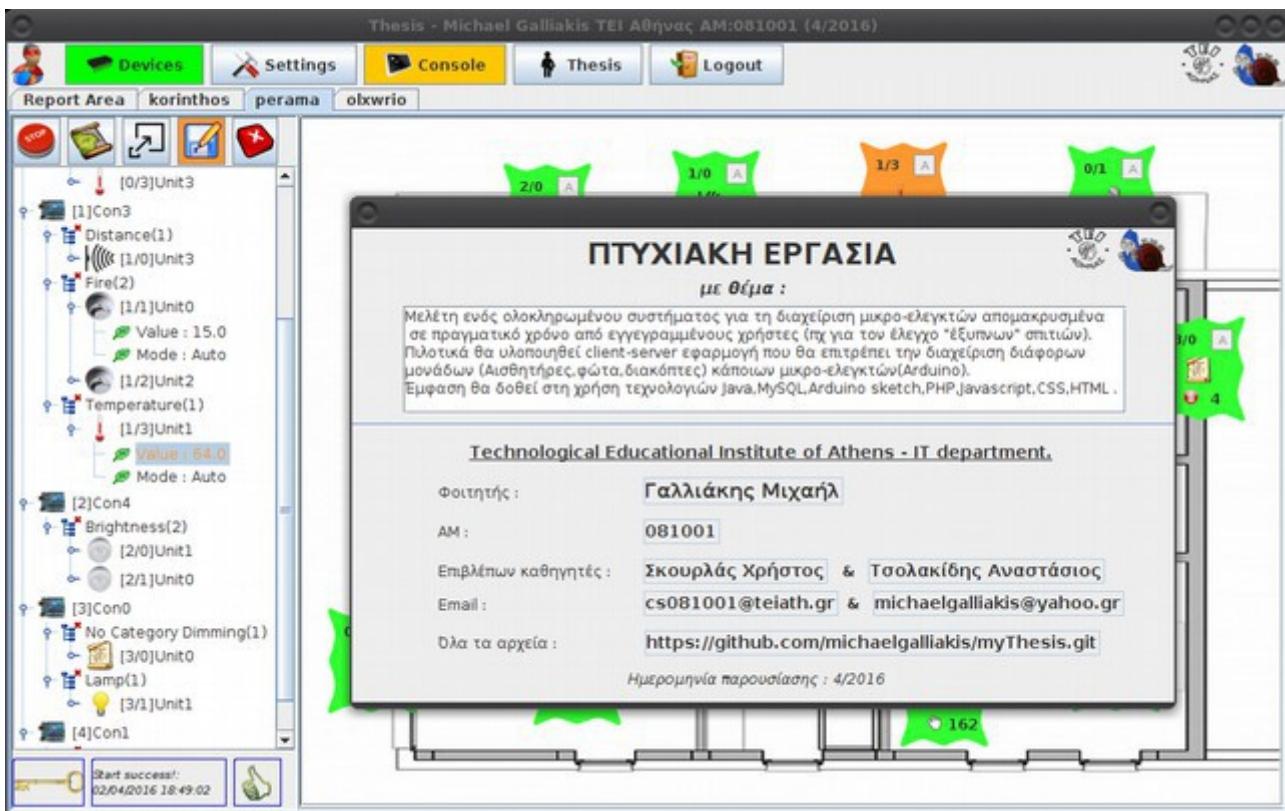
Στην κάτω φωτογραφία απλά φαίνεται η καρτέλα “Report Area” της περίπτωσης της από πάνω φωτογραφίας όπως επίσης και το παραθυράκι των devices .



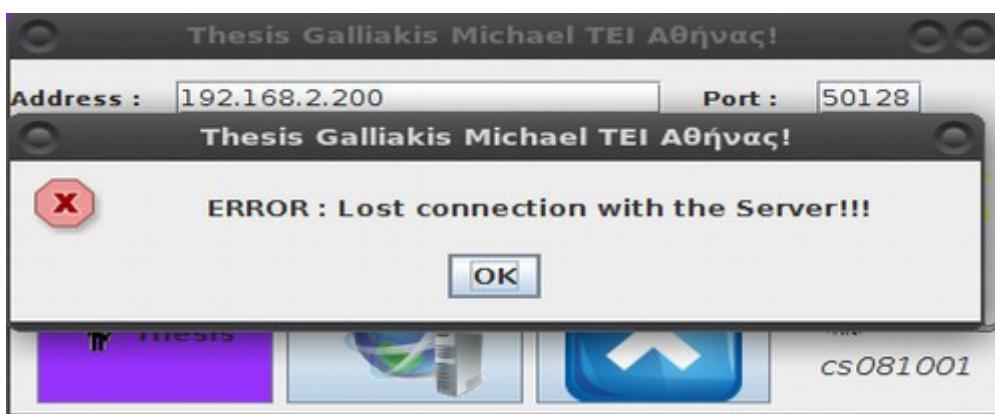


Ας παρουσιαστούν και 2 screenshot από δύο μόνο ενδεικτικές περιπτώσεις χρήσης του User Client .

Το πρώτο είναι όταν ο χρήστης έχει πατήσει το κουμπί “thesis” (είτε από το main παράθυρο είτε και από το παράθυρο του login) για να δει κάποιες πληροφορίες που αφορούν την πτυχιακή . Φαίνεται πια ότι το εν λόγω κουμπί δεν είναι πια μοβ μέχρι να κλείσει ο χρήστης το παράθυρο.



Το δεύτερο είναι απλά από την περίπτωση που για κάποιο λόγο χαθεί η σύνδεση με το server (πχ στην οποιαδήποτε αδυναμία επικοινωνίας μέσω δικτύου). Έτσι γίνεται αυτόματο logout του χρήστη και εμφανίζεται το ανάλογο μήνυμα .





Μεθοδολογία υλοποίησης του συστήματος

Ανάλυση, σχεδίαση και τρόπος σκέψης

Στην αρχή, αυτό που απασχόλησε ήταν να οριστικοποιηθεί τι ακριβώς θα κάνει το σύστημα (δημιουργήθηκαν σενάρια χρήσης) και από πόσα κομμάτια (εφαρμογές) πρέπει να αποτελείτε ώστε να μπορεί να λειτουργήσει.

Στην συνέχεια αναζητήθηκαν διάφορες τεχνολογίες πληροφορικής ώστε να βρεθούν τα πιο κατάλληλα εργαλεία (γλώσσες προγραμματισμού) που θα υλοποιούσαν το σύστημα.

Έτσι, επιλέχθηκε η Java διότι είναι μια γλώσσα προγραμματισμού που χρησιμοποιείται ευρέως (επομένως θα ήταν καλό μια εμπειρία και λίγη γνώση πάνω σε αυτή), δουλεύει ανεξάρτητα λειτουργικού συστήματος (επομένως παίζει σε Windows, Linux, MacOS και γενικά σε όποιο OS μπορεί να τρέξει Java) και έχει πολλά έτοιμα “εργαλεία” (βιβλιοθήκες) για να κάνεις ότι θέλεις. Επίσης, για την ιστοσελίδα επιλέχθηκε η PHP που είναι ιδανική για website εφαρμογές και παρέχει ευκολίες προγραμματισμού αλλά και διαχείρισης βάσεων δεδομένων. Μαζί με την PHP αποφασίστηκε να γραφόταν και λίγος κώδικας javascript για καλύτερη αλληλεπίδραση του χρήστη με το site αλλά και λίγο CSS για να είναι πιο όμορφο το περιβάλλον, πιο φιλικό και γνώριμο προς τον χρήστη. Επιπλέον, θα έπρεπε να χρησιμοποιηθεί και η HTML όπως και σε κάθε ιστοσελίδα. Ακόμη, θα γινόταν χρήση της Wiring γλώσσας για το προγραμματισμό των μικρο-ελεγκτών (arduino) που στην περίπτωση αυτή δεν θα γινόταν να επιλεχθεί και κάτι άλλο.

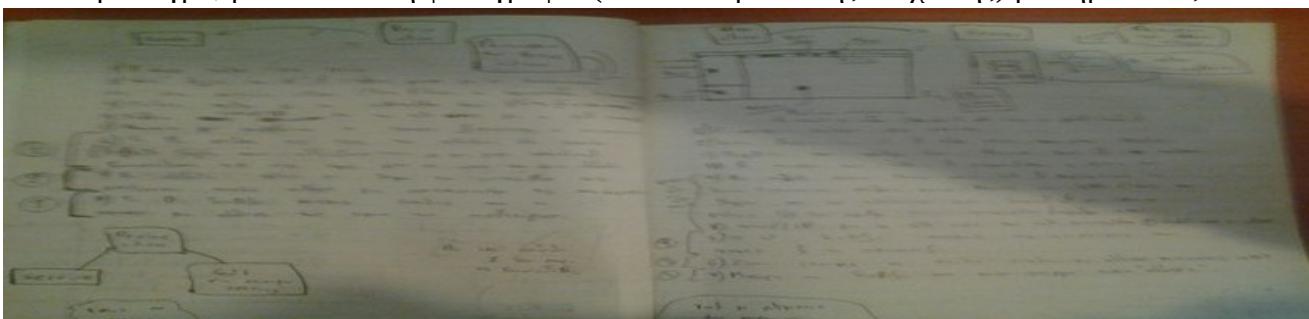
Όσον αφορά τη Database, επιλέχθηκε η MySQL επειδή είναι ένα καλό σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων και είναι και δωρεάν (όπως βέβαια δωρεάν είναι και όλες οι τεχνολογίες που αναφέρονται γενικά όπως και τα εργαλεία ανάπτυξης τους-πλατφόρμες). Οι εφαρμογές Java και η ιστοσελίδα θα αναπτύσσονταν με τη πλατφόρμα προγραμματισμού “Netbeans IDE”, τα arduino sketch θα γραφόντουσαν μέσα από το “Arduino IDE” και η διαχείριση της βάσης θα γινόταν από το “MySQL Workbench”.

Επιπλέον, θα γινόταν χρήση του προγράμματος Git μετά από ένα σημείο, όχι για την δυνατότητα να αναπτύσσετε ομαδικά η πτυχιακή (αφού ήταν ατομική) αλλά για να μην υπάρχει πρόβλημα στο να χαθούν οι εφαρμογές (από κάποιο λάθος-σφάλμα). Ακόμη βέβαια και για να υπάρχει η επιλογή να “γυρίζουμε πίσω” σε παλαιότερη “έκδοση” των εφαρμογών, στη περίπτωση που συμβεί κάποιο λάθος ή για να εντοπιστεί κάτι που έχει αλλαχθεί ή διαγράφει κλπ (χρήση του: gitlab.com private).

Σε επόμενη φάση, έγιναν κάποιες δοκιμές με demo κομμάτια για client-server εφαρμογή ώστε κατά κάποιο τρόπο να “σιγουρευτεί” ότι όλο αυτό το εγχείρημα θα ήταν εφικτό να υλοποιηθεί.

Αφότου λοιπόν ήταν εφικτό, έπρεπε να γίνει ανάλυση όλων των εφαρμογών αλλά και σχεδίαση όλων των λεπτομερειών τους. Έτσι δημιουργήθηκαν χειρόγραφα (και όχι με πιο επίσημο τρόπο, λόγω του ότι υπήρχε ένα μόνο άτομο για την πτυχιακή εργασία) διάφορες περιπτώσεις χρήσης με λόγια (όπως και interfaces, usecases, η βάση δεδομένων κα) και βρέθηκαν τρόποι με τους οποίους θα μπορούσαν να λειτουργήσουν όλα τα κομμάτια του συστήματος μαζί.

Για παράδειγμα, μια ενδεικτική φωτογραφία (από το τετράδιο της πτυχιακής) με σημειώσεις :





Πιο τεχνικά μέρη με λίγες επεξηγήσεις για ευκολότερη κατανόηση του κώδικα :

Ένα θέμα που απασχόλησε στην σχεδίαση της πτυχιακής ήταν να δημιουργηθεί ένα “πρωτόκολλο” επικοινωνίας για να γίνεται δομημένα και αυτοματοποιημένα η επικοινωνία με τη χρήση των socket (δηλαδή η επικοινωνία κυρίως μεταξύ των clients με το Server).

Όπως ήδη έχει αναφερθεί, με τα socket γίνεται να στηθεί ένα κανάλι επικοινωνίας μεταξύ δυο “σημείων” για να μπορούν να στέλνουν και να παραλαμβάνουν συμβολοσειρές μεταξύ τους δικτυακά αυτά τα δύο προγράμματα.

Έτσι κάθε αποστολή αποφασίστηκε να έχει την εξής μορφή :

#SystemID@Τύπος Αποστολέα(S,D,UD)\$Εντολή:Πλήθος παραμέτρων*(Παράμετρος0.0)
(Παράμετρος1.0|Παράμετρος1.1);

[Όπου Server , Device , UserDesktop ο τύπος αποστολέα]

Για παράδειγμα , για νέες τιμές ενός Device (που έχει ID στη βάση: 2 και systemID το 2D) στέλνει στο server την ακόλουθη εντολή :

#2D@D\$NewValues:5*(Con2[Unit0|190.0)(Con3[Unit0|50.0)(Con0[Unit1|153.0)(Con0[Unit2|55.0)
(Con1[Unit2|248.0);

Και πχ για αλλαγή των τιμών σε κάποια συσκευή από ένα χρήστη με ID βάσης 2 και ID συστήματος 2UD6 (το 6 έχει μπει για να είναι μοναδικό το ID διότι για κάθε socket του ίδιου χρήστη με το server[διαφορετική εποπτεία] έχουμε διαφορετικό ID...) στέλνεται το παρακάτω : #2UD6@UD\$ChangeValues:1*(Con3[Unit2|121);

- Επίσης, να αναφερθεί ότι για να γίνεται να υπάρχει ταυτόχρονη και ασύγχρονη επικοινωνία σε ένα socket πρέπει να υπάρχουν δυο thread ουσιαστικά από κάθε “πλευρά”, ένα για το read και ένα για το write ...

Ακόμη, έχουν αναπτυχθεί κλάσεις(κάποιες κοινές) που αναλαμβάνουν να κάνουν ομαδοποιημένα πράγματα, όπως πχ η ManageSocketMessage,ManageSocket,ManageDevice,Device,Tools κα .

Ενδεικτικά (επειδή κολλάει και με το παραπάνω) η ManageSocketMessage έχει μια συνάρτηση reload() που παίρνει μια συμβολοσειρά (όπως τις παραπάνω του πρωτοκόλλου) σαν παράμετρο και στη συνέχεια μπορεί κάποιος με διάφορες συναρτήσεις να πάρει στοχευμένα αυτό που θέλεις .

Δηλαδή πχ το getCommand(),getSumParameters(),isCorrect(),getMessage(),
ArrayList<ArrayList<String>>getParameters() κλπ .

Επίσης, σου παρέχει εργαλεία για να “κτίσεις” ένα νέο μήνυμα πολύ εύκολα με τη βοήθεια static συναρτήσεων όπως το newMessage() που παίρνει δεδομένα σαν παραμέτρους, για να είναι πιο απλή η δημιουργία και με λιγότερα συντακτικά λάθη .

- Με βάση το “πρωτόκολλο” είναι όλα τα μηνύματα που στέλνονται από την αρχή μέχρι το τέλος μιας socket σύνδεσης (Διάφορα command είναι : Cetification,Login, Answer,ChangeValues,ChangeModes,BringMeDevice,GetDevicesInfo κα).

Ακόμη να αναφερθεί ενδεικτικά ότι υπάρχει μια κλάση Device που ουσιαστικά διαχειρίζεται όλη τη συσκευή , δηλαδή όλους τους controlers και τις μονάδες τους .

Και για παράδειγμα (με τη βοήθεια HashMap) γίνεται πολύ εύκολα όταν παραλαμβάνει κάποιος ένα ChangeValues ή NewValues command (Οπως πχ είναι τα δύο πιο πάνω παραδείγματα) να πάρει τις παραμέτρους και να ενημερώσει πολύ εύκολα με τις νέες τιμές τις μονάδες που έγινε η αλλαγή .



Ενδεικτικό κομμάτι κώδικα java που έχει η κλάση Device :

```
private HashMap<String,HashMap<String,Unit>> allUnits ;  
private ManageSocketMessage msm ;  
  
.  
  
public void setValuesOfDevice(String valuesOfMessage)  
{  
    String conName,unitName,unitValue ;  
  
    msm.reload(valuesOfMessage);  
    for (int i = 0 ; i<msm.getParameters().size();i++)  
    {  
        conName = msm.getParameters().get(i).get(0) ;  
        unitName = msm.getParameters().get(i).get(1) ;  
        unitValue = msm.getParameters().get(i).get(2) ;  
  
        allUnits.get(conName).get(unitName).setValue(unitValue);  
    }  
}
```

To Unit είναι μια κλάση που έχει όλες τις απαραίτητες πληροφορίες και τα στοιχεία μιας μονάδας.

Άλλο ένα ενδεικτικό κομμάτι κώδικα από ένα “thread” που διαβάζει μηνύματα από κάποιο socket:

```
public void run() {  
    try {  
        String message ;  
        while (true) {  
            try {  
                message = in.readLine() ;  
            }  
            catch (Exception ex)  
            {  
                .  
                .  
                break ;  
            }  
            manSocMess.reload(message);  
  
            .  
  
            if (manSocMess.getCommand().equals("ERROR"))  
            {  
                .  
                .  
            }  
            .  
  
            else if (manSocMess.getCommand().equals("NewValues") || manSocMess.getCommand().equals("ChangeValues"))  
            {  
                device.setValuesOfDevice(manSocMess.getMessage());  
                //Tools.Debug.print("Read : " + manSocMess.getMessage());  
            }  
            .  
        }  
    }  
}
```

- ✓ Να αναφερθεί ότι λίγο-πολύ , όλο αυτό το σύστημα δουλεύει όπως και ένα chat . Δηλαδή υπάρχουν “κανάλια” σαν “ιδιωτικές” “συνομιλίες” και ανταλλάσσονται συμβολοσειρές με τη μορφή του “πρωτοκόλου” που δείχτηκε πιο πάνω στο κάθε “κανάλι”. Με άλλα λόγια όταν προκληθεί μια αλλαγή , είτε από το device είτε από κάποιον χρήστη που εποπτεύει το device στέλνετε (με σωστή μορφή του πρωτοκόλλου) σε όλους που είναι στο ίδιο “κανάλι” η νέα αλλαγή και ενημερώνεται κατάλληλα η κάθε “πλευρά”. Στο κάθε “κανάλι” υπάρχει ένα device(DeviceClient) και όλοι οι χρήστες (UserClient) που το εποπτεύουν, και η πρακτική διαχείριση όλων των “καναλιών” γίνεται από το Server .



Ενδεικτικά κάποιες βιβλιοθήκες της Java που χρησιμοποιήθηκαν :

```
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Iterator;
import java.util.Vector;
import java.io.File;
import java.util.Timer;
import java.util.TimerTask;
import javax.swing.JfileChooser;
import javax.xml.*;
import gnu.io.CommPortIdentifier;
import java.io.BufferedReader;
import java.io.PrintWriter;
import java.net.Socket;
import java.net.ServerSocket;
import gnu.io.SerialPort;
import gnu.io.SerialPortEvent;
import gnu.io.SerialPortEventListener;
import javax.swing.JoptionPane;
import javax.swing.JLabel;
import javax.swing.JTree;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.JDialog;
import javax.swing.JFrame;
import javax.swing.ImageIcon;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.eventMouseListener;
import java.awt.Component;
import java.awt.image.BufferedImage;
import java.awt.event.*;
import javax.imageio.ImageIO;
import org.w3c.dom.*;
import java.io.BufferedReader;
import java.awt.event.KeyListener;
import javax.swing.Jspinner;
import java.awt.Frame;
import javax.swing.JFrame;
import java.awt.Image;
import java.awt.Cursor;
import java.awt.Font;
import java.io.File;
import java.awt.Cursor;
```



Βάση δεδομένων(Database)

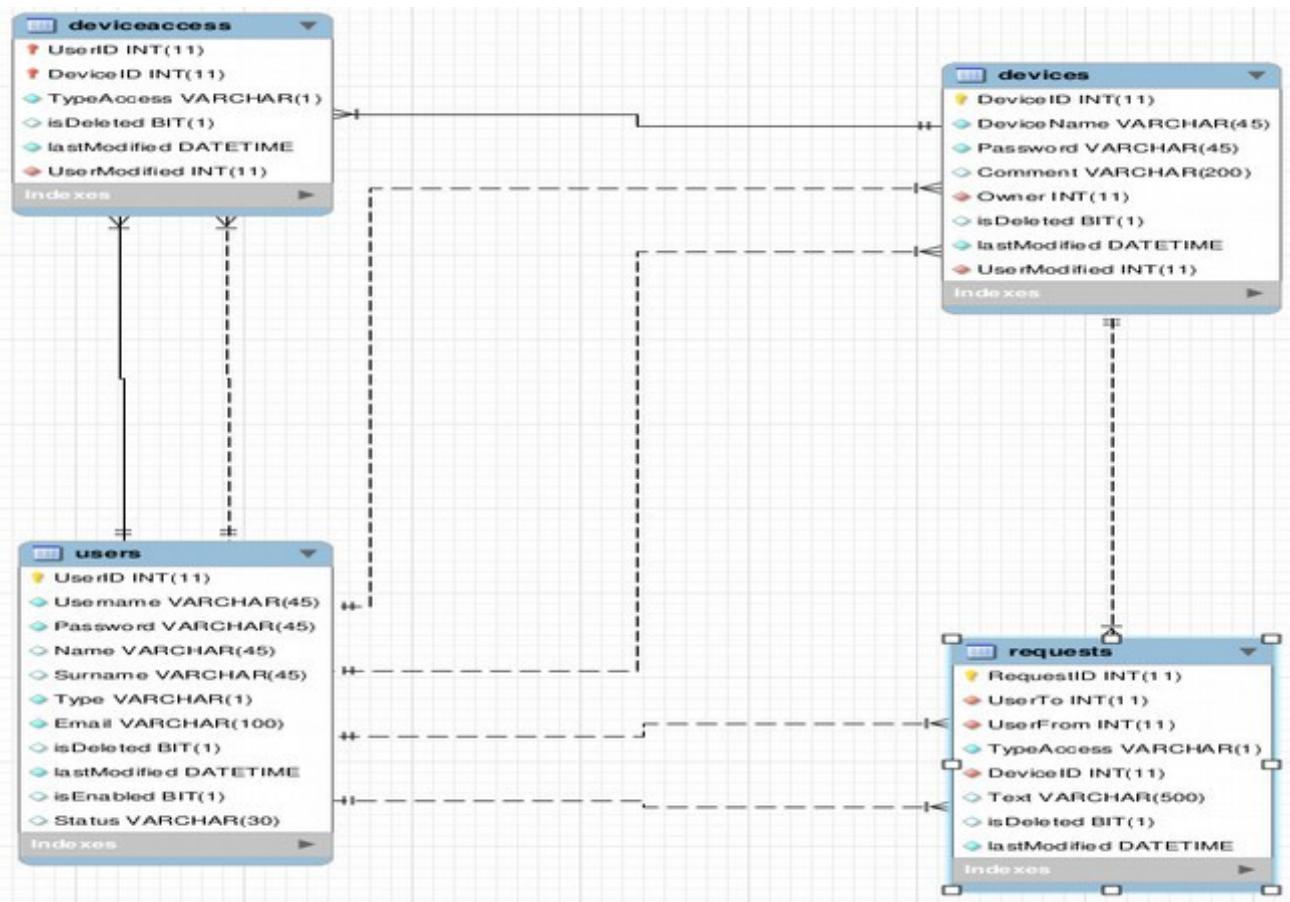
Η βάση έχει μόνο 4 πίνακες λογικά συσχετισμένους μεταξύ τους :

- users (Λογαριασμοί χρηστών)
- devices (Λογαριασμοί συσκευών)
- deviceaccess (Δικαιώματα των χρηστών πάνω σε συσκευές)
- requests (Αιτήσεις χριστών για να αποκτηθούν δικαιώματα πάνω σε συσκευές)

Στην παρούσα φάση τουλάχιστον δεν καταγράφονται δεδομένα από τις μονάδες ή πληροφορίες για το πότε και από που συνδέθηκε πχ κάποιος χρήστης κλπ .

Κάποιος χρήστης μπορεί να πάρει μόνο “ζωντανά” πληροφορίες από μια συσκευή που είναι συνδεδεμένος (μέσω του server), επομένως δεν υπάρχει η έννοια της οποιαδήποτε καταγραφής .
(Στα πλαίσια τουλάχιστον αυτής της Πτυχιακής)

Ένα σχεδιάγραμμα της βάσης δεδομένων από ένα εργαλείο του MySQL Workbench :



- Δηλαδή το σημαντικό από αυτό το σχεδιάγραμμα είναι ότι, ένας χρήστης μπορεί να έχει δικαιώματα σε πολλές συσκευές και μια συσκευή μπορεί να την χρησιμοποιούν πολλοί διαφορετικοί χρήστες.
- Όπως επίσης ένας χρήστης μπορεί να έχει “αιτήσεις” για να αποκτήσει δικαιώματα για πολλές συσκευές και μια συσκευή μπορεί να έχει “αιτήσεις” για δικαιώματα πάνω της από πολλούς χρήστες.



| # | Field | Schema | Table | Type | Character Set | Display Size | Precision | Scale |
|----|--------------|----------|-------|----------|---------------|--------------|-----------|-------|
| 1 | UserID | thesisv1 | users | INT | binary | 11 | 2 | 0 |
| 2 | Username | thesisv1 | users | VARCHAR | utf8 | 45 | 8 | 0 |
| 3 | Password | thesisv1 | users | VARCHAR | utf8 | 45 | 32 | 0 |
| 4 | Name | thesisv1 | users | VARCHAR | utf8 | 45 | 8 | 0 |
| 5 | Surname | thesisv1 | users | VARCHAR | utf8 | 45 | 8 | 0 |
| 6 | Type | thesisv1 | users | VARCHAR | utf8 | 1 | 1 | 0 |
| 7 | Email | thesisv1 | users | VARCHAR | utf8 | 100 | 15 | 0 |
| 8 | isDeleted | thesisv1 | users | BIT | binary | 1 | 1 | 0 |
| 9 | lastModified | thesisv1 | users | DATETIME | binary | 19 | 19 | 0 |
| 10 | isEnabled | thesisv1 | users | BIT | binary | 1 | 1 | 0 |
| 11 | Status | thesisv1 | users | VARCHAR | utf8 | 30 | 0 | 0 |

| MariaDB [thesisv1]> SELECT * FROM users ; | | | | | | | | | | | |
|-------------------------------------------|----------|----------------------------------|----------|----------|---------|----------|-----------------|-----------|---------------------|-----------|--------|
| # | UserID | Username | Password | Name | Surname | Type | Email | isDeleted | lastModified | isEnabled | Status |
| 12 | admin | 21232f297a57a5a743894a8e4a801fc3 | admin | admin | S | admin | admin@yahoo.gr | 0 | 0000-00-00 00:00:00 | 0 | NULL |
| 13 | maria | 263bce650e68ab4e23f2826376009fa5 | dsdfsd | maria | V | maria | maria@mpg.gr | 0 | 0000-00-00 00:00:00 | 0 | NULL |
| 14 | georgios | 0ff3023d92zeed7fd207ca9b8359725f | georgios | georgios | U | georgios | georgios@mpg.gr | 0 | 0000-00-00 00:00:00 | 0 | NULL |
| 15 | mikeo | 634379c32375875c137409ee5fd3474 | mikeo | mikeoI | U | mikeo | mike@mike.gr | 0 | 0000-00-00 00:00:00 | 0 | NULL |

Σχόλια πάνω στο πίνακα των users:

- Πρωτεύων κλειδί είναι το UserID που είναι ένας αύξων αριθμός .
- Υπάρχουν 3 τύποι χρηστών (Users,VIP,System)
- Με isDeleted γίνεται λογική διαγραφή μιας εγγραφής , ώστε να μην διαγράφεται από τη βάση αν κάποιος χρήστης θέλει να διαγραφεί .
- Το isEnabled είναι για πιθανόν προσωρινή απενεργοποίηση , μπλοκάρισμα ή μέχρι να γίνει πιστοποίηση του email κα.Προς το παρόν δεν χρησιμοποιείται πρακτικά .
- Status είναι μια ετικέτα κατάστασης με λέξεις κλειδιά που τελικά δεν χρησιμοποιήθηκε .
- Τα password είναι κρυπτογραφημένα με τον αλγόριθμο MD5 και έχουν 32 χαρακτήρες ώστε αν διαρρεύσει το περιεχόμενο της βάσης να είναι δύσκολο να ανακτηθούν οι κωδικοί και άρα με αυτό το τρόπο υπάρχει μεγαλύτερη ασφάλεια στο συστήμα .

| # | Field | Schema | Table | Type | Character Set | Display Size | Precision | Scale |
|---|--------------|----------|---------|----------|---------------|--------------|-----------|-------|
| 1 | DeviceID | thesisv1 | devices | INT | binary | 11 | 2 | 0 |
| 2 | DeviceName | thesisv1 | devices | VARCHAR | utf8 | 45 | 10 | 0 |
| 3 | Password | thesisv1 | devices | VARCHAR | utf8 | 45 | 32 | 0 |
| 4 | Comment | thesisv1 | devices | VARCHAR | utf8 | 200 | 92 | 0 |
| 5 | Owner | thesisv1 | devices | INT | binary | 11 | 2 | 0 |
| 6 | isDeleted | thesisv1 | devices | BIT | binary | 1 | 1 | 0 |
| 7 | lastModified | thesisv1 | devices | DATETIME | binary | 19 | 19 | 0 |
| 8 | UserModified | thesisv1 | devices | INT | binary | 11 | 2 | 0 |

| MariaDB [thesisv1]> SELECT * FROM devices ; | | | | | | | | | | |
|---------------------------------------------|------------|-------------------------------------|----------------------|--------------------------------------------|-------|-----------|---------------------|--------------|--|--|
| # | DeviceID | DeviceName | Password | Comment | Owner | isDeleted | lastModified | UserModified | | |
| 15 | korinthos | 19aed06450eaaf9007cd86cf09420d970 | spiti stin korinthio | | 12 | 0 | 0000-00-00 00:00:00 | 13 | | |
| 16 | perama | 19daad597cc70883fbd2bd8209e0824ea | Yo spiti sto perama! | I-PiLi-TT-I-TB IfLi,Li_ IfPiLi=I-TB IdSNZI | 13 | 0 | 0000-00-00 00:00:00 | 13 | | |
| 17 | melidon | fc979f4bed9aa466cc2687a04a298ca | melidon | I-PiLi-TT-I-TB IfLi,Li_ IfPiLi=I-TB IdSNZI | 13 | 0 | 0000-00-00 00:00:00 | 13 | | |
| 18 | loukia | fb084493a7e56f13a13e3523aa5e686de | loukia | I-PiLi-TT-I-TB IfLi,Li_ IfPiLi=I-TB IdSNZI | 14 | 0 | 0000-00-00 00:00:00 | 13 | | |
| 19 | margarites | fbb84493a7e56f13a13e3523aa5e686de | margarites | I-PiLi-TT-I-TB IfLi,Li_ IfPiLi=I-TB IdSNZI | 14 | 0 | 0000-00-00 00:00:00 | 13 | | |
| 20 | kavala | cce3f2aae56cc7ad99e35aa5e686de1997f | kavala | I-PiLi-TT-I-TB IfLi,Li_ IfPiLi=I-TB IdSNZI | 12 | 0 | 0000-00-00 00:00:00 | 12 | | |
| 21 | andros | e7b0ef4fc7081c7c19ed3d09547da8 | andros | I-PiLi-TT-I-TB IfLi,Li_ IfPiLi=I-TB IdSNZI | 13 | 0 | 0000-00-00 00:00:00 | 13 | | |

Σχόλια πάνω στο πίνακα των devices:

- Πρωτεύων κλειδί είναι το DeviceID , αλλά θα μπορούσε λόγω του ότι είναι πάντα μοναδικό να είναι και το DeviceName . Προτιμήθηκε για λόγους οικονομίας byte επειδή είναι και σε άλλους πίνακες ξένο κλειδί (Κάτι παρόμοιο έγινε και στους users)
- Owner είναι ξένο κλειδί από τους Users που δείχνει τον ιδιοκτήτη ,δηλαδή ποιος δημιουργείσαι τη συσκευή.
- UserModified είναι πάλι ξένο κλειδί από τους Users που δείχνει ποιος χρήστης έκανε την τελευταία αλλαγή στην εγγραφή
- Στο lastModified καταγράφεται η χρονική στιγμή που γίνεται μια αλλαγή σε κάποια εγγραφή (Πρακτικά δεν χρησιμοποιήθηκε).
- Στο password και isDeleted υπάρχει η ίδια λογική με το πίνακα users .



| # | Field | Schema | Table | Type | Character Set | Display Size | Precision | Scale |
|---|--------------|----------|----------|----------|---------------|--------------|-----------|-------|
| 1 | RequestID | thesisv1 | requests | INT | binary | 11 | 1 | 0 |
| 2 | UserTo | thesisv1 | requests | INT | binary | 11 | 2 | 0 |
| 3 | UserFrom | thesisv1 | requests | INT | binary | 11 | 2 | 0 |
| 4 | TypeAccess | thesisv1 | requests | VARCHAR | utf8 | 1 | 1 | 0 |
| 5 | DeviceID | thesisv1 | requests | INT | binary | 11 | 2 | 0 |
| 6 | Text | thesisv1 | requests | VARCHAR | utf8 | 500 | 92 | 0 |
| 7 | isDeleted | thesisv1 | requests | BIT | binary | 1 | 1 | 0 |
| 8 | lastModified | thesisv1 | requests | DATETIME | binary | 19 | 19 | 0 |

```
MariaDB [thesisv1]> SELECT * FROM requests ;
+-----+-----+-----+-----+-----+
| RequestID | UserTo | UserFrom | TypeAccess | DeviceID | Text
+-----+-----+-----+-----+-----+
| 1 | 13 | 14 | F | 19 | You want to have this device ?
| 2 | 13 | 13 | F | 20 | I want have this device , i want have it
| 3 | 12 | 13 | F | 21 | You want to have this device ?
| 4 | 14 | 13 | A | 21 | You want to have this device ?
| 6 | 15 | 15 | R | 21 | I want have it , i want have it , i want have it , i want have it
+-----+-----+-----+-----+-----+
```

5 rows in set (0.00 sec)

Σχόλια πάνω στο πίνακα των requests:

- Πρωτεύων κλειδί είναι RequestID που είναι ένας αύξων αριθμός.
- Το UserTo είναι το ID του χρήστη στον οποίο απευθύνεται η αίτηση. Δηλαδή ποιος χρήστης θα πάρει δικαιώματα πάνω στην συσκευή .
- Το UserFrom είναι το ID του χρήστη που έκανε την αίτηση(Οι αιτήσεις δηλαδή μπορεί να είναι αμφίδρομες , για παράδειγμα να κάνει αίτηση κάποιος που θέλει να πάρει δικαιώματα ή να κάνει κάποιος πρόσκληση για να δεχθεί κάποιος άλλος να πάρει δικαιώματα ...)

Αιτήσεις δηλαδή μπορεί να έχουν και οι Admin-Owners από κάποιους χρήστες που δεν έχουν ακόμη δικαιώματα στην συσκευή αλλά και οι χρήστες (που δεν έχουν ακόμη δικαιώματα) μπορεί να έχουν αιτήσεις αν κάποιος admin-owner τους κάνει “πρόσκληση” για να αποκτήσουν δικαιώματα σε μια συσκευή .

Στην ιστοσελίδα ,όταν θα συνδέεται ένας χρήστης, και σε κάποια συσκευή του είναι Admin-Owner του έρχονται όλα τα request για αυτή την συσκευή για να εγκρίνει. Άλλα του έρχονται και όλα τα request που έχουν UserTo το ID του και είναι από άλλο χρήστη(UserFrom) και αφορούν κάποια άλλη συσκευή που μέχρι στιγμής ο συνδεδεμένος χρήστης δεν έχει δικαιώματα.

(Μπορεί να μην είναι η καλύτερη δυνατή προσέγγιση , αλλά φαίνεται να παίζει αυτός ο τρόπος)

- TypeAccess είναι ο τύπος των δικαιωμάτων (Admin,FullAccess,Readonly)
- DeviceID είναι το ID του συγκεκριμένου Device .
- Text είναι ένα σχόλιο-κείμενο στο request που θα δει αυτός που είναι να το εγκρίνει.
- Στο lastModified και isDeleted υπάρχει η ίδια λογική με τους προηγούμενους πίνακες.
- Το UserTo,UserFrom&DeviceID είναι ξένα κλειδιά και θα μπορούσαν να είναι ζπλο πρ. κλ.

| # | Field | Schema | Table | Type | Character Set | Display Size | Precision | Scale |
|---|--------------|----------|--------------|----------|---------------|--------------|-----------|-------|
| 1 | UserID | thesisv1 | deviceaccess | INT | binary | 11 | 2 | 0 |
| 2 | DeviceID | thesisv1 | deviceaccess | INT | binary | 11 | 2 | 0 |
| 3 | TypeAccess | thesisv1 | deviceaccess | VARCHAR | utf8 | 1 | 1 | 0 |
| 4 | isDeleted | thesisv1 | deviceaccess | BIT | binary | 1 | 1 | 0 |
| 5 | lastModified | thesisv1 | deviceaccess | DATETIME | binary | 19 | 19 | 0 |
| 6 | UserModified | thesisv1 | deviceaccess | INT | binary | 11 | 2 | 0 |

```
MariaDB [thesisv1]> SELECT * FROM deviceaccess ;
+-----+-----+-----+-----+-----+-----+
| UserID | DeviceID | TypeAccess | isDeleted | lastModified | UserModified |
+-----+-----+-----+-----+-----+-----+
| 12 | 16 | F | 0000-00-00 00:00:00 | 12 |
| 12 | 17 | A | 0000-00-00 00:00:00 | 12 |
| 12 | 18 | R | 0000-00-00 00:00:00 | 13 |
| 13 | 15 | A | 0000-00-00 00:00:00 | 12 |
| 13 | 18 | F | 0000-00-00 00:00:00 | 14 |
| 14 | 15 | R | 0000-00-00 00:00:00 | 12 |
| 14 | 16 | A | 0000-00-00 00:00:00 | 14 |
| 14 | 17 | R | 0000-00-00 00:00:00 | 14 |
+-----+-----+-----+-----+-----+-----+
```

8 rows in set (0.00 sec)

Σχόλια πάνω στο πίνακα των deviceaccess:

- Το UserID και DeviceID είναι διπλό πρωτεύων κλειδί.
- Το TypeAccess είναι ο τύπος των δικαιωμάτων (A,R,F) .
- Στο lastModified,isDeleted&UserModified υπάρχει η ίδια λογική με τους προηγ.



Device Client (Simulator)

Όπως έχει αναφερθεί, ο προσομοιωτής του DeviceClient έχει σκοπό να παριστά μια εικονική συσκευή ώστε να γίνεται να πραγματοποιηθούν δοκιμές στο υπόλοιπο σύστημα, χωρίς την χρήση διάφορων πιθανόν arduino .

Πιο συγκεκριμένα έχει αναπτυχθεί μια διεπαφή(όπως φαίνεται και στην ενότητα της παρουσίασης) ώστε κάποιος χρήστης να μπορεί με γραφικό τρόπο να δημιουργεί εικονικούς μικρο-ελεγκτές και εικονικές μονάδες πάνω σε αυτούς . Επίσης, έχουν δημιουργηθεί κάποιες συναρτήσεις – “static μέθοδοι” που δημιουργούν τυχαίες μεταβολές στις τιμές(value) των μονάδων παίρνοντας υπόψιν τον τύπο και το mode τους .

Μια μικρή περιγραφή για το τεχνικό κομμάτι των βασικότερων λειτουργιών του Simulator:

Το πρόγραμμα ξεκινάει από τη κλάση **Fmain** που ουσιαστικά εμφανίζει στο χρήστη το κατάλληλο περιβάλλον ώστε να μπορέσει να δημιουργήσει ο εκάστοτε user ένα σενάριο μιας εικονικής συσκευής . Για κάθε νέα μονάδα ή αλλαγή/εμφάνιση κάποιας μονάδας εμφανίζεται ένα παραθυράκι που πρακτικά είναι ένας αντικείμενο της κλάσης **DnewUnit** . Η όλη διαχείριση της εικονικής συσκευής γίνεται με τη βοήθεια της **Device** κλάσης . Επίσης, το παράθυρο που έχει κάποιες πληροφορίες που αφορούν την πτυχιακή εργασία, είναι ουσιαστικά ένα αντικείμενο της κλάσης **Dthesis** . Όσο για την αμφίδρομη επικοινωνία της εφαρμογής με το Server, εκτελείτε παράλληλα με την κανονική ροή, ένα αντικείμενο της κλάσης **ListenThread** .

Όπως και σε κάθε άλλη Java εφαρμογή της πτυχιακής εργασίας, υπάρχουν οι βοηθητικές κλάσεις **ManageSocket** (για καλύτερη διαχείριση του socket) και **ManageSocketMessage**(για καλύτερη διαχείριση των μηνυμάτων πρωτοκόλλου) . Βέβαια, δεν λείπουν και οι κλάσεις **Tools** και **Globals** που περιέχουν static συναρτήσεις και μεταβλητές σύμφωνα με τις συγκεκριμένες ανάγκες της εφαρμογής που μεταξύ άλλων είναι και διάφορες random συναρτήσεις .

Επειδή είναι δύσκολο να γίνει πλήρη αναφορά για όλες τις τεχνικές λεπτομέρειες που αφορούν την υλοποίηση της εφαρμογής μέσα από αυτό το pdf αρχείο , **μπορεί ο αναγνώστης να βρει όλο το κώδικα της εφαρμογής στο Github** . Εκεί έχει γραφτεί πλήρη τεκμηρίωση του κώδικα και υπάρχει και το αντίστοιχο Javadoc αρχείο (html σελίδες) του project , όπου συνολικά τα σχόλια μαζί με το κώδικα είναι πάνω από 4000 γραμμές ώστε να δύναται κάποιος να καταλάβει εύκολα πως “λειτουργεί”.

Μια φωτογραφία από το Javadoc του DeviceClientSimulatorV1 :

| Class | Description |
|---------------------|---------------------------------------------------------------------------------------------------------------------------|
| Device | Κλάση που μας βοηθάει να κάνουμε καλύτερη διαχείριση του περιεχομένου μιας συσκευής . |
| DnewUnit | Κλάση που μέσω αυτής μπορεί κάποιος χρήστης με γραφικό περιβάλλον να διαχειριστεύει(νέο ή αλλαγή) ένα Unit του Simulator. |
| Dthesis | Κλάση που ουσιαστικά είναι μια φόρμα που έχει κάποιες πληροφορίες σχετικά με την Πτυχιακή . |
| Fmain | Η βασική κλάση από την οποία ξεκινάει το project και που πρακτικά είναι το αρχικό και κύριο παράθυρο. |
| Globals | Κλάση που έχει μέσα Static μεταβλητές και συναρτήσεις με σκοπό να χρησιμοποιούνται από παντού μέσα στο project . |
| ListenThread | Κλάση που αναλαμβάνει να διαβάζει συνεχώς ότι του στέλνει ο Server μέσα από το αινιγκτό socket . |
| ManageSocket | Κλάση για να διαχειρίζονται ένα socket και να έχουμε όλα τα απαραίτητα στοιχεία που χρειαζόμαστε μαζί . |
| ManageSocketMessage | Κλάση που διαχειρίζεται εξ ολοκλήρου τα διάφορα μηνύματα που μεταφέρονται μέσα από τα socket . |
| Tools | Κλάση που έχει μέσα Static συναρτήσεις - εργαλεία με σκοπό να χρησιμοποιούνται από παντού μέσα στο project . |
| Tools.Debug | Κλάση που βοηθάει στο Debug . |



Μια φωτογραφία από το *Gitlab* όπου εμφανίζεται ενδεικτικά η αρχή της Fmain κλάσης:

DeviceClientSimulatorV1/src/DeviceClientV1/Main.java · master · Michael Galliakis / myThesis · GitLab · Mozilla Firefox

Αρχείο Επεξεργασία Προβολή Ιστορικό Σελίδωσεικές Εργαλεία Βοήθεια

DeviceClientSimulatorV1 · https://gitlab.com/michaelgallikas/myThesis/blob/master/DeviceClientV1/Main.java · Search

GitLab

Go to dashboard

Project

Activity

Files

Commits

Builds

Graphs

Milestones

Issues

Merge Requests

Members

Labels

Wiki

Forks

Settings

michaelgallikas

Michael Galliakis / myThesis · Files

```
32 import org.w3c.dom.Element;
33 import org.w3c.dom.NodeList;
34 import org.xml.sax.SAXException;
35
36 /**
37 * Ο προσωποποιητικός διαλόγος αυτού μορφής της πλευράς δεν έχει καπού
38 * αρδιότι για να δοκιμάσει το Device Client σε πραγματικές συνθήκες.
39 * Οπούτε αναγνωρίζεται ο Simulator για να μπορούν να σταθούν,
40 * προσφέρεται κατ' αναλογία κάπτα κομμάτια του συστήματος (Server και UserClient).
41 */
42
43 /**
44 * Η παρούσα λάθος από την οποία δεκτεύεται το πρότεινο και που προκύπτει είναι το αρχικό και κύριο παραθύρο.
45 * Μετά από αυτή την φάσην έκλεψε, πλευράς-ήλιντρο, μπορεί κανόνις χρήστης να προστατεύεται τον προσωποποιητικό με
46 * τους ελέγχους που δείχνει όπως ακούει και με τις πονάδες που θα έχει κατέθεταις.
47 * Και δερδάει σε κάποιαν φάση και ωραίο έχει καταγραφεί τα καταλόγου στοιχεία (IP, Νόμιμο Ναυπηγείο, Πασσώρα)
48 * μπορεί να συνδεθεί στον Server που λέγεται του ρόλο καίς εκονικού Device Client.
49 *
50 * Επιπλέον από τη φάσην υπάρχει η δυνατότητα να αποθηκεύεται κάποιος χρήστης της
51 * οποίος λαμβάνεται όχι από το εκονικό Device Client.
52 *
53 * ανοικτά εννοιώσεις όχι από πλέοντας για να ανταπομητρήσουνται αυτήν της όλη τη
54 * εκονική Device Client.
55 *
56 * Ακόμη ο χρήστης έχει την δυνατότητα να είναι συνδεδεμένος ο εκονικός Device Client
57 * στο server να αλλάζει την ιδιόνοτητα που υπάρχει για να προλαμβάνει κάποια
58 * εκείνηνη αλλαγή στον φανονομετρικό Device Client (Simulator-Προσωποποιητικής)
59 *
60 * Βέβαιο επιπλέον είναι ένα πρόγραμμα με γραφικό περιβάλλον έχει και σύν κομιτή
61 * για να μπορεί ο χρήστης να δει καποτες πληροφορίες που μάρονται την πινακίδα.
62 * @author Michael Galliakis
63 */
64
65 public class Main extends javax.swing.JFrame {
66     DefaultListModel listModelOfControllers ;
67     MyTableModel tableModelOfUnits ;
68
69     ManageSocket manSocket ;
70     ListenThread myThread = null ;
71
72     String controller ;
73     public Device device ;
74     ManageSocketMessage manSocMess = new ManageSocketMessage();
75
76 /**
77 * Καταχειρωτής της κάθησης που κάνει όλα τις απαραίτητες διαδικασίες για
78 * να προστατεύεται καταλλήλως το αρχικό και κύριο παραθύρο της φάσης.
```

Και μια φωτογραφία από το Netbeans όπου εμφανίζεται ενδεικτικά η αρχή της DnewUnit κλάσης:

DnewUnit.java - Editor

```
1 package DeviceClientV1;
2
3 import java.util.List;
4 import java.util.Vector;
5 import javax.swing.DefaultComboBoxModel;
6 import javax.swing.JOptionPane;
7
8 /**
9 * Klass has négy rész: névök kiválasztás, szövegök bevitel és vezérlés.
10 * Az összes funkcióval 4 ablakot fog Unit és Simulator.
11 * H közel felső részben a felhasználóhoz van JDialog .
12 * Balról a névök listájának elérésétől kezdődően van a vezérlés .
13 * Újratölthető .
14 * Author Michael Gallaike
15 */
16
17 public class DnewUnit extends javax.swing.JDialog {
18     Fmain myParent ;
19     boolean newOrEdit ;
20     String firstUnitName = "",firstUnitName2 = "";
21     int selectedRowIndex ;
22     DefaultComboBoxModel cbModelOfControllers ;
23
24 /**
25 * Képernyőre írja a névök listáját .
26 * Param parent H adóhoz az "edit" is működik az előző mezők .
27 * Param modal True az előzőn való kiválasztás onTop minden más ablaknál az panel mellett .
28 * Ablak False az előzőtől eltérően az előzőn való kiválasztáson kívül minden más ablaknak az előzőtől eltér .
29 */
30     public DnewUnit(java.awt.Frame parent, boolean modal) {
31         this(parent,modal,true) ;
32     }
33
34 /**
35 * Képernyőre írja a névök listáját .
36 * Param parent H adóhoz az "edit" is működik az előző mezők .
37 * Param modal True az előzőn való kiválasztás onTop minden más ablaknál az panel mellett .
38 * Ablak False az előzőtől eltérően az előzőn való kiválasztáson kívül minden más ablaknak az előzőtől eltér .
39 * Param _newOrEdit True az előzőn való kiválasztásnál val true False az előzőn való kiválasztásnál .
40 * Edit az az új névök Unit .
41 */
42     public DnewUnit(java.awt.Frame parent, boolean modal,boolean _newOrEdit) { //true : "new" And false : "Edit"
43         super(parent, modal);
44         initComponents();
45         newOrEdit = _newOrEdit;
46         this.setIconImage(Globals.biLogo);
47         this.setLocationRelativeTo(parent);
48         try
49         {
50             myParent = (Fmain)parent ;
51             if (myParent!=null)
52             {
53                 fillData();
54
55                 if (!newOrEdit)
56                 {
57                     setTitle("View/Edit Unit");
58                     fillvaluesFromSelectedRow();
59                 }
60             }
61         }
62         catch (Exception e)
63         {
64             JOptionPane.showMessageDialog(null,e);
65         }
66     }
67
68     private void fillData()
69     {
70         Vector v = myParent.getUnits();
71         cbModelOfControllers = new DefaultComboBoxModel(v);
72         cbController.setModel(cbModelOfControllers);
73     }
74
75     private void fillvaluesFromSelectedRow()
76     {
77         if (newOrEdit)
78         {
79             firstUnitName = cbController.getSelectedItem().toString();
80             firstUnitName2 = firstUnitName;
81         }
82     }
83
84     private void fillvaluesFromEdit()
85     {
86         if (!newOrEdit)
87         {
88             firstUnitName = tfName.getText();
89             firstUnitName2 = firstUnitName;
90         }
91     }
92
93     private void fillvaluesFromView()
94     {
95         if (!newOrEdit)
96         {
97             firstUnitName = cbController.getSelectedItem().toString();
98             firstUnitName2 = firstUnitName;
99         }
100    }
101
102    private void fillvaluesFromDelete()
103    {
104        if (!newOrEdit)
105        {
106            firstUnitName = cbController.getSelectedItem().toString();
107            firstUnitName2 = firstUnitName;
108        }
109    }
110
111    private void fillvaluesFromEditAndView()
112    {
113        if (!newOrEdit)
114        {
115            firstUnitName = tfName.getText();
116            firstUnitName2 = firstUnitName;
117        }
118    }
119
120    private void fillvaluesFromEditAndDelete()
121    {
122        if (!newOrEdit)
123        {
124            firstUnitName = cbController.getSelectedItem().toString();
125            firstUnitName2 = firstUnitName;
126        }
127    }
128
129    private void fillvaluesFromViewAndDelete()
130    {
131        if (!newOrEdit)
132        {
133            firstUnitName = tfName.getText();
134            firstUnitName2 = firstUnitName;
135        }
136    }
137
138    private void fillvaluesFromEditAndViewAndDelete()
139    {
140        if (!newOrEdit)
141        {
142            firstUnitName = tfName.getText();
143            firstUnitName2 = firstUnitName;
144        }
145    }
146
147    private void fillvaluesFromEditAndViewAndDeleteAndEdit()
148    {
149        if (!newOrEdit)
150        {
151            firstUnitName = tfName.getText();
152            firstUnitName2 = firstUnitName;
153        }
154    }
155
156    private void fillvaluesFromEditAndViewAndDeleteAndView()
157    {
158        if (!newOrEdit)
159        {
160            firstUnitName = cbController.getSelectedItem().toString();
161            firstUnitName2 = firstUnitName;
162        }
163    }
164
165    private void fillvaluesFromEditAndViewAndDeleteAndDelete()
166    {
167        if (!newOrEdit)
168        {
169            firstUnitName = tfName.getText();
170            firstUnitName2 = firstUnitName;
171        }
172    }
173
174    private void fillvaluesFromEditAndViewAndDeleteAndEditAndView()
175    {
176        if (!newOrEdit)
177        {
178            firstUnitName = tfName.getText();
179            firstUnitName2 = firstUnitName;
180        }
181    }
182
183    private void fillvaluesFromEditAndViewAndDeleteAndEditAndDelete()
184    {
185        if (!newOrEdit)
186        {
187            firstUnitName = cbController.getSelectedItem().toString();
188            firstUnitName2 = firstUnitName;
189        }
190    }
191
192    private void fillvaluesFromEditAndViewAndDeleteAndEditAndViewAndDelete()
193    {
194        if (!newOrEdit)
195        {
196            firstUnitName = tfName.getText();
197            firstUnitName2 = firstUnitName;
198        }
199    }
199 }
```



Device Client

Όπως έχει αναφερθεί, ο Device Client έχει σκοπό να είναι ο ενδιάμεσος μεταξύ των μικρο-ελεγκτών και του Server και πιο συγκεκριμένα, αναλαμβάνει να συγχρονίζει απόλυτα την επικοινωνία τους. Αυτό σημαίνει ότι κυρίως αυτό που κάνει, είναι να στέλνει τα μηνύματα που λαμβάνει από τους ελεγκτές στον Server όπως επίσης να ενημερώνει κατάλληλα και το αντίστοιχο arduino όταν έρθει κάποιο μήνυμα από τον εξυπηρετητή (πχ για αλλαγή τιμής) με τη βοήθεια μιας *hashMap* δομής.

Μια μικρή περιγραφή για το τεχνικό κομμάτι των βασικότερων λειτουργιών του Deviceclient: Το πρόγραμμα ξεκινάει μέσα από τη κλάση **RealDeviceClient** που ουσιαστικά πέρα από κάποιους έλεγχους, αρχικοποιήσεις και διάβασμα του αρχείου **xml(PMGTCD.C.config)** ρυθμίσεων, αναλαμβάνει να ξεκινήσει μια διαδικασία σύνδεσης με το server αλλά και μια διαδικασία αναγνώρισης, συγχρονισμού και επικοινωνίας με όλα τα διαθέσιμα arduino που υπάρχουν στον υπολογιστή που εκτελείτε η εφαρμογή. Για κάθε arduino υπάρχει κάποιο αντίστοιχο αντικείμενο της κλάσης **ReadArduino** ώστε να μπορεί να γίνει η όλη διαχείριση του. Όσο για την αμφίδρομη επικοινωνία της εφαρμογής με το Server, εκτελείτε παράλληλα με την κανονική ροή, ένα αντικείμενο της κλάσης **ListenThread**. Ακόμη υπάρχει ένα αντικείμενο της κλάσης **Device** για καλύτερη διαχείριση όλων των ελεγκτών και μονάδων της συσκευής. Όπως και σε κάθε άλλη Java εφαρμογή της πτυχιακής εργασίας, υπάρχουν οι βοηθητικές κλάσεις **ManageSocket** (για καλύτερη διαχείριση του socket) και **ManageSocketMessage** (για καλύτερη διαχείριση των μηνυμάτων πρωτοκόλλου). Βέβαια, δεν λείπουν και οι κλάσεις **Tools** και **Globals** που περιέχουν static συναρτήσεις και μεταβλητές σύμφωνα με τις συγκεκριμένες ανάγκες της εφαρμογής.

Επειδή είναι δύσκολο να γίνει πλήρη αναφορά για όλες τις τεχνικές λεπτομέρειες που αφορούν την υλοποίηση της εφαρμογής μέσα από αυτό το pdf αρχείο, **μπορεί ο αναγνώστης να βρει όλο το κώδικα της εφαρμογής στο Github**. Εκεί έχει γραφτεί πλήρη τεκμηρίωση του κώδικα και υπάρχει και το αντίστοιχο Javadoc αρχείο (html σελίδες) του project, όπου συνολικά τα σχόλια μαζί με το κώδικα είναι πάνω από 2200 γραμμές ώστε να δύναται κάποιος να καταλάβει εύκολα πως “λειτουργεί”.

Μια φωτογραφία από το Javadoc του DeviceClientV1 :

| Class | Description |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| Device | Κλάση που μας βοηθάει να κάνουμε καλύτερη διαχείριση του περιχωρίου μιας συσκευής. |
| Globals | Κλάση που έχει μέσα Static μεταβλητές και συναρτήσεις με σκοπό να χρησιμοποιούνται από παντού μέσα στο project. |
| ListenThread | Κλάση που αναλαμβάνει να διαβάζει συνεχώς ότι του στέλνει ο Server μέσα από το ανοικτό socket. |
| ManageSocket | Κλάση για να διαχειρίζεστε ένα socket και να έχουμε όλα τα απαραίτητα στοιχεία που χρειαζόμαστε μαζί. |
| ManageSocketMessage | Κλάση που διαχειρίζεται εξ ολοκλήρου τα διάφορα μηνύματα που μεταφέρουντε μέσα από τα socket. |
| ReadArduino | Κλάση που με λίγα λόγια αναλαμβάνει την διαχείριση της επικοινωνίας του προγράμματος μας με τα arduino που είναι συνδεδεμένα πάνω στον υπολογιστή. |
| RealDeviceClient | Κλάση από την οποία ξεκινάει ο Device Client. Ουσιαστικά ο Device Client είναι ο ενδιάμεσος μεταξύ του Server και των Arduino. |
| Tools | Κλάση που έχει μέσα Static συναρτήσεις - εργαλεία με σκοπό να χρησιμοποιούνται από παντού μέσα στο project. |
| Tools.Debug | Κλάση που βοηθάει στο Debug. |



Μια φωτογραφία από το *Gitlab* όπου εμφανίζεται ενδεικτικά η αρχή της *RealDeviceClient*:

Και μια φώτο από το Netbeans όπου εμφανίζεται ενδεικτικά η αρχή της ListenThread κλάσης :



Server

Όπως έχει αναφερθεί, ο Server έχει σκοπό να είναι ο ενδιάμεσος μεταξύ των DeviceClients και των UserClients και πιο συγκεκριμένα, μεταξύ άλλων αναλαμβάνει να δημιουργεί “κανάλια” όταν συνδέονται πάνω του κάποια DeviceClients και να προσθέτει τα socket των UserClients που είναι για εποπτεία στα αντίστοιχα “κανάλια” των συσκευών.

Πρακτικά αυτό γίνεται με το να δημιουργούνται τόσες λίστες socket όσες είναι οι online συσκευές. Έτσι γενικά θα μπορούσε να ειπωθεί, ότι όταν κάποιος Userclient (μέσω του socket του) στείλει ένα μήνυμα στο Server, αναλαμβάνει ο εξυπηρετητής να το στείλει στο αντίστοιχο DeviceClient αλλά και σε όλα τα socket (των userclients εκτός του αποστολέα) που έχει η λίστα για τη εκάστοτε συσκευή...

Επίσης, να αναφερθεί ότι εκτός από το αρχείο που διαβάζει ο Server (που μεταξύ άλλων έχει και πληροφορίες σύνδεσης του με τη βάση δεδομένων&πόρτα) γίνεται προαιρετικά να δοθούν δυο παράμετροι όταν κάποιος χρήστης εκτελέσει το πρόγραμμα από τη console (1η παράμετρος είναι η πόρτα που θα “τρέχει” και η 2η είναι ένας αριθμός που καθορίζει το τύπο των μηνυμάτων που θα εμφανίζονται...). Ακόμη, στην αρχή κάθε σύνδεσης socket ενός client με το server πρέπει να γίνει η απαραίτητη πιστοποίηση του client με βάση ένα κλειδί(σε μορφή string). Οπότε μπορούν να δημιουργήσουν συνδέσεις socket με το server μόνο οι σωστοί clients και όχι κακόβουλα λογισμικά.

Μια μικρή περιγραφή για το τεχνικό κομμάτι των βασικότερων λειτουργιών του Server :

Το πρόγραμμα ξεκινάει μέσα από τη κλάση **Server** που ουσιαστικά πέρα από κάποιους έλεγχους, αρχικοποιήσεις και διάβασμα του αρχείου xml(**PMGTCs.config**) ρυθμίσεων , ξεκινάει ένα **serverSocket** ώστε να ακούει σε μια συγκεκριμένη πόρτα σύμφωνα με τις επιλογές του αρχείου ρυθμίσεων . Σε επόμενη φάση όποτε συνδέθει κάποιο socket από κάποιον client στο server , δημιουργείται ένα αντικείμενο τύπου **ClientServiceThread** και ξεκινάει να τρέχει παράλληλα μια διαδικασία και η main ροή του προγράμματος συνεχίζει να “ακούει” για νέες συνδέσεις socket άλλων clients . Αυτό που πρακτικά κυρίως κάνει η **ClientServiceThread** είναι ανάλογα το τύπο του client από το οποίο προέρχεται το ανοιχτό socket να κάνει κάποιες συγκεκριμένες ενέργειες. Για παράδειγμα , αν το socket είναι από μια συσκευή δημιουργείται ένα αντικείμενο **ManageDevice** το οποίο αναλαμβάνει να δημιουργήσει ένα καινούριο “κανάλι” ώστε να μπορούν αργότερα να μπουν σε αυτό άλλα socket από διάφορους χρήστες ώστε να πραγματοποιούν ζωντανές εποπτείες . Ωστόσο , η **ClientServiceThread** ,αν το socket που έχει πάρει από τη κλάση **Server** προέρχεται από κάποιο UserClient , τότε σύμφωνα με την επιθυμία του συγκεκριμένου socket (1)εποπτεία συγκεκριμένης συσκευής,2)πληροφορίες για όλες οι συσκευές του χρήστη,3) πιστοποίηση λογαριασμού και επιστροφή τύπου χρήστη) αναλαμβάνει να κάνεις τις απαραίτητες ενέργειες και στο τέλος να κλείσει η παράλληλη διαδικασία . Όταν τα socket είναι κάποιας συσκευής, ανοιχτής εποπτείας ή πληροφοριών των διαθέσιμων συσκευών κάποιου userClient , δεν κλείνουν και δημιουργούνται-ξεκινάνε αντίστοιχα thread αντικείμενα της κλάσης **ListenFromTheClient** ώστε ο server να είναι συνεχώς έτοιμος να λαμβάνει μηνύματα από τους clients .Και βέβαια με αυτό τον τρόπο μπορεί εύκολα ο server μετά από κάποιο συμβάν να επικοινωνήσει άμεσα με οποιοδήποτε client.Επίσης, για κάθε socket υπάρχει ένα αντικείμενο τύπου **ManageSocket** ώστε να γίνεται καλύτερη διαχείριση του εκάστοτε socket . Ακόμη υπάρχουν τόσα αντικείμενα όσες είναι και οι συνδεδεμένες συσκευές, της βοηθητικής κλάσης διαχείρισης συσκευής (**Device**) . Επιπλέον , η κλάση **ManageSocketMessage** επιτρέπει να γίνεται καλύτερη διαχείριση όσο αφορά τα μηνύματα του “πρωτοκόλλου” του συστήματος. Μαζί βέβαια με όλα τα προηγούμενα υπάρχουν και οι κλάσεις **Globals** και **Tools** που έχουν διάφορες static συναρτήσεις και μεταβλητές ώστε να μπορεί να λειτουργήσει όλη η εφαρμογή .



Επειδή είναι δύσκολο να γίνει πλήρη αναφορά για όλες τις τεχνικές λεπτομέρειες που αφορούν την υλοποίηση της εφαρμογής μέσα από αυτό το pdf αρχείο , **μπορεί ο αναγνώστης να βρει όλο το κώδικα της εφαρμογής στο Github** . Εκεί έχει γραφτεί πλήρη τεκμηρίωση του κώδικα και υπάρχει και το αντίστοιχο Javadoc αρχείο (html σελίδες) του project , όπου συνολικά τα σχόλια μαζί με το κώδικα είναι πάνω από 2000 γραμμές ώστε να δύναται κάποιος να καταλάβει εύκολα πως “λειτουργεί”.

Μια φωτογραφία από το Javadoc του ServerV1 :

The screenshot shows the JavaDoc index.html page for the ServerV1 package. The left sidebar lists the available classes: ClientServiceThread, Device, Globals, ListenFromTheClient, ManageDevice, ManageSocket, ManageSocketMessage, Server, Tools, and Tools.Debug. The main content area is titled "Package ServerV1" and contains a "Class Summary" table. The table has two columns: "Class" and "Description". Each row corresponds to one of the listed classes, providing a brief description of its purpose.

| Class | Description |
|-------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ClientServiceThread | Κλάση που αναλαμβάνει ως εξωτερικότερο το εκάπιτον socket παράλογα με το τύπο εφαρμογής που το δημιουργήσε. |
| Device | Κλάση που μας διεύθυνε ως κάνουμε καλύτερη διαχείριση του περιεχομένου μιας συσκευής. |
| Globals | Κλάση που έχει μέσα Static μεταβλητές και συμπρέπει με σκοπό να χρησιμοποιούνται από πολλούς μέρους στο πρόγραμμα. |
| ListenFromTheClient | Κλάση που διαβάζει συνεχός μηνύματα από τους clients είτε αυτοί είναι User είτε Device. |
| ManageDevice | Είναι μια κλάση που μας βοηθά στο να διαχειρίζομε τέσσερα DeviceClient που είναι συνδεδεμένο στο server μας όπως και το UserClient που έχουμε να κάνουμε με τη συγκεκριμένη DeviceClient και είναι και αυτό ταυτόχρονα συνδεδεμένο στο Server μας. |
| ManageSocket | Κλάση για να διευχειρίζομετε ένα socket και να έρχουμε δόλια τα απορίτητα στοιχεία που γεμίζουν τα μέρη. |
| ManageSocketMessage | Κλάση που διαχειρίζεται εξ ολοκλήρου τα διάφορα μηνύματα που μεταφέρονται μέσω από τα sockets. |
| Server | Από την κλάση Server λειτουργεί και ανοίγει ο εξωτερικής μας. |
| Tools | Κλάση που έχει μέσα Static συναρτήσεις - εργαλεία με σκοπό να χρησιμοποιούνται από πολλούς μέρους στο πρόγραμμα. |
| Tools.Debug | Κλάση που βοηθά στο Debug. |

Και μια φωτογραφία από το *Gitlab* όπου εμφανίζεται ενδεικτικά η αρχή της Server κλάσης:

ServerV1/src/ServerV1/Serverjava · master · Michael Galliakis / myThesis · GitHub · Mozilla Firefox

https://gitlab.com/michaelgallikas/myThesis/blob/master/ServerV1/src/ServerV1/Serverjava

Michael Galliakis / myThesis - Files

Search

Go to dashboard

Project Project Activity Files Commits Builds Graphs Milestones Issues Merge Requests Members Labels Wiki Forks Settings

michaelgallikas

```
1 package ServerV1;
2
3 import java.net.ServerSocket;
4 import java.net.Socket;
5 import java.net.SocketException;
6 import java.sql.Connection;
7 import java.sql.DriverManager;
8 import java.sql.ResultSet;
9 import java.sql.SQLException;
```

/*
 * Από τη κλασή Server δικτύων και μας γερίνει σε κλημεντίνη που
 * Η κλασή Server (που από τη μάθηση) αρχίζει προσέδεση σε πορτακά με την πόστη λειτουργίας.
 * Έτσι αποκετάει όποια μανιτάρια είναι Server Socket περιφέρεια να δεχθειν μανιτάρια από άλλα socket φέρει.
 * Μάλλον παραδείξει πώς την κατανοώντας από socket παρακολουθεύεται κατ' ξεχωρία είναι έγκαιος (αντικείμενο) και
 * που αποκετάει σε γενικότερο το socket.
 * Ο Server μας μανιτάρια να περιφέρει μέσα socket για πελάτευτες μανιτάρια.
 * Επιγραφή: Michael Galliakis

*/

```
20 public class Server {
21
22     /* Η μάθη μεροδασ που δικτύωσε το project
23     * ΕΛΛΕΓΕΙΣ ΤΗΣ ΠΟΝΔΑΣ ΝΕ ΤΗΝ ΡΟΔΗ ΚΑΙ ΣΤΗ ΔΟΥΚΕΙΑ ΜΑΝΙΤΑΡΙΩΝ ΤΟ -socket Server που γίνεται σε αρχείο
24     * Δημόσιο για κάθε socket που συνδέεται στο server που μανιτάρια σε ένα thread για το δεξιόταρο της
25     * και συνδέεται σε περιφέρεια για νέας συνδέσεως άλλου socket .
26     * Θέση αρχής παραδούσα στον το τρεπούντα με κονσόλα τη διαδοχή από πορτακά για τη πόστη στην π.
```

```
27
28     public static void main(String[] args)
29     {
30         Globals.printInfo();
31         Tools.Debug.print("*****", Tools.HIGH);
32         ServerSocket m_ServerSocket = null;
33         Tools.Debug.print("Starting Server ...", Tools.HIGH);
34         if (checkDatabaseAndLoadConfig())
35         {
36             if (args.length > 0)
37                 if (Tools.isNumber(args[0]))
38                     Globals.serverPort = Integer.parseInt(args[0]);
39             if (args.length > 1)
40                 if (Tools.isNumber(args[1]))
41                     Globals.vinMessages = Integer.parseInt(args[1]);
42         }
43     }
44 }
```



UserClient

Όπως έχει αναφερθεί, ο UserClient έχει σκοπό να δίνει την δυνατότητα σε κάποιον χρήστη να μπορεί να εποπτεύσει και να διαχειριστεί απομακρυσμένα κάποιες συσκευές με γραφική διεπαφή. Να αναφερθεί ενδεικτικά ,ότι ένας UserClient για κάθε ανοικτή εποπτεία έχει και από ένα socket με το server όπως επίσης έχει και αντίστοιχα για κάθε socket τόσα thread για να μπορεί να διαβάζει συνεχώς και ασύγχρονα μηνύματα από τις συσκευές (ή & από άλλους userclients) μέσω του Server. Όμως εκτός από τις εποπτείες ,ένας UserClient έχει πάντα ένα socket (και thread) ανοιχτό για να μπορεί να ενημερώνεται από το server για αλλαγές στις καταστάσεις των devices του(up,down). Που σημαίνει ότι πχ αν ένας χρήστης δεν έχει ανοίξει καμία εποπτεία , θα υπάρχει 1 ανοιχτό socket από το login που έχει κάνει, και αν πχ έχει 2 εποπτείες τότε θα υπάρχουν 3 socket με το Server κλπ.

Μια μικρή περιγραφή για το τεχνικό κομμάτι των βασικότερων λειτουργιών του UserClient:

Το πρόγραμμα ξεκινάει από τη κλάση **FuserLogin** που ουσιαστικά δίνει την δυνατότητα σε κάποιο χρήστη να συμπληρώσει τα κατάλληλα στοιχεία σύνδεσης με το server . Παράλληλα βέβαια η **FuserLogin** είναι υπεύθυνη να κάνει κάποιους έλεγχους, αρχικοποίησεις και διάβασμα του αρχείου xml(**PMGLoginSettings.config**) εφόσον υπάρχει , το οποίο περιέχει τα στοιχεία σύνδεσης από την προηγούμενη εκτέλεση της εφαρμογής (άμα το είχε επιλέξει ο χρήστης) . Με το που γίνει η πιστοποίηση χρήστη δημιουργείτε ένα παραθυράκι με όλες τις διαθέσιμες για το χρήστη συσκευές (αντικείμενο της κλάσης **Ddevices**) όπως επίσης και ένα παράθυρο (αντικείμενο της κλάσης **Fmain**) το οποίο δίνει κάποιες δυνατότητες στον χρήστη . Για κάθε καρτέλα εποπτείας ουσιαστικά δημιουργείτε ένα αντικείμενο της κλάσης **Ptab** . Και κάθε αντικείμενο της **Ptab** έχει μέσα της στα δεξιά της , ένα αντικείμενο της **BackgroundPanel** το οποίο με τη σειρά του έχει μέσα του τόσα **UnitControl** όσες είναι και οι μονάδες της αντίστοιχης συσκευής στην οποία γίνεται εποπτεία . Τώρα , για την αλλαγή τιμής κάποιας μονάδας (που δεν είναι τύπου Switch) εμφανίζεται ένα panel που πρακτικά είναι ένα αντικείμενο της κλάσης **DfChangeValue** . Επίσης, όταν ο χρήστης θέλει να αλλάξει κάποιο “description” ουσιαστικά εμφανίζεται ένα panel ενός αντικειμένου της κλάσης **DfChangeText**. Όσο για την αμφίδρομη επικοινωνία της εφαρμογής με το Server, εκτελούνται παράλληλα με την κανονική ροή, τόσα αντικείμενα της κλάσης **ListenThread** όσα είναι και τα ανοιχτά socket. Για παράδειγμα , σε 2 εποπτείες τρέχουν παράλληλα με τη κανονική ροή και 3 αντικείμενα (ουσιαστικά το “σώμα” της μεθόδου run με βάση το περιεχόμενο που έχουν οι μεταβλητές του εκάστοτε αντικειμένου) της **ListenThread** (1 για τις πληροφορίες των devices που πρέπει να υπάρχει ακόμη και όταν δεν υφίσταται κάποια ανοιχτή εποπτεία και 2 για τις ανοιχτές εποπτείες). Επίσης, για τη διαχείριση όλων των συσκευών στις οποίες κάνει εποπτεία ο χρήστης υπάρχουν τόσα αντικείμενα της κλάσης **Device** όσες είναι και οι εποπτείες. Ακόμη , το παράθυρο που έχει κάποιες πληροφορίες που αφορούν την πτυχιακή εργασία, είναι ουσιαστικά ένα αντικείμενο της κλάσης **Dthesis** . Για τέλος να αναφερθεί ότι όπως και σε κάθε άλλη Java εφαρμογή της πτυχιακής εργασίας, υπάρχουν οι βοηθητικές κλάσεις **ManageSocket** (για καλύτερη διαχείριση των socket) και **ManageSocketMessage**(για καλύτερη διαχείριση των μηνυμάτων πρωτοκόλλου) . Βέβαια, δεν λείπουν και οι κλάσεις **Tools** και **Globals** όπως επίσης και η κλάση **ManageUserClient** (που σκοπό έχει να αποφορτίσει άλλες κλάσεις της εφαρμογής για καλύτερη διαχείριση), που περιέχουν static συναρτήσεις και μεταβλητές σύμφωνα με τις συγκεκριμένες ανάγκες της εφαρμογής.

Επειδή είναι δύσκολο να γίνει πλήρη αναφορά για όλες τις τεχνικές λεπτομέρειες που αφορούν την υλοποίηση της εφαρμογής μέσα από αυτό το pdf αρχείο , **μπορεί ο αναγνώστης να βρει όλο το κώδικα της εφαρμογής στο Github** . Εκεί έχει γραφτεί πλήρη τεκμηρίωση του κώδικα και υπάρχει και το αντίστοιχο Javadoc αρχείο (html σελίδες) του project , όπου συνολικά τα σχόλια μαζί με το κώδικα είναι πάνω από 6800 γραμμές ώστε να δύναται κάποιος να καταλάβει εύκολα πως “λειτουργεί”.



Μια φωτογραφία από το Javadoc του UserClientV1 :

The screenshot shows the Javadoc interface for the UserClientV1 project. The left sidebar lists all classes, and the main area displays the class hierarchy and descriptions. The classes and their descriptions are:

- BackgroundPanel**: Κλάση η οποία έχει κληρονομήσεις JPanel. Πρακτικά είναι το panel που βρίσκεται πίσω από τις εκάστοτε μονάδες μιας συσκευής, δηλαδή στο δεξιό μέρος μιας καρτέλας (Ptab) που έχει μέσα τα Units με πολύχρωμη μορφή.
- Ddevices**: Κλάση η οποία έχει κληρονομήσεις JPanel. Και ουσιαστικά είναι το παραθερόπικο που μέσω αυτού μπορεί να δει και να αποζηγεί για εποικεία κύκλων χρήστης τις συσκευές στις οποίες έχει διακόπωτα.
- Device**: Κλάση που μας βοηθεί με κάνουμε καλύτερη διαχείριση του περιεχομένου μιας συσκευής.
- DfChangeText**: Κλάση η οποία ουσιαστικά είναι το παραθερόπικο (Dialog) που εμφανίζεται για να δοθεί η δυνατότητα στο χρήστη να αλλάξει το Description περιγραφή κάποιας μονάδας ή έλεγχη.
- DfChangeValue**: Κλάση η οποία ουσιαστικά είναι το παραθερόπικο (Dialog) που εμφανίζεται για να δοθεί η δυνατότητα στο χρήστη να αλλάξει την τιμή κάποιας μονάδας με την βοήθεια μπάρας ή και εκδημ με το παραβονιακό τρόπο της απλής εποικυμός με γραφή.
- Dthesis**: Κλάση που ουσιαστικά είναι μια φόρμα που έχει κάποιες πληροφορίες σχετικά με την Πτυχιά.
- Fmain**: Κλάση που έχει κληρονομήσεις JFrame και που ουσιαστικά είναι το κύριο παράθυρο του User Client.
- FuserLogin**: Κλάση η οποία κληρονομήσει την JFrame από την οποία ξεκινάει ο UserClient.
- Globals**: Κλάση που έχει μία Static μεταβλήτης και συνηθίστηκε με σκοπό να χρησιμοποιούνται από ποικιλό μέρος στο project.
- ListenThread**: Κλάση που περιλαμβάνει μια δοθεῖσις συνεργίας όπου τον στέλνει στο Server μέσω από το πιοκτό socket.
- ManageSocket**: Κλάση για να διεγείρθαστε ένα socket και να έρχεται όλα τα απαραίτητα στοιχεία που χρειαζόνται μαζί.
- ManageSocketMessage**: Κλάση που διαχειρίζεται για να λειτουργεί υποπρωτκάπι προς διάφορες άλλες κλάσεις και πιο συγκεκριμένα για να μπορούν οι κλάσεις **Tools** και **DfChangeText** να εκτελέσουν διάφορος static συνθετήσεις της.
- Ptab**: Κλάση που έχει κληρονομήσεις JPanel και που ουσιαστικά είναι μια καρτέλα εποικείας κύκλων συσκευής μέσω στο κύριο παράθυρο του User Client (Fmain).
- Tools**: Κλάση που έχει μία Static συνθετήσεις - εργαλεία με σκοπό να χρησιμοποιούνται από ποικιλό μέρος στο project.
- Tools.Debug**: Κλάση που βοηθά στο Debug.
- UnitControl**: Κλάση η οποία έχει κληρονομήσεις JPanel και που πρακτικά είναι υπούργηνη να αναπαραστέται με γραφικό τρόπο μια μονάδα.

Και μια φωτογραφία από το Gitlab όπου εμφανίζεται ενδεικτικά η αρχή της Fmain κλάσης :

```

package UserClientV1;

import java.awt.Color;
import java.awt.Component;
import java.awt.Frame;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.UIManager;
import static UserClientV1.ManageUserClient.initComponents(JFrame);

public class Fmain extends javax.swing.JFrame {
    public static boolean isConsoleReportEnable = false;

    /**
     * Καταπονητής που προετοιμάζει καταλλήλως το αντικείμενο έτσι ώστε μετά
     * στην δημιουργία του , δηλαδή πρωτικό κατώς διαμορφώνει το εκνούδιο
     * του χρήστη με ραφή για τόπο του , δασές των τούρτζων το εκνούδιο το ονόμα
     * του χρήστη και τέλιν ανατρέγει μέσα καρτέλα που έχει το "Ναρκεί Άγρα"
     * που φανεται και στις υπόλοιπες καρτέλες σε όποιον χρήστη έχει επιλεξει
     * να φανεται η "Οφελεί" σε όλες τις καρτέλες .
     * Δρασας πολυς Απ'Τρια ισημερινα στις παραπάνω κατηγοριες δεναιτι μηριτσιες
     * και ότι κανεις όλες τις εκενερευει υπο με συνδεση με το πονεγια . Δηλαδή
     * πρωτικά σα προκομψεις ενοι μονα προ τη φόρμα της συνδεσης του χρήστη (Εθίση FusorLogos)
     * και χρησιμοποιεταιστην περιττων αναπτησης της εφαρμογης για να μην δεναινειση συνεχεια
     * δυναμικ χρήστη , κωδικο , πόρτα και ιερ και να δεκτειση η εφαρμογη μιας πιο γρήγορα.
     * Ετσι κανονικα διαρκεια της εφαρμογης σταν να την τρεχειση "τελικος" χρήστης
     * το πονειση πρεμεν να είναι πονειση ταξιδι
     */
    public Fmain(boolean debug) {
        initComponents();
        setTitle("Globals.FmainTitle");
        if (debug)
    }
}

```



Ιστοσελίδα (Website)

Η ιστοσελίδα πρακτικά έχει 25 αρχεία **php** (που τρέχουν στον apache HTTP server) από τα οποία τα 18 έχουν μέσα τους και κώδικα **HTML**. Επίσης, υπάρχει 1 αρχείο **CSS** (για το design των σελίδων) και 1 αρχείο με **Javascript** (για διάφορους “ελέγχους” που τρέχουν στο browser) όπου χρησιμοποιούνται από τα php αρχεία.

Ακόμη υπάρχουν και κάποια αρχεία zip ώστε να μπορεί κάποιος χρήστης να τα “κατεβάσει”.

Επιπλέον, υπάρχει ένα αρχείο pdf (αυτό εδώ ουσιαστικά) με την τεκμηρίωση της πτυχιακής.

Να αναφερθεί επίσης ότι λόγω του ότι είναι ένα υποστηρικτικό κομμάτι της πτυχιακής εργασίας δεν δόθηκε μεγάλη έμφαση σε λεπτομέρειες πέρα από τη λειτουργικότητα . Όπως για παράδειγμα δεν υπάρχει απόλυτη ασφάλεια . Δηλαδή κάποιος αν “παρατηρήσει” το κώδικα μπορεί να βρει τρόπους (εφόσον έχει φτιάξει λογαριασμό χρήστη) να επέμβει στη βάση δεδομένων πέρα από το προβλεπόμενο τρόπο (Είτε με sql injection , είτε εκτελώντας κάποια php script που υπάρχουν ήδη με δικούς του παραμέτρους πάνω στο URL) . Πάντως , έχουν προβλεφθεί οι περιπτώσεις κακής χρήσης ή περιέργειας ενός τυπικού χρήστη , ώστε να μην δημιουργούνται από απλό λάθος προβλήματα στην βάση ή στην κανονική ροή χρήστης του Website .

Μια μικρή περιγραφή για τεχνικά μέρη που αφορούν την λειτουργία του Website :

Η ιστοσελίδα χρησιμοποιεί session ώστε να μπορέσει να πραγματοποιηθεί η σύνδεση κάποιου χρήστη .Για παράδειγμα , μετά από κάποιο επιτυχημένο Login δημιουργούνται 3 μεταβλητές στο session που έχουν το username, το ID βάσης και το τύπο του εκάστοτε συνδεδεμένου χρήστη .

Ενδεικτικός κώδικας :

```
$_SESSION['username']=$username;
$_SESSION['ID']=$row['UserID'];
$_SESSION['Type']=$row['Type'];
```

Ακόμη να αναφερθεί ενδεικτικά ότι στην αρχή κάθε σελίδας που απαιτεί πιστοποίηση χρήστη, υπάρχει κατάλληλος κώδικας ώστε να ελέγχει ότι έχει πραγματοποιηθεί επιτυχημένο login , πριν οποιαδήποτε άλλη ενέργεια και αλληλεπίδραση με τη βάση δεδομένων .

Πχ ενδεικτικά :

```
<!DOCTYPE html>
<?php
session_start(); //session starts here
if(!$_SESSION['username'])
{
    header("Location: Login.php");
}
if(isset($_GET['deviceID'])) ...
```

Επίσης, ένα ενδεικτικό κομμάτι κώδικα από Javascript :

```
function deleteMessage(mess)
{
    var r = confirm("Είσαι σίγουρος ότι θέλεις να διαγράψεις "+mess+";");
    if (r) {
        return true ;
    } else {
        return false ;
    }
}
```

Που χρησιμοποιείται πχ στο εξής σημείο :

```
<form action="EditDevice.php" method="post" onSubmit="return deleteMessage('τη συσκευή');">
    <input type="hidden" value="<?php echo $deviceID?>" name="deviceID" />
    <input type="submit" value="" name="deleteDevice" class="removeDevice" />
</form>
```



Ακόμη ενδεικτικά να φανεί μια περίπτωση ενός query μέσα από τη PHP , προς τη βάση δεδομένων που βρίσκεται στη σελίδα της invite.php :

```
<?php
include("Db_conection.php");
$user_username=$_SESSION['username'];
$user_ID=$_SESSION['ID'];

$view_users_query="SELECT u.Username,d.DeviceName, da.TypeAccess ,u.UserID ,
(SELECT TypeAccess FROM requests r WHERE r.DeviceID = $deviceID AND r.UserTo = u.UserID AND
r.UserFrom = $user_ID AND r.isDeleted = false) AS RequestAccessType
FROM users u
LEFT JOIN devices d ON d.OWner = u.UserID AND d.isDeleted = false AND d.DeviceID = '$deviceID'
LEFT JOIN deviceaccess da ON da.UserID = u.UserID AND da.DeviceID = '$deviceID' AND da.isDeleted = false
WHERE u.isDeleted = false AND u.isEnabled = true
ORDER BY 2 DESC , 3 DESC,1;";

$run=mysqli_query($dbcon,$view_users_query);

while($row=mysqli_fetch_array($run))
{
    $username=$row['Username'];
    $userID=$row['UserID'];
    $devicename=$row['DeviceName'];
    $access=$row['TypeAccess'];
    $requestAccessType=$row['RequestAccessType'];

?
}
```

Επειδή είναι δύσκολο να γίνει πλήρη αναφορά για όλες τις τεχνικές λεπτομέρειες που αφορούν την υλοποίηση της ιστοσελίδας μέσα από αυτό το pdf αρχείο , **μπορεί ο αναγνώστης να βρει όλο το κώδικα του website στο Github** . Εκεί υπάρχουν οι σχετικά μικροί και απλοί κώδικες (χωρίς όμως σχόλια τεκμηρίωσης λόγω του ότι η ιστοσελίδα είναι ένα υποστηρικτικό κομμάτι στο όλο σύστημα).

Μια ενδεικτική φωτογραφία από ένα κομμάτι από το κώδικα της ManageDevices.php του website :

```

<!DOCTYPE html>
<?php
session_start(); //session starts here
if($_SESSION['username']){
    header("Location: Login.php"); //redirect to login page to secure the welcome page without login access.
}
?>
<html>
<head>
<?php include('html/head.php');?>
</head>
<body>
<div id="container">
<?php $parentPath = __FILE__ ; include('header.php');?>
<div id="middle">
<br/>
<center>
<div align="left">
<form action="NewDevice.php">
<label for="newDevice"><font size="2" color="green">New Device:</font></label>
<input type="submit" class="newDeviceButton" value=""/>
</form>
</div>
</div>
<table border="1">
<thead>
<tr>
<th>Device ID</th>
<th>Comment</th>
<th>Right</th>
<th>Owner</th>
<th>Edit Device</th>
<th>Edit Access</th>
<th>Release Device</th>
</tr>
</thead>
<tbody>
<tr>
<td><?php $row['DeviceID'] ?></td>
<td><?php $row['Comment'] ?></td>
<td><?php $row['Right'] ?></td>
<td><?php $row['Owner'] ?></td>
<td><a href="EditDevice.php?DeviceID=<?php echo $row['DeviceID']; ?>">Edit Device</a></td>
<td><a href="EditAccess.php?DeviceID=<?php echo $row['DeviceID']; ?>">Edit Access</a></td>
<td><a href="ReleaseDevice.php?DeviceID=<?php echo $row['DeviceID']; ?>">Release Device</a></td>
</tr>
</tbody>
</table>
</div>
<?php
include("Db_conection.php");
$user_username=$_SESSION['username'];
$user_ID=$_SESSION['ID'];

$view_devices_query="SELECT d.DeviceID ,d.DeviceName,d.Comment,(SELECT Username FROM users WHERE UserID = d.Owner) AS Owner , da.TypeAccess
FROM devices d
LEFT JOIN deviceaccess da ON d.DeviceID = da.DeviceID AND da.UserID = '$user_ID' AND da.isDeleted = false
WHERE (d.Owner = '$user_ID'
OR da.UserID = '$user_ID') AND da.isDeleted = false ORDER BY 2";
$run=mysqli_query($dbcon,$view_devices_query); //here run the sql query.

while($row=mysqli_fetch_array($run)) //while look to fetch the result and store in a array prov.
?

```



Arduino

Εισαγωγή

Στην πτυχιακή αυτή ,το βάρος έπεισε κυρίως στο υπόλοιπο σύστημα και όχι στο κομμάτι του arduino που έχει να κάνει με την ηλεκτρονική περισσότερο. Οπότε αυτό που απασχόλησε κυρίως είναι το γεγονός να δύναται κάποιος χρήστης να βλέπει τις καταστάσεις των μονάδων απομακρυσμένα και ανάλογα τη φύση των μονάδων(πχ ένα θερμόμετρο από τη φύση του δεν μπορεί να του επιβληθεί κάποια τιμή όπως σε ένα led) να μπορεί να αλλάζει κάποιες τιμές σε αυτές. Δηλαδή το μέρος με το arduino υπήρξε κυρίως για να έχει έμπρακτο αποτέλεσμα το υπόλοιπο σύστημα και να υπάρχει χειροπιαστή χρησιμότητα ,με το να είναι ολοκληρωμένη η λειτουργία του .

Αναφέρεται αυτό για να διευκρινιστεί ότι δεν απασχόλησε πχ αν κάποιες μετρήσεις αισθητήρων δεν είναι απόλυτα ίδιες με την πραγματικότητα ή αξιοποιήσιμες τιμές ούτε πως θα δουλεύει αυτόνομα ένας “έξυπνος” χώρος. Ακόμη , παρόλο που γίνεται διαχείριση “έξυπνων” σπιτιών δεν σημαίνει ότι στα πλαίσια αυτής της πτυχιακής έχει ερευνηθεί και υλοποιηθεί κάποια περίπτωση ιδιαίτερης “ευφυίας” μέσα στο χώρο .

Με βάση λοιπόν τα προαναφερθέντα , δεν είναι σκοπός να μάθει ο αναγνώστης από αυτή την εργασία εξειδικευμένα πράγματα πάνω στο arduino (ούτε ενδείκνυται να μάθει κάποιος πως δουλεύει το arduino από αυτή εδώ τη πηγή).Παρόλα αυτά όμως θα γίνει μια προσπάθεια να εξηγηθούν κάποια πράγματα (με τι λίγη γνώση που υπάρχει) κυρίως σε πρακτικό επίπεδο για να μπορεί αυτός που διαβάζει την εργασία να “μπει στο κλίμα” και τουλάχιστον να καταλάβει τη λογική που υπάρχει ώστε να αποκτήσει μια βάση (καλό θα ήταν βέβαια να είναι όσο πιο σχετικός γίνεται με το χώρο και να έχει έστω μια μικρή ιδέα πάνω στον προγραμματισμό).

Επιπλέον , επειδή είναι το μοναδικό κομμάτι του συστήματος που ανάλογα τη περίπτωση χρειάζεται παραμετροποίηση από το χρήστη , θα γίνει μια πιο εκτεταμένη αναφορά , αφενός για να χρησιμοποιηθεί και σαν εγχειρίδιο και αφετέρου επειδή δεν υπάρχει τεκμηρίωση στον κώδικα των sketchs που βρίσκονται στο Github. Ετσι θα μπορεί ο κάθε χρήστης του συστήματος να είναι σε θέση (με αναζήτηση και φαντασία ,αν χρειαστεί) να προσαρμόσει κάποιο κώδικα arduino για τις δικές του ανάγκες .

Εν κατακλείδι , η παρουσίαση για το arduino αποσκοπεί ουσιαστικά στο να μπορεί ο χρήστης να φτιάχνει το δικό του κώδικα με τις δικές του απαιτήσεις για να μπορεί το “εκτελέσιμο” του να επικοινωνεί με το Device Client (με βάση βέβαια ένα “template sketch” που περιγράφεται στο τέλος μετά από όλα τα παραδείγματα).



Λίγα πρακτικά εισαγωγικά για το Arduino (Λαμπάκι που αναβοσβήνει)

Ενα από τα πιο εύκολα παραδείγματα για να δειχθεί πως γίνεται να προγραμματιστεί ένα Arduino (μέσα από το Arduino IDE) ,είναι να φανεί πως ένα led λαμπάκι μπορεί να ανάβει και να σβήνει .

- Ένα απλό κομμάτι κώδικα (sketch) που κάνει ένα led να αναβοσβήνει είναι το παρακάτω :

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 9 as an output.
  pinMode(9, OUTPUT);
}

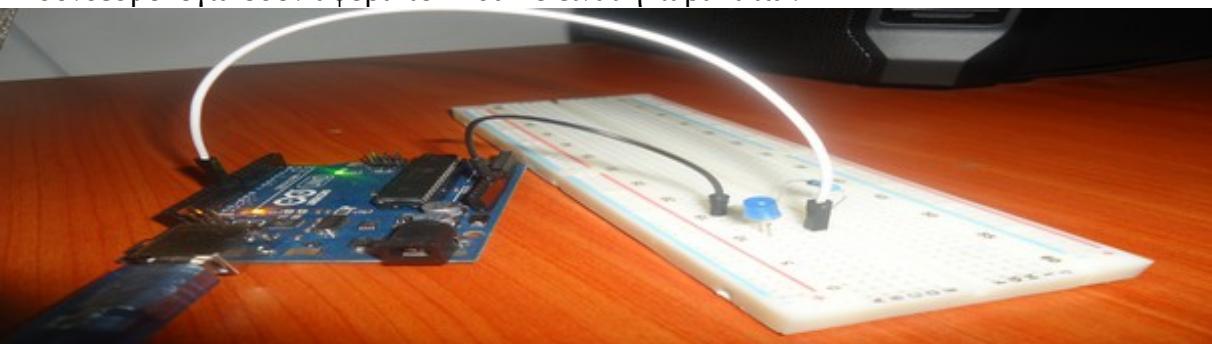
// the loop function runs over and over again forever
void loop() {
  digitalWrite(9, HIGH);    // turn the LED on (HIGH is the voltage level)
  delay(1000);             // wait for a second
  digitalWrite(9, LOW);     // turn the LED off by making the voltage LOW
  delay(1000);             // wait for a second
}
```

- ◆ Στην συνάρτηση **setup** θα μπορούσε να ειπωθεί ,ότι γίνονται οι όποιες αρχικοποιήσεις και εκτελείτε το “σώμα” της στην αρχή και μια μόνο φορά (ή μετά από reset).
Στην περίπτωση μας δηλώνουμε ότι το pin 9 του arduino θα είναι για έξοδο .
- ◆ **Το “σώμα” της συνάρτησης **loop** εκτελείτε συνέχεια σε επανάληψη .**

Στο παράδειγμα αυτό, στην αρχή κάνει στο pin 9 να έχει υψηλή τάση (ON),μετά καθυστερεί - “κολλάει” το πρόγραμμα για 1000ms (ένα δευτερόλεπτο) , στην συνέχεια κάνει το pin 9 να έχει χαμηλή τάση (OFF) , σαν τέταρτο βήμα κάνει το arduino να περιμένει ένα ακόμα δευτερόλεπτο και το επόμενο βήμα είναι να συνεχίσει ο βρόχος κανονικά από την αρχή.

Δηλαδή ανάβει, περιμένει, σβήνει, περιμένει , ανάβει κτλ .

Η συνδεσμολογία όσον αφορά το Arduino είναι η παρακάτω :



Δηλαδή , συνδέεται η γείωση (pin Gnd) με τη μια μεριά μιας αντίστασης και με την άλλη μεριά της αντίστασης συνδέεται ένα πόδι(το πιο κοντό) του led . Συγχρόνως το άλλο πόδι του led συνδέεται με το pin 9 για να λειτουργήσει ο παραπάνω κώδικας (sketch) .

Η αντίσταση χρειάζεται για να μην καεί το λαμπάκι (Όσο πιο μεγάλη τόσο μικρότερη φωτεινότητα θα υπάρχει και αντίστοιχα όσο πιο μικρή αντίσταση , θα υπάρχει μεγαλύτερη φωτεινότητα).

Και βέβαια το Arduino έχει συνδέσει με κάποιον υπολογιστή μέσω ενός καλωδίου USB .

Αντίσταση :

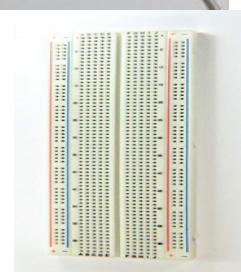
Αντίσταση είναι ένα μικρό σιδεράκι που στην μέση του ουσιαστικά έχει ένα υλικό για την αντίσταση του ηλεκτρικού ρεύματος.



Breadboard :

Είναι μια άσπρη πλακέτα (όπως φαίνεται στη φωτογραφία)και χρησιμοποιείται κυρίως για δοκιμές και για να βοηθάει στο να γίνουν διάφορες συνδέσεις μεταξύ του arduino (και γενικά ενός ελεγκτή) και διάφορων άλλων εξαρτημάτων με μεγαλύτερη άνεση .

Όπως φαίνεται στην φωτογραφία , οι οριζόντιες κουκκίδες της κάθε πλευράς (5) συγκοινωνούν και έτσι αν βαλθεί ένα καλωδιάκι σε μια κουκκίδα τότε θα περνάει το ίδιο ρεύμα και από τις υπόλοιπες και κατ' επέκταση θα υπάρξει σύνδεση. Αντίστοιχα το ίδιο γίνεται και στις κάθετες γραμμές(25) που χρησιμοποιούνται συνήθως για τη γείωση (-) και την υψηλή τάση (+) . Έτσι γίνεται πρακτικά με ένα πχ pin γείωσης του Arduino ,να είναι διαθέσιμες και άλλες 24 κουκκίδες Gnd για να γίνονται διάφορες πιθανές συνδέσεις.





Ένα άλλο κομμάτι κώδικα (sketch) που κάνει το led να αναβοσβήνει , όχι όμως με δύο καταστάσεις(on-off) αλλά με διαβαθμίσεις(αλλαγή στη φωτεινότητα) , είναι το παρακάτω :

```
int led = 9;           // the PWM pin the LED is attached to
int brightness = 0;    // how bright the LED is
int fadeAmount = 5;    // how many points to fade the LED by

// the setup routine runs once when you press reset:
void setup() {
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // set the brightness of pin 9:
  analogWrite(led, brightness);

  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
  if (brightness == 0 || brightness == 255) {
    fadeAmount = -fadeAmount;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(30);
}
```

1. Δηλώνεται και αρχικοποιούνται οι μεταβλητές :
 - led με τον αριθμό του αντίστοιχου pin του led μας .
 - brightness με το 0 για να είναι στην αρχή σβηστό το λαμπάκι .
 - Και τέλος ,το fadeAmount έχει τη τιμή 5 για να καθοριστεί το βήμα .
2. Στην συνάρτηση setup (που εκτελείτε μια φορά στην αρχή όπως ξανά-αναφέρθηκε) δηλώνεται και αυτή τη φορά ότι το pin 9 του arduino θα είναι για έξοδο .
3. Η συνάρτηση loop (που θα εκτελείτε συνέχεια) κάνει με την σειρά :
 - i. Το pin 9 του led να έχει τόσο ρεύμα όσο το brightness (δηλαδή στην αρχή 0)
 - ii. Αυξάνει όσο είναι το βήμα (5) το brightness .
 - iii. Έαν το brightness είναι 0 ή 255 αλλάζει το πρόσημο του βήματος (Δηλαδή πρακτικά από 5 γίνεται -5 όταν φτάσει το brightness να είναι 255 και αντίστοιχα όταν το brightness γίνει 0 επανέρχεται το βήμα στην τιμή 5)
 - iv. Κάνει μια καθυστέρηση 30 miliseconds πριν συνεχίσει ο βρόχος.

Παρατηρήσεις :

- ✓ Στα ψηφιακά pin (0..13) όταν θέλουμε να διαβάσουμε χρησιμοποιούμε τη **digitalRead(pin)** για να πάρουμε μια τιμή 2 καταστάσεων (HIGH και LOW)και τη **digitalWrite(pin)** για να δώσουμε μια τιμή 2 καταστάσεων
- ✓ Αν θέλουμε τώρα, όπως στο παράδειγμα μας, να προσομοιώσουμε μια ψηφιακή έξοδο σε αναλογική χρησιμοποιούμε τη **analogWrite(pin)** και δίνουμε μια τιμή μεταξύ 0 και 255 .
 - Να επισημανθεί εδώ , ότι από τα ψηφιακά pin αυτά που έχουν περισπωμένη (PWM) ,όπως και το pin 9, πάνω στη πλακέτα μπορούν να “δώσουν” σαν έξοδο κάποια τιμή μεταξύ 0 και 255 και έτσι στο παράδειγμα μας μπορούν να δώσουν διακριτό ρεύμα κάθε φορά . Τα pin όμως χωρίς την περισπωμένη, έχουν δύο καταστάσεις και έτσι όταν τους “δώσουμε” κάποια τιμή κάτω από το 128 το αντιλαμβάνονται σαν “LOW” και πάνω από το 127 το αντιλαμβάνονται σαν “HIGH” .
Δηλαδή αν πχ αντί για το pin 9 συνδέσουμε το pin 8 (και το αλλάζαμε και στο κώδικα) θα είχαμε το αποτέλεσμα του πρώτου sketch , δηλαδή θα άναψε και θα σβήνε “κοφτά” το led κάθε φορά.
- ✓ Τώρα ,τα αναλογικά pin (A0..A5) μπορούμε να τα χρησιμοποιήσουμε για να διαβάσουμε με την **analogRead(pin)** τιμές (μεταξύ 0-1023) από κάποια αναλογική μονάδα (πχ ένα θερμόμετρο).
Επίσης, τα χρησιμοποιούμε και για έξοδο, αλλά πρακτικά (από ότι παρατηρήθηκε και λέγετε με κάθε επιφύλαξη) δεν μπορούμε να δώσουμε διακριτές τιμές, δηλαδή καταλαβαίνει μόνο “HIGH” και “LOW” .



Διάβασμα και αποστολή από και προς τον υπολογιστή

Με το παρακάτω σχετικά απλό κομμάτι κώδικα (sketch) μπορούμε να δείξουμε πως γίνεται να αλληλεπιδράσουμε με το arduino από τον υπολογιστή μας και πιο συγκεκριμένα μέσα από το Arduino IDE χρησιμοποιώντας σαν σειριακό μέσο ένα καλώδιο USB .

```
int incomingByte = 0;
int ledPin = 9;

void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
}

void loop() {
  delay(30);
  if (Serial.available() > 0) {
    incomingByte = Serial.read();

    if (incomingByte==48)
      analogWrite(ledPin, 0);
    else
      analogWrite(ledPin, 255);

    Serial.println(incomingByte);
  }
}
```

- ➔ Στο sketch αρχικά δηλώνουμε 2 μεταβλητές μία για το pin του led μας (η συνδεσμολογία είναι η ίδια με το προηγούμενο παράδειγμα του αναβοσβήματος του led) και μια για να την χρησιμοποιήσουμε με το να διαβάζουμε bytes από το USB μας και κατ' επέκταση από το pc
- ➔ Στην συνάρτηση setup δηλώνουμε ότι ο ρυθμός μετάδοσης της σειριακής μας(USB) θα είναι 9600bps και ότι το pin 9 θα χρησιμοποιηθεί για έξοδο .
- ➔ Στην συνάρτηση loop που εκτελείτε συνέχεια , αρχικά δημιουργούμε μια καθυστέρηση 30ms και στη συνέχεια ελέγχουμε αν υπάρχουν δεδομένα μέσα στο σειριακό μας μέσο (USB).
Στην περίπτωση που υπάρχει κάτι στο μέσο , διαβάζουμε από το κανάλι ένα byte και αν η τιμή του είναι 48 (Στο πίνακα ASCII είναι ο χαρακτήρας 0) τότε κλείνουμε το led και σε οποιαδήποτε άλλη περίπτωση (σε οποιοσδήποτε άλλο αριθμό δηλαδή) ανοίγουμε το led . Η αμέσως επόμενη ενέργεια είναι να στείλουμε στο σειριακό μας μέσο το byte που μας ήρθε .
- ✓ Στο παράδειγμα μας φαίνεται πως μέσα από το Arduino IDE μπορούμε να εμφανίσουμε με πολύ απλό τρόπο ότι κάνουμε “print” από το κώδικα μας .
- ✓ Επίσης, βλέπουμε και διαπιστώνουμε πως υπάρχει η δυνατότητα να στείλουμε μέσα από τον υπολογιστή διάφορες πληροφορίες ούτως ώστε να πετύχουμε “συγχρόνως” αμφίδρομη επικοινωνία μεταξύ των δυο άκρων του καλωδίου USB δηλαδή μεταξύ του Arduino και ενός υπολογιστή .



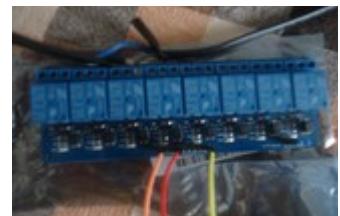
Διαχείριση ηλεκτρικών συσκευών από υπολογιστή(Άνοιγμα-κλείσιμο πορτατίφ)

Χρησιμοποιώντας το κώδικα που έχουμε από το προηγούμενο παράδειγμα μας και παραμετροποιώντας ελάχιστα τη συνδεσμολογία μπορούμε να ανοιγό-κλείνουμε ένα πορτατίφ πολύ εύκολα αλλά και οποιαδήποτε άλλη ηλεκτρική συσκευή με τη βοήθεια ενός ρελέ.

- ✓ **Ρελέ** είναι ένα εξάρτημα που μας επιτρέπει με μικρές τάσεις και λίγο ρεύμα να χειριστούμε πολύ μεγαλύτερες τάσεις και περισσότερο ρεύμα.

Για παράδειγμα, μπορούμε να το χρησιμοποιήσουμε σαν διακόπτη και αν του δώσουμε πχ 5V να μην αφήνει να περνάει ρεύμα από ένα καλώδιο και αν του δίνουμε 0V να αφήνει να περνάει...

| Ρελέ 8 επαφών :



Για να ανάψει ένα πορτατίφ χρειάζεται να παίρνει ρεύμα από μια πρίζα. Το καλώδιο που χρησιμοποιούμε ουσιαστικά έχει μέσα 2 μικρότερα καλώδια με ρεύμα, το ένα με το θετικό (+) και το άλλο με το αρνητικό (-). Έτσι για να δουλέψει το πορτατίφ αλλά και μια οποιαδήποτε ηλεκτρική συσκευή πρέπει να περνάει ρεύμα και από τα δυο καλώδια συγχρόνως. Οπότε αν εμείς κόψουμε το ένα από τα δυο καλώδια και βάλουμε τις 2 νέες άκρες του στο ρελέ, μπορούμε να δημιουργήσουμε ουσιαστικά ένα διακόπτη που θα τον ελέγχουμε μέσα από το arduino. Δηλαδή πχ αν το arduino στείλει από κάποια έξοδο του 5V τότε το ρελέ να κλείνει το κύκλωμα και να περνάει ρεύμα στο καλώδιο και ουσιαστικά στην περίπτωση του πορτατίφ να ανοίγει η λάμπα.

Έτσι με το sketch που έχουμε από το προηγούμενο παράδειγμα και με την συνδεσμολογία που φαίνεται δίπλα, μπορούμε να ανοίγουμε και να κλείνουμε ένα πορτατίφ .

- **Με το κόκκινο καλώδιο** συνδέουμε το pin 9 του arduino με το pin της τρίτης επαφής του Ρελέ.
- **Με το πορτοκαλί καλώδιο** συνδέουμε το Ground του arduino με το αντίστοιχο pin γείωσης του Ρελέ
- **Με το κίτρινο καλώδιο** συνδέουμε το Vcc του arduino με το αντίστοιχο pin Vcc του Ρελέ .
- **Το μεγάλο μαύρο καλώδιο** του πορτατίφ το έχουμε κόψει όπως και το ένα από τα δύο καλώδια που έχει μέσα και τις δύο νέες άκρες του τις έχουμε τοποθετήσει στην 3η επαφή του Ρελέ .





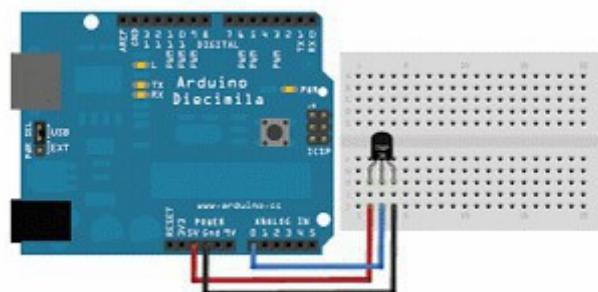
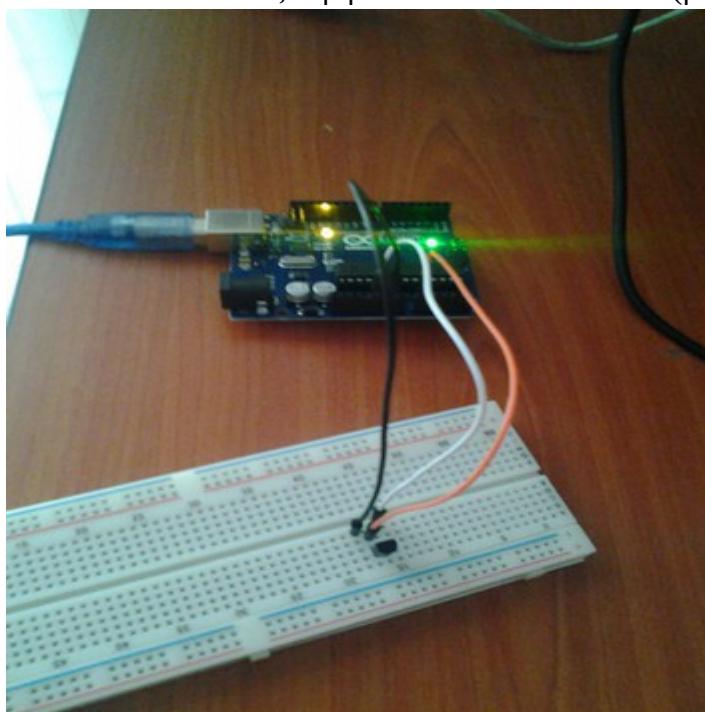
Αναλογικό Θερμόμετρο

Ένα απλό μικρό sketch που μετράει την θερμοκρασία και την εμφανίζει στην οθόνη κάθε μισό δευτερόλεπτο :

```
thermometroDemo S
int val;
int tempPin = A0;
void setup()
{
    Serial.begin(9600);
}
void loop()
{
    val = analogRead(tempPin);
    float mv = (val/1024.0)*5000;
    float cel = mv/10;
    Serial.print("TEMPRATURE = ");
    Serial.print(cel);
    Serial.print("*C");
    Serial.println();
    delay(500);
}
```

The serial monitor shows the following output:
TEMPTURE = 17.09*C
TEMPRATURE = 17.09*C
TEMPRATURE = 16.60*C
TEMPRATURE = 17.09*C
TEMPRATURE = 17.58*C
TEMPRATURE = 17.09*C
TEMPRATURE = 17.09*C
TEMPRATURE = 16.60*C
TEMPRATURE = 17.09*C

- Δηλώνεται ότι η μεταβλητή tempPin θα έχει ένα αναλογικό pin του Arduino και πιο συγκεκριμένα το A0 (Επειδή το θερμόμετρο αυτό είναι αναλογικό)
- Στην συνάρτηση setup (που όπως αναφέρθηκε ξανά, εκτελείτε τη πρώτη φορά μόνο ή μετά από reset) καθορίζουμε ότι ο ρυθμός μετάδοσης της σειριακής (USB) θα είναι 9600bps.
- Στο “σώμα” της συνάρτησης loop ουσιαστικά διαβάζεται μια μέτρηση από το Pin A0 και στην συνέχεια εφαρμόζονται κάποιοι τύποι για να βρεθούν οι βαθμοί Κελσίου .
Αμέσως μετά τυπώνεται η θερμοκρασία στη σειριακή έξοδο .
Και τέλος περιμένει - “κολλάει” 500ms (μισό δευτερόλεπτο) και ξανά συνεχίζει ο βρόχος.



- ◆ Η σύνδεση πάνω στο arduino είναι όπως φαίνεται και από τις φωτογραφίες .
Δηλαδή το ένα ακριανό πόδι το συνδέεται με το ground , το άλλο ακριανό με τα 5V και το μεσαίο με το A0.
- Tip : Έχει σημασία που θα μπει το ground και τα 5V στα ακριανά ποδαράκια .
Αν τα βάλετε ανάποδα θα γίνει καυτό το θερμόμετρο και πιθανόν να καεί (?) .
Στο θερμόμετρο η μια μεριά του είναι κυρτή οπότε παρατηρήστε τη φώτο.



Αισθητήρας Φωτιάς

Ένα μικρό sketch που διαβάζει μετρήσεις από έναν αισθητήρα φωτιάς και καθορίζει ανάλογα, τη φωτεινότητα ενός led. Επίσης, στέλνει στο pc και εμφανίζονται στην οθόνη τα ανάλογα μηνύματα:

```
Fotias S
const int analogInPin = A0;
const int analogOutPin = 9;

int sensorValue = 0;
int outputValue = 0;

void setup() {
    Serial.begin(9600);
}

void loop() {
    sensorValue = analogRead(analogInPin);

    outputValue = map(sensorValue, 0, 1023, 0, 255);

    analogWrite(analogOutPin, outputValue);

    Serial.print("sensor = ");
    Serial.print(sensorValue);
    Serial.print("\t output = ");
    Serial.println(outputValue);

    delay(2);
}
```

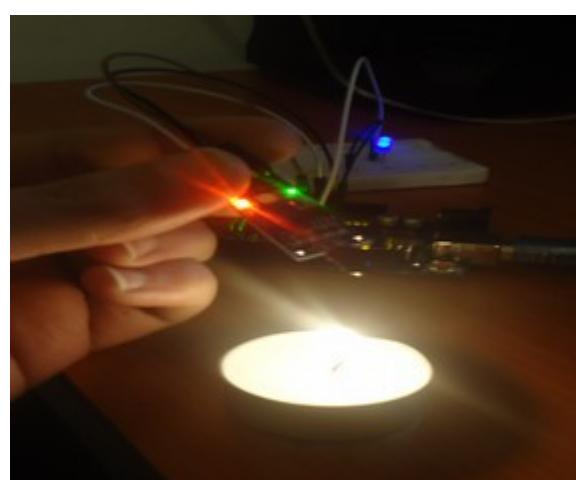
The screenshot shows the Arduino Serial Monitor window titled '/dev/ttyACM0'. It displays a series of lines of text where each line contains two entries: 'sensor' and 'output'. Both values are consistently shown as 882 and 219 respectively, indicating a constant output level. There is also a checked 'Autoscroll' option at the bottom.

- Δηλώνονται 2 σταθερές μεταβλητές για τα pin του arduino που θα χρησιμοποιηθούν , μία ακόμη για την μέτρηση που γίνεται και άλλη μία για τη τιμή που θα παίρνει το led .
- Στην συνάρτηση setup καθορίζεται πάλι ότι ο ρυθμός μετάδοσης θα είναι 9600bps.
- Στο “σώμα” της συνάρτησης loop ουσιαστικά διαβάζεται μια μέτρηση από το Pin A0 και στην συνέχεια με τη χρήση της συνάρτησης map βρίσκεται η φωτεινότητα που θα παίρνει κάθε φορά το led . Αμέσως μετά τυπώνονται τα περιεχόμενα των μεταβλητών .

Η map συνάρτηση (πρακτικά και με λίγα λόγια) παίρνει 5 ορίσματα,το πρώτο είναι μια τιμή ,το 2ο και 3ο είναι ένα εύρος και το 4ο και 5ο είναι ένα άλλο εύρος . Έτσι ανάλογα με το που βρίσκεται η τιμή στο πρώτο εύρος επιστρέφεται η ανάλογη-αντίστοιχη τιμή του δεύτερου εύρους.

Για παράδειγμα , η map(30,0,100,0,10) θα επέστρεψε 3 ή η map(100, 0, 255, 255, 0) θα επέστρεψε 155 ή ακόμη και η map(130,0,200,100,-100) θα επέστρεψε το -30 κλπ .

Στη πρώτη φότο από κάτω ανάβει πολύ το led και τα μηνύματα του είναι στο screenshot από πάνω.
Στη δεύτερη φότο από κάτω ανάβει λιγότερο το led και τα μηνύματα του είναι στη 3η φότο .



The screenshot shows the Arduino Serial Monitor window titled '/dev/ttyACM0'. It displays a series of lines of text where each line contains two entries: 'sensor' and 'output'. The 'sensor' values are 38, 37, 38, 37, 38, 37, 37, 37, 37, 38, 37, which correspond to the 1023 values from the previous screenshot. The 'output' values are all 9, indicating a lower output level for the dimmer LED.

Ο αισθητήρας έχει συνήθως 4 pin από τα οποία στο παράδειγμα χρησιμοποιήθηκαν τα 3 . Πιο συγκεκριμένα τα pin Gnd , Vcc και A0 του αισθητήρα συνδέθηκαν στα αντίστοιχα pin του arduino. Όσο για το led η σύνδεση έγινε όπως στο παράδειγμα με το led που αναβοσβήνει (pin 9) .



Αισθητήρας υγρασίας και θερμοκρασίας

Ενα sketch που με τη βοήθεια της βιβλιοθήκης DHT διαβάζονται μετρήσεις από έναν αισθητήρα υγρασίας και θερμοκρασίας και στην συνέχεια εμφανίζονται στην οθόνη :

```
#include "DHT.h"
#define DHTPIN 4 // what digital pin we're connected to
#define DHTTYPE DHT11 // DHT 11
DHT dht(DHTPIN, DHTTYPE);

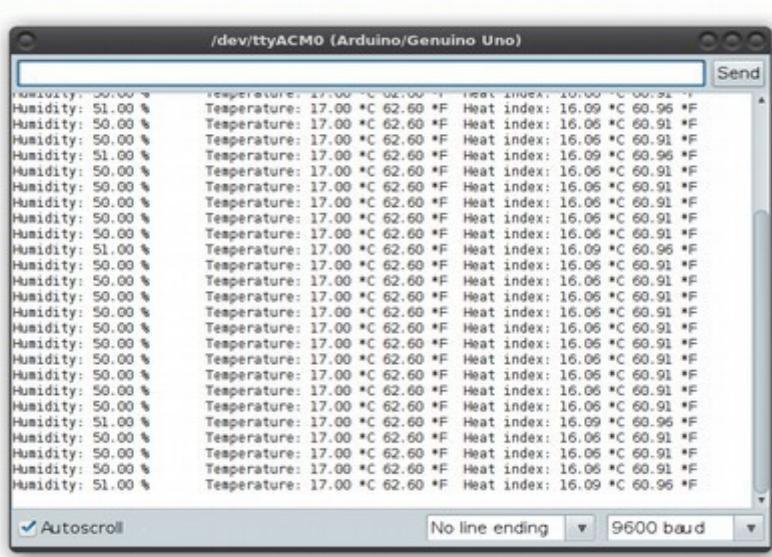
void setup() {
  Serial.begin(9600);
  Serial.println("DHTxx test!");

  dht.begin();
}

void loop() {
  delay(2000);
  float h = dht.readHumidity();
  float t = dht.readTemperature();
  float f = dht.readTemperature(true);

  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }
  float hif = dht.computeHeatIndex(f, h);
  float hic = dht.computeHeatIndex(t, h, false);

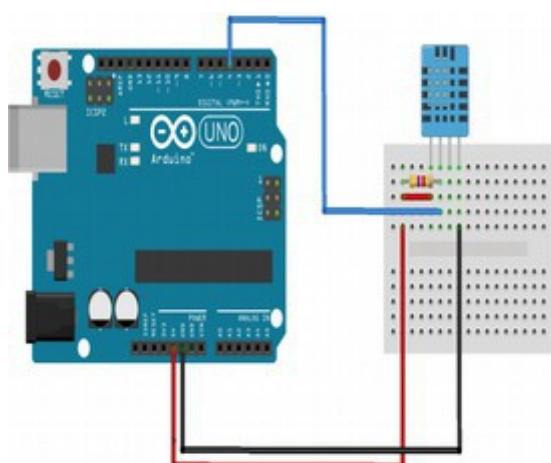
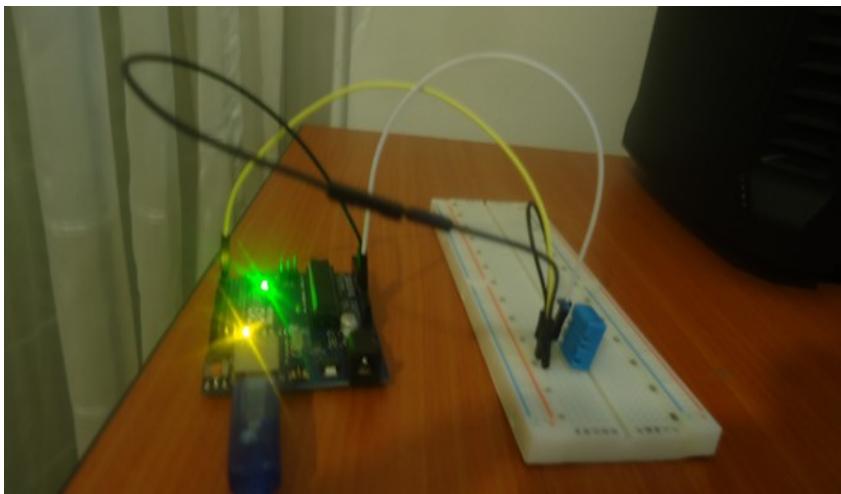
  Serial.print("Humidity: ");
  Serial.print(h);
  Serial.print(" %\t");
  Serial.print("Temperature: ");
  Serial.print(t);
  Serial.print(" *C\t");
  Serial.print(f);
  Serial.print(" *F\t");
  Serial.print("Heat index: ");
  Serial.print(hic);
  Serial.print(" *C ");
  Serial.print(hif);
  Serial.println(" *F");
}
```



- ➔ Αρχικά δηλώνεται η βιβλιοθήκη DHT και ορίζονται δύο σταθερές , μία για το pin που θα διαβάζονται οι μετρήσεις και μια για το τύπο του αισθητήρα του παραδείγματος .
- ➔ Επίσης, δημιουργείται το dht αντικείμενο της κλάσης DHT παίρνοντας σαν ορίσματα στον κατασκευαστή τις δύο σταθερές (του προηγούμενου βήματος) .
- ➔ Στη setup πέρα από ότι δηλώνεται το bitrate 9600 εκτελείται και η μέθοδος begin() του dht.
- ➔ Στην loop κάθε δύο δευτερόλεπτα παίρνονται μετρήσεις από τον αισθητήρα με τη βοήθεια έτοιμων μεθόδων που έχει η κλάση DHT . Ακόμη γίνεται και κάποιος έλεγχος στην περίπτωση που κάποιες μετρήσεις δεν είναι αριθμοί ούτως ώστε να εμφανίζεται το ανάλογο μήνυμα και να ξανά-ξεκινάει ο βρόχος της loop() .

Αν βέβαια όλες οι μετρήσεις είναι αριθμοί στέλνονται και εμφανίζονται οι μετρήσεις στο pc.

Αν κάποιος έχει διαβάσει τα προηγούμενα παραδείγματα του arduino δεν χρειάζεται περαιτέρω ανάλυση πέρα από δύο ενδεικτικές φωτογραφίες για το πως επιτεύχθηκε η συνδεσμολογία .



Made with Fritzing.org



Αισθητήρας απόστασης (Sonar)

Ενα sketch που ουσιαστικά με βάση τις μετρήσεις ενός αισθητήρα απόστασης αλλά και κάποιους τύπους βρίσκεται σε τι απόσταση (εκατοστά και ίντσες) υπάρχει εμπόδιο και βέβαια στέλνονται οι τιμές των αποστάσεων στον υπολογιστή και έτσι γίνεται να εμφανιστούν και στην οθόνη :

```
int trig = 3; // attach pin 3 to Trig
int echo = 4; //attach pin 4 to Echo

void setup() {
    Serial.begin(9600);

    pinMode(trig, OUTPUT);
    pinMode(echo, INPUT);
}

long duration, inches, cm;
void loop()
{
    digitalWrite(trig, LOW);
    delayMicroseconds(2);
    digitalWrite(trig, HIGH);
    delayMicroseconds(5);
    digitalWrite(trig, LOW);

    duration = pulseIn(echo, HIGH);

    inches = microsecondsToInches(duration);
    cm = microsecondsToCentimeters(duration);

    Serial.print(inches);
    Serial.print("in, ");
    Serial.print(cm);
    Serial.print("cm");
    Serial.println();
    delay(100);
}

long microsecondsToInches(long microseconds)
{
    return microseconds / 74 / 2;
}

long microsecondsToCentimeters(long microseconds)
{
    return microseconds / 29 / 2;
}
```

The screenshot shows the Arduino IDE interface. On the left is the code editor with the Sonar sketch. On the right is the串行监视器 (Serial Monitor) window, which displays a continuous stream of data. The data consists of two columns of measurements: inches and centimeters, listed sequentially from top to bottom. The first few lines of the output are:

| inches | cm |
|------------|----|
| 12in, 31cm | |
| 12in, 30cm | |
| 10in, 27cm | |
| 10in, 26cm | |
| 9in, 24cm | |
| 8in, 22cm | |
| 7in, 18cm | |
| 7in, 19cm | |
| 5in, 14cm | |
| 4in, 10cm | |
| 3in, 9cm | |
| 3in, 8cm | |
| 3in, 8cm | |
| 3in, 8cm | |
| 4in, 10cm | |
| 5in, 13cm | |
| 6in, 16cm | |
| 6in, 16cm | |
| 7in, 19cm | |
| 9in, 23cm | |
| 10in, 26cm | |
| 11in, 29cm | |
| 11in, 28cm | |
| 12in, 31cm | |
| 13in, 33cm | |
| 14in, 36cm | |
| 15in, 38cm | |
| 14in, 37cm | |
| 15in, 40cm | |
| 15in, 40cm | |
| 16in, 41cm | |
| 16in, 41cm | |

At the bottom of the Serial Monitor window, there is a checkbox labeled "Autoscroll" which is checked.

- ➔ Δηλώνονται 2 μεταβλητές για τα 2 pin που έχει ο αισθητήρας sonar (εκτός από το Vcc και Gnd).
- ➔ Στην setup δηλώνεται εκτός από το bitrate ότι θα είναι 9600 & ότι το 1o pin θα είναι output & το 2o input.
- ➔ Στην loop (στο περίπου) απλά στέλνεται από το 1 pin ένας παλμός ανά κάποια μικρό δευτερόλεπτα και στο τέλος διαβάζεται η επιστροφή για να βρεθεί με κάποιους τύπους η απόσταση του εμποδίου σε ίντσες και εκατοστά .
Στην συνέχεια απλά τυπώνεται η απόσταση με τις 2 εκδοχές της και μετά από 200 ms καθυστέρηση ξανά συνεχίζει ο βρόχος της loop.
- ✓ Η συνάρτηση “pulseIn” επιστρέφει το μήκος του παλμού (σε μικροδευτερόλεπτα) ή 0 εάν δεν υπάρχει παλμός μέσα σε ένα χρονικό όριο .
- Ο αισθητήρας έχει 4 pin όπως έχει γράφτει και πιο πάνω οπότε το Vcc και το Gnd του πρέπει να συνδέθούν με τα αντίστοιχα του arduino .
Το trig και το echo μπορούν να συνδεθούν όπου θέλει κάποιος αλλά με βάση το κώδικα πρέπει να συνδεθεί το trig με το pin 3 και το echo με το pin 4 του arduino .
- ✗ Η έξοδος του προγράμματος είναι έτσι γιατί το εμπόδιο πλησίαζε και απομακρύνόταν εκείνη την στιγμή





Φωτοαισθητήρας

Ενα μικρό sketch που διαβάζει μετρήσεις από έναν αισθητήρα φωτός και καθορίζει ανάλογα, τη φωτεινότητα ενός led. Επίσης, στέλνει στο pc και εμφανίζονται στην οθόνη τα ανάλογα μηνύματα:

```
int const fotoAistitiras = A1 ;
int const led = 9 ;
void setup(){
  Serial.begin(9600);
  pinMode(fotoAistitiras, INPUT) ;
  pinMode(led, OUTPUT) ;
}

void loop(){
  double fos = analogRead(fotoAistitiras);
  double fotinotitaStoLed = map(fos,0,400,255,0) ;
  if (fotinotitaStoLed<0) fotinotitaStoLed = 0 ;
  analogWrite(led,fotinotitaStoLed) ;
  Serial.print("O fotoaistitas exei: ");
  Serial.print(fos);
  Serial.print("\n");
  Serial.print("O Led exei fotinotita: ");
  Serial.print(fotinotitaStoLed);
  Serial.print("\n");

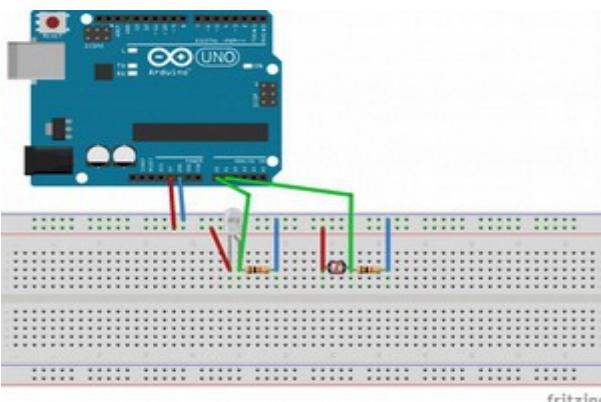
  delay(500);
}
```

```
O fotoaistitas exei: 6.00
O Led exei fotinotita: 252.00
O fotoaistitas exei: 18.00
O Led exei fotinotita: 244.00
O fotoaistitas exei: 28.00
O Led exei fotinotita: 238.00
O fotoaistitas exei: 7.00
O Led exei fotinotita: 251.00
O fotoaistitas exei: 7.00
O Led exei fotinotita: 251.00
O fotoaistitas exei: 7.00
O Led exei fotinotita: 251.00
O fotoaistitas exei: 8.00
O Led exei fotinotita: 250.00
O fotoaistitas exei: 15.00
O Led exei fotinotita: 246.00
O fotoaistitas exei: 424.00
O Led exei fotinotita: 0.00
O fotoaistitas exei: 335.00
O Led exei fotinotita: 42.00
```

Autoscroll

- Δηλώνονται 2 σταθερές μεταβλητές για τα pin του arduino που θα χρησιμοποιηθούν .
- Στην συνάρτηση setup καθορίζεται ότι ο ρυθμός μετάδοσης θα είναι 9600bps και ότι το pin του φωτοαισθητήρα θα είναι για INPUT και του led για OUTPUT
- (Καλό είναι να γίνεται αυτό σε όλες τις περιπτώσεις ακόμη και στα παραδείγματα που έχει λησμονηθεί ,επειδή μπορεί κάποιες φορές να μην μας “παίζει” και να φταίει που δεν κάναμε ρητά τη δήλωση για το κάθε pin αν θα είναι INPUT ή OUTPUT).
- Στο “σώμα” της συνάρτησης loop ουσιαστικά διαβάζεται μια μέτρηση από το Pin A1 για το πόσο φως υπάρχει στο χώρο και στην συνέχεια με τη χρήση της συνάρτησης map καθορίζεται η φωτεινότητα που θα πάρει το led μας .(Αν υπάρχει πολύ φως στο χώρο κλείνει το led & αντίθετα όσο λιγότερο φως υπάρχει τόσο αυξάνεται η φωτεινότητα του led) Αμέσως μετά τυπώνονται τα περιεχόμενα των μεταβλητών του παραδείγματος .

Στις 3 από κάτω φώτο φαίνονται οι περιπτώσεις(1)που δεν έχει φως,(2)έχει πάρα πολύ&(3)έχει λίγο



✓ **Αριστερά** φαίνεται η συνδεσμολογία και η μόνη διαφορά είναι ότι το led στο παράδειγμα,είναι συνδεδεμένο με το κλασικό τρόπο (όπως στα προηγούμενα παραδείγματα) και αντί το pin A0 που δείχνει η φώτο χρησιμοποιείται το pin 9.

✗ **Δεξιά** φαίνεται η έξοδος όταν πέφτει πολύ φως στον φωτοαισθητήρα .

```
O fotoaistitas exei: 176.00
O Led exei fotinotita: 143.00
O fotoaistitas exei: 272.00
O Led exei fotinotita: 82.00
O fotoaistitas exei: 499.00
O Led exei fotinotita: 0.00
O fotoaistitas exei: 517.00
O Led exei fotinotita: 0.00
O fotoaistitas exei: 470.00
O Led exei fotinotita: 0.00
O fotoaistitas exei: 492.00
O Led exei fotinotita: 0.00
O fotoaistitas exei: 496.00
O Led exei fotinotita: 0.00
O fotoaistitas exei: 472.00
O Led exei fotinotita: 0.00
O fotoaistitas exei: 478.00
O Led exei fotinotita: 0.00
O fotoaistitas exei: 455.00
O Led exei fotinotita: 0.00
```

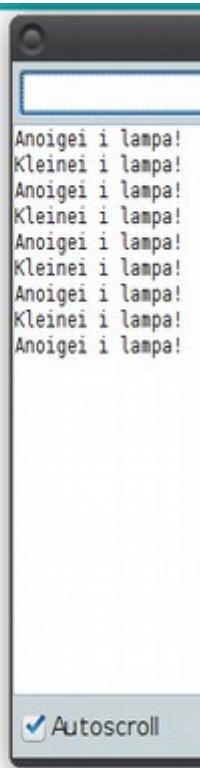


Αισθητήρας κίνησης

Ένα μικρό sketch που με τη βοήθεια ενός αισθητήρα κίνησης ανάβει ένα led όταν υπάρχει οποιαδήποτε μεταβολή θέσης σε κάτι μέσα στο χώρο :

```
int aistKinPin = 8;
int ledPin = 12;
long timeFotokyttaro = 0 ;
bool flagOpen = false;
void setup(){
  Serial.begin(9600);
  pinMode(aistKinPin, INPUT);
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, LOW);
}

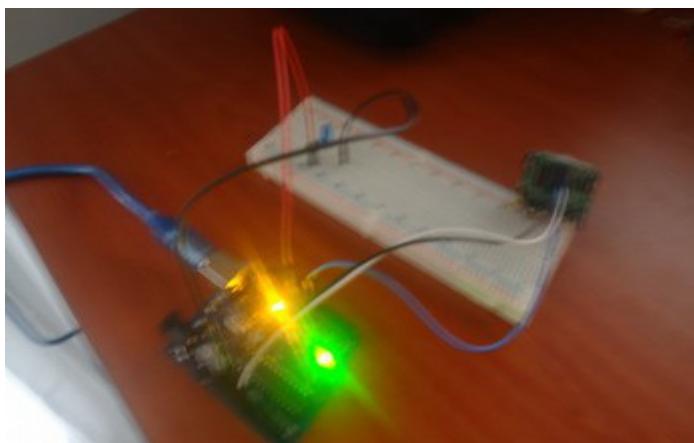
void loop(){
  if (timeFotokyttaro<3000)
    timeFotokyttaro++ ;
  if(digitalRead(aistKinPin) == HIGH){
    timeFotokyttaro = 0 ;
    if (!flagOpen)
    {
      flagOpen = true ;
      digitalWrite(ledPin, HIGH);
      Serial.println("Anoigei i lampa!");
    }
  }
  if(digitalRead(aistKinPin) == LOW){
    if (timeFotokyttaro>=3000 && flagOpen)
    {
      digitalWrite(ledPin, LOW);
      flagOpen = false ;
      Serial.println("Kleinei i lampa!");
    }
  }
}
```



- ➔ Δηλώνονται 4 μεταβλητές , 2 για τα pin που θα χρησιμοποιήθουν (led και αισθητήρα) , μια για έναν μετρητή και μια για μια bool “σημαία” .
- ➔ (Setup)Καθορίζεται bitRate 9600bps και δηλώνεται ρητά ότι το pin 8 θα είναι για είσοδο και το pin 12 για έξοδο όπως επίσης κλείνει το led .
- ➔ Η “loop” πρακτικά διαβάζει συνεχώς μετρήσεις από τον αισθητήρα και όταν ανιχνευτεί κίνηση ανάβει το λαμπάκι και στέλνεται ένα μήνυμα στο pc . Όταν ο αισθητήρας για ένα χρονικό περιθώριο 3000 επαναλήψεων της loop δεν εντοπίσει άλλη κίνηση τότε κλείνει το led και στέλνεται το ανάλογο μήνυμα στο pc.

Όσο για την συνδεσμολογία (πέρα από το led που συνδέεται στη γείωση και στο pin 12) συνδέονται τα 3 pin που έχει ο αισθητήρας με τα αντίστοιχα του arduino . Στον συγκεκριμένο αισθητήρα που έγινε η δοκιμή (ίσως να μην είναι έτσι όλοι οι αισθητήρες κίνησης) συνδέθηκε το μεσαίο του pin στο pin 8 του arduino(μπλε καλώδιο),το αριστερό του pin στο Ground(μαύρο)& το δεξιό του στα 5V(άσπρο) του arduino
(Το αριστερά και δεξιά είναι με βάση τη πλευρά που είναι τα πινάκια και όπως το κοιτάμε εμείς)

Η αριστερή κάτω εικόνα δεν έχει ανιχνεύσει κάποια κίνηση σε αντίθεση με τη δεξιά κάτω εικόνα .





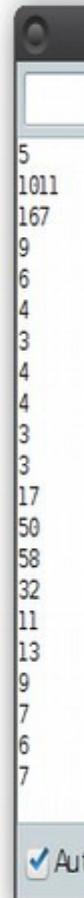
Αισθητήρας φωνής (Μικρόφωνο)

Ενα sketch που με τη βοήθεια ενός αισθητήρα φωνής αντιλαμβάνεται ήχους και στέλνει τις μεταβολές στον υπολογιστή ο οποίος στην περίπτωση μας τις εμφανίζει στην οθόνη :

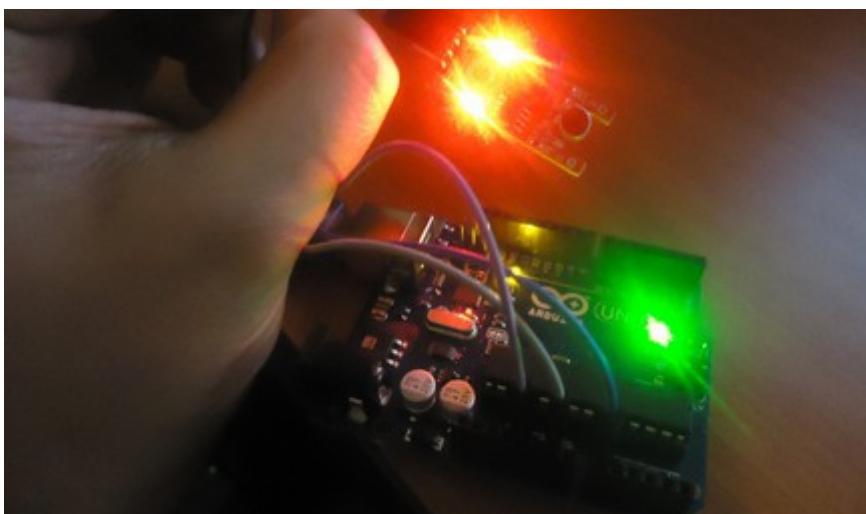
```
const int sampleWindow = 200; // Sample window width in ms
unsigned int sample;
void setup()
{
    Serial.begin(9600);
}
void loop()
{
    unsigned long startMillis= millis(); // Start of sample window
    unsigned int peakToPeak = 0; // peak-to-peak level

    unsigned int signalMax = 0;
    unsigned int signalMin = 1024;

    // collect data for 200 ms
    while (millis() - startMillis < sampleWindow)
    {
        sample = analogRead(A0);
        if (sample < 1024) // toss out spurious readings
        {
            if (sample > signalMax)
            {
                signalMax = sample; // save just the max levels
            }
            else if (sample < signalMin)
            {
                signalMin = sample; // save just the min levels
            }
        }
    }
    peakToPeak = signalMax - signalMin; // max - min = peak-peak amplitude
    //double volts = (peakToPeak * 3.3) / 1024; // convert to volts
    Serial.println(peakToPeak);
}
```



- ➔ Πρακτικά και με λίγα λόγια (χωρίς να επεξηγείτε απόλυτα) αυτό που κάνει το πρόγραμμα είναι να πράττει συνεχώς μετρήσεις από το μικρόφωνο και να βρίσκει τη μέγιστη και ελάχιστη τιμή για κάθε 200 ms που περνάνε .
Και έτσι σε επόμενη φάση να βρίσκει τη διαφορά μεταξύ του min και max και να την στέλνει στο τέλος κάθε επανάληψης της loop στον υπολογιστή.
- ✓ Millis() Επιστρέφει τα ms που έχουν περάσει από την έναρξη του sketch (Πόση ώρα τρέχει δηλαδή σε ms)
- ✗ Στις μετρήσεις που φαίνονται στην εικόνα αριστερά ανιχνεύθηκαν το 1011 ,167 γιατί εκείνη την ώρα ακούστηκε δυνατά και απότομα κάποιος στιγμιαίος ήχος και στα 17, 50, 58, 32, 11, 13 απλά ακουγόταν κάποιος να μιλάει .



Συνδεσμολογία:

- Ο αισθητήρας έχει 4 pin από τα οποία χρησιμοποιήθηκαν τα 3. Το A0 του , το Gnd και το (+) του αισθητήρα συνδέθηκαν στα αντίστοιχα του arduino δηλαδή στο A0, Gnd , και 5V .



Ηχειάκι (Buzzer)

Ενα sketch που με χρήση ενός buzzer ακούγεται μια μελωδία , δηλαδή μια ακολουθία από νότες :

```
#define c 3830 // 261 Hz
#define d 3400 // 294 Hz
#define e 3038 // 329 Hz
#define f 2864 // 349 Hz
#define g 2550 // 392 Hz
#define a 2272 // 440 Hz
#define b 2028 // 493 Hz
#define C 1912 // 523 Hz

#define R 0
int speakerOut = 2;
int DEBUG = 1;

void setup() {
  pinMode(speakerOut, OUTPUT);
  if (DEBUG) {
    Serial.begin(9600); // Set serial out if we want debugging
  }
}

int melody[] = { C, b, g, C, b, e, R, C, c, g, a, C };
int beats[] = { 16, 16, 16, 8, 8, 16, 32, 16, 16, 16, 8, 8 };
int MAX_COUNT = sizeof(melody) / 2; // Melody length, for looping.

long tempo = 10000;
int pause = 1000;
int rest_count = 100; //--BLETCHEROUS HACK: See NOTES

int tone_ = 0;
int beat = 0;
long duration = 0;

void playTone() {
  long elapsed_time = 0;
  if (tone_ > 0) { // if this isn't a Rest beat, while the tone has
    // played less long than 'duration', pulse speaker HIGH and LOW
    while (elapsed_time < duration) {
      digitalWrite(speakerOut, HIGH);
      delayMicroseconds(tone_ / 2);

      // DOWN
      digitalWrite(speakerOut, LOW);
      delayMicroseconds(tone_ / 2);

      // Keep track of how long we pulsed
      elapsed_time += (tone_);
    }
  } else { // Rest beat: loop times delay
    for (int j = 0; j < rest_count; j++) { // See NOTE on rest_count
      else { // Rest beat: loop times delay
        for (int j = 0; j < rest_count; j++) { // See NOTE on rest_count
          delayMicroseconds(duration);
        }
      }
    }
  }
}

void loop() {
  // Set up a counter to pull from melody[] and beats[]
  for (int i=0; i<MAX_COUNT; i++) {
    tone_ = melody[i];
    beat = beats[i];

    duration = beat * tempo; // Set up timing

    playTone();
    // A pause between notes...
    delayMicroseconds(pause);

    if (DEBUG) { // If debugging, report loop, tone, beat, and duration
      Serial.print(i);
      Serial.print(":");
      Serial.print(beat);
      Serial.print(" ");
      Serial.print(tone_);
      Serial.print(" ");
      Serial.println(duration);
    }
  }
}
```

✓ Είναι ένας ενδεικτικός κώδικας που απλά με τη βοήθεια ενός buzzer ακούγετε μια μελωδία .

Αν κάποιος έχει διαβάσει τα προηγούμενα παραδείγματα θα μπορέσει να καταλάβει έστω τι περίπου γίνεται .

■ Η σύνδεση (όπως φαίνεται και στην κάτω εικόνα) είναι πολύ απλή.
Το ένα πόδι συνδέεται με τη γείωση και το άλλο με το pin 2 του arduino .

Στην αριστερή εικόνα είναι απλά ένα buzzer και στην δεξιά μια έξοδος του παρά πάνω sketch :



```
1:16 2028 160000
2:16 2550 160000
3:8 1912 80000
4:8 2028 80000
5:16 3038 160000
6:32 0 320000
7:16 1912 160000
8:16 3830 160000
9:16 2550 160000
10:8 2272 80000
11:8 1912 80000
0:16 1912 160000
1:16 2028 160000
2:16 2550 160000
3:8 1912 80000
4:8 2028 80000
5:16 3038 160000
6:32 0 320000
7:16 1912 160000
8:16 3830 160000
9:16 2550 160000
10:8 2272 80000
11:8 1912 80000
0:16 1912 160000
1:16 2028 160000
2:16 2550 160000
3:8 1912 80000
4:8 2028 80000
5:16 3038 160000
6:32 0 320000
7:16 1912 160000
8:16 3830 160000
```



Κουμπί (Button)

Ένα sketch που μας δείχνει ένα τρόπο για να αξιοποιήσουμε ένα κουμπί – πλήκτρο με το να το προσδομοιώσουμε σε ένα ψηφιακό διακόπτη (Στην περίπτωση μας απλά ανοιγοκλείνει ένα led) :

```
const int buttonPin = 7;
const int ledPin = 9;
int buttonState = 0;
int ledLight = 0 ;
bool firstTime = true ;

void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT);
  digitalWrite(ledPin, ledLight);
  Serial.begin(9600) ;
}

void loop() {
  buttonState = digitalRead(buttonPin);

  if (buttonState == HIGH && firstTime) {
    if (ledLight==255)
      ledLight = 0 ;
    else
      ledLight = 255 ;

    digitalWrite(ledPin, ledLight);
    Serial.println((ledLight==255)? "Anoixe" : "Ekleise") ;
    firstTime = false ;
  } else if (buttonState == LOW){
    firstTime = true ;
  }
}
```

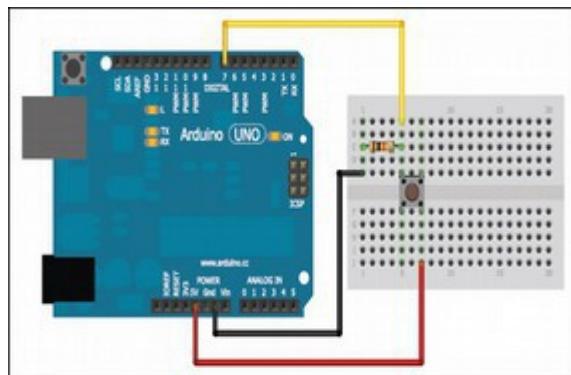


→ Μέχρι την “loop” είναι όλα εύκολα κατανοητά και επεξηγημένα σε άλλα παραδείγματα .

→ Μέσα στη “loop” πρακτικά όποτε πατηθεί το κουμπί (γίνει HIGH) αλλάζει η κατάσταση με την αντίθετη της κάθε φορά.

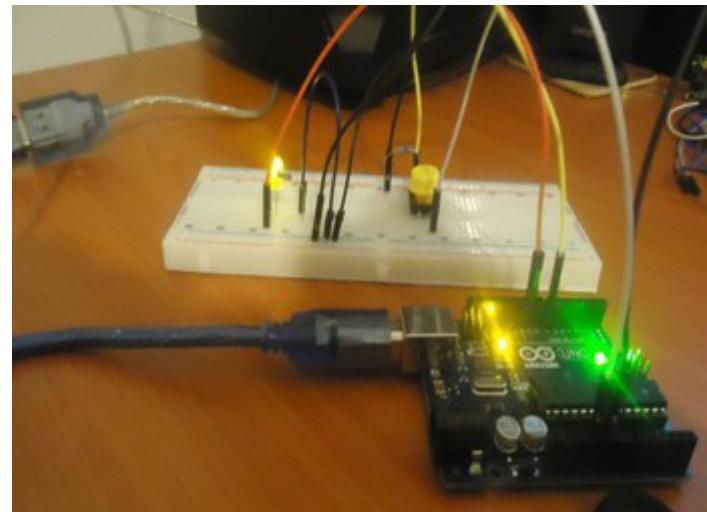
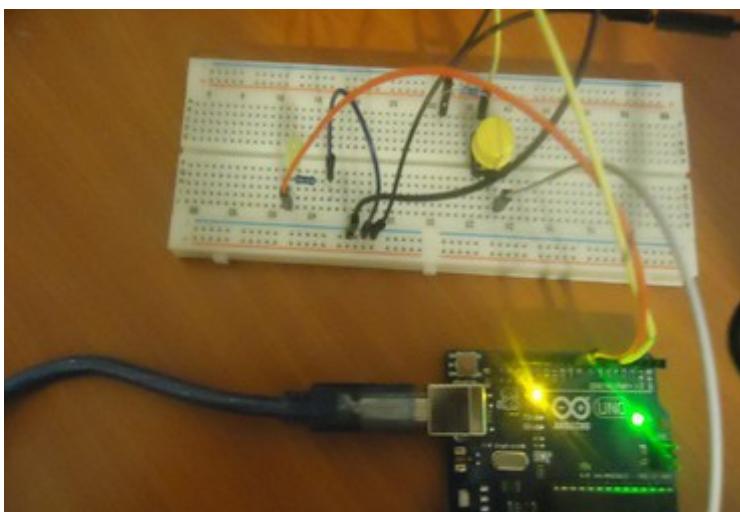
Το firstTime υπάρχει για να αλλάζει μόνο μια φορά όταν πατηθεί το κουμπί (Αν δεν υπήρχε , όσο θα ήταν πατημένο το κουμπί θα γινόταν εναλλαγή της κατάστασης – θα αναβόσβηνε).

Διαβάζεται “LOW” όταν δεν πατιέται το κουμπί.



- ✗ Η σύνδεση του κουμπιού έγινε όπως φαίνεται δίπλα στην φωτογραφία .
Το led συνδέθηκε με τον ίδιο τρόπο με τα προηγούμενα παραδείγματα (βλ. πρώτο παράδειγμα).

Στις 2 από κάτω φωτογραφίες φαίνονται οι δύο καταστάσεις του “διακόπτη” :





Άλλες μονάδες και εξαρτήματα

Είναι διαθέσιμα και άλλα εξαρτήματα και θα μπορούσαν να αξιοποιηθούν με το να χρησιμοποιηθούν στις συσκευές εποπτείας της πτυχιακής ,αλλά ενδεικτικά παρουσιάστηκαν μόνο τα παραπάνω .

Για παράδειγμα, θα μπορούσε να χρησιμοποιηθεί ένα ψηφιακό θερμόμετρο(1) αλλά λόγω του ότι έγινε χρήση του αναλογικού θερμόμετρου και του αισθητήρα υγρασίας & θερμοκρασίας δεν ήταν αναγκαίο κάτι τέτοιο.

Επίσης, θα μπορούσε να γίνει χρήση και ενός RGB Led(2) αλλά επειδή ουσιαστικά χειρίζεται σαν 3 διαφορετικά led (Red,Green,Blue) και πρακτικά ορίζεται η φωτεινότητα για το κάθε ένα χρώμα (ούτως ώστε ο συνδυασμός τους να εμφανίσει οποιαδήποτε άλλο χρώμα) , δεν κρίθηκε αναγκαίο να χρησιμοποιηθεί .

Ακόμη θα μπορούσε να αξιοποιηθεί και ένα ποτενσιόμετρο(3) ώστε για παράδειγμα να δύναται σε κάποιο φυσικό πρόσωπο να αλλάζει την φωτεινότητα ενός led απλά στρέφοντας το.

Θα ήταν αρκετά ενδιαφέρον επιπλέον να γινόταν χρήση και ενός αισθητήρα IR(4) μαζί με το τηλεχειριστήριο(4) του , που μέσω αυτών θα υπήρχε η δυνατότητα να αλλάζουν οι καταστάσεις διάφορων λαμπών και συσκευών με το τρόπο που αλλάζουν και τα κανάλια στην τηλεόραση .

Άλλά λόγω του ότι η πτυχιακή εργασία αποσκοπεί κυρίως στο κομμάτι απομακρυσμένης εποπτείας και όχι στο σχεδιασμό ενός έξυπνου σπιτιού, δεν δόθηκε βάρος σε περαιτέρω εξέλιξη.

Στην εικόνα από κάτω φαίνονται τα τέσσερα προαναφερθέντα εξαρτήματα :





Πρότυπο sketch για επικοινωνία με το Device Client

Ο κώδικας από το πρότυπο sketch που έχει γραφτεί, υποθετικά έχει 8 (θεωρητικά) σημεία από τα οποία στα **2 πιθανόν** να επέμβει κάποιος χρήστης ανάλογα τις ανάγκες του , στα **4 πρέπει** να τα προσαρμόσει σύμφωνα με τις μονάδες που έχει και τα υπόλοιπα **2 δεν** τα πειράζει.

Ας αναφερθούν πιο αναλυτικά αυτά τα 8 (θεωρητικά) σημεία , για να υπάρχει μια καλύτερη εικόνα:

- 1 Στο 1ο μέρος δηλώνονται οι διάφορες βιβλιοθήκες που μπορεί να χρειάζονται κάποιες από τις μονάδες που θα χρησιμοποιηθούν (όπως για παράδειγμα η DHT.h για τον αισθητήρα υγρασίας). Όπως επίσης μπορούν να δηλωθούν (αν θέλει κάποιος) σταθερές (define) .
 - 2 Στο 2o μέρος βάζει (κάποιος χρήστης) στη σταθερή μεταβλητή “SUMOFUNITS” το πλήθος όλων των μονάδων που θα χρησιμοποιήσει και στη μεταβλητή “controllerName” βάζει το όνομα του ελεγκτή του (που πρέπει να είναι μοναδικό αν έχει πάνω από ένα arduino) .
 - 3 Στο 3o μέρος δεν πειράζει (κάποιος χρήστης) την μεταβλητή για το **όνομα της συσκευής** , το **Version** του sketch όπως επίσης και τη **κλάση Unit** .
 - 4 Στο 4o μέρος μπορεί αν χρειάζεται να δηλωθούν ότι global μεταβλητές θέλει ο χρήστης.
 - 5 Στο 5o μέρος προσαρμόζεται κατάλληλα η **συνάρτηση setup**.
(1)Αρχικά πρέπει υποχρεωτικά να καθοριστεί το bit rate του σειριακού στα 9600bps.
(2)Επίσης, αυτό που πρέπει να υπάρχει στο σώμα της setup είναι οι αρχικοποιήσεις όλων των μονάδων,& πραγματοποιείται αυτό εκτελώντας για κάθε μονάδα την συνάρτηση setAll(...) με τις κατάλληλες παραμέτρους της (θα αναλυθεί το κομμάτι αυτό στη συνέχεια).
(3)Ακόμη, πρέπει να δηλωθούν ρητά για το κάθε pin που θα χρησιμοποιηθεί αν θα είναι για input ή output [με την συνάρτηση : “pinMode(...)”].
(4)Επιπλέον μπορεί (κάποιος χρήστης) να γράψει ότι άλλο θέλει στην συνάρτηση setup εφόσον είναι απαραίτητο για τις συγκεκριμένες μονάδες που έχει .
(όπως πχ για τον αισθητήρα υγρασίας πρέπει να εκτελεστεί η μέθοδος : “dht.begin();”).
 - 6 Στο 6o μέρος προσαρμόζεται κατάλληλα η **συνάρτηση loop** .
Πιο συγκεκριμένα , αν πχ υπάρχουν αισθητήρες , γίνονται όλες οι μετρήσεις και στο τέλος στέλνεται στην σειριακή έξοδο ένα μήνυμα σε μορφή του “πρωτοκόλλου” της Π.Ε .
Βέβαια, υπάρχουν μέσα στη συνάρτηση κάποια μικρά κομμάτια κώδικα που είναι του πρότυπου και δεν πρέπει να το αλλάξει κάποιος (θα τα επισυμανθούν στη συνέχεια αναλυτικά).
 - 7 Στο 7o μέρος προσθέτονται οι συναρτήσεις για τις εργασίες των διαφόρων μονάδων .
Για παράδειγμα , αν για να βρεθεί μια μέτρηση ενός αισθητήρα χρειάζονται πάνω από μια γραμμές κώδικα , καλό είναι να φτιαχτεί μια συνάρτηση που να επιστρέφει μόνο την μέτρηση και που στο σώμα της να γίνεται όλη η διαδικασία που χρειάζεται (οι πάνω από μια γραμμές κώδικα δηλαδή).
 - 8 Στο 8o μέρος δεν πρέπει να πειράζει κάποιος χρήστης κάτι.
Δηλαδή δεν πειράζει (κάποιος χρήστης) την **readBytesFromUSB()** συνάρτηση και τις μεθόδους τις **κλάσης Unit** όπως επίσης και τη συνάρτηση **findUnit()** .
- ✓ Κάθε αντικείμενο της κλάσης Unit έχει μαζεμένες όλες τις απαραίτητες πληροφορίες για μια μονάδα όπως επίσης έχει μεθόδους για να χειρίζεται μια μονάδα ευκολότερα.
✓ Στο πρότυπο που έχει φτιαχτεί, υπάρχει ένας πίνακας τύπου Unit (μεγέθους όσο είναι το SUMOFUNITS) με όλες τις μονάδες : “units[SUMOFUNITS]”.



Η Κλάση Unit έχει:

- 1 σταθερά (Την έκδοση του προτύπου sketch για να γνωρίζει ο εκάστοτε DeviceClient)
- 12 μεταβλητές.
 - Οι 8 είναι για τα χαρακτηριστικά της κάθε μονάδας , δηλαδή το name,tag,type,value,maxValue,minValue και το limit (ανατρέξτε αν θέλετε στη σελίδα 24 που λέει αναλυτικά για το κάθε ένα τι είναι, στη παρουσίαση του Simulator)
 - Οι 2 είναι για τα pin που μπορεί να έχει η μονάδα (συνήθως χρησιμοποιείται μόνο το 1)
 - Η μία είναι για να δηλωθεί αν είναι αναλογική ή ψηφιακή η μονάδα
 - Η τελευταία είναι για το diff. Δηλαδή τη διαφορά που πρέπει να υπάρχει από μια μέτρηση με την προηγούμενη της για να θεωρηθεί διαφορετική τιμή. Για παράδειγμα, μπορεί οι μετρήσεις κάποιου αισθητήρα να αλλάζουν συνεχώς κατά +-0.2 και εμείς να μην θέλουμε να στέλνει συνέχεια σαν νέα τιμή . Έτσι μπορούμε πχ να καθορίσουμε το diff να είναι 0.5 και έτσι μόνο αν μια μέτρηση είναι μεγαλύτερη ή μικρότερη κατά 0.5 θα θεωρείτε ότι άλλαξε η τιμή της μονάδας και επομένως δεν θα στέλνει στο DeviceClient συνέχεια, αλλά μόνο όταν θα υπάρχει “ουσιαστική” αλλαγή . (όπως επίσης μπορούμε να βάλουμε και diff = 1 για κάποιο θερμόμετρο ...)
- 7 συναρτήσεις
 - bool setValue(String unitValue) ; [Άλλάζει το value της μονάδας και τη τιμή στο pin]
 - bool setMode(double unitValue) ; [Άλλάζει το value της μονάδας και τη τιμή στο pin]
 - bool setMode(String unitMode); [Άλλάζει το mode της μονάδας]
 - String getValue(); [Επιστρέφει μέσα σε παρένθεση το όνομα του arduino, το όνομα της μονάδας και τη τιμή της μονάδας με “|” ανάμεσα τους]
 - String getMode(); [Επιστρέφει μέσα σε παρένθεση το όνομα του arduino, το όνομα της μονάδας και το mode της μονάδας με “|” ανάμεσα τους]
 - String getInfo(); [Επιστρέφει μέσα σε παρένθεση τα 8 χαρακτηριστικά της μονάδας με “|” ανάμεσα τους (Δεν χρειάζεται να χρησιμοποιηθεί από απλό χρήστη...)]
 - void setAll(String na,String ta,int ty,int mo,String va,double ma,double mi,String li,int pi1,int pi2,bool isAnal,float di) ; [Ουσιαστικά αρχικοποιεί όλες τις μεταβλητές της μονάδας. Οι πρώτες 8 παράμετροι είναι τα χαρακτηριστικά της μονάδας , οι 2 επόμενες είναι τα pin και οι 2 τελευταίες είναι μια παράμετρος bool για το αν είναι αναλογική η μονάδα και μια παράμετρος για το diff]

Συνοπτικά να αναφερθούν και δυο λόγια για τις συναρτήσεις που δεν είναι ανάγκη να γνωρίζει κάποιος χρήστης , δηλαδή τη readBytesFromUSB() και την findUnit().

Πρακτικά η “**readBytesFromUSB()**” διαβάζει το πρώτο γράμμα από ένα μήνυμα που έχει στείλει ο DeviceClient και ανάλογα με το γράμμα κάνει με τη βοήθεια ενός if τις κατάλληλες διαδικασίες .

Πχ Αν το πρώτο γράμμα είναι “m” τότε γίνεται η διαδικασία να διαβαστεί το υπόλοιπο μήνυμα από το USB ώστε να αλλάξει mode μια μονάδα .

Επίσης, μπορεί να διαβάσει σαν πρώτο γράμμα το “c” (που είναι για αλλαγή value), το “b” (που “λέει” στο arduino να αρχίσει να στέλνει) , το “e” (που “λέει” στο arduino να σταματήσει να στέλνει), το “v” (για να απαντήσει το arduino το VERSION του), το “d” (για να αποκτήσει το arduino device name σύμφωνα με το server...) , το “i” (για να στείλει το arduino την αρχικοποίηση όλων των μονάδων σύμφωνα με το πρωτόκολλο) και το “n” (για να αλλάξει όνομα το arduino – πχ όταν 2 arduino με ίδιο όνομα είναι συνδεδεμένα στο DeviceClient...).

Τα περισσότερα τα κάνει στην αρχή της σύνδεσης με το DeviceClient για να υπάρξει συγχρονισμός και “στήσιμο” της επικοινωνίας .Στην κανονική ροή συνήθως τα μηνύματα ξεκινάμε με “c” & “m”.

Η “**findUnit()**” ουσιαστικά με βάση ένα όνομα μονάδας επιστρέφει το αντίστοιχο αντικείμενο της συγκεκριμένης μονάδας που βρίσκεται μέσα στο πίνακα **units[]** .



Ας δούμε ένα **παράδειγμα** του πρότυπου , όταν έχει 2 μονάδες,ένα sonar & ένα φωτοαισθητήρα :

1ο βήμα είναι να δηλώσουμε ότι το SUMOFUNITS είναι 2 και το controllerName ας πούμε θα είναι "Ard1"

2ο βήμα είναι , στη συνάρτηση "**setup**" (εκτός από το bitrate που το κάνουμε να είναι στα 9600) να setάρουμε κατάλληλα το units[0] και το units[1] .

Πχ εκτελούμε το : **units[0].setAll("Sonar","","5,0,"0",3000,0,"NL",4,3,false,0.0);**

Name = "Sonar" ,tag = "", type = 5(Distance), mode = 0 (auto), value = "0" ,Max=3000 ,Min=0 , limit="NL" (no Limit) ,pin1 = 4 (trig), pin2 = 3 (echo), isAnalog = false ,diff = 0.0

Και το **units[1].setAll("Fotoaistitiras","","3,0,"0",255,0,"NL",A3,-1,true,0.0);**

Name = "Fotoaistitiras" ,tag = "", type = 3(Brightness), mode = 0 (auto), value = "0" ,Max=255 ,Min=0 , limit="NL" (no Limit) ,pin1 = A3, pin2 = -1 (Δεν χρησ.), isAnalog = true ,diff = 0.0

Όπως φάνηκε στα προηγούμενα παραδείγματα πρέπει να γίνουν οι ανάλογες συνδέσεις του arduino με τους αισθητήρες . Δηλαδή να συνδεθούν τα Ground και τα 5V όπως επίσης και τα pin του sonar στα pin 4,3 του arduino και το pin του φωτοαισθητήρα στο A3 του arduino.

Επίσης, μέσα στη "setup" πρέπει να καθοριστούν και τα pin αν θα είναι εισόδου ή εξόδου . Στην περίπτωση μας τα pin 3 και A3 είναι εισόδου και το pin 4 είναι εξόδου .

Όλα τα προηγούμενα φαίνονται στις επόμενες δύο φωτογραφίες :

```
//Αλλαγή των 2 από κάτω μεταβλητών ανάλογα την περίπτωση μας
const int SUMOFUNITS = 2 ;
String controllerName = "Ard1" ;
//Δεν πειράζουμε το κώδικα του προτύπου : Begin
String deviceName = "DRC" ;
class Unit
{
    ...
    ...
    //Δεν πειράζουμε το κώδικα του προτύπου : End

void setup()
{
    //Δεν πειράζουμε το κώδικα του προτύπου : Begin
    Serial.begin(9600);
    //Δεν πειράζουμε το κώδικα του προτύπου : End
    units[0].setAll("Sonar","","5,0,"0",3000,0,"NL",4,3,false,0.0);
    units[1].setAll("Fotoaistitiras","","3,0,"0",255,0,"NL",A3,-1,true,0.0);
    pinMode (3, INPUT); //Echo
    pinMode (4, OUTPUT); //Trig
    pinMode (A3, INPUT);
}
void loop()
```



3ο βήμα τώρα, με βάση τα παραδείγματα που έχουν παρουσιαστεί του φωτοαισθητήρα και του sonar, πρέπει να δημιουργηθούν 2 συναρτήσεις που απλά θα επιστρέφουν τις μετρήσεις από τους αισθητήρες.

Για παράδειγμα, δημιουργούνται οι δυο συναρτήσεις από την δίπλα φωτογραφία :

```
//Συναρτήσεις ανάλογα τις μονάδες μας :
double getBrightnessFromPhotosensor(int pin)
{
    double sensorValue = analogRead (pin);
    return sensorValue;
}

long getCMFromSonar(int trig,int echo)
{
    long duration;

    digitalWrite(trig, LOW);
    delayMicroseconds(2);
    digitalWrite(trig, HIGH);
    delayMicroseconds(5);
    digitalWrite(trig, LOW);
    duration = pulseIn(echo, HIGH);

    // convert the time into a distance
    int result = duration / 29 / 2;
    if (result>=1000)
        return 0;
    else
        return result ;
}

//Τέλος συναρτήσεων των μονάδων μας!

//ΔΕΝ ΡΕΙΣΑΖΟΥΜΕ το κώδικα του προτύπου : Begin
void readBytesFromUSB()
```

4ο βήμα και τελευταίο είναι να προσαρμοστεί η συνάρτηση loop και πιο συγκεκριμένα να γραφτεί κώδικας για τις 2 περιπτώσεις του παραδείγματος, ώστε μετά από τις κάθε μετρήσεις να δημιουργείται ένα μήνυμα (με βάση το πρωτόκολλο) που θα στέλνετε στον DeviceClient .

```
void loop()
{
//δεν μετράζουμε το κώδικα του προτύπου : Begin
    timer++;
    if (Serial.available() > 0) {
        readBytesFromUSB();
        timer=0;
    }
    if (timer==5000)
        timer=0;

    if ((sending) && (timer==0))
    {
        int sumOfParam = 0 ;
        String parameters = "" ;
//δεν μετράζουμε το κώδικα του προτύπου : End

        delay(3) ;
        float cm = getCMFromSonar(units[0].pin1,units[0].pin2) ;
        if ((units[0].value-units[0].diff) > cm || (units[0].value+units[0].diff) < cm)
        {
            sumOfParam += 1 ;
            units[0].value = cm ;
            parameters += units[0].getValue();
        }
        delay(3) ;
        double Brightness = getBrightnessFromPhotosensor(units[1].pin1) ;

        if ((units[1].value-units[1].diff) > Brightness || (units[1].value+units[1].diff) < Brightness)
        {
            sumOfParam += 1 ;
            units[1].value = Brightness ;
            parameters += units[1].getValue();
        }

//δεν μετράζουμε το κώδικα του προτύπου : Begin
        if (sumOfParam>0)
        {
            String post = "#"+deviceName+"#NewValues:"+ String(sumOfParam)+"#" ;
            Serial.println(post+parameters+"#");
        }
//δεν μετράζουμε το κώδικα του προτύπου : End
    }
}
//Συναρτήσεις ανάλογα τις μονάδες μας :
double getBrightnessFromPhotosensor(int pin)
```

Επειδή από μέτρηση σε μέτρηση πρέπει να υπάρχει ένα delay (για να μην γίνονται συνεχώς μετρήσεις...), δημιουργήθηκε ένα πρόβλημα, ότι όσο το πρόγραμμα “αδρανεί” δεν είναι σε θέση πάντα να διαβάζει τα μηνύματα από το USB.

Όποτε βγήκε το delay(...) και μπήκε ένας μετρητής (timer) ούτως ώστε να είναι το πρόγραμμα έτοιμο να διαβάζει συνεχώς από το USB και μόνο μετά από 5000 επαναλήψεις της loop να γίνονται οι μετρήσεις. Οπότε με αυτό το τρόπο επιτεύχθηκε αφενός να μην χάνονται μηνύματα από το USB και αφετέρου επιτεύχθηκε η καθυστέρηση που έπρεπε. Ο αριθμός των επαναλήψεων (5000) μπορεί να αλλάξει ανάλογα με τις μονάδες που υπάρχουν. Μπορούν να γίνουν διάφορες δοκιμές.

- Όπως φαίνεται (μέσα στο κόκκινο περίγραμμα) αρχικά παίρνεται μια μέτρηση από το sonar και στη συνέχεια γίνεται ένας έλεγχος για το αν η μέτρηση είναι μεγαλύτερη ή μικρότερη από τη προηγούμενη (περνώντας υπόψιν και το diff που βέβαια στη περύπτωση μας είναι 0). Έτσι αν υπάρξει καινούρια τιμή, η μεταβλητή sumOfParam αυξάνεται κατά ένα και το sonar παίρνει τη νέα τιμή (units[0].value) . Επιπλέον, προσθέτεται η επιστροφή από το getValue() στη μεταβλητή parameters ώστε να χτιστεί στο τέλος το μήνυμα που θα παραλάβει ο DeviceClient . Η ίδια ακριβός λογική ισχύει και για τον φωτοαισθητήρα .
- Η μεταβλητή sending χρησιμοποιείται πληστή στη περύπτωση που θέλει ο DeviceClient να σταματήσει να στέλνει μηνύματα το arduino μας ...



Μια φωτογραφία από το προηγούμενο παράδειγμα όπως φαίνεται από ένα UserClient:



Αν υποθετικά θέλουμε να έχουμε και ένα led ή ένα πορτατίφ στο παράδειγμα μας , το μόνο που πρέπει να γίνει είναι να αλλαχθεί το SUMOFUNITS (= 3) και να προσθεθούν 2 γραμμές στο setup , όπως φαίνεται στην πρώτη παρακάτω φωτογραφία (στη δεύτερη είναι ο UserClient) :

```
void setup()
{
    //Δεν πειράζουμε το κώδικα του προτύπου : Begin
    Serial.begin(9600);
    //Δεν πειράζουμε το κώδικα του προτύπου : End
    units[0].setAll("Sonar","","5.0","0",3000,0,"NL",4,3,false,0.0);
    units[1].setAll("Fotoaistitiras","","3.0","0",255,0,"NL",A3,-1,true,0.0);
    units[2].setAll("myLed","","2.3","255",255,0,"1TB",9,-1,true,0.0);
    pinMode (3, INPUT); //Echo
    pinMode (4, OUTPUT); //Trig
    pinMode (9, OUTPUT); //Led
    pinMode (A3, INPUT);
}
```



- Λίγο-πολύ , αυτή είναι η λογική του προτύπου . Στο Github υπάρχουν εκτός από το πιο πάνω παράδειγμα και δυο ακόμη πρότυπα sketch που καλύπτουν όλες τις μονάδες που έχουν δειχτεί σε όλα τα παραδείγματα της πτυχιακής εργασίας .
Κάποιες περιπτώσεις μονάδων μέσα στα πρότυπα sketch είναι λίγο πιο “ιδιόρρυθμες” , οπότε είναι λίγο πιο περίπλοκοι οι κώδικες από ότι σε αυτό με τους δυο αισθητήρες .
Παρά ταύτα όμως , μέσα από τη τεκμηρίωση αυτή καλύφθηκε η βάση που χρειάζεται να έχει κάποιος για να τα καταλάβει αλλά και να τα αλλάξει σύμφωνα με τις ανάγκες του.
- Αν κάποιος θέλει να χρησιμοποιήσει διαφορετικού τύπου μονάδες πιθανόν να πρέπει να βάλει λίγο φαντασία και εφευρετικότητα ώστε να μπορέσει να προσαρμόσει κατάλληλα το πρότυπο sketch με βάση τις δικές του απαιτήσεις .



Μια περίπτωση πραγματικής χρήσης του συστήματος

Σε αυτό το σημείο θα δειχθεί πως με το τελικό ολοκληρωμένο σύστημα πραγματοποιούνται τα αρχικά σενάρια χρήσης που καθορίστηκαν στις προδιαγραφές αυτής της πτυχιακής εργασίας. Για πρακτικούς λόγους βέβαια δεν θα παρουσιαστεί μια περίπτωση που θα λειτουργεί μέσα από το διαδίκτυο αλλά μια περίπτωση που θα δουλεύει μέσα σε ένα τοπικό δίκτυο(LAN). Μπορεί να έχουν μειωθεί οι αποστάσεις αλλά δεν έχει μειωθεί καθόλου η ουσία διότι δεν υπάρχει καμία απολύτως διαφορά πέρα του ότι χρησιμοποιούνται τοπικές διευθύνσεις iρ και όχι πραγματικές.

- ✓ Να αναφερθεί ότι το σύστημα μπορεί όντως να λειτουργήσει απομακρυσμένα μέσω του διαδικτύου διότι έχει δοκιμαστεί η περίπτωση που ο Server(ο οποίος είχε πραγματική iр μετά από port-forward στο router...) και ο DeviceClient ήταν στην Αθήνα και κάποιος είτε από τη Κρήτη είτε και από τη Γερμανία εποπτεύε σε πραγματικό χρόνο με τη βοήθεια του UserClient. Επίσης, παρατηρήθηκε ότι μέσω διαδικτύου (λόγω του ότι μεταφέρεται απλό κείμενο μέσα από τα socket) οι ταχύτητες ενημέρωσης εκατέρωθεν ήταν ίδιες ακριβώς με αυτές που υπάρχουν και σε ένα τοπικό δίκτυο (LAN) και επομένως η **απόδοση** ήταν ίδια.

Έτσι θα παρουσιαστεί πως μπορούν βήμα-βήμα να εκπληρωθούν τα αρχικά σενάρια χρήσης :

- 1) Αρχικά ,ένας “επισκέπτης” μέσω της ιστοσελίδα (a)δημιουργεί ένα λογαριασμό χρήστη (“newUser”) και στη συνέχεια (b)συνδέεται με αυτόν για να (c)δημιουργήσει ένα λογαριασμό συσκευής (“serres”).

(a)

Δημιουργία Νέου Λογαριασμού χρήστη:

(c)

(b)



Δημιουργία Νέου Λογαριασμού Συσκευής:



- 2) Σε επόμενη φάση ο νέος χρήστης (“newUser”) κάνει (a) πρόσκληση στο χρήστη maria για να αποκτήσει και αυτή δικαιώματα (συγκεκριμένα administrator rights) πάνω στη συσκευή “serres” που μόλις έφτιαξε. Επίσης, με τη βοήθεια της αναζήτησης (b) ψάχνει την συσκευή “andros” και κάνει αίτηση για να αποκτήσει δικαιώματα Administrator σε αυτήν.
Και τέλος, (c) βλέπει τις αιτήσεις που τον αφορούν.

(a)

| Device: | Comment: | Right: | Owner: | Edit Device | Edit Access | Release Device |
|---------|--------------------|---------|---------|-------------|-------------|----------------|
| serres | Magazi stis serres | (Owner) | newUser | | | |

Δικαιώματα συσκευής:
serres

| User: | Current Right: | Owner: | Admin: | Full Access: | Read only: | None: | Option: | Request Status: |
|----------|----------------|-----------------------|----------------------------------|-----------------------|-----------------------|----------------------------------|---------|-----------------|
| newUser | | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | | No Request |
| admin | | <input type="radio"/> | <input checked="" type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input checked="" type="radio"/> | | No Request |
| georgios | | <input type="radio"/> | <input checked="" type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input checked="" type="radio"/> | | No Request |
| maria | | <input type="radio"/> | <input checked="" type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input checked="" type="radio"/> | | 1 No Request |
| mikeole | | <input type="radio"/> | <input checked="" type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input checked="" type="radio"/> | | No Request |

Στην κάτω φώτο φαίνεται στο “Request Status” η εκκρεμή πρόσκληση για το χρήστη maria:

| | | | | | | | | |
|-------|--|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|--|---------------|
| maria | | <input type="radio"/> | | Administrator |
|-------|--|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|--|---------------|

(b)

Search Device Name : 1

Device Name

Βρέθηκε η συσκευή : [andros]

Γράψε κάποιο σχόλιο και κάνε αίτηση για να αποκτήσεις δικαιώματα στην συσκευή :
Tha mou doseis dikaiomata?

2

3

Administrator Full Access Read Only

(c)

| Προσκλήσεις που έχω κάνει για να πάρουν δικαιώματα κάποιοι άλλοι χρήστες στις συσκευές μου: | | | | |
|---------------------------------------------------------------------------------------------|-------|-------------|--------------------------------|---------|
| Device: | User | Right: | Invite: | Delete: |
| serres | maria | (Admin) | You want to have this device ? | |

| Αιτήσεις που έχω κάνει για να πάρω δικαιώματα σε μια νέα συσκεύη: | | | |
|-------------------------------------------------------------------|-------------|----------------------------|---------|
| Device: | Right: | Request: | Delete: |
| andros | (Admin) | Tha mou doseis dikaiomata? | |



3. Μόλις γίνουν όλες οι προηγούμενες ενέργειες, συνδέεται στην ιστοσελίδα μας ο χρήστης "maria" και (a) αποδέχεται την πρόσκληση και την αίτηση που έχει κάνει ο "newUser". Σε επόμενο χρόνο, ο χρήστης "newUser" (b) βλέπει στο "ManageDevices" την συσκευή "andros", όπως επίσης και τα δικαιώματα χρηστών πάνω στη συσκευή "serres".

(a)

| Αιτήσεις Χρηστών που θέλουν να αποκτήσουν δικαιώματα στις συσκευές μου : | | | | | |
|--------------------------------------------------------------------------|---------|--------------------------|----------------------------|-------------------------------------|--------------------------|
| Device: | User: | Right: | Request: | Accept: | Reject: |
| andros | newUser | Administrator (Admin) | Θα μου δοσεις δικαιοματα? | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| andros | mikeole | Read (Read) | Θα μου δώσεις δικαιώματα ; | <input checked="" type="checkbox"/> | <input type="checkbox"/> |

| Προσκλήσεις από άλλους χρήστες για να δεχτώ δικαιώματα σε νέες συσκευές: | | | | | |
|--------------------------------------------------------------------------|--------------------------|----------|--------------------------------|-------------------------------------|--------------------------|
| Device: | Right: | User: | Invite: | Accept: | Reject: |
| margarites | Full (Full) | georgios | You want to have this device ? | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| serres | Administrator (Admin) | newUser | You want to have this device ? | <input checked="" type="checkbox"/> | <input type="checkbox"/> |

| Προσκλήσεις που έχω κάνει για να πάρω δικαιώματα κάποιοι άλλοι χρήστες στις συσκευές μου: | | | | | |
|-------------------------------------------------------------------------------------------|---------|----------------|----------------------------|-------------------------------------|--------------------------|
| Device: | User: | Right: | Request: | Accept: | Delete: |
| andros | mikeole | Read (Read) | Θα μου δώσεις δικαιώματα ; | <input checked="" type="checkbox"/> | <input type="checkbox"/> |

Στην κάτω φώτο φαίνεται το "My Requests" της "maria", μετά τις αποδοχές που έκανε :

| Αιτήσεις Χρηστών που θέλουν να αποκτήσουν δικαιώματα στις συσκευές μου : | | | | | |
|--------------------------------------------------------------------------|---------|----------------|----------------------------|-------------------------------------|--------------------------|
| Device: | User: | Right: | Request: | Accept: | Reject: |
| andros | mikeole | Read (Read) | Θα μου δώσεις δικαιώματα ; | <input checked="" type="checkbox"/> | <input type="checkbox"/> |

| Προσκλήσεις από άλλους χρήστες για να δεχτώ δικαιώματα σε νέες συσκευές: | | | | | |
|--------------------------------------------------------------------------|----------------|----------|--------------------------------|-------------------------------------|--------------------------|
| Device: | Right: | User: | Invite: | Accept: | Reject: |
| margarites | Full (Full) | georgios | You want to have this device ? | <input checked="" type="checkbox"/> | <input type="checkbox"/> |

(b)

| Device: | Comment: | Right: | Owner: | Edit Device | Edit Access | Release Device |
|---------|--------------------------|--------------------------|---------|-------------|-------------|----------------|
| andros | To κάστρο μου στην Ανδρο | Administrator (Admin) | maria | | | |
| serres | Magazi stis serres | Owner (Owner) | newUser | | | |

| User: | Current Right: | Owner: | Admin: | Full Access: | Read only: | None: | Option: | Request Status: |
|----------|----------------|-----------------------|----------------------------------|-----------------------|-----------------------|-----------------------|---------|-----------------|
| newUser | | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | | No Request |
| maria | | <input type="radio"/> | <input checked="" type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | | No Request |
| admin | | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | | No Request |
| georgios | | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | | No Request |
| mikeole | | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | <input type="radio"/> | | No Request |

Φαίνεται στην από πάνω εικόνα ότι στην συσκευή "serres" έχουν δικαιώματα μόνο 2 χρήστες (newUser = Owner & maria = Admin) . Επίσης, φαίνεται το αλλαγμένο κουμπί της maria που τώρα πια μπορούν να αλλαχθούν(ή&να αφαιρεθούν)τα δικαιώματα της χωρίς να δημιουργείται "αίτηση".

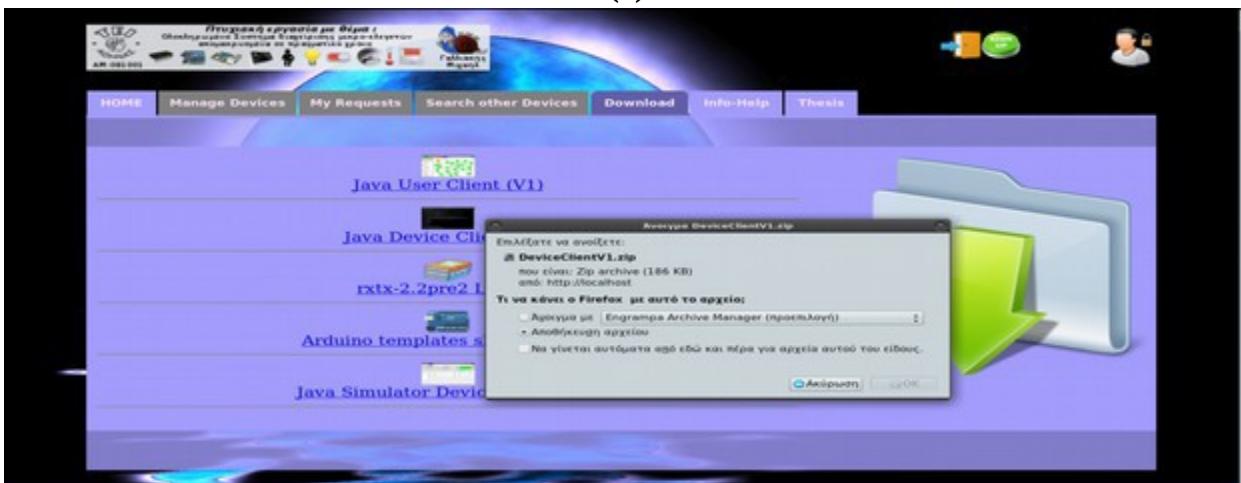


- ✓ Έτσι λοιπόν στο παραπάνω παράδειγμα ,εφόσον έχουν γίνει όλα τα προγενέστερα βήματα , μπορούν δύο διαφορετικοί χρήστες (“maria” και “newUser”) να εποπτεύουν ταυτόχρονα και τις δύο διαφορετικές συσκευές(“serres” και “andros”) δικτυακά (δηλαδή χωρίς άμεση φυσική επαφή ήτοι απομακρυσμένα) και σε πραγματικό χρόνο .

- Σε αυτό το σημείο να αναφερθεί ότι τα πράγματα που πρακτικά είναι διαθέσιμα και που θα χρησιμοποιηθούν για να φανεί στην πράξη ότι μπορούν να εκπληρωθούν τα αρχικά σενάρια χρήστης είναι ένας σταθερός υπολογιστής(desktop) ,ένα laptop , ένα raspberry pi , δύο arduino ,ένα router(ενός πάροχου διαδικτύου για την ύπαρξη LAN) και διάφορες μονάδες που έχουν παρουσιαστεί στο κεφάλαιο που αφορά το Arduino.
(Να διευκρινιστεί ακόμη ότι και οι 3 παραπάνω υπολογιστές έχουν ασύρματη κάρτα δικτύου και ανήκουν στο ίδιο τοπικό δίκτυο)
- Έτσι λοιπόν μέχρι τώρα πρακτικά ο apache Server (για την ιστοσελίδα) μαζί με το MySQL Server εκτελούνται στο Desktop .
- Επίσης, παίρνουμε σαν παραδοχή ότι στο Desktop “τρέχει” ήδη και ο Server του συστήματος , αλλά για λόγους συνοχής δεν παρουσιάζεται σε αυτή τη φάση ο τρόπος εκτέλεσης του , για να υπάρχει μια συνέχεια στη ροή βημάτων των “maria” και “newUser”.

4. Στο επόμενο βήμα πρέπει οι χρήστες “newUser” και “maria” να κατεβάσουν από την ιστοσελίδα ένα template sketch για να το προσαρμόσουν ανάλογα τις μονάδες που θα χρησιμοποιήσουν στα arduino τους, όπως επίσης πρέπει και να κάνουν τη διασύνδεση των arduino με τις μονάδες τους . (Έχει περιγραφεί αναλυτικά ο τρόπος που γίνεται αυτό).
5. Στη συνέχεια πρέπει οι δύο χρήστες (“newUser” και “maria”) να :
 - Κάνουν ο κάθε ένας **(a)**download από την ιστοσελίδα το DeviceClient.
 - Συνδέουν τους υπολογιστές τους (που στο παράδειγμα είναι raspberry pi και ένα laptop) με τα arduino τους.
 - **(b)**Συμπληρώσουν κατάλληλα το xml αρχείο ρυθμίσεων του κάθε DeviceClient.
 - **(c)**Εκτελέσουν ο κάθε ένας το DeviceClient στους υπολογιστές τους .
Να σημειωθεί ότι πρέπει ο υπολογιστής που θα “τρέξει” το DeviceClient να έχει μαζί με τη JavaVirtualMachine και 2 βιβλιοθήκες (RXTX) που είναι απαραίτητες(Περισσότερες πληροφορίες στο Παράρτημα) και τις οποίες μπορούν να βρεθούν και από το website.

(a)





(b)

```
GNU nano 2.2.6 File: PMGTCDC.config

<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<pmgThesisConfigDeviceClient>
<DeviceName>andros</DeviceName>
<Password>andros</Password>
<Address>192.168.2.200</Address>
<Port>50128</Port>
<secondsForNextTryConnect>60</secondsForNextTryConnect>
<secondsForNextSearchArduino>10</secondsForNextSearchArduino>
<viewMessage>False</viewMessage>
</pmgThesisConfigDeviceClient>
```

```
GNU nano 2.2.6 File: PMGTCDC.config

<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<pmgThesisConfigDeviceClient>
<DeviceName>serres</DeviceName>
<Password>serres</Password>
<Address>192.168.2.200</Address>
<Port>50128</Port>
<secondsForNextTryConnect>60</secondsForNextTryConnect>
<secondsForNextSearchArduino>10</secondsForNextSearchArduino>
<viewMessage>True</viewMessage>
</pmgThesisConfigDeviceClient>
```

Η πάνω αριστερή εικόνα είναι για το DeviceClient του laptop και η δεξιά είναι για το raspberry pi. Οπως φαίνεται με τη σειρά είναι : όνομα συσκευής, κωδικός συσκευής, διεύθυνση και πόρτα Server , ανά πόσα δευτερόλεπτα θα κάνει προσπάθεια για νέα σύνδεση αν χαθεί η τωρινή με το server , ανά πόσα δευτερόλεπτα θα “ψάχνει” για νέο arduino (στο -1 δεν ψάχνει ποτέ όπως και για τη περίπτωση της επανασύνδεσης με το server) και τέλος είναι True για να εμφανίζονται μηνύματα και False για να μην εμφανίζονται μηνύματα . Τα μηνύματα που μπορεί κάποιος χρήστης να επιλέξει να μην εμφανίζονται αφορούν κυρίως τα της ροής, δηλαδή αυτά που έρχονται από το arduino σαν αλλαγής κατάστασης (value-mode) και τα αντίστοιχα μηνύματα που έρχονται από το server.

Οι κάτωθεν φωτογραφίες είναι από την εκτέλεση των DeviceClient στο raspberry pi (η πρώτη) και στο laptop (η δεύτερη) :

(c)

```
File Edit View Search Terminal Help
pi@raspberrypi: ~ /gitlab/myThesis/DeviceClientV1 $ sudo java -jar DeviceClientV1.jar
[|P|I|: Thesis Michael Galliakakis 2016. ]
[|P|I|: TEI Athens - IT department ]
[|P|I|: Email : cs081001@teliauth.gr & michaelgalliaikakis@yahoo.gr . ]
[|P|I|: All files can be found in /home/michael/GitLab/myThesis ]
[|P|I|: "https://github.com/michaelgalliaikakis/mythesis.git" ]
[|P|I|: http://raspberrypi/DeviceClientV1/PMGTCDC.config ]
[|P|I|: Open 'PMGTCDC.config' File Success! ]
[|P|I|: serres ]
[|P|I|: serres ]
[|P|I|: 192.168.2.200 ]
[|P|I|: 00000000000000000000000000000000 ]
[|P|I|: 00000000000000000000000000000000 ]
[|P|I|: 10 ]
[|P|I|: 10 ]
[|P|I|: Verification OK! ]
[|P|I|: DeviceName and Password Correct - Login OK! ]
[|P|I|: /dev/ttyACM0 ]
[|P|I|: 0123 [Ard2|Mikrofono]3.00)(Ard2|Fotias|28.00); ]
[|P|I|: 0234 [Ard2|Mikrofono]2.00)(Ard2|Fotias|27.00); ]
[|P|I|: 05.00]; ]
[|P|I|: #WRCNNewValues:2*(Ard2|Mikrofono|3.00)(Ard2|Fotias|28.00); ]
[|P|I|: #WRCNNewValues:2*(Ard2|Mikrofono|2.00)(Ard2|Fotias|27.00); ]
[|P|I|: #WRCNNewValues:2*(Ard2|Mikrofono|38.00)(Ard2|Fotias|28.00); ]
[|P|I|: #WRCNNewValues:2*(Ard2|Mikrofono|31.00)(Ard2|Fotias|28.00); ]
[|P|I|: Stopped!
[|A|E|: Version : PMG:V1 ]
[|A|E|: deviceNameConfirm : 22000 ]
[|A|E|: #WRCNNewValues:1*(Ard2|Mikrofono|6.01|10.20|300.00|0.00|NL){Fotias|0|0|12.00|255.00|0.00|NL}(Mousiki|1|1|1|0.00|255.00|0.00|128SW)
[|K|mousiki|1|1|0|0.00|255.00|0.00|0.55W)(Fotistiko|2|3|255.00|0.00|128SW); ]
[|P|I|: Clean new controller! ]
[|P|I|: #22000#NewController:6*(Ard2|Mikrofono|0|16.00|300.00|0.00|NL){Fotias|0|0|12.00|255.00|0.00|NL}(Mousiki|1|1|1|0.00|255.00|0.00|128SW); ]
[|P|I|: #22000#NewController:1*(Ard2|Mikrofono|2|1|255.00|255.00|0.00|128SW); ]
[|P|I|: Prepare Local Device success! ]
[|P|I|: #22000#InitControllers:3*(Ard2|S5); ]
[|P|I|: InitControllers OK! ]
[|P|I|: InitializationFinished OK! ]
[|A|E|: #22000#NewValues:2*(Ard2|Mikrofono|17.00)(Ard2|Fotias|27.00); ]
[|A|E|: #22000#NewValues:2*(Ard2|Mikrofono|13.00)(Ard2|Fotias|28.00); ]
[|A|E|: #22000#NewValues:2*(Ard2|Mikrofono|14.00)(Ard2|Fotias|27.00); ]
```

```
michael@PMGSL: ~/GitLab/myThesis/DeviceClientV1/dist
File Edit Tabs Help
michael@PMGSL: ~/GitLab/myThesis/DeviceClientV1/dist$ cd ../../DeviceClientV1/dist/
michael@PMGSL: ~/GitLab/myThesis/DeviceClientV1/dist$ sudo java -jar DeviceClientV1.jar
[sudo] password for michael:
[|P|I|: Michael Galliakakis 2016. ]
[|P|I|: TEI Athens - IT department ]
[|P|I|: Email : cs081001@teliauth.gr & michaelgalliaikakis@yahoo.gr . ]
[|P|I|: All files can be found in /home/michael/GitLab/myThesis ]
[|P|I|: "https://github.com/michaelgalliaikakis/mythesis.git" ]
[|P|I|: http://raspberrypi/DeviceClientV1/PMGTCDC.config ]
[|P|I|: Open 'PMGTCDC.config' File Success! ]
[|P|I|: Open 'PMGTCDC.config' File Success! ]
[|P|I|: andros ]
[|P|I|: 192.168.2.200 ]
[|P|I|: 50128 ]
[|P|I|: 10 ]
[|P|I|: false ]
[|P|I|: Verification OK! ]
[|P|I|: DeviceName and Password Correct - Login OK! ]
WARNING: RXTX Version mismatch
    Jar version = RXTX-2.zpre1
    Java Version = RXTX-2.zpre2
[|P|I|: /dev/ttyUSB0 ]
[|P|I|: #WRCNNewValues:6*(Ard1|thermometro|22.46)(Ard1|Sonar|11.00)(Ard1|thermApolygra|19.00)(Ard1|Ygresaia|48.00)(Ard1|fotostititis|74.00)(Ard1|ledFotoastisti
[|P|I|: 198.00); ]
[|P|I|: Clean new controller! ]
[|P|I|: #22000#NewController:9*(Ard1|thermometro|9|0|22.46|100.00|-40.00|NL)(Sonar|5|0|11.00|3000.00|0.00|NL)(thermApolygra|9|0|19.00|100.00|-40.00|NL)(Ygr
[|P|I|: esia|10|0|48.00|100.00|-40.00|NL)(fotostititis|3|0|74.00|255.00|0.00|NL)(ledFotoastititis|2|3|198.00|255.00|0.00|1TB)(Dikoaptis|11|2|0.00|2.00|0.00|15W
[|P|I|: Prepare Local Device success! ]
[|P|I|: #22000#InitControllers:1*(Ard1|B);
[|P|I|: InitControllers OK! ]
[|P|I|: InitializationFinished OK! ]
RXTX fhs_lock() Error: creating lock file: /var/lock/LCK..ttyUSB0: File exists
RXTX fhs_lock() Error: creating lock file: /var/lock/LCK..ttyUSB0: File exists
```



Η επόμενη φωτογραφία είναι από την εκτέλεση του Server :

```
[michael @ PMGS: -] - [17:49:46 Wed Apr 06]
[$] sudo java -jar ServerVi.jar
[sudo] password for michael:
[::] #####
[::] Thesis Michael Galliakis 2016.
[::] TEI Athens - IT department.
[::] Email : cs@81001@teiath.gr & michaelgallakiis@yahoo.gr .
[::] All files can be found :
[::] "https://github.com/michaelgallakiis/myThesis.git"
[::] #####
[::] ****
[::] Starting Server ...
[::] Failed : File "PMGTCSS.config" not Exists!
[::] Successfully check Database
[::] Successfully connected to the database.
[::] Opened successfully the server at the door 50128.
[::] Started Server.
[::] Listen ...
[::] Device Client | temporaryID : 0 , SysID : 22D InitControllers OK!
[::] Device Client | temporaryID : 0 , SysID : 22D NewController OK!
[::] Device Client | temporaryID : 0 , SysID : 22D put Initialization Finished!
[::] Device Client | temporaryID : 1 , SysID : 21D InitControllers OK!
[::] Device Client | temporaryID : 1 , SysID : 21D NewController OK!
[::] Device Client | temporaryID : 1 , SysID : 21D put Initialization Finished!
[::] UserDesktop Client | temporaryID : 3 , SysID : 17U01 | GetDevicesInfo put message :$$PutDevicesInfo:2*(21|andros|A|U)(22|serres|A|U);
[::] UserDesktop Client | temporaryID : 4 , SysID : 17U02 BringMeDevice OK!
[::] UserDesktop Client | temporaryID : 5 , SysID : 17U03 BringMeDevice OK!
[::] UserDesktop Client | temporaryID : 7 , SysID : 13U01 | GetDevicesInfo put message :$$PutDevicesInfo:6*(15|korinthos|R|D)(16|perama|A|D)(17|melidoni|A|D)(18|olxwrioi|F|D)(21|andros|A|U)(22|serres|A|U);
[::] UserDesktop Client | temporaryID : 8 , SysID : 13U02 BringMeDevice OK!
[::] UserDesktop Client | temporaryID : 9 , SysID : 13U03 BringMeDevice OK!
```

To **raspberry pi** πρέπει την πρώτη φορά που θα χρησιμοποιηθεί, να συνδεθεί με οθόνη, πληκτρολόγιο και ποντίκι ώστε να γίνει σύνδεση με τα κατάλληλα στοιχεία και στην συνέχεια να αποθηκευτεί μέσα από γραφικό περιβάλλον, το δίκτυο του LAN μέσω του οποίου θα έχει πρόσβαση στο internet.

Για κάθε άλλη φορά όμως που θα ανοίγει το raspberry pi γίνεται να αποκτηθεί πρόσβαση σε αυτό μέσω κάποιου προγράμματος που χρησιμοποιεί το ασφαλές δικτυακό πρωτόκολλο **SSH**.

Για να βρεθεί από κάποιον υπολογιστή στο ίδιο LAN τι ip έχει πάρει κάθε φορά το raspberry pi από το wifi access point (Αν χρησιμοποιείται ένας DHCP Server και όχι χειροκίνητα να έχει πάντα την ίδια ip) μπορεί να χρησιμοποιηθεί κάποιο πρόγραμμα όπως το **nmap**.

Επίσης, για μεταφορά αρχείων υπάρχει η δυνατότητα και του **sftp** προγράμματος μέσα από command line .

Ένα ενδεικτικό παράδειγμα χρήσης του nmap και ssh :

```
[michael @ PMGS: -] - [17:51:04 Wed Apr 06]
[$] nmap -sP 192.168.2.2
Starting Nmap 6.40 ( http://nmap.org ) at 2016-04-06 17:51 EEST
Nmap scan report for 192.168.2.1 (192.168.2.1)
Host is up (0.0019s latency).
Nmap scan report for android-1b08a9f7d64da318 (192.168.2.3)
Host is up (0.021s latency).
Nmap scan report for raspberrypi (192.168.2.7)
Host is up (0.057s latency).
Nmap scan report for pmgsl (192.168.2.8)
Host is up (0.023s latency).
Nmap scan report for pmgs (192.168.2.200)
Host is up (0.000088s latency).
Nmap done: 256 IP addresses (5 hosts up) scanned in 10.89 seconds

[michael @ PMGS: -] - [17:51:18 Wed Apr 06]
[$] ssh pi@192.168.2.7
pi@192.168.2.7's password:

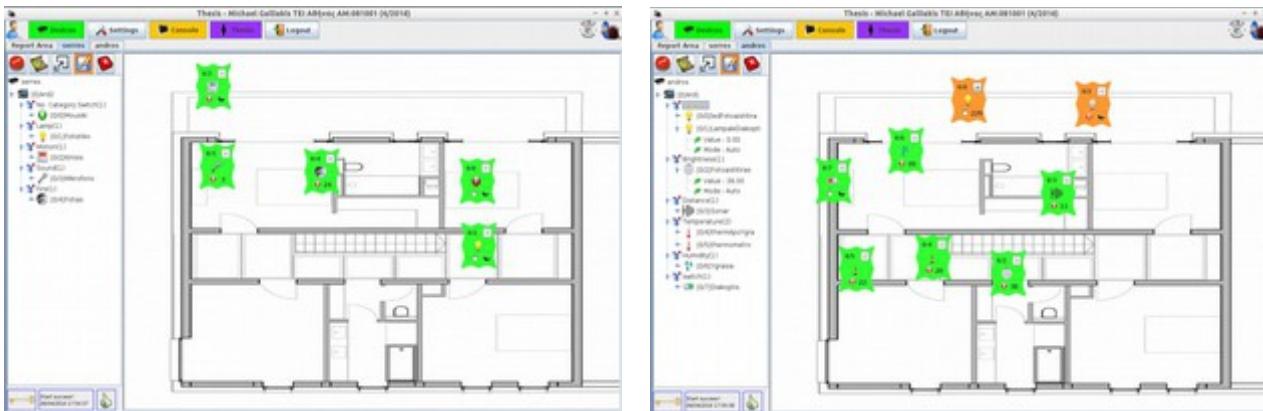
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Apr  3 15:38:22 2016
pi@raspberrypi ~ $ cd D
Desktop/          DeviceClientVi/ Documents/        Downloads/
pi@raspberrypi ~ $ cd DeviceClientVi/
pi@raspberrypi ~/DeviceClientVi $ sudo java -jar DeviceClientVi.jar
[ |P| : ##### ]
```

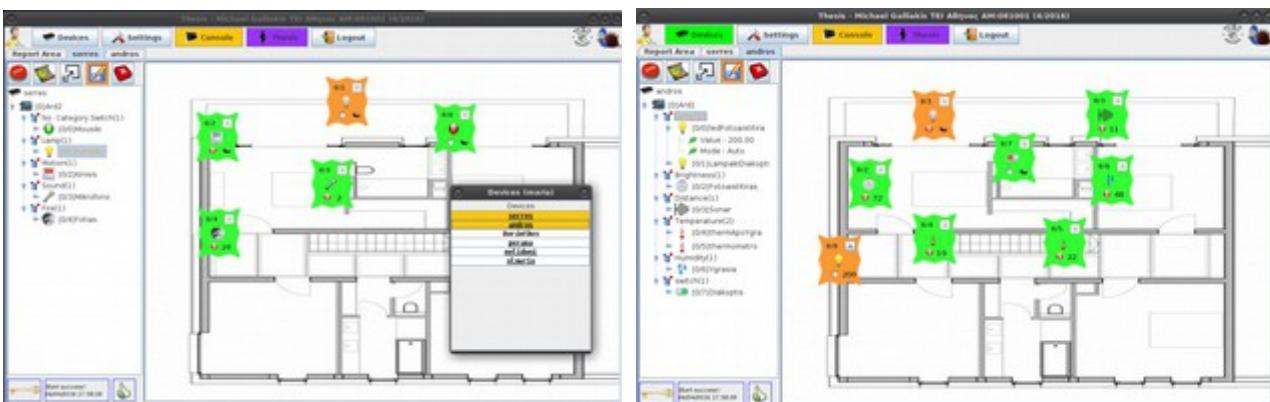


6. Μετά από όλα τα προηγούμενα βήματα, το μόνο που πρέπει πια να κάνουν οι χρήστες “newUser” και “maria” είναι να τρέξουν τον UserClient (ενώ βέβαια το έχουν ήδη κατεβάσει από την ιστοσελίδα) ώστε να πραγματοποιήσουν τις εποπτείες.

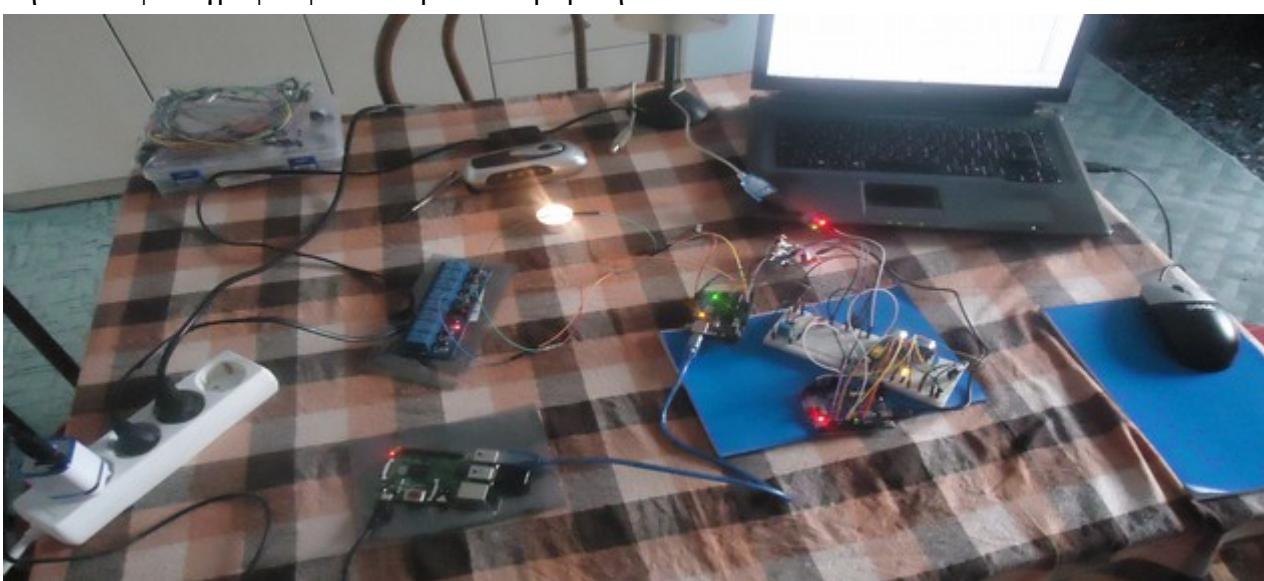
Στις 2 από κάτω φωτογραφίες φαίνονται οι δύο εποπτείες από τον χρήστη “newUser” :



Στις 2 κάτωθεν φωτογραφίες φαίνονται οι δύο εποπτείες από τον χρήστη “maria” :

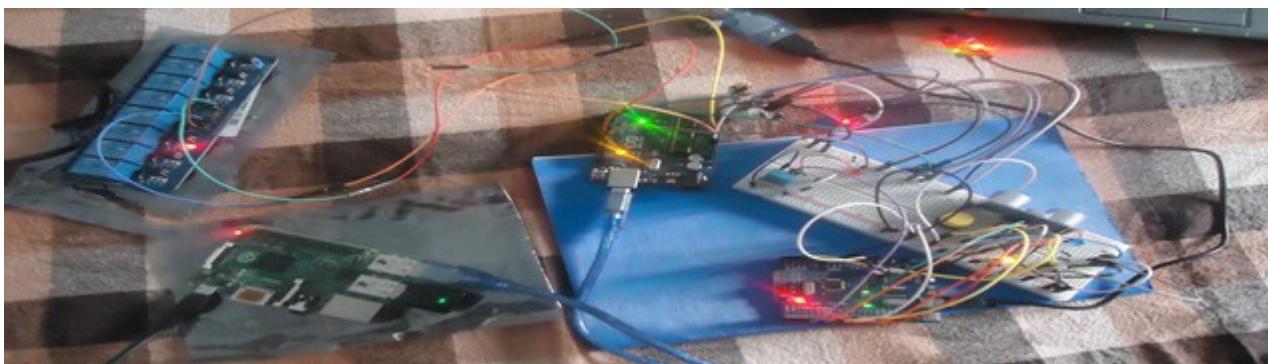


Στην κάτω φωτογραφία φαίνεται μια πανοραμική και των 2 συσκευών :

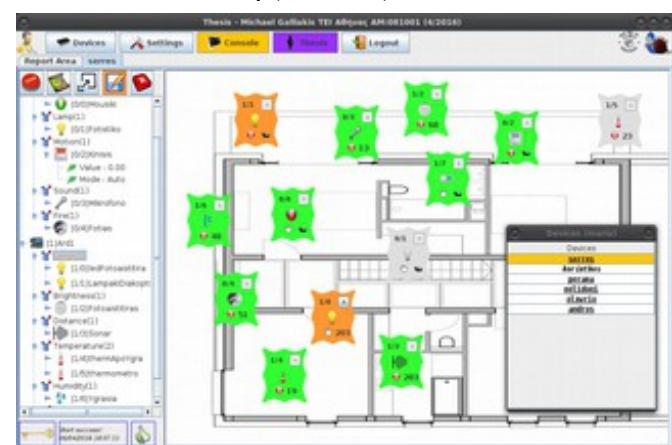
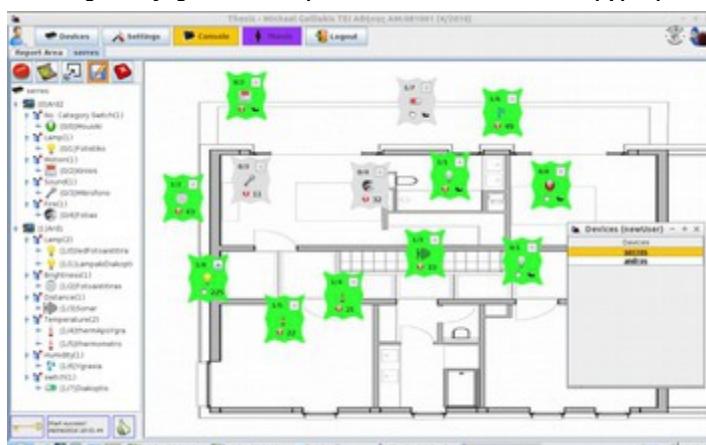




Στην κάτωθεν φωτογραφία φαίνεται μια πιο κοντινή εικόνα με τις 2 συσκευές :



Στις επόμενες φωτογραφίες φαίνεται η περίπτωση που και τα 2 arduino είναι συνδεδεμένα στο raspberry pi και επομένως ουσιαστικά υπάρχει μια μόνο online συσκευή (“serres”) :



```
File Edit View Search Terminal Help
[...]
[1]: #include "Arduino.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Servo.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"
[1]: #include "DHT.h"
[1]: #include "SoftwareSerial.h"
[1]: #include "Relay.h"]
[1]: [...]
```



- ✓ Θα μπορούσαν οι συσκευές “andros” & “serres” να βρίσκονται όντως σε κάποιους έξυπνους χώρους στην Άνδρο & στις Σέρρες και να γινόταν εποπτεία από οποιοδήποτε άλλο μέρος...



Πιθανές μελλοντικές επεκτάσεις

Λόγω της φύσης του συστήματος θα μπορούσαν να πραγματοποιηθούν πολλές επεκτάσεις .

Ωστόσο , θα περιγραφούν κάποιες μόνο από αυτές παρακάτω.

- ✓ Θα μπορούσαν στο μέλλον να δημιουργηθούν **userclients** σε μορφή app για να εκτελούνται μέσα από τα **κινητά τηλέφωνα** . Για παράδειγμα, θα ήταν σχετικά εύκολο να φτιαχτεί ένας UserClient σε **Android** διότι πρακτικά τα εκτελέσιμα του είναι σε Java. Οπότε λόγω του ότι υπάρχουν έτοιμες ρουτίνες για όλο το “background” ,το μόνο που θα έπρεπε να φτιαχτεί θα ήταν η νέα διεπαφή χρήστη (μόνο το νέο γραφικό περιβάλλον δηλαδή). Επίσης, όμως θα μπορούσαν να δημιουργηθούν εφαρμογές σε κινητά και για άλλα λειτουργικά συστήματα όπως για **IOS** ή για **Windows phone** . Αυτό είναι εφικτό διότι ο userclient επικοινωνεί με το server με τη βοήθεια socket και χρησιμοποιώντας ένα συγκεκριμένο “πρωτόκολλο” μηνυμάτων (με τη μορφή συμβολοσειρών) . Οπότε πρακτικά μπορεί να επικοινωνήσει μια οποιαδήποτε εφαρμογή (από όποια γλώσσα προγραμματισμού και αν έχει φτιαχτεί) με το server εφόσον χρησιμοποιεί socket και τηρεί το πρωτόκολλο μηνυμάτων .
- ✓ Ακόμη θα μπορούσε μελλοντικά να δημιουργηθεί μια εφαρμογή (που πιθανόν να “τρέχει” σε περιβάλλον γραμμής εντολών) που να συνδέεται με παρόμοιο τρόπο όπως οι άλλοι clients πάνω στο server και να δίνει τη δυνατότητα σε χρήστες (τύπου συστήματος) να έχουν “εικόνα” για το server ενώ αυτός εκτελείτε (**Manage client**). Για παράδειγμα, να μπορεί κάποιος απομακρυσμένος χρήστης συστήματος να δει ποιοι άλλοι χρήστες είναι συνδεδεμένοι πάνω στο server, τι ιρ έχουν κ.α . Ακόμη να μπορεί μέσω αυτής της εφαρμογής κάποιος χρήστης συστήματος να παίρνει στατιστικά , να βλέπει πόσα socket έχει πάνω του ο Server, να βλέπει την κίνηση ενός συγκεκριμένου “καναλιού” και πιθανόν να μπορεί να βλέπει και άλλα τέτοιου είδους πράγματα όπως επίσης και να παρατηρεί τυχών κακόβουλες ενέργειες .
- ✓ Μια άλλη μελλοντική δυνατότητα του συστήματος ,θα μπορούσε να ήταν να μπορούν να γίνονται **διάφορων ειδών καταγραφές** (τιμές μονάδων, συνδέσεις χρηστών κα) σε νέους πίνακες της βάσης δεδομένων . Αυτό βέβαια θα είχε σαν επακόλουθο να αναπτυχθεί παραπέρα και η ιστοσελίδα ώστε να μπορούν οι χρήστες μέσω των κατάλληλων νέων σελίδων να ανακτούν τις ανάλογες πληροφορίες που τυχών θα θέλουν .
- ✓ Επιπλέον , θα μπορούσε να μεγαλώσει η λίστα των τύπων μονάδων που υπάρχουν μέχρι τώρα ώστε να μπορεί ο **userClient** να υποστηρίζει περισσότερα εικονίδια μονάδων (πχ να υπάρχει εικονίδιο ανεμιστήρα , εικονίδιο βρύσης για αυτόματο πότισμα και άλλα πολλά).
- ✓ Μαζί με τα παραπάνω , θα μπορούσε (σε ένα υποθετικό σενάριο , στο οποίο θα υπήρχαν πάρα πολλοί ταυτόχρονοι χρήστες και συσκευές συνδεδεμένοι στο server) να δημιουργηθεί ένας διακομιστής (σαν **κεντρικός εξυπηρετητής**) για να μοιράζει το φόρτο σε άλλους **υπο-server** (που ουσιαστικά θα κάνουν τις λειτουργίες του τωρινού server) . Οι υπο-server αυτοί θα βρίσκονται σε άλλους υπολογιστές (με δικό τους επεξεργαστή,μνήμη κλπ) & θα μπορούν να εκτελεστούν σε πολλά μηχανήματα δυναμικά ανάλογα το φόρτο που θα υπάρχει (Θα πρέπει βέβαια να ρυθμιστούν κατάλληλα για να επικοινωνούν με το κεντρικό server) . Με αυτό το τρόπο , το σύστημα μας θα έχει πρακτικά κατανεμημένο server και θα μπορεί να υποστηρίξει πολλαπλάσιες συνδέσεις socket σε σχέση με το τωρινό σύστημα ...
- ✓ Επίσης, με βάση το πρωτόκολλο που έχει αναπτυχθεί και με την ίδια λογική θα μπορούσαν να προγραμματιστούν και **άλλοι τύποι ελεγκτών** , πέρα από arduino και έτσι να έχουμε μεγαλύτερη ταχύτητα ανταπόκρισης και περισσότερη αξιοπιστία υλικού.
- ✓ Ακόμη να αναφερθεί ότι θα μπορούσε να **εξελιχθεί** ιδιαίτερα **το κομμάτι του Arduino** ώστε πιθανόν να είναι πιο κοντά στην πραγματικότητα και με πιο αξιοποιήσιμες τελικές τιμές οι μετρήσεις από κάποιους αισθητήρες που αναφέρθηκαν .



Συμπεράσματα

Για την περάτωση της πτυχιακής αυτής εργασίας ήταν απαραίτητο να αξιοποιηθούν στην πράξη διάφορες γνώσεις από αυτές που αποκτήθηκαν μέσα από την διάρκεια φοίτησης στο τμήμα πληροφορικής του ΤΕΙ Αθήνας . Επίσης όμως, χρειάστηκε και μεγάλη έρευνα σε όλα τα επίπεδα όπως ακόμη και μελέτη διάφορων πηγών (κατά κύριο λόγο από βιβλία και το Διαδίκτυο) ώστε αφενός να αποκτηθούν νέες γνώσεις και αφετέρου να γίνει διεύρυνση των ήδη γνώσεων που υπήρχαν πάνω σε ορισμένα θέματα. Επιπλέον, μετά την πραγματοποίηση αυτού του εγχειρήματος αυξήθηκε κατά πολύ η εμπειρία που υπήρχε ,τόσο σε θεωρητικά όσο και σε πρακτικά μέρη , που αφορούν τις τεχνολογίες της πληροφορικής γενικότερα.

Πιο συγκεκριμένα , μέσα από αυτή τη πτυχιακή εργασία ,δόθηκε η ευκαιρία να γίνει η ανάλυση , η σχεδίαση και τέλος η υλοποίηση ενός συστήματος , παίρνοντας υπόψιν σαν απαιτήσεις ,τις πραγματικές ανάγκες χρήσης του και όχι απλά μια υποθετική περίπτωση . Έτσι το τελικό αποτέλεσμα πέρα από πείραμα ,μπορεί να χρησιμοποιηθεί έμπρακτα και σε πραγματικές συνθήκες. Ακόμη , μέσω της εργασίας αυτής αναζητήθηκαν και στη συνέχεια χρησιμοποιήθηκαν διάφορες γλώσσες προγραμματισμού και άλλα εργαλεία (όπως το git κ.α) ώστε να υπάρξει περαιτέρω γνώση , εμπειρία και μεγαλύτερη εξοικείωση πάνω σε όλα αυτά . Επίσης, η ανάπτυξη των projects έγινε όπως και σε μια πραγματική εταιρία ανάπτυξης λογισμικού , δηλαδή γράφτηκαν σχόλια στο κώδικα και υπάρχει κάποια σχετική τεκμηρίωση , ώστε να είναι πιο απλός-κατανοητός , πιο διαχειρίσιμος και πιο εύκολα μελλοντικά επεκτάσιμος ο κώδικας των εφαρμογών. Ο σκοπός ήταν να δημιουργηθεί μια βάση στο φοιτητή ώστε να μπορεί να αντεπεξέλθει στα καθήκοντα που μπορεί να έχει σε ένα πραγματικό επαγγελματικό περιβάλλον. Επιπλέον ,έγινε γνωστό και φάνηκε στη πράξη πως μπορούν συνδυαστικά διαφορετικές μεταξύ τους εφαρμογές και τεχνολογίες, να χρησιμοποιηθούν ώστε να υπάρξει ένα ενιαίο αποτέλεσμα και ουσιαστικά να υφίσταται ένα ολοκληρωμένο σύστημα .

Βέβαια ,θα μπορούσε να ειπωθεί ακόμη ότι μέσα από την πτυχιακή εργασία αυτή , φαίνεται στην πράξη πως μπορούν να αξιοποιηθούν οι υπολογιστές και τα δίκτυα ώστε να κάνουν ευκολότερη και καλύτερη την ζωή των ανθρώπων. Δηλαδή , φάνηκε πως μπορεί να αξιοποιηθεί έμπρακτα το internet of thinks (διαδίκτυο πραγμάτων). Που σημαίνει ότι με το “ολοκληρωμένο σύστημα που αφορά τη διαχείριση έξυπνων χώρων απομακρυσμένα σε πραγματικό χρόνο από πολλούς διαφορετικούς χρήστες” μπορεί να γίνει η διαχείριση διάφορων πραγμάτων μέσω του διαδικτύου . Για παράδειγμα ,εν δυνάμει γίνεται να ανοίγουν και να κλείνουν όλες οι ηλεκτρικές συσκευές (λάμπες,ανεμιστήρες,καφετειέρες κ.α) που υπάρχουν σε κάποιον χώρο από οποιαδήποτε άλλο σημείο μέσα στο κόσμο . Όπως επίσης γίνεται να μπορεί κάποιος χρήστης να γνωρίζει την “αίσθηση” κάποιου χώρου , με τη χρήση διάφορων ειδών αισθητήρων . Και μάλιστα όλα αυτά γίνονται σε πραγματικό χρόνο (δηλαδή “ζωντανά”) ακόμη και όταν πρόκειται κάποιος χρήστης να βρίσκεται σε κάποιο πολύ μακρινό (γεωγραφικό) μέρος από τον “έξυπνο” χώρο που εποπτεύει . Και αναφέρετε αυτό διότι όντως έγινε επιτυχημένη δοκιμή στο να ανοίξει και να κλείσει μια λάμπα που βρισκόταν στην Αθήνα , από δύο άτομα που το ένα ήταν στο Ρέθυμνο και το άλλο στην Κολωνία της Γερμανίας.

Για τέλος να αναφερθεί ότι μέσα από αυτήν την εργασία μεταξύ άλλων έγινε γνωστό πως πρέπει να φτιαχτεί μια ολοκληρωμένη εργασία από την αρχή μέχρι το τέλος της . Δηλαδή πέρα από το πρακτικό κομμάτι (της ανάπτυξης μιας εφαρμογής) δόθηκε και η ευκαιρία να μάθει ο φοιτητής της πτυχιακής αυτής πως να παρουσιάζει με τεκμηριωμένο τρόπο όλη την ερευνά που έχει γίνει .



Βιβλιογραφία

Γιακουμάκης, Μ. , & Διαμαντίδης, Ν (2009). Τεχνολογία Λογισμικού
Αθήνα: Εκδόσεις ΣΤΑΜΟΥΛΗ Α.Ε

Γκριτζάλη, Σ. - Κάτσικα, Σ. , & Γκριτζάλη, Δ.(2003). Ασφάλεια Δικτύων υπολογιστών (Τεχνολογίες και υπηρεσίες σε περιβάλλοντα ηλεκτρονικού επιχειρείν και ηλεκτρονικής διακυβέρνησης .
Αθήνα: Εκδόσεις Παπασωτηρίου.

Κοΐλιας, Χ. (2004). Δομές δεδομένων & οργανώσεις αρχείων (2^η έκδοση)
Αθήνα: Εκδόσεις Νέων Τεχνολογιών.

Σκουρλάς, Χ. (2000). Σχεσιακές βάσεις δεδομένων (1^η έκδοση).
Αθήνα: Εκδόσεις Νέων Τεχνολογιών.

Σκουρλάς, Χ. (2001). Υλοποίηση εφαρμογών με γλώσσα SQL (1^η έκδοση).
Αθήνα: Εκδόσεις Νέων Τεχνολογιών.

Τομαράς, Α. (1994) . C θεωρία και πράξη (1^η έκδοση).
Αθήνα: Εκδόσεις Νέων Τεχνολογιών.

Τομαράς, Α. (1999) . C++ Αντικειμενοστρεφής προγραμματισμός υπολογιστών (1^η έκδοση).
Αθήνα: Εκδόσεις Νέων Τεχνολογιών.

Cadenhead, R. , & Lemay, L. (2009). Πλήρες Εγχειρίδιο της JAVA 6 (2^η Ελληνική έκδοση).
Αθήνα: Εκδόσεις Μ. Γκιούρδας.

Comer, D. (2007). Δίκτυα και διαδίκτυα υπολογιστών και εφαρμογές τους στο Internet .
Αθήνα: Εκδόσεις Κλειδάριθμος.

Dennis, A. , & Haley Wixom, B. , & Tegarden, D. (2010). Ανάλυση & σχεδιασμός συστήματος με τη UML 2.0 (Μια αντικειμενοστρεφής προσέγγιση) (3^η Αμερικάνικη έκδοση) .
Αθήνα: Εκδόσεις Κλειδάριθμος.

Hall, M. , & Brown, L. (2006). Servlets και σελίδες διακομιστή JAVA (Τεχνολογίες πυρήνα)
(2η Αμερικάνικη έκδοση).
Αθήνα: Εκδόσεις Κλειδάριθμος.

Shneiderman, B. , & Plaisant, C. (2010). Σχεδίαση διεπαφής χρήστη (Στρατηγικές για αποτελεσματική επικοινωνία ανθρώπου – υπολογιστή) (5^η έκδοση)
Θεσσαλονίκη: Εκδόσεις ΤΖΙΟΛΑ.

Silberschatz, A.,& Baer Galvin, P.,& Gagne, G.(2009).Λειτουργικά Συστήματα (2^η Ελληνική έκδοση)
Αθήνα: Εκδόσεις ΙΩΝ



Sommerville, I. (2009) . Βασικές αρχές Τεχνολογίας Λογισμικού (8^η Αγγλική έκδοση)
Αθήνα: Εκδόσεις Κλειδάριθμος.

Stroustrup, B. (2009). Προγραμματισμός με τη C++ .
Αθήνα: Εκδόσεις Παπασωτηρίου.

Διαδικτυακή εγκυκλοπαίδεια ελεύθερου περιεχόμενου.
Retrieved from : <https://el.wikipedia.org/>

Arduino official page.
Retrieved from : <https://www.arduino.cc/>

Freestuff Forum (2007,Jan 8) - Μανώλης Μαρκατσέλας. CSS Στήσιμο ιστοσελίδας (2^ο παράδειγμα).
Retrieved from :<http://www.freestuff.gr/forums/viewtopic.php?t=26378>

Google μηχανή αναζήτησης .
Retrieved from : <https://www.google.gr/>

Stack Overflow community of programmers.
Retrieved from : <http://stackoverflow.com/>

W3schools web developer site.
Retrieved from : <http://www.w3schools.com/>



Παράτημα (οδηγίες για το “στήσιμο” του συστήματος)

Κάποιες χρήσιμες πληροφορίες για τι ακριβώς χρησιμοποιήθηκε :

- java version "1.8.0_66" [Java(TM) SE Runtime Environment (build 1.8.0_66-b17)]
- Netbeans IDE : 8.0.2
- Arduino IDE : 1.6.7
- PHP 5.6
- mysql Ver 15.1 Distrib 5.5.44-MariaDB, for Linux (x86_64)
- Server version: Apache/2.4.6 (CentOS)
- ✓ Για να παίξει ο Java Server πρέπει να υπάρχει στα Libraries του Project το **“mysql-connector-java-5.0.8-bin.jar”** για να μπορεί να επιτευχθεί η σύνδεση της βάσης δεδομένων με τη Java .
- ✓ Για να μπορέσει να διαβάζει ο DeviceClientV1 από τις σειριακές θύρες USB (Και κατά επέκταση από το Arduino) πρέπει να υπάρχουν 2 αρχεία (RXTXcomm.jar και librxtxSerial.so) στο φάκελο εγκατάστασης της Java (JDK) .
Σε περιβάλλον Linux πρέπει να μπει το **RXTXcomm.jar** μέσα στο ευρετήριο :
[JDK]/jre/lib/ext/ πχ όπως : jdk1.8.0_65/jre/lib/ext/**RXTXcomm.jar**
Και το **librxtxSerial.so** μέσα στο ευρετήριο :
[JDK]/jre/lib/[Αρχιτεκτονική Η/Υ σας]/ πχ όπως :
jdk1.8.0_65/jre/lib/amd64/**librxtxSerial.so**

Ανάλογα την αρχιτεκτονική (πχ 32bit ή 64bit) ή το λειτουργικό σύστημα πρέπει να χρησιμοποιηθεί το κατάλληλο αρχείο **librxtxSerial.so** .
Επίσης, στην περίπτωση διαφορετικού λειτουργικού συστήματος (όχι linux) θα χρειαστεί πιθανόν να ακολουθηθεί άλλος τρόπος εγκατάστασης.
Όλα τα αρχεία της RXTX 2.2 βρίσκονται στα παρακάνω link (μέσα στα zip αρχεία υπάρχουν README files για οδηγίες εγκατάστασης):
<http://rxtx.qbang.org/wiki/index.php/Download>
Binary : <http://rxtx.qbang.org/pub/rxtx/rxtx-2.2pre2-bins.zip>
Source : <http://rxtx.qbang.org/pub/rxtx/rxtx-2.2pre2.zip>

(Όλα τα παραπάνω αρχεία μπορείτε να τα βρείτε και στο Github)

Κάποιες πληροφορίες για μικρο “προβλήματα” και πως αντιμετωπίστηκαν:

- ◆ Αν ο υπολογιστής του server συνδέεται στο internet μέσω LAN (και ουσιαστικά δεν έχει πραγματική ip) , τότε πρέπει να γίνει port forward στο router ...
(Κάθε router έχει το δικό του τρόπο για να κάνει κάποιος port forward)
- ◆ Πρέπει να επιτραπεί στο firewall να έχει ανοικτή τη πόρτα του Server
Πχ στο Centos(Linux) πρέπει να εκτελεστεί η ακόλουθη εντολή :
sudo iptables -I INPUT -p tcp --dport 50128 --syn -j ACCEPT (Για άνοιγμα της 50128)
- ◆ Επίσης, στο Linux μπορεί να πρέπει να δοθούν τα κατάλληλα δικαιώματα σε κάποια πράγματα για να σας λειτουργήσει ο DeviceClientV1 όπως για παράδειγμα στις διάφορες συσκευές (πχ /dev/ttyACM0) των arduino ή στο ευρετήριο “/var/lock” .
Αν τρέχει το πρόγραμμα από το χρήστη root (ή με sudo) δεν υπάρχει κανένα πρόβλημα.