TARTU ÜLIKOOL
Arvutiteaduse instituut
Informaatika õppekava

Martin Johannes Liba

Arduino ilmateataja

Bakalaureusetöö (9 EAP)

Juhendajad: Alo Peets, Anne Villems, Taavi Duvin

Tartu 2017

Arduino ilmateataja

Lühikokkuvõte:

Käesoleva bakalaureusetöö eesmärgiks on luua Arduino ilmateataja, mis pärib ilmaandmed internetist ja kuvab need ekraanile. Ilmateataja ehitamist dokumenteeritakse viisil, et see oleks eestikeelseks õppematerjaliks vajutustundliku ekraani kasutamisest Arduinoga ja Arduinole juhtmevaba internetiühenduse loomisest. Lisaks on eesmärk võimaldada noortel lõputöö põhjal sarnane ilmateataja ehitada ja seeläbi tekitada noortes arusaama ning huvi programmeerimise ja arvutiteaduste vastu. Käesoleva töö esimeses peatükis tutvustatakse ehitatavat ilmateatajat. Teises peatükis kirjeldatakse projektiks vajalikke riistvara komponente ja tutvustatakse nende programmeerimisvõimalusi. Kolmandas peatükis kirjeldatakse, milliseid tarkvara ja riistvara ettevalmistusi on vaja enne riistvara kasutamist teha. Neljandas peatükis näidatakse põhilisi programmeerimisvõtteid ilmateataja programmeerimiseks, mis on ühtlasi õppematerjaliks ekraani kasutamiseks ja juhtmevaba internetiühenduse loomiseks Arduinoga.

Võtmesõnad:

Arduino, ESP8266, LCD ekraan, Programmeerimine

CERCS: P170(Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine)

Arduino weatherbox

Abstract:

The aim of this bachelor thesis is to construct Arduino weatherbox and record the process to create learning materials on how to use touchscreen display with Arduino and connect Arduino with internet wirelessly. Also students can recreate this weatherbox based on the thesis and get new knowledge and skills on programming and engineering. The first chapter introduces the device that is being created. The second chapter describes used hardware and software to build Arduino weatherbox. The third chapter describes what preparations need to be made before using the hardware. The fourth chapter explaines main programming techniques and functions to build the Arduino weatherbox.

Keywords:

Arduino, ESP8266, LCD display, Programming

CERCS: P170 (Computer science, numerical analysis, systems, control)

Sisukord

Sissejuhatus

Tänapäeval puudutatakse tihti meedias teemat, kuidas noored ei loe enam raamatuid, ei mängi pärast kooli õues sõpradega ja ei käi trennides vaid selle asemel pühendavad suurem osa ajast nutiseadmetes ja arvutis mängimise ning sotsiaalmeedias surfamise peale. Tehnoloogia arengus on mõned hakanud nägema ohtu, et järeltulev põlvkond väärtustab ainult virtuaalseid hüvesid ja ei oska enam päris maailmas hakkama saada.

Autori arvates tehnoloogia areng pole probleemiks vaid probleem on see, et paljud noored kui ka täiskasvanud on vaimustatud uutest nutividinatest, mis nende elu mugavamaks teevad, kuid nende ehitust ja põhilisi tööpõhimõtteid ei mõisteta. Seadmete tööpõhimõtete mõistmine on tähtis, kuna tehnoloogia on juba praegu lahutamatu osa meie maailmast ja tulevikus mängib see veel olulisemat rolli. Noori peaks rohkem õpetama mõistma, kuidas sellised vinged seadmed töötavad, et nad ehk ka ise kunagi sellise tehnika arendusele kaasa lööksid. Õpetustööga tuleks alustada koolinoortest ja muuta teekond tehnoloogia juurde nende jaoks lõbusaks ja huvitavaks.

Hea võimalus näidata riistvara omavahelist suhtlust ja õpetada programmeerimist on kasutada Arduino arendusplaate. Arduino arendusplaadid on sobivad koolides noortele programmeerimise õpetamiseks, kuna nende programmeerimine on algajatele jõukohane ja arendusplaatide kasutamise kohta leidub palju lisamaterjale internetis. Samuti on nende põhjal hea näidata, kuidas erinevaid riistvara komponente koos kasutada ja neid omavahel suhtlema panna. Hetkel pole piisavalt eestikeelseid õppematerjale, kuidas kasutada Arduinoga ühilduvat vajutustundliku ekraani andmete visualiseerimiseks. Seetõttu on selle lõputöö üks põhikomponent Arduinoga ühilduv vajutustundlik ekraan. Samuti on õppematerjalid puudulikud, kuidas enamasti ilma internetiühenduseta Arduino arendusplaatidele juurde anda juhtmevaba internetiühendus.

Lõputöö eesmärgiks on ehitada Arduino ilmateataja, mis saab andmeid ilmaportaalist ja kuvab neid ekraanile. Ilmateataja loomise protsessi on eesmärk põhjalikult dokumenteerida, et see oleks õppematerjaliks, kuidas kasutada Arduinoga ühilduvat vajutustundliku ekraani ja ühendada Arduino juhtmevabalt internetiga. Täiendavalt saavad huvilised lõputöö näitel

omaenda kätega endale sarnase ilmateataja ehitada ja selle käigus arendada teadmisi ja oskusi programmeerimisest ning riistvaraga töötamisest.

Käesoleva töö esimeses peatükis analüüsitakse, milline ehitatav ilmateataja olema hakkab. Teises peatükis antakse ülevaade kasutatavast riistvarast ja milliste vahenditega käib nende programmeerimine. Kolmandas peatükis näidatakse, kuidas süsteemi alamosad töötavad ning milliseid ettevalmistusi on vaja teha riistvara kasutamiseks. Neljandas peatükis kirjeldatakse olulisemaid programmeerimisvõtteid ekraani kasutamiseks ja internetiühenduse loomiseks, mida tehakse läbi ilmateataja jaoks vajalike programmeerimisvõtete selgitamise.

1. Arduino Ilmateataja

Käesolevas peatükis antakse ülevaade lõputöö raames loodavast Arduino ilmateatajast. Kirjeldatud on miks lõputöö eesmärkide täitmiseks just sellise seadme loomine on hea ja analüüsitakse loodavat ilmateatajat.

1.1 Ilm ja selle olulisus

Ilmaga seonduv on alati olnud eestlaste jaoks oluline, kuna elatakse niivõrd muutlikus kliimas. Ühe päeva jooksul võib kogeda ilmaolusid, mis on omased mitmele erinevale aastaajale. Seetõttu on eestlased harjunud regulaarselt ilmateadet jälgima, et arvestada muutlike oludega. Ilmastikuinfo saamiseks on üks levinuim viis internetist ilmaennustuse vaatamine. Ilmaportaalide sama päeva ilmateade on üsna täpne ja seda võib enamasti tõetruult võtta [1]. Eelnevast ajendatuna tekkis lõputöö eesmärkide täitmiseks idee luua odav ilmateataja, mis kuvab ilmaandmeid ekraanile. Kuna ilmaportaalide sama päeva ilmateade on täpne ei mõõdeta andmeid lokaalselt vaid päritakse internetist. See võimaldab hästi näidata, kuidas kasutada Arduinoga vajutustundliku ekraani ja luua internetiühendus.

1.2 Ilmateataja tutvustus

Ilmateataja eesmärk on olla kompaktne seade, mis kuvab ekraanile ilmaportaalist saadava ilmastikuinfo põhjal praegust ja järgnevate tundide ilmastikuandmeid. Kasutaja saab ilmateataja ekraanil kuvatavat infot kiiresti lugeda ja selle põhjal teha erinevaid otsuseid. Näiteks kas kodust lahkudes panna kotti vihmavari või kas hetkel on piisavalt soe, et randa minna. Seade uuendab oma andmeid pidevalt, et hoida kasutajat kursis kõige värskemate ilmaoludega. Lisafunktsionaalsusena kuvab võimalusel ilmateataja infot ka sisetemperatuuri ja õhuniiskuse kohta, mida mõõdetakse kohalikult. Ilmateataja koosneb võimalikult odavatest Arduinoga ühilduvatest riistvara komponentidest, et selle maksumus oleks jõukohane huvilistele, kes sooviksid sarnast seadet endalegi ehitada.

1.3 Ilmateataja analüüs

Mõistmaks paremini milline valmiv seade olema hakkab, analüüsitakse järgnevates alapeatükkides, milliseid andmeid seade kuvab ning kuidas kasutaja saab süsteemiga suhelda.

1.3.1 Funktsionaalsed nõuded

Järgnevalt kirjeldatakse, kuidas ilmateataja kasutaja saab süsteemiga suhelda ja mida süsteem tegema peab. Kirjeldatud on nõue ning seejärel nõude selgitus.

1. Süsteem kuvab infot seadme algkonfiguratsiooni õnnestumise kohta. Seadme ühendamiselt vooluvõrku kuvatakse infot, kas internetiga ühendamine õnnestus või mitte.
2. Süsteem kuvab ekraanile veateate, kui andmete päring ilmaportaalist ebaõnnestus. Seade uuendab kuvatavaid ilmastikuandmeid teatud ajaperioodi tagant, kui päring ebaõnnestub, kuvatakse ekraanile veateade.
3. Süsteem kuvab infot praeguste ilmaolude kohta. Kui seade on algkonfiguratsiooni lõpetanud ja internetis andmete päringu sooritanud, saab kasutaja vaadata informatsiooni praeguste ilmaolude kohta. Kuvatakse infot temperatuuri, tuulesuuna, tuule kiiruse, õhurõhu, õhuniiskuse ja ilmaseisundi kohta.
4. Süsteem kuvab infot ilma ja temperatuuri kohta pärast ühte, kolme ja viite tundi. Kuvatakse infot, milline on ilm ja temperatuur järgnevatel tundidel.
5. Süsteem kuvab infot sisetemperatuuri ja õhuniiskuse kohta.
6. Kasutaja saab liikuda kolme ekraanivaate vahel. Praegusi ilmaolusid ja ilmaennustust järgnevateks tundideks kuvatakse erinevatel vaadetel. Lisaks on kolmas ekraanivaade, kust saab infot sisetemperatuuri ja õhuniiskuse kohta. Liikumine käib vajutustundliku ekraani abil.
7. Süsteem uuendab ilmaandmeid iga 10 minuti järel. Pidev uuendamine on oluline, et hoida kasutajat kursis värskete ilmaandmetega.
8. Süsteemil on päevarežiim ja öörežiim. Süsteem kuvab ikoone erinevalt kui on päev või öö. Päeval kasutatakse päikeseikooni ja öösel kuuikooni.
9. Ekraan läheb puhkerežiimi, kui kasutaja ja ekraani vahel pole interaktsiooni toimunud ühe minuti jooksul. Kui kasutaja pole vajutanud nupule, mis vahetab ekraanivaateid, lülitatakse ekraan energia säästmiseks välja.

10. Süsteem lõpetab puhkerežiimis oleku, kui seadmele ollakse piisavalt lähedal. Puhkeržiim lõpetatakse, kui kasutaja viipab käega seadme ees või liigub sellele piisavalt lähedale.

1.3.2 Mittefunktsionaalsed nõuded

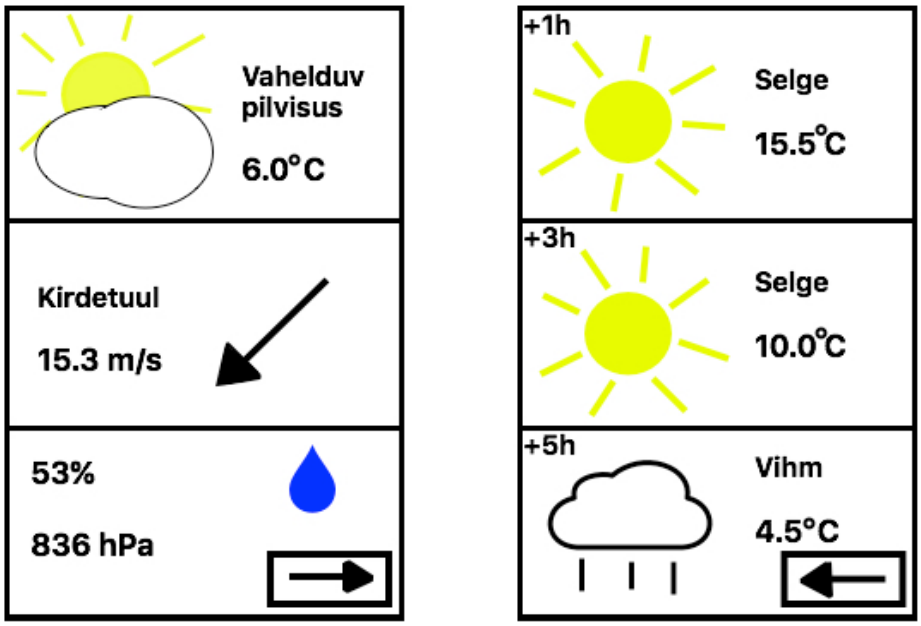
Järgnevalt kirjeldatakse, millised on kriteeriumid ilmateataja tööks. Kirjeldatud on nõue ning seejärel nõude selgitus.

1. Ekraanivaadete vahel saab kasutaja liikuda maksimaalselt kahe vajutusega ekraanil. Ilmaandete vaatamine peab olema kasutajale võimalikult vähe aeganõudev, seetõttu peab ekraanivaadete vahel olema võimalik kiiresti liikuda.
2. Seade peab töötama vaikselt. Kuna seade töötab pikema ajaperioodi vältel, ei tohi see kasutajat häirima hakata ja seetõttu on oluline seadme vaikne töö.
3. Seadme konfiguratsioon ei tohi kesta pikalt. Kui seade on vooluvõrku ühendatud siis pääsupunktiga ühendamine ei tohiks kesta kauem kui 20 sekundit, et kasutaja saaks võimalikult ruttu hakata ilmateatajat kasutama.
4. Andmete uuendamine toimub 5 sekundi jooksul. Ilmaandmete päringu tegemine ja andmete töötlemine ei tohi kesta kauem kui 5 sekundit.

Järgnevalt näidatakse, milline näeb välja kasutajaliides, mis järgib analüüsitud funktsionaalseid ja mittefunktsionaalseid nõudeid.

1.3.3 Kasutajaliides

Kasutajaliides on ilmateataja puhul väga oluline osa. Kasutajal peab olema mugav andmeid lugeda, ehk teksti suurus peab olema valitud piisavalt suur. Samuti on mõistlik andmete visualiseerimiseks kasutada lisaks tekstile ka ikoonikesi, mis aitavad informatsiooni kiiremini haarata. Ekraanivaadete vahel liikumise viis peaks olema lihtsasti mõistetav. Alljärgnevalt on toodud visandid ilmateataja kasutajaliidese põhivaadetest ehk ilmaandmeid kuvavatest vaadetest.



Joonis 1. Ekraani ilmavaadete visandid.

Sellega on loodava seadme nõuded ja kasutajaliidese väljanägemine on analüüsitud. Järgnevas alapeatükis kirjeldatakse, milliseid sarnaseid lahendusi on varem tehtud.

1.4 Ülevaade sarnastest lahendustest

Internetis leidub palju ilmaportaale ja on mitmeid erinevaid mobiilirakendusi kust on võimalik ilmateadet vaadata. Käesolevas töö tehakse hariduslikel eesmärkidel, mille käigus Arduino mikrokontrollerplatvormi kasutades luuakse ilmateataja. Arduino võimalused on piiratud ja seetõttu ilma näitavate mobiilirakenduste ja internetiportaalidega funktsionaalsuse mõttes ei konkureerita. Järgnevalt kirjeldatakse, milliseid Arduinol või muul sarnasel platvormil põhinevaid lahendusi on varem tehtud.

Nick Koumaris on loonud lahenduse nimega Art Deco Weather Forecast Dispaly. Projektis kuvatakse openweathermap.org ilmaportaalist saadavad ilmaandmed ekraanile. Mikrokontrollerina on kasutusel Wemos D1 Mini, mis on ESP8266 WiFi kiipi kasutav internetiga ühenduv arendusplaat. Ekraanina on kasutusel 1,8 tolline värviline ekraan, mis ei ole vajutustundlik. Ekraanile kuvatakse infot temperatuuri ja praeguse ilmaseisundi kohta. Antud projektis on kasutusele võetud võrreldes lõputöös loodava seadmega erinevat

mikrokontrollerplatvormi ning kasutusel pole vajutustundliku ekraani. Samuti visualiseeritakse erinevaid ilmaandmeid minimaalselt. Kogu Nick Koumarise loodud Art Deco Weather Forecast Display projekt on kättesaadav aadressil http://educ8s.tv/art-deco-weather-forecast-display/.

Jayraj Desai on teinud projekti Personal Weather Station. Projektis kogutakse erinevalt keskkonnatingimusi mõõtvaid andureid ja postitatakse tulemus Thingspeak lehele, kust saab väärtusi lugeda arvutist või telefonist. Mikrokontrollerplatvormina kasutatakse Arduino Nanot ja internetiga ühenduse loomiseks ESP8266 WiFi moodulit. Antud projektis kogutakse võrreldes lõputööga andmeid kohalikult, mitte ei võeta internetist ja neid ei kuvata ekraanile. Samuti ei saa andurite põhjal luua ilmaennustust vaid lugeda ainult hetke ilmaandmeid. Kogu Jayraj Desai tehtud ilmajaama projektiga saab tutvuda aadressil https://create.arduino.cc/projecthub/Blue_jack/personal-weather-station-arduino-esp8266-thingspeak-8d5cba.

Antud peatükis tutvustati bakalaureusetöö eesmärkide täitmiseks loodavat projekti. Analüüsiti loodava ilmateataja tööpõhimõtteid ja tutvustati kasutajaliidest. Järgmises peatükis kirjeldatakse riistvara ja tarkvara, mis on projektiks vajalikud.

2. Riistvara ja tarkvara

Käesoleva peatüki eesmärk on anda ülevaade antud töös kasutatavast riistvarast ja nende programmeerimisvõimalustest. Lisaks tutvustatakse, kuidas hakatakse internetis ilmaandmeid pärima.

2.1 Arduino

Arduino on mikrokontrolleriga arendusplaat, millega saab ühendada erinevaid andureid ja mooduleid ning nendelt saadavat sisendit töödelda ja vajadusel muuta väljundiks [2]. Arduino on populaarseks saanud tänu oma lihtsale ülesehitusele ja programmeerimisvõimalusele.

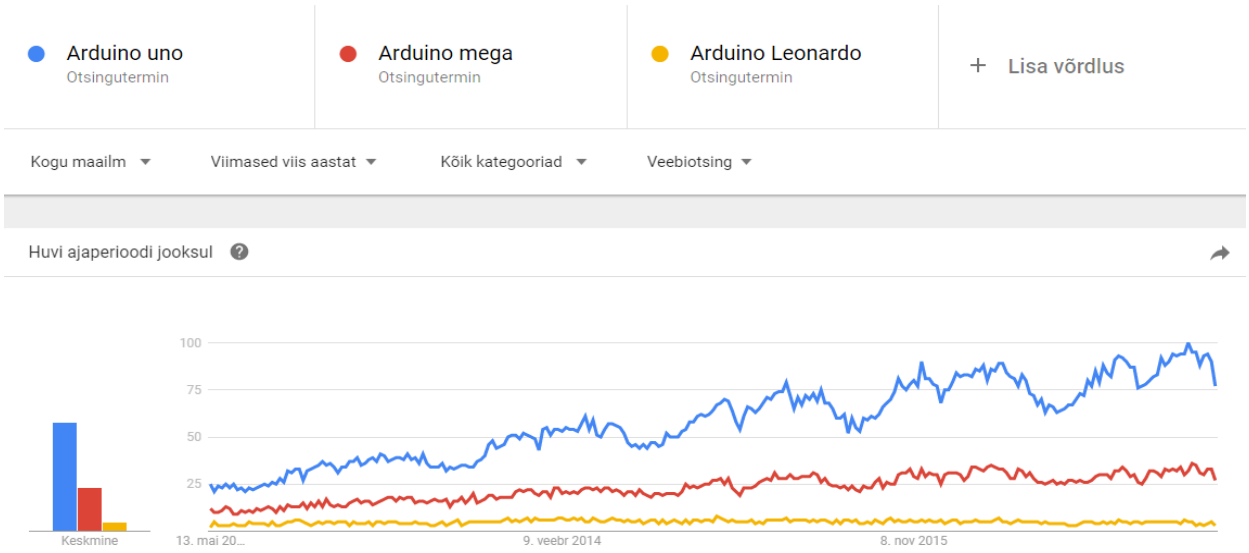
Arduinol on mitmeid erineva suuruse ja võimekusega arendusplaate, mis on sobilikud erineva raskusastme ja keerukusega projektide jaoks [2]. Arduino riistvara ja lähtekood on avatud, mis annab võimaluse ise ehitada Arduino arendusplaate vajaminevatest komponentidest ja soovi korral luua sellest oma versioone [3]. Algajal on lihtsam tellida juba valmis kontroller ja selle tundmaõppimisel konstrueerida erinevaid variante Arduinost.

Arduinot on ka võimalik ühendada arvutiga, teiste Arduinodega, anduritega ja muude erinevate elektrooniliste seadmetega ning nad omavahel suhtlema panna [3]. Selleks, et Arduino oskaks seadmetega suhelda tuleb arendusplaadile laadida programm, mis kirjeldab, kuidas suhtlus toimima peab. Programmerimiseks on kõige lihtsam kasutada arenduskeskkonda Arduino IDE (*IDE – Integrated Development Envoirement)*. Arduino arenduskeskkonna programmeerimiskeelt põhineb Wiring raamistikul [2] ja programmeerimiskeeltel C ning C++ [4].

2.2 Arduino Uno

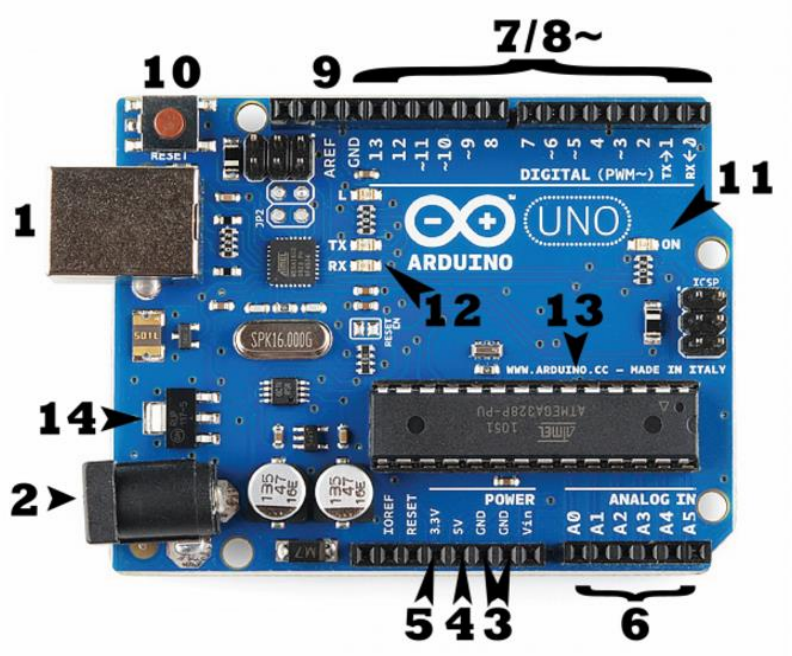
Arduino Uno on arendusplaat, mida paljud inimesed mõistavadki nimetuse Arduino all, kuna see on enimkasutatuim arendusplaat Arduinode seerias [5]. Uno populaarsust illustreerib ka joonis 2, kus on näha, et huvi arendusplaadi vastu ainult kasvab. Ka

käesolevas projektis kasutatakse Arduino Unot, mis lisaks populaarsusele on soodsaim Arduino arendusplaatide seas ning selle kohta on internetis saadaval kõige rohkem dokumentatsiooni ja näidisprojekte [5].



Joonis 2. Huvi Arduino Uno vastu viimasel 5 aastal Google statistika põhjal [6]

Arduino Uno arendusplaat (vt joonis 3) koosneb paljudest erinevatest komponentidest. Kõigepealt tuleb Arduino arendusplaadi puhul eristada, mida mõistetakse viigu ja mida pesa all. Viik on mikrokontrollerist väljaulatuv metallkontakt, mis on ühendatud arendusplaadi pesaga. Pesa kaudu saab arendusplaadiga ühendada andureid, mooduleid ja muud. Arduino viikude mõistet kasutatakse programmeerimisel, kus mikrokontrolleri viike seadistatakse erinevaid funktsioone täitma. Pesa mõistet kasutatakse, kui kirjeldatakse mingi teise seadme ühendamist Arduinoga. Arduino kasutamiseks on olulisim mõista arendusplaadil olevate mikrokontrolleri viikudega funktsionaalsusi., mis on ühendatud pesadega. Lisaks on plaadil veel erinevad tuled ja toitepesad, nupp plaadi lähtestamiseks ja muud osad.



Joonis 3. Arduino Uno arendusplaat [7]

Järgnevas loetelus on selgitatud Arduino Uno põhilised komponendid. Loetelu järjekorranumber vastab joonisel 3 olevale Arduino arendusplaadi osa tähistavale numbrile. Loetelu põhineb Arduino tutvustuses SparkFun internetilehel [7].

1. USB pistik (USB - *Universal Serial Bus)* port. USB pistiku abil saab Arduino arendusplaadile voolu anda ja programmikoodi laadida. Arduino kasutab USB B tüüpi pistikut.
2. Voolupesa. Voolupessa saab ühendada voolujuhtme, millega Arduinole voolu anda.
3. GND (GND – *ground*, eesti k maandus) pesad. Nende pesade kaudu käib vooluringi maandamine.
4. 5V (V - volt) pesa. 5V pesa kaudu saab arendusplaadi külge ühendatavaid lisakomponente varustada 5 voldise vooluga.
5. 3,3V pesa. 3,3V pesa kaudu saab arendusplaadi külge ühendatavaid lisakomponente varustada 3.3 voldise vooluga.
6. Analoogpesad (A0-A5) loevad signaali analoogsensoritest ja teisaldavad selle digitaalkujule, mida programm saab lugeda ja pärast signaali töötlemist digitaalpesa kaudu väljastada. Analoogpesa on võimalik vajadusel kasutada ka signaali väljastamiseks.
7. Digitaalsed pesad (D0-D13). Nende abil saab lugeda ja väljastada digitaalsignaali.

8. Pulsilaiusmodulatsiooni (*Pulse-Width Modulation* (PWM)) pesade abil saab simuleerida analoogväljundit.
9. AREF pesa abil saab vajadusel lisapesana lugeda kindla tugevusega analoogsisendit.
10. Lähtestamise nupp (*Reset button*) - Lähtestamise nupu vajutamisel taaskäivitatakse Arduino arendusplaadile laetud lähtekood. Kui programmi töö on katkenud saab nupu abil programmi taaskäivitada. Samuti on nupp mugav viis testimiseks, kui on tarvis koodi mitu korda jooksutada saab seda teha nupule vajutades.
11. Toite LED (LED - *Light-emitting Diode*) tuli. Toite LED tuli läheb põlema, kui Arduino arendusplaat on ühendatud vooluvõrguga. Kui pärast vooluga ühendamist tuli ei lähe põlema, on oht, et arendusplaadiga on midagi valesti.
12. TX (TX - *Transmit*) RX (RX - *Receive)* LED tuled - Nende LED tulede abil on aru saada, kui arendusplaat võtab andmeid vastu või saadab neid välja. Kui andmeid saadetakse, põleb TX LED tuli, kui andmeid vastu võetakse, põleb RX LED tuli.
13. ATmega328P mikrokontroller, juhib Arduino Uno arendusplaadi tööd.
14. Pingeregulaator. Pingeregulaator reguleerib arendusplaadi vooluringi lastavat pinget. Kui pinge on liiga suur, muudab pingeregulaator selle plaadile sobivaks.

Arduino Uno arendusplaadi toite pinge peab olema vahemikus 6-20V, soovituslik vahemik on 7-12V [5]. Sisend-väljundpesade alalisvoolu tugevus on 20mA (milliamprit) ja 3,3V pingena toitepesa alalisvoolu tugevus on 50 mA [5]. Uno muutmälu suurus on 2 kilobaiti ja välkmälu suurus 32 kilobaiti [5]. Välkmälu ehk programmimälu kasutatakse programmikoodi hoiustamiseks ja muutmälu muutujate ja muu info ajutiseks hoiustamiseks.

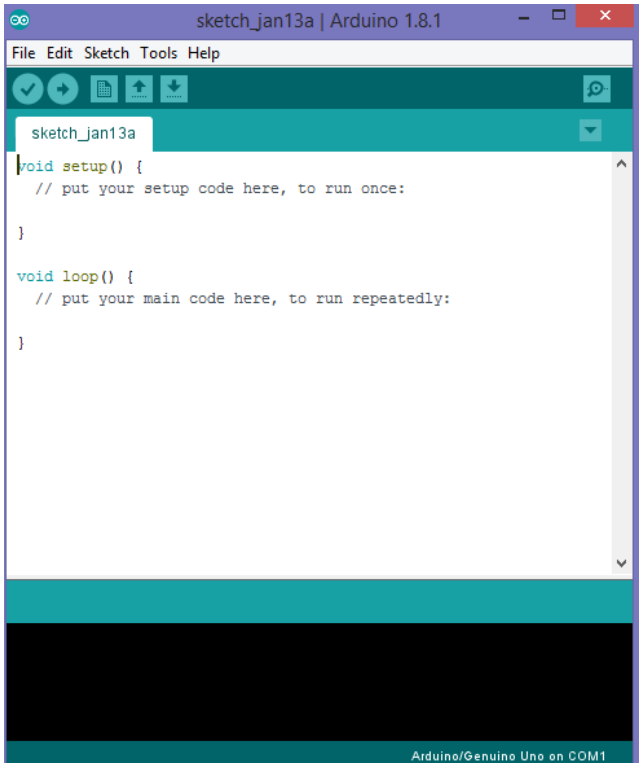
Edaspidi nimetatakse Arduino Uno arendusplaati käesolevas töös lihtsalt Arduinoks, kuna nagu eelpool mainitud siis enamasti mõistetaksegi Arduino all just Arduino Uno arendusplaati.

2.3 Arduino programmeerimine

Arduino programmeerimiseks on erinevaid programmeerimiskeskkondi ja -keeli. Ametlik Arduino kodulehelt alla laetav avatud lähtekoodiga programmeerimiskeskkond, on tarkvaraarenduskeskkond Arduino IDE [8], mida edaspidi nimetatakse Arduino arenduskeskkonnaks.

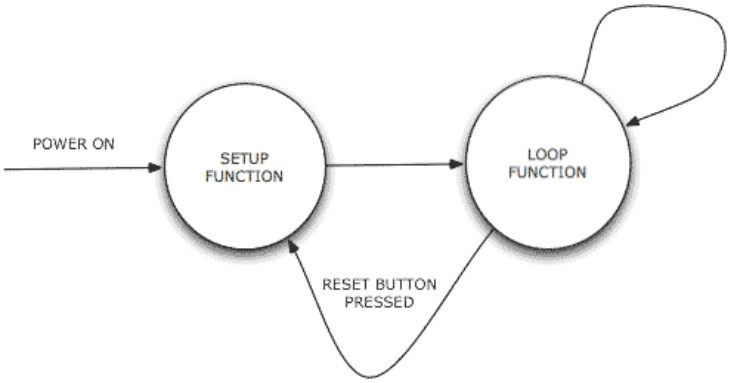
Peale Arduino arenduskeskkonna leidub mitmeid alternatiive, mille hulgas on visuaalse programmerimise ja koodi kirjutamise keskkondi. Arduino visuaalseks programmeerimiseks on näiteks Embrio [9], mis on mõeldud eelkõige noortele, kes tutvuvad programmeerimisega. Keerulisemate projektide puhul jääb visuaalsest programmeerimisest väheks. Koodikirjutamise alternatiivid Arduino arenduskeskkonnale on näiteks Programino IDE [9]. Alternatiivsete arenduskeskkondade kasutamine võib vajalikuks osutuda, kui projekt nõuab spetsiifilisemat lähenemist, mida on mugavam saavutada kasutades teist arenduskeskkonda Arduino asemel.

Käesolevas projektis kasutatakse programmeerimiseks samuti Arduino arenduskeskkonda (vt joonis 4.). Arenduskeskkonda saab kasutada nii Windows, Mac OS ja Linux operatsioonisüsteemidega arvutitel ja valida on veebipõhise ning töölaua versiooni vahel [8]. Veebiversiooni kasutamiseks on vajalik internetiühendus, mis alati ei pruugi saadaval olla. Samas uuendatakse veebiversiooni automaatselt uue versiooni või teekide tulekul [8], töölaua keskkonda kasutades tuleb seda ise teha. Arenduskeskkond on kirjutatud Javas ja kasutab lisaks teisi avatud lähtekoodiga tarkvara nagu näiteks Processing [8].



Joonis 4. Arduino arenduskeskkond.

Programmeerimiskeeleks Arduino arenduskeskkonnad on variatsioon keeltest C ja C++ [4]. Keel põhineb *Wiring* raamistikul [2], mis on loodud mikrokontroller arendusplaatide jaoks [10]. Koodi põhistruktuur (vt joonis 5.) jaguneb kaheks funktsiooniks *setup()* ja *loop()*.



Joonis 5. Arduino programmi põhistruktuur [11].

Funktsiooni *setup()* jooksutatakse üks kord programmi käivitamisel ja selles defineeritakse edaspidi vajaminevad muutujad ja kasutatavad arendusplaadi viigud [12]. Funktsioon *loop()* hakkab tööle pärast *setup()* funktsiooni ja seal asub põhiline osa koodist, mis kontrollib Arduino arendusplaadi tööd [13]. Erinevalt *setup()* funktsioonist jooksutatakse *loop()* funktsiooni tsükliliselt, kuni Arduino arendusplaadi vooluallikas on eemaldatud või vajutatakse lähtestamise nuppu plaadil [13]. Vajutades lähtestamise nuppu (vt joonis 3. *reset button*), hakkab programmi töö algusest pihta *setup()* funktsiooniga.

Arduino programmeerimiseks kasutatakse palju teeke. Teegid on funktsioonide kogumikud, mis teevad Arduino andurite, moodulite ja ekraanidega suhtlemise lihtsamaks. Näiteks, ekraani teek sisaldab funktsioone, mida saab kasutada ekraanile teksti, ringikeste ja paljude teiste kujundite kuvamiseks. Anduri teek sisaldab funktsioone, mille abil andurilt väärtusi lugeda. Palju erinevaid teeke on juba kaasas Arduino arenduskeskkonnaga, lisaks saab neid ise juurde panna, otsides internetist sobiva teegi.

Eelnevaga on Arduinoga seotud riistvara ja tarkvara läbi vaadatud. Järgnevates peatükkides kirjeldatakse, milliseid teisi riistvara komponente on ilmateataja jaoks tarvis, mida kasutatakse kood Arduinoga.

2.4 Vajutustundlik Ekraan

Arduino külge saab ühendada TFT (*Thin Film Transistor*) LCD (*Liquid Crystal Display*) ekraanimooduli, millele saab kuvada teksti, pilte ja erinevaid kujutisi. Ekraane pakuvad mitmed tootjad ja valikus on eri suurustes variante ning osad neist on vajutustundlikud. Kujutisi saab ekraanile kuvada kasutades teeke, mis sisaldavad ekraani töö kontrollimiseks vajaminevaid funktsioone. Näiteks funktsiooni teksti kirjutamiseks, pildi kuvamiseks või ristküliku joonistamiseks. Kõiki ekraane ei saa samade teekide abil kasutada. Sobivad teegid tuleb leida internetist vastavalt ekraani suurusele, tüübile ja margile.

Paljude ekraanide küljes on sisseehitatud microSD (SD-*secure digital*) mälukaardipesa. Mälukaardipessa saab sisestada microSD mälukaardi ja selle pealt kuvada ekraanile 24-bitises BMP (BMP - *Bitmap*) formaadis pilte. Lisaks saab mälukaarti kasutada andmete salvestamiseks, mis on kasulik, kuna Arduinol on toitevooluta olekus andmete salvestamiseks ruumi minimaalselt.

Käesolevas projektiks kasutatakse ilmaandmete visualiseerimiseks 2,8 tollist vajutustundliku ekraani (vt joonis 6), mis ühendatakse Arduino arendusplaadi külge. Antud ekraan on piisavalt suur, et sellelt teksti lugeda vähemalt poole meetri kauguselt. Ekraan on vajutustundlik, mis on oluline, sest seeläbi saab infot jaotada mitme ekraanivaate vahel, liikudes vaadete vahel ekraanil olevale nupule vajutades. Väiksem ekraan raskendaks andmete lugemist ekraanilt ja suurem viiks projekti kulutused kõrgemaks. Lisaks, ekraanid, mis on oluliselt suuremad kui 2,8 tolli, on Arduino Uno jaoks tehniliselt liiga nõudlikud. Arendusplaadi kiirus ja pesade arv ei jõuaks suuremat ekraani enam mõistliku kiirusega juhtida.

Joonis 6. TFT LCD ekraan, 2,8 tolli [14]

Järgnev loetelu on kasutatava 2,8 tollise TFT LCD ekraani spetsifikatsioon, mis põhineb Cyan Infitine internetilehel olevast ekraani tutvustusest [15]:

1) 2,8 tolline TFT LCD ekraan,
2) resolutsioon 240x320,
3) graafika draiver: IL9341,
4) microSD kaardi pesa,
5) takistuskihiga vajutustundlik ekraan,
6) 8-bitine digitaalne jadaliides,
7) 4 valget LED taustavalgustit.

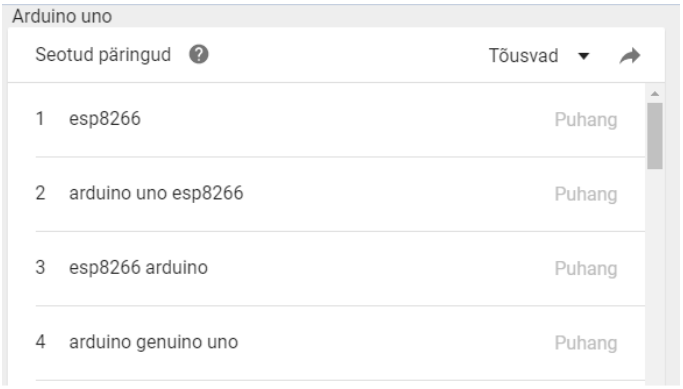
Ekraani nimetatakse vajutustundlikuks mitte puutetundlikuks, kuna nende tööpõhimõte on erinev. Puutetundlikku ekraani katab ühtlane elektriväli. Kui elektrit juhtiv keha puudutab ekraani, muutub selle tagajärjel ekraani elektriväli, mille põhjal tuvastatakse puutepunkt. Vajutustundliku ekraan koosneb kahest kindla elektritakistusega plastkileplaadist, mis tavaolekus kokku ei puutu. Kui ekraanile vajutada, puutuvad kaks takistuskihti omavahel kokku. Kokkupuutepunktis tekib elektriline kontakt, mille põhjal tehakse kindlaks puutepunkt [16].

Ekraani on mugav algajatel kasutada, kuna selle ühendamine Arduinoga ei nõua juhtmete jootmist plaadi külge. Piisab ekraaniviikude arendusplaadi pesadesse ühendamisest ning seejärel vajalike teekide internetist allalaadimisest, et ühendus ekraaniga luua. Vaadeldaval ekraanil on ka microSD mälukaardipesa, kuid antud projekti jaoks seda ei kasutata, kuna selleks vajalik funktsioon nõuab palju välkmälu ruumi. Lisaks hõivaks mälukaardi

funktsionaalsuse kasutamine neli Arduino digitaalpesa, mida on vaja teiste andurite ja moodulite ühendamiseks.

2.5 WiFi moodul

Arduino Uno arendusplaadile on vaja lisada internetitugi, et sooritada ilmaandemete päring internetist. Aastal 2014 tuli firma Espressif Systems välja WiFi kiibiga ESP8266. Pärast seda on huvi sellel kiibil põhinevate WiFi moodulite Arduinoga kasutamise vastu järjest tõusnud (vt joonis 7).

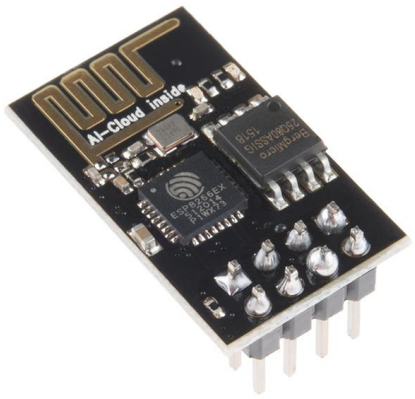


Joonis 7. Google statistika kõige rohkem kasvavatest Arduino Unoga seotud päringutest pärast ESP8266 väljatulekut aastal 2014 [17].

Käesolevas projektis kasutatakse Arduino Unole internetiühenduse võimaldamiseks ESP8266 kiibil põhinevad moodulplaati ESP8266 ESP-01 (vt joonis 8), mida edaspidi nimetatakse WiFi mooduliks. See on odavaim ESP8266 kiibil põhinev moodul, mida toodab kolmanda osapoole tootja AI-Thinker. Antud moodulit saab kasutada ühenduse loomiseks pääsupunktiga ja seeläbi tagada mikrokontrollerile juurdepääs internetile [18]. Peale selle saab ka moodulit ennast kasutada pääsupunktina [18]. Lisaks on võimalik mõlemat eelmainitud režiimi korraga kasutada. WiFi moodul kasutab andmete saatmiseks ja vastuvõtmiseks TCP/IP protokolli [18].

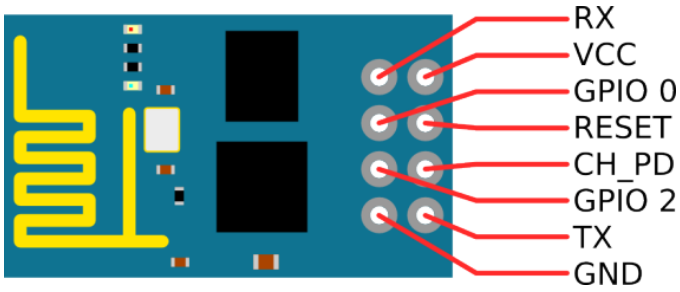
Moodulit on peale internetiühenduse loomise võimalik kasutada mikrokontrollerina, kuna sisaldab selleks vajaminevaid osi nagu välkmälu, muutmälu ja protsessor [18]. Ilmateataja tegemiseks on siiski vaja Arduino mikrokontrollerplatvormi, kuna selleks projektiks jääb

antud mooduli võimalustest väheks. WiFi moodulit kasutatakse vaid ilmaandmete päringuks. Mooduli eelis võrreldes teiste Arduino WiFi arendusplaatide ja moodulitega on tema hind, mis algab kolmest eurost [19]. Seepärast on selline moodul hea valik, kui on vajalik mikrokontroller ühendada internetiga.



Joonis 8. ESP8266 ESP-01 WiFi moodul [18].

WiFi mooduli küljes on 8 viiku (vt joonis 9). RX ja TX viigud on andmete vastuvõtmiseks ja saatmiseks. VCC külge ühendatakse vooluallikas ja GND viik on vooluringi maandamiseks. CH_PD viigu abil saab moodulit sisse ja välja lülitada [20]. RESET viik on mooduli lähtestamiseks. GPIO 0 ja GPIO 2 viike kasutatakse WiFi mooduli arendustarkvara uuendamiseks [20].



Joonis 9. ESP8266 ESP-01 viigud [21].

Moodul opereerib 3,3V pingega. Arduino arendusplaadil on 3,3V pingega toitepesa, mistõttu ei pea kõrvalist vooluallikat kasutama. Ülejäänud Arduino toite-ja signaalitöötluspesad toimivad aga 5V pingega. Seega tuleks kasutada pingeregulaatorit, mis muundaks 5V pinge 3,3V pingeks, et ühendada mooduli TX ja RX viike Arduinoga.

WiFi mooduliga kaasasolev arendustarkvara põhineb AT käskudel [18], mis on käsustik millega juhitakse enamasti kommunikatsioonivahendeid. Tabelis 1 on näha olulisemad AT käsud, mis on vajalikud projekti tarbeks, et WiFi moodulit kasutada andmete vastuvõtmiseks. Pikemat käskude loendit koos selgitustega saab lugeda aadressil *https://room-15.github.io/blog/2015/03/26/esp8266-at-command-reference/*.

Tabel 1. AT käsud [22].

Käsk	Funktsioon	Vastus
AT	Kontroll, kas moodul töötab.	OK
AT+RST	Lähtestamine.	OK
AT+GMR	Arendustarkvara versiooni kuvamine.	Oleneb arendustarkvarast. Enamasti kuvatakse versioon ja tootja info.
AT+CIOBAUD=<boodikiirus>	Boodikiiruse muutmine.	OK
AT+CWMODE= (1-3)	Mooduli töörežiimi seadistamine. Režiimi 1 puhul käitub moodul andmejaamana, 2 puhul pääsupunktina ja 3 puhul mõlema eelnevana.	+CWMODE:(1-3) OK
AT+CWLAP	Pääsupunktide loendi kuvamine.	Loend pääsupunktidest.
AT+CWJAP=<Pääsupunkti nimi>,<parool>	Pääsupunkti ühendamine.	WIFI CONNETED WIFI GOT IP OK
AT+CIPSTART=<ühenduse tüüp>, <aadress>, <pordi number>	Ühenduse loomine kliendina.	OK
AT+CIPSEND=<päringu pikkuse arv(ehk sümbolite arv päringus)>	Andmete/Päringu saatmine.	SEND OK, pärast mida ilmub sümbol >. Seejärel tuleb sisestada päring.
AT+CIPCLOSE	Ühenduse sulgemine.	OK

Kasutades Arduinot WiFi mooduli juhtimiseks on kaks võimalust. Üks võimalus on saata WiFi moodulile käske Arduino programmis, käsuga *Serial.print(AT käsk)*, misjärel tuleb lugeda moodulilt tulevat vastus ühe sümboli kaupa *Serial.read()* funktsiooniga. Teine võimalus on kasutada juba loodud teeke, mis sisaldavad valmis funktsioone. Teekide funktsioonid toimivad samamoodi AT käskude põhjal. Olemasolevate teekide kasutamine tagab WiFi mooduli suurema töökindluse.

2.6 Ultraheli kaugusandur

Käeviibutuse tuvastamiseks kasutatakse ultraheli kaugusandurit HC-SR04. Viibutuse tuvastamiseks sobiks veel liikumisandur, mis tuvastab liikumist anduri tuvastusraadiuses, kuid ei mõõda kaugust. Ultraheli kaugusandur tuvastab vahemaa anduri ja objekti vahel, kuhu andur on suunatud. See annab kaugusandurile suurema kasutuvõimaluse erinevates projektides, kui tavaline liikumisandur.

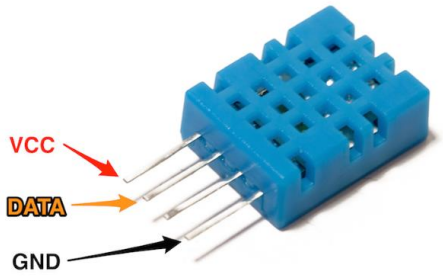
Alapeatüki järgnev osa tugineb HC-SR04 kasutajajuhendil [23]. HC-SR04 on andur (vt joonis 10), mille abil saab mõõta kaugust anduri ja anduri tuvastusalasse jääva objekti vahel. Kauguse mõõtmiseks kasutab HC-SR04 sonarit samal põhimõttel nagu delfiinid või nahkhiired ja kaugust tuvastatakse vahemikus 2-400 cm.

Joonis 10. Ultraheli kaugusandur HC-SR04 [23].

Moodul kasutab nelja viiku. VCC viigu külge ühendatakse vooluallikas ja GND viik on vooluringi maandamiseks. Trig ja Echo viigud ühendatakse digitaalsete pesade külge. Trig viiku kasutatakse, et andurile saata signaal mõõtmise alustamiseks. Kui signaal on saadud alustatakse aja mõõtmisega ja ultraheli andur saadab välja 40kHz heliimpulsi. Kui objektil tagasipõrkav impulss jõuab taas andurini saadetakse selle kohta signaal Echo viigu kaudu ja aeg pannakse seisma. Saadud ajavahemiku pikkust kasutades arvutatakse objekti kaugus.

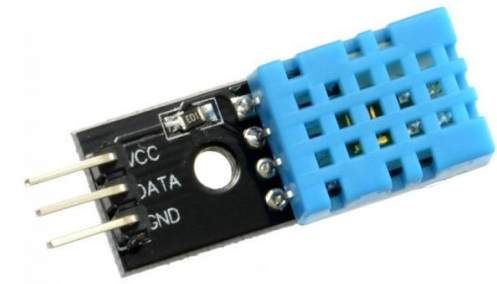
2.7 Temperatuuri ja õhuniiskuse andur

Sisetemperatuuri ja õhuniiskuse tuvastamiseks kasutatakse DHT11 andurit (vt joonis 11). Andur mõõdab suhtelist õhuniiskust ±5% täpsusega vahemikus 5-95%. Temperatuuri mõõtevahemik on -20 kuni 60 kraadi eksimisveaga ±2 kraadi. Müüdavad andurid on kalibreeritud spetsiaalses keskkonnas, mis tagab parema mõõtetäpsuse [24].



Joonis 11. DHT11 andur [25].

Anduri küljes on neli viiku, millest paremalt teist ei kasutata. Vasakult esimese viigu kaudu saab andur toidet, teise kaudu vahetatakse andmeid ja vasakult neljas on maandusviik. Andurist on ka koos trükkplaadiga versioone, mida on näha joonisel 12.



Joonis 12. Trükkplaadiga DHT11 andur [26].

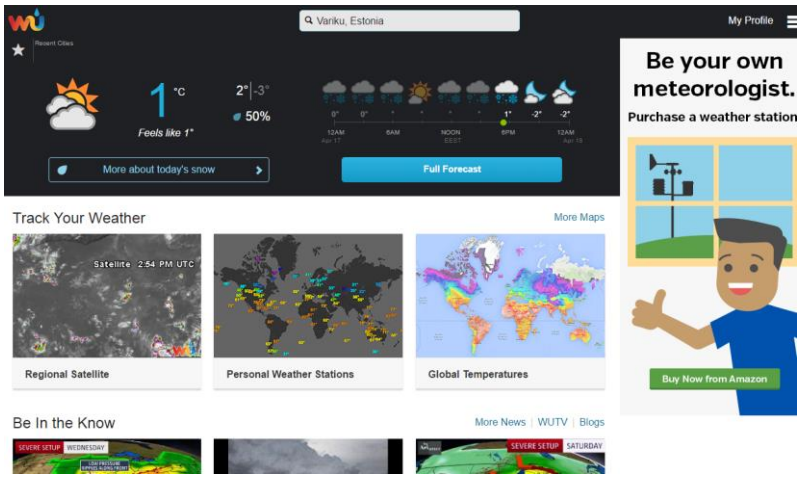
Trükkplaadiga anduril on nelja asemel kolm viiku samade funktsioonidega nagu varem kirjeldatud. Andurilt mõõteväärtuste saamiseks on kõige lihtsam kasutada selleks loodud spetsiaalseid teeke.

2.8 Ilmaandmete allikas

Ilmateataja jaoks on vaja ilmaandmeid, mida kuvada. Käesolevas projektis ei kasutata erinevaid ilmaandmeid mõõtvaid andureid, vaid andmete päring tehakse internetist Weather Underground (vt joonis 13) ilmaportaalist.

Järgnevad lõik tugineb Weather Underground kodulehel [27] oleval infol. Weather Underground pakub infot praeguse ilma kohta, kui ka ilmaennustust. Hetkel valitseva ilma info põhineb lähedal olevate ilmajaamade vaatlusandmetel. Ilmaennustus teenus põhineb patenteeritud ilmaennustussüsteemil *BestForecast*™. See süsteem kasutab innovatiivseid ilmaennustusmudeleid ja ristkontrollib nende väljundeid kohalike vaatluspunktide andmetega. Suurem osa ilmaandmetest kogub süsteem personaalsetest ilmajaamadest, mida on üle maailma kokku rohkem kui 250 000. Personaalsed ilmajaamad on inimeste enda soetatud väiksed ilmajaamad, mis on Weather Undergroundi poolt heakskiidetud. Lisaks personaalsetele ilmajaamadele kogutakse andmeid veel lennujaamades asuvatest mõõtejaamadest ja MADIS (*Meteorological Assimilation Data Ingest System*) süsteemi kuuluvatest ilmajaamadest.

Täpsemalt kasutatakse töös Weather Underground rakendusliidest (API- *Application Programming Interface*). Rakendusliides on reeglistik, kuidas saab kasutada rakenduse teenuseid.



Joonis 13. Kuvatõmmis Weather Underground kodulehelt.

Ilmaportaale, mis pakuvad rakendusliidese tuge on ka teisi, kuid Weather Underground rakendusliidesel on teiste ees mõned eeliseid, mis teevad sellest ühe parima tarkvara arendamiseks. Esiteks, päringut tehes saab valida, et tagastatav info oleks eesti keeles. Olulist rolli mängib see ilmaseisundit kirjeldava väärtuse juures. Kui tagastatav väärtus oleks inglise keeles, tuleks see tõlkida eesti keelde, mis omakorda nõuaks palju Arduino mäluruumi. Teiseks, rakendusliidese kasutajaks registreerimine on tasuta ja käib kiiresti. Teenuse kasutamise eest tuleb maksta vaid siis, kui päringute arv minutis ületab 10 korda või päevas 500 korda [28]. Teised sobilikud rakendusliidesed, mis pakuvad eestikeelsed tuge näiteks Accuweather, nõuavad registreerimiseks meilivahetust teenusepakkuja müügiosakonnaga, kus lepitakse kokku teenuse kasutamise tingimused, sealhulgas hind [29]. Samas pole ka portaali Weather Underground andmete kasutamine täiesti vaba. Portaal nõuab, et koos nende andmetega kuvatakse ka nende logo, mis on kättesaadav kodulehel. Käesolevas projektiks lahendatakse see probleem logo kleepimisega ilmateataja seadme külge, sest ekraanipind on väärtuslik ja logo muutmata on selle kuvamine väiksel ekraanil ebaotstarbekas.

Ilmaandmete päringuks on just rakendusliidese kasutamine efektiivne, kuna selle abil tehtud päringu vastus on JSON (*inglise keeles JavaScript Object Notation*) tekstivormingus [30], kujul „võti“:„väärtus“. Sellisest päringu vastusest saab mõistliku keerukusega eraldada vajamineva info. Kui ilmaandmete päring teha otse veebilehele, oleks vastus HTML (*Hypertext Markup Language*) vormingus tekst, kus on palju ebavajalikku informatsiooni, näiteks veebilehe kujunduse kohta. Vajalike ilmaandmete eraldamine nõuaks mahukamat keeletöötlust, mis hõivaks palju Arduino Uno väärtusliku mäluruumi.

Eespool kirjeldati ülevaatlikult, milline on ilmateataja ehitamiseks vajalik riistvara ja tarkvara. Järgnevalt vaadeldakse, millised tehnilised sammud on vajalikud seadme ehitamiseks.

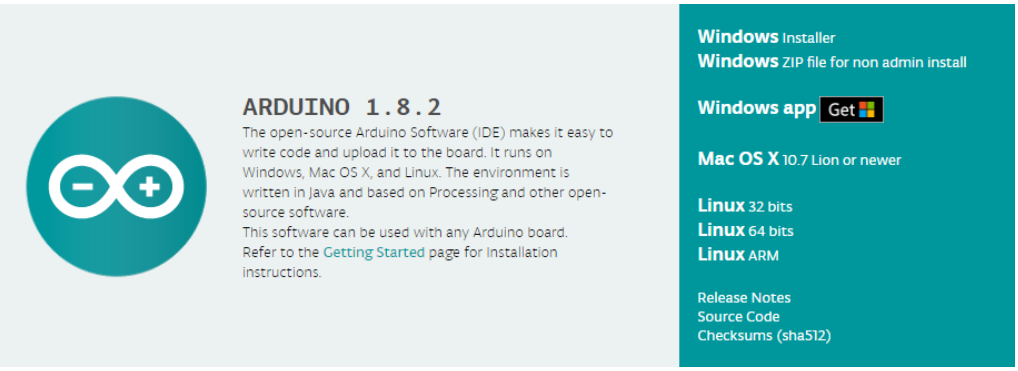
3. Ettevalmistused ilmateataja ehitamiseks

Enne ilmateataja konstrueerimist tuleb teha riistvaralisi ja tarkvaralisi ettevalmistusi. Selles peatükis kirjeldatakse, kuidas paigaldada arvutisse vajaminev tarkvara programmeerimiseks. Ühtlasi näidatakse, kuidas kasutada ja seadistada põhilisi riistvarakomponente Arduinot, 2,8 tollist vajutustundliku ekraani ja WiFi moodulit.

3.1 Arduino arenduskeskonna paigaldamine

Arduino programmerimiseks kasutatav arenduskeskkond tuleb internetist alla laadida ja arvutisse paigaldada. Arenduskeskkonda saab alla laadida aadressilt https://www.arduino.cc/en/Main/Software. Lehel pakutakse Arduino arenduskeskkonna veebiversiooni ja töölaua versiooni. Arduino projekte saab teha mõlemaga, kuid, et oleks võimalus internetiühenduse puudumisel programmeerida, kasutatakse käesolevas projektis töölaua versiooni.

Arduino arenduskeskkonna töölaua versiooni on võimalik paigaldada Windows, Mac OS ja Linux operatsioonisüsteemidele nagu on näha joonisel 14. Valida tuleb sobiv operatsioonisüsteem. Windows operatsioonisüsteemi jaoks soovitatavalt *"Windows Installer"* valik.



Joonis 14. Arduino arenduskeskkonna allalaadimisvaade [8].

Pärast sobiva variandi valimist avaneb aken, kus palutakse rahaliselt toetada Arduino arenduskeskkonda. Tasuta allalaadimiseks, vajutada nupule „*JUST DOWNLOAD"*. Seejärel laetakse arenduskeskkonna paigaldaja arvutisse. Kui allalaadimine on lõppenud, tuleb

paigaldamiseks avada allalaetud EXE-fail ja järgida paigaldusjuhendit arenduskeskkonna paigaldamiseks.

Programmeerimisel läheb Arduino arenduskeskkonnas kõige rohkem vaja kolme nuppu, mis on joonisel 15 tähistatud numbritega 1, 2 ja 3.

Joonis 15. Olulised nupud Arduino arenduskeskkonnas.

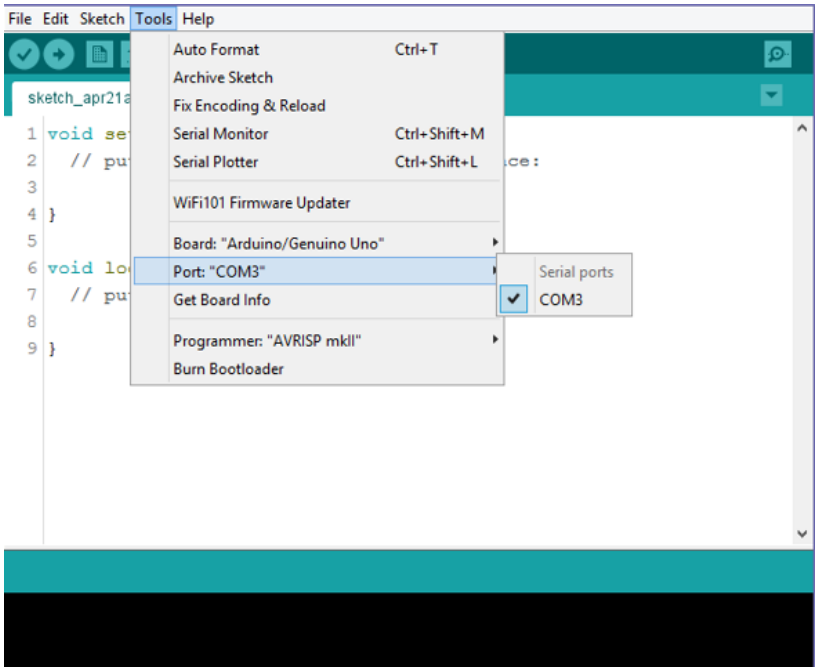
Nupule numbriga 1 vajutades, kontrollitakse, kas kirjutatud kood kompileerub. Nupule numbriga 2 vajutades, laaditakse programm Arduino arendusplaadile. Nupp numbriga 3 avab *Serial Monitori* ehk jadapordi monitori, kuhu saab kuvada andmeid Arduino jadapordiliidese kaudu käsuga *Serial.print()*.

3.2 Riistvara komponentide testimine ja seadistamine

Projektis kasutatavat riistvara müüvad mitmed pakkujad, kes paigaldavad seadmetele erinevat seadistust. Seetõttu tuleb veenduda, et need töötavad korrektselt ja toimivad töös kasutatavate teekidega. Järgnevates alapeatükkides näidatakse, kuidas seda teha ja ühtlasi tutvustatakse selle käigus ka ekraani algseadistamist.

3.2.1 Arduino ühendamine arvutiga ja näidisprogrammi käivitamine

Arduino arendusplaadile programmi laadimiseks tuleb see ühendada arvutiga USB kaabli abil. Kui ühendus on loodud, tuleb avada Arduino arenduskeskkond ja *„Tools“* menüüst *„Board“* menüü alt valida *„Arduino/Genuino Uno“ ja „Port“* menüü alt saadavalolev port. Seaded peaksid välja nägema sarnased nagu näha joonisel 16, erineda võib pordi number. Kui ühtegi porti saadaval pole, siis järelikult ühendus Arduino ja arvuti vahel ei toimi. Sel juhul tuleks proovida teisi arvutis olevaid USB pistikuid ühenduse loomiseks või vahetada USB kaablit.



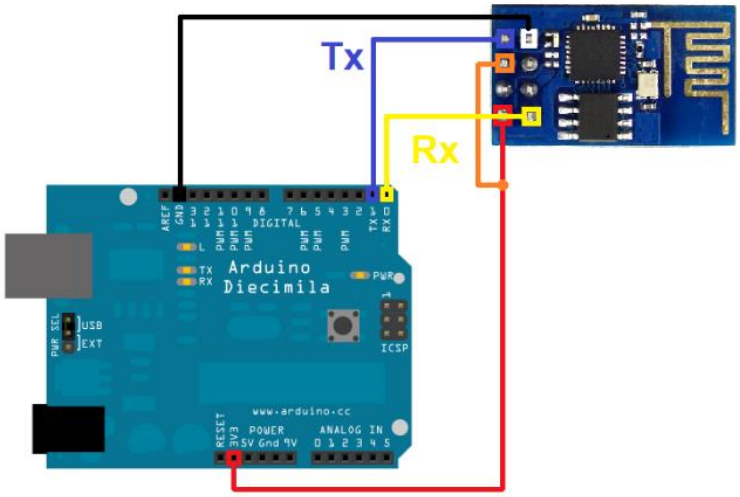
Joonis 16. Arduino ühendamine arvutiga.

Hea viis kontrollida, kas Arduino ühendus arvutiga ja arendusplaat ise toimib, on laadida arendusplaadile näiteprogramm *Blink*. *Blink* on üks paljudest näiteprogrammidest, mis on kaasas Arduino IDE arenduskeskkonnaga. Selleks, et näiteprogramm käivitada, tuleb „*File*“ menüüst valida *Examples -> Basics -> Blink*. Avanenud programm laadida Arduino arendusplaadile. Kui kõik toimib, hakkab arendusplaadil vilkuma LED tuli ühesekundilise intervalliga.

3.2.2 WiFi mooduli testimine

Kuna ESP8266 kiibil põhinevaid WiFi mooduleid pakuvad mitmed mitteametlikud tootjad, on vajalik enne mooduli töö alustamist kindlaks teha, et mooduliga kaasasolev arendustarkvara toimib ja moodul ühendub WiFi võrku. Lisaks demonstreerib mooduli katsetamine hästi, kuidas AT käsud toimivad.

WiFi mooduli testimiseks tuleks moodul ühendada Arduino arendusplaadiga nagu on näidatud joonisel 17. Mooduli testimiseks ei kasutata Arduinot mikrokontrollerina vaid vahelülina arvuti ja Arduino vahel ning seetõttu tuleb Arduino TX pesaga ühendada WiFi mooduli TX viik ja RX pesaga RX viik.



Joonis 17. Arduino ühendamine WiFi mooduliga testimiseks [31].

Moodulit tutvustavas peatükis kirjutati, et antud WiFi moodul opereerib 3,3V vooluga ja seetõttu peaks mooduli TX ja RX viigud ühendama läbi pingeregulaatori. Vaadeldav ühendusskeem, aga pingeregulaatorit ei kasuta. Ühendused on selliselt tehtud, kuna leidub allikaid, mis ütlevad, et TX ja RX viike võib ka ilma regulaatorita ühendada [21]. Töö autor otsustas samuti otseühendamise kasuks, kuna läbi regulaatori ühendatud WiFi mooduliga AT käskude vastuseid *Serial Monitoris* välja ei prinditud. Ilma *Serial Monitori* vahetulemusi printimata on programmeerimine väga tülikas, kuna raske on mõista, milliseid andmeid moodul täpselt tagastab. Autor ehitas seadet mitme kuu vältel ja WiFi moodul toimis ilma regulaatorita kogu aja vältel probleemideta. Pärast programmeerimist, kui enam polnud vaja midagi *Serial Monitori* abiga väljastada, katsetas autor moodulit ka pingeregulaatoriga ja moodul täitis ka siis oma ülesandeid korrektselt. Ehk pingeregulaatoriga moodul küll toimib, aga sel juhul *Serial Monitori* andmeid printida ei saa.

Pärast ühenduste tegemist tuleb Arduino arenduskeskkonnas avada uus tühi programm ja laadida see arendusplaadile. Laadimise ajal ei tohi Arduino arendusplaadi TX ja RX pesad olla mooduliga ühendatud olla, sest need on sel hetkel kasutusel Arduino programmeerimiseks. Kui programm on arendusplaadile laetud, tuleb avada Arduino arenduskeskkonnas *Serial monitor*. Seejärel ülemisele tekstisisestusribale sisestada käske, mis on joonisel 18 tähistatud punaselt.

Joonis 18. AT käsud WiFi mooduli testimiseks.

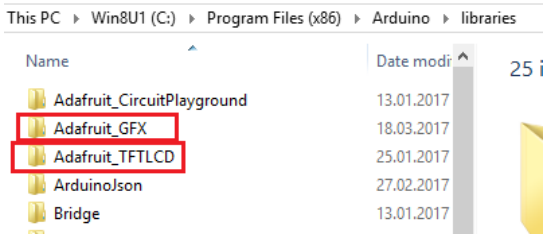
Käskude all on vastused käskudele. Käsk *AT* kontrollib, kas moodul töötab ja *AT+GMR*, kuvab mooduli tarkvara versiooni. Käsk *AT+CWLAP* kuvab loendi kohalikudest pääsupunktidest ning käsu *AT+CWJAP* abil saab mooduli ühendada pääsupunktiga. Käskude vasted võivad veidi erineda olenevalt WiFi mooduli tarkvara versioonist, kuid üldjoontes on sarnased. Katsetada võib veel teisi AT käske, mida on kirjeldatud AT käskude tabelis (vt tabel 1) peatükis 2.5.

3.2.3 Ekraani ühendamine Arduinoga ja teekide testimine

2,8 tollise ekraani testimiseks on vajalik ekraani küljes olevad viigud ühendada Arduino pesade külge, mis sobituvad omavahel ainult ühte moodi. Ilmateataja lõppversioonis, ühendatakse ekraan Arduino külge natukene teistmoodi, mida on vastava osa juures selgitatud.

Kui ekraan on ühendatud, tuleb alla laadida ekraani kasutamiseks vajalikud Adafruiti poolt loodud teegid, *Adafruit_GFX*, *Adafruit_TFTLCD* ja *TouchScreen*. *Adafruit_GFX* teegi saab

alla laadid Githubi lehelt https://github.com/adafruit/Adafruit-GFX-Library vajades nupule „*Clone or download*“, ja seejärel „*Download ZIP*“. Samamoodi tuleb teha teegiga *Adafruit_TFTLCD* lehelt https://github.com/adafruit/TFTLCD-Library ja *TouchScreen* teegiga, mille saab alla laadida lehelt https://github.com/adafruit/Touch-Screen-Library. Selleks, et teeke manuaalselt lisada, tuleb teegi kaust paigutada Arduino rakenduse kausta *libraries*. Tavaliselt on selle asukohaks *C:\Program Files (x86)\Arduino\libraries.* Joonisel 19 on näide kahest juurde lisatud teegist *libraries* kaustas.



Joonis 19. Lisatud teegid.

Ekraani ja selle jaoks vajalike teekide toimimise testimiseks tegi töö autor programmi *Ekraanitest.ino.* Programmi saab alla laadida lehelt Github https://github.com/MartinJLiba/2.8EkraaniTest. Testprogramm kirjutab ekraanile „Tere maailm“ ja pärast ekraanile vajutamist teksti „Puudutus“. Näidisprogrammi sisule selles töö etapis ei keskenduta, kuna ekraani programmeerimisvõtteid tutvustatakse neljandas peatükis.

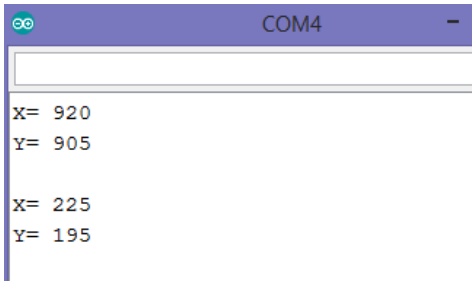
3.2.4 Ekraani kalibreerimine ja identifikaatori tuvastamine

Ekraanidel on erinevad identifikaatorid, mida kasutakse ekraaniga ühenduse loomiseks. Veel on vaja iga ekraan kalibreerida, et vajutustundlikus toimiks kasutatava ekraaniga korrektselt. Järgnevalt on selgitatud, kuidas tuvastada identifikaator ja kalibreerida ekraan.

Ekraani identifikaatori leidmiseks on *TFT_LCD* teegis funktsioon *readID()*. Identifikaatorit tasub pärast tuvastamist salvestada, kuna nõuab palju Arduino mälu, mida saab kasutada näiteks ikoonide joonistamiseks. Identifikaatori leidmiseks tuleb Ekraan ühendada Arduinoga ja Arduinole laadida juba eespool kasutatud programm *Ekraanitest.ino.* Seejärel

avada *Serial Monitor*, kuhu prinditakse identifikaator, mis tuleks salvestada, et seda kasutada ilmateataja programmi jaoks.

Ekraani vajutustundlikuse funktsionaalsuse kasutamiseks tuleb ekraan kalibreerida, sest muidu ei saa tuvastada täpseid koordinaate ekraanil. Antud ekraani kalibreerimiseks pole ühtegi programmi, mis väljastaks maksimaalsed ja minimaalsed pikkus-ja laiuskoordinaadid ja seetõttu tuleb seda teha käsitsi. Kalibreerimiseks tuleb taaskord kasutada programmi *Ekraanitest.ino*. Kui ekraan töötab, tuleb avada *Serial Monitor* ja vajutada ekraaninurgale, kus x-ja y-koordinaadid on kõige suuremad ja nurgale, kus need on kõige väiksemad (vt joonis 20).



Joonis 20. Ekraani koordinaatide lugemine.

Saadud maksimaalsed ja minimaalsed koordinaadid tuleb kirjutada programmis olevatesse muutujatesse (vt joonis 21), mida kasutatakse hiljem ka ilmateataja programmeerimisel ja saab kasutada ka teistes ekraani kasutavates projektides.

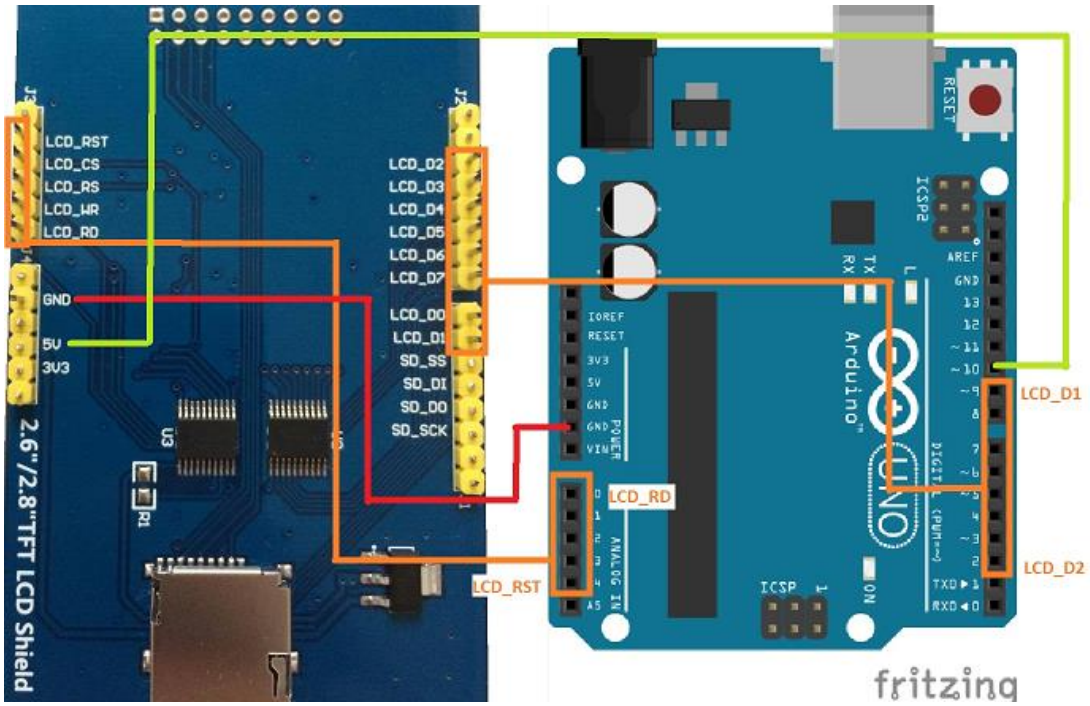
```
#define Minimum_x 225
#define Minimum_y 195
#define Maximum_x 920
#define Maximum_y 905
```

Joonis 21. Ekraani maksimaalsete ja minimaalsete koordinaatide muutujad.

Eelnevalt kirjeldati, kuidas valmistada ette riistvara, mida kasutatakse Arduino ilmateataja projektis. Järgnevalt on näidatud, kuidas omavahel kõik riistvara komponendid ühendada.

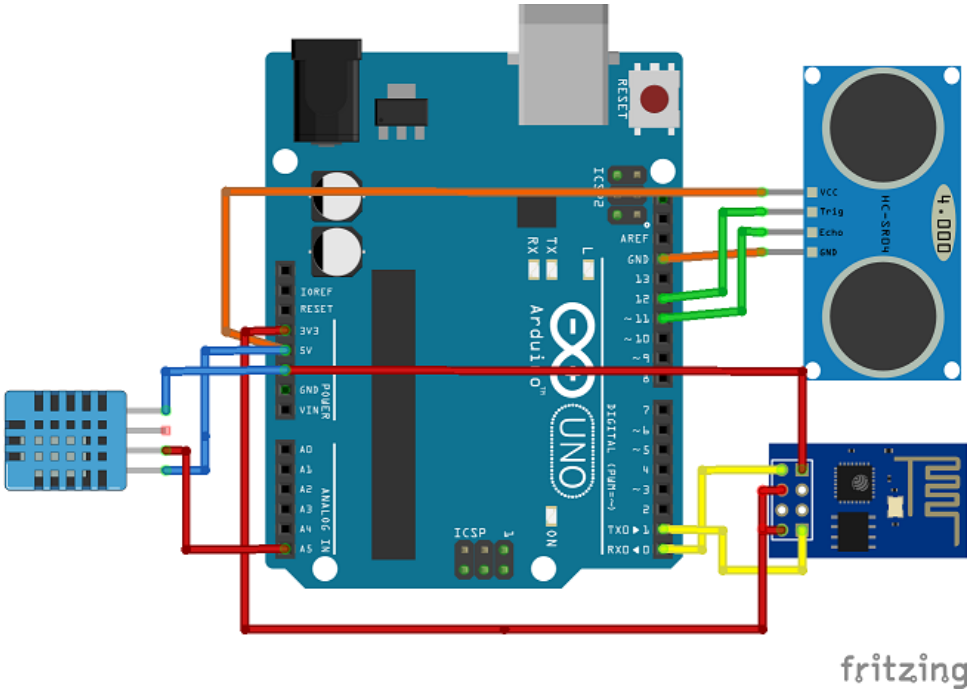
3.3 Riistvara ühendamine ilmateataja tarbeks

Ekraani, WiFi mooduli, kaugusanduri ning temperatuuri ja niiskuse anduri ühendamiseks kasutatakse ära enamus Arduino pesadest. Suurema osa nendest pesadest kulub ekraani ühendamiseks. Ekraani ei saa enam arendusplaadi külge ühendada nagu näidati ekraani testimise puhul peatükis 3.2.3. Ühendusi on vaja muuta, kuna muidu ei ole ruumi teiste andurite ja moodulite jaoks, milleks on ultraheli kaugusandur ja WiFi moodul. Ekraani ühendused Arduinoga on esitatud joonisel 22.



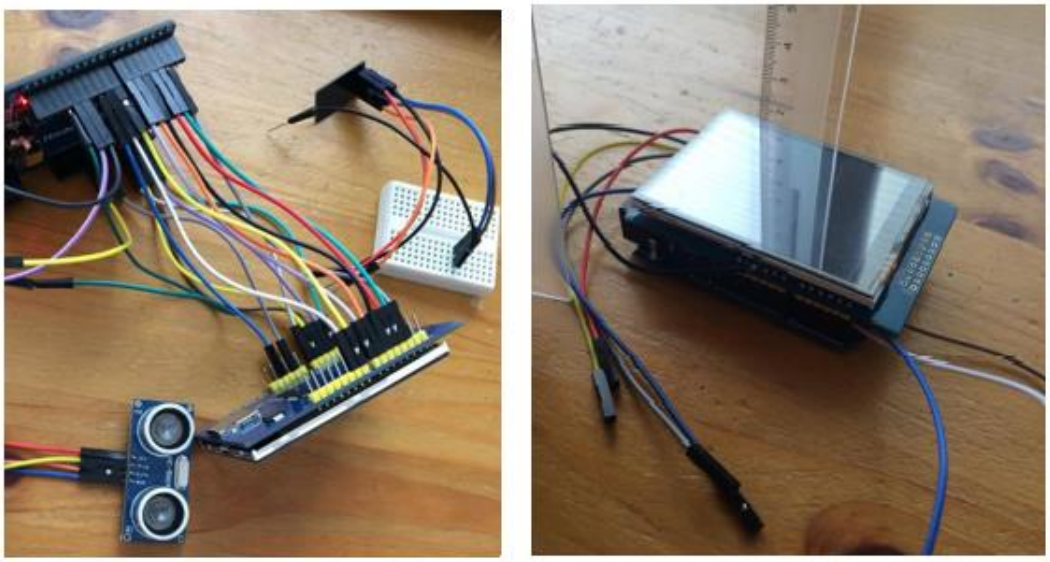
Joonis 22. Ekraani ühendamine Arduinoga. Joonisel olev Arduino Uno on Fritzing tarkvarast.

Joonisel 22 näidatud viisil on ekraan ühendatud Arduinoga kasutades vähimat võimalikke arvu ekraaniviike, et toimiks ekraan ja vajutustundlikus. SD viike ei ühendata, kuna projektis ei kasutata andmete lugemist või salvestamist microSD mälukaardile. Peale SD viikude on erinevalt ühendatud ekraani 5V viik, mis on ühendatud Arduino digitaalpesaga 10. Seeläbi kaotab ekraan vähesel määral taustakuva heledust, kuid on vajalik, ekraani väljalülitamiseks, et säästa seadme voolutarvet. Andurite ja WiFi mooduli ühendamist ekraanist vabaksjäänud Arduino pesadega on näidatud joonisel 23.



Joonis 23. WiFi mooduli, temperatuuri-ja õhuniiskusanduri ning kaugusanduri ühendamine Arduinoga. Joonise koostanud autor Fritzing tarkvaraga.

Ilmateataja kompaktseks väljanägemiseks lõppversioonis joodetakse riistvara ühendusjuhtmed Arduino arendusplaadi külge. Viigud, mis on ekraani küljes, kuid mida ei kasutada, saab näiteks näpitsatega ära lõigata. Lõigatud ekraaniviike on näha ka joonisel 22. Siiski on soovitatav teha jootmist alles siis, kui programmeerimine on lõpetatud. Selleks, et programmeerimise ajal saaks kõiki riistvara komponente koos kasutada, saab ühendusi teha kasutada spetsiaalseid ühendusjuhtmeid, mida on näha joonisel 24. Joonisel 24 on ka kokku joodetud ilmateataja versioon.



Joonis 24. Vasakul ilmateataja programmeerimise ajal, paremal pärast ühenduste jootmist.

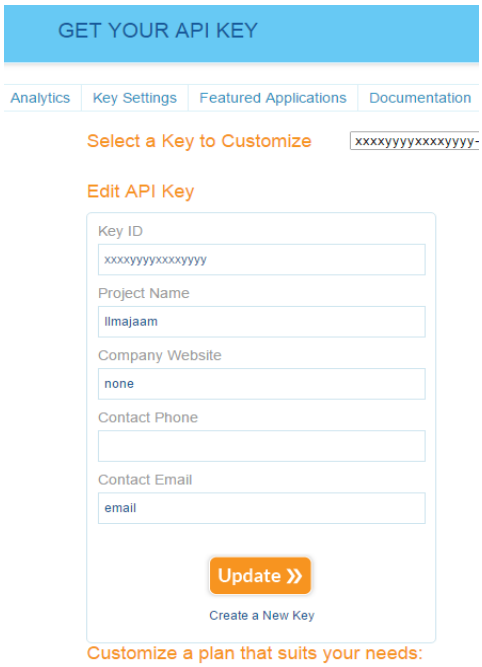
Eelnevas peatükis selgitati, kuidas ilmateataja riistvara komponendid ühendada. Enne programmeerimist on vajalik ka registreerida rakendusliidese võti, mida kirjeldatakse järgmises peatükis.

3.4 Rakendusliidese võtme registreerimine

Weather Undergoround rakendusliidese kasutamiseks tuleb registreerida unikaalne võti, mis tagab ligipääsu lehe rakendusliidesele. Võti koosneb tähtedest ja numbritest. Päringu tegemisel lisatakse võti päringu juurde ja seeläbi määratakse kindlaks, millisele infole on kasutajal juurdepääs. Võtme registreerimiseks tuleb minna internetilehele https://www.wunderground.com/signup?mode=api_signup ja täita nõutud väljad konto loomiseks. Kui konto on loodud, tuleb vajutada lehe allosas olevale lingile „*weather api*“ ja seejärel nupule *„Explore my options“*. Pakutavatest variantides tuleb valida *„Cumulus plan“* ja „*developer“* nagu on valitud joonisel 25. Pärast seda vajutada nupule *„Purchase key“*.

Joonis 25. Võtme valikud, kuvatõmmis Weather Underground rakendusliidese lehelt.

Edasi tuleb täita nõutud väljad ning vajutada uuesti nupule „*Purchase key*“. Genereeritud võti (vt joonis 26) on näha lehel „*Key Settings*“.



Joonis 26. Genereeritu võti, Kuvatõmmis Weather Underground rakendusliidese lehelt.

Testimaks, kuidas näeb välja päringu vastus, mis on JSON tekstivormingus tuleb sisestada päring http://api.wunderground.com/api/võti/conditions/hourly/lang:ET/q/autoip.json, brauseri aadressiribale, kus „võti” asendada genereeritud võtmega. Seda päringut läheb vaja ka programmeerimisel ilmaandmete saamiseks.

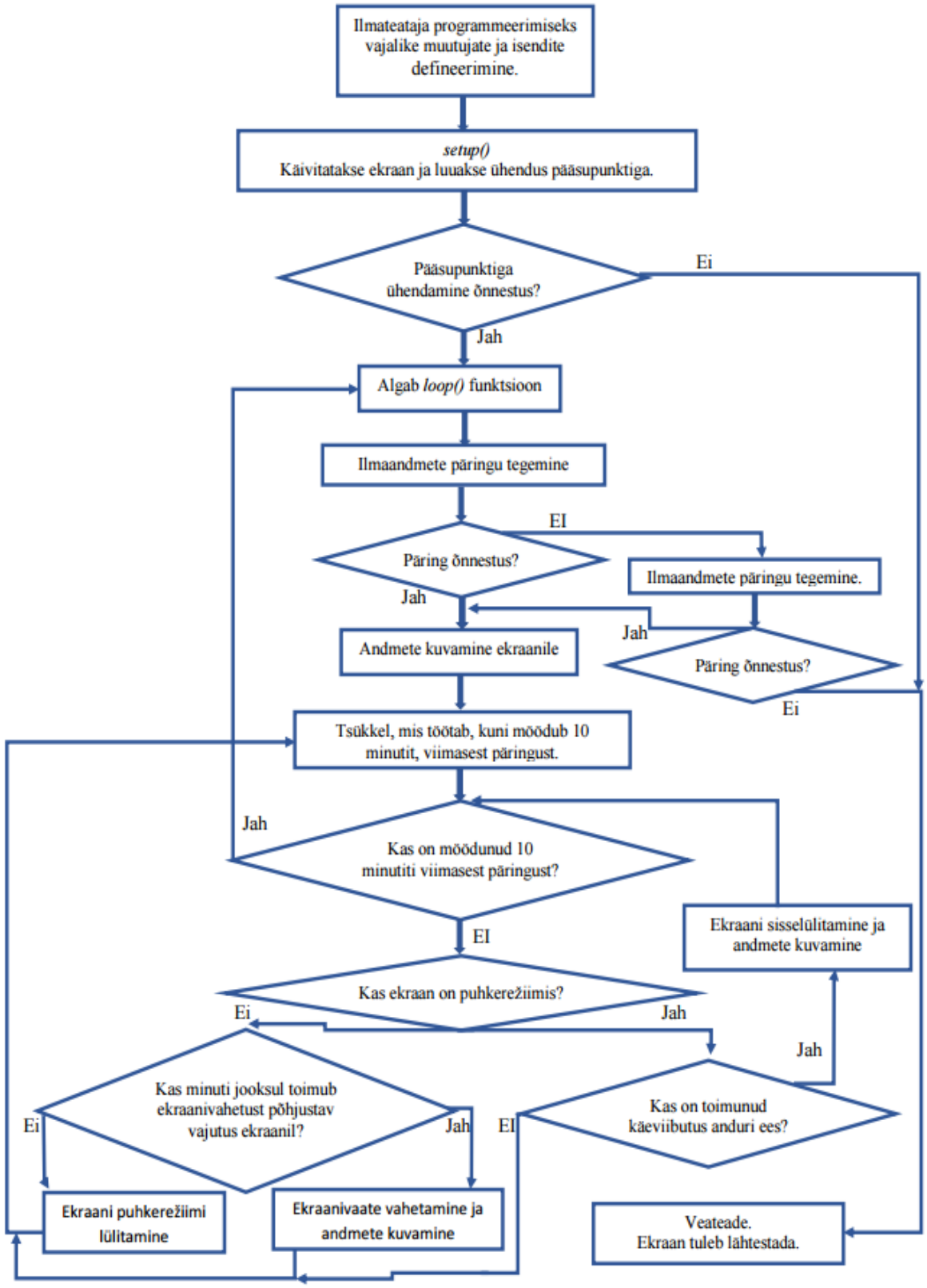
Antud peatükis tehti kindlaks, et kasutatav riistvara toimib korrektselt. Ühendati riistvara komponendid Arduinoga ja seadistati ekraani ilmateataja programmeerimiseks. Järgmises peatükis näidatakse põhilisi programmeerimisevõtted, mis on vajalikud ilmateataja tegemiseks.

4. Ilmateataja programmeerimine

Järgnevates alapeatükkides selgitatakse, millised teegid tuleb kasutusele võtta ilmateataja programmeerimiseks ja kuidas programmeerida andmete päring ning ekraanile kuvamine.

4.1 Ilmateataja programmi loogika üldskeem

Ilmateataja jaoks kirjutatud programm *ilmateataja.ino* asub projekti kaustas, mis on kättesaadav Github repositooriumis https://github.com/MartinJLiba/Arduino_ilmateataja
Programmiloogika paremaks mõistmiseks on koostatud programmi üldine voosekeem (*flow chart*) (vt joonis 27). Vooskeem aitab paremini mõista, mis järjekorras programmis erinevaid toiminguid sooritatakse. Ristkülikud tähistavad protsessi ja rööpkülikud otsuse tegemist.



Joonis 27. Ilmateataja programmi loogikaskeem.

Järgnevates alapeatükkides kirjeldatakse, milliseid programmeerimisvõtteid kasutati vooskeemil olevate protsesside lahendamiseks. Seeläbi on näha, kuidas kasutada Arduinoga vajutustundliku ekraani ja luua internetiühendus.

4.2 Teekide lisamine ja kasutuselevõtt

Vajutustundliku ekraani ja WiFi mooduli programmeerimiseks on vaja Arduino arenduskeskkonda lisada teeke. Ekraani jaoks vajalikud teekide lisamist tutvustati juba ekraani testimise käigus peatükis 3.2.3. WiFi mooduli programmeerimiseks tuleb lisada AT käskudel põhinev teek *ESP8266*, mille esialgne autor on Itead Studio. Töös kasutatakse circuit team poolt modifitseeritud varianti, mille saab alla laadida lehelt Github https://github.com/Circuito-io/ESP8266_SoftwareSerial. Temperatuuri ja niiskusanduri kasutamiseks on teek dht, mille saab alla laadida lehelt Github https://github.com/RobTillaart/Arduino/tree/master/libraries/DHTlib. Teekide paigaldamine käib samamoodi nagu kirjeldati peatükis 3.2.3.

Arduino ilmateataja projektiks on enne *ESP8266* teegi kasutamist vaja selles teha mõned muudatused. WiFi mooduliga suhtlemiseks kasutatakse Arduino jadapordiliidest. ESP8266.h failis tuleb välja kommenteerida või kustutada faili alguses asuv rida *#ifdef ESP8266_USE_SOFTWARE_SERIAL*, seeläbi kasutab teek Arduinoga suhtlemiseks jadapordiliidest. Teiseks, tuleb avada teegi kaustas olev fail ESP8266.cpp. Selles failis on vaja eemaldada kõik read, mis prindivad infot *Serial Monitori* käsuga *Serial.println().* See on vajalik, et hoida kokku Arduino mäluruumi. Eelnevaga on vajalikud muudatused tehtud ja teek on kasutamiseks valmis.

Teekide kasutuselevõtuks Arduino keskkonnas kasutatakse *#include* võtmesõna, mille järgi läheb teegi nimi. Teine viis teeke lisada on menüüst *sketch->add library*, kust tuleb valida soovitud teek. Veel on vajalik teekide kasutamiseks defineerida vajalikud viigud, mille kaudu käib moodulitega suhtlus ja luua objektid, mille abil saab teekides olevaid funktsioone välja kutsuda. Näiteks ekraanile andmete kuvamiseks tehakse objekt *tft*, mille kaudu prinditakse ekraanile teksti käsuga *tft.print()*. Viike defineeritakse *#define* võtmesõna abil ja objektid luuakse vastavalt teegis ettenähtud konstruktorile. Programmeerimiseks

vajalike teekide kasutuselevõttu ja objektide loomist kirjeldab joonis 28. Eeskuju ekraani viikude defineerimiseks ja isendite loomiseks saadud Cyan Infinite ekraanitutvustusest [15].

```
//Vajalike teekide kasutuselevõtt
#include <Adafruit_GFX.h>
#include <TouchScreen.h>
#include <Adafruit_TFTLCD.h>
#include <ESP8266.h>
#include <dht.h>
//Vajalike viikude defineerimine ekraani isendi loomiseks.
#define LCD_CS A3
#define LCD_CD A2
#define LCD_WR A1
#define LCD_RD A0
#define LCD_RESET A4
//Vajalike viikude defineerimine Touchscreen teegi isendi loomiseks.
#define YP A3
#define XM A2
#define YM 9
#define XP 8
//Analoogiviigu defineerimine temperatuuri ja õhuniiskuse anduri jaoks
#define dht_pin A5
//Ekraani objekti loomine
Adafruit_TFTLCD tft(LCD_CS, LCD_CD, LCD_WR, LCD_RD, LCD_RESET);
//Vajutustundlikuse isendi loomine
TouchScreen ts = TouchScreen(XP, YP, XM, YM, 300);
//WiFi mooduli kontrollimiseks objekti loomine. Serial tähistab jadapordiliidest.
ESP8266 esp(Serial);
dht DHT; //DHT teegi isendi loomine.
```

Joonis 28. Teekide kasutuselevõtt.

Järgnevalt vaadeldakse, kuidas ilmateataja programmeerimiseks kasutatakse teekides olevaid funktsioone.

4.3 Ekraaniga ühenduse loomine ja väljalülitamine

Ekraanile kujutiste kuvamiseks tuleb kõigepealt ekraaniga ühendus luua. Kuna ekraani vooluviik ühendati digitaalpesaga 10, tuleb see seada väljundiks ja viia pinge 5 voldiseks märksõnaga *HIGH*. Veel on vajalik ekraan lähtestada funktsiooni *reset()* abil ja seejärel ühendus luua funktsiooniga *begin()*. Funktsiooni *begin()* argumendiks on peatükis 3.2.4 leitud ekraani identifikaator. Ekraaniga ühenduse loomise koodi demonstreerib joonis 29.

```
pinMode(10, OUTPUT); //Digitaalviik D10 seadistamine väljundiks
digitalWrite(10, HIGH); //Digitaalviigu D10 5 voldiseks  määramine
tft.reset(); //Ekraani lähtestamine
uint16_t identifier = 37697; //Ekraani id muutuja väärtustamine
tft.begin(identifier); //Ekraaniga ühenduse loomine
```

Joonis 29. Ekraaniga ühenduse loomine.

Käesolevas projektis kasutatava ekraani väljalülitamiseks ühtegi olemasolevat funktsiooni varem polnud ja ekraanil endal puudub ka selline funktsionaalsus. Välja tuli mõelda alternatiivne lahendus. Ainus variant antud ekraani väljalülitamiseks on toitevoolu eemaldamine. Selleks loodi funktsioon *dimm()*, mis *digitalWrite()* funktsiooni kasutades ekraaniviikudega viib pesad madalaks ehk 0V peale. Seetõttu on ka ekraani toiteviik ühendatud digitaalpesaga D10, mitte Arduino 5V toitepesaga, kuna Arduino 5V toitepesa funktsionaalsus pole tarkvaraliselt muudetav. Funktsiooni *dimm()* kirjeldab joonis 30.

```
void dimm(){
  //Ekraani väljalülitamiseks, muudetakse kõik viigud, millega ekraan ühendatud on LOW peale, ehk neis on pärast seda 0V.
  digitalWrite(10, LOW); //Sellega on ühendatud ekraani 5V viik
  digitalWrite(2, LOW);
  digitalWrite(A0, LOW);
  . . . //kõik digitaal-ja analoogviigud, millega ekraan on ühendatud A0-A4, (D)2-(D)10
  }
```

Joonis 30. Ekraani väljalülitamise funktsioon *dimm()*.

Nagu varem mainitud, kaotas ekraan toiteviigu ühendamisel arendusplaadi digitaalpesaga vähesel määral oma taustavalguse heledusest, kuna ekraani voolutarve on veidi suurem, kui digitaalpesa suudab pakkuda. Võib tekkida küsimus, et kuidas ekraan üldse toimib, kui digitaalpesa ei paku piisavalt suurt voolutugevust. Ekraan suudab endiselt toimida, kuna ei saa voolu ainult ekraani vooluühendusviigu kaudu, vaid kasutab selleks ka teisi viike, mille kaudu ekraan on ühendatud. See võimaldab kasutada voolu võtmist Arduino digitaalpesast 5V pesa asemel. Tehtud muudatus ei häiri ekraani tööd ja endiselt on ekraanil kuvatav hästi nähtav.

Selleks, et ekraan taas sisse lülitada, tuleb funktsiooni *digitalWrite()* ja võtmesõna *HIGH abil* digitaalpesasse D10 pinge 5V peale viia. Seejärel kasutada funktsooni *begin()*, mis viib pinge 5 voldiseks ülejäänud pesades ja loob ühenduse ekraaniga.

Ekraan kasutab sisselülitatuna 25mA (milliamprit) voolu ja välja lülitatuna 0mA voolu. Ehk puhkerežiimis tarbivad voolu vaid WiFi moodul, kaugusandur ja DHT andur.

4.4 WiFi mooduli pääsupunktiga ühendamine ja päringu tegemine

Enne ilmaandmete päringu tegemist konfigureeritakse moodul ning seejärel ühendatakse pääsupunktiga, mis tagab juurdepääsu internetiühendusele. Funktsioon *ESP8266* teegis, mis seab WiFi mooduli andmeedastusjaama režiimi ja ühendab pääsupunktiga on *init()*. Funktsiooni argumentideks on pääsupunkti nimi ja parool. Antud funktsioon tagastab tõeväärtuse, kas ühendus õnnestus või mitte. Funktsiooni väljakutsumist demonstreerib allolev joonis (vt joonis 31).

```
esp.init(SSID, PASSWORD)
```

Joonis 31. Funktsiooni väljakutse pääsupunktiga ühendamiseks.

Ilmaandmete päringu tegemiseks kasutatakse *ESP8266* teegist kahte meetodi. Kõigepealt luuakse funktsiooniga *createTCP*() ühendus lehega, kust päringut tegema hakatakse. Seejärel saadtakse päring funktsiooniga *sendSingle()*, misjärel saab jadapordiliidese kaudu lugeda päringu vastust. Funktsioonide väljakutsumist illustreerib joonis 32.

```
esp.createTCP(adress, 80) //Lehega ühenduse loomine.
esp.sendSingle(request)  //Päringu saatmine
//GET http://api.wunderground.com/api/sinuvõti/conditions/hourly/lang:ET/q/Estonia/Tartu.json HTTP/1.1\r\n\r\n
```

Joonis 32. Kood lehega ühendamiseks ja päringu saatmiseks.

Funktsiooni *createTCP()* argumendiks on lehe aadress, millega ühendus luuakse, antu juhul api.wunderground.com Funktsiooni *sendSingle()* argumendiks antav päring Weather Underground rakendusliidesele on näha joonisel 32. *GET* tähistab, et soovime andmeid pärida. Edasi tuleb päring ise, kus on määratud, et soovime infot praeguste ilmaolude

(*conditions*) ja järgnevate tundide ilmaolude (*hourly*) kohta. Pärast seda määratakse ära, et vastust soovitakse eesti keeles (*lang:ET*). Lõpus on *HTTP* protokolli versioon ja uue rea tähised.

Pärast päringu saatmist loetakse ja töödeldakse saadavaid andmeid funktsioonis *processData()*. Olulisemad Arduino funktsioonid, mida *porcessData()* funktsioonis kasutatakse on *Serial.available()*, mis kontrollib kas jadapordiliidesest on sümboleid lugeda ja *Serial.read()*, mis loeb sümboli jadapordiliidesest. Funktsioonis eraldatakse vajalikku informatsiooni päringu vastusest mis on JSON tekstivormingus (vt joonis 33).


```
"weather":"Peamiselt pilves",
"temperature_string":"43.7 F (6.5 C)",
"temp_f":43.7,
"temp_c":6.5,
"relative_humidity":"69%",
```


Joonis 33. JSON tekstivormingus päringu vastus.

Järgnevalt on kirjeldatud päringu töötlemiseks tehtud funktsiooni *processData()* üldist põhimõtet. Jadapordiliidese kaudu loetakse ükshaaval vastuse sümboleid. Kui esineb jutumärk, lisatakse sümboleid tühjale sõnale järjest juurde, kuni esineb teine jutumärk ehk kokku saadakse jutumärkide vahel olev sõna. Seejärel võrreldakse, kas saadud sõna on võtmesõna nagu näiteks *weather* (vt joonis 33), kui jah, siis salvestatakse võtmesõna väärtus ilmaandmete listi, mis lõpuks tagastatakse.

4.5 Ekraanile teksti kuvamine

Ilmateataja puhul on tekstilise info kuvamine väga oluline. 2,8 tollisele ekraanile teksti kuvamiseks on vajalik mitme funktsiooni kasutamine. Enne ekraanile kujutiste kuvamist tuleb paika seada taustavärv. Selleks on funktsioon *clearScreen()*, mis seab taustavärvi mustaks, joonistab ekraani ümber raami ja jaotab kahe joonega ekraani kolmeks (vt joonis 1). Selleks vajalikke funktsioone on näha joonisel 34.

```
//Funktsioon ekraani puhastamiseks ja piirjoonte joonistamiseks.
void clearScreen(){
 tft.fillScreen(BLACK);//Taustavärvi seadmine mustaks.
 tft.drawLine(0,120,240,120,WHITE);//Infoeraldusjoone joonistamine
 tft.drawLine(0,220,240,220,WHITE);//Infoeraldusjoone joonistamine
 tft.drawRect(0,0,240,320,GREY);//Raami joonistamine ümber ekraani
}
```

Joonis 34. Ekraani kolmeksjaotamise funktsioon.

Programmis kasutatavad värvid tuleks eeldefineerida (vt joonis 35), et sama värvi saaks ühe muutujanimega mitmel pool kasutada.

```
//Värvide definitsioonid.
#define BLACK 0x0000 //Must
#define WHITE 0xFFFF //Valge
#define YELLOW 0xFFE0 //Kollane
. . .
```

Joonis 35. Värvide defineerimise näide.

Teksti kuvamiseks ekraanile on *Adafruit_GFX* teegis funktsioon *print()*. Enne kuvamist on vaja paika seada teksti suurus, värv ja asukoht ekraanil. Teksti kuvamist kirjeldab kood joonisel 36.

```
tft.setTextColor(WHITE); //Teksti värvi määramine
tft.setCursor(50,100); //Argumentideks ekraani x ja y koordinaadid
tft.setTextSize(2); //Teksti suuruse määramine
tft.print("Pilves"); //Ekraanile teksti kuvamine
```

Joonis 36. Teksti kuvamiseks vajalikud funktsioonid.

Teksti kuvamisel ekraanile tekib probleem täpitähtede ja sümbolite kuvamisega. Alljärgnev peatükk kirjeldab, kuidas seda probleemi lahendada.

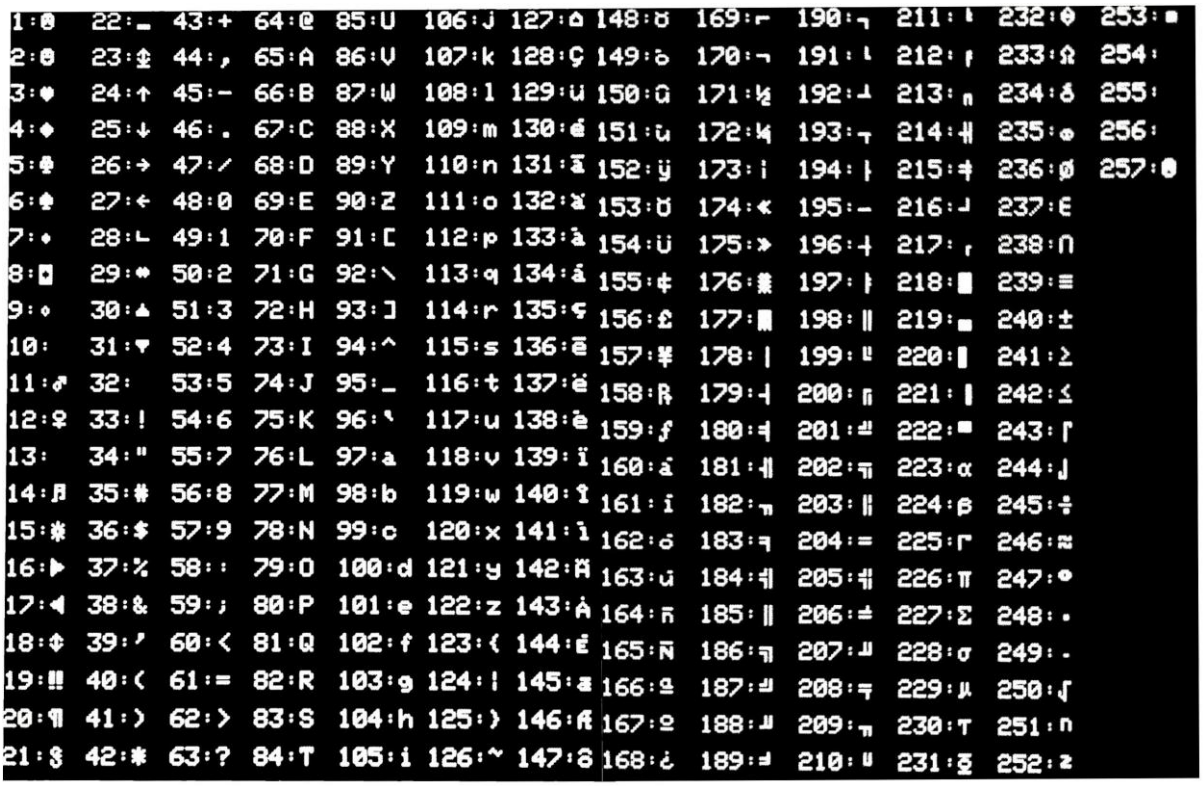
4.6 Täpitähtede ja sümbolite joonistamine ekraanile

Ekraani kasutamiseks valitud teegid, ei võimalda ekraanile kuvada täpitähti ja sümboleid neid lihtsalt *print()* funktsiooni kirjutades. Nende kuvamiseks kasutatakse sümbolite koode, mis teisendantakse sõnaks. Seejärel lisatakse sümbol juurde ülejäänud sõnale. Joonisel 37 on näidatud, kuidas kuvada ekraanil täpitähte õ.

```
tft.print("P"+String(char(147)) +"hjatuul"); //Põhjatuul
```

Joonis 37. Täpitähe kuvamine ekraanil.

Joonisel 38 on esitatud kõik sümbolid ja tähed, mida ekraanil saab kuvada koos nendele vastava koodiga. Kui lihtsalt *print()* funktsiooni kirjutades ei õnnestu sümbolit või tähte ekraanile kuvada, saab kasutada nende kuvamiseks joonisel 38 olevat koodi, kujul *char(kood)*.



Joonis 38. Võimalikud ekraanile kuvatavad sümbolid ja tähed koos koodidega.

Lisaks tekstilisele infole illustreerivad ilmaandmeid erinevad ikoonid. Nende ettevalmistamist ja kuvamist selgitab järgmine peatükk.

4.7 Ekraanile kuvatavate ikoonide ettevalmistamine ja kuvamine

Ilmaandmete visualiseerimiseks ekraanil kasutatakse abistavaid ikoone. Näiteks tuule suunda illustreeritakse noole kuvamisega Eesti kaarti kujutaval ikoonil. Pilves ilma kujutamiseks on ekraanile kuvatud pilve ikoon.

Ikoonide loomiseks ja kuvamiseks juhinduti Nick Koumarise õpetusest [32]. Ikoonide kuvamiseks on funktsioon *drawBitmap() Adafruit_GFX* teegist. Funktsiooni *drawBitmap()* ei saa kasutada nagu teisi teegi funktsioone kujul *tft.drawBitmap()*, kuna *Adfruit_GFX* teek on kirjutatud firma Adafruit poolt müüdavale 2,8 tollisele ekraanile. Teiste sarnaste ekraanide kasutamise puhul võib seetõttu esineda mõne teegi funktsionaalsusega probleeme nagu ka antud funktsiooniga. Seetõttu tuleb programmi kopeerida funktsioon *drawBitmap()* failist *Adafruit_GFX.cpp*, mis asub vastava teegi kaustas. Funktsiooni argumentideks (vt joonis 39) on laius-ja pikkuskoordinaat, ikooni baidimassiiv, ikooni mõõtmed ning ikooni värv.

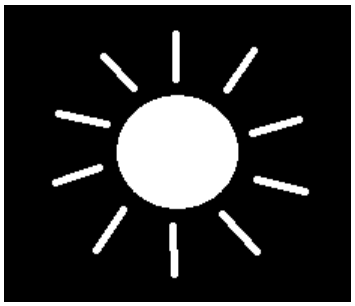
```
drawBitmap(180,220,cloud_icon,60,60,BLUE);//Pilve ikooni joonistamine
```

Joonis 39. Ikoonide kuvamise funktsioon.

Ikooni baidimassiivi valmistamiseks, mis on üheks *drawBitmap()* argumendiks on kolm sammu. Kõigepealt tuleb leida sobiv pilt, mis oleks ühevärvilisel või läbipaistval taustal ja mida oleks võimalik muuta mustvalgeks. Ikooni saab ise joonistada näiteks programmiga MS Paint. Pildi ettevalmistamise protsessi tutvustatakse päikese näitel, mida on näha joonisel 40.

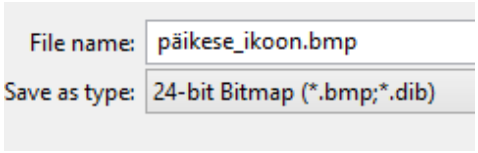
Joonis 40. Päikese esialgne pilt.

Seejärel on vaja pilt muuta mustvalgeks selliselt, et osa pildist mida ekraanil kuvatakse oleks valge ja taust must. Töötluseks sobib taaskord näiteks programm MS paint. Päikese ikooni tegemiseks töödeldud pilt on näha joonisel 41.



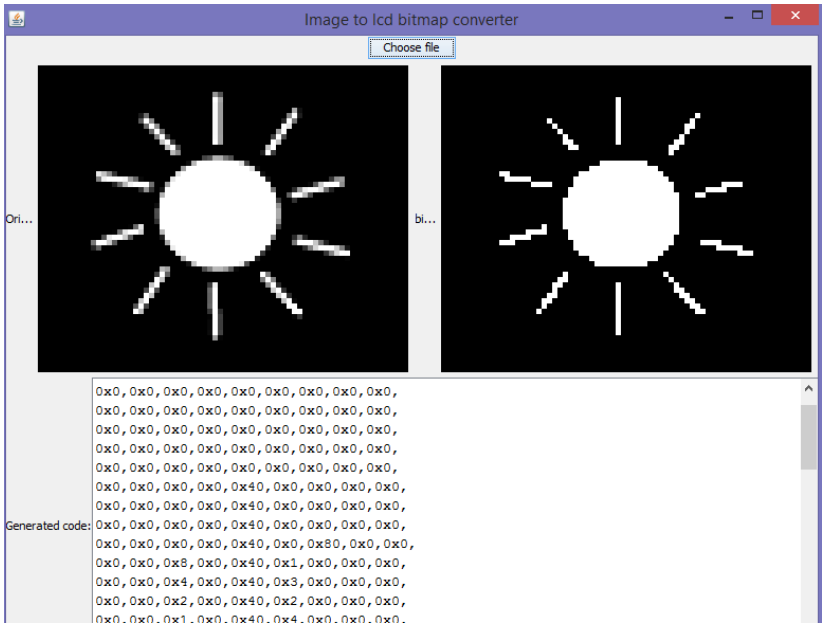
Joonis 41. Töödeldud päikese pilt.

Kui pilt on mustvalgeks värvitud, on vajalik muuta pildi mõõtmeid nii suureks nagu ekraanil seda kuvada on vaja. Mõõtmed tuleb meelde jätta, kuna neid on vaja hiljem funktsiooni *drawBitamap()* argumentideks. 2,8 tollisel ekraanil kuvamiseks on muudetud mõõtmetega päikese pildi suurus 70x70 pikslit. Pikkuse ja laiuse suhe ei pea olema võrdsed, kuid soovituslik on hoida kuvasuhet, et pilt ei moonduks. Ettevalmistatud pilt tuleb salvestada 24-bitises bitmap (BMP) formaadis failina (vt joonis 42).



Joonis 42. Pildi salvestamine BMP failina.

Salvestatud pilt tuleb avada programmiga Img2Code. Programmi saab alla laadida lehelt https://github.com/ehubin/Adafruit-GFX-Library/tree/master/Img2Code. Lehel olevast loetelust vajutada „*Img2Code.jar*“ ja seejärel „*Download*“. Programmi eraldi paigaldama ei pea vaid kasutamiseks avada allalaaditud programm. Programmis (vt joonis 43) tuleb valida eeltöödeldud pilt, mida soovitakse baidimassiiviks teisendada nupuga „*choose file*“. Teisendamise tagajärjel on „*Generated code:*“ aknas ikooni kuvamiseks vajaminev baidimassiiv.



Joonis 43. BMP faili teisendamine baidimassiiviks.

Koodi selguse huvides hoitakse pikki massiive põhiprogrammist erinevas failis. Selleks kopeeritakse baidimassiiv .c laiendiga faili, mille sisu on näidatud joonisel 44. Selleks, et faili sisu kasutada põhiprogrammiga peab see asuma kaustas kus on Arduino programm. Tavaliselt on kõik Arduino programmid kaustas *C:\Users\User\Documents\Arduino*. Faili kasutamiseks tuleb Arduino arenduskeskkond taaskäivitada või lisada fail menüüst *sketch -> add file*.

```
#include <avr/pgmspace.h>
const unsigned char päikese_ikoon []PROGMEM = {} //Siia kopeerida saadud baidimassiiv
```

Joonis 44. Faili sisu, kus hoitakse massiive.

Viimane samm on salvestada baidimassiiv Arduino programmis olevasse muutujasse, mida on näha joonisel 45. Muutuja nimeks on nimi nagu .c failis eelnevalt on defineeritud.

```
extern uint8_t päikese_ikoon[]; //ikooni nimi Arduino programmis
```

Joonis 45. Ikooni muutuja loomine.

Eelnevalt tutvustati, kuidas teha eeltööd funktsiooni *drawBitmap()* kasutamiseks. Funktsiooni kasutamisel on oluline ikooni mõõtmeteks valida samad mõõtmed nagu töötlemise käigus määrati, vastasel juhul ei kuvata ikooni korrektselt.

Iga ilmaolu kujutamiseks jaoks pole mõistlik eraldi baidimassiivi luua, kuna see nõuab palju välkmälu. Ekraanile kuvatavate ilmaolu kujutavate ikoonide jaoks kasutatakse *drawBitmap()* funktsiooniga vaid põhiliste ikoonide joonistamiseks nagu päike ja pilv. Selleks, et kujutada näiteks vihmasadu saab joonistada ekraanile pilveikooni ja selle alla neli ristkülikut, mis kujutavad vihmapiisku (vt joonis 46). Koos moodustavadki kujundid vihmasaju ikooni. Samamoodi saab teha ka lumesaju korral, joonistades valged ringid pilve alla.

```
//Funktsioon vihma ikooni joonistamiseks
void drawRainIcon(int x, int y){//Argumentideks ikooni x ja y alguspunktid
  drawCloud(x,y);
  //Järgnevalt joonistatakse vihmapiisad pilve alla.
  tft.fillRect(x+15,y+60,3,15,WHITE);
  tft.fillRect(x+35,y+60,3,15,WHITE);
  tft.fillRect(x+55,y+60,3,15,WHITE);
}
```

Joonis 46. Vihma ikooni loomine.

Kujundeid saab konstrueerida ka *drawBitamap()* funktsiooni kasutamata. Näiteks kuu ikooni tegemiseks (vt joonis 47), mida kasutatakse ilmateataja öörežiimis, joonistatakse ekraanile kaks ringi. Üks valge ring ja teine sama värvi nagu taustavärv. Ringid sätitakse nii, et moodustuks poolkuu.

```
//Funktsioon kuu ikooni joonistamiseks
void drawMoon(int x, int y){//Argumentideks ikooni x ja y alguspunktid
  tft.fillCircle(x,y,25, WHITE);//Valge ringi joonistamine
  tft.fillCircle(x+20,y,25,  BLACK);//Taustavärvi  ringi  joonistamine,  mis  katab  pool
hallist ringist. Seeläbi moodustub poolkuu ikoon.
}
```

Joonis 47. Kuu ikooni loomine.

Iga ikooni jaoks baidimassiivi loomine ja *drawBitmap()* funktsiooniga väljakutsumine on Arduino jaoks mälunõudev, kuna Arduino Uno mälu suurus on vaid 32 kilobaiti. Kui kasutusel on suurema mäluga mikrokontrollerplatvorm saaks kõiki ikoone baidimassiivi abil kujutada.

4.8 Nupu loomine ja vajutuse tuvastamine ekraanil

Ilmteatajas kuvatakse infot rohkem kui ühele ekraanivaatele ära mahub. Projektis kasutatav ekraan on vajutustundlik ja see annab võimaluse kasutada erinevaid ekraanivaateid, mille vahel infot jaotada. Ilmateataja ekraanil kuvatakse ühel ekraanivaatel praegused ilmaolud, teisel järgnevate tundide ilmaolusid ning kolmandal sisetemperatuuri ja õhuniiskust.

Ilmavaadete vahel liikumiseks on tehtud ekraanile nupp. Nupu loomiseks joonistatakse ristkülik ekraani alumisse nurka (vt joonis 1), kasutades *Adafruit_GFX* teegi funktsiooni *fillRect()*. Ristküliku sisse kuvatakse nool funktsiooni *print()* abil.

Nupuvajutuse tuvastamiseks on programmis tehtud funktsioon *getTouch()*. Funktsioon kutsutakse välja *loop()* funktsioonis, ehk tsükliliselt kontrollitakse kogu aeg, kas toimub nupuvajutus. Järgnevalt selgitatakse funktsiooni sisu (vt joonis 48). Kõigepealt salvestatakse vajutuse pikkus-, laius-ja kõrguskoordinaat punkti *t*. Seejärel kontrollitakse kõrguskoordinaadi võrdlemisel, kas on toimunud vajutus ekraanil. Kui vajutus on toimunud muudetakse ekraani kalibreerimisel saadud koordinaadid vastavaks ekraani mõõtmetega *map()* funktsiooni abil. Järgmiseks kontrollitakse puudutuse toimumist nupu sees. Selleks võrreldakse, kas x ja y puutekoordinaadid on risküliku piires. Nupuvajutuse tuvastamisel vahetatakse ekraanivaadet.

```
//Funktsioon, mis tuvastab nupuvajutuse ekraanil ja seejärel vahetab ekraanivaadet.
//Mõjutusi saanud Touchsceeen teegi näidisvisandist.
bool getTouch(){
 TSPoint t = ts.getPoint(); //Puutepunkti pikkuse, laiuse ja kõrguse koordinaadid
salvestatakse punkti t.
 //Viikude seadistamine väljundiks.
 pinMode(XM, OUTPUT);
 pinMode(YP, OUTPUT);
 if (t.z > ts.pressureThreshhold) { //Kui vajutus ekraanil on toimunud, hakatakse
koordinaate võrdlema nupu asukohaga.
  //Koordinaatide teisendamine ekraani mõõtmetele vastavateks koordinaatideks map()
funktsiooni abil.
  t.x = 240 - map(t.x, TS_MAXX, TS_MINX, 0,240);
  t.y = map(t.y, TS_MAXY, TS_MINY, 0,320);
  if(t.x>195 && t.x<240 && t.y>306 && t.y<326)) { //Kontroll, kas puudutus on toimunud
nupu sees.
   if(currentWeather_screen){//Kui hetkel oli ees järgnevate tundide ilmaandmete
ekraanivaade, kuvatakse pärast nupuvajutust hetkeilmaandmed.
     drawCurrentData();
     currentWeather_screen=false;
     forecast_screen=true;
     }
   else if(forecast_secreen){//Kui hetkel oli ees hetkeilmaandmete ekraanivaade,
kuvatakse pärast nupuvajutust järgnevate tundide ilmaandmed.
     forecastScreen();
     currentWeather_screen=true;
     forecast_condition=false;
     }
     return true;
  }
  else if (t.x > 10 && t.x < 240 && t.y > 10 && t.y < 150) { //Kui vajutus toimub
ekraani ülemise kasti sees kuvatakse toatemperatuuri ja õhuniiskust.
  . . .
  . . .
  //kood jätkub samal põhimõttel nagu eelnevas if lauses.
 }   return false; }
```

Joonis 48. Vajutuse tuvastamise funktsioon.

Vajutusetundliku ekraaniga suhtlemiseks pole alati selleks vaja nuppu luua. Ka ilmateataja puhul kuvatava toatemperatuuri ja õhuniiskuse vaate avamiseks eraldi nuppu ei loodud vaid vaade avaneb, kui vajutada kolmeks jaotatud ekraani ülemise kasti sisse (vt joonis 1). Vajutuse tuvastamise koordinaatide kontrolli on näha joonisel 48.

Sellega on programmeerimiseks vajalikud funktsioonid teegist *Adafruit_GFX* selgitatud. Lisaks kirjeldatud funktsioonidele saab infot teiste teegis olevate funktsioonide kohta vaadates teegi kaustas olevat *Adafruit_GFX.h* faili.

4.9 Käeviibutuse tuvastamine ultraheli kaugusanduriga

Ekraani puhkerežiimist väljatoomiseks tuvastatakse ultraheli kaugusanduriga objekti lähenemine seadme ees, mille tagajärjel lülitatakse ekraan sisse. Selleks muudetakse kõigepealt ultraheli kaugusanduri trig viigu pinge kõrgeks funktsiooniga *digitalWrite(trig, HIGH)* ja seejärel madalaks funktsiooniga *digitalWrite(trig, LOW)*. Selle tagajärjel saadab andur välja heliimpulsi. Aeg mikrosekundites, mis kulus signaali objektilt tagasi andurini jõudmiseks, tuvastatakse funktsiooniga *pulseIn(echo, LOW)*. Saadud ajavahemiku põhjal arvutatakse kaugus, mille alusel tuvastatakse, kas ekraan sisse lülitada.

4.10 Temperatuuri ja õhuniiskuse lugemine

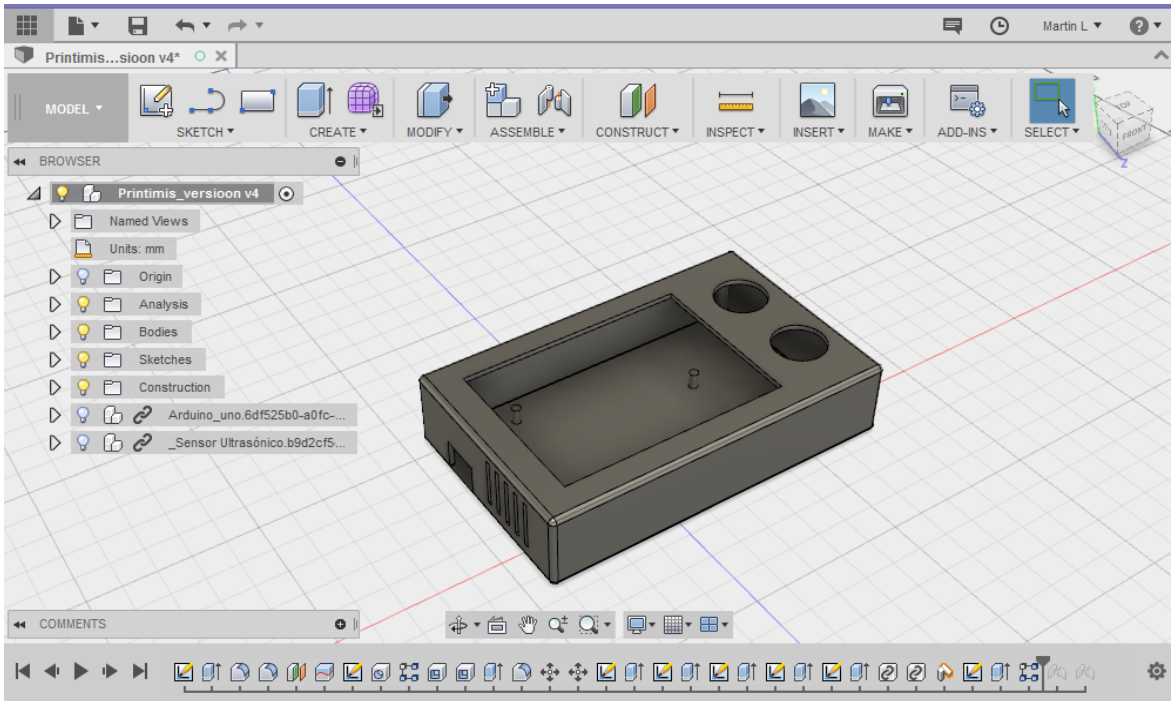
Temperatuuri ja õhuniiskuse andurilt info lugemiseks on kõige lihtsam viis kastutada teeki *dht*. Andurilt tuleva signaali lugemiseks on funktsiooni *read11(dht_pin)*. Temperatuuri lugemiseks saadud signaalist on funktsioon *temperature()* ja õhuniiskuse lugemiseks *humidity()*. Saadud väärtused loetakse ja kuvatakse ilmateataja ekraanile, kui toimub seda esilekutsuv ekraanivajutus.

4.11 Korpuse disain ja tulemused

Järgnevates alapeatükkides antakse ülevaade, kuidas disainiti ja valmistati ilmateatajale korpus ning tutvustatakse valminud ilmateataja lõpptulemusest.

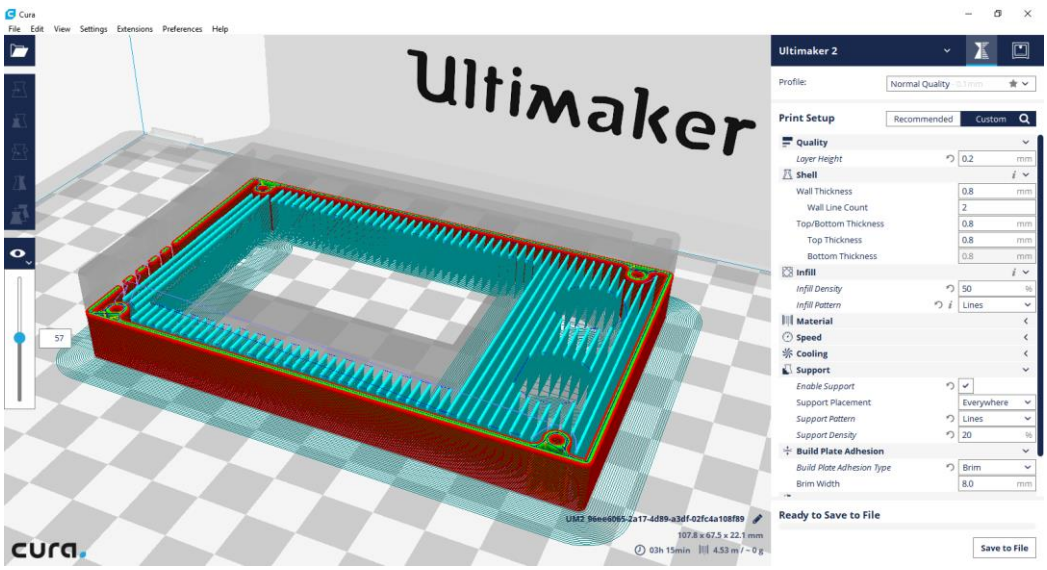
4.11.1 Ilmateataja korpus

Ilmateatajale kompaktse väljanägemise andmiseks disainis töö autor korpuse. Korpuse loomiseks disainiti tarkvaraga Autodesk Fusion 360 korpuse 3D mudel. Autodesk Fusion 360 on uus pilvepõhine disainimise tarkvara [33]. Tarkvara on loodud Windows ja MAC OS operatsioonisüsteemidele nii, et samu disaine saab muuta erinevates operatsioonisüsteemides [34]. Lisaks võimaldab pilves hoidmine lihtsalt jagada loodud disaine ja kutsuda inimesi oma projekti muudatusi tegema ning tarkvara on tasuta õpilastele, hobikasutajatele ja idufirmadele [34]. Joonisel 49 on näha, milline näeb välja keskkonnas disainitud ilmjaama korpus, mis koosneb põhjast ja kaanest.



Joonis 49. Tarkvaraga Autodesk Fusion 360 disainitud korpuse 3D mudel.

Mudel prinditi välja 3D printeriga Ultimaker, mis põhineb kiht kihil printimise tehnoloogial. Põhi ja kaas prinditi välja eraldi. Printeri jaoks sobilikud tööfailid valmistati kasutades Cura tarkvara. Joonisel 50 on näha printimiseks ettevalmistatud mudelit Cura tarkvaras, kus salvestatakse printeri tööfail.

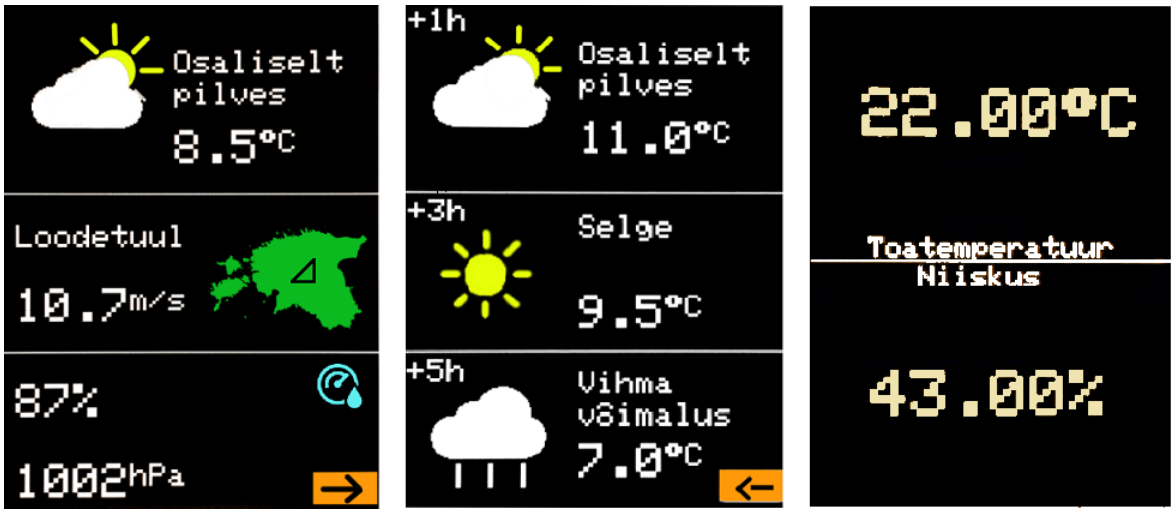


Joonis 50. Cura tarkvaraga printeri tööfaili tegemine.

Lisainfot 3D printimise kohta võib lugeda Alo Peetsi artiklist aadressil http://etu.ut.ee/2017/3d-printimine/.

4.11.2 Tulemused

Töö käigus ehitati Arduino ilmateataja, mis koosneb Arduinost, 2,8 tollisest vajutustundlikust ekraanist, WiFi moodulist ja kaugusandurist. Ilmateataja pärib andmed ilmaportaali Weather Underground rakendusliidese kaudu ja pärast andmete töötlust kuvab need vajutustundlikule ekraanile. Lisafunktsionaalsusena lisati tööle ka andur, mis mõõdab sisetemperatuuri ja õhuniiskust. Kokku on võimalus liikuda kolme ekraanivaate vahel, kus üks näitab praegusi ilmaolusid, teine ilmaolusid järgevatel tundidel ja kolmas sisetemperatuuri ning õhuniiskust. Valminud ilmateataja ekraanivaated on esitatud joonisel 51. Seadme puhkerežiimis lülitatakse ekraan välja, mille lõpetamiseks tuleb viibata käega seadme ees.



Joonis 51. Valminud ilmajaama ekraanivaated.

Ilmateataja parema väljanägemise eesmärgil loodi ka selle ümber 3D prinditud korpus. Väljaprinditud korpusega Arduino ilmateatajat on näha joonisel 52.

Joonis 52. Korpusega ilmateataja.

Töö tegemiseks vajalik riistvara maksumus püüti hoida võimalikult madalal. Välismaalt riistvara tellides maksab seade ligi kaks korda vähem kui Eestist ostes. Valmistatud ilmateataja kogumaksumus lõputöö tegemise ajal olevate hindade juures on Eestist riistvara ostes 54 eurot ja välismaalt tellides 26 eurot. Vastav riistvara hindasid kirjeldav tabelit on näha lisas 1.

Antud peatükis kirjeldati, kuidas käis ilmateataja programmeerimine. Lisaks näidati, kuidas ilmateataja lõppviimistlemiseks disainiti korpus ja milline oli valminud seadme lõpptulemus. Eelnevate peatükkidega on vajalikud sammud ilmateataja ehitamiseks kirjeldatud, mille käigus tutvustati vajutustundliku ekraani kasutamist Arduinoga ja Arduinole juhtmevaba internetiühenduse loomist.

Kokkuvõte

Käesoleva bakalaureusetöö eesmärgiks oli luua Arduino ilmateataja, mis kuvab vajutustundlikule ekraanile ilmaportaalist saadavaid ilmastikuandmeid. Ilmateataja ehitamist oli eesmärk dokumenteerida viisil, et selle põhjal valmiks eestikeelne õppematerjale Arduinoga ühilduva vajutustundliku ekraani kasutamise ja juhtmevaba internetiühenduse loomise kohta. Lisaks oli eesmärk bakalaureusetöös tehtav projekt koostada selliselt, et selle järeletegemisel arendaksid noored programmeerimisoskust ja teadmisi riistvara käsitlemisest.

Lõputöö käigus valmis nimetatud ilmateataja, mis põhineb Arduinol ning sellega ühendatavatest moodulitest ja anduritest. Ilmateataja teeb WiFi mooduli abil ilmastikuandmete päringu Weather Underground ilmaportaalist ja kuvab need andmed 2,8 tollisele vajutustundlikule ekraanile. Samuti lisati ilmateatajale andur, mis kuvab infot toatemperatuuri ja õhuniiskuse kohta. Arduino ilmateataja ehitamise käiku dokumenteeriti nõnda, et see sobiks õppematerjaliks, mõistmaks, kuidas kasutada Arduinoga vajutustundliku ekraani ning luua arendusplaadiga juhtmevaba internetiühendus. Lisaks valmistati ilmateatajale 3D prinditud korpus. Lõputöö põhjal on võimalik luua töötubasid, kus ehitatakse sarnane Arduino ilmateataja. Samuti saab töö põhjal seadme iseseisvalt järgi teha, mis arendab oskusi nii programmeerimises kui ka teadmisi, kuidas riistvara omavahel suhtlema panna. Veel sobib valminud õppematerjal mitte vaid ilmateataja ehitamiseks vaid ka teiste sarnaste projektide aluseks.

Enne bakaulaureusetöö kirjutamist puudus töö autoril varasem kokkupuude mikrokontrolleritega ja riistvara programmeerimisega, mistõttu oli ilmateataja projektiga alustamine väljakutsuv. Autor mõistis töö käigus, et riistvaralähedane programmeerimine nõuab lahenduseni jõudmiseks mitmeid kordi katsetamist ja otsast peale hakkamist. Samas aitas varasema kokkupuute puudumine kaasa õppematerjali koostamisele, kuna autor nägi enda kogemuse läbi, millised on just algajale rasked kohad. Töö käigus lõi autor ideest lahenduse, omandades selle käigus oskusi süsteemi analüüsist, disainimisest ja ehitamisest. Autor sai uusi teadmisi riistvaralähedasest programmeerimisest ning oskusi akadeemilise teksti vormistamisest.

Viidatud kirjandus

[1] Richard Robinson, „How Accurate Are Weather Forecasts?“, http://www.huffingtonpost.com/richard-robbins/how-accurate-are-weather-_b_6558770.html (12.03.2017)

[2] Arduino, „What is Arduino?“, https://www.arduino.cc/en/Guide/Introduction (19.12.2016)

[3] James Bruce, „What Is Arduino & What Can You Do With It? [Technology Explained]“, http://www.makeuseof.com/tag/arduino-technology-explained/ (7.01.2017)

[4] Arduino, „Language Reference“, https://www.arduino.cc/en/Reference/HomePage (7.01.2017)

[5] Arduino, „Arduino UNO & Genuino UNO“, https://www.arduino.cc/en/main/arduinoBoardUno (19.02.2016)

[6] Google, „Google Trends“, https://trends.google.com/trends/explore?q=Arduino%20uno,Arduino%20mega,Arduino%20Leonardo. (20.02.2017)

[7] Sparkfun, „What is an Arduino?“, https://learn.sparkfun.com/tutorials/what-is-an-arduino (7.02.2017)

[8] Arduino, „Software“, https://www.arduino.cc/en/Main/software (15.02.2017)

[9] Arduino, „Development Tools“, http://playground.arduino.cc/Main/DevelopmentTools (20.02.2017)

[10] Wiring, „What will YOU do with the W?“ http://wiring.org.co/ (15.02.1017)

[11] James Bruce, „First Steps With The Arduino: A Closer Look At The Circuit Board & The Structure Of A Program“, http://www.makeuseof.com/tag/steps-arduino-closer-circuit-board-structure-program/ (6.01.2017)

[12] Arduino, „setup()“, https://www.arduino.cc/en/reference/setup (15.03.1017)

[13] Arduino, „loop()“, https://www.arduino.cc/en/reference/loop (15.03.2017)

[14] Veebipood dhgate, „2.8 Inch TFT LCD Display“, http://www.dhgate.com/store/product/high-quality-1pcs-2-8-inch-tft-lcd-display/371072685.html_ (20.04.2017)

[15] Cyan | Infinite, „2.8“TFT LCD Touchscreen Shield“, http://cyaninfinite.com/tutorials/2-8-tft-lcd-touchscreen-shield/ (17.02.2017)

[16] Tech, „How do touch-screen monitors know where you're touching? http://computer.howstuffworks.com/question716.htm. (10.04.2017)

[17] Google, „Google Trends“, https://trends.google.com/trends/explore?date=2014-08-12%202017-05-11&q=Arduino%20uno,Arduino%20mega,Arduino%20Leonardo. (20.04.2017)

[18] Sparkfun, „WiFi Module – ESP8266“, https://www.sparkfun.com/products/13678 (27.02.2017)

[19] Veebipood Banggood, „Esp8266 ESP-01“, https://www.banggood.com/Upgraded-Version-1M-Flash-ESP8266-ESP-01-WIFI-Transceiver-Wireless-Module-p-979509.html?rmmds=search (27.02.2017)

[20] Esp8266 kommuun, „Modules“, http://www.esp8266.com/wiki/doku.php?id=esp8266-module-family (27.02.2017)

[21] Arduino Forum, „Connect to ESP8266 ONLY using Arduino Uno“, https://forum.arduino.cc/index.php?topic=283043.0 (14.04.2017)

[22] Room-15 „ESP8266 – AT Command Reference“ https://room-15.github.io/blog/2015/03/26/esp8266-at-command-reference/. (28.02.2017)

[23] Cytron Technologies, „HC-SR04 User's_Manual, https://docs.google.com/document/d/1Y-yZnNhMYy7rwhAgyL_pfa39RsB-x2qR4vP8saG73rE/edit (14.03.2017)

[24] Aosong, „DHT11 Digital Temperature And Humidity Sensor“, http://www.aosong.com/en/products/details.asp?id=109. (20.03.2017)

[25] M.Schwartz, „Monitor Your Home With the Raspberry Pi B+“ https://learn.adafruit.com/monitor-your-home-with-the-raspberry-pi-b-plus?view=all. (20.03.2017)

[26] Core Electronics, „DHT11 Temperature and Relative Humidity Senosror Module“, https://core-electronics.com.au/dht11-temperature-and-relative-humidity-sensor-module.html. (20.03.2017)

[27] Weather Underground, „About Our Data“,https://www.wunderground.com/about/data (17.01.2017)

[28] Weather Underground, „Pricing“, https://www.wunderground.com/weather/api/d/pricing.html (10.03.2017)

[29] AccuWeather, „AccuWeather API“, http://apidev.accuweather.com/developers/ (10.3.2017)

[30] Weather Underground, „API Home“, https://www.wunderground.com/weather/api (17.01.2017)

[31] Blynk Community, „ESP8266 connection on Arduino RX / TX Ports“, https://community.blynk.cc/uploads/default/original/2X/c/c1a381ac9adb1809eb667cce7e91871d31575558.png (15.04.2017)

[32] Nick Koumaris, „Arduino Bitmap Graphics Tutorial“, http://educ8s.tv/arduino-bitmap-graphics-tutorial.(20.03.2017)

[33] Alo Peets, „10 aastat hobi – 3D printimist“, http://etu.ut.ee/2017/3d-printimine/. (8.05.2017)

[34] Autodesk, „Overview“, https://www.autodesk.com/products/fusion-360/overview. (8.05.2017)

Lisad

I. Riistvara komponentide hinnad

Riistvara	Hind ostes Eestist	Hind tellides välismaalt
Arduino Uno	18€ http://www.ittgroup.ee/et/arduino-arendusplaadid/85-arduino-uno.html?search_query=arduino+uno&results=39	9,81€ http://www.ebay.com/itm/Best-UNO-R3-ATmega328P-Development-Board-For-Arduino-OriGinal-New-Version-2017-/272309975197?hash=item3f66f0789d:g:-ucAAOSwKtlWlSZ7
2,8 tolline ekraan	18€ http://www.ittgroup.ee/et/arduino-laiendusplaadid/960-puutetundlik-28-lcd-laiendusplaat-arduinole.html	9,72€ https://www.banggood.com/2_8-Inch-TFT-LCD-Shield-Touch-Display-Module-For-Arduino-UNO-p-989697.html?rmmds=search
ESP8266-ESP01	9€ https://www.oomipood.ee/product/esp8266_module_esp8266_wifi_serial_port_moodul?q=esp8266	2,89€ https://www.banggood.com/Upgraded-Version-1M-Flash-ESP8266-ESP-01-WIFI-Transceiver-Wireless-Module-p-979509.html?rmmds=search
HC-SR04	2,6€ http://www.ittgroup.ee/et/andurid-ja-nende-tarvikud/256-ultraheliandur-hc-sr04.html?search_query=hc-sr04&results=7	1,75€ https://www.banggood.com/Wholesale-Ultrasonic-Module-HC-SR04-Distance-Measuring-Ranging-Transducer-Sensor-p-40313.html?rmmds=search
DHT11	6€ https://www.oomipood.ee/product/dht11_sens_dht11_niiskusandur?q=dht	2€ https://www.banggood.com/KY-015-DHT11-Temperature-Humidity-Sensor-Module-For-Arduino-p-916173.html?rmmds=search

II. Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, **Martin Johannes Liba**,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose
   Arduino ilmateataja,
   mille juhendaja on Alo Peets, Anne Villems, Taavi Duvin.
   1.1. reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace-is lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
   1.2. üldsusele kättesaadavaks tegemiseks Tartu Ülikooli veebikeskkonna kaudu, sealhulgas digitaalarhiivi DSpace´i kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.
3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tartus, **11.05.2017**