TARTU ÜLIKOOL
MATEMAATIKA-INFORMAATIKATEADUSKOND
Arvutiteaduse instituut
Infotehnoloogia eriala

Johannes Ait

Visuaalse programmeerimise mooduli lisamine NXCEesti rakendusele

Bakalaureusetöö (6 EAP)

Juhendajad: Taavi Duvin
Anne Villems

TARTU 2015

Visuaalse programmeerimise mooduli lisamine NXCEesti rakendusele

Lühikokkuvõte:

NXCEesti on 2012. aastal loodud platvormiülene NXC kasutamise rakendus, millega saab luua programme LEGO Mindstorms NXT robotitele. Käesoleva töö raames lisati NXCEesti rakendusele visuaalse programmeerimise moodul, mis võimaldab LEGO Mindstorms NXT robotitele programme luua graafiliste plokkide ühendamise kauda. Töö koosneb kolmest osast. Esimeses osas antakse ülevaade visuaalsest programmeerimisest ja selle ajaloost. Teises osas tutvustatakse olemasolevat rakendust NXCEesti. Kolmandas peatükis kirjeldatakse töö käigus NXCEesti rakendusele tehtud muudatusi.

Võtmesõnad:

robootika, LEGO Mindstorms NXT, NXC, NXCEesti, visuaalne programmeerimine

Adding a visual programming module to NXCEesti

Abstract:

NXCEesti is a cross-platform environment for programming LEGO Mindstorms NXT robots using the NXC language. The aim of this bachelor's thesis is to add a visual programming module to the existing NXCEesti application. The paper consists of three parts. The first part gives an overview of visual programming and it's history. The second part introduces the existing NXCEesti application. The third part describes the additions made to the NXCEesti application in the course of the current thesis.

Keywords:

robotics, LEGO Mindstorms NXT, NXC, NXCEesti, visual programming

Sisukord

1. Sissejuhatus

Tänapäeva infoühiskonnas on reaalained muutumas järjest tähtsamaks. Samas on noorte vähene huvi reaalainete vastu Eestis juba pikka aega probleemiks olnud. Põhikooli ja gümnaasiumi õpilaste seas peetakse reaalaineid tihti igavaks ja liiga teoreetiliseks. Õpilased ei näe seost näiteks matemaatika ja päris elu vahel. Üheks võimaluseks reaalaineid noorte jaoks huvitavamaks muuta oleks robootika kaasamine õppekavasse.

Selleks loodigi 2007. aastal Eestis Kooliroboti projekt, mille eesmärgiks on äratada õpilastes huvi inseneriteaduste vastu. Projekti raames jagatakse koolidele õppematerjale ning viiakse läbi õpetajakoolitusi. Õppevahendina kasutatakse firma LEGO poolt väljatöötatud robootikakomplekti Mindstorms NXT. Põhikomplekti kuulub NXT programmeeritav juhtplokk, neli andurit, kolm mootorit, ühenduskaablid ja legoklotsid.

LEGO Mindstorms NXT robotite programmeerimiseks on olemas mitmeid võimalusi. Algajatele on sobivaim kasutada LEGO Mindstorms NXT komplektiga kaasasolevat tarkvara NXT-G, mis võimaldab hõlpsalt graafilises keskkonnas robotiprogramme koostada. Edasijõudnumad programmeerijad kasutavad aga enamasti tekstipõhiseid programmeerimiskeeli, kuna keerulisemate programmide loomine graafilises keskkonnas muutub kiiresti tülikaks. Tekstipõhistest programmeerimiskeeltest on üks populaarsemaid NXC (Not eXactly C).

Alles hiljuti puudus NXC keele jaoks arenduskeskkond, mis töötaks ühtmoodi kõigil levinud operatsioonisüsteemidel. Erinevatele operatsioonisüsteemidele olid NXC keeles programmeerimiseks erinevad rakendused ning see muutis keeruliseks universaalse NXC keele kasutusjuhendi loomise. Selle probleemi lahendamiseks arendas Tartu Ülikooli tudeng Priit Rand oma 2012. aasta magistritöö raames välja platvormiülese NXC arenduskeskkonna NXC Eesti. Rakendus põhineb Java programmeerimiskeelel ning on seega kasutatav nii Windows, Mac OS kui ka Linux operatsioonisüsteemidel.

Uue programmeerimiskeele selgeks saamine on aeganõudev ettevõtmine, eriti kui varasem kokkupuude programmeerimisega puudub. Viimastel aastatel on populaarsust kogunud programmeerimise õpetamine visuaalsete programmeerimiskeelte abil, kus programmi elemendid on esitatud graafiliste plokkidena ning programmeerija ülesandeks on nende plokkide muutmine ja omavahel kokkusobitamine. Visuaalne keel on algaja programmeerija jaoks kergesti arusaadav ja ei eelda keerulise süntaksi selgeks õppimist. Käesoleva bakalaureusetöö eesmärgiks on lisada NXC Eesti rakendusele visuaalse programmeerimise moodul, mis võimaldaks luua programme LEGO Mindstorms NXT robotite jaoks.

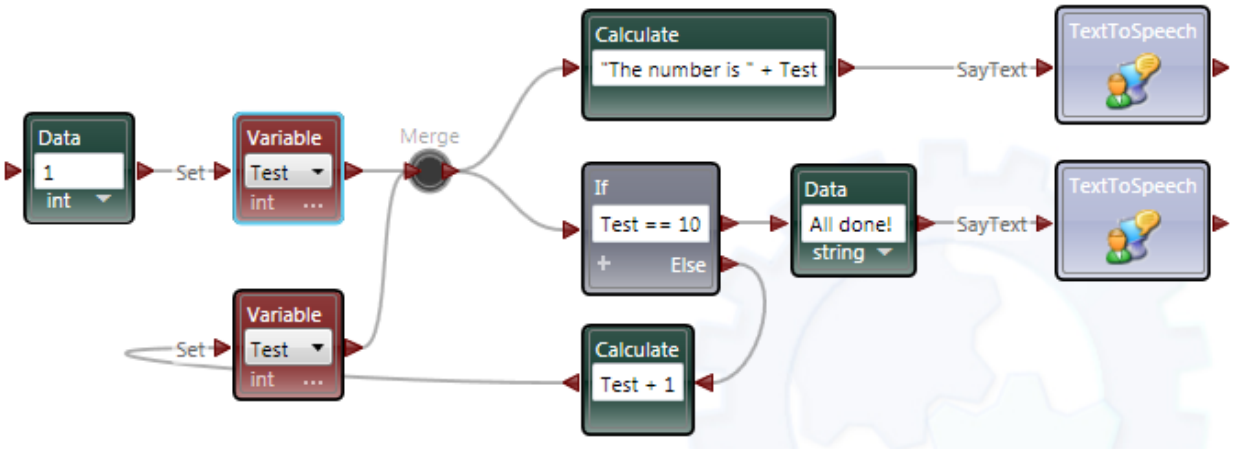
Töö koosneb kolmest peatükist. Esimeses peatükis antakse ülevaade visuaalsest programmeerimisest ja selle ajaloost ning tutvustatakse mõningaid visuaalse programmeerimise keskkondi. Teises peatükis antakse ülevaade NXC keelest ja rakendusest NXCEesti. Kolmandas peatükis kirjeldatakse käesoleva bakalaureusetöö käigus NXCEesti rakendusele tehtud muudatusi.

2. Ülevaade visuaalsest programmeerimisest

Tänapäeva maailmas on arvutid kõikjal meie ümber ning nõudlus inimeste järgi, kes arvuteid programmeerida oskaksid, aina kasvab. Programmeerijate arvu suurendamiseks tuleks koolinoortes juba varakult reaalainete ja programmeerimise vastu huvi tekitada. Üheks võimaluseks reaalainete õppimist huvitavamaks muuta on robootika õpetamine koolis ja huviringides. Roboti programmeerimiseks tuleb aga esmalt selgeks saada vastav programmeerimiskeel. Enamik programmeerimiskeeltes käib programmi loomine teksti sisestamise kaudu. Tekstipõhised programmeerimiskeeled kasutavad tavaliselt rangeid keelereegleid, mille vastu on algajal kerge eksida ning mille selgeks saamiseks võib palju aega kuluda. Programmeerimise lihtsamaks ja intuitiivsemaks muutmise eesmärgil on algajatele programmeerimise õpetamiseks hakatud kasutama visuaalseid programmeerimiskeeli.

Visuaalne programmeerimine erineb traditsioonilistest tekstipõhistest programmeerimisest selle poolest, et programmi väljendamiseks kasutatakse teksti asemel graafilisi objekte [1]. Paljudes visuaalse programmeerimise keskkondades on programm esitatud suunatud graafina, kus graafi tippudeks on mingid funktsioonid või protsessid ja graafi kaared näitavad andmete liikumist nende vahel. Sellist esitusviisi kasutab näiteks robotite programmeerimiseks mõeldud arenduskeskkond Microsoft Visual Programming Language (joonis 1) [2].

Samuti on populaarsed sellised visuaalse programmeerimise keskkonnad, kus programmi elemendid on esitatud graafiliste plokkidena, mida saab omavahel ühendada nagu pusletükke. Taolisi programmeerimiskeeli nimetatakse plokk-keelteks (*blocks languages*) [4]. Viimastel aastatel on plokk-keeli hakatud kasutama üha rohkem algajatele programmeerimise õpetamiseks. Üks populaarsemaid plokk-keeli, mida kasutatakse nii õpilastele kui ka õpetajatele programmeerimise õpetamiseks, on Scratch [6].



Joonis 1: Suunatud graafina esitatud programm Microsoft Visual Programming Language keskkonnas [3]

Plokk-keelte kasutuselevõtmisel programmeerimise õppevahendina on mitmeid põhjuseid. Erinevalt tekstipõhistest keeltest puudub plokk-keeltes vajadus keerulisi süntaksireegleid selgeks õppida. Plokkide omavahel ühendamine on lihtne ja intuitiivne ning algajal ei kulu palju aega oma esimese programmi tööle saamiseks. Plokkide kuju annab programmeerijale aimu, millised plokid omavahel ühilduvad ning mittesobituvaid plokke pole lihtsalt võimalik kokku panna, mistõttu on programmeerijal ka üsna raske mittetöötavat programmi luua. Näiteks võib arvutüüpi muutujat esitavatel plokkidel olla küljes ümmarguse kujuga pistik ning funktsiooniplokil, mis võtab parameetriks arvu, võib olla vastava kujuga pesa. Kui kasutaja proovib sinna sisse panna näiteks tekstitüüpi muutuja, siis ei lase arenduskeskkond tal seda teha.

Peale keeruliste süntaksireeglite selgeks saamise on algajatele suureks probleemiks ka programmeerimiskeele käskude meelde jätmine. Selle probleemi lahendamiseks kasutatakse plokk-keeltes käskude kollektsioone ehk sahtleid, kus käsud on kategoriseeritud nende funktsioonide järgi. Näiteks võivad ühes sahtlis olla loogikaga seotud käsuplokid ja teises sahtlis roboti sensoritega seotud käsuplokid. Seega peab programmeerija lihtsalt avama õige sahtli ning sealt vajaliku käsuploki üles leidma. Sarnase funktsiooniga käsuplokid on tavaliselt esitatud ka sama värviga, mis aitab neid üksteisest visuaalselt eristada.

Tänu nendele ja mitmetele teistele põhjustele on plokk-keeled muutunud populaarseks vahendiks koolinoortele programmeerimise tutvustamisel. Algajad programmeerijad saavad keskenduda loogika ja probleemi lahendamise oskuste arendamisele ning ei pea oma pead vaevama keerulise süntaksi ja käskude päheõppimisega. Samuti ei saa mainimata jätta, et graafiliste plokkide ühendamine on just noorema kooliastme lastele kindlasti lõbusam tegevus kui programmikoodi kirjutamine.

Kuigi plokk-keeled on kergesti õpitavad ja nende abil on võimalik kiiresti töötav programm koostada, siis keerukamate programmide loomisel võib plokkide hiirega vedamine muutuda liiga aeganõudvaks ja tüütuks. Veidi kogenuma programmeerija jaoks oleks koodi trükkimine kindlasti kiirem ja efektiivsem viis programmi loomiseks. Isegi lihtsa matemaatilise valemi väljendamiseks, mis oleks teksti kujul esitatav ühe rea koodiga, võib plokk-keeltes kuluda palju plokke. Samuti võtavad keerukamad programmid ekraanil palju ruumi ja võivad lõpuks muutuda hoomamatuks.

Järgnevates punktides antakse ülevaate hariduses ja robootikas kasutatavate visuaalsete programmeerimiskeelte ajaloost ning tutvustatakse LEGO Mindstorms NXT robotite programmeerimiseks kasutatavat visuaalprogrammeerimise keskkonda NXT-G.

2.1. Plokk-keelte ajalugu

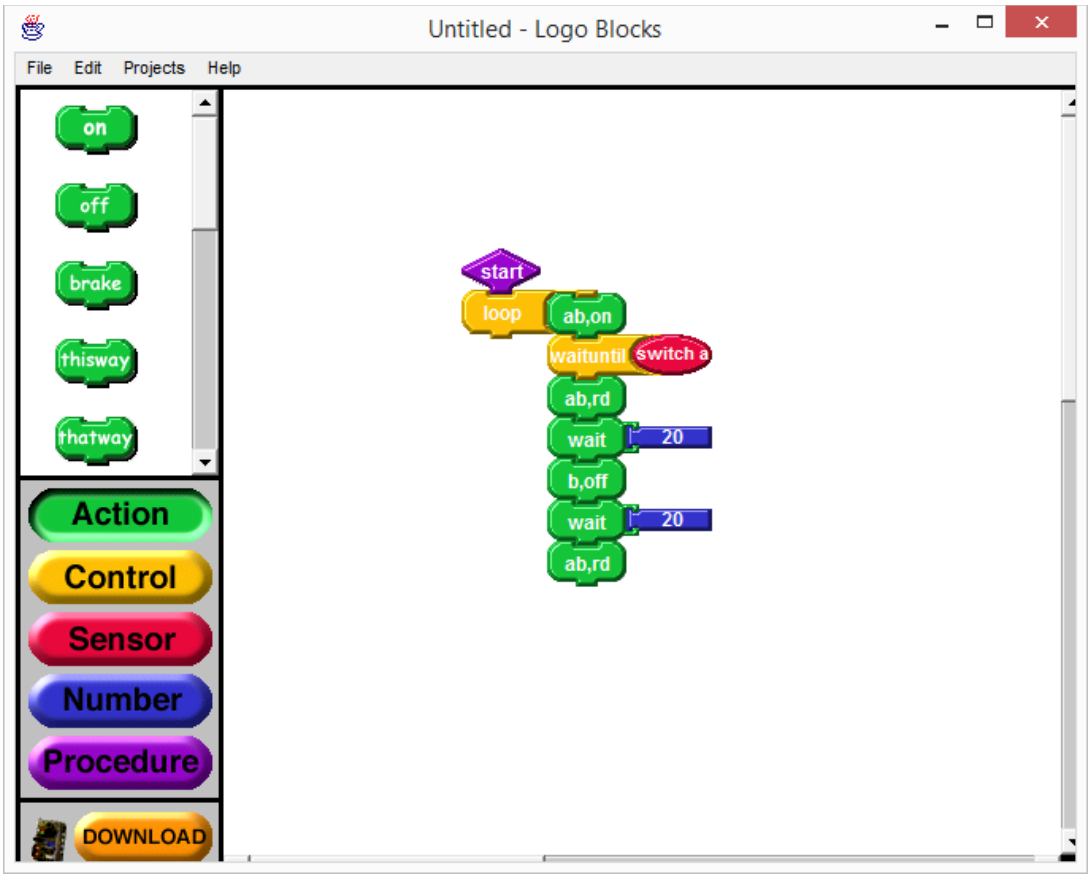
2.1.1. LogoBlocks

Üks esimestest plokk-keeltest on LogoBlocks, mis loodi aastal 1996 Massachusettsi Tehnoloogiainstituudis [5]. LogoBlocks põhineb 1967. aastal loodud hariduslikul programmeerimiskeelel Logo [6]. Logo kasutas lihtsat süntaksit ning selle eesmärgiks oli muuta programmeerimise õppimine piisavalt jõukohaseks nii täiskasvanutele kui ka lastele. Siiski oli tegemist tekstipõhise keelega, mis tähendas keele süntaksi ja käskude nimede selgeks õppimist.

Selle probleemi lahendamiseks võttis LogoBlocks kasutusele visuaalse programmeerimise keskkonna, kus programmi käsud olid esitatud graafiliste plokkidena ning programmi loomine käis omavahel kokku sobivate plokkide ühendamise kaudu. Süntaksi ja käskude nimede meelde jätmise asemel sai kasutaja nüüd vajalikud käsud õigest sahtlist üles otsida ja hiirega õigesse

kohta vedaga. LogoBlocks võimaldas luua programme LEGO RCX robotile, millest hiljem arenes välja LEGO Mindstorms NXT.

Joonisel 2 on näidatud LogoBlocks arenduskeskkonnas valminud näidisprogramm. Joonise vasakul pool on sahtlid, kust saab valida käsuplokke nende funktsioonide järgi. Joonisel on avatud tegevustega seotud käskude sahtel, kus asuvad plokid roboti mootorite kontrollimiseks. Joonise paremal pool on näha valminud programm. Programm liigutab robotit edasi, kuni see põrkab vastu seina (puutesensor tagastab tõese väärtuse). Vastu seina põrgates liigub robot 2 sekundit tagasi, muudab oma suunda, lülitades ühe mootori 2 sekundiks välja, ning seejärel jätkab edasi liikumist.



Joonis 2: LogoBlocks arenduskeskkond

LogoBlocks'is on kasutajal küll võimalik defineerida omaloodud protsesse, kuid neile ei saa ette anda argumente ega tagastada väärtusi. Samuti on LogoBlocks keskkonnas väga piiratud arv

olemasolevaid funktsioone, näiteks puuduvad funktsioonid arvust ruutjuure leidmiseks ja arvu ümardamiseks. Tekstiga seotud funktsioonid puuduvad LogoBlocks'is üldse, kuna ainukeseks andmetüübiks on arv.

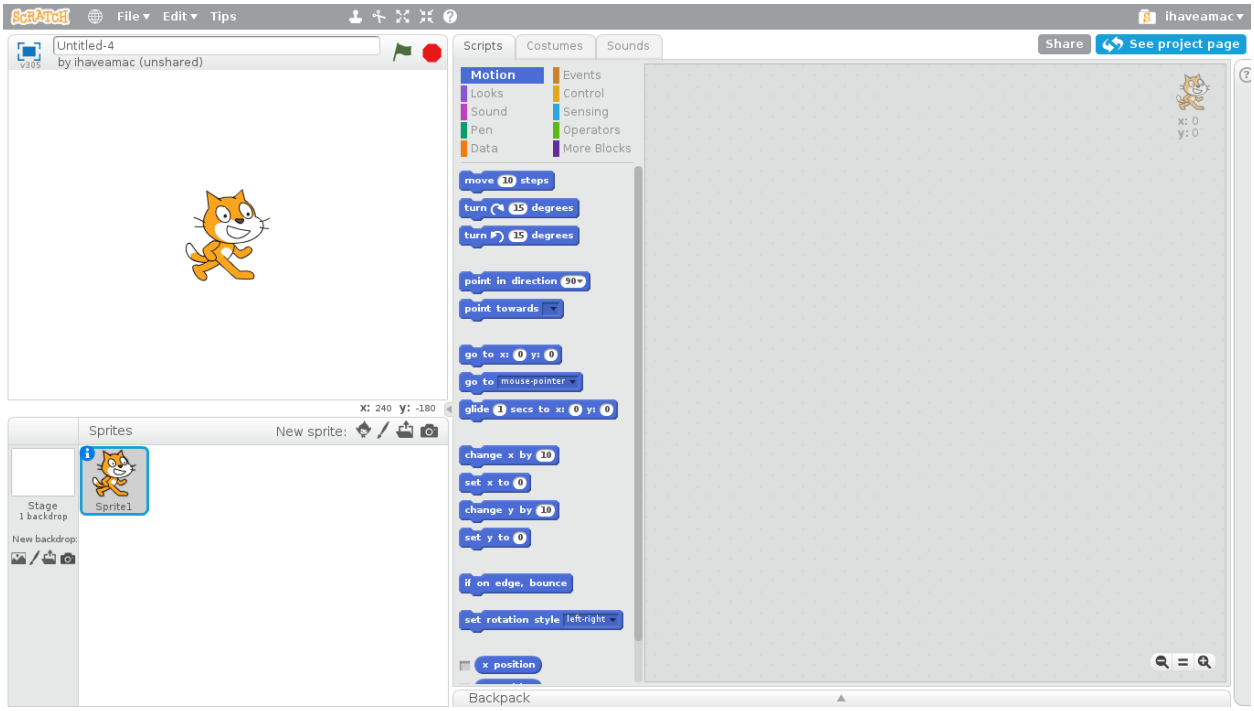
2.1.2. Scratch

LogoBlocks oli inspiratsiooniks 2007. aastal loodud visuaalse programmeerimise keskkonnale Scratch [7], mis on muutunud koolilastele programmeerimise õpetamisel üheks populaarseimaks vahendiks. Algselt oli tegemist arvutisse laetava programmiga, kuid Scratchi viimane versioon (2.0) asub täielikult kasutaja veebibrauseris. Scratch on mõeldud eeskätt 8-16 aastastele koolilastele, kuid seda on võimalik kasutada ka ülikoolides programmeerimist tutvustavate kursuste õppevahendina [8].

Scratch võimaldab kasutajatel luua interaktiivseid mänge ja animatsioone, õpetades samal ajal neile programmeerimise põhitõdesid ja arendades süstemaatilist mõtlemist. Erinevalt LogoBlocks keskkonnast ei ole Scratch'is programmeeritavaks objektiks füüsiline robot, vaid ekraanil olevad virtuaalsed objektid ehk spraidid (*sprites*). Spraitideks võivad olla näiteks tegelased mingis mängus või animatsioonis. Iga spraidiga saab siduda eraldi skripti, mis programmi käivitamisel kõik paralleelselt tööle hakkavad.

Scratch'is saab kasutaja importida enda loodud pildi- ja helifaile, mida oma projektis kasutada. See võib olla suureks motivaatoriks programmeerimise õppimisel, kuna kasutajal on võimalus luua just sellise sisuga projekte, mis talle endale huvi pakuvad. Scratch'i viimases versioonis on ka sisseehitatud pildi- ja helitöötlusvahendid, mis võimaldavad kasutajatel luua pilte ja helisid Scratch'i keskkonnas.

Joonisel 3 on esitatud Scratch'i arenduskeskkond, mis koosneb mitmest osast: Vasakul üleval on näha lava (stage), kus programmi käivitamisel esitatakse valminud mäng või animatsioon. Vasakul all on nimekiri spraitidest, kust saab valida, milline sprait hetkel aktiivne on. Keskel on näha erinevate funktsioonidega sahtlid ja neis asuvad käsuplokid. Käsuplokke saab lohistada paremal asuvasse tööalasse, kus toimub programmi loomine aktiivse spraidi jaoks.



Joonis 3: Scratch 2.0 arenduskeskkond [9]

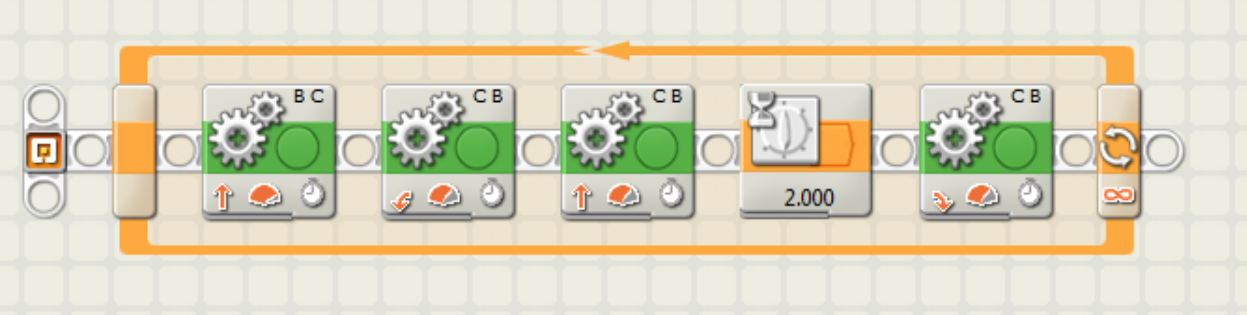
Scratch’is on suur valik olemasolevaid funktsiooniplokke ja kasutajal on võimalus ka uusi funktsiooniplokke luua. Erinevalt LogoBlocks keskkonnast saab Scratch’is funktsioonidele ka parameetreid anda, mis võimaldab luua abstraktsemat koodi.

Üks põhjus Scratch’i populaarsuse taga on Scratch’i veebileht, kus kasutajad saavad enda loodud projekte teistega jagada ja muljeid vahetada. Teiste kasutajate loodud projekte on võimalik alla laadida ja neid muuta. Käesoleva töö kirjutamise hetkel on Scratch’i kodulehel jagatud üle 9 miljoni projekti ning registreeritud kasutajaid on üle 6 miljoni [10].

2.2. NXT-G

NXT-G on LEGO Mindstorms NXT robootikakomplektiga kaasasolev ametlik arenduskeskkond ning see põhineb National Instruments LabVIEW programmil. Keskkonnas on võimalik valida erinevaid tegevuste plokke, mida saab lohistada töölauale ning omavahel juhtmetega ühendada.

Joonisel 4 on näha NXT-G keskkonnas koostatud programm, mis paneb roboti lõpututus tsüklis tegema järgmisi liigutusi: sõida viis sekundit otse; keera kaks sekundit vasakule; sõida viis sekundit otse; oota kaks sekundit; keera viis sekundit paremale.



Joonis 4: NXT-G keskkonnas loodud näidisprogramm.

Käesolevas peatükis anti ülevaade visuaalsest programmeerimisest ja tutvustati mõningaid visuaalse programmeerimise keskkondi.

3. NXC Eesti tutvustus

Käesolev peatükk annab esmalt lühikese ülevaate NXC keelest ning seejärel tutvustab Priit Ranna magistritöö raames loodud platvormiülest NXC arenduskeskkonda NXCEesti.

3.1. Ülevaade NXC keelest

Not Exactly C ehk lühendatult NXC on spetsiaalselt LEGO Mindstorms NXT robotite programmeerimiseks loodud kõrgetasemeline keel. Nimi *Not Exactly C* (“Pole Päris C”) tuleneb sellest, et NXC on süntatkiliselt sarnane C keelega, kuid erinevalt C-st ei ole NXC puhul tegemist üldotstarbelise programmeerimiskeelega. NXC põhineb madalama astme assemblerkeelel nimega *Next Byte Codes* (NBC), mida kasutatakse LEGO Mindstorms NXT juhtploki programmeerimiseks. NXC keeles kirjutatud lähtekoodi käivitamiseks tuleb see eelnevalt kompileerida NXT baitkoodiks. NXT juhtplokki on sisse ehitatud interpretaator, mis suudab tekkinud baitkoodi käivitada, et tekiks töötav programm [11].

NXC on loogiliselt jaotatud kaheks osaks. NXC keel kirjeldab programmide kirjutamiseks kasutatava süntaksi ning NXC programmeerimisliides kirjeldab programmi poolt kasutatavad süsteemifunktsioonid, konstandid ja makrod. Programmeerimisliidest kasutatakse peamiselt sisend- ja väljundseadmete haldamiseks ning programmi töö juhtimiseks. Järgnevalt on välja toodud mõned näited NXC programmeerimisliidesesse kuuluvatest funktsioonidest ja konstantidest:

- OnFwd(“mootorid”, “kiirus”) - pöörab mootoreid etteantud kiirusega
- Wait(“aeg”) - seiskab programmi töö etteantud ajaks
- OUT_A - porti A tähistav konstant
- SEC_3 - konstant, mis vastab 3000 millisekundile ehk 3 sekundile

NXC programm koosneb muutujatest ja koodiplokkidest. Koodiplokke on kahte tüüpi: tegumid (tasks) ja funktsioonid (functions). Funktsioon on mingit kindlat ülesannet täitev instruktsioonide kogum, mida saab programmi erinevatest osadest välja kutsuda. Tegumid on programmi osad, mis saavad robotis töötada paralleelsete lõimedena. Igas NXC programmis peab olema tegum

main, kust alustatakse programmi tööd. Maksimaalselt saab ühes programmis olla kokku 256 tegumit ja funktsiooni.

Joonisel 5 on esitatud kommenteeritud koodinäide lihtsast programmist, mis on kirjutatud NXC keeles. Programmi käivitudes liigub robot 4 sekundit edasi ja 4 sekundit tagasi ning seejärel peatub.

```
task main()
{
  OnFwd(OUT_A, 75); //pane mootor A pöörlema edaspidises suunas
  OnFwd(OUT_B, 75); //pane mootor B pöörlema edaspidises suunas
  Wait(4000); //oota 4 sekundit
  OnRev(OUT_AB); //pane mõlemad mootorid pöörlema tagurpidises suunas
  Wait(4000); //oota 4 sekundit
  Off(OUT_AB); //lülita mõlemad mootorid välja
}
```

Joonis 5: NXC näidisprogramm

Järgnevas punktis tutvustatakse Priit Ranna magistritöö raames loodud platvormiülest NXC kasutamise keskkonda NXCEesti.

3.2. Ülevaade NXCEesti rakendusest

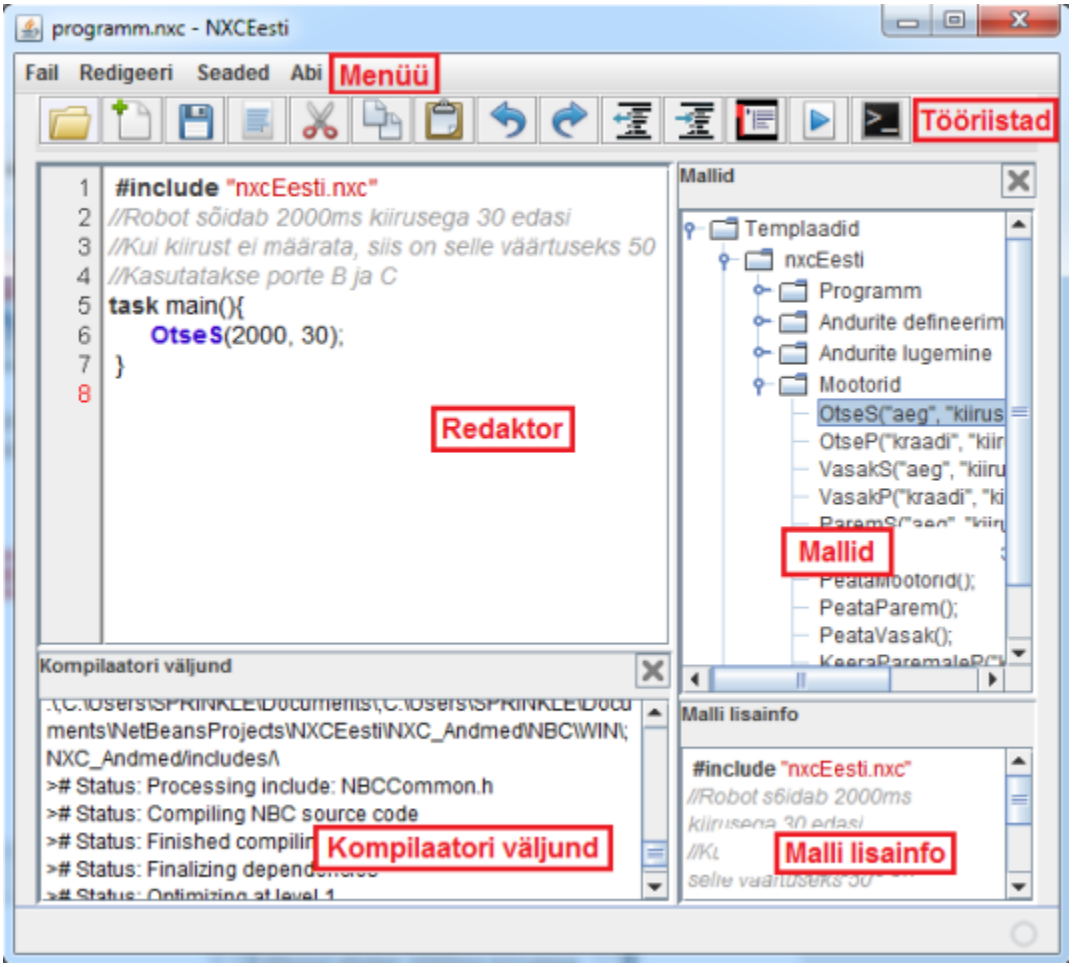
NXC programme on võimalik luua ja kompileerida nii Windows, Mac OS kui ka Linux operatsioonisüsteemides, kuid alles hiljuti puudus NXC arenduskeskkond, mis töötaks ühtmoodi kõigil eelnimetatud platvormidel. Kuna erinevatele operatsioonisüsteemidele olid NXC kasutamiseks erinevad rakendused, oli keeruline koostada universaalset NXC kasutamise juhendmaterjali. Selle probleemi lahendamiseks arendas Tartu Ülikooli tudeng Priit Rand oma 2012. aasta magistritöö raames välja eestikeelse NXC kasutamise keskkonna NXCEesti, mis on kirjutatud programmeerimiskeeles Java ning töötab seega nii Windows, Linux kui ka Mac OS operatsioonisüsteemidel [12]. Kuna NXC keeles kasutatavad sõnad ja funktsioonid võivad algaja eestikeelse programmeerija jaoks raskesti mõistetavad olla, loodi töö raames NXC keele jaoks ka eestikeelsete nimedega standardfunktsioonide teek.

NXCEesti on NXC kasutamise keskkond, mis sisaldab tekstiredaktorit ning igale platvormile sobivat kompilaatorit loodava programmi kompileerimiseks ja robotisse saatmiseks. Tarkvara valib automaatselt õige kompilaatori, tuvastades selleks eelnevalt kasutaja operatsioonisüsteemi.

Programmeerija abistamiseks on NXCEesti rakenduses mallid, mida on võimalik olemasolevasse programmiteksti lisada. Mallid on väikesed programmilõigud, nagu näiteks tingimuslaused või keele rakendusliidese funktsioonid koos märksõnadega, mida kasutajal muuta tuleb. Märksõna viitab tavaliselt sellele, millega ta tuleks asendada. Näiteks võib mallis olla märksõna “port”, mis tähendab, et see tuleks asendada mingile roboti pordile vastava konstandiga.

Kasutajaliides

Joonisel 6 on esitatud programmi NXCEesti peamine kasutajaliidese vaade. Punase tekstiga on tähistatud kasutajaliidese põhilised osad, mida järgnevalt ka lähemalt kirjeldatakse.



Joonis 6: NXC Eesti kasutajaliides [12]

Menüü

Menüüd sisaldavad failide loomiseks ja redigeerimiseks ning programmi seadete muutmiseks vajalikke valikuid.

Tööriistariba

Tööriistariba sisaldab lisaks menüüdes olevatele valikutele ka võimalusi programmis kasutatavate paneelide näitamiseks ja redigeeritava koodi kompileerimiseks.

Redaktor

Redaktor on ala, kus toimub loodava NXC programmi koodi kirjutamine. Märksõnad, rakendusliidese funktsioonid ja konstandid on esitatud rasvases kirjas ja kasutaja poolt määratud värvitoonis. Selleks, et kasutada programmiga kaasasolevat eestikeelsete standardfunktsioonide teeki, tuleb programmi algusesse lisada järgnev rida:
#include "nxcEesti.nxc".

Mallid

Mallid on NXCEesti osa, mis sisaldab erinevaid šabloone, mida saab programmi kirjutamisel kasutada. Mallid sisaldavad enamikke NXC rakendusliidese funktsioonidest, põhilisi programmi struktuure ning Priit Ranna magistritöö käigus tehtud tõlkeid ja standardfunktsioone.

Malli lisainfo

Antud vaates kuvatakse valitud malli kohta lisainfot, kui see on saadaval. Lisainfo sisaldab tavaliselt näidet malli kasutamise kohta.

Kompilaatori väljund

Loodava programmi kompileerimisel kuvatakse selles vaates kompileerimise käigus tekkinud staatuse- ja veateated.

Käesolevas peatükis anti ülevaade NXC keelest ja rakendusest NXCEesti. Järgmises peatükis kirjeldatakse NXCEesti rakendusele lisatud graafilise programmeerimise moodulit.

4. Graafilise programmeerimise mooduli lisamine NXCEesti rakendusele

Käesoleva bakalaureusetöö käigus lisati NXCEesti rakendusele graafilise programmeerimise moodul (edaspidi GPM). Valminud moodul sisaldab NXC eestikeelsete standardfunktsioonide teegis olevatele käskudele ja funktsioonidele vastavaid graafilisi plokke, mida omavahel ühendades on võimalik luua töötav NXC programm. Programmeerimise käigus saab kasutaja genereerida tekkinud plokkstruktuurile vastava NXC koodi, mida on seejärel võimalik NXCEesti rakenduses kompileerida ja robotisse saata. Joonisel 7 on näha GPM keskkonnas valminud lihtne näidisprogramm. Programmile vastava NXC koodi käivitamisel sõidab robot 2 sekundit edasi ja jääb siis seisma.

Joonis 7: GPM keskkonnas loodud näidisprogramm

Järgnevates punktides kirjeldatakse GPM kasutajaliidese loomise protsessi ja NXC koodi generaatori tööpõhimõtet.

4.1. Kasutajaliidese realiseerimine

Graafilise programmeerimise keskkonna loomiseks kasutati vabavaralist Java teeki OpenBlocks [13]. OpenBlocks on Massachusettsi Tehnoloogiainstituudis loodud teek, mille abil saab luua kasutajaliideseid plokk-keelte jaoks. Antud teek võimaldab määratud reegleid järgides koostada loodava plokk-keele spetsifikatsiooni, millele vastavalt tekitatakse kasutajaliides.

Keele defineerimiseks tuleb muuta OpenBlocks teegiga kaasasolevat faili *lang-def.xml*. Failis saab XML keeles kirjeldada loodavat plokk-keelt ja programmeerimise keskkonda vastava keele jaoks.

4.1.1. Plokiklasside kirjeldamine

Keele defineerimiseks tuleb koostada iga plokiklassi (*BlockGenus*) jaoks kirjeldus. Plokiklass määrab kõigi samasse klassi kuuluvate plokkide ühised omadused. Plokiklassi kirjelduses saab määrata järgnevaid põhilisi parameetreid:

- unikaalne nimetus (*name*)
- värv (*color*)
- silt (*initlabel*) - kasutajaliideses annab ploki silt programmeerijale aimu, mis funktsioon plokil on.
- pesad ja pistikud (*BlockConnectors*) - määrab, milliseid plokke antud plokk sisendiks võtta saab, ning millistele plokkidele saab plokk ise sisendiks olla. Näiteks kahe arvu võrdsust kontrollival plokil (joonis 8) on kaks arvu tüüpi pesa (*socket*) ning üks tõeväärtuse tüüpi pistik (*plug*). See tähendab, et antud plokk saab võtta sisendiks ainult selliseid plokke, millel on arvu tüüpi pistik ning saab ise olla sisendiks ainult sellistele plokkidele, millel on vähemalt üks tõeväärtuse tüüpi pesa. Pesade tüübid ja kujud kirjeldatakse dokumendi alguses.
- tüüp (*kind*) - määrab, kas tegemist on käsu-, funktsiooni- või andmeplokiga. Käsuplokke saab ühendada üksteise järgi, et moodustada käskude kogum. Funktsiooniplokid tagastavad mingi väärtuse ja neil on küljes pistik, millega neid saab ühendada teiste plokkide pesadesse. Andmeplokid ei saa teisi plokke sisendiks võtta, kuid võivad olla ise funktsiooni- ja käsuplokkidele sisendiks.

Joonisel 8 on toodud näide kahe arvu võrdsust kontrolliva plokiklassi kirjeldusest ning vastava ploki esitus kasutajaliideses.

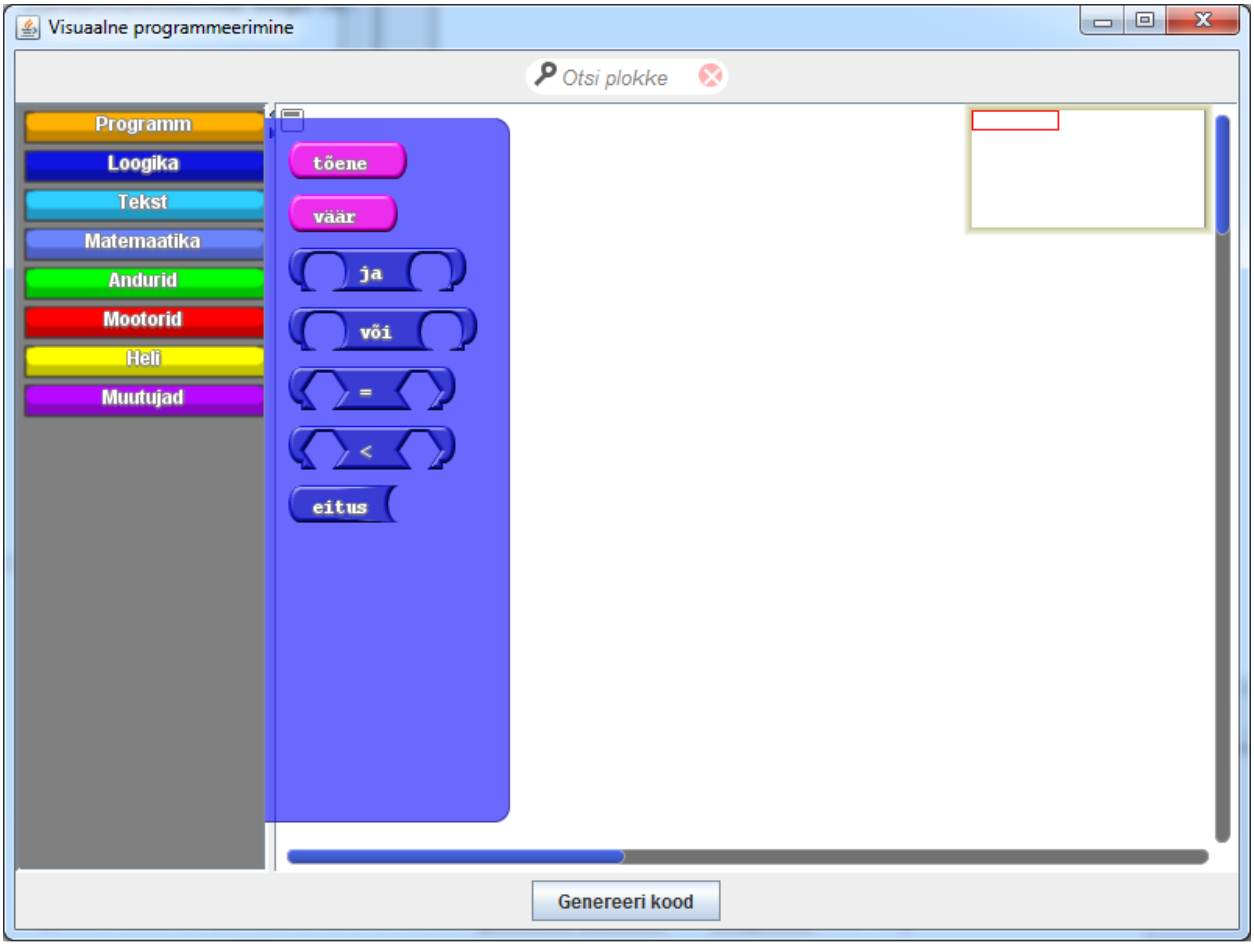
```
<BlockGenus name="equals" kind="function" initlabel="=" color="15 20 220">
        <description>
                <text>
                        Tagastab tõese väärtuse, kui arvud on võrdsed
                </text>
        </description>
        <BlockConnectors>
                <BlockConnector label="" connector-kind="plug"
                        connector-type="boolean" position-type="mirror"></BlockConnector>
                <BlockConnector label="" connector-kind="socket"
                        connector-type="number" position-type="bottom"></BlockConnector>
                <BlockConnector label="" connector-kind="socket"
                        connector-type="number" position-type="bottom"></BlockConnector>
        </BlockConnectors>
</BlockGenus>
```

Joonis 8: kahe arvu võrdsust kontrolliva plokiklassi kirjeldus ja ploki esitus kasutajaliideses

Lisas 2 on välja töödud kõik NXC eestikeelsete standardfunktsioonide teegis olevad käsud, funktsioonid ja andmetüübid, mille jaoks loodi käesoleva bakalaureusetöö käigus plokiklasside kirjeldused.

4.1.2. Töökeskkonna kirjeldamine

Töökeskkonna kirjelduses määratakse plokkide sahtlid (*BlockDrawer*) ning nendes asuvad plokid. Sahtlid on sarnase funktsiooniga plokkide kogumid, kust kasutaja saab valida vajaliku ploki ja selle hiirega töölauale vedada. Joonisel 9 on näidatud loogikatehetega seotud plokkide sahtli kirjeldus ja avatud sahtli esitus kasutajaliideses.




```
<BlockDrawer name="Loogika" type="factory" button-color="15 20 220">
        <BlockGenusMember>true</BlockGenusMember>
        <BlockGenusMember>false</BlockGenusMember>
        <BlockGenusMember>and</BlockGenusMember>
        <BlockGenusMember>or</BlockGenusMember>
        <BlockGenusMember>equals</BlockGenusMember>
        <BlockGenusMember>lessthan</BlockGenusMember>
        <BlockGenusMember>not</BlockGenusMember>
</BlockDrawer>
```

Joonis 9: loogikatehetega seotud sahtli kirjeldus ja esitus kasutajaliideses

Tabelis 1 on kirjeldatud käesoleva bakalaureusetöö käigus graafilise programmeerimise keskkonnale lisatud sahtlid ning nendes asuvad plokid.

Tabel 1: Graafilise programmerimise keskkonda lisatud sahtlid ja nendes asuvad plokid

Sahtli nimi	Sahtlis asuvad plokid
Programm	● Plokid programmi töö juhtimiseks (tsüklid ja tingimuslaused) ja ajastamiseks ● Peaülesande plokk, kust alustatakse programmi tööd
Loogika	● loogikatehete plokid ● tõeväärtuse tüüpi andmeplokid
Tekst	● funktsiooniplokid sõnede töötlemiseks ● sõne tüüpi andmeplokk
Matemaatika	● matemaatiliste tehete plokid ● arvutüüpi andmeplokk
Andurid	● plokid roboti andurite defineerimiseks ja andurite väärtuste lugemiseks
Mootorid	● plokid roboti mootorite juhtimiseks
Heli	● plokid heli tekitamiseks robotis
Ekraan	● plokid graafika ja teksti kuvamiseks roboti ekraanile
Nupud	● plokid, mis kontrollivad, kas mingit roboti nuppu on vajutatud
Muutujad	● plokid muutujate defineerimiseks ja kasutamiseks

Järgnevas punktis kirjeldatakse, kuidas toimub GPM kasutajaliideses loodud programmile vastava NXC koodi genereerimine.

4.2. NXC Koodi genereerimine

NXC Koodi generaatori realiseerimiseks uuris autor graafilise programmeerimise keskkonna Blockly lähtekloodi [14]. Blockly on avatud lähtekoodiga plokk-keeles programmeerimise keskkond, kus on võimalik genereerida plokkidele vastavat koodi JavaScript, Python ja XML keeltes. Kuna JavaScripti süntaks on üsna sarnane NXC süntaksile, sobis antud lähtekood eeskujuks NXC koodi generaatori loomisel. Järgnevalt kirjeldatakse loodud NXC koodi generaatori tööpõhimõtet.

Omavahel ühendatud plokid on OpenBlocks teegis esitatud puu struktuurina. Puu juureks on plokk, mis ei ole ühegi teise ploki sisendiks. Igal plokil on viide talle eelnevale ja järgnevale plokile, kui need eksisteerivad. Kui mingi plokk on teise ploki sisendiks, siis on tal viide teda sisendiks võtvale plokile. OpenBlocks teegis on ploki esindamiseks Java klass *Block*, kus on meetodid vastavale plokile eelnevate ja järgnevate ning tema sisendplokkide leidmiseks.

Enne koodi genereerimist otsitakse kõikide juurplokkide hulgast plokki nimega “põhiülesanne”. Põhiülesande plokk on tavaline käsuplokk, millele saab sisendiks anda teisi käsuplokke, kuid talle ei saa eelneda ega järgneda teisi käsuplokke. Põhiülesande plokk on iga programmi algusplokiks, millega peavad olema ühendatud kõik ülejäänud plokid, et tekiks töötav NXC programm. Kui koodi genereerimisel põhiülesande plokki ei leita või on neid rohkem kui üks, antakse kasutajale vastav veateade ja koodi ei genereerita.

Koodi genereerimiseks lisati OpenBlocks teegis olevasse Java klassi *Block* järgnevad meetodid:

- plokifunktsioon iga plokiklassi jaoks, mis tagastab koodijupi vastavasse klassi kuuluva ploki ja tema sisendplokkide jaoks.
- *blockToCode* - tagastab koodijupi ploki ja talle järgneva ploki jaoks (kutsudes välja neile vastavad plokifunktsioonid), kui see eksisteerib. Kui plokile vastab mingi matemaatiline või loogiline tehe, lisatakse vajadusel koodijupile ümber sulud. Sulgude lisamine või mittelisamine otsustatakse vastavalt tehete järjekorrale NXC keeles [8]. Kui tegu on käsutüüpi plokiga, lisatakse parema loetavuse ja struktureerituse saavutamiseks kõikide tagastatava koodijupi ridade ette tabulaator.

- *callBlockFunction* - kutsub välja plokile vastava plokifunktsiooni ning tagastab saadud koodisõne.
- *getOperationPriority* - tagastab matemaatilist või loogilist tehet esitava ploki tehtejärjekorra numbri. Mida väiksem järjekorranumber, seda kõrgem on tehte prioriteet.

Järgnevalt antakse detailsem kirjeldus plokifunktsioonide ja *blockToCode* meetodi tööst.

Plokifunktsioon

Plokifunktsioon on iga plokiklassi jaoks erinev, kuid kõik nad järgivad sama üldist algoritmi:

1. Kui plokil on sisendplokke, kutsu nende jaoks välja funktsioon *blockToCode* ning salvesta tagastatud koodisõned vastavatesse muutujatesse.
2. tagasta plokile vastav koodisõne, milles sisendplokkidele vastavad kohad on asendatud muutujates olevate väärtustega.

Tingimuslausele kui-siis (*if*) vastav plokifunktsioon tagastab näiteks järgneva koodijupi, kus muutujad `<tingimus>` ja `<tegevus>` asendatakse kui-siis ploki sisendplokkidele vastavate koodijuppidega:

```
kui (<tingimus>) {
<tegevus>
}
```

Kui-siis plokil saab olla kaks sisendplokki: üks tõeväärtuse tüüpi plokk ja üks käsutüüpi plokk, millele võib järgneda teisi käsuplokke. Muutuja <tingimus> väärtuse leidmiseks kutsutakse esimese sisendploki jaoks välja funktsioon *blockToCode*. Muutuja <tegevus> väärtuse leidmiseks tehakse sama järgmise sisendploki jaoks. Joonisel 10 on esitatud kommenteeritud Java kood kui-siis tingimuslausele vastava plokifunktsiooni jaoks.

```
public String ifToCode() {
    //salvesta koodisõne esimese sisendploki jaoks (tingimusplokk)
    String tingimus = getSocketBlockAt(0).blockToCode();

    //salvesta koodisõne teise sisendploki jaoks (käsuplokk)
    String tegevus = getSocketBlockAt(1).blockToCode();

    //loo koodisõne, kus sisendplokkidele vastavad kohad on asendatud
    //neile vastavate muutujate väärtustega
    String kood = "kui(" + tingimus + ") {\n" + tegevus + "}";

    //tagasta tekkinud koodisõne
    return kood;
}
```

Joonis 10: Tingimuslausele kui-siis vastava plokifunktsiooni jaoks loodud Java kood koos kommentaaridega.

blockToCode

Järgnevalt on esitatud blockToCode meetodi algoritmi kirjeldus:

1. kutsu välja plokile vastav plokifunktsioon, kasutades meetodit *callBlockFunction*, ning salvesta tagastatud koodisõne muutujasse.
2. kui tegu on käsuplokiga, siis:
    2.1. lisa muutujas oleva koodisõne iga rea ette tabulaator
    2.2. kui plokile järgneb teine käsuplokk, kutsu järgneva ploki jaoks välja *blockToCode* funktsioon ja liida tagastatud koodisõne muutujas olevale koodisõnele juurde.
3. kui tegu on matemaatilist või loogilist tehet esitava plokiga ning teda sisendiks võttev plokk (vanemplokk) esitab samuti matemaatilist või loogilist tehet, siis
    3.1. võrdle plokile vastava tehte prioriteeti tema vanemplokile vastava tehte prioriteediga.
    3.2. kui vanemplokile vastava tehte prioriteet on kõrgem, lisa muutujas olevale koodisõnele ümber sulud.
4. tagasta muutujasse salvestatud koodisõne

Joonisel 11 on näha kommenteeritud Java kood meetodi *blockToCode* jaoks.

```
public String blockToCode() {
    //kutsu välja plokile vastav plokifunktsioon ning
    //salvesta tagastatud koodisõne muutujasse blockCode
    String blockCode = callBlockFunction();
    //kui tegu on käsutüüpi plokiga
    if (isCommand()) {
        blockCode = indentStatement(blockCode); //lisa koodisõne igale reale tabulaator
        //kui plokile järgneb teine käsuplokk, kutsu välja järgneva ploki meetod blockToCode
        //ja lisa tagastatud koodisõne olemasolevale koodisõnele juurde
        Block afterBlock = getBlockById(getAfterBlockID());
        if (afterBlock != null)
            blockCode += afterBlock.blockToCode();
    }
    //kui plokk esitab matemaatilist või loogilist tehet
    else if (isOperation()){
        Block parentBlock = getParentBlock();
        //kui ploki vanemplokk esitab samuti tehet
        if (parentBlock.isOperation()) {
            //kui vanemploki tehtejärjekord on kõrgem, lisa tagastatavale koodisõnele sulud
            if (parentBlock.getOperationPriority() < getOperationPriority())
                blockCode = "(" + blockCode + ")";
        }
    }
    return blockCode; //tagasta koodisõne
}
```

joonis 11: Meetodit *blockToCode* esitav Java kood

Kogu programmi koodi genereerimiseks kutsutakse plokifunktsioon välja põhiülesande ploki jaoks, millega on ühendatud kõik ülejäänud programmi plokid.

GPM keskkonnas on kasutajal võimalik defineerida muutujaid ja anda neile suvaline nimetus. NXC keeles on aga muutujale nime andmisel mõned kitsendused [11]:

- muutujate nimed peavad algama suure või väikese algustähega, lubatud on kasutada ka alakriipsu (_)
- muutujate nimed ei tohi sisaldada täpitähti (õ,Õ,ä,Ä,ö,Ö,ü ja Ü)
- NXC keeles on kindel hulk märksõnu, mida ei tohi kasutada muutujate nimedena. Nendeks märksõnadeks on: asm, bool, break, byte, case, char, const, continue, default, do, else, enum, false, float, for, goto, if, inline, int, long, mutex, priority, repeat, return, safecall, short, start, static, stop, string, struct, sub, switch, tasks, true, typedef, unsigned, until, void ja while.

Töötava NXC koodi genereerimiseks lisatakse kõigi kasutaja poolt defineeritud muutujate nimede ette alakriips. Alakriipsu lisamise tulemusena genereeritakse töötav kood isegi siis, kui

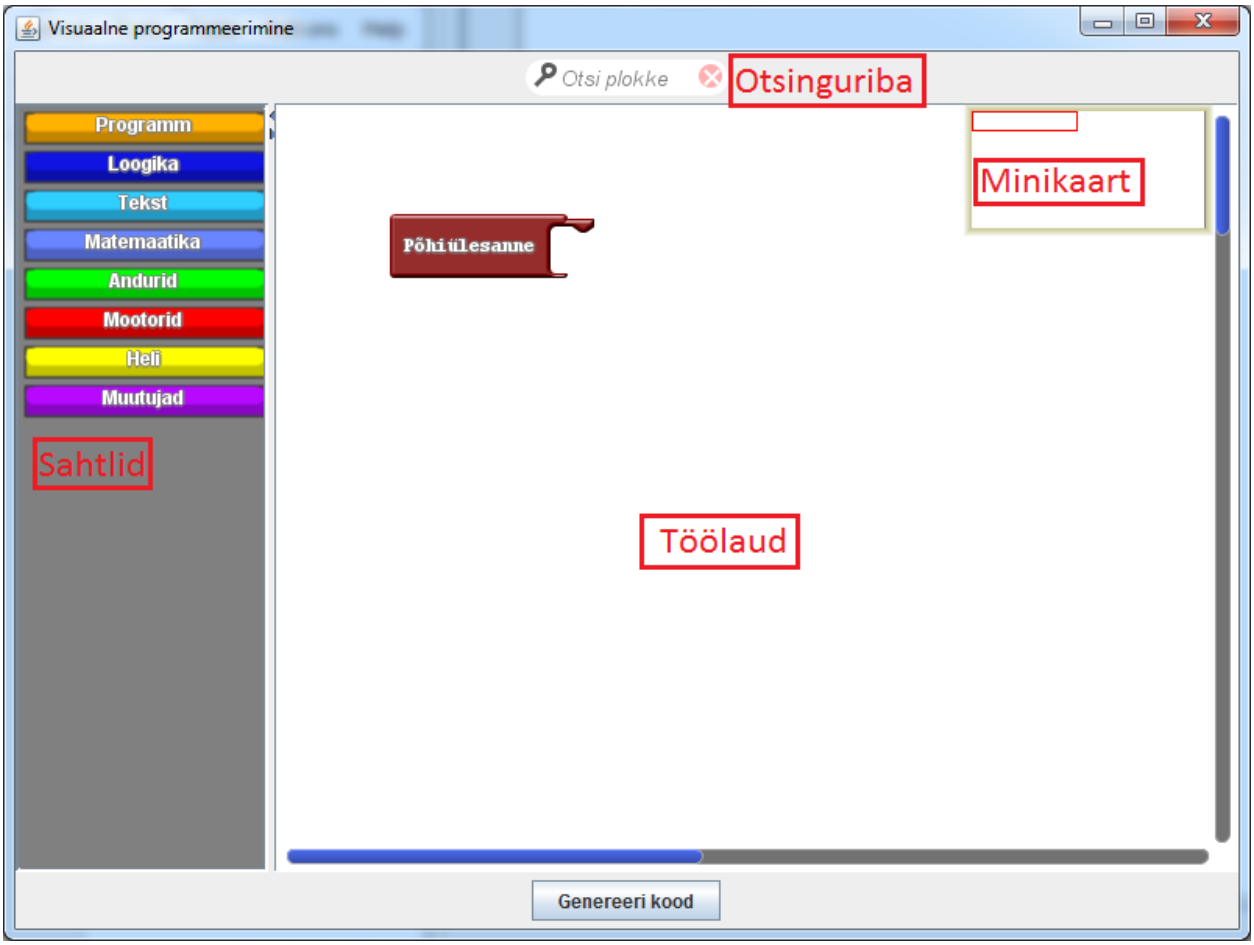
kasutaja otsustab muutuja nimeks anda mõne NXC märksõnadest. Muutujate nimedes sisalduvad täpitähed asendatakse samade tähemarkidega, aga ilma täppideta (näiteks täht “Ä” asendatakse tähega “A”).

4.3. Valminud mooduli kasutamine

GPM avamiseks lisati NXCEesti tööriistaribale puslejupi-kujuline nupp (joonis 12). Nupule vajutades avaneb uus aken, kus on võimalik graafiliste plokkide abil programm luua ning genereerida vastav NXC kood.

Joonis 12: NXCEesti tööriistaribale lisatud nupp (märgitud punaselt), mis avab graafilise programmeerimise mooduli akna.

Joonisel 13 on esitatud GPM kasutajaliides. Järgnevalt kirjeldatakse kasutajaliidese põhilisi osi.



Joonis 13: GPM kasutajaliides

Sahtlid

Sahtlid on sarnase funktsiooniga plokkide kogumid, kust kasutaja saab programmi loomiseks plokke valida ja neid hiirega töölauale vedada. Programmi loomist alustatakse plokist nimega "Põhiülesanne", mis asub sahtlis "Programm". Kõik ülejäänud programmi plokid tuleb anda põhiülesande plokile sisendiks.

Töölaud

Töölaual toimub programmi loomine ja plokkide ühendamine.

Minikaart

Minikaart aitab kasutajal kiiresti töölaual ringi liikuda.

Otsinguriba

Otsinguriba abil saab töölaualt ja sahtlitest otsida plokke nende sildi järgi. Kui töölaualt leitakse mõni otsingusõnale vastava sildiga plokk, tekib talle ümber kollane kontuurjoon. Kui leitud plokk asub mõnes sahtlis, tekib kontuurjoon vastava sahtli nupu ümber.

Genereeri kood

Nupule "Genereeri kood" vajutades genereeritakse töölaual olevale plokkide struktuurile vastav NXC kood, mis ilmub NXCEesti tekstiredaktorisse.

Käesolevas peatükis kirjeldati rakendusele NXCEesti lisatud visuaalse programmeerimise moodulit.

5. Kokkuvõte

NXCEesti on 2012. aastal loodud rakendus, mis võimaldab NXC keeles LEGO Mindstorms NXT robotitele programme luua. Käesoleva bakalaureusetöö eesmärgiks oli lisada NXCEesti rakendusele visuaalse programmeerimise moodul, kus oleks võimalik graafiliste plokkide ühendamise kaudu programme luua. Töö käigus tehti rakendusele järgmised täiendused:

- Visuaalse programmeerimise mooduli kasutajaliides.
- Koodi generaator, mis väljastab visuaalse programmeerimise moodulis koostatud programmile vastava NXC koodi.

Visuaalse programmeerimise moodulit saab kasutada koolilastele robotite programmeerimise õpetamisel.

Kasutatud kirjandus

1. Visual programming language. http://en.wikipedia.org/wiki/Visual_programming_language - vaadatud 14.05.15.
2. Microsoft Visual Programming Language. https://msdn.microsoft.com/en-us/library/bb483088.aspx - vaadatud 14.05.15.
3. Joonis 1. Microsoft Visual Programming Language. https://i-msdn.sec.s-msft.com/dynimg/IC234747.png - vaadatud 14.05.15.
4. Blocks Programming. http://el.media.mit.edu/logo-foundation/pubs/papers/blocks_programming.pdf - vaadatud 14.05.15.
5. LogoBlocks: A Graphical Programming Language for Interacting with the World. http://research.microsoft.com/en-us/um/people/abegel/mit/begel-aup.pdf - vaadatud 14.05.15.
6. What is LOGO. http://el.media.mit.edu/logo-foundation/logo/index.html - vaadatud 14.05.15.
7. The Scratch Programming Language and Environment. http://web.media.mit.edu/~jmaloney/papers/ScratchLangAndEnvironment.pdf. - vaadatud 14.05.15.
8. D. J. Malan, H. H. Leitner, Scratch for budding computer scientists, 2007
9. Joonis 3 - Scratch 2.0 arenduskeskkond http://upload.wikimedia.org/wikipedia/commons/7/76/Scratch_2.0_Default_screen.png - vaadatud 14.05.15.
10. Scratch statistics. https://scratch.mit.edu/statistics/ - vaadatud 14.05.15.
11. NXC Programmer's Guide. http://bricxcc.sourceforge.net/nbc/nxcdoc/NXC_Guide.pdf - vaadatud 14.05.15.
12. P. Rand, Platvormiülene NXC keskkond, 2012
13. Open Blocks Download Page. http://education.mit.edu/openblocks – vaadatud 14.05.15.
14. Blockly lähtekood. https://github.com/google/blockly - vaadatud 14.05.15.

Lisad

Lisa 1. Täiendatud NXCEesti rakendus

Täiendatud programm ja lähtekood on kättesaadavad järgnevalt aadressilt:

http://kodu.ut.ee/~sl2mmer/NXCEesti/NXCEesti.zip

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina **Johannes Ait** (sünnikuupäev 01.02.1991)

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose
"**Visuaalse programmeerimise mooduli lisamine NXCEesti rakendusele**"
mille juhendajad on Taavi Duvin ja Anne Villems,

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.