TALLINNA TEHNIKAÜLIKOOL
INSENERITEADUSKOND
Elektroenergeetika ja mehhatroonika instituut

ELEKTRIAHELATE ÕPPEAINE ATE3150 INTERAKTIIVSED ÕPPEMATERJALID MOODLE ÕPIKESKKONNAS

INTERACTIVE LEARNING MATERIALS FOR THE SUBJECT ELECTRIC-CIRCUITS ATE3150 IN MOODLE LEARNING ENVIRONMENT

BAKALAUREUSETÖÖ

Üliõpilane: Rainer Sild

Üliõpilaskood 154030

Juhendaja: Aleksander Kilk, vanemlektor

Tallinn 2021

AUTORIDEKLARATSIOON

Olen koostanud lõputöö iseseisvalt.
Lõputöö alusel ei ole varem kutse- või teaduskraadi või inseneridiplomit taotletud.
Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

"10" aprill 2021

Autor: .....
/ allkiri /

Töö vastab bakalaureusetöö/magistritööle esitatud nõuetele

"....." ..... 202…..

Juhendaja: .....
/ allkiri /

Kaitsmisele lubatud

"....."..... 202… .

Kaitsmiskomisjoni esimees .....
/ nimi ja allkiri /

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

Mina Rainer Sild

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose Elektriahelate õppeaine ATE3150 interaktiivsed õppematerjalid Moodle õpikeskkonnas,

mille juhendaja on Aleksander Kilk,

1.1 reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

1.2 üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

2. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.

3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

10.04.2021

---

[1] *Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või enam isikut oma ühise loomingulise tegevusega ning lõputöö kaas- või ühisautor(id) ei ole andnud lõputööd kaitsvale üliõpilasele kindlaksmääratud tähtajaks nõusolekut lõputöö reprodutseerimiseks ja avalikustamiseks vastavalt lihtlitsentsi punktidele 1.1. ja 1.2, siis lihtlitsents nimetatud tähtaja jooksul ei kehti.*

LÕPUTÖÖ LÜHIKOKKUVÕTE

Autor: Rainer Sild *Lõputöö liik:* Bakalaureusetöö

Töö pealkiri: Elektriahelate õppeaine ATE3150 interaktiivsed õppematerjalid Moodle õpikeskkonnas

Kuupäev: 10.04.2021 52 *lk (lõputöö lehekülgede arv koos lisadega)*

Ülikool: Tallinna Tehnikaülikool

Teaduskond: Inseneriteaduskond

Instituut: Elektroenergeetika ja mehhatroonika instituut

Töö juhendaja(d): vanemlektor Aleksander Kilk

Töö konsultant (konsultandid): Harry Sild

Sisu kirjeldus:

Interaktiivsete õppematerjalide ja ülesannete loomine elektriahelate õppeaines ATE3150 Moodle õpikeskkonnas. Ülesannete loomisel uuritakse võimalusi Moodle enda pistikprogrammide lisamisega ning seejärel minnakse üle veebistandardi HTML5 failide koostamisele, kasutades arvutiprogrammi Adobe Animate.

Lõputöö sisul on tugev rõhk koodi kirjutamisele – teisendatakse paar näidisülesannet Flashi faili kujult HTML5 kujule, kus üks ülesande faili pakett lubab suhelda Moodle'iga. Seejärel koostatakse originaalne ülesanne ning lõpuks seotakse tehtud ülesanded Moodle õpikeskkonnaga.

Märksõnad: interaktiivne õppematerjal, Moodle, Flash, HTML5, Actionscript, Javascript

ABSTRACT

Author: Rainer Sild *Type of the work:* Bachelor Thesis

Title: Interactive learning materials for the subject Electric-circuits ATE3150 in Moodle learning environment

Date: 10.04.2021 52 *pages (the number of thesis pages including appendices)*

University: Tallinn University of Technology

School: School of Engineering

Department: Department of Electrical Power Engineering and Mechatronics

Supervisor(s) of the thesis: Senior Lecturer Aleksander Kilk

Consultant(s): Harry Sild

Abstract:

Creation of interactive learning materials and assignments in the subject Electric-circuits ATE3150 in Moodle learning environment. When creating exercises, we explore the possibilities with Moodle's own plug-ins, and then move on to creating a web standard HTML5 file, using the executable program Adobe Animate. The dissertation has a strong emphasis on coding – we will convert couple of assignments from a Flash file format to HTML5, wherein one of the assignment file package allows us to communicate with Moodle. Then we will compile our own original assignment and finally integrate the assignments together with Moodle learning environment.

Keywords: interactive learning material, Moodle, Flash, HTML5, Actionscript, Javascript

LÕPUTÖÖ ÜLESANNE

Lõputöö teema:	**Elektriahelate õppeaine ATE3150 interaktiivsed õppematerjalid Moodle õpikeskkonnas**
Lõputöö teema inglise keeles:	**Interactive learning materials for the subject Electric-circuits ATE3150 in Moodle learning environment**
Üliõpilane:	**Rainer Sild, 154030**
Eriala:	**Elektrotehnika AAAB**
Lõputöö liik:	**bakalaureusetöö**
Lõputöö juhendaja:	**Aleksander Kilk**
Lõputöö kaasjuhendaja: (ettevõte, amet ja kontakt)	
Lõputöö ülesande kehtivusaeg:	**10.06.2021**
Lõputöö esitamise tähtaeg:	**18.05.2021**

_____	_____	_____
Üliõpilane (allkiri)	Juhendaja (allkiri)	Õppekava juht (allkiri)

_____
Kaasjuhendaja (allkiri)

1. Teema põhjendus

Elektrotehnika nähtuste ja seoste mõistmine ning elektriahelate analüüsioskused on elektriala inseneriõppe üheks aluseks. Samas hindavad üliõpilased õppeaine „Elektriahelad" omandamist keeruliseks ning suures ajamahus individuaalset tööd nõudvaks erinevate ülesandetüüpide lahendamise oskuste omandamiseks. Käesoleva töö raames on kavas luua Moodle õpikeskkonnas interaktiivseks kasutamiseks erinevate alateemade harjutus- ja kontrollülesandeid ning teste. See võimaldab üliõpilastele paremaid harjutusvõimalusi erinevate ülesandetüüpide lahendamisel, samuti test- ja kontrollülesannete alusel üliõpilaste jooksva õppetöö tulemuslikkuse hindamist.

2. Töö eesmärk

Töö eesmärgiks on üliõpilastele täiendavate võimaluste loomine Moodle õpikeskkonnas elektriahelate ülesannete lahendamise harjutamiseks ja omandatud oskuste kontrollimiseks.

.

3. Lahendamisele kuuluvate küsimuste loetelu:

Elektriahelate erinevate ülesandetüüpide ja harjutusülesannete koostamine.
Testülesannete koostamine ja rakendamine.
Mahukamate kontrollülesannete ja kodutööde variant-ülesannete koostamine.
Elektriahelate analüüsi illustreerivate interaktiivsete lahenduste loomine ja rakendamine.

4. Lähteandmed

Õppeaine „Elektriahelad I ATE 3150" programm, juhendmaterjalid ja üliõpilaste praktiliste tegevuste kava, õppeaine õppejõudude juhised.

5. Uurimismeetodid

Eelnevalt on kavas koostada erinevat tüüpi ülesanded ja seejärel rakendada need Moodle keskkonnas sealsete vahendite abil, lisades vajalikke vahendeid ja linke. Lõputöö koostaja teeb koostööd ja saab nõustamist Moodle Taltech keskuse haridustehnoloogidelt. Illustreerivate ja interaktiivsete õppematerjalide koostamiseks ja rakendamiseks kasutatakse nii Adobe Animate, Photoshop ja Blender programme kui Moodle tehnilise toe poolt võimaldatavaid tarkvaralisi vahendeid ja teiste vahendite abil loodud materjale.

6. Graafiline osa

Töö graafiline osa väljendub Moodle õpikeskkonda loodud interaktiivsete lahenduste kujunduses ja instruktaažis.

7. Töö struktuur

Õppeaine „Elektriahelad I ATE3150" programmist tulenev harjutus- ja testülesannete teisendamine ActionScriptist JavaScripti. Täiendava interaktiivse ülesande koostamine ning rakendamine Moodle õpikeskkonnas.

8. Kasutatud kirjanduse allikad

1. A. Bruce Carlson. Circuits. Engineering Concepts and Analysis of Linear Electric Circuits. Brooks/Cole, Thomson Learning, USA, 2000, 840 pp.
2. L. Neumann, P. Kalantarov. Elektrotehnika teoreetilised alused, I osa. Tallinn 1964, 284 lk.
3. L. Neumann, P. Kalantarov. Elektrotehnika teoreetilised alused, II osa.

Tallinn 1967, 446 lk.
4. Heljut Kalda. Elektrotehnika.Ülesannete lahendusi ja ülesandeid.Tallinn 1998, 52 lk.
5. Moodle õpikeskkonnas kasutatud elektrotehnika alased õppematerjalid ja konspektid.
6. Interneti otsingud võimalike elektrotehnika alaste õppematerjalide teemadel.
7. Harjutustundide ja koduste individuaalülesannete juhendid.

9. Lõputöö konsultandid

Moodle Taltech haridustehnoloogid.

Elektrotehnika valdkonna õppejõud ja teadlased.

10. Töö etapid ja ajakava

1. Tutvumine õppeaine „Elektriahelad I ATE3150" programmi teemade ajakavaga ja vastavate ülesannete valikuga (15.11.2020).
2. Tutvumine Moodle keskkonna tehnilise platvormi ja võimalustega interaktiivsete õppematerjalide loomiseks ja sisestamiseks Moodle õppematerjalidena (20.12.2020).
3. Alalis- ja vahelduvvoolu teemadel harjutus- ja testülesannete koostamine, sisestamine ja kasutusvõimaluste hindamine (31.01.2021).
4. Alalis- ja vahelduvvoolu ahelate kodutööde ülesannete ning kolmefaasiliste ahelate harjutusülesannete koostamine, sisestamine ja kasutusvõimaluste kontrollimine (28.02.2021).
5. Kolmefaasilise ahela kodutöö ja siirdeprotsesside teemal harjutus- ning testülesannete koostamine, sisestamine ja testimine (31.03.2021).
6. Lõputöö teksti esimese versiooni ja vormistuse esitamine juhendajale (15.04.2021).
7. Paranduste ja täiendustega lõputöö teise versiooni esitamine juhendajale (30.04.2021).
8. Lõpliku vormistatud kaitsmisvalmis lõputöö esitamine (15.05.2021).

SISUKORD

Jooniste loetelu

SISSEJUHATUS

Lõputöös uurime võimalusi, kuidas koostada tudengitele interaktiivseid õppematerjale õppeaines „Elektriahelad I". Kõik õppematerjalid laetakse üles Taltech Moodle veebilehele https://moodle.taltech.ee/. Moodle on veebikeskkond, mis lubab koolitajatele luua isikliku õpperuumi vastavalt oma nägemusele. Kõigepealt vaatame, millised on õppekeskkonna Moodle võimalused ning seejärel kasutame interaktiivsete veebilehekülgede loomiseks programmi Adobe Animate, mille põhisisu on elektrotehnilised õppematerjalid ja ülesanded. Ühtlasi muundame mõned olemasolevad õppematerjalid nüüdseks mitte toetatud Flashi failiformaadilt tänapäevasele veebistandardile HTML5 ja koostame uue interaktiivse ülesande. Lõputöös teostatud lahendused on hea baas lugejale, kes tahaks sarnase teemaga rohkem tutvuda ning ise tegeleda interaktiivse õppematerjali koostamise metoodikaga.

Lõputöös kasutatud arvutiprogrammid ja nende otstarve:

Adobe Animate – koodi teisendamine, uue koodi kirjutamine ja veebilehekülgede loomine

Microsoft Word – lõputöö kirjutamine

JPEXS Free Flash Decompiler – Flashi failide muundamine Adobe Animate formaati

Adobe Photoshop – jooniste redigeerimine ja koostamine

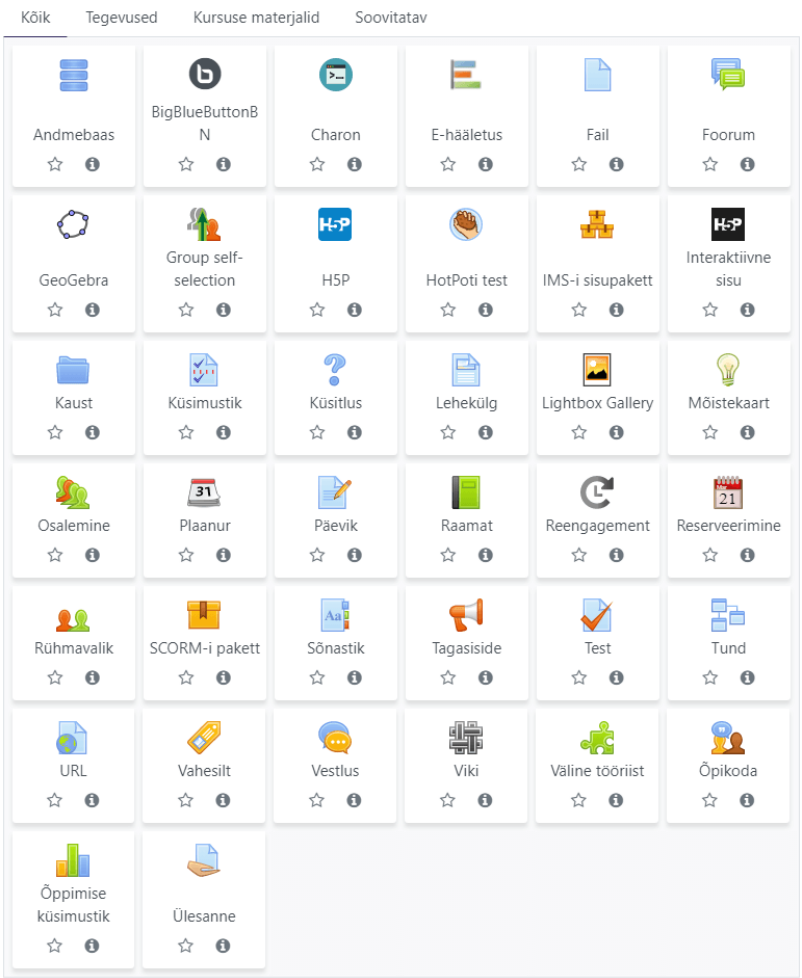
Blender – 3D mudelite ja animatsioonide koostamine

1. INTERAKTIIVSETE ÕPPEMATERJALIDE KOOSTAMISE VÕIMALUSED

Lõputöö eesmärgiks on lisada õppematerjale ja ülesandeid Moodle õpikeskkonda õppeainele „Elektriahelad I ATE3150". Koostatud materjal peab olema oma olemuselt võimalikult interaktiivne, et tudengitel oleks võimalus suhelda arvutis oleva õppematerjaliga. Materjal kuvatakse teksti, arvutuste, animatsioonide ja graafikute kujul. Paremaks mõistmiseks peab kasutaja saama muuta parameetreid nagu arvulised sisestused, nuppude olekud, graafilised skaalad jne.

1.1 Moodle võimalused

Moodle'is on sisseehitatud töövahendid, mis aitavad koolitajal muuta õpperuumi interaktiivsemaks. Õppeaine omanik võib vajutada nupule „Lisa tegevus või resurss" ning menüüst valida oma nägemusele sobiv töövahend.



Joonis 1. Moodle töövahendid

Lühiülevaade Moodle tööriistadest, mis lisavad õpperuumile interaktiivsust:

H5P - kõige suurema interaktiivsete võimalustega töövahend. Lubab koostada mitmeid erinevaid interaktiivseid ülesandeid, kus kasutatakse hiireklõpsamist või tekstisisestust. Väga detailsed juhendid, kuidas ülesandeid koostada saab vaadata https://h5p.org/content-types-and-applications
GeoGebra - matemaatika seotud ülesanded.
Küsitlus - võimalik koostada valikvastustega küsimused.
Fail - saab üles laadida interaktiivseid faile, mis ei ole seotud Moodle töövahenditega, nt Flash, Powerpoint, HTML5, video või animatsioonid.
URL – lubab linkida interakiivseid veebilehekülgi.
SCORM pakett - lubab suhelda failidega, mis ei ole seotud Moodle'i töövahenditega.
Test - saab koostada hindelisi ülesandeid valikvastuste ja teksti sisestuste kujul.

Kõik ülaltoodud tööriistad on kasutuskõlblikud tingimusel, et õppematerjalid ja ülesanded on lihtsalt arusaadavad. Elektriahelate õppeainest arusaamine on tudengitele tihti raske, teadmiste paremaks omandamiseks on hea koostada ülesandeid, kus on samaaegselt kuvatud muutuvad matemaatilised tulemused, joonised ja graafikud. Moodle'i tööriistad ei luba efektiivselt kontrollida ja reguleerida eelnimetatud parameetreid. Sobiva interaktiivsuse saavutamiseks peame koostama ülesanded programmeerimiskeele baasil.

Enne oma ülesande loomist on meile abiks, kui vaatame, kuidas näeb välja üks interaktiivne õppematerjal. Õppeainest Elektrotehnika I AME3140 on hea vaadata näiteid interaktiivsetest harjutus ülesannetest[1].

1.2 Sissejuhatus HTML'i ja JavaScript'i

HTML (Hyper Text Markup Language) on märgistuskeel, millega luuakse veebilehekülјed interneti brauseritele (Google Chrome, Mozilla Firefox jne). Uusim HTML versioon on HTML5. Veebilehekülgede kujundamiseks kasutatakse stiililehe keelt CSS (Cascading Style Sheets) ja veebilehekülje elementide (tekstikastid, nupud, graafikud, joonised jne) omavahelise suhtlemiseks kasutatakse programmeerimiskeelt JavaScript. [2]

Programmeerimiskeelte õppimine on väga ajakulukas ja keeruline ettevõtmine, mis võib kesta mitmeid kuid või aastaid. Lõputöö ülesande ajaliselt efektiivseks lahendamiseks kasutame programmi Adobe Animate, mille kasutajasõbralik liides lubab suhteliselt lihtsalt luua HTML5 lehekülgi, vajalikud on ainult algtasemel oskused JavaScriptis. [3]

JavaScript lubab meil lisada ja seostada HTML5 leheküljel keerulisi elemente. Programmeerimiskeel JavaScript on üsna keeruline, kuid väga laia kasutusalaga. Koodi paremaks mõistmiseks on soovitav vaadata või lugeda instruktaaže internetist[2, 4, 5]. Meie ülesannete puhul on kasulik meeles pidada järgmisi põhitõdesid ja märkusi:

1) Koodi mugavaks lugemiseks kirjutame järjekorras ülevalt alla ning iga koodi tegevus peab lõppema semikooloniga `;`

2) Muutujad lisame käsuga var, nt `var x = 10;` või `var y = „kümme“` , kus x on muutuja, mida saame kasutada numbrina, ja y on muutuja, mida saame kasutada tekstina, mille määrab jutumärgid.

3) Teksti, mida ei kasutata koodina, tähistame kommentaarina, kasutades // või /* kommentaari rea algusena ja */ kommentaari rea lõpuna, nt `//kommentaar` või `/*kommentaar*/`

4) Funktsioonid, mis on mõeldud soovitud ülesande lahendamiseks, kirjutame

```
function funktsiooninimi() {
        funktsioonikeha};
```

kus {} määrab ühe funktsiooni tegevuse. Ühes funktsioonis võib olla mitu tegevust.

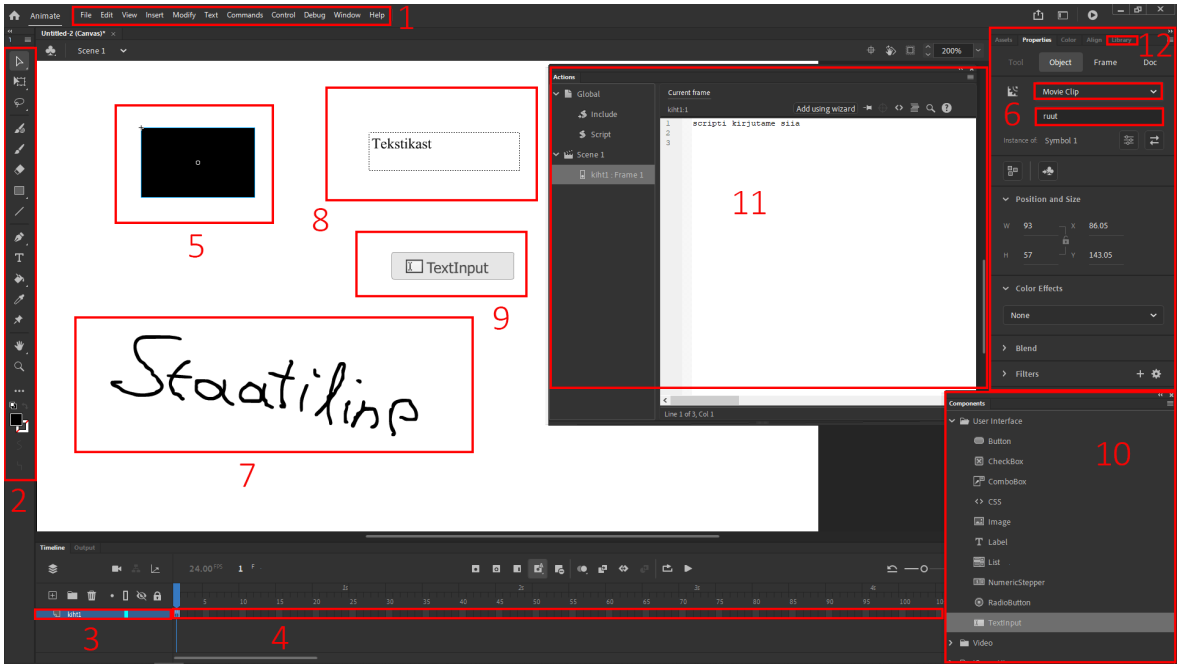
5) Elementide viitamiseks alustame hierarhiliselt kõrgema prioriteediga objektist ja iga järgnevalt madalama objekti eraldame punktiga. Sellist mõistet nimetame pesastatud objektiks. Viitame koodiprogrammis leheküljel olevale tekstikastile `document.elementjoonis.elementekstikast.text`

2 ADOBE ANIMATE

Valdav osa lõputööst on teostatud programmiga Adobe Animate, millega on võimalik koostada interaktiivseid animatsiooni faile, kasutades nüüdseks aegunud ActionScripti või meie puhul tänapäevast JavaScripti.

2.1 Menüü navigeerimine

Avame programmi Animate ja koostame HTML5 canvas. Veebilehitseja faili koostamiseks tutvume tähtsamate programmis olevate tööriistade ja menüüdega. [3, 6, 7]



Joonis 2. Adobe Animate programmi navigatsioon

Tähtsamad navigatsioonielemendid:

1) Pearibamenüü - redigeerimiseks, publitseerimiseks, faili importimiseks, testimiseks ja mitmete teiste funktsioonide avamiseks.

2) Tööriistamenüü - saame tööalal lisada, muuta, valida erinevaid kujundeid ja elemente.

3) Kihimenüü - aitab organiseerida tööalal olevaid elemente.

4) Kaadrimenüü - võimaldab mitmete kaadrite vahel liikuda, meie ülesannetes üle kahe kaadri ei lähe. Üldine kaader on ekvivalentne .html formaadis veebi leheküljele. Adobe Animate lubab koostada elemente, mille kaadrid ei ole seotud pealehel olevate kaadritega. Olemuselt on nad kaadrid peakaadris.

5) Kujund, mis oli algselt tavaline staatiline must kast, mille koostasime tööriista menüüga (2), on muudetud skripteeritavaks elemendiks. Kujundi saame elemendiks muuta, kui vajutame parema hiire klahviga kujundile ning valime „Convert to symbol". Adobe Animate sümbol on ekvivalentne .html elemendile.

6) Elemendi omaduste aken. Siin saame muuta, milliste funktsioonidega on element .html failis, kas töötab näiteks filmi failina, nupuna, tekstina jne. Tähtis on anda elemendile oma nimetus, millele saame JavaScriptis viidata, antud näitel on elemendi nimi „ruut".

7) Staatiline kujund, mis on ainult illustreeriva eesmärgiga.

8) Tekstikast, koostatud tööriista menüüga (2). Tekstikast saab olla staatiline või dünaamiline. Staatilises olekus on ta ainult illustreeriva eesmärgiga. Dünaamilises olekus töötab ta programmeeritava elemendina. Olekuid saame muuta elemendi omaduste aknast (6) ning kindlasti tuleb anda elemendile nimi.

9) Teksti sisestuskast, millele saame veebibrauseris anda väärtusi.

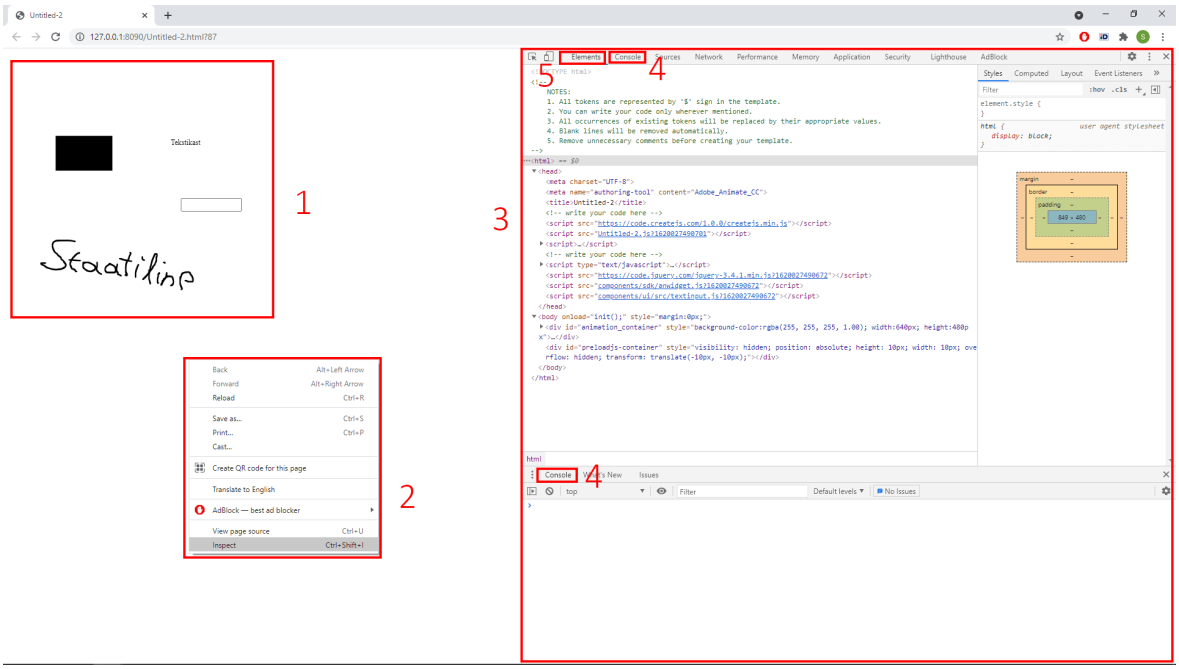
10) Komponentide aken, kus saame töölauale lisada erinevaid elemente. Akna saame lahti teha, kui valime ribamenüüst (1) Window>Components.

11) Koodi kirjutamise aken. Selles aknas saame oma ülesande muuta interaktiivseks, kasutades JavaScripti. Skript võimaldab meil suhelda elementidega, teha matemaatilisi tehteid, koostada graafikuid ja veel palju muud. Akna lahti tegemiseks valime ribamenüüst (1) Window>Actions.

12) Elementide teek, kus saame mugavalt valida ja muuta üksikute elementide (sümbolite) kujunduslikke ning skriptile iseloomulikke omadusi.

2.2 HTML5 testimine ja koodi valideerimine

Väga tihti on meil vaja kontrollida tehtud tööd ning parandada võimalikke vigu. HTML5 faili testimiseks valime ribamenüüst Control>Test. See avab vaikimisi brauseris uue interneti lehekülje, kus on meie poolt koostatud fail. [6, 7]



Joonis 3. Tehtud töö näide Animate programmis

Brauseriaknas (1) näeme tööalal koostatud musta kasti elementi nimega „ruut", käsitsi kirjutatud pilti tekstiga „Staatiline", dünaamilist tekstikasti, tekstiga „Tekstikast" ja tekstisisestuskasti. Täpsema info saamiseks avame brauseris infomenüü (3), mis avaneb, kui vajutame hiire parema klahviga internetilehekülje ja valime „Inspect" (2). Antud menüü on veebibrauseri põhine, minu töös on kasutatud Google Chrome'i. Infomenüüs on tähtis vaadata konsooliakent (4), kus kuvatakse programmeerimise käigus võimalikud tekkinud vead. Elemendiaken (5) lubab vaadata elementidega seotud informatsiooni.

Adobe Animate koodi kirjutamise aknas võib koodiridu olla sadu või tuhandeid, koodis vea tekkimise võimalus on väga tõenäoline. Adobe Animate programmis ei ole sisse ehitatud veaotsingut. Samas JavaScript on väga levinud programmeerimiskeel, mille koodi õigsust saab kontrollida interneti või koodi kirjutamiste aplikatsioonide kaudu. Otsingumootoris on vaja otsida „JavaScript validator". Otsingu tulemusi on mitmeid, nad kõik töötavad sarnaselt. Õigsuse kontrollimiseks kopeerime oma koodi ja kleebime validaatorisse ning vajutame „Validate". Validaator otsib üles võimalikud vead.

3 ÕPPEMATERJALI TEISENDAMINE

Elektrotehnika I õppeainest on meil mitmeid näiteid interaktiivsetest ülesannetest[1]. Probleem on selles, et nad on koostatud Adobe Flashiga .swf kujul. Alates veebruarist 2021 interneti veebilehitsejad enam ei toeta .swf faile. Väga paljud õppematerjalid on .swf formaadis, mis on vaja ümber ehitada HTML5'le. Uurime lähemalt Flashi ja HTML5 omapärasid ning vaatame kahe erineva raskusega ülesannet.

3.1 Flashi teisendamine

Flash failid on koostatud Adobe Flashis, mis on analoogne programm Adobe Animate'le. Flashi programeerimiskeeleks on ActionScript 1.0, ActionScript 2.0 või ActionScript 3.0. Keerulisi .swf faile, mis kasutavad ActionScript 1 ja 2, ei ole võimalik automaatselt teisendada HTML5 kujule. Erandiks on ActionScript 3.0, mis on väga sarnane keel JavaScriptile. Samas ei ole ActionScriptil enam mingit otstarvet, kuna internetibrauserid enam ei toeta .swf faile.

Adobe Animate publitseerib brauseri jaoks faile .swf ja .html kujul. Nende sisu võib olla pildid, filmid, interaktiivsed failid ja kõik muu, mida ActionScript ja JavaScript võimaldab. Flashi fail .swf ja HTML5 fail .html on lõpptoodang, Adobe Animate koostab neid faile, kasutades endale omast formaati .fla. Juhul, kui puudub originaal .fla fail, peame valmis Flashi faili dekompileerima .fla kujule. Dekompileeritud fail on ActionScripti kujul, sealt peame ta teisendama JavaScriptiks. Kõike ajakohasem dekompileerija on JPEXS Free Flash Decompiler.

Antud programm suudab rohkem, kui ainult dekompileerida. Sellega on võimalik vaadata kujundite, tekstikastide ja nuppude muutujate nimetusi. Skriptis olevad käsud viitavad elementide nimetustele, nimed on suure tähtsusega. Paljud elektrotehnika õppematerjalid on tehtud ActionScriptiga 2.0 või vanema versiooniga. Avades .fla faili adobe Animate'ga, näeme, et mõningad elemendid, näiteks teksti sisestuskastid, puuduvad, seega peame nad käsitsi lisama. Elementide nimetusi on hea vaadata JPEXS'i pealt. Soovitavalt on kõik elemendid sama nimetusega kui originaal .swf failis, siis on lihtsam skriptist lugeda, mis on mingi elemendi ülesanne.

3.2 ActionScripti ja JavaScripti eripärasused Adobe Animate'is

ActionScripti ja JavaScripti publitseeringud on funktsionaalsuselt samasugused, publitseeringu saamiseks peame meeles pidame Adobe Animate'is kahe kodeerimiskeele peamisi erinevusi[6,7]:

1) ActionScripti ja JavaScripti funktsiooni sisse saab panna muutujad, kuid JavaScripti muutujad ei ole väljaspool funktsiooni ligipääsetavad.

2) ActionScripti esimene kaader on 1 ja JavaScriptil 0.

3) ActionScriptil töötab dünaamiline teksti kasti sisendina ja väljundina. JavaScriptil töötab dünaamiline teksti kast väljundina ja on eraldi vaja lisada teksti sisestuskast.

4) ActionScripti teksti sisestuskast töötab kogu aeg, JavaScriptil on vaja teksti sisestuskasti jaoks lisada viivitusfunktsioon, kuna canvas ei ole veel jõudnud joonestada teksti sisestuskasti ekraanile `setTimeout(function () { kood siia }, 100);`

5) ActionScripti ürituste läbiviija (inglk. *event handler*) `onEnterFrame`, mis täidab sisestatud koodi lõuendi ühe kaadri muutumisel. Ekvivalentne kood JavaScriptis on `function tickHandler(e) { kood siia } createjs.Ticker.on("tick", tickHandler);`

6) ActionScriptis saab elemente peita `.visible=true/false` omadusega. Sama kood töötab ka JavaScriptis. Peale teksti sisestuskasti elemendi kasutame `document.getElementById("tekstikast").style.visibility ="visible";`

7) JavaScriptis saame funktsioonis muutujate väärtused välja tuua `return` käsuga.

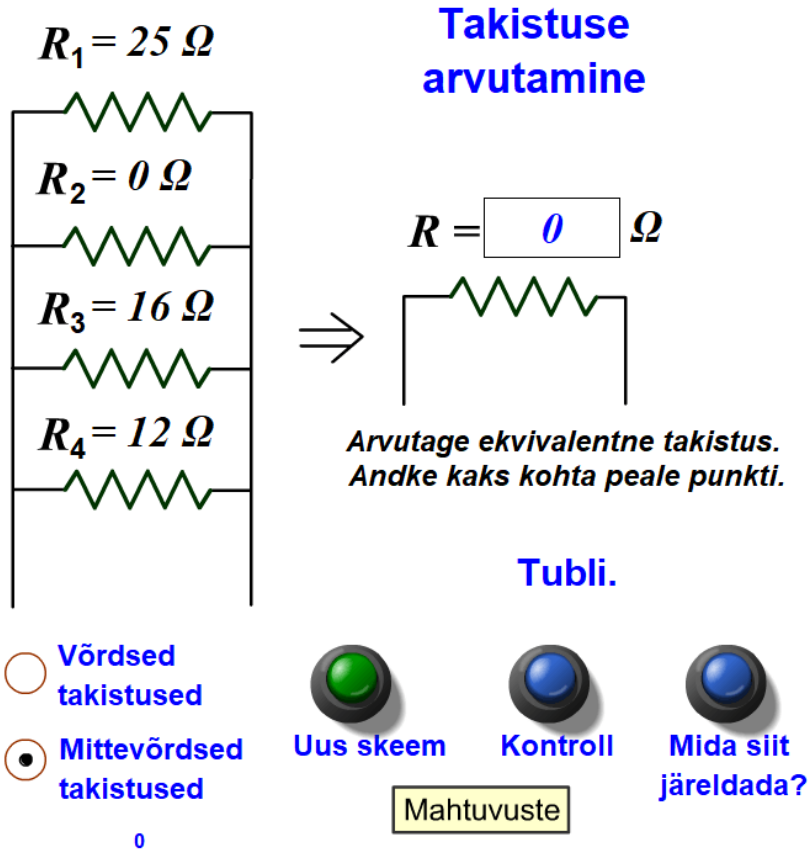
3.3 Harjutusülesande „Rööptakistus" teisendamine

Järgnevalt teisendame harjutusülesande „Rööptakistus" Flashi kujult HTML5 kujule. Uurime harjutuse eesmärki, võttes veel abiks plokkskeemi.

3.3.1 Harjutus „Rööptakistus"

Harjutus ise on mitmekülgne – eesmärgiks on arvutada rööbiti olevate takistuste koguväärtus. Rööptakistuse arv genereeritakse nupuga „uus skeem" vahemikus 2 kuni 4 ning saame valida, kas nad on võrdsete või mittevõrdsete väärtustega. Kontrolli nupuga saame kontrollida oma vastust ja järeldusnupp annab vihjeid. Kui vastus on õige, saame vastuse kasti sinise teksti „Tubli, õige", vastasel juhul punaselt „Ei ole õige". Mahtuvuste arvutuse nupp viib meid järgmisele leheküljele. Kuna mahtuvuste arvutamise kood on sarnane takistuste arvutamisele, keskendume ainult hilisemale. [1]

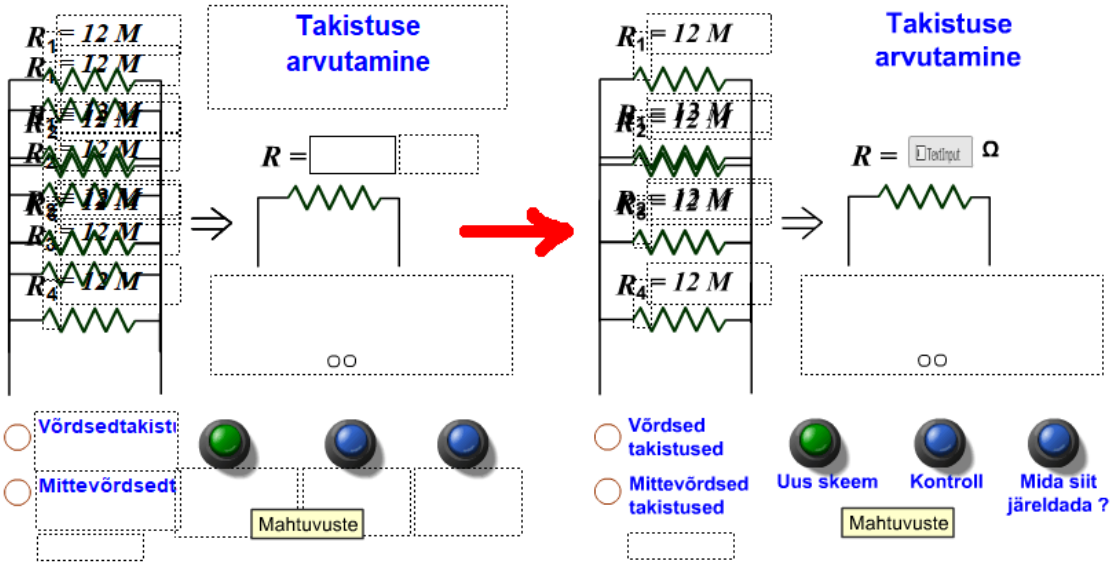
Ülesande vastus on $\frac{1}{R}=\frac{1}{R_1}+\frac{1}{R_2}+\frac{1}{R_3}+\frac{1}{R_4}$, mahtuvuse kaadris oleks olnud $C=C_1+C_2+C_3+C_4$



Joonis 4. Harjutus „Rööptakistus"

3.3.2 Flashi dekompilatsioon ja importimine

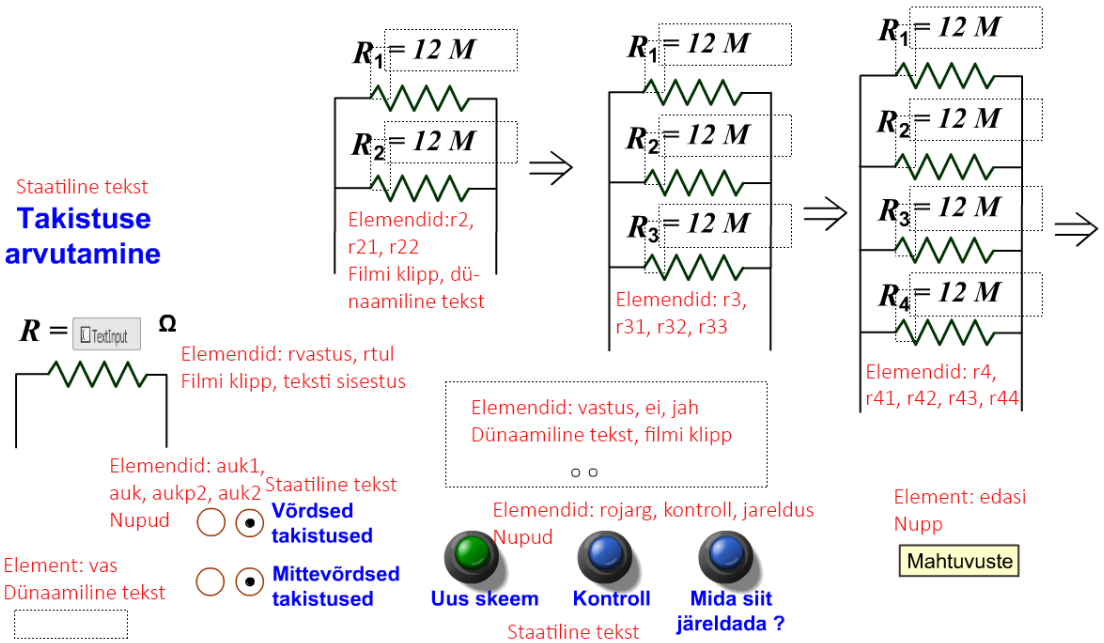
Flashi faili dekompileerime JPEX Flash Converteriga ning saadud .fla faili avame Adobe Animate'iga. Avades faili näitab see viga, mis ütleb, et ActionScript versioon 1.0 ei ole toetatud ning eemaldab kõik elementide nimed. Meil on vajalik võimalikult täpselt taastada elementide asukohad ja nimed, mida saame JPEXi abil lihtsalt teha. Joonisel 5 on toodud taastamise algus ja lõpp.



Joonis 5. Elementide taastamine

Taastamisel teeme muudatused, milles asendame mõningad dünaamilised teksti kastid staatilistega. Nad ei oma JavaScriptis erilist tähtsust ja see võimaldab ülesannet natukene lihtsustada. Edasiselt muudame .fla faili ActionScript 3.0 ümber JavaScript baasile. Selleks läheme ribamenüüsse, valime File>Convert To>HTML5 Canvas.

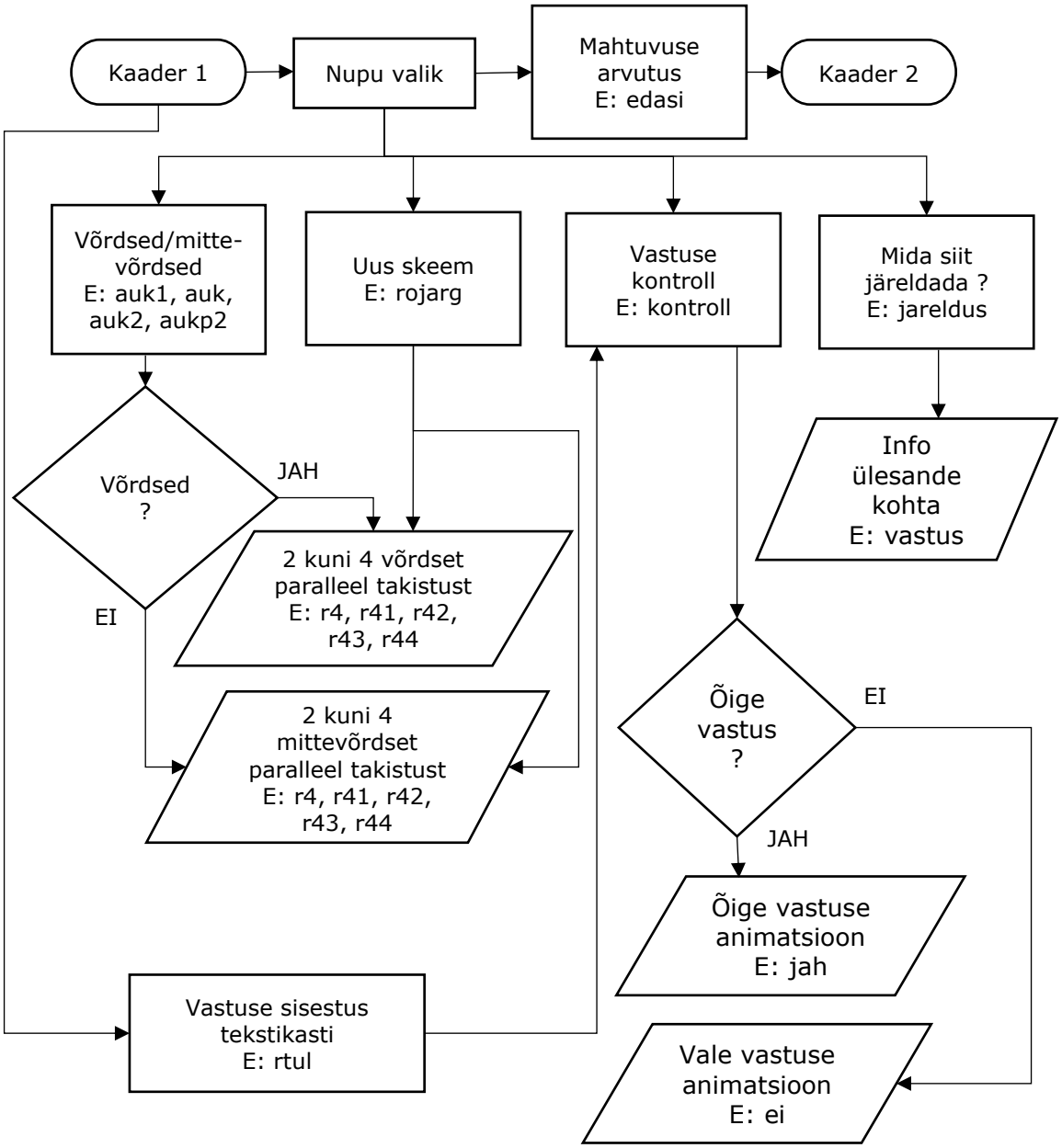
Üksikutest elementidest paremaks arusaamiseks toome nad eraldi välja joonisel 6. Kõik elemendid on skriptis kasutuses. Need omavad unikaalset nime ja funktsiooni, olles kas tekstikast, teksti sisestuskast, kujundus või nupp.



Joonis 6. Elemendid ükshaaval välja toodud

3.3.3 Ülesande plokkskeem

Paremaks harjutuse töövoost aru saamiseks koostame plokkskeemi. Skeemil on kujutatud esimese kaadri valikud. Kaadri funktsionaalsus lõppeb, kui minnakse üle kaader kahele. Plokkides on toodud elemendi ülesanne ja elemendi nimetused, tähistades sõnaga „E". Viidatud elemendid on ActionScripti ja JavaScripti koodides kasutuses.



Joonis 7. Rööptakistuse ülesande lahenduskäigu plokkskeem

3.3.4 JavaScripti lugemine

Järgnev kood on funktsionaalne ja töötab võimalikult sarnaselt harjutusülesandega „Rööptakistus". Teisendamise lahendus on paindlik, antud lahendus on minu poolt koostatud, kindlasti saab ülesandele läheneda mitmel erineval viisil. Originaalne ActionScripti kood on esitatud lisa 1. Koodi lõpus on koodirea lahtiseletused.

```
var root = this; //esimese kaadri nime muudame "root"

root.r2.visible = false; //peidame elemente
root.r3.visible = false;
root.r4.visible = false;
root.cvastus.visible = false;
root.rvastus.visible = false;
root.jah.visible = false;
root.ei.visible = false;
root.auk.visible = true;
root.auk1.visible = false;
root.aukp2.visible = true;
root.aukp.visible = false;

var abi = 1; //deklareerime kõik kasutatavad muutujad
var nr = 0;
var Ra;
var Rb;
var Rc;
var Rd;
var k;
var rr1;

function fl_MouseClickHandler_1() {//funktsioon, mis täidab järgmisi ülesandeid
    document.getElementById("rtul").style.visibility = "hidden"; //peidame teksti sisestuskasti rtul
    root.vastus.text = ""; //eemaldame teksti kastist vastus kõik väärtused
    root.gotoAndStop(1); //lähme üle teisele kaadrile, kondensaator
}
root.edasi.addEventListener("click", fl_MouseClickHandler_1);
//vajutades nuppu "edasi" aktiveerime funktsiooni fl_MouseClickHandller_1

function fl_MouseClickHandler_2() { //funktsioon, mis täidab järgmisi ülesandeid
    document.getElementById("rtul").style.visibility = "visible"; //muudab rtul nähtavaks
    root.rvastus.visible = true; //muudab rvastus nähtavaks
    root.vastus.text = "Arvutage ekvivalentne takistus. \rAndke kaks kohta peale punkti.";
    //annab väärtuse tekstikasti vastus
    nr = nr + 1; //annab +1 väärtuse muutujale nr
    Ra = Math.floor(Math.random() * 29); //annab suvalised väärtused muutujatele Ra, Rb, Rc, Rd, k
    Rb = Math.floor(Math.random() * 9);
    Rc = Math.floor(Math.random() * 19);
    Rd = Math.floor(Math.random() * 15);
    k = Math.floor(Math.random() * 3 + 1);

    function fl_MouseClickHandler_3() { //funktsioon, mis töidab järgmisi ülesandeid
        if (nr > 3) { //tingimusel, kui nr on suurem 3st
            if (Rb === 0 || Rc === 0 || Rd === 0 || Ra === 0) { //või loogika operant
                root.vastus.text = "Kui üks takistus on null\rsiis on kogutakistus null.";
            } else if (abi == 1) { //annab vastus tekstikastile väärtused tingimustel
                root.vastus.text = "Võrdsete takistuste korral\rväheneb takistus n korda.";
            } else {
                root.vastus.text = "Mitteõrdsete takistuste korral\ron kogutakistus alati väiksem\rkui kõige väiksem takistus.";
            }
        }
    }
    root.jareldus.addEventListener("click", fl_MouseClickHandler_3);
    //vajutades nuppu "jareldus", käivitab funktsiooni fl_MouseClickHandler_3

```

```
function fl_MouseClickHandler_4() { //funktsioon, mis täidab järgmisi ülesandeid
    root.auk1.visible = false;
    root.auk.visible = true;
    root.aukp2.visible = true;
    root.aukp.visible = false;
    abi = 1; //võrdsed väärtused
}
root.auk1.addEventListener("click", fl_MouseClickHandler_4); //vajutades nuppu auk1

function fl_MouseClickHandler_5() { //funktsioon, mis täidab järgmisi ülesandeid
    root.aukp2.visible = false;
    root.aukp.visible = true;
    root.auk.visible = false;
    root.auk1.visible = true;
    abi = 0; //mitte võrdsed väärtused
}
root.aukp2.addEventListener("click", fl_MouseClickHandler_5); //vajutades nuppu aukp2

if (abi === 1) {
    //kui on valitud võrdsed väärtused kast, siis Rb, Rc ja Rd on võrdsed Ra
    Rb = Ra;
    Rc = Ra;
    Rd = Ra;
}

root.r4.r41.text = "= " + Ra + " Ω"; //takistuste väärtuse kuvamine
root.r4.r42.text = "= " + Rb + " Ω";
root.r4.r43.text = "= " + Rc + " Ω";
root.r4.r44.text = "= " + Rd + " Ω";
root.r3.r31.text = "= " + Rb + " Ω";
root.r3.r32.text = "= " + Rc + " Ω";
root.r3.r33.text = "= " + Rd + " Ω";
root.r2.r21.text = "= " + Rc + " Ω";
root.r2.r22.text = "= " + Rd + " Ω";

if (k === 1) { //kui k on 1, siis 2 rööptakistit
    root.r2.visible = true;
    root.r3.visible = false;
    root.r4.visible = false;
    if (Rc === 0 || Rd === 0) {
        Ra = 0;
    } else {
        Ra = Math.round((100 * (Rc * Rd)) / (Rc + Rd)) / 100; //rööptakistuse väärtus
    }
}
if (k === 2) { //kui k on 2, siis 3 rööptakistit
    root.r2.visible = false;
    root.r3.visible = true;
    root.r4.visible = false;
    if (Rb === 0 || Rc === 0 || Rd === 0) {
        Ra = 0;
    } else {
        Ra = Math.round((100 * 1) / (1 / Rb + 1 / Rc + 1 / Rd)) / 100;
    }
}
if (k === 3) {
    root.r2.visible = false;
```

```
115         root.r3.visible = false;
116         root.r4.visible = true;
117         if (Ra === 0 || Rb === 0 || Rc === 0 || Rd === 0) {
118             Ra = 0;
119         } else {
120             Ra = Math.round((100 * 1) / (1 / Ra + 1 / Rb + 1 / Rc + 1 / Rd)) / 100;
121         }
122     }
123     root.vas.text = Ra; //spikker vastusega
124 }
125
126 root.rojarg.addEventListener("click", fl_MouseClickHandler_2);
127 //vajutades nuppu "rojarg" aktiveerime funktsiooni fl_MouseClickHandller_2
128
129 function rr() { //teksti sisestuskasti väärtuse funktsioon
130     return Number(rr1.value); //tagastame funktsioonist väärtuse rr1, mille saame rtul sisestusest
131 }
132
133 setTimeout(function () { //lisame viivituse 100ms ja koostame funktsiooni
134     document.getElementById("rtul").style.visibility = "hidden";
135     //teksti sisestus peidetud kuni vajutame nuppu "rojarg"
136     rr1 = document.getElementById("rtul"); //teksti sisestusekasti väärtuse leidmine rr1
137     function fl_MouseClickHandler_6() {
138         //funktsioon
139         if (Math.abs(Ra - rr()) / (rr() + 0.000001) < 0.05) {
140             //võrdleme teksti sisestuse rr() arvutuslike väärtusega Ra
141             root.ei.visible = false;
142             root.jah.visible = true;
143             root.jah.gotoAndPlay(0); //kui vastus on piirde seest, mängi animatsiooni jah
144         } else {
145             root.jah.visible = false;
146             root.ei.visible = true;
147             root.ei.gotoAndPlay(0); //kui vastus ei ole piirete sees, mängi animatsiooni ei
148         }
149     }
150     root.kontroll.addEventListener("click", fl_MouseClickHandler_6);
151     //vajutades nuppu "kontroll" aktiveerime funktsiooni fl_MouseClickHandler_6
152 }, 100); //funktsiooni viivitus
153
154 root.stop(); //peatab kaadri animatsiooni mängimise
```

Koodirea lahtiseletused[4, 5, 6, 7, 8, 9]:

1 – Anname väärtuse „this" muutujale „root". Sõna „this" viitab hetkekontekstile. Kui on mitu kaadrit, on mõistlik koostada mitu erinevat muutujat, nt root1, root2 jne.

3 kuni 13 – Peidame elemente, mis asuvad root kaadris. Peitmiseks ja nähtavaks tegemiseks kasutame loogikat true/false või 1/0.

15 kuni 22 – Deklareerime kõik kasutatavad muutujad. Kui muutujad on deklareeritud, viitame neile ilma eesliiteta „var", vastasel juhul kirjutame muutuja uuesti üle.

24 kuni 30 – Koostame funktsiooni fl_MouseClickHandler_1. Funktsioon peidab tekstisisestuskasti „rtul" ja läheb järgmisele kaadrile, kus leitakse kondensaatorite väärtused. Funktsiooni käivitame, kui vajutame nuppu „edasi".

32 kuni 42 ja 126 – Koostame funktsiooni fl_MouseClickHandler_2. Funktsioon teeb mõned elemendid nähtavaks ja annab muutujatele suvalised väärtused. Läbi selle funktsiooni käivitamise saame edasi käivitada funktsioonid MouseClickHandler_3, 4 ja 5. Funktsioon 2 käivitame, kui vajutame nuppu „rojarg"

44 kuni 56 – Koostame funktsiooni fl_MouseClickHandler_3. Funktsioon annab tekstikasti „vastus" kolm erinevat teksti väärtust, kui üks takistustest on null ning kui on valitud kas võrdsed või mittevõrdsed takistused. Funktsiooni käivitame, kui vajutame nuppu „jareldus".

58 kuni 74 – Funktsioonid 4 ja 5 lubavad valida, kas takistused on võrdsed või mitte.

76 kuni 91 – Kuvab numbrilised Ra, Rb, Rc, Rd väärtused tekstikastidesse. Kui eelnevalt valitud abi väärtus on 1, siis kõik takistuse väärtused võrduvad Ra.

93 kuni 124 – Muutuja k määrab ära, kas meil on 2, 3 või 4 rööbiti takistust. Muutuja Ra on meie poolt otsitav ülesande vastus.

129 kuni 136 – Teksti sisestuskasti „rtul" väärtuse ümber muutmine koodis kasutavaks muutujaks või funktsiooniks. Paneme tähele, et antud funktsioon käivitub pärast väikest viivitust, kuna Adobe Animate ei jõua veel lõuendile joonistada tekstikasti.

137 kuni 150 – Koostame funktsiooni fl_MouseClickHandler_6. Funktsioon võrdleb meie poolt pakutut vastust rr() tegeliku vastusega Ra. Kui vastus on tolerantside vahel, mängib animatsooni „jah", vastasel juhul animatsiooni „ei". Funktsiooni käivitab nupp „kontroll".

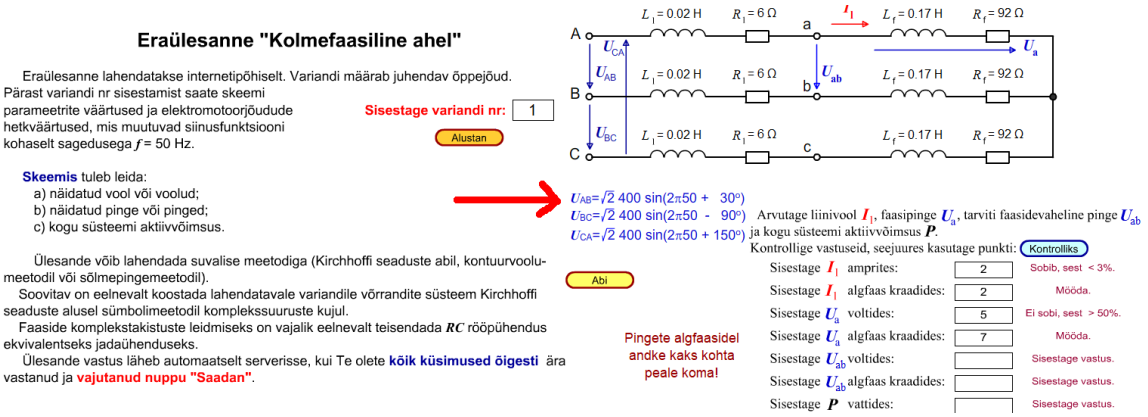
154 – Peatab kaadri nimega root mängimise. Kui jääb kirjutamata, siis HTML5 lehekülg jääb igavesti mängima meie kaadri animatsioone.

3.4 Harjutusülesande „Kolmefaasiline ahel“ teisendamine

Tõlgime ümber harjutusülesande „Kolmefaasiline ahel“ Flashilt HTML5’le. Lõputöö näites lühendame harjutusülesande 1600 realt 300 reale.

3.4.1 Harjutus „Kolmefaasiline ahel“

Harjutus koosneb kahest kaadrist, kus esimesel kaadril valitakse õppejõu poolt antud ülesande variant ja teisel kaadril antakse ülesande algandmed koos skeemidega. Vaja on leida pinged, voolud, faasinurgad ja võimsus. Vajutades nuppu „Kontroll“ saab teada, kas vastused on tolerantside piires. Kui kõik vastused on tolerantside vahel üles leitud, tekib uus nupp, mis lubab saata ülesande tulemused Moodle õpikeskkonda. Meie näites uurime ainult ühte vastuse kasti. Ülesande variante on 24, meie vaatame ainult esimest ülesande varianti. [1]



Joonis 8. Kolmefaasiline ahela kaader 1 ja kaader 2

Dekompileerimise protseduur on sama nagu harjutuses „Rööptakistus“. ActionScripti ja JavaScripti koodi võib lugeda lisa 2 ja lisa 3. Koodiread on kommenteeritud ja töötavad sarnaselt harjutusega „Rööptakistus“.

3.4.2 SCORM paketi aktiveerimine ja globaalne muutuja

Ülesande üks eripärasus on muutuja „nr“, mis on muutuja kaader 1 ja 2 vahel. Väärtuse üle kandmiseks peame „nr“ lisama globaalsete muutujate kategooriasse, joonisel 8.

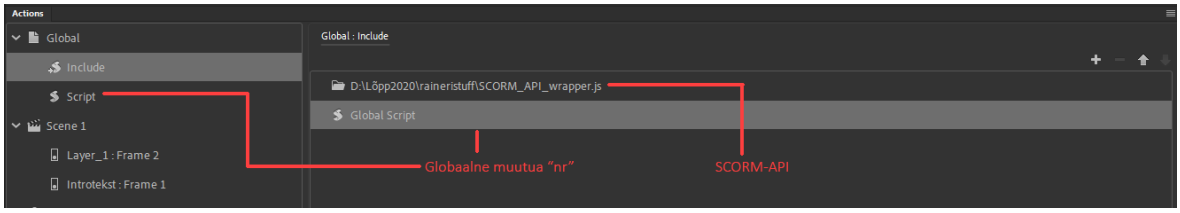
Ülesande teine eripärasus on muutuja „hind“ väärtuse saatmine Moodle õpikeskkonda, õppejõu aine leheküljel. Väärtus „hind“ iseloomustab õppejõu poolt antud ülesande variandi numbrit. Ülesande ja Moodle vahel suhtlemiseks kasutame SCORMi (Shareable Content Object Reference Model). SCORMile viitab Lisa 2 ja 3 > kaader 2 > skriptide esimesed 30 rida. Skriptis kasutame ainult käsku scorm.set, mis annab Moodle’sse

muutuja „hind" väärtuse. Mitmeid teisi käske võib vaadata internetist[10]. SCORM ei ole JavaScripti sisse ehitatud, et skript läheks korrektselt tööle, peame lisama SCORM-API (Application Programming Interface). Läbi API saame suhelda kahe meediumi vahel. API saame alla laadida pipwerks veebileheküljelt . API peame lisama globaalse välise skripti menüüse, mis on välja toodud joonisel 8.[11]

Publitseeritud SCORM baasil ülesande failid peame omakorda seostama .xml manifest failiga. See on fail, mis seob omavahel .html faile ja SCORM-API skripti. Manifesti võime koostada Notepadis, meie ülesandes on faili kood järgmiselt[1]:

```
<?xml version="1.0"?>
<manifest      identifier="SCORM_AS2_class_test"
                      version="1.2">

       <metadata>
              <schema>ADL SCORM</schema>
              <schemaversion>1.2</schemaversion>
              <adlcp:location>imsmetadata.xml</adlcp:location>
       </metadata>

  <organizations default="pipwerks">
    <organization identifier="pipwerks">
    <title>SCORM Kolmefaasiline ahel</title>
      <item identifier="CourseItem01" identifierref="SCO-Resource-01" isvisible="true">
      <title>Kirjutage vastused lahtritesse ja kontrollige tulemusi.</title>
      </item>
    </organization>
  </organizations>

  <resources>
    <resource identifier="SCO-Resource-01" type="webcontent" adlcp:scormtype="sco"
href="index.html">
      <file href="index.html"/>
      <file href="/libs/SCORM_API_wrapper.js"/>
    </resource>
  </resources>

</manifest>
```

Koodist võime välja lugeda, et kasutame SCORM versiooni 1.2, Moodle ülesande tiitel on "Kolmefaasiline ahel", ülesande avab index.html (peab asuma samas folderis), Moodle ning .html faili vahel suhtleb SCORM_API_wrapper.js (laetud alla pipwerks veebilehelt).



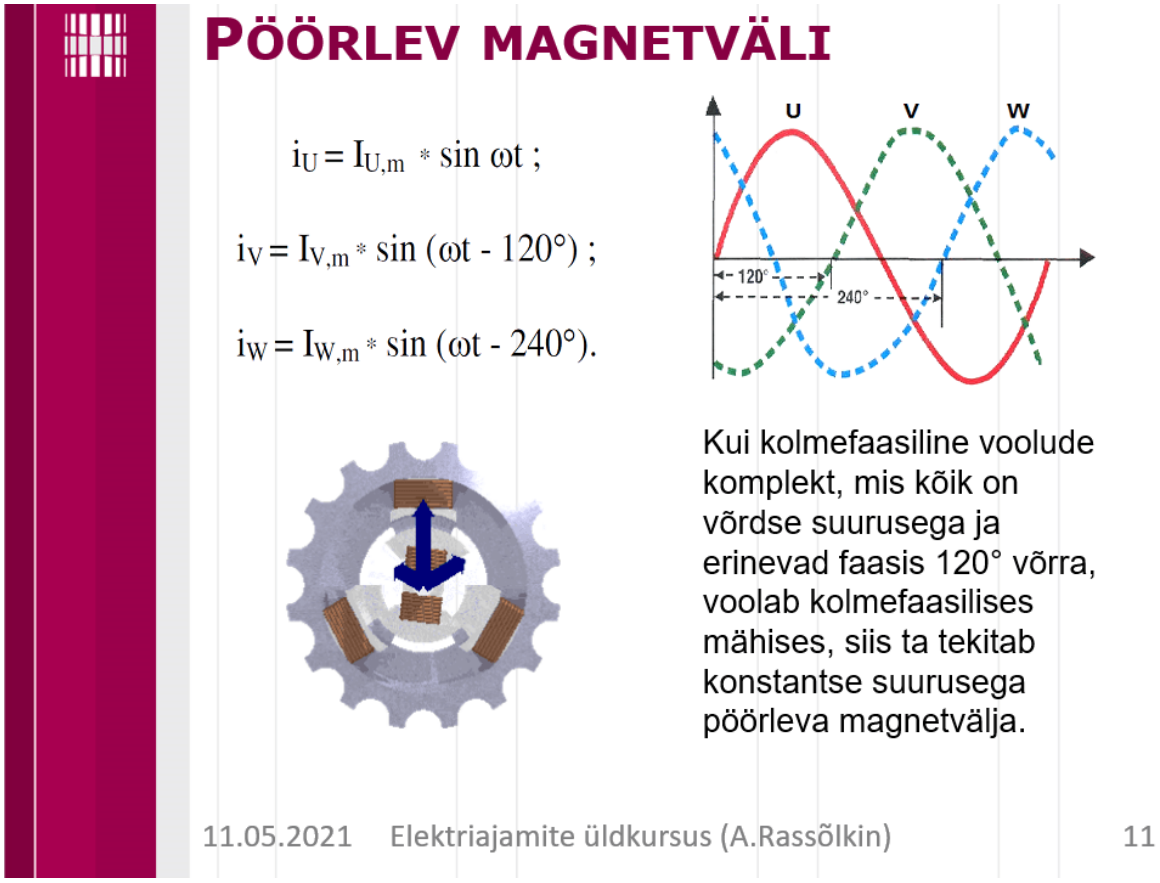
Joonis 9. SCORM-API asukoht ja globaalne muutuja

4 KOLMEFAASILISE ElektriTSIOONMOOTORI ÜLESANNE

Õppejõud palus proovida koostada kolmefaasiline elektrimootori interaktiivne ülesanne. Mootori tööprotsess animeerida ja võimalusel muuta parameetreid nagu võrgusagedus ja faasijärjestus.

4.1 Ülesande koostamine

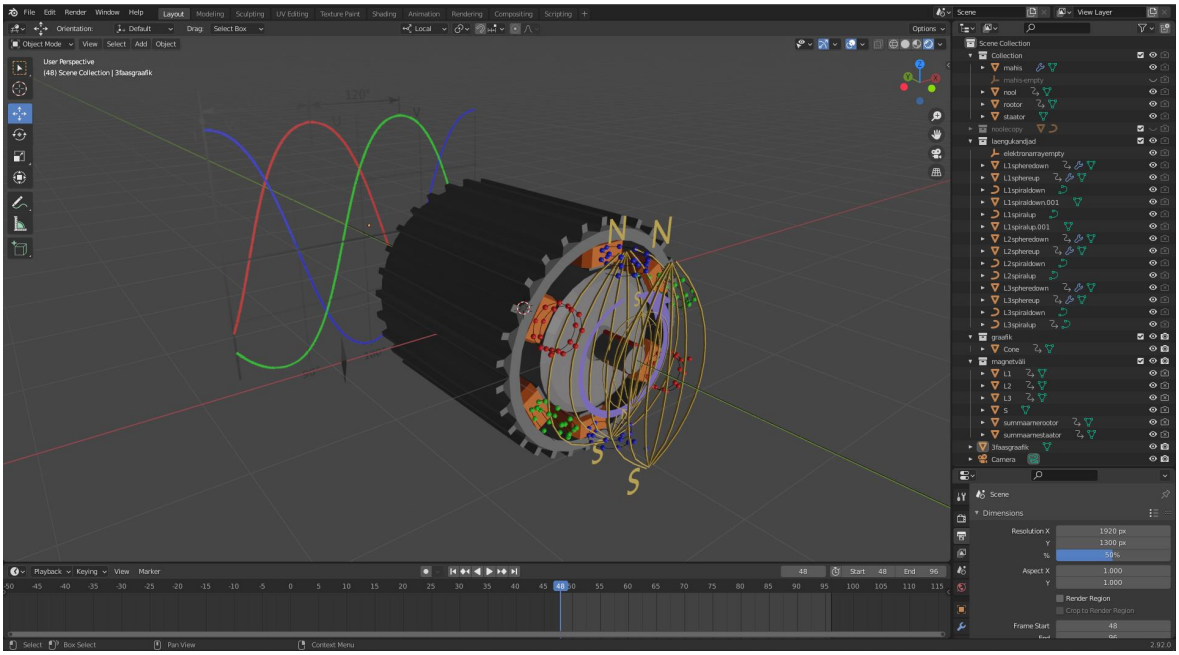
Ülesanne põhines „Pöörlev magnetväli" joonisel 9, kus on välja toodud faaside graafik, mootor ja animeeritud rootor. Antud näite põhjal oli vaja koostada interaktiivne õppematerjal. Annan näite oma ülesande esimesest versioonist, kasutades referentsiks erinevaid õppematerjale[12, 13].



Joonis 10. Pöörlev magnetväli

4.2 Animatsioonid Blenderis

Ülesande lahendamiseks on mitmeid variante, mille määravad ära koostaja kunstilised ja programmeerimisoskused. Kuna ülesanne oli seotud rohkem animeerimisega ja vähem programmeerimisega, otsustasin kasutada endale tuttavat programmi Blender, mis on vabavaraline 3D mudelite ja animatsioonide koostamise programm. Meie ülesande jaoks on vajalik välja renderdada eraldiseisvate animatsioonielementide kaadrid. Animatsioonielemendiks on näiteks laengute liikumine mähisel, magnetvälja muutumine, rootori pöörlemine jne. Elemendi kaadril on kujutatud ainult uuritavat objekti nähtamatul taustal PNG failina. PNG failid on tugevasti kompresseeritavad, mille ühe kaadri suurus on keskmisel 100kb. Kokku oli 350 kaadrit. Kaadrisageduse seadistasin 48 kaadrit ühe sekundi peale, mis vastab kahekordsele veebibrauseri animatsiooni mängimissageduse kordajale. Suurem kaadrite arv lubab ka animatsiooni aeglustamisel säilitada kvaliteeti. Kindlasti on võimalik animatsioone teha muude programmidega, isegi Adobe Animate endaga, kõik oleneb koostaja oskustest.



Joonis 11. Elektrimootori 3D mudel Blenderis

4.3 Animatsioonide sidumine Animate’is

Koostatud kaadrid on imporditud Adobe Animate’i. Iga iseseisev element on kihil välja toodud, kihi nimetused on vastavuses skriptis olevate elementide nimetustega[3]. Tähtis on arvestada, et animatsiooni element ei ole video, vaid veebibrauseril liikuv piltide järjestikune kogumik, mida nimetatakse tweeniks. Kui testida faili, töötavad animatsioonid. Ülesande interaktiivseks tegemiseks lisan nupud, millega saab muuta animatsioonielemente[6, 7]:

1) Nool – näitab ja peidab rootori ringliikumist paremini kujutavat noolt.

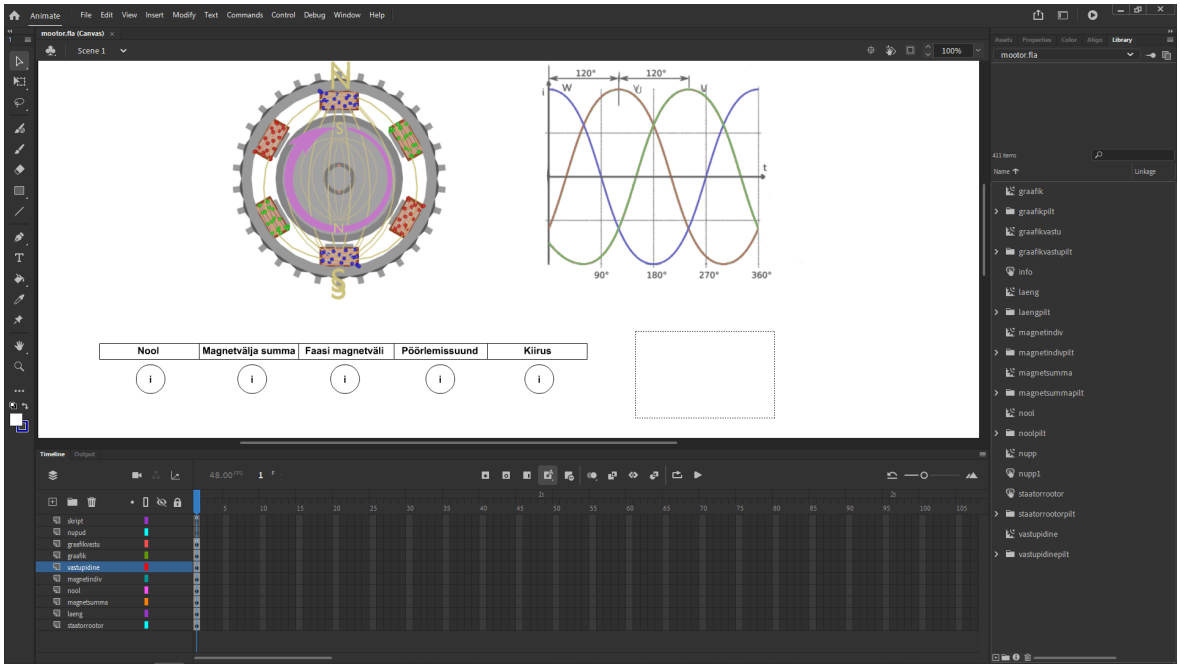
2) Magnetvälja summa – teeb nähtavaks staatori ja rootori summaarsed ringliikuvad magnetväljad ning mähiste laengute liikumised.

3) Faasi magnetväli – teeb nähtavaks staatori individuaalsed ringliikuvad ja polaarsust vahetavad magnetväljad ning mähiste laengute liikumised.

4) Kiirus – lubab aeglustada animatsiooni mängimise kiirust.

5) Pöörlemissuund – vahetab omavahel kahe faasi järjestuse, mis paneb rootori teisipidi tööle.

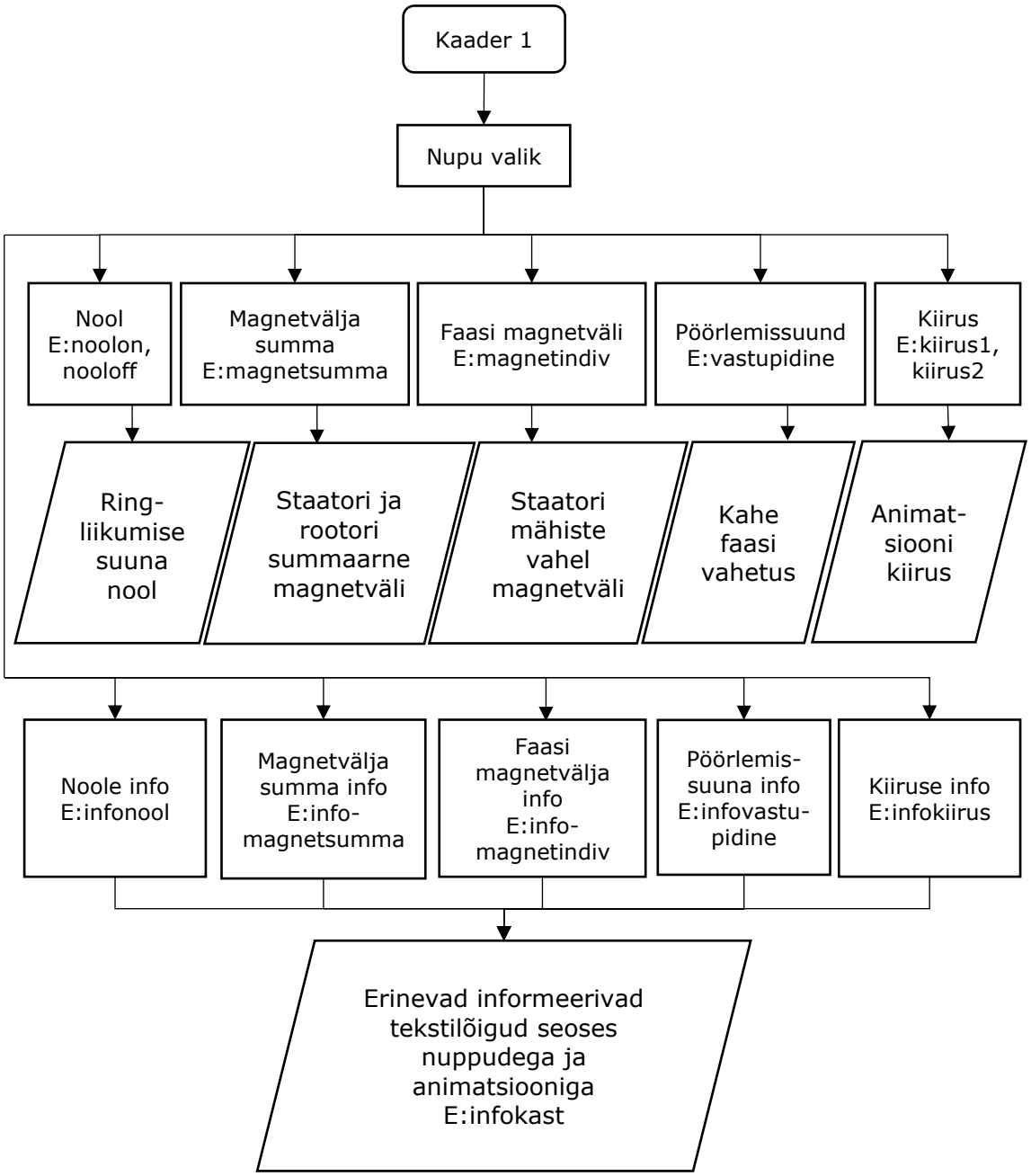
6) Infonupp interaktiivsete nuppude all – annab lisa infot nuppude tegevusest ja kujutatud mootori animatsioonidest.



Joonis 12. Elemendid Adobe Animate’is

4.4 Animatsiooni plokkskeem

Sarnaselt ülesandega „Rööptakistus" koostame plokkskeemi võimalikest interaktiivsetest elementidest. Elementide nimetused on tähistatud sõnaga „E". Töötav JavaScripti kood on esitatud lisas 4.



Joonis 13. Kolmefaasiline elektritsioonmootori töö animatsiooni plokkskeem

5 HTML5 FAILIDE LISAMINE MOODLE'I

Koostatud HTML5 failid saame publitseerida, kui valime File > Publish Settings > Publish. Publish Settingute all saame muuta erinevaid parameetreid seoses publitseerimisega. Standardina on meie HTML5 failide seas:

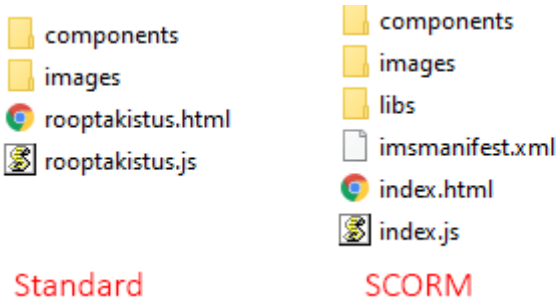
1) Components kaust – scriptid, mis on seotud tekstikastidega.

2) Images kaust – elementide joonised.

3) Failinimi.js – peamine skripti fail, mis seob omavahel elementide interaktsioone.

4) Failinimi.html – veebibrauseri fail, mis on veebilehekülje vaatamiseks.

5) SCORM paketile lisandub veel manifest.xml ja libs kaust, kus on API.



Joonis 14. Publitseeritud HTML failid

Kui avada fail .html, tehakse lahti veebibrauseri aken. Veebibrauser ei pruugi näidata faili sisu, sest viidatud skriptid ja elemendid asuvad arvutis endas, turvalisuse tõttu on brauserid blokeerinud sellise tegevuse. Failid peavad olema viidatud ja avatud Moodle'is eneses. Tehtud töö üleslaadimiseks teeme publitseeringust .zip faili. Valime „Lisa tegevus või ressurss", milleks on kaks võimalust.

1) Ressurss „Fail" – laeme ülesse .zip faili. Vajutame .zip failile ja pakime ta lahti. Valida .html fail ning vajutada „Määra peamine fail".

2) Ressurss „SCORM pakett" – laeme ülesse .zip faili SCORMist. Edasine tegevus pole vajalik, manifest fail paigutab ise kõik paika.

KOKKUVÕTE

Käesolevas töös on uuritud, kuidas koostada interaktiivseid õppematerjale ja ülesandeid õppeainele „Elektriahelad". Standardsed õppematerjalid on valdavalt staatilise teksti-, jooniste- ja valemitepõhine. Keeruliste õppeainete teadmiste paremaks omandamiseks on kasulik koostada õppematerjalid, kus tudeng saab harjutusega interaktiivselt suhelda. Interaktiivsus on inimese ja arvutitarkvara vaheline suhtlusprotsess, mis näiteks väljenduvad kujul - hiirega klõpsatavad nupud, sisestused tekstikasti või muunduvad joonised.

Nende keerukate õppematerjalide loomiseks kasutati ajakohast, laialt levinud ja suhteliselt hõlpsasti kasutatavat programmeerimiskeelt - JavaScript. Siduvaks struktuuriks on valitud HTML5 platvorm.

Lõpuutööks tuli omandada JavaScripti tasemel, mis lubaks koostada ja teisendada nüüdseks vananenud ActionScripti baasil harjutusülesanded. Abi oli programmist Adobe Animate, millega sai keskenduda valdavalt skripti koostamisele. Adobe Animate on tuttav inimestele, kes on koostanud Flashi põhiseid faile.

JavaScripti harjutamiseks ja ülesande interaktiivsuse paremaks nägemiseks on „Elektrotehnika I" õppeainest ümber teisandatud seitse interaktiivset harjutust. Lõputöös näiteks toodud kaks harjutusülesannet katavad võimalikud probleemide lahendused – Flashi dekodeerimine, elementidest arusaamine, skripti kirjutamine, skripti teisendamine ühest keelest teise ja publitseerimine.

Lõputöö aitab neid, kes soovivad koostada interaktiivseid õppematerjale. Koodi kirjutamise õppimine võib alguses olla keeruline, kuid lõpptulemused on seda väärt.

Lõputöö eesmärgid on edukalt saavutatud.

Bakalaureusetöö on kirjutatud TalTech elektrotehnika instituudi vanemlektori Aleksander Kilk juhendamisel. Lõputöö teema ja harjutusmaterjalid on pakutud juhendaja poolt.

SUMMARY

In my dissertation I studied how to compile interactive educational materials and tasks for the subject Electrical-circuits. Standard educational materials are mostly based on static text, drawings and formulas. In order to better acquire knowledge of complex subjects, it is useful to create study materials where the student can interact with the exercise. Interactivity is the process of communication between a person and computer software, which is expressed, for example, in the form of cllickable buttons, text box inputs or transforming drawings.

Creating these complex study materials was done by using an uptodate, widespread and relatively easy to use programming language - JavaScript. The HTML5 platform has been chosen as the binding structure.

The dissertation required learning of JavaScript to a level that would allow the creation and conversionbased on now obsolete ActionScript-based exercises. A great help was the program Adobe Animate, which allowed to focus mainly on writing scripts. Adobe Animate is familiar to people who have created Flash-based files.

In order to practice JavaScript and better see the interactivity of the task, seven interactive exercises have been converted from the subject "Electrical Engineering". The two example exercises in the thesis cover possible solutions to problems - decoding Flash, understanding elements, writing a script, converting a script from one language to another and publishing.

The dissertation helps those who want to create interactive study materials. Learning to write code can be difficult at first, but the end results are worth it.

The goals of the dissertation have been successfully achieved.

The bachelor's thesis was written under the supervision of Aleksander Kilk, Senior Lecturer at the TalTech Institute of Electrical Engineering. The topic of the dissertation and the exercise materials were acquired by the supervisor.

KASUTATUD KIRJANDUSE LOETELU

[1] TalTech Moodle AME3140 Elektrotehnika I. „Rööptakistus", „Voolutugevus", „Võimsus", „Pingeresonantsi tekkimine", „Neliklemmi uurimine", „Kolmefaasiline ahel". [Internet] https://moodle.taltech.ee/course/view.php?id=1210

[2] „HTML Tutorial for Beginners: HTML5 Crash Course [2021]" [Internet] https://www.youtube.com/watch?v=qz0aGYrrlhU&t=3657s&ab_channel=ProgrammingwithMosh

[3] „Intro to Adobe Animate 2020 [1/4] Beginners Tutorial" [Internet] https://www.youtube.com/watch?v=7huMYp7WpsI&t=2s&ab_channel=TipTut

[4] „Learn JavaScript – Full Course for Beginners" [Internet] https://www.youtube.com/watch?v=PkZNo7MFNFg&ab_channel=freeCodeCamp.org

[5] „JavaScript Tutorial for Beginners – Full Course in 8 Hours [2020]" [Internet] https://www.youtube.com/watch?v=Qqx_wzMmFeA&ab_channel=CleverProgrammer

[6] „How to make HTML5 GAME in Adobe Animate CC 2019 – 1 hour tutorial" [Internet] https://www.youtube.com/watch?v=wcKzqy_71vA&t=926s&ab_channel=MotionTuts

[7] „Animate CC: Interactive HTML5 Canvas Projects" [Internet] https://www.youtube.com/watch?v=ViDw06N96MQ&ab_channel=JosephLabrecque

[8] W3 Schools Learn JavaScript [Internet] https://www.w3schools.com/js/default.asp

[9] CreateJavaScript Easel JS [Internet] https://www.createjs.com/easeljs

[10] SCORM koodi käsud [Internet] https://scorm.com/scorm-explained/technical-scorm/run-time/run-time-reference/?utm_source=google&utm_medium=natural_search#section-2.

[11] Pipwerks. „SCORM API Wrapper" [Internet] https://pipwerks.com/laboratory/scorm/api-wrapper-javascript/

[12] „How Electric Motors Work – 3 phase AC induction motors as ac motor" [Internet] https://www.youtube.com/watch?v=59HBoIXzX_c&t=274s&ab_channel=TheEngineeringMindset

[13] A. Bruce Carlson. „Circuits. Engineering Concepts and Analysis of Linear Electric Circuits" [Raamat]

LISAD

Lisa 1. ActionScripti lugemine. Harjutus „Rööptakistus“. Koodi ja kommentaaride lugemine annab parema arusaamise, kuidas teisendada kood JavaScripti.

```
stop();          //peatab mängimise esimesel kaadril
r2._visible = 0; //peidab kujundi elemendid r2, r3, r4
r3._visible = 0;
r4._visible = 0;
sinupp._visible = 1; //näitab elemendid sinupp, kontroll ja jareldus
kontroll._visible = 1;
jareldus._visible = 1;
r3.r3 = "";    //annab elemendile r3 (tekstikast) tühja väärtuse.
r2.r2 = "";    //r3/r2/r4 tekstikast asub omakorda r3/r2/r4 kujundis
r4.r4 = "";
jah._visible = 0; //peidab kujundi elemedid jah, ei, vastusR, vastus C, tulemus ja tulemusC
ei._visible = 0;
vastusR._visible = 0;
vastusC._visible = 0;
tulemus._visible = 0;
tulemusc._visible = 0;
sinupp = "Kontroll"; //annab teksikasti elementidele vastavad teksti
jarel = "Mida siit \rjäreldada?";
ronupp = "Uus skeem";
vas = ""; //tekstikast vas on ilma väärtuseta
auk._visible = 1; //peidab või näitab nupuelemdid
auk1._visible = 0;
aukp2._visible = 1;
aukp._visible = 0;
jarel._visible = 0;
abi = 1; //muutujad tulevaste matemaatiliste arvutuste jaoks
nr = 0;
edasi.onPress = function() //vajutades nuppu edasi, läheb järgmisse kaadrisse, kus arvutatakse mahtuvust
{
  gotoAndStop(2);
};
rojarg.onPress = function() //vajutades nuppe rojarg, täidetakse järgmised käsud
{
  jarel = "Mida siit \rjäreldada?"; //tekstikast jarel saab väärtuse
  tulemus._visible = 1; //toob nähtavale
  vastusR._visible = 1;
  vastus = "Arvutage ekvivalentne takistus. \rAndke kaks kohta peale punkti."; //tekstikast vastus
  nr = nr + 1;              //muutuja nr, mis oli algselt null väärtusega, on nüüd väärtus 1
  jareldus.onPress = function()//vajutades nuppu jareldus, teeb järgmised
  {
    if(nr > 3)//kui nr vaartus on üle kolme, siis
    {
      if(Rb == 0 || Rc == 0 || Rd == 0 || Ra == 0)//operant "või", kui üks takistuste väärtustest
      {
        vastus = "Kui üks takistus on null\rsiis on kogutakistus null."; //kui üks väärtus 0, siis vastus on
      }
      else if(abi == 1)//kui abi on väärtusega 1
      {
        vastus = "Võrdsete takistuste korral\rväheneb takistus n korda.";
      }
      else//kui ei vasta mõlemale eelmise tingimusele
      {
        vastus = "Mitteõrdsete takistuste korral\ron kogutakistus alati väiksem\rkui kõige väiksem takistus.";
      }
    }
  };
  Ra = random(29);//takistuste väärtused valitakse suvaliselt 0 kuni 29
  Rb = random(9);
```

```
Rc = random(19);
Rd = random(15);
k = random(3) + 1;
auk1.onPress = function()//vajutades nuppu auk1
{
  auk1._visible = 0;
  auk._visible = 1;
  aukp2._visible = 1;
  aukp._visible = 0;
  abi = 1;
};
aukp2.onPress = function()//vajutades nuppu auk2p
{
  aukp2._visible = 0;
  aukp._visible = 1;
  auk._visible = 0;
  auk1._visible = 1;
  abi = 0;
};
if(abi == 1)                //vajutades nuppu auk 1, seab abi väärtuse 1, seega kõik takistused on võrdsed Ra
{
  Rb = Ra;
  Rc = Ra;
  Rd = Ra;
}
r4.r41 = "= " + Ra + " Ω"; //tekstikastile r41 väärtuse andmine, tekstikast asub elemendis r4
r4.r42 = "= " + Rb + " Ω"; //tekstikastile antake Rb väärtus, Rb on muutuja
r4.r43 = "= " + Rc + " Ω";
r4.r44 = "= " + Rd + " Ω";
tulemus.oom = "Ω";
r3.r31 = "= " + Rb + " Ω";
r3.r32 = "= " + Rc + " Ω";
r3.r33 = "= " + Rd + " Ω";
r2.r21 = "= " + Rc + " Ω";
r2.r22 = "= " + Rd + " Ω";
if(k === 1) //k genereeritakse suvaliselt ja määrab ära takistuste arvu
{
  r2._visible = 1; //2 takistust
  r3._visible = 0;
  r4._visible = 0;
  if(Rc == 0 || Rd == 0)
  {
    ra = 0;
  }
  else
  {
    ra = Math.round(100 * (rc * rd) / (rc + rd)) / 100; //tulemuse ümardamine
  }
}
if(k === 2)
{
  r2._visible = 0;
  r3._visible = 1; //3 takistust
  r4._visible = 0;
  if(Rb == 0 || Rc == 0 || Rd == 0)
  {
    ra = 0;
  }
  else
  {
    ra = Math.round(100 * 1 / (1 / rb + 1 / rc + 1 / rd)) / 100;
  }
}
if(k === 3)
{
  r2._visible = 0;
  r3._visible = 0;
```

```
126     r4._visible = 1; //4 takistust
127     if(Ra == 0 || Rb == 0 || Rc == 0 || Rd == 0)
128     {
129       ra = 0;
130     }
131     else
132     {
133       ra = Math.round(100 * 1 / (1 / ra + 1 / rb + 1 / rc + 1 / rd)) / 100;
134     }
135   }
136   vas = ra; //väikene spikker, mis ütleb takistuste koguväärtuse
137 };
138 onEnterFrame = function() //kui kaadris toimub mingi muudatus, siis
139 {
140   if(isFinite(tulemus.rtul) )//kui teksti sisestuskast rtul on lõplik
141   {
142     rr = tulemus.rtul; //muutuja rr, mille väärtuse sisestame teksti sisestuskasti rtul
143   }
144   kontroll.onPress = function() //vajutades nuppu kontroll
145   {
146     if(Math.abs(ra - rr) / (rr + 0.000001) < 0.05)
147     { //kui meiepoolne vastus rr erineb vähesel määral õigest vastusest ra
148       ei._visible = 0;
149       jah._visible = 1;
150       jah.gotoAndPlay(1); //siis vastus õige ja mängi animatsiooni jah
151     }
152     else
153     {
154       jah._visible = 0;
155       ei._visible = 1;
156       ei.gotoAndPlay(1); //siis vastus vale ja mängi animatsiooni ei
157     }
158   };
159 };
```

Lisa 2. ActionScripti lugemine. Harjutus „Kolmefaasiline ahel".

Koodi lugemine annab parema arusaamise, kuidas teisendada JavaScripti. Kood on oluliselt lühendatud, ära on võetud 23 ülesande varianti ja 6 vastuse kontrolli.

Kaader 1:

```
1 stop(); //peatab animatsiooni
2 Selection.setFocus("kood"); //variandi valik esimesel kaadril
3 alg.onRelease = function() {//Scene1.nr=nnr;
4                 if(isFinite(varnr)){nr=varnr};
5 if(nr>>0){nextScene();}  //kui variandi number sobib, siis järgmine stseen
6 else {Selection.setFocus("kood");
7                 ei.gotoAndPlay(1); } //kui variandi number ei sobi, siis mängi ei sobi
8 }
```

Kaader 2:

```
1   stop();
2   import pipwerks.SCORM; //SCORM API import
3   import flash.external.ExternalInterface;
4   var scorm: SCORM = new SCORM(); //SCORM parameetrid
5   var course: Object = {
6                   lesson_status: "",
7                   student_name: "",
8                   student_id: "",
9                   currentPage: 0,
10                  totalPages: 3
11  }
12                  function init(): Void { //tulemuste saatmine Moodle
13                          var success: Boolean = scorm.connect();
14                          if (success) {
15                                  success = scorm.set("cmi.core.score.raw", hind);
16                                  success = scorm.set("cmi.core.lesson_status", "completed");
17                                  if (success) {
18                                          scorm.save();
19                                  }
20                                  success = scorm.disconnect();
21                                  //exitCourse()
22                          }
23                  }
24                  function exitCourse(): Void {
25                          ExternalInterface.call("window.close");
26                  }
27  //kui ülesande tulemused sobivad, tekib nupp ja saab tööle panne funktsioon init, mis saadab
28  saatja.onPress = function () {
29                  hind = nr;
30                  saatja._visible = 0;
31                  vastamine._visible = 0;
32                  laks._visible = 1;
33                  laks.gotoAndPlay(1);
34
35                  init();
```

```
36 }
37 //muutujate väärtused ja elementide peitmine/näitamine
38 saak1 = 0;
39 saak2 = 0;
40 saak3 = 0;
41 saak4 = 0;
42 saak5 = 0;
43 saak6 = 0;
44 saak7 = 0;
45 juheL._visible = 1;
46 juheC._visible = 0;
47 tahtL._visible = 0;
48 tahtC._visible = 1;
49 deltaL._visible = 0;
50 deltaC._visible = 0;
51 kysdelta._visible = 0;
52 kystaht._visible = 1;
53 ubc._visible = 0;
54 uab._visible = 1;
55 uca._visible = 0;
56 vastus._visible = 0;
57 vastamine._visible = 0;
58 laks._visible = 0;
59 lapsb._visible = 0;
60 lapsc._visible = 0;
61 auk = 0;
62 vastusd._visible = 0;
63 saatja._visible = 0;
64 onn = 0;
65 valmis._visible = 0;
66 auka = 0;
67 kirje._visible = 0;
68 tekst._visible = 0;
69 abi._visible = 1; //nr=0;
70 //numb=nr;
71 //eelarvutused
72 r1 = 1 + Math.round(2 + (2 * nr + 30 / nr) / 10);
73 r2 = Math.round(50 + (2 * nr + 40 / nr) / 1);
74
75 L1 = Math.round(1 + (2 * nr + 10 / nr) / 20) / 100;
76 L2 = Math.round(4 + (2 * nr + 50 / nr) / 4) / 100;
77
78 c1 = Math.round(50 + (5 * nr + 30 / nr) / 3);
79 c2 = Math.abs(Math.round((60 - 0.3 * nr) / 1));
80 //
81 juheL.inda = "= " + L1 + " H";
82 juheL.indb = "= " + L1 + " H";
83 juheL.indc = "= " + L1 + " H";
84 juheL.indar = "= " + r1 + " \u03A9";
85 juheL.indbr = "= " + r1 + " \u03A9";
86 juheL.indcr = "= " + r1 + " \u03A9";
87 tahtL.inda = "= " + L2 + " H";
88 tahtL.indb = "= " + L2 + " H";
89 tahtL.indc = "= " + L2 + " H";
90 tahtL.indar = "= " + r2 + " \u03A9";
91 tahtL.indbr = "= " + r2 + " \u03A9";
92 tahtL.indcr = "= " + r2 + " \u03A9";
```

```
93  //variandi valik
94  if (nr <= 24) {
95                      aa = nr}
96  if (nr > 24 && nr < 49) {
97                      aa = nr - 24}
98  if (nr > 48 && nr < 73) {
99                      aa = nr - 48}
100 if (nr > 72 && nr < 97) {
101                     aa = nr - 72}
102 if (nr > 96 && nr < 121) {
103                     aa = nr - 96}
104 if (nr > 121) {
105                     aa = 12}
106 //arvutused
107 f = 50;
108 u = 230; //u=400/Math.sqrt(3);
109 ubgr = u * Math.cos(4 * Math.PI / 3);
110 ubgx = u * Math.sin(4 * Math.PI / 3);
111 ucgr = u * Math.cos(2 * Math.PI / 3);
112 ucgx = u * Math.sin(2 * Math.PI / 3);
113 uabgr = 400 * Math.cos(1 * Math.PI / 6);
114 uabgx = 400 * Math.sin(1 * Math.PI / 6);
115 ubcgr = 400 * Math.cos(3 * Math.PI / 2);
116 ubcgx = 400 * Math.sin(3 * Math.PI / 2);
117 ucagr = 400 * Math.cos(5 * Math.PI / 6);
118 ucagx = 400 * Math.sin(5 * Math.PI / 6);
119 xl1 = 2 * Math.PI * f * L1;
120 xl2 = 2 * Math.PI * f * L2; //reaktiivtakistused
121 xc1 = 1 / (2 * Math.PI * f * c1 * 0.000001);
122 xc2 = 1 / (2 * Math.PI * f * c2 * 0.000001);
123 rcj = r1 * xc1 * xc1 / (r1 * r1 + xc1 * xc1);
124 xcj = -r1 * r1 * xc1 / (r1 * r1 + xc1 * xc1); //liini roopC
125 rck = r2 * xc2 * xc2 / (r2 * r2 + xc2 * xc2);
126 xck = -r2 * r2 * xc2 / (r2 * r2 + xc2 * xc2); //koormuse roopC
127
128 onEnterFrame = function () {
129 //ülesande variant 1
130                     if (aa == 1) {
131                                         juheL._visible = 1;
132                                         juheC._visible = 0;
133                                         lops = 1;
134                                         tahtL._visible = 1;
135                                         tahtC._visible = 0;
136                                         deltaL._visible = 0;
137                                         deltaC._visible = 0;
138                                         v = "ab"
139                                         kysdelta._visible = 0;
140                                         kystaht._visible = 1;
141                                         ubc._visible = 0;
142                                         uab._visible = 1;
143                                         uca._visible = 0;
144                                         zar = r1 + r2;
145                                         zax = xl1 + xl2;
146                                         za = zar * zar + zax * zax;
147                                         if (auk == 0) {
148                                                             iar = u * zar / za;
149                                                             iax = -u * zax / za; // kolmfaasilise ahela arvutus
```

```
150                                     ia = Math.sqrt(iar * iar + iax * iax);
151                                     ibr = (ubgr * zar + ubgx * zax) / za;
152                                     ibx = (ubgx * zar - ubgr * zax) / za;
153                                     ib = Math.sqrt(ibr * ibr + ibx * ibx);
154                                     uar = iar * r2 - iax * xl2;
155                                     uax = iar * xl2 + iax * r2;
156                                     ua = Math.sqrt(uar * uar + uax * uax);
157                                     ubr = ibr * r2 - ibx * xl2;
158                                     ubx = ibr * xl2 + ibx * r2;
159                                     ub = Math.sqrt(ubr * ubr + ubx * ubx);
160                                     ulr = uar - ubr;
161                                     ulx = uax - ubx;
162                                     ul = Math.sqrt(ulr * ulr + ulx * ulx);
163                                     p = 3 * ia * ia * zar;
164                             }
165                             //katkise ahela arvutus
166                             else {
167                                     zaa = 4 * za;
168                                     iar = (ucagr * 2 * zar + ucagx * 2 * zax) / zaa;
169                                     iax = (ucagx * 2 * zar - ucagr * 2 * zax) / zaa;
170                                     ia = Math.sqrt(iar * iar + iax * iax);
171                                     uar = iar * r2 - iax * xl2;
172                                     uax = iar * xl2 + iax * r2;
173                                     ua = Math.sqrt(uar * uar + uax * uax);
174                                     ulr = uar;
175                                     ulx = uax;
176                                     ul = Math.sqrt(ulr * ulr + ulx * ulx);
177                                     p = 2 * ia * ia * zar;
178                             }
179                     }
180 }
181
182 //tulemuste kontroll, esimene lahter, teised arvutused on sarnased, ei lisa teisi
183 kont1.onRelease = function () { // vastuste kontroll
184                     if (onn > 0) {
185                             kirje._visible = 1;
186                     }
187                     auka = 0;
188
189                     if (isFinite(evas)) {
190                             vool1 = 1 * evas;
191                             if ((Math.abs((vool1 - ia) / (ia + 0.0001))) <= 0.03) {
192                                     tulem1 = "Sobib, sest σ < 3%.";
193                                     saak1 = 1; //õige vastus, tolerantside vahel
194                             } else {
195                             if (Math.abs((vool1 - ia) / (ia + 0.0001)) > 0.03 && Math.abs((vool1 - ia) / (ia + 0.0001)) <= 0.1) {
196                                             tulem1 = "Ei sobi, sest σ > 3%.";
197                                             saak1 = 0; //vale vastus
198                                     }
199                             if (Math.abs((vool1 - ia) / (ia + 0.0001)) > 0.1 && Math.abs((vool1 - ia) / (ia + 0.0001)) <= 0.5) {
200                                             tulem1 = "Ei sobi, sest σ > 10%.";
201                                             saak1 = 0; //vale vastus
202                                     }
203                                     if (Math.abs((vool1 - ia) / (ia + 0.0001)) > 0.5) {
204                                             tulem1 = "Ei sobi, sest σ > 50%.";
205                                             saak1 = 0; //vale vastus
206                                     }
```

```
207                                 }
208                 } else {
209                                 tulem1 = "Sisestage vastus."
210                 }
211
212 //originaalis, kui kõik 7 vastust õiget, tekib nupp valmis
213                 saak = saak1 + saak2 + saak3 + saak4 + saak5 + saak6 + saak7;
214                 if (saak == 7) {
215                                 valmis._visible = 1;
216                                 teade._visible = 0;
217                 } else {
218                                 valmis._visible = 0;
219                                 teade._visible = 1;
220                 }
221
222 }
223 //vajutades nuppu valmis
224 valmis.onRelease = function () {
225                 onn = onn + 1;
226                 if (onn == 1) {
227                                 valmis._visible = 0;
228                                 saatja._visible = 1;
229                                 vastamine._visible = 1;
230                                 kont1._visible = 0;
231                                 tulem1 = "";
232                                 tulem2 = "";
233                                 tulem3 = "";
234                                 tulem4 = "";
235                                 tulem5 = "";
236                                 tulem6 = "";
237                                 tulem7 = "";
238                                 auk = 0;
239                 } else {//mängib animatsiooni
240                                 valmis._visible = 0;
241                                 if (lops == 1) {
242                                                 lapsb._visible = 1;
243                                                 lapsb.gotoAndPlay(1);
244                                 } else {
245                                                 lapsc._visible = 1;
246                                                 lapsc.gotoAndPlay(1);
247                                 }
248                                 if (aa > 12) {
249                                                 vastusd._visible = 0;
250                                 } else {
251                                                 vastus._visible = 0;
252                                 }
253
254                 }
255 }
256 abi.onRelease = function () { // abi tekst lahti
257                 tekst._visible = 1;
258                 abi._visible = 0;}
259 tekst.sulge.onRelease = function () { // abi tekst kinni
260                 tekst._visible = 0;
261                 abi._visible = 1;}
```

Lisa 3. JavaScripti lugemine. Harjutus „Kolmefaasiline ahel".

Kaader 1:

```
this.stop();
var varnr; //muutuja deklareerimine

function fl_MouseClickHandler_1() {
	varnr = document.getElementById("kood"); //tekstisisestuskast
	nr = Number(varnr.value); //numbriline väärtus tekstisisestuskastist

	if (nr > 0) { //number peab olema suurem kui 0
		if (!Number.isInteger(nr)) { //number peab olema täisarv
			alert("Sisesta täisarv");
			return;
		}
		this.gotoAndStop(1); //kui sisestatud number sobib, siis järgmine kaader
	} else { //vastasel juhul sama kaader ja mägi animatsioon ei
		this.ei.gotoAndPlay(0);
	}
}
this.alg.addEventListener("click", fl_MouseClickHandler_1.bind(this));
```

Kaader 2:

```
var root = this;

function complete() {
	var success = pipwerks.SCORM.init()
	if (success) {
		success = pipwerks.SCORM.set("cmi.core.score.raw", hind);
		success = pipwerks.SCORM.set("cmi.core.lesson_status", "completed");
		if (success) {
			pipwerks.SCORM.save();
		}
		success = pipwerks.SCORM.quit();
		//exitCourse()
	}
}

function exitCourse() {
	ExternalInterface.call("window.close");
}
//tulemuste saatmine
function saatmine() {
	hind = nr;
	root.saatja.visible = 0;
	root.vastamine.visible = 0;
	root.laks.visible = 1;
	root.laks.gotoAndPlay(0);

	complete(); //aktiveerib funktsioon complete
}

root.saatja.addEventListener("click", saatmine);

var aa;
var hind;
var saak1 = 0;
var saak2 = 0;
var saak3 = 0;
var saak4 = 0;
var saak5 = 0;
var saak6 = 0;
var saak7 = 0;
root.juheL.visible = 1;
root.juheC.visible = 0;
root.tahtL.visible = 0;
root.tahtC.visible = 1;
```

```
45  root.deltaL.visible = 0;
46  root.deltaC.visible = 0;
47  root.kysdelta.visible = 0;
48  root.kystaht.visible = 1;
49  root.ubc.visible = 0;
50  root.uab.visible = 1;
51  root.uca.visible = 0;
52  root.vastamine.visible = 0;
53  root.laks.visible = 0;
54  root.lapsb.visible = 0;
55  root.lapsc.visible = 0;
56  var auk = 0;
57  root.saatja.visible = 0;
58  var onn = 0;
59  root.valmis.visible = 0;
60  var auka = 0;
61  root.tekst.visible = 0;
62  root.abi.visible = 1; //nr=0;
63
64  //uued muutujad onenterframe kastis
65  var lops;
66  var v;
67  var zar;
68  var zax;
69  var za;
70  var iar;
71  var iax;
72  var ia;
73  var ibr;
74  var ibx;
75  var ib;
76  var uar;
77  var uax;
78  var ua;
79  var ubr;
80  var ubx;
81  var ub;
82  var ulr;
83  var ulx;
84  var ul;
85  var p;
86  var zaa;
87  var aa;
88  var icr;
89  var icx;
90  var ic;
91  var ucr;
92  var ucx;
93  var uc;
94  var zf;
95  var r1;
96  var uer;
97  var uex;
98  var ue;
99  var ufr;
100 var ufx;
101 var uf;
102 var zdr;
103 var zdx;
104 var xl2k;
105 var zer;
106 var zex;
107 var ze;
108 //vastuste kast
109 var vool1;
110 //uued muutujad onenterframe kastis lõpp
111 var evas;
112 var evasval;
113
114 //convertedto html5
115 //numb=nr;
116 var r1 = 1 + Math.round(2 + (2 * nr + 30 / nr) / 10);
117 var r2 = Math.round(50 + (2 * nr + 40 / nr) / 1);
118
119 var L1 = Math.round(1 + (2 * nr + 10 / nr) / 20) / 100;
```

```
120 var L2 = Math.round(4 + (2 * nr + 50 / nr) / 4) / 100;
121
122 var c1 = Math.round(50 + (5 * nr + 30 / nr) / 3);
123 var c2 = Math.abs(Math.round((60 - 0.3 * nr) / 1));
124
125 root.juheL.inda.text = "= " + L1 + " H";
126 root.juheL.indb.text = "= " + L1 + " H";
127 root.juheL.indc.text = "= " + L1 + " H";
128 root.juheL.indar.text = "= " + r1 + " \u03A9";
129 root.juheL.indbr.text = "= " + r1 + " \u03A9";
130 root.juheL.indcr.text = "= " + r1 + " \u03A9";
131 root.tahtL.inda.text = "= " + L2 + " H";
132 root.tahtL.indb.text = "= " + L2 + " H";
133 root.tahtL.indc.text = "= " + L2 + " H";
134 root.tahtL.indar.text = "= " + r2 + " \u03A9";
135 root.tahtL.indbr.text = "= " + r2 + " \u03A9";
136 root.tahtL.indcr.text = "= " + r2 + " \u03A9";
137
138 //variandi numbrid
139 if (nr <= 24) {
140                         aa = nr
141 }
142 if (nr > 24 && nr < 49) {
143                         aa = nr - 24
144 }
145 if (nr > 48 && nr < 73) {
146                         aa = nr - 48
147 }
148 if (nr > 72 && nr < 97) {
149                         aa = nr - 72
150 }
151 if (nr > 96 && nr < 121) {
152                         aa = nr - 96
153 }
154 if (nr > 121) {
155                         aa = 12
156 }
157 //arvutused
158 var f = 50;
159 var u = 230; //u=400/Math.sqrt(3);
160 var ubgr = u * Math.cos(4 * Math.PI / 3);
161 var ubgx = u * Math.sin(4 * Math.PI / 3);
162 var ucgr = u * Math.cos(2 * Math.PI / 3);
163 var ucgx = u * Math.sin(2 * Math.PI / 3);
164 var uabgr = 400 * Math.cos(1 * Math.PI / 6);
165 var uabgx = 400 * Math.sin(1 * Math.PI / 6);
166 var ubcgr = 400 * Math.cos(3 * Math.PI / 2);
167 var ubcgx = 400 * Math.sin(3 * Math.PI / 2);
168 var ucagr = 400 * Math.cos(5 * Math.PI / 6);
169 var ucagx = 400 * Math.sin(5 * Math.PI / 6);
170 var xl1 = 2 * Math.PI * f * L1;
171 var xl2 = 2 * Math.PI * f * L2; //reaktiivtakistused
172 var xc1 = 1 / (2 * Math.PI * f * c1 * 0.000001);
173 var xc2 = 1 / (2 * Math.PI * f * c2 * 0.000001);
174 var rcj = r1 * xc1 * xc1 / (r1 * r1 + xc1 * xc1);
175 var xcj = -r1 * r1 * xc1 / (r1 * r1 + xc1 * xc1); //liini roopC
176 var rck = r2 * xc2 * xc2 / (r2 * r2 + xc2 * xc2);
177 var xck = -r2 * r2 * xc2 / (r2 * r2 + xc2 * xc2); //koormuse roopC
178
179 //teksti sisestuskastid ja sealt väärtuste leidmine
180 function evasfunc() {
181                         evas = Number(evasval.value);
182 }
183
184 function sisestused() {
185                         evasfunc();
186 }
187 //teksti sisestuskast
188 setTimeout(function () {
189                         evasval = document.getElementById("evasinput");
190
191                         evasval.value = "";
192
193                         sisestused();
194
```

```
195                 evasval.addEventListener("change", sisestused);
196                 evasval.addEventListener("input", sisestused);
197
198 }, 100);
199
200 //ülesande variant 1
201 function tickHandler(e) {
202                 if (aa == 1) {
203                                 root.juheL.visible = 1;
204                                 root.juheC.visible = 0;
205                                 lops = 1;
206                                 root.tahtL.visible = 1;
207                                 root.tahtC.visible = 0;
208                                 root.deltaL.visible = 0;
209                                 root.deltaC.visible = 0;
210                                 v = "ab"; //enterframe lõpus, praegumuutuja, hiljem teksti
211                                 root.kysdelta.visible = 0;
212                                 root.kystaht.visible = 1;
213                                 root.ubc.visible = 0;
214                                 root.uab.visible = 1;
215                                 root.uca.visible = 0;
216                                 zar = r1 + r2;
217                                 zax = xl1 + xl2;
218                                 za = zar * zar + zax * zax;
219                                 if (auk === 0) {
220                                                 iar = u * zar / za;
221                                                 iax = -u * zax / za; // kolmfaasilise ahela arvutus
222                                                 ia = Math.sqrt(iar * iar + iax * iax);
223                                                 ibr = (ubgr * zar + ubgx * zax) / za;
224                                                 ibx = (ubgx * zar - ubgr * zax) / za;
225                                                 ib = Math.sqrt(ibr * ibr + ibx * ibx);
226                                                 uar = iar * r2 - iax * xl2;
227                                                 uax = iar * xl2 + iax * r2;
228                                                 ua = Math.sqrt(uar * uar + uax * uax);
229                                                 ubr = ibr * r2 - ibx * xl2;
230                                                 ubx = ibr * xl2 + ibx * r2;
231                                                 ub = Math.sqrt(ubr * ubr + ubx * ubx);
232                                                 ulr = uar - ubr;
233                                                 ulx = uax - ubx;
234                                                 ul = Math.sqrt(ulr * ulr + ulx * ulx);
235                                                 p = 3 * ia * ia * zar;
236                                 }
237                                 //katkise ahela arvutus
238                                 else {
239                                                 zaa = 4 * za;
240                                                 iar = (ucagr * 2 * zar + ucagx * 2 * zax) / zaa;
241                                                 iax = (ucagx * 2 * zar - ucagr * 2 * zax) / zaa;
242                                                 ia = Math.sqrt(iar * iar + iax * iax);
243                                                 uar = iar * r2 - iax * xl2;
244                                                 uax = iar * xl2 + iax * r2;
245                                                 ua = Math.sqrt(uar * uar + uax * uax);
246                                                 ulr = uar;
247                                                 ulx = uax;
248                                                 ul = Math.sqrt(ulr * ulr + ulx * ulx);
249                                                 p = 2 * ia * ia * zar;
250                                 }
251                 }
252 //onEnterFrame asendus
253 createjs.Ticker.on("tick", tickHandler);
254
255 function vastusteKontroll() { // vastuste kontroll
256                 sisestused();
257
258                 if (onn > 0) {
259                                 root.kirje.visible = 1;
260                 }
261                 auka = 0; //auka==0 tuleb ära võtta, siis saab teise poole ka
262
263                 if (isFinite(evas)) { //evas on esimene teksti sisestuskast
264                                 vool1 = 1 * evas;
265                                 if ((Math.abs((vool1 - ia) / (ia + 0.0001))) <= 0.03) {
266                                                 root.tulem1.text = "Sobib, sest σ < 3%.";
267                                                 saak1 = 1;
268                                 } else {
269
```

```
270                                             if (Math.abs((vool1 - ia) / (ia + 0.0001)) > 0.03 && Math.abs((vool1 - ia) / (ia +
271 0.0001)) <= 0.1) {
272                                                     root.tulem1.text = "Ei sobi, sest σ > 3%.";
273                                                     saak1 = 0;
274                                             }
275                                             if (Math.abs((vool1 - ia) / (ia + 0.0001)) > 0.1 && Math.abs((vool1 - ia) / (ia + 0.0001))
276 <= 0.5) {
277                                                     root.tulem1.text = "Ei sobi, sest σ > 10%.";
278                                                     saak1 = 0;
279                                             }
280                                             if (Math.abs((vool1 - ia) / (ia + 0.0001)) > 0.5) {
281                                                     root.tulem1.text = "Ei sobi, sest σ > 50%.";
282                                                     saak1 = 0;
283                                             }
284                                     }
285                     } else {
286                                     root.tulem1.text = "Sisestage vastus.";
287                     }
288
289 root.kont1.addEventListener("click", vastusteKontroll);
290
291 //valmis on nupp, võib panna on click, tuleb nähtavale, kui saak = 7
292 function kuiValmis() {
293                     onn = onn + 1;
294                     if (onn == 1) {
295                                     root.valmis.visible = 0;
296                                     root.saatja.visible = 1;
297                                     root.vastamine.visible = 1;
298                                     root.kont1.visible = 0; //onn==2, siis võtab teise poole ka
299                                     evasval.value = "";
300                                     auk = 0;
301                     } else {
302                                     root.valmis.visible = 0;
303                                     if (lops == 1) {
304                                             root.lapsb.visible = 1;
305                                             root.lapsb.gotoAndPlay(1);
306                                     } else {
307                                             root.lapsc.visible = 1;
308                                             root.lapsc.gotoAndPlay(1);
309                                     }
310                                     if (aa > 12) {
311                                             root.vastusd.visible = 0;
312                                     } else {
313                                             root.vastus.visible = 0;
314                                     }
315
316                                     evasval.value = "";
317                                     root.valmis.visible = 0;
318                                     root.tulem1.text = "";
319
320                                     auk = 1;
321
322                     }
323 };
324 root.valmis.addEventListener("click", kuiValmis);
325
326
327 //abi kast, võib panna on click
328 function abiTekstLahti() { // abi tekst lahti
329                     root.tekst.visible = 1;
330                     root.abi.visible = 0;
331
332 }
333
334 root.abi.addEventListener("click", abiTekstLahti);
335
336 //abi kast, võib panna on click, tekst.sulge tähendab viide teksti ja sealt nupule sulge
337 function abiTekstKinni() { // abi tekst kinni
338                     root.tekst.visible = 0;
339                     root.abi.visible = 1;
340 }
341
    root.tekst.sulge.addEventListener("click", abiTekstKinni);

    root.stop();
```

Lisa 4. JavaScripti lugemine. Ülesande „Kolmefaasiline elektrimootortsioonmootor" elementide sidumine JavaScriptiga.

```
1  //animatsiooni elementide peitmine/näitamine
2  this.graafik.visible = true;
3  this.graafikvastu.visible = false;
4  this.vastupidine.visible = false;
5  this.nool.visible = false;
6  this.magnetsumma.visible = false;
7  this.magnetindiv.visible = false;
8  this.vastupidine.visible = false;
9  this.laeng.visible = false;
10 this.staatorrootor.visible = true;
11 this.kiirus2.visible = false;
12 //noole suund tööle
13 this.noolon.addEventListener("click", noolfunc.bind(this));
14 function noolfunc()
15 {              this.noolon.visible = false;
16                this.nooloff.visible = true;
17                this.nool.visible = true;
18                this.vastupidine.visible = false;
19                this.graafikvastu.visible = false;
20                this.staatorrootor.visible = true;
21                this.graafik.visible = true;}
22 //noole suund kinni
23 this.nooloff.addEventListener("click", noolfunc2.bind(this));
24 function noolfunc2()
25 {              this.noolon.visible = true;
26                this.nooloff.visible = false;
27                this.nool.visible = false;
28                this.vastupidine.visible = false;
29                this.graafikvastu.visible = false;
30                this.staatorrootor.visible = true;
31                this.graafik.visible = true;}
32 //summaarne magnetväli
33 this.magnetsummaon.addEventListener("click", magnetsummafunc.bind(this));
34 function magnetsummafunc()
35 {              this.magnetsumma.visible = true;
36                this.magnetindiv.visible = false;
37                this.vastupidine.visible = false;
38                this.graafikvastu.visible = false;
39                this.staatorrootor.visible = true;
40                this.laeng.visible = true;
41                this.graafik.visible = true;}
42 //individuaalne faasi magnetväli
43 this.magnetindivon.addEventListener("click", magnetindivfunc.bind(this));
44 function magnetindivfunc()
45 {              this.magnetindiv.visible = true;
46                this.magnetsumma.visible = false;
47                this.vastupidine.visible = false;
48                this.graafikvastu.visible = false;
49                this.staatorrootor.visible = true;
50                this.laeng.visible = true;
51                this.graafik.visible = true;}
```

```
52 //vastupidine suund
53 this.vastupidineon.addEventListener("click", vastupidinefunc.bind(this));
54 function vastupidinefunc()
55 {               this.vastupidine.visible = true;
56                 this.graafikvastu.visible = true;
57                 this.staatorrootor.visible = false;
58                 this.laeng.visible = false;
59                 this.magnetsumma.visible = false;
60                 this.nool.visible = false;
61                 this.magnetindiv.visible = false;
62                 this.graafik.visible = false;}
63 //animatsiooni kiirus
64 this.kiirus1.addEventListener("click", kiirusfunc.bind(this));
65 function kiirusfunc()
66 {               this.kiirus1.visible = false;
67                 this.kiirus2.visible = true;}
68
69 this.kiirus2.addEventListener("click", kiirusfunc2.bind(this));
70 function kiirusfunc2()
71 {               this.kiirus1.visible = true;
72                 this.kiirus2.visible = false;}
73
74 this.kiirus1.addEventListener("click", function(e){createjs.Ticker.framerate = 11;});
75 this.kiirus2.addEventListener("click", function(e){createjs.Ticker.framerate = 48;});
76
77 //nuppude all olevad info nupud
78 this.infomagnetsumma.addEventListener("click", infomagnetsummafunc.bind(this));
79 this.infomagnetindiv.addEventListener("click", infomagnetindivfunc.bind(this));
80 this.infokiirus.addEventListener("click", infokiirusfunc.bind(this));
81 this.infonool.addEventListener("click", infonoolfunc.bind(this));
82 this.infovastupidine.addEventListener("click", infovastupidingfunc.bind(this));
83
84 function infomagnetsummafunc()
85 {               this.infokast.text ="Summaarne rootori ja staatori magnetvälja kujutus. Rootoril olev indutseeritud magnetväli on
86 kraadiliselt tagapool.";}
87 function infomagnetindivfunc()
88 {               this.infokast.text ="Individuaalsed staatori voolufaaside kujutus. Magnetvälja suurus ja suund oleneb voolu
89 tugevusest ja polaarsusest.";}
90 function infokiirusfunc()
91 {               this.infokast.text ="Simulatsiooni kiiruse vähendamine.";}
92 function infonoolfunc()
93 {               this.infokast.text ="Pöörlemissuuna paremaks kujutamiseks koostatud nool.";}
94 function infovastupidingfunc()
95 {               this.infokast.text ="Kahe faasi U ja V algfaaside vahetus, mis paneb mootori teises suunas pöörlema.";}

    this.stop();
```