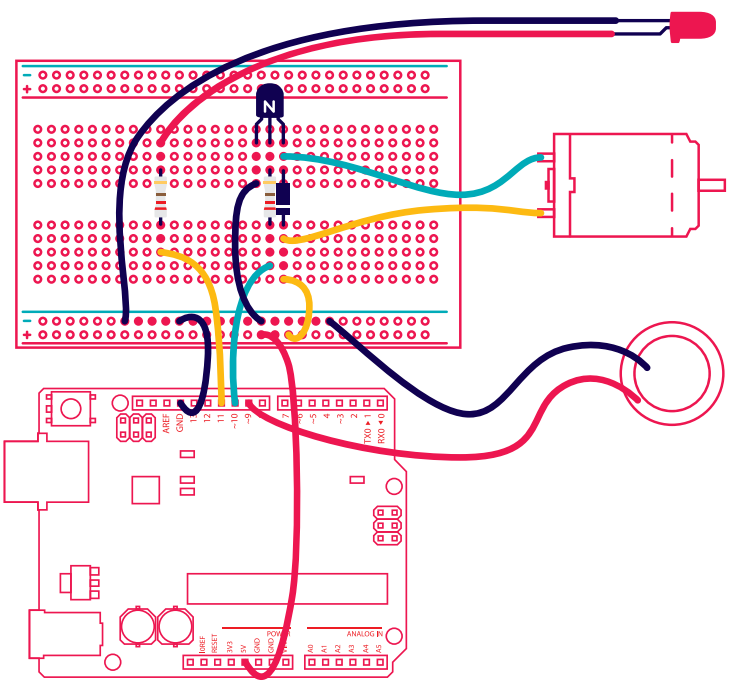
Elektroniikan ja ohjelmoinnin alkeita varhaiskasvatukseen ja alkuopetukseen

ROBO2

– Elektroniikkaa ja ohjelmointia



KÄSITYÖKOULU ROBOTTI

Toimitus: Iiro Tujula ja Roi Ruuskanen
Oikolukija: Marjukka Bastamow
Taitto: Lauri Tujula
Paino: Aleksipaino Group Oy, Trio-Offset, Höyläämötie 3, 00380 Helsinki

Oppaan kytkentäkuvissa hyödynnetty avointa Fritzing-ohjelmaa | www.fritzing.org

Oppaassa käytettäviä elektroniikan komponentteja voit hankkia elektroniikan komponenttien myyntiin erikoistuneista myymälöistä ja verkkokaupoista.

ISBN 978-952-68470-2-3

Opetus- ja kulttuuriministeriö on tukenut oppaan tekoa.

Opetus- ja kulttuuriministeriö
Undervisnings- och kulturministeriet

Käsityökoulu Robotti

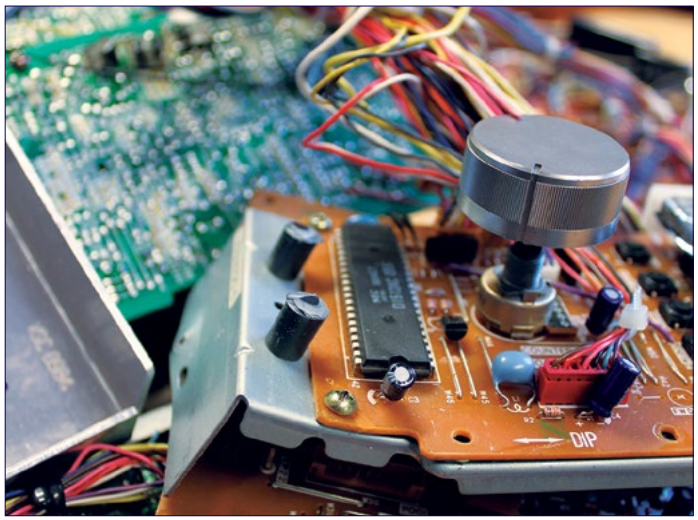
Käsityökoulu Robotti on lasten ja nuorten media- ja elektronisen taiteen koulu, jota ylläpitää voittoa tavoittelematon yleishyödyllinen yhdistys Käsityökoulu Robotti ry.
Käsityökoulu Robotin toimintaa ovat tukeneet Opetus- ja kulttuuriministeriö, Suomen kulttuurirahasto, Taiteen edistämiskeskus, Espoon kaupunki ja TOP-säätiö.

Käsityökoulu Robotti, Järvenperäntie 1-3, 02940 Espoo
info@kasityokoulurobotti.fi | www.kasityokoulurobotti.fi

KÄSITYÖKOULU ROBOTTI

Sisällysluettelo

Johdanto



Ympäröivä maailma on täynnä erilaisia laitteita, ohjelmoituja järjestelmiä ja elektroniikkaa. Elämämme kietoutuu digitaaliseen ja elektroniseen laitemaailmaan halusimme sitä tai emme. Ihmisten välinen vuorovaikutus tapahtuu yhä useammin erilaisten sähköisten ja ohjelmoitujen mediateknologisten käyttöliittymien kautta. Nämä laitteet ja ohjelmistot toimivat ja jäsentävät tapojamme olla ja toimia tavoilla, joita emme useinkaan kykene hahmottamaan. Ohjelmoidut algoritmit ja laitteiden sisälle piilotettu elektroniikka ohjaavat elämäämme huomaamattomasti taustalla.

Haluamme rohkaista tällä oppaalla teknologisoituvan ja digitalisoituvan maailman avaamiseen, tutkimiseen ja omaehtoiseen luomiseen jo pienestä pitäen. Oppaan sisällöt on suunnattu opettajille ja muille kasvatusalalla toimiville itseopiskelumateriaaliksi ja ideapankiksi. Oppaan kohderyhmänä ovat varhaiskasvatuksen ja alkuopetuksen toimijat, mutta tehtävistä löytyy haastetta ja innostavaa tekemistä kaiken ikäisille.

Tutustuttamalla ohjelmoituihin ja sähköisiin ympäristöihin voi teknologinen maailma tulla tutummaksi ja ymmärrettävämmäksi lapsille. Samalla käsitys asioiden keskinäisistä suhteista selkeytyy ja maailmankuva eheytyy. Opas tähtää luomaan edellytyksiä toteuttaa taidekasvatuslähtöistä mediakasvatusta, jossa tavoitteena on oman kokemuksen tunnistaminen mediakulttuurin keskellä ja omaehtoisen ilmaisun luominen median keinoin.

Opasta voi lukea kronologisena kokonaisuutena tai itsenäisinä osina tutustuen vain yhteen näkökulmaan. Parhaimmillaan tehtävät avautuvat välineinä päästä alkuun tutustumismatkalla ohjelmoinnin ja elektroniikan maailmaan. Näkökulmaksi ja pohdittavaksi matkalle voit ottaa vaikkapa kysymyksen ohjelmoidun ja ei ohjelmoidun maailman eroavaisuuksista. Tällöin kiinnostaviksi kysymyksiksi voivat nousta mm. Miltä elektroniikka ja koodi tuntuvat ihmisestä? Minkälaiselta ne saavat maailman vaikuttamaan? Miten ohjelmointi saa maailman toimimaan ja jäsentymään? Mitä laitteiden alla oikein tapahtuu? Näihin kysymyksiin voi saada vastauksia niiden melko yksinkertaistenkin työvälineiden avulla, joita oppaassa esittelemme.

Opas rakentuu viidestä osiosta. Ensin tutustumme elektroniikan perusteisiin, minkä jälkeen otamme askeleen kohti ohjelmointia ohjelmointileikkien avulla. Oppaan loppuosan oppimispolun muodostavat kolme Scratch-ohjelmaa: ScratchJr, Scratch ja Scratch for Arduino. Eri tehtävien välillä on löydettävissä myös johdonmukainen kronologia yksinkertaisemmasta ilmiöstä vaativampaan, sekä keskinäisiä viittauksia. Voit tutustua ja soveltaa oppaasta saamiasi ideoita haluamallasi laajuudella ja myös toteuttaa vain valittuja yksittäisiä tehtäviä.

Oppaan tehtäviä on Käsityökoulu Robotin viikottaisten opetusryhmien lisäksi kehitetty ja testattu Karamäen ja Kirkkojärven päiväkodeissa Espoossa sekä Keskustan päiväkodissa Naantalissa.

Tämän oppaan tehtävät löytyvät myös Käsityökoulu Robotin kotisivujen tehtäväpankista osoitteesta www.kasityokoulurobotti.fi. Kotisivuilta on löydettävissä myös aiemmin ilmestynyt *ROBO1 – Elektroniikaa ja askartelua* -opas.

Zzap!

Iiro Tujula ja Roi Ruuskanen
Käsityökoulu Robotti ry

Taidekasvatuslähtöinen teknologioiden oppiminen

Käsiteltäessä teknologisoitunutta ja digitalisoitunutta maailmaa lasten kanssa voi tulokulmia olla useita. Usein keskeinen näkökulma on lapsen tutustuttaminen ympäröiviin teknologisiin ratkaisuihin, mikä tarkoittaa teknologian toimintamallien käytön opettelua. Tähän lähestymiseen liittyvä oppimispolku voi alkaa oman mediateknologisen suhteen tiedostamisesta. Sen tiedostamisesta, missä tilanteessa ja millä tavoin mediateknologia näyttäytyy lapsen omassa elämässä.

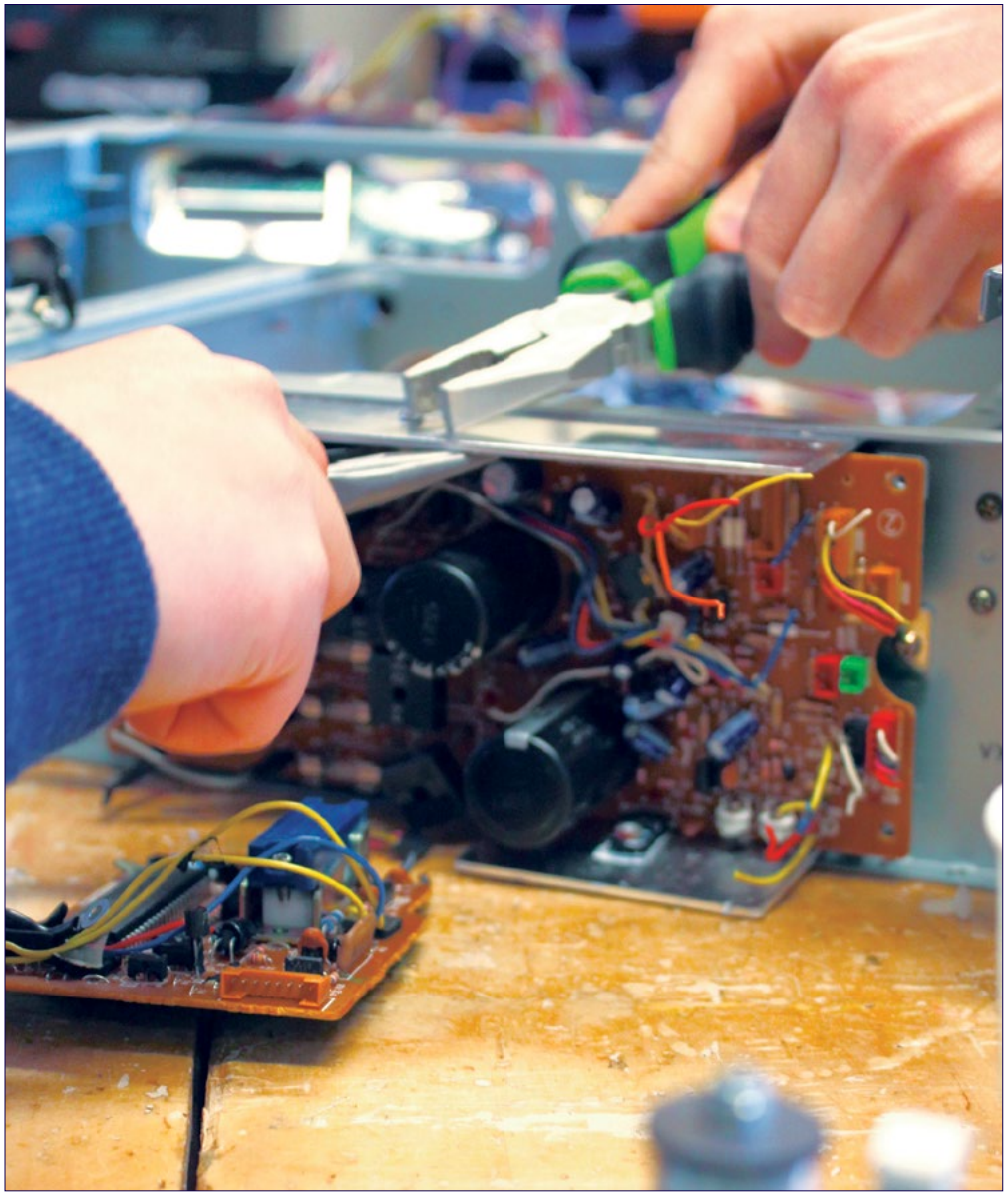
Mediateknologian tutkimisen ja omakohtaisen rakentelun kautta lapset oppivat paitsi tarvittavia taitoja, myös ymmärtämään, että heillä on mahdollisuus luoda itse teknologisia maailmoja. Teknologian demystifioimisen lisäksi kyse on valtarakenteiden ymmärtämisestä. Tällöin kysymykset teknologiasta saavat uuden näkökulman mediakäyttäytymisemme ja olemisemme ehtoja tarkastellen. Parhaimmillaan opetuksen keskiöön nousee uusia toiminta- ja ajattelutapoja, joissa lapsi ymmärtää, ettei teknologian käytössä tarvitse alistua vain muiden määrittelemiin toimintatapoihin vaan voi alkaa luomaan aktiivisesti omia ratkaisuja.

Teknologian toiminnan ymmärtämisen lisäksi on lasten kanssa hyvin oleellista pyrkiä hahmottamaan teknologian merkityksiä elämälle. Miltä teknologian käyttö saa asiat tuntumaan? Miten se saa meidät käyttäytymään? Millaisiksi keskinäiset suhteemme muodostuvat? Millaista inhimillinen teknologian käyttö voisi olla? Teknologian toiminnan hahmottaminen saattaa olla hankalaa aikuisellekin, sillä yleensä teknologia toimii ilman että kohtaisimme koodia, joka ohjaa laitetta, tai elektroniikkaa, jonka ympärille laitteen toiminta rakentuu. Tämä opas tarjoaa yhden väylän aloittaa tutustuminen aihealueeseen niin aikuisille kuin lapsillekin.

Oppaassa painottuu kognitiivinen oppimiskäsitys, jolloin tieto ymmärretään kaikkina mahdollisina tapoina käsitellä informaatiota: se hahmottuu paitsi käsitteellisinä ajatuksina myös tuntemuksina ja aistimuksina, jolloin informaatio tulee kaikkien aistien avulla saavutettavaksi ja käsiteltäväksi. Tällöin elektroniikan ja ohjelmoinnin opettelussa voidaan soveltaa moninaisia menetelmiä. Käsitellyksi ei tule ainoastaan perinteinen tapa liittää aihe matemaattis-luonnontieteelliseen kontekstiin rationaalisen jäsentelyn menetelmin, vaan rinnalle rakentuu taidekasvatuksellinen ote.

Taidekasvatus ei kata vain kuvallisen ilmaisun osa-alueita tai esteettisiä piirteitä, vaan taide on prosessi, missä oppija luo henkilökohtaisen ja syvän merkityksellisen suhteen käsiteltävään aiheeseen. Tuota aihetta hän purkaa, pohtii, venyttää, kasaa, järjestelee uudestaan, tarkastelee eri suunnista, vertaa, hylkää ja hyväksyy yhä uudestaan. Taiteellinen prosessi on suhteen luomista käsiteltäviin asioihin, ja joskus se vaatii ainoastaan käsillä käsittelyä, toisinaan taas käsitteellistä ajattelua.

Toimiessa pienten lasten kanssa korostuu heillä usein tarve päästä hahmottamaan ilmiötä käsin – usein mitä ripeämmin, sen parempi. Tärkeää on päästä vauhdilla luovaan prosessiin, joka lapsille ilmenee ominaisesti leikkinä. Erilaiset tietämisen tavat ilmenevät fantastisina mahdottomina kuvitelmina, joita jakaa toisille. Tärkeää on kokeilla turvallisesti ja tulla tutuksi teknologian mahdollisuuksien kanssa.



Kasvamassa digitaalisessa maailmassa

Tomi Dufva

Elämme digitalisoituvassa maailmassa. Digitalisaatio on yksi aikamme suuria piirteitä, mutta samalla se on ladattu ja monia tulkintoja sisältä termi. Digitalisaatio on läsnä enenevässä määrin, kun kuvaillaan tulevaisuuden maailmaa. Digitalisaatiota käytetään melko väljästi kuvaamaan digitaalisen teknologian yleistyvän käytön mukanaan tuomia muutoksia. Digitalisaatiolle on useita määritelmiä, jotka kuvastavat sekä siihen ladattuja suuria odotuksia että sen tuomia uhkakuvia. Puhutaan esimerkiksi "digiloikasta", joka tapahtuu kun kaikki on digitalisoitu, mitä digitalisoitavissa on. Toisaalta pelätään digitalisaation vievän suuren osan töistä, kun algoritmit kehittyvät ihmisiä ketterämmiksi yhä uusilla alueilla. Mutta mitä digitalisaatio oikeastaan tarkoittaa käytännön tasolla? Miten se heijastuu ihmisten arkeen ja mitä se edellyttää ihmisiltä?

Hyvin yleisellä tasolla voidaan todeta, että digitalisaatio pitää sisällään sekä automaation lisääntymisen että uudenlaiset ihmisten avoimeen yhteistoimintaan perustuvat toiminta-mallit. Yhteistä digitalisaation alle luettaviin ilmiöihin on se, että ne hyödyntävät digitaaliseen muotoon tallennettua tietoa ja tämän tiedon vaivatonta virtaamista tietoverkoissa. Digita-lisaatiossa ei sinänsä ole mitään mullistavaa; elämme jo nyt digitaalisten tietorakenteiden kyllästämässä yhteiskunnassa. Huomionarvoista on digitalisaation laajuus ja kattavuus. Digitalisaation eri piirteiden ymmärtäminen tuleekin kasvamaan uusien aluevaltausten, kuten esineiden internetin ja tekoälyn kehittymisen myötä.

Mitä tarkoittaa eläminen digitalisoidussa maailmassa ja millainen tuo maailma on? Esitän tässä puheenvuorossa aluksi muutamia filosofisia tapoja hahmottaa teknologiaa ja valotan lasten media- ja teknologiakasvatuksen syitä ja tärkeyttä.

Teknologia on uusi pyörä

Yhdysvaltalainen tietokirjailija Nicholas Carr (2011) on esittänyt, että digitaalinen teknologia on yksi historiamme käänteentekevistä keksinnöistä pyörän, kartan, kirjapainon tai mekaa-nisen kellon ohella. Kartan keksiminen muutti tapaamme havainnoida ja ajatella maailmaa: enää ei ollut tarvetta muistaa ulkoa reittejä, ja koko maailman hahmotus muuttui ajallisesta matkasta ennalta visualisoitavaksi reitiksi. Samaan tapaan mekaaninen kello oli Carrin mukaan se keksintö, joka muutti käsityksemme ajasta. Enää ei menty auringon liikkeiden tai kirkon kellon lyöntien mukaan, vaan mekaaninen kello saneli meille tarkempaa käsitystä ajasta. Ilman kelloa ei esimerkiksi teollistuminen – tehtaiden tarkkoine työvuoroineen – ollut mahdollista. Digitaalinen teknologia on Carrille vastaavanlainen muutos: se muuttaa tapaamme ajatella ja hahmottaa maailmaa. Etsimme tietoa internetistä, kommunikoimme Whatsappissa, lataamme kuvia Instagramiin. Digitaalinen teknologia yhdistää useat mediat yhteen linkittäen niitä toisiinsa ja luoden uudenlaisia käytäntöjä.

Ihminen teknologisena olentona

Ranskalainen filosofi Bernard Stiegler on kirjoittanut laajasti ihmisestä ja teknologiasta. Hän näkeekin ihmisen teknologisena olentona: teknologia on ihmisen luonto (Stiegler 1998). Hän nojaa käsityksensä siihen, että miltei kaikki tekemisemme tapahtuu teknisten välineiden avulla. Teknologia on tapamme olla olemassa ja jopa ajatella. Teknologian avulla lisäämme itseemme kykyjä, joita meillä ei ole. Sukellamme, lennämme ja matkustamme tekniikan avulla. Teknologia on Stieglerille kuitenkin ongelmallista. Hän kutsuukin teknologiaa pharma-coniksi, yhtäaikaa sekä myrkyksi että lääkkeeksi. Teknologia on kuin ulkoistettu muistimme, joka auttaa kehitystämme mutta myös edesauttaa unohtamista. Digitaalisen teknologian kohdalla hän tuo esille lääkkeen myös huumaavan vaikutuksen: samoin kuin lääkkeeseen voi jäädä koukkuun, on teknologiassa vastaavia piirteitä, jolloin sen parantava vaikutus lakkaa.

Digitaalisen tietokoneen luonne

Digitaalisen teknologian voi nähdä eroavan perinteisistä teknologioista sen ohjelmoidun luonteen vuoksi. Perinteiset koneet oli aiemmin rakennettu tiettyä päämäärää varten, mutta digitaalinen teknologia toi mahdolliseksi koneen käyttötarkoituksen muuttamisen. Samalla koneella voitiin nyt hoitaa sama, mihin aiemmin tarvittiin useita eri koneita. Lisäksi jokaista ohjelmaa voitiin muuttaa. Tämä muuttaminen tapahtuu juuri ohjelmoimisen kautta. Ohjel-mointi onkin digitaalisen teknologian suurin eroavaisuus muista teknologioista. Ohjelmointi luo teknologiaan uuden kerroksen, joka yhdistää "raudan" – teknologian fyysisen puolen – "softaan", ohjelmoinnin avulla tehtyihin työkaluihin. Digitalisaation yksi suuri ominaisuus onkin juuri tuo kaiken tuleminen ohjelmoiduksi.

Koodi on laki

Stanfordin professori ja digitaalisten oikeuksien puolestapuhuja Lawrence Lessig kirjoitti jo 1990-luvun lopussa, että "Koodi on laki" (Lessig 1998). Tällä hän tarkoitti sitä, että se, miten ohjelmoimme tietokoneitamme ja niiden sisältämiä ohjelmiamme luo maailmaamme lakeja. Digitaalisen teknologian toimintamallit eivät ole muuttamattomia luonnon lakeja vaan plastisia malleja, joiden rakenne on meidän käsissämme. Koodi ei myöskään ole arvovapaata vaan heijastaa laajalti niin tiedostettuja kuin tiedostamattomiakin arvovalintoja, oli kyseessä sitten ohjelmoijan, ohjelmointifirman tai laajemman kulttuurisen taustan käsitys hyvästä koodista. Useat tutkijat ovatkin esittäneet, että ilman ymmärrystä ohjelmoinnin luomien lakien muodostumisesta emme pysty täysivertaisesti osallistumaan yhteiskunnalliseen keskusteluun digitaalisesta elämästämme (ks. esim. Giroux 2011; Lessig 2006; Rushkoff 2010).

Käsin tehty digitaalinen media

Käsityön professori Seija Kojonkoski-Rännäli on kirjoittanut laajasti käsillä tekemisen merkityksestä ihmiselle (Kojonkoski-Rännäli 1998, 2014). Käsillä tehden hahmotamme ja luomme maailmaamme. Käsillä tehtyyn esineeseen tunnemme emotionaalista sidettä. Kojonkoski-Rännälin mukaan käsillä tekeminen myös rakentaa eettistä sidettä meidän ja maailman välille. Tekemällä käsin välitämme tekemästämme maailmasta. Digitaalinen maailma esittäytyy meille usein valmiina. Tulemme valmiiseen maailmaan ja toimimme siinä käyttäjinä ja asiakkaina mutta emme tekijöinä. Samoin kuin usean muun teknologian Kojonkoski-Rännäli näkee digitaalisen teknologian uhkana sen, että se etäännyttää meidät maailmastamme: emme koe vastuuta maailmasta, ja toisaalta menetämme nopeasti arvostuksen sen esineistä. Miten siis tehdä käsin digitaalisessa ja abstraktissa maailmassa? Teknologian asema käsillä tekemisessä onkin ongelmallinen. Teknologia mahdollistaa uusia käsin tekemisen muotoja, kuitenkin vähentäen juuri käsillä tekemisen määrää. Tietyssä mielessä juuri ohjelmoiminen on digitaalisen teknologian käsillä tekemistä. Tämä käsillä tekeminen tapahtuu abstraktin ja käsillä ulottumattomissa olevan koodin avulla. Kuitenkin juuri ohjelmoinnin kautta ohjelmoija luo maailmaansa. Ohjelmointi on olemassaoloa digitaalisessa maailmassa. Kun ohjelmointi ymmärretään käsillä tekemisenä, se ei ole vain tuotannollinen taito, vaan keino luoda yhteys digitalisoituneeseen yhteiskuntaan. Millainen tämä kehollinen yhteys sitten käytännössä on? Ohjelmoinnissa luodaan koodia, joka on tarkasti määritelty kuvaus halutusta palvelusta tai toiminnallisuudesta. Koodi toimii abstraktin toiminnallisuuden ja käsillä tekemisen välikappaleena. Vaikka koodikin on tietyssä mielessä abstraktia, se on "käsinkosketeltavan abstraktia". Se on selkeä objekti, jota voidaan muokata.

Käsillä tekeminen opettaa Kojonkoski-Rännälin mukaan tulevaisuuden kannalta tärkeitä taitoja, kuten visiointikykyä, oma-aloitteisuutta, holistisuutta ja organisointikykyä (Kojonkoski-Rännäli 2006). Ohjelmointi opettaa näiden lisäksi hahmottamaan, ymmärtämään ja haastamaan erityisesti digitaaliseen teknologiaan perustuvia rakenteita, juuri koska se tuo niiden abstraktin rakenteen "käsinkosketeltaviksi". Toisin sanoen ohjelmointi auttaa ymmärtämään näiden rakenteiden intentionaalisuutta ja tietoisesti tai tiedostamatta tehtyjä oletuksia.

Digitaaliset kansalaistaidot

Digitalisaation yhtenä vaarana on ihmisten jakautuminen digiosaajiin ja digitumpeloihin. Digiosaajat näkevät yhteiskunnan digitaaliset rakenteet, ymmärtävät niihin liittyvät oletukset ja osaavat myös kyseenalaistaa ne. Digitumpelot suhtautuvat digitaalisiin palveluihin, joita heidän on enenevässä määrin pakko käyttää, kuin ne olisivat taianomaisia tai kuin niillä olisi oma mieli. He ovat siis enemmän digitaalisuuteen pohjautuvien rakenteiden armoilla taidollisesti, mutta myös ymmärryksen tasolla. Sivuhuomautuksena voidaan mainita, että ns. diginatiivit saattavat kuulua enemmänkin digitumpeloihin kuin digiosaajiin, koska digitaalisuus on heille ikäänkuin annettua ja piilotettua. (ks. esim. Kupiainen 2013). Jotta tasa-arvoinen ja yhteistoimintaa korostava yhteiskunta olisi tulevaisuudessakin mahdollinen, tarvitaan digitaalisia kansalaistaitoja.

Digitaaliset kansalaistaidot perustuvat paitsi ajattelumallien, koodin loogisen ja algoritmisen luonteen sisäistämiseen, myös käsillä tekemisen kautta oppimiseen. Näistä jälkimmäinen on jäänyt vähemmälle huomiolle. Kuten edellä mainitsin, ohjelmoinnin opettaminen kädentaitona olisi yksi keino tuoda kehollista ymmärrystä digitalisaatiosta. Tämä edellyttää kuitenkin ohjelmoinnin käsittämistä laajemmin kuin vain osana matematiikan opetusta. Älyllisen ajattelun ohella ohjelmointi on työkalu itsensä ilmaisuun sekä keino ymmärtää ja muokata ympäröivää maailmaa. Toisin sanoen ohjelmointi on osa tulevaisuuden käsillä tekemistä.

Digitaalisuuden myötä ihmisen maailmassa oleminen on laajentunut abstrakteihin rakenteisiin, joissa kysymykset maailmasta tulevat uudenlaiseen valoon: digitaalinen maailma eroaa analogisesta. Vaikka digitalisaation diskurssi keinuu aina taianomaisista käsityksistä villeihin tekoälyfantasioihin, on digitaalinen maailma osa nyky-yhteiskuntaa. Digitalisaation abstrakti ja toisaalta piilotettu luonne vaikeuttavat tämän maailman ymmärtämistä, sekä sen suoraa kokemista tai "käsittämistä". Vaarana onkin, että vaikka digitaalinen teknologia ei suoraan määritä toimintaamme, vaikuttaa se siihen jopa huomaamattamme. Alamme olettaa että ohjelmien ja käyttöjärjestelmien pitääkin toimia juuri sillä tavalla, mihin olemme tottuneet. Tekoälyn tutkija Jaron Lanier ja MIT:n professori Sherry Turkle puhuvatkin siitä, miten alennamme odotuksiamme teknologialta ja mukaudumme niihin (Lanier 2011; Turkle 2011). Ohjelmoiminen käsillä tekemisenä auttaa sekä näkemään ohjelmoitua maailmaa paremmin että myös tuomaan käsin kosketeltavaa ymmärrystä. Tämän kehollisen kokemuksen kautta voi olla mahdollista tuntea kuuluvansa maailmaan luoden empaattisia ja eettisiä siteitä tähän maailmaan. Käsillä tekemisen aseman voikin nähdä yhä digitalisoituvassa maailmassa entistä tärkeämpänä ymmärryksen välineenä.

Taide käsin tekemisen metodina

Käsityön ja taiteen suhde on aina ollut läheinen. Taiteilija onkin käsillä tekemisen ammattilainen. Mutta toisin kuin käsityöläisyydessä, käsillä tekemisen tulos taiteessa ei ole samalla tapaa päämäärähakuinen. Taide antaa tilaa tutkia. Taiteen tekemisessä ilmenee ehkä selkeimmin ihmisen suora suhde maailmaan. Taide ei ihmettele onko havainnoitava maailma todellinen, vaan taiteen tekijän omat havainnot muodostavat todellisen maailman. Taiteen kautta tekijä ikäänkuin tutkii maailmaansa (Kojonkoski-Rännäli 2014).Tämän tutkivan ja uteliaan asenteen vuoksi taiteen käyttäminen käsillä tekemisen metodina vapauttaa tekijän tutkimaan kohdettaan uudella tavalla, kokeilemaan erilaisia asioita ja myös rakentamaan syvempää, omaan kokemukseen perustuvaa ymmärrystä. Ohjelmoiva taiteilija tutkii hänelle esiintyvää digitaalista maailmaa – usein ja erityisesti – ilman formaalin koulutuksen ohjeistamaa käytäntöä. Taiteen kautta ohjelmointi voi näyttäytyä työkaluna, jonka avulla voi ilmaista itseään ja luoda jotain uutta. Koodi toimii ikäänkuin siveltimenä, joka mahdollistaa sellaisten erilaisten, uusien näkökulmien esiin tuomisen, jotka eivät tulisi samanlaisina esille toisella medialla (Cox 2013). Toisaalta ohjelmointi on myös keino lisätä toiminnallisuutta perinteisemmin keinoin tuotettuihin objekteihin. Esimerkiksi erilaisten robottien teossa yhdistyvät fyysisen esineen työstö ja sen "älyn" ohjelmointi. Ohjelmointi toimii tällöin fyysisen ja virtuaalisen rajapintana; sen avulla saadaan uusi ulottuvuus käsillä tehtyihin esineisiin. Toisena esimerkkinä älyvaate tuo ohjelmoinnin lähes iholle: ilmansaasteista ilmoittava mekko tai jännittyneisyyttä ilmentävä paita sekä ulkoistavat kehollisia kokemuksia että myös kehollistavat ulkoista tietoa. Ohjelmointi toimii ikään kuin fyysisen ja abstraktin välikappaleena, ihmisen luomana tulkkina, joka tuo esille digitalisoitua informaatiota. Samanaikaisesti ohjelmoinnissa tehdyt valinnat myös peittävät ja vaimentavat muita signaaleja. Ohjelmoinnin käsittäminen työkaluna, siveltimenä, tuo tämän intentionaalisuuden selvemmin ilmi. Ohjelmointi on aktiivista osallistumista yhteiskuntaan ja maailmaan.

Digitalisaatio kasvatuksessa

Digitaalinen teknologia on jo nyt osa arkipäivästä elämäämme. Esineiden internetin ja tekoälyä käyttävien palveluiden kehittyessä ja yleistyessä olemme digitaalisuuden ympäröimiä. Tämä digitaalisuuden kaikkiaallisuus tekee digitaalisen teknologian ymmärryksestä tärkeää

kasvatuksessa kaikilla oppiasteilla. Digitaalisuus tulee näkyväksi jo varhaiskasvatuksessa: diginatiivit taaperot osaavat jo hiplata tabletteja, joskus jopa vanhempiaan paremmin. Tämä ei vielä tarkoita että he ymmärtäisivät digitaalista teknologiaa sen paremmin. Me näemme vain sen pinnan, jonka ohjelmoija on meille luonut. Opimme kuluttamaan, emme tekemään. Koodi luo lakeja ja arkkitehtuuria maailmaamme, mutta usein ongelmaksi muodostuu se, ettemme edes näe tai ymmärrä tämän arkkitehtuurin olemassaoloa. Kun lelut ja leikit korvaantuvat digitaalisilla, on hyväksi sekä tuoda ymmärrystä että kehollista kokemusta tähän maailmaan. Digitaalisuuden näyttäytyminen taianomaisena tai muuten mahdottomana ymmärtää, tulee esille niin ohjelmissa kuin fyysisissä laitteissa, kuten leluissa. Puhuvan nuken tai kauko-ohjattavan auton mennessä rikki sen arvo häviää: emme tiedä miten lelua voisi korjata. Toisaalta lelujen teknologia ohjaa ja määrittelee leikkejä: koodin luoma rakenne tulee esille jo pienten lasten leikeissä. Luovan käsillä tekemisen kautta digitaalinenkin maailma alkaa näyttäytyä ymmärrettävänä ja käsitettävänä asiana, jossa voi toteuttaa itseään ja josta voi muodostaa tietoisia mielipiteitä.

Viitteet:

Carr, N., 2011 The Shallows: What the Internet Is Doing to Our Brains, W. W. Norton & Company
Cox, G., 2013. Speaking Code, MIT Press.
Giroux, H.A., 2011. On Critical Pedagogy, Bloomsbury Publishing USA.
Kojonkoski-Rännäli, S., 1998. Ajatus käsissämme. Turun yliopisto, Rauman opettajankoulutuslaitos.
Kojonkoski-Rännäli, S., 2006. Tulevaisuuden käsityötaito. Available at: http://www.kaspaikka.fi/opettajanpoyta/artikkeleja/kojonkoski-rannali-futurassa.htm [Accessed March 27, 2014].
Kojonkoski-Rännäli, S., 2014. Käsin tekemisen filosofiaa, University of Turku.
Kupiainen, R., 2013. Diginatiivit ja käyttäjälähtöinen kulttuuri. widescreen.fi. Available at: http://widerscreen.fi/numerot/2013-1/diginatiivit/ [Accessed September 23, 2015]. Lanier, J., 2010. You Are Not a Gadget, Vintage.
Lessig, L., 1999. Code and Other Laws of Cyberspace, Basic Books (AZ).
Lessig, L., 2009. Code 2.0, CreateSpace.
Rushkoff, D., 2010. Program Or Be Programmed, OR Books.
Stiegler, B., 1998. Technics and Time: The fault of Epimetheus, Stanford University Press.
Turkle, S., 2011. Alone Together, Basic Books, Inc.

–

Tomi Dufva – Taiteen maisteri; Tohtori koulutettava, Aalto-yliopisto, Taiteiden ja suunnittelun korkeakoulu

Tomi Dufva on turkulainen taiteilija, taideopettaja ja tutkija. Väitöstutkimuksessaan Aalto yliopistossa Tomi tutkii luovaa ohjelmointia taidekasvatuksen menetelmänä. Tomin erikoisalueita ovat mm. koodinlukutaito, maker-kulttuuri ja elektroniikan ja koodauksen yhdistäminen perinteiseen kuvataiteeseen. Hän on Käsityökoulu Robotin toinen perustajajäsen.

Ohjelmoinnista ja ohjelmointiympäristöistä

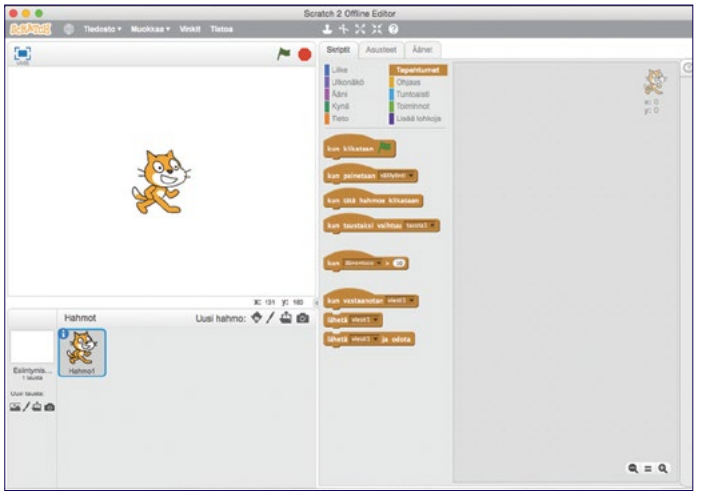
Oppaassa tutustutaan ohjelmointiin graafisten ohjelmointiympäristöjen kautta. Graafiset ohjelmointiympäristöt helpottavat ohjelmoimista tekemällä usein vaikealukuisesta ja monimutkaisesta tekstipohjaisesta ohjelmointikielestä helposti lähestyttävää. Graafiset ohjelmointiympäristöt sisältävät joskus myös tekstiä, mutta yleisesti niiden käyttäminen ei vaadi lukutaitoa, ja siksi ne soveltuvat ohjelmoinnin opettamiseen myös varhaiskasvatuksessa.

Tässä oppaassa luomme kokonaisuuden, jossa luodaan kaari alkaen ohjelmointileikeistä tabletilla tehtävään graafiseen ohjelmointiin (ScratchJr) sekä tietokoneella toteutettavaan graafiseen ohjelmointiin (Scratch), edeten lopulta graafiseen Scratch for Arduino (S4A) -ohjelmointiympäristöön, joka mahdollistaa itse rakennetun laitteen ohjelmoimisen tutuilla graafisilla ohjelmointimenetelmillä. Edellä mainituista ohjelmointiympäristöistä ScratchJr on täysin graafinen. Scratch ja Scratch for Arduino sisältävät myös tekstiä käyttöliittymissään. Ohjelmat ovat kuitenkin saatavilla suomen kielellä.

ScratchJr on ladattavissa ilmaiseksi iPadille App Storesta tai Android-tabletille Google Playstä.

ScratchJr

Ohjelmointiin lähdetään tutustumaan ensin ScratchJr-ohjelmointiympäristöllä, jolla on mahdollista rakentaa pieniä animoituja vuorovaikutuksellisia ohjelmia, kuten pelejä. Ohjelmointiympäristö ja käytetty graafinen ohjelmointikieli koostu selkeistä symboleista, eikä sen käyttäminen vaadi lukutaitoa. ScratchJr-ohjelmointiympäristössä on mahdollisuus käyttää itse digitaalisesti piirrettyjä hahmoja ja kuvia sekä valokuvia osana itse tehtyjä ohjelmia. Graafisille ohjelmointialustoille ominaisesti ScratchJr:ssä ohjelmointi tapahtuu raahamalla ja liittämällä toimintokomentoja yhteen. Ohjelma toimii iPadeilla ja useimmilla Android-tableteilla ja on ladattavissa App Storesta ja Google Playstä. ScratchJr-ohjelmiston ovat kehittäneet yhteistyössä Tufts University ja Massachusetts Institute of Technologyn (MIT) Lifelong Kindergarten -tutkimusryhmä.



Scratch on ladattavissa ilmaiseksi osoitteesta *scratch.mit.edu/scratch2download* tai se on käytettävissä selainpohjaisena osoitteessa *scratch.mit.edu*.

Scratch

Tutustuttuasi ohjelmoinnin logiikkaan ScratchJr:n avulla voit jatkaa hieman monipuolisempaan graafiseen ohjelmointiin Scratch-ohjelmointiympäristössä. Scratchilla voit tehdä moniuoitteisempia pelejä tai vaikkapa oman interaktiivisen soittimen. Erilaisia käsky- ja toimintapalikoita on paljon enemmän kuin ScratchJr:ssä. Poiketen ScratchJr:stä Scratchissä värilliset toimintokomennot on lisäksi merkitty sanoilla. Vaikka ohjelma ei osittaisen tekstipohjaisuuden vuoksi sovellu aivan suoraan luku- ja kirjoitustaidottomille, on sitä mahdollista käyttää varhaiskasvatuksessa ja esiopetuksessa opettajavetoisesti. Scratch- ja ScratchJr-ohjelmointiympäristöjen perusperiaatteet ovat samat. Molemmissa koodipätkiä luodaan valmiita palikkoja yhteen liittäen. Palikoita luetaan pääsääntöisesti vasemmalta oikealle ja ylhäältä alas, mutta erilaisia ehtolauseita noudattaen.

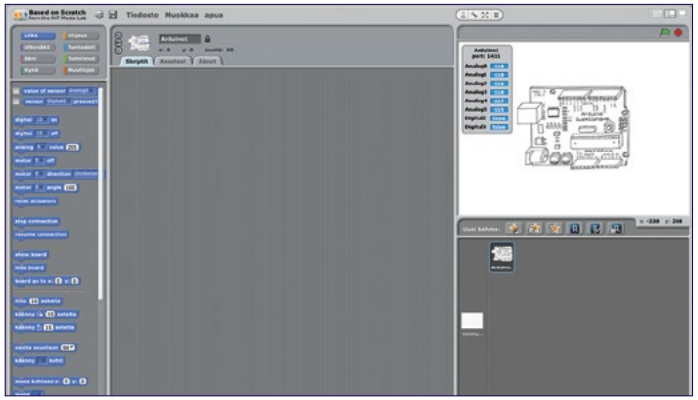
Scratch-ohjelma on ilmainen ja voit ladata sen osoitteesta *scratch.mit.edu/scratch2download*. Voit käyttää ohjelmaa myös selainpohjaisena sovelluksena, jolloin sinun ei tarvitse asentaa ohjelmaa koneellesi. Scratch-ohjelman on kehittänyt Massachusetts Institute of Technologyn (MIT) Lifelong Kindergarten -tutkimusryhmä.

S4A – Scratch for Arduino

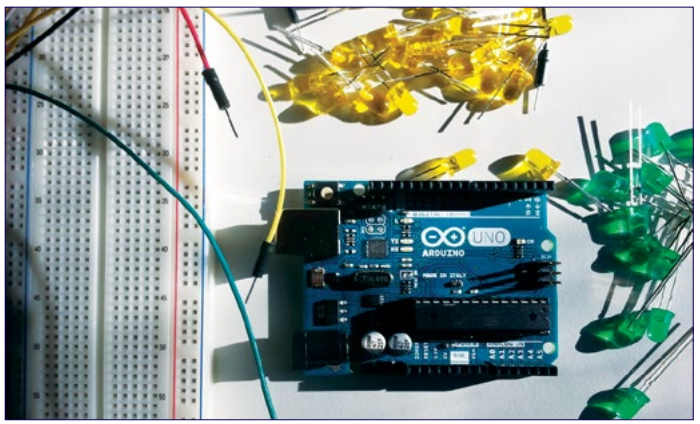
Scratch for Arduino kertoo jo nimessään mihin ohjelmistoa käytetään, eli Arduino-kehitysalustan ohjelmoimiseen. Lyhyesti Arduinon voi kuvailla olevan pienoistietokone, johon voi liittää erilaisia elektroniikan komponentteja, kuten ledejä tai moottoreita, ja jonka voi ohjelmoida ohjaamaan näitä komponentteja halutulla tavalla. S4A-ohjelmalla voimme ohjelmoida Arduinoa Scratchistä tutuilla graafisilla komentopalikoilla. S4A-ohjelmointiympäristön käyttäminen on hyvin helppoa varsinkin silloin kun olet ensin tutustunut Scratch-ohjelmointiympäristöön.

On olemassa myös muita Scratchiin perustuvia Arduinon ohjelmointiympäristöjä. Usein niiden käyttötavat ovat keskenään samanlaisia. Opettelemalla ensin yhden on sinun helppo tutustua myös toisiin. Vain kokeilemalla löydät omiin tarpeisiisi sopivimman ohjelmointiympäristön.

S4A-ohjelmointiympäristön voit ladata vain tietokoneelle (Windows, Mac, Linux) osoitteessa *s4a.cat*.



Arduino kehitysalusta.



Oppaassa käytetään Arduino kehitysalustoista nimenomaan Arduino UNO-mallia. Yhtälainen tuote löytyy myös nimellä Genuino UNO. Arduinosta on myös monia muita malleja UNO:n lisäksi, kuten Arduino Leonardo, Arduino Mega jne. Oppaan mallikuvat on tehty käyttäen Arduino UNO kehitysalustaa.

Jotta voit luoda kommunikointiyhteyden Arduino-kehitysalustan ja tietokoneella toimivan S4A-ohjelman välille, tulee sinun asentaa myös sopiva laiteohjelmisto Arduino- mikrokontrolleriin. Asennusohjeet löytyvät S4A:n kotisivuilta (s4a.cat) kohdasta “Downloads”.

Arduino-kehitysalusta

Arduino on edullinen ohjelmoitava kehitysalusta. Kehitysalusta on kuin pieni tietokone, jota pystyt itse hallitsemaan ohjelmoimalla sitä. Arduino-kehitysalusta esitellään perusteellisemmin sen käyttöön perustuvien tehtävien kautta. Arduinon käytön perusta rakentuu pinneistä, joihin voidaan liittää eri elektroniikan komponentteja. Voit esimerkiksi rakentaa oppaan ohjeiden perusteella oman sähköisen veistoksen, joka tuottaa valoa ja liikettä nappia painamalla.

Sähköturvallisuudesta

Elektroniikan rakentelu on hyvä tapa oppia sähkölaitteiden turvallista käsittelyä ja oppia sähköturvallisuudesta. Koska elektronisessa rakentamisessa ollaan tekemisissä sähkön kanssa, on aina kiinnitettävä huomiota sähkölaitteiden huolelliseen käsittelyyn. Tällöin vältytään esimerkiksi oikosuluilta, jotka voivat rikkoa laitteiston tai pahimmillaan aiheuttaa tulipalon.

Oppaan tarkoituksena on johdattaa havaitsemaan elektroniikan toimintatapoja ja tarjota kokemuksellisia lähtökohtia oivaltaa miten elektroniikkaan perustuva ympäristö rakentuu.
Rakentamisohjeet on tarkoitettu vain väliaikaisten kytkentöjen rakentamiseen kokeilujen toteuttamiseksi.

Oppaan lukija toteuttaa kytkennät aina omalla vastuullaan! Oppaan kirjoittajat eivät ota vastuuta mahdollisista vahingoista. Oppaassa esitetyt elektronisen rakentamisen kokeilut on aina syytä tehdä huolella ja aikuisen opastuksella. Mikäli kytkennässä ilmenee jotakin epäilyttävää, kuten komponenttien kuumenemista polttavan kuumaksi, poista virtalähde ja pura kytkentä.

Oppaan rakenteluideoissa käytetään ainoastaan pienjännitteisiä virtalähteitä eli paristoja, jotka ovat suhteellisen turvallisia. Älä koskaan liitä laitetta seinäpistokkeen kautta verkkovirtaan ellet ole käyttötavasta aivan varma. Verkkovirta on ihmiselle hengenvaarallista.

Poista paristo aina rakentamastasi laitteesta, jos sitä ei käytetä tai jos sen toiminnassa ilmenee jotakin omituista, kuten paristojen kuumenemista polttavan kuumaksi. Paristojen kuumeneminen on merkki oikosulusta ja siitä, että koneessa on jotakin vialla. Jos et löydä vikaa, poista kone kokonaan käytöstä. Säilytä paristot aina huolellisesti teippaamalla navat umpeen, jottei oikosulkua synny myöskään säilytyksen aikana. Arduino-tehtävissä käytetään virtalähteenä USB virtaa, jonka jännite on 5V. Tämä jännite vastaa alle neljää sormiparistoa. Irroita myös USB johto Arduinosta lopettaessasi laitteen käytön.

Älä kytke rakentamaasi laitetta sellaiseen virtalähteeseen, josta olet epävarma esimerkiksi jännitteen määrän suhteen. Liian pieni jännite ei vahingoita konetta, mutta liian suuri jännite voi helposti rikkoa laitteen elektroniset osat eli komponentit. Näin tapahtuu helposti esimerkiksi jos kytket led-valon suoraan 9 V:n paristoon.

Varmista myös ettet jätä litteitä 3 V:n nappiparistoja tai käytettäviä komponentteja perheen pienimpien ulottuville. Nieltynä paristot voivat olla hengenvaarallisia.

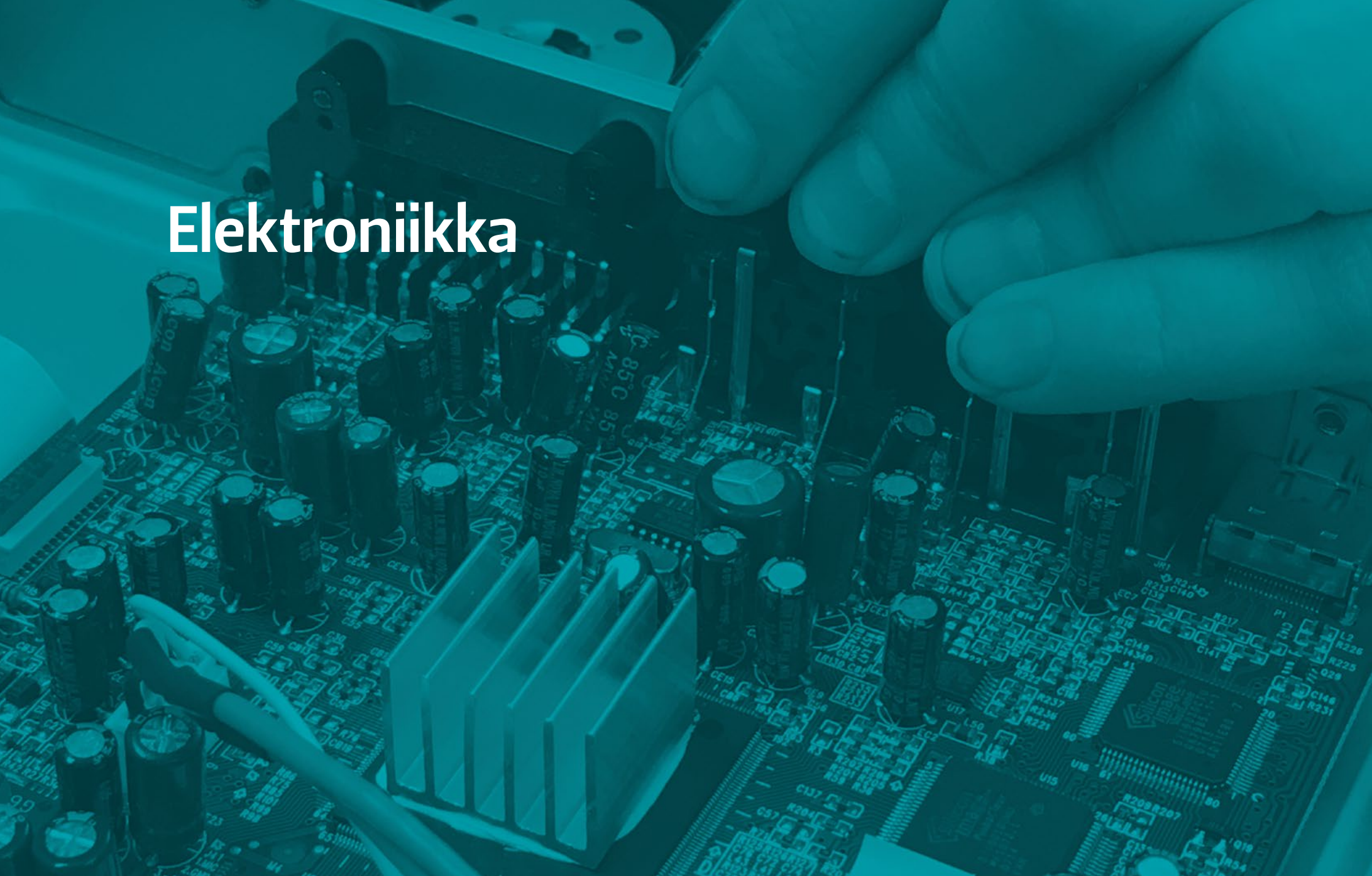
Ongelmatilanteet rakennettaessa

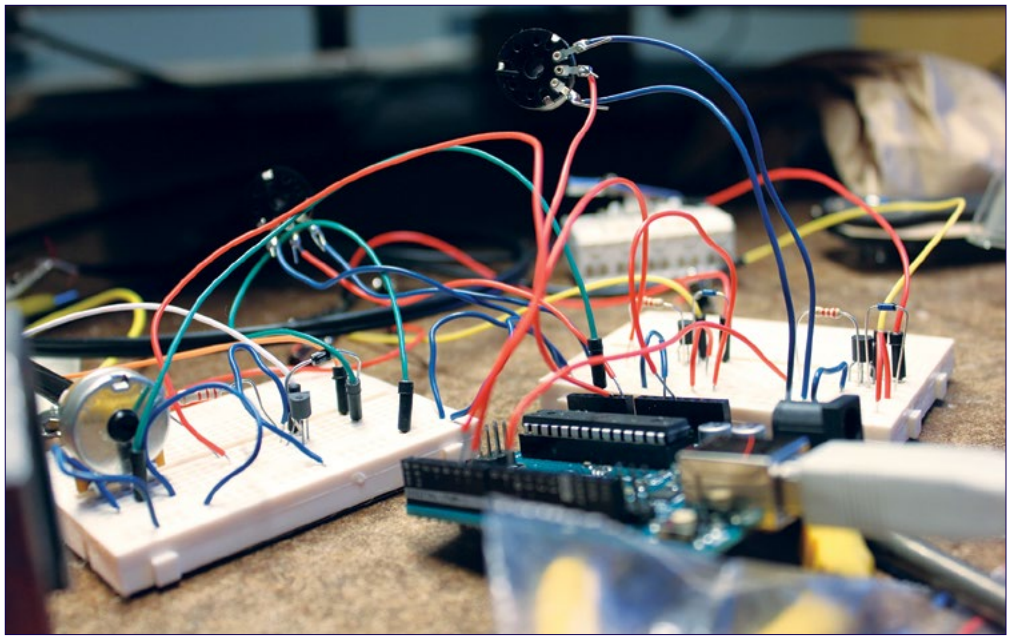
Toimittaessa elektroniikan ja ohjelmoinnin parissa on varsin yleistä, etteivät rakennettavat asiat toimi heti aluksi tai että ne lakkaavat jossakin vaiheessa toimimasta. Tämä on normaali osa prosessia. Sekä ohjelmointi että elektroninen rakentelu onkin usein virheiden etsimistä. Kun opit pikkuhiljaa tuntemaan paremmin elektroniikan ja ohjelmoinnin perusteita, helpottuu vikojen etsiminen.

Voi varsin hyvin yleistää, että yleisimmät virheet elektroniikassa ovat tyhjä paristo, väärin päin kytketty paristo, irronneet ja huonosti kytketyt johdot ja rikkoutuneet tai väärinpäin kytketyt komponentit.

Ohjelmoinnin osalta vika on useimmiten kirjoitusvirheessä, kuten puuttuvassa merkissä tai lauseraken-teessa. Ohjelmoinnissa saattaa esimerkiksi esiintyä virhe, jossa jokin komento toteuttaa kaksi toisensa poissulkevaa toimintoa samanaikaisesti, kuten vaikkapa napista painamalla led-valo menee sekä päälle että pois päältä. Ohjelmointi perustuu joko-tai-ajatteluun, jolloin jokin asia tapahtuu tai se ei tapahdu. Voit lukea ohjelmoinnin logiikasta lisää oppaan ohjelmointia käsittelevästä osiosta.

Elektroniikka





Sähkö ohjaa kaikkea, ohjataan siis sähköä

Kaikkien sähköisten laitteiden toimintaa, rakentamista, käyttöä ja suunnittelua kutsutaan yleisnimellä elektroniikka. Elektroniikan tavoitteena on usein toteuttaa jokin toiminta, kuten ohjata koneita tai siirtää tietoa. Elektroniikkaa voidaan käyttää myös osana taidetta, toteuttaen esimerkiksi liikkuvia eli kineettisiä veistoksia.

Kun huomiomme kiinnittyy tarkemmin vielä pienempiin elektroniikan osiin, puhutaan komponenteista. Komponentteja ovat yksittäiset osat, kuten esimerkiksi led-valo, sähkömoottori, painike ja vastus. Komponenteilla ohjataan sähkövirran kulkua, ne voivat toimia laitten aistimina tai niiden avulla käynnistetään esimerkiksi sähkömoottori tai valo. Elektroniikan kytkentöjen avulla sähkö saadaan kulkeutumaan aina sinne, mihin sen kulloinkin halutaan kulkevan. Voikin ajatella, että elektroniikan komponenteilla rakennetaan esimerkiksi tieverkosto, jossa on tietyt liikennesäännöt. Kun virtalähde kytketään kiinni, alkavat elektronit liikkumaan tieverkostolla sen komponenttien määrittelemien liikennesääntöjen mukaisesti. Onkin hauska yhteensattuma, että elektroniikan piirilevyt muistuttavat usein kaupunkien pienoismalleja tieverkostoineen.

Elektroniikka jaetaan usein digitaaliseen elektroniikkaan ja analogiseen elektroniikkaan. Analogisen ja digitaalisen ero on yksinkertaistettuna sitä, että analoginen on liukuvaa ja portaatonta kun taas digitaalinen on paloiteltua ja binääristä eli päällä tai pois päältä. Ero ilmenee havainnollisesti esimerkiksi analogista ja digitaalista kelloa katsoessa. Analogisessa kellossa voit nähdä viisarien liikkuvan sekuntienkin välillä, kun taas digitaalinen kello osoittaa vain tietyt ennalta määritellyt luvut numeerisina arvoina.

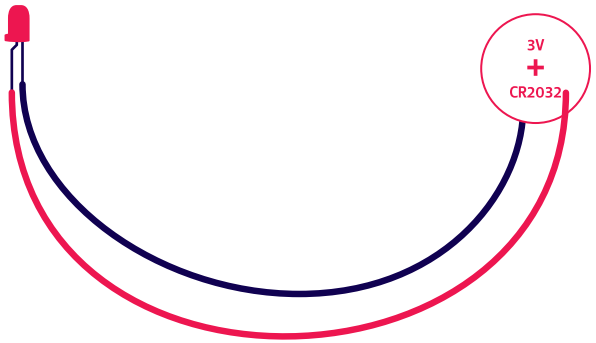
Digitaalisuuden perustana on jako binääriseen järjestelmään eli bitteihin. Olet ehkä kuullut puhuttavan ykkösistä ja nollista eli biteistä. Ykköset edustavat komentoa, että haluttu asia on päällä ja nolla puolestaan sitä, että haluttu asia ei ole päällä. Digitaalisessa elektroniikassa kaikki toimiikin sen perusteella, että asia on joko päällä tai pois, 1 tai 0. Emme kuitenkaan kohtaa näitä ykkösiä ja nollia ohjelmoitaessa, vaan käyttämämme ohjelmointikieli ja -ympäristö muuttavat koodin konekieleksi, joka koostuu biteistä.

Molemmissa tapauksissa, sekä digitaalisissa että analogisissa laitteissa, käyttämämme laitteen taustalla vaikuttaa elektroniikka joka perustuu sähkön ohjaamiseen. Voimme ohjata sähköä sekä kytkemällä komponentteja eri tavoin että luomalla koodin, joka määrittää onko sähkövirta päällä vai pois tietyssä komponentissa. Ymmärtääksemme teknologista elinympäristöämme paremmin on meidän hyvä perehtyä hieman sähkön ohjaamisen molempiin tasoihin, jotka eivät ole lopulta erotettavissa toisistaan.

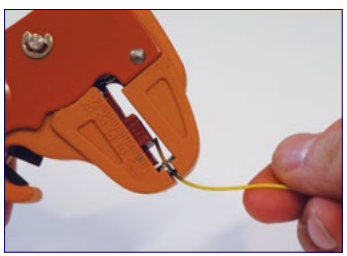
Virtapiiri solmussa

Tarvikkeet:

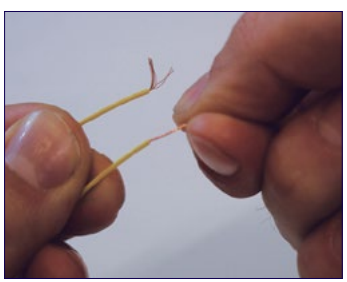
- Vähintään 10 metriä kaksinapaista sähkökaapelia – voit käyttää myös kahta sähköjohtoa kieputettuna yhteen
- Sivuleikkurit ja/tai kuorintapihdit
- Lattapihdit
- 3 V:n nappiparisto – esim. CR2032
- Led-valo
- 2 kpl sokeripaloja
- Pieni sokeripalaan sopiva ruuvimeisseli



Tässä ensimmäisessä leikin kautta tapahtuvassa elektronisen rakentelun tehtävässä tutustutaan elektroniikan perusteisiin sekä päästään rakentamaan ensimmäinen käytettävä laite – valo virtakaapelin päässä.

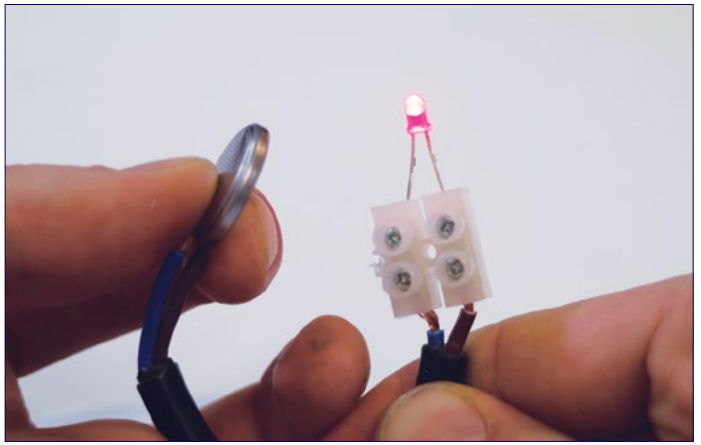


Valmistele tehtävä kuorimalla kaksinapaisen kaapelin molemmat päät auki käyttäen sivuleikkureita ja kuorintapihtejä. Kun useampia sähköjohtoja on yhdistetty samaan rakenteeseen, puhutaan kaapelista. Kaapelin suojamuovi lähtee parhaiten irti varovaisesti sivuleikkureilla leikaten, jonka jälkeen erilliset johdot on helppo kuoria kuorintapihdeillä. Paljastuva kuparilanka kannattaa pyörittää sormien välissä kierteelle, jotta se pysyy siististi kasassa.



Liitä led kaapelista tulevien johtojen toiseen päähän, toinen jalka toiseen ja toinen toiseen käyttäen sokeripalaa. Sokeripalan suuaukkoihin kiinnitetään halutut johdot ja suuaukkojen vastapuolille halutut komponentit. Johdot lukitaan kiristämällä sokeripalan ruuvit pienen ruuvimeisselin avulla. Tässä vaiheessa ei ole välttämätöntä välittää kumpi johto menee kumpaan ledin jalkaan.

Kieritä kaapeli rullan tai jonkin kappaleen ympärille, jotta sitä on helpompi käsitellä leikin aikana.

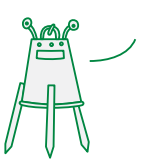


Tehtävän aluksi tutkikaa ensin paristoa. Millaisia merkintöjä paristosta löytyy? 3V kertoo pariston olevan kolmen voltin jännitteellä toimiva. +-merkintä kertoo pariston napaisuudesta eli siitä, miten päin se tulee kytkeä virtapiiriin. Nappipariston toinen puoli on miinusnapa, vaikka pohjassa ei ole tätä merkitty.

Istukaa aluksi rinkiin. Leikissä se, jolla on kädessään kaapelikerä, aloittaa kertomalla jonkin sähkölaitteen, jota on käyttänyt viimeksi. Mikä laite oli ja mitä sillä tehtiin? Tämän jälkeen hän antaa kaapelikerän ringin vastapuolella istuvalle samalla pitäen kiinni kaapelin siitä päästä, jossa on led-valo. Se, joka saa kerän, nimeää vuorostaan jonkin sähkölaitteen jota on käyttänyt, ottaa kaapelista kiinni omalta kohdaltaan ja antaa lopun kaapelikerän eteenpäin seuraavalle. Tavoitteena on, että kaapeli risteytyy ringin sisällä mahdollisimman paljon. Näin jatketaan kunnes viimeinenkin on saanut käteensä kaapelin pään ja nimennyt käyttämänsä sähkölaitteen.

Viimeisenä kaapelin saanut henkilö kytkee pariston johtoihin oikein päin niin että led syttyy. Paristoa saa kokeilla kumminpäin vain, mutta vain toinen kytkentä toimii.

Kaapelikerää aloitetaan kerimään vastakkaisessa järjestyksessä siten, että jokainen saa vuorollaan opetella pariston oikeinpäin kytkemistä. Tutkikaa ledin rakennetta. Ledin toinen jalka eli plusjalka on pidempi kuin toinen. Tästä tunnistaa miten päin led kytketään paristoon. Myös miinusjalka tulee kytkeä paristoon, jotta virtapiiri muodostuu ja laite toimii!

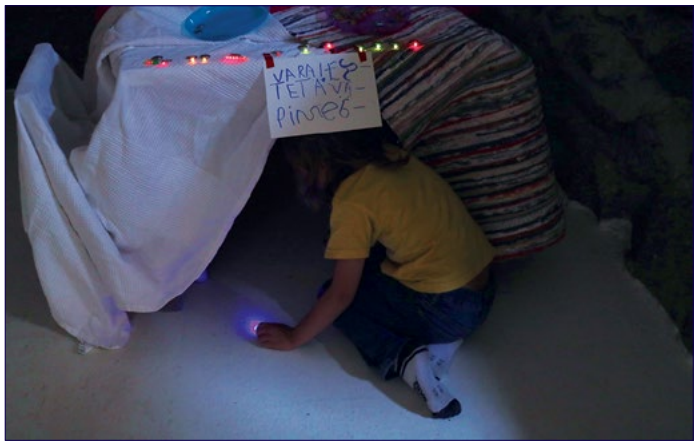


Miten sähkö löytää ledin luokse, vaikka johdot ovat aivan sekaisin? Kokeile piirtää kartta paristosta lediin ja ledistä takaisin paristoon. Miten johdot mahtavat kulkea seinien sisällä esimerkiksi kattolamppuun? Entä muissa erilaisissa laitteissa?

Valosivellin

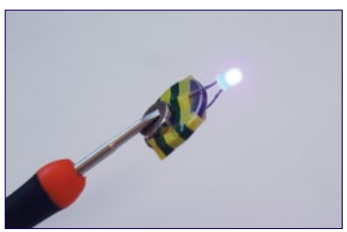
Tarvikkeet:

- 3 V:n nappiparisto – esim. CR2032
- Led-valoja
- Teippiä
- Kamera jossa on säädettävä valotusaika. Älypuhelimille, tableteille ja tietokoneille on myös saatavissa erilaisia valomaalausapplikaatioita – voit hakea niitä hakusanalla "light painting".



Kokeilkaa oman ledin kytkemistä omaan paristoon suoraan ilman sähköjohtoa. Ledejä kannattaa varata reilusti ja eri värisinä, sillä voitte kokeilla useamman ledin liittämistä samaan paristoon. Rakentakaa valosiveltimet ja tehkää valomaalauksia ottamalla valokuvia pitkällä valotusajalla!

Tutkikaa aluksi lediä. Ledin jalat ovat eri mittaiset, jolloin se kertoo, että komponentin napaisuudella on merkitystä – aivan kuten edellisessä tehtävässä huomioitiin. Pidempi jalka tulee kytkeä pariston (+) napaan ja lyhyempi (–) napaan. Teippaa paristo ja led yhteen.



Led-taskulamppu soveltuu mainiosti jatkokehiteltäväksi esimerkiksi erilaisiin majaleikkeihin. Laittakaa esimerkiksi seikkailuradan luolan suuaukolle tehtäväksi rakentaa oma led-taskulamppu.Kokeilkaa erivärisiä ledejä ja luokaa pimeään tilaan jännittävä valaistus!

Liittämällä kytkentään magneetin, saatte aikaiseksi metallipinnoille kiinnitettäviä valoja. Voisiko nämä kiinnittää heittämällä vaikkapa hetkellisesti metalliseen valotolppaan tai seinään? Muistakaa kuitenkin huolehtia osien siivoamisesta ja kierrättämisestä.

Katsokaa etteivät nappiparistot ole perheen pienimpien ulottuvissa! Paristot ovat nieltäessä hengenvaarallisia!

Aistivat koneet

Tässä leikissä pohditaan millaisia laitteita ympärillämme on ja miten ne toimivat reagoidessaan käyttäjään. Leikissä toimitaan tiettyjen ehtojen mukaisesti aivan kuten elektroniset laitteet. Samalla pohditaan erilaisten sensorien mahdollisuuksia reagoida ympäristön kanssa eri tavoin.

Tavoitteena on ymmärtää, että koneissa on usein jokin sensori eli tunnistin, kuten painike, joka tuntee kun jotain tapahtuu ja joka välittää tai päästää tiedon kulkemaan eteenpäin. Sensoreita on monenlaisia moniin eri tarkoituksiin. Sensori voi esimerkiksi tunnistaa, kun sitä painetaan tai kun esine kallistuu, tai vaikka kun valoisuudessa tai lämpötilassa tapahtuu muutos. Yksinkertainen ympäristöstämme löytyvä sensori on valokatkaisija. Valokatkaisija reagoi mekaanisesti katkaisemalla tai avaamalla virtapiirin, kun sitä painetaan, jolloin sähkö pääsee kulkemaan sen läpi.

Valitkaa yksi vapaaehtoinen, joka on kone. Muut ovat koneen testikäyttäjiä.

Ensin sovitaan millaisia sensoreita koneesta löytyy ja mitä mikäkin sensori tekee. Sensori voi olla esimerkiksi painokytkin eli nappi tai vaikkapa valoisuuskytkin, jolloin valojen sammuessa tapahtuu jotakin. Esimerkiksi navasta painamalla kone avaa ja sulkee silmät, niskaan puhaltamalla kone menee kyykkyyn, korvaan kuiskaamalla se avaa oven ja päätä silittämällä alkaa pitämään piipittävää ääntä. Muista määritellä tarkkaan tapahtuuko toiminto kerran vai toistuvasti sensorin reagoidessa. Esimerkiksi avaako kone oven yhden kerran auki, kun sen napaan painetaan, vai avaako se ovea koko ajan edestakaisin! Toimintojen kannattaa olla mahdollisimman yksinkertaisia.

Leikissä jokainen käyttää konetta vuorollaan yhtä sensoria painamalla. Kone reagoi sensorin käyttöön tarkasti. Suunnitelkaa ja parannelkaa koneen toimintaa yhdessä! Kokeilkaa ensin ryhmässä ja sitten pareittain.

Sensoreihin lisätään herkkyysominaisuus, jolloin ne lukevat kuinka lujaa jotakin toimintoa tehdään ja reagoivat sen mukaisesti. Kevyt puhallus laittaa koneen menemään kyykkyyn hitaasti, kun taas kova puhallus laittaa liikkeeseen lisää vauhtia.

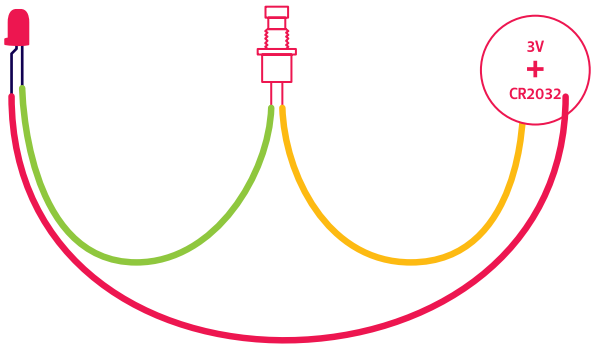
Yhdistetään useampi kone yhdeksi suuremmaksi koneeksi. Koneiden toimintoja voi yrittää yhdistää siten, että koneet reagoisivat myös toistensa toimintoihin. Esimerkiksi kun ensimmäisen koneen sensoria kosketaan, liikuttaa se kättä toisen koneen sensorin eteen, joka puolestaan päästää äänen.

Keksijän hatut päähän! Suunnitelkaa maailman hulluin laite!

Sähköinen veistos

Tarvikkeet:

- Led
- Painikenappi
- 5 x hauenleukajohdot
- Sähköjohtoa
- 3V nappiparisto – esim. CR2032
- Kuorimapihdit
- Sivuleikkurit
- Pahvilaatikko kotelointiin
- Ruuvimeisseli, kynä tai muu terävä esine ledin ja painikkeen reikien tekemiseksi pahvilaatikkoon.
- Sokeripaloja



Seuraavassa tehtävässä rakennetaan sähköinen veistos. Tässä tehtävässä opit tärkeitä taitoja jotka liittyvät elektroniseen rakentamiseen. Elektroniikkaa rakennettaessa on aina ensin hyvä rakentaa kokeilukappale eli prototyyppi. Kokeilukappaleen voit valmistaa rakentamalla kytkennät hauenleukojohtojen avulla. Prototyyppi ei kuitenkaan sovellu pysyväksi rakenteeksi, sillä sen liitokset saattavat helposti pettää.

Laitteen toiminta perustuu virtapiiriin, joka saa sähkövirran kulkemaan virtapiirissä oleviin komponentteihin – tässä tapauksessa lediin.

Aloitakaa leikkaamalla ja kuorimalla sähköjohdosta kaksi kappaletta noin viiden senttimetrin mittaista johdon pätkää. Kuorikaa johtojen päät noin puolen senttimetrin matkalta.

Asettakaa hauenleukajohdot lyhyiden johdonpätkien jatkoiksi. Kuvassa toisesta johdonpätkästä tulee keltainen hauenleukajohto ja toisesta punainen hauenlaukajohto.

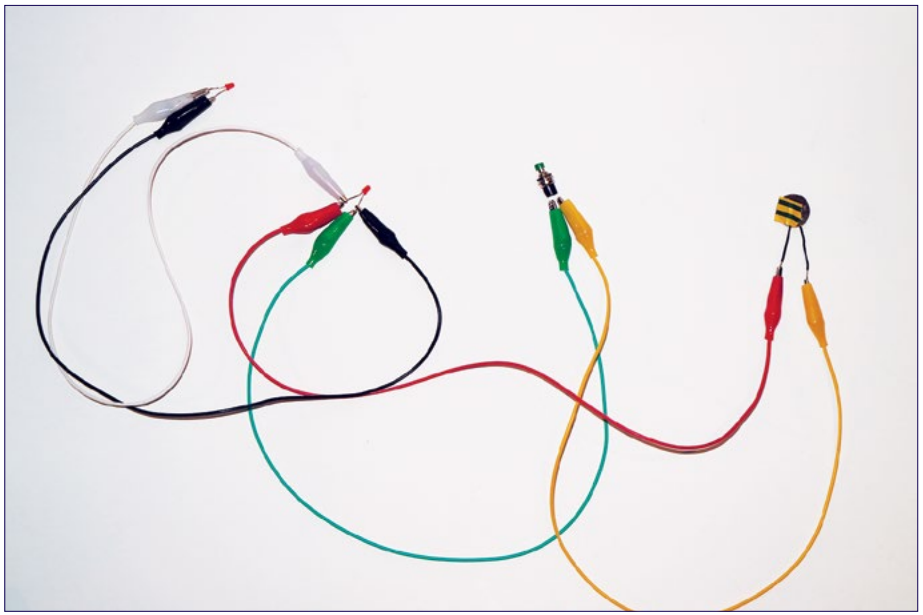
Teippaa johdonpätkät pariston eri pinnoille ja pidä mielessä, kumpi johto on pariston (+) puolella ja kumpi (–) puolella.

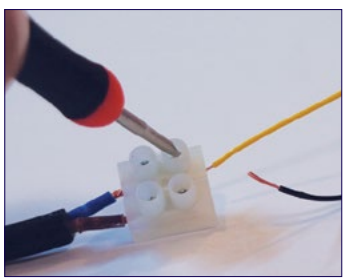
Kytkekää toinen hauenleukajohto, joka tulee pariston (+) puolelta, ledin pidempään jalkaan.

Seuraavaksi kytkekää ledin lyhyemmästä jalasta johto painikenapin toiseen jalkaan (ei väliä kumpaan). Painike rakentuu kahdesta jalasta, jotka yhdistyvät kun napista painetaan.

Lopuksi liittäkää hauenleukajohto myös painikenapin vapaan jalan ja pariston (–) puolelta tulevan johdon väliin.

Mikäli valo ei syty, on todennäköisin syy se, että paristo on kytketty väärinpäin. Mikäli valo palaa koko ajan, on todennäköistä että painikenapin kannoilla olevat hauenleukajohdot osuvat toisiinsa. Tarkistakaa etteivät hauenleukajohdon päät osu toisiinsa vahingossa missään kohtaa kytkentää.





Liittäkää sähköiseen veistokseen lisää ledejä kytkemällä ne rinnankytkennällä toisiinsa. Tämä tarkoittaa, että edellisen ledin (+) jalasta liitetään johto uuden ledin (+) jalkaan. Vastaavasti (–) jalasta siirrytään myös uuden ledin (–) jalkaan.

Mitä ledit voisivat olla veistoksessa? Millainen lopullinen veistos tulee olemaan. Mihin led asetetaan? Entä painike?

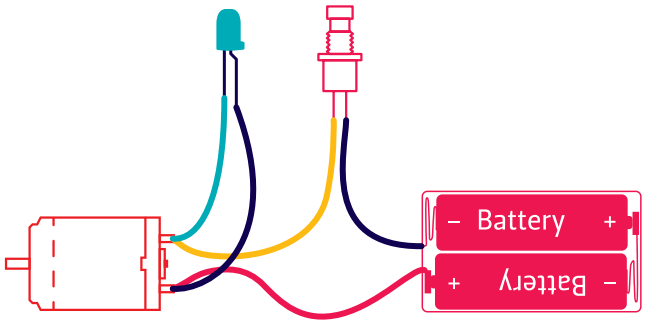
Nappipainikkeita on myös nelijalkaisina. Näissä painikkeissakin on kaksi napaa vaikka jalkoja on neljä. Nelijalkaisen painikkeen pohjassa on yleensä viiva. Viivan toisella puolella olevat jalat ovat sama napa kun taas toisella puolella olevat jalat ovat toinen napa.

Kun kytkentä toimii halutulla tavalla, voitte rakentaa siitä kestävämmän version, jonka voi siirtää laatikon sisälle. Kestävämpi versio rakennetaan käyttämällä sokeripaloja ja sähköjohtoa hauenleukajohtojen sijaan.

Kineettinen veistos

Tarvikkeet:

- Kahden AAA-pariston paristokotelo
- 2 × AAA paristoja
- 1 x Pienoissähkömoottori (käyttöjännitteen oltava 3V ja se voi olla ilmaistuna esim. 1,5–4.5V)
- 5 × hauenleukajohtoja
- 1 x sininen tai valkoinen led, jonka kynnysjännite on 3 volttia
- 1 x painikenappi



Tehtävässä rakennetaan kineettinen veistos. Kineettinen veistos tarkoittaa veistosta, jossa on liikettä.

Sähkömoottorin kytkentä rakennetaan samalla periaatteella kuin ledin, jolloin toinen paristosta tuleva johto tulee moottorin toiseen "korvaan" ja toinen johto vastaavasti toiseen.

Voit kiinnittää osat hauenleukajohdoilla, kuten teit edellisessäkin tehtävässä tai käyttää liitoksiin sähköjohtoa.

Sähköjohdoilla kiinnitettäessa johtojen kiinnitys moottoriin tapahtuu pyörittämällä johdon kuorittuja osia liittimen ympäri ja esimerkiksi kuumaliimaamalla liitoskohta tukevaksi ja suojatuksi. Varo pyörittämästä liian lujaa, jotteivät moottorin korvat katkea. Mikäli sinulle on tuttua tinan juottaminen kolvilla, niin tällöin suosittelemme liittämään osat juottamalla

Tutki sähkömoottorin toimintaa. Aloita liimaamalla teipistä pieni "lippu" sähkömoottorin akseliin. Tämä helpottaa hahmottamaan moottorin pyörimistä. Jatka kytkemällä moottorin korviin kiinni virtalähteen (1,5–4,5 V:n paristo) johtimet.

Muista, että sähkömoottorit pyörivät todella lujaa! Jos haluat hitaammin pyörivän moottorin, on yksi helppo vaihtoehto käyttää vaihteistomoottoreita. Nopean sähkömoottoriin voi liittää myös epäkeskopyörän, jolloin veistoksen voi saada liikkumaan tärisemällä. Nämä ohjeet löytyvät Robo1-julkaisusta

Tämän kineettisen veistoksen kytkentä on esimerkin mukaan tehty. Vihreän ledin päähän on pujotettu juomapillistä leikattu "suutin", sekä moottorin päälle kiinnitettty omaan paristoon kytketty punainen ledi.

Moottorin käynnistyttyä vaihda johtimien paikkoja keskenään. Tällöin moottorin pyörimis-suunta vaihtuu.

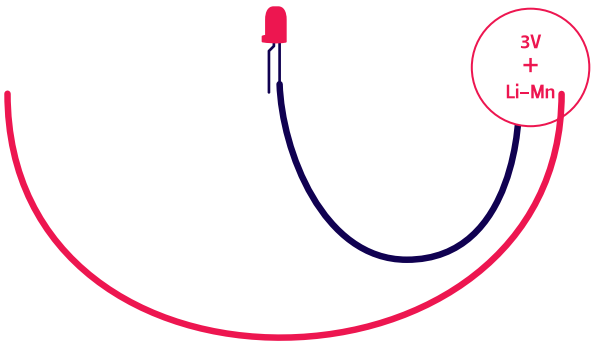
Muista käsitellä moottoria varoen ja vältä sen osumista mm. hiuksiin, jotta hiukset eivät takerru moottoriin kiinni.

Liitä lopuksi moottorin "korvista" lähtemään johdot ledille. Liitä ledin pidempi jalka jatkoksi (+) johtoon ja lyhyempi jalka jatkoksi moottoriin tulevaan (–) johtoon.

Mikä johtaa sähköä?

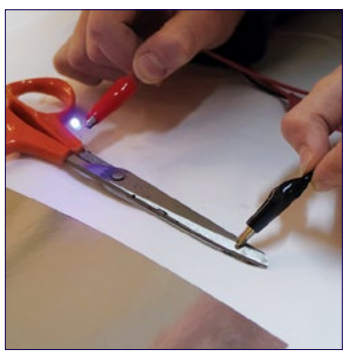
Tarvikkeet:

- 1 x 3V nappiparisto
- 1 × led
- 2 × hauenleukajohtoja
- 2 x sähköjohdon pätkä

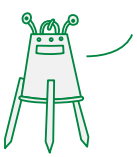


Hyvä tapa tutkia sähkönjohtavuutta eli sitä, mikä materiaali johtaa sähköä, on käyttää lediä ja virtapiiriä siten, että kytkimenä toimii sähköä johtava tai johtamaton materiaali.

Käytä sähköjohtoja pariston liittämiseksi hauenleukajohtojen päihin. Teippaa johdon päät pariston eri navoille. Jatka pariston (–) navasta lähtevästä johdonpätkästä hauenleukajohto ledin lyhyempään eli (–) jalkaan. Kytke ledin vapaan jalan sekä vapaan hauenleukajohdon liittimen väliin koekappale, pitämällä niitä kappaleen pinnalla.



Mikäli led syttyy palamaa kirkkaana, johtaa kappaleen pinta sähköä hyvin. Mikäli led palaa himmeästi, johtaa materiaali sähköä mutta hieman heikosti. Mikäli led ei pala, on materiaalin sähkönjohtavuus heikko tai se ei johda sähköä ollenkaan.



Mikä on yhteistä sähköä johtaville materiaaleille?

Taiteilijahaastattelu – Pekka ja Teija Isorättyä

Kuva: Pekka ja Teija Isorättyä

Pekka ja Teija Isorättyä ovat Torniosta kotoisin oleva taiteilijapariskunta. Isorättyöiden taiteen keskeinen materiaali on sähköinen ja mekaaninen teknologia, joka saa veistokset liikkumaan.

Miten kuvaisitte taidettanne ja taideteoksianne?

Olemme käyttäneet ilmaisuja kineettinen veistos, elektromekaaninen veistosinstallaatio ja taiderobotti.

Nostatte teoksissanne usein esiin ihmisen, luonnon ja koneiden välisen suhteen. Miten teknologiaan tulisi mielestänne suhtautua?

Teknologian käyttö ja kehittäminen on ihmiselle hyvin luonnollista, voisi sanoa jopa lajityypillistä toimintaa. Ihmisen suhde teknologiaan on pysynyt läpi historian jokseenkin samankaltaisena: varhaisista tulentekovehkeistä, jousipyssyistä ja kampakeramiikasta älypuhelimiin saakka teknologia on ollut ihmisen tapa ratkaista ongelmia ja laajentaa ja kehittää toimintakykyään.

Ihmisen luontosuhde sen sijaan on muuttunut: vielä reilut sata vuotta historiassa taaksepäin ihminen koki luonnon vastustajana. Tämä näkyy esimerkiksi Herman Melvillen

Kuva: Jussi Mankkinen

Kuva: Juha Arman

Moby Dick -romaanissa; nykyään on ihan käsittämätöntä että valaat nähdään vuonna 1851 kirjoitetussa kirjassa hirviöinä joita rohkeat miehet oikeutetusti tappavat.

Tällaiseen maailmankuvaan perustui kuitenkin koko teollisen aikakauden valtava tekninen kehitys.

2000-luvulla ihminen on alkanut viimein ymmärtää, että luonto on hauras ja sitä on päinvastoin varjeltava ja suojeltava kaikin tavoin. Toivottavasti ei ole liian myöhäistä. Tehokkain keino suojella luontoa on keksiä uutta luonnonmukaista teknologiaa. Nuorille ja lapsille haluaisin sanoa, että jos haluat suojella luontoa, opiskele tekniikkaa, ryhdy insinööriksi, joka keksii luontoa säästäviä ja suojelevia koneita vanhojen saastuttavien tilalle.

Miten elektroniikka ja muu käyttämänne teknologia soveltuu taiteelliseen työskentelyyn?

Taidetta voi tehdä mistä tahansa materiaalista. Roskan käyttö taiteen materiaalina voi viitata vaikka kierrätykseen mutta yhtä hyvin välinpitämättömyyteen. Öljyväri viittaa traditioon ja ylevyyteen ja taiteeseen itseensä, kun taas vesiväri on hieman sitä leikkisämpi ja arkisempi. Sähkötekniikan käyttö taiteessa viittaa teknologiaan sinänsä mutta myös yhä science fictioniin ja tulevaisuuteen. Esimerkiksi sähkötekniikan yhdistäminen johonkin orgaaniseen materiaaliin tuo mieleen hybridit.

Miten kuvailisitte teoksen valmistumisprosessia omalla kohdallanne?

Meidän taide lähtee liikkeelle usein jonkin paikan, ympäristön tai tilanteen tarkastelusta. Pohdimme minkälaisilla materiaaleilla ja muodoilla voisimme sitä parhaiten kuvata. Jos valitsemamme materiaali ja muoto mahdollistaa myös liikkeen rakentamisen teokseen, mietimme minkälaisella tekniikalla voisimme liikkeen parhaiten toteuttaa. Siinä on tärkeää, että rakennettu mekaaninen liike ja käytetty tekniikka on tasapainoinen veistoksen aiheen ja muun materiaalin kanssa.

Kertokaa teoksestanne "Merenneito" (2015).

Merenneito on työ jonka teimme Japanissa vuonna 2015. Sen kuori on tonnikalan nahkaa ja silkkiä ja se keinuu ja tanssii jäljitellen japanilaista butotanssia.

Merenneidon sydän on rakennettu auton tuulilasinpyyhkijän moottorista ja verenpaine-mittareiden käsipumpuista. Sydän pumppaa ilmaa veistoksen keskellä olevaan pussiin, joka täyttyessään nostaa merenneidon kalanpuoleista selkää ylöspäin, jolloin sen lantion mekaniikka työntää ylävartaloa eteen ja päätä taaksepäin kunnes lantio kosket-taa rajakatkaisinta, jolloin merenneito puhaltaa ilmapussinsa tyhjäksi nokkahuilun läpi aiheuttaen pitkän äänekkään huokauksen tai vihellyksen.

Merenneidon sisällä on myös neljä vanhoista pesukoneista peräisin olevaa vesipumppua, jotka toimivat 24 V:n DC-sähköllä. Ne pumppaavat kukin vuorollaan nestettä merenneidon pyrstön ja muiden jäsenien vastapainoiksi aseteltuihin pulloihin, joita on yhteensä kymmenen. Täyttyessään pullot nostavat jäseniä ja tyhjentyessään ne laskevat niitä. Merenneidossa on lisäksi liikkeentunnistin, kolme akkua, viisi rajakatkaisinta, viisi relettä, neljä aikarelettä sekä kymmeniä metrejä sähköjohtoja ja letkuja.

Ohjelmointi–leikit

Ohjelmointileikit – Koodausta ilman konetta

Ohjelmointi on toimintaohjeen kirjoittamista tietokoneelle – se kertoo koneelle mitä tehdä ja milloin.

Ohjelmointia voi kuvailla niin, että se on kuin reseptin antamista tietokoneelle. Reseptin perusteella tietokone leipoo ohjelman juuri niin tarkasti kuin reseptissä kerrotaan.

Toisaalta ohjelmointia voi tarkastella kuin kielenä. Se on vain digitaalisten koneiden kieli. Kielellä on tietty sanasto ja rakenne, mutta sitä voidaan käyttää myös luovasti luoden uusia yhdistelmiä olemassaolevista käsitteistä – aivan kuten ohjelmoitaessakin. On hyvä muistaa, että ohjelmoitaessa on monia tapoja tehdä sama asia eikä yhtä oikeaa ratkaisua ole. Lopulta tärkeää on, että koodattu ohjelma toimii. Mutta joskus myös väärin toimiva koodi saattaa osoittautua mielenkiintoiseksi mahdollisuudeksi kehittää ohjelmaa aivan uuteen suuntaan. Älä siis pelkää epäonnistumista.

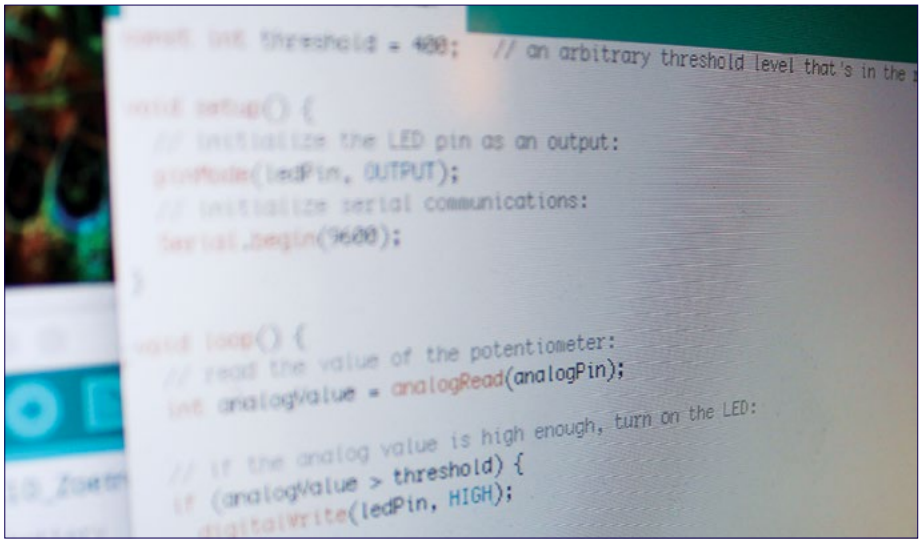
Kaikilla kielillä on myös oma tapansa kertoa asioita ja jäsennellä maailma, aivan kuten ihmisten eri kieletkin jotka ovat muotoutuneet eri kulttuureissa juuri omanlaisikseen. Niin myös eri ohjelmointikielet on kehitelty tiettyjen asioiden ohjelmoimiseen, ja tietyt ohjelmointikielet ovatkin optimaalisia juuri tiettyihin toimenpiteisiin ja tarkoituksiin. Ohjelmointikielen rinnastaminen ihmiskieleen paljastaa myös sen, että ohjelmointikielet ovat ihmisen luomia eivätkä jotakin ihmisistä erillistä. Kielen kehittäjät ja ohjelmiston luojat ovat päättäneet, kuinka ohjelma tai laite toimii ja mitä sillä voi määritellä.

Ohjelmoinnista voi ajatella myös, että luotu koodi on kuin koneen ajatukset muun laitteiston eli elektroniikan ollessa sen keho. Digitaalisessa maailmassa ohjelmoidut "ajatukset" ovat aina vain lopulta joko kyllä tai ei – binäärisesti ilmaistuna joko 1 tai 0. Tietokoneen aivot voivat esimerkiksi lähettää komennon johtoja pitkin käskien moottorin mennä päälle ja pois päältä samalla tavalla kuin voimme ajatella itsekin liikuttavamme jalkaa. Kuitenkaan ei tule erehtyä ajattelemaan, että ihminen olisi kuin ohjelmoitu tietokone.

Tämän luvun leikeissä ja tehtävissä näkökulmana on esitellä ohjelmoinnin logiikan perusteita. Ohjelmointileikeissä korostuu myös ohjelmoidun digitaalisen maailman ja analogisen, ihmisten kokemusmaailmaa lähempänä olevan jäsentymistavan, välinen vuorovaikutus.

Tehtävien keskeinen tavoite on luoda perustaa pohdinnoille ohjelmoidun maailman vaikutuksesta meihin – miltä se saa asiat tuntumaan ja näyttämään – sekä herätellä pohtimaan miten me ihmiset olemme vuorovaikutuksessa toisiimme ja maailmaan ympärillämme ohjelmoitujen laitteiden ja järjestelmien kautta.

Koodia kaikkialla!



Tässä tehtävässä huomio kiinnittyy ympäristössä olevaan ohjelmointiin. Mistä kaikkialta löytyy ohjelmointia? Missä ja miten se näkyy? Bongatkaa palohälytin, pesukone, elektroninen ovi, automaattinen valo jne. Tehtävässä oleellista on huomioida ohjelmoinnin yleisyys ympäristössä ja pohtia, mitä ohjelmoinnilla tehdään.

Leikin kulku:
Käy läpi esimerkkejä ohjelmoiduista asioista. Ohjelmoituja asioita ovat kaikki elektroniset laitteet ja järjestelmät, jotka tekevät asiat automaattisesti ihmisen puolesta. Näitä ovat esimerkiksi savun havaitseminen (palohälytin), oven avaaminen (automaattinen ovi), lämpötilan tarkka mittaaminen (digitaalinen lämpömittari), pesuohjelma (pesukone) ja automaattinen lukko (kulkulupakortilla toimiva lukko).

Leikin voi muuntaa myös kilpailuksi, jossa kisataan siitä, kuka löytää ensimmäisenä kymmenen asiaa joissa on ohjelmointia.

Taidekone

Tarvikkeet:
- Iso pahvilaatikko ja askartelu-välineitä
- Piirustuspaperia
- Tusseja ja kyniä
- Vapaavalintaisia kuvia tulkittaviksi
- Vapaasti sovellettavaa elektroniikkaa rakenteisiin, kuten led-valoja

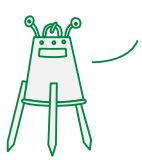
Leikissä luodaan omia komentoja ja tulkintoja, joilla määritellään millainen kuva piirtokoneesta tulee tulostaa. Mikäli käytössä on ruutupaperia, voidaan leikissä hyödyntää myös pikselileikin ajatusta komentorivillä kirjoittamisesta. Leikissä syntyy kuvan sanallistamis- ja tulkintatilanteita, joissa pohditaan kuinka hyvin kuvaus osuu yhteen yhdessä esitettyjen tulkintojen kanssa.

Leikissä valitaan yksi henkilö koneen sisälle tietokoneeksi, joka noudattaa hänelle annettuja piirtämisohjeita mahdollisimman tarkasti. Taidekoneen sisällä oleva henkilö ei näe alkuperäistä kuvaa, mutta kuulee ääneen lausutut tulkinnat kuvasta. Ääneen lausuttujen kuvausten perusteella kone tulkitsee, piirtää ja tulostaa uuden kuvan tulostusaukkoa pitkin.

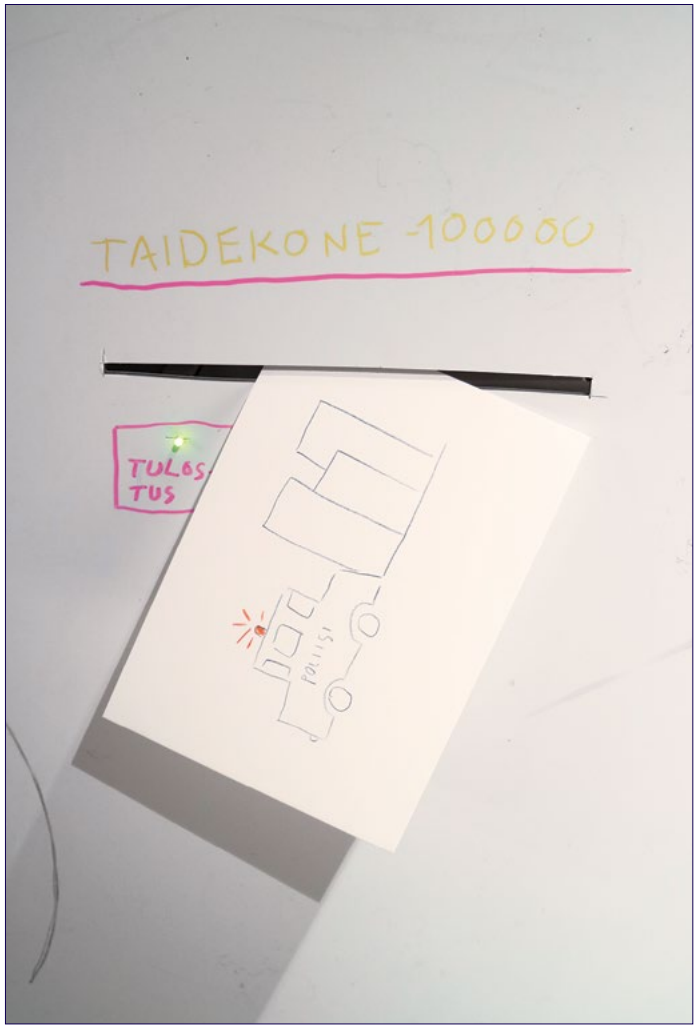
Valmistelu:
Lavastakaa iso pahvilaatikko taidekoneeksi. Taidekoneen kyljessä on tulostusaukko. Lisäksi taidekoneen tietokoneen (oppilaan) tulee olla helposti vaihdettavissa, joten jonkin kyljen kannattaa toimia koneen luukkuna. Kannattaa myös huolehtia, että sisälle pääsee tarpeeksi valoa jotta taidekone näkee piirtää.

Leikin kulku:
Ryhmässä katsotaan ja tulkitaan kuvaa. Kone kuuntelee mitä tulkintoja ja huomioita kuvasta kerrotaan. Tulkinnat ja huomiot voivat olla esimerkiksi: “Kuvan keskellä on musta koira. Koiran suussa on luu. Koira katsoo oikealle. Koiralla on punainen kaulapanta. Koiran häntä heiluu. Koiran vasemmalla puolella on puu.”

Yksi henkilö valitaan aina kerrallaan tietokoneeksi taidekoneen sisälle. Tämä henkilö ei saa nähdä alkuperäistä kuvaa. Tietokoneen tehtävänä on tulkita koneeseen syötetty ääniviesti (muiden kuvailut kuvasta) ja piirtää viestin mukainen kuva juuri sellaisena kuin sen kuvailujen perusteella ymmärtää. Tavoitteena on pohtia, miten muotoilla mahdollisimman tarkka kuvaus halutusta toiminnasta – tässä tapauksessa piirettävästä kuvasta luotu sanallinen kuvaus. Kuvailijat päättävät milloin kuva on valmis tulostettavaksi. Taidekone tulostaa valmiin kuvan taidekoneen tulostusaukosta. Aina jokaisen kierroksen lopuksi verrataan alkuperäistä kuvaa ja taidekoneen tulostamaa kuvaa yhdessä. Väärinymmärrykset lisäävät leikin hauskuutta!

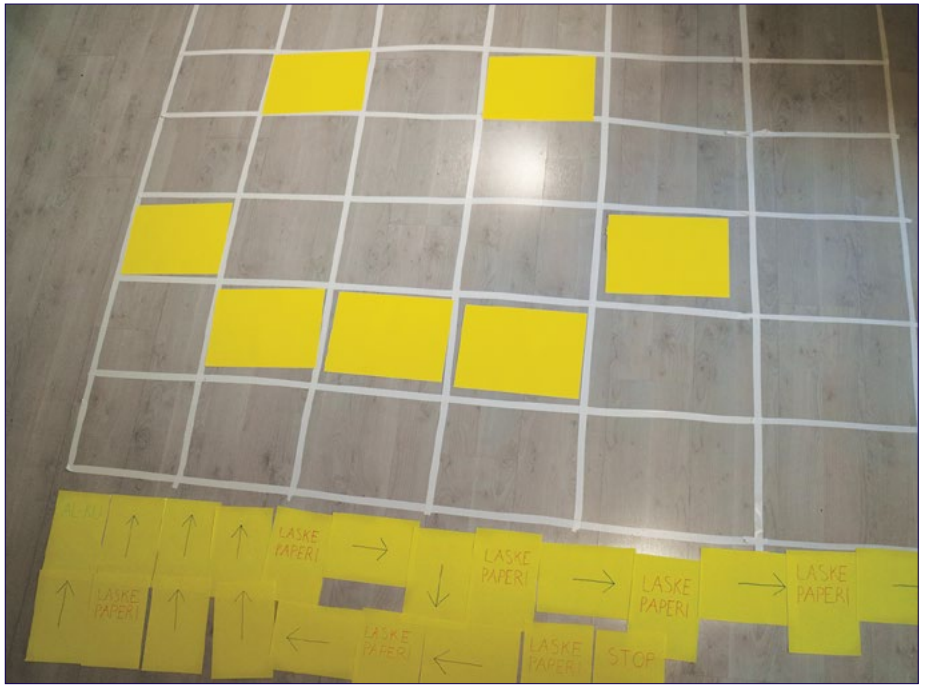


Kiinnittäkää huomio tarkkoihin ja yksiselitteisiin komentoihin!



Ohjelmointileikit

Pikselileikki



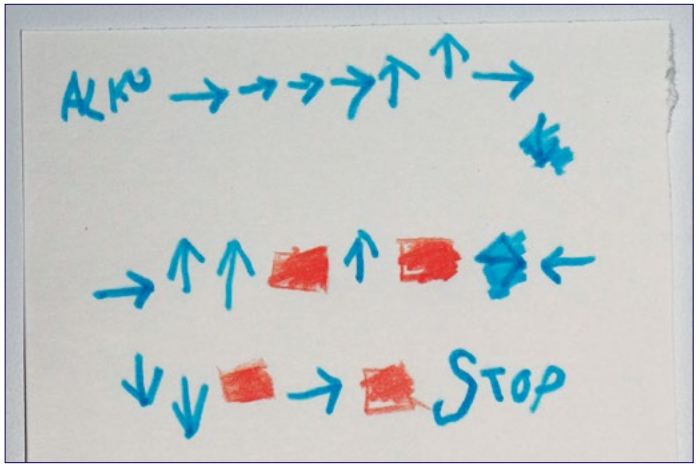
Tarvikkeet:

- Maalarinteippiä
- Paperille piirrettävät olevat komentomerkit.
- Erivärisiä papereita

Tässä leikissä tutustutaan kohteen liikuttamiseen tarkoin komentosarjoin eli skriptein. Liikkuminen tapahtuu siten, että yksi oppilas on vuorollaan pikseliruudukossa liikkuva "robotti", joka toimii komentojen perusteella. Liikkuminen tapahtuu 6×6-ruudukossa, jossa jokainen ruutu edustaa yhtä pikseliä. Digitaalinen kuva rakentuu monista pikseleistä, ja esimerkiksi digitaalisissa valokuvissa on usein miljoonia pikseleitä. Nyt luodussa 6×6-ruudukossa on 36 pikseliä (6 × 6 = 36). Leikin tavoitteena on liikkua tiettyyn ruutuun komentosarjaa noudattaen ja luoda komentosarjan mukainen kuva "värjäämällä" haluttu pikseli paperilla. Voitte käyttää erivärisiä papereita.

Valmistelu:

Teipatkaa lattialle 6×6-ruudukko maalarinteippiä käyttäen. Voitte tarvittaessa luoda myös suuremman ruudukon. Piirtäkää komentomerkeiksi nuolia, joilla osoittaa eri suuntiin kulkeminen, sekä komentomerkki jolla lasketaan paperi haluttuun ruutuun. Piirtäkää myös komentomerkit komentosarjan alulle ja lopulle.

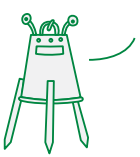


Leikin kulku:

Leikin aluksi opetellaan tulkitsemaan komentosarjoja. Alku-komento käynnistää liikkumisen aina samasta aloitusruudusta, jonka voitte määritellä itse. Aloitus voi tapahtua esimerkiksi pikseliruudkon vasemmasta alalaidasta. Tiettyyn suuntaan osoittava nuoli tarkoittaa, että liikutaan yksi pikseli kyseiseen suuntaan. Paperin laskemista kuvaava komentomerkki ilmaisee, milloin paperi lasketaan pikseliin. Voitte kehittää myös eri värisille papereille eri komentomerkkejä.

Leikki toteutetaan ryhmätyönä. Aluksi voidaan luoda sattumanvarainen pikselikuva arpomalla sattumanvarainen komentosarja. Kun komentosarja on arvottu, se levitetään maahan esille ja sitä luetaan vasemmalta oikealle. Tämän jälkeen valitaan yksi oppilas piirtorobotiksi. Piirtorobotille annetaan papereita, joita hänen on asetettava pikseleihin komentorivin mukaisesti.

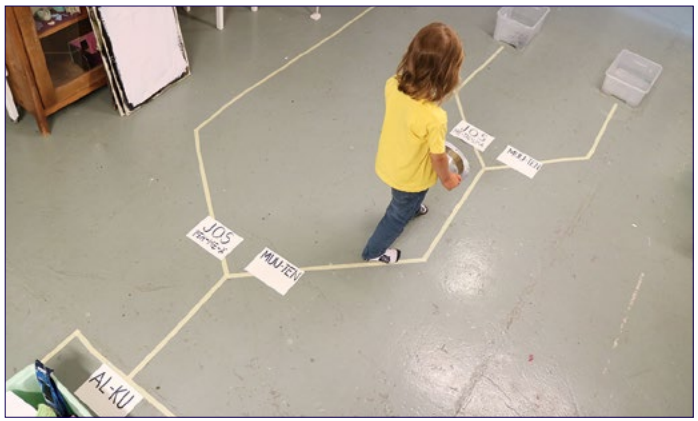
Muut toimivat tietokoneena, joka lukee komentoriviä. Komentorivi luetaan ääneen yhdessä robotin toteuttaessa aina yhden komennon kerrallaan. Leikissä on tärkeää toimia rauhallisesti, jotta komentorivi luetaan huolellisesti. Mikäli ääneenlukemisessa mennään sekaisin tai piirtorobotti ei onnistu noudattamaan komentoja, on kone käynnistettävä uudestaan, eli palattava alkuun.



Vaativuutta leikkiin voi tuoda ottamalla mukaan valmiiksi väritettyjä 6×6-mallikuvia. Tässä tehtävän versiossa tulkitaan kuva pikseli kerrallaan ja kirjoitetaan komentorivi sen rakentamisesta. Näiden komentorivien perusteella voidaan jälleen luoda uusia kuvia gridiin ja katsoa kuinka hyvin komentorivi vastaa alkuperäistä kuvaa.

Algoritmirata

Tarvikkeet:
- Erilaisia esineitä laatikossa/kassissa
- Kynä ja paperia
- Maalarinteippi

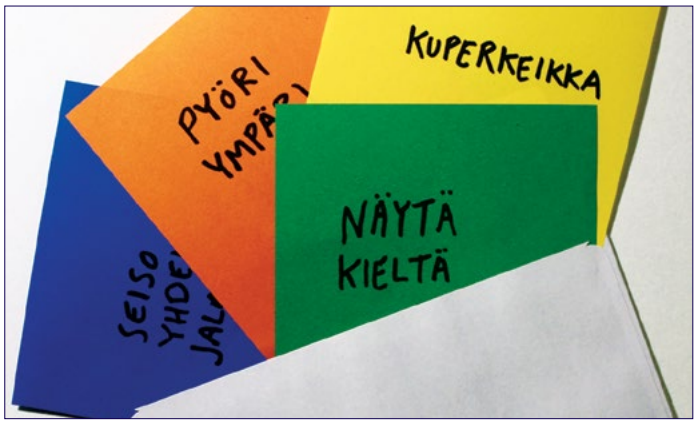


Leikin tarkoitus on harjoitella yksiselitteisten joko–tai-tulkintojen tekemistä ympäröivistä esineistä. Leikki havainnollistaa miten monimutkainen maailma voidaan jäsennellä tietokoneiden ymmärtämäksi. Leikissä esineet jaotellaan niiden ominaisuuksien mukaisesti algoritmirataa apuna käyttäen.

Valmistelu:
Teippaa maahan algoritmirata. Algoritmiradalla tarkoitetaan peräkkäin asetettuja toimintojen sarjoja, joita voi edetä aina valiten kahden vaihtoehdon välillä. Vaihtoehdot voi esittää esimerkiksi määrittelemällä vaihtoehdot seuraavasti: jos (esine on) sininen, mene vasemmalle, mutta muussa tapauksessa mene oikealle. Näin siirrytään aina uuteen tarkentavaan ja jaottelevaan kysymykseen. Näin jatketaan kunnes päästään tarpeeksi tarkkaan jaotteluun. Esimerkiksi tavoitteena voisi olla jaotella erilaisista esineistä kaikki siniset esineet erilleen, mutta siten, että metalliset lelut, pehmeät lelut ja muuta materiaalia olevat esineet ovat erillisissä laatikoissa.

Leikin kulku:
Leikissä asetutaan algoritmiradan alkupäähän, laatikon äärelle, jossa on esillä erilaisia esineitä. Vuorollaan jokainen ottaa yhden sattumanvaraisen esineen ja lähtee kulkemaan ensimmäiselle algoritmiradan risteykselle. Risteyksessä esitetään kysymys (opettaja kysyy ääneen ja auttaa tulkinnassa), joka liittyy esimerkiksi esineen muotoon, materiaaleihin tai käyttötarkoitukseen. Kysymykset kannattaa määritellä käytettävien esineiden mukaisesti. Ensimmäinen kysymys voi koskea esimerkiksi sitä, onko kyseessä lelu. Vaihtoehtoja on kaksi kappaletta: joko se on tai se ei ole. Jokainen kysymys tulee muotoilla siten, että siihen voidaan vastata vastaamalla joko–tai, jolloin toinen vastaus johtaa toisaalle ja toinen toisaalle. Mikäli esimerkissä esine olisi ollut leikkiauto, voisi seuraava kysymys koskea materiaalia, ollen esimerkiksi “Onko esine metallinen?”. Näin algoritmin avulla kaikki samanlaiset autot päätyvät samaan laatikkoon ja esineet järjestyvät halutulla tavalla. Leikin aikana voidaan kehittää uusia risteyksiä ja kysymyksiä tarkentamaan algoritmin toimintaa.

Kirjeleikki



Tarvikkeet:
- Paperia ja kyniä
- Kirjekuoria

Kirje sisältää komennon. Millaisen kirjeen sinä saat? Kirjeleikissä opitaan tulkitsemaan komentosarjoja liikkumalla.

Valmistelu:
Piirrä tai tulosta valmiita komentoja papereille, jotka mahtuvat kirjekuorien sisään. Valmistele yhden, kahden, kolmen ja neljän komennon sarjoja niputtamalla papereita peräjälkeen. Nämä niputetut komennot luovat komentosarjoja, joiden mukaan leikkijät liikkuvat. Aseta komentosarjat kirjekuoriin. Leikkiminen tapahtuu tilavassa huoneessa.

Leikin kulku:
Leikissä yksi henkilö toimii ohjelmoijana, joka lähettää kirjeitä olioille. Muut leikkijät ovat ohjelmoitavia olioita, jotka toteuttavat kirjeessä olevan komentosarjan heti kun saavat kirjeen ja avaavat sen. Ohjelmoija voi käyttää muutamaa henkilöä viestinviejinä, jotka välittävät viestit olioille. Kun oliot ovat suorittaneet liikkeen, he palauttavat komentorivin kirjekuoreen ja laskevat kirjekuoren maahan merkiksi, että kirjekuoren voi noutaa takaisin.

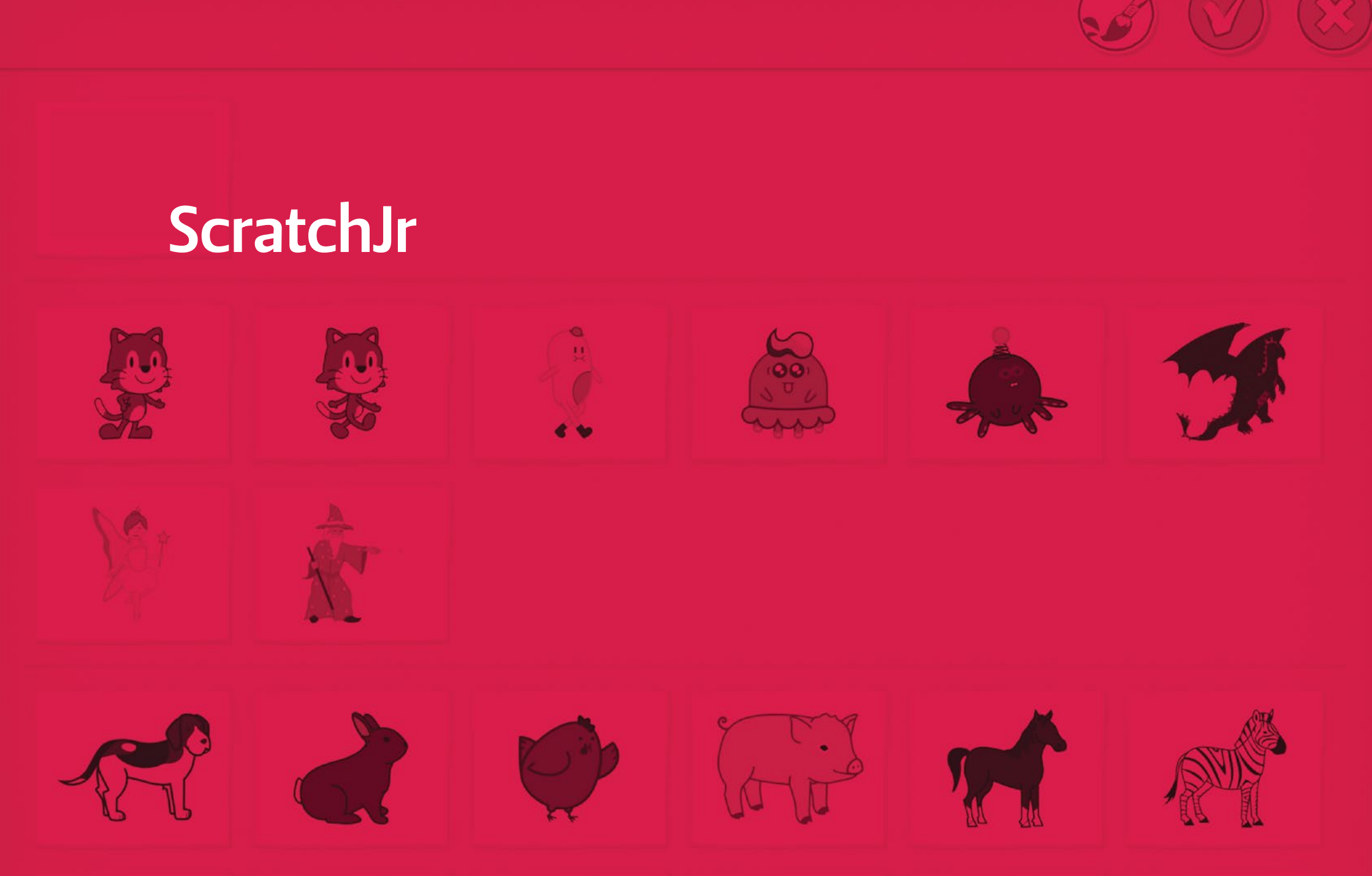
Kirjekuoret jaetaan uudestaan niin monta kertaa kuin halutaan. Leikkiä voi muokata myös siten, että hahmot aloittavat liikkumisen vasta kun ohjelmoida antaa aloitusmerkin tai jokin ennalta määritetty tapahtuma laukaisee hahmot liikkeelle. Tämä voi olla esimerkiksi pillinvihellys, jolloin kaikki aloittavat oman komentosarjansa.

Voit kehitellä lisää uusia komentoja. Komentosarjoihin voi lisätä myös kertojia lisäämällä alkuun esimerkiksi komennon “3 ×”, jolloin sama komentorivi toteutettaisiin kolme kertaa. Vai onko komentosarja päättymätön luuppi?

Voitte piirtää komennot kuvina!

Halutessasi voit käyttää leikissä apuna aiemmin luotua 6×6 pikseliruudukkoa!

ScratchJr



ScratchJr – ohjelmoinnin alkeita tabletilla

ScratchJr on graafinen ohjelmointiapplikaatio tablettitietokoneille. Ohjelma on käytettävissä täysin graafisena eli se ei vaadi lukutaitoa käyttäjältään. Ohjelmalla voi luoda vuorovaikutteisia ohjelmia esimerkiksi omien digitaalisten piirroksien pohjalta. Ohjelma sopii mainiosti ohjelmoinnin alkeiden oppimiseen, ja sen mahdollisuudet ovat laajat. Ohjelmoinnin oppiminen kannattaa aloittaa pienin askelin hauskan kokeilun kautta!

ScratchJr:n asentaminen
ScratchJr on ladattavissa iPad- ja Android-tablettitietokoneille ilmaiseksi. iPad-ohjelman voit ladata laitteen AppStoresta ja Android-laitteille Google Playstä.

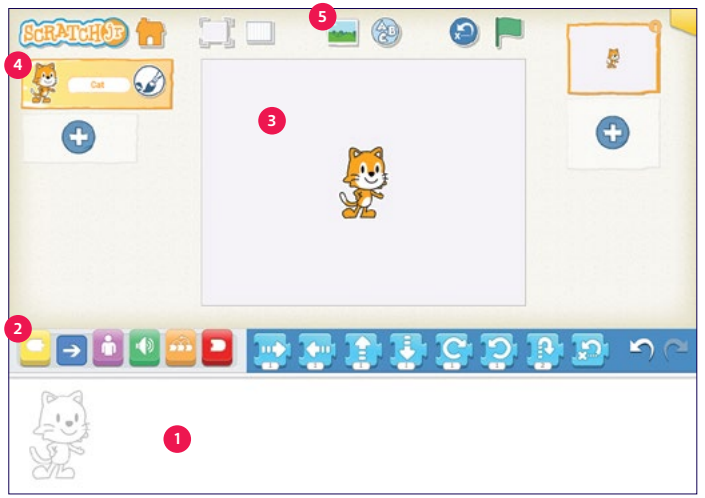
ScratchJr-projektin aloittaminen
Aloita avaamalla ScratchJr-ohjelma.

Paina **Koti-kuvaketta.** (1)

Aloita kokonaan uusi projekti painamalla **plus-kuvaketta.** (2) Tälle sivulle avautuvat myös kaikki tulevat projektisi, joihin voit palata myöhemmin.

Ruudulle avautuu ScratchJr-ohjelmointiympäristö. Ohjelmointi ScratchJr-ohjelmointiympäristössä perustuu hahmojen ohjelmoimiseen graafisilla palikoilla. Palikat asetellaan toistensa jatkeeksi kuin palapelin palat. Muotoutuvaa skriptiä eli komentosarjaa luetaan vasemmalta oikealle. Ohjelmointi tapahtuu raahaamalla komentopalikka ohjelmointialueelle sen yläpuolella olevasta skriptivalikoista. ScratchJr:ssä ohjelmoinnilla voidaan vaikuttaa hahmojen toimintaan. Jokaista hahmoa ohjelmoidaan erikseen ja voit päättää miten osat toimivat, liikkuvat tai reagoivat käyttäjään tai toisiinsa.

ScratchJr-ohjelmointiympäristössä oleellista on ymmärtää, että ohjelmoinnissa annetaan komentoja eri hahmoille. Nämä komennot kertovat miten ja milloin hahmon halutaan toimivan. Ohjelmoitaessa voidaan eri hahmot asettaa vuorovaikutukseen keskenään, jolloin syntyy jo hieman monimutkaisempia ohjelmia.



ScratchJR:n perusosat:
skriptialue (1)
skriptit ja skriptien ylävalikko (2)
ohjelmaruutu (3)
hahmovalikko (4)
taustavalikko (5)

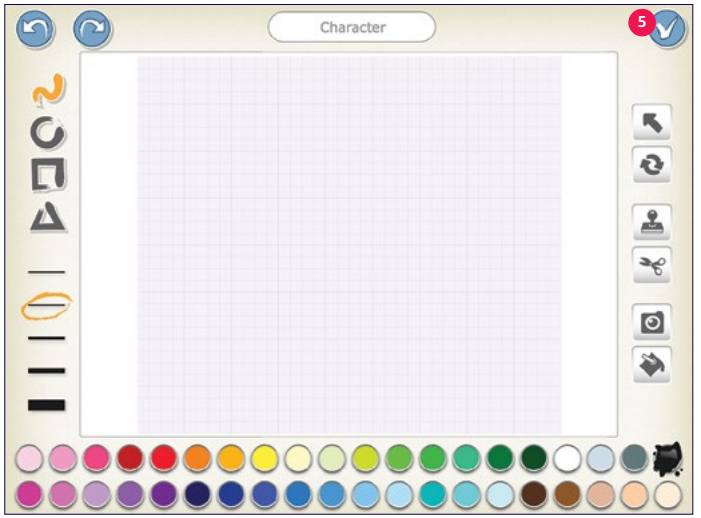
Hahmon piirtämisen työkalut:
Peru muokkaus (6)
Palauta muokkaus (7)
Kynätyökalu (8)
Ympyrätyökalu (9)
Neliötyökalu (10)
Kolmiotyökalu (11)
Viivan paksuus (12)
Värit (13)
Valmis-painike (14)
Siirto-työkalu (15)
Pyöritys-työkalu (16)
Kopiotyökalu (17)
Poistamis-työkalu (18)
Kamera-työkalu (19)
Maalikannu-työkalu (20)

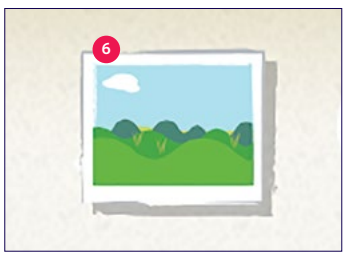
Ohje jatkuu seuraavalla aukeamalla.

Hahmojen ja taustojen luominen ScratchJr:ssä

Ennen kuin pääset luomaan oman hahmon poista ohjelmassa vakioasetuksena oleva kissa-hahmo painamalla kissahahmoa kunnes kissahahmo alkaa keinua ja **rasti-kuvake** (1) ilmestyy kissan viereen. Painamalla rastia poistat hahmon. Voit nyt piirtää oman hahmon. Paina ensin hahmovalikon **plus-kuvaketta** (2) ja hahmovalikko aukeaa.

Hahmovalikosta löydät kaikki omat hahmosi ja ScratchJr-ohjelmassa valmiina olevat hahmot. Valitse **tyhjä paperi** (3) ja paina **sivellin-kuvaketta** (4) ylälaidasta, niin pääset piirtoikkunaan jossa voit luoda oman hahmon.





Piirrä oma hahmo vapaasti kokeillen! Kun hahmosi on valmis paina **valmis-painiketta.** (5)

Voit seuraavaksi piirtää ohjelmaasi taustan. Huomioithan, että taustoja ei voi tehdä vuorovaikutteisiksi ScratchJr:ssä. Yhteen ohjelmaan on mahdollista luoda useita eri taustoja. Pääset piirtämään ensimmäisen taustakuvan painamalla **maisema-painiketta** (6) ohjelmointi-ikkunan ylälaidasta.

Taustan piirtäminen tapahtuu muuten samalla tavalla kuin hahmonkin. Valitset ensin **tyhjän taustan** ja painat **sivellin-painiketta.** Kun tausta on valmis paina **valmis-painiketta.**

Olet nyt luonut ensimmäisen hahmon ja taustan ja on aika tutustua ohjelmointiin.

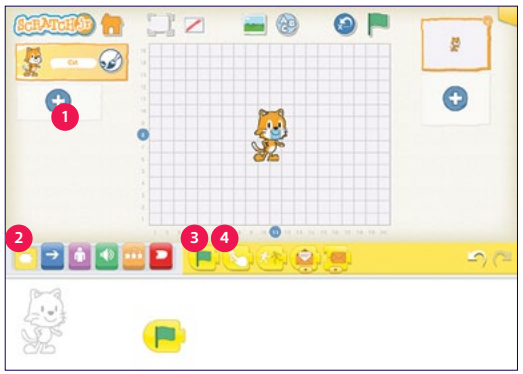
Ohjelmoinnin harjoittelun voi aloittaa hyvin pienryhmätyöskentelynä yhdelläkin tabletilla. Yhteisesti suunniteltussa, piirretyssä ja ohjelmoidussa projektissa kaikki pääsevät tekemään pienen osansa!

Piirtäkää hahmo ja tausta yhteistyönä!

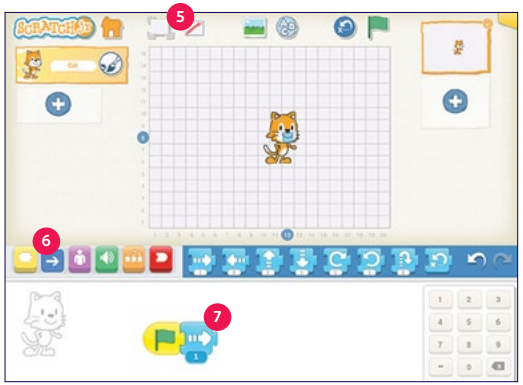
ScratchJr-projektien jakaminen on mahdollista sähköpostitse, ja ne ovat avattavissa toisilla tablettitietokoneilla joissa on ScratchJr asennettuna.

Kohti maalia

Ensimmäisenä ScratchJr-harjoituksena ohjelmoimme hahmon liikkumaan kohti maalia. Voitte käyttää aiemmin piirtämäänne hahmoa ja taustaa tehtävän pohjana. Pohtikaa mitä piirtämänne hahmo tavoittelee? Missä hahmo seikkailee? Mikä voisi olla sen maali?



Piirtäkää ensin hahmolle maali. Luokaa maali uutena hahmona painamalla ohjelmointi-ikkunassa **plus-painiketta.** (1) Asettakaa uusi hahmo ohjelmaikkunaan haluamaanne sijaintiin.

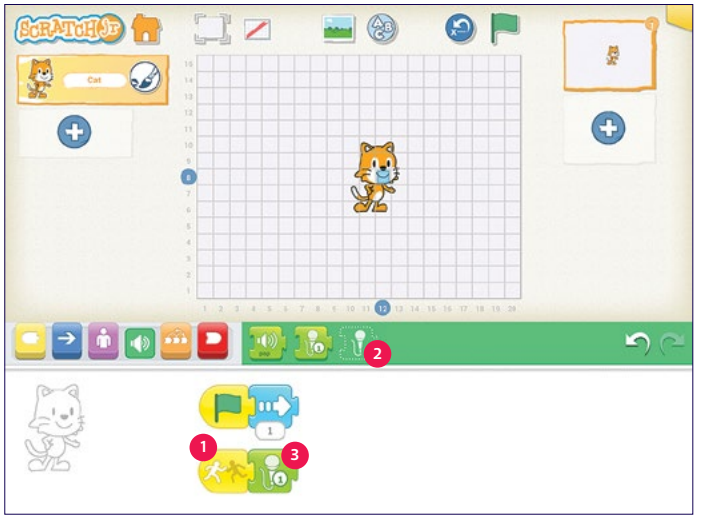


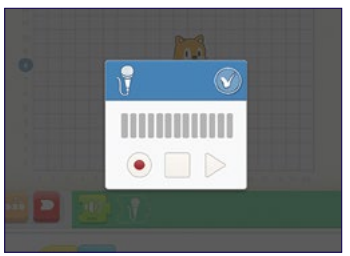
Valitkaa seuraavaksi aiemmin luotu hahmo hahmovalikosta ja asettakaa se toiselle puolelle ohjelmaikkunaa. Miettikää miten voitte ohjelmoida hahmon niin että se pääsee maaliinsa?

Aloittakaa ohjelmoiminen **keltaisesta aloitusvalikosta.** (2) Päättäkää miten haluatte, että hahmo lähtee liikkeelle kohti maalia. Toimiiko aloitusmerkkinä hahmolle se, että painetaan **vihreää lippua** (3) vai lähteekö hahmo liikkeelle, kun sitä **painetaan sormella?** (4) Raahatkaa valitsemanne komentopalikka alas ohjelmointialueelle.

Laittakaa seuraavaksi **ruudukko** (5) päälle. Ruudukon avulla pystytte laskemaan ja mittaamaan hahmon ottamia askelia.

Avatkaa seuraavaksi **sininen liike-valikko.** (6) Laskekaa seuraavaksi kuin monta askelta ja mihin suuntaan hahmon pitää liikkua ja raahatkaa tarvittavat palat keltaisen aloituspalikan perään ohjelmointalueelle. Voitte muuttaa liikepalikoiden askelmääriä painamalla nuolen alla olevia **numeroita.** (7)





Hahmoille voi rakentaa usean eri komentorivin eli skriptin, joiden mukaan hahmo toimii. Rakentakaa hahmolle kuvan mukainen skripti. Löydättekö tarvittavat komentopalikat valikoista?

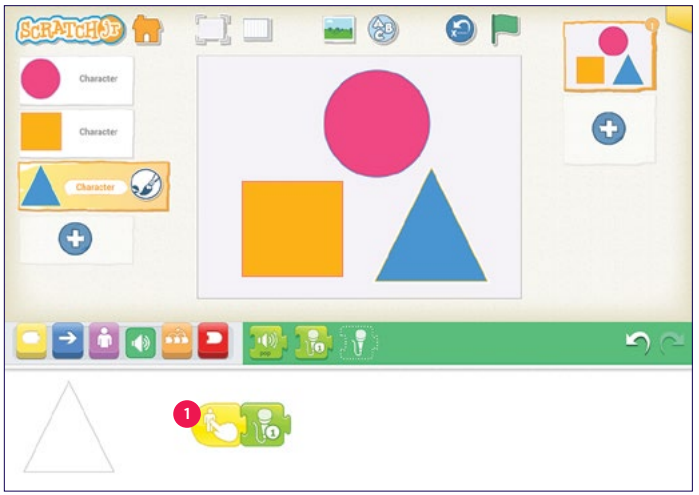
Otitte käyttöön uuden aloituspalkan, jolla hahmo aktivoituu **kun toinen hahmo kosketaa sitä.** (1) Hauska tapa osoittaa hahmojen koskettamista on äänittää hahmolle jokin ääni. Voitte äänittää oman äänen vihreästä valikosta löytyvällä **mikrofoni-painikkeella.** (2) Äänitystyökalu aukeaa automaattisesti painaessasi mikrofoni-painiketta.

Äänittämäsi ääni ilmestyy vihreään komentovalikkoon **numeroituna mikrofoni-komentona,** (3) jonka voit raahata alas ohjelmointialueelle skriptiriviin. Millainen ääni kuuluu kun hahmo pääsee maaliin?

Leikittekö pikselileikkiä aiemmin? Muistelkaa miten ruudukolla liikkuminen tapahtui leikissä!

Äänikone

Äänikone on fantasiasoitin; syntetisaattori, joka soittaa tekemiänne ääniä. ScratchJr-hahmot voivat olla muitakin kuin henkilöitä, eläimiä, kulkuneuvoja tai muita esineitä. ScratchJr:ssä kaikkien asioiden, joiden halutaan liikkuvan tai joiden toivotaan olevan muuten vuorovaikutteisia niin, että niissä on jokin toiminto tulee olla omina hahmoinaan. Näin myös soittimen painikkeet pitää luoda omina hahmoinaan.



Piirtäkää ensin soittimeen painikkeet. Harjoitelkaa neliö-, ympyrä- ja kolmiotyökalujen käyttöä piirtäessänne painikkeita. Piirtäkää yksi kolmio, yksi ympyrä ja yksi neliö.

Maalikannutyökalulla voitte värittää painikkeet helposti. Valitkaa ensin maalikannutyökalu ja haluamanne väri. Tämän jälkeen painakaa piirtämänne muodon keskelle värittömään kohtaan, niin se täyttyy automaattisesti väristä. Harjoitelkaa maalikannutyökalun käyttöä kaikkiin painikkeisiin!

Seuraavaksi on aika ohjelmoida painikkeet ja äänittää niihin äänet. Valitkaa ensin keltaisesta koodivalikosta **kosketus-aloituspalikka.** (1) Valitkaa seuraavaksi vihreä äänivalikko ja äänittäkää edellisen tehtävän mukaisesti äänet fantasiasoittimen painikkeille.

Pohtikaa millaisia äänet voivat olla? Tehkää äänet itse tai äänittäkää ne ympäristöstänne. Piirtäkää lisää painikkeita tarpeen mukaan!

Jos käytössänne on useampi tabletti, voitte rakentaa usemman soittimen ja pitää konsertin!

Elävä taulu

Testailkaa vapaasti eri ohjelmointi-palikoita. On aika tehdä elävä taulu!

Luokaa uusi hahmo ja piirtäkää vapaasti muotoja. Päättäkää miten muodot lähtevät liikkeelle ja raahatkaa perään erilaisia ohjelmointi-komentoja eri valikoista.

Luokaa lisää hahmoja. Miettikää miten ne voisivat liikkua? Katoavatko ne ja tulevat taas esille? Kuuluuko niistä ääniä?

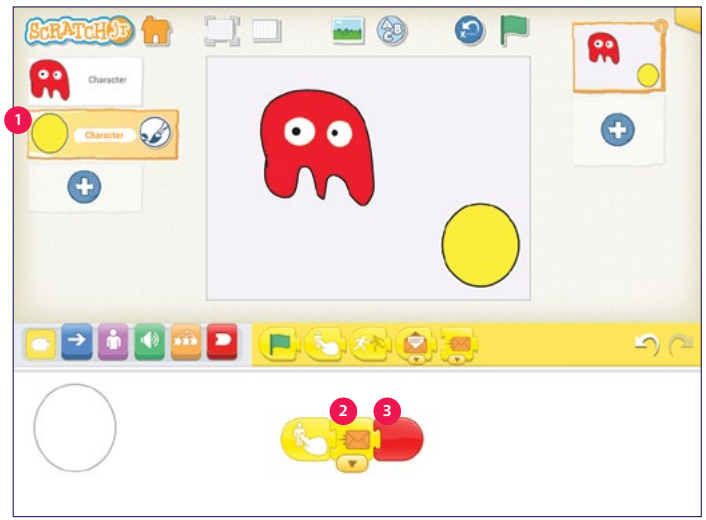
Miten hahmot lähtevät liikkeelle? Pitääkö niitä painaa vai painetaanko vihreää lippua? Ehkäpä toinen hahmo voi saada toisen liikkeelle? Lukekaa seuraavasta tehtävästä miten se tehdään.

Maalatkaa taulunne koodilla!

Ohjattava hahmo

ScratchJr:ssä on mahdollista ohjata hahmoja myös toisten hahmojen kautta. Rakennamme seuraavaksi vuorovaikutteisen elävän taulun, jonka saa liikkeelle painikkeesta.

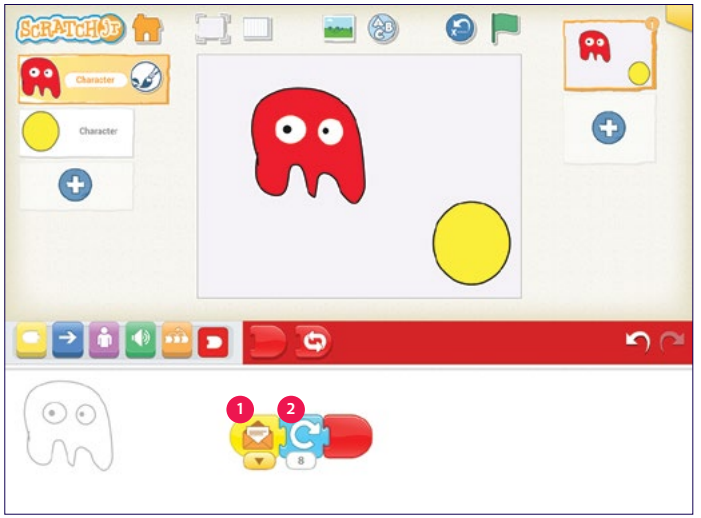
Piirtäkää ensin haluamanne kuva. Piirtäkää kuva hahmona, jotta voitte ohjelmoida siihen toimintoja. Piirtäkää seuraavaksi ohjainpainike. Voitte piirtää haluamanne painikkeen. Voitte esimerkiksi hyödyntää ympyrä-, neliö- tai kolmiotyökaluja.



Seuraavaksi molemmille piirretyille hahmoille ohjelmoidaan toiminnot. Aloittakaa painikkeesta valitsemalla se hahmovalikosta, jolloin sen ympärillä on **keltainen reunus.** (1)

Valitkaa keltaisesta valikosta ensin kosketus-komentopalikka ja raahatkaa se ohjelmointikenttään. Valitkaa samasta valikosta seuraavaksi **kirjeen lähetys-palikka** (2) ja liittäkää se skriptijonoon seuraavaksi. Tämän jälkeen voitte sulkea skriptin punaisesta valikosta löytyvällä punaisella **lopetus-palikalla.** (3)

Muistateko kirjeleikin? Saitko sinä millaisen komentokirjeen?

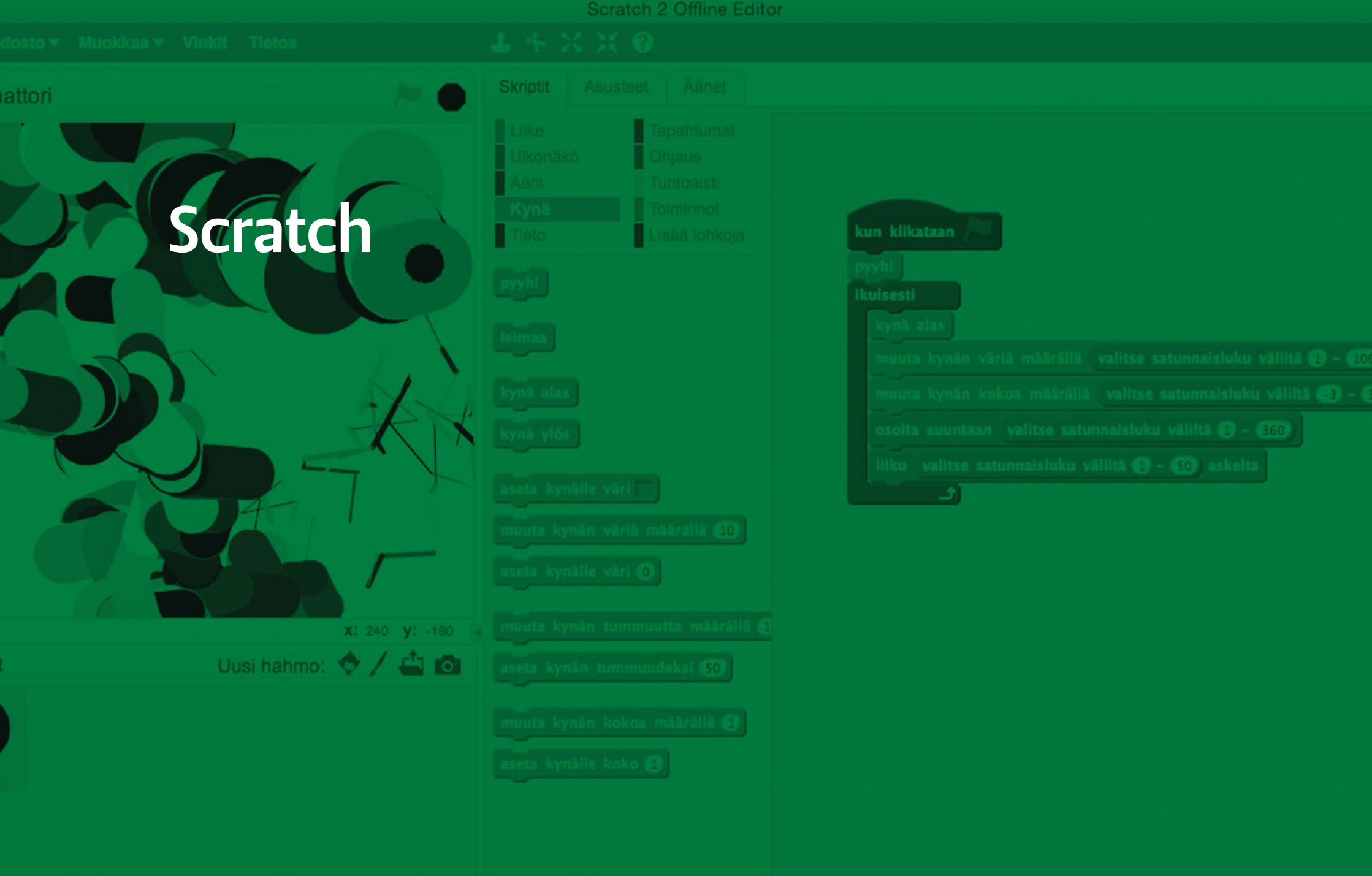


Ohjelmoidaan seuraavaksi piirtämänne kuva. Valitkaa kuva ensin hahmovalikosta. Tämän jälkeen raahataan keltaisesta ohjelmointivallikosta **saa kirjeen -palikka** (1) ohjelmointikenttään. Raahatkaa samaan skriptiin sinisestä valikosta **pyörin oikealle -palikka.** (2) Painakaa pyörimis-palikan alla olevaa numeroa, ja voitte määrittää kuinka monta askelta kuva pyörii. Päättäkää skripti punaisesta valikosta löytyvällä lopetus-palikalla. Kuvanne liikkuu nyt kun painiketta painetaan!

Piirtäkää lisää painikkeita ja ohjelmoikaa niistä jokainen lähettämään eri värinen kirje. Ohjelmoikaa hahmolle uudet skriptit jokaiselle eri väriselle saadulle kirjeelle. Miten hahmonne liikkuu, kun se saa kunkin kirjeen?

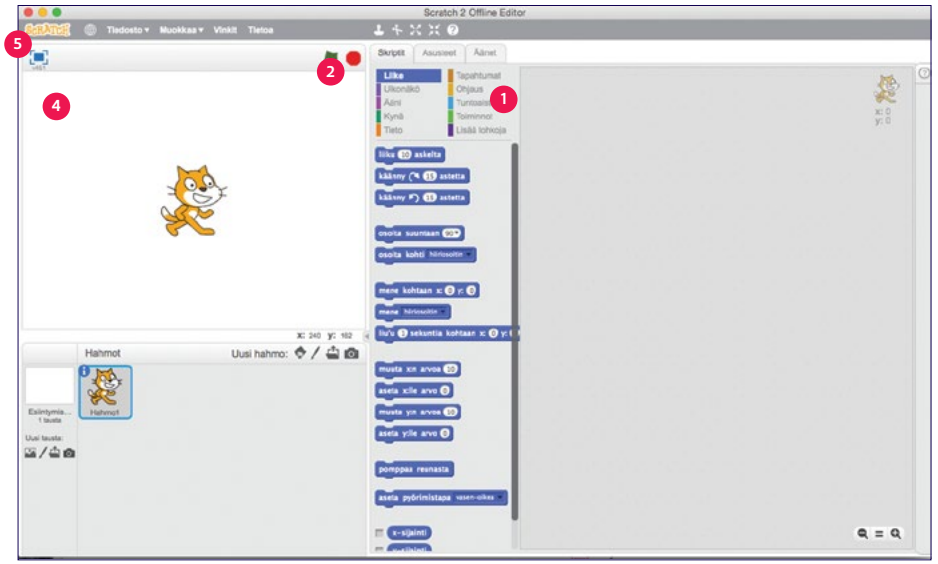
Entä olisiko kuvassa useampia hahmoja jotka reagoivat painikkeisiin? Tai entä jos hahmot reagoisivatkin toisiinsa?

Osaisitteko tehdä peliohjaimen?


Scratch 2 Offline Editor
Muokkaa
Vinkit
Tietoa
Skriptit
Asusteet
Äänet
Liike
Ulkonäkö
Ääni
Kynä
Tieto
Tapahtumat
Ohjaus
Tuntoaisti
Toiminnot
Lisää lohkoja
pyyhi
leimaa
kynä alas
kynä ylös
aseta kynälle väri
muuta kynän väriä määrällä 10
aseta kynälle väri 0
muuta kynän tummuutta määrällä
aseta kynän tummuudeksi 50
muuta kynän kokoa määrällä 1
aseta kynälle koko 1
x: 240 y: -180
Uusi hahmo:
Scratch
kun klikataan
pyyhi
ikuisesti
kynä alas
muuta kynän väriä määrällä valitse satunnaisluku väliltä 1 -
muuta kynän kokoa määrällä valitse satunnaisluku väliltä -3 -
osoita suuntaan valitse satunnaisluku väliltä 1 - 360
liiku valitse satunnaisluku väliltä 1 - 50 askelta

Scratch – Graafista ohjelmointia tietokoneella

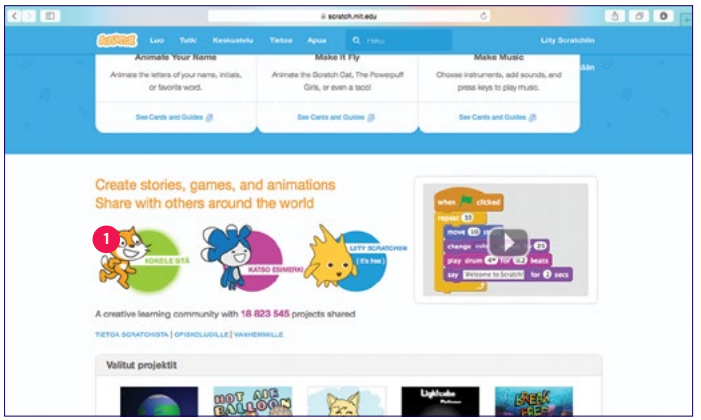
Scratch on graafinen ohjelmointympäristö tietokoneille. Ohjelman käytäminen vaatii kuitenkin myös lukutaitoa käyttäjältään. Kuitenkin komentojen, eli skriptien luominen ja seuraaminen on helposti hahmotettavissa ScratchJr ohjelmointiapplikaation pohjalta. Ohjelmalla voi luoda vuorovaikutteisia ja interaktiivisia ohjelmia. Keskeistä ohjelmalle on se, että siinä ohjelmoidaan eri hahmoja. Hahmot voidaan laittaa vuorovaikutukseen keskenään, jolloin rakentuu kokonainen ohjelma. Scratch-ohjelmointiympäristö sopii mainiosti ohjelmoinnin alkeiden oppimiseen, ja sen mahdollisuudet ovat laajat. Ohjelmoinnin oppiminen kannattaa aloittaa pienin askelin hauskan kokeilun kautta!



Scratchin aloittaminen

Scratch on ohjelmointiympäristö, jonka avulla voit luoda omia ohjelmia graafisen ohjelmointikielen avulla. Scratch on huomattavasti monipuolisempi versio ScratchJr:stä. Molemmissa ohjelmointiympäristöissä on **koodialue**, (1) johon rakennetaan hahmon koodi-skriptejä raahaamalla. **Skriptivalikko** (2) pitää sisällään kaikki käytettävissä olevat koodin rakennusosat. **Hahmo- ja taustavalikko** (3) mahdollistaa uusien hahmojen sekä taustojen luomisen. **Ohjelmaruutu** (4) näyttää rakentamasi ohjelman, ja sen saa koko näytön kokoiseksi vasemmassa yläkulmassa olevasta **komentopainikkeesta.** (5)

Molemmat ohjelmat toimivat samalla logiikalla, jossa ohjelmoija luo jokaiselle hahmolle omat koodit raahaamalla hiirellä halutun komennon skriptivalikosta koodialueelle. Scratchissä ohjelmointikäskyt eli skriptit rakentuvat ensisijaisesti päällekkäin luettavaksi ylhäältä alas.



Voit tehdä Scratchillä monia asioita, joita ScratchJr:llä et voi toteuttaa. Voit esimerkiksi käyttää tietokoneesi web-kameraa rakentamasi ohjelman osana.

Voit ladata ja asentaa ohjelman koneellesi osoitteesta *scratch.mit.edu/scratch2download.*

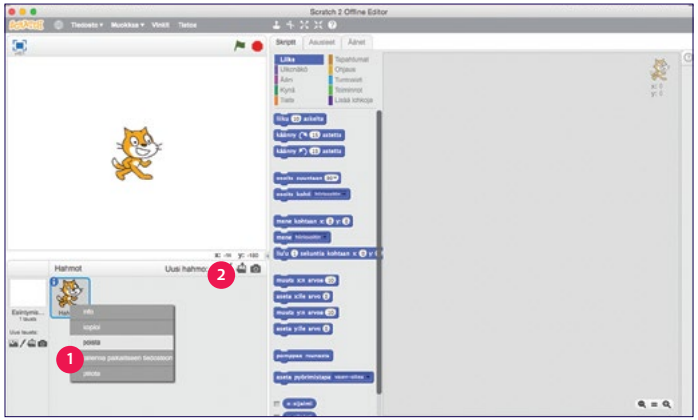
Helpoin tapa käyttää Scratch-ohjelmaa on käyttää sitä internetissä selainversiona osoitteessa *scratch.mit.edu.* Käynnistääksesi ohjelman paina sivustolla olevaa kuvaketta **"Try it out".** (1) Ohjelman voi vaihtaa suomenkieliseksi ylävalikon vasemmassa laidassa olevasta maapallokuvakkeesta. Esimerkeissä käytämme suomenkielistä versiota.

Scratchin selainpohjainen versio ei eroa juurikaan koneelle asennettavasta versiosta.

Scratchin selainpohjainen versio sallii sinun liittyä osaksi Scratch-yhteisöä, jolloin voit jakaa omia projektejasi Scratchin kotisivujen kautta.

Voit tutustua eri käyttäjien Scratch-projekteihin Scratchin kotisivulla. Paina vain auki haluamasi projekti, ja painamalla vihreää lippua ohjelma käynnistyy. Voit halutessasi myös katsoa mistä osista projekti koostuu, ja muokata siitä itsellesi sopivan. Paina auki olevassa projektissa oikeassa yläkulmassa olevaa Katso sisälle -painiketta. Voit halutessasi muokata ja jatkaa toisten ohjelmia, ja käyttää niitä omien omien projektiesi inspiraationa ja raaka-aineena. Ohjelmakoodien jakaminen on yleinen ja tärkeä keino edistää ohjelmoinnin kehittymistä. Älä siis turhaan omi ohjelmaasi vaan jaa se myös muiden muokattavaksi. Vastavuoroisesti opit paljon toisilta.

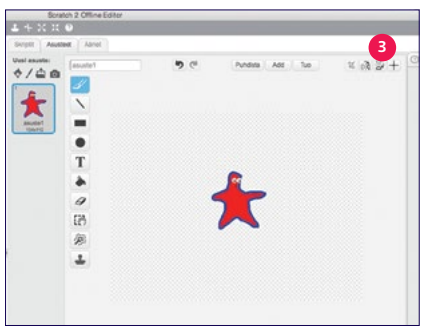
Uuden hahmon luominen



Ensimmäisenä tehtävän on hyvä tutustua yhden hahmon liikuttamiseen. Luo kokonaan uusi projekti ja **poista oletushahmona oleva kissahahmo painamalla hahmon päällä hiiren kakkospainiketta ja valitsemalla Poista.** (1) Luo uusi hahmo painamalla Uusi hahmo -valikosta **sivellin-kuvake.** (2)

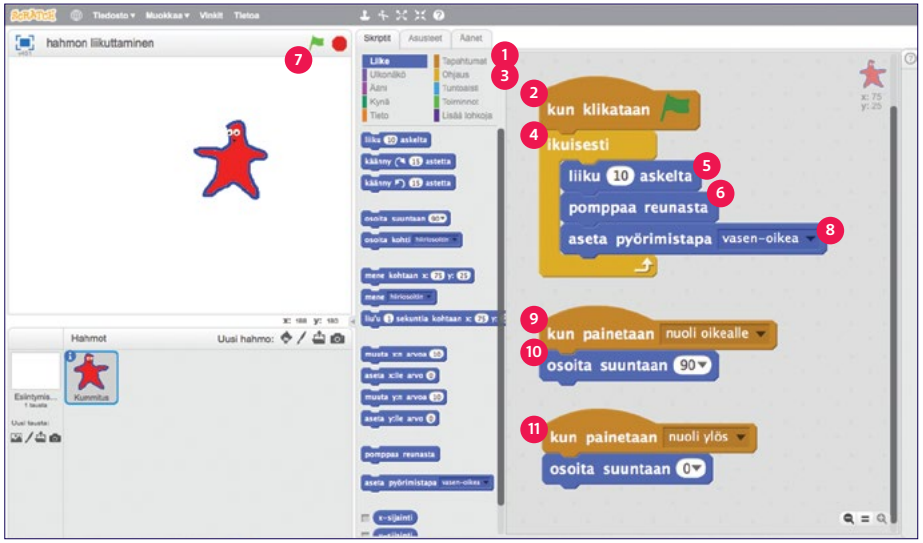
Aloita piirtämällä uusi hahmo. Tutustu ensin rauhassa eri piirtotyökaluihin. Piirtotyökalut ovat kynä/sivellin, viiva, neliö, ympyrä, tekstikenttä, maalikannu, pyyhekumi, rajaustyökalu, taustan putsaus ja kloonaustyökalu. Voit valita eri värejä piirtotyökaluille. Voit myös tuoda esimerkiksi valokuvan ohjelman hahmoksi valitsemalla Tuo-komennosta koneellesi tallennettu kuva. Muista, että voit zoomata näkymää zoomaustyökalulla.

Piirrettyäsi hahmon muista keskittää se oikein. Tämä on tärkeä työvaihe, jotta hahmo liikkuu ympäristössä asianmukaisesti ilman virheitä. **Keskittämällä valitset hahmollesi keskipisteen.** (3)



Nimeä uusi hahmo valitsemalla Hahmot-valikosta haluttu hahmo ja painamalla hiiren kakkospainiketta ja valitsemalla “info”. Näin pääset vaihtamaan Hahmo1-nimen tilalle haluamasi nimeni.

Hahmon liikuttaminen



Aloita hahmon liikuttaminen yksinkertaisella koodilla. Valitse Skriptit-välilehdestä **Tapahtumat-valikosta** (1) aloituskomennoksi **”kun klikataan (vihreä lippu)”.** (2) Seuraavaksi valitse **Ohjaus-valikosta** (3) tapahtuman kestoksi **”ikuisesti”** (4): Raahaa tämän komennon sisään Liike-valikosta tapahtumaksi **”liiku 10 askelta”** (5) sekä **”pomppaa reunasta”.** (6) Koodin rakenne mukailee aina tämän esimerkin rakennetta. Erilaisten ohjauskomentojen ja niiden ehtojen alle voidaan määritellä erilaisia toimintoja, jotka tapahtuvat siinä järjestyksessä kuin ne ovat koodissa.

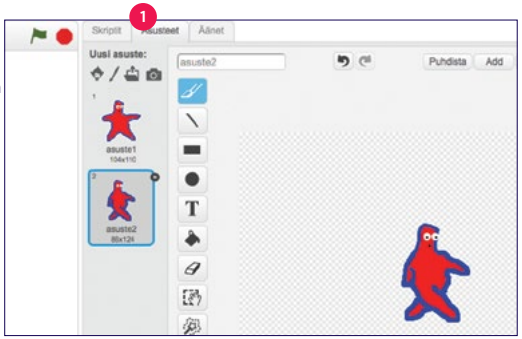
Kokeile hahmon liikuttamista. Ohjelma alkaa, kun painat **vihreää lippua** (7) ohjelmaruudun yläpuolella. Pystyt lopettamaan ohjelman aina painamalla punaista ympyrää.

Voit muokata hahmon liikettä hitaammaksi tai nopeammaksi määrittelemällä askelille eri arvon.

Jotta saat hahmon kääntymään vaakasuunnassa, lisää koodiin ”pomppaa reunasta”-komennon jälkeen Liike-välilehdestä **”aseta pyörimistapa [vasen-oikea]”** (8). Nyt hahmosi pyörii aina seinästä pompattuaan vain vaakasuunnassa.

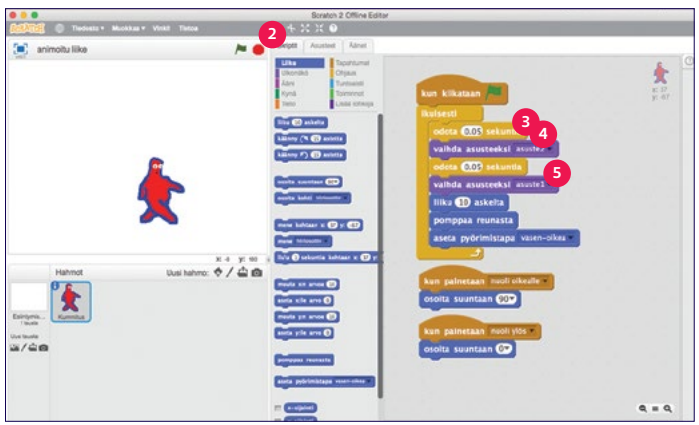
Lisätään liikkeeseen vielä mahdollisuus vaihtaa suuntaa. Valitse Tapahtumat-välilehdestä **”kun painetaan [nuoli oikealle]”** (9) -komento ja raahaa se koodialueelle. Aseta komennoksi Liike-välilehdestä **“osoita suuntaan (90) oikea”.** (10) Lisää samalla tavalla myös komento **“kun painetaan [nuoli ylös] osoita suuntaan (0) ylös”.** (11)

Animoitu liike



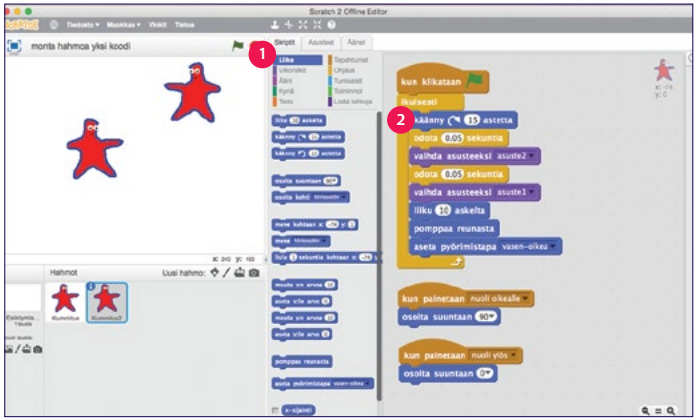
Tässä tehtävässä tutustutaan hahmon animoimiseen käyttäen pohjana edellä luotua liikkuvaa hahmoa.

Animaatio luo illuusion liikkuvasta hahmosta. Illuusio syntyy, kun kuva vaihtuu sopivalla tahdilla kahden tai useamman kuvan välillä. Aloita animointi siis **Asusteet-välileh-destä** (1) ja klikkaa Asuste1-hahmoa hiiren kakkospainikkeella sekä valitse "kopioi". Valitse Asuste2 aktiiviseksi ja muokkaa piirtoeditorilla hahmolle animaation toinen asento



Palaa **Skriptit-välilehteen.** (2) Valitse Ohjaus-välilehdestä ehtolause **"odota (1) sekuntia"** (3) ja raahaa se jo olemassa olevan, liikkumista määrittelevän koodin alkuun. Määrittele arvoksi yhden sekunnin sijaan 0.05 sekuntia. Valitse **Ulkonäkö-valikosta "vaihda asusteeksi [asuste2]"** (4) ja raahaa se tapahtumaksi "odota (0.05) sekuntia" -komennon jälkeen. Lisää tämän jälkeen tapahtumaksi jälleen "odota (0.05) sekuntia" ja Ulkonäkö-välilehdestä jälleen **"vaihda asusteeksi [asuste1]".** (5)

Monta hahmoa

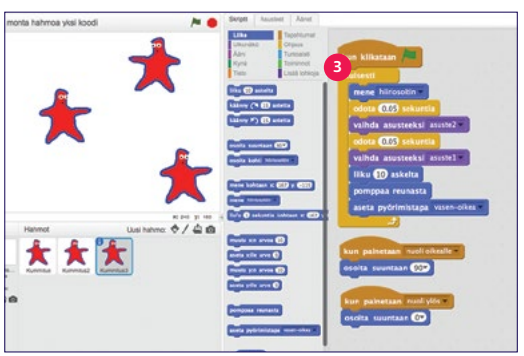


Tässä tehtävässä luodaan ohjelma, jossa ohjelmaruudussa on samanaikaisesti monta hahmoa, mutta jokaisella niistä on oma skripti. Jokainen hahmo liikkuu siis omalla tavallaan samassa ohjelmassa.

Pääset vauhdilla liikkeelle jatkamalla tehtävää suoraan Animoitu liike -tehtävän pohjalta.

Hahmo 2

Valitse Hahmot-valikosta hahmo klikkaamalla hiiren kakkospainiketta ja valitse "kopioi". Mene uuden hahmon Skriptit-välilehteen, valitse **Liike-välilehdestä** (1) esimerkiksi **"käänny (15) astetta",** (2) ja raahaa se hahmon liikettä määrittävän skriptin alkuun. Voit poistaa samalla ylimääräiset komennot "pomppaa reunasta" sekä "aseta pyörimistapa [vasen-oikea]".

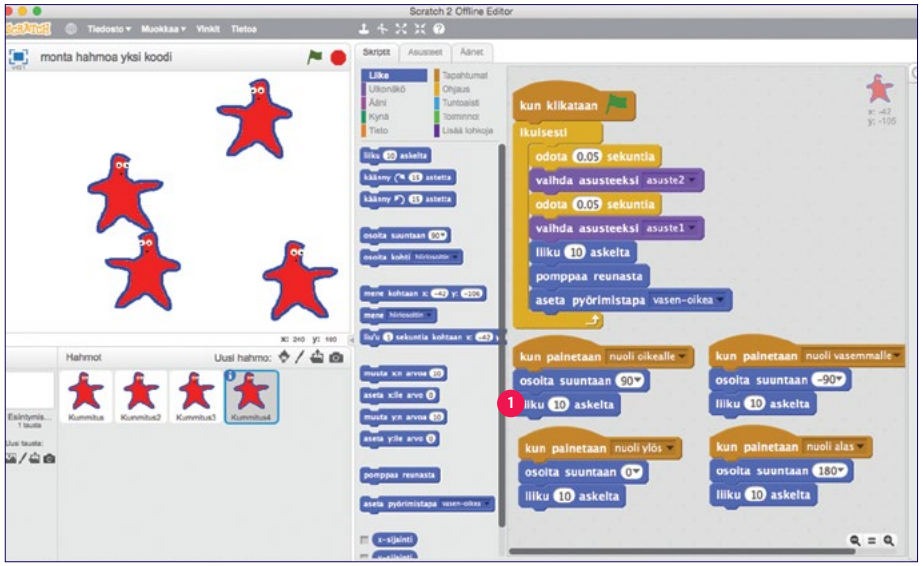


Nyt toinen hahmo juoksee ympyrää.

Hahmo 3

Kopioi jälleen ensimmäinen hahmo. Tee samat muutokset kuin Hahmo 2:n kohdalla, mutta määritä Liike-väli-lehdestä komennoksi **"mene [hiiriosoitin]"** (3) ja aseta se skriptin alkuun. Tämä hahmo seuraa siis hiirtä.

Ohje jatkuu seuraavalla aukeamalla.

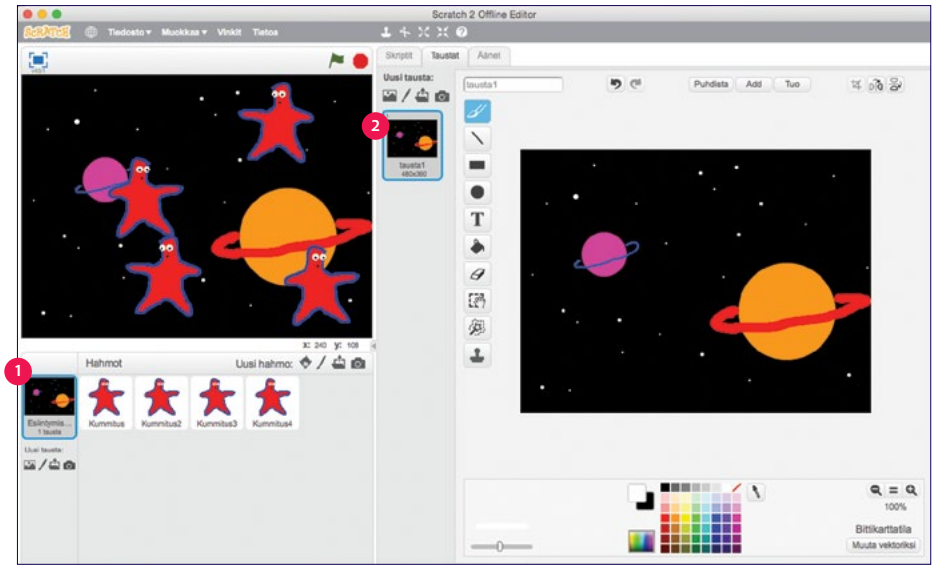


Hahmo 4
Kopioi jälleen ensimmäinen hahmo. Muokkaa skriptiä siten, että lisäät jo olemassa olevien "kun painetaan [nuoli oikealle]" > "osoita suuntaan (90) oikea" -komentojen jatkoksi **"liiku (10) askelta"** (1) -komennon. Kopioi tämä skripti neljä kertaa ja vaihda jokaiseen uudet suunnat: "kun painetaan (nuoli vasemmalle)" > "osoita suuntaan (-90)", "kun painetaan (nuoli ylös)" > "osoita suuntaan (0)", "kun painetaan (nuoli alas)" > "osoita suuntaan (180)".

Tämä hahmo noudattaa näppäimistöstä annettuja suuntakomentoja.

Animoitua liikettä tehdessä kokeilkaa eri aika-arvoja, niin saatte liikkeen juuri sopivaksi omalle hahmollenne!

Taustan luominen



Valitse **"Esiintymistausta"** (1) aktiiviseksi, josta taustat-valikosta **"tausta1".** (2) Piirtoeditorilla voit piirtää halutun ympäristön.

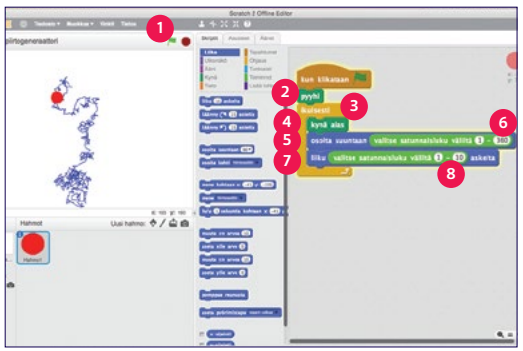
Muistattehan että hahmojen tai taustojen ei tarvitse olla esittäviä!

Piirustusgeneraattori

Seuraavassa tehtävässä luodaan automaattinen piirustusgeneraattori, joka toimii sattumanvaraisesti luodun koodin varassa.

Aloita ohjelman luominen luomalla haluamasi näköinen piirrin. Piirtimenä voi toimia myös aiemmin luomasi hahmo. Esimerkissä piirtimenä toimii punainen ympyrä.

Aloita koodin rakentaminen jälleen valitsemalla Toiminnot-välilehdestä **“kun klikataan (vihreä lippu)”.** (1) Tämän jälkeen koodiin määritellään komennoksi **Kynä-valikosta “pyyhi”.**

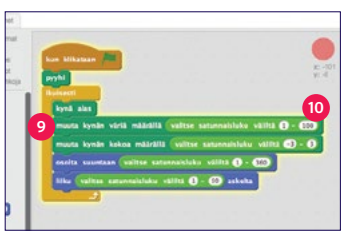


(2) Nyt joka kerta kun vihreää lippua painetaan, puhdistuu myös edellinen piirustus pois ohjelmaruudusta.

Jatka koodia valitsemalla Tapahtumat-valikosta **“ikuisesti”** (3) ja raahaa se koodin rakenteeseen. Tämän sisään määritellään tapahtumaksi **“kynä alas”.** (4) Seuraavaksi komennoksi määrittele Liike-valikosta **“osoita suuntaan”,** (5) ja sen arvoksi määrittele Toiminnot-valikosta **“valitse satunnaisluku välillä ( ) - ( )”.** (6) Valitse arvoksi 1 - 360. Voit kokeilla myös muita arvoja. Kun kynälle on määritelty kulma, eli suunta, johon se sattumalta kääntyy, määritellään kuinka paljon kynä liikkuu jokaisen kierroksen aikana. Määrittele liike-valikosta **”liiku ( ) askelta”.** (7) Määritä tähänkin sattumanvarainen arvo välillä 1-10 valitsemalla Toiminnot-valikosta **“valitse satunnaisluku välillä ( ) - ( )”** (8) ja anna kenttiin arvoiksi 1 ja 10. Yksi kierros koostuu ”ikuisesti”-komennon sisällä olevasta rakenteesta, jota ohjelma lukee alusta loppuun.

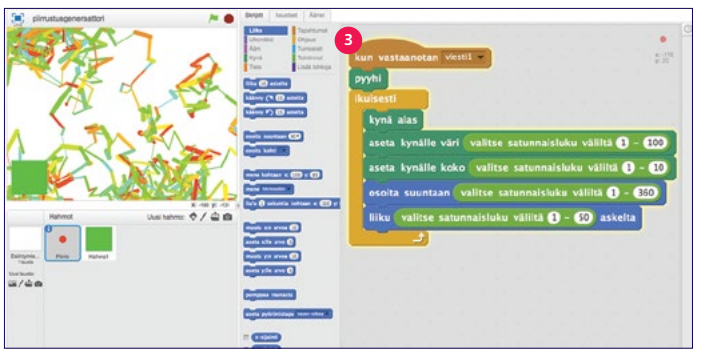
Paina **vihreää lippua** käynnistääksesi piirtogeneraattorin.

Tarkastele muodostuvaa kuvaa koko näytön kokoisena. Mitä kuvasta voi hahmottaa?



Lisää seuraavaksi piirtimeen uusia värejä ja eri kokoisia siveltimiä. Valitse Kynä-välilehdestä **“muuta kynän väriä määrällä ( )”,** (9) ja määritä sille satunnainen arvo Toiminnot-valikosta **“valitse satunnaisluku välillä ( ) - ( )”.** (10) Tee samanlainen muuttuja koskemaan myös kynän kokoa. Kokeile lisätä koodiin myös muita uusia komentoja, kuten eri Liike-valikon komentoja.

Hahmojen keskinäinen kommunikointi



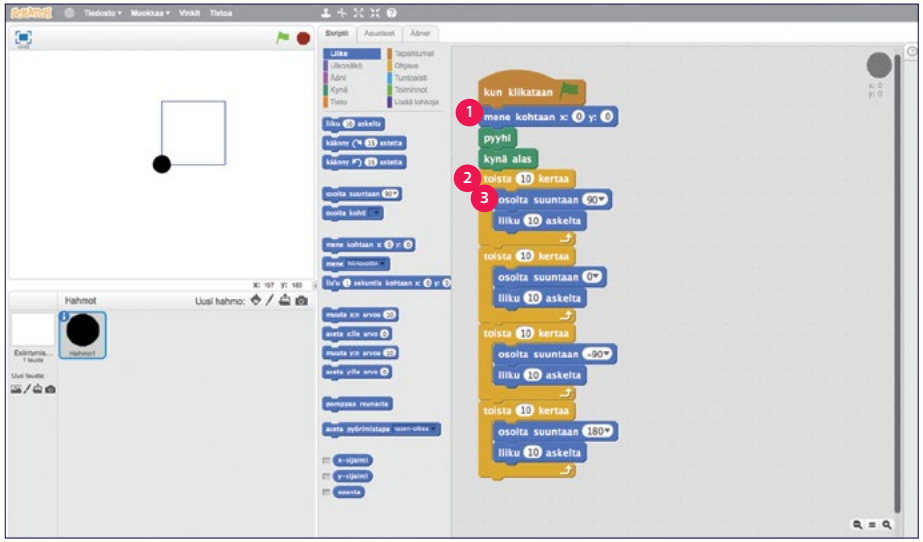
Tässä tehtävässä piirtogeneraattorille luodaan käynnistyspainike, jotta piirtogeneraattorin voi käynnistää ilman vihreää lippua omasta painikkeesta. Jatka siis tätä tehtävää piirtogeneraattorin pohjalta.

Aloita luomalla uusi hahmo **”Uusi hahmo” -valikosta.** (1) Piirrä hahmosta esimerkiksi vihreä neliö, jolloin se kuvastaa selkeästi, että painiketta saa klikata. Aloita hahmon koodi valitsemalla Tapahtumat-valikosta “kun tätä hahmoa klikataan” -komentokuvake**.** Komennon jälkeen tapahtuvaksi toiminnoksi valitaan Tapahtumat-valikosta **“lähetä [viesti1]”** (2) -komentokuvake.

Muokkaa seuraavaksi Piirrin-hahmon koodia. Vaihda koodin aloituskomennoksi **“kun vastaanotan [viesti1]”.** (3)

Vihreää laatikkoa klikkaamalla piirtogeneraattori käynnistyy.

Hahmojen liikuttaminen tarkasti



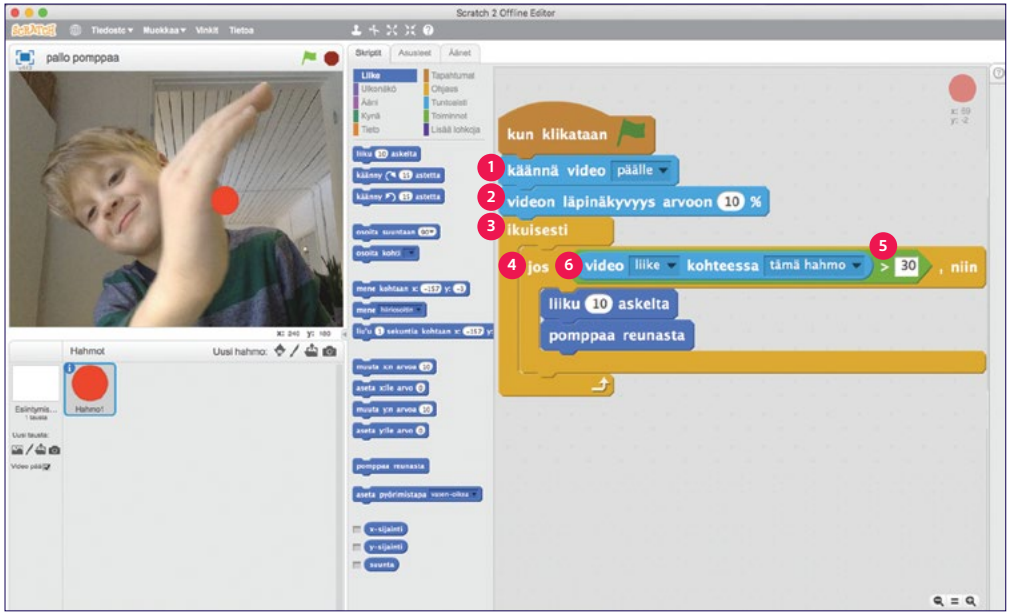
Hahmojen tarkkaan liikuttamiseen kannattaa tutustua kynää käyttämällä. Tällöin pystyt helposti hahmottamaan, miten koodisi hahmoa liikuttaa. Samalla voit asettaa tavoitteeksi piirtää tietyn muotoisia kuvioita.

Voit käyttää tämänkin tehtävän pohjana piirustusgeneraattoria. Muokkaa koodi alkamaan "kun klikataan (vihreä lippu)" -komennosta. Määritä Liike-valikosta **"mene kohtaan x:0 y:0".** (1) Näin piirustus alkaa ohjelmaikkunan keskeltä. Seuraavaksi määritellään, että koko ohjelmaikkuna pyyhitään sekä asetetaan kynä alas. Tämän jälkeen määritellään liikkeet. Valitse Ohjaus-valikosta **"toista (10) kertaa"** (2) -komento ja liitä se skriptin jatkeeksi. Tämän sisään määritellään tapahtumiksi Liike-valikosta **"osoita suuntaan (90 oikea)"** (3) -komento. Määritä tähän suuntaan tapahtuvaksi liikkeeksi 10 askelta.

Jatka skriptiä samalla tavalla, mutta vaihda seuraavan skriptin alkuun "osoita suuntaan (0 ylös)" -komento. Jatka näin muodostaaksesi neliön.

Kokeile määritellä liikkeiden kestoiksi kymmenen kerran sijaan esimerkiksi yksi sekunti. Samaan lopputulokseen voi päästä useilla erilaisilla koodeilla! Kokeile myös luoda erilaisia muotoja.

Lisätty todellisuus ja pallomeri

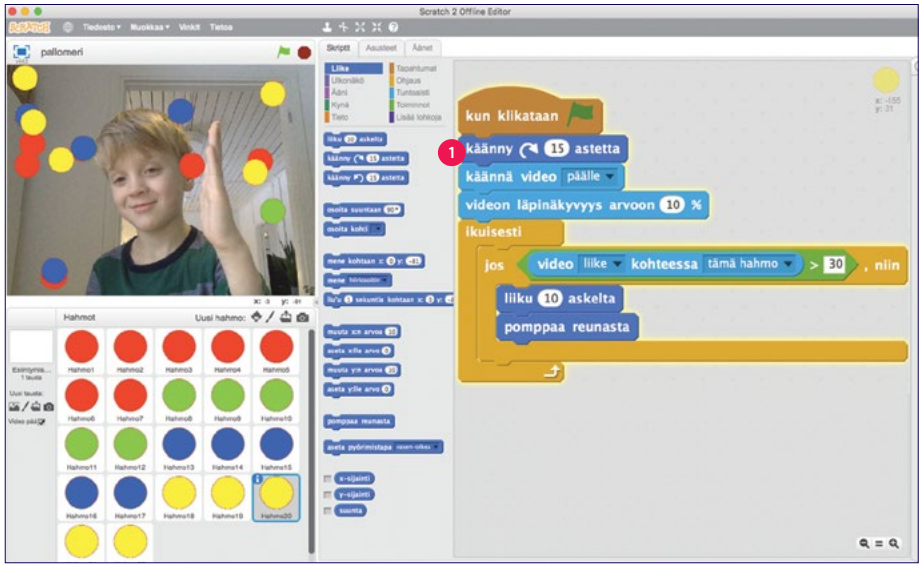


Tässä tehtävässä tutustutaan siihen, kuinka voit ohjelmoida Scratchillä ohjelman, jossa voit omalla kehollasi osallistua digitaaliseen maailmaan tai jossa digitaalinen maailma tulee lisänä meidän ihmisten maailmaan. Tällöin voidaan puhua esimerkiksi lisätystä todellisuudesta, augmented realitystä.

Aloita luomalla yksi värillinen ympyrä. Määritä ohjelma alkamaan vihreää lippua painamalla. Seuraavaksi komennoksi määritä Tuntoaisti-valikosta **"käännä video [päälle]"** (1) -komento. Määritä samasta valikosta **"videon läpinäkyvyys arvoon (10)%".** (2)

Aseta seuraavan komennon ehtolauseeksi **"ikuisesti"** (3) ja tämän sisään **"jos [ ] , niin"** (4) -komento. Tämä komento pyytää täyttämään ehdon. Haluamme määritellä, että mikäli hahmon kohdalla on liikettä enemmän kuin määritellyn raja-arvon verran, tapahtuu hahmossa määritelty asia eli liikkuminen. Valitse siis Toiminnot-valikosta ja sielä **"[ ] > [ ]"** (5) -komento. Raahaa tämä edellisen ehtorakenteeseen. Valitse Tuntoaisti-valikosta **"video [liike] kohteessa [tämä hahmo]"** (6) -komento ja raahaa se ehtorakenteen ensimmäiseen kenttään. Määrittele seuraavaksi arvo liikkeelle, jota video lukee. Voit tulkita arvoja akti-voimalla ohjelmaruutuun kentän, joka osoittaa liikkeen määrää, ruksittamalla "video [liike] kohteessa [tämä hahmo]" -komennon Tuntoaisti-valikosta. Samasta paikasta saat sen myös lopuksi pois päältä. Sopiva arvo liikken tunnistamiselle on esimerkiksi 30.

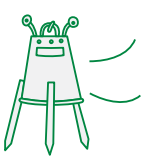
Ohje jatkuu seuraavalla aukeamalla.



Lopuksi määritellään mitä tapahtuu, jos hahmon kohdalla on liikettä. Valitse tapahtumaksi "liiku (10) askelta" sekä tämän jatkoksi "pomppaa reunasta".

Muokataan vielä lopuksi koodia siten, että saamme ympyrän liikkumaan monipuolisemmin. Määritellään koodin alkuun, että jokaisella kerralla kun vihreää lippua painetaan, lähtee ympyrä aina hieman eri suuntaan liikkeelle. Käytä tähän **"käänny (15) astetta"** (1) -komentoa.

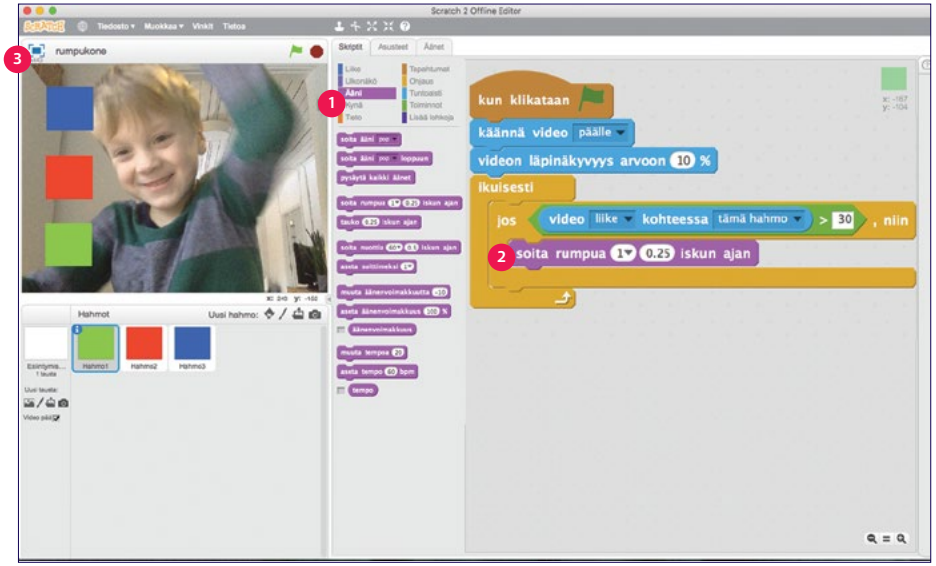
Voit luoda myös helposti interaktiivisen AR-pallomeren kopioimalla yksittäisiä palloja niin monta kappaletta kuin jaksat.



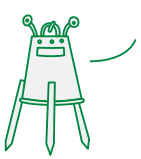
Kokeilkaa interaktiivista pallomerta koko näytön kokoisena ja mikäli mahdollista, heijastakaa se videotykillä seinään.

Mitä muita hahmoja ja toimintoja voisit hallita tällä tavoin?

Rumpukone

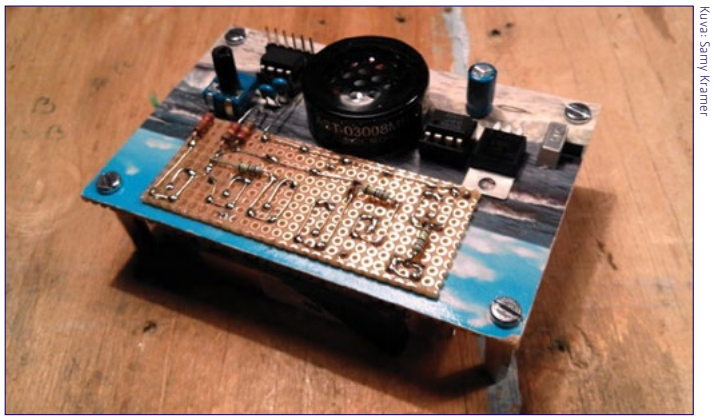


Pystytte luomaan interaktiivisen rumpukoneen muokkaamalla ensimmäistä pallotehtävää. Vaihtakaa liikkumisen sijaan tapahtumaksi **"Ääni-valikosta"** (1) haluttu ääni. Esimerkissä käytetään **Scratchin valmiita rumpuääniä.** (2) Voitte luoda uusia elementtejä joita soittaa, kopioimalla aina edellisen hahmon ja muokkaamalla niiden toimintoja. Kokeilkaa erilaisia valmiita ääniä sekä omien äänien äänittämistä. Muistakaa suurentaa ohjelma **koko ruudun kokoiseksi.** (3)



Minkälainen on teidän fantastisin rumpukone? Minkälaisia ääniä sillä voi soittaa? Liikkuuko se soitettaessa? Kokeilkaa seuraavaksi asettaa tietokoneen kamera siten, että koko luokka voi soittaa rumpukonetta yhdessä.

Taiteilijahaastattelu – Jari Suominen



Kuva: Samy Kramer

Jari Suominen on turkulainen muusikko, mediataiteilija ja tutkija. Hänen erikoisalaansa ovat elektroniikka, ääni ja ohjelmointi.

Mistä koostuu työnkuvasi taiteen, median ja tekniikan parissa?

Teen mediataidetta, johon monesti liittyy ääni. Lisäksi toimin osana Anna Breu -kollektiivia. Teen paljon myös yhteistyötä monen taiteilijan kanssa. Usein toimin tällöin teknikkona tehden mittatilausprojekteja taiteilijoille.

Millaisia teemoja teoksissasi ja työskentelyssäsi nousee?

Usein käytän sanaparia minimalistinen-psykedelia. Mutta ehkä siihen tulee lisätä helsinkiläisen muusikon Arttu Partisen käyttämä käsite pöllöily – tietynlainen asioiden naivi kyseenalaistaminen.

Vaikuttaako pöllöily myös teknologiasuhteeseesi?

Ei oikeastaan. Mitä tulee tekniikkaan, niin se on mulle media tai työkalu. Sen pitää olla tietyssä mielessä näkymätöntä. Joskus installaatiossa tekniikka jätetään tarkoitukselliseti näkyviin. Useinkaan en pidä siitä, että tekniikka tai laite itsessään on se teos. Tietysti arvostan tekijöitä, vaikka henkilökohtaisesti vierastan sellaista lähestymistapaa.

Missä vaiheessa teknologia tulee mukaan teokseesi? Kuinka paljon teknologia määrittelee lopullista teosta?

Sanoisin, että siinä vaiheessa kun teosta pohditaan, helpottaa kymmenen vuoden kokemus teknologian luomisen parissa paljon. Varsinkin Anna Breun kanssa kun brainstormataan uutta teosta ja käydään läpi konsepteja, niin hyvin harvoin tulee vastaan konsepteja, että täytyy sanoa ettei me osata tehdä tätä. Ja usein siitä vielä jatketaan

Kuva: Shogun Kunitoki

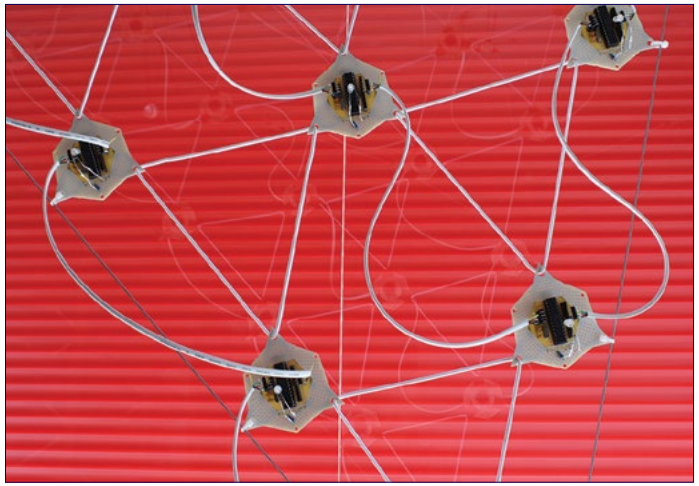
kehittelyä. Jos menee hyvin, niin päädytään siihen ettei edes tule tekniikkaa, vaan teemme jotain muuta, esimerkiksi veistoksen. Teoksen luominen on aina etenemistä eräänlaisella polulla, jossa ei tarvitse pohtia onko jokin mahdollista vai ei. Tyypillisesti meillä se kaari on sellainen, että mennään yksinkertaisesta hyvin monimutkaiseen ja takaisin yksinkertaiseen.

Olet käyttänyt käsitettä mediataiteilijan työkalupakki. Mitä se sisältää?

Työkalupakki on pikkaisen väärä suomennos kun englanniksi sana olisi "framework". Tarkoittaa isoa kasaa valmiita "palikoita" jotka on suunniteltu niin, että niitä voi vapaasti yhdistellä. Kun niitä käyttää koko ajan niin ne on jo aika hyvin testattu. Joku projekti voi mennä yhdellä ja jossain tarvitsee kaksi, ja ne voi viedä yhteen. Monesti ne on täysin Linux-pohjaisia komentoriviohjelmia. Sitten on iso läjä Arduino-koodeja joita käytän, ja ne on valmiina muokattaviksi.

Miten mielestäsi ohjelmoitava teknologia sopii taiteen tekemisen välineeksi?

Nykyään on paljon päteviä tyyppejä, jotka tuntee systeemit ja tietää miten asiat tehdään, jolloin pystyy ilmaisemaan itseään niillä välineillä ilman kompromisseja. Ja se vaatii ennen kaikkea kokemusta. Toisaalta mediataiteessa ja ennen kaikkea sähkötaiteessa



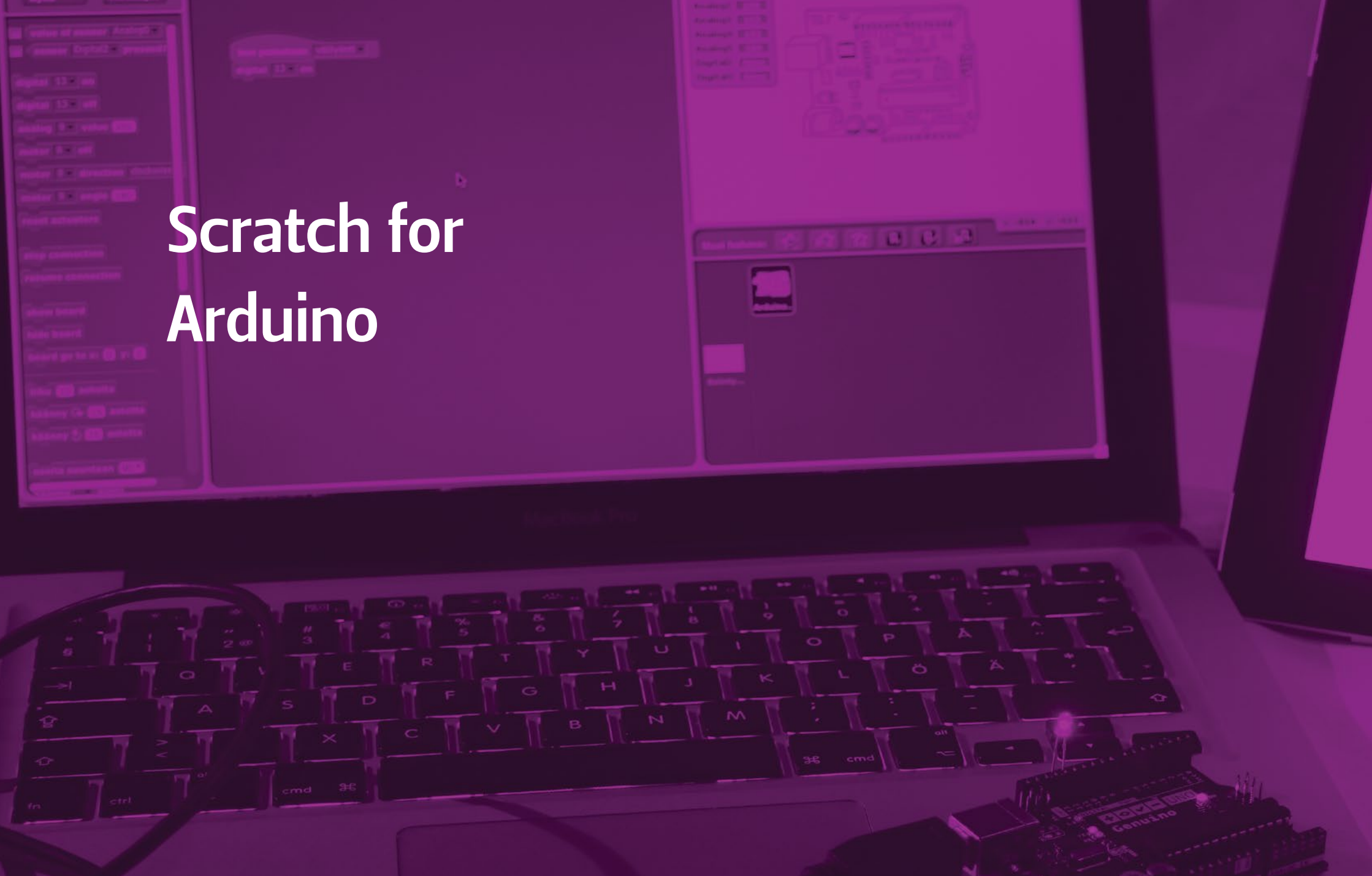
Kuva: Jari Suominen

on omituista, että moni patentoi tekniikoita. Tilanne on vähän sama kuin joku sanoisi maalarille, että tuolla tekniikalla ja tuo asia on jo maalattu.

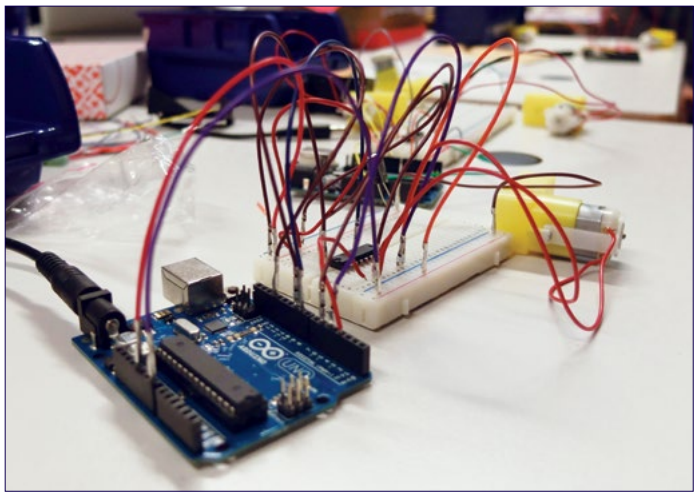
Mistä oma kiinnostuksesi mediataidetta ja ohjelmoitavaa teknologiaa kohtaan on lähtenyt?

Ekalta luokalta asti oon koodannut ja näpytellyt jotain, vaikka toki oli pitkä aika etten tehnyt mitään varsinaista koodausta. Yläasteella tehtiin jotain kokeiluja. Sitten opiskelin tietojenkäsittelytiedettä yliopistolla, josta en tiedä miten sinne päädyin. Sitä kautta musiikin kautta tietokoneavusteisen musiikin tutkimukseen ja ääniohjelmointiin. Sitten kun meni Taikkiin (nykyinen Aalto-yliopisto) ja siellä käydyn Arduino-mikrokontrol-lerikurssin kautta sai sitten alkusysäyksen että tää usta pitäis ottaa haltuun. Siitä ei kauaakaan kun tein jo ensimmäisen tilaustyön ja siitä eteenpäin projekti projektilta syventynyt kiinnostus ja taidot.

Scratch for Arduino



Laitteiden ohjelmointia Arduino-kehitysalustalla ja graafisella ohjelmointiympäristöllä



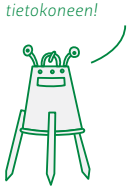
Tässä osiossa tutustutaan Arduino-kehitysalustan käyttöön Scratch for Arduino -ohjelmointiympäristön avulla. Arduino on avoin digitaalisesti ohjelmoitava elektroniikan kehitysalusta, jota voidaan ohjelmoida ohjaamaan esimerkiksi valoja ja moottoreita. Sen kanssa voidaan myös käyttää sensoreita jotka reagoivat ympäristöön. Arduinoa käyttämällä voit rakentaa itse laitteiston kytkemällä siihen haluttuja komponentteja sekä ohjelmoimalla laitteistolle toimintaohjeet.

Arduino koostuu kahdesta pääosasta, jotka ovat fyysinen Arduino-kehitysalusta ja sen ohjaamiseen käytettävä ohjelmointiympäristö Arduino IDE, jota käytetään tietokoneella. Arduino IDE on korvattavissa erilaisilla vaihtoehtoisilla ohjelmointiympäristöillä, kuten oppaassa käyttämällämme Scratch for Arduino (S4A) -ohjelmointiympäristöllä. S4A on helppo askel oppaassa aiemmin tutuksi tulleen graafisen Scratch-ohjelmoinnin jälkeen.

Kaikki ohjelmointiympäristöt toimivat jollakin ohjelmointikielellä, ja ne ladataan lopuksi ohjelmoitavaan koneeseen. Latausvaiheessa ohjelmointikieli muutetaan konekieleksi, joka koostuu lopulta binäärisestä, siis luvuista 1 ja 0 koostuvasta, digitaalisesta lauseesta. Tällöin laitteessa sähkövirrat ovat joko päällä tai pois päältä. Näitä digitaalisia sähkövirtoja ohjataan siis ohjelmoimalla. Arduino-kehitysalusta toimii siis kuin rakennettavan laitteen aivoina, ja muu laitteen elektroniikka sen kehona. Sen eri liittimistä eli pinneistä voi ajatella lähtevän erilaisia hermosäikeitä kohti kehon eri osia käskien komponentteja menemään päälle ja pois. Ohjelmoijana syötät halutut ajatukset eli koodin laitteen aivoihin, josta eri komennot lähtevät eri osille.

Ei muuta kuin ohjelmoimaan sähkövirtoja!

Huomioithan myös, että et voi käyttää Arduinoa tabletilla, vaan tarvitset tietokoneen!



Scratch for Arduinon asentaminen

Tarvikkeet:
- Tietokone
- Arduino-kehitysalusta
- USB A to B -kaapeli

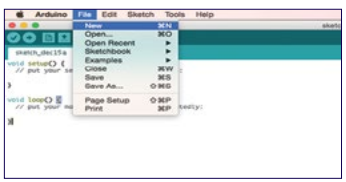
Arduino-kehitysalustaan pitää tehdä pieni ennakkovalmistelu Arduino IDE:n kautta ennen kuin voit siirtyä käyttämään S4A-ohjelmointiympäristöä. Ohjelmassa tulee ladata firmware-skripti, jotta Scratch for Arduino osaa ohjata Arduino-kehitysalustaa, kun se liitetään kiinni tietokoneeseen.

Asentamisessa on kolme vaihetta: Arduino IDE -ohjelmointiympäristön asentaminen tietokoneelle, S4A firmw sentaminen tietokoneelle.

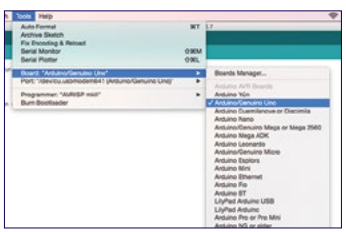
Arduino IDE:n lataaminen ja esiasetukset

Kun asennat Arduino IDE:n tietokoneellesi, ohjelman nimi on pelkästään Arduino. Selkeyden vuoksi puhumme tässä oppaassa Arduino IDE:stä puhuttaessa tietokoneelle asennetusta ohjelmasta.

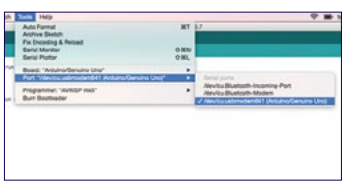
Lataa ensin Arduino IDE osoitteesta *www.arduino.cc* > Downloads. Kun olet asentanut Arduino IDE-ohjelmointiympäristön, voit avata ohjelman.



Jos uusi projekti ei avaudu automaattisesti, tai sinulla on auki vanha projekti, valitse File > New. Liitä Arduino kiinni tietokoneeseen USB A to B -kaapelilla.



Tarkista seuraavaksi, että sinulla on Arduino IDE:ssä oikeat asetukset oikealle Arduino-kehitysalustalle. Mene valikkoon Tools > Board ja valitse Arduino/Genuino Uno. Mikäli käytät jotakin toista Arduino-kehitysalustaa, valitse sen mukainen nimi valikosta.



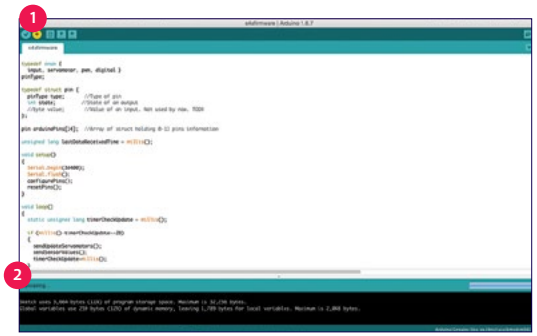
Seuraavaksi valitse käyttöön oikea USB-portti menemällä Tools > Serial Port-valikkoon. Mikäli käytät Windows-käyttöjärjestelmää, löytyy oikea yhteysportti listalta jostakin COM-alkuisesta vaihtoehdosta. Applen käyttöjärjestelmissä valitse /dev/cu.usbmodem-alkuinen vaihtoehto. Linux-käyttöjärjestelmissä oikea portti löytyy /dev/ttyACM-alkuisesta vaihtoehdosta.

S4AFirmwaren lataaminen koneelle ja Arduinoon

Etsi seuraavaksi laiteohjelmisto eli firmware osoitteesta *s4a.cat* > Downloads -kohdasta löytyvän linkin kautta (tai suoraan *http://vps34736.ovh.net/S4A/S4AFirmware16.ino*).

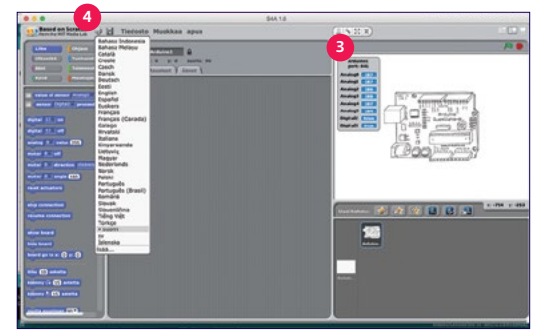
Ohje jatkuu seuraavalla aukeamalla.

Kopioi ja liitä edelliseltä sivulta löytämäsi firmware-skripti Arduino IDE:hen. Lataa se Arduinoon ohjelman yläreunassa olevaa **Upload-kuvaketta** (1) painamalla. Huomioithan että Arduino-kehitysalustan tulee olla liitettynä USB A to B -kaapelilla tietokoneeseen.



Kun Arduino ilmoittaa **"Done uploading"**, (2) voit avata S4A-ohjelman, jonka asentaminen neuvotaan seuraavassa kappaleessa. Nyt S4A-ohjelma kykenee kommunikoimaan Arduino-kehitysalustan kanssa USB-kaapelin ollessa kytkettynä kehitysalustan ja tietokoneen välillä. Muistathan, että firmware tulee ladata jokaiseen Arduino-kehitysalustaan aina erikseen.

Huomaathan että S4A-ohjelma ei saa olla auki samaan aikaan, kun lataat firmwarea Arduino IDE:n kautta!



S4A ja sen lataaminen

Lataa Scratchiin pohjautuva Arduinon ohjelmointiympäristö osoitteesta s4a.cat -> Downloads.

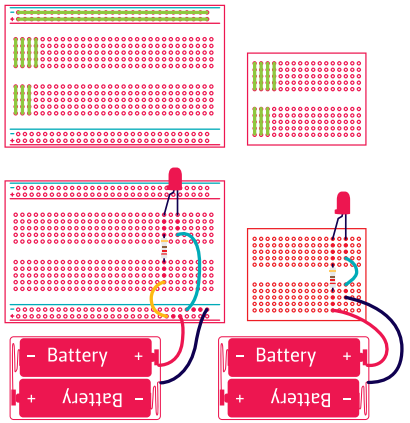
S4A-ohjelmointiympäristön käytössä ajatuksena on, että voit tutustua helposti ja selkeästi ohjelmoinnin logiikkaan aiemmin tutuksi tulleella Scratchiin pohjaavalla graafisella ohjelmointikielellä, sekä soveltaa jo aiemmin oppimaasi elektroniikan ohjelmoimiseksi.

S4A-ohjelmointiympäristö mukailee alkuperäisen Scratchin (v. 1.0) ulkoasua. Kaikki sen toiminnot ovat kuitenkin pääosin yhtäläiset Scratchin uusien versioiden kanssa. Pelkästään Arduinoa ohjelmoitaessa (käytettäessä ledejä, pietsoelementtiä tai moottoria) et juurikaan tarvitse S4A:n **ohjelmaruutua.** (3) Ohjelmaruutu on kuitenkin tarpeellinen, mikäli haluat luoda ohjelman joka reagoi ulkoiseen painikkeeseen. Voit vaihtaa ohjelman suomen kieliseksi **maapallokuvakkeesta.** (4)

Muista aina S4A-ohjelmointiympäristöä käyttäessäsi, että Arduinossa tulee olla ladattuna S4AFirmware -koodi, eli oikea laiteohjelma, jotta kykenet ohjaamaan Arduino-kehitysalustaa. Jos käytät välissä Arduinoa sen omassa IDE-ohjelmointiympäristössä, tulee sinun aina ladata S4AFirmware-koodi uudelleen Arduinoon IDE:n kautta.

Koekytkentälevyyn tutustuminen

Scratch for Arduinon ja Arduino-kehitysalustan avulla voimme ohjata ja ohjelmoida elektronisia komponentteja, kuten moottoreita, valoja tai mikropiirejä. Rakentaessa laitetta käytämme koekytkentälevyä. Koekytkentälevy on kuin piirilevy, jossa voimme kiinnittää ja irrottaa komponentteja helposti kokeillessamme eri kytkentöjä oikean ratkaisun löytämiseksi. Tutustumme ensin koekytkentälevyyn ilman Scratch for Arduinoa luomalla suljetun virtapiirin painikkeella.



Tarvikkeet:

- Led
- 220 ohmin etuvastus (punainen, punainen, ruskea, kultainen)
- 2 × AAA -paristoja
- Kahden AAA-pariston paristokotelo
- Hyppylankoja
- Painikenappi

Huom. Kytkentäkaavion suurennus löytyy oppaan sivulta 94. Kuva:1.

Koekytkentälevy rakentuu useista kytkentäkiskoista, joista näkyvillä on vain reiät. Kiskot on merkitty kuvassa vihreällä. Rivit toistuvat samalla tavalla koko koekytkentälevyssä. Koekytkentälevyjä voi olla eri kokoisia ja mallisia. Ohessa näet kaksi yleisintä mallia.

Jokainen kytkentäkisko voidaan yhdistää toiseen kytkentäkiskoon hyppylangalla. Hyppylangaksi kutsutaan tavallista lyhyttä sähköjohdon pätkää, jolla kytketään kaksi kytkentäkiskoa yhteen. Kuvassa on esitetty sinisillä hyppylangoilla erilaisia tapoja kytkeä useampi kisko yhteen.

Seuraavaksi kytketään led päälle koekytkentälevyllä tehdyllä kytkennällä. Liitä aluksi pariston (+) ja (–)-johtimet eri riveille. Nyt molemmilla kiskoilla kulkevat pariston eri navat.

Valitse seuraavaksi mikä tahansa poikittainen kisko, josta haluat lähteä tekemään ledille kytkentää. Liitä (+)-kiskolta hyppylanka valitsemaasi poikittaiskiskoon.

Etuvastus rajoittaa sähkövirtaa sen verran, ettei led tuhoudu liian suuresta sähkövirrasta ja jotta sähkö ei kulu liian nopeasti virtalähteestä eli paristosta. Vastusten voimakkuus ilmoitetaan ohmeina ("tunnus"), ja siitä käytetään käsitettä resistanssi. Sopiva vastuksen resistanssi on 220 ohmia. Vastuksen resistanssin suuruus luetaan väriraidoista.Esimerkiksi 220 ohmin vastuksen värikoodi on punainen, punainen, ruskea. Internetissä on monia laskureita, joiden avulla voit tunnistaa vastuksen resistanssin. Tunnistaminen on helppoa internetissä olevissa tunnistusohjelmilla tai vaikkapa kännykälle ladattavalla applikaatiolla. Voit hakea niitä esimerkiksi hakusanalla "Resistor color code calculator". Monesti myös myyntipaketeista löytyy värikartta, josta voit lukea vastuksen resistanssin.

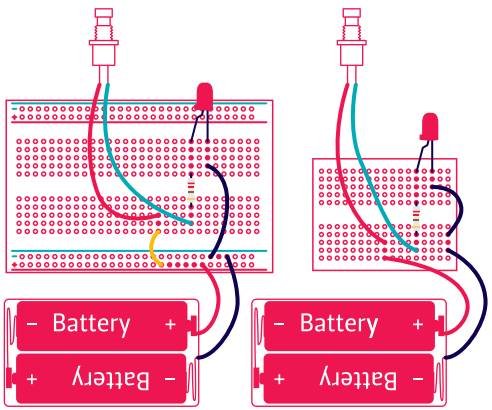
Ohje jatkuu seuraavalla aukeamalla.

Liitä seuraavaksi poikittaiskiskoon etuvastuksen toinen jalka. Kytke se lähtemään samasta rivistä johon hyppylanka tulee (+)-kiskolta. Liitä sitten vastuksen toinen jalka koekytkentälevyn keskiuran toiselle puolelle. Nyt etuvastus on kuin silta kahden kytkentäkiskon välillä.

Liitä seuraavaksi ledin pitkä jalka riville johon etuvastus tulee. Ledin pidempi jalka on sen (+)-jalka. (–)-jalka tulee puolestaan mille tahansa vapaalle poikittaiselle kytkentäkiskolle. Kytke lopuksi vielä hyppylangalla (–)-kisko yhteen saman (–)-kiskon kanssa, johon pariston (–)-johdin tulee. Kun kytket paristot paristokoteloon, led syttyy, sillä suljetussa virtapiirissä sähkövirta pääsee kulkemaan ja vaikuttamaan komponenttiin – tässä tapauksessa lediin!

Mikäli led ei syty, tarkista että kytkennät ovat kiinni ja menevät varmasti oikein siten, että jokainen jalka on yhteydessä vain seuraavan osan kanssa, jolloin komponentit ovat peräjälkeen luoden virtapiirin.

Huom. Kytkentäkaavion suurennus löytyy oppaan sivulta 95. Kuva:2.



Painikkeen liittäminen

Liitä seuraavaksi kytkentään vielä painike. Siirrä ensin etuvastus eri riville kuin missä on hyppylanka joka saapuu (+)-kiskolta. Liitä painikkeen toinen jalka (ei väliä kumpi) etuvastuksen kanssa samalle riville. Toinen painikkeen johto liitetään puolestaan hyppylangan kanssa samalle riville. Nyt painikkeesta painamalla muodostuu suljettu virtapiiri, jolloin sähkövirta pääsee kulkemaan ja sytyttää ledin.

Kuten huomaat, voit rakentaa tuttuja kytkentöjä monilla menetelmillä: hauenleukajohdoilla, sokeripaloilla ja muilla liittimillä, teippaamalla ja liimaamalla, juottamalla sekä koekytkentälevyllä.

Huomaa myös, että sähköjohtojakin on monenlaisia. Yksisäikeisen sähköjohdon liittäminen koekytkentälevyyn on helpompaa kuin monisäikeisen! Voit käyttää sitä esimerkiksi kun liität painikkeen, moottorin tai paristokotelon koekytkentälevyyn.

Ledin vilkuttaminen

S4A:llä voimme luoda ohjelman, joka on kuin painike. Voimme kirjoittaa automaattisen koodin, joka painaa "painikkeen" päälle ja ottaa sen pois päältä.

S4A-ohjelmointiympäristö mukailee Scratch-ohjelmointiympäristöä. Seuraavaksi luomme koodin, joka käynnistää koekytkentälevyllä olevan ledin. Valmistele laitteeseen sopiva kytkentä koekytkentälevyllä. Käytä kytkennän tekemiseen apuna kuvaa. Tässä tehtävässä Arduino antaa tarvittavan sähkövirran ledille.

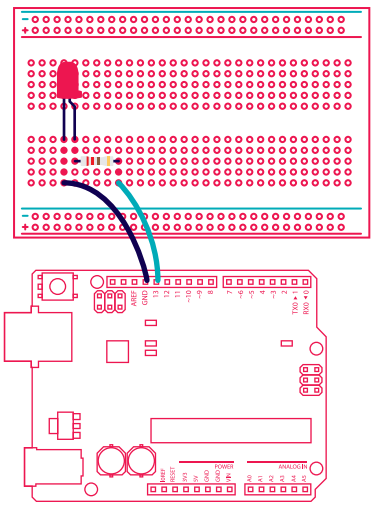
Tarvikkeet:

- Arduino UNO
- Led
- 220 ohmin vastus
- Koekytkentälevy
- Hyppylankoja
- USB A to B -kaapeli

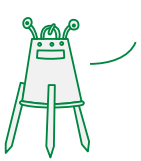
Huom. Kytkentäkaavion suurennus löytyy oppaan sivulta 95. Kuva:3.

Kytkennän rakentaminen

Aloita kytkemällä led. Ledin lyhyt eli (–)-jalka, kytketään hyppylangalla Arduinon pinniin jossa lukee GND. Tämä on kuin Arduinon (–)-napa, eli maa (engl. ground).

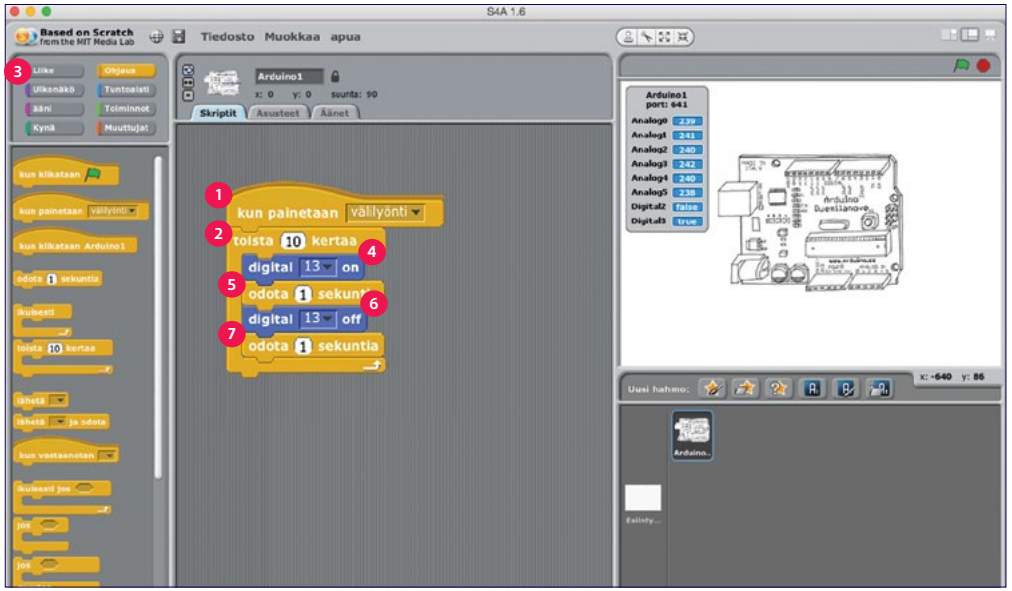


Ledin pitkä (+)-jalka täytyy kytkeä sivuttaissuunnassa eri kiskolle kuin ledin (–)-jalka. Samalta kiskolta, johon ledin (+)-jalka asetetaan, täytyy lähteä vastus. Tässä tapauksessa vastuksen tehtävänä on suojata lediä, jottei sen läpi pääse kulkemaan liian paljon sähkövirtaa, joka rikkoisi sen. Tällaista vastusta kutsutaan etuvastukseksi. Etuvastuksen voit kytkeä miten päin tahansa, mutta muista, että senkin jalat tulevat eri kiskoille. Vapaaksi jääneen jalan kiskolta kytketään lopuksi hyppylangalla vastus Arduinon pinniin 13.



Muistatko vielä, miten ledin saa pidempien johtojen päähän? Voit hyödyntää tekniikkaa tässä tehtävässä, ja alkaa valmistelemaan ohjelmoitavaa veistosta!

Ohje jatkuu seuraavalla aukeamalla.



Ohjelmakoodin rakentaminen

Aloita koodin rakentaminen Ohjaus-välilehdestä. Raahaa Skriptit-välilehteen **"kun painetaan välilyönti" -komento.** (1) Tämä komento määrittelee ohjelman alkamaan, kun tietokoneen välilyöntiä painetaan tietokoneen näppäimistöstä. Määrittele samasta Ohjaus-välilehdestä toiminnon määreeksi **"toista 10 kertaa".** (2) Voit kokeilla myöhemmin vaihtaa kertojen lukumäärää.

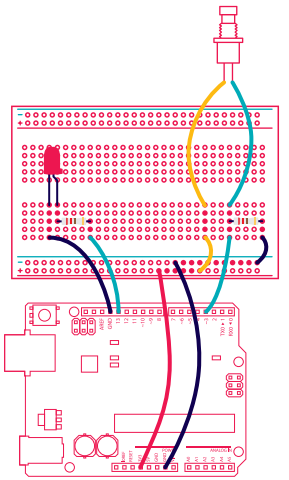
Seuraavaksi muokkaamme halutuksi toiminnoksi vilkkuvan valon toistuvaksi edellä määritellyn 10 kertaa. Valitse seuraavaksi **Liike-välilehdestä** (3) komento **"digital 13 on"** (4) ja raahaa se edellisen komennon sisään. Skriptin palikoiden logiikka on tuttu Scratch-ohjelmointiympäristöstä. Raahaamalla "digital 13 on" komennon jatkoksi Ohjaus-välilehdestä **"odota 1 sekuntia"** (5)-komennon, voit määrittää ajan, jonka valo on päällä. Tämän jälkeen toiminnoksi määritellään Liike-välilehdestä valitsemalla **"digital 13 off",** (6) joka raahataan edellisten komentojen jatkeeksi. Lopuksi määritellään kauanko valo pidetään pois päältä. Tämä komento määritellään raahaamalla viimeiseksi komennoksi **"odota 1 sekuntia".** (7)

Nyt muotoiltu skripti vilkuttaa siis valoa 10 kertaa kun välilyöntiä painetaan.

Vilkkuva valo nappia painamalla

Kytkennän rakentaminen

Jotta voit ohjelmoida valon vilkkumaan nappia painamalla, tulee kytkentään tietysti liittää nappi. Rakenna kytkentä kuvan mukaisesti. Kytkennässä napin toinen jalka tulee kytkentäkiskolle, joka yhdistyy Arduinon 5 V:n käyttöjännitteeseen. Huomaa, että jännite voidaan ohjata aluksi koekytkentälevyn (+)-kiskolle, josta jännite ohjataan edelleen napin ohjattavaksi. Koekytkentälevyn (+)-kiskon voi ajatella toimivan kuin jatkojohto, joka liitetään seinän pistorasiaan.



Napin toinen pää yhdistetään pinniin 2. Kytkentää ei voi kuitenkaan jättää näin, sillä silloin vaarana on Arduinon pinnien tuhoutuminen liian suurella sähkövirralla. Jotta sähkövirta saadaan pienemmäksi, liitetään napista tulemaan vielä vastus joka yhdistetään koekytkentälevyn (-)-kiskolle. Sama (-)-kisko tulee liittää vielä Arduinossa vapaaseen GND-pinniin. Voit liittää myös led-valon GND-hyppylangan tähän samaan koekytkentälevyn (-)-kiskoon tai hypätä suoraan Arduinon GND-pinniin.

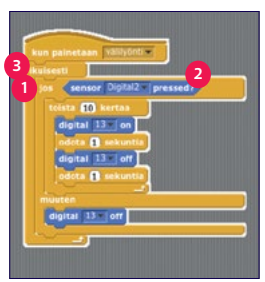
Tarvikkeet:

- Arduino UNO
- Led
- 2 x 220 ohmin vastus
- Painikenappi
- Koekytkentälevy
- Hyppylankoja
- USB A to B-kaapeli

Huom. Kytkentäkaavion suurennus löytyy oppaan sivulta 96. Kuva:4.

Ohjelmakoodin rakentaminen

Muokataan S4A-koodia siten, että valo alkaa vilkkumaan kun nappia painetaan. Jotta toiminto tapahtuisi, on aluksi luettava napin painallus, jonka jälkeen voimme määritellä mitä sitten tapahtuu.



Voit jatkaa skriptin rakentamista edellisen skriptin pohjalta. Raahaa samalle Skriptit-alueelle Ohjaus-välilehdestä **"jos / muuten" -komento.** (1) Vie komento 10 kertaa toistuvan vilkkuvan valon määreeksi. Komento pyytää täyttämään ehdon tyhjällä lokerolla. Valitse Liike-välilehdestä **"sensor digital2 pressed?"** (2) -ehto, ja raahaa se ehdon kohdalle. Lisää vielä "Muuten"-komennon alle "digital 13 off", jotta led ei pala, kun nappia ei paineta. Lisää Ohjaus-välilehdestä **"ikuisesti"-komento** (3) ja raahaa se välilyönnin painamisen jälkeen kattamaan koko skriptiä.

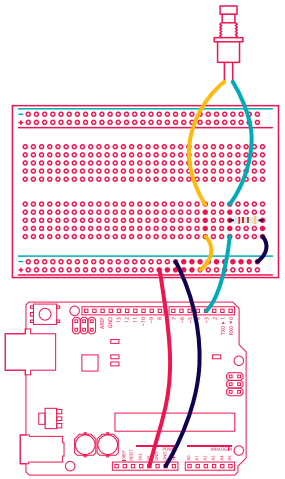
Nyt kun Arduino tunnistaa, että painike joka on kytketty pin2:een on painettu alas, aloittaa se sille määritellyn komennon. Komentona käytetään tällöin edellä ollutta valon vilkuttamista pinnissä 13.

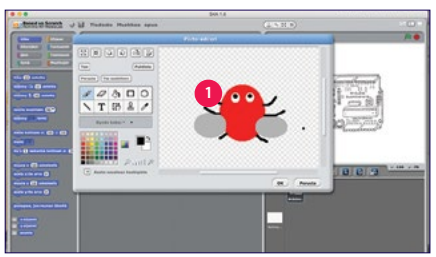
Painikenapilla ohjattava ötökkä

Tarvikkeet:
- Arduino UNO
- 220 ohmin vastus
- Painikenappi
- Koekytkentälevy
- Hyppylankoja
- USB A to B -kaapeli

Huom. Kytkentäkaavion suurennus löytyy oppaan sivulta 96. Kuva:5.

Opittuasi kytkemään ulkoisen painikkeen Arduinoon voit käyttää sitä myös S4A-ohjelmointiympäristössä luodun ohjelman ohjaamiseen. Voit siis luoda ohjelman, joka reagoi napin painallukseen -luotte siis ikäänkuin pelin ja peliohjaimen.





Kytkennän rakentaminen

Kytkentä rakennetaan samalla tavalla kuin neuvotaan luvussa "Vilkkuva valo nappia painamalla" mutta ilman lediä.

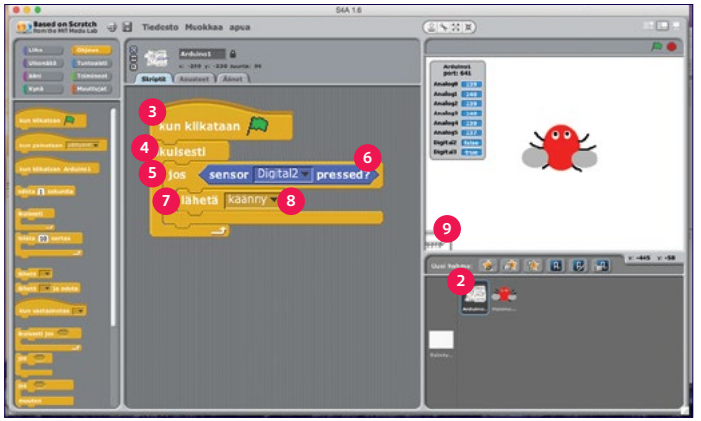
Ohjelmakoodin rakentaminen

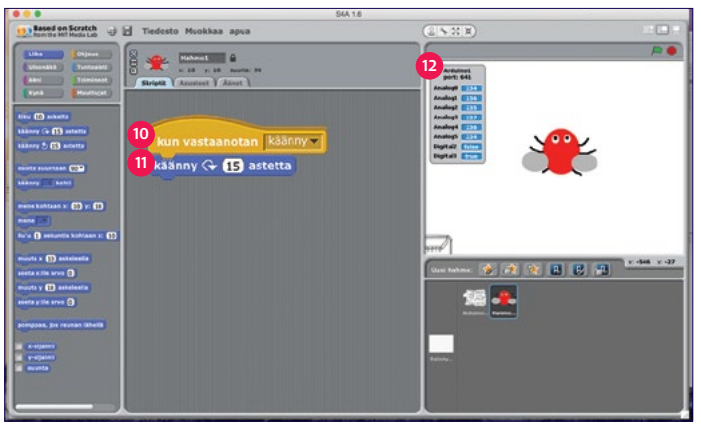
Luo uusi hahmo S4A-ohjelmointiympäristössä. Älä poista Arduino1-hahmoa, jotta voit jatkossakin lukea painiketta joka on liitetty Arduinoon.

Piirrä haluamasi hahmo **"Piirrä uusi hahmo" -valikosta.** (1) S4A:n hahmon ulkoasun muokkaamisen valikko poikkeaa ulkonäöltään Scratchin samasta valikosta, mutta se pitää sisällään samat työkalut. Pienellä tutustumisella löydät varmasti kaikki tarpeelliset työkalut. Muista asettaa asusteen keskipiste.

Palaa Skriptit-välilehteen ja **valitse aktiiviseksi Arduino1-hahmo.** (2) Aloita ohjelmakoodi Ohjaus-valikosta valitsemalla **"kun klikataan (vihreä lippu)"** (3) -komento ja raahaa se ohjelmointikenttään. Jatka ohjelmoitavaa komentosarjaa **"ikuisesti"** (4) -komennolla. Raahaa vielä tämän sisään ehtolause **"jos < >".** (5)Täytä ehtolauseen ehto valitsemalla Liike-valikosta **"sensor Digital2 pressed?"-komento.** (6) Tällä komennolla luetaan, onko Arduinon pinniin 2 liitetty painike painettu. Jatka koodia määrittelemällä lähetettäväksi viesti, mikäli painike on painettu. Valitse Ohjaus-valikosta **"lähetä [ ]" -komento** (7) ja raahaa se koodiin. Valitse komennon alavalikosta "uusi..." ja kirjoita viestiksi **"käänny".** (8)

Huomaathan että Arduino1-hahmo on siirrettävä ohjelmakentästä pois näkyviltä raahaamalla se ohjelmakentän alanurkkaan. (9)





Valitse aktiiviseksi Hahmo1. Valitse Ohjaus-valikosta **"kun vastaanotan [ ]"-komento.** (10) Täytä alavalikosta vastaanotettavaksi komennoksi "käänny". Tällöin, kun Arduinon nappia painetaan, lukee S4A-ohjelman Arduino1-hahmo painikkeen painalluksen ja lähettää viestin "käänny". Nyt Hahmo1 lukee "käänny"-viestin, ja tätä seuraavaksi komennoksi määritellään **"käänny 15 astetta".** (11)

Voit poistaa **Arduino1-hahmon arvoja ilmoittavan kentän** (12) ohjelmaruudusta klikkaamalla sitä hiiren kakkospainikkeella ja valitsemalla "piilota".

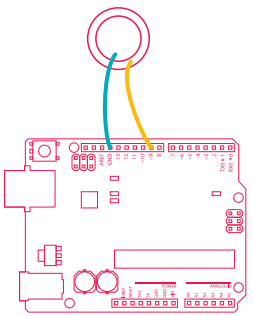
Seuraavaksi voit kokeilla rakentaa esimerkiksi painikkeeseen reagoivan piirustusgeneraattorin. Avaa piirustusgeneraattorin koodi S4A:lla, ja lisää tämän ohjeen painike lähettämään viesti, joka käynnistää ohjelman.

Mökälaite

Tarvikkeet:
- Arduino UNO
- Pietsoelementti
- USB A to B -kaapeli

Huom. Kytkentäkaavion suurennus löytyy oppaan sivulta 96. Kuva:6.

Tuotamme seuraavaksi ääntä Arduinon avulla. Voit siis käyttää Arduinoa myös ohjelmoitavana soittimena! Äänen tuottaa tässä tapauksessa pietsoelementti, joka värähtelee sähkövirran mukaan, ja tuottaa siten kuultavaa ääntä. Muista, että pietsoelementti on eri asia kuin pieni kovaääninen (kaiutinelementti), joka ei toimi tässä tapauksessa. Pienen kovaäänisen liittäminen suoraan Arduinon pinniin voi myös vahingoittaa Arduinoa.



Kytkennän rakentaminen

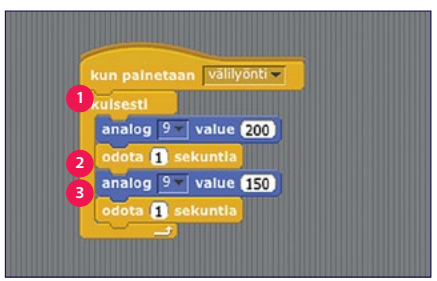
Kytkennän rakentaminen on hyvin helppoa. Kytkennässä pietsoelementin (+)-johto kytketään joihinkin Arduinon pinneistä 5, 6 tai 9. Esimerkissä kytkentä on tehty pinniin 9. (–)-johto liitetään Arduinon GND-pinniin.

Ohjelmakoodin rakentaminen

Arduino kykenee lähettämään tietyistä pinneistä (mm. 9, 6 ja 5) sähkösignaalia hyvin nopeasti eri aikavälein eli eri taajuuksilla. Muuttamalla sähkösignaalin taajuutta ääni kuuluu eri korkeuksilla. Taajuutta muutetaan nopeasti analog [ ] value ( ) -komennolla arvojen 0-255 välillä, eikä digital [ ] on/off -komennolla.

Aloita koodi esimerkiksi **“kun painetaan [välilyönti]” -komennosta.** (1) Valitse tapahtumaksi “ikuisesti”, jolloin haluttu ääni toistuu jatkuvasti.

Ohjelmakoodin rakenne on samanlainen kuin lediä vilkutettaessa. Komennoksi asetetaan tällä kertaa Liike-valikosta **“analog [9] value (200)”.** (2) Tätä arvoa muuttamalla voit muokata äänen taajuutta. Tämän jälkeen määritetään kuinka kauan ääni soi valitsemalla **“odota [ ] sekuntia” -komento** (3) ja antamalla sille haluttu arvo. Halutessasi voit jatkaa koodiin lisää erilaisia ääniä.



Esimerkissä ääni soi arvolla 200 yhden sekunnin ajan, jonka jälkeen arvolla 150 toisen sekunnin ajan. Tämän jälkeen koodi palaa jälleen alkuun ja jatkaa soimista ikuisesti.

Voit rakentaa tällä tavalla kuinka pitkän koodin tahansa ja luoda halutun äänen.

“analog [9] value (190), odota (0.5) sekuntia, analog [9] value (220), odota (0.7) sekuntia, analog [9] value (180), odota (1.5) sekuntia, analog [9] value (140), odota (2) sekuntia, analog [9] value (200), odota (1) sekuntia, analog [9] value (80), odota (5) sekuntia”

Ohjelmoitava sähkömoottori

Tässä tehtävässä opitaan tuottamaan ja ohjaamaan liikettä Arduinon avulla. Liikkeen synnyttää pienoissähkömoottori. Oleellista on huomioida, ettei moottoria voida liittää suoraan Arduinon ohjauspinneihin, sillä pinnit eivät voi antaa tarpeeksi virtaa moottorin käynnistämiseksi. Apuna käytetään transistoria, jolla ohjataan Arduinon 5V:n pinnistä 5V:n käyttöjännitettä moottoriin kytkemällä. Nyt käytössä oleva 2N3904-transistori toimii kuin kytkimenä, jota ohjataan sähkövirran avulla päälle ja pois.

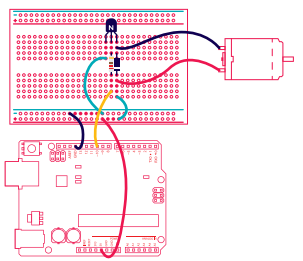
Tarvikkeet:
- Arduino UNO
- Koekytkentälevy
- 2N3904- tai 2N2222-transistori
- 220 ohmin vastus
- 1N4007-diodi
- Hyppylankoja
- Sähköjohtoa
- Pienoissähkömoottori jonka käyttöjännite on 5V (käyttöjännite ilmaistuna esim. 3-6V tai 4.5-9V) ja jonka ottama virta on enintään 150mA. Voit käyttää moottorina myös sellaista mini-värinämoottoria jonka arvot ovat edellä esitettyjen arvojen rajoissa.
- USB A to B -kaapeli

Huom. Kytkentäkaavion suurennus löytyy oppaan sivulta 97. Kuva:7.

Kytkennän rakentaminen

Kytkentä kannattaa hahmottaa kahtena osana. Toinen osa on Arduinon pinnistä 10 (voit käyttää Arduinon pinnejä 10, 11, 12 tai 13) lähtevä ohjausvirta, joka menee vastuksen kautta transistorin keskimmäiseen jalkaan. Ole huolellinen transistorin oikein päin kytkemisessä.

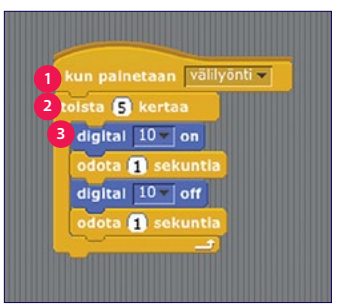
Toinen osa on virtapiiri, joka muodostuu Arduinon GND-pinnistä lähtevällä sähköjohdolla, joka menee transistorin vasemmanpuoleisimman jalan kautta transistorin oikean laidan jalkaan jatkaen sähkömoottoriin ja sähkömoottorin toisesta korvasta Arduinon 5V:n pinniin. Sähkömoottorin johtojen väliin tulee liittää diodi kuvan mukaisella tavalla. Huomioi diodin valkoisen merkkiviivan suunta, joka osoittaa että diodi on asetettu oikein päin (diodin jalat ovat samanmittaiset, eikä niistä voi katsoa sen napaisuutta).



Ohjelmakoodin rakentaminen

Ohjelmakoodi rakennetaan samalla tavalla kuin esimerkiksi ledin vilkuttaminen.

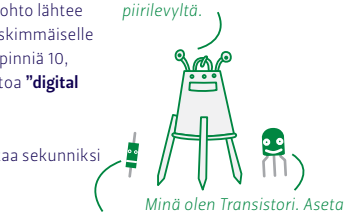
Aloita ohjelmakoodi valitsemalla komento, joka aloittaa toiminnon. Esimerkiksi **“kun painetaan [välilyönti]”.** (1) Tämän jälkeen tapahtumaksi määritellään kuinka monta kertaa haluttu toiminto toteutetaan. Esimerkissä moottori laitetaan päälle ja pois viisi kertaa. Raahaa Ohjaus-valikosta **“toista [ ] kertaa” -komento** (2) ja määrittele haluttu arvo. Tämän jälkeen määritellään oikea pinni käynnistymään ja menemään pois päältä halutuin aikavälein – aivan kuten ledin vilkutus -ohjeissa. Valitse pinni sen mukaan, mistä johto lähtee vastuksen kautta transistorin keskimmäiselle kannalle. Esimerkissä käytetään pinniä 10, joten koodissa käytetään komentoa **”digital [10] on/off”.** (3)



Moottori käynnistyy nyt viisi kertaa sekunniksi päälle ja sekunniksi pois.

Diodi on komponentti, joka päästää sähkövirran kulkemaan vain yhteen suuntaan. Oppaan kytkennässä sitä käytetään suojaamaan Arduinoa.

Transistori on nykyisin käytössä olevan digitaalitekniikan keskeisimpiä komponentteja. Transistoreja on yhdessä mikropiirissä usein miljoonia kappaleita, mutta kooltaan ne ovat vain hyvin pieniä. Tällainen mikropiiri löytyy esimerkiksi Arduinon piirilevyltä.



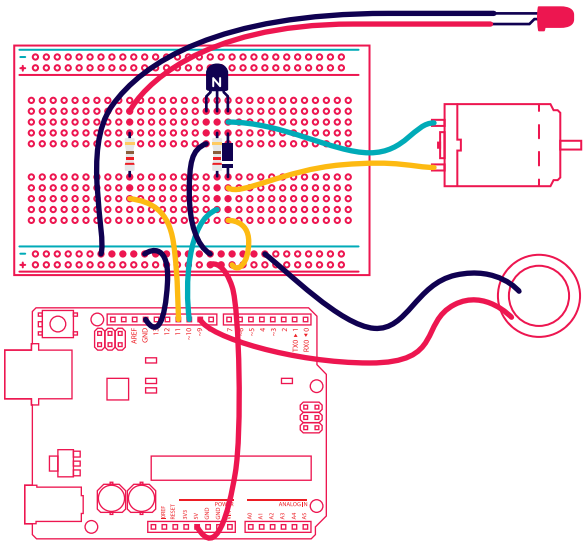
Minä olen Diodi.

Minä olen Transistori. Aseta minut koekytkentälevylle kasvot, eli tasainen pinta itseäsi päin.

Liikkuva, vilkkuva ja mökäävä veistos

Tarvikkeet:

- Arduino UNO
- Koekytkentälevy
- 2N3904- tai 2N2222-transistori
- 2 x 220 ohmin vastus
- 1N4007-diodi
- Pietsoelementti
- 1 × led
- Hyppylankoja
- Sähköjohtoa
- Pienoissähkömoottori tai mini-värinämoottori (ks. edellisen tehtävän tarvikelista)
- USB A to B -kaapeli



Lopuksi yhdistämme edellä esitetyt sisällöt yhdeksi kokonaisuudeksi eli teokseksi, joka rakentuu ohjelmoitavasta liikkeestä, valosta sekä äänestä!

Kokonaisuudesta voitte rakentaa juuri sellaisen teoksen kuin haluatte. Pohtikaa esimerkiksi mitä liike mahdollistaa ja mitä valo tuo lisää teokseen.

Kytkennän rakentaminen

Kytkentöjen rakenne kannattaa hahmottaa kolmena kokonaisuutena, joissa kytkennät on rakennettu erikseen yhdelle ledille, moottorille sekä pietsoelementille. Jokaisen komponentin (led, pietsoelementti ja sähkömoottori) kytkennät on esitelty edellisissä kappaleissa. Huomioi, että jokaisen komponentin (–)-johto yhdistyy koekytkentälevyssä yhteiseen (–)-kiskoon, joka johtaa mihin tahansa Arduinon gnd-pinniin.

Muista tehdä sähköjohdoista tarpeeksi pitkiä, mikäli haluat vetää moottorin, valot ja pietson hieman kauemmas koekytkentälevystä, esimerkiksi rakennettaessa liikkuvaa tärinärobottia. Kiinnitä osat robottiin käyttäen kuumaliimaa. Muista ettei sähkömoottorilla voi pyörittää kovinkaan painavaa kappaletta. Suosi siis hyvin kevyitä materiaaleja.

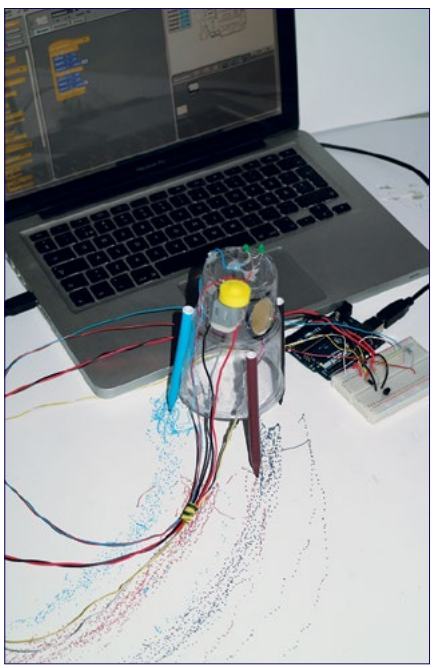
Liian suuri rasitus moottorissa voi aiheuttaa transistorin ja Arduinon vioittumisen. Voit liittää moottoriin esimerkiksi kevyen epäkeskokappaleen joka synnyttää tärisevän liikkeen tai voit käyttää valmista mini-värinämoottoria.

Ohjelmakoodin rakentaminen

Tämän jälkeen määritellään kuinka monta kertaa toiminto tapahtuu. Esimerkissä tapahtuman halutaan **toistuvan [5] kertaa.** (1)

Tapahtumassa käynnistetään **moottori (pinni 10),** (2) **led (pinni 11)** (3) sekä **pietso (pinni9)** (4) päälle sekunniksi, käyttämällä "digital [ ] on"-komentoja. Huomaa, että **pietson äänen arvoksi on määritelty 100** (5) (välillä 0-255) ja se asetetaan "analog [ ] value ( )"-komennolla.

Tämän tapahtuman kestoksi määritellään 1 sekuntia **"odota [1] sekuntia"-komennolla.** (6)



Tämän jälkeen toiminnot keskeytetään sekunniksi käyttämällä **"digital [ ] off"-komentoja,** (7) sekä **"analog [ ] value (0)"-komentoa.** (8)

Muista myös määritellä, **kauanko signaalit ovat pois päältä,** (9) ennenkuin palataan komentorivin alkuun. Tämä luuppi toistuu halutun määrän, eli esimerkissä 5 kertaa.soittamaan laskevan äänen.

Näin voit luoda päivitetyn version Robo1-julkaisun tärinärobotista! Nyt se liikkuu, vilkuttaa valoa ja päästää ääntä. Mitähän seuraavaksi?!

Arduinon ongelmatilanteita

Mikäli S4A-ohjelmointiympäristö ei tunnista Arduinoa, lataa uudestaan S4AFirmware käyttäen Arduino IDE -ohjelmointiympäristöä.

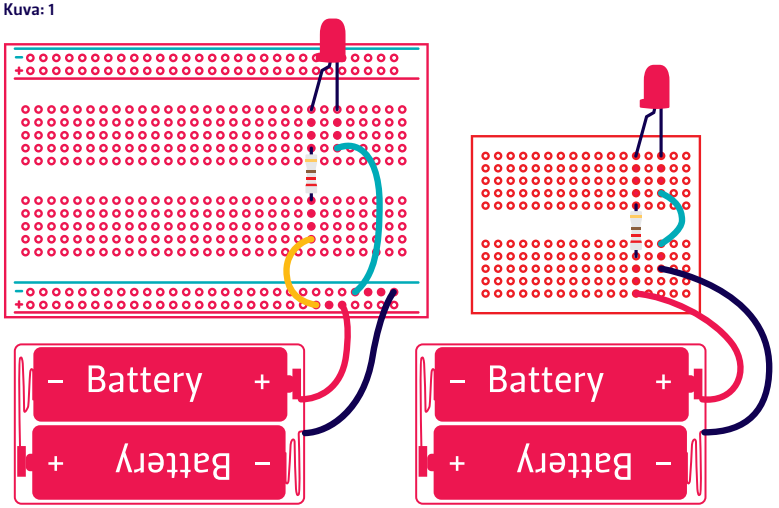
Tarkista, että Arduino IDE varmasti lataa S4AFirmware-ohjelman. IDE ilmaisee onnistuneen lataamisen "Done uploading" tekstillä ohjelmakentän alalaidassa.

Mikäli lataus ei onnistu, tarkista IDE:n Tools-valikosta, että käytössäsi on oikeat Board ja Port. Ohjeet löytyvät oppaan sivulta 81.

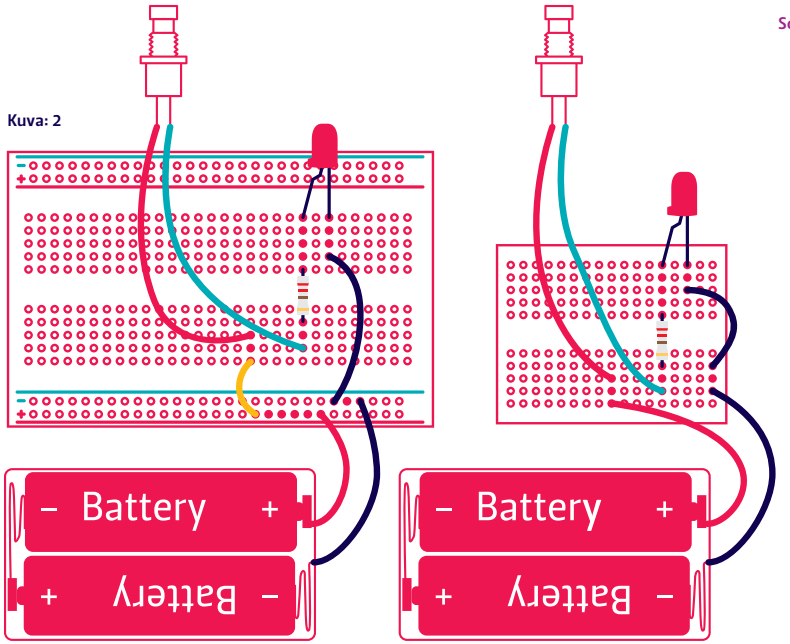
S4A:ssa ongelmatilanne, jossa haluttu ohjelma ei käynnisty, saattaa muodostua jos olet ohjelmoinut vahingossa väärää hahmoa. Tarkista, että ohjelmakoodit koskevat juuri niitä hahmoja, joita on tarkoitus ohjata. Sensoreita ja moottoreita käyttäessäsi ohjelmoi Arduino1-nimistä hahmoa.

Kirjassa esiintyvät Arduino-kytkentäkuvat

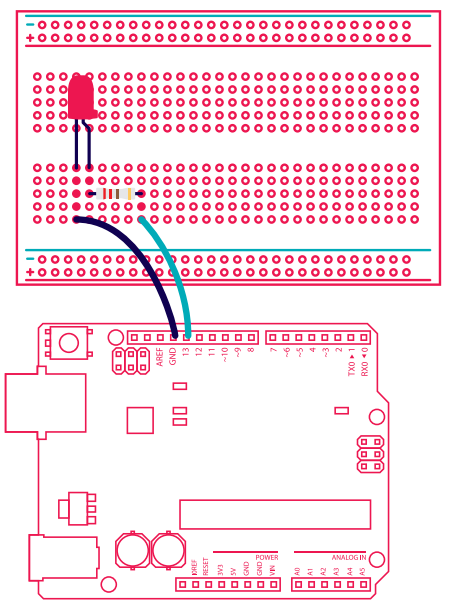
Kuva: 1



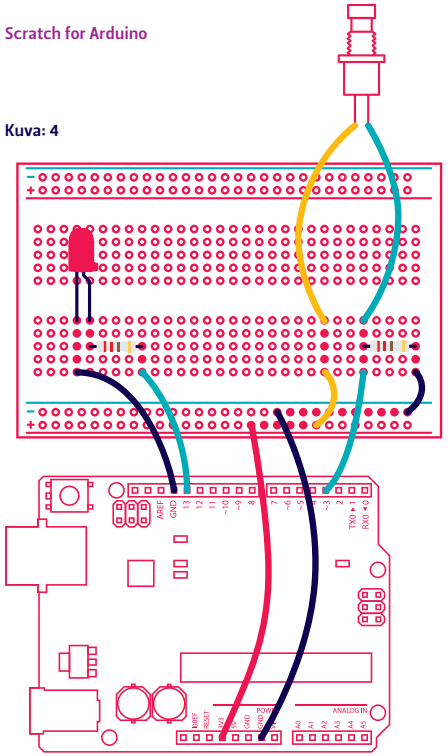
Kuva: 2



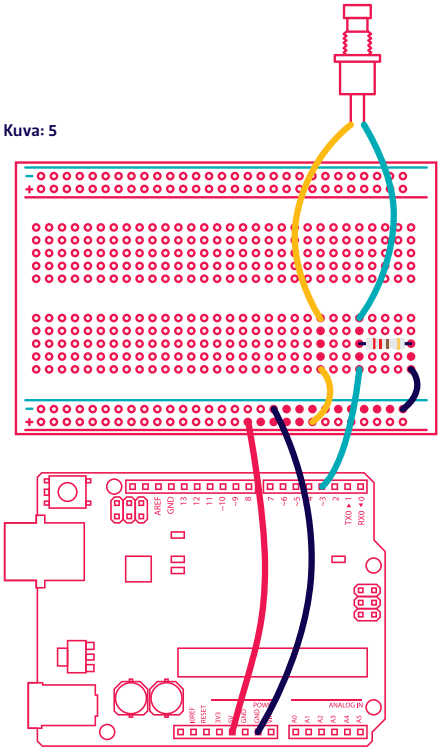
Kuva: 3



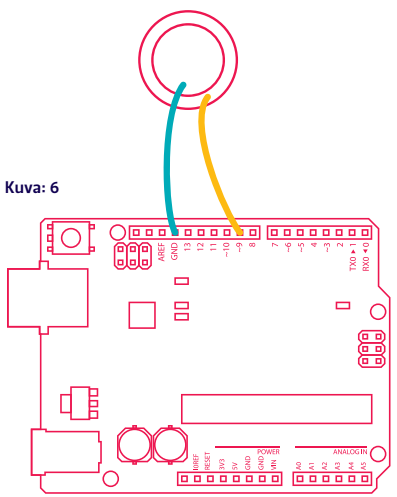
Kuva: 4



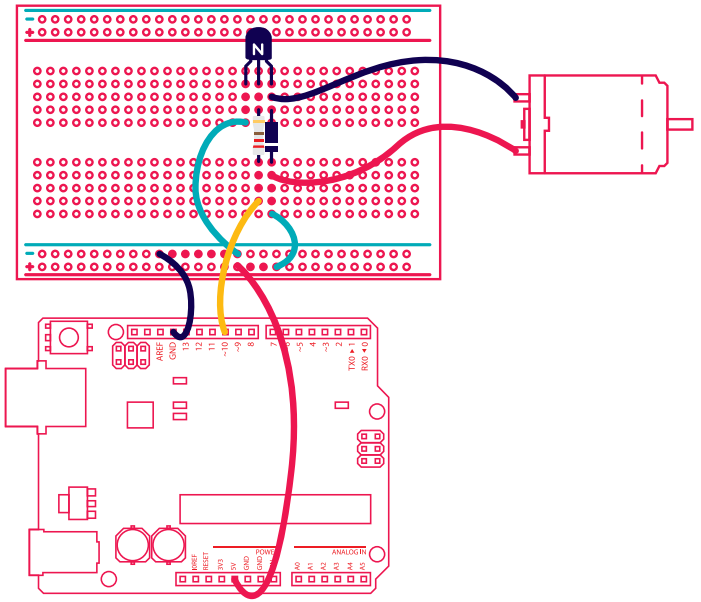
Kuva: 5



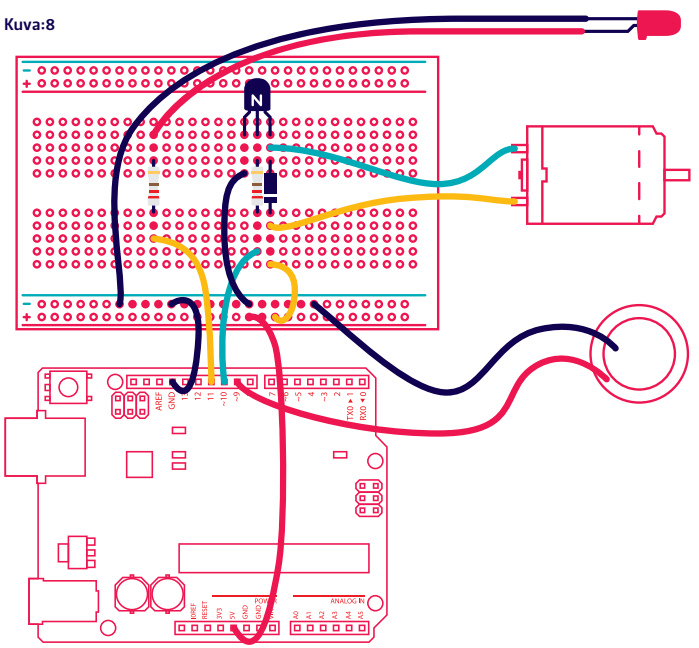
Kuva: 6



Kuva: 7



Kuva:8



ROBO2 – Elektroniikkaa ja ohjelmointia

Käsityökoulu Robotin ROBO2 – Elektroniikkaa ja ohjelmointia –opas johdattelee tutkimusretkelle teknologisoituvaan ja digitalisoituvaan maailmaan. Opas sisältää sarjan innostavia elektronisen rakentamisen ja ohjelmoinnin tehtäviä hyödynnettäväksi osana omaa opetustoimintaa. Oppaan sisällöt on suunnattu varhaiskasvatuksen ja alkuopetuksen opettajille ja muille kasvatusalalla toimiville itseopiskelumateriaaliksi ja ideapankiksi. Opas tähtää luomaan edellytyksiä toteuttaa taidekasvatuslähtöistä mediakasvatusta, jossa tavoitteena on oman kokemuksen tunnistaminen mediakulttuurin keskellä ja omaehtoisen ilmaisun luominen median keinoin. Oppaan kohderyhmänä ovat varhaiskasvatuksen ja alkuopetuksen toimijat, mutta tehtävistä löytyy haastetta ja innostavaa tekemistä kaiken ikäisille!

Oppaan tehtävät löytyvät myös Käsityökoulu Robotin kotisivujen tehtäväpankista osoitteesta *www.kasityokoulurobotti.fi*. Kotisivuilta on löydettävissä myös aiemmin ilmestynyt ROBO1 – Elektroniikkaa ja askartelua –opas.

www.kasityokoulurobotti.fi