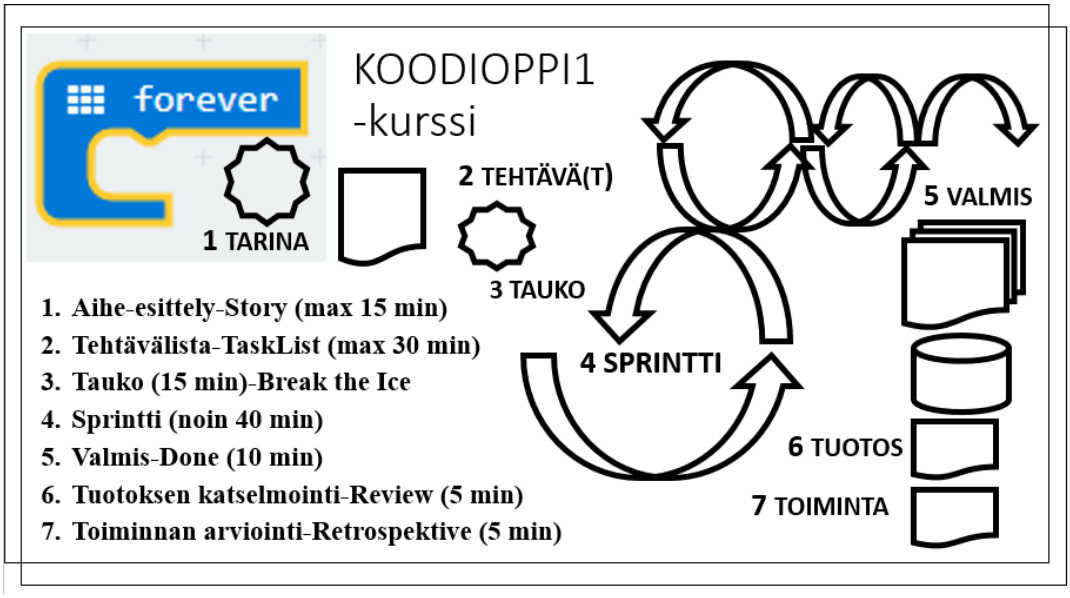
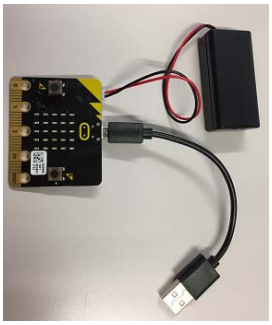
Otto Burman

Micro:bit
Koodioppi

Johdanto

OHJELMOINTIOSAAMISEN KEHITTÄMINEN





Kuva 1: Micro:bit-laite

Koodioppi-kurssit ovat suunnattu kaikille oppijoille. Se on suunniteltu johdannoksi myös peruskoulun päättäneille, opettajille sekä ohjelmoinnista kiinnostuneille vanhemmille, joilla on halu oppia ohjelmoinnin perusasioita.

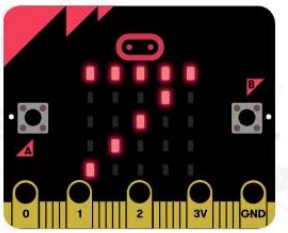
Läppäreiden peruskäyttötaito on edellytys oppimistuokioiden aikana viihtymiselle ja uuden oppimiselle. Osallistujan on osattava käyttää internettiä ja tunnettava jonkin verran englannin kielen perussanastoa. Kaikkein tärkeintä on halu oppia ymmärtämään tietojenkäsittelyn ja ohjelmoinnin peruskäsitteitä.

Oppimistavoitteet ovat asetettu sopiviksi kaikille, jotka haluavat oppia tai opettaa ohjelmoinnin perusteita. Materiaali perustuu sen tekijän yli 20 vuoden tietojenkäsittelyn opetuskokemukseen ja seurailee Microsoftin (2017) esittelemiä yleisiä perusajatuksia ohjelmoinnista (https://makecode.microbit.org/courses/csintro).

Kuva 2: Ohjelmointi

Ohjelmointiympäristönä on Micro:bit JavaScript editori, jonka avulla on helppoa ymmärtää ohjelmoinnissa käytetyt perusrakenteet. Aluksi ohjelmoidaan Block-editorilla. Taitojen kehittyessä siirrytään tekemään sovelluksia myös JavaScript-ohjelmointikielellä (`basic.showNumber(7)`). Ohjelmointiympäristön käyttö on helppoa ja sitä käytetään tavallisessa nettiselaimessa. Katso miltä se näyttää osoitteessa https://makecode.microbit.org/#.

Johdanto



Kuva 3: Käyttöliittymä

KOODIOPPI1-perusteet sisältävät viisi oppimistapahtumaa. Toteutus voidaan viedä läpi muutamassa päivässä tai yhden tai jopa viiden kalenteriviikon kuluessa. Tapahtumat ovat johdantoa ohjelmointiin ja sisältävät myös ohjausta ammattialan pedagogisiin työtapoihin ja peruskäsitteisiin. Oppiminen on suunniteltu toteutuvan enintään kahden tunnin työskentelyjaksoissa. Perusteita käsittelevässä osassa koodataan yksinkertaisia sovelluksia micro:bit-laitteelle. Sivustolla https://makecode.microbit.org/# on joitakin aiheeseen liittyviä esimerkkejä: valitse Projects-> ja sitten Examples, jolloin pääsee tutustumaan esimerkkeihin.

KOODIOPPI2-kurssit ovat syventäviä oppimistapahtumia ja käsittävät myös viisi toteutusta. Tämäkin osuus voidaan toteuttaa lyhyenä tai usean kalenteriviikon kuluessa. Syventävän kurssin aikana tehdään pienryhmissä oppimisprojekteja, joissa noudatetaan yhteistoiminnallisia oppimismenetelmiä ja ammattialalle tyypillisiä työ- ja toimintatapoja. Lisätietoja pedagogisista lähestymistavoista on kerrottu tarkemmin osoitteessa oppiscrum.fi.

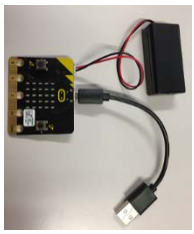
1 KOODIOPPI1-peruskurssit

Kaksoistuntien rakenne on suunniteltu toteutettavaksi noudattamaan yhteistoiminnallista oppimisprosessia seuraavin työskentelyvaihein:

- **Aihe-esittely**-Story (max 15 min): Tässä esitellään ”käyttäjätarina”, joka toimii johdantona sovelluksen rakentamisen ja siihen liittyvän tehtävälistan tekoon. Tavoite on suunnata toteuttajien kiinnostus esiteltyyn tehtävään ja sen toteuttamiseen.
- **Tehtävälista**-ToDo-TaskList (max 30 min): tiedon hakuja ja yhteisten tehtävien muodostaminen, jakaminen, käsitteistä keskusteleminen ja pohdintaa oman parin kanssa tai omassa pienryhmässä. Yhdessä sovitut tehtävät nimetään ja jaetaan ryhmän oppijoiden kesken. Tavoite on ymmärtää tehtävän kokonaisuus ja sen muodostuminen osatehtävistä, joiden toteuttamisesta ovat kaikki ryhmän jäsenet vastuussa.
- **Tauko**-Break the Ice(15 min): Intensiivisen työskentelyn mahdollistavat sopivat hengähdystauot ja ryhmän jäsenten vapaamuotoinen keskustelu. Tavoite on tutustua ja tutustuttaa osallistujat muihin ryhmän jäseniin.
- **Sprintti** (noin 40 min): Sovittujen osatehtävien toteuttaminen. Vaihe voi käsittää toteutuksen määrittelyä ja suunnittelua. Tärkein tavoite on aikaansaada ”toimiva” tuote tai tuotos. Tämän vaiheen aikana tehdään yhdessä sovittuja ohjelmointitehtäviä yksin, parityöskentelynä ja/tai pienryhmissä. Tekemällä oppimisen tavoite on myös vuorovaikuttaa ryhmässä ja oppia yhdessä oppimisen taitoja.
- **Valmis**-Done (10 min): Tämä vaihe käsittää valmiiden osatehtävien esittelyä sekä syntyneen tuotteen toiminnallisuuden testausta. Tärkeä osa on tarkoituksenmukainen dokumentointi (esimerkiksi ohjelmakoodin kommentointi) sekä laaditun sovelluksen tallentaminen myöhempää käyttöä varten.
- **Tuotoksen arviointi**-Review (5 min): Oppijat esittelevät toisilleen aikaansaannoksia. Syntyneitä tuotoksia korjaillaan tai parannellaan palautteen perusteella. Tavoite on oppia antamaan ja vastaanottamaan palautetta valmiin tuotoksen korjaamiseksi.
- **Toiminnan arviointi**-Retrospektive (5 min): Oppijat pohtivat ryhmän jäsenten toimintaa ja arvioivat myös omaa panostaan. Kunkin tapahtuman lopuksi kerrataan mitä saatiin aikaan ja kuinka työskentely sujui. Tavoitteena on oppia sanoittamaan omaa toimintaa ja osaamista.

Viiden oppimistapahtuman jälkeen valitaan yhdessä osallistujien joukosta ”KOODIOPPI1-koodari”, jonka valinnan kriteerinä on muiden osallistujien auttaminen ja tukeminen! Tästä annetaan kunniakirja nimetylle henkilölle.

1.1 Ensimmäinen KOODIOPPI1-kaksoistunti

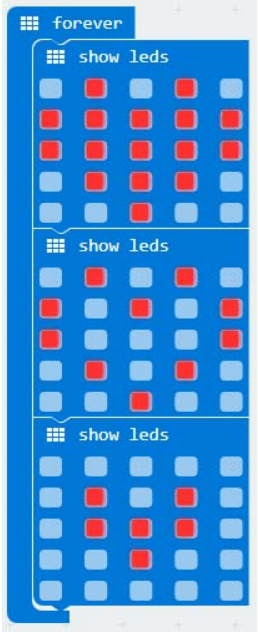


Kuva 4: Micro:bit ja usb sekä virtalähde

Ensin Micro:bit laitteisto liitetään usb-liittimen avulla tietokoneeseen. Sitten testataan kaikkien osallistujien laitteet sekä hiiren ja näppäimistön käytön perusosaaminen suorittamalla seuraava ohjelma ohjelmaeditorissa, joka on kuvassa 5.

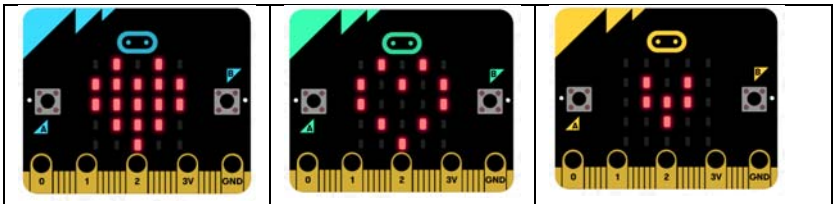
Editori aukeaa osoitteessa https://makecode.microbit.org/#.

Block Editori	JavaScript Editori

	`basic.forever(() => {`
	`  basic.showLeds(`
	`  . # . # .`
	`  # # # # #`
	`  # # # # #`
	`  . # # # .`
	`  . . # . .`
	`  );`
	`  basic.showLeds(`
	`  . # . # .`
	`  # . # . #`
	`  # . . . #`
	`  . # . # .`
	`  . . # . .`);
	`  basic.showLeds(`
	`  . . . . .`
	`  . # . # .`
	`  . # # # .`
	`  . . # . .`
	`  . . . . .`);
	`})`

Kuva 5: Ohjelmaeditorit

Tuloksena on animaatio, joka näkyy ohjelmaeditorin virtuaalisessa käyttöliittymässä (kuva 6).



Kuva 6: Sydänanimaatio

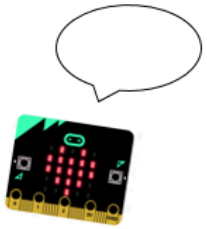
Sovellus ladataan omalle tietokoneelle Download-komennolla 

.
Ladattu tiedosto ”raahataan” / siirretään ladatut-kansiosta tietokoneen MICROBIT-kansioon. MICROBIT-kansio oli syntynyt automaattisesti, kun laite liitettiin usb-porttiin. Tästä kansiosta ohjelmakoodi (.hex) asentuu automaattisesti Micro:bit-laitteelle ja sovellus on nyt käytettävissä. Muista liittää virtalähde Micro:bittiin!

Xtra-lisätehtävä: Muuta sydän nauravan naaman animaatioksi → ☹😐☺ !

Johdanto

Seuraavaksi esitellään KOODIOPPI1 ensimmäisen oppimistapahtuman ja oppimisprosessin konkreettinen toteutusesimerkki. Tarkoituksenmukaista on muodostaa 3-4 (max 6) hengen pöytäkunnat ja näihin työparit, jotka toteuttavat tehtäviä yhdessä ja mielellään ääneen pohtien erilaisia ratkaisuja.



Aihe-esittely (max 15 min): Jokaisen oppimistapahtuman aihe-esittelyssä kuvataan tarinan tai kertomuksen muodossa mitä oppimistapahtuman tehtävällä tavoitellaan. Yksinkertainen esimerkki aloitukseen sopivasta ”tarinasta” voisi olla: ”*Mikä on sinun nimesi? Anna tietokoneen näyttää se!*”

Kuva 7: Kuka?

Usein tarina ja aihe-esittelyt edellyttävät tavoitteiden syvällistä käsittelyä ja edellyttävät oppijoilta uuden tiedon hankkimista, uusien käsitteiden ymmärtämistä sekä niiden osaamista esitetyn tarinan ymmärtämiseksi. Tämän aloitustehtävän tavoitteena on oppijoiden tutustuminen toisiinsa.

Aihe-esittelyn oppimistapahtumat käynnistetään yhteisellä pohdinnalla, joka ei vaadi välttämättä tietojenkäsittelylaitetta. Tarkoitus on saada oppijat keskustelemaan keskenään ja ensisijainen tavoite on, että aihe-esittely on hauskaa keskustelua ja vapaamuotoista rupattelua.

Toinen keino oppia ja opettaa uusia käsitteitä on johdattaa niihin konkreettisella ja usein kinesteettisellä tavalla. Tietojenkäsittelyssä on hyvä oppia piirtämään paperille, leikata ja sommitella uusia kuvia kuvakirjastoista ja aikakauslehdistä sekä tehdä alustava toteutusmalli konkreettisesti leikkaamalla ja liimaamalla!

Tehtävälista-TaskList (max 30 min)
Tarinaa ryhdytään purkamaan osiin ryhmässä muiden oppijoiden kanssa ääneen pohtien. Tuloksena syntyy sovelluksen tietojen ja toimintojen alustava määritys. Esimerkkitarinasta voidaan poimia käsitteitä sovelluksen rakentamista varten. Esimerkkitarinan kolme tietoihin ja toimintoihin liittyvää osaa ovat *(1) sinun nimesi (2) anna tietokoneen (3) näyttää se.* Käsitteet voidaan liittää laadittavan sovelluksen tietoihin ja toimintaan vaikkapa seuraavasti:

1. *sinun nimesi:* Pohditaan ja määritetään mitä tarkoitetaan tällä: Onko kyseessä oppijan itsensä etunimi tai sukunimi vai molemmat (vai onko kyseessä tietokoneen nimi?). Oppijat esittelevät itsensä vierustoverilleen ja sopivat millä nimellä tietokone esittelee omistajansa (tai itsensä).
2. *anna tietokoneen:* määritetään tietokone(laite) ja sen ominaisuuksia, jotka mahdollistavat tietojen syöttämisen, tietojen varastoinnin ja niiden esittämisen (eli sovelluksen arkkitehtuurin kuvaaminen, johon liitetään usein käsitteet käyttöliittymä, laskenta sekä tietovarasto). Keskustelun tuloksena sovitaan, että tietokoneena on micro:bit-laite, näyttönä ja käyttöliittymänä on laitteen led-käyttöliittymä ja tiedot

(nimi) esitetään kirjaimina (merkkijonona). Tässä voidaan yhdessä tutustua lyhyesti Block Editorin Basic- ja Input-rakenteisiin (https://makecode.microbit.org/#).

3. *näyttää se:* pohditaan ja määritetään miten tai missä tulos näytetään, jolloin kuvataan käyttöliittymää ja pohditaan ohjelmakoodia, jonka avulla nimi näkyy.

Aivan aluksi laadittavaa sovellusta pyritään hahmottamaan ilman tietokonetta paperilla ja kynillä sekä piirtämällä kuvia ja kuvioita selkeyttämään yhteistä näkemystä sekä kokonaiskuvaa laadittavasta sovelluksesta. Tässä esimerkissä voidaan tarkastella kuvaa 7 tarkemmin.

Sovelluksen ensimmäiseksi toteutettavaksi tehtävälistaksi ja työrupeamaksi (sprintti eli pyrähdys) muodostuu jotakuinkin seuraavan kaltainen tehtäväluettelo/-lista:

- Etu- tai sukunimen valinta – annetaan nimi:?
- Nimen esittäminen (show string) käyttöliittymässä – valitaan ohjelmalause: ?
- Ohjelmakoodin kirjoittaminen – laaditaan ohjelma: ?
- Siirretään ohjelma micro:bit-laitteelle – download-toiminto

Tauon jälkeen jatketaan listan kehittelyä.

Tauko (15 min) – Break the Ice
Kukin esittelee itsensä ryhmälle tai vierustoverilleen ja kertoo itsestään ainakin seuraavat seikat: Minä olen (nimi ja ikä) _____ ja ___ vuotta vanha, asun _____, harrastan _____, …

Sprintti (noin 40 min)
Tässä esimerkissä toteutus tapahtuu kolmessa työvaiheessa. Ohjaajan tehtävänä on huolehtia, että toteutus tapahtuu kiireettömästi ja että kaikki osallistujat saavat tuotoksen kolmannenkin vaiheen valmiiksi.

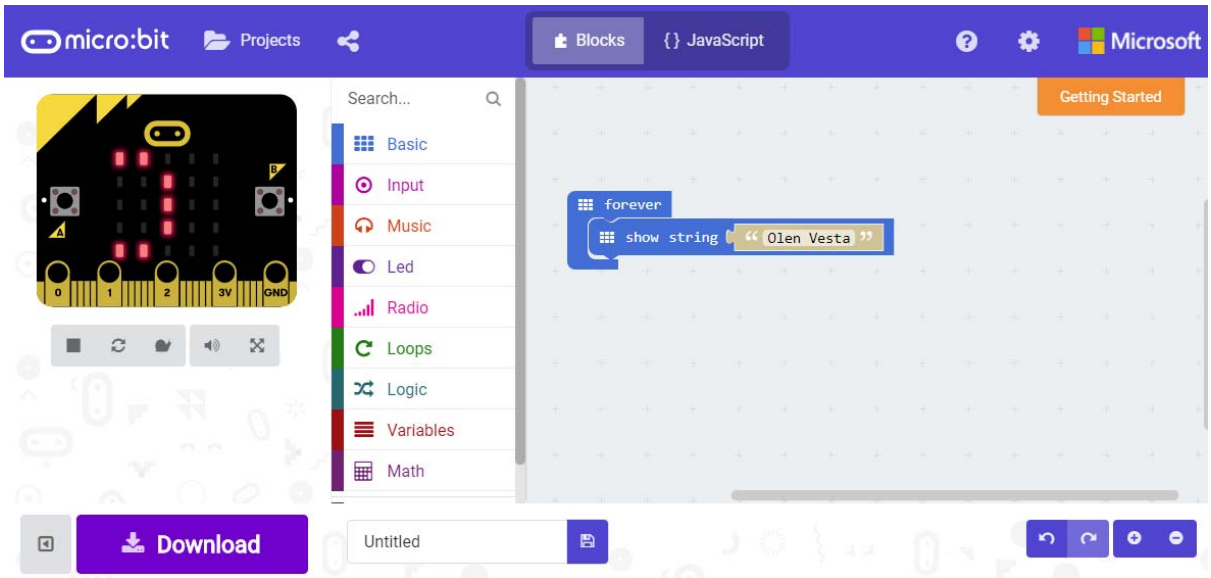
VAIHE 1
Ennen taukoa pohdittua työlistaa tarkennetaan ja sovelluskehityksen tehtävä-(todo-)listaksi saadaan esimerkiksi

- Nimen valinta (etu- vai sukunimi) ja sen kirjaaminen järjestelmään (show string) – valitaan ohjelmalauseet: ?
- Nimen esittäminen (for ever / on start) käyttöliittymässä – valitaan ohjelmalauseet: ?
- Ohjelmakoodin koostaminen ja sovelluksen laatiminen Block-editorissa – laaditaan ohjelma: ?
- Ohjelmakoodin testaaminen – suoritetaan ja arvioidaan esitettävät tiedot ja toiminta: ?
- Sovelluksen dokumentointi (ohjelmakoodin kommentointi) ja sen tallentaminen kahteen paikkaan hex-muodossa: (1) työkansioon myöhempää käyttöä varten ja MICROBIT-kansioon, josta sovellus siirtyy micro:bit-laitteelle.
- Syntyneen sovelluksen käyttö ja testaus sekä sen toiminnallisuuden esittely – esitellään muille osallistujille ja pyydetään palautetta: ?

Johdanto

Tehtäville sovitaan vastuuhenkilöt tai -henkilöitä, joiden tehtävänä on esitellä mitä ja miten tehtävä on tehty tai ratkaistu. Tämä ei tarkoita suinkaan, että vain nimetyt henkilö(t) olisi(vat) yksin vastuussa tehtävästä – kaikki auttavat kaikkia!

Ohjelmointitehtävänä tämä on varsin yksinkertainen ja sen ratkaisu on esitetty kuvassa 8. Tässä yhteydessä harjoitellaan käsitteitä ja pyritään näkemään kokonaisuus aitona tietojenkäsittelyn harjoitustehtävänä.



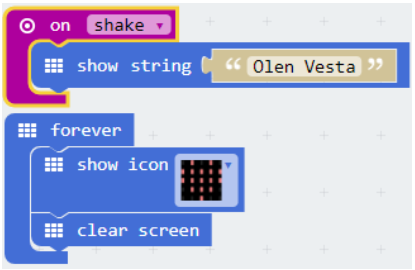
Kuva 8: Nimen esittäminen

VAIHE 2

Sovelluskehityksen seuraava vaihe muodostetaan saadun palautteen ja yhteisen pohdinnan tuloksena ja syntyy seuraava 2. sprintin tehtäväluettelo, sillä nyt huomataan, että käyttöliittymää voidaan kehittää. Esimerkiksi heiluttamalla micro:bit-laitetta (Input - on shake) aktivoi halutun tietokoneen toiminnan (show string). Tarinaa ja sovelluksen määritystä kehitetään ”*on shake*”-toiminnan pohjalta edelleen ja tarina voisi olla suunnilleen seuraava: ”*Laitteen ravistaminen kirjoittaa nimeni näkyviin!*”. Sovelluksen uusistunut tehtävälista muodostuu nyt seuraavaksi:

- Nimen esittäminen (on shake, show string) käyttöliittymässä – valitaan ohjelmalauseet: ?
- Ohjelmakoodin kirjoittaminen – laaditaan ohjelma: ?
- Pareittain testataan ja arvioidaan toimivuus: ?

Kuvassa 9 on esitetty yksi mahdollinen ratkaisu tähän tehtävään.



Kuva 9: On Shake - ravistaminen!

VAIHE 3

Tarinaa ja sovelluksen määritystä kehitetään vielä lisää, sillä tietokoneella voidaan automatisoida niin kysymys kuin vastauskin. Kolmas tarkennettu tarina voisi olla suunnilleen seuraava: *”Painamalla A-näppäintä laite kysyy ”Kuka?” ja näyttää sykkivän sydämen. Painamalla B-näppäintä se näyttää nimesi!”*

Sovelluksen ratkaisu JavaScript-kielellä (ilman sydämen tykytystä) on seuraava:

```
input.onButtonPressed(Button.A, () => {
    basic.showString("Kuka?")
})
input.onButtonPressed(Button.B, () => {
    basic.showString("Vesta?")
})
```

Ryhmä korjaa yhdessä ohjelmakoodia kolmannen määrityksen mukaiseksi! Ohjelma voidaan toteuttaa kopioimalla esitetyt ohjelmalauseet JavaScript-editoriin. Paina JavaScript-editori aktiiviksi kuten on esitetty kuvassa 10. Painamalla Block editorin näppäintä saadaan toinen esitystapa näkyviin.

Kuva 10: Editorin valintanäppäin

Valmis- Done (10 min)

Ohjelmakoodin testaaminen – suoritetaan ja arvioidaan sovelluksen toimintaa ja ladataan sovellus micro:bit-laitteelle. Esitellään sovellusta muille osallistujille ja pyydetään osallistujat esittelemään itsensä oman sovelluksensa avulla.
Sovelluksen dokumentointi: ohjelmakoodia tarkoituksenmukaisesti kommentoimalla.
Sovelluksen tallentaminen: tallennetaan (hex-muodossa) lataamalla se omalle koneelle ja siirtämällä tiedosto omaan työskentelykansioon.

Tuotoksen katselmointi-review (5 min)

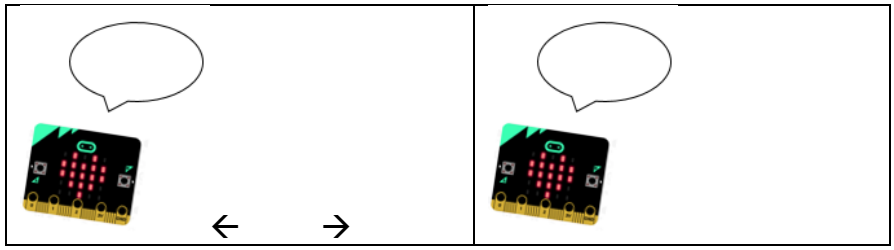
Kukin pohtii lyhyesti ääneen vierustoverille/ryhmälle syntynyttä tuotosta: mitä se tekee nyt ja mitä voitaisiin vielä mahdollisesti kehittää tai parantaa.

Toiminnan arviointi-retrospective (5 min)

Kukin kuvailee ääneen omaa toimintaansa vierustoverille/ryhmälle: kerrotaan siis mikä meni hyvin ja mitä voisi parantaa seuraavalla kerralla.

1.2 Toinen KOODIOPPI1 kaksoistunti

Aihe-esittely-Story (max 15 min)
Tarina: Tietokone1 Kysyy painettaessa A-näppäintä Tietokone2:lta ”Kuka olet?” Tietokone2 vastaa kysyjälle nimensä, kun B-näppäintä painetaan. Nimi näkyy Tietokone1:n näytöllä. Tietokone2 kysyy myös Tietokone1:ltä tämän nimen kun 2:n A-näppäintä painetaan, jolloin tähän vastataan painamalla Tietokone1:n B-näppäintä ja nimi näkyy Tietokone2:n käyttöliittymässä.



Kuva 11: Vuorovaikutteisen sovelluksen hahmotelma

Tehtävää varten tarvitaan kaksi Micro: bit-laitetta.

Tehtävälista-TaskList (max 30 min)

Tiedonhakutehtävä:
Mitä tarkoittaa viestiryhmän valinta ja mitä tekee ohjelmakoodi ”radio set group”?

Valitse käynnistettäessä radion viestiryhmä
Tietokone1, A-näppäin: Lähetä viesti ”Kuka?”
Tietokone1, B-näppäin: Lähetä viesti ”Vesta”
Tietokone1: Vastaanota ja näytä tullut viesti.
Tietokone2: Kopioi Tietokone1:n ohjelmat itsellesi.

Tauko (15 min)–Break the Ice
Kaikille osallistujille annetaan juokseva numero alkaen nrosta 1. Kukin parittoman numeron saanut lisää omaan numeroonsa luvun 1. Parit muodostetaan parillisten numeroiden mukaan: 2,4,6, ...

Sprintti (noin 40 min)
Ohjelmointitehtävä on varsin yksinkertainen ja sen koodi on esitetty seuraavassa kuvassa (

```
input.onButtonPressed(Button.A, () => {
    radio.sendString("Kuka?")
```

```
})
input.onButtonPressed(Button.B, () => {
    radio.sendString("Vesta!")
})
radio.onDataPacketReceived( ({ receivedString }) =>  {
    basic.showString(receivedString)
})
radio.setGroup(2)
```

VAIHE 1
Laaditaan ohjelma, ladataan se micro:bit-laitteelle ja testataan sovelluksen toimivuutta. Kukin vaihtaa omassa ohjelmassaan radio set group arvon parinsa kanssa samaksi ja kysyy A-nappia painamalla ”Kuka?” ja saa parilta vastauksen (pari vastaa painamalla B-näppäintä).

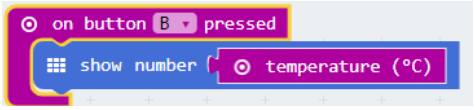
VAIHE 2
Seuraavaksi jokainen yrittää saada selville muiden osallistujien nimet. Vaihtamalla radio set group arvon joksikin muuksi löydetään tilassa olevia muita micro:bit-laitteita, jotka ovat noin 10 metrin päässä. Hän, joka saa ensimmäisenä viiden muun henkilön nimen selville sovelluksen avulla, huutaa BINGO! Kaikki löydetyt kuusi osallistujaa nousevat ylös ja heille taputetaan raivokkaasti!

Valmis- Done (10 min)
Ohjelmakoodin testaaminen – suoritetaan ja arvioidaan sovelluksen toimintaa: Ladataan sovellus omaan työkansioon sekä MICROBIT-kansioon (micro:bit-laitteelle). Sovelluksen dokumentointi tapahtuu kommentoimalla omaa ohjelmaa tarkoituksenmukaisesti ja tallennetaan hex-muodossa omalle koneelle.

Tuotoksen katselmointi-review (5 min)
Kukin pohtii lyhyesti ääneen vierustoverille/ryhmälle syntynyttä tuotosta: mitä se tekee nyt ja mitä voitaisiin vielä mahdollisesti kehittää tai parantaa.

Toiminnan arviointi (5 min)
Kukin kuvailee ääneen omaa toimintaansa vierustoverille/ryhmälle: kerrotaan siis mikä meni hyvin ja mitä voisi parantaa seuraavalla kerralla.

Xtra-lisätehtävä jos aikaa jää: Tutki mikä on lämpötila ja lähetä tieto siitä parillesi, kun hänen micro:bit-tietokoneensa pyytää sitä!



Kuva 12: Näytetään lämpötila

Pohtikaa yhdessä minkälaisia uusia sovelluksia tämä osaaminen mahdollistaa! Esimerkiksi kahdella micro:bit-laitteella voi tarkkailla akvaarion lämpötilaa – vai voiko?

1.3 Kolmas KOODIOPPI1 kaksoistunti

Aihe-esittely-Story (max 15 min)

Micro:bit-koodieditorissa on suuri määrä erilaisia ohjelmoinnnin aputyökaluja. Näiden läpikäynti tässä ei ole tarkoituksenmukaista ja se jätetäänkin kunkin osallistujan oman kiinnostuksen varaan. Lyhyesti todetaan että aputyökalut voidaan jakaa kahteen osaan: perustyökaluihin ja lisätyökaluihin.

Työkalujen tapahtumia, toimintoja ja tietoja saa näkyviin klikkaamalla kuvaketta ja valitsemalla kiinnostava kohde napauttamalla sitä taas hiirellä. Kohde putoaa editoriin näkyviin. Se voidaan poistaa helposti siirtämällä se hiirellä takaisin työkalulaatikkoon.

Työkalulaatikot	
Perustyökalut	Lisätyökaluja
Basic Input Music Led Radio Loops Logic Variables Math	Functions Arrays Text Game Images Pins Serial Control

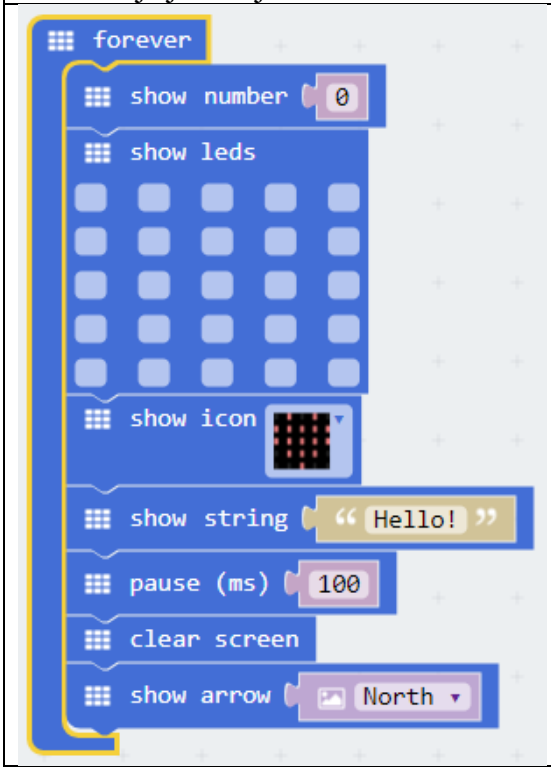
Basic Kuvakkeen alta avautuvia työkaluja ja niiden mahdollisuuksia tutkitaan yhdessä. Yksityiskohtainen opastus löytyy osoitteesta https://makecode.microbit.org/reference/basic.

BlockEditor	JavaScript
Tapahtumia	

on start
forever
	// on start – tapahtuu laitteen käynnistyessä
	
	// suoritetaan toistuva ohjelma(koodi)
	basic.forever(() => {
	
	})

Johdanto

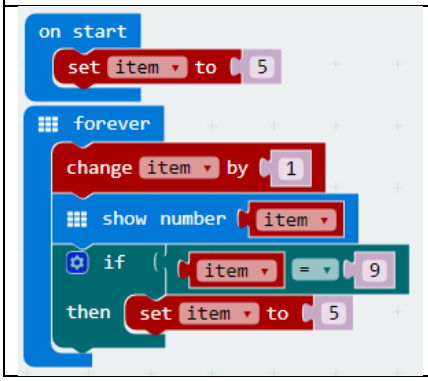
Aluksi kerrataan ja tutustaan lähemmin kahteen tapahtumaan: ”on start” ja ”forever”. Nimensä mukaisesti ”on start”-tapahtuma aktivoi sen sisällä olevat ohjelmalauseet eli toiminnot, kun järjestelmä käynnistyy.

Toimintoja ja tietoja	
	basic.forever(() => { basic.showNumber(0) basic.showLeds(` . . . . . . . . . . . . . . . . . . . . . . . . . `) basic.showIcon(IconNames.Heart) basic.showString("Hello!") basic.pause(100) basic.clearScreen() basic.showArrow(ArrowNames.North) })

”forever”-tapahtuma vastaavasti pitää sen sisälle asetetut ohjelmalauseet aktiivisina ”aina”. Tapahtumaa voidaan hallita toiminnoilla eri tavoin. Esimerkiksi tapahtuman suorittaminen voidaan pysäyttää hetkeksi ”pause”-toiminnolla, jossa keskeytyksen pituus on tieto, joka ilmoitetaan millisekunteina (1000 ms = 1 sek).

Tietoja voidaan muuttaa usealla eri tavalla. Annetaan alkuarvo kirjoittamalla se suoraan toimintoon: ”show string”-tapahtuman alkuarvoksi on asetettu ”Hello”. Tietoja voidaan muuttaa myös erilaisten toimintojen eli ohjelmalauseiden avulla.

Soveltamisesimerkki Block Editor	JavaScript

	let item = 0
	item = 5
	basic.forever(() => {
	item += 1
	basic.showNumber(item)
	if (item == 9) {
	item = 5
	}
	})

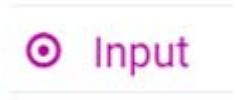
Soveltamisesimerkissä ”on start”-tapahtumassa ”set item to”-toiminto asettaa item-muuttujalle arvoksi numeron 5 (item=5). Seuraavaksi käynnistyvä ”for ever”-tapahtuma toistaa kaikkia sen sisälle asetettuja toimintoja. Ensimmäisellä kerralla change item by-toiminto lisää muuttujaan arvon 1 (item +=1), jolloin sen arvo on 6. ”show number”-toiminto näyttää item-muuttujan arvon. Seuraavaksi if then-toiminto vertaa item-muuttujan arvoa. Mikäli se on saanut arvon 9 niin toiminto muuttaa item-muuttujan arvon takaisin lukuarvoksi 5.

Tietojen antamiseen ja muuttamiseen toimintojen avulla palataan kurssin edetessä useaan kertaan.

TEHTÄVÄ: Mene sivulle https://makecode.microbit.org/reference/basic. Valitse Show icon-esimerkki. Siirry sivun alalaitaan, jossa näkyy seuraavat toiminnat,

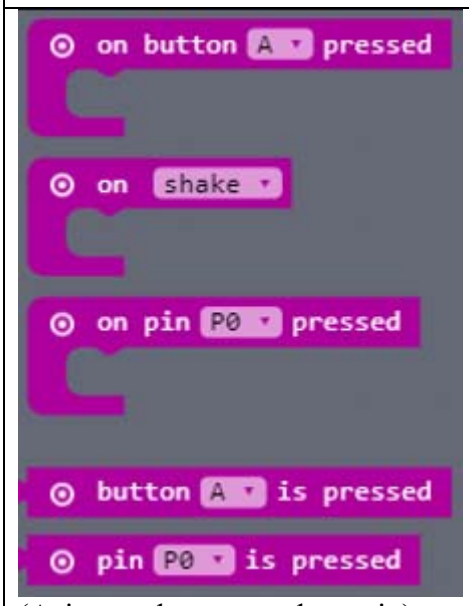
jotka ovat vasemmalta oikealle
Avaa micro:bit editorin;
JavaScript-koodin;
Näyttää micro:bit laitteen;
Lataa ohjelman hex-muodossa;

Paina  jolloin micro:bit editori aukeaa. Vaihda ”on start”-tapahtuma ”forever”-tapahtumaksi. Lisää uusi ikoni ja toteuta nyt nauravan naaman animaatio → ☹😐☺! Lataa syntynyt sovellus omalle laitteelle ja siirrä se MICROBIT-kansioon. Totea sovelluksen toimivuus.



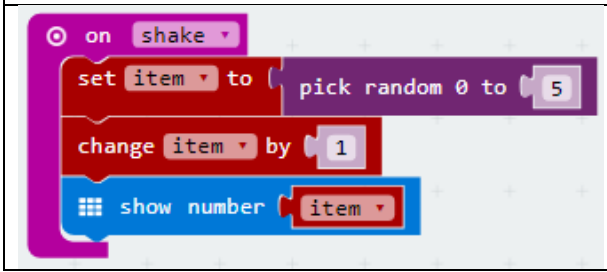
Seuraavaksi tutustutaan lyhyesti tapahtumiin ja tietoihin. Nämä ovat kiinteästi yhteydessä käyttöliittymän käsittelyyn liittyviin toimintoihin sekä micro:bit-käyttöjärjestelmän muihin perusominaisuuksiin. (https://makecode.microbit.org/reference/input)

Käyttöjärjestelmän tapahtumia, toimintoja ja tietoja	
Block Editor	JavaScript

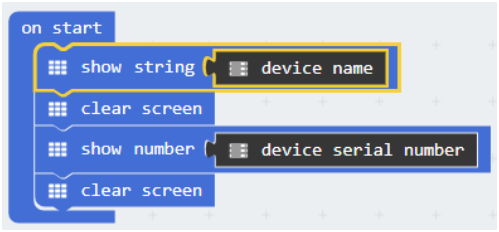
 (Asiaan palataan myöhemmin)	`input.onButtonPressed(Button.A, () => {`
	`})`
	`input.onGesture(Gesture.Shake, () => {`
	`})`
	`input.onPinPressed(TouchPin.P0, () => {`
	`})`
	`input.buttonIsPressed(Button.A)`
	`input.pinIsPressed(TouchPin.P0)`

Tehtävä: Pohditaan yhdessä mitkä ovat tapahtumia, mitkä ovat tietoja ja mitkä ovat toimintoja, joiden avulla asetetaan alkuarvoja. Pohtimista auttaa kuva 13, jossa tapahtuma ”on button pressed B” käynnistää toiminnon ”show number”, joka näyttää numeerisen tiedon temperature-muuttujasta (lämpötila).

Seuraava soveltamisesimerkki näyttää item-muuttujan arvon, joka ensin vaihtelee satunnaisesti välillä 0-5 ja siihen lisätään sitten aina luku 1. Näin saadaan aikaan arpakuutiota muistuttava sovellus, joka näyttää satunnaisesti lukuja välillä 1-6.

Soveltamisesimerkki Block Editor	JavaScript
	input.onGesture(Gesture.Shake, () => { item = Math.random(6) item += 1 basic.showNumber(item) })

Xtra-lisäoppia: Laitteen käynnistyessä ”on start”-tapahtuman aikana näytetään ”show string”-toiminnolla ”device name”-tieto, joka sisältää tiedon laitteen nimestä, käyttöliittymä tyhjennetään ”clear screen”-toiminnolla. ”show number”-toiminto esittää laitteen sarjanumeron. Huomataan että tiedot voivat olla merkkimuotoisia tai numeerisia!

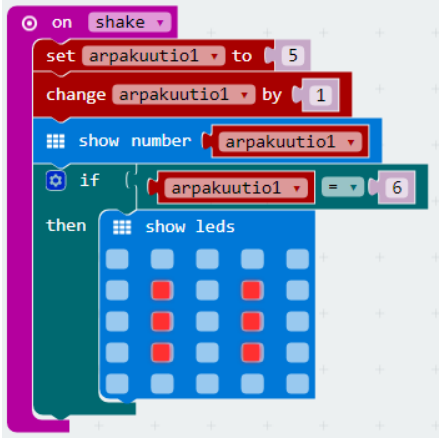


Kuva 13: Tiedot ja toiminnot esimerkki

Tehtävälista-TaskList (max 30 min)

Tarina1: *Haluan sovelluksen (micro:bit-arpakuution), joka näyttää ravistettaessa satunnaisen luvun välillä 1-6 niin kuin arpakuutiossa on tapana.*

Tehtäväksi muodostuu tehtävän ja ratkaisuun liittyvien uusien käsitteiden ymmärtäminen. Tämän sovelluksen ratkaisu löytyy vähitellen. Onneksi ohessa on malliksi prototyyppi ohjelmasta, joka näyttää arpakuutio1-muuttujan arvon ollessa 6. Tätä kuvaa voidaan käyttää apuna tiedonhakujen tuloksia pohdittaessa.

Block Editor	JavaScript
	let arpakuutio1 = 0 input.onGesture(Gesture.Shake, () => { arpakuutio1 = 5 arpakuutio1 += 1 basic.showNumber(arpakuutio1) if (arpakuutio1 == 6) { basic.showLeds(` . . . . . . # . # . . # . # . . # . # . . . . . . `) } })

Tarinan tehtävälista muodostuu pohdinta- ja tiedonhakutehtävistä:

- Miten saadaan aikaan ravistus (”on shake”) tapahtuma?
- Mitä muita mahdollisia tapahtumia löytyy ”on shake”-tapahtuman valinnan lisäksi?
- Miten muodostetaan uusi muuttuja, jonka nimi on arpakuutio1?
- Miten muuttujalle annetaan jokin lukuarvo?

Johdanto

- Miten muuttujan lukuarvo esitetään käyttöliittymässä?
- Miten lukuarvo esitetään arpakuutiossa?
- Miten muodostetaan yksinkertainen if then-ehtorakenne?

Tauko (15 min) – Break the Ice
Työparit ja –ryhmät muodostetaan osallistujista. Parit keskustelevat tehtävälistan sisällöstä

Sprintti (40 min)

VAIHE1
Toteutetaan edellä esitetyt tiedonhakutehtävät ja laaditaan ”Arpakuutiosovelluksen” prototyyppi. Testataan ja todetaan sen toimivuus lataamalla sovellus micro:bit laitteelle. Viimeistellään sovellukset satunnaistettu arpakuutio: Tiedonhakujen ja pohdinnan jälkeen voidaankin käyttää seuraavaa esimerkkiä sovelluksen edelleen kehittämiseksi. Prototyypin ohjelmaa korjataan korvaamalla aikaisemman arpakuutio1-muuttujan arvon asettaminen satunnaisluvuksi, jonka arvo vaihtelee välillä 0-5.

Kuva 14: Satunnaisluvun luominen

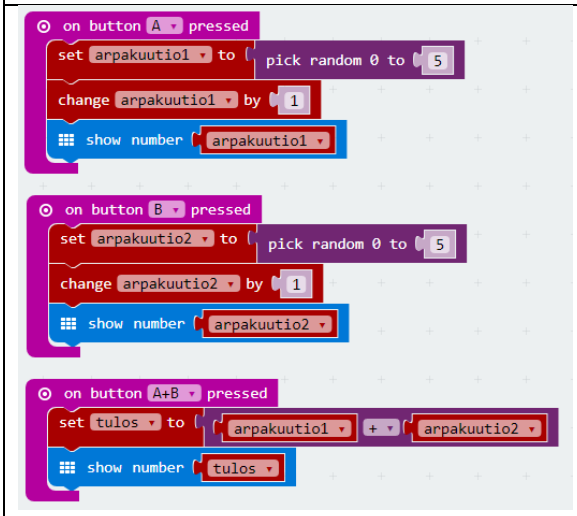
Johdanto

VAIHE2

Tarinaa täydennetään vastaamaan useiden lautapelien noudattamaa ajatusta kahdesta arpakuutiosta.

Tarina2: *Haluan, että nyt käytetään kahta arpakuutiota! Tulos näytetään yhteenlaskun tuloksena. Haluan siis sovelluksen, joka näyttää satunnaisen luvun välillä 1-6 painettaessa A-näppäintä ja samoin kun B-näppäintä painetaan. Painettaessa A ja B näppäimiä yhtä aikaa sovellus laskee arvotut numerot yhteen ja näyttää tuloksen!*

Block Editori	JavaScript

	let tulos = 0
	let arpakuutio2 = 0
	let arpakuutio1 = 0
	input.onButtonPressed(Button.A, () => {
	  arpakuutio1 = Math.random(6)
	  arpakuutio1 += 1
	  basic.showNumber(arpakuutio1)
	})
	input.onButtonPressed(Button.B, () => {
	  arpakuutio2 = Math.random(6)
	  arpakuutio2 += 1
	  basic.showNumber(arpakuutio2)
	})
	input.onButtonPressed(Button.AB, () => {
	  tulos = arpakuutio1 + arpakuutio2
	  basic.showNumber(tulos)
	})

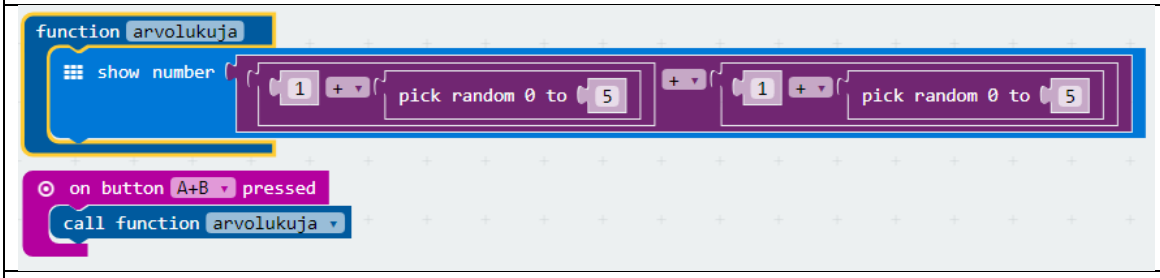
Toteutetaan sovellus annetun vihjeiden ja viimeistellään ohjeen mukaan!

VAIHE 3

Tarina3: *Haluan, että nyt käytetään kahta arpakuutiota! Painettaessa A ja B näppäimiä yhtä aikaa sovellus laskee arvotut numerot yhteen ja näyttää tuloksen!*

Tämän tarina toteuttamiseksi voidaan rakentaa aliohjelma eli funktio, joka suorittaa annetun tehtävän. Tämä edellyttää tiedonhakutehtävän suorittamisen: Miten muodostetaan funktioita, joiden avulla saadaan aikaan lyhyempää ohjelmakoodia? Arvioikaa yhdessä näiden kahden toteutuksen ohjelmakoodin eroja ymmärrettävyyden ja ohjelmien jatkokehittelyn kannalta.

Block Editor



JavaScript

```
function arvolukuja()  {
    basic.showNumber(1 + Math.random(6) + (1 + Math.random(6)))
}
input.onButtonPressed(Button.AB, () => {
    arvolukuja()
})
```

Valmis- Done (10 min)
Ohjelmakoodin testaaminen – suoritetaan ja arvioidaan sovelluksen toimintaa ja ladataan sovellus micro:bit-laitteelle. Esitellään sovellusta muille osallistujille ja pyydetään osallistujat esittelemään oman sovelluksensa. **Pelillistetään sovelluksen testaus:** Parit heittävät arpaa kahdella arpakuutiolla. Voittajat muodostavat aina uudet parit. Uudet parit heittävät taas arpaa ja voittajista muodostetaan taas uudet parit. Kilpailua jatketaan, kunnes kaikkien osallistujien voittaja on selvillä! Voittaja saa osakseen valtavat aplodit!

Sovelluksen dokumentointi: ohjelmakoodia tarkoituksenmukaisesti kommentoimalla.
Sovelluksen tallentaminen: tallennetaan (hex-muodossa) lataamalla se omalle koneelle ja siirtämällä tiedosto omaan työskentelykansioon.

Tuotoksen katselmointi-review (5 min)
Kukin pohtii lyhyesti ääneen vierustoverille/ryhmälle syntynyttä tuotosta: mitä se tekee nyt ja mitä voitaisiin vielä mahdollisesti kehittää tai parantaa.
Toiminnan arviointi-retrospective (5 min)
Kukin kuvailee ääneen omaa toimintaansa vierustoverille/ryhmälle: kerrotaan siis mikä meni hyvin ja mitä voisi parantaa seuraavalla kerralla.

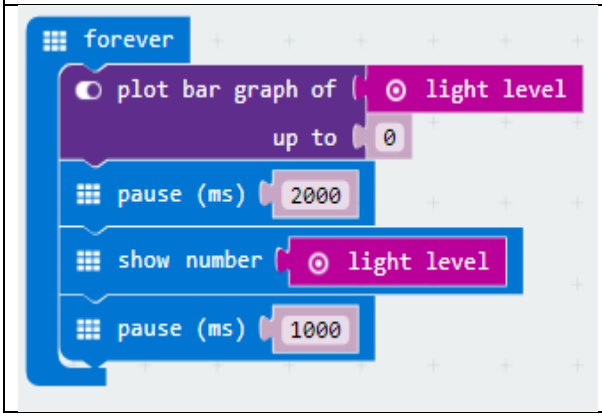
1.4 Neljäs KOODIOPPI1 kaksoistunti

Aihe-esittely-Story (max 15 min)

Input

Tietojen syöttöön ja niiden esittämiseen on Micro:bit-laitteessa on suuri määrä erilaisia toimintoja. Input-näppäimen takaa löytyy mm. antureita, joilla voi tutkia ympäristöä. Alla oleva lyhyt sovellusohjelma näyttää valon määrän lukuarvona 0-255 (pimeästä kirkkaaseen). Led-näppäimen löytyvä ”*plot bar graph of*”-tapahtuman avulla näytetään valoisuutta mittaavan anturin tiedot. Oppimisen kannalta anturin arvo näytetään tässä sovelluksessa myös lukuna (*show number*-tapahtuma). ”*pause*”-tapahtuman avulla hidastetaan tietojen esittämistä!

Block Editor	JavaScript

	`basic.forever(() => {`
	`  led.plotBarGraph(`
	`  input.lightLevel(),`
	`  0`
	`  )`
	`  basic.pause(2000)`
	
	`basic.showNumber(input.lightLevel())`
	`  basic.pause(1000)`
	`})`

Tapahtuma	Selitys	Vaihtoehdot
`on Button A Pressed`	Painiketta A painetaan	A, B, A+B
`on Gesture(Shake)`	Liike(tunnistin) (Shake)	Lukuisia eri vaihtoehtoja
`on Pin Pressed(P0)`	Pin-nappi painettu alas (P0)	P0-P2
`on Pin Released(P0)`	Pin-nappi vapautettu (P0)	P0-P2
`set Accelerometer Range`	Aseta kiihtyvyysmittarin alue	1g-8g
`calibrate Compass`	Kalibroidaan kompassi	
Tietoja		**HUOM.**
`button Is Pressed(Button.A)`	Painiketta A painetaan	käytä ehtolauseessa
`pin Is Pressed(TouchPin.P0)`	Pin P0 on painettu	käytä ehtolauseessa
`acceleration(Dimension.X)`	Kiihtyvyys	näyttää numerona
`light Level`	Valoisuuden taso	0-255
`compass Heading`	Kompassin suunta	Oletus 90 astetta
`temperature`	Lämpötila	Celsiusastetta
`rotation(Pitch)`	Kierto (pitch, roll)	x- tai y-akselilla astetta
`magneticForce`	Magneettinen voima (tesla)	x-,y-,z-suunnata ja tesla
`running Time()`	Käyntiaika ms	millisekuntteina
`running Time Micros()`	Käyntiaika micros	mikrosekuntteina

*) Lähde: https://makecode.microbit.org/reference/input.

Johdanto

Tehtävälista-TaskList (max 30 min)

Tapahtuma tai tieto	Mihin sitä voidaan käyttää
1. Painiketta A painetaan	Käynnitetään jokin toiminto painettaessa A, B, A+B-näppäintä
2. Liike(tunnistin) (Shake)	Käynnnistetään jokin toiminto kun laitetta liikutetaan
3. Pin-nappi painettu alas (P0)	Käynnistetään jokin toiminto
4. Pin-nappi vapautettu (P0)	Käynnistetään jokin toiminto
5. Aseta kiihtyvyysmittarin alue	Kiihtyvyysmittarin herkkyysalueen asettaminen 1g-8g
6. Kalibroidaan kompassi	Asetetaan kompassi käyttökuntoon
Anturit ja niiden tiedot	
7. Painiketta A painetaan	Käytetään ehtolauseessa
8. Pin P0 on painettu	Käytetään ehtolauseessa
9. Kiihtyvyys	Näyttää numerona anturin arvon
10. Valoisuuden taso	Näyttää numerona valoisuuden pimeästä kirkkaaseen 0-255
11. Kompassin suunta	Näyttää kompassin suunnan (kalibroinnin jälkeen) 90 astetta=North
12. Lämpötila	Näyttää lämpötilan Celsiusastetta
13. Kierto (pitch, roll)	Näyttää laitteen asennon x- tai y-akselilla astetta
14. Magneettinen voima (Tesla)	Näyttää magneettisen kappaleen suunnan x-y-z ja magn.voiman
15. Käyntiaika ms	Näyttää millisekuntteja aloituksesta
16. Käyntiaika micros	Näyttää mikrosekuntteina

(https://makecode.microbit.org/reference)

Tarinoita: Seuraavat lauseet esittävät tarinoita, joiden perusteella voidaan ryhtyä toteuttamaan sovelluskehitystä.

1. Mikä on lämpötila tässä huoneessa?
   Mikä on lämpötila lämpöpatterilla (kaksi laitetta)?
   https://makecode.microbit.org/reference/input/temperature
2. Ovatko valot päällä? Menikö joku laatikolle ja avasi sen (kaksi laitetta)?
   https://makecode.microbit.org/examples/plot-light-level
3. Mene pohjoiseen? Minne aurinko laskee?
   https://makecode.microbit.org/projects/compass
4. Oletko noussut jo ylös? Kaaduitko? (kaksi laitetta)
   https://makecode.microbit.org/examples/egg-and-spoon
5. Kumpi on reaktioiltaan nopeampi?
   https://makecode.microbit.org/projects/reaction-time
6. Paljonko kello on?
   https://makecode.microbit.org/projects/watch
7. Missä bitti luuraa?
   https://makecode.microbit.org/projects/hot-or-cold
8. Kivi – Paperi – Sakset
   https://makecode.microbit.org/projects/rock-paper-scissors/code
9. Arvotaan aloittaja (heitetään rahalla: kruuna vai klaava)!
   https://makecode.microbit.org/projects/coin-flipper

Tauko (15 min) – Break the Ice
Muodostetaan parit, jotka valitsevat yhdessä yhden toteutettavan tehtävän/tarinan.

Tehtävä(t) - ToDo (45 min)

Johdanto

Laaditaan / hahmotellaan tehtävä: Suunnitellaan toiminnot ja tiedot. Toteutetaan parityönä. Etsitään toinen pari, joka on toteuttanut jonkin muun tehtävän. Vaihdetaan hex-tiedostoja ja tutustutaan laadittuun lähdekoodiin, testataan toisen parin sovelluksen toimivuus asentamalla se omiin laitteisiin.

Valmis- Done (10 min)

Pelillistetään sovelluksen testaus: Parit arvioivat toisen parin sovelluksen ja sopivat keskenään kumpi esitellään tai jaetaan myös muille osallistujille.

Sovelluksen dokumentointi: ohjelmakoodia tarkoituksenmukaisesti kommentoimalla.

Sovelluksen tallentaminen: tallennetaan (hex-muodossa) lataamalla se omalle koneelle ja siirtämällä tiedosto omaan työskentelykansioon.

Tuotoksen katselmointi-review (5 min)

Kukin pohtii lyhyesti ääneen vierustoverille/ryhmälle syntynyttä tuotosta: mitä se tekee nyt ja mitä voitaisiin vielä mahdollisesti kehittää tai parantaa.

Toiminnan arviointi-retrospective (5 min)

Kukin kuvailee ääneen omaa toimintaansa vierustoverille/ryhmälle: kerrotaan siis mikä meni hyvin ja mitä voisi parantaa seuraavalla kerralla.

1.5 Viides KOODIOPPI1 kaksoistunti

Yhdistetään 1-4 tehtävien osaaminen. Tämä toteutus perustuu ”learning cafee”-oppimiskahvila malliseen toteutukseen. Tässä mallissa kukin pöytäryhmä toteuttaa ja viimeistelee itselleen soveltuvan esimerkkitoteutuksen. Valmiit työt esitellään ensin ryhmässä ja sitten kaikille osallistujille. Saadun palauteen perusteella pohditaan oman ryhmän sovellusta ja koodausosaamisen kehittymistä.

Aihe-esittely-Story (max 15 min)
Muodostetaan pöytäryhmät. Ryhmät valitsevat toteutettavan sovelluksen. Mallina voi olla edellisen kaksoistunnin kehitelmät tai jotain aivan uutta, joka auttaa ryhmää ja sen jäseniä ymmärtämään paremmin koodausta ja siihen liittyviä käsitteitä ☺. Koodaukseen liittyviä peruskäsitteitä ovat esimerkiksi: algoritmit, muuttujat, ehtorakenteet, loopit eli iteraatiot, koordinaattijärjestelmä, vertailuoperaattorit, bitit, tavut ja binäärijärjestelmä, taulukot. Näistä löytyy valtavasti yksityiskohtaista tietoa Internetistä. Tässä joitakin linkkejä liittyen micro:bit-maailmaan **Muuttujat:** Numeerinen muuttuja; Aritmeettiset operaattorit; Vertailuoperaattorit (tosi ja epätosi); https://makecode.microbit.org/types/number. Merkkimuotoinen muuttuja; ASCII-merkit (32-126) https://makecode.microbit.org/types/string. Tosi ja epätosi – vertailun tulos https://makecode.microbit.org/types/boolean. Taulukot, luettelot https://makecode.microbit.org/types/array. **Funktiot:** Valmiit; Omat määritykset https://makecode.microbit.org/types/function.

Tehtävälista-TaskList (max 30 min)
Ryhmät kehittelevät itselleen tehtävälistan ja pohtii ”mitäs vielä pitäisi tehdä sovelluksen esittelemiseksi?” Valitaan pöytäkunnasta nk. kirjuri, joka valmistautuu esittelemään sovelluksen sen valmistuttua. Viimeistellään omaa sovellusta ja vaihdetaan hex-tiedostoja ja tutustutaan laadittuun lähdekoodiin, testataan sovelluksen toimivuus asentamalla se omiin laitteisiin.
Tauko (15 min) – Break the Ice
Tauko pidetään sopivassa kohdassa…
Tehtävä(t) - ToDo (45 min)
Learning café on yhteistoimintamenetelmä, joka soveltuu yli kymmenen hengen ryhmille. Osallistujat jaetaan pöytiin, joista kukin käsittelee samaa tai eri teemaa useinkin eri näkökulmasta. Pöydässä on kirjuri, joka pysyy paikallaan ja muut kulkevat pöydästä toiseen käyden kaikki pöydät läpi ja osallistuen kuhunkin keskusteluun. Kirjuri kuvaa aina uusille tulijoille mitä on kirjattu edellisistä keskusteluista. Usein on järkevää, että osallistujat kirjaavat suoraan pöydällä olevalle (paperiselle) pöytäliinalle tai fläpille syntyneitä huomioita. Tilaisuus on rentoa keskustelua ja pyritään saamaan aikaan kahvilamainen ilmapiiri, vaikka aikataulutus saa aikaan ilmapiirin sähköistystä ja stressin tuntua. Kierros päättyy, kun kaikki pöytäryhmät on käyty läpi (osallistujat ovat kuulleet ajatuksia kaikista pöydistä) ja pöytäryhmän jäsenet ovat palanneet alkuperäiseen pöytäänsä. Aikataulutus kannattaa tehdä seuraavasti: Kirjuri laatii koosteen pöytäkunnan omasta ajattelusta yhdessä muiden kanssa. Tähän varataan aikaa alussa noin 10-15 minuuttia. Kooste saa olla kestoltaan

vain muutamia minuutteja. Ohjaaja ilmoittaa kaikille yhteisesti pöytäkuntien siirtymisistä seuraavaan pöytään (kovalla äänellä). Aina uuden ryhmän tullessa pöydän kirjuri kertoo ja kertaa mitä edellisen ryhmän kanssa on keskusteltu. Pöytäkunnan keskustelu saa kestää enintään viisi minuuttia, jolloin neljän ryhmän kierto kestää 20 minuuttia! Ohjaajan tulee huolehtia aikataulun noudattamisesta. Kun kaikki ryhmät ovat palanneet alkuperäisiin paikkoihinsa, kirjuri esittelee uudet ajatukset ja keskusteluttaa ryhmää näistä. Tähän varataan vielä 5-10 minuuttia. Kahvilan kierto ja uusien ideoiden kehittely kestävät siis kokonaisuudessa 45 minuuttia (15 + 20 + 10).

Valmis- Done (10 min)
Sovelluksen tallentaminen ja jakaminen (hex-muodossa) joko omalle koneelle ja tai jakamalla sen osoite share-toiminnolla ja s-postilla kaikille pöytäryhmäläisille.

Tuotoksen katselmointi-review (5 min) sekä toiminnan arviointi-retrospective (5 min)
Nyt valitaan yhdessä osallistujien joukosta ”KOODIOPPI1-koodari”, jonka valinnan kriteerinä on muiden osallistujien auttaminen ja tukeminen! Tästä annetaan kunniakirja nimetylle henkilölle. Valinta tapahtuu suljettuna lippuäänestyksenä, ohjaaja laskee äänet ja julistaa koodioppi1-koodarin!

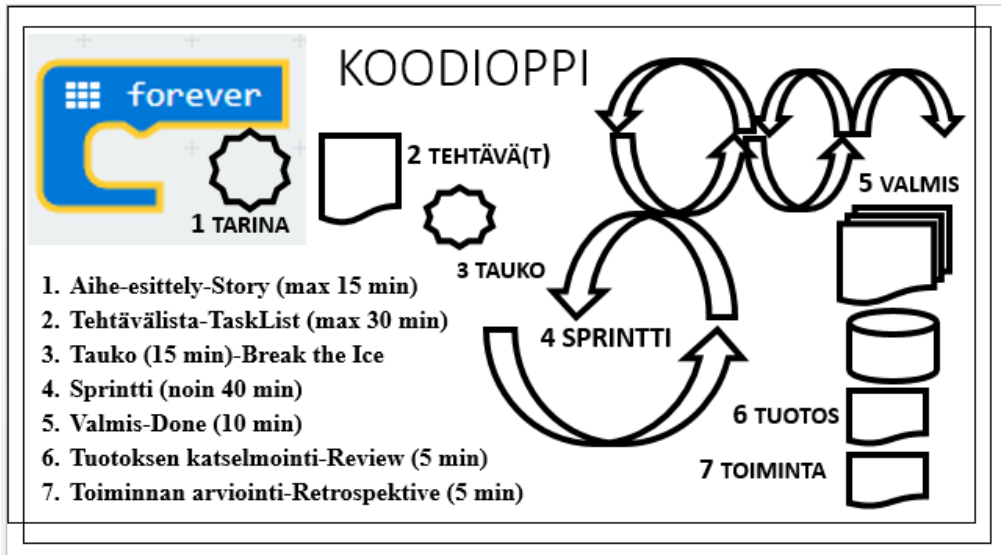
Koodioppi1-peruskurssi on päättynyt.

Ohjaajan ohje

Muistutukseksi vielä, että oppimistapahtumaa vetävän ohjaajan tehtävälista noudattaa seuraavaa 7 vaihetta ja sisältöä:

1. Aihe-esittely-Story (max 15 min)
2. Tehtävälista-TaskList (max 30 min)
3. Tauko (15 min)-Break the Ice
4. Sprintti (noin 40 min)
5. Valmis-Done (10 min)
6. Tuotoksen katselmointi-Review (5 min)
7. Toiminnan arviointi-Retrospektive (5 min)

Ohjaajan tärkein tehtävä on saada osallistujat sopimaan keskenään tehtävistä, tekemään sovitut työt sekä esittelemään tuloksia muille.



Kuva 15: Ohjaajan seinätaulu

2 KOODIOPPI2-jatkokurssit

Tavoitteena on yhteistoiminnallisen projektityöskentelyn oppiminen. Toteutus koostuu useammasta toteutusprojektista. Projektit muodostuvat prototyypin ja siitä kehitetyn tuotantoversion laadinnasta. Prototyypin kehittely kestää kaksi oppimistapahtumaa. Tämän jälkeen kehitetään ”julkaistava toimiva” tuotos. Tämä vaihe koostuu myös kahdesta oppimistapahtumasta. Viimeinen vaihe, viides projektin osa, on varattu ryhmien tuotoksien yhteiseen esittelyyn ja tiedon keskinäiseen tiedon jakamiseen.
Toteutuksen voi valita kahdesta aihealueesta

- Sovellukset micro:bit ympäristössä
- Android-sovellusten tekeminen App Inventor-ympäristössä

KOODIOPPI2 projektien kaksoistuntien rakenne:

- StandUp-kokous (noin 5 min): Kukin työryhmän jäsen kertoo toisille jäsenille mitä on tehnyt projektin tehtävien eteen aikaisemmin ja mitä aikoo tehdä seuraavaksi. Pohjana ovat projektin ”käyttäjätarina” sekä sovitut ToDo-tehtävät.
- ToDo-tehtävät (max 40 min): Oppijat tekevät yhdessä sovittuja ohjelmointitehtäviä parityöskentelynä tai pienryhminä (2-4 osallistujaa). Tämä hetki on yhdessä sovittujen tehtävien tekemistä, uusien todo-listojen muodostamista, jakamista ja käsitteistä keskustelemista ja oman sekä muiden toiminnan pohdintaa oman ryhmän kesken.
- Tauko (15 min)
- Testaus (15 min): ToDo-tehtävät
- Done-tilanne (15 min). Oppijat esittelevät valmiiksi saatuja ohjelmointitehtäviä parityöskentelynä tai pienryhminä (2-4 osallistujaa). Tämän tavoite on oppia jakamaan tietoa ryhmässä, antamaan ja saamaan palautetta sekä kehittää omia oppimistaitoja vertaisten kanssa.
- Dokumentointi (10 min): Oppijat tallentavat tekemänsä työt ja mahdollisesti jakavat tehtyjä tuotoksia toistensa kanssa.
- Arviointi (5 min): Projektiryhmän jäsenet arvioivat lopputulosta ja antavat palautetta omasta ja muiden toiminnasta.

Peruskäsitteitä, joita selvennetään kurssin aikana:

- tietokoneen peruskäyttö (Win 10) ja selain (Chrome)
- kansiot ja hakemistot
- käyttöliittymä, laskenta ja tietovarastot
- algoritmit
- muuttujat
- ehtorakenteet
- iteraatiot - loopit
- koordinaattijärjestelmä
- bitti, tavut ja binäärisyys
- radio/verkko
- taulukot
- projektin toteuttaminen