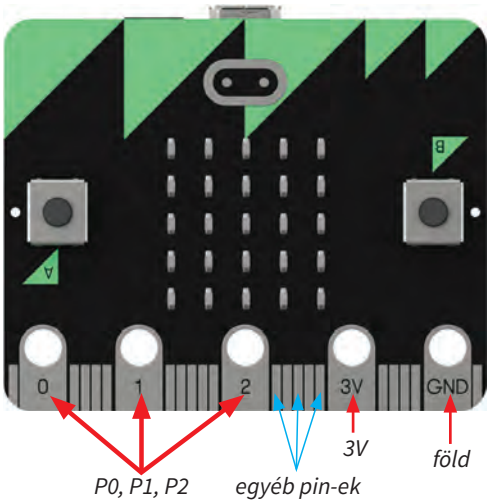
Micro:bit haladó

- Az előző évben megismerkedtünk a BBC micro:bit részeivel, érzékelőivel, blokkos programozásával, 25 LED-es kijelzőjének vezérlésével.
- Az idén a robotika alapjait tekintjük át. Használjuk csatlakozóit, melyeken keresztül külső érzékelők jeleit vihetjük be, például távolságmérő, vagy adatokat vihetünk ki, például külső LED-eket, hangszórót, motorokat vezérelhetünk. Kisautót vezérlünk rádiós kapcsolaton keresztül, és Bluetoothon keresztül létesítünk kapcsolatot mobiltelefonunkkal.
- A blokkos programozás mellett megvizsgáljuk, hogyan néznek ki utasításaink a JavaScript nyelven.

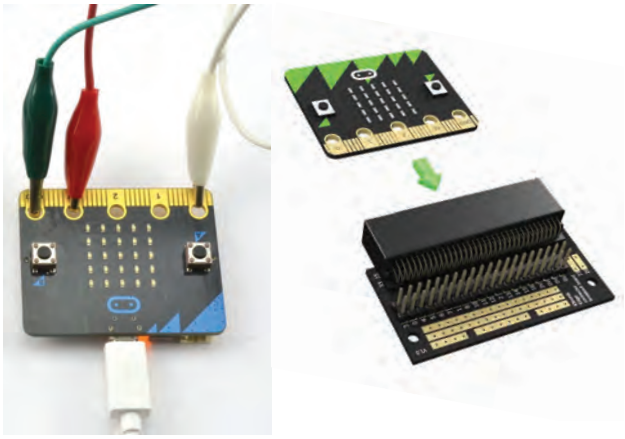
Csatlakozók



A micro:bit 25 csatlakozóval (pin) rendelkezik.

- 3 V pozitív egyenfeszültség és GND (Ground), azaz föld. Ha e kettőre kötünk egy elektromos eszközt, LED-et, motort stb., az folyamatosan világít, illetve működik.
- 3 db analóg/digitális input/output főcsatlakozó: P0, P1, P2. Főleg ezeket használjuk. Ezek programozhatóak. Például a jelzőlámpa háromszínű lámpáit köthetjük rá, és beállíthatjuk, hogy villogjon. Vagy kisautó motorját vezérelhetjük, mikor menjen, mikor ne, milyen sebességgel.
- 20 egyéb, vékony csatlakozó.

Az analóg a folyamatosan változó jel, például a hőmérséklet, a digitális csak néhány konkrét értéket vehet fel, például ég a lámpa vagy nem.



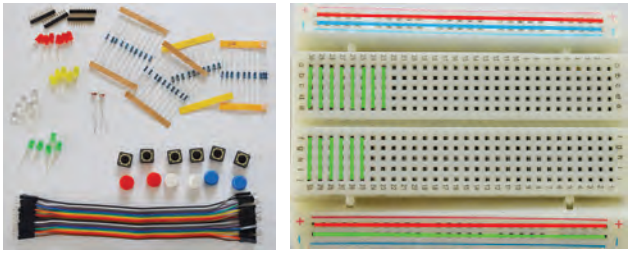
Az első öthöz krokodilcsipesszel, banándugóval, vagy sok egyéb módon csatlakozhatunk. Vigyázzunk, hogy a csipeszek ne érjenek össze, ne érjenek hozzá más csatlakozókhoz. A vékonyakhoz speciális dugó (breakout board) kell.

És már működnek is az eszközeink!

De ha szabályozni akarjuk működésüket, például menjen a kisautó, vagy ki-be kapcsolni egy LED-lámpát, programoznunk kell a micro:bitet. Készítsünk először egy villogót, majd egy közlekedési lámpát!
Ehhez meg kell beszélnünk néhány dolgot elektronikából.

Egy kis elektronika

Mi kell ahhoz, hogy áramköröket összerakjunk?



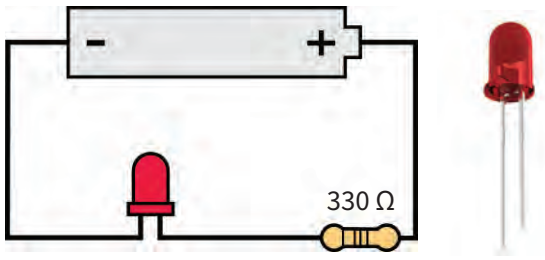
Szükségünk van alkatrészekre: LED-diódák, ellenállások, vezetékek, kapcsolók, érzékelők, motorok stb.

Be kell szereznünk egy próbapanelt (breadboard), amin az áramköröket összeállítjuk. Ez sok lyukat tartalmaz, amibe az alkatrészek lábait lehet bedugni és összekötni másokkal. A lyukak az ábrán látható módon vannak összekötve a próbapanel hátsó oldalán. (Lehet panel nélkül is összeállítani az áramköröket csak bizonytalan lesz a működése, könnyen szétcsúszhat.)

Kell egy feszültségforrás, ami esetünkben a micro:bit. Lehet a 3 V-os kimenet is, de kaphat feszültséget programozhatóan, a csatlakozókból (pinekből) is. Ha zárt az áramkör, az áram a feszültségforrás egyik sarkából (3 V, pin) a másikba (GND vagy föld) folyik, működik a kapcsolásunk. Ha nyitott, akkor nem folyik áram, akkor nem működik.

Ezért kellenek kapcsolók, melyek hol nyitják, hol pedig zárják az áramkört. Így lehet például villogtatni egy lámpát.

Végül tisztában kell lennünk a LED működésével, mint a leggyakoribb áramköri elemmel. A LED (Light-Emitting Diode, azaz fényt kibocsátó dióda) egy félvezetőből készült fényforrás. Nagyon energiatakarékos, egyre inkább átveszi a hagyományos izzólámpák szerepét. 2–5 V-ot igényel a világításhoz. Van egy különlegessége: egyenfeszültséget igényel, és a két lába közül a hosszabbikra kell kötni a pozitív sarkot, a rövidebbet pedig földelni. Fordítva nem világít.

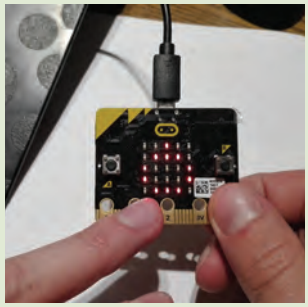


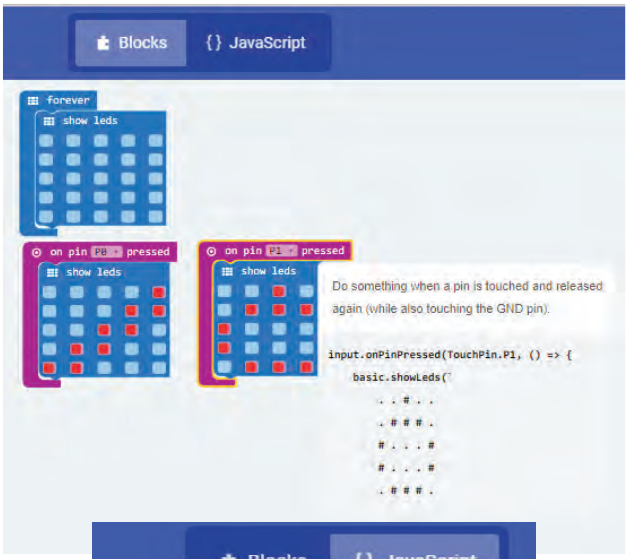
Bekötéséhez szükséges egy ellenállás, amit sorba kell vele kötni. Ugyanis amikor a LED világít, nagyon pici az ellenállása. Kicsi az ellenállása akkor is, ha nyitóirányban van bekötve, így tönkreteheti a micro:bit-et.

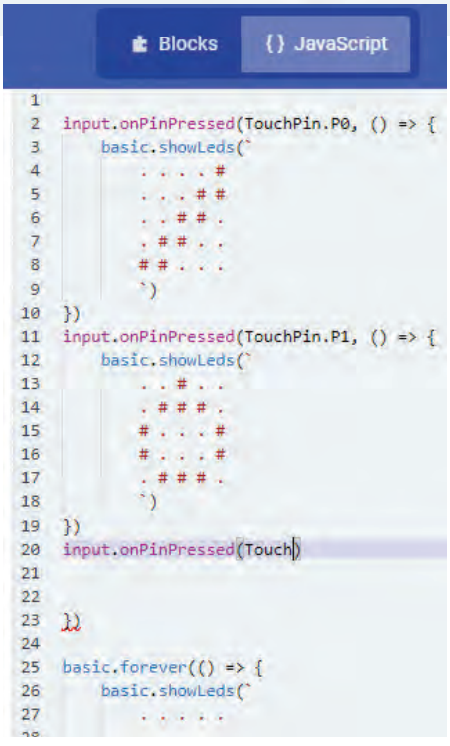
Ennyi ismerettel bátran nekiláthatunk a szereléshez!

Gyümölcsök

Próbáljuk ki a micro:bit csatlakozóit! Írjunk olyan programot, hogy ha megfogjuk a P0-at, rajzoljon ki egy banánt, ha a P1-et, akkor egy almát, ha P2-t, akkor egy körtét!







Közben kezdjünk ismerkedni a *JavaScriptes* programozási felülettel! Ha az egérmutatót rávisszük az *on pin P1 pressed* feltételsorra, kiírja az utasítást *JavaScript* nyelven.
Váltsunk át blokknézetről erre a nyelvre! Ugyanazt a programot látjuk, csak más nyelven. Azt nézzük meg, hogyan írtuk meg eddig a programot, és folytassuk a harmadik gyümölcs beillesztésével, a látott utasítások mintájára!
Ha kész, vissza is válthatunk blokknézetbe. Ellenőrizzük, jól dolgoztunk-e!
Végül próbáljuk ki! Egyik ujjunkat tegyük a GND (föld) pinre, másik ujjunkat pedig valamelyik megadott csatlakozóra. Ilyenkor testünkön keresztül zárul az áramkör, és mutatja az aktuális gyümölcsöt.

Egészítsük ki előző összeállításunkat valódi gyümölcsökkel! Ha megfogjuk őket, azokon keresztül záródjon az áramkör, és a gyümölcsnek megfelelő kép jelenjen meg a LED-kijelzőn!

Megvalósítás: Vegyünk 3 dobozt! Mindegyik tetejéből álljon ki egy vasszög, abba szúrjuk bele a 3 gyümölcsöt! A szögeket kössük rá a pinekre. Tegyünk egy fémszalagot csuklónk alá, ezt kössük a földre (GND). És kész a gyümölcsfelismerő játék!

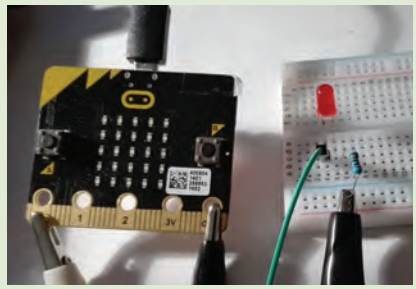
Készíthetünk belőle bűvészmutatványt is! Szükségünk van két micro:bitre, a kettő között pedig rádiós kapcsolat! Az egyiket kapcsolgatjuk a gyümölcsökön keresztül, a másikon pedig, akár száz méter távolságban, jelenjen meg a képe!

A közönség csodálkozik, honnan tudja a kijelző, melyik gyümölcsöt fogtuk meg!

Ki a gyorsabb? Készíts játékot, amellyel lemérheted, neked vagy a társadnak rövidebb a reakcióideje! Véletlenszerű időpontban jelenjen meg egy kép a kijelzőn. Aki gyorsabban nyomja meg a gombját, az nyer, az ő jele jelenjen meg a kijelzőn!

Villogás

Készítsünk most villogót! A P0 pint programozzuk be úgy, hogy egyszer kapjon feszültséget a LED, másodszor ne!



```
basic.forever(function () {
    pins.digitalWritePin(DigitalPin.P0, 1)
    basic.pause(500)
    pins.digitalWritePin(DigitalPin.P0, 0)
    basic.pause(500)
})
```

Az utasítás, amivel feszültséget adunk a P0 pinre: *digital write pin P0 to 1*.
Erre felgyullad a LED, vár fél másodpercet.

Most oltsuk le. Írjunk 0-t a P0-ra, de azt már JavaScriptben:
pins.digitalWritePin(DigitalPin.P0, 0)
basic.pause(500)
és várjunk fél másodpercet!

```
basic.forever(() => {
    pins.digitalWritePin(DigitalPin.P0, 1)
    basic.pause(500)

})

```

Töltsük fel a micro:bitre, és már villog is!

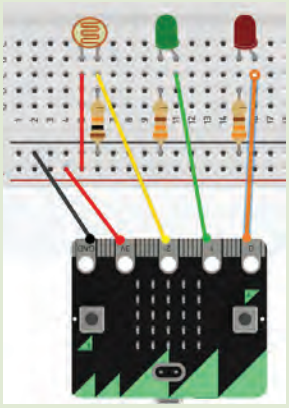
Nem kell külön ciklusba tennünk, hogy folyamatosan villogjon, mert a *forever* pont ezt csinálja.

Készíts piros-kék villogót, ami felváltva villog!
Gyorsítsd vagy lassítsd a villogást a várakozási idő megváltoztatásával!
Készíts jelzőlámpát! Ha van 3D nyomtatód, nyomtasd ki, és tedd bele a LED-eket! (A különböző színű LED-eket külön pinekre kell tenned, és ne felejtsd el a védőellenállásokat is bekötni az áramkörbe!)

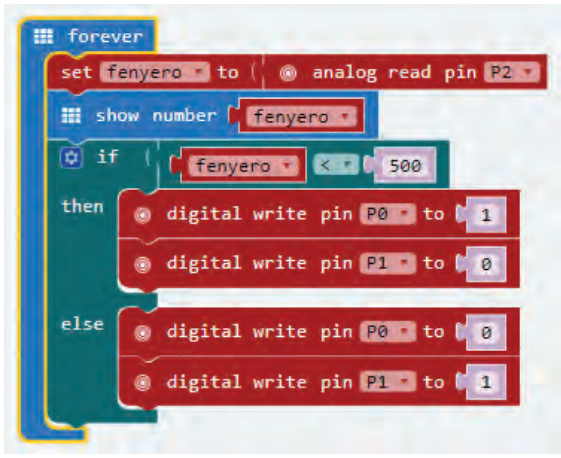
Fényerősségmérés

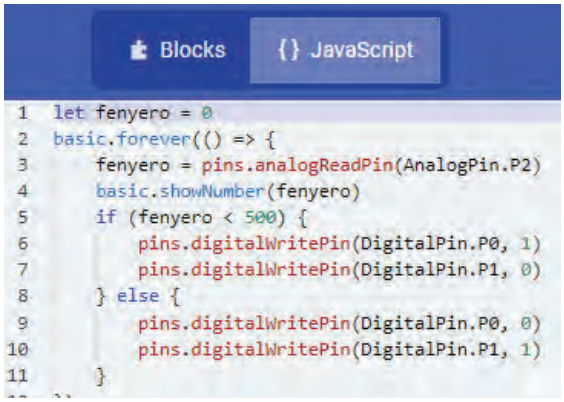
Takarékoskodjunk az energiával, csak akkor világítsanak az utcai lámpák, ha sötét van!
Nézzük meg, hogyan használhatunk analóg érzékelőket, mint például a fényerősségmérőt!

Készítsünk egy olyan mérőműszert, ami a kis fényerőt pirossal jelzi, a nagy fényerőt zölddel! Ehhez szükségünk lesz egy fényérzékeny ellenállásra. Egy 10 kΩ-os ellenállással sorba kötve tegyük be az áramkörbe! Egyik lábát kössük be a P2 pinre. A piros LED-et a P0-ra, a zöldet a P1-re tegyük.
Ezután írjuk meg a programot!
Hozzunk létre egy változót, amibe beírjuk *(set fenyero to)* a szenzorról beolvasott értéket *(analog read pin P2)*.



Mivel nem tudjuk, mekkora a fényerő értéke, melynek alapján eldöntjük, melyik LED égjen, írassuk ki *(show number)* a beolvasott értéket, vagyis a *fenyero* változó értékét!
Változtassuk a fényerőt, takarjuk le a szenzort, határozzuk meg a határértéket, ami már sötétnek számít, és ezt írjuk be az elágazás feltételébe!
Nézzük meg, hogyan néz ki programunk JavaScriptben is!





Figyeljük meg, hol, milyen zárójeleket kell használni!

Az elágazás feltétele kerek zárójelbe kerül, az igaz és hamis esethez tartozó utasítások kapcsos zárójelbe kerülnek.

Kiegészítés

A fényerősség-változást használják ki a fénykapuk. Amikor elhaladunk az érzékelő előtt, kinyit egy ajtót.
Két fotocellás érzékelővel már a helyiségben tartózkodók létszámát is tudjuk követni. Ha az első érzékelő előtt halad el először, akkor belép, ha a második előtt, akkor kifelé megy. A belépőkkel növeljük a számlálót, a kilépőkkel csökkentsük. Azt is beépíthetjük a programba, hogy ha mindenki kijött, oltsa le a villanyt a szobában. Ez sok esetben jobb, mint a mozgásérzékelő.
Hasonló analóg jelet ad a távolságmérő szenzor, ami a tolatóradar működésének alapja. Ha túl kicsi a távolság, sípol.
A hőmérsékletmérés is analóg. Ezt használják ki a klímaberendezések. Ha a hőfok egy bizonyos

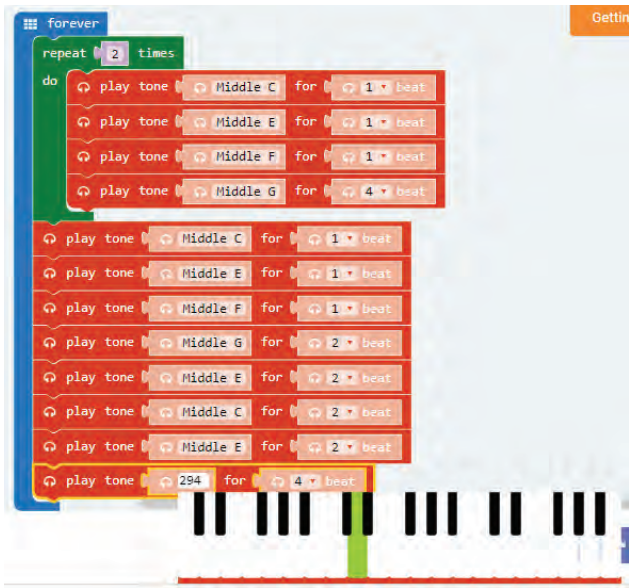
szint alá süllyed, bekapcsol a fűtés. Addig melegít, amíg el nem éri a kívánatosat, aztán lekapcsol.

Írjuk meg az energiatakarékos utcai lámpa vezérlőprogramját! Egy LED legyen, ami egy bizonyos fényerősség alatt bekapcsol, különben nem ég.

Zene

Egy kis pihentető, egy kis művészet. Mert zenélni is tud a micro:bit!

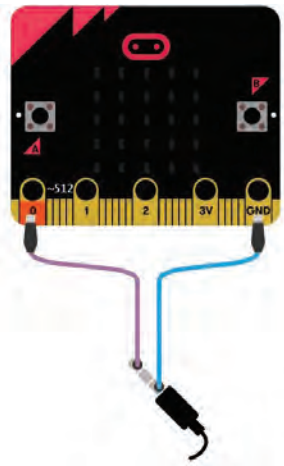
A *Music* (zene) menüben találhatjuk a használható utasításokat. A *start melody* paranccsal kész dallamot játszhatunk le, például a *Happy birthday*-t. De kézzel is beírhatunk egy zenét, akár saját szerzeményt is. Úgy adhatjuk meg a kottát, mintha zongoráznánk. A *play tone* utasítás meghatározza melyik hangot mennyi ideig játssza le. Az ismétlésekhez használjunk ciklust!



Blocks | JavaScript

```
basic.forever(() => {
    for (let i = 0; i < 2; i++) {
        music.playTone(262, music.beat(BeatFraction.Whole))
        music.playTone(330, music.beat(BeatFraction.Whole))
        music.playTone(349, music.beat(BeatFraction.Whole))
        music.playTone(392, music.beat(BeatFraction.Breve))
    }
    music.playTone(262, music.beat(BeatFraction.Whole))
    music.playTone(330, music.beat(BeatFraction.Whole))
    music.playTone(349, music.beat(BeatFraction.Whole))
    music.playTone(392, music.beat(BeatFraction.Double))
    music.playTone(330, music.beat(BeatFraction.Double))
    music.playTone(262, music.beat(BeatFraction.Double))
    music.playTone(330, music.beat(BeatFraction.Double))
    music.playTone(294, music.beat(BeatFraction.Breve))
})
```

JavaScriptben a frekvenciákat kell megadni. Figyeljük meg, hogyan lehet megadni egy számlálóciklus, hányszor ismételjen!



Mivel a micro:bitben nincs hangszóró, ezért a fejhallgató jackdugóját két krokodilcsipeszes vezetékkel csatlakoztathatjuk a lapkához, az ábrán látható módon.

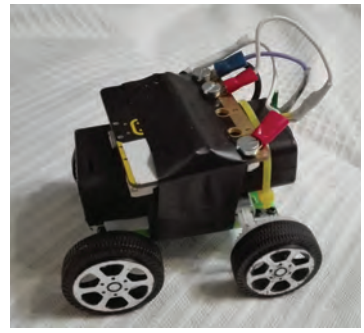
Kiegészítés

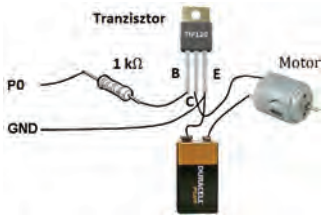
Sokkal nagyobb lehetőségekkel rendelkezünk hangok és zenék előállítására, ha a Python nyelvet használjuk. Például képes géphanggal felolvasni egy beírt szöveget: *speech.say("Hello microbit")* utasítással, természetesen angol kiejtéssel. Így már beszélhet is kis robotunk, amit építettünk. Rádiós kapcsolattal pedig kívülről is irányíthatjuk, mit mondjon...

Motorok

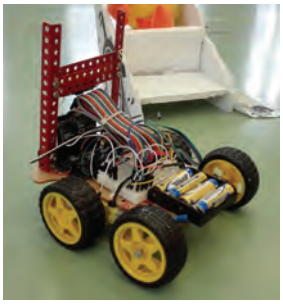
Elérkeztünk a mozgatásért felelős motorok működtetéséhez. Autókat hajtunk meg, ventilátort használunk, ha meleg van, ajtókat nyitunk. Csak villanymotorokkal foglalkozunk! Az első forog gyorsabban, lassabban, folyamatosan, a második csak bizonyos szöggel (0–180°) fordul el, csupán bizonyos helyzetekben áll meg.

Ha programozható kisautót szeretnénk készíteni, elég egy olyan kisautót átalakítani, amelyik megy, ha bekapcsoljuk, és megáll, ha kikapcsoljuk. Ha egy micro:bittel kapcsolgatjuk, távirányítós autó lesz belőle.





A megépítéshez szükségünk lesz még egy kis elektronikára, hiszen a micro:bit elég gyenge áramot ad, a motor pedig erőset igényel. Ezt úgy oldhatjuk meg, hogy egy megfelelő teljesítményű tápegységre kötjük az autót, amit a micro:bit kapcsolgat. Ez lehet egy motordriver vagy egy tranzisztor is.



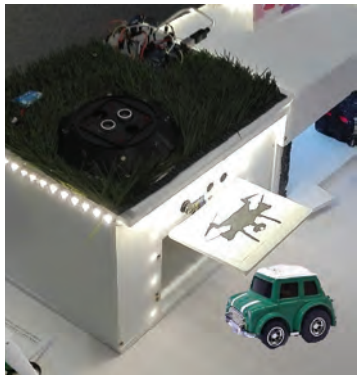
A tranzisztor olyan eszköz, amivel gyengeáramú jellel erősáramú áramkört lehet kapcsolgatni.

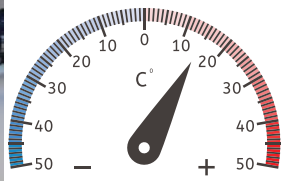
A tranzisztor (TIP120) bázis (B) lábán keresztül szabályozzuk, hogy mit csináljon a motor. A kollektor (C) lábán kapja a motor az erős jelet. Az emitter (E) lábát pedig leföldeljük a tápegység és a micro:bit GDN csatlakozójára. (Ezt a kapcsolást sok, nagyobb feszültséggel működő áramkörben használhatjuk.)
Ha a kisautó első két kerekét külön hajtjuk meg, külön motorokkal, be is tud fordulni.
Irányításhoz a rádiós kapcsolatban levő másik micro:bit orientációs szenzorát használjuk, melynek a mozgatásával tudjuk vezetni az autót.
A sebességét akkor tudjuk változtatni, ha analóg kimenetről hajtjuk meg. Ezzel 0 és 1023 közötti értékkel lehet finoman változtatni.

Szervomotor – kiegészítés

A másik nagy motorcsaládba, a szervomotorok tartoznak. Ezeket konkrét pozíciókba lehet beállítani. Például egy garázs ajtónyitójánál, egyik helyzet a zárt, másik a rá merőleges, nyílt állás.

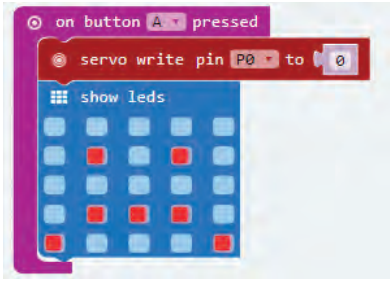
A szervo pedig egy negyed kört tesz meg, míg kinyitja.

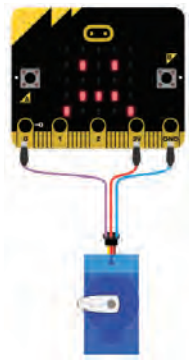




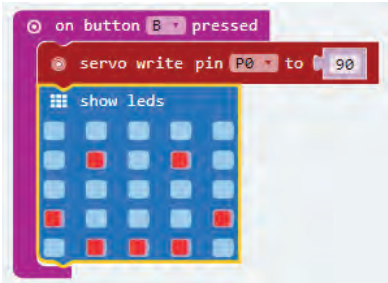
Vagy egy mérőműszer mutatója, ami a mért értékkel arányosan fordul el.

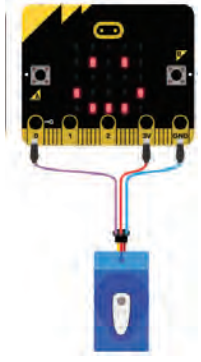
Írjunk programot az ajtónyitóra! Az A és B gomb nyomására nyisson illetve zárjon.





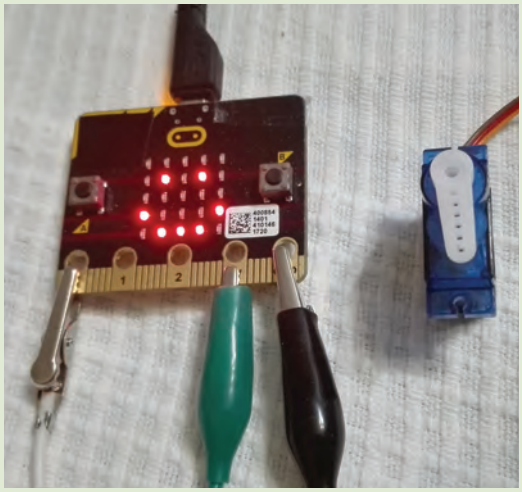
A szervót működtető parancsot a *Pinek* (Pins) menüben találjuk. A *servo write pin P0 to 0* a P0 kimenetre kössük a vezérlő vezetéket, GND és 3 V-re a feszültséget. Értéknek azt a fokot adjuk meg, ahány fokra álljon be 0° és 180° között.





Egy kis képecskével jelezhetjük, melyik az aktuális.
Ha gyengén működik, kössük 5 V-ra!

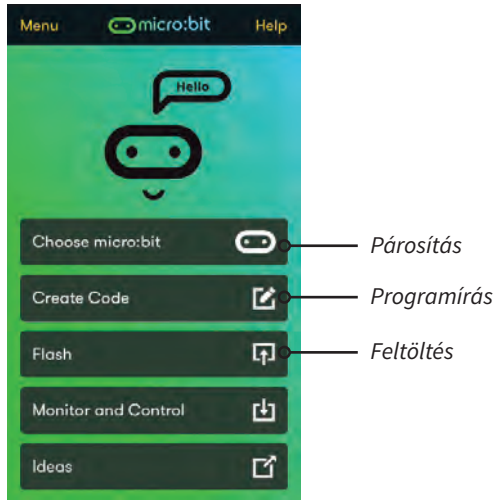
Készíts hőmérőt!
A 10 °C és 40 °C közötti értéket mutassa 90°-kal elfordulva!
Érzékelőnek a micro:bit hőmérőszenzorát használhatjuk.
Készíthetsz zöld android-figurát hungarocellből, ami a fejét forgatja egy szervomotorral.
Készíts kis házikót, amelynek ajtaját nyitogathatod a szervomotorral! Néhány szervomotorral robotkart építhetsz fémépítőből!



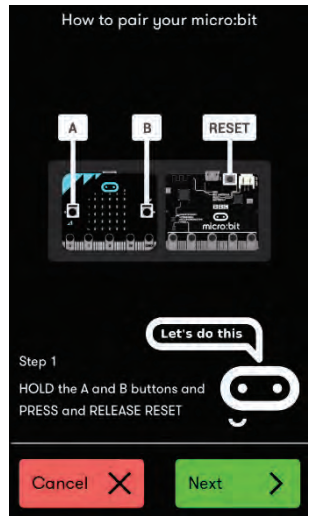
Mobilkapcsolat

- Szeretnéd programozni a micro:bitedet, de nincs nálad a laptopod? Nincs, ahol megírd és feltöltsd a programot? Van egy jó hírem! Nálad van a mobilod? Készítsd el azzal!
- A böngészővel lehet futtatni a microbit.orgon a blokkos felületet. Írd meg a programot, és töltsd le a .HEX fájlt a mobilodra.

Ezt kell áttöltened a micro:bitre.
Mivel a lapka kezeli a Bluetooth kapcsolatot, ezért csak párosítani kell a két készüléket, hogy kommunikálni tudjanak egymással.

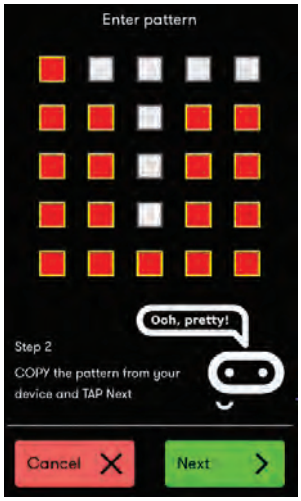


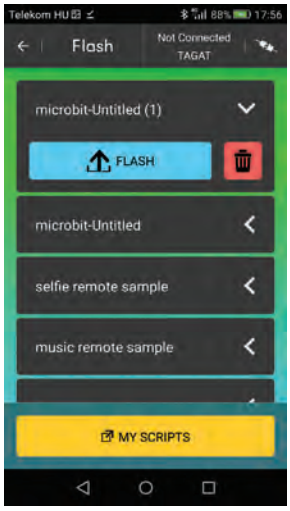
A kapcsolat létrehozásához először le kell tölteni a micro:bit applikációt a telefonra. Ezt elindítva lépésről lépésre mutatja az app, hogyan kell párosítást elvégezni.



A micro:bit ad egy kódot egy mintán keresztül, ezt kell átmásolni a mobilon megjelenő minta helyére.
Ha sikerült létrehozni a kapcsolatot, következik a feltöltés *(Flash)*. Megjelennek a letöltött .hex fájlok. Ki kell választani, melyiket szeretnénk rátölteni a lapkára, és már indul is az áttöltés.
Ezzel kész is vagyunk, fut a programunk a micro:biten.

Persze árammal is el kellett látnunk ehhez a lapkát, például 2 × 1,5 V-os elemekkel. Jó programozást akár utazás közben is!

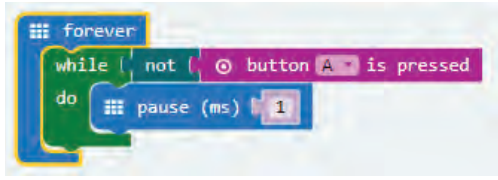




Stopper

Sok játékban egymással vagy az idővel versenyzünk. Pl. egy drónügyességi versenyen, egy feladatot kell végrehajtani minél gyorsabban. Ilyenkor jól jön egy stopperóra! Készítsünk ilyet micro:bitből!

Ehhez azt kell tudni, hogy amikor használjuk a mikrovezérlőnket, fut benne egy óra, melyet bármikor lekérdezhetünk. A lekérdezést végző utasítás: a „runningtime (ms)”. Ez ezredmásodpercben adja meg a bekapcsolástól eltelt időt. Nekünk viszont két esemény közti időt kell mérni, ezért egy véletlenszerűen megjelenő számjli kirajzolásakor kérdezzük le az órát, majd az A gomb lenyomásakor ismét. Ezt a két adatot mentsük el egy-egy változóba, mert a mért időnk ennek a kettőnek a különbsége lesz.



Időméréshez használjunk egy ciklust, ami addig ismétlődik, amíg az A gomb lenyomásával ki nem lépünk belőle.

Tedd felhasználóbaráttá a programodat! Figyelmeztessen, hogy mindjárt kezdődik a mérés, figyelemfelkeltő legyen, ha már fut az időmérés.

Ha hosszabb időt mérsz, kerekítsd az időt másodpercre!

Egy lehetséges megvalósítást láthatsz itt.