App Inventor Faragó Csaba

http://faragocsaba.hu/app-inventor

Az App Inventor segítségével applikációkat lehet készíteni Androidos okostelefonokra. Néhány fontosabb oldal:

- http://appinventor.mit.edu - ez az App Inventor főoldala
- http://ai2.appinventor.mit.edu - itt tudunk alkalmazásokat készíteni
- http://appinventor.mit.edu/explore/ai2/tutorials.html - App Inventor példák
- http://www.appinventor.org - részletesen kidolgozott App Inventor tananyagok

Ez a sorozat elsősorban az alap App Inventor tutorial alkalmazásaira épít. Mielőtt elkezdenénk, olvassuk el ezt, mivel az App Inventor használata valamelyest bonyolultabb mint a Scratch-é: App Inventor beállítások. A program az alábbiakat tartalmazza:

- Beszélj hozzám
- Minigolf
- Rajzasztal
- Hangyanyomkodó
- Űrhajó
- Térképes kvíz
- Kapcsolat a robot és az okostelefon között

App Inventor - Beszélj hozzám!

Az első alkalmazásunk a beszélj hozzám, melynek angol nyelvű útmutatói itt találhatóak:

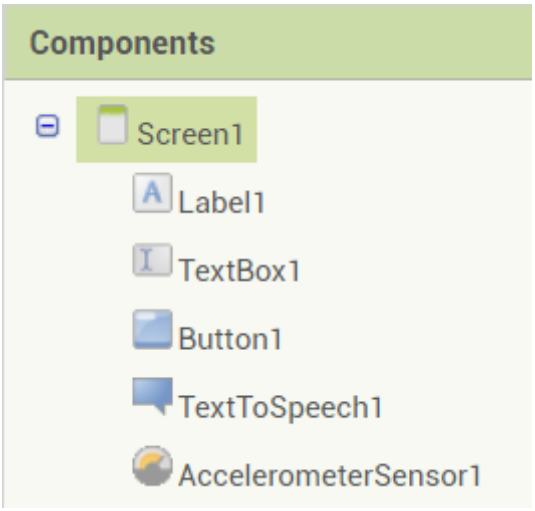
- https://www.youtube.com/embed/Vdo8UdkgDD8
- https://www.youtube.com/embed/0hikoCvM3oc
- http://appinventor.mit.edu/explore/sites/all/files/hourofcode/TalkToMePart1_2perpage.pdf
- http://appinventor.mit.edu/explore/sites/all/files/hourofcode/TalkToMePart2_2perpage.pdf

Mindenkinél állítsuk be az alkalmazást. Ehhez az alábbi leírás nyújt segítséget: App Inventor beállítások.

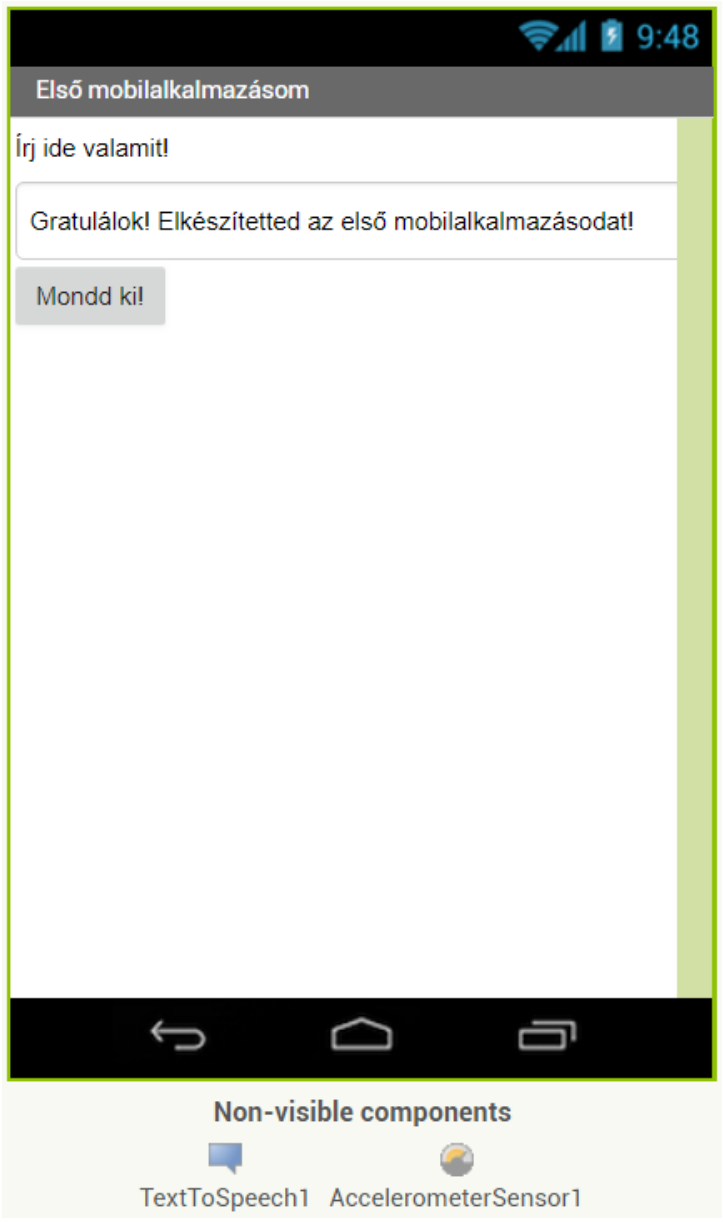
Kezdjük el a projekt készítését: Projects → Start new project. Az App Inventor két fő részből áll: a Designer, ahol a képernyőképet lehet alakítani, és a Blocks, ahova a program kerül. Designer módban a bal oldali palettáról drag and drop módszerrel dobjuk rá a nézőben:

- User Interface → Label. A tulajdonságainál (Properties) alul a szöveget (Text) írjuk át erre (idézőjelek nélkül): „Írj ide valamit!”
- User Interface → TextBox. Text: „Gratulálok! Elkészítetted az első mobilalkalmazásodat!”
- User Interface → Button. Text: „Mondd ki!”
- Media → TextToSpeech. Ez egy úgynevezett nem látható komponens, a nézőke alatt jelenik meg.
- Sensors → AccelerometerSensor. Ez szintén nem látható komponens.
- Az alapból ott található Screen1 komponens címét (Title) írjuk át erre: „Első mobilalkalmazásom”.

Ha mindent jól csináltunk, a komponens nézet így néz ki:



A későbbiekben, amikor már ennél több komponens lesz, nevüket át fogjuk írni; most az egyszerűség kedvéért meghagyjuk az alapértelmezettet.
A megjelenítőben ezt látjuk:

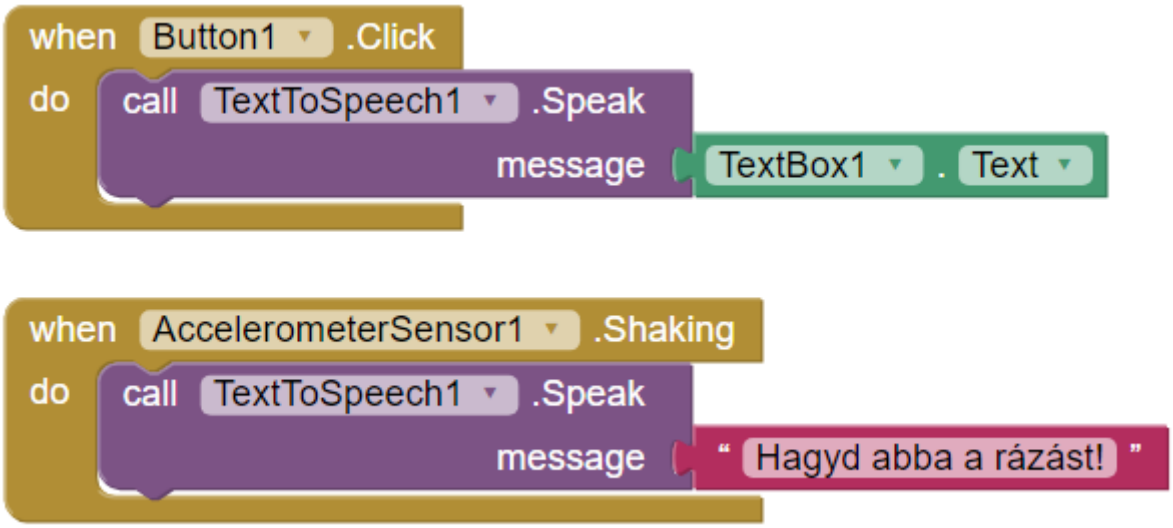


Most a jobb felső sarokban váltsunk át blokk nézetre (Blocks):

Itt tegyük a következőt:

- Ezt válasszuk ki: baloldalon Screen1 → Button1 → when Button1.Click do (legfelső).
- Ezt tegyük a már képernyőn levő struktúra belsejébe: Screen1 → TextToSpeech1 → call TextToSpeech1.Speak message.
- A message-hez illesszük ezt: Screen1 → TextBox1 → TextBox1.Text (nem a set TextBox1-et!)
- Screen1 → AccelerometerSensor1 → when AccelerometerSensor1.Shaking do.
- Ezen belül call TextToSpeech1 Speak message.
- Az üzenet (message) legyen Built-in → Text → itt válasszuk ki a legfelső elemet (A text string), és oda írjuk bele ezt: „Hagyd abba a rázást!”

A végeredmény az alábbi:



Próbáljuk ki a programot: írjunk be valamit a szövegmezőbe, kattintsunk a nyomógombra, ill. rázzuk meg a mobiltelefont.

Végeredmény: Beszeljhozzam.aia.

App Inventor - Minigolf

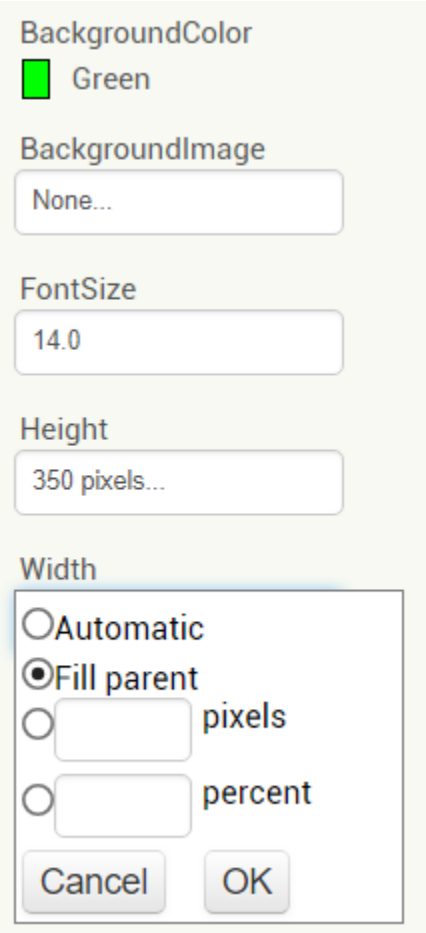
Ennek a játéknak a célja a rajzvászon (Canvas) bemutatása. Mozgatható szereplőket csak a rajzvászonra helyezhetjük el. A Scratch-et egyébként úgy is felfoghatjuk, mint egy rajzvászon; az App Inventor ilyen értelemben többet tud, mivel a rajzvásznon kívül is tudunk komponenseket elhelyezni. Viszont az App Inventor szereplőválasztáka igen vérszegény: mindössze egy labdát (pontosabban fogalmazva: egy kitöltött kört) lehet kiválasztani, ill. egy általános kép szereplőt (ImageSprite), melyhez feltölthetünk mi magunk képet, ráadásul igen sok hasznos funkció is hiányzik.

Angol nyelvű útmutatók:

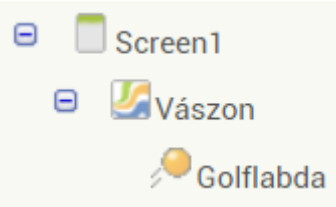
- https://www.youtube.com/embed/w0yxJSlC00w
- http://appinventor.mit.edu/explore/sites/all/files/hourofcode/BallBounceTutorial_2perpage.pdf
- http://appinventor.mit.edu/explore/ai2/minigolf.html

Első lépés: golflabda. Design:

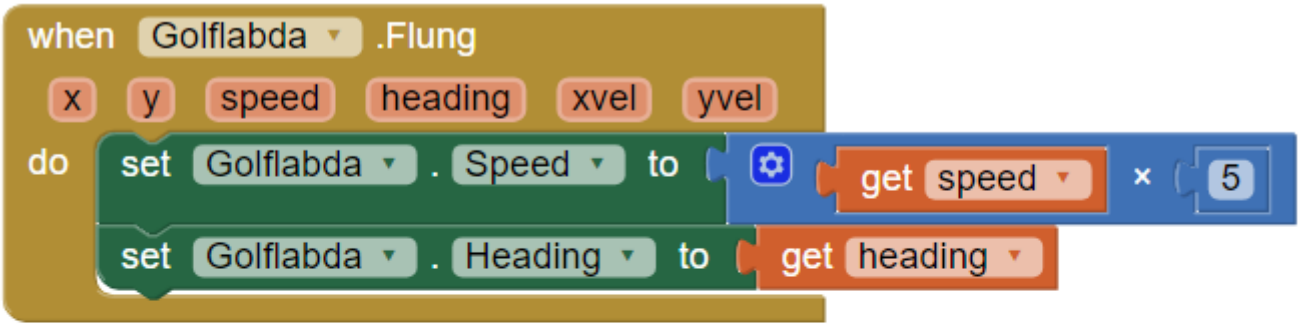
- Screen1
  - Title (cím): Minigolf
  - ScreenOrientation (képernyő irány): Portrait (portré)
- Drawing and Animation → Canvas
  - Név (Rename): Vászon
  - BackgroundColor (háttérszín): zöld
  - Height (magasság): 350 pixels (képpont)
  - Width (szélesség): Fill parent (szülő kitöltése)

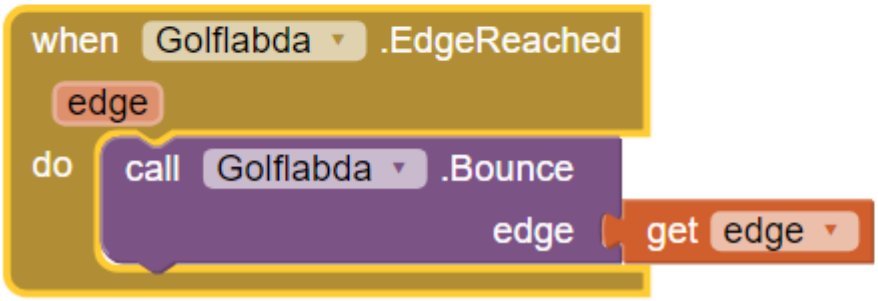


- Drawing and Animation → Ball: dobjuk rá a rajzvászonra
  - Név: Golflabda
  - Radius (átmérő): 10
  - PaintColor (kitöltés színe): fehér
  - X: 150
  - Y: 300



Blokk nézetben valósítsuk meg a következő programot:



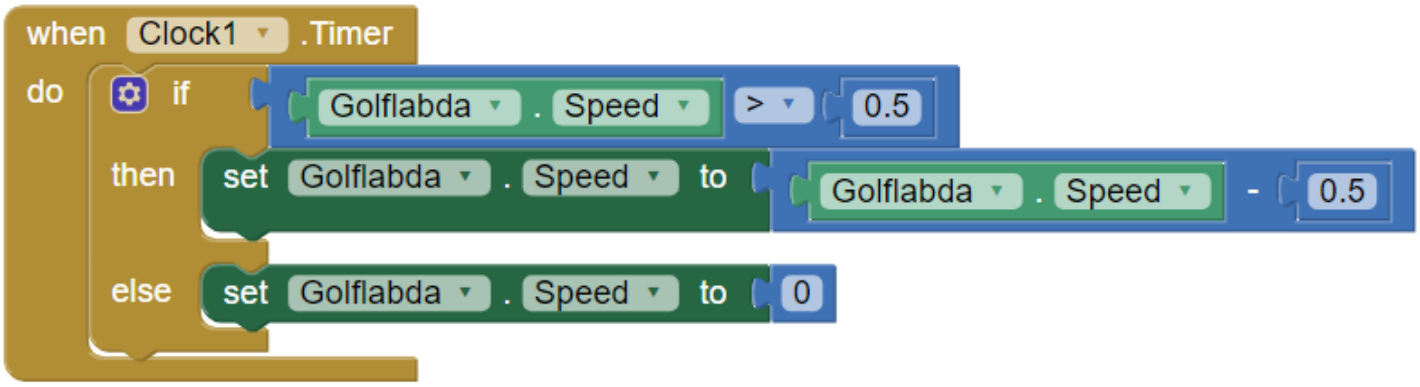


Próbáljuk ki: ujjunkkal a telefonon próbáljuk meg „lökdösni” a labdát.

Második lépés: lassulás. Megvalósítjuk, hogy lassuljon a golflabda. Design módban:

- Sensors → Clock (óra): ez egy ún. nem látható komponens.
  - TimeInterval (időintervallum): 100

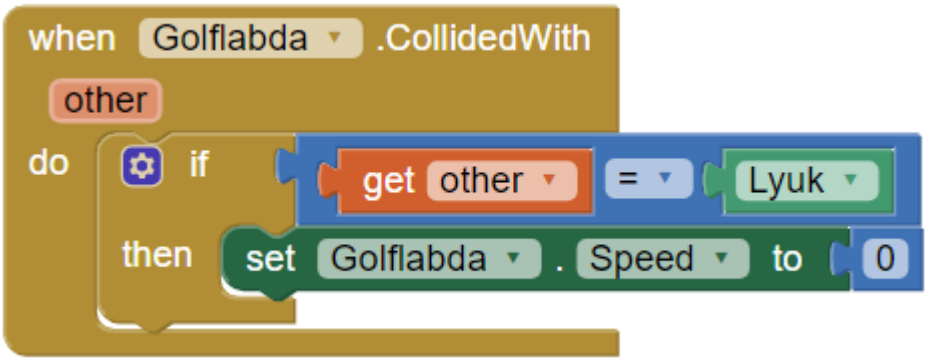
A kódja (Blocks):



Harmadik lépés: lyuk. Design:

- Drawing and Animation → Ball: a vászonra dobjuk rá
  - Név: Lyuk
  - Radius: 15
  - X: 150
  - Y: 50

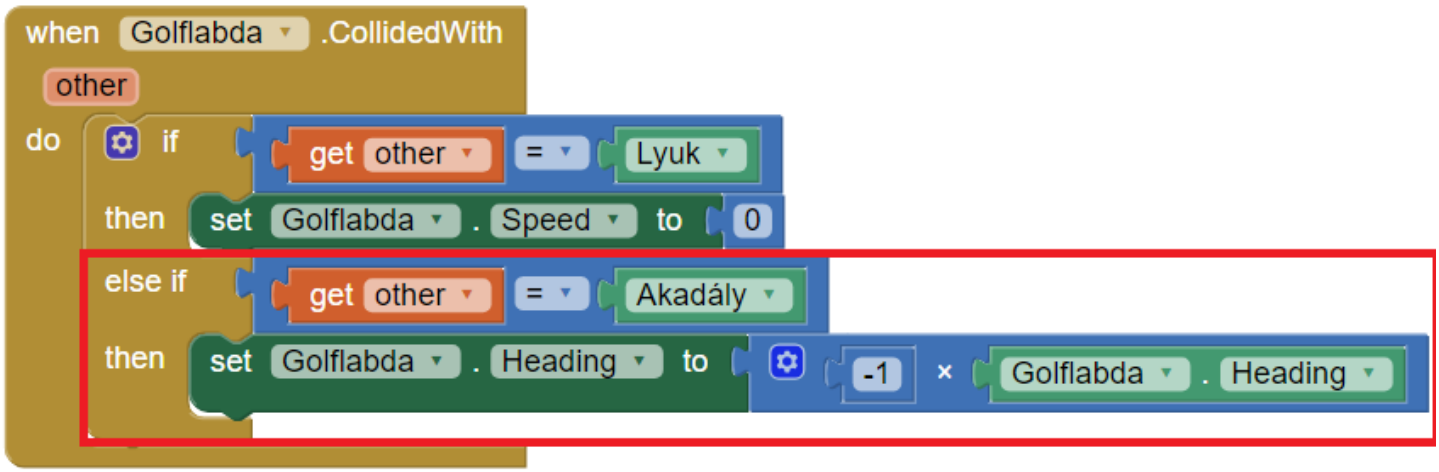
A kódban, ha ütközik a golflabda a lyukkal, akkor egyelőre csak megállítjuk a golflabdát. Az ütközés ellenőrzéskor meg kell vizsgálnunk, hogy mivel ütközik:



Negyedik lépés: akadály. Design:

- Drawing and Animation → ImageSprite: a vászonra ejtsük rá
  - Picture: ezt a képet mentsük le a saját számítógépünkre: http://appinventor.mit.edu/explore/sites/all/files/tutorials/miniGolf/golf_obstacle1.png, majd töltsük fel (Upload File…)
  - X: 100
  - Y: 125

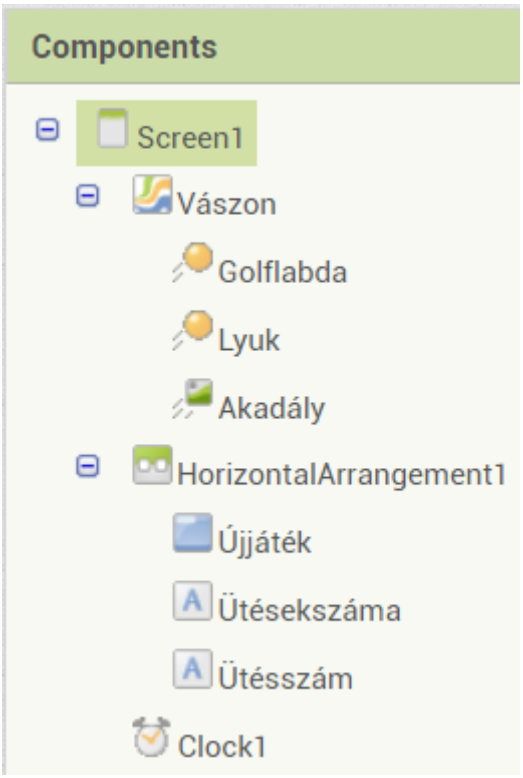
A kódban (Blocks) ha a golflabda ütközik az akadállyal, akkor visszapattan. (Megjegyzés: ez csak vízszintes pattanásnál működik helyesen.)



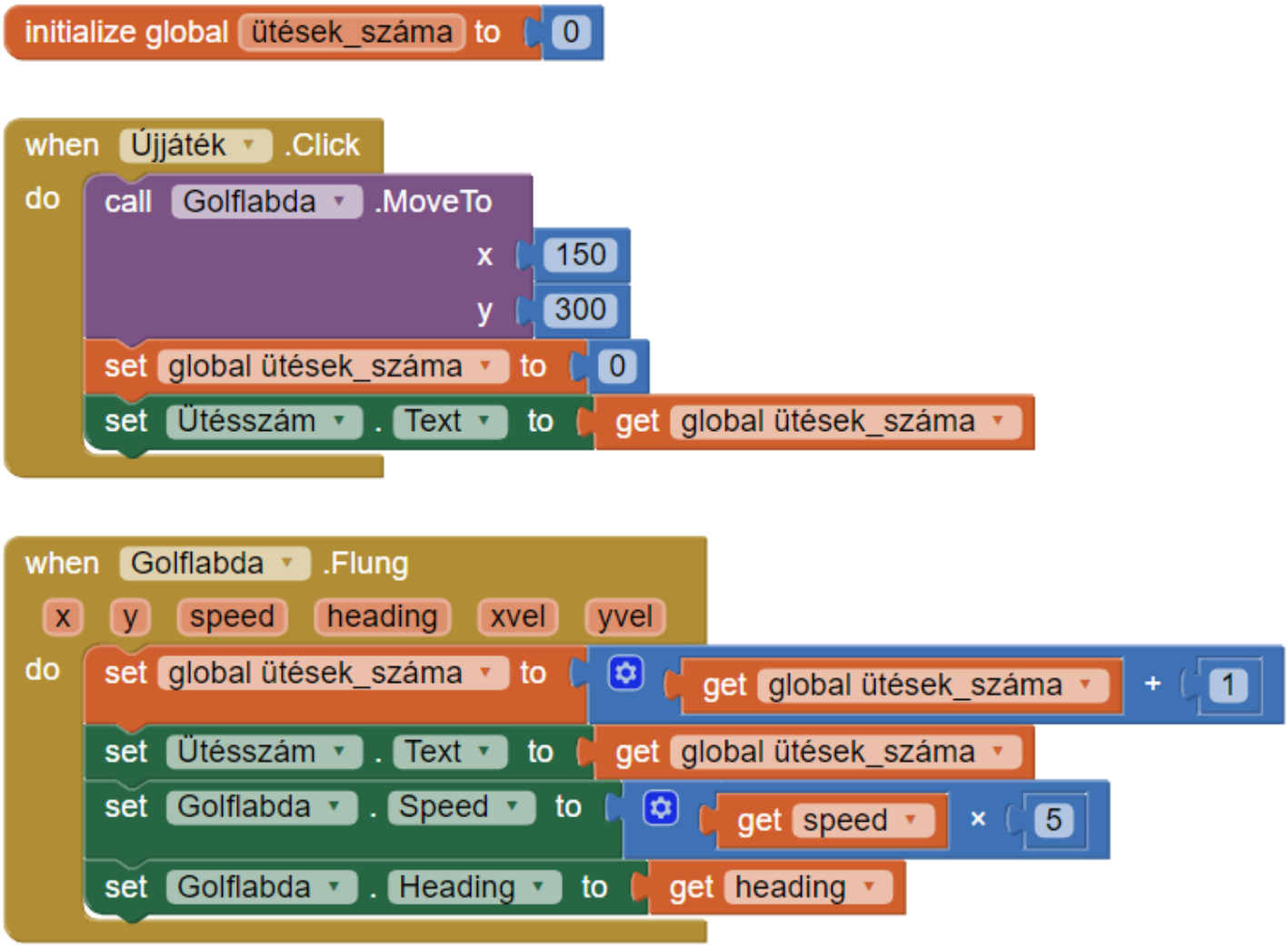
Ötödik lépés: új játék, pontszámítás. Design:

- Layout → HorizontalArrangement (vízszintes elrendezés): ez kerüljön a vászon alá
  - AlignHorizontal (vízszintes igazítás): Center (középre)
  - AlignVertical (függőleges igazítás): Center
  - Width: Fill parent
- User interface → Button (nyomógomb): ez kerüljön bele az imént létrehozott vízszintes elrendezésbe.
  - Név: Újjáték
  - Text: Új játék
- User interface → Label (címke): ez is a vízszintes elrendezésbe kerüljön, jobbra a nyomógombtól.
  - Név: Ütésekszáma
  - Text: Ütések száma:
- User interface → Label (címke): ez is a vízszintes elrendezésbe kerüljön, jobbra az előző címkétől.
  - Név: Ütésszám
  - Text: 0

A komponensek végeredménye az alábbi:



A kódban felveszünk egy globális változót, mellyel az ütéseket számoljuk, meg kell valósítani az új játék nyomógomb kódját, és a golflabda lökésének a kódját is módosítjuk, az alábbi módon:



Végeredmény: Minigolf.aia.

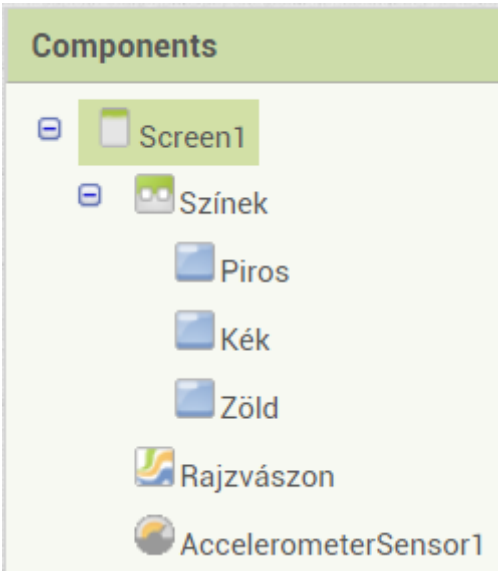
App Inventor - Rajzasztal

Ebben a programban az ujjunkkal fogunk rajzolni. Angol nyelvű útmutatók:

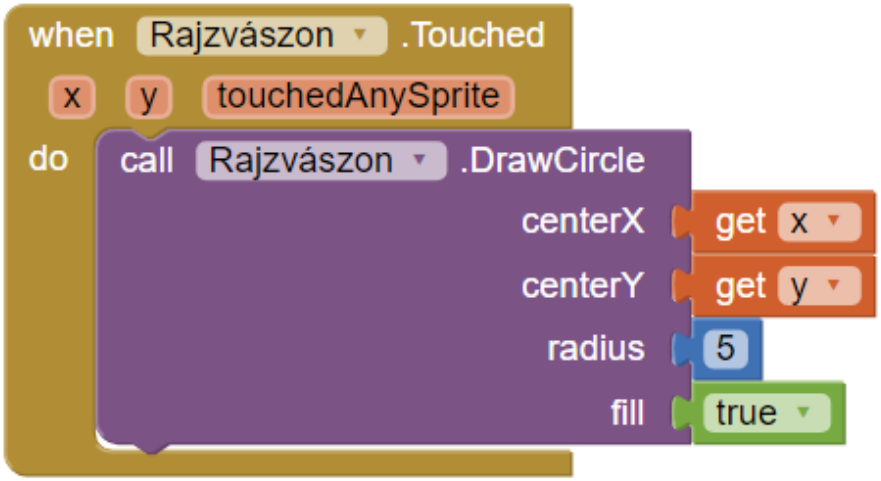
- https://www.youtube.com/embed/fQKNzLYEN0M
- http://appinventor.mit.edu/explore/sites/all/files/hourofcode/DigitalDoodle_2perpage.pdf
- http://appinventor.mit.edu/explore/ai2/paintpot-part1.html
- http://appinventor.mit.edu/explore/ai2/paintpot-part2.html

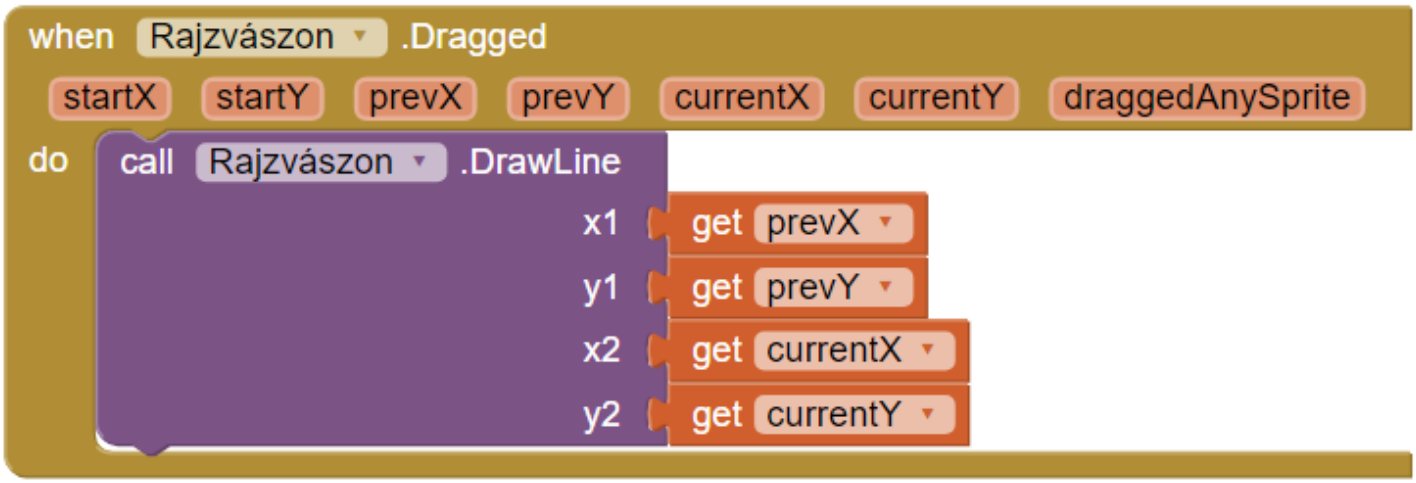
Design:

- Screen1
  - Title: Rajzasztal
  - ScreenOrientation: Portrait
- Layout → HorizontalArrangement
  - Név: Színek
  - AlignHorizontal: Center
  - Width: Fill Parent
- User Interface → Button: a Színek alá kerüljön
  - Név: Piros
  - BackgroundColor: Red (piros)
  - Text: Piros
- User Interface → Button: a Színek alá kerüljön
  - Név: Kék
  - BackgroundColor: Blue (kék)
  - Text: Kék
- User Interface → Button: a Színek alá kerüljön
  - Név: Zöld
  - BackgroundColor: Green (zöld)
  - Text: Zöld
- Drawing and Animation → Canvas: ez a Színek elrendezés alá kerüljön
  - Név: Rajzvászon
  - Height: Fill parent
  - Width: Fill parent
  - LineWidth: 2.0
- Sensors → AccelerometerSensor

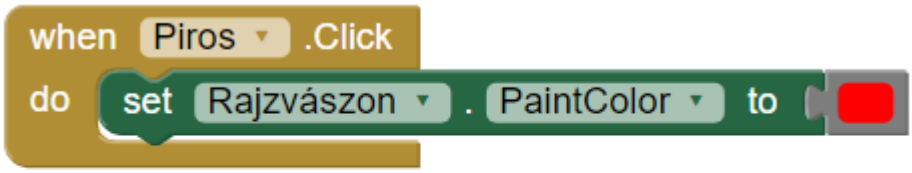


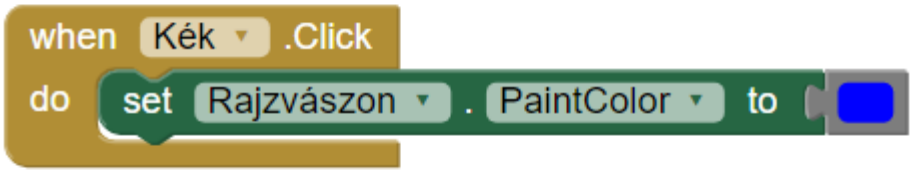
A kód (Blocks): először készítsük el a pont és a vonal rajzolót:

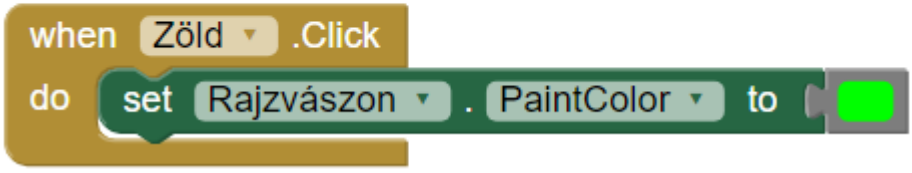




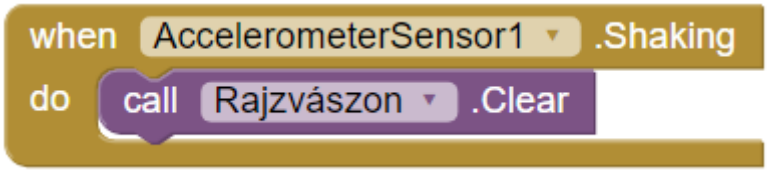
Próbáljuk ki, majd valósítsuk meg a színválasztást is:







Végül a képernyőt rázással tudjuk törölni:

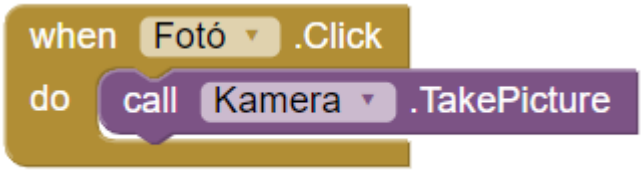


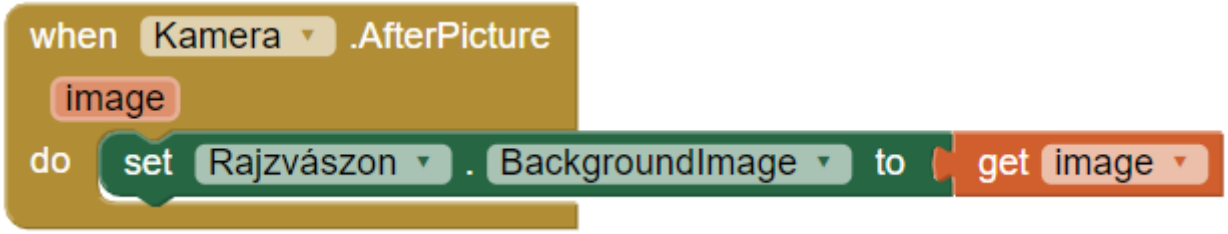
Fotó hozzáadása

A programot fotóval egészítjük ki: a hátteret egy frissen készített fotóval töltjük ki. Ehhez a tervező nézetben:

- Szúrjuk be egy fényképezőgép (Media -> Camera) komponenst (ami egy nem látható komponens), legyen a neve Kamera.
- Szúrjunk be egy nyomógombot, pl. a színek mellé; legyen a neve és a felirata is Fotó.

Majd a kódot a következőképpen egészítsük ki:





Amikor a Fotó nyomógombra kattintunk, akkor ki kell választanunk, hogy melyik felvevővel készítjük a fotót, majd miután elkészítettük és jóváhagytuk, akkor megjelenik, és összefirkálhatjuk.

Ha az elkészült alkalmazást önálló programként feltelepítjük, akkor a tapasztalat szerint fotó készítéskor többnyire hibaüzenetet ír ki. A hiba oka az, hogy alapértelmezésben általában nincs jogosultsága a programoknak a tárhelyhez, ezt külön meg kell adni. A legtöbb esetben ez a beállításokon belül az alkalmazások között van, vagy úgy, hogy azon belül vannak a jogosultságok, és a tárhelynél meg kell keresnünk a mi alkalmazásunkat, vagy az alkalmazáson belül kell jogosultságot adnunk a tárhely eléréshez.

Végeredmény: Rajzasztal.aia.

App Inventor - Hangyanyomkodó

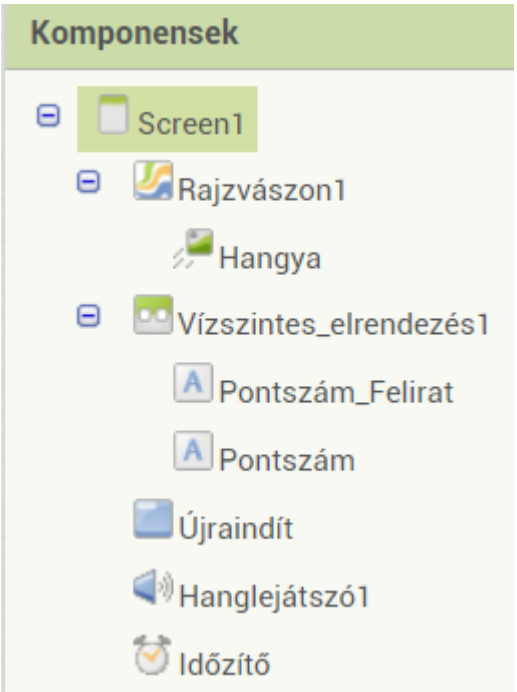
Ebben a játékban egy hangyát kell agyonnyomni, mielőtt átmenne máshova. Angol nyelvű útmutató:

- http://appinventor.mit.edu/explore/ai2/molemash.html

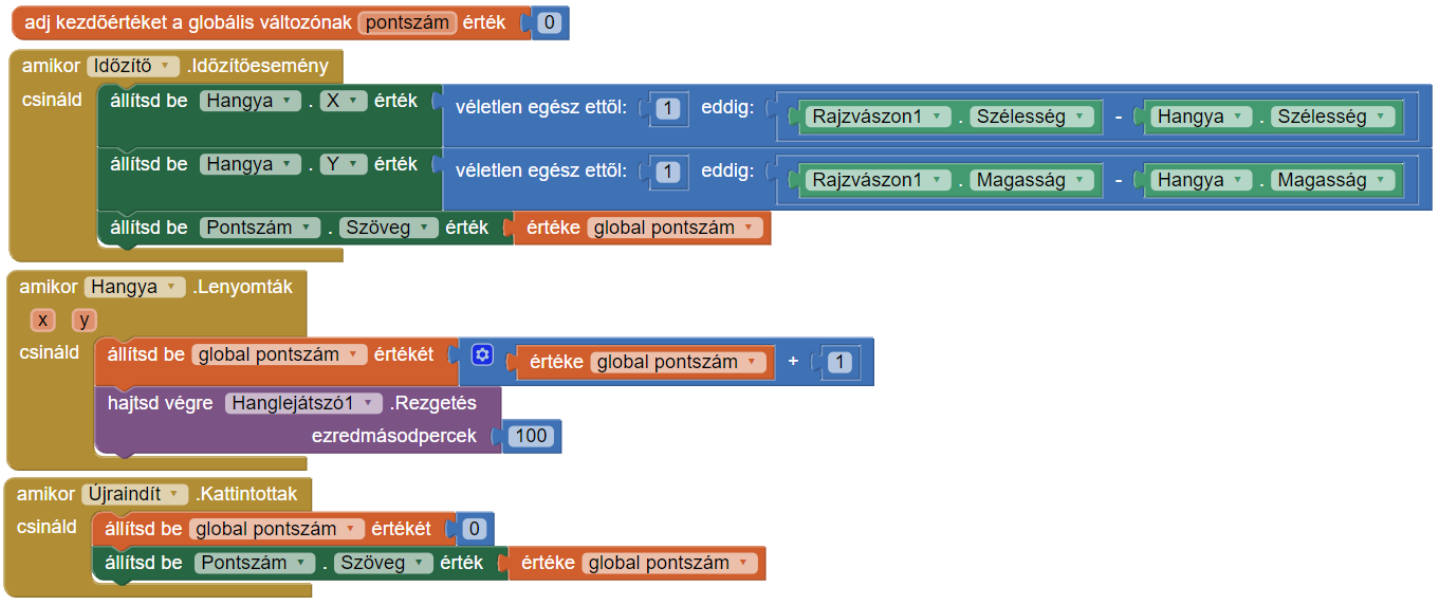
(A leírás a képernyőképekkel együtt a magyar verziót felhasználva készült. Az írás pillanatában ez egy teszt szerveren elérhető: http://ai2-test.appinventor.mit.edu/?locale=hu.)

Design:

- Screen1
  - Cím: Hangyanyomkodó
  - Képernyőirány: Portré
- Rajz és animáció → Rajzvászon
  - Magasság: Szülő komponens kitöltése
  - Width: Szülő komponens kitöltése
- Rajz és animáció → Szereplő: ez kerüljön bele a fenti vászonba
  - Név: Hangya
  - Magasság: 50 képpont
  - Szélesség: 50 képpont
  - Kép: keressünk az interneten egy hangyás képet (tipp: http://nobugs4u.com/wp-content/uploads/odorous-house-ant-lg.jpg), mentsük le, majd töltsük fel.
- Elrendezés → Vízszintes elrendezés: ebbe kerüljön az alábbi kettő
- Felhasználói felület → Címke
  - Név: Pontszám_Felirat
  - Szöveg: Pontszám:
- Felhasználói felület → Címke
  - Név: Pontszám
  - Szöveg: 0
- Felhasználói felület → Gomb
  - Név: Újraindít
  - Szöveg: Újraindít
- Média → Hanglejátszó: ez egy nem látható komponens
- Érzékelők → Óra
  - Időzítő intervallum: 500



Először a hangya véletlenszerű mozgását valósítsuk meg: az időzítő hatására ugorjon a vászon véletlen helyére. Ezt követően számoljuk a pontokat, ha sikerült agyonnyomni a hangyát. Végül valósítsuk meg az újraindítás kódját. A végeredmény az alábbi:



Végeredmény: Hangyanyomkodo.aia.

App Inventor Űrhajó

Most megvalósítjuk az űrhajós játék mobiltelefonos változatát. A Scratch változat itt látható: https://scratch.mit.edu/projects/238536384/. (Ehhez hasonló program található itt: http://appinventor.mit.edu/explore/ai2/space-invaders.html.) A megvalósítás során néhány előre valószínűleg nem várt nehézséggel nézünk szembe:

- Az App Inventorban nem lehet szereplőről másolatot készíteni. Ez egy olyan korlátozás, mely kizárja pl. azt, hogy egyszerre több golyó legyen a képernyőn.
- Az App Inventor szereplőválasztéka igen gyér: lehet kitöltött kör (ami a golyónak pont megfelel), és egy általános szereplő, amihez nekünk magunknak kell feltöltenünk a képét. A képek többnyire pixelesek (és valószínűleg nem is tud a rendszer vektoros képet kezelni).
- Ezzel kapcsolatos következő probléma: a kép beszerzése. Fontos ugyanis, hogy a háttér átlátszó legyen. (Az ilyen képek elkészítése kívül esik ezen a tanfolyamon)

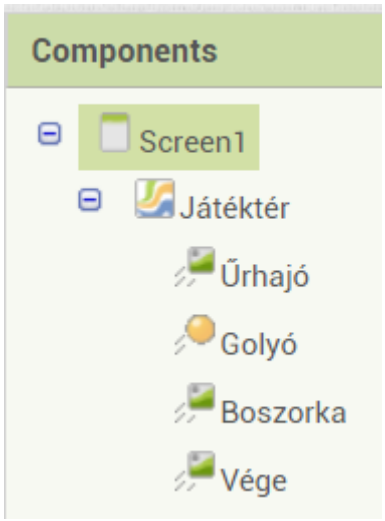
A mostani programhoz az alábbi szereplőkre lesz szükség, melyeket a megvalósítás felgyorsítása érdekében innen feltölthetőek:

- Űrhajó
- Boszi
- Vége

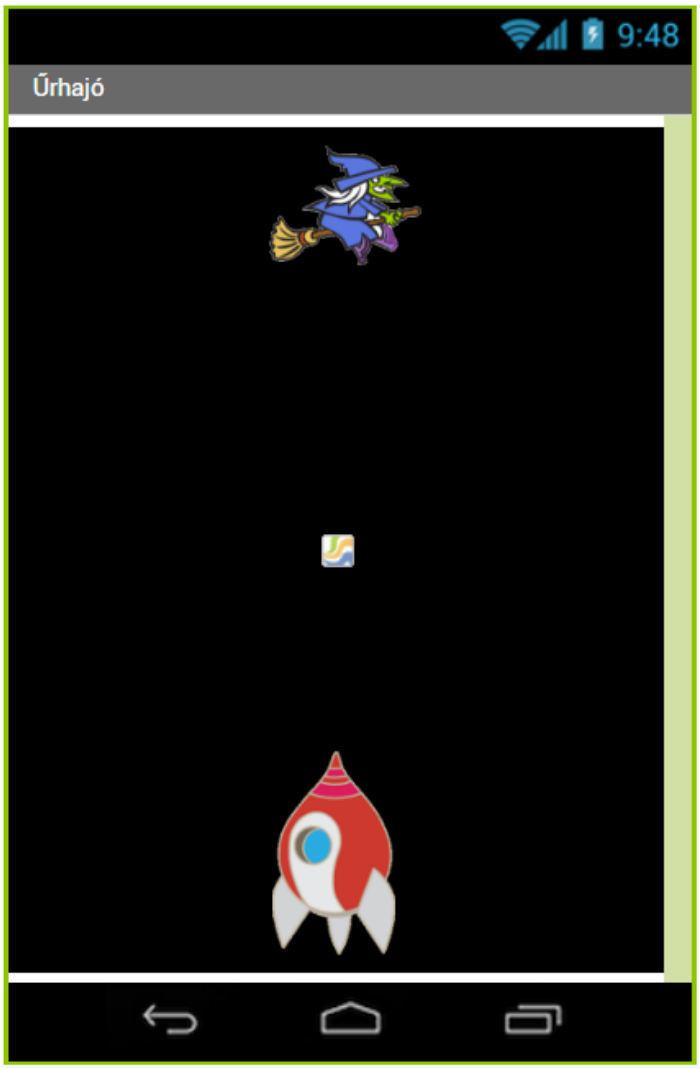
A képernyő portré módú legyen. A designerben hozzuk létre az alábbit:

- A Screen1-en adjunk nevet a játéknak, pl. Űrhajó.
- Adjunk hozzá egy vásznat (Canvas), amely kitölti a teljes ablakot. A színe legyen fekete.
- A vászonra helyezzünk 4 szereplőt:
  - Űrhajó (ImageSprite, a magasság-szélesség legyen 100x70 képpont, a képe (Picture) legyen a fenti űrhajó, az X-Y pozíciója (120, 300))

- Boszorka (ImageSprite, 70x80 képpont, boszorkány kép, (120, 0), a sebessége (Speed) legyen 10.0)
- Vége (ImageSprite, 50x150, vége kép, (87, 127), kapcsoljuk ki a láthatóságát)
- Golyó (Ball, narancssárga, átmérő: 10, a pozíció mindegy, kapcsoljuk ki a láthatóságát)

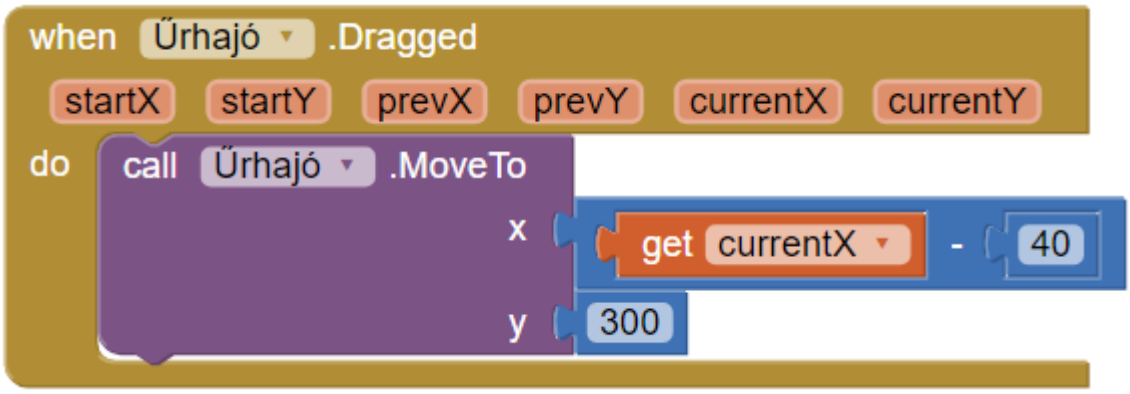


A képernyő így néz ki:

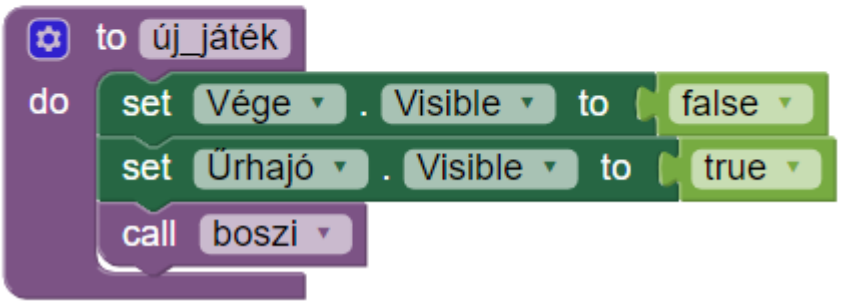


Készítsük el a kódot!

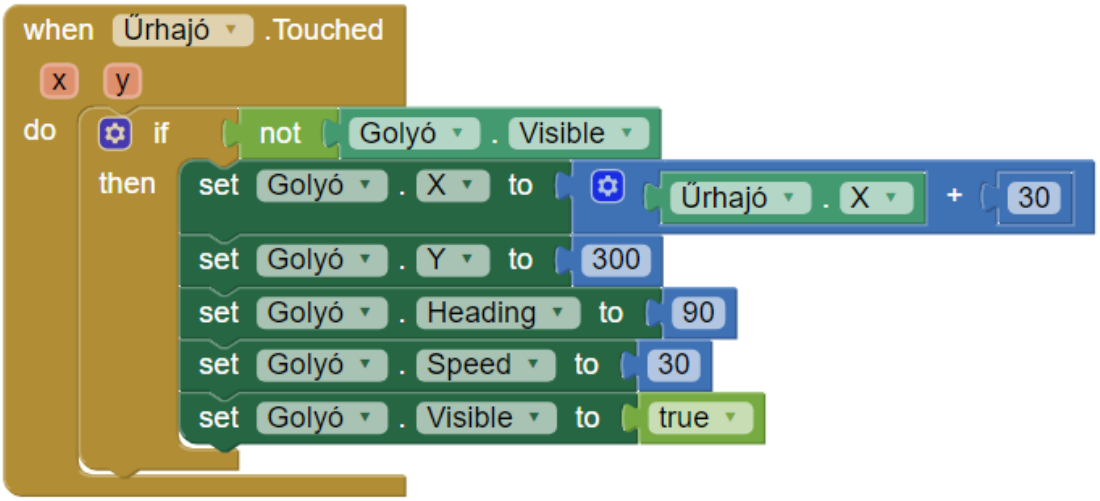
- Az űrhajó mozgatásának a kódja az alábbi. Mivel igen látványos, kezdhetjük ezzel. Mozgatni a szereplő vonszolásával tudjuk.

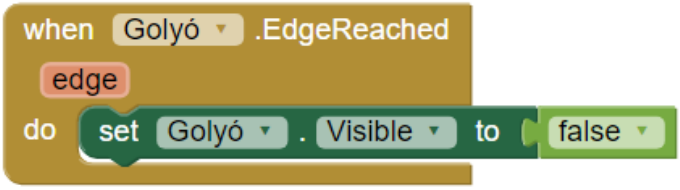


- A játék elején az beállító rész a következő. Célszerű a fenti kódrészlet fölé helyezni. Az új_játék egy eljárás: Procedures → to … do; itt adjunk neki nevet. (Az boszi szintén egy eljárás, amit később valósítunk meg; azt egyelőre hagyjuk ki.)

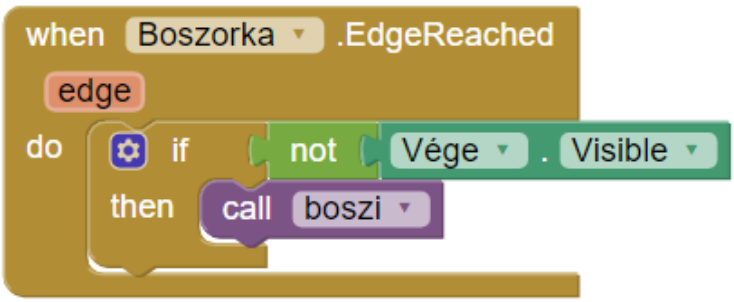


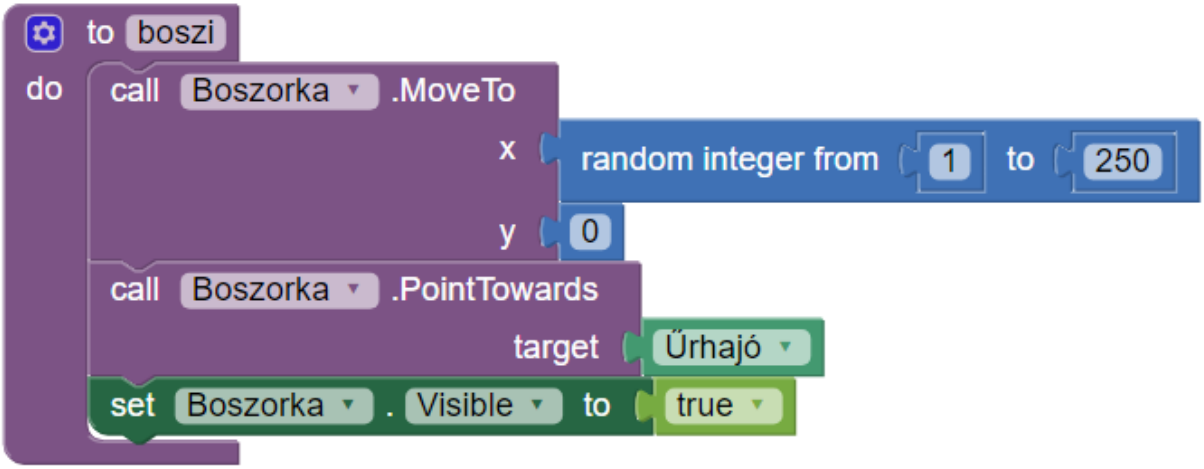
- Ha a játékos rákattint az űrhajóra, akkor az űrhajó lő egyet. Amíg egy golyó repül, másikat nem lehet lőni. Ennek ez a kódja:



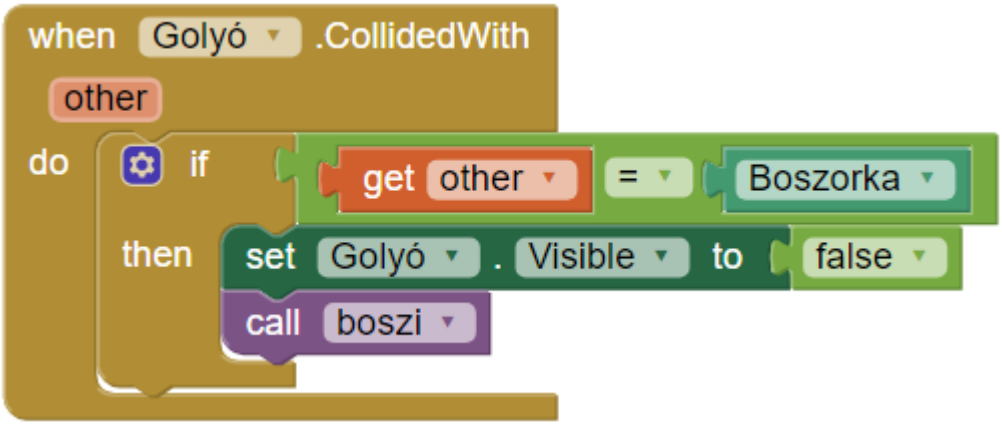


- A boszorkány kódja az alábbi. Ha a boszorkány eléri a játéktér szélét, akkor egy fenti véletlenszerű pozícióba ugrik, és elindul az űrhajó felé. Ne feledjük a már elkészített új_játék eljárásba beletenni az eljárás meghívását.

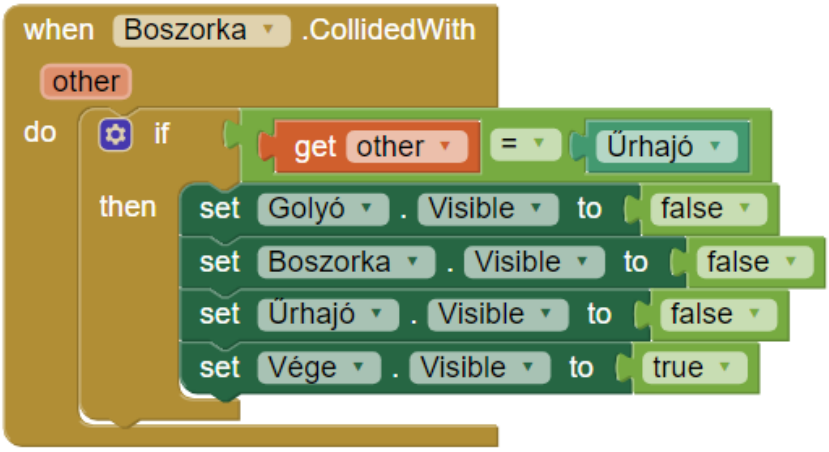




- Ha a golyó eltalálja a boszorkányt, akkor az eltűnik, majd újraindul.



- Ha a boszorkány nekimegy az űrhajónak, akkor véget ér a játék.



- Végül a játék újraindul, ha rákattintunk a VÉGE feliratra.

Végeredmény: Urhajo.aia.

Még nagyon sok kiterjesztési lehetőség van: több élet, több szereplővel megoldott több golyó és több boszorkány, gyorsuló, nehezedő játék, pontszámítás stb.

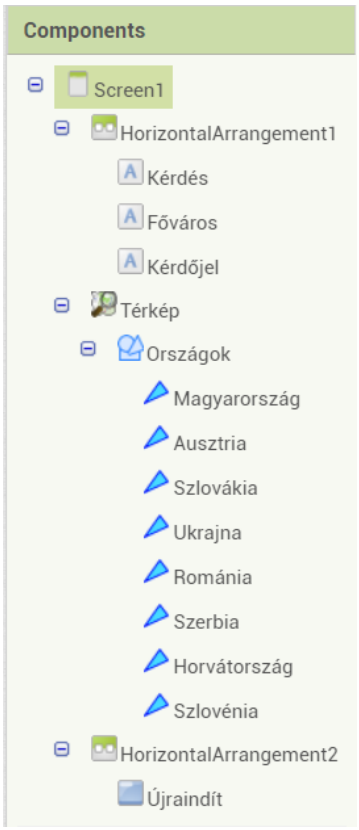
App Inventor - Térképes kvíz

Ez a program egy kvíz: Magyarország és szomszédai fővárosait kell eltalálnunk a térképen. Angol nyelvű útmutató:

- http://appinventor.mit.edu/explore/ai2/state-geography-quiz.html
- http://appinventor.mit.edu/explore/sites/all/files/ai2tutorials/stateQuiz/StateQuizTutorial.pdf

A játék a térkép (Map) komponenst mutatja be. A területek körülrajzolásához a geojson formátumot használjuk. Az interneten számos területet találunk meg ebben a formában; Európa országai (ami alapján ez a kvíz is elkészült), itt található: https://github.com/leakyMirror/map-of-europe.

Design

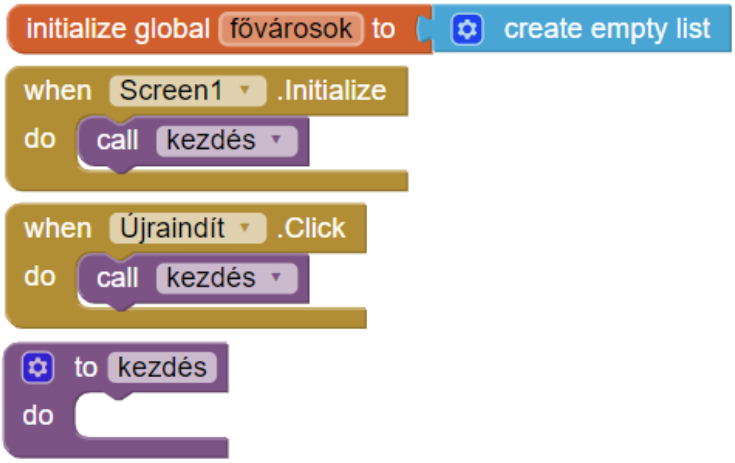


- Screen1 title: Magyarország és szomszédai
- Felső HorizontalArrangement: kerüljön bele 3 felirat (Label), melyeknél a HasMargins legyen kikapcsolva.
    - Első felirat szövege: "Melyik ország fővárosa "
    - Második felirat szövege: egyelőre maradjon az alapértelmezett
    - Harmadik felirat szövege: "?"
- Maps -> Map
    - CenterFromString: 45,20
    - Height, Width: Fill parent…
    - ZoomLevel: 4
- A fenti térképbe kerüljön bele egy Maps -> FeatureCollection
    - Source: magyarorszag-szomszedai.geojson (ez a fájl tartalmazza az országokat; töltsük le, csomagoljuk ki, és a fájt töltsük fel)
- Legyen alul egy Újraindít nyomógomb (a példában középre zárva).

A képernyőterv kb. így néz ki:

Blocks

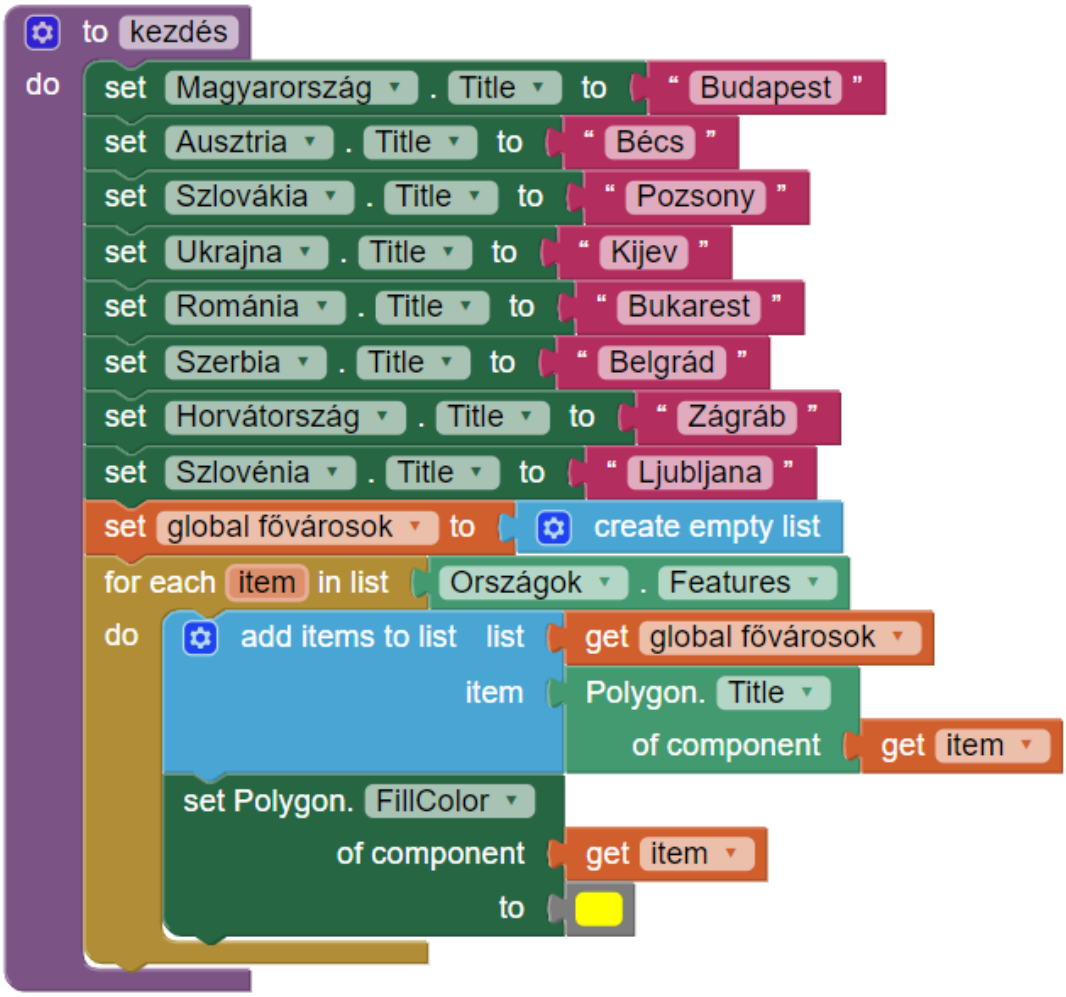
Először hozzunk létre egy fővárosok globális, kezdetben üres listát, valamint egy kezdés eljárást, melyet meghívunk rögtön a képernyő betöltésekor, valamint újraindításkor is:



Valósítsuk meg a kezdési beállításokat végrehajtó eljárást! Itt a következőket kell tenni:

- Mindegyik országnak adjuk meg címében a fővárosát. (A zöld részeket a Térkép -> Országok alatt találjuk, az egyes országokra kattintva. Ha az egyiket kiválasztottuk, akkor már tudjuk duplázni.)
- Töltsük fel a fővárosok globális listát az országok "címeivel". Itt amiatt kell ismét kiüríteni a lista tartalmát, mert ha az újraindítás nyomógombra kattintva jutottunk ide, akkor lehetséges, hogy nem üres.
- Színezzük be sárgára az egyes országokat.

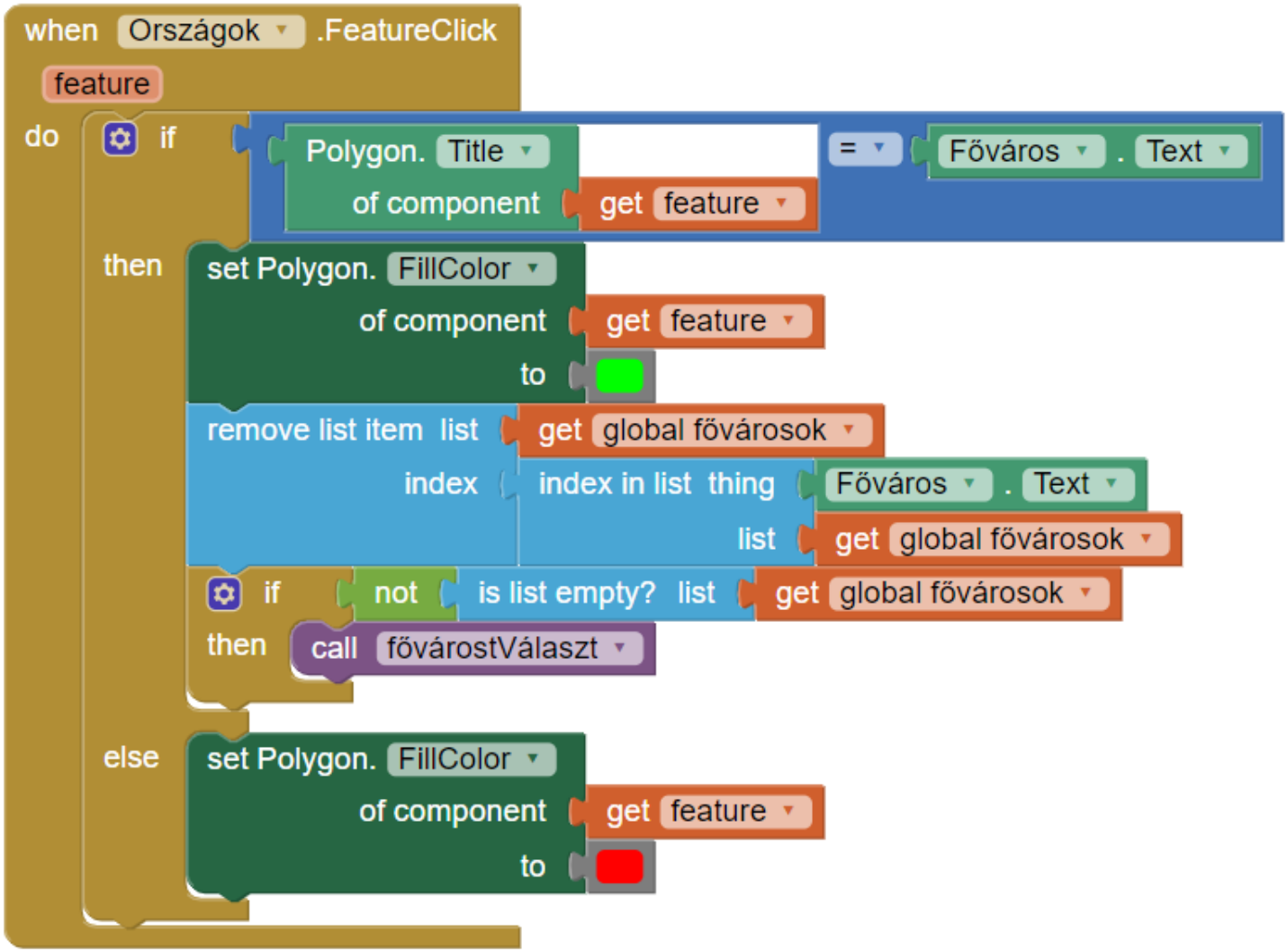
A kék komponenseket a Built-in -> Lists, míg a Polygon-t legalul az Any Component -> Any Polygon alatt.



Készítsünk egy függvényt fővárostVálaszt névvel, amely kiválaszt egy véletlenszerű fővárost, és beleírja a kérdésbe. Ezt rögtön hívjuk is meg a kezdeti beállító eljárás végén. Amiatt szervezzük ki külön eljárásba, mert máshonnan is meg fogjuk hívni.

Végül azt az eljárást kell megvalósítanunk, ami az egyes országokra kattintva fut le. Ezt a Screen1 -> Térkép -> Országok részen belül találjuk. Itt:

- Ha eltalálta a játékos a megfelelő országot (tehát a kiválasztott terület címe megegyezik a kérdés második részével), akkor egyrészt beszínezzük zöldre az adott országot, kivesszük a fővárosok listából az eltalált elemet, majd ha még nem fogyott el mind, választunk egy véletlenszerű fővárost.
- Ha nem találja el, akkor a kiválasztott ország színét pirosra változtatjuk.



Végeredmény: Kviz.aia.

Ezt a játékot számos ponton lehet továbbfejleszteni:

- Ha egy ország már "kizöldült", akkor rákattintva ne legyen ismét piros.

- Akkor is másik kérdést tegyen fel, ha a játékos nem találta el.
- A piros (helytelen) csak egy rövid ideig villanjon fel, ne maradjon úgy.
- A végén tűnjön el a kérdés, és írjon ki egy alkalmas üzenetet.
- Időmérőt is be lehetne tenni.
- Egyéb térképe egyéb kérdésekkel: pl. Magyarország megyéi.

Ezek viszont már azok a lépések

Kapcsolat a robot és az okostelefon között

Makeblock app

Ha Bluetooth verziójú robotunk van, akkor telepítsük fel a Makeblock alkalmazást a Play webáruházból. Töltsük az eredeti firmware-t a következőképpen:

- Ha még nem tettük meg, telepítsük fel az mBlock alkalmazást (http://www.mblock.cc). (//Megjegyzés: az mBlock 3 a Scratch 2-re, az mBlock 5 pedig a Scratch 3-ra épül. Az írás pillanatában csak az mBlock 3-ml tudjuk elkészíteni a lenti programot, így azt használjuk..)
- Csatlakoztassuk a robotot USB kábelen keresztül, majd az mBlock programból Connect → Serial Port - COMx, utána Connect → Reset Default Program → mBot.
- Válasszuk le a robotot a számítógépről és indítsuk újra.

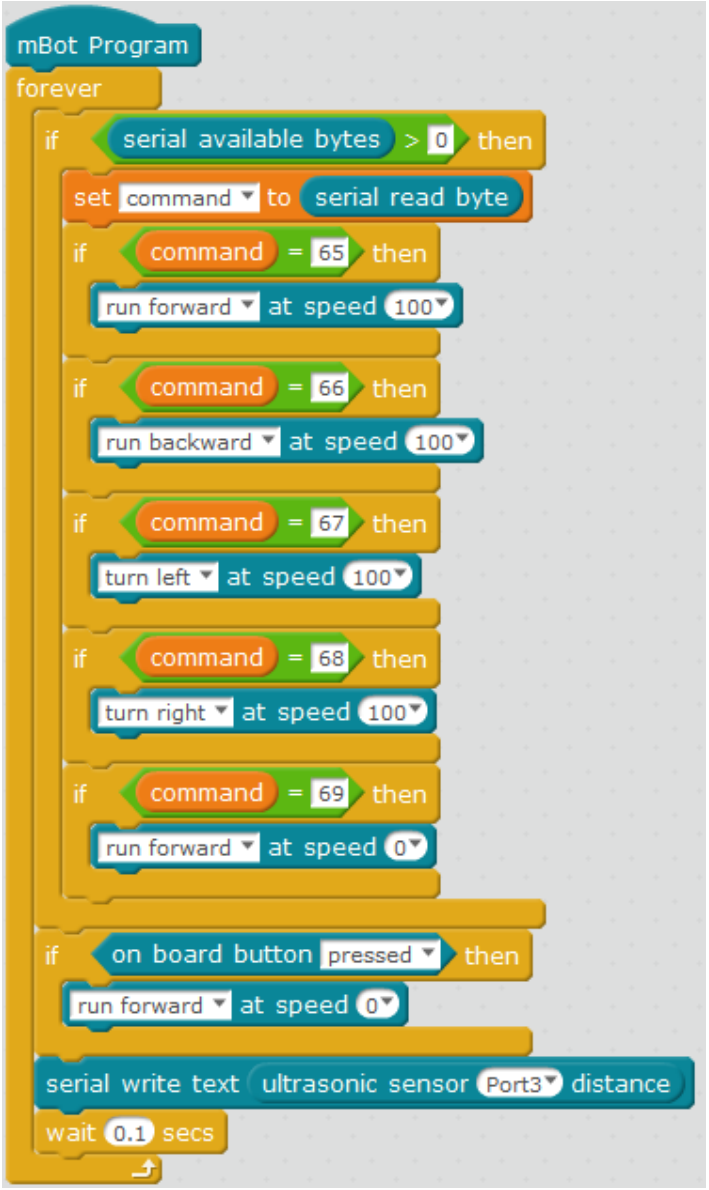
Az Makeblock alkalmazásban csatlakozzuk a robothoz Bluetooth segítségével. Számos lehetőséget biztosít: távirányítás, vonalkövetés, zenélés, lámpa bekapcsolása, értékelések, kisebb programok írása.

Saját robot program

Egy hasonlót saját magunk is elkészíthetünk, ez viszont némi előkészületeket igényel.

- Ha nem az alapértelmezett program van feltöltve, töltsük azt fel a fent leírtak segítségével.
- Az okostelefonunkat párosítsuk a robottal: Beállítások → Bluetooth → kapcsoljuk be, majd frissítsük a párosított eszközök listáját. Ha mindent jól csináltunk, megtalálja a robotot Makeblock néven, azt fogadjuk el.
- A robot program elkészítéséhez az Arduino utasításkésztlet is hozzá kell adni az mBlock fejlesztőkörnyezethez (Extensions → Arduino); a serial (soros) parancsok ott érhetőek el (ezek valójában a Bluetooth parancsok).

Most készítsük el a következő programot:



A programot innen is letölthetjük: [távirányító program].

Utasításként az A, B, C, D és E betűket (egészen pontosan azok ASCII kódját) tudja értelmezni, és ez alapján hajtja végre a műveleteket, ami jelen esetben a robot mozgatását jelenti. A robotot le tudjuk állítani a rajta található nyomógomb segítségével. (Ha nem működik, teszt céllal először érdemes egyszerűbb műveletet végrehajtani, pl. LED-ek világítása.)

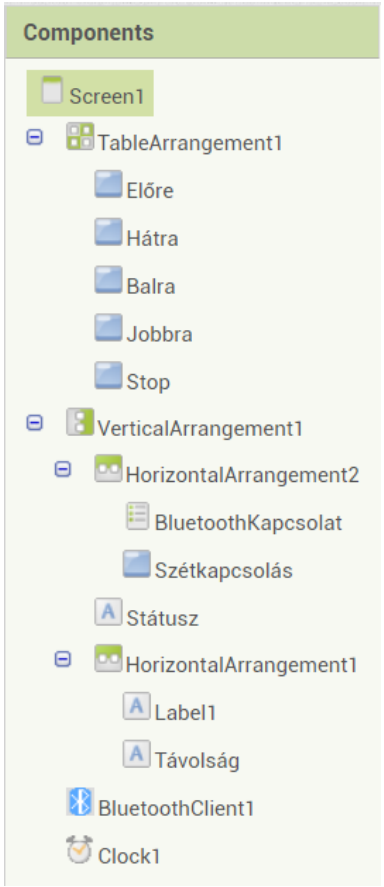
Töltsük fel a programot a robotra: csatlakoztassuk a robotot USB kábelen keresztül, majd Connect → Serial Port → COMx, majd jobb kattintás a robot programjára → Upload to Arduino.

(*Megjegyzés: a feltöltést követően általában "szemetet" talál a soros porton, aminek következtében elindul a robot. Ezt a problémát sajnos nem sikerült kiküszöbölni. Emiatt feltöltéskor érdemes megfordítani a robotot, majd nyomjuk meg a roboton található nyomógombot a leállításhoz.*)

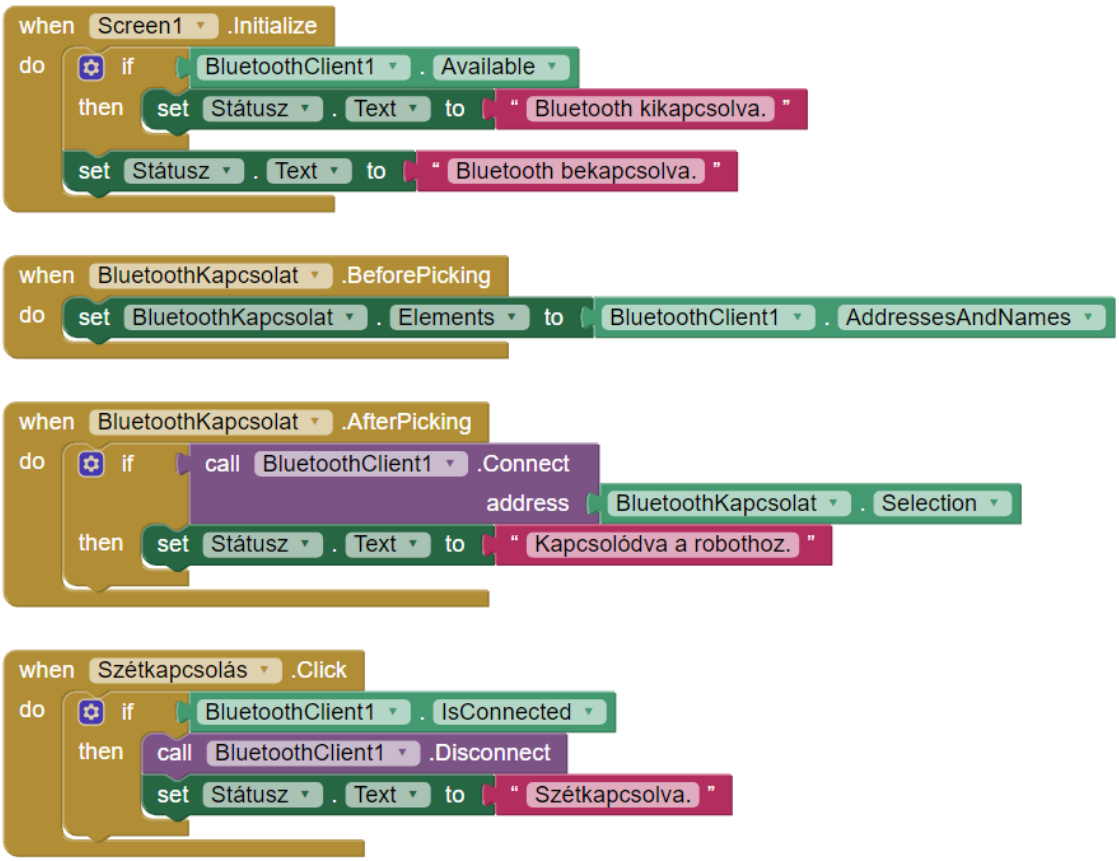
Saját robot app

Az Android alkalmazás felépítése az alábbi:

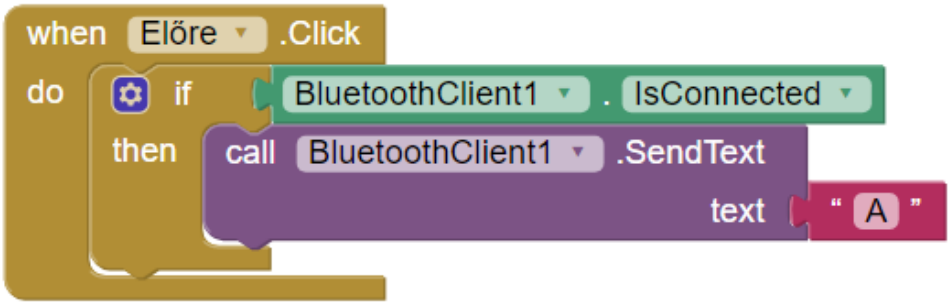
A nyomógombként ábrázolt „Kapcsolódás” valójában egy listaválasztó gomb (User Interface → ListPicker), a többi az, aminek látszik. A komponens nézet:

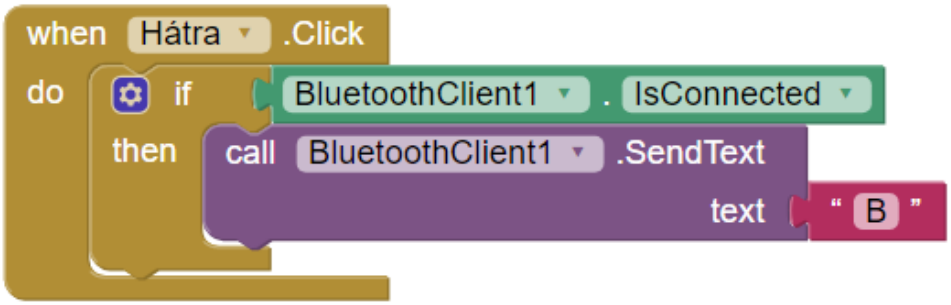


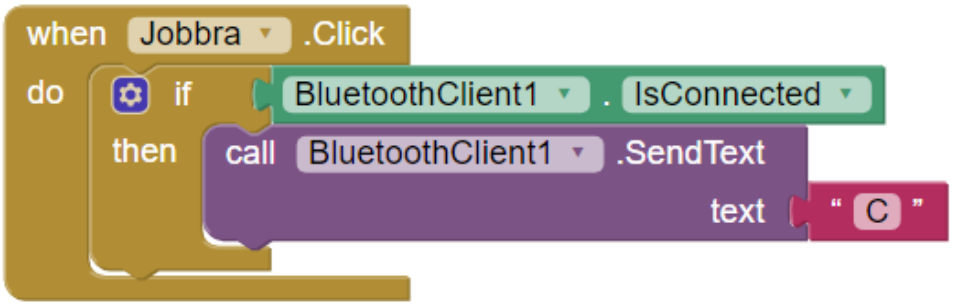
Az óra időzítőjét állítsuk 100 ezredmásodpercre. A kapcsolódással kapcsolatos kód az alábbi:

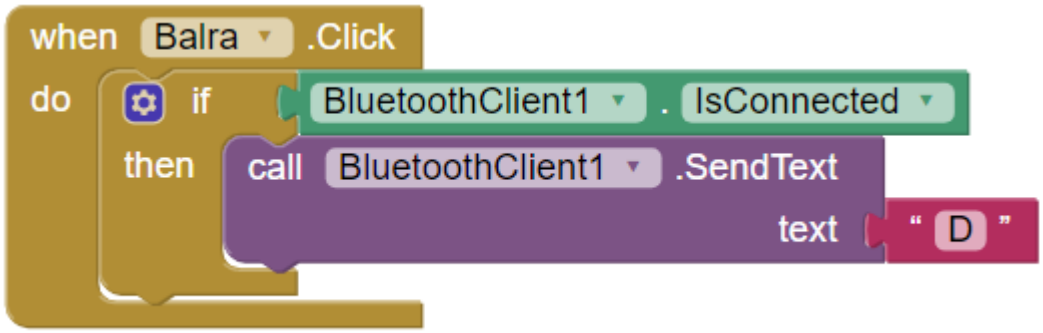


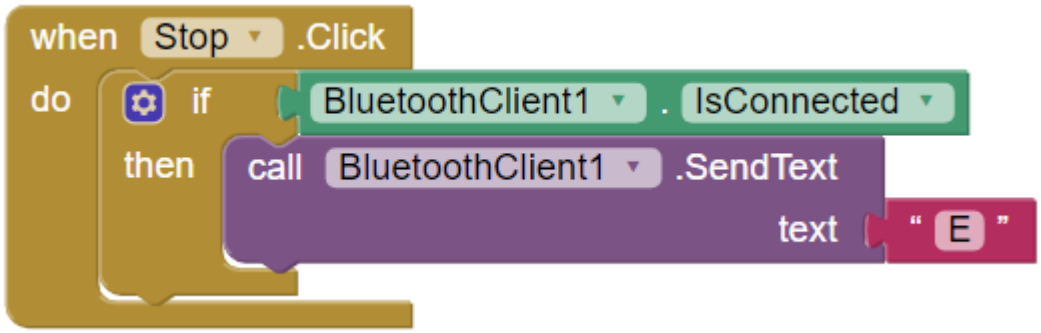
A kiadható utasítások kódja:



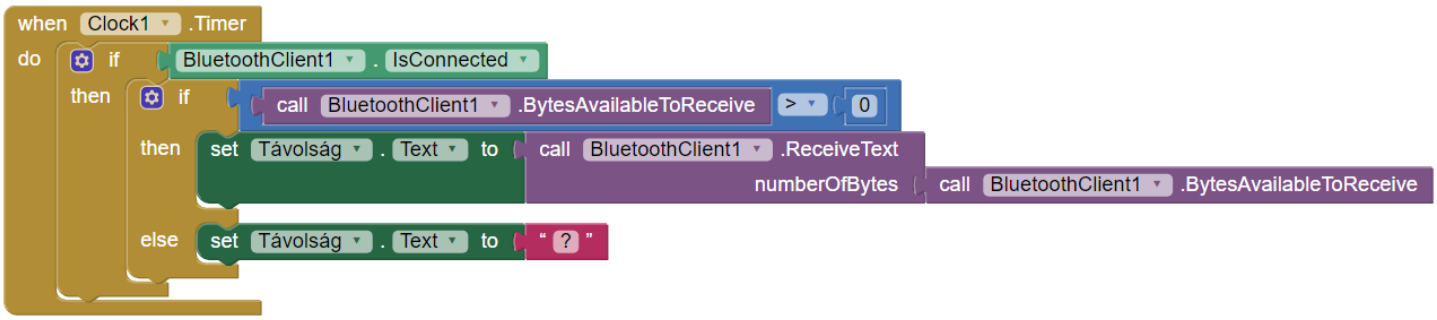




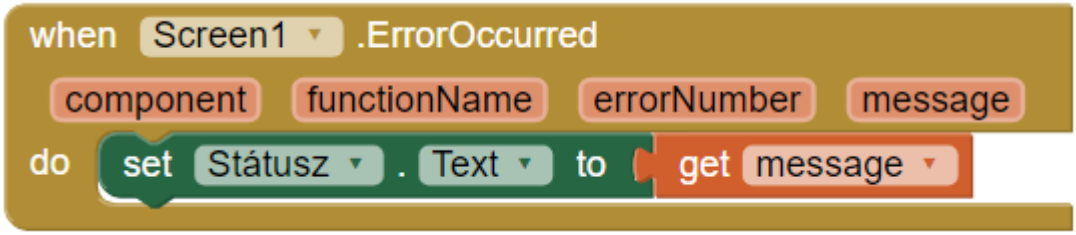




Az adatok fogadását az alábbi kódrészlet valósítja meg:



Végül egy alap hibakezelést is megvalósítunk, mivel a vezeték nélküli kapcsolat igen sok hibát rejthet magában.



A végeredmény ez: Távirányító.

Megjegyzés: elképzelhető, hogy a fentiek megvalósítása meghaladja egy alkalom kereteit. Emiatt azt javaslom, hogy töltsük le és importáljuk ezt: Távirányító (kapcsolat). Ez a bluetooth kapcsolatot valósítja meg. Így elég csak az adatküldésre és adatfogadásra szorítkoznunk.

Teszt

Ezzel létrehoztunk egy távirányítós kisautót, ahol a távirányító az okostelefon, a kisautó pedig a robot. A kipróbáláshoz telepítsük önálló alkalmazásként az appot, indítsuk el, majd kattintsunk a Kapcsolódás nyomógombra. Ott - ha korábban megtörtént a párosítás - a listából válasszuk ki azt az elemet, amelynek

a kód utáni neve az, hogy Makeblock. Sikeres kapcsolódást követően irányíthatjuk a nyomógombok segítségével, valamint az alkalmazásunkban láthatjuk a legközelebbi akadály távolságát.

Az alkalmazás elkészítéséhez sokat segítettek az alábbi angol nyelvű leírások:

- https://appinventor.pevest.com/?p=520
- https://openlab.makeblock.com/topic/589a3d318de529de3aa46a63