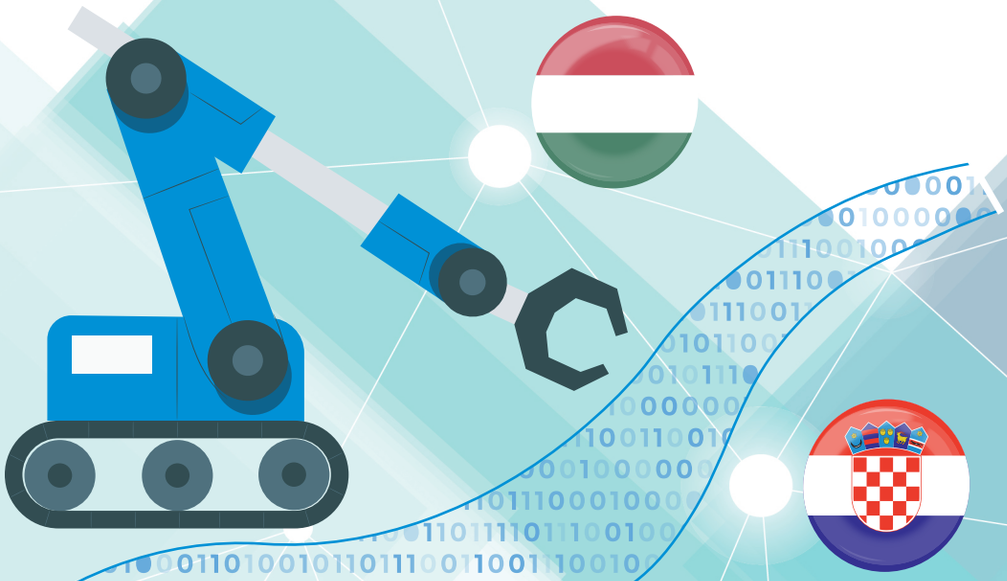
Interreg
European Regional Development Fund

A cross-border region where rivers connect, not divide

Hungary-Croatia
Cross-border Co-operation Programme

ROBOTics in Interregional COoperation



ROBOTICO

3-lingual learning handbook (English, Croatian, Hungarian)

Autors:
Dejan Rakijašić
Branko Pleadin
Tihomir Robotić

Đurđevac, July 2022

“This document has been produced with the financial assistance of the European Union. The content of the document is the sole responsibility of Vocational school Đurđevac and can under no circumstances be regarded as reflecting the position of the European Union and/or the Managing Authority. ”

A cross-border region where rivers connect, not divide

“Ovaj dokument proizveden je uz financijsku pomoć Europske unije. Sadržaj ovog dokumenta isključiva je odgovornost Strukovne škole Đurđevac i ni pod kojim uvjetima ne odražava stav Europske unije i/ili Upravljačkog tijela.”

Prekogranična regija- gdje rijeke spajaju, a ne razdvajaju

“A jelen dokumentum az Európai Unió társfinanszírozásával valósult meg. A tartalma kizárólag az Đurđevac Szakiskola felelősségi körébe tartozik. Az itt megtalálható anyagok nem tekinthetők olyanoknak, mint amelyek az Európai Unió és / vagy a HU-HR Interreg V-A CBC Program Irányító Hatóságának hivatalos állásfoglalását tükrözik. ”

Egy határon átnyúló régió, ahol a folyók összekötnek, nem elválasztanak

Index:

Sadržaj

Tartalom

1. Arduino

Arduino is an open-source hardware and software company, project and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices. Its hardware products are licensed under a CC-BY-SA license, while software is licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL), permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially from the official website or through authorized distributors. The name Arduino comes from a bar in Ivrea, Italy, where some of the founders of the project used to meet.[1]

Arduino je ime za otvorenu računalnu i softversku platformu koja omogućava dizajnerima i konstruktorima stvaranje uređaja i naprava koje omogućuju spajanje računala s fizičkim svijetom tj. stvaranje interneta stvari. Arduino je stvorila talijanska tvrtka SmartProjects 2005. rabeći 8-bitne mikrokontrolere Atmel AVR, da bi stvorili jednostavnu, malu i jeftinu platformu s kojom bi mogli lakše povezivati računala s fizičkim svijetom. Dizajneri su izabrali ime Arduino po imenu kafića u kojem su se sastajali kada su stvarali projekt.[2]

Az Arduino egy nyílt számítógép- és szoftverplatform elnevezése, amely lehetővé teszi a tervezők és a kivitelezők számára, hogy olyan eszközöket és modulokat hozzanak létre, amelyek lehetővé teszik a számítógépek összekapcsolását a fizikai világgal, azaz a dolgok internetének létrehozását. Az Arduino-t az olasz SmartProjects cég hozta létre 2005-ben 8 bites Atmel AVR mikrokontrollerek segítségével, hogy egy egyszerű, kicsi és olcsó platformot hozzanak létre, amellyel könnyebben tudták összekapcsolni a számítógépeket a fizikai világgal. A tervezők az Arduino nevet annak a kávézónak a neve után választották, ahol a projekt elkészítésekor találkoztak.

1.1. Arduino integrated development environment installation

1.1. Instalacija Arduino integrirane razvojne okoline

1.1. Az Arduino integrált fejlesztői környezet telepítése

Start your web browser and in the search field type “Arduino”

Pokrenite web preglednik i u polje za pretragu upišite „Arduino“.

Indítsa el a webböngészőt, és írja be az „Arduino” kifejezést a keresőmezőbe.

[1] https://en.wikipedia.org/wiki/Arduino
[2] https://hr.wikipedia.org/wiki/Arduino

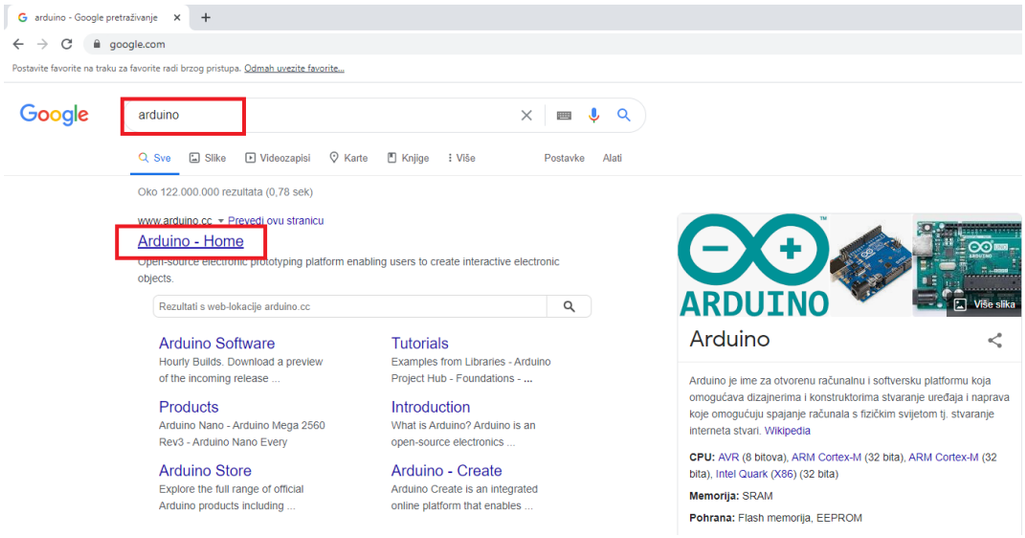


Figure 1: Official Arduino IDE website

By clicking on the first search result Arduino – Home you will be directed to https://www.arduino.cc/ webpage. It is an official Arduino project webpage that allows you to download Arduino integrated development environment, read the manual or buy Arduino devices.

Klik na rezultat pretrage Arduino – Home preusmjerit će vas na stranicu https://www.arduino.cc/ . To je službena stranica Arduino projekta na kojoj možete preuzeti integriranu razvojnu okolinu, pročitati dokumentaciju ili kupiti Arduino uređaje.

Az Arduino - Home keresési eredményre kattintva átirányítja Önt a https://www.arduino.cc/ oldalra. Ez az Arduino projekt hivatalos oldala, ahol letöltheti az integrált fejlesztői környezetet, elolvashatja a dokumentációt vagy vásárolhat Arduino eszközöket.

Follow the integrated development environment installation instructions.
Slijedite upute za instalaciju integrirane razvojne okoline.
Kövesse az utasításokat az integrált fejlesztői környezet telepítéséhez.

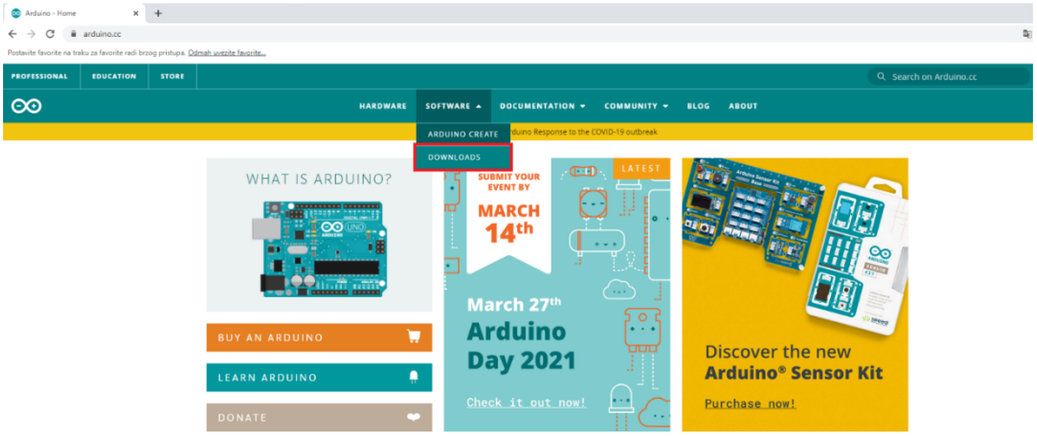
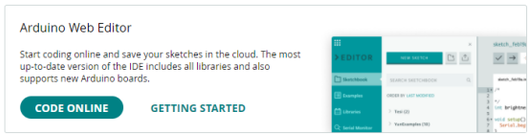


Figure 2: Download the Arduino integrated development environment



Downloads

Figure 3: Choose Arduino version for your operating system

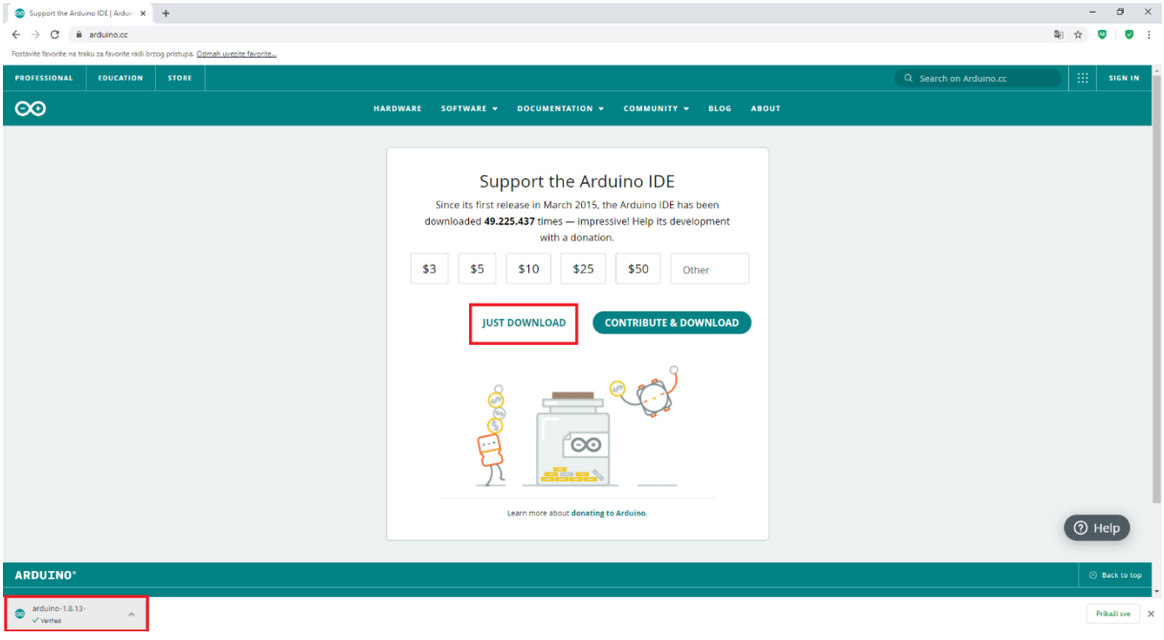


Figure 4: Run the installation file

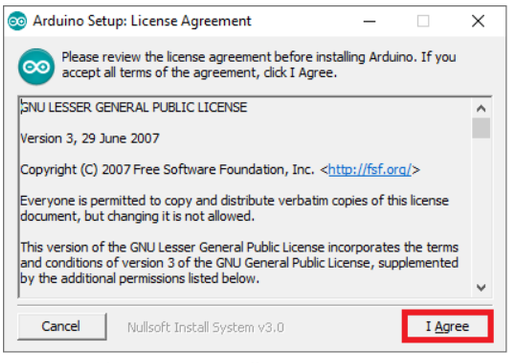


Figure 5: Accept licence agreement

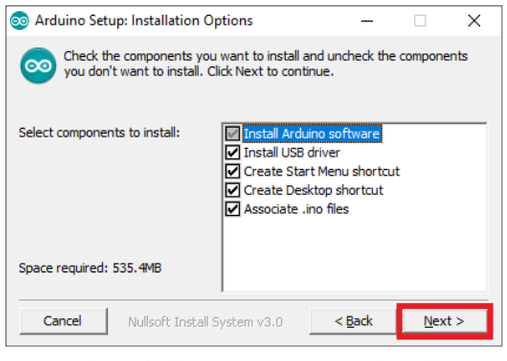


Figure 6: Choose components to install

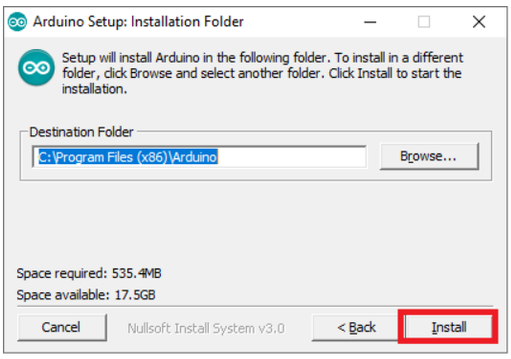


Figure 7: Choose installation folder

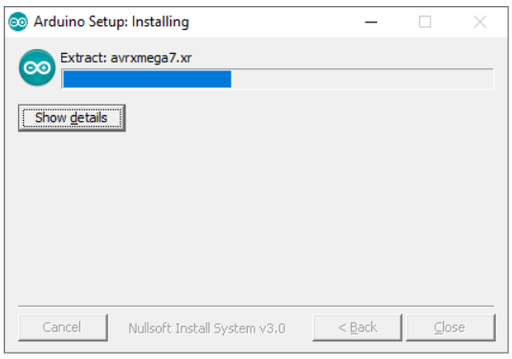


Figure 8: Installation progress

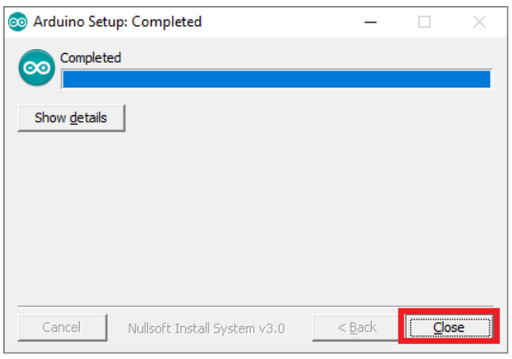


Figure 9: Click close button after installation has finished

You have installed the program. Now you can run the program by clicking two times on the Arduino icon which should be on your desktop.

Ovime je instalacija završena. Sada možete pokrenuti program tako da na radnoj površini dva puta kliknete na ikonu Arduino.

Ezzel a telepítés befejeződik. Most már futtathatja a programot, ha duplán kattint az asztalon az Arduino ikonra.

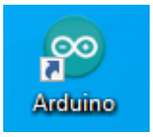


Figure 10: Running the program

1.2. Arduino integrated development environment introduction

1.2. Upoznavanje s Arduino razvojnom okolinom

1.2. Ismerkedés az Arduino fejlesztői környezettel

After running the program, you will see the Arduino integrated development environment starting screen with the empty program template.

Nakon pokretanja programa otvorit će se početni ekran Arduino integrirane razvojne okoline sa praznim predloškom programa.

A program elindítása után megnyílik az Arduino integrált fejlesztői környezet kezdőképernyője egy üres programsablonnal.

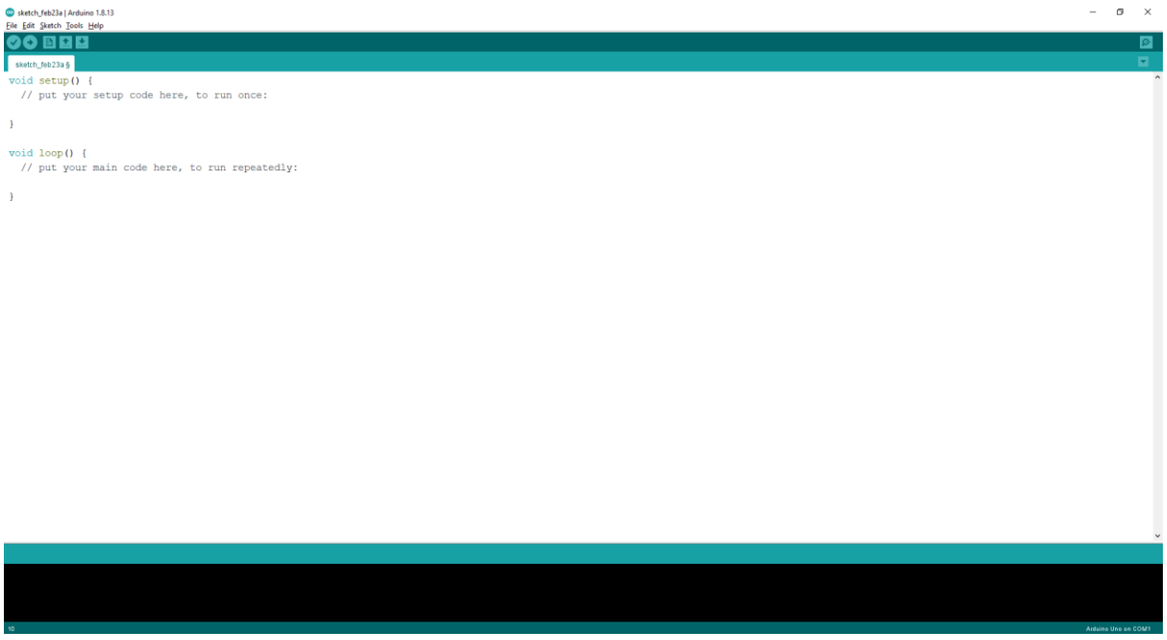


Figure 11: Arduino integrated development environment starting screen

This empty template contains two functions that are mandatory in every Arduino program:

- setup() – Program code that runs at the beginning of the program or when the program resets
- loop() – infinite loop for running the Arduino program

It is not possible to run the Arduino program unless we connect the Arduino device to a computer. A connection is established via a USB cable which is generally delivered with the Arduino device. To upload our program to the Arduino board, we have to set up the correct communication port between the computer and the Arduino device.

Prazni predložak sadrži samo dvije funkcije i to one koje mora imati svaki Arduino program:

- setup() – programski kod koji se izvodi na početku programa ili kad se program resetira
- loop() – beskonačna petlja unutar koje se izvodi Arduino program

Arduino program nije moguće izvesti ukoliko nismo priključili Arduino uređaj na računalo. Povezivanje se izvodi putem USB kabela koji u pravilu dolazi s Arduino uređajem. Kako bi se napisani programi ispravno prenijeli na Arduino pločicu, potrebno je podesiti ispravan port za komunikaciju između računala i Arduino uređaja.

Az üres sablon csak két funkciót tartalmaz, és azokat, amelyekkel minden Arduino programnak rendelkeznie kell:

- • setup() – programkód, amely a program elején vagy a program alaphelyzetbe állításakor kerül végrehajtásra
- • loop() – egy végtelen ciklus, amelyen belül az Arduino program végrehajtásra kerül

Az Arduino program végrehajtása nem lehetséges, ha nem csatlakoztattuk az Arduino eszközt a számítógéphez. A csatlakozás USB-kábellel történik, amely általában az Arduino eszközhöz tartozik. Az írott programok Arduino kártyára való helyes átviteléhez be kell állítani a megfelelő portot a számítógép és az Arduino eszköz közötti kommunikációhoz.

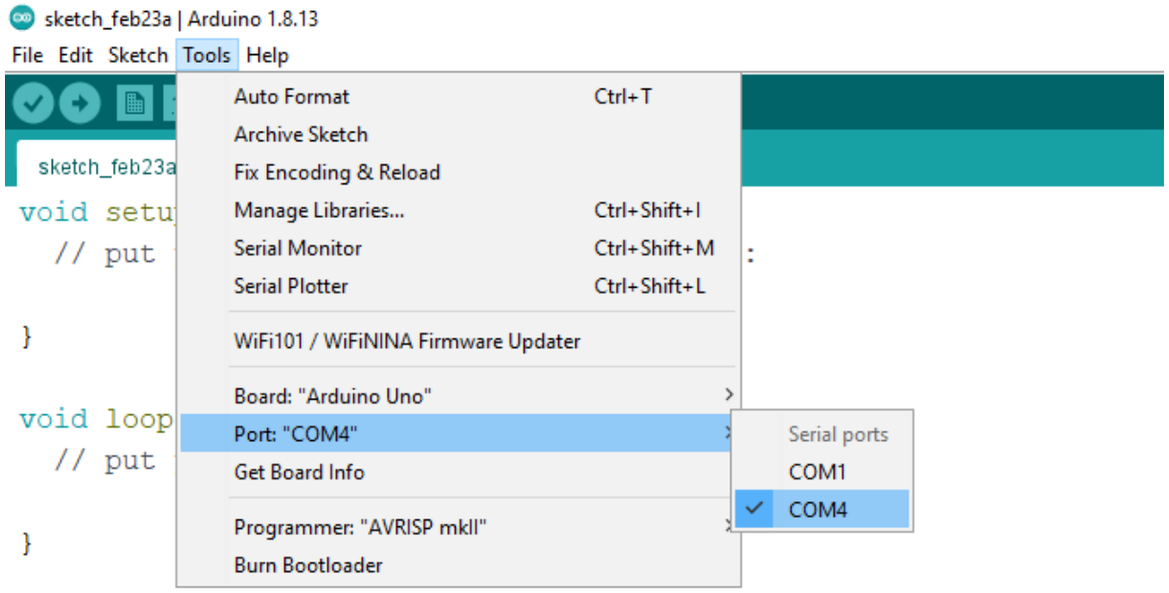


Figure 12: Setting up communication port

1.3. My first Arduino program – Hello World!

1.3. Moj prvi Arduino program – Hello World!

1.3. Az első Arduino programom – Hello World!

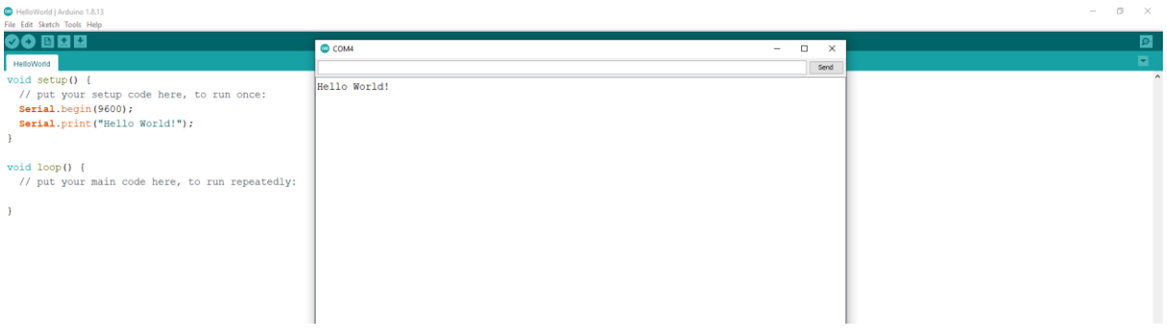


Figure 13: Hello World program!

Program writing steps:

1. In the void setup() function type the following code lines:

   Serial.begin(9600);
   Serial.print(“Hello World!“);

   Serial.begin(9600); instruction opens a serial port and sets the data transfer rate to 9600 bps
   Serial.print(“Hello World!“); instruction writes “Hello World” text on the serial port

2. After writing the program code, you need to click on the following three options in exact order

   
   Verify – compiles the code and checks it for errors. If the code has errors, you need to correct them before going into the next phase.

   
   Upload – uploads correctly compiled code on Arduino board. Wait till the upload is finished (follow the progress bar)!

   
   Serial monitor – opens up the communications port dialogue window in which you can see program output, or you can use it for data input

3. In the upper right corner of your dialogue window click the “send” button. The program result should be visible as it is shown in figure 13.

If we move Serial.print(“Hello World!“); instruction in the loop() part of the program, program will write its result inside the loop as it is shown in figure 14.

Postupak izrade programa:

1. Unutar void setup() funkcije napišite linije koda:

   Serial.begin(9600);
   Serial.print(“Hello World!“);

   Naredba Serial.begin(9600); otvara serijski port i postavlja brzinu prijenosa podataka na 9600 bps
   Naredba Serial.print(“Hello World!“); ispisuje tekst „Hello World“ na serijski port

2. Nakon što ste napisali kod, potrebno je kliknuti na sljedeće tri opcije redom

   
   Provjeri – prevodi kod i provjerava sadrži li kod greške. Ako kod sadrži greške, potrebno ih je ispraviti prije prelaska u sljedeću fazu

   
   Prenesi – prenosi ispravno preveden kod na Arduino pločicu. Pričekajte dok se prenošenje ne dovrši (pratite traku napretka) !

   
   Serial Monitor – otvara dijaloški okvir komunikacijskog porta na kojem možete pratiti ispis podataka, ili ga iskoristiti za unos podataka.

3. U gornjem desnom uglu dijaloškog okvira kliknite na opciju “Pošalji“. Rezultat programa bi trebao biti vidljiv kao što je prikazano na slici 13.

Ukoliko naredbu Serial.print(“Hello World!“); prebacimo u dio programa loop(), ispis će se izvoditi u petlji kao što je prikazano na slici 14.

A program elkészítésének folyamata:

1. 1. A void setup() függvénybe írja be a kódsorokat:

   Serial.begin(9600);
   Serial.print(“Hello World!“);

   Parancs Serial.begin(9600); megnyitja a soros portot és 9600 bps-ra állítja az adatátviteli sebességet
   Parancs Serial.print("Hello World!"); kiírja a "Hello World" szöveget a soros portra

2. Miután megírta a kódot, a következő három lehetőségre kell kattintania sorrendben

   
   Ellenőrzés – összeállítja a kódot, és ellenőrzi a kódot a hibákért. Ha a kód hibákat tartalmaz, azokat ki kell javítani, mielőtt a következő lépésre lépne

   
   Feltöltés – feltölti a helyesen összeállított kódot az Arduino táblára. Várja meg, amíg az átvitel befejeződik (kövesse a folyamatjelző sávot)!

   
   S Soros monitor – megnyit egy párbeszédpanelt a kommunikációs porton, ahol az adatkimenetet figyelheti, vagy adatbevitelre használhatja.

3. 2. A párbeszédpanel jobb felső sarkában kattintson a "Küldés" lehetőségre. A program kimenetének láthatónak kell lennie a 13. ábrán látható módon.

Amennyiben a Serial.print("Hello World!") parancsot átváltjuk loop() programba, a nyomtatás ciklusosan történik, ahogy az a 14. ábrán látható.

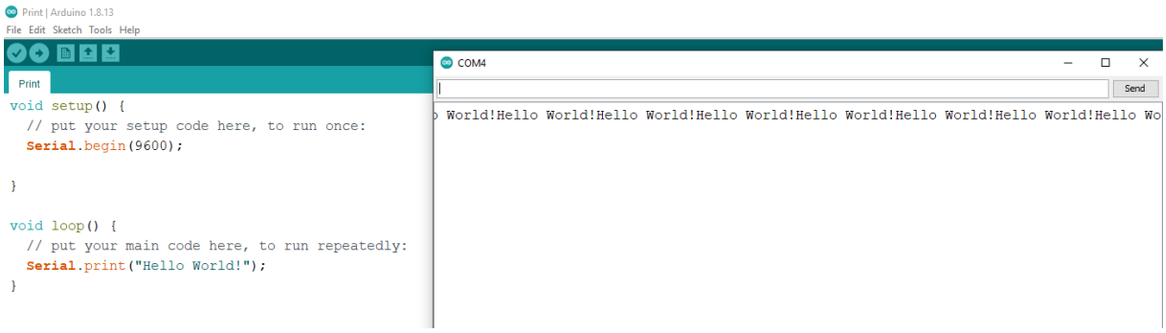


Figure 14: Hello World inside a loop

Figure 15 shows what happens if we use Serial.println() instead of Serial.print().

Na slici 15 je prikazan rezultat ispisa ukoliko umjesto Serial.print() koristimo naredbu Serial.println().

A 15. ábra a nyomtatás eredményét mutatja, ha a Serial.println() parancsot használjuk a Serial.print() helyett.

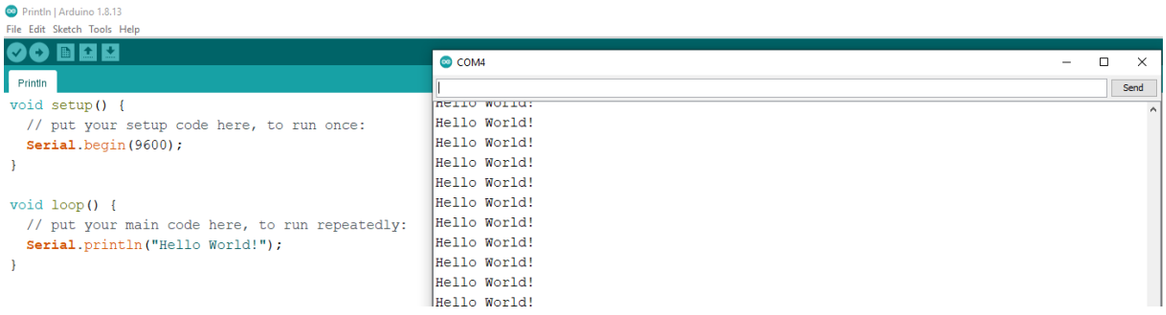


Figure 15: Hello World inside a loop with the Serial.println() instruction

1.4. Arithmetical and logical operations

1.4. Aritmetičke operacije i logičke operacije

1.4. Számtani és logikai műveletek

Arduino supports following arithmetical and logical operations

Arithmetical operations		**Logical operations**	
Operation	**Operator**	**Operation**	**Operator**
Addition	+	Logical AND	&&
Subtraction	-	Logical OR	||
Multiplication	*	Logical NOT	!
Division	/		
Modulo	%		

Arduino podržava sljedeće osnovne aritmetičke i logičke operacije:

Aritmetičke operacije		**Logičke operacije**	
Operacija	**Operator**	**Operacija**	**Operator**
Zbrajanje	+	Logičko I	&&
Oduzimanje	-	Logičko ILI	||
Množenje	*	Logičko NE	!
Dijeljenje	/		
Modulo	%		

Az Arduino a következő alapvető számtani és logikai műveleteket támogatja:

Számtani műveletek		**Logikai műveletek**	
Művelet	**Operátor**	**Művelet**	**Operátor**
Összead**ás**	+	Logikai ÉS	&&
Kivon**ás**	-	Logikai VAGY	||
Szorz**ás**	*	Logikai NEM	!
Oszt**ás**	/		
Sz**á**zalék	%		

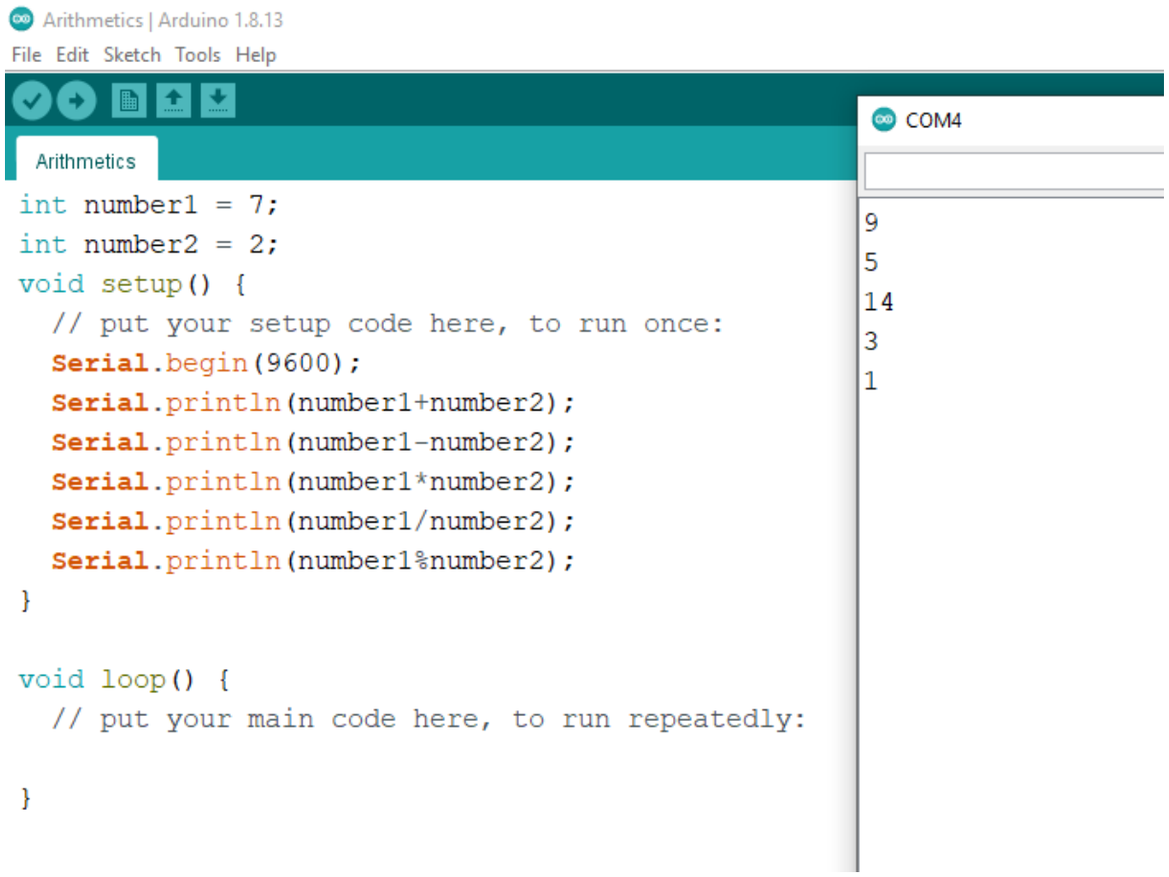


Figure 16: Arithmetical operations

Figure 16 shows the result of a program in which we created two integer variables (number1 and number2). With those two numbers, we perform the arithmetical operation from the table. Notice that before variables number1 and number2 are using the keyword int. It is a keyword for the numerical, integer data type.

Slika 16 prikazuje rezultat primjera u kojem smo kreirali dvije cjelobrojne varijable (broj1 i broj2) i nad njima obavili aritmetičke operacije spomenute u tablici. Primijetimo da je ispred varijabli broj1 i broj2 navedena ključna riječ int. To je oznaka za brojčani, cjelobrojni tip podataka.

A 16. ábra egy olyan példa eredményét mutatja, amelyben két egész változót (szám1 és szám2) hoztunk létre, és végrehajtottuk rajtuk a táblázatban említett számtani műveleteket. Vegyük észre, hogy az 1-es és 2-es számú változók előtt az int kulcsszó áll. Ez egy numerikus, egész számú adattípus címke.

1.5. Data types
1.5. Tipovi podataka
1.5. Adattípusok

The most commonly used data types which are used in Arduino programming are shown in the following table.

Data type	**Keyword**	**Value range**
Logical	boolean	true or false (0 or 1)
Byte	byte	From 0 till 255
Integer	int	From -32 768 till 32 767
Long integer	long	From -2 147 483 648 till 2 147 483 647
Real	float	From -3.4028235E+38 till 3.4028235E+38
Real with double precision	double	On most of Arduino devices (Uno, Mega and Nano), it has same precision as float (32 bits) On MKR1000 and Zero Arduino devices it allows double precision (64 bits)
Character	char	One ASCII character
Character array	String	Array of ASCII characters

Najčešće korišteni tipovi podataka koje koristimo u Arduino programiranju su prikazani u ovoj tablici.

Tip podatka	**Oznaka**	**Raspon vrijednosti**
Logički	boolean	true ili false (0 ili 1)
Bajt	byte	Od 0 do 255
Cjelobrojni	int	Od -32 768 do 32 767
Dugi cijeli broj	long	Od -2 147 483 648 do 2 147 483 647
Realni	float	Od -3.4028235E+38 do 3.4028235E+38
Realni dvostruke preciznosti	double	Na većini Arduino uređaja (Uno, Mega i Nano), jednaka preciznost kao i float (32 bita) Na MKR1000 i Zero Arduino uređajima omogućuje dvostruku preciznost (64 bita)
Znakovni	char	Jedan znak iz ASCII tablice
Niz znakova	String	Više znakova iz ASCII tablice

A leggyakrabban használt adattípusokat, amelyeket az Arduino programozásban használunk, a következő táblázat mutatja be.

Adattípus	**Címke**	**Értékek tartománya**
Logikai	boolean	true vagy false (0 vagy 1)
Bájt	byte	0-tól 255-ig
Egész szám	int	32 768-tól 32 767-ig
Hosszú egész szám	long	2 147 483 648-tól 2 147 483 647-ig
Reális	float	3.4028235E+38-tól 3.4028235E+38-ig
Dupla pontosság elérése	double	A legtöbb Arduino eszközön (Uno, Mega és Nano) ugyanaz a pontosság, mint a float (32 bit) Dupla pontosságot (64 bit) tesz lehetővé az MKR1000 és a Zero Arduino eszközökön
Jel	char	Egy karakter az ASCII táblázatból
Karakterek sora	String	További karakterek az ASCII táblázatból

The following example shows printing the variable content for each data type.

U sljedećem primjeru prikazan je ispis varijabli za svaki od navedenih tipova podataka.

A következő példa a felsorolt adattípusok változóinak kimenetét mutatja be.

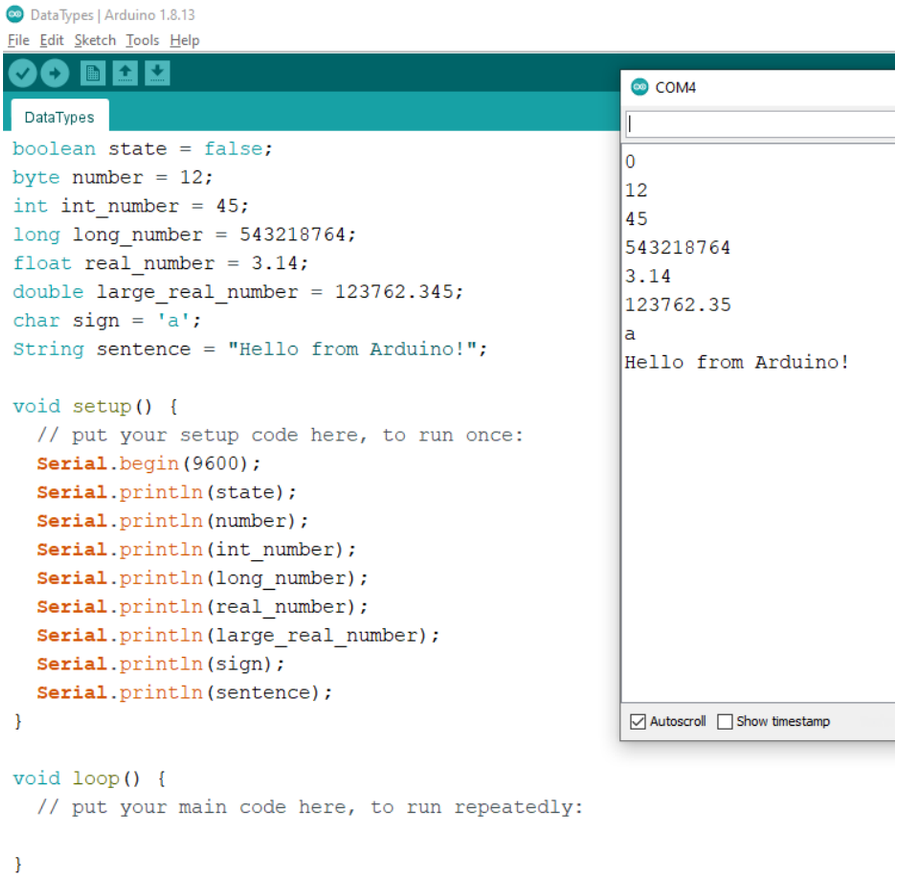


Figure 17: Data types

1.6. Serial monitor data reading
1.6. Čitanje podataka pomoću serial monitora
1.6. Adatok olvasása serial (soros) monitor segítségével

Till now we were using a serial port for printing data from the Arduino program. What if we want to use the Serial monitor console for reading data and use it as input parameters for the Arduino program? Figure 18 shows how we are going to achieve this goal.

Do sad smo serial port koristili samo za ispis podataka iz Arduino programa. Što ako želimo preko Serial monitor konzole učitati podatke u Arduino program kao ulazne parametre? Slika 18 nam prikazuje kako ćemo to napraviti.

Eddig csak az Arduino programból származó adatok nyomtatására használtuk a soros portot. Mi van, ha adatokat szeretnénk betölteni az Arduino programba bemeneti paraméterként a soros monitor konzolon keresztül? A 18. ábra bemutatja, hogyan fogjuk ezt megtenni.

Figure 18: Serial port data read

int number = 0;	Declares integer variable named number and initializes its value to 0. In this variable we will store data we will input by using a Serial monitor.
number = Serial.parseInt();	Every data which is read from the serial monitor is treated as a set of characters. If we want this data store as a numerical value we will "transform" it to a number by using the parseInt() function and assign this value to a variable named number.
Serial.println(number)	Prints assigned numerical value over the serial port on the serial monitor. In our case, we printed value 123.

Notice that the transformation of input data to a numerical value and its printing is wrapped inside the following instructions block:

```
if (Serial.available() > 0){

}
```

Now we will explain the meaning of the if instruction.

int broj = 0;	Naredba koja deklarira cjelobrojnu varijablu pod nazivom broj i inicijalizira ju na vrijednost 0. U varijablu broj ćemo spremiti podatak koji unesemo preko Serial monitora.
broj = Serial.parseInt();	Podatak koji se unosi preko serial monitora tretira se kao skup znakova. Ukoliko ga želimo pohraniti kao brojčanu vrijednost, pomoću funkcije parseInt() ga moramo „pretvoriti“ u cijeli broj i pridružiti varijabli broj.
Serial.println(broj:)	Ispisuje brojčanu vrijednost preko serial porta na serial monitor. U našem primjeru, upisali smo broj 123

Primijetimo da je pretvaranje ulaznog podatka u cijeli broj i njegov ispis „zamotano“ u sljedeći blok naredbi:

```
if (Serial.available() > 0){

}
```

U nastavku ćemo objasniti značenje naredbe if.

int szám = 0;	Parancs, amely deklarál egy szám nevű egész változót és inicializálja azt 0 értékre. A számváltozóba mentjük a Serial monitoron keresztül bevitt adatokat.
szám = Serial.parseInt();	A soros monitoron keresztül bevitt adatokat a rendszer karakterkészletként kezeli. Ha numerikus értékként szeretnénk tárolni, akkor a parseInt() függvény segítségével egész számmá kell "konvertálni" és hozzáadni a számváltozóhoz.
Serial.println(broj:)	Számértéket nyomtat a soros porton keresztül a soros monitorra. Példánkban a 123-as számot írtuk be

Figyeljük meg, hogy a bemeneti adatok egész számmá való konvertálása és kinyomtatása a következő parancsblokkban van "csomagolva":

```
if (Serial.available() > 0){

}
```

Az alábbiakban elmagyarázzuk az if parancs jelentését.

1.7. Conditional statements

1.7. Naredbe grananja

1.7. Elágazó parancsok

Conditional statements are used in programming when we want to perform one of two possible tasks – testing or selection. In the previous example, before we can use the serial port, we want to check if the serial port is available. For this purpose, we use if conditional statement in the form of testing.

1.7.1 If Else conditional statement

Let's try solving the following problem: we want to check if the number entered in our Arduino program is positive, negative, or zero. There are two approaches to solving this problem:

First approach – If conditional statement	**Second approach – If Else conditional statement**

If number > 0 then
  Print("Number is positive!")
If number < 0 then
  Print("Number is negative!")
If number = 0 then
  Print("Number is 0!")
	If number > 0 then
	  Print("Number is positive!")
	Else If number < 0 then
	  Print("Number is negative!")
	Else
	  Print("Number is 0!")

Grananje u programiranju koristimo onda kad želimo napraviti jednu od dvije stvari – obaviti testiranje ili omogućiti izbor. U prethodnom primjeru prije nego radimo sa serial portom, želimo provjeriti da li je on dostupan. U tu svrhu iskoristili smo if grananje.

1.7.1 If Else grananje

Pokušajmo riješiti sljedeći problem: za svaki uneseni broj naš Arduino program mora utvrditi je li broj pozitivan, negativan ili jednak nuli. Postoje dva pristupa rješenju:

Prvi pristup – naredba If	**Drugi pristup – naredba If Else**

Ako je broj > 0 onda
  Ispiši("Broj je pozitivan!")
Ako je broj < 0 onda
  Ispiši("Broj je negativan!")
Ako je broj = 0 onda
  Ispiši("Unijeli ste 0!")
	Ako je broj > 0 onda
	  Ispiši("Broj je pozitivan!")
	Inače Ako je broj < 0 onda
	  Ispiši("Broj je negativan!")
	Inače
	  Ispiši("Unijeli ste 0!")

Az elágazást akkor használjuk a programozásban, ha két dolog egyikét szeretnénk végrehajtani – tesztelni vagy kiválasztani. Az előző példában, mielőtt a soros porttal dolgoznánk, ellenőrizni akarjuk, hogy elérhető-e. Erre a célra az if elágazást használtuk.

1.7.1 If Else elágazás

Próbáljuk meg megoldani a következő problémát: Arduino programunknak minden egyes beírt számhoz meg kell határoznia, hogy a szám pozitív, negatív vagy nullával egyenlő. A megoldásnak két megközelítése van:

Első megközelítés – If parancs	**Második megközelítés – If Else parancs**

Ha a szám > 0 akkor
  Írd ki(“A szám pozitív!“)
Ha a szám < 0 akkor
  Írd ki(“A szám negatív!“)
Ha a szám = 0 akkor
  Írd ki(“A szám 0-val egyenlő!“)
	Ha a szám > 0 akkor
	  Írd ki (“Broj je pozitivan!“)
	Egyébként ha a szám < 0 akkor
	  Írd ki(“A szám negatív!“)
	Egyébként
	  Írd ki(“ 0 írtunk be!“)

By using the first approach we always do all three checks. If the user enters number 5, the program concludes that the number is positive. Although we have concluded that the number is positive, if the program is written in the first approach it will perform additional checks if the number is negative or equal to zero, which is inefficient.

The second approach works differently: if we enter number 5, the conclusion is that the number is positive so we can safely skip additional checks. If we enter -7, the first check concludes that the number isn't positive, the second check concludes that the number is negative so we don't need to do the third check. Worst case scenario – if we enter 0, the first check concludes that the number isn't positive, the second check concludes that the number isn't negative so the only logical explanation is that the number equals zero. By using the first approach we always do all three checks. By using the second approach in the best case scenario one check is enough, while all three checks are done only in the worst-case scenario.

Prvi pristup uvijek radi sve tri provjere. Ako je korisnik unio broj 5, program je utvrdio da je broj pozitivan. Bez obzira na to, program pisan prema prvom pristupu će svejedno provjeriti je li 5 negativan, odnosno je li jednak nuli, što je neefikasno.

Drugi pristup radi ovako: ukoliko smo unijeli 5, zaključili smo da je broj pozitivan pa ostale provjere možemo preskočiti. Ukoliko smo unijeli -7, prva provjera je utvrdila da broj nije pozitivan, druga provjera je utvrdila da je broj negativan pa treću provjeru preskačemo. Najgori slučaj – ako unesemo 0, prvom provjerom ustanovimo da broj nije pozitivan, nakon druge provjere ustanovili smo da broj nije negativan pa je jedino logično objašnjenje da je broj nula. Prvim pristupom uvijek radimo sve tri provjere. Drugim pristupom u najboljem slučaju moramo napraviti samo jednu provjeru, dok se tri provjere rade samo u najgorem slučaju.

Az első megközelítés mindig mindhárom ellenőrzést elvégzi. Ha a felhasználó beírta az 5-ös számot, a program megállapította, hogy a szám pozitív. Ettől függetlenül az első megközelítés szerint megírt program továbbra is ellenőrzi, hogy az 5 negatív-e, azaz egyenlő-e nullával, ami nem hatékony.

A második megközelítés a következőképpen működik: ha 5-öt írtunk be, akkor arra a következtetésre jutottunk, hogy a szám pozitív, így kihagyhatjuk a többi ellenőrzést. Ha -7-et írtunk be, akkor az első ellenőrzés megállapította, hogy a szám nem pozitív, a második ellenőrzés azt találta, hogy a szám negatív, ezért a harmadik ellenőrzést kihagyjuk. A legrosszabb eset – ha 0-t írunk be, az első ellenőrzéskor azt tapasztaljuk, hogy a szám nem pozitív, a második ellenőrzés után azt tapasztaljuk, hogy a szám nem negatív, így az egyetlen logikus magyarázat, hogy a szám nulla. Mindhárom ellenőrzést mindig az első megközelítéssel végezzük. A második megközelítésnél a legjobb esetben csak egy ellenőrzést kell elvégeznünk, míg három ellenőrzést csak a legrosszabb esetben.

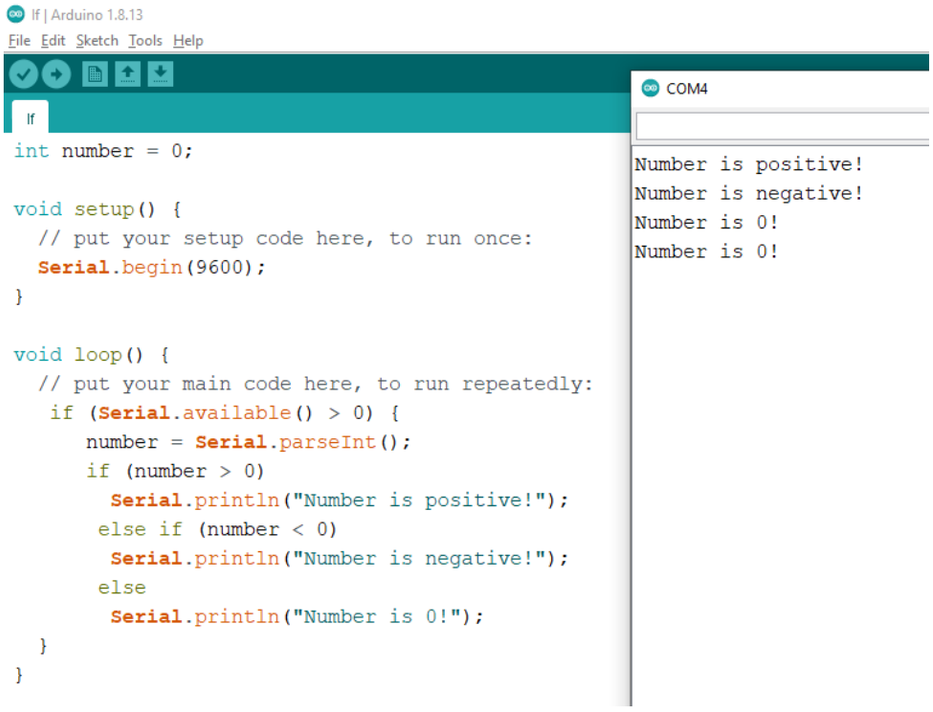


Figure 19: If Else conditional statement

1.7.2. Switch case conditional statement

1.7.2. Switch case grananje

1.7.2. Switch case elágazás

Assuming we know exactly the values we can expect as input data in our program and if we have a designated set of instructions for each of the input parameters we will use Switch Case conditional statement. Such a conditional statement is much more efficient in the realization of multiple selections.

Ukoliko znamo točne vrijednosti koje korisnik može unijeti u naš program i za svaku od njih imamo predviđenu određenu naredbu, koristimo Switch Case grananje. Takvo grananje je puno efikasnije prilikom realizacije višestrukih izbornika.

Ha pontosan ismerjük azokat az értékeket, amelyeket a felhasználó beírhat a programunkba, és mindegyikhez külön parancsunk van, akkor a Switch Case elágazást használjuk. Az ilyen elágazás sokkal hatékonyabb több menü megvalósítása esetén.

Figure 20: Switch Case conditional statement

In our example expected input values are 1, 2, and 3. The program will read those values and prints them in textual form. If the user enters any other value except those which are expected, the program runs the default block. In other words:

In case of user input 1

Print (“One“)

In case of user input 2

Print (“Two“)

In case of user input 3

Print (“Three“)

For all other cases

print (“Unallowed value!“)

Break instruction allows fast exit from the switch block after the assigned instruction is complete. In this way, we skip the checking of unnecessary cases.

U našem primjeru korisnik može unijeti 1, 2 ili 3, a program ispisuje unesenu vrijednost u tekstualnoj formi. Ako korisnik unese bilo koju drugu vrijednost osim očekivanih, izvodi se default blok. Drugim riječima:

U slučaju da je korisnik unio 1

Ispiši (“Jedan“)

U slučaju da je korisnik unio 2

Ispiši (“Dva“)

U slučaju da je korisnik unio 3

Ispiši (“Tri“)

Za sve ostale slučajeve:

Ispiši (“Nedozvoljena vrijednost“)

Naredba break omogućuje brzi izlaz iz switch strukture nakon što se izvela naredba, kako se ne bi nepotrebno provjeravale ostale grane.

Példánkban a felhasználó beírhat 1-et, 2-t vagy 3-at, és a program a beírt értéket szöveges formában írja ki. Ha a felhasználó a várttól eltérő értéket ad meg, akkor az alapértelmezett blokk kerül végrehajtásra. Más szavakkal:

Abban az esetben, ha a felhasználó 1-t írt be

Írd ki (“Egy“)

Abban az esetben, ha a felhasználó 2-t írt be

Írd ki (“Kettő“)

Abban az esetben, ha a felhasználó 3-t írt be

Írd ki (“Három“)

Minden más esetben:

Írd ki (“Nem megengedett érték“)

A break parancs lehetővé teszi a gyors kilépést a kapcsolószerkezetből a parancs végrehajtása után, így a többi ágat nem kell szükségtelenül ellenőrizni.

1.8. Programming loops
1.8. Programske petlje
1.8. Programhurkok

Loops are used for repeating certain instructions multiple times. If the number of repetitions is known to us, we will use for loop. If the number of repetitions is unknown, we will use a while loop.

Petlje služe za ponavljanje naredbi određeni broj puta. Ako nam je broj ponavljanja unaprijed poznat, koristimo for petlju. Ako broj ponavljanja nije poznat, koristimo while petlju.

A hurkok a parancsok bizonyos számú ismétlésére szolgálnak. Ha az ismétlések száma előre ismert, akkor for ciklust használunk. Ha az ismétlések száma nem ismert, akkor a while ciklust használjuk.

1.8.1 For loop
1.8.1 Petlja for
1.8.1 For ciklus

When using for loop number of repetitions is known and in every step of the loop, we can tell how many steps are till the loop finishes. We know this because of one specific variable we call counter. The following examples will demonstrate a general form of for loop and concrete example for printing numbers from 1 till n. If the user inputs value 3, the program will print values 1, 2, and 3. In our example, we will input number 10. The counter variable in this example is named i.

```
For (initial_counter_value; final_counter_value; counter_increment)
    Instruction;
```

Za petlju for uvijek znamo broj ponavljanja i u svakom koraku znamo koliko smo blizu završetku petlje. Razlog tome je korištenje posebne varijable koja se naziva brojač. U nastavku je prikazan opći oblik for petlje, i konkretan primjer koji ispisuje brojeve od 1 do n. Ako korisnik unese 3, ispisat će se brojevi 1,2,3. U našem primjeru, unijeli smo broj 10. brojač čiju vrijednost ispisujemo predstavlja varijabla i.

```
For (početna_vrijednost_brojača; završna_vrijednost_brojača; promjena_brojača)
    Naredba;
```

A for ciklusnál mindig tudjuk az ismétlések számát, és minden lépésnél tudjuk, milyen közel vagyunk a ciklus végéhez. Ennek oka egy speciális változó, az úgynevezett számláló használata. Az alábbiakban látható a for ciklus általános formája, és egy konkrét példa, amely kiírja a számokat 1-től n-ig. Ha a felhasználó 3-at ír be, akkor az 1,2,3 számok kerülnek kinyomtatásra. Példánkban 10-et írtunk be. A számlálót, amelynek értékét kiírjuk, az i változó képviseli.

```
For (számláló_kezdő_értéke; számláló_befejező_értéke; számláló_változása)
    Parancs;
```

Figure 21: for loop

1.8.2 While loop

1.8.2 Petlja while

1.8.2 While ciklus

Unlike for loop, while loop is a conditional loop which means we do not know the number of repetitions in advance. This is best explained in the following way:

While (repetition condition is met)

Run instruction

Sooner or later this loop will stop. When? When we input such data which will break the repeating condition. The following example reads numbers from the serial port and prints them on the serial monitor. While loop works until the input value is zero. Number zero breaks the condition for repeating while loop. However, since this portion of code is within an infinite loop called loop(), the program will continue its work until we turn off the Arduino device.

Za razliku od for petlje, while je uvjetna petlja za koju unaprijed ne znamo broj ponavljanja. Najlakše ćemo ju objasniti na sljedeći način:

Dok je (uvjet ispunjen)

Izvodi naredbu

Petlja će prije ili kasnije stati s radom. Kada? Onda kad se pojavi takav podatak koji prekida uvjet za ostanak u petlji. U nastavku je primjer koji čita brojeve sa serial porta i ispisuje ih na serial monitor. Petlja while radi sve dok se ne unese broj 0. Brojka nula prekida petlju, međutim, s obzirom da je taj dio programa unutar beskonačne petlje loop() koja izvodi program sve dok je Arduino uređaj uključen, program nastavlja s radom, ispočetka.

A for ciklustól eltérően a while egy feltételes ciklus, amelynek nem tudjuk előre az ismétlések számát. Ennek legegyszerűbb módja a következő:

Amíg (a feltétel teljesül)

Hajtsd végre a parancsot

A ciklus előbb-utóbb leáll. Mikor? Aztán amikor olyan információ jelenik meg, amely megszegi a ciklusban maradás feltételét. Az alábbiakban látható egy példa, amely beolvassa a számokat a soros portról, és kinyomtatja a soros monitorra. A while ciklus a 0 szám beviteléig fut. A nulla szám megszakítja a ciklust, de mivel a programnak ez a része egy végtelen ciklus() cikluson belül van, amely addig futja a programot, amíg az Arduino eszköz be van kapcsolva, a program tovább fut, a kezdetektől fogva.

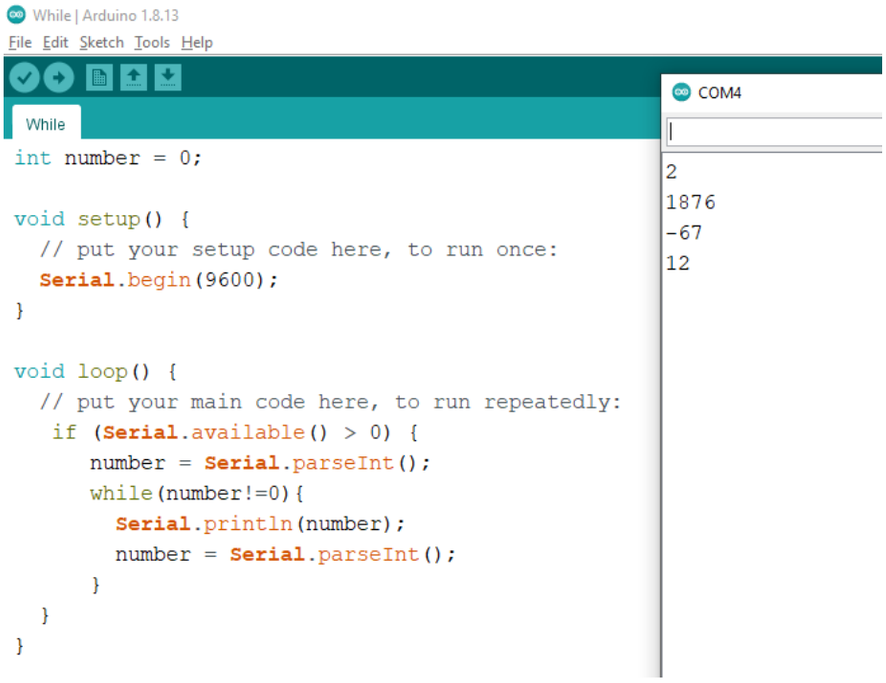


Figure 22: While loop

1.9. Libraries

1.9. Korištenje knjižnica (libraries)

1.9. Könyvtár használata (libraries)

The Arduino IDE (development environment) includes numerous examples of program code, however if we want to use some additional elements (e.g. sensors, screens, etc.) we have to write the code so that the microcontroller can communicate with such elements. This is sometimes not easy and takes time, but even for that there is already a ready-made code that someone made and which is free (Arduino community), so we can include it in the form of a library (library) in our own code.

To turn on the library, click on: Sketch> Include Library> Add .ZIP Library…

Select the .zip file (or unzipped folder), click OK!

Arduino IDE (razvojno okruženje) uključuje brojne primjere programskoga koda, međutim ako želimo koristiti neke dodatne elemente (npr. senzori, ekrani i sl.) moramo napisati kod tako da mikrokontroler može s takvim elementima komunicirati. To nije ponekad jednostavno i zahtijeva vremena, no i za to postoji već gotov kod koji je netko napravio i koji je besplatan (Arduino community), te ga možemo u obliku knjižnica (libraries) uključiti u vlastiti kod.

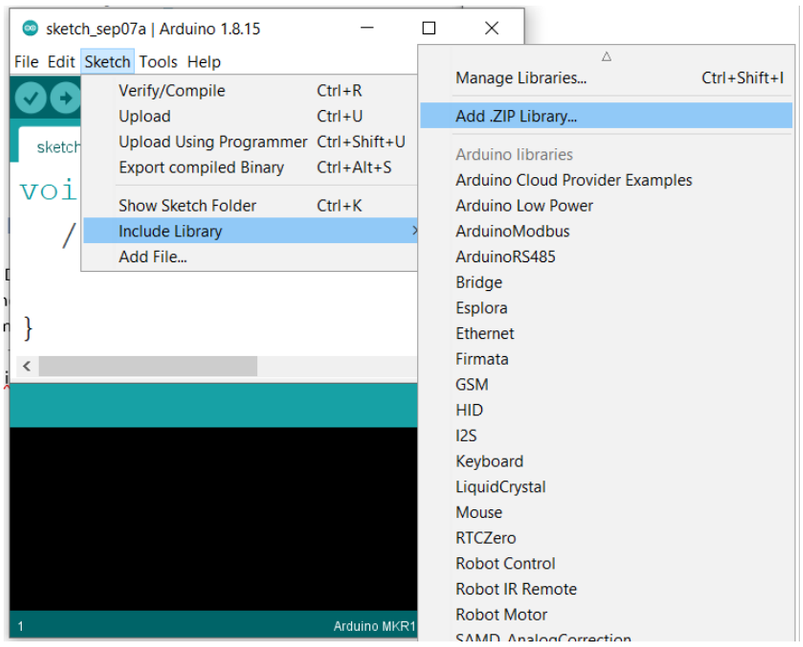
Za uključivanje knjižnica kliknemo na: *Sketch > Include Library > Add .ZIP Library…*

Odaberemo .zip datoteku (ili raspakiranu mapu), te kliknemo OK!

Az Arduino IDE (fejlesztői környezet) számos példát tartalmaz programkódra, azonban ha további elemeket (pl. szenzorok, képernyők stb.) szeretnénk használni, akkor meg kell írnunk a kódot, hogy a mikrokontroller kommunikálni tudjon ezekkel az elemekkel. Ez néha nem egyszerű és időigényes, de erre már van kész kód, amit valaki készített, és ez ingyenes (Arduino közösség), amit könyvtárak formájában beépíthetünk a saját kódunkba.

A könyvtárak felvételéhez kattintson a következő gombra: *Sketch > Include Library > Add .ZIP Library…*

Válassza ki a .zip fájlt (vagy kicsomagolt mappát), és kattintson az OK gombra!



2. Android

Android is a mobile operating system based on a modified version of the Linux kernel and other open source software, designed primarily for touchscreen mobile devices such as smartphones and tablets. Android is developed by a consortium of developers known as the Open Handset Alliance and commercially sponsored by Google.[3] During this course, we will program in a way we will connect a mobile device with the Arduino device. After we completed Arduino programming, we will look into the programming for Android. For this purpose, we will be using the App Inventor tool which was developed by the American MIT institute.

Android je otvoreni operacijski sustav za mobilne uređaje, kao što su pametni telefoni i tablet računala, američke tvrtke Google Inc. temeljen na jezgri Linux i drugom softveru otvorenog kôda.[4] Mi ćemo se baviti programiranjem mobilnih uređaja na način da povežemo mobilni uređaj sa Arduino uređajem. Nakon što smo obradili Arduino programiranje, sada ćemo se pozabaviti programiranjem za Android. U tu svrhu koristit ćemo alat App Inventor for Android koji je razvijen na Američkom institutu MIT.

Az Android egy nyílt operációs rendszer mobil eszközökhöz, például okostelefonokhoz és táblagépekhez, az amerikai Google Inc. cégtől, amely Linux kernelen és más nyílt forráskódú szoftvereken alapul. A mobil eszközök programozásával fogunk foglalkozni oly módon, hogy a mobileszközt az Arduino készülékhez csatlakoztassuk. Most, hogy foglalkoztunk az Arduino programozással, az Android programozásával fogunk foglalkozni. Erre a célra az App Inventor for Android eszközt fogjuk használni, amelyet az American Institute MIT-ben fejlesztettek ki.

2.1. MIT App Inventor

MIT App Inventor is an Integrated Development Environment that works as a Web application. It is a program you don't need to install. All you need to do is log in to a web application by Google account. However, the App Inventor program doesn't run on the desktop computer or the web, it runs on a mobile phone. Because of that, the program needs to be uploaded on a mobile device or run in the emulator. We will explain both approaches.

MIT App Inventor je integrirana razvojna okolina koja se koristi kao Web aplikacija. Sam App Inventor nije potrebno instalirati, već se dovoljno prijaviti na Web aplikaciju pomoću Google korisničkog računa. Međutim, program napisan u App Inventoru se ne izvodi na stolnom računalu ili na Webu, već na mobilnom uređaju. Zbog toga je potrebno napisani program ili prebaciti na mobilni uređaj, ili pokrenuti u simulatoru Android uređaja kojeg instaliramo na stolno računalu. U nastavku će biti objašnjena oba pristupa.

Az MIT App Inventor egy integrált fejlesztői környezet, amelyet webalkalmazásként használnak. Nem szükséges magát az App Inventort telepíteni, de elegendő egy Google felhasználói fiókkal bejelentkezni a webalkalmazásba. Az App Inventorban írt program azonban nem asztali számítógépen vagy a weben fut, hanem mobileszközön. Emiatt vagy át kell vinni az írott programot egy mobil eszközre, vagy le kell futtatni egy Android készülék szimulátorban, amit asztali számítógépre telepítünk. Az alábbiakban mindkét megközelítést ismertetjük.

[3] https://en.wikipedia.org/wiki/Android_(operating_system)
[4] https://hr.wikipedia.org/wiki/Android_(operacijski_sustav)

2.2. Installation of App Inventor Android Emulator

2.2. Instalacija Android APP Inventor emulatora

2.2. Az Android APP Inventor emulator telepítése

Downloading and installing the emulator begins with a web browser opening and typing “app inventor download” keywords in Google search. By clicking on a link, we will download the installation file. The whole procedure is explained with screenshots.

Preuzimanje i instalacija emulatora počinje tako da otvorimo web preglednik i u Google tražilicu upišemo pojam „app inventor download“. Pokretanjem linkova preuzet će se instalacijska datoteka koja će instalirati emulator na računalo. Postupak je objašnjen u sljedećim slikama.

Az emulátor letöltése és telepítése úgy kezdődik, hogy megnyit egy webböngészőt, és beírja az „app inventor download” kifejezést a Google keresőjébe. A hivatkozások futtatása letölti a telepítőfájlt, amely telepíti az emulátort a számítógépre. Az eljárást a következő ábrák ismertetik.

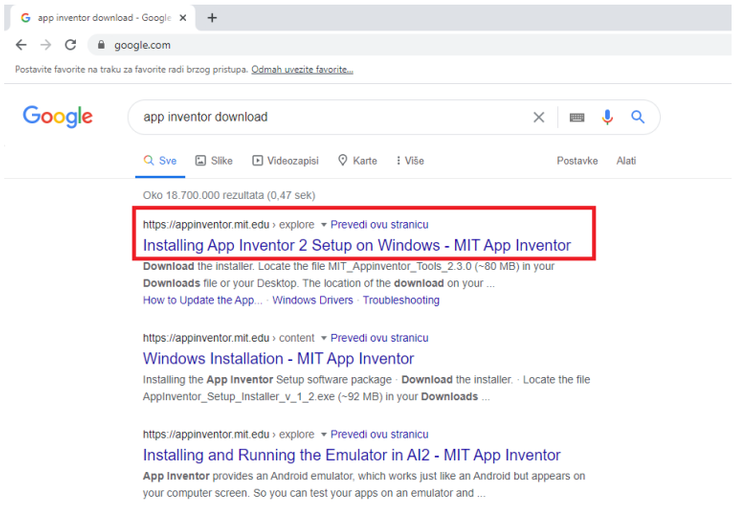


Figure 23: App Inventor emulator downloading

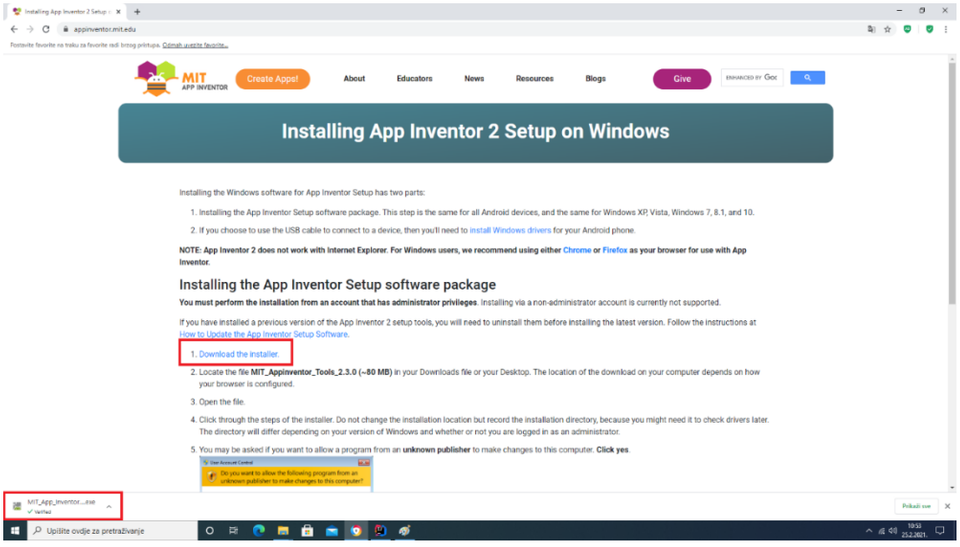


Figure 24: App Inventor emulator downloading

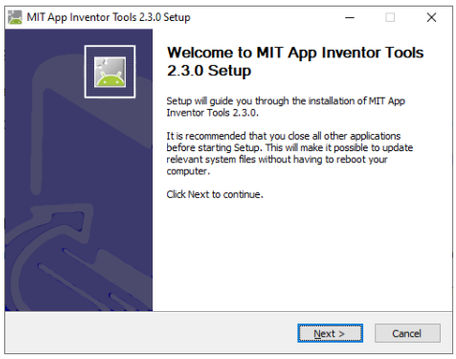


Figure 25: Running an installation

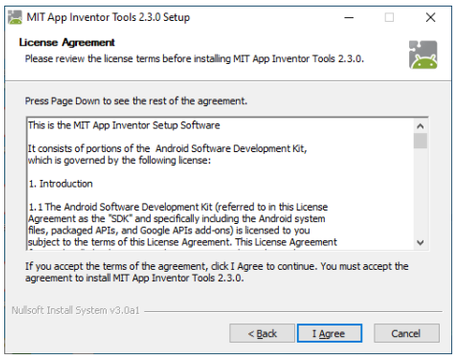


Figure 26: Accept the license agreement

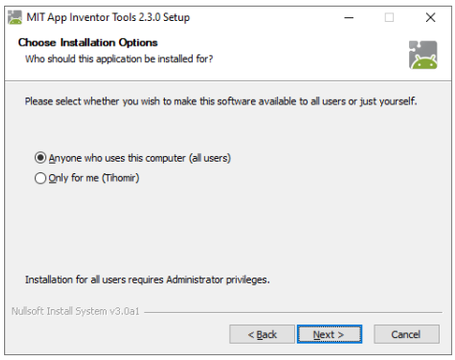


Figure 27: Users selection

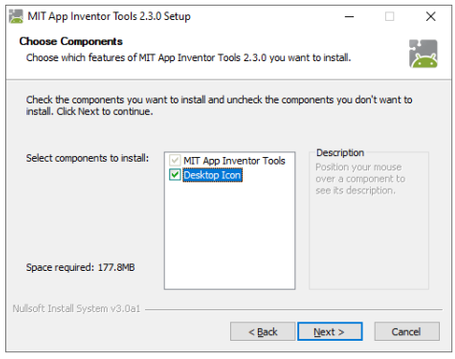


Figure 28: Choose installation components

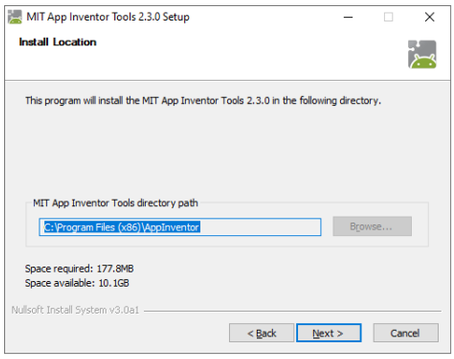


Figure 29: Choose an installation path

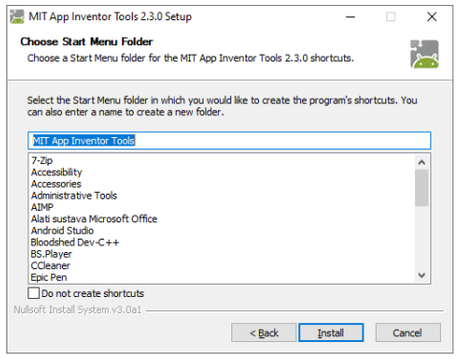


Figure 30: Choose a name of the Start menu folder

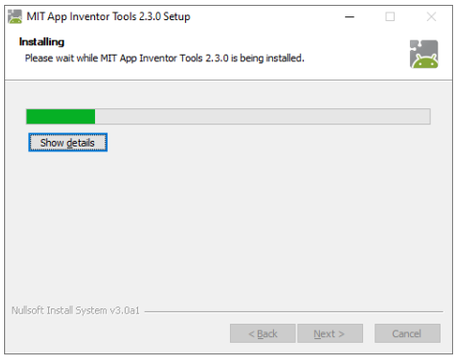


Figure 31: Installation progress

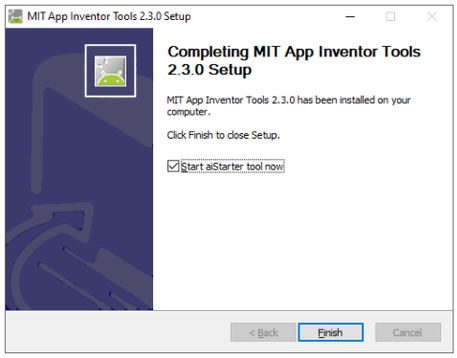


Figure 32: Installation complete

2.3. My first Android program – Hello World!

2.3. Moj prvi Android program – Hello World!

2.3. Az első Android programom – Hello World!

As early mentioned, we will use the App inventor environment as a web application. We will run a web browser, type in App inventor, follow the links and log in by using a Gmail user account.

Kao što je ranije navedeno, App Inventor razvojnu okolinu koristimo kao Web aplikaciju. Pokrenuti ćemo Web preglednik, upisati App Inventor, slijediti linkove i prijaviti se u App Inventor pomoću Gmail korisničkog računa.

Mint korábban említettük, az App Inventor fejlesztői környezetet webalkalmazásként használjuk. Elindítunk egy webböngészőt, beírjuk az App Inventor szót, követjük a linkeket, és Gmail-fiókkal bejelentkezünk az App Inventorba.

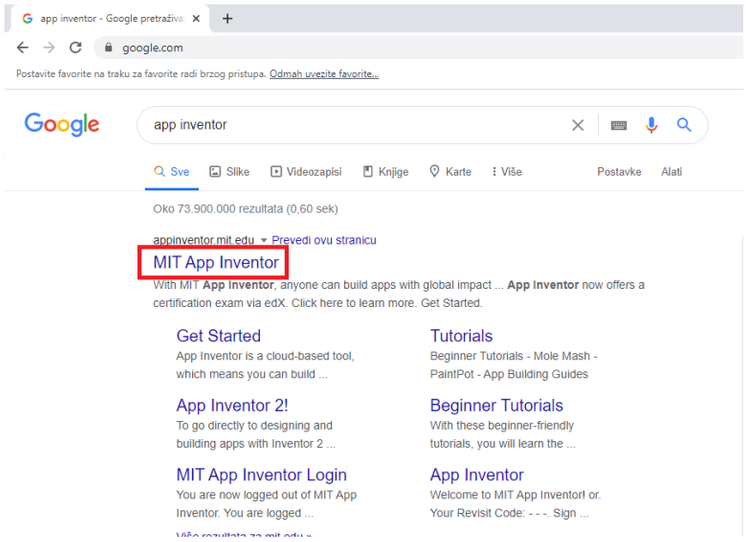


Figure 33: App inventor login

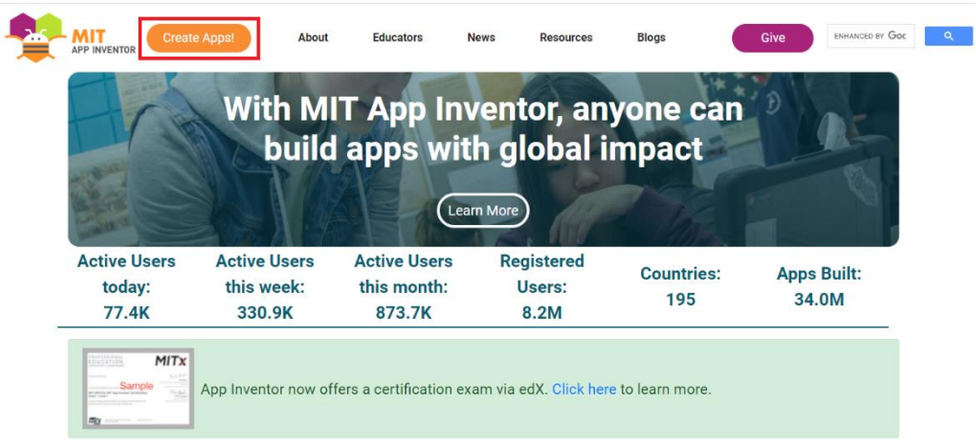


Figure 34: App inventor login

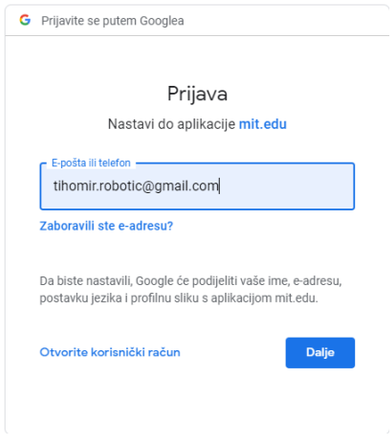


Figure 35: Login by Gmail account

After successful login, you will see the App Inventor starting screen. After the first login, you will also see the App Inventor welcome screen which offers guides for creating new projects. Also, there is an option to create a new empty project. From now on we will be creating new projects by clicking on the "Start new project" option in the upper right corner of the screen.

Nakon prijave pokrenuti će se početni ekran App Inventor web aplikacije. Prilikom prve prijave prikazati će se ekran dobrodošlice koji nudi pomoć pri izradi jednostavnih aplikacija, ili mogućnost kreiranja novog praznog projekta. Ubuduće, novi projekt ćemo kreirati pomoću „Start new project“ opcije prikazane u gornjem lijevom uglu ekrana.

Bejelentkezés után elindul az App Inventor webalkalmazás kezdőképernyője. Amikor először jelentkezik be, megjelenik egy üdvözlő képernyő, amely segítséget nyújt egyszerű alkalmazások létrehozásához, vagy új üres projekt létrehozásához. A jövőben a képernyő bal felső sarkában látható "Új projekt indítása" opcióval új projektet hozunk létre.

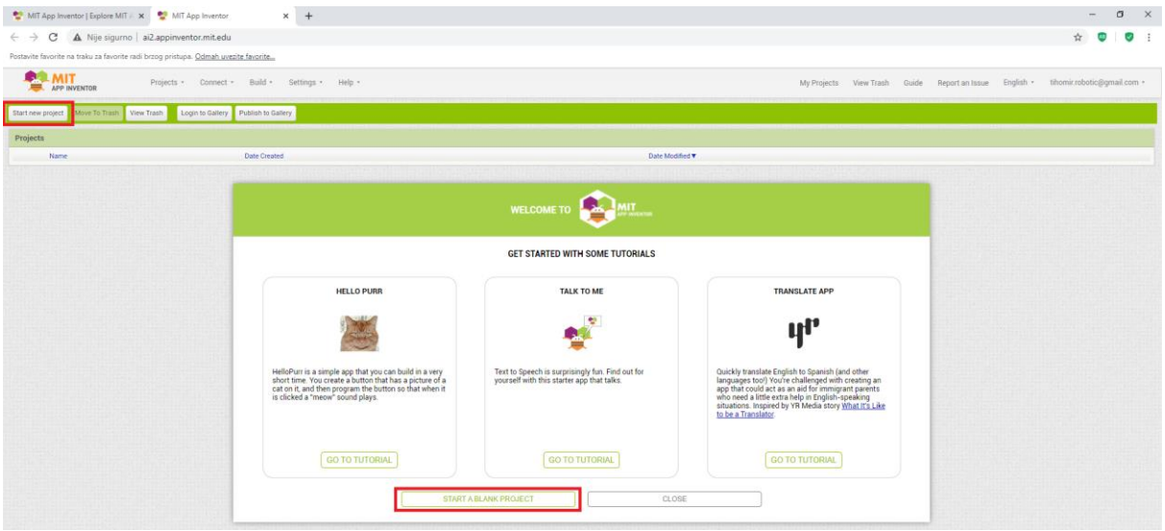


Figure 36: App Inventor welcome screen

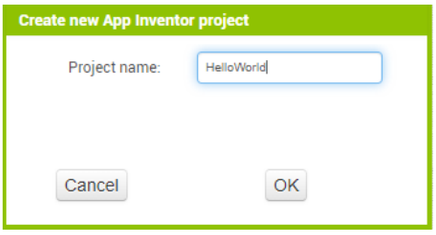


Figure 37: Create new App Inventor project

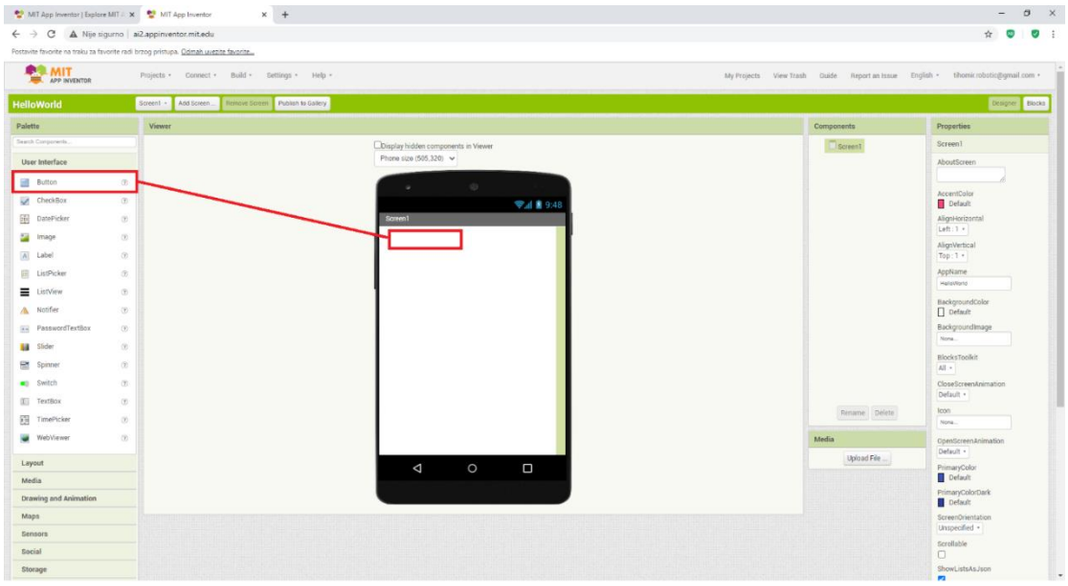


Figure 38: Project development environment

After creating a project named "Hello World" standard App inventor project development environment is opened. Our first project will consist of a single button that will show the "Hello World" message on click. The first step is, of course, adding a button on the application layout as shown in the previous figure. Using meaningful names for application controls is highly recommended. In a complex project names like Button1, Button2, Button3, etc. will quickly become confusing and inefficient. The following figures will show the procedure of renaming controls.

Nakon kreiranja prvog projekta pod nazivom „Hello World“ otvara se standardna radna okolina App Inventora u kojoj ćemo izrađivati naše projekte. Naš prvi projekt će se sastojati od jednog gumba koji će ispisivati poruku „Hello World“ nakon što ga pritisnemo. Prvi korak je naravno, dodavanje gumba na ekran kao što je prikazano na prethodnoj slici. U projektima je poželjno koristiti smislena pravila za imenovanje kontrola koje dodajemo u projekt. U složenim projektima Button1, Button2, Button3 itd. će ubrzo postati zbunjujuće i neefikasno. U nastavku je prikazan postupak preimenovanja kontroli.

Az első „Hello World” projekt létrehozása után megnyílik az App Inventor standard munkakörnyezete, amelyben elkészítjük projektjeinket. Első projektünk egyetlen gombból fog állni, amely megnyomásakor kinyomtatja a „Hello World” üzenetet. Az első lépés természetesen egy gomb hozzáadása a képernyőhöz az előző képen látható módon. A projektekben célszerű értelmes szabályokat használni a projekthez hozzáadott vezérlők elnevezésére. Az összetett projektekben a Button1, Button2, Button3 stb. hamarosan zavaróvá és hatástalanná válik. A vezérlő átnevezésének eljárása alább látható.

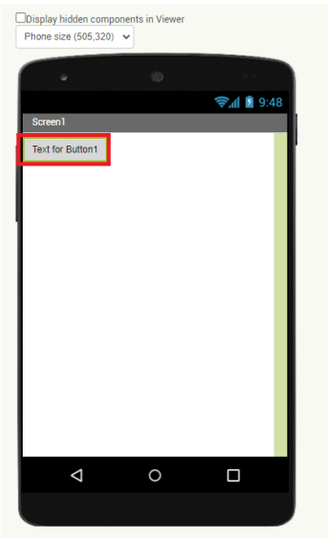


Figure 39: Control selection

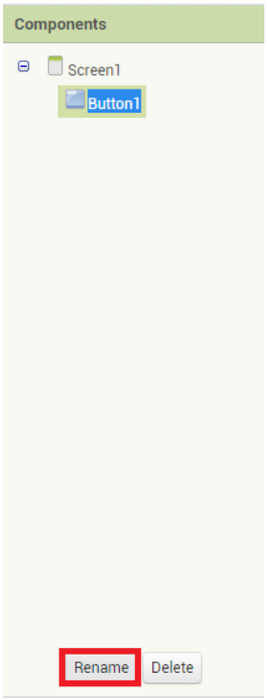


Figure 40: Control renaming

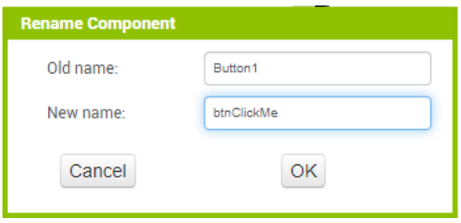


Figure 41: Setting up a new name

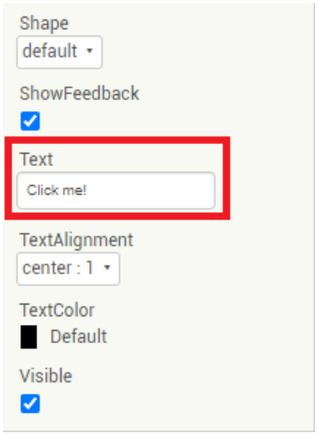


Figure 42: Setting up a control text

To see a "Hello World" message, a button is not enough. If we want to see the message, we require another control that is called Notifier. Notifier control can be dragged and dropped on the application layout, in the same manner, we did with the button. Notice that the Notifier is not visible on the application layout.

Za ispis poruke „Hello World“ gumb nije dovoljan. Potrebno je dodati još jednu kontrolu koja se zove Notifier. Kontrolu Notifier ćemo također odvući na ekran korisničkog sučelja, isto kao što smo napravili sa gumbom. Primijetimo da kontrola Notifier nije vidljiva na korisničkom sučelju.

Egy gomb nem elég a „Hello World” üzenet kinyomtatásához. Szükséges egy másik Notifier nevű vezérlő hozzáadása. A Notifier vezérlőt is áthúzzuk a felhasználói felület képernyőjére, ugyanúgy, mint a gombbal. Figyelje meg, hogy a Notifier vezérlőelem nem látható a felhasználói felületen.

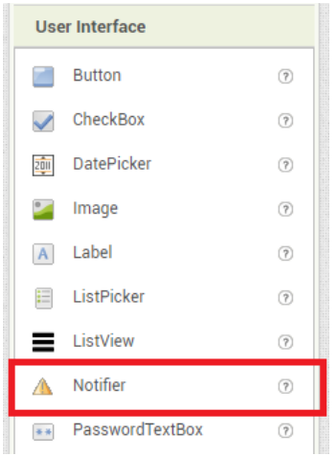


Figure 43: Adding the Notifier control

Figure 44: Notifier is added, but not visible on the layout

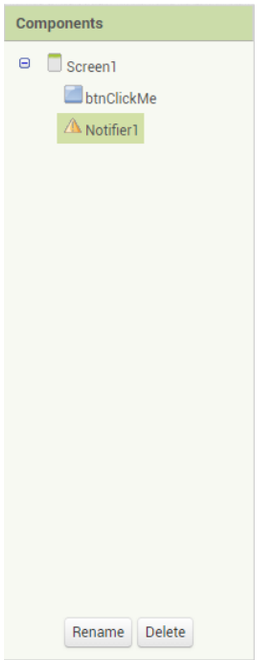


Figure 45: Notifier renaming

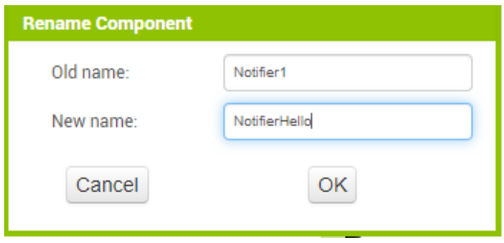


Figure 46: Choosing a new name for Notifier

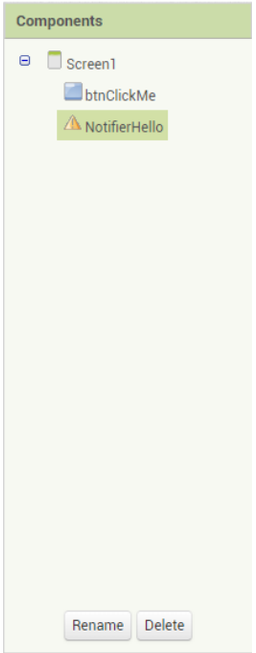


Figure 47: Renamed Notifier

Now as the required controls are added to the layout, we can start programming their functionality. For that, we need to switch from designer mode to programming by clicking the "Blocks" button.

Dodali smo potrebne kontrole na korisničko sučelje. Da bismo počeli sa programiranjem njihovih funkcionalnosti, potrebno je iz dizajnerskog načina rada preći u programerski. To ćemo postići tako da pritisnemo na gumbić „Blocks“.

A felhasználói felülethez hozzáadtuk a szükséges vezérlőket. A funkcióik programozásának megkezdéséhez át kell váltani a tervezési módból a programozási módba. Ezt a "Blocks" gomb megnyomásával érjük el.

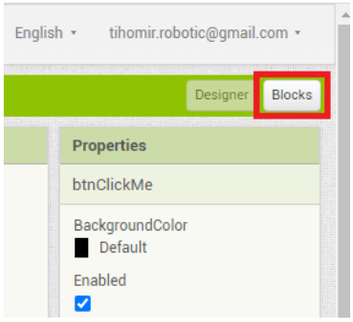


Figure 48: Switching to programming mode

Now we need to connect btnClickMe button with NotifierHello Notifier. We will also add a text that will Notifier display on the screen.

Dodali smo potrebne kontrole na korisničko sučelje. Da bismo počeli sa programiranjem njihovih funkcionalnosti, potrebno je iz dizajnerskog načina rada preći u programerski. To ćemo postići tako da pritisnemo na gumbić „Blocks“.

Most össze kell kötni a btnClickMe gombot a NotifierHello vezérlővel, amiben felvesszük a képernyőre nyomtatandó szöveget.

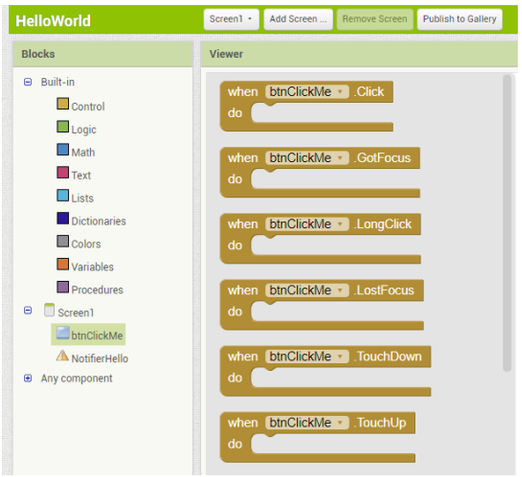


Figure 49: On btnClikcMe button add when btnClickme click function

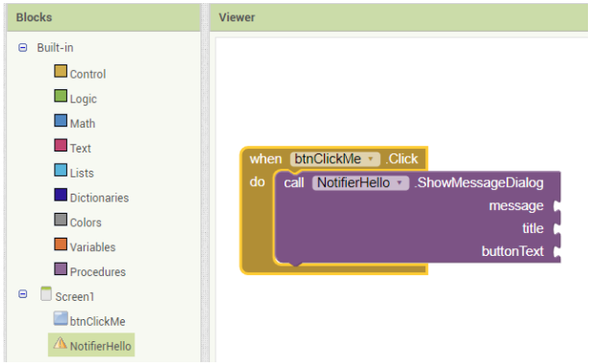


Figure 50: For Notifier, choose ShowMessageDialog function and insert it within the Notifier

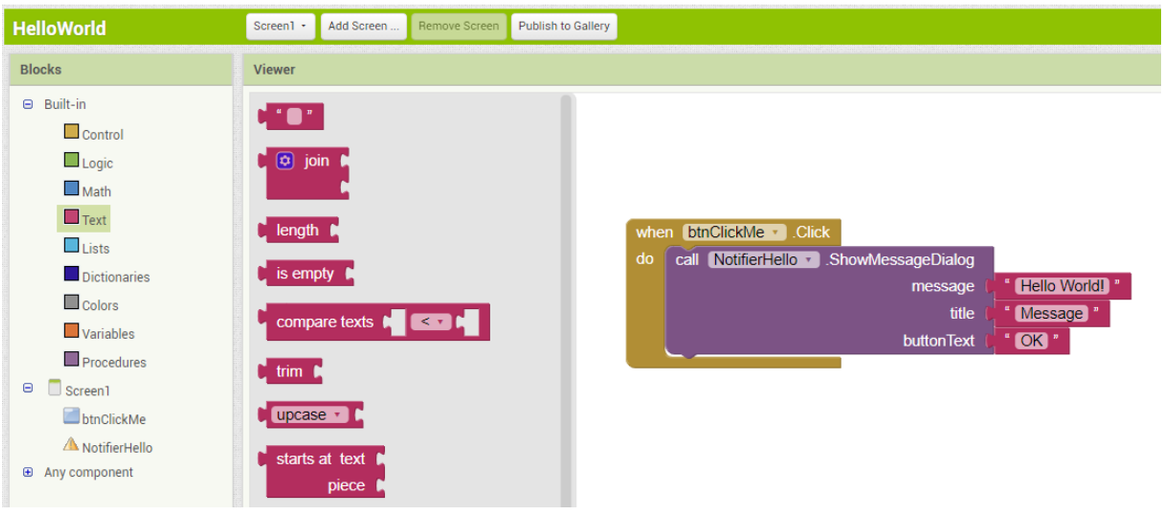


Figure 51: Add text message to a Notifier

Notifier is a function with 3 arguments: message - the message notifier displays, title - a title shown on the Notifier window, buttonText - a text for closing the notifier. Set up the message by adding text controls as the figure shows.

Now we will load our program in the emulator and run it. First, we run the aiStarter application that is necessary for running the emulator. The icon should be on a desktop.

Notifier je funkcija koja ima tri argumenta: message – poruka koja se ispisuje, title – naslov prozora za prikaz obavijesti, buttonText – tekst koji se prikazuje na gumbiću za zatvaranje obavijesti. Podesimo poruke tako da dodamo pripadne tekstualne kontrole kao što je prikazano na slici.

Sada ćemo naš program učitati u emulator i izvesti. Najprije pokrenemo aplikaciju koja je potrebna za pokretanje emulator dvoklikom na ikonu aiStarter koja bi se trebala nalaziti na radnoj površini.

A Notifier egy olyan függvény, amelynek három argumentuma van: message - a kinyomtatott üzenet, title - az értesítést megjelenítő ablak címe, button text - az értesítést lezáró gombon megjelenő szöveg. Tegyük testre az üzeneteket a megfelelő szöveges vezérlők hozzáadásával a képen látható módon.

Most betöltjük a programunkat az emulátorba, és futtatjuk. Először is elindítjuk az emulátor indításához szükséges alkalmazást az aiStarter ikonra való dupla kattintással, amelynek az asztalon kell lennie.

Figure 52: Starting up the emulator

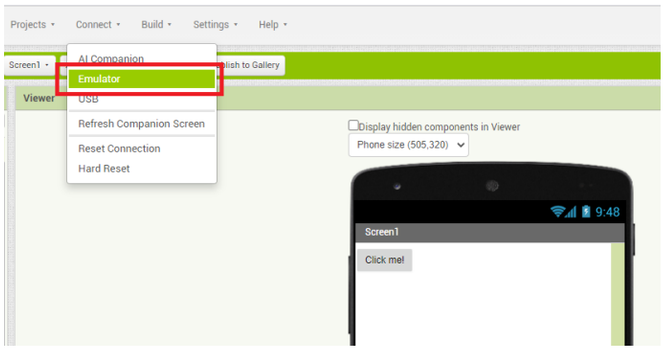


Figure 53: Loading the application in the emulator

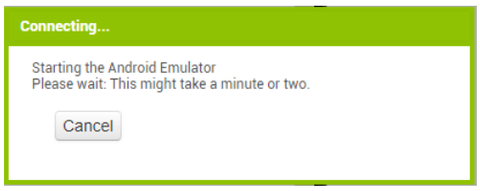


Figure 54: Starting up the emulator

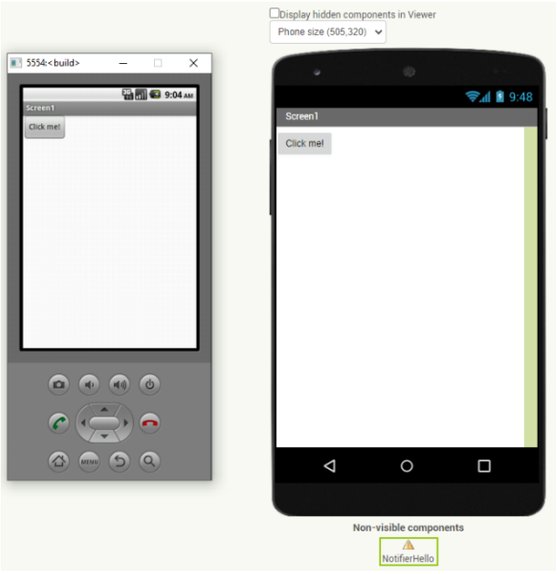


Figure 55: Emulator with our loaded application

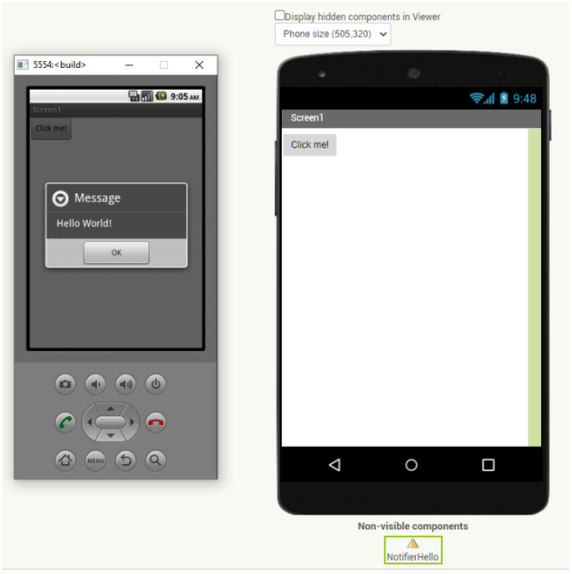


Figure 56: Running the application inside the emulator

2.4. App Inventor connection methods

In a previous chapter, we learned how to run the App Inventor program inside an emulator. The emulator is not a real mobile device, it is merely a program that behaves like a mobile device and is used for testing. There are three ways of loading the App Inventor program on a mobile device. All of them are visible in the connect menu:

1. AI Companion
2. Emulator
3. USB

AI Companion is used for loading a program to a mobile device over a WiFi network by using MIT AI2 Companion application. This application must be installed on a mobile phone. Application and tutorial are available on the link: https://appinventor.mit.edu/explore/ai2/setup-device-wifi.html.

Working with the emulator is explained in the previous chapter. Emulator and tutorial are available on the link: https://appinventor.mit.edu/explore/ai2/setup-emulator.html.

Direct connection via USB cable requires a bit more steps. The complete procedure is explained at https://appinventor.mit.edu/explore/ai2/setup-device-usb.html. To connect App Inventor with a mobile device via USB cable it is required to:

- Install App Inventor software used by an emulator - the procedure is explained in chapter 2.2
- Install AI Companion
- Turn the USB debugging on the mobile phone (developer options) - whole procedure is explained at https://developer.android.com/studio/debug/dev-options
- Install Android drivers on the computer - follow the link https://software.intel.com/content/www/us/en/develop/articles/intel-usb-driver-for-android-devices.html

2.4. Načini povezivanja App Inventora s mobilnim uređajima

U ranijem poglavlju naučili smo kako se naš Android program može isprobati u emulatoru. Emulator nije fizički mobilni uređaj, već samo program koji „glumi“ mobilni uređaj i služi za testiranje. Postoje 3 načina kako možemo učitati program iz App Inventora u mobilni uređaj i ti načini su vidljivi u izborniku Connect:

1. AI Companion
2. Emulator
3. USB

AI Companion služi za učitavanje programa u mobilni uređaj preko WiFi mreže uporabom MIT AI2 Companion aplikacije koju je potrebno preuzeti i instalirati na mobilni uređaj. Aplikacija i postupak su dostupni na poveznici https://appinventor.mit.edu/explore/ai2/setup-device-wifi.html.

Rad s emulatorom je detaljno objašnjen u prethodnom poglavlju. Postupak i emulator su dostupni na poveznici https://appinventor.mit.edu/explore/ai2/setup-emulator.html.

Direktno povezivanje preko USB-a zahtjeva nešto više koraka. Kompletni postupak objašnjen je na poveznici https://appinventor.mit.edu/explore/ai2/setup-device-usb.html. Od preduvjeta za povezivanje APP Inventora s mobilnim uređajem preko USB kabla potrebno je:

- Instalirati App Inventor softver koji koristi emulator – postupak objašnjen u poglavlju 2.2
- Instalirati ranije spomenuti AI Companion
- Uključiti USB ispravljanje grešaka na mobilnom uređaju (opcije za razvojne inženjere) – detaljan postupak objašnjen na https://developer.android.com/studio/debug/dev-options
- Instalirati upravljačke programe na računalo za povezivanje s Android uređajem preko USB-a – poveznica https://software.intel.com/content/www/us/en/develop/articles/intel-usb-driver-for-android-devices.html

2.4. Az App Inventor mobileszközökhöz való csatlakoztatásának módjai

Az előző fejezetben megtudtuk, hogyan tesztelhető az Android programunk emulátorban. Az emulátor nem fizikai mobileszköz, hanem csak egy mobileszközt "személyesítő" program, amelyet tesztelésre használnak. Háromféleképpen tölthetünk be egy programot az App Inventorból mobileszközre, és ezek a módok láthatók a Csatlakozás menüben:

1. AI Companion
2. Emulator
3. USB

Az AI Companion az MIT AI2 Companion alkalmazás segítségével tölthető be programok mobileszközre WiFi hálózaton keresztül, amelyet le kell tölteni és telepíteni kell a mobileszközre. A jelentkezés és az eljárás a linken érhető el: https://appinventor.mit.edu/explore/ai2/setup-device-wifi.html.

Az emulátorral való munkavégzés részletes leírása az előző fejezetben található. Az eljárás és az emulátor elérhető ezen a linken: https://appinventor.mit.edu/explore/ai2/setup-emulator.html.

Az USB-n keresztüli közvetlen csatlakozáshoz még néhány lépésre van szükség. A teljes eljárás leírása ezen a linken található: https://appinventor.mit.edu/explore/ai2/setup-device-usb.html. Az APP Inventor mobileszközhöz USB-kábellel történő csatlakoztatásának előfeltételei:

- Telepítse az emulátort használó App Inventor szoftvert – a 2.2 fejezetben ismertetett eljárás szerint
- Telepítse a korábban említett AI Companiont
- USB-hibakeresés engedélyezése mobileszközön (Fejlesztői beállítások) – a részletes eljárás leírása: https://developer.android.com/studio/debug/dev-options
- Telepítse a vezérlőprogramokat a számítógépre, hogy USB-n keresztül csatlakozzon az Android-eszközhöz – link: https://software.intel.com/content/www/us/en/develop/articles/intel-usb-driver-for-android-devices.html

2.5. APK file generating

2.5. Generiranje APK datoteke

2.5. APK fájl generálása

Described methods for connecting App Inventor with the mobile device will load the application on a mobile phone, but only inside the MIT Appinventor 2 test application. In other words, our application "exists" on a mobile phone as long as the connection between App Inventor and mobile phone is active. If we want to permanently load our application on a mobile phone, we need to generate apk file and load it on the mobile device. Apk file will be permanently loaded on the mobile device by using AI Companion¸application. (AI Companion must be installed on mobile device).

Opisani načini povezivanja APP Inventora s mobilnim uređajem će učitati aplikaciju na mobilni uređaj ali samo unutar MIT Appinventor 2 test aplikacije. Drugim riječima, aplikacija postoji na mobilnom uređaju sve dok je veza između mobilnog uređaja i App Inventora aktivna. Ako želimo da aplikacija trajno ostane pohranjena u mobilnom uređaju, potrebno je generirati takozvanu apk datoteku i prebaciti ju na mobilni uređaj. Apk datoteku ćemo trajno učitati na mobilni uređaj uporabom aplikacije AI Companion koja mora biti instalirana na mobilni uređaj.

Az APP Inventor mobileszközhöz való csatlakoztatásának leírt módszerei betöltik az alkalmazást a mobileszközre, de csak az MIT Appinventor 2 tesztalkalmazáson belül. Más szavakkal, az alkalmazás mindaddig létezik a mobileszközön, amíg a mobileszköz és az App Inventor közötti kapcsolat aktív. Ha azt szeretnénk, hogy az alkalmazás tartósan a mobileszközön maradjon, akkor létre kell hozni egy úgynevezett apk fájlt, és át kell vinni a mobil eszközre. Az Apk fájlt véglegesen feltöltjük a mobileszközre az AI Companion alkalmazás segítségével, amelyet telepíteni kell a mobileszközre.

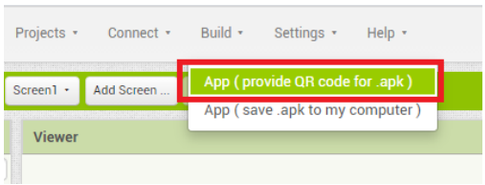


Figure 57: Apk file generating

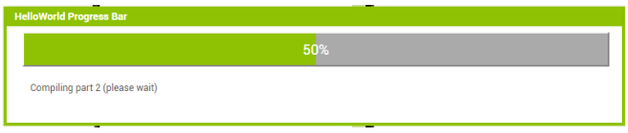


Figure 58: Apk file generating progress bar

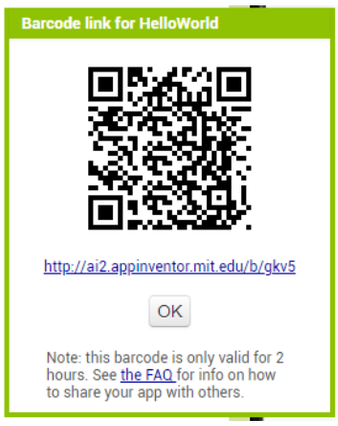


Figure 59: Apk QR code

Figure 60: QR code scanning by using the "scan QR code" option inside AI Companion application

Figure 61: Running the "Hello World" application on a real mobile phone

2.6. The Calculator application

2.6. Aplikacija kalkulator

2.6. Aplikáció alkalmazás

One of the most commonly used applications for programming learning is the calculator. There are a lot of calculators we can download from the App Inventor Gallery. To access the gallery, we will clock on the option Login to Gallery.

Jedna od najčešće korištenih aplikacija za učenje programiranja jest kalkulator. Postoje brojni gotovi kalkulatori koje možemo preuzeti iz galerije gotovih programa za App Inventor. Galeriji se pristupa pomoću opcije Login to Gallery.

A programozás tanulására az egyik leggyakrabban használt alkalmazás a kalkulátor. Számos kész alkalmazás létezik, amelyeket letölthetünk az App Inventor kész programok galériájából. A galéria a Login the Gallery opcióval érhető el.

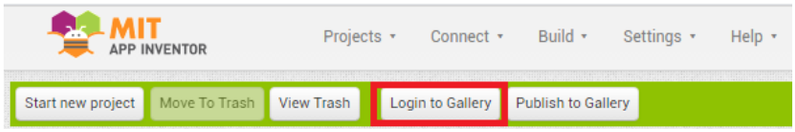


Figure 62: App Inventor Gallery

After clicking on Login to Gallery, we will log in by using our Gmail account. On the "search" bar we will type the name of the application we are searching (for instance Calculator) and type the Search button. We can load the selected application by clicking the Load App Into MIT App Inventor option.

For a calculator user interface, we will need the following controls:

- Buttons (b0, b1, b2, b_plus, etc)
- Textbox (display result line)

For a calculator programming logic, in the Blocks part we will need the following programming blocks:

Nakon toga, logiramo se pomoću Gmail korisničkog računa, u područje za pretragu upišemo okvirni naziv aplikacije koju tražimo (npr. calculator) i pritisnemo tipku search. Odabranu aplikaciju učitamo kao naš projekt pomoću opcije Load App Into MIT App Inventor.

Za izradu korisničkog sučelja kalkulatora potrebne su nam sljedeće kontrole:

- Gumbići (b0, b1, b2, b_plus, itd)
- Textbox (Linije za prikaz rezultata)

Za izradu kalkulatora u dijelu Blocks, potrebni su nam sljedeći programski blokovi:

Ezt követően a Gmail felhasználói fiókunkkal bejelentkezünk, a keresőmezőbe beírjuk a keresett alkalmazás (pl. calculator) kísérleti nevét, majd megnyomjuk a keresés gombot. A kiválasztott alkalmazást a Load App Into MIT App Inventor opcióval töltjük be saját projektünkként.

Az alkalmazás felhasználói felületének létrehozásához a következő vezérlőkre van szükségünk:

- Gombok (b0, b1, b2, b_plus, stb.)
- Textbox (Sorok az eredmények megjelenítéséhez)

Az alkalmazáslétrehozásához a Blokkok részben a következő programblokkra van szükségünk:

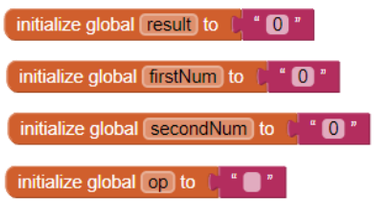


Figure 63: Variables declaration

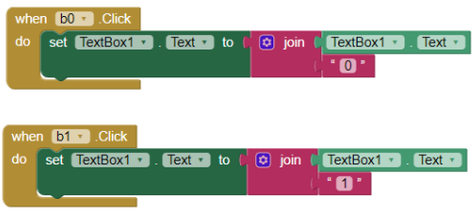


Figure 64: Number input buttons

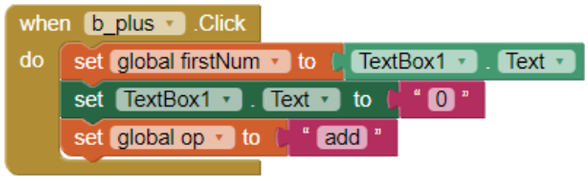


Figure 65: Arithmetical operations buttons

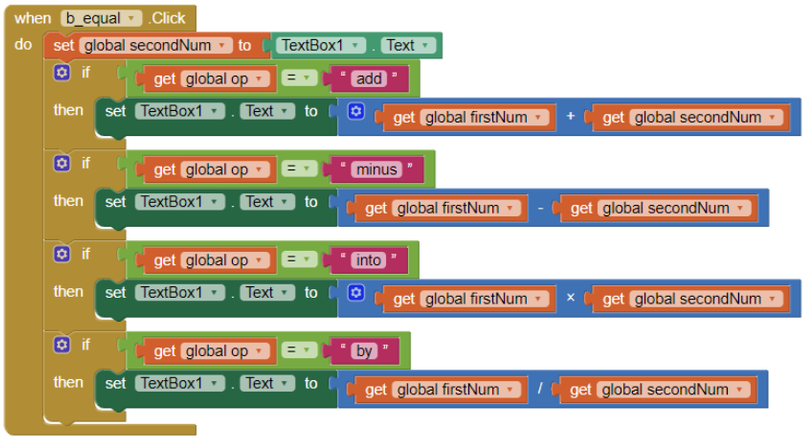


Figure 66: Button equals

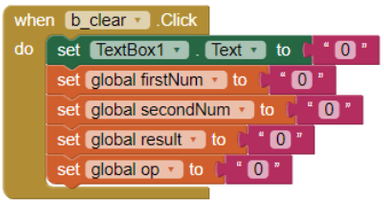


Figure 67: Button C

For practice - download and test one calculator from Gallery, and after that, build your calculator from scratch by following instructions.

Za vježbu – preuzmite i pokrenite jedan gotov kalkulator, a nakon toga izradite vlastiti kalkulator sljedeći upute!

Gyakorlásképpen - tölts le és futtass egy kész alkalmazást, majd az utasításokat követve készítsd el a saját alkalmazásod!

2.7. MakeBlockRemoteControler application
2.7. Aplikacija MakeBlockRemoteControler
2.7. MakeBlockRemoteControler aplikáció

Now we will develop an application that will be used as a remote controller for controlling MakeBlock robot vehicle. The android part of the application will be built in App Inventor and it will run on a mobile device. MakeBlock robot is Arduino driven so it requires an application developed in Arduino Integrated Development Environment.

Sada ćemo izraditi aplikaciju koju ćemo koristiti kao daljinski upravljač za upravljanje MakeBlock robotskim vozilom. Aplikacija za Android će biti izrađena u App Inventoru i izvodit će se na Android uređaju. Na MakeBlock robotu koji je pogonjen Arduino uređajem, potrebno je učitati aplikaciju izrađenu u Arduino integriranoj razvojnoj okolini.

Most létrehozunk egy alkalmazást, amelyet távirányítóként fogunk használni a MakeBlock robotjármű vezérlésére. Az Android-alkalmazás az App Inventorba épül, és az Android-eszközön fog futni. Az Arduino-alapú MakeBlock roboton egy Arduino integrált fejlesztői környezetben létrehozott alkalmazást kell betölteni.

2.7.1. Arduino code
2.7.1. Arduino kod
2.7.1. Arduino kód

```
#include "MeMegaPi.h"
#include <SoftwareSerial.h>

//Encoder Motor
MeEncoderOnBoard motor1(SLOT1);
MeEncoderOnBoard motor2(SLOT2);
MeEncoderOnBoard motor3(SLOT3);

//DC motor (hvataljka)
MeMegaPiDCMotor hvataljka(PORT4B);

//MeBluetooth bluetooth (PORT_5);

//Initial motor speed
int motorSpeed = 150;


char readBT(){
   char btIn;
   btIn = (char) Serial3.read();
   return btIn;
}
```

```
void setup()
{
 Serial3.begin(115200);
 //bluetooth.begin(115200);    //The factory default baud rate is 115200
}

void loop()
{
 if (Serial3.available())
 {
  char cmd = readBT();

  //FORWARD
  if (cmd == '1')
  {
   motor1.setMotorPwm(motorSpeed);
   motor2.setMotorPwm(-motorSpeed);
  }

  //LEFT
  if (cmd == '2')
  {
   motor1.setMotorPwm(motorSpeed);
   motor2.setMotorPwm(motorSpeed);
  }

  //RIGHT
  if (cmd == '3')
  {
   motor1.setMotorPwm(-motorSpeed);
   motor2.setMotorPwm(-motorSpeed);
  }

  //BACKWARD
  if (cmd == '4')
  {
   motor1.setMotorPwm(-motorSpeed);
   motor2.setMotorPwm(motorSpeed);
  }

  //STOP
  if (cmd == '5')
  {
   motor1.setMotorPwm(0);
   motor2.setMotorPwm(0);
   motor3.setMotorPwm(0);
  }
```

```
  //ARM UP
  if (cmd == '6')
  {
    motor3.setMotorPwm(-220);
  }

  //ARM DOWN
  if (cmd == '7')
  {
    motor3.setMotorPwm(180);
  }

  //GRIPPER CLOSE
  if (cmd == '8')
  {
    hvataljka.run(150);
    delay(2000);
    hvataljka.stop();
  }

  //GRIPPER OPEN
  if (cmd == '9')
  {
    hvataljka.run(-150);
    delay(2000);
    hvataljka.stop();
  }

  //SPEED UP
  if (cmd == 'U')
  {
    if (motorSpeed < 245){
      motorSpeed = motorSpeed + 10;
    }
  }

  //SLOW DOWN
  if (cmd == 'D')
  {
    if (motorSpeed > 100){
      motorSpeed = motorSpeed - 10;
    }
  }

 }
}
```

2.7.2. Android application

2.7.2. Android aplikacija

2.7.2. Android aplikáció

In order for Android application to successfully communicate with the Makeblock Ultimate robot series 2.0, it is not enough to use classical Bluetooth Module. It is necessary to download BluetoothLE module available at the following link:

http://iot.appinventor.mit.edu/assets/resources/edu.mit.appinventor.ble-20200828.aix.

After downloading, it is necessary to load the module in the AppInventor project by using the option Extension as shown in the figure. After that, it is necessary to build the graphical user interface which allows the movement of the robot in 4 directions, lifting and lowering the robot arm, opening and closing the grip, speeding up and slowing down the movement of the robot. Expected user interface layout and control names are shown in the following figures.

Da bi Android aplikacija uspješno komunicirala s Makeblock Ultimate robotom serije 2.0, nije dovoljno koristiti klasičan Bluetooth modul, nego je potrebno preuzeti BluetoothLE modul dostupan na poveznici http://iot.appinventor.mit.edu/assets/resources/edu.mit.appinventor.ble-20200828.aix.

Preuzeti modul potrebno je učitati u AppInventor projekt pomoću opcije Extension kao što prikazuje slika. Nakon toga, potrebno je izraditi grafičko korisničko sučelje aplikacije koje omogućuje kretanje robota u četiri smjera, podizanje i spuštanje robotske ruke, otvaranje i zatvaranje hvataljke te ubrzavanje i usporavanje robota. Očekivani izgled sučelja i nazivi kontrola prikazani na slijedećim slikama.

Ahhoz, hogy az Android alkalmazás sikeresen kommunikáljon a Makeblock Ultimate robotsorozat 2.0-val, nem elég a klasszikus Bluetooth modult használni, hanem le kell tölteni a linken elérhető BluetoothLE modult : http://iot.appinventor.mit.edu/assets/resources/edu.mit.appinventor.ble-20200828.aix.

A letöltött modult be kell tölteni az AppInventor projektbe a képen látható Kiterjesztés opció használatával. Ezt követően létre kell hozni az alkalmazás grafikus felhasználói felületét, amely lehetővé teszi a robot négyirányú mozgását, a robotkar emelését és leengedését, a megfogó nyitását és zárását, valamint a robot gyorsítását és lassítását. A felület várható megjelenése és a vezérlőelemek nevei a következő képeken láthatók.

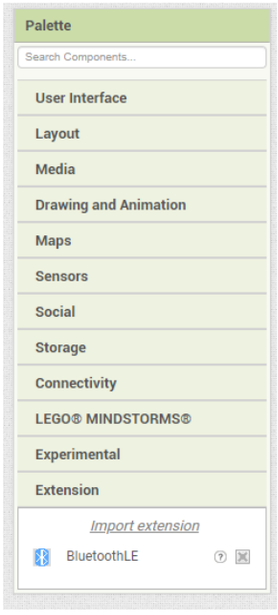


Figure 68: BluetoothLE module loading

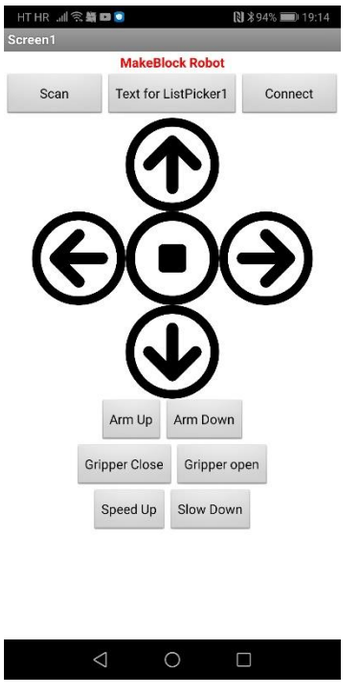


Figure 69: User interface

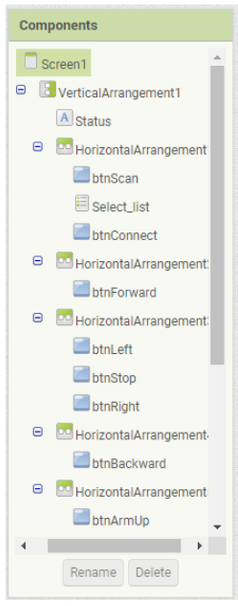


Figure 70: Controls and their names

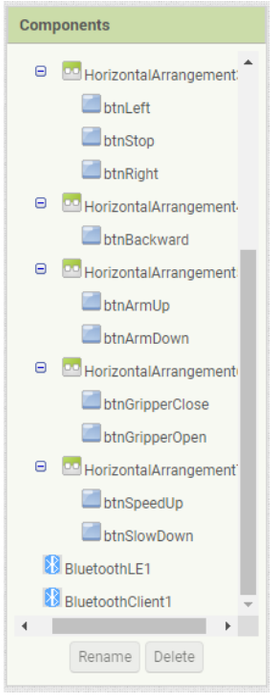


Figure 71: Controls and their names

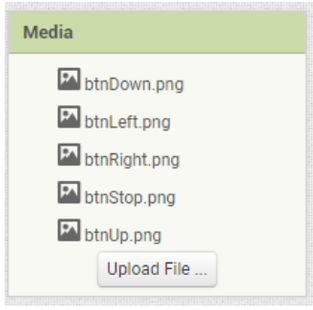


Figure 72: Images for our buttons loaded with the option "Upload Image"

After user interface, we're moving on to application functionality programming by using blocks.

Nakon korisničkog sučelja, prelazimo na programiranje aplikacijske funkcionalnosti pomoću blokova.

A felhasználói felület után áttérünk az alkalmazás funkcióinak blokkok segítségével történő programozására.

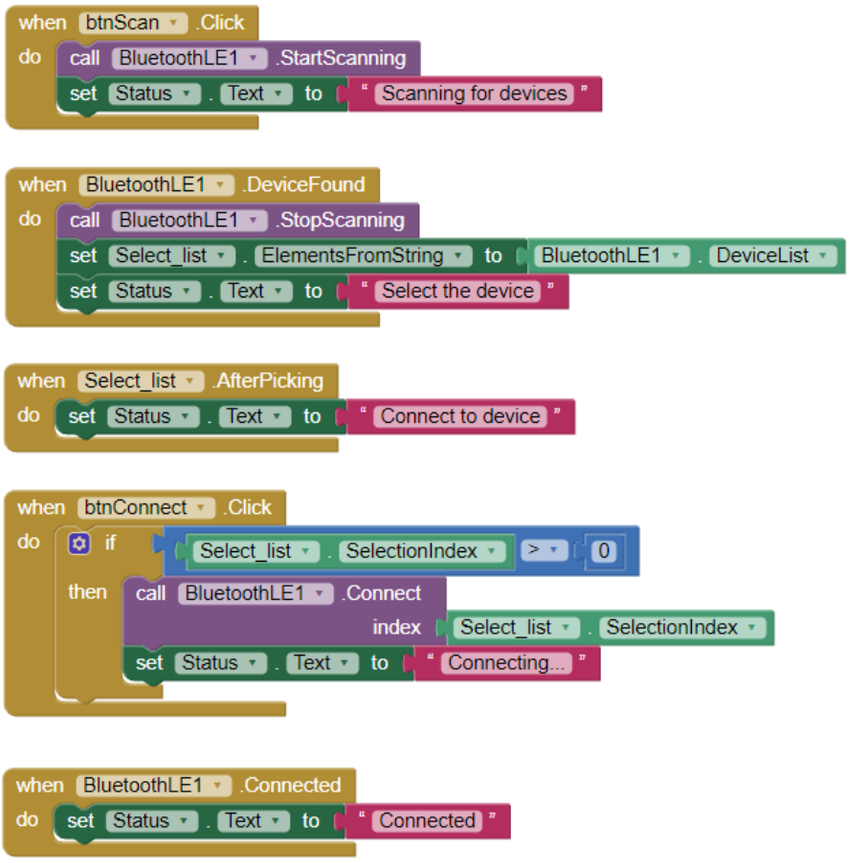


Figure 73: Blocks that describe the functionality of the buttons btnScan, btnConnect and the list Select_list, enables the connectivity of a mobile device with the robot by a Bluetooth and writing the status of the application on the Status label

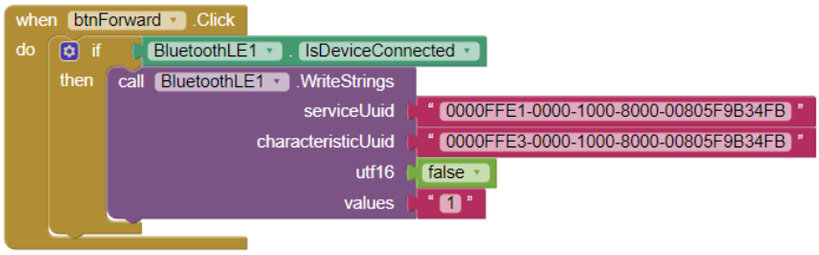


Figure 74: btnForward functionality block

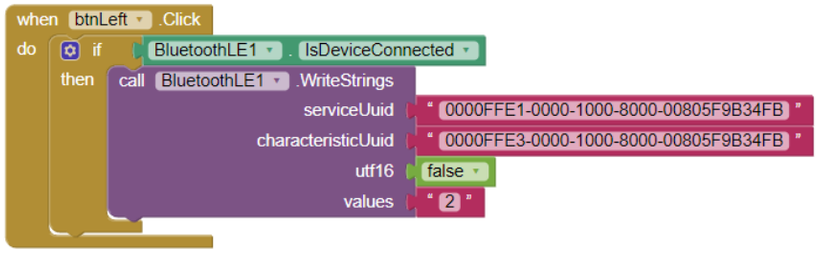


Figure 75: btnLeft functionality block

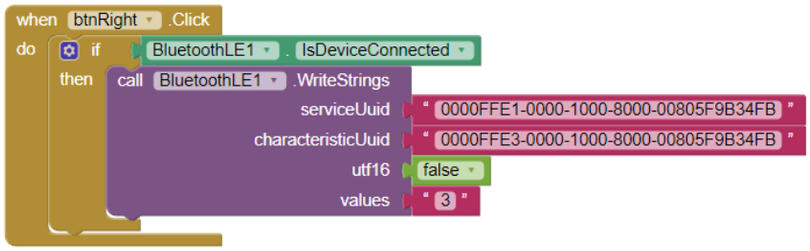


Figure 76: btnRight functionality block

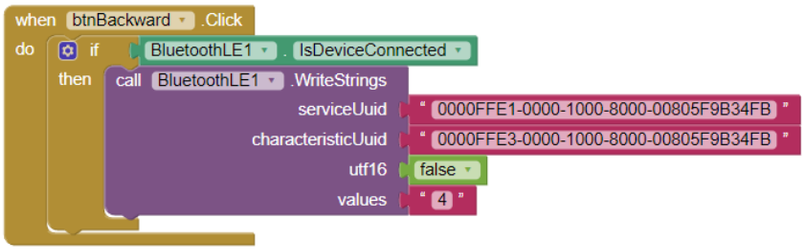


Figure 77: btnBackward functionality block

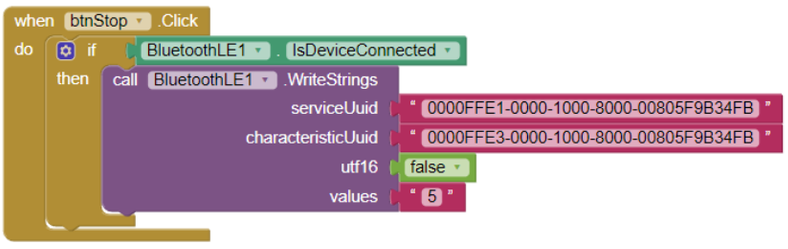


Figure 78: btnStop functionality block

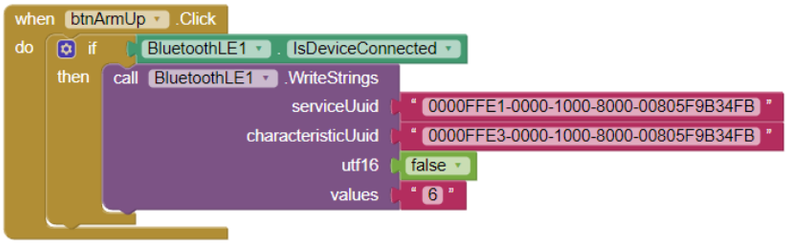


Figure 79: btnArmUp functionality block

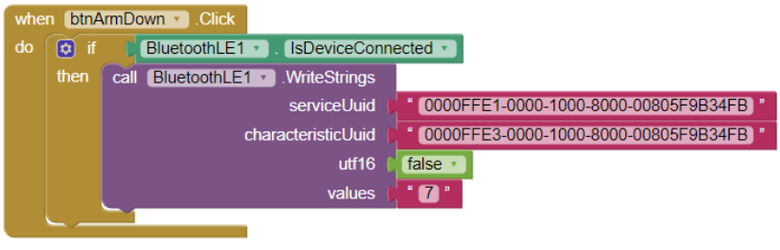


Figure 80: btnArmDown functionality block

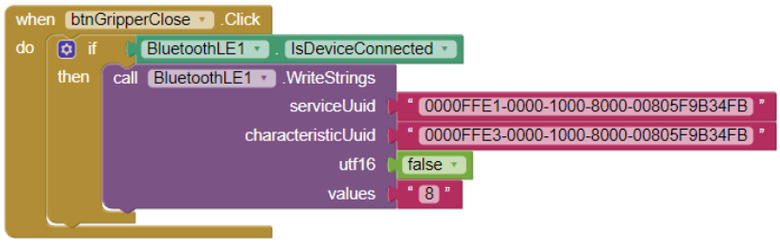


Figure 81: btnGripperClose functionality block

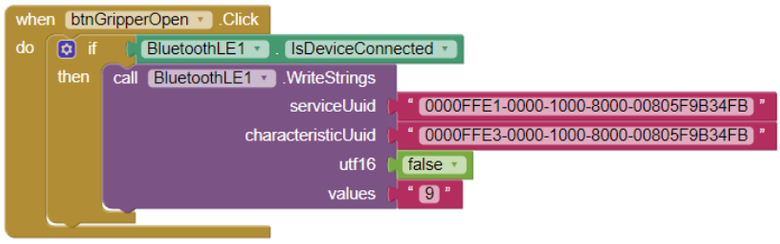


Figure 82: btnGripperOpen functionality block

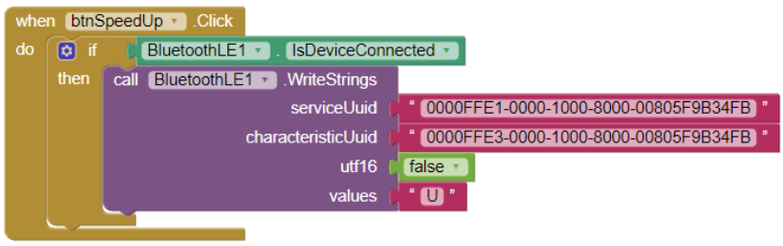


Figure 83: btnSpeedUp functionality block

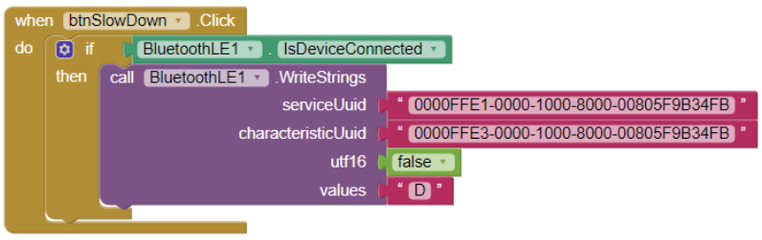


Figure 84: btnSlowDown functionality block

3. IoT – Internet of Things / Internet stvari / Internet eszközök

Internet of things means connecting devices over the Internet. It represents a network infrastructure in which physical and virtual “things” of all kinds communicate and are invisibly integrated. Connecting devices can be wireless and provides new opportunities for interoperability not only between different systems and brings new possibilities for their control, monitoring and provision of advanced services. These can be, for example, refrigerators, washing machines, or various safety systems. The term was coined by Peter T. Lewis.

Internet stvari (engl. Internet of things) označava povezivanje uređaja putem interneta. Predstavlja mrežnu infrastrukturu u kojoj fizičke i virtualne "stvari" svih vrsta komuniciraju i nevidljivo su integrirane. Spajanje uređaja može biti bežično i omogućava nove mogućnosti za međusobnu interakciju ne samo između različitih sustava i donosi nove mogućnosti njihove kontrole, praćenje i pružanje naprednih usluga. To mogu biti primjerice hladnjaci, perilice, ili razni sigurnosni sustavi. Pojam je skovao Peter T. Lewis.[5]

A tárgyak internete eszközök csatlakoztatását jelenti az interneten keresztül. Olyan hálózati infrastruktúrát képvisel, amelyben mindenféle fizikai és virtuális "eszköz" kommunikál egymással, és láthatatlanul integrálódnak. A csatlakoztatott eszközök vezeték nélküliek lehetnek, és új lehetőségeket tesznek lehetővé a kölcsönös interakcióban nemcsak a különböző rendszerek között, hanem új lehetőségeket kínálnak azok vezérlésére, felügyeletére és fejlett szolgáltatások nyújtására. Ilyenek lehetnek például hűtőszekrények, mosógépek, vagy különféle biztonsági rendszerek. A kifejezést Peter T. Lewis alkotta meg.

3.1. Hardware
3.1. Upoznavanje sa sklopovljem
3.1. Ismerkedés az áramkörrel

For the application of the Internet of Things, we will use the Arduino board MKR1000, or a development set with the specified board and electronic elements and assemblies. The Arudino MKR1000 is a development board with a 32-bit ARM microcontroller architecture with a built-in WiFi module. More details about the Arduino MKR100 board can be found at the following link: https://store.arduino.cc/arduino-mkr1000-wifi-with-headers-mounted

Za primjenu Internet stvari koristit ćemo Arduino pločicu MKR1000, odnosno razvojni set sa navedenom pločicom te elektroničkim elementima i sklopovima. Arudino MKR1000 razvojna je pločica sa 32-bitnim mikrokontrolerom ARM arhitekture sa ugrađenim WiFi modulom.
Više detalja o Arduino MKR100 pločici može se pronaći na sljedećoj poveznici: https://store.arduino.cc/arduino-mkr1000-wifi-with-headers-mounted

Az Internet of Things alkalmazásához az Arduino MKR1000-es kártyát, azaz egy fejlesztőkészletet használunk a megadott kártyával és elektronikus elemekkel, áramkörökkel. Az Arduino MKR1000 egy fejlesztő kártya 32 bites ARM architektúrájú mikrokontrollerrel, beépített WiFi modullal.

További részletek az Arduino MKR100 tábláról az alábbi linken találhatók:
https://store.arduino.cc/arduino-mkr1000-wifi-with-headers-mounted

[5] https://hr.wikipedia.org/wiki/Internet_stvari

3.1.1 Arduino MKR1000

The Arduino MKR1000 development board consists of a microcontroller but also other elements needed for its operation. To work with the development board, it is important to know the derived connections (pins) that can be directly connected to the experimental board or conductors. They serve as inputs/outputs of the microcontroller. Just as a personal computer (PC) has the ability to connect input (eg keyboard, mouse, touch screen,…) and output (eg screen, speaker,…) units, so the microcontroller has the ability to connect input (eg button, potentiometer, …) and output (eg LED, display,…) units to the input/output connection pins.

The pins of the Arduino MKR1000 board (microcontroller) are marked:

- D0 - D14, te D15-D21 – digital inputs/outputs
- A0 – A6 (D15-D21) – analog inputs
- D0 – D8, D10, A3(D18) and A4(D19) - PWM outputs
- A0 is also a 'real' analog output, ie a DAC (digital-to-analog converter) output

The inputs and outputs of the microcontroller can be divided (according to the type of signal that is brought to them, or that we get on them) into digital and analog. Digital inputs and outputs imply two (binary) logic states on them: logic "0" - 0V and logic "1" -5V (3.3V). Analog inputs and outputs involve analog-to-digital (ADC) and digital-to-analog (DAC) signal conversion, because the computer understands nothing but two states (binary digits) 0 and 1.

In addition to analog and digital signals, these connectors have some other functions such as communication (with other computers), special built-in circuits (so-called timers, etc.).

Arduino MKR1000 razvojna pločica sastoji se od mikrokontrolera ali i ostalih elemenata koji su potrebni za njegov rad. Za rad sa razvojnom pločicom važno je upoznati izvedene priključke (tzv. pinove) koji se izravno mogu povezati sa eksperimentalnom pločicom ili vodičima. Oni služe kao ulazi/izlazi mikrokontrolera. Kao što osobno računalo (PC) ima mogućnost povezivanja ulaznih (npr. tipkovnica, miš, dodirni zaslon, …) i izlaznih (npr. zaslon, zvučnik, …) jedinica, tako i mikrokontroler ima mogućnost povezivanja ulaznih (npr. tipkalo, potenciometar, …) i izlaznih (npr. LED, zaslon, …) jedinica na ulazno/izlazne priključne (pinove).

Priključci (pinovi) Arduino MKR1000 pločice (odnosno mikrokontrolera) označeni su:

- D0 - D14, te D15-D21 – to su tzv. digitalni ulazi/izlazi
- A0 – A6 (D15-D21) – analogni ulazi
- D0 – D8, te D10, A3(D18) i A4(D19) su tzv PWM izlazi
- A0 je također 'pravi' analogni izlaz, tj. izlaz DAC (digitalno-analognog pretvornika)

Ulaze i izlaze mikrokontrolera možemo podijeliti (prema vrsti signala koji je na njih doveden, odnosno koji dobijemo na njima) na digitalne i analogne. Digitalni ulazi i izlazi podrazumijevaju dva (binarno) logička stanja na njima: logički „0“ – 0V i logički „1“ -5V (3.3V). Analogni ulazi i izlazi podrazumijevaju analogno-digitalnu (ADC), odnosno digitalno-analognu (DAC) pretvorbu signala, jer računalo ne razumije ništa osim dva stanja (binarne znamenke) 0 i 1.

Osim analognih i digitalnih signala, navedeni priključci imaju i neke druge funkcije poput komunikacije (s drugim računalima), posebnih ugrađenih sklopova (tzv. vremenskih brojača i sl.).

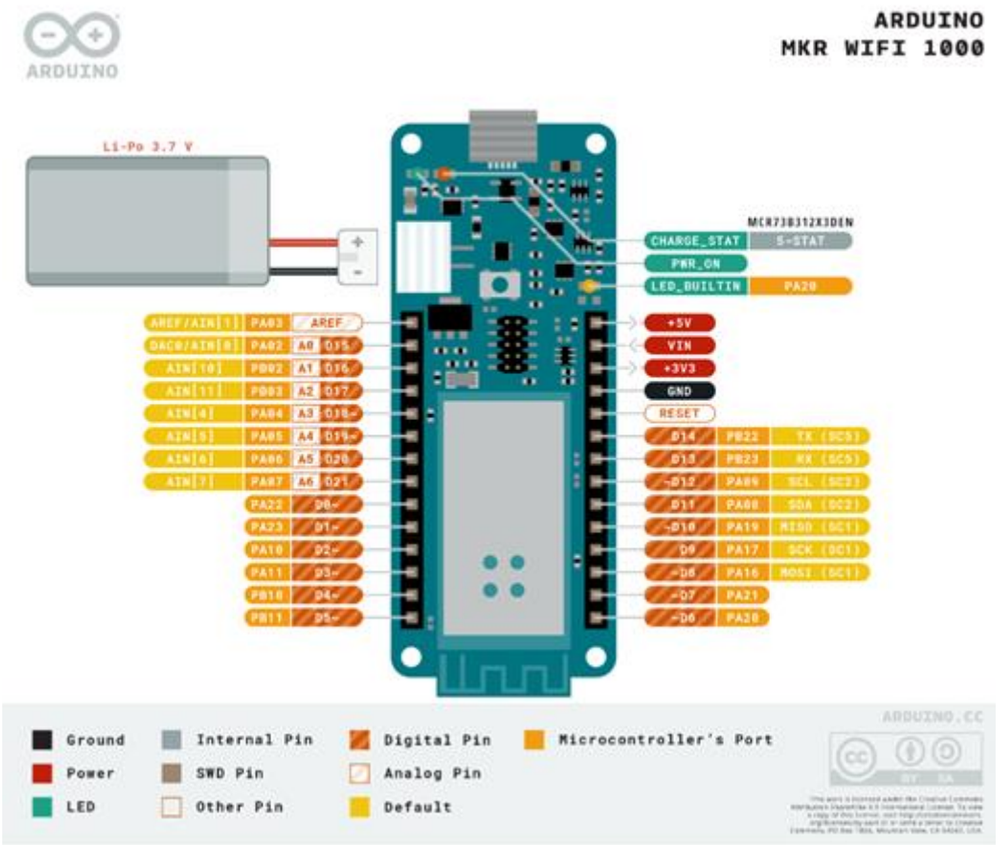
Az Arduino MKR1000 fejlesztőkártya egy mikrokontrollerből és egyéb, a működéséhez szükséges elemekből áll. A fejlesztő kártyával való munkavégzéshez fontos ismerni a származtatott csatlakozásokat (ún. tűket), amelyek közvetlenül csatlakoztathatók a kísérleti kártyához vagy vezetékekhez. Ezek a mikrokontroller be-/kimeneteiként szolgálnak. Ahogy a személyi számítógép (PC) képes bemeneti (pl. billentyűzet, egér, érintőképernyő stb.) és kimeneti (pl. képernyő, hangszóró, ...) egységek csatlakoztatására, úgy a mikrokontroller is képes bemeneti (pl. gomb, potenciométer, ...) és kimeneti (pl. LED, képernyő, ...) egységeket csatlakoztatni a bemeneti/kimeneti csatlakozókhoz (tüskékhez).

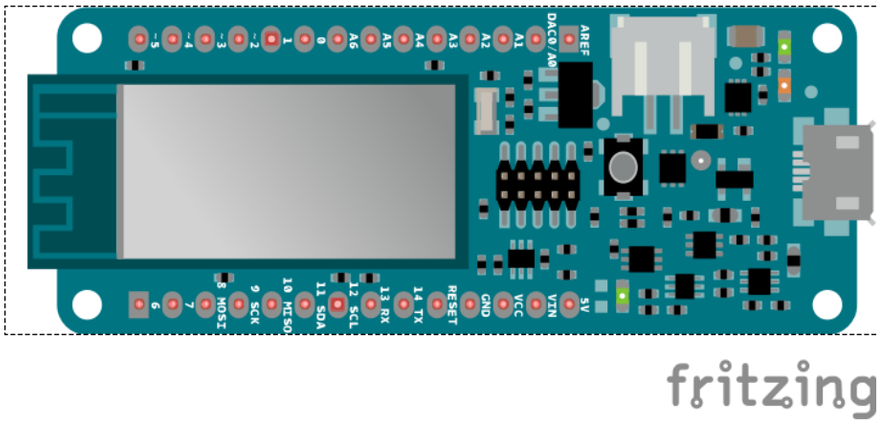
Az Arduino MKR1000 kártya (vagyis a mikrokontroller) csatlakozói (tüskék) jelöléssel vannak ellátva:

- D0 - D14, te D15-D21 – ezek az ún. digitális bemenetek/kimenetek
- A0 – A6 (D15-D21) – analóg bemenetek
- D0 – D8, valamint D10, A3(D18) és A4(D19) az ún. PWM kimenetek
- az A0 is szintén egy „igazi“ analóg kimenet, azaz DAC kimenet (digitális-analóg konverter)

A mikrokontroller be- és kimenetei (a rájuk betáplált, azaz tőlük kapott jel típusától függően) digitálisra és analógra oszthatók. A digitális bemenetek és kimenetek két (bináris) logikai állapotot tartalmaznak: logikai "0" - 0V és logikai "1" -5V (3,3V). Az analóg bemenetek és kimenetek analóg-digitális (ADC) vagy digitális-analóg (DAC) jelátalakítást jelentenek, mivel a számítógép nem ért mást, mint két állapotot (bináris számjegyet), 0 és 1.

Az említett csatlakozók az analóg és digitális jeleken kívül más funkciókat is ellátnak, mint például kommunikáció (más számítógépekkel), speciális beépített áramkörök (ún. időszámlálók stb.).





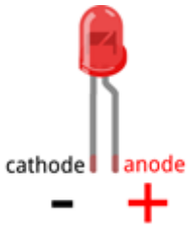
3.1.2 Electronic components

The electronic components that we will use most often are:

Pushbutton

A pushbutton is a type of switch that allows us to connect a circuit by pressing it and thus turn on a device. The pushbutton connects its two outputs only as long as we hold it down.

LED (Light Emitting Diode)



An LED is a semiconductor component (diode) that emits light when connected to a power supply. The LED is used as a basic indicator that something is on/off.

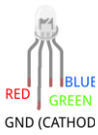
Resistor

A resistor is an electronic component that provides resistance to the flow of current through a conductor, and the basic application of a resistor will be in a circuit with an LED. The LED must not be connected directly to the current, ie without a resistor of appropriate resistance, because too much current that would then flow through the LED could destroy this component. The amount of resistance of an individual resistor is read by means of colored rings printed on it (or according to the numerical markings on the small packages of the element).

Potentiometer

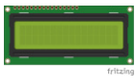
A potentiometer is a resistor to which we can change the amount of resistance by having a small shaft that we can rotate in both directions and thus change the amount of resistance of that element.

RGB LED



RGB LEDs are actually 3 LEDs in one housing, by combining (8 bit values, 0-255) red, green and blue LEDs we can get other colors.

LCD



The 16x2 LCD can print 16 characters in 2 lines. Use 40 dots (5x8) to display each character

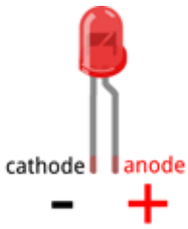
3.1.2 Elektronički elementi

Elektronički elementi koje ćemo najčešće koristiti su:

Tipkalo (engl. pushbutton)

Tipkalo je vrsta prekidača koja nam omogućuje da pritiskom na njega spojimo strujni krug i na taj način uključimo neki uređaj. Tipkalo spaja svoja dva izlaza samo dok ga držimo pritisnutim.

LED (engl. Light Emitting Diode)



LED je poluvodički element (dioda) koji emitira svjetlost kada ga spojimo na napajanje. LED nam služi kao osnovni pokazatelj da je nešto uključeno.

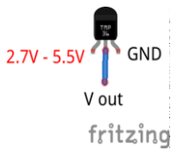
Otpornik (engl. resistor)

Otpornik je elektronički element koji pruža otpor protjecanju struje kroz vodič, a osnovna primjena otpornika bit će nam u strujnom krugu sa LED. LED se ne smije spojiti izravno u strujni tj. bez otpornika odgovarajućeg otpora, jer bi prevelik iznos struje koja bi tada protjecala kroz LED mogao uništiti tu komponentu. Iznos otpora pojedinog otpornika očitava se pomoću prstenova u boji koji su na njemu otisnuti (ili prema brojčanim oznakama na malim pakiranjima elementa).

Potenciometar (engl. potentiometer)

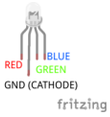
Potenciometar je otpornik kojemu možemo mijenjati iznos otpora tako što ima izvedenu malu osovinu koju možemo zakretati u oba smjera i na taj način mijenjati iznos otpora tog elementa.

Senzor temperature TMP36



Senzor temperature mjeri temperaturu u rasponu od 0.1V (-40°C) do 2.0V (150°C).

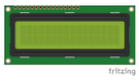
RGB LED



RGB LED zapravo su 3 LED u jednom kućištu, kombinacijom (8 bitne vrijednosti, 0-255) crvene, zelene i plave LED možemo dobiti ostale boje.

LCD

ispisati 16 znakova u 2 reda. Za prikaz svakog znaka koristi 40 točkica (5x8



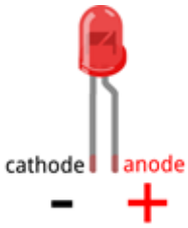
3.1.2 Elektronikai elemek

Elektronikai elemek, amelyeket a legtöbbször használunk majd a következők:

Nyomógomb (engl. pushbutton)

A nyomógomb egyfajta kapcsoló, amely lehetővé teszi, hogy megnyomásával összekapcsoljunk egy áramkört, és így bekapcsoljunk egy eszközt. A gomb csak nyomva tartva köti össze a két kimenetét.

LED (engl. Light Emitting Diode)



A LED egy félvezető elem (dióda), amely fényt bocsát ki, ha tápegységhez csatlakoztatjuk. A LED alapvető jelzésként szolgál, ha valami be van kapcsolva.

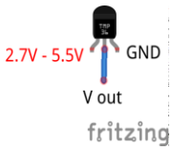
Ellenállás (engl. resistor)

Az ellenállás olyan elektronikus elem, amely ellenáll a vezetőn áthaladó áramnak, és az ellenállás alapvető alkalmazása egy LED-es áramkörben történik. A LED-et nem szabad közvetlenül az áramra kötni, azaz megfelelő ellenállású ellenállás nélkül, mert a LED-en átfolyó túl sok áram tönkreteheti az alkatrészt. Az egyes ellenállások ellenállásának leolvasása a rányomott színes gyűrűk segítségével történik (vagy az elem kis csomagolásán található számjelzések alapján).

Potenciométer (engl. potentiometer)

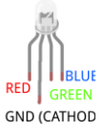
A potenciométer olyan ellenállás, amelynek ellenállása úgy változtatható meg, hogy van egy kis tengelye, amely mindkét irányba elforgatható, és így az adott elem ellenállása változtatható.

Hőmérséklet szenzor TMP36



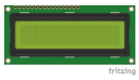
A hőmérséklet-érzékelő a hőmérsékletet 0,1 V (-40°C) és 2,0 V (150°C) tartományban méri.

RGB LED



Az RGB LED-ek tulajdonképpen 3 LED egy tokban, a (8 bites értékek, 0-255) piros, zöld és kék LED kombinálásával más színeket is kaphatunk.

LCD



A 16x2 LCD kijelző 16 karaktert tud megjeleníteni 2 sorban. 40 pontot (5x8) használ az egyes karakterek megjelenítéséhez.

3.1.3 Breadboard

3.1.3 Eksperimentalna pločica

3.1.3 Kísérleti alaplap

The experimental board (breadboard) is used to connect electronic components and circuits, without the need to develop and make an PCB, or when it is necessary to do it quickly and easily (for the purpose of testing, learning,…). The holes on the experimental board connect the internal conductors that connect the plugged elements. On the left and right sides of the experimental tile, the area is intended to connect the power source (marked in red and blue), and these holes are connected vertically (+ and -).

Holes in the inner part of the experimental plate, for easier navigation, marked with letters (A - J) and numbers (1 - 30), are used to connect elements and assemblies to each other, ie to the power supply. The figure below shows the connection of the MKR1000 plates to the elements on the experimental plate. Note that the holes connecting the elements are marked in green (of course, not all the holes marked in green are connected), and the channels in the middle of the experimental tile that separate the connected elements left and right.

Eksperimentalna pločica (engl. breadboard) služi nam da bismo povezali elektroničke elemente i sklopove, bez potrebe za razvojem i izradom elektroničke pločice, odnosno kada je to potrebno napraviti brzo i jednostavno (u svrhu isprobavanja, učenja, …). Rupice na eksperimentalnoj pločici povezuju unutarnji vodiči koji spajaju utaknute elemente. S lijeve i desne strane eksperimentalne pločice područje je namijenjeno spajanju izvora napajanja (označeno crvenom i plavom bojom), a te rupice su povezne okomito (+ i -).

Rupice u unutarnjem dijelu eksperimentalne pločice, radi lakšeg snalaženja označene slovima (A – J) i brojevima (1 - 30), služe za povezivanje elemenata i sklopova međusobno, odnosno na napajanje. Donja slika prikazuje povezivanje MKR1000 pločice sa elementima na eksperimentalnoj pločici. Primijetite da su zelenom bojom označene rupice koje povezuju elemente (naravno nisu sve rupice označene zelenom bojom povezane), te kanalić na sredini eksperimentalne pločice koji odvaja lijevo i desno spojene elemente.

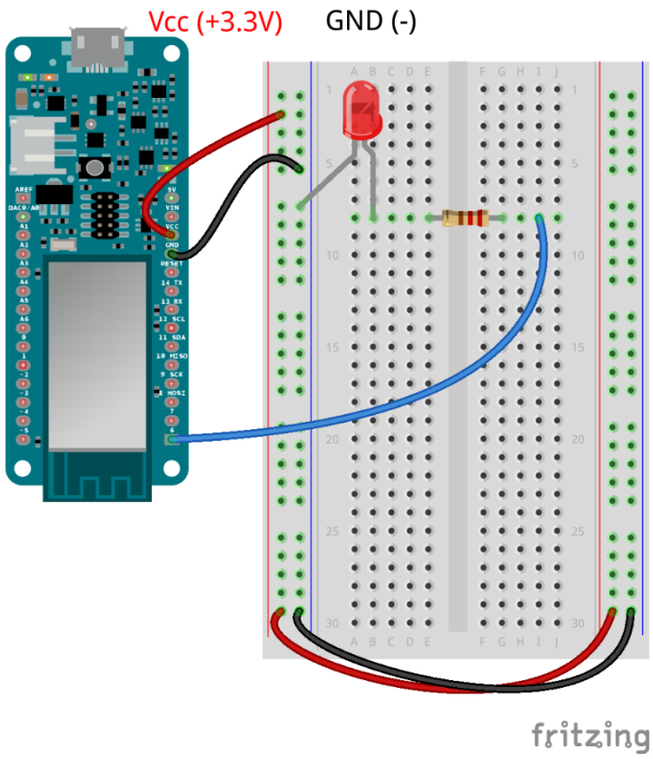
A kísérleti alaplap (angolul kenyértábla) elektronikus elemek, áramkörök összekapcsolására szolgál, elektronikus kártya fejlesztése, elkészítése nélkül, azaz amikor azt gyorsan és egyszerűen kell megtenni (tesztelés, tanulás, . ..). A kísérleti alaplapon lévő lyukakat belső vezetékek kötik össze, amelyek összekötik a behelyezett elemeket. A kísérleti alaplap bal és jobb oldalán található az áramforrás csatlakoztatására szolgáló terület (piros és kék színnel jelölve), és ezek a furatok függőlegesen (+ és -) vannak összekötve.

A kísérleti alaplap belső részén található lyukak, amelyeket betűkkel (A - J) és számokkal (1 - 30) jelöltek a könnyebb hivatkozás érdekében, az elemek és áramkörök egymáshoz, azaz a tápegységhez való csatlakoztatására szolgálnak. Az alábbi ábrán látható az MKR1000 kártya csatlakoztatása a kísérleti alaplap elemeihez. Figyeljük meg, hogy az elemeket összekötő lyukak zölddel vannak jelölve (természetesen nem minden zölddel jelölt lyuk van összekötve), illetve a kísérleti lemez közepén található csatorna, amely elválasztja a bal és jobb oldali összefüggő elemeket.

NOTICE:
The Arduino MKR1000 board can actually be connected directly to the experimental board, but for clarity it is set as in the picture.

NAPOMENA:
Arduino MKR1000 pločica može se zapravo izravno priključiti na eksperimentalnu pločicu, ali je zbog preglednosti postavljeno kao na slici.

MEGJEGYZÉS:
Az Arduino MKR1000 kártya valójában közvetlenül csatlakoztatható a kísérleti alaplaphoz, de az áttekinthetőség kedvéért az ábrán látható módon van elhelyezve.



3.2. IDE – Integrated Development Enviroment

3.2. Upoznavanje s razvojnom okolinom

3.2. Ismerkedés a fejlesztői környezettel

We will use the Arduino development environment described in Chapter 1, but we actually have the following choices:

1. Arduino desktop development environment (IDE): https://www.arduino.cc/en/software
2. Arduino Web Development Environment (IDE): https://create.arduino.cc/editor
3. Arduino IoT Cloud development environment: https://www.arduino.cc/en/IoT/HomePage

The following link provides a brief guide to working with the Arduino MKR1000 board:

https://www.arduino.cc/en/Guide/MKR1000#use-your-arduino-mkr-1000-on-the-arduino-web-ide

Inside the Arduino IDE sub file> examples there are code examples, and also at www.arduino.cc.

Koristit ćemo Arduino razvojnu okolinu opisanu u 1. poglavlju, s tim da zapravo imamo sljedeći izbor:

1. Arduino desktop razvojnu okolinu (IDE): https://www.arduino.cc/en/software
2. Arduino web razvojnu okolinu (IDE): https://create.arduino.cc/editor
3. Arduino IoT Cloud razvojnu okolinu: https://www.arduino.cc/en/IoT/HomePage

Na sljedećoj poveznici nalazi se kratki vodič vezano za rad sa Arduino MKR1000 pločicom:
https://www.arduino.cc/en/Guide/MKR1000#use-your-arduino-mkr-1000-on-the-arduino-web-ide

Unutar Arduino IDE pod *datoteka>primjeri* nalaze se primjeri koda, a također i na www.arduino.cc.

Az 1. fejezetben leírt Arduino fejlesztői környezetet fogjuk használni, azzal a ténnyel, hogy valójában a következő választási lehetőségek állnak rendelkezésünkre:

1. Arduino desktop fejlesztői környezet (IDE): https://www.arduino.cc/en/software
2. Arduino web fejlesztői környezet (IDE): https://create.arduino.cc/editor
3. Arduino IoT Cloud fejlesztői környezet: https://www.arduino.cc/en/IoT/HomePage

Az alábbi link egy rövid oktatóanyagot tartalmaz az Arduino MKR1000 kártyával való munkavégzésről:
https://www.arduino.cc/en/Guide/MKR1000#use-your-arduino-mkr-1000-on-the-arduino-web-ide

Az Arduino IDE-n belül a file>examples alatt kódpéldák találhatók, valamint ezen a címen is:
www.arduino.cc.

3.2.1 Digital inputs and outputs

3.2.1 Digitalni ulazi i izlazi

3.2.1 Digitális bemenetek és kimenetek

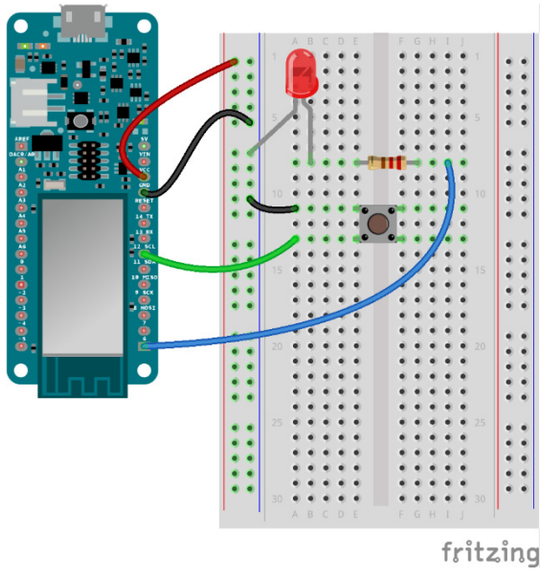
We will use a pushbutton as an example of a digital input, and an LED as an example of a digital output. The Arduino MKR1000 board already has a built-in LED that is connected to pin6 (so that LED will also light up next to the one that is additionally connected). The button is connected to pin12.

Kao primjer digitalnog ulaza koristit ćemo tipkalo, a kao primjer digitalnog izlaza LED. Arduino MKR1000 pločica već ima ugrađenu LED koja je povezana na pin6 (tako da će svijetliti i ta LED uz ovu koja je dodatno spojena). Tipkalo je spojeno na pin12.

Példaként egy gombot fogunk használni a digitális bemenetre, a LED-et pedig a digitális kimenetre. Az Arduino MKR1000 kártyán már van egy beépített LED, amely a 6-os érintkezőhöz csatlakozik (így ez a LED világít a kiegészítőleg csatlakoztatott mellett). A gomb a 12-es érintkezőhöz csatlakozik.

Instructions:

```
//pin6 - LED
#define LED 6
//pin12 - pushbutton
#define button 12
void setup() {
//LED - OUTPUT
pinMode(LED, OUTPUT);
//button - INPUT
pinMode(button, INPUT_PULLUP);
}
void loop() {
  //condition
 if (digitalRead(button) == LOW)
//if button is pressed
 {
  digitalWrite(LED, HIGH); //LED
ON
 }
 else{
  digitalWrite(LED, LOW); //LED
OFF
 }
}
```




```
//Configures the specified pin to behave either as an input or an output.
pinMode(pin, INPUT/OUTPUT);

//Write a HIGH or a LOW value to a digital pin.
digitalWrite(pin, HIGH/LOW);
//Reads the value from a specified digital pin, either HIGH or LOW.
digitalRead(pin);
```

3.2.2 Digital inputs and outputs

3.2.2 Analogni ulazi i izlazi

3.2.2 Analóg bemenetek és kimenetek

As an example of an analog input we will use a potentiometer, and as an example of an analog output again an LED. We will change the voltage at the analog input of the microcontroller with a potentiometer (changing the resistance of the potentiometer will affect the voltage change), since the computer cannot 'understand' the analog size (voltage change), the analog-to-digital converter will convert this voltage into integers ranging from 0 to 1023 (10 bit number). With analog output, the microcontroller will change the output voltage value by changing the duration of logic "1" at the output and again the range of integer values within the command is from 0 to 255 (8 bit number).

LED is connected to pin6, and the potentiometer to pin12!

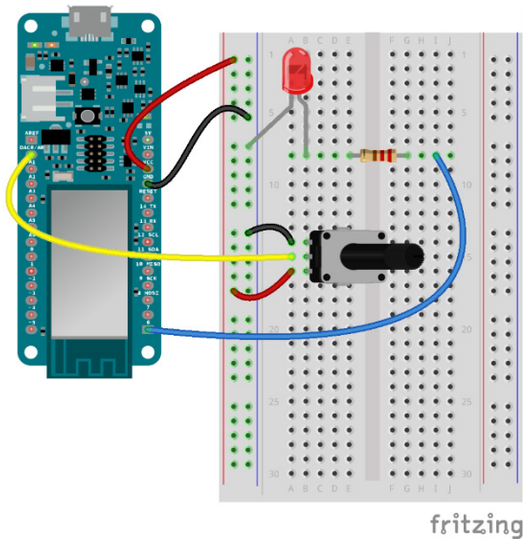
Kao primjer analognog ulaza koristit ćemo potenciometar, a kao primjer analognog izlaza ponovno LED. Potenciometrom ćemo na analognom ulazu mikrokontrolera mijenjati napon (promjena otpora potenciometra utjecat će na promjenu napona), s obzirom da računalo ne može 'razumjeti' analognu veličinu (promjenu napona) analogno-digitalni pretvornik će taj napon pretvoriti u cjelobrojne brojeve u rasponu od 0 do 1023 (10 bitni broj). Kod analognog izlaza mikrokontroler će vrijednost napona na izlazu mijenjati na način da mijenja trajanje logičke „1“ na izlazu a opet raspon cjelobrojnih vrijednosti unutar naredbe je od 0 do 255 (8 bitni broj).

LED spajamo na pin6, a potenciometar na pin12!

Példaként egy potenciométert fogunk használni az analóg bemenetre, a LED-et pedig az analóg kimenetre. A mikrokontroller analóg bemenetén a feszültséget potenciométerrel változtatjuk (a potenciométer ellenállásának változása befolyásolja a feszültség változását), mivel a számítógép nem tudja „megérteni” az analóg értéket (feszültségváltozást), az analóg-digitális konverter ezt a feszültséget egész számokká alakítja a 0 és 1023 közötti tartományban (10 bites szám). Az analóg kimenettel a mikrokontroller megváltoztatja a kimeneti feszültség értékét oly módon, hogy a kimeneten megváltoztatja a logikai "1" időtartamát, és ismét az egész értékek tartománya a parancson belül 0-tól 255 (8 bites szám).

A LED-et a pin6-ra, a potenciométert a pin12-re kötjük!

```
//pin6 - LED
#define LED 6
// potentiometer – A0
#define potentiometer A0
void setup() {
//LED - OUTPUT
pinMode(LED, OUTPUT);
//potentiometer - INPUT
pinMode(potentiometer, INPUT);
}
void loop() {
  /* reads the value from the
*potentiometer (pinA0), converts
*values range from 0-1023 to 0-
*255, and stores in variable pot
*/  int
pot=map(analogRead(potentiometer),
0,1023,0,255);
  // sends the value of the
variable pot (0-255) to the analog
OUTPUT (LED)
  analogWrite(LED, pot);
}
```



Instructions:

```
// the microcontroller reads an analog value at a defined INPUT and
converts it to an integer value in the range 0-1023
analogRead(pin);

// the microcontroller changes the mean voltage value at the defined OUTPUT
according to the range of integer values 0-255.
analogWrite(pin, 0-255);
```

Naredbe:

```
//naredba kojom mikrokontroler čita analognu vrijednost na definiranom
ULAZU i pretvara u cjelobrojnu vrijednost u rasponu 0-1023
analogRead(pin);
//naredba kojom mikrokontroler mijenja srednju vrijednost napona na
definiranom IZLAZU prema rasponu cjelobrojnih vrijednosti 0-255.
analogWrite(pin, 0-255);
```

Parancsok:

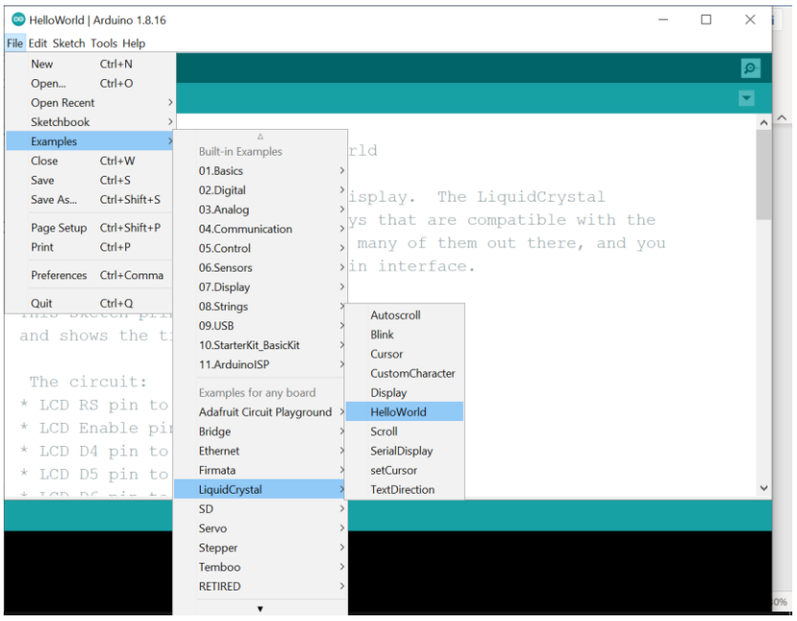
```
// parancs, amellyel a mikrokontroller beolvassa az analóg értéket a
definiált INPUT-on, és egész értékké alakítja a 0-1023 tartományban
analogRead(pin);
// parancs, amellyel a mikrokontroller megváltoztatja a feszültség
középértékét a meghatározott OUTPUT-on a 0-255 egész értékek tartományának
megfelelően.
analogWrite(pin, 0-255);
```

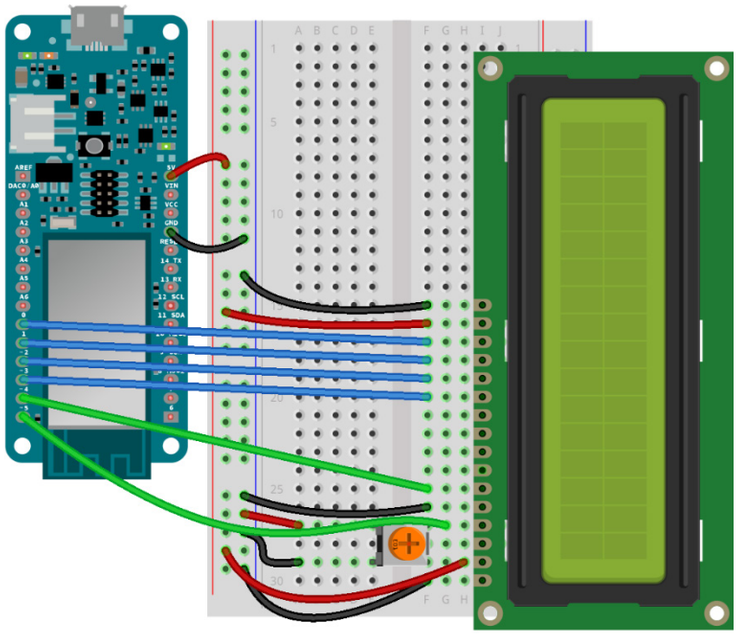
3.2.3 LCD

To use the LCD we need to include the #include <LiquidCrystal.h> library, or we will open the LiquidCrystal> HelloWorld example.

Da bismo koristili LCD moramo uključiti biblioteku #include <LiquidCrystal.h>, odnosno otvorit ćemo primjer *LiquidCrystal > HelloWorld.*

Az LCD használatához meg kell nyitnunk az #include <LiquidCrystal.h> könyvtárat, azaz megnyitjuk a *LiquidCrystal > HelloWorld* példát.





In the code we will change the following according to the connection from the picture:
U kodu ćemo promijeniti sljedeće prema spoju sa slike:
A kódban a következőt változtatjuk a képen látható kapcsolatnak megfelelően:

```
const int rs =5, en = 4, d4 = 3, d5 = 2, d6 = 1, d7 = 0;
```

Program code:

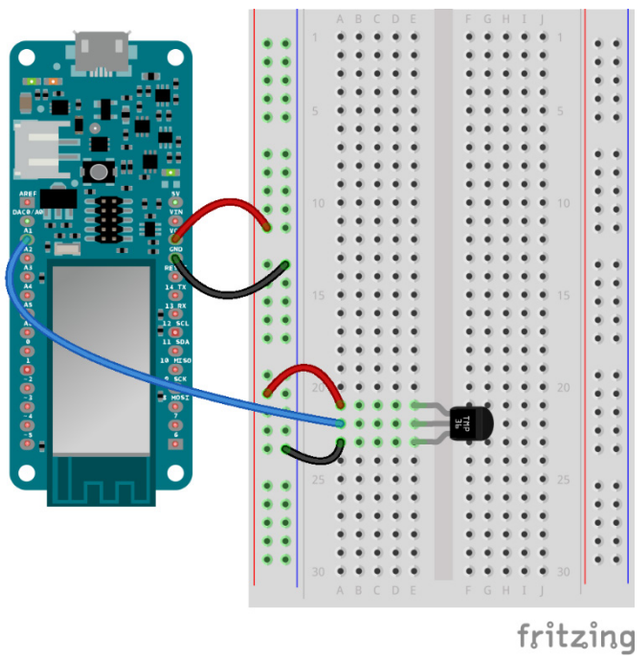
```
//http://www.arduino.cc/en/Tutorial/LiquidCrystalHelloWorld
#include <LiquidCrystal.h>
const int rs =5, en = 4, d4 = 3, d5 = 2, d6 = 1, d7 = 0;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.print("hello, world!");
}
void loop() {
  // set the cursor to column 0, line 1
  // (note: line 1 is the second row, since counting begins with 0):
  lcd.setCursor(0, 1);
  // print the number of seconds since reset:
  lcd.print(millis() / 1000);
}
```

3.2.4 Temperature sensor

3.2.4 Senzor temperature

3.2.4 Hőmérsékleti szenzor

Connect the sensor according to the following figure:
Spojimo senzor prema sljedećoj slici:
Az érzékelő csatlakozása a képen látható módon történik:



The sensor can be connected to a 3.3V or 5V power supply (2.7V to 5.5V), and regardless of the power supply, the voltage reading on it will be between 0V and 1.75V. To convert a 10-bit analog voltage value to temperature, we use the following expression:

Pin voltage in mV = (value with A0) * (3300/1024)

This expression converts the number 0-1023 from A0 to a voltage of 0-3300mV (0V - 3.3V)

The resolution of the sensor is 10 mV / ° C with an offset of 500 mV to allow negative temperatures, so the final expression from which we get the temperature in ° C:

tempC = (analogRead (A0) * (3300/1024)) -500) / 10;

Senzor možemo spojiti na napajanje 3.3V ili 5V (2.7V do 5.5V), te bez obzira na napajanje očitanje napona na njemu bit će između 0V i 1.75V. Za pretvorbu 10-bitne analogne vrijednosti napona u temperaturu koristimo sljedeći izraz:

Napon na pinu u mV = (*vrijednost sa A0*) * (3300/1024)

Ovaj izraz pretvara broj 0-1023 sa A0 u napon 0-3300mV (0V – 3.3V)

Razlučivost senzora je 10 mV /°C s pomakom od 500 mV kako bi se omogućile negativne temperature, stoga je konačan izraz iz kojega dobijemo temperaturu u °C:

tempC=((analogRead(A0) * (3300/1024)) -500) / 10 ;

Az érzékelő 3,3 V-os vagy 5 V-os tápegységre csatlakoztatható (2,7 V-tól 5,5 V-ig), és a tápegységtől függetlenül 0 V és 1,75 V között lesz a leolvasott feszültség. A feszültség 10 bites analóg értékének hőmérsékletté alakításához a következő kifejezést használjuk:

Pin feszültség mV-ban = (*A0 érték*) * (3300/1024)

Ez a kifejezés a 0-1023 számot A0-ból 0-3300 mV feszültséggé alakítja (0V – 3.3V)

Az érzékelő felbontása 10 mV /°C, 500 mV-os eltolás mellett a negatív hőmérsékletek figyelembevétele érdekében, így a végső kifejezés, amelyből megkapjuk a hőmérsékletet °C-ban:

tempC=((analogRead(A0) * (3300/1024)) -500) / 10 ;

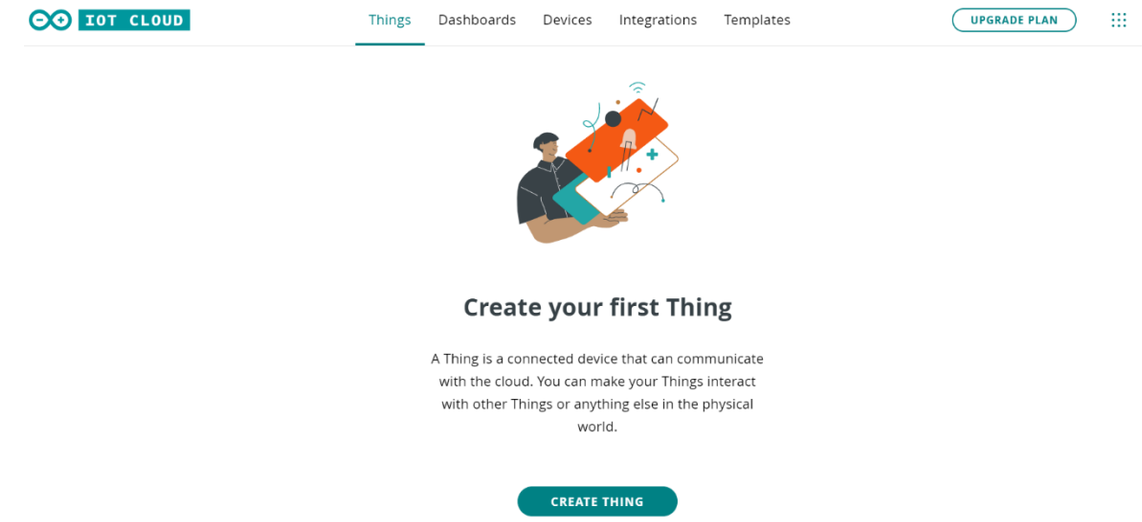
Program code:

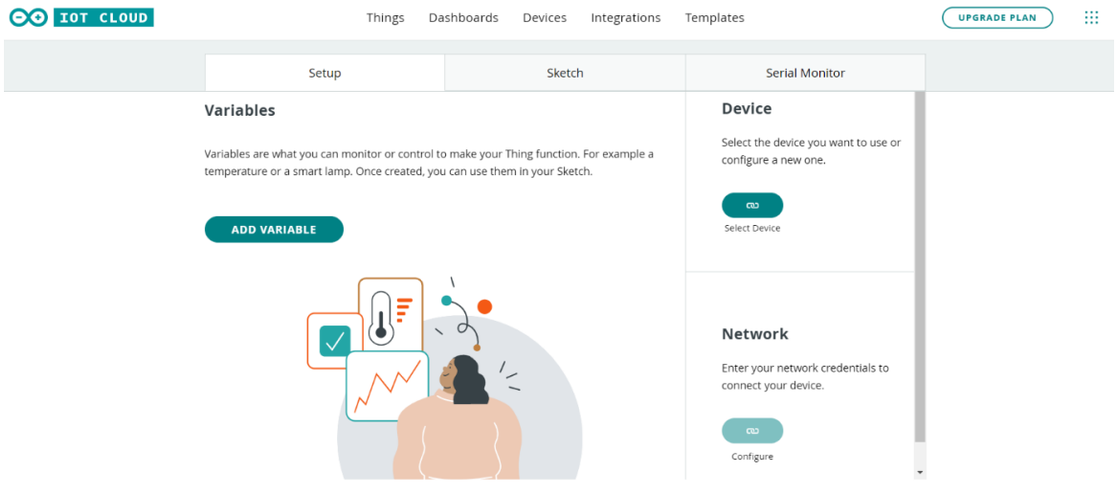
```
void setup() {
pinMode(A0, INPUT);
Serial.begin(115200);
}

void loop() {
 float tempC=((analogRead(A0) * (3300/1024)) - 500 ) / 10 ;
Serial.print(analogRead(A0));
Serial.print(" ");
Serial.println(tempC);
delay(1000);
```

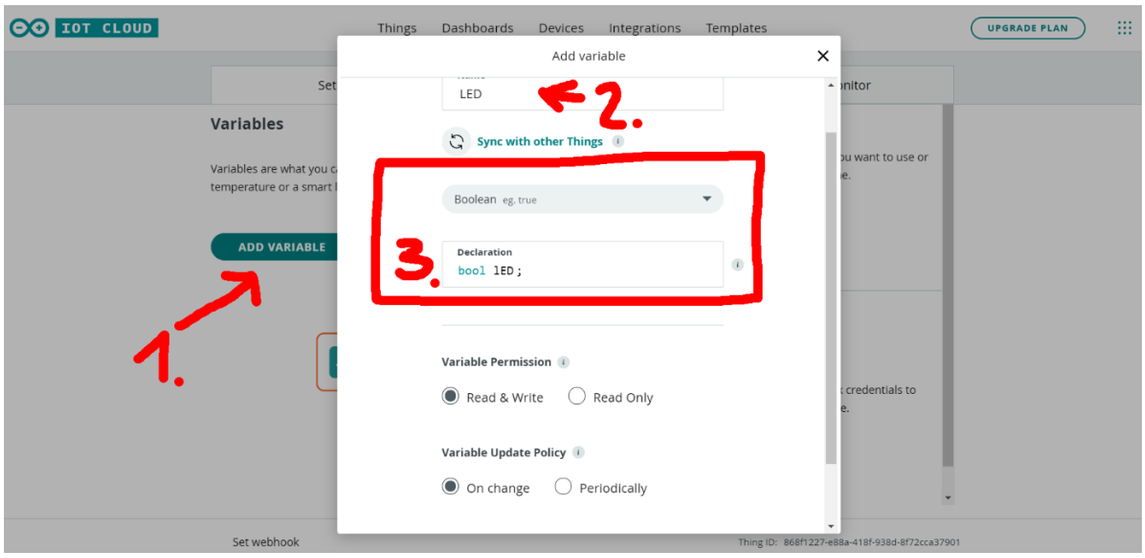
3.2.5 Arduino IoT cloud

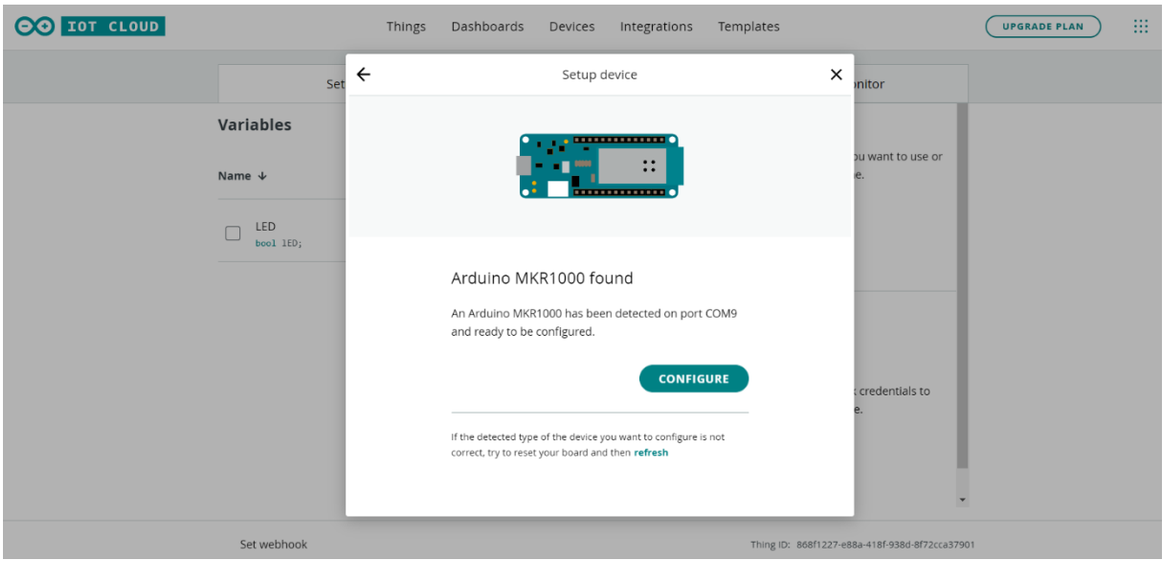
3.2.5 Arduino IoT cloud razvojna okolina

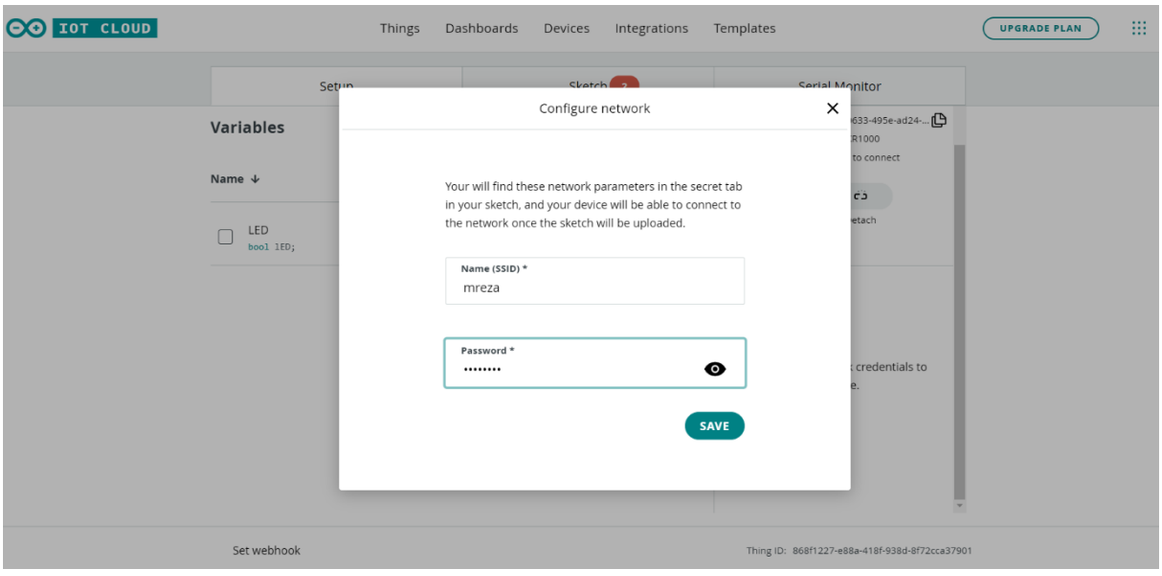
3.2.5 Az Arduino IoT cloud fejlesztői környezete

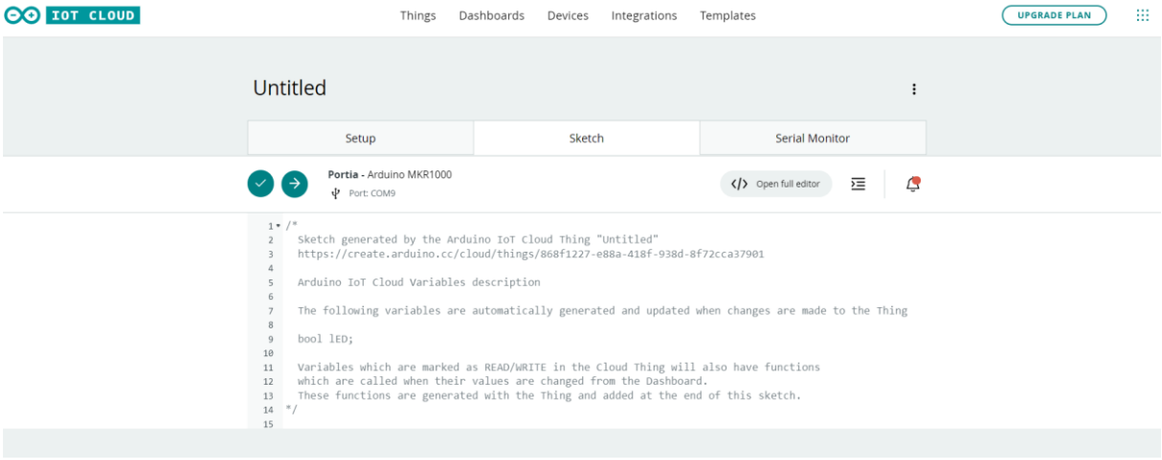


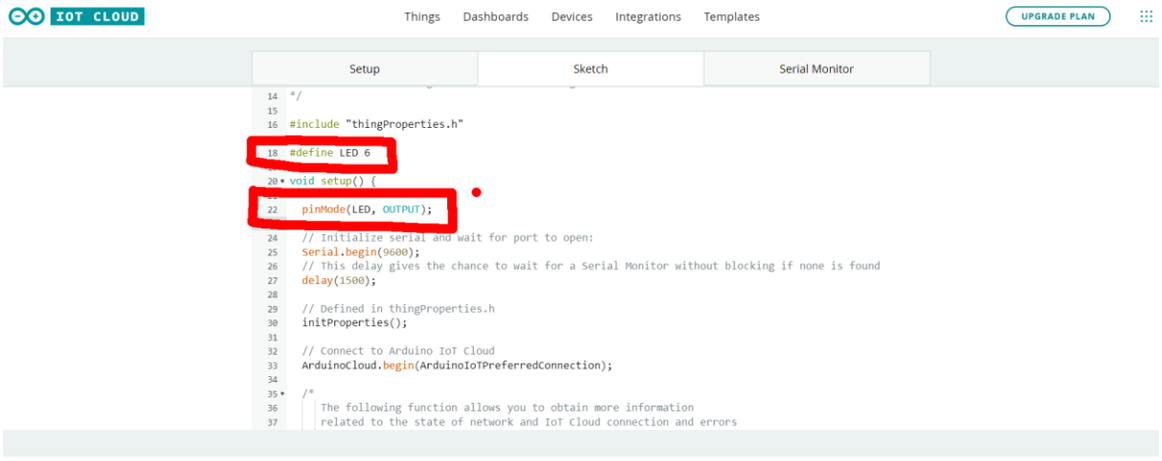


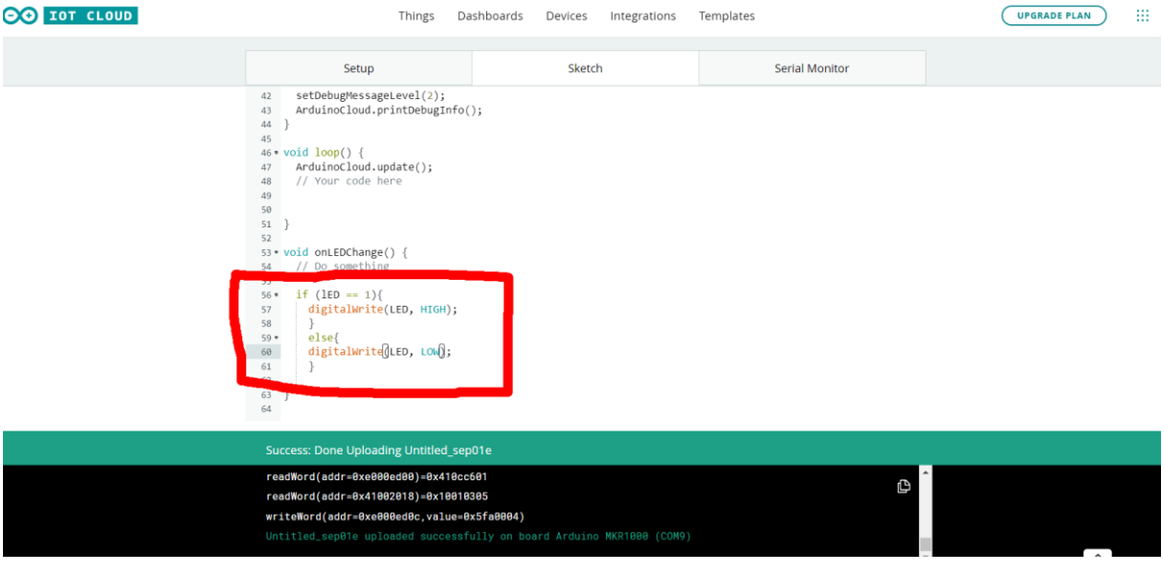


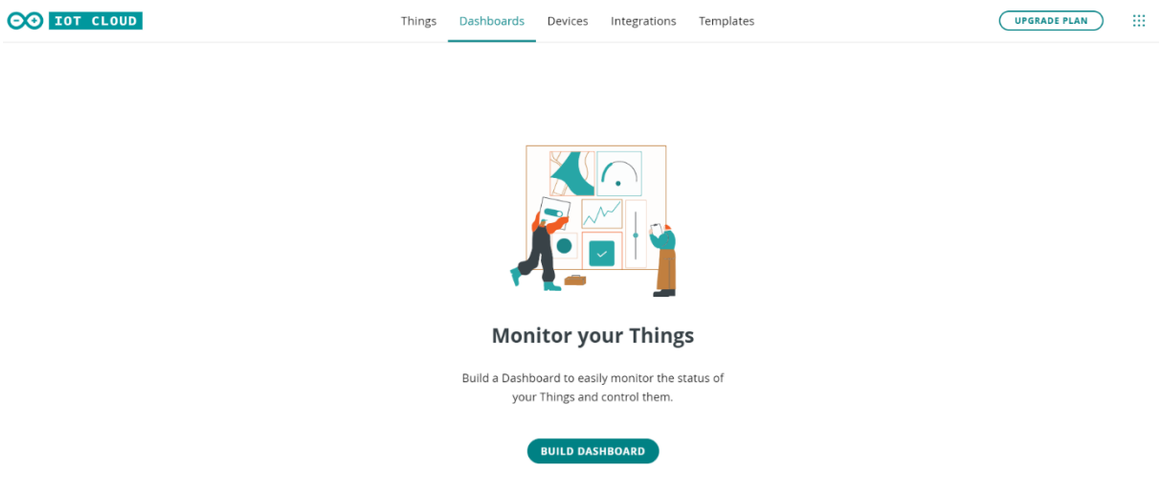

IOT CLOUD
Things
Dashboards
Devices
Integrations
Templates
UPGRADE PLAN
Setup device
Variables
Name
LED
bool lED;
Arduino MKR1000 found
An Arduino MKR1000 has been detected on port COM9 and ready to be configured.
CONFIGURE
If the detected type of the device you want to configure is not correct, try to reset your board and then refresh
Set webhook
Thing ID: 868f1227-e88a-418f-938d-8f72cca37901

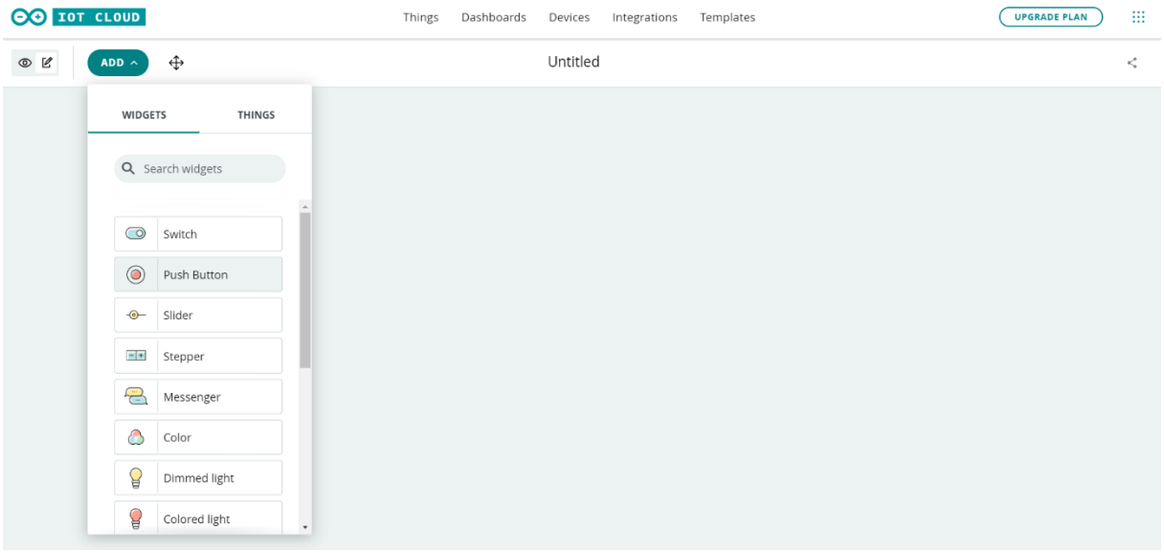

IOT CLOUD
Things
Dashboards
Devices
Integrations
Templates
UPGRADE PLAN
Setup
Sketch
Serial Monitor
Configure network
Variables
Name
LED
bool lED;
Your will find these network parameters in the secret tab in your sketch, and your device will be able to connect to the network once the sketch will be uploaded.
Name (SSID) *
mreza
Password *
SAVE
Set webhook
Thing ID: 868f1227-e88a-418f-938d-8f72cca37901

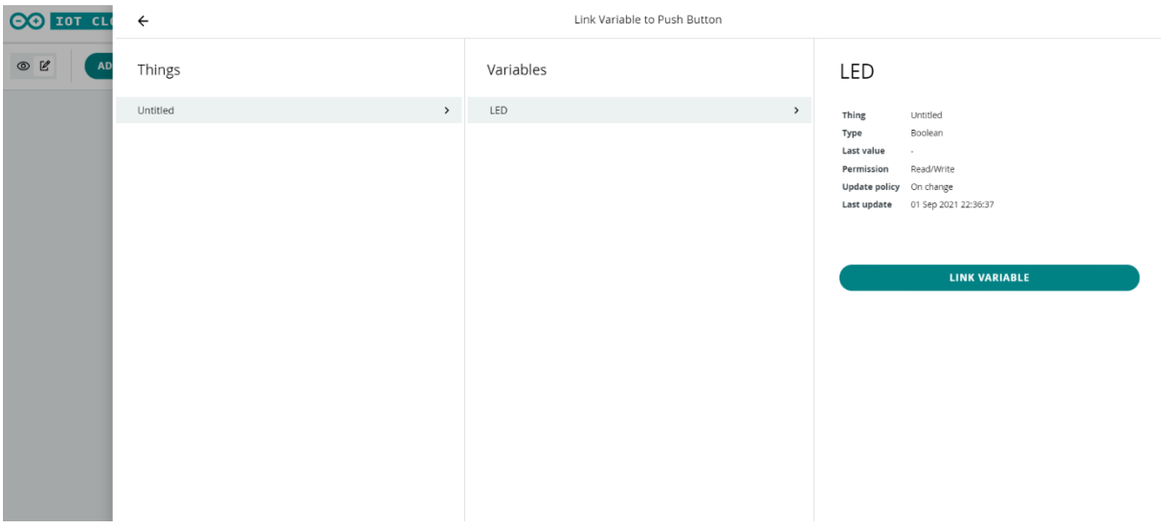

IOT CLOUD
Things
Dashboards
Devices
Integrations
Templates
UPGRADE PLAN
Untitled
Setup
Sketch
Serial Monitor
Portia - Arduino MKR1000
Port: COM9
Open full editor
/*
Sketch generated by the Arduino IoT Cloud Thing "Untitled"
https://create.arduino.cc/cloud/things/868f1227-e88a-418f-938d-8f72cca37901
Arduino IoT Cloud Variables description
The following variables are automatically generated and updated when changes are made to the Thing
bool lED;
Variables which are marked as READ/WRITE in the Cloud Thing will also have functions
which are called when their values are changed from the Dashboard.
These functions are generated with the Thing and added at the end of this sketch.
*/

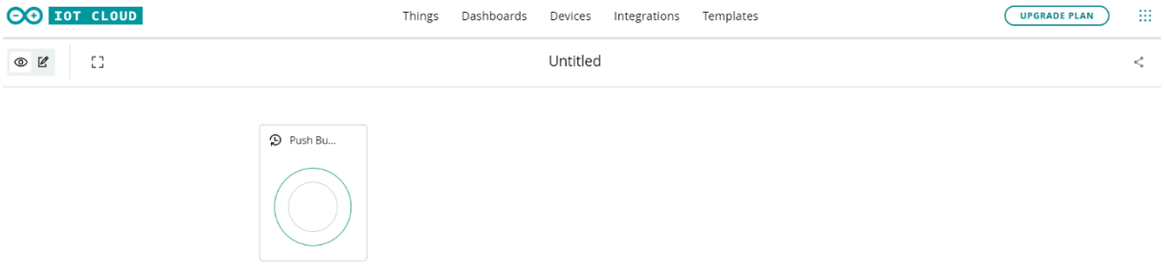

IOT CLOUD
Things
Dashboards
Devices
Integrations
Templates
UPGRADE PLAN
Setup
Sketch
Serial Monitor
*/
#include "thingProperties.h"
#define LED 6
void setup() {
pinMode(LED, OUTPUT);
// Initialize serial and wait for port to open:
Serial.begin(9600);
// This delay gives the chance to wait for a Serial Monitor without blocking if none is found
delay(1500);
// Defined in thingProperties.h
initProperties();
// Connect to Arduino IoT Cloud
ArduinoCloud.begin(ArduinoIoTPreferredConnection);
/*
The following function allows you to obtain more information
related to the state of network and IoT Cloud connection and errors


IOT CLOUD
Things
Dashboards
Devices
Integrations
Templates
UPGRADE PLAN
Setup
Sketch
Serial Monitor
setDebugMessageLevel(2);
ArduinoCloud.printDebugInfo();
}
void loop() {
ArduinoCloud.update();
// Your code here
}
void onLEDChange() {
// Do something
if (LED == 1){
digitalWrite(LED, HIGH);
}
else{
digitalWrite(LED, LOW);
}
}
Success: Done Uploading Untitled_sep01e
readWord(addr=0xe000ed00)=0x410cc601
readWord(addr=0x41002018)=0x10010305
writeWord(addr=0xe000ed0c,value=0x5fa0004)
Untitled_sep01e uploaded successfully on board Arduino MKR1000 (COM9)


IOT CLOUD
Things
Dashboards
Devices
Integrations
Templates
UPGRADE PLAN
Monitor your Things
Build a Dashboard to easily monitor the status of your Things and control them.
BUILD DASHBOARD


IOT CLOUD
Things
Dashboards
Devices
Integrations
Templates
UPGRADE PLAN
ADD
Untitled
WIDGETS
THINGS
Search widgets
Switch
Push Button
Slider
Stepper
Messenger
Color
Dimmed light
Colored light


Link Variable to Push Button
Things
Untitled
Variables
LED
LED
Thing Untitled
Type Boolean
Last value -
Permission Read/Write
Update policy On change
Last update 01 Sep 2021 22:36:37
LINK VARIABLE

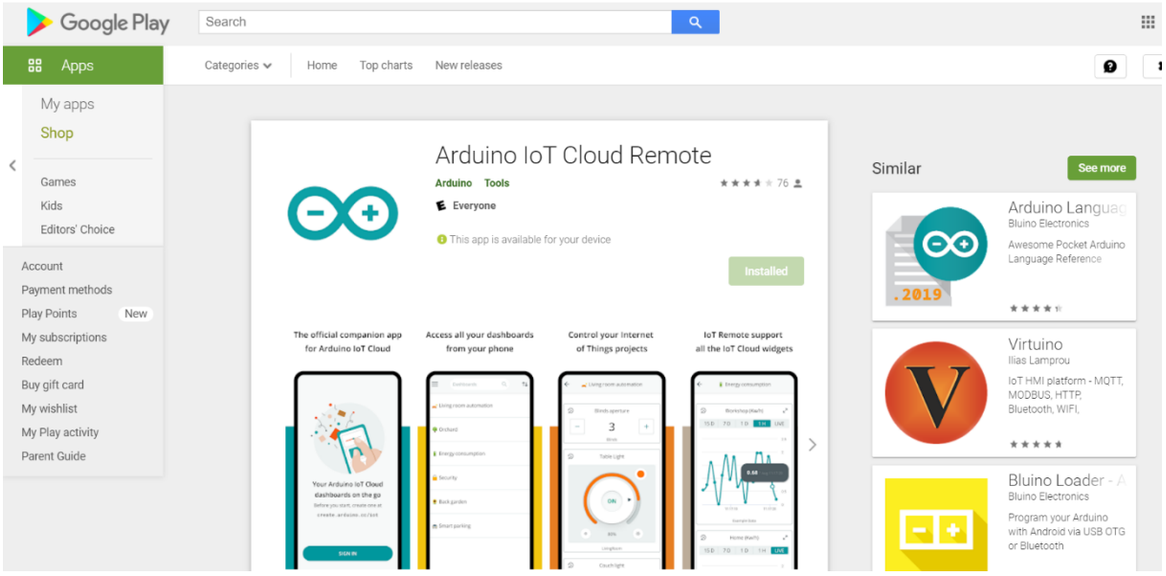

IOT CLOUD
Things
Dashboards
Devices
Integrations
Templates
UPGRADE PLAN
Untitled
Push Bu...

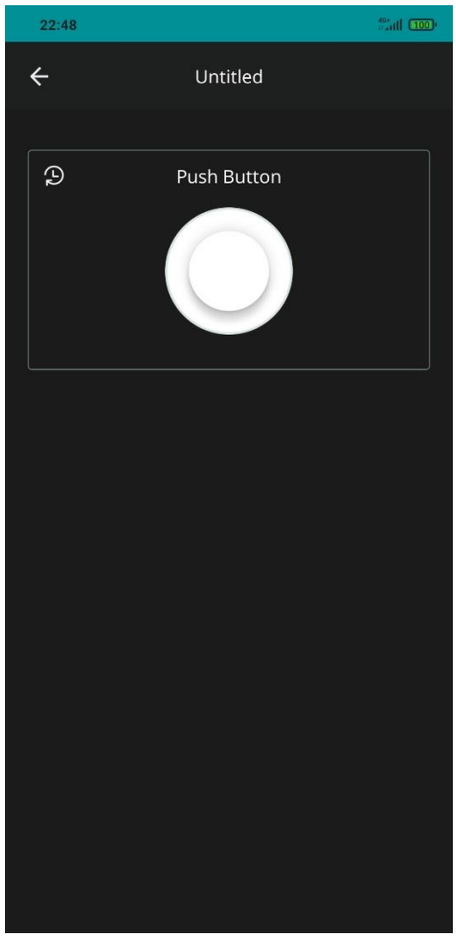
3.2.6 Arduino IoT cloud Android app

3.2.6 Arduino IoT cloud mobilna aplikacija

3.2.6 Arduino IoT cloud mobilaplikáció

Android application:
https://play.google.com/store/apps/details?id=cc.arduino.cloudiot&hl=en_US&gl=US





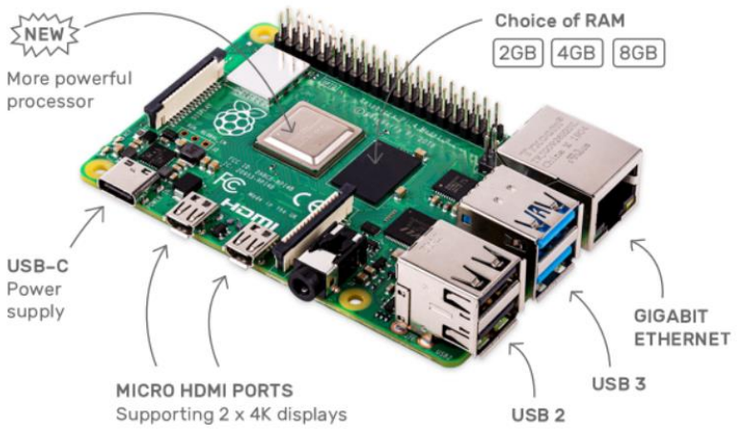
4. Raspberry Pi

"The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer monitor or TV and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python. It's capable of doing everything you'd expect a desktop computer to do, from browsing the internet and playing high-definition video, to making spreadsheets, word-processing, and playing games. What's more, the Raspberry Pi has the ability to interact with the outside world and has been used in a wide array of digital maker projects, from music machines and parent detectors to weather stations and tweeting birdhouses with infra-red cameras. We want to see the Raspberry Pi being used by kids all over the world to learn to program and understand how computers work."

„Raspberry Pi je jeftino računalo veličine kreditne kartice koje se priključuje na monitor ili televizor računala i koristi standardnu tipkovnicu i miš. Razvijeno je u Ujedinjenom Kraljevstvu od strane Raspberry Pi Foundation. Riječ je o malom uređaju koji omogućuje ljudima svih dobi da istražuju računarstvo, te da nauče programirati na jezicima kao što su Scratch i Python. Može učiniti sve što očekujete od stolnog računala, od pregledavanja interneta i reprodukcije videozapisa visoke razlučivosti, do izrade proračunskih tablica, obrade riječi i igranja igara. Štoviše, Raspberry Pi ima sposobnost interakcije s vanjskim svijetom, te je korišten u širokom rasponu projekata digitalnih proizvođača, od glazbenih strojeva i roditeljskih detektora do meteoroloških stanica i tweeting kućica za ptice s infracrvenim kamerama."

„A Raspberry Pi egy olcsó hitelkártya méretű számítógép, amely számítógép-monitorhoz vagy televízióhoz csatlakoztatható, és szabványos billentyűzetet és egeret használ. Az Egyesült Királyságban fejlesztette ki a Raspberry Pi Foundation. Ez egy kis eszköz, amely lehetővé teszi minden korosztály számára, hogy felfedezzék a számítástechnikát, és megtanuljanak programozni olyan nyelveken, mint a Scratch és a Python. Mindenre képes, ami egy asztali számítógéptől elvárható, a webböngészéstől és a nagyfelbontású videók lejátszásától a táblázatok létrehozásáig, a szövegszerkesztésig és a játékokig. Sőt, a Raspberry Pi képes kölcsönhatásba lépni a külvilággal, és számos digitális készítői projektben használták, a zenegépektől és szülői detektoroktól kezdve az időjárási állomásokig és az infravörös kamerákkal ellátott madárházakig."

www.raspberrypi.org



Raspberry pi 4 www.raspberrypi.org https://codeclubworld.org/

4.1. RPi hardware

4.1. RPi sklopovlje

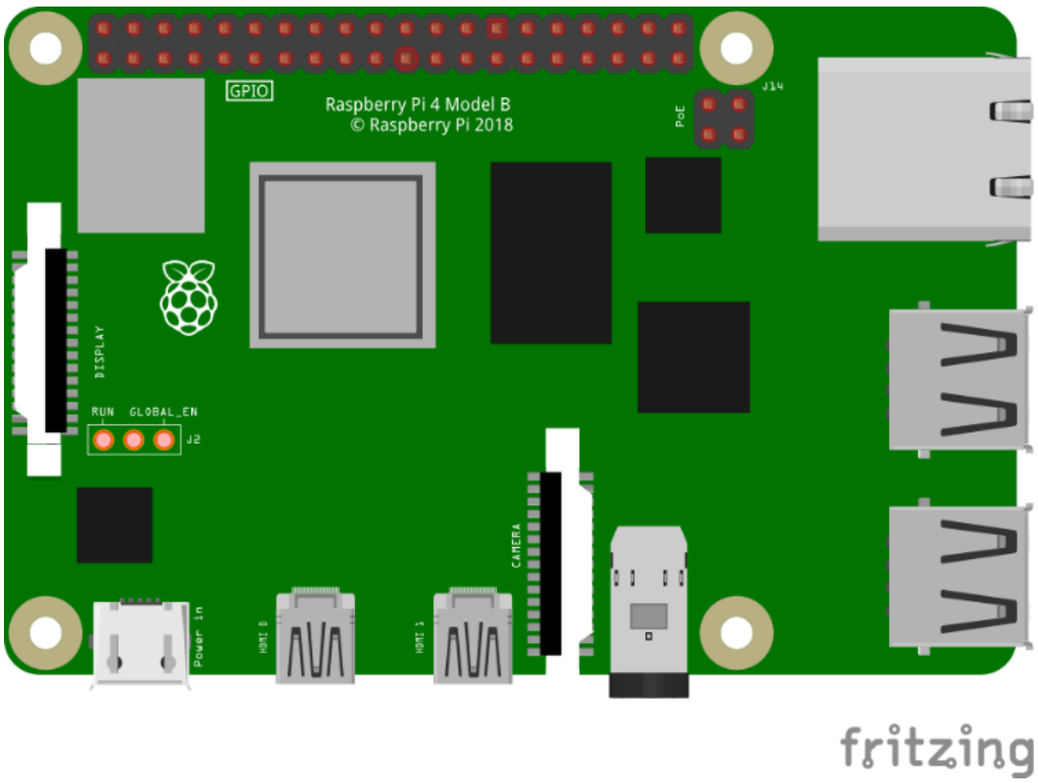
4.1. RPi áramkör

Raspberry PI is a "single board" computer, ie everything is integrated on one board. On the board we can find various "familiar" ports such as micro HDMI, USB, Ethernet, but also the so-called. GPIO port. GPIO (General-Purpose Input / Output) - connectors on which RPi can control LEDs, electric motors, read data from sensors, etc.

Raspberry PI je „single bord“ računalo, tj. sve je integrirano na jednoj pločici. Na pločici možemo pronaći razne „poznate“ priključke poput micro HDMI, USB, Ethernet, ali i tzv. GPIO priključak. GPIO (General-Purpose Input/Output) – priključci na koje RPi može upravljati sa LED, elektromotorima, čitati podatke sa senzora i sl.

A Raspberry PI egy "single board" számítógép, azaz minden egy táblára van integrálva. A táblán találhatunk különféle „ismert” csatlakozókat, mint például a micro HDMI, USB, Ethernet, de ún. GPIO portot is. GPIO (General-Purpose Input/Output) – portok, amelyeken az RPi vezérelheti a LED-eket, villanymotorokat, kiolvashatja az érzékelők adatait, stb.

https://www.raspberrypi.org/documentation/computers/os.html#outputs

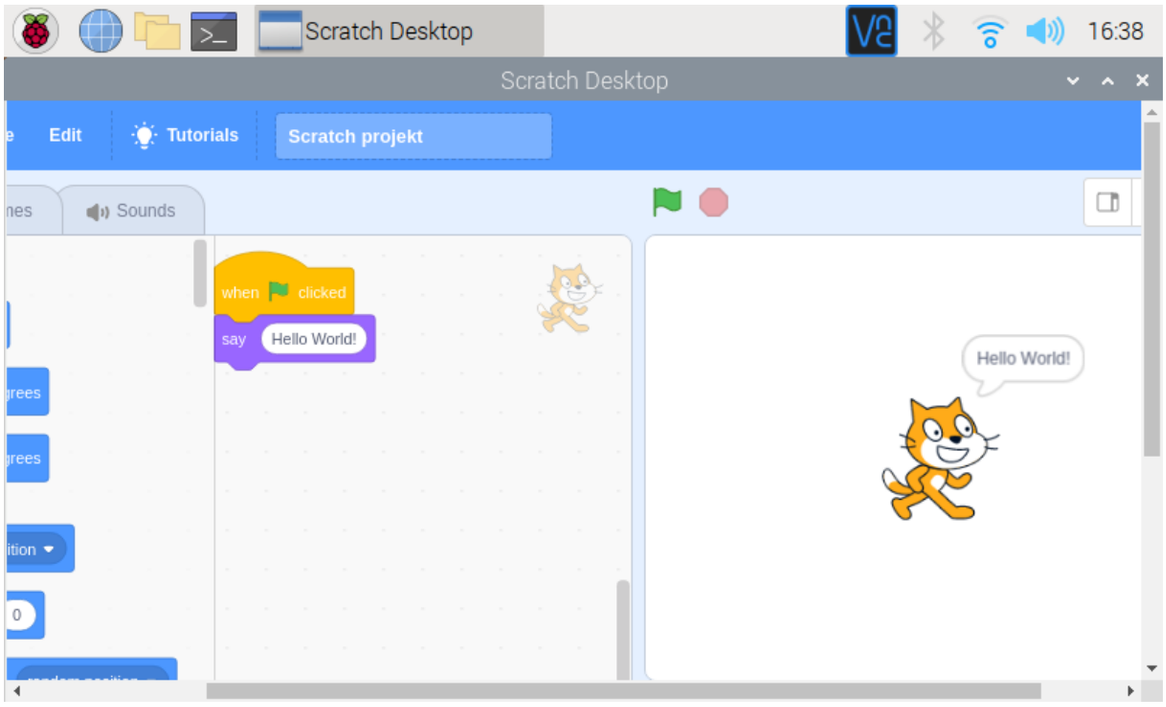


4.2. RPi programming - Scratch 3 and Python

Scratch – visual programming developed at the Massachusetts Institute of Technology (MIT), uses blocks of pre-written code that we put together as pieces of a puzzle.

Scratch – vizualno programiranje razvijeno u Massachusetts Institute of Technology (MIT), koristi blokove unaprijed napisanog koda koje slažemo kao dijelove slagalice.

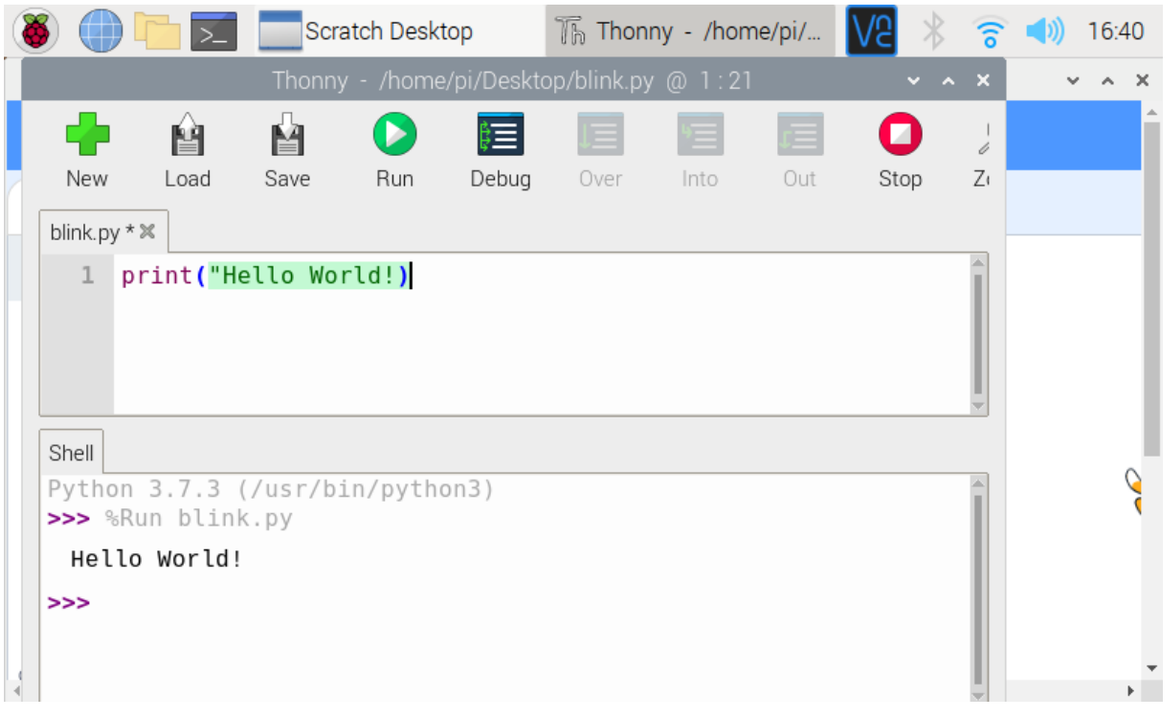
A Scratch – a Massachusetts Institute of Technology (MIT) által kifejlesztett vizuális programozás előre megírt kódblokkokat használ, amelyeket puzzle-darabokként rakunk össze.



Python - a programming language by Guido van Rossum named after the comedy company Monty Python, is a great continuation of programming after mastering programming in Scratch.

Python – programski jezik autora Guida van Rossuma nazvan po komičarskoj družini Monty Python, sjajan je nastavak programiranja nakon savladavanja programiranja u Scratch-u.

Python – Guido van Rossum programozási nyelve, amely a Monty Python komikus társulat után kapta a nevét, remek folytatása a programozásnak a Scratch programozás elsajátítása után.

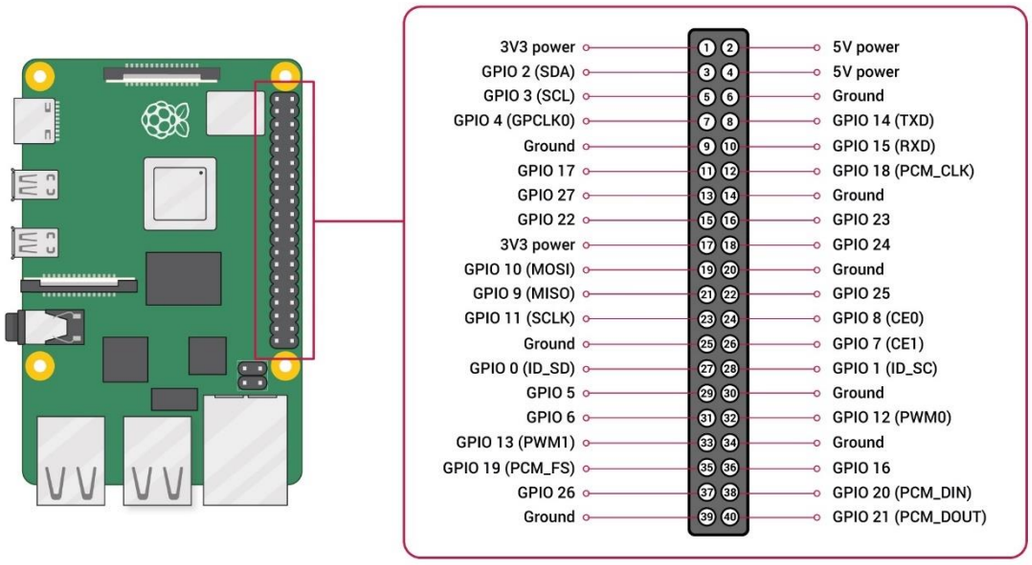


4.3. GPIO - Scratch 3 and Python

As already mentioned Raspberry PI has a so-called. GPIO - general-purpose Input / Output, ie general purpose input / output interface. It is a 40-pin connector with which we can control external devices (everything that an Arduino board can do): for example, we can turn on / off the lights, control the vehicle, read data from various sensors, etc.

Kao što je već spomenuto Raspberry PI ima tzv. GPIO – general-purpose Input/Output, odnosno ulazno/izlazno sučelje opće namjene. To je priključak od 40 pinova pomoću kojega možemo upravljati vanjskim uređajima (sve što može i Arduino pločica): npr. možemo paliti/gasiti svjetla, upravljati vozilom, čitati podatke sa raznih senzora i sl.

Mint már említettük, a Raspberry PI ún GPIO – általános célú Input/Output, azaz általános célú bemeneti/kimeneti interfész. Ez egy 40 tűs csatlakozó, amivel külső eszközöket vezérelhetünk (minden, amire az Arduino tábla képes): például fel-/lekapcsolhatjuk a lámpákat, vezethetjük a járművet, kiolvashatjuk a különböző szenzoroktól származó adatokat, stb.

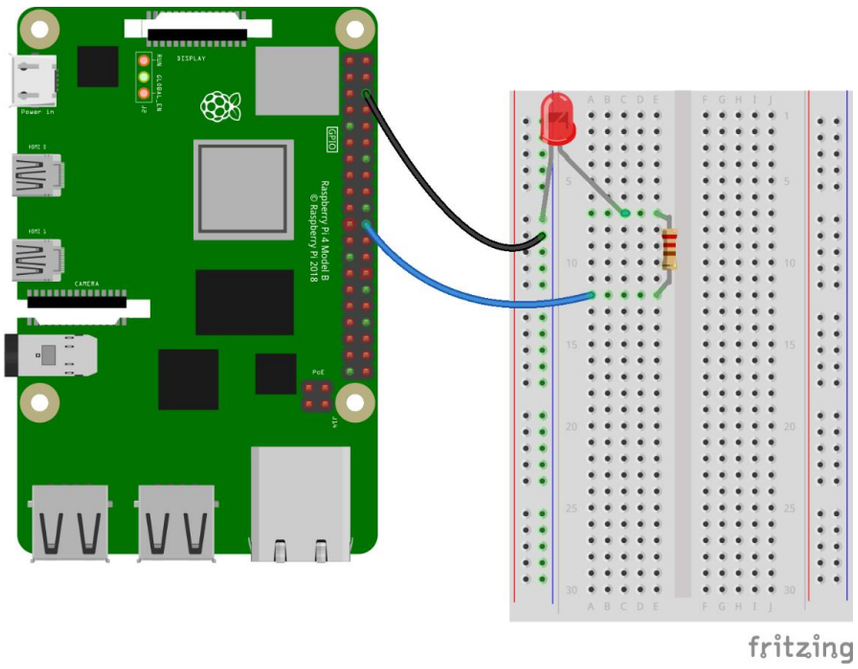


https://www.raspberrypi.org/documentation/computers/images/GPIO-Pinout-Diagram-2.png

Example: Turning the LED connected to pin25 (GPIO) on / off.

Primjer: Paljenje/gašenje LED povezane na pin25 (GPIO).

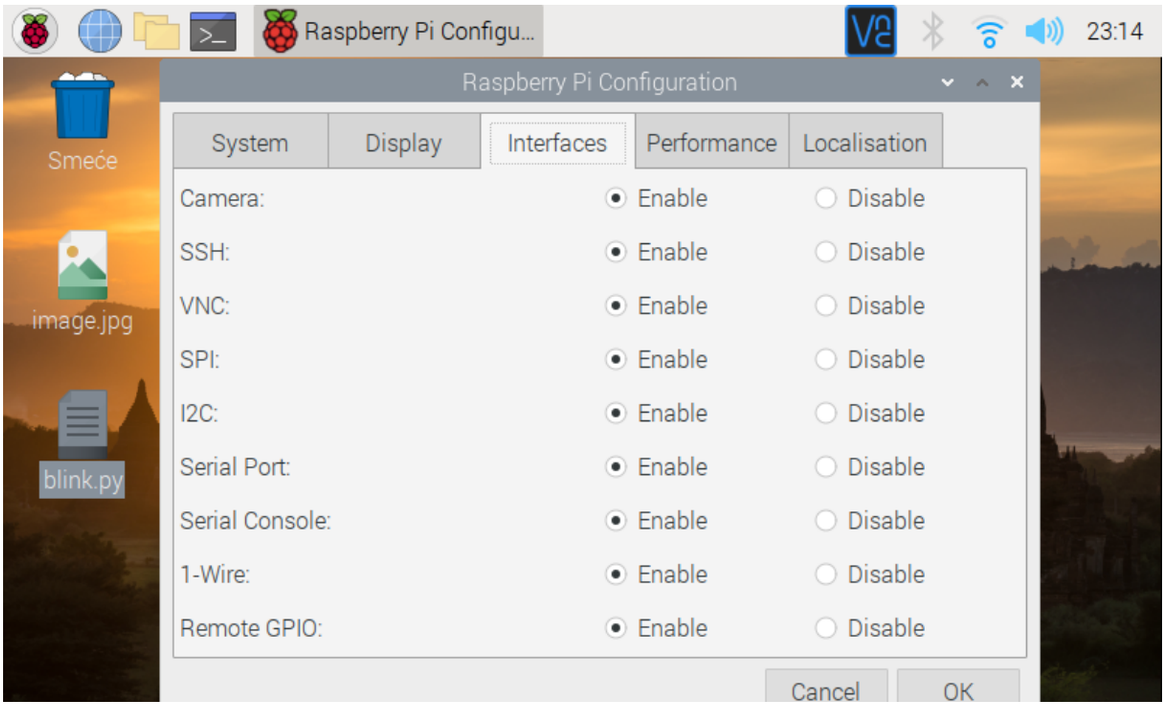
Példa: A LED be/ki kapcsolása a 25-ös érintkezőhöz csatlakozik (GPIO).



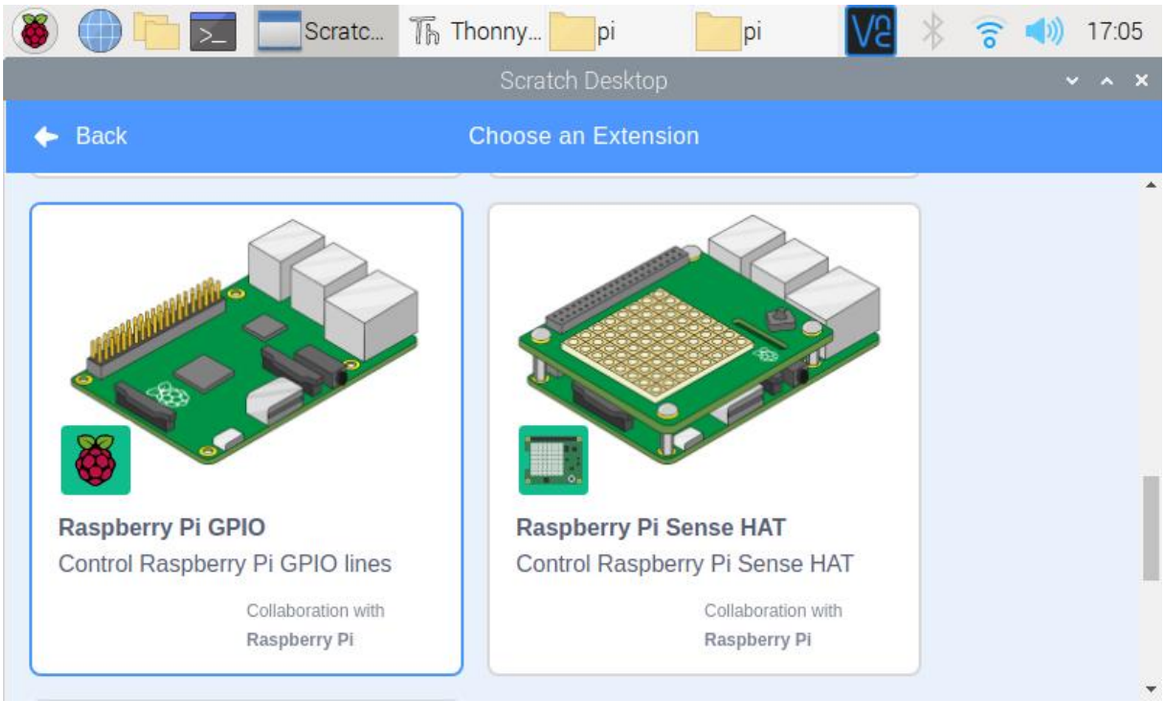
In order to use GPIO in Scratch 3 we need to add the extension "Raspberry Pi GPIO"!

Da bismo mogli koristiti GPIO u Scratch 3 moramo dodati proširenje „Raspberry Pi GPIO“!

A GPIO használatához a Scratch 3-ban hozzá kell adnunk a „Raspberry Pi GPIO” kiterjesztést!



Scratch 3:



Python:

To use GPIO from Python we need a library called GPIO zero, and for this example we need a library to work with LED:

```
from gpiozero import LED
```

and the time management library:

```
from time import sleep
```

Da bismo koristili GPIO iz Pythona potrebna na nam je knjižnica koja se zove GPIO zero, a za ovaj primjer nam je potrebna knjižnica i za rad sa LED:

```
from gpiozero import LED
```

te knjižnica za upravljanje vremenom:

```
from time import sleep
```

A Python GPIO használatához szükségünk van egy GPIO zero nevű könyvtárra, és ebben a példában szükségünk van egy olyan könyvtárra is, amely LED-del működik:

```
from gpiozero import LED
```

és az időgazdálkodási könyvtár:

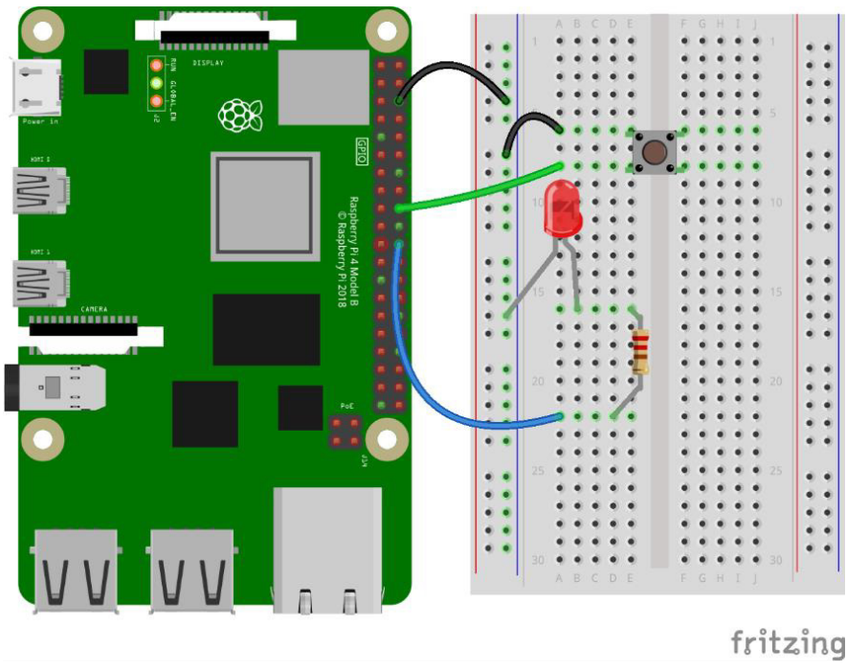
```
from time import sleep
```

```
from gpiozero import LED
from time import sleep
led = LED(25)
while True:  #infinite loop
    led.on()
    sleep(1)
    led.off()
    sleep(1)
```

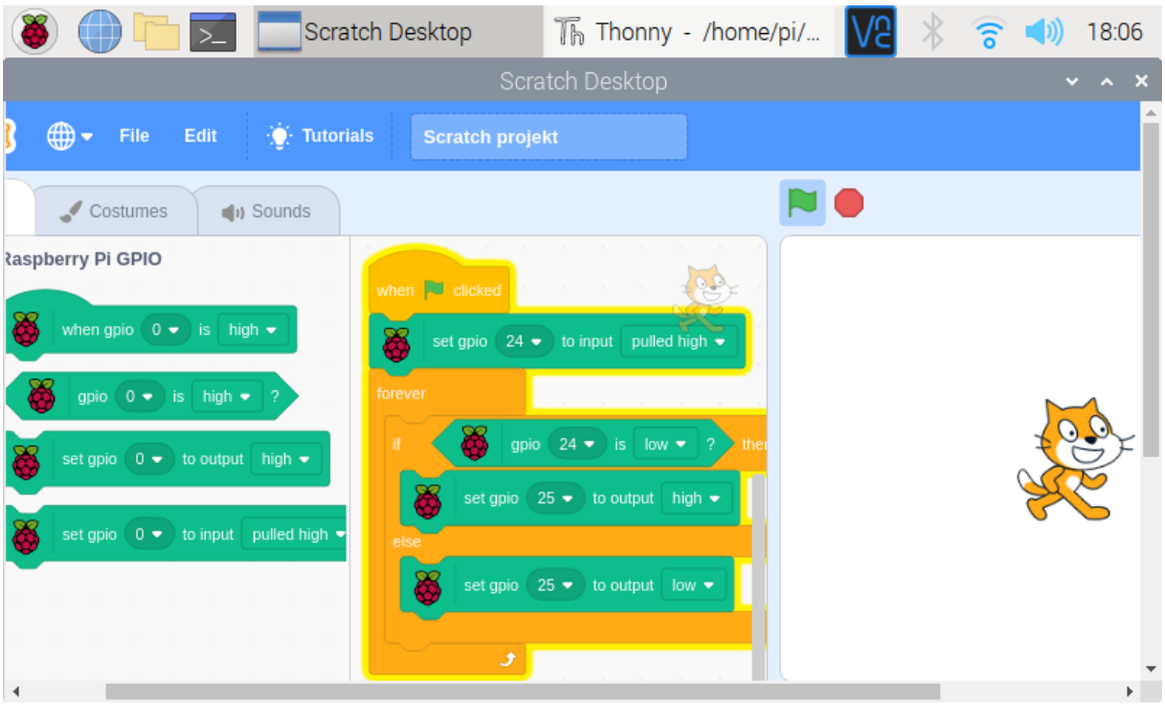
Example: Switching on the LED (pin25) if the button (pin24) is pressed.

Primjer: Uključivanje LED (pin25) ako je pritisnuta tipka (pin24).

Példa: A LED (25-ös tű) bekapcsolása, ha megnyomja a gombot (pin24).



Scratch:



Python:

```
from gpiozero import LED, Button
led = LED(25)
button = Button(24)
while True:  #infinite loop
    if button.is_pressed:  #if the pushbutton is pressed
        led.on() #LED ON
    else:         #else
        led.off() #LED OFF
```

4.3. Camera module

We will use a module with a camera called camera module V2 based on Sony IMX219 8MPX resolution sensor, which connects to the Raspberry Pi to the intended camera port. To be able to use the camera, we need to enable it in the "Raspberry Pi Configuration" settings:

Koristit će modul sa kamerom tzv. camera module V2 baziran na Sony IMX219 senzoru rezolucije 8MPX, koji se povezuje sa Raspberry Pi na predviđeni priključak za kameru. Da bismo mogli koristit kameru, moramo to omogućiti u postavkama „Raspberry Pi Configuration“:

Egy modult fog használni egy kamerával a Sony IMX219 érzékelőn alapuló V2 kameramodul 8MPX felbontással, amely a mellékelt kameraporton csatlakozik a Raspberry Pi-hez. A kamera használatához engedélyeznünk kell a „Raspberry Pi Configuration” beállításoknál:

Camera control with Python / Kontrola kamere pomoću Pythona / Kamera vezérlése a Python használatával:

```
from picamera import PiCamera
from time import sleep
camera = PiCamera()

camera.start_preview()
sleep(10)
camera.stop_preview()
```

We should see what the camera is recording on the screen. If necessary, we can rotate the screen by 90 °, 180 °, 270 °, e.g.

Na zaslonu bismo trebali vidjeti ono što kamera snima. Ako je potrebno, možemo rotirati zaslon za 90°, 180°, 270°, npr.

A képernyőn látnunk kell, hogy mit rögzít a kamera. Ha szükséges, elforgathatjuk a képernyőt 90°, 180°, 270°, pl.

```
camera.rotation=180
```

Photography / Fotografija / Fotó:

```
camera.capture('/home/pi/Desktop/image.jpg')
```

Video:

```
camera.start_recording('/home/pi/Desktop/video.h264')
sleep(10)
camera.stop_recording()
```

Example: Taking photos by pressing the button connected to pin24. Photos will be stored on the desktop.

Primjer: Snimanje fotografija pritiskom na tipku spojenu na pin24. Fotografije će biti pohranjene na radnu površinu.

Példa: Fényképezés a pin24-hez csatlakoztatott gomb megnyomásával. A fotók az asztalra kerülnek.

```
from picamera import PiCamera
from gpiozero import Button
camera = PiCamera()
button = Button(24)

camera.start_preview()
button.wait_for_press()
camera.capture('/home/pi/Desktop/image.jpg')
camera.stop_preview()
```

We can use the PiCamera library to access various settings (some of them):

Knjižnicu PiCamera možemo koristiti za pristup raznim postavkama (neke od njih):

A PiCamera könyvtár segítségével különféle beállításokat érhetünk el (néhány közülük):

https://magpi.raspberrypi.org/books/beginners-guide-4th-ed

Automatic white balance mode (modes: off, auto, sunlight, cloudy, shade, tungsten, fluorescent, incandescent, flash, or horizon)
camera.awb_mode = 'auto'
Brightness (0 do 100)
camera.brightness = 50
Contrast (0 do 100)
camera.contrast = 0
Exposure (0 do 100)
camera.exposure_compensation = 0
camera.exposure_mode = 'auto'
ISO settings (100, 200, 320, 400, 500, 640, 800) - the higher the value the camera will work better in low light conditions, but then the image is grainier.
camera.ISO = 0
Resolution
camera.resolution = (1920, 1080)

Automatski način balansa bijele boje (modovi: off, auto, sunlight, cloudy, shade, tungsten, fluorescent, incandescent, flash, or horizon)
camera.awb_mode = 'auto'
Svjetlina (0 do 100)
camera.brightness = 50
kontrast (0 do 100)
camera.contrast = 0
ekspozicija (0 do 100)
camera.exposure_compensation = 0
camera.exposure_mode = 'auto'
ISO postavke (100, 200, 320, 400, 500, 640, 800) – što je vrijednost veća kamera će bolje raditi u uvjetima slabijeg osvjetljenja, ali je onda slika zrnatija.
camera.ISO = 0
Rezolucija fotografije
camera.resolution = (1920, 1080)

Automatikus fehéregyensúly mód (módozatok: off, auto, sunlight, cloudy, shade, tungsten, fluorescent, incandescent, flash, or horizon)
camera.awb_mode = 'auto'
világosság (0 do 100)
camera.brightness = 50
kontraszt (0 do 100)
camera.contrast = 0
ekszpozíció (0 do 100)
camera.exposure_compensation = 0
camera.exposure_mode = 'auto'
ISO beállítások (100, 200, 320, 400, 500, 640, 800) – minél nagyobb az érték, annál jobban fog működni a kamera gyenge fényviszonyok mellett is, de ekkor szemcsésebb lesz a kép.
camera.ISO = 0
a fénykép felbontása
camera.resolution = (1920, 1080)

5. Robotic arm
5.Robotska ruka
5. Robotkar

www.arduino.cc/products

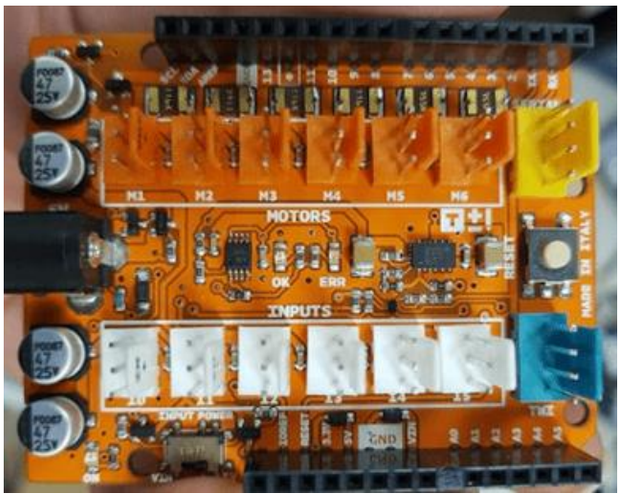
5.2. Assembling / Sastavljanje/ Összeszerelés

https://www.arduino.cc/en/Guide/Braccio

The robotic arm must be assembled according to the enclosed instructions, and special attention should be paid to the cables that need to be routed through the middle of the arm structure and the exact arrangement of the servo motor of the arm. Looking from the bottom up, the motors are marked with M1, M2, M3, M4, M5 and M6, so they should be connected to the marked places on the enclosed plate where the motor drivers are located and to which the AC adapter is connected.

Robotsku ruku potrebno je sastaviti prema priloženim uputama, a posebno treba obratiti pažnju na kabele koje je potrebnu provući kroz sredinu konstrukcije ruke te točan raspored servo motora ruke. Gledajući odozdo prema gore motori su označeni sa M1, M2, M3, M4, M5 i M6, te ih tako treba priključiti na označena mjesta na priloženoj pločici na kojoj se nalaze driveri za motore i na koju se povezuje AC adapter.

A robotkart a mellékelt utasítások szerint kell összeállítani, és kiemelt figyelmet kell fordítani a kar szerkezetének közepén átvezetendő kábelekre és a kar szervomotorjának pontos elrendezésére. Alulról felfelé nézve a motorok M1, M2, M3, M4, M5 és M6 jelzéssel vannak ellátva, így azokat a mellékelt táblán azokra a megjelölt helyekre kell kötni, ahol a motorok meghajtói találhatók, és ahova a váltóáramú adapter csatlakoztatva van.



https://www.arduino.cc/en/Guide/Braccio

The board for connecting the motor and power supply of the robotic arm is intended for direct connection to Arduino UNO (Uno SMD, Uno WiFi, Due, Mega 2560, Ethernet, Leonardo, Leonardo ETH, M0, M0 Pro, Yun *, Tian *) boards. However, we can connect it with other tiles and then we need to pay attention to the following layout of connectors:

Pločica za priključak motora i napajanja robotske ruke predviđena je za izravno priključivanje na Arduino UNO (Uno SMD, Uno WiFi, Due, Mega 2560, Ethernet, Leonardo, Leonardo ETH, M0, M0 Pro, Yun*, Tian*) pločice. Međutim, možemo ju povezati i sa drugim pločicama a onda je potrebno obratiti pažnju na sljedeći raspored priključaka:

A robotkar motorja és tápegysége az Arduino UNO (Uno SMD, Uno WiFi, Due, Mega 2560, Ethernet, Leonardo, Leonardo ETH, M0, M0 Pro, Yun*, Tian*) kártyához való közvetlen csatlakozásra szolgál. . Azonban más csempékhez is csatlakoztathatjuk, és ekkor a csatlakozások alábbi elrendezésére kell ügyelni:

Connector Name	**Shield pin**
M1	11
M2	10
M3	9
M4	6
M5	5

Connector Name	Shield pin
M6	3
I0	A0 (aka 14)
I1	A1 (15)
I2	A2 (16)
I3	A3 (17)
I4	A4 (18)
I5	A5 (19)
TWI	SCL, SDA
Serial	RX0, TX0

Although other motors will not be connected to this board, the following board limitations should be noted:

- M1 through M4 are limited to 1.1A
- M5 and M6 are limited to 750mA

Iako se na navedenu pločicu neće povezivati drugi motori, treba imati na umu sljedeća ograničenja pločice:

- M1 do M4 ograničeni su na 1.1A
- M5 i M6 ograničeni su na 750mA

Bár ehhez a kártyához más motor nem csatlakoztatható, a következő korlátozásokat kell szem előtt tartani:

- M1-től M4-ig korlátozva vannak 1.1A
- M5 és M6 korlátozva vannak 750mA

5.3. Programming

5.3. Programiranje

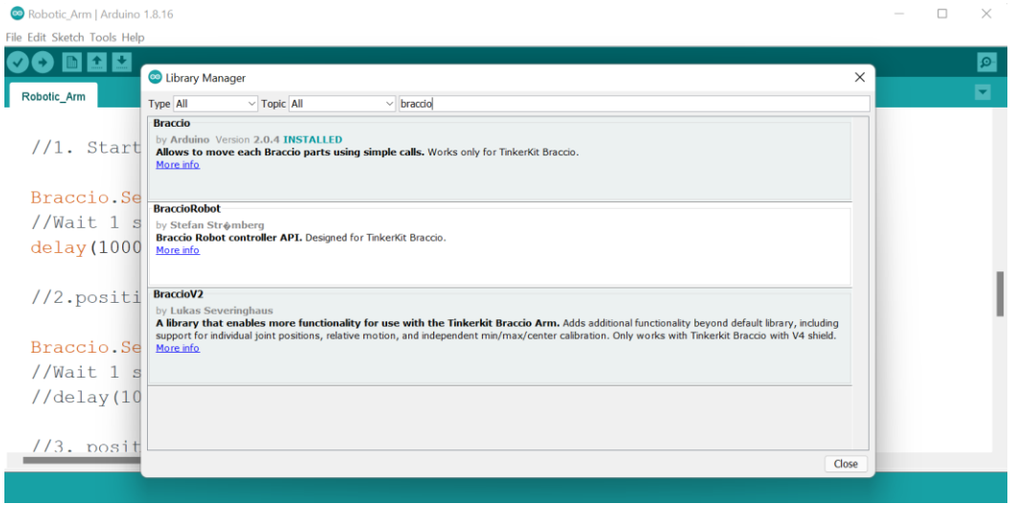
5.3. Programozás

In order to be able to use ready-made code examples and more easily program a robot, we need to include the necessary libraries:

Da bismo mogli koristiti gotove primjere koda i lakše programirali robotsku moramo uključiti potrebne biblioteke:

A kész kódpéldák használatához és a robotkar egyszerűbb programozása érdekében be kell építeni a szükséges könyvtárakat:

Sketch > Include library > Manage libraries

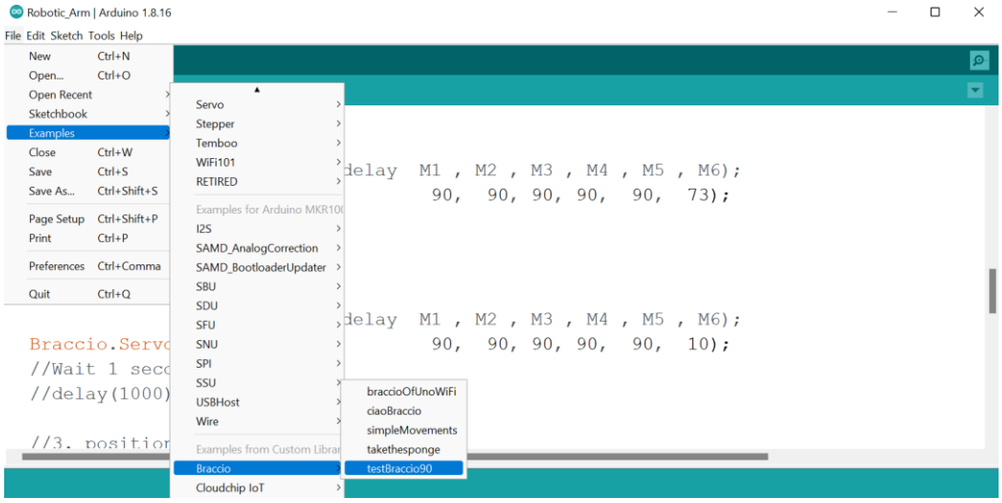


After installing the library we can see the following examples:

Nakon instalacije biblioteke možemo vidjeti sljedeće primjere:

A könyvtár telepítése után a következő példákat láthatjuk:

File > Exemples > Braccio



6. CNC machine
6. CNC uređaj
6. CNC eszköz

PRECAUTIONS!

The CNC device has moving and other parts that can injure the person handling it, and by careless handling cause damage to the CNC device itself, ie the object being processed. Before using the CNC device, read the manufacturer's instructions and use the device in compliance with all precautionary measures prescribed by the manufacturer.

It is recommended that students prepare files with instructions according to which the CNC device should process the subject, and then the teacher should turn on the CNC device for processing the subject. Handling of the CNC device should be done exclusively by teachers, and the CNC device and other parts of it should not be accessible to students without teacher supervision.

The CNC device must be assembled according to the enclosed manufacturer's instructions.

MJERE OPREZA!

CNC uređaj ima pokretne i ostale dijelove koji mogu ozlijediti osobu koja njime rukuje, te nepažljivim rukovanjem nanijeti štetu samom CNC uređaju, odnosno predmetu kojeg se obrađuje. Prije korištenja CNC uređaja pročitajte upute proizvođača, te uređaj koristite poštujući sve mjere opreza koje je propisao proizvođač.

Preporuka je da učenici pripreme datoteke sa naredbama prema kojima CNC uređaj treba obraditi predmet, te da onda nastavnik uključi CNC uređaj za obradu predmeta. Rukovanje sa CNC uređajem trebali bi odraditi isključivo nastavnici, te CNC uređaj i ostali njegovi dijelovi ne bi smjeli biti dostupni učenicima bez nadzora nastavnika.

CNC uređaj potrebno je sastaviti prema priloženim uputama proizvođača.

ÓVINTÉZKEDÉSEK!

A CNC-eszköznek vannak mozgó és egyéb részei, amelyek sérülést okozhatnak a kezelőben, a gondatlan kezelés pedig magában a CNC-eszközben, vagyis a megmunkálás alatt álló tárgyban is megsérülhet. A CNC eszköz használata előtt olvassa el a gyártó utasításait, és használja a készüléket a gyártó által előírt óvintézkedések betartásával.

Javasoljuk, hogy a tanulók készítsenek fájlokat olyan parancsokkal, amelyek szerint a CNC-eszköz feldolgozza az objektumot, majd a tanár bekapcsolja a CNC-eszközt az objektum feldolgozásához. A CNC-eszköz kezelését kizárólag tanárok végezhetik, a CNC-eszközt és annak egyéb alkatrészeit tanári felügyelet nélkül ne férhessenek hozzá a tanulók.

A CNC készüléket a mellékelt gyártói utasítások szerint kell összeszerelni.

Material processing on this CNC device can be done using milling cutters, drills and lasers (depending on what we want to achieve). The file that contains instructions on how the CNC machine will process the material is called the G-code. G-code can be obtained (converted) from a .STL (.OBJ, .GLB) file, ie from standard files we have created for 3D printing (eg in thinkercad.com).

There are quite a few free (open source) conversion programs, even online, and one of them is Ultimaker Girl https://ultimaker.com/software/ultimaker-cura

The conversion process itself is very simple and comes down to opening the .STL file and saving to G-code.

Obrada materijala na ovom CNC uređaju može se vršiti pomoću glodala, svrdla i lasera (ovisno što želimo postići). Datoteka koja sadrži upute kako će CNC uređaj obrađivati materijal zove se G-kod. G-kod možemo dobiti (pretvoriti) iz .STL (.OBJ, .GLB) datoteke, dakle iz standardnih datoteka koje smo napravili za 3D ispis (npr. U thinkercad.com).

Besplatnih (otvorenoga koda) programa za konverziju ima prilično, čak i online, a jedan od njih je Ultimaker Cura https://ultimaker.com/software/ultimaker-cura

Sam proces konverzije je vrlo jednostavan i svodi se na otvaranje .STL datoteke i spremanje u G-kod.

Az anyagfeldolgozás ezen a CNC-eszközön maróval, fúróval és lézerrel végezhető (attól függően, hogy mit szeretnénk elérni). Az a fájl, amely utasításokat tartalmaz arra vonatkozóan, hogy a CNC gép hogyan dolgozza fel az anyagot, G-kódnak nevezik. A G-kód beszerezhető (konvertálható) .STL (.OBJ, .GLB) fájlból, azaz olyan szabványos fájlokból, amelyeket 3D nyomtatáshoz készítettünk (pl. a thinkercad.com-ban).

Van jó néhány ingyenes (nyílt forráskódú) konvertáló program, még online is, ezek közül az egyik az Ultimaker Cura https://ultimaker.com/software/ultimaker-cura

Maga az átalakítási folyamat nagyon egyszerű, és az STL fájl megnyitásában és G-kódban való mentésében merül ki.

6.1. Engraving and milling
6.1. Graviranje glodalom
6.1. Gravirozás marógéppel

PRECAUTIONS!

CNC milling cutters can be very sharp, and in combination with moving parts of the CNC device can cause injuries. Before using the CNC device, read the manufacturer's instructions and use the device in compliance with all precautionary measures prescribed by the manufacturer.

The cutter set comes in a CNC package!

MJERE OPREZA!

Glodala za CNC uređaj mogu biti vrlo oštra, te u kombinaciji sa pomičnim dijelovima CNC uređaja mogu nanijeti ozljede. Prije korištenja CNC uređaja pročitajte upute proizvođača, te uređaj koristite poštujući sve mjere opreza koje je propisao proizvođač.

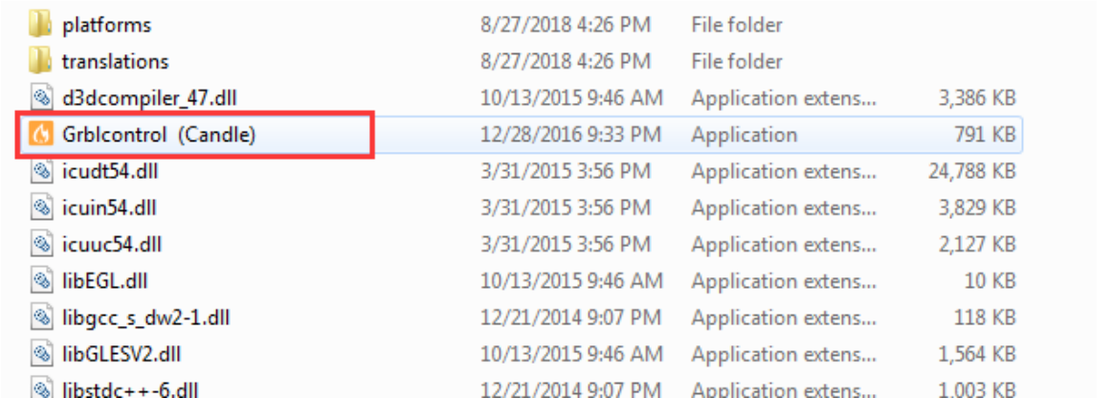
Set glodala dolazi u pakiranju CNC uređaja!

ÓVINTÉZKEDÉSEK!

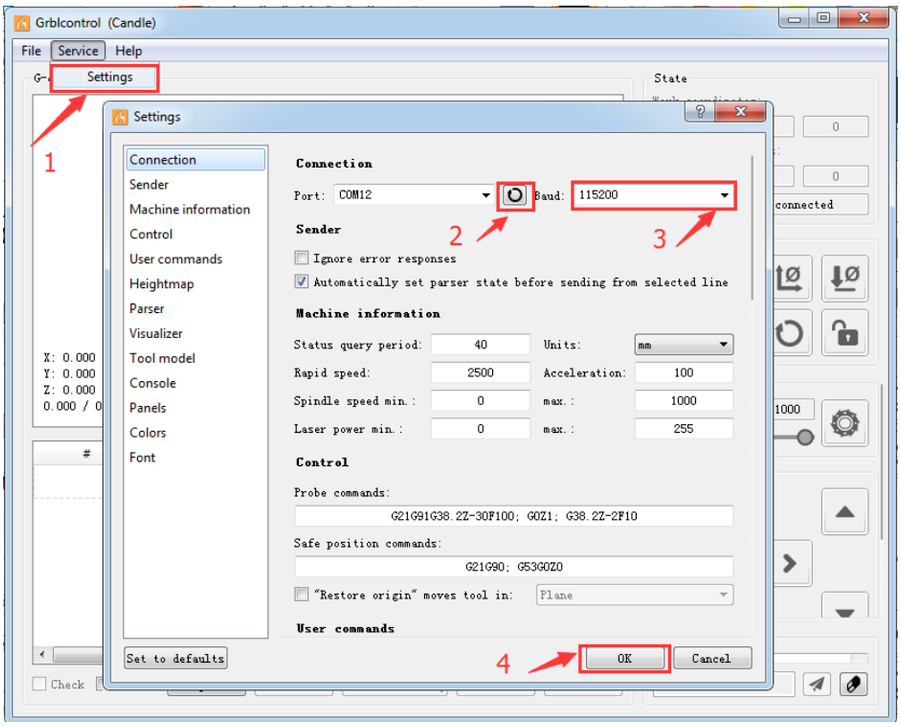
A CNC-eszközök marószerszámai nagyon élesek lehetnek, és a CNC-eszközök mozgó alkatrészeivel kombinálva sérüléseket okozhatnak. A CNC eszköz használata előtt olvassa el a gyártó utasításait, és használja a készüléket a gyártó által előírt óvintézkedések betartásával.

A marókészlet CNC készülékcsomagban érkezik!

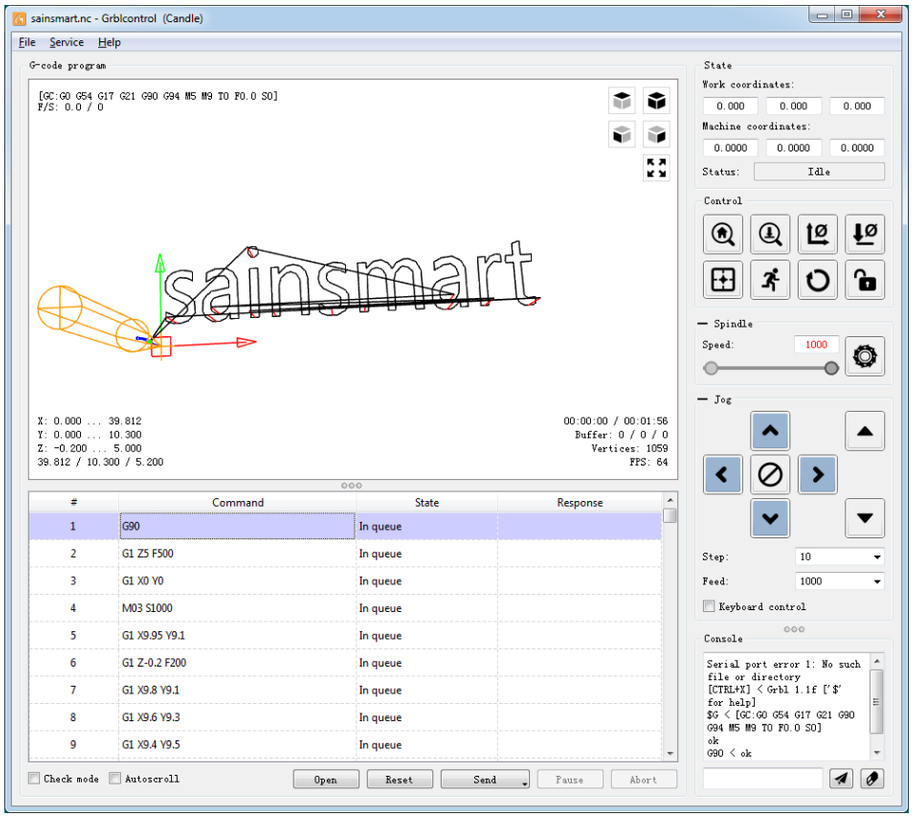
1. Program launches / Pokretanje programa / Program elindítása



2. Connecting PC with CNC machine / Povezivanje PC-a sa kontrolerom CNC uređaja / A PC csatlakoztatása a CNC eszközvezérlőhöz



3. Interface description / Opis sučelja / A munkafelület leírása

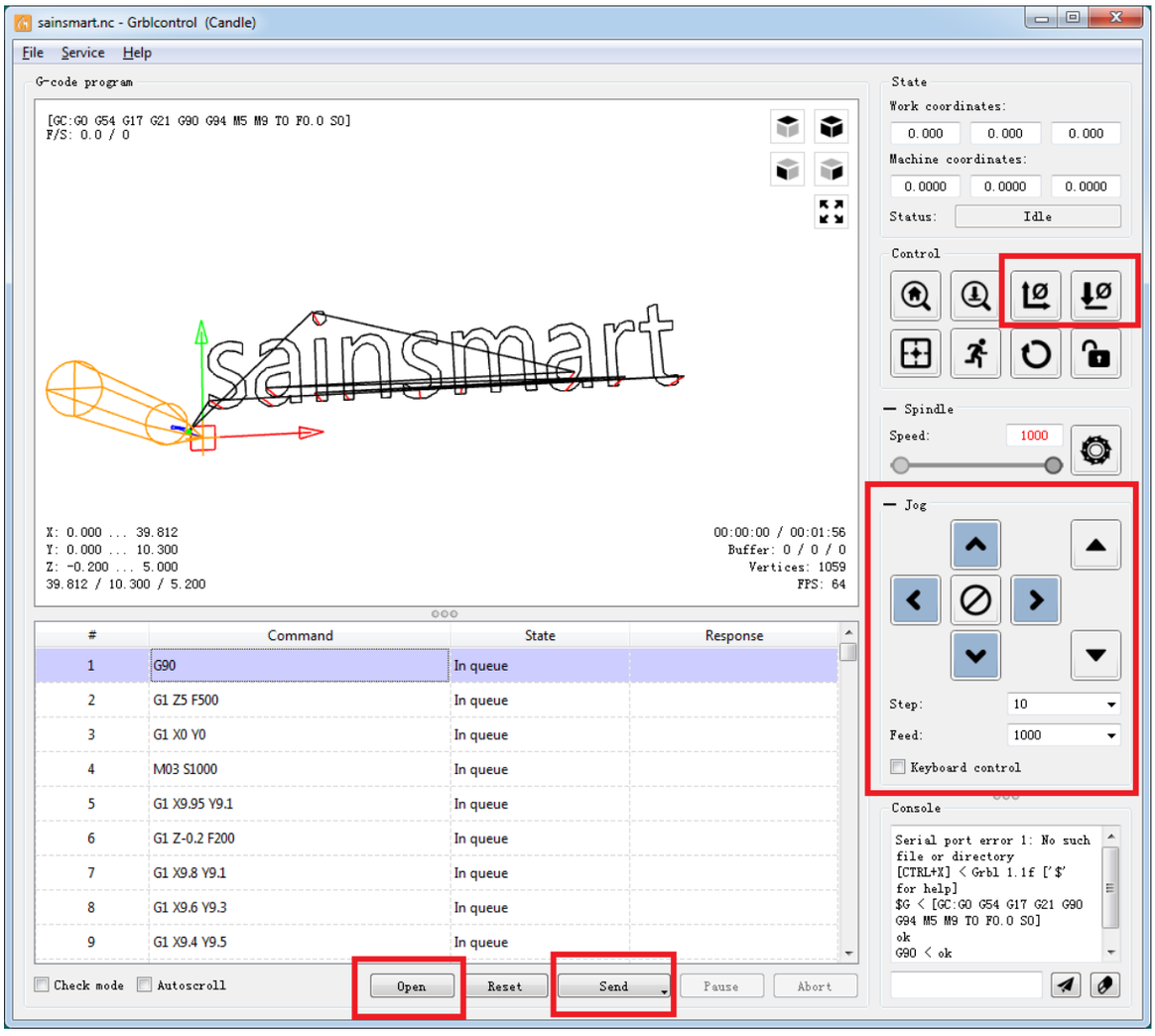


In the main space we see a 3D view of the processing, by holding down the left mouse button we can change the angle. Work coordinates is a display of coordinates, then we have 'Control' - control actions, and 'spindle' - determining the speed of rotation of the cutter. 'Jog' is used for manual positioning, and below are the 'open' and 'send' buttons for opening a file and sending it for processing.

U glavnom prostoru vidimo 3D pregled obrade, držanjem lijeve tipke miša možemo mijenjati kut. Work coordinates je prikaz koordinata, zatim imamo 'Control' – kontrolne radnje, te' spindle' – određivanje brzine vrtnje glodala. 'Jog' -služi za ručno pozicioniranje, a dolje su još 'open' i 'send' tipke za otvaranje datoteke i slanje na obradu.

A fő területen 3D-s áttekintést látunk a feldolgozásról, a bal egérgombot nyomva tartva a szöget tudjuk változtatni. A munkakoordináták a koordináták kijelzése, ezután van a 'Control' - vezérlési műveletek és az "orsó" - a marógép forgási sebességének meghatározása. 'Jog' - a kézi pozicionálásra szolgál, alatta pedig a 'open' és a 'send' gombok is találhatók a fájl megnyitásához és feldolgozásra küldéséhez.

4. Start processing / Pokretanje obrade / A feldolgozás megkezdése



When we open the G-code file (open), we position the tool (cutter) using the marked arrows (Jog), then under (Control) we press the marked icons to save the axis coordinates (XYZ) and finally we press the (send) button.

Kada otvorimo G-kod datoteku (open), pozicioniramo alat (glodalo) pomoću označenih strelica (Jog), zatim pod (Control) pritisnemo označene ikonice za spremanje koordinata osi (XYZ) i na kraju pritisnemo tipku (send).

Amikor megnyitjuk a G-kód fájlt (open), helyezzük el a szerszámot (marót) a megjelölt nyilakkal (Jog), majd a (Control) alatt nyomjuk meg a megjelölt ikonokat a tengelykoordináták mentéséhez (XYZ), végül nyomjuk meg az (send) gombot.

6.2. Laser engraving

6.2. Graviranje laserom

6.2. Gravirozás lézerrel

PRECAUTIONS!

The laser module for a CNC device can cause eye damage and burns. Before using the CNC device, read the manufacturer's instructions and use the device in compliance with all precautionary measures prescribed by the manufacturer. Working with the Laser module, as well as focusing, should definitely be done by teachers!

Always use the supplied goggles when working with the laser module!

To focus on the Laser module, see the instructions at the link:

MJERE OPREZA!

Laser modul za CNC uređaj može izazvati oštećenje oka i opekline. Prije korištenja CNC uređaja pročitajte upute proizvođača, te uređaj koristite poštujući sve mjere opreza koje je propisao proizvođač. Rad sa Laser modulom, kao i fokusiranje svakako bi trebali raditi nastavnici!

Prilikom rada sa laser modulom uvijek koristite priložene zaštitne naočale!

Za fokusiranje Laser modula pogledajte upute na poveznici:

ÓVINTÉZKEDÉSEK!

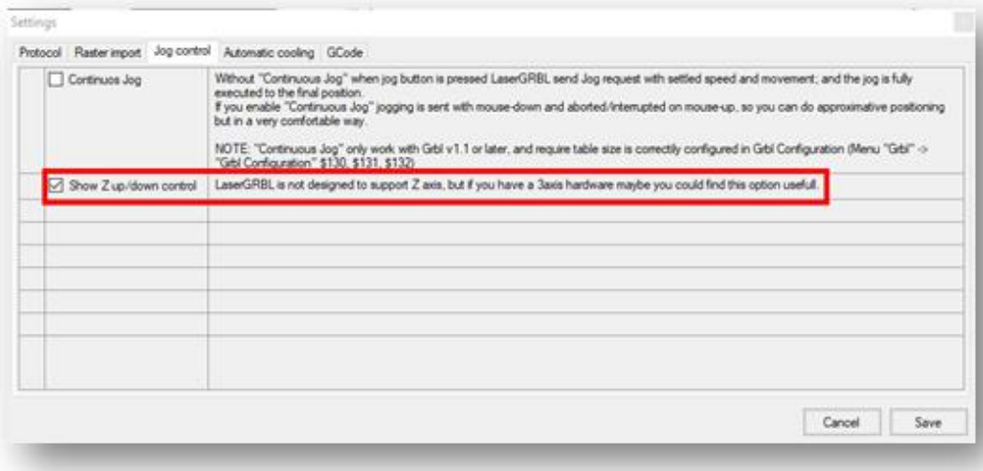
A CNC-eszköz lézermodulja szemkárosodást és égési sérülést okozhat. A CNC eszköz használata előtt olvassa el a gyártó utasításait, és használja a készüléket a gyártó által előírt óvintézkedések betartásával. A Lézer modullal való munkavégzést, valamint a fókuszálást mindenképpen a tanároknak kell elvégezniük!

A lézermodullal végzett munka során mindig használja a mellékelt védőszemüveget!

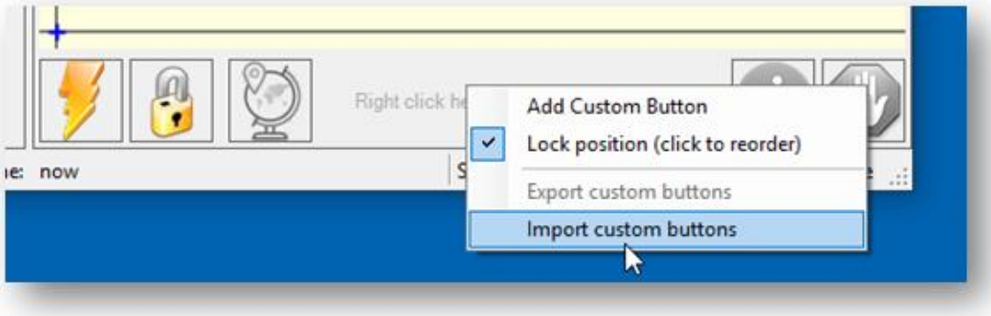
A lézermodul fókuszálásához tekintse meg a linken található utasításokat:

https://docs.sainsmart.com/article/wzvu9798tc-genmitsu-3018-pro-5-5-w-laser-module-user-guide

1. Download and install Laser GRBL / Preuzimanje i instalacija LaserGRBL / Töltse le és telepítse a LaserGRBL-t: https://lasergrbl.com/download/
2. Enabling the Z-axis / Omogućavanje Z-osi / A Z-tengely engedélyezése

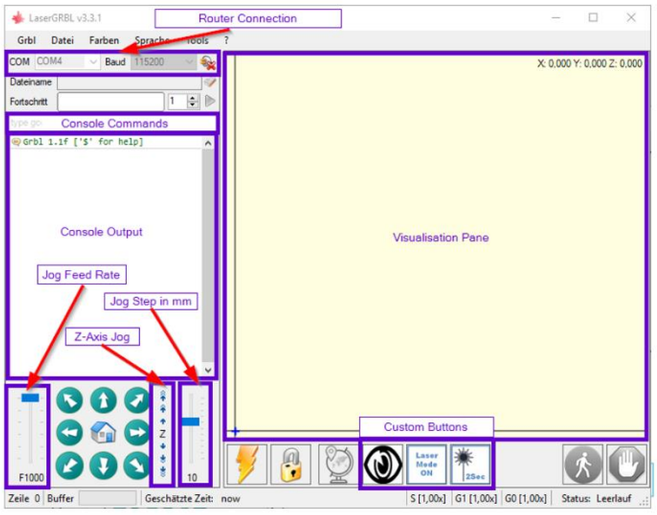


3. Add custom buttons / Dodavanje prilagođenih gumba / Egyéni gombok hozzáadása



Turn on laser at low power(S100), press again to turn off.

Set for Laser mode ($32=1)



Turn on the laser at low power(S100) for 2 seconds,
then turn it off again (useful when setting an origin position)

In the ‘Open’ window, select the downloaded file and click the ‘Open’ button. Now a short dialog is displayed for each of the three additional buttons. You can now select for each individual button contained in the archive file whether it should be imported or not. Select ‘Yes’ for each button.

U prozoru 'Otvori' odaberite preuzetu datoteku i kliknite gumb 'Otvori'. Sada se prikazuje kratki dijaloški okvir za svaki od tri dodatna gumba. Sada za svaki pojedinačni gumb koji se nalazi u arhivskoj datoteci možete odabrati hoće li se uvesti ili ne. Odaberite "Da" za svaki gumb.

A „Megnyitás” ablakban válassza ki a letöltött fájlt, és kattintson a „Megnyitás” gombra. Most egy rövid párbeszédpanel jelenik meg mind a három további gombhoz. Most az archív fájlban található minden egyes gombnál kiválaszthatja, hogy importálja-e vagy sem. Válassza az „Igen” lehetőséget minden gombhoz.

7. 3D modeling and printing
7.3D modeliranje i ispis
7.3D modellezés és nyomtatás

The process of creating 3D objects using the additive manufacturing method consists of 3 basic phases:

1. 3D modeling
2. preparation of a 3D model for printing
3. 3D printing

Proces izrade 3D predmeta metodom aditivne proizvodnje sastoji se od 3 osnovne faze:

1. 3D modeliranja
2. pripreme 3D modela za ispis
3. 3D ispisa

A 3D tárgyak additív gyártási módszerrel történő létrehozásának folyamata 3 alapvető fázisból áll:

1. 3D modellezés
2. a 3D modell nyomtatásának előkészületei
3. 3D nyomtatás

7.1. 3D modeling
7.1. 3D modeliranje
7.1. 3D modellezés és nyomtatás

Nowadays, there are several different software tools available on the market for 3D modeling, i.e. creating a computer 3D model. This technology is called Computer-Aided Design (CAD) so, in accordance with that, 3D models are often called CAD models. Among the well-known CAD software tools, we should mention AutoCAD, SolidWorks, Catia, etc. The best-known software tools for this purpose for beginners are Tinkercad, SketchUp and similar.

U današnje vrijeme na tržištu je dostupno više različitih programskih alata za 3D modeliranje, odnosno izradu računalnog 3D modela. Ova tehnologija naziva se računalno potpomognuti dizajn (eng. CAD, Computer-Aided Design), pa se u skladu s time izrađeni 3D modeli često nazivaju CAD modelima. Od poznatijih CAD programskih alata valja spomenuti AutoCAD, SolidWorks, Catia, itd. Najpoznatiji programski alati ovakve namjene za početnike su Tinkercad, SketchUp i slični.

Napjainkban többféle szoftvereszköz is elérhető a piacon a 3D modellezéshez, azaz számítógépes 3D modell készítéséhez. Ezt a technológiát számítógéppel segített tervezésnek (CAD, Computer-Aided Design) nevezik, így az ennek megfelelően készített 3D modelleket gyakran CAD modelleknek is nevezik. Az ismertebb CAD-szoftverek közül meg kell említeni az AutoCAD-et, a SolidWorks-t, a Catiát stb. A legismertebb kezdő szoftvereszközök erre a célra a Tinkercad, a SketchUp és hasonlók.

7.1.1. Creation of 3D model in Tinkercad

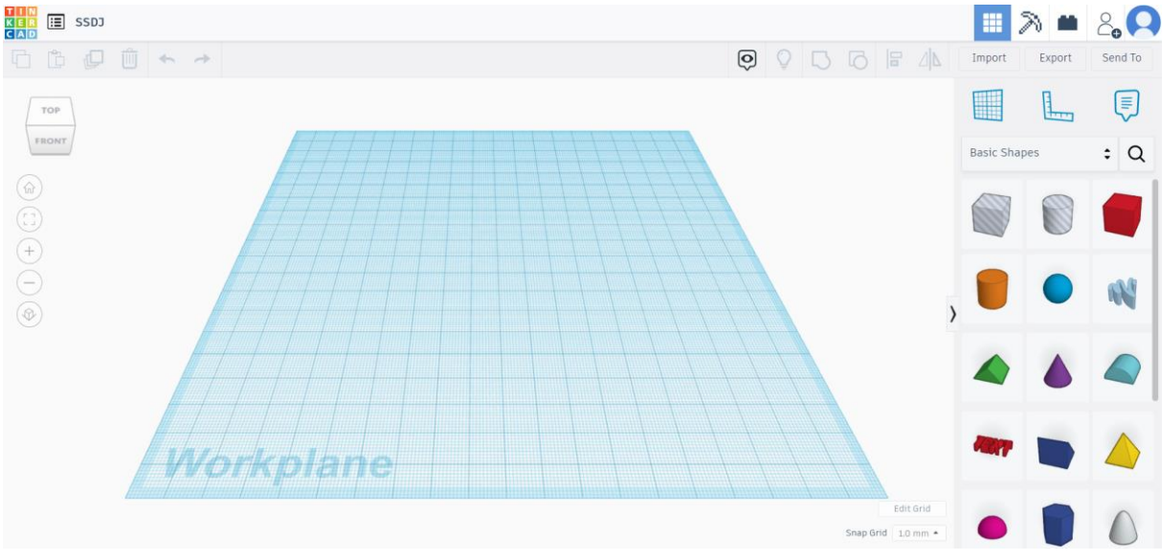
7.1.1. Izrada 3D modela u Tinkercadu

7.1.1. 3D modell elkészítése Tinkercad segítségével

Tinkercad is an online programming tool available at: https://www.tinkercad.com. The basic interface is shown in the following image:

Tinkercad je online programski alat dostupan na adresi: https://www.tinkercad.com. Osnovno sučelje prikazano je sljedećom slikom:

A Tinkercad egy online programozási eszköz, amely a következő címen érhető el: https://www.tinkercad.com. Az alap munkaterület a következő képen látható:



Control of the display of the workspace is possible with the following combinations of mouse and keys:

- zoom – turning the mouse wheel
- moving the workspace – pressing the mouse wheel and dragging the mouse in the desired direction
- rotating the workspace – right mouse button or **Ctrl** + left mouse button

Upravljanje prikazom radnog prostora moguće je sljedećim kombinacijama miša i tipki:

- zumiranje – okretanje kotačića miša
- pomicanje radnog prostora – pritisak na kotačić miša i povlačenje miša u željenom smjeru
- zakretanje radnog prostora – desna tipka miša ili **Ctrl** + lijeva tipka miša

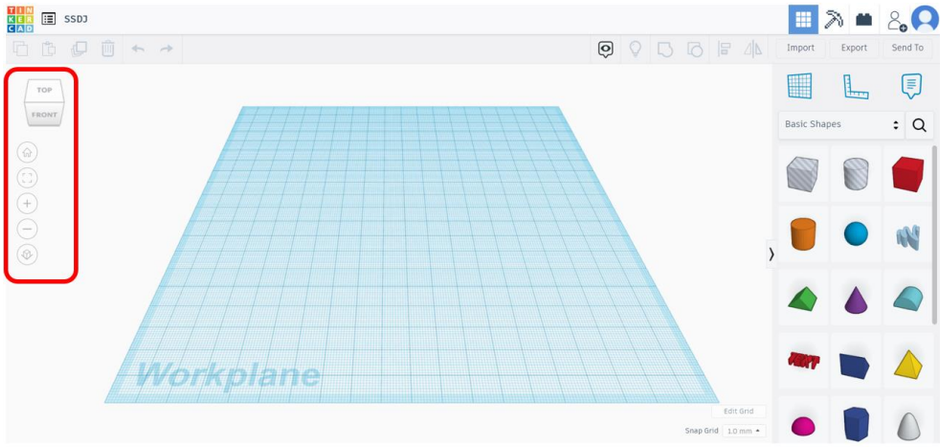
A munkaterület megjelenítésének kezelése az egér és a billentyűk alábbi kombinációival lehetséges:

- nagyítás – az egér görgőjének elforgatása
- a munkaterület mozgatása – az egér görgőjének lenyomása és az egér húzása a kívánt irányba
- a munkaterület elforgatása – jobb egérgomb vagy Ctrl + bal egérgomb

All the listed options for managing the display of the workspace can be selected through the functions available on the left side of the workspace.

Sve navedene opcije upravljanja prikazom radnog prostora moguće je odabrati putem funkcija dostupnih s lijeve strane radnog prostora.

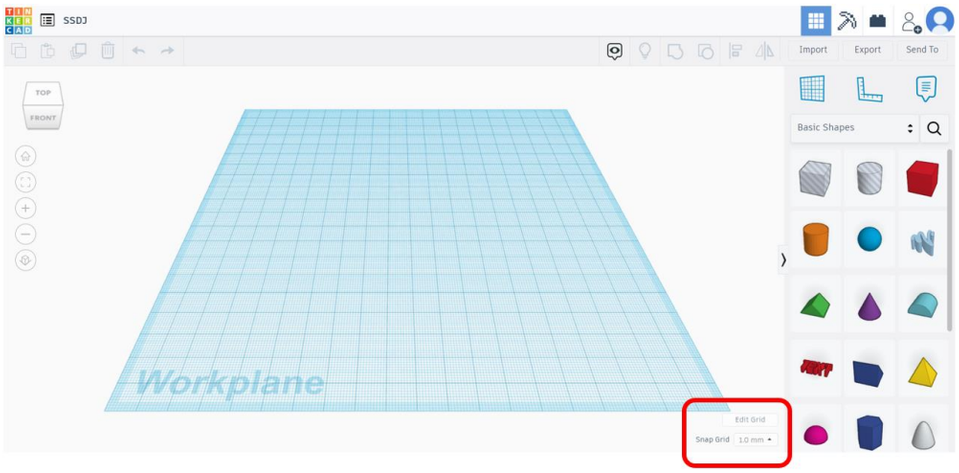
A felsorolt munkaterület-kezelési lehetőségek mindegyike kiválasztható a munkaterület bal oldalán elérhető funkciókkal.

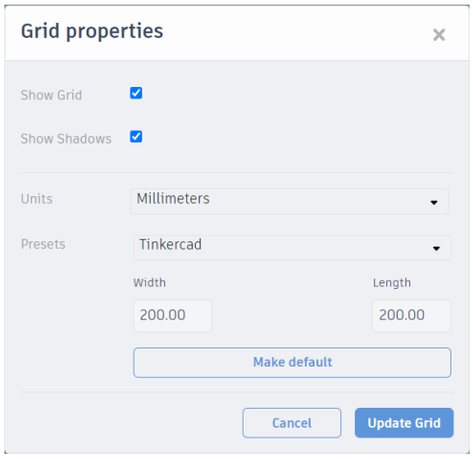


Before starting with creation of the 3D model, it is recommended to adjust the grid of the workspace.

Prije početka izrade 3D modela poželjno je podesiti mrežu radnog prostora.

Még a 3D modell létrehozásának kezdetén kívánatos a munkaterület rácsának beállítása.

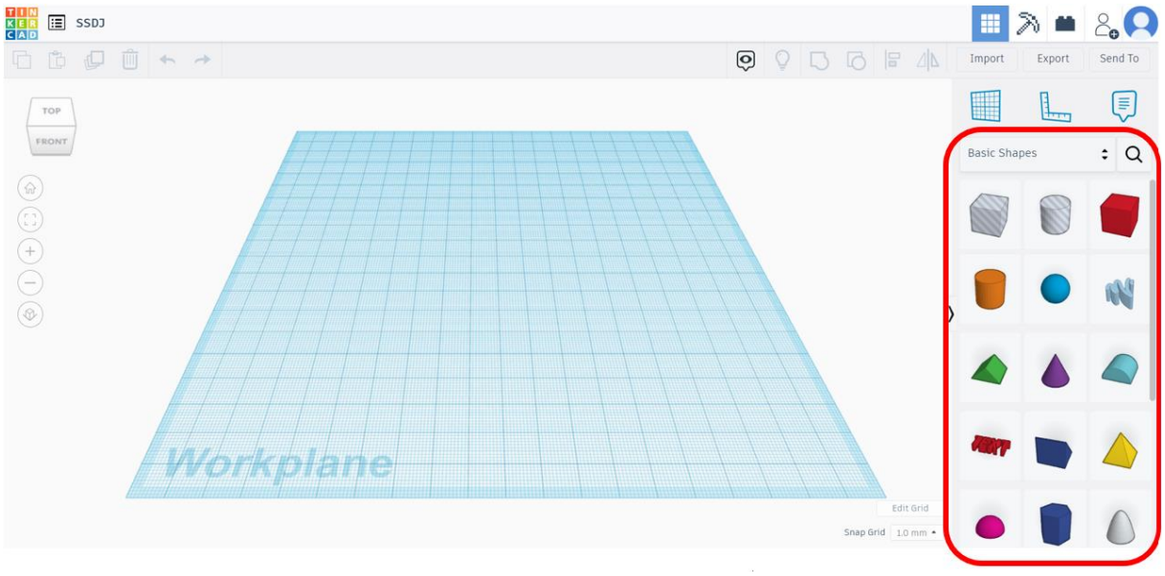




Each 3D model can be built by combining several basic or complex geometric shapes. Adding the available geometric shapes is possible from the menu on the right.

Svaki 3D model moguće je izgraditi kombinacijom više osnovnih ili složenih geometrijskih oblika. Dodavanje već pripremljenih geometrijskih oblika dostupno je iz izbornika s desne strane.

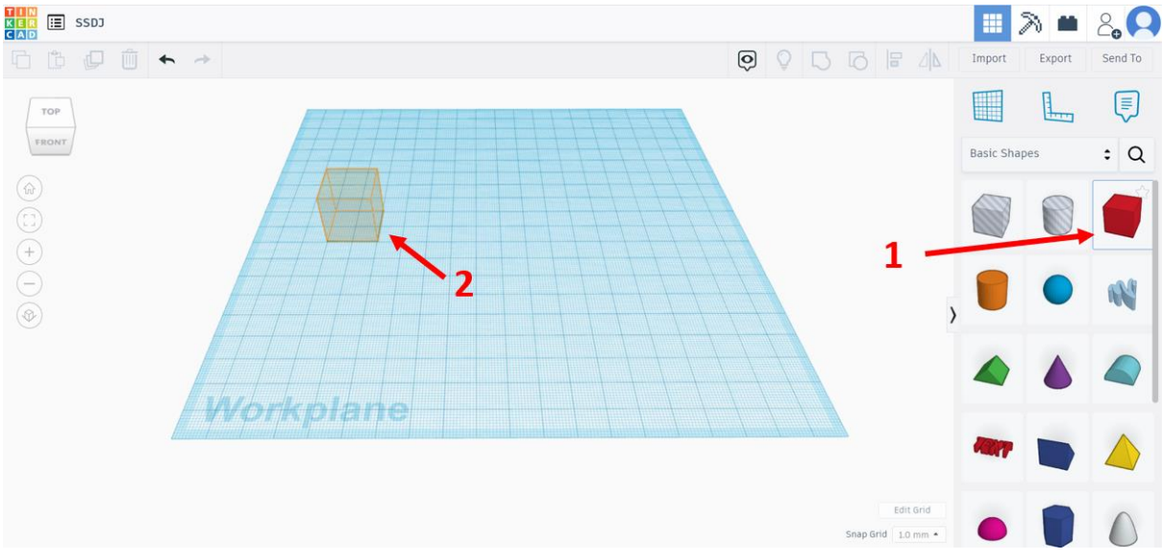
Minden 3D modell több alapvető vagy összetett geometriai alakzat kombinálásával is megépíthető. A jobb oldali menüből elérhető a már elkészített geometriai formák hozzáadása.



We start adding a geometric shape to the workspace by clicking on the desired shape and then placing it in the desired place in the workspace.

Dodavanje geometrijskog oblika u radni prostor započinjemo klikom na željeni oblik i zatim postavljanjem u radni prostor na željeno mjesto.

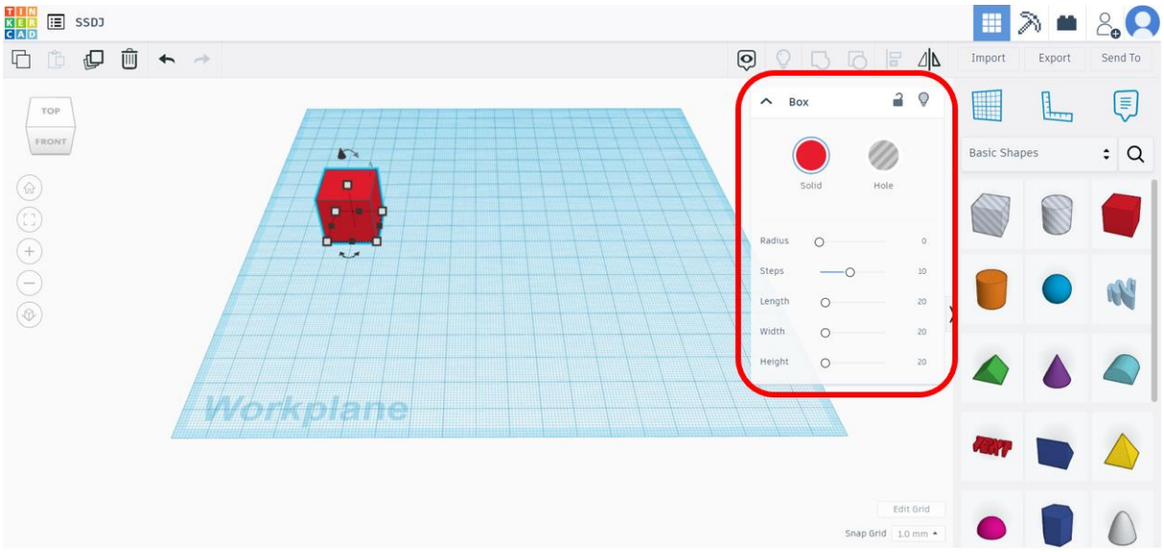
Egy geometriai alakzat hozzáadását a munkaterülethez úgy kezdjük, hogy rákattintunk a kívánt alakzatra, majd elhelyezzük a munkaterület kívánt helyére.



After placing the geometric shape in the workspace, a window for editing basic parameters (dimensions, properties, etc.) opens. The same window can be displayed by subsequent clicking on any geometric shape in the workspace.

Nakon postavljanja geometrijskog oblika u radni prostor otvara se prozor za uređivanje osnovnih parametara (dimenzije, svojstva, itd.). Isti prozor moguće je prikazati i naknadnim klikom na bilo koji geometrijski oblik koji se nalazi u radnom prostoru.

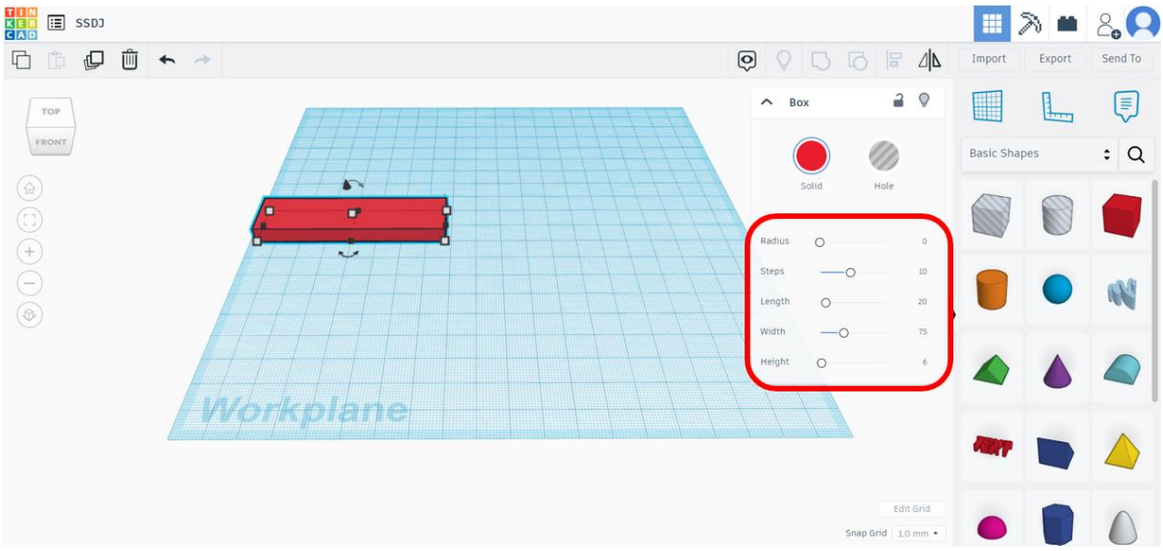
A geometriai alakzat munkaterületen történő elhelyezése után megnyílik egy ablak az alapvető paraméterek (méretek, tulajdonságok stb.) szerkesztésére. Ugyanez az ablak megjeleníthető a munkaterület bármely geometriai alakzatára történő utólagos kattintással.



Determine the dimensions of the geometric shape by moving the slider or by keyboard input.

Pomicanjem klizača ili slobodnim unosom odredimo dimenzije geometrijskog oblika.

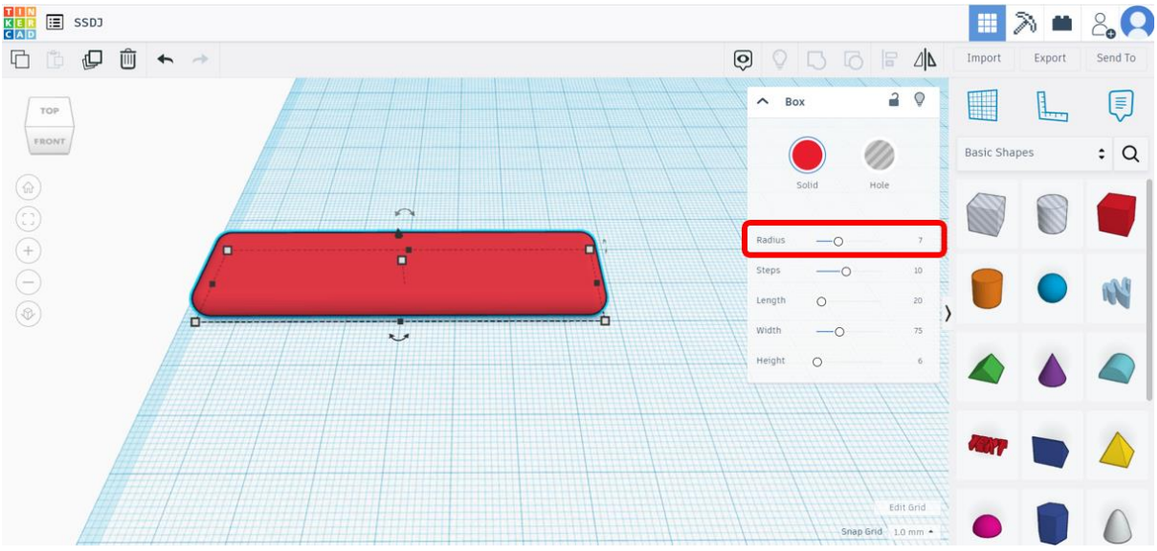
A csúszka mozgatásával vagy szabad bevitellel határozzuk meg a geometriai alakzat méreteit.



By changing the ***Radius*** parameter we achieve smoothing of the edges of the geometric shape.

Promjenom parametra ***Radius*** postižemo zaglađivanje bridova geometrijskog oblika.

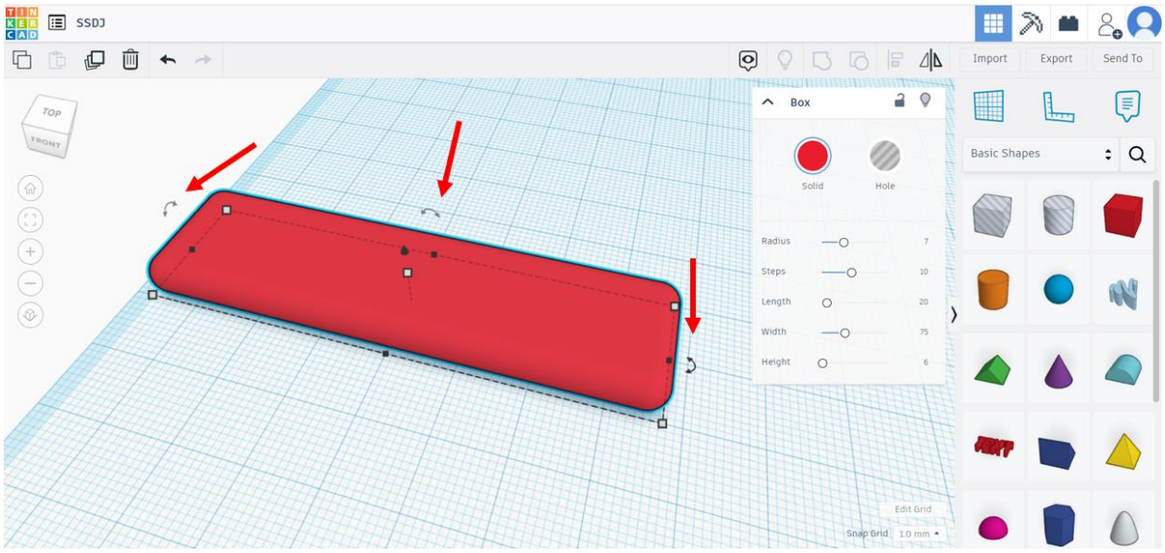
A **Radius** paraméter változtatásával a geometriai alakzat éleinek simítását érjük el.



We can move any geometric shape in the workspace using the mouse dragging method. We can also rotate each geometric shape around the main axes by clicking on one of the three double-sided arrows.

Bilo koji geometrijski oblik možemo pomicati u radnom prostoru koristeći metodu povlačenja mišem. Također, svaki geometrijski oblik možemo i rotirati oko glavnih osi klikom na jednu od tri dvostrane strelice.

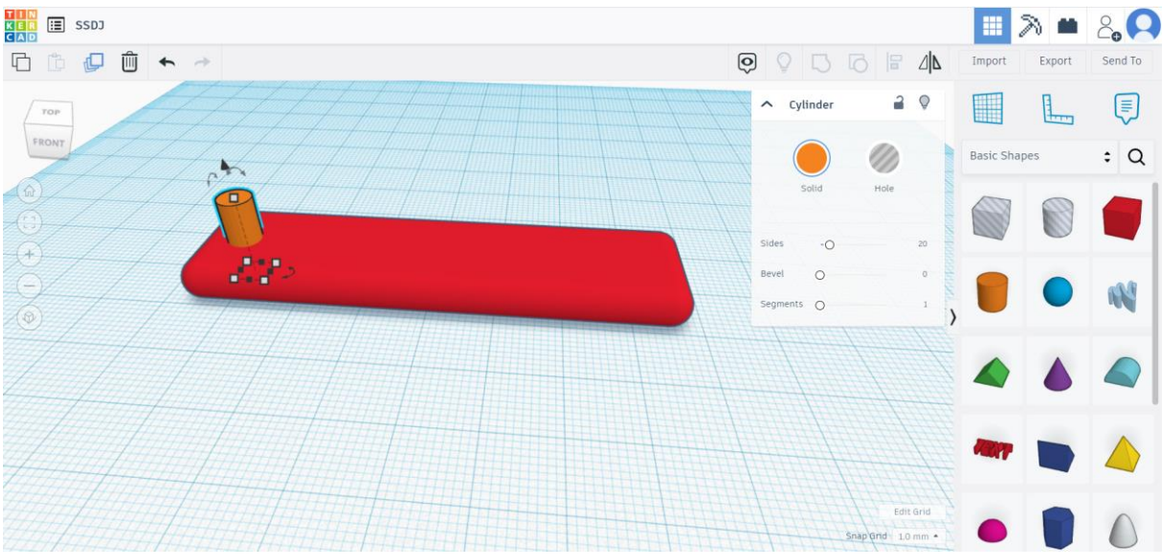
Az egérhúzás módszerével tetszőleges geometriai alakzatot mozgathatunk a munkaterületen. Az egyes geometriai alakzatokat a főtengelyek körül is elforgathatjuk, ha a három kétoldalas nyíl valamelyikére kattintunk.



Since we usually build a 3D model by combining several different geometric shapes, we add the next shape to the workspace and place it in the desired position in relation to the previously added geometric shapes.

Budući da 3D model uobičajeno gradimo kombinacijom više različitih geometrijskih oblika, u radni prostor dodajemo sljedeći oblik i postavljamo ga na željenu poziciju u odnosu na već dodane geometrijske oblike.

Mivel általában több különböző geometriai alakzat kombinálásával készítünk 3D-s modellt, a következő alakzatot hozzáadjuk a munkaterülethez, és a már hozzáadott geometriai alakzatokhoz képest a kívánt pozícióba helyezzük.



Most of the available geometric shapes are initially set to the ***Solid*** property, which is a shape with real (solid) filling in space. Each such shape represents the addition of a certain volume to the workspace.

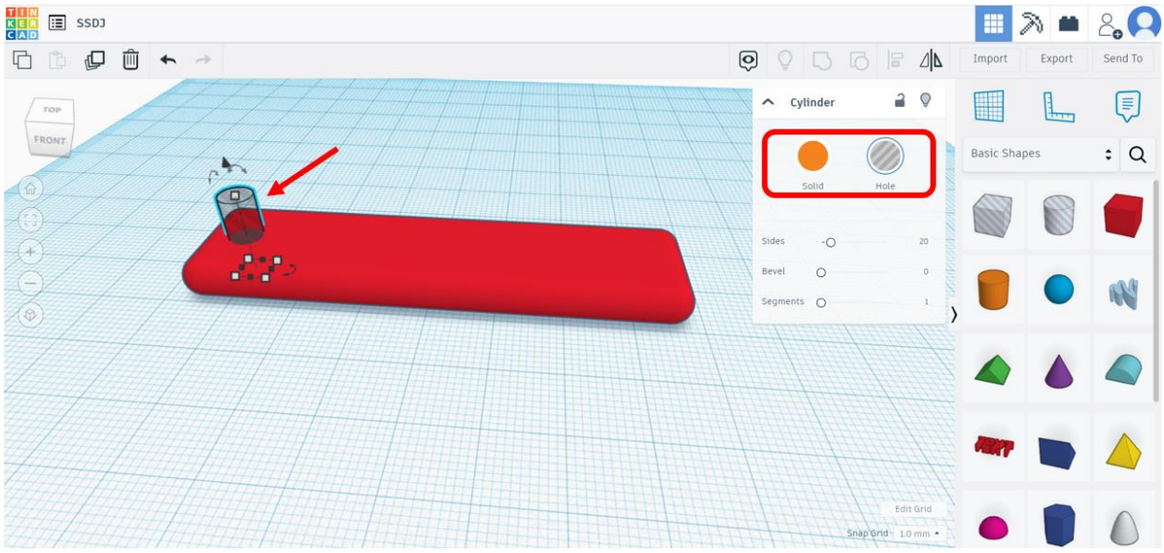
Alternatively, it is possible to select the ***Hole*** property, which allows us to cut out a certain geometric shape from the existing ones. In other words, it is about subtracting a certain volume from the existing workspace. After selecting this property, the geometric shape becomes transparent in the workspace.

Većini dostupnih geometrijskih oblika inicijalno je zadano svojstvo ***Solid***, što je oblik sa stvarnom ispunom u prostoru. Svaki takav oblik predstavlja dodavanje određenog volumena u radni prostor.

Alternativno je moguće odabrati i svojstvo ***Hole***, čime postižemo izrezivanje određenog geometrijskog oblika iz postojećih. Drugim riječima, radi se oduzimanju određenog volumena iz postojećeg radnog prostora. Nakon odabira ovog svojstva geometrijski oblik postaje proziran u radnom prostoru.

A legtöbb elérhető geometriai alakzat kezdetben a ***Solid*** tulajdonságra van beállítva, ami egy valós térkitöltésű alakzat. Minden ilyen alakzat egy bizonyos térfogat hozzáadását jelenti a munkaterülethez.

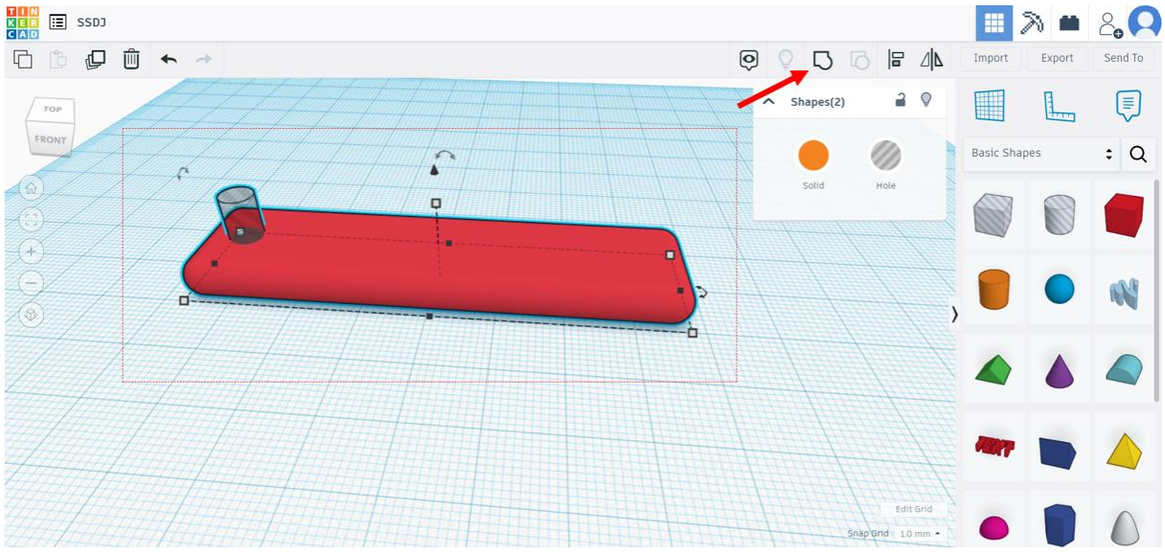
Alternatív megoldásként lehetőség van a ***Hole*** tulajdonság kiválasztására, amellyel a meglévőkből kivághatunk egy bizonyos geometriai formát. Más szóval arról van szó, hogy ki kell vonni egy bizonyos térfogatot a meglévő munkaterületből. A tulajdonság kiválasztása után a geometriai alakzat átlátszóvá válik a munkaterületen.



The final effect of cutting will be visible after grouping multiple geometric shapes into one shape. In order to achieve this, it is necessary to select individually the shapes that we want to group (***Shift*** key + left mouse button) or to select all shapes in the workspace (by clicking on the left mouse button and dragging the selection box around all shapes). After that, we need to choose the Group option from the top menu.

Konačni efekt izrezivanja bit će vidljiv nakon grupiranja više geometrijskih oblika u jedan oblik. Kako bi se to postiglo potrebno je pojedinačno odabrati oblike koje želimo grupirati (tipka **Shift** + lijeva tipka miša) ili odabrati sve oblike u radnom prostoru (klikom na lijevu tipku miša i povlačenjem selekcijskog okvira oko svih oblika). Nakon toga potrebno je opciju ***Group*** iz gornjeg izbornika.

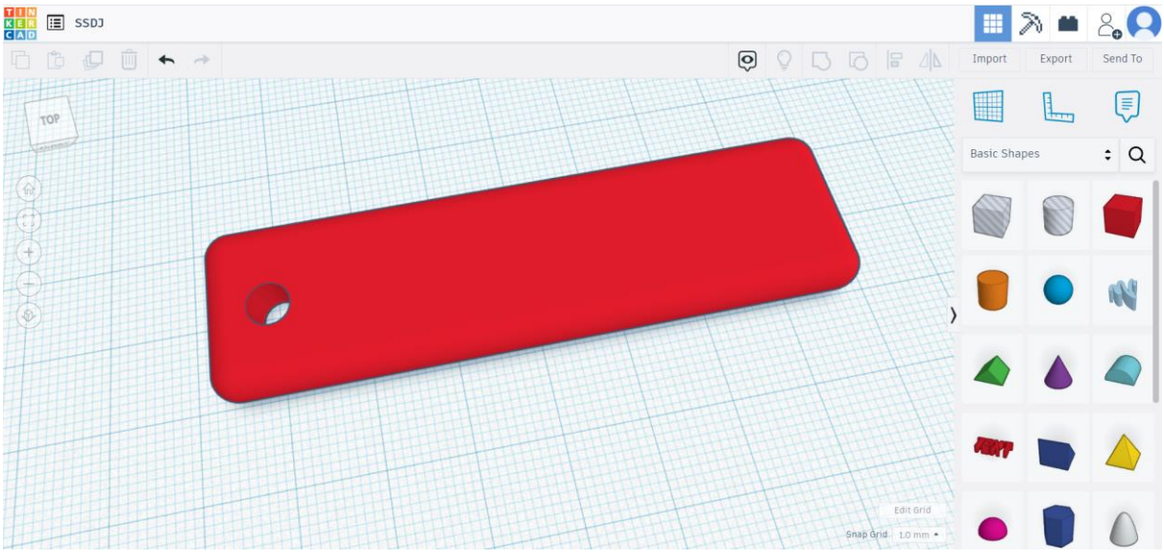
A végső vágási hatás több geometriai alakzat egyetlen alakzatba csoportosítása után lesz látható. Ennek eléréséhez külön-külön kell kiválasztani a csoportosítani kívánt alakzatokat (**Shift** billentyű + bal egérgomb), vagy ki kell jelölni az összes alakzatot a munkaterületen (bal egérgombbal kattintva és a kijelölődobozt az összes alakzat köré húzva). Ezt követően szükség van a ***Group*** opcióra a felső menüben.



The result of grouping of the ***Solid*** and ***Hole*** shapes in this example is the volume of the ***Hole*** shape cut out from the volume of the ***Solid*** shape.

Rezultat grupiranja ***Solid*** i ***Hole*** oblika u ovom primjeru je izrezan volumen ***Hole*** oblika iz volumena ***Solid*** oblika.

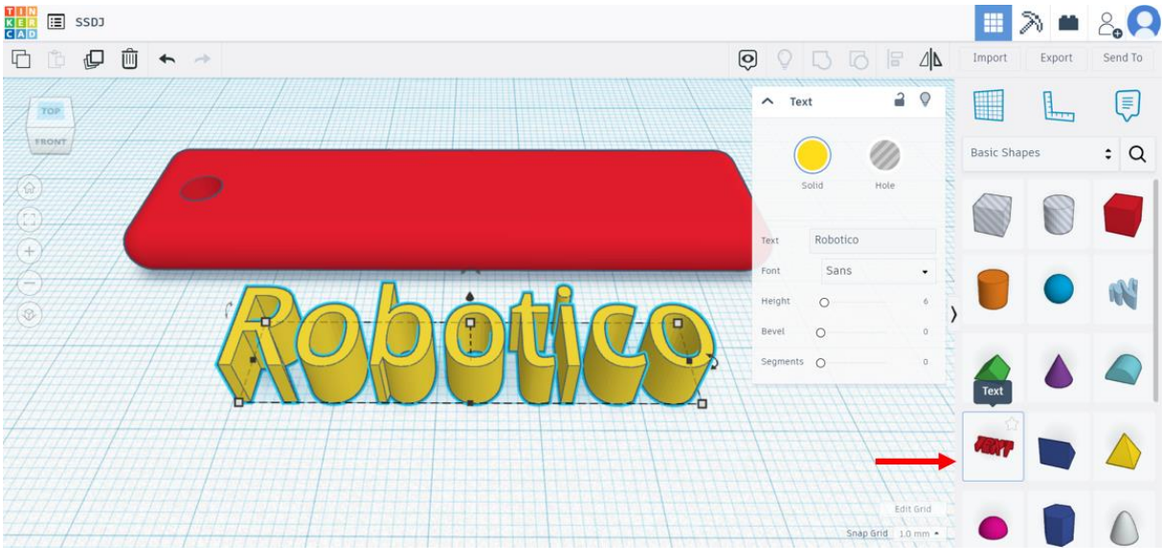
Ebben a példában a ***Solid*** és a ***Hole*** alakzatok csoportosításának eredménye a ***Hole*** alak térfogatából kivágott ***Solid*** alakzat térfogata.



As one of the shapes, it is possible to add a 3D text, which we can edit the text content and dimensions of.

Kao jedan od oblika moguće je dodati i 3D tekst, kojem zatim možemo urediti tekstualni sadržaj i dimenzije.

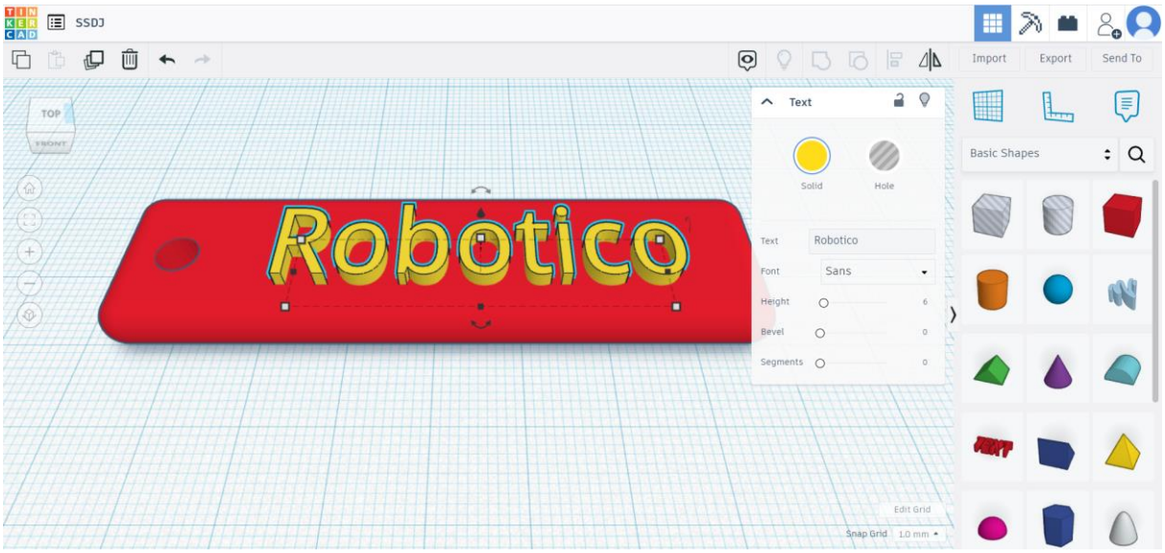
Lehetőség van egy 3D-s szöveg hozzáadására az egyik alakzatként, amelyhez ezután szerkeszthetjük a szöveg tartalmát és méreteit.



If we place the text over previously created geometric shapes, it is possible to achieve the effect of highlighting the text on the surface of a certain shape.

Ako tekst postavimo preko prethodno kreiranih geometrijskih oblika, moguće je postići efekt isticanja teksta na površini određenog oblika.

Ha a szöveget korábban létrehozott geometriai formák fölé helyezzük, akkor egy adott forma felületén elérhetjük azt a hatást, hogy a szöveget kiemeljük.



If we set the ***Hole*** property to the text, we will get the effect of carved letters in the material.

Ako tekstu postavimo svojstvo ***Hole*** dobit ćemo efekt urezanih slova u materijalu.

Ha a ***Hole*** tulajdonságot beállítjuk a szövegre, akkor az anyagba vésett betűk hatását kapjuk.

7.1.2. Export of the 3D model to a appropriate format

7.1.2. Izvoz 3D modela u prikladan format

7.1.2. A 3D modell exportálása megfelelő formátumba

In order for the created 3D model to be suitable for printing on a 3D printer, it must be exported to a file of the appropriate format. In Tinkercad, this option is available by clicking the ***Export*** button.

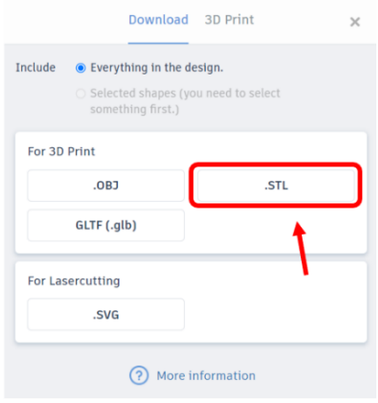
Kako bi kreirani 3D model bio pogodan za ispis na 3D printeru potrebno ga je izvesti u datoteku odgovarajućeg formata. U programskom alatu Tinkercad ta opcija dostupna je klikom na gumb ***Export***.

Ahhoz, hogy az elkészített 3D modell alkalmas legyen 3D nyomtatón történő nyomtatásra, azt megfelelő formátumú fájlba kell exportálni. A Tinkercadben ez a lehetőség az ***Exportálás*** gombra kattintva érhető el.

One of the standard formats recognized by almost every 3D printing software is the STL format.

Jedan od standardnih formata koje prepoznaje gotovo svaki programski alat za 3D ispis je STL format.

Az egyik szabványos formátum, amelyet szinte minden 3D nyomtatáshoz használt szoftver felismer, az STL formátum.



After choosing the format, it is necessary to specify the file name and the location of saving on the computer. The created 3D model is ready for further processing and printing on a 3D printer.

Nakon odabira formata potrebno je zadati naziv datoteke i lokaciju spremanja na računalu. Kreirani 3D model spreman je za daljnju obradu i ispis na 3D printeru.

A formátum kiválasztása után meg kell adni a fájl nevét és a mentés helyét a számítógépen. Az elkészített 3D modell készen áll a további feldolgozásra és 3D nyomtatón történő nyomtatásra.

7.2. Preparation of a 3D model for printing

7.2. Priprema 3D modela za ispis

7.2. A 3D modell nyomtatásának előkészületei

As 3D printers from different manufacturers are usually controlled by different software tools, the created 3D model must be prepared for printing using a software dedicated for working with a specific model of 3D printer. This example shows the basic preparation of a 3D model in the Prusa Slicer software. More detailed instructions for using the software are available at the following link: https://www.prusa3d.com/page/prusaslicer_424.

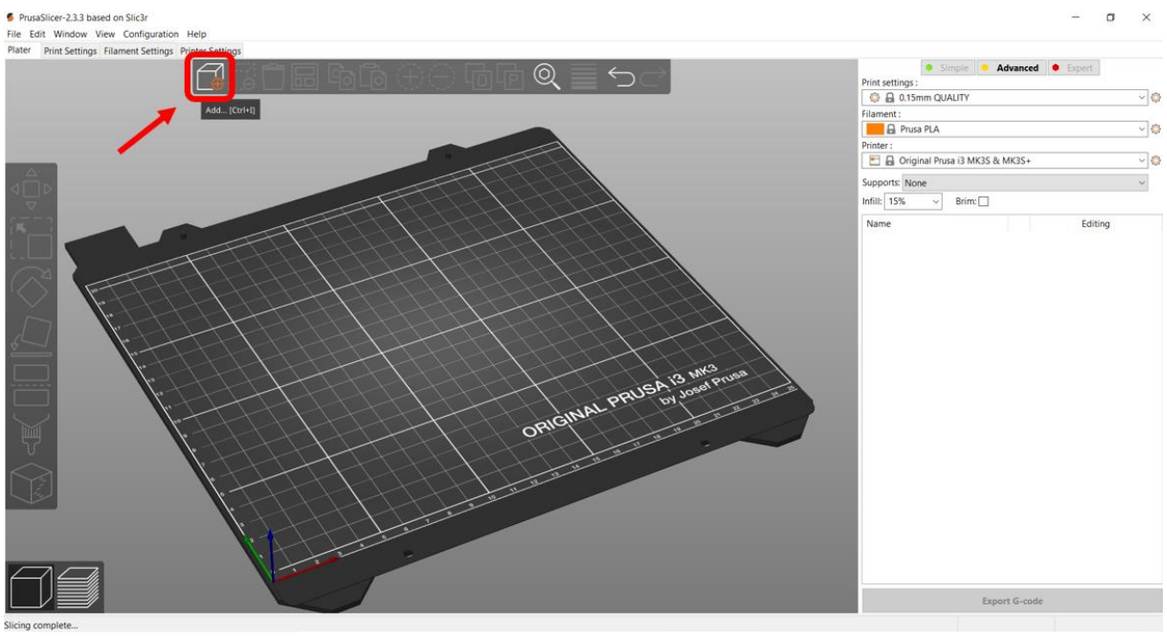
Kako su 3D printeri različitih proizvođača uglavnom upravljani različitim programskim alatima, izrađeni 3D model potrebno je pripremiti za ispis koristeći programski alat namijenjen radu s određenim modelom 3D printera. U ovom primjeru prikazana je osnovna priprema 3D modela u programskom alatu Prusa Slicer. Detaljnije upute za korištenje programskog alata dostupne su na sljedećoj poveznici: https://www.prusa3d.com/page/prusaslicer_424.

Mivel a különböző gyártók 3D nyomtatóit többnyire különböző szoftvereszközök vezérlik, az elkészített 3D modellt egy adott 3D nyomtatómodellhez tervezett szoftvereszközzel kell nyomtatásra előkészíteni. Ez a példa egy 3D-s modell alapvető előkészítését mutatja be a Prusa Slicer szoftvereszközben. A szoftvereszköz használatára vonatkozó részletesebb utasítások az alábbi linken érhetők el: https://www.prusa3d.com/page/prusaslicer_424.

The created 3D model is added to the working environment by selecting the ***Add…*** option.

Izrađeni 3D model dodaje se u radno okruženje odabirom opcije ***Add…***

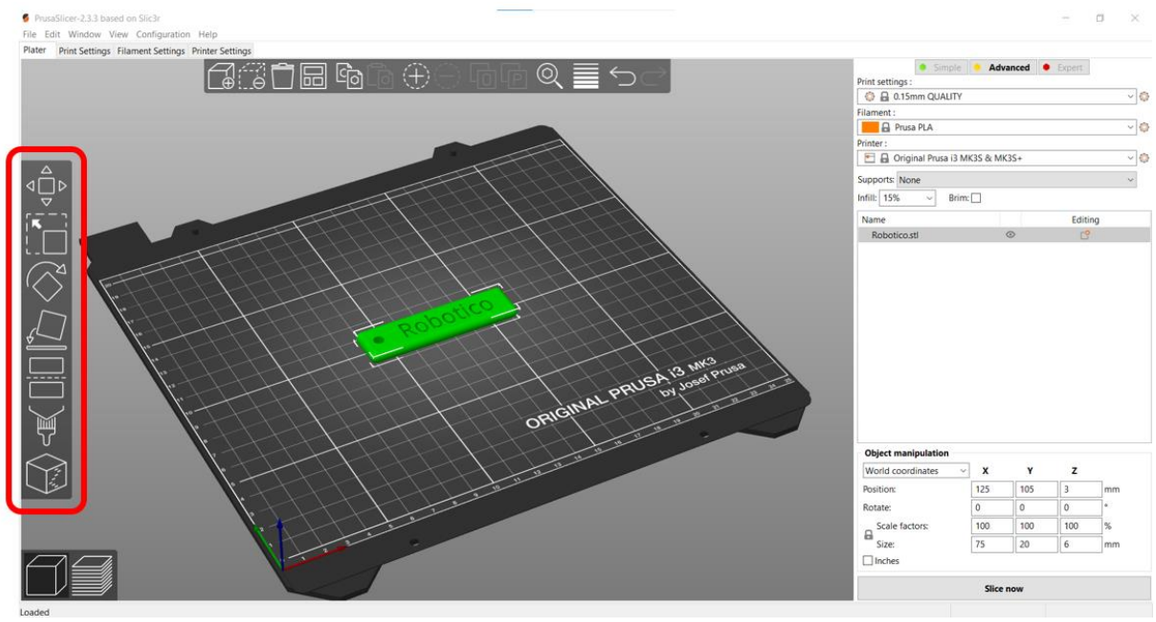
A létrehozott 3D-s modell hozzáadódik a munkakörnyezethez a ***Hozzáadás...*** opció kiválasztásával.



The added 3D model needs to be placed on the printing surface in an optimal way, which implies the orientation of the 3D model so as to ensure good adhesion to the surface and to reduce the need for support structures as much as possible during printing. The 3D model can be moved or rotated on the surface and its size can be changed while keeping the proportions. The above options are available from the menu on the left.

Dodani 3D model potrebno je smjestiti na podlogu za ispis na optimalan način, što podrazumijeva orijentaciju 3D modela tako da se osigura dobro prianjanje za podlogu te da se prilikom ispisa smanji potreba za potpornim strukturama koliko god je moguće. 3D model je moguće pomicati ili rotirati po podlozi te mu mijenjati veličinu uz zadržavanje proporcija. Navedene opcije dostupne su iz izbornika s lijeve strane.

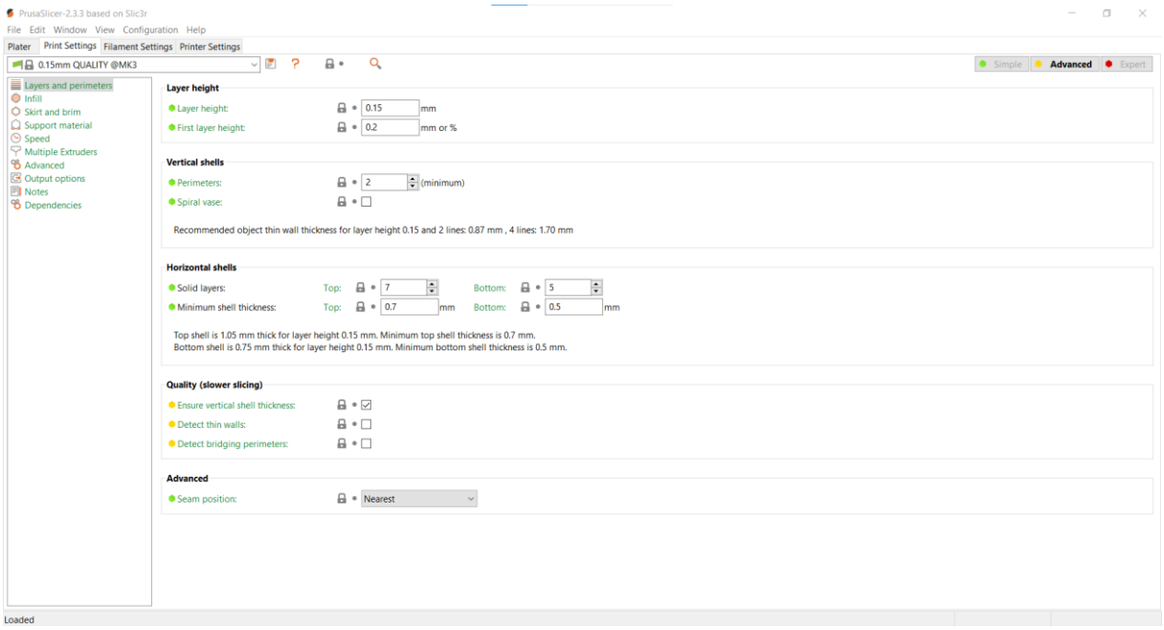
A hozzáadott 3D-s modellt a nyomtatáshoz optimális módon kell a hordozóra helyezni, ami azt jelenti, hogy a 3D-s modellt úgy kell elhelyezni, hogy a hordozóhoz való jó tapadás biztosítva legyen, és a lehető legnagyobb mértékben csökkentse a tartószerkezetek hozzáadásáank szükségességét a nyomtatás során. A 3D modell a munkafelületen mozgatható vagy forgatható, és az arányok megtartása mellett átméretezhető. A fenti lehetőségek a bal oldali menüből érhetők el.



To prepare the 3D model for printing, the most important are the print settings that we adjust in the ***Print Settings*** window. The most important settings include choosing the thickness of the printing layer, the density and shape of the final 3D object infill, the type of support structures, etc.

Za pripremu 3D modela za ispis najvažnije su postavke ispisa koje podešavamo u prozoru ***Print Settings***. Najvažnije postavke uključuju odabir debljine sloja ispisa, gustoću i oblik ispune konačnog 3D objekta, vrste potpornih struktura, itd.

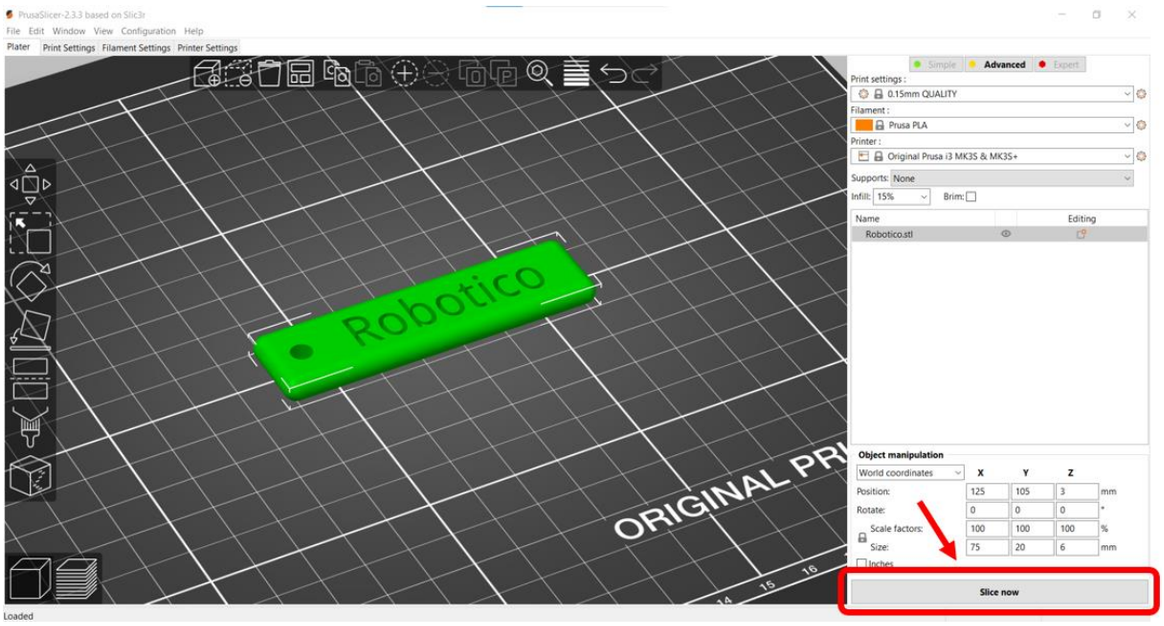
A 3D modell nyomtatásra való előkészítéséhez a legfontosabbak a nyomtatási beállítások, amelyeket a ***Print Settings*** (Nyomtatási beállítások) ablakban állítunk be. A legfontosabb beállítások közé tartozik a nyomtatási réteg vastagságának megválasztása, a végső 3D objektum kitöltés sűrűsége és alakja, a tartószerkezetek típusa, stb.



After the print settings have been adjusted, the software cuts the 3D model into layers. We select the ***Slice now*** option.

Nakon podešenih postavki ispisa slijedi softversko izrezivanje 3D modela na slojeve. Odabiremo opciju ***Slice now***.

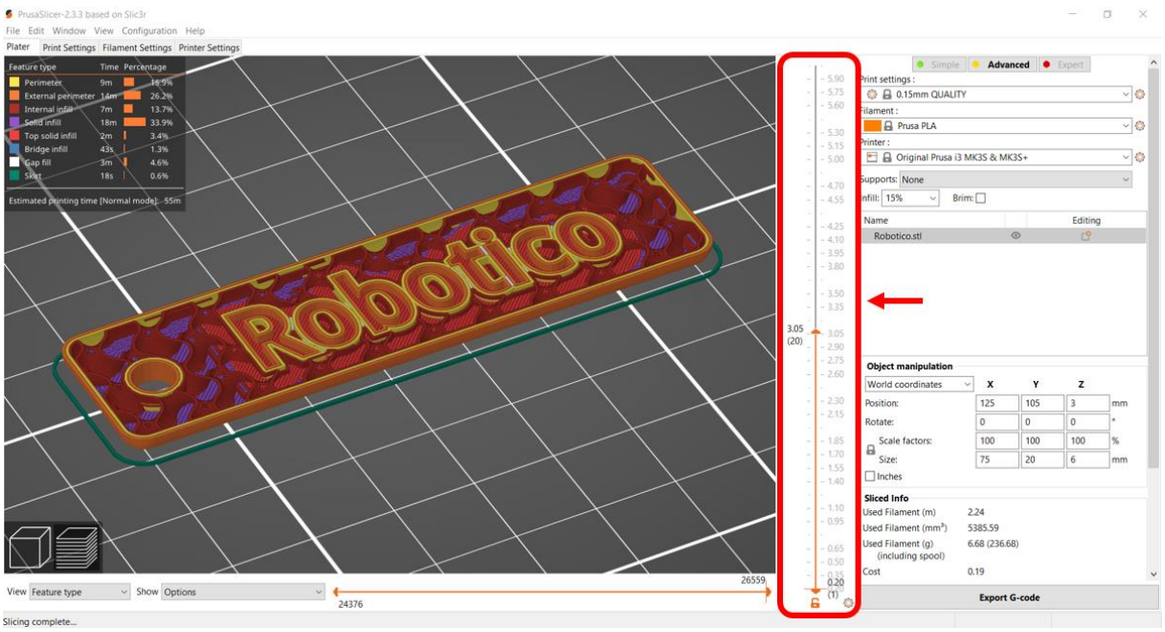
A nyomtatási beállítások módosítása után a szoftver rétegekre vágja a 3D modellt. Válasszuk ki a ***Slice now*** (Szelet most) opciót.



After cutting into layers, by moving the vertical slider it is possible to display the appearance of each layer for printing on a 3D printer.

Nakon izrezivanja na slojeve, pomicanjem vertikalnog klizača moguće je prikazati izgled svakog pojedinog sloja za ispis na 3D printeru.

A rétegekre vágás után a függőleges csúszka mozgatásával lehetőség nyílik az egyes rétegek megjelenésének megjelenítésére 3D nyomtatón történő nyomtatáshoz.



If we are satisfied with the quality of the print settings, we select the ***Export G-code*** option to create an executable file with the control code, which needs to be run on the 3D printer.

Ako smo zadovoljni s kvalitetom postavki ispisa, odabiremo opciju ***Export G-code***, čime kreiramo izvršnu datoteku s upravljačkim kodom, koju je potrebno pokrenuti na 3D printeru.

Ha elégedettek vagyunk a nyomtatási beállítások minőségével, akkor a ***Export G-code*** opciót választjuk, amely a vezérlőkóddal egy futtatható fájlt hoz létre, amelyet a 3D nyomtatón kell futtatni.

7.3. 3D printing

7.3. 3D ispis

7.3. 3D nyomtatás

Although 3D printing technology is very similar in different types of 3D printers and different types of materials used, it is important to follow the manufacturer's instructions regarding the use of a specific model of 3D printer.

For the 3D printer Prusa MK3S+, which was used during the implementation of the project, detailed instructions are available at the following link: https://cdn.prusa3d.com/downloads/manual/prusa3d_manual_mk3s_en.pdf#_ga=2.259784777.258587629.1658707338-876609576.1658707338.

The book at the following link can serve as an additional source of knowledge about the basics of 3D printing: https://www.prusa3d.com/page/basics-of-3d-printing-with-josef-prusa_490.

Iako je tehnologija 3D ispisa vrlo slična kod različitih vrsta 3D printera i različitih vrsta korištenih materijala, važno je slijediti upute proizvođača u vezi korištenja konkretnog modela 3D printera.

Za 3D printer Prusa MK3S+, koji je korišten tijekom provedbe projekta, detaljne upute dostupne su na sljedećoj poveznici: https://cdn.prusa3d.com/downloads/manual/prusa3d_manual_mk3s_en.pdf#_ga=2.259784777.258587629.1658707338-876609576.1658707338.

Kao dodatan izvor znanja o osnovama 3D ispisa može poslužiti knjiga na sljedećoj poveznici: https://www.prusa3d.com/page/basics-of-3d-printing-with-josef-prusa_490.

Bár a 3D nyomtatási technológia nagyon hasonló a különböző típusú 3D nyomtatókban és a különböző típusú anyagokban, fontos, hogy kövesse a gyártó utasításait egy adott 3D nyomtatómodell használatával kapcsolatban.

A projekt megvalósítása során használt Prusa MK3S+ 3D nyomtatóhoz a részletes útmutató az alábbi linken érhető el: https://cdn.prusa3d.com/downloads/manual/prusa3d_manual_mk3s_en.pdf#_ga=2.259784777.258587629.1658707338-876609576.1658707338.

Az alábbi linken található könyv további ismeretforrásként szolgálhat a 3D nyomtatás alapjairól: https://www.prusa3d.com/page/basics-of-3d-printing-with-josef-prusa_490.

Vocational school Đurđevac
Dr. Ivana Kranjčeva 5, 48350 Đurđevac
http://ss-strukovna-djurdjevac.skole.hr/
+385 48 812 223
ured@ss-strukovna-djurdjevac.skole.hr

Educational District Centre of Kaposvár
Szántó u. 5, 7400, Kaposvár
http://kk.gov.hu/kaposvar
+36 82 795 240
peter.stickel@kk.gov.hu

Primary school Ferdinandovac
Dravska 66, 48356 Ferdinandovac
http://os-ferdinandovac.skole.hr/
+385 48 817 709
ured@os-ferdinandovac.skole.hr

PORA Regional Development Agency of Koprivnica Križevci County provided techical assistance to the lead beneficiary Vocational School Đurđevac and beneficiary Primary school Ferdinandovac in preparation and implementation of the project.