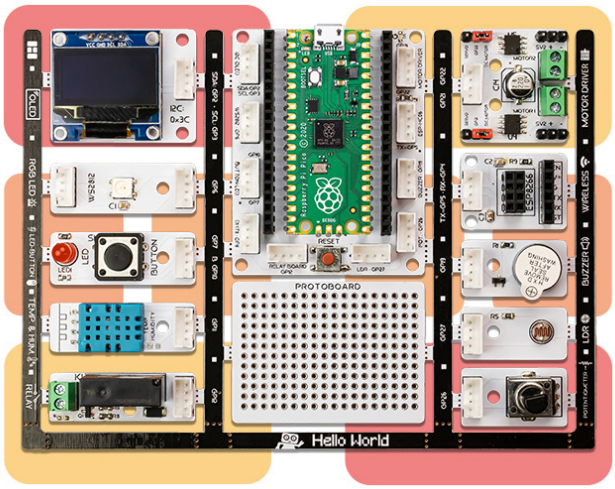
Project Book

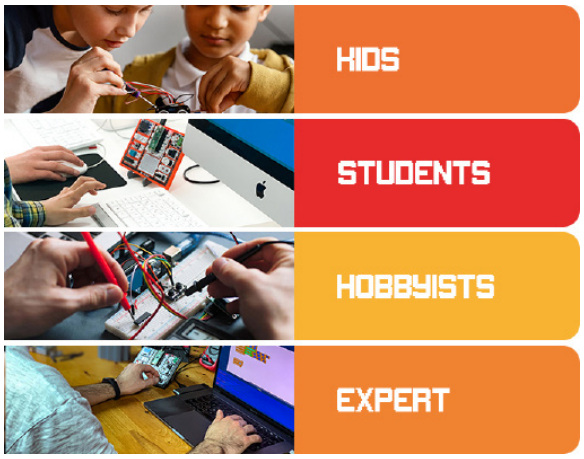
İÇİNDEKİLER

PicoBricks Nedir?

PicoBricks, maker projelerinde kullanılmak üzere tasarlanmış bir elektronik geliştirme kartı ve yazılımıdır. On adet ayrılabilir modülü olan PicoBricks ile çeşitli projeler yaratabilirsiniz. Ayrıca kendi modüllerinizi eklemek için kullanabileceğiniz bir alan da bulunur!



PicoBricks elektronik ve kodlamayla ilgilenen herkes için uygundur. Yeni başlayanlar için , PicoBricks'in modüler donanım tasarımı, Scratch benzeri blok kodlama ortamı ve simülatörü sayesinde başlangıç yapmak oldukça kolay olacaktır. Tecrübesi olanlar elektroniği daha derinlemesine inceleyebilir veya Python'da kodlama yapmayı geliştirebilir. Uzman makerlar bile PicoBricks ile oldukça hızlı prototipler yapılmasını beğeneceklerdir.



Diğer geliştirme boardlarından farklı olarak PicoBricks, her seviyedeki makerlar için inanılmaz bir esnekliğe sahiptir! Bricks IDE'de farklı projeler ve senaryolar için örnek kodlar vardır.

MicroBlocks ve PicoBricks'in sürükle-bırak özelliği ile sıfırdan en iyiye kodlamayı öğrenin. MicroBlocks şimdiye kadar yaratılmış ve maker endüstrisinde yaygın olarak bilinen en kolay kodlama deneyimini sunar.

Bir sorun mu var? **support@robotistan.com** adresine yazabilirsin.

POWERED BY
MicroBlocks

KODLAMA ORTAMLARI

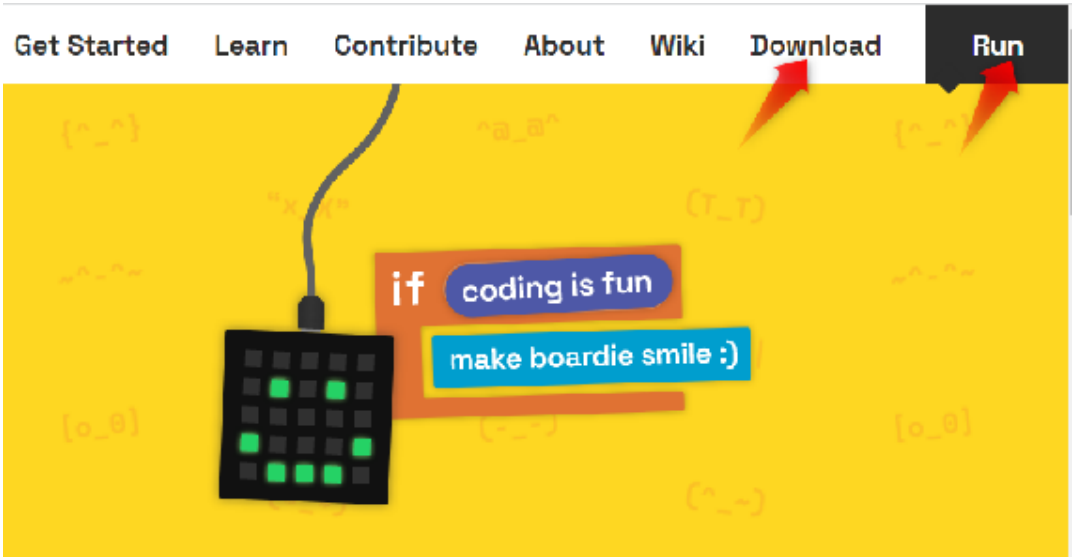
1. Kodlama Ortamları

Bilgisayarda kod yazmak için programcılar genellikle bir yazılıma ihtiyaç duyarlar. İyi bir kodlama ortamı, kod yazan kişiye öneriler sunmalıdır. Ayrıca kodunu çalıştırdıktan sonra bir hata varsa bunu programcıya göstermeli ve hata hakkında ayrıntılı bilgiler sunmalıdır. Picobricks'i hem metin tabanlı hem de blok tabanlı editörler kullanarak kodlayabilirsin. MicroBlocks, Picobricks'i bloklarla kodlayabileceğin güçlü özellilkleri olan bir editördür. Micropython öğrenmeye yeni başlayanlar için Thonny editörü en doğru seçimlerden biridir. İster sadece MicroPython kodla istersen bir elektronik devre kartını kodla Thonny birçok desteği sana sağlamaktadır. Arduino C diline hakimsen ya da C dilini öğrenmek istiyorsan Picobricks ve Arduino IDE senin için çok iyi seçimler olacaktır.

1.1. MicroBlocks Kodlama Ortamı

MicroBlocks, Picobricks için blok tabanlı kodlar yazarak projeler geliştirebileceğin online ve offline editörleri olan bir platformdur. MicroBlocks gerçek zamanlı olarak çalışan bir ortamdır. Kod bloğuna tıkladığında ya da başlat butonuna tıkladığında kodlar Picobricks'e otomatik olarak yüklenir ve çalıştırılır. Kodların derlenip indirilmesini beklemene gerek kalmaz. MicroBlocks'ta kodları bir kez çalıştırdığında USB bağlantısını kesebilir ve Picobricks'i farklı bir güç kaynağı ile besleyebilirsin. Kart üzerindeki kodlar otomatik olarak çalışacaktır. Picobricks'i metin tabanlı olarak kodlamak için Thonny IDE ve Arduino IDE'yi kullanabilirsin.

Picobricks'i MicroBlocks ile kodlamak için https://microblocks.fun/ adresini tarayıcıda açalım (Google Chrome ve Edge tarayıcılar tavsiye edilmektedir).



MicroBlocks'u bir Chrome veya Edge tarayıcısında çalıştırmak için herhangi bir şey yüklemen gerekmez; ekranın sağ üst kısmındaki menüde Run butonuna tıklayarak online editörü çalıştırabilir. Download butonuna tıklayarak işletim sistemine uygun sürümü indirerek bilgisayarınıza kurabilirsin.

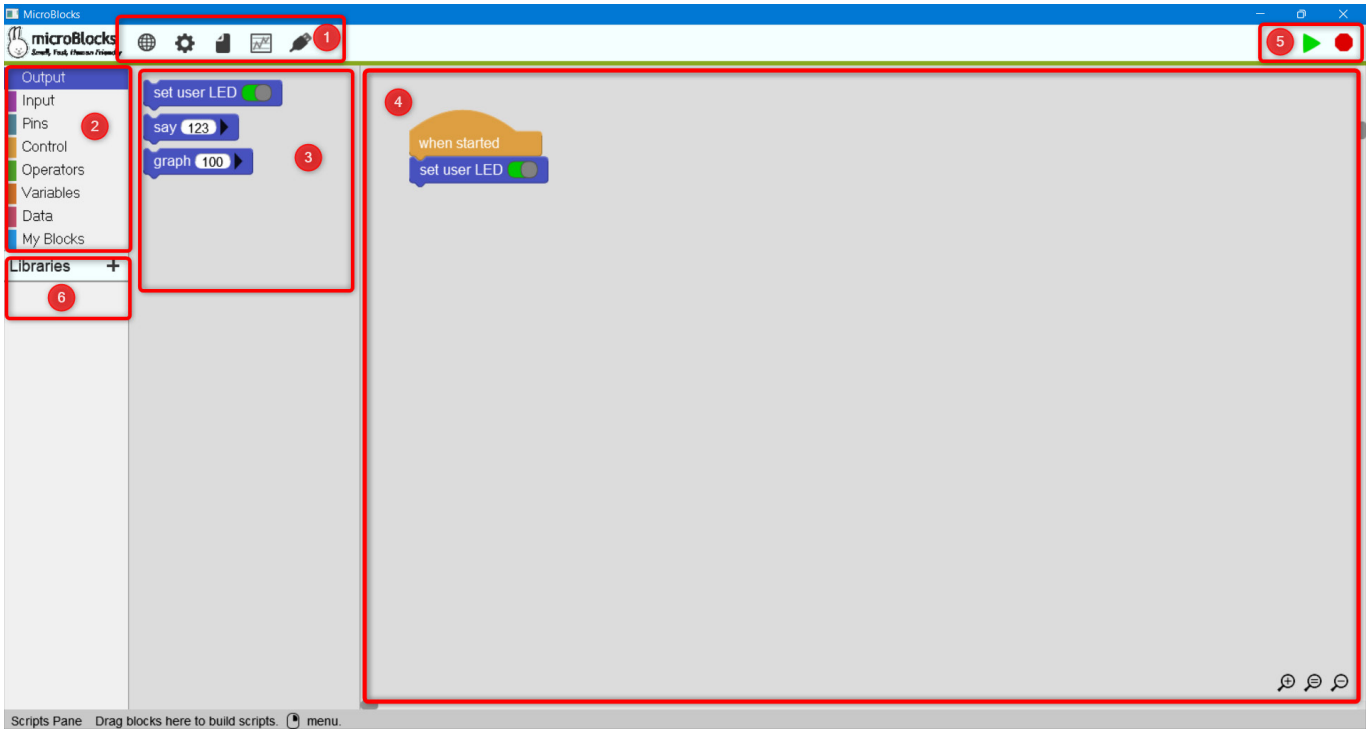
MicroBlocks Web editörü tarayıcına kaydedebilir ve internet erişimi olmadan da kullanabilirsin. MicroBlocks Web uygulamasını kaydetmek için tarayıcında MicroBlocks'u çalıştır, ardından tarayıcının URL çubuğunun sağ üst köşesindeki yükle düğmesini tıkla.

Google Chrome

Microsoft Edge

1.1.1. Arayüz Tanıtımı

MicroBlocks programını açtığında görseldeki gibi bir arayüz seni karşılayacaktır. Aşağıda program arayüzünün detaylı anlatımını inceleyebilirsin.



1. Menü Bar ( ): Bu kısımda soldan sağa birinci buton, programın dil seçeneğini değiştirmemizi sağlar. İkinci buton, gelişmiş MicroBlocks kod seçeneklerini görebildiğimiz ve firmware güncellemesi yapılan menü iken üçüncü buton kaydetme seçeneklerini sunar. Dördüncü buton, verileri çizmek için grafik bloğu tarafından kullanılan bir grafik penceresi açarken en sağdaki beşinci buton, Picobricks'e bağlanmak için kullanılır.

2.Block Categories: Bu alan, MicroBlocks'ta programlamak için kullanılan blokların kategorilerini içerir. Her kategori için farklı renkler kullanılarak kategoriler gruplandırılmıştır. Kategoriler seçildikçe, o kategorideki ilgili bloklar, 3 numaralı Blok Palette alanında listelenecektir.

3.Block Palette: Block categories alanında seçimler yapıldıkça, bu alanda belirli işlevlere sahip bloklar listelenecektir. Bu alandaki blokları 4 numaralı Scripting area kısmına sürükleyip bırakarak kodlar yazılır.

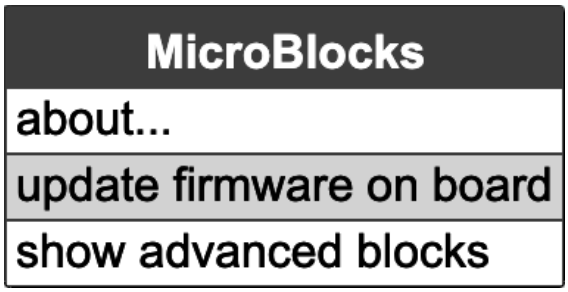
4.Scripting Area: Bu, tüm kodlama etkinliklerinin gerçekleştiği alandır. Kullanıcı komut dosyaları ve özel bloklar (fonksiyonlar) oluşturmak için blokları bu alana sürüklenip bırakılır.

5. Start/Stop Buttons ( ): Bu alanda, MicroBlocks programlarının yürütülmesini kontrol etmek için kullanılan Başlat ve Durdur adlı iki simge bulunmaktadır.

6. Library List (Libraries +): Bu alanda, kullanıcı komut dosyalarının ve mikro cihazların gereksinimlerine bağlı olarak yüklenen kütüphaneler bulunmaktadır.

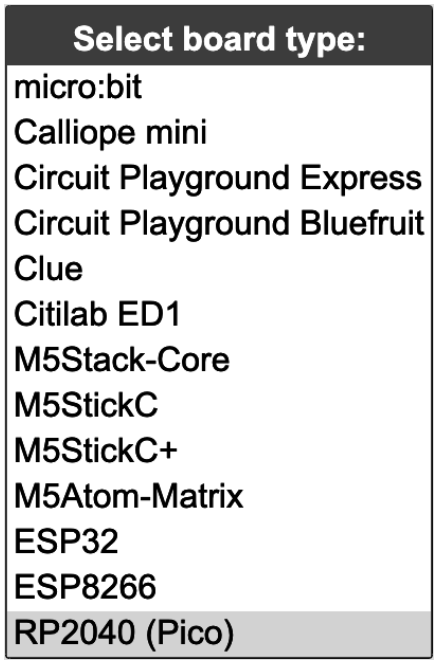
1.1.2. MicroBlocks-Picobricks Bağlantısı ve Çalıştırma

Picobricks'i offline editöre bağlamak için Raspberry Pi Pico üzerindeki beyaz BOOTSEL düğmesini basılı tutarken kartı USB kablo ile bilgisayarınıza bağlamalısın.



MicroBlocks offline editörü aç ve MicroBlocks menüsünden önce MicroBlocks butonuna (dişli simgesi), sonra update firmware on board seçeneğine tıkla.

Firmware kurulumu sadece birkaç saniye sürecek ve bittiğinde MicroBlocks Picobricks'e otomatik olarak bağlanacaktır.

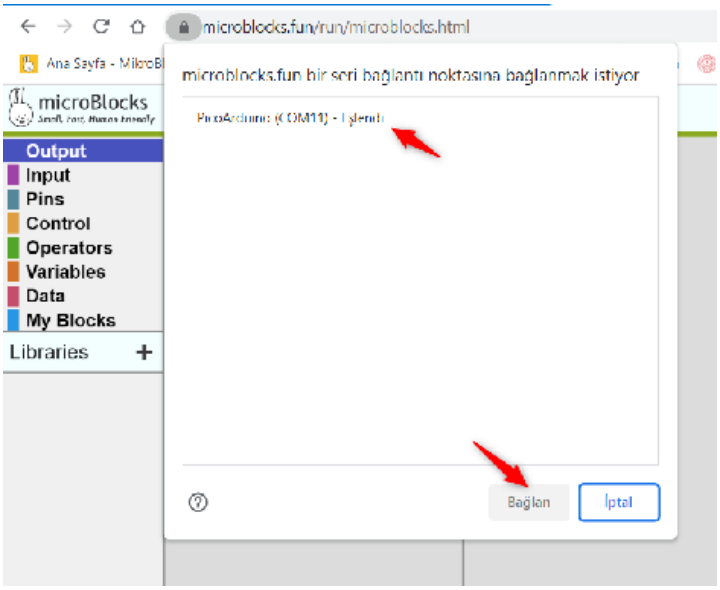


Picobricks'i online editöre bağlamak için ise extra birkaç işlem gereklidir. Güvenlik nedeniyle tarayıcı, kullanıcıya sormadan kartın USB sürücüsüne erişemez. İlk önce menüden MicroBlocks butonuna (dişli simgesi), sonra update firmware on board seçeneğine tıkla ve açılan listeden RP2040 (Pico) seçeneğine tıklayarak kart tipini seç.

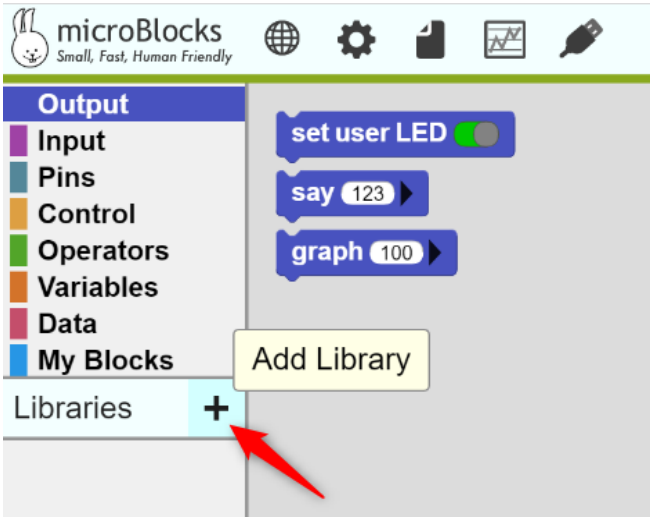
Kart seçimini yaptığında Firmware İnstall penceresi açılacaktır. Ok butonuna tıkladığında vm_pico adında .uf2 türündeki firmware dosyasını kaydetmeni isteyecektir. Bilgisayarında herhangi bir konuma bu dosyayı kaydet.

Daha sonra Pico üzerindeki beyaz BOOTSEL düğmesini basılı tutarken kartı bilgisayarına bağla. Kartı taktığında içindeki dosyaların bulunduğu yeni bir klasör açılacaktır. İndirdiğin vm_pico.uf2 dosyasını bu klasöre kopyala ve klasörün otomatik olarak kapanmasını birkaç saniye bekle.

Bağlan butonuna tıkladığında mikro aygıtların takılı olduğu sistem USB bağlantı noktalarını görüntülenir. Bu pencerede önce Pico aygıtını seçip sonra Bağlan butonlarına tıklayarak Picobricks'i MicroBlocks'a bağlayabilirsin. Bağlantı başarılı olduğunda, USB simgesinin arkasında yeşil daire belirecektir.

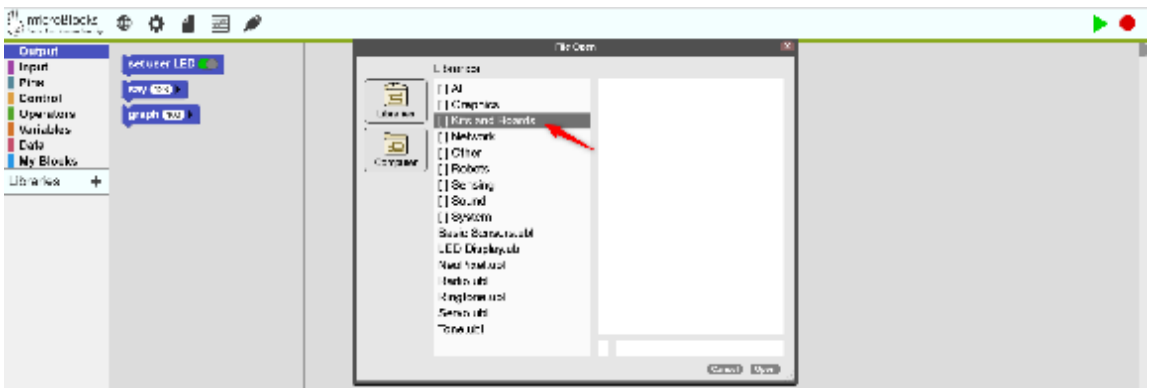


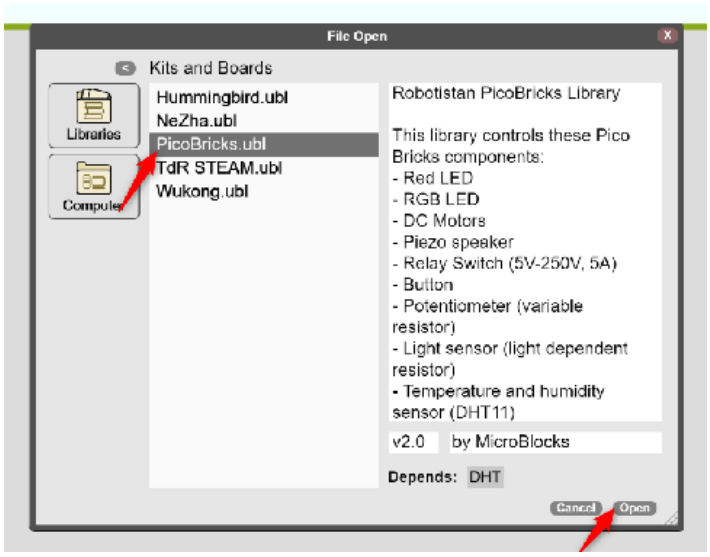
MicroBlocks gerçek zamanlı çalışan bir kodlama editörüdür. Kodların yazıldıktan sonra derlenerek karta yüklenmesi işlemleri yoktur. Kod bloklarının üzerine tıkladığında kodlar çalışacaktır. İlk olarak Picobricks'in kütüphanesini Microblocks editörüne import etmen gerekiyor. Bunun çin Add Library butonuna tıklamalısın.

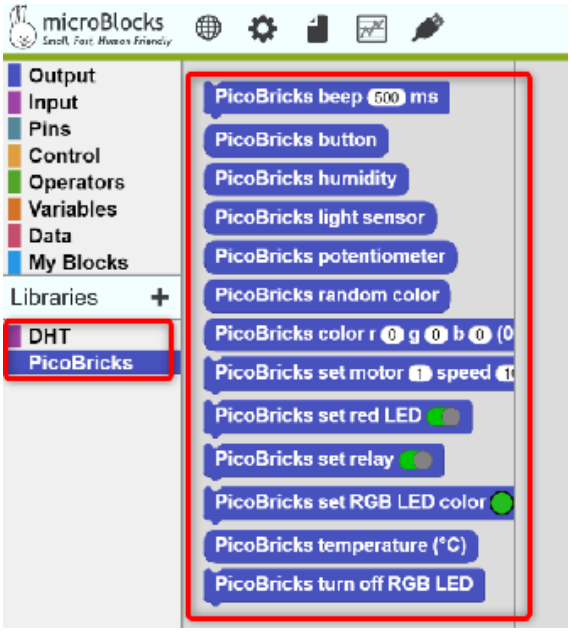


Açılan File Open penceresinde Kits and Boards butonuna tıklayarak Microblocks ile kodlayabileceğin cihaz listesini aç.

Açılan listeden PicoBricks.ubl seçeneğine ve sonra Open butonuna tıkla.







Her şey yolunda gittiyse Kod blokları panelinde PicoBricks kütüphanesi ve kod blokları görüntülenecektir.

Şimdi ilk kodumuzu yazarak çalıştıralım. Önce control menüsündeki when started bloğunu sürükleyerek kod yazma alanına bırak. Sonra Picobricks kategorisinden Picobricks set red LED bloğunu sürükleyerek when started bloğunun altına ekle. Start tuşuna bastığında Picobricks üzerindeki kırmızı ledin yandığını göreceksin.

MicroBlocks'ta kodlarınızı düzenledikten sonra Start butonuna tıkladığında kodların Picobricks'e yüklenerek çalıştırılacaktır.

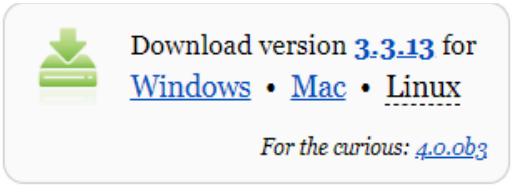
Stop butonu ise kodların çalışmasını durdurur. Fakat Picobricks'e yüklenen kodlar silinmez. USB bağlantısını kesebilir, Picobricks'i harici besleme ile çalıştırabilirsin.

Daha önce Picobricks'i MicroBlocks ile kodlamak için gerekli firmware dosyasını Pico'ya yüklediysen USB simgesine tıklayarak bağlantıyı sağlayabilirsin. İlk defa MicroBlocks Picobricks bağlantısı kuracaksan başlık 1.1.2 de ki adımları izleyebilirsin.

Microblocks editörü kullanımı hakkında detaylı bilgi için,

https://wiki.microblocks.fun/ide adresini ziyaret edin.

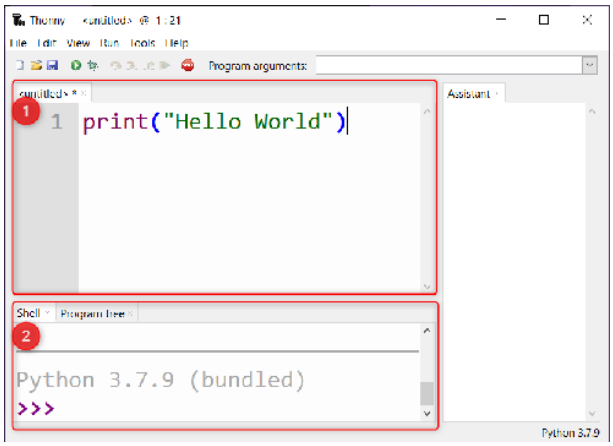
1.2. Thonny IDE (MicroPython) Kodlama Ortamı



Picobricks'in kalbinde Raspberry Pi Pico yer almaktadır. Thonny Raspberry Pi Pico'yu dolayısıyla Picobricks'i kodlamak için harika bir seçimdir.

1.2.1 Thonny Kurulumu

https://thonny.org/ adresini ziyaret et. Sisteminize uygun versiyonu seçip bilgisayarına indir. Ardından kurulumu gerçekleştir. Komut istemcisini kullanarak $ pip install thonny komutuyla da Thonny IDE'yi bilgisayarınıza yükleyebilirsin.



1.2.2. Thonny Arayüzü

Thonny'i başlattığında aşağıdaki gibi bir pencere ile karşılaşırsın. 1 no'lu kısıma kodlarımızı yazacağız. 2 no'lu kısımda kodlarımızın çıktılarını göreceğiz.

A: Boş bir kod dosyası açar.

B: Varolan bir kod dosyanı açmanı sağlar.

C: Üzerinde çalıştığın kod dosyandaki değişiklikleri kaydetmeni sağlar.

D: Yazdığın kodu belirlediğin yorumlayıcı ortamda çalıştırır.

E: Kodunda hata denetimi yapmanı sağlar.

F: Hata ayıklamak için kod satırlarını sırasıyla çalıştırmanı sağlar.

G: Hata ayıklarken kod satırındaki komutlar içinde dolanmanı sağlar.

H: Hata ayıklamadan çıkmanı sağlar.

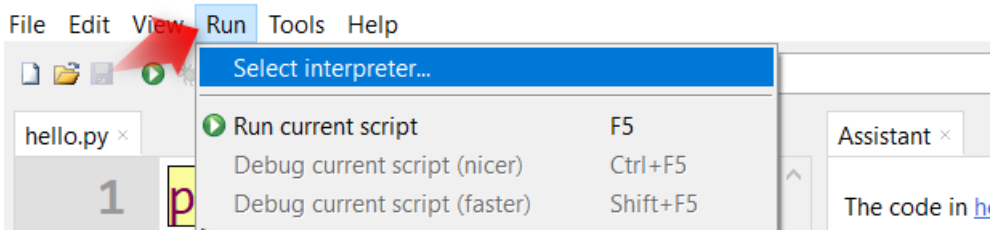
I: Hata ayıklama modundan çalıştırma moduna geçiş yapmanı sağlar.

J: Kodunun çalışmasını sona ermesini sağlar.

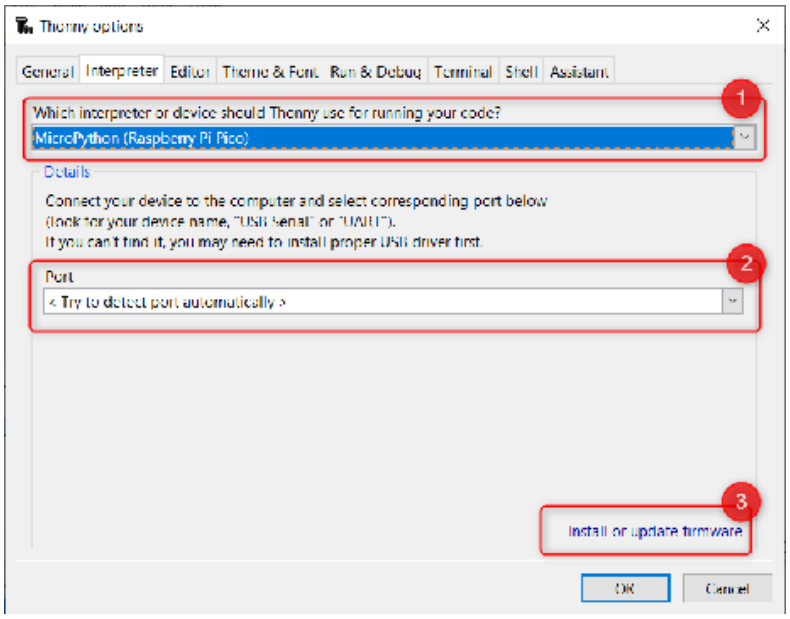
1.2.3. Raspberry Pi Pico'ya MicroPython Firmware Yükleme

Raspberry Pi Pico'nun yazacağımız MicroPython kodlarını anlayabilmesi için ona özel bir işletim sistemi kurmalıyız. Buna firmware diyoruz. Thonny editörünü açıp Run menüsünden Select interpreter seçeneğine tıkla.

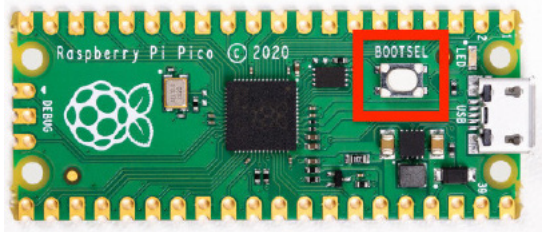
1 no'lu alanda gösterilen açılır listeden Raspberry Pi Pico'yu seç. 2 no'lu alanı görseldeki gibi bırak, 3 no'lu alana tıkla.



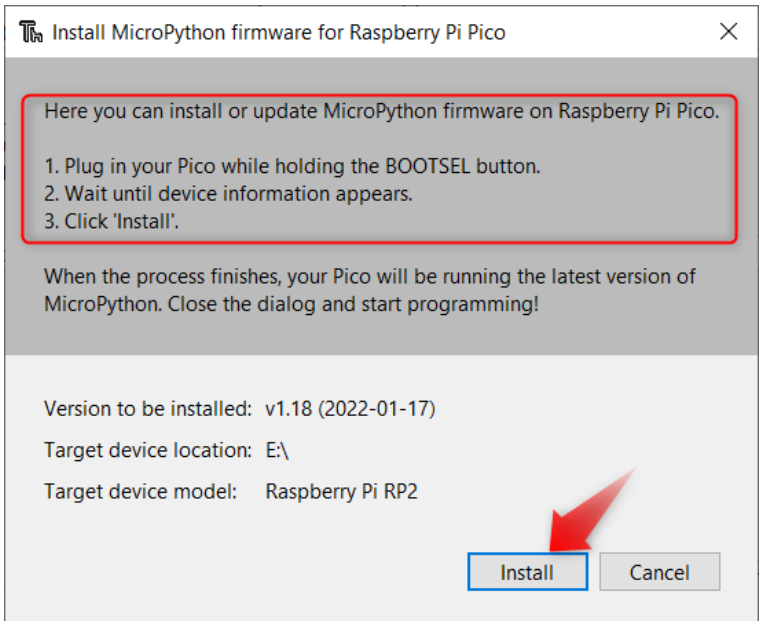
Pico'yu, üzerindeki beyaz renkli bootsel butonuna basılı tutarken bilgisayarının USB portuna kablo ile bağla



Install butonun aktif hale geldikten sonra butonu bırakabilirsin. Install butonuna basıp firmware'ın yüklemesini bekle.



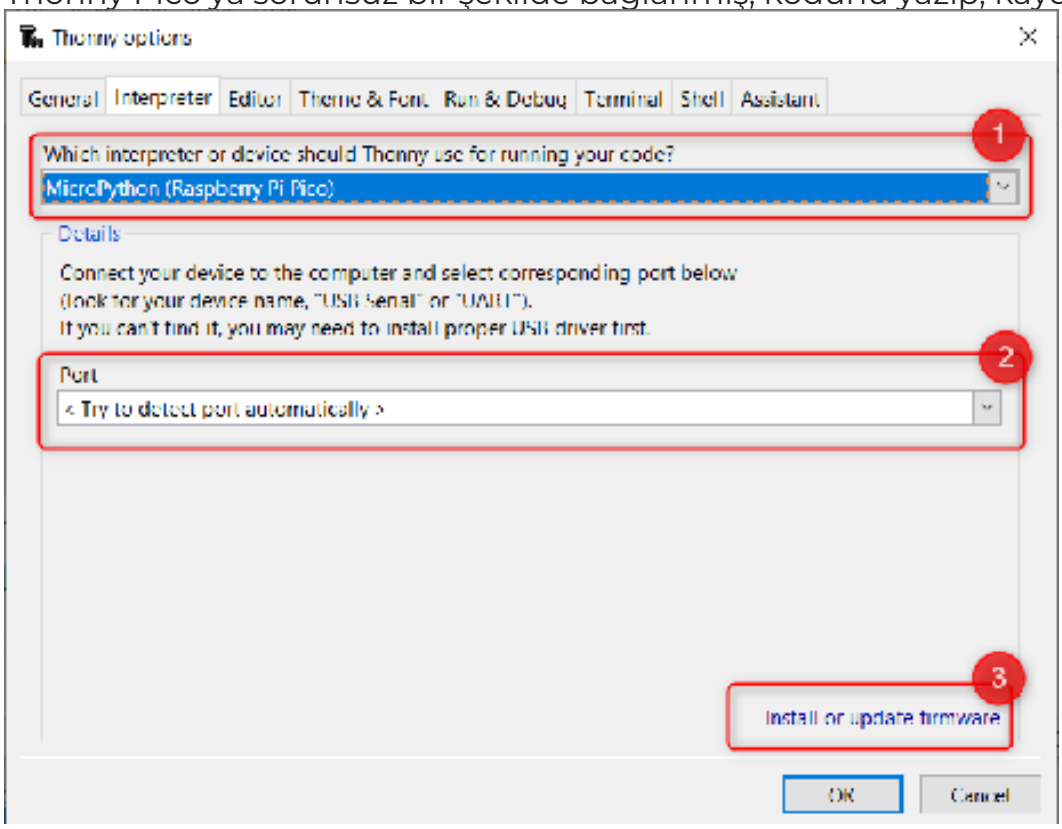
Yükleme tamamlandıktan sonra Close butonuna tıklayıp kurulumu tamamla.

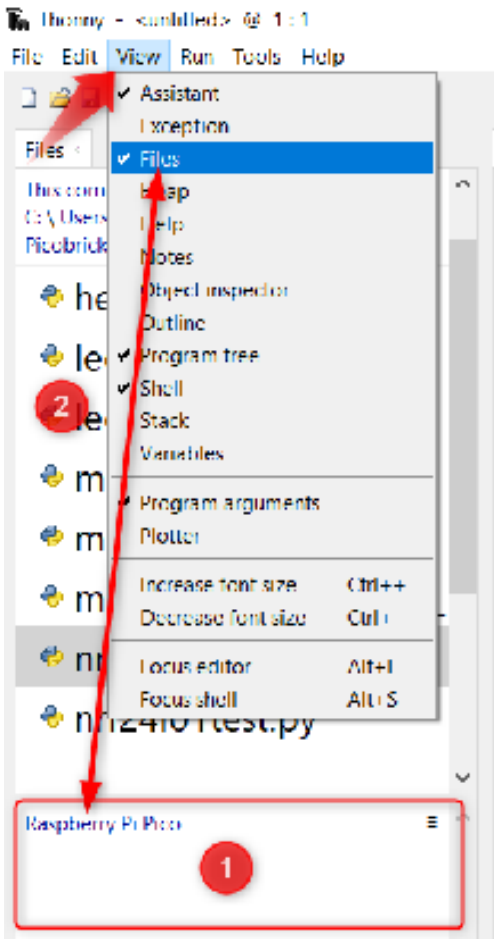


1.2.4. Raspberry Pi Pico'ya Kod Yükleme ve Çalıştırma

Pico'nun kablosunu doğrudan bilgisayarın USB portuna tak. Bootsel butonuna basılı tutmana gerek yoktur. Thonny'de Run menüsünden Select interpreter seçeneğini seç. 1 no'lu kısımda Raspberry Pi Pico'nun seçili olduğundan emin ol. OK butonuna tıklayarak pencereyi kapat.

View menüsünden Files seçeneğini aktifleştir. Ekranın sol tarafına uzun bir dosya gezgini sekmesi yerleşecektir. 1 no'lu kısımda Raspberry Pi Pico ifadesini görüyorsan Thonny Pico'ya sorunsuz bir şekilde bağlanmış, kodunu yazıp, kaydedip çalıştırmaya

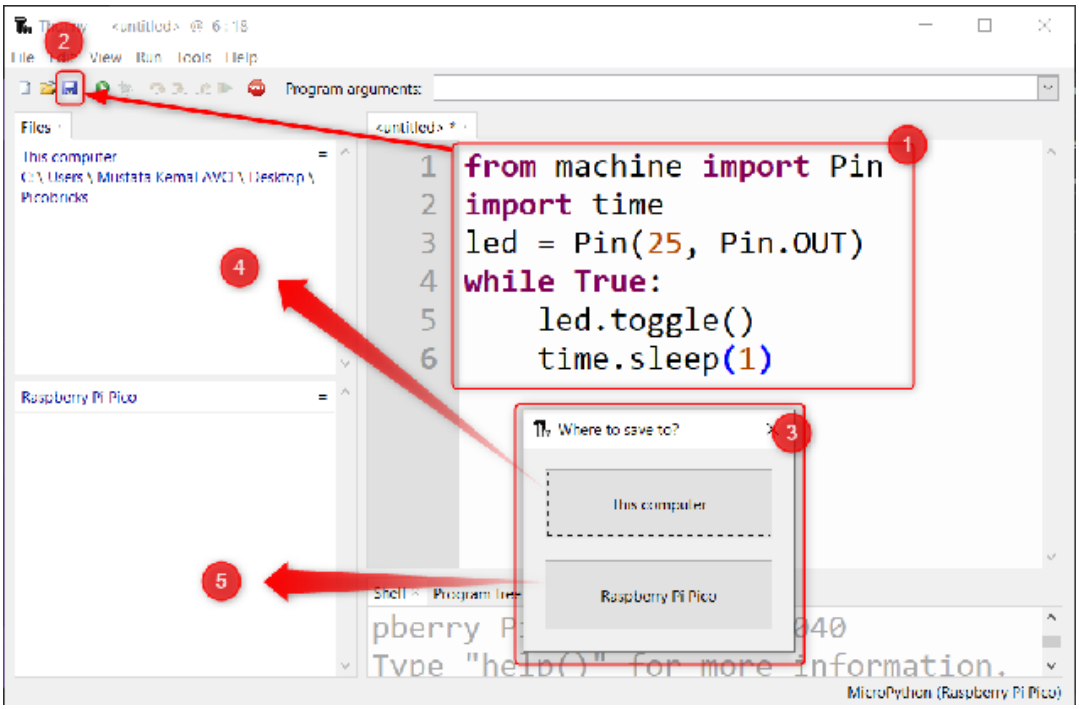




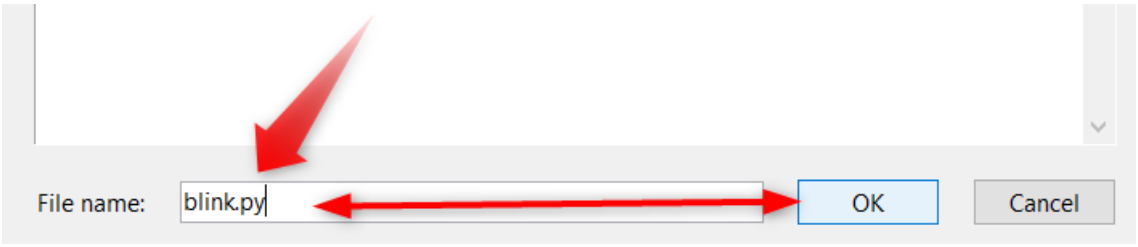
hazırsın demektir. Menünün arkasında kalan 2 no'lu kısım ise bilgisayarındaki çalışma dizinini gösteren dosya gezgini alanıdır.

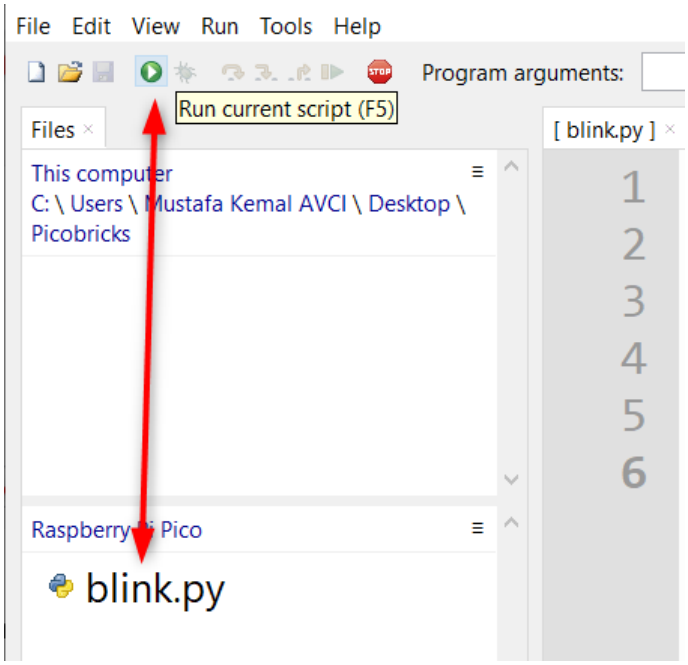
Thonny'de yazdığın MicroPython kodları Raspberry Pi Pico vb. mikro kontrol kartları için düzenlenmiş kütüphanelerden oluşur ve MicroPython adını alır. Yazım kuralları ve hemen hemen tüm kütüphaneler MicroPython ile aynı çalışmaktadır.

Yazılım dünyasının "hello world" uygulaması ne ise fiziksel programlama için "blink" uygulaması odur. 1 no'lu alanda gösterilen kodu yaz. 2 no'lu alandaki kaydetme düğmesine tıkla. Thonny sana 3 no'lu alandaki pencere ile kodunu bilgisayarındaki çalışma dizinine mi yoksa Pico'nun yerleşik belleğine mi kaydetmek istediğini sorar. Bilgisayarını seçersen oluşacak dosya 4 no'lu alanda, Pico'yu seçersen oluşacak dosya 5 no'lu alanda gözükecektir.



Kayıt yeri penceresinden Raspberry Pi Pico'yu seçip File Name alanına "blink.py" yazıp OK düğmesine tıkla.





Pico'nun dosya gezgini alanında "blink.py" dosyasını gördükten sonra klavyeden F5 tuşuna ya da araç çubuğundan yeşil renkli Run butonuna tıkladığında kod dosyası Pico tarafından çalıştırılacaktır. Pico'nun üzerindeki dahili LED'in 1 saniye aralıkla yanıp söndüğünü görüyorsan ilk kodunu başarılı bir şekilde yazıp çalıştırmışsın demektir. Tebrikler :)

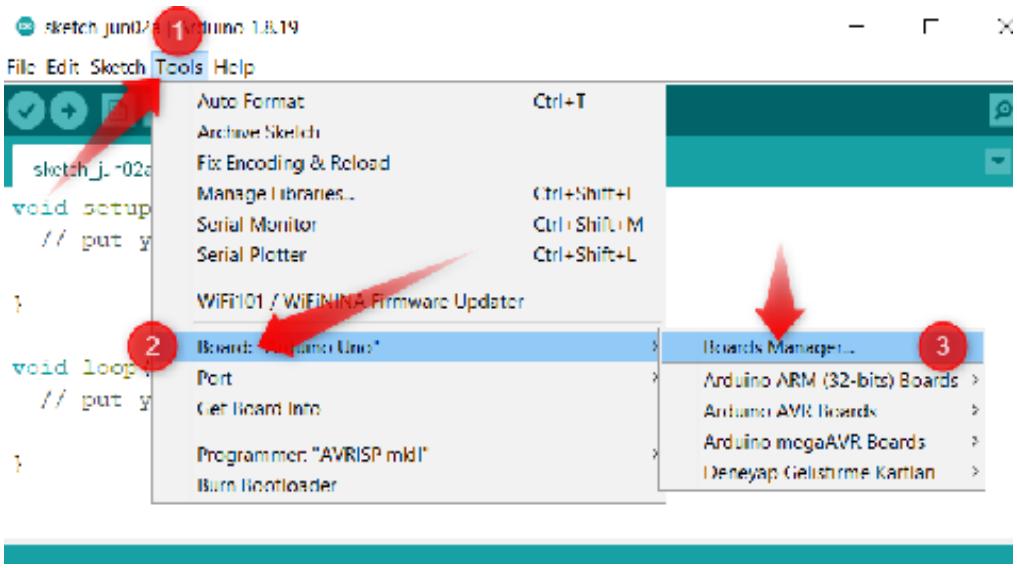
Önemli bir not: Yazdığın kodların çalıştır komutu vermeden, Pico açılır açılmaz çalışmasını istersen kodunu "main.py" adıyla Pico'nun ana dizinine kaydetmelisin.

1.3. Arduino IDE Kodlama Ortamı

Picobricks bize Arduino C ile kodlama imkanı sunmaktadır. Picobricks'in kalbindeki Raspberry Pi Pico'yu yaygın olarak kullanılan Arduino IDE ile kodlamaya başlamak oldukça kolay.

https://www.arduino.cc/en/software bağlantısından Arduino IDE 1.8.x kurulum dosyasını bilgisayarına indirip kurulumu yap.

Öncelikle Arduino IDE'ye Raspberry Pi Pico'yu eklemelisin. Arduino IDE'yi başlat. Ardından Tools>Board>Boards Manager yolunu izle.

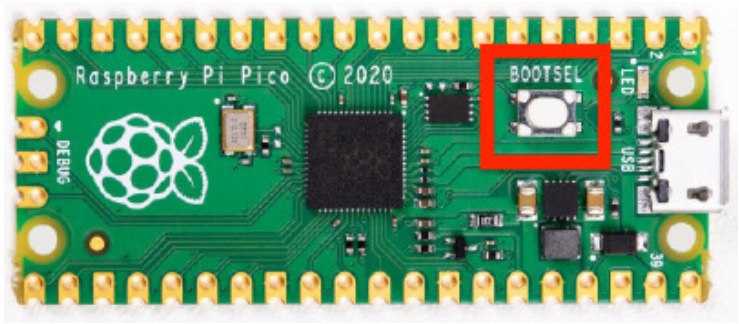


1 no'lu alana "Raspberry Pi Pico" yaz. Biraz beklediğinde Arduino Mbed OS RP2040 Boards seçeneğine tıklayıp 2 no'lu alandaki install butonuna tıkla.

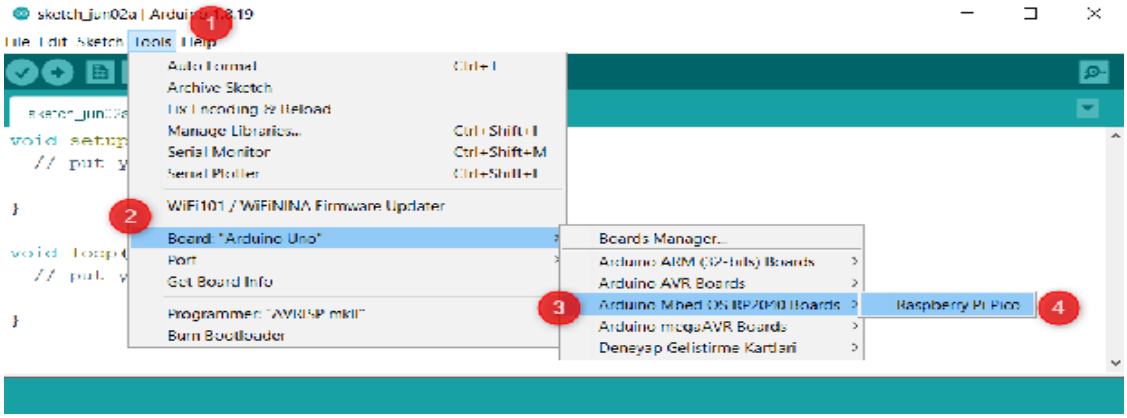
Tüm bu kurulumlar sırasında senden isteyeceği onayları kabul etmelisin. Kurulum tamamlanıp close butonuna tıkladığında Pico'yu Arduino IDE'ye eklemiş olursun.

1.3.1. Arduino IDE ile Kod Yazımı ve Çalıştırma

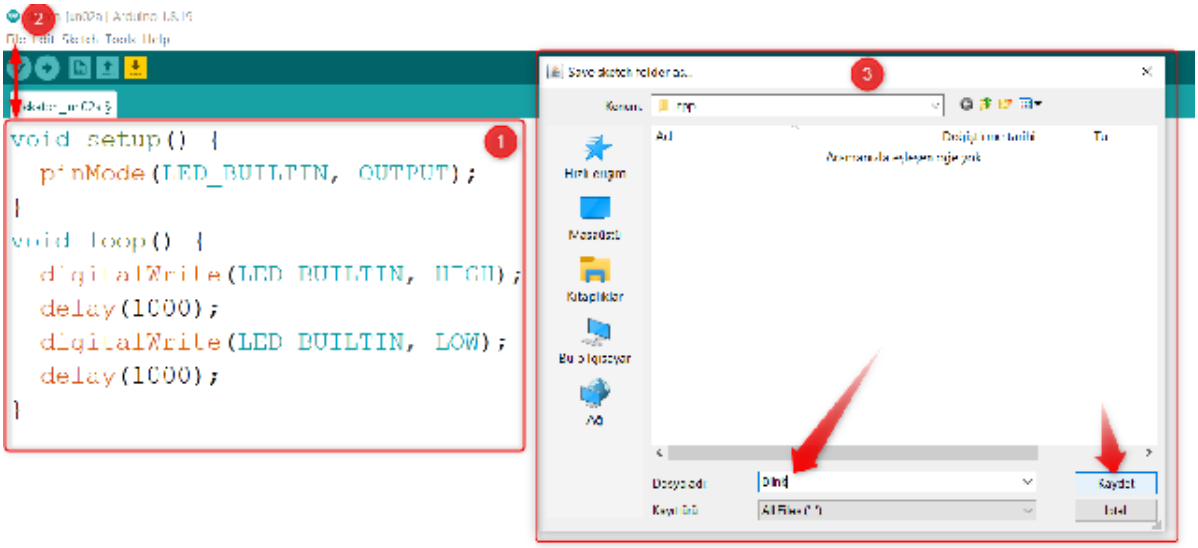
Pico'yu Arduino IDE ile kodlamayı istediğinde, sadece ilk seferinde BOOTSEL butonuna basılı tutarak bilgisayarınıza bağlamalısın.

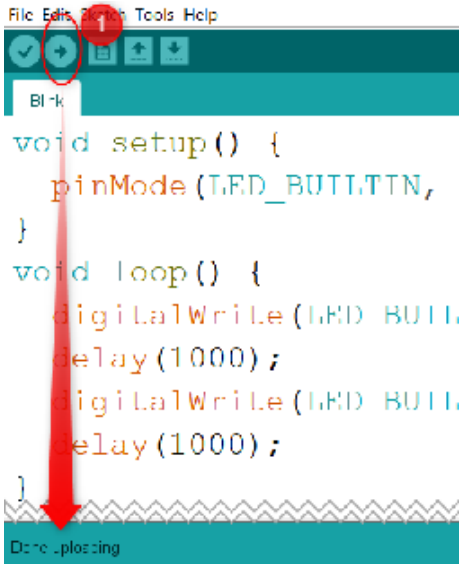


Bu sayede Pico'yu bootloader modunda bağlanacak, harici bellek olarak bilgisayarın tarafından tanınacaktır. Bootsel butonuna basılı tutarak Pico'yu bilgisayarına bağla. Pico'yu bilgisayarın flash bellek olarak gördükten sonra Tools>Board>Arduino Mbed OS RP2040 boards> Raspberry Pi Pico yolunu izleyerek kartınızı aktif hale getir.



Aşağıdaki 1 no'lu alandaki kodu hatasız şekilde yazıp File>Save yolunu izleyerek bilgisayarında herhangi bir yere "Blink" adıyla kaydet.





Kaydetme işleminin ardından 1'nolu alandaki "Upload" butonuna tıklayarak kodun derlenip Pico'nun içine kaydedilmesini sağlamalıyız. Alt kısımda Done uploading ifadesini gördüğümüzde kodumuz Pico'da çalışacak ve dahili LED 1 saniye aralıklarla yanıp sönecektir. Önemli Not: Arduino IDE ile Picobircks'i kodlarken Micropython veya Microblocks firmwarelarından ilk geçişte BOOTSEL butonuna basarak bilgisayarına bağla. Sonraki kod yüklemelerinde BOOTSEL'e basmana gerek yoktur. Keyifli projeler :)

PROJELER

2.1. Blink

Gerçek hayatta işi yeni öğrenmeye başlayan çalışan, önce en temel görevi üstlenir. Temizlik görevlisi ise süpürgenin kullanımını, aşçı ise mutfak araç gereçlerini, garson ise tepsi taşımayı... Bu örnekleri arttırabiliriz. Yazılım geliştirmeye yeni başlayanların ilk yazdıkları kod "Hello World" olarak bilinir. Kullandıkları dil de ekrana ya da konsol penceresine program başlar başlamaz "Hello World" yazdırmak, programlamaya atılan ilk adımdır. Bir bebeğin emeklemeye başlaması gibi... Robotik kodlamaya diğer adıyla fiziksel programlamaya atılan ilk adım ise Blink uygulamasıdır. Robotik kodlamaya göz kırpmak anlamını taşır. Basit olarak bir LED'in bağlantısını devre kartına yaparak yapılan kodlama ile LED'in sürekli yanıp sönmesini sağlanır. Robotik kodlama alanında kendini geliştirmiş kişilere bu seviyeye nasıl geldiklerini sorun. Size verecekleri cevap şöyle başlar; her şey bir LED yakmak ile başladı!

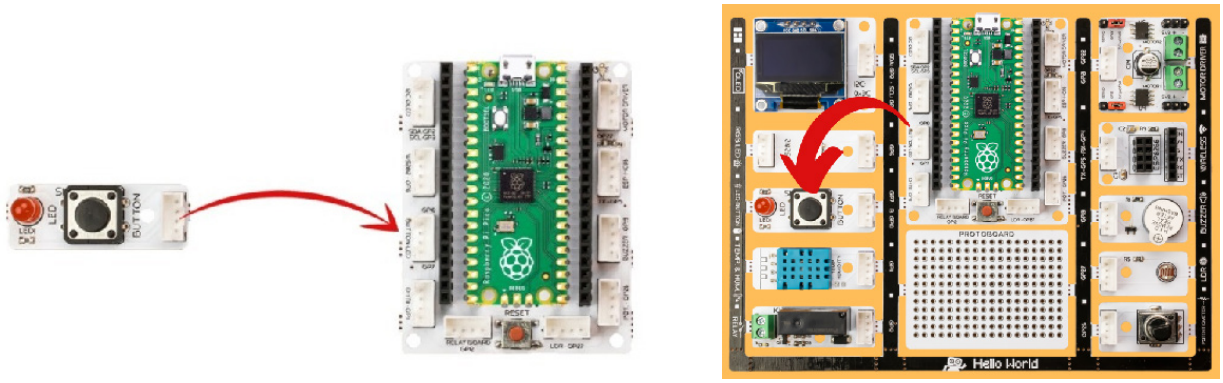
LED'ler elektronik cihazların dilidir. LED'ler sayesinde programcı cihazın görevin hangi aşamasında olduğunu, varsa sorunun ne olduğunu, hangi seçeneklerin aktif olduğunu kullanıcılara ifade eder. Bu projede Picobricks ile üzerinde yer alan LED'lerin çeşitlerini öğrenip onları nasıl yakıp söndüreceğini öğreneceksin.

2.1.1. Proje Detayları ve Algoritma

Picobricks üzerinde 1 adet 5mm kırmızı LED, 1 adet WS2812B RGB LED bulunmaktadır. Normal LED'ler tek renk yanabilirken RGB renkler hem ana hem ara renkler olmak üzere farklı renklerde yanabilmektedir. Bu projede Picobricks üzerindeki kırmızı LED kullanacağız.

Projede Picobricks üzerindeki kırmızı LED yakılması, belirli bir süre geçtikten sonra söndürülmesi, tekrar belirli bir süre geçtikten sonra yakılması ve bu işlemlerin sürekli tekrarlanması için gerekli kodlar yazacağız.

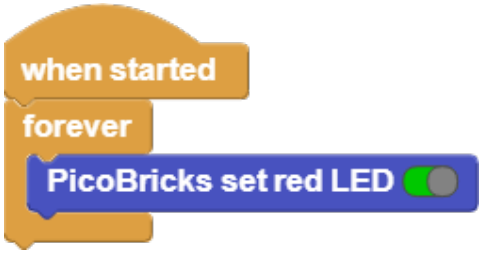
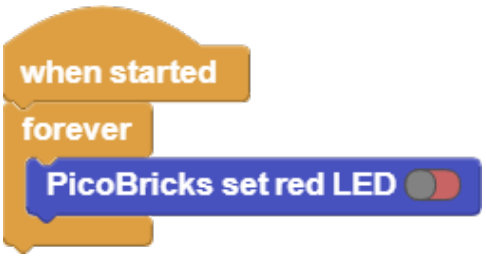
2.1.2. Bağlantı Şeması

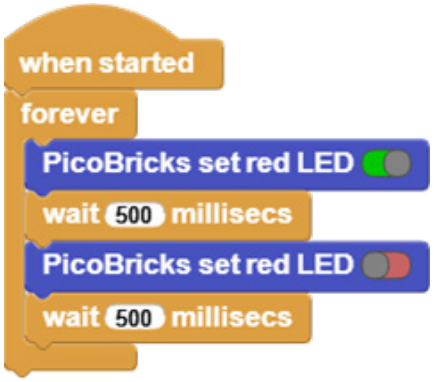


Picobricks'in modüllerini kablo bağlantısı yapmadan kodlayabilir ve çalıştırabilirsin. Modülleri board'dan ayırarak kullanacaksan grove kablolar ile modül bağlantılarını yapmalısın.

2.1.3. Projenin MicroBlocks ile Kodlanması

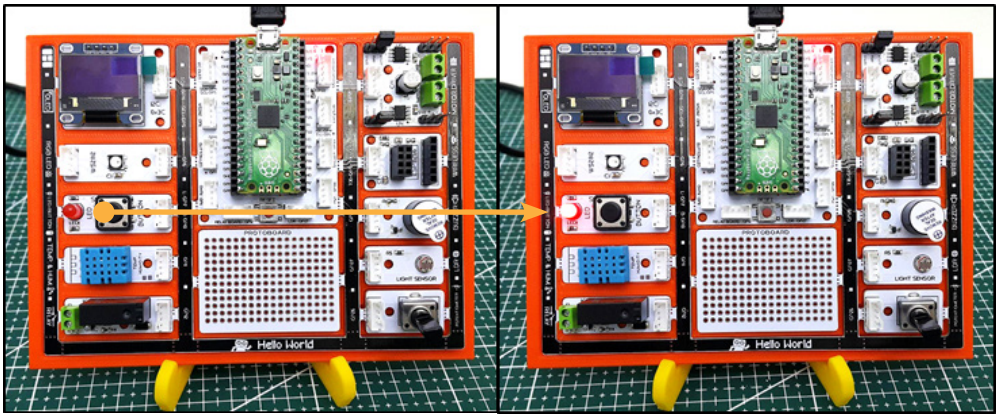
MicroBlocks-Picobricks bağlantısı ve kütüphane yükleme işlemleri yaptıysan ilk proje için takip etmen gereken adımlar aşağıdaki tabloda ayrıntılı olarak belirtilmiştir.

1	İlk olarak Picobricks başladığında yazdığınız kodların çalışabilmesi için Control menüsünde when started bloğunu sürükleyerek kod yazma alanına bırak.	
2	Daha sonra yazdığın kodların Picobricks çalıştığı sürece sürekli olarak çalışması için yine Control menüsünden forever bloğunu sürükleyerek when started bloğunun altına ekle.	
3	Kırmızı LED'in yanması için Picobricks kütüphanesindeki kod blokları arasından PicoBricks set red LED bloğunu sürükleyerek forever bloğunun içine bırak. Start tuşuna basarak kırmızı LED'in yanıp yanmadığını test et.	
4	Şimdi kırmızı LED'i söndürmek için Picobricks set red LED bloğundaki onay kutusuna bir kere tıklayarak onay kutusunu kırmızı yani kapalı konuma getir ve tekrar Start tuşuna basarak LED'in sönüp sönmediğini test et.	
5	Kırmızı ledi kod bloğuyla yakıp söndürdükten sonra belirli zaman aralıklarıyla ledin kendi kendine yanıp sönmesi için gerekli kodları yazacağız. Control kategorisinden wait 500 millisecs bloğunu sürükleyerek PicoBricks set red LED bloğunun altına ekle.	

6	Şimdi tekrar Picobricks set red LED bloğunu wait 500 millisecs bloğunun altına ekle ve onay kutusunu kapalı konuma getir. Daha sonra tekrar wait 500 millisecs bloğunu en alta ekle. Start tuşuna bastığında Picobricks üzerindeki kırmızı ledin 500 milisaniye aralıklarla yanıp söndüğünü göreceksin. wait 500 millisesc bloğundaki 500 sayısı milisaniyeyi temsil eder. Bu sayıyı dilediğin gibi değiştirebilirsin. 1000 yaptığında kırmızı LED 1000 milisaniye yani 1 saniye aralıklarla yanıp sönecektir.	

Projenin MicroBlocks kodlarına erişmek için tıkla.

2.1.4. Proje Görseli



2.1.5. Proje Önerisi

Farklı zaman aralıkları ile LED'i yakabilir miyiz? Örneğin; birkaç kez saniyede bir, birkaç kez yarım saniyede bir LED'in yanıp sönmesi.

2.1.6. Projenin MicroPython Kodları

```
from machine import Pin
import utime
led = Pin(7, Pin.OUT)

while True:
    led.toggle()
    utime.sleep(0.5)
```

2.1.7. Projenin Arduino C Kodları

```
void setup() {
  pinMode(7, OUTPUT);
}

void loop() {
  digitalWrite(7, HIGH);
  delay(500);
  digitalWrite(7, LOW);
  delay(500);
}
```

2.2. Etki - Tepki

Newton hareket yasalarında açıkladığı gibi her etkiye karşı bir tepki oluşur. Elektronik sistemler kullanıcılardan komutlar alırlar ve görevlerini yerine getirirler. Genellikle bu iş için bir tuş takımı, dokunmatik ekran veya bir buton kullanılır. Elektronik cihazlar görevlerinin sona erdiğini ve görev süresince olan bitenden kullanıcıyı haberdar etmek için sesli, yazılı veya görsel olarak tepki verirler. Bu tepkilerin kullanıcıyı haberdar etmenin yanı sıra olası arızada hatanın nerede olabileceğinin anlaşılmasında yardımcı olabilmektedir.

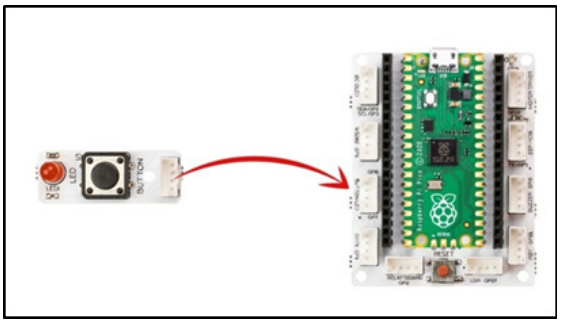
Bu projede Picobricks'in buton-LED modülünü kodlayarak projelerinde kullanıcıdan bir komutun nasıl alınacağını ve bu komuta nasıl tepki verileceğini öğreneceksin.

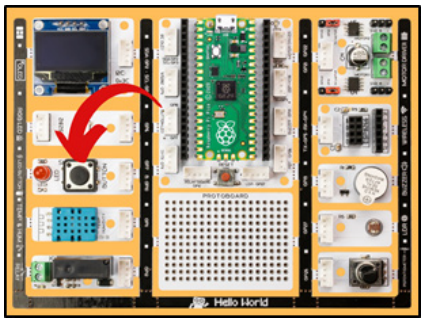
2.2.1. Proje Detayları ve Algoritma

Elektronik sistemlerde farklı türlerde butonlar kullanılmaktadır. Kilitli butonlar, push butonlar, anahtarlamalı butonlar... Picobricks üzerinde 1 adet push buton bulunmaktadır. Anahtar gibi çalışmaktadırlar, basıldığında akımı iletirler bırakıldığında akımı iletmezler. Projede butonun akım iletip iletmediğini kontrol ederek basılma durumunu anlayacağız. Basılmışsa LED'i yakacak, basılmamışsa LED'i söndüreceğiz.

2.2.2. Bağlantı Şeması

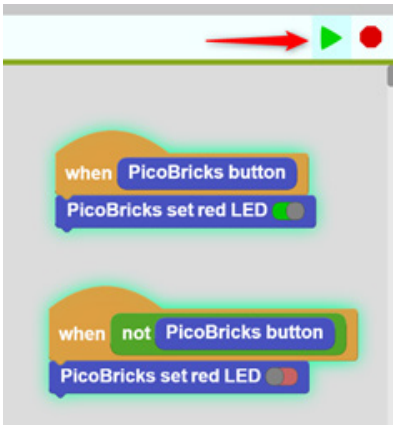
Picobricks'in modüllerini kablo bağlantısı yapmadan kodlayabilir ve çalıştırabilirsin. Modülleri board'dan ayırarak kullanacaksan grove kablolar ile modül bağlantılarını yapmalısın.





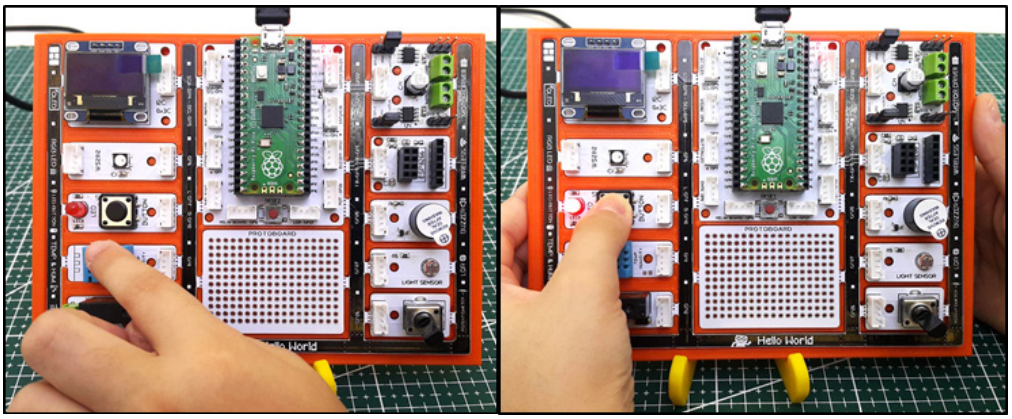
2.2.3. Projenin MicroBlocks ile Kodlanması

1	Butona basılma durumunda çalıştırmamız gereken bir komut olduğu için Control kategorisindeki when bloğunu sürükleyelim. Bu blok belirlediğimiz koşulu sürekli denetler koşul doğrulanırsa altındaki komutu çalıştırır.	
2	Koşulumuz Picobricks'in butonuna basılması olduğu için Picobricks kategorisindeki PicoBricks button bloğunu when bloğuna yerleştir.	
3	Koşul gerçekleştiğinde Picobricks de bulunan kırmızı ledin yanmasını istiyoruz. Bu yüzden Picobricks kategorisindeki PicoBricks set red LED bloğunu when bloğunun altına yerleştir.	
4	Butona basılmadığında kırmızı LED'in sönük kalmasını istiyoruz. O yüzden Control kategorisinden ikinci when bloğunu sürükle. Koşul alanına ise butona basılmadığında ifadesini oluşturabilmek için operators kategorisindeki  bloğunu yerleştir.	

5	Picobricks'in butonuna basılmadığında ifadesini oluşturmak için Picobricks kategorisindeki PicoBricks button bloğunu not bloğuna yerleştir.	
6	Butona basılmayan her an kırmızı ledin sönük kalmasını istiyoruz. O yüzden Picobricks kategorisinden PicoBricks set red LED bloğunu when bloğunun altına yerleştirip anahtarını kapalı hale getir.	
7	MicroBlocks'un Start butonuna bastığında kodlar gerçek zamanlı çalışacaktır. Picobricks deki butona bastığında kırmızı LED yanacak, bıraktığında sönecektir.	

Projenin MicroBlocks kodlarına erişmek için tıkla.

2.2.4. Proje Görseli



2.2.5. Proje Önerisi

Bu projede butona basıldığında LED'in yanması buton bırakıldığında LED'in sönmesi sağlanmıştır. Sen de butona bir kere basılıp bırakıldığında LED'in yanması, tekrar basılıp bırakıldığında LED'in sönmesi için gerekli kodları yazabilirsin.

2.2.6. Projenin MicroPython Kodları

```
from machine import Pin
led = Pin(7, Pin.OUT)
push_button = Pin(10, Pin.IN,Pin.PULL_DOWN)

while True:
  logic_state = push_button.value()
    if logic_state == True:
      led.value(1)
  else:
      led.value(0)
```

2.2.7. Projenin Arduino C Kodları

```
void setup() {
  pinMode(7, OUTPUT);
  pinMode(10,INPUT);
}
void loop() {
  if (digitalRead(10) == 1){
        digitalWrite(7, HIGH);
  }
  else{
    digitalWrite(7, LOW);
  }
  delay(10);
}
```

2.3. Otonom Aydınlatma

Elektronik sistemlerin verilen görevi, topladığı veriler doğrultusunda karar vererek otomatik olarak yapmasına otonom olma durumu denir. Elektronik sistemlerin çevresinden veri toplamasını sağlayan bileşenlerine sensör (algılayıcı) denir. Ortamdaki ışık seviyesinin hangi düzeyde olduğu, hava sıcaklığının kaç derece olduğu, su akış hızının kaç lt/dk olduğu, ses şiddetinin ne kadar olduğu gibi birçok veri sensörler tarafından toplanarak elektrik sinyalleri olarak yani veri olarak PicoBricks'e iletilir. Picobricks'te birçok sensör yer almaktadır. Sensörlerden verinin nasıl alındığını ve bu verilerin nasıl yorumlanılıp kullanılacağını bilmek, kitap okumanın kelime dağarcığını geliştirmesi gibi proje fikirlerini geliştirecektir.

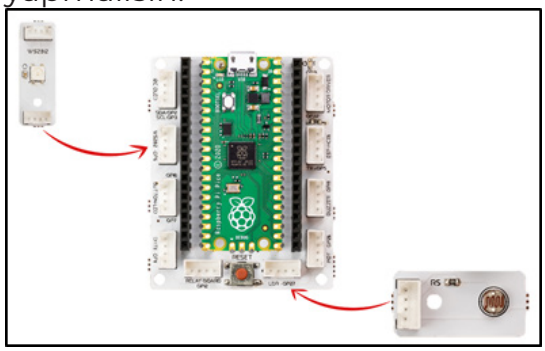
Bu projede PicoBricks ile hava karardığında aydınlatmanın otomatik yandığı sistemlerin çalışma sistemlerini anlamak için ışık miktarı düştüğünde LED'in yanmasını sağlayacağız.

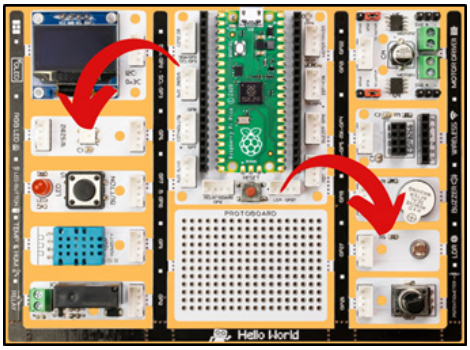
2.3.1. Proje Detayları ve Algoritma

Sensörler dış ortamlardaki verileri algılayarak mikrokontrolcülere veriler gönderen elektronik bileşenlerdir. LDR sensör de ortamdaki ışık miktarını algılayarak analog değerler gönderir. Projemizde önce LDR sensör değerlerini okuyarak ortam aydınlık ve karanlık olduğunda gelen verileri kontrol edecek, sonra bu verilere göre bir sınır belirleyerek ışık miktarı bu sınırın altındaysa Picobricks'in RGB LED'ini yakacak değilse LED'i söndüreceğiz.

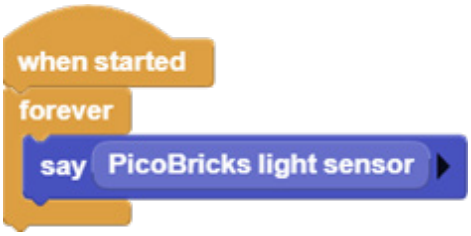
2.3.2. Bağlantı Şeması

Picobricks'in modüllerini kablo bağlantısı yapmadan kodlayabilir ve çalıştırabilirsin. Modülleri board'dan ayırarak kullanacaksan grove kablolar ile modül bağlantılarını yapmalısın.

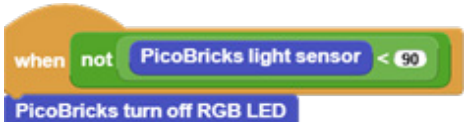
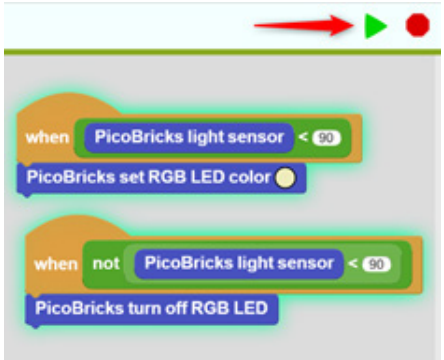




2.3.3. Projenin MicroBlocks ile Kodlanması

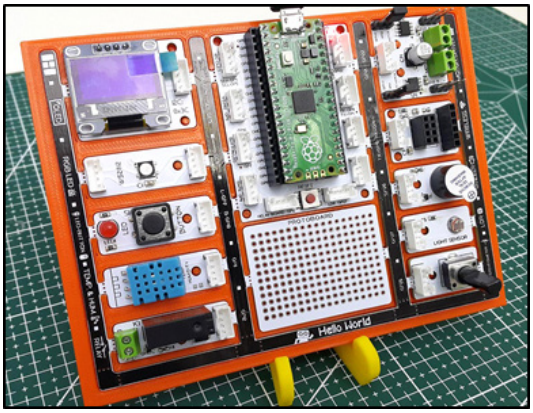
1	İlk olarak Picobricks başladığında yazdığın kodların çalışabilmesi için Control menüsünde when started bloğunu sürükleyerek kod yazma alanına bırak.	
2	Daha sonra yazdığın kodların Picobricks çalıştığı sürece sürekli olarak çalışması için Control menüsünden forever bloğunu sürükleyerek when started bloğunun altına ekle.	
3	Otonom aydınlatmayı sağlamak için ilk olarak LDR sensörden ortam aydınlıkken ve karanlıkken gelen değerleri görmemiz ve bu değerlere göre işlem yapmamız gerekiyor. Bunun için Output kategorisindeki say 123 bloğunu sürükleyerek forever bloğunun içine bırak.	
4	Daha sonra Picobricks kategorisindeki PicoBricks light sensor bloğunu sürükleyerek say bloğundaki 123 yazan yuvarlağa bırak. Start tuşuna basarak sensörden gelen değerleri gör. Light Sensor bloğu ortamdaki ışık seviyesini yüzdesel (%) olarak verir. Tam aydınlık ortamda 100 değeri, elinle kapattığında 90 ve altında , tam karanlıkta ise 0'a yakın değerler görmelisin. Değerleri gördükten sonra kodları silebilirsin.	
5	Şimdi ortam karanlık olduğunda RGB LED'in yanması için Control kategorisindeki when bloğunu sürükleyerek kod yazma alanına bırak. Bu blok when started bloğundan farklı olarak belirlediğimiz koşulu sürekli denetler koşul doğrulanırsa altındaki komutu çalıştırır. when started bloğu ise Start tuşuna bastığınız andan itibaren altındaki blokları çalıştırır.	

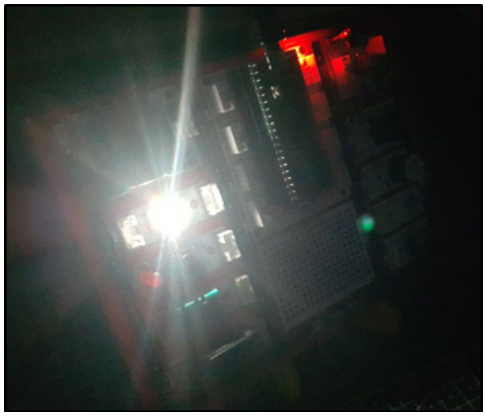
6	Daha sonra when bloğuna koşulu tanımlamak için operators kategorisindeki 3<4 bloğunu sürükleyerek when bloğundaki yuvarlak kısma bırak. 3<4 bloğu koşul işlemlerinde büyüklük-küçüklük kontrolü yapmaktadır. Bloktaki 3 ve 4 yazan alanlara kontrol etmek istediğiniz blokları yerleştirebilirsin. Programlamada eşittir, eşit değildir, büyüktür, küçüktür gibi farklı operatörler sıklıkla kullanılmaktadır. Biz de bu projede sensörden gelen değerin 90 sayısından küçük olup olmadığını kontrol edeceğiz.	
7	Şimdi Picobricks kategorisinden PicoBricks light sensor bloğunu sürükleyerek when bloğunda 3<4 operatöründeki 3 yazan yuvarlağın içine bırak.	
8	when bloğunda koşulu tamamlamak için 4 sayısını silerek klavyeden 90 yaz. Bu sayede program Picobricks sensör değerlerinin % 90' dan küçük olup olmadığını kontrol edecek ve %90'dan küçük olduğunda when bloğu altındaki kodları çalıştıracaktır.	
9	Ortam karanlık olduğunda yani LDR sensörü değeri 90'dan küçük olduğunda ledin yanabilmesi için Picobricks kategorisindeki PicoBricks set RGB LED bloğunu sürükleyerek when bloğunun altına bırak. Yeşil yuvarlağa tıklayarak açılan renk paletinden dilediğiniz rengi seçebilirsin.	

10	Bu aşamaya kadar ortam karanlık olduğunda LED'in yanması için gerekli kodları yazdık. Şimdi ise ortam aydınlık olduğunda yani koşul gerçekleşmediğinde yapılacak işlemleri programa eklemeliyiz. Bunun için Control kategorisinden tekrar when bloğu alarak kod yazma alanına bırak.	
11	when bloğunda koşul alanına operators kategorisindeki not bloğunu yerleştir. Bu sayede LDR sensör değeri 90' dan küçük değilse koşulunu oluşturabiliriz.	
12	Tekrar operators kategorisindeki 3<4 operatörünü sürükleyerek not bloğuna yerleştir	
13	Picobricks kategorisinden Picobricks light sensor bloğunu sürükleyerek 3<4 operatöründeki 3 yazan yuvarlağa yerleştir ve 4 sayısını silerek 90 yaz.	
14	LDR sensör değeri 90'dan küçük olmadığında rgb ledin sönmesi için Picobricks kategorisindeki PicoBricks turn off RGB LED bloğunu sürükleyerek when bloğunun altına yerleştir	
15	Start tuşuna basarak kodlarını test et. Herşey yolunda gittiyse Picobricks LDR sensörü elinizle kapattığında RGB LED belirlediğiniz renkte yanacak, elini kaldırdığında ise sönecektir.	

Projenin MicroBlocks kodlarına erişmek için tıkla.

2.3.4. Proje Görseli





2.3.5. Proje Önerisi

Bu projede Picobricks üzerindeki LDR sensör verilerine ortam karanlıksa LED'i yaktık aydınlıksa LED'i söndürdük. Sen de LDR sensör verilerini işleyerek evindeki bir gece lambasını ya da masa lambasını karanlıkta otomatik olarak yanması için kodlayabilirsin. Bunun için Picobricks üzerindeki röleyi kullanabilirsin.

2.3.6. Projenin MicroPhyton kodları

```
from machine import Pin, ADC
 ldr = ADC(Pin(27))
led = Pin(7, Pin.OUT)

while True:

    print(ldr.read_u16())
    if(ldr.read_u16()>20000):
        led.on()
    else:
        led.off()
```

2.3.7. Projenin Arduino C Kodları

```
void setup() {
  pinMode(7, OUTPUT);
  pinMode(27,INPUT);
}
void loop() {

  if (analogRead(27)>500){

    digitalWrite(7, HIGH);
  }
  else{
       digitalWrite(7, LOW);
  }
  delay(10);

}
```

2.4. Termometre

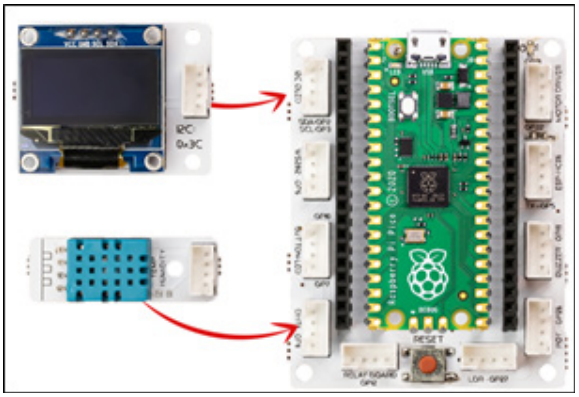
Sensörler elektronik sistemlerin duyu organlarıdır. Hissetmek için derimizi, görmek için gözümüzü, duymak için kulağımızı, tatmak için dilimizi ve koklamak için burnumuzu kullanırız. Picobriks'te hali hazırda bir çok duyu organı(sensör) vardır. Ayrıca yenileri de eklenebilir. Nem, sıcaklık, ışık ve daha birçok sensörü kullanarak çevreyle etkileşim sağlayabilirsiniz. Picobricks ortam sıcaklığını başka hiçbir çevre bileşenine ihtiyaç duymadan ölçebilir.

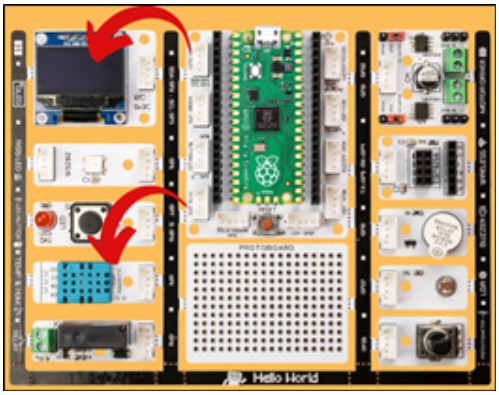
Ortam sıcaklığı seralarda, kuluçka makinelerinde, ilaçların taşınmasında kullanılan ortamlarda kısaca sıcaklık değişiminin sürekli takip edilmesi gereken durumlarda kullanılmaktadır. Projelerinde sıcaklık değişimi üzerine bir işlem yapacaksan ortam sıcaklığını nasıl ölçeceğini bilmelisin. Bu projede Picobricks ile ortam sıcaklığını OLED ekranda gösterecek bir termometre hazırlayacaksın.

2.4.1. Proje Detayları ve Algoritma

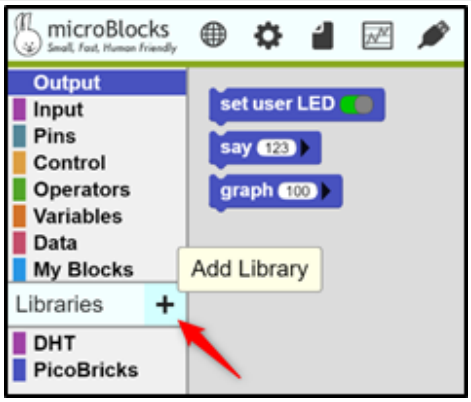
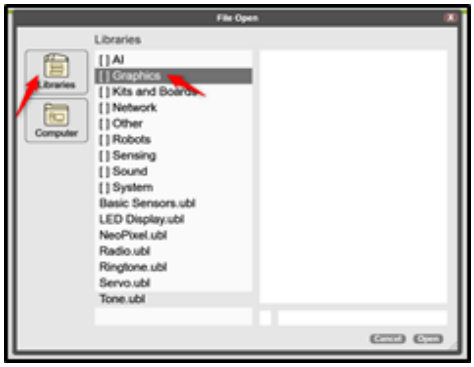
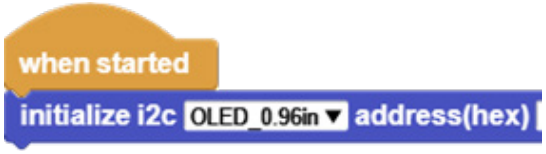
Picobricks'te DHT11 modülü bulunmaktadır. Bu modül ortamdaki sıcaklık ve nem miktarını algılayarak mikrokontrolcüye veriler gönderebilmektedir. Bu bu projede Picobricks üzerindeki DHT11 sıcaklık ve nem sensörünün ölçtüğü sıcaklık değerlerini Picobricks OLED ekrana yazdırmak için gerekli kodları yazacağız.

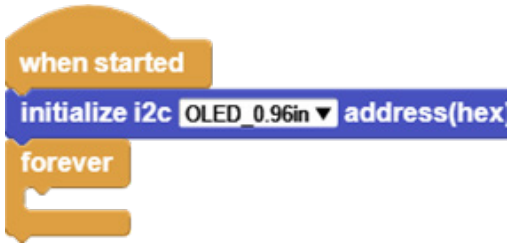
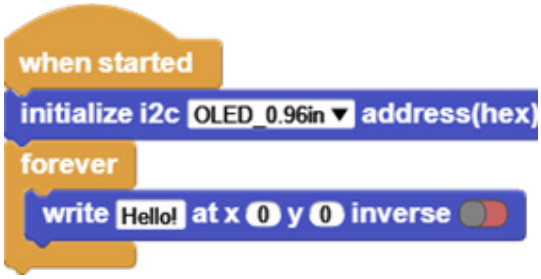
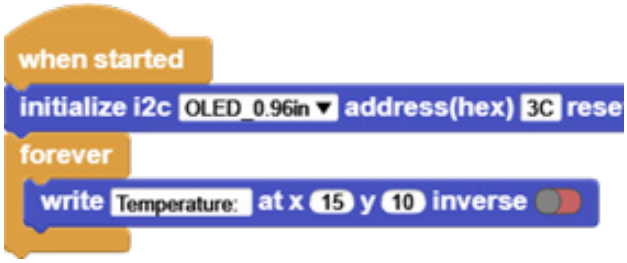
2.4.2. Bağlantı Şeması

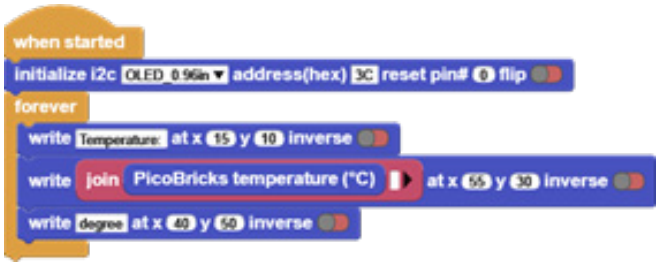




2.4.3. Projenin MicroBlocks ile Kodlanması

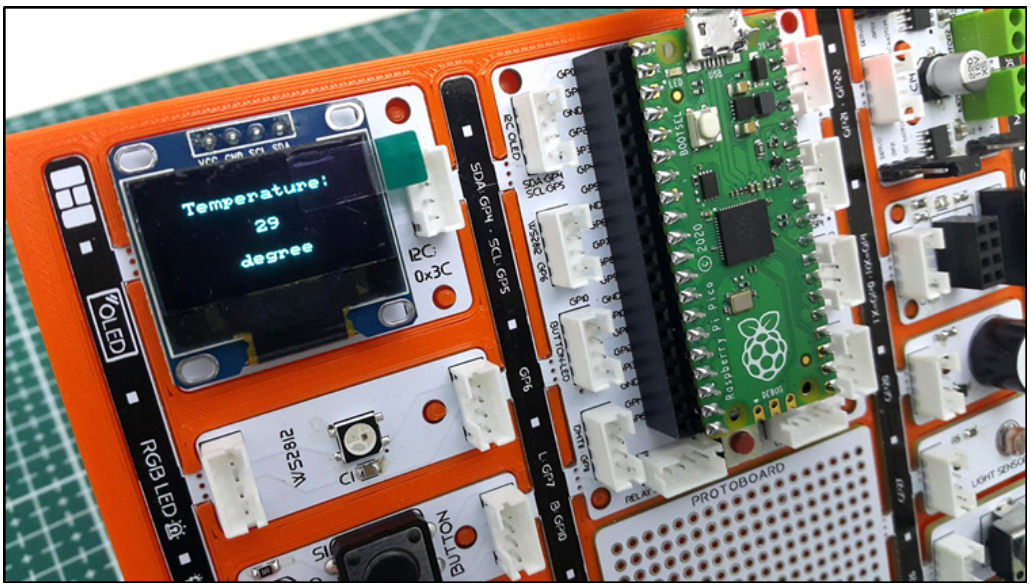
1	İlk olarak OLED kütüphanesini MicroBlocks editörüne import etmeniz gerekiyor. Bunun çin Add Library butonuna tıklamalısın.	
2	Açılan File Open penceresinde önce Library sonra Graphics seçeneklerine tıkla.	
3	Grafik kütüphanelerinden OLED Graphics.ubl kütüphanesine tıkla ve pencerenin sağ alt konumundaki open butonuna tıkla. OLED kütüphanesini kod kategorileri listesine eklendiğini göreceksin.	
4	OLED kütüphanesi ekledikten sonra Picobricks başladığında yazdığın kodların çalışabilmesi için Control menüsünde when started bloğunu sürükleyerek kod yazma alanına bırak	
5	Daha sonra OLED ekranın tanımlama bloğunu eklemen gerekiyor. Bunun için OLED kategorisindeki initialize i2c OLED bloğunu sürükleyerek when started bloğunun altına ekle	

6	Daha sonra yazdığın kodların Picobricks çalıştığı sürece sürekli olarak çalışması için Control menüsünden forever bloğunu sürükleyerek OLED tanım bloğunun altına ekle	
7	OLED ekrana yazı yazdırmak için OLED kategorisindeki write Hello! at x0 y0 inverse bloğunu sürükleyerek forever bloğunun içine bırak. Start tuşuna bastığında ekranda Hello! yazdığını göreceksin. Buradaki Hello! yazısını silip dilediğin metni yazarak ekranda görüntülenmesini sağlayabilirsin.	
8	Şimdi write bloğundaki Hello! yazısını silip Sicaklik: yaz ve x konumunu 15, y konumunu 10 yap.	
9	Tekrar OLED Graphics kategorisinden write bloğu al ve forever bloğunun içine sürükleyip bırak ve x konumunu 55, y konumunu 30 yap. OLED ekrana sayısal değerler yazdırmak için öncelikle bu sayısal değerleri metinsel ifadeye dönüştürmelisin. Bunun için Data kategorisinden join micro blocks bloğunu sürükleyerek write bloğundaki Hello! yazan yuvarlak alana bırak. Daha sonra Picobricks kategorisinden PicoBricks temperature (C) bloğunu sürükleyerek join bloğundaki micro yazan yuvarlağa yerleştir ve blocks yazısını sil	

10	Son olarak tekrar OLED Graphics kategorisinden write bloğunu sürükleyip forever bloğunun içine bırak. Hello! yazısını silerek derece yaz ve x konumunu 40, y konumunu 50 yap. Start tuşuna basarak Picobricks'in dahili sıcaklık sensörünün ölçtüğü ortam sıcaklığını OLED ekranda görüntüleyebilirsin.	

Projenin Microblocks kodlarına erişmek için tıkla.

2.4.4. Proje Görseli



2.4.5. Proje Önerisi

Sende projeni geliştirmek için ortamdaki sıcaklığın 30 derecenin üzerine çıktığında kırmızı LED'in yanmasını ve ekranda uyarı ifadesinin belirmesini sağlayabilirsin.

2.4.6. Micropython Codes of the Project

```
from machine import Pin, I2C, ADC
from ssd1306 import SSD1306_I2C
import utime

WIDTH  = 128
HEIGHT = 64

sda=machine.Pin(4)
scl=machine.Pin(5)
i2c=machine.I2C(0,sda=sda, scl=scl, freq=2000000)
oled = SSD1306_I2C(128, 64, i2c)

pico_temp = ADC(4)
conversion_factor=3.3 / (65536)

while True:
    oled.fill(0)
    oled.show()
    reading =pico_temp.read_u16()*conversion_factor
    temperature= 27 - (reading - 0.706)/0.001721
    oled.text("Temperature:",15,10)
    oled.text(str(int(temperature)),55,30)
    oled.show()
    utime.sleep(0.5)
```

2.4.7. Projenin Arduino C Kodları

```
#include <Wire.h>
#include "ACROBOTIC_SSD1306.h"

int temperature;
int reading;
int conversion_factor=3.3 / (65536);

void setup() {
  pinMode(4,INPUT);
  Serial.begin(115200);
  Wire.begin();
```

```
  oled.init();
  oled.clearDisplay();

#if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
  clock_prescale_set(clock_div_1);
#endif
}

void loop() {
  reading = analogRead(4)*conversion_factor;
  temperature = 27 - (reading - 0.706)/0.001721;
  //String string_temperature = String(temperature);

  oled.setTextXY(3,1);
  oled.putString("Sicaklik: ");
  oled.setTextXY(3,1);
  //oled.putString(string_temperature);
  Serial.println(temperature);
  delay(10);
}
```

2.5. Grafik Monitör

Çevremizdeki elektronik eşyalara baktığımızda değiştirilebilir birçok özelliklerinin olduğunu mühendisler tarafından kullanıcının en çok işine yarayacak şekilde tasarlandıklarını fark edersin. Aydınlatma sistemleri, pişirme sistemleri, ses sistemleri, temizlik sistemleri gibi. bir çok sistem kullanıcısı tarafından çalışma şekli , miktarı, yöntemi vb. özellikleri değiştirilebilir şekilde programlanır.

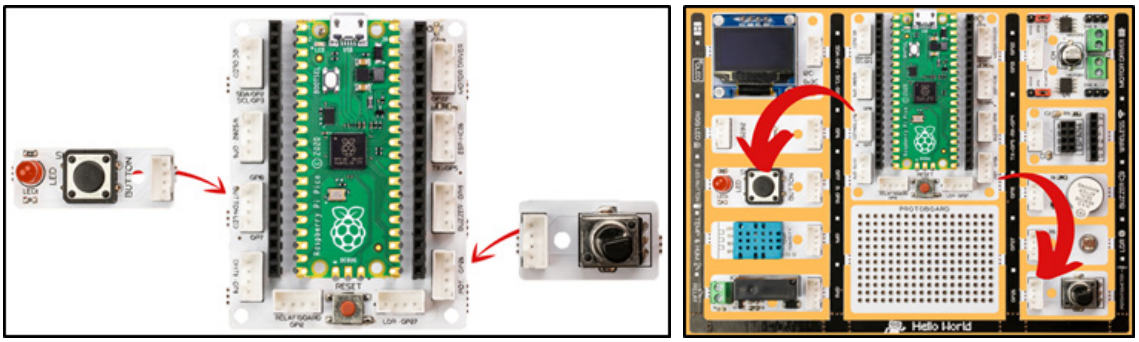
Robotik projelerde ses seviyesini değiştirme, motor hızını değiştirme , ışığın parlaklığını değiştirme işlemlerinde elektrik geriliminin daha düşük veya yüksek etki yaratacak şekilde gönderilmesi sağlanır. Bileşene giden elektrik sinyalinin sıklığı azaltılarak daha düşük seviyede çalışması giden elektrik sinyallerinin sıklığı artırılarak yüksek seviyede çalışması sağlanabilir.

Ekranı olmayan sistemlerde bazı sensörleri ve sistemin çalışmasında görev alan değişkenleri takip etmek için gerçek zamanlı grafik monitörler kullanılır. Arızanın tespit edilmesi için grafik monitörler oldukça kolaylık sağlamaktadır..

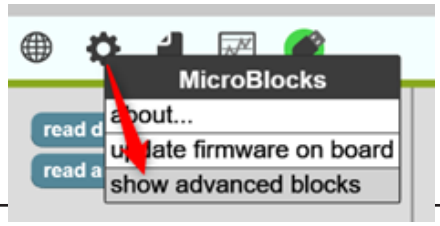
2.5.1. Proje Detayları ve Algoritma

Bu projede potansiyometre ile kırmızı LED'in parlaklığı artırıp azalttığımız bir proje hazırlayacağız. Ayrıca bu işlem sırasında meydana gelen elektriksel değişimi Microblocks grafik monitöründen eş zamanlı olarak takip edeceğiz. Picobricks başladığında potansiyometre değeri sürekli okunarak LED'in parlaklık değerini ayarlanacaktır. Elektrik sinyalinin frekansının değiştirilerek etkisinin azaltıldığı uygulamalara PWM denmektedir. Potansiyometreden okuduğumuz analog değerleri PWM sinyalleri olarak kırmızı LED'e göndereceğiz ve aydınlatma şiddetini ayarlayabileceğiz.

2.5.2. Bağlantı Şeması



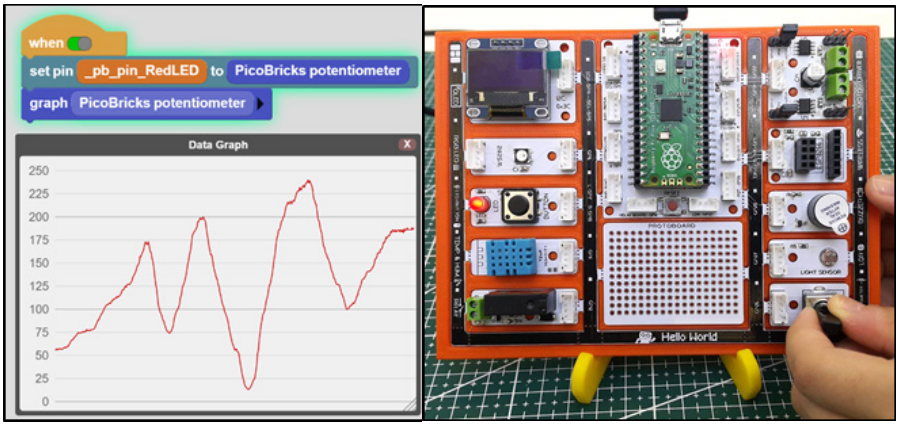
2.5.3. Projenin MicroBlocks ile Kodlanması

1	Picobricks açık olduğu sürece çalışacak kodları hazırlamak için öncelikle control kategorisinden when bloğunu sürükle	
2	LED'in parlaklığını ayarlamak için ona 3.3V luk gerilimi 1024 parçaya bölerek gönderebiliyoruz. Bunu yapabilmek için pins kategorisinden set pin 1 to 1023 bloğunu when bloğunun altına sürükle. 1 yerine Picobricks üzerinde kırmızı LED'in bağlı olduğu GPIO numarasını yazmalısın.	
3	Picobrickste kırmızı LED'in pin nosunu eklemek için ikonuna tıklayın. Açılan menüden "show advanced blocks" u seçin	
4	Variables kategorisinde hazır değişkenleri göreceksiniz. Picobricks'in tüm modüllerinin pin numaraları "_pb_pin_" ön ekiyle bulabilirsiniz. Kırmımzı LED'in bağlı olduğu pin'i ifade etmek için _pb_pin_RedLED bloğunu 1 yazan alana sürükleyip bırakın.	
5	1023 yazan alan Kırmızı LED'e hangi seviyede akımın gönderileceğini ayarladığımız alandır. Bu değeri potansiyometrede alacağımız için Picobricks kategorisinden PicoBricks potantiometer bloğunu buraya yerleştirin.	
6	Potansiyometrenin gönderdiği sayısal veriyi grafik üzerinde görmek için output kategorisinden graph 100 bloğunu sürükle.	

7	graph bloğuna PicoBricks potentiometer bloğunu yerleştirdiğinde grafik monitor de değerlerin görünmesi için gerekli kodu yazıp projeyi tamamlamış oluyoruz. Kodunu çalıştırıp grafik ikonuna  tıklayarak Potansiyometrenin değerinin değişimini izleyebilirsin..	

Projenin Microblocks kodlarına erişmek için tıkla.

2.5.4. Proje Görseli



2.5.5. Proje Önerisi

Sen de potansiyometreyi çevirdikçe buzzer'dan çıkan sesin şiddetini değiştiren ve akan veriyi grafik monitorde gösteren bir proje hazırlayabilirsiniz.

2.5.6. Projenin Micropython Kodları

```
from machine import Pin,ADC,PWM
from utime import sleep

led=PWM(Pin(7))
pot=ADC(Pin(26,Pin.IN))
led.freq(1000)

while True:
    led.duty_u16(int((pot.read_u16()))
```

```
    print(str(int((pot.read_u16()))))
    sleep(0.1)
```

2.5.7. Projenin Arduino C Kodları

```
void setup() {
  pinMode (7,OUTPUT);
  pinMode (26,INPUT);
  Serial.begin(9600);
}

void loop() {

  int pot_val = analogRead(26);
  int led_val = map(pot_val, 0, 1023, 0, 255);
  digitalWrite(7, led_val);
  Serial.println(led_val);
  delay(100);
}
```

2.6. Ritme Hükmet

Hayatımızdaki birçok olgu dijitalleştirilmiştir. Bunlardan biri de seslerdir. Sesin tonu ve şiddeti elektriksel olarak işlenebilmektedir. Yani notaları elektronik olarak çıkarabiliriz. Müziği oluşturan seslerin en küçük birimine nota denir. Her notanın bir frekansı ve şiddeti vardır. Yazacağımız kodlarla frekans ve şiddet uygulayarak hangi notanın çalınmasını ve ne kadar sürmesi gerektiğini ayarlayabiliriz.

Bu projede Picobricks ile bir şarkının melodisini buzzer modülünü kullanarak çalacak, potansiyometre modülü ile ritmi ayarlayabilecek bir müzik sistemi hazırlayacağız. Programlama terminolojisinde önemli bir yere sahip değişken kullanımını da bu projede öğreneceksin.

2.6.1. Proje Detayları ve Algoritma

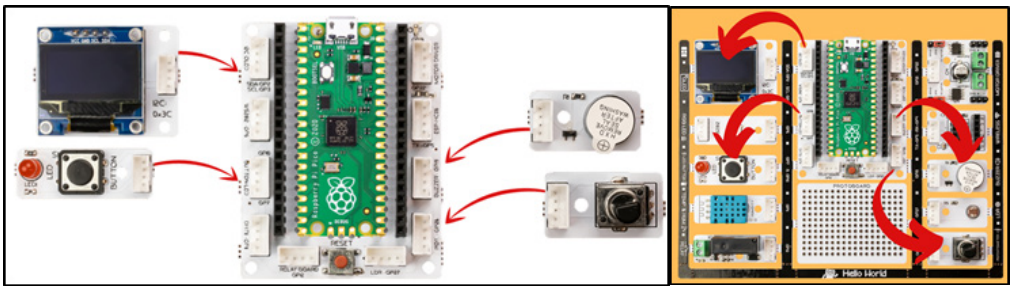
Notalarını bildiğimiz her şarkıyı Picobricks ile çalabilirsin. Şarkıyı başlatmak için buton-LED modülünü, şarkının hızını ayarlayabilmek için potansiyometre modülünü notaları çalabilmek için buzzer modülünü kullanacağız.

Potansiyometre analog giriş modülüdür. Değişken dirençtir. Üzerinden geçen akım miktarı çevrildikçe bir musluğun açılıp kapatılması gibi artar ve azalır. Bu akım miktarını kodlarla kontrol ederek şarkının hızını ayarlayacağız. Buzzer'lar üzerlerinden geçen akımın şiddetine göre ses seviyelerini, gerilim frekansına göre ses tonlarını değiştirmektedirler. Microblocks ile kolayca buzzer modülünden istediğimiz notaları, tonlarını ve sürelerini ayarlayarak kodlayabiliriz.

Projede butona basılma durumunu kontrol edeceğiz. Butona basıldığında melodinin çalmaya başlamasını sağlayacağız. Melodinin çalması sırasında notaların çalınma sürelerini aynı oranda artırıp azaltabilmek için rithm adında bir değişken kullanacağız. Picobricks başladıktan sonra kullanıcının potansiyometre ile ister melodi çalarken ister çalmadan önce rithm değişkenini ayarlayabilmesini sağlayacağız. Picobricks açık olduğu sürece potansiyometre değerini (0-1023) 128'e bölüp rithm değişkene atayacağız. Değişkenler, kullanıcı tarafından ya da sensörler tarafından değiştirilebilecek değerleri kodlarımızda kullanmak istediğimizde kullandığımız veri yapılarıdır. Kullanıcı şarkıyı başlatmak için butona bastığında ise notaların süreleri rithm değişkenine göre hesaplanan süre boyunca çalmasını sağlayacak nota kodlarını hazırlayacağız.

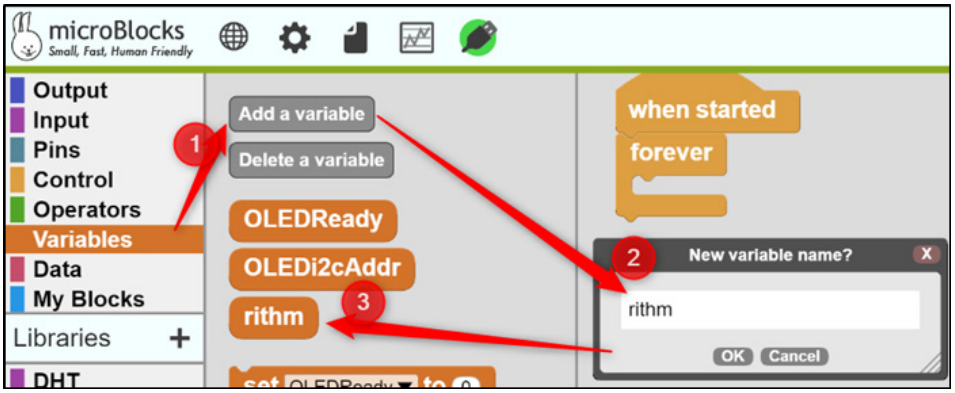
2.6.2. Bağlantı Şeması

2.6.3. Projenin MicroBlocks ile kodlanması



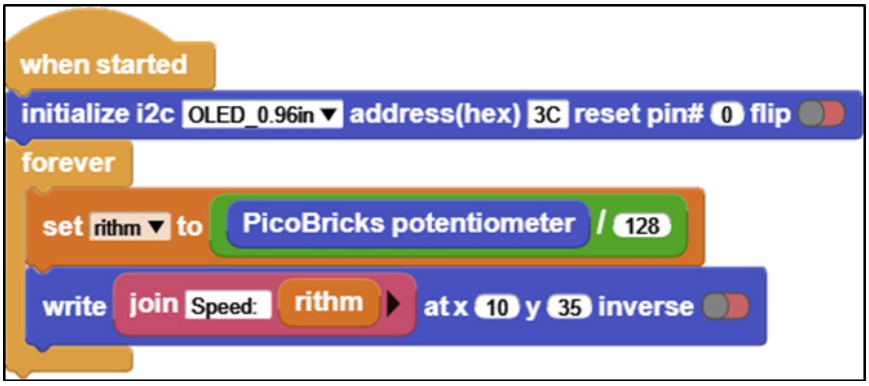
Picobricks'in başlar başlamaz kodları sürekli olarak çalıştırması için Control kategorisinden "When started" , altına "forever" bloğunu sürükle. Kullanıcının hız ayarını ekranda gösterebilmek için OLED ekran kütüphanesini ekle ve ekranı başlatan kodu forever bloğunun öncesine yerleştir. Variables kategorisinde Add a variable butonuna tıkla. Açılan diyalog penceresine rithm yazıp onayladığında rithm değişkeni oluşturulur.

Variables kategorisindeki set OledReady to 0 bloğunun içindeki siyah üçgene tıklayarak değişken adını rithm olarak seç. Ardından forever bloğunun içine yerleştir. Potansiyometreden okunacak değeri 128'e bölerek nota süresinin hesaplanmasında



kullanacağız. Operators kategorisindeki 10/2 bloğunu değişken bloğuna yerleştirip paydayı 128 yap. Potansiyometreyi okuyabilmek için PicoBricks kategorisindeki PicoBricks potentiometer bloğunu bölme işlemi bloğundaki 10 alanına sürükle. OLED kategorisindeki write bloğunu al. Hello alanına Data kategorisinden Join bloğun yerleştirip rithm değişkenini yerleştir.

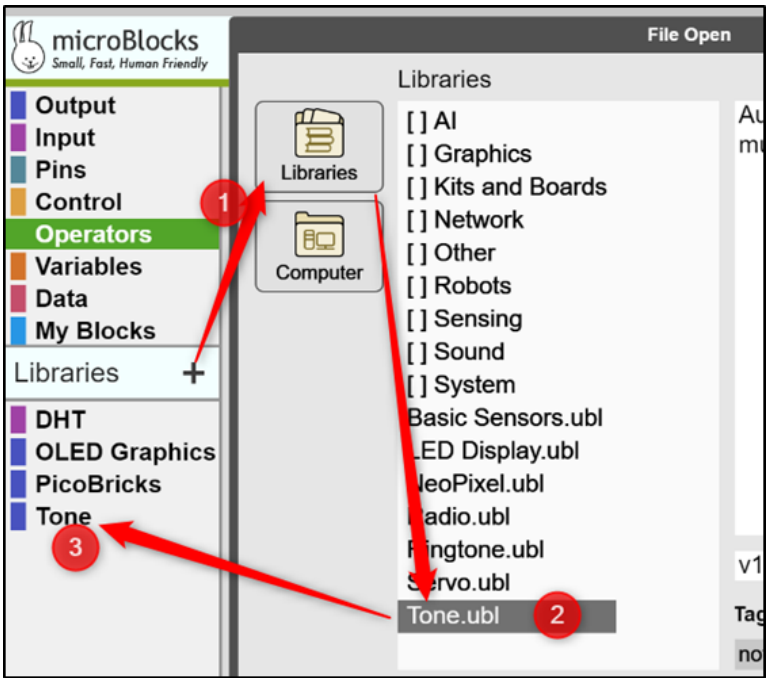
Picobricks üzerindeki butona basılma durumunun sürekli denetlenmesi için Control kategorisinden when bloğunu sürükle. İçine Picobricks button bloğunu yerleştir.



lanına ekle.

when bloğunun altına 

bloğunu ekle. C nota adı, 0 nota



tonunu, 500 ise çalınma süresini ayarlamanızı sağlar. Notaların çalınma sürelerini rithm değişkenine bölerek belirleyeceğiz. play note bloğunun 500 alanına operators kategorisinden 10/2 bloğunu sürükle.

rithm değişkeni en düşük 0 değeri almaktadır. Bir sayının 0'a bölümü sonsuz olacağı

için rithm değerine bir ekleyeceğiz. Bölme işlemindeki payda(2) alanına operators kategorisindeki "10+2" toplam bloğunu sürükle. 10 alanına rithm değişkenini yerleştir ve 2 rakamını 1 olarak değiştir. play note bloğunun üzerinde sağ tıklayıp duplicate ve duplicate all seçenekleriyle şarkındaki nota sayısınca blokları çoğalt.

Notaların adlarını sırasıyla yaz. Her notanın varsayılan vuruş sürelerini de bölme operatörünün 10 alanına yaz 1000 değeri tam vuruş, 500 değeri yarım vuruş 250 değer ise çeyrek vuruş süresini ifade eder. when bloğundan hemen sonra

OLED kategorisinden write bloğunu ekleyerek kullanıcıya Now Playing ifadesini gösteriyoruz. Melodi bittiğinde ise ekranı temizleyeceğiz.

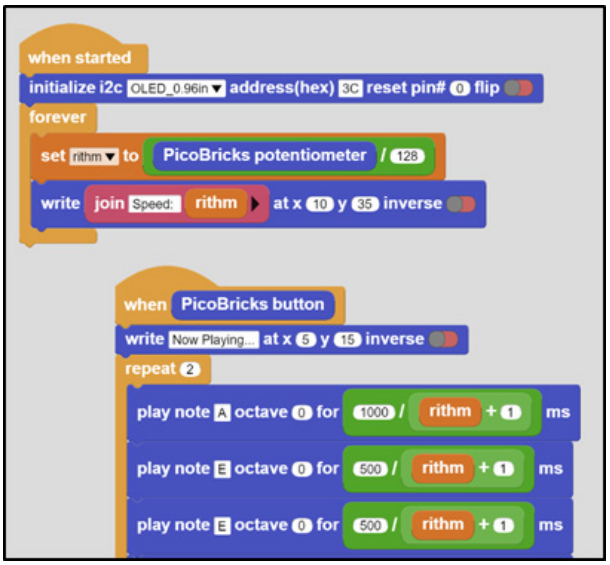
Bu notlar melodinin nakarat kısmıdır. O yüzden iki defa tekrar etmesini ve ardından ekranı temizlemesini kodlamalıyız. Sadece notların olduğu kısmı control

kategorisinden repeat 10 bloğunu alarak içine yerleştir. Tekrar sayısını 2 olarak ayarla. Döngüden sonraya ise OLED kategorisinden clear kodunu ekle.

Kodları çalıştırdığında Picobricks üzerindeki potansiyometre ile melodi hızını belirle. Melodiniz çalarken ya da çalmadan önce potansiyometreyi çevirerek melodinin

hızına hükmet.before playing your melody.

Projenin MicroBlocks kodlarına erişmek için [tıkla.]

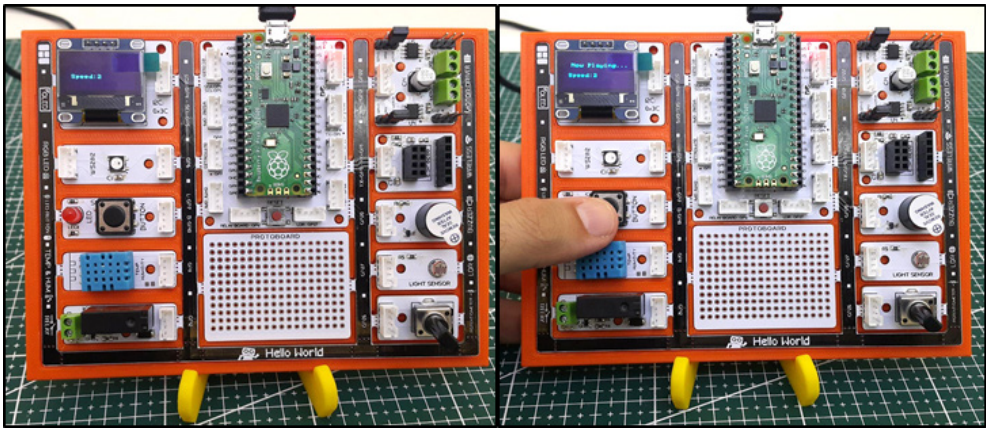


2.6.4. Proje Görseli

2.6.5. Proje Önerisi

Projeni daha görsel hale getirmek için çalınan notaya göre farklı renkte LED yakabilir,

nota isimlerini ve çalma hızını OLED ekranda gösterebilirsin.



2.6.6. Projenin Micropython Kodları

```
from machine import Pin,PWM,ADC
from utime import sleep

button= Pin(10,Pin.IN,Pin.PULL_DOWN)
pot=ADC(Pin(26))
buzzer= PWM(Pin(20))

global rithm
pressed = False

tones = {
“A3”: 220,
“D4”: 294,
“E4”: 330,
“F4”: 349
}

mysong = [“A3”,”E4”,”E4”,”E4”,”E4”,”E4”,”E4”,”F4”,”E4”,”D4”,”F4”,”E4”]
noteTime = [1,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5,1]

def playtone(frequency):
    buzzer.duty_u16(6000)
    buzzer.freq(frequency)

def playsong(pin):
    global pressed

    if not pressed:
```

```
        pressed = True
        for i in range(len(mysong)):
            playtone(tones[mysong[i]])
            sleep(noteTime[i]/rithm+1)
        buzzer.duty_u16(0)

button.irq(trigger=Pin.IRQ_RISING, handler=playsong)
while True:
    rithm= pot.read_u16()
    rithm= int(rithm/6400)+1
```

2.6.7. Projenin Arduino C kodları

```
#include <Wire.h>
#include “ACROBOTIC_SSD1306.h”

int buzzer = 20;
int pot =26;
int button= 10;

int Re = 294;
int Mi = 330;
int Fa = 349;
int La = 440;

void setup()
{
  Wire.begin();
  oled.init();
  oled.clearDisplay();

  pinMode(buzzer,OUTPUT);
  pinMode(26,INPUT);
  pinMode(button,INPUT);

}

void loop()
{
  int rithm = (analogRead(pot))/146;
```

```
  String char_rithm = String(rithm);
  oled.setTextXY(3,4);
  oled.putString("Speed: ");
  oled.setTextXY(3,10);
  oled.putString(char_rithm);

  delay(10);

  if (digitalRead(button) == 1){

    oled.clearDisplay();
    oled.setTextXY(3,2);
    oled.putString("Now playing...");

    tone(buzzer, La); delay (1000/(rithm+1));
    tone(buzzer, Mi); delay (500/(rithm+1));
    tone(buzzer, Mi); delay (500/(rithm+1));
    tone(buzzer, Mi); delay (500/(rithm+1));
    tone(buzzer, Mi); delay (500/(rithm+1));
    tone(buzzer, Mi); delay (500/(rithm+1));
    tone(buzzer, Mi); delay (500/(rithm+1));
    tone(buzzer, Fa); delay (500/(rithm+1));
    tone(buzzer, Mi); delay (500/(rithm+1));
    tone(buzzer, Re); delay (500/(rithm+1));
    tone(buzzer, Fa); delay (500/(rithm+1));
    tone(buzzer, Mi); delay (1000/(rithm+1));

    oled.clearDisplay();
  }
  noTone(buzzer);
}
```

2.7 Tepkini Göster

Şimdi de dikkat ve refleks geliştiren bir oyun hazırlayacağız. Hızlı hareket etmek ve dikkatin uzun süre sağlanabilmesi çocukların önemli gelişimsel özelliklerindendir.

Okul öncesi ve ilkokul dönemindeki çocukların dikkat sürelerini ve reflekslerini artırıcı etkinlikler yapması çocukların hoşlarına gittiği gibi ebeveynlerinin ve öğretmenlerinin de istediği bir durumdur. Hazırlayacağımız elektronik sistemi dikkat artırıcı ve refleks geliştirici bir oyun olacak. Projeyi bitirdikten sonra siz de arkadaşlarınla yarışabilirsin :)

Bu projede her programlama dilinde kullanılan rastgelelik durumunu öğreneceksin. Picobricks ile OLED ekran, Buton-LED ve Buzzer modülünü kullanarak elektronik bir sistem geliştireceğiz.

2.7.1. Proje Detayları ve Algoritma

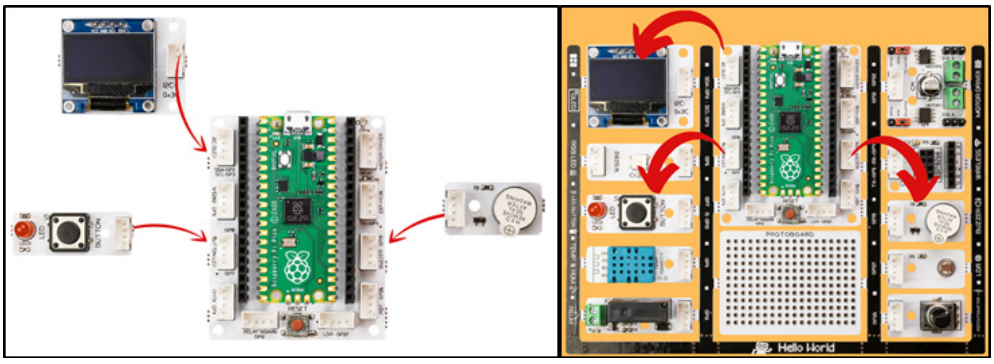
Picobricks açılır açılmaz bir zamanlayıcı çalışmaya başlar. Bu zamanlayıcı ile saniyenin binde 1'ini ölçebiliriz. Saniyenin binde birine milisaniye denir. Zamanlayıcılar günlük yaşamda birçok elektronik sistemde kullanılırlar. Zaman ayarlı aydınlatmalar, fırınlar, ütüler, mutfak robotları...

Projemiz çalışmaya başladığında OLED ekranda karşılama mesajı görüntüleyeceğiz. Ardından kullanıcıya oyuna başlaması için yapması gerekeni ekrana yazdıracağız. Oyuna başlayabilmek için butona basılacak butona basıldıktan sonra 3'ten geriye doğru ekranda sayılarak oyuncunun hazırlanmasını isteyeceğiz. Geri sayımın bitiminden sonra 2-10 saniye arasında rastgele bir süre içinde kırmızı LED yanacak. Kırmızı LED yandıktan sonra hemen zamanlayıcıyı sıfırlayacağız. Tekrar butona basılır basılmaz zamanlayıcıyı ölçeceğiz. Elde ettiğimiz bu değer milisaniye cinsinden olacak. Oyuncunun tepki süresi olarak bu değeri ekranda göstereceğiz.

2.7.2. Bağlantı Şeması

2.7.3. Projenin MicroBlocks ile Kodlanması

OLED Ekran kütüphanesi ekledikten sonra Picobricks başlar başlamaz ekranda yönerge ifadelerine yer verelim.

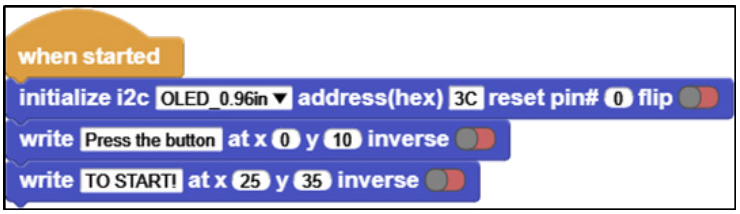


Kullanıcı Picobricks'in butonuna bastığında 3,2,1 diye geri sayması için Control kategorisinden for i in 10 döngüsünü alıyor 10 sayısını 3 olarak ayarlıyoruz. Bu döngü, 1 den 3 e kadar i değişkenini her turda bir artırarak 3 defa içindeki kodu çalıştıracaktır. Her geçen saniye ekranı temizleyip 3...,2...,1... yazdırmak için OLED ekran bloklarını ve

bekleme bloğunu görseldeki gibi for bloğunun içine yerleştirmelisin.

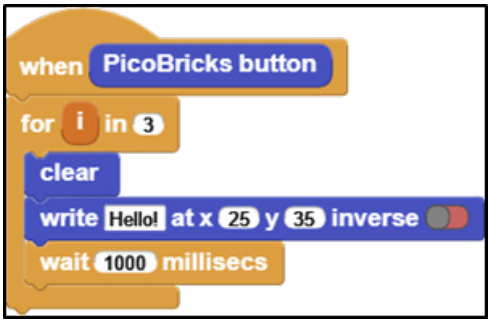
Data kategorisinden bloğunu write 

bloğundaki hello alanına

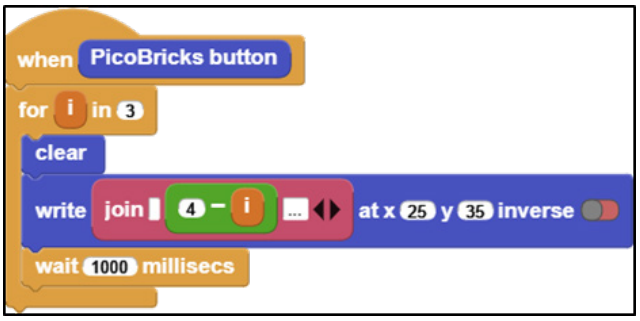


yerleştir, micro yazısını sil, blocks alanına ise çıkarma operatörünü yerleştir. Çıkarma işlemi olarak 4 - i yap.

for döngüsünün hemen altına ekranı temizleyip oyunun başladığını bildiren bloğu ekliyoruz. Rastgele bekleme süresini ayarlamak için bekleme bloğuna random 1 to 10 bloğunu yerleştir. İlk alanına 1000 yaz, ikinci alanına ise 5000 yaz. Bu komut 1 saniye ile 5 saniye aralığında rastgele bir zaman üretecektir. Hemen ardından kırmızı

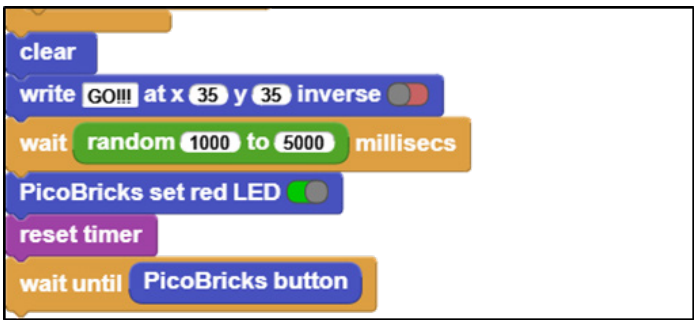


LED'i açacak kodu ekle. Input kategorisinden reset timer bloğu ile zamanlayıcıyı sıfırlıyoruz. Control kategorisindeki wait until bloğuna PicoBricks button bloğunu ekleyerek butona basılana kadar beklenilmesini sağlıyoruz.



Butona basıldığında timer değerini ölçmek için score değişkeni oluştur. wait until bloğunun altına score değişkenine timer değer bloğunu yerleştir. Ardından kırmızı LED'i söndürüp 200 ms lik beep yap. Son olarak ekranı temizleyin. Oyuna yeniden başlamak için Picoboard üzerindeki reset düğmesine basılmasını ifade edip tepki süresini ekrana yazıdırıyoruz. Keyifli oyunlar.

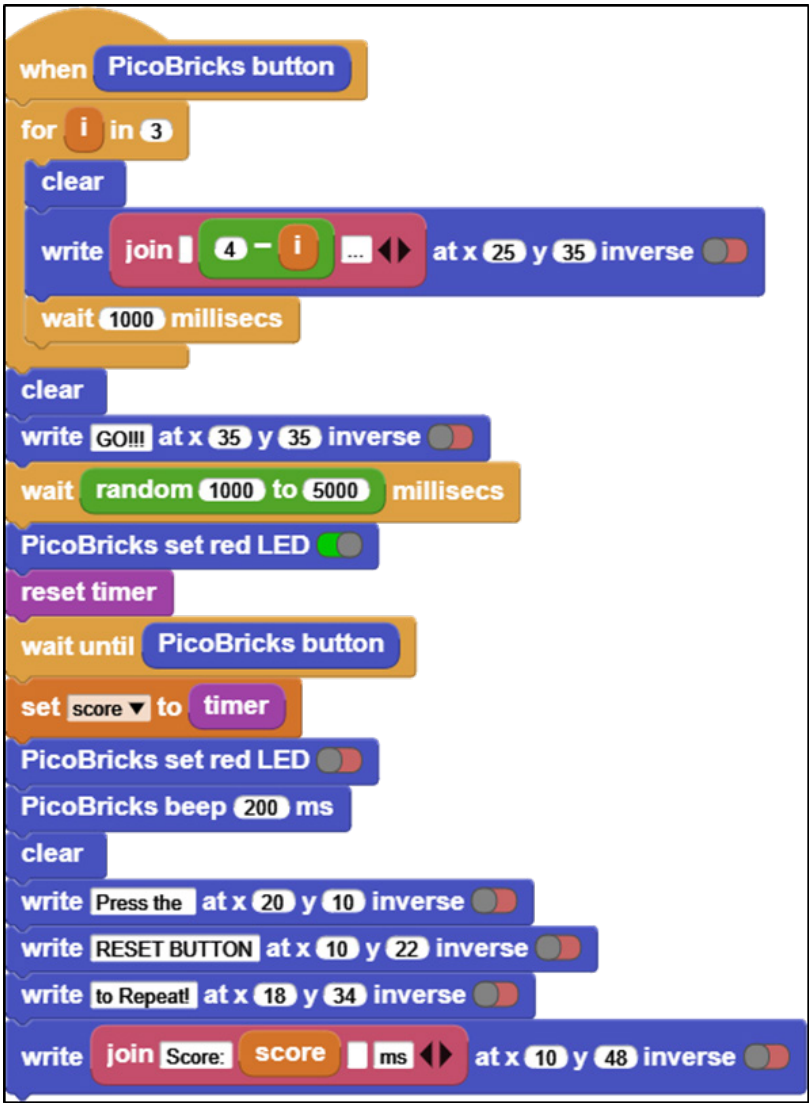
Projenin MicroBlocks kodlarına erişmek için tıkla.



2.7.4. Projenin Görselleri

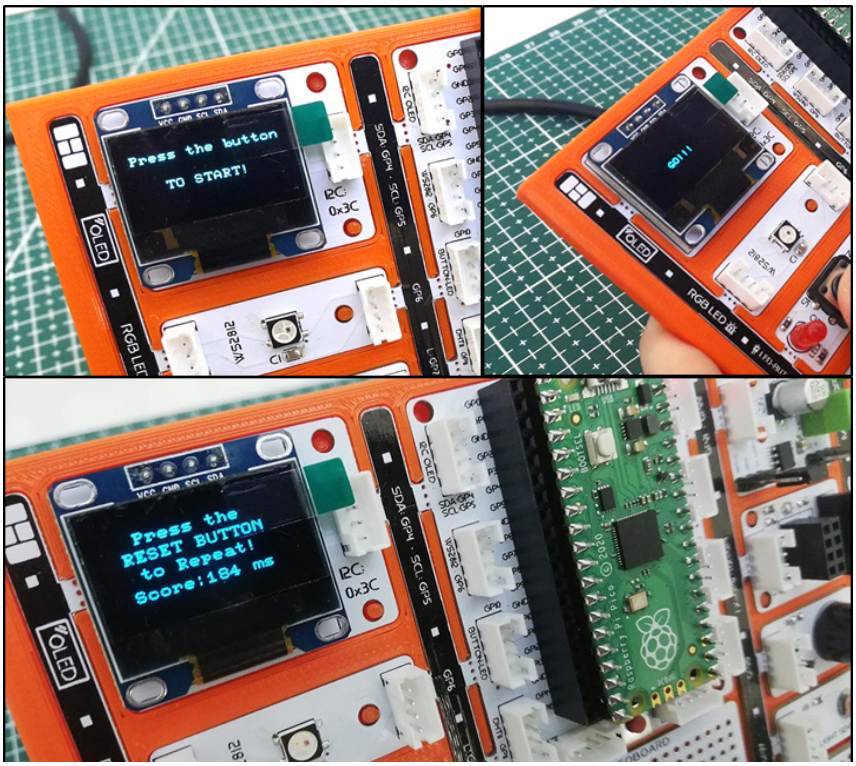
2.7.5. Proje Önerisi

Oyuna yeniden başlayabilmek için Picobricks'in resetlenmesi gerekiyor. Sen de



oyuna yeniden başlamak için butona yeniden basılmasını isteyecek şekilde projeni geliştirebilirsin. Ayrıca oyun sonunda en yüksek score ve son scorer'un da ekrana yazdırılmasını sağlayabilirsin.

2.7.6. Projenin MicroPython Kodları




```
from machine import Pin, I2C, Timer
from ssd1306 import SSD1306_I2C
import utime
import urandom

WIDTH  = 128
HEIGHT = 64

sda=machine.Pin(4)
scl=machine.Pin(5)
i2c=machine.I2C(0,sda=sda, scl=scl, freq=1000000)
oled = SSD1306_I2C(128, 64, i2c)
button = Pin(10,Pin.IN,Pin.PULL_DOWN)
led = Pin(7,Pin.OUT)

pressed = False

oled.fill(0)
oled.show()
oled.text(“Press the button”,0,10)
oled.text(“TO START!”,25,35)
oled.show()
```

```
def button_handler(pin):
    global pressed
    if not pressed:
        timer_reaction= utime.ticks_diff(utime.ticks_ms(), timer_start)
        oled.fill(0)
        oled.show()
        oled.text("Your Time",15,30)
        oled.text(str(timer_reaction), 20, 40)
        oled.show()

led.value(0)
utime.sleep(urandom.uniform(1,5))
led.value(1)
timer_start=utime.ticks_ms()

button.irq(trigger=machine.Pin.IRQ_RISING, handler=button_handler)
```

2.7.7.Projenin Arduino C Kodları

```
#include <Wire.h>
#include "ACROBOTIC_SSD1306.h"

int buzzer = 20;
int button = 10;
int led = 7;
int La = 440;

int old_time = 0;
int now_time = 0;
int score = 0;
String string_score ;

void setup()
{
  Wire.begin();
  oled.init();
  oled.clearDisplay();

  pinMode(led,OUTPUT);
```

```
  pinMode(buzzer,OUTPUT);
  pinMode(button,INPUT);
Serial.begin(9600);
}

void loop()
{
  oled.setTextXY(3,0);
  oled.putString("Press the button");
  oled.setTextXY(5,4);
  oled.putString("TO START");

  if (digitalRead(button) == 1){

    for (int i=3;i>0;i--){

      String string_i = String(i);
      oled.clearDisplay();
      oled.setTextXY(4,8);
      oled.putString(string_i);
      delay(1000);

    }

    oled.clearDisplay();
    oled.setTextXY(4,6);
    oled.putString("GO!!!");

    int random_wait = random(1000, 5000);
    delay(random_wait);

    digitalWrite(led, HIGH);
    old_time=millis();

    while(!(digitalRead(button) == 1)){

      now_time=millis();

      score = now_time-old_time;
      string_score = String(score);
```

```
    }
      digitalWrite(led, HIGH);
      tone(buzzer, La);
      delay (200);
      noTone(buzzer);

      oled.clearDisplay();
      oled.setTextXY(1,4);
      oled.putString("Press the");
      oled.setTextXY(2,3);
      oled.putString("RESET BUTON");
      oled.setTextXY(3,3);
      oled.putString("to Repeat!");

      oled.setTextXY(6,3);
      oled.putString("Score: ");
      oled.setTextXY(6,9);
      oled.putString(string_score);
      oled.setTextXY(6,13);
      oled.putString(" ms");
      Serial.println(score);
      delay(10000);
      oled.clearDisplay();
  }
}
```

2.8. Zamanlayıcı

Zaman ölçmek günlük hayatımızda farkına varmadan yaptığımız basit ama önemli bir iştir. Ameliyattaki cerrah, toplantısına yetişmeye çalışan bir iş insanı, kazanmaya çalışan sporcu, sınavı bitirmeye çalışan bir öğrenci ya da satranç müsabakası... Zaman ölçmek için akıllı kol saatleri, telefonlar hatta profesyonel kronometreler kullanılmaktadır. Elektronik sistemler içinde zaman oldukça doğru kullanılması gereken bir değişkendir. Örneğin bir Çamaşır makinesi; tamburun ne kadar süre saat yönünde ne kadar saat yönü tersine döneceği, deterjanı eritip alabilmek için

kaç sn su akması gerektiği hep zaman ölçerek yapılan görevlerdir. Zamanın önemli olduğu projeler geliştirmek için onu nasıl kullanacağını bilmelisin.

Bu projede Picobircks ile OLED ekran, buton ve potansiyometre modüllerini kullanarak kendi zaman ölçme aygıtını yapacaksın. Bir Timer...

2.8.1. Proje Detayları ve Algoritma

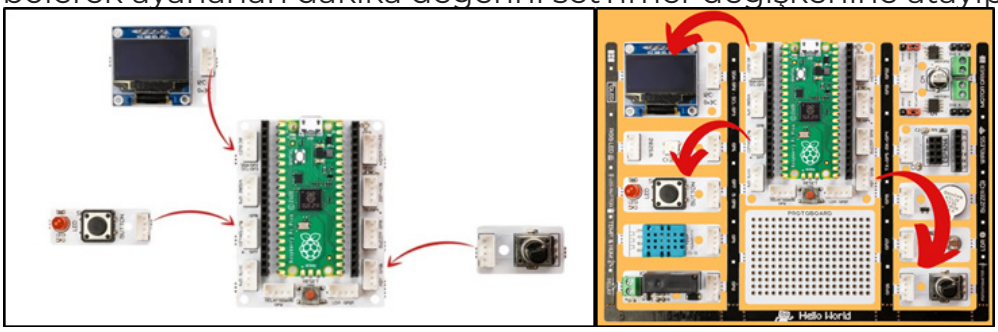
Picobricks başladığında ekrana projeyi tanıtan ve yönerge içeren bir ifade yerleştirelim. Kullanıcı potansiyometreyi çevirdikce 0-60 dakika aralığında bir süre belirleyecek. Kullanıcı potansiyometre ile süreye karar verdikten sonra Picobricks'in butonuna bastığında dakika saniye ve salise ekranda geri doğru saymaya başlayacak. Eğer zaman geriye doğru akarken butona basılırsa Timer duracak ve kalan süreyi ekranda gösterecek. Butona basılmadan dakika, saniye ve salise sıfır değerine ulaşırsa ekrana sürenin dolduğunu ifade eden bildirim gösterilecek ve program durdurulacak.

2.8.2. Bağlantı Şeması

2.8.3. Projenin MicroBlocks ile Kodlanması

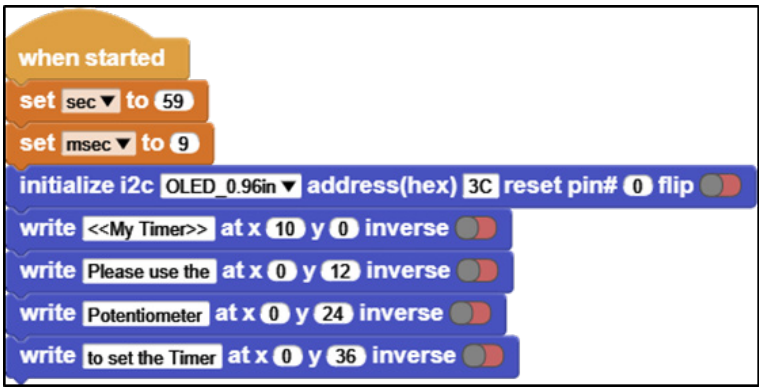
Öncelikle projede kullanılacak değişkenleri oluştur. setTimer, min, sec,msec. sec ve msec değişkenlerinin değerlerini sırasıyla 59 ve 9 olarak ata. Ardından açılış ekranının kodlarını yerleştir.

Butona basılana kadar potansiyometreden okunan değeri 60 ile çarpıp 1023'e bölerek ayarlanan dakika değerini setTimer değişkenine atayıp bu değişkeni ekrana

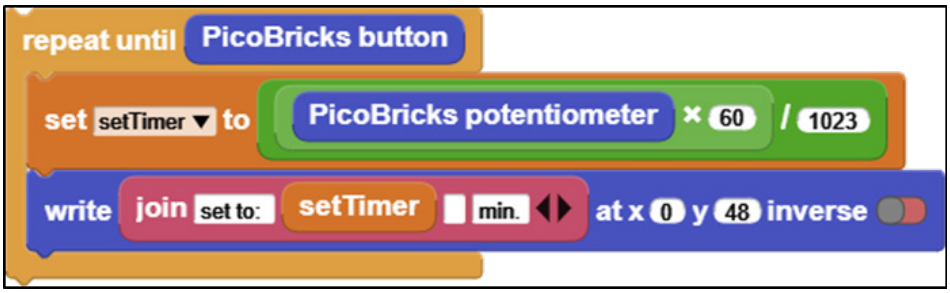


yazdıralım.

repeat until bloğundan sonra butona basıldığında ekranı temizleyip kısa bir süre beklendikten sonra GO! yayınını yapalım.

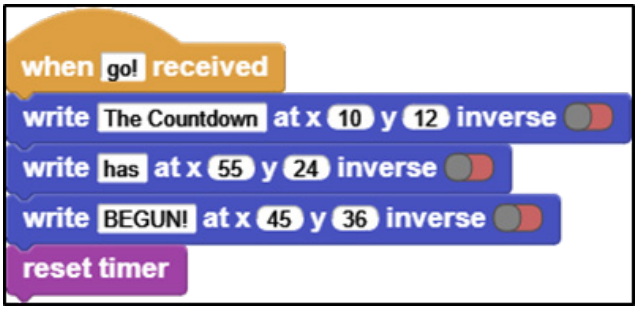


GO! yayını alındığında ekrana geri sayımın başladığını bildiren ifadeyi yerleştirelim. Ardından zamanlayıcıyı sıfırlayalım.

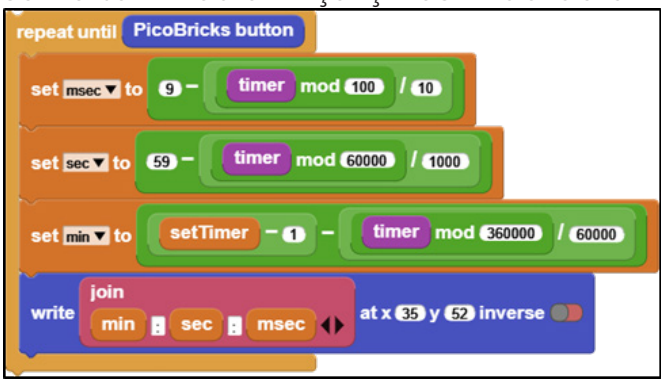


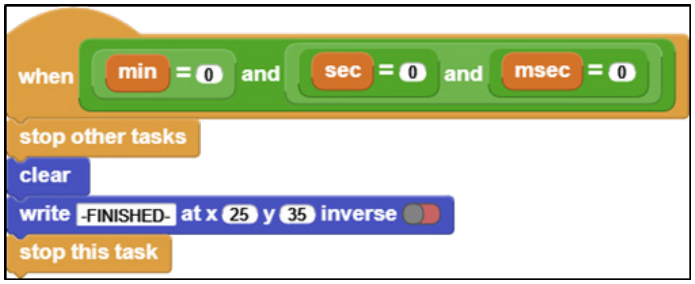
Butona ikinci kez basılana kadar msec, sec ve min değerlerini zamanlayıcıyı ölçerek ayarlayalım. msec milisaniyeyi (0-9) sec saniyeyi(0-59) min(0-59) dakikayı göstermektedir. Ardından OLED ekran üzerinde değişkenlerin gösterilmesini

onlandırılmak isteniyorsa butona basıldığında kalan sürenin ğlayalım.



Son olarak sec, min ve msec değişkenleri 0 olması durumunu sürekli kontrol edelim. Bu değişkenlerin üçü de sıfır olduğunda zaman doldu anlamına gelmektedir. O yüzden diğer kod bloklarının çalışmasını durdurup ekrana bitiş mesajını yazdırdıktan sonra tüm kodların çalışmasını durduralım.





Kodların son hali aşağıdaki gibi olmalıdır. Başla butonuna bastığında projen çalışacaktır.

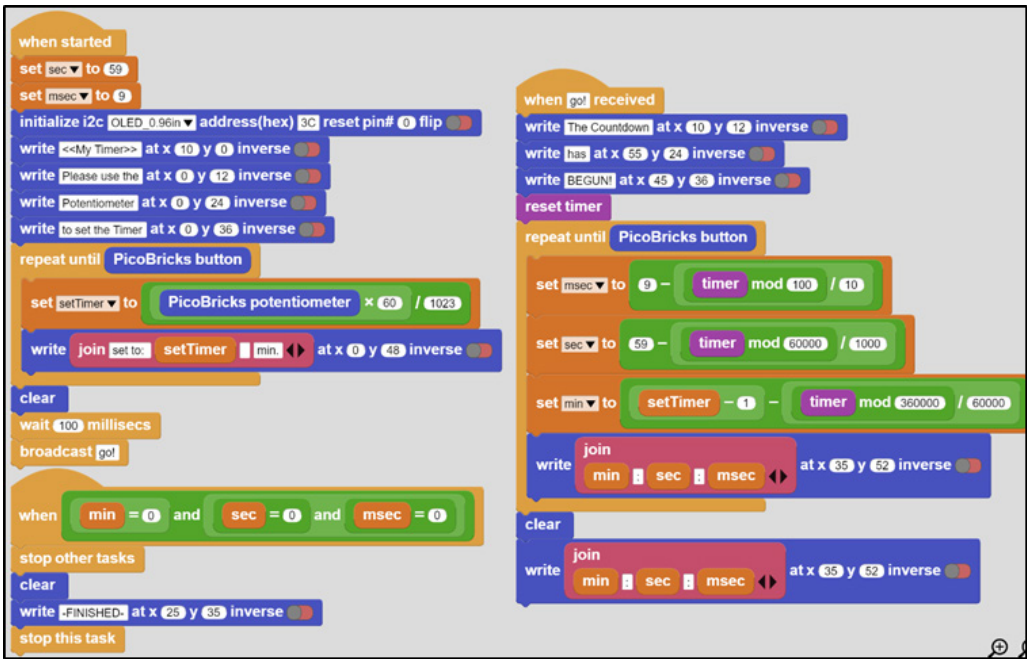
Projenin MicroBlocks kodlarına erişmek için tıkla.

2.8.4. Proje Görselleri

2.8.5. Proje Önerisi

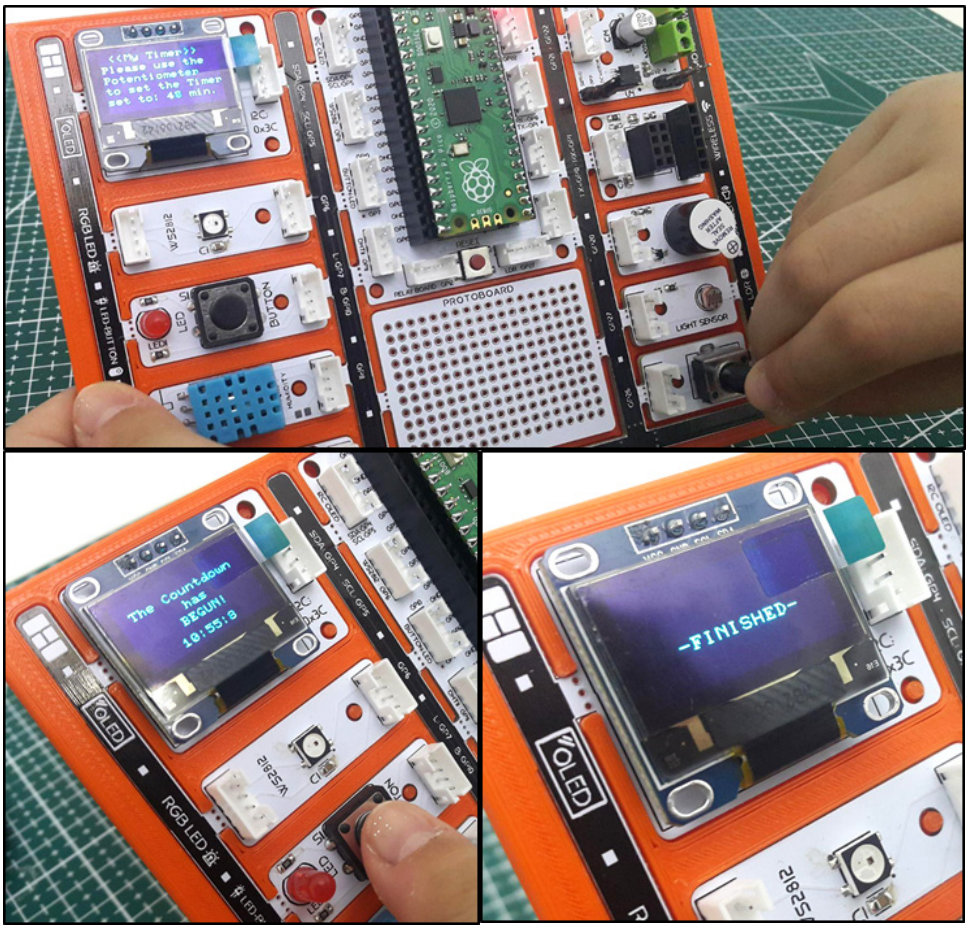
Timer'ın başlangıcına beep sesi ekleyebilirsin. Zaman sıfırlandığında ise buzzer ile

değişik ve yüksek tondan uyarılar vererek sürenin dolduğunu uzaktan duyulacak şekilde bildirebilirsin.



2.8.6. Projenin MicroPython Kodları

```
from machine import Pin, I2C, ADC, Timer
from ssd1306 import SSD1306_I2C
import utime
```



```
WIDTH  = 128
HEIGHT = 64
sda=machine.Pin(4)
scl=machine.Pin(5)
i2c=machine.I2C(0,sda=sda, scl=scl, freq=1000000)
oled = SSD1306_I2C(128, 64, i2c)
pot = ADC(Pin(26))
button = Pin(10,Pin.IN,Pin.PULL_DOWN)

oled.fill(0)
oled.show()

time=Timer()
time2=Timer()
time3=Timer()

def minute(timer):
    global setTimer
    setTimer -=1

def second(timer):
    global sec
    sec-=1
    if sec==-1:
        sec=59

def msecond(timer):
    global msec
    msec-=1
    if msec==-1:
        msec=99

sec=59
msec=99

global setTimer
while button.value()==0:
    setTimer=int((pot.read_u16()*60)/65536)+1
    oled.text("Set timer:" + str(setTimer) + " min",0,12)
    oled.show()
    utime.sleep(0.1)
```

```
    oled.fill(0)
    oled.show()

setTimer-=1

time.init(mode=Timer.PERIODIC,period=60000, callback=minute)
time2.init(mode=Timer.PERIODIC,period=1000, callback=second)
time3.init(mode=Timer.PERIODIC,period=10, callback=msecond)

utime.sleep(0.2)

while button.value()==0:
    oled.text("min:" + str(setTimer),50,10)
    oled.text("sec:" + str(sec),50,20)
    oled.text("ms:" + str(msec),50,30)
    oled.show()
    utime.sleep(0.01)
    oled.fill(0)
    oled.show()
    if(setTimer==0 and sec==0 and msec==99):
        utime.sleep(0.1)
        msec=0
        break;

oled.text(str(setTimer),60,10)
oled.text(str(sec),60,20)
oled.text(str(msec),60,30)
oled.text("Time is Over!",10,48)
oled.show()
```

2.8.7. Projenin Arduino C Kodları

```
#include <Wire.h>
#include "ACROBOTIC_SSD1306.h"

int minute;
int second = 59;
int milisecond = 9;
int setTimer;
```

```
void setup() {

  pinMode(10,INPUT);
  pinMode(26,INPUT);

  Wire.begin();
  oled.init();
  oled.clearDisplay();
}

void loop() {

  oled.setTextXY(1,2);
  oled.putString("<<My Timer>>");
  oled.setTextXY(3,1);
  oled.putString("Please use the");
  oled.setTextXY(4,1);
  oled.putString("Potantiometer");
  oled.setTextXY(5,0);
  oled.putString("to set the Timer");
  delay(3000);
  oled.clearDisplay();

    while(!(digitalRead(10) == 1))
  {
    setTimer = (analogRead(26)*60)/1023;
    oled.setTextXY(3,1);
    oled.putString("set to:");
    oled.setTextXY(3,8);
    oled.putString(String(setTimer));
    oled.setTextXY(3,11);
    oled.putString("min.");
  }
    oled.clearDisplay();
    oled.setTextXY(1,1);
    oled.putString("The Countdown");
    oled.setTextXY(2,3);
    oled.putString("has begin!");

    while(!(digitalRead(10) == 1))
```

```
{
  milisecond = 9- (millis()%100)/10;
  second = 59-(millis()%60000)/1000;
  minute = (setTimer-1)-((millis()%360000)/60000);

  oled.setTextXY(5,3);
  oled.putString(String(minute));
  oled.setTextXY(5,8);
  oled.putString(String(second));
  oled.setTextXY(5,13);
  oled.putString(String(milisecond));
  oled.setTextXY(5,6);
  oled.putString(":");
  oled.setTextXY(5,11);
  oled.putString(":");
}
  oled.setTextXY(5,3);
  oled.putString(String(minute));
  oled.setTextXY(5,8);
  oled.putString(String(second));
  oled.setTextXY(5,13);
  oled.putString(String(milisecond));
  oled.setTextXY(5,6);
  oled.putString(":");
  oled.setTextXY(5,11);
  oled.putString(":");
  delay(10000);

  if (minute==0 & second==0 & milisecond==0){

  oled.setTextXY(5,3);
  oled.putString(String(minute));
  oled.setTextXY(5,8);
  oled.putString(String(second));
  oled.setTextXY(5,13);
  oled.putString(String(milisecond));
  oled.setTextXY(5,6);
  oled.putString(":");
  oled.setTextXY(5,11);
  oled.putString(":");
  oled.putString("-finished-");
```

```
    oled.setTextXY(7,5);
    delay(10000);
    }
}
```

2.9. Çalar Saat

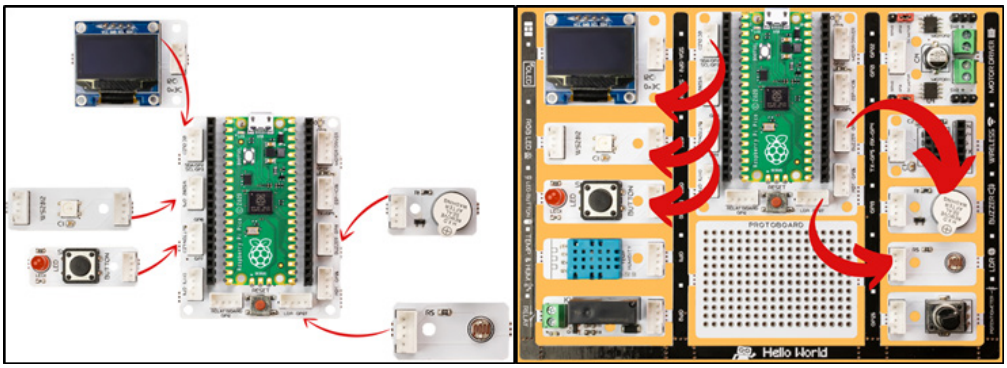
Küresel ısınma dünyamızın iklimini her geçen gün daha kötü etkiliyor. Ülkeler küresel ısınmanın etkilerini azaltmak için birçok tedbiri devreye sokuyor ve anlaşmalar imzalıyorlar. Yenilenebilir enerji kaynaklarının kullanılması ve enerjinin verimli kullanılması fabrikalardan evimizin odalarına kadar her yerde dikkat edilmesi gereken bir konudur. Şehirlerde yol ve park aydınlatmalarının insan hatasından dolayı açık kalması, yüksek enerji tüketen aydınlatma araçlarının kullanılması gibi birçok sebep enerji verimliliğini düşürmektedir. Ortamın ışık, sıcaklık ve nem değerlerini ölçerek sadece gerek duyulduğunda ve doğru miktarlarda kullanılmasını sağlayan birçok elektronik ve dijital sistem mühendisler tarafından geliştirilmekte ve programlanmaktadır.

Bu projede Picobricks'teki ışık sensörünü kullanarak gün ışığına göre ayarlanan bir saat alarmı hazırlayacağız.

2.9.1. Proje Detayları ve Algoritma

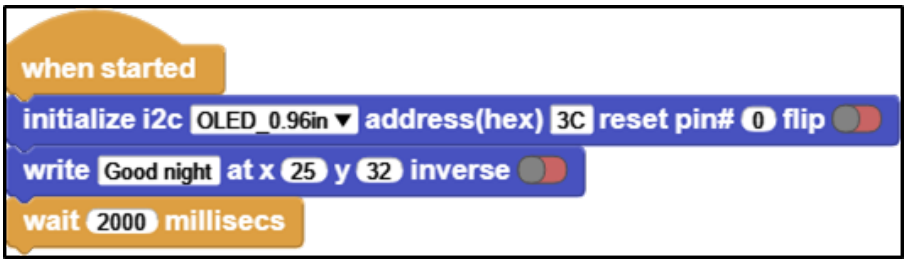
Bu projede basit bir alarm uygulaması yapacağız. Tasarlayacağımız alarm sistemi sabah olduğunda otomatik olarak çalacak şekilde kurgulanmıştır. Bunun için projede LDR sensör kullanacağız. Gece olduğunda OLED ekranda kullanıcıya iyi geceler mesajı görüntülenecek, sabah olduğunda ise buzzer sesi ile alarm çalacak, ekranda kullanıcıya günaydın mesajı gösterilecek ve ışıklı bildirim amacıyla RGB LED beyaz renkte yanacak. Kullanıcının alarmı durdurması için ise Picobricks'in butonuna basması gerekecek. Alarm durdurulana kadar devam eden bu işlemlerden sonra butona basıldığında buzzer ve RGB LED kapanacak ve OLED ekranda kullanıcıya iyi günler mesajı gösterilecek.

2.9.2. Bağlantı Şeması



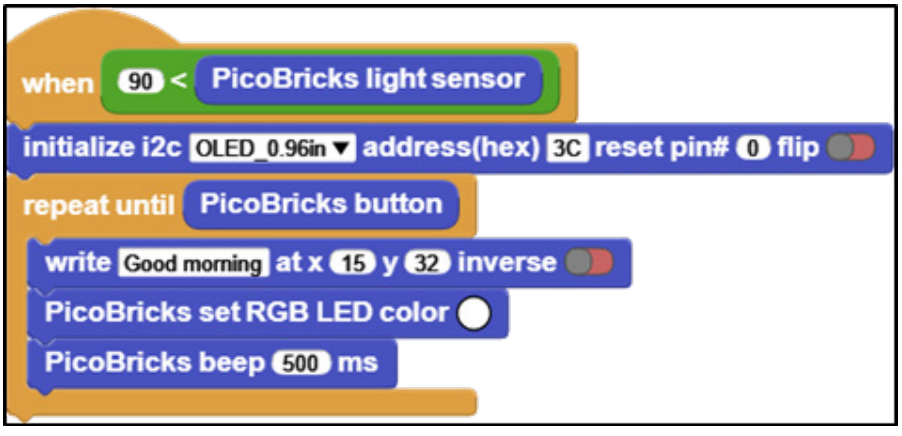
2.9.3. Projenin MicroBlocks ile Kodlanması

İlk olarak programa OLED Graphics kütüphanesini ekle ve Pico başladığında ekranda “Good night” yazısının yazması için when started bloğunun altına OLED kategorisinden initialize i2c ve write bloklarını ekle. Eğer gündüz bu projeyi yapıyorsan programı test etmek için LDR sensörü elinle kapatman gerekecek. Bu sebeple write bloğundan sonra wait bloğu ile programı 2 saniye bekletin ve o 2 saniye içerisinde elinle LDR sensörü kapat.

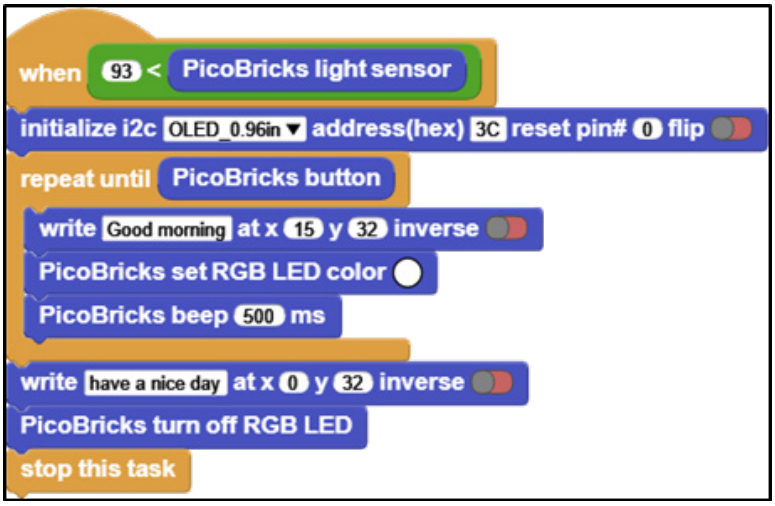


Otonom aydınlatma projesinde LDR sensör değerlerini say bloğu ile okumuş ve ortam aydınlıkken 90 ve üzeri, karanlıkken 90’ın altında değerler görmüştük. Bu projede de gece ve gündüz farkını belirlemek için 90 değerini kullanabiliriz. say bloğunu kullanarak bulunduğunuz ortamdaki LDR sensör değerini görüntüleyip alarmın çaldığı değeri değiştirebilirsin.

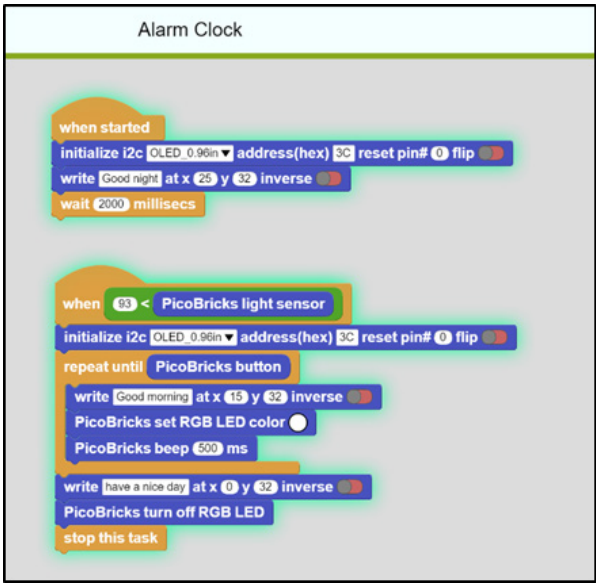
LDR sensörden gelen değer 90’dan büyük olduğunda alarmın devreye girmesi için when bloğu kullanmalı ve koşulu belirtmeliyiz. when bloğunun altına ise kullanıcı butona basana kadar alarmın çalmaya devam edebilmesi için repeat until bloğunu kullanmalı ve Picobricks’in butonuna basılması bloğunu koşul olarak belirtmeliyiz. Belirtilen koşul sağlanana kadar içindeki blokları çalıştıran repeat until bloğunun içine ise RGB LED, buzzer ve OLED ekran kodlarını yerleştirmelisin. Ekran yazılarını, RGB LED rengini ve buzzerın çalma süresini değiştirebilirsin.



Buraya kadar yazılan kodlar, sistem başladığında ve sabah olduğunda yapılacak işlemleri yerine getiren kodlardır. Şimdi alarm çalarken butona basıldıktan sonra yapılacak işlemler için gerekli kodları yazalım.



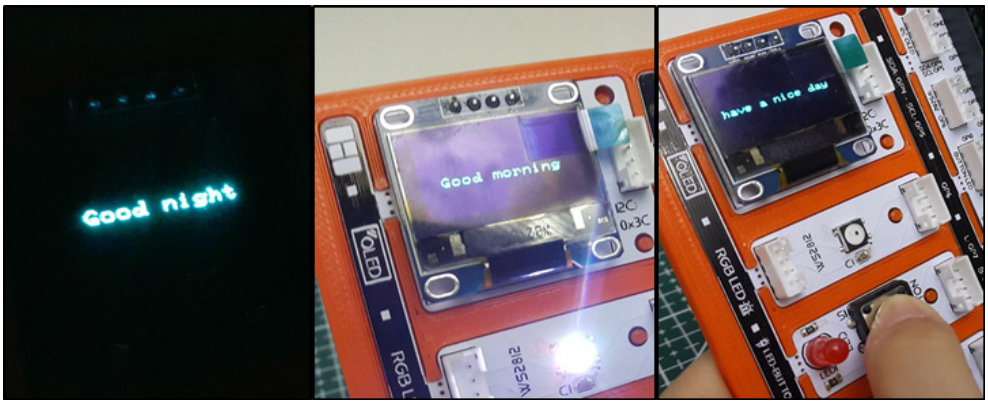
Alarm çalarken butona basıldığında program repeat until bloğunun dışına çıkacak ve bir alt satırdaki kodları çalıştıracaktır. Bu sebeple RGB LED'in kapatılması ve ekran yazısı bloklarını repeat until bloğunun altına yazmalısın.



Kodların tamamını yazarak Start tuşuna bastığında ekranda "Good night" yazısı görüntülenecek ve eğer ortam karanlıksa aydınlık olana kadar program bekleyecektir. Eğer aydınlık bir ortamda projeyi test ediyorsan Start tuşuna bastıktan sonra 2 saniye içinde LDR sensörü elinle kapat. Daha sonra ortam aydınlandığında OLED ekran, buzzer ve RGB LED ile alarmın çalıştığını göreceksin. Alarmı butona basarak durdurabilirsin.

Projenin MicroBlocks kodlarına erişmek için tıkla.

2.9.4. Proje Görseli



2.9.5. Proje Önerisi

Projeye alarm sesi olarak bip sesi yerine melodi ekleyerek geliştirebilirsin. Ya da LDR sensör ile gün ışığına göre ayarlanan alarm yerine buton ve OLED ekran üzerinden saat bilgisinin düzenlendiği ve belirtilen saatte çalan alarm geliştirebilirsin.

2.9.6. Projenin Micropython Kodları

```
from machine import Pin, I2C, ADC, PWM
from ssd1306 import SSD1306_I2C
import utime
from ws2812 import NeoPixel

WIDTH  = 128
HEIGHT = 64
sda=machine.Pin(4)
scl=machine.Pin(5)
i2c=machine.I2C(0,sda=sda, scl=scl, freq=1000000)
neo = NeoPixel(6, n=1, brightness=0.3, autowrite=False)

oled = SSD1306_I2C(128, 64, i2c)
ldr = ADC(Pin(27))
button = Pin(10,Pin.IN,Pin.PULL_DOWN)
buzzer = PWM(Pin(20, Pin.OUT))
buzzer.freq(1000)
BLACK = (0, 0, 0)
WHITE = (255, 255, 255)

oled.fill(0)
oled.show()
```

```
neo.fill(BLACK)
neo.show()

if ldr.read_u16()<4000:
    wakeup = True
else:
    wakeup = False

while True:
    while wakeup==False:
        oled.fill(0)
        oled.show()
        oled.text("Good night",25,32)
        oled.show()
        utime.sleep(1)
        if ldr.read_u16()<4000:
            while button.value()==0:
                oled.fill(0)
                oled.show()
                oled.text("Good morning",15,32)
                oled.show()
                neo.fill(WHITE)
                neo.show()
                buzzer.duty_u16(6000)
                utime.sleep(1)
                buzzer.duty_u16(0)
                utime.sleep(0.5)
                wakeup=True
            neo.fill(BLACK)
            neo.show()
    oled.fill(0)
    oled.show()
    oled.text("Have a nice day!",0,32)
    oled.show()
    if ldr.read_u16()>40000:
        wakeup= False

    utime.sleep(1)
```

2.9.7. Projenin Arduino C Kodları

```
#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
#include <avr/power.h>
#endif
#define PIN       6

#define NUMPIXELS 1
Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);
#include <Wire.h>
#include “ACROBOTIC_SSD1306.h”
int button;
void setup() {
  Wire.begin();
  oled.init();
  oled.clearDisplay();

#if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
  clock_prescale_set(clock_div_1);
#endif
  pinMode(10,INPUT);
  pinMode(27,INPUT);
  pinMode(20,OUTPUT);

  pixels.begin();
  pixels.setPixelColor(0, pixels.Color(0, 0, 0));
  pixels.show();
}

void loop() {

    oled.setTextXY(4,3);
    oled.putString(“Good night”);

    if (analogRead(27)<200){

      while(!(button == 1)){

        button=digitalRead(10);

        oled.setTextXY(4,2);
```

```
        oled.putString("Good morning");
        pixels.setPixelColor(0, pixels.Color(255, 255, 255));
        pixels.show();
        tone(20,494);
      }
        oled.clearDisplay();
        oled.setTextXY(4,1);
        oled.putString("Have a nice day");
        noTone(20);
        pixels.setPixelColor(0, pixels.Color(0, 0, 0));
        pixels.show();
        delay(10000);
    }
}
```

2.10. Rengini Bil

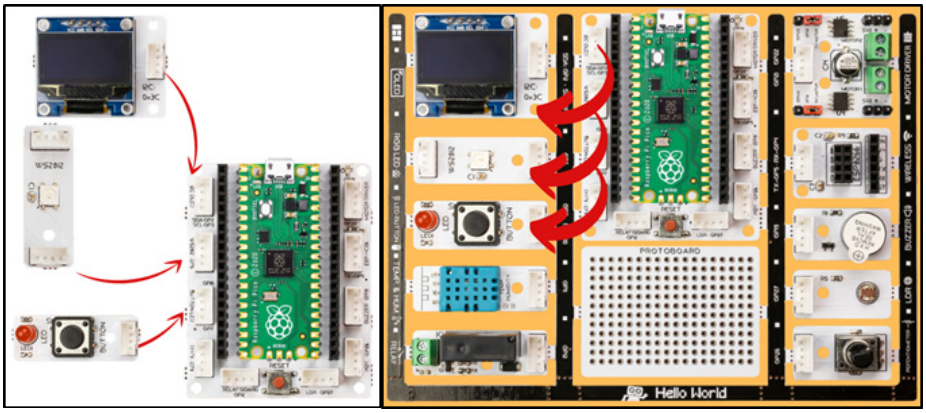
Elektronik sistemler üzerinde LED'ler sıklıkla kullanılır. Her butonun her seçeneğin yanında küçük LED'ler bulunabilmektedir. Tek bir LED'i değişik renklerde yanmasını sağlayarak birden fazla LED'in yaptığı işi tek bir LED ile yapılması sağlanabilmektedir. Bu türde çalışan LED'lere RGB LED denir. Adını Red, Green, Blue renk isimlerinin baş harflerinden alır. Bu LED'in diğer avantajı da 3 ana rengin karışımlarında da yanabilmesidir. Mor, turkuaz, turuncu...

Bu projede her programlama dilinde kullanılan rastgelelik durumunu öğreneceksin. Picobricks'in RGB LED, OLED ekran ve buton modülü ile eğlenceli bir oyun hazırlayacağız.

2.10.1. Proje Detayları ve Algoritma

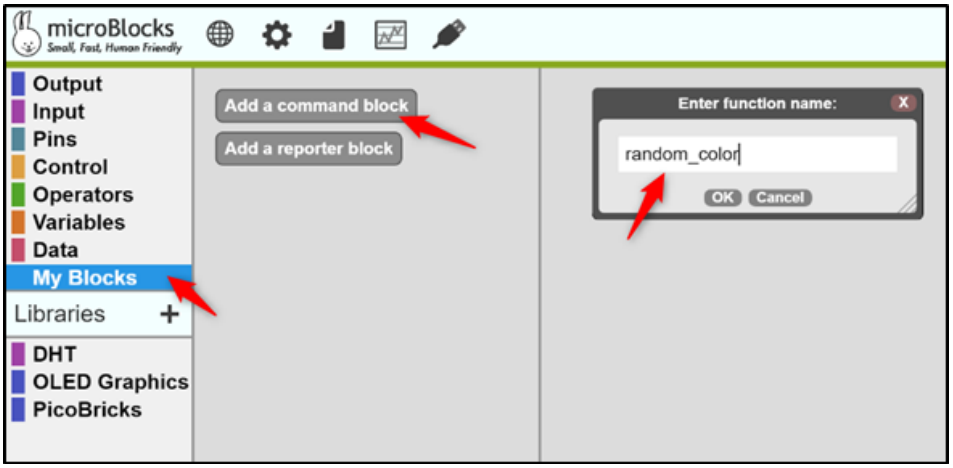
Projede inşa edeceğimiz oyun kullanıcının renkleri doğru veya yanlış bilmesi üzerine kurgulanacaktır. Picobricks üzerindeki RGB LED'de kırmızı, yeşil, mavi ve beyaz renklerden birisi rastgele olarak yanacak, aynı anda OLED ekranda yine bu dört renkten birisinin adı rastgele olarak yazılacaktır. Kullanıcı cevap hakkını kullanmak için 1,5 saniye içerisinde Picobricks'in butonuna basmalıdır. Oyun 10 kere tekrarlanacak, her tekrarda renkler eşlediğinde kullanıcı butona basarsa ya da eşleşmediğinde kullanıcı butona basmazsa 10 puan alacaktır. Renkler eşleşmediği halde kullanıcı butona basarsa 10 puan kaybedecektir. On tekrar sonunda kullanıcının aldığı puan OLED ekranda gösterilecektir. Kullanıcı dilerse butona basmayarak cevap hakkını kullanmayabilir.

2.10.2. Bağlantı Şeması



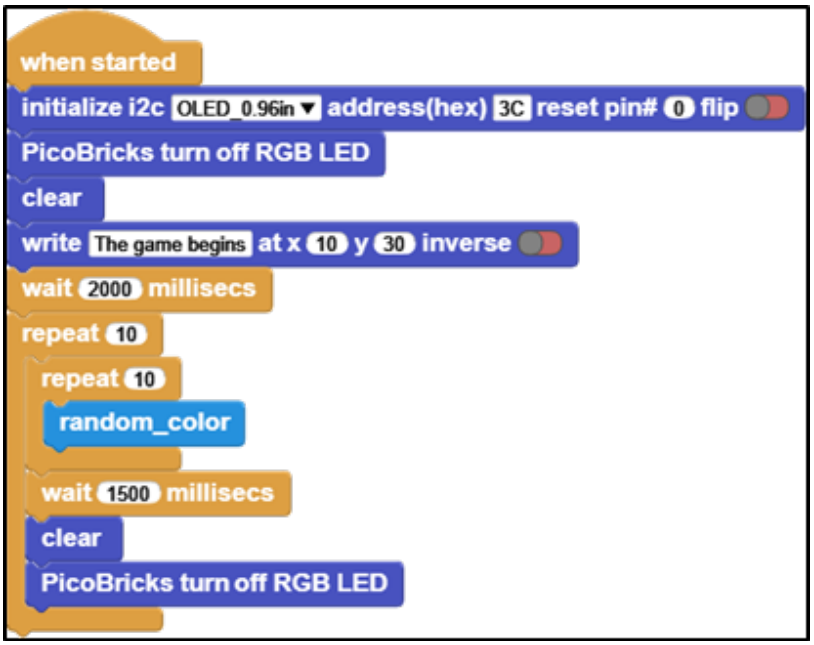
2.10.3. Projenin MicroBlocks ile Kodlanması

Projede OLED ekran kullanacağımız için öncelikle OLED Graphics kütüphanesini programa eklemelisiniz. Daha sonra OLED ekranda ve RGB LED'de rastgele renkler oluşturmak için OLED_color ve RGB_color adında iki adet değişken oluşturmalısınız. Programın daha stabil çalışması için renklerin rastgele belirlenmesi işlemini bir fonksiyon oluşturup bu fonksiyonu çağırarak gerçekleştirebiliriz. Fonksiyonlar bir veya daha fazla işlem satırından oluşan kodların bir kod bloğu şeklinde yapılandırılması ile oluşturulur. Fonksiyonlar oluşturulduktan sonra programın herhangi bir yerinde sadece fonksiyon adı kullanılarak çağrılabilir. Fonksiyon oluşturmak için kod kategorileri bölümünde My Blocks butonuna tıklamalı, sonra Add a command block butonuna tıklamalı ve açılan Enter function name penceresinde oluşturacağınız fonksiyona isim vermelisin. Biz bu projede oluşturacağımız fonksiyona random_color adını verebiliriz.

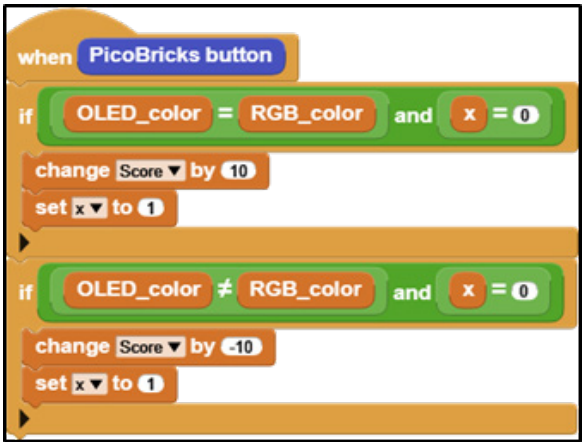


Fonksiyon oluşturduktan sonra kod yazma alanına define random_color bloğu gelecektir. Bu bloğun altına fonksiyon kodlarını yazabilirsin. random_color fonksiyonunda ilk olarak daha önce oluşturduğumuz OLED_color ve RGB_color değişkenlerine 1 ile 4 arasında rastgele sayı atıyoruz. Bunun için operators kategorisindeki random bloğunu kullanmalısın. Daha sonra değişkenlere atanan rastgele değerleri if bloğu ile karşılaştırarak RGB LED rengi ve OLED ekran yazılarını düzenlemelisin.

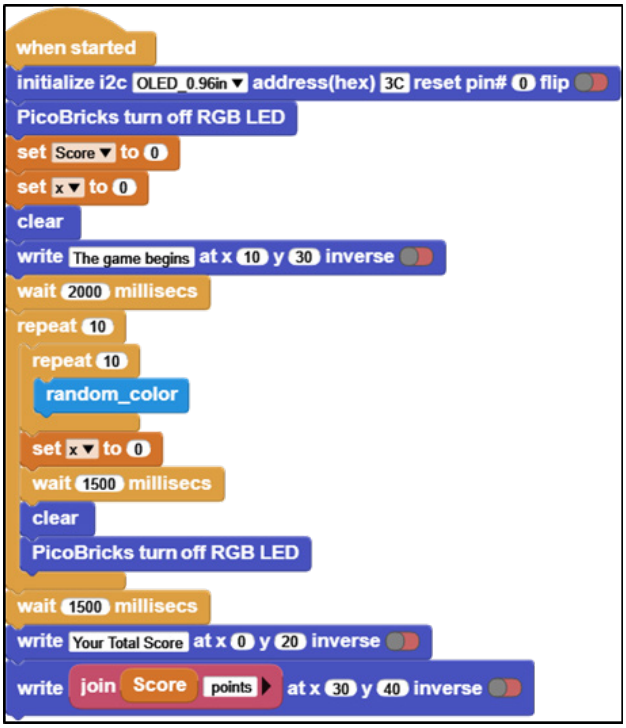
random_color fonksiyonunu tanımladıktan sonra programımızı when started bloğunun altına yazabiliriz. Bu kod grubunda önce program başladığında çalışması gereken kodları yazmalısın. Başlangıç bloklarından sonra repeat 10 bloğunun içinde önceden oluşturduğun random_color fonksiyonunu çağırmalısın. 10 kere random_color çalıştıktan sonra programı bir süre bekletmeli, ekranı temizlemeli ve RGB LED'i kapatmalısın. Tüm bu işlemleri 10 kere tekrarlamalısın. Buraya kadar olan kodları çalıştırdığınızda RGB LED ve OLED ekranda 1,5 saniye aralıklarla 10 kere rastgele rengin belirdiğini göreceksin.

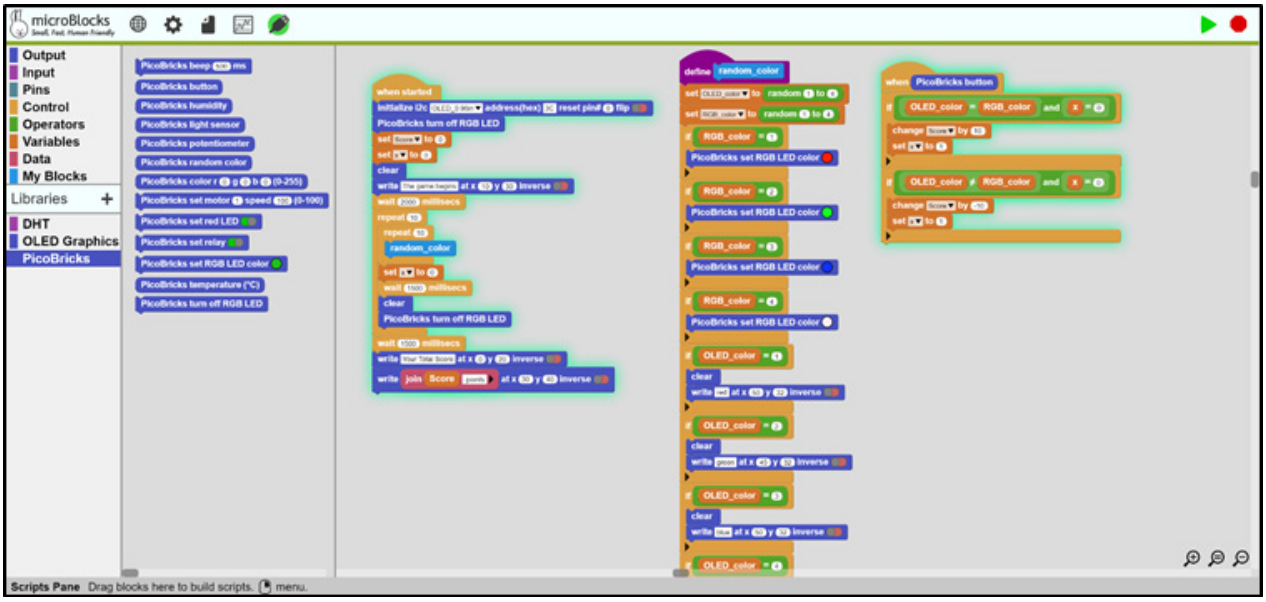


Bu aşamada OLED ekran ve RGB LED'deki renkler aynı olduğunda kullanıcının butona basması ve puan kazanması için Score ve x adında iki adet değişken daha oluştur. when bloğuna koşul olarak Picobricks'in butonuna basılması bloğunu ekle. Daha sonra iki tane if bloğu ile renk değişkenlerinin eşit olup olmadığını karşılaştır. Renkler eşitse skor değişkenini 10 arttır eşit değilse 10 azalt. Burada x değişkenini kullanma sebebimiz, sadece 1 kere puan artışı yapabilmek. and operatörünü içerisinde ikinci koşul olarak x değişkenini kullanmazsanız butona basıldığı süre boyunca Score değişkeni anlık olarak 10'ar 10'ar artacaktır.



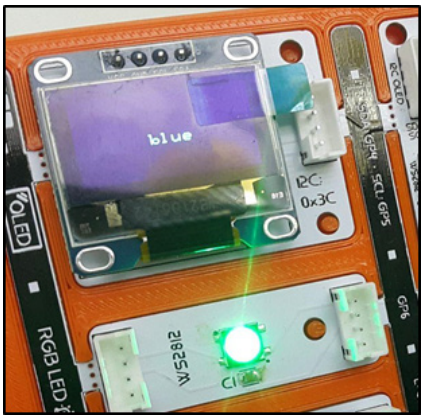
Butona basıldığında programın yapması gereken işlemleri düzenledikten sonra oyun bittiğinde ekranda kullanıcının skorunun yazılması için gerekli kodları ana programa eklemelisin. Score ve x değişkenlerinin başlangıç değerlerini “0” olarak belirt ve 10 tekrar içinde her tekrarda x değişkeninin değerini “0” olarak atayın. Bu sayede oyun içerisinde butona her basıldığında tek seferlik puan artışı gerçekleşecektir.

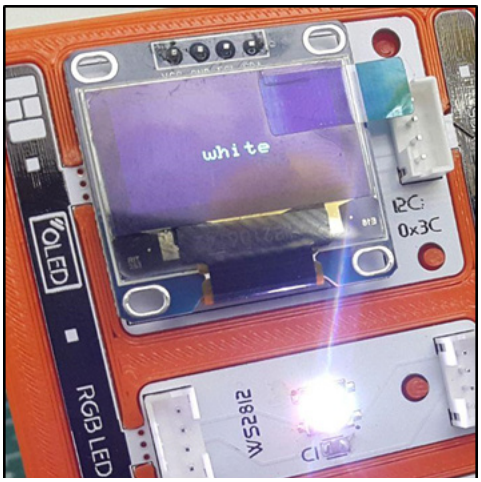


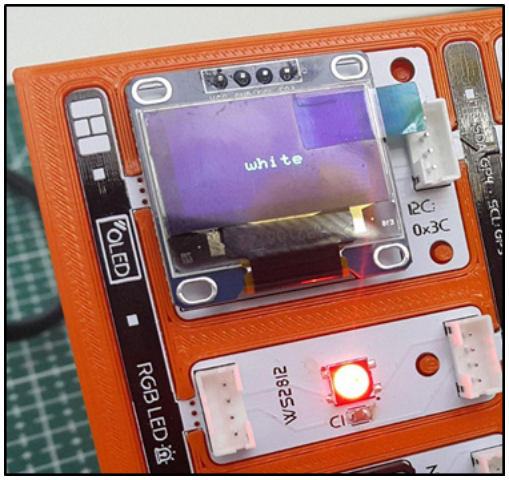


Projenin MicroBlocks kodlarına erişmek için [tıkla.]

2.10.4. Proje Görseli







2.10.5. Proje Önerisi

Oyunu biraz daha zorlaştırarak eğlenceli hale getirebilirsin. Örneğin renklerin tekrarlanma süresini kısaltarak oyunu hızlandırabilirsin. Ya da kullanıcı yanlış yerde butona bastığında puan kaybetmesi yerine oyunu bitirip tekrar başlatabilirsin.

2.10.6. Projenin MicroPython Kodları

```
from machine import Pin, I2C
from ssd1306 import SSD1306_I2C
import utime
import urandom
import _thread
from ws2812 import NeoPixel

WIDTH  = 128
HEIGHT = 64
sda=machine.Pin(4)
scl=machine.Pin(5)
i2c=machine.I2C(0,sda=sda, scl=scl, freq=1000000)
neo = NeoPixel(6, n=1, brightness=0.3, autowrite=False)

oled = SSD1306_I2C(128, 64, i2c)

button = Pin(10,Pin.IN,Pin.PULL_DOWN)
RED = (255, 0, 0)
GREEN = (0, 255, 0)
BLUE = (0, 0, 255)
WHITE = (255, 255, 255)
BLACK = (0, 0, 0)

oled.fill(0)
oled.show()

neo.fill(BLACK)
neo.show()

global button_pressed
score=0
button_pressed = False

def random_rgb():
    global ledcolor
    ledcolor=int(urandom.uniform(1,4))
    if ledcolor == 1:
        neo.fill(RED)
        neo.show()
```

```
    elif ledcolor == 2:
        neo.fill(GREEN)
        neo.show()
    elif ledcolor == 3:
        neo.fill(BLUE)
        neo.show()
    elif ledcolor == 4:
        neo.fill(WHITE)
        neo.show()

def random_text():
    global oledtext
    oledtext=int(urandom.uniform(1,4))
    if oledtext == 1:
        oled.fill(0)
        oled.show()
        oled.text("RED",45,32)
        oled.show()
    elif oledtext == 2:
        oled.fill(0)
        oled.show()
        oled.text("GREEN",45,32)
        oled.show()
    elif oledtext == 3:
        oled.fill(0)
        oled.show()
        oled.text("BLUE",45,32)
        oled.show()
    elif oledtext == 4:
        oled.fill(0)
        oled.show()
        oled.text("WHITE",45,32)
        oled.show()

def button_reader_thread():
    while True:
        global button_pressed
        if button_pressed == False:
            if button.value() == 1:
                button_pressed = True
                global score
```

```
            global oledtext
            global ledcolor
            if ledcolor == oledtext:
                score += 10
            else:
                score -= 10
        utime.sleep(0.01)

_thread.start_new_thread(button_reader_thread, ())

oled.text("The Game Begins",0,10)
oled.show()
utime.sleep(2)

for i in range(10):
    for j in range (10):
        random_text()
        random_rgb()
    button_pressed=False
    utime.sleep(1.5)
    oled.fill(0)
    oled.show()
    neo.fill(BLACK)
    neo.show()
utime.sleep(1.5)
oled.fill(0)
oled.show()
oled.text("Your total score:",0,20)
oled.text(str(score), 30,40)
oled.show()
```

2.10.7. Projenin Arduino C Kodları

```
#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
#include <avr/power.h>
#endif
#define PIN       6
#define NUMPIXELS 1
```

```
Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);
#define DELAYVAL 500

#include <Wire.h>
#include “ACROBOTIC_SSD1306.h”

int OLED_color;
int RGB_color;
int score = 0;
int x = 0;

int button;

void setup() {

  Wire.begin();
  oled.init();
  oled.clearDisplay();

#if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
  clock_prescale_set(clock_div_1);
#endif

  pixels.begin();
  pixels.clear();
}

void loop() {

  oled.clearDisplay();
  oled.setTextXY(3,1);
  oled.putString(“The game begins”);
  pixels.setPixelColor(0, pixels.Color(0, 0, 0));
  pixels.show();
  delay(2000);
  oled.clearDisplay();

  for (int i=0;i<10;i++){

    random_color();
    pixels.show();
```

```
    button = digitalRead(10);
    if (button == 1){

      if(OLED_color==RGB_color & x==0){
        score=score+10;
        x=1;
      }
      if(OLED_color!=RGB_color & x==0){
        score=score-10;
        x=1;
      }
    }
    delay(2000);
    oled.clearDisplay();
    pixels.setPixelColor(0, pixels.Color(0, 0, 0));
    pixels.show();
    x=0;
  }

  String string_scrore=String(score);
  oled.clearDisplay();
  oled.setTextXY(2,5);
  oled.putString("Score: ");
  oled.setTextXY(4,7);
  oled.putString(string_scrore);
  oled.setTextXY(6,5);
  oled.putString("points");

  delay(10000);
}

void random_color(){

  OLED_color = random(1,5);
  RGB_color = random(1,5);

    if (OLED_color == 1){
      oled.setTextXY(3,7);
      oled.putString("red");
  }
```

```
  if (OLED_color == 2){
    oled.setTextXY(3,6);
    oled.putString("green");
  }
  if (OLED_color == 3){
    oled.setTextXY(3,6);
    oled.putString("blue");
  }
  if (OLED_color == 4){
    oled.setTextXY(3,6);
    oled.putString("white");
  }
  if (RGB_color == 1){
    pixels.setPixelColor(0, pixels.Color(255, 0, 0));
  }
  if (RGB_color == 2){
    pixels.setPixelColor(0, pixels.Color(0, 255, 0));
  }
  if (RGB_color == 3){
    pixels.setPixelColor(0, pixels.Color(0, 0, 255));
  }
  if (RGB_color == 4){
    pixels.setPixelColor(0, pixels.Color(255, 255, 255));
  }
}
```

2.11. Sihirli Lamba

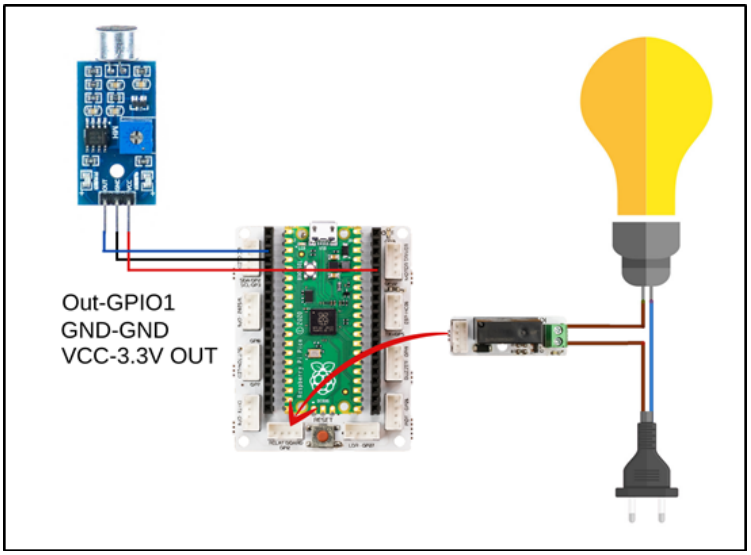
Çoğumuz, filmlerde alkış sesiyle sihirliymişçesine yanıp sönen lambaları veya açılıp kapanan kapıları görmüşüzdür. Çekimlerde bu kapıları kapatan , lambaları söndüren set yardımcıları bulunmaktadır. Peki ya biz bunu otomatik olarak gerçekleştirsek nasıl olur? Ortamda meydana gelmesini beklediğimiz ses şiddeti değişikliğini elektrik sinyaline dönüştüren sensörler vardır. Bunlara ses sensörü denmektedir.

Bu projede alkış ile yanıp sönebilen bir aydınlatma lambası düzeneği hazırlarken Picobricks ile Röle modülü ve ses seviyesi sensörünü kullanmayı ayrıca durum kontrolünü öğreneceksin.

2.12.1. Proje Detayları ve Algoritma

Bu projede evlerde kullandığımız bir lambayı sesle açıp kapatacağız. Picobricks ses seviyesi sensörü kullanarak inşa edeceğimiz projemizde alkış sesi çıkararak açma-kapama işlemlerini gerçekleştireceğiz. Önceki projelerde olduğu gibi sensörlerin kullanıldığı projelerde kodları yazmaya başlamadan önce sadece sensörü çalıştırarak yapmak istediğimiz işlemlerde sensörün hangi değerleri gönderdiğini görmek daha sonra bu değerleri baz alarak projenin kodlarını yazmak ilerlemenizi kolaylaştıracaktır. Bu projede de öncelikle MicroBlocks say123 bloğu ses şiddeti sensörü değerlerini okuyacağız ve lambanın açılıp kapanması için gerekli ses miktarını belirleyeceğiz. Daha sonra bu ses seviyesine ulaşıldığında Picobricks üzerindeki röleyi açarak röleye bağlı olan lambanın yanmasını sağlayacağız. Röleler 220 volt alternatif akımı açıp kapatarak kontrol edebilmektedirler. Açılan lambayı geri kapatmak için yine ses sensörü verisini kullanacağız. Burada aynı ses miktarı ile lambanın kapalıyken açılması, açıkken kapanması için değişken kullanmamız gerekmektedir.

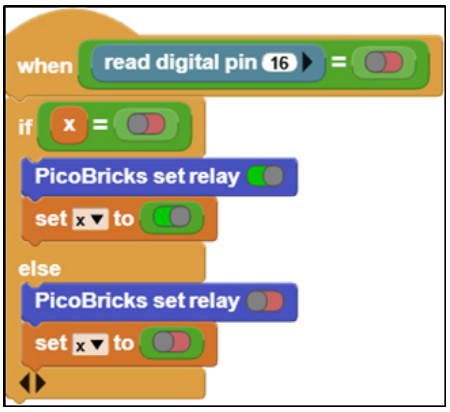
2.12.2. Bağlantı Şeması



2.11.3. Projenin MicroBlocks ile Kodlanması

Microblocks kodlarını yazarken öncelikle x adında bir değişken oluşturuyor ve hem değişkenin hem de rölenin başlangıç değerini false olarak atıyoruz. x değişkenini lambanın açıksa kapanması kapalıysa açılması işleminde kullanacağız.

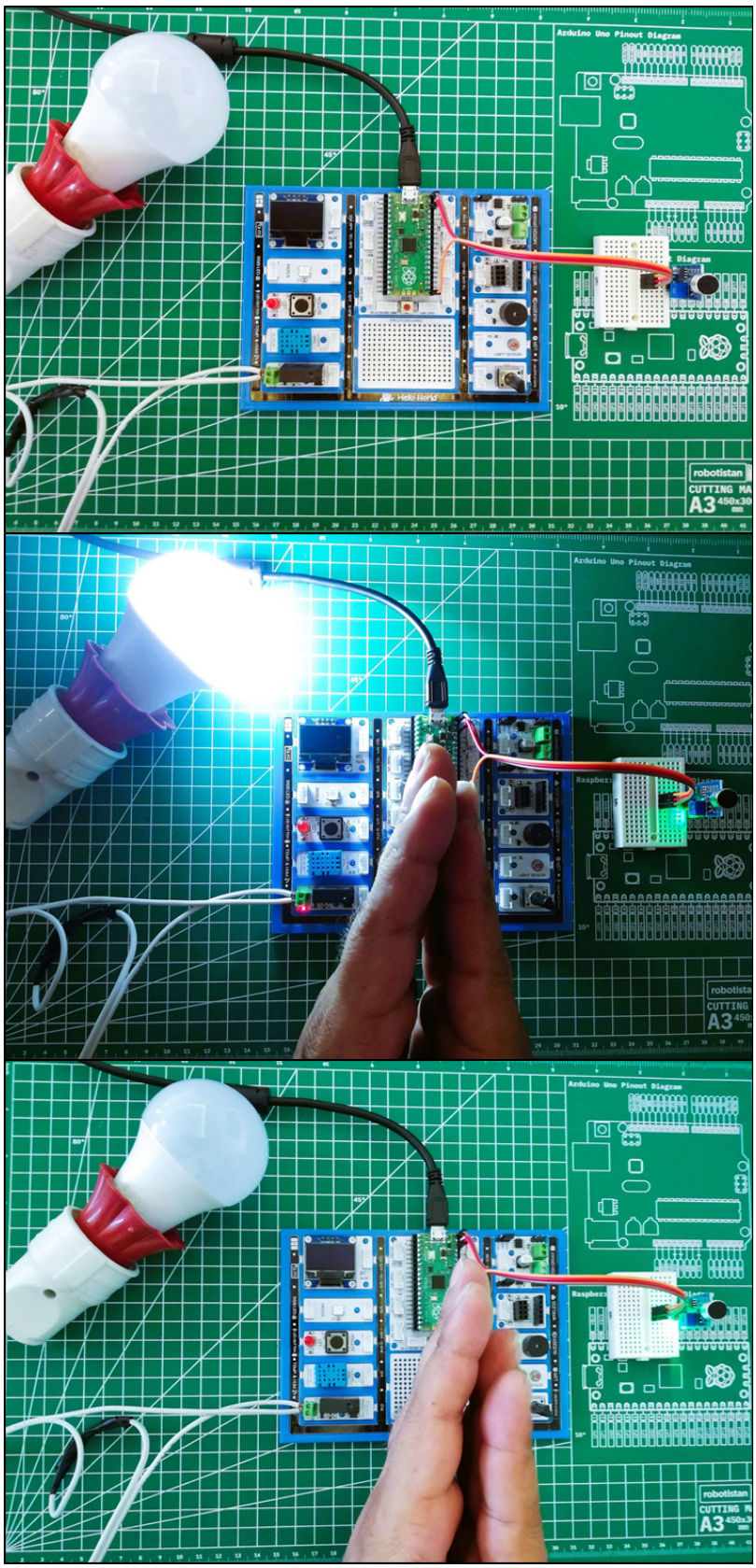
Dijital ses sensörü bağlı olduğu dijital pin üzerinden bize ses algılandığında “0”, boştayken “1” değeri gönderir. MicroBlocks bu değerleri “True” ve “False” olarak almaktadır. Yazdığımız kodlarda 16 numaralı dijital pinden gelen değer “0” yani “False” ise röle devreye girecektir. Röle devreye girdiğinde x değişkeninin değerine bakılacak, eğer x False ise röle açılacak ve x değişkeninin değeri “True” olarak değiştirilecektir, değilse röle kapatılacak ve x değişkeninin değeri tekrar “False” olarak atanacaktır. Bu kodlar alkış yaptığımızda lambanın açılması, tekrar alkış yaptığımızda lambanın kapanmasını sağlayacaktır.



Projenin MicroBlock kodlarına ulaşmak için tıkla.

2.11.4. Projenin Yapım Aşamaları

Proje inşa edilirken iki kablolu priz ve duy kullanılmıştır. Faz kablosu kesilerek kesilen iki uç röleye bağlanmıştır. Diğer kabloyu kestiğinde tehlikeli durum oluşmaması için elektrik bandı ile yalıtıma dikkat etmelisin. Üç kablolu priz kullanırsan faz kablosu olan kahverengi kabloyu keserek röleye bağlamalısın.



2.11.5. Proje Önerisi

Sihirli lamba projesi ile evindeki masa lambasını sesle kontrol edebilirsin. Röleye lamba bağlayabileceğin gibi evdeki diğer elektrikli aletleri de bağlayarak sesle ya da bluetooth üzerinden cihazları açıp kapatarak kontrol edebilirsin.

2.11.6. Projenin MicroPython Kodları

```
from machine import Pin
sensor=Pin(16,Pin.IN)
relay=Pin(12,Pin.OUT)
x=0
while True:
    if sensor.value()==0:
        if x==0:
            relay.value(1)
            x=1
        else:
            relay.value(0)
            x=0
```

2.11.7. Projenin Arduino C Kodları

```
int x=0;

void setup() {
  pinMode(16,INPUT);
  pinMode(12,OUTPUT);
  digitalWrite(12,LOW);
}

void loop() {

  if (digitalRead(16)==0){
    digitalWrite(12,HIGH);
    x=1;
  } else{
    digitalWrite(12,LOW);
    x=0;
  }
}
```

2.12. Akıllı Serinletici

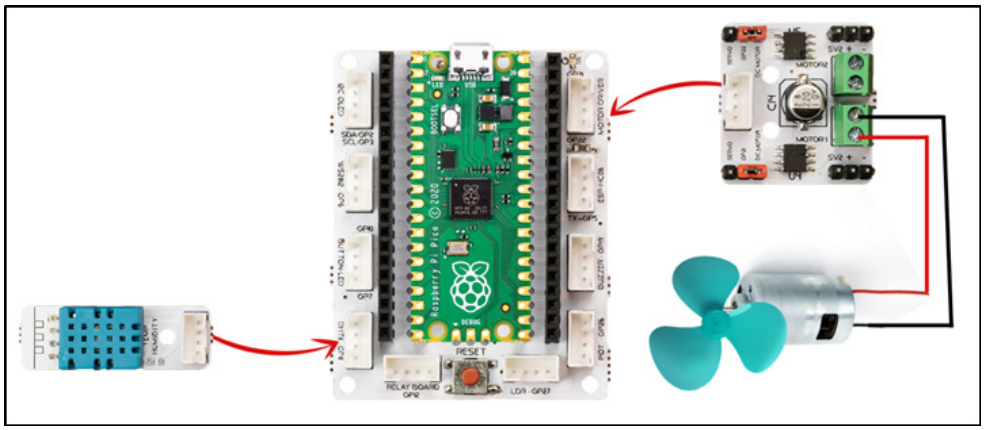
Yaz aylarında serinlemek için kış aylarında ısınmak için klimalar kullanılır. Klimalar ısıtma ve soğutma derecesini bulunduğu ortamın sıcaklığına göre ayarlamaktadır. Fırınlar yemeği pişirirken kullanıcının ayarladığı sıcaklık değerine çıkmaya ve o sıcaklığı korumaya çalışırlar. Bu iki elektronik cihazda sıcaklığı kontrol etmek için özel sıcaklık sensörleri kullanmaktadır. Ayrıca seralarda sıcaklık ve nem birlikte ölçülür. Bu iki değer istenen düzeyde dengede kalabilmesi için fan ile hava akımını sağlanmaya çalışılır.

Picobricks'te sıcaklığı ve nemi ayrı ayrı ölçebilir ve bu ölçümler ile çevreyle etkileşime girebilirsiniz. Bu projede Picobricks ile sıcaklığa göre fan hızını otomatik ayarlayan bir serinletme sistemi hazırlayacağız. Böylelikle DC motor çalışma sistemini ve motor hız ayarı yapmayı öğreneceksin.

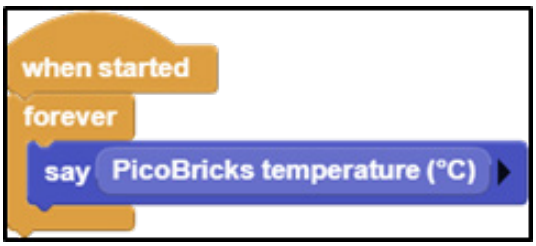
2.12.1. Proje Detayları ve Algoritma

Projemizde öncelikle Picobricks üzerindeki DHT11 sıcaklık ve nem sensörünün ölçtüğü sıcaklık değerlerini görüntüleyeceğiz. Daha sonra sonra bir sıcaklık sınırı belirleyerek DHT11 modülünden gelen sıcaklık değeri bu sınıra ulaştığında Picobricks'e bağlı DC motorun dönmeye başlaması, sıcaklık değeri belirlediğimiz sınırın altına indiğinde ise DC motorun durması için gerekli kodları yazacağız.

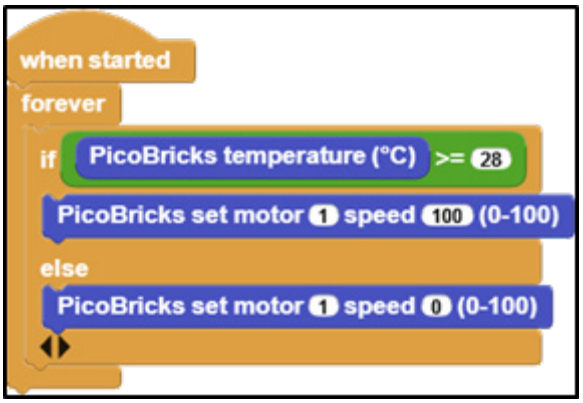
2.12.2. Bağlantı Şeması



2.12.3. Projenin MicroBlocks ile Kodlanması



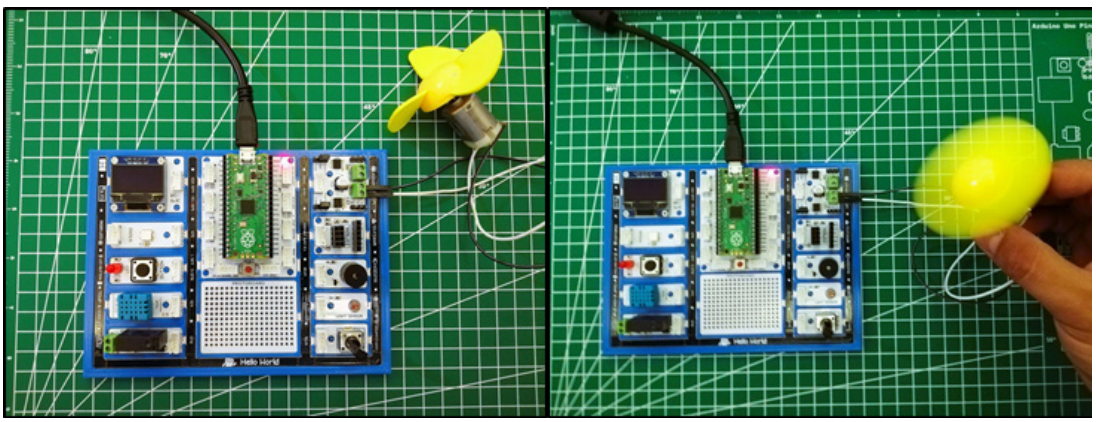
Fanın ne zaman devreye girerek çalışacağına karar vermek için öncelikle DHT11 sensöründen gelen değerleri görmemiz ve bu değerlere göre işlem yapmamız gerekiyor. Bunun için Output kategorisindeki say123 bloğunu kullanabilirsin. Daha sonra Picobricks kategorisindeki PicoBricks temperature bloğunu sürükleyerek say bloğundaki 123 yazan yuvarlağa bırak. Start tuşuna basarak sensörden gelen değerleri gör. Oda sıcaklığında 25 derece civarında değerler görmelisin. Picobricks üzerindeki DHT11 modülüne parmağını değdirerek bir süre bekle. DHT11'in parmağındaki ısıdan etkilenerek sensörden gelen sıcaklık değerinin arttığını göreceksin. Değerleri gördükten sonra say ve temperature bloklarını silebilirsiniz.



Projenin MicroBlocks kodlarına erişmek için tıkla.

Bulunduğun ortama göre fanın devreye girmesi için bir sıcaklık değeri belirledikten sonra if else bloğu içinde Picobricks set motor bloğunu kullanarak projeyi gerçekleştirebilirsin. Fanın hızını 0 ile 100 arasında değiştirebilirsin.

2.12.4. Proje Görseli



2.12.5. Proje Önerisi

Picobricks üzerindeki OLED ekranın kullanarak sıcaklığı ekrana yazdırabilir ve fanın devreye girdiği sıcaklığı takip edebilirsin.

Picobricks modüler yapıdadır, modüller kırılarak ayrılabilir ve grove kablolar ile Pico boarda bağlanarak kullanılabilir. Projemizde yaptığımız akıllı serinletici devresini, robot araba şasesine montajlayarak bulunduğun ortamda otonom olarak gezen ve aynı anda ortamı serinleten bir proje geliştirebilirsin.

2.12.6. Projenin MicroPython Kodları

Geçerli sıcaklığın shell penceresine yazdıran kodlar:

```
from machine import Pin
from dht import DHT11
from utime import sleep
dht_sensor = DHT11(11)

while True:
    sleep(1) # It was used for DHT11 to measure.
    dht_sensor.measure() # Use the sleep() command before this line.
    temp=dht_sensor.temperature()
    print(temp)
```

Projenin Kodları:

```
from machine import Pin
from dht import DHT11
from utime import sleep
dht_sensor = DHT11(11)
m1 = Pin(21, Pin.OUT)
m1.low()

while True:
    sleep(1) # It was used for DHT11 to measure.
    dht_sensor.measure() # Use the sleep() command before this line.
```

```
    temp=dht_sensor.temperature()
    print(temp)
    if temp>=28.0:
        m1.high()
    else:
        m1.low()
```

2.12.7. Projenin Arduino C Kodları

```
#include “EspDHT.h”
EspDHT dht;

void setup()
{
  dht.setup(11, EspDHT::DHT11);
  pinMode(21,OUTPUT);
}

void loop()
{
  delay(100);
  dht.readSensor();
  float temperature = dht.getTemperature();

  if(temperature>27){
    digitalWrite(21,HIGH);
  } else{
    digitalWrite(21,HIGH);
  }
}
```

2.13. Buzz Wire Game

Projeler her zaman sorun çözmek ve işleri kolaylaştırmak için olmak zorunda değil. Eğlenmek ve gelişmek için de projeler hazırlayabilirsin. Dikkat ve konsantrasyon bir çok insanın geliştirmek istediği özellikleridir. Bunu yapabileceği uygulamalar oldukça ilgi çekicidir. Buzz Wire Game'i Picobricks ile yapmaya ne dersin?

Bilgisayarlar 0 ve 1 ler ile çalışır ifadesini mutlaka duymuşsunuzdur. 0 elektriğin yokluğunu 1 ise varlığını temsil eder. 0 ve 1 ler belirli sayıda ve sıralamadaki kombinasyonlarla bir araya gelerek anlamlı verileri oluştururlar. Elektronik sistemlerde 0 ve 1 ler doğrudan bir durumu kontrol etmek için kullanılabilmektedir. Kapı kapalı mı değil mi? Işık açık mı kapalı mı? Sulama sistemi devrede mi değil mi? bunun gibi bilgiler elde etmek için durum kontrolü yapılmaktadır.

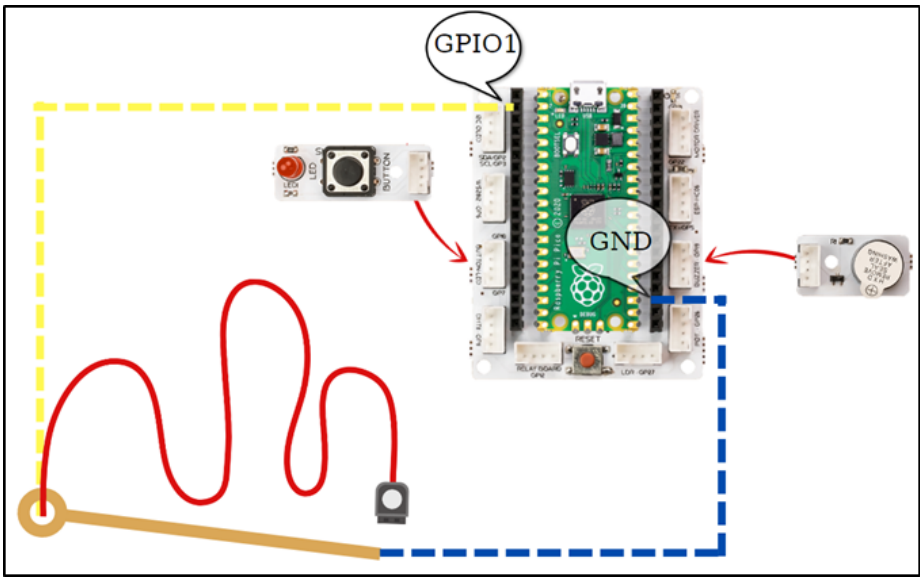
Bu projede Picobricks ile buzzer ve LED modülünü kullanarak iletken tel yardımı ile dikkat ve konsantrasyon geliştirici Buzz Wire Game oyununu elektronik olarak hazırlayacağız. Bu projeyi hazırlarken buton olmayan ancak buton gibi kullanılacak bir giriş tekniği de öğrenmiş olacaksın.

2.13.1. Proje Detayları ve Algoritma

Projeyi hazırlamak için 2 adet erkek-erkek jumper kablo ve 15 cm uzunluğunda iletken bükülebilir tele ihtiyacın var. Oyuncu hazır olduğunda oyunu başlatmak için butona basması istenecek. Butona basıldığında oyuncunun elinde jumper kablo iletken tele değerse Picobricks bunu algılayıp sesli ve yazılı uyarı verecek. Oyun başladıktan bitene kadar geçen sürede OLED ekranda gösterilecek.

Kullanıcı butona bastıktan sonra timer'ı resetliyoruz. Daha sonra Picobricks'in GPIO1 nolu pinine bağlı iletken tel'e 3.3V'luk gerilim vereceğiz. Oyuncunun elinde tuttuğu kablonun bir ucu Picobricks üzerinde GND pinine bağlı olacak. Eğer oyuncu elindeki jumper kabloyu iletken tele dokundurursa GPIO1 nolu pin Pasif/Kapalı/0 konumuna düşecektir. Daha sonra Oyunun bittiğini bildirir ışıklı, yazılı ve sesli geri bildirimde bulunulacak ardından OLED ekran üzerinde geçen süre milisaniye cinsinden gösterilecek. 5 saniye sonunda oyuncunun yeniden başlamak için butona basması istenecek.

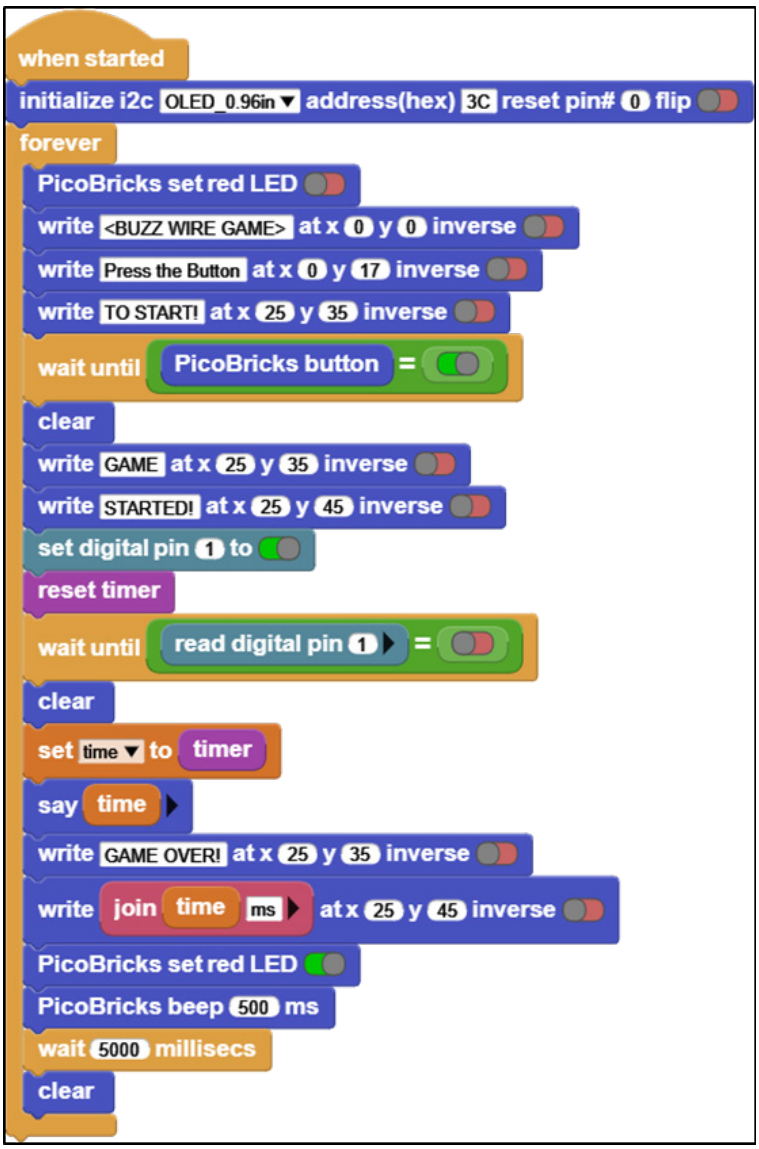
2.13.2. Bağlantı Şeması



2.13.3. Projenin MicroBlocks ile Kodlanması

PicoBricks başladığında OLED ekranı başladıktan hemen sonra sonsuz bir döngü açıyoruz. Çünkü oyun bittikçe başa dönecek. Sonsuz döngünün içine ilk önce kırmızı ledi kapatıp OLED ekrana başlangıç mesajı ifadelerini yerleştiriyoruz. Ardından Control kategorisinden wait until bloğunu yerleştirerek Picobricks'in butonuna tıklanana kadar döngüyü bekletiyoruz.

Butona basıldıktan sonra kodların devamında OLED ekrana oyunun başladığını bildiren ifade ekliyoruz. Ardından GPIO1 pinini açarak 3.3V gerilim veriyoruz. Zamanlayıcıyı sıfırladıktan sonra wait until bloğu içinde GPIO1 pini 0 yani kapalı olana kadar bekliyoruz. burada Oyuncunun elindeki kabloyu tele dokundurma durumu kontrol ediliyor. Dokunma algılandığında döngüye kaldığı yerden devam edilip gerekli ışıklı yazılı ve sesli bildirimler sunuluyor. Son olarak 5 saniye bekleyip döngünün başına dönülüyor.



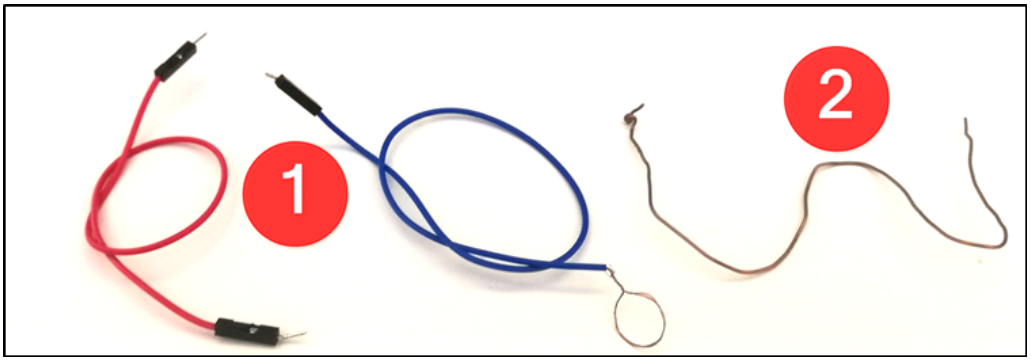
Projenin kodlarına erişmek için tıkla.

2.13.4. Projenin Yapım Aşamaları

PicoBricks base kit ile beraber,

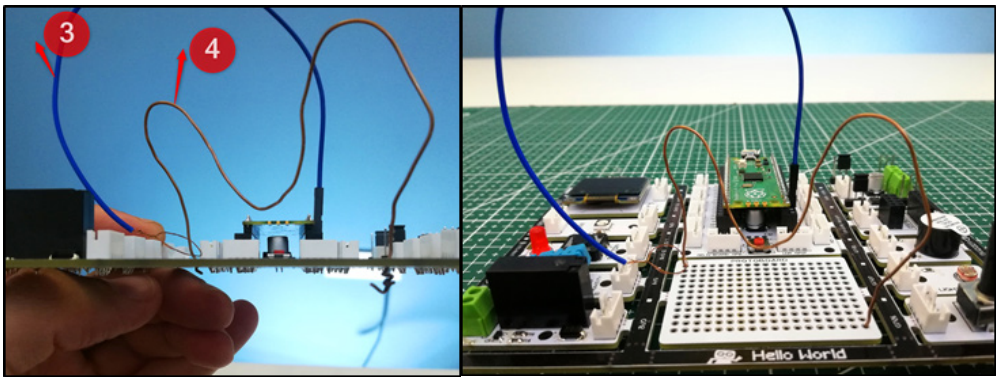
1: 2 adet 20 cm erkek-erkek jumper kablo. GND'ye takılacak kablonun bir ucu 4- 5 cm soyulup halka şekline getir.

2: 0.8 mm kalınlığında 15-20 cm iletken tel malzemelerini hazırla.



Protoboard üzerine iletken teli dilediğin şekilde bükerek deliklerinden geçir bir ucunu geçirmeden önce erkek ucunu PicoBoard üzerindeki GND pinine bağlı olan diğer ucunu halka haline getirdiğin kablonun içinde geçirmelisin.

3: İletken tel

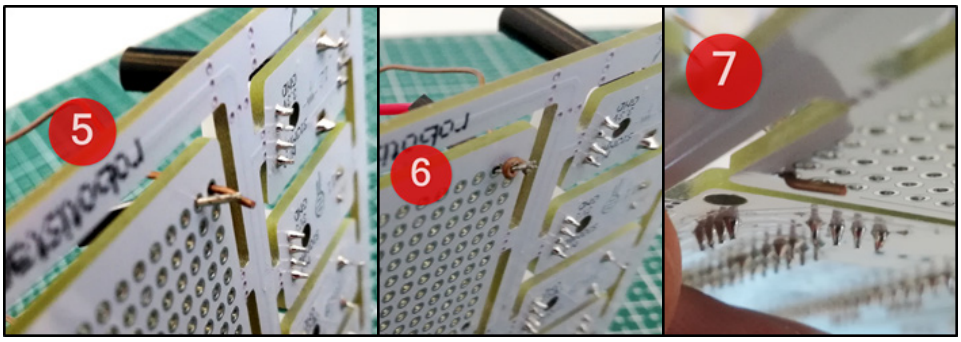


4: Bir ucu GND pinine bağlı ucu halka haline getirilmiş Jumper kablo

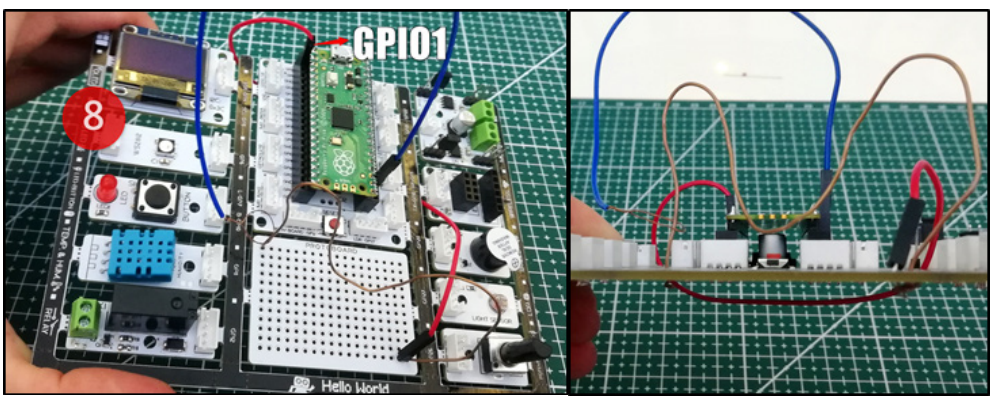
5: İki ucu da erkek olan jumper kablonun bir ucunu, protoboard'a yerleştirdiğin iletken telin ucunun hemen yanındaki deliğe tak.

6: Protoboard'ın altından jumper kablonun ucu ile iletken telin ucunu birbirine dola.

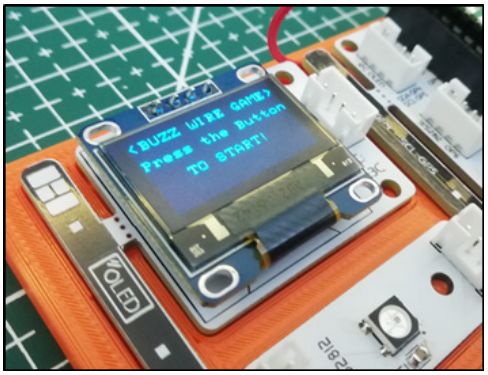
7: İletken telin protoboard'a yerleştirilmiş diğer ucunun da çıkmaması için bük.

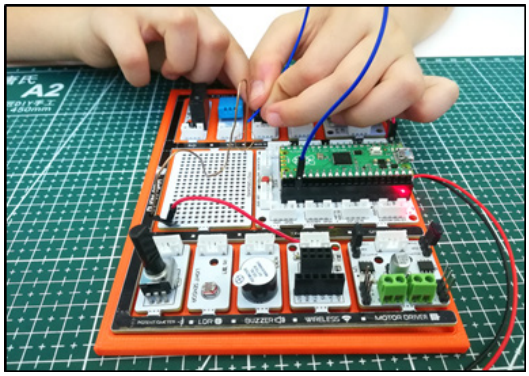


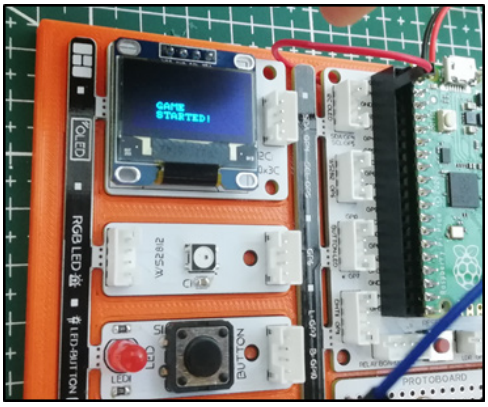
8: 6. adımda iletken telin ucuna doladığın jumper kablonun diğer erkek ucunu Picoboard üzerindeki GPIO1 nolu pine tak

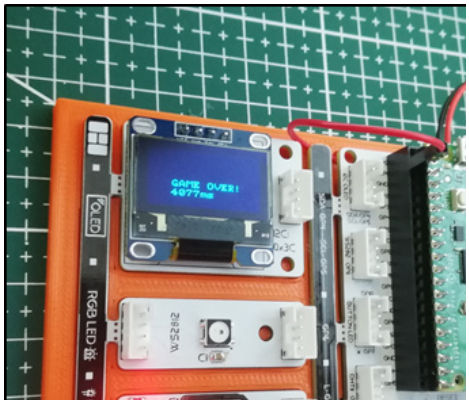


Kurulumu tamamladıysan kodlarını yükledikten sonra oyuna başlayabilirsin. İyi eğlenceler.









2.13.5. Proje Önerisi

Projeye fiziksel ve yazılımsal geliştirmeler yapabilirsin. Başlangıç ve bitiş noktalarını yalıtkan bant ile kapatarak oyuncunun oyuna başlarken ve bitirirken sorun yaşamasını engelleyebilirsin. Yazılımsal olarak ise oyuncu elindeki kabloyu tele değdirmeden diğer uca getirip bıraktığında butona basar ve skorunu OLED ekranda görmesini sağlayabilirsin.

2.13.6. Projenin MicroPython Kodları

```
from machine import Pin, I2C, Timer
from ssd1306 import SSD1306_I2C
from utime import sleep

WIDTH  = 128
HEIGHT = 64

sda=machine.Pin(4)
scl=machine.Pin(5)
i2c=machine.I2C(0,sda=sda, scl=scl, freq=1000000)
```

```
oled = SSD1306_I2C(128, 64, i2c)

wire=Pin(1,Pin.OUT)
led = Pin(7,Pin.OUT)
buzzer=Pin(20, Pin.OUT)
button=Pin(10,Pin.IN,Pin.PULL_DOWN)
endtime=0
while True:
    led.low()
    oled.fill(0)
    oled.show()
    oled.text("<BUZZ WIRE GAME>",0,0)
    oled.text("Press the button",0,17)
    oled.text("TO START!",25,35)
    oled.show()
    while button.value()==0:
        print("press the button")
    oled.fill(0)
    oled.show()
    oled.text("GAME",25,35)
    oled.text("STARTED",25,45)
    oled.show()
    wire.high()
    timer_start=utime.ticks_ms()
    while wire.value()==1:
        print("Started")
    endtime=utime.ticks_diff(utime.ticks_ms(), timer_start)
    print(endtime)
    oled.fill(0)
    oled.show()
    oled.text("GAME OVER!",25,35)
    oled.text(endtime + "ms" ,25,45)
    oled.show()
    led.high()
    buzzer.high()
    sleep(5)
```

2.13.7. Projenin Arduino C Kodları

```
#include <Wire.h>
#include “ACROBOTIC_SSD1306.h”

int Time=0;
unsigned long Old_Time=0;

void setup() {

  pinMode(20,OUTPUT);
  pinMode(7,OUTPUT);
  pinMode(1,OUTPUT);
  pinMode(10,INPUT);

  Wire.begin();
  oled.init();
  oled.clearDisplay();

#if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
  clock_prescale_set(clock_div_1);
#endif
}

void loop() {

  digitalWrite(7,LOW);

  oled.setTextXY(2,1);
  oled.putString(“BUZZ WIRE GAME”);
  oled.setTextXY(4,2);
  oled.putString(“Press Button”);
  oled.setTextXY(5,3);
  oled.putString(“TO START!”);

  while (!(digitalRead(10)==1)){

  }

  oled.clearDisplay();
  oled.setTextXY(3,6);
```

```
  oled.putString("GAME");
  oled.setTextXY(5,4);
  oled.putString("STARTED");

  digitalWrite(1,HIGH);
  Old_Time=millis();

  while(!(digitalRead(1)==0)){

    Time=millis()-Old_Time;
  }

  String(String_Time)=String(Time);

  oled.clearDisplay();
  oled.setTextXY(3,4);
  oled.putString("GAME OVER");
  oled.setTextXY(5,4);
  oled.putString(String_Time);
  oled.setTextXY(5,10);
  oled.putString("ms");

  digitalWrite(7,HIGH);
  digitalWrite(20,HIGH);
  delay(500);
  digitalWrite(20,LOW);
  delay(5000);

  Time=0;
  Old_Time=0;
  oled.clearDisplay();
}
```

2.14. Dinazor Oyunu

Geliştirilecek elektronik sistemler görevini itme, çekme, döndürme, kaldırma, indirme gibi hareketle sonuçlanan işlerle yerine getireceklerse projede aktuatör olarak pnomatik sistemler ya da elektrik motorlu sistemler kullanılır. Picobricks, projelerinde yazdığın kodları harekete geçirebilecek sistemler üretebilmen için iki farklı motor tipini desteklemektedir. DC motor ve DC motorların elektronik olarak hareketlerinin düzenlendiği Servo motorlar. Servo motorlar dönüş açısı değeri verildiğinde o açıya dönen motorlardır. Model uçakların kanatlarında, uçağa yön verebilmek için kanat uçlarındaki yaprakları servo motorlar ile hareket ettirilmektedir. RC teknelerde aracın yönünü değiştirmek için de aynı mantıkla servo motorlar kullanılmaktadır. Ayrıca tam tur dönebilen, akıllı sürekli servo olarak bilinen gelişmiş servo motorlar, evlerimizde kullandığımız akıllı süpürgelerin tekerleklerinde de kullanılmaktadır.

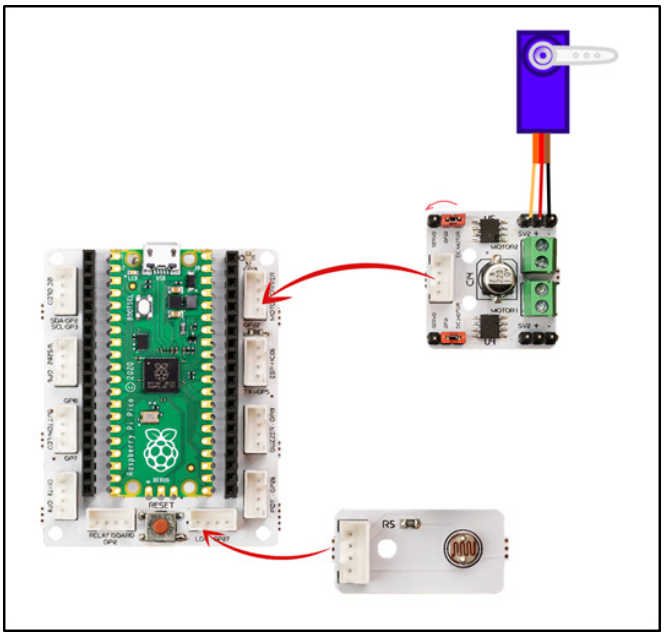
Bu projede PicoBricks ile Servo motorların nasıl kontrol edilebildiğini öğreneceksin.

2.14.1. Proje Detayları ve Algoritma

Bu projede Google Chrome offline dinasour game'i otomatik olarak Picobricks'e oynatacağız. Oyunda Picobricks engelleri algılayarak otomatik olarak dinazorun hareketlerini kontrol edecek. Oyun esnasında dinazorun karşısına çıkan engelleri algılamak için picobricks LDR sensör kullanacağız. LDR sensör yüzeyine temas eden ışık miktarını ölçerek analog sinyaller gönderebilmektedir. Sensörü bilgisayar ekranına sabitleyerek beyaz ve siyah renkler arasındaki ışık miktarı farkından yararlanarak dinazorun önüne engel gelip gelmediğini algılayabiliriz. Engel algılandığında ise servo motor kullanarak klavyedeki boşluk tuşuna otomatik olarak basılmasını sağlayabiliriz. Bu sayede dinazor engelleri kolaylıkla aşacaktır. Proje kodlarını yazarken öncelikle LDR sensörü bilgisayar ekranına sabitleyerek beyaz ve siyah zemindeki sensör verilerine okuyacak daha sonra bu verilere göre servo motorun hareket etmesi için gerekli kodları yazacağız.

2.14.2. Bağlantı Şeması

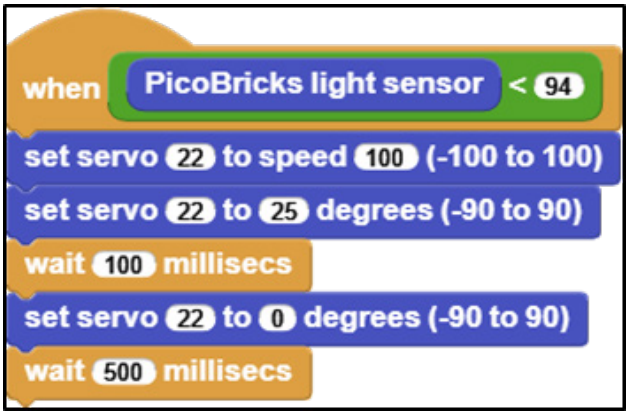
Not: Motor sürücü grove kablo girişinin sağ ve sol tarafında üçlü pinler bulunmakta ve bu pinler 2'li jumper ile kısa devre yapılmaktadır. DC motor kullanırken DC motor tarafında takılı olması gereken jumper, servo motor kullanırken çıkarılarak servo tarafına takılmalıdır.



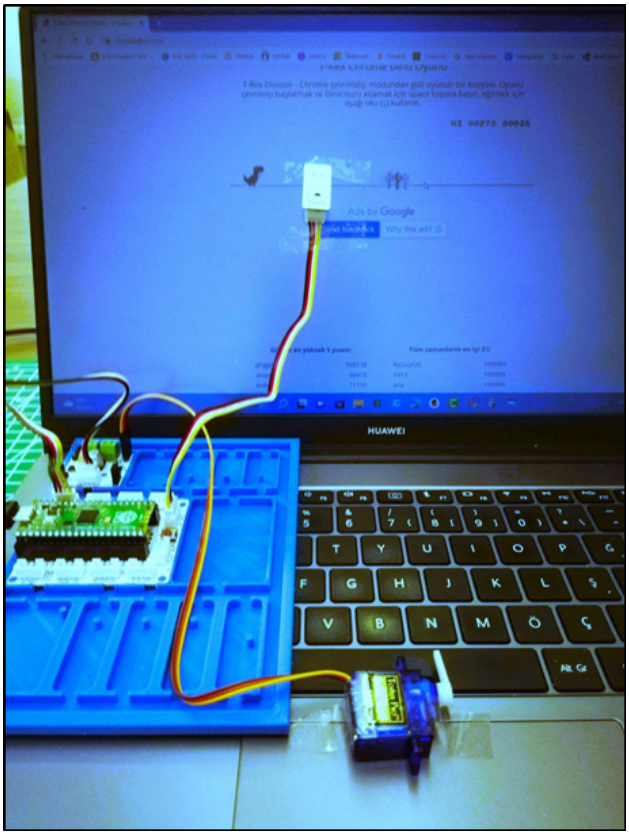
2.14.3. Projenin MicroBlocks ile Kodlanması

Projenin çalışması için öncelikle bulunduğunuz ortama göre değişecek olan LDR sensör değerlerini okumalısın. Bunun için say123 bloğunu kullanabilirsin. Chrome offline dinosaur game aç. Sensörü dinazordan 3-4 cm sağ tarafa ve yol çizgisinin hemen üzerine bant yardımıyla sabitle. Sensörün ekrana değdiğinden emin olduktan sonra sensör değerlerini oku. Beyaz zeminde iken gelen değerler, engel geldiğinde gelen değerlerden farklı olacaktır. Aradaki farkı gözlemleyerek belirleyeceğiniz sınır değerini when bloğu ile koda dönüştür. LDR sensör değeri belirlediğin değerden küçük olduğunda servonun açısının 25 derece değişip eski konumuna geri gelmesi için gerekli kodları yaz ve servo motoru space tuşuna otomatik olarak basacak şekilde klavyenize sabitle.

Projenin MicroBlocks kodlarına ulaşmak için [tıkla.]



2.14.4. Proje Görseli



2.14.5. Proje Önerisi

Oyunda ilk başta zemin rengi beyaz figürler siyah renklidir. Belirli bir aşamadan sonra ise renkler tersine dönmektedir. Bu sebeple LDR sensör verileri değişmektedir. Bu problemi çözmek için değişken ve fonksiyon kullanarak oyun beyaz zemindeyken bir kod grubu, siyah zemindeyken diğer kod grubunu çalıştırabilir ya da bu farkı algılaması için ikinci bir LDR sensör takabilirsin.

Picobricks ve modülleri basitten karmaşığa birçok proje geliştirmemize imkan sunmaktadır. Günlük hayatta oynadığımız bir bilgisayar oyununu otomatik olarak Picobricks'e oynattığımız bu projeyi geliştirerek minecraft gibi farklı oyunlarda da kullanabilirsin.

2.14.6. Projenin MicroPython Kodları

Işık sensörünün okuduğu değeri shell penceresine yazdıracak kodlar:

```
from machine import Pin, ADC
from utime import sleep
```

```
ldr=ADC(27)

while True:
    print(ldr.read_u16())
    sleep(0.01)
```

Projenin Kodları:

```
from machine import Pin, ADC,PWM
from utime import sleep

ldr=ADC(27)
servo=PWM(Pin(21))
servo.freq(50)

while True:
    sleep(0.01)
    if ldr.read_u16()>40000:
        servo.duty_u16(2000)
        sleep(0.1)
        servo.duty_u16(1350)
        sleep(0.5)
```

2.14.7. Projenin Arduino C Kodları

```
#include <Servo.h>
Servo myservo;

void setup() {
  myservo.attach(22);
  myservo.write(20);
  pinMode(27,INPUT);
}

void loop() {

  int light_sensor=analogRead(27);

  if(light_sensor>100){
```

```
    int x=45;
    int y=20;

    myservo.write(x);
    delay(100);
    myservo.write(y);
    delay(500);
  }
}
```

2.15. Gece Gündüz

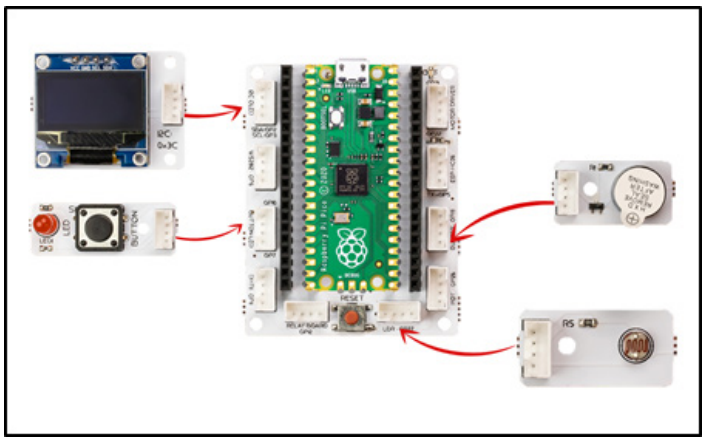
Okulda oynadığınız Gece Gündüz oyununu elektronik olarak oynamaya ne dersin? Öğretmen gece dediğinde kafamızı öne eğip masanın üzerindeki kolumuza yasladığımız, gündüz dediğinde başımızı kaldırdığımız bir oyundur gece-gündüz oyunu. Bu oyun dikkatini ve refleksini kullanacağın bir oyun olacak. Bu projede 0,96" 128x64 piksel I2C OLED ekranı kullanacağız. OLED ekranlar yapay ışık kaynağı olarak kullanılabildikleri için ekran üzerindeki karakterleri mercek ve ayna kullanarak büyütebilir ve istediğiniz düzleme yansıtabilirsin. Akıllı gözlükler ve otomobil camlarına bilgilendirme, yol ve trafik bilgisi yansıtabilen sistemler OLED ekranlar kullanılarak yapılabilmektedir.

Işık sensörleri bulundukları ortamın ışık seviyelerini ölçebilen, fotodiyot da denilen sensörlerdir. Işığa maruz kalan sensörün elektrik geçirgenliği değişmektedir. Biz de kodlayarak ışık sensörünü kontrol edip, ışık miktarının etkilediği elektronik sistemler geliştirebilmekteyiz.

2.15.1. Proje Detayları ve Algoritma

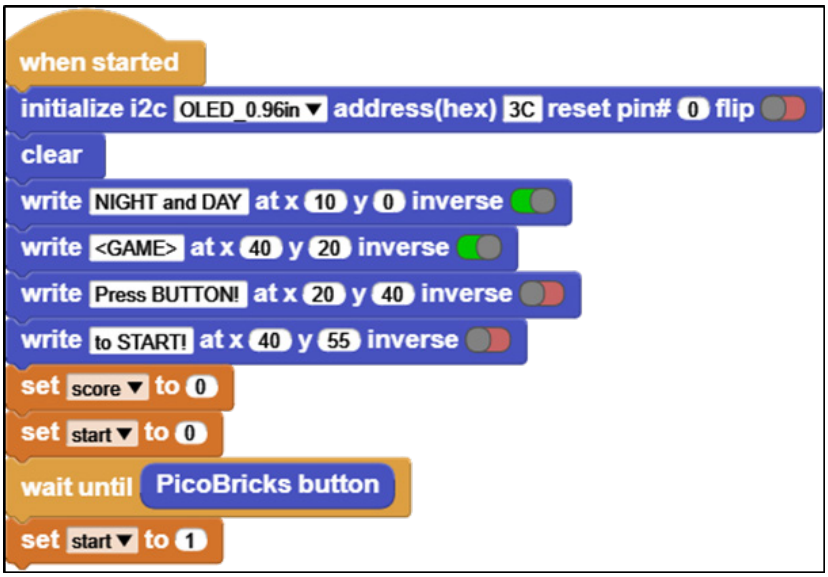
Önce oyuncunun oyuna başlaması için butona basmasını isteyeceğiz. Ardından PicoBricks'in OLED ekranında NIGHT ve DAY ifadelerini rastgele olarak 2'şer saniye boyunca gösterilmesini sağlayacağız. Oyuncu, eğer OLED ekranda yazan kelime NIGHT ise 2 saniye içinde LDR sensörünün üzerini eliyle kapatmalı, eğer OLED ekranda GÜNDÜZ kelimesi yazıyorsa LDR sensörünün üzerinden elini kaldırmalı. Oyuncunun her doğru tepkisi 10 puan kazanmasını sağlayacak, yanlış tepkide ise oyun bitecek ve ekranda oyunun bitmesini bildiren yazılı ifade yer alacak, buzzerdan farklı tonda bir ses çalacak ve OLED ekranda puan bilgisi yer alacaktır. Eğer oyuncu toplam 10 doğru tepki verip 100 puan elde ederse "Congratulation" yazısı ile oyuna yeniden başlayabilmek için RESET butonuna basılmasını bildiren ifade OLED ekranda gösterilip buzzerdan farklı tonda notalar çalacaktır.

2.15.2. Bağlantı Şeması



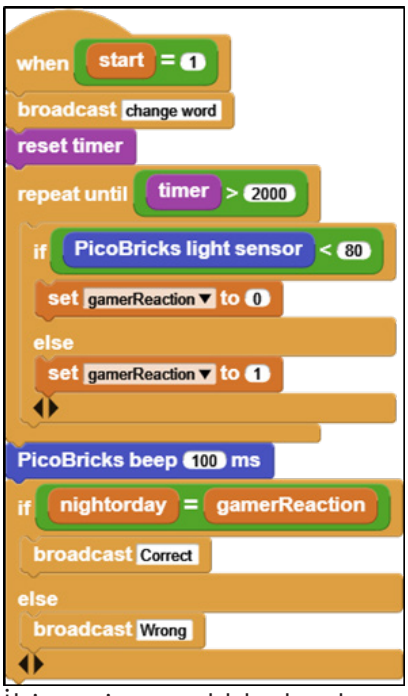
2.15.3. Projenin MicroBlocks ile Kodlanması

Picobricks başladığında OLED ekranı tanımlayıp başlangıç ekran mesajlarını yazdır.. Ardından score, start, nightorday ve gamerReaction isimli değişkenler oluştur. Oyun butona basıldıktan sonra başlayacağı için wait until bloğundan sonra start değişkeni 1 yapılır.

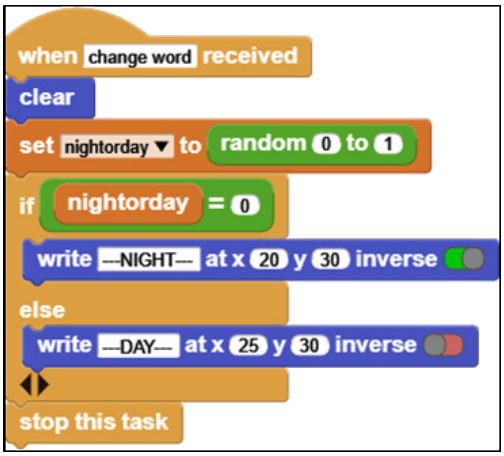


Oyun başladığında her iki saniyede “change word” yayını yapılacak. Bu haber sayesinde ekranda DAY veya NIGHT ifadesi yazdırılacak. Kelimenin değişmeden ekranda kaldığı 2 saniye süresince oyuncunun tepkisi LDR sensöründen okunan değere göre gamerReaction değişkenine atanacak. LDR değeri 80’den büyükse sensörün üstü açıktır değilse sensör kapatılmıştır. 80 değerini kendi çalışma koşulların için değiştirebilirsin.

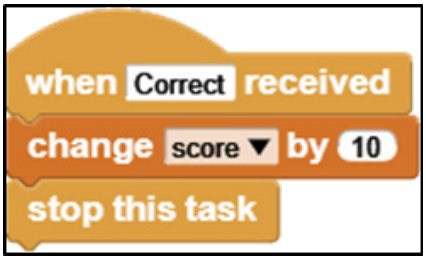
Geçen iki saniyenin sonunda ise gamerReaction ile rastgele belirlenen kelime(nightorday) kıyaslanarak Correct ve Wrong yayınları yapılıyor. Bu yayınlar ise oyunda puan kazanma ve oyunun bitmesi için kullanılacak.



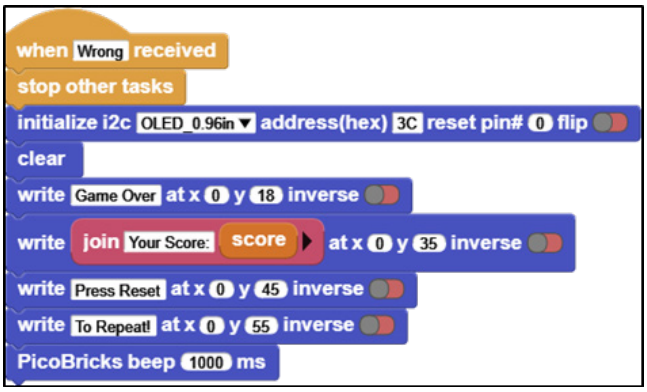
İki saniye aralıklarla change word iletisi alındığında ekran temizlenerek 0 ya da 1 değer rastgele nightorday değişkenine atanır. Değer 0 ise ekrana NIGHT 0 değil ise ekrana DAY yazdırılır. Yayın bloğunun 1 defa çalışması için bu komut dizisi en sonda stop this task komutu ile durdurulur.



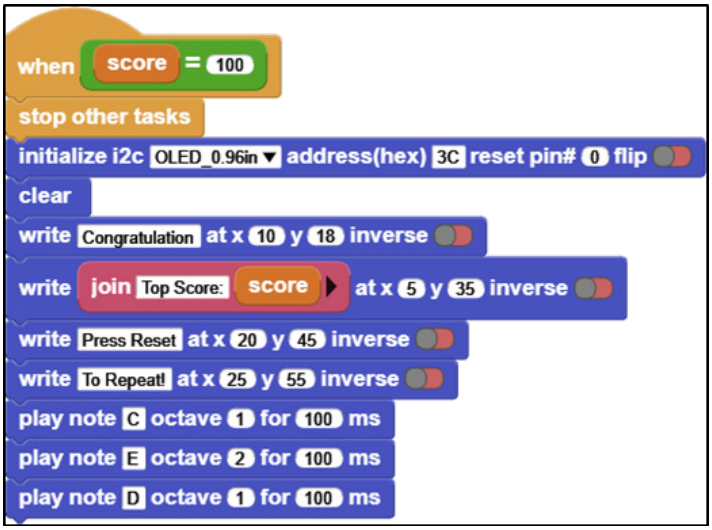
Oyuncu doğru tepki verdiğinde yayınlanan ”Correct” iletisi için score değişkenini bir seferlik 10 arttıran kodların yer aldığı kod bloğu aşağıdaki gibidir.



Oyuncu yanlış tepki verdiğinde yayınlanan “Wrong” iletisi için diğer tüm kodları durdurup ekrana oyunun bitmesi ile ilgili yazıyı ve score değerini yazdıran kodların yer aldığı kod bloğu aşağıdaki gibidir.

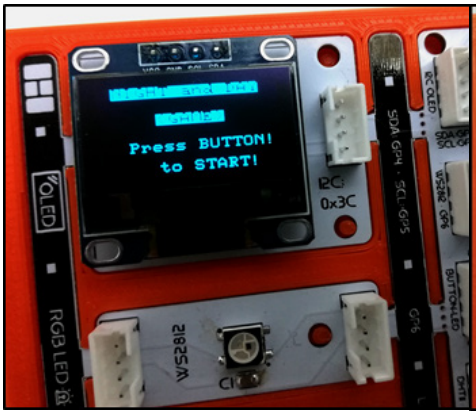


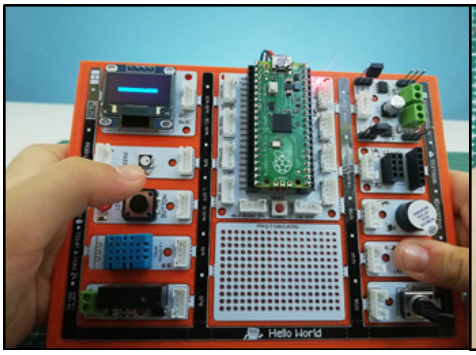
Oyuncu oyunu hatasız bitirdiğinde score değişkeni 100 değerini alacaktır. Bunu yakalayıp oyuncuyu tebrik edip oyunu durduran kodların yer aldığı kod bloğu aşağıdaki gibidir.

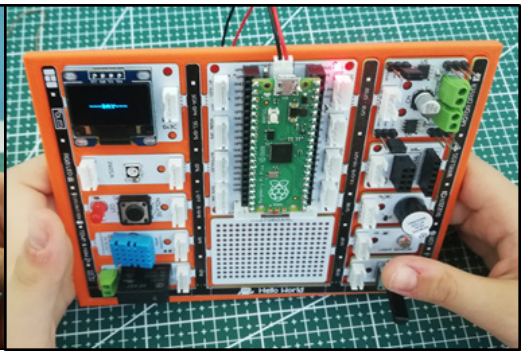


Projenin kodlarına erişmek için tıkla.

2.15.4. Proje Görseli







2.15.5. Proje Önerisi

Projeye LDR sensörün bulunduğun ortama göre gönderdiği değerleri alarak oyun içerisinde sensör değerine göre işlem yapılacak sınırın otomatik belirlenmesi, yani LDR sensör değeri kalibrasyonu kodları ekleyerek projeyi geliştirebilirsin. Oyuna zorluk seviyesi ekleyebilirsin. Potansiyometre ile zorluk seviyesi kolay orta ve zor seçimi yaptırılabilir. Kolay seçiliyken kelimelerin değişme süresi 2 saniye orta seçiliyken 1.5 saniye zor seçiliyken 1 saniye olabilir.

2.15.6. Projenin MicroPython Kodları

```
from machine import Pin, I2C, Timer, ADC, PWM
from ssd1306 import SSD1306_I2C
from utime import sleep
import utime
import urandom

WIDTH  = 128
HEIGHT = 64
sda=machine.Pin(4)
scl=machine.Pin(5)
i2c=machine.I2C(0,sda=sda, scl=scl, freq=1000000)
oled = SSD1306_I2C(128, 64, i2c)

buzzer = PWM(Pin(20))
buzzer.freq(440)
ldr=ADC(27)
button=Pin(10,Pin.IN,Pin.PULL_DOWN)

oled.text("NIGHT and DAY", 10, 0)
oled.text("<GAME>", 40, 20)
oled.text("Press the Button", 0, 40)
oled.text("to START!", 40, 55)
oled.show()

def changeWord():
    global nightorday
    oled.fill(0)
    oled.show()
    nightorday=round(urandom.uniform(0,1))
    if nightorday==0:
        oled.text("---NIGHT---", 20, 30)
        oled.show()
    else:
        oled.text("---DAY---", 20, 30)
        oled.show()

while button.value()==0:
    print("Press the Button")
    sleep(0.01)
```

```
oled.fill(0)
oled.show()
start=1
global score
score=0
while start==1:
    global gamerReaction
    global score
    changeWord()
    startTime=utime.ticks_ms()

    while utime.ticks_diff(utime.ticks_ms(), startTime)<=2000:
        if ldr.read_u16()>20000:
            gamerReaction=0
        else:
            gamerReaction=1
        sleep(0.01)

    buzzer.duty_u16(2000)
    sleep(0.05)
    buzzer.duty_u16(0)
    if gamerReaction==nightorday:
        score += 10
    else:
        oled.fill(0)
        oled.show()
        oled.text("Game Over", 0, 18, 1)
        oled.text("Your score " + str(score), 0,35)
        oled.text("Press RESET",0, 45)
        oled.text("To REPEAT",0,55)
        oled.show()
        buzzer.duty_u16(2000)
        sleep(0.05)
        buzzer.duty_u16(0)
        break;
    if score==100:
        oled.fill(0)
        oled.show()
        oled.text("Congratulation", 10, 10)
        oled.text("Top Score: 100", 5, 35)
```

```
        oled.text("Press Reset", 20, 45)
        oled.text("To REPEAT", 25,55)
        oled.show()
        buzzer.duty_u16(2000)
        sleep(0.1)
        buzzer.duty_u16(0)
        sleep(0.1)
        buzzer.duty_u16(2000)
        sleep(0.1)
        buzzer.duty_u16(0)
        break;
```

2.15.7. Projenin Arduino C Kodları

```
#include <Wire.h>
#include "ACROBOTIC_SSD1306.h"

int Gamer_Reaction = 0;
int Night_or_Day = 0;
int Score = 0;
int Sayac=0;

double currentTime = 0;
double lastTime = 0;
double getLastTime(){
    return currentTime = millis()/1000.0 - lastTime;
}

void _delay(float seconds) {
  long endTime = millis() + seconds * 1000;
  while(millis() < endTime) _loop();
}

void _loop() {
}

void loop() {
  _loop();
}

void setup() {
```

```
  pinMode(10,INPUT);
  pinMode(27,INPUT);
  pinMode(20,OUTPUT);

  Wire.begin();
  oled.init();
  oled.clearDisplay();

#if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
  clock_prescale_set(clock_div_1);
#endif

  oled.clearDisplay();
  oled.setTextXY(1,3);
  oled.putString("NIGHT and DAY");
  oled.setTextXY(2,7);
  oled.putString("GAME");
  oled.setTextXY(5,2);
  oled.putString("Press BUTTON!");
  oled.setTextXY(6,4);
  oled.putString("to START!");

  Score = 0;

  while(!(digitalRead(10) == 1))
  {
    _loop();
  }
  _delay(0.2);

  while(1){
  if (Sayac==0){

  delay(500);
  Change_Word();
  lastTime = millis()/1000.0;

  while(!(getLastTime() > 2))
  {
    _loop();
```

```
  if(analogRead(27) > 500){
     Gamer_Reaction = 0;
  }else{
     Gamer_Reaction = 1;
  }
}
digitalWrite(20,HIGH);
delay(250);
digitalWrite(20,LOW);

if(Night_or_Day == Gamer_Reaction){
    Correct();

}else{
    Wrong();

} _loop();

if(Score==100){
oled.clearDisplay();
oled.setTextXY(1,1);
oled.putString("Congratulation");
oled.setTextXY(3,1);
oled.putString("Your Score: ");
oled.setTextXY(3,13);
String String_Score=String(Score);
oled.putString(String_Score);
oled.setTextXY(5,3);
oled.putString("Press Reset");
oled.setTextXY(6,3);
oled.putString("To Repeat!");

  for(int i=0;i<3;i++){

  digitalWrite(20,HIGH);
  delay(500);
  digitalWrite(20,LOW);
  delay(500);
  } Sayac=1;
 }
}
```

```
 }
}

void Correct (){
  Score += 10;
  oled.clearDisplay();
  oled.setTextXY(3,4);
  oled.putString("10 points");
}

void Change_Word (){
  oled.clearDisplay();
  Night_or_Day=random(0,2);

  if (Night_or_Day==0){
    oled.setTextXY(3,6);
    oled.putString("NIGHT");
  }else{
    oled.setTextXY(3,7);
    oled.putString("DAY");
  }
}

void Wrong (){
  oled.clearDisplay();
  oled.setTextXY(1,3);
  oled.putString("Game Over");
  oled.setTextXY(3,1);
  oled.putString("Your Score: ");
  oled.setTextXY(3,13);
  String String_Score=String(Score);
  oled.putString(String_Score);
  oled.setTextXY(5,3);
  oled.putString("Press Reset");
  oled.setTextXY(6,3);
  oled.putString("To Repeat!");
  digitalWrite(20,HIGH);
  delay(1000);
  digitalWrite(20,LOW);
  Sayac=1;
}
```

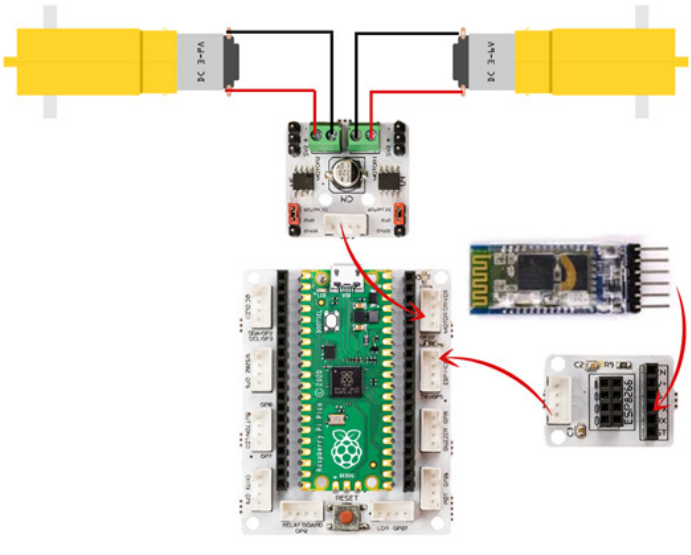
2.16. Ses Kontrollü Robot Araba

Gelişen ve gelişmeye devam eden yapay zeka uygulamaları insan özelliklerini tanımakta, öğrenmekte ve onun gibi davranmaya çalışmaktadır. Yapay zekayı en kısa haliyle öğrenebilen yazılımlar olarak ifade edebiliriz. Bazen görüntüyü bazen sesi bazen ise sensörlerden topladığı verileri kullanarak öğrenirler. Geliştiricilerinin belirlediği algoritmalar sayesinde bunu yaparlar ve elde ettikleri sonuçlara göre kullanıdıkları alanlarda karar verme süreçlerinde yardımcı olurlar. Kısaca karar verme sürecinin hızlı ve hatasız yapılması gereken durumlarda artık yapay zeka uygulamaları kullanılmaktadır. Pazarlama alanından savunma sanayisine, eğitimden sağlığa, ekonomiden eğlenceye her alanda yapay zeka verimi arttırıp maliyetleri düşürmektedir.

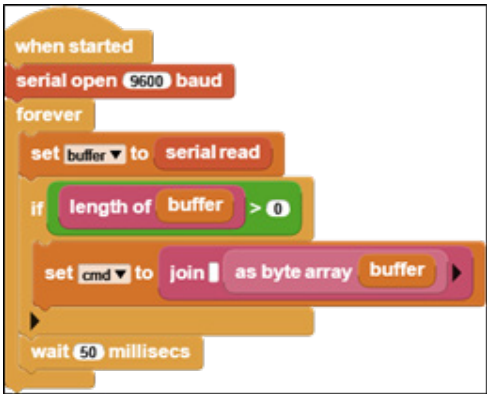
PicoBricks ile yapacağımız bu projede konuşarak kontrol edebileceğiniz 2WD araba yapacağız. PicoBricks 2 adet 6V DC motoru ve bluetooth ile kablosuz iletişim kurmanızı sağlamaktadır.

2.16.1. Proje Detayları ve Algoritma

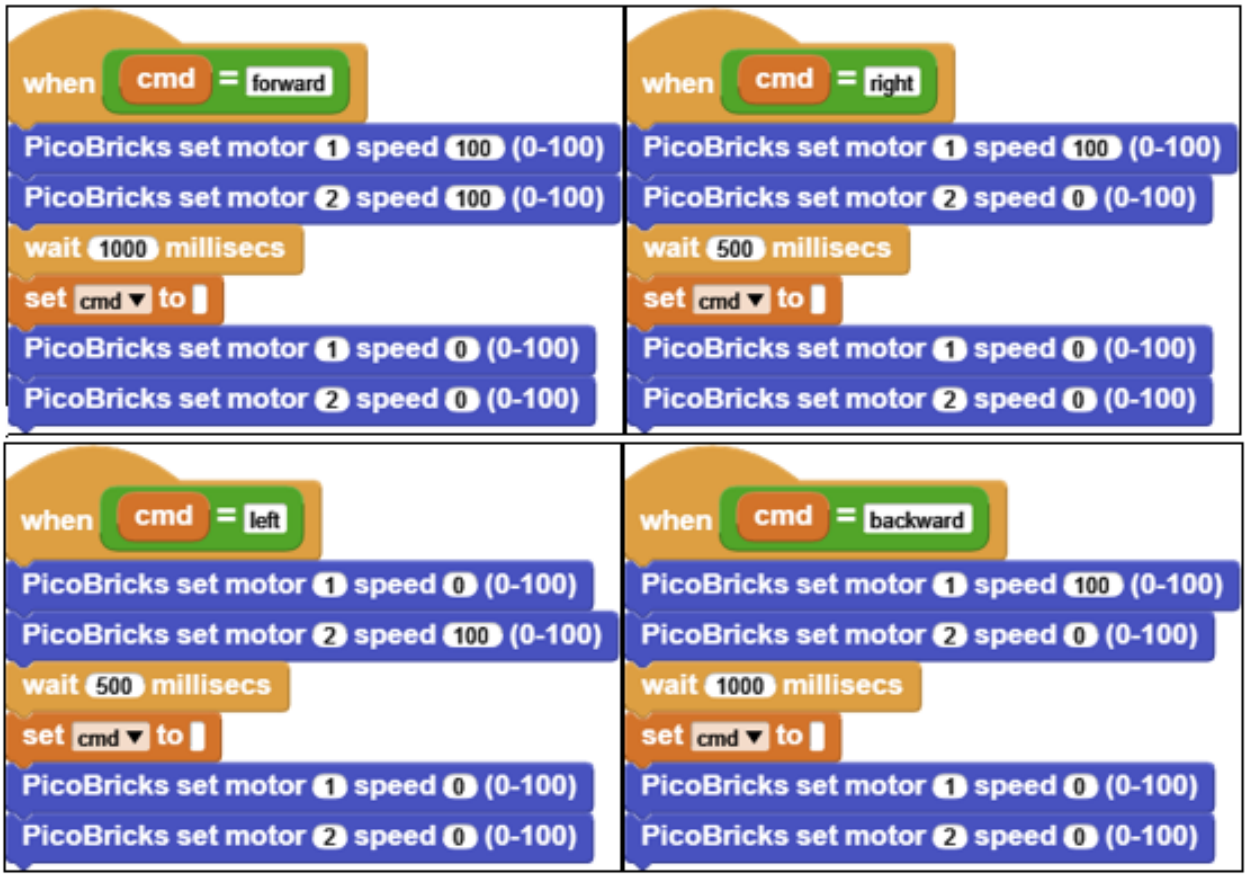
Projede setin içinden çıkan robot araba kiti montajlanarak cep telefonu üzerinden kontrol edilecektir. HC05 bluetooth modülü PicoBricks ile cep telefonu arasında kablosuz olarak iletişim kurabilmemizi sağlayan bir modüldür. Projede cep telefonuna yüklenen mobil uygulama sayesinde telefondan gönderilen komutlar HC05 modülü üzerinden PicoBrickse iletilecek ve bu verilere göre de robot araba hareket edecektir. Cep telefonundan ileri, geri, sağ, sol butonları ile robot arabayı yönlendirebileceğimiz gibi sesli komutla da PicoBrickse veriler gönderebiliriz. Projede robot arabanın hareketlerini kontrol etmek için sesli komutlar vereceğiz.

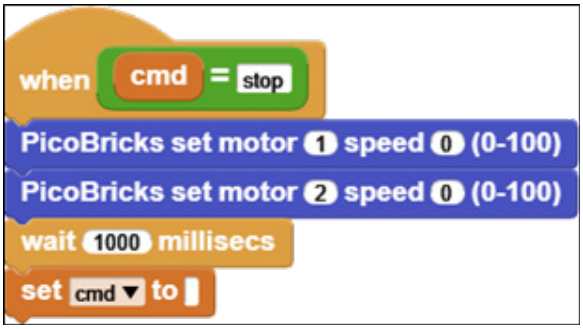


2.16.3. Projenin MicroBlocks ile Kodlanması



Projenin montajını tamamladıktan sonra kod kısmında ilk olarak cmd ve buffer adında iki değişken oluşturarak 9600 bant genişliğinde tanımlanan seri port okuma işlemini buffer değişkeni ile yapıyor ve gelen bilgileri cmd değişkenine aktarıyoruz. Eğer tüm kodları yazdığınız halde telefon ile PicoBricks arasında iletişim sağlamamıyorsan bant genişliğini 9600 yerine 115200 olarak değiştirmeyi deneyebilirsin.



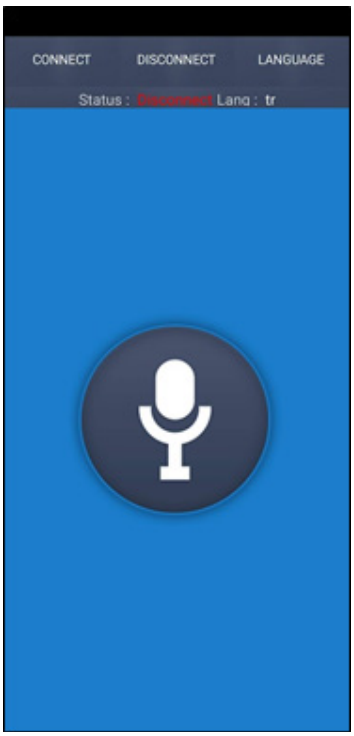


Projenin MicroBlocks kodlarına ulaşmak için tıkla.

HC05 modülü ile seri port üzerinden gelen bilgilerin tutulduğu cmd değişkeni ile when blokları kullanarak karşılaştırma işlemleri yapıyoruz. Gelen bilgi “forward” ise robot arabanın ileri doğru hareket edebilmesi için motor1 ve motor2 bloklarını 0 ile 100 değerleri arasında çalıştırmalıyız. Pilinizin doluluk durumu ile doğru orantılı olarak aracınızın hızı değişecektir. Aynı şekilde bluetooth üzerinden gelen bilgi “right” ise aracın sağa dönmesi “left” ise sola dönmesi, “backward” ise geri gitmesi, “stop” ise aracın durması için gerekli kodları yazıyoruz. Bu kodlarda aracın hareketleri belirli bir süre yapması sağlanmıştır. Dilersen süreleri uzatabilir ya da kaldırabilirsin.

Kodları yazarak PicoBricks üzerinde çalıştırdıktan sonra android cihazlar için aşağıdaki bağlantıdan mobil uygulamayı indirerek telefona kuruyoruz.

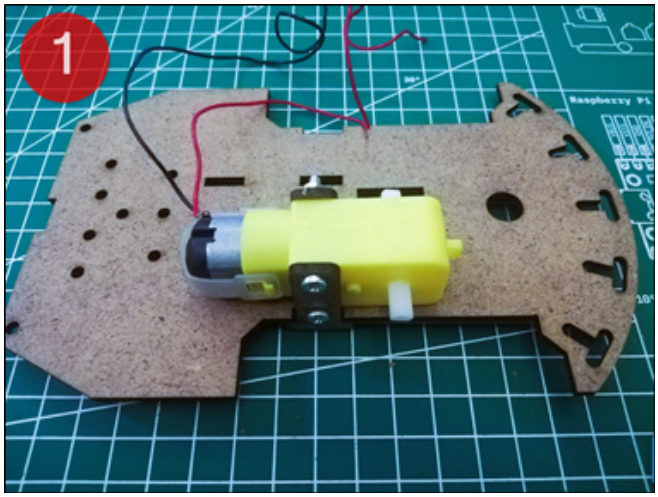
https://play.google.com/store/apps/details?id=appinventor.ai_cempehlivan92.Arduino_Sesli_Kontrol&hl=tr



Language butonu ile dil seçeneklerini açıp “en” seçeneği ile dili ingilizce “tr” seçeneği ile dili türkçe olarak düzenleyebilirsin.

2.16.4. Projenin Yapım Aşamaları

1: Setin içerisinden çıkan 2WD robot araba şasesine birinci motoru vidalayarak sabitle.

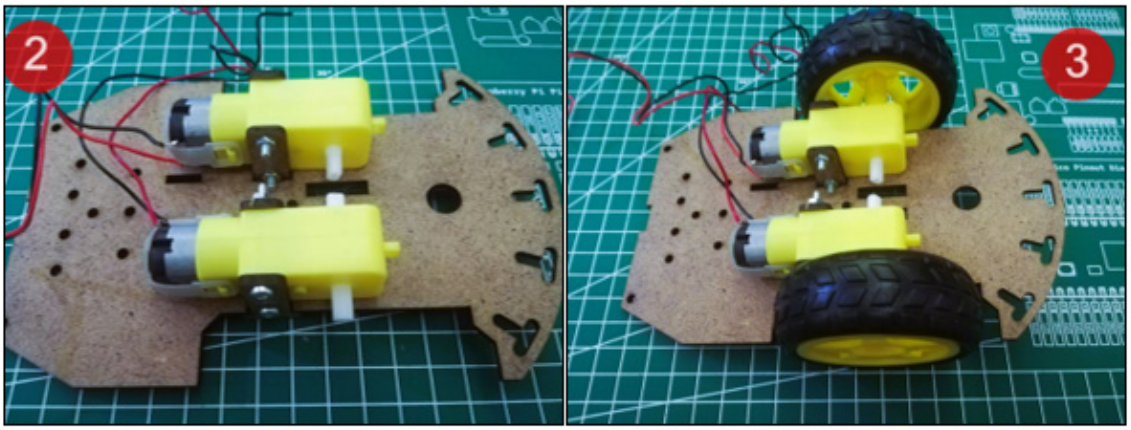


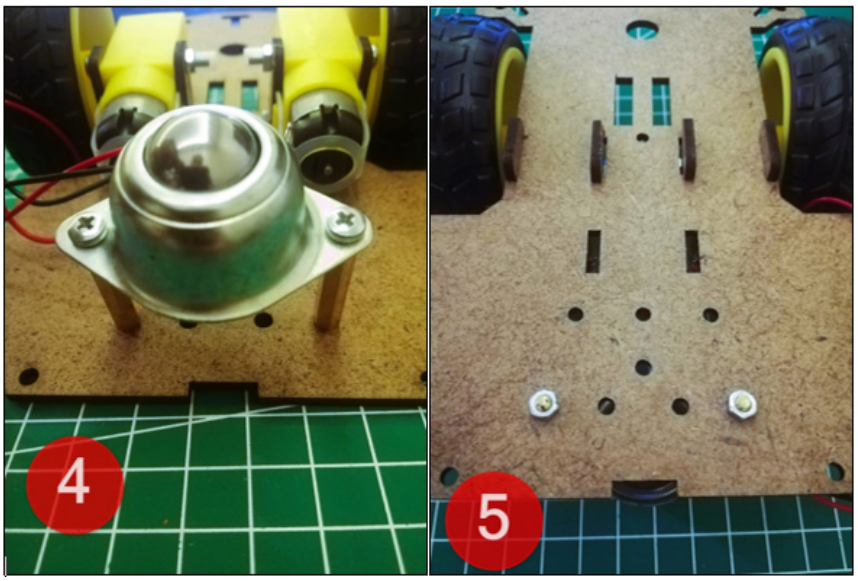
2: İkinci motoru şaseye vidalayarak sabitle.

3: Tekerleri motorlara tak

4: Şasenin altından sarhoş tekeri aralayıcı kullanarak sabitle

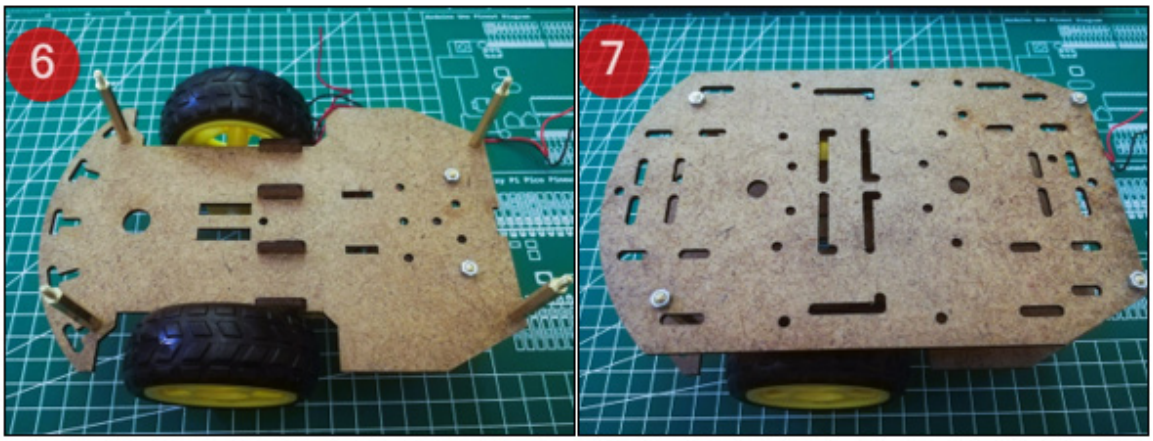
5: Şasenin üstünden aralayıcıyı somunla sabitle





6: 4 adet aralayıcı alt şasenin dört köşesine sabitle

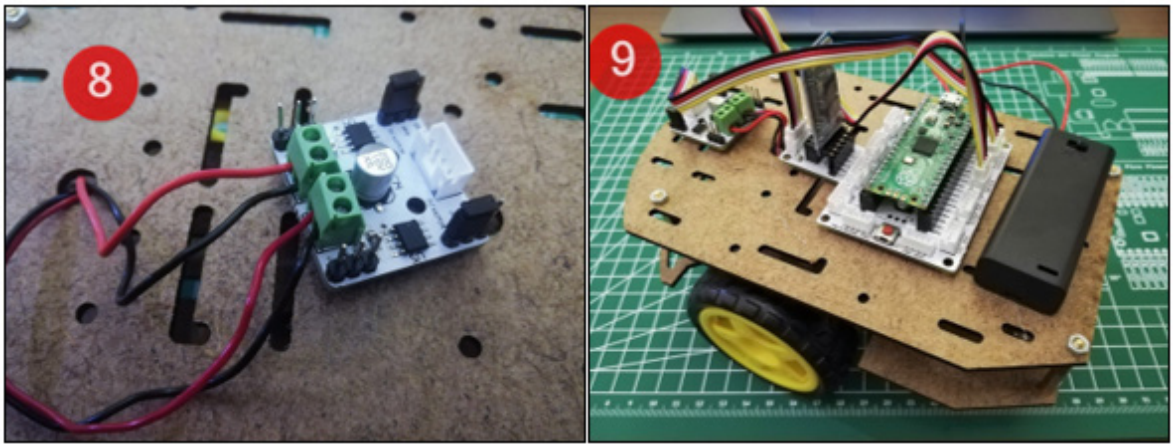
7: Üst şaseyi tak va somunlarla sabitle



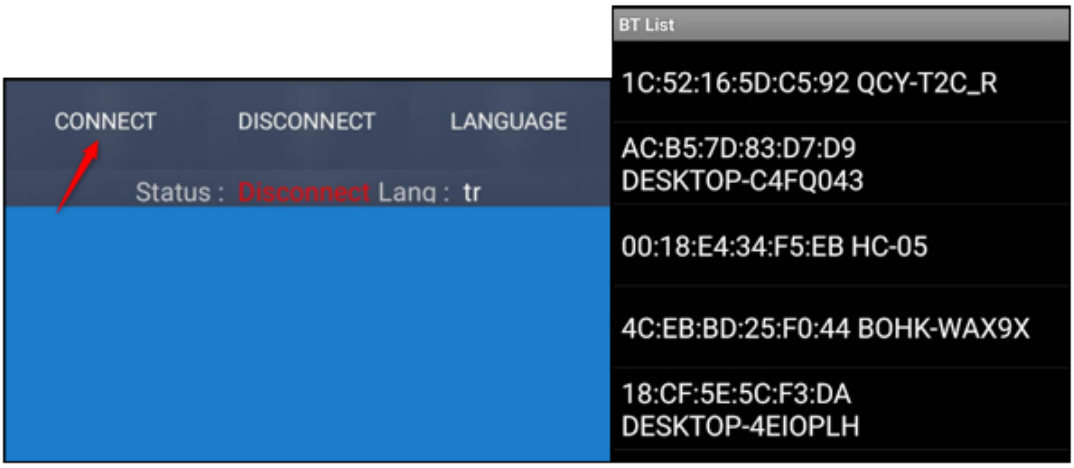
8: Motorların kablolarını motor sürücüdeki klemenslere tak.

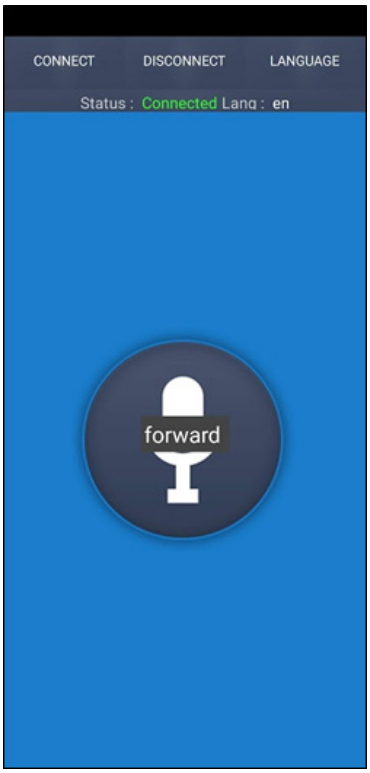
9: Motor sürücü, Bluetooth modülü, Picobricks board ve pil kutusunu şaseye sıcak silikon kullanarak sabitle.

Proje montajını, kodların yazılmasını ve mobil uygulama kurulumunu gerçekleştirdikten sonra şimdi robot arabanı sesli olarak kontrol edebilirsin. Bunun



için HC05 bluetooth modülü ile cep telefonunu eşleştirmelisin. Cihazının bluetooth ayarlarına girerek HC05 modülünü seçmeli ve eşleştirmelisin. Eşleştirme işleminde modül için gerekli parola “1234” ya da “0000” dır. Modül ile cihazı eşleştirdikten sonra uygulamayı açarak Connect butonuna tıkla ve HC05 modülüne seçerek bağlan.





Bağlantı gerçekleştiğinde Status bölümünde yeşil renkli Connected yazısını göreceksin. Mikrofon simgesine tıklayarak “forward”, “backward”, “right”, “left” komutları ile aracı hareket ettirebilirsin.

2.16.5. Proje Önerisi

Bu projede cep telefonuna yüklediğimiz mobil uygulama üzerinden sesli komutlar vererek robot arabayı hareket ettirdik. Sen de iki eksen robot kol projesindeki pan-tilt mekanizmasına HC05 bluetooth modülü bağlayarak sesli komutlar ile ya da butonlar ile mekanizmayı kontrol edebilirsin. Aynı şekilde bu projedeki robot arabayı sesli komutlar yerine butonlar kullanarak kontrol edebileceğin bir mobil uygulama deneyebilir ya da MIT Appinventor editörü ile projene özel mobil uygulama geliştirebilirsin.

HC05 Bluetooth modülü ile sadece motor sürücü ve motor değil PicoBricks üzerindeki diğer modülleri de çalıştırabilirsin. Örneğin, mobil uygulama üzerinden RGB ledi dilediğin renkte yakabilir, DHT11 modülünden sıcaklık ve nem değerlerini, LDR sensör üzerinden ışık değerlerini okuyabilir, OLED ekrana yazılar yazdırabilirsin. MIT Appinventor editörü ile bu işlemler için özel olarak yazılmış bir mobil uygulama ve uygulamadan gelen verilerin otomatik olarak çalışması için Microblocks ile yazılmış hazır kodlar var. Aşağıdaki linkten Microblocks dosyasını indirip çalıştırarak ve android apk dosyasını indirip telefonuna kurarak tüm bu özellikleri çalıştırabilirsin.

https://drive.google.com/file/d/11Qg6AwgKb9OH6EtXWw5mDEEhwdHA9MSp/view?usp=drivesdk

2.16.6. Projenin MicroPython Kodları

```
from machine import Pin, UART
from utime import sleep
uart = UART(0,9600) #If connection cannot be established, try 115200.
m1 = Pin(21, Pin.OUT)
m2 = Pin(22, Pin.OUT)
m1.low()
m2.low()
while True:
  sleep(0.05)
  if uart.any():
    cmd = uart.readline()
  if cmd==b'F':
    m1.high()
    m2.high()
  elif cmd==b'B':
    m1.high()
    m2.low()
  elif cmd==b'R':
    m1.high()
    m2.low()
  elif cmd==b'L':
    m1.low()
    m2.high()
  elif cmd==b'S':
    m1.low()
    m2.low()
  cmd=""
```

2.16.7. Projenin Arduino C Kodları

```
void setup() {
 Serial1.begin(9600);
}

void loop() {
 if (Serial1.available() > 0) {

    char sread = Serial1.read();
    Serial.println(sread);

    if (sread == 'f') {
    Forward();
  } else if(sread == 'b'){
    Backward();
  } else if(sread == 'r'){
    Turn_Right();
  } else if(sread == 'l'){
    Turn_Left();
```

```
    } else if(sread == 's'){
      Stop();
    }
  }
}

void Forward(){
  digitalWrite(21,HIGH);
  digitalWrite(22,HIGH);
  delay(1000);
  digitalWrite(21,LOW);
  digitalWrite(22,LOW);
}
void Backward(){
  digitalWrite(21,HIGH);
  digitalWrite(22,LOW);
  delay(1000);
  digitalWrite(21,LOW);
  digitalWrite(22,LOW);
}
void Turn_Left(){
  digitalWrite(21,LOW);
  digitalWrite(22,HIGH);
  delay(500);
  digitalWrite(21,LOW);
  digitalWrite(22,LOW);
}
void Turn_Right(){
  digitalWrite(21,HIGH);
  digitalWrite(22,LOW);
  delay(500);
  digitalWrite(21,LOW);
  digitalWrite(22,LOW);
}
void Stop(){
  digitalWrite(21,LOW);
  digitalWrite(22,LOW);
  delay(1000);
}
```

Arduino kodlarını yükledikten sonra aşağıdaki linkten android uygulamayı indirerek terminal modunu aç ve araç hareketleri için f,b,r,l,s harflerini kullan.

https://play.google.com/store/apps/details?id=com.giumig.apps.bluetoothserialmonitor&hl=tr

2.17. İki Eksen Robot Kol

Robot kollar endüstriyel alanda insan gücünün yerini almıştır. Bir insanın taşıyamayacağı ağırlık ve büyüklükteki yükleri taşıma ve döndürme işlerini fabrikalarda robot kollar üstlenmektedir. Milimetrenin binde biri hassasiyetinde konumlandırılabilmeleri de insan elinin sergileyebileceği hassasiyetin üzerindedir. Otomobil fabrikalarının üretim videolarını izlediğinde robot kolların ne kadar hayati bir öneme sahip olduğunu göreceksin. Robot denmesinin sebebi programlanarak sonsuz tekrarla aynı işi yapabilmelerinden kaynaklanmaktadır. Kol denmesinin sebebi ise bizim kollarımız gibi eklemli bir yapıya sahip olmasından kaynaklanmaktadır. Bir robot kolun kaç farklı doğrultuda dönme ve hareket etme kabiliyeti varsa o kadar eksenli olarak ifade edilmektedir. Robot kollar alüminyum ve çeşitli metalleri oyma ve şekil verme işlerinde de kullanılmaktadır. 7 eksenli CNC Router olarak geçen bu cihazlar bir heykeltraşın çamura şekil vermesi gibi metallere şekil verebilmektedirler.

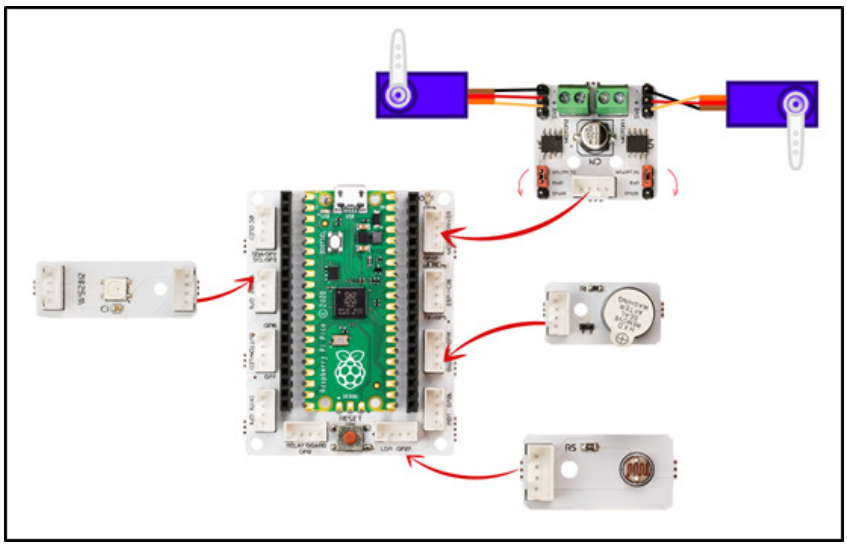
Robot kollarda kullanılma amacına göre bir tür elektrik motoru olan step motor ve servo motorlar kullanılmaktadır. PicoBricks servo motorlarla projeler yapmanıza olanak sağlamaktadır.

2.17.1. Proje Detayları ve Algoritma

Kuruluma hazırlık amacıyla öncelikle servo motorları 0 dereceye ayarlamak için kodlarını yazıp yükleyeceğiz. LDR sensörünün üzerine bir cisim konulduğunda robot kol aşağı eğilecek ve açık olan kıskacını kapatacak. Kıskaç kapandıktan sonra robot kol tekrar yukarı kalkacak. Robot kolun her hareketinin sonucunda buzzer dan kısa bir bip sesi çıkacak. LDR sensörünün üzerine cisim yerleştirildiğinde RGB LED kırmızı renkte yanacak. Cisim robot kol tarafından tutulup havaya kaldırıldığında ise RGB LED yeşil renkte yanacak.

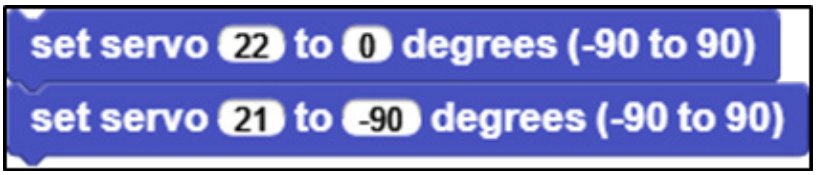
Servo motorun hareketleri çok hızlıdır. Hareketi yavaşlatmak için 30 milisaniye aralıklarla 2 şer derece toplamda 90 derece hareket ile servo motorları kodlayacağız. Kıskacın kapanması için bunu yapmayacağız.

Servonun tutma ve bırakma işlevini yerine getirebilmesi için buradaki bağlantıdan gerekli parçaları 3B yazıcıdan basıp montajlayın.

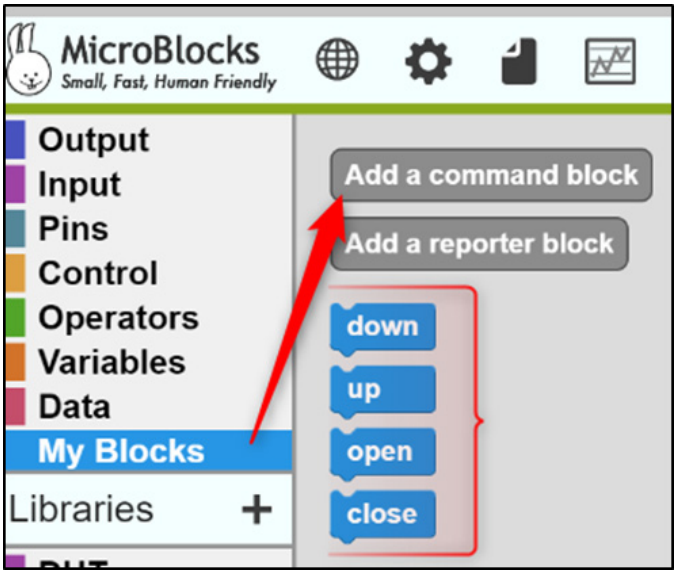


2.17.3. Projenin MicroBlocks ile Kodlanması

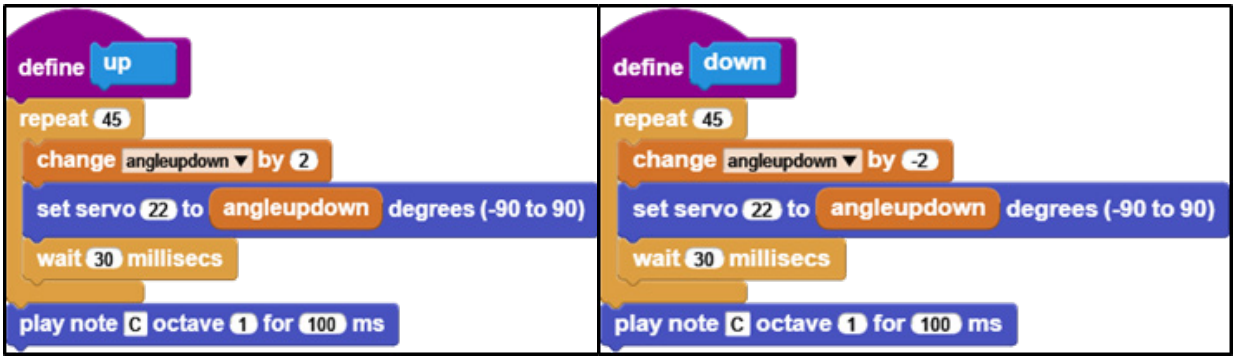
Microblocks u açıp Picobricks'e bağlan. Servo kütüphanesini ekle. Tutma motoru 21 nolu pine, eğilme motoru 22 nolu pine bağla. Robot kolun tutma ucunun servo değerini -90 a ayarla. Eğilme motorunun açısını 0 a ayarla. Aşağıdaki blokları hazırlayıp üzerine tıklayınca kodlar çalışacak ve servolar ayarladığın açıya dönecekler..



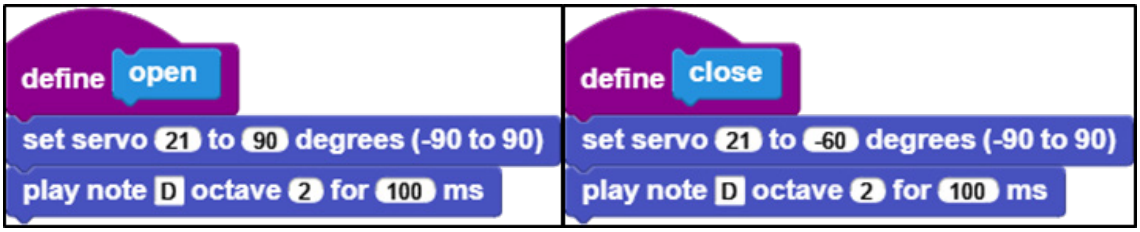
Servo motor hareketlerini ayrı bloklar halinde hazırlayacağız. Bunun için My Blocks kategorisinden up, down, open, close isimli dört command bloğu oluştur.



Yukarı ve aşağı kol hareketi için up ve down bloklarını 45 tekrarda 30'ar milisaniyede 2 şer derece dönecek şekilde kodlayın. Açı değerini değiştirebilmek için angleupdown isimli değişken oluşturun.down bloğunda -2 derece, up bloğunda 2 derece değişim yapılacak.

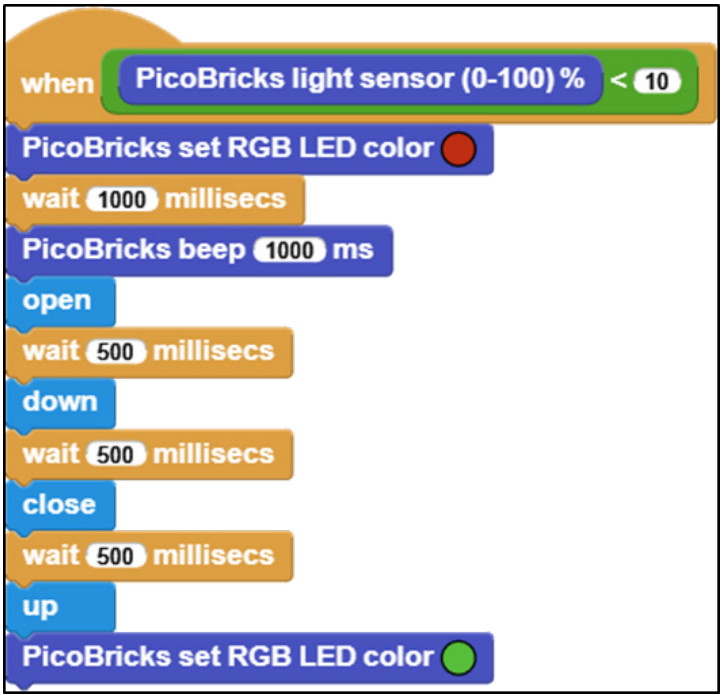


Tutucu servoyu açık hale getirmek için açısını 90 kapalı hale getirmek için -60 olarak kodla. Tüm servo hareketlerinin sonuna Tone kütüphanesini ekleyerek dilediğin bir ses ekle. 100 ms çalması yeterlidir.



Picobricks başladığında tutucu servo açık konuma gelmeli, angleupdown değişkeni başlangıç değeri olan 90 olmalı ve robot kol yukarı bakacak şekilde 90 dereceye dönmeli. RGB LED sönmeli.

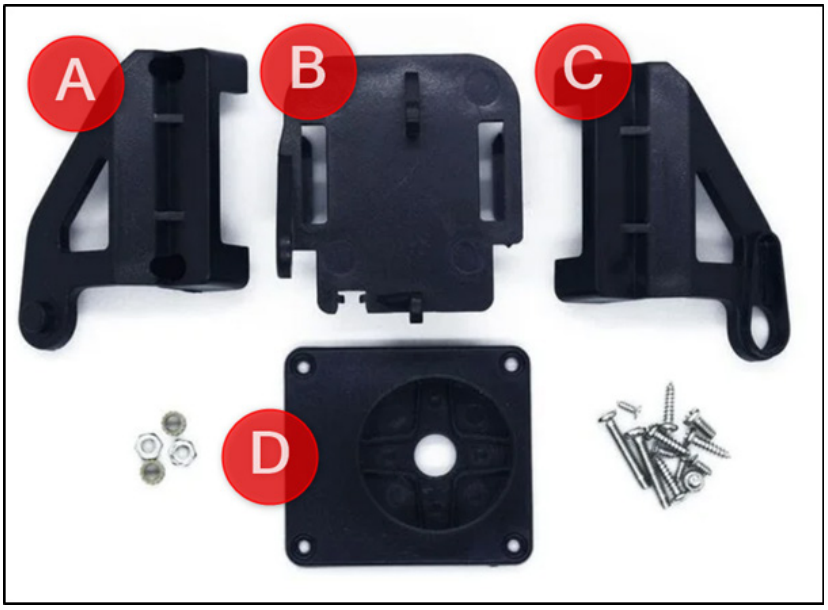
LDR sensörünün üzerine bir cisim geldiğini değeri 10 un altında düştüğünde anlarız. Sizde bu değer farklı olabilir. Bloğun üzerine tıklayarak kaç okuduğunu öğrenebilirsin. Önce RGB LED kırmızı renkte yanar. Cisimi tutup kaldırmak için tutma motoru açılır , aşağı hareket yapılır, tutma motoru kapatılır ve yukarı hareketi gerçekleşir. Son olarak RGB LED yeşil renkte yakılır. Her hareket arasına 500 ms bekleme koyarak hareketin daha rahat izlenmesini sağlayabilirsin.

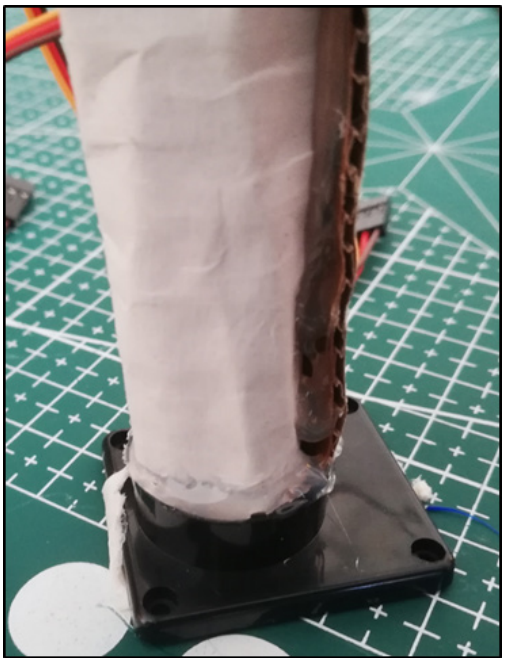


Projenin kodlarına erişmek için tıkla.

2.17.4. Projenin Yapım Aşamaları

Projeyi hazırlamak için Pan-Tilt kitinin parçalarını hazırla. 3B yazıcıda basılmış parçaları, Atık karton parçaları , sıcak silikon yapıştırıcı ve makası da yanında bulundur.

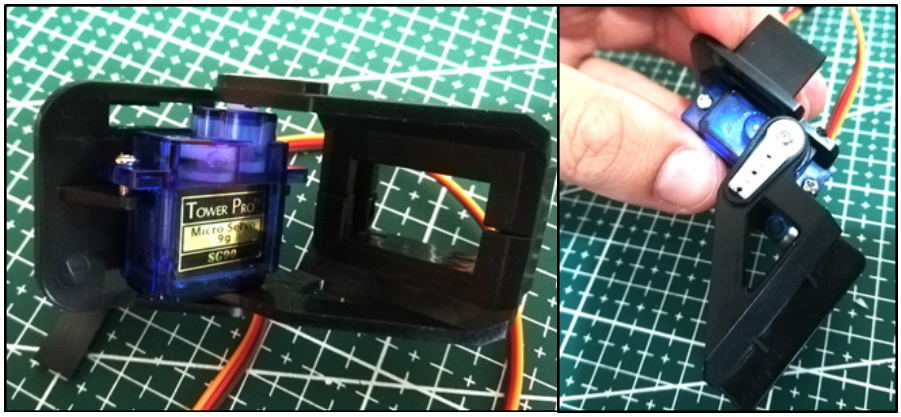




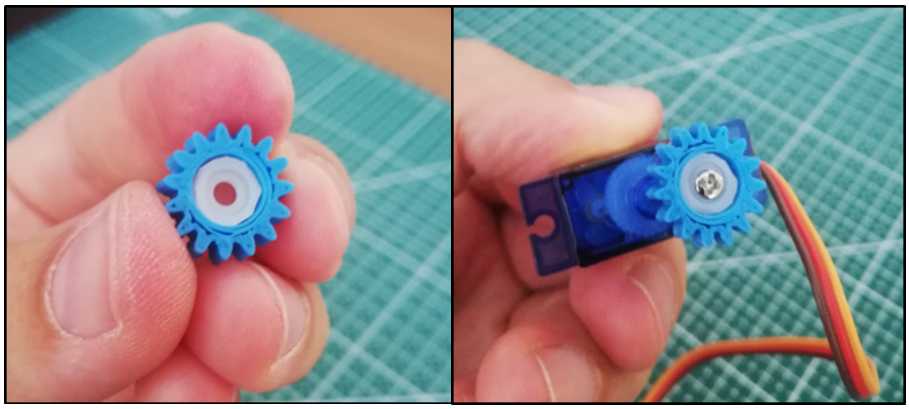
1.Öncelikle robot kolun sabit kolunu hazırlayacağız. D parçasının yuvarlak kısmına 8 cm yüksekliğinde kartondan silindir yapın. D parçasına yerleştirip silikon ile yapıştır.

2.C parçasına servo motor paketinden çıkan başlığı biraz kısaltarak yerleştir. Pan Tilt kitinden çıkan en küçük vidalar ile sabitle.

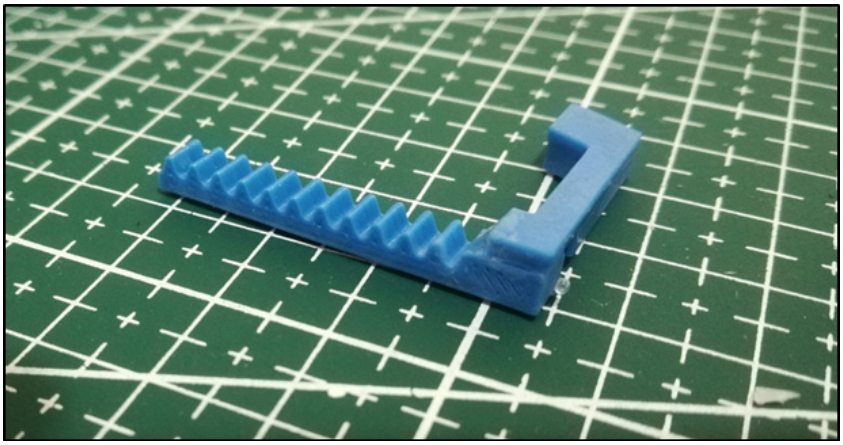
3.A ve C parçalarını sivri uçlu 2 adet vida ile birbirine sabitle..



4.Servo motoru C parçasına içten tak. Ardından B parçasınına servo motoru yerleştirip vidala

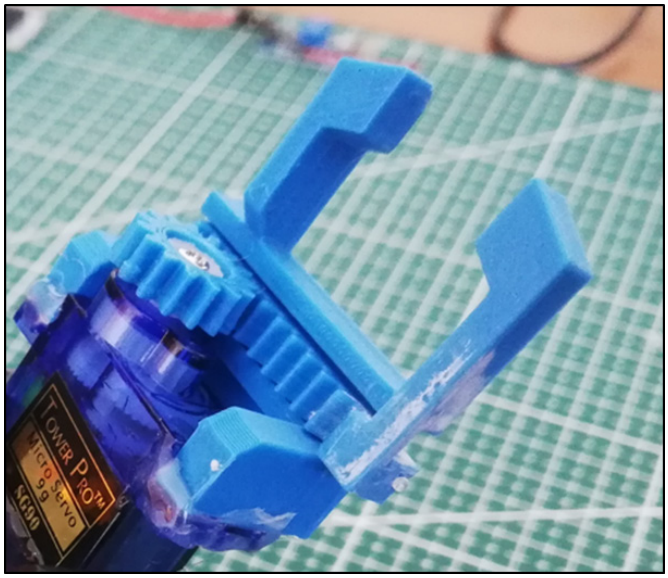


5.Tutucu için 3B yazıcıda bastığın dişli parçasını ortasına servo motor başlıklarından birini keserek dişlinin içine yerleştir. Ardından servo motora vidala.

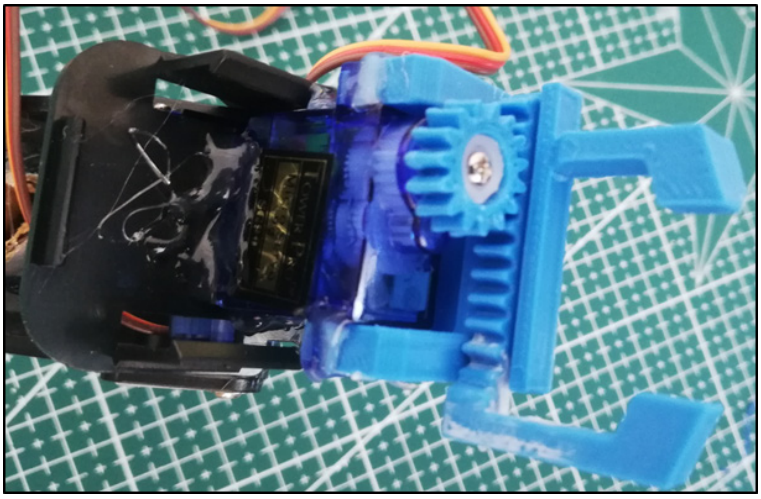


6. Adhere together the 3D printed Linear gear and the handle with strong adhesive.

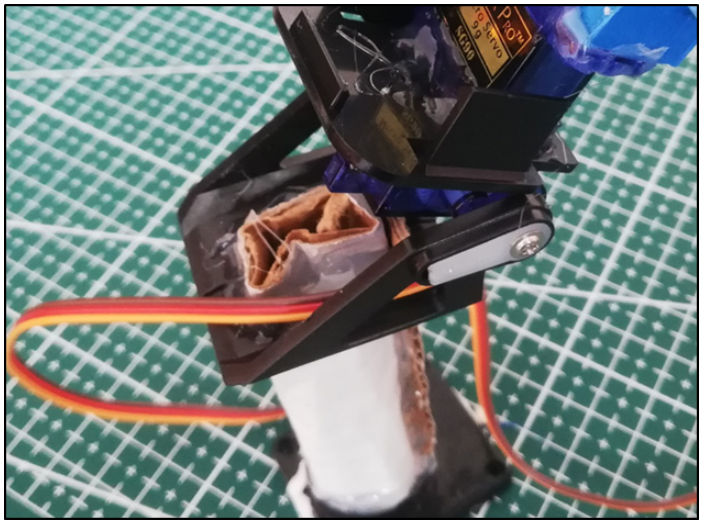
7.Servoyu 3B baskı tutucuya yerleştirip sabitle. İster sıcak silikon ile istersen vidalayarak bunu yapabilirsin. Lineer dişliye servo dişlisini yerleştirirken tam açık halde olmasına dikkat et.



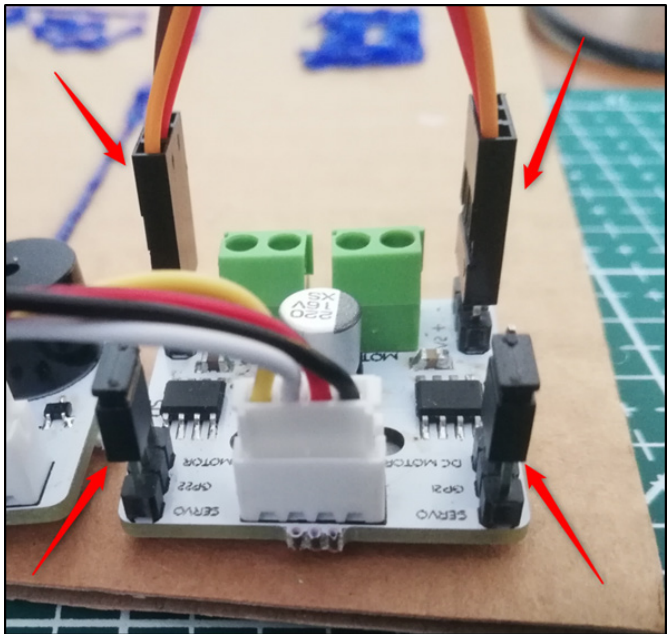
8. Tutma servosu sistemini B parçasına silikon ile yapıştır.



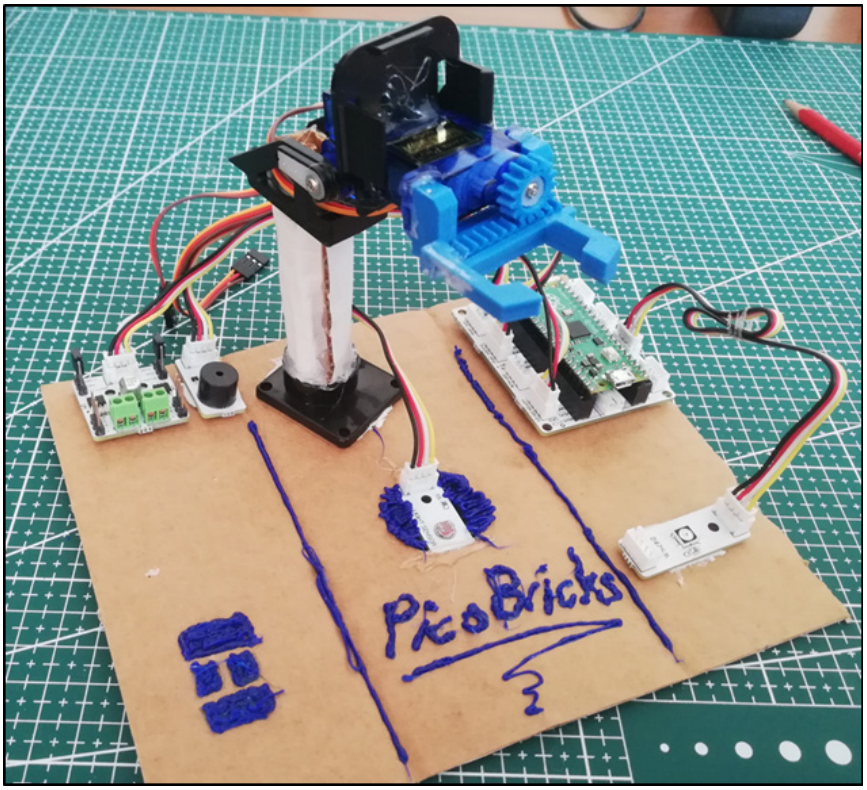
9. 3. adımda hazırladığın parçayı ilk adımda kartondan hazırladığın silindirin üstünden geçirip silikon ile sabitle.



10. Motor sürünün jumperlarını Servo pinlerine tak. Tutucu servonun kablosunu GPIO21'e eğilme servosunun kablosunu GPIO22 ye tak.



11. Motor sürücü, buzzer, LDR ve RGB LED modülünü bir platforma yerleştirip robot kolu uygun şekilde platforma yerleştir. 3B Kalem yazıcı ile projeni dilediğin gibi özelleştirebilirsin.



12. Picobricks'i USB ile ya da 3'lü kalem pil ile Picoboard'da bulunan power jak'ından beslersen Robot kolu çalıştırabilirsin.

2.17.5. Proje Önerisi

2 eksen robot kol projesine HC05 modülü ekleyerek mobil uygulama ile cep telefonundan kontrol ederek geliştirebilirsin.

2.17.6. Projenin MicroPython Kodları

Servo motorları kalibre eden kodlar:

```
from machine import Pin, PWM
servo1=PWM(Pin(21))
servo2=PWM(Pin(22))

servo1.freq(50)
servo2.freq(50)

servo1.duty_u16(8200) # 180 degree
servo2.duty_u16(4770) # 90 degree
```

Project Codes:

```
from machine import Pin, PWM, ADC
from utime import sleep
from ws2812 import NeoPixel


neo = NeoPixel(6, n=1, brightness=0.3, autowrite=False)
ldr=ADC(27)
buzzer=PWM(Pin(20, Pin.OUT))
servo1=PWM(Pin(21))
servo2=PWM(Pin(22))


servo1.freq(50)
servo2.freq(50)
buzzer.freq(440)
```

```
RED = (255, 0, 0)
GREEN = (0, 255, 0)
BLACK = (0, 0, 0)
angleupdown=4770



def up():
    global angleupdown
    for i in range (45):
        angleupdown +=76
        servo2.duty_u16(angleupdown)
        sleep(0.03)
    buzzer.duty_u16(2000)
    sleep(0.1)
    buzzer.duty_u16(0)


def down():
    global angleupdown
    for i in range (45):
        angleupdown -=76
        servo2.duty_u16(angleupdown)
        sleep(0.03)
    buzzer.duty_u16(2000)
    sleep(0.1)
    buzzer.duty_u16(0)


def oppen():
    servo1.duty_u16(8200) #180 degree
    buzzer.duty_u16(2000)
    sleep(0.1)
    buzzer.duty_u16(0)
```

```
def close():
    servo1.duty_u16(2490) #30 degree
    buzzer.duty_u16(2000)
    sleep(0.1)
    buzzer.duty_u16(0)


oppen()
servo2.duty_u16(angleupdown)
neo.fill(BLACK)
neo.show()
while True:
    if ldr.read_u16()>40000:
        neo.fill(RED)
        neo.show()
        sleep(1)
        buzzer.duty_u16(2000)
        sleep(1)
        buzzer.duty_u16(0)
        oppen()
        sleep(0.5)
        down()
        sleep(0.5)
        close()
        sleep(0.5)
        up()
        neo.fill(GREEN)
        neo.show()
        sleep(0.5)
```

2.17.7. Projenin Arduino C Kodları

```
#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
#include <avr/power.h>
#endif
#define PIN       6
#define NUMPIXELS 1
Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_GRB + NEO_KHZ800);
#define DELAYVAL 500

#include <Servo.h>

Servo myservo1;
Servo myservo2;


int angleupdown;


void setup() {

  pinMode(20,OUTPUT);
  pinMode(27,INPUT);

  pixels.begin();
  pixels.clear();

  myservo1.attach(21);
  myservo2.attach(22);


}
```

```
void loop() {

  Open();
  angleupdown=180;
  myservo2.write(angleupdown);
  pixels.setPixelColor(0, pixels.Color(0, 0, 0));
  pixels.show();

  if(analogRead(27)>500){

    pixels.setPixelColor(0, pixels.Color(255, 0, 0));
    pixels.show();
    delay(1000);
    tone(20,700);
    delay(1000);
    noTone(20);

    Open();
    delay(500);
    Down();
    delay(500);
    Close();
    delay(500);
    Up();
    pixels.setPixelColor(0, pixels.Color(0, 255, 0));
    pixels.show();
    delay(10000);
  }
}

void Open(){
  myservo1.write(180);
}

void Close(){
  myservo1.write(30);
}
```

```
void Up(){

  for (int i=0;i<45;i++){

    angleupdown = angleupdown+2;
    myservo2.write(angleupdown);
    delay(30);
    }
}

void Down(){

  for (int i=0;i<45;i++){

    angleupdown = angleupdown-2;
    myservo2.write(angleupdown);
    delay(30);
    }
}
```

2.18. Akıllı Ev

İşyerleri, fabrikalar, evlerimiz hatta hayvan barınakları... Yaşam alanlarımızı istemediğimiz davetsiz misafirlere karşı korumak için kullanılabilecek farklı elektronik sistemler bulunmaktadır. Bu sistemler ev ve iş yeri güvenlik sistemleri olarak üretilmekte ve pazarlanmaktadır. Güvenlik kameralarının ürettiği görüntülerin işlenip yorumlandığı sistemler bulunduğu gibi insan bedenini ve hareketlerinin sensörler ile tespit edip harekete geçen güvenlik sistemleri de mevcuttur. Güvenlik sistemleri bir tür alarmlı saat gibi kurulur ve belirlenen zaman diliminde tanımlanmayan bir hareketlilik algılandığında sesli ve ışıklı uyarılar verir. İşletme veya ev sahibine bildirimde bulunur ayrıca güvenlik birimlerine de otomatik bildirimlerde bulunabilmektedir.[Yakalanmış hırsız karikatürü ve güvenlik logosu kullanılabilir.]

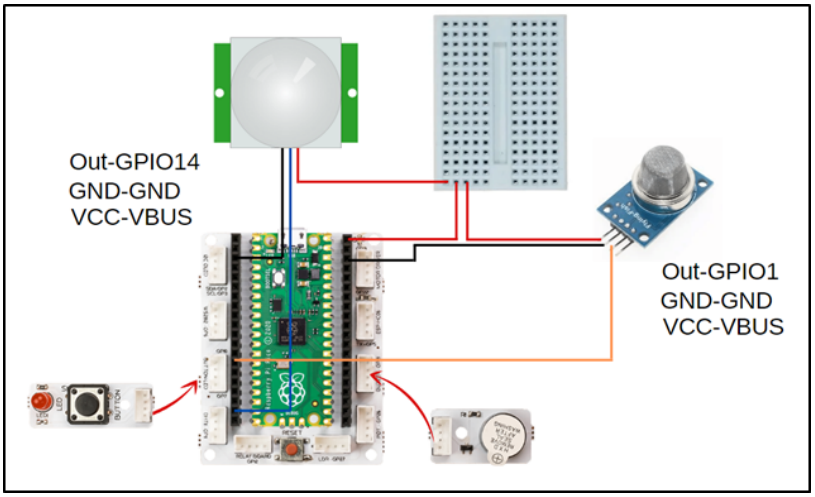
Gaz kaçağı yangın vb. durumlarda zehirlenmeleri önlemek için ev ve işyerlerinde gaz sensörleri kullanılır. Olumsuz bir durumda yüksek sesle alarm vererek ortamda yaşayan insanlar uyarılır.

PicoBricks ile HC-SR501 ve MQ-2 gaz sensörü kullanarak maket bir akıllı ev projesi hazırlayacağız. Bu sensör HC-SR501 , PIR sensörü olarak da ifade edilmekte, insan vücudunun yansıttığı kızılötesi dalgaların değişimlerini yakalayarak hareketi tespit etmektedirler.

2.18.1. Proje Detayları ve Algoritma

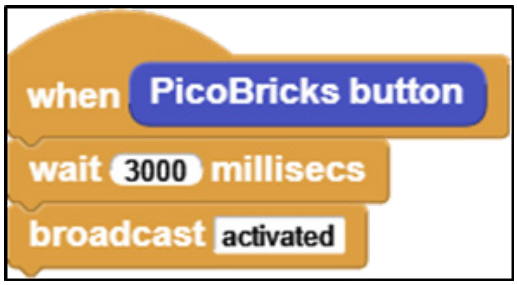
HC-SR501 PIR sensörü hareket algıladığında 3 saniye boyunca dijital çıkış vermektedir. Proje maketinde Picoboard, buzzer ve buton LED modülünü kullanacağız. Tüm parçalar maketin içinde olmalı. Picobricks başladığında alarm sisteminin devreye girmesi için butona basılması gerekmektedir. Butona basıldıktan sonra elin maketi içinden çekilmesi için 3 saniye bekleme koymalıyız. 3 saniyenin sonunda kırmız LED yanar ve alarm sistemi devreye girer. Alarm sistemi devredeyken bir hareket algıladığında kırmızı LED yanıp sönmeye başlar ve buzzer'dan alarm sesi çalınır. Susturmak için Picobricks'in yeniden başlatılması gerekmelidir. MQ-2 sensörü ise sürekli devrededir. Zehirli bir gaz algıladığında ise buzzer ve kırmızı LED ile bunu bildirecektir.

2.18.2. Bağlantı Şeması

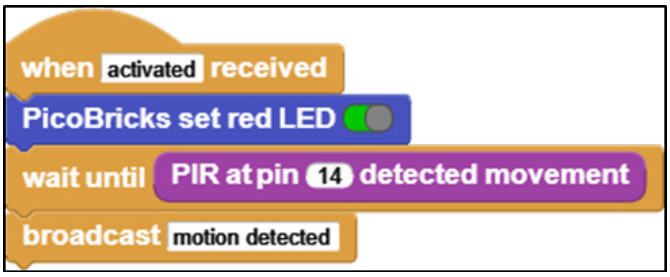


2.18.3. Projenin MicroBlocks ile Kodlanması

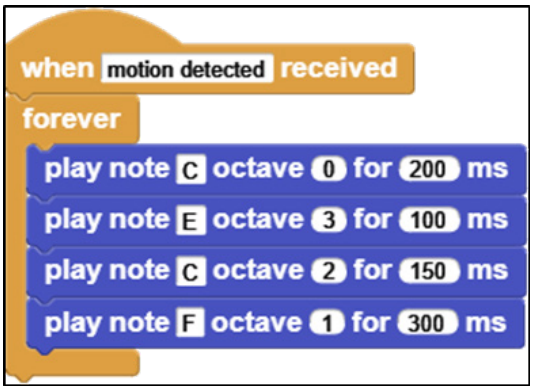
Alarm sisteminin aktifleşmek için butona basılmasını bekler. Butona basıldıktan 3 saniye sonra activated yayını yapılır.

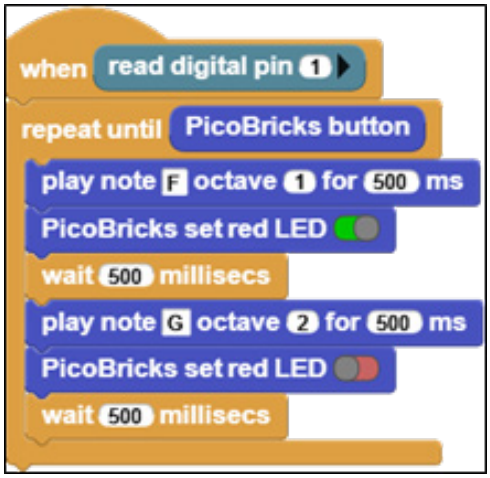


Alarm sistemi aktif hale geldiğini kırmızı LED'i yakarak belirt. Library>Sensing>PIR.ubl yolunu izle ve PIR sensörünün bloğunu MicroBlocks'a ekle. Pin numarasını 14 olarak değiştir. Hareket algılayana kadar kodun işletilmesini beklet. Hareket algılandığında motion detected yayını yap.



motion detected yayını alındığında sürekli olarak buzzerdan alarm sesi çıkar. Alarm sesini Tone.ubl kütüphanesini ekleyerek dilediğin gibi kişiselleştirebilirsin.



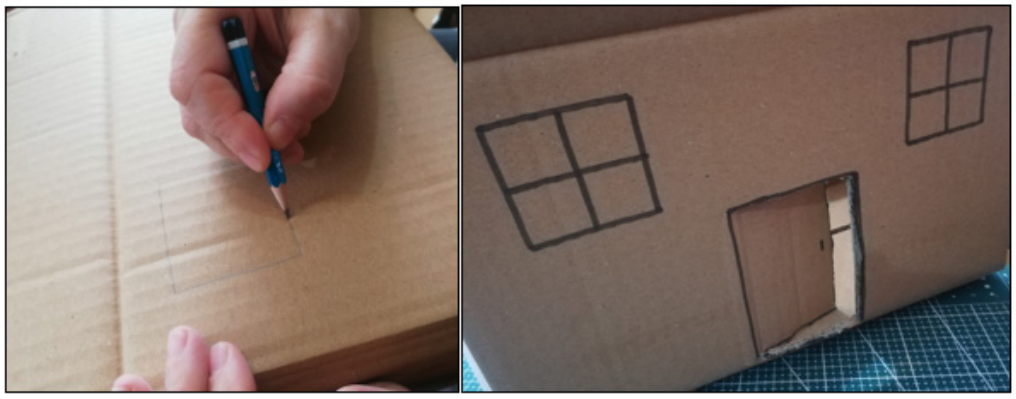


MQ-2 sensörü gaz algıladığında dijital çıkış vermektedir. Dijital çıkış verdiğinde PicoBricks'in butonuna basılana kadar alarm çalmasını ve kırmızı LED'in yanıp sönmesini sağlıyoruz.

Projenin kodlarına erişmek için tıkla.

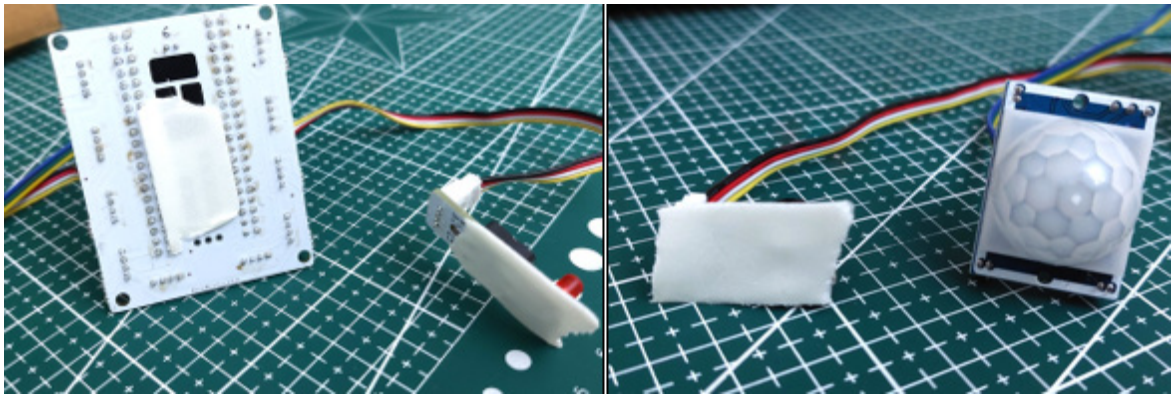
2.18.4. Projenin Yapım Aşamaları

Projeyi çalıştırmak için karton bir kutuyu maket ev haline getirmelisin. Makas, kalem, bant, yapıştırıcı, maket bıçağı lazım olacaktır. Kutunun üzerine pencere ve kapıları kurşun kalemle çizin.Kapı bölümünü maket bıçak ile kes.

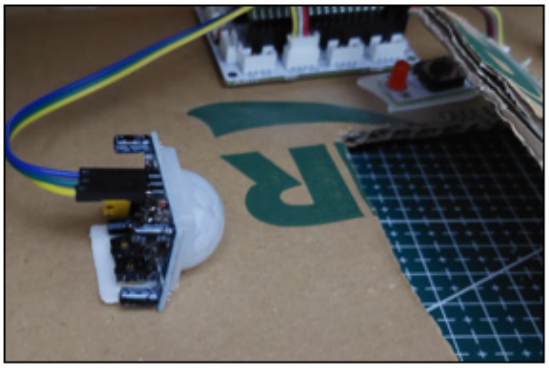


Çatı kısmını yapmak için başka bir kartondan faydalanabilirsin.

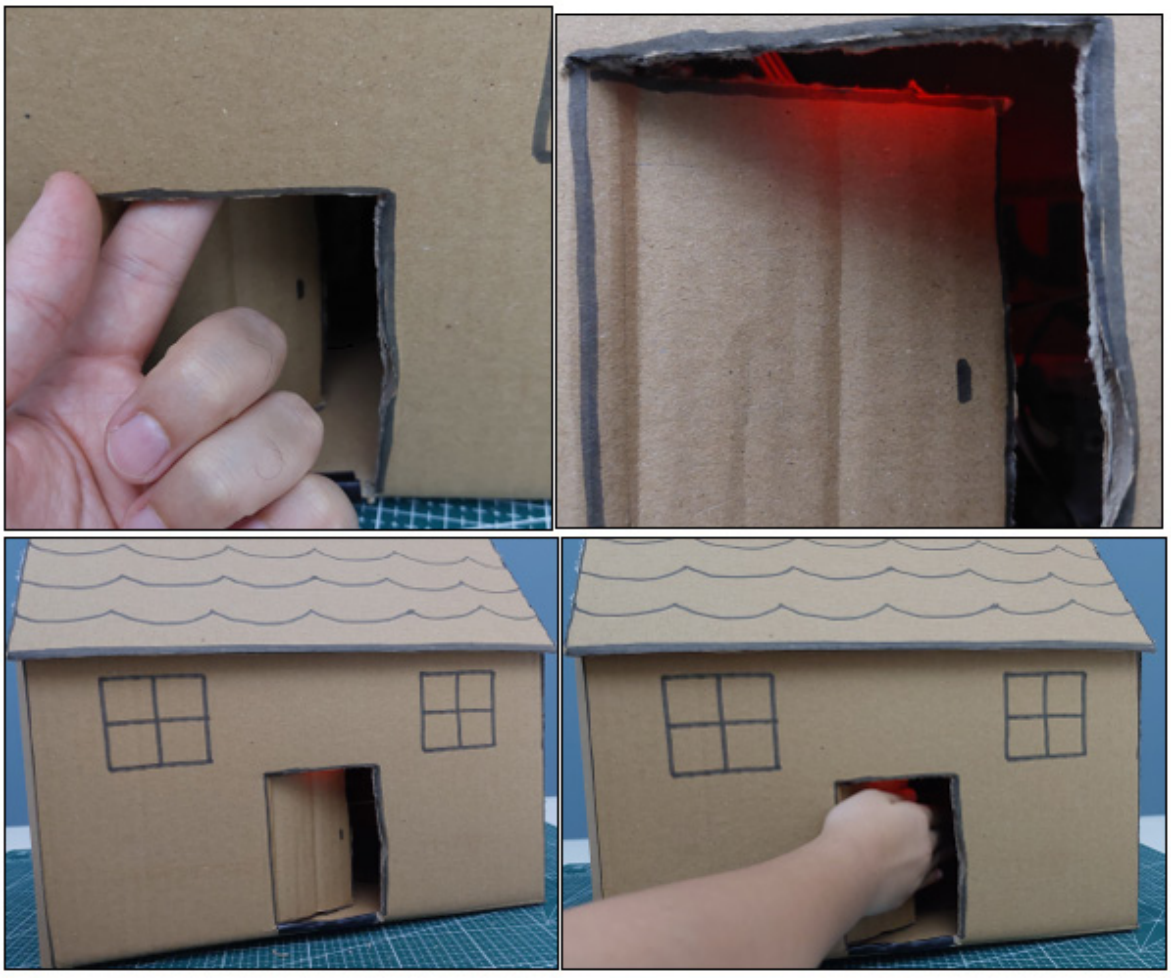
Picobricks parçalarının altına çift taraflı köpük bant yapıştır.



Picobricks'in parçalarını maket evinin içine yerleştir. PIR sensörün içeriden kapıyı doğrudan görecek şekilde konumlandır. Buton modülünü ise içeriden kapının hemen üstüne yapıştır.




Pil kutusunu Picoboard a bağlayıp açtığında kodlar çalışmaya başlayacak. Butona bastıktan 3 saniye sonra alarm sistemi devreye girecek ve kırmızı LED yanacak. Elini kapıdan içeri sokar sokmaz buzzer alarm çalmaya başlayacaktır.



Çakmak gazını evin içine tuttuğunuzda alarm sisteminin yine devreye girmesi beklenir.

2.18.5. Proje Önerisi

25. proje olan akıllı serayı yaptıktan sonra akıllı ev projesine ESP8266 modünü ekleyerek eve hırsız girdiğinde ev sahibinin telefonuna internet üzerinden bildirim gönderebilir ve projeyi IOT projesine dönüştürebilirsin.Dalgıç pompa ile ev tavanına yangın söndürme boruları çekebilir, bu sayede evde yangın çıktığında otomatik olarak söndürülmesi sağlayabilirsin.

2.18.6. Projenin MicroPython Kodları

```
from machine import Pin, PWM
from utime import sleep

PIR=Pin(14, Pin.IN)
MQ2=Pin(1,Pin.IN)
buzzer=PWM(Pin(20,Pin.OUT))
redLed=Pin(7,Pin.OUT)
button=Pin(10,Pin.IN,Pin.PULL_DOWN)


activated=0
gas=0

while True:
    if button.value()==1:
        activated=1
        gas=0
        sleep(3)
        redLed.value(1)
        buzzer.duty_u16(0)
    if MQ2.value()==1:
        gas=1
    if activated==1:
        if PIR.value()==1:
buzzer.duty_u16(6000)
            buzzer.freq(440)
            sleep(0.2)
            buzzer.freq(330)
            sleep(0.1)
            buzzer.freq(494)
            sleep(0.15)
```

```
            buzzer.freq(523)
            sleep(0.3)
    if gas==1:
            buzzer.duty_u16(6000)
            buzzer.freq(330)
            sleep(0.5)
            redLed.value(1)
            buzzer.freq(523)
            sleep(0.5)
            redLed.value(0)
```

2.18.7. Projenin Arduino C Kodları

```
void actived (){
  digitalWrite(7,1);
  while(!(digitalRead(14) == 1))
  {
    _loop();
  }
  motion_dedected();
}

void motion_dedected (){
  while(1) {
     tone(20,262,0.25*1000);
     delay(0.25*1000);
     tone(20,330,0.25*1000);
     delay(0.25*1000);
     tone(20,262,0.25*1000);
     delay(0.25*1000);
     tone(20,349,0.25*1000);
     delay(0.25*1000);

     _loop();
  }
}

void _delay(float seconds) {
  long endTime = millis() + seconds * 1000;
  while(millis() < endTime) _loop();
```

```
}

void _loop() {
}

void loop() {
  _loop();
}

void setup() {

  pinMode(10,INPUT);
  pinMode(1,INPUT);
  pinMode(20,OUTPUT);
  pinMode(7,OUTPUT);
  pinMode(14,INPUT);

  while(1) {
    if(digitalRead(10) == 1){
      _delay(3);
      actived();
    }
    if(digitalRead(1) == 1){
      while(!(digitalRead(10) == 1))
      {
        _loop();
        tone(20,349,0.5*1000);
        delay(0.5*1000);
        digitalWrite(7,1);
        _delay(0.5);
        tone(20,392,0.5*1000);
        delay(0.5*1000);
        digitalWrite(7,0);
        _delay(0.5);
      }
    }
    _loop();
  }
}
```

2.19. Obur Kumbara

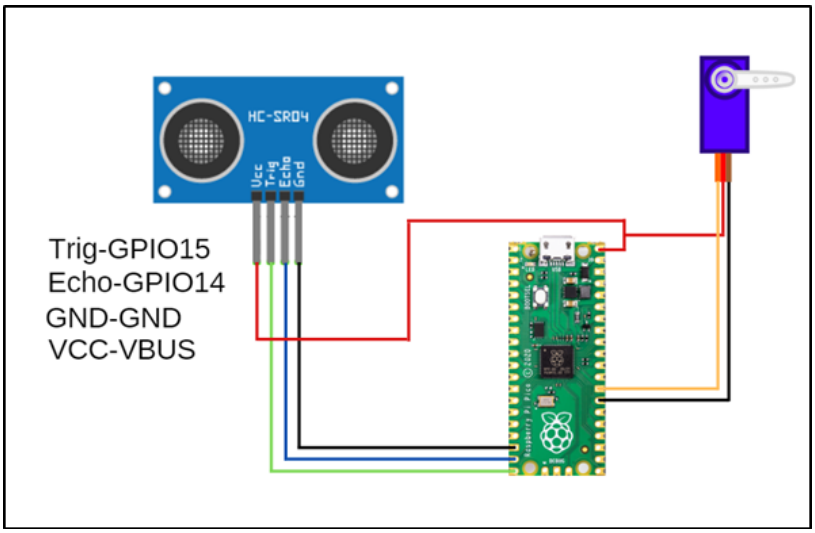
Ultrasonic sensörler ses dalgalarından etkilenerek elektriksel değişim gösteren sensörlerdir. Bu sensörler kulağımızın algılayamayacağı frekansta ses dalgaları gönderirler ve yansıyan ses dalgalarının geri geliş süresinin hesaplanması ile mesafe bilgisi üretirler. Biz programcılar ise ölçülen mesafeyi ve mesafedeki değişimleri anlamlandırarak projeler geliştiririz. Otomobillerin ön ve arkasında bulunan park sensörleri ultrasonic sensörlerin günlük yaşamda en çok karşımıza çıktığı yerlerdir. Doğada bu yöntemle yönünü bulan canlıyı biliyor musun? Yarasalar kör oldukları için çıkardıkları seslerin yansımalarından yollarını bulurlar.

Bir çoğumuz para biriktirmeyi severiz. Azar azar biriktirdiğimiz paraların ihtiyaç halinde işe yaraması çok güzel bir duygudur. Bu projede kendine çok eğlenceli ve sevecen bir kumbara yapacaksın. Kumbarayı yaparken servo motor ve ultrasonik mesafe sensörünü kullanacaksın.

2.19.1. Proje Detayları ve Algoritma

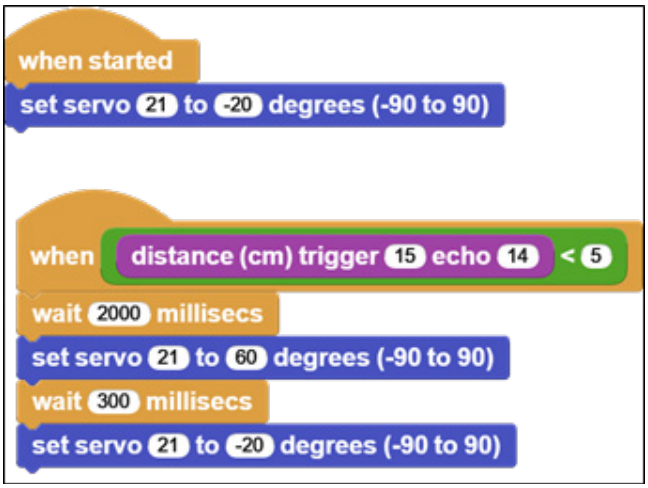
Bu projede HC-SR04 ultrasonic mesafe sensörü ve SG90 servo motor kullanılacaktır. Kumbaranın haznesine kullanıcı parak bıraktığında mesafe sensörü yakınlığı algılayacak ve Picobrickse gönderecektir. Picobricks de bu bilgiye göre servo motor çalıştırarak kolu yukarı kaldıracak, parayı kumbaranın içine atacak ve kol tekrar aşağı inecektir.

2.19.2. Bağlantı Şeması



2.19.3. Projenin MicroBlocks ile Kodlanması

Kodları yazarken ilk olarak servo motorun açısını düzenlemelisin. Kumbaranın kapağını kapalı konuma getirmek için servo motora açı değerleri girmeli ve en uygun değerli belirlemelisin. Daha sonra kumbaranın kapağının açık olduğu pozisyondaki servo motor açısını bulmalısın. Kumbara ilk başladığında kapak kapalı konumda olmalıdır. Ultrasonic mesafe sensöründen gelen değer 5 cm den küçük olduğunda 2 saniye bekleyip servo motor çalışarak kapağı yukarı kaldırmalı ve 300 milisaniye sonra tekrar çalışarak kapağı kapalı konuma getirmelidir.

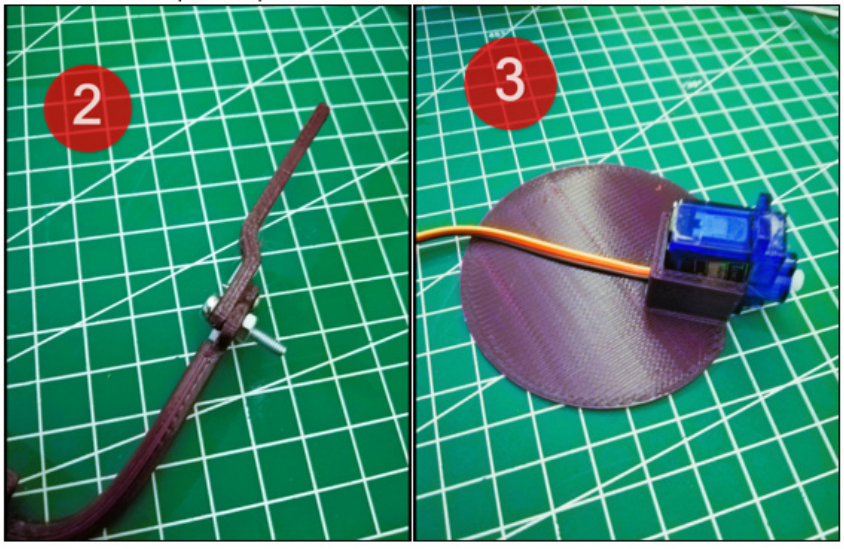


Projenin MicroBlocks kodlarına ulaşmak için tıkla.

2.19.4. Projenin Yapım Aşamaları

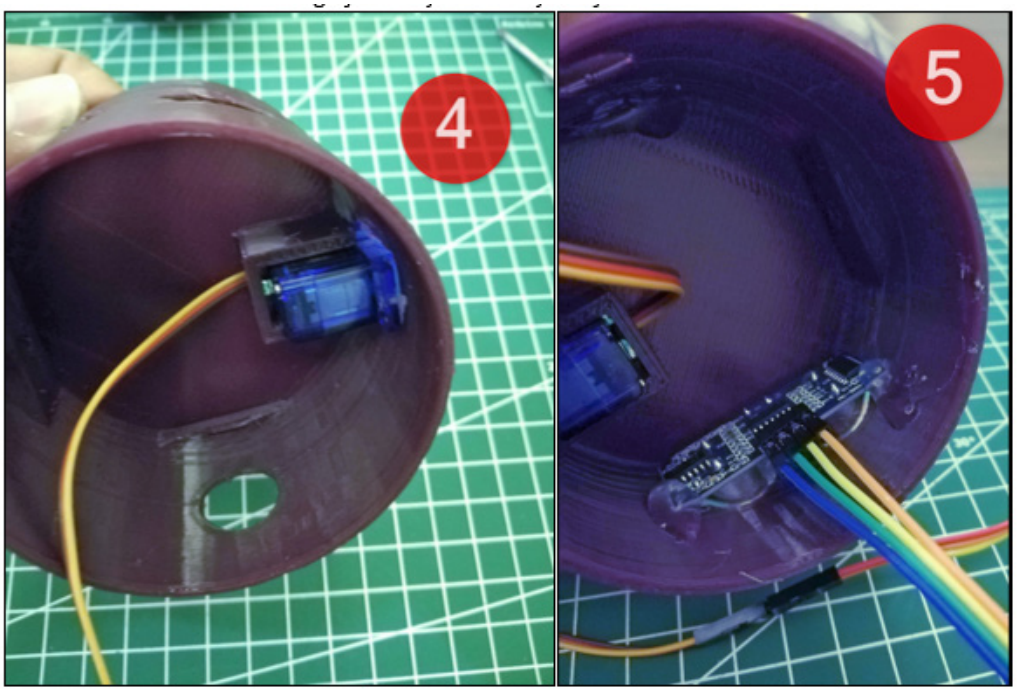
Projenin orjinal dosyaları ve yapım aşamaları sayfasına buraya tıklayarak erişebilirsin. Bu linkteki çalışmadan farklı olarak biz HC-SR04 ultrasonic mesafe sensörü kullanacağız. HC-SR04 ultrasonic mesafe sensörüne göre güncellenmiş 3D çizim dosyalarını bu linkten indirebilir ve 3D baskı alabilirsin.

1: Servo motorun plastik aparatını 2 adet vida ile kumbara koluna sabitle.



2: Kumbara kolunun ikinci parçasını M3 vida ve somun ile haznenin bulunduğu birinci parçaya sabitle.

3: servo motoru kablosunu geçirerek yuvasına yerleştir.

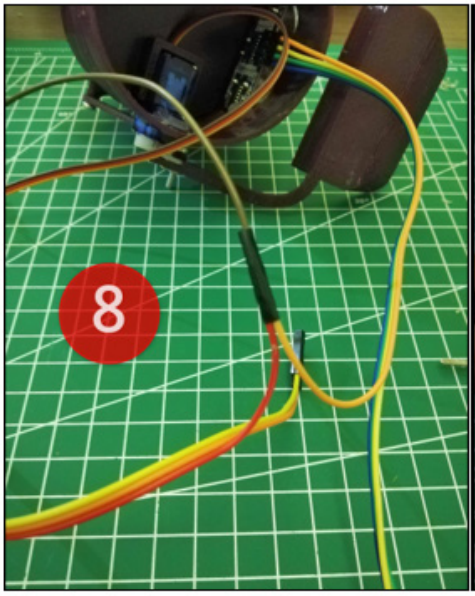


4: Servo motor ve yuvasını kumbaranın gövdesine yerleştir. Burada sıcak silikon kullanabilirsin.

5: Ultrasonic mesafe sensörünü kumbara gövdesine yerleştir ve sıcak silikon ile sabitle.

6: Kumbara kolunu servo motora tak ve üst kapağa M3 vida ile sabitle.

7: kumbara kolunu gövdeye M2 vida ile sabitle.



8: Servo motor ve ultrasonic mesafe sensörün kablolarını tak ve güç kablolarını birleştir.

9: Devre şemasına göre servo motor ve ultrasonic mesafe sensörün kablolarını picoya tak.

10: Pico nun USB kablosu tak ve kabloları toparlayarak alt kapağı tak. Hepsi bu kadar.

2.19.5. Proje Önerisi

Obur kumbara projesine RGB led modülü ekleyerek her para atıldığında dilediğin renkte ışık yanmasını sağlayabilir, buzzer ekleyerek her para atıldığında ses çıkmasını sağlayabilirsin. Ayrıca OLED ekran ekleyerek kaç kere para atıldığını ekrana yazdırabilirsin.

2.19.6. Projenin MicroPython Kodları

```
from machine import Pin, PWM
from utime import sleep
import utime

servo=PWM(Pin(21,Pin.OUT))
trigger = Pin(15, Pin.OUT)
echo = Pin(14, Pin.IN)

servo.freq(50)
servo.duty_u16(4010) #70 degree

def getDistance():
  trigger.low()
  utime.sleep_us(2)
  trigger.high()
  utime.sleep_us(5)
  trigger.low()
  while echo.value() == 0:
    signaloff = utime.ticks_us()
  while echo.value() == 1:
    signalon = utime.ticks_us()
  timepassed = signalon - signaloff
  distance = (timepassed * 0.0343) / 2
  print("The distance from object is ",distance,"cm")
  return distance

while True:
  sleep(0.01)
  if int(getDistance())<=5:
    servo.duty_u16(7050) #150 degree
    sleep(0.3)
    servo.duty_u16(4010)
```

2.19.7. Projenin Arduino C Kodları

```
#include <NewPing.h>
#include <Servo.h>
Servo myservo;

#define TRIGGER_PIN  15
#define ECHO_PIN     14
#define MAX_DISTANCE 400

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

void setup() {
  myservo.attach(21);
  myservo.write(20);
}

void loop() {

  delay(50);
  int distance=sonar.ping_cm();

  if(distance<10){

    int x=100;
    int y=20;
    delay(2000);
    myservo.write(x);
    delay(300);
    myservo.write(y);
  }
}
```

2.20. Confirming Door

Güvenlik sistemlerinin kapsamına bina ve oda girişlerinde yetkileri kontrol edebilen teknolojiler de girmektedir. Hastanelerin ameliyathanelerine sadece görevli personellerin girebildiği kartlı giriş sistemleri ilk akla gelen örneklerden biridir. Ayrıca askeri güvenlik merkezlerinde her düzeyden kişi veya personelin girmemesi gereken alanların giriş kapıları kartlı ve şifreli giriş teknolojileriyle donatılmaktadır. Bina ve oda girişlerinde kullanılan bu elektronik sistemler yetkisiz kişilerin girişini engellediği gibi giriş çıkış bilgilerinin kayıt altında tutulmasını da sağlamaktadır. Şifreli giriş, kartlı giriş, parmak izi tarama, yüz tarama, retina taraması ve ses tanıma teknolojileri elektronik giriş sistemlerinde kullanılan doğrulama yöntemleridir.

RFID ve NFC gibi sistemler bugün temassız ödeme teknolojilerinin temel halleridir. Kredi kartlarındaki temassız ödeme teknolojisi teknik olarak farklı olsa da çalışma mantığı aynıdır. Okuyucu ve kart arasında yer alacak maksimum mesafe, kullanılan teknolojileri birbirinden ayıran özelliklerin başında gelir. Alışveriş mağazalarından çıkarken özellikle giyim mağazalarında ürünlerin üzerindeki NFC etiketler girişteki okuyuculara takılırsa öterler. O sistemlerde de bir tür RFID teknolojisi kullanılmaktadır.

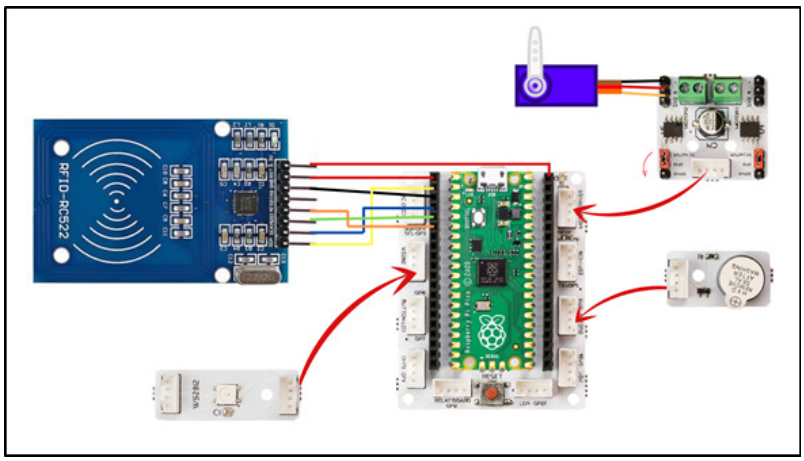
Bu projede maket ev üzerinde kartlı giriş sistemi hazırlayacağız. Kullanacağımız elektronik bileşenler MFRC522 RFID okuyucu ve 13.56 Mhz'lik kartlardır.

2.20.1. Proje Detayları ve Algoritma

Maket'in kapısının yanına MFRC522 okuyucu dışarıdan görünecek şekilde yerleştir. RGB LED ve buzzer da dışarıdan görünecek şekilde kapının bulunduğu duvarı üzerine yerleştir. Picoboard maket içinde kalabilir. Maketin giriş kapısı servonun başlığına bağlı olmalı servo da 0 dereceye ayarlı iken kapının kapalı olması sağlanmalıdır. Kapıyı açacak RFID /NFC etiketinin seri nosunu tespit edip homeowner değişkeni oluşturup seri noyu bu değişkene atamalısın.

Picobricks başladığında kapıyı kapalı konuma getir. RFID okuyucuya bir kart gösterildiğinde buzzerdan bip sesi çıkmasını sağla. Okutulan kartın seri numarası homeowner değişkenindeki seri no ile eşleşiyorsa RGB LED i yeşil renkte yak. Ardından kapının açılmasını sağla. Kapı açıldıktan 3 saniye sonra kapının kapanmasını sağla. Okutulan kartın seri numarası homeowner değişkeni ile eşleşmiyorsa RGB LED i kırmızı renkte yak. Buzzerdan farklı tonda bir ses çıkar.

2.20.2. Bağlantı Şeması

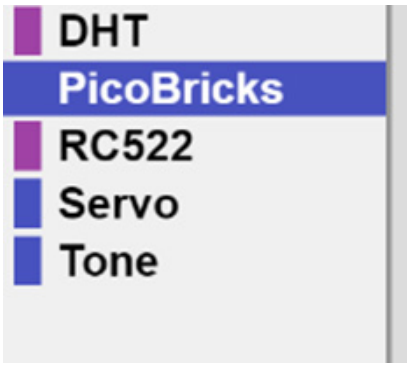


RC522 RFID Reader Module	PicoBricks
VCC	3.3V
RST	GP0
GND	GND
IRQ	Not connected
MISO	GP4
MOSI	GP3
SCK	GP2
SDA	GP1

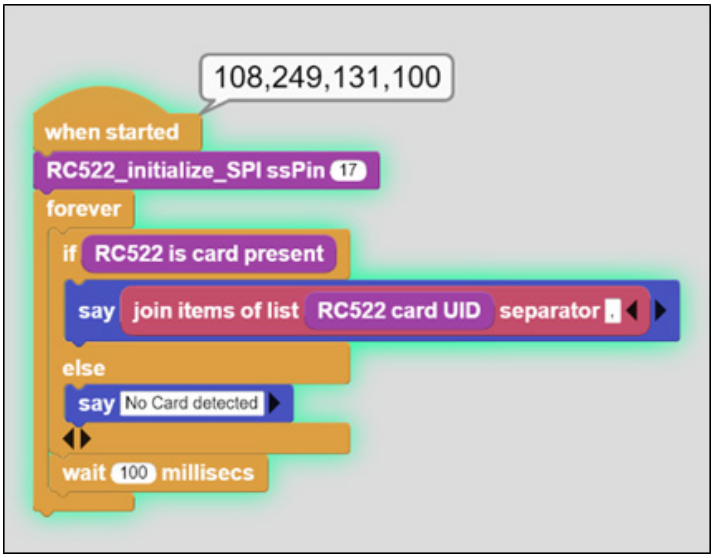
*** Microblocks kullanılırken bağlanmaz.

2.20.3. Projenin MicroBlocks ile Kodlanması

Önce geçerli kullanıcının kart bilgisini okumalıyız. Ardından projenin kodlarını hazırlayacağız. homeowner adında bir değişken oluştur. RC522 ve Servo kütüphanelerini ekle.



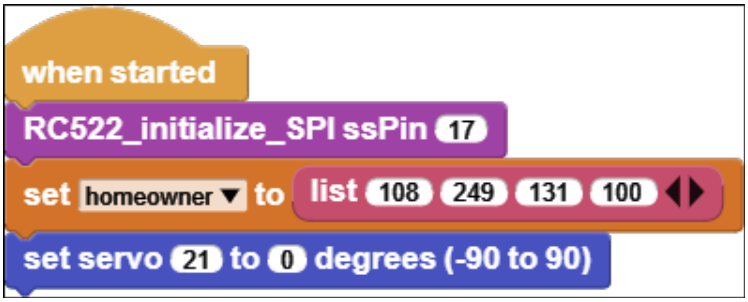
Picobricks başladığında “RC522_initialize” bloğu ile SDA pinini belirtip modülü tanımla. Forever döngüsünün içine if else yapısını yerleştir. Kart okunduğunda kart seri no’sunu ekrana yazdıracağız. “RC522 card UID” bloğu 4 elemanlı bir listedir. Bu değeri göstermek veya karşılaştırmak için “Data” kategorisindeki “join items of list” bloğunu kullanmalısın.



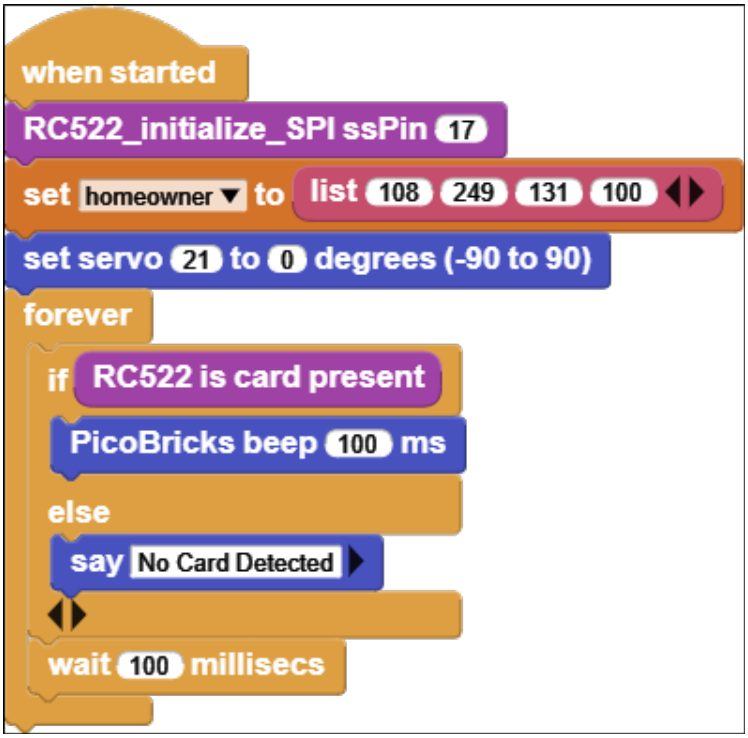
Kart Seri No’sunun okumak için gerekli MicroBlocks kodları için tıkla.

Kodu çalıştırdığınızda RC522’ye kartınızı gösterin. Kod bloklarının üstünde çıkan kart seri numarasını homeowner değişkenine Data kategorisindeki “list” bloğu ile atayın. Senin kartının seri numarası burada yazandan farklıdır.

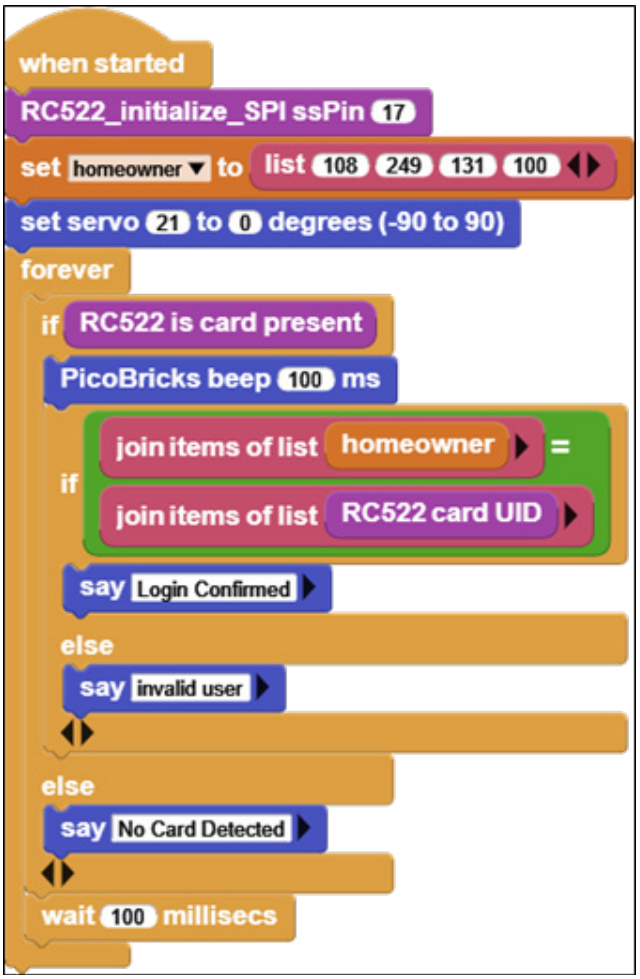
Artık projenin algoritmasının kodlarını yazabiliriz. Picbricks başladığında RC522’yi tanımla , homeowner değişkenini tanımla ve servo açısını ayarla



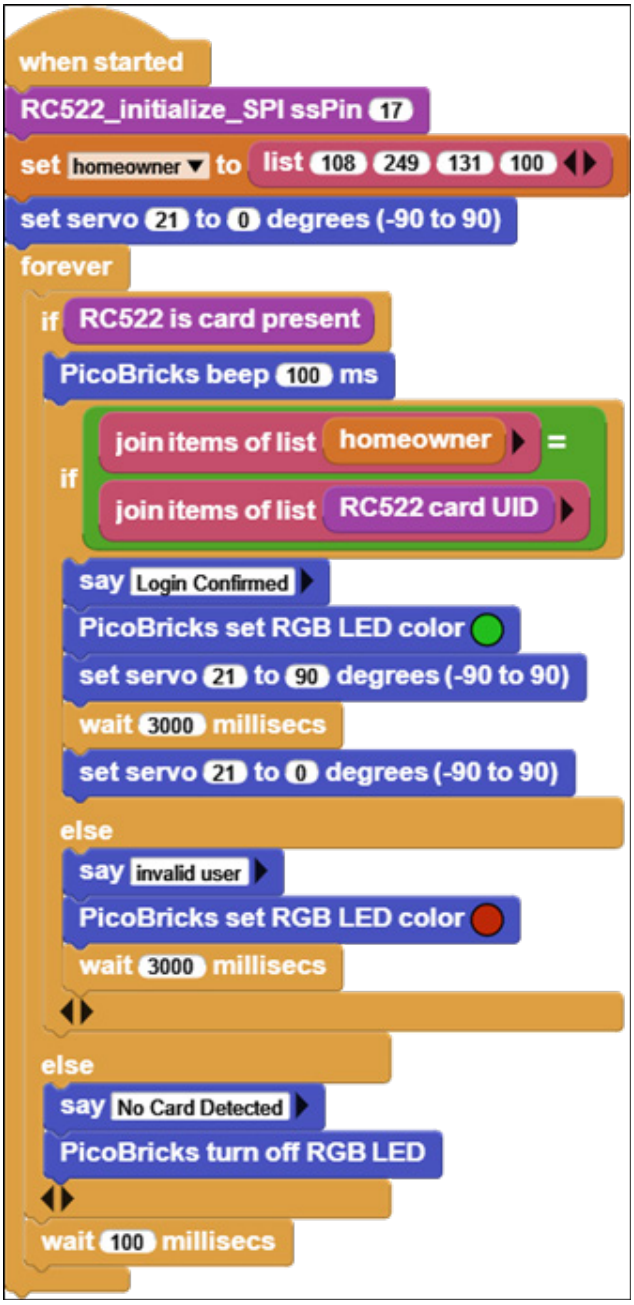
Kart gösterildiğinde buzzer’dan bip sesi çıkarılacaktır. Kart gösterilmediği durumlarda “No Card detected” ifadesi yazdırılacaktır. Forever döngüsünün içinde if else yapısı yerleştirerek gerekli kodları aşağıdaki gibi hazırla.



Bip sesi çıktıktan hemen sonra homeowner değişkeni ile okunan kart'ın seri numarasını karşılaştıracak "if else" yapısını hazırla. Eğer Seri numaraları eşleşiyorsa "Login Confirmed" ifadesini yazdır. Seri numaralar eşleşmiyorsa " invalid user" ifadesini yazdır. Kodların aşağıdaki gibi olmalı.



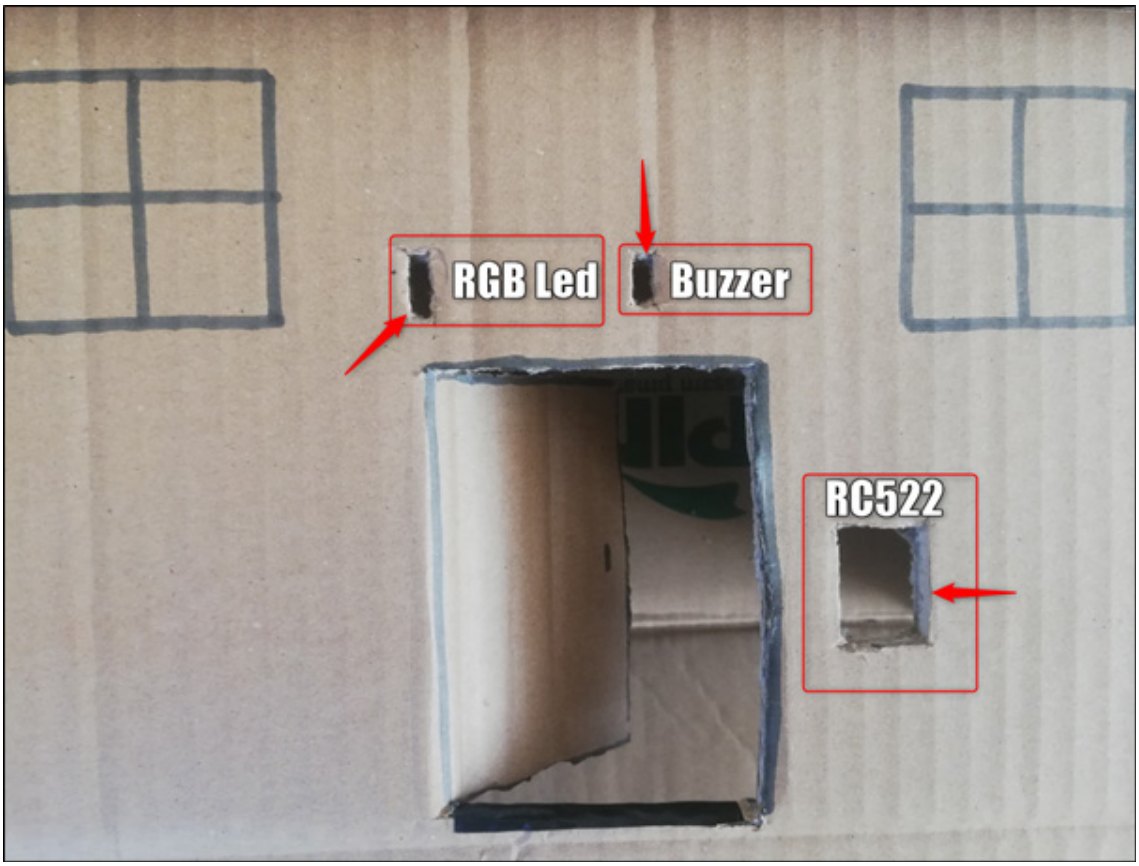
Doğru kart gösterildiğinde Servonun hareket etmeli ve RGB LED yeşil renkte yanmalı. 3 saniye beklenmeli ve servo eski açısına geri dönmeli. Yanlış kart gösterildiğinde RGB LED kırmızı renkte yanmalı. 3 saniye beklenmeli. Kart okutulmadığı durumlarda ise RGB LED söndürülmelidir. Projenin bitmiş kodları aşağıdaki gibi olmalıdır.



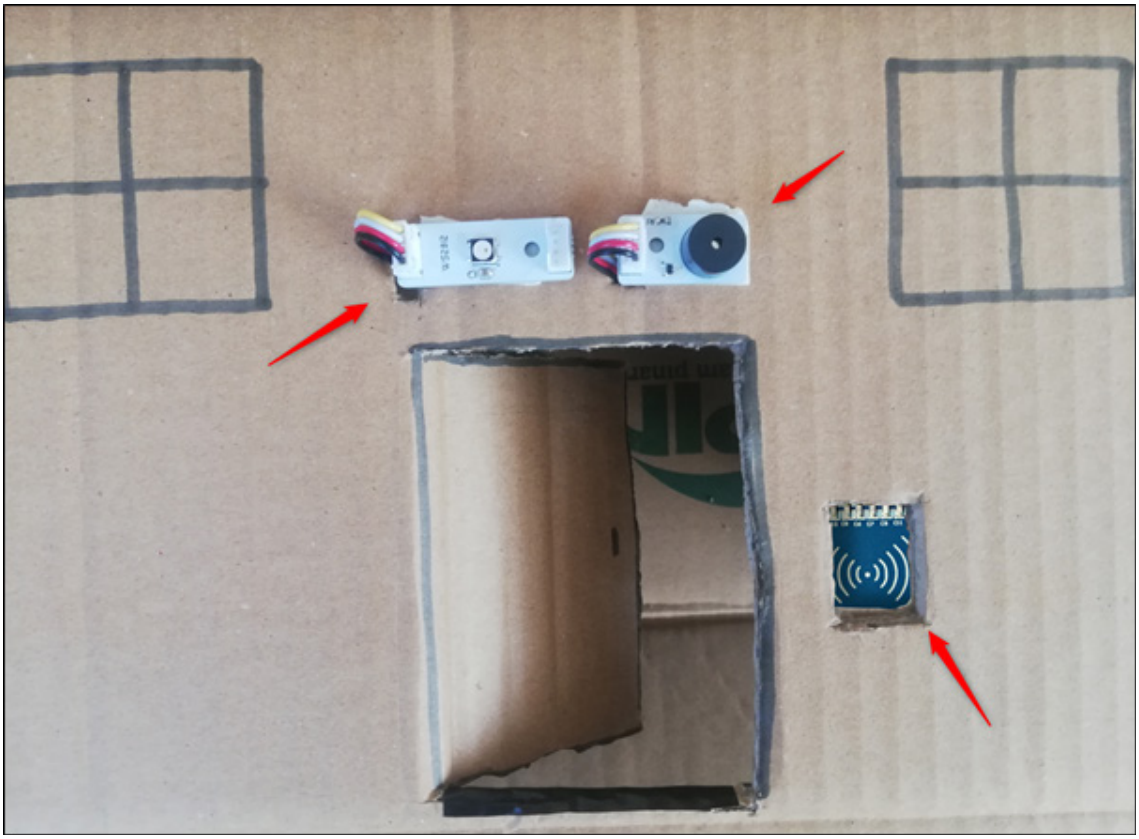
Projenin MicroBlocks kodlarına erişmek için tıkla.

2.20.4. Projenin Yapım Aşamaları

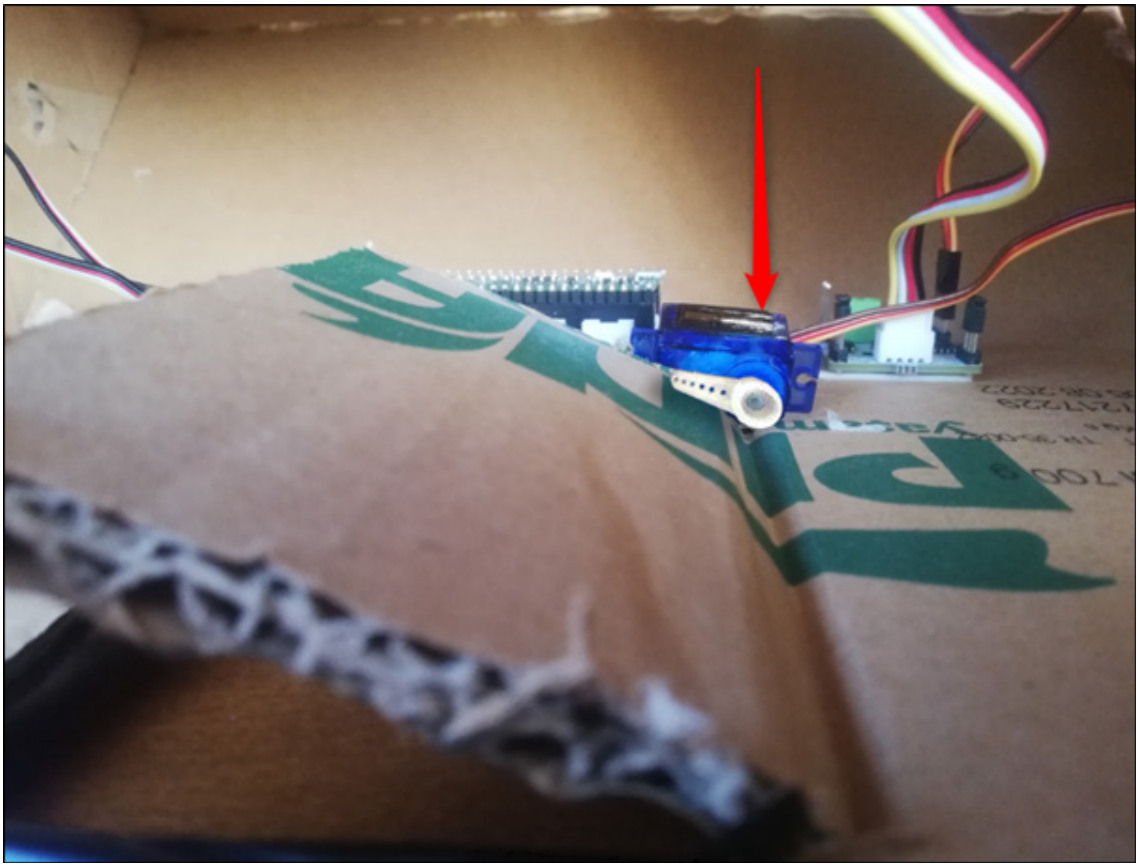
18 numaralı Akıllı Ev projesinde kullandığın ev maketi üzerinde projeyi yapacağız. Ev maketinin üzerinde RGB LED, Buzzer ve RC522 RFID okuyucu için delikler aç.



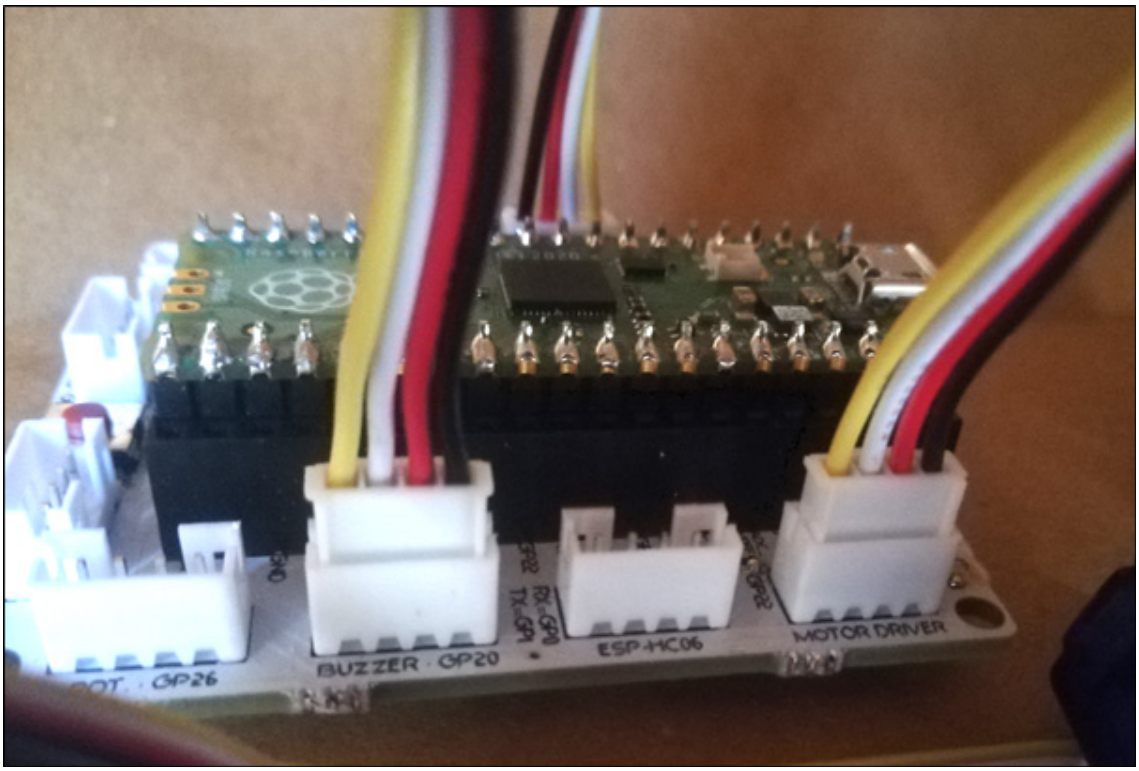
RGB Led ve Buzzer'ın arkasına çift taraflı köpük bant yapıştırıp kutunun üzerine yapıştır. RC522 yi maketin içinden görseldeki gibi yerleştir.



Servo motoru, maketin içine kapının sol üst köşesinde menteşe olarak duracak şekilde çift taraflı bant ile yapıştır. Servo başlığını kapıya sıcak silikon ya da sıvı yapıştırıcı ile yapıştır.



Son olarak Pico board'ı ve 2'li anahtarlı pil kutusunu maket evin içine yerleştirin ve kablo bağlantılarını tamamlayın. Projenizin son kontrollerini yaptıktan sonra çalışmaya hazırdır.



2.20.5. Proje Önerisi

Otomatik kapı projesinde RFID kartlara kişi isimleri verebilir, projeye OLED ekran ekleyerek kart okutulduğunda kartı okutan kişinin adını OLED ekrana yazdırabilirsin.

2.20.6. Projenin MicroPython Kodları

The code to be run to learn the Card ID:

```
from machine import Pin, SPI
from mfrc522 import MFRC522

sck = Pin(2, Pin.OUT)
mosi = Pin(3, Pin.OUT)
miso = Pin(4, Pin.OUT)
sda = Pin(1, Pin.OUT)
rst = Pin(0, Pin.OUT)
spi = SPI(0, baudrate=100000, polarity=0, phase=0, sck=sck, mosi=mosi, miso=miso)
rdr = MFRC522(spi, sda, rst)

while True:
    (stat, tag_type) = rdr.request(rdr.REQIDL)
    if stat == rdr.OK:
        (stat, raw_uid) = rdr.anticoll()
        if stat == rdr.OK:
            uid = ("0x%02x%02x%02x%02x" % (raw_uid[0], raw_uid[1], raw_uid[2], raw_
uid[3]))
            print(uid)
            utime.sleep(1)
```

Project Codes:

```
from machine import I2C, Pin, SPI, PWM
from mfrc522 import MFRC522
from ws2812 import NeoPixel
from utime import sleep

servo = PWM(Pin(21))
servo.freq(50)
servo.duty_u16(1350) #servo set 0 angle 8200 for 180.

buzzer = PWM(Pin(20, Pin.OUT))
buzzer.freq(440)

neo = NeoPixel(6, n=1, brightness=0.3, autowrite=False)
RED = (255, 0, 0)
GREEN = (0, 255, 0)
```

```
BLACK = (0, 0, 0)

sck = Pin(2, Pin.OUT)
mosi = Pin(3, Pin.OUT)
miso = Pin(4, Pin.OUT)
sda = Pin(1, Pin.OUT)
rst = Pin(0, Pin.OUT)
spi = SPI(0, baudrate=100000, polarity=0, phase=0, sck=sck, mosi=mosi, miso=miso)
homeowner = “0x734762a3”
rdr = MFRC522(spi, sda, rst)

while True:

    (stat, tag_type) = rdr.request(rdr.REQIDL)
    if stat == rdr.OK:
        (stat, raw_uid) = rdr.anticoll()
        if stat == rdr.OK:
            buzzer.duty_u16(3000)
            sleep(0.05)
            buzzer.duty_u16(0)
            uid = (“0x%02x%02x%02x%02x” % (raw_uid[0], raw_uid[1], raw_uid[2], raw_
uid[3]))
            print(uid)
            sleep(1)
            if (uid==homeowner):
                neo.fill(GREEN)
                neo.show()
                servo.duty_u16(6000)
                sleep(3)
                servo.duty_u16(1350)
                neo.fill(BLACK)
                neo.show()

            else:
                neo.fill(RED)
                neo.show()
                sleep(3)
                neo.fill(BLACK)
                neo.show()
                servo.duty_u16(1350)
```

2.20.7. Projenin Arduino C Kodları

2.21. Otomatik Çöp Kovası

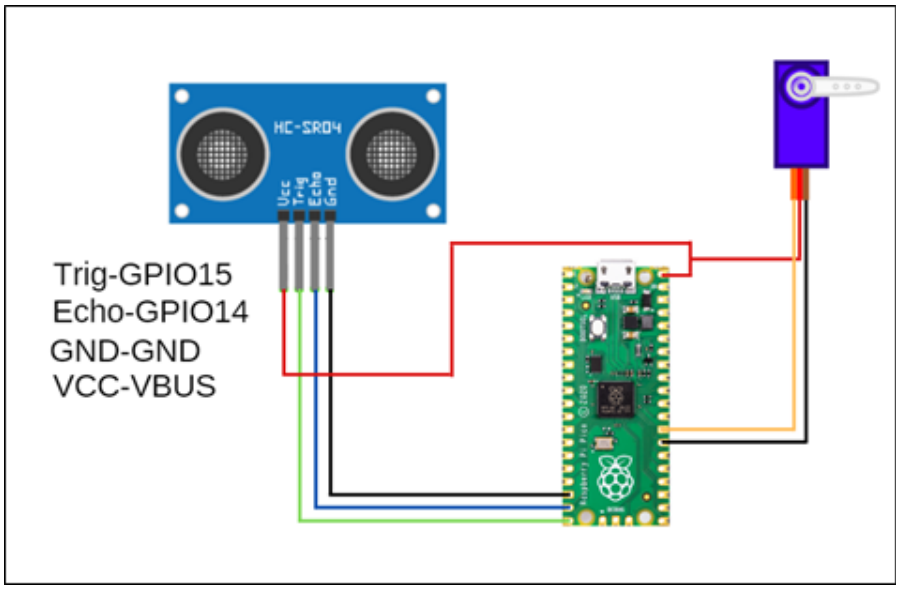
Covid-19 pandemisi insanların günlük yaşamdaki rutinlerini bir çok alanda değiştirdi. Temizlik, çalışma, alışveriş, sosyal hayat gibi bir çok alanda insanlar uymak zorunda olduğu bir dizi yeni kurallarla tanıştı. Covid-19 yeni iş alanlarının doğmasına ve gelişmesine zemin oluşturduğu gibi bazı ürünlerinde ön plana çıkmasını sağlamıştır. El hijyenin çok önemsendiği bir dönemde kimse çöpünü atmak için çöp kovasının kapağına dokunmak istemezdi. Yanına yaklaşıldığında kapağı otomatik açılan dolduğunda ise tek bir hareketle içindeki poşeti büzüp çıkarıp atmaya hazır hale getiren çöp kovaları maliyetinin çok üstünde fiyatlara alıcı buldu. Ayrıca otomatik dezenfektan makineleri elimizi altına tuttuğumuzda belirli bir miktar sıvıyı avucumuzun içine boşaltmakla temassız bir hijyen sağlıyordu. Otomatik dezenfektan sıkıcılarda maliyetinin çok üzerindeki fiyatlara raflarda yer aldı. Bu iki üründe çalışma sistemi açısından benzerlikler yer almaktadır. Otomatik dezenfektan sıkıcılarda doğrudan elektrik motorlu bir pompa sıvıyı dışarı aktardığı gibi bazı modellerde de servo motorun gücü ile pompalanma sistemine dayanan cihazlar bulunmaktaydı. Otomatik çöp kovalarında ise kapağı açan servo motor kullanılmakta el hareketini algılamak için ise kızılötesi ya da ultrasonic sensörler kullanılmaktaydı.

Bu projede PicoBricks ile ultrasonik sensör ve servo motor kullanarak odan için hareketli ve otomatik şık bir çöp kovası yapacaksın.

2.21.1. Proje Detayları ve Algoritmaü

Bu projede HC-SR04 ultrasonic mesafe sensörü ve SG90 servo motor kullanılacaktır. Çöp kutusunun kapağının önüne kullanıcı elini yaklaştırdığında mesafe sensörü yakınlığı algılayacak ve Picobrickse gönderecektir. Picobricks de bu bilgiye göre servo motor çalıştırarak çöp kovasının kapağını açacak kısa bir süre sonra tekrar aşağı indirecektir.

2.21.2. Bağlantı Şeması



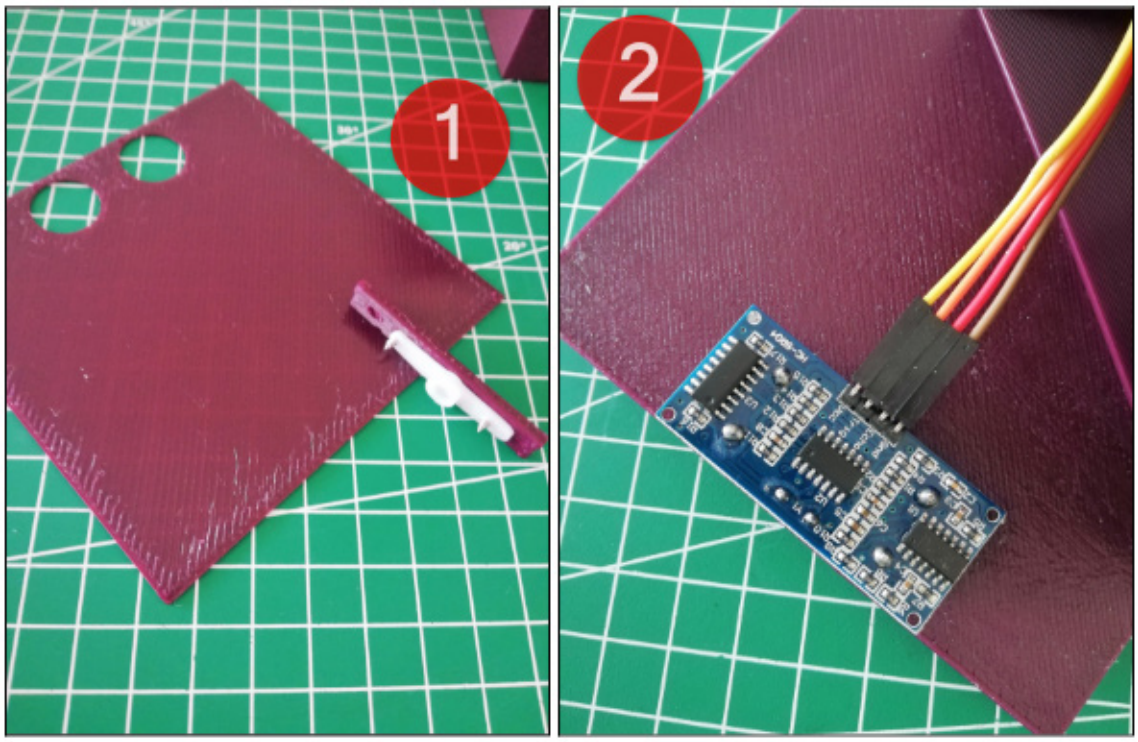
2.21.3. Projenin MicroBlocks ile Kodlanması

Kodları yazarken ilk olarak servo motorun açısını düzenlemelisin. Çöp kovasının kapağını kapalı konuma getirmek için servo motora açı değerleri girmeli ve en uygun değerli belirlemelisin. Daha sonra çöp kovasının kapağının açık olduğu pozisyondaki servo motor açısını bulmalısın. Çöp kovası ilk başladığında kapak kapalı konumda olmalıdır. Ultrasonic mesafe sensöründen gelen değer 10 cm den küçük olduğunda 2 saniye bekleyip servo motor çalışarak kapağı yukarı kaldırmalı ve 2 saniye sonra tekrar çalışarak kapağı kapalı konuma getirmelidir.

Projenin MicrBlocks kodlarına ulaşmak için tıkla.

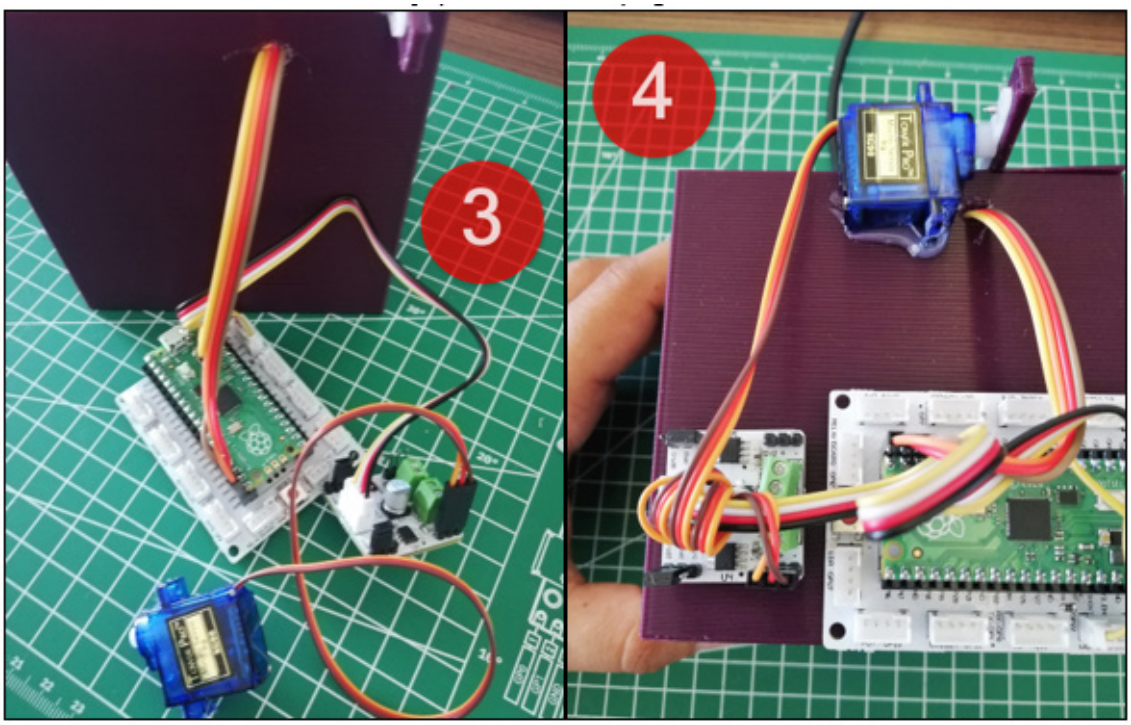
2.21.4. Projenin Yapım Aşamaları

Projenin 3D çizim dosyalarını bu linkten indirebilir ve 3D baskı alabilirsin.



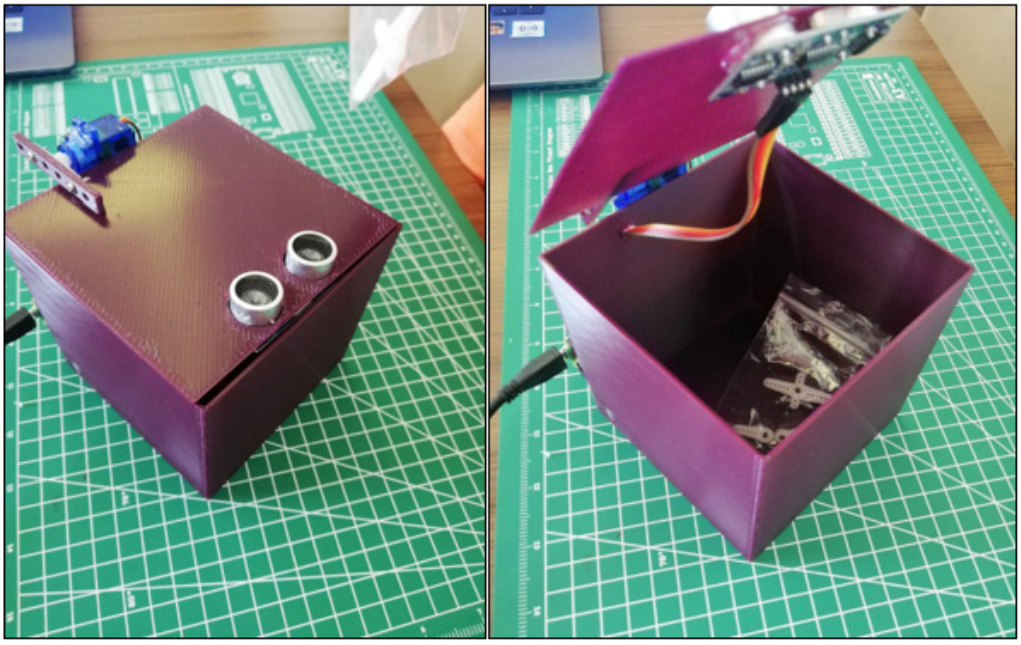
1: Servo motorun aparatının çöp kovası kapağına vidalarayarak sabitle.

2: Ultrasonic mesafe sensörünü çöp kovasının kapağına sıcak silikon ile sabitle.



3: Ultrasonic mesafe sensörünün kablolarını kutudaki delikten geçirerek picobrickste devre şemasında gösterilen pinlere tak, servo motor ve motor sürücü bağlantılarını yap.

4: Servo motor, picobricks ve motor sürücü parçalarını sıcak silikon ile kutuya sabitle.



herşey yolunda gittiyse çöp kovasına elini yaklaştırdığında kovanın kapağı açılacak ve çöpü attıktan sonra tekrar geri kapanacaktır.

2.21.5. Proje Önerisi

Evimizdeki bir çöp kovasını sensör ve motor kullanarak otomatik hale getirdiğimiz bu projedeki gibi evdeki bir çekmecenin ya da bir dolap kapağının yine sensör ve motorlar yardımı ile otomatik olarak açılmasını sağlayabilirsin.

2.21.6. Projenin MicroPython Kodları

```
from machine import Pin, PWM
from utime import sleep
import utime

servo=PWM(Pin(21,Pin.OUT))
trigger = Pin(15, Pin.OUT)
echo = Pin(14, Pin.IN)

servo.freq(50)
servo.duty_u16(1920) #15 degree

def getDistance():
  trigger.low()
  utime.sleep_us(2)
  trigger.high()
  utime.sleep_us(5)
  trigger.low()
  while echo.value() == 0:
    signaloff = utime.ticks_us()
  while echo.value() == 1:
    signalon = utime.ticks_us()
  timepassed = signalon - signaloff
  distance = (timepassed * 0.0343) / 2
  print("The distance from object is ",distance,"cm")
  return distance

while True:
  sleep(0.01)
  if int(getDistance())<=10:
    servo.duty_u16(4010) #70 degree
    sleep(0.3)
    servo.duty_u16(1920)
```

2.21.7. Projenin Arduino C Kodları

```
#include <NewPing.h>
#include <Servo.h>
Servo myservo;

#define TRIGGER_PIN  15
#define ECHO_PIN     14
#define MAX_DISTANCE 400

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

void setup() {
  myservo.attach(21);
  myservo.write(20);
}

void loop() {

  delay(50);
  int distance=sonar.ping_cm();

  if(distance<10){

    int x=70;
    int y=20;
    delay(2000);
    myservo.write(x);
    delay(2000);
    myservo.write(y);
  }
}
```

2.22. Dijital Cetvel

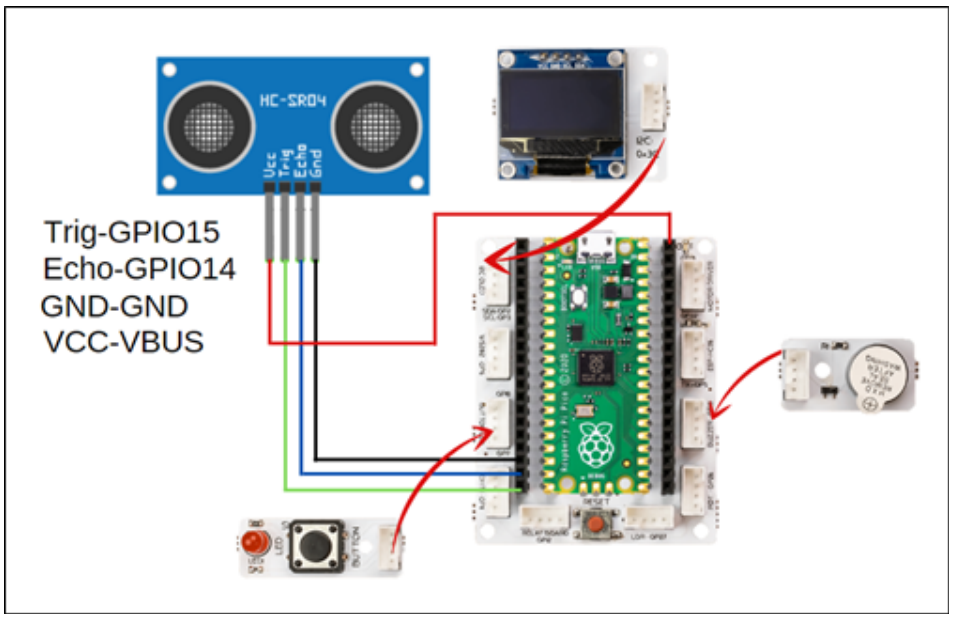
Uzunluk ölçmek için birçok araç kullanılmaktadır. Bu araçların başında cetveller gelmektedir. Ölçeceğimiz yere ve büyüklüğüne göre ölçü aletimiz farklılaşmaktadır. İnşaat ve mimaride şerit metreler , küçük ve milimetrik hassasiyet gerektiren nesneler için kumpaslar kullanılmaktadır. Ayrıca hem büyük hem de hassas ölçüm yapılması gereken bir alan ölçülmek isteniyorsa lazer ve kızılötesi sistemler ile çalışan mesafe ölçerler kullanılmaktadır. Sağlık sektöründe kullanılan ultrasonografi cihazları da benzer mantıkla çalışmakta ancak ölçümlerini görsellere dönüştürmektedir.

Projemizde PicoBricks ile ultrasonik sensör kullanarak mesafe değerini butona basıldığında OLED ekranda göstereceğimiz dijital bir cetvel hazırlayacağız. Ultrasonik sensörler yaydıkları ses dalgalarının geri dönüş sürelerine göre mesafe tespiti yaparlar.

2.22.1. Proje Detayları ve Algoritma

Picobricks başladığında OLED ekranda yönerge görüntülenir. Kullanıcı butona bastıktan sonra 1 saniye boyunca 50'şer milisaniye aralıklarla 20 ölçüm yapılır ve ortalaması alınır. Ölçüm sırasında kırmızı LED açık kalır ölçüm tamamlandığında kırmızı LED söndürülür. Ortalama değere sensörün ucu ile kutunun arkasındaki mesafe eklenir. Son mesafe değeri OLED ekranda gösterilir.

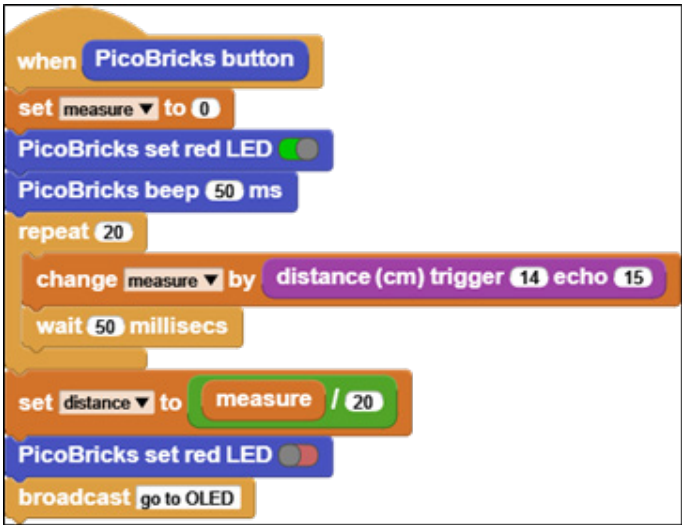
2.22.2. Bağlantı Şeması



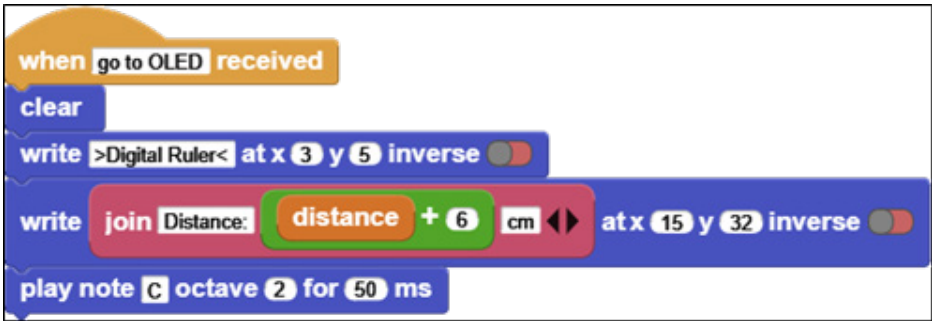
2.22.3. Projenin MicroBlocks ile Kodlanması

MicroBlocks'u başlat ve Picobricks i bağla. Tone,Distance,OLED Graphics kütüphanelerini ekle.Picobricks başladığında OLED ekranı tanımla.

Ölçüm değerinin ortalamasını alabilmek için measure mesafeyi hesaplamak için distance isimli değişkenleri oluştur. Butona basıldığında ölçümün başladığını bildirmek için kırmızı LED'i aç. Kısa bir bip sesinden sonra 50 milisaniye aralıkla 20 ölçüm yapıp ortalamasını distance değerine ata. Kırmızı LED'i kapatıp sonucu OLED ekrana yazdırmak için go to OLED yayını yap.



distance değişkenini ekrana yazdırırken sensörün ölçtüğü değere kutunun büyüklüğünü de ekle. Projemizde kullandığımız kutunun kalınlığı 6 cm olduğu için distance değişkenine 6 cm ekleyip ekrana öyle yazdırdık. İnce bir ses çıkarıp ölçümün tamamlandığını kullanıcıya bildir.

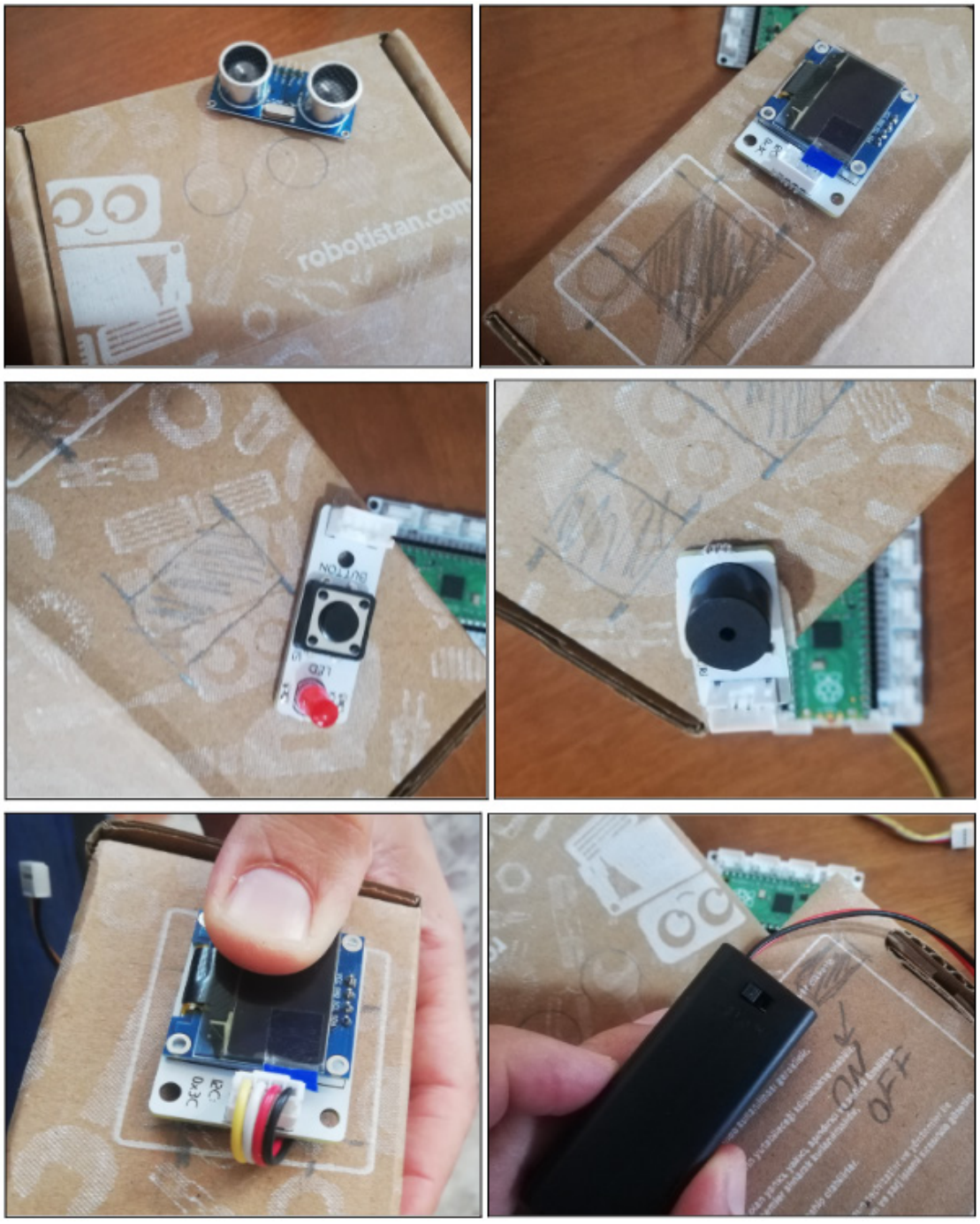


Projenin kodlarına ulaşmak için tıkla.

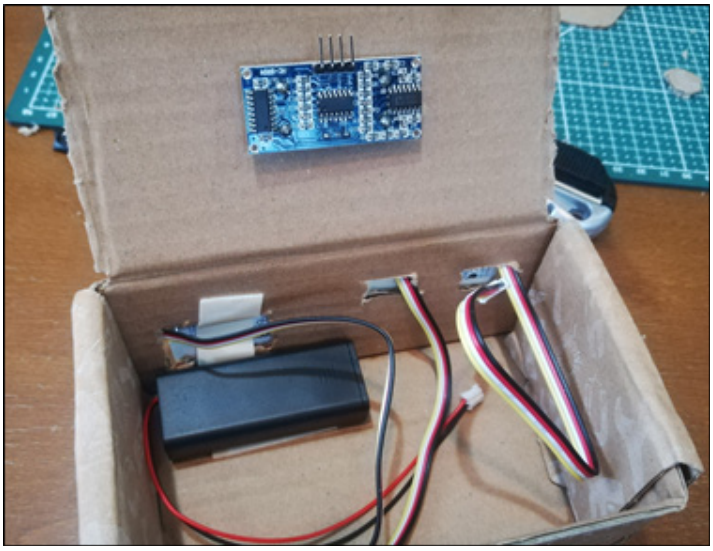
2.22.4. Projenin Yapım Aşamaları

Projeyi hazırlamak için çift taraflı köpük bant, maket bıçağı, yaklaşık 15x10x5 cm büyüklüğünde atık karton kutuya ihtiyacın var.

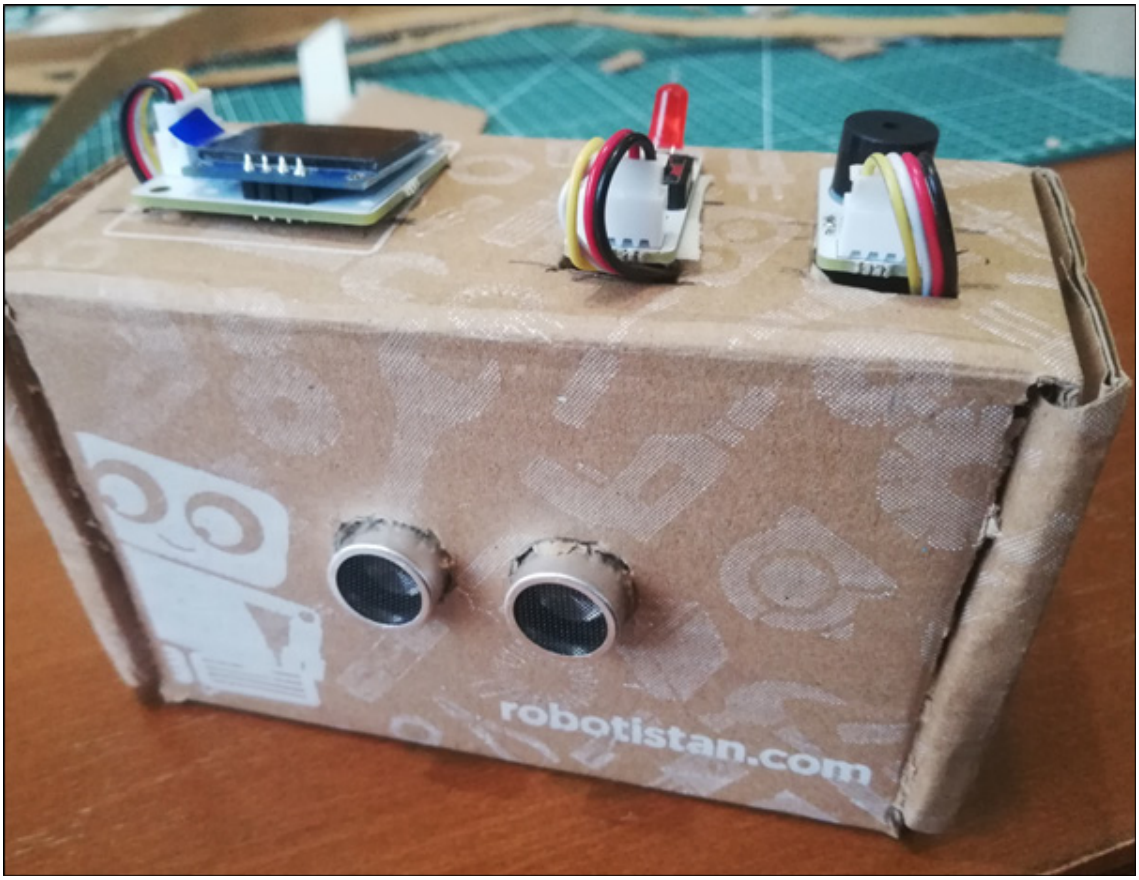
1- Ultrasonik sensör, OLED ekran, button LED modülü, buzzer, pil kutusu için kabloları geçireceğimiz delikleri maket bıçağı ile kesip çıkar.



2- Tüm kabloları kutunun içine sarkıt ve modüllerin arkasını çift taraflı köpük bant ile kutuya yapıştır.Ultrasonik sensörün trig pinini GPIO14, echo pinini GPIO15 nolu pine tak. VCC pinini ise Picoboard üzerindeki VBUS pinine takmalısın.



3- Tüm modüllerin kablo bağlantılarını tamamladıktan sonra 2li pil kutusunu Picoboard'ın power jag'ına takıp anahtarını açabilirsin. Dijital cetvel projesi işte bu kadar!



2.22.5. Proje Önerisi

Dijital cetveli duvara ya da tavan sabitleyerek boy ölçer dönüştürebilirsin. Boyölçer duvarda ya da tavanda sabit olacağı için butona basılarak ölçüm yapılamayacaktır. Bunu için HC05 bluetooth modülünü projeye ekleyebilir ve mobil uygulamadan komut geldiğinde ölçüm yapmasını sağlayabilirsin.

2.22.6. Projenin MicroPython Kodları

```
from machine import Pin, PWM, I2C
from utime import sleep
from ssd1306 import SSD1306_I2C
import utime

redLed=Pin(7,Pin.OUT)
button=Pin(10,Pin.IN,Pin.PULL_DOWN)
buzzer=PWM(Pin(20,Pin.OUT))
buzzer.freq(392)
trigger = Pin(15, Pin.OUT)
echo = Pin(14, Pin.IN)

WIDTH  = 128
HEIGHT = 64
sda=machine.Pin(4)
scl=machine.Pin(5)
i2c=machine.I2C(0,sda=sda, scl=scl, freq=1000000)
oled = SSD1306_I2C(128, 64, i2c)
measure=0
finalDistance=0

def getDistance():
  trigger.low()
  utime.sleep_us(2)
  trigger.high()
  utime.sleep_us(5)
  trigger.low()
  while echo.value() == 0:
    signaloff = utime.ticks_us()
  while echo.value() == 1:
    signalon = utime.ticks_us()
  timepassed = signalon - signaloff
```

```
  distance = (timepassed * 0.0343) / 2
  return distance

def getMeasure(pin):
   global measure
   global finalDistance
   redLed.value(1)
   for i in range(20):
      measure += getDistance()
      sleep(0.05)
   redLed.value(0)
   finalDistance = (measure/20) + 6
   oled.fill(0)
   oled.show()
   oled.text(“>Digital Ruller<”, 2,5)
   oled.text(“Distance “ + str(round(finalDistance)) +” cm”, 0, 32)
   oled.show()
   print(finalDistance)
   buzzer.duty_u16(4000)
   sleep(0.05)
   buzzer.duty_u16(0)
   measure=0
   finalDistance=0

button.irq(trigger=machine.Pin.IRQ_RISING, handler=getMeasure)
```

2.22.7. Projenin Arduino C Kodları

```
#include <Wire.h>
#include “ACROBOTIC_SSD1306.h”
#include <NewPing.h>

#define TRIGGER_PIN  15
#define ECHO_PIN     14
#define MAX_DISTANCE 400

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

#define T_B 493
```

```
int distance = 0;
int total = 0;

void setup() {
  pinMode(7,OUTPUT);
  pinMode(20,OUTPUT);
  pinMode(10,INPUT);
  Wire.begin();
  oled.init();
  oled.clearDisplay();

#if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
  clock_prescale_set(clock_div_1);
#endif
}

void loop() {

  delay(50);
  if(digitalRead(10) == 1){

    int measure=0;
    digitalWrite(7,HIGH);
    tone(20,T_B);
    delay(500);
    noTone(20);

    for (int i=0;i<20;i++){

      measure=sonar.ping_cm();
      total=total+measure;
      delay(50);
    }

    distance = total/20+6;
    digitalWrite(7,LOW);

    delay(1000);
    oled.clearDisplay();
    oled.setTextXY(2,1);
```

```
    oled.putString(“>Dijital Rule<”);
    oled.setTextXY(5,1);
    oled.putString(“Distance: “);
    oled.setTextXY(5,10);
    String string_distance=String(distance);
    oled.putString(string_distance);
    oled.setTextXY(5,12);
    oled.putString(“cm”);

    measure=0;
    distance=0;
    total=0;
  }
}
```

2.23. Air Piano

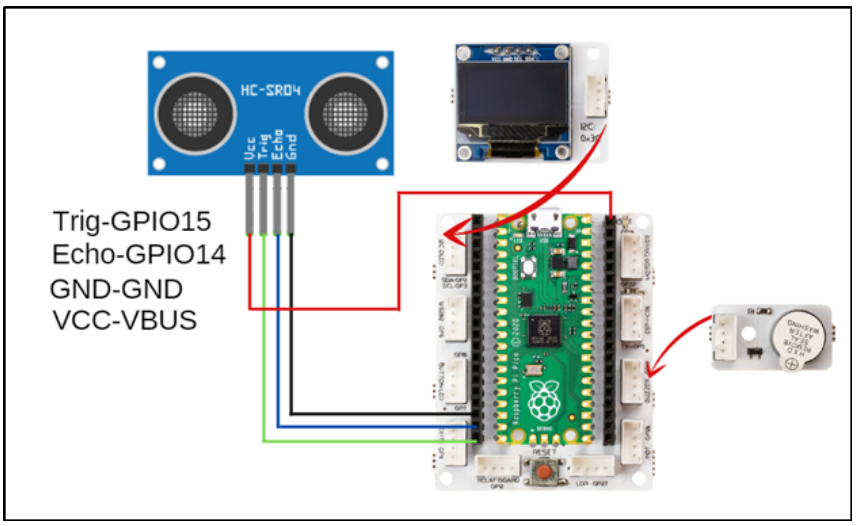
Elektronik teknolojisinin gelişimi ile üretimi zor, pahalı, yüksek kaliteli ses üreten müzik aletleri dijitalleşmiştir. Piyanolar bu enstrümanların başında gelmektedir. Dijital piyanoların her bir tuşu farklı frekansta elektrik sinyalleri üretir. Böylelikle hoparlörlerinden 88 farklı notayı çalabilmektedir. Dijital enstrümanların tuşlarının gecikme süresi, hoparlörün kalitesi, sesin çözünürlüğü gibi faktörler kaliteyi etkileyen faktörler olarak ortaya çıkmıştır. Elektro gitarlarda tuşlar yerine tellerdeki titreşimler dijitalleştirilir. Üflemeli enstrümanlarda ise ses çıkışına takılan yüksek çözünürlüklü mikrofonlar sayesinde çalınan notalar elektrik sinyallerine dönüştürülüp kayıt edilebilmektedir. Elektronik teknolojisindeki bu gelişim yüksek maliyetli müzik aletlerine ulaşım kolaylaştırmış, müzik eğitimi daha geniş çeşitliliğe kavuşmuş ve daha geniş kitleye yayılmıştır.

Bu projede PicoBricks ile 8 nota çalabilen basit bir piyano yapacağız. Bu piyanonun hoparlörü buzzer olacak. Piyanonun tuşlarının görevini ise ultrasonic sensör üstlenecek.

2.23.1. Proje Detayları ve Algoritma

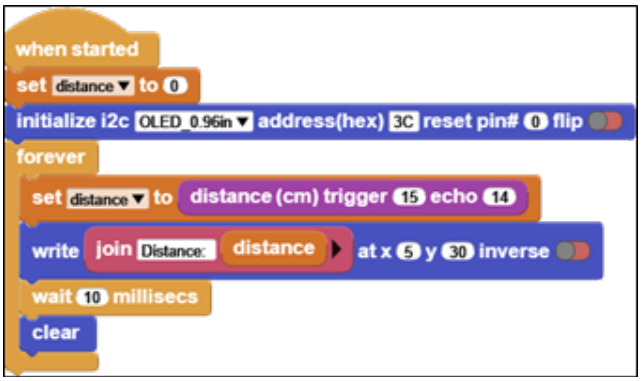
Bu projede HC-SR04 Ultrasonic mesafe sensörü ve PicoBricks üzerindeki buzzer modülünü kullanarak piyano uygulaması yapacağız. Mesafe sensöründen gelen değerlere göre buzzer ın farklı notalar çalmasını sağlayacak ve sensöre elimizi yaklaştırıp uzaklaştırarak melodiler oluşturacağız. Ayrıca oled ekrana anlık olarak mesafe çalınan nota bilgilerini yazdıracağız.

2.23.2. Wiring Diagram



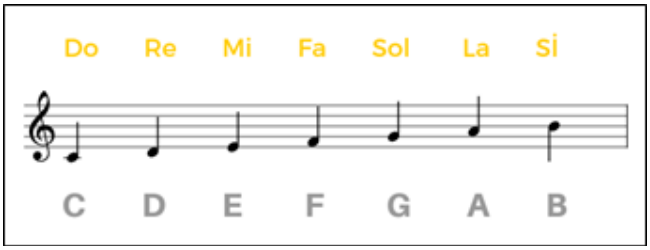
2.23.3. Projenin MicroBlocks ile Kodlanması

Projenin kodlarını MicroBlocks ile yazmaya başlarken öncelikle Add Library butonuna tıklayarak Tone kütüphanesini, Graphics kategorisindeki OLED Graphics kütüphanesini ve Sensing kategorisindeki Distance kütüphanelerini projemize import etmemiz gerekiyor. Kütüphaneleri ekledikten sonra HC-SR04 modülü için pin tanımlamalarını yapmalı ve sensörden gelen verilerin alınarak işlenmesi için distance adında bir değişken oluşturmalı ve sensör bilgilerini değişkene aktarmak ve OLED ekranda göstermek için gerekli kodları yazmalıyız.



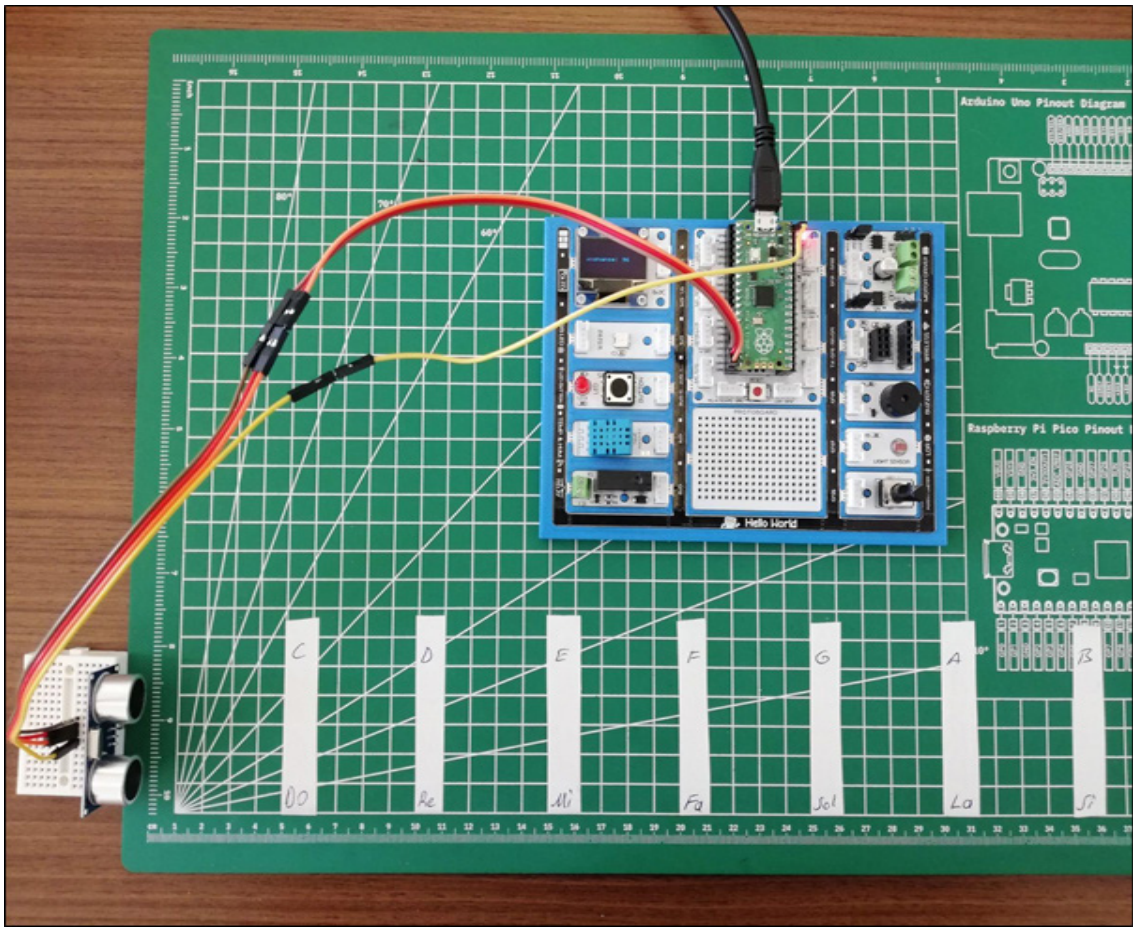
Her notanın bir harf karşılığı bulunmaktadır. Tone kütüphanesi de notaların harf karşılıklarını yazarak çalıştıracağımız bloklar içerir.

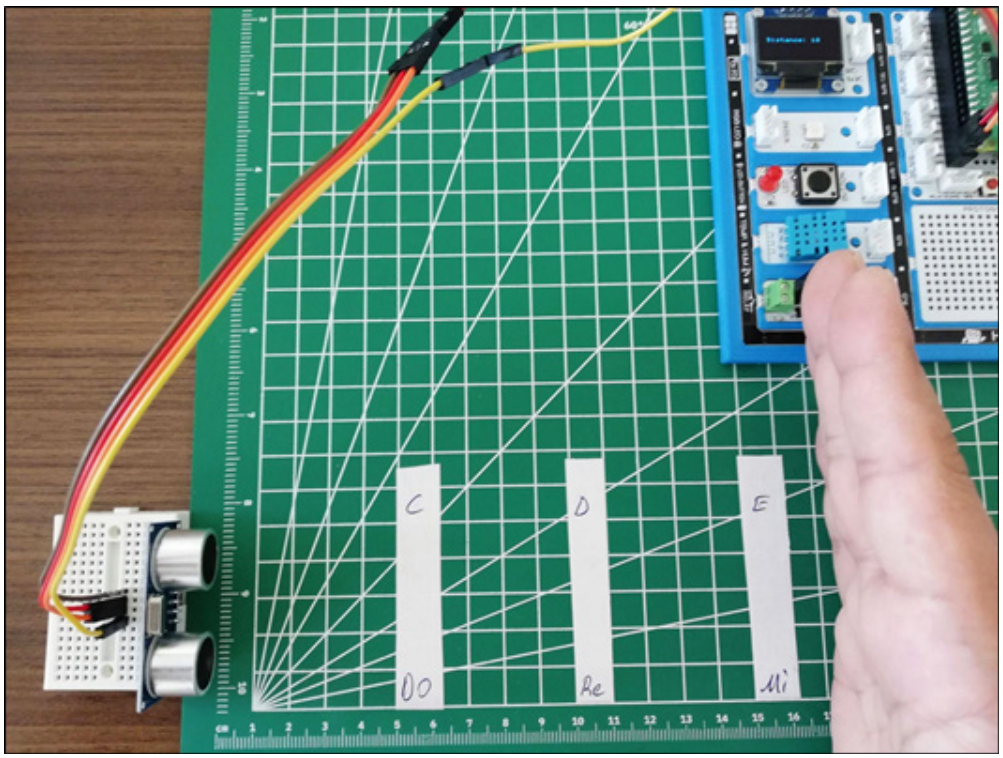
Çalmak istediğimiz notaların harf karşılıklarını Play bloğundaki alana yazmalıyız. distance değişkenindeki mesafe değerlerini karşılaştırarak 5 er cm aralıklarla notaların sırayla değişmesi için gerekli kodları yazdıktan sonra play bloğundaki oktav değerini dilediğiniz gibi değiştirerek projeyi çalıştırabilirsin.



Projenin MicrBlocks kodlarına ulaşmak için tıkla.

2.23.4. Projenin Çalıştırılması





2.23.5. Proje Önerisi

PicoBricks üzerinde bir adet anlık buton bulunmaktadır. Pico ya 7 adet buton bağlayarak her tuşa basıldığında farklı nota çalmasını sağlayabilir, oktav değeri ve vuruş süreleri için butonlar kullanabilir ve piyano projesini geliştirebilirsin.

2.23.6. Projenin MicroPython Kodları

```
from machine import Pin, PWM, I2C
from utime import sleep
import utime
from ssd1306 import SSD1306_I2C
import _thread

buzzer=PWM(Pin(20,Pin.OUT))
trigger = Pin(15, Pin.OUT)
echo = Pin(14, Pin.IN)

WIDTH  = 128
HEIGHT = 64
sda=machine.Pin(4)
scl=machine.Pin(5)
i2c=machine.I2C(0,sda=sda, scl=scl, freq=1000000)
oled = SSD1306_I2C(128, 64, i2c)

measure=0

def getDistance():
  trigger.low()
  utime.sleep_us(2)
  trigger.high()
  utime.sleep_us(5)
  trigger.low()
  while echo.value() == 0:
    signaloff = utime.ticks_us()
  while echo.value() == 1:
    signalon = utime.ticks_us()
  timepassed = signalon - signaloff
  distance = (timepassed * 0.0343) / 2
  return distance
```

```
def airPiano():
    while True:
        global measure

        if measure>5 and measure<11:
            buzzer.duty_u16(4000)
            buzzer.freq(262)
            sleep(0.4)

        elif measure>10 and measure<16:
            buzzer.duty_u16(4000)
            buzzer.freq(294)
            sleep(0.4)

        elif measure>15 and measure<21:
            buzzer.duty_u16(4000)
            buzzer.freq(330)
            sleep(0.4)

        elif measure>20 and measure<26:
            buzzer.duty_u16(4000)
            buzzer.freq(349)
            sleep(0.4)

        elif measure>25 and measure<31:
            buzzer.duty_u16(4000)
            buzzer.freq(392)
            sleep(0.4)

        elif measure>30 and measure<36:
            buzzer.duty_u16(4000)
            buzzer.freq(440)
            sleep(0.4)

        elif measure>35 and measure<41:
            buzzer.duty_u16(4000)
            buzzer.freq(494)
            sleep(0.4)
        else:
            buzzer.duty_u16(0)
```

```
_thread.start_new_thread(airPiano, ())

while True:
    measure=int(getDistance())
    oled.text("Distance " + str(measure)+ " cm", 5,30)
    oled.show()
    sleep(0.01)
    oled.fill(0)
    oled.show()
```

2.23.7. Projenin Arduino C Kodları

```
#include <Wire.h>
#include "ACROBOTIC_SSD1306.h"
#include <NewPing.h>

#define TRIGGER_PIN  15
#define ECHO_PIN     14
#define MAX_DISTANCE 400

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

#define T_C 262
#define T_D 294
#define T_E 330
#define T_F 349
#define T_G 392
#define T_A 440
#define T_B 493

const int Buzzer = 20;

void setup() {
  pinMode(Buzzer,OUTPUT);

  Wire.begin();
  oled.init();
  oled.clearDisplay();

#if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
```

```
  clock_prescale_set(clock_div_1);
#endif
}

void loop() {

  delay(50);
  int distance=sonar.ping_cm();

  if(distance>5 & distance<11)
  {
    tone(Buzzer,T_C);
  }

  else if(distance>10 & distance<16)
  {
    tone(Buzzer,T_D);
  }

  else if(distance>15 & distance<21)
  {
    tone(Buzzer,T_E);
  }

  else if(distance>20 & distance<26)
  {
    tone(Buzzer,T_F);
  }

  else if(distance>25 & distance<31)
  {
    tone(Buzzer,T_G);
  }

  else if(distance>30 & distance<36)
  {
    tone(Buzzer,T_A);
  }

  else if(distance>35 & distance<41)
  {
```

```
    tone(Buzzer,T_B);
  }

  else
  {
    noTone(Buzzer);
  }

  oled.clearDisplay();
  oled.setTextXY(2,4);
  oled.putString("Distance: ");
  oled.setTextXY(4,6);
  String string_distance=String(distance);
  oled.putString(string_distance);
  oled.setTextXY(4,8);
  oled.putString("cm");
}
```

2.24. Labirent Çözen Robot

Kodlama eğitimi programlama dillerinin tarihi kadar eskidir. Günümüzde kodlama eğitimini yaygınlaştırmak, heyecanlı ve eğlenceli hale getirmek için farklı ürünler kullanılmaktadır. Bunların başında eğitsel robotlar gelmektedir. Robotları hazırlamak ve kodlamak çocukların mühendislik ve kodlama becerilerini geliştirmektedir. Kodlama eğitimini yaygınlaştırmak, öğretmen ve öğrencileri teşvik etmek için kurum ve kuruluşlar tarafından robotik yarışmalar düzenlenmektedir. Bu yarışmalardan biri de Labirent Çözen Robot yarışmalarıdır. Bu robotlar önce labirentte dolanarak varış noktasını öğrenir ve başlangıç noktasına geri dönerler. Daha sonra labirente tekrar başladıklarında en kısa yoldan en hızlı şekilde varış noktasına ulaşmaya çalışırlar. Robotlar labirenti öğrenirken mesafe sensörlerinden faydalanırlar. Kızılötesi ya da ultrasonic sensörler bu robotlarda görev almaktadır.

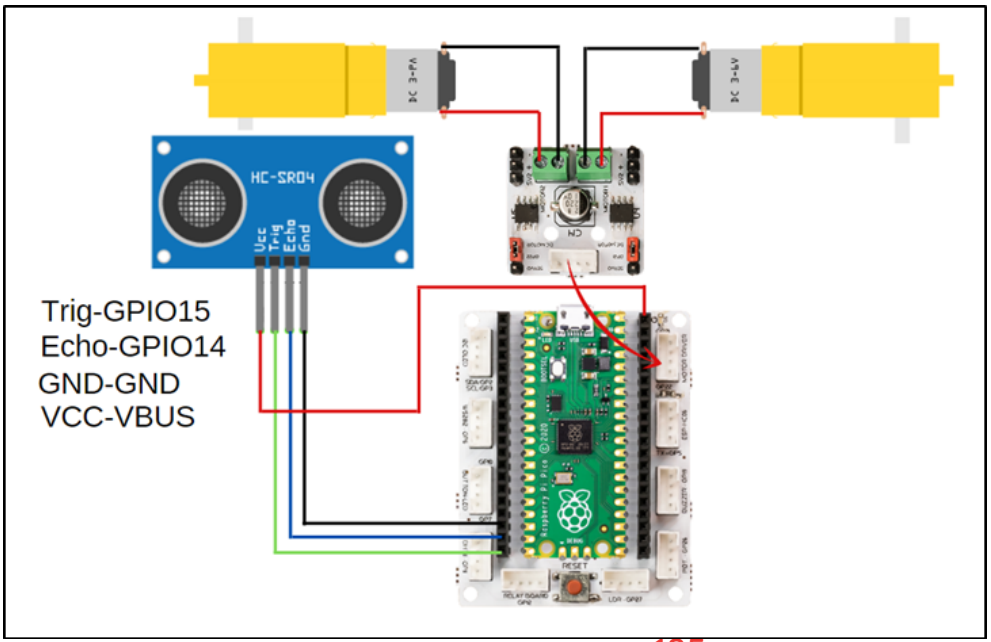
Ev ve işyerlerinde kullanılan akıllı robot süpürgeler de labirent çözen robotların algoritmalarına yakın mantıkla çalışmaktadır. Engelleri sürekli kontrol edip haritalayan algoritmaları sayesinde süpürme işini eksiksiz ve çarpmadan yapmaya çalışırlar. Akıllı süpürgelerin çoğunda mesafe ölçme ve engel algılamak için lazer ile yüksek hassasiyetli ölçüm yapan LİDAR ve kızılötesi sensörler görev almaktadır.

Bu projede PicoBricks ile labirent çözen robot yarışmalarına hazırlanabileceğiniz basit bir robot yapacağız.

2.24.1. Proje Detayları ve Algoritma

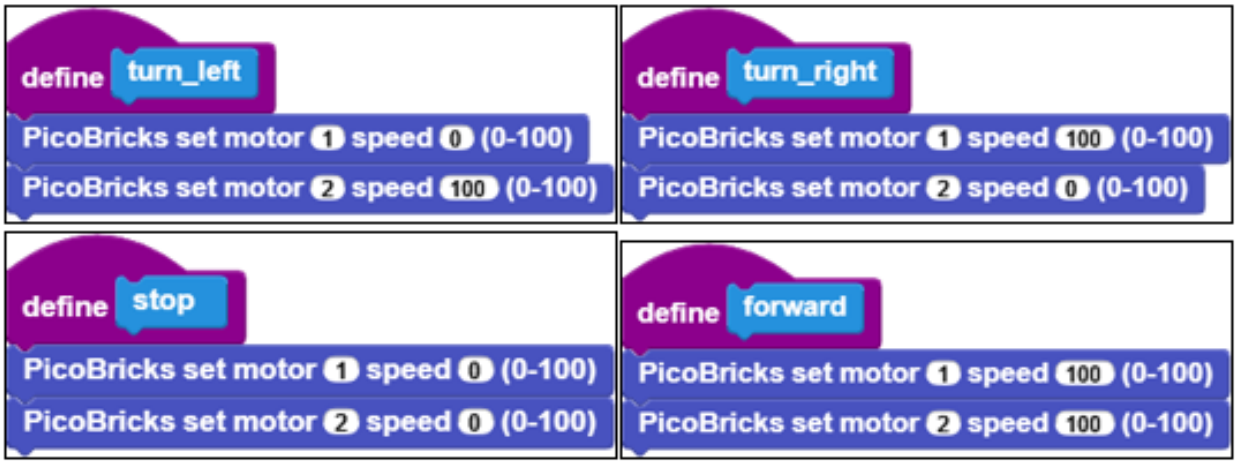
Labirent çözen robot projesinde setin içerisinden çıkan 2WD robot araba kitini kullanacağız. Robotun önündeki mesafeyi algılayarak hareketlerine kendi kendine karar verebilmesi için HC-SR04 ultrasonic mesafe sensörü kullanacağız. Labirent içerisinde robot araba önündeki mesafeyi algılayarak önü boş ise ilerleyecek. Eğer mesafe 5 cm den küçük ise araba sağa dönecek, tekrar mesafeyi ölçecek eğer sağ taraftaki mesafe 5 cm den büyük ise ilerleyerek yoluna devam edecek, küçükse sola dönerek ilerleyecek. Bu şekilde sağa ve sola dönerek labirent içerisinde boş olan yollardan aracın ilerlemesini ve labirentten çıkmasını sağlayacağız.

2.24.2. Wiring Diagram

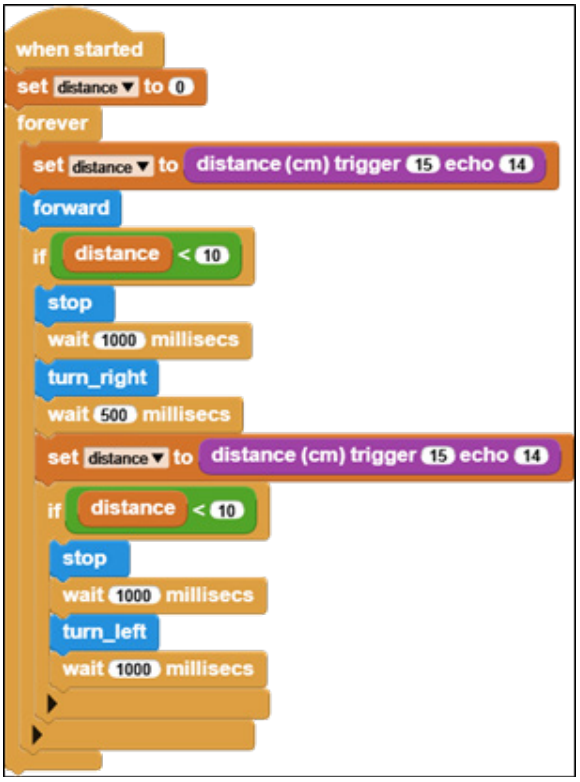


2.24.3. Projenin MicroBlocks ile Kodlanması

Projenin kodlarını hazmaya robot arabanın hareketlerini yapması için gerekli kodları yazarak başlayabiliriz. Bunun için forward, turn right, turn left ve stop adında dört adet blok (fonksiyon) tanımlamalıyız. Bu fonksiyonlarda motorların dönmesi için gerekli blokları yerleştireceğiz.



Fonksiyonları tanımladıktan sonra Sensing kategorisinden Distance uzantısını ekleyerek HC-SR04 ultrasonic mesafe sensörünün trig ve echo pin numaralarını distance bloğuna yazıyor ve distance adında değişken oluşturarak sürekli olarak bu değişkenin değerinin sensörden gelen değer olmasını sağlıyoruz. Daha sonra sensör değeri 5 cm den küçükse arabanın durması ve sağa dönmesini sağlıyoruz. Sağ taraf boş ise robot araba ileri doğru giderek yoluna devam edecektir. Fakat sağa döndüğünde sağ tarafın dolu olma ihtimaline karşı tekrar mesafe ölçerek mesafe yine 5 cm den küçükse aracın sol tarafa dönerek ilerlemesi için gerekli kodları yazıyoruz. Bu sayede araç boş olan yola doğru giderek labirenti tamamlayacaktır.

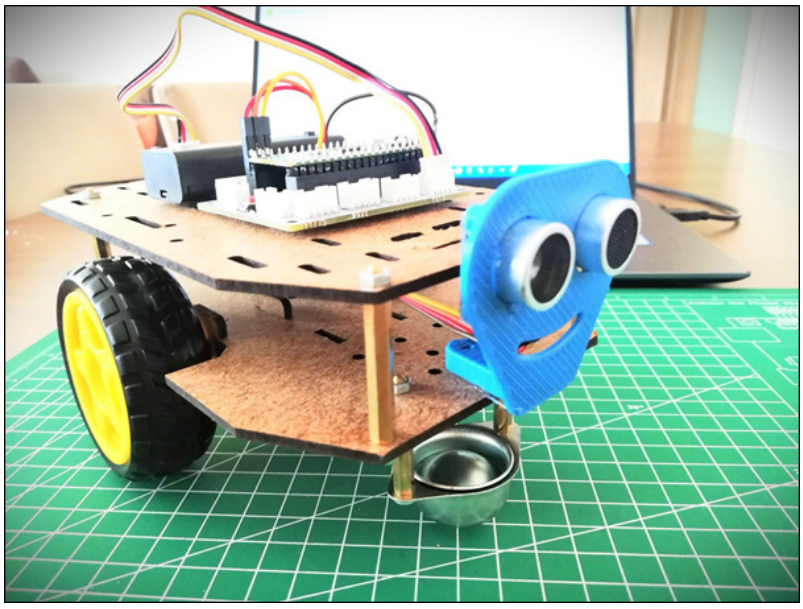


Projenin MicrBlocks kodlarına ulaşmak için tıkla.

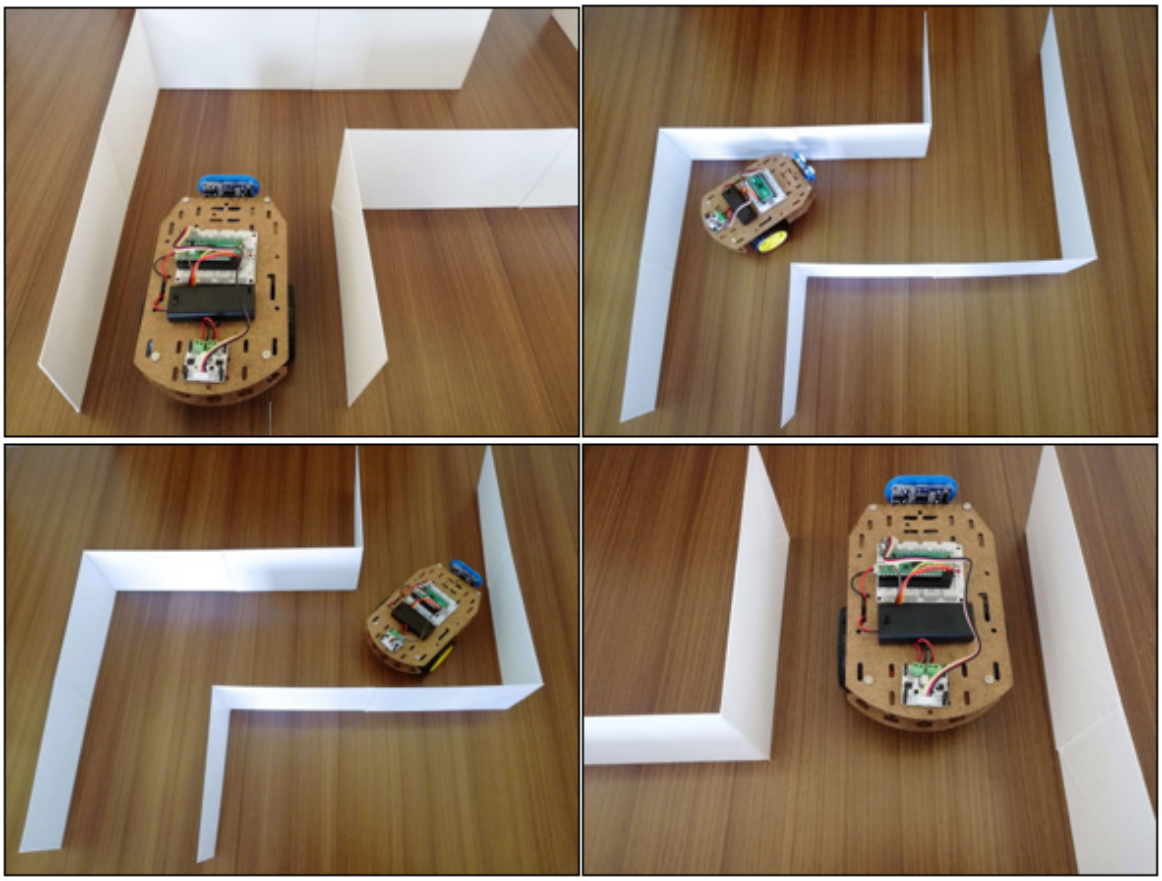
wait bloğuyla motorların çalışma süreleri belirlenmektedir. Pilin doluluk durumuna göre motorların çalışma süreleri farklı olacağı için projeyi yaparken bekleme sürelerini kendi robot aracına göre optimize etmelisin.

2.24.4. Projenin Yapım Aşamaları

Bu projede 2.2.18. bölümündeki ses kontrollü araba projesinde 2WD robot araba montajı için yaptığın adımları takip ederek labirent çözen robot arabayı inşa edebilirsin. HC05 bluetooth modülünü bu projede kullanmayacağız. HC-SR04 ultrasonic mesafe sensörünü robota monte edebilmek için gerekli parçayı buradaki bağlantıdan indirerek 3D yazıcıdan baskı alabilir ve araca montaj yapabilirsin.



Robo montajını tamamladıktan sonra labirent inşası için karton kutular kullanabilirsin. Mukavvadan duvarlar yapabilir ya da 3D baskı duvarlar kullanarak sıcak silikon ile duvarları birleştirip labirenti inşa edebilirsin.



2.24.5. Proje Önerisi

Bu projede HC-SR04 ultrasonic mesafe sensörünü kullanarak 2WD robot arabanın labirent içerisinde boş olan yollardan ilerleyerek labirentten çıkmasını sağladık. Servo motor kullanarak HC-SR04 modülünü sağa sola dönderebilir ve boş olan yolları aracın dönmesine gerek kalmadan araca algılatabilir ve hareketlerini daha hızlı yapmasını sağlayabilirsin. Ya da 3 adet HC-SR04 ultrasonic mesafe sensörünü aracın ön sağ ve sol kısımlarına montajlayarak 3 yöndeki mesafeleri aynı anda ölçüp aracın boş olan yola doğru yönelmesini sağlayarak projeyi geliştirebilirsin.

2.24.6. Projenin MicroPython Kodları

```
from machine import Pin,
from utime import sleep
import utime

trigger = Pin(15, Pin.OUT)
echo = Pin(14, Pin.IN)

m1 = Pin(21, Pin.OUT)
m2 = Pin(22, Pin.OUT)

m1.low()
m2.low()
def getDistance():
  trigger.low()
  utime.sleep_us(2)
  trigger.high()
  utime.sleep_us(5)
  trigger.low()
  while echo.value() == 0:
    signaloff = utime.ticks_us()
  while echo.value() == 1:
    signalon = utime.ticks_us()
  timepassed = signalon - signaloff
  distance = (timepassed * 0.0343) / 2
  return distance
measue=0
while True:
   measure=int(getDistance())
   m1.high()
   m2.high()
```

```
    if measure<10:
        m1.low()
        m2.low()
        sleep(1)
        m1.high()
        m2.low()
        sleep(0.5)
        measure=int(getDistance())
        if measure<10:
            m1.low()
            m2.low()
            sleep(1)
            m1.low()
            m2.high()
            sleep(1)
```

2.24.7. Projenin Arduino C Kodları

```
#include <NewPing.h>

#define TRIGGER_PIN  15
#define ECHO_PIN     14
#define MAX_DISTANCE 400

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

void setup() {
  pinMode(21,OUTPUT);
  pinMode(22,OUTPUT);
}

void loop() {

  delay(50);
  int distance=sonar.ping_cm();
  Forward();

  if(distance<5){

    Stop();
```

```
    delay(1000);
    Turn_Right();
    delay(500);
    int distance=sonar.ping_cm();

    if(distance<5){

      Stop();
      delay(1000);
      Turn_Left();
      delay(1000);
    }
  }
}

void Forward(){
  digitalWrite(21,HIGH);
  digitalWrite(22,HIGH);
}
void Turn_Left(){
  digitalWrite(21,LOW);
  digitalWrite(22,HIGH);
}
void Turn_Right(){
  digitalWrite(21,HIGH);
  digitalWrite(22,LOW);
}
void Stop(){
  digitalWrite(21,LOW);
  digitalWrite(22,LOW);
}
```

2.25. Akıllı Sera

Küresel ısınmanın etkisiyle iklimlerde meydana gelen hızlı değişimler tarım alanındaki faaliyetlerde verimin düşmesine sebep olmaktadır. 1500'lü yıllarda Daniel Barbaro tarihte bilinen ilk serayı yaptı. Seralar kontrol edilebilen hava, su, ısı ve ışık koşulları sunabilen bitki yetiştirmeye elverişli ortamlardır. Seralarda ısıyı dengelemek için ısıtıcılar , sulama yapmak için elektrikli su motorları, nem oranını ayarlamak ve tozlaşmayı sağlayabilmek için fanlar kullanılmaktadır. Teknolojinin de gelişmesiyle üretici seranın durumunu istediği yerden telefonuyla takip edebilmekte ve yapılması gereken işleri yapabilmektedir. Bu teknolojinin genel adı Nesnelerin İnterneti (IOT) dir.

Seralarda sıcaklık, nem ve oksijen miktarını ölçmek için özel sensörler kullanılır. Ayrıca sulamaya karar vermek için toprak nemini ölçen özel sensörler kullanılmaktadır. Sulama verimliliğini artırmak için ise elektronik kontrollü damla sulama sistemleri kullanılmaktadır.

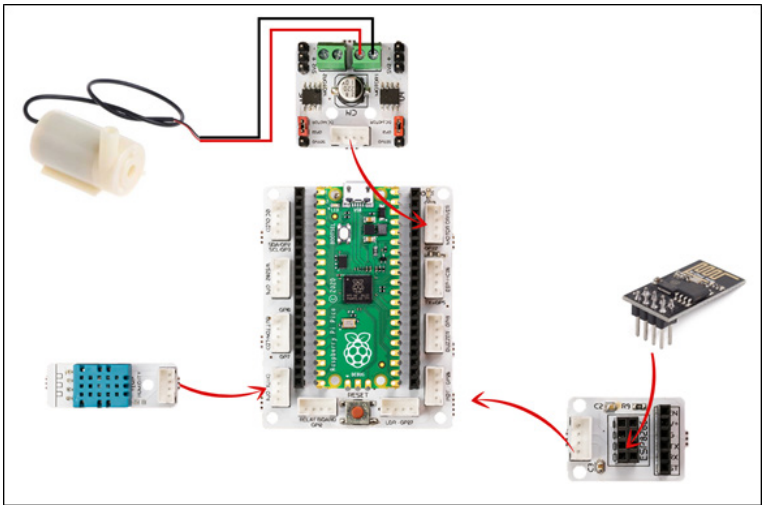
Bu projede PicoBricks ile IOT teknolojisini barındıran basit bir sera hazırlayacağız. PicoBricks'i ESP8266 wifi modülü ile bu serada kullanacağız. Bu sayede serayı internet üzerinden takip edebildiğimiz bir nesne haline getireceğiz.

2.25.1. Proje Detayları ve Algoritma

Hazırlayacağın sera maketinin içinde toprak nemi sensörü, üstten asılı olarak DHT11 sıcaklık ve nem sensörü bulunacaktır. Maketin dışındaki su tankının içine dalgıç pompa konulacak, pompanın ucundan çıkan hortum ise seradaki toprağa gidecektir. Picoboard sera maketinin dışında uygun bir yere yerleştirilecektir.

Picobricks başladığında ESP 8266 wifi modülü sayesinde wifi yayını yapmaya başlar. Aynı ağa bağlı akıllı telefondan Esp8266'nın IP adresini girdiğimizde Serayı kontrol edeceğimiz web sayfası ile karşılaşırız. Burada sıcaklık ve nem değerlerini görebiliriz. Dilersek sulama komutu vererek sulama işlemini başlatabiliriz.

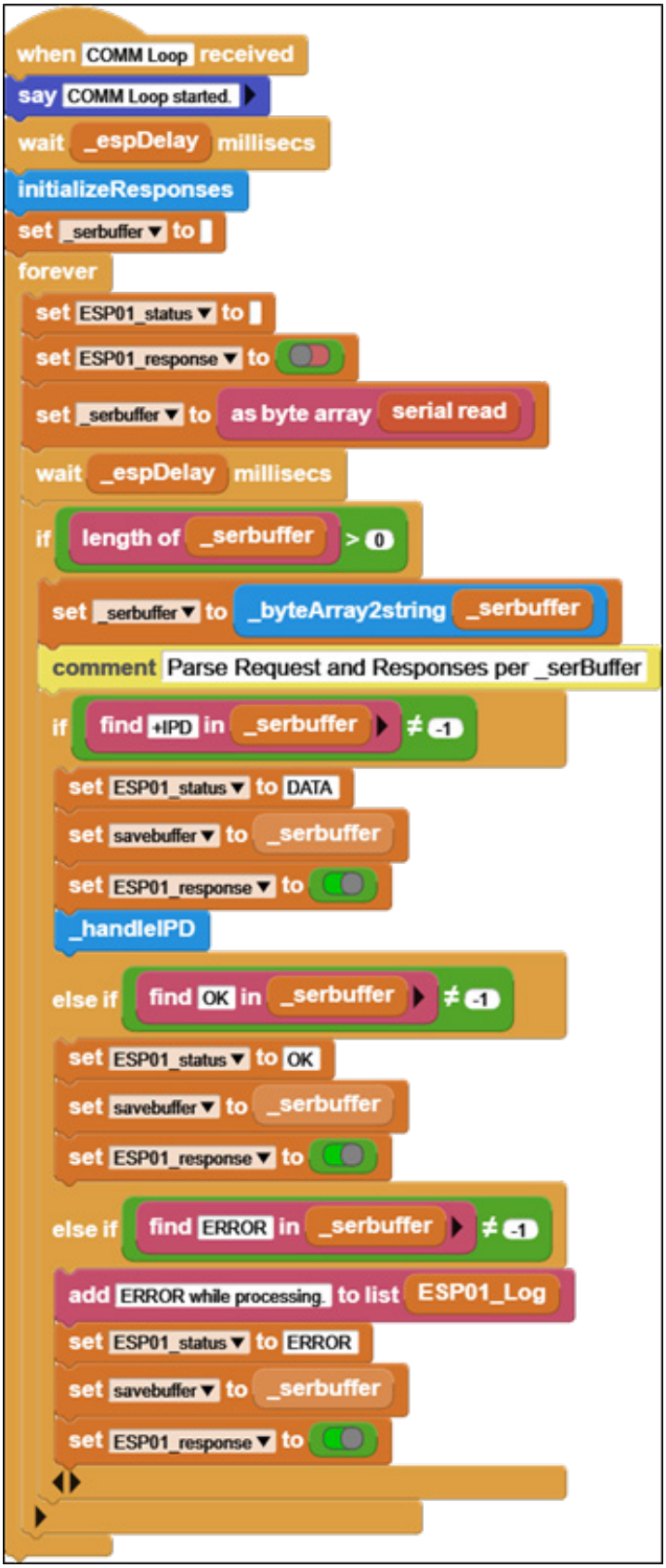
2.25.2. Wiring Diagram



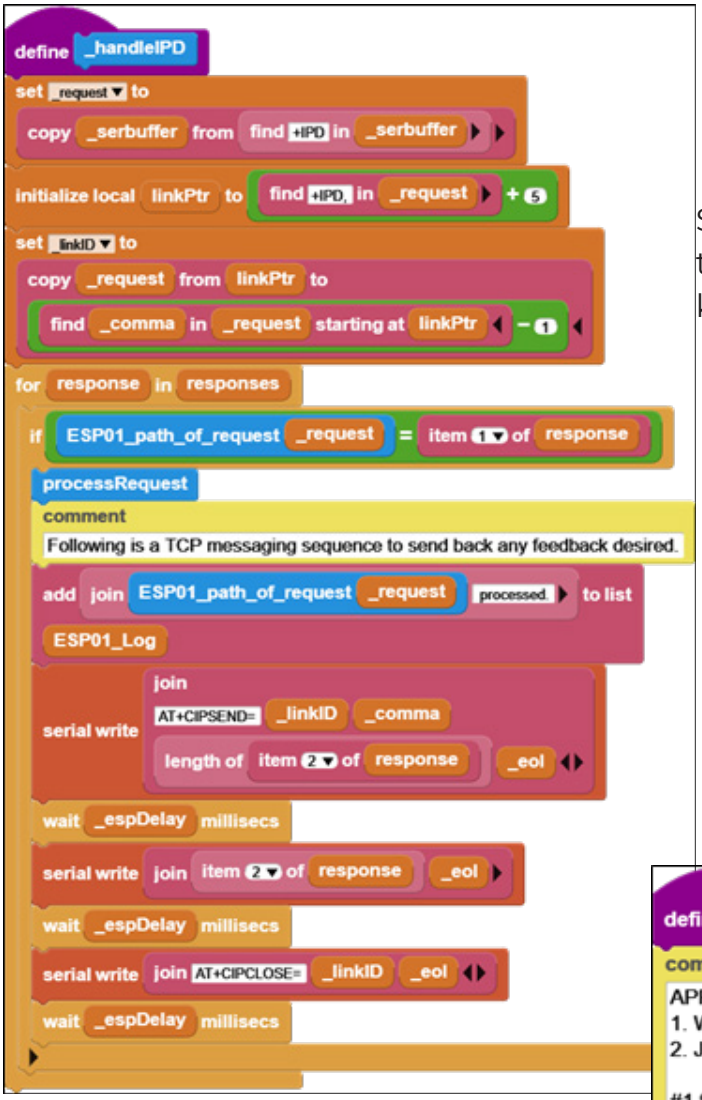
2.25.3. Projenin MicroBlocks ile Kodlanması

Picobricks başladığında Sera'nın bağlanacağı Wifi ayarlarının yapıldığı ve ESP8266'nın kablosuz ağa bağlanarak IP almasını sağlayan kod blokları aşağıdaki gibidir. ESP01_connect_to bloğuna Akıllı serayı bağlayacağınız kablosuz ağın SSID ve Şifre bilgilerini yaz.

Seri iletişimin sürekli denetlendiği isteklerin ve yanıtların yönlendirildiği kod bloğu.

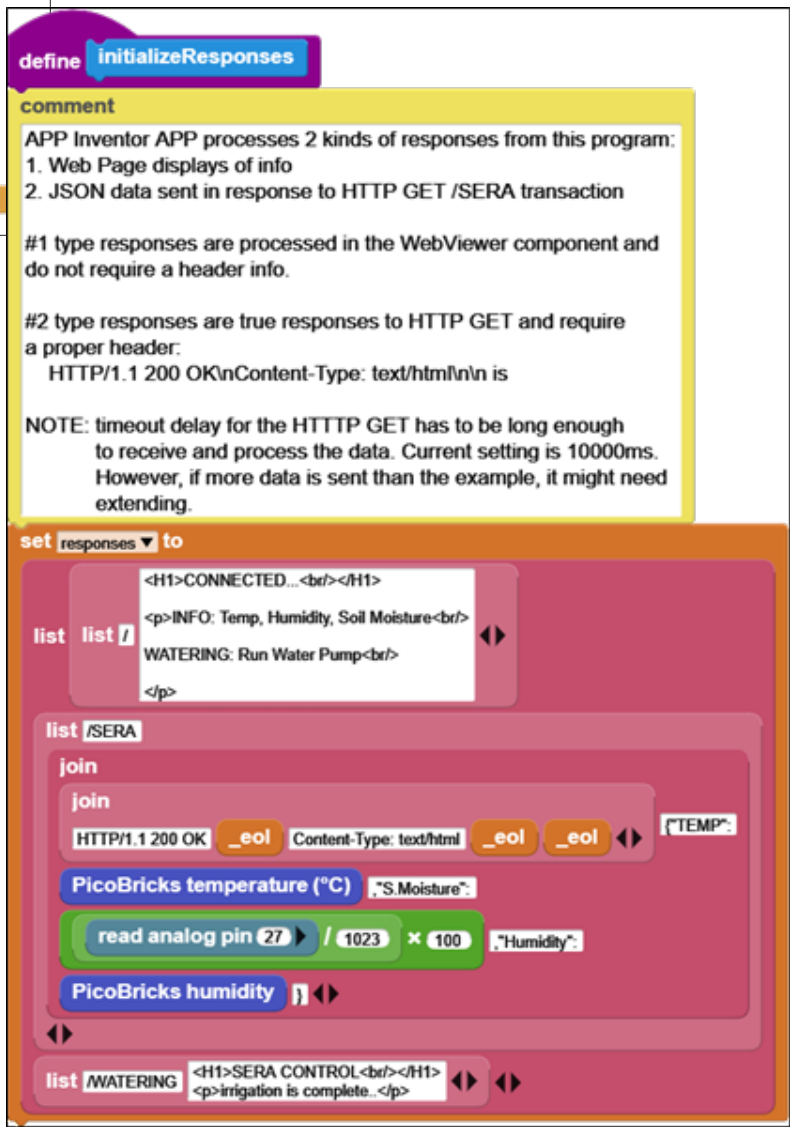


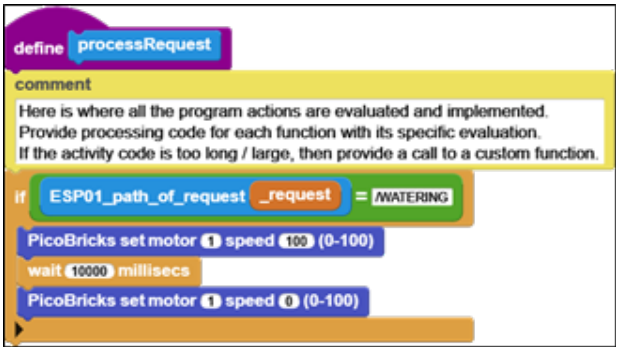
Sorguları yakalayıp cevaba yönlendiren kod bloğu.



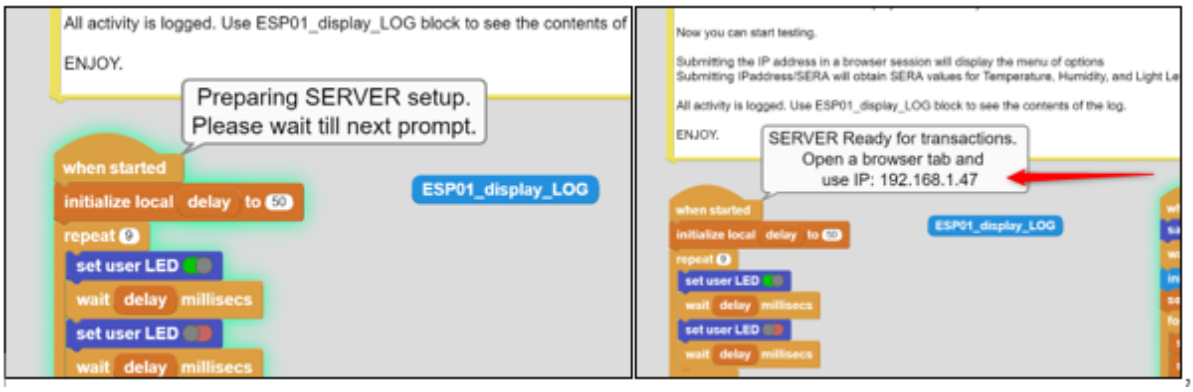
Sera bilgilerini ve sulama işleminin tamamlandığını cevap olarak gönderen kod bloğu.

Sulama komutunun dalgıç pompayı çalıştırdığı kod bloğu.

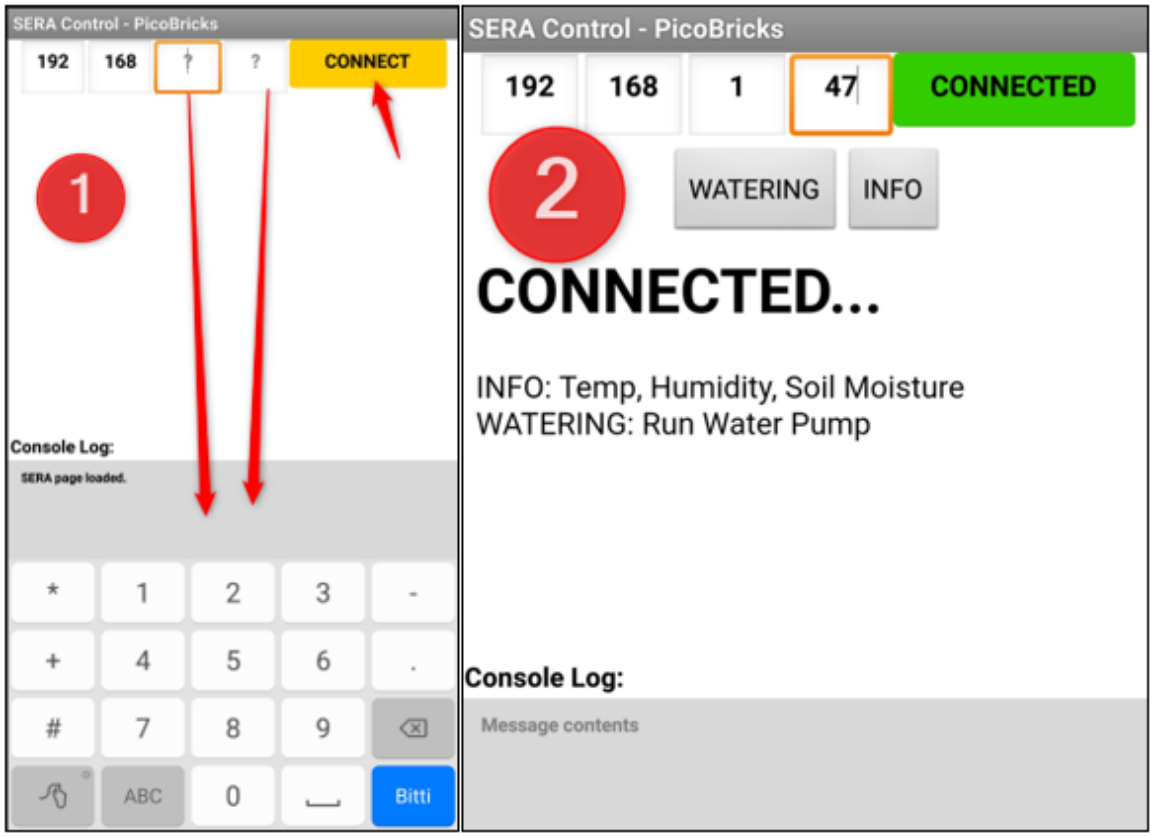




Picobricks'i MikcroBlocks'a bağlayıp kodları çalıştırdıktan yaklaşık 20 saniye sonra Sera wifi ağına bağlanacak ve konuşma baloncuğu içinde atanan IP adresi görünecektir. Bu IP adresini Mobil uygulamaya yazmak için not et.

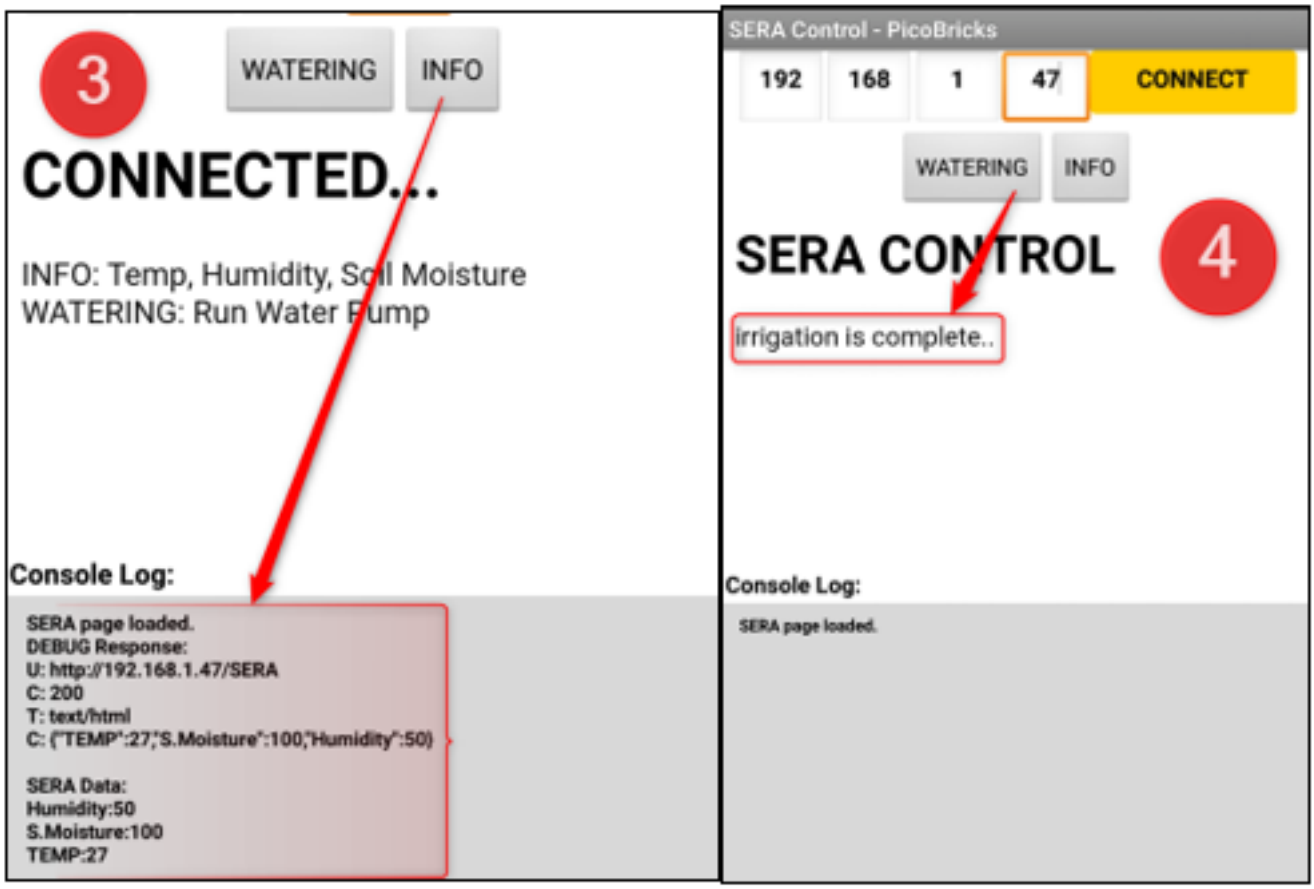


1- Sera Control uygulamasını cihazınıza yükle. MicroBlocks kodlarının belirttiği IP adresini son iki hanesini "?" olan kutulara yazıp CONNECT butonuna tıkla. 2 nolu görseldeki ekran ile karşılaştığında Sera ile mobil uygulama Wifi üzerinden birbirine bağlanmış demektir.



3- INFO butonuna tıklarsan 20 saniye sonra Seradaki sensörlerin okuduğu veriler consolda görüntülenir.

4- WATERING butonuna basarsan 10 saniye içinde dalgıç pompa çalışır v e 10 saniye sonra durur. Ardından sulama işleminin bittiği ifadesi ekranda belirir.



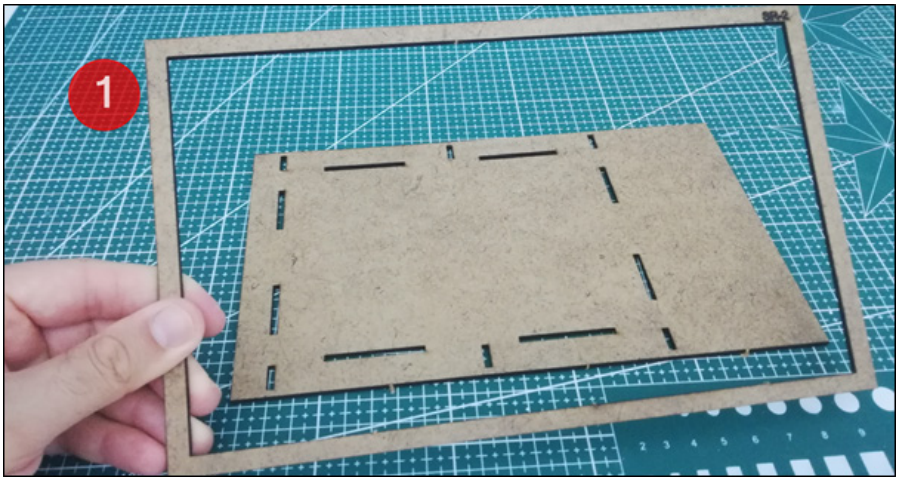
Projenin kodlarına erişmek için tıkla.

Gerekli Android uygulamasını indirmek için tıkla.

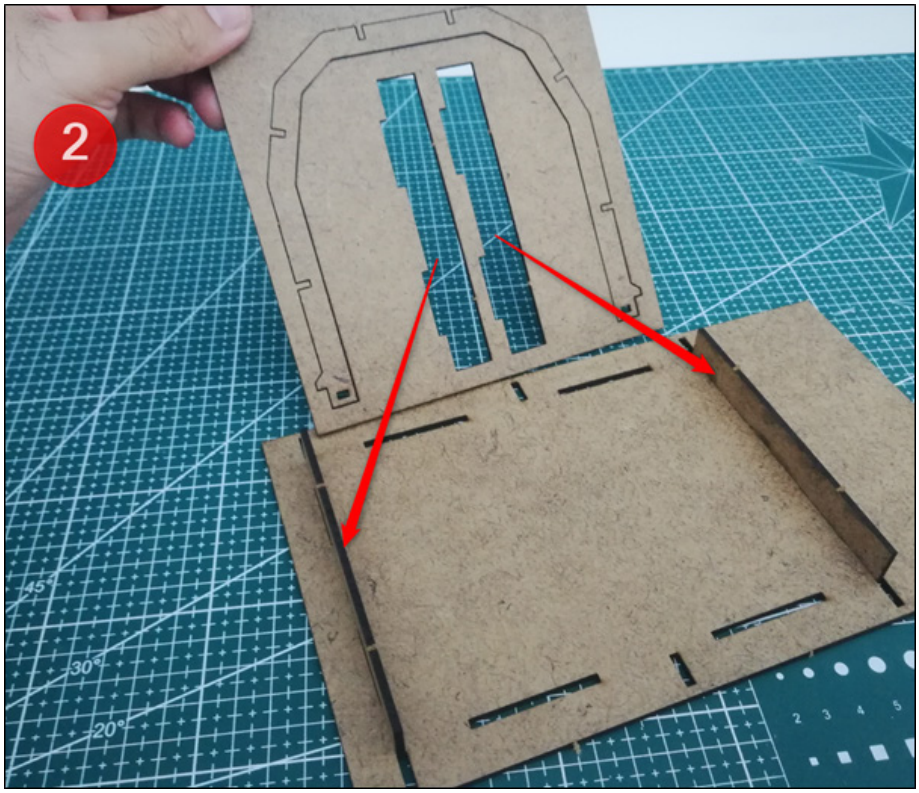
MIT App Inventor Dosyası için tıkla.

2.25.4. Projenin Yapım Aşamaları

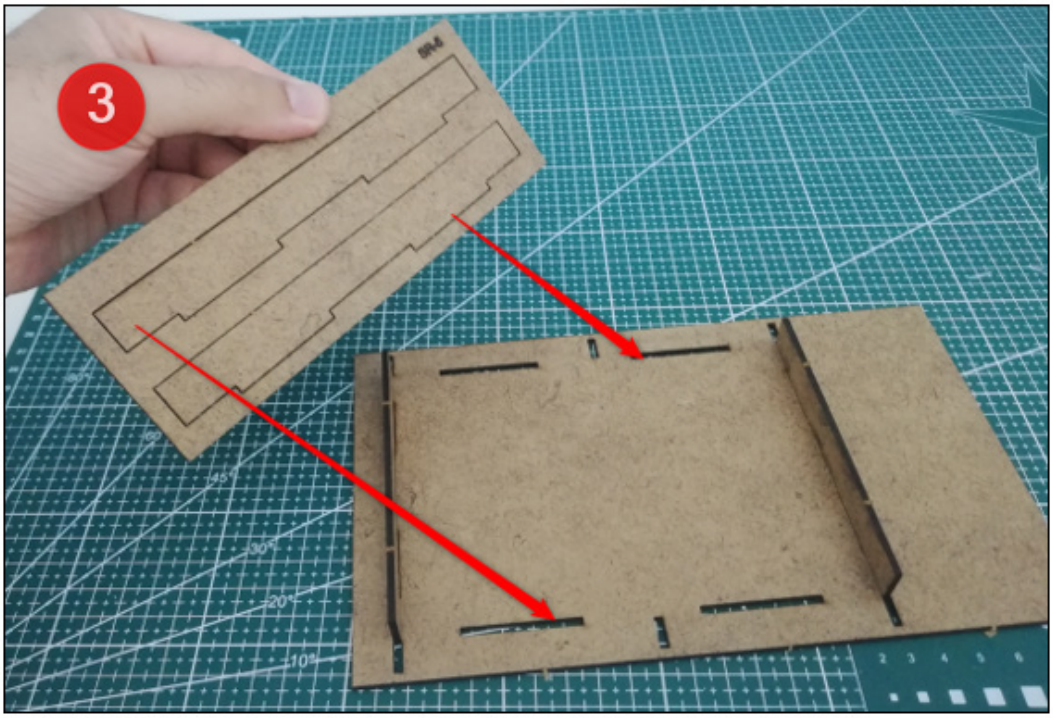
1-Sera kitindeki SR-2 kodlu parçadan Sera maketin zeminini ayır.



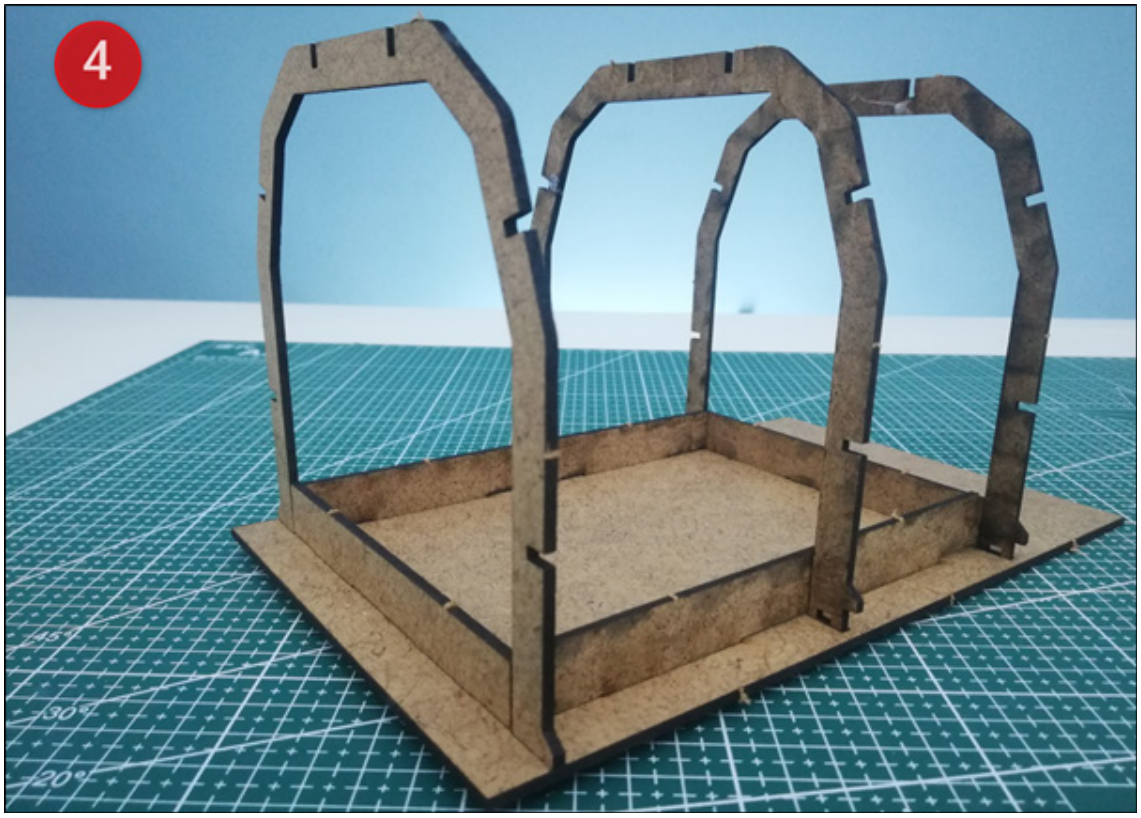
2- SR-3 parçasının ortasındaki parçaları Seranın zeminine tak.



3- SR-4 parçasındaki sera iç duvarlarını çıkarıp seranın zeminine tak.



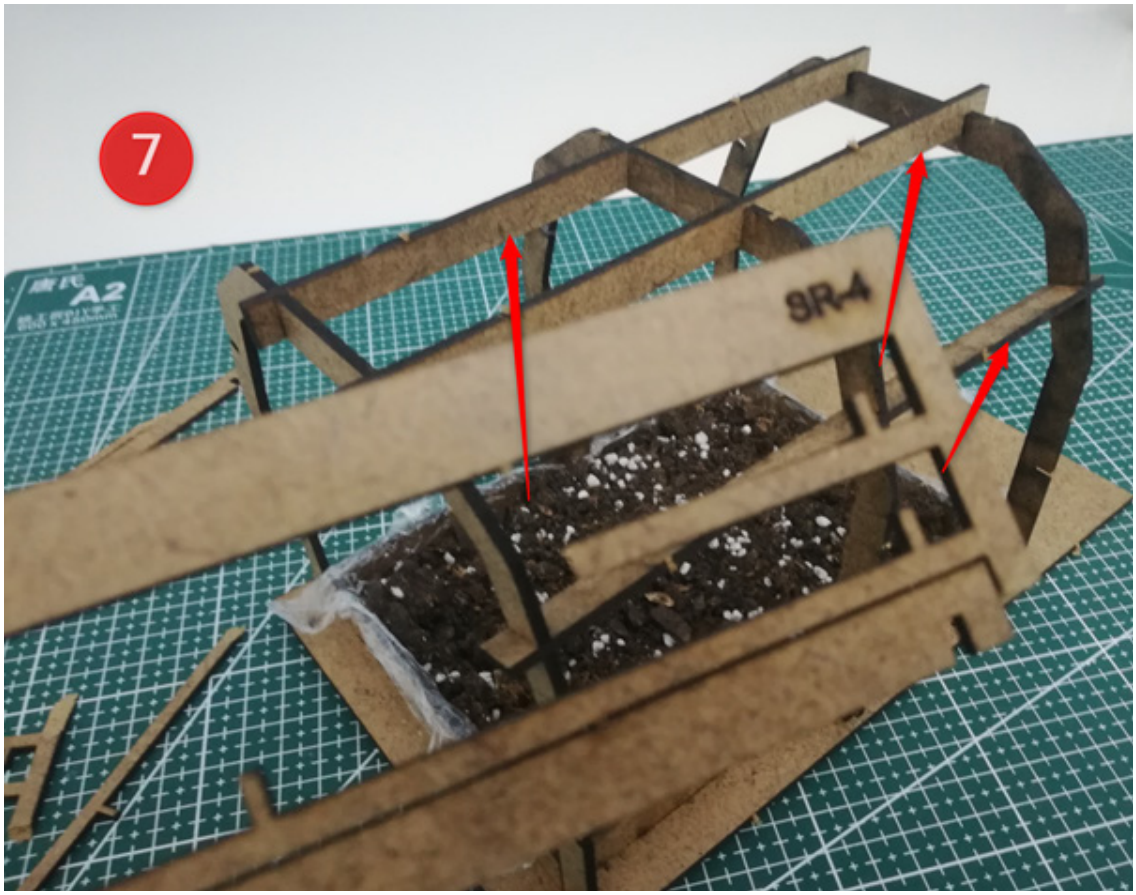
4- SR-1 ve SR-3 deki Sera kemerlerini söküp sera zeminine yerleştir.



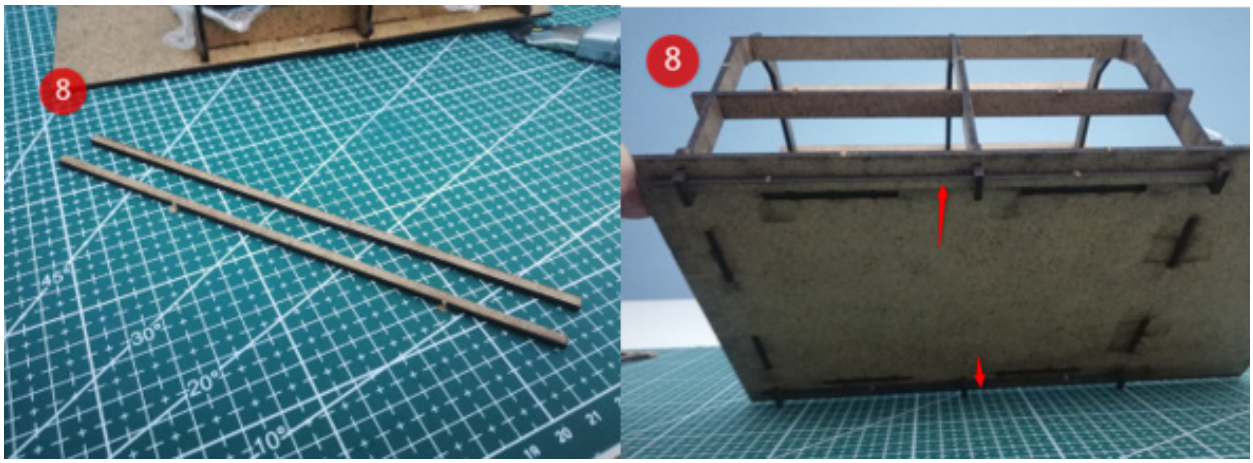
5-Toprak konulacak dikdörtgen bölgeye streç film ile kapla. Sulama sonrası maket parçalarını korumuş olursun.

6- Bitki toprağını seranın içine dök. Boş yer kalmayacak şekilde doldur.

7- SR-4 deki parçaları seranın üzerindeki çentiklere yerleştir.



8-SR-4 de kalan iki ince düz parçayı seranın alt taraftan iki kenarındaki deliklerden geçir. Bu işlem seranın daha sağlam olmasını sağlar.

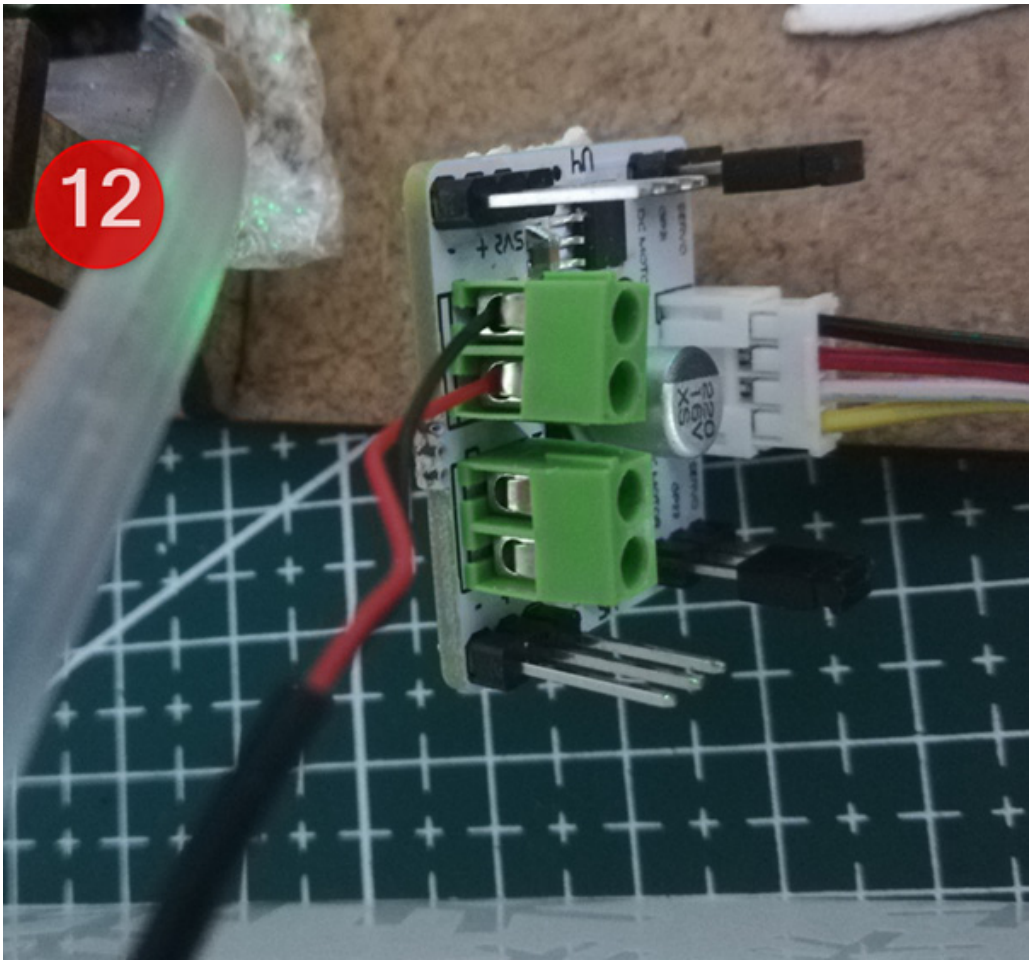


9- Dalgıç pompasına hortumu geçir. Serada damla sulama sistemine benzer bir sulama sistemi kuracaksın. Toprağın sulanmasını istediğin yerlerinden hortumu geçir. Hortumu su tankına ucu yetişecek kadar kesmeyi unutmayın.

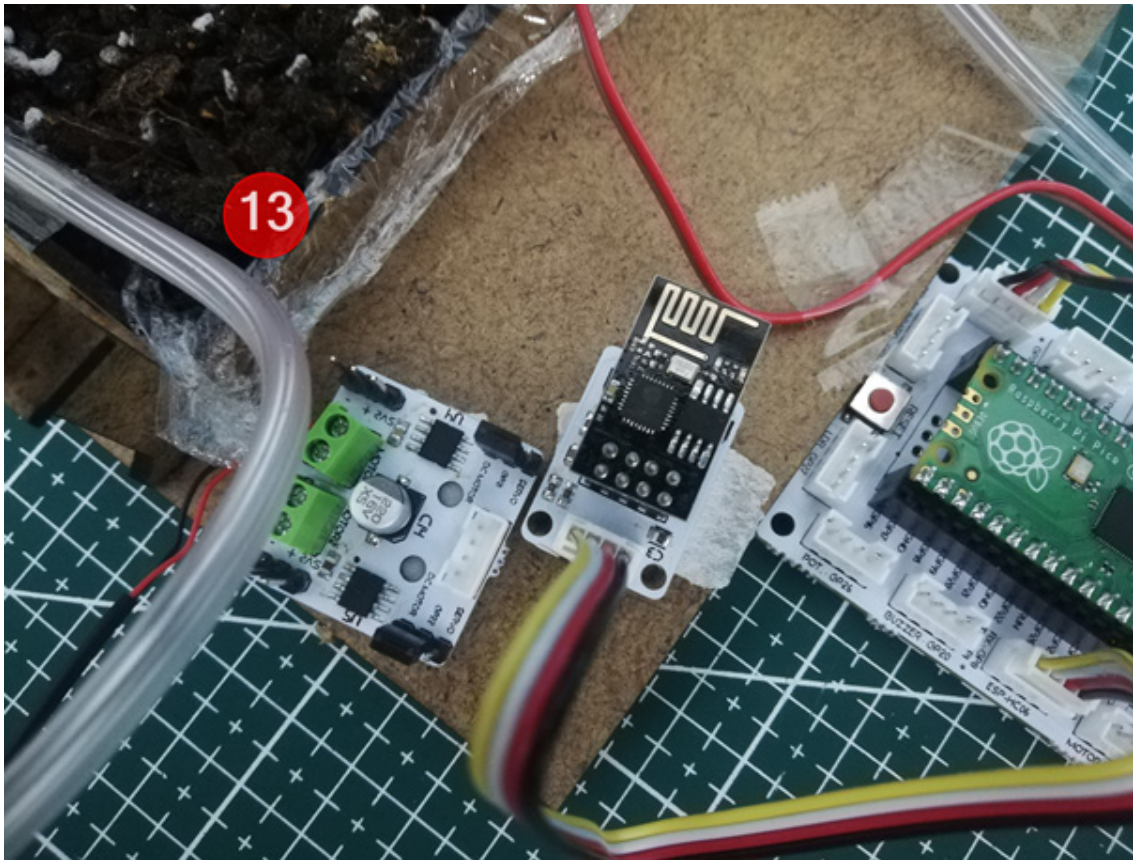
10- Erkek jumper kablo ucu ile ya da maket bıçağı ile hortum üzerinde suyun damlayabileceği kadar delikler aç.

11- DHT 11 sıcaklık ve nem sensörünü sera maketinin üstüne Toprak Nemi Sensörünü toprağın içine yerleştir.

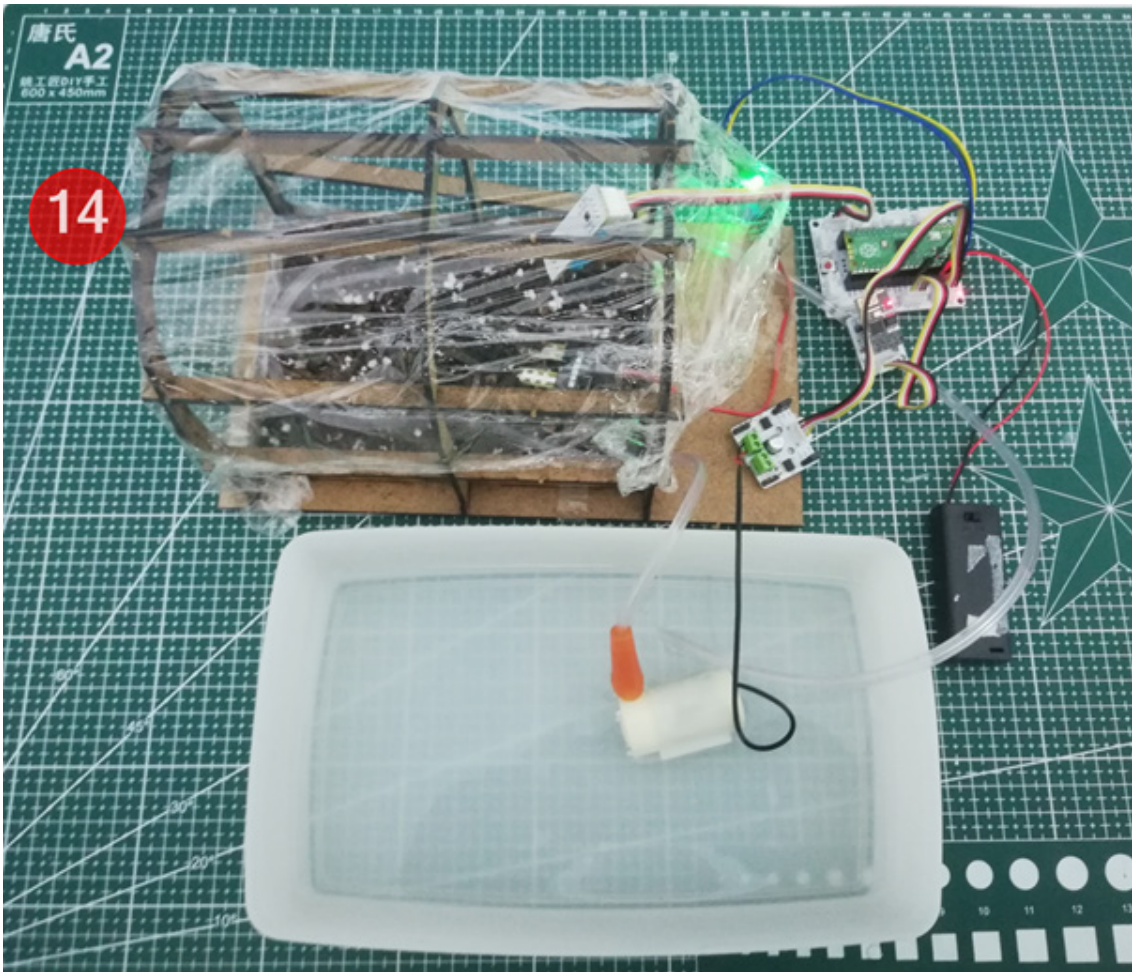
12- Dalgıç pompanın kırmızı ve siyah kablolarını motor sürücüye görseldeki gibi tak.



13- ESP8266 wifi modülünü ve motor sürücüyü seranın uygun bir bölümüne sabitleyip Picobıard'a olan bağlantısını yap.



14- Toprak Nemi sensörünü Picoboard üzerinde GPIO27 nolu pine tak 2 li kalem pili Picoboard'ın güç girişine tak. Dalgıç pompayı ve hortumun ucunu su dolu derin bir kaba yerleştir. Motor sürücünün ıslanmamasına dikkat edin.



2.25.5. Proje Önerisi

Akıllı sera projesinde sera girişine OLED ekran ekleyerek içerideki nem ve sıcaklık değerlerini takip edebilir, MQ2 gaz sensörü, karbondioksit sensörü, hava kalitesi sensörü gibi sensörler ekleyerek seranın içerisindeki hava durumunu wifi üzerinden cep telefonu ile takip edebilirsin. Ayrıca sera içerisine DC fan ve röle ekleyerek wifi üzerinden cep telefonu ile içerideki hava kalitesi durumuna göre havalandırmayı açıp kapatabilirsin.

2.25.6. Projenin MicroPython Kodları

```
import utime
import uos
import machine
from machine import Pin, ADC
from dht import DHT11
from utime import sleep

dht_sensor = DHT11(11)
smo_sensor=ADC(27)
m1 = Pin(21, Pin.OUT)
m1.low()

print("Machine: \t" + uos.uname()[4])
print("MicroPython: \t" + uos.uname()[3])

uart0 = machine.UART(0, baudrate=115200)
print(uart0)

def Connect_WiFi(cmd, uart=uart0, timeout=5000):
    print("CMD: " + cmd)
    uart.write(cmd)
    utime.sleep(7.0)
    Wait_ESP_Rsp(uart, timeout)
    print()

def Rx_ESP_Data():
    recv=bytes()
    while uart0.any()>0:
        recv+=uart0.read(1)
    res=recv.decode('utf-8')
    return res

def Send_AT_Cmd(cmd, uart=uart0, timeout=2000):
    print("CMD: " + cmd)
    uart.write(cmd)
    Wait_ESP_Rsp(uart, timeout)
    print()

def Wait_ESP_Rsp(uart=uart0, timeout=2000):
```

```
    prvMills = utime.ticks_ms()
    resp = b""
    while (utime.ticks_ms()-prvMills)<timeout:
        if uart.any():
            resp = b"".join([resp, uart.read(1)])
    print("resp:")
    try:
        print(resp.decode())
    except UnicodeError:
        print(resp)

Send_AT_Cmd('AT\r\n')         #Test AT startup
Send_AT_Cmd('AT+GMR\r\n')     #Check version information
Send_AT_Cmd('AT+CIPSERVER=0\r\n')
Send_AT_Cmd('AT+RST\r\n')     #Check version information
Send_AT_Cmd('AT+RESTORE\r\n')  #Restore Factory Default Settings
Send_AT_Cmd('AT+CWMODE?\r\n')  #Query the WiFi mode
Send_AT_Cmd('AT+CWMODE=1\r\n') #Set the WiFi mode = Station mode
Send_AT_Cmd('AT+CWMODE?\r\n')  #Query the WiFi mode again
Send_AT_Cmd('AT+CWJAP="YourWifiSSID","YourWifiPassword"\r\n', timeout=5000)
#Connect to AP
utime.sleep(3.0)
Send_AT_Cmd('AT+CIFSR\r\n')    #Obtain the Local IP Address
utime.sleep(3.0)
Send_AT_Cmd('AT+CIPMUX=1\r\n')
utime.sleep(1.0)
Send_AT_Cmd('AT+CIPSERVER=1,80\r\n')    #Obtain the Local IP Address
utime.sleep(1.0)

while True:
    res =""
    res=Rx_ESP_Data()
    utime.sleep(2.0)
    if '+IPD' in res: # if the buffer contains IPD(a connection), then respond with HTML
handshake
        id_index = res.find('+IPD')

        if '/WATERING' in res:
            print('Irrigation Start')
            m1.high()
            utime.sleep(10)
```

```
        m1.low()
        print('Irrigation Finished')
        connection_id =  res[id_index+5]
        print("connectionId:" + connection_id)
        print ('! Incoming connection - sending webpage')
        uart0.write('AT+CIPSEND='+connection_id+',200'+'\r\n')
        utime.sleep(1.0)
        uart0.write('HTTP/1.1 200 OK'+'\r\n')
        uart0.write('Content-Type: text/html'+'\r\n')
        uart0.write('Connection: close'+'\r\n')
        uart0.write(''+'\r\n')
        uart0.write('<!DOCTYPE HTML>'+'\r\n')
        uart0.write('<html>'+'\r\n')
        uart0.write('<body><center><H1>CONNECTED...<br/></H1></center>'+'\r\n')
        uart0.write('<body><center><H1>Irrigation Complete.<br/></H1></center>'+'\
r\n')
        uart0.write('</body></html>'+'\r\n')
    elif '/SERA' in res:
        #sleep(1) # It was used for DHT11 to measure.
        dht_sensor.measure() # Use the sleep() command before this line.
        temp=dht_sensor.temperature()
        hum=dht_sensor.humidity()
        smo=round((smo_sensor.read_u16()/65535)*100)
        sendStr="\"TEMP\":{}, \"Humidity\":{}, \"S.Moisture\":{}%".format(temp,hum,smo)
        sendText="{"+sendStr+"}"
        strLen=46+len(sendText)
        connection_id =  res[id_index+5]
        print("connectionId:" + connection_id)
        print ('! Incoming connection - sending webpage')
        atCmd="AT+CIPSEND="+connection_id+","+str(strLen)
        uart0.write(atCmd+'\r\n')
        utime.sleep(1.0)
        uart0.write('HTTP/1.1 200 OK'+'\r\n')
        uart0.write('Content-Type: text/html'+'\r\n')
        uart0.write(''+'\r\n')
        uart0.write(sendText+'\r\n')

    elif '/' in res:

        print("resp:")
```

```
            print(res)
            connection_id =  res[id_index+5]
            print("connectionId:" + connection_id)
            print ('! Incoming connection - sending webpage')
            uart0.write('AT+CIPSEND='+connection_id+',200'+'\r\n')
            utime.sleep(3.0)
            uart0.write('HTTP/1.1 200 OK'+'\r\n')
            uart0.write('Content-Type: text/html'+'\r\n')
            uart0.write('Connection: close'+'\r\n')
            uart0.write(''+'\r\n')
            uart0.write('<!DOCTYPE HTML>'+'\r\n')
            uart0.write('<html>'+'\r\n')
            uart0.write('<body><center><H1>CONNECTED.<br/></H1></center>'+'\r\n')
            uart0.write('<center><h4>INFO:Get Sensor Data</br>WATERING:Run Water
Pump</h4></center>'+'\r\n')
            uart0.write('</body></html>'+'\r\n')
        utime.sleep(4.0)
        Send_AT_Cmd('AT+CIPCLOSE='+ connection_id+'\r\n') # once file sent, close
connection
        utime.sleep(3.0)
        recv_buf="" #reset buffer
        print ('Waiting For connection...')
```

2.25.7. Projenin Arduino C Kodları

3. Kaynaklar

http://microblocks.fun/

3.1. 3D Modeller

3.1.1 İki Eksen Robot Kol Projesi 3D Parçaları

https://www.thingiverse.com/thing:2302957/files

3.1.2. Obur Kumbara Projesi 3D Parçaları

Orijinal proje sayfası: https://www.thingiverse.com/thing:2824451

Güncellenmiş 3D çizimler: https://drive.google.com/file/d/1_gjo-hEDvhdewtPoorDOv1YhhsNd74ZJ/view?usp=sharing

3.1.3. Otomatik Çöp Kovası Projesi 3D Parçaları

https://drive.google.com/file/d/1hUMhnLaWBmhva2CYZOe3uCm8TFLWkbGQ/view?usp=sharing

3.2. MicroPython Kütüphaneleri

3.2.1 DHT11 Sıcaklık ve Nem Sensörü

https://drive.google.com/file/d/1A2QsrCT2y-BRLQrUz2M6h0M8dDXCF5Yq/view?usp=sharing

3.2.2 0.96” 128x64 I2C Oled Ekran

https://drive.google.com/file/d/19lgjQicHvsazIy_8SzoHbJdaDYw7rVM9/view?usp=sharing

3.2.3 MFRC522 Rfid Okuyucu

https://drive.google.com/file/d/1Edj3uFLOGEOYwb6_e-Szf33aSaSpVLOf/view?usp=sharing

3.2.4 WS2812B Adreslenebilir RGB Şerit Led

https://drive.google.com/file/d/1ldLYPqqj13wpbGTzrl1qt8n6A739PmUo/view?usp=sharing

3.3. Arduino C Kütüphaneleri

3.3.1. OLED Ekran Kütüphanesi

https://drive.google.com/file/d/1eav_HrVoyyRuFrL0VFzdzD2MGPNK7Mbf/view?usp=sharing

3.3.2. WS2812B Adreslenebilir RGB Şerit Led Kütüphanesi

https://drive.google.com/file/d/1JNGSt_zO2o4bnBeUJ0__uBBD757abQ5x/view?usp=sharing

3.3.3. DHT11 Kütüphanesi

https://drive.google.com/file/d/1Vyjrmg77zFqhEfq4suMuRX1SgDjI-2D-/view?usp=sharing

3.3.4. HC-SR04 Ultrasonic Mesafe Sensörü Kütüphanesi

https://drive.google.com/file/d/1mScjZvIHOfIrOs8ug9x0TEQKi42mcWz8/view?usp=sharing

3.4. Android Uygulamalar

3.4.1. Sera Control Android Uygulaması(.apk)

https://drive.google.com/file/d/1XMQ5Egw2e8kHAPiA67geUeKF1yPysW3o/view?usp=sharing

3.4.2. Sera Control MITAppInventor 2 Proje Dosyası

https://drive.google.com/file/d/1Ob3Q3HA34TFcHCklkYoWUyuKgt0dbViC/view?usp=sharing

3.4.3. Ses Kontrollü Robot Araba Projesi Android Uygulaması (.apk)

https://drive.google.com/file/d/11Qg6AwgKb9OH6EtXWw5mDEEhwdHA9MSp/view?usp=drivesdk