**Arduino ile Micro SD Kart Modülü Kullanımı**

Merhaba arkadaşlar, bugünkü yazımızda arduino ile micro sd kart modülü kullanımı konusunu inceleyeceğiz.

**+**

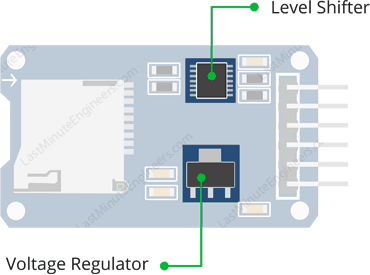
Table of contents

Arduino projeniz için büyük miktarda veri kaydetmek, örneğin GPS verilerini depolamanın bir yoluna ihtiyacınız olabilmektedir.

Çözüm, her dijital kamera ve telefonda kullandığımız SD veya micro SD kartlardır. Günümüz teknolojisi ile üretilen micro sd kartlara GB’larca veri kaydedilebilmektedir. Projenizin büyüklüğüne göre istediğiniz ölçüde micro sd kart satın alarak bu konuya çözüm bulabilirsiniz. Ama asıl çözülmesi gereken nokta, micro sd kartınızı projenize nasıl dahil edeceksiniz? Sorunuzun cevabı [Arduino Micro SD Kart Modülü](https://robocombo.com/Mikro-SD-Kart-Modulu,PR-1191.html)…

**Micro SD Kart Modülüne Genel Bakış**

Micro SD kart modülü, arduino projenize veri kaydı eklemeyi kolaylaştıran iki ana bileşen içerir:

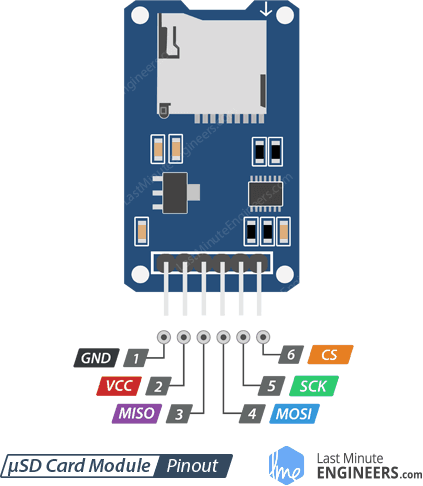
**[](https://hayaletveyap.com/wp-content/uploads/2019/12/Arduino-ile-Micro-SD-Kart-Modulu-Kullanimi-1.png)**

|  |
| --- |
| **Önemli:** Herhangi bir standart micro sd kartın çalışma voltajı 3,3 V’tur. Bu yüzden doğrudan 5V kullanan devrelere bağlayamayız. Aslında, 3.6V’u aşan voltajlar micro sd karta kalıcı hasar verecektir. Modül, üzerinde voltaj regülatörü yongası mevcuttur. |

Micro sd kartlarla bağlantı kurmanın iki yolu vardır; SPI modu ve SDIO modu. SDIO modu çok daha hızlıdır ve cep telefonlarında, dijital kameralarda vb. kullanılır. Ancak daha karmaşıktır ve koruma kilidi mevcuttur. Bu nedenle, bizim gibi arduino projeleri üzerinde çalışan makerlar tarafından SDIO modu tercih edilmemektedir. Bunun yerine, her SD kart modülü ile uyumlu ve basit, herhangi bir mikrodenetleyici ile kullanması kolay olan SPI modu tercih edilmektedir.

**Micro SD Kart Modülü Pinout**

Micro sd kart modülünün bağlanması oldukça kolaydır. 6 pini vardır:

**[](https://hayaletveyap.com/wp-content/uploads/2019/12/Arduino-ile-Micro-SD-Kart-Modulu-Kullanimi-2.png)**

VCC pini modül için güç sağlar ve Arduino’daki 5V pine bağlanmalıdır.

GND Arduino GND pinine bağlanmalıdır.

MISO (Master In Slave Out) , Micro SD Kart Modülünden SPI çıkışıdır.

MOSI (Master Out Slave In) , Micro SD Kart Modülüne SPI girişidir.

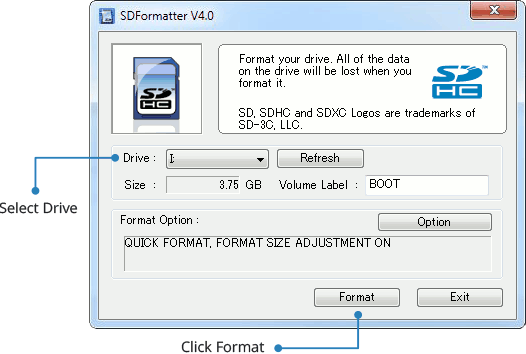
SCLK (Seri Saat) pini, Arduino tarafından üretilen, seri haberleşmeyi sağlayan, veri iletimini senkronize eden saat darbelerini kabul eder.

SS (Slave Select) pini, Arduino (Master) tarafından SPI veri yolundaki belirli cihazları etkinleştirmek ve devre dışı bırakmak için kullanılır.

**Mikro SD kartın hazırlanması**

Mikro SD kartı modüle takmadan ve Arduino’ya bağlamadan önce kartı düzgün bir şekilde biçimlendirmelisiniz. Micro sd kart, arduino kütüphanesi için gerekli olan ve neredeyse tüm diğer SD kütüphaneleri için kart FAT16 veya FAT32 formatında biçimlendirilmelidir.

Kartı biçimlendirmek için [**SDA**](https://en.m.wikipedi0.org/wiki/SD_Association) tarafından kodlanan resmi **SD kart biçimlendirici programı** kullanmanızı önemle tavsiye ederiz. [**Formatlayıcıyı**](https://www.sdcard.org/downloads/formatter/index.html) indirin ve bilgisayarınızda çalıştırın, doğru sürücüyü seçin ve FORMAT’a tıklayın.

**[](https://hayaletveyap.com/wp-content/uploads/2019/12/Arduino-ile-Micro-SD-Kart-Modulu-Kullanimi-3.png)**

**Kablolama – Micro SD Kart Modülünü Arduino ya Bağlama**

Kartınız kullanıma hazır olduğuna göre başlayabiliriz!

Başlamak için mikro SD kart modülünü breadboard’a takın. Modüldeki VCC pinini Arduino’daki 5V pinine, GND pinini GND ‘ye bağlayın.

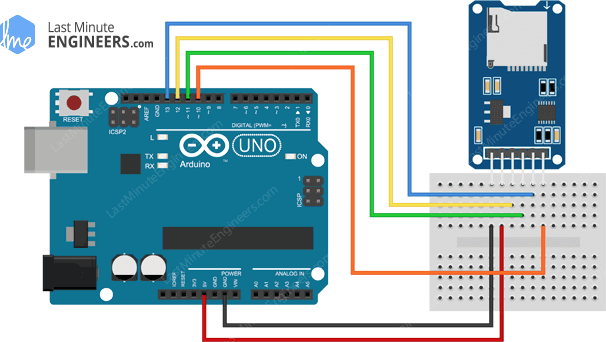
Micro SD kartlar çok fazla veri aktarımı gerçekleştirdiğinden dolayı, SPI pinlerine bağlayarak en iyi performansı alabilirsiniz.

Her Arduino Kartında buna göre bağlanması gereken farklı SPI pinleri olduğunu unutmayın. UNO / Nano gibi Arduino kartları için bu pinler dijital 13 SCLK, 12 MISO ve 11 MOSI‘dir. Ayrıca ‘chip / slave select’ SS hattı için dördüncü bir pine ihtiyacınız olacaktır. Genellikle bu pin 10’dur, ancak istediğiniz herhangi bir pini kullanabilirsiniz.

Mega’nız varsa, pinler farklıdır! Dijital 50 MISO, 51 MOSI, 52 SCLK ve 53 SS kullanmalısınız. Hızlı anlamak için aşağıdaki tabloya bakın.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | MOSI | MİSO | SCK | CS |
| Arduino Uno | 11 | 12 | 13 | 10 |
| Arduino Nano | 11 | 12 | 13 | 10 |
| Arduino Mega | 51 | 50 | 52 | 53 |

Yukarıda belirtilenden farklı bir Arduino kartı kullanıyorsanız, devam etmeden önce [**Arduino SPI**](https://www.arduino.cc/en/Reference/SPI) bağlantısını inceleyebilirsiniz.

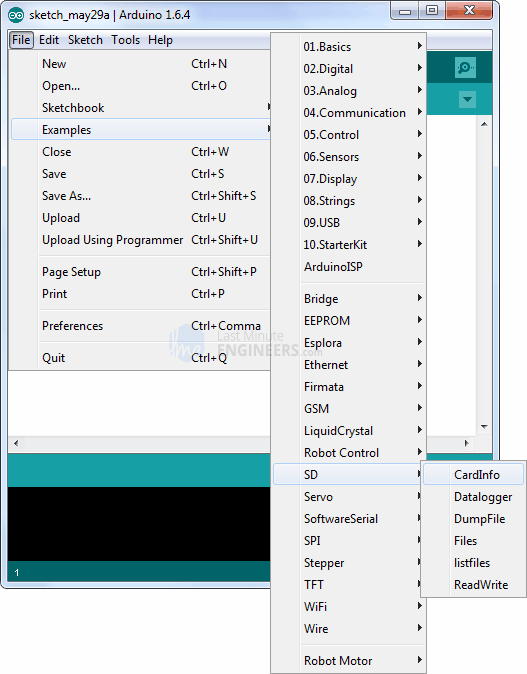
**[](https://hayaletveyap.com/wp-content/uploads/2019/12/Arduino-ile-Micro-SD-Kart-Modulu-Kullanimi-4.png)**

Micro sd kart modülünü arduino uno’ya kablolama işlemini bitirdiysek veri yüklemeye geçebiliriz.

**Arduino Kodu – SD kart modülünün CardInfo ile test edilmesi**

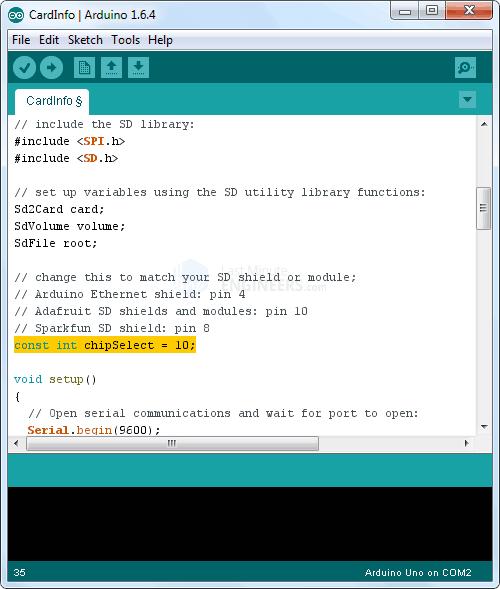
Bir sd kart ile iletişim kurmak biraz uğraştırabilir, ancak neyse ki SD kartlardan okumayı ve SD kartlara yazmayı basitleştiren [**SD**](https://www.arduino.cc/en/Reference/SD) adında çok güzel bir kütüphane mevcut. Bunu Örnekler alt menüsünde görebilirsiniz.

CardInfo örnek çizimini seçin.

**[](https://hayaletveyap.com/wp-content/uploads/2019/12/Arduino-ile-Micro-SD-Kart-Modulu-Kullanimi-5.png)**

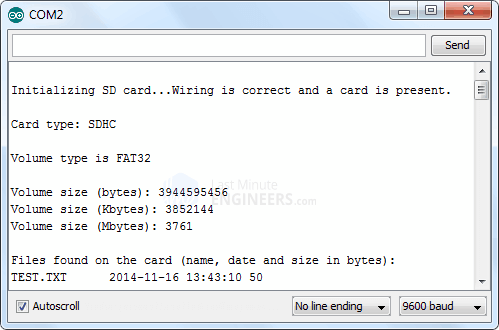
Bu görsel, bir SD kartın desteklenip desteklenmediğini anlamaya çalışırken çok yararlı olabilir. Herhangi bir yeni kartı denemeden önce, görseldeki şekilde sistemi bir kez çalıştırmanızı öneririz!

Kod içerisindeki **chipSelect** hattının doğru şekilde başlatıldığından emin olun, biz # 10 numaralı dijital pini kullanıyoruz, siz kendi pininize göre kodu düzenleyin.

**[](https://hayaletveyap.com/wp-content/uploads/2019/12/Arduino-ile-Micro-SD-Kart-Modulu-Kullanimi-6.png)**

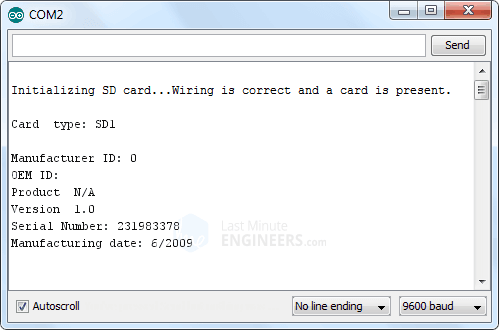
Şimdi SD kartı modüle takın ve kodu yükleyin.

Kodu çalıştırdığınızda muhtemelen aşağıdaki gibi bir şey elde edersiniz:

**[](https://hayaletveyap.com/wp-content/uploads/2019/12/Arduino-ile-Micro-SD-Kart-Modulu-Kullanimi-7.png)**

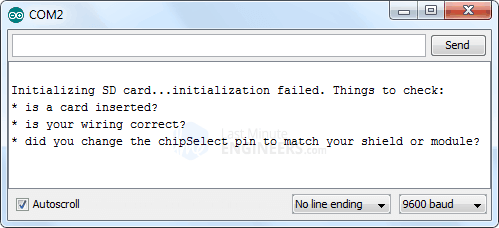
Buradaki bilgiler de; kart türünün SDHC (SD Yüksek Kapasite) olduğunu, format türünün FAT32 olduğunu ve kartın boyutunu yaklaşık 4 GB vb. göreceksiniz.

Merdiven altı üretilmiş kötü bir kartınız varsa, şunları görebilirsiniz:

**[](https://hayaletveyap.com/wp-content/uploads/2019/12/Arduino-ile-Micro-SD-Kart-Modulu-Kullanimi-8.png)**

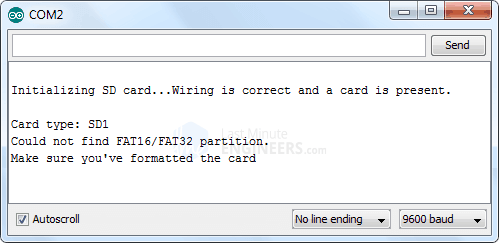
Kart çoğunlukla veri iletimi sağladı, ancak verilerin hepsi kötü. Üretici Kimliği / OEM Kimliği olmadığını ve Ürün Kimliği’nin ‘Yok’ olduğunu göreceksiniz. Bu, kartın bazı üretim hataları olduğunu gösterir. Kaliteli bir kart değil. Böyle bir karta sahipseniz, yeniden biçimlendirmeyi deneyebilirsiniz.

SD kartı çıkarmayı ve kodu tekrar çalıştırmayı deneyin, aşağıdaki gibi bir görüntü elde ederseniz;

**[](https://hayaletveyap.com/wp-content/uploads/2019/12/Arduino-ile-Micro-SD-Kart-Modulu-Kullanimi-9.png)**

Bakın, SD kartı bile başlatamadı. Bu, bir kart için üretim hatası varsa veya kart kalıcı olarak hasar görürse ortaya çıkan bir durumdur.

Kablolama ve üretim doğruysa ancak SD kart düzgün biçimlendirilmemişse, şöyle bir şey alırsınız:

**[](https://hayaletveyap.com/wp-content/uploads/2019/12/Arduino-ile-Micro-SD-Kart-Modulu-Kullanimi-10.png)**

**Arduino Kodu – Veri Okuma ve Yazma**

SD kartı başarıyla çalıştırdığınızı düşünürsek, bir sonraki denemeye geçelim. Aşağıdaki kod, bir dosyadan veri yazma ve okuma ile ilgili temel bir açıklama içerir. Ayrıntılı dökümüne başlamadan önce taslağı deneyin.

#include <**SPI**.h>

#include <**SD**.h>

File myFile;

// change this to match your SD shield or module;

const int chipSelect = 10;

void setup()

{

 // Open serial communications and wait for port to open:

**Serial**.begin(9600);

 while (!**Serial**) {

   ; // wait for serial port to connect. Needed for Leonardo only

 }

**Serial**.print("Initializing SD card...");

 if (!**SD**.begin()) {

**Serial**.println("initialization failed!");

   return;

 }

**Serial**.println("initialization done.");

 // open the file. note that only one file can be open at a time,

 // so you have to close this one before opening another.

 myFile = **SD**.open("test.txt", FILE\_WRITE);

 // if the file opened okay, write to it:

 if (myFile) {

**Serial**.print("Writing to test.txt...");

   myFile.println("testing 1, 2, 3.");

   // close the file:

   myFile.close();

**Serial**.println("done.");

 } else {

   // if the file didn't open, print an error:

**Serial**.println("error opening test.txt");

 }

 // re-open the file for reading:

 myFile = **SD**.open("test.txt");

 if (myFile) {

**Serial**.println("test.txt:");

   // read from the file until there's nothing else in it:

   while (myFile.available()) {

**Serial**.write(myFile.read());

   }

   // close the file:

   myFile.close();

 } else {

   // if the file didn't open, print an error:

**Serial**.println("error opening test.txt");

 }

}

void loop()

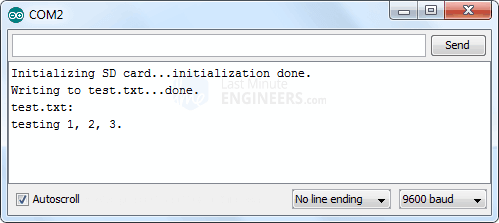
{

 // nothing happens after setup

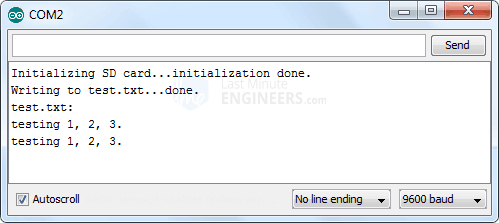
}Kopyala

 Kopyala

Kod doğru yüklenirse, her şey yolundaysa, ekranda aşağıdakiler görünür.

**[](https://hayaletveyap.com/wp-content/uploads/2019/12/Arduino-ile-Micro-SD-Kart-Modulu-Kullanimi-11.png)**

Arduino’nuzu sıfırlar ve kodun tekrar çalışmasına izin verirseniz; yazılan yeni veriler, önceki verilerin üzerine yazılmadan dosyaya eklenir.

**[](https://hayaletveyap.com/wp-content/uploads/2019/12/Arduino-ile-Micro-SD-Kart-Modulu-Kullanimi-12.png)**

**Kod Açıklaması:**

Kod, dahili **SD kütüphanesi** ve SPI arayüzü üzerinden SD kart ile kolayca iletişim kurmamızı sağlayan **SPI kütüphanesini** dahil etmekle başlar.

#include <**SPI**.h>

#include <**SD**.h>

Kopyala

Kütüphaneler dahil edildikten sonra, yaptığımız bir sonraki şey, SD kart modülünün chipSelect (CS) Arduino pinini bildirmektir. CS pini, Arduino dijital pinlerinden biri olarak sabitlenmeyen tek pindir. SPI arabirimini kullandığımız ve bu pinler zaten SPI kütüphanesinde bildirildiğinden diğer SPI pinlerini bildirmemize gerek yok. Pini bildirdikten sonra, SD kartta veri depolamak için kullanılacak bir **myFile** nesnesi oluşturalım.

const int chipSelect = 10;

File myFile;

Kopyala

Ardından, setup () bölümünde: Sonuçları ekranda göstermek için seri iletişimi başlatırız. Şimdi, SD.begin() fonksiyonunu kullanarak SD kartı başlatacağız ve başlatma başarılı olursa “ **if** ” ifadesi gerçekleşir ve “**String**” başlatma yapılır. Başarısız ise “ ***başlatma başarısız!*** ”Yazdırılır ve program sona erer.

**Serial**.begin(9600);

**Serial**.print("Initializing SD card...");

 if (!**SD**.begin()) {

**Serial**.println("initialization failed!");

   return;

 }

**Serial**.println("initialization done.");

Kopyala

Ardından, SD.open() işlevi “ ***test.txt*** ” adlı dosyayı açacaktır. Diğer FILE\_WRITE parametresi dosyayı okuma-yazma modunda açar.

myFile = **SD**.open("test.txt", FILE\_WRITE);

Kopyala

Dosya açıldıktan sonra, ekranda “***test.txt yazdırılıyor”*** görmeliyiz. Ardından myFile.println() işlevini kullanarak “test 1, 2, 3” metnini yazacağız. Bundan sonra, dosyaya yazılan verilerin kaydedilmesini sağlamak için close() işlevini kullanmamız gerekir.

 if (myFile) {

**Serial**.print("Writing to test.txt...");

   myFile.println("testing 1, 2, 3.");

   myFile.close();

**Serial**.println("done.");

 } else {

**Serial**.println("error opening test.txt");

 }

Kopyala

Şimdi, yazma işleminin başarılı olup olmadığını kontrol etmek için SD.open() işlevini kullanacağız, ancak bu sefer “ ***test.txt*** ” dosyası zaten oluşturulmuş olduğu için, işlev sadece dosyayı açar. Sonra myFile.read() işlevini kullanarak ekran da göreceğiz. read() işlevi aslında bir seferde yalnızca bir karakter okur, bu nedenle tüm karakterleri okumak için “while” döngüsünü ve myFile.available() işlevini kullanmamız gerekir. Sonunda da dosyayı kapatmamız gerekiyor.

myFile = **SD**.open("test.txt");

 if (myFile) {

**Serial**.println("test.txt:");

   while (myFile.available()) {

**Serial**.write(myFile.read());

   }

   myFile.close();

 } else {

**Serial**.println("error opening test.txt");

 }

Kopyala

Bu, dosyaların nasıl okunacağını ve yazılacağını göstermek için sadece bir demo taslağı olduğundan, kodu birden çok kez çalıştırmanın bir anlamı yoktur. Böylece tüm kod bir loop() koymak yerine sadece bir kez çalışan setup() işlevine yerleştirilmiştir.

void loop()

{

}

Kopyala

**Dikkat edilmesi gereken bazı şeyler**

* Dizeler, değişkenler vb. yazmak için seri nesneler gibi print () ve println () işlevlerini kullanabilirsiniz.
* Read() yalnızca bir karakter döndürür. Tam satır veya sayı okumaz!
* close() Tüm verilerin kalıcı olarak yazıldığından emin olmak için işiniz bittiğinde dosya (lar)! Bu, kullanılan RAM miktarını azaltır.
* Dosyaları bir dizinde açabilirsiniz. Örneğin, dizinde bir dosya açmak istiyorsanız, SD.open(“/myfiles/example.txt”) kullanabilirsiniz. Kodu kendi dosyalarınıza göre düzenleyiniz.
* SD kart kütüphanesi ‘uzun dosya adlarını’ desteklemiyor. Bunun yerine, [**dosya adları için 3 biçimini**](https://en.m.wikipedi0.org/wiki/8.3_filename) kullanın ve dosya adlarını kısa tutun! Örneğin datalog.txt ismi idealdir ancak “Sensörümün günlük dosyası.txt” gibi isimler iyi değildir!
* Ayrıca dosya adlarının ‘büyük / küçük harf’ duyarlılığına sahip olmadığını unutmayın, bu nedenle datalog.txt dosyası DataLog.Txt dosyası ile aynıdır ve DATALOG.TXT dosyası ile aynıdır

**SD Kütüphanesi’ndeki diğer faydalı fonksiyonlar**

**SD nesnesiyle kullanılan işlevler**

**SD** nesnesiyle kullanabileceğiniz birkaç kullanışlı işlev vardır. Bunlardan birkaçı aşağıda listelenmiştir:

* Sadece bir dosyanın var olup olmadığını kontrol etmek istiyorsanız, exists(“filename.txt”) kodunu kullanın.
* remove(“unwanted.txt”) kodu ile bir dosyayı silebilirsiniz dikkatli olun! Bu gerçekten tamamen siler ve silinen verileri geri getirecek bir ‘**Geri Dönüşüm Kutusu**‘ yoktur.
* mkdir(“/mynewdir”) kullanarak bir alt dizin oluşturabilirsiniz. Zaten bu kodu daha önce kullandıysanız, SD.exists() kodunu öncelikli olarak kullanabilirsiniz.

**“File” nesnesiyle kullanılan işlevler**

**File** nesneleriyle kullanabileceğiniz birkaç işlev vardır:

* Bir dosyada seek() kodu, okuma / yazma imlecini yeni bir konuma taşıyacaktır. Örneğin seek(0) çok kullanışlı olabilir, imleci dosyanın başlangıcına götürecektir.
* Aynı şekilde, dosyada nerede olduğunuzu söyleyen position() kodunu çağırabilirsiniz.
* Bir dosyanın boyutunu bilmek istiyorsanız, size() kodunu kullanabilirsiniz.
* Dizinler / klasörler özel dosyalardır, isDirectory() kodunu çağırarak bir dosyanın dizin de olup olmadığını belirleyebilirsiniz.
* Bir dizininiz olduğunda, openNextFile() kodunu kullanarak dizindeki tüm dosyaları getirebilirsiniz.
* Bir dosyanın adını bilmeniz gerekebilir (örneğin, bir dizinde openNextFile () öğesini çağırdıysanız, 8.3 biçimlendirilmiş karakter dizisine bir işaretçi döndürecek olan name () öğesini doğrudan Serial.print () yapabilirsiniz.