

In einem Webformular können Lottozahlen ermittelt, angezeigt und gespeichert werden. Der Arduino „spielt“ für dich 6 Zahlen. Anschließend werden die Lottozahlen gezogen, aufsteigend sortiert und auf die Anzahl der übereinstimmenden Zahlen untersucht.

So sieht es aus:

← ↻ 🏠 ⚠ Nicht sicher | 192.168.1.32/a

Lottozahlen

Eigene IP: 192.168.1.79
IP des Ethernet-Shields: 192.168.1.32

Lottozahlen speichern
Lottozahlen anzeigen
Datei entfernen

17.08.2022 Uhrzeit: 19:43:36
Gespielte Zahlen 4 11 13 23 33 43

Gezogene Zahlen 1 17 23 27 32 41

1 Treffer: 23

17.08.2022 Uhrzeit: 19:43:37
Gespielte Zahlen 8 9 20 24 34 45

Gezogene Zahlen 8 27 35 36 37 40

1 Treffer: 8

17.08.2022 Uhrzeit: 19:43:45
Gespielte Zahlen 4 5 13 18 28 30

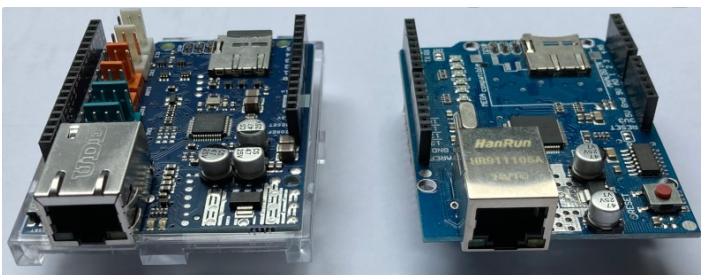
Gezogene Zahlen 3 5 15 16 22 32

1 Treffer: 5

17.08.2022 Uhrzeit: 19:43:47
Gespielte Zahlen 3 17 20 26 27 30

Gezogene Zahlen 5 15 20 21 40 47

1 Treffer: 20

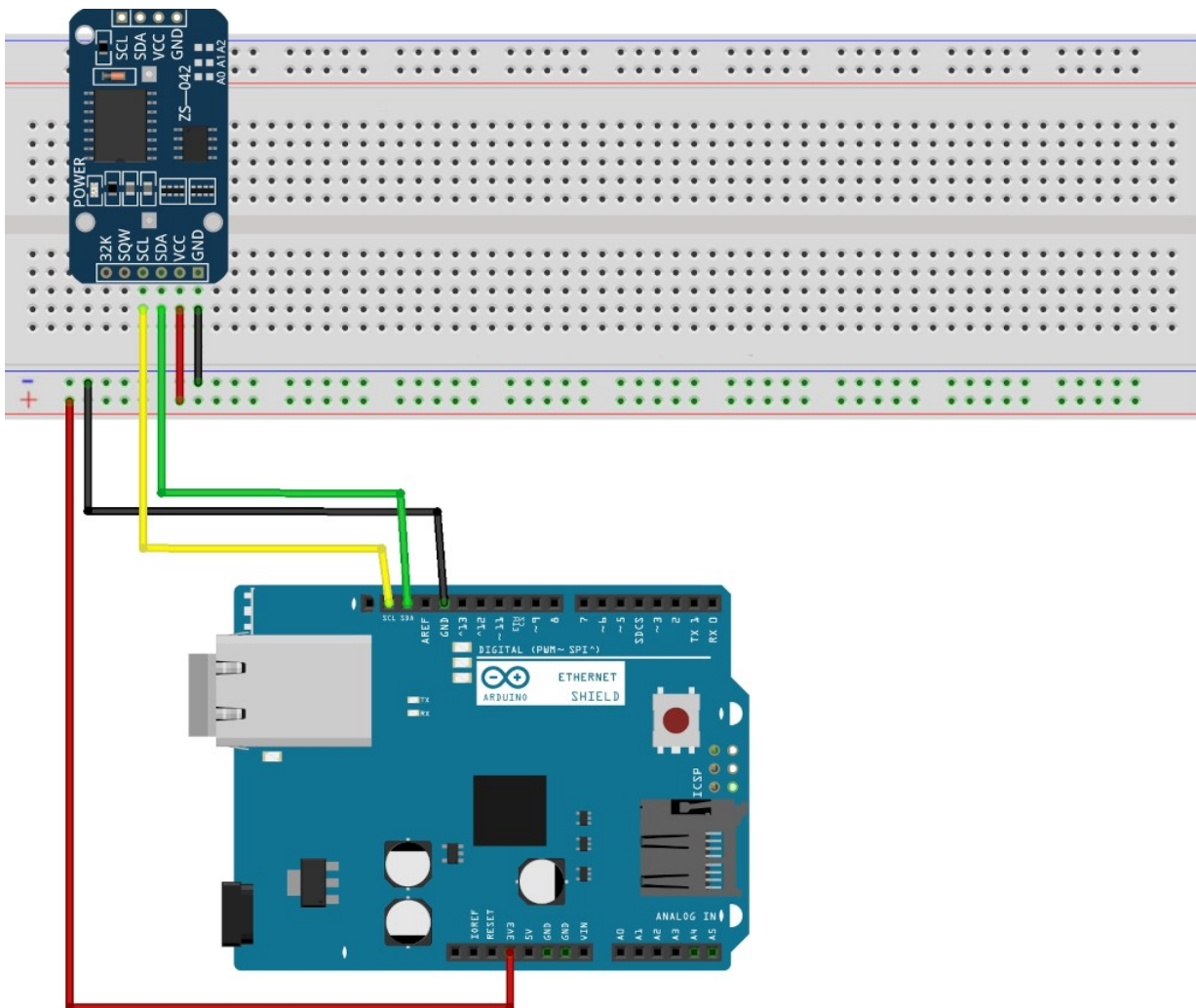


Für diese Anleitung benötigst du ein sogenanntes „Shield“, eine Platine, die einfach auf den Arduino aufgesteckt wird. Auf ihr befindet sich ein LAN-Anschluss (RJ45). Alle digitalen und analogen Anschlüsse stehen auch weiterhin zur Verfügung.

Benötigte Bauteile:

- ➡ RTC-Modul
- ➡ Ethernet-Shield
- ➡ Leitungsdrähte

Baue die Schaltung auf.



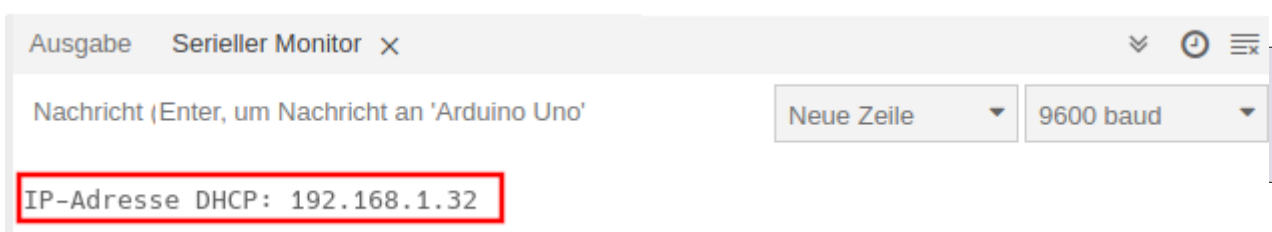
fritzing

Im Regelfall befindet sich in einem lokalen Netzwerk ein DHCP-Server, der jedem Gerät im Netzwerk automatisch eine IP-Adresse zuteilt. Wenn du dich für diese Alternative entscheidest, musst die im Seriellen Monitor angezeigte Adresse für die Schaltungen verwenden.

Die MAC-Adresse ist die Hardware-Adresse jeder einzelnen Netzwerkschnittstelle (LAN oder WLAN), mit der jedes Gerät im Netzwerk eindeutig identifiziert werden kann. Sie besteht aus sechs Bytes in hexadezimaler Schreibweise, die durch „:“ oder „-“ getrennt werden.

Du kannst die im Programm verwendete „erfundene“ MAC-Adresse übernehmen: Die Gefahr, dass sich ein Gerät mit der gleichen MAC-Adresse im Netzwerk befindet, ist äußerst gering.

Im Seriellen Monitor siehst du die IP-Adresse des Ethernet-Shields:



Um Speicherplatz zu sparen, kannst du auch eine statische Adresse verwenden.

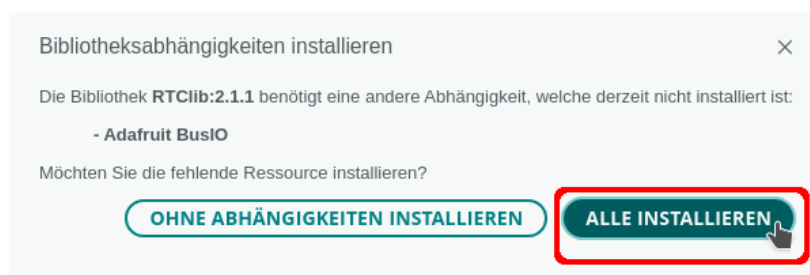
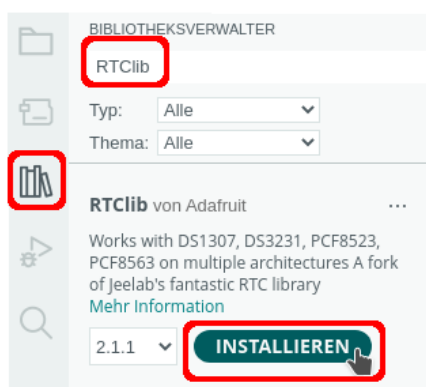
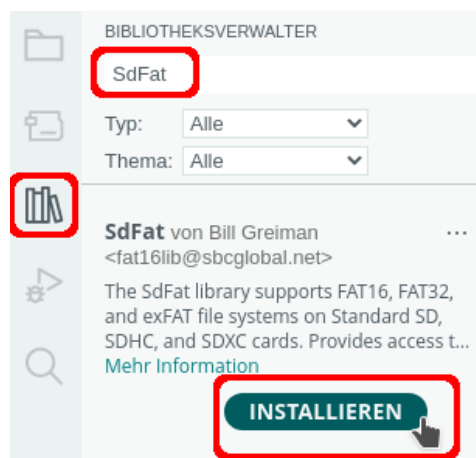
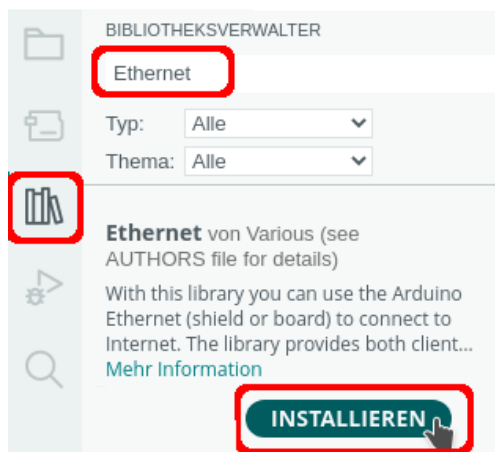
Bei der Definition der Variablen muss eine Zeile hinzugefügt werden:

```
// feste IP
IPAddress ip(192, 168, 1, 200);
```

Im setup-Teil muss eine Zeile geändert werden:

```
// Ethernet starten DHCP
// Ethernet.begin(mac);
// Ethernet starten feste IP
Ethernet.begin(mac, ip);
```

Benötigte Bibliotheken:



Binde die benötigten Bibliotheken ein und definiere die Variablen:

```
# include <SdFat.h>
# include <RTCLib.h>
# include <Ethernet.h>

// Bezeichnung der SD-Karte
SdFat SD;

// Bezeichnung des RTC-Moduls
RTC_DS3231 rtc;

// Bezeichnung der Textdatei
File ZiehungLotto;

// Datenpin für das SD-Kartenmodul
# define DatenPin 4

// Array für die gezogenen Zahlen
int LottoZahl[6];

// Array für die angekreuzten Zahlen
int SpielZahl[6];

// Minimum/Maximum der Zufallszahlen
const byte Minimum = 1;
const byte Maximum = 49;

// Anzahl der zu ziehenden Zahlen
const byte Anzahl = 6;

// MAC-Adresse und IP definieren
byte mac[] = {0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED};

// Name des Servers vergeben
EthernetServer server(80);
```

Der setup-Teil. Beachte die Kommentare.

```
void setup()
{
  /*
   * wenn Datum und Zeit nicht korrekt -> Datum/Zeit setzen
   * Jahr, Monat, Tag, Stunde, Minute, Sekunde
   * rtc.adjust(DateTime(2020, 12, 13, 13, 31, 30));
   */
  // RTC-Modul starten
  rtc.begin();

  // Zufallsgenerator starten
  randomSeed(analogRead(0));
  Serial.begin(9600);

  // auf serielle Verbindung warten
  while (!Serial) {};
```

```
// Ethernet starten DHCP
Ethernet.begin(mac);

// Server starten
server.begin();

// SD-Karte starten
SD. begin(DatenPin);

// zur Kontrolle IP-Adresse anzeigen
// localIP -> Adresse, die im Browser eingegeben wird
Serial.print(F("IP des Ethernet-Shields: "));
Serial.println(Ethernet.localIP());
}
```



Der loop-Teil. Beachte die Kommentare.

```
void loop()
{
  String AnzahlTreffer = "";
  String GespielteZahlen = "";
  String GezogeneZahlen = "";
  byte Treffer = 0;
  String Auswertung = "";

  // neue Anfrage
  EthernetClient Client = server.available();
  if (Client)
  {
    // solange der Klient verbunden ist ...
    while (Client.connected())
    {
      if (Client.available())
      {
        // auf Eingabe warten ...
        char Zeichen = Client.read();

        // ... und die Antwort zusammenbauen
        Auswertung += Zeichen;

        // \n = Seite vom Klienten vollständig geladen
        if (Zeichen == '\n')
        {
          // HTTP-Anforderung senden
          Client.println("HTTP/1.1 200 OK");
          Client.println("Content-type:text/html");
          // Leerzeile zwingend erforderlich
          Client.println();
          Client.println(F("<!doctype html>"));
          Client.println(F("<html>"));
          Client.println(F("<body>"));
          Client.println(F("<h1>Lottozahlen</h1><hr />"));
        }
      }
    }
  }
}
```

```

/*
  IP-Adressen anzeigen
  Client.remoteIP() -> eigene IP
  Ethernet.localIP() -> IP des Ethernet-Shields
*/
Client.print(F("<b>Eigene IP: "));
Client.print(Client.remoteIP());
Client.print(F("</b>"));
Client.print(F("<br><b>IP des Ethernet-Shields: "));
Client.print(Ethernet.localIP());
Client.print(F("</b>"));
Client.println(F("<hr />"));

// Buttons
Client.print(F("<form>"));

// speichern
Client.print(F("<input type='button'"));
Client.print(F(" style='cursor:pointer;background-color:#CFFB00;'"));
Client.println(F(" onClick=\"location.href='s'\" value='Lottozahlen speichern'>"));

// anzeigen
Client.print(F("<input type='button'"));
Client.print(F(" style='cursor:pointer;background-color:#CAD8EC;'"));
Client.println(F(" onClick=\"location.href='a'\" value='Lottozahlen anzeigen'>"));

// entfernen
Client.print(F("<input type='button' "));
Client.print(F(" style='cursor:pointer;background-color:#F48A93;'"));
Client.println(F(" onClick=\"location.href='e'\" value='Datei entfernen'>"));
Client.println(F("</form><hr />"));

/*
  ein Klick auf einen der Buttons erzeugt eine GET-Anweisung in der Form
  GET /a HTTP/1.1 -> Button anzeigen auswerten
  GET /s HTTP/1.1 -> Button speichern auswerten
  GET /e HTTP/1.1 -> Button entfernen auswerten
  indexOf durchsucht den String nach diesen Zeichen (> 0 = Zeichen gefunden)
*/

// Auswertung der Anzeige muss vor dem Schließen der Seite erfolgen
if (Auswertung.indexOf("a") > 0)
{
  // Datei öffnen und Zahlen anzeigen
  ZiehungLotto = SD.open("Lottozahlen.txt");
  if (ZiehungLotto)
  {
    // solange sich Zeilen in der Datei befinden ...
    while (ZiehungLotto.available())
    {
      // ... werden sie gelesen und ausgegeben
      Client.write(ZiehungLotto.read());
    }
  }
}

```

```
// Datei schließen
ZiehungLotto.close();
}
else Client.println(F("Die Datei ist leer oder nicht vorhanden!"));
}
Client.println(F("</body></html>"));

// Zeit, um die Antwort zu übertragen
delay(1);

// Verbindung beenden
Client.stop();

// entfernen
if (Auswertung.indexOf("e") > 0)
{
    if (SD.exists("Lottozahlen.txt"))
    {
        // Datei entfernen
        SD.remove("Lottozahlen.txt");
    }
}

// speichern
if (Auswertung.indexOf("s") > 0)
{
    // Ziehung der "angekreuzten" Zahlen
    Ziehung();
    ZahlenSortieren(LottoZahl, 6);

    // String GespielteZahlen aus den Elementen des Arrays LottoZahl
    // zusammensetzen
    for (int i = 0; i < Anzahl; i++)
    {
        GespielteZahlen = GespielteZahlen + LottoZahl[i] + " ";
        SpielZahl[i] = LottoZahl[i];
    }

    // Ziehung der Lottozahlen
    Ziehung();

    // Zahlen sortieren
    ZahlenSortieren(LottoZahl, 6);

    // Vergleichen der Elemente der Arrays SpielZahl und LottoZahl
    for (int i = 0; i < Anzahl; i++)
    {
        for (int ii = 0; ii < Anzahl; ii++)
        {
            // Übereinstimmung gefunden -> Treffer um 1 erhöhen
            // AnzahlTreffer: String für übereinstimmende Zahlen erzeugen
```

```

        if (SpielZahl[i] == LottoZahl[ii])
        {
            AnzahlTreffer = AnzahlTreffer + SpielZahl[i] + " ";
            Treffer ++;
        }
    }
}

// String GezogeneZahlen aus den Elementen des Arrays LottoZahl
// zusammensetzen
for (int i = 0; i < Anzahl; i++)
{
    GezogeneZahlen = GezogeneZahlen + LottoZahl[i] + " ";
}

/*
    Datei zum Schreiben (FILE_WRITE) öffnen
    wenn sie noch nicht existiert, wird sie erstellt
*/
ZiehungLotto = SD.open("Lottozahlen.txt", FILE_WRITE);
if (ZiehungLotto)
{
    // aktuelle Zeit holen
    DateTime aktuell = rtc.now();
    char Datum[] = "DD.MM.YYYY";
    ZiehungLotto.print(aktuell.toString(Datum));

    // Zeit schreiben
    char Zeit[] = " Uhrzeit: hh:mm:ss";
    ZiehungLotto.println(aktuell.toString(Zeit));
    ZiehungLotto.println(F("<br>"));

    // Daten in die Datei schreiben
    ZiehungLotto.println(F("Gespielte Zahlen"));
    ZiehungLotto.println(GespielteZahlen);
    ZiehungLotto.println(F("<br>"));
    ZiehungLotto.println(F("-----"));
    ZiehungLotto.println(F("<br>"));
    ZiehungLotto.println(F("Gezogene Zahlen"));
    ZiehungLotto.println(GezogeneZahlen);
    ZiehungLotto.println(F("<br>"));
    ZiehungLotto.println(F("-----"));
    ZiehungLotto.println(F("<br>"));

    // keine Treffer
    if (Treffer == 0) ZiehungLotto.println(F("keine Treffer!"));

    // getroffene Zahlen anzeigen
    else
    {
        ZiehungLotto.print(Treffer);
        ZiehungLotto.print(F(" Treffer: "));
        ZiehungLotto.print(AnzahlTreffer);
    }
    ZiehungLotto.println(F("<br>"));
}

```

```

        ZiehungLotto.println(F("-----"));
        ZiehungLotto.println(F("<br>"));

        // Datei schließen
        ZiehungLotto.close();
    }
}
}
}
}
}
}
}
}

```

Jetzt fehlen noch die Methoden zum Sortieren der Zahlen und für die Ziehung der Lottozahlen.

```

void ZahlenSortieren(int Zahlen[], int Groesse)
{
    // unsortierten Bereich des Arrays durchlaufen
    for (int i = 0; i < (Groesse - 1); i++)
    {
        // bei jedem Durchlauf wird das jeweils letzte Element weggelassen
        for (int ii = 0; ii < (Groesse - (i + 1)); ii++)
        {
            /*
                wenn die aktuelle Zahl größer als die nachfolgende Zahl ist
                -> aktuelle Zahl temporär speichern
                -> Zahlen vertauschen
                -> temporäre Zahl der nachfolgenden Zahl zuweisen
            */
            if (Zahlen[ii] > Zahlen[ii + 1])
            {
                int zwischengespeicherteZahl = Zahlen[ii];
                Zahlen[ii] = Zahlen[ii + 1];
                Zahlen[ii + 1] = zwischengespeicherteZahl;
            }
        }
    }
}

```

```

int Ziehung()
{
    for (int i = 0; i < Anzahl; i++)
    {
        /*
            die Lottozahl gilt solange als bereits gezogen
            bis in der for-Schleife nachgewiesen wird, dass sie neu ist
            und BereitsGezogen den Wert false hat
        */
        bool BereitsGezogen = true;

        while (BereitsGezogen)
        {
            // Zahl ziehen
            LottoZahl[i] = random(Minimum, Maximum);
            BereitsGezogen = false;
        }
    }
}

```

```
/*
    Zufallszahl mit den bereits gezogenen Zahlen vergleichen
    i wird in der ersten for-Schleife hochgezählt
    alle bisherigen Zahlen (ii) werden mit den bisher gezogenen
    (außer der gerade gezogenen) verglichen (ii < i)
    ist sie identisch, erhält BereitsGezogen den Wert true
    es wird erneut eine Zufallszahl bestimmt
    wenn die Zahl neu ist, (BereitsGezogen hat noch den Wert false)
    wird die while-Schleife verlassen und die nächste Zahl gezogen
*/

for (int ii = 0; ii < i; ii ++){
    if (LottoZahl[i] == LottoZahl[ii]) BereitsGezogen = true;
}
}

/*
    das Array mit den Zahlen wird an das Hauptprogramm zurückgegeben
    beim ersten Aufruf sind es die gespielten Zahlen
    beim zweiten Aufruf die gezogenen Lottozahlen
*/
return LottoZahl[Anzahl];
}
```