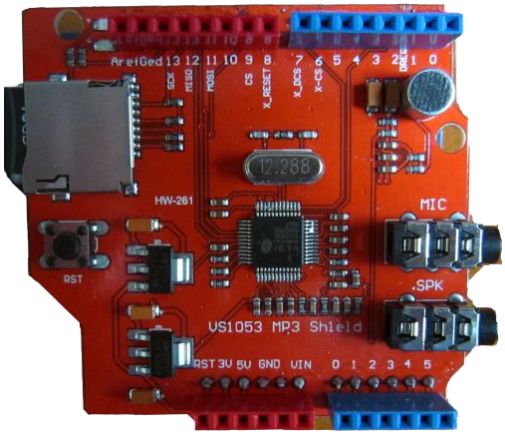
mp3-Player mit einem mp3-Shield



Mit einem mp3-Shield können mp3-Dateien abgespielt werden.

Du benötigst eine Micro-SD-Karte, die mit FAT32 formatiert wurde.

Kopiere mehrere mp3-Dateien auf die SD-Karte.
Du musst die Namen nach folgendem Muster ändern:
track001.mp3, track002.mp3 ...

Benötigte Bibliotheken:

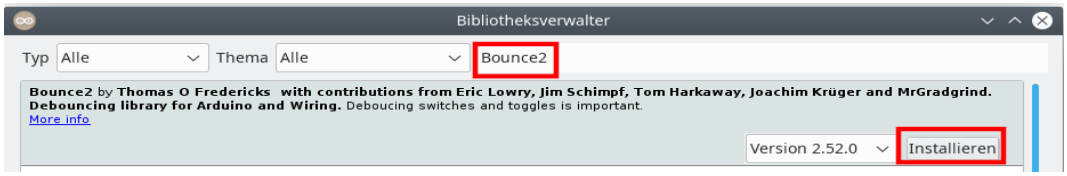
Sketch → Bibliothek einbinden → Bibliotheken verwalten

Bei Programmen, die einen Taster verwenden, taucht immer das Problem auf, dass ein längerer Druck auf den Taster die gewünschte Aktion mehrfach ausführt. Selbst ein kurzes delay() kann das Problem nicht lösen, sondern höchsten abmildern. Dieses Phänomen wird „Prellen“ genannt.
Das kleine Beispielprogramm verdeutlicht das Problem:

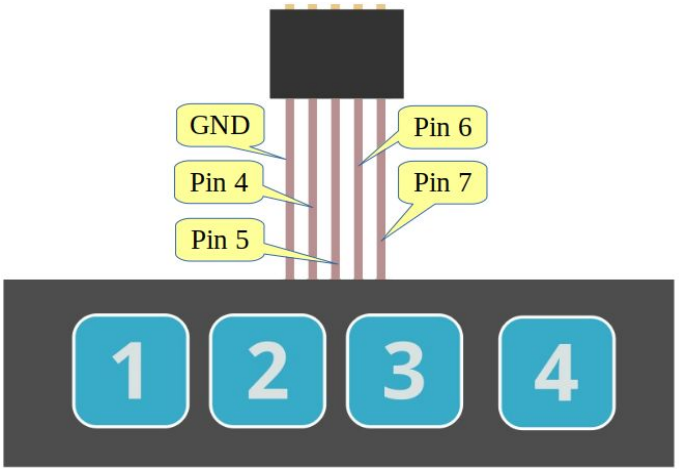
```
void setup()
{
  pinMode(5, INPUT_PULLUP);
  Serial.begin(9600);
}

void loop()
{
  int TasterStatus = digitalRead(5);
  if (TasterStatus == LOW)
  {
    delay(200);
    Serial.println("Taster gedrueckt!");
  }
}
```

Ein längerer Druck auf den Taster produziert die mehrfache Ausgabe des Textes.
Die Bibliothek Bounce2 löst genau dieses Problem:



Stecke das mp3-Shield auf den Arduino und verbinde das Tastenfeld mit den dazugehörigen Pins.



Methoden der Bibliothek vs1503

Methode	**Ausführung**
begin()	Player starten
getState()	Status des Players abfragen 0 = der Player wurde nicht gestartet 1 = der Start des Players war erfolgreich
setVolume(links, rechts)	Lautstärke setzen -> links, rechts 1, 1 sehr laut je größer der Wert, desto leiser
playTrack(Nummer) track001.mp3 → playTrack(1) track002.mp3 → playTrack(2)	spielt den Track (Nummer) Track darf eine wav oder mp3-Datei sein
playMP3(Dateiname) track001.mp3 → playMP3(track001.mp3) track002.mp3 → playMP3(track002.mp3)	im Unterschied zu playTrack() dürfen Dateinamen angegeben werden
stopTrack()	stoppt den gerade laufenden Track
pauseMusic()	pausiert den gerade laufenden Track
resumeMusic()	setzt die Wiedergabe nach der Pause fort
setBassAmplitude(Wert);	Bässe einstellen: erlaubte Werte: 0 bis 15
setTrebleAmplitude(Wert)	Höhen einstellen: erlaubte Werte: -8 bis 7
SendSingleMIDInote()	spielt ein „Beep“

Im Kopf des Programms werden die benötigten Bibliotheken eingebunden und die Variablen definiert.
Beachte die Kommentare.

```
# include <vs1053_SdFat.h>
# include <SdFat.h>
# include <Bounce2.h>

// Bezeichnung der SD-Karte
SdFat sd;

// Bezeichnung des mp3-Shields
vs1053 MP3player;

// Variable für das Lesen des Verzeichnisses
File Verzeichnis;
File Datei;

// die Taster
int TASTER1 = 5;
int TASTER2 = 4;
int TASTER3 = 7;
int TASTER4 = 6;

// "Prellverhinderer" für die Tasten starten
Bounce Taste_play = Bounce();
Bounce Taste_naechster = Bounce();
Bounce Taste_Pause = Bounce();
Bounce Taste_weiter = Bounce();
// wenn ein Taster gedrückt wurde
int TasterStatus;

// Tracknummer/Anzahl der Tracks
int Track = 1;
int TrackMax = 0;
```

Der setup-Teil:

```
void setup()
{
  pinMode(TASTER1, INPUT_PULLUP);
  pinMode(TASTER2, INPUT_PULLUP);
  pinMode(TASTER3, INPUT_PULLUP);
  pinMode(TASTER4, INPUT_PULLUP);
  // Instanzen des Objekts Bounce für jede Taste zuordnen
  // Zeitintervall einstellen
  Taste_play.attach(TASTER1);
  Taste_play.interval(20);
  Taste_naechster.attach(TASTER2);
  Taste_naechster.interval(20);
  Taste_Pause.attach(TASTER3);
  Taste_Pause.interval(20);
```

```
Taste_weiter.attach(TASTER4);
Taste_weiter.interval(20);

// Seriellen Monitor starten
Serial.begin(9600);

// SD Karte starten
sd.begin(SD_SEL, SPI_FULL_SPEED);

// Anzahl der Tracks im Wurzelverzeichnis zählen
// Char-Array für den Dateinamen
char Dateiname[13];
sd.chdir("/", true);

while (Datei.openNext(sd.vwd(), O_READ))
{
  Datei.getName(Dateiname, sizeof(Dateiname));

  // handelt es sich um eine Musikdatei (isFnMusic)
  if (isFnMusic(Dateiname) )
  {
    Serial.print(Dateiname);

    // Dateigröße ermitteln, in MB umwandeln, Punkt durch Komma ersetzen
    float DateiGroesse = Datei.fileSize();
    String Groesse = String(DateiGroesse/1000000);
    Groesse.replace(".", ",");
    Serial.println(" " + Groesse + " MB");
    TrackMax ++;
  }
  Datei.close();
}

Serial.println();
Serial.println("Anzahl der Tracks: " + String(TrackMax));
Serial.println("------------------------------");

// Player starten
MP3player.begin();

// Höhen: erlaubte Werte: -8 bis 7
MP3player.setTrebleAmplitude(4);

// Bässe: erlaubte Werte 0 bis 15
MP3player.setBassAmplitude(7);

// Status des Players ermitteln
if (MP3player.getState() == 1) Serial.println("Player erfolgreich
gestartet");
```

```
  // Lautstärke setzen -> links, rechts -> 1, 1 sehr laut
  // je größer die Werte desto leiser
  MP3player.setVolume(40,40);
}
```

Der loop-Teil. Beachte die Kommentare.

```
void loop()
{
  // aktuellen Track abspielen
  if (Taste_play.update())
  {
    if (Taste_play.read() == LOW)
    {
      // kurzes "Beep" spielen
      MP3player.SendSingleMIDInote();
      MP3player.SendSingleMIDInote();

      // laufenden Track stoppen/aktuellen Track abspielen
      MP3player.stopTrack();
      Serial.println("Spiele Track " + String(Track));
      MP3player.playTrack(Track);
    }
  }

  // nächster Track
  if (Taste_naechster.update())
  {
    if (Taste_naechster.read() == LOW)
    {
      // kurzes Beep spielen
      MP3player.SendSingleMIDInote();
      MP3player.SendSingleMIDInote();
      MP3player.stopTrack();

      // wenn der letzte Track gespielt wurde
      // -> Neustart mit 1
      if (Track < TrackMax) Track ++;
      else Track = 1;
      Serial.println("Spiele Track " + String(Track));
      MP3player.playTrack(Track);
    }
  }

  // Pause
  if (Taste_Pause.update())
  {
    if (Taste_Pause.read() == LOW)
    {
      MP3player.SendSingleMIDInote();
      MP3player.SendSingleMIDInote();
```

```
      // Track pausieren
      Serial.println("Pause Track " + String(Track));
      MP3player.pauseMusic();
    }
  }

  // weiter abspielen
  if (Taste_weiter.update())
  {
    if (Taste_weiter.read() == LOW)
    {
      MP3player.SendSingleMIDInote();
      MP3player.SendSingleMIDInote();

      // Wiedergabe fortsetzen
      Serial.println("Weiter Track " + String(Track));
      MP3player.resumeMusic();
    }
  }
}
```