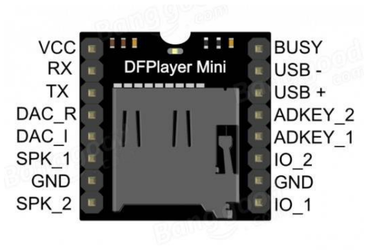
Let's Code Blacksburg! “Make an MP3 Player”

All – https://github.com/LetsCodeBlacksburg/arduino-mp3-player v2018-06-27a_tweeks (CC)(BY)(SA)

This 3hr hands on workshop steps you through building and programming your very own MP3 player based on the DFRobot DFPlayer Mini MP3 player. It requires:



- 1 – Mini USB cable (blue)
- 1 – Arduino board (with extra SVG sensor pin headers or sensor shield)
- 1 – 9v Alkaline Battery w/clip & barrel connector
- 1 – DFPlayer mini, serial controllable MP3 player
- 1 – 1GiB µSD card (w/sounds or music preloaded)
- 1 – Speaker (4-8ohm, 3W max)
- 1 – TM1637 4 x 7-segment LED module (1-wire)
- 2 - 1kΩ resistors (for MP3 serial lines)
- 1 – IR remote & sensor (not used yet)
- 14 – Female/Female dupont connecting wires (assumes SVG connections)
- (opt) - breadboard (if building other things like buttons, etc)

1) Use First Cookbook Recipe - Blink

Before you build anything or do any real programming, you need to first open your the **Files / Examples / 01. Basics / Blink**, click the compile/upload icon _ _, and verify that blink is actually working (blinking your pin 13 LED indicator). If that works, then try changing the blink speed of the LED. This will verify that your computer and the installed arduino IDE software (from www.arduino.cc) is properly configured to talk to your arduino. Get TA sign-off before proceeding:

WARNING: Once you get something working, ALWAYS:

- *SAVE YOUR CODE after each step*
- *give it a meaningful name (like “MP3-Player_01_blink”, in this case)*
- *Before starting new code, open a new program with **Files / New***

TA SIGN-OFF: ____________

2) Use Cookbook Recipe – 7 Segment LED Display w/TM1637 Module

WARNING: Don't use the library/driver in the cookbook. Instead use
After building the hardware and testing the example code found under **File / Examples / TM1637 / TM1637Test**, be sure that all such tests change the following lines for your build's pin usage:

```
…
#define CLK 8//pins definitions for TM1637 and can be changed to other ports
#define DIO 9
…
```

Now that you have the basic display working, save that test code with a meaningful name (e.g. MP3-Player_02_LED-test), and start a new program (use **File / New**) and replace all code with the following for displaying two digit music track numbers:

```
#include <Arduino.h>
#include <TM1637Display.h>

// Module connection pins (Digital Pins). Should match your connections.
#define CLK 8
#define DIO 9

// The amount of time (in milliseconds) between tests
#define TEST_DELAY   20

TM1637Display display(CLK, DIO);
bool leadingZeros = false;

void setup()
{
  display.setBrightness(0xff);
}

void loop()
{
  // This is just a quick for loop to simulate flipping through tracks...
  Serial.println("============== start of playlist ============");

                                  //sndFile++ just increments each loop
  for (int sndFile=0 ; sndFile<100 ; sndFile++) {
    Serial.println(sndFile); // Use to troubleshoot what's going on.
        display.showNumberDec(sndFile, leadingZeros);   // This is it
    delay(TEST_DELAY);
  }
  Serial.println("============== end of playlist ============");
  leadingZeros = leadingZeros ^ true;     // This toggles w&w/out zeros
}
```

NOTE: Be sure to make the CLK and DIO lines match the wiring that you connected your displays pins to.

This is the recipe that will give us our the base hardware and code for our MP3 player's numerical display so you'll want to save it! (e.g. MP3-Player_02_tm1673_sndFile-number . You'll thank us later. ;)

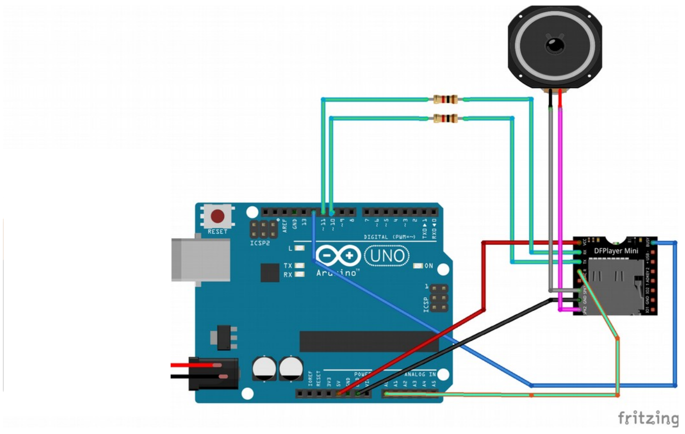
TA SIGN-OFF: ____________

3) Hook Up the DFPlayer Mini, serial controlled MP3 player:

The DFPlayer is powered by 5v, but its input pins are 3.3v and it is very sensitive and will die if it is not hooked up exactly right.

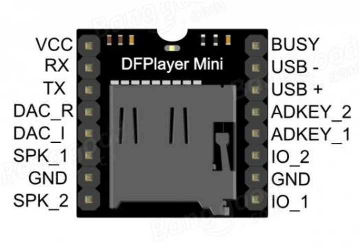
Before hooking up this circuit, leave everything unplugged from USB and the battery and get TA-helper sign-off before you plug in USB or power it up. Hooking this MP3 player wrong can destroy the serial control lines, making it useless for this project.

WARNING: Do not hook up USB cable or 9V battery before getting TA signoff on this section. Not being extra careful at this stage can easily blow the DFPlayer's serial I/O control lines. Ask me how I know… I inadvertently blew two DFPlayers while writing this workshop. Learn from my mistakes.

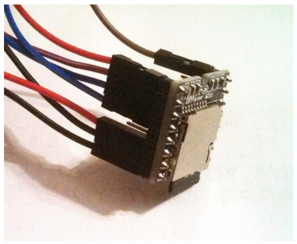


The DFPlayer is connected to 5volts for power, but its serial send/receive lines run at 3.3v. Notice (right) how they MUST be connected to the Arduino *through* two 1k ohm resistors.

If someone has not soldered your resistors into your connect wires (as seen above), then it is recommended you use the included breadboard + resistors to connect the DFPlayer's s. If you do not have breadboard circuit experience, then see the instructor or TA for help, or quickly have a look at the **"Breadboard or Protoboard Recipe" in the cookbook**. Skipping the resistors, will likely kill DFPlayer's serial control lines. DO NOT HOOK UP USB POWER until the TA-Signoff below has been checked off.



Here's what the DFPlayer looks like all using the direct wired method for hooking it up. Pin 1 (5v) is the upper left pin combined with pin 7 (GND) is how we power it. On the upper right is the unit's pin 16 (_BUSY) line which is how we can watch in our program to monitor when the DFPlayer is busy playing a song.



Before proceeding or connecting power or USB, get a TA or Instructor to sign off that you have your unit hooked up correctly.

TA SIGN-OFF: ____________

4) Test Code the DFPlayer MP3 player:
Talking to the DFPlayer requires that you install the <DFRobotDFPlayerMini.h> library. Do this from the arduino menu **Sketch / Include Library / Manage Libraries** and search for the `DFPlayer_Mini_MP3` library. Install it if not already installed.

Here's the code for implementing a simple, sequential sound file MP3 player:

```
#include "Arduino.h"
#include <SoftwareSerial.h>
#include <DFRobotDFPlayerMini.h>

//// HARDWARE SETTINGS/HOOKUPS
///////////////////////////////////////
// Hardware wiring of DFPlayer
//
// DFPlay mini  --Vcc(1)--RX(2)--TX(3)--DACR(4)--GND(7)  +(6) -(8)  ...  _BUSY
//                 |       |      |       |       |      |    |           |
//     Arduino    5v      11*    10*     A0      GND     |    |          12*
//                                                       \Spkr/
//                              * requires 1k resisitor
//DFPlayer mp3 player settings
const int ardRX=10;       // The arduino software Receive lins (goes to DFPlayer TX)
const int ardTX=11;       // The arduino software Transmit line (goes to DFPlayer RX)
const int dfBusy=12;      // From DFPlayer pin 16, active low (HIGH != playing sound)

int sndFile=1;            // Sound file pointer (all files must start with 0001
format).
                          // All files assumed to be in dir named "MP3"
int fileCount=12;         // Either set the max # of files here, or load
                          // it from myDFPlayer.readFileCounts()
  // NOTE: readFileCounts() also sees deleted files and is less reliable.
  // One way to address is by reformatting the SDcard before re-loading musuc.

int setVol=20;            //Set volume value (0~30)

// Software Serial Pins To DFPlayer
SoftwareSerial mySoftwareSerial(10,11); // RX, TX

DFRobotDFPlayerMini myDFPlayer;
void printDetail(uint8_t type, int value);

//// LARGE SETUP BLOCK
/////////////////////////////////////////
void setup(){

//  Set up DFPlay mini
  delay(250);
  pinMode(ardRX, INPUT);
  pinMode(ardTX, OUTPUT);
  pinMode(dfBusy, INPUT);
  delay(100);
  mySoftwareSerial.begin(9600);
  delay(100);
  Serial.begin(9600);
  Serial.println(F("DFRobot DFPlayer Setup"));
  Serial.println(F("Initializing DFPlayer ... (May take 3~5 seconds)"));

  // Check for DFPlay initialization via softserial
  if (!myDFPlayer.begin(mySoftwareSerial)) {
    // if it did not work
    delay(20);
    Serial.println(F("Unable to begin:"));
    Serial.println(F("1.Please recheck the connection!"));
    Serial.println(F("2.Please insert the SD card!"));
    // T-SHOOTING
    Serial.print("INFO: player state / file counts: ");
    Serial.print(myDFPlayer.readState()); //read mp3 state
    Serial.print(" / ");
    Serial.println(myDFPlayer.readFileCounts()); //read all file counts in SD card
    while(true);      // Hang forever if error
  }

  // if serial setup worked
  Serial.println(F("DFPlayer Mini online."));
  Serial.println(F("Reinsert SD card to start playing."));
  // configure settings
  myDFPlayer.setTimeOut(500); //Set serial communictaion time out 500ms
  myDFPlayer.volume(setVol);  //Set volume value (0~30).
  myDFPlayer.EQ(DFPLAYER_EQ_NORMAL);
  myDFPlayer.outputDevice(DFPLAYER_DEVICE_SD);

///// Print Status:
  Serial.print("SETUP-INFO: player state= ");
  Serial.print(myDFPlayer.readState());         //read mp3 state
  Serial.print(" / volume setting(0-30)= ");
  Serial.print(myDFPlayer.readVolume());        //read current volume
  Serial.print(" / EQ setting= ");
  Serial.print(myDFPlayer.readEQ());          //read EQ setting
  Serial.print(" / fileCount= ");
  Serial.println(myDFPlayer.readFileCounts());  //read all file counts in SD card
}


//// MAIN LOOP
////////////////////////////////////////////
void loop()
{
  // If at the last sound file, then loop back to the top (1)
  if (sndFile == (fileCount+1) ) {
    sndFile = 1;
  }

  // Begin playing the next sound file...
  Serial.println("************** PLAYING ****************");
   //myDFPlayer.playFolder("MP3", sndFile);
   myDFPlayer.playMp3Folder(sndFile);
  //myDFPlayer.play(sndFile);   //Play the next mp3
  delay(100);                 // wait to start


  // While DFPlayer _BUSY line is active(low), do whatever you want.
  while(!digitalRead(dfBusy) == true ){
    Serial.println("Player _BUSY, still playing...");
    Serial.println("Good time to read buttons or do stuff.\n");
    delay(300);
  }

  delay(200);
  sndFile++;     // Increment to the next sound file
}
```

```
//// printDetail()
/////////////////////////////////////
void printDetail(uint8_t type, int value){
  switch (type) {
    case TimeOut:
      Serial.println(F("Time Out!"));
      break;
    case WrongStack:
      Serial.println(F("Stack Wrong!"));
      break;
    case DFPlayerCardInserted:
      Serial.println(F("Card Inserted!"));
      break;
    case DFPlayerCardRemoved:
      Serial.println(F("Card Removed!"));
      break;
    case DFPlayerCardOnline:
      Serial.println(F("Card Online!"));
      break;
    case DFPlayerPlayFinished:
      Serial.print(F("Number:"));
      Serial.print(value);
      Serial.println(F(" Play Finished!"));
      break;
    case DFPlayerError:
      Serial.print(F("DFPlayerError:"));
      switch (value) {
        case Busy:
          Serial.println(F("Card not found"));
          break;
        case Sleeping:
          Serial.println(F("Sleeping"));
          break;
        case SerialWrongStack:
          Serial.println(F("Get Wrong Stack"));
          break;
        case CheckSumNotMatch:
          Serial.println(F("Check Sum Not Match"));
          break;
        case FileIndexOut:
          Serial.println(F("File Index Out of Bound"));
          break;
        case FileMismatch:
          Serial.println(F("Cannot Find File"));
          break;
        case Advertise:
          Serial.println(F("In Advertise"));
          break;
        default:
          break;
      }
      break;
    default:
      break;
  }
}
```

There is much more info, t-shooting and DFPlayer functionality included in the dfplayer-test code including the very useful playMp3Folder(sndFile) function as well as code for detecting button presses on our github repo here: https://github.com/LetsCodeBlacksburg/arduino-mp3-player .

Are you able to play music with your Arduino and the MP3 player now? If so, get TA sign-off and be sure to SAVE YOUR PROGRAM. Ours is called: 04_dfplayer_mp3_test

TA SIGN-OFF: ____________

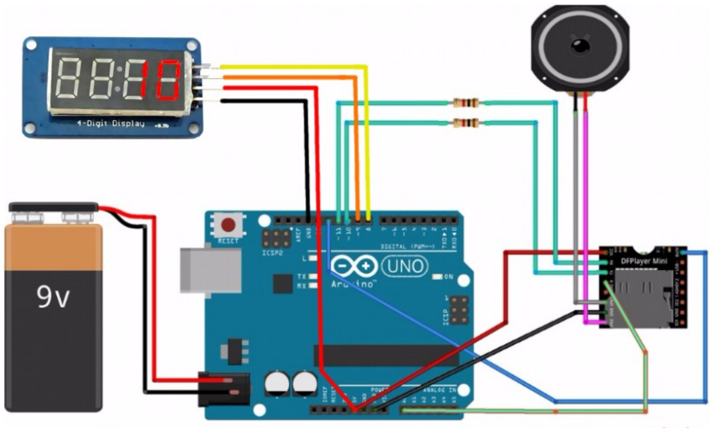
Stretch Goal #1: If you have your code working, check out the class example code 04_dfplayer_mp3_test_nextbutton to see how to read a button press to control your MP3 player. Check out all the functions available to you in the program 04_dfplayer_mp3_test_full.
What can you do with buttons?

5) Bringing It All Together!

Now combine your code from the MP3 player and the LED display (tm1637) to show the number of the song playing, without duplicating your `setup()` and `loop()` blocks of code.

HINT-1: We recommend starting with whatever your biggest program is and adding the smaller program's libraries, variables, setup() and loop() code into the larger program's code blocks.

HINT-2: There's some refinement and smoother functionality in the example code 04_dfplayer_mp3_test_nextbutton. Using it as a starting place and leveraging the info of the sndFile counter as your data source for displaying your song track number will make things much easier.



WARNING: Combining code is dangerous. You can easily corrupt your old, saved programs. So before you even start modifying your code to merge the two, save the destination program with a new name so there's no risk at nuking your old/working program. Then you can always go back to working code if things go south.

Did you get it working? Show it off, get TA sign-off.. and of course.. SAVE YOUR WORK! :)

TA SIGN-OFF: ____________