Music Box

-

Construction Manual

1. Introduction

- **Project overview**

The DIY Music Box Kit is a hands-on project designed to teach the fundamentals of electronics, programming, and 3D printing in an engaging and fun way. Suitable for beginners and makers of all levels, this kit allows users to build and customize their own unique music box from scratch.

By assembling 3D-printed parts, soldering electronic components, and programming the Arduino Nano, users will gain valuable skills in STEM fields while creating a functional and personalized music box. The project blends creative craftsmanship with modern technology, making it perfect for hobbyists, students, and anyone with an interest in DIY electronics.

With this kit, users will learn how to:

- Assemble a 3D-printed structure
- Solder basic electronic components
- Program melodies using Arduino
- Customize their own music box with unique features and designs

This educational project is ideal for developing problem-solving skills, creativity, and technical know-how, while providing a rewarding and enjoyable building experience.

- **Target audience**

This project is designed for beginners in 3D printing and electronics, as well as makers of all skill levels.

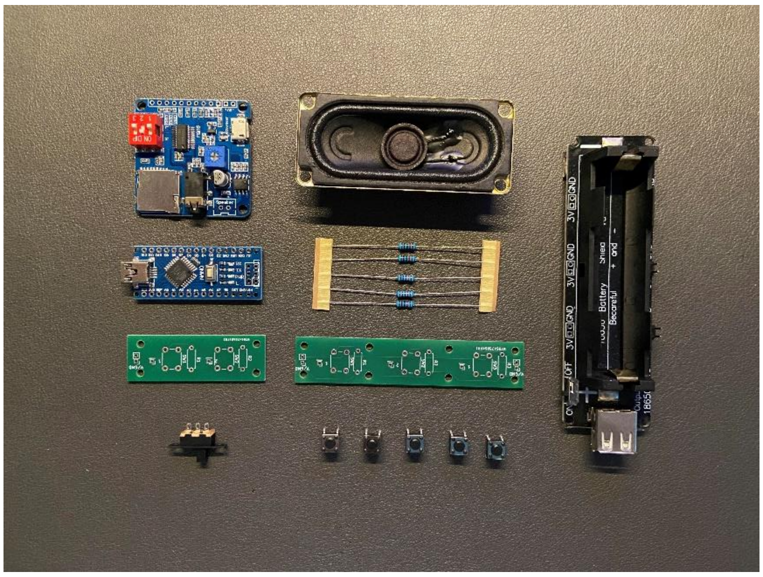
- **Copyright:**

All the 3D parts designs listed in this document are protected by copyright. By downloading it, you agree to use the files for personal purposes only. This means that you do NOT have the right to sell 3D prints of the digital files, including selling products made from molds.

The digital files (STL and other formats) may never be resold, shared, remixed or given away.

2. List of Required Materials

You can purchase all the electronic components in my kit available on my Etsy store. Alternatively, you can source these components from your local electronics supplier, but be cautious about compatibility with the 3D printed parts.



Components included in the kit:

- Arduino Nano
- SV5W sound card
- 18650 battery shield

- Resistors, push buttons and a switch
- 2 custom PCBs
- Screws and hex key

Even though we check all the components before sending them to you, it is recommended to verify that all parts are present in the kit and that they are working properly!

Materials to provide (not included in the kit):

- A 18650 battery
- SD card (max 32GB)

You will also need:

- A computer
- A soldering iron
- A 3D printer (if needed to print parts)
- A multimeter (for debugging)

3. Step 1: Printing the 3D parts

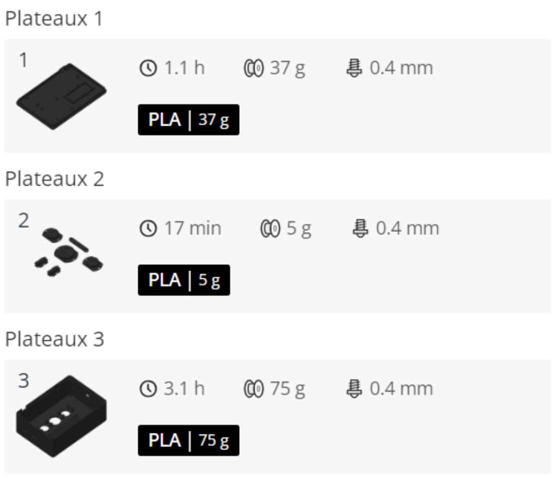
There are 8 parts to print:

- 1x front panel
- 1x back panel
- 1x play button
- 2x back button
- 2x volume button
- 1x Arduino lock

STL files are available on https://makerworld.com/fr/@fabrique_cyril. You will find the printing profiles for the BambuLab A1 and A1 Mini printers.

Otherwise, this design is designed for the following print profile:

- Generic PLA (210°C + printing bed 60°C)
- 0.4mm nozel
- With supports



Plateaux 1

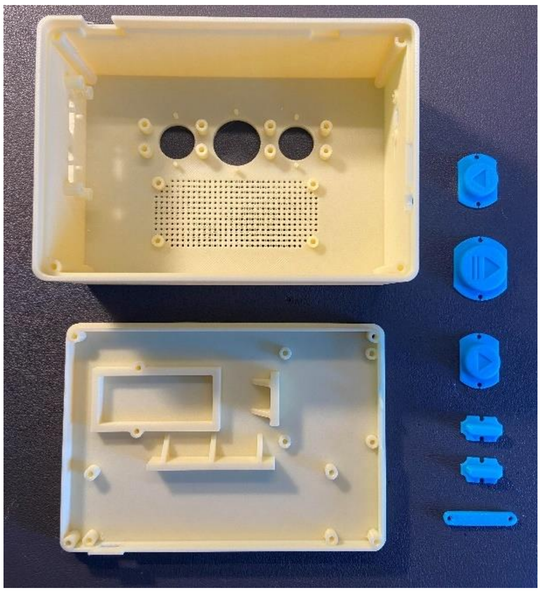
1 1.1 h 37 g 0.4 mm
PLA | 37 g

Plateaux 2

2 17 min 5 g 0.4 mm
PLA | 5 g

Plateaux 3

3 3.1 h 75 g 0.4 mm
PLA | 75 g



4. Step 2: Assembling the Electronic Components

- **Introduction to soldering and safety precautions**

Soldering is a fundamental skill in electronics that involves joining components together by melting solder (a metal alloy) to create a reliable electrical connection. It's essential for assembling circuits, like those in your DIY Music Box, and is a valuable skill for anyone interested in electronics.

While soldering is a straightforward process, it requires precision and attention to safety. Here's a quick guide to get you started.

What You'll Need for Soldering:

- Soldering iron
- Solder (typically lead-free for safety)
- Soldering stand
- Sponge or brass wire (for cleaning the tip)
- Safety glasses
- Ventilation or fume extractor

Basic Steps to Soldering:

1. Heat the Joint: Turn on your soldering iron and let it heat up. Place the tip of the iron on the component lead and the pad where you want to make the connection. Hold it there for a couple of seconds to heat both parts.

2. Apply Solder: Once the joint is heated, touch the solder to the lead and pad—not directly to the soldering iron. The heat from the joint will melt the solder, allowing it to flow and create a bond.
3. Remove the Iron: Once enough solder has flowed into the joint, remove the soldering iron and let the joint cool naturally.
4. Check the Joint: A good solder joint should be smooth, shiny, and cone-shaped. If it's dull or has gaps, it may be a "cold joint," which could cause connection issues.

Safety Precautions:

- Wear safety glasses: Protect your eyes from accidental splashes of molten solder.
- Work in a well-ventilated area: Soldering can produce fumes, especially from the flux, which can be harmful if inhaled. Always solder in a space with proper ventilation or use a fume extractor.
- Avoid burns: The tip of the soldering iron can reach temperatures over 300°C (570°F). Always place the iron back in its stand when not in use and avoid touching the tip or freshly soldered joints.
- Wash your hands after soldering: If you're using lead-based solder, make sure to wash your hands thoroughly after handling it.
- Unplug when not in use: Always unplug the soldering iron when you're finished or taking a break. Never leave it unattended while on.

By following these basic steps and safety precautions, you can ensure that your soldering experience is both effective and safe. With practice, you'll be able to create clean, solid connections for your DIY projects.

- **Soldering resistors, push buttons, and other components to the PCBs**

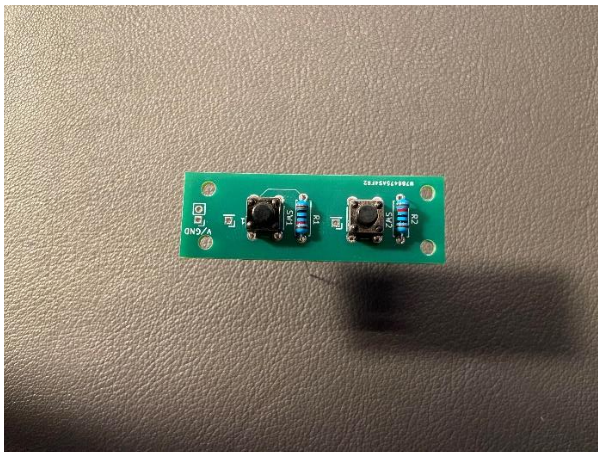
The push buttons will serve as an interface to control the music box:

- - play/pause
- - previous music
- - next music
- - Volume +
- - Volume –

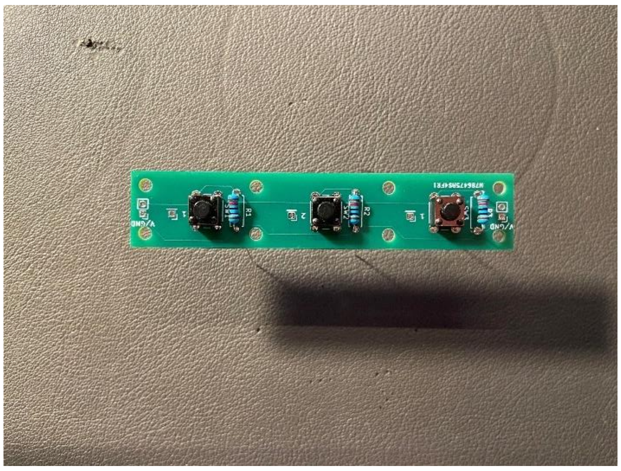
A push button is always followed by a pull up resistor (10kΩ).

Note: The direction of the resistances does not matter. On the other hand, the buttons must be placed in the direction as in the photo (you cannot, however, make a mistake otherwise it will not fit into the PCB).

The PCB with 2 buttons will be used for volume control (+ and – buttons). You will also need to add 2 wires on the opposite sides of buttons and resistors to send commands to the Arduino.



This PCB with 3 buttons will be used for play/pause, previous and next music. You will also need to add 3 wires on the opposite sides of buttons and resistors to send commands to the Arduino.



To transmit power to the PCB with 2 buttons, you will need to solder 2 wires between both squared holes (GND) and between both rounded holes (+5V).

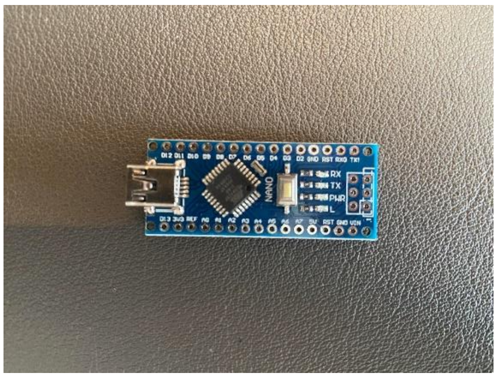
- **Soldering the Arduino Nano and the SV5W sound card**

Soldering resistors and push buttons was a good training, now let's get to the heart of the matter: the microcontrollers!

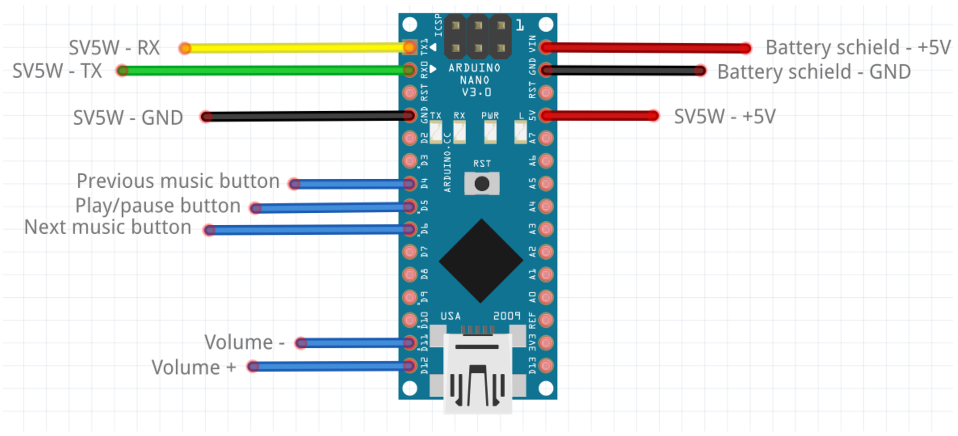
Arduino:

The Arduino Nano is a small, versatile microcontroller board designed for easy integration into various electronics projects. It is based on the ATmega328 microcontroller (similar to the Arduino Uno) but comes in a compact form factor, making it ideal for projects where space is limited. The Nano features 14 digital input/output pins (of which 6 can be used as PWM outputs), 8 analog input pins, a 16 MHz clock speed, 32 KB of flash memory, 2 KB of SRAM, and 1 KB of EEPROM.

It can be powered via USB or an external power supply and has built-in communication interfaces like I2C, SPI, and UART, allowing it to easily connect with sensors, displays, and other components. The Arduino Nano is popular in DIY electronics, robotics, and wearable technology due to its small size, low power consumption, and wide compatibility with Arduino software and libraries.



Follow this soldering diagram:



Note: the Arduino RX/TX pins must be inverted with the SV5W RX/TX pins

SV5W:

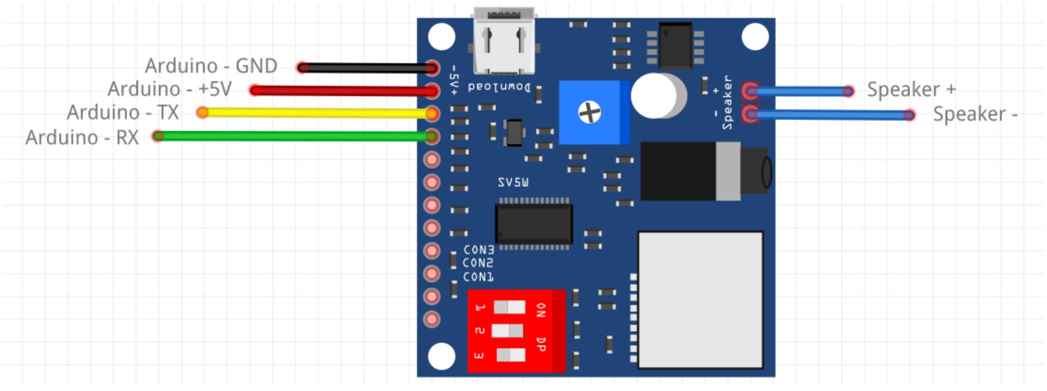
The SV5W sound card is a compact audio processing module designed for projects requiring sound playback and control. It operates at 5V, making it compatible with a wide range of microcontrollers like Arduino, Raspberry Pi, and other DIY electronics platforms. This sound card is capable of playing back audio files (often in formats like WAV or MP3) from external storage devices such as microSD cards, providing high-quality sound output in a small form factor.

The SV5W sound card typically features simple interfaces for controlling playback, volume, and track selection, with input/output pins for easy integration into a larger circuit. Some versions may include a built-in amplifier or external speaker connections, making it versatile for projects such as music boxes, interactive toys, sound effects in robotics, or even home automation systems.

Important : for the circuit to work correctly, the switches must be positioned like this:

- 1 : OFF
- 2 : OFF
- 3 : ON

Follow this soldering diagram:



Be careful with the directions of the speaker + and – symbol must match with the SV5W.

- **Connecting the 18650 battery shield + the toggle switch**

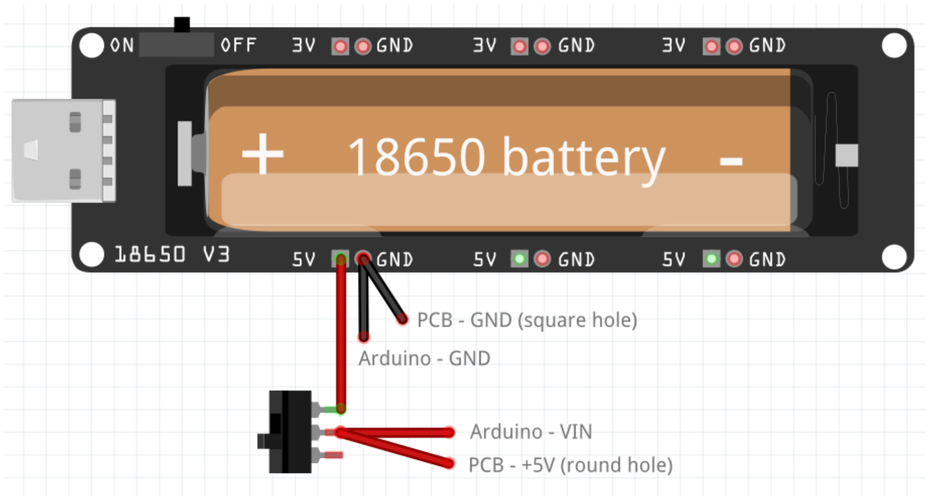
The 18650 battery shield is a versatile power management module designed to hold and utilize one or more 18650 lithium-ion batteries, commonly used in portable electronics due to their high capacity and rechargeability. This shield acts as both a battery holder and a power supply module, converting the battery's output voltage (typically 3.7V per cell) to standard voltages like 5V or 3.3V, suitable for powering microcontrollers, sensors, and other low-power devices.

Key features of the 18650 battery shield often include:

- Battery protection: Built-in overcharge, over-discharge, and short-circuit protection, ensuring the safety of the battery and connected devices.
- Charging capabilities: The shield usually has a USB or micro-USB port for charging the 18650 batteries directly, making it convenient for recharging without needing an external charger.
- Multiple output options: It typically provides multiple voltage outputs, such as 5V and 3.3V, making it compatible with a wide range of electronics projects.
- LED indicators: Status LEDs show the charging state and battery level.

This shield is ideal for use in portable projects, such as robotics, IoT devices, or any DIY electronics requiring a reliable and rechargeable power source. It simplifies power management by providing consistent, regulated voltage and ensures safe and efficient battery use.

Follow this soldering diagram:

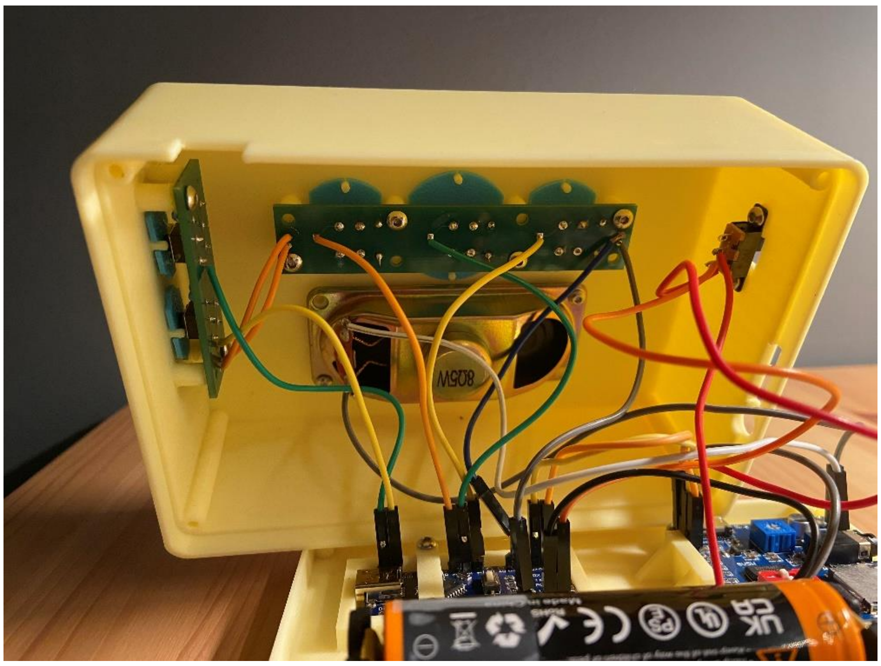


Note: be careful of the switch state before soldering it! It must be exactly as in this diagram to ensure that the switch is in "off" mode.

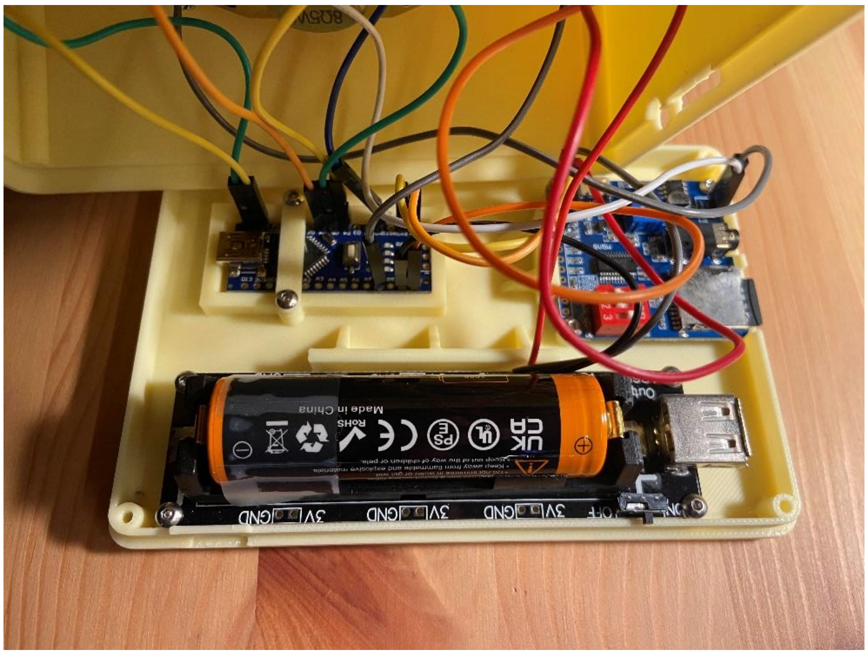
5. Step 3: Assembling the Music Box

Now that you have the 3D printed parts and soldered circuit, it's time to put it all together!

- Screw both PCBs with the M2.5x6 screws as shown above. Make sure the push buttons are facing the printed buttons
- Screw the speaker using M2.5x6 screws
- Screw the toggle switch using M2.5x4 screws



- Screw the battery shied using M2.5x6 screws
- Screw the SV5W using M2.5x6 screws
- Place the Arduino in its slot and screw the Arduino lock printed part above it then screw it using M2.5x6 screws



6. Step 4: Programming the Arduino Nano

- **Installing Arduino IDE software**

Windows

Download the latest release available on https://www.arduino.cc/en/software.

- Double-click the executable (.exe) file.
- Follow the instructions in the installation guide.
- When completing the setup, leave Run Arduino IDE ticked to launch the application, or launch it later from the Start Menu.

macOS

Download the latest release available on https://www.arduino.cc/en/software.

- Double-click the disk image (.dmg) file.
- Drag and drop the Arduino IDE application into the Applications folder.
- Launch Arduino IDE the same way you would launch any other application (such as ⌘ + Space for Spotlight and search for "Arduino").

Linux

Download the latest release available on https://www.arduino.cc/en/software.

- Find the AppImage file in your file manager.
- Make the AppImage file executable:
    - Right-click the file.
    - Choose Properties,
    - Select the Permissions.
    - Tick the Allow executing file as program box.
- Double-click the AppImage file to launch Arduino IDE.

If you need help, refer to the help section available on https://www.arduino.cc/en/software

- **Explanation of the code**

The Arduino program is available on https://github.com/CyrilBvt13/MusicBox.

This code allows you to control an MP3 player using physical buttons connected to a microcontroller (like an Arduino). The buttons let you navigate between tracks, play/pause the music, and adjust the volume. The program uses debounce (via delay(500)) to prevent buttons from triggering multiple actions when pressed.

Defining button pins

```
const int pinPrev = 4;
const int pinPlayPause = 5;
const int pinNext = 6;
const int lowVol = 11;
const int highVol = 12;
```

These constants define the pin numbers to which the buttons are connected on the Arduino board. Each button has a specific function:

- pinPrev: button for going to the previous track.
- pinPlayPause: button to play or pause the music.
- pinNext: button for skipping to the next track.
- lowVol: button to decrease the volume.
- highVol: button to increase the volume.

Creating the DYPlayer instance

```
DY::Player player;
```

player is an instance of thePlayer class, provided by the DYPlayerArduino library, to interact with the MP3 player.

Variable to track the play state

```
bool isPlaying = false;
```

isPlaying is a boolean variable that tracks the current playback state. It's false by default, meaning the music is initially paused.

The setup() function

```
void setup() {
  pinMode(pinPrev, INPUT);
  pinMode(pinPlayPause, INPUT);
  pinMode(pinNext, INPUT);
  pinMode(lowVol, INPUT);
  pinMode(highVol, INPUT);

  player.begin();
  delay(100);
  player.setVolume(20);
  player.setCycleMode(DY::PlayMode::Sequence);
}
```

This function runs once when the program starts:

- The pinMode() commands set the button pins as input (to detect when the buttons are pressed).
- player.begin() initializes communication with the MP3 player.
- delay(100) adds a slight delay to ensure the MP3 module initializes correctly.
- player.setVolume(20) sets the volume to 20 (on a scale where 30 is maximum).
- player.setCycleMode(DY::PlayMode::Sequence) sets the playback mode to sequential (playing the tracks in order).

The loop() function

This function runs continuously and checks the state of the buttons to execute the appropriate commands.

```
if (digitalRead(pinPrev) == HIGH) {
  player.previous();
  delay(500);
}
```

If the "previous" button (pinPrev) is pressed (the signal becomes HIGH), the previous track is played by calling player.previous(). The 500 ms delay helps to prevent button bounce (unintended multiple triggers).

```
if (digitalRead(pinPlayPause) == HIGH) {
  if (isPlaying) {
    player.pause();
    isPlaying = false;
  } else {
    player.play();
    isPlaying = true;
  }
  delay(500);
}
```

If the "Play/Pause" button (pinPlayPause) is pressed, the program checks the current play state:

If the music is playing (isPlaying is true), it pauses the music and changes the state to false.

Otherwise, the music starts playing and the state changes to true.

```
if (digitalRead(pinNext) == HIGH) {
  player.next();
  delay(500);
}
```

If the "next" button (pinNext) is pressed, the player skips to the next track with player.next().

```
if (digitalRead(lowVol) == HIGH) {
  player.volumeDecrease();
  delay(500);
}
```

If the "decrease volume" button (lowVol) is pressed, the volume is decreased with player.volumeDecrease().

```
if (digitalRead(highVol) == HIGH) {
  player.volumeIncrease();
  delay(500);
}
```

If the "increase volume" button (highVol) is pressed, the volume is increased with player.volumeIncrease().

- **Uploading the code to the Arduino Nano**

Now that you have a functional code, It is time to upload it on the Arduino board.

Connect the board to your computer

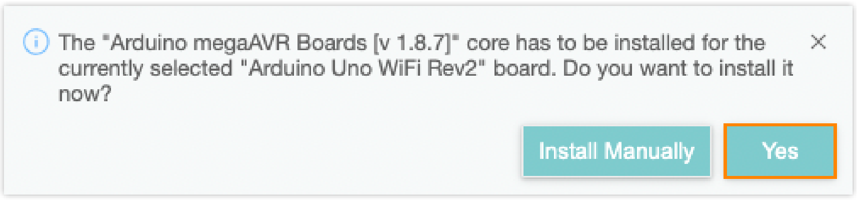
Connect to board to your computer with a USB cable. This will both power the board and allow the IDE to send instructions to the board. You'll need a data USB cable (a charge-only cable will not work), with connectors that fit both the board and your computer.

Install board package

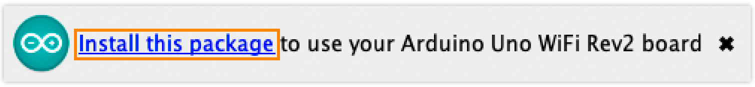
To compile and upload sketches for your board Arduino IDE needs a collection of files for that board called a board package.

When Arduino IDE detects a board with a missing board package, it may ask you to install the missing files:

- In IDE 2, click Yes.


The "Arduino megaAVR Boards [v 1.8.7]" core has to be installed for the currently selected "Arduino Uno WiFi Rev2" board. Do you want to install it now?

- In IDE 1, click Install this package:


Install this package to use your Arduino Uno WiFi Rev2 board

If no prompt appears, proceed with the next step.

Select board and port

Port and board selection can be managed in two ways:

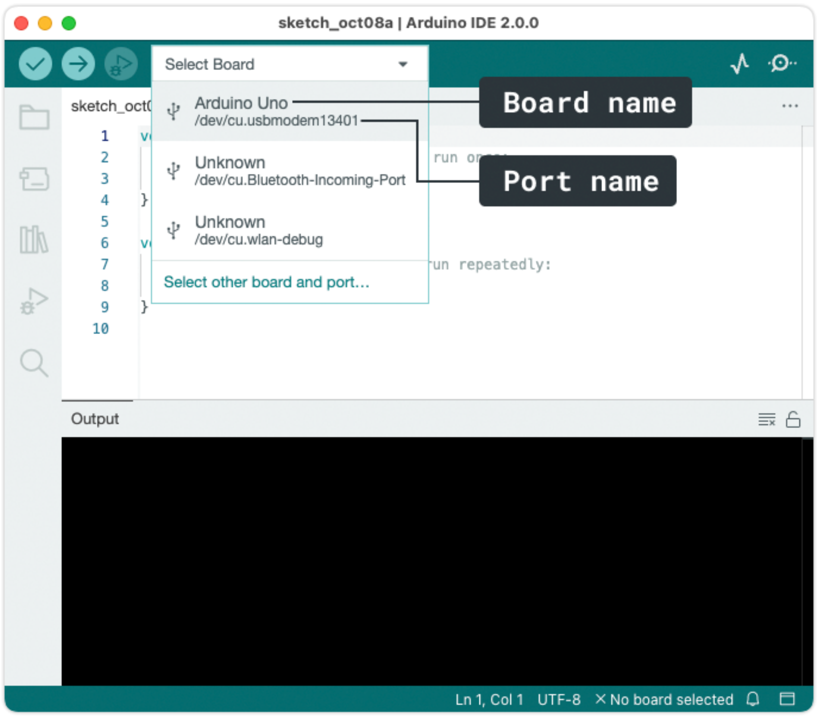
- Using the board selector (requires IDE 2)
- Using the Tools menu

Using the board selector

The board selector is only available in IDE 2.

Follow these steps to use a connected board:

1. Find the board selector and click to open.
2. A list of ports will be displayed. If a board could be identified, the board name will be displayed, otherwise, it will display “Unknown”.


sketch_oct08a | Arduino IDE 2.0.0
Select Board
Arduino Uno
/dev/cu.usbmodem13401
Unknown
/dev/cu.Bluetooth-Incoming-Port
Unknown
/dev/cu.wlan-debug
Select other board and port...
Board name
Port name
Output
Ln 1, Col 1 UTF-8 No board selected

3. Click on a port to select it.

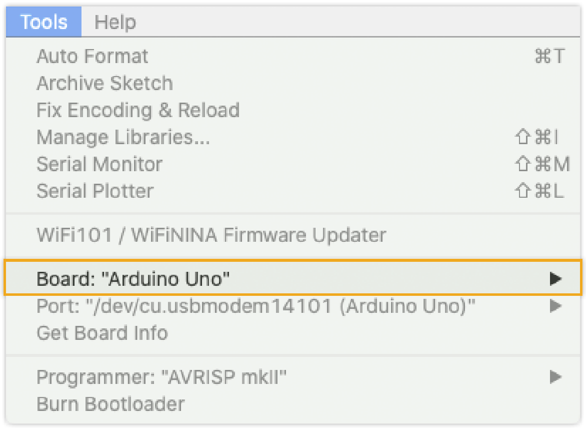
When you select a board, the following may occur:

- If it is unknown, the “Select Other board and port” dialog will open. See Select board and port in Arduino IDE for details.
- If the board could be identified, but you are missing the board platform, you may be asked to install it:
    - Select Yes to automatically install the board package in the background.
    - Select Install manually to view the package in the Board Manager.

Using the Tools menu

Select board:

1. Click on *Tools* in the menu bar and find the *Board* row. If a board is currently selected it will be displayed here.



Board: "Arduino Uno"
Port: "/dev/cu.usbmodem14101 (Arduino Uno)"
Programmer: "AVRISP mkII"

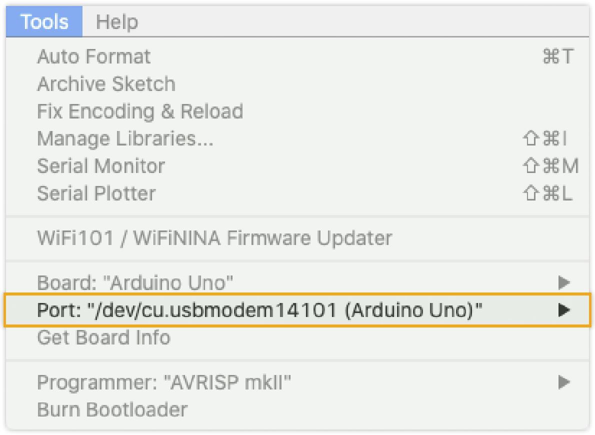
2. Hover over the *Board* row to reveal the installed board packages. Arduino Uno, Arduino Mega, Arduino Nano (classic) use Arduino AVR boards.

3. Click on a board to select it.

Select port:

In IDE 2, the *Tools* > *Port* option will not display if Arduino IDE doesn't detect any ports.

1. Click on *Tools* in the menu bar and find the *Port* row. If a board is currently selected it will be displayed here.



Board: "Arduino Uno"
Port: "/dev/cu.usbmodem14101 (Arduino Uno)"
Programmer: "AVRISP mkII"

2. Hover over the *Port* to reveal all ports. For Arduino devices, the board name will typically be displayed after the port, for example:

   - COM3 (Arduino Uno)
   - /dev/cu.usbmodem14101 (Arduino Uno)
   - /dev/ttyACM0 (Arduino Uno)

3. Click on a port to select it. If the port with your board is already selected you don't have to do anything. If you don't see your board in the list, see If your board does not appear in the port menu.

5. Upload the sketch

1. Make sure you have opened our code for the Music Box
2. **Optional:** Click the ✓ **Verify** button to try compiling the sketch and check for errors.
3. Click the → **Upload** button to program the board with the sketch.

Your sketch will start running on the board. It will run again each time the board is reset/powered.

7. Step 5: Customizing your Music Box

- **How to add music**

To add music to your Music Box, you'll need to open the microSD card on your computer. Simply drag and drop audio files in either WAV or MP3 format onto the card. It's important to name the files in sequence as follows: 00001.wav, 00002.wav, and so on. The Music Box will recognize these filenames and play the tracks in the corresponding order.

Once the files are correctly named and transferred, reinsert the microSD card into the Music Box, and your custom melodies will be ready to play!

8. Step 6: Testing and Adjustments

- **Checking that buttons and music work properly**

Once your Music Box is fully assembled, it's important to check that the buttons and songs are functioning correctly.

Start by powering on the device by pushing the switch, a red led should light up on the Arduino meaning that it is well powered.

Then press each button to ensure they respond properly. Each button should trigger a specific action, such as playing a song or switching to the previous/next track. Make sure the sound quality is clear. If you notice any issues, such as unresponsive buttons or distorted audio, double-check your solder connections and ensure that the audio files on the microSD card are named and formatted correctly.

- **Troubleshooting common issues**

Here you will find a non-exhaustive list of the most common problems. If your problem is not listed, do not hesitate to contact us on our social networks or at lefablabdecyril@gmail.com.

Nothing appends when I push the switch:

- check that the battery is fully charged and that it is correctly positioned in its shield
- check at each soldering point that the current is flowing correctly with a multimeter

Nothing appends when I push the control buttons (volume, play/pause and previous/next music):

- check the components (resistor and push buttons), are they correctly placed?
- check at each soldering point that the current is flowing correctly with a multimeter
- check in the code that you have correctly configured the correct pins in relation to your connections
- check that the 3 switches on the SV5W card are in position 0, 0 and 1 to allow control by RX/TX pins

- **Tips for improving audio quality**

To improve audio quality, you can either modify player.setVolume(20); in the setup function of the Arduino code to increase the volume or you can turn the potentiometer located on the SV5W card.

Double-check your solder connections and ensure that the audio files on the microSD card are correct.

9. Conclusion

- **Summary of the skills learned**

By completing this DIY Music Box project, you gained a variety of valuable skills in several key areas:

- **Basic Electronics:** You learned how to work with components like resistors, push buttons, and PCBs, understanding how circuits function and how to connect different parts to create a functional device.
- **Soldering Techniques:** You mastered the process of soldering, gaining hands-on experience in safely and effectively joining electronic components to a circuit board.
- **Programming with Arduino:** You explored the basics of coding by programming the Arduino Nano to control your music box. You learned how to upload and customize melodies, enhancing your programming skills.
- **3D Printing:** If you printed the music box parts yourself, you gained knowledge in preparing and printing 3D models, as well as best practices for assembling 3D-printed components.
- **Troubleshooting:** You developed problem-solving skills by diagnosing and fixing common issues, whether in the assembly process, programming, or electronics.

- **Creativity and Customization:** You unlocked your creativity by designing and programming your own melodies, and customized the appearance and functionality of your music box to reflect your personal style.

These skills provided you with a strong foundation in STEM (Science, Technology, Engineering, and Mathematics) and gave you the confidence to tackle more complex DIY projects in the future.

- **Encouragement to share or gift the project**

Now that you've completed your DIY Music Box, why not take it a step further? Sharing your creation with friends and family is a wonderful way to showcase your new skills, inspire others to explore the world of STEM, and spread the joy of crafting something unique. You can also turn this music box into a thoughtful, personalized gift for someone special—a handmade project with custom melodies and designs is sure to make a lasting impression!

Whether you share your build process with others or gift the final product, you're passing along not just a beautiful music box, but also the creativity, effort, and learning that went into it.

Thank you for embarking on this DIY Music Box journey! By completing this project, you've not only created a unique and personalized piece but also gained valuable skills in electronics, programming, and 3D printing. I hope this project has sparked your creativity and deepened your interest in STEM.

Feel free to share your creation with others, or gift it to someone special—your hard work is something to be proud of! Keep exploring, learning, and building, and remember that the world of DIY projects offers endless possibilities.

Thank you once again for choosing this kit.

Happy making!