Dominate the Rhythm

Picobricks | Beginner | 30 Mins

1 Overview

Let's turn PicoBricks into a mini music player! 🎵

In this project you'll code the buzzer to play a tune, use the potentiometer to change how fast the music plays, and press a button to start the song.

Music is made of notes. Each note has its own special number called a frequency. If we tell the buzzer the right numbers, in the right order, it can play any song we like!

🎯 What you'll learn

- How to make a buzzer play musical notes
- How to use a potentiometer (dial) to control speed
- How to use a button to start your music
- How to create and use variables

2 What you'll need

🔧 What you'll need

- PicoBricks Kit
- PicoBricks Brick IDE
- MicroUSB to USB cable
- Laptop or tablet for coding

3 Key words

💡 Key words

Potentiometer

A potentiometer is a knob or dial that controls how much electricity flows through a circuit. Think of it like a tap for water: turn it one way and more comes out, turn it the other way and less comes out.

A volume dial on a speaker is a potentiometer, turn it up for louder sound and down for quieter.

Variable

A variable is something that can change. Think of it like a labelled box: you pop something inside, and the label tells you what it is. You can swap what's inside whenever you like.

A variable called score might start at 0, then change to 1, 2 and 3 as you collect points in a game.

Frequency

A frequency is just a number that tells a buzzer which note to play. A small number makes a low sound, and a big number makes a high sound.

Setting a buzzer to a low number gives a deep, rumbly note, while a high number makes a squeaky, high-pitched one, a bit like the left and right ends of a piano.

Buzzer

A buzzer is a tiny speaker that makes beeps and tones when you send it electricity. By telling it different numbers, you can make it play notes and tunes.

The beep a microwave makes when your food is ready comes from a buzzer.

Button

A button is a switch you press to tell your project to do something. The code can check whether it's being pressed or not.

Pressing a doorbell is just like pressing a button, one press sends a signal that makes something happen.

OLED screen

An OLED screen is a small display that shows words, numbers and pictures from your code. On some kits it's labelled the "LED display", but it's the same little screen.

Like the small screen on a fitness watch showing your steps, an OLED screen can show a score or a countdown.

PicoBricks

PicoBricks is an all-in-one coding kit that joins up handy parts, like a screen, a buzzer, buttons and lights, onto one board, so you can build projects without any fiddly wiring.

It's a bit like a Lego baseboard for electronics: everything clicks together in one place, ready to code.

Module

A module is a small part that does one job, like a buzzer, a button or a screen. You connect modules to the main board to give your project new powers.

Like adding different attachments to a toy, each module snaps on to add a new ability, sound, light or buttons.

Function

A function is a chunk of code you give a name to, so you can use it again and again without building it from scratch each time. Think of it like a favourite recipe, you write down the steps once, then follow it whenever you need it.

Like writing out a pancake recipe once. Whenever you fancy pancakes, you just follow the recipe again rather than working it all out from scratch.

Repeat loop

A repeat loop does its instructions a set number of times, then stops. You tell it exactly how many goes you want.

"Repeat 3 times: blow out a candle." After the third candle, the loop is finished, no matter what.

Forever loop

A forever loop repeats its instructions again and again without ever stopping (until you switch off or reset). It's perfect for anything you want to keep happening over and over.

Like a clock's second hand that keeps ticking round and round, a forever loop keeps running the same steps over and over.

If statement

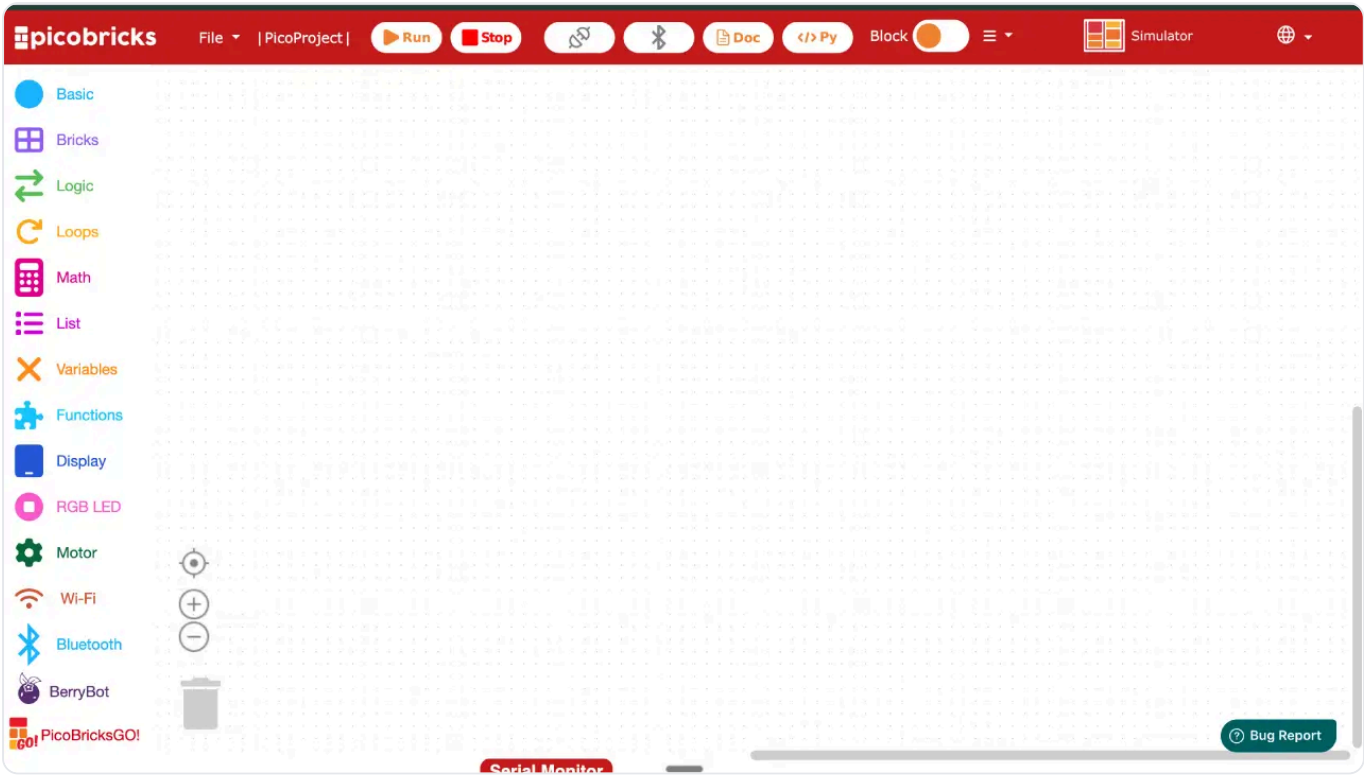
An if statement checks whether something is true, and only runs its instructions when it is. It's how code makes decisions.

"If it's cold, put on a coat." If it's warm, you skip the coat, the action only happens when the check is true.

4 Set up your PicoBricks

1. Open your favourite web browser. We recommend either Google Chrome or Microsoft Edge. Type ide.picobricks.com into the address bar.
2. Connect the PicoBricks board to your computer using the USB cable.

3. Connect the PicoBricks board to the Brick IDE by selecting the **Connection** button at the top of the code area.



5 Create your variables

We need two variables. One called `beat` and one called `rhythm`.

1. Select **Variables**.
2. Select **Create a Variable**.
3. Type `beat` and press **Enter** on your keyboard.
4. Do exactly the same again to make the `rhythm` variable, but this time type `rhythm`.

> The `beat` variable will control how long each note lasts.
>
> The `rhythm` variable will store the speed we read from the potentiometer.

6 Make the music function

A function is a chunk of code you can use again and again. This one plays our tune.

1. Select **Functions**. Select and drag a `to do something` block to the code area and drop it.
2. Select `do something` and type **music**.
3. Select **Bricks** and select and drag a `Play Buzzer Freq` block to the code area. Attach it within the `music` block.
4. Change **300** to **880**.

5. Select **Loops**. Select and drag a `wait` block to the code area and attach it under the `Play Buzzer Freq 880` block.
6. Select **Math**. Select and drag a `1 + 1` block to the code area and attach it within the `wait` block where it says **1**.
7. Select **Variables**. Select the `beat` variable and drag it to the code area, attaching it within the first **1** of the `wait` block.
8. Change the **+** to **x**.
9. Change the remaining **1** to **2**.

This makes the first note last for two beats.

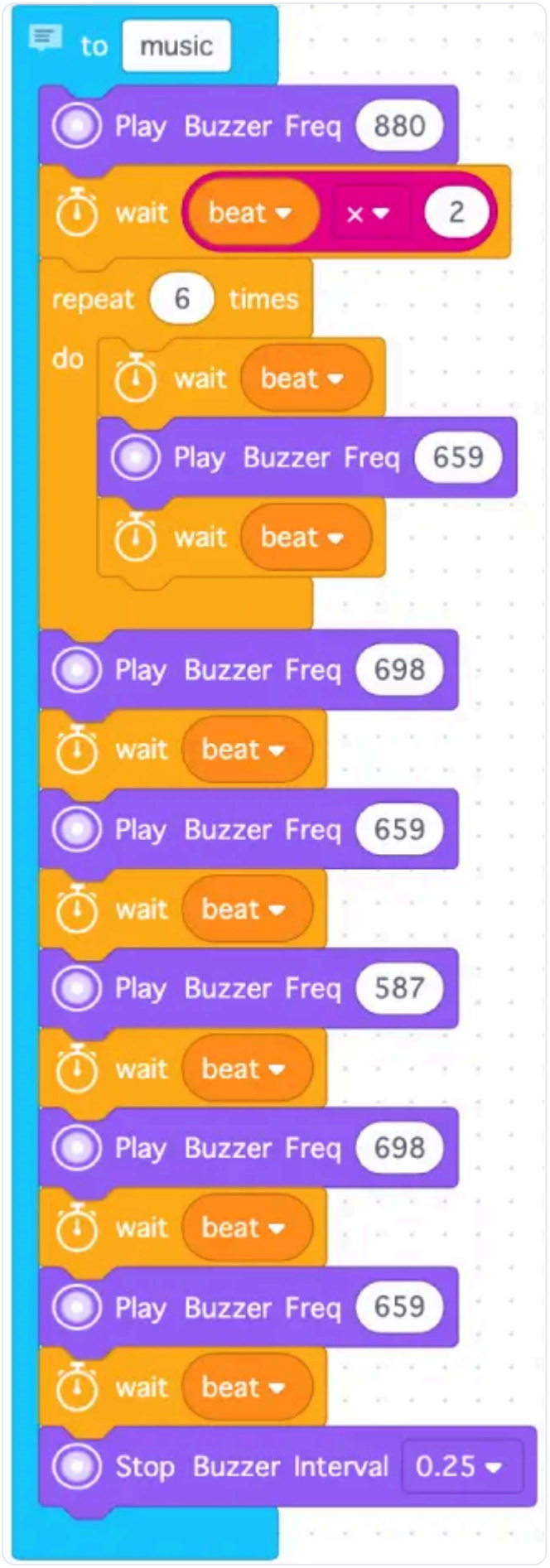
Now add the repeating part of the tune.

1. Select **Loops**. Select and drag a `repeat` block to the code area and attach it below the `wait for 2` block.
2. Change **10** to **6**.
3. Select **Loops**. Select and drag a `wait` block to the code area and attach it within the `repeat` block.
4. Select **Variables**. Select and drag a `beat` block to the code area and attach it within the `wait` block.
5. Select **Bricks**. Select and drag a `Play Buzzer Freq` block to the code area and attach it below the `wait beat` block.
6. Change **300** to **659**.
7. Duplicate the `wait` block by right-clicking and selecting **Duplicate**. Attach the duplicated block under the `Play Buzzer Freq 659` block.

Next, build the rest of the tune.

1. Duplicate a `Play Buzzer Freq` block and attach it under the `repeat` block. Change the number to **698**.
2. Duplicate a `wait` block and attach it below the `Play Buzzer Freq 698` block.
3. Duplicate a `Play Buzzer Freq 659` block and attach it under the `wait` block.
4. Duplicate a `wait` block and attach it under the `Play Buzzer Freq 659` block.
5. Duplicate a `Play Buzzer Freq 659` block and attach it under the `wait` block. Change **659** to **587**.
6. Duplicate a `wait` block and attach it under the `Play Buzzer Freq 587` block.
7. Duplicate a `Play Buzzer Freq 698` block and attach it below the `wait beat` block.
8. Duplicate a `wait beat` block and attach it under the `Play Buzzer Freq 698` block.
9. Duplicate a `Play Buzzer Freq 659` block and attach it under the `wait beat` block.

10. Duplicate a `wait beat` block and attach it below the `Play Buzzer Freq 659` block.
11. Select **Bricks**. Select and drag a `Stop Buzzer Interval` block to the code area and attach it under the `wait beat` block.



7 Build the start-up code

This code runs as soon as you switch on. It reads the dial, shows the speed on the screen, and waits for you to press the button.

> 🧠 **Tricky maths alert!**
>
> This part uses some clever maths to turn the dial into a speed. Don't worry if it feels hard, even grown-ups find this bit tricky!
>
> Just follow each step carefully and copy the blocks exactly. The code reads the dial and works out how fast to play your tune.

1. Select **Basic**. Select and drag a `PicoBricks` block to the code area and drop it.
2. Select **Loops**. Select and drag a forever block to the code area and attach it below the `PicoBricks` block.
3. Select **Variables**. Select and drag a `set rhythm` block to the code area and attach it within the `forever` block.
4. Select **Math**. Select and drag a `round` block to the code area and attach it within the `set rhythm to` block.
5. Select **Math**. Select and drag a `1 + 1` block to the code area and attach it within the **3.1** of the `round` block. Change the **+** to **x**.
6. Select **Math**. Select and drag a `1 + 1` block to the code area and attach it within the second **1** of the `round` block. Change the **+** to **÷**.
7. Select **Bricks**. Select and drag a `Read Potentiometer` block to the code area and attach it within the first **1** of the `round` block.
8. Change the next **1** to **7** and the remaining **1** to **65535**.

Now show the speed on the screen.

1. Select **Display**. Select and drag a `Clear Screen Buffer` block to the code area and attach it above the `set rhythm to` block.
2. Select **Display**. Select and drag a `Write Text to Screen` block to the code area and attach it under the `set rhythm to` block.
3. Change the **X** value to **15**, the **Y** value to **30**, and the text to **Speed**.
4. Duplicate the `Write Text To Screen` block and attach it under the `Write Text To Screen X 15 Y 30 "Speed"` block.
5. Change the **X** value to **70**.
6. Select **Variables**. Select and drag a `rhythm` block to the code area and attach it in place of the text.

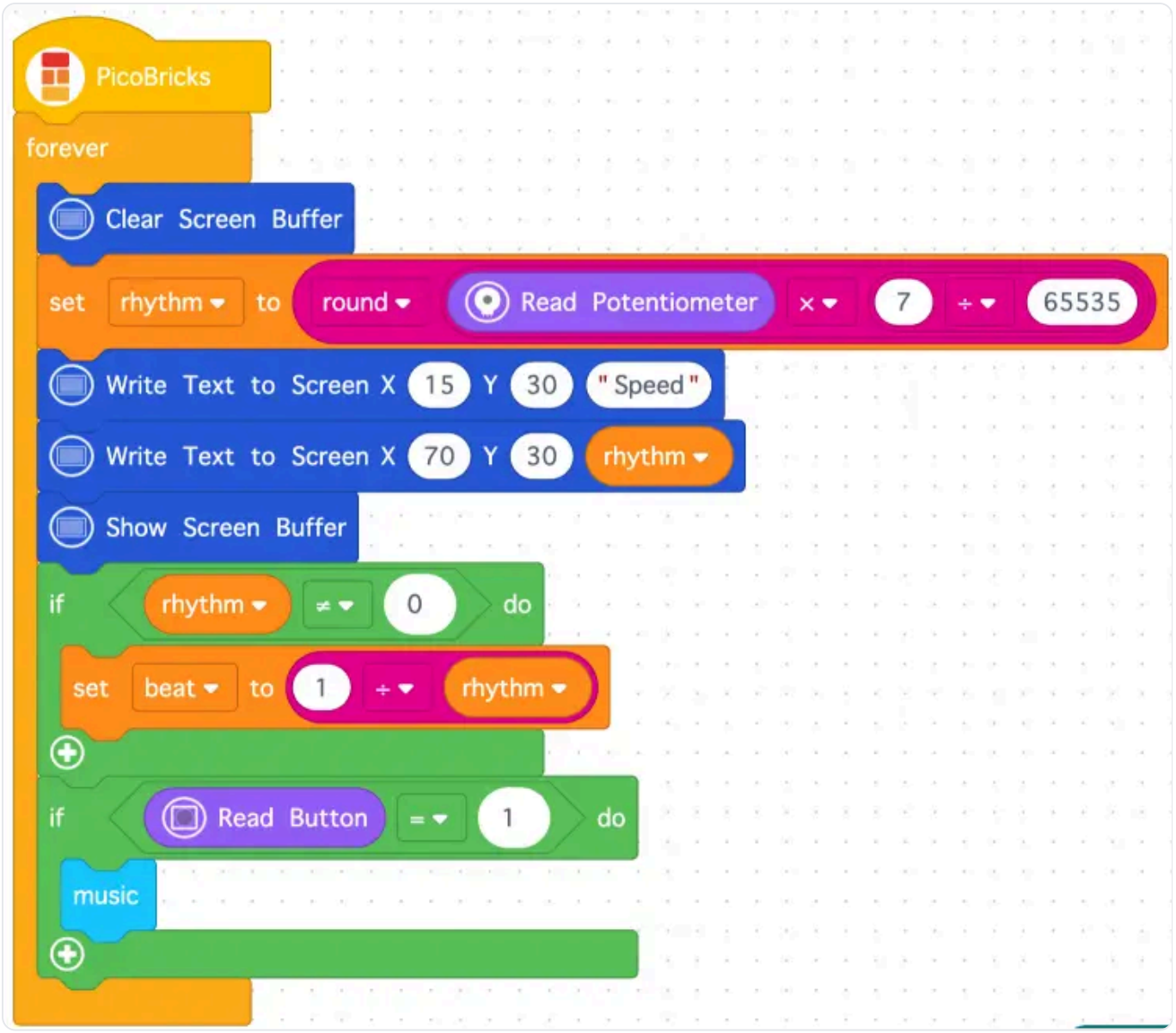
7. Select **Display**. Select and drag a `Show Screen Buffer` block to the code area and attach it under the `Write Text To Screen X 70 Y 30 rhythm` block.

Next, set the beat.

1. Select **Logic**. Select and drag an if block to the code area, attaching it under the `Show Screen Buffer` block.
2. Select **Logic**. Select and drag an `=` block to the code area and attach it within the blank space of the `if` block.
3. Change the **=** to **≠**.
4. Select **Variables**. Select and drag a `rhythm` block to the code area and attach it within the first blank of the `if` block.
5. Select **Math**. Select and drag a `0` block to the code area and attach it within the blank space of the `if` block.
6. Select **Variables**. Select and drag a `set rhythm` block to the code area and attach it within the `if rhythm ≠ 0 do` block.
7. Change **rhythm** to **beat** by selecting the little arrow on the block.
8. Select **Math**. Select and drag a `1 + 1` block to the code area and attach it within the `set beat to` block.
9. Change the **+** to a **÷** sign.
10. Select **Variables**. Select and drag a `rhythm` block to the code area and attach it within the second **1** of the `set beat to` block.

Now add the button press.

1. Select **Logic**. Select and drag an `if` block to the code area and attach it under the `if rhythm ≠ 0 do` block.
2. Select **Logic**. Select and drag an `=` block to the code area and attach it within the blank space of the `if` block.
3. Select **Bricks**. Select and drag a `Read Button` block to the code area and attach it within the first blank space of the `if` block.
4. Select **Math**. Select and drag a `0` block to the code area and attach it in the blank space of the `if` block.
5. Change the **0** to **1**.
6. Select **Functions**. Select and drag a `music` block to the code area and attach it within the `if Read Button = 1 do` block.



8 Run it and listen

Now the code is finished, let's run it and check it works.

Select the **Run** button at the top of the code area.

Turn the potentiometer to change the speed of the music. You'll see the speed appear on the LED display. When you're ready, press the button to play your tune! 🎶

🎶 You have made your own PicoBricks music player. Brilliant work!

9 Try it yourself

🚀 Try it yourself

Challenge: Try changing the frequency numbers in your music function to make up your very own tune. What happens if you use bigger numbers? What about smaller ones?

10 Stuck? Quick fixes

Problem	Try this
You can't hear any music	Check the buzzer module is connected properly with its cable.
The speed doesn't change when you turn the dial	Check the potentiometer module is plugged in with its cable.
Nothing happens when you press the button	Make sure the button module is connected, and check your `if Read Button = 1` block.
Your code won't run	Check the PicoBricks board is plugged into your computer with the USB cable.