MODULE 4: Creating Your Soundbank, Variables, & Tracks

Module Structure

Icons on slides

in the bottom right corner indicates a coding activity students can complete. Instructions for coding activities are in the Student Coding Activities Workbook

in the bottom right corner indicates a writing activity students can complete. Instructions for writing activities are in the Student Writing Activities Workbook

Module Overview

In Module 4, students continue to examine the **layers** of racism and music. Students will learn how to create a SOUNDBANK and code instrumental layers using variables and functions in their song.

Objective & Key Areas of Learning

Make Beats

- Songs are created by layering many different soundtracks on top of each other.
- You can upload your own sounds through EarSketch in the sound browser — just select + Add sound.
- You can record or upload sounds with a Creative Commons License into EarSketch.

Learn Code

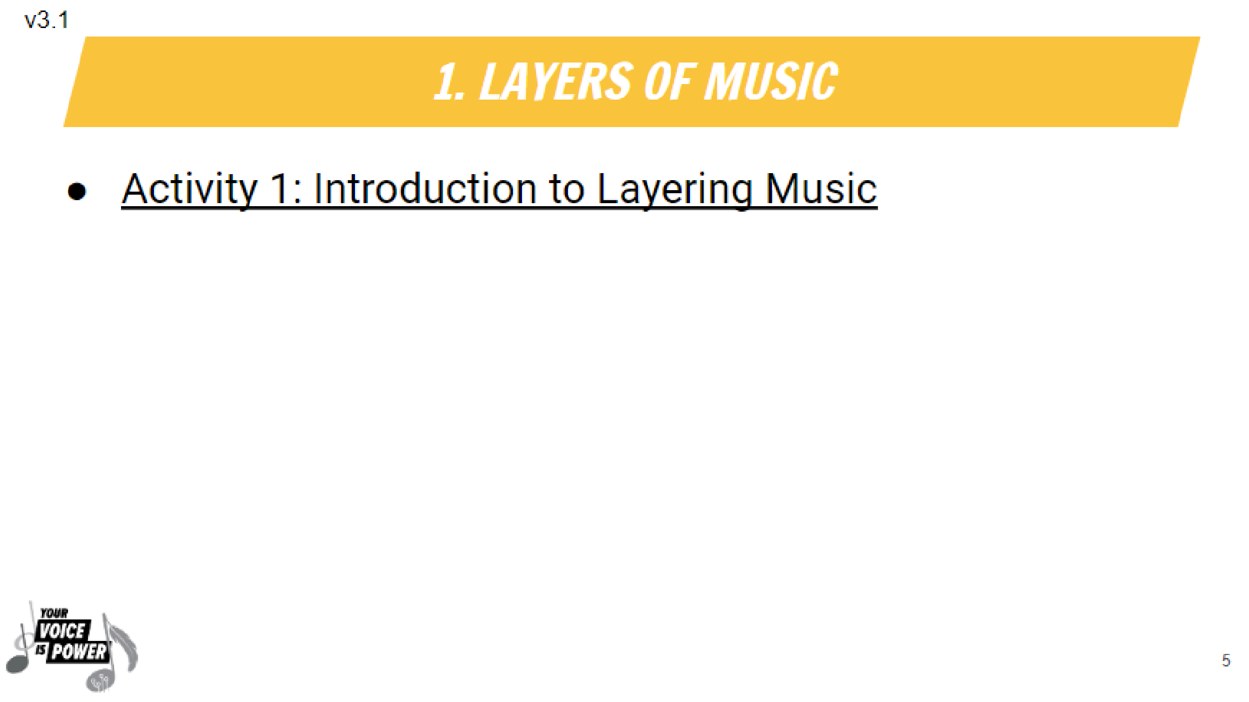
- A variable creates a space in the computer's memory to store data.
- A function is a piece of code that performs an action.
- fitMedia() is a function that uses four parameters (sound clip, track, starting measure, and ending measure) to add audio clips to a track.

Module Resources

- Slides for Module 4
- Videos:
    - Layering in Music
    - Peuple Invincible
    - Adding your first sound with fitMedia()
    - Using variables for sounds and adding song structure

1. MINDS-ON: LAYERS OF MUSIC

Slide 5



v3.1

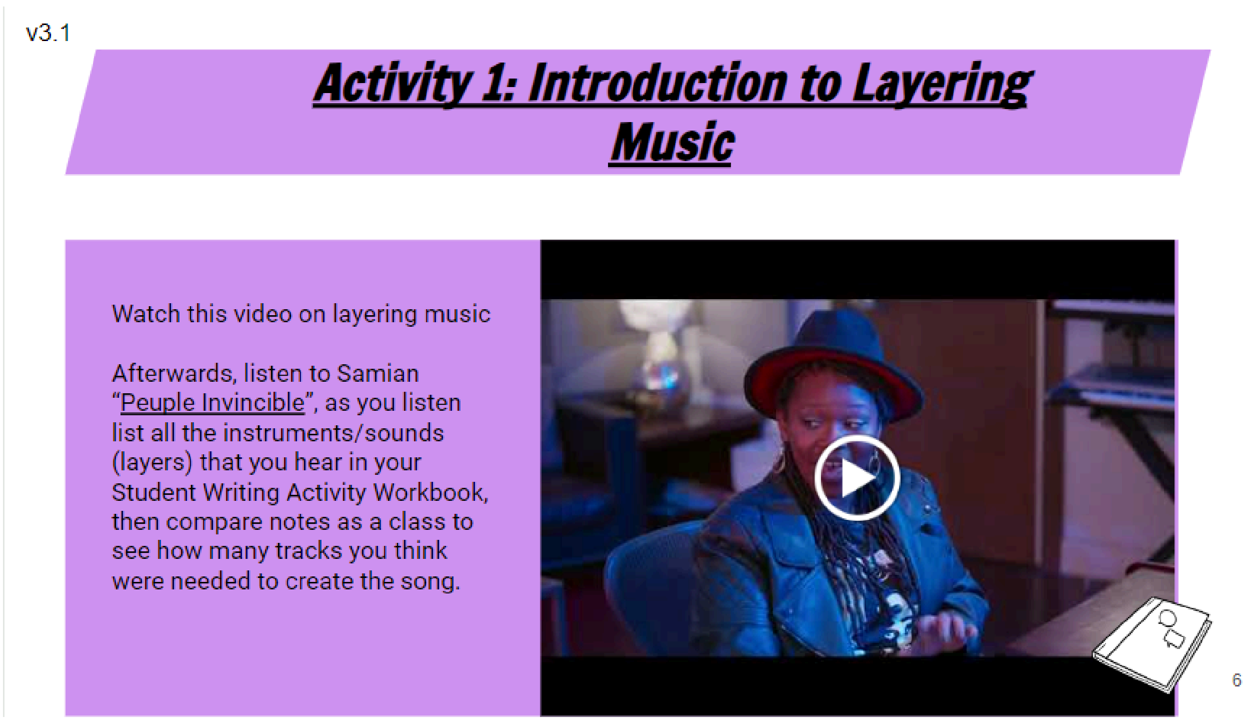
1. LAYERS OF MUSIC

- Activity 1: Introduction to Layering Music

5

Activity 1: Introduction to Layering Music (15 minutes)

Slide 6



v3.1

Activity 1: Introduction to Layering Music

Watch this video on layering music

Afterwards, listen to Samian "Peuple Invincible", as you listen list all the instruments/sounds (layers) that you hear in your Student Writing Activity Workbook, then compare notes as a class to see how many tracks you think were needed to create the song.

6

Instructor Dialogue

Now that you've explored the layers of racism in the last lesson, we are going to explore the layers that exist within music.
How many different instruments and sounds do you think Samian used to create the song?

*Let's listen to the song, as you listen write in your Student Writing Activity Workbook **Module 4 - Activity 1: Introduction to Layering Music** the different sounds and instruments you hear*

Instructor Cue

Play verse 1 of Samian's Peuple Invincible again and have the students write down in their Student Writing Activity Workbook **Module 4 - Activity 1: Introduction to Layering Music** the various instruments and sounds that make up the song

Have students share some of the sounds and then Watch the video: Layering in Music (8 minutes)

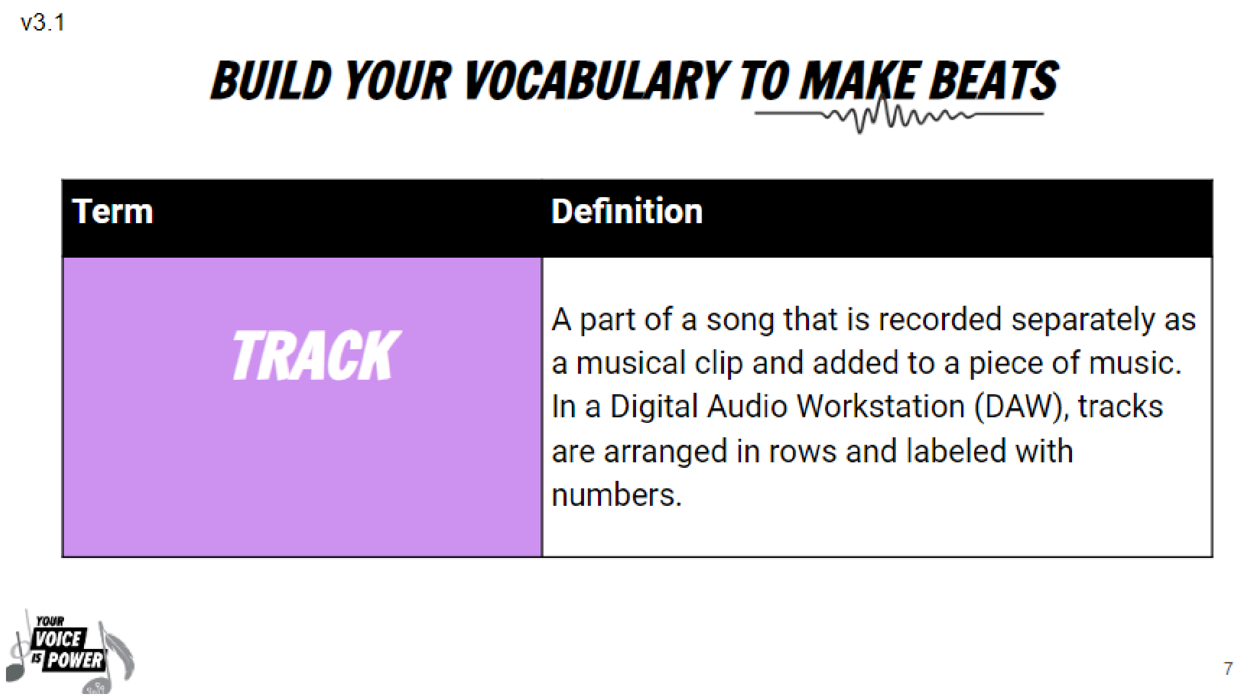
Student Writing Activity: Module 4 - Activity 1: Introduction to Layering Music

Students should follow the instructions in their Student Writing Activity Workbook

Listen to Samian's Peuple Invincible and list the instruments and sounds you hear in the song.

List the instruments you hear in the song
Answers will vary but might include cymbals, wind, string instruments, flute, drums, chanting

Slide 7



v3.1

BUILD YOUR VOCABULARY TO MAKE BEATS

Term	Definition
TRACK	A part of a song that is recorded separately as a musical clip and added to a piece of music. In a Digital Audio Workstation (DAW), tracks are arranged in rows and labeled with numbers.

YOUR VOICE IS POWER

7

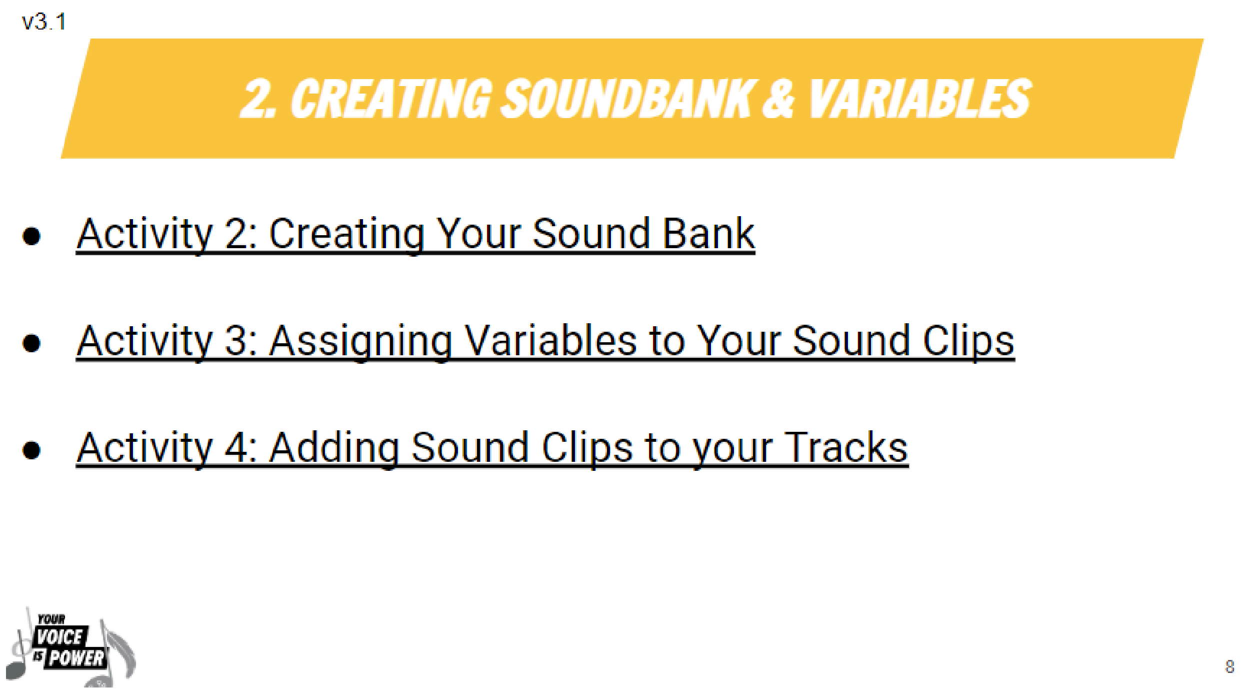
Instructor Dialogue

Each of the sounds you heard makes up a layer of music, in our DAW each layer is on a single track.

Today, you are going to learn how to do exactly what the artist Chalece did.
You will import sounds into your song and use the fitMedia() function to code at least 5 tracks to create a short song.

2. ACTION: CREATING A SOUNDBANK AND VARIABLES

Slide 8



v3.1

2. CREATING SOUNDBANK & VARIABLES

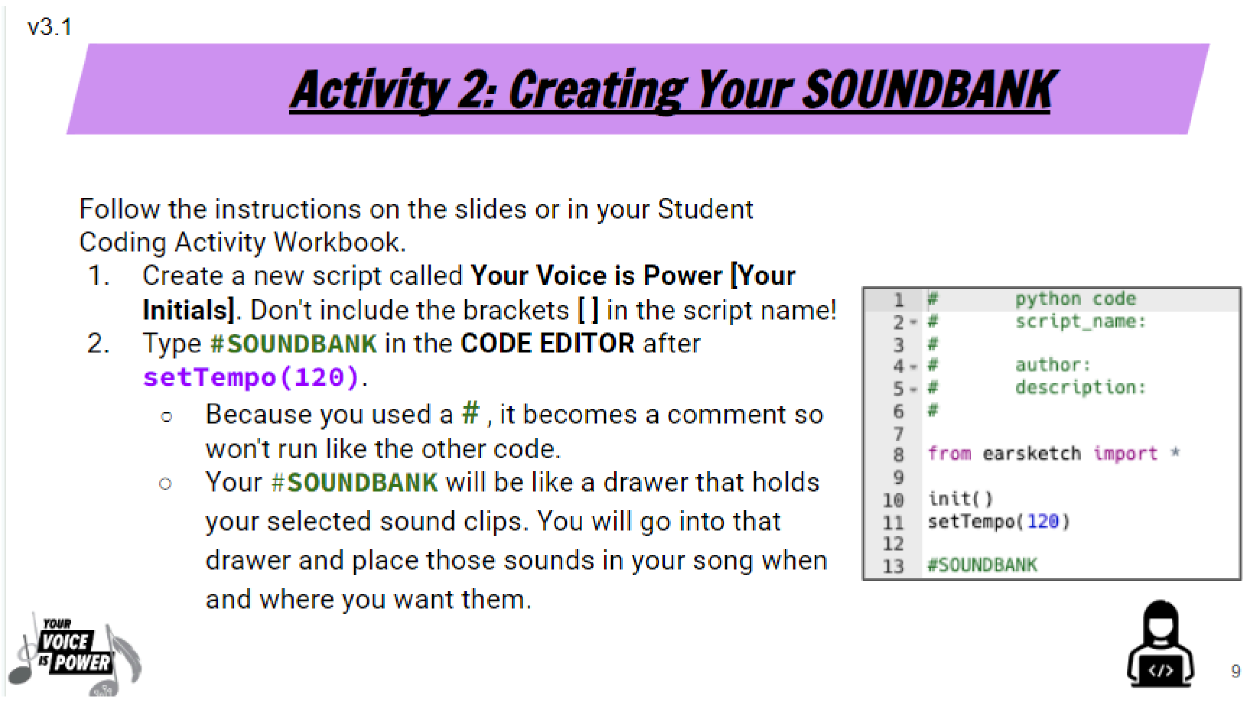
- Activity 2: Creating Your Sound Bank
- Activity 3: Assigning Variables to Your Sound Clips
- Activity 4: Adding Sound Clips to your Tracks

YOUR VOICE IS POWER

8

Activity 2: Creating Your Sound Bank (15 minutes)

Slide 9



v3.1

Activity 2: Creating Your SOUNDBANK

Follow the instructions on the slides or in your Student Coding Activity Workbook.

1. Create a new script called **Your Voice is Power [Your Initials]**. Don't include the brackets **[ ]** in the script name!
2. Type **#SOUNDBANK** in the **CODE EDITOR** after **setTempo(120)**.
   - Because you used a **#** , it becomes a comment so won't run like the other code.
   - Your #**SOUNDBANK** will be like a drawer that holds your selected sound clips. You will go into that drawer and place those sounds in your song when and where you want them.

```
#         python code
#         script_name:
#
#         author:
#         description:
#

from earsketch import *

init()
setTempo(120)

#SOUNDBANK
```

9

Instructor Dialogue

Now that you have the outline of your script, let's add some music. EarSketch has audio stems from genres such as R&B, Hip Hop, EDM, House, Pop, and Funk (and many more), and from over 300 different instrument samples. We are proud to present sound clips from our participating Indigenous Canadian musicians—Samian, Jayli Wolf, and Dakota Bear. In addition to these, we have sound clips from recording artists and sound engineers such as Pharrell, Ciara, Common, Richard Devine, and Young Guru. (Select the artist's name to learn more about their music.)

You will create a #SOUNDBANK with all our favourite sounds that we will use during our song.

Follow along the next two slides and then try it yourself following the instructions in your Student Coding Activity Workbook: ***Module 4 - Activity 2: Creating Your Soundbank***

Instructor Cue

AFTER you do a demo or show them the slides demonstrating the steps, have the students begin the activity *Student Coding Activities Workbook* ***Module 4 - Activity 2: Creating Your Soundbank***

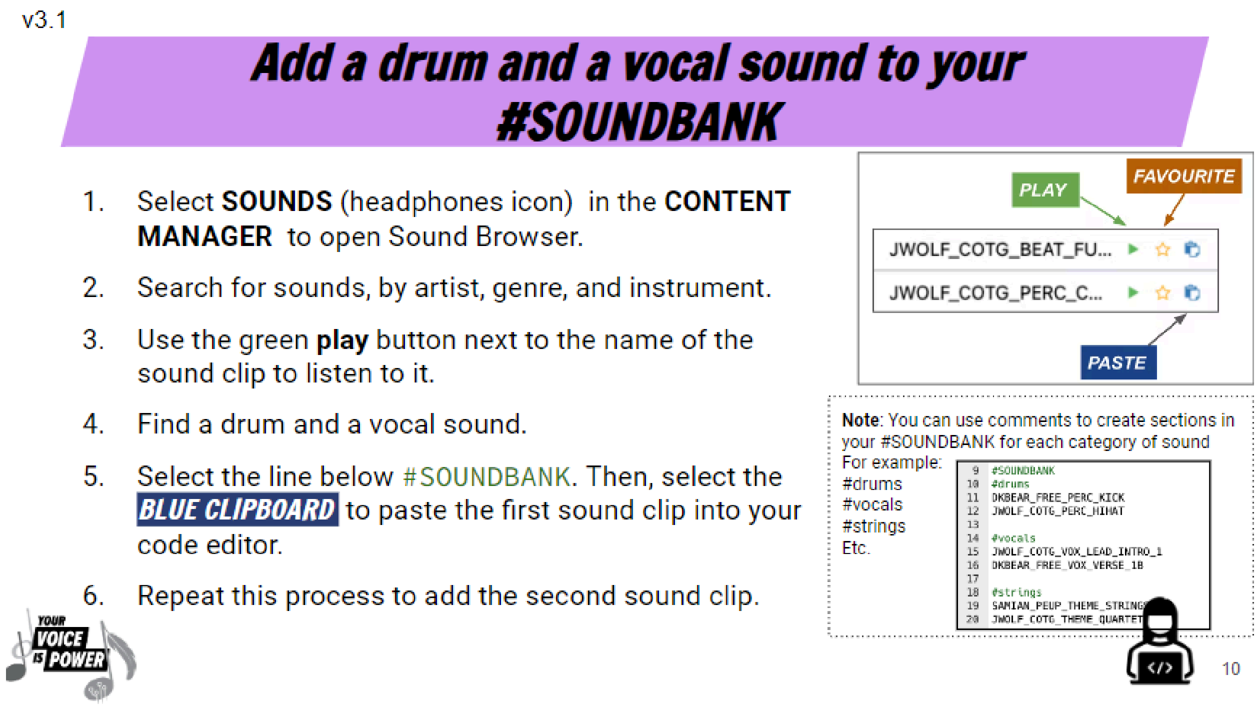
To complete a demo follow the steps on the slides to add a SOUNDBANK to your script and then search for a sound and add it to your soundbank.

OR

Show the students the next two slides to help them understand the steps they will perform to complete the activity.

The Student Coding Workbook contains detailed instructions including screenshots.

Slide 10



v3.1

Add a drum and a vocal sound to your #SOUNDBANK

1. Select **SOUNDS** (headphones icon) in the **CONTENT MANAGER** to open Sound Browser.
2. Search for sounds, by artist, genre, and instrument.
3. Use the green **play** button next to the name of the sound clip to listen to it.
4. Find a drum and a vocal sound.
5. Select the line below #SOUNDBANK. Then, select the ***BLUE CLIPBOARD*** to paste the first sound clip into your code editor.
6. Repeat this process to add the second sound clip.

Note: You can use comments to create sections in your #SOUNDBANK for each category of sound
For example:
#drums
#vocals
#strings
Etc.

```
#SOUNDBANK
#drums
DKBEAR_FREE_PERC_KICK
JWOLF_COTG_PERC_HIHAT

#vocals
JWOLF_COTG_VOX_LEAD_INTRO_1
DKBEAR_FREE_VOX_VERSE_1B

#strings
SAMIAN_PEUP_THEME_STRINGS
JWOLF_COTG_THEME_QUARTET
```

10

Instructor Dialogue

After you create a new script and add a comment to create a SOUNDBANK,

Search for sounds in the CONTENT MANAGER
Use the PLAY button to preview the sound
Select a drum and vocal sound and add them to your soundbank.

Now it's your turn, follow the instructions in your Student Coding Activity Workbook:
***Module 4 - Activity 2: Creating Your Soundban**k*

Instructor Cue

Have the students begin the activity *Student Coding Activities Workbook* ***Module 4 - Activity 2: Creating Your Soundban**k*

Related Video tutorials:

- Adding your first sound with fitMedia()

Student Coding Activity: Module 4 - Activity 2: Creating Your Soundbank

Students should follow the instructions in their Student Coding Activity Workbook

Related Video tutorials:

- Adding your first sound with fitMedia()

1. Create a new script called **Your Voice is Power [Your Initials]**.

   Don't include the brackets **[ ]** in the script name!

2. Create your sound bank.

   Your sound bank will be like a drawer that holds your selected sound clips. You will go into that drawer and place those sounds in your song when and where you want them.

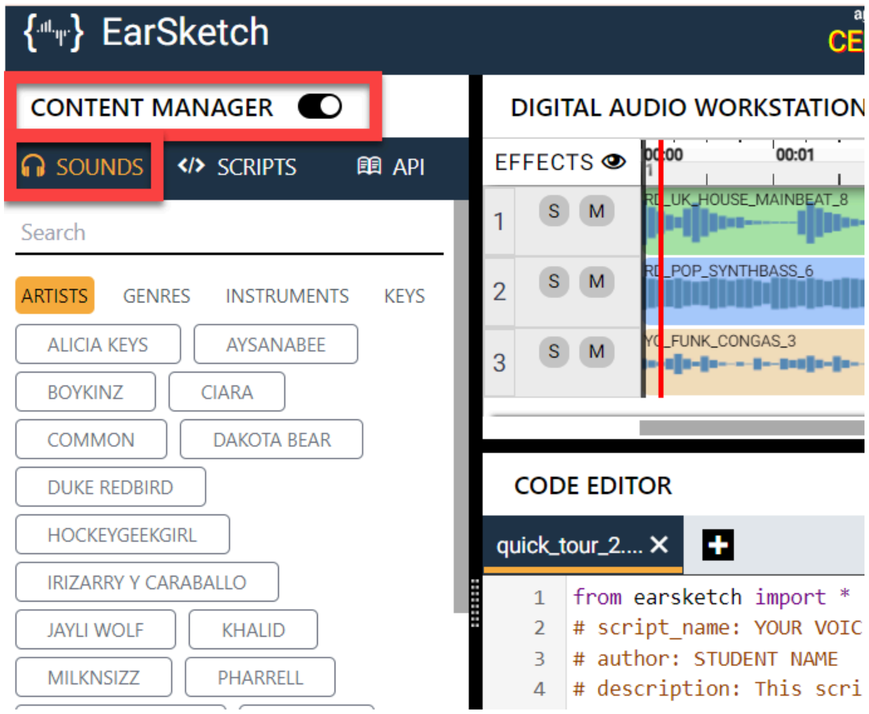
   2.1. Type **`#SOUNDBANK`** in the **CODE EDITOR** after **`setTempo(120)`**

   Do you remember what the # does? We saw that in Module 2. The # makes it a comment so it won't be executed as a line of code.

   Your finished code should look similar to the following, the code changes are highlighted:

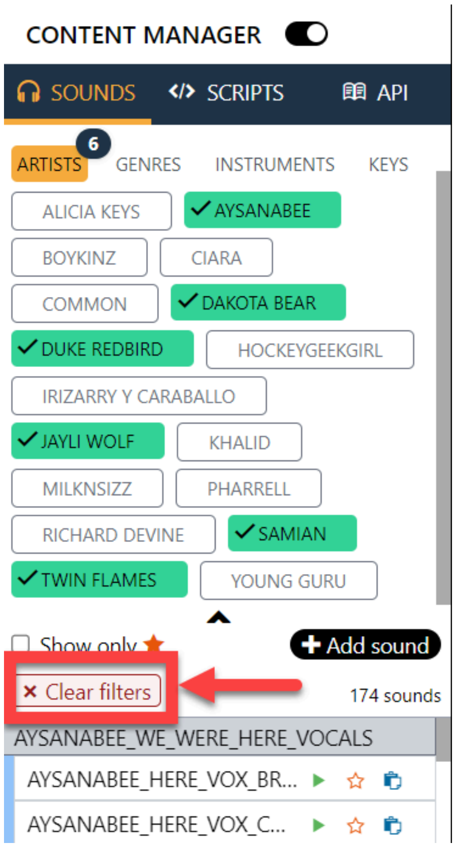
```
# description:
from earsketch import *

setTempo(120)

# SOUNDBANK
```

1. Listen to and choose some great beats
   1.1. Access the sound browser by selecting **SOUNDS** (headphones icon) in the **CONTENT MANAGER**

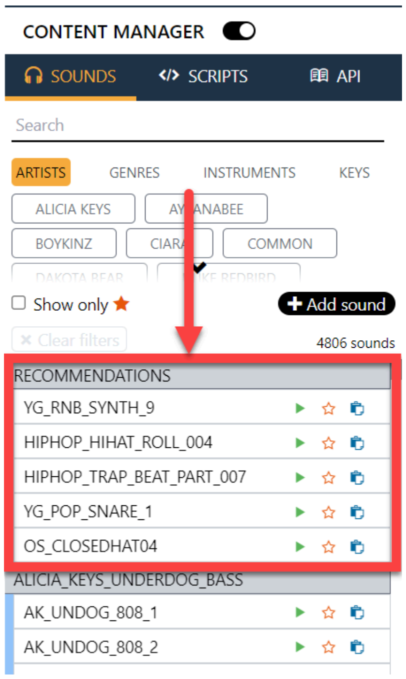


```
from earsketch import *
# script_name: YOUR VOIC
# author: STUDENT NAME
# description: This scri
```

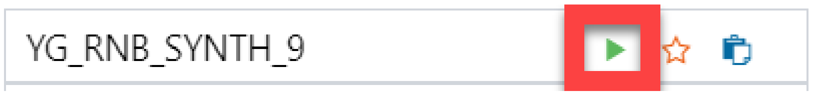
1.2. Search for sounds by artist, genre, or instrument. You can reset your search at any time using the **Clear filters** button in the bottom left corner.



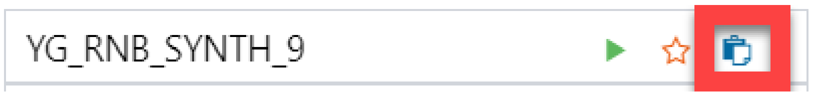
1.3. If you have no filters applied, the sound manager will list recommendations for sounds that fit your script or sounds used by others who have searched for similar sounds.



1.4. Use the green play button next to the name of the sound clip to listen to it and decide if you like it.



1.5. Use the blue clipboard icon to paste a sound clip into your code editor under the **#SOUNDBANK** heading. TIP: It will paste the sound clip to wherever the cursor was last located in the **CODE EDITOR**.



1.6. Add two sound clips to your **#SOUNDBANK** – one drum sound and one vocal sound.

TIP: You might want to add comments to create separate sections in your #SOUNDBANK for sound clips from different musical instruments. For example:

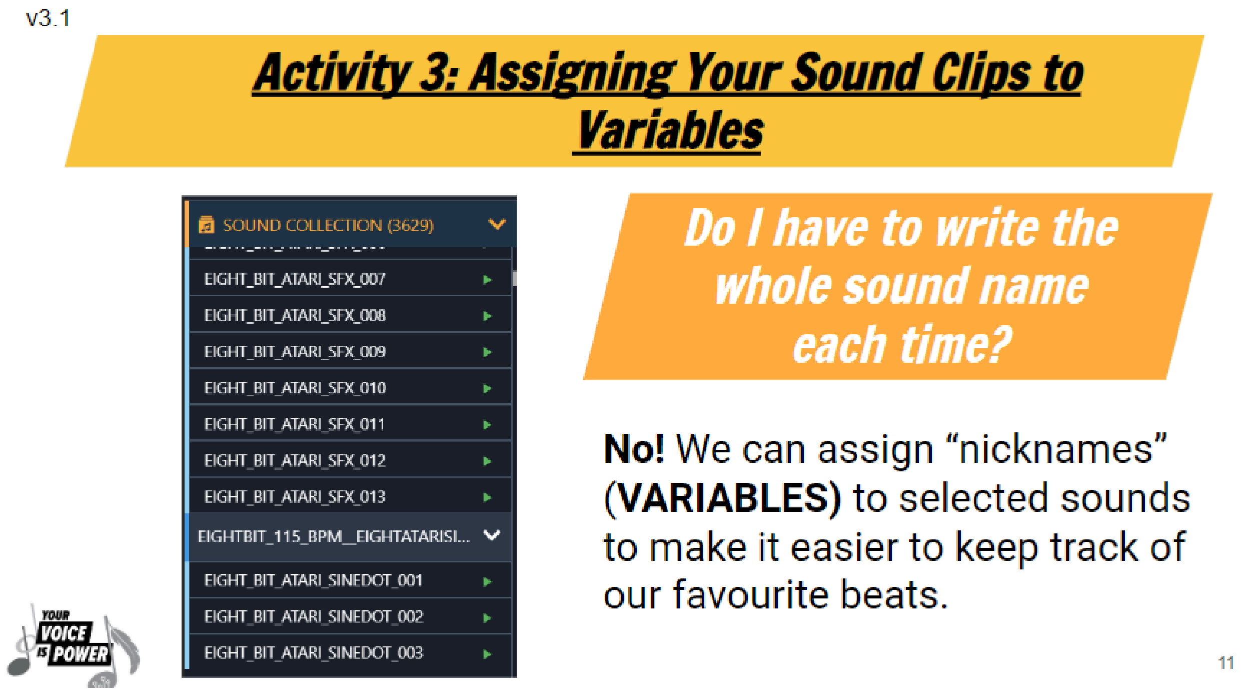
```
#drums
#vocals
#strings
```

Your finished code should look similar to the following, the code changes are highlighted:

```
# description:
from earsketch import *
setTempo(120)

# SOUNDBANK
#drums
SAMIAN_PEUP_BEAT_FULL
#vocals
TFLAMES_OC_VOX_BKUP_CHOR_1
```

Activity 3: Assigning Variables to Your Sound Clips (15 minutes)

Slide 11



v3.1

Activity 3: Assigning Your Sound Clips to Variables

SOUND COLLECTION (3629)

EIGHT_BIT_ATARI_SFX_007
EIGHT_BIT_ATARI_SFX_008
EIGHT_BIT_ATARI_SFX_009
EIGHT_BIT_ATARI_SFX_010
EIGHT_BIT_ATARI_SFX_011
EIGHT_BIT_ATARI_SFX_012
EIGHT_BIT_ATARI_SFX_013
EIGHTBIT_115_BPM__EIGHTATARISI...
EIGHT_BIT_ATARI_SINEDOT_001
EIGHT_BIT_ATARI_SINEDOT_002
EIGHT_BIT_ATARI_SINEDOT_003

Do I have to write the whole sound name each time?

No! We can assign “nicknames” (**VARIABLES**) to selected sounds to make it easier to keep track of our favourite beats.

11

Instructor Dialogue

*Some of the sounds have pretty long names. E.g., **ENTREP_BEAT_DRUMBEAT**.*

*We can use **variables** to give our sounds nicknames.*

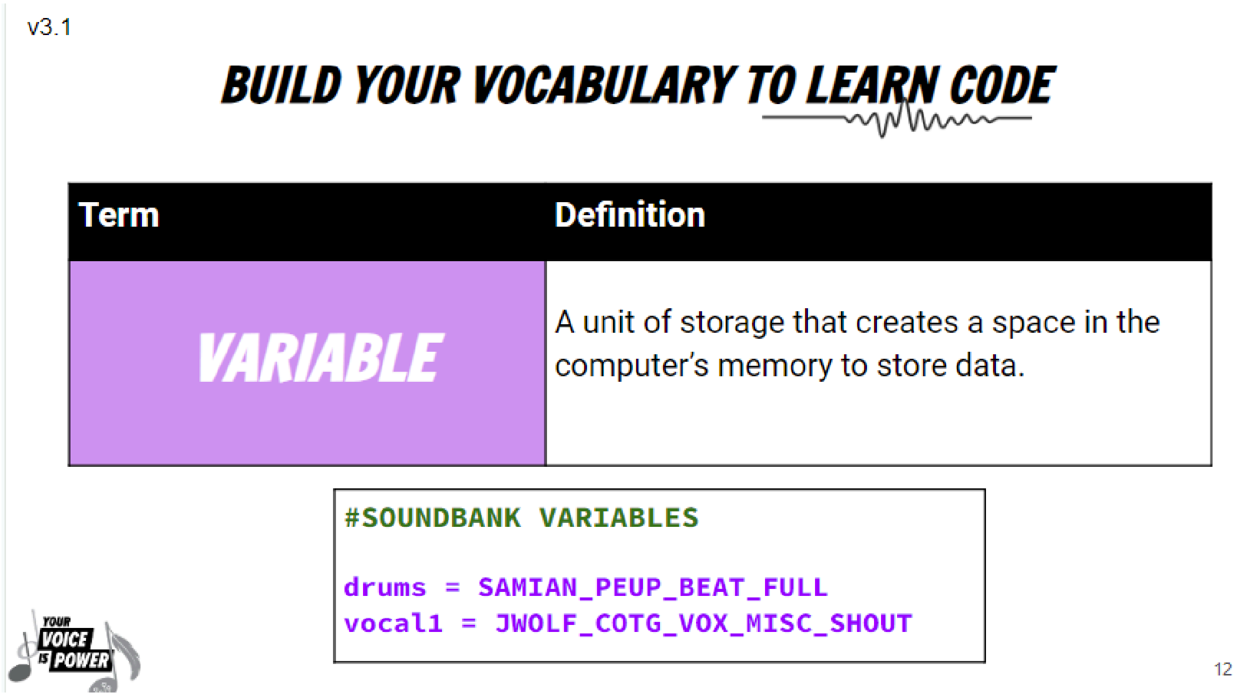
Example:

drum=ENTREP_BEAT_DRUMBEAT

*Now, when you want **ENTREP_BEAT_DRUMBEAT**, you can use its nickname or variable - **drum**.*

Look at your #SOUNDBANK in your code editor. It's time to assign these clips to some variables.

Slide 12



v3.1

BUILD YOUR VOCABULARY TO LEARN CODE

Term	Definition
VARIABLE	A unit of storage that creates a space in the computer's memory to store data.

```
#SOUNDBANK VARIABLES

drums = SAMIAN_PEUP_BEAT_FULL
vocal1 = JWOLF_COTG_VOX_MISC_SHOUT
```

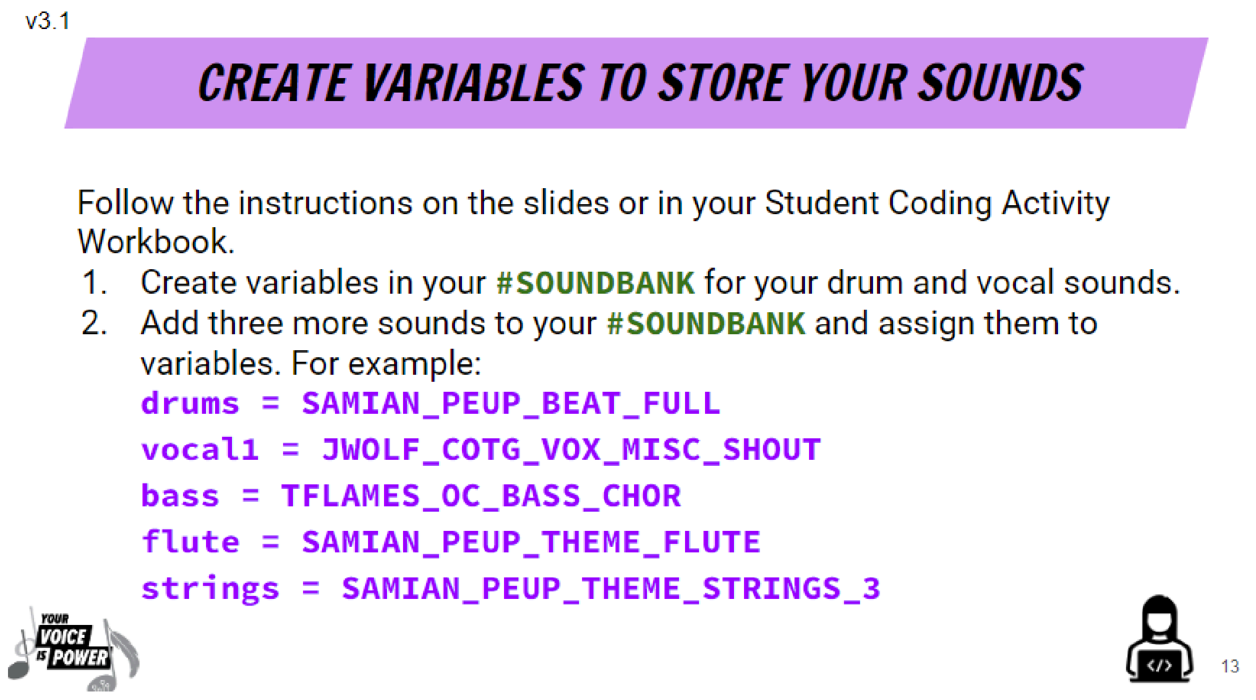
YOUR VOICE IS POWER

12

Instructor Dialogue

*A variable is like a container that stores data or information. You give this container a name to remind you what information you put inside it. You can put whatever kind of data you like inside! The variable can store numbers, or words and sentences called **strings**, or unchangeable data called **constants**. We will use variables to store our sound clip names.*

Slide 13



v3.1

CREATE VARIABLES TO STORE YOUR SOUNDS

Follow the instructions on the slides or in your Student Coding Activity Workbook.

1. Create variables in your **#SOUNDBANK** for your drum and vocal sounds.
2. Add three more sounds to your **#SOUNDBANK** and assign them to variables. For example:

```
drums = SAMIAN_PEUP_BEAT_FULL
vocal1 = JWOLF_COTG_VOX_MISC_SHOUT
bass = TFLAMES_OC_BASS_CHOR
flute = SAMIAN_PEUP_THEME_FLUTE
strings = SAMIAN_PEUP_THEME_STRINGS_3
```

YOUR VOICE IS POWER

13

Instructor Dialogue

Follow the instructions on your Student Coding Activity Workbook to use variables for your sounds and to add some additional sounds to your song

Follow along the next two slides and then try it yourself following the instructions in your Student Coding Activity Workbook: ***Module 4 - Activity 3: Assigning Variables to Your Sound Clips***

Instructor Cue

Leave the next slide with an example of the finished code displayed while the students complete the activity.

AFTER you do a demo or show them the slide explaining the steps, have the students begin the activity *Student Coding Activities Workbook* ***Module 4 - Activity 3: Assigning Variables to Your Sound Clips***

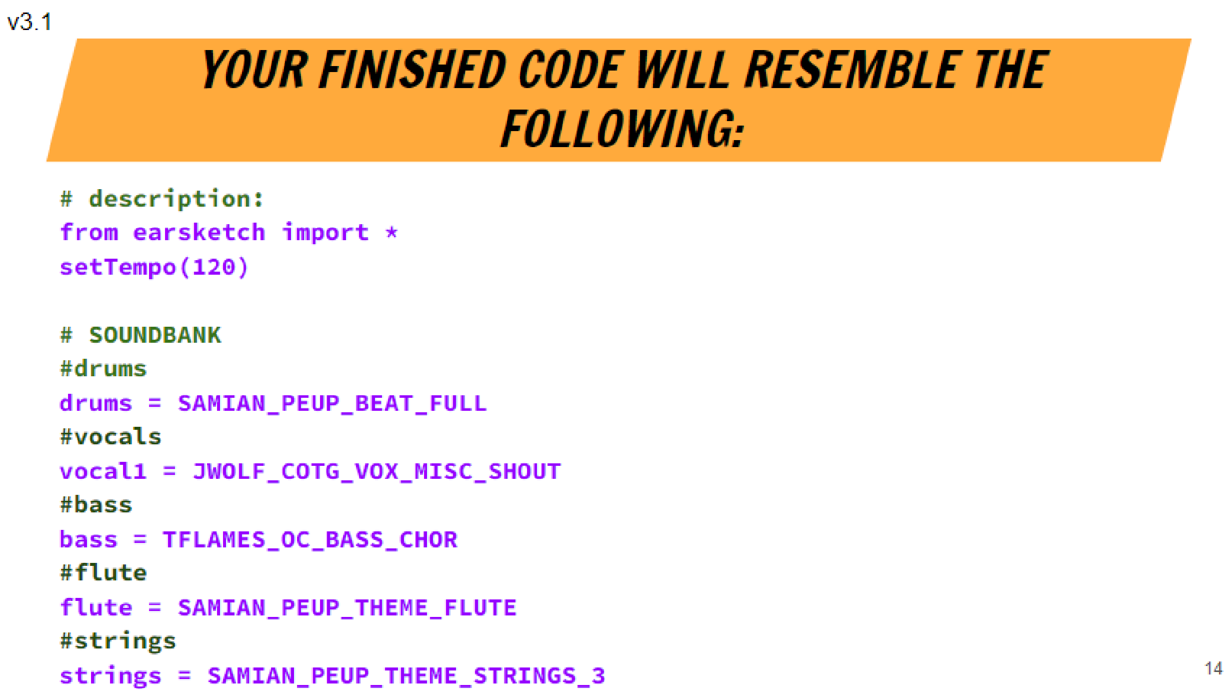
To complete a demo follow the example on the slide to assign a sound clip to a variable

OR

Show the students the slide to help them understand the steps they will perform to complete the activity.

The Student Coding Workbook contains detailed instructions including screenshots.

Slide 14



v3.1

YOUR FINISHED CODE WILL RESEMBLE THE FOLLOWING:

```
# description:
from earsketch import *
setTempo(120)

# SOUNDBANK
#drums
drums = SAMIAN_PEUP_BEAT_FULL
#vocals
vocal1 = JWOLF_COTG_VOX_MISC_SHOUT
#bass
bass = TFLAMES_OC_BASS_CHOR
#flute
flute = SAMIAN_PEUP_THEME_FLUTE
#strings
strings = SAMIAN_PEUP_THEME_STRINGS_3
```

14

Instructor Dialogue

Follow the instructions on your Student Coding Activity Workbook ***Module 4 - Activity 3: Assigning Variables to Your Sound Clips*** *to use variables for your sounds and to add some additional sounds to your song*

Instructor Cue

While this slide is displayed, have the students complete the activity Student Coding Activities Workbook **Module 4 - Activity 3: Assigning Variables to Your Sound Clips**

Student Coding Activity: Module 4 - Activity 3: Assigning Variables to Your Sound Clips

Students should follow the instructions in their Student Coding Activity Workbook

Related Video tutorials:

- [Adding your first sound with fitMedia()]
- [Using variables for sounds and adding song structure]

Continue with your code from the end of Module 4 - Activity 2 or create a new script with the following code

```
# description:
from earsketch import *
setTempo(120)

# SOUNDBANK
#drums
SAMIAN_PEUP_BEAT_FULL
#vocals
TFLAMES_OC_VOX_BKUP_CHOR_1
```

Assign the clips from your **#SOUNDBANK** to variables

1. Create a variable with a name that will help you remember the sound it will store and use the assignment operator '=' to assign your variable a value.

   Ex. **drums=ENTREP_BEAT_DRUMBEAT**
2. Assign each sound clip in your **#SOUNDBANK** to a variable to give it a nickname you can use in your code.
3. Add 3 more sound clips to your **#SOUNDBANK.**

   Think about the different instruments you heard in Samian's song. What different instruments could you add to your song?

4. Assign each of your new sound clips to a variable.

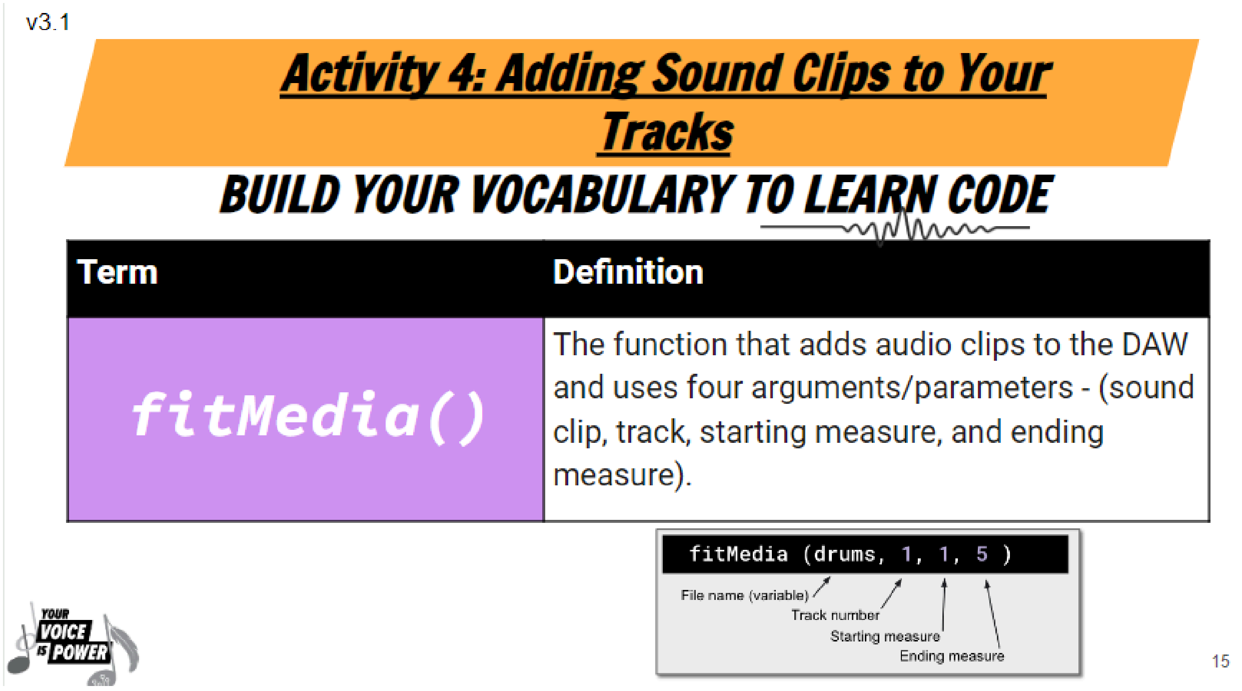
   Your finished code should look similar to the following, the variables are highlighted:

```
# description:
from earsketch import *
setTempo(120)

# SOUNDBANK
#drums
drums = SAMIAN_PEUP_BEAT_FULL
```

```
#vocals
vocal1 = TFLAMES_OC_VOX_BKUP_CHOR_1
#bass
bass = TFLAMES_OC_BASS_CHOR
#flute
flute = SAMIAN_PEUP_THEME_FLUTE
#strings
strings = SAMIAN_PEUP_THEME_STRINGS_3
```

Activity 4: Adding Sound Clips to your Tracks (15 minutes)

Slide 15


v3.1

Activity 4: Adding Sound Clips to Your Tracks

BUILD YOUR VOCABULARY TO LEARN CODE

Term	Definition
fitMedia()	The function that adds audio clips to the DAW and uses four arguments/parameters - (sound clip, track, starting measure, and ending measure).

```
fitMedia (drums, 1, 1, 5 )
```

File name (variable)
Track number
Starting measure
Ending measure
YOUR VOICE IS POWER
15

Instructor Dialogue

Now we will revisit fitMedia(). This time, you are going to be in charge of 'fitting the media' in where you want! We have our sounds and even assigned them cool nicknames (or variables). Now, you are ready to remix them in your song using the function fitMedia(). Remember, a function is a piece of code that performs a task. The fitMedia() function adds an audio file to a specified track at specific start and end times.

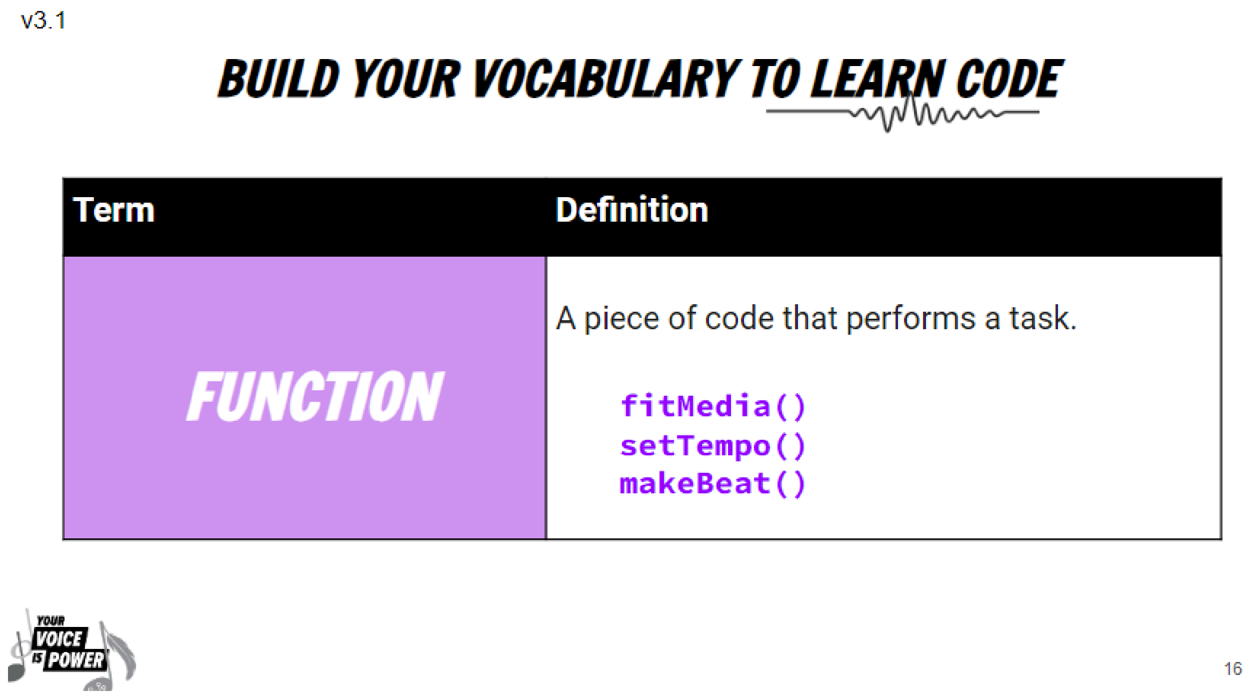
The fitMedia() function takes in four input parameters:

- ***sound**: The sound clip placed in the Digital Audio Workstation.*

- ***track***: *Track on which music is placed.*
- ***start***: *The measure at which the sound clip will start.*
- ***end***: *The measure at which the sound clip will end.*

Ex. fitMedia(drums,1,1,5)

Slide 16



v3.1

BUILD YOUR VOCABULARY TO LEARN CODE

Term	Definition

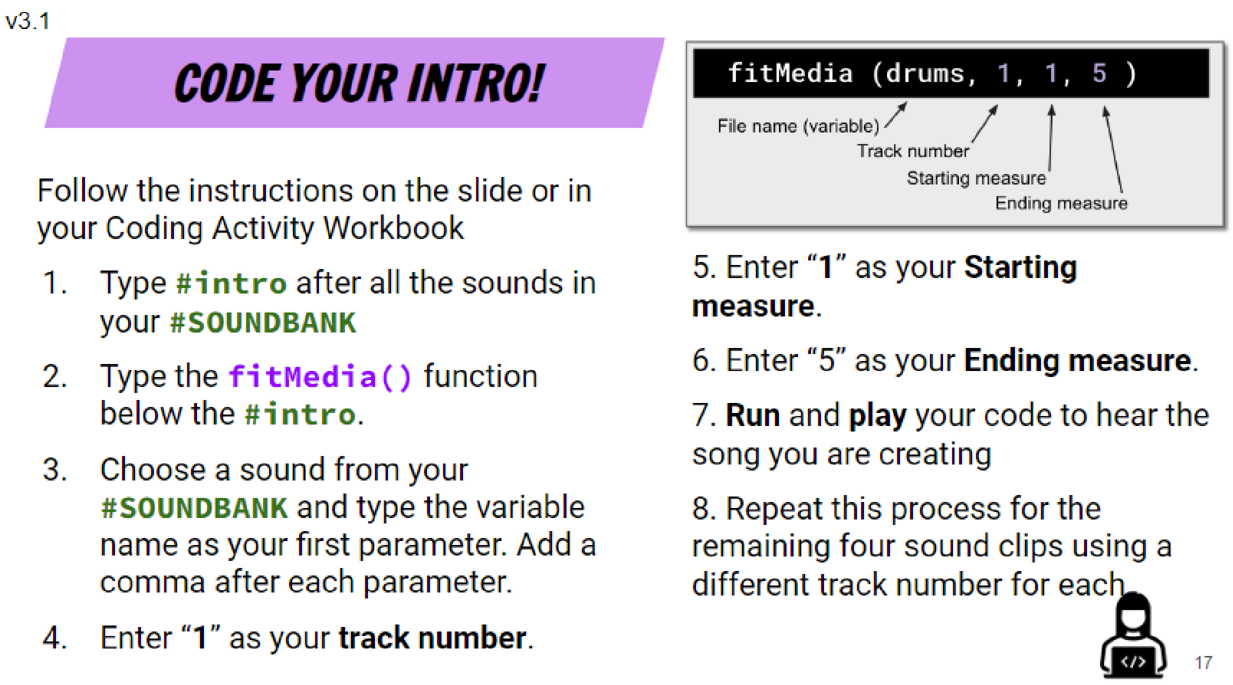
FUNCTION	A piece of code that performs a task.
	`fitMedia()`
	`setTempo()`
	`makeBeat()`

16

Instructor Dialogue

fitMedia is an example of a function. A function is a piece of code that performs a task—often consisting of many smaller actions.

Slide 17



v3.1

CODE YOUR INTRO!

Follow the instructions on the slide or in your Coding Activity Workbook

1. Type **#intro** after all the sounds in your **#SOUNDBANK**
2. Type the **fitMedia()** function below the **#intro**.
3. Choose a sound from your **#SOUNDBANK** and type the variable name as your first parameter. Add a comma after each parameter.
4. Enter “**1**” as your **track number**.

```
fitMedia (drums, 1, 1, 5 )
```

File name (variable)
Track number
Starting measure
Ending measure

5. Enter “**1**” as your **Starting measure**.
6. Enter “5” as your **Ending measure**.
7. **Run** and **play** your code to hear the song you are creating
8. Repeat this process for the remaining four sound clips using a different track number for each.

17

Instructor Dialogue

Practice using fitMedia to add sounds to your song by following the instructions in your Student Coding Workbook to add an intro to your song.

*Follow along and then try it yourself following the instructions in your Student Coding Activity Workbook: **Module 4 - Activity 4: Adding Sound Clips to your Tracks***

Instructor Cue

Leave the next slide with an example of the finished code displayed while the students complete the activity.

AFTER you do a demo or show them the slide demonstrating the steps, have the students begin the activity Student Coding Activities Workbook **Module 4 - Activity 4: Adding Sound Clips to your Tracks**

To complete a demo follow the steps on the slides to add a fitMedia() command to your script then run and play the song.

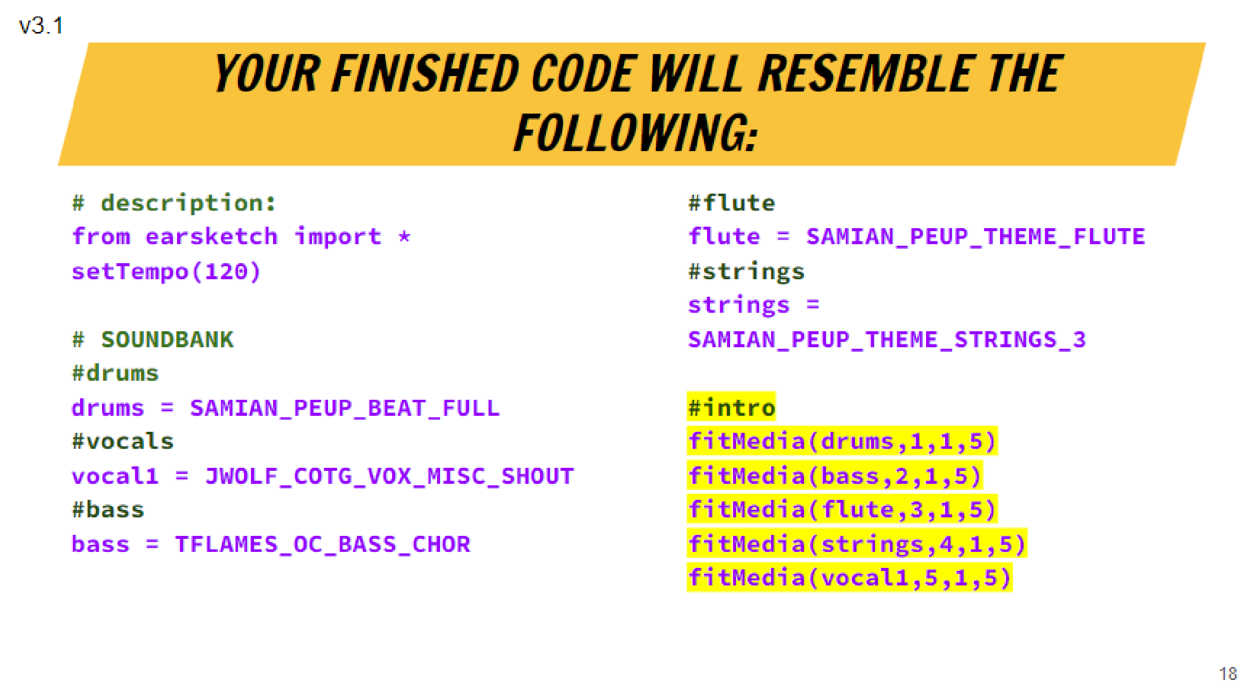
OR

Show the students the slide to help them understand the steps they will perform to complete the activity.

The Student Coding Workbook contains detailed instructions including screenshots.

Related Video tutorials:
- Adding your first sound with fitMedia()
- Using variables for sounds and adding song structure

Slide 18


v3.1

YOUR FINISHED CODE WILL RESEMBLE THE FOLLOWING:

```
# description:
from earsketch import *
setTempo(120)

# SOUNDBANK
#drums
drums = SAMIAN_PEUP_BEAT_FULL
#vocals
vocal1 = JWOLF_COTG_VOX_MISC_SHOUT
#bass
bass = TFLAMES_OC_BASS_CHOR
```

```
#flute
flute = SAMIAN_PEUP_THEME_FLUTE
#strings
strings =
SAMIAN_PEUP_THEME_STRINGS_3

#intro
fitMedia(drums,1,1,5)
fitMedia(bass,2,1,5)
fitMedia(flute,3,1,5)
fitMedia(strings,4,1,5)
fitMedia(vocal1,5,1,5)
```

18

Instructor Dialogue

Follow the instructions on your Student Coding Activity Workbook ***Module 4 - Activity 4: Adding Sound Clips to your Tracks*** *to add fitMedia() commands to your script*

Instructor Cue

While this slide is displayed, have the students complete the activity Student Coding Activities Workbook **Module 4 - Activity 4: Adding Sound Clips to your Tracks**

Student Coding Activity: Module 4 - Activity 4: Adding Sound Clips to your Tracks

Students should follow the instructions in their Student Coding Activity Workbook

Related Video tutorials:

- [Adding your first sound with fitMedia()]()
- [Using variables for sounds and adding song structure]()

Continue with your code from the end of Module 4 - Activity 3 or create a new script with the following code:

```
# description:
from earsketch import *
setTempo(120)

#  SOUNDBANK
#drums
drums = SAMIAN_PEUP_BEAT_FULL
#vocals
vocal1 = TFLAMES_OC_VOX_BKUP_CHOR_1
#bass
bass = TFLAMES_OC_BASS_CHOR
#flute
flute = SAMIAN_PEUP_THEME_FLUTE
#strings
strings = SAMIAN_PEUP_THEME_STRINGS_3
```

Use the function **`fitMedia()`** to add the new sounds as an introduction to your song.

1. Type **`#intro`** to your script in the code editor after all the sounds you added to your **#SOUNDBANK**.
    - **`#intro`** is a comment to help organize the structure of your song. Of course, this is all practice so you can learn the skills. You can change the sounds later. It may not end up being the intro of your final song!

2. Select enter to type on the line below **`#intro`**

3. On the next line, after **`#intro`**, type **`fitMedia()`**

- Notice that the code editor tries to help you by automatically filling in some of the parentheses. You can accept these autosuggestions or type the command yourself.

`fitMedia()` expects the following parameters

- **sound** (the sound to play)
- **track** (which track number to place the sound)
- **start** (the measure at which the sound will start)
- **end** (the measure at which the sound will end). The parameters must be inside the parentheses and must be separated by commas.

4. Specify the values in the table below as parameters for fitMedia(). The final result should look similar to the following but with your variable name instead of *'drums'*

```
fitMedia(drums,1,1,5)
```

Parameter	Value
sound	Choose a sound from your #SOUNDBANK and type the variable name
track	1
start	1
end	5

Your finished code should look similar to the following, the code changes are highlighted:

```
# description:
from earsketch import *
setTempo(120)

# SOUNDBANK
#drums
drums = SAMIAN_PEUP_BEAT_FULL
#vocals
vocal1 = TFLAMES_OC_VOX_BKUP_CHOR_1
#bass
bass = TFLAMES_OC_BASS_CHOR
#flute
```

```
flute = SAMIAN_PEUP_THEME_FLUTE
#strings
strings = SAMIAN_PEUP_THEME_STRINGS_3

#intro
fitMedia(drums, 1,1,5)
```

5. Select **Run** and **Play** to hear your selected sound play in your song.

6. Repeat this process for the remaining four sound clips. But specify a different track (2, 3, 4, etc.) for each sound. Make sure the end is a number bigger than start.
   - For now start all your sounds on measure 1 and end them on measure 5.
   - If you want to add sound clips directly from the sound browser, you can select the blue clipboard icon to paste the sound name directly into the `fitMedia()` function.
   - When you play the song , all the sounds play at the same time which may sound great or may sound terrible. You can experiment with different sounds or which sounds you choose to play to see what sounds good together. We will learn how to start and stop sounds at different parts of the song in Module 6.

Your finished code will look similar to the following:

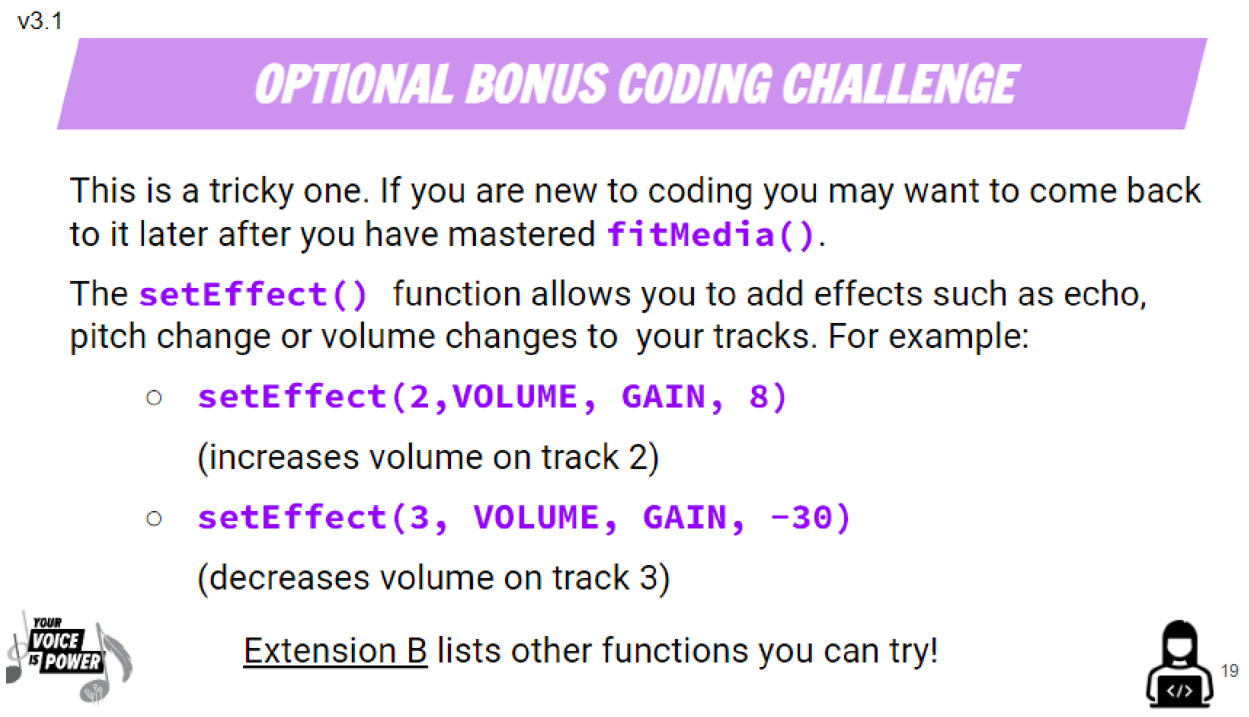
```
# description:
from earsketch import *
setTempo(120)

# SOUNDBANK
#drums
drums = SAMIAN_PEUP_BEAT_FULL
#vocals
vocal1 = JWOLF_COTG_VOX_MISC_SHOUT
#bass
bass = TFLAMES_OC_BASS_CHOR
#flute
flute = SAMIAN_PEUP_THEME_FLUTE
#strings
strings = SAMIAN_PEUP_THEME_STRINGS_3
```

```
#intro
fitMedia(drums, 1,1,5)
fitMedia(bass,2,1,5)
fitMedia(flute,3,1,5)
fitMedia(strings,4,1,5)
fitMedia(vocal1,5,1,5)
```

Instructor Cue

Extension A below shows students how to add sound clips they like to favourites.

Slide 19



v3.1

OPTIONAL BONUS CODING CHALLENGE

This is a tricky one. If you are new to coding you may want to come back to it later after you have mastered `fitMedia()`.

The `setEffect()` function allows you to add effects such as echo, pitch change or volume changes to your tracks. For example:

- `setEffect(2,VOLUME, GAIN, 8)`

  (increases volume on track 2)
- `setEffect(3, VOLUME, GAIN, -30)`

  (decreases volume on track 3)

Extension B lists other functions you can try!

19

Instructor Dialogue

You can make a great song with just setTempo and fitMedia, but for those of you who get the hang of fitMedia quickly and want to go a little further, you might want to check out setEffect. It can be used to change the volume in case you want a particular sound to be quieter or louder, or you can use it to add special effects.

If you want to try it, follow the instructions in your Student Coding Activity Workbook: ***Module 4 - Optional Bonus Challenge: Kick it up a notch!***

Instructor Cue

Have the students begin the activity Student Coding Activities Workbook **Module 4 - Optional Bonus Challenge: Kick it up a notch!**

There are many other functions available for students to use when they are ready.

Change the volume on one of the tracks. This is one of the most common questions students ask when they start building their song, they may find that they can't hear the vocals because an instrument is too loud, so they can use this to increase or decrease the volume of a track.

Extension B below lists other functions for students to try!

Student Coding Activity: Module 4 - Optional Bonus Challenge: Kick it up a notch!

Students should follow the instructions in their Student Coding Activity Workbook

This is a tricky one, so if you are new to coding, you may want to come back to it later when you have mastered `fitMedia()`.

`setEffect()` is a function that allows you to add effects to sounds on different tracks to add echos, delays, change pitch, and change volume.

For example, as you work on your song, you might notice that the keyboards are too loud and you can't hear the vocals. You can use `setEffect()` to increase or decrease the volume of all the sounds on a particular track.

```
setEffect(track, type, parameter, value)
```

Parameter	Description
track	The number of the track on which to apply the effect
type	Which effect you want to do e.g. **VOLUME** (to change volume) **PITCHSHIFT** (to change pitch)
parameter	What part of the effect are you changing, e.g. **GAIN** to change volume level,

	PITCHSHIFT_SHIFT to specify how much to adjust pitch
value	The value for the parameter. For exmaple **GAIN** can range from -60.0 to 12.0, **PITCHSHIFT_SHIFT** can range from -12.0 to 12.0

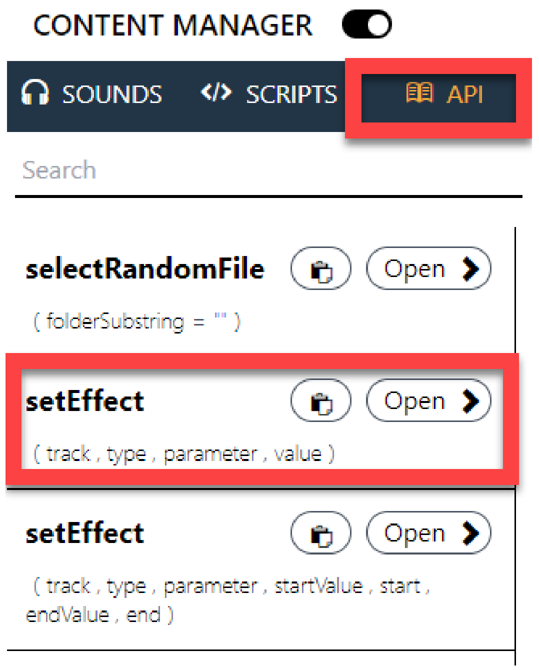
`setEffect(2,VOLUME, GAIN, 8)` increases the volume on track 2
`setEffect(3, VOLUME, GAIN, -30)` decreases the volume on track 3

If you applied these examples to your script the completed code would resemble the following. The code changes are highlighted

```
# description:
from earsketch import *
setTempo(120)

# SOUNDBANK
drums = SAMIAN_PEUP_BEAT_FULL
#vocals
vocal1 = JWOLF_COTG_VOX_MISC_SHOUT
#bass
bass = TFLAMES_OC_BASS_CHOR
#flute
flute = SAMIAN_PEUP_THEME_FLUTE
#strings
strings = SAMIAN_PEUP_THEME_STRINGS_3

#intro
fitMedia(drums, 1,1,5)
fitMedia(bass,2,1,5)
fitMedia(flute,3,1,5)
fitMedia(strings,4,1,5)
fitMedia(vocal1,5,1,5)

# increase volume on track 2
setEffect(2,VOLUME,GAIN,8)
# decrease volume on track 3
```

`setEffect(3,VOLUME,GAIN,-30)`

This is NOT a comprehensive explanation. To learn more, go to **CONTENT MANAGER | API** and look up `setEffect()`.

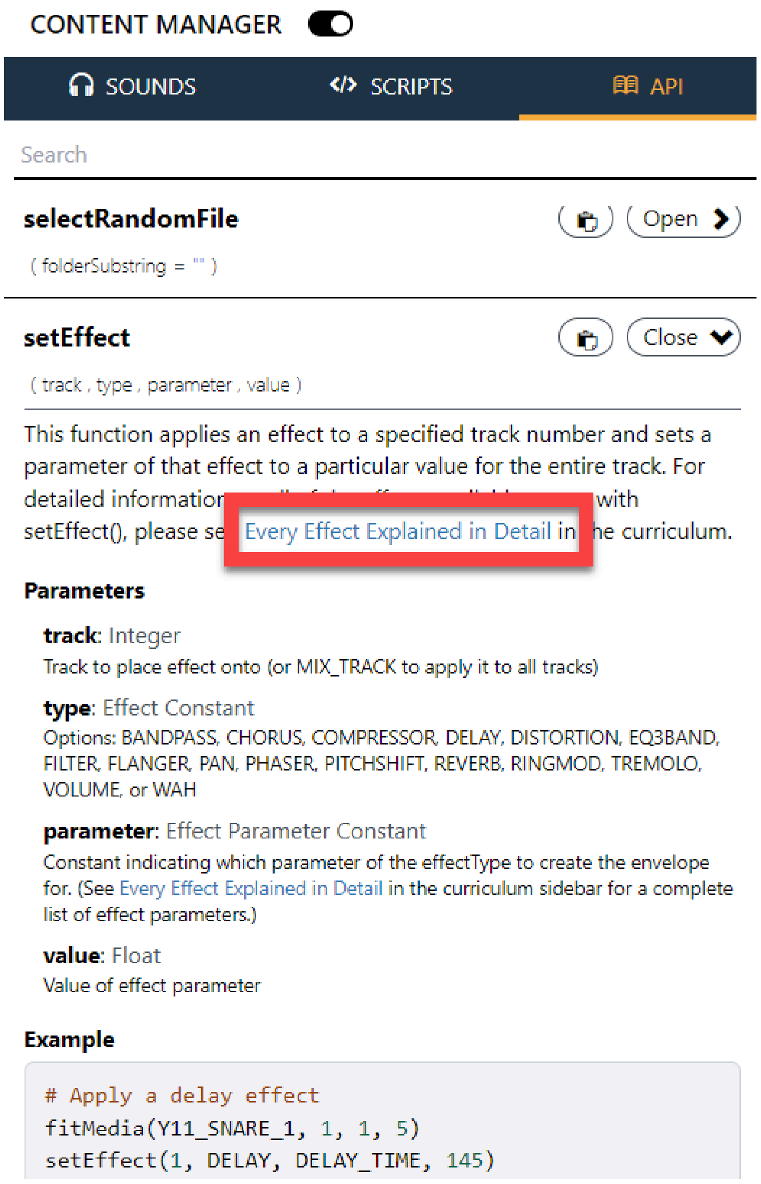


selectRandomFile (folderSubstring = "")

setEffect (track , type , parameter , value)

setEffect (track , type , parameter , startValue , start , endValue , end)

In the API documentation there is a link to [Every Effect Explained in Detail](). These two resources will help you figure out how to use effects. Coders rely heavily on API (Application Programming interface) documentation to figure out how to do new things. An API is a function someone else created that has an interface which allows other programmers to use it.



selectRandomFile

(folderSubstring = "")

setEffect

(track , type , parameter , value)

This function applies an effect to a specified track number and sets a parameter of that effect to a particular value for the entire track. For detailed informatior with setEffect(), please se Every Effect Explained in Detail in he curriculum.

Parameters

track: Integer
Track to place effect onto (or MIX_TRACK to apply it to all tracks)

type: Effect Constant
Options: BANDPASS, CHORUS, COMPRESSOR, DELAY, DISTORTION, EQ3BAND, FILTER, FLANGER, PAN, PHASER, PITCHSHIFT, REVERB, RINGMOD, TREMOLO, VOLUME, or WAH

parameter: Effect Parameter Constant
Constant indicating which parameter of the effectType to create the envelope for. (See Every Effect Explained in Detail in the curriculum sidebar for a complete list of effect parameters.)

value: Float
Value of effect parameter

Example

```
# Apply a delay effect
fitMedia(Y11_SNARE_1, 1, 1, 5)
setEffect(1, DELAY, DELAY_TIME, 145)
```

There are more functions you can explore listed in Module 4 - Extension B.

Activity 5: Record and Add a Sound to EarSketch (10 minutes + homework)

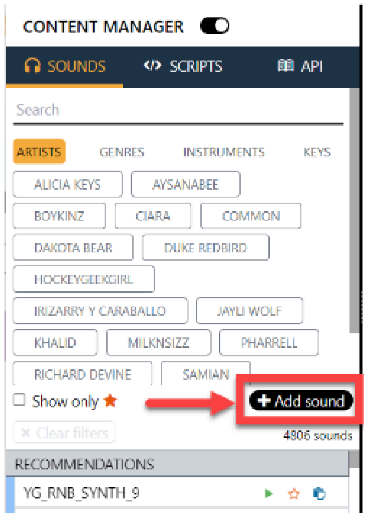
Slide 20

v3.1

Activity 5: Record and Add a Sound to EarSketch

Follow the instructions on the slides or in your Student Coding Activity Workbook.

1. Continue with your **Your Voice is Power [Your Initials]** script.
2. Select the **Add Sound** button in the Sound browser.



20

Instructor Dialogue

You may wish to add your own sound (either lyrics you say or sing) or something you've recorded or get from another copyright-free source.

*In this activity, you will record and add your own voice and learn how to add it to your song. Follow along the next four slides and then try it yourself following the instructions in your Student Coding Activity Workbook: **Module 4 - Activity 5: Record and Add a Sound to Earsketch***

If you want there is an additional coding challenge to search a free library of sounds to add additional sounds to your song.

*Let's start by selecting the **Add sound** button.*

Instructor Cue

AFTER you do a demo or show them the slides demonstrating the steps, have the students begin the activity *Student Coding Activities Workbook* ***Module 4 - Activity 2: Creating Your Soundbank***

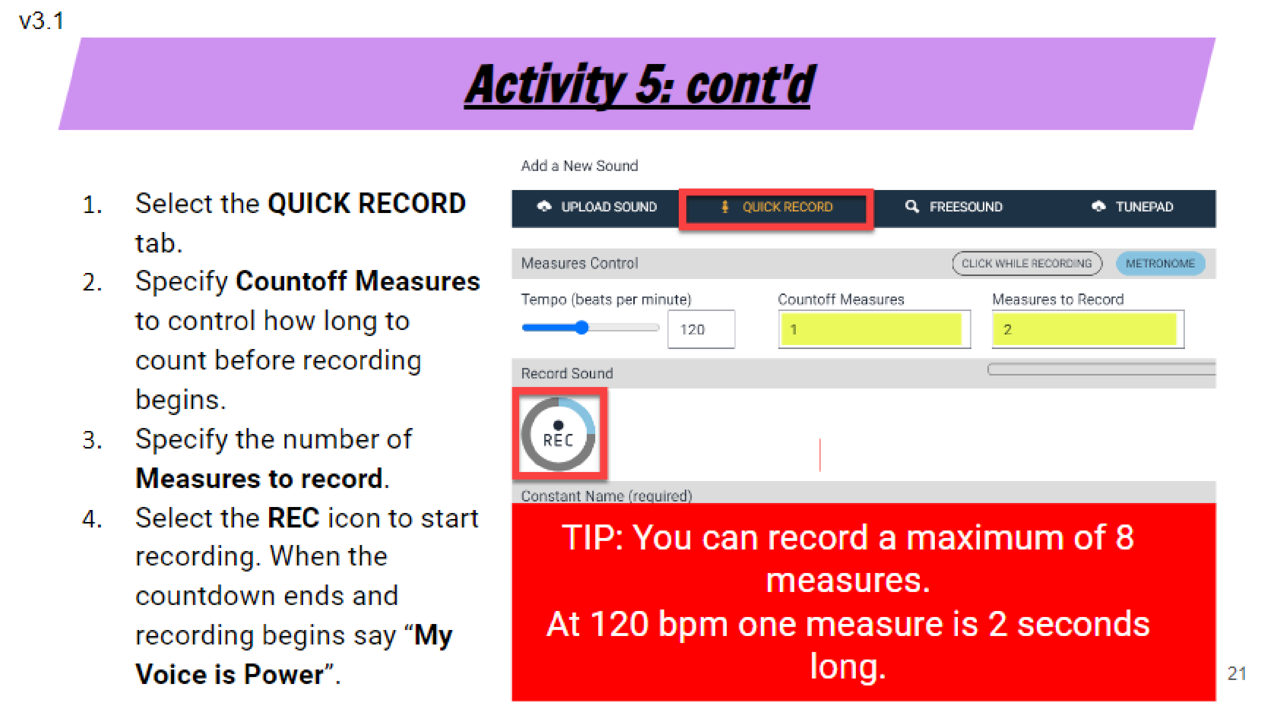
To complete a demo follow the steps on the slides to record a sound.

OR

Show the students the next four slides to help them understand the steps they will perform to complete the activity.

The Student Coding Workbook contains detailed instructions including screenshots.

Slide 21



v3.1

Activity 5: cont'd

1. Select the **QUICK RECORD** tab.
2. Specify **Countoff Measures** to control how long to count before recording begins.
3. Specify the number of **Measures to record**.
4. Select the **REC** icon to start recording. When the countdown ends and recording begins say "**My Voice is Power**".

Tempo (beats per minute): 120

Countoff Measures: 1

Measures to Record: 2

TIP: You can record a maximum of 8 measures.
At 120 bpm one measure is 2 seconds long.

21

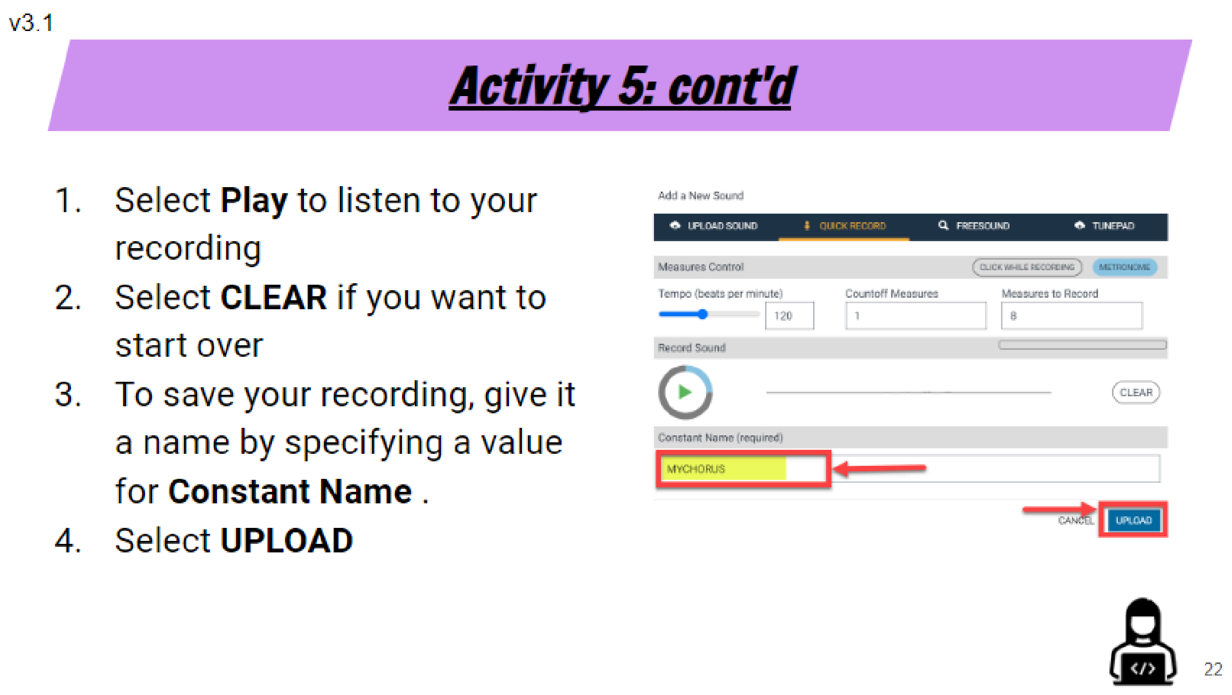
Instructor Dialogue

Next select ***QUICK RECORD*** *to record your own sound.*

You can change the duration of the countdown before recording starts using ***Countoff measures***

You can change how long to record using ***Measures to record****. You can record a maximum of 8 measures, at 120 BPM that would be 16 seconds. So if you are recording a verse or chorus you will probably have to break it up and record 2-4 lines at a time.*

Slide 22



v3.1

Activity 5: cont'd

1. Select **Play** to listen to your recording
2. Select **CLEAR** if you want to start over
3. To save your recording, give it a name by specifying a value for **Constant Name** .
4. Select **UPLOAD**

Add a New Sound

Measures Control

Tempo (beats per minute): 120

Countoff Measures: 1

Measures to Record: 8

Record Sound

Constant Name (required): MYCHORUS

22

Instructor Dialogue

*After you finish recording you can listen to your recording using **Play***

*If you want to re-record select **CLEAR***

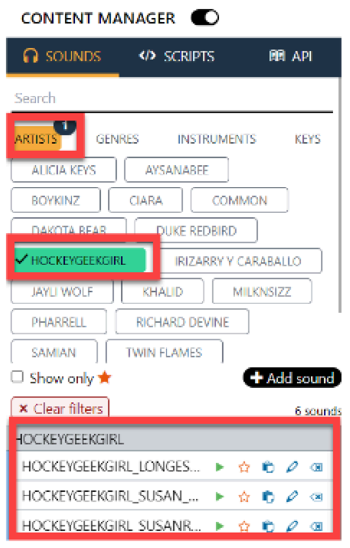
*Once you have a recording you like save it by giving it a name and selecting **UPLOAD***

Slide 23

v3.1

Activity 5: cont'd

1. Clear any filters you have in the sound library.
2. Search **SOUNDS** by **ARTISTS**.
3. Your login name will be listed as an artist (e.g. HOCKEYGEEKGIRL)
4. Select your Name as an artist and you will see your sound clip listed.



23

Instructor Dialogue

Now you can add your sound to your script!

To find your sound in the CONTENT MANAGER, clear all your filters and search for SOUNDS by ARTISTS. Your username will be listed as an artist. Select your name as the artist to search for and you should see your sound listed.

Instructor Cue

If anyone has trouble finding their sounds, check to make sure they haven't got any other filters applied by selecting Clear Filters and then starting the search again

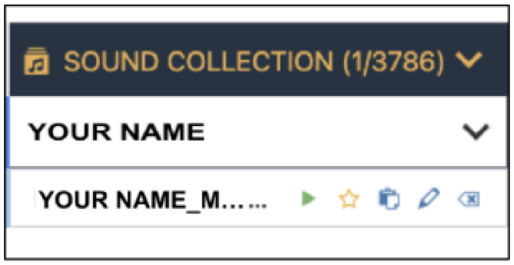
---

Slide 24

v3.1

Activity 5: cont'd

1. Add your sound clip to the **#SOUNDBANK**
   a. Assign it to a variable
   b. Replace one of the tracks in your **#intro** with your sound by changing the variable name in one of the **fitMedia()** functions
2. Select **Run** and **Play**



24

Instructor Dialogue

Now you can add your sound to your script!

Add it to your soundbank and use it in one of your fitMedia statements then Run and Play your song to hear it in the song!

*Now it's your turn, follow the instructions in your Student Coding Workbook **Module 4 - Activity 5: Record and Add a Sound to Earsketch***

Instructor Cue

Have students complete the activity **Module 4 - Activity 5: Record and Add a Sound to Earsketch** In their Student Coding Workbook.

Student Coding Activity: Module 4 - Activity 5: Record and Add a Sound to Earsketch

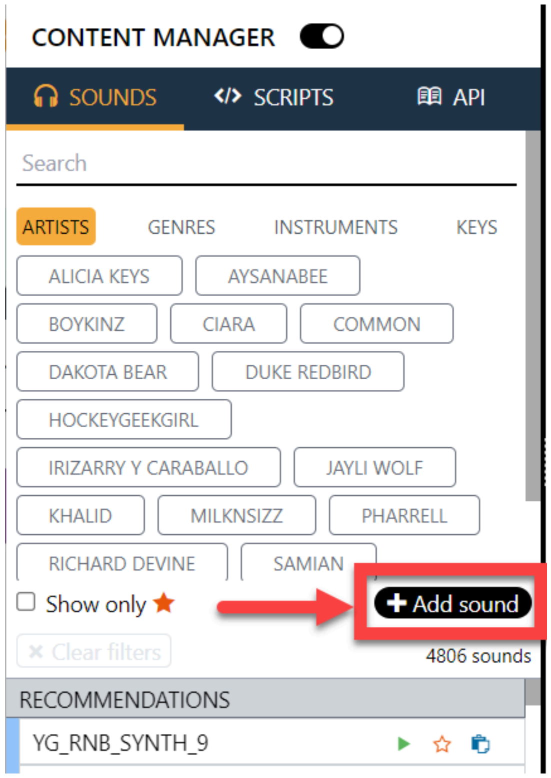
Students should follow the instructions in their Student Coding Activity Workbook

You may want to add your own sounds to the sound browser so you can include them in your song. You can say or sing your own lyrics, upload something you've recorded, or upload clips from another copyright-free source.

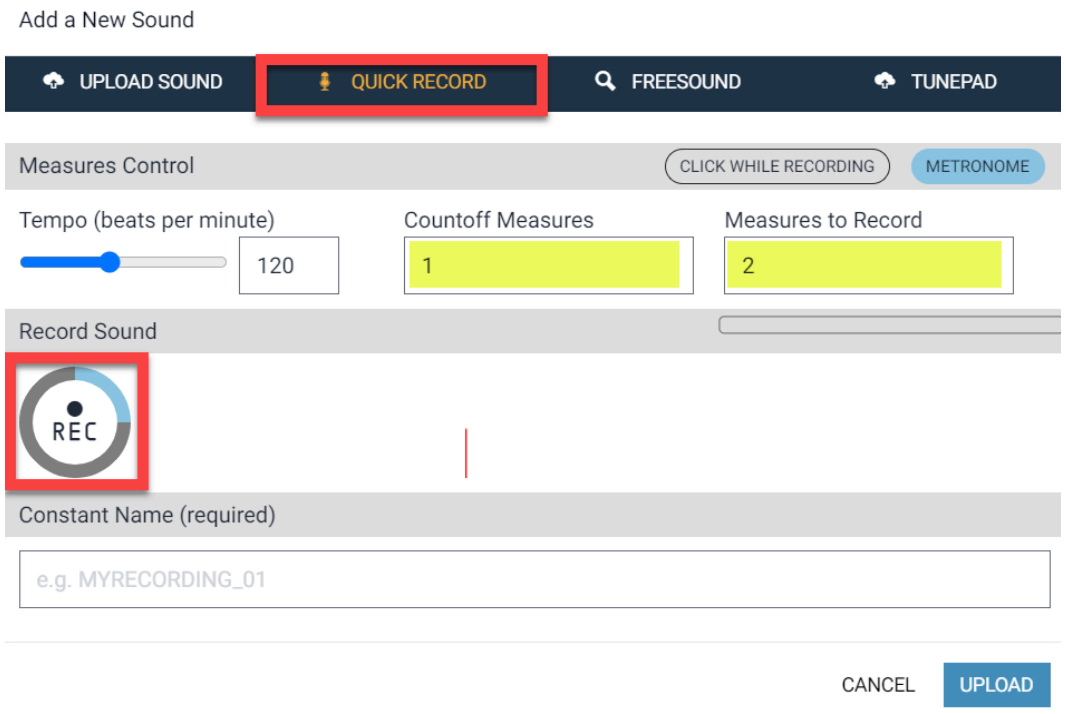
To record your own sound.

1. Open your **Your Voice is Power [Your Initials]** script.

2. Select the **+Add sound** button in the Sound browser.



3. Select the **QUICK RECORD** tab
   Countoff Measures determines how many measures will count down after you select **REC** before recording begins.
   Measures to Record determines how many measures are recorded. You can record a maximum of 8 measures at a time. With a tempo of 120 BPM, one measure is 2 seconds long. So you can record a 16 second clip.


Add a New Sound

Measures Control

Tempo (beats per minute): 120

Countoff Measures: 1

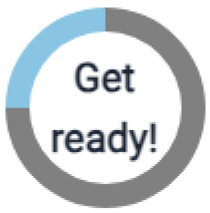
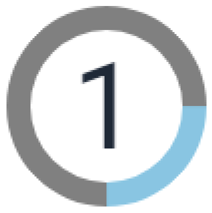
Measures to Record: 2

Record Sound

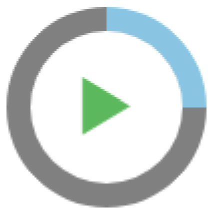
Constant Name (required)

e.g. MYRECORDING_01

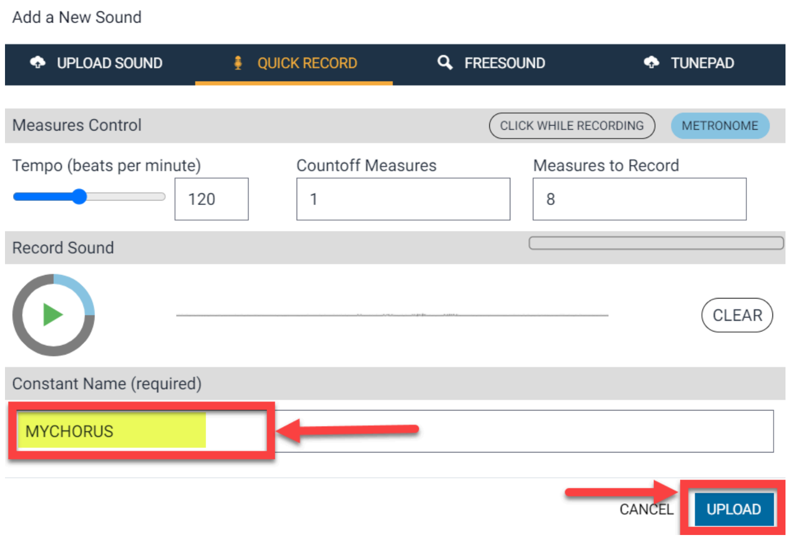
4. When you are ready to record, select the **REC** icon to start the countdown, the icon will display a Get Ready message during the countdown, and will display the number 1 when it is recording. When it starts recording say “My Voice is Power.”

Counting down to start of recording, length of countdown is controlled by changing **Countoff measures**	
Recording has begun, the number displayed counts the number of measures into the recording. Increase the number of measures recorded by changing **Measures to record**.	

5. Preview your recorded sound using the **Play** button. If you want to re-record. Select **Clear**.

Play your sound clip.	
Clear your sound so you can re-record.	

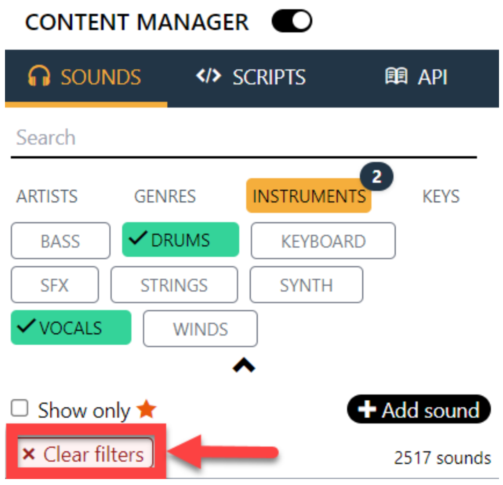
6. Once you are satisfied with your recording, type a name for your recording in the field **Constant Name** and select **UPLOAD.**


Add a New Sound

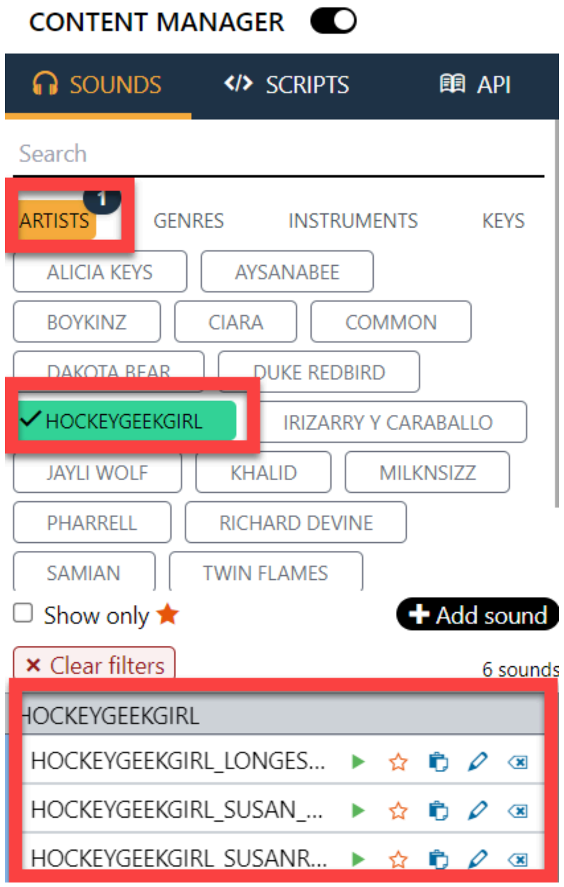
Tempo (beats per minute): 120
Countoff Measures: 1
Measures to Record: 8

Constant Name (required): MYCHORUS

7. Add your recorded sound to your song
   - Go to the sound browser. If you have any filters applied select **Clear Filters**



- List sounds by **ARTISTS**. Select your username from the list of artists. Your sound clip will be listed in the search results.



- Add your sound clip to the **`#SOUNDBANK`**
  - Assign the sound clip to a variable.
  - Replace one of the tracks you have in your **`#intro`** with your new sound by changing the variable name in one of the **`fitMedia()`** functions.

8. Select **Run** and **Play** to listen to your song with the sound you added.

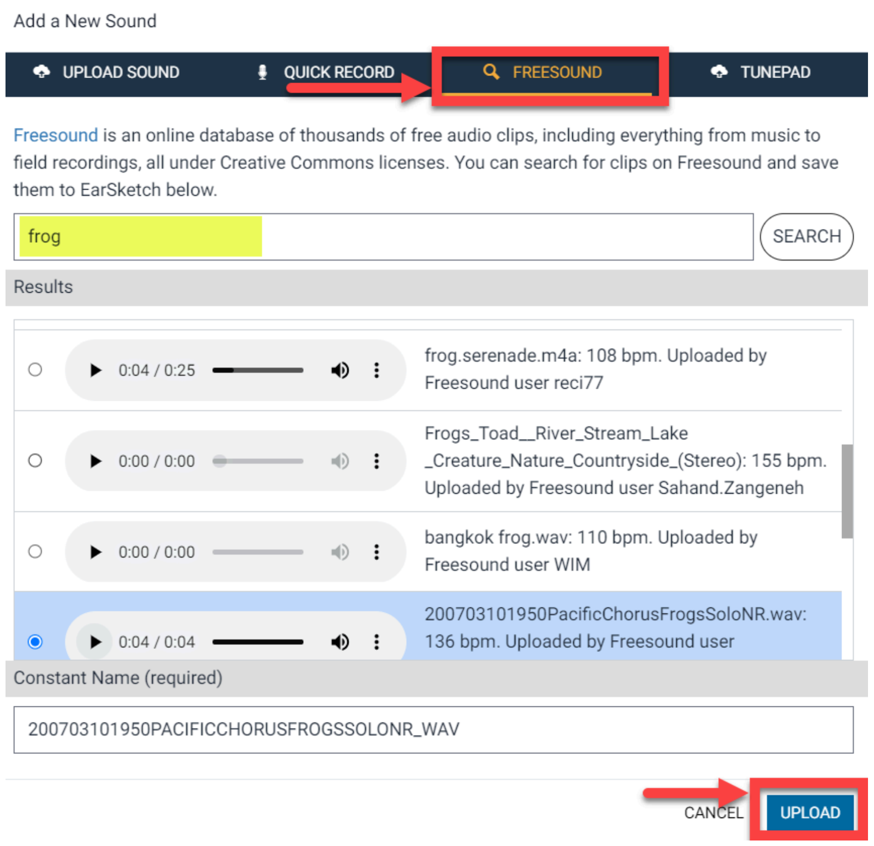
Your finished code would look similar to the following, the code changes are highlighted

```
# description:
from earsketch import *
setTempo(120)

# SOUNDBANK
drums = SAMIAN_PEUP_BEAT_FULL
#vocals
vocal1 = JWOLF_COTG_VOX_MISC_SHOUT
MyVoiceIsPower = HOCKEYGEEKGIRL_MYVOICEISPOWER
#bass
bass = TFLAMES_OC_BASS_CHOR
#flute
flute = SAMIAN_PEUP_THEME_FLUTE
#strings
strings = SAMIAN_PEUP_THEME_STRINGS_3

#intro
fitMedia(drums, 1,1,5)
fitMedia(bass,2,1,5)
fitMedia(flute,3,1,5)
fitMedia(strings,4,1,5)
fitMedia(MyVoiceIsPower,5,1,5)
```

Optional Bonus Challenge: Module 4 - Add one of the copyright free sounds in FREESOUND.

When you select **+Add Sound**, one of the tabs listed is **FREESOUND**. **FREESOUND** gives you the ability to search and add sounds from an online database of thousands of free audio clips you can use in your songs because they are available under the creative commons license. We have not provided detailed instructions, but the screenshot below should help. Sometimes optional features are not well documented and programmers have to experiment to figure out how it works. Can you figure it out?

TIP: After you upload the sound, you can locate it by searching for **GENRES | FREESOUND**


Add a New Sound

Freesound is an online database of thousands of free audio clips, including everything from music to field recordings, all under Creative Commons licenses. You can search for clips on Freesound and save them to EarSketch below.

frog

frog.serenade.m4a: 108 bpm. Uploaded by Freesound user reci77

Frogs_Toad__River_Stream_Lake_Creature_Nature_Countryside_(Stereo): 155 bpm. Uploaded by Freesound user Sahand.Zangeneh

bangkok frog.wav: 110 bpm. Uploaded by Freesound user WIM

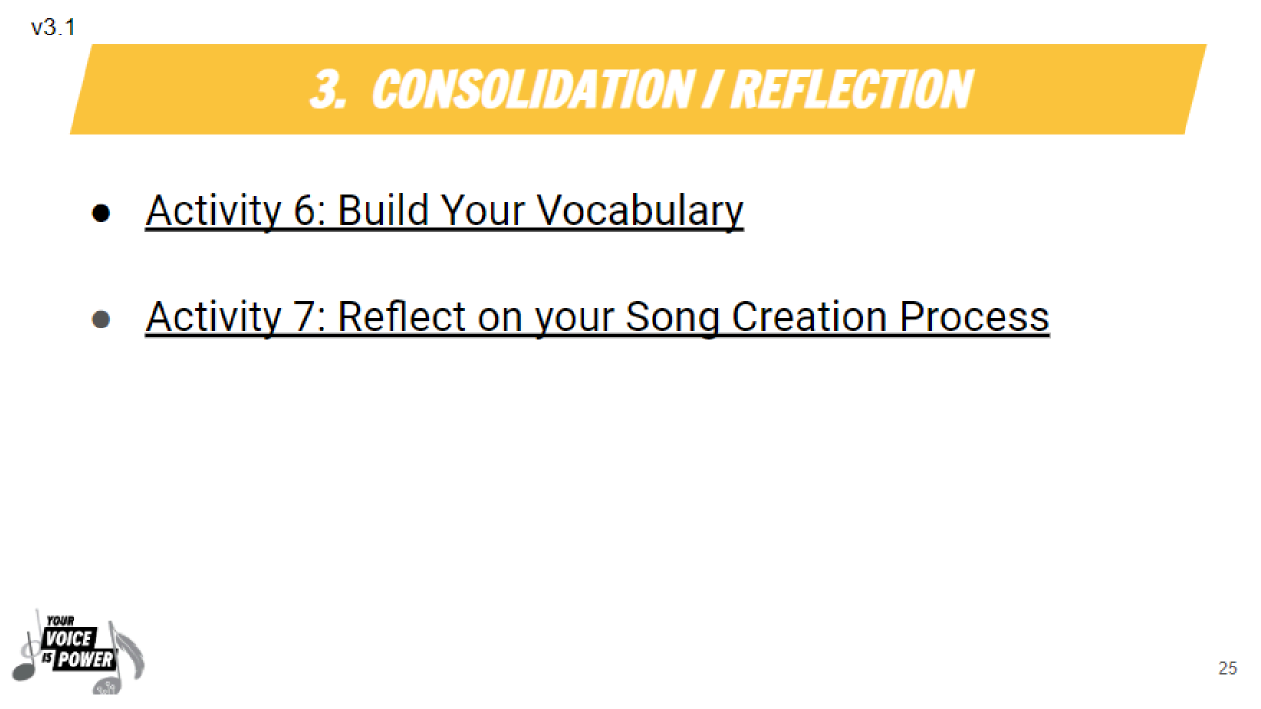
200703101950PacificChorusFrogsSoloNR.wav: 136 bpm. Uploaded by Freesound user

Constant Name (required)

200703101950PACIFICCHORUSFROGSSOLONR_WAV

3. CONSOLIDATION/REFLECTION

Slide 25



v3.1

3. CONSOLIDATION / REFLECTION

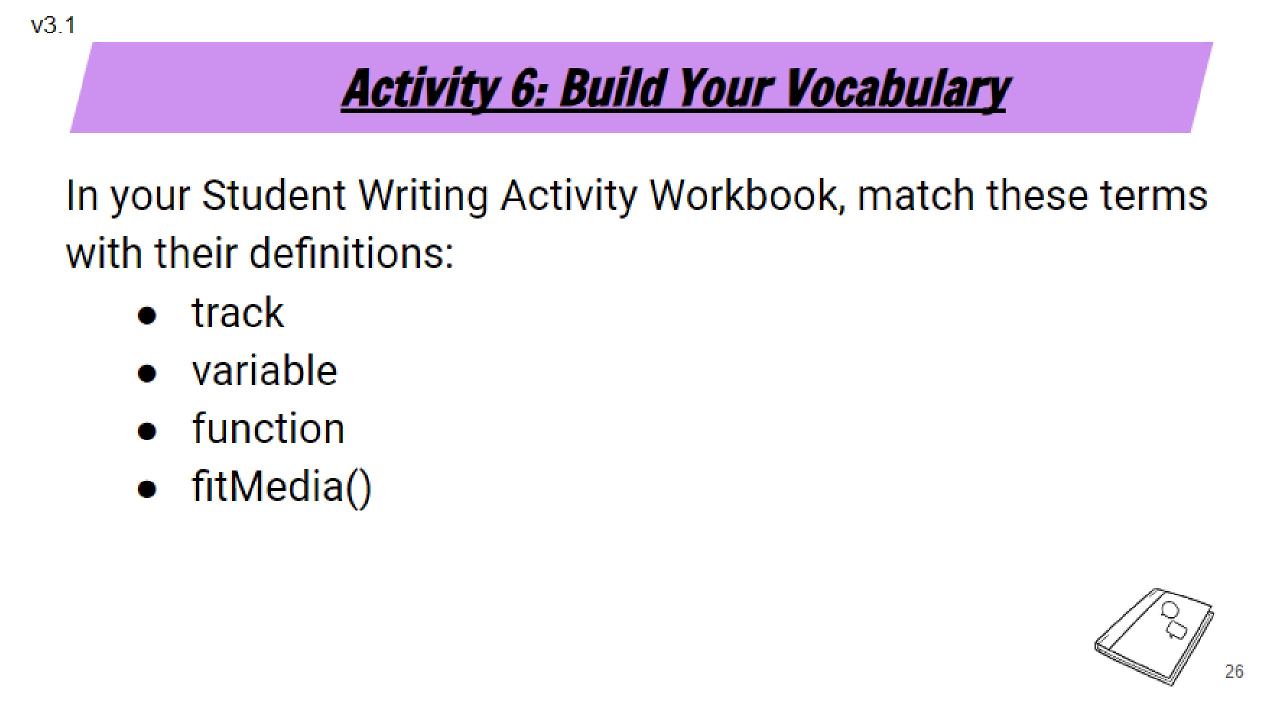
- Activity 6: Build Your Vocabulary
- Activity 7: Reflect on your Song Creation Process

25

Activity 6: Build Your Vocabulary (5 minutes)

Slide 26

In



v3.1

Activity 6: Build Your Vocabulary

In your Student Writing Activity Workbook, match these terms with their definitions:

- track
- variable
- function
- fitMedia()

26

Instructor Dialogue

Match the following terms with their definitions in your Student Writing Activity Workbook ***Module 4 - Activity 6: Build your Vocabulary***

Instructor Cue

Have the student complete **Module 4 - Activity 6: Build your Vocabulary** in their Student Writing Activity Workbook

---

Student Writing Activity: Module 4 - Activity 6: Build Your Vocabulary

Students should follow the instructions in their Student Writing Activity Workbook

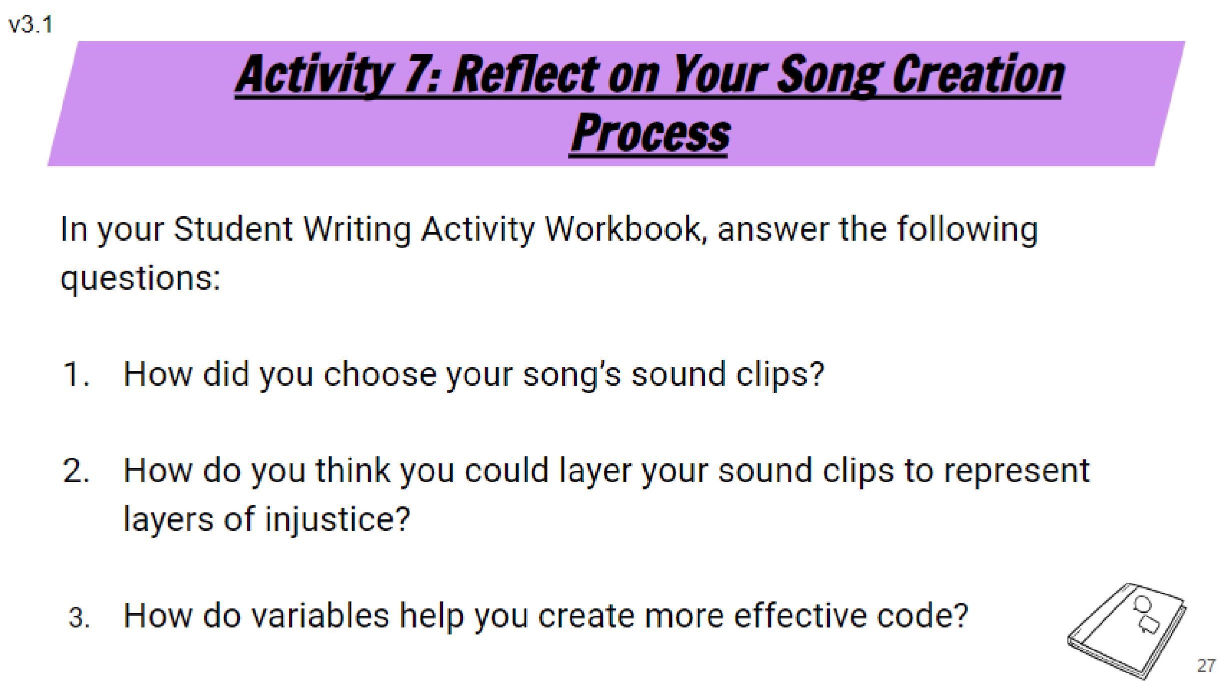
Write the following terms beside the correct definitions:

- function
- fitMedia()
- track
- variable

Term	Definition
track	A part of a song that is recorded separately as a musical clip and added to a piece of music. In a DAW, these are arranged in rows and labeled with numbers.
variable	A unit of storage that creates a space in the computer's memory to store data.
function	A piece of code that performs a task.
fitMedia()	The function that adds audio clips to a track and uses four arguments/parameters - (sound clip, track, starting measure, ending measure).

Activity 7: Reflect on Your Song Creation Process (10 minutes)

Slide 27



v3.1

Activity 7: Reflect on Your Song Creation Process

In your Student Writing Activity Workbook, answer the following questions:

1. How did you choose your song's sound clips?
2. How do you think you could layer your sound clips to represent layers of injustice?
3. How do variables help you create more effective code?

27

Instructor Dialogue

Take some time to reflect on what we learned by answering these questions in your Student Writing Activity Workbook ***Module 4 - Activity 7: Reflect on your song creation process***

Instructor Cue

Have the student complete **Module 4 - Activity 7: Reflect on your song creation process** in their Student Writing Activity Workbook

---

Student Writing Activity: Module 4 - Activity 7: Reflect on your song creation process

Students should follow the instructions in their Student Writing Activity Workbook

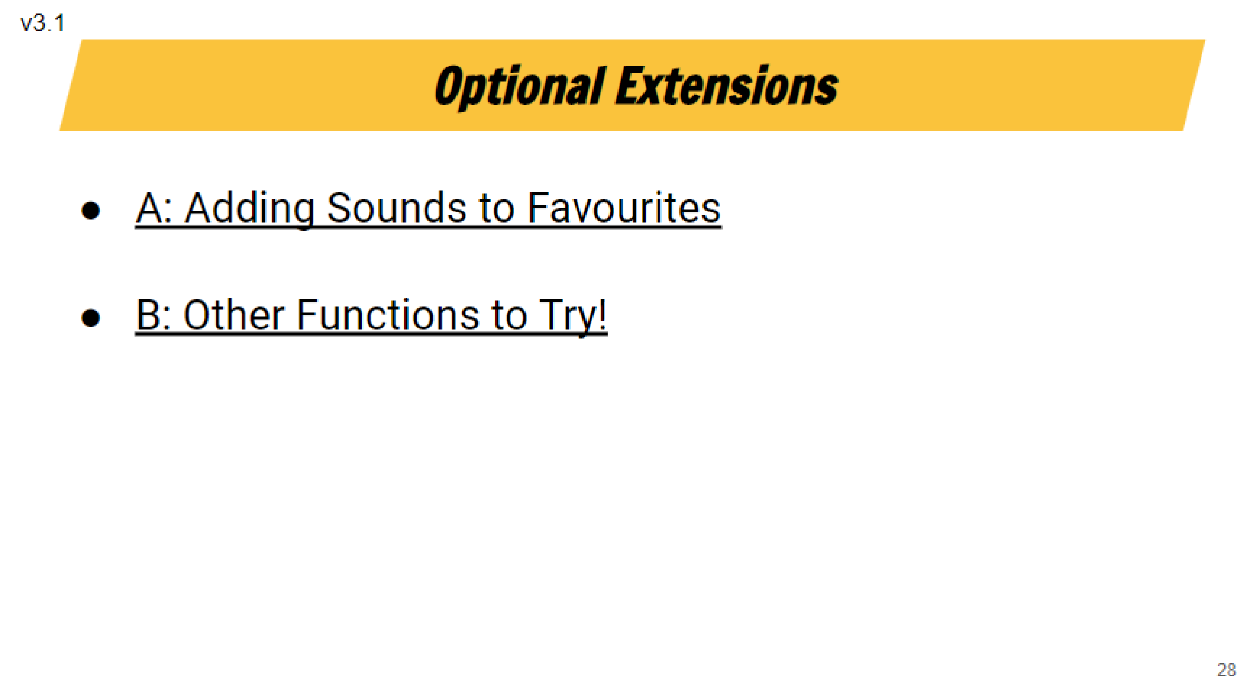
Answer the following questions:

Questions	Answers
How did you choose your song's sound clips?	**Answers will vary**

How do you think you could layer your sound clips to represent layers of injustice?	**Answers will vary**
How do variables help you create more effective code?	**Answers will vary**

EXTENSIONS

Slide 28



v3.1

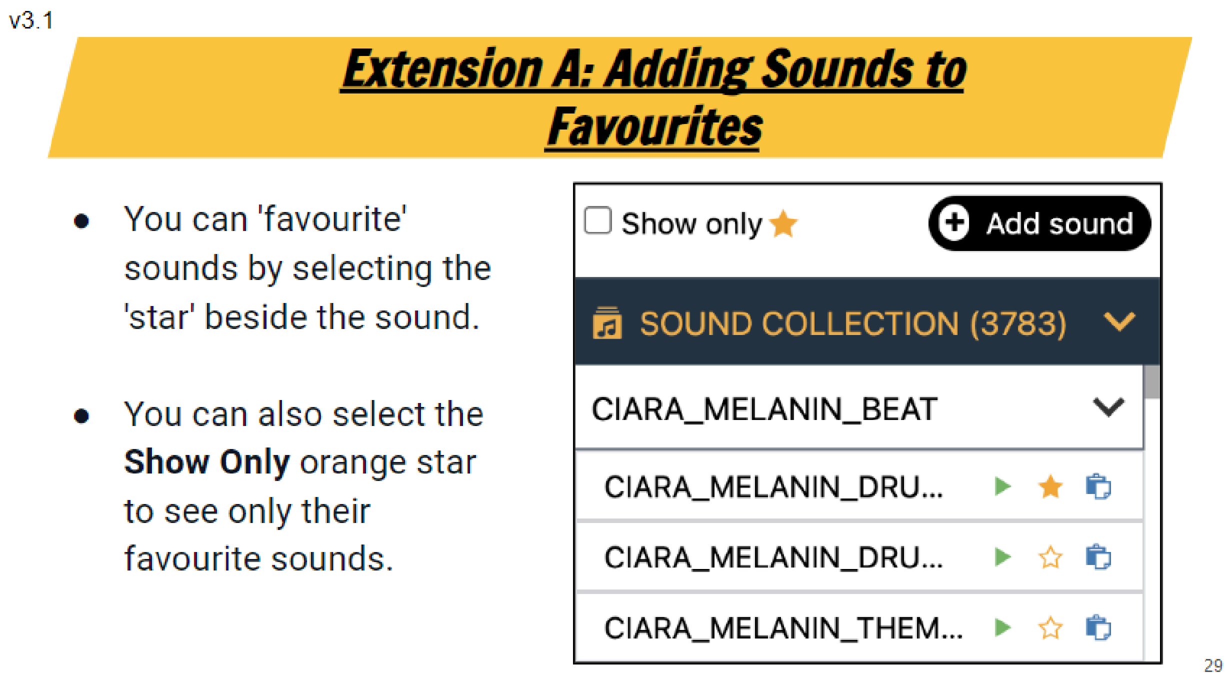
Optional Extensions

- A: Adding Sounds to Favourites
- B: Other Functions to Try!

28

Extension A: Adding Sounds to Favourites

Slide 29



v3.1

Extension A: Adding Sounds to Favourites

- You can 'favourite' sounds by selecting the 'star' beside the sound.
- You can also select the **Show Only** orange star to see only their favourite sounds.

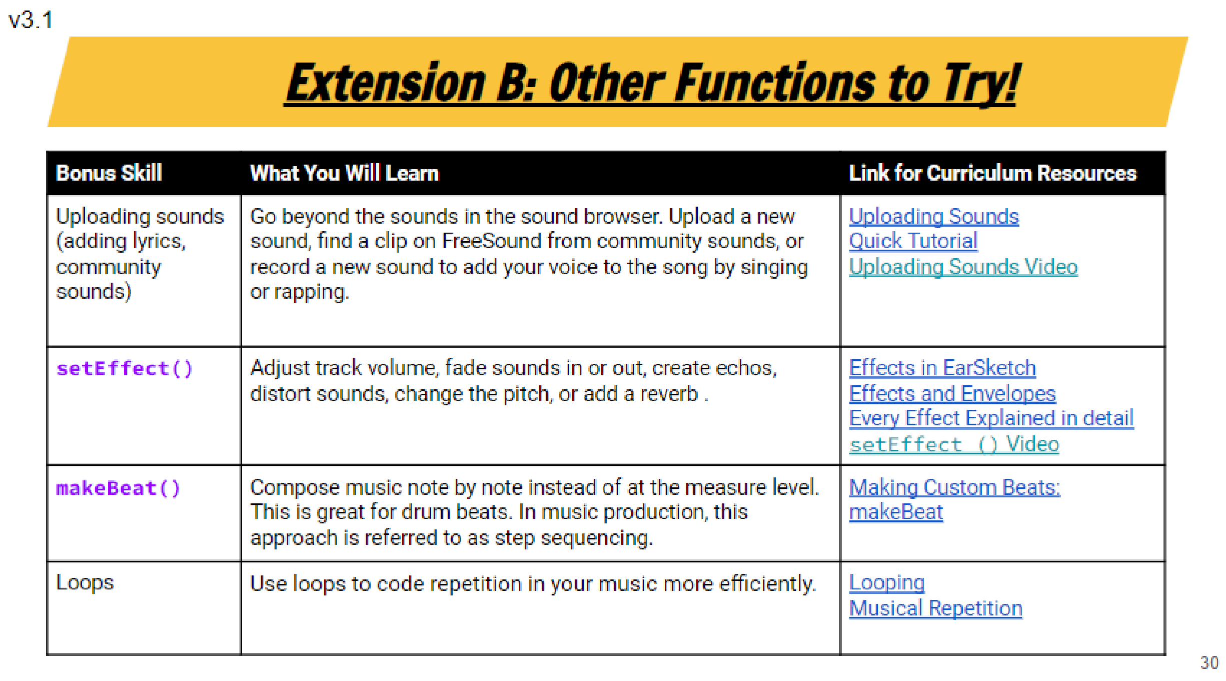
29

Instructor Dialogue

You can can 'favourite' sounds by selecting the 'star' beside the sound.
*Then you can select the **Show Only** orange star to see only your favourite sounds.*

Extension B: Other Functions to Try

Slide 30



v3.1

Extension B: Other Functions to Try!

Bonus Skill	What You Will Learn	Link for Curriculum Resources
Uploading sounds (adding lyrics, community sounds)	Go beyond the sounds in the sound browser. Upload a new sound, find a clip on FreeSound from community sounds, or record a new sound to add your voice to the song by singing or rapping.	Uploading Sounds Quick Tutorial Uploading Sounds Video
`setEffect()`	Adjust track volume, fade sounds in or out, create echos, distort sounds, change the pitch, or add a reverb .	Effects in EarSketch Effects and Envelopes Every Effect Explained in detail `setEffect ()` Video
`makeBeat()`	Compose music note by note instead of at the measure level. This is great for drum beats. In music production, this approach is referred to as step sequencing.	Making Custom Beats: makeBeat
Loops	Use loops to code repetition in your music more efficiently.	Looping Musical Repetition

30

Instructor Dialogue

You can make a great song with just fitMedia and setTempo, but for those of you who enjoy coding or those of you who really get into creating your song , there are lots of additional functions you can explore, check out the API section and the Curriculum section to find documentation and examples.

RESOURCES

Module 4 contains no extra resources.